



Technical Document

NiagaraAX EIBnet/IP Driver User Guide

Updated: March 19, 2008



NiagaraAX EIBnet/IP Driver User Guide

Copyright © 2008 Tridium, Inc.
All rights reserved.
3951 Westerre Pkwy, Suite 350
Richmond
Virginia
23233
U.S.A.

Copyright Notice

The software described herein is furnished under a license agreement and may be used only in accordance with the terms of the agreement.

This document may not, in whole or in part, be copied, photocopied, reproduced, translated, or reduced to any electronic medium or machine-readable form without prior written consent from Tridium, Inc.

The confidential information contained in this document is provided solely for use by Tridium employees, licensees, and system owners; and is not to be released to, or reproduced for, anyone else; neither is it to be used for reproduction of this Control System or any of its components.

All rights to revise designs described herein are reserved. While every effort has been made to assure the accuracy of this document, Tridium shall not be held responsible for damages, including consequential damages, arising from the application of the information contained herein. Information and specifications published here are current as of the date of this publication and are subject to change without notice.

The release and technology contained herein may be protected by one or more U.S. patents, foreign patents, or pending applications.

Trademark Notices

BACnet and ASHRAE are registered trademarks of American Society of Heating, Refrigerating and Air-Conditioning Engineers. Microsoft and Windows are registered trademarks, and Windows NT, Windows 2000, Windows XP Professional, and Internet Explorer are trademarks of Microsoft Corporation. Java and other Java-based names are trademarks of Sun Microsystems Inc. and refer to Sun's family of Java-branded technologies. Mozilla and Firefox are trademarks of the Mozilla Foundation. Echelon, LON, LonMark, LonTalk, and LonWorks are registered trademarks of Echelon Corporation. Tridium, JACE, Niagara Framework, NiagaraAX and Vykon are registered trademarks, and Workbench, WorkPlaceAX, and AXSupervisor, are trademarks of Tridium Inc. All other product names and services mentioned in this publication that is known to be trademarks, registered trademarks, or service marks are the property of their respective owners. The software described herein is furnished under a license agreement and may be used only in accordance with the terms of the agreement.

NiagaraAX EIBnet/IP Driver User Guide

19 March 2008

This documents usage of the EIBnet/IP (KNX-IP) driver for the NiagaraAX framework.

COMPATIBILITY	4
EIBNET/IP SPECIFICATIONS.....	4
PLATFORMS SUPPORTED	4
MAPPING OF KNX DATA TYPES TO PROXY EXTENSIONS	5
<i>Data Conversions between Data Type and Numeric.....</i>	<i>6</i>
<i>Data Conversions between Data Type and Boolean</i>	<i>9</i>
<i>Data Conversions between Data Type and Enum</i>	<i>12</i>
<i>Data Conversions between Data Type and String.....</i>	<i>15</i>
INSTALLATION.....	19
QUICK START.....	20
EIBNET/IP COMPONENT GUIDES.....	21
EIBNETIPNETWORK	21
EIBNETIPDEVICE	23
KNXNUMERICPROXYEXT	25
KNXBOOLEANPROXYEXT	25
KNXENUMPROXYEXT	25
KNXSTRINGPROXYEXT	25
EIBNET/IP VIEWS	28
DEVICE MANAGER.....	28
POINT MANAGER	30
SPECIAL CONSIDERATIONS.....	35
ACTIONS MAY BE ADDED TO READ-ONLY POINTS.....	35
<i>Switch Actions “Trigger On”, “Trigger Off”, and “Toggle”</i>	<i>35</i>
<i>Switch Control Actions “Manual On”, “Manual Off”, and “Auto”</i>	<i>36</i>
<i>Dimming Actions “Step Up”, “Step Down”, and “Break”</i>	<i>36</i>
<i>Counter Actions “Increment”, “Decrement”, “Preset”, and “Reset”</i>	<i>37</i>
<i>Date Action “New Date”</i>	<i>37</i>
<i>Time Action “New Time”</i>	<i>38</i>
OTHER SPECIAL CIRCUMSTANCES	38
<i>Data Type 16.000 String ASCII on a Boolean Point</i>	<i>38</i>
<i>Data Type 16.001 String 8859.1 on a Boolean Point.....</i>	<i>38</i>
DOCUMENT CHANGE LOG	39

Compatibility

The following sections provide compatibility details on the EIBnet/IP driver.

- [EIBnet/IP Specifications](#)
- [Platforms](#)
- [Mapping of KNX Data Types to Proxy Extensions](#)

EIBnet/IP Specifications

The EIBnet/IP driver was written to be compliant with KNX specifications Volume 3: System Specifications, Part 8:EIBnet/IP version 1.0 (Draft Proposal). This driver implements an EIBnet/IP client that can access services and data in a KNX system via an EIBnet/IP server.

The following list details support contained in the initial driver version:

EIBnet/IP – Core

EIBnet/IP – Device Management

Not Supported

EIBnet/IP – Tunneling

Supports tunneling on KNX Data Link Layer

Does not support cEMI Raw

Does not support KNX Busmonitor

EIBnet/IP –Routing

Not Supported

EIBnet/IP – Remote Logging

Not Supported

EIBnet/IP – Remote Configuration

Not Supported

EIBnet/IP – Object Server

Not Supported

Platforms Supported

The EIBNET/IP driver will function on all AX platforms that support IP communications.

Mapping of KNX Data Types to Proxy Extensions

The EIBnet/IP driver allows the creation of any of the following ControlPoint/KnxPrxyExt combinations:

- NumericPoint / KnxNumericProxyExt
- NumericWritable / KnxNumericProxyExt
- BooleanPoint / KnxBooleanProxyExt
- BooleanWritable / KnxBooleanProxyExt
- StringPoint / KnxStringProxyExt
- StringWritable / KnxStringProxyExt
- EnumPoint / KnxEnumProxyExt
- EnumWritable / KnxEnumProxyExt

In addition, the behavior of the point or proxy extension is modified depending on the selected data type in the KnxProxyExt.

The following sections, sorted by data type, show how each of the KnxProxyExt types behave. Note that in some instances of read-only base point types (NumericPoint, BooleanPoint, StringPoint, and EnumPoint), dynamic actions may be added to the base point depending on the selected data type of the proxy extension.

- **Data Conversions between Data Type and Numeric**
- **Data Conversions between Data Type and Boolean**
- **Data Conversions between Data Type and Enum**
- **Data Conversions between Data Type and String**

Data Conversions between Data Type and Numeric

Table 1 Data Conversions for NumericPoint and NumericWritable			
Data Type	Conversion from KNX data to a Numeric value (Reads)	Conversion from a Numeric value to KNX data (Writes)	Actions Added to Read-Only Parent Point
UNKNOWN_1BIT	convert to value 0 or 1	if 0 write 0x00(false), else write 0x01(true)	—
1.001_SWITCH	convert to value 0 or 1	if 0 write 0x00(false), else write 0x01(true)	—
1.002_BOOL	convert to value 0 or 1	if 0 write 0x00(false), else write 0x01(true)	—
1.003_ENABLE	convert to value 0 or 1	if 0 write 0x00(false), else write 0x01(true)	—
1.004_RAMP	convert to value 0 or 1	if 0 write 0x00(false), else write 0x01(true)	—
1.005_ALARM	convert to value 0 or 1	if 0 write 0x00(false) else write 0x01(true)	—
1.006_BINARY_VALUE	convert to value 0 or 1	if 0 write 0x00(false), else write 0x01(true)	—
1.007_STEP	convert to value 0 or 1	if 0 write 0x00(false), else write 0x01(true)	—
1.008_UP_DOWN	convert to value 0 or 1	if 0 write 0x00(false), else write 0x01(true)	—
1.009_OPEN_CLOSE	convert to value 0 or 1	if 0 write 0x00(false), else write 0x01(true)	—
1.010_START	convert to value 0 or 1	if 0 write 0x00(false), else write 0x01(true)	—
1.011_STATE	convert to value 0 or 1	if 0 write 0x00(false), else write 0x01(true)	—
1.012_INVERT	convert to value 0 or 1	if 0 write 0x00(false), else write 0x01(true)	—
1.013_DIM_SEND_STYLE	convert to value 0 or 1	if 0 write 0x00(false), else write 0x01(true)	—
1.014_INPUT_SOURCE	convert to value 0 or 1	if 0 write 0x00(false) else write 0x01(true)	—
UNKNOWN_2BIT	convert to value in range 0 - 3	if value in range 0 - 3, write the value	—
2.001_SWITCH_CONTROL	convert least significant bit only to value 0 or 1	if value in range 0 - 3, write the value	—
2.002_BOOL_CONTROL	convert least significant bit only to value 0 or 1	if value in range 0 - 3, write the value	—
2.003_ENABLE_CONTROL	convert least significant bit only to value 0 or 1	if value in range 0 - 3, write the value	—
2.004_RAMP_CONTROL	convert least significant bit only to value 0 or 1	if value in range 0 - 3, write the value	—
2.005_ALARM_CONTROL	convert least significant bit only to value 0 or 1	if value in range 0 - 3, write the value	—
2.006_BINARY_VALUE_CONTROL	convert least significant bit only to value 0 or 1	if value in range 0 - 3, write the value	—

Table 1 Data Conversions for NumericPoint and NumericWritable			
Data Type	Conversion from KNX data to a Numeric value (Reads)	Conversion from a Numeric value to KNX data (Writes)	Actions Added to Read-Only Parent Point
2.007_STEP_CONTROL	convert least significant bit only to value 0 or 1	if value in range 0 - 3, write the value	—
2.008_DIRECTION_1CONTROL	convert least significant bit only to value 0 or 1	if value in range 0 - 3, write the value	—
2.009_DIRECTION_2CONTROL	convert least significant bit only to value 0 or 1	if value in range 0 - 3, write the value	—
2.010_START_CONTROL	convert least significant bit only to value 0 or 1	if value in range 0 - 3, write the value	—
2.011_STATE_CONTROL	convert least significant bit only to value 0 or 1	if value in range 0 - 3, write the value	—
2.012_INVERT_CONTROL	convert least significant bit only to value 0 or 1	if value in range 0 - 3, write the value	—
UNKNOWN_4BIT	convert to value 0 -15	if value in range 0 -15, write the value	—
3.007_CONTROL_DIMMING	convert to value 0 -15	if value in range 0 -15, write the value	Increase/Decrease/Break
3.008_CONTROL_BLINDS	convert to value 0 -15	if value in range 0 -15, write the value	Up/Down/Break
3.009_MODE_BOILER	convert mode bits to value 0 or 1 or 2	NOT SUPPORTED, write not allowed	—
UNKNOWN_1BYTE	unsigned integer	write the value as unsigned	—
4.001_CHAR_ASCII	if character is a hex digit character, convert to number in range 0 - 15	convert value 0 – 15 to '0' to '9' or 'A' to 'F'	—
4.002_CHAR_8859	if character is a hex digit character, convert to number in range 0 - 15	convert value 0 – 15 to '0' to '9' or 'A' to 'F'	—
5.001_SCALING	convert to number in range 0 -100	write the value	Increment, Decrement, Reset, Preset
5.003_ANGLE	convert to number in range 0 -360	write the value	Increment, Decrement, Reset, Preset
5.004_REL_POS_VALVE	convert to number in range 0 -255	write the value	—
5.010_UNSIGNED_COUNT	unsigned integer	write the value	Increment, Decrement, Reset, Preset
6.010_SIGNED_COUNT	signed integer	write the value	Increment, Decrement, Reset, Preset
6.020_STATUS_MODE_3	convert value 0 or 1 or 2	NOT SUPPORTED, write not allowed	—
UNKNOWN_2BYTE	unsigned integer	write the value as unsigned	—
7.001_UNSIGNED_COUNT	unsigned integer	write the value	Increment, Decrement, Reset, Preset
7.010_PROP_DATA_TYPE	unsigned integer	write the value	—

Table 1 Data Conversions for NumericPoint and NumericWritable			
Data Type	Conversion from KNX data to a Numeric value (Reads)	Conversion from a Numeric value to KNX data (Writes)	Actions Added to Read-Only Parent Point
8.001_SIGNED_COUNT	signed integer	write the value	Increment, Decrement, Reset, Preset
9.001_TEMP	signed float	write the value	Increment, Decrement, Reset, Preset
9.002_TEMPD	signed float	write the value	Increment, Decrement, Reset, Preset
9.003_TEMP_A	signed float	write the value	Increment, Decrement, Reset, Preset
9.004_LUX	signed float	write the value	Increment, Decrement, Reset, Preset
9.005_WSP	signed float	write the value	Increment, Decrement, Reset, Preset
9.006_PRES	signed float	write the value	Increment, Decrement, Reset, Preset
9.010_TIME_1	signed float	write the value	Increment, Decrement, Reset, Preset
9.011_TIME_2	signed float	write the value	Increment, Decrement, Reset, Preset
9.020_VOLT	signed float	write the value	Increment, Decrement, Reset, Preset
9.021_CURR	signed float	write the value	Increment, Decrement, Reset, Preset
UNKNOWN_3BYTE	unsigned integer	write the value as unsigned	—
10.001_TIME_OF_DAY	NOT SUPPORTED, generates a fault condition	NOT SUPPORTED, write not allowed	—
11.001_DATE	NOT SUPPORTED, generates a fault condition	NOT SUPPORTED, write not allowed	—
UNKNOWN_4BYTE	unsigned integer	write the value as unsigned	—
12.001_UNSIGNED_COUNT	unsigned integer	write the value	Increment, Decrement, Reset, Preset
13.001_SIGNED_COUNT	signed integer	write the value	Increment, Decrement, Reset, Preset
14.000_ACCELERATION thru 14.079_WORK	signed float	write the value	—
15.000_ACCESS_DATA	NOT SUPPORTED, generates a fault condition	NOT SUPPORTED, write not allowed	—
16.000_STRING_ASCII	convert string to base 10 number if possible	convert base 10 to String, truncated at 14 characters	—
16.001_STRING_8859	convert string to base 10 number if possible	convert base 10 to String, truncated at 14 characters	—

Data Conversions between Data Type and Boolean

Table 2 Data Conversions for BooleanPoint and BooleanWritable			
Data Type	Conversion from KNX data to a Boolean value (Reads)	Conversion from a Boolean value to KNX data (Writes)	Actions Added to Read-Only Parent Point
UNKNOWN_1BIT	convert to true or false	if true write 0x00(false), else write 0x01(true)	Trigger On, Trigger Off, Toggle
1.001_SWITCH	convert to true or false	if true write 0x00(false), else write 0x01(true)	Trigger On, Trigger Off, Toggle
1.002_BOOL	convert to true or false	if true write 0x00(false), else write 0x01(true)	Trigger On, Trigger Off, Toggle
1.003_ENABLE	convert to true or false	if true write 0x00(false), else write 0x01(true)	Trigger On, Trigger Off, Toggle
1.004_RAMP	convert to true or false	if true write 0x00(false), else write 0x01(true)	Trigger On, Trigger Off, Toggle
1.005_ALARM	convert to true or false	if true write 0x00(false), else write 0x01(true)	Trigger On, Trigger Off, Toggle
1.006_BINARY_VALUE	convert to true or false	if true write 0x00(false), else write 0x01(true)	Trigger On, Trigger Off, Toggle
1.007_STEP	convert to true or false	if true write 0x00(false), else write 0x01(true)	Trigger On, Trigger Off, Toggle
1.008_UP_DOWN	convert to true or false	if true write 0x00(false), else write 0x01(true)	Trigger On, Trigger Off, Toggle
1.009_OPEN_CLOSE	convert to true or false	if true write 0x00(false), else write 0x01(true)	Trigger On, Trigger Off, Toggle
1.010_START	convert to true or false	if true write 0x00(false), else write 0x01(true)	Trigger On, Trigger Off, Toggle
1.011_STATE	convert to true or false	if true write 0x00(false), else write 0x01(true)	Trigger On, Trigger Off, Toggle
1.012_INVERT	convert to true or false	if true write 0x00(false), else write 0x01(true)	Trigger On, Trigger Off, Toggle
1.013_DIM_SEND_STYLE	convert to true or false	if true write 0x00(false), else write 0x01(true)	Trigger On, Trigger Off, Toggle
1.014_INPUT_SOURCE	convert to true or false	if true write 0x00(false), else write 0x01(true)	Trigger On, Trigger Off, Toggle
UNKNOWN_2BIT	convert least significant bit (only) to true or false	v bit set to 0 or 1 by write value, c bit set to 0 or 1 by override status of write value	Manual On, Manual Off, Auto
2.001_SWITCH_CONTROL	convert least significant bit (only) to true or false	v bit set to 0 or 1 by write value, c bit set to 0 or 1 by override status of write value	Manual On, Manual Off, Auto
2.002_BOOL_CONTROL	convert least significant bit (only) to true or false	v bit set to 0 or 1 by write value, c bit set to 0 or 1 by override status of write value	Manual On, Manual Off, Auto
2.003_ENABLE_CONTROL	convert least significant bit (only) to true or false	v bit set to 0 or 1 by write value, c bit set to 0 or 1 by override status of write value	Manual On, Manual Off, Auto
2.004_RAMP_CONTROL	convert least significant bit (only) to true or false	v bit set to 0 or 1 by write value, c bit set to 0 or 1 by override status of write value	Manual On, Manual Off, Auto
2.005_ALARM_CONTROL	convert least significant bit (only) to true or false	v bit set to 0 or 1 by write value, c bit set to 0 or 1 by override status of write value	Manual On, Manual Off, Auto
2.006_BINARY_VALUE_CONTROL	convert least significant bit (only) to true or false	v bit set to 0 or 1 by write value, c bit set to 0 or 1 by override status of write value	Manual On, Manual Off, Auto
2.007_STEP_CONTROL	convert least significant bit (only) to true or false	v bit set to 0 or 1 by write value, c bit set to 0 or 1 by override status of write value	Manual On, Manual Off, Auto
2.008_DIRECTION_1CONTROL	convert least significant bit (only) to true or false	v bit set to 0 or 1 by write value, c bit set to 0 or 1 by override status of write value	Manual On, Manual Off, Auto

Table 2 Data Conversions for BooleanPoint and BooleanWritable			
Data Type	Conversion from KNX data to a Boolean value (Reads)	Conversion from a Boolean value to KNX data (Writes)	Actions Added to Read-Only Parent Point
2.009_DIRECTION_2CONTROL	convert least significant bit (only) to true or false	v bit set to 0 or 1 by write value, c bit set to 0 or 1 by override status of write value	Manual On, Manual Off, Auto
2.010_START_CONTROL	convert least significant bit (only) to true or false	v bit set to 0 or 1 by write value, c bit set to 0 or 1 by override status of write value	Manual On, Manual Off, Auto
2.011_STATE_CONTROL	convert least significant bit (only) to true or false	v bit set to 0 or 1 by write value, c bit set to 0 or 1 by override status of write value	Manual On, Manual Off, Auto
2.012_INVERT_CONTROL	convert least significant bit (only) to true or false	v bit set to 0 or 1 by write value, c bit set to 0 or 1 by override status of write value	Manual On, Manual Off, Auto
UNKNOWN_4BIT	convert to number 0 -15, return true if number > 0	NOT SUPPORTED, write not allowed	—
3.007_CONTROL_DIMMING	NOT SUPPORTED, generates a fault condition	NOT SUPPORTED, write not allowed	—
3.008_CONTROL_BLINDS	NOT SUPPORTED, generates a fault condition	NOT SUPPORTED, write not allowed	—
3.009_MODE_BOILER	NOT SUPPORTED, generates a fault condition	NOT SUPPORTED, write not allowed	—
UNKNOWN_1BYTE	convert to number, return true if number > 0	if false write 0x00(0), else write 0xFF(255)	—
4.001_CHAR_ASCII	If character is '0' return false, If character is '1' return true, Otherwise, fault.	if false write 0x30('0'), else write 0x31('1')	—
4.002_CHAR_8859	If character is '0' return false, If character is '1' return true, Otherwise, fault.	if false write 0x30('0') else write 0x31('1')	—
5.001_SCALING	convert to number, return false if number = 0, else return true	if false write 0x00(0%), else write 0xFF(100%)	—
5.003_ANGLE	convert to number, return false if number = 0, else return true	if false write 0x00(0), else write 0x01(1)	—
5.004_REL_POS_VALVE	convert to number, return false if number = 0, else return true	if false write 0x00(0), else write 0x64(100)	—
5.010_UNSIGNED_COUNT	convert to number, return false if number = 0, else return true	if false write 0, else write 1	—
6.010_SIGNED_COUNT	convert to number, return false if number = 0, else return true	if false write 0, else write 1	—
6.020_STATUS_MODE_3	convert value 0 or 1 or 2, return false if number = 0, else return true	NOT SUPPORTED, write not allowed	—
UNKNOWN_2BYTE	convert to number, return false if number = 0, else return true	if false write 0, else write 1	—
7.001_UNSIGNED_COUNT	convert to number, return false if number = 0, else return true	if false write 0, else write 1	—
7.010_PROP_DATA_TYPE	convert to number, return false if number = 0, else return true	if false write 0, else write 1	—
8.001_SIGNED_COUNT	convert to number, return false if number = 0, else return true	if false write 0, else write 1	—
9.001_TEMP	convert to number, return false if number = 0, else return true	if false write 0, else write 1	—
9.002_TEMPD	convert to number, return false if number = 0, else return true	if false write 0, else write 1	—
9.003_TEMP_A	convert to number, return false if number = 0, else return true	if false write 0, else write 1	—
9.004_LUX	convert to number, return false if number = 0, else return true	if false write 0, else write 1	—

Table 2 Data Conversions for BooleanPoint and BooleanWritable			
Data Type	Conversion from KNX data to a Boolean value (Reads)	Conversion from a Boolean value to KNX data (Writes)	Actions Added to Read-Only Parent Point
9.005_WSP	convert to number, return false if number = 0, else return true	if false write 0, else write 1	—
9.006_PRES	convert to number, return false if number = 0, else return true	if false write 0, else write 1	—
9.010_TIME_1	convert to number, return false if number = 0, else return true	if false write 0, else write 1	—
9.011_TIME_2	convert to number, return false if number = 0, else return true	if false write 0, else write 1	—
9.020_VOLT	convert to number, return false if number = 0, else return true	if false write 0 else write 1	—
9.021_CURR	convert to number, return false if number = 0, else return true	if false write 0, else write 1	—
UNKNOWN_3BYTE	convert to number, return false if number = 0, else return true	if false write 0, else write 1	—
10.001_TIME_OF_DAY	NOT SUPPORTED, generates a fault condition	NOT SUPPORTED, write not allowed	—
11.001_DATE	NOT SUPPORTED, generates a fault condition	NOT SUPPORTED, write not allowed	—
UNKNOWN_4BYTE	convert to number, return false if number = 0, else return true	if false write 0, else write 1	—
12.001_UNSIGNED_COUNT	convert to number, return false if number = 0, else return true	if false write 0, else write 1	—
13.001_SIGNED_COUNT	convert to number, return false if number = 0, else return true	if false write 0, else write 1	—
14.000_ACCELERATION thru 14.079_WORK	convert to number, return false if number = 0, else return true	if false write 0, else write 1	—
15.000_ACCESS_DATA	NOT SUPPORTED, generates a fault condition	NOT SUPPORTED, write not allowed	—
16.000_STRING_ASCII	if string = lexicon(stringToBoolean.trueTxt) return true, if string = lexicon(stringToBoolean.falseTxt) return false, else fault	if false return text from slot "falseTxt", else return text from slot "trueTxt"	—
16.001_STRING_8859	if string = lexicon(stringToBoolean.trueTxt) return true, if string = lexicon(stringToBoolean.falseTxt) return false, else fault	if false return text from slot "falseTxt", else return text from slot "trueTxt"	—

Data Conversions between Data Type and Enum

Table 3 Data Conversions for EnumPoint and EnumWritable			
Data Type	Conversion from KNX data to an Enum value (Reads)	Conversion from an Enum value to KNX data (Writes)	Actions Added to Read-Only Parent Point
UNKNOWN_1BIT	enum ordinal 0 or 1	if ordinal =0 write 0x00(false), else write 0x01(true)	—
1.001_SWITCH	enum ordinal 0 or 1	if ordinal =0 write 0x00(false), else write 0x01(true)	—
1.002_BOOL	enum ordinal 0 or 1	if ordinal =0 write 0x00(false), else write 0x01(true)	—
1.003_ENABLE	enum ordinal 0 or 1	if ordinal =0 write 0x00(false), else write 0x01(true)	—
1.004_RAMP	enum ordinal 0 or 1	if ordinal =0 write 0x00(false), else write 0x01(true)	—
1.005_ALARM	enum ordinal 0 or 1	if ordinal =0 write 0x00(false), else write 0x01(true)	—
1.006_BINARY_VALUE	enum ordinal 0 or 1	if ordinal =0 write 0x00(false), else write 0x01(true)	—
1.007_STEP	enum ordinal 0 or 1	if ordinal =0 write 0x00(false), else write 0x01(true)	—
1.008_UP_DOWN	enum ordinal 0 or 1	if ordinal =0 write 0x00(false), else write 0x01(true)	—
1.009_OPEN_CLOSE	enum ordinal 0 or 1	if ordinal =0 write 0x00(false), else write 0x01(true)	—
1.010_START	enum ordinal 0 or 1	if ordinal =0 write 0x00(false), else write 0x01(true)	—
1.011_STATE	enum ordinal 0 or 1	if ordinal =0 write 0x00(false), else write 0x01(true)	—
1.012_INVERT	enum ordinal 0 or 1	if ordinal =0 write 0x00(false), else write 0x01(true)	—
1.013_DIM_SEND_STYLE	enum ordinal 0 or 1	if ordinal =0 write 0x00(false), else write 0x01(true)	—
1.014_INPUT_SOURCE	enum ordinal 0 or 1	if ordinal =0 write 0x00(false), else write 0x01(true)	—
UNKNOWN_2BIT	enum ordinal 0 - 3	if ordinal in range 0 - 3, write the ordinal value	—
2.001_SWITCH_CONTROL	enum ordinal 0 - 3	if ordinal in range 0 - 3, write the ordinal value	—
2.002_BOOL_CONTROL	enum ordinal 0 - 3	if ordinal in range 0 - 3, write the ordinal value	—
2.003_ENABLE_CONTROL	enum ordinal 0 - 3	if ordinal in range 0 - 3, write the ordinal value	—
2.004_RAMP_CONTROL	enum ordinal 0 - 3	if ordinal in range 0 - 3, write the ordinal value	—
2.005_ALARM_CONTROL	enum ordinal 0 - 3	if ordinal in range 0 - 3, write the ordinal value	—
2.006_BINARY_VALUE_CONTROL	enum ordinal 0 - 3	if ordinal in range 0 - 3, write the ordinal value	—
2.007_STEP_CONTROL	enum ordinal 0 - 3	if ordinal in range 0 - 3, write the ordinal value	—
2.008_DIRECTION_1CONTROL	enum ordinal 0 - 3	if ordinal in range 0 - 3, write the ordinal value	—

Table 3 Data Conversions for EnumPoint and EnumWritable			
Data Type	Conversion from KNX data to an Enum value (Reads)	Conversion from an Enum value to KNX data (Writes)	Actions Added to Read-Only Parent Point
2.009_DIRECTION_2CONTROL	enum ordinal 0 - 3	if ordinal in range 0 - 3, write the ordinal value	—
2.010_START_CONTROL	enum ordinal 0 - 3	if ordinal in range 0 - 3, write the ordinal value	—
2.011_STATE_CONTROL	enum ordinal 0 - 3	if ordinal in range 0 - 3, write the ordinal value	—
2.012_INVERT_CONTROL	enum ordinal 0 - 3	if ordinal in range 0 - 3, write the ordinal value	—
UNKNOWN_4BIT	enum ordinal 0 - 15	if ordinal in range 0 - 15, write the ordinal value	—
3.007_CONTROL_DIMMING	enum ordinal 0 - 15	if ordinal in range 0 - 15, write the ordinal value	—
3.008_CONTROL_BLINDS	enum ordinal 0 - 15	if ordinal in range 0 - 15, write the ordinal value	—
3.009_MODE_BOILER	enum ordinal 0 or 1 or 2	if ordinal in range 0,1,2, write the mode 0, 1, or 2 code	—
UNKNOWN_1BYTE	unsigned integer used as enum ordinal	write ordinal value	—
4.001_CHAR_ASCII	if character is a hex digit character, convert to number in range 0 - 15, and use as ordinal	convert enum ordinal value 0 - 15 to '0' to '9' or 'A' to 'F'	—
4.002_CHAR_8859	if character is a hex digit character, convert to number in range 0 - 15, and use as ordinal	convert enum ordinal value 0 - 15 to '0' to '9' or 'A' to 'F'	—
5.001_SCALING	enum ordinal in range 0 -100	write ordinal value	—
5.003_ANGLE	enum ordinal in range 0 -360	write ordinal value	—
5.004_REL_POS_VALVE	enum ordinal in range 0 -255	write ordinal value	—
5.010_UNSIGNED_COUNT	use value as enum ordinal	write ordinal value	—
6.010_SIGNED_COUNT	use value as enum ordinal	write ordinal value	—
6.020_STATUS_MODE_3	enum ordinal 0 or 1 or 2	if ordinal in range 0,1,2, write the mode 0, 1, or 2 code. Set status bits to 0	—
UNKNOWN_2BYTE	use value as enum ordinal	write ordinal value	—
7.001_UNSIGNED_COUNT	use value as enum ordinal	write ordinal value	—
7.010_PROP_DATA_TYPE	use value as enum ordinal	write ordinal value	—
8.001_SIGNED_COUNT	use value as enum ordinal	write ordinal value	—
9.001_TEMP	use integer portion of value as enum ordinal	write ordinal value	—
9.002_TEMPD	use integer portion of value as enum ordinal	write ordinal value	—
9.003_TEMP_A	use integer portion of value as enum ordinal	write ordinal value	—

Table 3 Data Conversions for EnumPoint and EnumWritable			
Data Type	Conversion from KNX data to an Enum value (Reads)	Conversion from an Enum value to KNX data (Writes)	Actions Added to Read-Only Parent Point
9.004_LUX	use integer portion of value as enum ordinal	write ordinal value	—
9.005_WSP	use integer portion of value as enum ordinal	write ordinal value	—
9.006_PRES	use integer portion of value as enum ordinal	write ordinal value	—
9.010_TIME_1	use integer portion of value as enum ordinal	write ordinal value	—
9.011_TIME_2	use integer portion of value as enum ordinal	write ordinal value	—
9.020_VOLT	use integer portion of value as enum ordinal	write ordinal value	—
9.021_CURR	use integer portion of value as enum ordinal	write ordinal value	—
UNKNOWN_3BYTE	use value as enum ordinal	write ordinal value	—
10.001_TIME_OF_DAY	NOT SUPPORTED, generates a fault condition	NOT SUPPORTED, write not allowed	—
11.001_DATE	NOT SUPPORTED, generates a fault condition	NOT SUPPORTED, write not allowed	—
UNKNOWN_4BYTE	use value as enum ordinal	write ordinal value	—
12.001_UNSIGNED_COUNT	use value as enum ordinal	write ordinal value	—
13.001_SIGNED_COUNT	use value as enum ordinal	write ordinal value	—
14.000_ACCELERATION thru 14.079_WORK	use integer portion of value as enum ordinal	write ordinal value	—
15.000_ACCESS_DATA	NOT SUPPORTED, generates a fault condition	NOT SUPPORTED, write not allowed	—
16.000_STRING_ASCII	use string as enum tag, fault if does not match any tag	write enum tag truncated at 14 characters	—
16.001_STRING_8859	use string as enum tag, fault if does not match any tag	write enum tag truncate at 14 characters	—

Data Conversions between Data Type and String

Table 4 Data Conversions for StringPoint and StringWritable			
Data Type	Conversion from KNX data to a String value (Reads)	Conversion from a String value to KNX data (Writes)	Actions Added to Read-Only Parent Point
UNKNOWN_1BIT	<hex string>	Convert string to a number, and if 0 write 0x00(false), else write 0x01(true)	—
1.001_SWITCH	Device facets trueText or falseText, OR if facets don't exist then "0" or "1".	If text=device facets trueText write 1, If text=device faces falseText write 0, If text="1" write 1, if text="0" write 0.	—
1.002_BOOL	Device facets trueText or falseText, OR if facets don't exist then "0" or "1".	If text=device facets trueText write 1, If text=device faces falseText write 0, If text="1" write 1, if text="0" write 0.	—
1.003_ENABLE	Device facets trueText or falseText, OR if facets don't exist then "0" or "1".	If text=device facets trueText write 1, If text=device faces falseText write 0, If text="1" write 1, if text="0" write 0.	—
1.004_RAMP	Device facets trueText or falseText, OR if facets don't exist then "0" or "1".	If text=device facets trueText write 1, If text=device faces falseText write 0, If text="1" write 1, if text="0" write 0.	—
1.005_ALARM	Device facets trueText or falseText, OR if facets don't exist then "0" or "1".	If text=device facets trueText write 1, If text=device faces falseText write 0, If text="1" write 1, if text="0" write 0.	—
1.006_BINARY_VALUE	Device facets trueText or falseText, OR if facets don't exist then "0" or "1".	If text=device facets trueText write 1, If text=device faces falseText write 0, If text="1" write 1, if text="0" write 0.	—
1.007_STEP	Device facets trueText or falseText, OR if facets don't exist then "0" or "1".	If text=device facets trueText write 1, If text=device faces falseText write 0, If text="1" write 1, if text="0" write 0.	—
1.008_UP_DOWN	Device facets trueText or falseText, OR if facets don't exist then "0" or "1".	If text=device facets trueText write 1, If text=device faces falseText write 0, If text="1" write 1, if text="0" write 0.	—
1.009_OPEN_CLOSE	Device facets trueText or falseText, OR if facets don't exist then "0" or "1".	If text=device facets trueText write 1, If text=device faces falseText write 0, If text="1" write 1, if text="0" write 0.	—
1.010_START	Device facets trueText or falseText, OR if facets don't exist then "0" or "1".	If text=device facets trueText write 1, If text=device faces falseText write 0, If text="1" write 1, if text="0" write 0.	—
1.011_STATE	Device facets trueText or falseText, OR if facets don't exist then "0" or "1".	If text=device facets trueText write 1, If text=device faces falseText write 0, If text="1" write 1, if text="0" write 0.	—
1.012_INVERT	Device facets trueText or falseText, OR if facets don't exist then "0" or "1".	If text=device facets trueText write 1, If text=device faces falseText write 0, If text="1" write 1, if text="0" write 0.	—
1.013_DIM_SEND_STYLE	Device facets trueText or falseText, OR if facets don't exist then "0" or "1".	If text=device facets trueText write 1, If text=device faces falseText write 0, If text="1" write 1, if text="0" write 0.	—
1.014_INPUT_SOURCE	Device facets trueText or falseText, OR if facets don't exist then "0" or "1".	If text=device facets trueText write 1, If text=device faces falseText write 0, If text="1" write 1, if text="0" write 0.	—
UNKNOWN_2BIT	<hex string>	Attempt to convert string to int using base 16, write not allowed if not convertible or out of range 0 – 3.	—
2.001_SWITCH_CONTROL	"no control:<hex string>" or "control at function value 0:<hex string>" or "control at function value 1:<hex string>"	NOT SUPPORTED, write not allowed.	—
2.002_BOOL_CONTROL	"no control:<hex string>" or "control at function value 0:<hex string>" or "control at function value 1:<hex string>"	NOT SUPPORTED, write not allowed	—
2.003_ENABLE_CONTROL	"no control:<hex string>" or "control at function value 0:<hex string>" or "control at function value 1:<hex string>"	NOT SUPPORTED, write not allowed	—

Table 4 Data Conversions for StringPoint and StringWritable			
Data Type	Conversion from KNX data to a String value (Reads)	Conversion from a String value to KNX data (Writes)	Actions Added to Read-Only Parent Point
2.004_RAMP_CONTROL	"no control:<hex string>" or "control at function value 0:<hex string>" or "control at function value 1:<hex string>"	NOT SUPPORTED, write not allowed	—
2.005_ALARM_CONTROL	"no control:<hex string>" or "control at function value 0:<hex string>" or "control at function value 1:<hex string>"	NOT SUPPORTED, write not allowed	—
2.006_BINARY_VALUE_CONTROL	"no control:<hex string>" or "control at function value 0:<hex string>" or "control at function value 1:<hex string>"	NOT SUPPORTED, write not allowed	—
2.007_STEP_CONTROL	"no control:<hex string>" or "control at function value 0:<hex string>" or "control at function value 1:<hex string>"	NOT SUPPORTED, write not allowed	—
2.008_DIRECTION_1CONTROL	"no control:<hex string>" or "control at function value 0:<hex string>" or "control at function value 1:<hex string>"	NOT SUPPORTED, write not allowed	—
2.009_DIRECTION_2CONTROL	"no control:<hex string>" or "control at function value 0:<hex string>" or "control at function value 1:<hex string>"	NOT SUPPORTED, write not allowed	—
2.010_START_CONTROL	"no control:<hex string>" or "control at function value 0:<hex string>" or "control at function value 1:<hex string>"	NOT SUPPORTED, write not allowed	—
2.011_STATE_CONTROL	"no control:<hex string>" or "control at function value 0:<hex string>" or "control at function value 1:<hex string>"	NOT SUPPORTED, write not allowed	—
2.012_INVERT_CONTROL	"no control:<hex string>" or "control at function value 0:<hex string>" or "control at function value 1:<hex string>"	NOT SUPPORTED, write not allowed	—
UNKNOWN_4BIT	<hex string>	Attempt to convert string to int using base 16, write not allowed if not convertible or out of range 0 -15	—
3.007_CONTROL_DIMMING	"lexicon(dimming.increase) + ":" + step value" OR "lexicon(dimming.decrease) + ":" + step value"	NOT SUPPORTED, write not allowed	—
3.008_CONTROL_BLINDS	"lexicon(blinds.up) + ":" + step value" OR "lexicon(blinds.down) + ":" + step value"	NOT SUPPORTED, write not allowed	—
3.009_MODE_BOILER	"lexicon(boiler.calculated OR boiler.fixed) + ":" + lexicon(boiler.mode0 OR boiler.mode1 OR boiler.mode2 OR boiler.undefined) + ":" + <hex string>"	NOT SUPPORTED, write not allowed	—
UNKNOWN_1BYTE	<hex string>	Attempt to convert string to int using base 16, write not allowed if not convertible or out of range 0 -255	—
4.001_CHAR_ASCII	single character	Write if single character, else write not allowed	—
4.002_CHAR_8859	single character	Write if single character, else write not allowed	—
5.001_SCALING	number as a string	Attempt to convert string to integer, write not allowed if not convertible or out	—

Table 4 Data Conversions for StringPoint and StringWritable			
Data Type	Conversion from KNX data to a String value (Reads)	Conversion from a String value to KNX data (Writes)	Actions Added to Read-Only Parent Point
		of range	
5.003_ANGLE	number as a string	Attempt to covert string to integer, write not allowed if not convertible or out of range	—
5.004_REL_POS_VALVE	number as a string	Attempt to covert string to integer, write not allowed if not convertible or out of range	—
5.010_UNSIGNED_COUNT	number as a string	Attempt to covert string to integer, write not allowed if not convertible or out of range	—
6.010_SIGNED_COUNT	number as a string	Attempt to covert string to integer, write not allowed if not convertible or out of range	—
6.020_STATUS_MODE_3	<hex string>	NOT SUPPORTED, write not allowed	—
UNKNOWN_2BYTE	<hex string>	"Attempt to covert string to number using base 16, write not allowed if not convertible or out of range"	—
7.001_UNSIGNED_COUNT	number as a string	Attempt to covert string to number using base 16, write not allowed if not convertible or out of range	—
7.010_PROP_DATA_TYPE	number as a string	Attempt to covert string to number using base 16, write not allowed if not convertible or out of range	—
8.001_SIGNED_COUNT	number as a string	Attempt to covert string to number using base 16, write not allowed if not convertible or out of range	—
9.001_TEMP	number as a string	Attempt to covert string to number using base 16, write not allowed if not convertible or out of range	—
9.002_TEMPD	number as a string	Attempt to covert string to number using base 16, write not allowed if not convertible or out of range	—
9.003_TEMP_A	number as a string	Attempt to covert string to number using base 16, write not allowed if not convertible or out of range	—
9.004_LUX	number as a string	Attempt to covert string to number using base 16, write not allowed if not convertible or out of range	—
9.005_WSP	number as a string	Attempt to covert string to number using base 16, write not allowed if not convertible or out of range	—
9.006_PRES	number as a string	Attempt to covert string to number using base 16, write not allowed if not convertible or out of range	—
9.010_TIME_1	number as a string	Attempt to covert string to number using base 16, write not allowed if not convertible or out of range	—
9.011_TIME_2	number as a string	Attempt to covert string to number using base 16, write not allowed if not convertible or out of range	—
9.020_VOLT	number as a string	Attempt to covert string to number using base 16, write not allowed if not convertible or out of range	—
9.021_CURR	number as a string	Attempt to covert string to number using base 16, write not allowed if not convertible or out of range	—
UNKNOWN_3BYTE	<hex string>	Attempt to covert string to number using base 16, write not allowed if not convertible or out of range	—

Table 4 Data Conversions for StringPoint and StringWritable			
Data Type	Conversion from KNX data to a String value (Reads)	Conversion from a String value to KNX data (Writes)	Actions Added to Read-Only Parent Point
10.001_TIME_OF_DAY	String time in format "hh:mm am/pm dow"	NOT SUPPORTED on WRITABLE points, can be written by Set Time actions on read only points.	Set Time
11.001_DATE	Example: "03-Oct-06"	NOT SUPPORTED on WRITABLE points, can be written by Set Date actions on read only points.	Set Date
UNKNOWN_4BYTE	<hex string>	Attempt to covert string to int using base 16, write not allowed if not convertible or out of range	—
12.001_UNSIGNED_COUNT	number as a string	Attempt to covert string to int using base 10, write not allowed if not convertible or out of range	—
13.001_SIGNED_COUNT	number as a string	Attempt to covert string to int using base 10, write not allowed if not convertible or out of range	—
14.000_ACCELERATION thru 14.079_WORK	number as a string	Attempt to covert string to float using base 10, write not allowed if not convertible or out of range	—
15.000_ACCESS_DATA	<hex string>	NOT SUPPORTED, write not allowed	—
16.000_STRING_ASCII	String	Write String, truncated at 14 characters	—
16.001_STRING_8859	String	Write String, truncated at 14 characters	—

Installation

To use the NiagaraAX EIBnet/IP driver, you must have a target host that is licensed for “eibnetIp”. In addition, other device limits or proxy point limits may exist in your license.

From your PC, use the Niagara Workbench 3.*n.nn* installed with the “installation tool” option (checkbox “This instance of Workbench will be used as an installation tool”). This option installs the needed distribution files (*.dist* files) for commissioning various models of remote JACE platforms. The dist files are located under your Niagara install folder in various revision-named subfolders under the “sw” folder.

When installing Workbench on your PC, you should also select the **eibnetIp** module.

Apart from installing the 3.*n.nn* version of the Niagara distribution files in the JACE, make sure to install the eibnetIp module too (if not already present, or upgrade if an older revision). For more details, see “About the Commissioning Wizard” in the *JACE NiagaraAX Install and Startup Guide*.

Following this, the JACE is now ready for EIBNET/IP software integration, as described in the rest of this document.

Quick Start

1. Create a new station
 - Follow the installation and configuration instructions preceding this.
 - On Workbench menu “Tools”, select “New Station”.
 - In the “Nav” pane of Workbench, expand the new station, and double-click the “drivers” branch. This will display a “Driver Manager” in the right pane.
 - Click “New” in the “Driver Manager”. Select the “Eibnet Ip Network”, and click “Add”. Answer the remain wizard prompts until the new network is added.
 - Right click on “config.bog” for the new station in the Nav tree, and select “Save”.
 - At this point you will need to download the new station to the JACE and start the station.
2. Download and Start the new station on the target JACE.
3. Configure and Learn
 - Open the running station in Workbench
 - View the property sheet for the EibnetIpNetwork
 - Expand the “LinkLayer” property and select the Ethernet adapter that is connected to the EIBnet/IP interface device.
 - Set “Inter Message Delay” property to 0.015 sec.
 - Save the station and restart the JACE.
 - Double click the EibnetIpNetwork in the Nav tree to display the Device Manager
 - Click Discover and wait for the discovery to complete.
 - Select a discovered device in the top pane, and click Add.
 - You now have an EibnetIpDevice
 - Navigate to the “Points” folder under the device and double click the points folder to display the EibnetIpPointManager view.
 - Click “Discover” button and navigate to an “efs” file that has been exported from ETS3. An “efs” file is a file with an “.efs” extension, and was produced by exporting ETS project (in ETS, do an “Export”/”Extract to OPC”).
 - Select and add EIBNET/IP points to your database.

EIBnet/IP Component Guides

EibnetIpNetwork

The EibnetIpNetwork provides all the configuration parameters necessary to allow the driver to communicate with a network of EIBNET/IP devices.

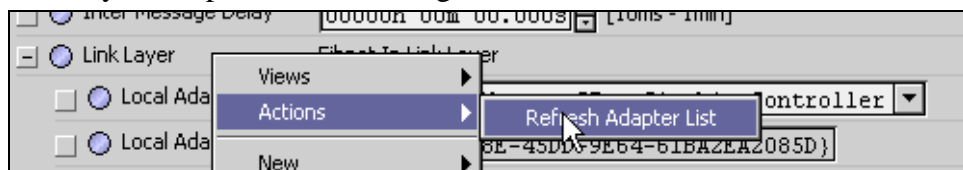
The EibnetIpNetwork is the "network-level" component in the NiagaraAX architecture. It has the standard network component properties such as status and enabled (see "Driver Architecture / Common network components" in the *NiagaraAX Drivers Guide* for more information), as well as properties unique to configuring an EibnetIp network.

EibnetIpNetwork (Eibnet Ip Network)	
<input type="checkbox"/> Status	{ok}
<input type="checkbox"/> Enabled	<input checked="" type="radio"/> true
<input type="checkbox"/> Fault Cause	
<input checked="" type="checkbox"/> Health	Ok [02-Oct-06 7:41 AM EDT]
<input checked="" type="checkbox"/> Alarm Source Info	Alarm Source Info
<input checked="" type="checkbox"/> Monitor	Ping Monitor
<input checked="" type="checkbox"/> Tuning Policies	Tuning Policy Map
<input checked="" type="checkbox"/> Poll Scheduler	Basic Poll Scheduler
<input type="checkbox"/> Inter Message Delay	00000h 00m 00.015s [10ms - 1min]
<input type="checkbox"/> Link Layer	Eibnet Ip Link Layer
<input type="checkbox"/> Local Adapter	3Com 3C920 Integrated Fast Ethernet Controller (3C905C-TX Compatible)
<input type="checkbox"/> Local Adapter Id	{2B8A11C1-88A3-40C0-B94A-A56ED1D39BB1}
<input type="checkbox"/> Local Ip Address	137.19.60.110
<input type="checkbox"/> Local Port Min	3500 [0 - max]
<input type="checkbox"/> Local Port Max	4000 [0 - max]
<input type="checkbox"/> Local Port	3500 [0 - max]
<input type="checkbox"/> Adapter Debug	<input checked="" type="radio"/> true
<input type="checkbox"/> Tunnel Layer	Eibnet Ip Tunnel Layer
<input checked="" type="checkbox"/> EibnetIpDevice	Eibnet Ip Device

NOTE: In the following properties, the grayed out properties are inherited from the base AX driver classes, and as such are only touched on here. For more details, refer to the "Common network components" section in the *NiagaraAX Drivers Guide* document.

- Status – The status of the network. Will normally be {ok}. See Fault Cause property for more information if not {ok}.
- Enabled – Enables or Disables the EIBnet/IP Driver
- Fault Cause – if the "Status" is fault, the fault cause will appear here..
- Monitor – container for monitor (ping) properties.
- Tuning Policies - A container for tuning policies which determines how and when proxy points are read and written.
- Poll Scheduler - The basic poll scheduler enables/disables polling, determines the fast/normal/slow poll rates, and maintains statistics about proxy extension polls.

- **Inter Message Delay** – The minimum amount of time to wait between receiving a message on the EIBnet/IP bus, and sending the next request. This gives time for some EIBnet/IP devices to prepare for receiving messages again. Note that this setting this value to non-zero has a negative impact on overall throughput, but may be necessary if a slow-to-turn-around EIBNET/IP device is on the network.
- **Link Layer** – A container for all properties that pertain to the link layer. Click on the ‘+’ to view the properties in this container:
 - **Local Adapter** – a pull-down selection of Ethernet adapters available on the platform. For an off-line station, this box will contain “none” as the only selection. The station must be running in order to obtain other selections. Select the adapter that is to be connected to the EIBnet/IP network.
 - **Local Adapter Id** – A read-only property that is updated whenever the Local Adapter property is changed. On Windows platforms, Local Adapter ID is a “GUID” value, and on QNX platforms, it is a path value.
 - **Local Ip Address** – A read-only property that is updated whenever the Local Adapter property is changed. This is the IP address of the selected adapter..
 - **Local Port Min** – The first port on which to attempt to open a socket for EIBnet/IP communications using UDP. The IP port number to use for EIBnet/IP communications on the local adapter. An attempt will be made to open a UDP socket using Local Port Min as the port number. If the port is not available, then additional attempts to open a UDP socket will be attempted at Local Port Min + 1, Local Port Min + 2, etc, until the Local Port Max is reached. The first available port in the range Local Port Min to Local Port Max will then be used for all Eibnet/IP UDP communications.
 - **Local Port Max** – The Maximum port on which to attempt to open a socket for EIBnet/IP communications using UDP. See also Local Port Min.
 - **Local Port** – The UDP port currently selected for EIBnet/IP communications.
 - **Adapter Debug** – If set to true, debug information on available adapters will be sent to stdOut during station start.
 - **Refresh Adapter List**¹ – An action accessible by right-clicking on the LinkLayer component in the navigation tree:



This action re-populates the adapter list by re-reading the adapters from the platform services of the station. This action needs to be invoked, for instance, if the station was engineered off-line, or was transferred from another platform, in order for the new platform’s adapters to be displayed. Remember to save the station after the list is updated.

- **Tunnel Layer** – Contains EIBnet/IP tunnel layer. Currently there are no user-viewable properties.

¹ Feature added starting with driver versions 3.1.31; 3.2.17; 3.3.1

EibnetIpDevice

The EibnetIpDevice provides all the configuration parameters necessary to allow the driver to communicate with a given EIBnet/IP device. The EibnetIpDevices are always children of an EibnetIpNetwork. EibnetIpDevices also serve as a container for all of the data points in an EIB network that need to be monitored or controlled.

The EibnetIpDevice is the "device-level" component in the NiagaraAX architecture.

EibnetIpDevice (Eibnet Ip Device)	
Status	{unackedAlarm}
Enabled	true
Fault Cause	
Health	Ok [02-Oct-06 8:18 AM EDT]
Alarm Source Info	Alarm Source Info
Ip Address	137.19.61.180
Mac Address	000e8c008296
Description	IP Interface N148 for Tridium
Device Info	Device Hardware Dtb
Services Info	V1 KNX Core V1 KNX Device Manager V1 KNX Tunneling
Data Endpoint	Eibnet Ip Hpai
Connection	Eibnet Ip Connection
Ets Export File	local: file:/C:/Program Files/Ets/Database/export.efs
Points	Eibnet Ip Point Device Ext

NOTE: In the following properties, the grayed out properties are inherited from the base AX driver classes, and as such are only touched on here. For more details, refer to the “Common device components” section in the *NiagaraAX Drivers Guide* document.

- Status – The status of the device. Will normally be {ok}. A value of {down} indicates that the last ping to the device was not answered (the “Report Exception Message” is used as the ping message).
- Enabled – Enables or Disables communication to the associated device from EIBnet/IP Driver
- Fault Cause – if the “Status” property value is {fault}, the fault cause is displayed here.
- Health – contains metadata about the health of this device on the network:
 - Down – indicates if this device is down – should be false under normal operation.
 - Alarm – indicates if this device is in alarm – should be false under normal operation
 - Last OK Time – the last time of successful communication to this device
 - Last Fail Time – the last time of unsuccessful communication to this device
 - Last Fail Cause – the reason of the last comm. Failure (example – “bad checksum)

- **Alarm Source Info** – configuration items for alarms generated from the ping process (device up and device down events)
- **IP Address** – the IP address of the target device. This address should be set to a static address in the IP interface using the ETS tool software.
- **Mac Address** – this is the MAC address of the IP interface. This value is read once at station start or during device learn.
- **Description** – this is the descriptive information about the IP device that is set into the device using the ETS tool software.
- **Device Info** – the entire device information block retrieved from the device at station start or device learn. As an example:

Device Info		Device Hardware DIB
<input type="checkbox"/>	KnX Medium	TP1
<input type="checkbox"/>	Status	OK
<input type="checkbox"/>	Individual Address	0.0.1
<input type="checkbox"/>	Project Number	0
<input type="checkbox"/>	Installation Number	0
<input type="checkbox"/>	Serial Number	000100200de8
<input type="checkbox"/>	Mac Address	000e8c008296
<input type="checkbox"/>	Friendly Name	IP Interface N148 for Tridium

- **Services Info** – The list of EIBnet/IP services supported by the device. This is read from the device once at station start or device learn.
- **Data Endpoint** – the IP address and port number that the device is currently using for UDP communications.
- **Connection** – the status and parameters of an established connection between the station and the EIBnet/IP device.

Connection		Eibnet Ip Connection
<input type="checkbox"/>	Channel Id	65 [0 - 255]
<input type="checkbox"/>	Channel Status	OK
<input type="checkbox"/>	Type Code	Tunnel
<input type="checkbox"/>	Individual Address	0.0.1

- **ETS Export File** – the last used path to a file used to “learn” points. This value is set by the point manager during the learn process.
- **Points** – a container for all data items (attributes) in this device which need to be polled for data.

KnxNumericProxyExt

KnxBooleanProxyExt

KnxEnumProxyExt

KnxStringProxyExt

All EIBnet/IP proxy extension types (KnxNumericProxyExt, KnxBooleanProxyExt, KnxEnumProxyExt, and KnxStringProxyExt) share the same set of configuration properties. Any instance of any of the EIBnet/IP proxy extension types is a proxy for one group address in an EIB network.

The EIBnet/IP proxy extension types take on the readable-writable personality of the control point they are attached to. For example, a KnxNumericProxyExt, when used as an extension on a NumericPoint has “read only” functionality, but when used on as an extension on a NumericWritable can read and write the attribute values.

The EIBnet/IP proxy extension types are the "point-level" component in the NiagaraAX architecture.

Proxy Ext		Knx Numeric Proxy Ext group:3/1/0	
<input type="checkbox"/>	Status	{stale}	
<input type="checkbox"/>	Fault Cause		
<input type="checkbox"/>	Enabled	<input checked="" type="radio"/> true	
<input type="checkbox"/>	Device Facets	>>	
<input type="checkbox"/>	Conversion	Default	
<input type="checkbox"/>	Tuning Policy Name	Default Policy	
<input type="checkbox"/>	Read Value	0.0 {ok}	
<input type="checkbox"/>	Write Value	0.0 {ok}	
<input type="checkbox"/>	Poll Enable	<input type="radio"/> false	
<input type="checkbox"/>	Poll Once On Subscribed	<input type="radio"/> false	
<input type="checkbox"/>	Poll Once On Operational	<input type="radio"/> false	
<input type="checkbox"/>	Poll Until Answer Received On Poll Once	<input type="radio"/> false	
<input type="checkbox"/>	Poll After Write	<input type="radio"/> false	
<input type="checkbox"/>	Poll Frequency	Normal	
<input type="checkbox"/>	Group Addresses	3/1/0	add delete edit
<input type="checkbox"/>	Data Type	2-byte	9.001 deg C
<input type="checkbox"/>	Hide Dynamic Actions	<input type="radio"/> false	

NOTE: In the following properties, the grayed out properties are inherited from the base AX driver classes, and as such are only touched on here. For more details, refer to the “ProxyExt properties” section in the *NiagaraAX Drivers Guide* document.

- Status – The status of the device. Will normally be {ok}. A value of {down} indicates that the last ping to the device was not answered (the “Report Exception Message” is used as the ping message).
- Fault Cause – if the “Status” property value is {fault}, the fault cause is displayed here.
- Enabled – Enables or Disables communication to the associated device from EIBnet/IP Driver
- Device Facets – contains metadata about the health of this device on the network:
 - Conversion
 - Tuning Policy Name
 - Read Value
 - Write Value
- **Poll Enable** – enable or disable on a proxy ext basis the generation of a “read” request to the main group address configured for this proxy extension. Note that in EIB, a group address will respond to a read request only if it is configured to do so, so for group addresses where there is no bus device configured to send the value in response to a read, it would make sense to not poll the group address. In other cases, it may simply be desired to not put additional traffic on the bus if not absolutely necessary.
- **Poll Once On Subscribed**² – Boolean, defaults to {false}. Used to force a poll whenever the point enters a subscribed state, such as when a user views it on a point list. If enabled, the resulting poll is independent, and will occur independent of any other poll setting (for instance, it will occur even if “Poll Enabled” is false, or “Poll Scheduler” rate is 0 or disabled.) Behavior can be modified by the "Poll Until Answer Received On Poll Once" property defined below.
- **Poll Once On Operational**² – Boolean, defaults to {false}. Used to force a poll whenever the point status changes from disabled-to-enabled, or down-to-up, or fault-to-noFault. If enabled, the resulting poll is independent, and will occur independent of any other poll setting (for instance, it will occur even if “Poll Enabled” is false, or “Poll Scheduler” rate is 0 or disabled. Behavior can be modified by the "Poll Until Answer Received On Poll Once" property defined below.
- **Poll Until Answer Received On Poll Once**² – Boolean, defaults to {false}. If “Poll Once On Subscribed” or “Poll Once On Operational” are set to true, then if this value is also set to true, the poll once behavior is modified to "poll until one valid value is received" instead of "poll once and forget". This has the effect of subscribing the point to the poll scheduler until such time as the point receives a LDataInd message addressed to the first group address in the list, and the value is a valid value for this point type. When these conditions are satisfied, the point is unregistered from the poll scheduler.
- **Poll After Write**² – Boolean, defaults to {false} Independent of “Poll Enable” property above. Used to enable or disable a poll for value after a write.

² Property added with driver versions 3.1.31; 3.2.17; 3.3.1

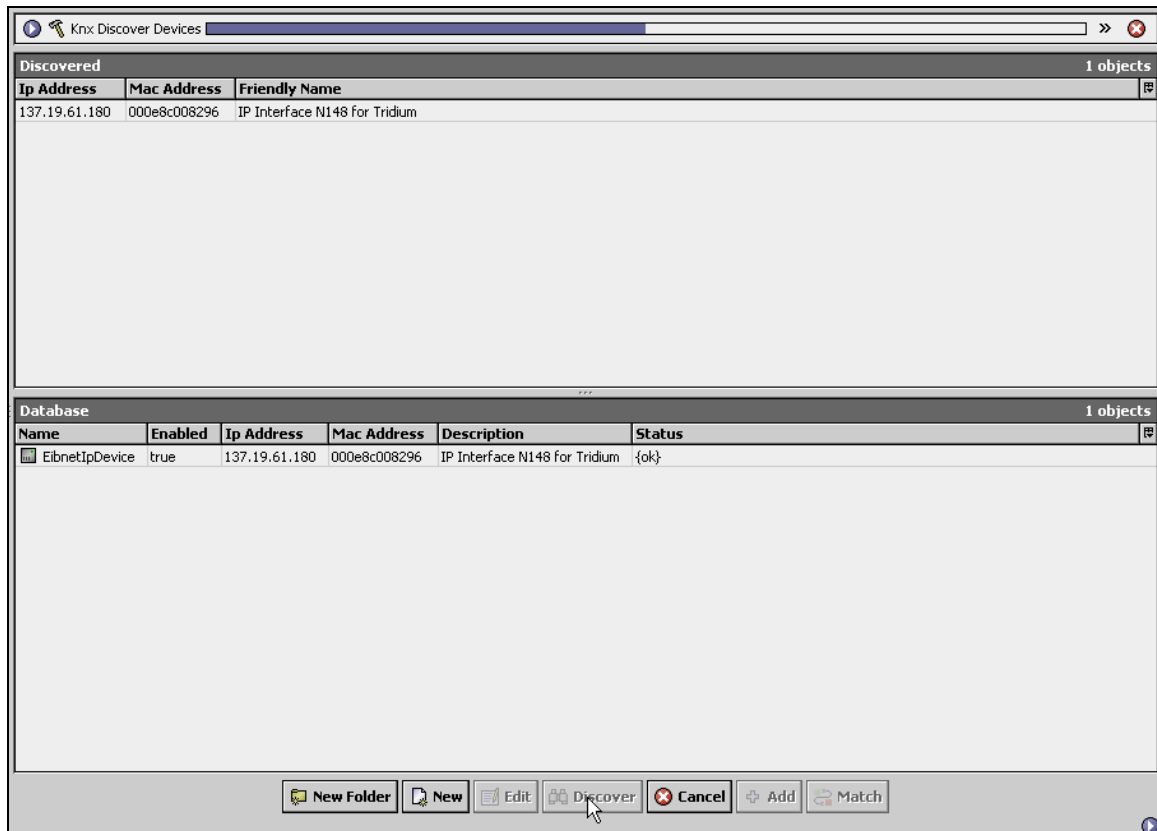
- **Poll Frequency** – Fast, Normal, or Slow. Rates are determined by the Poll Scheduler settings on the EibnetIpNetwork container.
- **Group Addresses** – A list of group addresses from which the value of this proxy extension can be updated. The first group address in the list is the primary group address and cannot be deleted, but can be edited. The primary group address is the address to which any read or write request will be directed from this proxy extension. The remaining group addresses are used to update the proxy extension output value whenever a message is received from any of these addresses.
- **Data Type** – the KNX data conversion type. Tells the driver how the data contained in an EIBnet/IP message is to be interpreted. To facilitate ease selection, the data types are organized by data size, from 1-bit to 14-byte in the left-hand pull down, with each appropriate data type that matches that data size contained in the right-hand pull down. This entry should match the data type configured for the point within the ETS tool software for the group address. In addition to the standard KNX data types, there are special “unknown” data types for each data size that behave slightly differently depending on what type of base point the proxy ext is on.
- **Hide Dynamic Actions** – Boolean, defaults to {false}. Provides a editable way to hide actions that are added dynamically to certain combinations of read-only point type and “Data Type”. See “[Actions May Be Added To Read-Only Points](#)”. If true, then the dynamic actions will be hidden, if false, then the dynamic actions will be visible.

EIBnet/IP Views

Device Manager

The Device Manager is the default view when you double-click on a EibnetIpNetwork in the Nav tree. This manager view provides a quick and easy way to display and learn EIBNET/IP interface devices that are on the IP network.

Below is an example Device Manager view for EibnetIp:



The EIBnet/IP Device Manager consists of either one or two main panes, depending on whether or not the “Discover” button has been clicked. The view above shows a typical EIBnet/IP Device Manager view.

The “New Folder”, “New”, and “Edit” buttons are not unique to the EIBnet/IP Device Manager, and are explained in the *NiagaraAX Drivers Guide* in the “About the Device Manager” section.

The “Discover” button is used to launch a device discovery process.

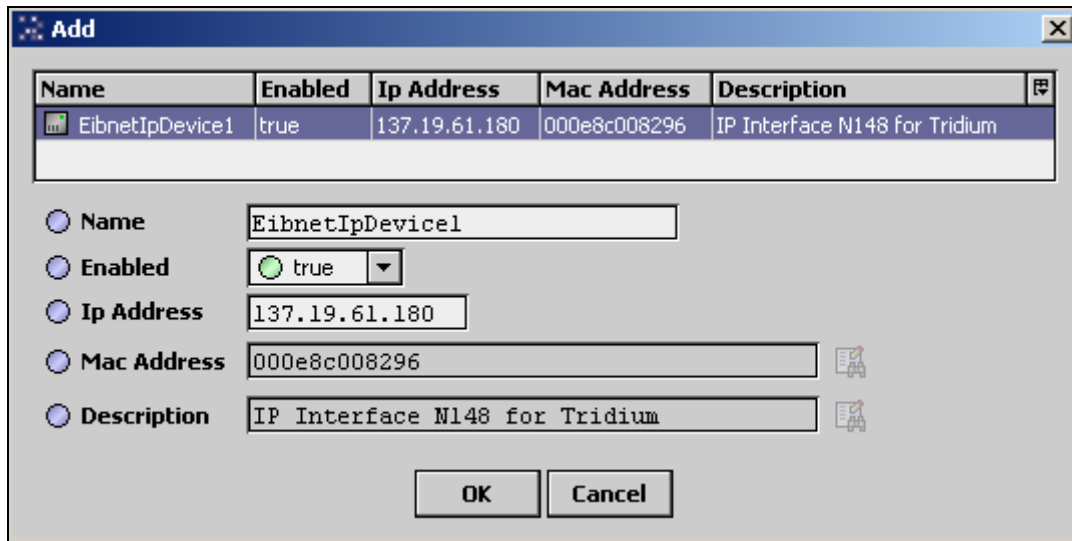
The “Match” button is used to match a device discovered in the top pane with a device that already exists in the database in the bottom pane. This is useful if the IP address is changed (either statically or via DHCP) in the IP device, or if the IP device is replaced.

The “Cancel” button will end a discovery that is in process.

The “Add” button allows manual addition of a device object into the database.

The “Discover” button does implement functionality that is unique and tailored to discovering EIBNET/IP devices. By clicking the “Discover” button, the “learn” mode of the manager is invoked (the panes will be split, and a “discovery” table will be displayed in the top pane) and a discovery process will be launched. Any discovered devices will be displayed in the top pane.

If you highlight one or more rows in the top “Discovered” pane, the “Add” button becomes active. You can now add the selected devices to the station database by clicking the “Add” button. This will pop up the “Add” dialog box:



Name	Enabled	Ip Address	Mac Address	Description
EibnetIpDevice1	true	137.19.61.180	000e8c008296	IP Interface N148 for Tridium

Name: EibnetIpDevice1

Enabled: true

Ip Address: 137.19.61.180

Mac Address: 000e8c008296

Description: IP Interface N148 for Tridium

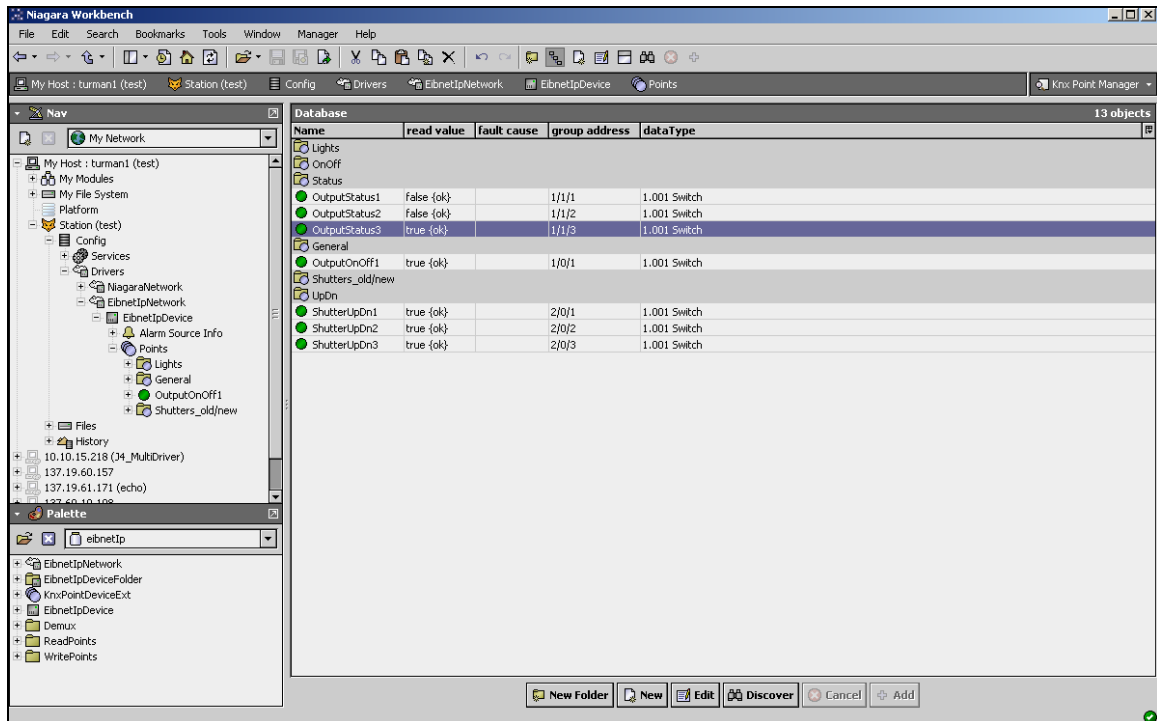
OK Cancel

The “Add” dialog box affords you the opportunity to tweak the display name, enabled state, and/or address of each of the selected devices. Click the “OK” button to add the devices to the database, or click “Cancel” to bail out.

Point Manager

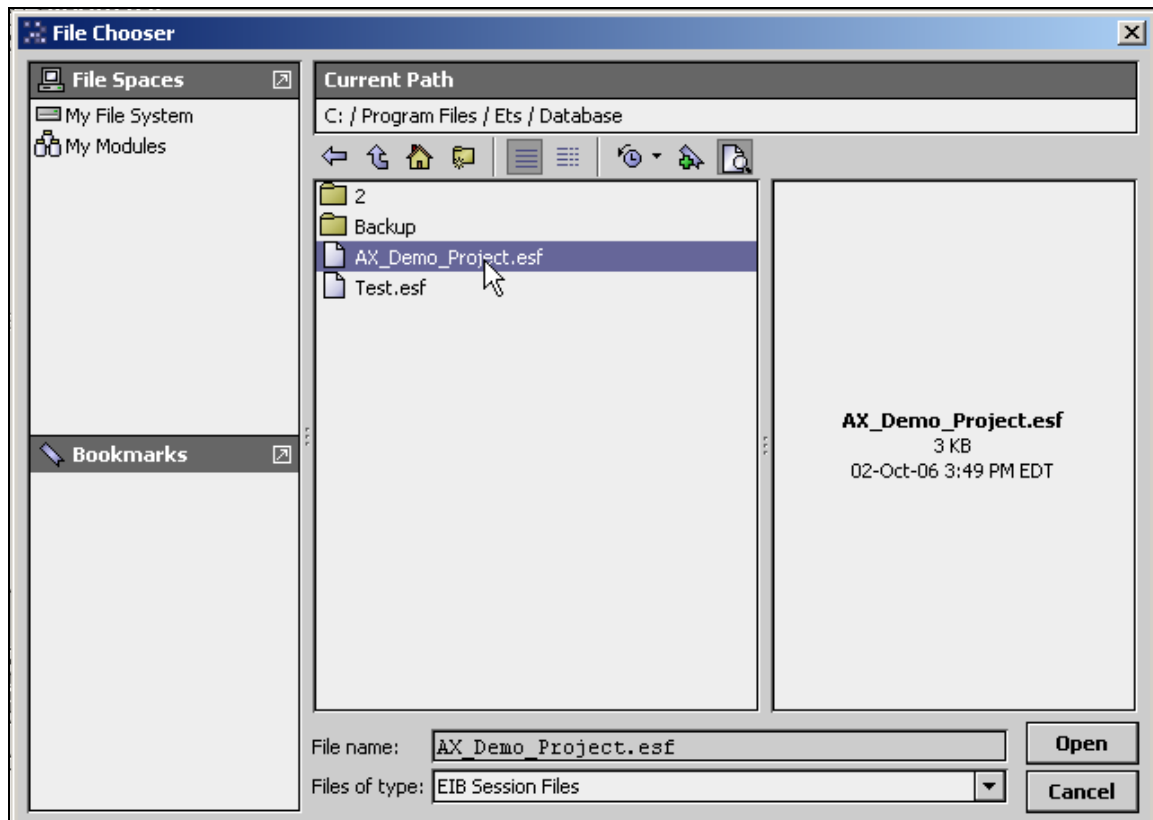
The Point Manager is the default view when you double-click on a “points” folder (a EibnetIpPointDeviceExt type folder) under a EibnetIpDevice in the Nav tree. This manager view provides a quick and easy way to display and learn KNX points from an export of an ETS3 project.

The EIBnet/IP Point Manager is the default view for any EibnetIpPointDeviceExt container. The Point Manager is a table-based view, where each row represents a unique group address within a EIB network. Below is an example Point Manager view:



The “New Folder”, “New”, and “Edit” buttons are not unique to the EibnetIp point Manager, and are explained in the *NiagaraAX Drivers Guide* in the “About the Point Manager” section. The “Match” button is not used for the EibnetIp driver.

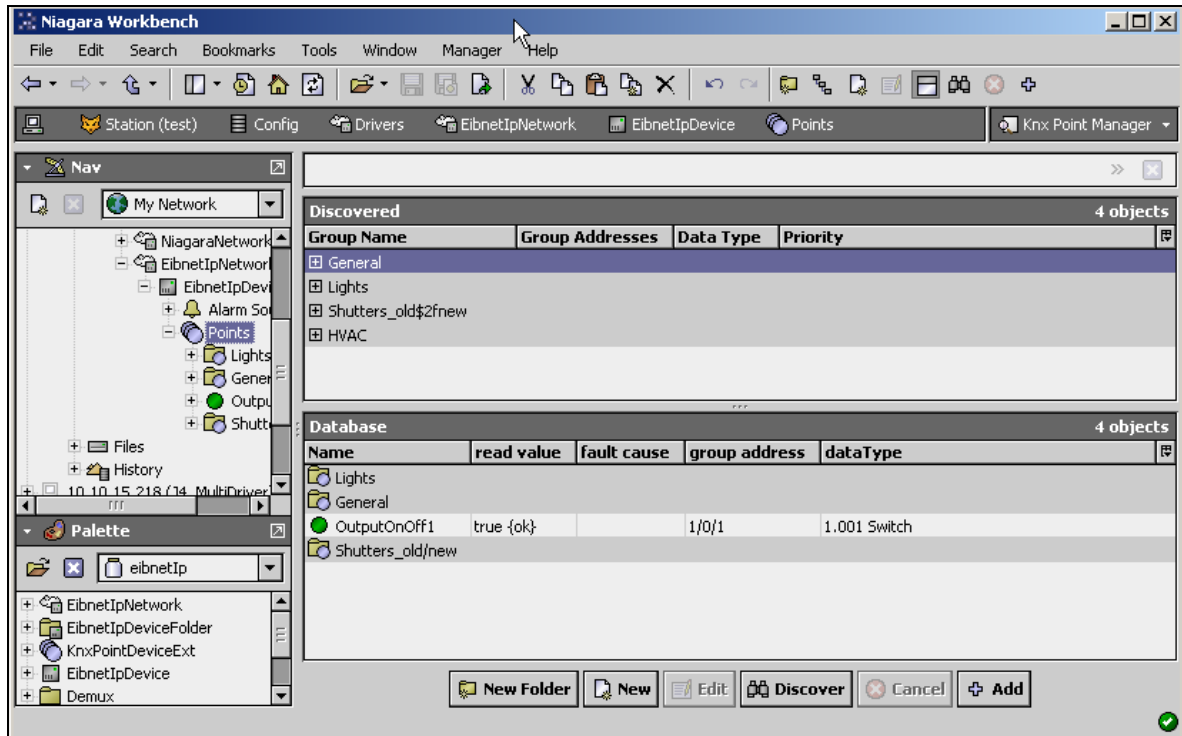
The “Discover” button implements functionality that is unique and tailored to discovering EIBnet/IP. By clicking the “Discover” button, the “learn” mode of the manager is invoked (the panes will be split, and a “discovery” table will be displayed in the top pane) and a dialog box will be popped up to prompt the user for some additional data to guide the discovery process, as shown here:



This dialog box contains a file chooser to select a file with an extension of “esf” (a EIB Session File). The path defaults to the last used file path. An “esf” file is an export file from ETS3 Software tool.

NOTE: To obtain an “esf” file for an ETS project, open the project in ETS first, and then select File/Extract from the pull down menus, then select “Export To OPC Server” from the resulting dialog box.

When you click “Open” from the File Chooser dialog box, the selected “esf” file will be opened and parsed, and the resulting information displayed in the form a “Discovered” pane on the top half of the point manager display:



Note that the first learn pane organizes the discovered points by main, middle, and subgroup names, with the subgroup name becoming the target point name (default). Each of the main and middle group names becomes a folder name. Each resulting folder that has a “+” mark in front of the name may be expanded to view contents by clicking on the “+” sign to expand. In the following example, there is a main group name “Lights” and a middle groups “OnOff”, “Status”, “Dimm”, “Brightness”, and “PriorityControl”. In addition, each of the middle groups in turn contains one or more learned points:

Group Name	Group Addresses	Data Type	Priority
General			
Lights			
OnOff			
OutputOnOff1	1/0/1	1.001 Switch	low
OutputOnOff2	1/0/2	1.001 Switch	low
OutputOnOff3	1/0/3	1.001 Switch	low
OutputOnOff4	1/0/4	1.001 Switch	low
OutputOnOff5	1/0/5	1.001 Switch	low
OutputOnOff6	1/0/6	1.001 Switch	low
ParcialOnOff2	1/0/8	1.001 Switch	low
ParcialOnOff1	1/0/9	1.001 Switch	low
GeneralOnOff	1/0/10	1.001 Switch	low
DimmerOnOff	1/0/20	1.001 Switch	low
Status			
Dimm			
Brightness			
PriorityControl			

The grayed lines in the learned points display can be thought of as folders, and each non-grayed out line is a learned point.

The display line for a learned point shows the point name (derived from the subgroup name in the efs file), the “primary” group address of the point, the point type as configured in ETS3, and the point priority. There is an additional field that is not viewable in the single line display format of the learned point display – if there are multiple group addresses associated with the point, you can view them by double clicking the point to bring up the “Add” dialog: (*note the multiple group addresses*)

Name	Type	Facets	group address	dataType	poll enable	poll once on su
OutputOnOff1	Boolean Point	trueText=true,falseText=false	1/0/1	1.001 Switch	false	false

Name: OutputOnOff1
 Type: Boolean Point
 Facets: trueText=true,falseText=false
 group address: 1/0/1, 1/0/9, 1/0/10
 dataType: 1-bit, 1.001 Switch
 poll enable: false
 poll once on subscribed: false
 poll once on operational: false
 poll until answer after poll once: false
 poll after write: false
 poll frequency: Normal
 Unique Actions Hidden?: true

add delete edit

OK Cancel

To add one or more points to the database, select them (multi-select by using Ctrl-click or Shift-click operations with the mouse), and then select add.

Single or multiple discovered points can be added as proxy points with EibnetIp proxy extensions by selecting the discovered row(s) in the top pane, and clicking add. Doing so will cause the “Add” dialog box to appear:

Name	Type	Facets	group address	dataType	poll enable	poll once on subscribed	poll or
OutputOnOff1	Boolean Point	trueText=true,falseText=false	1/0/1	1.001 Switch	false	false	false
OutputOnOff2	Boolean Point	trueText=true,falseText=false	1/0/2	1.001 Switch	false	false	false
OutputOnOff3	Boolean Point	trueText=true,falseText=false	1/0/3	1.001 Switch	false	false	false
OutputOnOff4	Boolean Point	trueText=true,falseText=false	1/0/4	1.001 Switch	false	false	false
OutputOnOff5	Boolean Point	trueText=true,falseText=false	1/0/5	1.001 Switch	false	false	false
OutputOnOff6	Boolean Point	trueText=true,falseText=false	1/0/6	1.001 Switch	false	false	false
PartialOnOff2	Boolean Point	trueText=true,falseText=false	1/0/8	1.001 Switch	false	false	false
PartialOnOff1	Boolean Point	trueText=true,falseText=false	1/0/9	1.001 Switch	false	false	false

Name: OutputOnOff1
 Type: Boolean Point
 Facets: trueText=true,falseText=false
 group address: 1/0/1, 1/0/9, 1/0/10
 dataType: 1-bit, 1.001 Switch
 poll enable: false
 poll once on subscribed: false
 poll once on operational: false
 poll until answer after poll once: false
 poll after write: false
 poll frequency: Normal
 Unique Actions Hidden?: true

OK Cancel

In this dialog, the points may be edited individually or in a batch. Note that any edits apply to ALL selected (highlighted) points in the Add dialog box. The “Type” determines which base point type will be created, and the “dataType” governs how the data contained in KNX data packets are to be interpreted. *There may be reasons for changing these at this point, but in general the most appropriate type of base point is already selected. You might want to verify that the Type does not need to be changed to a “writable” version of the same type.*

For a list of behaviors with various combination of “Type” and “dataType”, see the section [“Mapping of KNX Data Types to Proxy Extensions”](#).

Once the point(s) are satisfactorily edited, click “OK” to create the proxy points corresponding to the group addresses.

Special Considerations

The following sections explain special considerations:

- [Actions may be added to Read-Only points](#)
- [Other Special Circumstances](#)

Actions may be added to Read-Only points

For certain combinations of base point type (Numeric Point, Boolean Point, Enum Point, or String Point) and data type, dynamic action slots will be added to the point at startup, if the actions do not already exist. These actions can be hidden if desired by setting the “h” attribute in the config flags for the dynamic slot for the action, available through the slot sheet. See the tables in the section “[Mapping of KNX Data Types to Proxy Extensions](#)” for list of when actions are added.

The addition of the dynamic actions provides write functionality to the read-only point.

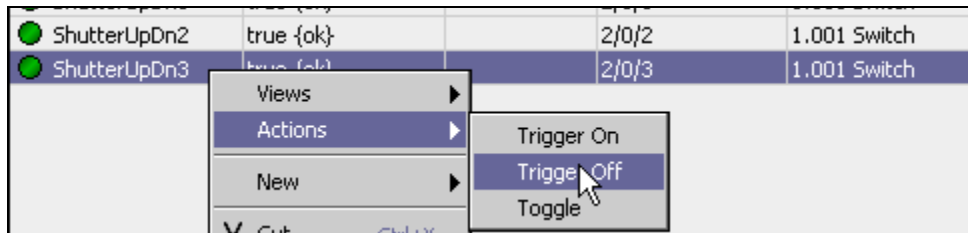
The text that appears for the command is customizable through the module lexicon (preferred), or directly through the display name for the slot on the slot sheet for the base point.

Switch Actions “Trigger On”, “Trigger Off”, and “Toggle”

These dynamic actions are added if both of the following is true:

- Data Type is a “1-bit” type, *including* “unknown_1bit”
- The Point Type is a BooleanPoint (not BooleanWritable)

Switch actions added to a Boolean Point



Trigger On: Causes a value of 1 to be written to the group address

Trigger Off: Causes a value of 0 to be written to the group address

Toggle: Causes a value opposite of the last known value to be written to the group address.

Lexicon entries may be used to modify the text that appears in the action menu:

```
triggerOnAction = Trigger On
triggerOffAction = Trigger Off
toggleAction = Toggle
```

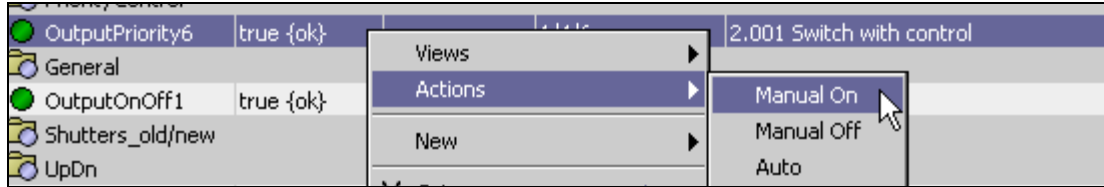
Any or all actions may be hidden or unhidden by modifying the “config flags” of the action slot on the slot sheet of the base point and selecting the “hidden” flag.

Switch Control Actions “Manual On”, “Manual Off”, and “Auto”

These dynamic actions are added if both of the following is true:

- Data Type is a “2-bit” type, *including* “unknown_2bit”
- The Point Type is a BooleanPoint (not a BooleanWritable)

Switch actions added to a Boolean Point



Manual On: Causes a manual override value of 1 to be written to the group address

Manual Off: Causes a manual override value of 0 to be written to the group address

Auto: Sends an auto command value.

Lexicon entries may be used to modify the text that appears in the action menu:

switchControlOnAction = Manual On

switchControlOffAction = Manual Off

switchControlAutoAction = Auto

Any or all actions may be hidden or unhidden by modifying the “config flags” of the action slot on the slot sheet of the base point and selecting the “hidden” flag.

Dimming Actions “Step Up”, “Step Down”, and “Break”

These dynamic actions are added if both of the following is true:

- Data Type is a “4-bit” type, *excluding* “unknown_4bit”
- The Point Type is a NumericPoint (not a NumericWritable point)

Dimming actions added to a String Point



Step Up: Causes a step up value to be written to the group address (note 1)

Step Down: Causes a step down value to group address (note 1)

Break: Sends a break command value.

Note 1: the step value is set in the dynamic property “stepSize” on the proxy extension.

Lexicon entries may be used to modify the text that appears in the action menu:

stepUpAction = Up

stepDownAction = Down

stepBreakAction = Break

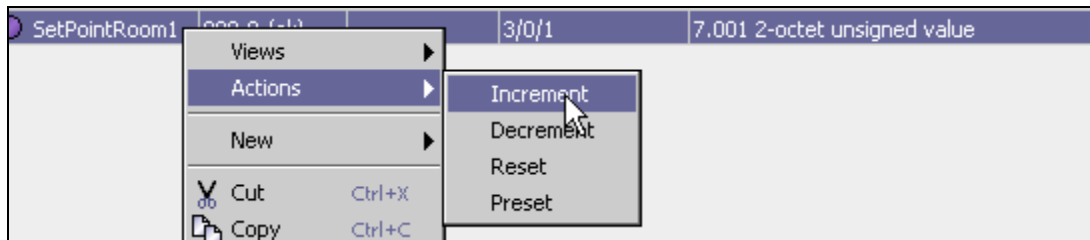
Any or all actions may be hidden or unhidden by modifying the “config flags” of the action slot on the slot sheet of the base point and selecting the “hidden” flag.

Counter Actions “Increment”, “Decrement”, “Preset”, and “Reset”

These dynamic actions are added if both of the following is true:

- Data Type is one of the following:
 - 5.010 8 bit unsigned value
 - 6.010 8 bit signed value
 - 7.001 2-octet unsigned value
 - 8.001 2-octet signed value
 - 12.001 4-octet unsigned value
 - 13.001 4-octet signed value
- The Point Type is a NumericPoint (is not a NumericWritable point)

Counter actions added to a Numeric Point



Increment: Prompts the user to input a value by which to increase the current value.

Decrement: Prompts the user to input a value by which to increase the current value.

Reset: Sends the value “0”

Preset: Prompts the user to input a value to send.

Lexicon entries may be used to modify the text that appears in the action menu:

```
incrementAction = Increment
decrementAction = Decrement
resetCountAction = Reset
presetCountAction = Preset
```

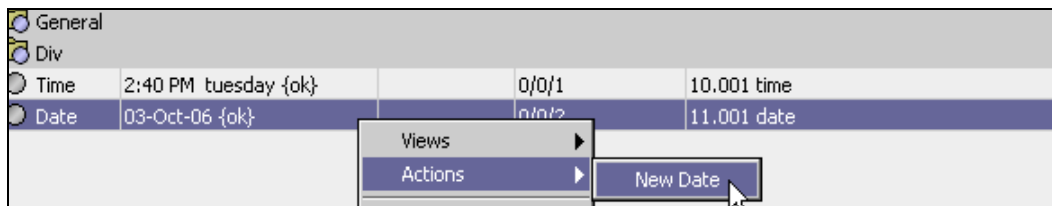
Any or all actions may be hidden or unhidden by modifying the “config flags” of the action slot on the slot sheet of the base point and selecting the “hidden” flag.

Date Action “New Date”

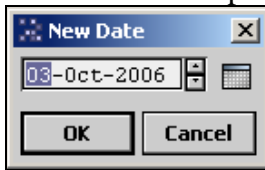
This dynamic action is added if both of the following is true:

- Data Type is “11.001 date”
- The Point Type is a StringPoint (is not a Writable point)

Date action added to a String Point



New Date: Prompts the user to input a date to send to the group address:



Lexicon entries may be used to modify the text that appears in the action menu:

modifyDateAction = New Date

The action may be hidden or unhidden by modifying the “config flags” of the action slot on the slot sheet of the base point and selecting the “hidden” flag.

Time Action “New Time”

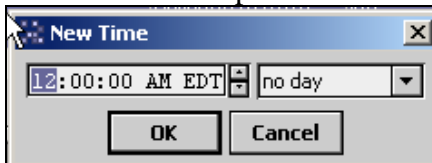
This dynamic action is added if both of the following is true:

- Data Type is “10.001 time”
- The Point Type is StringPoint (is not a Writable point)

Time action added to a String Point



New Time: Prompts the user to input a time to send to the group address



Lexicon entries may be used to modify the text that appears in the action menu:

modifyTimeAction = New Time

The action may be hidden or unhidden by modifying the “config flags” of the action slot on the slot sheet of the base point and selecting the “hidden” flag.

Other Special Circumstances

Data Type 16.000 String ASCII on a Boolean Point

Data Type 16.001 String 8859.1 on a Boolean Point

If a point is configured that consists of a Boolean Point with a KnxBooleanProxyExt , and the data type is 16.000 or 16.001, then there are two additional slots added to the proxy extension:

- Slot “falseTxt”: Contains a string that will be written if the value of the point transitions to false.
- Slot “trueTxt”: Contains a string that will be written if the value of the point transitions to true.

Document Change Log

- Updated: March 19, 2008
New EibnetIpNetwork, Link Layer action “Refresh Adapter List” described on page [22](#).
Additional ProxyExt properties “Poll Once On Subscribed,” “Poll Once On Operational,”
“Poll Until Answer Received On Poll Once,” and “Poll After Write” described on page [26](#).
Related to this, a few screenshots were also updated. Updated references to the *NiagaraAX Drivers Guide* instead of the *NiagaraAX User Guide*.
- October 10, 2006: Initial draft document.