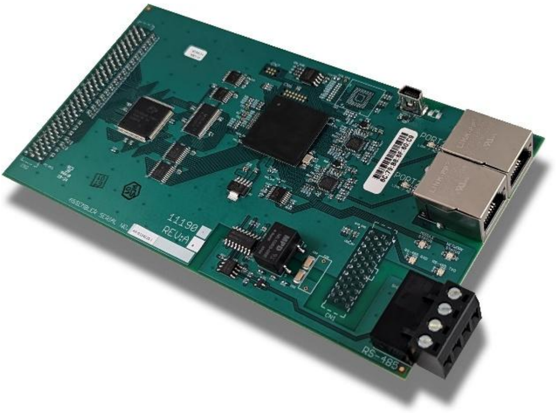
Instruction Manual

ICC

INDUSTRIAL CONTROL COMMUNICATIONS, INC.

PC61910P231/ASD-MVETHV2

Multiprotocol Ethernet and RS-485 Interface Card



⚠CAUTION

Thank you for purchasing the Multiprotocol Ethernet and RS-485 Interface Card.

- This product is designed to connect the Toshiba® T300MVi®, T300MV2®, & MTX® series of medium voltage inverters to Ethernet and RS-485 communication networks. Please read this instruction manual thoroughly in order to become familiar with proper interface card handling, installation and usage procedures.
- Improper handling may inhibit correct operation or cause premature interface card failure.
- Please deliver this instruction manual to the end user of the interface card, and retain it in an accessible location.
- For inverter usage instructions, please refer to the applicable manufacturer's inverter instruction manual.

January 16, 2025
Part #11191

Multiprotocol Ethernet and RS-485 Interface Card Instruction Manual

Part Number 11191
Printed in U.S.A.

Industrial Control Communications, Inc. 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 Toshiba T300MVi, T300MV2, or MTX medium voltage inverter to Industrial Ethernet and RS-485 networks using the PC61910P231/ASD-MVETHV2 Multiprotocol Ethernet and RS-485 interface card. This instruction manual does not contain inverter usage instructions. Please refer to this instruction manual in conjunction with the applicable manufacturer's 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.

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. Open only the access door to the control section of the inverter.
- 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 card 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 inverter doors or remove any covers 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 parameters are incorrectly configured, or configured without adequate understanding of the inverter and the load equipment, severe damage to the inverter load equipment may occur. Confirm the settings of all parameters 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. Open only the access door to the control section of the inverter.
- 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 parameter initialization may reset all inverter parameters to their factory default settings. After performing this operation, remember to reenter any custom parameter values prior to starting operation.

Icons

The following icons are used throughout this manual:

Note 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.

Tip 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

1.1 Product Overview

This interface card allows information to be transferred seamlessly between a Toshiba medium voltage inverter and various fieldbus networks with minimal configuration requirements. The interface card 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 and an isolated half-duplex RS-485 port. In addition to the supported fieldbus protocols, the interface card 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 card, 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 card, and keep this instruction manual in a safe place for future reference or unit inspection.

Note that different interface card 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 card as it must correspond to this manual's respective release date in order for all documented aspects to apply.

Supported Protocols

The interface card currently provides 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)
- IEC 61850 Server
- Modbus RTU Slave

1.2 Features and Specifications

Table 1: Features

Item	Description
Simultaneous Ethernet Protocols	Supports all standard unmodified Ethernet (SUE) protocols simultaneously. Ethernet protocols coexist with and operate independently of the serial protocols.
Serial Protocols	Any single serial (RS-485) protocol can be enabled. The serial protocol coexists with and operates independently of the Ethernet protocols.
Configuration Studio	Graphical user interface for discovery, configuration, management, monitoring and firmware update
Communication Loss Detection	Configurable actions for "fail-safe" conditions
Field Upgradeable	Firmware updates are 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
LED Indicators	Module Status, Network Status, 2 x Ethernet Link/Activity, RS-485 RXD, RS-485 TXD
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: Serial Hardware Specifications

Item	Description
Number of RS-485 Ports	1
Standard	TIA/EIA-485
Unit Load	1/8
Maximum Devices	32 (1 unit load), 256 (1/8 unit load)
Isolation	Galvanic isolation (1.5kV)
Internal Biasing Resistors	1K Ohm pull-up to 5V, 1K Ohm pull-down to COM
Internal EOL Termination	None
Failsafe Receiver	Full-failsafe (open, shorted, terminated and undriven)
Differential Output Max	5.0V (no load)
Differential Output Min	2.4V (terminated and fully loaded)
Signals	2-wire half-duplex: D+, D-, COM, SHIELD *SHIELD is not internally connected
Baud Rates	1200, 2400, 4800, 9600, 19200, 38400, 57600, 76800, 115200
Stop Bits	1, 2
Parity	None, Even, Odd
Port/Connector Type	Pluggable terminal block
Cable Type	Twisted pair, bare end wires
Cable Length	4000 feet (1200 meters) per segment max at 100kbps
Topologies	Linear/Bus/Daisy-chain

Table 5: 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 6: EtherNet/IP Specifications

Item	Description
Conformance Tested	ODVA EtherNet/IP Conformance Test Software Version CT-13
Product Type Code	12 (Communications Adapter)
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 25 input words, user configurable
I/O Output Size	Max 10 output words, user configurable
Generic (User Configurable) Assembly Instances	100 (input) and 150 (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 7: 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)s
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 8: 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-DDB-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 9: 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 25 input words, user configurable
I/O Output Size	Max 10 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 10: 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	25 MV (Measured Value integers)
Generic Control Objects	10 APC (Controllable Analog Process Value integers)
Authentication	None/Password, configurable

Table 11: Modbus RTU Specifications

Item	Description
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)
Broadcast	Yes, supported with function codes: 5, 6, 15, 16
Max Read Register Size	125 registers
Max Write Register Size	123 registers
Register Data Type	16-bit integer
Processing Time	Typically less than 1ms

Table 12: Applicable Inverters

Series	Model	Capacity
T300MVi, T300MV2, & MTX	All models	All capacities

Table 13: Environmental Specifications

Item	Description
Operating Environment	Indoors, less than 1000m above sea level, do not expose to direct sunlight or corrosive / explosive gasses
Operating Temperature	-10 ~ +50°C (+14 ~ +122°F)
Storage Temperature	-40 ~ +85°C (-40 ~ +185°F)
Relative Humidity	20% ~ 90% (without condensation)
Vibration	5.9m/s^2 (0.6G) or less (10 ~ 55Hz)
Cooling Method	Self-cooled
RoHS (Lead free)	Yes

1.3 Component Overview

Figure 1 provides an overview of the important interface card components.

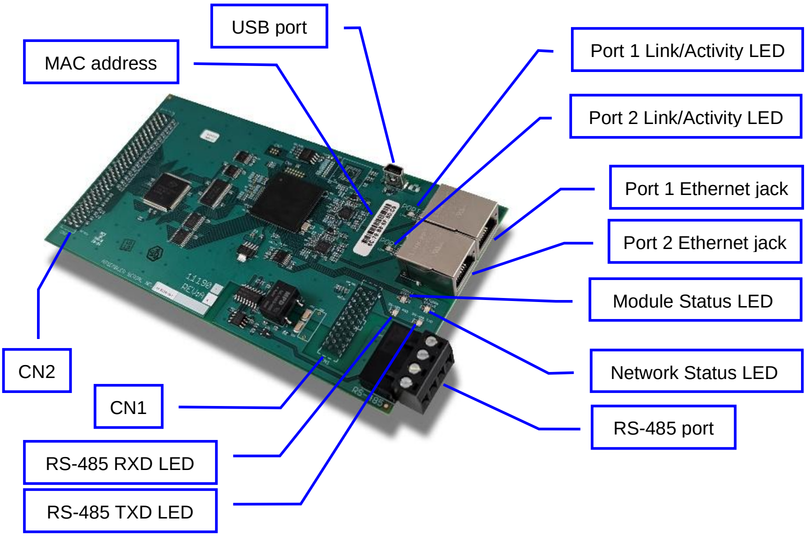


Figure 1: Interface Card Component Overview

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 interface cards can be connected together by daisy-chaining one of the ports to the next inverter in line. In ring topologies, a ring redundancy protocol (i.e. MRP, DLR) must be supported by all devices on the network and enabled on the participating Ethernet ports.

RS-485 Port

Optically isolated 2-wire half-duplex RS-485 port. Pluggable terminal block. Refer to section 3.

Connector CN1 and CN2

Connects to mating connectors on the inverter's control board. Refer to section 2.2.

USB Port

USB 2.0 port with mini-B connector. Used to access the interface card via the studio (refer to section 6) and as a USB flash drive (refer to section 8).

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 and Activity LEDs

One set of LEDs are provided for each Ethernet port. These LEDs provide insight into the Ethernet network's status and activity. Refer to section 1.4.

RS-485 RXD and TXD LEDs

These LEDs provide insight into the RS-485 network's status and activity. Refer to section 1.4.

1.4 LED Indicators

1.4.1 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
Red Blink	Activity	Indicates when a packet is transmitted or received on this port

1.4.2 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	Refer to the TROUBLESHOOTING section.

1.4.3 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

1.4.4 RS-485 RXD

LED Activity	Status	Note
Off	No Receive Data	There is no data detected on the network.
Green Light	Receive Data	Lights when receiving data. This does not indicate the validity of the data with respect to a particular protocol: only that data exists and is being detected. The LED will also light when this device transmits data.

1.4.5 RS-485 TXD

LED Activity	Status	Note
Off	No Transmit Data	This device is not transmitting data.
Green Light	Transmit Data	Lights when this device is transmitting data.

2 INSTALLATION

2.1 Pre-Installation Instructions

Before opening the drive, please observe all safety precautions as outlined on the unit's front cover and in the operation manual. Installation of the interface card should only be performed by a qualified technician familiar with the maintenance and operation of the drive.

2.2 Installation Procedure

1. ⚡ **CAUTION!** Verify that all input power sources to the drive have been turned OFF and are locked and tagged out.

2. ⚠ **DANGER!** ⚡ Wait at least 5 minutes for the drive's electrolytic capacitors to discharge before proceeding to the next step. **Do not touch any internal parts with power applied to the drive, or for at least 5 minutes after power to the drive has been removed. A hazard exists temporarily for electrical shock even if the source power has been removed**.

3. Open the drive's control section door only. (Refer to the drive manual and drawings).

4. Install the provided standoff onto the drive control board (Refer to the drive manual and drawings).

5. Press the interface card firmly onto the appropriate connectors on the drive's control board. (Refer to the drive manual and drawings.)

6. Install the provided screw into the mounting standoff.

7. Connect the various network cables to their respective plugs/terminal blocks. Ensure that any terminal blocks are fully seated into their respective headers, and route the network cables such that they are located well away from any electrical noise sources, such as ASD input power or motor wiring. Also take care to route all cables away from any sharp edges or positions where they may be pinched.

8. Take a moment to verify that the interface card is seated properly, and that all network cables have sufficient clearance from electrical noise sources.

9. Close the drive's cabinet door.

10. Turn the power source to the drive ON, and verify that the drive functions properly. If the drive does not appear to power up, or does not function properly, immediately turn power OFF. **Repeat steps 1 and 2 to remove all power from the drive.** Then, verify all connections. Contact Toshiba© for assistance if the problem persists.

3 RS-485 PORT

Figure 2 shows the network connections to the interface card's RS-485 terminal block.

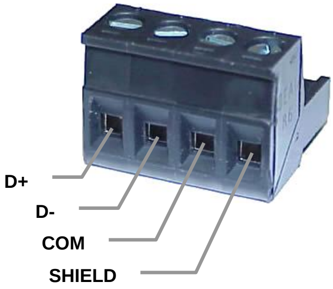


Figure 2: RS-485 Terminal Block Connections

Table 14: RS-485 Signals

Signal	Description
D+	This signal is required and must be connected to the "non-inverting" data line. The "D+" and the "D-" data lines must use twisted pair wiring.
D-	This signal is required and must be connected to the "inverting" data line. The "D+" and the "D-" data lines must use twisted pair wiring.
COM	This signal is required and must be connected to the RS-485 signal common. It will improve data integrity and help prevent damaging the port. The RS-485 network should have only one signal common. This should NOT be connected to the chassis ground or network shield.
SHIELD	This signal is optional and has no internal connection on this device. Its purpose is simply to provide a cable shield chaining location between devices. The shield is then typically connected to ground at one location only.

4 INVERTER PARAMETER CONFIGURATION

Note that whenever any of the parameters documented in this section are changed, they must be written to the EEPROM and the control board must be initialized (power cycled OFF and back ON again) before the changes will take effect.

For further details regarding these parameters, please refer to the appropriate drive **Instruction Manual**.

4.1 General Configuration

Program…Communication…Comm configuration…COMM_TYPE must be set to `0400h`.

4.2 Receive (Command) Data Configuration

Program…Communication…Comm addressing…Comm read addressing…SCAN_R_ADRS must be set to `0`.

Program…Communication…Comm addressing…Comm read addressing…SCAN_R_SIZE must be set to the number of command items the drive is to receive. For example, if only two command words are to be sent to the drive, then this value can be set to 2. This will enable the first two `SCAN_RCVxx_AS` parameters. Note that there is no penalty in setting this parameter to its maximum value (10): any "excess" parameters that are not written via the network will have no effect on the drive's operation.

Program…Communication…Comm addressing…Comm read addressing…SCAN_RCV01_AS – SCAN_RCV10_AS should be set as needed to reference the desired command words. Any unused addresses should be programmed as `DUST`. Note that although all 10 of these parameters can be written via the interface card at any time, only `SCAN_R_SIZE` number of parameters will be recognized by the drive.

4.3 Write (Status) Data Configuration

Program…Communication…Comm addressing…Comm write addressing…SCAN_W_ADRS should be set to `10`.

Program…Communication…Comm addressing…Comm write addressing…SCAN_W_SIZE must be set to the number of status items the drive is to transmit. For example, if only four status words are to be received from the drive, then this value can be set to 4. This will enable the first four `SCAN_WRxx_AS` parameters. Note that there is no penalty in setting this parameter to its maximum value (25). Also note that any parameter larger than `SCAN_W_SIZE` that is accessed via the interface card will always return a value of 0. For example, if `SCAN_W_SIZE` is set to a value of `5` and `SCAN_WR07_AS` is accessed via the network, then the value returned will always be 0.

Program…Communication…Comm addressing…Comm write addressing…SCAN_WR01_AS – SCAN_WR25_AS should be set as needed to reference the desired status words. Any unused addresses should be programmed as `DUST`. Note that although all 25 of these parameters can be read via the interface card at any time, only `SCAN_W_SIZE` number of parameters will contain actual data.

5 PARAMETER NUMBERING AND BEHAVIOR

5.1 Register Numbers

Each parameter has a corresponding register number, as defined in the Table 15 and can be conveniently referenced in the studio (section 7.7). These register numbers are used when accessing parameters via certain Ethernet protocols (i.e. Modbus, AB CSP). The terms “parameter” and “register” refer to data stored on the inverter and will be used interchangeably throughout this documentation.

Note that not all of the available registers that exist in the register map have meaningful data. Please refer to sections 4.2 and 4.3.

Table 15: Parameter-to-Register Mapping

Parameter	Register Number	Input to ASD / Output from ASD	Read Only or Read/Write
SCAN_RCV01_AS	1	Input “Command” Data	Read/Write
SCAN_RCV02_AS	2	Input “Command” Data	Read/Write
SCAN_RCV03_AS	3	Input “Command” Data	Read/Write
SCAN_RCV04_AS	4	Input “Command” Data	Read/Write
SCAN_RCV05_AS	5	Input “Command” Data	Read/Write
SCAN_RCV06_AS	6	Input “Command” Data	Read/Write
SCAN_RCV07_AS	7	Input “Command” Data	Read/Write
SCAN_RCV08_AS	8	Input “Command” Data	Read/Write
SCAN_RCV09_AS	9	Input “Command” Data	Read/Write
SCAN_RCV10_AS	10	Input “Command” Data	Read/Write
SCAN_WR01_AS	11	Output “Status” Data	Read Only
SCAN_WR02_AS	12	Output “Status” Data	Read Only
SCAN_WR03_AS	13	Output “Status” Data	Read Only
SCAN_WR04_AS	14	Output “Status” Data	Read Only
SCAN_WR05_AS	15	Output “Status” Data	Read Only
SCAN_WR06_AS	16	Output “Status” Data	Read Only
SCAN_WR07_AS	17	Output “Status” Data	Read Only
SCAN_WR08_AS	18	Output “Status” Data	Read Only
SCAN_WR09_AS	19	Output “Status” Data	Read Only
SCAN_WR10_AS	20	Output “Status” Data	Read Only
SCAN_WR11_AS	21	Output “Status” Data	Read Only
SCAN_WR12_AS	22	Output “Status” Data	Read Only
SCAN_WR13_AS	23	Output “Status” Data	Read Only
SCAN_WR14_AS	24	Output “Status” Data	Read Only
SCAN_WR15_AS	25	Output “Status” Data	Read Only
SCAN_WR16_AS	26	Output “Status” Data	Read Only
SCAN_WR17_AS	27	Output “Status” Data	Read Only
SCAN_WR18_AS	28	Output “Status” Data	Read Only
SCAN_WR19_AS	29	Output “Status” Data	Read Only
SCAN_WR20_AS	30	Output “Status” Data	Read Only

Parameter	Register Number	Input to ASD / Output from ASD	Read Only or Read/Write
SCAN_WR21_AS	31	Output “Status” Data	Read Only
SCAN_WR22_AS	32	Output “Status” Data	Read Only
SCAN_WR23_AS	33	Output “Status” Data	Read Only
SCAN_WR24_AS	34	Output “Status” Data	Read Only
SCAN_WR25_AS	35	Output “Status” Data	Read Only

5.2 Scanned Parameters

The interface card provides network access to the specified list of parameters contained in the *param.xml* file located in the “*WEB*” folder of the interface card’s file system. These parameters are constantly being read and/or written (as applicable) by the interface card, and their current values are therefore mirrored in the interface card’s internal memory.

Accesses to any parameter will always be successful. Even if an inverter parameter does not exist in the *param.xml* file, the interface card still maintains a placeholder location in its internal mirroring memory for that parameter. Care must be taken to utilize only the parameters that are known to exist (refer to sections 4.2 and 4.3) and that are also specified in the *param.xml* file.

The *param.xml* file should not be modified except under very special circumstances: contact technical support for guidance prior to any modification of this file.

6 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 10 and 0 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.

If the timeout action is set to “Apply Fail-safe Values”, the card can write fail-safe values to any inverter parameter when a timeout event is triggered. Refer to section 7.5.1 for details on how to configure fail-safe values.

7 ASDMV CONFIGURATION STUDIO

7.1 Overview

The interface card is discovered, configured and updated by the Configuration Studio PC application (refer to Figure 3). The studio typically requires an Ethernet connection for remote discovery, network setting, configuration, real-time monitoring, and firmware updates. Configuration, real-time monitoring 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 studio configuration concepts. For protocol-specific configuration, refer to the relevant protocol section.

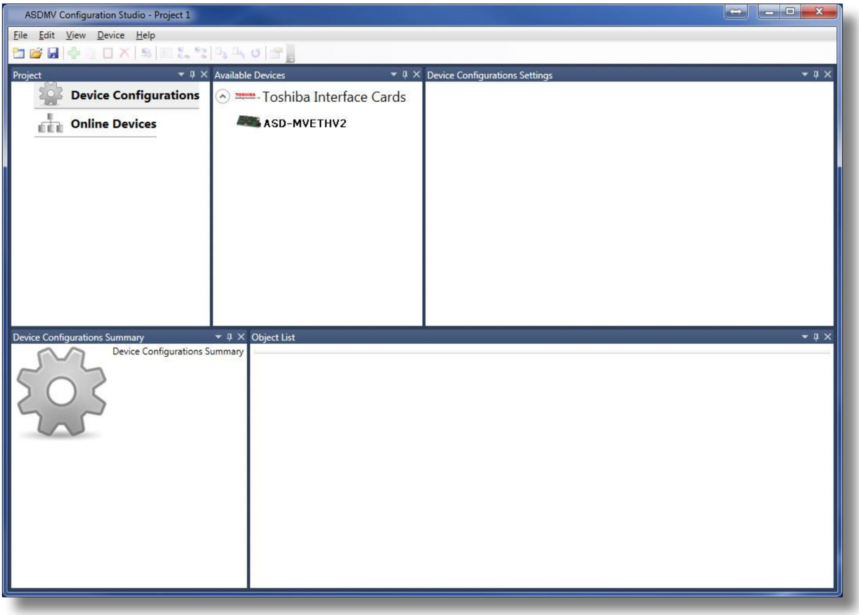

ASDMV Configuration Studio - Project 1
File Edit View Device Help
Project
Device Configurations
Online Devices
Available Devices
Toshiba Interface Cards
ASD-MVETHV2
Device Configurations Settings
Device Configurations Summary
Device Configurations Summary
Object List

Figure 3: 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 can be online with multiple devices simultaneously.

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 control 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.

7.2 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.

7.3 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.

7.4 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.

7.4.1 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").

7.4.2 Network Configuration

The interface 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.

7.4.3 Traffic Storm Protection

A traffic storm occurs when packets flood the network. This excess traffic can cause processing overloads in the device and affect operation. The network traffic must be analyzed to determine whether traffic storms exist and if they are causing issues with the device before enabling this feature.

If this optional feature is enabled, the device will detect a traffic storm condition and take action to prevent processing overload. If the number of received packets exceed the configured threshold within a one millisecond interval, the device will drop all received packets thereafter in an attempt to prevent processing overload. Packets will be dropped for up to 100 milliseconds before the device resumes normal operation and restarts its check for a traffic storm condition. Note that inappropriate threshold values may disrupt communication with the device and negatively impact the overall application.

Note that this feature does not affect packet forwarding/routing between the two Ethernet ports of the device.

Enable

Enables the traffic storm protection feature.

Threshold

Defines the number of packets received in a 1ms period before traffic storm protection is engaged. The allowable range is 5 to 100 packets. The default is 16 packets.

7.5 Internal Logic Settings

7.5.1 Fail-safe Values

7.5.1.1 Overview

The interface card can be configured to perform a specific set of actions when network communication is 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

7.5.1.2 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-communication conditions. These conditions then trigger timeout processing events. If it is desired to not 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.

7.5.1.3 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 parameter(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.

Parameter

Enter the parameter (in section 5).

Data Type

This is the size of valid values and is fixed to “16-Bit Unsigned” allows for a range of timeout values between 0 and 65535.

Value

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

7.5.2 Fail-safe Example

This example will demonstrate how to add one timeout object which will assign a value of 2000 (20.00Hz) to function code S05 (frequency command). In the **Project** panel, select the device and add **Internal Logic…Fail-safe Values…Timeout Object** as shown in Figure 4. The red error indicators are normal at this stage as the **Timeout Object Settings** have not yet been configured.

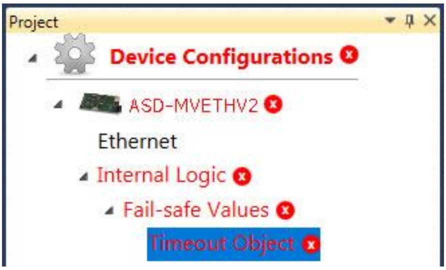
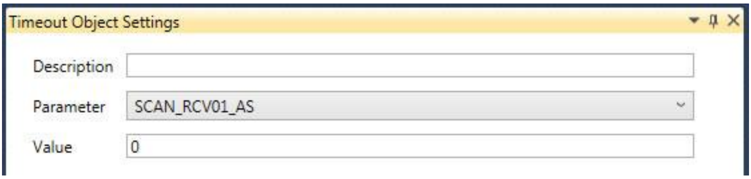


Figure 4: Timeout Object Project Panel

Next, configure the **Timeout Object Settings** as shown in Figure 5.



Description

Parameter SCAN_RCV01_AS

Value 0

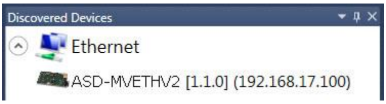
Figure 5: Timeout Object Settings

The example is complete.

7.6 Discovery over Ethernet

Depending on the currently-enabled driver, the Configuration Studio will automatically discover the device on the Ethernet network, regardless of whether or not the interface 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 6.)

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.



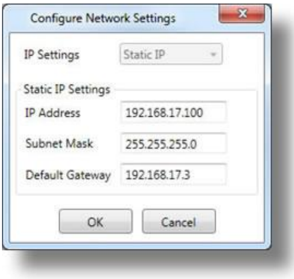
Ethernet
ASD-MVETHV2 [1.1.0] (192.168.17.100)

Figure 6: Configuration Studio Discovery over Ethernet

The network settings of a discovered interface 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...**

The network settings pop-up should appear similar to Figure 7. 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.



IP Settings: Static IP
IP Address: 192.168.17.100
Subnet Mask: 255.255.255.0
Default Gateway: 192.168.17.3

Figure 7: Remotely Configure Network Settings

7.7 Manage Device Parameters

The accessibility of the inverter parameters can be adjusted. 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 list serves as a useful reference when configuring objects.

7.8 Monitor Device Parameters

The inverter's parameter values can be monitored and commanded in real-time. 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.

7.9 Backup and Restore Parameters

The parameter values can be backed up from the inverter and restored to the inverter. 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 7.7). 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…**

7.10 Restore Factory Settings

When connected via USB, the interface card can be restored to 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**.

7.11 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.

8 FILE SYSTEM

8.1 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 and the embedded web server. 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 interface card is installed.

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

Interacting with the 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 interface 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 interface card is running normally in a production environment.

8.2 USB with Windows Explorer

To use Microsoft Windows Explorer, first open either "Windows Explorer" or "My Computer". Refer to Figure 8. 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 9.

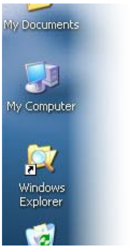


Figure 8: Accessing Windows Explorer

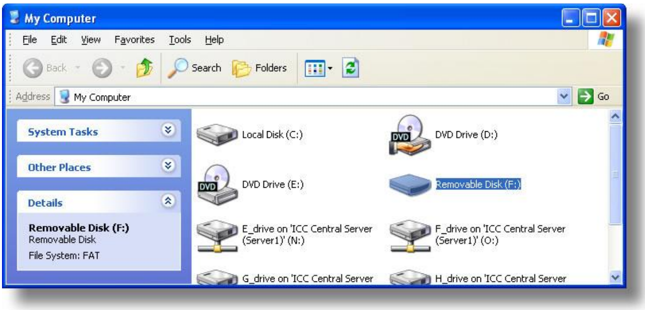
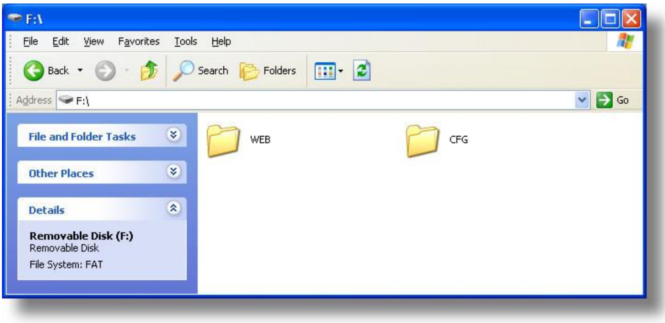


Figure 9: Removable Disk with Windows Explorer

Windows Explorer will then display the file system's contents (refer to Figure 10.) 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.



Address: F:\

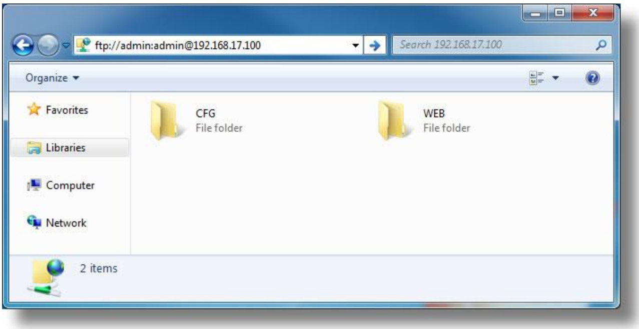
Removable Disk (F:)
Removable Disk
File System: FAT

Figure 10: USB File Access via Windows Explorer

8.3 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 11.



ftp://admin:admin@192.168.17.100

Figure 11: 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 interface card’s file system.

8.4 Loading New Web Server Content

The interface card’s web server resides in the file system and can be updated in the field. This section will discuss how to update the web server.

Besides the new “WEB” folder containing the new web server, the update requires a USB or FTP connection as described earlier in this section. To update the web server, complete the following steps:

1. Navigate to the interface card’s file system (see section 8.2 or 8.3).
2. Backup the “WEB” folder if desired by copying it to the local computer.
3. Delete the “WEB” folder from the interface card’s file system.
4. Copy the new “WEB” folder to the interface card’s file system.

5. Although it is not typical, if your *param.xml* file was specially modified (for a custom application, for example), it may be necessary to re-apply those modifications using the **Manage Device Parameters** (refer to section 7.7). Please consult technical support for any questions related to customized versions of *param.xml*.

6. Power-cycle or reset the card.

7. Clear your internet browser's cache to ensure that the new web server content will be properly loaded from the interface card.

9 FIRMWARE

9.1 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 interface card production activities prior to initiating the firmware update procedure.

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

9.2 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 interface card may be temporarily inaccessible during the firmware update process.

5. *Locally via USB:* Connect a USB cable between the interface 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 interface card has compatible network settings.

7. Once the firmware update process has started, do not interrupt the interface card as this may corrupt the firmware. Do NOT manually power-cycle the inverter or reboot the interface card. Do NOT disturb the USB or Ethernet (FTP) connection.

8. After the firmware update has been completed, the interface card will reset automatically. When the interface card boots up again, it will be running the new application firmware, which can be confirmed by observing the version displayed in Configuration Studio.

10 ETHERNET PROTOCOL-SPECIFIC INFORMATION

This section will discuss topics that are specific to each of the supported Ethernet protocols. To enable a protocol, add the protocol to the Ethernet configuration.

10.1 Modbus/TCP

10.1.1 Overview

The interface card supports Schneider Electric's Modbus/TCP protocol, release 1.0. The interface card 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.

10.1.2 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.

10.1.3 Functions

Supported Modbus/TCP functions are indicated in Table 16.

Table 16: 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

10.1.4 Holding & Input Registers

The inverter registers by default are mapped as both holding registers (4X) and input registers (3X) and are accessed by using the inverter register numbers described in section 5.1. Examples are shown in Table 17.

Table 17: Modbus/TCP Register Examples

Function Code	Holding/Input Register Number
SCAN_RCV01_AS	1
SCAN_WR01_AS	11

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 40001 is the same as Modbus holding register 1. The same description applies to input registers (3X).

For example, from a Modbus/TCP master's point of view, in order to access the parameter SCAN_WR01_AS (register 11) as a holding register, the Modbus/TCP master must execute the Read

Multiple Registers function code and target register 11. This will similarly apply when accessing an inverter parameter as an Input Register.

10.1.5 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 indices into various bits of existing registers. What this means is that when a discrete is accessed, that discrete is resolved by the interface card 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:

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

Equation 1

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:

$$\text{bit} = (\text{discrete} - 1)\,\%\,16$$

Equation 2

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 1 and Equation 2 in a calculation example. Say, for instance, that we are going to read coil #34. Using Equation 1, we can determine that coil #34 resides in register #3, as $\lfloor 3.0625 \rfloor = \lfloor 3\ r1 \rfloor$ = 3. Then, using Equation 2, we can determine that the bit within register #3 that coil #34 targets is (34-1)%16 = 1, as 33%16 = mod(2 r1) = 1. Therefore, reading coil #34 will return the value of register #3, bit #1.

10.1.6 Connection Timeout Options

In the studio's Project panel, navigate to **ASD-MVETHV2…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.

<u>Enable Supervisory Timer</u>

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**.

<u>Enable Connection Timer</u>

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 7.5.1.

10.1.7 Node Settings

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

10.1.8 Holding/Input Register Remap Settings

In the studio's **Project** panel, add **ASD-MVETHV2…Ethernet…Modbus/TCP Server…Node…Holding/Input Register Remap**.

The holding/input register remap objects are **OPTIONAL**. By default, all inverter parameters are already mapped as both holding (4X) and input (3X) registers (refer to section 10.1.4). For user convenience, register remap objects can be created to map any inverter parameter to holding/input register 5001 to 5050.

At times, it may be convenient to access inverter parameters in bulk Modbus transactions. This may be especially true in situations where it is desired to access certain parameters that are non-contiguous. For example, if it were desired to read the parameters SCAN_WR01_AS (register 11), SCAN_WR21_AS (register 31), and SCAN_WR25_AS (register 35), 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 11 for a quantity of 25 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 parameters can be grouped together in any order and accessed efficiently via the Modbus/TCP "read multiple registers" and "write multiple registers" functions. 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

Remap register that maps to the specified inverter parameter. Select from 5001 to 5050.

Parameter

Parameter (in section 5) that is accessed by the **Remap Register**.

Data Type

Fixed to 16-Bit Unsigned. This is equivalent to two bytes.

10.2 EtherNet/IP

10.2.1 Overview

EtherNet/IP is a network adaptation of ODVA's Common Industrial Protocol (CIP). The interface 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 the generic I/O assembly instances 100 and 150 (refer to section 10.2.4), which is entirely user-configurable. 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 processing time and allows for time-critical performance.

With explicit messaging (refer to section 10.2.6), 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).
- The ESI file can be obtained from the product web page on the internet.
- 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 12 (Communications Adapter).
- 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.

10.2.2 Server Settings

In the studio, navigate to **ASD-MVETHV2…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.

10.2.3 Connection Timeout Options

In the studio's Project panel, navigate to **ASD-MVETHV2…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 the run/idle header being set 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 7.5.1.

10.2.4 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 10 "command" parameter values can be sent to the inverter, and up to 25 "status" parameter values can be sent back to the controller. Therefore, up to 25 Produced and 10 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 18.

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 parameter 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 parameter. It is recommend to start at word offset 0.

Parameter

The inverter parameter (in section 5) associated with the word offset. For the Produced Data Word object, enter a "status" parameter to be monitored. For the Consumed Data Word object, enter a "command" parameter that can be written.

Data Type

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

Table 18: EtherNet/IP User-Configurable I/O Data Format

Consumed Data (PLC to Inverter)		Produced Data (Inverter to PLC)	
Word Offset	**Parameter**	**Word Offset**	**Parameter**
0	SCAN_RCVxx_AS	0	Any
1	SCAN_RCVxx_AS	1	Any
:	SCAN_RCVxx_AS	:	Any
8	SCAN_RCVxx_AS	23	Any
9	SCAN_RCVxx_AS	24	Any

The default I/O configuration is described in Table 19.

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

Table 19: EtherNet/IP Default User-Configurable I/O Data Format

Consumed Data (PLC to Inverter)		Produced Data (Inverter to PLC)	
Word Offset	**Parameter**	**Word Offset**	**Parameter**
0	SCAN_RCV01_AS	0	SCAN_WR01_AS
1	SCAN_RCV02_AS	1	SCAN_WR02_AS
2	SCAN_RCV03_AS	2	SCAN_WR03_AS
3	SCAN_RCV04_AS	3	SCAN_WR04_AS
4	SCAN_RCV05_AS	4	SCAN_WR05_AS
5	SCAN_RCV06_AS	5	SCAN_WR06_AS
6	SCAN_RCV07_AS	6	SCAN_WR07_AS
7	SCAN_RCV08_AS	7	SCAN_WR08_AS
8	SCAN_RCV09_AS	8	SCAN_WR09_AS
9	SCAN_RCV10_AS	9	SCAN_WR10_AS
10	NA	10	SCAN_WR11_AS
11	NA	11	SCAN_WR12_AS
12	NA	12	SCAN_WR13_AS
13	NA	13	SCAN_WR14_AS
14	NA	14	SCAN_WR15_AS
15	NA	15	SCAN_WR16_AS
16	NA	16	SCAN_WR17_AS
17	NA	17	SCAN_WR18_AS
18	NA	18	SCAN_WR19_AS
19	NA	19	SCAN_WR20_AS
20	NA	20	SCAN_WR21_AS

21	NA	21	SCAN_WR22_AS
22	NA	22	SCAN_WR23_AS
23	NA	23	SCAN_WR24_AS
24	NA	24	SCAN_WR25_AS

10.2.5 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 10.2.4). For a generic class 1 I/O application example, refer to section 10.2.10.1 or 10.2.11.

10.2.6 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 parameters by specifying the appropriate class code, instance number and attribute identifier. The class code is 0xA2. The instance number is the register number that is associated with the targeted inverter parameter (refer to section 5.1). The attribute identifier is 1, which is the 16-bit value of the parameter being accessed. Examples are shown in Table 20.

Table 20: Get/Set Attribute Single Examples

Parameter	Register Number	Service	Class	Instance	Attribute
SCAN_RCV01_AS	1	0x0E / 0x10	0xA2	1	1
SCAN_WR01_AS	11	0x0E / 0x10	0xA2	11	1

10.2.7 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 parameters 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 10.2.8. The elements are 16-bit values.

10.2.8 Inverter Parameter Access Tag Format

Any inverter parameter (in section 5) can be accessed with its own unique tag name, or an array tag can be used to access a group of parameters with one PLC instruction. The “tag name” is the ASCII representation of the parameter itself.

Examples

“SCAN_RCV01_AS”.......................SCAN_RCV01_AS
“SCAN_RCV10_AS”.......................SCAN_RCV10_AS
“SCAN_WR01_AS” ....SCAN_WR01_AS
“SCAN_WR25_AS” ....SCAN_WR25_AS

For explicit messaging examples, refer to sections 10.2.12 and 10.2.13.

10.2.9 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.

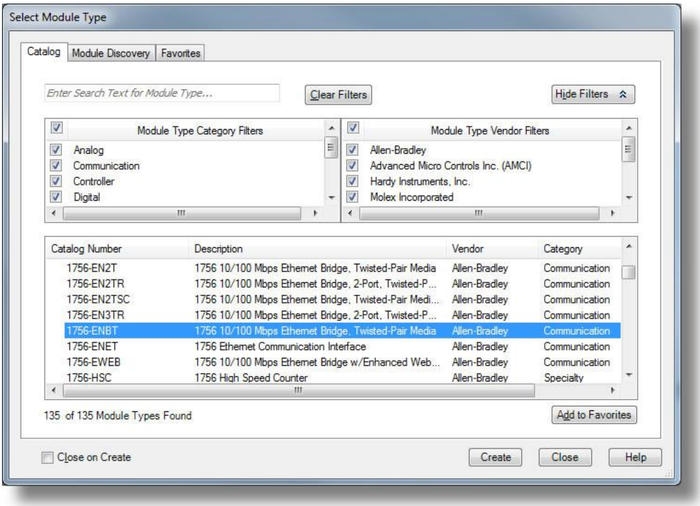
1) Run RSLogix 5000, and create a new configuration.

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 12.



Module Type Category Filters: ☑ Analog, ☑ Communication, ☑ Controller, ☑ Digital

Module Type Vendor Filters: ☑ Allen-Bradley, ☑ Advanced Micro Controls Inc. (AMCI), ☑ Hardy Instruments, Inc., ☑ Molex Incorporated

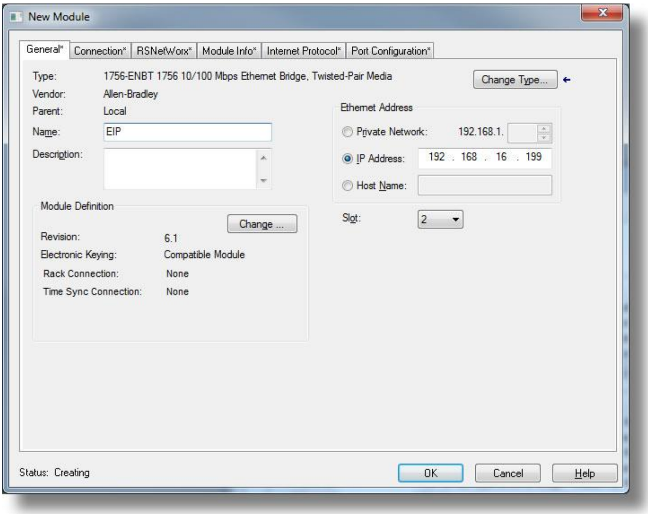
Catalog Number	Description	Vendor	Category
1756-EN2T	1756 10/100 Mbps Ethernet Bridge, Twisted-Pair Media	Allen-Bradley	Communication
1756-EN2TR	1756 10/100 Mbps Ethernet Bridge, 2-Port, Twisted-P...	Allen-Bradley	Communication
1756-EN2TSC	1756 10/100 Mbps Ethernet Bridge, Twisted-Pair Medi...	Allen-Bradley	Communication
1756-EN3TR	1756 10/100 Mbps Ethernet Bridge, 2-Port, Twisted-P...	Allen-Bradley	Communication
1756-ENBT	1756 10/100 Mbps Ethernet Bridge, Twisted-Pair Media	Allen-Bradley	Communication
1756-ENET	1756 Ethernet Communication Interface	Allen-Bradley	Communication
1756-EWEB	1756 10/100 Mbps Ethernet Bridge w/Enhanced Web...	Allen-Bradley	Communication
1756-HSC	1756 High Speed Counter	Allen-Bradley	Specialty

135 of 135 Module Types Found

□ Close on Create

Figure 12: Adding a New Module

6) The “New Module” window will open. Refer to Figure 13.



Type: 1756-ENBT 1756 10/100 Mbps Ethernet Bridge, Twisted-Pair Media
Vendor: Allen-Bradley
Parent: Local
Name: EIP
Description:

Module Definition
Revision: 6.1
Electronic Keying: Compatible Module
Rack Connection: None
Time Sync Connection: None

Ethernet Address
○ Private Network: 192.168.1.
● IP Address: 192 . 168 . 16 . 199
○ Host Name:

Slot: 2

Status: Creating

Figure 13: 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.

10.2.10 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 example in section 10.2.11. This section must be completed prior to attempting any of the following AOP example(s).

EtherNet/IP I/O messaging allows the inverter’s parameter 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 14.

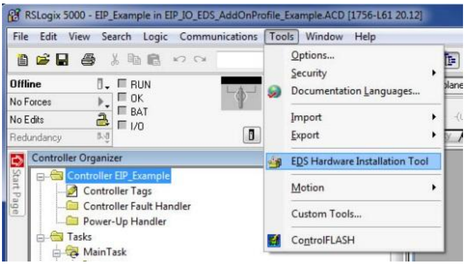
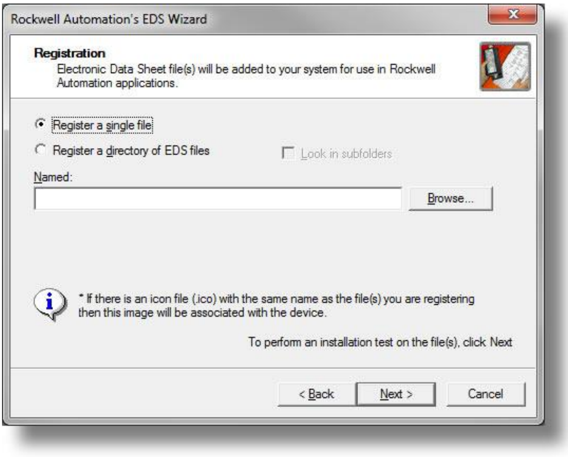


Figure 14: EDS Hardware Installation Tool Menu

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 15.


Rockwell Automation's EDS Wizard

Registration

Electronic Data Sheet file(s) will be added to your system for use in Rockwell Automation applications.

● Register a single file

○ Register a directory of EDS files   □ Look in subfolders

Named:

* If there is an icon file (.ico) with the same name as the file(s) you are registering then this image will be associated with the device.

To perform an installation test on the file(s), click Next

Figure 15: 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”.

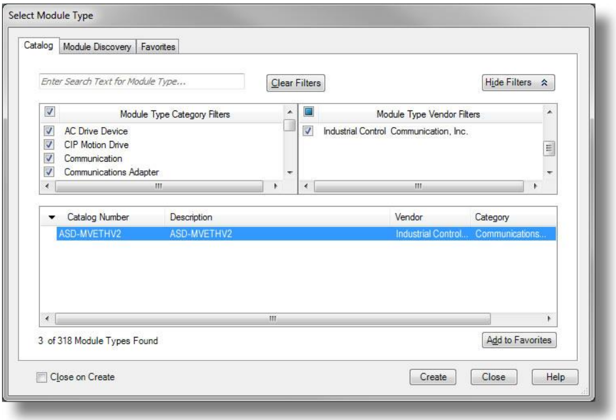
7) The task summary will list the interface card as the device to register. Click “Next”.

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 16.



☑ Module Type Category Filters
☑ AC Drive Device
☑ CIP Motion Drive
☑ Communication
☑ Communications Adapter

Module Type Vendor Filters
☑ Industrial Control Communication, Inc.

Catalog Number	Description	Vendor	Category
ASD-MVETHV2	ASD-MVETHV2	Industrial Control...	Communications...

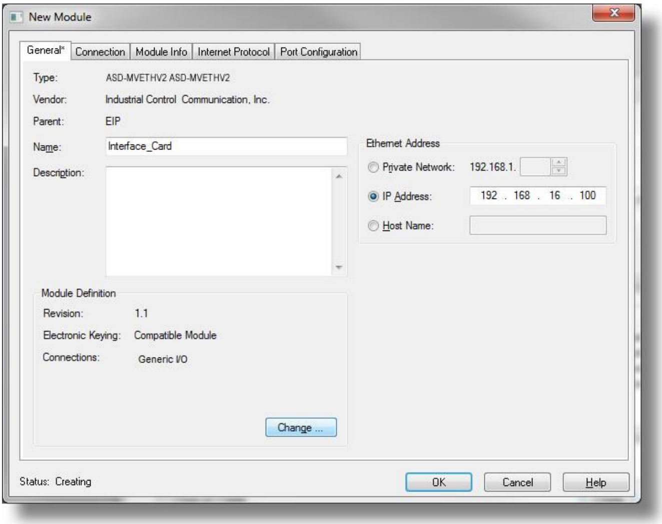
3 of 318 Module Types Found

□ Close on Create

Figure 16: 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 17.



Type: ASD-MVETHV2 ASD-MVETHV2
Vendor: Industrial Control Communication, Inc.
Parent: EIP
Name: Interface_Card
Description:

Ethernet Address
○ Private Network: 192.168.1.
● IP Address: 192 . 168 . 16 . 100
○ Host Name:

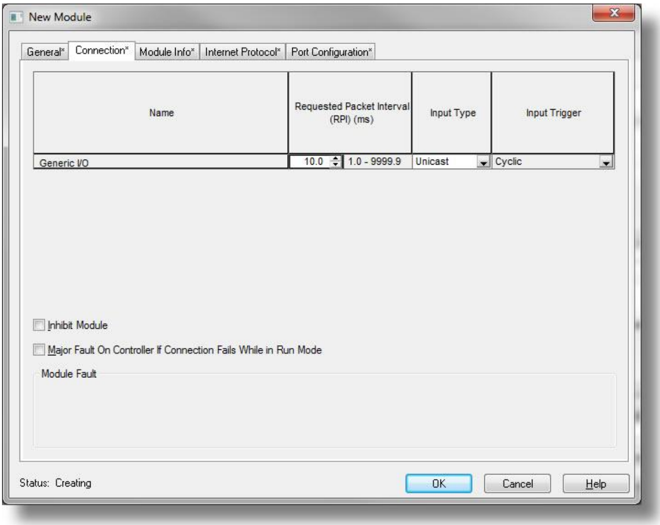
Module Definition
Revision: 1.1
Electronic Keying: Compatible Module
Connections: Generic I/O

Status: Creating

Figure 17: 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 18.



New Module

Name	Requested Packet Interval (RPI) (ms)		Input Type	Input Trigger
Generic I/O	10.0	1.0 - 9999.9	Unicast	Cyclic

□ Inhibit Module

□ Major Fault On Controller If Connection Fails While in Run Mode

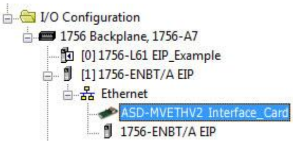
Module Fault

Status: Creating

Figure 18: 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 19.



```
I/O Configuration
  1756 Backplane, 1756-A7
    [0] 1756-L61 EIP_Example
    [1] 1756-ENBT/A EIP
      Ethernet
        ASD-MVETHV2_Interface_Card
        1756-ENBT/A EIP
```

Figure 19: AOP Interface Card I/O Configuration

15) Continue with the AOP Generic I/O Messaging example in section 10.2.10.1.

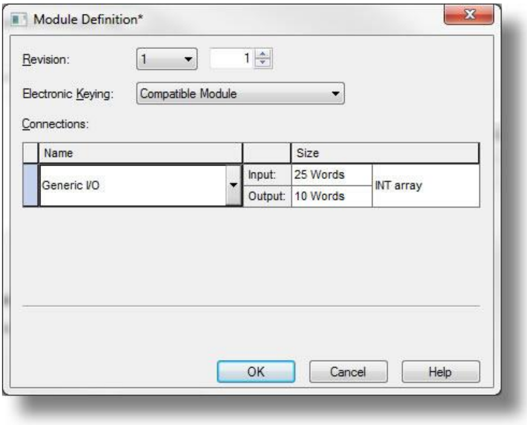
10.2.10.1 ControlLogix Example: EDS Add-On Profile (AOP) Generic I/O Messaging

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

1) Complete all steps in section 10.2.10.

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 20.



Revision: 1 1
Electronic Keying: Compatible Module
Connections:

Name		Size	
Generic I/O	Input:	25 Words	INT array
	Output:	10 Words	

Figure 20: AOP Generic I/O Module Definition

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

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

<u>Size:</u> Because all inverter data is stored as 16-bit parameters, change the data type to “INT array”.

<u>Input:</u> 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 10.2.4. The Input Size must be set to the number of 16-bit parameters 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 25 relevant parameters (SCAN_WR01_AS to SCAN_WR25_AS). We therefore set the Input Size to 25 Words.

<u>Output:</u> 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 10.2.4. The Output Size must be set to the number of 16-bit parameters 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 10 relevant parameters (SCAN_RCV01_AS to SCAN_RCV10_AS). We therefore set the Output Size to 10 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.

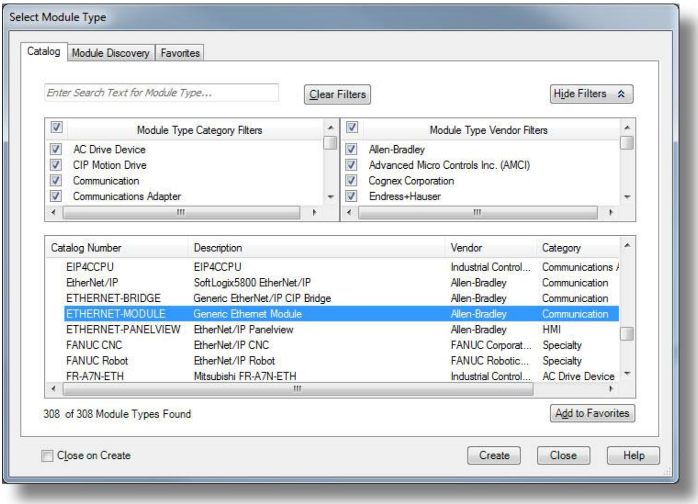
10.2.11 ControlLogix Example: I/O Messaging

This section will demonstrate how to setup and use an EtherNet/IP I/O connection via vendor-specific assembly instances 100 & 150. Do not complete this example if you are using EDS AOP, in which case, please refer to the AOP examples. EtherNet/IP I/O messaging allows the inverter’s parameter 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 21.



Select Module Type

Catalog | Module Discovery | Favorites

Enter Search Text for Module Type...

Clear Filters

Hide Filters

☑ Module Type Category Filters
☑ AC Drive Device
☑ CIP Motion Drive
☑ Communication
☑ Communications Adapter

☑ Module Type Vendor Filters
☑ Allen-Bradley
☑ Advanced Micro Controls Inc. (AMCI)
☑ Cognex Corporation
☑ Endress+Hauser

Catalog Number	Description	Vendor	Category
EIP4CCPU	EIP4CCPU	Industrial Control...	Communications /
EtherNet/IP	SoftLogix5800 EtherNet/IP	Allen-Bradley	Communication
ETHERNET-BRIDGE	Generic EtherNet/IP CIP Bridge	Allen-Bradley	Communication
ETHERNET-MODULE	Generic Ethernet Module	Allen-Bradley	Communication
ETHERNET-PANELVIEW	EtherNet/IP Panelview	Allen-Bradley	HMI
FANUC CNC	EtherNet/IP CNC	FANUC Corporat...	Specialty
FANUC Robot	EtherNet/IP Robot	FANUC Robotic...	Specialty
FR-A7N-ETH	Mitsubishi FR-A7N-ETH	Industrial Control...	AC Drive Device

308 of 308 Module Types Found

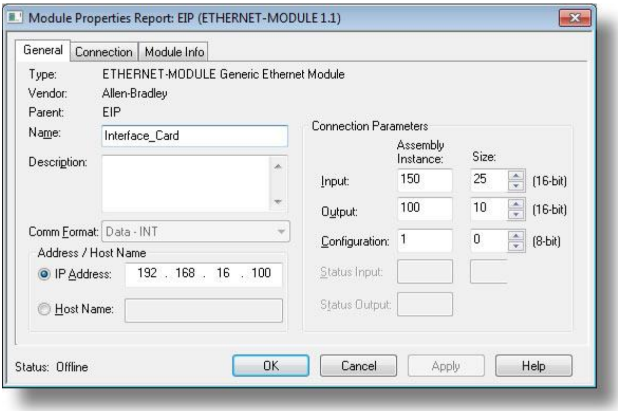
Add to Favorites

☐ Close on Create

Create | Close | Help

Figure 21: Adding a New Generic Ethernet Module

4) The module properties dialog box will open (refer to Figure 22). 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 parameters, change the “Comm Format” selection to “Data-INT”. Enter the IP address of the targeted interface card.



Module Properties Report: EIP (ETHERNET-MODULE 1.1)

General | Connection | Module Info

Type: ETHERNET-MODULE Generic Ethernet Module

Vendor: Allen-Bradley

Parent: EIP

Name: Interface_Card

Description:

Comm Format: Data - INT

Address / Host Name

● IP Address: 192 . 168 . 16 . 100

○ Host Name:

Connection Parameters

	Assembly Instance:	Size:	
Input:	150	25	(16-bit)
Output:	100	10	(16-bit)
Configuration:	1	0	(8-bit)
Status Input:			
Status Output:			

Status: Offline

OK | Cancel | Apply | Help

Figure 22: Interface Card Module Properties

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

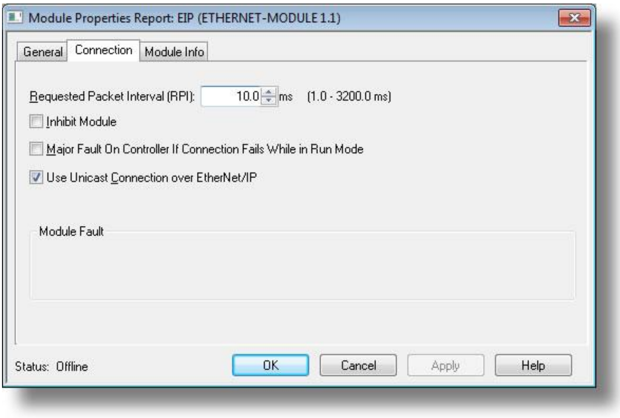
Input: The Input Assembly is the collection of monitor data that is produced by the interface card and is received as an input to the PLC. Its structure is defined by the Produced Data Configuration as described in section 10.2.4. The Input Assembly Instance must be set to 150 when connecting to the generic I/O assembly instances, and the size must be set to the number of 16-bit parameters 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 25 relevant parameters (SCAN_WR01_AS to SCAN_WR25_AS). We therefore set the Input Size to 25.

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 10.2.4. The Output Assembly Instance must be set to 100 when connecting to the generic I/O assembly instances, and the size must be set to the number of 16-bit parameters 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 10 relevant parameters (SCAN_RCV01_AS to SCAN_RCV10_AS). We therefore set the Output Size to 10.

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 23.



Module Properties Report: EIP (ETHERNET-MODULE 1.1)

Requested Packet Interval (RPI): 10.0 ms (1.0 - 3200.0 ms)

□ Inhibit Module

□ Major Fault On Controller If Connection Fails While in Run Mode

☑ Use Unicast Connection over EtherNet/IP

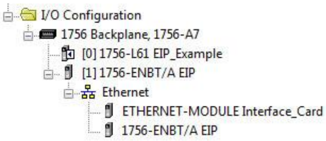
Module Fault

Status: Offline

Figure 23: 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 24.

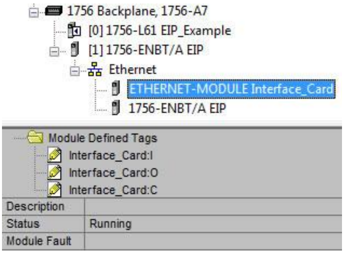


```
I/O Configuration
  1756 Backplane, 1756-A7
    [0] 1756-L61 EIP_Example
    [1] 1756-ENBT/A EIP
      Ethernet
        ETHERNET-MODULE Interface_Card
        1756-ENBT/A EIP
```

Figure 24: I/O Configuration Tree

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 25. Also confirm that the interface card's "Network Status" LED should be solid green, indicating an "online/connected" state.

8) By double-clicking "Controller Tags" in the project tree, it is possible to view the newly-added tags. Refer to Figure 26. 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.

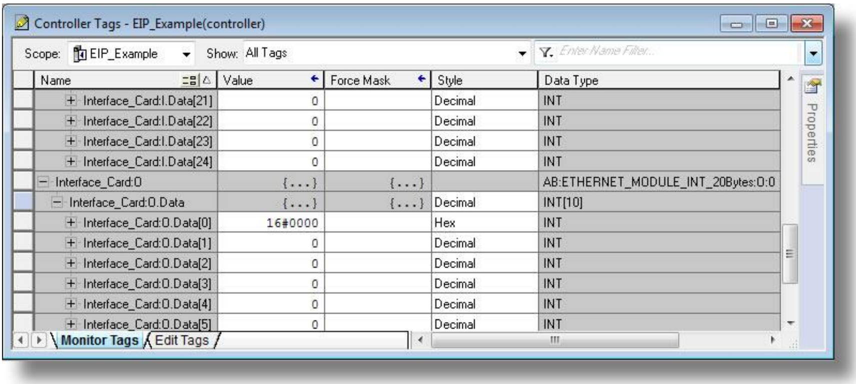


```
1756 Backplane, 1756-A7
  [0] 1756-L61 EIP_Example
  [1] 1756-ENBT/A EIP
    Ethernet
      ETHERNET-MODULE Interface_Card
      1756-ENBT/A EIP
Module Defined Tags
  Interface_Card:I
  Interface_Card:O
  Interface_Card:C
```

Description	
Status	Running
Module Fault	

Figure 25: Online Module Status

9) By double-clicking “Controller Tags” in the project tree, it is possible to view the newly-added tags. Refer to Figure 26. 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.



Controller Tags - EIP_Example(controller)

Scope: EIP_Example Show: All Tags

Name	Value	Force Mask	Style	Data Type
Interface_Card:I.Data[21]	0		Decimal	INT
Interface_Card:I.Data[22]	0		Decimal	INT
Interface_Card:I.Data[23]	0		Decimal	INT
Interface_Card:I.Data[24]	0		Decimal	INT
Interface_Card:O	{...}	{...}		AB:ETHERNET_MODULE_INT_20Bytes:O:0
Interface_Card:O.Data	{...}	{...}	Decimal	INT[10]
Interface_Card:O.Data[0]	16#0000		Hex	INT
Interface_Card:O.Data[1]	0		Decimal	INT
Interface_Card:O.Data[2]	0		Decimal	INT
Interface_Card:O.Data[3]	0		Decimal	INT
Interface_Card:O.Data[4]	0		Decimal	INT
Interface_Card:O.Data[5]	0		Decimal	INT

Figure 26: Controller Tags for I/O Access

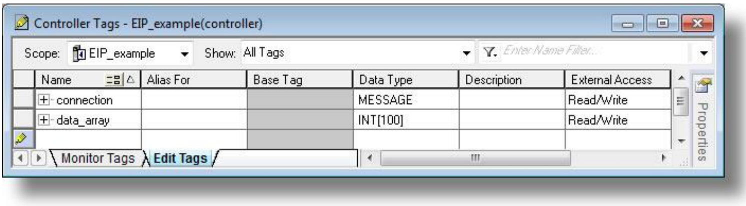
10.2.12 ControlLogix Example: Read Parameters

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

1) Create new Tags.

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

b) The “Controller Tags” window appears. Refer to Figure 27.



Controller Tags - EIP_example(controller)

Scope: EIP_example Show: All Tags

Name	Alias For	Base Tag	Data Type	Description	External Access
connection			MESSAGE		Read/Write
data_array			INT[100]		Read/Write

Figure 27: Create New Tags

c) Select the “Edit Tags” tab at the bottom.

d) Create a new tag by entering “connection” in the first blank Name field, and change its Data Type to “MESSAGE”. This tag will contain configuration information for the MSG instruction.

e) Select the “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[100]” in the Data Type field. This tag is an array of INTs that will be able to hold up to 100 16-bit parameters from the inverter. Always make sure that the destination tag size is large enough to hold all elements to be read.

2) **Add a MSG instruction to the main program.**

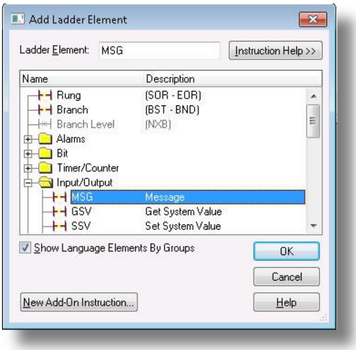
a) Double-click “MainRoutine” under Tasks …MainTask …MainProgram in the controller organizer view.

b) Right click on the first ladder logic rung in the MainRoutine window and select “Add Ladder Element...”

c) The “Add Ladder Element” window appears.

d) Select the “MSG” instruction in the Input/Output folder. Refer to Figure 28.

e) Click OK.



Add Ladder Element

Ladder Element: MSG

Name	Description
Rung	(SOR - EOR)
Branch	(BST - BND)
Branch Level	(NXB)
Alarms	
Bit	
Timer/Counter	
Input/Output	
MSG	Message
GSV	Get System Value
SSV	Set System Value

☑ Show Language Elements By Groups

Figure 28: Adding a MSG Instruction

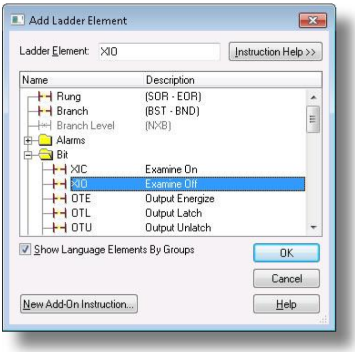
3) **Add an XIO element to the main program.**

a) Right click on the ladder logic rung containing the MSG instruction in the MainRoutine window and select “Add Ladder Element...” again.

b) The “Add Ladder Element” window appears.

c) Select the “XIO” element in the Bit folder. Refer to Figure 29.

d) Click OK.



Add Ladder Element

Ladder Element: XIO

Name	Description
Rung	(SOR - EOR)
Branch	(BST - BND)
Branch Level	(NXB)
Alarms	
Bit	
XIC	Examine On
XIO	Examine Off
OTE	Output Energize
OTL	Output Latch
OTU	Output Unlatch

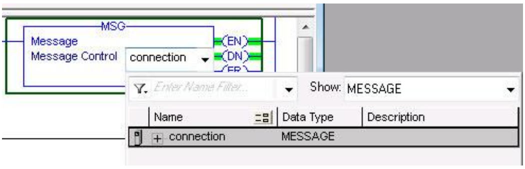
☑ Show Language Elements By Groups

Figure 29: Adding an XIO Element

4) **Configure the MSG instruction.**

a) Edit the “Message Control” field on the MSG instruction to use the previously-created “connection” tag. Refer to Figure 30.

b) Click the message configuration button (“…”) in the MSG instruction. The “Message Configuration” window will open. Refer to Figure 31.

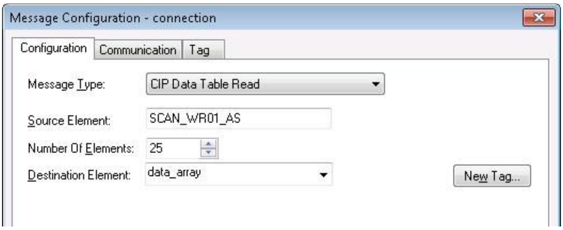


MSG
Message
Message Control: connection

Show: MESSAGE

Name	Data Type	Description
connection	MESSAGE	

Figure 30: MSG Instruction Tag Assignment



Message Configuration - connection

Message Type: CIP Data Table Read

Source Element: SCAN_WR01_AS

Number Of Elements: 25

Destination Element: data_array

Figure 31: MSG Instruction Configuration

c) "Configuration" tab settings:

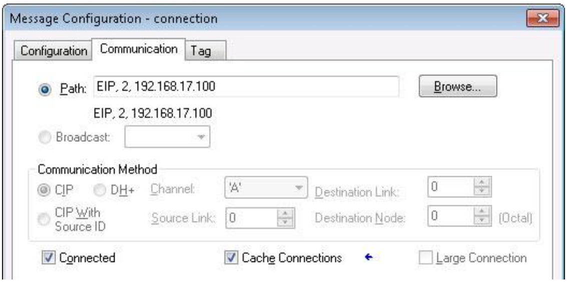
i) Change the "Message Type" to "CIP Data Table Read".

ii) In the "Source Element" field, enter the read tag you wish to access (refer to section 10.2.6.) In this example, we will be reading a total of 25 parameters beginning at parameter SCAN_WR01_AS.

iii) Enter the Number Of Elements to read. In this example, we will read 25 parameters.

iv) For the Destination Element, select "data_array".

d) "Communication" tab settings (refer to Figure 32):



Message Configuration - connection

Configuration | Communication | Tag

● Path: EIP, 2, 192.168.17.100 Browse...

EIP, 2, 192.168.17.100

○ Broadcast:

Communication Method

● CIP ○ DH+ Channel: 'A' Destination Link: 0

○ CIP With Source ID Source Link: 0 Destination Node: 0 (Octal)

☑ Connected ☑ Cache Connections ☐ Large Connection

Figure 32: Setting the Communication Path

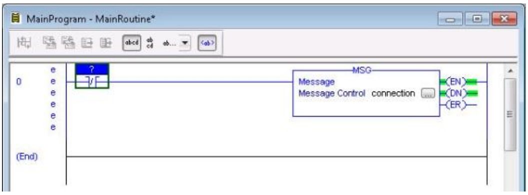
i) Enter the Path to the interface card. A typical path is formatted as "*Local_ENB,2,target_IP_address*", where:

- *Local_ENB* is the name of the 1756-ENBx module in the local chassis (we named ours "EIP" in section 10.2.9),
- *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,2,192.168.17.100".

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 33.



MainProgram - MainRoutine*

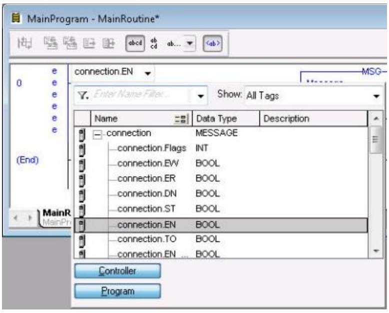
0 ? MSG Message Message Control connection EN DN ER

(End)

Figure 33: 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 34. 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.



Name	Data Type	Description
connection	MESSAGE	
connection.Flags	INT	
connection.EW	BOOL	
connection.ER	BOOL	
connection.DN	BOOL	
connection.ST	BOOL	
connection.EN	BOOL	
connection.TO	BOOL	
connection.EN ...	BOOL	

Figure 34: Configure XIO Element

6) **The program is now complete. Refer to Figure 35.**

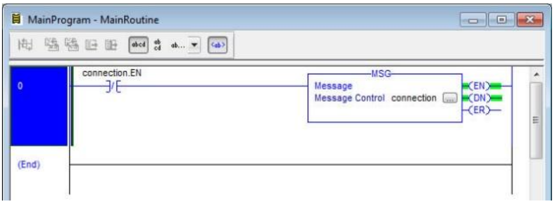


Figure 35: Complete Program

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

a) To view the values of the parameters being read from the interface card, double-click “Controller Tags” in the controller organizer view.

b) Select the “Monitor Tags” tab and expand the data_array tag.

c) 25 parameter values starting at parameter SCAN_WR01_AS are being continuously read from the interface card and placed in the 25 sequential offsets of data_array starting at the 0th offset (data_array[0]).

10.2.13 ControlLogix Example: Reading and Writing Multiple 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 36 shows an example of four MSG instructions, two for reading and two for writing. It is evident from this logic that “rd_connection_1” and “rd_connection2” are the two independent message controller tags created for the read instructions. Note the addition of the en_wr_xx 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_wr_xx 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.

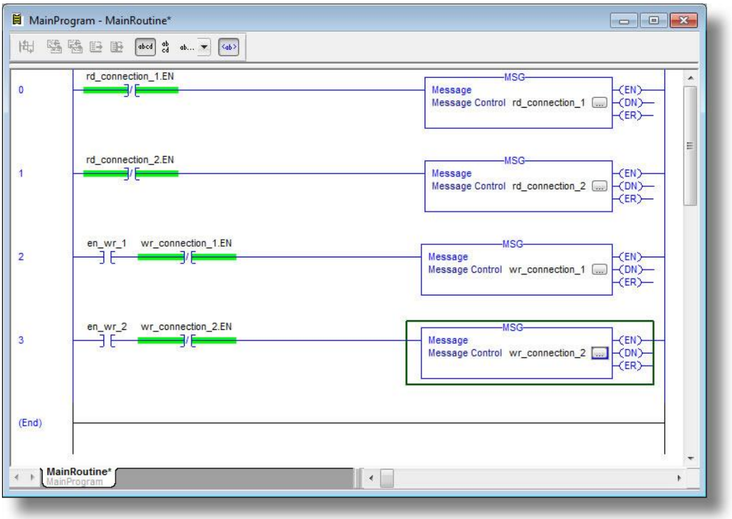
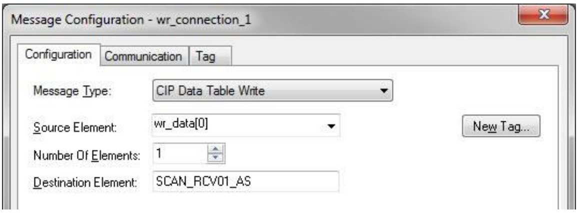


Figure 36: Reading and Writing via MSG Instructions

Figure 37 shows the configuration details of the example wr_connection_xx MSG instruction. Note that the chosen “Message Type” is “CIP Data Table Write”, and that this instruction will only be writing to one inverter parameter. The value of the controller tag specified in the Source Element will be written to the inverter parameter specified in the Destination Element.



Message Type: CIP Data Table Write

Source Element: wr_data[0]

Number Of Elements: 1

Destination Element: SCAN_RCV01_AS

Figure 37: MSG Configuration for Writing

Note that when writing data via explicit messaging, use caution to ensure that the commanded parameters 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 parameters. In other words, if the I/O messaging example procedure detailed in section 10.2.10 or 10.2.11 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.

10.3 Allen Bradley CSP (PCCC)

10.3.1 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 10.2.3.

10.3.2 Explicit Messaging Via Read/Write Services

Register (parameter) 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 21. 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 10.3.3.

Table 21: 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

10.3.3 Inverter Parameter File Number Offset Format

Use integer file number 10 (N10). The offset/element is the register number of the parameter. Refer to section 5.1 for the Parameter-to-Register mapping. Table 22 provides some examples of various combinations of file/section numbers and offsets/elements which can be used to access inverter parameters.

Table 22: CSP Target Register Examples

Parameter	Target Register	File/Section Number	Offset/Element	Address Format
SCAN_RCV01_AS	1	N10	1	N10:1
SCAN_RCV10_AS	10	N10	10	N10:10
SCAN_WR01_AS	11	N10	11	N10:11
SCAN_WR25_AS	35	N10	35	N10:35

In addition to providing access to the inverter parameters in their "standard" numerical locations as mentioned above, the parameters can also be accessed in a special "assembly object" type format by targeting integer file N50. What this means is that when N50 is targeted for reading, what is actually returned by the interface card is the user-defined parameter data as ordered by the EtherNet/IP produced data word configuration (refer to section 10.2.4). Similarly, when N50 is targeted for writing, the written data is disseminated to the inverter's parameters 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 parameters can be conveniently provided by performing only one read and/or write instruction targeting file N50.

Because both the EtherNet/IP consumed and produced data word configurations are comprised of up to 25 parameter definitions, the targeted "offset/element" must be within the range of 0 to 24 inclusive. Refer to Table 23 for some examples of N50 accesses.

Table 23: Examples of EtherNet/IP-Style Bulk Access via File N50

File/Section Number	Offset/Element	Address Format	Start Target Parameter of Configuration Array	Max Number of Accessible Elements
N50	0	N50:0	1st	32
N50	:	:	:	:
N50	15	N50:15	16th	16
N50	:	:	:	:
N50	24	N50:24	24th	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”.

10.3.4 SLC-5/05 Example: Read Parameters

This example program will show how to continuously read parameters 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.

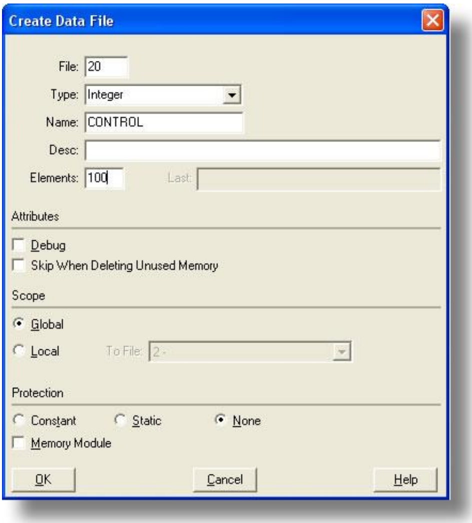
1) Run RSLogix 500, and create a new configuration.

2) Create a control and a data file.

a) Right click Data Files and select New… The “Create Data File” dialog box appears (refer to Figure 38).

b) To create a control file, enter a file number (e.g. 20), set the type to “Integer”, enter a descriptive name (e.g. “CONTROL”), and enter a number of elements (e.g. 100). Click OK to create the file. The control file is used to store configuration information pertaining to the functionality of the MSG instruction which will perform the data read.

c) Follow the same procedure to create a data file. This file will be used to store the incoming data read from the interface card. Enter a file number (e.g. 18), set the type to “Integer”, enter a descriptive name (e.g. “DATA”), and enter a number of elements (e.g. 200). Refer to Figure 39. Click OK to create the file.



Create Data File

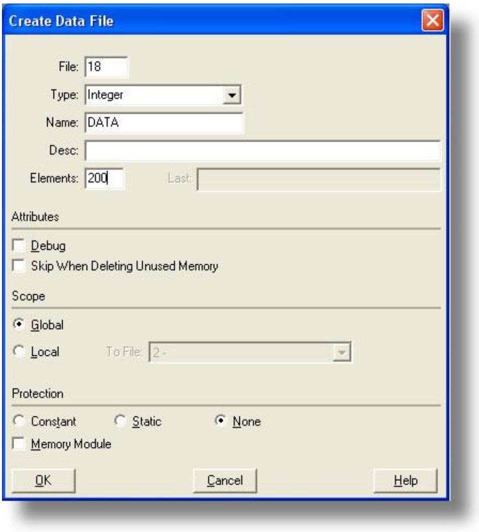
File: 20
Type: Integer
Name: CONTROL
Desc:
Elements: 100 Last:

Attributes
□ Debug
□ Skip When Deleting Unused Memory

Scope
● Global
○ Local To File: 2 -

Protection
○ Constant ○ Static ● None
□ Memory Module

Figure 38: Creating a Control File



Create Data File

File: 18

Type: Integer

Name: DATA

Desc:

Elements: 200 Last:

Attributes

☐ Debug

☐ Skip When Deleting Unused Memory

Scope

● Global

○ Local To File: 2 -

Protection

○ Constant ○ Static ● None

☐ Memory Module

Figure 39: Creating a Data File

3) **Add a MSG instruction to the program.**

a) If not already visible, double-click "LAD2" under Project…Program Files in the controller organizer view to bring up the ladder logic program.

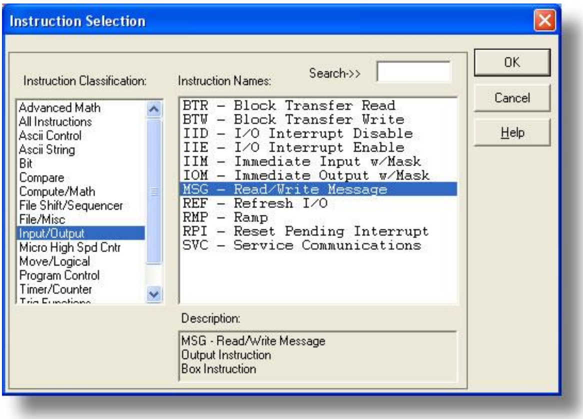
b) Right click on the default rung number on the left-hand side of the LAD2 window and select "Insert Rung".

c) Right click on the rung number of the new editable rung and select "Append Instruction".

d) Select the "MSG" instruction from the "Input/Output" classification, then click OK. Refer to Figure 40.

4) **Add an XIO element to the program.**

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



Instruction Selection

Instruction Classification:

Advanced Math
All Instructions
Ascii Control
Ascii String
Bit
Compare
Compute/Math
File Shift/Sequencer
File/Misc
Input/Output
Micro High Spd Cntr
Move/Logical
Program Control
Timer/Counter
Trig Functions

Instruction Names: Search>>

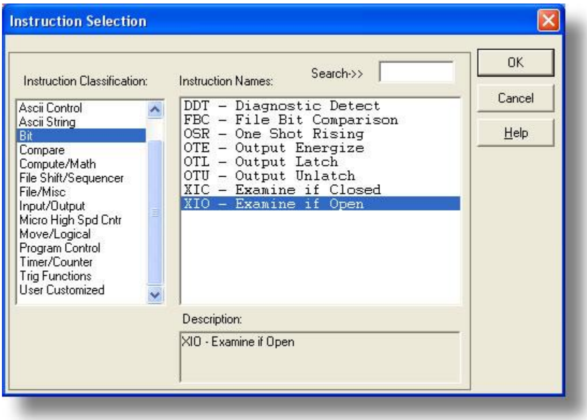
```
BTR - Block Transfer Read
BTW - Block Transfer Write
IID - I/O Interrupt Disable
IIE - I/O Interrupt Enable
IIM - Immediate Input w/Mask
IOM - Immediate Output w/Mask
MSG - Read/Write Message
REF - Refresh I/O
RMP - Ramp
RPI - Reset Pending Interrupt
SVC - Service Communications
```

Description:

MSG - Read/Write Message
Output Instruction
Box Instruction

Figure 40: MSG Instruction Selection

b) Select the “XIO” instruction from the “Bit” classification, then click OK. Refer to Figure 41.



Instruction Classification:

- Ascii Control
- Ascii String
- Bit
- Compare
- Compute/Math
- File Shift/Sequencer
- File/Misc
- Input/Output
- Micro High Spd Cntr
- Move/Logical
- Program Control
- Timer/Counter
- Trig Functions
- User Customized

Instruction Names:

```
DDT - Diagnostic Detect
FBC - File Bit Comparison
OSR - One Shot Rising
OTE - Output Energize
OTL - Output Latch
OTU - Output Unlatch
XIC - Examine if Closed
XIO - Examine if Open
```

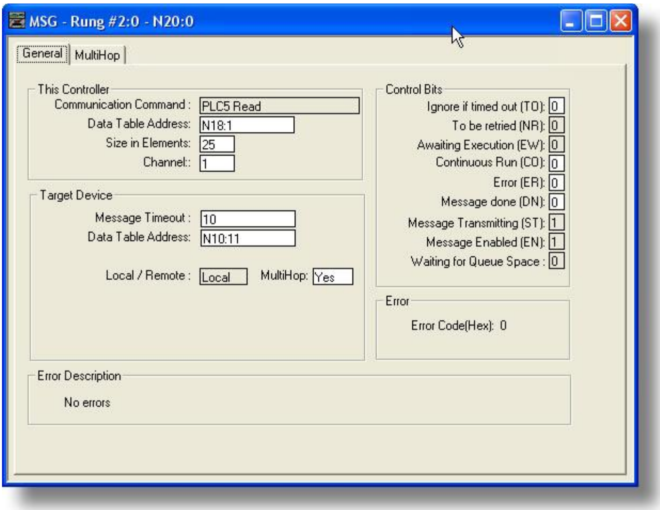
Description: XIO - Examine if Open

Figure 41: 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 42.



This Controller
Communication Command : PLC5 Read
Data Table Address: N18:1
Size in Elements: 25
Channel:: 1

Target Device
Message Timeout : 10
Data Table Address: N10:11
Local / Remote : Local MultiHop: Yes

Control Bits
Ignore if timed out (TO): 0
To be retried (NR): 0
Awaiting Execution (EW): 0
Continuous Run (CO): 0
Error (ER): 0
Message done (DN): 0
Message Transmitting (ST): 1
Message Enabled (EN): 1
Waiting for Queue Space : 0

Error
Error Code(Hex): 0

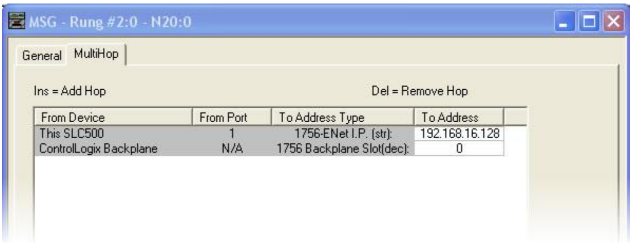
Error Description
No errors

Figure 42: MSG Configuration, "General" Tab

c) In this example, we will be reading a total of 25 parameters beginning at N10:11 (register 11 / parameter SCAN_WR01_AS). 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 N10:11 (starting target register=11) 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 43.

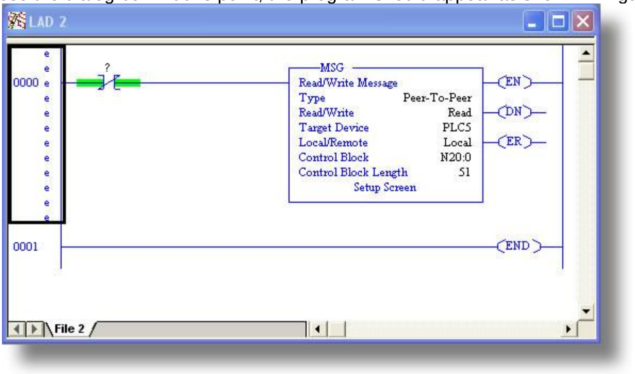


Ins = Add Hop   Del = Remove Hop

From Device	From Port	To Address Type	To Address
This SLC500	1	1756-ENet I.P. (str):	192.168.16.128
ControlLogix Backplane	N/A	1756 Backplane Slot(dec):	0

Figure 43: MSG Configuration, "MultiHop" Tab

f) Close the dialog box. At this point, the program should appear as shown in Figure 44.



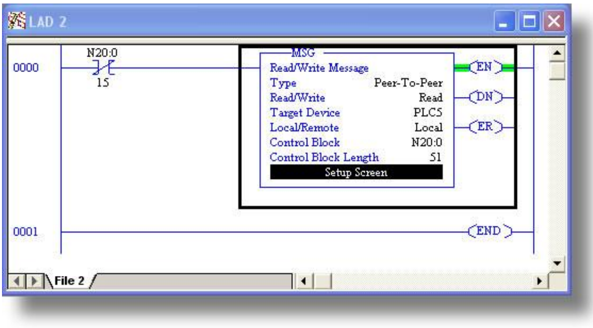
MSG
Read/Write Message
Type Peer-To-Peer
Read/Write Read
Target Device PLC5
Local/Remote Local
Control Block N20:0
Control Block Length 51
Setup Screen

Figure 44: PLC Program after MSG Instruction Configuration

6) Assign a tag to the XIO element.

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 45.

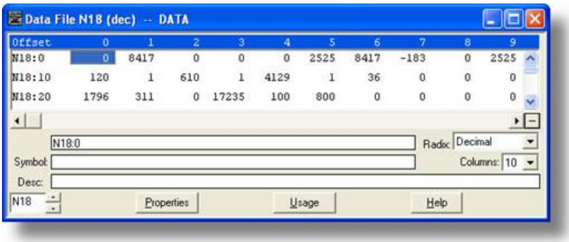


MSG
Read/Write Message
Type Peer-To-Peer
Read/Write Read
Target Device PLC5
Local/Remote Local
Control Block N20:0
Control Block Length 51
Setup Screen

Figure 45: Completed PLC Program

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

To view the parameters being read from the interface card, double-click the data file N18 under "Data Files" in the controller organizer view. 25 parameter values starting at register #11 are being continuously read from the interface card and placed in the 25 sequential offsets of N18 starting at N18:1. Refer to Figure 46.



Data File N18 (dec) -- DATA

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

N18:0

Radix: Decimal

Symbol:

Columns: 10

Desc:

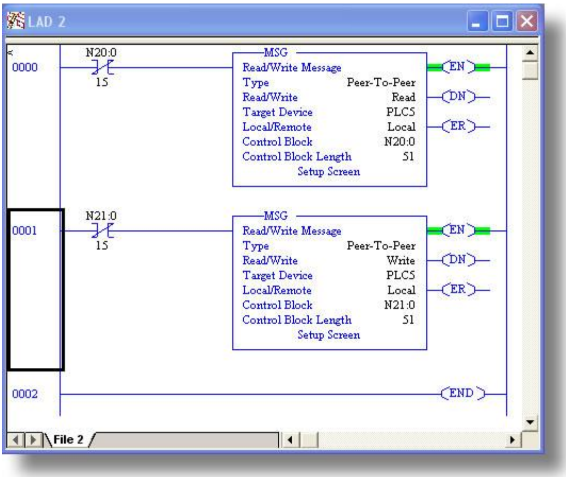
N18

Figure 46: Monitoring the Data Being Read from the Inverter

10.3.5 SLC-5/05 Example: Reading and Writing Multiple 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 control file.

Figure 47 shows an example of two MSG instructions, one for reading and one for writing. It is evident from this logic that N20 and N21 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.

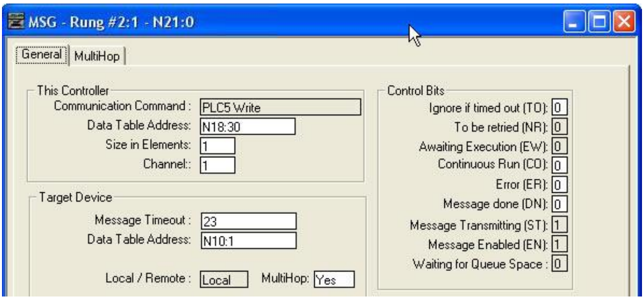


MSG
Read/Write Message
Type Peer-To-Peer
Read/Write Read
Target Device PLC5
Local/Remote Local
Control Block N20:0
Control Block Length 51
Setup Screen

MSG
Read/Write Message
Type Peer-To-Peer
Read/Write Write
Target Device PLC5
Local/Remote Local
Control Block N21:0
Control Block Length 51
Setup Screen

Figure 47: Reading and Writing via MSG Instructions

Figure 48 shows the configuration details of the "write" MSG instruction. Note that this instruction will only be writing to one inverter parameter: namely, register 1 (parameter SCAN_RCV01_AS). The source Data Table Address in this case is N18:30.



This Controller

Communication Command : PLC5 Write

Data Table Address: N18:30

Size in Elements: 1

Channel:: 1

Target Device

Message Timeout : 23

Data Table Address: N10:1

Local / Remote : Local

MultiHop: Yes

Control Bits

Ignore if timed out (TO): 0

To be retried (NR): 0

Awaiting Execution (EW): 0

Continuous Run (CO): 0

Error (ER): 0

Message done (DN): 0

Message Transmitting (ST): 1

Message Enabled (EN): 1

Waiting for Queue Space : 0

Figure 48: MSG Configuration for Writing

10.4 BACnet/IP

- 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 7.5.1).

10.4.1 Protocol Implementation Conformance Statement

BACnet Protocol

Date:	March 13, 2024
Vendor Name:	ICC, Inc.
Product Name:	Ethernet and RS-485 interface
Product Model Number:	PC61910P231/ASD-MVETHV2
Applications Software Version:	V1.1.4
Firmware Revision:	V1.1.4
BACnet Protocol Revision:	2
Product Description:	

Communication board for Toshiba T300MVi, T300MV2, & MTX series medium voltage drives

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 devise.) □ 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
□ ISO 10646 (UCS-2)
□ IBM™/Microsoft™ DBCS
□ ISO 10646 (UCS-4)
□ ISO 8859-1
□ 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 24: 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 25: 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	

Property	Object Type		
	Binary Input	Binary Output	Binary Value
Active Text	R	R	
Inactive Text	R	R	

R – readable using BACnet services
W – readable and writable using BACnet services

Table 26: 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 27: 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
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

10.4.2 *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 28: Analog Input Object Instance Summary

Instance ID	Object Name	Description
AI1	SCAN_WR01_AS	Status data 1
AI2	SCAN_WR02_AS	Status data 2
AI3	SCAN_WR03_AS	Status data 3
AI4	SCAN_WR04_AS	Status data 4
AI5	SCAN_WR05_AS	Status data 5
AI6	SCAN_WR06_AS	Status data 6
AI7	SCAN_WR07_AS	Status data 7
AI8	SCAN_WR08_AS	Status data 8
AI9	SCAN_WR09_AS	Status data 9
AI10	SCAN_WR10_AS	Status data 10
AI11	SCAN_WR11_AS	Status data 11
AI12	SCAN_WR12_AS	Status data 12
AI13	SCAN_WR13_AS	Status data 13
AI14	SCAN_WR14_AS	Status data 14
AI15	SCAN_WR15_AS	Status data 15
AI16	SCAN_WR16_AS	Status data 16
AI17	SCAN_WR17_AS	Status data 17
AI18	SCAN_WR18_AS	Status data 18
AI19	SCAN_WR19_AS	Status data 19
AI20	SCAN_WR20_AS	Status data 20
AI21	SCAN_WR21_AS	Status data 21
AI22	SCAN_WR22_AS	Status data 22
AI23	SCAN_WR23_AS	Status data 23
AI24	SCAN_WR24_AS	Status data 24
AI25	SCAN_WR25_AS	Status data 25

Table 29: Analog Output Object Instance Summary

Instance ID	Object Name	Description
AO1	SCAN_RCV01_AS	Command data 1
AO2	SCAN_RCV02_AS	Command data 2
AO3	SCAN_RCV03_AS	Command data 3
AO4	SCAN_RCV04_AS	Command data 4
AO5	SCAN_RCV05_AS	Command data 5

Instance ID	Object Name	Description
AO6	SCAN_RCV06_AS	Command data 6
AO7	SCAN_RCV07_AS	Command data 7
AO8	SCAN_RCV08_AS	Command data 8
AO9	SCAN_RCV09_AS	Command data 9
AO10	SCAN_RCV10_AS	Command data 10

10.4.3 Server Settings

In the studio's Project panel, navigate to **ASD-MVETHV2…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.

10.4.4 Node Settings

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

10.4.5 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…0x3FFFFE).

10.4.6 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).

10.4.7 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…0x3FFFFE).

Parameter

The inverter parameter (in section 5) that the BACnet object's present value will access.

Data Type

Fixed to 16-bit 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.)

10.4.8 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…0x3FFFFE).

Parameter

The inverter parameter (in section 5) that the BACnet object’s present value will access.

Data Type

Fixed to 16-bit 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.

10.4.9 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…0x3FFFFE).

Parameter

The inverter parameter (in section 5) that the BACnet object's present value will access.

Data Type

Fixed to 16-bit 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.

10.4.10 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…0x3FFFFE).

Parameter

The inverter parameter (in section 5) that the BACnet object's present value will access.

Bitmask

Specifies which bit(s) in the 16-bit value designated by the "Parameter" that the binary object will map to. This mechanism allows up to 16 binary objects to be simultaneously assigned to one parameter (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 parameter.

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 parameter at the bit location(s) indicated in the bitmask. If all of the bit locations at the designated parameter 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 parameter) 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.

10.4.11 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).

Parameter

The inverter parameter (in section 5) that the BACnet object's present value will access.

Bitmask

Specifies which bit(s) in the 16-bit value designated by the "Parameter" that the binary object will map to. This mechanism allows up to 16 binary objects to be simultaneously assigned to one parameter (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 parameter.

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 parameter 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 parameter 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 parameter at the bit location(s) indicated in the bitmask. If all of the bit locations at the designated parameter 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 parameter) 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.

10.4.12 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…0x3FFFFE).

Parameter

The inverter parameter (in section 5) that the BACnet object's present value will access.

Bitmask

Specifies which bit(s) in the 16-bit value designated by the "Parameter" that the binary object will map to. This mechanism allows up to 16 binary objects to be simultaneously assigned to one parameter (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 parameter.

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 parameter 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 parameter 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 parameter at the bit location(s) indicated in the bitmask. If all of the bit locations at the designated parameter 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.

10.4.13 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).

Parameter

The inverter parameter (in section 5) 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.

10.4.14 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).

Parameter

The inverter parameter (in section 5) 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.

10.4.15 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…0x3FFFFE).

Parameter

The inverter parameter (in section 5) that the BACnet object’s present value will access.

Data Type

Fixed at 16-Bit Unsigned.

<u>Automatic Number of States</u>

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

<u>Number of States</u>

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.

<u>Offset by One</u>

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.

<u>The effect of the "Offset by One" field when writing:</u> 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.

<u>The effect of the "Offset by One" field when reading:</u> 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.

<u>Relinquish Default</u>

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.

10.5 PROFINET IO

10.5.1 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 obtained from the product web page on the internet.
- Supports several user-configurable I/O modules with up to 25 input words and 10 output words.
- 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 perform a timeout action. 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.

10.5.2 Device Settings

In the studio's **Project** panel, navigate to **ASD-MVETHV2…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.

10.5.3 Connection Timeout Options

In the studio's **Project** panel, navigate to **ASD-MVETHV2…Ethernet…PROFINET IO**. The following configuration options will determine the actions to be taken by the interface 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 7.5.1.

10.5.4 Cyclic I/O Produced and Consumed Data Access Settings

In the studio’s **Project** panel, add **ASD-MVETHV2…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: 25 WORDS, OUT: 10 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 10 “command” parameter values can be sent to the inverter, and up to 25 “status” parameter values can be sent back to the controller. Therefore, up to 25 Produced and 10 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 30.

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 parameter 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 parameter. It is recommended to start at word offset 0.

Parameter

The inverter parameter (in section 5) associated with the word offset. For the Produced Data Word object, enter a “status” parameter to be monitored. For the Consumed Data Word object, enter a “command” parameter that can be written.

Data Type

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

Table 30: PROFINET User-Configurable Module I/O Data Format

Consumed Data (PLC to Inverter)		Produced Data (Inverter to PLC)	
Word Offset	**Parameter**	**Word Offset**	**Parameter**
0	SCAN_RCVxx_AS	0	Any
1	SCAN_RCVxx_AS	1	Any
:	SCAN_RCVxx_AS	:	Any
8	SCAN_RCVxx_AS	23	Any
9	SCAN_RCVxx_AS	24	Any

The default I/O configuration is described in Table 31.

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

Table 31: PROFINET Default User-Configurable Module I/O Data Format

Consumed Data (PLC to Inverter)		Produced Data (Inverter to PLC)	
Word Offset	**Parameter**	**Word Offset**	**Parameter**
0	SCAN_RCV01_AS	0	SCAN_WR01_AS
1	SCAN_RCV02_AS	1	SCAN_WR02_AS

Consumed Data (PLC to Inverter)		Produced Data (Inverter to PLC)	
Word Offset	**Parameter**	**Word Offset**	**Parameter**
2	SCAN_RCV03_AS	2	SCAN_WR03_AS
3	SCAN_RCV04_AS	3	SCAN_WR04_AS
4	SCAN_RCV05_AS	4	SCAN_WR05_AS
5	SCAN_RCV06_AS	5	SCAN_WR06_AS
6	SCAN_RCV07_AS	6	SCAN_WR07_AS
7	SCAN_RCV08_AS	7	SCAN_WR08_AS
8	SCAN_RCV09_AS	8	SCAN_WR09_AS
9	SCAN_RCV10_AS	9	SCAN_WR10_AS
10	NA	10	SCAN_WR11_AS
11	NA	11	SCAN_WR12_AS
12	NA	12	SCAN_WR13_AS
13	NA	13	SCAN_WR14_AS
14	NA	14	SCAN_WR15_AS
15	NA	15	SCAN_WR16_AS
16	NA	16	SCAN_WR17_AS
17	NA	17	SCAN_WR18_AS
18	NA	18	SCAN_WR19_AS
19	NA	19	SCAN_WR20_AS
20	NA	20	SCAN_WR21_AS
21	NA	21	SCAN_WR22_AS
22	NA	22	SCAN_WR23_AS
23	NA	23	SCAN_WR24_AS
24	NA	24	SCAN_WR25_AS

10.5.5 Acyclic Data Access

Any inverter parameter 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 parameter register number described in section 5.1. The length is specified according to the number of bytes to access. Since each register corresponds to 2 bytes of data, the length must be an even number.

10.5.6 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 at the vendor of the software.

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

10.5.6.1 Register the GSDML file

Open the TIA Portal project. Navigate to **Options…Manage general station description files (GSD)** as shown in Figure 49.

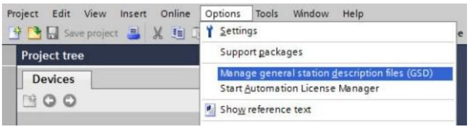
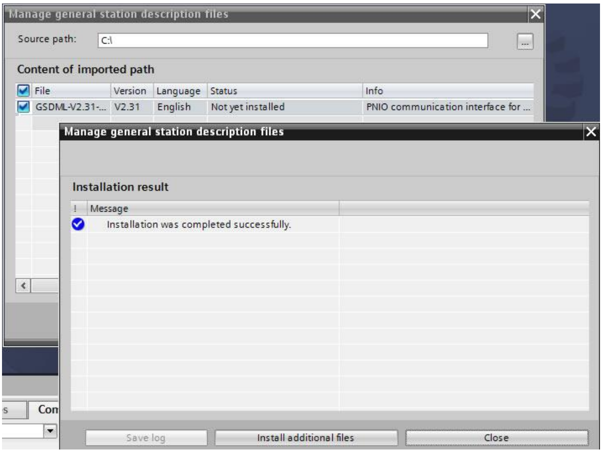


Figure 49: Install GSD File Menu Option

Locate and select the GSDML file and click the **Install** button. Confirm that the installation was completed successfully as shown in Figure 50 and click the **Close** button. It is recommended to use the latest GSDML, which is available via the product web page on the internet.


Manage general station description files

Source path: C:\

Content of imported path

☑ File	Version	Language	Status	Info
☑ GSDML-V2.31-...	V2.31	English	Not yet installed	PNIO communication interface for ...

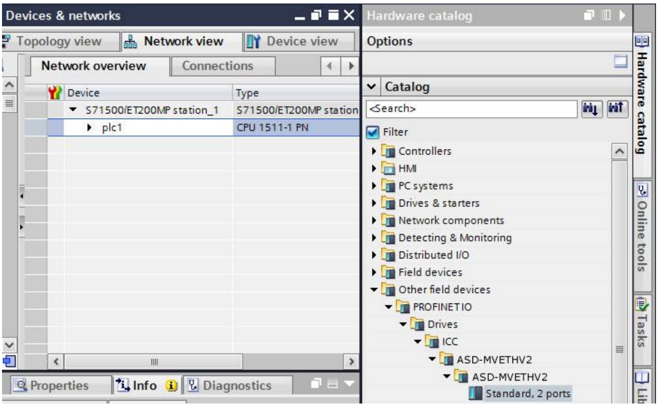
Manage general station description files

Installation result

!	Message
✓	Installation was completed successfully.

Figure 50: 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 51.

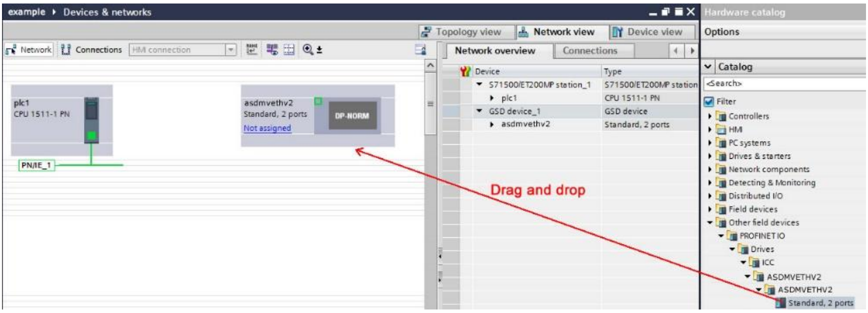

Devices & networks

Device	Type
S71500/ET200MP station_1	S71500/ET200MP station
plc1	CPU 1511-1 PN

Figure 51: Updated Hardware Catalog

10.5.6.2 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 52.



Device	Type
S71500/ET200MP station_1	S71500/ET200MP station
plc1	CPU 1511-1 PN
GSD device_1	GSD device
asdmvethv2	Standard, 2 ports

Figure 52: Add Device to Configuration

10.5.6.3 *Select the IO controller*

On the device, click "Not assigned" and select the appropriate PLC PROFINET interface as shown in Figure 53. This will assign the device to the PROFINET IO system as shown in Figure 54.

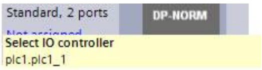


Figure 53: Select IO Controller

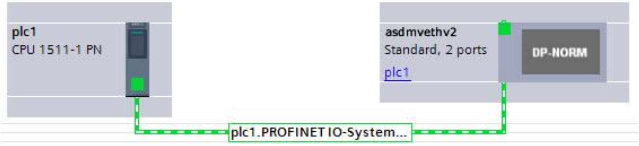
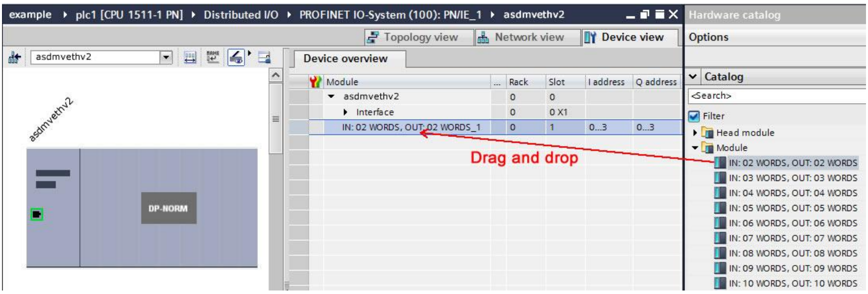


Figure 54: PROFINET IO System

10.5.6.4 *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: 02 WORDS, OUT: 02 WORDS" is selected. Select a module with the appropriate input and output sizes for your specific application. Refer to Figure 55.

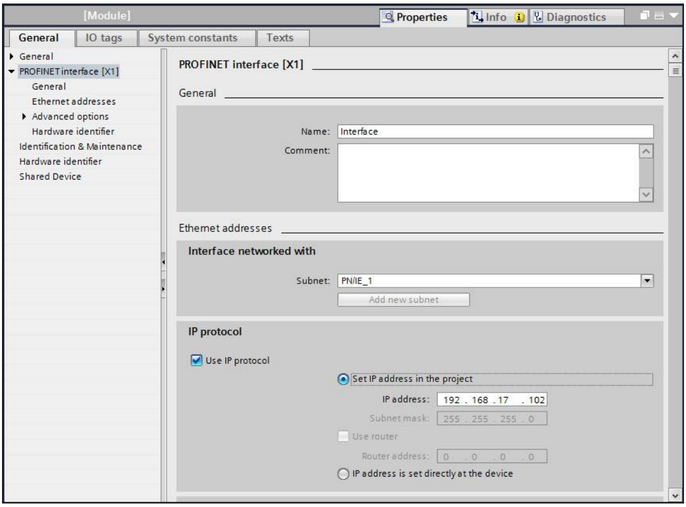

example ▸ plc1 [CPU 1511-1 PN] ▸ Distributed I/O ▸ PROFINET IO-System (100): PN/IE_1 ▸ asdmvethv2
Topology view
Network view
Device view
asdmvethv2
Device overview
asdmvethv2
DP-NORM
Drag and drop
Hardware catalog
Options
Catalog
<Search>
Filter
Head module
Module
IN: 02 WORDS, OUT: 02 WORDS
IN: 03 WORDS, OUT: 03 WORDS
IN: 04 WORDS, OUT: 04 WORDS
IN: 05 WORDS, OUT: 05 WORDS
IN: 06 WORDS, OUT: 06 WORDS
IN: 07 WORDS, OUT: 07 WORDS
IN: 08 WORDS, OUT: 08 WORDS
IN: 09 WORDS, OUT: 09 WORDS
IN: 10 WORDS, OUT: 10 WORDS

Module	...	Rack	Slot	I address	Q address
asdmvethv2		0	0		
Interface		0	0 X1		
IN: 02 WORDS, OUT: 02 WORDS_1		0	1	0...3	0...3

Figure 55: Add IO Module

10.5.6.5 *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 shown in Figure 56.



PROFINET interface [X1]

General

Name: Interface

Comment:

Ethernet addresses

Interface networked with

Subnet: PN/IE_1

IP protocol

☑ Use IP protocol

● Set IP address in the project

IP address: 192 . 168 . 17 . 102

Subnet mask: 255 . 255 . 255 . 0

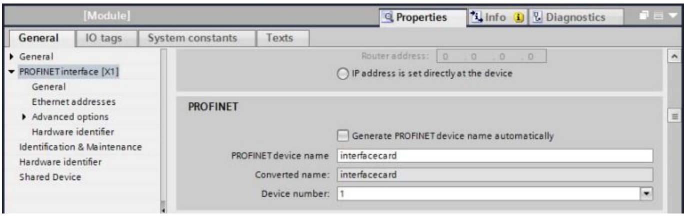
□ Use router

Router address: 0 . 0 . 0 . 0

○ IP address is set directly at the device

Figure 56: Assign Unique Compatible IP Address

Assign a unique **PROFINET device name** as shown in Figure 57.



Router address: 0 . 0 . 0 . 0

○ IP address is set directly at the device

PROFINET

□ Generate PROFINET device name automatically

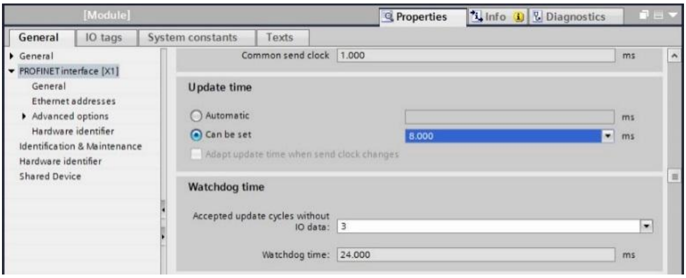
PROFINET device name: interfacecard

Converted name: interfacecard

Device number: 1

Figure 57: Assign Unique Device Name

Set the IO cycle **Update time** and **Watchdog time** as shown in Figure 58.



Common send clock: 1.000 ms

Update time

○ Automatic ms

● Can be set: 8.000 ms

□ Adapt update time when send clock changes

Watchdog time

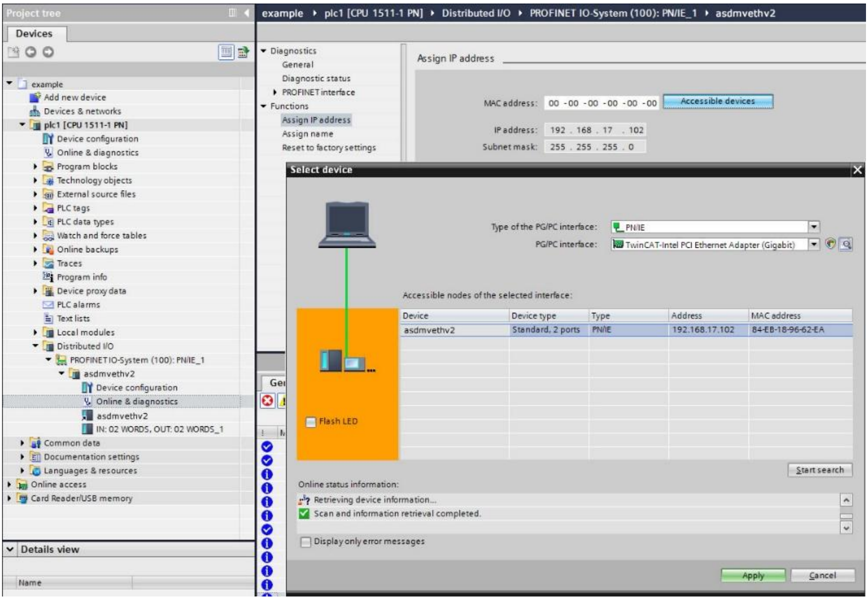
Accepted update cycles without IO data: 3

Watchdog time: 24.000 ms

Figure 58: Set IO Cycle Update Time

10.5.6.6 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 59. Select the device and click the **Apply** button.



MAC address: 00 -00 -00 -00 -00 -00

IP address: 192 . 168 . 17 . 102

Subnet mask: 255 . 255 . 255 . 0

Type of the PG/PC interface: PN/IE

PG/PC interface: TwinCAT-Intel PCI Ethernet Adapter (Gigabit)

Accessible nodes of the selected interface:

Device	Device type	Type	Address	MAC address
asdmvethv2	Standard, 2 ports	PN/IE	192.168.17.102	84-EB-18-96-62-EA

Online status information:

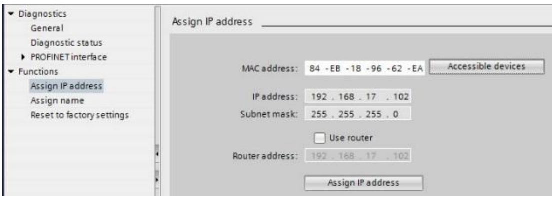
Retrieving device information...

Scan and information retrieval completed.

□ Display only error messages

Figure 59: 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 60.



MAC address: 84 -EB -18 -96 -62 -EA

IP address: 192 . 168 . 17 . 102

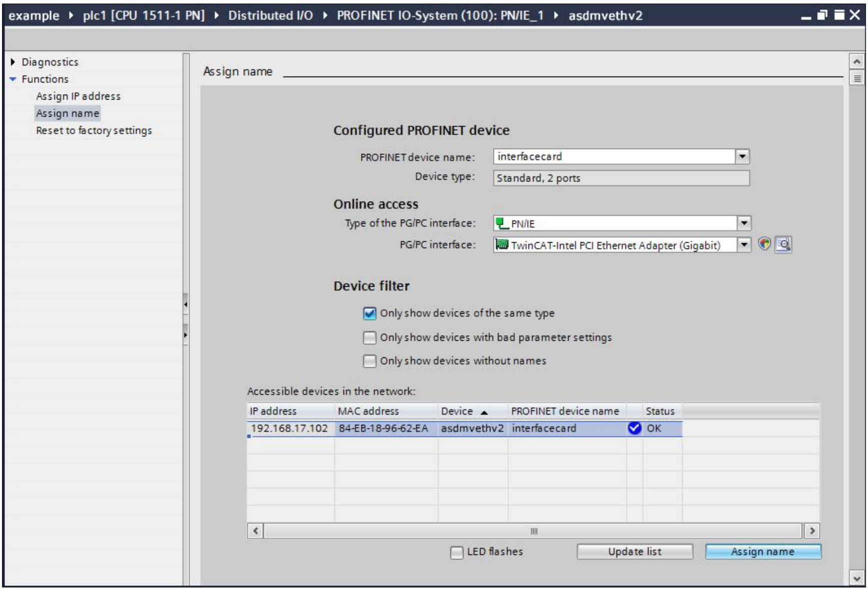
Subnet mask: 255 . 255 . 255 . 0

□ Use router

Router address: 192 . 168 . 17 . 102

Figure 60: 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 61.


example ▸ plc1 [CPU 1511-1 PN] ▸ Distributed I/O ▸ PROFINET IO-System (100): PN/IE_1 ▸ asdmvethv2
▸ Diagnostics
▾ Functions
Assign IP address
Assign name
Reset to factory settings
Assign name

Configured PROFINET device

PROFINET device name: interfacecard

Device type: Standard, 2 ports

Online access

Type of the PG/PC interface: PN/IE

PG/PC interface: TwinCAT-Intel PCI Ethernet Adapter (Gigabit)

Device filter

☑ Only show devices of the same type

☐ Only show devices with bad parameter settings

☐ Only show devices without names

Accessible devices in the network:

IP address	MAC address	Device ▲	PROFINET device name	Status
192.168.17.102	84-EB-18-96-62-EA	asdmvethv2	interfacecard	✓ OK

☐ LED flashes

Figure 61: Assign Name

10.5.6.7 *Save the configuration*

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

10.5.7 *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.

10.5.7.1 *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 62.

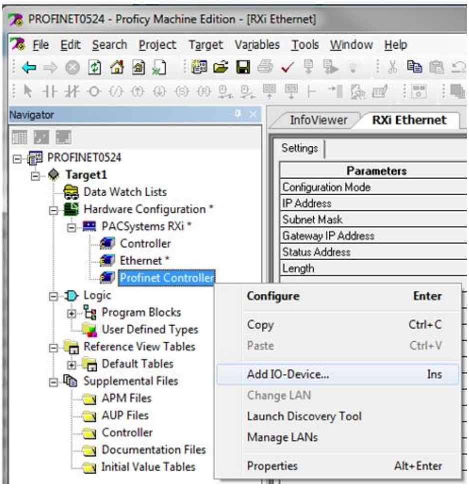


Figure 62: Add IO-Device

Click the **Have GSDML…** button as shown in Figure 63.

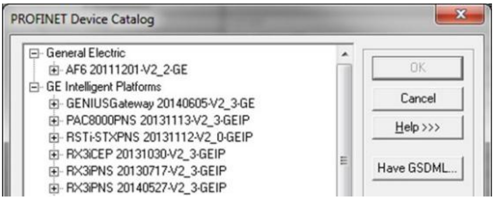
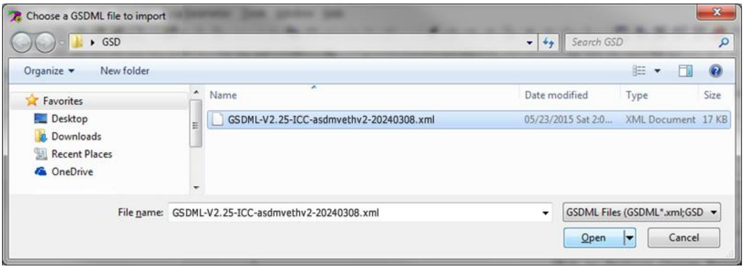

PROFINET Device Catalog
General Electric
AF6 20111201-V2_2-GE
GE Intelligent Platforms
GENIUSGateway 20140605-V2_3-GE
PAC8000PNS 20131113-V2_3-GEIP
RSTi-STXPNS 20131112-V2_0-GEIP
RX3iCEP 20131030-V2_3-GEIP
RX3iPNS 20130717-V2_3-GEIP
RX3iPNS 20140527-V2_3-GEIP
OK
Cancel
Help >>>
Have GSDML...

Figure 63: Have GSDML

Locate and select the GSDML file. Click the **Open** button to register the GSDML as shown in Figure 64. It is recommended to use the latest GSDML, which is available via the product web page on the internet.



Name	Date modified	Type	Size
GSDML-V2.25-ICC-asdmvethv2-20240308.xml	05/23/2015 Sat 2:0...	XML Document	17 KB

File name: GSDML-V2.25-ICC-asdmvethv2-20240308.xml

GSDML Files (GSDML*.xml;GSD

Figure 64: Register GSDML

This will update the **PROFINET Device Catalog**. Locate the device in the **PROFINET Device Catalog** as shown in Figure 65.

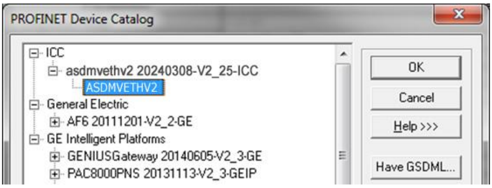


Figure 65: Updated PROFINET Device Catalog

10.5.7.2 Add the Device to the Configuration

Select the device in the **PROFINET Device Catalog** and click the **OK** button as shown in Figure 65. The device is added under the **Profinet Controller** node as shown in Figure 66.

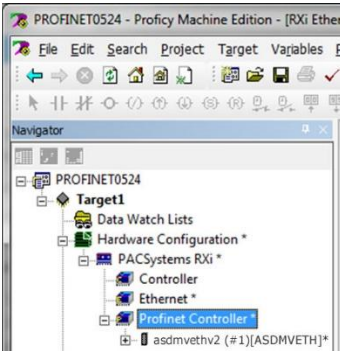
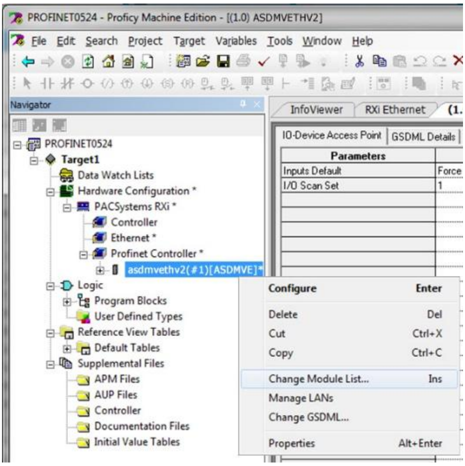


Figure 66: Added Device to Configuration

10.5.7.3 Assign IO Module

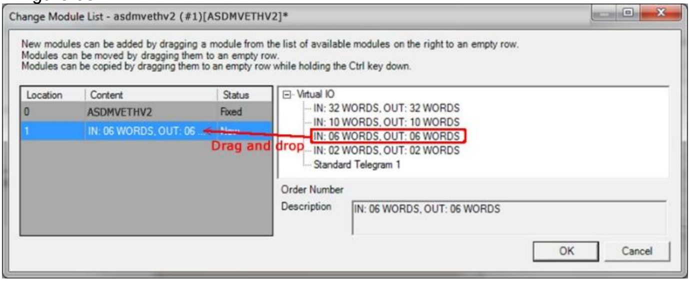
In the **Navigator** panel, right-click on the device and select **Change Module List…** as shown in Figure 67.



Parameters	
Inputs Default	Force
I/O Scan Set	1

Figure 67: 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 68.



Change Module List - asdmvethv2 (#1)[ASDMVETHV2]*

New modules can be added by dragging a module from the list of available modules on the right to an empty row.
Modules can be moved by dragging them to an empty row.
Modules can be copied by dragging them to an empty row while holding the Ctrl key down.

Location	Content	Status
0	ASDMVETHV2	Fixed
1	IN: 06 WORDS, OUT: 06 ...	New

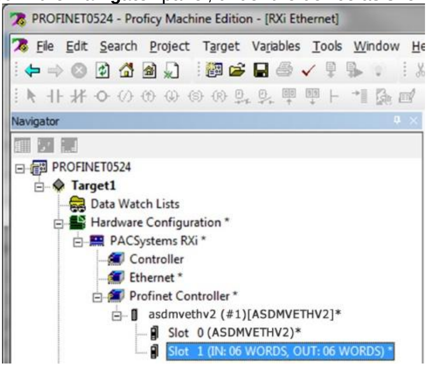
Order Number

Description: IN: 06 WORDS, OUT: 06 WORDS

Drag and drop
Virtual IO
IN: 32 WORDS, OUT: 32 WORDS
IN: 10 WORDS, OUT: 10 WORDS
IN: 06 WORDS, OUT: 06 WORDS
IN: 02 WORDS, OUT: 02 WORDS
Standard Telegram 1
OK
Cancel

Figure 68: Add IO Module

The module will be reflected in the **Navigator** panel, under the device as shown in Figure 69.



PROFINET0524 - Proficy Machine Edition - [RXi Ethernet]
File Edit Search Project Target Variables Tools Window He
Navigator
PROFINET0524
Target1
Data Watch Lists
Hardware Configuration *
PACSystems RXi *
Controller
Ethernet *
Profinet Controller *
asdmvethv2 (#1)[ASDMVETHV2]*
Slot 0 (ASDMVETHV2)*
Slot 1 (IN: 06 WORDS, OUT: 06 WORDS) *

Figure 69: Added IO Module

10.5.7.4 Configure the Device Properties

In the **Navigator** panel, right-click on the device and select **Properties** as shown in Figure 70.

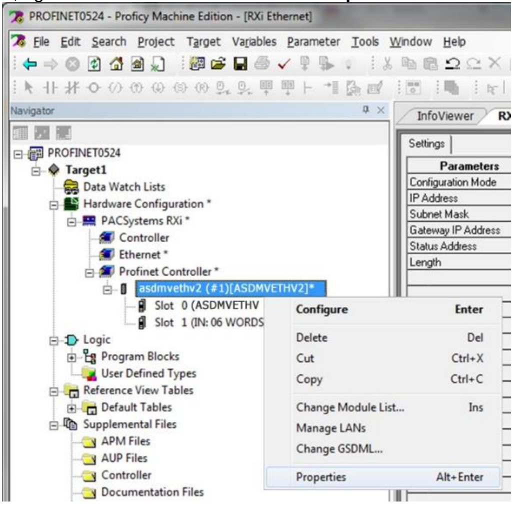


Figure 70: Select Device Properties

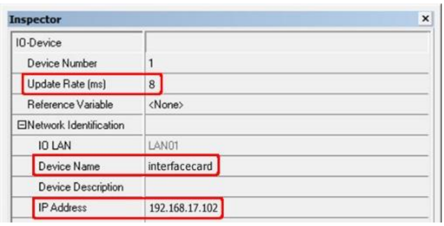
Set the properties to match the configuration on the device. The properties must be appropriate for the application and the PROFINET network.

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 71.



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 71: Set Device Properties

10.5.7.5 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.

10.6 IEC 61850 Server

10.6.1 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 user-configurable.

Some 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**.

10.6.2 Server Settings

In the studio, navigate to **ASD-MVETHV2…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 7.4.1.

10.6.3 GOOSE Communication Parameters

In the studio's Project panel, navigate to **ASD-MVETHV2…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.

10.6.4 Generic Process I/O Status and Control Object Settings

The Generic Status objects define the structure of "status" data, which are typically read-only parameters. The Generic Control objects will define the structure of the "command" data, which are writeable parameters. 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 parameter 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.

Parameter

The inverter parameter (in section 5) associated with this object. For the Generic Status object, enter a “status” parameter to be monitored. For the Generic Control object, enter a “command” parameter 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 summarized in Table 32.

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

Table 32: IEC 61850 Server Default User-Configurable I/O

Type	Name	Instance	Parameter
Generic Control	Control0	0	SCAN_RCV01_AS
Generic Control	Control1	1	SCAN_RCV02_AS
Generic Control	Control2	2	SCAN_RCV03_AS
Generic Control	Control3	3	SCAN_RCV04_AS
Generic Control	Control4	4	SCAN_RCV05_AS
Generic Control	Control5	5	SCAN_RCV06_AS
Generic Control	Control6	6	SCAN_RCV07_AS
Generic Control	Control7	7	SCAN_RCV08_AS
Generic Control	Control8	8	SCAN_RCV09_AS
Generic Control	Control9	9	SCAN_RCV10_AS
Generic Status	Status0	0	SCAN_WR01_AS
Generic Status	Status1	1	SCAN_WR02_AS
Generic Status	Status2	2	SCAN_WR03_AS
Generic Status	Status3	3	SCAN_WR04_AS
Generic Status	Status4	4	SCAN_WR05_AS
Generic Status	Status5	5	SCAN_WR06_AS
Generic Status	Status6	6	SCAN_WR07_AS
Generic Status	Status7	7	SCAN_WR08_AS
Generic Status	Status8	8	SCAN_WR09_AS
Generic Status	Status9	9	SCAN_WR10_AS
Generic Status	Status10	10	SCAN_WR11_AS
Generic Status	Status11	11	SCAN_WR12_AS
Generic Status	Status12	12	SCAN_WR13_AS
Generic Status	Status13	13	SCAN_WR14_AS
Generic Status	Status14	14	SCAN_WR15_AS

Generic Status	Status15	15	SCAN_WR16_AS
Generic Status	Status16	16	SCAN_WR17_AS
Generic Status	Status17	17	SCAN_WR18_AS
Generic Status	Status18	18	SCAN_WR19_AS
Generic Status	Status19	19	SCAN_WR20_AS
Generic Status	Status20	20	SCAN_WR21_AS
Generic Status	Status21	21	SCAN_WR22_AS
Generic Status	Status22	22	SCAN_WR23_AS
Generic Status	Status23	23	SCAN_WR24_AS
Generic Status	Status24	24	SCAN_WR25_AS

11 SERIAL PROTOCOL-SPECIFIC INFORMATION

This section will discuss topics that are specific to each of the supported serial (RS-485) protocols. To enable a protocol, add the protocol to the RS-485 configuration.

11.1 Modbus RTU

11.1.1 Overview

The interface card supports the Modbus RTU slave protocol. Other notes of interest are:

- Specific bits within inverter registers can be accessed as either coils (0X references) or discrete inputs (1X references).
- Supports broadcast on write function codes.
- Write data checking is not available. For example, if a write is performed to a register with a data value that is out-of-range of the corresponding parameter, no Modbus exception will be immediately returned.
- The driver can be configured to detect a timeout (communication loss) and perform a timeout action.

11.1.2 Functions

The supported Modbus/TCP functions are indicated in Table 33.

Table 33: Supported Modbus RTU Functions

Function Code	Function	Support Broadcast
1	Read coils	No
2	Read input status	No
3	Read multiple registers	No
4	Read input registers	No
5	Write coil	Yes
6	Write single register	Yes
8	Diagnostics (subfunction 0 only)	No
15	Force multiple coils	Yes
16	Write multiple registers	Yes

11.1.3 Holding & Input Registers

The inverter registers by default are mapped as both holding registers (4X) and input registers (3X) and are accessed by using the inverter register numbers described in section 5.1. Examples are shown in Table 34.

Table 34: Modbus RTU Register Examples

Function Code	Holding/Input Register Number
SCAN_RCV01_AS	1
SCAN_WR01_AS	11

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 40001 is the same as Modbus holding register 1. The same description applies to input registers (3X).

For example, from a Modbus master’s point of view, in order to access the parameter SCAN_WR01_AS (register 11) as a holding register, the Modbus master must execute the Read Multiple Registers function code and target register 11. This will similarly apply when accessing an inverter parameter as an Input Register.

11.1.4 *Coil & Discrete Input Mappings*

The Modbus 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 indices into various bits of existing registers. What this means is that when a discrete is accessed, that discrete is resolved by the interface card 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:

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

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

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

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

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

For clarity, let's use Equation 3 and Equation 4 in a calculation example. Say, for instance, that we are going to read coil #34. Using Equation 3 we can determine that coil #34 resides in register #3, as $\lfloor 3.0625 \rfloor = \lfloor 3 \text{ r}1 \rfloor = 3$. Then, using Equation 4, we can determine that the bit within register #3 that coil #34 targets is (34-1)%16 = 1, as 33%16 = mod(2 r1) = 1. Therefore, reading coil #34 will return the value of register #3, bit #1.

11.1.5 *Slave Settings*

In the studio's Project panel, navigate to **ASD-MVETHV2…RS-485…Modbus RTU Slave**.

<u>Baud Rate</u>

Selects the baud rate of the network. If the desired baud rate is not available, please contact technical support for assistance.

<u>Parity</u>

Selects the parity and the number of stop bits.

<u>Response Delay</u>

Defines the time in milliseconds that the driver waits before responding to master requests. This is a useful feature for certain master devices or infrastructure components (such as radio modems) that may require a given amount of time to place themselves into a "receiving mode" where they are capable of listening for slave responses. If no delay is required, setting this field to 0 instructs the driver to send its responses as soon as possible.

11.1.6 *Connection Timeout Options*

In the studio's Project panel, navigate to **ASD-MVETHV2…RS-485…Modbus RTU Slave**. The following configuration options will determine the actions to be taken if there is no valid activity on the network.

<u>Enable Connection Timer</u>

This timer provides the ability for the driver to monitor activity on the network. If a valid Modbus packet is not detected for more than the **Timeout** time setting, then the driver assumes that the master or network has experienced some sort of unexpected problem, and will perform the **Timeout Action**.

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 7.5.1.

11.1.7 Node Settings

In the studio's Project panel, navigate to **ASD-MVETHV2…RS-485…Modbus RTU Slave…Node**.

Address

Defines the station address (1…247) for this node.

11.1.8 Holding/Input Register Remap Settings

In the studio's **Project** panel, add **ASD-MVETHV2… RS-485…Modbus RTU Slave …Node…Holding/Input Register Remap**.

The holding/input register remap objects are **OPTIONAL**. By default, all inverter parameters are already mapped as both holding (4X) and input (3X) registers (refer to section 11.1.2). For user convenience, register remap objects can be created to map any inverter parameter to holding/input register 5001 to 5050.

At times, it may be convenient to access inverter parameters in bulk Modbus transactions. This may be especially true in situations where it is desired to access certain parameters that are non-contiguous. For example, if it were desired to read the parameters SCAN_WR01_AS (register 11), SCAN_WR21_AS (register 31), and SCAN_WR25_AS (register 35), 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 11 for a quantity of 25 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 parameters can be grouped together in any order and accessed efficiently via the Modbus "read multiple registers" and "write multiple registers" parameters. 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

Remap register that maps to the specified inverter parameter. Select from 5001 to 5050.

Parameter

Parameter (in section 5) that is accessed by the **Remap Register**.

Data Type

Fixed to 16-Bit Unsigned. This is equivalent to two bytes.

12 TROUBLESHOOTING

Although by no means exhaustive, Table 35 provides possible causes behind some of the most common errors experienced when using the interface card.

Table 35: Troubleshooting

Problem	Symptom	Solution
No communication between the interface card and the inverter	The Module Status LED blinks a red error code: 7-1-4	• Check the inverter parameter configuration (refer to section 4). • Check the installation (refer to section 2).
No communication between the Ethernet network and the interface card	Communication cannot be established, the Ethernet “link” LED is off, or the Ethernet “activity” LED flashes only infrequently or not at all	• Confirm that the interface card is running normally (Module Status LED is not blinking red) and connected to the local Ethernet network. • Ensure that the interface card is configured 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 card 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 communication between the RS-485 network and the interface card	Communication cannot be established, the RS-485 RXD LED is on solid, or the RS-485 RXD and TXD LEDs flash only infrequently or not at all	• Confirm that the interface card is running normally (Module Status LED is not blinking red). • If the RS-485 RXD LED is on solid, the polarity of the data lines (D-, D+) may be reversed. • Ensure that the pluggable terminal block is fully inserted into the port. • Ensure that the RS-485 wires are making metal-to-metal contact with each terminal in the pluggable terminal block. • Ensure that each wire is securely fastened in each terminal. • Confirm that the signal common is connected to all RS-485 devices. • Check for broken wires. • Ensure that the interface card is configured with the correct serial settings (baud rate, stop bit, parity bit). • Confirm that the address programmed into the controller equipment matches that of the interface card, as displayed by the studio.

Problem	Symptom	Solution
No PROFINET communication	PROFINET I/O communication cannot be established. The “Network Status” LED is not solid green.	• Confirm that the interface card’s PROFINET device name matches the name assigned in the controller’s configuration. • Confirm that the interface 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 interface card is connected to a 100Mbps full duplex capable switch. • Ensure that the interface card can be discovered using the controller’s discovery tool.
Unable to command the inverter via network communication	Writing to command parameters has no apparent effect on inverter operation	• Check the inverter parameter configuration (refer to section 4). • Check the param.xml (refer to section 5.2). • If using the inverter’s terminal contacts, refer to the inverter’s instruction manual to determine the appropriate behavior and priority.
The interface card is unresponsive	Ethernet “activity” LED is solid on	If the interface card is connected in a ring topology, all devices in the ring must be configured with the same ring redundancy protocol (i.e. MRP, DLR). 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.
XML socket connection failed	Message on a web server tab information window	TCP port 843 is blocked by a firewall, router or some other intermediate network equipment.
New web server content not loading after web server update	Old web server content is displayed	The internet browser has cached the old web server content. Clear the internet browser’s cache before attempting to load the new web server content.
Web page does not display properly	Corrupt web server	• Ensure that USB and FTP are disconnected. • Delete the “WEB” folder from the interface card’s file system and copy a valid “WEB” folder to the interface card’s file system. • Use the Configuration Studio.
Studio cannot discover the interface card	The studio does not display the interface card under “Online Devices”	• Confirm that the interface 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.

Problem	Symptom	Solution
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 interface card's file system may be corrupt. Please format the interface card's file system and then restore the configuration (refer to section 7.10). If the file system cannot be formatted, please contact technical support for additional assistance.
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.

INDUSTRIAL CONTROL COMMUNICATIONS, INC.

230 Horizon Drive, Suite 100
Verona, WI 53593
USA
Tel: +1 (608) 831-1255
https://www.iccdesigns.com