Technical Document

Niagara KNXnet/IP Driver Guide

January 11, 2017

niagara4

Niagara KNXnet/IP Driver Guide

Tridium, Inc.
3951 Westerre Parkway, Suite 350
Richmond, Virginia 23233
U.S.A.

Confidentiality

Trademark notice

Copyright and patent notice

About this guide

This guide describes how to set up and use the Niagara KNXnet/IP driver.

Document change log

Updates (changes and additions) to this document are listed below.

- Updated: January 11, 2017
  Added more FAQ's and Property details
- Updated: January 9, 2017
  Added more FAQ's from Beta test feedback
- Updated: January 5, 2017
  Added more FAQ's from Beta test feedback
- Updated: September 28, 2016
  Added property details
- Updated: August 4, 2016
  Added property topics
- Updated: August 3, 2016
  Draft for Beta version x.x.x.x.x.30
- Updated: July 26, 2016
  Added Property reference chapter and draft property topics
- Updated: July 25, 2016
  Added comments from Bryce and typos corrected.
  Edited About KNX DataDefs,
  Edited Points in Fault — KNXnet/IP Interface FAQ,
  Added topics for Add Devices to the KNX Network
  Added topic for Add Points
- Updated: July 24, 2016
  Added About, Examining and Updating KNX Data Defs
  Added KNXnet/IP driver conflict FAQ
- Updated: July 16, 2016
  Updated with various entries
  - Issue to Bryce for comment
- Updated: June 06, 2016
  Admin
  - Renamed all topic <id> tor HTML output consistency
  - Chapters are now Chapter topics
  - Source BookMap progressed skeleton
- Initial release document: April 04, 2016

Related documentation

Several other documents are available for learning how to use the Niagara KNXnet/IP driver.

- *Niagara Drivers Guide* explains concepts.
- *NiagaraAX User Guide* explains concepts.

Contents

Chapter 1 Getting started

Topics covered in this chapter

♦ Quick start
♦ KNX requirements
♦ KNXnet/IP driver modules

The following topics describe how to get started:

Quick start

The basic steps to configure a Niagara station for communication with a KNX System involves setting up Workbench, commissioning the Supervisor or JACE platform and configuring input proxy points.

Step 1 Plan and then configure the KNX system using the ETS software tool.

Step 2 Using ETS, **Export** the ETS Project. This will provide the source of KNX system data for the Niagara KNXnet/IP driver.

Step 3 Copy the KNXnet/IP driver modules to the Niagara **Modules** folder in Workbench.

Step 4 Commission the Supervisor or JACE platform.

Step 5 Set up a KNX Network in the station.

Step 6 Set up one or more KNX Devices in the station.

Step 7 Set up one or more KNX Points in the station.

Step 8 Set up alarms and other components.

KNX requirements

Requirements include the version of Workbench supported and platform licensing requirements.

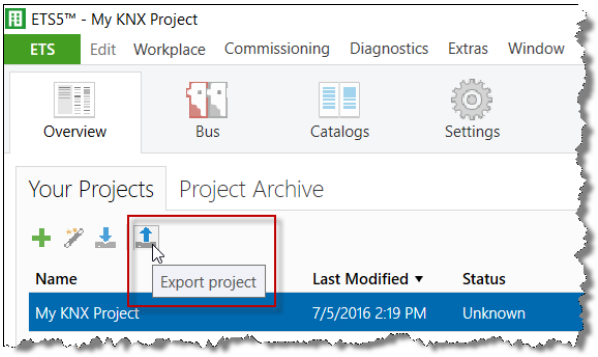
Systems Integrator requirements

The procedures in this document assume that you:

- Are Niagara certified and experienced at configuring stations.

KNX Project requirements

Using ETS, **Export** the ETS Project. This will provide the source of KNX system data for the Niagara KNXnet/IP driver.



Platform prerequisites

The KNXnet/IP driver requires a NiagaraAX or Niagara 4 platform that supports Hot Spot VM (virtual machine) from Oracle. It does not support the JACE 2, 4, or 5. The KNXnet/IP driver only supports an IP connection to the KNX network.

Version of Niagara

The KNXnet/IP driver is supported by the following versions of Niagara:

- NiagaraAX-3.8
- Niagara 4.1

Licensing requirements

- The **knxnetIp** feature must be present in both your Niagara Workbench platform **and** station platform licenses.
- Attributes associated with the **knxnetIp** feature are listed below:

Attribute	Description
device.limit	This attribute is common to most Niagara features and it defines the maximum number of devices that can be connected to this driver
history.limit	This attribute is common to most Niagara features and it defines the maximum number of histories that can be used for this feature
installation.limit	This attribute defines the maximum number of KNX Installation that can be connected to this driver
interface.limit	This attribute defines the maximum number of Niagara Platform interfaces that can be connected to this driver
point.limit	This attribute is common to most Niagara features and it defines the maximum number of points used on this feature
schedule.limit	This attribute is common to most Niagara features and it defines the maximum number of schedules used on this feature

KNXnet/IP driver modules

KNX integration requires Niagara modules for both NiagaraAX and Niagara 4 versions.

NiagaraAX KNX modules

Two modules are required for the KNXnet/IP driver:

- knxnetIp.jar
- docknxnetIp.jar

Niagara 4 KNX modules

Three modules are required for the KNXnet/IP driver:

- knxnetIp-rt.jar
- knxnetIp-doc.jar
- knxnetIp-wb.jar

Install modules in Workbench

The latest KNX .jar files may or may not be present in the Workbench **Modules** folder. It is important to work with the latest modules.

Prerequisites:

- A version of Workbench that supports the KNXnet/IP driver must be installed on the PC or laptop computer. For KNXnet/IP driver requirements, see KNX requirements, page 7.
- Access to Niagara Central to retrieve the modules .jar files if you do not already have the latest modules.

Use this procedure to manage the KNXnet/IP driver modules.

Step 1 Check the Niagara*version*\modules folder (where *version* is the version of Niagara you are using, for example `Niagara\Niagara-3.8.41\modules`).

Step 2 If needed, download the latest module .jar files from Niagara Central and save them in the Niagara *version*\modules folder.

Chapter 2 Setting up the station

Topics covered in this chapter

♦ Add a KNX Network
♦ Load the KNX Data Defs
♦ Configure connection to KNXnet/IP network
♦ Add Devices to the KNX Network
♦ Add Points from an ETS Project file

The following topics describe how to set up the station:

Add a KNX Network

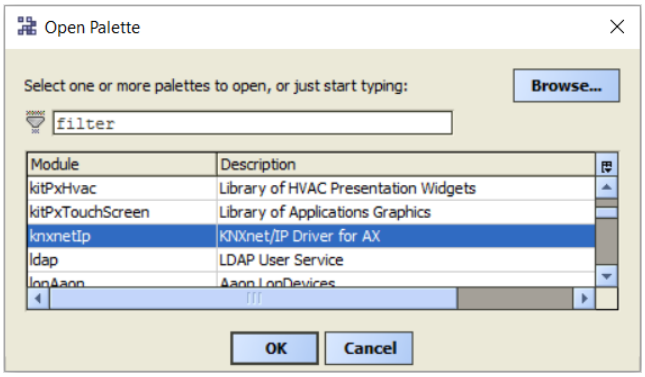
Adding the network is the first step to configure the station.

Prerequisites:

- The KNXnet/IP driver modules must be available in the Niagara*version*\modules folder (where *version* is the version of Workbench).

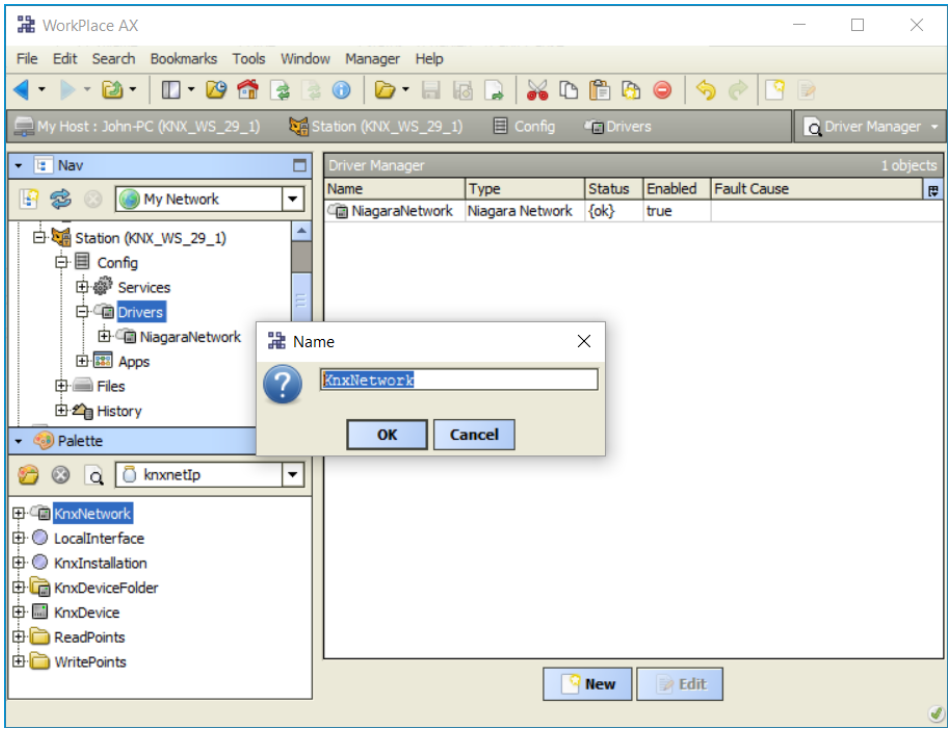
Step 1 In the Nav tree, expand the station and double-click the **Drivers** node.

Step 2 In the **Driver Manager** view, open the KNXnet/IP driver palette in the side bar.



Step 3 Select the **KNX network driver Module** and click **OK**.

Step 4 Drag and drop or copy the **KNX network driver** component from the palette to the **Driver Manager** view pane.



Step 5 Name the network and click **OK**.

Step 6 The newly added **KNX network driver** will initially appear with a **Status** of `{down,alarm,unackedAlarm}`, because its **KNX Data Defs** have not yet been loaded.

You are now ready to load the **KNX Data Defs** and configure the **KNX network driver**.

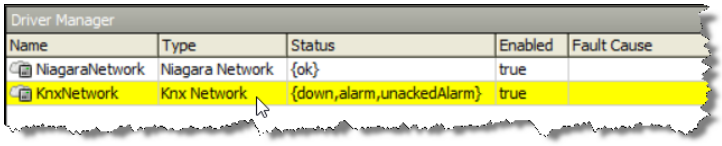
Load the KNX Data Defs

Loading the KNX Data Defs is a pre-requisite to setting up the network. See About KNX Data Defs, page 24.

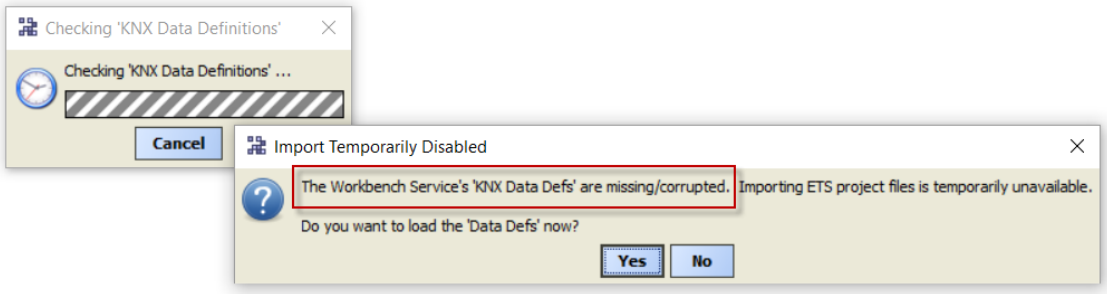
Prerequisites:

- The **KNX network driver** must have been be added to the **Drivers** node in the station.

Step 1 In the **Driver Manager** view, double click on the **KNX Network driver** and open its **Device Manager** view.

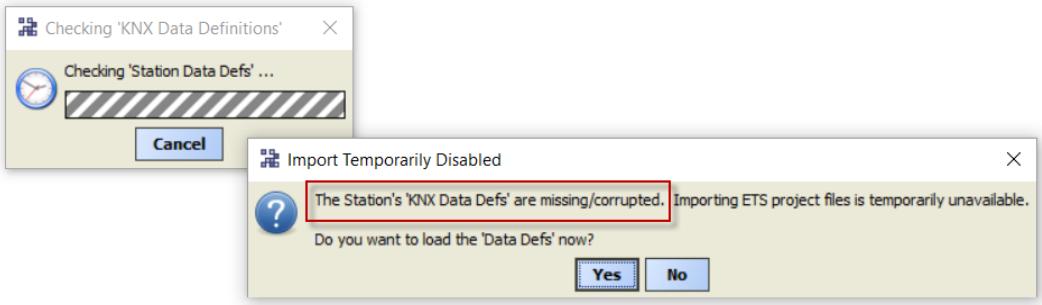


Step 2 If this is the first time your Workbench has been used with a **KNX Network**, then the **Import Temporarily Disabled** dialogue will appear.



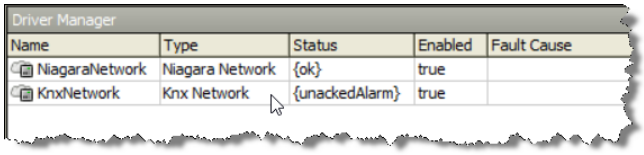
Step 3 Select the **Import Temporarily Disabled** dialogue and click **Yes** to load the Workbench Service's **KNX Data Defs**.

Step 4 The **Import Temporarily Disabled** dialogue will appear, with reference to the station's **KNX Data**.



Step 5 Select the **Import Temporarily Disabled** dialogue and click **Yes** to load the station's **KNX Data Defs**.

Step 6 In the **Driver Manager** view, the `{down}` status of the **KNX Network** has cleared.



You are now ready to configure the **KNX network driver's** connection to the physical **KNXnet/IP network**.

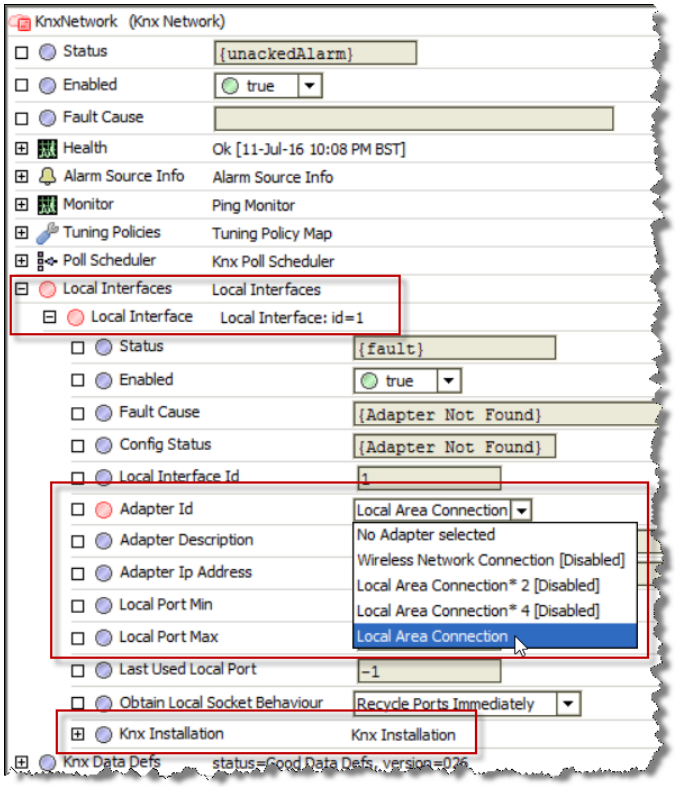
Configure connection to KNXnet/IP network

The **KNX network driver's** connection to the **physical KNXnet/IP network** can now be configured.

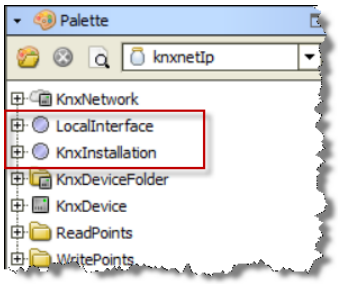
Prerequisites:

- The **KNX network driver** must have been be added to the **Drivers** node and the **Data Defs** must be loaded in Workbench and the station.

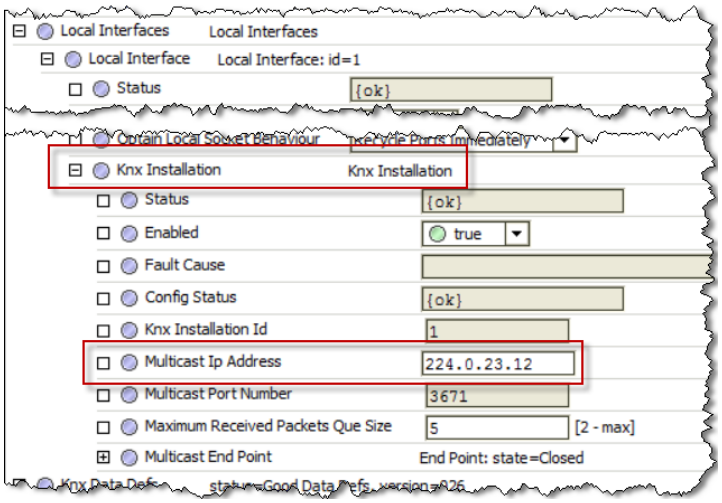
Step 1 In the **Property Sheet** view of the **KNX Network driver** a default **Local Interface** has been added to the **KNX Network driver** and a default **KNX Installation** has been added to the **Local Interface**. Select the **Adapter Id** that you wish to use for this connection to the **physical KNXnet/IP network**.



*NOTE: Additional **Local Interfaces** can be added from the palette for each of the platform's TCP/IP Interfaces you wish to use for connection to a **physical KNXnet/IP network**. All the **Local Interfaces** in the station must have different **Adapter Id's**.*

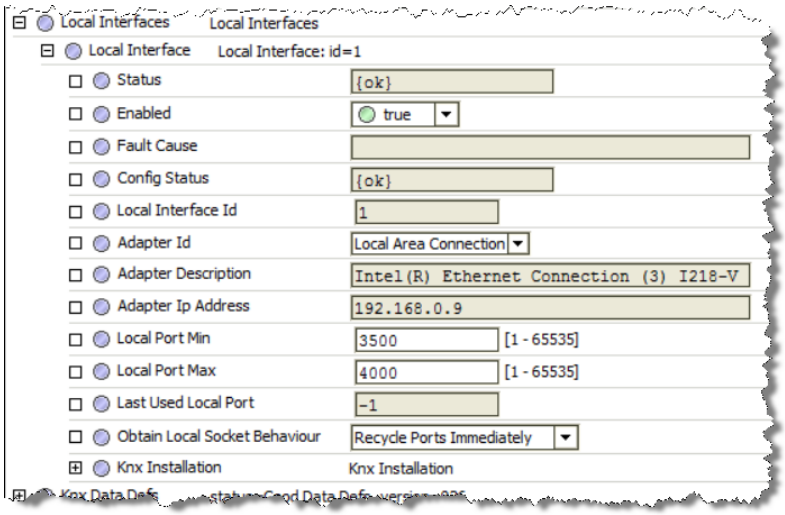


Step 2 If you *are not* using the **Default IP Multicast Address** of `224.0.23.12`, then you should enter it in the **KNX Installation** property.



*NOTE: If you are using more than one **KNX IP Multicast Address**, you can add additional **KNX Installation** instances from the palette to any **Local Interface**. All the **KNX Installation** instances under a **Local Interface** must have different **KNX IP Multicast Addresses**.*

Step 3 Click **Save** to save your settings.



You are now ready to add **KNXnet/IP Interface devices** to the **KNX network driver**.

Add Devices to the KNX Network

Niagara KNX Device instances can be added to the station in three ways:

- **Discovered** on a connected physical KNXnet/IP network
- **Imported** from an ETS project file (`*.knxproj`)
- **Manual** input

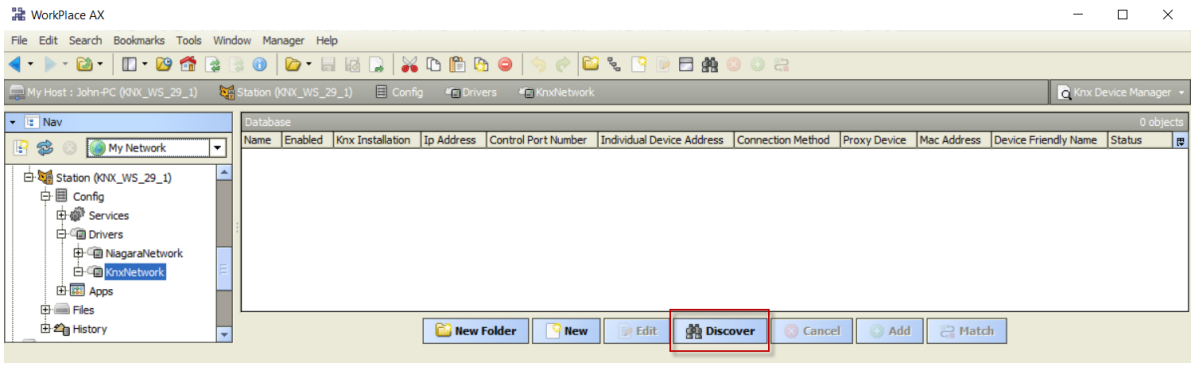
Discover Devices on the KNXnet/IP network

Niagara KNX Device instances can be added to the station by **Discovery** on a connected physical KNXnet/IP network as follows:

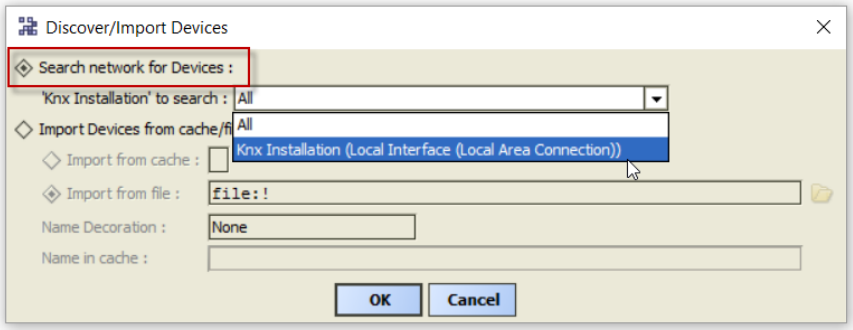
Prerequisites:

- The **KNX network driver** must have been be added to the **Drivers** node, the **Data Defs** must be loaded and connection to the **physical KNXnet/IP network** configured.

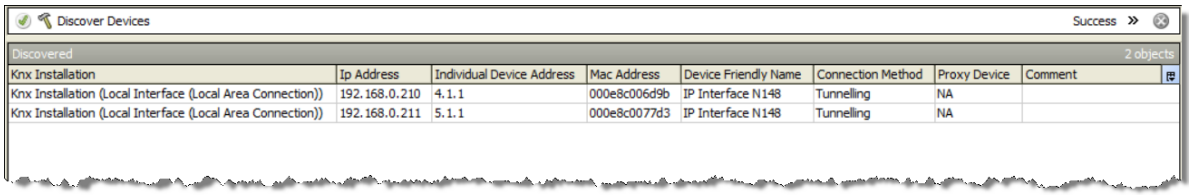
Step 1 In the **Driver Manager** view, double click on the **KNX Network driver** to open its **Device Manager** view.



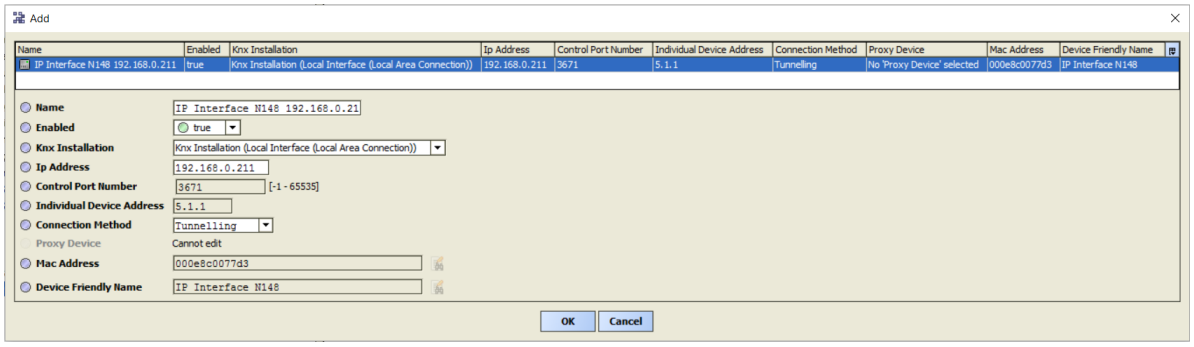
Step 2 Click **Discover** to display the **Discover/Import Devices dialogue**.



Step 3 Select the **Search network for Devices** option, choose which **Knx Installation** to search and click **OK**. The Discovery process will take about 10 seconds for each **Knx Installation** searched.



Step 4 Drag Device(s) from the **Discovered** pane into the **Database** pane or click **Add** to display the **Add dialogue**.



*NOTE: When Devices are added by this method, the **Knx Installation** and **Ip Address** properties will have been recorded during the discovery process and require no further configuration.*

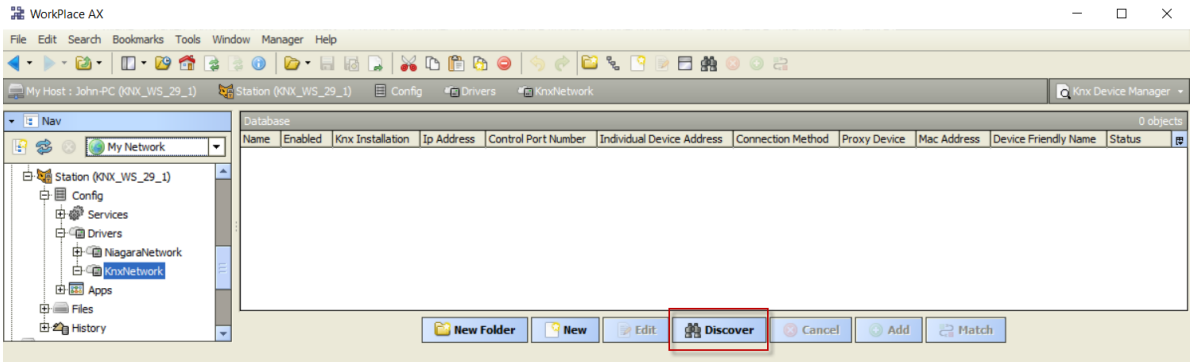
Step 5 Click **OK** in the **Add dialogue**.

Import Devices from an ETS Project file

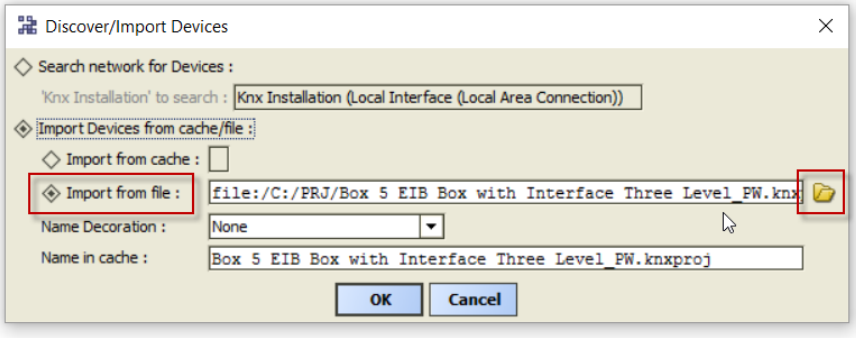
Prerequisites:

- The **KNX network driver** must have been be added to the **Drivers** node, the **Data Defs** must be loaded and connection to the **physical KNXnet/IP network** configured.

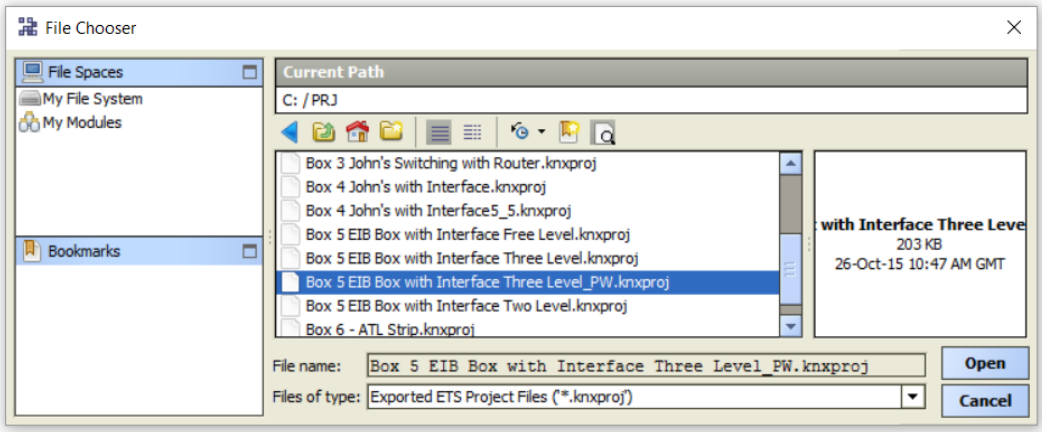
Step 1 In the **Driver Manager** view, double click on the **KNX Network driver** to open its **Device Manager** view.



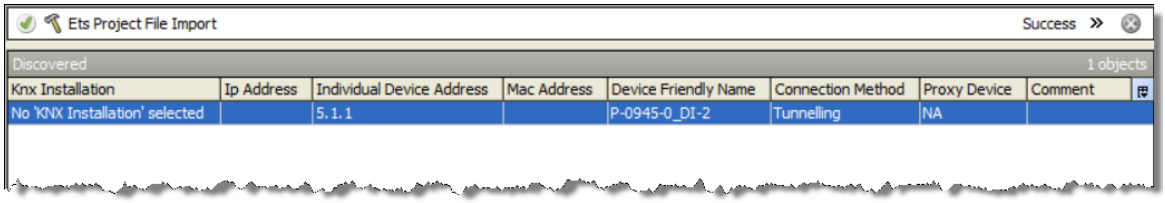
Step 2 Click **Discover** to display the **Discover/Import Devices dialogue**.



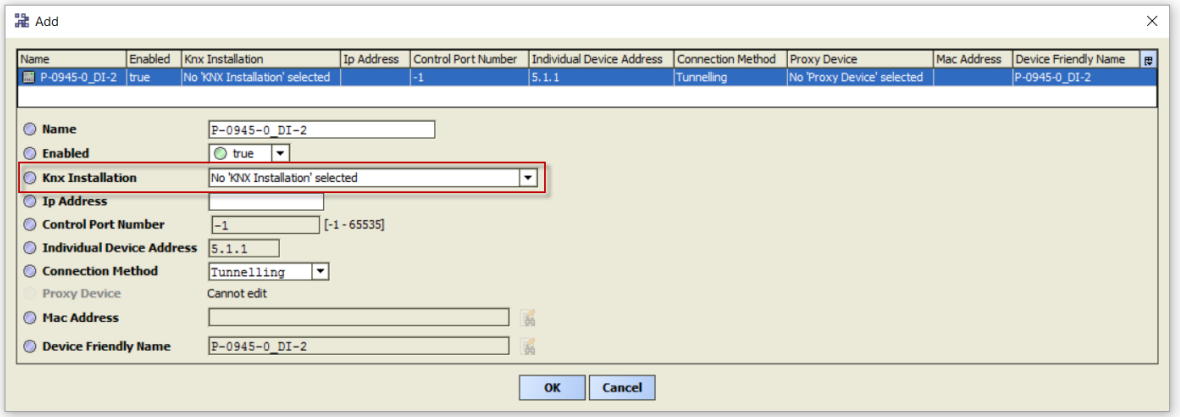
Step 3 Select the **Import Devices from cache/file** and the **Import from file** options and click on the File Chooser to display the **File Chooser dialogue**.



Step 4 Navigate to and select your **ETS Project file**, click **Open** and then click **OK** in the **Discover/Import Devices dialogue**. The time taken to import an **ETS Project file** is proportional to its size.



Step 5 Drag Device(s) from the **Discovered** pane into the **Database** pane or click **Add** to display the **Add dialogue**.



*NOTE: When Devices are added by this method, only the **Individual Device Address** property can be reliably derived from the **ETS Project file**. You will need to configure the **Knx Installation** property of each Device. It is **not necessary** to configure the **Ip Address** property because this will be discovered from the Device itself when the **KNX network driver** first tries to connect to the Device, by using its **Individual Device Address**.*

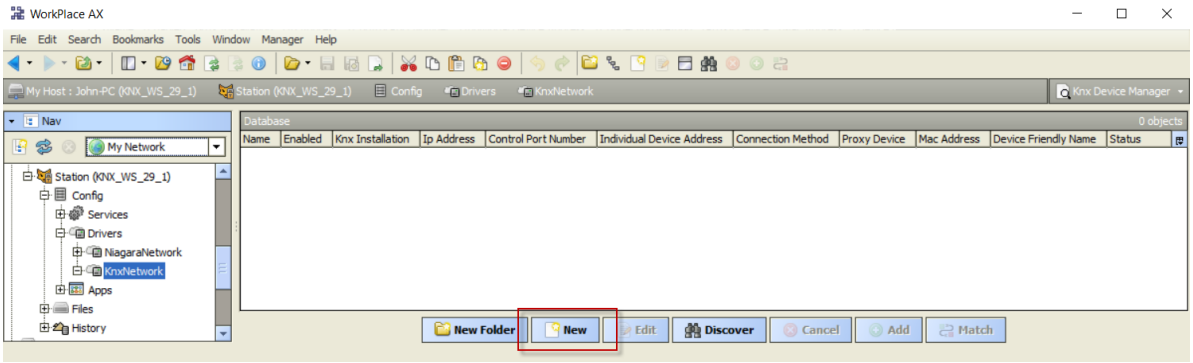
Step 6 Click **OK** in the **Add dialogue**.

Input Devices Manually

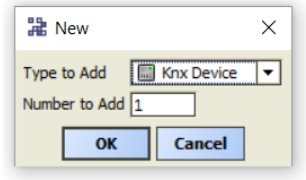
Prerequisites:

- The **KNX network driver** must have been be added to the **Drivers** node, the **Data Defs** must be loaded and connection to the **physical KNXnet/IP network** configured.

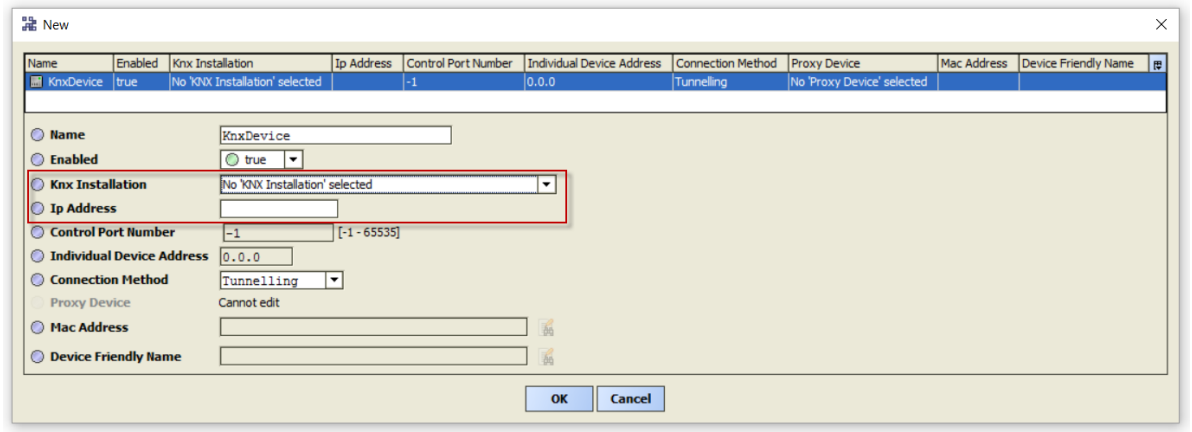
Step 1 In the **Driver Manager** view, double click on the **KNX Network driver** to open its **Device Manager** view.



Step 2 Click **New** to display the **New [Devices] dialogue**.



Step 3 Enter the **Number [of Devices] to Add** and click **OK** to display the **New dialogue**.



*NOTE: When Devices are added by this method, you will need to configure **both** the **Knx Installation** and **Ip Address** properties of each Device. It is **not necessary** to configure the **Individual Device Address** property because this will be discovered from the Device itself when the **KNX network driver** first tries to connect to the Device, by using the **Ip Address**.*

Step 4 Click **OK** in the **New dialogue**.

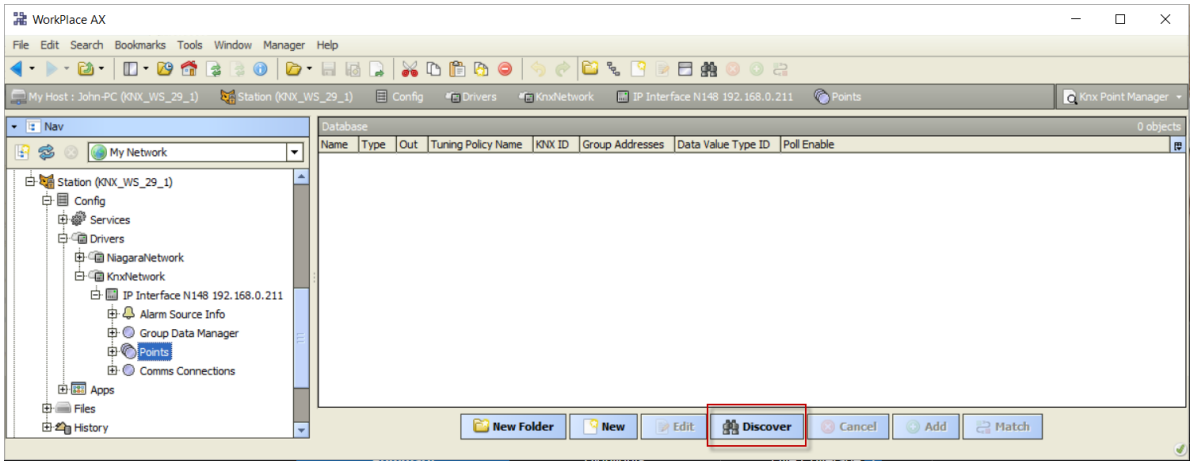
Add Points from an ETS Project file

The only method to **Add** KNX points through **Discovery** is by using the ETS Project file:

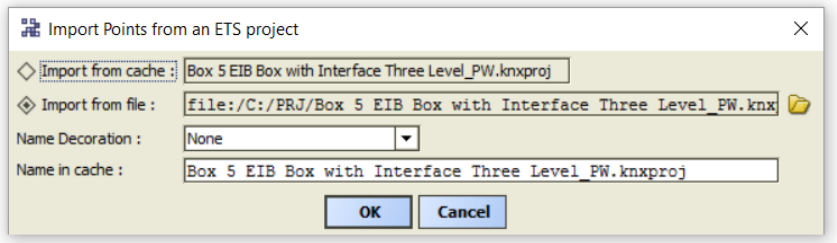
Prerequisites:

- One or more **KNX Device instances** must have been be added and configured to the **KNXnet/IP driver**.

Step 1 Double click on the **Points** node to open its **Point Manager** view.

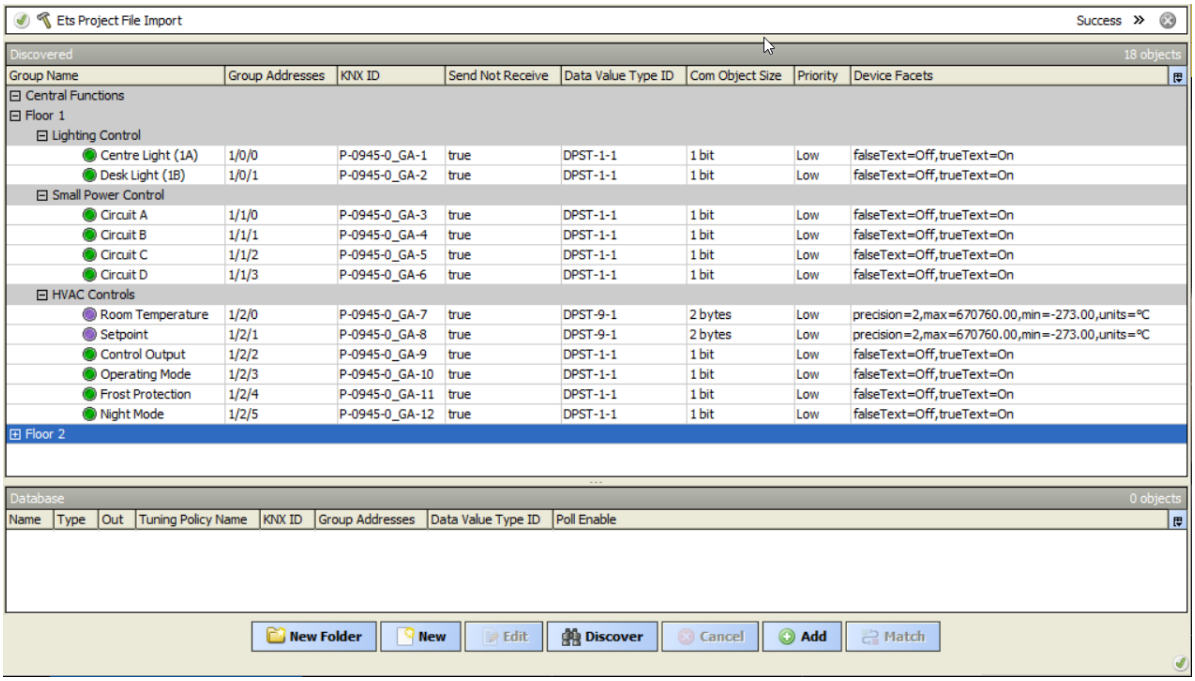


Step 2 Click **Discover** to display the **Import Points from an ETS project dialogue**.

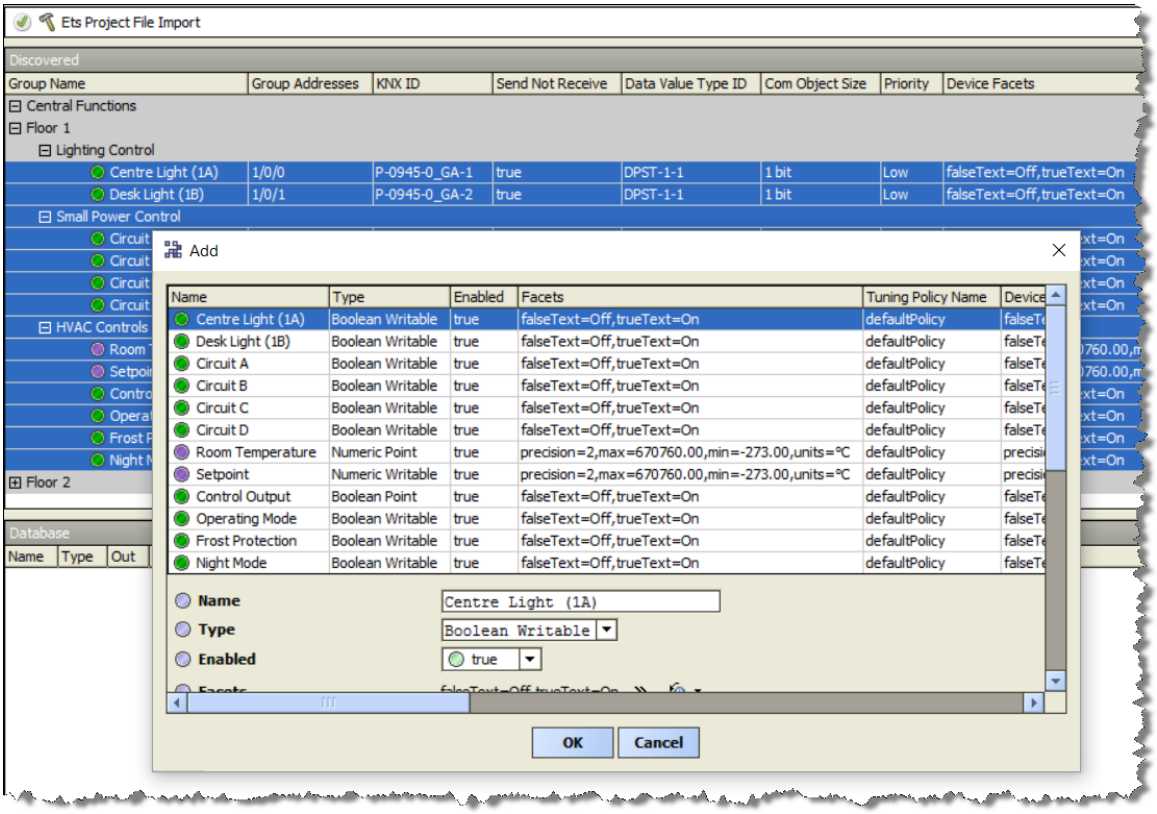


*NOTE: The cache retains all the previously loaded ETS project files and you can choose to either use one of these or import a new one. When you choose to **Import from cache** a check is performed to detect changes to the original file since it was last imported into the cache.*

Step 3 Select the **Import from cache** or **Import from file** option and click **OK**.



Step 4 Drag Point(s) from the **Discovered** pane into the **Database** pane or click **Add** to display the **Add dialogue**.



Step 5 Click **OK** in the **Add dialogue** to add points into the Database.

Chapter 3 Driver reference

Topics covered in this chapter

♦ About the KNXnet/IP driver
♦ Compatibility
♦ KNXnet/IP driver summary
♦ About KNX Data Defs

About the KNXnet/IP driver

The Niagara KNXnet/IP driver integrates a KNX system into a Niagara station.

The KNXnet/IP driver uses a familiar Niagara driver hierarchy of an upper-tier parent `network` component and one or more child `device` components, each with `device ext` (extension) child components.

Compatibility

KNXnet/IP Services compatibility

The following list details the KNXnet/IP Services support implemented in this KNXnet/IP driver:

KNXnet/IP Service	Support
Core Services	Fully Supported
Device Management	Not Supported (client only)
Tunnelling	Data Link Layer — Fully Supported CEMI Raw— Not Supported KNX Busmonitor— Not Supported
Routing	Not Supported
Remote Logging	Not Supported
Remote Configuration and Diagnostics	Not Supported
Object Server	Not Supported

KNX System Specification editions

Here are the specific KNX System Specification document editions:

- 03_01_02 Glossary v01.04.00 AS.pdf
- 03_02_06 Communication Medium KNX IP v01.00.02 AS.pdf
- 03_03_02 Data Link Layer General v01.02.02 AS.pdf
- 03_03_07 Application Layer v01.06.04 AS.pdf
- 03_06_03 EMI_IMI v01.03.03 AS.pdf
- 03_07_02 Datapoint Types v01.08.03 AS.pdf
- 03_07_03 Standardized Identifier Tables v01.03.01 AS.pdf
- 03_08_01 Overview v01.04.02 AS.pdf

- 03_08_02 Core v01.05.02 AS.pdf
- 03_08_03 Management v01.06.02 AS.pdf
- 03_08_04 Tunnelling v01.05.04 AS.pdf

KNXnet/IP driver summary

Here are some highlights of the Niagara KNXnet/IP driver:

- New driver called KNXnet/IP which has no references to the term EIB
- New driver license `feature` – **knxnetIp**
- KNXnet/IP driver in NiagaraAX-3.8 and Niagara 4.1 versions.
- JACE-3E, JACE-6 and JACE-6E implementation (Not applicable in JACE-2)
- Supports both ETS4 and ETS5 KNX tools
- Uses the more data-rich ETS `Project` file as a source of KNX system data for Niagara Discovery
- Supports ETS Two Level, Three Level and Free Level group addressing
- Supports the full list of current KNX `Datapoint Types` defined in the KNX `Master Data xml` file
- Supports Manufacturer Specific KNX `Datapoint Types` with values exposed as `hex` strings
- Ready for **KNX IP Secure** and **KNX Data Secure** (Secure Application Layer)
- Enhanced **Point Discovery** to improve pre-setting `Point Facets`, `Point Names`, `Hierarchy` and `Point types`
- Supports complex multi-value KNX `Datapoint Types`
- Consistent Niagara Framework® modelling (**KNXnet/IP-device** = Niagara—Device and **KNX Group Address** = Niagara—Point)
- Supports **KNXnet/IP Tunnelling**
- Supports **Proxy Routing**
- Enhanced **Bus Data Received** functionality
- Supporting `doc_User Guide` and Workbench `HTML` Help

About KNX Data Defs

In order for the KNXnet/IP driver to translate **Group Address** data from the encoded native KNX (on the wire) communications it receives from the KNX devices to the human friendly format displayed in the properties of Niagara Control Points, and vice-versa, it needs some information about the standard **KNX Datapoint Types**, as published in the KNX specification document `03_07_02 Datapoint Types v01.08.03 AS.pdf` (or later version).

This information is collectively referred to as the **KNX Data Definitions** or **KNX Data Defs** in the KNXnet/IP driver, and it is stored in a file called `knx_extra.xml`. In addition there is information about the units, minimum & maximum values and compatible Niagara Control Point types, for each of the **KNX Datapoint Types**. The Niagara Workbench contains a copy of the **KNX Data Defs**. Each Niagara station also has its own copy of **KNX Data Defs**. To examine the **KNX Data Defs**, see Examining the KNX Data Defs, page 25.

The `knx_extra.xml` file works along similar lines to KNX's `knx_master.xml` file, which is part of the ETS installation and is updated from time to time with new **KNX Datapoint Types** by the KNX organization. To update the **KNX Data Defs**, see Updating the KNX Data Defs, page 25.

*NOTE: The `knx_extra.xml` file is fundamental to the correct operation of the KNXnet/IP driver and it is internally **signed** to avoid the possibility of introducing erroneous operation of the KNXnet/IP driver.*

Each release of the KNXnet/IP driver contains a version of `knx_extra.xml` that is current at the time of release, however newer versions of the `knx_extra.xml` file will be published to keep pace with changes to the KNX `knx_master.xml`.

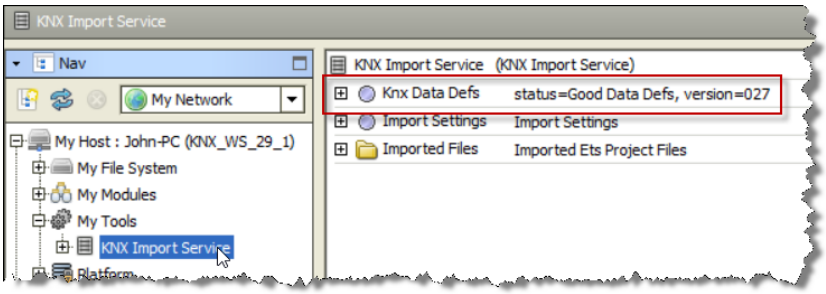
*NOTE: There is no correlation between the version number of the **KNX Data Defs** (for example `027`) and the version number of the KNXnet/IP driver module (for example `3.8.41.1.0.29`).*

Examining the KNX Data Defs

To examine the current status and version of the **KNX Data Defs**:

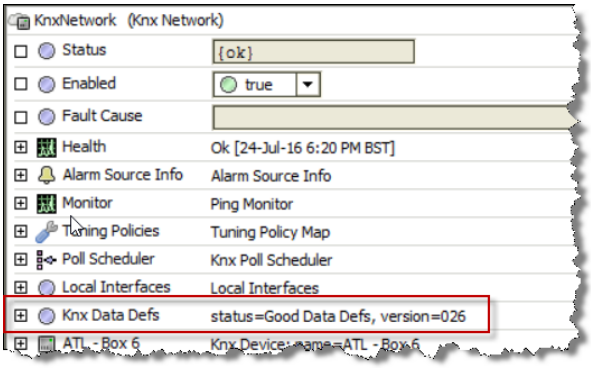
In the Niagara Workbench

You can examine the current status and version of the **KNX Data Defs** in use by the Niagara Workbench in the **KNX Import Service**:



In a Niagara Station

You can examine the current status and version of the **KNX Data Defs** in use by a Niagara station in the **Property Sheet** of the **KNX Network driver**:



To update the **KNX Data Defs**, see Updating the KNX Data Defs, page 25.

Updating the KNX Data Defs

To update the version of the **KNX Data Defs** in the Niagara Workbench and Niagara Station:

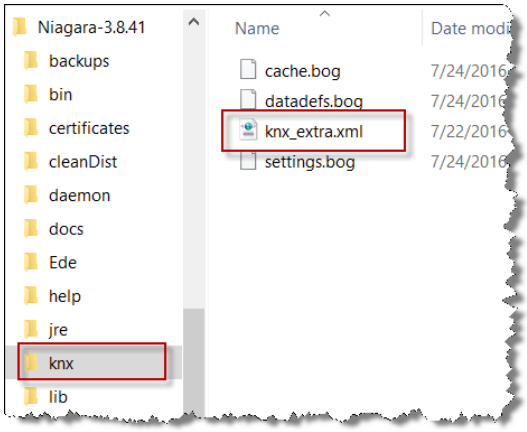
Prerequisites: Currently, new versions of the `knx_extra.xml` file are published to a *DropBox* folder available here:

`https://www.dropbox.com/sh/uwkdggd10fi1955/AAAhMkD6CoQtLx-A_DPEoguga?dl=0`

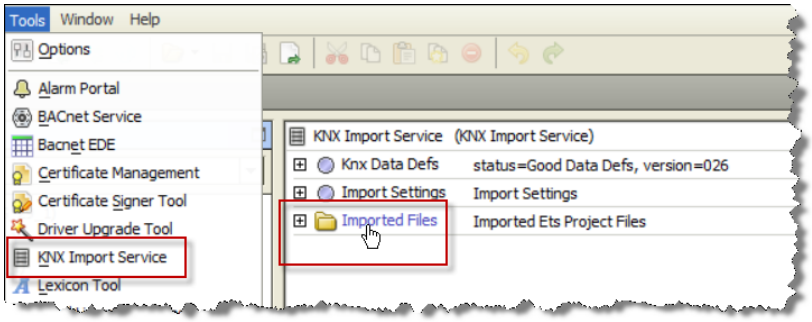
The latest version will always be `knx_extra.xml` in the root of this shared *DropBox* folder.

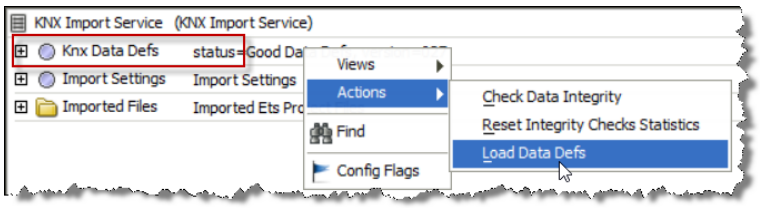
NOTE: Older versions are available in the `Previous Versions of knx_extra.xml` sub-folder, however it should not be necessary to use these and if one were required then it would need to be renamed to remove the `_vXXX` suffix.

Step 1 Having obtained the latest `knx_extra.xml` from *DropBox*, put it in the **knx** sub-folder of your Niagara installation on your PC (for example:. `C:\Niagara\Niagara-3.8.41.1\knx\knx_extra.xml`)

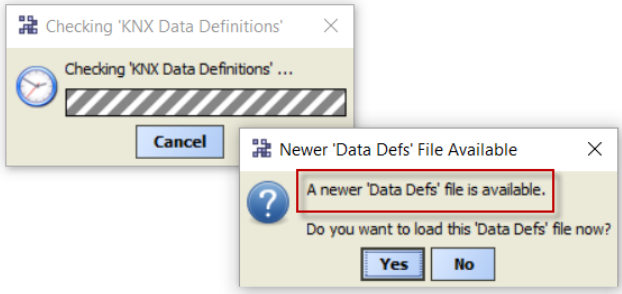


Step 2 Using the Niagara Workbench, navigate to the **KNX Import Service** on the **Tools** menu.



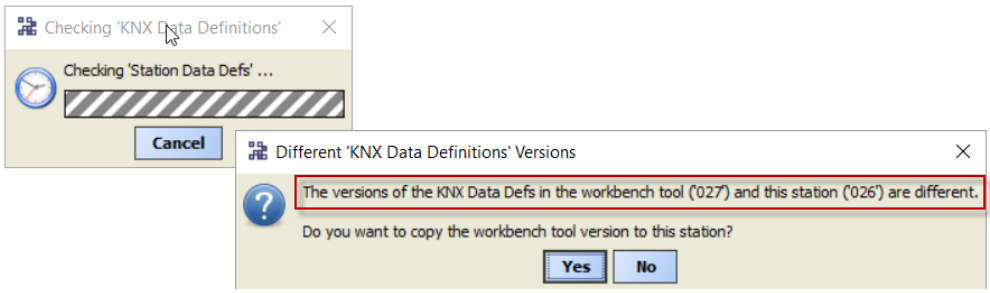


Step 3 Either right click **Knx Data Defs** and click the **Load Data Defs** action, or click on **Imported Files** to load the **Ets Project File Import Manager** which will examine the new `knx_extra.xml` file that was copied into the **knx** sub-folder of your Niagara installation. You will be presented with a dialogue prompting that there is a newer version available. Click **Yes**.

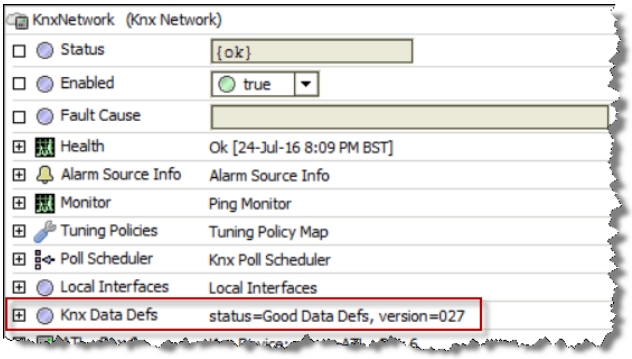


Now that the Niagara Workbench has been updated with a new **KNX Data Defs**, the Niagara Station can be updated.

Step 4 Navigate to the **KNX Device Manager** or **KNX Point Manager** views of the **KNXnet/IP driver**. As either view loads it will check the Station's version against the Workbench's version and prompt if the two versions are different. Click **Yes**.



*NOTE: There is an issue in v29 of the **KNX Network driver**, whereby after updating the **KNX Data Defs**, examination of the of the Niagara station in the **Property Sheet** of the **KNX Network driver** will still show the older version number in spite of the file being correctly updated. If the station is **Rebooted**, the updated version of the **KNX Data Defs** will be correctly shown.*



Chapter 4 Property reference

Topics covered in this chapter

Knx Network

The **Knx Network** component is the base container for all KNXnet/IP devices and their child data objects (KNX proxy points).

Property	Value	Description
Status [component]	text	Read-only field. Indicates the condition of the component at last polling. • `{ok}` indicates that the component is licensed and polling successfully. • `{down}` indicates that polling is unsuccessful, perhaps because of an incorrect property. • `{disabled}` indicates that the **`Enable`** property is set to `false`. • `{fault}` indicates another problem.
Enabled [general]	`true` or `false`	Activates and deactivates use of the component.

Property	Value	Description
Fault Cause	text	Read-only field. If the network, component, or extension is in fault, indicates the reason. This field is empty unless a fault exists.
Health	text	Health contains historical properties about the relative health of the network in the station, including historical timestamps. See Knx Network/Health, page 30
Alarm Source Info	Alarm Source Info	Alarm Source Info contains a set of properties for configuring and routing alarms and are used to populate an alarm if the network does not respond to a monitor ping. See Knx Network/Alarm Source Info, page 30
Monitor	Ping Monitor	Holds the configuration for the "ping mechanism" used by driver networks. Monitor provides verification of the general health of the network, plus the network's "pingables" (typically, devices) by ensuring that each device is minimally "pinged" at some repeating interval. See Knx Network/Monitor, page 31
Tuning Policies	Tuning Policy Map	See Knx Network/Tuning Policies, page 32
Poll Scheduler	Knx Poll Scheduler	See Knx Network/Poll Scheduler, page 33
Local Interfaces	Local Interfaces	See Knx Network/Local Interface, page 34
Knx Data Defs		See Knx Network/Knx Data Defs, page 36

Knx Network/Health

Health contains historical properties about the relative health of the network in the station, including historical timestamps.

Type	Value	Description
Down	`true` or `false` (default)	Displays the health of the network.
Alarm	`true` or `false` (default)	Displays the health of the network.
Last OK Time	`date time`	Displays the last date and time the network health was OK.
Last Fail Time	`date time`	Displays the last date and time the network health failed.
Last Fail Cause	text	Displays the reason for the last failure of the network health.

Knx Network/Alarm Source Info

Alarm Source Info contains a set of properties for configuring and routing alarms and are used to populate an alarm if the network does not respond to a monitor ping.

Type	Value	Description
Alarm Class	text	Use this property to select an alarm class from the option list. The alarm class specifies the alarm routing options for this component.
Source Name	text	Displays the name in an alarm record that identifies the source of the alarm. **NOTE:** For how to format this information on a report, click on the help icon to the right of the field.
To Fault Text	text	Enter the text that you would like to display when the component transitions to a Fault state. **NOTE:** For how to format this information on a report, click on the help icon to the right of the field.
To Offnormal Text	text	Enter the text that you would like to display when the component transitions to an Offnormal (alarm) state. **NOTE:** For how to format this information on a report, click on the help icon to the right of the field.
To Normal Text	text	Enter the text that you would like to display when the component transitions to a Normal state. **NOTE:** For how to format this information on a report, click on the help icon to the right of the field.
Hyperlink Ord or Hyperlink	Ord, BQL Query or path	Associates an ord, BLQ query or path with an alarm state on the component. When an alarm is reported in the console, the Hyperlink button activates. Clicking this button links to the location you specify here.
Sound File	ord	The path to a sound file that plays when the current component is in an alarm state. Use the folder icon to browse to the file. Click the arrow icon to the right of the folder icon to test the path.
Alarm Icon	file path	Use this property to enter or choose the path to a graphic file that will be added to the display in the "timestamp" column of the alarm table in the Console Recipient view. Use the folder icon to browse to the file that you want to use. Click the arrow icon to the right of the folder icon to test the path that you enter.
Alarm Instructions	# instructions	Each alarm can have "instructions" assigned to it so that any time an alarm is generated, the instructions are presented with the alarm notification to provide information that may be important or helpful to the user.
Meta Data [alarms]	text	Allows you to enter new facets for the extension.

Knx Network/Monitor

Holds the configuration for the “ping mechanism” used by driver networks. Monitor provides verification of the general health of the network, plus the network’s “pingables” (typically, devices) by ensuring that each device is minimally “pinged” at some repeating interval.

Property	Value	Description
Ping Enabled	`true` (default) or `false`	Enables or disables the Monitor Ping.
Ping Frequency	`hh—mm—ss` Defaults to 5secs	Sets the Ping Frequency.
Alarm On Failure	`true` (default) or `false`	Enables or disables an Alarm if the Monitor Ping fails.
Start Up Alarm Delay	`hh—mm—ss` Defaults to 2minutes 0secs	Sets a delay on station start up before the Monitor Ping Alarm can occur.

Knx Network/Tuning Policies

The network Tuning Policies holds one or more collections of *rules* for evaluating both **write requests** (e.g. to writable proxy points) as well as the acceptable *freshness* of **read requests** from polling.

Property	Value	Description
Default Policy	Tuning Policy	Holds information and configuration for the Default Tuning Policy. See Knx Network/Tuning Policies/Default Policy, page 32.
(Custom Tuning Policies)		

Knx Network/Tuning Policies/Default Policy

This is the default tuning policy which is always installed. More tuning policies may be created by copying this default policy.

Property	Value	Description
Min Write Time	`hh-mm-ss, 0ms ... +inf` Defaults to 00–00–00	Applies to writable proxy points, especially ones that have one or more linked inputs. Specifies the minimum amount of time allowed between writes. Provides a method to throttle rapidly changing values so that only the last value is written. If this property value is 0 (default), this rule is disabled (all value changes attempt to write).
Max Write Time	`hh-mm-ss, 0ms ... +inf` Defaults to 00–00–00	Applies to writable proxy points. Specifies the maximum *wait time* before rewriting the value, in case nothing else has triggered a write. Any write action resets this timer. If property value is 0 (default), this rule is disabled.
Write On Start	`true` (default) or `false`	Applies to writable proxy points. Determines behavior at station `startup`. • If `true`, (default) a write occurs when the station first reaches *steady state*. • If set to `false`, a write does not occur when the station reaches *steady state*. Note: Consider setting this to false in most tuning policies, except for tuning policies selectively assigned to more critical writable proxy points. This is particularly important for large networks with many writable proxy points. For example, a

Property	Value	Description
		Network with 4,000 writable proxy points, if configured with only the **Default Tuning Policy** (at default values), will upon station startup attempt to write to all 4,000 points, putting a significant load on the station. As a consequence, it is possible that in this scenario the driver (network) may generate **write queue overflow** exceptions.
Write On Up	`true` (default) or `false`	Applies to writable proxy points. Determines behavior when proxy point (and parent device) transitions from `down`. • If `true`, (default) a write occurs when the parent device transitions from `down` to `up`. • If set to `false`, a write does not occur when the parent device transitions from `down` to `up`.
Write On Enabled	`true` (default) or `false`	Applies to writable proxy points. Determines behavior when a proxy point's status transitions from `disabled` to `normal` (enabled). • If `true`, (default) a write occurs when writable point transitions from `disabled`. • If set to `false`, a write does not occur when writable point transitions from `disabled`. Note: The disabled-to-enabled status transition can be inherited globally by points if the parent device had been set to disabled—or network-wide if the driver network was set to disabled. Therefore, be aware that if left at `true` in tuning policies, that all associated writable points receive a write upon either the device or network when it transitions from status `disabled` to `enabled`.
Stale Time	`hh-mm-ss, 0ms ... +inf` Defaults to 00–00–00	Applies to all proxy points. • If set to a non-zero value, points become `stale` (status stale) if the configured time elapses without a successful read, indicated by Read Status `{ok}`. • If set to `zero` (default), the stale timer is disabled, and points become `stale` immediately when unsubscribed. By default, proxy point status `stale` is indicated by **tan** background color. In addition, `stale` status is considered *invalid* for any downstream-linked control logic.

Knx Network/Poll Scheduler

The network Poll Scheduler maintains a group of four *rate* buckets to service *pollables*, three of which correspond to configured poll rates (slow, normal and fast) and one (dibs stack) which is allocated for *pollables* that transition to a *subscribed* state.

Property	Value	Description
Poll Enabled	`true` (default) or `false`	Enables or Disables the Poll.
Fast Rate	`hh-mm-ss, 1ms ... +inf` Defaults to 00–00–01	Sets the Fast Rate of Polling.

Property	Value	Description
Normal Rate	`hh-mm-ss, 1ms ... +inf` Defaults to 00–00–05	Sets the Normal Rate of Polling.
Slow Rate	`hh-mm-ss, 1ms ... +inf` Defaults to 00–00–30	Sets the Slow Rate of Polling.
Statistics Start	`date time`	Read-only field.
Average Poll	`time`	Read-only field.
Busy Time	`time`	Read-only field.
Total Polls	`Numeric` over `time`	Read-only field.
Dibs Polls	`Numeric` % (`Numeric/Numeric`)	Read-only field.
Fast Polls	`Numeric` % (`Numeric/Numeric`)	Read-only field.
Normal Polls	`Numeric` % (`Numeric/Numeric`)	Read-only field.
Slow Polls	`Numeric` % (`Numeric/Numeric`)	Read-only field.
Dibs Count	current=`numeric` average=`numeric`	Read-only field.
Fast Count	current=`numeric` average=`numeric`	Read-only field.
Normal Count	current=`numeric` average=`numeric`	Read-only field.
Slow Count	current=`numeric` average=`numeric`	Read-only field.
Fast Cycle Time	average = `time`	Read-only field.
Normal Cycle Time	average = `time`	Read-only field.
Slow Cycle Time	average = `time`	Read-only field.

Knx Network/Local Interface

Holds information and configuration for the Local Interface.

Property	Value	Description
Status	`text`	Read-only field. Displays the current state of the Local Interface.
Enabled	`true` (default) or `false`	Enables or Disables the Local Interface.
Fault Cause	`text`	Read-only field. If the Local Interface is in fault, indicates the reason. This field is empty unless a fault exists.

Property	Value	Description
Config Status	`text`	Read-only field. Displays the current configuration state of the Local Interface.
Local Interface Id	`Selection Dialogue`	Displays the available Adapters in a selection box dialogue.
Adapter Description	`text`	Read-only field.
Adapter IP Address	`text`	Read-only field.
Local Port Min	`numeric, 1 — 65535` Defaults to 3500	When dynamically choosing a new port number, the driver will choose the next available port number by starting at this setting or by cycling through the range. This choice of behavior being selectable by **Obtain Local Socket Behaviour**. See UDP ports, page 58.
Local Port Max	`numeric, 1 — 65535` Defaults to 4000	When dynamically choosing a new port number, the driver will choose the next available port number by starting at **Local Port Min** or by cycling through the range to a maximum of this setting. This choice of behavior being selectable by **Obtain Local Socket Behaviour**. See UDP ports, page 58.
Last Used Local Port	`numeric`	Read-only field.
Obtain Local Socket Behaviour	`Recycle Ports Immediately` (default) or `Cycle Through All Ports`	When dynamically choosing a new port number, the driver will choose the next available port number by starting at the port-minimum (default setting) or by cycling through the range. See UDP ports, page 58

Knx Network/Local Interface/Knx Installation

Holds information and configuration for the KNX Installation.

Property	Value	Description
Status	`text`	Read-only field. Displays the current state of the KNX Installation.
Enabled	`true` (default) or `false`	Enables or Disables the KNX Installation.
Fault Cause	`text`	Read-only field. If the KNX Installation is in fault, indicates the reason. This field is empty unless a fault exists.
Config Status	`text`	Read-only field. Displays the current configuration state of the KNX Installation.
Knx Installation Id	`numeric`	Read-only field. Displays the ID which has been read from the KNX Installation.
Multicast Ip Address	`nnn.nnn.nnn.nnn` Defaults to 224.0.23.12	

Property	Value	Description
Multicast Port Number	`numeric`	Read-only field.
Maximum Received Packets Que Size	`numeric, 2 — max` Defaults to 5	

Knx Network/Local Interface/Knx Installation/Multicast End Point

Holds information and configuration for the Multicast End Point.

Property	Value	Description
End Point State	`text`	Read-only field. Displays the current state of the End Point.
Local IP Address	`nnn.nnn.nnn.nnn`	Read-only field. Displays the Local IP Address.
Local Port	`numeric`	Read-only field. Displays the Local Port.
Remote IP Adddress	`nnn.nnn.nnn.nnn`	Read-only field. Displays the Remote IP Address.
Remote Port	`numeric`	Read-only field. Displays the Remote Port.
Max Packet Size	`numeric, 128 — 512` Defaults to 128.	

Knx Network/Knx Data Defs

Knx Data Defs properties are part of the **KNX Import Service**. **Knx Data Defs** is a container for the specifications of the **KNX Datapoint Types** which are imported from the `knx_extra.xml` file.

Knx Device

The **Knx Device** component is the container for all KNXnet/IP proxy points.

Property	Value	Description
Status [component]	text	Read-only field. Indicates the condition of the component at last polling. • `{ok}` indicates that the component is licensed and polling successfully. • `{down}` indicates that polling is unsuccessful, perhaps because of an incorrect property. • `{disabled}` indicates that the **`Enable`** property is set to `false`. • `{fault}` indicates another problem.
Enabled [general]	`true` or `false`	Activates and deactivates use of the component.
Fault Cause	text	Read-only field. If the network, component, or extension is in fault, indicates the reason. This field is empty unless a fault exists.

Property	Value	Description
Health	text	Health contains historical properties about the relative health of the device in the station, including historical timestamps. See Knx Device/Health, page 37.
Alarm Source Info	Alarm Source Info	Alarm Source Info contains a set of properties for configuring and routing alarms and are used to populate an alarm if the device does not respond to a monitor ping. See Knx Device/Alarm Source Info, page 38.
Config Status	`string`	Displays the status of the configuration (read only).
Device Id	`1, 2, ... max`	Displays Device Id (read only)
IP Address	`nnnn.nnnn.nnnn.nnnn`	The IP Address of the device.
Control Port Number	`-1, 1, 2, ... 65535`	The Control Port Number of the device.
Individual Device Address	`nnn.nnn.nnn`	Individual Device Address
Mac Address	`string`	Displays the Mac Address of the KNXnet/IP Interface device (read only).
Device Friendly Name	`string`	Displays the Friendly Name of the KNXnet/IP Interface device (read only).
DIBs	Description Information Blocks	Holds information on the Device. See Knx Device/DIBs, page 39.
Group Data Manager	Group Data Manager	Holds information and configuration for the Group Data Manager. See Knx Device/Group Data Manager, page 39.
Points	Knx Point Device Ext	Holds information and configuration for the KNX proxy points. See Knx Proxy Ext, page 46.
Connection Method	`Tunnelling` (default) or `Proxy Routing`	The Connection Method.
Comms Connections	Connections	Holds information and configuration of the Comms connections. See Knx Device/Comms Connections, page 40.

Knx Device/Health

Health contains historical properties about the relative health of the device in the station, including historical timestamps.

Property	Value	Description
Down	`true` or `false` (default)	Displays the health of the device (read only).
Alarm	`true` or `false` (default)	Displays the health of the device (read only).
Last OK Time	`date time`	Displays the last date and time the device health was OK (read only).

Property	Value	Description
Last Fail Time	`date time`	Displays the last date and time the device health failed (read only).
Last Fail Cause	text	Displays the reason for the last failure of the device health (read only).

Knx Device/Alarm Source Info

Alarm Source Info contains a set of properties for configuring and routing alarms and are used to populate an alarm if the device does not respond to a monitor ping.

Type	Value	Description
Alarm Class	text	Use this property to select an alarm class from the option list. The alarm class specifies the alarm routing options for this component.
Source Name	text	Displays the name in an alarm record that identifies the source of the alarm. **NOTE:** For how to format this information on a report, click on the help icon to the right of the field.
To Fault Text	text	Enter the text that you would like to display when the component transitions to a Fault state. **NOTE:** For how to format this information on a report, click on the help icon to the right of the field.
To Offnormal Text	text	Enter the text that you would like to display when the component transitions to an Offnormal (alarm) state. **NOTE:** For how to format this information on a report, click on the help icon to the right of the field.
To Normal Text	text	Enter the text that you would like to display when the component transitions to a Normal state. **NOTE:** For how to format this information on a report, click on the help icon to the right of the field.
Hyperlink Ord or Hyperlink	Ord, BQL Query or path	Associates an ord, BLQ query or path with an alarm state on the component. When an alarm is reported in the console, the Hyperlink button activates. Clicking this button links to the location you specify here.
Sound File	ord	The path to a sound file that plays when the current component is in an alarm state. Use the folder icon to browse to the file. Click the arrow icon to the right of the folder icon to test the path.
Alarm Icon	file path	Use this property to enter or choose the path to a graphic file that will be added to the display in the "timestamp" column of the alarm table in the Console Recipient view. Use the folder icon to browse to the file that you want to use. Click the arrow icon to the right of the folder icon to test the path that you enter.

Type	Value	Description
Alarm Instructions	# instructions	Each alarm can have "instructions" assigned to it so that any time an alarm is generated, the instructions are presented with the alarm notification to provide information that may be important or helpful to the user.
Meta Data [alarms]	text	Allows you to enter new facets for the extension.

Knx Device/DIBs

Description Information Blocks.

Property	Value	Description
Device Info	Device Info D I B	Holds information for the KNX/netIP Iterface device. See Knx Device/DIBs/Device Info, page 39.
Supported Service Families	`No Supported Service Families`	Supported Service Families (read only).

Knx Device/DIBs/Device Info

Description Information Blocks of the KNXnet/IP Interface device.

Property	Value	Description
Knx Medium	`alphanumeric`	Knx Medium (read only).
Device Status	`alphanumeric`	Device Status (read only).
Individual Address	`nnn.nnn.nnn`	Individual Address (read only).
Project Number	`alphanumeric`	Project Number (read only).
Installation Number	`alphanumeric`	Installation Number (read only).
Serial Number	`alphanumeric`	Serial Number (read only).
Routing Multicast Address	`nnn.nnn.nnn.nnn`	Routing Multicast Addres (read only).
Mac Address	`alphanumeric`	Mac Address (read only).
Device Friendly Name	`alphanumeric`	Device Friendly Name (read only).

Knx Device/Group Data Manager

Group Data Manager.

Property	Value	Description
L Data Worker (hidden)	Knx Worker	Holds the configuration for the Link Data Worker. See Knx Device/Group Data Manager/L Data Worker, page 40
Hop Count (hidden)	`0, 1 , 2 ... 7` Defaults to 6	Each communications frame contains a hop count. A frame's hop count is decremented by the KNXnet/IP Router devices to avoid looping messages. When it becomes zero, the frame is discarded from the KNX network.
Max Pending Reads	`1, 2, 3 ... 10` Defaults to 4	Controls how many concurrently *Active* **Group Address Read Operations** are allowed. *Active* means that a particular operation has reached the head of the **Group Data Operation Queue** and the Comms.Stack has sent an `L_Data_req` message to the KNXnet/IP Interface devices and received an `L_Data_con` reply, but has not yet received a corresponding `L_Data_ind` message. This limit is intended to prevent the KNX twisted-pair line from being swamped with this driver's traffic although there is no clear guidance on this in the KNX specifications.
Read Before Write Timeout	`100ms ... 6secs` Defaults to 0.5 secs	Read Before Write Timeout.

Knx Device/Group Data Manager/L Data Worker

L Data Worker holds the configuration for the Link Data Worker. It is a hidden slot.

Property	Value	Description
Max Queue Size (hidden)	`1, 2, 3 ... max` Defaults to 50	This is the **Group Data Operation Queue**. It holds a list of Group Data Operations (Group Address reads or writes) waiting to be started as soon as the comms. stack is able.

Knx Device/Comms Connections

Connections.

Property	Value	Description
Maximum Received Packets Que Size	`2, 3 ... max` Defaults to 5	Controls the size of the **Control End Point** receive queue. Because the **Control End Point** is only used to connect, maintain and disconnect the **Tunnel Connection**, it sees very little traffic. (only 2 or 3 messages to connect or disconnect and 1 or 2 messages per minute to maintain the connection). If this value were too small it would show up in the **Rx Frames Lost Queue Full** counter under the Knx Device/Comms Connections/Comms Counters, page 42.
Control End Point	End Point: `state=Closed` or `state=Open`	Holds information and configuration for the Control End Point. See Knx Device/Comms Connections/Control End Point, page 41.
Comms Counters (hidden)	Connections Comms Counters	Holds information on the Communications Counters. See Knx Device/Comms Connections/Comms Counters, page 42.

Property	Value	Description
Include In Trace (hidden)	`true` (default) or `false`	Controls whether or not this information is included in the station Spy Log.
Tunnel Conn	Tunnel Connection	Holds information and configuration for the Tunnel Connection. See Knx Device/Comms Connections/Tunnel Conn, page 42.

Knx Device/Comms Connections/Control End Point

Holds information and configuration for the Control End Point.

Property	Value	Description
End Point State	`state=Closed` or `state=Open`	The current state of the End Point (read-only).
Local IP Address	`nnn.nnn.nnn.nnn`	(read-only) The Local IP address (read-only).
Local Port	`nnnn` Defaults to —1 if no connection	The Local Port number (read-only).
Remote IP Address	`nnn.nnn.nnn.nnn`	The IP address of the KNXnet/IP Interface device (read-only).
Remote Port	`nnnn` Defaults to —1 if no connection	The Port number at KNXnet/IP Interface device (read-only).
Max Packet Size	`128 ... 512` Defaults to 128	The Maximum size of a packet.
Comms Counters (hidden)	End Point Comms Counters	Holds information on the Communications Counters. See Knx Device/Comms Connections/Control End Point/Comms Counters, page 41
Include In Trace (hidden)	`true` (default) or `false`	Controls whether or not this information is included in the station Spy Log.

Knx Device/Comms Connections/Control End Point/Comms Counters

Communications statistics counters for the Control End Point. By default, this is a hidden folder.

Property	Value	Description
Invalid Frame Headers (hidden)	`0, 1, 2 ... max`	Invalid Frame Headers (read-only).
Rx Own Packets Ignored (hidden)	`0, 1, 2 ... max`	Rx Own Packets Ignored (read-only).
Packets Sent (hidden)	`0, 1, 2 ... max`	Packets Sent (read-only).
Packets Received (hidden)	`0, 1, 2 ... max`	Packets Received (read-only).

Knx Device/Comms Connections/Comms Counters

Communications statistics counters for the Comms Connections. By default, this is a hidden folder.

Property	Value	Description
Rx Frames Lost Queue Full (hidden)	`0, 1, 2 ... max`	Rx Frames Lost Queue Full (read-only). The **Maximum Received Packets Que Size** (see Knx Device/Comms Connections, page 40) controls the size of the **Control End Point's** receive queue and it has a default of 5. If this value were too small, there would be an indication of **lost** frames here.
Rx Frames Lost No Packet Worker (hidden)	`0, 1, 2 ... max`	Rx Frames Lost No Packet Worker (read-only).

Knx Device/Comms Connections/Tunnel Conn

Tunnel Connection.

Property	Value	Description
Maximum Received Packets Que Size	`2, 3 ... max` Defaults to 50	Maximum Received Packets Que Size. Controls the size of the Tunnel Connection's **Data End Point** receive queue. All **Group Address** messages pass through this queue and End Point so it is much busier than the **Control End Point** (see KNXnet/IP Tunnel Connection — Control, page 51). Because it handles more traffic, its default queue size of 50 is larger than that of the **Control End Point**.
Inter Message Delay	`15ms ... 5secs` Defaults to 15ms	Inter Message Delay. This setting exists to 'pace' the **out-going data requests** from the driver. This can be used to reduce the rate of data requests in cases where a KNXnet/IP Interface device cannot cope with the traffic volume, caused possibly by its implementation settings or by its operating speed. The default setting of minimum 15ms has proved successful with **Siemens** interface devices. Any problems arising from this being set too small would manifest as Control Points intermittently having a **Read Fault: Timed out waiting for L_Data_con** fault condition.
Remote Control Hpai	Knx Hpai	Holds information on the Knx Hpai. See Knx Device/Comms Connections/Tunnel Conn/Remote Control Hpai, page 43.
Data End Point	`state=Closed` or `state=Open`	The current state of the Data End Point (read-only). See Knx Device/Comms Connections/Tunnel Conn/Data End Point, page 43.
Last Connect Error	`string`	Last Connect Error (read only).
Channel Id	`0 ... 255` Defaults to –1 if no connection	Channel Id (read only).
Channel Status	`no connection` or `connection`	Channel Status (read only).

Property	Value	Description
Always Send Heart Beats	`true` (default) or `false`	Always Send Heart Beats.
Wrong Seq Number Reaction (hidden)	`Send Disconnect Request` (default) or `Just Ignore` or `Send NAK E_Sequence_Number`	Defines what reaction the driver should make to a Wrong Sequence Number.
Include In Trace (hidden)	`true` (default) or `false`	Controls whether or not this information is included in the station Spy Log.
Individual Address	`nn.nn.nn`	Individual Address (read only).
Confirmation Timeout (hidden)	`hh mm ss`	Confirmation Timeout (read only).
Comms Counters (hidden)	Tunnel Connection Comms Counters	Holds information on the Communications Counters. See Knx Device/Comms Connections/Tunnel Conn/Comms Counters, page 44.
Request Acknowledgements (hidden)	`true` (default) or `false`	Request Acknowledgements.

Knx Device/Comms Connections/Tunnel Conn/Remote Control Hpai

KNX Hpai.

Property	Value	Description
IP Address	`nnn.nnn.nnn.nnn`	The IP address (read-only).
Port	`0, 1, ... 65535` Defaults to —1 if no connection	The Port number (read-only).

Knx Device/Comms Connections/Tunnel Conn/Data End Point

Data End Point.

Property	Value	Description
End Point State	`state=Closed` or `state=Open`	The current state of the End Point (read-only).
Local IP Address	`nnn.nnn.nnn.nnn`	(read-only) The Local IP address (read-only).
Local Port	`nnnn` Defaults to —1 if no connection	The Local Port number (read-only).
Remote IP Address	`nnn.nnn.nnn.nnn`	The IP address of the KNXnet/IP Interface device (read-only).
Remote Port	`nnnn` Defaults to —1 if no connection	The Port number at KNXnet/IP Interface device (read-only).

Property	Value	Description
Max Packet Size	`128 ... 512` Defaults to 128	The Maximum size of a packet.
Comms Counters (hidden)	End Point Comms Counters	Holds information on the Communications Counters. See Knx Device/Comms Connections/Tunnel Conn/Data End Point/Comms Counters, page 44.
Include In Trace (hidden)	`true` (default) or `false`	Controls whether or not this information is included in the station Spy Log.

Knx Device/Comms Connections/Tunnel Conn/Data End Point/Comms Counters

Communications statistics counters for the Data End Point. By default, this is a hidden folder.

Property	Value	Description
Invalid Frame Headers (hidden)	`0, 1, 2 ... max`	Invalid Frame Headers (read-only).
Rx Own Packets Ignored (hidden)	`0, 1, 2 ... max`	Rx Own Packets Ignored (read-only).
Packets Sent (hidden)	`0, 1, 2 ... max`	Packets Sent (read-only).
Packets Received (hidden)	`0, 1, 2 ... max`	Packets Received (read-only).

Knx Device/Comms Connections/Tunnel Conn/Comms Counters

Communications statistics counters for the Tunnel Connection. By default, this is a hidden folder.

Property	Value	Description
Rx Frames Lost Queue Full (hidden)	`0, 1, 2 ... max`	Rx Frames Lost Queue Full (read-only). You can monitor the success or otherwise of received data by inspecting this counter. Depending on the number and frequency of lost **Rx Frames**, you can try increasing the **Maximum Received Packets Que Size** (see Knx Device/Comms Connections/Tunnel Conn, page 42) in steps of 50, until no more **Rx Frames** are being lost.
Rx Frames Lost No Packet Worker (hidden)	`0, 1, 2 ... max`	Rx Frames Lost No Packet Worker (hidden)
Frames Received (hidden)	`0, 1, 2 ... max`	Frames Received (read-only).
Frames Received Wrong Source IP Address (hidden)	`0, 1, 2 ... max`	Frames Received Wrong Source IP Address (read-only).
Invalid Frames Received (hidden)	`0, 1, 2 ... max`	Invalid Frames Received (read-only).

Property	Value	Description
Closed Because All Clients Unregistered (hidden)	0, 1, 2 ... max	Closed Because All Clients Unregistered (read-only).
Closed Because Rx Packet Queue Size Changed (hidden)	0, 1, 2 ... max	Closed Because Rx Packet Queue Size Changed (read-only).
Closed Because Connection State Response Timeout (hidden)	0, 1, 2 ... max	Closed Because Connection State Response Timeout (read-only).
Closed Because Connections Processor Stopping (hidden)	0, 1, 2 ... max	Closed Because Connections Processor Stopping (read-only).
Good Ack Received (hidden)	0, 1, 2 ... max	Good Ack Received (read-only).
Rx Wrong Service Type Frames (hidden)	0, 1, 2 ... max	Rx Wrong Service Type Frames (read-only).
Closed Because Ack Error Received (hidden)	0, 1, 2 ... max	Closed Because Ack Error Received (read-only).
Closed Because No Ack Received (hidden)	0, 1, 2 ... max	Closed Because No Ack Received (read-only).
Ack Frames Sent (hidden)	0, 1, 2 ... max	Ack Frames Sent (read-only).
Request Frames Sent (hidden)	0, 1, 2 ... max	Request Frames Sent (read-only).
Rx Frames With Wrong Channel Id (hidden)	0, 1, 2 ... max	Rx Frames With Wrong Channel Id (read-only).
Rx Frames Correct Sequence Number (hidden)	0, 1, 2 ... max	Rx Frames Correct Sequence Number (read-only).
Rx Frames Repeated Sequence Number (hidden)	0, 1, 2 ... max	Rx Frames Repeated Sequence Number (read-only).
Rx Frames Wrong Sequence Number (hidden)	0, 1, 2 ... max	Rx Frames Wrong Sequence Number (read-only).
Closed Because Debug Rx Wrong Sequence Number (hidden)	0, 1, 2 ... max	Closed Because Debug Rx Wrong Sequence Number (read-only).

Property	Value	Description
Rx Ack Frames (hidden)	`0, 1, 2 ... max`	Rx Ack Frames (read-only).
Rx Unknown Channel Sub Service (hidden)	`0, 1, 2 ... max`	Rx Unknown Channel Sub Service (read-only).
Good Confirm Received (hidden)	`0, 1, 2 ... max`	Good Confirm Received (read-only).
Error Confirm Received (hidden)	`0, 1, 2 ... max`	Error Confirm Received (read-only).
No Confirm Received (hidden)	`0, 1, 2 ... max`	No Confirm Received (read-only).

Knx Proxy Ext

This Proxy Ext is common to all points, both Read and Write types of Boolean, Numeric, String and Enum.

All **KNXnet/IP proxy extension** types (`KnxBooleanProxyExt`, `KnxNumericProxyExt`, `KnxStringProxyExt` and `KnxEnumProxyExt`) share the same set of configuration properties. Any instance of any of the **KNXnet/IP proxy extension** types is a proxy for one or more group address in a KNX Installation.

The **KNXnet/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 **KNXnet/IP proxy extension** types are the point-level component in the Niagara architecture.

Type	Value	Description
Status [component]	text	Read-only field. Indicates the condition of the component at last polling. • `{ok}` indicates that the component is licensed and polling successfully. • `{down}` indicates that polling is unsuccessful, perhaps because of an incorrect property. • `{disabled}` indicates that the **`Enable`** property is set to `false`. • `{fault}` indicates another problem.
Fault Cause	text	Read-only field. If the network, component, or extension is in fault, indicates the reason. This field is empty unless a fault exists.
Enabled	`true` or `false`	Enables or Disables communication to the associated device from the KNXnet/IP driver.
Device Facets		
Conversion		
Tuning Policy Name		
Read Value		

Type	Value	Description
Write Value		
KNX ID		
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 Value Type ID		
Poll Enable	`true` or `false`	Enable or Disable to add/remove the group address(es) configured on this proxy extension, to the `Poll Scheduler`. *Note that in KNX, 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	`true` or `false`	This is 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 independently of any other poll setting (for instance, it will occur even if `Poll Enabled` is false, or if the `Poll Scheduler` rate is zero or disabled.) The behavior can be modified by the `Poll Until Answer Received On Poll Once` property.
Poll once on operational	`true` or `false`	This is 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 independently of any other poll setting (for instance, it will occur even if `Poll Enabled` is false, or if the `Poll Scheduler` rate is zero or disabled.) The behavior can be modified by the `Poll Until Answer Received On Poll Once` property.
Poll until answer after poll once	`true` or `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 data addressed to the first group address in the list provided 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	`true` or `false`	This is independent of the `Poll Enable` property. It is used to enable or disable a poll for a value after a write.
Poll frequency	`Fast`, `Normal`, or `Slow`	Poll rates are determined by the `Poll Scheduler` settings.

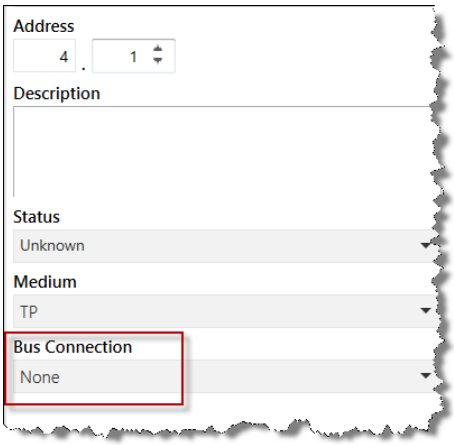
Chapter 5 FAQ's

Topics covered in this chapter

- ETS Bus Connection
- ETS Different Projects
- ETS Project Password
- KNXnet/IP Interface device
- KNXnet/IP Router device
- KNXnet/IP device address
- KNXnet/IP device — Multicast
- KNXnet/IP Tunnelling
- KNXnet/IP Tunnel Connection — Control
- KNXnet/IP Tunnel Connection — Data
- KNXnet/IP Routing
- KNXnet/IP driver conflict
- KNX data request rate
- KNX network — Duplicate
- KNX traffic overload protection
- Multicast
- Proxy Routing
- Points in Fault — KNXnet/IP Interface
- Points in Fault — Unsolicited
- Stale points — KNX project
- The 'Read' queue is full
- UDP ports
- Writing to Compound Structures

ETS Bus Connection

Does the 'Bus Connection' property require configuration in ETS?

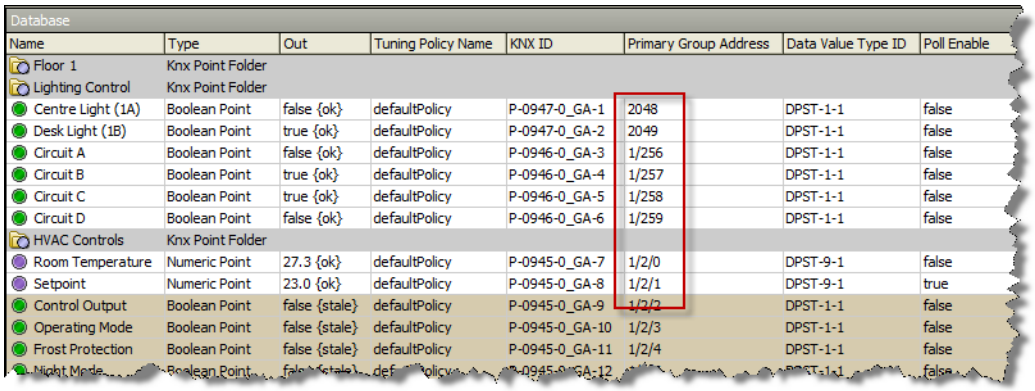


No. It is not necessary to configure the **Bus Connection** property in ETS in order to import devices from an exported ETS Project file.

ETS Different Projects

Can I Discover points from different ETS projects and add them to the same Niagara Device?

Yes you can. You can also add points from different projects which use different **Group Address Style** strategies. In this example, points from projects with Two Level, Three Level and Free Group Address Styles are all added into the same Niagara Device.

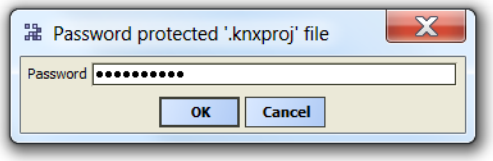


Database

Name	Type	Out	Tuning Policy Name	KNX ID	Primary Group Address	Data Value Type ID	Poll Enable
Floor 1	Knx Point Folder						
Lighting Control	Knx Point Folder						
Centre Light (1A)	Boolean Point	false {ok}	defaultPolicy	P-0947-0_GA-1	2048	DPST-1-1	false
Desk Light (1B)	Boolean Point	true {ok}	defaultPolicy	P-0947-0_GA-2	2049	DPST-1-1	false
Circuit A	Boolean Point	false {ok}	defaultPolicy	P-0946-0_GA-3	1/256	DPST-1-1	false
Circuit B	Boolean Point	true {ok}	defaultPolicy	P-0946-0_GA-4	1/257	DPST-1-1	false
Circuit C	Boolean Point	true {ok}	defaultPolicy	P-0946-0_GA-5	1/258	DPST-1-1	false
Circuit D	Boolean Point	false {ok}	defaultPolicy	P-0946-0_GA-6	1/259	DPST-1-1	false
HVAC Controls	Knx Point Folder						
Room Temperature	Numeric Point	27.3 {ok}	defaultPolicy	P-0945-0_GA-7	1/2/0	DPST-9-1	false
Setpoint	Numeric Point	23.0 {ok}	defaultPolicy	P-0945-0_GA-8	1/2/1	DPST-9-1	true
Control Output	Boolean Point	false {stale}	defaultPolicy	P-0945-0_GA-9	1/2/2	DPST-1-1	false
Operating Mode	Boolean Point	false {stale}	defaultPolicy	P-0945-0_GA-10	1/2/3	DPST-1-1	false
Frost Protection	Boolean Point	false {stale}	defaultPolicy	P-0945-0_GA-11	1/2/4	DPST-1-1	false
Night Mode	Boolean Point	false {stale}	defaultPolicy	P-0945-0_GA-12		DPST-1-1	false

ETS Project Password

A password protected dialogue is presented when I try to discover Devices and Points. Why is this and what Password do I enter?



The KNX ETS tool has the capability of exporting a project with a Password and it can only be opened by the KNXnet/IP driver if the correct credentials are presented. You will need to enter the same password that was used in the project setup of ETS.

KNXnet/IP Interface device

What is a KNXnet/IP Interface device?

A **KNXnet/IP Interface device** is a hardware device which supports KNXnet/IP Tunnelling only. A single interface, such as the **Weinzierl 730** and the **Siemens N 148/22** may support multiple KNXnet/IP Tunnelling connections. These devices can have simultaneous tunnelling connections which are managed by defining multiple KNX individual device addresses on the device. In the KNXnet/IP driver, this is called the **Individual Device Address** and allows the KNX network to be accessed by both ETS and the KNXnet/IP driver.

KNXnet/IP Router device

What is a KNXnet/IP Router device?

A **KNXnet/IP Router device** is a hardware device which supports both KNXnet/IP Tunnelling and KNXnet/IP Routing connections as well as having the filter table to allow the device to perform as a coupler. Some KNXnet/IP Routers, such as the **Siemens N 146/02** allow multiple Tunnelling connections as well.

KNXnet/IP device address

What KNXnet/IP device address information does the KNX driver need and how does it obtain it?

The two primary KNXnet/IP device types are **Interfaces** and **Routers**. There are three addresses the KNXnet/IP driver needs to communicate with either of these device types:

- **Ip Address**: Obtained from the KNXnet/IP device either over the KNX network using the KNXnet/IP driver's **Search Network For Devices** feature or by **Manual** data entry.
- **Control Port Number**: Obtained from the KNXnet/IP device either over the KNX network using the KNXnet/IP driver's **Search Network For Devices** feature or by **Manual** data entry or by **default**.

  *NOTE: The **default Control Port Number** is `3671`.*
- **Individual Device Address**: Obtained from the KNXnet/IP device either over the KNX network using the KNXnet/IP driver's **Search Network For Devices** feature or by **Manual** data entry or by using the KNXnet/IP driver's **Import Devices from `knxproj` file**

  *NOTE: **Manual** data entry is not available in v29 of the KNXnet/IP driver.*

The KNXnet/IP driver's **Search Network For Devices** (**Device Discovery**) feature uses a **Multicast** connection.

*NOTE: If both the **Ip Address** and **Control Port Number** are known by the KNXnet/IP driver, then it will not initiate a **Search Request** via a **Multicast** connection.*

KNXnet/IP device — Multicast

Does the KNX driver support KNXnet/IP devices which do not support Multicast?

Yes, providing the **Ip Address** and **Control Port Number** are known to the KNXnet/IP driver. They can both be entered **Manually**.

*NOTE: **Multicast** connection support is part of the **`Core: KNXnet/IP`** service, which is mandatory for all certified KNXnet/IP devices.*

KNXnet/IP Tunnelling

What is KNXnet/IP Tunnelling?

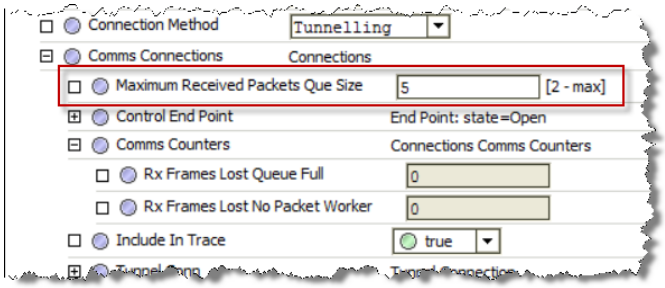
KNXnet/IP Tunnelling is the primary method of interfacing to a KNX system. It allows for **Unicast** communication from a single external device to the KNX system. This is akin to using a USB or Serial Interface to interface to the KNX system.

The KNXnet/IP driver supports **KNXnet/IP Tunnelling**.

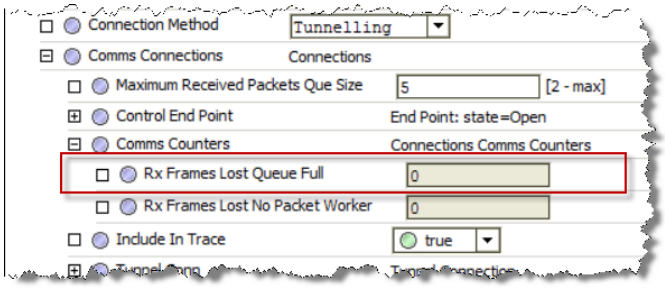
KNXnet/IP Tunnel Connection — Control

What maintains the 'Tunnel Connection'?

The **Control End Point** is used to connect, maintain and disconnect the **Tunnel Connection**. To be more precise, **Control** of the Tunnel Connection is the **only** thing the **Control End Point** does. The **Control End Point** sees very little traffic, only 2 or 3 messages to connect or disconnect and 1 or 2 messages per minute to maintain the connection.



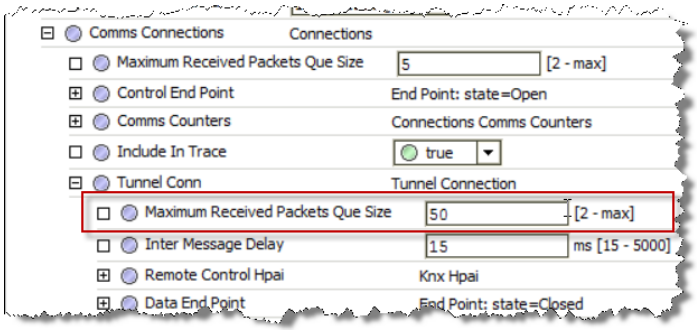
The **Maximum Received Packets Que Size** (see Knx Device/Comms Connections, page 40) controls the size of the **Control End Point's** receive queue and it has a default of 5. If this value were too small, there would be an indication of **lost** frames in the hidden **Rx Frames Lost Queue Full** Communications Counter (see Knx Device/Comms Connections/Comms Counters, page 42).



KNXnet/IP Tunnel Connection — Data

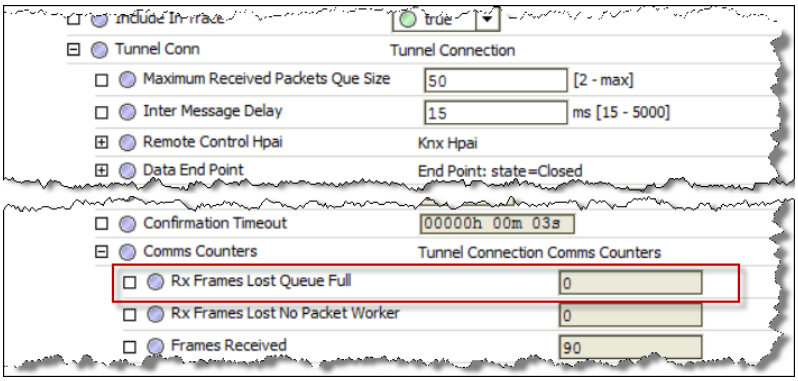
Which queue is used for data through the 'Tunnel Connection'?

The **Maximum Received Packets Que Size** (see Knx Device/Comms Connections/Tunnel Conn, page 42) controls the size of the Tunnel Connection's **Data End Point** receive queue. All **Group Address** messages pass through this queue and End Point so it is much busier than the **Control End Point** (see KNXnet/IP Tunnel Connection — Control, page 51). Because it handles more traffic, its default queue size of 50 is larger than that of the **Control End Point**.



Unlike the **Group Data Operation Queue** (see The 'Read' queue is full, page 58), the necessary size of this queue doesn't depend on the number and frequency of Control Point polling. Instead it depends on how quickly the KNXnet/IP driver can process incoming messages, which in turn depends on the overall CPU usage in the Platform.

You can monitor the success or otherwise of received data by inspecting the **Rx Frames Lost Queue Full** counter in the hidden Communications Counters (see Knx Device/Comms Connections/Tunnel Conn/Comms Counters, page 44). Depending on the number and frequency of lost **Rx Frames**, you can try increasing the **Maximum Received Packets Que Size** in steps of 50, until no more **Rx Frames** are being lost.



KNXnet/IP Routing

What is KNXnet/IP Routing?

KNXnet/IP Routing is a **Multicast**-based telegram, which allows a KNXnet/IP router to perform the function of a line or area coupler. This means the backbone of a KNX system can be Ethernet-based, allowing a much higher speed of transmission and more flexibility when installing. The KNXnet/IP router will also manage a filter table to manage the flow of traffic where needed.

The KNXnet/IP driver does not support **KNXnet/IP Routing**.

KNXnet/IP driver conflict

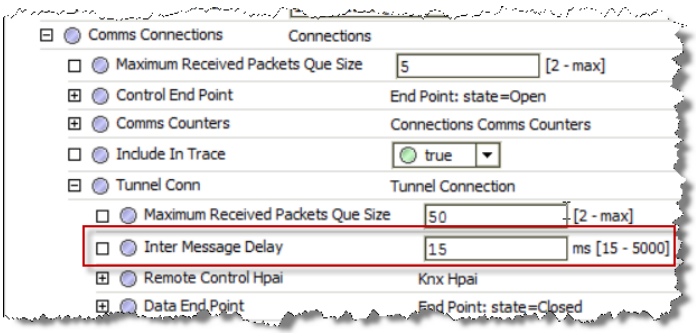
Can I have both the KNXnet/IP driver and EIBnet/IP driver running in my Niagara station?

No. There is an absolute certainty of conflict between the **KNXnet/IP driver** and the *older* **EIBnet/IP driver** if they are running concurrently. If there is an **EIBnet/IP driver** in the Niagara station then it must be deleted from the station before the **KNXnet/IP driver** is enabled.

KNX data request rate

What controls the rate of KNX data requests from the driver?

The **Inter Message Delay** (see Knx Device/Comms Connections/Tunnel Conn, page 42) setting in the **Tunnel Connection** exists to 'pace' the **out-going data requests** from the driver. This can be used to reduce the rate of data requests in cases where a KNXnet/IP Interface device cannot cope with the traffic volume, caused possibly by its implementation settings or by its operating speed. The **Inter Message Delay** default setting of minimum 15ms has proved successful with **Siemens** interface devices. Any problems arising from this being set too small would manifest as Control Points intermittently having a **Read Fault: Timed out waiting for L_Data_con** fault condition.



Each **out-going data request** actually involves 6 packets travelling between the **KNXnet/IP driver** and the **KNXnet/IP Interface** as follows:

- Request from KNXnet/IP driver to KNXnet/IP Interface
- Acknowledgement from KNXnet/IP Interface to KNXnet/IP driver
- Confirmation from KNXnet/IP Interface to KNXnet/IP driver
- Acknowledgement from KNXnet/IP driver to KNXnet/IP Interface
- Reply from KNXnet/IP Interface to KNXnet/IP driver
- Acknowledgement from KNXnet/IP driver to KNXnet/IP Interface

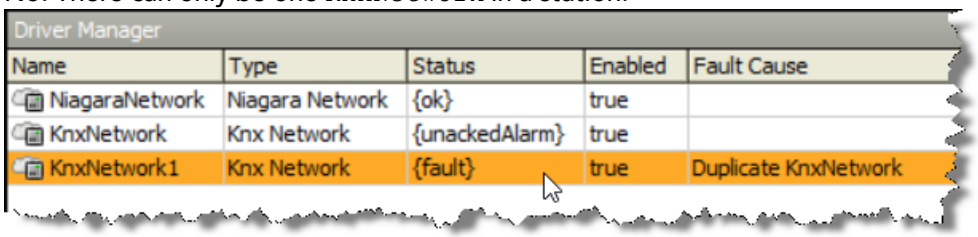
This appears as several flashes on the KNXnet/IP Interface's LEDs.

*NOTE: There is an important difference in the implementation of **Inter Message Delay** between the **KNXnet/IP driver** and the **EIBnet/IP driver**. In the **EIBnet/IP driver** the **Inter Message Delay** was applied between **all** out-going messages, including 'Acknowledgements' and **all** connection control messages. In the **KNXnet/IP driver** the **Inter Message Delay** is **only** applied between **data request messages**.*

KNX network — Duplicate

Can I have more than one KNX network in my Niagara station?

No. There can only be one `KnxNetwork` in a station.

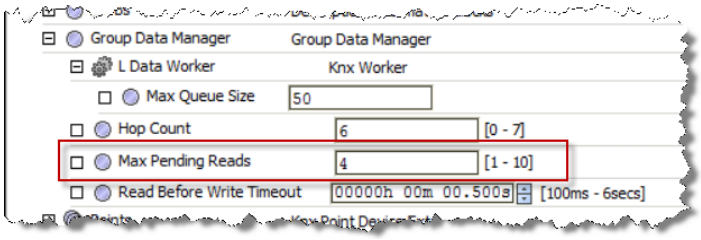


KNX traffic overload protection

How does the driver prevent communications traffic swamping the KNX twisted pair line?

The KNXnet/IP driver prevents the KNX twisted pair line from being swamped with traffic, by controlling the number of concurrently **active** Group Address Read Operations and is set by the **Max Pending Reads** property in the **Group Data Manager** (see Knx Device/Group Data Manager, page 39). The term **active** in this context, means that a particular **Group Address Read Operation** has firstly reached the head of the **Group Data Operation Queue** (see The 'Read' queue is full, page 58) and secondly, the Communications Stack has sent an `L_Data_req` message to the KNXnet/IP device and received an `L_Data_con` reply, but has not yet received a corresponding `L_Data_ind` message.

The default value of **Max Pending Reads** is 4 but unfortunately, there is no clear guidance in the KNX Specs as to what an acceptable **Max Pending Reads** value should be, however having this value too small would **not** cause a The 'Read' queue is full, page 58 fault.



Multicast

Does the KNXnet/IP driver support Multicast?

The term **Multicast** can apply to both **IP Multicast** and the KNX **point-to-multi-point, connection-less (multicast) Transport Layer communication mode**.

In the case of **IP Multicast**, the KNXnet/IP driver does support it in order to facilitate **Device Discovery** on the network.

*NOTE: The **default IP Multicast Address** is* `224.0.23.12`..

With regard to the KNX **point-to-multi-point, connection-less (multicast) Transport Layer communication mode**, the KNXnet/IP driver inherently supports this communication mode in so far as it makes use of the `A_GroupValue_Read-PDU`, `A_GroupValue_Response-PDU` and `A_GroupValue_Write-PDU` **Application Layer services** which are only supported using this communication mode.

Proxy Routing

What is Proxy Routing?

The KNXnet/IP driver is unable to directly support the **KNXnet/IP Routing** Service. **Proxy Routing** is a mechanism we have introduced to provide the following:

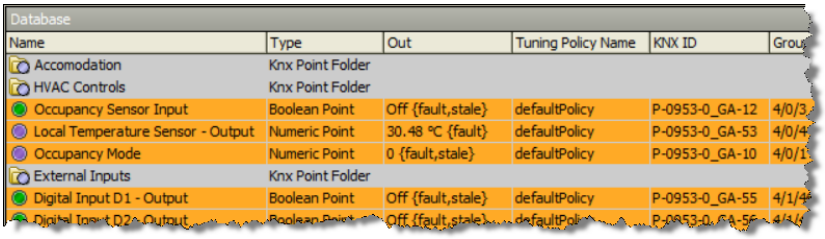
- To enable the KNXnet/IP driver to communicate with **KNXnet/IP Router** devices whose IP subnet differs from the Niagara Host IP subnet.
- To allow the KNXnet/IP driver to communicate with multiple **KNXnet/IP Router** devices without using up a **KNXnet/IP Tunnelling** connection in each **KNXnet/IP Router**.

Proxy Routing relies on configuring a **KNXnet/IP device** in the Niagara Station using a **KNXnet/IP Tunnelling** connection to one **KNXnet/IP Router** device which has the same IP subnet address as the Niagara Host. The **KNXnet/IP Router** device's filtering needs to be configured to allow it to route messages from the **KNXnet/IP Tunnelling** connection to other **KNXnet/IP Router** devices and vice-versa.

The other **KNXnet/IP Router** devices can then be configured as **KNXnet/IP devices** in the Niagara Station with their **Connection Method** property set to **Proxy Routing** and their Proxy Device address and other connection properties set to the **KNXnet/IP device** which was configured to use the **KNXnet/IP Tunnelling** connection described above.

Points in Fault — KNXnet/IP Interface

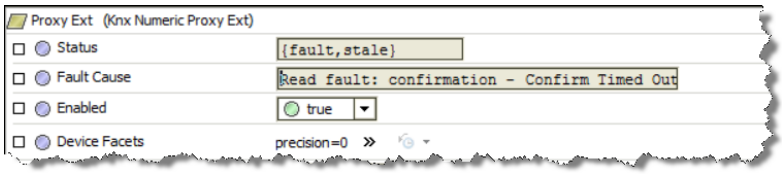
Why, when discovered points are added to the Database, are they in {fault}?



One of the many reasons why points can go to a `{fault}` condition, relates to the **R1** firmware revision of the **Siemens N 148/22** KNXnet/IP Interface device. It may also occur with other KNXnet/IP Interface devices.

The problem is caused by the KNXnet/IP Interface device failing to respond with an **Acknowledge** message although message confirmation has been requested by the KNXnet/IP driver.

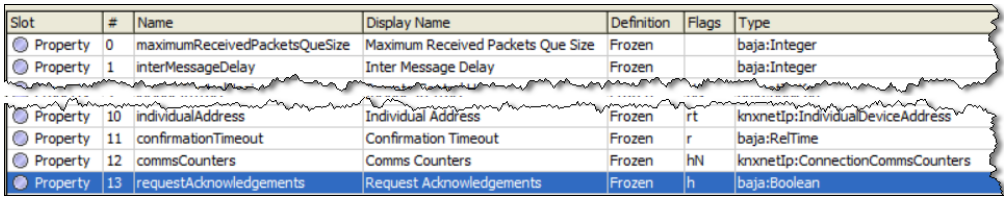
You may also observe that the **Fault Cause** property of the **Point Proxy Extension** indicates `Read fault: confirmation — Confirm Timed Out`.



One way to overcome this problem is to upgrade the **R1** firmware revision level of the **Siemens N 148/22** KNXnet/IP Interface device (or any other KNXnet/IP Interface device) . However, it is beyond the scope of this document to detail the steps to accomplish this but it is believed that manufacturers of KNXnet/IP Interface devices do provide tools and guidance to upgrade their firmware.

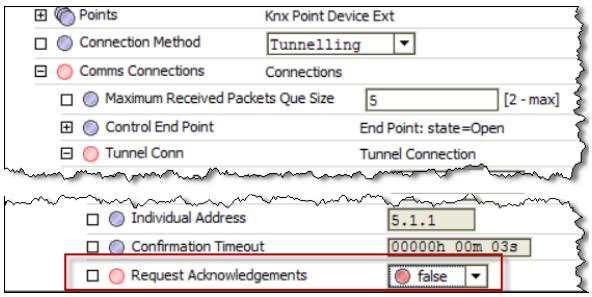
Another way to overcome the problem, is available within the KNXnet/IP driver. To do this, the KNXnet/IP driver must be configured not to request an acknowledgement to its messages. This can be configured by using a `hidden` property in the **Tunnel Connection** of the KNXnet/IP driver. (see Knx Device/Comms Connections/Tunnel Conn, page 42).

Unhide the **requestAcknowledgements** slot of the **Knx Network / Knx Device / Comms Connections / Tunnel Conn**.



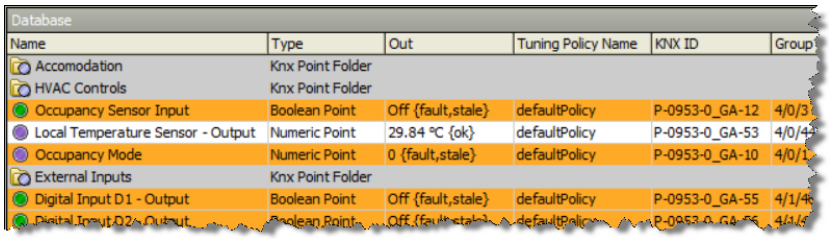
Slot	#	Name	Display Name	Definition	Flags	Type
Property	0	maximumReceivedPacketsQueSize	Maximum Received Packets Que Size	Frozen		baja:Integer
Property	1	interMessageDelay	Inter Message Delay	Frozen		baja:Integer
Property	10	individualAddress	Individual Address	Frozen	rt	knxnetIp:IndividualDeviceAddress
Property	11	confirmationTimeout	Confirmation Timeout	Frozen	r	baja:RelTime
Property	12	commsCounters	Comms Counters	Frozen	hN	knxnetIp:ConnectionCommsCounters
Property	13	requestAcknowledgements	Request Acknowledgements	Frozen	h	baja:Boolean

Change the **Request Acknowledgements** property to `false`.



Points in Fault — Unsolicited

Why do some points randomly change state between {stale} and/or {fault} and {ok}?



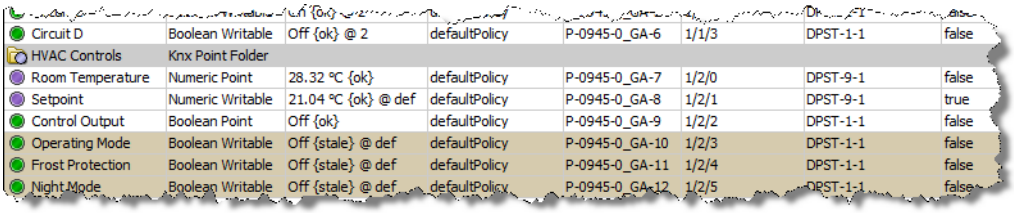
Name	Type	Out	Tuning Policy Name	KNX ID	Group
Accomodation	Knx Point Folder				
HVAC Controls	Knx Point Folder				
Occupancy Sensor Input	Boolean Point	Off {fault,stale}	defaultPolicy	P-0953-0_GA-12	4/0/3
Local Temperature Sensor - Output	Numeric Point	29.84 °C {ok}	defaultPolicy	P-0953-0_GA-53	4/0/4
Occupancy Mode	Numeric Point	0 {fault,stale}	defaultPolicy	P-0953-0_GA-10	4/0/1
External Inputs	Knx Point Folder				
Digital Input D1 - Output	Boolean Point	Off {fault,stale}	defaultPolicy	P-0953-0_GA-55	4/1/4

If after appropriate time-outs, the KNXnet/IP driver, fails to receive requested data from the KNX device, points will go to a `{fault}` and/or `{stale}` state.

If the KNX device subsequently transmits data, for example a change of state or value, then this unsolicited message is received by the KNXnet/IP driver thereby causing a change of proxy point status.

Stale points — KNX project

Why, when discovered points are added to the Database, some are {stale} and some are not?



Circuit D	Boolean Writable	Off {ok} @ 2	defaultPolicy	P-0945-0_GA-6	1/1/3	DPST-1-1	false
HVAC Controls	Knx Point Folder						
Room Temperature	Numeric Point	28.32 °C {ok}	defaultPolicy	P-0945-0_GA-7	1/2/0	DPST-9-1	false
Setpoint	Numeric Writable	21.04 °C {ok} @ def	defaultPolicy	P-0945-0_GA-8	1/2/1	DPST-9-1	true
Control Output	Boolean Point	Off {ok}	defaultPolicy	P-0945-0_GA-9	1/2/2	DPST-1-1	false
Operating Mode	Boolean Writable	Off {stale} @ def	defaultPolicy	P-0945-0_GA-10	1/2/3	DPST-1-1	false
Frost Protection	Boolean Writable	Off {stale} @ def	defaultPolicy	P-0945-0_GA-11	1/2/4	DPST-1-1	false
Night Mode	Boolean Writable	Off {stale} @ def	defaultPolicy	P-0945-0_GA-12	1/2/5	DPST-1-1	false

One of the many reasons why points can go to a `{stale}` state relates to the ETS project file.

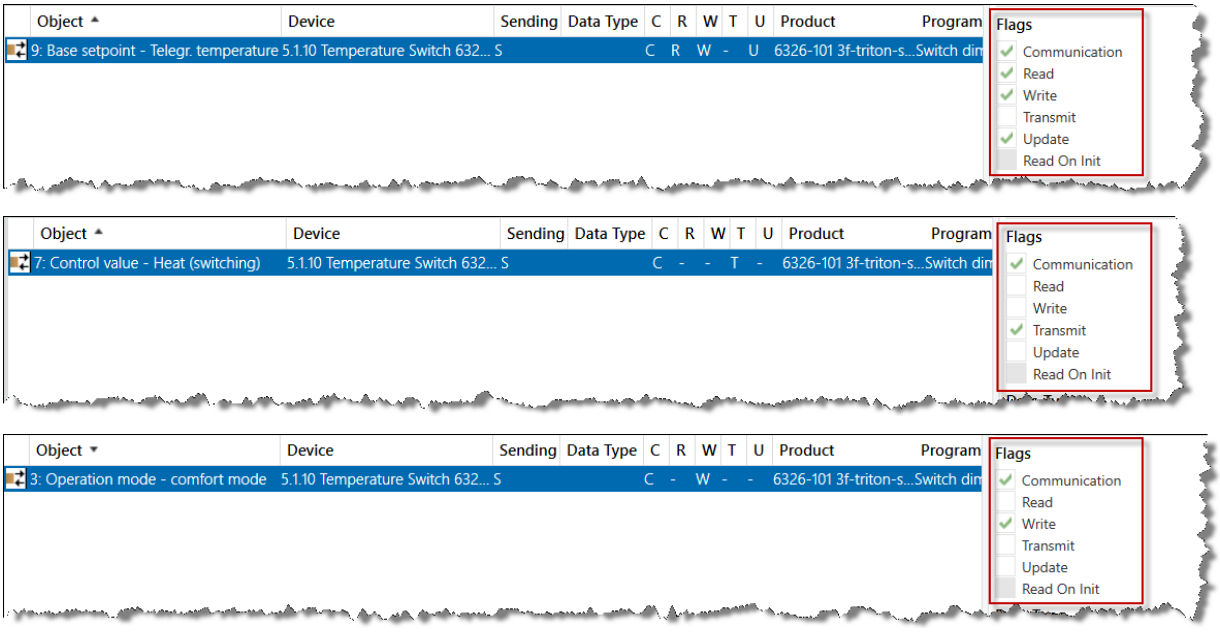
When points are added to the Database from a ETS project file, their **Subscription** properties are automatically configured according to which **Communications Flags** are set in the ETS project, for the KNX Device's **Group Objects** which are associated with the point's **Group Address**.

If a **Group Address** has at least one KNX **Device Group Object** associated with it that has its **Read Communication Flag** property enabled then the **Poll once on subscribed**, **Poll once on operational** and **Poll until answer after poll once** properties are set to `true`. This will trigger an immediate poll of the **Group Address** (because this view is itself subscribed to the point).

If a **Group Address** has no associated KNX **Device Group Objects** that have their **Read Communication Flag** enabled then its **Subscription** properties will all be set to `false` by default. The value will however be updated (and become not `{stale}`) in the KNXnet/IP driver when any subsequent unsolicited messages regarding the **Group Address** are received from the KNX system.

Here are the **Communications Flags** settings for some of the points in the example shown above:

*NOTE: The **Control Output (Control value)** object has its `Transmit` flag set. This will cause an unsolicited message transmission from the KNX device.*



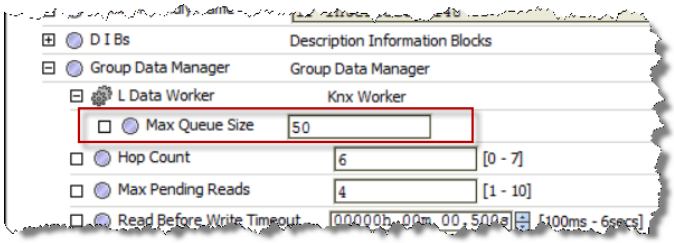
The 'Read' queue is full

Please explain the Read fault: The 'Read' queue is full?

This fault occurs whenever an attempt is made to enqueue (add an item to a queue) a read request of a particular **GroupAddress**, where the queue is already full and the **GroupAddress** in question is not already in the queue. The queue in question is the **Group Data Operation Queue** (see Knx Device/Group Data Manager, page 39) which holds a list of **Group Data Operations** (GroupAddress reads or writes) waiting to be started as soon as the communications stack is able.

The **Group Data Operation Queue** has its size exposed under the hidden **L Data Worker** child of the **Group Data Manager**, immediately preceding the **Hop Count** property and can be seen in the example below (see Knx Device/Group Data Manager/L Data Worker, page 40). The default value of the **Group Data Operation Queue** is 50.

Subject to how many **Group Addresses** are being polled and how often and the amount of available RAM, you could try increasing the value to 1000 to overcome the fault.



UDP ports

Is the IP port adjustable?

This question is sometimes asked by IT departments who need control over the IP port usage on their network.

Protocol:

The KNX System specifications specifies that **UDP** (User Datagram Protocol) is used for KNXnet/IP connections rather than **TCP** (Transmission Control Protocol) and therefore the following commentary on **Ports** and **Messaging** is in relation to **UDP**.

Device and Driver Ports:

There are four logical **UDP ports** allocated for each tunnel connection between the KNXnet/IP driver and a KNXnet/IP Interface device. Two ports (one for Control and one for Data) are at the device and two ports are at the driver (Control and Data). The **Control port** is used for such activities such as 'Opening' and 'Closing' the connection and for 'Heart-Beat'. The **Data port** is used for Point Data.

Port numbers:

It is usual that **KNXnet/IP Interface devices** use port **3671** for both Control and Data but this may not always be the case. In the **KNXnet/IP driver**, the two ports it uses to communicate with the device are allocated *dynamically* when establishing and maintaining a connection. In this way, the driver can separate the messaging between multiple devices concurrently by allocating different port numbers. In the driver the Control and Data ports for each device will *always* be different.

Port number selection:

The port numbers used by the **KNXnet/IP Interface device** will be set by the ETS tool (providing that the device is *able* to be configured). The port numbers which are chosen dynamically by the **KNXnet/IP driver** for each **Local Interface** will be in a range between a port-minimum (0-65535) and port-maximum (0-65535). The default values of these are **3500** and **4000** although they can be changed during setup of the driver. (See Knx Network/Local Interface, page 34). Clearly if the range is reduced then this will start to restrict the

number of potential device connections and ultimately if the range offers only one port number, the driver will not function because two ports is a *minimum* for each device connection.

Recycling port numbers:

When dynamically choosing a new port number, the **KNXnet/IP driver** will choose the next available port number by starting at the port-minimum (default setting) or by cycling through the range, this choice of behavior being selectable during setup of the driver. (See Knx Network/Local Interface, page 34).

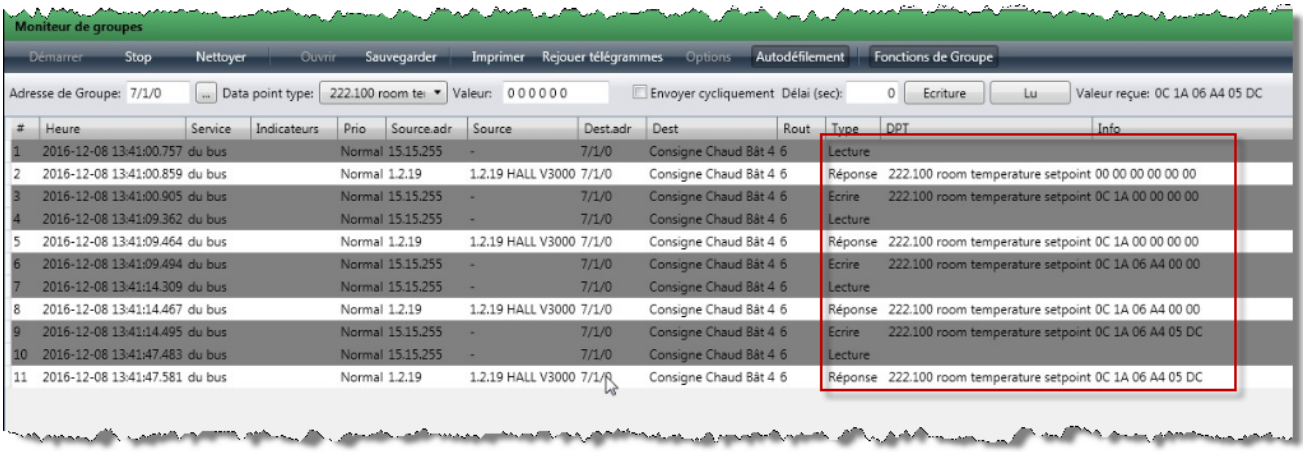
Writing to Compound Structures

How does the KNXnet/IP driver write to a KNX 'Compound Structure' DPT?

The KNX standard specifies that some Datapoint Types (DPT) have **Compound Structures** where several data fields are contained within one DPT.

Datapoint Type 222.100 for example, is a DPT with 3 sets of Room Temperature Setpoints, each of which is a 16 bit-Float Value. The room temperature setpoint comfort (TempSetpComf), room temperature setpoint standby (TempSetpStdby) and room temperature setpoint economy (TempSetpEco) are all encoded within the same DPT.

From the standpoint of the Niagara driver data model, each of these fields is a separate proxy point but on the wire, KNX compounds them into one Group Address. When it writes to the group address the KNXnet/IP driver must therefore, first know the current value of every field before overwriting the DPT. To do this, the driver **reads** the whole Group Address before individually overwriting a data field. This behavior can be seen in the following ETS Group Monitor diagnostic of Group Address 7/1/0 which is a 222.100 DPT:



- Line 1: Read Group Address
- Line 2: Response Data `00 00 00 00 00 00`
- Line 3: Write TempSetpComf Data `0C 1A 00 00 00 00` (Decimal 3098)
- Line 4: Read Group Address
- Line 5: Response Data `0C 1A 00 00 00 00`
- Line 6: Write TempSetpStdby Data `0C 1A 06 A4 00 00` (Decimal 1700)
- Line 7: Read Group Address
- Line 8: Response Data `0C 1A 06 A4 00 00`
- Line 9: Write TempSetpEco Data `0C 1A 06 A4 05 DC` (Decimal 1500)
- Line 10: Read Group Address
- Line 11: Response Data `0C 1A 06 A4 05 DC`

Glossary

BatiBUS	BatiBUS was an open field bus for home appliances control and communication. Now, after merging with two other protocols, it is part of the KNX standard
BMS	Building Management System
EHS	The European Home Systems (EHS) protocol was aimed at home appliances control and communication. Now, after merging with two other protocols, it is part of the KNX standard
EIB	The European Installation Bus or Instabus is a decentralised open system to manage and control electrical devices within a facility. Now, after merging with two other protocols, it is part of the KNX standard
ETS	The Engineering Tool Software (ETS) is a PC software tool which enables the design, engineering and configuration of installations based on KNX certified products. The tool, which is manufacturer independent, enables a system integrator to combine products from different manufacturers into one solution
ETS Project	An ETS Project consists of KNX devices and the links between them. It may also contain KNX catalog data. ETS Projects are managed and maintained by the ETS software tool
KNX	KNX is a worldwide Standard for control in both commercial and residential buildings
KNX device	KNX devices are KNX system components that are connected together by a two wire bus allowing them to exchange data. Some typical types of KNX devices are Sensors and Actuators.
KNX Installation	A KNX Installation comprises KNX devices which are accessible through a KNX IP device
KNXnet/IP driver	This is the Niagara KNXnet/IP driver supporting the KNX standard.
KNXnet/IP Routing	KNXnet/IP Routing is a multicast-based telegram, which allows a KNX IP router to perform the function of a line or area coupler.
KNXnet/IP Tunnelling	KNXnet/IP Tunnelling is the primary method of interfacing to a KNX system and allows for a point-to-point communication (unicast) from a single external device to the KNX system. This is akin to using a USB or Serial Interface.
OSI	The Open Systems Interconnection model (OSI model) is a conceptual model that standardizes the communications functions of a telecommunication or computer system without regard to their underlying internal structure and technology. Its goal is the interoperability of diverse communication systems with standard protocols

Index

A

C

D

F

G

K

M

P

Q

R

S

W