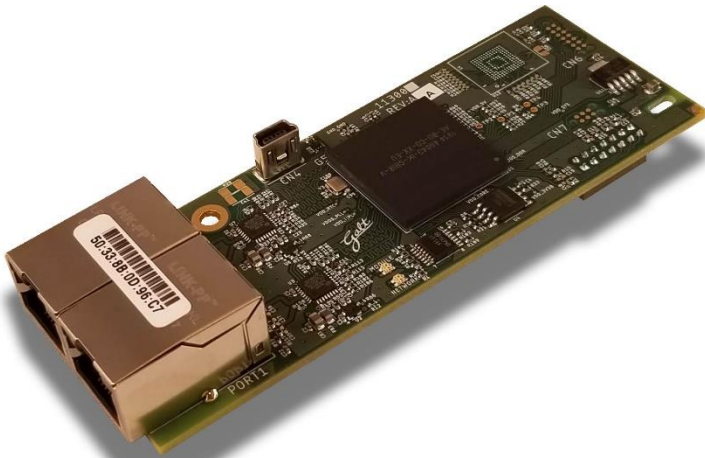




Galt Electric

G500 SERIES

G500ETH Multiprotocol Ethernet Interface



CAUTION

Thank you for purchasing the G500ETH Multiprotocol Ethernet Interface.

- This product is designed to connect the G500 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 inverter instruction manual.



G500ETH Multiprotocol Ethernet Interface Instruction Manual

Part Number 11133

Printed in U.S.A.

©2021 Galt Electric.

All rights reserved

Galt 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

This instruction manual has been prepared to help you connect your G500 series inverter to Industrial Ethernet networks using the G500ETH Multiprotocol Ethernet interface card. This instruction manual does not contain inverter usage instructions. Please refer to this instruction manual in conjunction with the applicable inverter instruction manual 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.

Related Publications

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



- **G500 Series Inverter Operation Manual**

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:

 WARNING	Failure to heed the information indicated by this symbol may lead to dangerous conditions, possibly resulting in death or serious bodily injuries.
 CAUTION	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 bus voltage is less than 36 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 appropriate inverter Operation Manual, 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 bus voltage is less than 36 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 (set function code P00.18 to a value of 1) 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 –

1	PRE-OPERATION INSTRUCTIONS.....	8
1.1	Product Overview	8
1.2	Features and Specifications	8
1.3	Unpacking and Product Confirmation	15
1.3.1	Shipment Confirmation	15
1.3.2	Component Overview	16
1.4	LED Indicators	17
1.4.1	Standard LEDs.....	17
1.4.2	Ethernet Link/Activity LEDs	17
1.4.3	Ethernet Speed LEDs	17
2	INSTALLATION	18
2.1	Pre-Installation Instructions.....	18
2.2	Installation Procedure	18
3	INVERTER FUNCTION CODE SETTINGS	20
3.1	Network Settings	22
3.1.1	IP Address (P16.02 ~ P16.05)	22
3.1.2	Subnet Mask (P16.06 ~ P16.09)	22
3.1.3	Default Gateway Address (P16.10 ~ P16.13).....	22
4	FUNCTION CODE NUMBERING AND BEHAVIOR.....	23
4.1	Register Address	23
4.2	Function Code Synchronization	25
4.3	Internal Function Codes	26
4.3.1	Command Word (Function Code P100.01)	26
4.3.2	Motor Group Selection Word (Function Code P100.02)	27
4.3.3	Command Process Data (Function Codes P100.20 - P100.30)	27
4.3.4	Status Word (Function Code P100.51)	28
4.3.5	Motor Group Feedback Word (Function Code P100.52)	28
4.3.6	Operation Mode Word (Function Code P100.53).....	29
4.3.7	Scanned Parameters Cycle Time (Function Code P100.54)	29
4.3.8	Number of Scanned Parameters (Function Code P100.55)	29
4.3.9	Max Invalid RX Count (Function Code P100.56)	29
4.3.10	Max Unexpected RX Count (Function Code P100.57)	29
4.3.11	Status Process Data (Function Codes P100.70 - P100.80).....	29
5	TIMEOUT PROCESSING	32
6	GALT ELECTRIC CONFIGURATION STUDIO	33
6.1	Overview	33
6.2	General Object Editing Activities.....	35
6.3	Device Settings.....	36
6.4	Process Data Configuration.....	36
6.4.1	Process Data Assignment Object.....	36
6.5	Ethernet Settings	37
6.5.1	Authentication	37
6.5.2	Network Configuration	37

6.6	Batch Update Mode	37
6.7	Internal Logic Settings	38
6.7.1	Fail-safe Values	38
6.8	Discovery over Ethernet.....	39
6.8.1	Discovery on the Local Ethernet Network	39
6.8.2	Discovery over the Internet.....	40
6.9	Manage Device Parameters.....	42
6.10	Monitor Device Parameters	43
6.11	Backup and Restore Parameters	44
6.12	Restore Factory Settings.....	45
6.13	Database.....	45
6.14	Diagnostic Object	45
6.15	Help.....	46
7	FILE SYSTEM.....	47
7.1	Overview	47
7.2	USB with Windows Explorer	47
7.3	FTP with Windows Explorer	48
8	FIRMWARE	49
8.1	Overview	49
8.2	Update Procedure.....	49
9	PROTOCOL-SPECIFIC INFORMATION	50
9.1	Modbus/TCP Server.....	50
9.1.1	Overview	50
9.1.2	Unit ID	50
9.1.3	Functions	50
9.1.4	Holding & Input Registers.....	50
9.1.5	Coil & Discrete Input Mappings	51
9.1.6	Connection Timeout Options	51
9.1.7	Node Settings	52
9.1.8	Holding/Input Register Remap Settings	52
9.2	EtherNet/IP Server	53
9.2.1	Overview	53
9.2.2	Server Settings	53
9.2.3	Connection Timeout Options	54
9.2.4	Generic Class 1 I/O Produced and Consumed Data Settings	54
9.2.5	Generic Class 1 (I/O) Connection Access.....	55
9.2.6	AC/DC Drive Profile Class 1 (I/O) Connection Access	56
9.2.7	Explicit Messaging Via Get/Set Attribute Single Services.....	58
9.2.8	Explicit Messaging Via Data Table Read/Write Services.....	59
9.2.9	Inverter Function Code Access Tag Format.....	59
9.2.10	ControlLogix Examples: Setup	59
9.2.11	ControlLogix Example: EDS Add-On Profile (AOP)	61
9.2.12	ControlLogix Example: I/O Messaging	66
9.2.13	ControlLogix Example: Read a Block of Function Codes	73
9.2.14	ControlLogix Example: Reading and Writing MSG Instructions.....	77
9.3	Allen Bradley CSP (PCCC) Server	78
9.3.1	Overview	78
9.3.2	Explicit Messaging Via Read/Write Services.....	78

9.3.3	Inverter Function Code File Number Offset Format	78
9.3.4	SLC-5/05 Example: Read Function Codes	80
9.3.5	SLC-5/05 Example: Reading and Writing.....	84
9.4	BACnet/IP Server	85
9.4.1	Protocol Implementation Conformance Statement	85
9.4.2	Default Supported Objects.....	89
9.4.3	Server Settings	92
9.4.4	Node Settings	92
9.4.5	Device Object Settings	92
9.4.6	BACnet Object Settings.....	92
9.5	PROFINET IO.....	99
9.5.1	Overview	99
9.5.2	Device Settings.....	99
9.5.3	Connection Timeout Options	99
9.5.4	Cyclic I/O Produced and Consumed Data Access Settings.....	100
9.5.5	PROFdrive Profile	101
9.5.6	Acyclic Data Access	105
9.5.7	TIA Portal (STEP 7) Hardware Configuration Example	105
9.5.8	GE Proficy Configuration Example	110
9.6	MELSEC/SLMP Server	115
9.6.1	Overview	115
9.6.2	Read/Write Commands	115
9.6.3	Server Settings	116
9.6.4	Connection Timeout Options	116
9.7	CC-Link IE Field Basic Server.....	118
9.7.1	Overview	118
9.7.2	Server Settings	118
9.7.3	Produced and Consumed Data Settings.....	118
9.8	MELSEC Client.....	120
9.8.1	Overview	120
9.8.2	Read/Write Commands	120
9.8.3	Connection Timeout Options	120
9.8.4	Remote Device Settings	120
9.8.5	Command and Monitor Data Object Settings.....	121
9.8.6	Diagnostics Objects.....	122
9.9	SLMP Client.....	123
9.9.1	Overview	123
9.9.2	Read/Write Commands	123
9.9.3	Connection Timeout Options	123
9.9.4	Remote Device Settings	123
9.9.5	Command and Monitor Data Object Settings.....	124
9.9.6	Diagnostics Objects.....	125
9.10	IEC 61850 Server.....	126
9.10.1	Overview	126
9.10.2	Server Settings	126
9.10.3	GOOSE Communication Parameters.....	126
9.10.4	Generic Process I/O Status and Control Object Settings	126
10	TROUBLESHOOTING	128

1 PRE-OPERATION INSTRUCTIONS

1.1 Product Overview

The G500ETH Multiprotocol Ethernet interface allows information to be transferred seamlessly between a G500 inverter and several Ethernet-based fieldbus networks with minimal configuration requirements. The interface installs directly onto the inverter, and presents two RJ-45 jacks with an embedded 10BASE-T/100BASE-TX 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 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 correspond to this manual's respective release date in order for all documented aspects to apply.

Supported Protocols

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

- Modbus/TCP Server
- EtherNet/IP Server (DLR node)
- Allen Bradley CSP Server (also known as "PCCC" and "AB Ethernet")
- BACnet/IP Server
- PROFINET IO Device (MRP client)
- Mitsubishi MELSEC / SLMP Server (also known as "MC protocol")
- CC-Link IE Field Basic Server
- Mitsubishi MELSEC Client
- SLMP Client
- IEC 61850 Server

1.2 Features and Specifications

Table 1: Features

Item	Description
Simultaneous Protocols	Supports all standard unmodified Ethernet (SUE) protocols simultaneously
Galt Electric Configuration Studio	Graphical user interface for discovery, configuration, and firmware update
Communication Loss Detection	Configurable actions for "fail-safe" conditions
Field Upgradeable	Firmware updates automatically handled by the studio
Parameter Management	Advanced management of parameter access
Parameter Backup and Restore	Drive cloning

Table 2: General Hardware Specifications

Item	Description
Power Supply	Directly powered by the inverter or via USB
Grounding	Referenced to inverter's 5V power supply ground
LED Indicators	Module Status, Network Status, 2 x Ethernet Link/Activity, 2 x Ethernet Speed
USB Port	USB 2.0, mini-B 5-pin

Table 3: Ethernet Hardware Specifications

Item	Description
Number of Ports	2 (internal switch)
Standard	IEEE 802.3 10BASE-T/100BASE-TX Ethernet compliant
Communication Speed and Duplex	10Mbps half/full, 100Mbps half/full (auto sense optimal speed and duplex)
Connector Type	RJ-45 Shielded
Auto MDI-X	Yes (supports all straight-through and cross-over cables)
Cable Type	CAT5-type 8-conductor UTP patch cables
Cable Length	100m per segment max
Topologies	Star/Tree, Linear/Bus/Daisy-chain, Ring (MRP / DLR)

Table 4: Modbus/TCP Specifications

Item	Description
Conformance Class	Class 0, Class 1 (partial), Class 2 (partial)
Read Function Codes	Read coils (1), Read input status (2), Read multiple registers (3), Read input registers (4), Diagnostics (8)
Write Function Codes	Write coil (5), Write single register (6), Force multiple coils (15), Write multiple registers (16)
Number of Connections	8
Max Read Register Size	125 registers
Max Write Register Size	123 registers
Register Data Type	16-bit integer
Unit (slave) ID	Ignored, echoed in response
TCP Port	502
Response Time	Min 160us, Typically less than 1ms

Table 5: EtherNet/IP Specifications

Item	Description
Conformance Tested	ODVA EtherNet/IP Conformance Test Software Version CT-13
Product Type Code	2 (AC Drive)
AC/DC Drive Profile	Yes
UCMM	Yes
Class 3 (Explicit) Messaging	Yes
Class 1 (Implicit I/O) Messaging	Yes
Class 1 Unicast T→O	Yes
Class 1 Multicast T→O	Yes
Number of Connections	16 (Total for both Class 1 and Class 3)
RPI	Min 1ms
I/O Input Size	Max 32 input words, user configurable
I/O Output Size	Max 32 output words, user configurable
Generic (User Configurable) Assembly Instances	100 (input) and 150 (output)
AC/DC Drive Profile Assembly Instances	20 (input) and 70 (output), 21 (input) and 71 (output)
Data Table Read/Write	Yes
DLR	Device Level Ring Node
Class 1 UDP Port	2222 (0x08AE)
Explicit Messaging Port	44818 (0xAF12)
Explicit Messaging Response Time	Min 160us, Typically less than 1ms

Table 6: Allen Bradley CSP (PCCC) Specifications

Item	Description
Read Services	PLC5 Read (DF1 protocol typed read, 0x68), PLC5 Word Range Read (DF1 protocol word range read, 0x01), SLC Read (DF1 protocol protected typed logical read with three address fields, 0xA2)
Write Services	PLC5 Write (DF1 protocol typed write, 0x67) , PLC5 Word Range Read (DF1 protocol word range write, 0x00), SLC Read (DF1 protocol protected typed logical write with three address fields, 0xAA)
Data Type	16-bit Integer
File Type	N (Integer)
Logical ASCII Addressing	Yes
Logical Binary Addressing	Yes
Max Read Size	240 bytes (120 16-bit Integers)
Max Write Size	240 bytes (120 16-bit Integers)

Table 7: BACnet/IP Specifications

Item	Description
BACnet IP	Annex J
Protocol Revision	2
Standard Device Profile (Annex L)	BACnet Application Specific Controller (B-ASC)
BACnet Interoperability Building Blocks (BIBB)	ReadProperty-B (DS-RP-B), ReadPropertyMultiple-B (DS-RPM-B), WriteProperty-B (DW-WP-B), Dynamic Device Binding-B (DM-DOB-B), Dynamic object Binding-B (DM-DOB-B)
Segmentation	No
Max APDU Length	1444 bytes
Character Sets	ANSI X3.4
Object Types	Analog Output, Analog Input, Analog Value, Binary Output, Binary Input, Binary Value, Multi-state Output, Multi-state Input, Multi-state Value
Priority Array	Yes
UDP Port	47808 (0xBAC0, configurable)
Response Time	Min 160us, Typical less than 1ms

Table 8: PROFINET Specifications

Item	Description
Protocol Level	RT (real-time)
RT Conformance Class	Class B
Netload Class	III
I/O Cycle Time	Min 1ms
I/O Input Size	Max 32 input words, user configurable
I/O Output Size	Max 32 output words, user configurable
MRP	Media Redundancy Protocol Client
DCP	Discovery, set station name, set IP address
LLDP	Yes
I&M	I&M0
Alarms	Plug, Pull
Number of Controllers	Allows access to only 1 controller

Table 9: MELSEC (MC Protocol) / SLMP Server Specifications

Item	Description
Frame Types	4E (MT), 3E (ST), 1E
Transport Types	TCP/IP, UDP/IP
3E (ST) / 4E (MT) Frame Read Function Codes	CPU Model Name Read (0x0101), Device Memory Batch Read (0x0401, Word units), Device Memory Random Read (0x0403, Word units) , Node Search (0x0E30), Device Info Compare (0x0E32), Status Read (0x0E44), Communication Setting Get (0x0E45)
3E (ST) / 4E (MT) Frame Write Function Codes	Device Memory Batch Write (0x1401, Word units), Device Memory Random Write (0x1402, Word units) , IP Address Set (0x0E31)
1E Frame Read Function Codes	Device Memory Batch Read (0x01, Word units)
1E Frame Write Function Codes	Device Memory Batch Write (0x03, Word units)
Number of Connections	8
Max Read Points	724 points (varies with function code and frame type)
Max Write Points	719 points (varies with function code and frame type)
3E Device Types	Data Register (0xA8), Link Register (0xB4), Index Register (0xCC), File Register (0xAF and 0xB0)
1E Device Types	Data Register (0x4420), Link Register (0x5720), File Register (0x5220 and 0x5A52)
TCP Port	2009 (Configurable)
UDP Port	2009 (Configurable)
Response Time	Min 160us, Typically less than 1ms

Table 10: CC-Link IE Field Basic Server Specifications

Item	Description
Max Occupied Stations	1
RWw Cyclic Size	Max 32 command words, user configurable
RWr Cyclic Size	Max 32 status words, user configurable
UDP Port	61450
Response Time	Min 160us, Typically less than 1ms

Table 11: MELSEC MC Protocol Client Specifications

Item	Description
Frame Types	4E (MT), 3E (ST), 1E, Auto-Detect
Transport Types	TCP/IP, UDP/IP
3E (ST) / 4E (MT) Frame Read Function Codes	Device Memory Batch Read (0x0401, Word units)
3E (ST) / 4E (MT) Frame Write Function Codes	Device Memory Batch Write (0x1401, Word units)
1E Frame Read Function Codes	Device Memory Batch Read (0x01, Word units)
1E Frame Write Function Codes	Device Memory Batch Write (0x03, Word units)
Number of Connections	8
Max Read Points	719 points (varies with frame type)
Max Write Points	719 points (varies with frame type)
3E Device Types	Any
1E Device Types	Any

Table 12: SLMP Client Specifications

Item	Description
Frame Types	4E (MT), 3E (ST), Auto-Detect
Transport Types	TCP/IP, UDP/IP
3E (ST) / 4E (MT) Frame Read Function Codes	Device Memory Batch Read (0x0401, Word units)
3E (ST) / 4E (MT) Frame Write Function Codes	Device Memory Batch Write (0x1401, Word units)
Number of Connections	8
Max Read Points	719 points (varies with frame type)
Max Write Points	719 points (varies with frame type)
3E Device Types	Any

Table 13: IEC 61850 Server Specifications

Item	Description
Unbuffered Reports	Yes, writeable
GOOSE	Type 1, data set writeable
Dynamic Data Sets	Yes, maximum of 10 data sets
Generic Status Objects	100 MV (Measured Value integers)
Generic Control Objects	100 APC (Controllable Analog Process Value integers)
Authentication	None/Password, configurable

Table 14: Applicable Inverters

Series	Type	Capacity	Software version
G500	G500-00000UL-00	All capacities	All versions

Table 15: Environmental Specifications

Item	Description
Environmental Specifications	Refer to the G500 Operation Manual.

1.3 Unpacking and Product Confirmation

1.3.1 Shipment Confirmation

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

- G500ETH interface board (refer to Figure 1).
- One separate M3 x 6mm mounting screw (see Figure 2).



Figure 1: G500ETH Interface Board



Figure 2: M3 x 6mm Mounting Screw

1.3.2 Component Overview

Figure 3 provides an overview of the important interface card components.

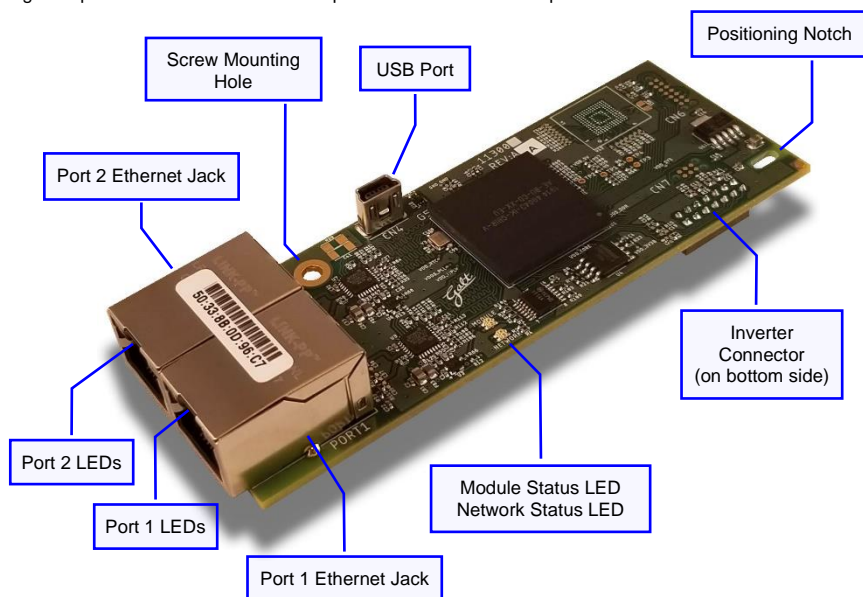


Figure 3: G500ETH Component Overview

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).

Port 1 and Port 2 Ethernet Jacks

Either jack can freely be used in star topology networks (with external switch). In linear topologies, a series of cards can be connected together by daisy-chaining one of the ports to the next inverter in line. In ring topologies, MRP (Media Redundancy Protocol) or DLR (Device Level Ring), must be supported by all devices on the network.

Standoff Mounting Hardware

The provided M3 x 6mm screw is used to secure the card to the standoff located on the inverter's control board. Refer to section 2.2.

Inverter Control Board Connector

Attaches to the inverter's connector board, which may vary depending on the inverter model.

USB Port

USB 2.0 port with mini-B connector. Used to access the card via the Galt Electric Configuration Studio (refer to section 5) and as a USB flash drive (refer to section 7). The USB connection can supply power to the card when the card is not installed on an inverter or the inverter is powered off. USB should only be connected when performing maintenance and configuration. Otherwise, the USB cable should be disconnected.

Module Status and Network Status LEDs

These LEDs indicate the current status of the interface card and protocols in use. Refer to section 1.4.

Ethernet Link/Activity and Speed LEDs

A pair of LEDs are provided on each Ethernet port. These LEDs provide insight into the Ethernet network's status, activity, and speed. Refer to section 1.4.

1.4 LED Indicators

1.4.1 Standard LEDs

1.4.1.1 Network Status LED

LED Activity	Status	Note
Off	Device Off	The inverter power is off
Green Blink / Red Blink	Startup	Startup blink sequence
Green Blink	No Connection	EtherNet/IP connection is not established
Green Off	No Connection	PROFINET connection is not established
Green On	Connection Established	EtherNet/IP or PROFINET connection is established

1.4.1.2 Module Status LED

LED Activity	Status	Note
Off	Device Off	The inverter power is off
Green Blink / Red Blink	Startup	Startup blink sequence
Green On	Device On	Normal status
Green Blink	Discovery identification	PROFINET discovery and identification (DCP)
Red Blink	Error Code	Record the error code sequence and contact technical support

1.4.2 Ethernet Link/Activity LEDs

LED Activity	Status	Note
Green On	Link	A valid Ethernet link exists: communication is possible on this port
Green Off	No Link	A valid Ethernet link does not exist: communication is not possible on this port
Green Blink	Activity	Indicates when a packet is transmitted or received on this port

1.4.3 Ethernet Speed LEDs

LED Activity	Status	Note
Yellow On	100Mbps	Fast 100Mbps
Yellow Off	10Mbps	10Mbps

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 bus voltage is less than 36 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.
- Additional option cards may be used when the G500ETH is installed in the inverter. Please consult with the factory first to confirm compatibility.

2.2 Installation Procedure

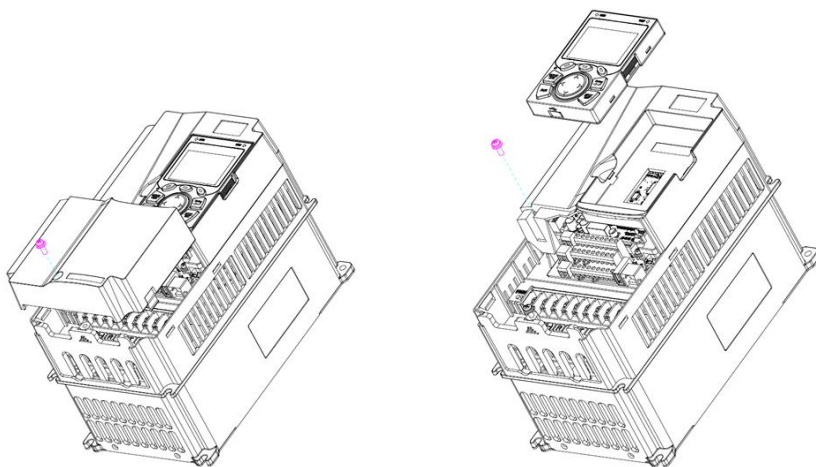


Before installing the interface card, perform all wiring for the main circuit terminals and control circuit terminals.

1. Remove the front cover from the inverter to expose the control printed circuit board (control PCB).



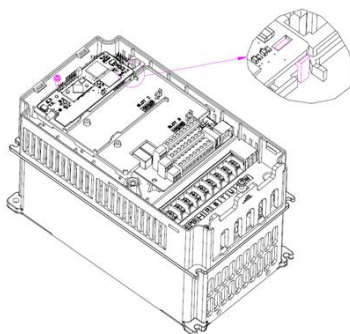
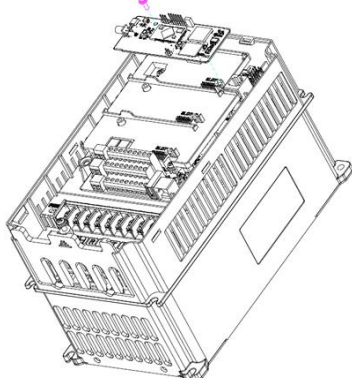
For installation instructions, refer to the G500 Operation Manual.



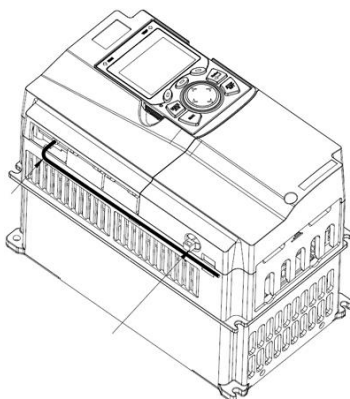
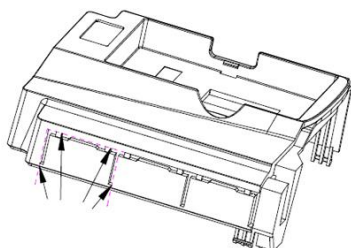
2. Engage connector CN3 (on the back of the interface card) into the "Slot 1", "Slot 2", or "Slot 3" connector on the connector board. Ensure that the connectors are fully engaged.



Ensure that the interface card is fully aligned and seated into the communication port. Failure to do so may lead to insufficient connector insertion and result in contact failure.



3. Secure the interface card to the connector board PCB by tightening the included M3 x 6mm screw into the standoff mounting hole.
4. Cut off the appropriate tabs on the side of the inverter's front cover and open the knockout for cabling.
5. Connect the network cable(s) as necessary. Insert the Ethernet cable(s) into the Ethernet jack(s), 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 cable(s) are not being pinched and are not routed near any power-carrying wiring.



For reinstallation instructions, refer to the G500 Operation Manual.

3 INVERTER FUNCTION CODE SETTINGS

The inverter function codes listed in Table 16 and Table 17 are important for proper operation of the end-to-end communication system. Some of these parameters must be set to specific values, and some may have multiple allowable settings depending on the desired operation of the overall application. 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 communications with, and control of, the inverter.



For further details regarding these function codes, please refer to the **G500 Operation Manual**.

Table 16: Inverter Control Function Codes

Function Code	Name	Detailed Parameter Description	Required Value for Control via G500ETH
P00.01	Running command channel	0: Keypad 1: Terminal 2: Communication	2: Communication
P00.02	Communication running command channel	0: Modbus 1: PROFIBUS/CANopen/DeviceNET/ G500XLT 2: Ethernet 3: EtherCat/Profinet/EtherNetIP/ G500ETH 4: PLC programmable card 5: Wireless communication card Note: 1, 2, 3, 4 and 5 are extended functions which are applicable with corresponding cards.	3: EtherCat/Profinet/ EtherNetIP/G500ETH
P00.06	A frequency command selection	0: Set via keypad 1: Set via AI1 2: Set via AI2 3: Set via AI3 4: Set via high speed pulse HDIA 5: Set via simple PLC program 6: Set via multi-step speed running 7: Set via PIDcontrol 8: Set via Modbus communication	13: Set via EtherCat/Profinet/ EtherNetIP/G500ETH communication
P00.07	B frequency command selection	9: Set via PROFIBUS/CANopen/ DeviceNET/G500XLT communication 10: Set via Ethernet communication 11: Set via high speed pulse HDIB 12: Set via pulse string AB 13: Set via EtherCat/Profinet/ EtherNetIP/G500ETH communication 14: Set via PLC card 15: Reserved	
P00.09	Combination mode of settings source	0: A 1: B 2: (A + B) 3: (A - B) 4: Max(A, B) 5: Min(A, B)	0: A 1: B 2: (A + B) 3: (A - B) 4: Max(A, B) 5: Min(A, B)

Table 17: Inverter Communication Error Function Codes

Function Code	Name	Detailed Parameter Description	Associated Fault Code
P16.24	Slot 1 extension card identification time	0.0 - 600.0s (Default value 0.0s)	60: Card slot 1 identification failure (F1-Er)
P16.25	Slot 2 extension card identification time	Sets the time allowed for the extension card to identify itself after the inverter has detected the card.	61: Card slot 2 identification failure (F2-Er)
P16.26	Slot 3 extension card identification time	When this parameter is set to 0.0, identification fault detection is not performed.	62: Card slot 3 identification failure (F3-Er)
P16.27	Slot 1 extension card communication timeout time	0.0 - 600.0s (Default value 5.0s)	63: Card slot 1 card communication timeout fault (C1-Er)
P16.28	Slot 2 extension card communication timeout time	Sets the maximum amount of time that can elapse without the inverter receiving communications from the extension card after the inverter has detected the card.	64: Card slot 2 card communication timeout fault (C2-Er)
P16.29	Slot 3 extension card communication timeout time	When this parameter is set to 0.0, card communication timeout fault detection is not performed. It is recommended that this parameter be kept at its default value of 5.0s.	65: Card slot 3 card communication timeout fault (C3-Er)
P16.57	G500ETH card timeout time	0.0 - 600.0s (Default value 0.0s) Sets the amount of time that a network timeout event must be active to cause the inverter to fault. If network communications recover before this time has expired, no fault is triggered. When this parameter is set to 0.0 or the inverter's running command channel (P00.01, P00.02) is not set to G500ETH, network timeout faults are not triggered. Refer to section 5 for details on this feature..	71: G500ETH fault (E-E71)

3.1 Network Settings

The network settings can **optionally** be set using the inverter parameters described in this section. However, to avoid invalid configuration, it is recommended to use the Configuration Studio to modify the network settings whenever possible. Please consult with your network administrator for the proper settings of these fields. Note that if the inverter's network settings are invalid, the network settings stored on the interface card will take precedence and will be written to the inverter.



After modifying the network parameters, power-cycle the inverter for the new settings to take effect.

Table 18: Network Settings Function Codes

Function Code	Name
P16.02	IP address most significant octet
P16.03	IP address high octet
P16.04	IP address low octet
P16.05	IP address least significant octet
P16.06	Subnet mask most significant octet
P16.07	Subnet mask address high octet
P16.08	Subnet mask address low octet
P16.09	Subnet mask address least significant octet
P16.10	Default gateway address most significant octet
P16.11	Default gateway address low octet
P16.12	Default gateway address high octet
P16.13	Default gateway address least significant octet

3.1.1 IP Address (P16.02 ~ P16.05)

The IP address is created from the octet values specified in P16.02 ~ P16.05. The byte order is defined according to the following example for an IP address of 192.168.16.100:

IP Address	192	168	16	100
Function Code	P16.02	P16.03	P16.04	P16.05

3.1.2 Subnet Mask (P16.06 ~ P16.09)

The subnet mask is created from the octet values specified in P16.06 ~ P16.09. The byte order is defined according to the following example for a subnet mask of 255.255.255.0:

Subnet Mask	255	255	255	0
Function Code	P16.06	P16.07	P16.08	P16.09

3.1.3 Default Gateway Address (P16.10 ~ P16.13)

The default gateway address is created from the octet values specified in P16.10 ~ P16.13. The default gateway address can be "disabled" by setting P16.10 ~ P16.13 to a value of 0. The byte order is defined according to the following example for a default gateway address of 192.168.16.1:

Gateway Address	192	168	16	1
Function Code	P16.10	P16.11	P16.12	P16.13

4 FUNCTION CODE NUMBERING AND BEHAVIOR

4.1 Register Address

All accessible inverter function codes can be referenced by their register address and can be conveniently referenced in the configuration studio (section 6.9). These same register addresses are used when accessing function codes via certain Ethernet protocols. The terms “function code” and “register” refer to data stored on the inverter and will be used interchangeably throughout this documentation. For a complete list of function codes, refer to the **G500 Series Inverter Operation Manual**.

All inverter function codes are exposed as register addresses according to a mathematical conversion formula which combines two elements (the function code group and the function code number) to create a unique register address for each function code. To determine the register address for a given function code, therefore, the function code group is first multiplied by 100, then added to the function code number. This operation is expressed mathematically via Equation 1. The function code to register address conversion examples are shown in the Table 19.

$$\text{Register Address} = (\text{Function Code Group} \times 100) + \text{Function Code Number} \quad \text{Equation 1}$$

For convenience, the function code to register address mapping can be referenced in the Configuration Studio (section 6.9).

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 address 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 4.2), 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 19: Function Code-to-Register Address Conversion Examples

Function Code Group	Group Number	Register Address Example Using Equation 1
P00: Basic functions	0	P00.00 (speed control mode): $(0 \times 100) + 0 = 0$: P00.99: $(0 \times 100) + 99 = 99$
P01: Start/stop control	1	P01.00 (running mode of start): $(1 \times 100) + 0 = 100$: P01.99: $(1 \times 100) + 99 = 199$
P02: Parameters of motor 1	2	P02.00 (type of motor 1): $(2 \times 100) + 0 = 200$: P02.99: $(2 \times 100) + 99 = 299$
P03: Vector control of motor 1	3	P03.00 (speed loop proportional gain 1): $(3 \times 100) + 0 = 300$: P03.99: $(3 \times 100) + 99 = 399$
P04: V/F control	4	P04.00 (V/F curve setup of motor 1): $(4 \times 100) + 0 = 400$: P04.99 $(4 \times 100) + 99 = 499$
P05: Input terminals	5	P05.00 (HDI input type): $(5 \times 100) + 0 = 500$: P05.99: $(5 \times 100) + 99 = 599$
P06: Output terminals	6	P06.00 (HDO output type): $(6 \times 100) + 0 = 600$: P06.99: $(6 \times 100) + 99 = 699$

Function Code Group	Group Number	Register Address Example Using Equation 1
P07: HMI	7	P07.00 (User password): $(7 \times 100) + 0 = 700$: P07.99: $(7 \times 100) + 99 = 799$
P08: Enhanced functions	8	P08.00 (Acceleration time 2): $(8 \times 100) + 0 = 800$: P08.99: $(8 \times 100) + 9 = 899$
P09: PID control	9	P09.00 (PID reference source): $(9 \times 100) + 0 = 900$: P09.99: $(9 \times 100) + 99 = 999$
P10: Simple PLC and multi-step speed control	10	P10.00 (Simple PLC mode): $(10 \times 100) + 0 = 1000$: P10.99: $(10 \times 100) + 99 = 1099$
P11: Protection parameters	11	P11.00 (Phase-loss protection): $(11 \times 100) + 0 = 1100$: P11.99: $(11 \times 100) + 99 = 1199$
P12: Parameters of motor 2	12	P12.00 (Type of motor 2): $(12 \times 100) + 0 = 1200$: P12.99: $(12 \times 100) + 99 = 1299$
P13: Control parameters of synchronous motor	13	P13.00 (Reduction rate of the injection current of synchronous motor): $(13 \times 100) + 0 = 1300$: P13.99: $(13 \times 100) + 99 = 1399$
P14: Serial communication function	14	P14.00 (Local communication address): $(14 \times 100) + 0 = 1400$: P14.99: $(14 \times 100) + 99 = 1499$
P15: Functions of communication extension card 1	15	P15.00 (Extension Card 1 Type): $(15 \times 100) + 0 = 1500$: P15.99: $(15 \times 100) + 99 = 1599$
P16: Functions of communication extension card 2	16	P16.00 (Extension Card 2 Type): $(16 \times 100) + 0 = 1600$: P16.99: $(16 \times 100) + 99 = 1699$
P17: State-check functions	17	P17.00 (Set frequency): $(17 \times 100) + 0 = 1700$: P17.99: $(17 \times 100) + 99 = 1799$
P18: Closed-loop control state check	18	P18.00 (Actual frequency of encoder): $(18 \times 100) + 0 = 1800$: P18.99: $(18 \times 100) + 99 = 1899$
P19: Extension card state check	19	P19.00 (State of card slot 1): $(19 \times 100) + 0 = 1900$: P19.99: $(19 \times 100) + 99 = 1999$
P20: Encoder of motor 1	20	P20.00 (Encoder type display): $(20 \times 100) + 0 = 2000$: P20.99: $(20 \times 100) + 99 = 2099$
P21: Position control	21	P21.00 (Positioning mode): $(21 \times 100) + 0 = 2100$: P21.99: $(21 \times 100) + 99 = 2199$
P22: Spindle positioning	22	P22.00 (Spindle positioning mode selection): $(22 \times 100) + 0 = 2200$: P22.99: $(22 \times 100) + 99 = 2299$
P23: Vector control of motor 2	23	P23.00 (Speed loop proportional gain 1): $(23 \times 100) + 0 = 2300$: P23.99: $(23 \times 100) + 99 = 2399$

Function Code Group	Group Number	Register Address Example Using Equation 1
P24: Encoder of motor 2	24	P24.00 (Encoder type display): $(24 \times 100) + 0 = 2400$: P24.99: $(24 \times 100) + 99 = 2499$
P25: Extension I/O card input functions	25	P25.00 (HDI3 input type selection): $(25 \times 100) + 0 = 2500$: P25.99: $(39 \times 100) + 99 = 2599$
P26: Output functions of extension I/O card	26	P26.00 (HDO2 output type): $(26 \times 100) + 0 = 2600$: P26.99: $(26 \times 100) + 99 = 2699$
P27: PLC functions	27	P27.00 (PLC function enable): $(27 \times 100) + 0 = 2700$: P27.99: $(27 \times 100) + 99 = 2799$
P28: Master/slave control functions	28	P28.00 (Master/slave mode selection): $(28 \times 100) + 0 = 2800$: P28.99: $(28 \times 100) + 99 = 2899$
P90: Customized function group 1	90	P90.00: $(90 \times 100) + 0 = 9000$: P90.99: $(90 \times 100) + 99 = 9099$
P91: Customized function group 2	91	P91.00: $(91 \times 100) + 0 = 9100$: P91.99: $(91 \times 100) + 99 = 9199$
P92: Customized function group 3	92	P92.00: $(92 \times 100) + 0 = 9200$: P92.99: $(92 \times 100) + 99 = 9299$
P93: Customized function group 4	93	P93.00: $(93 \times 100) + 0 = 9300$: P93.99: $(93 \times 100) + 99 = 9399$
P100: Control/status functions	100	P100.00: $(100 \times 100) + 0 = 10000$: P100.99: $(100 \times 100) + 99 = 10099$

4.2 Function Code Synchronization

The option card uses an internal database to cache all of the inverter's function codes. These function codes are constantly being read and/or written (as applicable), and their current values are therefore mirrored in the interface card's internal memory. The inverter provides two methods to synchronize parameters with the option card.

A limited number of parameters can be synchronized using high-speed, cyclic access. This method provides very fast function code synchronization by using a block of command process data and a block of status process data that is exchanged between the inverter and option card every communication cycle. Refer to sections 4.3.3 and 4.3.11 for more details on process data parameters.

The other method of synchronization is performed on a request-response basis, providing a single read or write transaction per communication cycle. Synchronization of these function codes is much slower than process data parameters and depends on the total number of function codes that exist on the inverter. Parameters that are synchronized using this method are referred to as "scanned" parameters. Only those function codes selected in the **Manage Device Parameters** configuration window in the Configuration Studio (refer to section 6.9 for details) are synchronized between the card and inverter in this manner.

This synchronization approach provides quick and deterministic response times to network requests and also allows the option card's network driver to be decoupled from inverter communications. Because of this decoupling, accesses to any function code (Pxx.00 to Pxx.99, where "xx" is any valid function code group from Table 19) will always be successful. Even if an inverter function code corresponding to a given register does not exist, the interface card still maintains a placeholder location in its internal

mirroring memory for that register. This feature allows for the block access of non-contiguous registers, as described in section 4.1.

One of the principle disadvantages of parameter caching is that write data checking is not available. This means that when the value of a parameter is modified via a network protocol, 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 command function code with a data value that is out-of-range, the interface card will not generate a corresponding error. However, the register can be read over the network at a later time to confirm whether or not that the written value “took hold” in the inverter.

4.3 Internal Function Codes

The interface card defines special internal parameters that do not exist on the inverter and are only available in the card's internal memory. These parameters provide command and status information that are otherwise unavailable on the inverter.

4.3.1 Command Word (Function Code P100.01)

The command word bit-mapping is described in Table 20.

Table 20: Command Word Bit-Mapping

Bit	Name	Value	Description
0	Start/Stop ¹	1	Start the drive
		0	Stop the drive
1	Reverse/Forward	1	Reverse
		0	Forward
2	Jog Operation	1	Jog enabled
		0	Jog disabled
3	Coast Stop ¹	1	Coast to stop
		0	No effect
4	Emergency Stop ¹	1	Emergency stop
		0	No effect
5	Reset Fault ^{1,2}	1	Reset fault
		0	No effect
6...10	Reserved		
11	Control Mode Switching	1	Switch torque/speed control mode
		0	Use current torque/speed control mode
12	Clear Power Consumption	1	Clear power consumption
		0	No effect
13	Pre-excitation Command	1	Enable pre-excitation
		0	Pre-excitation disabled
14	DC Brake Command	1	Enable DC brake
		0	DC brake disabled
15	Heartbeat Command	1	Set heartbeat value to 1
		0	Set heartbeat value to 0

1. These command bits are evaluated by the card using a priority, from highest to lowest, as follows: Reset Fault, Emergency Stop, Coast Stop, Start/Stop. Lower priority bits are

automatically cleared by the card when higher priority bits are set. Higher priority bits must be cleared before lower priority bits may be set.

2. This command bit is ignored if the drive is not faulted.

4.3.2 Motor Group Selection Word (Function Code P100.02)

The motor group selection word is described in Table 21.

Table 21: Motor Group Selection Word

Name	Value	Description
Motor Group Selection	0	Select motor group 1
	1	Select motor group 2
	2...65535	Reserved

4.3.3 Command Process Data (Function Codes P100.20 - P100.30)

These parameters are sent to the inverter from the card at high speed, every communication cycle and are used to command the inverter. There are 11 parameters available (Command Process Data 0 - 10). Each parameter is a placeholder for a single process data item. Section 6.4 provides details on configuring the data assigned to these parameters.

Table 22 details the default command process data. Since the process data mapping is configurable, the system integrator is responsible for managing, maintaining, and documenting the actual configuration. Always use the studio to confirm the configuration before commissioning the device.

Table 22: Default Command Process Data Mapping

Command Parameter	Process Data Assignment	Description
P100.20: Command Process Data 0	Frequency command	0.01Hz
P100.21: Command Process Data 1	PID reference	0.1%
P100.22: Command Process Data 2	PID feedback	0.1%
P100.23: Command Process Data 3	Torque command	0.1%
P100.24: Command Process Data 4	Voltage command	0.1%

4.3.4 Status Word (Function Code P100.51)

The status word bit-mapping is described in Table 23.

Table 23: Status Word Bit-Mapping

Bit	Name	Value	Description
0	Running/Stopped	1	Inverter is running
		0	Inverter is stopped
1	Reverse/Forward ¹	1	Reverse
		0	Forward
2	Inverter Fault	1	Faulted
		0	Not faulted
3	POFF	1	Inverter in POFF state
		0	Inverter not in POFF state
4...7	Reserved		
8	Bus Voltage Established	1	Ready to run
		0	Not ready to run
9...10	Reserved		
11	Motor Type Feedback	1	Synchronous motor
		0	Asynchronous motor
12	Overload Warning	1	Overload pre-alarm
		0	No overload pre-alarm
13...14	Reserved		
15	Heartbeat Feedback	1	Heartbeat value is 1
		0	Heartbeat value is 0

1. Only updated when Running/Stopped bit is 1, otherwise shows last running direction.

4.3.5 Motor Group Feedback Word (Function Code P100.52)

The motor group feedback word is described in Table 24.

Table 24: Motor Group Feedback Word

Name	Value	Description
Motor Group Feedback	0	Motor group 1 selected
	1	Motor group 2 selected
	2...65535	Reserved

4.3.6 Operation Mode Word (Function Code P100.53)

The operation mode word is described in Table 25.

Table 25: Operation Mode Word

Name	Value	Description
Run/Stop Mode	0	Keyboard control
	1	Terminal control
	2	Communication control
	3...65535	Reserved

4.3.7 Scanned Parameters Cycle Time (Function Code P100.54)

The measurement, in milliseconds, of the last parameter scan cycle time. This is the amount of time it took the option card to synchronize all scanned parameters with the inverter during the last parameter scan cycle.

4.3.8 Number of Scanned Parameters (Function Code P100.55)

This parameter shows the number of parameters that were synchronized during the last parameter scan cycle.

4.3.9 Max Invalid RX Count (Function Code P100.56)

The maximum number of consecutive, invalid frames received by the option card from the inverter. This parameter is a metric of the communication health between the option card and inverter. The value may be reset by writing a 0 to the parameter so that a new maximum can be determined.

4.3.10 Max Unexpected RX Count (Function Code P100.57)

The maximum number of consecutive, valid, but unexpected frames received by the option card from the inverter. This parameter is a metric of the communication health between the option card and inverter. The value may be reset by writing a 0 to the parameter so that a new maximum can be determined.

4.3.11 Status Process Data (Function Codes P100.70 - P100.80)

These parameters are received by the card from the inverter at high speed, every communication cycle and are used to monitor the status of the inverter. There are 11 parameters available (Status Process Data 0 - 10). Each parameter is a placeholder for a single process data item. Section 6.4 provides details on configuring the data assigned to these parameters.

Table 26 details the default status process data. Since the process data mapping is configurable, the system integrator is responsible for managing, maintaining, and documenting the actual configuration. Always use the studio to confirm the configuration before commissioning the device.

Table 26: Default Status Process Data Mapping

Status Parameter	Process Data Assignment	Description
P100.70: Status Process Data 0	Status word 2	See Table 27
P100.71: Status Process Data 1	Running frequency	0.01Hz
P100.72: Status Process Data 2	Frequency reference	0.01Hz
P100.73: Status Process Data 3	Bus voltage	0.1V
P100.74: Status Process Data 4	Output voltage	V
P100.75: Status Process Data 5	Output current	0.1A
P100.76: Status Process Data 6	Actual output torque	0.1%
P100.77: Status Process Data 7	Actual output power	0.1%

Status Parameter	Process Data Assignment	Description
P100.78: Status Process Data 8	Fault code	Refer to P07.27 description in the G500 Operation Manual
P100.79: Status Process Data 9	PID reference	0.1%
P100.80: Status Process Data 10	PID feedback	0.1%

Table 27: Status Word 2 Bit-Mapping

Bit	Name	Value	Description
0	Jogging	1	Inverter is jogging
		0	Inverter is not jogging
1	Frequency Level Detection FDT1	1	Output frequency has exceeded the FDT1 detection value
		0	Output frequency has decreased below the FDT1 detection value
2	Frequency Level Detection FDT2	1	Output frequency has exceeded the FDT2 detection value
		0	Output frequency has decreased below the FDT2 detection value
3	Frequency Arrival	1	Output frequency is within the detection range of the set frequency
		0	Output frequency is outside of the detection range of the set frequency
4	Running at Zero Speed	1	Inverter is running at zero speed
		0	Inverter in not running at zero speed
5	Upper Frequency Reached	1	The inverter is running at the upper limit frequency
		0	The inverter is not running at the upper limit frequency
6	Lower Frequency Reached	1	The inverter is running at the lower limit frequency
		0	The inverter is not running at the lower limit frequency
7	Pre-excitation Feedback	1	Pre-excitation enabled
		0	Pre-excitation disabled
8	Underload Warning	1	Underload pre-alarm
		0	No underload pre-alarm
9	Simple PLC State Completed	1	The simple PLC's current stage has completed
		0	The simple PLC's current stage has not completed
10	Simple PLC Cycle Completed	1	The simple PLC's cycle has completed
		0	The simple PLC's cycle has not completed
11	Positioning Completed	1	Position control positioning completed
		0	Position control positioning not completed
12	Spindle Zeroing Completed	1	Spindle zeroing completed
		0	Spindle zeroing not completed
13	Spindle Scale-Division Completed	1	Spindle scale-division completed
		0	Spindle scale-division not completed
14	Speed Limiting	1	The speed has been limited while in torque control
		0	The speed has not been limited
15	Speed/Position Control Switching Completed	1	Speed/position control switch-over completed
		0	Speed/position control switch-over not completed

5 TIMEOUT PROCESSING

The interface card can detect breaks in network communications and trigger a timeout event. The card's timeout options are configured on a per-protocol basis by the connection timeout options settings. Refer to the specific protocol section in section 9 for details on these settings. If a timeout timer is enabled, a timeout event will be triggered when a break in network communications exceeds the configured timeout time. If no timeout timers are enabled, timeout processing will be disabled.

The card supports two, mutually exclusive, timeout actions when a timeout event is triggered. When the timeout action is set to "Fault Drive", the card continuously reports the health of the network to the inverter using a timeout flag. The timeout flag is set when a timeout event is triggered. The timeout flag is cleared when network communications resume. The inverter can be configured to fault if the timeout flag remains set for a specific period of time. Refer to P16.57 in section 3 for details on this inverter setting.

Alternatively, when the timeout action is set to "Apply Fail-safe Values", the card can write fail-safe values to any inverter function codes when a timeout event is triggered. Refer to section 6.7.1 for details on how to configure fail-safe values.

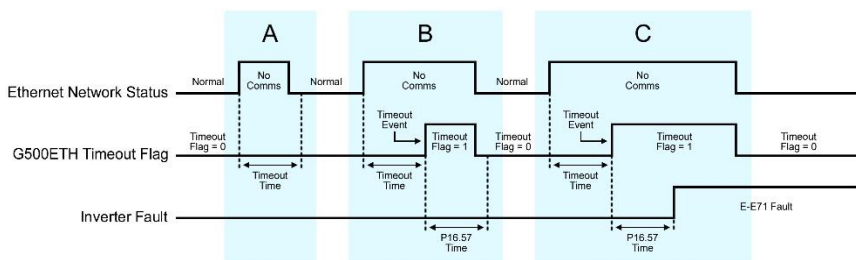


Figure 4: Timeout Processing Diagram

Figure 4 above shows the relative timing of the various signals, settings, and events associated with timeout processing when the card's timeout action is set to "Fault Drive". The diagram depicts three different scenarios:

- Ethernet communications are briefly interrupted, but recover before the card's timeout time has elapsed. Therefore, no timeout event is triggered and the card's timeout flag remains deasserted.
- Ethernet communications are interrupted for a period of time exceeding the card's timeout time. This triggers a timeout event and the card asserts its timeout flag. However, communications recover and the card deasserts its timeout flag before the inverter's P16.57 time has elapsed. Therefore, no fault is triggered.
- Ethernet communications are interrupted for a period of time exceeding the card's timeout time. This triggers a timeout event and the card asserts its timeout flag. The timeout flag remains asserted for a period of time exceeding the inverter's P16.57 time and the inverter faults. Communications recover and the card deasserts its timeout flag. However, the inverter remains faulted until the fault is reset.

6 GALT ELECTRIC CONFIGURATION STUDIO

6.1 Overview

The interface card is discovered, configured and updated by the Galt Electric Configuration Studio PC application (refer to Figure 5). The studio typically requires an Ethernet connection for remote discovery, network setting, configuration, and firmware updates. Configuration and firmware updates are also possible via USB. Otherwise, under normal operation, USB should be disconnected. To obtain the latest release of the Configuration Studio, refer to the [product web page](#) on the internet or contact technical support. The remainder of this section will provide only a brief introduction to the configuration concepts. For protocol specific configuration, refer to the relevant protocol section in section 9.

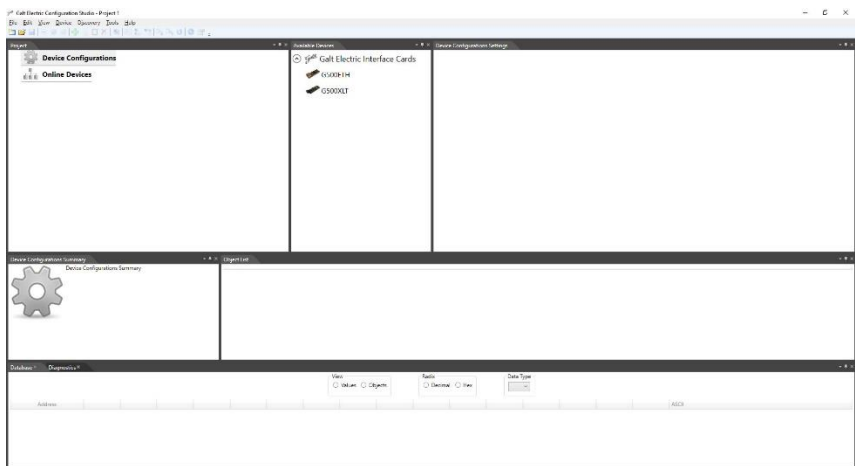


Figure 5: Galt Electric Configuration Studio

Creating a Device Configuration

A device can be added to the **Project** panel for configuration by first selecting the **Device Configurations** list heading and then:

- Double-clicking on the device in the **Available Devices** panel.
- Right-clicking on the device in the **Available Devices** panel and choosing **Add** from the context-sensitive menu.
- Hitting the <ENTER> key on the keyboard when the device is selected in the **Available Devices** panel.
- Dragging the device from the **Available Devices** panel into the **Project** panel.
- Selecting it and selecting **Add Selected Device** from the **Edit** menu.
- Selecting it and clicking the **Add** button in the toolbar.

The device will then be added to the list of **Device Configurations**.

Going Online with a Device

All connected devices are automatically added to the **Discovered Devices** panel. This panel is shown by selecting the **Online Devices** list heading in the **Project** panel. To go online with a device:

- Double-click on it in the **Discovered Devices** panel.
- Right-click on it in the **Discovered Devices** panel and choose **Go Online** from the context-sensitive menu.
- Hit the <ENTER> key on the keyboard when the device is selected in the **Discovered Devices** panel.
- Drag it from the **Discovered Devices** panel into the **Project** panel.
- Select it and select **Go Online with Device** from the **Edit** menu.

- Select it and click the **Go Online** button in the toolbar.

When the studio goes online with a device, its configuration is automatically read. While the studio is online with a device, it will appear in green text in the **Discovered Devices** panel. The studio may be online with multiple devices simultaneously. Note that the online configuration is read-only. The online configuration must first be uploaded into a project before it can be modified.

Uploading a Device's Configuration into a Project

The current configuration of an online device can be uploaded into the **Project** panel by selecting a device under the **Online Devices** list heading and then:

- Right-clicking on it and choosing **Upload Configuration** from the context-sensitive menu.
- Dragging it from the **Online Devices** heading into the **Device Configurations** heading.
- Selecting it and selecting **Upload Configuration to Project** from the **Device** menu.
- Selecting it and clicking the **Upload Configuration** button in the toolbar.

The device's configuration will then be added to the list of **Device Configurations**. Once the configuration is uploaded into the project, it may be modified.

Removing a Device Configuration from a Project

A configuration can be removed from a project by:

- Selecting the device in the **Project** panel and dragging it. A trash can icon will appear at the bottom of the **Project** panel, and dragging and dropping the device in the trash will remove it from the project.
- Hitting the <DELETE> key on the keyboard when the device is selected in the **Project** panel.
- Right-clicking on the device in the **Project** panel and choosing **Remove** from the context-sensitive menu.
- Selecting **Remove Selected Item** from the **Edit** menu when the device is selected.
- Clicking on the **Remove** button in the toolbar when the device is selected.

Going Offline with a Device

To go offline with a device:

- Select the device in the **Project** panel and drag it. A trash can icon will appear at the bottom of the **Project** panel, and dragging and dropping the device in the trash will go offline with it.
- Hit the <DELETE> key on the keyboard when the device is selected in the **Project** panel.
- Right-click on the device in the **Project** panel and choose **Go Offline** from the context-sensitive menu.
- Select **Go Offline with Device** from the **Edit** menu when the device is selected.
- Click on the **Go Offline** button in the toolbar when the device is selected.

Downloading a Configuration to a Device

To download a configuration to an online device, first select the device under the **Device Configurations** heading in the **Project** panel, and then navigate to **Device...Download Configuration to Device**. If the studio is currently online with only one compatible device, then the configuration will be downloaded to the online device. Otherwise, a device selection prompt is displayed to select which device to download the configuration to. Do not power off the device or interrupt the connection once the download is in progress as this may corrupt the firmware and/or the configuration.



Note Stop all other communication to the device when downloading.

Updating Firmware

The studio automatically manages firmware updates when going online with a device and downloading a configuration to a device. Download the latest studio from the [product web page](#) to obtain the latest firmware. Do not power off the device or interrupt the connection once the update is in progress as this may corrupt the firmware and/or the configuration.

Resetting an Online Device

To reset an online device, first select the device in the **Project** panel and then navigate to **Device...Reset Device**.

General Configuration Process

To configure a device, add the desired protocol(s) and configure any objects associated with the respective protocol(s). Any changes will take effect once the configuration is downloaded to a device.

Note that numeric values can be entered not only in decimal but also in hexadecimal by including "0x" before the hexadecimal number.

6.2 General Object Editing Activities

The following editing activities apply for all types of configuration objects and project elements.

Adding an Object

To add an object, click on an item (protocol driver or Node, for example) in the **Project** panel. Any available objects for that item will be listed in the **Available Objects** panel (the panel title depends on the currently-selected item). An object can then be added to the item by:

- Double-clicking on it.
- Right-clicking on it and choosing **Add** from the context-sensitive menu.
- Hitting the <ENTER> key on the keyboard when the object is selected.
- Dragging it into the **Project** panel.
- Selecting it and selecting **Add Selected Device** from the **Edit** menu.
- Selecting it and clicking the **Add** button in the toolbar.

The object's configurable fields can then be populated with valid values (where applicable).

Viewing an Object

In the **Project** panel, select a parent object to display a summary of all its child objects. For example, selecting a protocol driver will display the driver's configuration in the **Summary** panel and list of current objects in the **Object List** panel.

Updating an Object

To update an object, select the object in the **Project** panel and make any required changes in the **Settings** panel.

Deleting an Object

An object can be deleted by performing one of the three following actions:

- Selecting the object in the **Project** panel and dragging it. A trash can icon will appear at the bottom of the **Project** panel, and dragging the object to the trash will then delete it from the project.
- Hitting the <DELETE> key on the keyboard when the object is selected in the **Project** panel.
- Right-clicking on the object in the **Project** panel and choosing **Remove** from the context-sensitive menu.
- Selecting **Remove Selected Item** from the **Edit** menu when the object is selected.
- Clicking on the **Remove** button in the toolbar when the object is selected.

Note that this action cannot be undone. Deleting an object will also delete all of its child objects.

Copying and Pasting an Object

To copy an object, first click on an item in the **Project** panel. An object can then be copied by:

- Right-clicking on it and choosing **Copy** from the context-sensitive menu.
- Pressing the <CTRL+C> keys on the keyboard.
- Holding the <CTRL> key and dragging the item to the desired location in the **Project** panel.
- Dragging the item to a new location under a different parent object in the **Project** panel.
- Selecting **Copy Selected Item** from the **Edit** menu.

- Clicking on the **Copy** button in the toolbar.

To paste an object, first click on an item at the desired location in the **Project** panel. An object can then be pasted by:

- Right-clicking on it and choosing **Paste** from the context-sensitive menu.
- Pressing the <CTRL+V> keys on the keyboard.
- Dropping an item onto the desired location in the **Project** panel after holding the <CTRL> key and dragging the item.
- Dropping an item onto a new location under a different parent object in the **Project** panel after dragging the item.
- Selecting **Paste Item** from the **Edit** menu.
- Clicking on the **Paste** button in the toolbar.

After pasting an object, the object's configurable fields can then be modified with valid values (where applicable).

Note that the studio allows you to copy and paste items between different locations, including different devices. This is useful for copying partial configurations from one device to another.

Reordering Objects

Objects can be reordered in the **Project** panel by dragging the item to the desired location. If the item is dragged outside of the items in the project tree, it will be moved to the end.

6.3 Device Settings

The following fields can be configured for a device. To view or edit device settings, click on the device in the **Project** panel. The settings are then available in the **Settings** panel.

Device Description

Each device added to a project can be individually tagged with a unique description string of up to 32 characters in length. This allows the devices within a project or an automation system to be clearly identifiable with their location or functional purpose.

Inverter Timeout

Defines the maximum number of milliseconds without receiving valid communications from the inverter before triggering a Host Communication Error. To disable inverter timeout detection, set this field to 0.

6.4 Process Data Configuration

The card supports user-configurable process data. Process data is exchanged between the card and inverter at high speed, every communication cycle. Command process data is sent to the inverter from the card and status process data is received by the card from the inverter. There are a maximum of 11 command process data parameters (Command Process Data 0 - 10) and 11 status process data parameters (Status Process Data 0 - 10) that can be exchanged between the card and the inverter (refer to sections 4.3.3 and 4.3.7 for details on these parameters). The selection of which data is mapped to these process data parameters is performed by adding a process data assignment object.



Inverter parameters P16.32 - P16.53 configure which process data is exchanged between the card and inverter. The card automatically overwrites these parameters on startup for all configured process data assignments.

6.4.1 Process Data Assignment Object

A process data assignment object defines the data assigned to a specific process data parameter.

Description

This field is strictly for user reference: it is not used at any time by the device.

Command/Status Parameter

Selects the command/status process data parameter that will contain the value for the assigned data.

Assignment

Selects the data assignment for the configured process data parameter.

6.5 Ethernet Settings

The **Ethernet Settings** panel contains Ethernet-related items that are not specific to any given protocol. These settings must be appropriately configured regardless of any Ethernet control protocols that may be enabled. The **Ethernet Settings** panel is then available whenever the **Ethernet** port is selected in the **Project** panel.

6.5.1 Authentication

Be sure to make a note of the new settings whenever authentication credentials are changed, as they must be entered whenever the web page is accessed or an FTP session is initiated.

User Name

The username is case-sensitive and can contain letters ("a...z" and "A...Z") and numbers ("0...9").

Password

The password is case-sensitive and can contain letters ("a...z" and "A...Z") and numbers ("0...9").

6.5.2 Network Configuration

The card supports a static IP address. The IP Address, Subnet Mask and Default Gateway fields must be configured. Please consult with your network administrator for the proper settings of these fields.

6.6 Batch Update Mode

The Configuration Studio supports a batch update mode for quickly updating firmware, and optionally, the configuration on all discovered devices without user interaction. While in batch update mode, the studio will automatically go online with a card, update the firmware, update the configuration if a matching configuration is found in the project, and then go offline with the card. It will do this for all discovered devices while in this mode. For each discovered device, the studio creates a log entry in the batch update log detailing the actions performed on the card.

Entering Batch Update Mode from within the Studio

To start batch update mode when the studio is open, select **Start Batch Update Mode** from the **Tools** menu. After the studio has entered batch update mode, pressing the ESC key will exit batch update mode. If any devices were discovered while in batch update mode, the studio will display a prompt to view the batch update log.

Launching the Studio in Batch Update Mode

The batch update mode can also be started when the studio is launched by using the "-b" or "-B" command line switch, and optionally, specifying a project file path to load. For example, the command line options "-b MyProject.gcsproj" will load the project titled "MyProject" and start batch update mode. When batch update mode is entered using this method, the user cannot exit batch update mode using the ESC key.

Note that the command line options can also be used with a custom shortcut by appending them to the executable path in the **Target** field of the shortcut. This would allow a user to double click on the shortcut to launch the studio in batch update mode.

Viewing the Batch Update Log

After the studio has updated a card while in batch update mode, a log is available that can be accessed by selecting **Open Batch Update Log** from the **Help** menu. The log details the actions that the studio performed on discovered devices during the last batch update session.

At the end of the log, the studio records statistics for the batch update session. The statistics include the following information:

Devices Discovered

The total number of devices discovered while in batch update mode.

Successful

The total number of devices that were updated successfully.

Failed

The total number of devices that the studio failed to update.

Not Updated

The total number of devices that were not updated. This can occur if a device is already up to date, or if a device has limited network connectivity and cannot be updated.

Firmware Updated

The total number of firmware updates performed.

Configuration Updated

The total number of configuration updates performed.

Errors

The total number of devices that encountered an error while being updated. Note that this does not necessarily imply that the device failed to update.

6.7 Internal Logic Settings

6.7.1 Fail-safe Values

6.7.1.1 Overview

The card can be configured to perform a specific set of actions when network communications are lost (timeout event). This allows each inverter parameter to have its own unique “fail-safe” condition in the event of network interruption. Support for this feature 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:

- The timeout time
- Timeout Object configuration

6.7.1.2 Timeout Time

The timeout time is the maximum number of milliseconds for a break in network communications before a timeout will be triggered. This timeout setting is configured at the protocol level as part of a driver's configuration, and used by the protocol drivers themselves to determine abnormal loss-of-communications conditions. These conditions then trigger timeout processing events. If it is not desired to have a certain protocol trigger timeout processing events, then the protocol's timeout time may be set to 0 (the default value) to disable this feature.

For some protocols, the timeout time is set by the master device (PLC, scanner, etc.), and a timeout time setting is therefore not provided in the Configuration Studio's driver configuration. Additionally, not all protocols support timeout detection: refer to the protocol-specific sections of this manual for more information.

6.7.1.3 Timeout Object Configuration

A timeout object is used as part of the timeout processing to set certain parameters to “fail-safe” values. When a timeout event is triggered by a protocol, the timeout objects are parsed and written to the corresponding function code(s). The timeout object(s) will be executed sequentially from first to last. To add a timeout object, select the device in the **Project** panel, then add **Internal Logic...Fail-safe Values...Timeout Object**. The following paragraphs describe the configurable fields of a timeout object:

Description

This field is strictly for user reference: it is not used at any time by the device.

Function Code

Enter the function code (refer to section 4) that this object will write to.

Value

Enter the “fail-safe” timeout value that the function code encompassed by this timeout object will be automatically written with upon processing a timeout event triggered by a protocol.

6.7.1.4 Fail-safe Example

This example will demonstrate how to add one timeout object which will assign a value of 0 to function code P100.01 (command word). In the **Project** panel, select the device and add **Internal Logic...Fail-safe Values...Timeout Object** as shown in Figure 6. The red error indicators are normal at this stage as the **Timeout Object Settings** have not yet been configured.

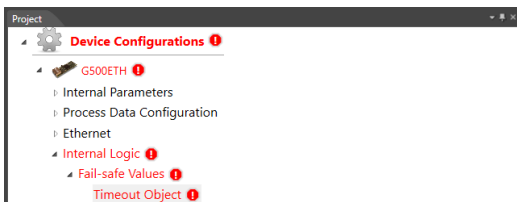


Figure 6: Timeout Object Project Panel

Next, configure the **Timeout Object Settings** as shown in Figure 7.

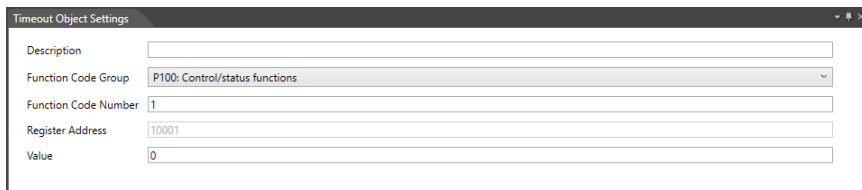


Figure 7: Timeout Object Settings

The example is complete.

6.8 Discovery over Ethernet

6.8.1 Discovery on the Local Ethernet Network

Depending on the currently-enabled driver, the Configuration Studio will automatically discover the device on the Ethernet network, regardless of whether or not the card's network settings are compatible with the subnet upon which they reside. All connected devices are automatically added to the **Discovered Devices** panel. This panel is shown by selecting the **Online Devices** list heading in the **Project** panel. In the **Discovered Devices** panel, discovered Ethernet devices will be listed under **Ethernet** and will display the firmware version in brackets and the current IP address in parentheses to the right of the device name (refer to Figure 8.)

In order for the studio to discover devices, certain UDP Ethernet traffic (port 4334) 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 studio is unable to discover any devices on the current subnet, be sure to check the computer's firewall settings during troubleshooting, and add the studio as a program exception to the firewall configuration if necessary. It may be necessary to restart your PC before the new firewall configuration can take effect.

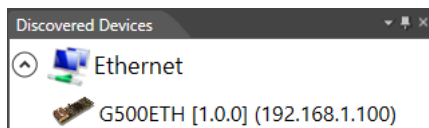


Figure 8: Configuration Studio Discovery over Ethernet

The network settings of a discovered card can be configured remotely by:

- Right-clicking on the device in the **Project** panel and choosing **Configure Network Settings...** from the context-sensitive menu.
- Selecting the device in the **Project** panel and navigating to **Device...Configure Network Settings...**

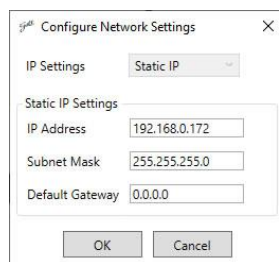


Figure 9: Remotely Configure Network Settings

The network settings pop-up should appear similar to Figure 9. Modify the network settings as necessary and click the OK button for the changes to take effect. Note that this will cause the device to become temporarily inaccessible and may trip the inverter.

6.8.2 Discovery over the Internet

The studio supports connecting to remote devices over the internet. However, due to internet routing limitations, the studio cannot automatically discover remote devices in the same manner as on the local Ethernet network. Therefore, some configuration is required to provide details to the studio on how to connect to the remote device. This is performed by opening the **Remote Sites** window by navigating to **Discovery...Remote Sites...**

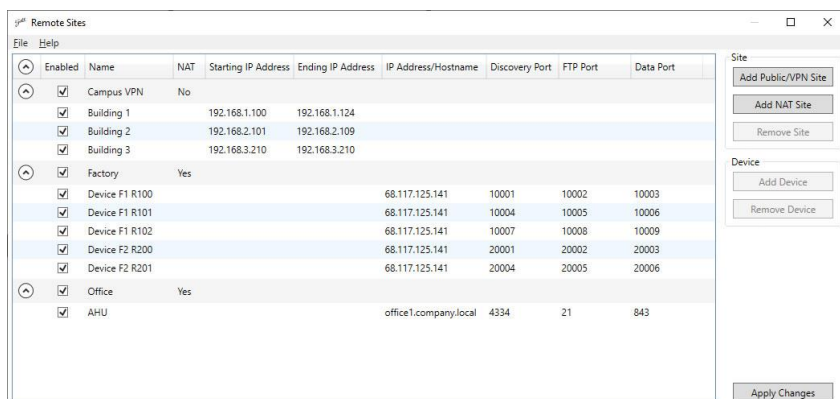


Figure 10: Remote Sites Window

6.8.2.1 How to Configure Remote Sites

To enable discovery of devices at remote sites, add a site then add one or more devices to the site. There are three types of remote sites Public, Virtual Private Network (VPN), and Network Address Translation (NAT).

Public Sites

Devices at public sites are directly accessible via the internet and have a public IP address assigned to them.



Tip

This is the simplest site type but is not very common, as it is a security risk to expose devices directly to the internet.

VPN Sites

VPN sites require a VPN connection to be established between your computer or network and the site's network.



If your site has VPN access, this is the easiest and most secure way to connect to devices remotely.

NAT Sites

These sites use a NAT router that has a public IP address or hostname. Devices behind the router have private IP addresses and are not directly accessible from the internet. The router must be configured for port forwarding to map publicly accessible ports to private IP address and port combinations.



This is the most complex site type because public to private IP address and port mappings must be configured in both the site's NAT router and in the studio.

Configuring Public/VPN Sites

Public/VPN sites are configured by defining one or more device ranges. A device range is a block of consecutive IP addresses from a starting IP address to an ending IP address.

Configuring NAT Sites

NAT sites are configured on a per-device basis by entering the publicly accessible IP address or hostname of the NAT router and the public ports on the router for discovery, FTP, and data that are forwarded to the appropriate ports on each device.

Configuring NAT Routers for Port Forwarding

Your site's network administrator must configure the site's NAT router to forward public ports to the following private ports for each device that will be accessed remotely.

Discovery Port: 4334 (UDP)

FTP Port: 21 (TCP)

Data Port: 843 (TCP)

6.9 Manage Device Parameters

The accessibility of the inverter parameters can be adjusted (refer to Figure 11). This is an advanced feature and must only be used after consulting technical support to determine the appropriate settings for the target application. The **Manage Device Parameters** configuration window is found by:

- Right-clicking on the device in the **Project** panel and choosing **Manage Parameters...** from the context-sensitive menu.
- Selecting the device in the **Project** panel and navigating to **Device...Manage Device Parameters...**

A parameter is accessible and actively scanned (read from and written to the inverter) only if its corresponding checkbox is enabled. Likewise, a parameter is inaccessible if its checkbox is disabled.

Note The parameter list serves as a useful reference when configuring objects.

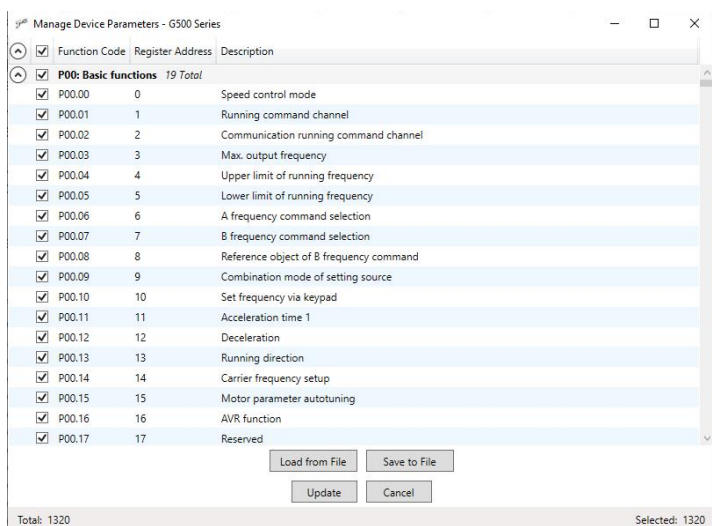


Figure 11: Manage Device Parameters

6.10 Monitor Device Parameters

The inverter's parameter values can be monitored and commanded in real-time (refer to Figure 12). The **Monitor Device Parameters** window is found by:

- Right-clicking on the online device in the **Project** panel and choosing **Monitor Parameters...** from the context-sensitive menu.
- Selecting the online device in the **Project** panel and navigating to **Device... Monitor Device Parameters...**

Radix and data type selections are available at the top of the window to select the display format of the **Value** column. The **Bits 15...0** column shows the current value of each bit in the parameter's value, starting at bit 15 on the left and continuing to bit 0 on the right. The bits are grouped into 4-bit nibbles for readability purposes.

A filter option, found at the top of the window, can be used to filter which parameters are shown. The filter function compares the filter text to each parameter's description, group name, parameter number, and communications number to display matching parameters. To use the filter function, simply type a word, or portion of a word, into the filter entry box. To reset the filter, click the **X** button to the right of the filter entry box. The filtering function is case insensitive.

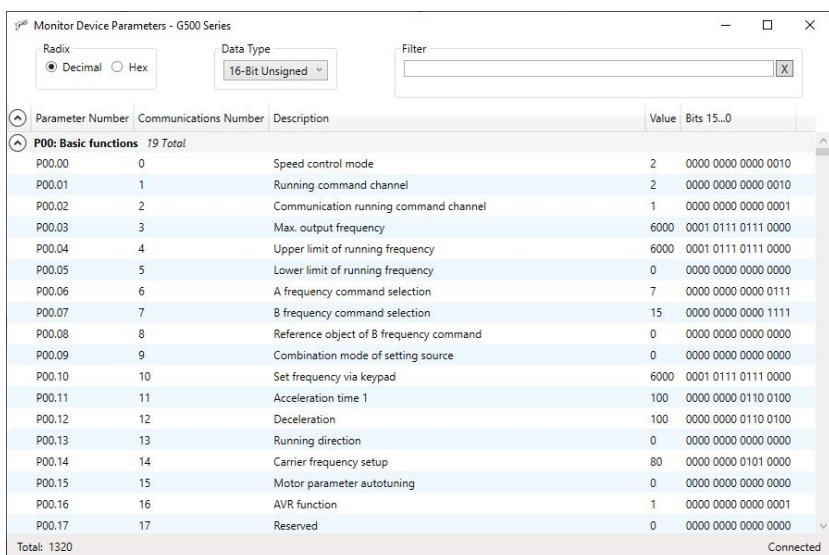


Figure 12: Monitor Device Parameters

6.11 Backup and Restore Parameters

The parameter values can be backed up from the inverter and restored to the inverter (refer to Figure 13 and Figure 14). This allows for easy inverter cloning. The backup parameter values are stored as a CSV file. A parameter value can be excluded from the list by disabling the corresponding checkbox. The parameter value can also be modified before the backup and restore is executed. Note that backup and restore does not modify the parameter list (refer to section 6.9). The backup and restore parameter configurations are found by:

- Right-clicking on the device in the **Project** panel and choosing **Backup Parameters...** or **Restore Parameters...** from the context-sensitive menu.
- Selecting the device in the **Project** panel and navigating to **Device...Backup Parameters from Device...** or **Restore Parameters to Device...**

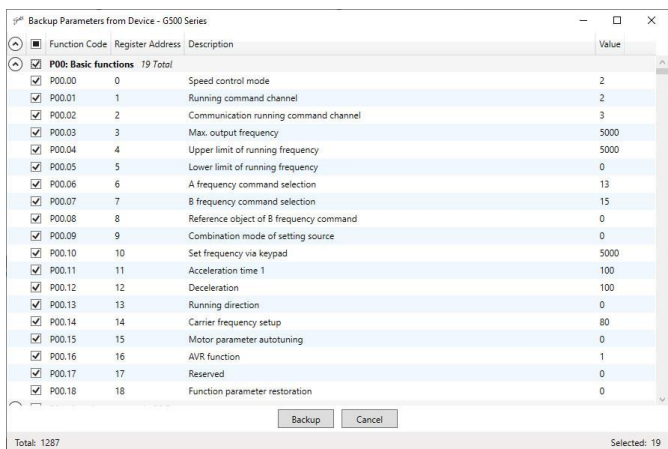


Figure 13: Backup Parameters

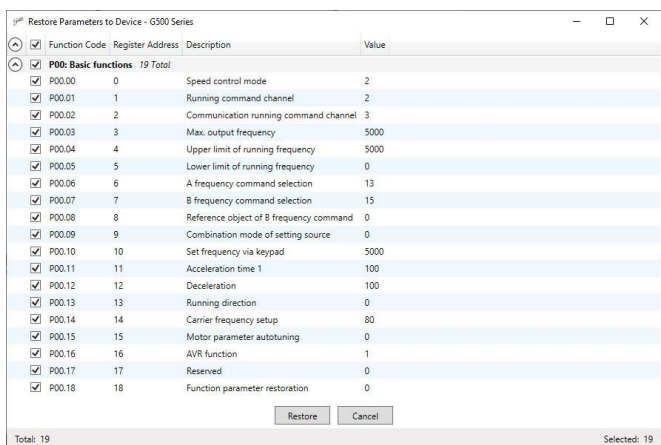


Figure 14: Restore Parameters

6.12 Restore Factory Settings

The interface card (connected via USB) can be restored to the factory settings. Note that the filesystem will be reformatted, which will destroy all custom modifications and configurations. Please backup the configuration before executing this feature. The factory settings can be restored by:

- Right-clicking on the device in the **Project** panel and choosing **Restore Factory Settings**.
- Selecting the device in the **Project** panel and navigating to **Restore Factory Settings**.

6.13 Database

To interact in real-time with inverter parameters, select the online device in the **Project** panel and then select the **Database** panel. Parameter values can be edited by double-clicking the desired location in the database. If the **Database** panel is not visible, it can be enabled via **View...Database**.

6.14 Diagnostic Object

Diagnostic objects are optional and can only be added to specific protocol objects.

TX Counter

A 32-bit counter that increments when the driver transmits a packet.

RX Counter

A 32-bit counter that increments when the driver receives a valid packet.

RX Error Counter

A 32-bit counter that increments when the driver receives an error response packet, or when an error occurs upon reception of a packet.

Current Status

Indicates the status of the most-recently received packet. This field is updated each time the "RX Counter" or "RX Error Counter" increments. Refer to Table 28 for a list of supported codes.

Last Error

Indicates the last reception error that occurred. This field is updated each time the "RX Error Counter" increments. Refer to Table 28 for a list of supported codes.

Table 28: Diagnostic Codes

Diagnostic Code (Hex)	Description
0x00	No Error
0xE0	No Connection
0xF0	Invalid Data Address
0xF1	Data Error
0xF2	Write To Read-Only
0xF3	Read From Write-Only
0xF4	Target Busy
0xF5	Target Error
0xF6	Cannot Execute
0xF7	Mode Error
0xF8	Other Error
0xF9	Memory Error
0xFA	Receive Error
0xFB	Invalid Function

0xFC	Invalid Packet
0xFD	Security Error
0xFE	Checksum Error
0xFF	Timeout Error

6.15 Help

Links to videos and documents can be found in the **Help** menu. Please review these links before contacting technical support for more in-depth assistance.

7 FILE SYSTEM

7.1 Overview

The interface card's on-board file system is used by the application firmware. Currently, the application firmware's main use of the file system is to store XML-encoded configuration files. The studio must be used to manage the configuration via USB or FTP. Do not manually access the configuration files unless instructed by technical support.

The configuration is only read at unit boot-up. Therefore, if a new configuration file is loaded, that unit must be rebooted for the new configuration take effect. Rebooting a unit can be performed by power-cycling the inverter in which the card is installed.

Interacting with the file system can be performed via USB (using a mini-B USB cable) as the interface card enumerates as a standard USB mass storage device ("flash drive"). The file system can also be accessed via FTP if the card has compatible network settings. Users can interact with the files on the interface card's file system in the same manner as though they were traditional files stored on a local or remote PC.

Note that the USB and FTP connection will prevent the file system from being accessed by other interfaces, such as the web server. Therefore, USB and FTP should only be connected when performing maintenance and configuration. USB and FTP should be disconnected while the card is running normally in a production environment.

7.2 USB with Windows Explorer

To use Microsoft Windows Explorer, first open either "Windows Explorer" or "My Computer". Refer to Figure 15. 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 16.



Figure 15:
Accessing
Windows Explorer

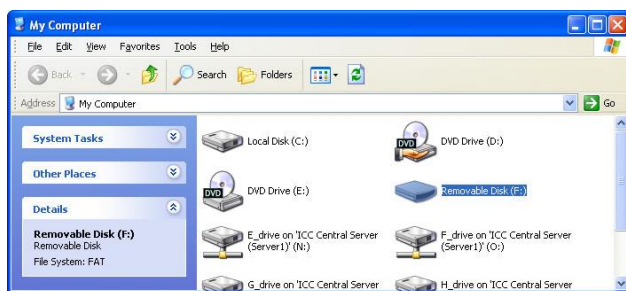


Figure 16: Removable Disk with Windows Explorer

Windows Explorer will then display the file system's contents (refer to Figure 17.) 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.

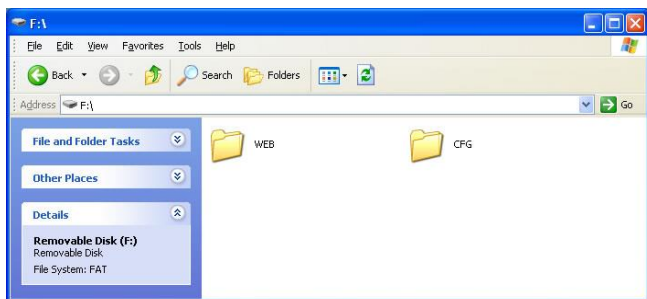


Figure 17: USB File Access via Windows Explorer

7.3 FTP with Windows Explorer

To use FTP with Microsoft Windows Explorer, first open either "Windows Explorer" or "My Computer". Please note that the indicated procedure, prompts and capabilities outlined here can vary depending on such factors as the installed operating system, firewalls and service packs.

In the "Address" field, type in "ftp://admin:admin@" and then the IP address of the target interface card (if the user name and password have been changed from its default, then replace the first "admin" with the new user name and the second "admin" with the password.) Refer to Figure 18.

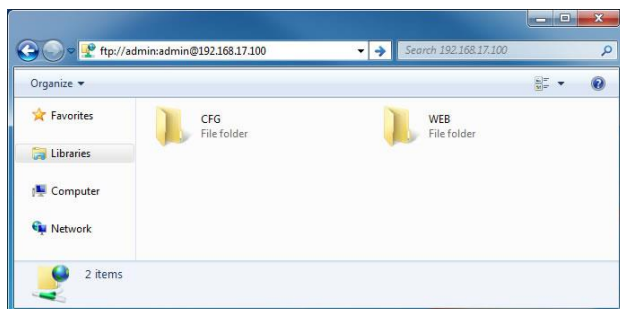


Figure 18: FTP via Windows Explorer

Note that the behavior of Windows Explorer FTP will vary from PC to PC. If you are having issues connecting FTP, there are other FTP client tools available such as Windows Command Prompt, Core FTP, FileZilla, SmartFTP etc. that can also be used to reliably access the card's file system.

8 FIRMWARE

8.1 Overview

The interface card's embedded firmware 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. In order to ensure that the firmware update is successful, and in the interest of equipment and personnel safety, it is strongly recommended to stop all of the card's production activities prior to initiating the firmware update procedure.



Note Failure to follow the firmware update procedure could result in corrupt firmware!

8.2 Update Procedure

1. Always back up your configuration to a PC for later recovery if necessary.
2. Download and install the latest Configuration Studio, which can be obtained from the [product web page](#).
3. 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.
4. Ensure that the device is in a safe state prior to initiating the firmware update. The card may be temporarily inaccessible during the firmware update process.
5. Locally via USB: Connect a USB cable between the card and the PC and open the studio. If the studio contains newer firmware, it will automatically prompt you to update the firmware. Proceed with the firmware update.
6. Remotely Via FTP: Connect an Ethernet cable and ensure that the card has compatible network settings.
7. Once the firmware update process has started, do not interrupt the card as this may corrupt the firmware. Do NOT manually power-cycle the inverter or reboot the card. Do NOT disturb the USB or Ethernet (FTP) connection.
8. After the firmware update has been completed, the card will reset automatically. When the card boots up again, it will be running the new application firmware, which can be confirmed by observing the version displayed in the Configuration Studio.

9 PROTOCOL-SPECIFIC INFORMATION

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

9.1 Modbus/TCP Server

9.1.1 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. Other notes of interest are:

- Supports up to 8 simultaneous connections.
- Modbus/TCP should not be confused with Modbus (serial) over TCP. Modbus over TCP is not compatible with Modbus/TCP and is not supported.
- The driver can be configured to detect a timeout (communication loss) and perform a timeout action.

9.1.2 Unit ID

The "unit identifier" (UI) field (also known as the slave ID and node ID) of the request packet is ignored and is echoed in the response. Any Unit ID value is allowed.

9.1.3 Functions

The supported Modbus/TCP functions are listed in Table 29.

Table 29: 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
8	Diagnostics (subfunction 0 only)	-
15	Force multiple coils	2
16	Write multiple registers	0

9.1.4 Holding & Input Registers

By default, the inverter registers are mapped as both holding registers (4X) and input registers (3X) and are accessed by targeting the inverter register addresses (described in section 4.1) plus 1. Examples are shown in Table 30.

Table 30: Register Address-to-Register Number Examples

Function Code	Register Address	Conversion	Holding/Input Register Number
P17.01 (Output Frequency)	1701	1701 + 1	1702
P00.11 (Acceleration time 1)	11	11 + 1	12
P08.09 (Jump frequency 1)	809	809 + 1	810

The 4X and 3X only serve as a naming convention for holding register and input register respectively, and should NOT be included as part of the actual on-the-wire register number. To further clarify, Modbus register 41702 is the same as Modbus holding register 1702. The same description applies to input registers (3X).

For example, from a Modbus/TCP master's point of view, in order to access the output frequency (function code P17.01, register address 1701, register number 1702) as a holding register, the Modbus/TCP master must execute the Read Multiple Registers function code and target register 1702. This will similarly apply when accessing an inverter function code as an Input Register.

9.1.5 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:

$$\text{register} = \left\lfloor \frac{\text{discrete} + 15}{16} \right\rfloor \quad \text{Equation 2}$$

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:

$$\text{bit} = (\text{discrete} - 1) \% 16 \quad \text{Equation 3}$$

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 2 and Equation 3 in a calculation example. Say, for instance, that we are going to read coil #34. Using Equation 2, 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 3, 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.

9.1.6 Connection Timeout Options

In the studio's Project panel, navigate to **G500ETH...Ethernet...Modbus/TCP Server**. The following configuration options will determine the actions to be taken if the connection is abnormally terminated or lost. 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. Note that a certain degree of caution must be exercised when using the timeout feature to avoid "nuisance" timeouts from occurring.

Enable Supervisory Timer

This timer provides the ability for the driver to monitor timeout occurrences on the overall receive activity for all connections.

- The timer will start after receiving the first request. Once the timer is started, it cannot be disabled.
- If the driver experiences no receive activity for more than the **Timeout** time setting, then the driver assumes that the client or network has experienced some sort of unexpected problem, and will perform the **Timeout Action**.

Enable Connection Timer

This timer provides the ability for the driver to monitor timeout occurrences and errors within the scope of each client connection.

- If a particular open socket experiences no activity for more than the **Timeout** time setting, then the driver assumes that the client or network has experienced some sort of unexpected problem, and will close that socket and perform the **Timeout Action**.
- If a socket error occurs (regardless of whether the error was due to a communication lapse or abnormal socket error), the driver will perform the **Timeout Action**. Specifically, do not perform inadvisable behavior such as sending a request from the client device, and then closing the socket prior to successfully receiving the server's response. The reason for this is because the server will experience an error when attempting to respond via the now-closed socket. Always be sure to manage socket life cycles "gracefully", and do not abandon outstanding requests.

Timeout

Defines the maximum number of milliseconds for a break in network communications before a timeout event will be triggered.

Timeout Action

Select an action from the drop down menu:

"None" No effect. The inverter will continue to operate with the last available settings.

"Apply Fail-safe Values" Apply the fail-safe values as described in section 6.7.1.

"Fault Drive" The behavior will depend on the timeout conditions set by the inverter (function code P16.57), which may result in an E-E71 fault. Refer to section 3.

9.1.7 Node Settings

There are no node settings. A node is simply a container for objects.

9.1.8 Holding/Input Register Remap Settings

In the studio's **Project** panel, add **G500ETH...Ethernet...Modbus/TCP Server...Node...Holding/Input Register Remap**.

The holding/input register remap objects are **OPTIONAL**. By default, all inverter function codes are already mapped as both holding (4X) and input (3X) registers (refer to section 9.1.2). For user convenience, register remap objects can be created to map any inverter function code to holding/input register addresses 15000 to 15049 (i.e. holding/input register numbers 15001 to 15050).

At times, it may be convenient to access inverter function codes in bulk Modbus transactions. This may be especially true in situations where it is desired to access certain function codes that are non-contiguous. For example, if it were desired to read the inverter's output frequency (function code P17.01, register number 1702), output current (function code P17.03, register number 1704), and output torque (function code P17.36, register number 1737), 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 number 1702 for a quantity of 36 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 by making use of the register remap objects. Non-contiguous inverter function codes 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.

Description

This 32-character (max) field is strictly for user reference: it is not used at any time by the driver.

Remap Register Address

Remap register that maps to the specified inverter function code. Select from 10101 to 10150.

Function Code

Function code (refer to section 4) that is accessed by the **Remap Register**.

9.2 EtherNet/IP Server

9.2.1 Overview

EtherNet/IP is a network adaptation of ODVA's Common Industrial Protocol (CIP). The card supports the EtherNet/IP server protocol, including the CSP server variant.

The interface card supports both implicit (class 1 I/O) and explicit (UCMM and class 3) messaging. Class 1 connections support two different types of I/O messaging. One type is the generic I/O assembly instances 100 and 150, which are entirely user-configurable (refer to section 9.2.5). The other type is the AC/DC drive profile assembly instances 20 & 70 or 21 & 71, which require no user configuration (refer to section 9.2.6). With I/O messaging, the data field contains only real-time I/O data. The meaning of the data is pre-defined at the time the connection is established. I/O messages are short and have low overhead, and therefore minimizes the processing time and allows for the time-critical performance.

With explicit messaging (refer to section 9.2.7), 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 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 and found to comply with ODVA EtherNet/IP Conformance Test Software Version CT-13.
- The interface card's product type code is 2 (AC Drive).
- The EDS file can be downloaded from the [product web page](#) on the internet.
- Supports DLR (Device Level Ring) node.
- Supports unconnected messages (UCMM), and up to 16 simultaneous class 1 (I/O) or class 3 (explicit) connections.
- 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 (communication loss) occurs, the driver can be configured to perform a timeout action. For class 1 connections, the timeout value is dictated by the scanner/client and is at least four times the RPI (Requested Packet Interval). For class 3 connections, the timeout value is also dictated by the scanner/client, but is typically a much larger value than for class 1 connections.

9.2.2 Server Settings

In the studio, navigate to **G500ETH...Ethernet...EtherNet/IP Server**.

Device Name

The device name is used for identification of a device on the EtherNet/IP network. This string is accessible as the "product name" attribute of the identity object. Enter a string between 1 and 32 characters in length.

DLR

Device Level Ring is a ring redundancy protocol. All devices in a DLR ring must support DLR.

- If the checkbox is cleared (default setting), the interface card will not operate correctly in a DLR ring. By disabling this option, the interface card should not be installed in a DLR ring.
- If the checkbox is checked, the interface card can participate and will operate correctly in a DLR ring. By enabling this option, the interface card can be installed successfully in a DLR ring.

9.2.3 Connection Timeout Options

In the studio's Project panel, navigate to **G500ETH...Ethernet...EtherNet/IP Server**. The following configuration options will determine the actions to be taken if the connection is abnormally terminated or lost. 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. Note that a certain degree of caution must be exercised when using the timeout feature to avoid "nuisance" timeouts from occurring.

Scanner Idle State Behavior

EtherNet/IP scanners (such as PLCs) have the option of adding a "run/idle" header to all class 1 (I/O) data packets sent to devices. This header is intended to signify when the scanner is in the "running" state or the "idle" state. For example, an Allen Bradley ControlLogix PLC will set the run/idle header to idle when its processor keyswitch is placed in the "PROG" position.

The Invoke Timeout on Scanner Idle State setting configures the behavior of the interface card when the scanner sets the run/idle header to idle.

- If the checkbox is not checked (default setting), then the driver will maintain the last consumed I/O data values received from the scanner. For example, if the scanner commanded the inverter to run prior to setting the run/idle header to idle, then the inverter will continue to run.
- If the checkbox is checked, then the driver will perform the **Timeout Action**.

Timeout Action

Select an action from the drop down menu:

"None" No effect. The inverter will continue to operate with the last available settings.

"Apply Fail-safe Values" Apply the fail-safe values as described in section 6.7.1.

"Fault Drive" The behavior will depend on the timeout conditions set by the inverter (function code P16.57), which may result in an E-E71 fault. Refer to section 3.

9.2.4 Generic Class 1 I/O Produced and Consumed Data Settings

The Produced Data Word and Consumed Data Word objects are only applicable when connecting to assembly instances 100 and 150 (generic I/O), which is typically the case. The Produced Data Word defines the structure of status data sent from the inverter back to the controller (T->O, target to originator). The Consumed Data Word objects will define the structure of the command data sent from the EtherNet/IP controller (for example, a ControlLogix PLC) to the inverter (O->T, originator to target). These objects allow the creation of custom-built I/O data. Up to 32 "command" function code values can be sent to the inverter, and up to 32 "status" function code values can be sent back to the controller. Therefore, up to 32 Produced and 32 Consumed Data Word objects can be created. If a consumed word offset is not defined, that data will be ignored by the inverter. If a produce word offset is not defined, the value will default to 0. The size of the actual I/O produced and consumed data is determined by the client upon initial connection establishment. Since a data word utilizes 2 bytes, the size must be an even number of bytes. The I/O data format is summarized in Table 31.

Description

This 32-character (max) field is strictly for user reference: it is not used at any time by the driver.

Produced Data Word Offset

The value from the associated inverter function code will populate this word offset of the produced data that is to be sent to the client. It is recommend to start at word offset 0.

Consumed Data Word Offset

The consumed data received from the client at this word offset will contain the value to be written to the associated inverter function code. It is recommend to start at word offset 0.

Function Code

The inverter function code (refer to section 4) associated with the word offset. For the Produced Data Word object, enter a "status" function code to be monitored. For the Consumed Data Word object, enter a "command" function code that can be written.

Data Type

Each data word is fixed to 16-Bit Unsigned. This is equivalent to two bytes.

Table 31: EtherNet/IP User-Configurable I/O Data Format

Consumed Data (PLC to Inverter)		Produced Data (Inverter to PLC)	
Word Offset	Function Code	Word Offset	Function Code
0	Any	0	Any
1	Any	1	Any
:	Any	:	Any
30	Any	30	Any
31	Any	31	Any

The default I/O configuration is described in Table 32.



Note Always use the studio to confirm the configuration before commissioning the device

Table 32: EtherNet/IP Default User-Configurable I/O Data Format

Consumed Data (PLC to Inverter)		Produced Data (Inverter to PLC)	
Word Offset	Function Code	Word Offset	Function Code
0	P100.01 (Command word)	0	P100.51 (Status word)
1	P100.20 (Frequency command)	1	P100.70 (Status word 2)
2	P100.21 (PID reference command)	2	P100.71 (Running frequency)
3	P100.22 (PID feedback command)	3	P100.72 (Frequency reference)
4	P100.23 (Torque command)	4	P100.73 (Bus voltage)
5	P100.24 (Voltage command)	5	P100.74 (Output voltage)
6	P100.02 (Motor group selection)	6	P100.75 (Output current)
7	None	7	P100.76 (Output torque)
8	None	8	P100.77 (Output power)
9	None	9	P100.79 (PID reference)
10	None	10	P100.80 (PID feedback)
11	None	11	P100.78 (Fault code)
12	None	12	P100.53 (Operation mode)
13	None	13	P100.52 (Motor group feedback)
:	None	:	None
30	None	30	None
31	None	31	None

9.2.5 Generic Class 1 (I/O) Connection Access

Clients may access the class 1 endpoint by opening a connection to assembly instances 100 and 150. The structure of I/O consumed and produced data for this assembly instance pair is entirely user-configurable (refer to section 9.2.4). The generic class 1 I/O connection is mutually exclusive of the AC/DC drive profile class 1 I/O connection. For a generic class 1 I/O application example, refer to section 9.2.11.

9.2.6 AC/DC Drive Profile Class 1 (I/O) Connection Access

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 structure of I/O consumed and produced data for the AC/DC drive profile class 1 I/O is predefined and fixed to 4 input bytes and 4 output bytes (refer to Table 34 and Table 35). It is highly recommended to complete the reading of this section to understand the data mapping and the implications of using the AC/DC drive profile. Note that when using the AC/DC drive profile class 1 I/O, the produced word and consumed word configuration do not apply (refer to section 9.2.4). For an AC/DC drive profile class 1 I/O application example, refer to section 9.2.12.2.

The AC/DC drive profile implementation provides support for several required CIP objects, which are specified in Table 33. 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 33: 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

Table 34: 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 P100.01 (command word), bit 0 (start/stop bit) and bit 1 (forward/reverse bit).

Run Rev: reverse rotation command (0=reverse rotation off, 1=reverse rotation on). Maps to inverter function code P100.01 (command word), bit 0 (start/stop bit) and bit 1 (forward/reverse bit).

Fault Reset: Inverter reset command (0=no action, 0→1 rising edge=reset). Maps to inverter function code P100.01 (command word), bit 15 (reset fault bit).

NetCtrl: Run/stop control source selection (0=local control, 1=network control). Maps to inverter function code P00.01 (running command channel).

NetRef: Speed reference source selection (0=local control, 1=network control). Maps to inverter function code P00.02 (communication running command channel).

Speed Reference: Inverter speed reference in RPM. Maps to function code P100.20 (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 4 in order to determine the appropriate frequency command value (in units of Hz) to be written to function code P100.20.

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

Equation 4

The “number of motor poles” term which appears in the numerator of Equation 4 is obtained from the setting of inverter function code P02.17 (Number of pole pairs of synchronous motor 1) multiplied by 2. Note that the value of P02.17 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.

Table 35: 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 P100.51, bit 2 (status word, inverter fault 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 P100.51 (status word), bit 0 (running/stopped bit) and bit 1 (forward/reverse bit).

Running2 (REV): Running reverse status signal (0=not running reverse, 1=running reverse). Maps to function code P100.51 (status word), bit 0 (running/stopped bit) and bit 1 (forward/reverse 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: Inverter command reference status (0=command reference is not from network, 1=command reference is from network). Maps to inverter function code P100.53 (operation mode status). CtrlFromNet will be 1 whenever the operation mode status is a value of 3 (communication mode), and will be 0 otherwise.

RefFromNet: Inverter speed reference status (0=speed reference is not from network, 1=speed reference is from network). Same as NetRef value.

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 the output frequency (function code P100.70) is at least the FDT level (function code P08.32).

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 P100.70 (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 5 in order to determine the appropriate operating speed (in units of RPM) to be written to the network.

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

The “number of motor poles” term which appears in the denominator of Equation 5 is obtained from the setting of inverter function code P02.17 (Number of pole pairs of synchronous motor 1). Note that the value of P02.17 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.

9.2.7 Explicit Messaging Via Get/Set Attribute Single Services

Get attribute single (0x0E) and set attribute single (0x10) are common services that can access the inverter function codes by specifying the appropriate class code, instance number and attribute identifier.

The class code is 0xA2 (Drive Parameter Object class). The instance number is the register address (of the targeted inverter function code, refer to section 4.1) + 1. The attribute identifier is 1, which is the 16-bit value of the function code being accessed. Examples are shown in Table 36.

Table 36: Get/Set Attribute Single Examples

Function Code	Register Address	Service	Class	Instance	Attribute
P17.01 (Output frequency)	1701	0x0E / 0x10	0xA2	$1701 + 1 = 1702$	1
P00.11 (Acceleration time 1)	11	0x0E / 0x10	0xA2	$11 + 1 = 12$	1
P08.09 (Jump frequency 1)	809	0x0E / 0x10	0xA2	$809 + 1 = 810$	1

9.2.8 Explicit Messaging Via Data Table Read/Write Services

Data table read (0x4C) and data table write (0x4D) services provide a direct method of accessing the inverter function codes by reference to “tag names”. Tags are read via the EtherNet/IP “data table read” service, and written via the EtherNet/IP “data table write” service.

To read data, the client must reference a starting “source element” and the “number of elements” to read. Similarly, to write data, the client must reference a starting “destination element” and the “number of elements” to write. The “number of elements” can be any quantity from 1 to the maximum allowable length, while the “source element” and “destination element” must be tag names constructed according to the naming conventions shown in section 9.2.9. The elements are 16-bit values.

9.2.9 Inverter Function Code Access Tag Format

Any inverter function code (refer to section 4) 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. The “tag name” is essentially the ASCII representation of the function code itself. Tag names are generated according to the following structure:

[function code group]_[function code offset]

Where

[function code group] is a [3 to 4]-character field, and is the ASCII character(s) for the function code's group. The characters are case-sensitive. Refer to Table 19.

[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”.

Table 37: Data Table Read/Write Examples

Function Code	Service	Tag
P17.01 (Output frequency)	0x4C / 0x4D	P17_01
P00.11 (Acceleration time 1)	0x4C / 0x4D	P00_11
P08.09 (Jump frequency 1)	0x4C / 0x4D	P08_09
P100.51 (Status word)	0x4C / 0x4D	P100_51

For explicit messaging examples, refer to sections 9.2.13 and 9.2.14.

9.2.10 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) Select the "1756-ENBT/A", and click "Create". Refer to Figure 19.

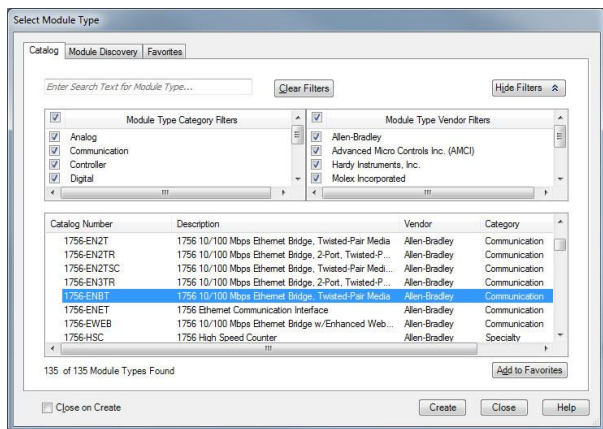


Figure 19: Adding a New Module

- 6) The "New Module" window will open. Refer to Figure 20.

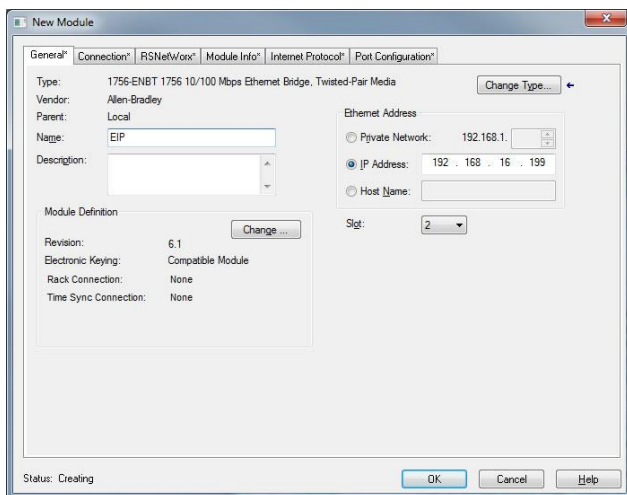


Figure 20: Identifying the New Module

- 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 Internet Protocol tab from the Module Properties dialog box and confirm that the IP Settings are configured correctly.

9.2.11 ControlLogix Example: EDS Add-On Profile (AOP)

This section will demonstrate how to setup and use an EtherNet/IP I/O connection via EDS Add-On Profile. This example only applies to RSLogix5000 V20 (and later) that support EDS Add-On Profile. Otherwise, refer to the I/O examples in section 9.2.12. This section must be completed prior to attempting any of the following AOP example(s).

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) Register the interface card's EDS file. In the menu bar, navigate to Tools...EDS Hardware Installation Tool. Refer to Figure 21.

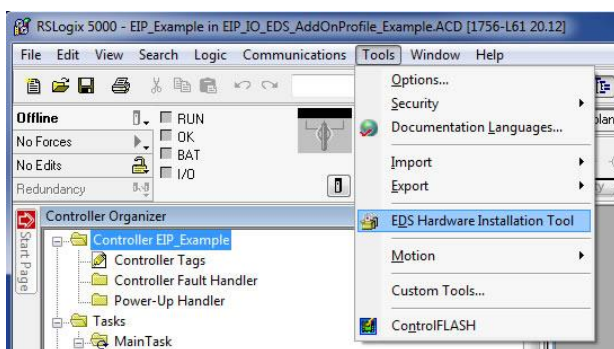


Figure 21: EDS Hardware Installation Tool Menu

- 2) This will start the "EDS Wizard". Click "Next".
- 3) Select "Register an EDS file(s)" and click "Next".
- 4) The "Registration" dialog will appear. Refer to Figure 22.

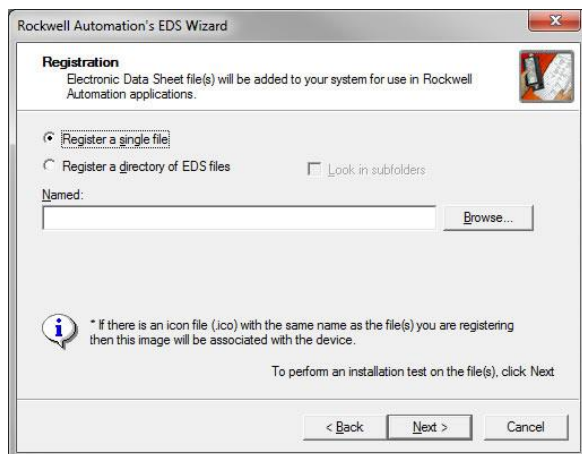


Figure 22: EDS Registration

Click "Browse", select the interface card's EDS file, and click "Next".

- 5) Ensure that there are no errors in the test results. Click "Next".
- 6) A graphic image of the interface card is displayed. Click "Next".
- 7) The task summary will list the interface card as the device to register. Click "Next".
- 8) "You have successfully completed the EDS Wizard". Click "Finish".
- 9) The interface card is now available as a module.
- 10) Right click on the 1756-ENBT/A node under the "I/O Configuration" in the controller organizer view and choose "New Module..."
- 11) Find the interface card in the "Select Module" dialog box as shown in Figure 23.

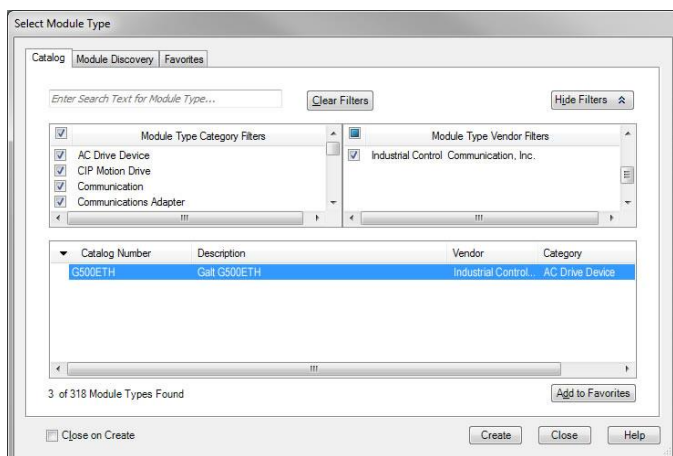


Figure 23: Adding a New Interface Card Module

Select the interface card and click "Create".

- 12) The “New Module” properties dialog box will open as shown in Figure 24.

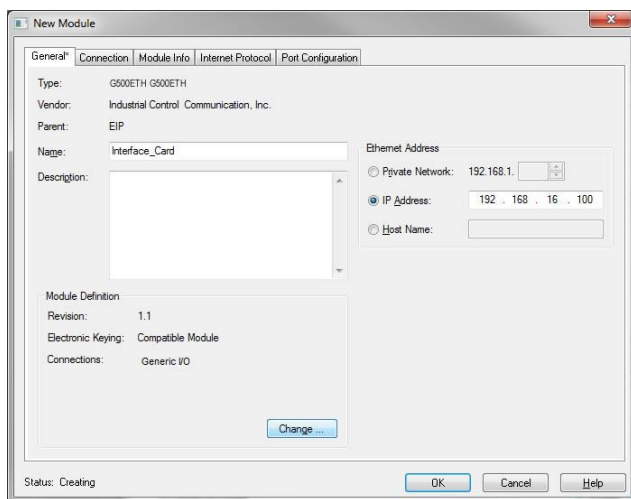


Figure 24: AOP Interface Card Module Properties

Enter a “Name” which will allow easy identification of the inverter on the network (the tags created in RSLogix 5000 will be derived from this “Name”). Enter the “IP address” of the targeted interface card.

- 13) Click on the “Connection” tab. Refer to Figure 25.

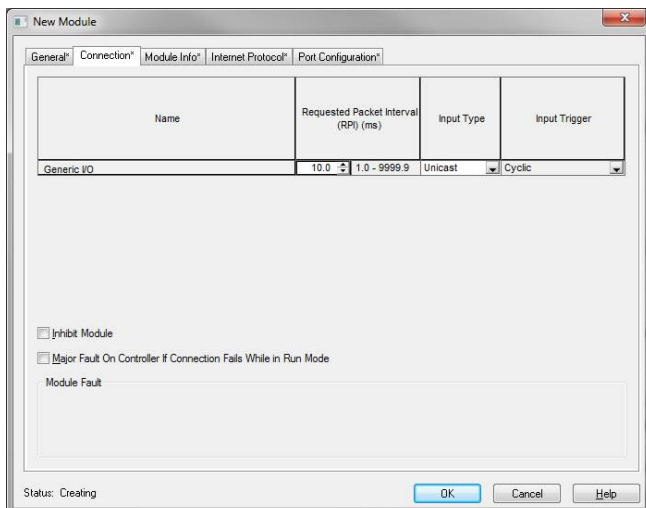


Figure 25: AOP New Module Properties Connection Tab

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 1ms. Click “OK” when done.

- 14) You should now see the interface card in the 1756-ENBT/A branch under the I/O Configuration in the controller organizer view. The full I/O Configuration tree should appear similar to Figure 26.

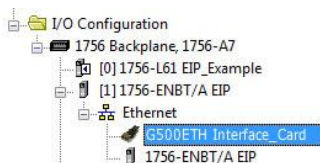


Figure 26: AOP Interface Card I/O Configuration

- 15) Continue with the AOP Generic I/O Messaging example in section 9.2.11.1 or AOP AC/DC Drive Profile example in section 9.2.11.2.

9.2.11.1 **ControlLogix Example: EDS Add-On Profile (AOP) Generic I/O Messaging**

This section will demonstrate how to configure the EtherNet/IP Generic I/O connection.

- 1) Complete all steps in section 9.2.11.
- 2) Locate the interface card in the 1756-ENBT/A branch under the "I/O Configuration" in the controller organizer view. Right click on the interface card, choose "Properties", and select the "General" tab.
- 3) Configure the Generic I/O connection. Refer to Figure 27.

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

Name: In this example, select Generic I/O.

Size: Because all inverter data is stored as 16-bit function codes, change the data type to "INT array".

Input: The Input 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 9.2.4. The Input 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 default produced data word configuration, with 3 relevant function codes (P100.51, P100.70, and P100.71). We therefore set the Input Size to 3 Words.

Output: The Output 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 9.2.4. The Output 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 default consumed data word configuration, with 2 relevant function codes (P100.01 and P100.20). We therefore set the Output Size to 2 Words.

When done, click "OK".

- 4) 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. Also confirm that the interface card's "Network Status" LED should be solid green, indicating an "online/connected" state.
- 5) By double-clicking "Controller Tags" in the project tree, it is possible to view the newly-added tags. The Interface_Card:I tag allows viewing of the input data, and the Interface_Card: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.

9.2.11.2 **ControlLogix Example: EDS Add-On Profile (AOP) AC/DC Drive Profile**

This section will demonstrate how to configure the EtherNet/IP AC/DC drive profile I/O connection.

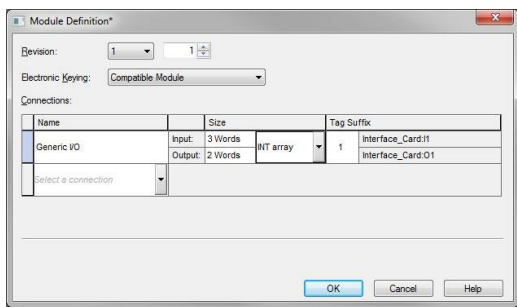


Figure 27: AOP Generic I/O Module Definition

- 1) Complete all steps in section 9.2.11.
- 2) Locate the interface card in the 1756-ENBT/A branch under the "I/O Configuration" in the controller organizer view. Right click on the interface card, choose "Properties", and select the "General" tab.
- 3) Configure the AC/DC Drive Profile connection. Refer to Figure 28.

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

Name: In this example, select AC/DC Drive Profile 21 71.

Size: Because all inverter data is stored as 16-bit function codes, change the data type to "INT array".

When done, click "OK".

- 4) 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. Also confirm that the interface card's "Network Status" LED should be solid green, indicating an "online/connected" state.
- 5) By double-clicking "Controller Tags" in the project tree, it is possible to view the newly-added tags. The Interface_Card:I tag allows viewing of the input data, and the Interface_Card: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. The AC/DC drive profile I/O data is described in section 9.2.6.

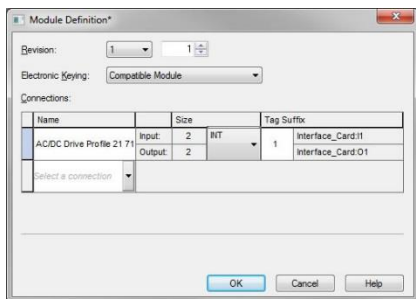


Figure 28: AOP AC/DC Drive Profile Module Definition

9.2.12 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 or 20 & 70 or 20 & 71. 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 "Create". Refer to Figure 29.

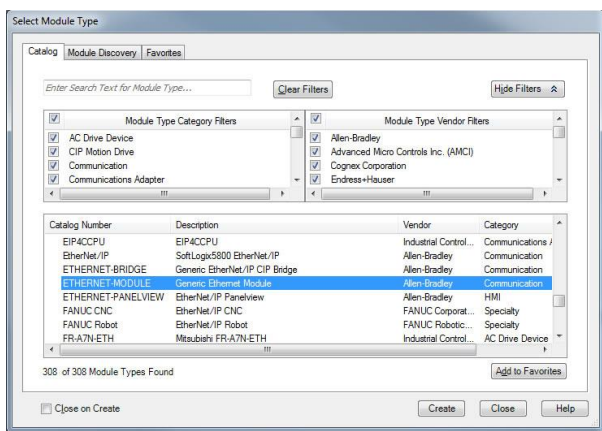


Figure 29: Adding a New Generic Ethernet Module

- 4) The module properties dialog box will open (refer to Figure 30). Enter a Name 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 codes, change the "Comm Format" selection to "Data-INT". Enter the IP address of the targeted interface card.

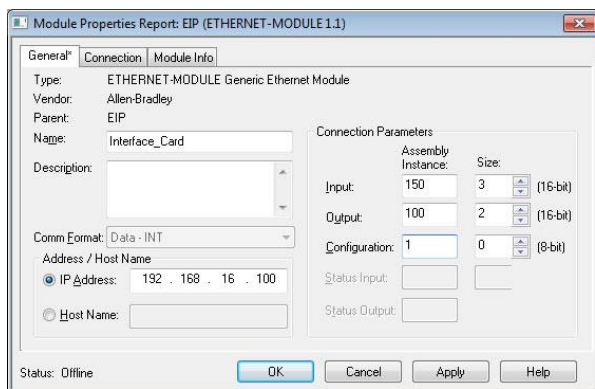


Figure 30: Interface Card Module Properties

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 9.2.4. The Input Assembly Instance must be set to 150 when connecting to the generic 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 default produced data word configuration, with three relevant function codes (P100.51, P100.70, and P100.71). 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 9.2.4. The Output Assembly Instance must be set to 100 when connecting to the generic 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 default consumed data word configuration, with two relevant function codes (P100.01 and P100.20). 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 Interface_Card”) 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 31.

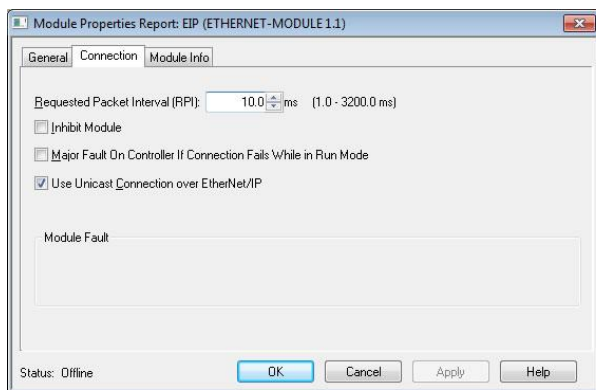


Figure 31: Module Properties Connection Tab

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 1ms. Click OK when done.

- 6) After adding the I/O Module to the configuration, the full I/O Configuration tree should appear similar to Figure 32.
- 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 33. Also confirm that the interface card's “Network Status” LED should be solid green, indicating an “online/connected” state.

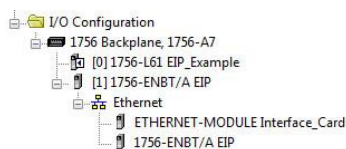


Figure 32: I/O Configuration Tree

- 8) By double-clicking "Controller Tags" in the project tree, it is possible to view the newly-added tags. Refer to Figure 34. The Interface_Card:C configuration tag is unused, the Interface_Card:I tag allows viewing of the input data, and the Interface_Card: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.
- 9) By double-clicking "Controller Tags" in the project tree, it is possible to view the newly-added tags. Refer to Figure 34. The Interface_Card:C configuration tag is unused, the Interface_Card:I tag allows viewing of the input data, and the Interface_Card: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.

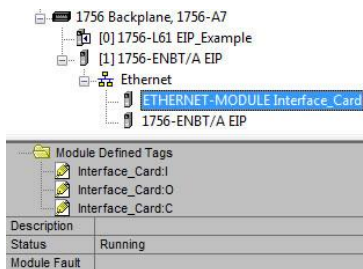


Figure 33: Online Module Status

Name	Value	Force Mask	Style	Data Type
Interface_Card:C	{...}	{...}		AB:ETHERNET_MODULE:C:0
Interface_Card:I	{...}	{...}		AB:ETHERNET_MODULE_INT_6bytes:I:0
Interface_Card:I.Data	{...}	{...}		
Interface_Card:I.Data[0]	16#0001		Hex	INT
Interface_Card:I.Data[1]	16#0108		Hex	INT
Interface_Card:I.Data[2]	3558		Decimal	INT
Interface_Card:O	{...}	{...}		AB:ETHERNET_MODULE_INT_4bytes:O:0
Interface_Card:O.Data	{...}	{...}		
Interface_Card:O.Data[0]	16#0001		Hex	INT
Interface_Card:O.Data[1]	3558		Decimal	INT

Figure 34: Controller Tags for I/O Access

We can directly interact with these tags in order to control and monitor the inverter. In Figure 34, for example, we can see that the first 16-bit word of output data (Interface_Card:O.Data[0]) has been set to a hexadecimal value of 0x0001. The default consumed data word configuration word offset 0 references function code P100.01, which is the inverter's command register. A value of 0x0001, therefore, means that the start bit has been turned ON.

Similarly, we can see that the second 16-bit word of output data (Interface_Card:O.Data[1]) has been set to a decimal value of 3558. The default consumed data word configuration word offset 1 references function code P100.20, 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 0x0001, 0x0108 and 3558 corresponding to P100.51 (status word), P100.70 (status word2) and P100.71 (output frequency), respectively, are consistent with the inverter running at the parameters commanded by the output tag.

9.2.12.1 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 (refer to section 9.2.4). The add-on instruction is optional and provided for user convenience.

- 1) Complete all the steps in section 9.2.11.
- 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 35.

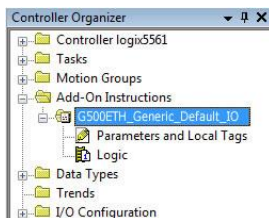


Figure 35: Generic Default IO Add-On Instruction

- 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 36.

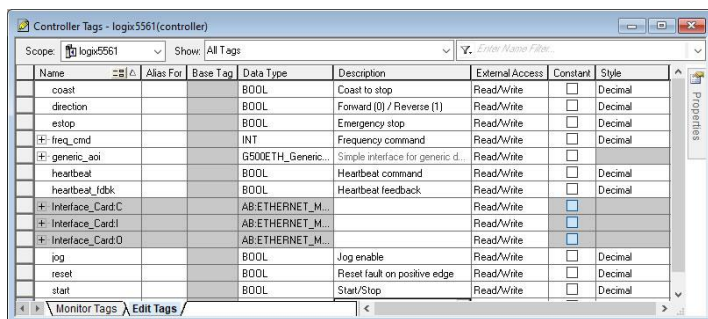


Figure 36: Create Generic Default AOI Tags

- 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 37.

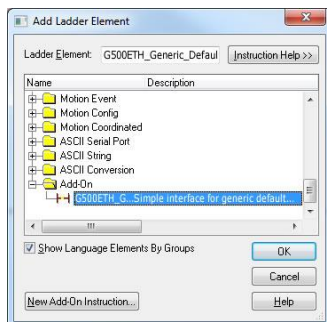


Figure 37: Add Generic Default Add-On Instruction

- 9) Click OK.
- 10) Edit the add-on instruction according to Figure 38.

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

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. The add-on instruction is optional and provided for user convenience.

- 1) Complete all the steps in section 9.2.11. 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 39.

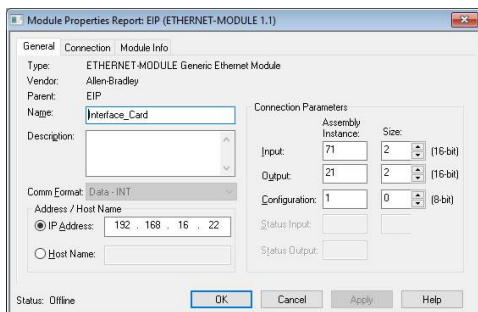


Figure 39: AC/DC Drive Profile Generic Ethernet Module Configuration

- 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 40.
- 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 41.

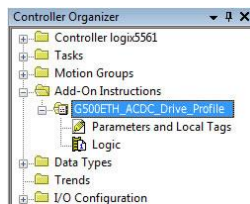


Figure 40: AC/DC Drive Profile Add-On Instruction

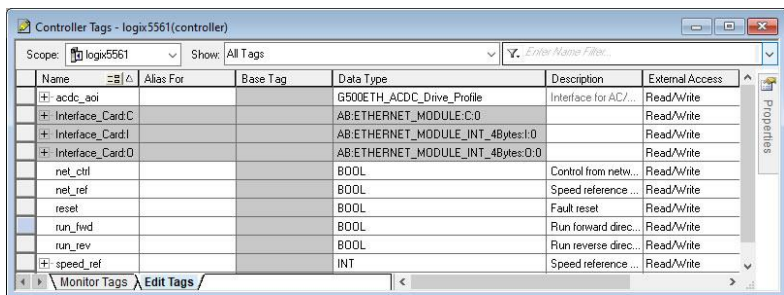


Figure 41: Create AC/DC Drive Profile AOI Tags

- 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 42.

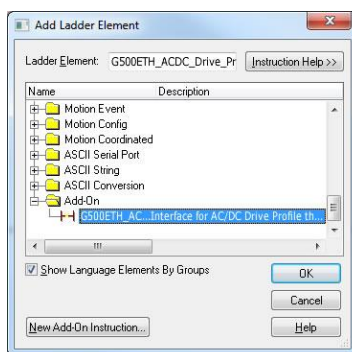


Figure 42: Add AC/DC Drive Profile Add-On Instruction

- 9) Click OK.
- 10) Edit the add-on instruction according to Figure 43.

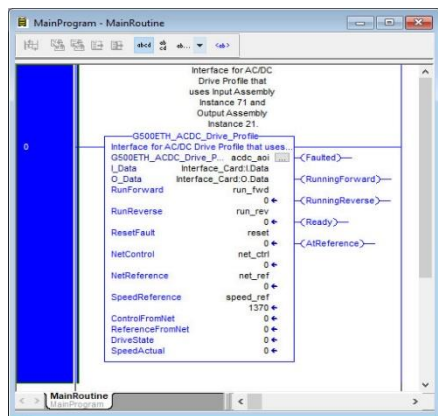


Figure 43: Configure AC/DC Drive Profile AOI

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

9.2.13 ControlLogix Example: Read a Block of Function Codes

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.

- Double click "Controller Tags" in the controller organizer view.
- The "Controller Tags" window appears. Refer to Figure 44.

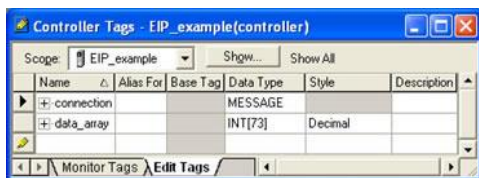


Figure 44: Create New Tags

- Select the "Edit Tags" tab at the bottom.
- 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.
- 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.
- Collapse the "connection" tag again by clicking on the "-" sign next to the tag name.
- 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.

- Double click "MainRoutine" under Tasks ...MainTask ...MainProgram in the controller organizer view.
- Right click on the first ladder logic rung in the MainRoutine window and select "Add Ladder Element..."
- The "Add Ladder Element" window appears.
- Select the "MSG" instruction in the Input/Output folder. Refer to Figure 45.
- Click OK.

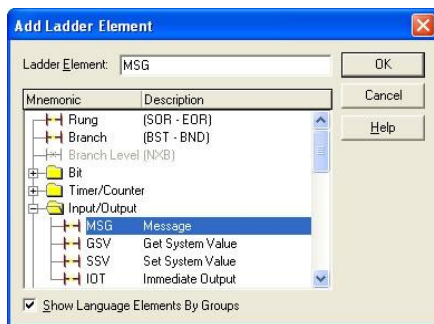


Figure 45: Adding a MSG Instruction

3) Add an XIO element to the main program.

- Right click on the ladder logic rung containing the MSG instruction in the MainRoutine window and select "Add Ladder Element..." again.
- The "Add Ladder Element" window appears.
- Select the "XIO" element in the Bit folder. Refer to Figure 46.

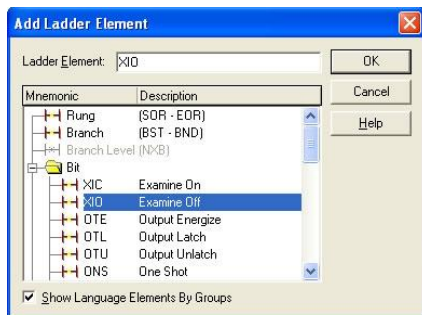


Figure 46: Adding an XIO Element

- d) Click OK.

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 47.

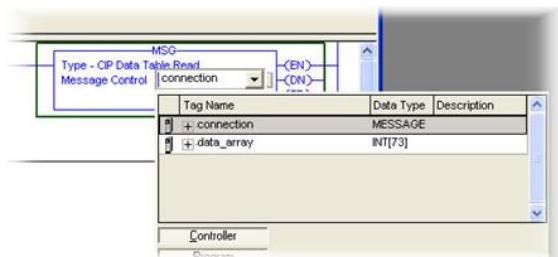


Figure 47: MSG Instruction Tag Assignment

- b) Click the message configuration button ("...") in the MSG instruction. The "Message Configuration" window will open. Refer to Figure 48.



Figure 48: 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 9.2.9). In this example, we will be reading a total of 21 function codes beginning at function code P17.01 (output frequency).
 - Enter the Number Of Elements to read. In this example, we will read 21 function codes.
 - For the Destination Element, select "data_array[50]".
- d) "Communication" tab settings (refer to Figure 49):

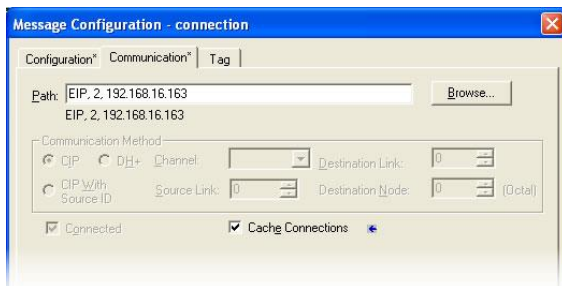


Figure 49: Setting the Communication Path

- i) 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 9.2.10),
 - 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,2,192.168.16.163".
- ii) 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".
- e) Click "OK" to close the MSG Configuration dialog. At this stage, MainRoutine should look like Figure 50.

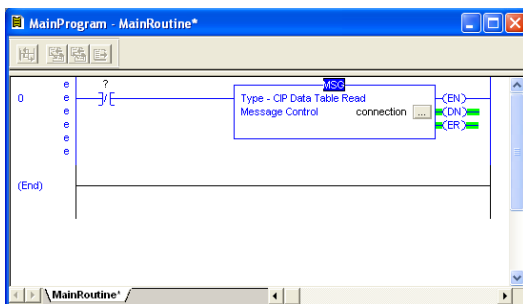


Figure 50: MainRoutine

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 51. 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.

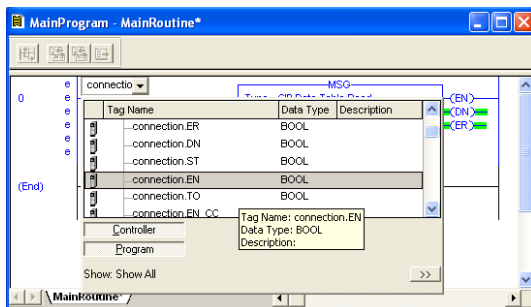


Figure 51: Configure XIO Element

6) The program is now complete. Refer to Figure 52.

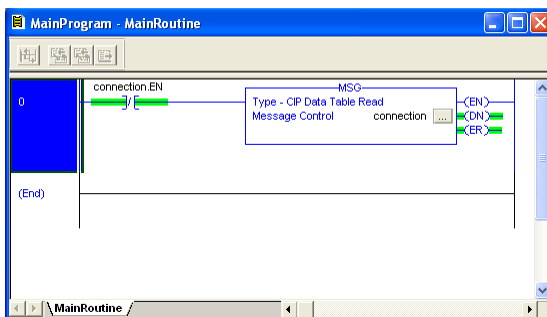


Figure 52: Complete Program

7) Save, download and run the program.

- To view the values of the function codes being read from the interface card, double-click "Controller Tags" in the controller organizer view.
- Select the "Monitor Tags" tab and expand the data_array tag.
- 21 function code values starting at function code P17.01 are being continuously read from the interface card and placed in the 21 sequential offsets of data_array starting at the 50th offset (data_array[50]).

9.2.14 ControlLogix Example: Reading and Writing MSG Instructions

Often times, applications may need to both read data from and write data to the inverter. 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.

Figure 53 shows an example of three MSG instructions, one for reading and two for writing (the inverter's frequency command and command word). Note the addition of the en_xx_wr XIC elements for the write logic. 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.

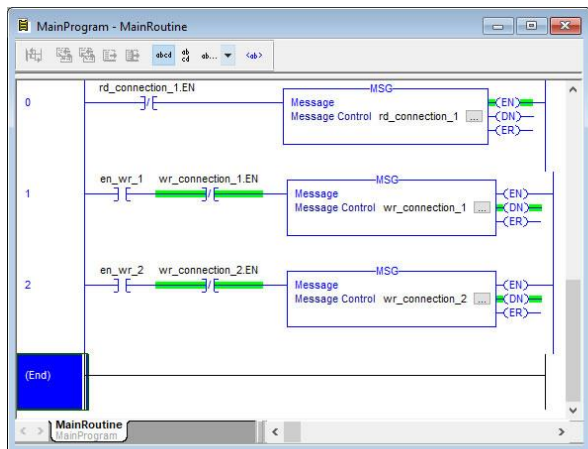


Figure 53: Reading and Writing via MSG Instructions

Figure 54 shows the configuration details of the example wr_connection_2 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 P100.20). The Source Element in this case is the 2nd element (starting from index 0) of an INT array tag named "wr_data".

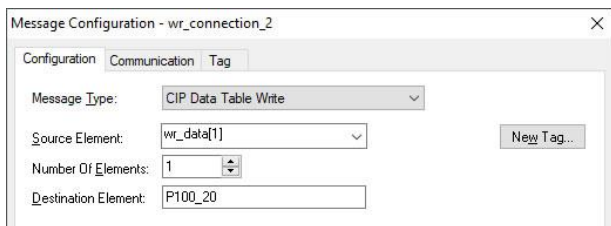


Figure 54: MSG Configuration for Writing

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 9.2.11 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.

9.3 Allen Bradley CSP (PCCC) Server

9.3.1 Overview

Ethernet-enabled Allen-Bradley legacy PLCs (such as the PLC5E, SLC-5/05, and MicroLogix 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. Note that CSP runs under EtherNet/IP and is enabled by default when EtherNet/IP is added to the configuration.

If a connection timeout or socket-level error occurs, the driver can be configured to perform a timeout action as described in section 9.2.3.

9.3.2 Explicit Messaging Via Read/Write Services

Function code 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. The supported read and write services are listed in Table 38. To read and write data, the client must reference a “target address” and the “size of elements”. The target address is constructed according to the conventions shown in section 9.3.3.

Table 38: CSP (PCCC) Read/Write Services

Service	Code
PLC5 Typed Read	0x68
PLC5 Typed Write	0x67
PLC5 Word Range Read	0x01
PLC5 Word Range Write	0x00
SLC Typed Read	0xA2
SLC Typed Write	0xAA

9.3.3 Inverter Function Code File Number Offset Format

The formulas to calculate the file number and offset from the function code are provided using Equation 6 and Equation 7.

$$\text{File number} = \text{Function code group number} + 10 \quad \text{Equation 6}$$

$$\text{File offset} = \text{Function code offset} \quad \text{Equation 7}$$

Refer to section 4.1 for converting function codes to register addresses. In Equation 6 and Equation 7, “file number” and “file offset” is restricted only by the limitations of the programming software. Refer to section 4.1 for the function code to register mapping. Table 39 provides some examples of various combinations of file/section numbers and offsets/elements which can be used to access inverter function codes. Note that there are multiple different combinations of file/section numbers and offsets/elements that will result in the same inverter function code being accessed.

Table 39: CSP Target Register Examples

Function Code	Function Code Group Number	File/Section Number (Equation 6)	Function Code Offset	File Offset/Element	Address Format
P00.01	0	$0 + 10 = \text{N10}$	1	1	N10:1
P08.09	8	$8 + 10 = \text{N18}$	9	9	N18:9
P17.02	17	$17 + 10 = \text{N27}$	2	2	N27:2
P26.47	26	$26 + 10 = \text{N36}$	47	47	N36:47

In addition to providing access to the inverter function codes in their “standard” numerical locations as mentioned above, the function codes can also be accessed in a special “assembly object” type format by targeting integer file N111. What this means is that when N111 is targeted for reading, what is actually returned by the interface card is the user-defined function code data as ordered by the EtherNet/IP produced data word configuration (refer to section 9.2.4). Similarly, when N111 is targeted for writing, the written data is disseminated to the inverter’s function codes according to the definition contained in the EtherNet/IP consumed data word configuration. By appropriate configuration of the EtherNet/IP consumed and produced data word configuration, therefore, bulk access to non-contiguous but frequently-used inverter function codes can be conveniently provided by performing only one read and/or write instruction targeting file N111.

Because both the EtherNet/IP consumed and produced data word configurations are comprised of 32 function code definitions, the targeted “offset/element” must be within the range of 0 to 31 inclusive. Refer to Table 40 for some examples of N111 accesses.

Table 40: Examples of EtherNet/IP-Style Bulk Access via File N111

File/Section Number	Offset/Element	Address Format	Start Target Function Code of Configuration Array	Max Number of Accessible Elements
N111	0	N111:0	1st	32
N111	:	:	:	:
N111	15	N111:15	16th	16
N111	:	:	:	:
N111	31	N111:31	32nd	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 6 and Equation 7. RSLogix500 examples are shown in the following sections.

9.3.4 SLC-5/05 Example: Read Function Codes

This example program will show how to continuously read a block of function codes from the inverter with a single MSG instruction. This action is performed via the Typed Read (a.k.a. "PLC5 Read") message type. Only one read request is outstanding at any given time. Note that the steps for the MicroLogix and PLC5E may vary slightly, but in general are similar.

1) Run RSLogix 500, and create a new configuration.

2) Create a control and a data file.

- Right click Data Files and select New... The "Create Data File" dialog box appears (refer to Figure 55).
- 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.
- 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 56. Click OK to create the file.

The "Create Data File" dialog box is shown. It has a title bar with a close button. The fields are: File: 20, Type: Integer (dropdown), Name: CONTROL, Desc: (empty), Elements: 100, Last: (empty). Under Attributes, there are checkboxes for Debug and Skip When Deleting Unused Memory, both unchecked. Under Scope, there are radio buttons for Global (selected) and Local, and a To File: 2: dropdown. Under Protection, there are radio buttons for Constant, Static, and None (selected), and a checkbox for Memory Module which is unchecked. At the bottom are OK, Cancel, and Help buttons.

Figure 55: Creating a Control File

3) Add a MSG instruction to the program.

- If not already visible, double-click "LAD2" under Project...Program Files in the controller organizer view to bring up the ladder logic program.
- Right click on the default rung number on the left-hand side of the LAD2 window and select "Insert Rung".
- Right click on the rung number of the new editable rung and select "Append Instruction".
- Select the "MSG" instruction from the "Input/Output" classification, then click OK. Refer to Figure 57.

4) Add an XIO element to the program.

- Right click on the rung number of the rung currently being edited and select "Append Instruction" again.

The "Create Data File" dialog box is shown. It has a title bar with a close button. The fields are: File: 18, Type: Integer (dropdown), Name: DATA, Desc: (empty), Elements: 200, Last: (empty). Under Attributes, there are checkboxes for Debug and Skip When Deleting Unused Memory, both unchecked. Under Scope, there are radio buttons for Global (selected) and Local, and a To File: 2: dropdown. Under Protection, there are radio buttons for Constant, Static, and None (selected), and a checkbox for Memory Module which is unchecked. At the bottom are OK, Cancel, and Help buttons.

Figure 56: Creating a Data File

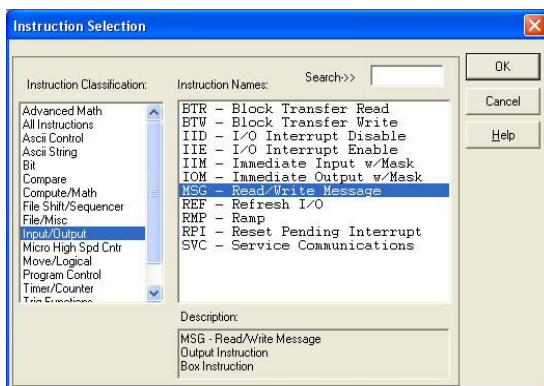


Figure 57: MSG Instruction Selection

- b) Select the “XIO” instruction from the “Bit” classification, then click OK. Refer to Figure 58.

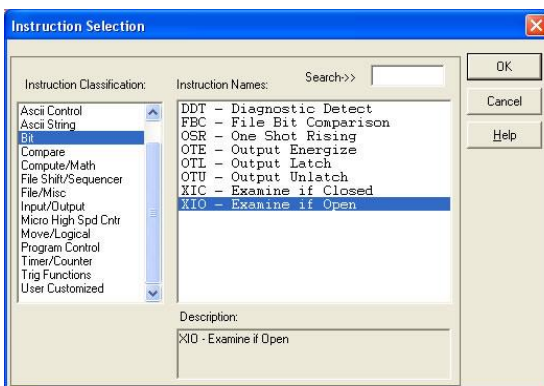


Figure 58: XIO Instruction Selection

5) Configure the MSG instruction.

- a) Set the “Read/Write” field to “Read”, “Target Device” field to “PLC5”, “Local/Remote” field to “Local”, and “Control Block” to “N20:0”.
- b) 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 59.

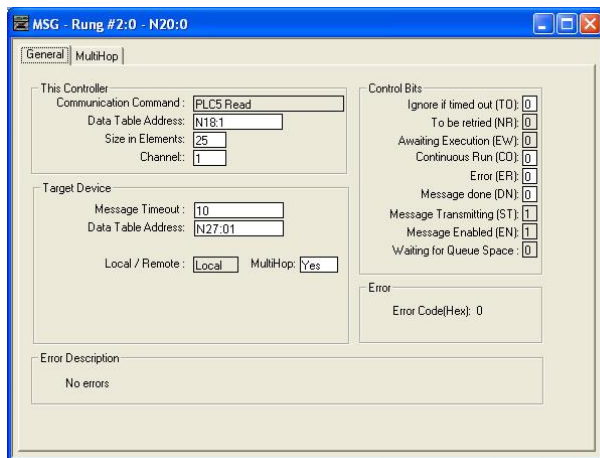


Figure 59: MSG Configuration for Reading, "General" Tab

- c) In this example, we will be reading a total of 25 function codes beginning at N27:01 (function code P17.01). 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).
- d) Under "Target Device", set the "Data Table Address" field to N27:01 (starting function code P17.01) and set the "MultiHop" field to Yes to cause the "MultiHop" tab to appear.
- e) 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 60.

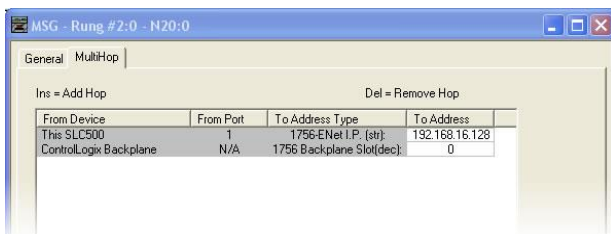


Figure 60: MSG Configuration, "MultiHop" Tab

- f) Close the dialog box. At this point, the program should appear as shown in Figure 61.

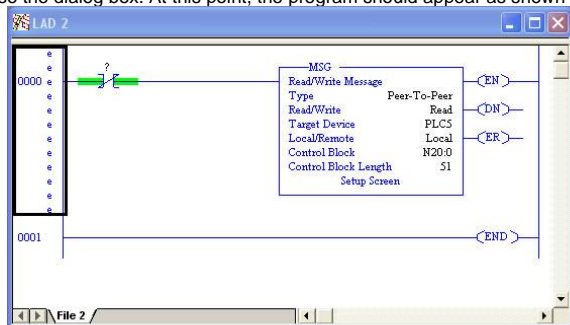


Figure 61: PLC Program after MSG Instruction Configuration

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 62.

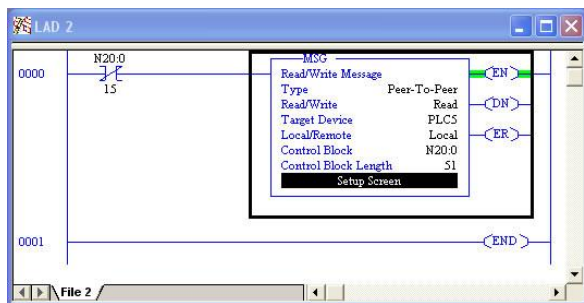


Figure 62: Completed PLC Program

8) Save, download, and run the program.

- a) To view the function codes being read from the interface card, double-click the data file N18 under "Data Files" in the controller organizer view. 25 function code values starting at function code P17.01 are being continuously read from the interface card and placed in the 25 sequential offsets of N18 starting at N18:1. Refer to Figure 63.

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 63: Monitoring the Data Being Read from the Inverter

9.3.5 SLC-5/05 Example: Reading and Writing

Often times, applications may need to both read data from and write data to the inverter. 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 control file.

Figure 64 shows an example of two MSG instructions, one for reading and one for writing. It is evident from this logic that N20.0 and N21.0 are the two independent message control files created for these instructions. Note that the “Read/Write” field of each of the MSG instructions is set according to their function.

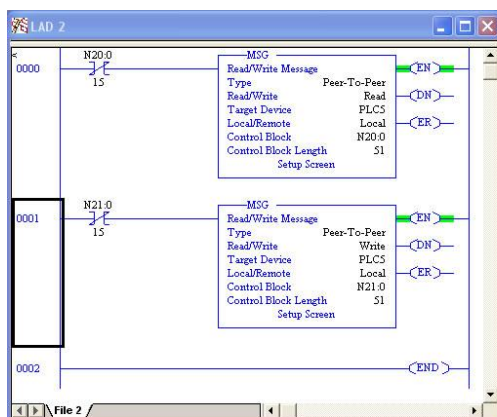


Figure 64: Reading and Writing via MSG Instructions

Figure 65 shows the configuration details of the “write” MSG instruction. Note that this instruction will only be writing to one inverter function code: namely, P100.20. The source Data Table Address in this case is N110:20.

Figure 65: MSG Configuration for Writing

9.4 BACnet/IP Server

- The interface card supports the BACnet/IP (Annex J) protocol over Ethernet via a configurable UDP port (default value of 47808).
- The BACnet driver does not trigger timeout events (section 6.7.1).

9.4.1 Protocol Implementation Conformance Statement

BACnet Protocol

Date: January 11, 2020
 Vendor Name: ICC, Inc.
 Product Name: Galt Electric G500
 Product Model Number: G500ETH
 Applications Software Version: V1.1.1
 Firmware Revision: V1.1.1
 BACnet Protocol Revision: 2
 Product Description:

The G500ETH Multiprotocol Ethernet interface allows information to be transferred seamlessly between a G500 inverter and several Ethernet-based fieldbus networks.

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-DDB-B)
☒ Device Management – Dynamic Object Binding-B (DM-DOB-B)

Segmentation Capability:

None

- ☐ Segmented requests supported Window Size _____
☐ 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
Analog Value	Real	Real, Unsigned, Integer, Null
Binary Output	Enumerated	Enumerated, Boolean, Real, Unsigned, Integer, Null
Binary Input	Enumerated	N/A
Binary Value	Enumerated	Enumerated, Boolean, Real, Unsigned, Integer, Null
Multi-state Output	Unsigned	Enumerated, Real, Unsigned, Integer, Null
Multi-state Input	Unsigned	N/A
Multi-state Value	Unsigned	Enumerated, Real, Unsigned, Integer, Null

Notes:

- The Null data type is used to relinquish a previously-commanded entry at the targeted priority in the priority array.

Object Types/Property Support Tables:

Table 41: BACnet Device Object Types /Properties Supported

Property	Object Type
	Device
Object Identifier	R
Object Name	R
Object Type	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
Device Address Binding	R
Database Revision	R

R – readable using BACnet services

W – readable and writable using BACnet services

Table 42: BACnet Binary Object Types /Properties Supported

Property	Object Type		
	Binary Input	Binary Output	Binary Value
Object Identifier	R	R	R
Object Name	R	R	R
Object Type	R	R	R
Present Value	R	W	W
Status Flags	R	R	R
Event State	R	R	R
Out-of-Service	R	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

Table 43: BACnet Analog Object Types /Properties Supported

Property	Object Type		
	Analog Input	Analog Output	Analog Value
Object Identifier	R	R	R
Object Name	R	R	R
Object Type	R	R	R
Present Value	R	W	W
Status Flags	R	R	R
Event State	R	R	R
Out-of-Service	R	R	R
Units	R	R	R
Priority Array		R	R
Relinquish Default		R	R

R – readable using BACnet services

W – readable and writable using BACnet services

Table 44: BACnet Multi-state Object Types /Properties Supported

Property	Object Type		
	Multi-state Input	Multi-state Output	Multi-state Value
Object Identifier	R	R	R
Object Name	R	R	R
Object Type	R	R	R
Present Value	R	W	W
Status Flags	R	R	R
Event State	R	R	R
Reliability	R	R	R
Out-of-Service	R	R	R
Number of States	R	R	R
Priority Array		R	R
Relinquish Default		R	R

R – readable using BACnet services

W – readable and writable using BACnet services

9.4.2 Default Supported Objects

This section will describe the default objects. Since the objects are configurable, the system integrator is responsible for managing, maintaining, and documenting the actual configuration.



Note Always use the studio to confirm the configuration before commissioning the device.

Table 45: Analog Input Object Instance Summary

Instance ID	Object Name	Description	Function Code	Units
AI1	Run Freq	Running frequency	P100.70	Hertz
AI2	Freq Ref	Frequency reference	P100.71	Hertz
AI3	Bus Voltage	DC bus voltage	P100.72	Volts
AI4	Output Voltage	Output voltage	P100.73	Volts
AI5	Output Current	Output current	P100.74	Amps
AI6	Output Torque	Actual output torque	P100.75	Percent
AI7	Output Power	Actual output power	P100.76	Percent
AI8	Fault Code	Fault code	P100.78	No Units
AI9	PID Ref	PID reference	P100.79	Percent
AI10	PID Fbk	PID feedback	P100.80	Percent
AI11	Torque Ref	Torque reference	P17.15	Percent
AI12	Operation Mode	Operation mode	P100.53	No Units
AI13	Motor Group Fbk	Motor group feedback	P100.52	No Units
AI14	Operation Time	Current running time	P17.26	Minutes
AI15	Mtr Power Factor	Motor power factor	P17.25	No Units
AI16	Power Cons Hi	Inverter power consumption (high word)	P07.15	Megawatt-hours
AI17	Power Cons Lo	Inverter power consumption (low word)	P07.16	Kilowatt-hours
AI18	Fault Hist 1	Fault History 1 (most recent)	P07.27	No Units
AI19	Fault Hist 2	Fault History 2	P07.28	No Units
AI20	Fault Hist 3	Fault History 3	P07.29	No Units
AI21	Fault Hist 4	Fault History 4	P07.30	No Units
AI22	Fault Hist 5	Fault History 5	P07.31	No Units
AI23	Fault Hist 6	Fault History 6 (least recent)	P07.32	No Units

Table 46: Analog Output Object Instance Summary

Instance ID	Object Name	Description	Function Code	Units
AO1	Freq Cmd	Frequency command	P100.20	Hertz
AO2	PID Ref Cmd	PID reference command	P100.21	Percent
AO3	PID Fbk Cmd	PID feedback command	P100.22	Percent
AO4	Torque Cmd	Torque command	P100.23	Percent
AO5	Voltage Cmd	Voltage command	P100.24	Percent
AO6	Motor Group Sel	Motor group selection	P100.02	No Units

Table 47: Analog Value Object Instance Summary

Instance ID	Object Name	Description	Function Code	Units
AV1	Accel Time1	Acceleration time 1	P00.11	Seconds
AV2	Decel Time 1	Deceleration time 1	P00.12	Seconds

Table 48: Binary Input Object Instance Summary

Instance ID	Object Name	Description	Function Code	Bit	Active Text
					Inactive Text
BI1	Running/Stopped	Inverter running status	P100.51	0	Running
					Stopped
BI2	Reverse/Forward	Running direction	P100.51	1	Reverse
					Forward
BI3	Inverter Fault	Inverter fault present	P100.51	2	Fault
					OK
BI4	POFF	System power failure	P100.51	3	POFF
					OK
BI5	Bus Volt Present	DC bus voltage established	P100.51	8	Ready
					Not Rdy
BI6	Motor Type	Motor type feedback	P100.51	11	Synchron
					Inductn
BI7	Overload Warning	Overload pre-alarm	P100.51	12	Overload
					OK
BI8	Heartbeat Fbk	Heartbeat feedback value	P100.51	15	1
					0
BI9	Jogging	Inverter jogging status	P100.70	0	Jogging
					No Jog
BI10	FDT1	Frequency level detection FDT1	P100.70	1	Yes
					No
BI11	FDT2	Frequency level detection FDT2	P100.70	2	Yes
					No
BI12	Freq Arrival	Frequency arrival signal	P100.70	3	Yes
					No
BI13	Zero Speed	Running at zero speed	P100.70	4	Yes
					No
BI14	At Upper Freq	Upper limit frequency reached	P100.70	5	Yes
					No
BI15	At Lower Freq	Lower limit frequency reached	P100.70	6	Yes
					No
BI16	Pre-excitatn Fbk	Pre-excitation enable feedback	P100.70	7	Enabled
					Disabled
BI17	Undrload Warning	Underload pre-alarm	P100.70	8	Undrload
					OK
BI18	Speed Limiting	Frequency is limited	P100.70	14	Yes
					No
BI19	Control State	Toque/speed control state	P17.40	4	Torque
					Speed

Table 49: Binary Output Object Instance Summary

Instance ID	Object Name	Description	Function Code	Bit	Active Text
					Inactive Text
BO1	Run/Stop Cmd	Run/stop command	P100.01	0	Run
					Stop
BO2	Rev/Fwd Sel	Reverse/forward selection	P100.01	1	Reverse
					Forward
BO3	Jog Operation	Jog operation enable	P100.01	2	Enable
					Disable
BO4	Coast Stop	Coast to stop command	P100.01	3	Coast
					No Actn
BO5	Emergency Stop	Emergency stop command	P100.01	4	E Stop
					No Actn
BO6	Reset Fault	Reset fault command	P100.01	5	Reset
					No Actn
BO7	Ctrl Mode Switch	Control mode switching	P100.01	11	Enable
					Disable
BO8	Clear Pwr Cons	Clear power consumption	P100.01	12	Clear
					No Actn
BO9	Pre-excitation Cmd	Pre-excitation enable command	P100.01	13	Enable
					Disable
BO10	DC Brake Cmd	DC brake enable command	P100.01	14	Enable
					Disable
BO11	Heartbeat Cmd	Heartbeat reference command	P100.01	15	1
					0
BO12	Clear Fault Hist	Clear fault history	P00.18	1	Clear
					No Actn

9.4.3 Server Settings

In the studio's Project panel, navigate to **G500ETH...Ethernet...BACnet/IP Server**.

UDP Port

This is the UDP port on which to transmit and receive BACnet/IP packets on the local subnet. The default value is 47808 (0xBAC0). To ensure successful communications, use caution when using a port setting other than the default value.

9.4.4 Node Settings

There are no node settings. A node is simply a container for objects.

9.4.5 Device Object Settings

A Device Object is automatically added to every node, and cannot be removed. The Device Object contains several configurable fields that must be appropriately set for each device residing on a BACnet network.

Device Name

Defines the node's name. The device name must be unique across the entire BACnet network. Enter a string of between 1 and 32 characters in length.

Instance Number

Defines the node's instance number. The instance number must be unique across the entire BACnet network. Enter a value between 0...4194302 (0x0...0x3FFFE).

9.4.6 BACnet Object Settings

The BACnet server hosts BACnet objects which contain many different properties for any BACnet client on the network to access. The driver supports a variety of different BACnet objects. All supported

properties of these objects are readable, while only the present value property is writable (for Outputs and Values only).

9.4.6.1 Analog Input Object Settings

Object Name

The name of the BACnet object. Enter a string of between 1 and 32 characters in length. All object names must be unique within a node.

Instance

The BACnet object's instance number. Enter a value between 0...4194302 (0x0...0x3FFFFFFE).

Function Code

The inverter function code (refer to section 4) that the BACnet object's present value will access.

Data Type

Specifies how the object's value will be stored internally in the device and whether the value should be treated as signed or unsigned.

Multiplier

The amount that associated network values are scaled by prior to being stored into the database or after being retrieved from the database. Upon retrieval from the database, raw data is multiplied by the multiplier to produce a network value (to send to a client).

Offset

The amount that associated network values are offset by prior to being stored into the database or after being retrieved from the database. Upon retrieval from the database, after applying the multiplier, data is offset by this value to produce a network value (to send to a client).

Units

Select the desired units from this dropdown menu. If the desired units are not available in the dropdown menu, select "Other Units" and enter the appropriate enumerated value (as defined by the BACnet Specification) in the "Unit Value" field.

Unit Value

This field is enabled only when the "Units" selection is set to "Other Units". Enter the appropriate enumerated value (as defined by the BACnet Specification.)

9.4.6.2 Analog Output Object Settings

Object Name

The name of the BACnet object. Enter a string of between 1 and 32 characters in length. All object names must be unique within a node.

Instance

The BACnet object's instance number. Enter a value between 0...4194302 (0x0...0x3FFFFFFE).

Function Code

The inverter function code (refer to section 4) that the BACnet object's present value will access.

Data Type

Specifies how the object's value will be stored internally in the device and whether the value should be treated as signed or unsigned.

Multiplier

The amount that associated network values are scaled by prior to being stored into the database or after being retrieved from the database. Upon retrieval from the database, raw data is multiplied by the multiplier to produce a network value (to send to a client).

Offset

The amount that associated network values are offset by prior to being stored into the database or after being retrieved from the database. Upon retrieval from the database, after applying the multiplier, data is offset by this value to produce a network value (to send to a client).

Units

Select the desired units from this dropdown menu. If the desired units are not available in the dropdown menu, select "Other Units" and enter the appropriate enumerated value (as defined by the BACnet Specification) in the "Unit Value" field.

Unit Value

This field is enabled only when the "Units" selection is set to "Other Units". Enter the appropriate enumerated value (as defined by the BACnet Specification.)

Relinquish Default

Defines the default value to be used for an object's present value property when all entries in the object's priority array are NULL.

9.4.6.3 Analog Value Object Settings

Object Name

The name of the BACnet object. Enter a string of between 1 and 32 characters in length. All object names must be unique within a node.

Instance

The BACnet object's instance number. Enter a value between 0...4194302 (0x0...0x3FFFFFFE).

Function Code

The inverter function code (refer to section 4) that the BACnet object's present value will access.

Data Type

Specifies how the object's value will be stored internally in the device and whether the value should be treated as signed or unsigned.

Multiplier

The amount that associated network values are scaled by prior to being stored into the database or after being retrieved from the database. Upon retrieval from the database, raw data is multiplied by the multiplier to produce a network value (to send to a client).

Offset

The amount that associated network values are offset by prior to being stored into the database or after being retrieved from the database. Upon retrieval from the database, after applying the multiplier, data is offset by this value to produce a network value (to send to a client).

Units

Select the desired units from this dropdown menu. If the desired units are not available in the dropdown menu, select "Other Units" and enter the appropriate enumerated value (as defined by the BACnet Specification) in the "Unit Value" field.

Unit Value

This field is enabled only when the "Units" selection is set to "Other Units". Enter the appropriate enumerated value (as defined by the BACnet Specification.)

Relinquish Default

Defines the default value to be used for an object's present value property when all entries in the object's priority array are NULL.

9.4.6.4 Binary Input Object Settings

Object Name

The name of the BACnet object. Enter a string of between 1 and 32 characters in length. All object names must be unique within a node.

Instance

The BACnet object's instance number. Enter a value between 0...4194302 (0x0...0x3FFFFFFE).

Function Code

The inverter function code (refer to section 4) that the BACnet object's present value will access.

Bitmask

Specifies which bit(s) in the 16-bit value designated by the "Function Code" that the binary object will map to. This mechanism allows up to 16 binary objects to be simultaneously assigned to one function code (each binary object mapping to a single bit of that 16-bit word). It is possible to map binary objects to multiple bits within the designated function code.

The effect of the "Bitmask" field when reading: When the present value property of a binary object is read by a BACnet client, the bitmask is used to determine the active/inactive state of the object by inspecting the value in the designated function code at the bit location(s) indicated in the bitmask. If all of the bit locations at the designated function code are set, then the object's state will be returned as "active". Else, the object's state will be returned as "inactive". This resultant state is reversed prior to being placed on the network if the object's "Polarity" is set to "Reverse".

Active Text

Specifies the description of the object's "active" state. Enter a string of up to 32 characters in length. This field is optional and may be left blank.

Inactive Text

Specifies the description of the object's "inactive" state. Enter a string of up to 32 characters in length. This field is optional and may be left blank.

Polarity

Indicates the relationship between the physical state of the object (as stored in the function code) and the logical state represented by the object's present value property. If the physical state is active high, select "Normal" from this dropdown menu. If the physical state is active low, select "Reverse" from this dropdown menu. For further detail, refer to the "Bitmask" behavioral description.

9.4.6.5 Binary Output Object Settings**Object Name**

The name of the BACnet object. Enter a string of between 1 and 32 characters in length. All object names must be unique within a node.

Instance

The BACnet object's instance number. Enter a value between 0...4194302 (0x0...0x3FFFFE).

Function Code

The inverter function code (refer to section 4) that the BACnet object's present value will access.

Bitmask

Specifies which bit(s) in the 16-bit value designated by the "Function Code" that the binary object will map to. This mechanism allows up to 16 binary objects to be simultaneously assigned to one function code (each binary object mapping to a single bit of that 16-bit word). It is possible to map binary objects to multiple bits within the designated function code.

The effect of the "Bitmask" field when writing: When the present value property of a binary object is set to "active" by a BACnet client, then the bit(s) in the designated function code indicated by the bitmask are set. Similarly, when the present value property of the object is set to "inactive", then the bit(s) in the designated function code indicated by the bitmask are cleared. This setting/clearing behavior is reversed if the object's "Polarity" is set to "Reverse".

The effect of the "Bitmask" field when reading: When the present value property of a binary object is read by a BACnet client, the bitmask is used to determine the active/inactive state of the object by inspecting the value in the designated function code at the bit location(s) indicated in the bitmask. If all of the bit locations at the designated function code are set, then the object's state will be returned as "active". Else, the object's state will be returned as "inactive". This resultant state is reversed prior to being placed on the network if the object's "Polarity" is set to "Reverse".

Active Text

Specifies the description of the object's "active" state. Enter a string of up to 32 characters in length. This field is optional and may be left blank.

Inactive Text

Specifies the description of the object's "inactive" state. Enter a string of up to 32 characters in length. This field is optional and may be left blank.

Polarity

Indicates the relationship between the physical state of the object (as stored in the function code) and the logical state represented by the object's present value property. If the physical state is active high, select "Normal" from this dropdown menu. If the physical state is active low, select "Reverse" from this dropdown menu. For further detail, refer to the "Bitmask" behavioral description.

Relinquish Default

Defines the default value to be used for an object's present value property when all entries in the object's priority array are NULL.

9.4.6.6 Binary Value Object Settings**Object Name**

The name of the BACnet object. Enter a string of between 1 and 32 characters in length. All object names must be unique within a node.

Instance

The BACnet object's instance number. Enter a value between 0...4194302 (0x0...0x3FFFFFFE).

Function Code

The inverter function code (refer to section 4) that the BACnet object's present value will access.

Bitmask

Specifies which bit(s) in the 16-bit value designated by the "Function Code" that the binary object will map to. This mechanism allows up to 16 binary objects to be simultaneously assigned to one function code (each binary object mapping to a single bit of that 16-bit word). It is possible to map binary objects to multiple bits within the designated function code.

The effect of the "Bitmask" field when writing: When the present value property of a binary object is set to "active" by a BACnet client, then the bit(s) in the designated function code indicated by the bitmask are set. Similarly, when the present value property of the object is set to "inactive", then the bit(s) in the designated function code indicated by the bitmask are cleared.

The effect of the "Bitmask" field when reading: When the present value property of a binary object is read by a BACnet client, the bitmask is used to determine the active/inactive state of the object by inspecting the value in the designated function code at the bit location(s) indicated in the bitmask. If all of the bit locations at the designated function code are set, then the object's state will be returned as "active". Else, the object's state will be returned as "inactive".

Active Text

Specifies the description of the object's "active" state. Enter a string of up to 32 characters in length. This field is optional and may be left blank.

Inactive Text

Specifies the description of the object's "inactive" state. Enter a string of up to 32 characters in length. This field is optional and may be left blank.

Relinquish Default

Defines the default value to be used for an object's present value property when all entries in the object's priority array are NULL.

9.4.6.7 Multi-state Input Object Settings

Object Name

The name of the BACnet object. Enter a string of between 1 and 32 characters in length. All object names must be unique within a node.

Instance

The BACnet object's instance number. Enter a value between 0...4194302 (0x0...0x3FFFFE).

Function Code

The inverter function code (refer to section 4) that the BACnet object's present value will access.

Data Type

Fixed at 16-Bit Unsigned.

Automatic Number of States

When enabled, this option automatically sets the Number of States setting to the largest value allowed by the selected Data Type.

Number of States

Defines the number of states that the object's present value property may have. The present value of a multi-state object is restricted to a range of 1...Number of States.

Offset by One

When enabled, this option automatically offsets the object's present value property by one compared to the object's value in the database. This option is useful when mapping a multi-state object to an enumeration that starts at zero, as zero is an invalid value for a multi-state object's present value property.

The effect of the "Offset by One" field when writing: When the present value property of a multi-state object is written by a BACnet client, the network value is decremented by one before being stored into the device's database.

The effect of the "Offset by One" field when reading: When the present value property of a multi-state object is read by a BACnet client, the raw data retrieved from the device's database is incremented by one to produce a network value.

9.4.6.8 Multi-state Output Object Settings

Object Name

The name of the BACnet object. Enter a string of between 1 and 32 characters in length. All object names must be unique within a node.

Instance

The BACnet object's instance number. Enter a value between 0...4194302 (0x0...0x3FFFFE).

Function Code

The inverter function code (refer to section 4) that the BACnet object's present value will access.

Data Type

Fixed at 16-Bit Unsigned.

Automatic Number of States

When enabled, this option automatically sets the Number of States setting to the largest value allowed by the selected Data Type.

Number of States

Defines the number of states that the object's present value property may have. The present value of a multi-state object is restricted to a range of 1...Number of States.

Offset by One

When enabled, this option automatically offsets the object's present value property by one compared to the object's value in the database. This option is useful when mapping a multi-state object to an

enumeration that starts at zero, as zero is an invalid value for a multi-state object's present value property.

The effect of the "Offset by One" field when writing: When the present value property of a multi-state object is written by a BACnet client, the network value is decremented by one before being stored into the device's database.

The effect of the "Offset by One" field when reading: When the present value property of a multi-state object is read by a BACnet client, the raw data retrieved from the device's database is incremented by one to produce a network value.

Relinquish Default

Defines the default value to be used for an object's present value property when all entries in the object's priority array are NULL.

9.4.6.9 Multi-state Value Object Settings

Object Name

The name of the BACnet object. Enter a string of between 1 and 32 characters in length. All object names must be unique within a node.

Instance

The BACnet object's instance number. Enter a value between 0...4194302 (0x0...0x3FFFE).

Function Code

The inverter function code (refer to section 4) that the BACnet object's present value will access.

Data Type

Fixed at 16-Bit Unsigned.

Automatic Number of States

When enabled, this option automatically sets the Number of States setting to the largest value allowed by the selected Data Type.

Number of States

Defines the number of states that the object's present value property may have. The present value of a multi-state object is restricted to a range of 1...Number of States.

Offset by One

When enabled, this option automatically offsets the object's present value property by one compared to the object's value in the database. This option is useful when mapping a multi-state object to an enumeration that starts at zero, as zero is an invalid value for a multi-state object's present value property.

The effect of the "Offset by One" field when writing: When the present value property of a multi-state object is written by a BACnet client, the network value is decremented by one before being stored into the device's database.

The effect of the "Offset by One" field when reading: When the present value property of a multi-state object is read by a BACnet client, the raw data retrieved from the device's database is incremented by one to produce a network value.

Relinquish Default

Defines the default value to be used for an object's present value property when all entries in the object's priority array are NULL.

9.5 PROFINET IO

9.5.1 Overview

The PROFINET IO device driver allows a controller to interact with the interface card via cyclic data exchange and acyclic read/write requests. The I/O data is entirely user-configurable, and is utilized when a standard I/O module is chosen during network configuration.

Some other notes of interest are:

- Allows simultaneous access to only 1 PROFINET controller.
- Supports conformance class B and real time (RT) communication.
- Supports the highest Netload Class III.
- Supports MRP (Media Redundancy Protocol) client.
- Supports DCP (Discovery Control Protocol).
- Supports alarms.
- Supports I&M.
- The lowest supported I/O Cycle Update Time (in STEP 7 or an equivalent hardware configuration tool) is 1ms.
- The GSDML file can be downloaded from [product web page](#) on the internet.
- Supports several user-configurable I/O modules with up to 32 input words and 32 output words.
- Supports the PROFIdrive profile version 4.1.
- No explicit module selection is required on the interface card: the module will be selected automatically according to the controller's configuration.
- If a timeout occurs on the RT connection, the driver can be configured to trigger a timeout event as described in section 6.7.1. The timeout value is dictated by the PROFINET controller and is at least three times the IO Cycle update time. The timeout value is also known as the "IO Cycle Watchdog" time.

9.5.2 Device Settings

In the studio's **Project** panel, navigate to **G500ETH...Ethernet...PROFINET IO**.

Device Name

The device name / station name must be unique across the entire PROFINET network, because it is used by controllers to uniquely identify PROFINET devices. This string must conform to the device name requirements contained in the PROFINET specification.

9.5.3 Connection Timeout Options

In the studio's **Project** panel, navigate to **G500ETH...Ethernet...PROFINET IO**. The following configuration options will determine the actions to be taken by the card if the PROFINET IO connection is abnormally terminated or lost.

Controller Stop State Behavior

PROFINET controllers (such as PLCs) include an "APDU Data Status" in all cyclic (I/O) command data packets sent to devices. This status includes a "run/stop" flag intended to signify when the controller is in the "run" state or the "stop" state. For example, a Siemens SIMATIC PLC will set the run/stop flag to stop when its processor dipswitch is placed in the "STOP" position.

The Invoke Timeout on Controller Stop State setting configures the behavior of the interface card when the controller sets the run/stop flag to stop.

- If the checkbox is not checked (default setting), then the driver will maintain the last consumed I/O data values received from the controller. For example, if the controller commanded the inverter to run prior to setting the run/stop flag to stop, then the inverter will continue to run.
- If the checkbox is checked, then the driver will perform the **Timeout Action**.

Timeout Action

Select an action from the drop down menu:

"None"..... No effect. The inverter will continue to operate with the last available settings.

"Apply Fail-safe Values" Apply the fail-safe values as described in section 6.7.1.

"Fault Drive" The behavior will depend on the timeout conditions set by the inverter (function code P16.57), which may result in an E-E71 fault. Refer to section 3.

9.5.4 Cyclic I/O Produced and Consumed Data Access Settings

In the studio's **Project** panel, add **G500ETH...Ethernet...PROFINET IO...Produced Data Word** and/or **Consumed Data Word**.

The Produced Data Word and Consumed Data Word objects are only applicable when using the I/O module "IN: 32 WORDS, OUT: 32 WORDS", which is typically the case. The Produced Data Word defines the structure of status data sent from the inverter to the controller. The Consumed Data Word objects will define the structure of the command data sent from the controller (for example, a Siemens PLC) to the inverter. These objects allow the creation of custom-built I/O data. Up to 32 "command" function code values can be sent to the inverter, and up to 32 "status" function code values can be sent back to the controller. Therefore, up to 32 Produced and 32 Consumed Data Word objects can be created. If a consumed word offset is not defined, that data will be ignored by the inverter. If a produce word offset is not defined, the value will default to 0. The size of the actual I/O produced and consumed data is determined by the PROFINET controller. The I/O data format is summarized in Table 50.

Description

This 32-character (max) field is strictly for user reference: it is not used at any time by the driver.

Produced Data Word Offset

The value from the associated inverter function code will populate this word offset of the produced data that is to be sent to the controller. It is recommended to start at word offset 0.

Consumed Data Word Offset

The consumed data received from the controller at this word offset will contain the value to be written to the associated inverter function code. It is recommended to start at word offset 0.

Function Code

The inverter function code (refer to section 4) associated with the word offset. For the Produced Data Word object, enter a "status" function code to be monitored. For the Consumed Data Word object, enter a "command" function code that can be written.

Data Type

Each data word is fixed to 16-Bit Unsigned (equivalent to two bytes.) The data word is transferred in little endian format.

Table 50: User-Configurable Module I/O Data Format

Consumed Data (PLC to Inverter)		Produced Data (Inverter to PLC)	
Word Offset	Function Code	Word Offset	Function Code
0	Any	0	Any
1	Any	1	Any
:	Any	:	Any
30	Any	30	Any
31	Any	31	Any

The default I/O configuration is described in Table 51.



Note Always use the studio to confirm the configuration before commissioning the device.

Table 51: Default User-Configurable Module I/O Data Format

Consumed Data (PLC to Inverter)		Produced Data (Inverter to PLC)	
Word Offset	Function Code	Word Offset	Function Code
0	P100.01 (Command word)	0	P100.51 (Status word)
1	P100.20 (Frequency command)	1	P100.70 (Status word 2)
2	P100.21 (PID reference command)	2	P100.71 (Running frequency)
3	P100.22 (PID feedback command)	3	P100.72 (Frequency reference)
4	P100.23 (Torque command)	4	P100.73 (Bus voltage)
5	P100.24 (Voltage command)	5	P100.74 (Output voltage)
6	P100.02 (Motor group selection)	6	P100.75 (Output current)
7	None	7	P100.76 (Output torque)
8	None	8	P100.77 (Output power)
9	None	9	P100.79 (PID reference)
10	None	10	P100.80 (PID feedback)
11	None	11	P100.78 (Fault code)
12	None	12	P100.53 (Operation mode)
13	None	13	P100.52 (Motor group feedback)
:	None	:	None
30	None	30	None
31	None	31	None

9.5.5 PROFIdrive Profile

For optimal interoperability, the interface card supports the PROFIdrive profile version 4.1. Use of the PROFIdrive profile is optional and is not recommended unless specifically required in the PROFINET system specification. No explicit configuration of the interface card is necessary in the studio when using the PROFIdrive profile. The controller **must** support the PROFIdrive profile and **must** be configured to use the "Standard Telegram 1" module on the interface card. If the controller does not support the PROFIdrive profile, use the configurable I/O "IN: 32 WORDS, OUT: 32 WORDS" module. The PROFIdrive profile is only partially described in this manual due to its complexity. The complete PROFIdrive profile specifications can be obtained from <http://www.profibus.com/>.

Some other notes of interest include:

- Implements Application Class 1 (standard drive)
- Supports only Standard Telegram 1 (ST1, PZD-2/2) on slot 1 (similar to Profibus PPO type 3)
- Supports only Speed Control Mode

9.5.5.1 PROFIdrive Standard Telegram 1

The standard telegram 1 mapping is described in Table 52.

Table 52: Standard Telegram 1

IO Data Word Offset	Setpoint (PLC to Inverter)		Actual Value (Inverter to PLC)	
	Significance	Description	Significance	Description
0	STW1	Control word 1	ZSW1	Status word 1

IO Data Word Offset	Setpoint (PLC to Inverter)		Actual Value (Inverter to PLC)	
	Significance	Description	Significance	Description
1	NSOLL_A	Reference speed setpoint	NIST_A	Speed actual

9.5.5.2 PROFIdrive Control and Status Words

The control word, STW1, is the principal means for controlling the drive. It is sent by the controller (PLC) to the device (inverter). The bitmapping for the control word is described in Table 53. The status word, ZSW1, returns status information from the inverter to the controller. The bitmapping for the status word is described in Table 54.

Table 53: STW1 Control Word Mapping

Bit	Value	Significance	Description
0	1	ON	Run command ON
	0	OFF	Run command OFF
1	1	ON2	No coast stop
	0	OFF2	Coast to a stop
2	1	ON3	No quick stop
	0	OFF3	Quick stop
3	1	Enable Operation	Enable inverter operation
	0	Disable Operation	Disable inverter operation
4	1	Enable Ramp Generator	Enable the ramp frequency generator (RFG)
	0	Disable Ramp Generator	Hold the output frequency to 0 Hz
5	1	Unfreeze Ramp Generator	Unfreeze the RFG
	0	Freeze Ramp Generator	Freeze the RFG with the current output frequency
6	1	Enable Setpoint	Enable command
	0	Disable Setpoint	Disable command
7	1	Fault Acknowledge	Reset the alarm on a positive edge (0→1 transition)
	0	No significance	Do not reset the alarm
8 - 9	Not used	---	---
10	1	Control By PLC	Enable remote control. The IO process data is valid.
	0	No Control By PLC	Disable remote control. The IO process data is not valid.
11	1	Reverse Direction	Run in the reverse direction
	0	Forward Direction	Run in the forward direction
12 - 15	Not used	---	---

Table 54: ZSW1 Status Word Mapping

Bit	Value	Significance	Description
0	1	Ready To Switch ON	Ready to run command ON
	0	Not Ready To Switch ON	Not ready to run command ON
1	1	Ready to Operate	Ready to run
	0	Not Ready To Operate	Not ready to run
2	1	Operation Enabled	Running
	0	Operation Disabled	Running disabled
3	1	Fault Present	Inverter tripped as indicated by ALM. Refer to function code P100.51 bit 2.
	0	No Fault	No trip present as indicated by ALM. Refer to function code P100.51 bit 2.
4	1	Coast Stop Not Activated	Follows STW1 bit 1, ON2 active
	0	Coast Stop Activated	Follows STW1 bit 1, OFF2 active
5	1	Quick Stop Not Activated	Follows STW1 bit 2, ON3 active
	0	Quick Stop Activated	Follows STW1 bit 2, OFF3 active
6	1	Switch ON Inhibited	Not ready to run command ON
	0	Switch ON Not Inhibited	Ready to run command ON
7	Not Used	---	---
8	1	Speed Within Tolerance	Actual value equals the reference value and is within the tolerance as indicated by frequency arrival. Refer to function code P08.36.
	0	Speed Out Of Tolerance	Actual value differs from the reference value or is outside of the tolerance as indicated by frequency arrival. Refer to function code P08.36.
9	1	Control Requested	Control by PLC is enabled.
	0	No Control Requested	Control is not possible by the controller.
10	1	Frequency Reached Or Exceeded	The actual value \geq max reference value as indicated by FDT1. Refer to function code P08.32.
	0	Frequency Not Reached	The actual value $<$ max reference value as indicated by FDT1. Refer to function code P08.32.
11 - 15	Not used	---	---

9.5.5.3 PROFIdrive Reference Speed Setpoint and Actual Speed

The speed setpoint value, NSOLL_A, is the commanded speed reference (normalized) sent from the controller to the inverter. Similarly, the speed actual value, NIST_A, is the actual operating speed (normalized) of the inverter sent back to the controller. As the inverter natively operates in units of Hz, the following conversion equations are applied within the interface card:

NSOLL_A: The inverter reference speed setpoint is a normalized value. The interface card applies the conversion indicated in Equation 8 in order to determine the appropriate frequency command to be written to function code P100.20 (frequency command).

$$\text{Hz} = (\text{NSOLL_A} \times \text{Max Frequency}) / 0x4000$$

Equation 8

NIST_A: The inverter operating actual speed is a normalized value that is calculated from inverter function code P100.52 (output frequency). The interface card applies the conversion indicated in Equation 9 in order to determine the appropriate operating speed actual (normalized).

$$\text{NIST_A} = (\text{Hz} \times 0x4000) / \text{Max Frequency}$$

Equation 9

The "Max Frequency" term which appears in Equation 8 and Equation 9 is obtained from the setting of inverter function code P00.03 (maximum output frequency).

A normalized value of 0x4000 corresponds to 100% of the maximum frequency. A positive normalized value indicates forward rotation and a negative normalized value indicates reverse rotation.



The value of P00.03 is read by the interface card only at boot-up. 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.

9.5.5.4 PROFIdrive State Diagram

The state diagram is displayed in Figure 66.

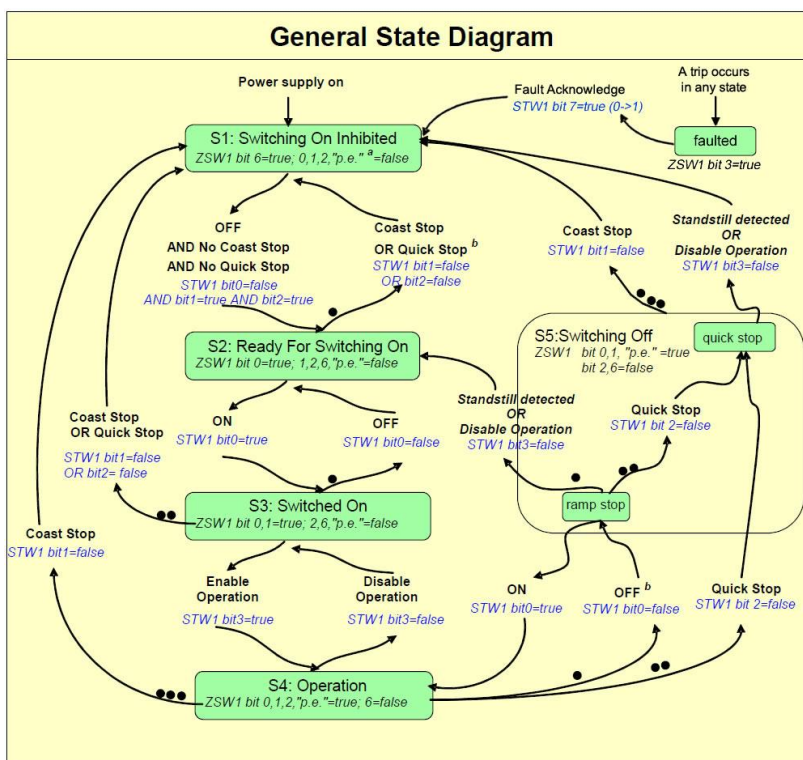


Figure 66: PROFIdrive State Diagram

9.5.5.5 PROFIdrive-Specific Parameters

The PROFIdrive-specific parameters are shown in Table 55. The parameters are read-only.

Table 55: PROFIdrive-Specific Parameters

PNU	Index	Description
711	None	NSOLL_A – Speed setpoint A
712	None	NIST_A – Speed actual A
833	None	STW1 – Control word 1
834	None	ZSW1 – Status word 1
922	None	Telegram selection = 1 (Standard telegram 1)
923	1,2,5,6	List of all parameters for signals
944	None	Fault message counter
947	0..3	Fault number (M16...M19)
964	0..6	Drive Unit identification
965	None	Profile identification number = Profile 3, Version 4.1
975	0..7	DO identification
980	0..5	Number list of defined parameter
1401	None	DO IO Data reference parameter

9.5.6 Acyclic Data Access

Any inverter function code can be accessed via PROFINET acyclic services. To accomplish this, set the API to 0, Slot to 1 and SubSlot to 1. The record number/index value is equivalent to the desired function code register address described in section 4.1. The length is specified according to the number of bytes to access. Since each function code corresponds to 2 bytes of data, the length must be an even number.

9.5.7 TIA Portal (STEP 7) Hardware Configuration Example

The following example will use TIA Portal V13 (STEP 7) to demonstrate the basic hardware configuration procedure to configure a PROFINET device. The procedure, in general, will apply to similar configuration software. The example will not cover all features of TIA Portal. Any questions regarding TIA Portal (or similar configuration software) must be directed to the vendor of the software.

This example assumes that there is already an existing TIA Portal project with the desired PLC.

9.5.7.1 Register the GSDML File

Open the TIA Portal project. Navigate to **Options...Manage general station description files (GSD)** as shown in Figure 67.

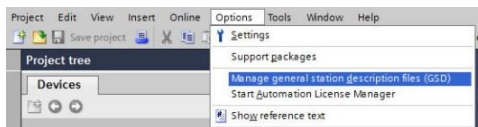


Figure 67: Install GSD File Menu Option

Locate and select the GSDML file and click the **Install** button. Confirm that the installation was completed successfully as shown in Figure 68 and click the **Close** button. It is recommended to use the latest GSDML, which is available via the [product web page](#) on the internet.

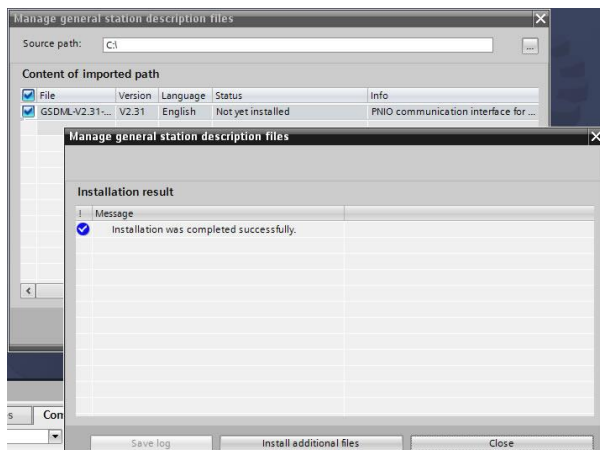


Figure 68: Successfully Installed GSDML File

This will update the **Hardware catalog**. Locate the device in the **Hardware catalog**. In the **Project tree**, double-click on **Device & networks**. Select the **Network view** tab and locate the device in the **Hardware catalog** as shown in Figure 69.

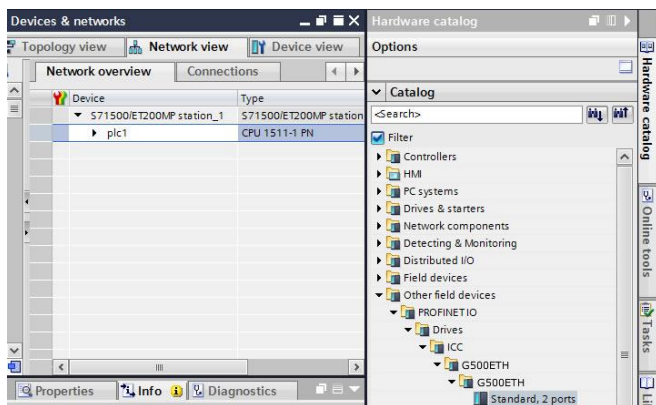


Figure 69: Updated Hardware Catalog

9.5.7.2 Add the Device to the Configuration

Select the device in the **Hardware catalog** and drag the device into the PROFINET IO system configuration as shown in Figure 70.

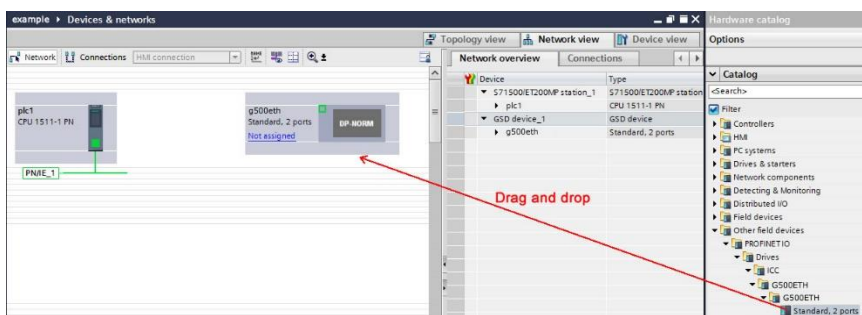


Figure 70: Add Device to Configuration

9.5.7.3 Select the IO Controller

On the device, click "Not assigned" and select the appropriate PLC PROFINET interface as shown in Figure 71. This will assign the device to the PROFINET IO system as shown in Figure 72.

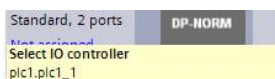


Figure 71: Select IO Controller

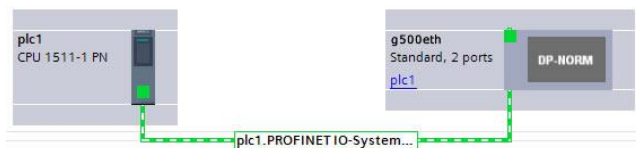


Figure 72: PROFINET IO System

9.5.7.4 Assign IO Module

Click on the device and then click on the **Device view** tab. In the **Hardware catalog**, expand **Module** and add a module "IN: XX WORDS, OUT: YY WORDS" into **Slot 1**. The module will determine the input and output sizes. In this example, the module "IN: 03 WORDS, OUT: 03 WORDS" is selected. Select a module with the appropriate input and output sizes for your specific application. Refer to Figure 73.

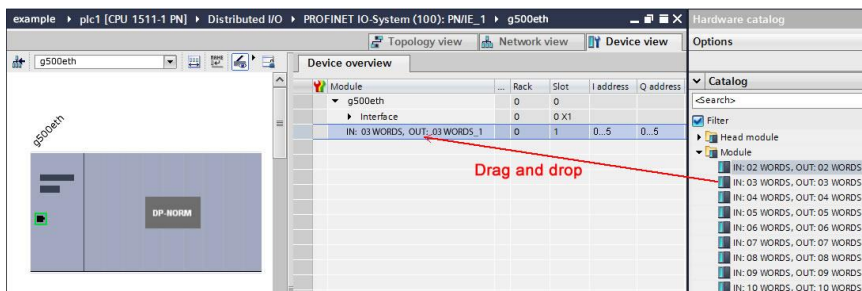


Figure 73: Add IO Module

9.5.7.5 Configure the Device Properties

Select the device and navigate to the **Properties** tab. Select the **PROFINET interface [X1]** node. Assign a unique and compatible **IP address** for this device as shown in Figure 74.

The screenshot shows the 'PROFINET interface [X1]' configuration window. The 'General' tab is selected. Under 'General', the 'Name' field is 'Interface' and the 'Comment' field is empty. Under 'Ethernet addresses', the 'Interface networked with' dropdown is set to 'PNIE_1'. Under 'IP protocol', the 'Use IP protocol' checkbox is checked. The 'Set IP address in the project' radio button is selected, with the IP address field showing '192.168.17.102' and the subnet mask field showing '255.255.255.0'. The 'Router address' field is set to '0.0.0.0'.

Figure 74: Assign Unique Compatible IP Address

Assign a unique **PROFINET** device name as shown in Figure 75.

The screenshot shows the 'PROFINET' configuration section. The 'Generate PROFINET device name automatically' checkbox is unchecked. The 'PROFINET device name' field is 'interfacelcard', the 'Converted name' field is 'interfacelcard', and the 'Device number' dropdown is set to '1'.

Figure 75: Assign Unique Device Name

Set the I/O cycle **Update time** and **Watchdog time** as shown in Figure 76.

The screenshot shows the 'Update time' and 'Watchdog time' configuration sections. Under 'Update time', the 'Automatic' radio button is unchecked, and the 'Can be set' radio button is selected with a value of '8.000' ms. Under 'Watchdog time', the 'Accepted update cycles without IO data' dropdown is set to '3' and the 'Watchdog time' field is '24.000' ms.

Figure 76: Set I/O Cycle Update Time

9.5.7.6 Online Device Discovery and Configuration

In the **Project tree**, expand **plc1...Distributed I/O...PROFINET IO-System (100):PN/IE_1**. Expand the device and double-click **Online & diagnostics**. In the next panel, expand **Functions** and select the **Assign IP address** node. Click the **Accessible devices** button. Select the appropriate **PG/PC interface** and click the **Start search** button to discover and display the PROFINET devices on the network as shown in Figure 77. Select the device and click the **Apply** button.

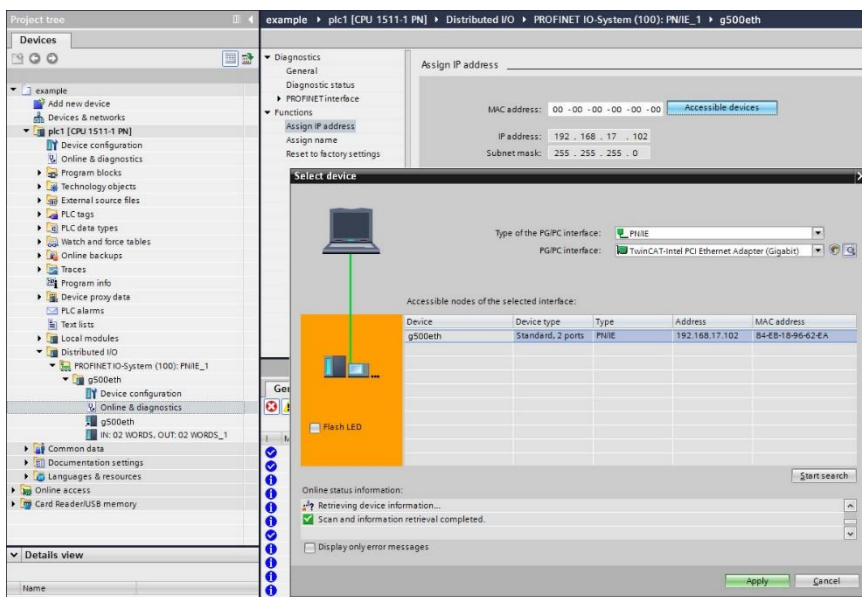


Figure 77: Discover PROFINET Devices on the Network

If the **IP address** does not match the values set in the configuration, click the **Assign IP address** button as shown in Figure 78.

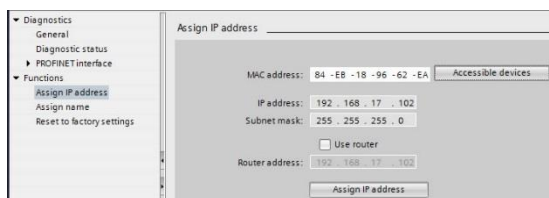


Figure 78: Assign IP Address

Navigate to **Functions...Assign name**. If the **PROFINET device name** does not match, select the device and click the **Assign name** button as shown in Figure 79.

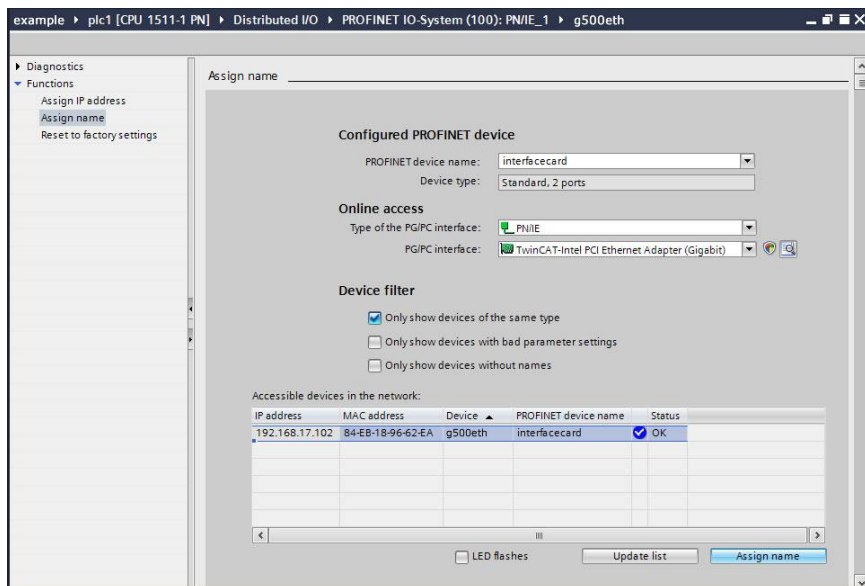


Figure 79: Assign Name

9.5.7.7 Save the Configuration

The hardware configuration is now complete. Save and perform any necessary compilation of the configuration. Download the application and configuration to the PLC. The PLC application program can then be started. Please consult with the vendor of your PROFINET PLC software for additional programming and configuration details.

9.5.8 GE Proficy Configuration Example

The following example will use GE Proficy Machine Edition SIM11 to demonstrate the basic procedure for configuring a PROFINET device. The example will not cover all features of Proficy. Any questions regarding Proficy (or similar configuration software) must be directed at the vendor of the software.

This example assumes that there is already an existing Proficy project with the desired PLC.

9.5.8.1 Register the GSDML File

Open the Proficy project. In the **Navigator** panel, right-click **Profinet Controller** and select **Add IO-Device...** as shown in Figure 80.

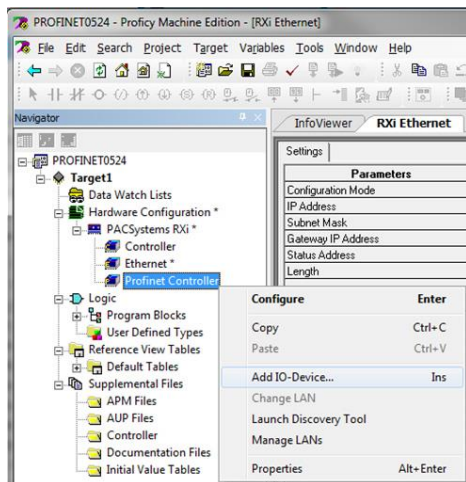


Figure 80: Add IO-Device

Click the **Have GSDML...** button as shown in Figure 81.

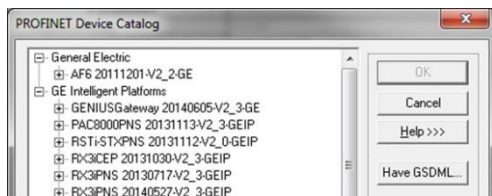


Figure 81: Have GSDML

Locate and select the GSDML file. Click the **Open** button to register the GSDML as shown in Figure 82. It is recommended to use the latest GSDML, which is available via the [product web page](#) on the internet.

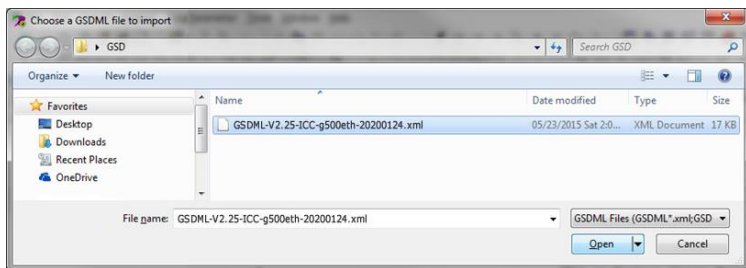


Figure 82: Register GSDML

This will update the **PROFINET Device Catalog**. Locate the device in the **PROFINET Device Catalog** as shown in Figure 83.



Figure 83: Updated PROFINET Device Catalog

9.5.8.2 Add the Device to the Configuration

Select the device in the **PROFINET Device Catalog** and click the **OK** button as shown in Figure 83. The device is added under the **Profinet Controller** node as shown in Figure 84.

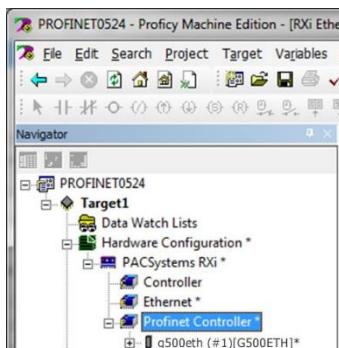


Figure 84: Added Device to Configuration

9.5.8.3 Assign IO Module

In the **Navigator** panel, right-click on the device and select **Change Module List...** as shown in Figure 85.

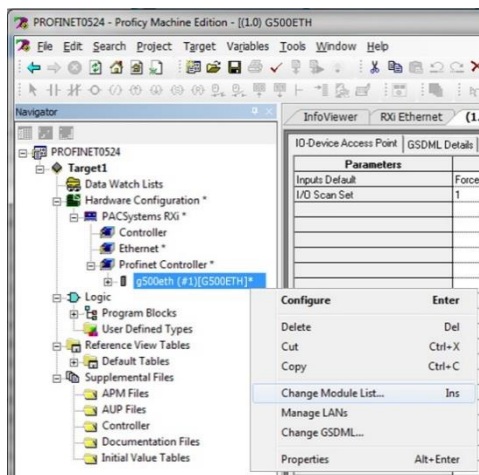


Figure 85: Change Module List

Select a module and drag the module into the available slot. The available slots and modules will vary depending on the specific device. Select a module appropriate for your application. Click the **OK** button as shown in Figure 86.

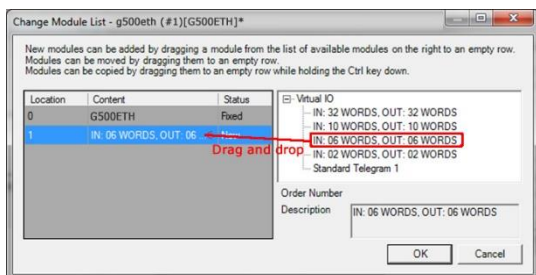


Figure 86: Add IO Module

The module will be reflected in the **Navigator** panel, under the device as shown in Figure 87.

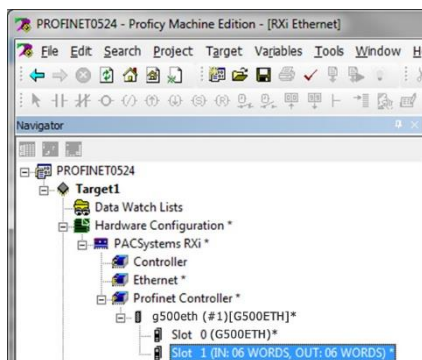


Figure 87: Added IO Module

9.5.8.4 Configure the Device Properties

In the **Navigator** panel, right-click on the device and select **Properties** as shown in Figure 88.

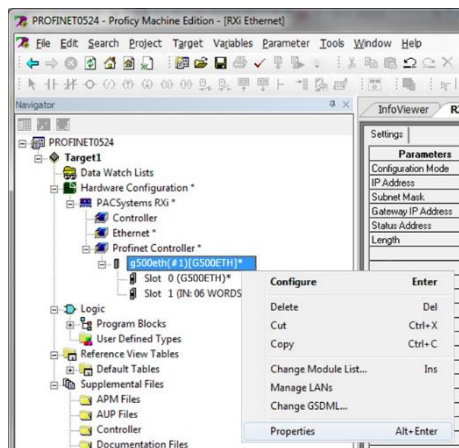


Figure 88: Select Device Properties

Set the properties to match the configuration on the device. The properties must be appropriate for the application and the PROFINET network.

Set the **Update Rate (ms)**. For this example, the **Update Rate (ms)** is set to "8" ms.

Assign a unique **Device Name**. For this example, the **Device Name** is set to "interfacecard".

Assign a unique and compatible **IP Address**. For this example the **IP Address** is set to "192.168.17.102".

The resulting properties are shown in Figure 89.

9.5.8.5 Save the Configuration

The device configuration is now complete. Save and perform any necessary compilation of the configuration. Download the application and configuration to the PLC. The PLC application program can then be started. Please consult with the vendor of your PROFINET PLC software for additional programming and configuration details.

Inspector	
IO-Device	
Device Number	1
Update Rate (ms)	8
Reference Variable	<None>
Network Identification	
IO LAN	LAN01
Device Name	interfacecard
Device Description	
IP Address	192.168.17.102

Figure 89: Set Device Properties

9.6 MELSEC/SLMP Server

9.6.1 Overview

The Mitsubishi MELSEC communication protocol (MC protocol) is also known as SLMP (Seamless Messaging Protocol). It is an application-level protocol implemented on top of the Ethernet TCP/IP and UDP/IP layers that is typically used to read and write data from/to devices supporting compatible 4E (MT), 3E (ST) and 1E frame types. As a server device, the interface card is waiting for a client device to initiate a request. The interface card will then respond with the appropriate data.

For more information regarding the MELSEC protocol, refer to the Mitsubishi MELSEC Communication Protocol reference manual. The SLMP specification is available from CLPA (CC-Link Partner Association).

Other notes of interest are:

- Supports both TCP and UDP simultaneously.
- Supports up to 8 simultaneous TCP connections.
- The TCP port is user-configurable.
- The UDP port is user-configurable and defines both the local source port as well as the remote destination port.
- The max frame size is 1460 bytes.
- The 3E/4E Device Memory Random Write command will attempt to write to all requested device points even if an error is encountered. Ensure that all requested device points are valid before using Device Memory Random Write.
- The driver can be configured to detect a timeout (communication loss) and perform a timeout action.

9.6.2 Read/Write Commands

If applicable, the network number is 0, the PC number (node number) is 0xFF, the module I/O number (process number) is 0x3FF, and the module station number is 0.

Table 56 lists the supported commands. Device read/write commands only support 16-bit word access.

Table 56: MELSEC / SLMP Server Commands

Frame Type	Command Name	Command Code	Subcommand Code	Max Points
3E/4E	CPU Model Read	0x0101	0x0000	-
3E/4E	Node Search	0x0E30	0x0000	-
3E/4E	IP Address Set	0x0E31	0x0000	-
3E/4E	Device Info Compare	0x0E32	0x0000	-
3E/4E	Status Read	0x0E44	0x0000	-
3E/4E	Communication Setting Get	0x0E45	0x0000	-
3E/4E	Device Memory Batch Read	0x0401	0x0000	724 (3E) / 722 (4E)
3E/4E	Device Memory Batch Write	0x1401	0x0000	719 (3E) / 717 (4E)
3E/4E	Device Memory Random Read	0x0403	0x0000	360 (3E) / 359 (4E)
3E/4E	Device Memory Random Write	0x1402	0x0000	360 (3E) / 358 (4E)
1E	Device Memory Batch Read	0x01	-	256
1E	Device Memory Batch Write	0x03	-	256

Table 57 lists the supported internal device memory types. There is no functional difference between the device types. All device types are equal and simply provide access to the register mapping.

Table 57: MELSEC / SLMP Server Device Types

Device Type	3E/4E Device Code	1E Device Code	Data Type
Data Register	0xA8	0x4420	16-Bit Word
Link Register	0xB4	0x5720	16-Bit Word
Index Register	0xCC	-	16-Bit Word
File Register	0xAF, 0xB0	0x5220, 0x5A52	16-Bit Word

The device point is the register address (refer to section 4.1).

9.6.3 Server Settings

In the studio's Project panel, navigate to **G500ETH...Ethernet...MELSEC/SLMP Server**.

TCP Port

Defines the local TCP port (1025...65534) on which the driver will listen for connections from the client. Ensure that this port assignment is unique, and does not conflict with ports utilized by other drivers.

UDP Port

Defines the local UDP port (1025...65534) on which the driver will listen for requests from the client. Ensure that this port assignment is unique, and does not conflict with ports utilized by other drivers.

9.6.4 Connection Timeout Options

In the studio's Project panel, navigate to **G500ETH...Ethernet...MELSEC/SLMP Server**. The following configuration options will determine the actions to be taken if the connection is abnormally terminated or lost. 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. Note that a certain degree of caution must be exercised when using the timeout feature to avoid "nuisance" timeouts from occurring.

Enable Supervisory Timer

This timer provides the ability for the driver to monitor timeout occurrences on the overall receive activity for all connections.

- The timer will start after receiving the first request. Once the timer is started, it cannot be disabled.
- If the driver experiences no receive activity for more than the **Timeout** time setting, then the driver assumes that the client or network has experienced some sort of unexpected problem, and will perform the **Timeout Action**.

Enable Connection Timer

This timer provides the ability for the driver to monitor timeout occurrences and errors within the scope of each client connection.

- If a particular open socket experiences no activity for more than the **Timeout** time setting, then the driver assumes that the client or network has experienced some sort of unexpected problem, and will close that socket and perform the **Timeout Action**.
- If a socket error occurs (regardless of whether the error was due to a communication lapse or abnormal socket error), the driver will perform the **Timeout Action**. Specifically, do not perform inadvisable behavior such as sending a request from the client device, and then closing the socket prior to successfully receiving the server's response. The reason for this is because the server will experience an error when attempting to respond via the now-closed socket. Always be sure to manage socket life cycles "gracefully", and do not abandon outstanding requests.

Timeout

Defines the maximum number of milliseconds for a break in network communications before a timeout event will be triggered.

Timeout Action

Select an action from the drop down menu:

-
- "None" No effect. The inverter will continue to operate with the last available settings.
- "Apply Fail-safe Values" Apply the fail-safe values as described in section 6.7.1.
- "Fault Drive" The behavior will depend on the timeout conditions set by the inverter (function code P16.57), which may result in an E-E71 fault. Refer to section 3.

9.7 CC-Link IE Field Basic Server

9.7.1 Overview

CC-Link IE Field Basic (CCIEF Basic) is an application-level protocol implemented on top of the Ethernet UDP/IP layer. As a server device, the interface card is waiting for a client device to initiate cyclic communication containing the cyclic RWw (command) data. The interface card will then respond with cyclic RWr (status) data.

For more information regarding the CC-Link IE Field Basic protocol and specification, contact the CLPA (CC-Link Partner Association).

Other notes of interest are:

- The UDP port is 61450.
- Occupies 1 station.
- Supports up to 32 cyclic RWw (command) words and 32 cyclic RWr (status) words.
- The driver can be configured to detect a timeout (communication loss) and perform a timeout action.

9.7.2 Server Settings

In the studio's Project panel, navigate to **G500ETH...Ethernet...CC-Link IE Field Basic Server**.

Timeout Action

Select an action from the drop down menu:

- "None" No effect. The inverter will continue to operate with the last available settings.
- "Apply Fail-safe Values" Apply the fail-safe values as described in section 6.7.1.
- "Fault Drive" The behavior will depend on the timeout conditions set by the inverter (function code P16.57), which may result in an E-E71 fault. Refer to section 3.

9.7.3 Produced and Consumed Data Settings

In the studio's **Project** panel, add **G500ETH...Ethernet...CC-Link IE Field Basic Server ...Produced I/O Data...Produced Data Word** and/or **Consumed I/O Data...Consumed Data Word**.

The Produced Data Word objects defines the structure of cyclic RWr status words sent from the inverter back to the controller. The Consumed Data Word objects will define the structure of the cyclic RWw command words sent from the CC-Link IE Field Basic controller to the inverter. These objects allow the creation of custom-built I/O data. Up to 32 "command" function code values can be sent to the inverter, and up to 32 "status" function code values can be sent back to the controller. Therefore, up to 32 Produced and 32 Consumed Data Word objects can be created. If a consumed word offset is not defined, that data will be ignored by the inverter. If a produce word offset is not defined, the value will default to 0. The I/O data format is summarized in Table 58.

Description

This 32-character (max) field is strictly for user reference: it is not used at any time by the driver.

Produced Data Word Offset

The value from the associated inverter function code will populate this word offset of the produced data that is to be sent to the client. It is recommend to start at word offset 0.

Consumed Data Word Offset

The consumed data received from the client at this word offset will contain the value to be written to the associated inverter function code. It is recommend to start at word offset 0.

Function Code

Function code (refer to section 4) associated with the word offset. For the Produced Data Word object, enter a "status" function code to be monitored. For the Consumed Data Word object, enter a "command" function code that can be written.

Data Type

Each data word is fixed to 16-Bit Unsigned. This is equivalent to two bytes.

Table 58: CC-Link IE Field Basic User-Configurable I/O Data Format

Consumed Data (PLC to Inverter)		Produced Data (Inverter to PLC)	
Word Offset	Function Code	Word Offset	Function Code
0	Any	0	Any
1	Any	1	Any
:	Any	:	Any
30	Any	30	Any
31	Any	31	Any

The default I/O configuration is described in Table 59. Always use the studio to confirm the configuration before commissioning the device.

Table 59: CC-Link IE Field Basic Default User-Configurable I/O Data Format

Consumed Data (PLC to Inverter)		Produced Data (Inverter to PLC)	
Word Offset	Function Code	Word Offset	Function Code
0	P100.01 (Command word)	0	P100.51 (Status word)
1	P100.20 (Frequency command)	1	P100.70 (Status word 2)
2	P100.21 (PID reference command)	2	P100.71 (Running frequency)
3	P100.22 (PID feedback command)	3	P100.72 (Frequency reference)
4	P100.23 (Torque command)	4	P100.73 (Bus voltage)
5	P100.24 (Voltage command)	5	P100.74 (Output voltage)
6	P100.02 (Motor group selection)	6	P100.75 (Output current)
7	None	7	P100.76 (Output torque)
8	None	8	P100.77 (Output power)
9	None	9	P100.79 (PID reference)
10	None	10	P100.80 (PID feedback)
11	None	11	P100.78 (Fault code)
12	None	12	P100.53 (Operation mode)
13	None	13	P100.52 (Motor group feedback)
:	None	:	None
30	None	30	None
31	None	31	None

9.8 MELSEC Client

9.8.1 Overview

The Mitsubishi MELSEC communication protocol (MC protocol) is an application-level protocol implemented on top of the Ethernet TCP/IP and UDP/IP layers that is typically used to read and write data from/to devices supporting compatible 4E (MT), 3E (ST) and 1E frame types. As a client device, the interface card will initiate a read/write request to a server device. The server device will then respond with the appropriate data.

For additional MELSEC protocol information, refer to the Mitsubishi MELSEC Communication Protocol reference manual.

Other notes of interest are:

- Supports both TCP and UDP transport layers.
- Target up to 8 remote server devices.
- Supports option to auto-detect the frame type. The frame type is prioritized in the following order: 3E, 4E, and 1E. Otherwise the frame type can be manually selected (refer to the server equipment user's manual).
- User-specified device codes allow the ability to target any device type, even if the device is not explicitly supported by the driver.
- The driver can be configured to perform a timeout action.

9.8.2 Read/Write Commands

Table 60 lists the supported commands. Device read/write commands only support 16-bit word access.

Table 60: MELSEC Client Commands

Frame Type	Command Name	Command Code	Subcommand Code	Max Points
3E/4E	Device Memory Batch Read	0x0401	0x0000	719
3E/4E	Device Memory Batch Write	0x1401	0x0000	719
1E	Device Memory Batch Read	0x01	-	256
1E	Device Memory Batch Write	0x03	-	256

9.8.3 Connection Timeout Options

In the studio's Project panel, navigate to **G500ETH...Ethernet...MELSEC Client**. The following configuration options will determine the actions to be taken if the connection is abnormally terminated or lost. 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. Note that a certain degree of caution must be exercised when using the timeout feature to avoid "nuisance" timeouts from occurring.

Timeout Action

Select an action from the drop down menu:

"None"..... No effect. The inverter will continue to operate with the last available settings.

"Apply Fail-safe Values" Apply the fail-safe values as described in section 6.7.1.

"Fault Drive" The behavior will depend on the timeout conditions set by the inverter (function code P16.57), which may result in an E-E71 fault. Refer to section 3.

9.8.4 Remote Device Settings

In the studio's Project panel, navigate to **G500ETH...Ethernet...MELSEC Client...Remote Device**.

Name

This 32-character (max) field is strictly for user reference: it is not used at any time by the driver.

IP Address

Defines the destination IP address. This is the IP address of the targeted remote server device.

Transport Type

Select TCP or UDP.

TCP Port

This field is only enabled only if the **Transport Type** is set to TCP. Defines the destination TCP port (1...65535). This is the "listening" port on the server.

UDP Port

This field is only enabled only if the **Transport Type** is set to UDP. Defines the destination UDP port (1...65535). This is the "listening" port on the server. This also defines the local source port. Ensure that this port assignment is unique, and does not conflict with ports utilized by other drivers.

Response Timeout

Defines the time in milliseconds to wait for a response from the server, after sending a request. If a response is not received within this time, the **Timeout Action** is performed.

Request Delay

Defines the time in milliseconds to wait before sending the next request.

Frame Type

This field designates the frame type to use when communicating with the server. If the required Frame Type is unknown, Auto-Detect can be selected to automatically determine the frame type when the driver establishes initial contact with the server. In this case, once the frame type has been successfully detected, the driver will then remember this type and use it as the initial preference for subsequent connection attempts. The frame type is prioritized in the following order: 3E, 4E and 1E. Note that the auto-detection procedure may cause a communication error to be indicated on the server during the initial connection attempt, but this error can be cleared by power-cycling the server. To determine the required Frame Type for your equipment, please consult the server device documentation.

9.8.5 Command and Monitor Data Object Settings

In the studio's Project panel, add **G500ETH...Ethernet...MELSEC Client...Command Data** and/or **Monitor Data**. The **Command Data** object will execute the Device Batch Read command to read command data from the targeted remote server device. The **Monitor Data** object will execute the Device Batch Write command to write status data to the targeted remote server device.

Description

This 32-character (max) field is strictly for user reference: it is not used at any time by the driver.

Device Code

For user convenience, a selection of well-known device codes is provided. However, this in no way limits the variety of device codes that can be targeted. Any device code that is supported by the server device can be specified by choosing "Other Device Code" from the drop-down box, and then entering the code in the **Code Value** field.

Code Value

This field is enabled only if **Device Code** is set to "Other Device Code". If the desired device code is not available in the **Device Code** drop-down, then choose "Other Device Code" and enter the appropriate code in this field.

Starting Point

Specifies the initial point in a device's range of points. Enter a value from 0...16777215 (0x0...0xFFFFFFF). If the connection uses 1E frames and the **Device Code** or **Code Value** targets a bit device, then the starting point should be a multiple of 16.

Number of Words

Specifies the number of words (1...719) from the **Starting Point** that are to be accessed. If the connection uses 1E frames, then the maximum value should be limited to 256 words: if a larger value is

entered, the value will be internally limited to 256. The server may also impose additional device-specific limitations (please consult the server device's documentation).

Function Code

The function code (refer to section 4) associated with the word offset. For the Monitor Data object, enter a "status" function code to be monitored. For the Command Data object, enter a "command" function code that can be written.

Network Number

Specifies the network number of the target station. Fixed to 0.

PC/Station Number

Specifies the PC/station number of the target station. Fixed to 0xFF.

Module IO Number

Specifies the module I/O number (also known as the processor number) of the target station. Fixed to 0x3FF.

9.8.6 Diagnostics Objects

Each command and monitor data object can optionally include a diagnostics object for debugging and diagnostics. Refer to 6.14.

Diagnostic Index

Enter the diagnostic index at which to store the diagnostics information.

9.9 SLMP Client

9.9.1 Overview

SLMP (Seamless Messaging Protocol) is an application-level protocol implemented on top of the Ethernet TCP/IP and UDP/IP layers that is typically used to read and write data from/to devices supporting compatible 4E (MT) and 3E (ST) frame types. As a client device, the interface card will initiate a read/write request to a server device. The server device will then respond with the appropriate data.

For more information regarding the SLMP, the specification is available from CLPA (CC-Link Partner Association).

Other notes of interest are:

- Supports both TCP and UDP transport layers.
- Target up to 8 remote server devices.
- Supports option to auto-detect the frame type. The frame type is prioritized in the following order: 3E and 4E. Otherwise the frame type can be manually selected (refer to the server equipment user's manual).
- User-specified device codes allow the ability to target any device type, even if the device is not explicitly supported by the driver.
- The driver can be configured to perform a timeout action.

9.9.2 Read/Write Commands

Table 61 lists the supported commands. Device read/write commands only support 16-bit word access.

Table 61: SLMP Client Commands

Frame Type	Command Name	Command Code	Subcommand Code	Max Points
3E/4E	Device Memory Batch Read	0x0401	0x0000	719
3E/4E	Device Memory Batch Write	0x1401	0x0000	719

9.9.3 Connection Timeout Options

In the studio's Project panel, navigate to **G500ETH...Ethernet...SLMP Client**. The following configuration options will determine the actions to be taken if the connection is abnormally terminated or lost. 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. Note that a certain degree of caution must be exercised when using the timeout feature to avoid "nuisance" timeouts from occurring.

Timeout Action

Select an action from the drop down menu:

"None" No effect. The inverter will continue to operate with the last available settings.

"Apply Fail-safe Values" Apply the fail-safe values as described in section 6.7.1.

"Fault Drive" The behavior will depend on the timeout conditions set by the inverter (function code P16.57), which may result in an E-E71 fault. Refer to section 3.

9.9.4 Remote Device Settings

In the studio's Project panel, navigate to **G500ETH...Ethernet...SLMP Client...Remote Device**.

Name

This 32-character (max) field is strictly for user reference: it is not used at any time by the driver.

IP Address

Defines the destination IP address. This is the IP address of the targeted remote server device.

Transport Type

Select TCP or UDP.

TCP Port

This field is only enabled only if the **Transport Type** is set to TCP. Defines the destination TCP port (1...65535). This is the "listening" port on the server.

UDP Port

This field is only enabled only if the **Transport Type** is set to UDP. Defines the destination UDP port (1...65535). This is the "listening" port on the server. This also defines the local source port. Ensure that this port assignment is unique, and does not conflict with ports utilized by other drivers.

Response Timeout

Defines the time in milliseconds to wait for a response from the server, after sending a request. If a response is not received within this time, the **Timeout Action** is performed.

Request Delay

Defines the time in milliseconds to wait before sending the next request.

Frame Type

This field designates the frame type to use when communicating with the server. If the required Frame Type is unknown, Auto-Detect can be selected to automatically determine the frame type when the driver establishes initial contact with the server. In this case, once the frame type has been successfully detected, the driver will then remember this type and use it as the initial preference for subsequent connection attempts. The frame type is prioritized in the following order: 3E and 4E. Note that the auto-detection procedure may cause a communication error to be indicated on the server during the initial connection attempt, but this error can be cleared by power-cycling the server. To determine the required Frame Type for your equipment, please consult the server device documentation.

9.9.5 Command and Monitor Data Object Settings

In the studio's Project panel, add **G500ETH...Ethernet...SLMP Client...Command Data** and/or **Monitor Data**. The **Command Data** object will execute the Device Batch Read command to read command data from the targeted remote server device. The **Monitor Data** object will execute the Device Batch Write command to write status data to the targeted remote server device.

Description

This 32-character (max) field is strictly for user reference: it is not used at any time by the driver.

Device Code

For user convenience, a selection of well-known device codes is provided. However, this in no way limits the variety of device codes that can be targeted. Any device code that is supported by the server device can be specified by choosing "Other Device Code" from the drop-down box, and then entering the code in the **Code Value** field.

Code Value

This field is enabled only if **Device Code** is set to "Other Device Code". If the desired device code is not available in the **Device Code** drop-down, then choose "Other Device Code" and enter the appropriate code in this field.

Starting Point

Specifies the initial point in a device's range of points. Enter a value from 0...16777215 (0x0...0xFFFFFFFF). If the connection uses 1E frames and the **Device Code** or **Code Value** targets a bit device, then the starting point should be a multiple of 16.

Number of Words

Specifies the number of words (1...719) from the **Starting Point** that are to be accessed. The server may also impose additional device-specific limitations (please consult the server device's documentation).

Function Code

The function code (refer to section 4) associated with the word offset. For the Monitor Data object, enter a "status" function code to be monitored. For the Command Data object, enter a "command" function code that can be written.

Network Number

Specifies the network number of the target station (0x0...0xEF).

Node Number

Specifies the node number of the target station (0x1...0x78, 0x7D, 0x7E, 0xFF).

Processor Number

Specifies the processor number of the target station (0x3D0...0x3D3, 0x3E0...0x3E3, 0x3FF).

9.9.6 *Diagnostics Objects*

Each command and monitor data object can optionally include a diagnostics object for debugging and diagnostics. Refer to 6.14.

Diagnostic Index

Enter the diagnostic index at which to store the diagnostics information.

9.10 IEC 61850 Server

9.10.1 Overview

The IEC 61850 server driver allows an IEC 61850 client to interact with the interface card via GGIO (generic process I/O) objects, unbuffered reports and GOOSE messages. The GGIO process data is entirely user-configurable. The unbuffered report control block can be configured with any IEC 61850 configuration tool. The GOOSE communication parameters are also user-configurable.

Other notes of interest are:

- The ICD file can be obtained from the [product web page](#) on the internet.
- The CID file is generated by the studio. In the studio, navigate to **File...Generate File...Generate IEC 61850 CID File**.

9.10.2 Server Settings

In the studio's Project panel, navigate to **G500ETH...Ethernet...IEC 61850 Server**.

IED Name

The IED name is used for identification of this device on the IEC 61850 network. Enter a string between 1 and 32 characters in length.

Subnetwork Name

The name of the station "subnetwork" on which this device resides but it is not relevant to the operation of this device. This name is only meaningful for SCD files and is used when merging CID files to build a substation-wide SCD file.

TCP Port

The TCP listening port for accepting connections and receiving requests.

Authentication

Enable or disable password authentication. If authentication is enabled, the client must provide a valid password when attempting to connect to this device. The password is defined in section 6.5.1.

9.10.3 GOOSE Communication Parameters

In the studio's Project panel, navigate to **G500ETH...Ethernet...IEC 61850 Server**. The following configuration options will determine the values used in the GOOSE message.

Priority

802.1Q priority level encoded in the Ethernet frame as the Priority Code Point (PCP) that can be used for QoS.

VLAN ID

802.1Q virtual LAN (VLAN) ID that can be used in a VLAN-aware network.

Application ID

The client will only process GOOSE messages with a matching application ID.

Destination Multicast Address

The GOOSE message will be sent to the specified destination multicast address.

9.10.4 Generic Process I/O Status and Control Object Settings

The Generic Status objects define the structure of "status" data, which are typically read-only function codes. The Generic Control objects will define the structure of the "command" data, which are writeable function codes. These objects allow the creation of custom-built GGIO process I/O. Up to 100 Generic Status and 100 Generic Control objects can be created. Note that the inverter function code value is a 16-bit integer, so only the lower 16-bits of the object's 32-bit integer value is useful.

Name

The name of this object as discovered by the client. The name is automatically generated depending on the instance.

Instance

The instance number of this object. It is recommend to start at 0.

Function Code

The function code (refer to section 4) associated with the word offset. For the Generic Status object, enter a "status" function code to be monitored. For the Generic Control object, enter a "command" function code that can be written.

Data Type

Each data word is fixed to 16-Bit Unsigned. This is equivalent to two bytes.

Include in DataSet

Include or exclude this object from the fixed dataset.

The default configuration is described in Table 62. Always use the studio to confirm the configuration before commissioning the device.

Table 62: IEC 61850 Server User-Configurable I/O

Type	Name	Instance	Function Code
Generic Control	Control0	0	P100.01 (Command word)
Generic Control	Control1	1	P100.20 (Frequency command)
Generic Control	Control2	2	P100.21 (PID reference command)
Generic Control	Control3	3	P100.22 (PID feedback command)
Generic Control	Control4	4	P100.23 (Torque command)
Generic Control	Control5	5	P100.24 (Voltage command)
Generic Control	Control6	6	P100.02 (Motor group selection)
Generic Status	Status0	0	P100.51 (Status word)
Generic Status	Status1	1	P100.70 (Status word 2)
Generic Status	Status2	2	P100.71 (Running frequency)
Generic Status	Status3	3	P100.72 (Frequency reference)
Generic Status	Status4	4	P100.73 (Bus voltage)
Generic Status	Status5	5	P100.74 (Output voltage)
Generic Status	Status6	6	P100.75 (Output current)
Generic Status	Status7	7	P100.76 (Output torque)
Generic Status	Status8	8	P100.77 (Output power)
Generic Status	Status9	9	P100.79 (PID reference)
Generic Status	Status10	10	P100.80 (PID feedback)
Generic Status	Status11	11	P100.78 (Fault code)
Generic Status	Status12	12	P100.53 (Operation mode)

10 TROUBLESHOOTING

Although by no means exhaustive, Table 63 provides possible causes behind some of the most common errors experienced when using the interface card.

Table 63: Troubleshooting

Problem	Symptom	Solution
Inverter fault	Inverter displays a fault code	<ul style="list-style-type: none"> Refer to the Troubleshooting section in the G500 Operation Manual.
No communications between the interface card and the inverter	"MODULE STATUS" LED is flashing red every 1 second.	<ul style="list-style-type: none"> Confirm that the interface card connector is properly seated. Rebooting the interface card via the Galt Electric Configuration Studio disrupts the communication with the inverter. Reset the fault. If the card is connected in a ring topology, all devices in the ring must be configured with the same ring redundancy protocol (i.e. MRP). The appropriate ring redundancy protocol must also be enabled on the interface card. Otherwise a ring topology will create an Ethernet loop and result in undefined/erratic behavior.
No communications between the network and the interface card	Communications cannot be established, the Ethernet "link" LED is off, or the Ethernet "activity" LED flashes only infrequently or not at all	<ul style="list-style-type: none"> Confirm that the card is running normally (Module Status LED is not blinking red) and connected to the local Ethernet network. Ensure that the card is programmed with compatible network settings. Consult with your network administrator to determine the compatible settings. Confirm that the destination IP address programmed into the controller equipment or computer matches that of the interface card, as displayed by the studio. Confirm that intermediate firewalls or routers have been configured to allow access to the interface via the applicable TCP/UDP ports. Try a known working Ethernet cable and switch. 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.
No PROFINET communication	PROFINET I/O communication cannot be established. The "Network Status" LED is not solid green.	<ul style="list-style-type: none"> Confirm that the card's PROFINET device name matches the name assigned in the controller's configuration. Confirm that the card's network settings match the settings assigned in the controller's configuration. Confirm that the I/O cycle update time is set to 1ms or larger. Ensure that the card is connected to a 100Mbps full duplex capable switch. Ensure that the card can be discovered using the controller's discovery tool.

Problem	Symptom	Solution
Unable to control the inverter via network communications	Writing to command and frequency function codes/registers has no apparent effect on inverter operation	<ul style="list-style-type: none"> Confirm that the applicable inverter function codes are set to allow network control (refer to section 3). If using the inverter's terminal contacts, refer to the inverter's instruction manual to determine the appropriate behavior and priority.
Studio cannot discover the card	The studio does not display the card under "Online Devices"	<ul style="list-style-type: none"> Confirm that the card is running normally and connected via USB or to the local Ethernet network. It is preferable to connect via USB as there are scenarios in which the Ethernet discovery is not available or is disabled. Confirm that the module and network status LEDs blink the green/red startup sequence when power is first applied. Add the studio as an exception to the computer's firewall. Add UDP port 4334 as an exception to the firewall. Temporarily disable the computer's firewall.
Studio cannot access file system	The studio displays an error when uploading and downloading the configuration.	If the studio continually displays an error regarding access to the file system, the card's file system may be corrupt. Please format the card's file system and then restore the configuration (refer to section 6.12). If the file system cannot be formatted, please contact technical support for additional assistance.
Cannot change network settings via inverter keypad	Network setting parameters revert to default or last known settings	<ul style="list-style-type: none"> Ensure that all inverter network setting parameters are configured appropriately. The IP Address, Subnet Mask, and Default Gateway Address must ALL be valid and compatible. Refer to section 3.1. Remove any USB or FTP connection.
Firmware-generated error	"MODULE STATUS" LED is flashing red. The number of times the LED flashes indicates an error code.	Record the error code blinking pattern and contact technical support for further assistance.



GALT ELECTRIC

California USA
www.galtelectric.com
1-800-511-7734