



# INVERTER

Plug-in option

## A8NXL

# INSTRUCTION MANUAL

*RS-485 multiprotocol  
communication interface*



September 2021



PRE-OPERATION INSTRUCTIONS	1
INSTALLATION	2
WIRING	3
INVERTER SETTINGS	4
REGISTER NUMBERING AND BEHAVIOR	5
MITSUBISHI NETWORK PARAMETER UTILITY	6
MITSUBISHI CONFIGURATION STUDIO	7
FIRMWARE	8
PROTOCOL-SPECIFIC INFORMATION	9
TROUBLESHOOTING	10

Thank you for choosing this Mitsubishi inverter plug-in option. This instruction manual provides handling information and precautions for use of this equipment. Incorrect handling may cause unexpected failures or damage. In order to ensure optimal performance, please read this manual carefully prior to use of the equipment. Please forward this manual to the end user of the equipment.

### Safety instructions

Do not attempt to install, operate, maintain or inspect this product until you have read through this instruction manual and any related documents carefully, and can use the equipment properly. Do not use this product until you have a full working knowledge of the equipment, safety information and instructions. In this instruction manual, the safety instruction levels are classified into "WARNING" and "CAUTION".



#### Warning

Assumes that incorrect handling may cause hazardous conditions resulting in death or severe injury.



#### Caution

Assumes that incorrect handling may cause hazardous conditions resulting in moderate or slight injury, or may cause physical damage only.



#### Caution

Please note that even the level may lead to serious consequence depending on conditions. Please be sure to follow the instructions of both levels as they are critical to personnel safety.

### ◆ Electrical Shock Prevention



#### Warning

- Do not open the front cover of the inverter while power is on or while the inverter is running, as an electrical shock may result.
- 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, resulting in an electrical shock.
- If power is off, do not remove the front cover except when necessary for wiring or periodic inspection. While the front cover is removed, accidental contact with exposed high-voltage terminals and internal components may occur, resulting in an electrical shock.
- Prior to starting wiring or inspection, confirm that input power to the inverter has been switched off via observation of the inverter's display panel. Additionally, wait for at least 10 minutes after removal of input power, and then confirm that all residual voltage has been dissipated by using a voltage meter. Internal DC bus capacitors may contain high voltages for several minutes after removal of input power, resulting in a dangerous situation should anything come into contact with them.
- All personnel involved in the installation or inspection of this equipment should be fully competent to perform the required work.
- Always install plug-in options prior to wiring main power.
- Do not touch the plug-in option with wet hands.
- Do not subject the cables to scratches, excessive stress, heavy loads or pinching.

## ◆ Injury Prevention



### Caution

- To prevent explosions or similar damage, apply only the voltages specified in the instruction manual to each terminal.
- To prevent explosions or similar damage, ensure that all cables are properly connected to the correct terminals.
- To prevent explosions or similar damage, observe all wiring polarity indicators.
- To prevent burns from hot components, do not touch the inverter while power is on, or for some time after power is removed.

## ◆ Additional Instructions

Please note the following points to prevent equipment damage, injury or electrical shock.



### Caution

#### Transportation and mounting

- Do not install or operate the plug-in option if it is damaged or has parts missing.
- Do not stand on or rest heavy objects on the equipment.
- Check that the mounting orientation is correct.
- Prevent conductive items such as screws and metal fragments, or flammable substances such as oil from entering the inverter.
- If halogen-based materials (fluorine, chlorine, bromine, iodine, etc.) infiltrate into a Mitsubishi product, the product will be damaged. Halogen-based materials are often included in fumigants which are used to sterilize or disinfect wooden packaging. When packaging, prevent residual fumigant components from being infiltrated into Mitsubishi products, or use an alternative sterilization or disinfection method for packaging, such as heat disinfection, etc. Whenever possible, sterilization or disinfection of wooden packaging should be performed prior to packaging the product.

#### Trial run

- To prevent unexpected equipment movement, confirm and adjust all required parameters prior to starting operation.



### Warning

#### Usage

- Do not modify the equipment.
- Do not remove any inverter or option parts unless specifically instructed to do so in this manual.



## Caution

### Usage

- Performing a "parameter clear" or "all parameter clear" will reset all inverter parameters to their factory default settings. After performing one of these operations, remember to reenter any custom parameter values prior to starting operation.
- To prevent damage from electrostatic discharge, always touch a grounded piece of metal prior to touching any equipment.

### Maintenance, inspection and parts replacement

- Do not perform hi-pot tests on the equipment.

### Disposal

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

## General instructions

- For clarity purposes, illustrations in this manual may be drawn with covers or safety guards removed. Never operate the inverter in this manner. Ensure all covers and safety guards are properly installed prior to starting operation.

## – CONTENTS –

<b>1</b>	<b>PRE-OPERATION INSTRUCTIONS</b>	<b>10</b>
1.1	Product Overview .....	10
1.2	Features and Specifications .....	11
1.3	Unpacking and Product Confirmation .....	18
1.3.1	Shipment Confirmation .....	18
1.3.2	Component Overview .....	19
1.4	LED Indicators .....	20
1.4.1	Port Activity LED Description .....	20
1.4.2	Status LED Description.....	21
<b>2</b>	<b>INSTALLATION</b>	<b>22</b>
2.1	Pre-Installation Instructions.....	22
2.2	Installation Procedure .....	23
<b>3</b>	<b>WIRING</b>	<b>27</b>
3.1	Terminals.....	27
3.2	Wiring .....	28
<b>4</b>	<b>INVERTER SETTINGS</b>	<b>30</b>
4.1	Network Setting .....	31
4.1.1	Status Code (Pr. 1303) .....	31
4.1.2	Run Mode (Pr. 1304).....	32
4.1.3	Protocol (Pr. 1305).....	32
4.1.4	Address (Pr. 1306).....	33

4.1.5	Baud Rate (Pr. 1307) .....	33
4.1.6	Parity/Stop Bits (Pr. 1308) .....	34
4.1.7	Timeout Time (Pr. 1309) .....	34
4.1.8	Response Delay (Pr. 1310) .....	34
4.1.9	Number of Retries (Pr. 1311) .....	34
4.1.10	Device Instance Number (Upper 3 Digits) (Pr. 1315) .....	35
4.1.11	Device Instance Number (Lower 4 Digits) (Pr. 1316) .....	35
4.1.12	Max Master (Pr. 1317) .....	35
4.1.13	Max Info Frames (Pr. 1318) .....	35
4.1.14	Fail-safe Timeout Time (Pr. 1319) .....	35
4.1.15	Firmware Version (Pr. 1350) .....	35
<b>4.2</b>	<b>Operation Mode Setting</b> .....	<b>37</b>
4.2.1	Operation Mode Indication .....	37
4.2.2	Operation Mode Switching & Comm. Startup Mode (Pr. 79, Pr. 340) .....	38
<b>4.3</b>	<b>Operation &amp; Speed Command Source (Pr. 338, Pr. 339, Pr. 550)</b> .....	<b>41</b>
<b>4.4</b>	<b>Communication EEPROM Write Selection (Pr. 342)</b> .....	<b>42</b>
<b>4.5</b>	<b>Operation at communication error occurrence</b> .....	<b>43</b>
4.5.1	Operation selection at communication error occurrence (Pr. 500 to Pr. 502, Pr. 779) .....	43
4.5.2	Fault and measures .....	50
<b>4.6</b>	<b>Inverter reset</b> .....	<b>52</b>
<b>5</b>	<b>REGISTER NUMBERING AND BEHAVIOR</b> .....	<b>54</b>
<b>5.1</b>	<b>Parameter Synchronization</b> .....	<b>54</b>
5.1.1	Cyclic Registers .....	55
5.1.2	Scanned Registers .....	56
<b>5.2</b>	<b>Register Numbers</b> .....	<b>57</b>
<b>5.3</b>	<b>Inverter Command Register</b> .....	<b>64</b>

5.4	Inverter Reset Register .....	65
5.5	Alarm History Clear Register .....	65
5.6	All Parameter Clear Register .....	65
5.7	Inverter State Register .....	66
5.8	Process Parameters Scan Cycle Time Register .....	67
5.9	Configuration Parameters Scan Cycle Time Register .....	67
5.10	Number of Process Parameters Register .....	67
5.11	Number of Configuration Parameters Register .....	67
5.12	DriveStatus Register .....	68
5.13	SetpointSpeed Register .....	69
5.14	ActualSpeed Register .....	69
5.15	Fault History Codes .....	69
<b>6</b>	<b>MITSUBISHI NETWORK PARAMETER UTILITY</b>	<b>70</b>
6.1	Overview .....	70
6.2	Features .....	71
<b>7</b>	<b>MITSUBISHI CONFIGURATION STUDIO</b>	<b>72</b>
7.1	Overview .....	72
7.2	General Object Editing Activities .....	77
7.3	Device Settings .....	79
7.3.1	Process Data Tuning .....	79
7.3.2	Status LED Settings .....	80
7.4	USB Virtual COM Port Settings .....	82
7.5	USB Serial Capture Window .....	83
7.6	Port Diagnostics .....	86
7.7	Batch Update Mode .....	86

<b>7.8</b>	<b>Internal Logic Settings .....</b>	<b>88</b>
7.8.1	Fail-safe Values.....	88
7.8.1.1	Overview .....	88
7.8.2	Timeout Time .....	88
7.8.2.1	Timeout Object Configuration .....	88
<b>7.9</b>	<b>Database Logic .....</b>	<b>89</b>
7.9.1	Overview .....	89
7.9.2	Database Logic Settings.....	92
7.9.3	Enable Trigger.....	92
7.9.3.1	Trigger Options.....	93
<b>7.10</b>	<b>Manage Device Parameters.....</b>	<b>94</b>
<b>7.11</b>	<b>Monitor Device Parameters.....</b>	<b>95</b>
<b>7.12</b>	<b>Backup and Restore Parameters .....</b>	<b>97</b>
<b>7.13</b>	<b>Restore Factory Settings.....</b>	<b>99</b>
<b>7.14</b>	<b>Help.....</b>	<b>99</b>
<b>8</b>	<b>FIRMWARE .....</b>	<b>100</b>
<b>8.1</b>	<b>Overview .....</b>	<b>100</b>
<b>8.2</b>	<b>Update Procedure.....</b>	<b>100</b>
<b>9</b>	<b>PROTOCOL-SPECIFIC INFORMATION .....</b>	<b>102</b>
<b>9.1</b>	<b>BACnet MS/TP.....</b>	<b>102</b>
9.1.1	Protocol Implementation Conformance Statement (PICS).....	102
9.1.2	Default Supported Objects.....	110
9.1.3	BACnet MS/TP Server Settings .....	113
9.1.4	BACnet Object Settings .....	113
9.1.4.1	Analog Input Object Settings.....	113



9.1.4.2	Analog Output Object Settings.....	114
9.1.4.3	Analog Value Object Settings .....	115
9.1.4.4	Binary Input Object Settings.....	117
9.1.4.5	Binary Output Object Settings.....	118
9.1.4.6	Binary Value Object Settings .....	119
9.1.4.7	Multi-state Input Object Settings .....	121
9.1.4.8	Multi-state Output Object Settings.....	122
9.1.4.9	Multi-state Value Object Settings .....	123
<b>9.2</b>	<b>Modbus RTU .....</b>	<b>125</b>
9.2.1	Overview .....	125
9.2.2	Holding & Input Registers .....	125
9.2.3	Coil & Discrete Input Mappings.....	126
9.2.4	Holding/Input Register Remap Settings .....	127
<b>9.3</b>	<b>Metasys N2.....</b>	<b>129</b>
9.3.1	Default Supported Objects.....	129
9.3.2	Metasys N2 Slave Settings .....	132
9.3.3	Metasys Object Settings .....	132
9.3.3.1	Analog Input Object Settings.....	133
9.3.3.2	Analog Output Object Settings.....	134
9.3.3.3	Binary Input Object Settings.....	135
9.3.3.4	Binary Output Object Settings.....	135
9.3.3.5	Internal Float Object Settings.....	136
9.3.3.6	Internal Integer Object Settings.....	137
9.3.3.7	Internal Byte Object Settings.....	137
<b>9.4</b>	<b>Siemens FLN (P1) .....</b>	<b>139</b>
9.4.1	Default Supported Objects.....	139
9.4.2	Siemens FLN Slave Settings .....	142
9.4.3	Node Settings .....	142
9.4.4	FLN Object Settings.....	143

9.4.4.1	Logical Analog Input (LAI) Object Settings .....	143
9.4.4.2	Logical Analog Output (LAO) Object Settings .....	144
9.4.4.3	Logical Digital Input (LDI) Object Settings .....	145
9.4.4.4	Logical Digital Output (LDO) Object Settings .....	146
<b>9.5</b>	<b>DMX-512 .....</b>	<b>148</b>
9.5.1	Connections .....	148
9.5.2	Default Supported Objects .....	149
9.5.3	DMX-512 Slave Settings .....	150
9.5.4	DMX Parameter Settings .....	150
<b>9.6</b>	<b>Generic Serial .....</b>	<b>151</b>
9.6.1	Generic Serial Slave Settings .....	151
9.6.2	Transactions .....	152
9.6.2.1	Slave Transaction .....	152
9.6.2.2	Consumer Transaction .....	152
9.6.2.3	Transaction Settings .....	152
9.6.3	Packet Data Objects .....	153
9.6.3.1	Constant Data .....	153
9.6.3.2	Drive Register Data .....	153
9.6.3.3	Variable Drive Register Data .....	154
9.6.3.4	Ignored Characters .....	156
9.6.3.5	Variable Ignored Characters .....	156
9.6.3.6	Ranged Byte .....	156
9.6.3.7	Bitmasked Byte .....	157
9.6.3.8	Register Matched Byte .....	157
9.6.3.9	List Matched Character .....	157
9.6.3.10	Checksum .....	158
9.6.3.11	CRC .....	159



# PRE-OPERATION INSTRUCTIONS

---

## 1.1 Product Overview

---

The A8NXLT RS-485 multiprotocol communication interface allows information to be transferred seamlessly between an 800-series inverter and several different RS-485-based fieldbus networks with minimal configuration requirements. The interface installs directly onto the inverter's control board, and presents a fixed, screw-style terminal block for connection to the RS-485 network.

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

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

The A8NXLT may be referred to throughout the remainder of the manual as the device, interface, card, and option (or any combination thereof).

### Supported Protocols

The interface currently provides support for the following fieldbus protocols:

- BACnet MS/TP Server
- Modbus RTU Slave
- Metasys N2 Slave
- Siemens FLN Slave
- DMX-512 Slave
- Generic Serial Slave



- *BACnet is a registered trademark of ASHRAE*
- *Modbus is a registered trademark of Schneider Electric USA, Inc., licensed to the Modbus Organization, Inc.*
- *Metasys is a registered trademark of Johnson Controls, Inc.*

## 1.2 Features and Specifications

Table 1: Features

Item	Description
Multiple RS-485 Protocols	Active protocol and other network settings selected via inverter parameters
Mitsubishi Network Parameter Utility	Graphical user interface for configuring option card parameters for RS-485 networks.
Mitsubishi Configuration Studio	Graphical user interface for discovery, object configuration, firmware update, and troubleshooting
Communication Loss Detection	Configurable actions for “fail-safe” conditions
PLC-Style Data Manipulation	Database logic allows construction of complex autonomous data conditioning
Field Upgradeable	Firmware updates automatically handled by the studio
Parameter Management	Advanced management of parameter scan priority
Parameter Backup and Restore	Drive cloning

**Table 2: General Hardware Specifications**

Item	Description
Power Supply	Directly powered by the inverter or via USB
Grounding	Referenced to inverter's 5V power supply / isolated from inverter control power common
LED Indicators	Module Status, RS-485 TX, RS-485 RX
USB Port	USB 2.0, mini-B 5-pin

**Table 3: RS-485 Hardware Specifications**

Item	Description
Number of Ports	1
Isolation	1.5kV galvanic isolation
Standard	TIA/EIA-485
Communication Speed and Duplex	300 - 250k baud (protocol dependent), half duplex
Connector Type	Fixed, 3 position, screw-style terminal block
Cable Type	Twisted pair
Cable Length	Up to 1,200 m (4,000 ft)
Topologies	Daisy-chain

**Table 4: BACnet MS/TP Specifications**

Item	Description
Data Link	Master-Slave/Token Passing (MS/TP)
Protocol Revision	12
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 (DS-WP-B), WritePropertyMultiple-B (DS-WPM-B), COV-B (DS-COV-B), Dynamic Device Binding-B (DM-DDB-B), Dynamic Object Binding-B (DM-DOB-B), DeviceCommunicationControl-B (DM-DCC-B), ReinitializeDevice-B (DM-RD-B)
Segmentation	Not supported
Max APDU Length	480 bytes
Character Sets	ISO 10646 (UTF-8)
Object Types	Analog Output, Analog Input, Analog Value, Binary Output, Binary Input, Binary Value, Device, Multi-state Output, Multi-state Input, Multi-state Value
Priority Array	Yes
Baud Rates	9600, 19200, 38400, 57600, 76800, 115200
Parity	No Parity
Stop Bits	1
Data Bits	8

**Table 5: 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)
Max Read Register Size	125 registers
Max Write Register Size	123 registers
Register Data Type	16-bit integer
Baud Rates	2400, 4800, 9600, 19200, 38400, 57600, 115200
Parity	No Parity, Odd Parity, Even Parity
Stop Bits	1, 2
Data Bits	8

**Table 6: Metasys N2 Specifications**

Item	Description
Object Types	Analog Output, Analog Input, Binary Output, Binary Input, Internal Float (ADF), Internal Integer (ADI), Internal Byte (BD)
Baud Rates	9600
Parity	No Parity
Stop Bits	1
Data Bits	8

**Table 7: Siemens FLN Specifications**

Item	Description
Application Number	7360
Application Descriptor	MIT S A800
Revision String	ICC0
Revision Number	0
Object Types	LAI, LAO, LDI, LDO
Baud Rates	2400, 4800, 9600, 19200, 38400, 57600, 115200
Parity	No Parity
Stop Bits	1
Data Bits	8



**Table 8: DMX-512 Specifications**

Item	Description
RDM Support	No
Baud Rates	250000
Parity	No Parity
Stop Bits	2
Data Bits	8

**Table 9: Generic Serial Specifications**

Item	Description
Packet Definition	Transaction-based
Packet Delimiter	Packet Gap Interval (Character Times)
Encoding	Binary, ASCII
Error Checking	Checksum, CRC
Baud Rates	300, 600, 1200, 2400, 4800, 9600, 19200, 38400, 57600, 115200
Parity	No Parity, Odd Parity, Even Parity
Stop Bits	1, 2
Data Bits	7, 8

**Table 10: Environmental Specifications**

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

## 1.3 Unpacking and Product Confirmation

### 1.3.1 Shipment Confirmation

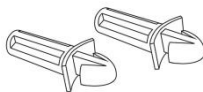
Check the enclosed items. Confirm that the correct quantity of each item was received, and that no damage occurred during shipment.



Plug-in option: qty. 1



Communication option LED display cover: qty. 1



5.5mm Spacer: qty. 2



M3 x 8mm Mounting screws: qty. 3



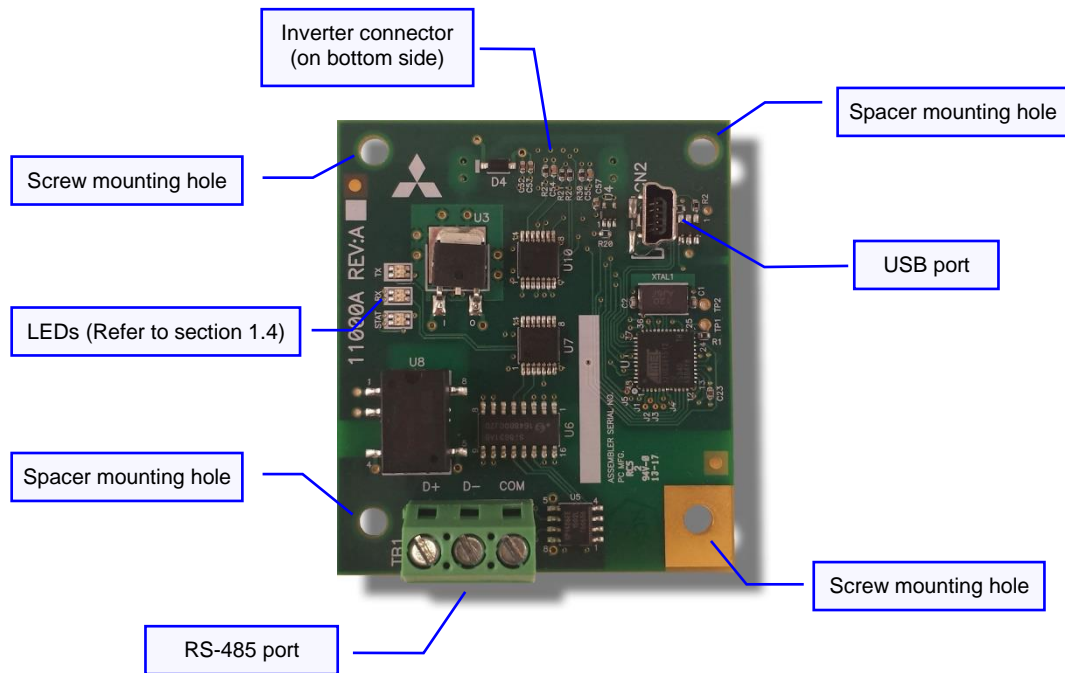
Grounding bar: qty. 1



USB interface cable: qty. 1

### 1.3.2 Component Overview

1



## 1.4 LED Indicators

The option board contains several bi-color LEDs (visible through the LED display cover after mounting) that provide a visual indication of the unit's overall status.

### 1.4.1 Port Activity LED Description

RS-485 Transmit (TX)

LED Activity	Status	Note
Off	Not Transmitting	The device is not transmitting data on the RS-485 network
Green Blink	Transmitting	The device is transmitting data on the RS-485 network
Green On	Transmitting	The device is transmitting data rapidly on the RS-485 network

RS-485 Receive (RX)

LED Activity	Status	Note
Off	Not Receiving	The device is not receiving data on the RS-485 network
Green Blink	Receiving	The device is receiving data on the RS-485 network
Green On	Receiving	The device is receiving data rapidly on the RS-485 network

## 1.4.2 Status LED Description

### Module Status (STATUS)

LED Activity	Status	Note
Off	Device Off	Power is not applied to the option card
Green Blink, Red Blink	Startup	Startup blink sequence
Green Blink (Rapid)	Sync	The option card is synchronizing parameters with the inverter
Green On	Device On	Normal status
Green Blink	USB Connection	An active USB connection is present
Red Blink	Error Code	Refer to the TROUBLESHOOTING section

## **INSTALLATION**

---

### **2.1 Pre-Installation Instructions**

---

Check that the inverter's input power and the control circuit power are both OFF.



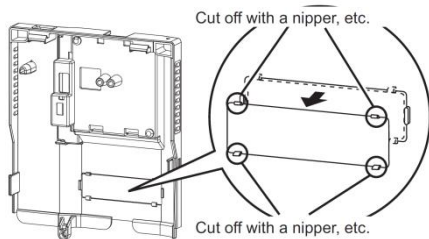
#### **Caution**

- To prevent explosions or similar damage, do not install or remove the plug-in option with input power ON.
- To prevent damage from electrostatic discharge, always touch a grounded piece of metal prior to touching any equipment.

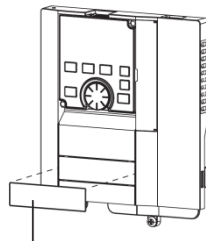
## 2.2 Installation Procedure

### ◆ Installing the communication option LED display cover

- 1) Remove the inverter's front cover (refer to chapter 2 of the inverter's *Instruction Manual (Detailed)* for information on how to remove the front cover.)
- 2) Cut off the tabs on the back side of the inverter's front cover with a nipper, etc. and open the knockout for the communication option LED display cover.



- 3) Fit the communication option LED display cover into the knockout on the front side of the inverter's front cover, orienting it with the LED positions on the plug-in option circuit board. Push the LED display cover firmly until the tabs lock into place.



Communication option LED display cover



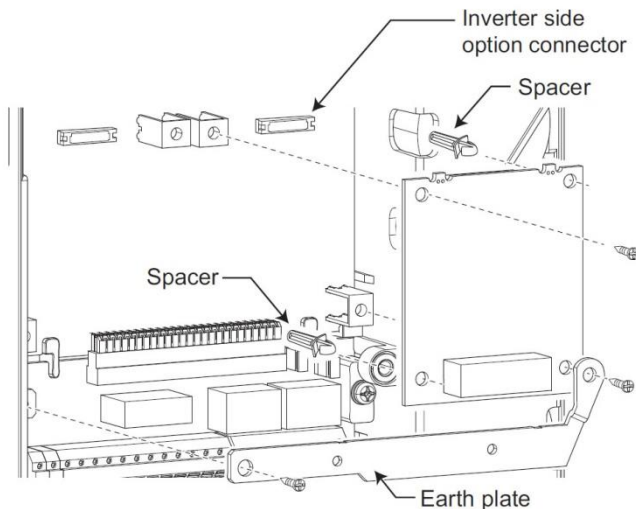
### Caution

- To prevent hand injury, avoid contacting the tabs on the backside of the inverter's front cover.



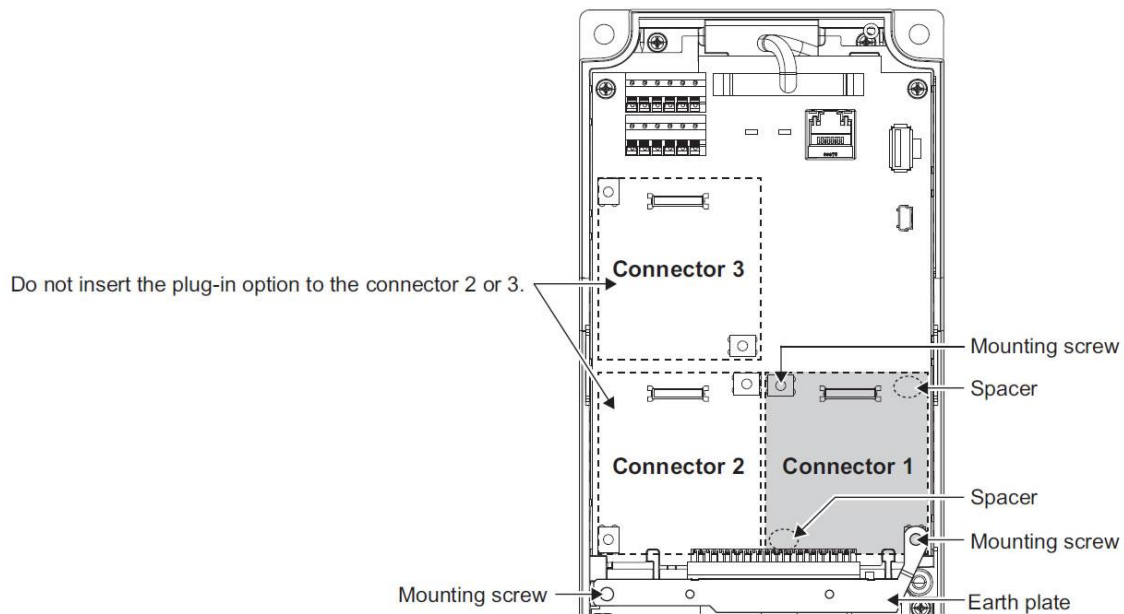
### ◆ Installing the option

- 1) Insert spacers from the back side of the option card into the 2 mounting holes as indicated.
- 2) Securely attach the option card to the inverter's option Connector 1 (refer to Figure 1 on page 25 for the location of Connector 1 on the inverter's control board). Ensure that the option card is fully seated on the inverter's option connector.
- 3) Install the grounding bar as indicated, using a mounting screw to attach the left side of the grounding bar to the inverter's standoff, and another mounting screw to attach the right side of the grounding bar through the option card to the standoff below it.
- 4) Install the remaining mounting screw through the hole on the upper left side of the option card to the standoff below it.
- 5) Tighten all mounting screws to a torque setting of 0.33 N·m to 0.40 N·m.



### ◆ Completing the installation

- 1) Attach the network cable(s) to the plug-in option's RS-485 port as required, and route the cable(s) out of the inverter along with other control-level wiring. Ensure that sufficient clearance exists between the communication cables and the inverter's input power and output motor wiring.
- 2) Reinstall the inverter's front cover (refer to chapter 2 of the inverter's *Instruction Manual (Detailed)* for additional information.)



**Figure 1: Installation Positions of Screws and Spacers**

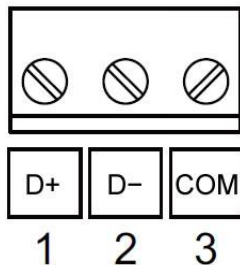


- To minimize the possibility of damage, hold only the sides of the circuit board when installing or uninstalling the plug-in option. Stress applied to circuit board components by directly pressing/pulling on them may cause component damage and subsequent failure.
- Use caution to avoid dropping mounting screws into the inverter enclosure when installing or uninstalling the plug-in option.
- To minimize the possibility of malfunction due to electrical noise, always install the included grounding bar.
- The plug-in option must only be installed on the inverter control board's option Connector 1 (refer to Figure 1 on page 25.) If installed onto the Connector 2 or Connector 3 positions, the inverter will fault with an E.2 or E.3 fault code.
- Even if the plug-in option is installed onto the correct Connector 1 position, the inverter may still fault with an E.1 fault code if the option card is not properly installed or fully seated onto the inverter control board's option connector.

Mounted position	Fault indication
Option Connector 1	E. 1
Option Connector 2	E. 2
Option Connector 3	E. 3

- When removing the plug-in option, first remove the three M3x8mm mounting screws and grounding bar. Lastly, remove the option card by grasping it on its left and right side and pulling it straight away from the inverter.

## 3.1 Terminals



Terminal no.	Terminal name	Definition
1	D+	RS-485 non-inverting data signal
2	D-	RS-485 inverting data signal
3	COM	RS-485 common-mode reference

(1) Be sure to check the following before connecting the inverter to the network.

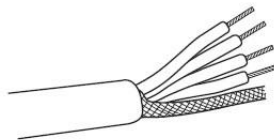
- Check that the A8NXLT is securely inserted into the inverter. (Refer to page 22.)
- Check that the correct network settings are set for the option board. (Refer to page 31.)
- Check that the RS-485 cable is firmly connected to the A8NXLT.

## 3.2 Wiring

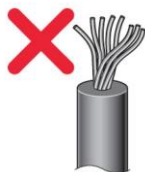
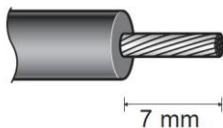
Use network connection cable rated for use with RS-485 networks. Because 3 signals are required, it is recommended to use either a single twisted pair plus one conductor cable or a double twisted pair cable.

(1) Strip the sheath of the RS-485 communication cable and wind wires and shield cables to use.

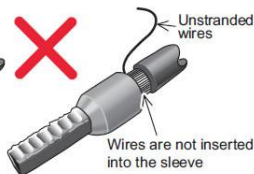
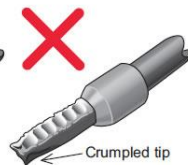
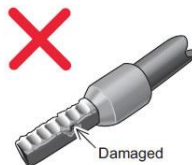
Strip off the sheath for the below length. If the length of the sheath peeled is too long, a short circuit may occur with neighboring wires. If the length is too short, wires might come off. Wire the stripped cable after twisting it to prevent it from becoming loose. In addition, do not solder it.



Cable stripping length



Use a blade terminal as necessary. When using the blade terminal, use care so that the twisted wires do not come apart.



**NOTE**

- Blade terminal commercially available (as of February 2012. The product may be changed without notice.)

Terminal screw size	Wire size (mm <sup>2</sup> )	Blade terminal model		Manufacturer	Crimping tool name
		With insulation sleeve	Without insulation sleeve		
M3	0.3 to 0.5	AI 0,5-6WH	A 0,5-6	Phoenix Contact Co.,Ltd.	CRIMPFOX 6
	0.5 to 0.75	AI 0,75-6GY	A 0,75-6		

(2) Loosen the terminal screw and insert the cable into the terminal according to the terminal assignment. A single twisted pair must be used for the RS-485 data signals. The RS-485 common-mode signal may use either a single conductor or a twisted pair depending on the cabling being used.

Tighten each cable with fixing screws to the recommended tightening torque.

Screw Size	Tightening torque	Cable size	Screwdriver
M3	0.4 N·m	0.2 mm <sup>2</sup> to 0.8 mm <sup>2</sup> 24 AWG to 18 AWG	Small  flat-blade screwdriver (Tip thickness: 0.4mm/ tip width: 2.5mm)

**NOTE**

- Undertightening can cause cable disconnection or malfunction. Overtightening can cause a short circuit or malfunction due to damage to the screw or unit.

**Caution**

- After wiring, wire offcuts must not be left in the inverter. They may cause an error, failure or malfunction.

# 4

## INVERTER SETTINGS

The inverter parameters listed in Table 11 are critical for overall operation of the end-to-end communication system. Some of these parameters must be set to specific values, and some may have multiple allowable settings depending on the desired operation of the overall application. Although there may be many other inverter parameters that will require configuration for your specific application, it is important to understand the manner in which the following parameters will impact successful communications with, and control of the inverter.

**Table 11: Inverter Settings**

Parameter Number	Name	Refer to Page
79	Operation mode selection	38
338	Communication operation command source	41
339	Communication speed command source	41
340	Communication startup mode selection	38
342	Communication EEPROM write selection	42
349 <sup>*1</sup>	Communication reset selection	52
500 <sup>*1</sup>	Communication error execution waiting time	43
501 <sup>*1</sup>	Communication error occurrence count display	44
502	Stop mode selection at communication error	45
541 <sup>*1</sup>	Frequency command sign selection	-
550 <sup>*2</sup>	NET mode control source selection	41
779	Operation frequency during communication error	45
1303	Status code	31
1304	Run mode	32
1305	Protocol	32
1306	Address	33
1307	Baud rate	33
1308	Parity/Stop bits	34
1309	Timeout time	34
1310	Response delay	34

Parameter Number	Name	Refer to Page
1311	Number of retries	34
1315	Device instance number (Upper 3 digits)	35
1316	Device instance number (Lower 4 digits)	35
1317	Max master	35
1318	Max info frames	35
1319	Fail-safe timeout	35
1350	Firmware version	35

<sup>\*1</sup> Parameters which can be displayed when the plug-in option (A8NXLT) is mounted.

<sup>\*2</sup> The setting is applied after an inverter reset or power-ON.

## 4.1 Network Setting

The network settings can **optionally** be set using the inverter parameters described in this section. However, to avoid invalid configuration, it is recommended to use the Mitsubishi Network Parameter Utility software to modify the network settings whenever possible. Refer to section 6 for details on the NPU software.

### 4.1.1 Status Code (Pr. 1303)

This parameter displays the current status of the option board. Table 12 details the status code values.

**Table 12: Status Code Values**

Value	Status
0	Normal
6	USB to Serial Pass-Through Mode
7	Invalid configuration parameters
65535 (0xFFFF)	Network Communication Error
All other values	Internal Error, contact Mitsubishi for assistance



### 4.1.2 Run Mode (Pr. 1304)

This parameter displays the current running mode of the option board.

Table 13: Run Mode Values

Value	Mode
0	Startup
1	Synchronizing Drive Parameters
2	Running
3	Error

### 4.1.3 Protocol (Pr. 1305)

This parameter selects the active RS-485 protocol.

Table 14: Protocol Values

Value	Protocol
0	RS-485 Port Disabled
1	BACnet MS/TP
2	Modbus RTU
3	Metasys N2
4	Siemens FLN
5	DMX-512
6	Generic Serial

#### 4.1.4 Address (Pr. 1306)

This parameter configures the RS-485 slave address. The valid range is dependent upon the selected protocol.

**Table 15: Protocol Address Ranges**

Protocol	Address Range
BACnet MS/TP	0...127
Modbus RTU	1...247
Metasys N2	1...255
Siemens FLN	0...98
DMX-512	1...512
Generic Serial	0...65535

#### 4.1.5 Baud Rate (Pr. 1307)

This parameter configures the RS-485 baud rate. The value is equal to the actual baud rate divided by 100. For example, for 9600 baud, this parameter should be set to a value of 96. The valid range is dependent upon the selected protocol.

**Table 16: Baud Rate Setting Ranges**

Protocol	Baud Rate Setting Range
BACnet MS/TP	96...1152
Modbus RTU	24...1152
Metasys N2	96
Siemens FLN	24...1152
DMX-512	2500
Generic Serial	24...1152

#### 4.1.6 Parity/Stop Bits (Pr. 1308)

This parameter configures the RS-485 parity and number of stop bits. This setting only applies when Modbus RTU or Generic Serial is selected as the active protocol. All other protocols use a fixed value.

Table 17: Parity/Stop Bits Values

Value	Mode
0	No Parity, 1 Stop Bit
1	Odd Parity, 1 Stop Bit
2	Even Parity, 1 Stop Bit
3	No Parity, 2 Stop Bits

#### 4.1.7 Timeout Time (Pr. 1309)

This parameter defines the RS-485 timeout time in milliseconds for a break in network communications before the option board will detect a timeout condition. This parameter defines the APDU Timeout when BACnet MS/TP is selected as the active protocol. The Fail-safe Timeout Time (Pr. 1319) defines the timeout detection time for BACnet MS/TP.

#### 4.1.8 Response Delay (Pr. 1310)

This parameter defines the time in milliseconds that the option board will wait before responding to master requests. This setting is ignored when BACnet MS/TP or DMX-512 is selected as the active protocol.

#### 4.1.9 Number of Retries (Pr. 1311)

This parameter sets the number of times that the option board will retry a request when a response is not received. This setting only applies when BACnet MS/TP is selected as the active protocol. The valid range is 0 to 10.

#### **4.1.10 Device Instance Number (Upper 3 Digits) (Pr. 1315)**

This parameter configures the upper 3 decimal digits of the option board's BACnet MS/TP device instance number. The valid range for the device instance number is 0 to 4194302, therefore the valid range for this parameter is 0 to 419.

#### **4.1.11 Device Instance Number (Lower 4 Digits) (Pr. 1316)**

This parameter configures the lower 4 decimal digits of the option board's BACnet MS/TP device instance number. The valid range for the device instance number is 0 to 4194302.

#### **4.1.12 Max Master (Pr. 1317)**

This parameter configures option board's BACnet MS/TP max master. This defines the highest allowable address for MS/TP master nodes on the network. Any address higher than this will not receive the token from the option board. The valid range is 0 to 127.

#### **4.1.13 Max Info Frames (Pr. 1318)**

This parameter configures the option board's BACnet MS/TP max info frames. This defines the maximum number of information frames the device may send before it must pass the token. The valid range is 1 to 100.

#### **4.1.14 Fail-safe Timeout Time (Pr. 1319)**

This parameter defines the BACnet MS/TP timeout time in milliseconds for a break in network communications before the option board will detect a timeout condition.

Note that due to the nature of BACnet MS/TP, there is always token passing traffic which does not necessarily indicate the presence of a client. Therefore, "network communications" in the context of timeout processing is defined as the device receiving BACnet Confirmed Request PDUs from the client.

#### **4.1.15 Firmware Version (Pr. 1350)**

This parameter displays the firmware version of the option board. The value is equal to the actual firmware version multiplied by 1000. For example, a firmware version of 1.023 will show a value of 1023.



- The inverter must be power cycled after changing any option board parameters for the settings to take effect.
- If an option board parameter is set to an invalid value, the option board will automatically default the value for that parameter.
- The inverter must be set to Network operation mode or configured to allow parameters to be written from the option board in order for the option board to modify parameter values.
- When using the Network Parameter Utility to configure the option board parameters, these parameters will be updated to reflect the values configured in the utility.

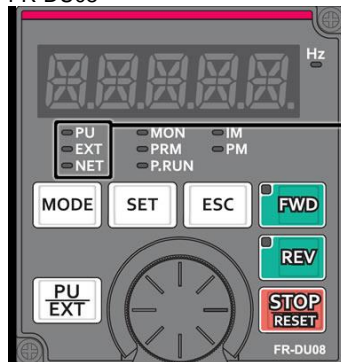
## 4.2 Operation Mode Setting

Three operation modes are available when a communication option card is installed into an inverter.

1. PU operation [PU] ..... The inverter is controlled by the operating panel (FR-DU08).
2. External operation [EXT] ..... The inverter is controlled by the ON/OFF switching of external signals connected to the control circuit terminals (factory default.)
3. Network operation [NET] ..... The inverter is controlled from the network via the communication option card (the operating commands and frequency command can be input via the control circuit terminals depending on the settings of *Pr. 338 Communication operation command source* and *Pr. 339 Communication speed command source*. Refer to page 41.)

### 4.2.1 Operation Mode Indication

FR-DU08



Operation mode indication (the inverter operates in accordance with the indicated LED.)

PU: PU operation mode

EXT: External operation mode

NET: Network operation mode

## 4.2.2 Operation Mode Switching & Comm. Startup Mode (Pr. 79, Pr. 340)

### ◆ Operation mode switching conditions

Prior to switching the operation mode, confirm that:

- The inverter is stopped
- Both the STF and STR signals are off
- The *Pr. 79 Operation mode selection* setting is correct. Refer to the appropriate inverter *user's manual (applied)* for further information regarding *Pr. 79*.

### ◆ Operation mode selection at power on and after recovery from a momentary power failure

The operation mode at power on and after recovery from a momentary power failure can be selected via *Pr. 340*. A value other than "0" will select network operation mode. After activating network operation mode, parameter writes from the network are enabled.




- When *Pr. 340* is changed, the new setting is validated after powering on or resetting the inverter.
- *Pr. 340* can be changed via the operation panel regardless of the operation mode.
- When setting a value other than 0 in *Pr. 340*, confirm that the initial settings of the inverter are correct.
- Refer to the inverter's *Instruction Manual (Detailed)* for additional information regarding *Pr. 79* and *Pr. 340*.

<b>Pr. 340 Setting</b>	<b>Pr. 79 Setting</b>	<b>Operation Mode at Power-On or Power Recovery</b>	<b>Operation Mode Switchover</b>
0 (default)	0 (default)	External operation mode	Switching among external, PU, and NET operation modes is enabled <sup>*1</sup>
	1	PU operation mode	PU operation mode fixed
	2	External operation mode	Switching between external and NET operation modes is enabled, switching to PU operation mode is disallowed
	3, 4	External/PU combined operation mode	Operation mode switching is disallowed
	6	External operation mode	Switching among external, PU, and NET operation modes is enabled while running
	7	X12 (MRS) signal ON...external operation mode	Switching among external, PU, and NET operation modes is enabled <sup>*1</sup>
		X12 (MRS) signal OFF...external operation mode	External operation mode fixed (forcibly switched to external operation mode)
1, 2 <sup>*2</sup>	0	NET operation mode	Same as when Pr. 340 = "0"
	1	PU operation mode	
	2	NET operation mode	
	3, 4	External/PU combined operation mode	
	6	NET operation mode	
	7	X12 (MRS) signal ON...NET operation mode	
		X12 (MRS) signal OFF...external operation mode	
10, 12 <sup>*2</sup>	0	NET operation mode	Switching between PU and NET operation modes is enabled <sup>*3</sup>
	1	PU operation mode	Same as when Pr. 340 = "0"
	2	NET operation mode	NET operation mode fixed
	3, 4	External/PU combined operation mode	Same as when Pr. 340 = "0"
	6	NET operation mode	Switching between PU and NET operation modes is enabled while running <sup>*3</sup>
	7	External operation mode	Same as when Pr. 340 = "0"



<sup>\*1</sup> The operation mode cannot be directly changed between PU mode and NET mode.

<sup>\*2</sup> *Pr. 340* settings "2" and "12" are mainly used for communication operation using the inverter's RS-485 port. When a value other than "9999" (automatic restart after momentary power failure) is set in *Pr. 57 Restart coasting time*, the inverter will resume the same operation state which it was in prior to a momentary power failure if such a failure occurs. When *Pr. 340* is set to "1" or "10" and a start command is active, then the start command will be deactivated if a momentary power failure occurs.

<sup>\*3</sup> The operation mode can be changed between PU mode and NET mode with the  key on the operating panel (FR-DU08) and X65 signal.

### 4.3 Operation & Speed Command Source (Pr. 338, Pr. 339, Pr. 550)

Refer to the inverter's *user's manual (applied)* for further details.

Parameter Number	Name	Default Value	Setting Range	Description
338	Communication operation command source	0	0	Communication option card is the run command source (A8NXLT run command is valid.)
			1	External input is the run command source. (A8NXLT run command is invalid.)
339	Communication speed command source	0	0	Communication option card is the speed command source (A8NXLT speed command is valid.)
			1	External input is the speed command source. (A8NXLT speed command is invalid.)
			2	External input is the speed command source. (If there is no external input, the frequency command via communication is valid, and the frequency command from terminal 2 is invalid.)
550	NET mode operation command source selection	9999	0	Communication option card control is valid (A8NXLT control is valid.)
			1	Control via the inverter's RS-485 port (A8NXLT control is invalid.)
			9999	Communication option automatic recognition. Normally, control via the inverter's RS-485 port is valid. When an A8NXLT communication option card is installed, that communication option card's control is made valid instead of the inverter's RS-485 port.

## 4.4 Communication EEPROM Write Selection (Pr. 342)

When parameters are written via communications, by default both volatile RAM and nonvolatile EEPROM contents are modified. Due to the limited write cycle lifetime of EEPROM memory, however, it may be desirable to modify only the contents of RAM when frequent parameter writes via communications are necessary.

Parameter Number	Name	Default Value	Setting Range	Description
342	Communication EEPROM write selection	0	0	Parameter values modified via communications are written to both EEPROM and RAM.
			1	Parameter values modified via communications are written only to RAM.

When frequently modifying parameter values via communications, change the value of *Pr. 342* to a "1" in order to write them only to RAM. Performing frequent parameter writes to EEPROM will shorten the lifetime of the component.



- When Pr. 342 is set to a value of "1" (write to RAM only), powering off the inverter will erase the changed parameter values. Therefore, the parameter values available when power is switched on again are those that were previously stored in EEPROM.

## 4.5 Operation at communication error occurrence

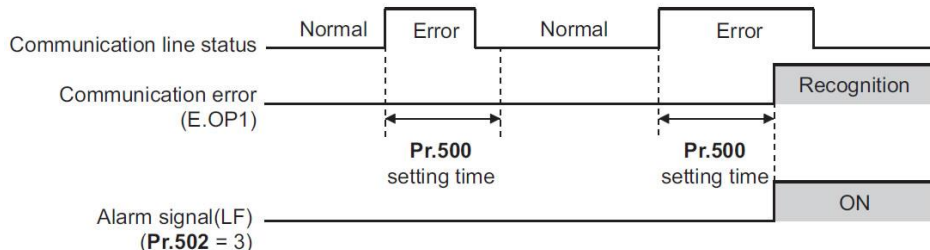
### 4.5.1 Operation selection at communication error occurrence (Pr. 500 to Pr. 502, Pr. 779)

You can select operations at communication error occurrences by setting Pr. 500 to Pr. 502, Pr. 779 under network operation.

#### ◆ Waiting time for the communication line error output after a communication error

Waiting time for the communication error output after a communication line error occurrence can be set.

Parameter Number	Name	Setting Range	Minimum setting increments	Initial Value
500	Communication error execution waiting time	0 to 999.8 s	0.1 s	0 s

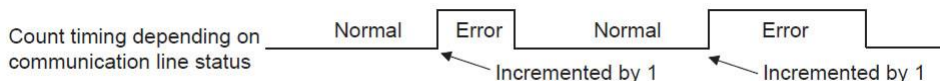


When a communication line error occurs and lasts longer than the time set in Pr. 500, it is recognized as a communication error. If the communication returns to normal within the time, it is not recognized as a communication error, and the operation continues.

#### ◆ Displaying and clearing the communication error count

The cumulative count of communication error occurrences can be displayed. Write "0" to clear this cumulative count.

Parameter Number	Name	Setting Range	Minimum setting increments	Initial Value
501	Communication error occurrence count display	0	1	0



At the point of communication line error occurrence, *Pr. 501 Communication error occurrence count display* is incremented by 1.

#### NOTE

- Communication error count is temporarily stored in the RAM memory. The error count is stored in EEPROM only once per hour. If power reset or converter reset is performed, *Pr. 501* setting will be the one that is last stored to EEPROM depending on the reset timing.

◆ **Inverter operation at a communication error occurrence**

How the inverter operates at a communication line error or an option unit fault can be set.

Parameter Number	Name	Setting Range	Description
502	Stop mode selection at communication error	0 (Initial Value), 1, 2, 3	Refer to page 46.
779 <sup>*1</sup>	Operation frequency during communication error	0 to 590 Hz	When a communication error occurs, the inverter operates at the set frequency.
		9999 (Initial Value)	The inverter operates at the frequency set before the communication error occurs.

<sup>\*1</sup> Valid when *Pr. 502* = "3".

◆ **About setting**

- Operation at an error occurrence

Error definition	Pr. 502 setting	Operation	Indication	Fault output
Communication line	0	Continued *1	Normal indication *1	Not provided *1
	1			
	2			
	3			
Communication option itself	0, 3	Coast to stop	E. 1 lit	Provided
	1, 2	Decelerated to stop	E. 1 lit after stop	Provided after stop

\*1 When the communication returns to normal within the time period set in *Pr. 500*, the communication option error (E.OP1) does not occur.

- Operation after the time in *Pr. 500* elapses after an error occurrence

Error definition	Pr. 502 setting	Operation	Indication	Fault output
Communication line	0	Coast to stop	E.OP1 lit	Provided
	1	Decelerated to stop	E.OP1 lit after stop	Provided after stop
	2			Not provided
	3	Continues operation with the <i>Pr. 779</i> setting	Normal indication	
Communication option itself	0, 3	Kept stopped <sup>*2</sup>	E.OP1 kept lit <sup>*2</sup>	Kept provided <sup>*2</sup>
	1, 2			

<sup>\*2</sup> When an error occurs, the motor is decelerated or coasts to stop, and outputs the fault, independently of the *Pr. 500* setting.



- Operation at error removal

Error definition	Pr. 502 setting	Operation	Indication	Fault output
Communication line	0	Kept stopped	E.OP1 kept lit	Kept provided
	1			
	2	Restart	Normal indication	Not provided
	3	Normal operation		
Communication option itself	0, 3	Kept stopped	E. 1 kept lit	Kept provided
	1, 2			



- The protective function [E.OP1 (fault data: HA1)] is activated at error occurrences on the communication line. The protective function [E.1 (fault data: HF1)] is activated at error occurrences in the communication circuit inside the option.
- Fault output indicates the fault (ALM) signal and fault bit output.
- When the fault output setting is active, fault records are stored in the faults history. (A fault record is written to the faults history at a fault output.) When the fault output setting is not active, fault record is overwritten to the faults history temporarily but not stored. After the error is removed, the fault indication is reset, changing the display back to normal, and the last fault is displayed in the faults history.
- When the *Pr. 502* setting is "1" or "2", the deceleration time is the ordinary deceleration time setting (e.g. *Pr. 8*, *Pr. 44*, *Pr. 45*).
- The acceleration time at a restart is the ordinary acceleration time setting (e.g. *Pr. 7*, *Pr. 44*).
- When the *Pr. 502* setting is "2", the operation/speed command at a restart is the one given before the error occurrence.

- When a communication line error occurs at the *Pr. 502* setting of "2", removing the error during deceleration causes acceleration to restart at that point. (Acceleration is not restarted if the error is that of the option unit itself.)
- When *Pr. 502* = "3", in order to continue the operation in the case of a communication line error, provide a safety stop countermeasure other than via communication. One counter measure is to input a signal to the terminal RES.

## 4.5.2 Fault and measures

### ◆ Inverter operation in each operation mode at error occurrences

Location	Status		Operation mode		
			Network operation	External operation	PU operation
Inverter	Inverter operation		Inverter trip	Inverter trip	Inverter trip
	Data communication		Continued	Continued	Continued
Communication line	Inverter operation		Inverter trip <sup>*1</sup>	Continued	Continued
	Data communication		Stop	Stop	Stop
Communication option	Communication option connection error	Inverter operation	Inverter trip <sup>*1</sup>	Inverter trip <sup>*1</sup>	Inverter trip <sup>*1</sup>
		Data communication	Continued	Continued	Continued
	Error of communication option itself	Inverter operation	Inverter trip <sup>*1</sup>	Continued	Continued
		Data communication	Stop	Stop	Stop

<sup>\*1</sup> Depends on the *Pr. 502* setting

◆ Measures at error occurrences

Fault indication	Error definition	Measures
E.OP1	Communication line error	<p>Check the LED status of the option unit and remove the cause of the alarm. (Refer to page 20 for LED indication status)</p> <ul style="list-style-type: none"> <li>• Check the other nodes on the network.</li> <li>• Inspect the master.</li> </ul>
E.1, E.2, E.3	Option fault	<ul style="list-style-type: none"> <li>• Insert the communication option to the inverter option connector 1.</li> <li>• Check the connection between the inverter and option unit for poor contact, etc. and remove the cause of the error.</li> </ul>

\*1 When faults other than the above are displayed, refer to the Instruction Manual (Detailed) of the inverter and remove the cause of the error.

## 4.6 Inverter reset

### ◆ Operation conditions of inverter reset

Which resetting method is allowed or not allowed in each operation mode is described below.

Resetting method			Operation mode		
			Network operation	External operation	PU operation
Reset from the network	Inverter reset (Refer to page 65) <sup>*1</sup>		Allowed	Disallowed	Disallowed
	Option card reset <sup>*2</sup>		Allowed	Disallowed	Disallowed
	Error reset at inverter fault (Refer to page 64) <sup>*3</sup>	Pr. 349 = 0	Allowed	Allowed	Allowed
		Pr. 349 = 1		Disallowed	Disallowed
Turn on the inverter RES signal (terminal RES)			Allowed	Allowed	Allowed
Switch off inverter power			Allowed	Allowed	Allowed
Reset from the PU/DU	Inverter reset		Allowed	Allowed	Allowed
	Reset at inverter fault		Allowed	Allowed	Allowed

<sup>\*1</sup> Inverter reset can be made any time.

<sup>\*2</sup> Option card reset can be made any time.

<sup>\*3</sup> Reset can be made only when the protective function of the inverter is activated.



- When a communication line error has occurred, reset cannot be made from the network.
- The inverter is set to the External operation mode if it has been reset in Network operation mode in the initial status. To resume the network operation, the inverter must be switched to the Network operation mode again. Set a value other than "0" in *Pr. 340* to start in the Network operation mode. (Refer to page 37.)
- The inverter cannot be controlled for about 1 s after release of a reset command.

◆ **Error reset operation selection at inverter fault**

An error reset command from communication option can be invalid in the External operation mode or PU operation mode. Use Bit 11 of the inverter command register (register 1) for error reset commands via the network. (Refer to page 64.)

Parameter Number	Name	Default Value	Setting Range	Description
349	Communication reset selection	0	0	Error reset is enabled independently of operation mode
			1	Error reset is enabled only in the network operation mode

## **5.1 Parameter Synchronization**

---

The option card uses an internal mirroring memory to cache all of the inverter's parameters. This provides quick and deterministic response times on the RS-485 network and also allows the option card's network driver to be decoupled from inverter communications. However, one of the principle disadvantages of parameter caching is that write data checking is not available. This means that when the value of a register is modified via a network protocol, the interface card itself is not able to determine if the new value will be accepted by the inverter (the value may be out-of-range, or the inverter may be in a state in which it will not accept new values being written via communications, etc.) For example, if a write is performed to a command register with a data value that is out-of-range, the interface card will not generate a corresponding error. However, the register can be read over the network at a later time to confirm whether or not that the written value "took hold" in the inverter.

Even if an inverter parameter corresponding to a given register does not exist, the interface card still maintains a placeholder location in its internal mirroring memory for that register. This feature allows for the block access of non-contiguous registers as described in section 5.2.

The inverter provides two methods to synchronize parameters with the option card. A limited number of parameters can be synchronized using high-speed, cyclic access. This method provides very fast parameter synchronization by using a block of command data and a block of monitor data that is exchanged between the inverter and option card every communication cycle. Registers that are synchronized using this method are referred to as "cyclic" registers.

The other method of synchronization is performed on a request-response basis, providing a single read or write transaction per communication cycle. Synchronization of these parameters is much slower than cyclic registers and depends on the total number of parameters. Registers that are synchronized using this method are referred to as "scanned" registers.

### 5.1.1 Cyclic Registers

The registers shown in Table 18 are synchronized using the high-speed, cyclic method.

**Table 18: Cyclic Register List**

Register	Description
1	Command register
201	Output frequency
202	Output current
203	Output voltage
205	Frequency setting value
207	Motor torque
208	Converter output voltage
213	Input power
214	Output power
215	Input terminal status
216	Output terminal status
217	Load meter
218	Motor excitation current
252	PID set point
253	PID measured value
254	PID deviation value
1226	DriveControl
1227	DriveStatus (lower 16 bits)
1228	DriveStatus (upper 16 bits)
1229	SetpointSpeed
1230	CommandSpeed
1231	ActualSpeed



### **5.1.2 Scanned Registers**

All non-cyclic registers are constantly being “scanned” by the interface card, which is to say that they are constantly being read and/or written (as applicable) in a round-robin fashion using a write-first approach (i.e. writes have priority over reads and will be written immediately after detection).

The option card incorporates an optimized method of synchronizing higher priority “process” parameters and lower priority “configuration” parameters. Registers 2000 to 4096 are considered configuration parameters. All other parameters are considered process parameters. One exception to this is the PLC function user parameters located at registers 3150 to 3199. These parameters can optionally be included as process parameters using the Configuration Studio. Additionally, priority tuning of the process parameters can also be adjusted using the Configuration Studio. Refer to section 7.3.1 for details on these settings.

## 5.2 Register Numbers

All accessible inverter parameters are referenced by their register number as defined in Table 19 and can be conveniently referenced in the Configuration Studio (section 7). Note that the register list is not exhaustive, the registers may not exist for all inverters, and the register data contents may vary depending on the inverter. The register numbers are used when accessing and configuring registers via an RS-485 protocol. Information regarding several commonly used registers is included in this manual for user convenience. Registers 5 to 7 are documented in the inverter user manual “Mitsubishi inverter protocol” section. Registers 201 to 449 are documented in the inverter user manual “List of inverter monitored items / command items”. For information regarding the inverter parameters, registers 2000 to 4096, refer to the inverter user manual “Parameter List”. Registers 1500 to 1999 are general purpose data registers that may be used in conjunction with internal database logic (section 7.9) configured on the option card.

Note that not all of the available registers that exist in the interface card's register map have corresponding parameters that exist in the inverter. In other words, if a read from or write to a register that does not correspond to an existing inverter register/parameter takes place, the read/write may be successful (depending on the specific register accessed; refer to section 5.1), but the data will have no meaning. This feature is beneficial in situations where the accessing of non-contiguous registers can be made more efficient by accessing an all-inclusive block of registers (some of which correspond to inverter parameters and some of which do not), while only manipulating those in your local programming that are known to exist.

**Table 19: Register Parameter List**

Register	Description
1	Command register (refer to section 5.3)
5	Inverter reset (refer to section 5.4)
6	Alarm history clear (refer to section 5.5)
7	All parameter clear (refer to section 5.6)
102	Inverter state (refer to section 5.7)
103	Process parameters scan cycle time (refer to section 5.8)
104	Configuration parameters scan cycle time (refer to section 5.9)
105	Number of process parameters (refer to section 5.10)
106	Number of configuration parameters (refer to section 5.11)
201	Output frequency
202	Output current
203	Output voltage
205	Frequency setting value
206	Motor speed
207	Motor torque
208	Converter output voltage
209	Regenerative brake duty
210	Electronic thermal relay function load factor
211	Output current peak value
212	Converter output voltage peak value
213	Input power
214	Output power
215	Input terminal status
216	Output terminal status
217	Load meter
218	Motor excitation current
219	Position pulse

Register	Description
220	Cumulative energization time
222	Orientation status
223	Actual operation time
224	Motor load factor
225	Cumulative power
226	Position command (lower 16 bits)
227	Position command (upper 16 bits)
228	Current position (lower 16 bits)
229	Current position (upper 16 bits)
230	Drop pulse (lower 16 bits)
231	Drop pulse (upper 16 bits)
232	Torque order
233	Torque current order
234	Motor output
235	Feedback pulse
238	Trace status
240	PLC function user monitor 1
241	PLC function user monitor 2
243	PLC function user monitor 3
246	Motor temperature
250	Power saving effect
251	Cumulative saving power
252	PID set point
253	PID measured value
254	PID deviation value
258	Option input terminal status1
259	Option input terminal status2
260	Option output terminal status

Register	Description
261	Motor thermal load factor
262	Transistor thermal load factor
264	PTC thermistor resistance
265	Output power (with regenerative display)
266	Cumulative regenerative power
267	PID measured value 2
268	Second PID set point
269	Second PID measured value
270	Second PID deviation
271	Cumulative pulse
272	Cumulative pulse carrying-over times
273	Cumulative pulse (control terminal option)
274	Cumulative pulse carrying-over times (control terminal option)
280	Integrated power on time
281	Running time
282	Saving energy monitor
284	Fault code 1 – Most recent fault (refer to section 5.15)
285	Fault code 2 (refer to section 5.15)
286	Fault code 3 (refer to section 5.15)
287	Fault code 4 (refer to section 5.15)
288	Fault code 5 (refer to section 5.15)
289	Fault code 6 (refer to section 5.15)
290	Fault code 7 (refer to section 5.15)
291	Fault code 8 – Least recent fault (refer to section 5.15)
295	Second PID measured value 2
296	Second PID manipulated variable
300	Current position 2 (lower 16 bits)
301	Current position 2 (upper 16 bits)

Register	Description
302	PID manipulated variable
449	Run command
1224	DriveControlMaskWrite (lower 16 bits)
1225	DriveControlMaskWrite (upper 16 bits)
1226	DriveControl
1227	DriveStatus (lower 16 bits) (refer to section 5.12)
1228	DriveStatus (upper 16 bits)
1229	SetpointSpeed (refer to section 5.13)
1230	CommandSpeed
1231	ActualSpeed (refer to section 5.14)
1232	SpeedScaleNumerator (lower 16 bits)
1233	SpeedScaleNumerator (upper 16 bits)
1234	SpeedScaleDenominator (lower 16 bits)
1235	SpeedScaleDenominator (upper 16 bits)
1236	RatedSpeed
1237	PoleCount
1238	RatedCurrent (lower 16 bits)
1239	RatedCurrent (upper 16 bits)
1240	RatedVoltage
1241	MotorType
1242	DriveMode
1243	SupportedModes (lower 16 bits)
1244	SupportedModes (upper 16 bits)
1245	AccelerationDeltaSpeed (lower 16 bits)
1246	AccelerationDeltaSpeed (upper 16 bits)
1247	AccelerationDeltaTime (lower 16 bits)
1248	AccelerationDeltaTime (upper 16 bits)
1249	DecelerationDeltaSpeed (lower 16 bits)

Register	Description
1250	DecelerationDeltaSpeed (upper 16 bits)
1251	DecelerationDeltaTime (lower 16 bits)
1252	DecelerationDeltaTime (upper 16 bits)
1253	QuickDecelerationDeltaSpeed (lower 16 bits)
1254	QuickDecelerationDeltaSpeed (upper 16 bits)
1255	QuickDecelerationDeltaTime (lower 16 bits)
1256	QuickDecelerationDeltaTime (upper 16 bits)
1257	MaxSpeed (lower 16 bits)
1258	MaxSpeed (upper 16 bits)
1259	MinSpeed (lower 16 bits)
1260	MinSpeed (upper 16 bits)
1261	TargetTorque
1262	ActualTorque
1263	TorqueSlope (lower 16 bits)
1264	TorqueSlope (upper 16 bits)
1265	TorqueProfileType
1266	RatedTorque (lower 16 bits)
1267	RatedTorque (upper 16 bits)
1268	TorqueScaleNumerator (lower 16 bits)
1269	TorqueScaleNumerator (upper 16 bits)
1270	TorqueScaleDenominator (lower 16 bits)
1271	TorqueScaleDenominator (upper 16 bits)
1272	DisableOptionCode
1273	ShutdownOptionCode
1500-1599	General purpose registers
2000-4096	<i>Pr. 0</i> and higher. To calculate the register number, add 2000 to the parameter number. For example, <i>Pr. 123</i> is register 2123 (123 + 2000).
4097	Product ID

Register	Description
4098	Firmware Version
4099	Status Code
4100	Run Mode
4101	Protocol
4102	Address
4103	Baud Rate
4104	Parity
4105	Timeout
4106	Response Delay
4107	Number of Retries
4121	Device Instance Lo
4122	Device Instance Hi
4123	Max Master
4126	Max Info Frames



## 5.3 Inverter Command Register

The command word is register 1 and the bit-mapping is described in Table 20. These bits are mapped to DriveControl (register 1226).

Table 20: Inverter Command Register Mapping

Bit	Significance	Value	Description
0	Reserved	Not used	-
1	EnableOp	1	Enable operation
		0	Disable operation
2	EnableRevOp	1	Enable reverse operation
		0	Disable reverse operation
3	QuickStop	1	Enable quick stop
		0	Disable quick stop
4 - 10	Reserved	Not used	-
11	ResetMalfunction	1	Reset fault condition on a positive edge (0→1 transition)
		0	-
12 - 13	Reserved	Not used	-
14	NetRef	1	Request torque/speed reference from network
		0	Request local reference
15	NetCtrl	1	Request drive control from network
		0	Request local drive control

## 5.4 Inverter Reset Register

The inverter reset register is register 5. A value of 0x9696 or 0x9966 will reset the inverter.

## 5.5 Alarm History Clear Register

The alarm history clear is register 6. A value of 0x9696 will clear the alarm history.

## 5.6 All Parameter Clear Register

The all parameter clear is register 7. Refer to Table 21 for the appropriate value.

**Table 21: All Parameter Clear Register**

Pr. Data	Communi- cation Pr. *1	Calibration Pr. *2	Other Pr. *3	HEC HF3 HFF
H9696	○	×	○	○
H9966	○	○	○	○
H5A5A	×	×	○	○
H55AA	×	○	○	○

\*1 Refer to communication related parameters.

\*2 Refer to the list of calibration parameters.

\*3 Pr. 75 is not cleared.

## 5.7 Inverter State Register

The inverter state is register 102 and the states are listed in Table 22.

Table 22: Inverter State

Value	Description
0	NOT_READY_TO_SWITCH_ON
1	SWITCH_ON_DISABLED
2	READY_TO_SWITCH_ON
3	SWITCHED_ON
4	OPERATION_ENABLED
5	DISABLE
6	SHUTDOWN
7	QUICK_STOP
8	FAULT_REACTION_ACTIVE
9	FAULTED

## **5.8 Process Parameters Scan Cycle Time Register**

---

The process parameters scan cycle time is register 103. This is a measurement, in hundredths of seconds, of the last process parameter scan cycle time. This register can be used to assist in tuning process data scan priority. Note that when process parameter scan priority is increased, the update latency of process parameters decreases. However, the update latency of configuration parameters increases.

## **5.9 Configuration Parameters Scan Cycle Time Register**

---

The configuration parameters scan cycle time register is 104. This is a measurement, in hundredths of seconds, of the last configuration parameter scan cycle time. This register can be used to assist in tuning process data scan priority. Note that when process parameter scan priority is increased, the update latency of process parameters decreases. However, the update latency of configuration parameters increases.

## **5.10 Number of Process Parameters Register**

---

The number of process parameters is register 105. This register contains the number of process parameters that were synchronized during the last process parameter scan cycle. Because the option card uses an optimization algorithm to automatically detect the parameters that exist on the inverter, the number of process parameters for the first scan cycle will be the maximum number of possible parameters. For all subsequent scan cycles, this value will reflect the actual number of process parameters that exist on the inverter and are being synchronized.

## **5.11 Number of Configuration Parameters Register**

---

The number of configuration parameters is register 106. This register contains the number of configuration parameters that were synchronized during the last configuration parameter scan cycle. Because the option card uses an optimization algorithm to automatically detect the parameters that exist on the inverter, the number of configuration parameters for the first scan cycle will be the maximum number of possible parameters. For all subsequent scan cycles, this value will reflect the actual number of configuration parameters that exist on the inverter and are being synchronized.

## 5.12 DriveStatus Register

The DriveStatus is register 1227 and the bit-mapping is described in Table 23.

Table 23: DriveStatus Register Mapping

Bit	Significance	Value	Description
0	ReadyToSwitchOn	1	Power supply is switched on, electronics initialized, main contact, if available, has dropped out, pulses are inhibited
		0	-
1	SwitchedOn	1	Voltage at the power converter, i.e. the main contact is closed (if present)
		0	-
2	OpEnabled	1	Drive function enabled
		0	-
3	Faulted	1	Unacknowledged faults present
		0	-
4	VoltageEnabled	1	Power enabled
		0	Power disabled
5	QuickStopDisabled	1	Quick stop disabled
		0	Executing quick stop request
6	SwitchOnDisabled	1	Drive in SWITCH_ON_DISABLED
		0	-
7	Reserved	Not used	-
8	RefFromNet	1	Network torque/speed reference
		0	Local torque/speed reference
9	CtrlFromNet	1	Control is from network
		0	Control is local
10	SetpointReached	1	Actual output frequency equals frequency set point
		0	-

Bit	Significance	Value	Description
11	LimitActive	1	Internal (manufacturer specific) limitation is active (e.g. current, power or speed limit)
		0	-
12 - 13	Reserved	Not used	-
14	RevOpEnabled	1	Drive function enabled in reverse mode
		0	-
15	Stopping	1	Drive is stopping
		0	-

### 5.13 SetpointSpeed Register

The SetpointSpeed is register 1229 and specifies the target speed in RPM. A positive value indicates forward motion, while a negative value indicates reverse motion. The value must be scaled based on the SpeedScale factor. By default, the SetpointSpeed is specified in 0.1 RPM units.

### 5.14 ActualSpeed Register

The ActualSpeed is register 1231 and reflects the actual speed in RPM. A positive value indicates forward motion, while a negative value indicates reverse motion. The value is scaled based on the SpeedScale factor. By default, the ActualSpeed is specified in 0.1 RPM units.

### 5.15 Fault History Codes

The fault history codes are reflected in registers 284 (most recent fault) to 291 (least recent fault). The fault code is described in the inverter user manual "List of fault display" section.

## 6.1 Overview

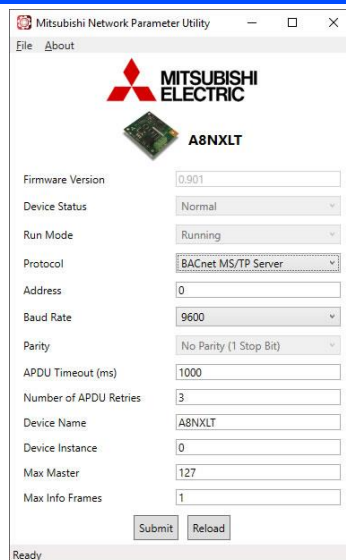


Figure 2: Mitsubishi Network Parameter Utility

The Network Parameter Utility (NPU) software is an end-user application intended to quickly and easily configure the inverter's parameters for the option card's network settings in the field via USB. The communication settings for the network-facing port on the option card are dependent upon the system in which the card is being installed. The NPU is designed to configure only these network communication settings. The latest release of the NPU can be downloaded from the [product web page](#).

## 6.2 Features

---

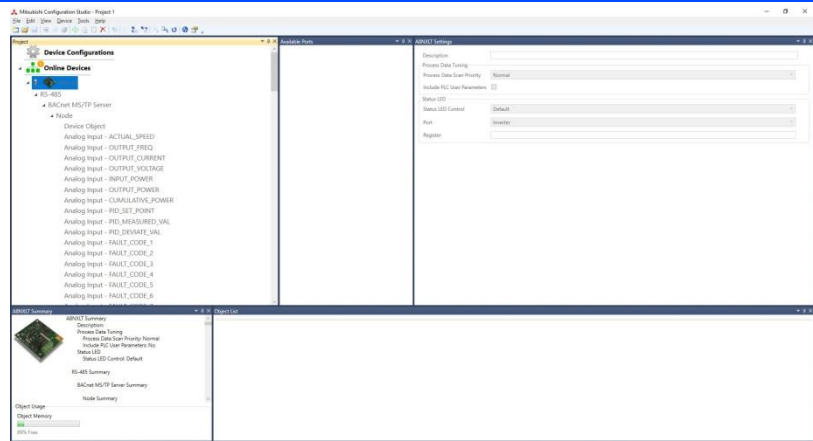
- Light-weight, portable application - no installation necessary, simply unzip and run the exe.
- Includes all necessary USB drivers and firmware files.
- Automatically discovers and connects to the device via USB.
- Allows configuration of standard, field-configurable network settings such as protocol, baud rate, parity, address, and other protocol-specific settings.
- Detects outdated firmware on the connected device and allows the user to update the firmware on the device to the latest version.
- Only shows those protocols and options actually configured on the connected device.
- Self-documented - each setting provides a useful tooltip to the user that describes the setting.



- The NPU software writes to the inverter's option parameters at *Pr. 1300* to *Pr. 1343*. Refer to section 4.1 for details.
- The option card must be installed on the inverter and the inverter must be powered on in order to use the NPU.
- The inverter must be configured to accept parameter writes from the option card. Refer to section 4.2 for details.



## **7.1 Overview**



**Figure 3: Mitsubishi Configuration Studio**

The card is discovered, configured and updated by the Mitsubishi Configuration Studio PC application. The studio must be installed prior to connecting a card to ensure that the appropriate USB drivers are installed. The studio requires a USB connection for reading/writing a configuration and updating the firmware. The latest release of the Configuration Studio can be downloaded from the [product web page](#). The remainder of this section will provide only a brief introduction to the configuration concepts. For protocol-specific configuration, refer to the relevant protocol section.

### Creating a Device Configuration

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

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

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

### Going Online with a Device

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

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

When the studio goes online with a device, its configuration is automatically read. While the studio is online with a device, it will appear in green text in the **Discovered Devices** panel. The studio may be online with multiple devices simultaneously.

### 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.

### Importing a Configuration from a Project File

An existing project file can be imported into the currently-active project. Click **File...Import Project**, and then select the desired \*.mcsproj file. The contents of the imported file will be merged with the active project.

### 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.

### 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**.

### Interacting with the Database

To interact with a device's database, select the device in the **Project** panel and then select the **Database** panel. If the **Database** panel is not visible, it can be enabled via **View...Database**. When an online device is selected, data values are updated from the device in real-time, and values can be edited by double-clicking the desired location in the database.

### **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 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.

#### **7.3.1 Process Data Tuning**

These options tune how parameters are scanned during the device's asynchronous read cycles. There are two categories of scanned data, process parameters and configuration parameters. Process parameters are updated quickly while configuration parameters are updated slowly. Refer to section 5.1 for additional information.

### **Process Data Scan Priority**

Adjusts the priority applied to scanned process parameters during the device's read cycle. This adjusts the update latency balance between scanned process parameters and configuration parameters.

Note that when this is set to **High**, the update latency for scanned process parameters decreases while the update latency for configuration parameters increases.



### **Include PLC User Parameters**

Check this option to include the PLC function user parameters (Pr. 1150 - 1199) in the scanned process data. If unchecked, the PLC function user parameters will be scanned as configuration parameters.

## **7.3.2 Status LED Settings**

The device's status LED is software-controlled and can be configured to indicate a wide range of information to the user, providing useful insight into the operation of the device. These settings define the behavior of the device's status LED.

### **Status LED Control**

Selects the method for controlling the device's status LED. Regardless of the option selected, upon startup, the status LED will flash the green, red, green, red startup sequence. Additionally, if an internal error occurs, the status LED will always flash red indicating the error code.

#### **Default**

This option is the default behavior of the status LED. The LED is solid green when power is applied to the device. The LED flashes green when a USB connection has been established between the device and a PC.

#### **Port Activity**

This option allows the selected port's TX and RX activity to be indicated by the status LED. For each LED cycle, if the port has transmitted any bytes since the last cycle, the status LED will light green for half of the cycle. If the port has received any bytes since the last cycle, the status LED will light red for half of the cycle.

#### **Register Value**

This option configures the status LED to be fully controlled by a value located in the device's internal register database. This enables the status LED to be directly controlled via communications or by internal logic applied to data stored in the device's database. Table 24 lists the supported LED states.

**Table 24: Status LED States**

Value	LED State
0	Off
1	Green On
2	Red On
3	Green Flashing
4	Red Flashing
5...255	Off

#### **Port**

Selects the port for which the TX and RX activity will be indicated by the status LED.

Note that this option only applies when the Port Activity option is selected for the Status LED Control setting.

#### **Register**

Defines the inverter register number used to control the status LED.

Note that this option only applies when the Register Value option is selected for the Status LED Control setting.

## 7.4 USB Virtual COM Port Settings

---

The card can be configured to enumerate as a USB virtual COM port, providing direct serial communications between the card and a PC through the USB connection. The COM port can be used for various tasks, depending on the selected mode. This section details the different functions of the virtual COM port.

### Mode

Select the desired mode for how the USB virtual COM port will be used. The available options are detailed below.

#### Serial Pass-Through

Select this option to cause the card to behave as a USB to serial converter. Any data sent to the USB virtual COM port will be sent on the physical serial port and any data received by the physical serial port will be received from the USB virtual COM port. Note that while the card is in this mode all other functionality of the card is disabled, regardless of other configuration settings.

#### Serial Redirect

Select this option to redirect communications from the selected serial port to the USB virtual COM port. By selecting this option, the card will communicate with the PC over the virtual COM port using the settings configured on the associated serial port. This allows the card to communicate with the PC using any of the supported serial port protocols. Note that the physical serial port is disabled when the card is configured in this mode.

#### Serial Sniffer

Select this option to sniff the received and transmitted packets on the selected serial port and output the data to the virtual COM port. When this mode is selected, the card will attempt to output every packet that the protocol driver configured on the serial port receives and transmits.

Because the sniffer operates independently from the physical serial port (so as not to impact communications), there may be times when the sniffer cannot output a received or transmitted packet due to the USB connection being unable to output characters faster than they are exchanged on the physical serial port. When this occurs, the sniffer will output the characters "ERR: Sniffer Packet Overflow" or "ERR: Sniffer Buffer Overflow".

Additionally, the sniffer is able to detect receive errors on the serial port such as parity, overrun, and framing errors. If a receive error occurs on one or more characters of a packet, the sniffer will output the characters "ERR: Receive Error".

Note that because the serial sniffer mode captures packets at the protocol driver level, a protocol must be configured on the selected serial port to output data to the USB virtual COM port. For convenience, there is a special "USB Serial Sniffer Settings" protocol selection to configure the serial port for sniffing only.

### **Serial Port**

Select the desired serial port to target for use with the USB virtual COM port.

### **Sniffer Output Format**

Select the desired output format of the serial sniffer data. The formatted data option outputs the captured data as ASCII text characters and includes annotations for whether the packet was received or transmitted, as well as a relative timestamp of when the packet was received or transmitted. The raw data option outputs the captured data as unmodified, binary characters.

## ***7.5 USB Serial Capture Window***

---

The USB Serial Capture Window allows you to connect to a card's USB Virtual COM port to view and save network packets captured by the card. The card's USB Virtual COM port must be configured for Serial Sniffer mode and the Sniffer Output Format must be set to Formatted Data.

When connected, the capture window will display the card's most recent received and transmitted packets. All packets captured during the duration of the session may be saved once the session has ended, even though they all may not be displayed in the window. The status bar at the bottom of the window tracks the duration of the connection as well as the total number of packets the card has received and transmitted.

To open the USB Serial Capture Window, select **USB Serial Capture Window...** from the **Tools** menu.

### Capturing Packets

To begin capturing packets, the card must first be configured with the appropriate USB Virtual COM port settings as described above. Once configured, the card will appear in the **COM Port** selection box. Select the desired device from this drop down and connect to the device. To connect to the device, perform one of the following actions:

- Select **Connect** from the **Connection** menu.
- Click on the **Connect** button in the toolbar.

Note that connecting to a device will clear the capture log automatically.

### Clearing the Capture Log

All captured data may be cleared at any time while connected to a device or after disconnecting from a device. This will also reset the connection time duration and all counters. To reset all captured data, perform one of the following actions:

- Select **Clear Log** from the **Edit** menu.
- Click on the **Clear Log** button in the toolbar.
- Hit the <DELETE> key on the keyboard.
- Right click on the capture output and select **Clear Log**.

### Pausing the Display

While capturing, the output window will display only the most recent packets. Therefore, as new packets are captured and displayed in the window, old packets are removed from the display. At any time during capturing, the display updating may be paused so that no packets are added or removed. To pause the display, perform one of the following actions:

- Select **Pause Display** from the **Display** menu.
- Click on the **Pause Display** button in the toolbar.
- Right click on the capture output and select **Pause Display**.

Note that even though the display does not update when paused, packets are still being captured in the background.

### Ending a Capture Session

The capture session is ended by disconnecting from the selected device. To disconnect from the device, perform one of the following actions:

- Select **Disconnect** from the **Connection** menu.
- Click on the **Disconnect** button in the toolbar.

### Saving the Captured Data

Once a capture session has ended, the entire captured data may be saved. The data can be saved either as a Wireshark capture file or as a plain text document.

#### Wireshark Capture File

The captured data can be saved as a file which can be opened, decoded, and analyzed by Wireshark. Wireshark is a free network protocol analyzer and is available at <http://www.wireshark.org/>.

Any protocol capture may be viewed with Wireshark. However, Wireshark currently only supports decoding BACnet MS/TP packets and has limited support for Modbus RTU.

To save the captured data as a Wireshark capture file, perform one of the following actions:

- Select **Save As Wireshark Capture...** from the **File** menu.
- Click on the **Save As Wireshark Capture...** button in the toolbar.
- Hit the <CTRL+S> keys on the keyboard.

#### Text Document

The captured data can also be saved as a plain text document. To save the captured data as a text document, perform one of the following actions:

- Select **Save As Text...** from the **File** menu.
- Click on the **Save As Text...** button in the toolbar.
- Hit the <CTRL+SHIFT+S> keys on the keyboard.

## 7.6 Port Diagnostics

---

The Port Diagnostics window provides real-time monitoring of the number of bytes sent and received by the selected port. In addition, the Port Diagnostics also displays receive errors detected by the port.

## 7.7 Batch Update Mode

---

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

### Entering Batch Update Mode from within the Studio

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

### Launching the Studio in Batch Update Mode

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

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

## **Viewing the Batch Update Log**

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

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

### **Devices Discovered**

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

### **Successful**

The total number of devices that were updated successfully.

### **Failed**

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

### **Not Updated**

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

### **Firmware Updated**

The total number of firmware updates performed.

### **Configuration Updated**

The total number of configuration updates performed.

### **Errors**

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



## 7.8 Internal Logic Settings

---

### 7.8.1 Fail-safe Values

#### 7.8.1.1 Overview

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

There are two separate elements that comprise the timeout configuration:

- The timeout time
- Timeout Object configuration

### 7.8.2 Timeout Time

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

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

#### 7.8.2.1 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). 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.

### Register

Enter the register number (refer to section 5.2) corresponding to the inverter parameter.

### 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 register encompassed by this timeout object will be automatically written with upon processing a timeout event triggered by a protocol.

### Length

Enter the number of consecutive registers for this timeout object. Each register will be set to the value specified in the **Value** setting.

## **7.9 Database Logic**

---

### **7.9.1 Overview**

A variety of database logic operations are included which provide PLC-style manipulation of database values. Categories such as logical, arithmetic and filtering operations allow for autonomous control over value modification and data movement within the database. High-level signal conditioning is also realizable via the construction of compound formulas derived from the elemental building block operations provided. To add database logic operations to a device, select the device in the **Project** panel, then add **Internal Logic...Database Logic**.

Database logic operations are executed in sequential order, according to the ordinal position in which the operations are listed in the **Project** panel under the **Database Logic** heading.

Some notes of interest for the database logic operations are as follows:

## All Database Logic Operations

- All inputs to an operation may either be a value located in the internal database or a constant value.
- A floating-point “Multiplier” field is available on each database-sourced input and on the output which allows the inputs to be scaled prior to operation execution, and the result to be scaled after operation execution. The input is multiplied by the input multiplier, and the result is divided by the output multiplier.
- All operations can be dynamically enabled/disabled using an optional “Enable Trigger” element (refer to section 7.9.3 for more information on Enable Trigger behavior.)
- The outputs of all operations must be stored in the internal database.

## Logical Operations

- *Not*, *And*, *Or*, and *Exclusive Or* operations can be performed on either a bitwise or logical basis, depending on the selection of the “Operation Type”. When a logical operation type is chosen, non-zero input values are considered to be “true” and zero input values are considered to be “false”. The output value of the logical operation will then be written to the database as “1” for true and “0” for false.
- The *Copy* operation outputs the input value.
- The *Bit Copy* operation outputs the value of a single bit from the input database location to a single bit in the output database location. No other bits in the output database location are modified by this operation.
- The *Indirect Copy* operation outputs the value at the database location specified by the input source to the database location specified by the output destination. This operation can be used to access different database locations dynamically. It could also be used to create reusable database logic subroutines by selecting a different input and output location for the subroutine during each execution cycle.
- The *Shift* operation outputs the input value bit-shifted by the shift amount.
- The *Compare* operation outputs a “1” if the comparison evaluates to true, otherwise it outputs a “0”.
- The *Flag Test & Set* operation tests if the bit flags specified in the input mask are set in the input value and sets the bit flags specified in the output mask in the output value. This operation can test for ALL flags set/cleared or ANY flags set/cleared. If the flag test evaluates as true, all bit flags specified in the output mask in the output value are set, otherwise the flags are cleared. Only the bits specified in the output mask in the output value are modified by this operation.

- The *Value Change Detection* operation outputs a “1” if a change is detected in the input value between the last execution cycle and the current execution cycle, otherwise it outputs a “0”.
- The *Multiplexer* operation outputs one of its two inputs, depending on the selection. If *Selection* is zero, *Input 1* is output. If *Selection* is non-zero, *Input 2* is output.
- The *Byte Reverse* operation reverses the byte order of the input value and outputs the result.

### **Arithmetic Operations**

- The *Add* operation calculates the expression  $[Input\ 1] + [Input\ 2]$ .
- The *Subtract* operation calculates the expression  $[Input\ 1] - [Input\ 2]$ .
- The *Multiply* operation calculates the expression  $[Input\ 1] \times [Input\ 2]$ .
- The *Divide* operation calculates the expression  $[Input\ 1] / [Input\ 2]$ .
- The *Modulo* operation calculates the expression  $[Input\ 1] \bmod [Input\ 2]$ .
- The *Exponential* operation calculates the expression  $[Input\ 1]^{Exponent}$ . “Input 1” can be a database value, a constant value, or e (exponential function).
- The *Nth Root* operation calculates the expression  $\sqrt[Degree]{Input\ 1}$ .
- The *Logarithm* operation calculates the expression  $\log_{Base}(Input\ 1)$ . “Base” can be a database value, a constant value or e (natural logarithm).
- The *Random* operation outputs a random number between *Input 1* and *Input 2*. Note that the operation is limited to producing only 32,768 unique values.
- The *Divide*, *Exponential*, *Nth Root* and *Logarithm* operations output an integer-rounded value when an integer data type is used.

### **Trigonometric Operations**

- The *Sine* operation calculates the expression  $\sin(Input\ 1)$ , where *Input1* is in radians.
- The *Cosine* operation calculates the expression  $\cos(Input\ 1)$ , where *Input1* is in radians.
- The *Tangent* operation calculates the expression  $\tan(Input\ 1)$ , where *Input1* is in radians.
- The *Arc Sine* operation calculates the expression  $\sin^{-1}(Input\ 1)$ , where the output is in radians.

- The *Arc Cosine* operation calculates the expression  $\cos^{-1}(\text{Input } 1)$ , where the output is in radians.
- The *Arc Tangent* operation calculates the expression  $\tan^{-1}(\text{Input } 1)$ , where the output is in radians.

### **Filtering Operations**

- The *Debounce Filter* and *Hysteresis Filter* operations are functionally identical with the single exception that the *Debounce Filter* does not use a “Value Tolerance” (it is fixed at 0).
- In order for the output of the *Debounce Filter* or *Hysteresis Filter* to change (i.e. reflect the input value), “Input 1” must first change to a value outside of the “Value Tolerance” range and then remain within the “Value Tolerance” range of the new value for the entire “Stable Time”.

## **7.9.2 Database Logic Settings**

### **Scan Rate**

Defines the scan cycle time in milliseconds (50ms minimum) of the database logic processing task. All operations are evaluated for execution in sequential order at this frequency. Note that this does not necessarily mean that each operation is guaranteed to execute every scan cycle: only that it will be evaluated as to whether or not it should execute. Namely, if an “Enable Trigger” element is added to an operation, then the trigger must evaluate to “true” for the operation to execute during that scan cycle. Refer to section 7.9.3 for more information on Enable Trigger behavior.

### **7.9.3 Enable Trigger**

Each database logic operation can optionally include an “Enable Trigger” element, which provides dynamic conditional execution capabilities. By default (i.e. if an enable trigger element is not added to the operation), each operation is automatically triggered to execute every scan cycle. If it is desired for an operation to execute conditionally, however, then an enable trigger element can be added to it. The enable trigger element defines an “Enable Value”, which specifies a byte-size trigger value that can reside at any location in the internal database. When implemented, the enable value is evaluated every scan cycle: if this value is non-zero (or zero when the “Inverted” Trigger Option is used), the operation will execute.

The enable value itself can be modified by any communication driver currently running on the device, which enables networked devices to dynamically control the execution of database logic operations. The enable value can also be the output result of other database logic operations. While the output of any database operation can be used for this purpose, such a scenario may most typically use the output of a “compare” operation in order to control whether or not other

operations should execute (e.g. execute a certain operation only when some process variable is greater than a certain value, etc.) Allowing the conditional execution of database logic operations to be based on data values obtained via communications or as a result of other database logic operations enables the construction of flexible, hierarchical and dynamic data evaluation and manipulation engines.

### **Enable Value Register**

Specifies the register that will be used to enable the associated database logic operation.

#### **7.9.3.1 Trigger Options**

The enable trigger can perform basic logic on the enable value to determine if an operation should execute using a variety of trigger options. These setting determine what logic should be applied to the enable value when evaluating whether or not the operation should execute.

### **Inverted**

Specifies whether the enable logic should be inverted. This applies to both the evaluation of whether or not the operation should execute as well as resetting the enable value when the auto reset option is used.

### **Auto Reset**

Allows the enable value to be automatically reset upon completion of the operation. The actual value written to the enable value depends on the other trigger options selected. If no options are selected, a value of 0 is written to the enable value. If the inverted option is used, a value of 1 is written to the enable value. If the bitmask option is used, each bit selected in the bitmask is written to a 0 (or a 1 if the inverted option is used) in the enable value.

### **Bitmask**

If this option is used, it selects which bits in the enable value to evaluate. Every selected bit in the enable value must be 1 (or 0 when the inverted option is used) for the operation to execute.

## 7.10 Manage Device Parameters

The parameter number and register (communications number) of all available parameters can be viewed using the **Manage Device Parameters** window. This window can be 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...**

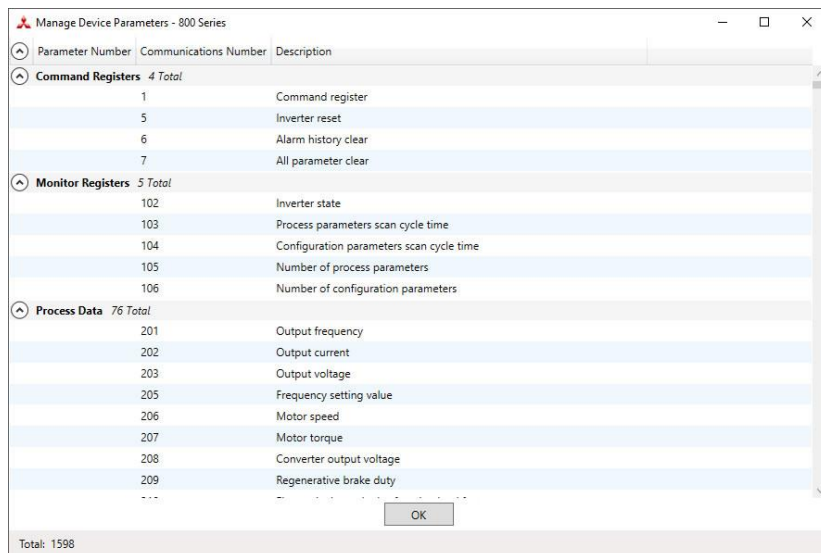


Figure 4: Manage Device Parameters

## 7.11 Monitor Device Parameters

The inverter's parameter values can be monitored and commanded in real-time (refer to Figure 5). 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.



Monitor Device Parameters - 800 Series

Radix: ☒ Decimal ☐ Hex

Data Type: 16-Bit Unsigned

Filter:  X

Parameter Number	Communications Number	Description	Value	Bits 15..0
<b>Command Registers</b> 4 Total				
<b>Monitor Registers</b> 1 Total				
<b>Process Data</b> 92 Total				
201		Output frequency	600	0000 0010 0101 1000
202		Output current	0	0000 0000 0000 0000
203		Output voltage	390	0000 0001 1000 0110
205		Frequency setting value	600	0000 0010 0101 1000
206		Motor speed	180	0000 0000 1011 0100
207		Motor torque	0	0000 0000 0000 0000
208		Converter output voltage	3398	0000 1101 0100 0110

Total: 1093

Connected

Figure 5: Monitor Device Parameters

## 7.12 Backup and Restore Parameters

The parameter values can be backed up from the inverter and restored to the inverter (refer to Figure 6 and Figure 7). This allows for easy inverter cloning. The backup parameter values are stored as a CSV file. A parameter value can be excluded by disabling the corresponding checkbox. The parameter value can also be modified before the backup and restore is executed. The backup and restore parameter configurations are found by:

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

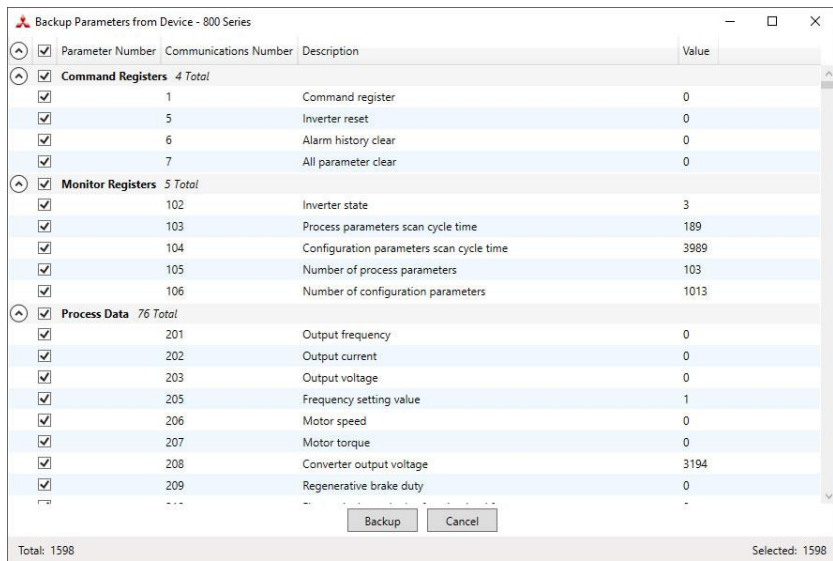
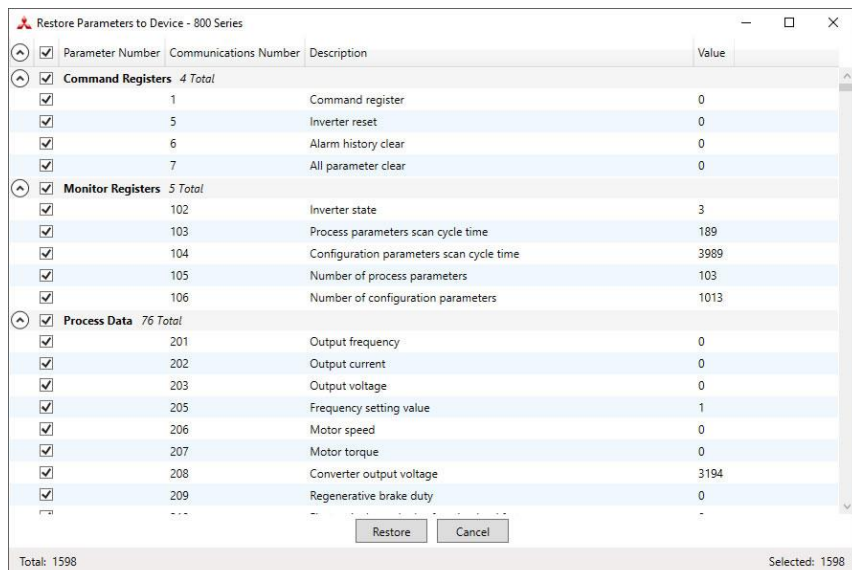


Figure 6: Backup Parameters



**Figure 7: Restore Parameters**

## 7.13 Restore Factory Settings

---

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

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

## 7.14 Help

---

For assistance in understanding configuration objects and fields, simply hover the mouse over the object or text to display a useful tooltip. Ensure that the **Help...Show Help Tooltips** option is checked. Links to videos and documents can be found in the **Help** menu. Please review the tooltips and links before contacting technical support for more in-depth assistance.



---

### 8.1 Overview

---

The interface card's embedded firmware can be updated in the field. Firmware updates may be released for a variety of reasons, such as custom firmware implementations, firmware improvements and added functionality as a result of user requests. Additionally, it may be necessary to load different firmware onto the unit in order to support various protocols. In order to ensure that the firmware update is successful, and in the interest of equipment and personnel safety, it is strongly recommended to stop all of the card's production activities prior to initiating the firmware update procedure. **Failure to follow the firmware update procedure could result in corrupt firmware!**

### 8.2 Update Procedure

---

Firmware update steps:

1. Always back up your configuration to a PC for later recovery if necessary.
2. Download and install the latest Configuration Studio or NPU software, which can be obtained from the [product web page](#).
3. Please be sure to read the firmware release notes and updated user's manual for any important notices, behavior precautions or configuration requirements prior to updating your firmware.
4. Ensure that the device is in a safe state prior to initiating the firmware update. The card may be temporarily inaccessible during the firmware update process.
5. Connect a USB cable between the card and the PC and open the studio or NPU. If the software contains newer firmware, it will automatically prompt you to update the firmware. Proceed with the firmware update.
6. Once the firmware update process has started, do not interrupt the card as this may corrupt the firmware. Do NOT manually power-cycle the inverter or reboot the card. Do NOT disturb the USB connection.

7. After the firmware update has been completed, the card will reset automatically. When the card boots up again, it will be running the new application firmware, which can be confirmed by observing the version displayed in the **Device...Device Info** in the studio or in the **Firmware Version** setting in the NPU.



## PROTOCOL-SPECIFIC INFORMATION

---

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

### 9.1 BACnet MS/TP

---

- The interface card supports the BACnet MS/TP protocol over RS-485 at baud rates of 9600, 19200, 38400, 57600, 76800, and 115200.
- Supports up to 100 simultaneous COV subscriptions.

#### 9.1.1 Protocol Implementation Conformance Statement (PICS)

##### BACnet Protocol

Date:	September 20, 2019
Vendor Name:	ICC, Inc.
Product Name:	Mitsubishi Inverter FR-800
Product Model Number:	A8NXLT
Applications Software Version:	V1.400
Firmware Revision:	V1.400
BACnet Protocol Revision:	12
Product Description:	

The Mitsubishi 800-series inverter family delivers multi-use performance, accuracy and system flexibility in an ultra-reliable hardware package.

##### 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)
- ☒ Data Sharing – WritePropertyMultiple-B (DS-WPM-B)
- ☒ Data Sharing – COV-B (DS-COV-B)
- ☒ Device Management – Dynamic Device Binding-B (DM-DDB-B)
- ☒ Device Management – Dynamic Object Binding-B (DM-DOB-B)
- ☒ Device Management – DeviceCommunicationControl-B (DM-DCC-B)
- ☒ Device Management – ReinitializeDevice-B (DM-RD-B)

#### **Segmentation Capability:**

None

- |                                                             |                   |
|-------------------------------------------------------------|-------------------|
| <input type="checkbox"/> Able to transmit segmented message | Window Size _____ |
| <input type="checkbox"/> Able to receive segmented message  | 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, RS-485 (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, 57600, 76800, 115200
- ☐ 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 devices.) ☐ 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

### Networking Security Options:

- ☒ Non-secure Device – is capable of operating without BACnet Network Security
- ☐ Secure Device – is capable of using BACnet Network Security (NS-SD BIBB)
  - ☐ Multiple Application-Specific Keys:
  - ☐ Supports encryption (NS-ED BIBB)
  - ☐ Key Server (NS-KS BIBB)

### Character Sets Supported:

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

- |                                                       |                                               |                                     |
|-------------------------------------------------------|-----------------------------------------------|-------------------------------------|
| <input checked="" type="checkbox"/> ISO 10646 (UTF-8) | <input type="checkbox"/> IBM™/Microsoft™ DBCS | <input type="checkbox"/> ISO 8859-1 |
| <input type="checkbox"/> ISO 10646 (UCS-2)            | <input type="checkbox"/> ISO 10646 (UCS-4)    | <input type="checkbox"/> JIS X 0208 |

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

**Table 25: BACnet Device Object Types/Properties Supported**

Property	Object Type
	Device
Object Identifier	W
Object Name	W
Object Type	R
System Status	R
Vendor Name	R
Vendor Identifier	R
Model Name	R
Firmware Revision	R
Application Software Version	R
Protocol Version	R
Protocol Revision	R
Protocol Services Supported	R
Protocol Object Types Supported	R
Object List	R
Max APDU Length Accepted	R
Segmentation Supported	R
APDU Timeout	W (10...65535)
Number Of APDU Retries	W (0...10)
Max Master	W (1...127)
Max Info Frames	W (1...100)
Device Address Binding	R
Database Revision	R
Active COV Subscriptions	R

R – readable using BACnet services

W – readable and writable using BACnet services

**Table 26: 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
Reliability	R	R	R
Out Of Service	R	R	R
Priority Array		R	R
Relinquish Default		R	R
Polarity	W	W	
Active Text	R	R	R
Inactive Text	R	R	R

R – readable using BACnet services

W – readable and writable using BACnet services

**Table 27: 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
Reliability	R	R	R
Out Of Service	R	R	R
Units	R	R	R
Priority Array		R	R
Relinquish Default		R	R
COV Increment	W	W	W

R – readable using BACnet services

W – readable and writable using BACnet services

**Table 28: BACnet Multi-state Object Types /Properties Supported**

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

R – readable using BACnet services

W – readable and writable using BACnet services

### 9.1.2 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. Always use the studio to confirm the configuration before commissioning the device.

**Table 29: Analog Input Object Instance Summary**

Instance ID	Object Name	Description	Register	Units
AI1	ACTUAL_SPEED	Actual speed	1231	RPM
AI2	OUTPUT_FREQ	Output frequency in 0.01 Hertz (6000=60.00Hz)	201	Hz
AI3	OUTPUT_CURRENT	Output current in 0.1 or 0.01 Amp (depends on inverter capacity)	202	Amps
AI4	OUTPUT_VOLTAGE	Output voltage in 0.1 Volt (1000=100.0V)	203	Voltage
AI5	INPUT_POWER	Input power in 0.1 or 0.01 kW (depends on inverter capacity)	213	kW
AI6	OUTPUT_POWER	Output power in 0.1 or 0.01 kW (depends on inverter capacity)	214	kW
AI7	CUMULATIVE_POWER	Energy consumption	225	kWh
AI8	PID_SET_POINT	PID set point in 0.1%	252	%
AI9	PID_MEASURED_VALUE	PID measured value in 0.1%	253	%
AI10	PID_DEVIATION_VALUE	PID deviation value in 0.1%	254	%
AI11	FAULT_CODE_1	Fault code 1 (most recent)	284	None
AI12	FAULT_CODE_2	Fault code 2	285	None
AI13	FAULT_CODE_3	Fault code 3	286	None
AI14	FAULT_CODE_4	Fault code 4	287	None
AI15	FAULT_CODE_5	Fault code 5	288	None
AI16	FAULT_CODE_6	Fault code 6	289	None
AI17	FAULT_CODE_7	Fault code 7	290	None
AI18	FAULT_CODE_8	Fault code 8 (least recent)	291	None

**Table 30: Analog Output Object Instance Summary**

Instance ID	Object Name	Description	Register	Units
AO1	SETPOINT_SPEED	Setpoint speed	1229	RPM
AO2	INVERTER_RESET	Fault reset	5	None
AO3	ALARM_HISTORY_CLEAR	Clear alarm (fault) history	6	None
AO4	ALL_PARAM_CLEAR	Clear all parameters	7	None

**Table 31: Binary Input Object Instance Summary**

Instance ID	Object Name	Description	Register	Bit (Mask)	Active Text
					Inactive Text
BI1	OP_ENABLED	Drive function enabled	1227	2 (0x0004)	enabled
					disabled
BI2	FAULTED	Unacknowledged faults present	1227	3 (0x0008)	faulted
					no fault
BI3	QUICK_STOP_DISABLED	Quick stop disabled	1227	5 (0x0020)	stop
					disabled
BI4	REF_FROM_NET	Speed reference from network	1227	8 (0x0100)	on
					off
BI5	CTRL_FROM_NET	Control reference from network	1227	9 (0x0200)	on
					off
BI6	SETPOINT_REACHED	Reached commanded speed	1227	10 (0x0400)	on
					off
BI7	REV_OP_ENABLED	Drive function enabled in reverse mode	1227	14 (0x4000)	enabled
					disabled
BI8	STOPPING	Drive is stopping	1227	15 (0x8000)	stopping
					off



**Table 32: Binary Output Object Instance Summary**

Instance ID	Object Name	Description	Register	Bit (Mask)	Active Text
					Inactive Text
BO1	OP_CMD	Enable operation	1	1 (0x0002)	operate
					off
BO2	REV_OP_CMD	Enable reverse operation	1	2 (0x0004)	reverse
					off
BO3	QUICK_STOP	Quick stop	1	3 (0x0008)	stop
					off
BO4	FAULT_RESET	Reset fault	1	11 (0x0800)	reset
					off
BO5	NET_REF_CMD	Request reference from network	1	14 (0x4000)	on
					off
BO6	NET_CTRL_CMD	Request control from network	1	15 (0x8000)	on
					off

### 9.1.3 BACnet MS/TP Server Settings

In the studio's **Project** panel, navigate to **A8NXLT...RS-485...BACnet MS/TP Server**.

### 9.1.4 BACnet Object Settings

In the studio's **Project** panel, navigate to **A8NXLT...RS-485...BACnet MS/TP Server...Node** and add an object from the **Available Objects** panel.

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 the present value property is writable (for Outputs and Values only).

#### 9.1.4.1 Analog Input Object Settings

##### Object Name

The name of the BACnet object. Enter a string of between 1 and 16 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).

##### Register

The inverter register number (refer to section 5.2) that the BACnet object's present value will access.

##### Data Type

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

##### Multiplier

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

## **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.)

## **Default COV Increment**

Defines the default COV Increment value to use for this object for change of value detection. When COV reporting is enabled for this object, COV notifications will be generated if the present value property changes by the COV increment value, or more, since the last notification. A COV increment value of 0 indicates that any change to the present value property will generate a COV notification.

### **9.1.4.2 Analog Output Object Settings**

## **Object Name**

The name of the BACnet object. Enter a string of between 1 and 16 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).

## **Register**

The inverter register number (refer to section 5.2) that the BACnet object's present value will access.

## **Data Type**

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

### **Multiplier**

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

### **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.)

### **Default COV Increment**

Defines the default COV Increment value to use for this object for change of value detection. When COV reporting is enabled for this object, COV notifications will be generated if the present value property changes by the COV increment value, or more, since the last notification. A COV increment value of 0 indicates that any change to the present value property will generate a COV notification.

### **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 relinquished (written to NULL).

## **9.1.4.3 Analog Value Object Settings**

### **Object Name**

The name of the BACnet object. Enter a string of between 1 and 16 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).

### **Register**

The inverter register number (refer to section 5.2) that the BACnet object's present value will access.

### **Data Type**

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

### **Multiplier**

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

### **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.)

### **Default COV Increment**

Defines the default COV Increment value to use for this object for change of value detection. When COV reporting is enabled for this object, COV notifications will be generated if the present value property changes by the COV increment value, or more, since the last notification. A COV increment value of 0 indicates that any change to the present value property will generate a COV notification.

### **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 relinquished (written to NULL).

#### 9.1.4.4 Binary Input Object Settings

##### **Object Name**

The name of the BACnet object. Enter a string of between 1 and 16 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).

##### **Register**

The inverter register number (refer to section 5.2) that the BACnet object's present value will access.

##### **Bitmask**

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

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 register at the bit location(s) indicated in the bitmask. If all of the bit locations at the designated register 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 8 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 8 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 register) and the logical state represented by the object's present value property. If the physical state is active high, select "Normal" from this dropdown menu. If the physical state is active low, select "Reverse" from this dropdown menu. For further detail, refer to the "Bitmask" behavioral description.

### **9.1.4.5 Binary Output Object Settings**

#### **Object Name**

The name of the BACnet object. Enter a string of between 1 and 16 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).

#### **Register**

The inverter register number (refer to section 5.2) that the BACnet object's present value will access.

#### **Bitmask**

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

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 register 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 register 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 register at the bit location(s) indicated in the bitmask. If all of the bit locations at the designated register 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 8 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 8 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 register) 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 relinquished (written to NULL).

### **9.1.4.6 Binary Value Object Settings**

#### **Object Name**

The name of the BACnet object. Enter a string of between 1 and 16 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).



## **Register**

The inverter register number (refer to section 5.2) that the BACnet object's present value will access.

## **Bitmask**

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

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 register 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 register 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 register at the bit location(s) indicated in the bitmask. If all of the bit locations at the designated register 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 8 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 8 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 relinquished (written to NULL).

#### 9.1.4.7 Multi-state Input Object Settings

##### **Object Name**

The name of the BACnet object. Enter a string of between 1 and 16 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).

##### **Register**

The inverter register number (refer to section 5.2) that the BACnet object's present value will access.

##### **Data Type**

Fixed at 16-Bit Unsigned.

##### **Automatic Number of States**

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

##### **Number of States**

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

##### **Offset by One**

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

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

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

#### **9.1.4.8 Multi-state Output Object Settings**

##### **Object Name**

The name of the BACnet object. Enter a string of between 1 and 16 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).

##### **Register**

The inverter register number (refer to section 5.2) 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 relinquished (written to NULL).

#### **9.1.4.9 Multi-state Value Object Settings**

##### **Object Name**

The name of the BACnet object. Enter a string of between 1 and 16 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).

##### **Register**

The inverter register number (refer to section 5.2) 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 relinquished (written to NULL).

## 9.2 Modbus RTU

### 9.2.1 Overview

The interface card supports the Modbus RTU protocol.

**Other notes of interest include:**

- Supported Modbus RTU functions are indicated in Table 33.
- The register mapping is provided in section 5.2.
- Inverter registers can be addressed as holding registers (4X references) and input registers (3X references).
- Specific bits within inverter registers can be accessed as either coils (0X references) or discrete inputs (1X references).
- Write data checking is not available (refer to section 5.1.) For example, if a write is performed to a register with a data value that is out-of-range of the corresponding parameter object, no Modbus exception will be immediately returned.
- The driver can be configured to detect a timeout (communication loss) and perform a timeout action.

**Table 33: Supported Modbus RTU Functions**

Function Code	Function
1	Read coils
2	Read input status
3	Read multiple registers
4	Read input registers
5	Write coil
6	Write single register
8	Diagnostics (subfunction 0 only)
15	Force multiple coils
16	Write multiple registers

### 9.2.2 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 5.2. 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 40201 is the same as Modbus holding register 201. The same description applies to input registers (3X).

For example, from a Modbus RTU master's point of view, in order to access the output frequency (register 201) as a holding register, the Modbus RTU master must execute the Read Multiple Registers function code and target register 201. This will similarly apply when accessing an inverter register as an Input Register.

### 9.2.3 Coil & Discrete Input Mappings

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

Discretes 1...16 map to register #1, bit0...bit15 (bit0=LSB, bit15=MSB)  
Discretes 17...32 map to register #2, bit0...bit15, and so on.

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

$$\text{register} = \left\lfloor \frac{\text{discrete} + 15}{16} \right\rfloor \quad \text{Equation 1}$$

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:

$$\text{bit} = (\text{discrete} - 1) \% 16 \quad \text{Equation 2}$$

Where “discrete”  $\in [1 \dots 65535]$ , “bit”  $\in [0 \dots 15]$ , 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  $\lfloor 3.0625 \rfloor = \lfloor 3 \text{ r}1 \rfloor = 3$ . Then, using Equation

2, we can determine that the bit within register #3 that coil #34 targets is  $(34-1)\%16 = 1$ , as  $33\%16 = \text{mod}(2 \text{ r1}) = 1$ . Therefore, reading coil #34 will return the value of register #3, bit #1.

## 9.2.4 Holding/Input Register Remap Settings

In the studio's **Project** panel, add **A8NXLT...RS-485...Modbus RTU Slave...Holding/Input Register Remap**.

Holding/input register remap objects are **OPTIONAL**. By default, all inverter registers are already mapping as both holding (4X) and input (3X) registers (refer to section 9.2.2). For user convenience, register remap objects can be created to map any inverter register to holding/input register 5001 to 5050.

At times, it may be convenient to access inverter registers in bulk Modbus transactions. This may be especially true in situations where it is desired to access certain registers that are non-contiguous. For example, if it were desired to read the inverter's output frequency (register 201), converter output voltage (register 208), and PID deviation value (register 254), 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 201 for a quantity of 54 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 registers can be grouped together in any order and accessed efficiently via the Modbus RTU "read multiple registers" and "write multiple registers" function codes. The net effect is one of being able to transfer larger blocks of registers using fewer Modbus transactions, which results in improved network utilization and simpler data manipulation code on the Modbus master device.

### Description

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

### Remap Register

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



## **Register**

Inverter register (refer to section 5.2) that is accessed by the **Remap Register**.

## 9.3 Metasys N2

The interface card supports the Johnson Controls Metasys® N2 Open protocol. Some notes of interest are:

- Supports fully-configurable analog input, analog output, binary input, binary output, internal float, internal integer, and internal byte object types.
- The Metasys device type for the driver is VND.
- The driver can be configured to detect a timeout (communication loss) and perform a timeout action.

### 9.3.1 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. Always use the studio to confirm the configuration before commissioning the device.

Table 34: Analog Input Object Instance Summary

Instance	Object Name	Description	Register
1	ACTUAL_SPEED	Actual speed	1231
2	OUTPUT_FREQ	Output frequency in 0.01 Hertz (6000=60.00Hz)	201
3	OUTPUT_CURRENT	Output current in 0.1 or 0.01 Amp (depends on inverter capacity)	202
4	OUTPUT_VOLTAGE	Output voltage in 0.1 Volt (1000=100.0V)	203
5	INPUT_POWER	Input power in 0.1 or 0.01 kW (depends on inverter capacity)	213
6	OUTPUT_POWER	Output power in 0.1 or 0.01 kW (depends on inverter capacity)	214
7	CUMULATIVE_POWER	Energy consumption	225
8	PID_SET_POINT	PID set point in 0.1%	252
9	PID_MEASURED_VALUE	PID measured value in 0.1%	253
10	PID_DEVIATION_VALUE	PID deviation value in 0.1%	254
11	FAULT_CODE_1	Fault code 1 (most recent)	284

Instance	Object Name	Description	Register
12	FAULT_CODE_2	Fault code 2	285
13	FAULT_CODE_3	Fault code 3	286
14	FAULT_CODE_4	Fault code 4	287
15	FAULT_CODE_5	Fault code 5	288
16	FAULT_CODE_6	Fault code 6	289
17	FAULT_CODE_7	Fault code 7	290
18	FAULT_CODE_8	Fault code 8 (least recent)	291

**Table 35: Analog Output Object Instance Summary**

Instance	Object Name	Description	Register
1	SETPOINT_SPEED	Setpoint speed	1229
2	INVERTER_RESET	Fault reset	5
3	ALARM_HISTORY_CLEAR	Clear alarm (fault) history	6
4	ALL_PARAM_CLEAR	Clear all parameters	7

**Table 36: Binary Input Object Instance Summary**

Instance	Object Name	Description	Register	Bit (Mask)
1	OP_ENABLED	Drive function enabled	1227	2 (0x0004)
2	FAULTED	Unacknowledged faults present	1227	3 (0x0008)
3	QUICK_STOP_DISABLED	Quick stop disabled	1227	5 (0x0020)

Instance	Object Name	Description	Register	Bit (Mask)
4	REF_FROM_NET	Speed reference from network	1227	8 (0x0100)
5	CTRL_FROM_NET	Control reference from network	1227	9 (0x0200)
6	SETPOINT_REACHED	Reached commanded speed	1227	10 (0x0400)
7	REV_OP_ENABLED	Drive function enabled in reverse mode	1227	14 (0x4000)
8	STOPPING	Drive is stopping	1227	15 (0x8000)

**Table 37: Binary Output Object Instance Summary**

Instance	Object Name	Description	Register	Bit (Mask)
1	OP_CMD	Enable operation	1	1 (0x0002)
2	REV_OP_CMD	Enable reverse operation	1	2 (0x0004)
3	QUICK_STOP	Quick stop	1	3 (0x0008)
4	FAULT_RESET	Reset fault	1	11 (0x0800)
5	NET_REF_CMD	Request reference from network	1	14 (0x4000)
6	NET_CTRL_CMD	Request control from network	1	15 (0x8000)

Instance	Object Name	Description	Register	Bit (Mask)

### 9.3.2 Metasys N2 Slave Settings

In the studio's **Project** panel, navigate to **A8NXL...RS-485...Metasys N2 Slave**.

### 9.3.3 Metasys Object Settings

In the studio's **Project** panel, navigate to **A8NXL...RS-485...Metasys N2 Slave...Node** and add an object from the **Available Objects** panel.

The N2 slave driver hosts Metasys objects which contain a variety of different attributes for an N2 master to access. The Metasys specification allows a maximum of 256 instances of each object type.

- **Analog input (AI)** objects are used for monitoring analog status items. AI objects support low alarm limits, low warning limits, high warning limits, high alarm limits and differential values. Change of state (COS), alarm and warning functions can also be enabled. An AI object will accept an override command, but will not change its actual value or indicate override active. A “multiplier” is associated with the object, which can provide scaling between the network value and raw (database) value.
- **Analog output (AO)** objects are used for setting and monitoring analog control and configuration items. An AO value can be modified by issuing an override command. Issuing a release command will not cause the AO to automatically return to its pre-override value, nor will the AO automatically return to its pre-override value after a certain time period of no communication. A “multiplier” is associated with the object, which can provide scaling between the network value and raw (database) value.
- **Binary input (BI)** objects are used for monitoring discrete (digital) status items. BI objects support COS, alarm enabling and normal/alarm status indications. A BI object will accept an override command, but will not change its actual value or indicate override active. A “bitmask” is associated with the object, which provides mapping information between the network value and specific bits in the database.

- **Binary output (BO)** points are used for setting and monitoring discrete control and configuration items. A BO value can be modified by issuing an override command. Issuing a release command will not cause the BO to automatically return to its pre-override value, nor will the BO return to its pre-override value after a certain time period of no communication. A “bitmask” is associated with the object, which provides mapping information between the network value and specific bits in the database.
- **Internal float (ADF)** objects are used for setting and monitoring internal floating point values. An internal float value can be modified by issuing an override command or a write attribute command. Issuing a release command will not cause the internal float to automatically return to its pre-override value, nor will the internal float automatically return to its pre-override value after a certain time period of no communication. A “multiplier” is associated with the object, which can provide scaling between the network value and raw (database) value.
- **Internal integer (ADI)** objects are used for setting and monitoring internal integer values. An internal integer value can be modified by issuing an override command or a write attribute command. Issuing a release command will not cause the internal integer to automatically return to its pre-override value, nor will the internal integer automatically return to its pre-override value after a certain time period of no communication. A “multiplier” is associated with the object, which can provide scaling between the network value and raw (database) value.
- **Internal byte (BD)** objects are used for setting and monitoring internal byte values. An internal byte value can be modified by issuing an override command or a write attribute command. Issuing a release command will not cause the internal byte to automatically return to its pre-override value, nor will the internal byte automatically return to its pre-override value after a certain time period of no communication. A “multiplier” is associated with the object, which can provide scaling between the network value and raw (database) value.

### 9.3.3.1 Analog Input Object Settings

#### Object Name

The name of the N2 object. Enter a string of between 1 and 32 characters in length.

#### Instance

The N2 object's instance number. Enter a value between 1...256.

### **Register**

The inverter register number (refer to section 5.2) that the N2 object's value will access.

### **Data Type**

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

### **Multiplier**

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

## **9.3.3.2 Analog Output Object Settings**

### **Object Name**

The name of the N2 object. Enter a string of between 1 and 32 characters in length.

### **Instance**

The N2 object's instance number. Enter a value between 1...256.

### **Register**

The inverter register number (refer to section 5.2) that the N2 object's value will access.

### **Data Type**

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

### **Multiplier**

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

### 9.3.3.3 Binary Input Object Settings

#### **Object Name**

The name of the N2 object. Enter a string of between 1 and 32 characters in length.

#### **Instance**

The N2 object's instance number. Enter a value between 1...256.

#### **Register**

The inverter register number (refer to section 5.2) that the N2 object's value will access.

#### **Bitmask**

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

The effect of the "Bitmask" field when reading: When the current state of a binary object is read by a Metasys master, the bitmask is used to determine the state of the object by inspecting the value in the designated register at the bit location(s) indicated in the bitmask. If all of the bit locations at the designated register indicated by a checkmark in the bitmask are set, then the object's state will be returned as "1". Else, the object's state will be returned as "0".

### 9.3.3.4 Binary Output Object Settings

#### **Object Name**

The name of the N2 object. Enter a string of between 1 and 32 characters in length.

#### **Instance**

The N2 object's instance number. Enter a value between 1...256.

#### **Register**

The inverter register number (refer to section 5.2) that the N2 object's value will access.



## **Bitmask**

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

The effect of the “Bitmask” field when reading: When the current state of a binary object is read by a Metasys master, the bitmask is used to determine the state of the object by inspecting the value in the designated register at the bit location(s) indicated in the bitmask. If all of the bit locations at the designated register indicated by a checkmark in the bitmask are set, then the object’s state will be returned as “1”. Else, the object’s state will be returned as “0”.

The effect of the “Bitmask” field when writing: When the current state of a binary object is overridden to “1” by a Metasys master, then the bit(s) in the designated register indicated by a checkmark in the bitmask are set. Similarly, when the current state of the object is overridden to “0”, then the bit(s) in the designated register indicated by a checkmark in the bitmask are cleared.

### **9.3.3.5 Internal Float Object Settings**

#### **Object Name**

The name of the N2 object. Enter a string of between 1 and 32 characters in length.

#### **Instance**

The N2 object’s instance number. Enter a value between 1...256.

#### **Register**

The inverter register number (refer to section 5.2) that the N2 object’s value will access.

#### **Data Type**

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

### **Multiplier**

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

#### **9.3.3.6 Internal Integer Object Settings**

### **Object Name**

The name of the N2 object. Enter a string of between 1 and 32 characters in length.

### **Instance**

The N2 object's instance number. Enter a value between 1...256.

### **Register**

The inverter register number (refer to section 5.2) that the N2 object's value will access.

### **Data Type**

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

### **Multiplier**

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

#### **9.3.3.7 Internal Byte Object Settings**

### **Object Name**

The name of the N2 object. Enter a string of between 1 and 32 characters in length.

### **Instance**

The N2 object's instance number. Enter a value between 1...256.

### **Register**

The inverter register number (refer to section 5.2) that the N2 object's value will access.

### **Data Type**

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

### **Multiplier**

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

## 9.4 Siemens FLN (P1)

The interface card supports the Siemens FLN (P1) protocol. Some notes of interest are:

- Supports fully-configurable logical analog input (LAI), logical analog output (LAO), logical digital input (LDI), logical digital output (LDO) point types.
- Supports custom application number, application descriptor, revision string, and revision number.
- The driver can be configured to detect a timeout (communication loss) and perform a timeout action.

### 9.4.1 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. Always use the studio to confirm the configuration before commissioning the device.

Table 38: LAI Object Summary

Point Number	Object Name	Description	Register	Slope	Intercept	Units
3	ACTUAL_SPEED	Actual speed	1231	0.1	0	RPM
4	OUTPUT_FREQ	Output frequency in 0.01 Hertz (6000=60.00Hz)	201	0.01	0	HERTZ
5	OUTPUT_CURRENT	Output current in 0.1 or 0.01 Amp (depends on inverter capacity)	202	0.01	0	AMP
6	OUTPUT_VOLTAGE	Output voltage in 0.1 Volt (1000=100.0V)	203	0.1	0	VOLT
7	INPUT_POWER	Input power in 0.1 or 0.01 kW (depends on inverter capacity)	213	0.01	0	KW

Point Number	Object Name	Description	Register	Slope	Intercept	Units
8	OUTPUT_POWER	Output power in 0.1 or 0.01 kW (depends on inverter capacity)	214	0.01	0	KW
9	CUMULATIVE_POWER	Energy consumption	225	1	0	KWH
10	PID_SET_POINT	PID set point in 0.1%	252	0.1	0	PCT
11	PID_MEASURED_VALUE	PID measured value in 0.1%	253	0.1	0	PCT
12	PID_DEVIATION_VALUE	PID deviation value in 0.1%	254	0.1	0	PCT
13	FAULT_CODE_1	Fault code 1 (most recent)	284	1	0	
14	FAULT_CODE_2	Fault code 2	285	1	0	
15	FAULT_CODE_3	Fault code 3	286	1	0	
16	FAULT_CODE_4	Fault code 4	287	1	0	
17	FAULT_CODE_5	Fault code 5	288	1	0	
18	FAULT_CODE_6	Fault code 6	289	1	0	
19	FAULT_CODE_7	Fault code 7	290	1	0	
21	FAULT_CODE_8	Fault code 8 (least recent)	291	1	0	

**Table 39: LAO Object Summary**

Point Number	Object Name	Description	Register	Slope	Intercept	Units
22	SETPOINT_SPEED	Setpoint speed	1229	0.1	0	RPM
23	INVERTER_RESET	Fault reset	5	1	0	
24	ALARM_HISTORY_CLEAR	Clear alarm (fault) history	6	1	0	
25	ALL_PARAM_CLEAR	Clear all parameters	7	1	0	

**Table 40: LDI Object Summary**

Point Number	Object Name	Description	Register	Bit (Mask)	On Text
					Off Text
26	OP_ENABLED	Drive function enabled	1227	2 (0x0004)	EN
					DIS
27	FAULTED	Unacknowledged faults present	1227	3 (0x0008)	FLT
					NO FLT
28	QUICK_STOP_DISABLED	Quick stop disabled	1227	5 (0x0020)	STOP
					DIS
30	REF_FROM_NET	Speed reference from network	1227	8 (0x0100)	ON
					OFF
31	CTRL_FROM_NET	Control reference from network	1227	9 (0x0200)	ON
					OFF
32	SETPOINT_REACHED	Reached commanded speed	1227	10 (0x0400)	ON
					OFF
33	REV_OP_ENABLED	Drive function enabled in reverse mode	1227	14 (0x4000)	EN
					DIS
34	STOPPING	Drive is stopping	1227	15 (0x8000)	STOP
					OFF

**Table 41: LDO Object Summary**

Point Number	Object Name	Description	Register	Bit (Mask)	On Text
					Off Text
35	OP_CMD	Enable operation	1	1 (0x0002)	OP
					OFF

Point Number	Object Name	Description	Register	Bit (Mask)	On Text
					Off Text
36	REV_OP_CMD	Enable reverse operation	1	2 (0x0004)	REV OFF
37	QUICK_STOP	Quick stop	1	3 (0x0008)	STOP OFF
38	FAULT_RESET	Reset fault	1	11 (0x0800)	RESET OFF
39	NET_REF_CMD	Request reference from network	1	14 (0x4000)	ON OFF
40	NET_CTRL_CMD	Request control from network	1	15 (0x8000)	ON OFF

### 9.4.2 Siemens FLN Slave Settings

In the studio's **Project** panel, navigate to **A8NXLT...RS-485...Siemens FLN Slave**.

### 9.4.3 Node Settings

In the studio's **Project** panel, navigate to **A8NXLT...RS-485...Siemens FLN...Node**.

#### Application Number

Defines the unique application number for the application. The default is 7360.

#### Application Descriptor

Enter the application descriptor string (up to 12 ASCII characters). The default is "MITS A800".

#### Revision String

Defines the unique 4-character revision string. The default is ICC0.

### **Revision Number**

Defines the revision number (0...255) for this application. The default is 0.

## **9.4.4 FLN Object Settings**

In the studio's **Project** panel, navigate to **A8NXLT...RS-485...Siemens FLN...Node** and add an object from the **Available Objects** panel.

### **9.4.4.1 Logical Analog Input (LAI) Object Settings**

#### **Point Descriptor**

The name of the point. Enter a string of between 1 and 12 characters in length. All point names must be unique within a node, and characters must be valid for encoding in RAD50 format ("A"... "Z", "0"... "9", blank, "\$", ".", and "?").

#### **Point Number**

The point number (1...99). Note that point numbers that are reserved by the FLN specification cannot be used (1, 2, 20, 29 and 99).

#### **Register**

The inverter register number (refer to section 5.2) that the point's value will access.

#### **Data Type**

Fixed at 16-Bit Unsigned.

#### **Default Value**

Defines the factory default physical value of the point (0...32767).

#### **Max Value**

Defines the maximum physical value that this point can attain (0...32767).



### **Slope**

Defines the floating-point slope value of the point. The master uses this value to calculate a floating-point analog value using the equation  $y=mx+b$ , where  $y$  is the analog value,  $x$  is the physical value stored in the database,  $m$  is the slope, and  $b$  is the intercept.

### **Intercept**

Defines the floating-point intercept value of the point. The master uses this value to calculate a floating-point analog value using the equation  $y=mx+b$ , where  $y$  is the analog value,  $x$  is the physical value stored in the database,  $m$  is the slope, and  $b$  is the intercept.

### **Units**

Enter an ASCII string of up to 6 characters in length which represent the engineering units of the point. Characters must be valid for encoding in RAD50 format ("A"... "Z", "0"... "9", blank, "\$", ".", and "?").

## **9.4.4.2 Logical Analog Output (LAO) Object Settings**

### **Point Descriptor**

The name of the point. Enter a string of between 1 and 12 characters in length. All point names must be unique within a node, and characters must be valid for encoding in RAD50 format ("A"... "Z", "0"... "9", blank, "\$", ".", and "?").

### **Point Number**

The point number (1...99). Note that point numbers that are reserved by the FLN specification cannot be used (1, 2, 20, 29 and 99).

### **Register**

The inverter register number (refer to section 5.2) that the point's value will access.

### **Data Type**

Fixed at 16-Bit Unsigned.

### **Default Value**

Defines the factory default physical value of the point (0...32767).

### **Max Value**

Defines the maximum physical value that this point can attain (0...32767).

### **Slope**

Defines the floating-point slope value of the point. The master uses this value to calculate an analog value using the equation  $y = mx + b$ , where  $y$  is the analog value,  $x$  is the physical value,  $m$  is the slope, and  $b$  is the intercept.

### **Intercept**

Defines the floating-point intercept value of the point. The master uses this value to calculate an analog value using the equation  $y = mx + b$ , where  $y$  is the analog value,  $x$  is the physical value,  $m$  is the slope, and  $b$  is the intercept.

### **Units**

Enter an ASCII string of up to 6 characters in length which represent the engineering units of the point. Characters must be valid for encoding in RAD50 format ("A"... "Z", "0"... "9", blank, "\$", "." and "?").

## **9.4.4.3 Logical Digital Input (LDI) Object Settings**

### **Point Descriptor**

The name of the point. Enter a string of between 1 and 12 characters in length. All point names must be unique within a node, and characters must be valid for encoding in RAD50 format ("A"... "Z", "0"... "9", blank, "\$", "." and "?").

### **Point Number**

The point number (1...99). Note that point numbers that are reserved by the FLN specification cannot be used (1, 2, 20, 29 and 99).

### **Register**

The inverter register number (refer to section 5.2) that the point's value will access.

### **Bitmask**

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

The effect of the “Bitmask” field when reading: When the value of an LDI object is read by an FLN controller, the bitmask is used to determine the value of the object by inspecting the value in the designated register at the bit location(s) indicated in the bitmask. If all of the bit locations at the designated register indicated by a checkmark in the bitmask are set, then the object’s value will be returned as “on”. Else, the object’s value will be returned as “off”.

### **On Text**

Enter an ASCII string of up to 6 characters in length which represent the On Text of the point. Characters must be valid for encoding in RAD50 format (“A”...“Z”, “0”...“9”, blank, “\$”, “.” and “?”).

### **Off Text**

Enter an ASCII string of up to 6 characters in length which represent the Off Text of the point. Characters must be valid for encoding in RAD50 format (“A”...“Z”, “0”...“9”, blank, “\$”, “.” and “?”).

## **9.4.4.4 Logical Digital Output (LDO) Object Settings**

### **Point Descriptor**

The name of the point. Enter a string of between 1 and 12 characters in length. All point names must be unique within a node, and characters must be valid for encoding in RAD50 format (“A”...“Z”, “0”...“9”, blank, “\$”, “.” and “?”).

### **Point Number**

The point number (1...99). Note that point numbers that are reserved by the FLN specification cannot be used (1, 2, 20, 29 and 99).

### **Register**

The inverter register number (refer to section 5.2) that the point’s value will access.

### **Bitmask**

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

The effect of the “Bitmask” field when reading: When the value of an LDI object is read by an FLN controller, the bitmask is used to determine the value of the object by inspecting the value in the designated register at the bit location(s) indicated in the bitmask. If all of the bit locations at the designated register indicated by a checkmark in the bitmask are set, then the object’s value will be returned as “on”. Else, the object’s value will be returned as “off”.

The effect of the “Bitmask” field when writing: When the value of an LDO object is commanded to “on” by an FLN controller, then the bit(s) in the designated register indicated by a checkmark in the bitmask are set. Similarly, when the value of an LDO object is commanded to “off”, then the bit(s) in the designated register indicated by a checkmark in the bitmask are cleared.

### **On Text**

Enter an ASCII string of up to 6 characters in length which represent the On Text of the point. Characters must be valid for encoding in RAD50 format (“A”...“Z”, “0”...“9”, blank, “\$”, “.” and “?”).

### **Off Text**

Enter an ASCII string of up to 6 characters in length which represent the Off Text of the point. Characters must be valid for encoding in RAD50 format (“A”...“Z”, “0”...“9”, blank, “\$”, “.” and “?”).

## 9.5 DMX-512

This interface card supports the DMX-512 protocol, which allows any connected non-DMX equipment to receive data from a universal DMX controller device. Some notes of interest are:

- Fully configurable to occupy any number of sequential DMX channels.
- Capable of using all 512 DMX channels.
- Each inverter parameter uses 2 DMX channels with configurable byte order.
- The driver can be configured to detect a timeout (communication loss) and perform a timeout action.

### 9.5.1 Connections

This section describes the typical connections used to connect a DMX controller device to the interface card.

While there are a variety of different DMX-512 connector types in existence, most standard DMX-512 connectors use either XLR 5-pin or 3-pin connectors (refer to Figure 8 and Figure 9). A female connector is fitted to a transmitter device (e.g. a console,) while a male connector is fitted to a receiver device (e.g. a dimmer or servo).



**Figure 8: 5-Pin XLR Connector**



**Figure 9: 3-Pin XLR Connectors**

An appropriate wiring harness must be used when connecting the DMX-512 network to the interface card's RS-485 port. This can be accomplished by using off-the-shelf DMX-512 cabling with bare-wire terminations on one end, or by simply

cutting a standard DMX-512 cable in half and stripping back the wires. Refer to Table 42 for an overview of DMX-512 pin assignments and connections.

**Table 42: DMX-512 Pin Assignments**

Pin	Usage	Card Connection
1	Network GND reference	COM
2	Primary data-	D-
3	Primary data+	D+
4	Optional secondary data- (not available on 3-pin connectors)	N/A
5	Optional secondary data+ (not available on 3-pin connectors)	N/A

## 9.5.2 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. Always use the studio to confirm the configuration before commissioning the device.

**Table 43: DMX-512 Channel Summary**

Channel	Name	Description	Register (Bit)	Resolution	Off Value	On Value
1	Net Ctrl Cmd	Request control from network	1 (Bit 15)	8-Bit	0...127	128...255
2	Net Ref Cmd	Request reference from network	1 (Bit 14)	8-Bit	0...127	128...255
3	Op Cmd	Enable operation	1 (Bit 1)	8-Bit	0...127	128...255
4	Rev Op Cmd	Enable reverse operation	1 (Bit 2)	8-Bit	0...127	128...255
5	SETPOINT_SPEED	Setpoint speed (Coarse)	1229	16-Bit	N/A	N/A
6		Setpoint speed (Fine)				

### 9.5.3 *DMX-512 Slave Settings*

In the studio's **Project** panel, navigate to **A8NXLT...RS-485...DMX-512 Slave**.

### 9.5.4 *DMX Parameter Settings*

In the studio's **Project** panel, navigate to **A8NXLT...RS-485...DMX-512 Slave...Node** and add an object from the **Available Objects** panel.

#### Description

The description of the parameter. Enter a string of between 1 and 32 characters in length.

#### Register

The inverter register number (refer to section 5.2) that the parameter's value will access.

#### Resolution

Fixed at 16-Bit.

#### Channel Order

Selects the order of the channels that comprise the value of this parameter. High byte first means the first DMX channel is the most significant byte, or coarse value. Low byte first means the first DMX channel is the least significant byte, or fine value.

## 9.6 Generic Serial

The Generic Serial driver can be used to communicate as a slave with any serial device using configurable ASCII and/or binary data packets. This includes devices such as ASCII serial devices and devices using custom or proprietary serial protocols. Some notes of interest are:

- Supports communication with almost any serial device.
- Supports both master-slave transactions and producer-consumer transactions.
- Transactions are defined at a packet level by adding configurable packet data objects.
- Versatile packet matching options allow handling a variety of different packets with a single transaction definition.
- Requests are matched to transactions based on the order that the transactions are defined. This allows defining a “default” or “exception” transaction last to be initiated for undefined requests.
- Transactions support using database logic to manipulate received data before placing the result into the response.
- Supports variable-sized data fields and packets containing a variable number of data elements.
- Supports binary, ASCII hexadecimal, ASCII decimal, and ASCII text data encodings.
- Supports unsigned integer, signed integer, and IEEE-754 floating point number formats.
- Full support for 8-bit, 16-bit, and 32-bit checksum and CRC fields.
- The driver can be configured to detect a timeout (communication loss) and perform a timeout action.

### 9.6.1 Generic Serial Slave Settings

In the studio's **Project** panel, navigate to **A8NXLT...RS-485...Generic Serial Slave**.

#### Data Bits

Select between 7 or 8 data bits per character.

#### Packet Gap Interval

Defines the number of character times of silence on the network that indicates the end of a packet.



## 9.6.2 Transactions

Communications is established by defining configurable transactions which will be matched to a received packet based on the order they are defined. Each transaction consists of a packet expected to be received from a device on the network and, optionally, a response packet to transmit back to that device.

### 9.6.2.1 Slave Transaction

In the studio's **Project** panel, navigate to **A8NXLT...RS-485...Generic Serial Slave...Slave Transaction** and add an object from the **Available Objects** panel to the **Expected Request** or **Response** packet definition.

A slave transaction is a transaction in which a master device sends a request to the driver and expects a response. When a slave transaction is added to the configuration, an expected request and response is automatically added below it. The expected request defines a packet that is expected to be received from the master. The response defines the packet to send to the master after receiving the request.

### 9.6.2.2 Consumer Transaction

In the studio's **Project** panel, navigate to **A8NXLT...RS-485...Generic Serial Slave...Consumer Transaction** and add an object from the **Available Objects** panel to the **Expected Consumed Data** packet definition.

A consumer transaction is a transaction in which a producer device sends a data packet to the driver and does not expect a response. When a consumer transaction is added to the configuration, a consumed data packet is automatically added below it. The consumed data packet defines a packet that is expected to be received from producers on the network.

### 9.6.2.3 Transaction Settings

#### Description

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

#### Uses Database Logic

Check this option if the transaction requires database logic to run between receiving the expected request and building the response.

### 9.6.3 Packet Data Objects

Packet data objects are used to build a packet. Every character or field that is present in a packet is defined by one or more packet data objects. The order of the fields in the packet dictates the order that the corresponding packet data objects must be added. There are various types of packet data objects available in order to facilitate the definition of a packet.

#### 9.6.3.1 Constant Data

Adds constant data characters to the packet.

##### Name

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

##### Constant Data Characters

Defines the constant data that is encoded in the packet. Enter up to 16 hexadecimal bytes or ASCII characters.

#### 9.6.3.2 Drive Register Data

Adds data that is mapped to the inverter's registers. For transmitted packets, the data is read from the inverter and put into the packet. For received packets, the data from the packet is written to the inverter.

##### Name

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

##### Element Encoding

Selects the encoding used for a data element in the packet.

##### Number Format

Selects the format for interpreting the value of a number.

##### Fixed Element Size

Check this option if the size of a data element is fixed. If this option is unchecked, the driver will automatically determine the size of a variable-sized data element.

### **Element Size**

Enter the number of characters reserved for each data element in the packet.

### **Byte Order**

Selects the byte ordering used for multi-byte data elements in the packet.

### **Number of Elements**

Defines the number of data elements in the packet to map to the device's database.

### **Starting Register**

Defines the starting inverter register number where the packet's data elements are mapped.

### **Internal Data Type**

Specifies how each data element in the packet will be stored internally in the option card. This defines the resolution of the value and whether the value should be treated as signed or unsigned.

### **Multiplier**

The amount that data values are scaled by prior to being stored into the database or after being retrieved from the database. Prior to storage into the database, data values are divided by the multiplier to produce database values. Upon retrieval from the database, database values are multiplied by the multiplier to produce data values.

#### **9.6.3.3 Variable Drive Register Data**

Adds data that is mapped to the device's database which consists of a variable number of elements. The data from a received packet is written to the database. If the actual number of elements in the packet is less than the defined maximum, the remaining database values will be set to 0.

### **Name**

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

### **Element Encoding**

Selects the encoding used for a data element in the packet.

### **Number Format**

Selects the format for interpreting the value of a number.

### **Fixed Element Size**

Check this option if the size of a data element is fixed. If this option is unchecked, the driver will automatically determine the size of a variable-sized data element.

### **Element Size**

Enter the number of characters reserved for each data element in the packet.

### **Byte Order**

Selects the byte ordering used for multi-byte data elements in the packet.

### **Max Number of Elements**

Defines the maximum number of data elements in the packet to map to the device's database. If the actual number of elements in the packet is less than this, the remaining database values will be 0.

### **Starting Register**

Defines the starting inverter register number where the packet's data elements are mapped.

### **Internal Data Type**

Specifies how each data element in the packet will be stored internally in the option card. This defines the resolution of the value and whether the value should be treated as signed or unsigned.

### **Multiplier**

The amount that data values are scaled by prior to being stored into the database or after being retrieved from the database. Prior to storage into the database, data values are divided by the multiplier to produce database values. Upon retrieval from the database, database values are multiplied by the multiplier to produce data values.

#### **9.6.3.4 Ignored Characters**

Adds "Don't Care" characters whose values are ignored.

### **Name**

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

### **Number of Characters**

Defines the number of "Don't Care" characters in the packet that will be ignored.

#### **9.6.3.5 Variable Ignored Characters**

Adds a variable number of "Don't Care" characters whose values are ignored.

### **Name**

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

### **Max Number of Characters**

Defines the maximum number of "Don't Care" characters in the packet that will be ignored.

#### **9.6.3.6 Ranged Byte**

Adds a single byte that must be within a defined range.

### **Name**

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

### **Min Value**

Defines the minimum value the byte can have to be considered a match.

### **Max Value**

Defines the maximum value the byte can have to be considered a match.

#### **9.6.3.7 Bitmasked Byte**

Adds a single byte that must match a defined value after a bitmask is applied.

### **Name**

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

### **Bitmask**

Specifies the bit(s) in the byte that are relevant for detecting a match.

### **Value**

Defines the value the byte must have to be considered a match after applying the bitmask.

#### **9.6.3.8 Register Matched Byte**

Adds a single byte that must match the current value stored at a location in one of the drive's registers.

### **Name**

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

### **Register**

Defines the inverter register number containing the value the byte must have to be considered a match.

#### **9.6.3.9 List Matched Character**

Adds a single character that must match a value from a list of values.

### **Name**

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

### **Value List**

Defines a list of possible values the character can have to be considered a match. Enter up to 15 hexadecimal bytes or ASCII characters.

#### **9.6.3.10 Checksum**

Adds a checksum field that is calculated from a defined start offset in the packet up to, but not including, the location of the checksum field itself.

### **Name**

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

### **Checksum Data Type**

Selects the data type and defines the width of the checksum.

### **Encoding**

Selects the encoding used for the checksum in the packet.

### **Checksum Size**

The number of characters reserved for the checksum in the packet.

### **Byte Order**

Selects the byte ordering for how the checksum is encoded in the packet.

#### ***Checksum Calculation Parameters***

### **Start Offset**

Defines the starting offset in the packet where the checksum calculation will begin.

### **Use 2 Chars/Byte**

Check this option to first convert 2 ASCII characters from the packet into a byte value which will be used in the checksum calculation. If this option is unchecked, the raw bytes from the packet will be used directly in the checksum calculation.

### **Final Operation**

Selects an optional final operation to apply to the checksum value at the end of the calculation.

#### **9.6.3.11 CRC**

Adds a cyclic redundancy check field that is calculated from a defined start offset in the packet up to, but not including, the location of the CRC field itself.

### **Name**

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

### **CRC Data Type**

Selects the data type and defines the width of the CRC.

### **Encoding**

Selects the encoding used for the CRC in the packet.

### **CRC Size**

The number of characters reserved for the CRC in the packet.

### **Byte Order**

Selects the byte ordering for how the CRC is encoded in the packet.

### ***CRC Calculation Parameters***

### **Start Offset**

Defines the starting offset in the packet where the CRC calculation will begin.



### **Use 2 Chars/Byte**

Check this option to first convert 2 ASCII characters from the packet into a byte value which will be used in the CRC calculation. If this option is unchecked, the raw bytes from the packet will be used directly in the CRC calculation.

### **Bit Order (Shift Direction)**

Selects which bit in each byte is considered first (and by correlation, the direction bits are shifted) when calculating the CRC.

### **Polynomial**

Defines the generator polynomial used in the CRC calculation. Enter the hexadecimal representation of the polynomial's bit sequence, most-significant bit first, omitting the highest-order bit.

### **Initial Value**

Enter the value that the CRC will be initialized to at the beginning of the CRC calculation.

### **Final XOR Value**

Enter the value that the CRC will be XORed with at the end of the CRC calculation.

# 10 TROUBLESHOOTING

Although by no means exhaustive, the following table provides possible causes behind some of the most common errors experienced when using this option card.

Problem	Symptom	Solution
Inverter displays E.1, E.2, or E.3 alarm, or the card is unresponsive	Inverter cannot recognize the option	<ul style="list-style-type: none"><li>• Confirm that the interface card connector is properly seated. Refer to section 2.</li><li>• Downloading a configuration or updating firmware via the studio disrupts the communication with the inverter. Reset the fault.</li></ul>
No communications between the network and the card	Communications cannot be established, the RS-485 "TX" LED is off, or the RS-485 "RX" LED flashes only infrequently, not at all, or is solid on	<ul style="list-style-type: none"><li>• Confirm that the card is running normally (Status LED is solid green, not blinking red or rapidly blinking green). Refer to section 1.4.2.</li><li>• Confirm that the RS-485 wiring is correct as described in section 3.</li><li>• Ensure that the card is programmed with compatible network settings. Refer to section 4.1.</li><li>• If the "RX" LED is solid on, this typically indicates reversed wiring polarity. Swap the RS-485 D+ and D- wires.</li><li>• If the "RX" LED is off, this indicates the card is not receiving any data. Confirm that the master device is sending requests on the network.</li></ul>

Problem	Symptom	Solution
Unable to control the inverter via network communications	Cannot write to command parameters via network communications, or writing to these parameters has no apparent effect	<ul style="list-style-type: none"> <li>Set the inverter to NET mode. The inverter will reject all command and parameter write requests from the network if it is not in NET mode. Refer to section 4.2.</li> <li>If using the inverter's terminal contacts, refer to the inverter's instruction manual to determine the appropriate behavior and priority</li> <li>Clear all parameter settings to default and reconfigure the parameters.</li> </ul>
NPU cannot discover the card	The NPU displays "No Device Connected" and the status bar states, "Retrieving Network Settings..."	<ul style="list-style-type: none"> <li>Confirm that the card is running normally and connected via USB.</li> <li>Confirm that the status LED blinks the green/red startup sequence when power is first applied.</li> <li>Confirm that the status LED blinks green while a USB cable is connected.</li> <li>Ensure that the option card is installed on the inverter, the inverter is powered, and no alarms are displayed on the inverter's keypad.</li> </ul>
NPU cannot update parameters	The NPU times out while resetting the card after clicking the "Submit" button	<ul style="list-style-type: none"> <li>Ensure that the inverter is configured to accept parameter writes from the option card. Refer to section 4.2.</li> </ul>

Problem	Symptom	Solution
Studio cannot discover the card	The studio does not display the card under "Online Devices"	<ul style="list-style-type: none"> <li>• Confirm that the card is running normally and connected via USB.</li> <li>• Confirm that the status LED blinks the green/red startup sequence when power is first applied.</li> <li>• Confirm that the status LED blinks green while a USB cable is connected.</li> <li>• Reinstall the Configuration Studio and run the USB driver installer again.</li> </ul>
Firmware-generated error	"STATUS" LED is flashing red. The number of times the LED flashes indicates an error code.	<ul style="list-style-type: none"> <li>• 6 flashes indicates the card is in USB to Serial Pass-Through mode.</li> <li>• 7 flashes indicates invalid configuration parameters. Ensure that the configuration loaded on the option card using the Mitsubishi Configuration Studio contains a definition for the selected protocol set via <i>Pr. 1305</i>. Refer to section 4.1.3.</li> <li>• Record the error code blinking pattern and contact technical support for further assistance.</li> </ul>

## REVISIONS

[illegible]