

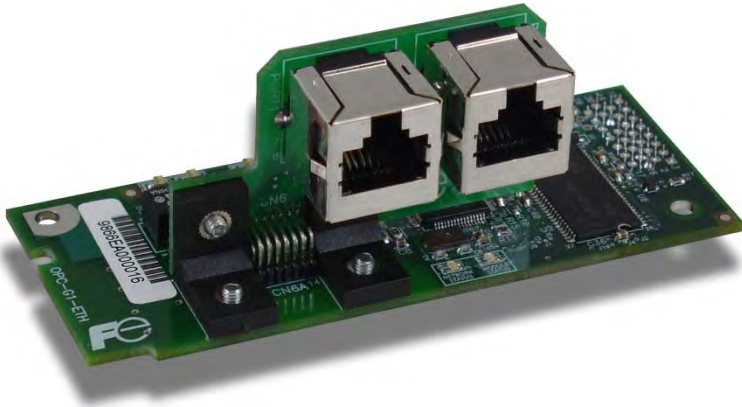


**FRENIC-MEGA**

# **OPC-G1-ETH**

## **Multiprotocol Ethernet Interface**

---



### **⚠ CAUTION**

Thank you for purchasing the OPC-G1-ETH Multiprotocol Ethernet Interface.

- This product is designed to connect the FRENIC-Mega series of inverters to Ethernet communication networks. Please read this instruction manual thoroughly in order to become familiar with the proper interface handling, installation and usage procedures.
- Improper handling may inhibit correct operation or cause premature interface failure.
- Please deliver this instruction manual to the end user of the interface, and retain it in an accessible location.
- For inverter usage instructions, please refer to the applicable FRENIC-Mega inverter instruction manual.



## OPC-G1-ETH Multiprotocol Ethernet Interface Instruction Manual

Part Number 10821

Printed in U.S.A.

©2011 Fuji Electric.

All rights reserved

Fuji Electric reserves the right to make changes and improvements to its products without providing notice.

### Notice to Users

PRODUCTS ARE NOT AUTHORIZED FOR USE AS CRITICAL COMPONENTS IN LIFE-SUPPORT DEVICES OR SYSTEMS. Life-support devices or systems are devices or systems intended to sustain life, and whose failure to perform, when properly used in accordance with instructions for use provided in the labeling and user's manual, can be reasonably expected to result in significant injury.

No complex software or hardware system is perfect. Bugs may always be present in a system of any size. In order to prevent danger to life or property, it is the responsibility of the system designer to incorporate redundant protective mechanisms appropriate to the risk involved.

## Preface

Thank you for purchasing the OPC-G1-ETH Multiprotocol Ethernet Interface. This instruction manual has been prepared to help you connect your FRENIC-Mega inverter to a variety of Ethernet control networks.

This instruction manual does not contain inverter usage instructions. Please refer to this instruction manual in conjunction with the FRENIC-Mega Instruction Manual (INR-SI47-1457-E) in order to become familiar with the proper handling, installation and operation of this product. Improper handling or installation procedures may result in incorrect operation or premature product failure.

Please keep this instruction manual in a safe place.

### Related Publications

Listed below are publications that are recommended for reference in conjunction with this instruction manual.



- RS-485 Communication User's Manual..... (MEH448)
- FRENIC-Mega Instruction Manual..... (INR-SI47-1457-E)

These documents are subject to change without notice. Please be sure to refer to the most recent available versions.

## Safety precautions

Please read this instruction manual thoroughly prior to proceeding with installation, connections, operation, or maintenance and inspection. Additionally, ensure that all aspects of the system are fully understood, and familiarize yourself with all safety information and precautions before operating the inverter.

Safety precautions in this instruction manual are classified into the following two categories:

 <b>WARNING</b>	Failure to heed the information indicated by this symbol may lead to dangerous conditions, possibly resulting in death or serious bodily injuries.
 <b>CAUTION</b>	Failure to heed the information indicated by this symbol may lead to dangerous conditions, possibly resulting in minor or light bodily injuries and/or substantial property damage.

Failure to heed the information contained under the CAUTION title can also result in serious consequences. These safety precautions are of utmost importance and must be observed at all times.

## Installation and wiring

### **WARNING**

- To avoid electrical shock, remove all power from the inverter and wait at least five minutes prior to starting installation. Additionally, confirm that the DC link bus voltage as measured between the P (+) and N (-) terminals is less than 25 VDC.
- Installation should be performed only by qualified personnel.
- To avoid electrical shock, do not operate the inverter with the front cover or wiring cover removed, as accidental contact with exposed high-voltage terminals and internal components may occur.
- To prevent explosions or similar damage, ensure that all cables are properly connected to the correct terminals, and observe all wiring polarity indicators.

### **CAUTION**

- Do not install or operate the interface if it is damaged or has parts missing.
- Prevent conductive items such as screws and metal fragments, or flammable substances such as oil, lint, paper fibers and sawdust from entering the inverter and interface card enclosure.
- Incorrect handling during installation or removal may cause equipment failure.
- Do not subject the cables to scratches, excessive stress, heavy loads or pinching.
- To prevent damage due to electrostatic discharge, always touch a grounded piece of metal prior to touching any equipment.
- Do not stand on or rest heavy objects on the equipment.
- To prevent burns from hot components, do not touch the inverter while power is on, or for some time after power is removed.
- Electrical noise may be emitted from the inverter, motor and wires. Always implement appropriate countermeasures to prevent nearby sensors and devices from malfunctioning due to such noise.

## Operation

### **WARNING**

- To avoid electrical shock, do not open the front cover of the inverter while power is on or while the inverter is running.
- To avoid electrical shock, do not operate switches with wet hands.
- If the inverter's function codes are incorrectly configured, or configured without adequate understanding of the FRENIC-Mega Instruction Manual (INR-SI47-1457-E) and FRENIC-Mega User's Manual (MEH642), the motor may rotate with a torque or at a speed not permitted for the machine. Confirm the settings of all function codes prior to running the inverter.

## Maintenance, inspection, and parts replacement

### WARNING

- To avoid electrical shock, remove all power from the inverter and wait at least five minutes prior to starting inspection. Additionally, confirm that the DC link bus voltage as measured between the P (+) and N (-) terminals is less than 25 VDC.
- Maintenance, inspection, and parts replacement should be performed only by qualified personnel.
- Remove all watches, rings and other metallic objects prior to starting work.
- To avoid electrical shock or other injuries, always use insulated tools.

## Disposal

### CAUTION

- Contact the local or state environmental agency in your area for details on the disposal of electrical components and packaging.

## Other

### WARNING

- Do not attempt to modify the equipment: doing so may cause electrical shock or injuries.
- For clarity purposes, illustrations in this manual may be drawn with covers or safety guards removed. Ensure all covers and safety guards are properly installed prior to starting operation.
- Do not perform hi-pot tests on the equipment.
- Performing a data initialization (function code H03) may reset all inverter function codes to their factory default settings. After performing this operation, remember to reenter any custom function code values prior to starting operation.

## Icons

The following icons are used throughout this manual:



Indicates information which, if not heeded, can result in the product not operating to full efficiency, as well as information concerning incorrect operations and settings which may result in accidents.



Indicates information that can prove handy when performing certain settings or operations.



Indicates a reference to more detailed information.

## – TABLE OF CONTENTS –

<b>1</b>	<b>PRE-OPERATION INSTRUCTIONS.....</b>	<b>8</b>
1.1	Product Overview.....	8
1.2	Unpacking and Product Confirmation .....	9
1.2.1	Shipment Confirmation.....	9
1.2.2	Component Overview.....	10
1.3	LED Indicators.....	12
1.3.1	Network Status LED.....	12
1.3.2	Module Status LED.....	12
1.3.3	Ethernet Link/Activity LEDs.....	12
1.3.4	Ethernet Speed LEDs.....	12
1.4	Environmental Specifications.....	12
<b>2</b>	<b>INSTALLATION .....</b>	<b>13</b>
2.1	Pre-Installation Instructions.....	13
2.2	Installation Procedure .....	13
<b>3</b>	<b>INVERTER FUNCTION CODE SETTINGS .....</b>	<b>15</b>
3.1	Inverter Control-Related Settings.....	15
3.2	Inverter Reaction to Network Timeout Conditions .....	16
<b>4</b>	<b>FUJI FINDER APPLICATION .....</b>	<b>17</b>
4.1	Installation .....	17
4.2	USB Driver Installation .....	20
4.2.1	Windows XP .....	20
4.2.2	Windows 7 .....	23
4.3	Overview .....	24
4.4	Ethernet Tab .....	25
4.5	USB Tab .....	25
4.6	Configuring the IP Address.....	25
<b>5</b>	<b>EMBEDDED WEB SERVER.....</b>	<b>26</b>
5.1	Overview .....	26
5.2	Page Select Tabs.....	27
5.3	Monitor Tab.....	27
5.3.1	Information Window .....	27
5.3.2	Function Code Group Selection List.....	27
5.3.3	Function Code List.....	28
5.3.4	Function Code List Filter .....	29
5.3.5	Radix Selection .....	29
5.4	BACnet Tab.....	30
5.4.1	Information Window .....	30
5.4.2	Device Identifiers .....	30
5.4.3	Submitting Changes.....	31
5.4.4	Reinitialize Prompt.....	31
5.5	Config Tab.....	32
5.5.1	Information Window .....	32
5.5.2	Authentication Configuration .....	32



5.5.3	Timeout Configuration.....	32
5.5.4	Submitting Changes.....	33
5.5.5	Reinitialize Prompt.....	33
<b>5.6</b>	<b>EtherNet/IP Tab .....</b>	<b>34</b>
5.6.1	Information Window .....	34
5.6.2	Device Identification.....	34
5.6.3	Run/Idle Flag Behavior.....	35
5.6.4	Class 1 (I/O) Data Configuration Arrays.....	35
5.6.5	Submitting Changes.....	36
5.6.6	Reinitialize Prompt.....	36
<b>5.7</b>	<b>Modbus Tab .....</b>	<b>37</b>
5.7.1	Information Window .....	37
5.7.2	Supervisory Timer Selection .....	37
5.7.3	Register Remap Configuration .....	38
5.7.4	Submitting Changes.....	39
5.7.5	Reinitialize Prompt.....	39
<b>5.8</b>	<b>Alarm Tab.....</b>	<b>40</b>
5.8.1	Information Window .....	40
5.8.2	Email Configuration.....	41
5.8.3	Alarm Configuration .....	42
5.8.4	Submitting Changes.....	43
<b>5.9</b>	<b>Dashboard Tab .....</b>	<b>44</b>
5.9.1	Information Window .....	44
5.9.2	Virtual Keypad.....	45
5.9.3	Gauge Window Navigation.....	46
5.9.4	Gauge Window Configuration .....	46
5.9.5	Submitting Changes.....	49
<b>5.10</b>	<b>Customizing the Embedded Web Server .....</b>	<b>50</b>
5.10.1	Customization Overview .....	50
5.10.2	XTPro Overview.....	50
5.10.3	XTPro Web Browser-Based Implementation .....	51
5.10.4	XTPro HMI-Based Implementation.....	52
5.10.5	XTPro Supported Commands .....	52
<b>6</b>	<b>FUNCTION CODE NUMBERING AND BEHAVIOR.....</b>	<b>53</b>
6.1	Register Numbers .....	53
6.2	Scanned Function Codes .....	55
6.3	Commonly Used Function Codes.....	55
<b>7</b>	<b>FILE SYSTEM &amp; FIRMWARE .....</b>	<b>57</b>
7.1	Overview .....	57
7.2	Windows Explorer.....	58
7.3	Loading New Application Firmware .....	59
<b>8</b>	<b>PROTOCOL-SPECIFIC INFORMATION .....</b>	<b>60</b>
<b>8.1</b>	<b>Modbus/TCP .....</b>	<b>60</b>
8.1.1	Overview.....	60
8.1.2	Coil & Discrete Input Mappings.....	61
<b>8.2</b>	<b>EtherNet/IP .....</b>	<b>62</b>
8.2.1	Overview.....	62
8.2.2	ODVA AC/DC Drive Profile .....	63
8.2.3	ControlLogix Examples: Setup.....	65



8.2.4	ControlLogix Example: I/O Messaging .....	66
8.2.5	ControlLogix Example: Generic Default I/O Add-On Instruction .....	69
8.2.6	ControlLogix Example: AC/DC Drive Profile Add-On Instruction .....	71
8.2.7	Explicit Messaging Tag Reference .....	73
8.2.8	ControlLogix Explicit Messaging Example: Read a Function Code Block .....	74
8.2.9	ControlLogix Explicit Messaging Example: Read a Single Function Code .....	79
8.2.10	ControlLogix Explicit Messaging Example: Multiple MSG Instructions .....	79
8.2.11	ControlLogix Explicit Messaging Example: Reading and Writing .....	80
<b>8.3</b>	<b>Allen Bradley CSP .....</b>	<b>81</b>
8.3.1	Overview .....	81
8.3.2	Tag Reference .....	81
8.3.3	SLC-5/05 Example: Read a Register Block .....	83
8.3.4	SLC-5/05 Example: Read a Single Register .....	86
8.3.5	SLC-5/05 Example: Multiple MSG Instructions .....	87
8.3.6	SLC-5/05 Example: Reading and Writing .....	88
<b>8.4</b>	<b>BACnet/IP .....</b>	<b>89</b>
8.4.1	Protocol Implementation Conformance Statement .....	89
8.4.2	Supported Objects .....	92
8.4.3	Supported Object Details .....	94
<b>9</b>	<b>TROUBLESHOOTING .....</b>	<b>95</b>



## **1 PRE-OPERATION INSTRUCTIONS**

### **1.1 Product Overview**

The OPC-G1-ETH Ethernet multiprotocol communication interface allows information to be transferred seamlessly between a FRENIC-Mega inverter and several different Ethernet-based fieldbus networks with minimal configuration requirements. The interface installs directly onto the inverter, and presents two RJ-45 jacks with an embedded 10/100BaseT Ethernet switch for connection to the Ethernet network. In addition to the supported fieldbus protocols, the interface also hosts a fully-customizable embedded web server, which provides access to inverter information via a standard web browser for remote monitoring, configuration and control.

Before using the interface, please familiarize yourself with the product and be sure to thoroughly read the instructions and precautions contained in this manual. In addition, please make sure that this instruction manual is delivered to the end user of the interface, and keep this instruction manual in a safe place for future reference or unit inspection.

Note that different interface firmware versions may provide varying levels of support for the various protocols. When using this manual, therefore, always keep in mind the release date of the firmware version running on your interface as it must match this manual's respective release date in order for all documented aspects to apply.

The primary features of the OPC-G1-ETH are as follows:

#### **Supported Protocols**

The interface currently provides server support for the following fieldbus protocols:

- Modbus/TCP Server
- EtherNet/IP Server
- Allen Bradley CSP Server (also known as "PCCC" and "AB Ethernet")
- BACnet/IP Server

#### **Ethernet Ports**

IEEE 802.3 10/100BaseT Ethernet compliant. Shielded RJ45 connectors accept standard CAT5-type 8-conductor unshielded twisted-pair (UTP) patch cables. MDI/MDI-X auto-crossover allows the use of any combination of straight-through and cross-over Ethernet cables. Supports multiple simultaneous protocols.

#### **USB Port**

USB 2.0 port with mini-B connector provides composite USB device functionality. USB connection allows for product identification and firmware updating. Additionally, the OPC-G1-ETH enumerates as a standard USB mass storage device ("flash drive") for configuration file copying and web page customization.

#### **Custom Embedded Web Server**

Open XML-based socket data transfer allows end users to create their own custom web server content and load it onto the unit's internal file system via USB. The factory-default web server content provides configuration and real-time inverter function code monitoring & control via standard web browsers such as Microsoft Internet Explorer and Mozilla Firefox. The default web server requires the latest version of Adobe Flash Player browser plug-in. Refer to section 5.

#### **XML Configuration File Upload/Download**

All interface configuration files are stored in the unit's internal filesystem in XML format. These files can be transferred to/from a PC via USB, which provides the capability for PC-based file backup and easy configuration copying to multiple units. Configuration files can also be viewed and edited via standard text editors, XML editors and web browsers. Refer to section 7.1.

#### **Email-Based Alarm Notifications**

Up to 20 configurable alarm conditions can be programmed into the interface. Value, logical comparison and time-based conditions can be provided for the interface to autonomously monitor any available inverter register. When an alarm condition is triggered, a notification email can be sent to up to four destination email addresses. Refer to section 5.8.

## **Network Timeout Action**

A configurable network timeout action can be programmed that allows inverter function codes to have their own unique "fail-safe" conditions in the event of a network interruption. Refer to section 5.5.3.

## **Field-Upgradeable**

As new firmware becomes available, the interface can be upgraded in the field by the end-user via USB. Refer to section 7.3 for more information.

## **EtherNet/IP Data Access Options**

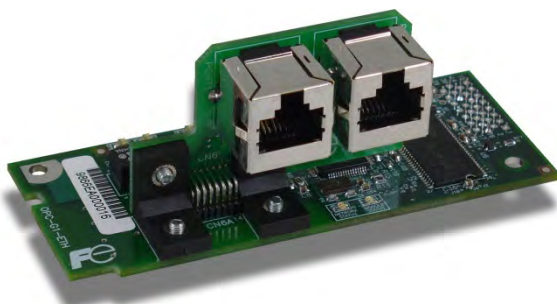
The EtherNet/IP protocol provides access to inverter data via explicit messaging, user-defined I/O assembly instances, and the ODVA AC/DC drive profile. Refer to section 8.2 for more information.

## **1.2 Unpacking and Product Confirmation**

### ***1.2.1 Shipment Confirmation***

Check the enclosed items. Confirm that the correct quantity of each item was received, and that no damage occurred during shipment.

- OPC-G1-ETH interface board (see Figure 1).
- Two M3 x 6mm mounting screws (see Figure 2).



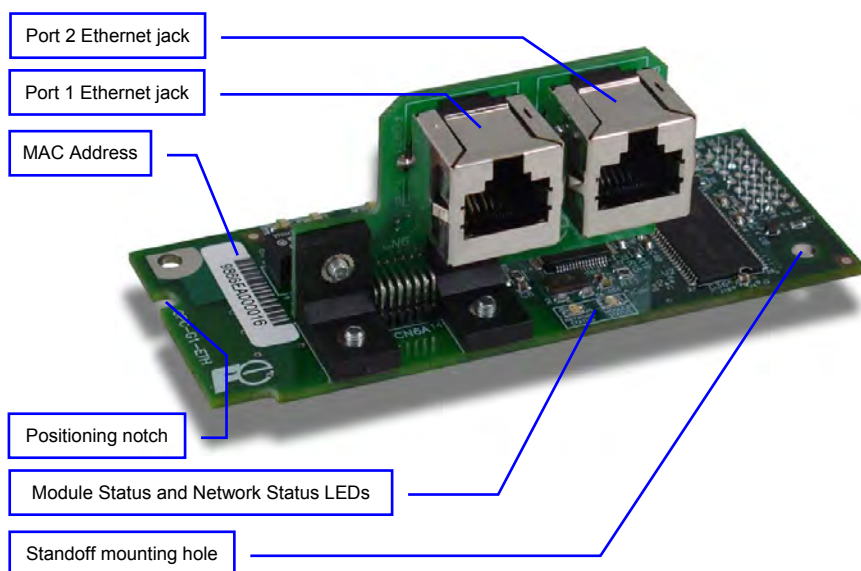
**Figure 1: OPC-G1-ETH Interface Board**



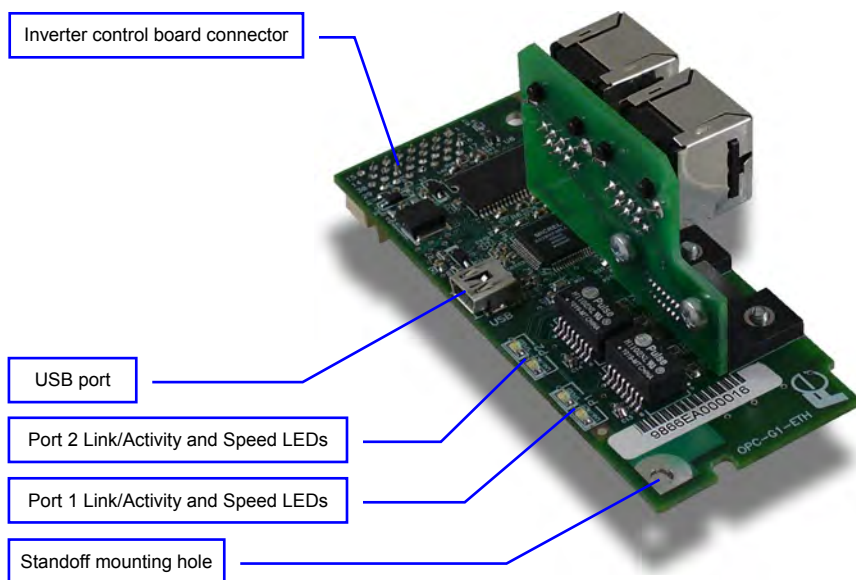
**Figure 2: Qty. 2 M3 x 6mm Mounting Screws**

### 1.2.2 Component Overview

Figure 3 and Figure 4 provide an overview of the important interface card components.



**Figure 3: OPC-G1-ETH Component Overview (Front Side)**



**Figure 4: OPC-G1-ETH Component Overview (Back Side)**

### **Port 1 and Port 2 Ethernet Jacks**

Either jack can freely be used star topology networks (with external switch). In linear topologies, a series of OPC-G1-ETH cards can be connected together by daisy-chaining one of the ports to the next inverter in line.

### **MAC Address**

Barcode sticker that indicates the card's unique Ethernet MAC Address. The MAC Address can be used to identify specific cards discovered with the Fuji Finder application (refer to section 4.3).

### **Positioning Notch**

Aligns with the positioning key on the inverter chassis to ensure that the interface card is installed into the correct communication port (refer to section 2.2).

### **Module Status and Network Status LEDs**

These LEDs indicate the current status of the interface card and protocols in use. Refer to section 1.3.

### **Standoff Mounting Holes**

The provided M3 x 6mm screw are inserted here to secure the card to the standoffs located on the inverter's control board. Refer to section 2.2.

### **Inverter Control Board Connector**

Attaches to the "A-port" on the inverter's control board.

### **USB Port**

USB 2.0 port with mini-B connector. Used to access the card via the Fuji Finder program (refer to section 4.5) and as a USB flash drive (refer to section 7.1).

### **Ethernet Link/Activity and Speed LEDs**

One set of LEDs are provided for each Ethernet port ("P1" for Port 1 and "P2" for Port 2). These LEDs provide insight into the Ethernet network's status and activity. Refer to section 1.3.

## 1.3 LED Indicators

### 1.3.1 Network Status LED

- Conforms to the prescribed “network status LED” behavior as dictated in the EtherNet/IP specification, Volume 2, Chapter 9.

### 1.3.2 Module Status LED

- Conforms to the prescribed “module status LED” behavior as dictated in the EtherNet/IP specification, Volume 2, Chapter 9.
- Contact technical support if a blinking red error code is observed.

### 1.3.3 Ethernet Link/Activity LEDs

- The green “LNK/ACT” LEDs (one for each Ethernet port) are lit whenever a viable Ethernet network is connected, and blink when network packets are sent or received on the associated port.

### 1.3.4 Ethernet Speed LEDs

- The amber “SPEED” LEDs (one for each Ethernet port) are lit if the link speed is 100 Mbps, and are off if the link speed is 10 Mbps.

## 1.4 Environmental Specifications

The interface’s environmental specifications are detailed in Table 1.

**Table 1: Environmental Specifications**

Item	Specification
Operating Environment	Indoors, less than 1000m above sea level, do not expose to direct sunlight or corrosive / explosive gasses
Operating Temperature	-10 ~ +50°C (+14 ~ +122°F)
Storage Temperature	-40 ~ +85°C (-40 ~ +185°F)
Relative Humidity	20% ~ 90% (without condensation)
Vibration	5.9m/s <sup>2</sup> (0.6G) or less (10 ~ 55Hz)
Cooling Method	Self-cooled
Communication Speed	10/100BaseT auto sensing



This device is lead-free / RoHS-compliant. *Lead Free*

## 2 INSTALLATION

### 2.1 Pre-Installation Instructions


#### WARNING

- To avoid electrical shock, remove all power from the inverter and wait at least five minutes prior to starting installation. Additionally, confirm that the DC link bus voltage as measured between the P (+) and N (-) terminals is less than 25 VDC.
- Installation should be performed only by qualified personnel.
- To avoid electrical shock, do not operate the inverter with the front cover or wiring cover removed, as accidental contact with exposed high-voltage terminals and internal components may occur.
- To prevent explosions or similar damage, ensure that all cables are properly connected to the correct terminals, and observe all wiring polarity indicators.

### 2.2 Installation Procedure



Before installing the interface card, perform all wiring for the main circuit terminals and control circuit terminals.

1. Remove the front cover from the inverter to expose the control printed circuit board (control PCB). As shown in Figure 5, there are three option connection ports (A-port, B-port and C-port). The OPC-G1-ETH card is mechanically keyed for, and can only be installed into, the A-port (bottom-most) position.
-  To remove the front cover, refer to the FRENIC-MEGA Instruction Manual, Chapter 2, Section 2.3. The keypad enclosure must also be opened on 30kW and larger inverters.

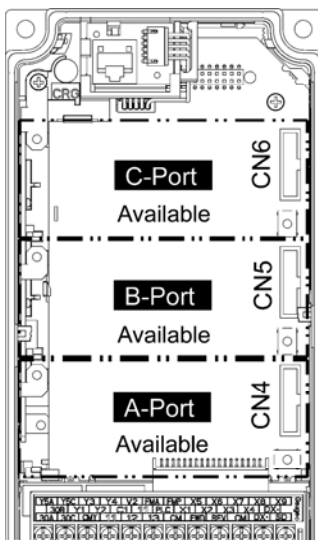
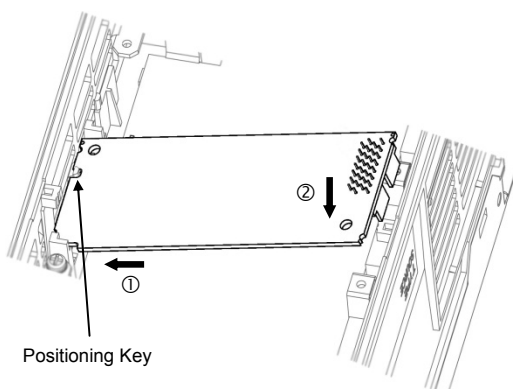


Figure 5: Option Port Locations on 0.4 kW Inverter

2. Rest the left-hand side of the interface card on the control PCB's A-port mounting support. Align the positioning notch on the interface card with the A-port positioning key, and then slide the interface card to the left to engage the key into the notch. Refer to step ① in Figure 6.
3. Rotate the right-hand side of the interface card downward to engage connector CN1 (on the back of the interface card) into the A-port connector (CN4) on the inverter's control PCB. Ensure that the connectors are fully engaged. Refer to step ② in Figure 6.

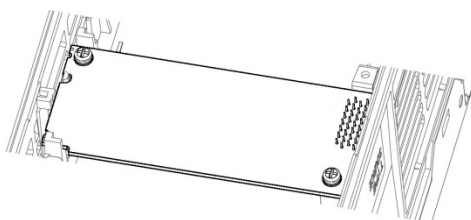


To ensure that the interface card is fully aligned and seated into the communication port, be sure to perform steps ① and ② in the proper order. Failure to do so may lead to insufficient connector insertion and result in contact failure.



**Figure 6: Mounting the Interface Card**

4. Install the two M3 x 6mm screws included with the kit into the standoff mounting holes located at the upper-left and lower-right hand corners of the interface card. Tighten the screws to secure the interface card to the control board PCB. Refer to Figure 7.



**Figure 7: Interface Card Mounting Completed**

5. Connect the network cables as necessary. Insert the Ethernet cables into the Ethernet jacks, making sure that they are fully seated. Ensure that the cables are routed in such a way that they will not be pinched and are not located near any power-carrying wiring, such as the inverter's input power or motor wires.
6. Reinstall all covers removed in step 1. Take a moment to confirm that the Ethernet cables are not being pinched and are not routed near any power-carrying wiring.



For reinstallation instructions, refer to the FRENIC-MEGA Instruction Manual, Chapter 2, Section 2.3. The keypad enclosure must also be closed on 30kW and larger inverters.

### 3 INVERTER FUNCTION CODE SETTINGS

Depending on the desired operation of the overall application, the inverter function codes listed in Table 2 are important for proper operation of the end-to-end communication system. Although there may be many other function codes that will require configuration for your specific application, it is important to understand the manner in which the following function codes will impact successful control of the inverter.



For further details regarding these function codes, please refer to the FRENIC-Mega Instruction Manual (INR-SI47-1457-E), Chapter 5 "FUNCTION CODES", FRENIC-Mega User's Manual, "y codes: Link Functions", and RS-485 Communication User's Manual (MEH448), Chapter 5, Section 5.2 "Data Formats."

**Table 2: Function Code Settings Overview**

Code	Name	Setting Range	Default Value
Y98	Bus Link Function (Mode Selection)	0 to 3	0

#### 3.1 Inverter Control-Related Settings

The following function codes relate to whether or not the inverter is to be controlled (command word and/or frequency command) from the network, or whether the inverter will be locally-controlled (and therefore only monitored and/or configured via the network.)

##### **Bus Link Function (Mode Selection) (y98)**

If the inverter is to be controlled from the network, then set the value of y98 to 3 (fieldbus option). A setting of 3 for y98 may also be appropriate even if H30 is configured for an alternate (local) control scheme.

When the inverter is controlled from the network, a selection of reference commands (S?? function codes) are available for controlling the inverter's speed. If multiple reference commands are being modified from the network, then the interface card invokes a hierarchy to determine which reference is to be passed to the inverter as its main reference command.

The S?? function code hierarchy is as follows, listed from highest to lowest priority:

- S01 (frequency reference / per-unit)
- S05 (frequency reference / Hz)
- S19 (speed command)
- S02 (torque command)
- S03 (torque current command)
- S13 (PID command)

The highest-priority S?? function code with a non-zero value will be used as the inverter's main reference command.



### 3.2 Inverter Reaction to Network Timeout Conditions

Function codes o27 and o28 specify the inverter's reaction when a network timeout occurs. Table 3 lists the settings for o27 and o28.

**Table 3: Inverter Reaction to Network Timeout Conditions (Function Codes o27 and o28)**

o27 Value	o28 Value	Inverter reaction when a timeout occurs	Remarks
0, 4 to 9	---	Immediately coast to a stop and trip $\overline{Er5}$ .	
1	0.0s to 60.0s	After the time specified by o28, coast to a stop and trip $\overline{Er5}$ .	
2	0.0s to 60.0s	If the communications link is restored within the time specified by o28, ignore the communications error. After the timeout, coast to a stop and trip $\overline{Er5}$ .	
3, 13 to 15	---	Maintain present operation, ignoring the communications error (no $\overline{Er5}$ trip).	
10	---	Immediately decelerate to a stop. Trip $\overline{Er5}$ after stopping.	Inverter function code F08 specifies the deceleration time.
11	0.0s to 60.0s	After the time specified by o28, decelerate to a stop. Trip $\overline{Er5}$ after stopping.	Same as above.
12	0.0s to 60.0s	If the communications link is restored within the time specified by o28, ignore the communications error. After the timeout, decelerate to a stop and trip $\overline{Er5}$ .	Same as above.
13	---	Immediately turns the run command OFF (no $\overline{Er5}$ trip).	
14	---	Issues a command to run the motor in a forward direction (no $\overline{Er5}$ trip).	Forward rotation must be enabled.
15	---	Issues a command to run the motor in a reverse direction (no $\overline{Er5}$ trip).	Reverse rotation must be enabled.

## 4 FUJI FINDER APPLICATION

The Fuji Finder application is a Microsoft Windows® -based PC program which provides several configuration and maintenance utilities for the OPC-G1-ETH, such as Ethernet-based IP address configuration and USB-based firmware updating. The Fuji finder installation files are located on the CD-ROM included with the OPC-G1-ETH kit.

### 4.1 Installation

The Fuji Finder setup will install all required files and USB device drivers. **Note that the Fuji Finder program should be installed before connecting any OPC-G1-ETH devices to the computer's USB port, as the program contains product-specific USB drivers that must be installed on the computer prior to initial connection of the target device.**

#### Launch the install executable

To start the installation of the Fuji Finder application, run the "SETUP" executable located on the CD-ROM included with the OPC-G1-ETH kit.

#### Review the installation message (Figure 8) and click "Next"



**Figure 8: Installer Welcome Screen**

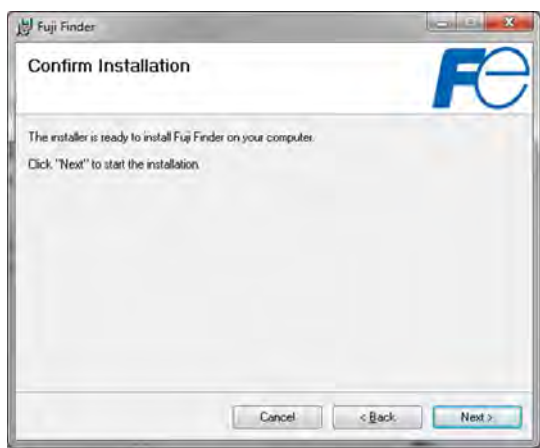
### **Select the install folder**

Select the folder where you want the Fuji Finder to be installed (Figure 9).



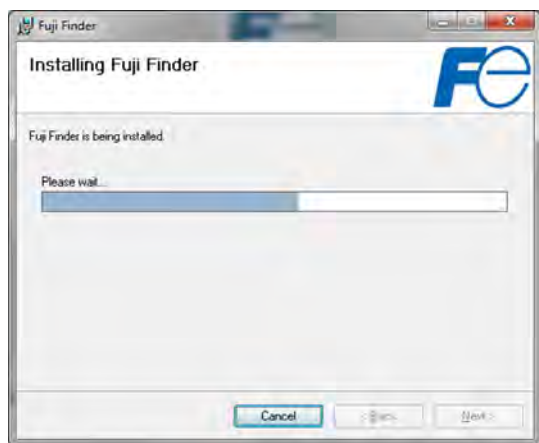
**Figure 9: Installation Folder Selection Screen**

### **Confirm and click "Next" to start installation (Figure 10)**



**Figure 10: Installation Confirmation Screen**

**Wait while the configuration utility is being installed (Figure 11)**

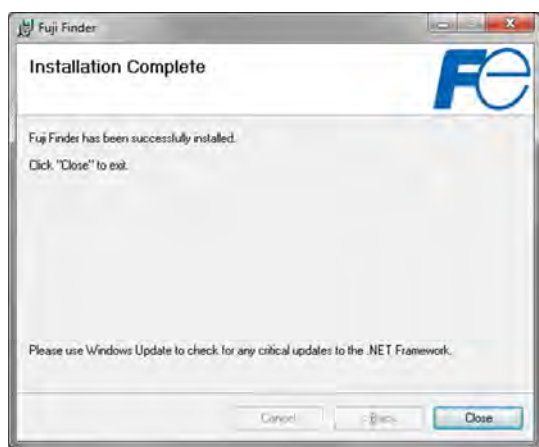


**Figure 11: Installation Progress Screen**

If you are prompted by the operating system during this stage that it can't verify the publisher of this driver software, choose **"Install this driver software anyway"**.

**Installation complete**

Click **"Close"** to exit the installer (Figure 12).



**Figure 12: Installation Complete Screen**

## 4.2 USB Driver Installation

By using a USB mini-B cable, an OPC-G1-ETH card can be connected to the PC, powered, and updated. This section explains setting up the PC to work with an OPC-G1-ETH.

After the Fuji Finder application is installed, the PC will be able to automatically install the appropriate USB driver when an OPC-G1-ETH card is connected via USB. *Note that the Fuji Finder application must be installed prior to connecting an OPC-G1-ETH card to the PC: the PC will not be able to automatically install the USB driver until the Fuji Finder installation is complete.*

The following sections will provide an overview of the USB driver installation procedure for Windows XP and Windows 7. Other versions of Windows may have slightly different procedures.

### 4.2.1 Windows XP

If this is the first time connecting the card to your computer, the operating system will prompt you to install the card's USB driver.

#### Found New Hardware Wizard

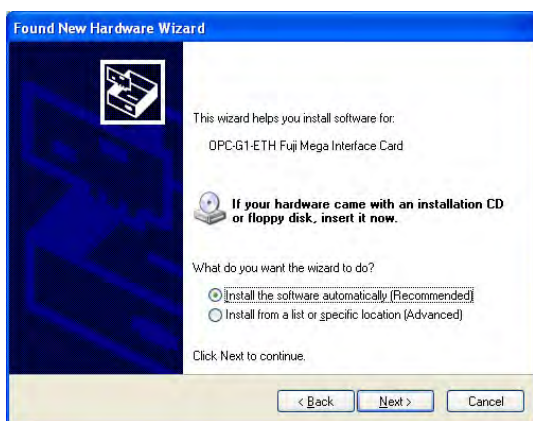
Select **"No, not this time"** when prompted for Windows to connect to Windows Update (Figure 13).



Figure 13: Found New Hardware Screen

## Select Recommended Install

Accept the default action (Figure 14).



**Figure 14: Select Install Type Screen**

## The Wizard Will Search for the Appropriate Driver (Figure 15)



**Figure 15: Searching for Driver**

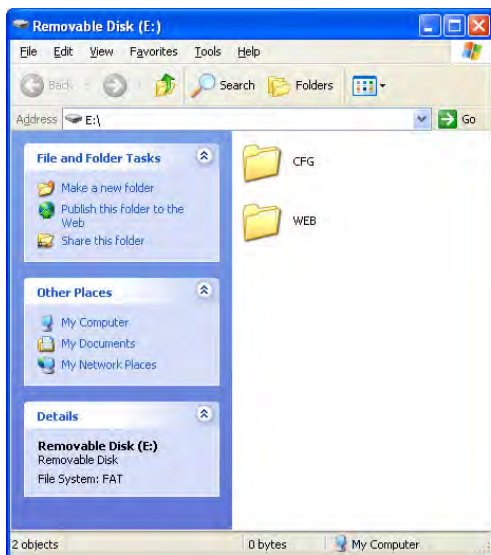
## Device Installation Complete

The device driver has been successfully installed (Figure 16). Click **Finish** to close the wizard.



**Figure 16: USB Driver Installation Complete Screen**

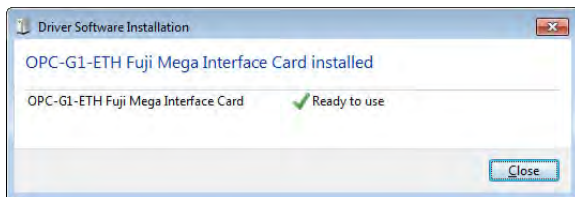
Because the OPC-G1-ETH interface card is a USB composite device (which is to say that it presents multiple virtual representations to the computer), after the vendor-specific USB driver has been installed as detailed above, Windows will then additionally identify the device as a USB mass storage device (also known as a “flash drive” or “removable disk”), and automatically install the default Windows USB mass storage device drivers. Once this is completed, the operating system may automatically pop up a Windows Explorer window showing the file contents of the OPC-G1-ETH's on-board filesystem (Figure 17). This window can be closed at this time if desired.



**Figure 17: Windows Explorer Removable Disk View**

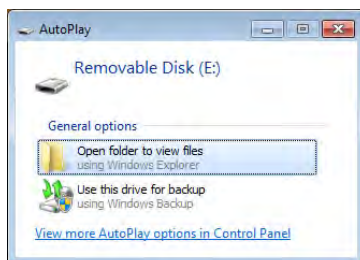
#### 4.2.2 Windows 7

Enhancements in Windows 7 automate the installation of USB drivers to such a degree that no user interaction should be required. When the OPC-G1-ETH card is initially connected to the computer, the operating system should automatically locate and install the appropriate USB driver. Once completed, the operating system may display a dialog box similar to that shown in Figure 18.



**Figure 18: Windows 7 USB Driver Successful Installation**

Because the OPC-G1-ETH interface card is a USB composite device (which is to say that it presents multiple virtual representations to the computer), after the vendor-specific USB driver has been installed as detailed above, Windows will then additionally identify the device as a USB mass storage device (also known as a “flash drive” or “removable disk”), and automatically install the default Windows USB mass storage device drivers. Once this is completed, the operating system may automatically pop up an AutoPlay window showing the options available for the removable disk (Figure 19). This window can be closed at this time if desired.



**Figure 19: Windows 7 AutoPlay**



### 4.3 Overview

The “Fuji Finder” application is a simple Windows PC program, which when executed discovers all OPC-G1-ETH cards on the current Ethernet subnet, regardless of whether or not their network parameters are currently compatible with the subnet upon which they reside. The utility is also used to update firmware. These functions are accessed via two tabs available on the Finder application main program window (refer to Figure 20 and Figure 21.)

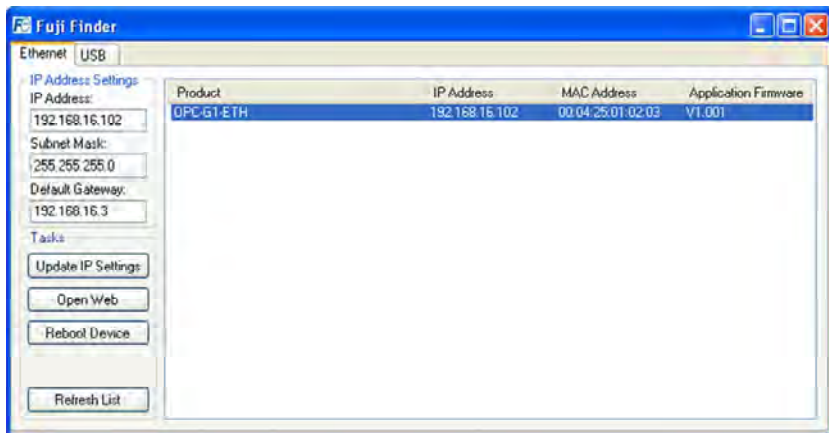


Figure 20: Fuji Finder Ethernet tab

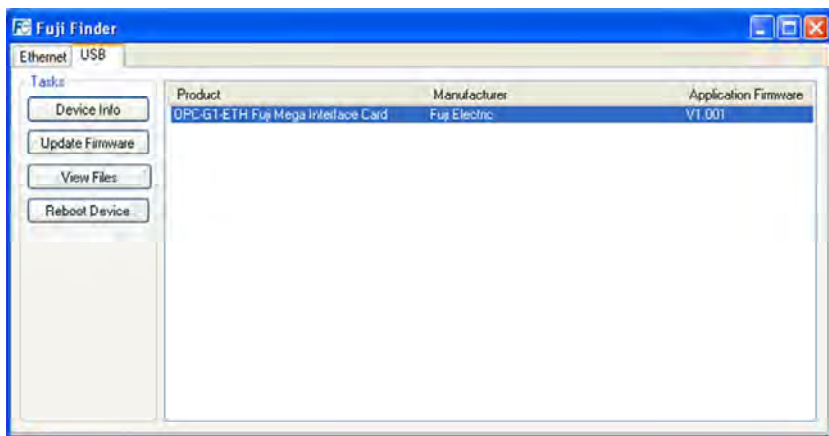


Figure 21: Fuji Finder USB tab

---

## 4.4 Ethernet Tab

---

All devices discovered on the current Ethernet subnet can be organized in ascending or descending order by clicking on the desired sort header (Product, IP Address, MAC Address or Application Firmware). The buttons on the left side of the window perform the following actions:

**Update IP Settings:** Allows configuration of the static IP parameters. Refer to section 4.6 for more information.

**Open Web:** Opens a web browser page of the selected device. Refer to section 5.

**Reboot Device:** Reboots the currently-selected interface card. The Finder application will then automatically rescan the network.

**Refresh List:** Causes the Finder application to rescan the network.

Note that in order for the Finder application to discover devices on the Ethernet subnet, certain UDP traffic must be allowed in and out of the computer, and firewall applications (such as Windows Firewall) are often configured to block such traffic by default. If the Finder is unable to discover any devices on the current subnet, be sure to check the computer's firewall settings during troubleshooting, and add an exception to the firewall configuration if necessary. The Finder application uses UDP port 4334.

---

## 4.5 USB Tab

---

All devices connected to the computer via USB can be organized in ascending or descending order by clicking on the desired sort header (Product, Manufacturer or Application Firmware). The buttons on the left side of the window perform the following actions:

**Device Info:** Provides general device info.

**Update Firmware:** Update the firmware. Refer to section 7.3.

**View Files:** Provides access to the on-board file system with Windows Explorer. Refer to section 7.1.

**Reboot Device:** Reboots the interface card. The Finder will automatically detect the card once it has completed rebooting.

---

## 4.6 Configuring the IP Address

---

Before you can access the interface card from your web browser or begin using it as a part of your automation network, you must assign it an IP address that is appropriate for the subnet on which the card will reside. The interface card comes from the factory configured with a static IP address of **192.168.16.102**. Modifying the IP address can be accomplished with the Finder application via the following procedure:

1. Connect the interface card to the same Ethernet subnet on which the computer resides and apply power to the inverter in which it is installed.
2. Start the Fuji Finder application.
3. The Finder application will scan the Ethernet network for Fuji devices and then display each device's information on the "Ethernet" tab. Refer to Figure 20. If multiple devices are discovered, identify the targeted device via its unique **MAC address** (printed on a barcode label on the left-hand side of the interface card).
4. To change the IP address, select the device in the list of detected devices.
5. Configure the **IP Address Settings** with the desired **IP Address**, **Subnet Mask** and **Default Gateway** in the appropriate boxes. Click the **Update IP Settings** button to apply the changes.
6. A popup dialog box will prompt you to reboot. Click **Yes**. The dialog box will automatically close when the device is done rebooting.
7. The Finder application will automatically rescan the network. Confirm that the new IP address has been accepted by the device.

## 5 EMBEDDED WEB SERVER

### 5.1 Overview

The interface contains an embedded web server (also known as an HTTP server), which allows users to access the inverter's internal data in a graphical manner with web browsers such as Microsoft Internet Explorer or Mozilla Firefox. In this way, the inverter can be monitored, configured and controlled from across the room or from across the globe.

In order to view the interface's factory-default web page, the free Adobe (formerly Macromedia) Flash Player browser plug-in is required. If the plug-in is not already installed on your computer, then your browser will automatically be redirected to the appropriate Adobe download web site when you initially attempt to access the interface's web page. Alternatively, the plug-in can be downloaded directly from Adobe website. Always ensure that you have the latest version of the Flash Player installed: if some aspect of the web page does not appear to be displayed properly, installing the latest Flash Player update usually resolves the problem.

To access an interface's embedded web server, either use the finder application (refer to section 4) and select the "Open Web" button when the target unit is highlighted, or just directly enter the target unit's IP address into the address (URL) field of your web browser. Refer to Figure 22 for a representative screenshot of the web server interface.

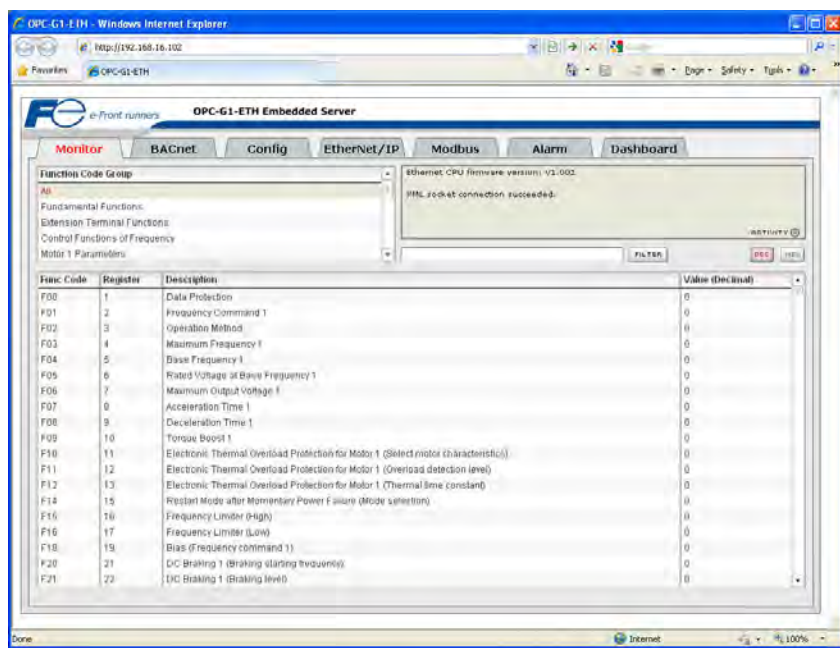


Figure 22: Embedded Web Server

In order to access the web server and view the function code values, destination TCP ports 80 and 2000 must be accessible from the client computer. If an "XML socket connection failed" error message is displayed in the information window, and no function code values are shown, this is typically indicative of port 2000 being blocked by a firewall or Ethernet router situated between the client computer and the interface card.

## 5.2 Page Select Tabs

The web interface is subdivided into several different “tabs” of associated information, much the same as how folders in a filing cabinet are arranged. Refer to Figure 23. To change tabs, just click on the tab you wish to view. The title of the currently-selected tab is red. Note that because different protocols are supported by the interface with different firmware images, not all tabs may be accessible with the firmware image currently loaded. The titles of tabs that are not accessible are grayed-out, and clicking them has no effect.



Figure 23: Page Select Tabs

## 5.3 Monitor Tab

### 5.3.1 Information Window

Figure 24 shows the Information Window, which is located in the upper-right hand corner of the monitor tab. This window displays various informational messages regarding the status of the interface card or web browser session. There is also an “activity” indicator located in the lower-right hand corner of the Information Window, which blinks periodically to show the status of data communication between the web browser and the interface card. If you do not observe the activity indicator blink at all for several seconds or more, it is possible that the web browser may have lost contact to the web server due to an inverter power cycle or a network problem: to reestablish communications, select “refresh” on your web browser.

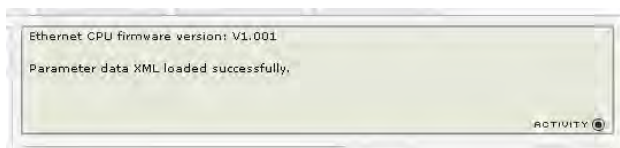


Figure 24: Monitor Tab Information Window

### 5.3.2 Function Code Group Selection List

The Function Code Group Selection List is located in the upper-left hand corner of the Monitor Tab. Refer to Figure 25. Individual groups can be selected by clicking on the group name. Multiple groups may also be selected by holding down the CTRL key while clicking on the group names, or a range of groups can be selected by first selecting the starting group, and then holding down the SHIFT key while selecting the last group in the range. When a function code group is selected, the function codes contained in that group are displayed in the Function Code List (refer to section 5.3.3). The following function code groups are available:

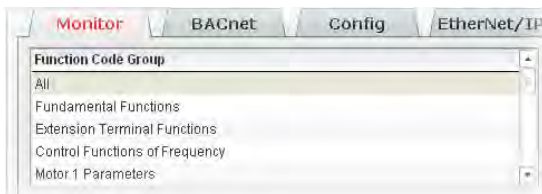


Figure 25: Function Code Group Selection List

**All:** All function codes/registers are available.

**Fundamental Functions:** F function codes are available.

**Extension Terminal Functions:** E function codes are available.

**Control Functions of Frequency:** C function codes are available.

**Motor 1 Parameters:** P function codes are available.

**Motor 2 Parameters:** A function codes are available.

**Motor 3 Parameters:** b function codes are available.

**Motor 4 Parameters:** r function codes are available.

**High Performance Functions:** H function codes are available.

**Application Functions 1:** J function codes are available.

**Application Functions 2:** d function codes are available.

**Link Functions:** y function codes are available.

**Command Data:** S function codes are available.

**Monitor Data 1:** M function codes are available.

**Monitor Data 2:** W function codes are available.

**Alarm Data 1:** X function codes are available.

**Alarm Data 2:** Z function codes are available.

**Operational Functions:** o function codes are available.

### 5.3.3 Function Code List

The bottom half of the Monitor tab contains the function code list (refer to Figure 26). The function codes that are displayed in the list at any given time depend on the function code groups that are currently selected (refer to section 0), as well as whether or not any filters have been applied (refer to section 5.3.4).

The first column of the Function Code List shows the inverter function code designation that is normally used when accessing a given function code via the inverter's keypad. Note that this column is for user convenience and inverter user's manual cross-reference.

The second column of the Function Code List shows the register number for the corresponding function code. Certain protocols require the use of a register number to access the function code (refer to section 6). The third column contains the function code descriptions, which are used by the filter function. The last column performs two functions: it displays the current value of the function code, and (for writable function codes) also allows changing the function code's value by clicking on the number in the value column and entering the new value.

Func. Code	Register	Description	Value (Decimal)
F00	1	Data Protection	0
F01	2	Frequency Command 1	0
F02	3	Operation Method	0
F03	4	Maximum Frequency 1	600
F04	5	Base Frequency 1	600
F05	6	Rated Voltage at Base Frequency 1	230
F06	7	Maximum Output Voltage 1	230
F07	8	Acceleration Time 1	500
F08	9	Deceleration Time 1	500
F09	10	Torque Boost 1	0
F10	11	Electronic Thermal Overload Protection for Motor 1 (Select motor characteristics)	1
F11	12	Electronic Thermal Overload Protection for Motor 1 (Overload detection level)	300
F12	13	Electronic Thermal Overload Protection for Motor 1 (Thermal time constant)	50
F14	15	Restart Mode after Momentary Power Failure (Mode selection)	0
F15	16	Frequency Limiter (High)	700
F16	17	Frequency Limiter (Low)	0
F18	19	Bias (Frequency command 1)	0
F20	21	DC Braking 1 (Braking starting frequency)	0
F21	22	DC Braking 1 (Braking level)	0

**Figure 26: Function Code List**

Some items to keep in mind when interacting with the Function Code List are:

- When entering new function code values, be sure that the number being entered is appropriate for the currently-selected radix (refer to section 5.3.5): for example, an entered value of "1000" in hexadecimal is equal to 4096 in decimal.
- If desired, the column widths can be changed by dragging the vertical bars that separate the header row's cells to a different position.
- If you begin changing a function code value and then decide to abandon the change, pressing the ESC key on your keyboard will abandon the change and redisplay the current function code value.
- When editing a function code value, clicking someplace off the entry cell is equivalent to hitting the ENTER key.

### 5.3.4 Function Code List Filter

A filter function provides Function Code List search capabilities. To use the filter function, simply type a word or portion of a word into the filter entry box and then click the "filter" button. Refer to Figure 27.

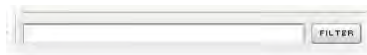


Figure 27: Function Code List Filter

The filter will then display only those function codes currently available in the Function Code List that satisfy the search criteria. For example, to find all monitor data 1 function codes that contain some derivative of the word "volt" (such as "voltage" or "volts"), select the "Monitor Data 1" group, enter "volt" in the filter entry box, and then click the "filter" button.

Once a filter has been entered, it will continue to be applied to all information normally displayed in the Function Code List for as long as the filter term is left in the filter entry box. Continuing the previous example where we filtered on the root term "volt" in the monitor data 1 group, we can then easily apply this filter to all available function codes simply by selecting the "All" function code group. The Function Code List will now display all command, monitor, configuration etc. function codes that contain the root term "volt".

To remove the filter, delete any characters contained in the filter entry box and then click the "filter" button.

### 5.3.5 Radix Selection

Figure 28 shows the radix selection buttons. These selection buttons allow changing the Function Code List "value" column data display and entry radix between decimal and hexadecimal formats.

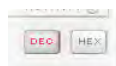


Figure 28: Radix Selection

When "DEC" is selected, the "value" column heading will be "*Value (Decimal)*", current function code values will be displayed in decimal, and values to be written to function codes must be entered in decimal format. For example, to change the inverter's frequency command to 40.00Hz, enter the decimal value 4000.

Similarly, when "HEX" is selected, the "value" column heading will be "*Value (Hexadecimal)*", current function code values will be displayed in hexadecimal, and values to be written to function codes must be entered in hexadecimal format. For example, to turn on bit #10 in the inverter's operation command word, enter the hexadecimal number 0400.

## 5.4 BACnet Tab

The BACnet tab provides for the configuration of the device on a BACnet/IP network. Refer to Figure 29.

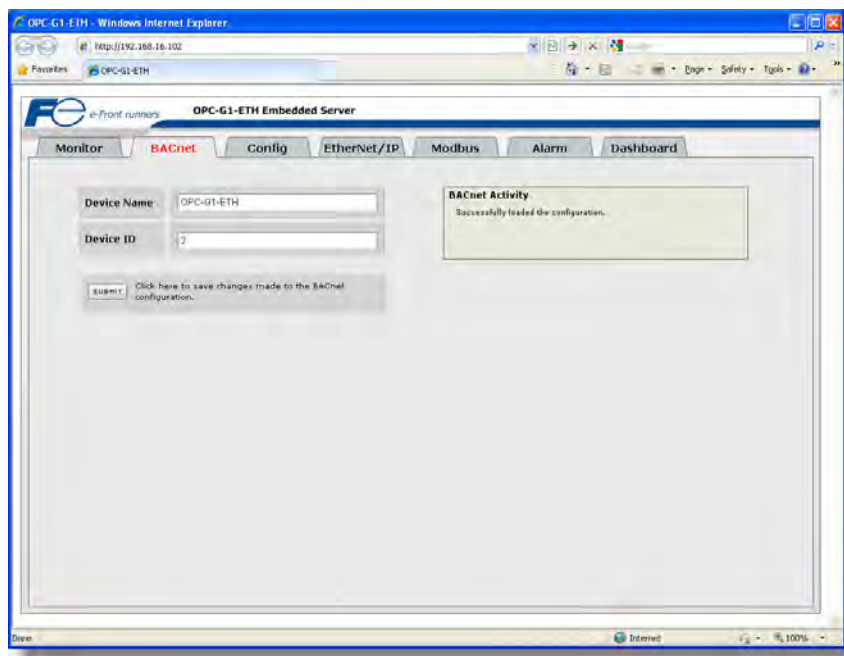


Figure 29: BACnet Tab

### 5.4.1 Information Window

Figure 30 shows the Information Window, which is located in the upper-right hand corner of the BACnet tab. This window displays various informational messages regarding the status of the BACnet configuration (loading or submitting).



Figure 30: BACnet Tab Information Window

### 5.4.2 Device Identifiers

A BACnet device's name and ID (the Object\_Name and Object\_Identifier properties, respectively, of the Device Object) must be unique across the entire BACnet network because they are used to uniquely identify BACnet devices. The text entry boxes shown in Figure 31 are used to configure these unique device identifiers on every inverter.



Figure 31: BACnet Device Identifiers

### 5.4.3 Submitting Changes

Whenever either of the BACnet configuration elements (Device Name or Device ID) has been changed, the “submit” button located in the left-hand portion of the web page must be clicked in order to write these settings to the interface card’s filesystem. Refer to Figure 32.



Figure 32: Submit BACnet Changes

### 5.4.4 Reinitialize Prompt

After successfully submitting a configuration, the user will be prompted to reinitialize the interface card for the new configuration to take effect. Refer to Figure 33. It is recommended to first submit all desired configurations before reinitializing the interface card. Please allow 10 seconds for the interface card to reinitialize at which time it will then be operating with the recently-submitted configuration. Reinitializing the interface card will not disturb the “back-end” communication with the drive or the functionality of the embedded switch.

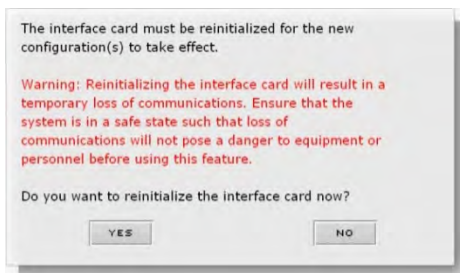


Figure 33: Reinitialize Device Prompt

Note that configuration elements are read from the filesystem only when the interface card boots up. If the user does not wish to reinitialize the interface card at this time, the interface card can also be power-cycled at a later time for the configuration to take effect.



## 5.5 Config Tab

The Config tab provides access to various configuration items. Refer to Figure 34.

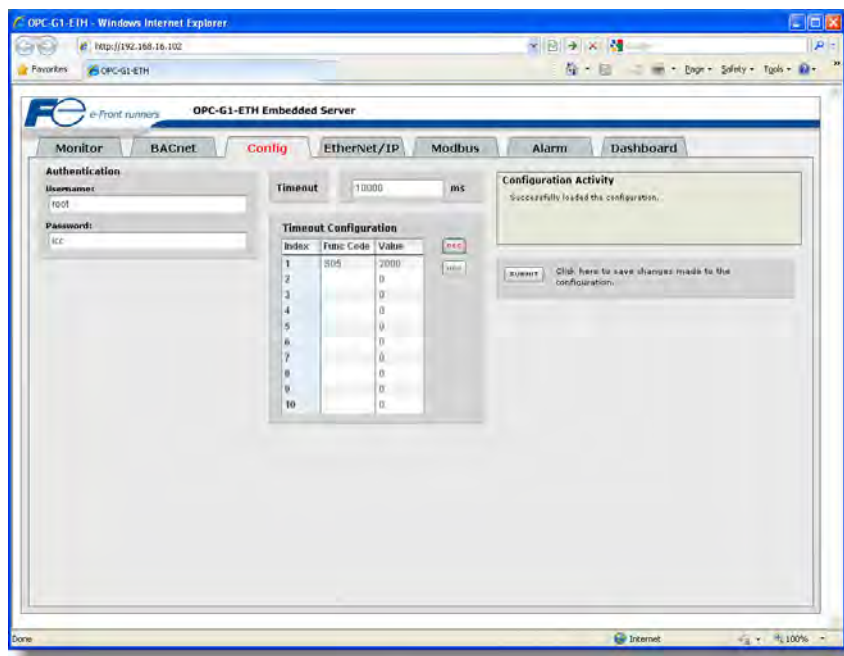


Figure 34: Config Tab

### 5.5.1 Information Window

Figure 35 shows the Information Window, which is located in the upper-right hand corner of the Config tab. This window displays various informational messages regarding the status of the configuration parameters (loading or submitting).



Figure 35: Config Tab Information Window

### 5.5.2 Authentication Configuration

Figure 36 shows the entry boxes used to modify the authentication credentials. The case-sensitive username and password can contain letters ("a".."z" and "A".."Z") and numbers ("0".."9"), and can each be up to 80 characters in length.

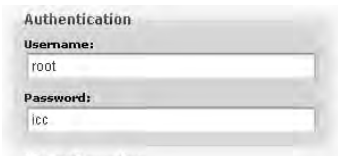


Figure 36: Authentication Configuration

### 5.5.3 Timeout Configuration

The interface can be configured to perform a specific set of actions when network communications are lost. The drive also has the ability to perform an independent set of actions when a timeout occurs (section 3.2). Support for the interface and drive timeout features varies depending on the protocol: refer to the protocol-specific section of this manual for further information.

There are two separate elements that comprise the timeout configuration (refer to Figure 37):

- The timeout time
- The timeout configuration array

The **timeout time** is a 32-bit unsigned value. This time setting is used by certain protocols in order to determine abnormal loss-of-communications conditions and, optionally, to trigger a timeout processing event. The default timeout time is 10s.

The **timeout configuration array** allows up to 10 function code/value pairs to be designated by the user. When a timeout event is triggered by a protocol, the timeout configuration array indexes are parsed. If the "function code" field for an index is blank, then this index is "disabled" and therefore ignored. If, on the other hand, the "function code" field is populated, then the value contained in the "value" field is automatically written to the designated function code. This flexible mechanism allows up to 10 designated inverter function codes to have their own unique "fail-safe" conditions in the event of a network interruption to allow the user to determine any inverter behavior they may desire (stop the inverter, fault the inverter, ramp to a preset speed, etc.)

For example, Figure 37 shows a timeout time of 10s, and one timeout entry assignment. If a protocol that makes use of timeout processing triggers a timeout event, then a value of 2000 will automatically be written to inverter function code S05 (the frequency command). Provided the inverter has a valid "run" command and is currently configured to use the network frequency command as its master frequency command, it will ramp to 20.00Hz.

If timeout/failsafe processing is not desired, just clear the "function code" fields for all indexes (disabled). This is the default condition.

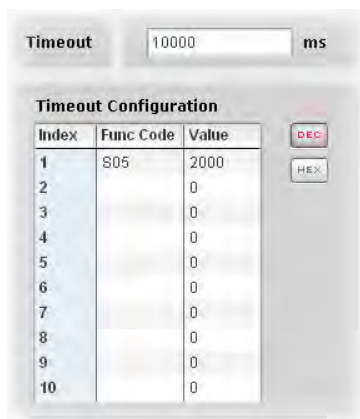
"DEC" and "HEX" selection buttons are also available, and allow changing the "value" column data display and entry radix between decimal and hexadecimal formats, respectively. These buttons provide the ability to interact with the various inverter function codes in their most natural radix (e.g. a hexadecimal command word vs. a decimal frequency command value).

#### 5.5.4 Submitting Changes

Whenever any of the configuration elements has been changed, the "submit" button located in the right-hand portion of the web page must be clicked in order to write these settings to the interface card's filesystem. Refer to Figure 38.

#### 5.5.5 Reinitialize Prompt

Refer to section 5.4.4.



Index	Func Code	Value
1	S05	2000
2		0
3		0
4		0
5		0
6		0
7		0
8		0
9		0
10		0

Figure 37: Timeout Configuration



Figure 38: Submit Configuration Changes

## 5.6 EtherNet/IP Tab

The EtherNet/IP tab provides access to configuration items related to communication on an EtherNet/IP network. Refer to Figure 39.

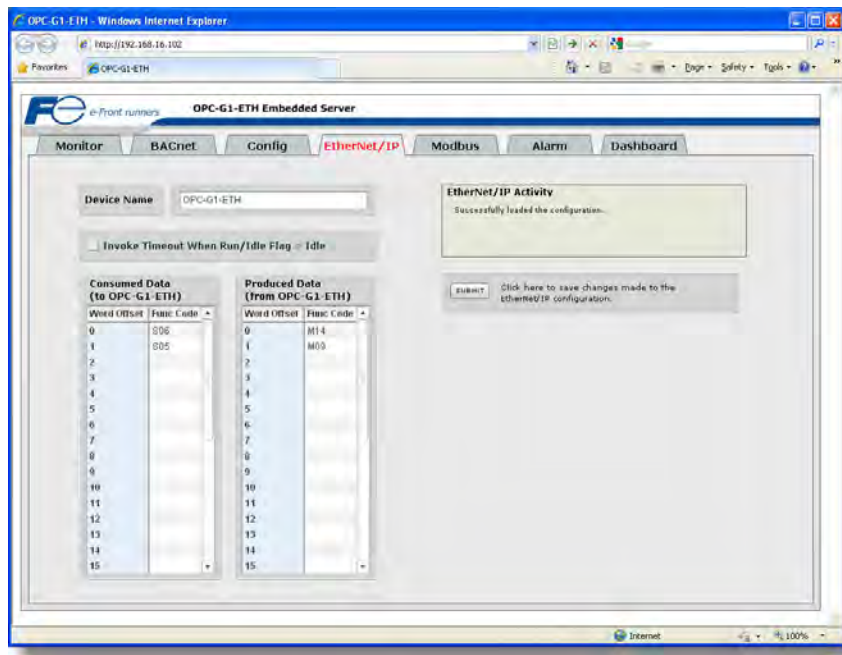


Figure 39: EtherNet/IP Tab

### 5.6.1 Information Window

Figure 40 shows the Information Window, which is located in the upper-right hand corner of the EtherNet/IP tab. This window displays various informational messages regarding the status of the EtherNet/IP configuration parameters (loading or submitting).



Figure 40: EtherNet/IP Tab Information Window

### 5.6.2 Device Identification

A text entry box is available which allows customization of the device's name for identification on the EtherNet/IP network. This string is accessible as the "product name" attribute of the identity object. Refer to Figure 41.



Figure 41: EtherNet/IP Device Identification

### 5.6.3 Run/Idle Flag Behavior

EtherNet/IP clients (such as PLCs) have the option of adding a 32-bit "run/idle" header to all class 1 (I/O) data packets sent to devices. Bit 0 of this header is called the "run/idle flag" by the EtherNet/IP specification, and is intended to signify when the client is in a "running" state or an "idle" state. A running state (run/idle flag = Run) is indicated whenever the client is performing its normal processing (e.g. scanning its ladder logic). An idle state (run/idle flag = Idle) is indicated otherwise. For example, Allen Bradley ControlLogix PLCs will set their run/idle flag to Idle whenever their processor keyword is placed in the "PROG" position, presumably in preparation to receive a new application program from RSLogix.

☐ Invoke Timeout When Run/Idle Flag = Idle

**Figure 42: Run/Idle Flag Behavior Selection**

The behavior of EtherNet/IP devices when they receive I/O data from a controller with the run/idle flag set to Idle is not specified in the EtherNet/IP specification. The interface card allows the option of two different behavioral responses when a run/idle flag = Idle condition is received, depending on the state of the checkbox indicated in Figure 42.

- If the checkbox is cleared (default setting), then the interface card will maintain the last I/O data values received from the client. For example, if the inverter was being commanded to run prior to the run/idle flag being set to Idle, then it will continue to run.
- If the checkbox is checked, then the interface card will invoke its user-configured timeout processing (refer to section 5.5.3).

### 5.6.4 Class 1 (I/O) Data Configuration Arrays

The interface card supports two different types of EtherNet/IP class 1 (I/O) data transfer. One type is included with the implementation of the AC/DC drive profile, and requires no user configuration. The other type, however, is entirely user-configurable, and is utilized when the client opens a connection to the interface using assembly instances 100 and 150.

The user-configurable data arrays consist of two separate elements (refer to Figure 43.) The consumed data configuration defines the structure of the command data sent from the EtherNet/IP controller (for example, a ControlLogix PLC) to the inverter (O->T direction), and the produced data configuration defines the structure of the status data sent from the inverter back to the controller (T->O direction). These arrays allow the creation of custom-built I/O data. Up to 32 command function codes can be sent to the inverter, and up to 32 status function codes can be sent back to the controller. Each box in the "function code" column is capable of containing an inverter function code. Because all inverter function codes are 16-bit data elements, each box therefore represents two bytes of consumed or produced data.

Consumed Data (to OPC-G1-ETH)		Produced Data (from OPC-G1-ETH)	
Word Offset	Func Code	Word Offset	Func Code
0	S06	0	M14
1	S05	1	M09
2		2	
3		3	
4		4	
5		5	
6		6	
7		7	
8		8	
9		9	
10		10	
11		11	
12		12	
13		13	
14		14	
15		15	

**Figure 43: EtherNet/IP Class 1 (I/O) Data Configuration**

Each of the function code array locations are numbered 0-31 in the "word offset" column. Clicking on a box in an array allows the user to enter a function code that will be referenced at that location when data is either consumed from the controller or produced to the network. If an attempt is made to enter an invalid function code, an error dialog box will appear. A blank value indicates that no function code is referenced at that location, which will cause the corresponding consumed data to be ignored and produced data to be a default value of 0.

As an example, looking at the default configuration shown in Figure 43, we can see that each array contains two defined function codes. Therefore, up to 4 "meaningful" bytes of data can be both received and sent (the qualifier "meaningful" is used here because the connection sizes configured in the

controller may request larger consumed and/or produced data sizes, but all unreferenced consumed data will be ignored, and all unreferenced produced data will contain dummy "0" values). The first word (two bytes) of consumed data will be written to function code S06 (operation command register) and the second word will be written to function code S05 (frequency command). Similarly, the first word of produced data will contain the value of function code M14 (status register) and the second word will contain the value of function code M09 (output frequency).

### **5.6.5 Submitting Changes**

Whenever any of the EtherNet/IP configuration elements (Device Name, I/O array configurations etc.) have been changed, the "submit" button located in the right-hand portion of the web page must be clicked in order to write these settings to the interface card's filesystem. Refer to Figure 44.



**Figure 44: Submit Configuration Changes**

### **5.6.6 Reinitialize Prompt**

Refer to section 5.4.4.

## 5.7 Modbus Tab

The Modbus tab provides access to configuration items related to communication on a Modbus/TCP network. Refer to Figure 45.

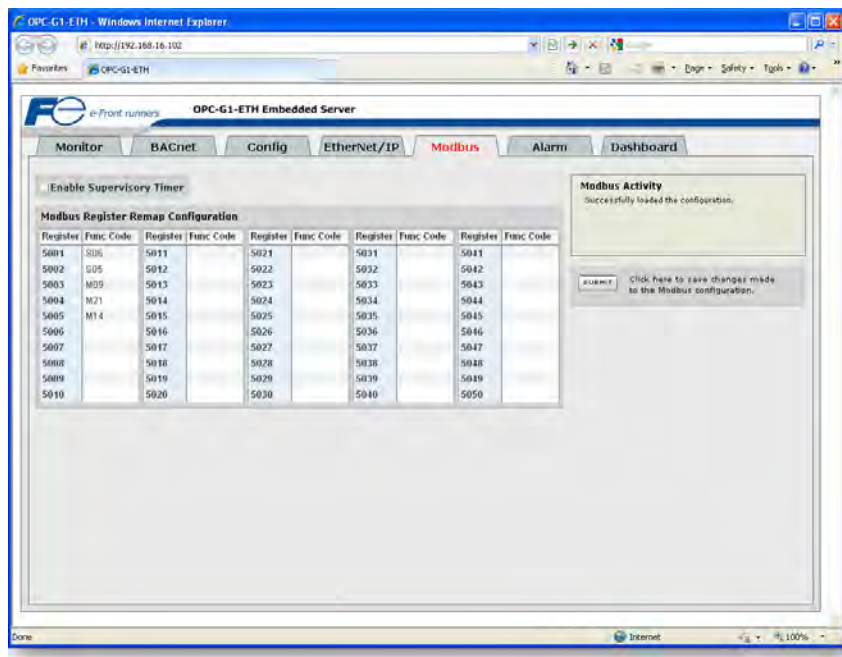


Figure 45: Modbus Tab

### 5.7.1 Information Window

Figure 46 shows the Information Window, which is located in the upper-right hand corner of the Modbus tab. This window displays various informational messages regarding the status of the Modbus configuration parameters (loading or submitting).

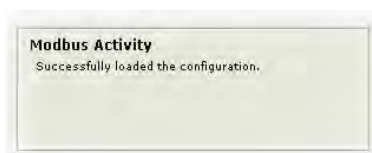


Figure 46: Modbus Tab Information Window

### 5.7.2 Supervisory Timer Selection

Figure 47 shows the checkbox which enables the interface card's Modbus "supervisory timer" function. This timer provides the ability for the interface card to monitor timeout occurrences between successive Modbus/TCP socket connections, as opposed to the standard timeout functionality (refer to section 5.5.3), which monitors timeout occurrences only within the scope of each client socket connection. While this feature provides an additional level of fail-safe functionality for those applications that require it, there are several ramifications that must be understood prior to enabling this capability. Please contact technical support for a more in-depth explanation.

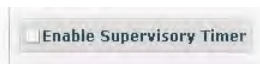


Figure 47: Supervisory Timer Selection

### 5.7.3 Register Remap Configuration

At times, it may be convenient to access inverter registers (function codes) in bulk Modbus transactions. This may be especially true in situations where it is desired to access certain registers that are natively non-contiguous. For example, if it were desired to read the inverter's operating frequency (function code M09, register 2058), DC link bus voltage (function M21, register 2070), and operation status (function code M14, register 2063), this could be accomplished in two different ways:

1. Implement three separate Modbus read transactions, each one reading one register only, or
2. Implement one single Modbus read transaction, starting at register 2058 for a quantity of 13 registers. Then, pick out the registers of interest and ignore the rest of the response data.

While both of these methods will certainly work, neither one of them is optimized for the task at hand, which is to access three specific register values. A fully optimized solution can be realized, however, by making use of the interface card's Modbus register remapping capabilities. This mechanism operates by allocating a block of 50 user-configurable registers (5001..5050) that remap to other inverter registers. In this way, non-contiguous inverter registers can be grouped together in any order and accessed efficiently via the Modbus/TCP "read multiple registers" and "write multiple registers" function codes. The net effect is one of being able to transfer larger blocks of registers using fewer Modbus transactions, which results in improved network utilization and simpler data manipulation code on the Modbus master device.

Figure 48 shows the register remap configuration array. Clicking on an entry field in the "Func Code" column allows the user to enter an inverter function code that will then be accessible at the register indicated in the adjacent "Register" column. A blank assignment in the "Func Code" column indicates that no inverter function code is remapped at that location, which results in written values being ignored and read values returned as a default value of 0. Note that remapped inverter registers (function codes) are still accessible at their original locations: remapping simply provides an additional means of accessing the original register's value.

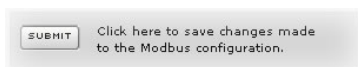
Register	Func Code	Register	Func Code	Register	Func Code	Register	Func Code	Register	Func Code
5001	S06	5011		5021		5031		5041	
5002	S05	5012		5022		5032		5042	
5003	M09	5013		5023		5033		5043	
5004	M21	5014		5024		5034		5044	
5005	M14	5015		5025		5035		5045	
5006		5016		5026		5036		5046	
5007		5017		5027		5037		5047	
5008		5018		5028		5038		5048	
5009		5019		5029		5039		5049	
5010		5020		5030		5040		5050	

**Figure 48: Modbus/TCP Register Remap Configuration**

As an example, the configuration shown in Figure 48 reveals that a total of five inverter registers (function codes) have been remapped. Function code S06, register 1799 (operation command word) has been remapped to register 5001. Function code S05, register 1798 (frequency command) has been remapped to register 5002. Function code M09, register 2058 (output frequency) has been remapped to register 5003. Function code M21, register 2070 (DC link bus voltage) has been remapped to register 5004. Function code M14, register 2063 (operation status word) has been remapped to register 5005. With this configuration, it is now possible to efficiently interact with these five non-contiguous inverter registers via just two Modbus "read/write multiple registers" transactions. Writing to the frequency command and command word can be accomplished with a single "write multiple registers" transaction by writing a quantity of two registers starting at register 5001. Similarly, reading the output frequency, DC link bus voltage and operation status word (in that order) can be accomplished with a single "read multiple registers" transaction by reading a quantity of three registers starting at register 5003.

#### **5.7.4 Submitting Changes**

Whenever the Modbus configuration has been changed, the "submit" button located on the right-hand portion of the web page must be clicked in order to write these settings to the interface card's filesystem. Refer to Figure 49.



**Figure 49: Submit Configuration Changes**

#### **5.7.5 Reinitialize Prompt**

Refer to section 5.4.4.



## 5.8 Alarm Tab

The Alarm tab provides a configurable mechanism by which the interface card can autonomously monitor any available inverter function code and send emails to up to four recipients when a certain condition is detected. The alarm conditions have both value and time constraints, and can be configured to retrigger at a fixed interval as long as the alarm condition continues to be satisfied. Twenty individually-configurable alarms are available. Refer to Figure 50.

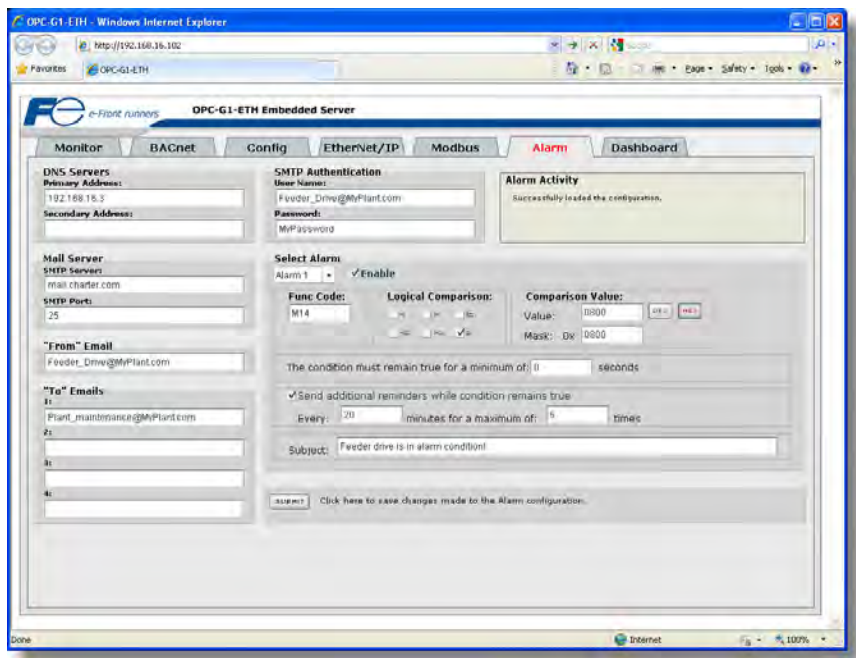


Figure 50: Alarm Tab

### 5.8.1 Information Window

Figure 51 shows the Information Window, which is located in the upper-right hand corner of the Alarm tab. This window displays various informational messages regarding the status of the Alarm configuration parameters (loading or submitting).



Figure 51: Alarm Tab Information Window

## 5.8.2 Email Configuration

In order for an alarm trigger to successfully send a notification email, some network settings must first be configured properly (refer to Figure 52).

**DNS Servers:** Enter the dotted-decimal IP addresses of the primary and secondary DNS servers which will be used to resolve the configured SMTP server name. Only the primary DNS server is required, but if a secondary DNS server is entered, then it will be used if the primary server is inaccessible.

**Mail Server:** Enter the SMTP server address as a name or as a dotted-decimal IP address, and the SMTP port (default=25) that the SMTP server listens for incoming emails on.

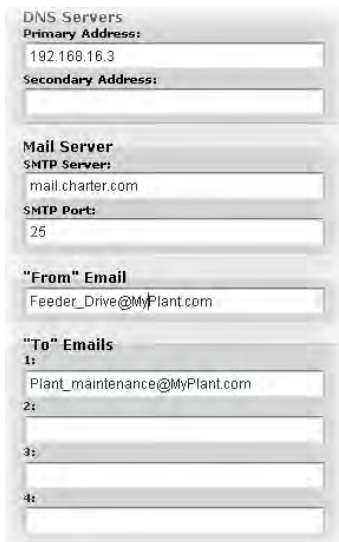
**"From" Email:** Enter the email address that will appear as the sender's email address in the email headers.

**"To" Emails:** Up to four recipients can be designated to receive alarm emails. Blank entries will not be processed by the interface.

**SMTP Authentication:** If the email server in use does not require authentication, then these entries can be disregarded. Some email servers do require that clients wishing to send emails first authenticate themselves. If the email server in use requires authentication, then enter the user name and password as indicated in Figure 53. The following authentication mechanisms are supported and are listed from highest priority to lowest priority:

- DIGEST-MD5
- CRAM-MD5
- LOGIN
- PLAIN

The highest priority authentication mechanism that is supported by the email server will be used.



The form is titled "Email Configuration" and contains several sections:

- DNS Servers:**
  - Primary Address:** 192.168.16.3
  - Secondary Address:** (empty field)
- Mail Server:**
  - SMTP Server:** mail.charter.com
  - SMTP Port:** 25
- "From" Email:** Feeder\_Drive@MyPlant.com
- "To" Emails:**
  - 1: Plant\_maintenance@MyPlant.com
  - 2: (empty field)
  - 3: (empty field)
  - 4: (empty field)

Figure 52: Email Configuration



The form is titled "SMTP Authentication" and contains two fields:

- User Name:** Feeder\_Drive@MyPlant.com
- Password:** MyPassword

Figure 53: SMTP AUTH Configuration

### 5.8.3 Alarm Configuration

The interface supports twenty independently-configurable alarms. As shown in Figure 54, each alarm has a variety of configuration elements, which will be explained further below.

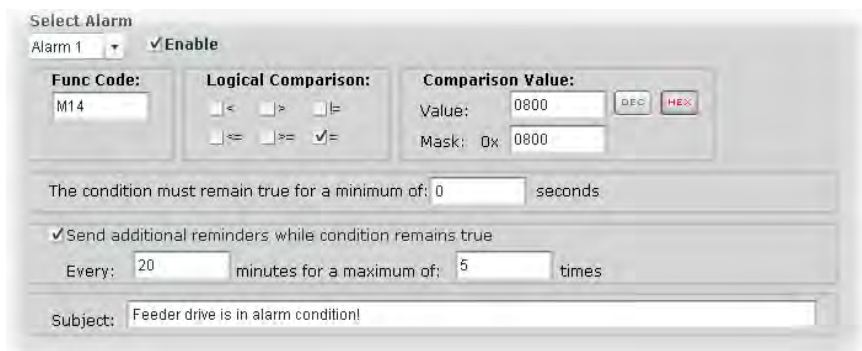


Figure 54: Alarm Configuration Box

**Alarm Selection:** This drop-down box allows the selection of one of the twenty available alarms. When an alarm is selected, that alarm's current configuration parameters will be populated in the alarm configuration box.

**"Enable" Check Box:** If checked, this alarm is active and will be evaluated every second. If unchecked, this alarm is inactive and will therefore not be evaluated.

**Func Code:** Enter the inverter function code that this alarm will continuously monitor. For example, the alarm displayed in Figure 54 is configured to monitor M14, which is the operation status register.

**Logical Comparison:** Choose a comparison operator which will be used to compare the current value of the indicated "Func Code" with the reference "Comparison Value". Available selections are "less than" (<), "less than or equal to" (<=), "greater than" (>), "greater than or equal to" (>=), "not equal to" (!=), and "equal to" (=).

**Comparison Value:** The reference comparison value is comprised of two subcomponents: a "Mask" field and a "Value" field. Each time the alarm is evaluated, the current value of the indicated "Func Code" is first bit-wise "AND"ed with the "Mask" field. The resulting derived value is then compared with the "Value" field by way of the "Logical Comparison" operator. While the "Mask" field is always a hexadecimal number, the display and entry radix of the "Value" field can be changed between decimal and hexadecimal with the associated "DEC" and "HEX" buttons.

Function codes that correspond to "analog" process variables (e.g. frequencies, voltages, etc.) should typically have their "Mask" fields set to 0xFFFF, which causes all data bits to be retained for the "Value" field comparison. For function codes that correspond to "enumerated" process variables (e.g. status words where each bit of the function code indicates a different item), however, the "Mask" can be chosen to single out one or more specific data bits of the function code. For example, the "Mask" value of 0x0800 displayed in Figure 54 isolates bit #11 of the operation status register, which indicates whether or not an inverter alarm exists. The "Value" field is also set to a hexadecimal value of 0x0800, so the alarm condition will be evaluated as "true" when bit #11 of the operation status register equals 1.

**The Condition Must Remain True For A Minimum Of:** Alarm analysis processing is performed by the interface card once per second. Enter the number of seconds that the condition must be continuously evaluated as "true" for the alarm to be triggered. A time of 0 seconds means that just a single evaluation of "true" will immediately trigger the alarm.

**Send Additional Reminders While The Condition Remains True:** If this check box is unchecked, then only one email transmission event will occur when an alarm condition is triggered: further email transmissions will not be attempted for this alarm unless the alarm condition is first evaluated as "false" (which resets the alarm), and then once again is triggered by a subsequent event.

If this check box is checked, then as long as the alarm condition continues to be evaluated as “true”, subsequent email transmissions will be automatically retriggered every indicated number of minutes for a maximum of the indicated number of times. If at any time during the subsequent transmissions the alarm condition is evaluated as “false”, then the alarm will be reset and email transmissions for this alarm will stop (until the next time the alarm is triggered, of course).

**Subject:** Enter a string of up to 128 characters in length which will appear in the “subject” line of the alarm email. The body of the alarm email is empty.

#### **5.8.4 Submitting Changes**

Whenever any of the Alarm configuration elements (alarm settings or email configuration parameters) have been changed, the “submit” button located in the lower right-hand portion of the web page must be clicked in order to write these settings to the interface card’s filesystem.

Note that because these configuration elements are read from the filesystem only when the interface card boots up, the act of submitting configuration changes will also reset the interface card. Please allow 10 seconds for the interface card to reboot, at which time it will then be operating with the recently-submitted configuration. Refer to Figure 55.



**Figure 55: Submit Configuration Changes**

## 5.9 Dashboard Tab

The Dashboard Tab provides access to a virtual keypad, as well as a variety of gauges, meters and graphs that can be configured to provide an at-a-glance graphical overview of critical application variables in real-time. A total of 10 gauge windows are available (two at a time), and each gauge window can be configured to display any scanned function code's value via one of six different gauge types. User-defined engineering units, scaling and range limits are also configurable. Refer to Figure 56.



Figure 56: Dashboard Tab

### 5.9.1 Information Window

Figure 57 shows the Information Window, which is located in the upper-right hand corner of the Dashboard Tab. This window displays various informational messages regarding the status of the Dashboard configuration parameters (loading or submitting).



Figure 57: Dashboard Tab Information Window

### 5.9.2 Virtual Keypad

A "virtual keypad" is displayed on the left-hand side of the dashboard tab, and acts as an interface for several useful pieces of control and monitor information. For an overview of the virtual keypad interface, refer to Figure 58. Note that it is recommended to suspend all external protocol-based communications with PLC's, etc when using the virtual keypad, as other protocols may simultaneously be writing to the inverter's frequency command and operation command word, resulting in seemingly unpredictable behavior.

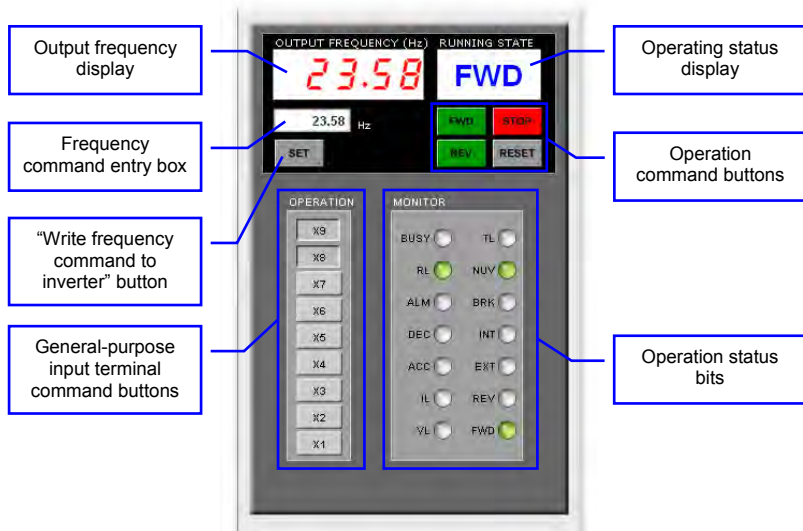


Figure 58: Virtual Keypad Overview

**Output frequency display:** Indicates the current output frequency of the inverter in large red numbers, as reported by inverter function code M09. The image in Figure 58 indicates that the associated inverter is currently running at 23.58Hz.

**Frequency command entry box:** Allows the user to enter a new frequency command for the inverter, which is subsequently scaled and written to inverter function code S05 when the "SET" button is clicked.

**"SET" button:** Clicking this button will scale and write the value contained in the frequency command entry box to inverter function code S05. Note that the inverter will use this frequency command as its master frequency reference only when configured accordingly (refer to section 3.1).

**General-purpose input terminal command buttons:** These buttons (labeled "X1" through "X9") map to the corresponding bits in the inverter's operation command word (function code S06, bit #2 .. bit #10). The usage of these bits varies depending on the configuration of inverter function codes E01 to E09. When a given button is clicked and shown in its depressed state, the corresponding bit is set to a "1". When clicked again (and therefore shown in its non-depressed state), the corresponding bit is set to a "0". As an example, the image in Figure 58 shows X1..X7 as OFF ("0"), and X8 and X9 as ON ("1"). Note that controlling these operation command word bits will only affect the inverter when it is configured accordingly (refer to section 3.1).

**Operating status display:** Indicates the current state of the inverter based on bits in the inverter operation status register (function code M14). Possible displays include "STOP", "FWD", "REV" and "FAULT".

**Operation command buttons:** Clicking on these buttons enables different control actions to be imposed on the inverter, as follows:

- **FWD:** sets bit #0 ("FWD") and clears bit #1 ("REV") in the operation command word (function code S06).

- **REV:** sets bit #1 ("REV") and clears bit #0 ("FWD") in the operation command word (function code S06).
- **STOP:** clears both bit #0 ("FWD") and bit #1 ("REV") in the operation command word (function code S06).
- **RESET:** writes a value of "1" to function code S14 (alarm reset command). This will reset a faulted inverter regardless of the current operation command mode (H30, Y98 etc.) Note that if the inverter was running (the "FWD" or "REV" buttons were the last buttons pressed on the virtual keypad before the fault occurred), the STOP button must be clicked prior to clicking the RESET button in order to clear the FWD and REV bits in the operation command word. The inverter will ignore reset commands issued through function code S14 as long as a valid run command still exists in the operation command word.

Note that the inverter will follow the FWD, REV and STOP button commands only when configured accordingly (refer to section 3.1).

**Operation status bits:** These "virtual LEDs" map to the corresponding bits of the same name in the inverter's operation status word (function code M14). When a given bit in the status word is "1", then its corresponding indicator will be lit. The indicator will not be lit if its status word bit is "0". As an example, the image in Figure 58 shows FWD (bit #0), NUV (bit #5) and RL (bit #12) ON, and all other bits OFF.

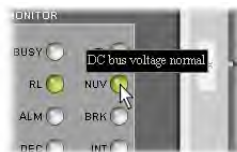


Figure 59: Virtual LED Tooltips



Hovering the cursor over the virtual LEDs will bring up a tooltip which provides a brief summary of the indicated function. Refer to Figure 59.

### 5.9.3 Gauge Window Navigation

Figure 60 shows the two buttons that provide for navigation of the gauge windows. Gauge windows are displayed two at a time in the Dashboard Tab, and by clicking the "right" or "left" buttons, the gauge windows will scroll in the corresponding direction.



Figure 60: Gauge Window Navigation

### 5.9.4 Gauge Window Configuration

Each of the gauge windows can be independently configured to display a user-defined function code with a variety of flexible configuration options. While the behavior and presentation may vary slightly depending on the specific gauge chosen, all of the gauges share the following common elements (refer to Figure 61 for an example):

**Gauge Selector:** A drop-down selection box in the upper left-hand corner of the gauge window, which allows the user to select the type of gauge that will be displayed.

**Title:** A text entry box located above the gauge, in which the user can enter a descriptive gauge title comprised of up to 16 characters.

**Units:** A text entry box in which the user can enter an engineering units string comprised of up to 8 characters. This units string will be appended to all locations in the gauge window that display the designated function code's current value.

**Function Code:** The designated function code whose value is to be reflected on the gauge. Note that only scanned function codes may be displayed in Dashboard gauges (refer to section 6.2 for a discussion of scanned function codes).

**Multiplier:** The multiplier value is a floating-point number that is used to scale the raw value of a function code. As its name suggests, the multiplier value is multiplied by the designated function code's current raw value in order to calculate the gauge's indicated value. Negative values can also be used if desired.

**Min Value:** The gauge's minimum indicated value. Negative values can be used if desired (e.g. if a negative Multiplier attribute is used to generate a negative indicated value). Not all gauges allow adjustment of the min value.



**Max Value:** The gauge's maximum indicated value. Similar to the Min Value attribute, negative values can be used if desired. Indicated value characteristics can even be inverted by setting the Max Value attribute to a value less than the Min Value attribute.

**Update Button:** Clicking the update button will apply the current configuration attribute settings to the gauge. Note, however, that simply updating the gauge's current display properties does not write these settings to the interface card's filesystem. To save the current configuration of all the gauge windows to the filesystem, the Dashboard tab's "submit" button must be selected (refer to section 5.9.5).

**Current Value:** The current indicated value of the designated function code is numerically displayed with the configured Units string at the bottom of each gauge window.

The following is a summary of the different available gauge types:

**Gauge:** Refer to Figure 61. This type of meter implements a rotary dial-type display format. The indicated value and units are shown numerically on the face of the gauge, and via the red indicator needle. The yellow needle shows the previous indicated value, thereby providing a simple historical reference. The "Min Value" attribute is not configurable; this gauge always starts at 0.

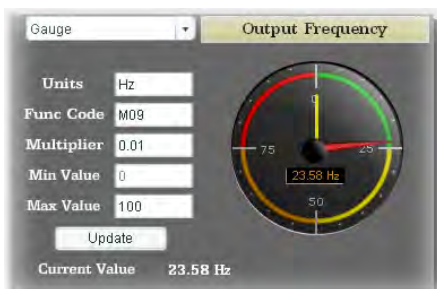


Figure 61: Gauge

**BarGraph:** Refer to Figure 62. This type of meter implements a linear bar graph display format. Hovering the mouse pointer over the red portion of the graph pops up a tooltip which displays the current indicated value and units.

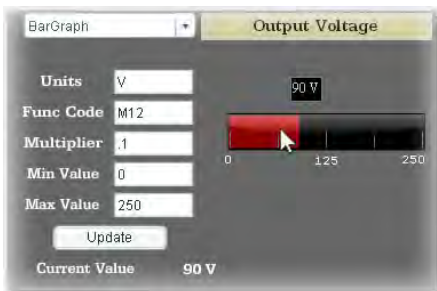


Figure 62: BarGraph

**Meter:** Refer to Figure 63. This type of meter implements a common panel meter-type display format. The units string is shown on the face of the meter. All raw function code values are interpreted as positive numbers (i.e. 0..0xFFFF equates to 0..65535<sub>10</sub>.)

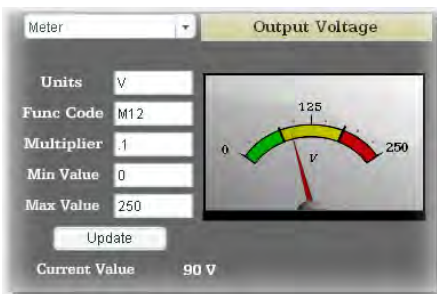


Figure 63: Meter



**Pos/Neg Meter:** Refer to Figure 64. Similar to the “meter” gauge, this type of meter also implements a common panel meter-type display format, but in this instance the indicated value can be positive or negative (two’s complement interpretation). In other words, raw function code values of 0..0x7FFF equate to 0..32767<sub>10</sub>, and values of 0x8000..0xFFFF equate to -32768..-1. Because the meter placard is always centered around zero, the “Min Value” attribute is not configurable, and the “Max Value” attribute is used for both the maximum positive indicated value as well as the maximum negative indicated value.

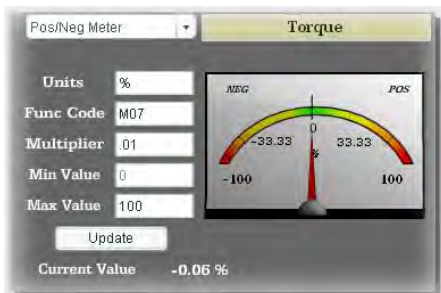


Figure 64: Pos/Neg Meter

**Thermometer:** Refer to Figure 65. This type of meter implements the universally-identifiable thermometer display format. Hovering the mouse pointer over the red “mercury” portion of the graph pops up a tooltip which displays the current indicated value and units.

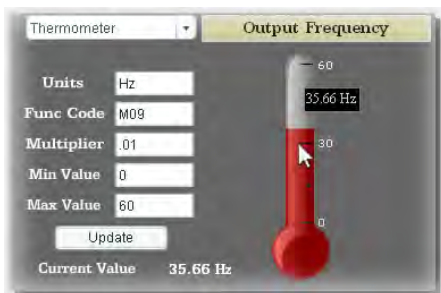


Figure 65: Thermometer

**Line Graph:** Refer to Figure 66. This type of graph implements a continuously-scrolling historical data logging line graph. Up to 80 seconds worth of historical data is available. Hovering the mouse pointer anywhere on the graph displays a vertical reference line at the corresponding time, and pops up a tooltip which displays the current indicated value at that time.

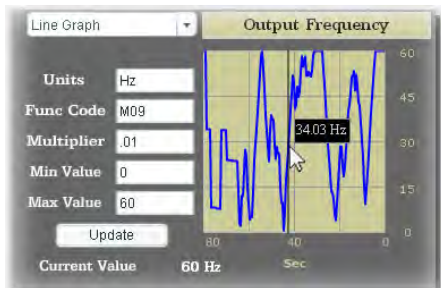
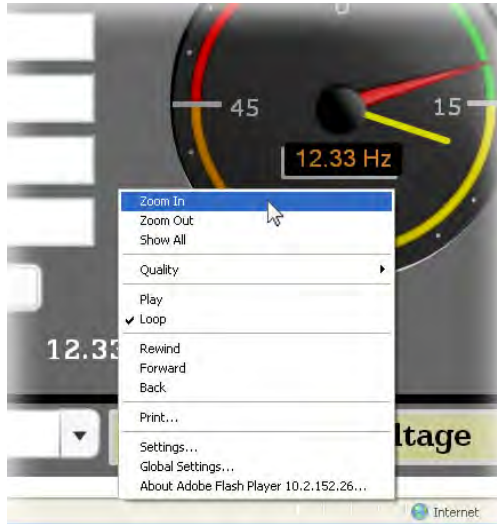


Figure 66: Line Graph



At times, it may be convenient to zoom in on a particular gauge or meter in order to more clearly see the indicator, or to fill the computer screen with a particular gauge's image. This can be easily accomplished with the web browser's Flash Player plug-in by right-clicking on the gauge and selecting the desired zoom level (refer to Figure 67).



**Figure 67: Zooming**

### 5.9.5 Submitting Changes

Whenever any of the gauge window configuration items in the Dashboard Tab have been changed, the "submit" button located on the right-hand portion of the web page must be selected in order to write these settings to the interface card's filesystem. Refer to Figure 68. Note that submitting the Dashboard Tab configuration does not require rebooting of the interface card: the changes take effect immediately, and the interface card continues its operation without interruption.



**Figure 68: Submit Dashboard Changes**

## 5.10 Customizing the Embedded Web Server

---

### 5.10.1 Customization Overview

It is possible for end-users to customize the embedded web server in order to create their own application-specific or corporate “look and feel”. Knowledge of authoring dynamic web content is required. Using windows explorer, it is possible to load customized web server content into the “WEB” folder on the interface card’s file system (refer to section 7.2). Usually, this web server content contains programming which implements the XML socket-based XTPro protocol (refer to section 5.10.2). Via XTPro, the embedded web server can gain access to any inverter parameter and the interface card file system resources, and manipulate them as required.

#### **Notes**

- There is an XML file located in the “WEB” folder called “*frenicMegaParam.xml*”, which contains definitions for all inverter function codes that are available via the interface card. This file must not be removed, as it contains the definition of all available parameters not only for active web server content, but also for the interface card itself. All other files in the “WEB” folder may be deleted or replaced if desired by the user.
- The default HTML file targeted by the web server is “index.htm”. Therefore, when customizing the web server content, ensure that initial file “index.htm” exists.
- All files accessed by the web server itself must reside in the “WEB” folder. Note that this does not restrict active web server content to using only the “WEB” folder, however, as XTPro “read\_file” and “write\_file” commands can access any existing location on the file system.
- If the factory-default “WEB” folder contents need to be recovered (if they are accidentally deleted, for example), they can be downloaded from the device’s product page on the internet.
- Two simultaneous web server sessions are supported. Note that the number of available simultaneous web server sessions is independent of the number of available simultaneous XTPro XML sockets.

### 5.10.2 XTPro Overview

XTPro is an acronym for **XML TCP/IP Protocol**. The XTPro specification is an application-layer (positioned at level 7 of the OSI model) messaging protocol that provides XML-based client/server communication via TCP port 2000. Typically, XTPro is used for the implementation of graphical user interfaces (GUIs), such as advanced web servers or HMIs that have the ability to request information via XML sockets, and then manipulate and/or display the information in a rich application-specific manner.

XTPro is a request/response protocol that provides services specified by commands. For more information on XTPro, refer to the separate [XTPro Specification](#). This section will cover the device-specific implementation of the XTPro protocol.

### 5.10.3 XTPro Web Browser-Based Implementation

A representative implementation based upon using a web browser as the client is detailed in Figure 69. In this scenario, the client application is developed by using an active web server authoring tool (such as Adobe Flash®). The active content is then embedded into one or more HTML files and loaded onto the device's file system (refer to section 5.10.1 for detailed information regarding customization of the web server content). Accessing the device's web server via a standard web browser then loads the active content, which initiates communication with the server.

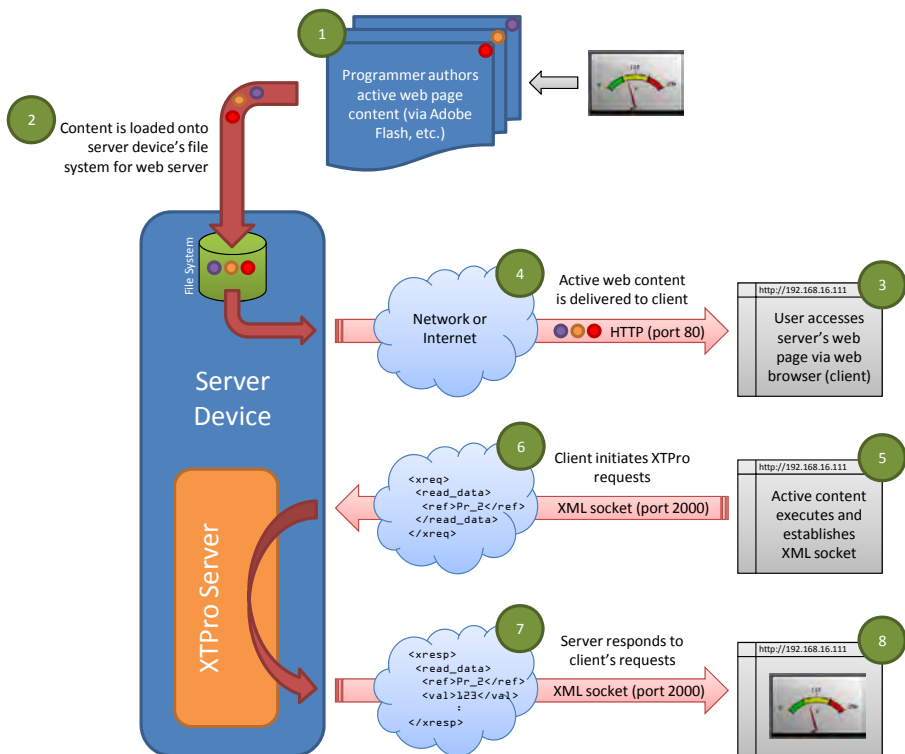


Figure 69: Web Browser-Based Implementation

#### 5.10.4 XTPro HMI-Based Implementation

A representative implementation based upon a stand-alone HMI client is detailed in Figure 70. In this scenario, the client application is developed by using tools provided by the HMI manufacturer, and is hosted independently of the actual server device.

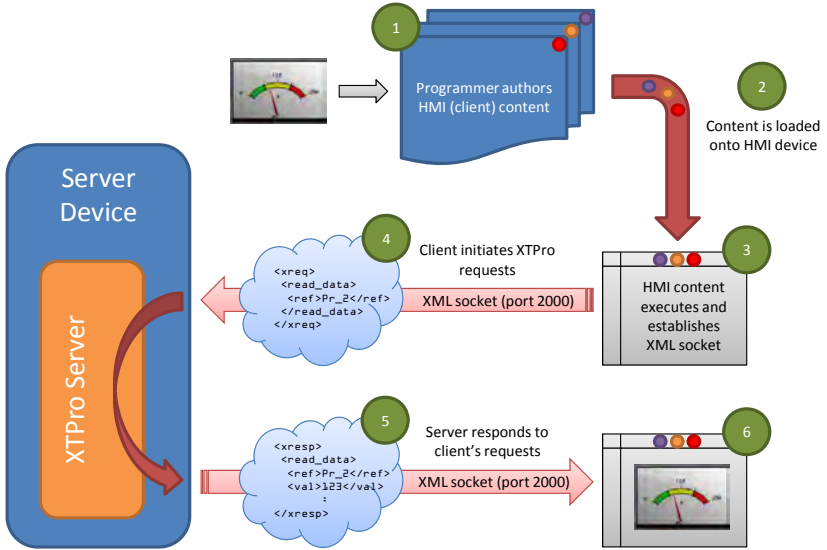


Figure 70: HMI-Based Implementation

#### 5.10.5 XTPro Supported Commands

For a summary of XTPro commands which are supported on the OPC-G1-ETH, refer to Table 4.

Table 4: Supported XTPro Commands

Command	Supported	Notes
<b>noop</b>	Yes	-
<b>vzn</b>	Yes	Supports XTPro specification version 1
<b>id</b>	Yes	-
<b>read_data</b>	Yes	"reference" is the inverter's function code (e.g. "F07" for acceleration time #1), while "data_value" is a 16-bit hexadecimal value (e.g. "1F4" for a decimal value of 500)
<b>write_data</b>	Yes	
<b>load_file</b>	Yes	The absolute file path must start with a forward slash '/'
<b>store_file</b>	Yes	
<b>reinit</b>	Yes	Reinitializes only the configurable drivers and services (does not perform a complete device soft reboot)
<b>auth</b>	Yes	Authorization is not required
<b>cov</b>	Yes	COV notification messages are sent every 200ms

#### Notes

- Two simultaneous XTPro connections are available.

## 6 FUNCTION CODE NUMBERING AND BEHAVIOR

### 6.1 Register Numbers

All accessible inverter function codes can be referenced by their Modbus register indexes, as defined in the RS-485 User's Manual (MEH448), section 3 (Table 3.2). These same register numbers are used when accessing function codes via certain Ethernet protocols (i.e. Modbus/TCP, AB CSP). The terms "function code" and "register" refer to data stored on the inverter and will be used interchangeably throughout this documentation. The max supported register number is 4964. Because the RS-485 User's Manual contains information for several Fuji inverter families, the relevant information will be paraphrased here for the specific case of the FRENIC-Mega.

All inverter function codes are exposed as register indexes according to a mathematical conversion formula which combines two elements (a function code group number and function code offset) to create a unique register number for each function code. Each function code group ("E" / Extension Terminal Functions, for example) is assigned a specific function code group number (refer to Table 5). Each function code also has an offset number, which is the function code without the leading letter (the offset number for function code E05, for example, is 5). To determine the register number for a given function code, therefore, the group number is first multiplied by 256, then added to the offset number plus 1. This operation is expressed mathematically via Equation 1.

$$\text{register} = (\text{group number} \times 256) + \text{offset number} + 1 \quad \text{Equation 1}$$

As an example, let's calculate the register number for output frequency (function code M09). According to Table 5, the group number for the "M" function code group is 8. It is also evident that the offset number for M09 is 9. Inserting the group number and offset number into Equation 1, we arrive at the result indicated in Equation 2.

$$(8 \times 256) + 9 + 1 = 2058 \quad \text{Equation 2}$$

While manually calculating all of the register numbers for the function codes of interest is certainly possible by using Equation 1, it may be more convenient to simply reference the "Register" column on the monitor tab of the default web interface (refer to section 5.3.3).

Note that not all of the available registers that exist in the interface card's register map have corresponding function codes that exist in the inverter. In other words, if a read from or write to a register number that does not correspond to an existing inverter function code takes place, the read/write may be successful (depending on the specific register accessed; refer to section 6.2), but the data will have no meaning. This feature is beneficial in situations where the accessing of non-contiguous registers can be made more efficient by accessing an all-inclusive block of registers (some of which correspond to inverter function codes and some of which do not), while only manipulating those in your local programming that are known to exist.

**Table 5: Function Code-to-Register Conversion Examples**

Function Code Group		Group Number	Example
Code	Name		
F	Fundamental Functions	0	F07 (acceleration time 1): $(0 \times 256) + 7 + 1 = 8$
E	Extension Terminal Functions	1	E98 (terminal [FWD] function): $(1 \times 256) + 98 + 1 = 355$
C	Control Functions	2	C20 (jogging frequency): $(2 \times 256) + 20 + 1 = 533$
P	Motor 1 Parameters	3	P03 (motor 1 rated current): $(3 \times 256) + 3 + 1 = 772$
H	High Performance Functions	4	H11 (deceleration mode): $(4 \times 256) + 11 + 1 = 1036$
A	Motor 2 Parameters	5	A05 (motor 2 torque boost): $(5 \times 256) + 5 + 1 = 1286$
o	Operational Functions	6	o01: $(6 \times 256) + 1 + 1 = 1538$
S	Command Data	7	S05 (frequency command): $(7 \times 256) + 5 + 1 = 1798$
M	Monitor Data 1	8	M09 (output frequency): $(8 \times 256) + 9 + 1 = 2058$
r	Motor 4 Parameters	10	r02 (motor 2 base frequency): $(10 \times 256) + 6 + 1 = 2563$
J	Application Functions 1	13	J03 (PID proportional gain): $(13 \times 256) + 3 + 1 = 3332$
y	Link Functions	14	y98 (bus link function): $(14 \times 256) + 98 + 1 = 3683$
W	Monitor Data 2	15	W32 (PID output): $(15 \times 256) + 32 + 1 = 3873$
X	Alarm Data 1	16	X00 (alarm history / latest): $(16 \times 256) + 0 + 1 = 4097$
Z	Alarm Data 2	17	Z53 (3 <sup>rd</sup> last alarm torque): $(17 \times 256) + 53 + 1 = 4406$
b	Motor 3 Parameters	18	b12 (motor 3 starting frequency): $(18 \times 256) + 12 + 1 = 4621$
d	Application Functions 2	19	d24 (zero speed control): $(19 \times 256) + 24 + 1 = 4889$

## 6.2 Scanned Function Codes

The interface card provides network access to the specified list of function codes contained in the *frenicMegaParam.xml* file located in the "WEB" folder of the interface card's file system. These function codes are constantly being "scanned" by the interface card, which is to say that they are constantly being read and/or written (as applicable), and their current values are therefore mirrored in the interface card's internal memory. Only those function codes specified in the *frenicMegaParam.xml* file will represent meaningful values.

The principle disadvantage of scanned function codes is that write data checking is not available. This means that when the value of a scanned function code is modified via a network protocol or via the web browser's monitor tab, the interface card itself is not able to determine if the new value will be accepted by the inverter (the value may be out-of-range, or the inverter may be in a state in which it will not accept new values being written via communications, etc.) For example, if a write is performed to a scanned command function code with a data value that is out-of-range, the interface card will not generate a corresponding error. However, if end-to-end confirmation of such data writes is required, then the function code can be read over the network at a later time to confirm that the written value "took hold" in the inverter. If the value was not accepted by the inverter, then the unsuccessful write can be observed by reading the current (unchanged) value of the function code during a subsequent network transaction. If the unsuccessful write was initiated via the web browser's monitor tab, then the displayed function code will revert back to its original value automatically.

Accesses to any function code (?00..?99, where "?" is any valid function code group letter from Table 5) will always be successful. Even if an inverter function code corresponding to a given register does not exist in the *frenicMegaParam.xml* file, the interface card still maintains a placeholder location in its internal mirroring memory for that function code. This feature allows for the block access of non-contiguous registers (function codes) as described in section 6.1. Care must be taken to utilize only the function codes that are known to exist and that are also specified in the *frenicMegaParam.xml* file.

## 6.3 Commonly Used Function Codes

For a complete listing of all available function codes, their bit mappings, scaling values, etc., please refer to the Fuji FRENIC-Mega Instruction Manual (Fuji document #INR-S147-1457-E) and the Fuji RS-485 User's Manual (Fuji document #MEH448). As a user convenience, the structures of the commonly-used "Operation command" (function code S06) and "Operation status" (function code M14) are replicated here (refer to Table 6 and Table 7, respectively).

**Table 6: Structure of "Operation command" (Function code S06)**

Data format [14] Operation command

15	14	13	12	11*1	10	9	8	7	6	5	4	3	2	1	0
RST	XR (REV)	XF (FWD)	0	EN	X9	X8	X7	X6	X5	X4	X3	X2	X1	REV	FWD
↑	General-purpose input		Unused	EN terminal	General-purpose input								FWD: Forward command REV: Reverse command		
Alarm reset															

\*1 bit11: The EN terminal is a bit dedicated for monitor and the terminal command cannot be input through communications. (Applicable only with FRN□□G1□□E and FRN□□G1□□A.)

(All bits are turned ON when set to 1.)

(Example) When S06 (operation command) = FWD, X1 = ON

0000 0000 0000 0101<sub>b</sub> = 0005<sub>H</sub> Consequently,

⇒

00 <sub>H</sub>	05 <sub>H</sub>
-----------------	-----------------



**Table 7: Structure of "Operation status" (Function code M14)**

Data format [16] Operation status

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BUSY	0	0	RL	ALM	DEC	ACC	IL	VL	0	NUV	BRK	INT	EXT	REV	FWD

(All bits are turned ON or become active when set to 1.)

Bit	Symbol	Description	Support*1				Bit	Symbol	Description	Support*1			
			Mini	Eco	Multi	MEGA				Mini	Eco	Multi	MEGA
0	FWD	During forward rotation	○	○	○	○	8	IL	During current limiting	○	○	○	○
1	REV	During reverse rotation	○	○	○	○	9	ACC	During acceleration	○	○	○	○
2	EXT	During DC braking (or during pre-exciting)	○	○	○	○	10	DEC	During deceleration	○	○	○	○
3	INT	Inverter shut down	○	○	○	○	11	ALM	Alarm relay (for any fault)	○	○	○	○
4	BRK	During braking (fixed to 0 for FRENIC-Mini)	x	○	○	○	12	RL	Communications effective	○	○	○	○
5	NUV	DC link circuit voltage established (0 = undervoltage)	○	○	○	○	13	0	—	x	x	x	x
6	TL	During torque limiting	x	x	○	○	14	0	—	x	x	x	x
7	VL	During voltage limiting	○	○	○	○	15	BUSY	During function code data writing	○	○	○	○

\*1 The "Support" column indicates whether each inverter type supports the corresponding bit or not. The symbol "○" means the code is supported and the symbol "x" means that the code is not supported (fixed to 0).

## 7 FILE SYSTEM & FIRMWARE

### 7.1 Overview

The interface card's on-board filesystem is used to store files for use by the application firmware. Currently, the application firmware's main use of the filesystem is to store XML-encoded configuration files and the embedded web server. The XML-encoded configuration files dictate the characteristics of the various protocols and features. Each protocol that requires configuration will have its own XML file stored on the filesystem. For easy identification, the filename will begin with the corresponding protocol which it configures. For example, a BACnet/IP configuration file's filename will begin with "bips", and an EtherNet/IP file will begin with "eips".

Whenever the configuration for a specific protocol is completed, it is suggested that a backup copy of the configuration file be downloaded from the unit to a PC. One reason for this is in case it becomes necessary to restore a previous configuration at a later time. Another reason is that it may be desirable to load multiple units with the same configuration, as a downloaded configuration file can be uploaded again to any compatible unit, allowing the user to easily clone multiple units with the same configuration.

Each time the interface card boots up, it will search the filesystem for the configuration files required by the protocols currently operating in the unit. If it does not find a required file, it will create one and initialize it with factory-default values. Therefore, if it is ever desired to reset a protocol's configuration to factory-default values, this can be easily accomplished by simply deleting the appropriate configuration file from the filesystem and rebooting the unit.

Note that the application firmware uses specific filenames for the configuration files. This means that if a file with a different filename is loaded onto the unit, it will be stored correctly, but will not be used by the application firmware. Similarly, if an existing configuration file's filename is changed, then the unit will again create a default configuration file at next boot-up, which will be stored in the filesystem alongside the file with the changed name.

Configuration files are only read by the protocol drivers at unit boot-up. Therefore, if a new configuration file is loaded onto a unit's filesystem, that unit must be rebooted for the configuration file's settings to take effect. Rebooting a unit can be performed by power-cycling the inverter in which the card is installed, or by selecting the "Reboot Device" button in the Finder utility.

The embedded web server is customizable and is located in the "WEB" folder. All web page related items should reside in the "WEB" folder.

Interacting with the filesystem is performed via USB (using a mini-B USB cable) as the interface card enumerates as a standard USB mass storage device "flash drive". Users can interact with the files on the interface card's filesystem in the same manner as though they were traditional files stored on a local or remote PC.

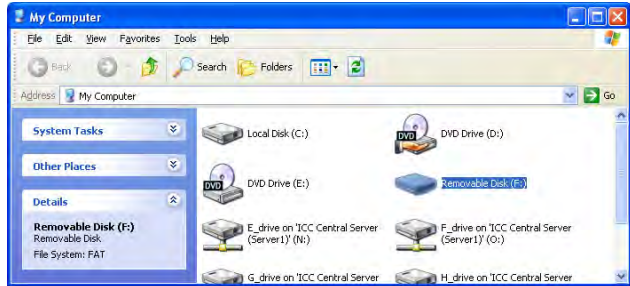
## 7.2 Windows Explorer

To use Microsoft Windows Explorer, first open either “Windows Explorer” or “My Computer”. Refer to Figure 71. Note that the indicated procedure, prompts and capabilities outlined here can vary depending on such factors as the installed operating system and service packs.

The interface card will typically be displayed as a removable medium such as a Removable Disk. Refer to Figure 72.

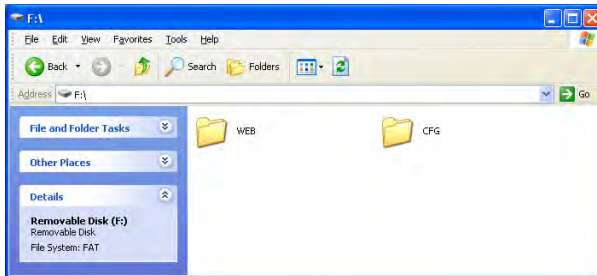


**Figure 71:**  
Accessing  
Windows Explorer



**Figure 72: Removable Disk with Windows Explorer**

Windows Explorer will then display the filesystem's contents (refer to Figure 73.) You can now perform normal file manipulation actions on the available files and folders (cut, copy, paste, open, rename, drag-and-drop transfers etc.) in the same manner as though you were manipulating any traditional file and folder stored on your computer's hard drive.



**Figure 73: File Access via Windows Explorer**

### 7.3 Loading New Application Firmware

The interface card's embedded firmware resides in flash memory that can be updated in the field. Firmware updates may be released for a variety of reasons, such as custom firmware implementations, firmware improvements and added functionality as a result of user requests. Additionally, it may be necessary to load different firmware onto the unit in order to support various protocols.

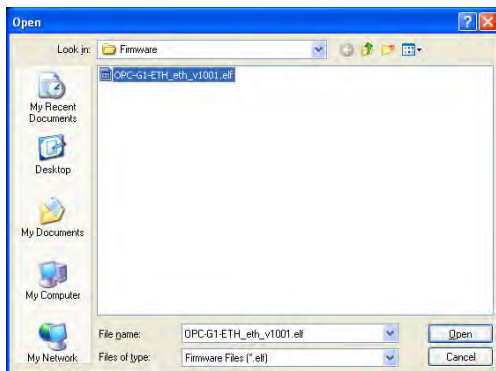
We are continually striving to enhance the functionality and flexibility of our products, and therefore periodically release new embedded firmware to achieve these goals and meet customer requests. Flash firmware files and all related documentation (such as updated user manuals) can be requested from technical support. It is suggested that users first check with technical support, and then periodically afterwards to determine if new firmware has been released and is available to upgrade their units.

Besides the new firmware file, firmware updates require the Fuji Finder and a USB connection as described earlier in this section. To update the firmware, complete the following steps:

1. Navigate to the USB tab of Fuji Finder and click the Update Firmware button.
2. Locate the firmware file (refer to Figure 74) and click Open.
3. Allow the update procedure to complete.

Some notes on uploading new firmware:

- Please be sure to read the firmware release notes and updated user's manual for any important notices, behavior precautions or configuration requirements prior to updating your firmware. For example, upgrading to a new firmware version may affect user-defined configuration files; prior to starting an update procedure always back up your configuration file to a PC for later recovery if necessary.
- The firmware cannot be downloaded from the unit, because it does not reside in the unit's filesystem like configuration files do.
- When the firmware update procedure is initiated, normal operation of the interface card is disabled. After the process has been completed (typically requiring 30-40 seconds), the unit will reset automatically. When the unit boots up again, it will be running the new application firmware, which can be confirmed by observing the version displayed in the Fuji Finder (refer to Figure 21).



**Figure 74: Locate Firmware File**

## 8 PROTOCOL-SPECIFIC INFORMATION

This section will discuss topics that are specific to each of the supported protocols.

### 8.1 Modbus/TCP

#### 8.1.1 Overview

The interface card supports Schneider Electric's Modbus/TCP protocol, release 1.0. The interface is conformance class 0 and partial class 1 and class 2 compliant, and allows up to 8 simultaneous Modbus/TCP client connections (sockets). Other notes of interest are:

- Supported Modbus/TCP functions are indicated in Table 8.

**Table 8: Supported Modbus/TCP Functions**

Function Code	Function	Modbus/TCP Class
1	Read coils	1
2	Read input status	1
3	Read multiple registers	0
4	Read input registers	1
5	Write coil	1
6	Write single register	1
15	Force multiple coils	2
16	Write multiple registers	0

- To calculate the register number for a function code, refer to section 6.1.
- Inverter registers can be addressed as either holding registers (4X references) or input registers (3X references). For example, accessing the output frequency involves accessing holding register 42058 or input register 32058 (i.e. offset 2058). Please note that the most significant digit (4 or 3) is only used as a naming convention for holding registers and input registers. When accessing the output frequency, specify a register value of 2058 (not 42058 or 32058).
- Specific bits within inverter registers can be accessed as either coils (0X references) or discrete inputs (1X references).
- Because the transaction is handled locally within the interface card, write data checking is not available for scanned registers/function codes (refer to section 6.2.) For example, if a write is performed to a register with a data value that is out-of-range of the corresponding function code, no Modbus exception will be immediately returned.
- The "unit identifier" (UI) field of the request packets is ignored.
- The socket timeout time is determined by the "timeout" setting on the web server's "Config" tab (refer to section 5.5.3). This means that if a particular open socket experiences no activity for more than the timeout time setting, then the interface assumes that the client or network has experienced some sort of unexpected problem, and will close that socket.
- Because the socket timeout determination is performed on a per-socket basis, note that a certain degree of caution must be exercised when using the network timeout feature to avoid "nuisance" timeouts from occurring. Specifically, do not perform inadvisable behavior such as sending a request from the master device to the interface, and then closing the socket prior to successfully receiving the unit's response. The reason for this is because the interface will then experience an error when attempting to respond via the now-closed socket, which will immediately trigger the timeout action. Always be sure to manage socket life cycles "gracefully", and do not abandon outstanding requests.
- If a socket timeout occurs (regardless of whether it was due to a communication lapse or abnormal socket error), the driver will trigger a timeout event as described in section 5.5.3.

- If the drive is configured to cause a trip (section 3.2), a Modbus TCP socket must first be successfully established before the trip can be cleared.

### 8.1.2 Coil & Discrete Input Mappings

The Modbus/TCP driver provides read/write support for coils (0X references) and read-only support for discrete inputs (1X references). These will collectively be referred to from here on out as simply “discretes”. Accessing discretes does not reference any new physical data: discretes are simply indexes into various bits of existing registers. What this means is that when a discrete is accessed, that discrete is resolved by the interface into a specific register, and a specific bit within that register. The pattern of discrete-to-register/bit relationships can be described as follows:

Discrete 1...16 map to register #1, bit0...bit15 (bit0=LSB, bit15=MSB)

Discrete 17...32 map to register #2, bit0...bit15, and so on.

Arithmetically, the discrete-to-register/bit relationship can be described as follows: For any given discrete, the register in which that discrete resides can be determined by Equation 3.

$$register = \left\lfloor \frac{discrete + 15}{16} \right\rfloor \quad \text{Equation 3}$$

Where the bracket symbols “ $\lfloor \rfloor$ ” indicate the “floor” function, which means that any fractional result (or “remainder”) is to be discarded, with only the integer value being retained.

Also, for any given discrete, the targeted bit in the register in which that discrete resides can be determined by Equation 4.

$$bit = (discrete - 1) \% 16 \quad \text{Equation 4}$$

Where “discrete”  $\in [1...65535]$ , “bit”  $\in [0...15]$ , and “%” is the modulus operator, which means that any fractional result (or “remainder”) is to be retained, with the integer value being discarded (i.e. it is the opposite of the “floor” function).

For clarity, let's use Equation 3 and Equation 4 in a calculation example. Say, for instance, that we are going to read coil #34. Using Equation 3, we can determine that coil #34 resides in register #3, as  $\lfloor 3.0625 \rfloor = \lfloor 3 \text{ r } 1 \rfloor = 3$ . Then, using Equation 4, we can determine that the bit within register #3 that coil #34 targets is  $(34-1)\%16 = 1$ , as  $33\%16 = \text{mod}(2 \text{ r } 1) = 1$ . Therefore, reading coil #34 will return the value of register #3, bit #1.

## 8.2 EtherNet/IP

---

### 8.2.1 Overview

The EtherNet/IP protocol is an application-level protocol implemented on top of the Ethernet TCP/IP and UDP/IP layers. It shares its object model with ControlNet and DeviceNet through the Common Industrial Protocol (CIP). This protocol allows the transfer of data and I/O over Ethernet.

EtherNet/IP incorporates both the TCP and UDP layers of Ethernet in the transmission of data. Because TCP/IP is a point-to-point topology, EtherNet/IP uses this layer only for explicit messaging; i.e. those messages in which the data field carries both protocol information and instructions for service performance. With explicit messaging, nodes must interpret each message, execute the requested task and generate responses. These types of messages can be used to transmit configuration, control and monitor data.

The UDP/IP protocol layer, which has the ability to multicast, is used for implicit (I/O) messaging. With I/O messaging, the data field contains only real-time I/O data; no protocol information is sent because the meaning of the data is pre-defined at the time the connection is established, which in turn minimizes the processing time of the node during run-time. I/O messages are short and have low overhead, which allows for the time-critical performance needed by controllers.

The interface card supports both explicit and I/O messaging. Further, two different types of I/O messaging are supported. One type (invoked when the client opens a connection to the interface using assembly instances 20 & 70 or 21 & 71) is included with the implementation of the AC/DC drive profile, and requires no user configuration. The other type, however, is entirely user-configurable, and is invoked when the client opens a connection to the interface using assembly instances 100 and 150.

The following sections demonstrate specific examples of how to use EtherNet/IP to transfer data between the inverter and Allen-Bradley Logix-brand PLCs.

#### Some other notes of interest are:

- The interface card supports the EtherNet/IP protocol, as administered by the Open DeviceNet Vendor Association (ODVA).
- This product has been self-tested by Fuji Electric and found to comply with ODVA EtherNet/IP Conformance Test Software Version A-5.
- I/O connection sizes for assembly instances 100 and 150 are adjustable between 0 and 64 bytes (32 function codes max @ 2 bytes per function code = 64 bytes). Because function codes are 16-bit elements, however, connection sizes cannot be odd (i.e. 1, 3, 5 etc.)
- The interface card's product type code is 2 (AC Drive.)
- Supports unconnected messages (UCMM), and up to 16 simultaneous class 1 (I/O) or class 3 (explicit) connections.
- Assembly instances 100 and 150: if a function code entry in the consumed data configuration array is blank, then any consumed data that corresponds to that location will be ignored. Conversely, if a function code entry in the produced data configuration array is blank, then any produced data that corresponds to that location will be a default value of 0. Refer to section 5.6.4 for further information on the data configuration arrays.
- Class 1 implicit I/O supports both multicast and point-to-point (unicast) when producing data in the T→O direction.
- Point-to-point class 1 connected messages will be produced targeting the IP address of the device that instantiated the connection, UDP port 0x08AE (UDP port 2222).
- If a class 1 point-to-point connection is established in the (T→O) direction, no more class 1 connections can be established.
- If a class 1 connection's consuming half (O→T) times out, then the producing half (T→O) will also time-out and will stop producing.
- If a class 1 or class 3 connection timeout occurs, the driver will trigger a timeout event as described in section 5.5.3. The timeout value is dictated by the scanner/client and is at a minimum, four times the scan rate (Requested Packet Interval) for class 1. The typical timeout value for class 3 messaging is usually much larger and is also dictated by the scanner/client.
- If the drive is configured to cause a trip (section 3.2), a class 1 or class 3 connection must first be successfully established before the trip can be cleared.
- Convenient and simple interface using Add-On Instructions for RSLogix 5000 (version 16 and up).

## 8.2.2 ODVA AC/DC Drive Profile

The interface card supports the ODVA AC/DC drive profile. No special EtherNet/IP configuration of the interface card is required when using the AC/DC drive profile: all that is needed is that the controller must target either assembly instances 20 & 70 or 21 & 71 in its connection parameters.

The AC/DC drive profile implementation provides support for several required CIP objects, which are specified in Table 9. While the various supported attributes of all of these objects are accessible via explicit messaging, the main intent of using the AC/DC drive profile is to interact with the predefined input and output assembly instances via an I/O connection. The structure of these assembly instances is defined by the EtherNet/IP specification in order to engender interoperability among different vendor's products. This section will focus primarily on the format of the AC/DC drive profile I/O assemblies supported by the interface card, and the inverter data which their various constituent elements map to.

**Table 9: AC/DC Drive Profile-Related Objects**

Class Code	Object Name
0x04	Assembly Object
0x28	Motor Data Object
0x29	Control Supervisor Object
0x2A	AC Drive Object

## Output Instances

**Table 10: Output Instances 20 and 21 Detail**

Instance	Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
20	0						Fault Reset		Run Fwd
	1								
	2	Speed Reference (Low Byte)							
	3	Speed Reference (High Byte)							
21	0		NetRef	NetCtrl			Fault Reset	Run Rev	Run Fwd
	1								
	2	Speed Reference (Low Byte)							
	3	Speed Reference (High Byte)							

## Output Instance Mapping Detail

**Run Fwd:** forward rotation command (0=forward rotation off, 1=forward rotation on). Maps to inverter function code S06, bit 0 (function code S06 / operation command word, FWD bit).

**Run Rev:** reverse rotation command (0=reverse rotation off, 1=reverse rotation on). Maps to inverter function code S06, bit 1 (function code S06 / operation command word, REV bit).

**Fault Reset:** Inverter reset command (0=no action, 0→1 rising edge=reset). Maps to inverter function code S06, bit 15 (function code S06 / operation command word, RST bit).



**NetCtrl:** Not used (value is ignored).

**NetRef:** Not used (value is ignored).

**Speed Reference:** Inverter speed reference in RPM. Maps to function code S05 (frequency command). The speed reference component of the AC/DC drive profile output instances is always in units of RPM. Therefore, the interface card applies the RPM-to-Hz conversion indicated in Equation 5 in order to determine the appropriate frequency command value (in units of Hz) to be written to function code S05.

$$Hz = \frac{RPM \times \text{number of motor poles}}{120} \quad \text{Equation 5}$$

The “number of motor poles” term which appears in the numerator of Equation 5 is obtained from the setting of inverter function code P01 (Motor number of poles). Note that the value of P01 is read by the interface card only at boot-up, so if the value of this function code is changed, then the interface card must be rebooted in order for it to read the new value from the inverter.

## Input Instances

**Table 11: Input Instances 70 and 71 Detail**

Instance	Byte	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
70	0						Running1		Faulted
	1								
	2	Speed Actual (Low Byte)							
	3	Speed Actual (High Byte)							
71	0	At Reference	Ref from Net	Ctrl From Net	Ready	Running2 (REV)	Running1 (FWD)	Warning	Faulted
	1	Drive State							
	2	Speed Actual (Low Byte)							
	3	Speed Actual (High Byte)							

## Input Instance Mapping Detail

**Faulted:** Inverter fault signal (0=not faulted, 1=faulted). Maps to function code M14, bit 11 (operation status word, ALM bit).

**Warning:** This bit is not used (it is always 0).

**Running1 (FWD):** Running forward status signal (0=not running forward, 1=running forward). Maps to function code M14, bit 0 (operation status word, FWD bit).

**Running2 (REV):** Running reverse status signal (0=not running reverse, 1=running reverse). Maps to function code M14, bit 1 (operation status word, REV bit).

**Ready:** Inverter ready signal (0=not ready, 1=ready). The Ready bit will be 1 whenever the Drive State attribute (see below) is in the Ready, Enabled or Stopping state.

**CtrlFromNet:** This bit is not used (it is always 0).

**RefFromNet:** This bit is not used (it is always 0).

**AtReference:** Up-to-speed signal (0=not up-to-speed, 1=up-to-speed). Set to 1 if the inverter is running (either Running1 = 1 or Running2 = 1) and both the ACC bit (bit #9) and DEC bit (bit #10) in the operation status word (function code M14) are 0.

**Drive State:** Indicates the current state of the Control Supervisor Object state machine. Refer to the ODVA EtherNet/IP specification (object library) for detailed information on the Control Supervisor Object state machine.

**Speed Actual:** Inverter operating speed in RPM. Maps to function code M09 (output frequency). The speed actual component of the AC/DC drive profile input instances is always in units of RPM. Therefore, the interface card applies the Hz-to-RPM conversion indicated in Equation 6 in order to determine the appropriate operating speed (in units of RPM) to be written to the network.

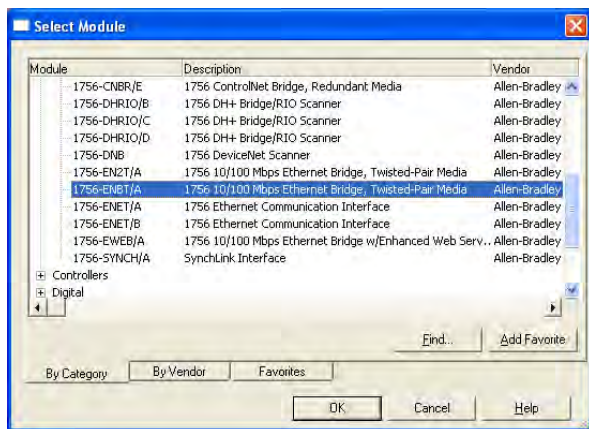
$$RPM = \frac{Hz \times 120}{\text{number of motor poles}} \quad \text{Equation 6}$$

The “number of motor poles” term which appears in the denominator of Equation 6 is obtained from the setting of inverter function code P01 (Motor number of poles). Note that the value of P01 is read by the interface card only at boot-up, so if the value of this function code is changed, then the interface card must be rebooted in order for it to read the new value from the inverter.

### 8.2.3 ControlLogix Examples: Setup

This section will demonstrate how to initially setup a ControlLogix PLC (such as a 1756-L61) coupled with a 1756-ENBT/A communication interface (adjust this procedure according to your specific equipment). Later sections will provide specific read/write examples using this configuration with I/O or explicit messaging.

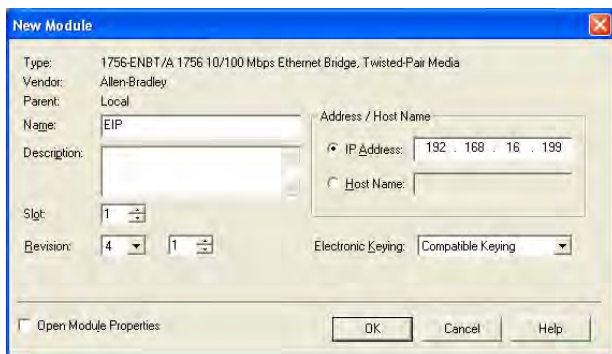
- 1) Run RSLogix 5000, and create a new configuration.
- 2) To add a 1756-ENBT/A to your I/O configuration, first switch to offline mode.
- 3) Right click on the I/O Configuration node in the controller organizer view and choose “New Module...”
- 4) The “Select Module” window will open.
- 5) Under “Communications”, select “1756-ENBT/A”, and click OK. Refer to Figure 75.



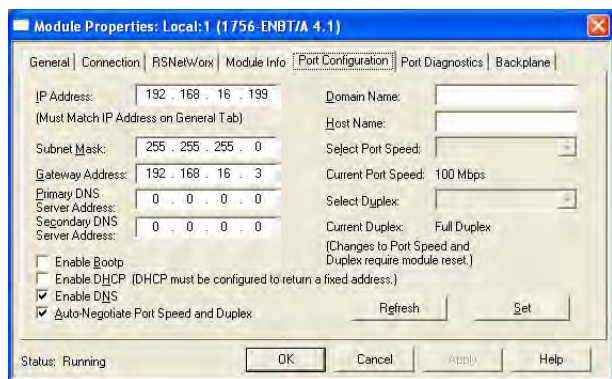
**Figure 75: Adding a New 1756-ENBT/A Module**

- 6) The “New Module” window will open. Refer to Figure 76.
- 7) Assign the Ethernet module a name (we will use “EIP”) and an IP address, deselect “Open Module Properties”, and click OK.
- 8) Download the configuration.

- 9) Switch to online mode. Right click on the 1756-ENBT/A module in the I/O Configuration and choose "Properties".
- 10) Select the Port Configuration tab from the Module Properties dialog box.
- 11) Confirm that the IP Address, Subnet Mask and Gateway Address fields are configured correctly. The IP Address must match the IP Address entered when the new module was first created. Refer to Figure 77.



**Figure 76: Configuring the New Module**



**Figure 77: Confirming the Module's Properties**

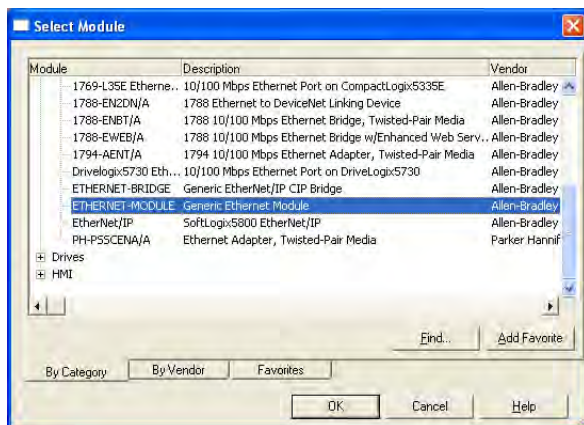
- 12) Apply any changes to the settings using the "Set" button.
- 13) You should now be able to confirm that the 1756-ENBT/A module is configured properly by (for example) opening the module's web interface in a web browser.

#### **8.2.4 ControlLogix Example: I/O Messaging**

This section will demonstrate how to setup and use an EtherNet/IP I/O connection via vendor-specific assembly instances 100 & 150. EtherNet/IP I/O messaging allows the inverter's function codes to be directly mapped into tags in the ControlLogix PLC. Once an I/O connection is established, it is automatically synchronized at an interval defined by the Requested Packet Interval (RPI).

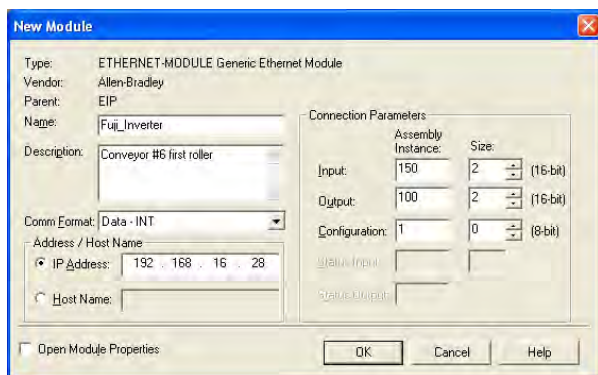
- 1) Switch to offline mode.

- 2) Right click on the 1756-ENBT/A node under the I/O Configuration in the controller organizer view and choose "New Module..."
- 3) Choose "Generic Ethernet Module" in the Select Module dialog box and click "OK". Refer to Figure 78.



**Figure 78: Adding a New Generic Ethernet Module**

- 4) The module properties dialog box will open (refer to Figure 79). Enter a Name and Description which will allow easy identification of the inverter on the network (the tags created in RSLogix 5000 will be derived from this Name). Because all inverter data is stored as 16-bit function code values, change the "Comm Format" selection to "Data-INT". Enter the IP address of the targeted interface card.



**Figure 79: Interface Card Module Properties**

In the "Connection Parameters" portion of the dialog box, enter the following information:

**Input:** The Input Assembly is the collection of monitor data that is produced by the interface card and is received as an input to the PLC. Its structure is defined by the Produced Data Configuration as described in section 5.6.4. The Input Assembly Instance must be set to 150 when connecting to the vendor-specific I/O assembly instances (or 70/71 when using the ODVA AC/DC drive profile), and the size must be set to the number of 16-bit function codes that we wish to receive from the interface card. For the purposes of this example, we are assuming that the produced configuration

array is defined as shown in Figure 43, with two relevant function codes (M14 and M09). We therefore set the Input Size to 2.

**Output:** The Output Assembly is the collection of command & configuration data that is sent as an output from the PLC and consumed by the interface card. Its structure is defined by the Consumed Data Configuration as described in section 5.6.4. The Output Assembly Instance must be set to 100 when connecting to the vendor-specific I/O assembly instances (or 20/21 when using the ODVA AC/DC drive profile), and the size must be set to the number of 16-bit function codes that we wish to send to the interface card. For the purposes of this example, we are assuming that the consumed configuration array is defined as shown in Figure 43, with two relevant function codes (S06 and S05). We therefore set the Output Size to 2.

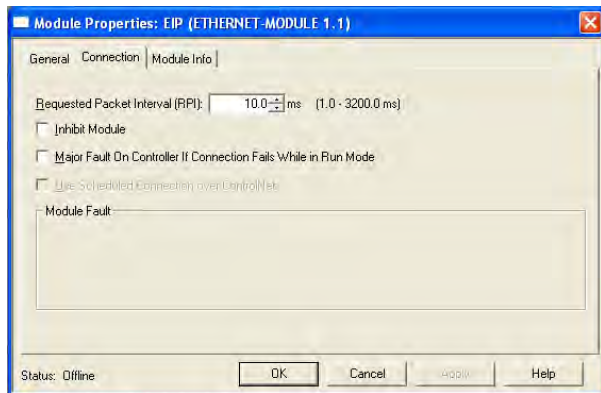
**Configuration:** The Configuration Assembly Instance is unused, and its instance number and size are therefore irrelevant (you can just enter "1" and "0", respectively).

When done, click "OK".

- 5) You should now see the new module (named "ETHERNET-MODULE Fuji\_Inverter") in the 1756-ENBT/A branch under the I/O Configuration in the controller organizer view. Right click on this new module, choose "Properties", and select the Connection tab. Refer to Figure 80.

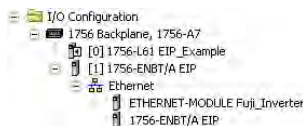
Confirm the setting of the Requested Packet Interval (RPI). The RPI defines the amount of time (in milliseconds) between data exchanges across an I/O connection. The smallest RPI supported by the interface card is 2ms. However, the lowest recommended RPI is 10ms.

When done, click "OK".



**Figure 80: Interface Card Module Properties Connection Tab**

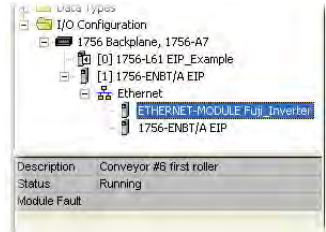
- 6) After adding the I/O Module to the configuration, the full I/O Configuration tree should appear similar to Figure 81.
- 7) Switch to online mode and download the project to the PLC. Verify that the newly-added inverter is available and operating correctly by observing any indications shown on the inverter's icon. When the inverter's icon is selected, its status and any available error messages will be displayed in the area below the project tree. Refer to Figure 82. Also confirm that the interface card's "Network Status" LED should be solid green, indicating an "online/connected" state.



**Figure 81: I/O Configuration Tree**

- 8) By double-clicking "Controller Tags" in the project tree, it is possible to view the newly-added tags. Refer to Figure 83. The Fuji\_Inverter:C configuration tag is unused, the Fuji\_Inverter:I tag allows viewing of the input data, and the Fuji\_Inverter:O tag allows modification of the output data. These tags will be synchronized with the inverter at whatever rate was established for the module's RPI.

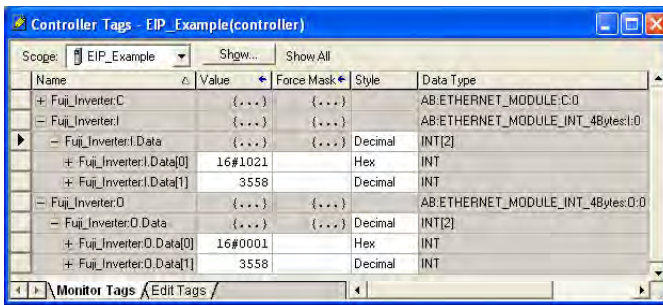
We can directly interact with these tags in order to control and monitor the inverter. In Figure 83, for example, we can see that the first 16-bit word of output data (Fuji\_Inverter.O.Data[0]) has been set to a hexadecimal value of 0x0001. Referring back to Figure 43, we can see that the first element of the consumed data configuration references function code S06, which is the inverter's operation command register. A value of 0x0001, therefore, means that the FWD (run forward) bit has been turned ON.



**Figure 82: Online Module Status**

Similarly, we can see that the second 16-bit word of output data (Fuji\_Inverter.O.Data[1]) has been set to a decimal value of 3558. Once again referring back to Figure 43, we can see that the second element of the consumed data configuration references function code S05, which is the inverter's frequency command register. A value of 3558, therefore, equates to a frequency command of 35.58Hz.

The input data from the inverter shows similar expected results. Values of 0x1021 and 3558 corresponding to M14 (status register) and M09 (output frequency), respectively, are consistent with the inverter running at the parameters commanded by the output tag.

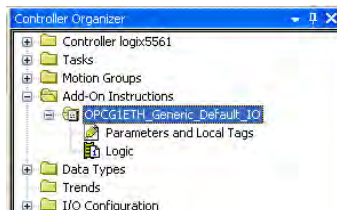


**Figure 83: Controller Tags for I/O Access**

### 8.2.5 ControlLogix Example: Generic Default I/O Add-On Instruction

The generic default I/O add-on instruction is a simple interface to command and monitor the inverter. It is based on the vendor-specific assembly instances 100 & 150 and the default produce and consume data configuration arrays (refer to section 5.6.4).

- 1) Complete all the steps in section 8.2.4.
- 2) Right click on "Add-On Instructions" in the controller organizer view and select "Import Add-On Instruction". Browse and import the generic default I/O add-on instruction. Refer to Figure 84.

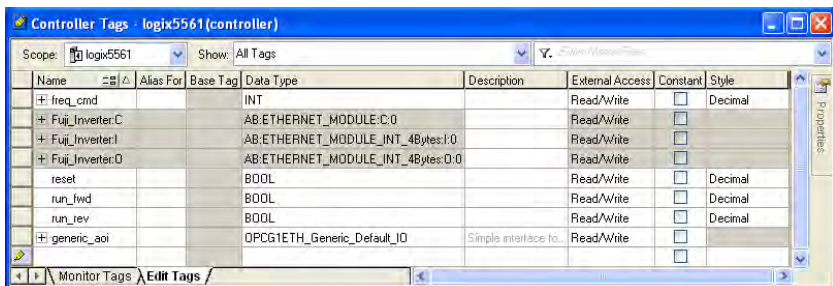


**Figure 84: Generic Default IO Add-On Instruction**

- 3) Double click "Controller Tags" in the controller organizer view and select the "Edit Tags" tab at the bottom.

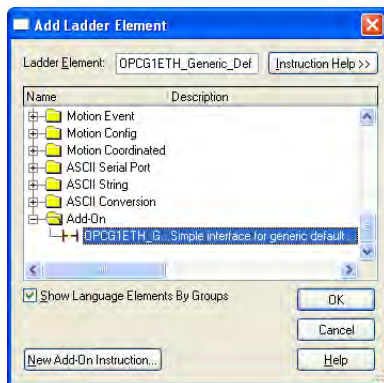


- 4) Create the tags in Figure 85.



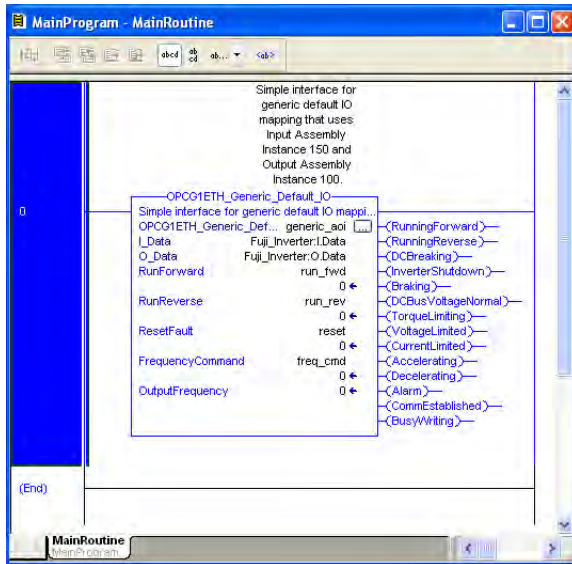
**Figure 85: Create Generic Default AOI Tags**

- 5) Double click "MainRoutine" under Tasks ...MainTask ...MainProgram in the controller organizer view.
- 6) Right click on the first ladder logic rung in the MainRoutine window and select "Add Ladder Element..."
- 7) The "Add Ladder Element" window appears.
- 8) Select the generic default I/O add-on instruction in the Add-On folder. Refer to Figure 86.



**Figure 86: Add Generic Default Add-On Instruction**

- 9) Click OK.
- 10) Edit the add-on instruction according to Figure 87.



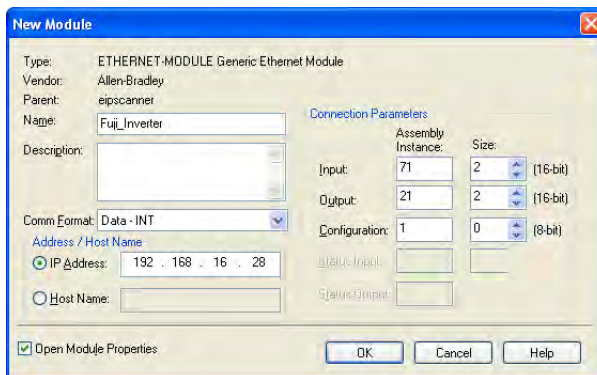
**Figure 87: Configure Generic Default AOI**

- 11) The program is now complete.
- 12) Save, download and run the program.

### 8.2.6 ControlLogix Example: AC/DC Drive Profile Add-On Instruction

The AC/DC drive profile add-on instruction is a simple interface to command and monitor the inverter. It is based on the assembly instances 21 & 71.

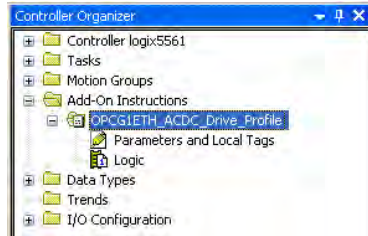
- 1) Complete all the steps in section 8.2.4. Please note that the Assembly Input Instance must be changed to 71 and the Assembly Output Instance must be changed to 21. Refer to Figure 88.



**Figure 88: AC/DC Drive Profile Generic Ethernet Module Configuration**

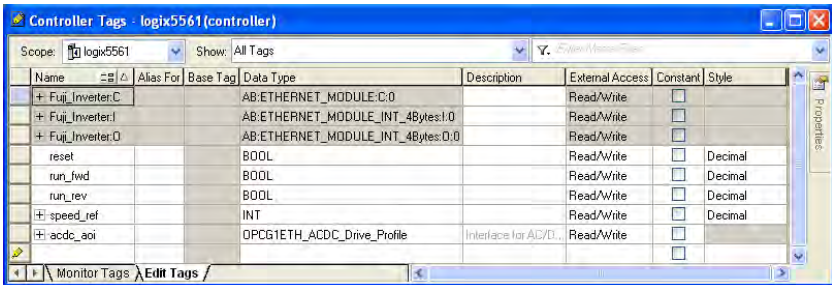
- 2) Right click on "Add-On Instructions" in the controller organizer view and select "Import>Add-On Instruction". Browse and import the AC/DC drive profile add-on instruction. Refer to Figure 89.





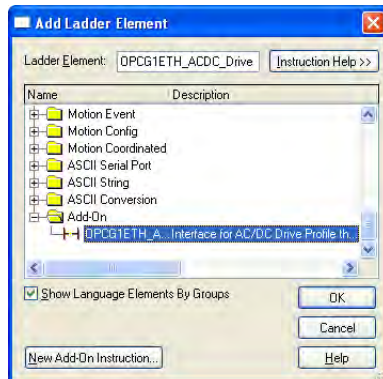
**Figure 89: AC/DC Drive Profile Add-On Instruction**

- 3) Double click "Controller Tags" in the controller organizer view and select the "Edit Tags" tab at the bottom.
- 4) Create the tags in Figure 90.



**Figure 90: Create AC/DC Drive Profile AOI Tags**

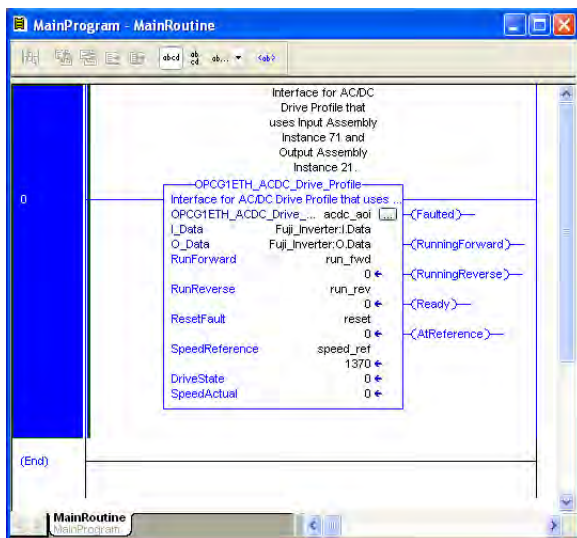
- 5) Double click "MainRoutine" under Tasks ...MainTask ...MainProgram in the controller organizer view.
- 6) Right click on the first ladder logic rung in the MainRoutine window and select "Add Ladder Element..."
- 7) The "Add Ladder Element" window appears.
- 8) Select the AC/DC drive profile add-on instruction in the Add-On folder. Refer to Figure 91.



**Figure 91: Add AC/DC Drive Profile Add-On Instruction**

- 9) Click OK.

- 10) Edit the add-on instruction according to Figure 92.



**Figure 92: Configure AC/DC Drive Profile AOI**

- 11) The program is now complete.
- 12) Save, download and run the program.

### 8.2.7 Explicit Messaging Tag Reference

When class 3 (explicit messaging) connections are used, function code contents are read from and written to the interface card via EtherNet/IP by reference to "tag names". The "tag name" is essentially the ASCII representation of the function code itself. Tags are read via the EtherNet/IP "data table read" service, and tags are written via the EtherNet/IP "data table write" service.

Any given scanned or non-scanned function code can be accessed with its own unique tag name, or an array tag can be used to access a group of function codes with one PLC instruction. Tag names are generated according to the following structure:

**[function code group][function code offset]**

#### Where

**[function code group]** is a 1-character field, and is the ASCII character for the function code's group. Refer to Table 5.

**[function code offset]** is a 2-character field corresponding to the function code offset. If the offset is less than 10, it must be pre-pended by 0. Valid offsets are "00" to "99".

#### Examples

Write "acceleration time 1" .....	F07
Write "frequency command" .....	S05
Read "operation status" .....	M14
Read "output power" .....	W22

To read data from the interface card, the application PLC program must reference a "source element" from which to start reading and the "number of elements" to read. The "source element" will be a tag name constructed according to the naming convention shown above. The "source element" can be either a base tag (such as "M01"), or an offset from a base tag (such as "M01[8]", which starts at function code M01 + 8 = function code M09, the inverter's output frequency register).

In a similar manner, to write data to the interface card, the application PLC program must reference a "destination element" to which to start writing and the "number of elements" to write. Again, the "destination element" will be a tag name constructed according to the naming convention shown above.

Whether reading or writing, the "number of elements" can be any quantity of function codes from 1 to the maximum allowable length.

### 8.2.8 ControlLogix Explicit Messaging Example: Read a Function Code Block

This example program will show how to continuously read a block of function codes from the inverter with a single MSG instruction. Only one read request is outstanding at any given time.

#### 1) Create new Tags.

- a) Double click "Controller Tags" in the controller organizer view.

- b) The "Controller Tags" window appears. Refer to Figure 93.

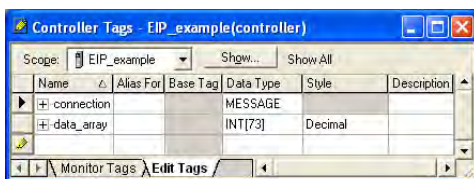


Figure 93: Create New Tags

- c) Select the "Edit Tags" tab at the bottom.
- d) Create a new tag by entering "connection" in the first blank Name field, and change its Data Type to "MESSAGE". This tag will contain configuration information for the MSG instruction.
- e) Select the "Monitor Tags" tab. Expand the "connection" tag by clicking on the "+" sign next to the tag name. Scroll down to the connection.UnconnectedTimeout field and change its value from the default 30000000 (30s in 1uS increments) to 1000000 (1s). This value determines how long to wait before timing out and retransmitting a connection request if a connection failure occurs. Refer to Figure 94.

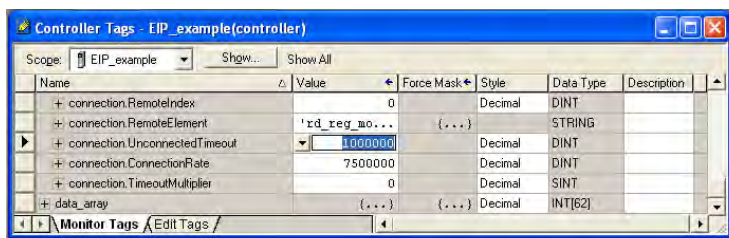


Figure 94: Reduce the UnconnectedTimeout Value

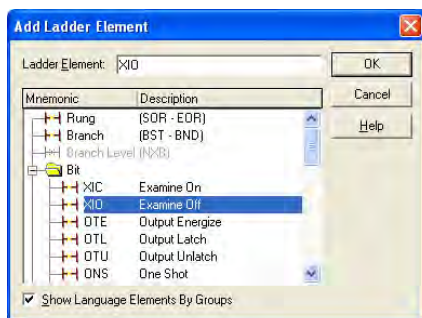
- f) Collapse the "connection" tag again by clicking on the "-" sign next to the tag name.
- g) Select the "Edit Tags" tab again. Create another new tag by entering "data\_array" in the next blank Name field, and change its Data Type by typing in "INT[73]" in the Data Type field. This tag is an array of INTs that will be able to hold up to 73 16-bit function codes from the inverter. Always make sure that the destination tag size is large enough to hold all elements to be read.

- 2) **Add a MSG instruction to the main program.**
  - a) Double click "MainRoutine" under Tasks ...MainTask ...MainProgram in the controller organizer view.
  - b) Right click on the first ladder logic rung in the MainRoutine window and select "Add Ladder Element..."
  - c) The "Add Ladder Element" window appears.
  - d) Select the "MSG" instruction in the Input/Output folder. Refer to Figure 95.
  - e) Click OK.



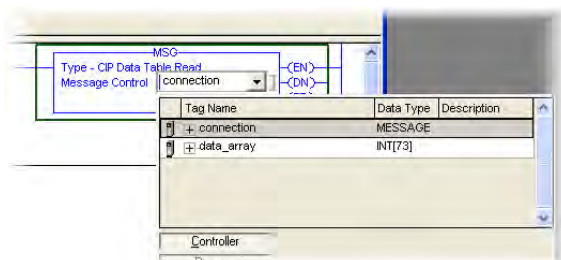
**Figure 95: Adding a MSG Instruction**

- 3) **Add an XIO element to the main program.**
  - a) Right click on the ladder logic rung containing the MSG instruction in the MainRoutine window and select "Add Ladder Element..." again.
  - b) The "Add Ladder Element" window appears.
  - c) Select the "XIO" element in the Bit folder. Refer to Figure 96.
  - d) Click OK.



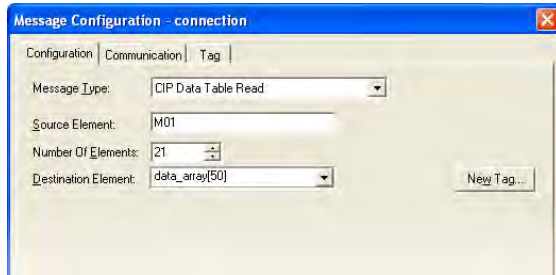
**Figure 96: Adding an XIO Element**

- 4) **Configure the MSG instruction.**
  - a) Edit the "Message Control" field on the MSG instruction to use the previously-created "connection" tag. Refer to Figure 97.



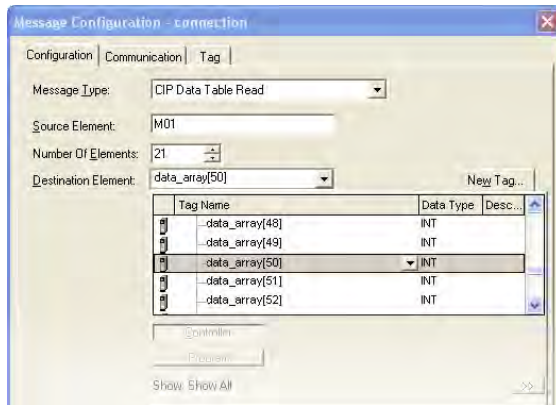
**Figure 97: MSG Instruction Tag Assignment**

- b) Click the message configuration button ("...") in the MSG instruction. The "Message Configuration" window will open. Refer to Figure 98.



**Figure 98: MSG Instruction Configuration**

- c) "Configuration" tab settings:
- Change the "Message Type" to "CIP Data Table Read".
  - In the "Source Element" field, enter the read tag you wish to access (refer to section 8.2.5.) In this example, we will be reading a total of 21 function codes beginning at function code M01 (per-unit frequency reference – final command).
  - Enter the Number Of Elements to read. In this example, we will read 21 function codes.
  - For the Destination Element, either directly type in "data\_array[50]", or select element #50 in the data\_array tag via the drop-down box (refer to Figure 99). The destination could be any offset in the data\_array tag, as long as the offset plus the Number Of Elements (21) does not exceed the tag's defined size (73).

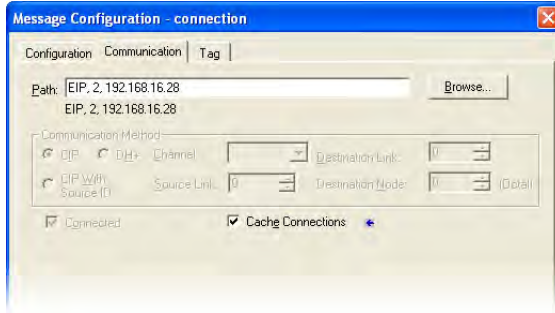


**Figure 99: Selecting the Destination Element**

- d) "Communication" tab settings (refer to Figure 100):
- Enter the Path to the interface card. A typical path is formatted as "*Local\_ENB,2,target\_IP\_address*", where:
    - Local\_ENB* is the name of the 1756-ENBx module in the local chassis (we named ours "EIP" in section 8.2.3),
    - 2* is the Ethernet port of the 1756-ENBx module in the local chassis, and
    - target\_IP\_address* is the IP address of the target node.

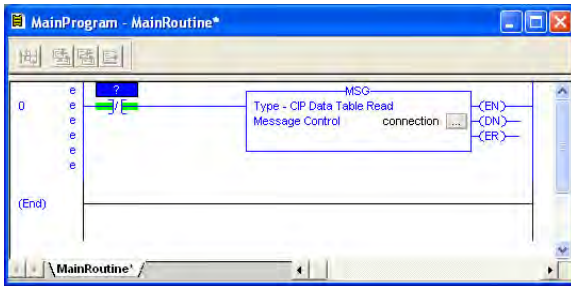
In our example, this path would be entered as "EIP,2,192.168.16.28".
  - If "Cache Connections" is enabled (checked), the connection remains open after transmission. If disabled (unchecked), the connection is opened before and closed

after every transmission. For efficiency, it is recommended to enable “Cache Connections”.



**Figure 100: Setting the Communication Path**

- e) Click “OK” to close the MSG Configuration dialog. At this stage, MainRoutine should look like Figure 101.



**Figure 101: MainRoutine**

#### 5) Assign a tag to the XIO element.

- a) Double-click on the XIO element located to the left of the MSG block. In the drop-down box, double-click on the “connection.EN” field. Refer to Figure 102. This configuration causes the MSG instruction to automatically retrigger itself when it completes. While this is acceptable for the purposes of this example, it can produce high network utilization. In actual practice, it may be desirable to incorporate additional logic elements to allow triggering the MSG instruction at a specific rate or under specific conditions.

#### 6) The program is now complete. Refer to Figure 103.

#### 7) Save, download and run the program.

- a) To view the values of the function codes being read from the interface card, double-click “Controller Tags” in the controller organizer view.
- b) Select the “Monitor Tags” tab.
- c) Expand the data\_array tag. Refer to Figure 104.
- d) 21 function code values starting at function code M01 are being continuously read from the interface card and placed in the 21 sequential offsets of data\_array starting at the 50<sup>th</sup> offset (data\_array[50]). In Figure 104, we can see that data\_array[50] (function code M01 / per-unit frequency reference – final command) has a value of 11860 (11860/20000 = 59.3% of max frequency), data\_array[58] (function code M09 / output frequency) has a value of 3558 (35.58Hz), etc.



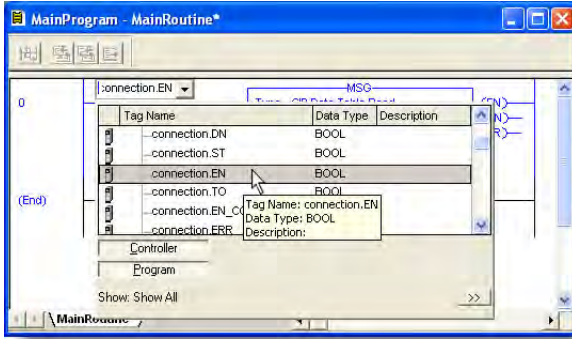


Figure 102: Configure XIO Element

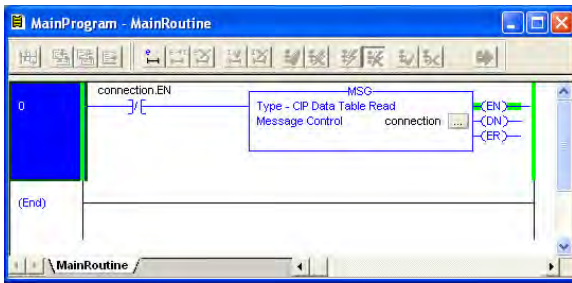
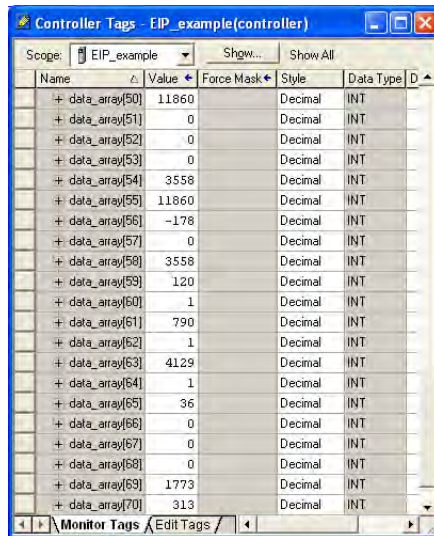


Figure 103: Complete Program



Name	Value	Force Mask	Style	Data Type	D
+ data_array[50]	11860		Decimal	INT	
+ data_array[51]	0		Decimal	INT	
+ data_array[52]	0		Decimal	INT	
+ data_array[53]	0		Decimal	INT	
+ data_array[54]	3558		Decimal	INT	
+ data_array[55]	11860		Decimal	INT	
+ data_array[56]	-178		Decimal	INT	
+ data_array[57]	0		Decimal	INT	
+ data_array[58]	3558		Decimal	INT	
+ data_array[59]	120		Decimal	INT	
+ data_array[60]	1		Decimal	INT	
+ data_array[61]	790		Decimal	INT	
+ data_array[62]	1		Decimal	INT	
+ data_array[63]	4129		Decimal	INT	
+ data_array[64]	1		Decimal	INT	
+ data_array[65]	36		Decimal	INT	
+ data_array[66]	0		Decimal	INT	
+ data_array[67]	0		Decimal	INT	
+ data_array[68]	0		Decimal	INT	
+ data_array[69]	1773		Decimal	INT	
+ data_array[70]	313		Decimal	INT	

Figure 104: Viewing the Function Code Values

### 8.2.9 ControlLogix Explicit Messaging Example: Read a Single Function Code

The configuration and execution for reading a single function code is in general identical to that required for reading a block of function codes as detailed in section 8.2.8. The only difference is in the configuration of the MSG instruction. Figure 105 shows an example MSG instruction's Configuration tab, which will read a single tag (function code M14, the inverter's operation status register) and place it in the first element (offset 0) of data\_array.

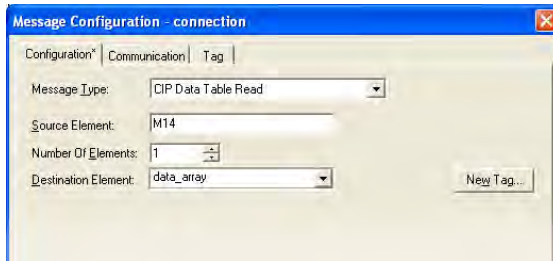


Figure 105: Read the Inverter's Operation Status Register

### 8.2.10 ControlLogix Explicit Messaging Example: Multiple MSG Instructions

At times, reading from different groups of function codes may be necessary. For example, a specific application may require access to some function codes in both the Monitor Data 1 and Monitor Data 2 function code groups. To accomplish this task, multiple MSG instructions will need to be implemented in the PLC program.

The configuration and execution for implementing multiple MSG instructions is in general identical to that required for implementing just one MSG instruction. Each MSG instruction will require its own message controller tag. In the case of read MSG instructions, more than one instruction may use the same Destination Element tag, but the storage locations must not overlap. Figure 106 shows an example of two MSG instructions, each accessing different read tags. It is evident from this logic that "rd\_connection" and "rd\_connection2" are the two independent message controller tags created for these instructions.

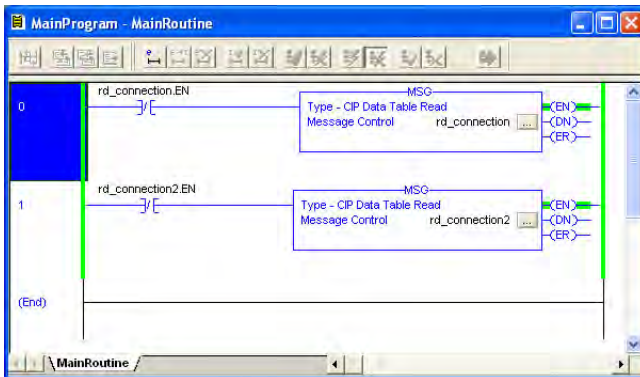


Figure 106: Reading Via Multiple MSG Instructions



### 8.2.11 ControlLogix Explicit Messaging Example: Reading and Writing

Often times, applications may need to both read data from and write data to the inverter. At a minimum, this will require two MSG instructions and two message controller tags. Figure 107 shows an example of three MSG instructions, one for reading and two for writing (the inverter's frequency command and operation command word). The only item of note that differentiates this example from the multiple-read example in section 8.2.10 is the addition of the en\_xx\_wr XIC elements. The reason for the addition of these elements is that while reading from a remote device is often continuously performed (monitoring), data is typically written to the remote device only when necessary (i.e. when the value to write has changed). This conserves both network bandwidth and potentially EEPROM lifespans on the target device. The en\_xx\_wr elements in this example, therefore, would typically be replaced in an actual application program by user-provided logic that controls the conditions under which write operations would be performed.

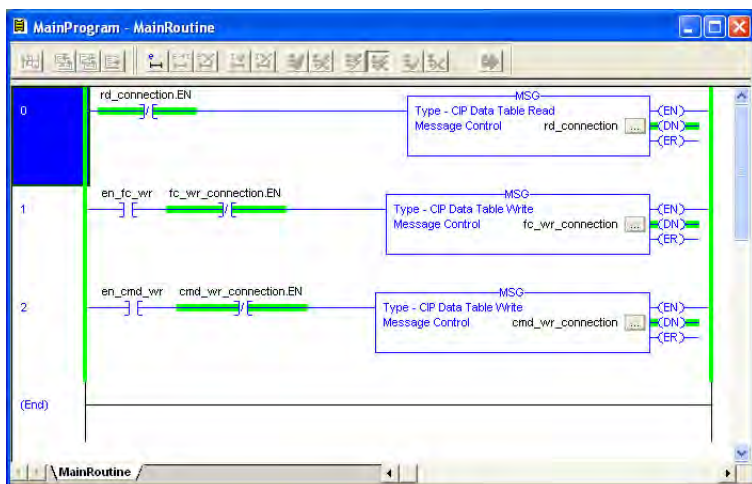


Figure 107: Reading and Writing via MSG Instructions

Figure 108 shows the configuration details of the example fc\_wr\_connection MSG instruction. Note that the chosen "Message Type" is "CIP Data Table Write", and that this instruction will only be writing to one inverter function code: namely, the frequency command (Destination Element is S05). The Source Element in this case is the 2<sup>nd</sup> element (starting from index 0) of an INT array tag named "wr\_data".

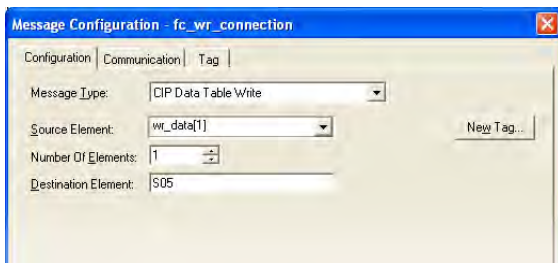


Figure 108: MSG Configuration for Writing

Note that when writing data via explicit messaging, use caution to ensure that the commanded function codes are not also simultaneously being commanded in the background via I/O messaging. Indeterminate behavior can occur if MSG instructions and background I/O data transfers are both writing to the same function codes. In other words, if the I/O messaging example procedure detailed in section 8.2.4 has already been implemented, and the same program is now being modified to implement explicit messaging, then it is recommended to inhibit the target module by selecting the "Inhibit Module" checkbox in the Connection tab of the Module Properties dialog.

## 8.3 Allen Bradley CSP

### 8.3.1 Overview

Ethernet-enabled Allen-Bradley legacy PLCs (such as the PLC5E and SLC-5/05 series) use a protocol called CSP (Client Server Protocol) to communicate over the Ethernet network. The flavor of CSP used by these PLCs is also known as "PCCC" (Programmable Controller Communication Commands) and "AB Ethernet". The interface card supports CSP for direct connectivity to these PLCs.

Since CSP relies on the use of EtherNet/IP class 3 messaging, timeout events also apply to CSP connections. If a connection timeout or socket-level error occurs, the driver will trigger a timeout event as described in section 5.5.3.

### 8.3.2 Tag Reference

Register contents are read from and written to the interface card via CSP by reference to an integer "file/section number" and an "offset/element" within that file. Reading is performed via the CSP "PLC5 Read" (DF1 protocol typed read) service, and writing is performed via the CSP "PLC5 Write" (DF1 protocol typed write) service.

The formula to calculate which register (function code) is targeted in the interface card is provided in Equation 7.

$$\text{target register} = (\text{file number} - 10) \times 100 + \text{offset} \quad \text{Equation 7}$$

Refer to chapter 6 for converting function codes to register numbers. In Equation 7, "target register"  $\in [1 \dots 4964]$ , "file number"  $\in [10 \dots 59]$  (which means N10...N59), and "offset" is restricted only by the limitations of the programming software (but is a value of 4964 max). Table 12 provides some examples of various combinations of file/section numbers and offsets/elements which can be used to access inverter registers. Note that there are multiple different combinations of file/section numbers and offsets/elements that will result in the same inverter register being accessed.

**Table 12: CSP Target Register Examples**

File/Section Number	Offset/Element	Start Target Register
N10	2	2
N12	62	262
N11	162	262
N27	98	1798
N20	798	1798
N59	64	4964
N10	4964	4964

In addition to providing access to the inverter registers in their "standard" numerical locations as mentioned above, the registers can also be accessed in a special "assembly object" type format by targeting integer file N60. What this means is that when N60 is targeted for reading, what is actually returned by the interface card is the user-defined register data as ordered by the EtherNet/IP produced data configuration array (refer to section 5.6.4). Similarly, when N60 is targeted for writing, the written data is disseminated to the inverter's registers according to the definition contained in the EtherNet/IP consumed data configuration array. By appropriate configuration of the EtherNet/IP consumed and produced data configuration arrays, therefore, bulk access to non-contiguous but frequently-used inverter registers can be conveniently provided by performing only one read and/or write instruction targeting file N60.

Because both the EtherNet/IP consumed and produced register configuration arrays are comprised of 32 function code definitions, the targeted "offset/element" must be within the range of 0 to 31 inclusive. Refer to Table 13 for some examples of N60 accesses.

**Table 13: Examples of EtherNet/IP-Style Bulk Access via File N60**

Offset/Element	Start Target Function Code of Configuration Array	Max Number of Accessible Elements
0	1 <sup>st</sup>	32
:	::	
15	16 <sup>th</sup>	16
:	::	
31	32 <sup>nd</sup>	1

The application PLC program uses a MSG instruction that is configured with a "Data Table Address" from which to start the access and a "Size in Elements" which determines the number of items to access (read or write). The "Data Table Address" is constructed by selecting a "File/Section Number" and an "Offset/Element" according to Equation 7. For example, a "File/Section Number" of N27 and "Offset/Element" of 99 = N27:99, which corresponds to register 1799 (the inverter's operation command register, function code S06).

### 8.3.3 SLC-5/05 Example: Read a Register Block

This example program will show how to continuously read a block of registers from the inverter with a single MSG instruction. Only one read request is outstanding at any given time.

- 1) Run RSLogix 500, and create a new configuration.
- 2) Create a control and a data file.
  - a) Right click Data Files and select New... The "Create Data File" dialog box appears (refer to Figure 109).
  - b) To create a control file, enter a file number (e.g. 20), set the type to "Integer", enter a descriptive name (e.g. "CONTROL"), and enter a number of elements (e.g. 100). Click OK to create the file. The control file is used to store configuration information pertaining to the functionality of the MSG instruction which will perform the data read.
  - c) Follow the same procedure to create a data file. This file will be used to store the incoming data read from the interface card. Enter a file number (e.g. 18), set the type to "Integer", enter a descriptive name (e.g. "DATA"), and enter a number of elements (e.g. 200). Refer to Figure 110. Click OK to create the file.

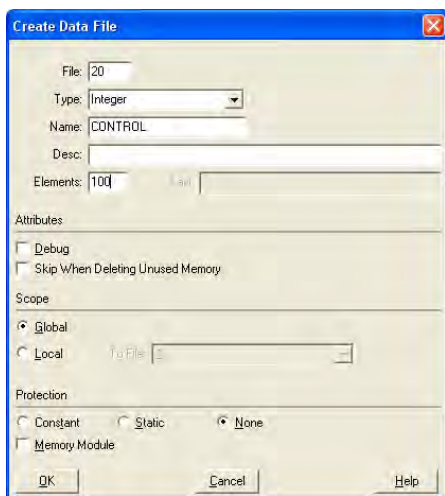


Figure 109: Creating a Control File

- 3) Add a MSG instruction to the program.
  - a) If not already visible, double-click "LAD2" under Project...Program Files in the controller organizer view to bring up the ladder logic program.
  - b) Right click on the default rung number on the left-hand side of the LAD2 window and select "Insert Rung".
  - c) Right click on the rung number of the new editable rung and select "Append Instruction".
  - d) Select the "MSG" instruction from the "Input/Output" classification, then click OK. Refer to Figure 111.

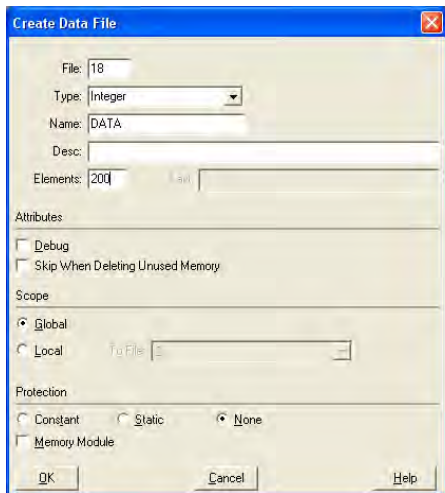


Figure 110: Creating a Data File

- 4) Add an XIO element to the program.
  - a) Right click on the rung number of the rung currently being edited and select "Append Instruction" again.
  - b) Select the "XIO" instruction from the "Bit" classification, then click OK. Refer to Figure 112.

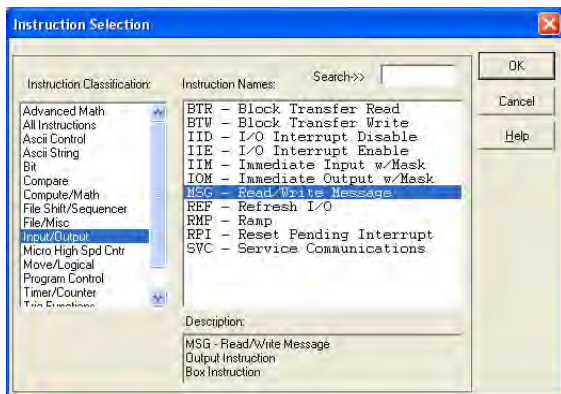


Figure 111: MSG Instruction Selection

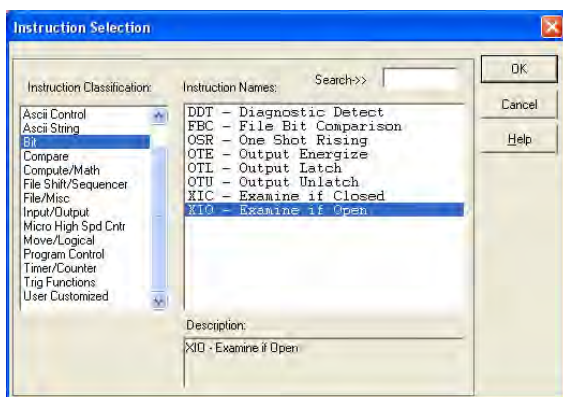


Figure 112: XIO Instruction Selection

## 5) Configure the MSG instruction.

- Set the "Read/Write" field to "Read", "Target Device" field to "PLC5", "Local/Remote" field to "Local", and "Control Block" to "N20:0".
- Upon hitting the <ENTER> key while in the "Control Block" entry box, the MSG Properties dialog box should appear (or it can be opened by clicking on the "Setup Screen" button at the bottom of the MSG instruction). Refer to Figure 113.
- In this example, we will be reading a total of 25 registers beginning at N30:50 (register 2050 / function code M01). To configure this, under "This Controller" set the "Data Table Address" field to N18:1, set the "Size in Elements" field to 25, and set the "Channel" field to 1 (Ethernet).
- Under "Target Device", set the "Data Table Address" field to N30:50 (starting target register=2050) and set the "MultiHop" field to Yes to cause the "MultiHop" tab to appear.
- Under the "MultiHop" tab settings, set the "To Address" in the first row to the inverter's IP address, and the "To Address" in the second row to 0. Refer to Figure 114.
- Close the dialog box. At this point, the program should appear as shown in Figure 115.

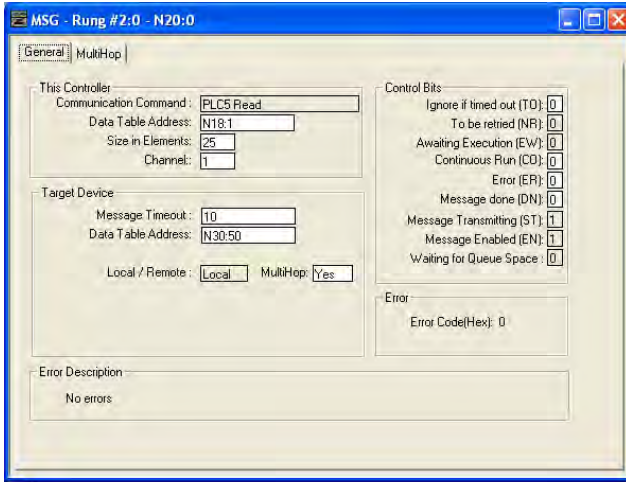


Figure 113: MSG Configuration, "General" Tab

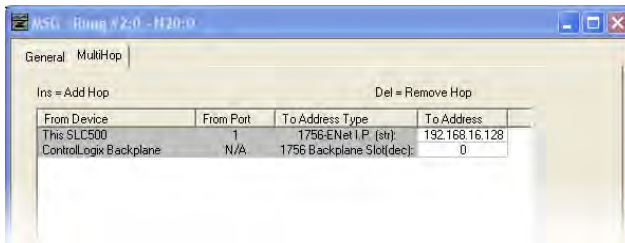


Figure 114: MSG Configuration, "MultiHop" Tab

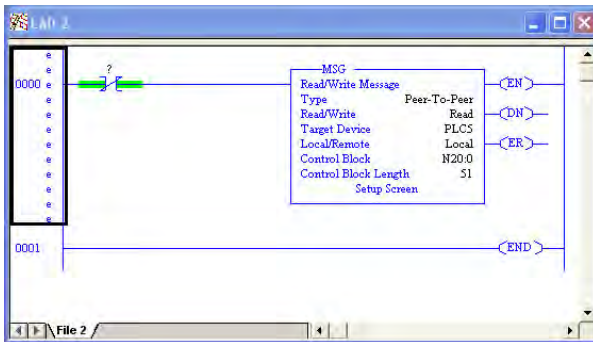
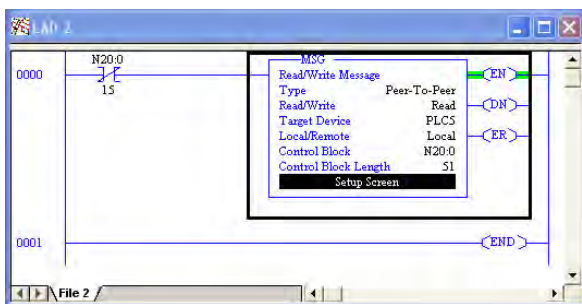


Figure 115: PLC Program after MSG Instruction Configuration

**6) Assign a tag to the XIO element.**

- a) Double-click on the XIO element located to the left of the MSG block. Type in N20:0/15 (MSG instruction's enable bit). This configuration causes the MSG instruction to automatically retrigger itself when it completes. While this is acceptable for the purposes of this example, it can produce high network utilization. In actual practice, it may be desirable to incorporate additional logic elements to allow triggering the MSG instruction at a specific rate or under specific conditions.

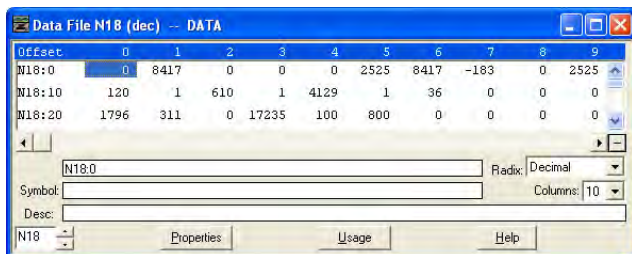
**7) The program is now complete. Refer to Figure 116.**



**Figure 116: Completed PLC Program**

**8) Save, download, and run the program.**

- a) To view the registers being read from the interface card, double-click the data file N18 under "Data Files" in the controller organizer view. 25 register values starting at register #2050 are being continuously read from the interface card and placed in the 25 sequential offsets of N18 starting at N18:1. Refer to Figure 117. We can see that N18:9 (register 2058 / output frequency / function code M09) has a value of 2525 (25.25Hz), N18:12 (register 2061 / output voltage / function code M12) has a value of 610 (61.0V), etc.



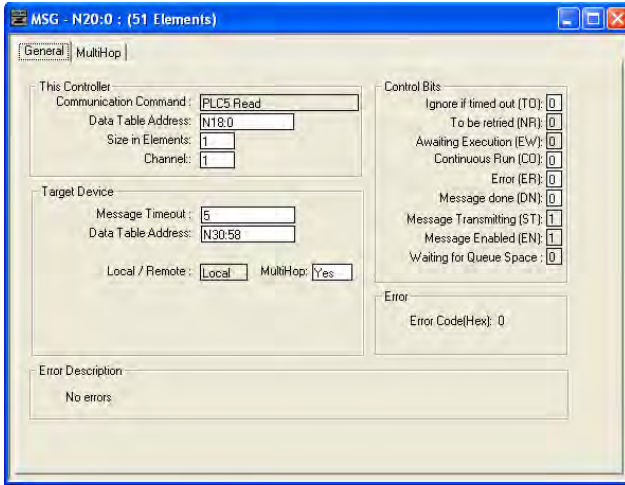
Offset	0	1	2	3	4	5	6	7	8	9
N18:0	0	8417	0	0	0	2525	8417	-183	0	2525
N18:10	120	1	610	1	4129	1	36	0	0	0
N18:20	1796	311	0	17235	100	800	0	0	0	0

**Figure 117: Monitoring the Data Being Read from the Inverter**

### 8.3.4 SLC-5/05 Example: Read a Single Register

The configuration and execution for reading a single register is in general identical to that required for reading a block of registers as detailed in section 8.3.3. The only difference is in the configuration of the MSG instruction. Figure 118 shows an example MSG instruction's General tab, which will read a single element (N30:58, which corresponds to register 2058 / output frequency / function code M09) and place it in the first element (offset 0) of N18.



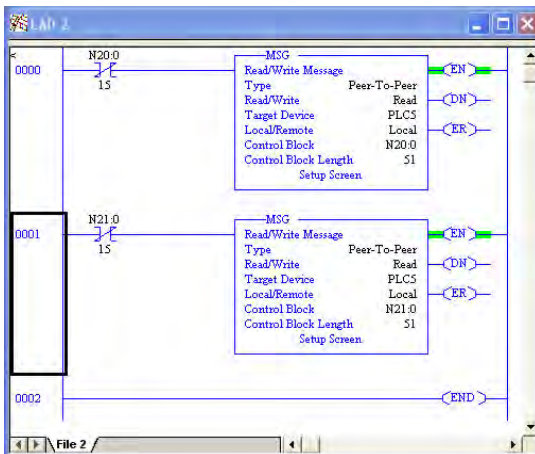


**Figure 118: Read the Inverter's Output Frequency Register**

### 8.3.5 SLC-5/05 Example: Multiple MSG Instructions

At times, reading from different groups of registers may be necessary. For example, a specific application may require some registers located in various disjoint locations in the register map. To accomplish this task efficiently, multiple MSG instructions can be implemented in the PLC program.

The configuration and execution for implementing multiple MSG instructions is in general identical to that required for implementing just one MSG instruction. Each MSG instruction will require its own message control file. In the case of read MSG instructions, more than one instruction may use the same data file to store the received register values, but the storage locations must not overlap. Figure 119 shows an example of two MSG instructions, each accessing different target integer files. It is evident from this logic that N20 and N21 are the two independent message control files created for these instructions.



**Figure 119: Multiple MSG Instructions**



### 8.3.6 SLC-5/05 Example: Reading and Writing

Often times, applications may need to both read data from and write data to the inverter. At a minimum, this will require two MSG instructions and two message control files. Figure 120 shows an example of two MSG instructions, one for reading and one for writing. Note that the "Read/Write" field of each of the MSG instructions is set according to their function.

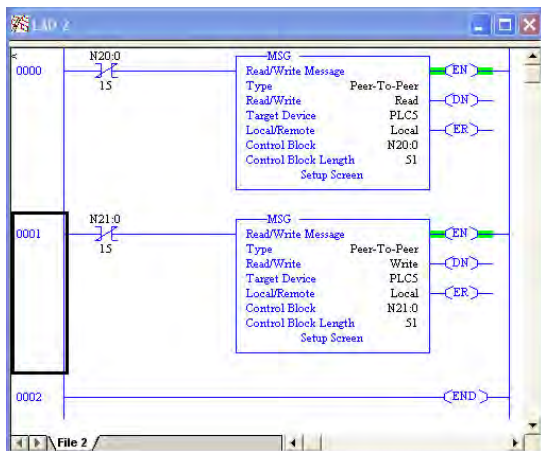
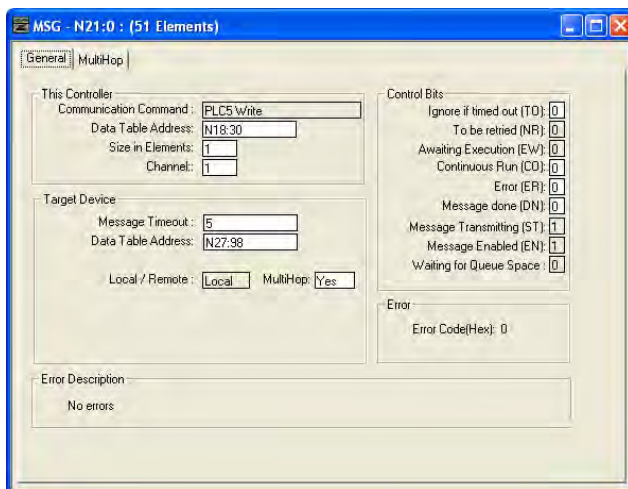


Figure 120: Reading and Writing via MSG Instructions

Figure 121 shows the configuration details of the "write" MSG instruction. Note that this instruction will only be writing to one inverter register: namely, register 1798 (function code S05 / frequency command). The source Data Table Address in this case is N18:30.



The figure shows the configuration details for the "write" MSG instruction. The "General" tab is selected. The "This Controller" section shows "Communication Command" as "PLC5 Write", "Data Table Address" as "N18:30", "Size in Elements" as "1", and "Channel" as "1". The "Target Device" section shows "Message Timeout" as "5", "Data Table Address" as "N27:98", and "Local / Remote" as "Local". The "Control Bits" section shows various status bits: "Ignore if timed out (ITD)" (0), "To be retried (NR)" (0), "Awaiting Execution (EW)" (0), "Continuous Run (CO)" (0), "Error (ER)" (0), "Message done (DN)" (0), "Message Transmitting (ST)" (1), "Message Enabled (EN)" (1), and "Waiting for Queue Space" (0). The "Error" section shows "Error Code(Hex)" as "0".

Figure 121: MSG Configuration for Writing

## 8.4 BACnet/IP

- The interface card supports the BACnet/IP (Annex J) protocol over Ethernet via UDP port 47808.
- The BACnet driver does not trigger timeout events (section 5.5.3).

### 8.4.1 Protocol Implementation Conformance Statement

#### BACnet Protocol

Date:	August 5, 2011
Vendor Name:	Fuji Electric
Product Name:	Fuji Electric FRENIC-Mega Inverter
Product Model Number:	OPC-G1-ETH
Applications Software Version:	V1.020
Firmware Revision:	V1.020
BACnet Protocol Revision:	2
Product Description:	

The Fuji Electric FRENIC-Mega series is a family of high-performance multifunctional inverters. Other features include ROHS compliance, built-in EMC filter, and long-life design.

#### BACnet Standard Device Profile (Annex L):

- ☐ BACnet Operator Workstation (B-OWS)
- ☐ BACnet Building Controller (B-BC)
- ☐ BACnet Advanced Application Controller (B-AAC)
- ☒ BACnet Application Specific Controller (B-ASC)
- ☐ BACnet Smart Sensor (B-SS)
- ☐ BACnet Smart Actuator (B-SA)

#### BACnet Interoperability Building Blocks Supported (Annex K):

- ☒ Data Sharing – ReadProperty-B (DS-RP-B)
- ☒ Data Sharing – ReadPropertyMultiple-B (DS-RPM-B)
- ☒ Data Sharing – WriteProperty-B (DS-WP-B)
- ☒ Device Management – Dynamic Device Binding-B (DM-DOB-B)
- ☒ Device Management – Dynamic Object Binding-B (DM-DOB-B)

#### Segmentation Capability:

None

- |  |                   |
|--|-------------------|
| <input type="checkbox"/> Segmented requests supported  | Window Size _____ |
| <input type="checkbox"/> Segmented responses supported | Window Size _____ |

#### Standard Object Types Supported:

See “Object Types/Property Support Table”.

#### Data Link Layer Options:

- ☒ BACnet IP, (Annex J)
- ☐ BACnet IP, (Annex J), Foreign Device
- ☐ ISO 8802-3, Ethernet (Clause 7)
- ☐ ANSI/ATA 878.1, 2.5 Mb. ARCNET (Clause 8)
- ☐ ANSI/ATA 878.1, RS-485 ARCNET (Clause 8), baud rate(s) \_\_\_\_\_
- ☐ MS/TP master (Clause 9), baud rate(s): 9600, 19200, 38400, 76800
- ☐ MS/TP slave (Clause 9), baud rate(s): \_\_\_\_\_
- ☐ Point-To-Point, EIA 232 (Clause 10), baud rate(s): \_\_\_\_\_
- ☐ Point-To-Point, modem, (Clause 10), baud rate(s): \_\_\_\_\_



- ☐ LonTalk, (Clause 11), medium: \_\_\_\_\_
- ☐ Other: \_\_\_\_\_

#### Device Address Binding:

Is static device binding supported? (This is currently for two-way communication with MS/TP slaves and certain other device.) ☐ Yes ☒ No

#### Networking Options:

- ☐ Router, Clause 6 - List all routing configurations
- ☐ Annex H, BACnet Tunneling Router over IP
- ☐ BACnet/IP Broadcast Management Device (BBMD)
- Does the BBMD support registrations by Foreign Devices? ☐ Yes ☐ No

#### Character Sets Supported:

Indicating support for multiple character sets does not imply that they can all be supported simultaneously.

- ☒ ANSI X3.4 ☐ IBM™/Microsoft™ DBCS ☐ ISO 8859-1
- ☐ ISO 10646 (UCS-2) ☐ ISO 10646 (UCS-4) ☐ JIS C 6226

If this product is a communication gateway, describe the types of non-BACnet equipment/networks(s) that the gateway supports: N/A

#### Datatypes Supported:

The following table summarizes the datatypes that are accepted (in the case of a write property service) and returned (in the case of a read property service) when targeting the present value property of each supported object type.

Object Type	Service	
	Read Property	Write Property
Analog Output	Real	Real, Unsigned, Integer, Null
Analog Input	Real	N/A
Binary Output	Enumerated	Enumerated, Boolean, Real, Unsigned, Integer, Null
Binary Input	Enumerated	N/A

#### Notes:

- The Null data type is used to relinquish a previously-commanded entry at the targeted priority in the priority array.
- When writing to Binary Output objects, all non-zero values are interpreted as a "1".

# Object Types/Property Support Table

Table 14: BACnet Object Types /Properties Supported

Property	Object Type				
	Device	Binary Input	Binary Output	Analog Input	Analog Output
Object Identifier	R	R	R	R	R
Object Name	R	R	R	R	R
Object Type	R	R	R	R	R
System Status	R				
Vendor Name	R				
Vendor Identifier	R				
Model Name	R				
Firmware Revision	R				
Appl Software Revision	R				
Protocol Version	R				
Protocol Revision	R				
Services Supported	R				
Object Types Supported	R				
Object List	R				
Max APDU Length	R				
Segmentation Support	R				
APDU Timeout	R				
Number APDU Retries	R				
Max Master					
Max Info Frames					
Device Address Binding	R				
Database Revision	R				
Present Value		R	W	R	W
Status Flags		R	R	R	R
Event State		R	R	R	R
Reliability		R	R	R	R
Out-of-Service		R	R	R	R
Units				R	R
Priority Array			R		R
Relinquish Default			R		R
Polarity		R	R		
Active Text		R	R		
Inactive Text		R	R		

R – readable using BACnet services

W – readable and writable using BACnet services

## 8.4.2 Supported Objects

**Table 15: Binary Input Object Instance Summary**

Instance ID	Object Name	Description	Active/ Inactive Text
BI1	FWD_ROT_STATUS	Forward rotation status	forward/off
BI2	REV_ROT_STATUS	Reverse rotation status	reverse/off
BI3	EXT_STATUS	DC injection braking	braking/off
BI4	INVERTER_SHUTDOWN	Inverter shutdown	on/off
BI5	BRAKING	Braking	braking /off
BI6	NUV	DC bus voltage normal	on/off
BI7	TORQUE_LIMITING	Torque limited	on/off
BI8	VOLTAGE_LIMITING	Voltage limited	on/off
BI9	CURRENT_LIMITING	Current limited	on/off
BI10	ACCELERATING	Accelerating	on/off
BI11	DECELERATING	Decelerating	on/off
BI12	ALARM	Alarm	on/off
BI13	COMM_ESTABLISHED	Communications established	on/off
BI14	BUSY_WRITING	Busy writing	on/off

**Table 16: Binary Output Object Instance Summary**

Instance ID	Object Name	Description	Active/ Inactive Text
BO1	FWD_ROT_CMD	Forward rotation command	forward/off
BO2	REV_ROT_CMD	Reverse rotation command	reverse/off
BO3	X1	General purpose input	on/off
BO4	X2	General purpose input	on/off
BO5	X3	General purpose input	on/off
BO6	X4	General purpose input	on/off
BO7	X5	General purpose input	on/off
BO8	X6	General purpose input	on/off
BO9	X7	General purpose input	on/off
BO10	X8	General purpose input	on/off
BO11	X9	General purpose input	on/off
BO12	EN_TERMINAL	Enable terminal	on/off
BO13	XF_FWD	General purpose input	on/off
BO14	XR_REV	General purpose input	on/off
BO15	ALARM_RESET	Alarm reset	on/off

**Table 17: Analog Input Object Instance Summary**

Instance ID	Object Name	Description	Units
AI1	OUTPUT_FREQ	Output frequency	Hz
AI2	OUTPUT_CURRENT	Output current	Amps
AI3	OUTPUT_VOLTAGE	Output voltage	Volts
AI4	INPUT_POWER	Input power	kW
AI5	OUTPUT_POWER	Output power	kW

**Table 18: Analog Output Object Instance Summary**

Instance ID	Object Name	Description	Units
AO1	FREQ_REF	Frequency command	Hz
AO2	ACCEL_TIME	Acceleration time	Seconds
AO3	DECEL_TIME	Deceleration time	Seconds

### **8.4.3 Supported Object Details**

#### **Binary Input Objects**

- BI1 ..... Indicates whether the inverter is running forward. Corresponds to function code M14, bit 0.
- BI2 ..... Indicates whether the inverter is running reverse. Corresponds to function code M14, bit 1.
- BI3 ..... Indicates DC injection braking or pre-exciting. Corresponds to function code M14, bit 2.
- BI4 ..... Indicates inverter shutdown. Corresponds to function code M14, bit 3.
- BI5 ..... Indicates braking. Corresponds to function code M14, bit 4.
- BI6 ..... Indicates normal DC bus voltage. Corresponds to function code M14, bit 5.
- BI7 ..... Indicates torque limited. Corresponds to function code M14, bit 6.
- BI8 ..... Indicates voltage limited. Corresponds to function code M14, bit 7.
- BI9 ..... Indicates current limited. Corresponds to function code M14, bit 8.
- BI10 ..... Indicates acceleration. Corresponds to function code M14, bit 9.
- BI11 ..... Indicates deceleration. Corresponds to function code M14, bit 10.
- BI12 ..... Indicates alarm. Corresponds to function code M14, bit 11.
- BI13 ..... Indicates communications established. Corresponds to function code M14, bit 12.
- BI14 ..... Indicates function code write in progress. Corresponds to function code M14, bit 15.

#### **Binary Output Objects**

- BO1 ..... Forward command. Corresponds to function code S06, bit 0.
- BO2 ..... Reverse command. Corresponds to function code S06, bit 1.
- BO3 ..... X1 command. Corresponds to function code S06, bit 2.
- BO4 ..... X2 command. Corresponds to function code S06, bit 3.
- BO5 ..... X3 command. Corresponds to function code S06, bit 4.
- BO6 ..... X4 command. Corresponds to function code S06, bit 5.
- BO7 ..... X5 command. Corresponds to function code S06, bit 6.
- BO8 ..... X6 command. Corresponds to function code S06, bit 7.
- BO9 ..... X7 command. Corresponds to function code S06, bit 8.
- BO10 ..... X8 command. Corresponds to function code S06, bit 9.
- BO11 ..... X9 command. Corresponds to function code S06, bit 10.
- BO12 ..... EN terminal command. Corresponds to function code S06, bit 11.
- BO13 ..... XF (FWD) command. Corresponds to function code S06, bit 13.
- BO14 ..... XR (REV) command. Corresponds to function code S06, bit 14.
- BO15 ..... Activates the alarm reset. Corresponds to function code S06, bit 15.

#### **Analog Input Objects**

- AI1 ..... The output frequency of the inverter in 0.01 Hertz units (6000=60.00Hz). Corresponds to function code M09.
- AI2 ..... The output current of the inverter in 0.1 or 0.01 Amp units (depends on inverter capacity). Corresponds to function code W05.
- AI3 ..... The output voltage of the inverter in 0.1 Volt units (1000=100.0V). Corresponds to function code W06.
- AI4 ..... Input power of the inverter in 0.01 kW units. Corresponds to function code W21.
- AI5 ..... Output power of the inverter in 0.01 kW units. Corresponds to function code W22.

#### **Analog Output Objects**

- AO1 ..... Frequency command of the inverter in 0.01 Hertz units. Corresponds to function code S05.
- AO2 ..... Sets the acceleration time in 0.1 second units. Corresponds to function code S08.
- AO3 ..... Sets the deceleration time in 0.1 second units. Corresponds to function code S09.

## 9 TROUBLESHOOTING

Although by no means exhaustive, Table 19 provides possible causes behind some of the most common errors experienced when using the interface card.

**Table 19: Troubleshooting**

Problem	Symptom	Solution
No communications between the interface card and the inverter	Inverter displays "Er4" code	<ul style="list-style-type: none"> <li>Confirm that the interface card connector is properly seated.</li> <li>Rebooting the interface card via the Fuji Finder application disrupts the communication with the inverter. Reset the fault.</li> </ul>
No communications between the network and the interface card	Communications cannot be established, or the Ethernet port's "LNK/ACT" LED flashes only infrequently or not at all	<ul style="list-style-type: none"> <li>Confirm that the destination IP address programmed into the controller equipment or computer matches that of the interface card, as displayed by the finder utility.</li> <li>Confirm that intermediate firewalls or routers have been configured to allow access to the interface via the applicable TCP/UDP ports.</li> <li>If attempting to access the web server on a computer whose web browser is configured to use a proxy server, ensure that the proxy server is accessible to the computer, and that the interface card is accessible to the proxy server.</li> </ul>
Firmware-generated error	"MODULE STATUS" LED is flashing red with an "x-y-z" 3-blink sequence. The number of LED flashes indicates an error code.	Contact technical support for further assistance.
XML socket connection failed	Message on a web server tab information window	TCP port 2000 is blocked by a firewall, router or some other intermediate network equipment.
Unable to control the inverter via network communications	Writing to command and frequency function codes/registers has no apparent effect on inverter operation	Confirm that the applicable inverter function codes are set to allow network control (refer to section 3.1).