

PROFINET IO Connector V2.1.0

Operating Manual

General Data Protection Regulation (GDPR)	1
Cybersecurity information	2
Security Information for Industrial Edge App	3
Introduction to PROFINET IO Connector	4
Getting Started with PROFINET IO Connector	5
Installing PROFINET IO Connector	6
Configuring PROFINET IO Connector	7
Working with PROFINET IO Connector	8
Troubleshooting	9
Changes	10

Legal information

Warning notice system

This manual contains notices you have to observe in order to ensure your personal safety, as well as to prevent damage to property. The notices referring to your personal safety are highlighted in the manual by a safety alert symbol, notices referring only to property damage have no safety alert symbol. These notices shown below are graded according to the degree of danger.

DANGER

indicates that death or severe personal injury **will** result if proper precautions are not taken.

WARNING

indicates that death or severe personal injury **may** result if proper precautions are not taken.

CAUTION

indicates that minor personal injury can result if proper precautions are not taken.

NOTICE

indicates that property damage can result if proper precautions are not taken.

If more than one degree of danger is present, the warning notice representing the highest degree of danger will be used. A notice warning of injury to persons with a safety alert symbol may also include a warning relating to property damage.

Qualified Personnel

The product/system described in this documentation may be operated only by **personnel qualified** for the specific task in accordance with the relevant documentation, in particular its warning notices and safety instructions. Qualified personnel are those who, based on their training and experience, are capable of identifying risks and avoiding potential hazards when working with these products/systems.

Proper use of Siemens products

Note the following:

WARNING

Siemens products may only be used for the applications described in the catalog and in the relevant technical documentation. If products and components from other manufacturers are used, these must be recommended or approved by Siemens. Proper transport, storage, installation, assembly, commissioning, operation and maintenance are required to ensure that the products operate safely and without any problems. The permissible ambient conditions must be complied with. The information in the relevant documentation must be observed.

Trademarks

All names identified by ® are registered trademarks of Siemens AG. The remaining trademarks in this publication may be trademarks whose use by third parties for their own purposes could violate the rights of the owner.

Disclaimer of Liability

We have reviewed the contents of this publication to ensure consistency with the hardware and software described. Since variance cannot be precluded entirely, we cannot guarantee full consistency. However, the information in this publication is reviewed regularly and any necessary corrections are included in subsequent editions.

Table of contents

1	General Data Protection Regulation (GDPR)	5
2	Cybersecurity information	7
3	Security Information for Industrial Edge App	9
4	Introduction to PROFINET IO Connector	11
4.1	Overview	12
4.2	Data Flows	14
4.2.1	PROFINET IO Connector and IIH Essentials	14
4.2.2	PROFINET IO Connector as PROFINET interface for Custom App	14
4.3	Quantity Structure	16
5	Getting Started with PROFINET IO Connector	21
6	Installing PROFINET IO Connector	25
6.1	Prerequisites	25
6.2	System Requirements	26
6.3	Overview of the installation process	30
6.4	Buying an app	31
6.5	Copy Apps to IEM Catalog	36
6.6	Install PROFINET IO Connector Application	37
7	Configuring PROFINET IO Connector	41
7.1	Configuration using four files	43
7.2	Configuration of Databus and IIH Essentials apps	51
7.2.1	Configuration of Databus	51
7.2.2	Configuration of IIH Essentials	53
7.3	PROFINET Configuration in TIA Portal	55
7.3.1	Overview	55
7.3.2	Configuring PROFINET Driver as Controller	55
7.3.3	Configuring PROFINET Driver as Device	62
7.4	Configuring PROFINET IO Connector using JSON file	66
7.4.1	Configuring PROFINET IO Connector as PN controller	66
7.4.1.1	Introduction	66
7.4.1.2	Downloading the JSON file	67
7.4.1.3	Example of JSON Configuration for Controller	67
7.4.1.4	Configure PROFINET IO Connector using JSON file	68
7.4.2	Configuring PROFINET IO Connector as PN Device	70
7.4.2.1	Introduction	70
7.4.2.2	Example of JSON Configuration for Device	71
7.5	Configuring PROFINET IO Connector using Common Configurator	72

7.5.1	Introduction to Common Configurator	72
7.5.2	Connecting the Common Configurator to Databus	74
7.5.3	Configuring PROFINET IO Connector as a PN Controller	76
7.5.3.1	Overview	76
7.5.3.2	Configuring data sources.....	76
7.5.3.3	Configuring tags	80
7.5.4	Configuring PROFINET IO Connector as a PN Device	82
7.5.4.1	Overview	82
7.5.4.2	Configuring data sources.....	82
8	Working with PROFINET IO Connector.....	87
8.1	Cyclic Read PROFINET IO Data.....	88
8.1.1	MQTT Payload for Cyclic Read PROFINET IO Data.....	88
8.1.2	PROFINET IO Read - Metadata.....	88
8.1.3	PROFINET IO Read - Binary Payload	92
8.1.4	PROFINET IO Read - JSON Payload.....	95
8.1.4.1	JSON Payload (no tags defined, no oversampling).....	96
8.1.4.2	JSON Payload (no tags defined, oversampling activated).....	96
8.1.4.3	JSON Payload (with tags, no oversampling)	97
8.1.4.4	JSON Payload (with tags, with oversampling)	98
8.2	PROFINET IO Write	99
8.3	PROFINET Data Record Write.....	101
8.4	PROFINET Data Record Read	105
9	Troubleshooting	109
9.1	Application log files.....	110
9.2	Application statistics	112
9.3	Tips and tricks - Typical issues.....	116
9.4	Configuring Layer-2-Access.....	121
9.4.1	Introduction.....	121
9.4.2	How to configure Layer-2-Access?.....	121
9.4.3	PROFINET Addresses	124
9.5	Diagnostic features by version.....	125
10	Changes	129
10.1	Breaking changes.....	129

General Data Protection Regulation (GDPR)

Siemens adheres to the principles of data protection, in particular the principles of data minimization (Privacy by Design).

For this product, PROFINET IO Connector, this means:

Personal data

There is no personal data* collected but following data is stored to allow machine to machine communication:

- Databus credentials
- Tags data and metadata from field devices
- Timestamp
- Smart device information and app usage data

If the customer links the data mentioned above to other data (e.g. shift plans) or if the customer saves personal information on the same medium (e.g. hard disk) and thus creates a personal reference, the customer has to ensure that the guidelines regarding data protection are observed.

Note

* This section refers to any personal data processed by the Application other than the personal data contained in log-files / tracking data (if any). "Personal data" are any information relating to an identified or identifiable natural person. Please note that IP-addresses, device identifiers such as IMEI, UDID, IMSI, MAC-address, MSISDN, location data or machine data (if machine data tracks events triggered by user interaction with the machine) usually qualify as personal data.

Purposes

The data mentioned above is required for the following purposes:

- Access protection and security measures
- Message system for traceability and availability
- For app diagnosis

Storage of the data is affected for a suitable purpose and is limited to what is strictly necessary, as the information is indispensable in order to identify the authorized operators.

Securing of data

The above data will not be stored anonymously or pseudonymized, as the purpose (identification of the operating personnel) cannot be achieved otherwise.

The above data will be used only within the product and within the Edge eco-system and will not be automatically passed on to third parties or unauthorized persons.

The above data is secured by adequate technical measures, such as storing and encryption of process data in databases.

The tags data and metadata from field devices data will be used only within the product and will not be automatically passed on to third parties or unauthorized persons.

The customer must ensure the access protection as part of his process configuration.

Deletion policy

This product does not provide an automatic deletion for the databus or PLC credentials already provided by the user. In case the user provides a different databus or PLC credentials, the previous credentials will be overwritten. Since, there is no explicit delete option, the user could provide invalid databus or PLC credentials to delete the existing valid credentials.

In short, the collected log data will be automatically deleted once the limit is reached (oldest entries first).

Data configuration

The customer can configure the data collected via the product as follows:

- Using the App Configuration files
- Using Common Configurator

Cybersecurity information

Siemens provides products and solutions with industrial cybersecurity functions that support the secure operation of plants, systems, machines and networks.

In order to protect plants, systems, machines and networks against cyber threats, it is necessary to implement – and continuously maintain – a holistic, state-of-the-art industrial cybersecurity concept. Siemens' products and solutions constitute one element of such a concept.

Customers are responsible for preventing unauthorized access to their plants, systems, machines and networks. Such systems, machines and components should only be connected to an enterprise network or the internet if and to the extent such a connection is necessary and only when appropriate security measures (e.g. firewalls and/or network segmentation) are in place.

For additional information on industrial cybersecurity measures that may be implemented, please visit

<https://www.siemens.com/global/en/products/automation/topic-areas/industrial-cybersecurity.html>.

Siemens' products and solutions undergo continuous development to make them more secure. Siemens strongly recommends that product updates are applied as soon as they are available and that the latest product versions are used. Use of product versions that are no longer supported, and failure to apply the latest updates may increase customer's exposure to cyber threats.

To stay informed about product updates, subscribe to the Siemens Industrial Cybersecurity RSS Feed under

<https://new.siemens.com/global/en/products/services/cert.html>.

Security Information for Industrial Edge App

Security information (assumption/constraint) for Industrial Edge Apps are as follows:

- Only authorized internal operators will have access to Industrial Edge Device with-in secure network using VPN connection.
- Perimeter firewall configuration responsibility lies with end customer.
- Security guidelines for usage of USB sticks within shop floor are applied.
- Creating users with appropriate access rights needs to be done during commissioning and it is the responsibility of the operator.
- Customer is responsible for configuring the application as per the installation/user manual, based on system requirements and technical capabilities of app documented so that the Automation System performance is not impacted.
- The system is installed in an environment that ensures physical access is limited to authorized maintenance personnel only. Managing unauthorized attachment of removable devices is the responsibility of the operator.
- The platform including hardware, firmware and operating system is securely configured and maintained by the operator.
- The operator is capable of protecting the environment from malware infection.
- Centralized IT security components (Active Directory, Centralized IT Logging Server) are provided and well secured by the operator and can be trusted.
- The operator personnel accessing the system is well trained in the usage of the system and general information security aspects like password handling, removable media, etc. are in place.
- Operator is responsible for the Confidentiality, Integrity, and Availability (CIA) of data stored outside the Industrial Edge Device.
- Operator is responsible for configuring the PLCs with appropriate read/write access levels (Legitimization) and configure Industrial Edge Apps with appropriate passwords for data collection from PLC's.
- Customer takes care about time sync of Industrial Edge Management and Industrial Edge Device.
- For S7+ browsing, from PLC firmware V2.9 and greater the onus lies on the user while trusting the PLC server certificate.

Introduction to PROFINET IO Connector

The PROFINET IO Connector is an application for Industrial Edge Management. This app implements a PROFINET Controller which cyclically reads the PROFINET IO (PN IO) data of the configured PROFINET network. This means, you can directly connect PROFINET devices to the Industrial Edge Device (IED). This data is published to the Databus application (MQTT broker).

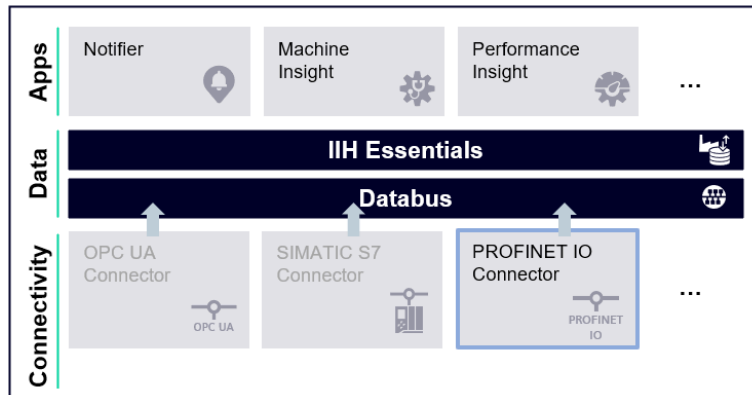
The PROFINET IO Connector app also provides other PROFINET functionality, for example, writing IO-Data and acyclic services (PROFINET Data Records).

4.1 Overview

The PROFINET IO Connector enables the following use cases for Industrial Edge Management:

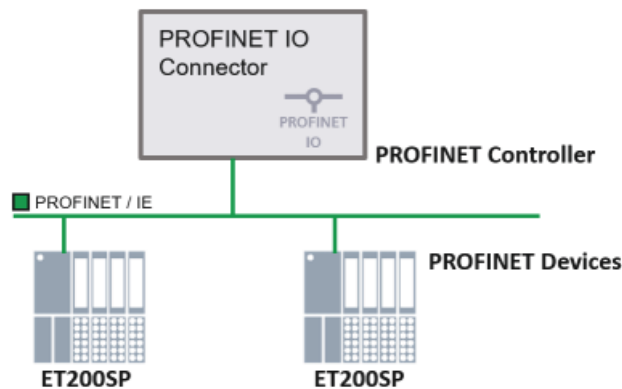
PROFINET IO Connector for IIH Essentials

The IIH Essentials is an important central component of the Edge System. It offers a REST API to apps to access information (historic data). The IIH Essentials receives the data from various connectors, for example, PROFINET IO Connector.

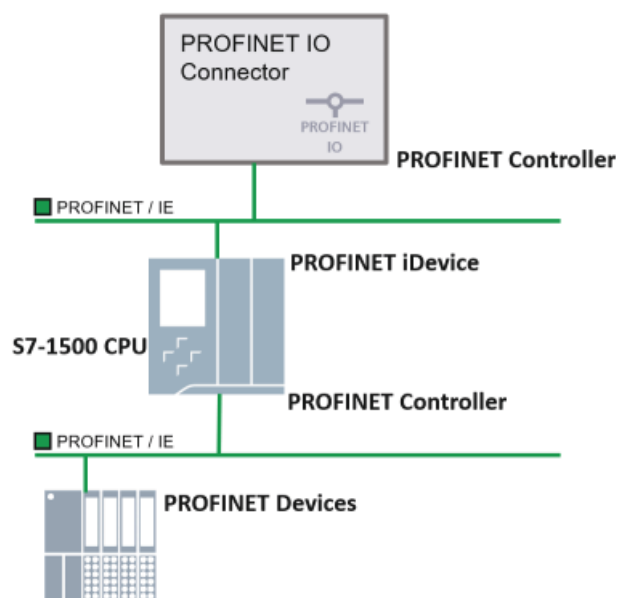


PROFINET IO Connector as PROFINET Controller

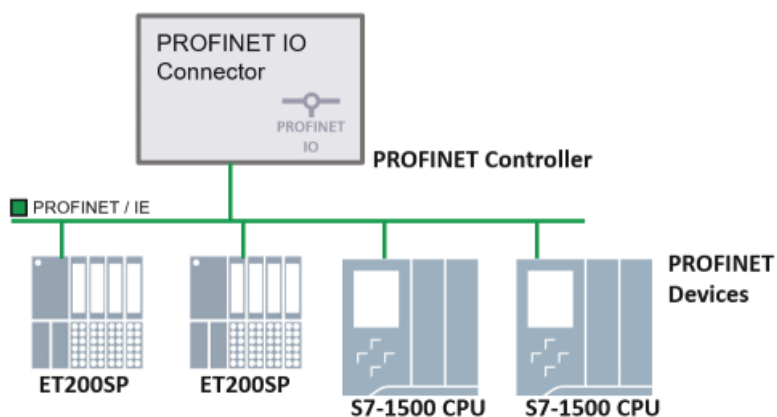
The PROFINET IO Connector operates as PROFINET Controller. Therefore, you can directly connect PROFINET devices to the IED. For example, you can directly connect PROFINET IO devices (ET200SP) to IED.



When you want to connect an existing SIMATIC CPU, you must configure the CPU as PROFINET iDevice. In this way, you can record data from this CPU as well.



You can combine and connect both PROFINET devices and PLCs as PROFINET iDevices.

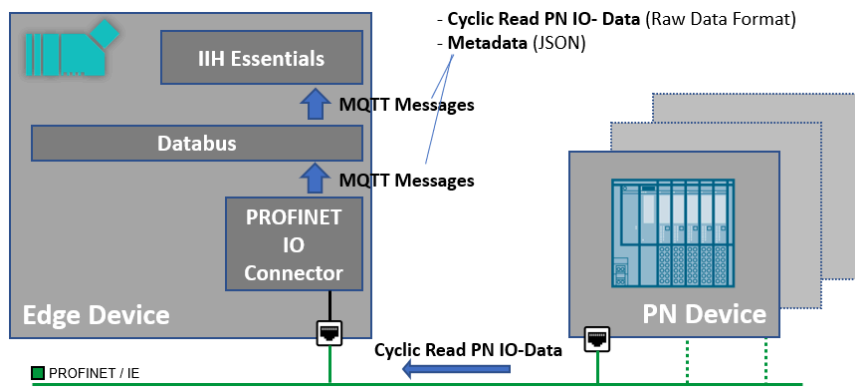


4.2 Data Flows

Based on the use cases described in previous section Overview (Page 12), the data flow and functionality of PROFINET IO Connector may differ.

4.2.1 PROFINET IO Connector and IIH Essentials

The common use case uses the PROFINET IO Connector as data source for IIH Essentials. IIH Essentials is the interface for the data processing. The following image depicts the principle:

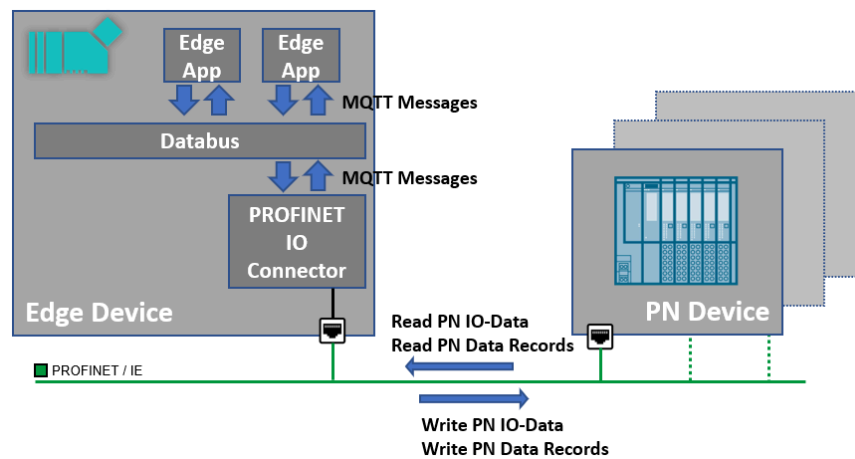


4.2.2 PROFINET IO Connector as PROFINET interface for Custom App

All the communications of the PROFINET IO Connector are executed through the Databus. Custom application can use this interface as well. In this way, the full functional range of the PROFINET IO Connector can be used.

The PROFINET IO Connector offers different main functionalities as follows:

- High-speed recording of PROFINET IO-data (cyclic read)
 - Raw data format (binary payload): smallest overhead for best performance
 - JSON payload: easier handling at client side
- Write PROFINET IO-data
- PROFINET Data Record Write
- PROFINET Data Record Read



To offer these functions, the app publishes various MQTT topics as follows:

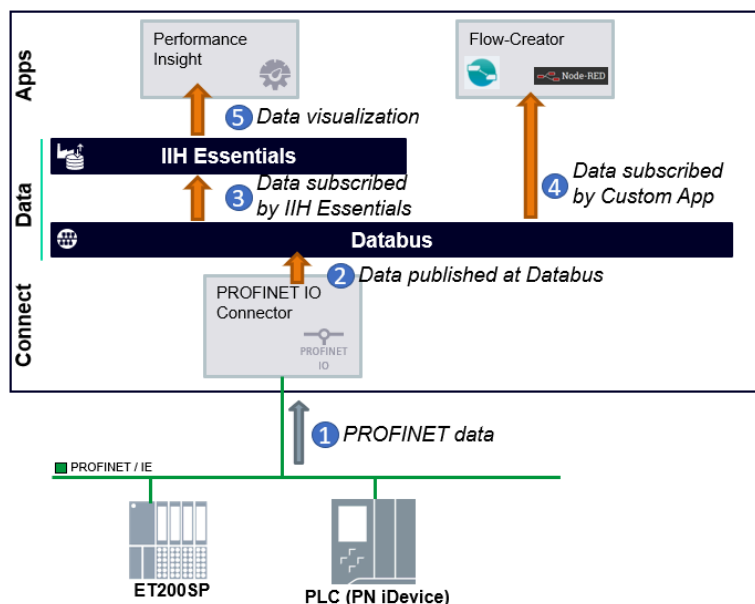
- The cyclic IO data in a RAW Data Format (binary payload) for best performance
- The metadata to describe the structure of the Raw Data as JSON MQTT message
- The cyclic IO data as JSON (optional)
- The Data Record Read response

The app subscribes several topics as follows:

- Write PROFINET IO-data
- PROFINET Data Record Read Request
- PROFINET Data Record Write Request

4.3 Quantity Structure

When using the PROFINET IO Connector you must consider quantity structures at different levels. The following image depicts the relevant data flows:



Limits in PROFINET Communication

The limit is defined by the possible PROFINET traffic. The PROFINET IO Connector bases on the PROFINET Driver for controller in the variant Linux Native Ethernet Interface. Therefore, you must cover the limits of this software driver:

Feature	Theoretical Limits	Successfully Tested
Number of PN Devices	maximum 128	12
Number of Ethernet Frames per Millisecond	maximum 13 frames/millisecond	12
IO memory per PN device	maximum 1.024 byte	20
IO memory in total	maximum 8.192 byte	240

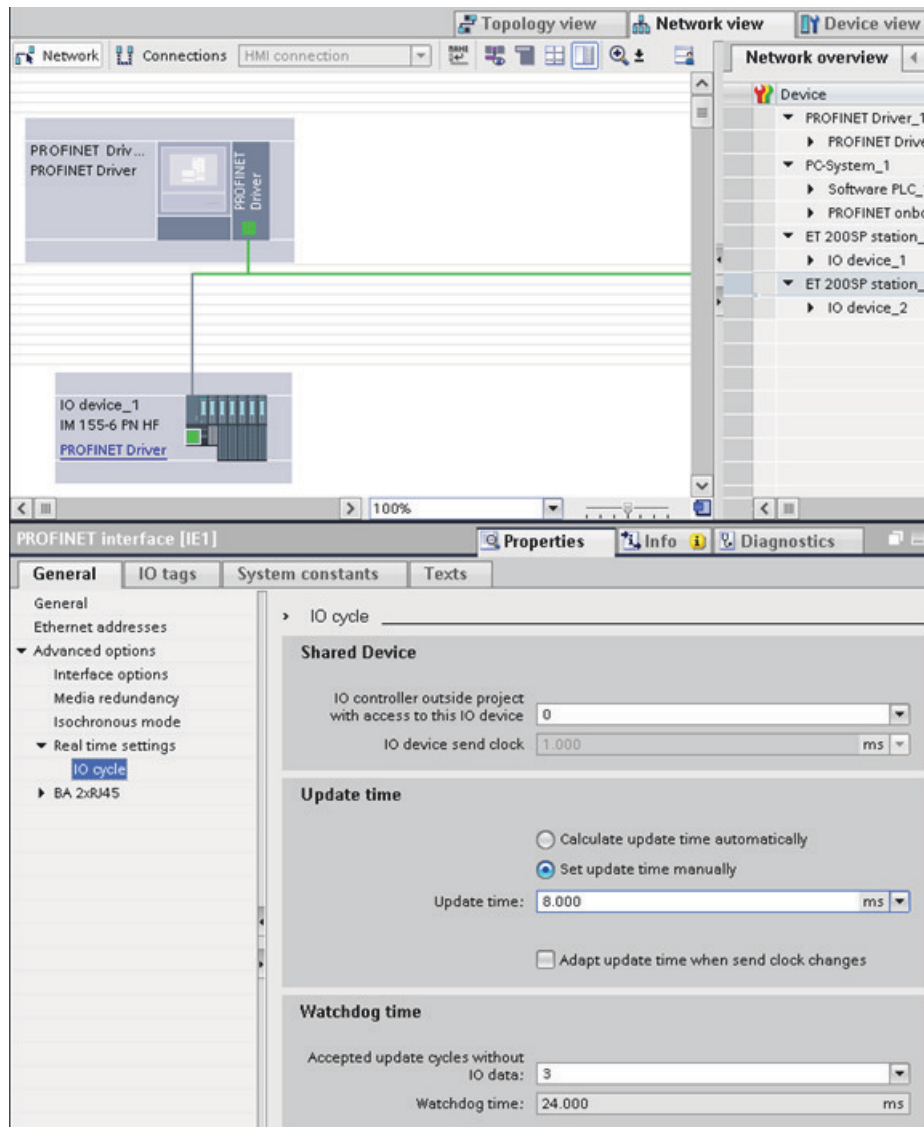
Note

The PROFINET IO Connector as IO Controller is also successfully tested with 20 connected PN devices.

Note

When the system load is very high, for example, when another Industrial Edge App is started at the IED, the PROFINET communication might be interrupted for a short period of time. The connection is established again automatically. But the PROFINET IO Connector cannot provide any data during this time.

It is advisable to configure the PROFINET bus cycle in TIA Portal as big as possible for your application. You can increase the update time for PN Devices in TIA Portal as follows:



Limits in Publishing to Databus

Databus is a shared service. Therefore, the limit depends on the overall system:

- Number of publishers and amount of published data in sum
- Number of subscribers and amount of subscribed data in sum

Average Publish Interval	MQTT Traffic in Sum	Test Result
100 milliseconds	1.2 MB/second	✓ Ok
100 milliseconds	2.4 MB/second	X Failed

4.3 Quantity Structure

Note

To reduce the amount of data on the databus, you can use the binary payload format instead of the JSON payload format. Binary format reduces the payload approximately by factor 10 or more.

To know how large a payload message is, you can look in the application log. The relevant log line looks like this:

PN Read: size 1st cyclic binary payload is '26040' bytes

This size with the configured publishing interval leads to the total traffic for MQTT. The formula is as follows:

$\text{MQTT traffic} = (\text{size.of.payload}) * (1 / (\text{publishing interval}))$

For example, the 26.040 bytes payload size and 100 milliseconds publish interval gives:

$26040 \text{ Byte} * 1/0.1 \text{ sec} \approx 260\text{KB/sec}$

Limits with IIH Essentials**Test Environment**

- IED: IPC 227E, Celeron 2930(4C/4T), 8GB, 240 GB SSD
- Binary Publish of PNIO Read Data
- PROFINET Configuration
 - 9 stations (PN iDevice), 50 Byte IO Input memory each

Test Cases with IIH Essentials

Recording Cycle	Number of Tags Defined (tag definition)	Oversampling Factor	Resulting MQTT Payload Size*	Resulting MQTT Publishing Interval	Number of Archived Tags in IIH Essentials	IIH Essentials CPU Load	Test Result
1 millisecond	50	500	234.552 bytes	500 milliseconds	10	30%-50%	✓ Ok
5 milliseconds	60	100	46.952 bytes	500 milliseconds	50	60%-80%	✓ Ok
5 milliseconds	60	100	46.952 bytes	500 milliseconds	0**	5%-10%	✓ Ok
5 milliseconds	90	500	234.552 bytes	2.5 seconds	0**	5%-10%	✓ Ok

* When using the binary publishing format, the size of the payload only depends on the size of the PROFINET IO memory and on the oversampling factor. The number of tags is not relevant, when publishing binary.

** In this test case, no tags were stored in IIH Essentials.

Limits in Processing in Custom App

The limit is about the capability of the current version of the Databus to process the MQTT messages and the custom app to subscribe and to process data provided by the PROFINET IO Connector. A common approach is to subscribe JSON payload with the PN IO Data Read.

Defined Tags	Recording Cycle	Oversampling	Publishing Interval	Tags per Second	Test Result
70	1 millisecond	100	100 milliseconds	70.000	X System Overload
70	2 milliseconds	100	200 milliseconds	35.000	✓ Ok

Shorter publishing intervals mean a higher system load. If your application allows, you should use publishing intervals above 200 milliseconds.

Note

For limits regarding the message size, refer PROFINET IO Read - Binary Payload (Page 92) or PROFINET IO Read - JSON Payload (Page 95) depend on the configuration (binary or json).

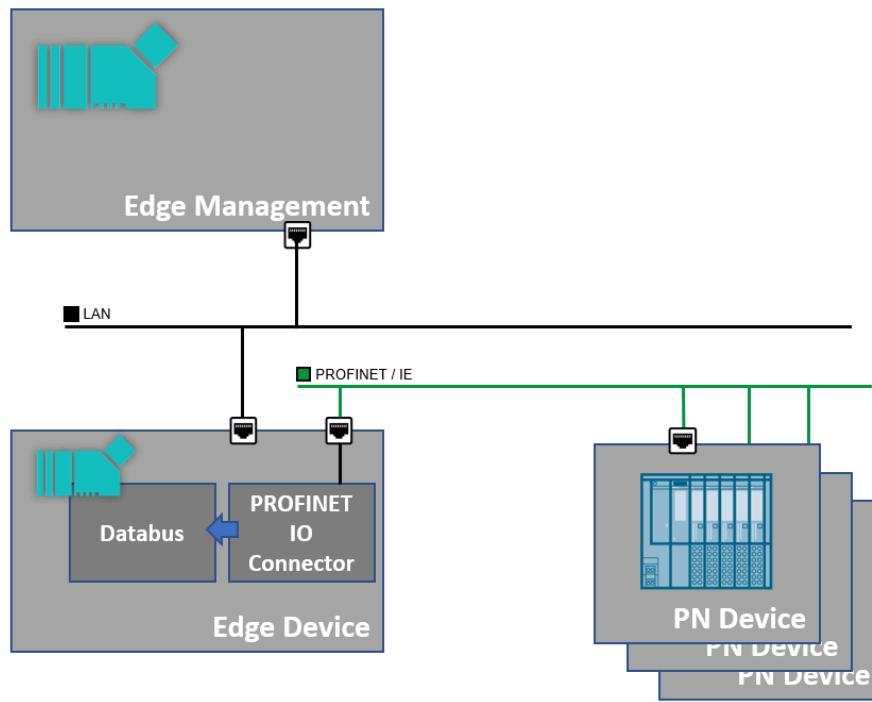
Getting Started with PROFINET IO Connector

To set up PROFINET IO Connector and bring into operation, you must execute the following steps:

1. Configure IED with Layer-2-Access as described in System Requirements (Page 26).
2. Install PROFINET IO Connector app on IED as described in Install PROFINET IO Connector Application (Page 37).
3. Configure PROFINET network with TIA Portal. PROFINET IO Connector must be the PROFINET Controller. Refer, Configuring PROFINET IO Connector (Page 41).
4. Configure PROFINET IO Connector as required. For example, cycle times, oversampling, and so on. Refer, Configuring PROFINET IO Connector (Page 41).
5. Create the tag definition file. It is mandatory when you use PROFINET IO Connector with the IIH Essentials. It is optional when you use PROFINET IO Connector as PROFINET interface for custom app. Refer, Configuring PROFINET IO Connector (Page 41).
6. Download the configuration files (PN config, app config, tag-definition, and user credentials) on the IED. You can download the sample configuration files from here (<https://support.industry.siemens.com/cs/document/109793251>).

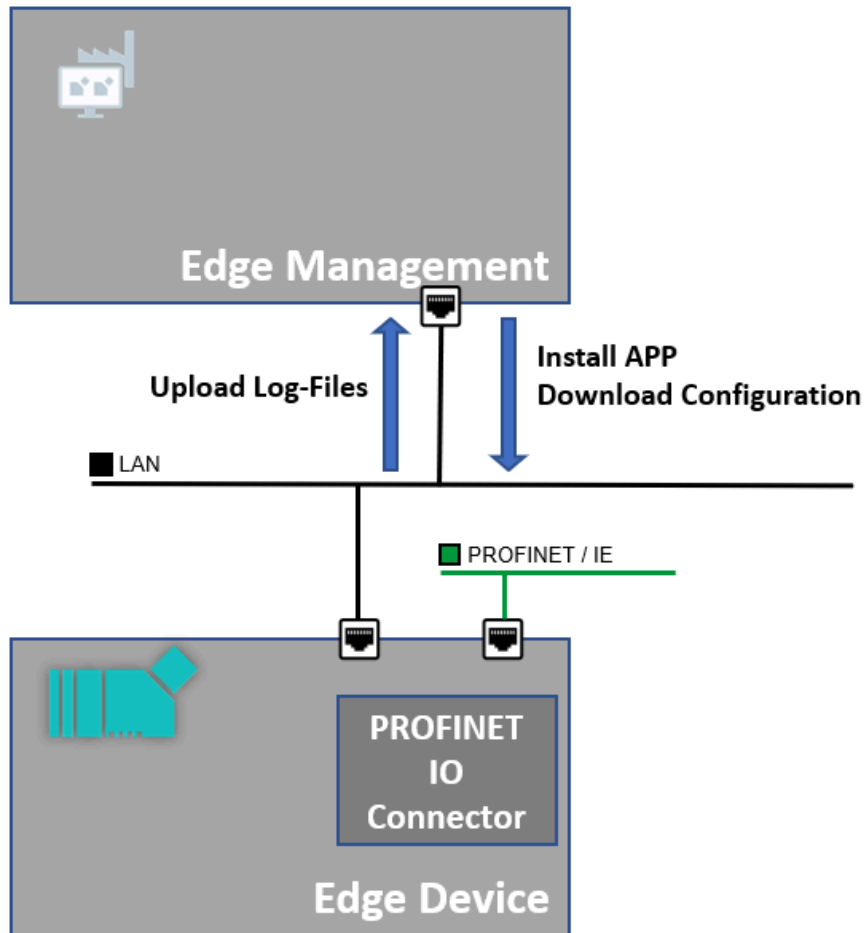
Involved Components

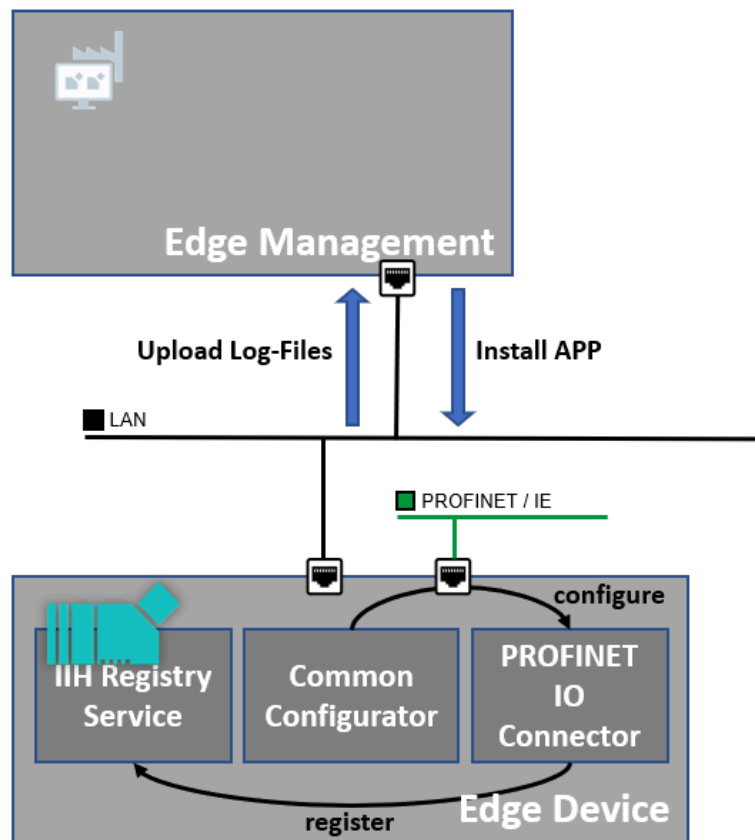
Apart from the IED with the installed PROFINET IO Connector and the connected PN Devices/ iDevices, Industrial Edge Management (IEM) is also considered.



To be able to use the PROFINET IO Connector, the following configuration steps are required:

- The IEM is used to install the app in the IED.
- The configuration must be inserted via IEM or Common Configurator, as shown in the following images respectively:





Installing PROFINET IO Connector

6.1 Prerequisites

The PROFINET IO Connector uses PROFINET communication. To guarantee proper work, refer PROFINET network specification (<https://www.profibus.com/>).

6.2 System Requirements

The system requirements for PROFINET IO Connector app are as follows:

Component		Requirement
Industrial Edge Management (IEM)	Platform	VM - ISO: Version 1.1.9
	Databus	V 1.1.15
Industrial Edge Device (IED)	IED Model	IPC 227E with minimum Quad core
	IED Version	From 'ied-os-1.1.0-39-amd64'
	Hard Disk	Minimum 1 GB available
	RAM	1 GB of available RAM to install app on IED
IIH Essentials	Version	V 1.8
External Dependency	Performance Insight (Optional)	V 1.1.0
	TIA Portal	V16+HSP or V17
	TIA Portal HSP (Hardware Support Package; included V17)	HSP 0307 (PN Driver V2.2) (SIOS 72341852)

The IIH Essentials is optional. The user app can directly subscribe JSON messages from the Databus.

If you want to use the Common Configurator to configure the PROFINET IO Connector app, different requirements are needed:

Component		Requirement
Industrial Edge Management (IEM)	Platform	V 1.5.6
	Databus	V 1.1.15
Industrial Edge Device (IED)	IED Model	IPC 227E with minimum Quad core
	IED Version	From 'ied-os-1.3.0-57-amd64'
	Hard Disk	Minimum 2 GB available
	RAM	Minimum 1 GB available
IIH Essentials*	Version	V 1.8
IIH Registry Service	Version	V 1.8.2
IIH Semantics	Version	V 1.4.0
Common Configurator	Version	V 1.8
External Dependency	Performance Insight (Optional)	V 1.1.0
	TIA Portal	V16+HSP or V17
	TIA Portal HSP (Hardware Support Package; included in V17)	HSP 0307 (PN Driver V2.2) (SIOS 72341852)

Industrial Edge Management (IEM)

In the current version of PROFINET IO Connector, the app is tested with IEM V1.1.0. This IEM V1.1.0 version supports Industrial Edge Device onboarding including 'Layer-2-Access' configuration.

Industrial Edge Device (IED)

IED must be onboarded in IEM. IED needs special network configuration for 'Layer-2-access' to run the PROFINET IO Connector. From IED V1.1, the 'Layer-2-Access' is an integrated functionality. This is required to operate the PROFINET IO Connector.

Note

- It is recommended to configure the 'Layer-2-Access' during onboarding process in the IEM. For more information, refer How to Configure Layer-2-Access? (Page 121)
- When you are using more than one application using the Layer-2-Access, the resulting IP address range must be greater than 2.

During the onboarding in IEM, you must configure the following network interfaces of the IED:

- Network interface 1: standard interface with gateway functionality.
- Network interface 2: Profinet interface with activated 'Layer-2-Access'.

New Edge Device

1 Device

2 Network Interface

3 Proxy

Back Next

Network Interface

Gateway Interface	MAC Address	DHCP	IPv4	Netmask	Gateway	Primary DNS	Secondary DNS	L2	Actions
✓	d4:f5:27:2a:42:69	✓	192.168.178.227	255.255.255.0	192.168.178.1	—	—	✓	
✓	d4:f5:27:2a:42:68	✓	192.168.152.143	255.255.255.0	—	—	—	✓	

NTP Server

Add Network Interface

☐ Gateway Interface

MAC Address
d4:f5:27:2a:42:68

☐ DHCP

Static

IPv4

192

168

152

143

Netmask

255

255

255

0

Gateway

DNS

Primary DNS

Secondary DNS

Layer 2 for Apps (L2)

Start IP Address

192

168

152

144

Netmask

255

255

255

0

IP Address Range

16

Add

The IP address depicted in the above images is the sample value. You must configure these values as required.

Databus app

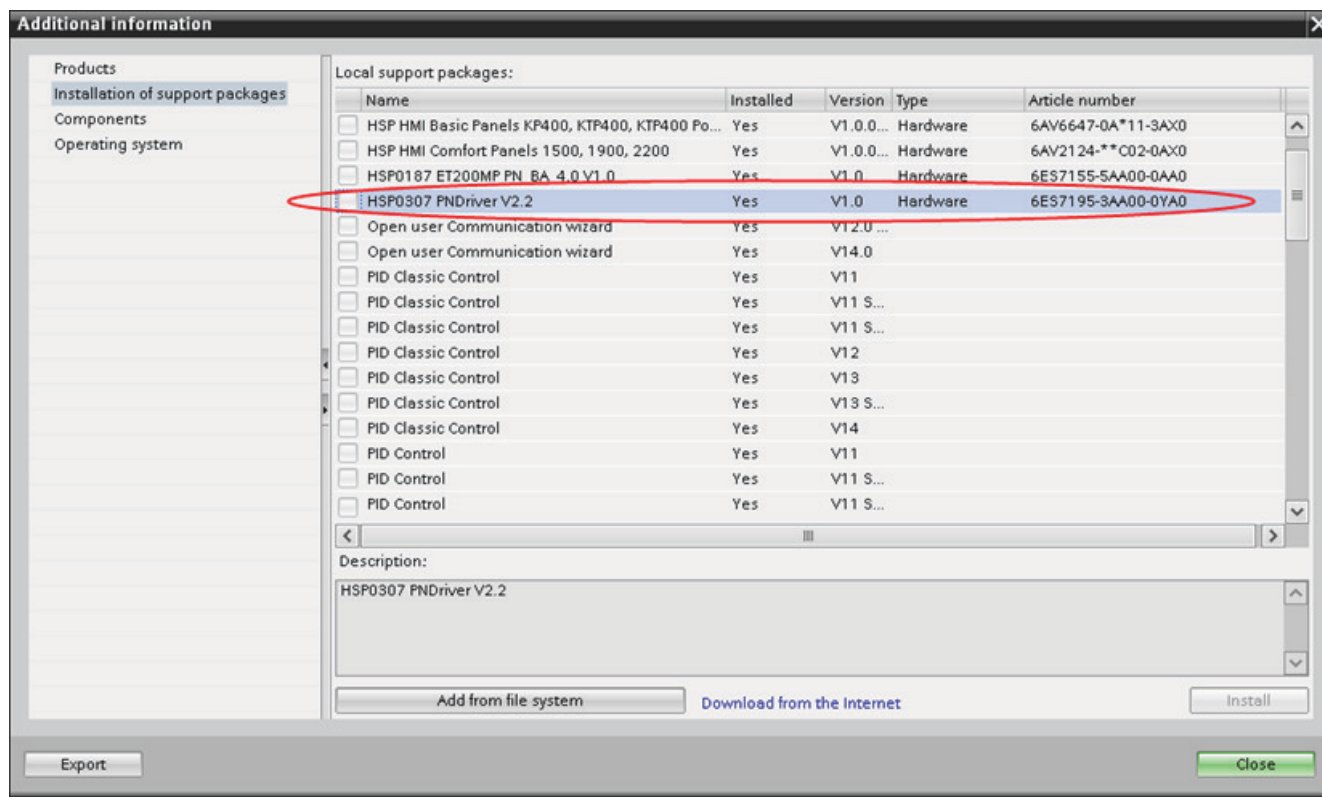
The PROFINET IO Connector publishes its data to Databus. The Databus provides this published data to other subscribing apps. You must install Databus app on the IED.

TIA Portal V16 (HSP for the PN Driver V2.2)

The PN Driver V2.2 is not included automatically in TIA Portal V16. You must install the HSP (Hardware Support Package). You can download the needed HSP 0307 from the Siemens support pages.

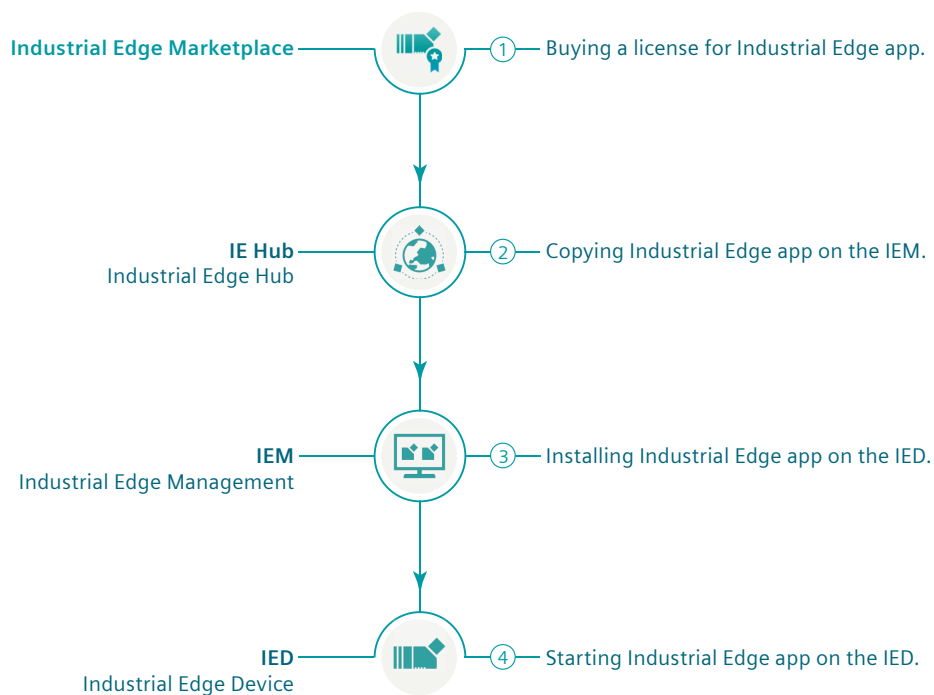
Note

With TIA Portal V17, this HSP is not required. The PN Driver V2.2 is integrated in TIA Portal V17.



6.3 Overview of the installation process

Installation process of an Industrial Edge app on an IED:



6.4 Buying an app

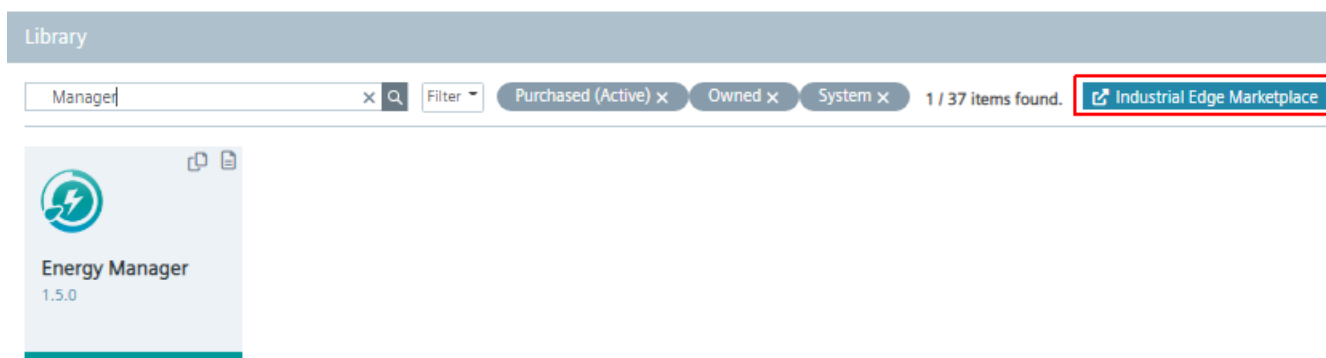
You can use the IE Marketplace to purchase an app or app license. To purchase an app, you need an access code.

Requirement

You have received the access code from your regional Siemens contact.

Procedure

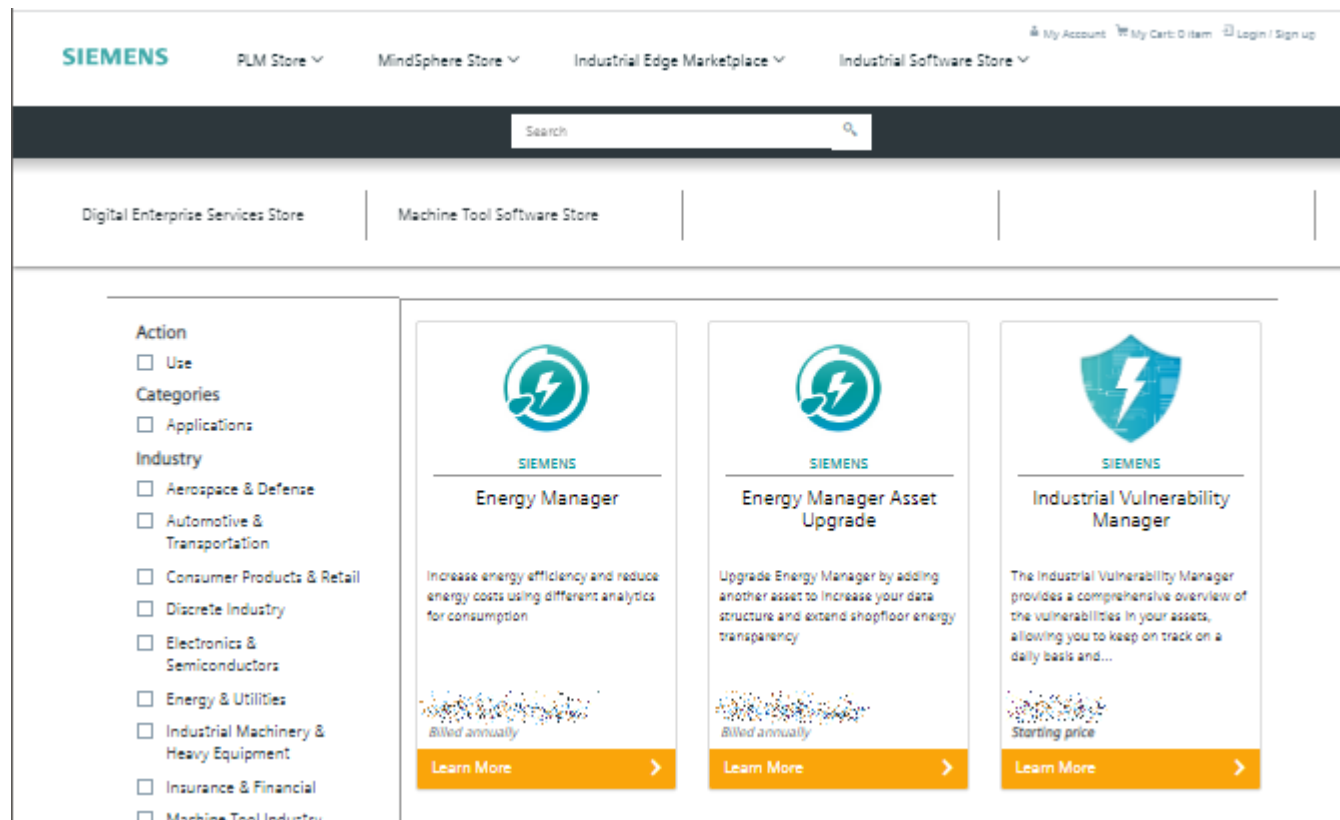
1. In the "Library" screen of IE Hub, click "Industrial Edge Marketplace".



2. Click "Manufacturing & Process Industries".
The Industrial Edge Marketplace opens.

6.4 Buying an app

3. Click the tile of the app you want to buy.



The app description and details are displayed.

On the left-hand side, you can see all the preconditions and requirements that apply to the execution of this app in IEM.

You can purchase all the products you require in one transaction.

4. Enter the number of licenses required in the "Quantity" input field.

Siemens PLM > Industrial Edge Marketplace > Manufacturing & Process Industries > Performance Insight

**Prerequisites** 2 REQUIRED

- * Industrial Edge Hub Access
- * Data Service

Additional Requirements

Hardware Demands
Industrial Edge Management License
Industrial Edge Devices in Industry Mall

Terms & Conditions

Universal Customer Agreement
Industrial Edge Supplemental Terms (alpha code 'IE')

Performance Insight

Provided by SIEMENS

Increase your machine's productivity and detect the potential for improvement with best in class data visualization tailored to your needs. Proven SIMATIC automation know-how now available on Industrial Edge.

Are you a machine manufacturer or machine operator, working in the field of industrial automation?

Do you want to improve machine productivity and service?

Are you interested in creating new business models to generate additional revenue?

Performance Insight enables you to achieve your goals by tracking the important machine performance indicators. With various visualization options, machine conditions are displayed according to the user's needs.

Subscription per installed app instance including 3 Assets.

Quantity
Minimum: 1

Total:

Add to Cart

Disclaimer: Taxes may apply

Infrastructure provider: AWS

Billing Term: Yearly

Subscription Term: Yearly

Supported Purchase Methods: Credit Card, Payment On Account

5. Click "Add to Cart".
The shopping cart is displayed.

Cart

 Billing Term: Yearly Subscription Term: Yearly Additional Info Prerequisites 2 REQUIRED * Industrial Edge Hub Access * Data Service	Qty 10 Minimum: 1	Price 	Supported Purchase Methods Credit Card Payment On Account	Delete
---	--	-----------	---	--------

Continue Shopping

Payment Type

Total:

Prices and offers in the cart are subject to change until the order is submitted.

6. Enter the received access code.

6.4 Buying an app

7. Add other products that are required to use the app to the shopping cart.
To do this, click on the corresponding links under "Prerequisites" and add all the desired products to the shopping cart in the same way.

 **Cart**

 ... Billing Term: Yearly Subscription Term: Yearly Additional Info	Qty 10 Minimum: 1	Price ...	Supported Purchase Methods Credit Card Payment On Account 
 Industrial Edge Hub Access Billing Term: Yearly Subscription Term: Yearly Additional Info	Qty 1 Valid Input: 1		Supported Purchase Methods Credit Card Payment On Account 
 Data Service Billing Term: Yearly Subscription Term: Yearly Additional Info	Qty 10 Minimum: 1		Supported Purchase Methods Credit Card Payment On Account 

Total: 

Prices and offers in the cart are subject to change until the order is submitted.

Payment Type 

8. Select your preferred payment type from the "Payment type" selection list.
Only "Credit Card" payment method is available for third-party apps.

9. Check again the information provided.

Note

You can edit the number of licenses again. Then click "Apply Quantity Changes".

10. Click "Checkout" and follow the instructions on the screen.

Cart

Item	Qty	Price	Supported Purchase Methods	Action
[Redacted] Billing Term: Yearly Subscription Term: Yearly Additional Info	10 Minimum: 1	[Redacted]	Credit Card Payment On Account	Delete
Industrial Edge Hub Access Billing Term: Yearly Subscription Term: Yearly Additional Info	1 Valid Input: 1		Credit Card Payment On Account	Delete
Data Service Billing Term: Yearly Subscription Term: Yearly Additional Info	10 Minimum: 1		Credit Card Payment On Account	Delete

Promo Code

Total: [Redacted]

Prices and offers in the cart are subject to change until the order is submitted.

Payment Type
Payment On Account

After you purchase the app, it appears in the "Library" section of IE Hub. From here, you can copy the app to your IEM instances. The number of licenses, the license itself and other details are displayed under "Licenses". If necessary, you can purchase additional licenses of the app in question from this location.

6.5 Copy Apps to IEM Catalog

You can copy the PROFINET IO Connector application and required system apps directly to the catalog of one of your IEM instances.

To copy an app to the IEM catalog, follow these steps:

1. Click  on the App tile in "Library" section.
The "Copy Application to IEM catalog" dialog box is displayed.
2. Specify the IEM instance from the drop-down list to which you want to copy the app.
3. Click "Copy".
The app is started copying and a job is created. You can check the status of the job in the job status screen of the specific IEM instance.

6.6 Install PROFINET IO Connector Application

You can install PROFINET IO Connector application using the catalog. To run the PROFINET IO Connector application, you must install the following on the corresponding Industrial Edge Device (IED):

- Databus
- IIH Essentials
- PROFINET IO Connector app

Note

All applications must be deployed in a single IED.

Prerequisite

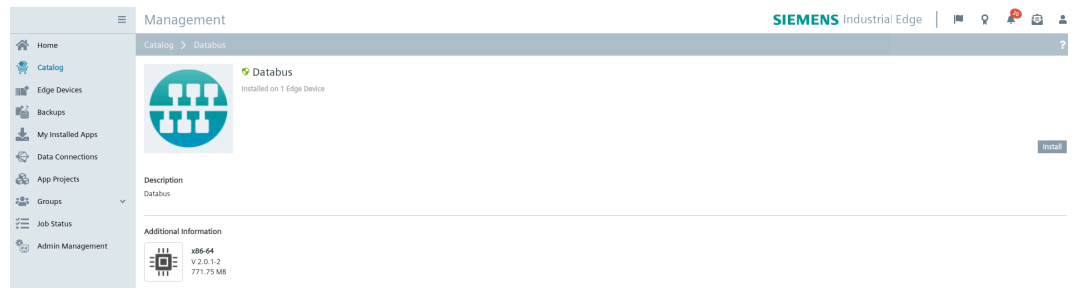
- You must be logged into "Industrial Edge Management".
- The apps have been copied to catalog as described in Copy Apps to IEM Catalog (Page 36).

Procedure

The PROFINET IO Connector application installation consists of the following steps:

Step 1: Databus deployment

1. Launch "Catalog" and click "Databus".
The page is displayed as follows:



2. Click "Install".
The "Install App" dialog box is displayed.
3. Select the IEDs in which you want to install the app.

6.6 Install PROFINET IO Connector Application

4. Do one of the following:
 - Click "Install Later" to schedule the installation date and time.
 - Click "Install Now" to install the app now.

If you select "Install Now", then an alert message is displayed.

5. Click "Allow".
The Databus is started installing in the corresponding IEDs.

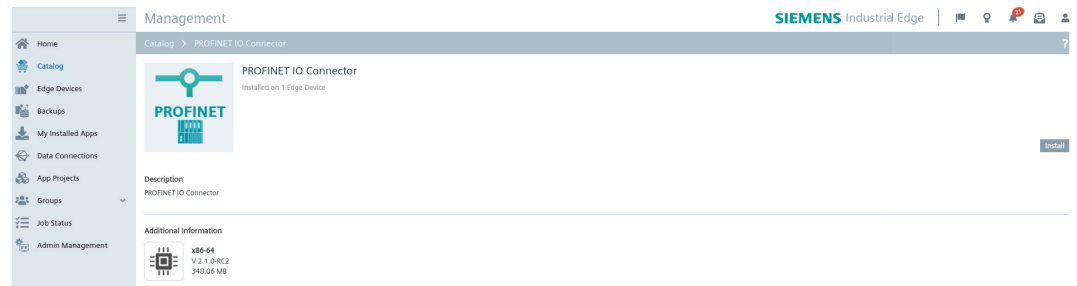
Note

Similarly, you can install IIH Essentials.

Step 2: PROFINET IO Connector application deployment

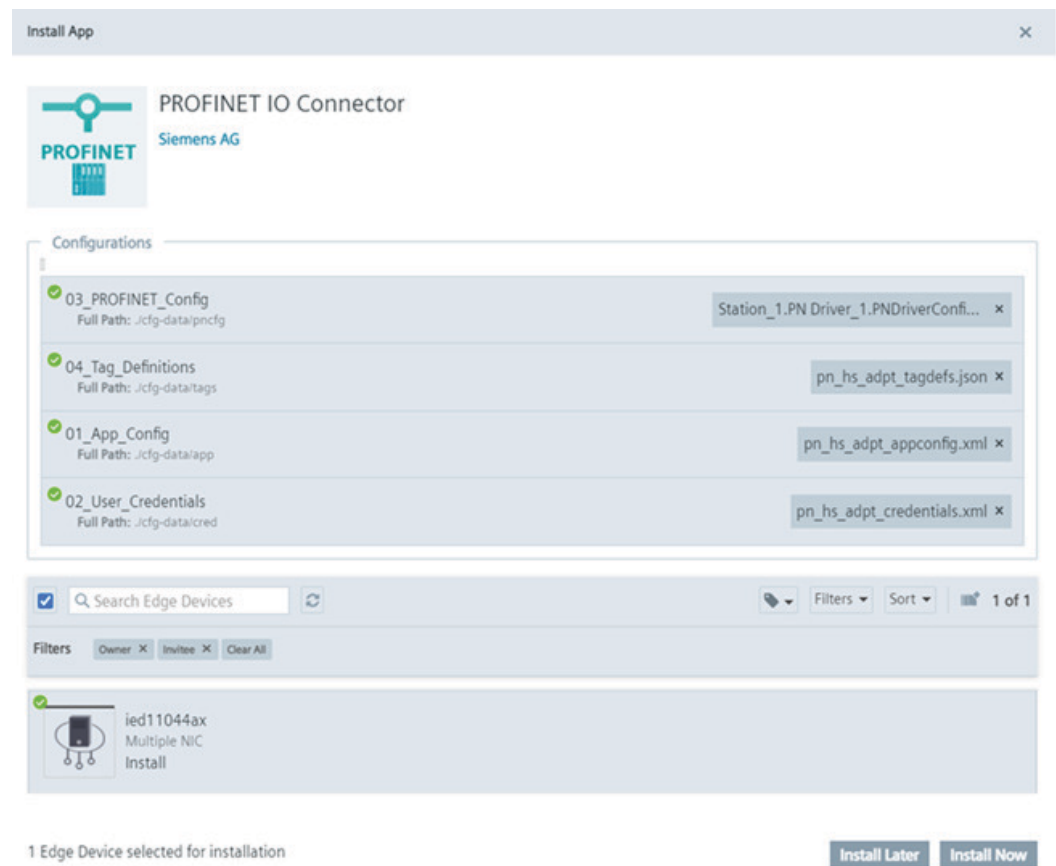
1. Launch "Catalog" and Click "PROFINET IO Connector".

The page is displayed as follows:



2. Click "Install".

The "Install App" dialog box is displayed as follows:



You must select all required configuration files in "Configuration" section. The selected configuration files are displayed with green-check circles.

Note

When used with Common Configurator, no configuration file must be uploaded. It is only relevant when you want to use the old configuration method.

Note

Sample Configuration Files

You can download the sample configuration files from here (<https://support.industry.siemens.com/cs/document/109793251>).

This section comes under 4 file Configuration method also referred as "Legacy mode". If you prefer to use the 4 file configuration method, it is still available for configuration. However, if you opt to use the Common Configurator, you will only need to provide a single configuration file in JSON format. For more information, refer Configuring PROFINET IO Connector using Common Configurator (Page 72).

3. Select the IEDs in which you want to install the app.
4. Do one of the following:
 - Click "Install Later" to schedule the installation date and time.
 - Click "Install Now" to install the app now.

If you select "Install Now", then an alert message is displayed.
5. Click "Allow".

The PROFINET IO Connector is started installing in the corresponding IEDs.

Configuring PROFINET IO Connector

The PROFINET IO Connector app can be used according to the project's needs. You can configure the behavior of app using the configuration files.

The Databus needs some configuration for the PROFINET IO Connector (topics and user). These settings must fit to the PROFINET IO Connector configuration.

The PROFINET configuration is configured with the TIA Portal.

There are three ways to configure the PROFINET IO Connector:

1. Four separate configuration files (legacy; File based configuration via IEM)
2. One JSON configuration file (recommended; File based configuration via IEM)
3. Configuration using the Common Configurator application (UI-based)

Note

Using the Common Configurator through a user interface, you can create a JSON configuration file. This generated JSON file is identical to the one employed for file-based configuration via the IEM.

Profinet IO Connector configuration is carried out using the TIA Portal. For more information, refer PROFINET Configuration in TIA Portal (Page 55).

Switch between configuration methods

1. Configure App with four config files, then switch to use one JSON file

If the application was initially configured with four configuration files, switching to the JSON config is straightforward. The Common Configurator can either pass the new JSON config to the app, or the JSON file can be manually uploaded via the IEM.

When the Common Configurator requests the configuration from the PROFINET IO Connector, the app converts the currently loaded configuration to the JSON file. Then the required changes can be edited instead of creating the whole JSON.

2. Configure App with Common Configurator, then switch to use four config files

Because the PROFINET IO Connector ignores the four config files when a JSON config exists, the new JSON config has to be deleted before updating and using the four config files.

Note

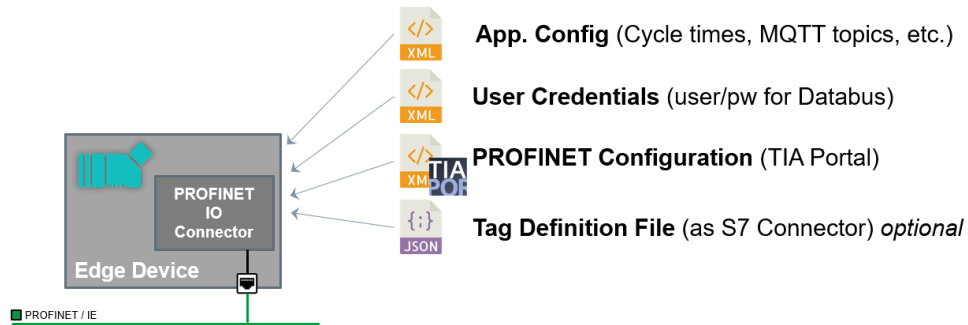
Currently, it is not possible to delete the newly created JSON file configuration because it was automatically generated by the app and was not passed through the IEM. In this scenario, attempting to delete the configuration via the IEM results in an error message: "Application configuration not found."

Follow below steps as a workaround:

- **Download Configuration File:**
 - Download the `cs/profinet-conn-config.json` file from the IED where the PROFINET IO Connector is installed.
 - **Upload Configuration File via IEM:**
 - Access the IEM interface.
 - Upload the `profinet-conn-config.json` file that you downloaded in the previous step.
 - **Delete Configuration File via IEM:**
 - Locate the `profinet-conn-config.json` file that you previously uploaded.
 - Delete this configuration file from the IEM interface.
-

7.1 Configuration using four files

The PROFINET IO Connector application requires configuration files as depicted in the following image:



- For user credentials for Databus (`pn_hs_adpt_credentials.xml`)
- For application settings (`pn_hs_adpt_appconfig.xml`)
- For PROFINET configuration (for example, generated from TIA Portal)
- For optional tag definition file (`pn_hs_adpt_tagdefs.json`)

Note

- When you change any config file, you must restart the app (for example, using IEM) to activate the changed configuration.
- Four file Configuration method also referred as "Legacy mode". You can still configure using 4 file configuration method.
If you want to use the Common Configurator, only one configuration file is required that is known as JSON file. For more information, refer Configuring PROFINET IO Connector using Common Configurator (Page 72).

Note

Sample Configuration Files

You can download the sample configuration files from here (<https://support.industry.siemens.com/cs/document/109793251>).

Update all configuration files from IEM

Perform below steps to update all or selected configuration files from the IEM .

1. Navigate to "IEM > My Installed Apps > PROFINET IO Connector > Update Configuration" section. The page is displayed as follows:

Update Configuration [X]

PROFINET IO Connector
Siemens AG
Installed on ied11044ax

Configurations

✓ 04_Tag_Definitions Full Path: .icfg-data/tags	pn_hs_adpt_tagdefs.json ×
✓ 01_App_Config Full Path: .icfg-data/app	pn_hs_adpt_appconfig.xml ×
✓ 03_PROFINET_Config Full Path: .icfg-data/pncfg	Station_1.PN Driver_1.PNDriverConf... ×
✓ 02_User_Credentials Full Path: .icfg-data/cred	pn_hs_adpt_credentials.xml ×

Actions

None [v]

Update Later **Update Now**

2. Select all required configuration files in "Configurations" section.
The selected configuration files are displayed with green-check circles.
3. In the "Actions" section, select "Operation – Restart".
4. Click "Update Later" or "Update Now" as per the requirement.
The selected configuration files will be updated.

User credentials for Databus

The XML file `pn_hs_adpt_credentials.xml` contains the username and password for the Databus (MQTT broker). This configuration must fit to the Databus configuration.

Application Settings

The XML file `pn_hs_adpt_appconfig.xml` contains several parameters for the PROFINET IO Connector app. Each line has a comment. You can adjust the parameters according to your needs. For example:

- Name of the PN config file, created by TIA Portal.
- Name of the optional tag definition file.
- MQTT topic names for published and subscribed topics.
- Cycle time for reading the PN IO data.
- Oversampling factor (number of PN IO cycles are transferred with one MQTT message).

Name of the MQTT Broker

You must configure the name of the MQTT Broker to publish/subscribe the MQTT topics.

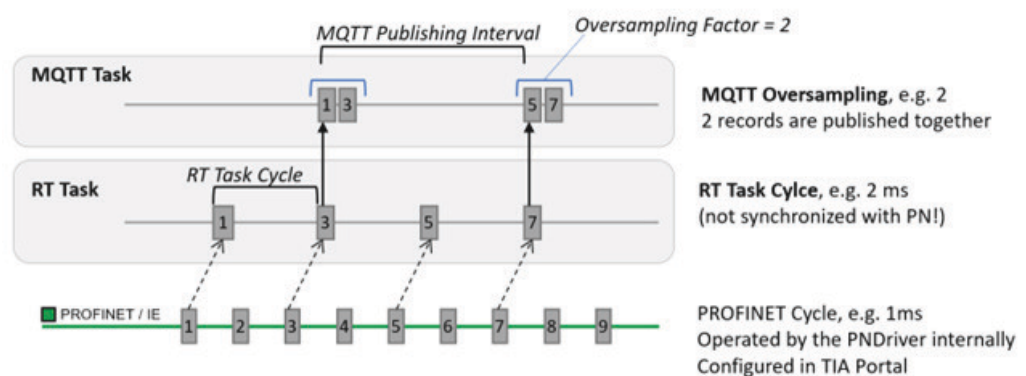
You must configure the "ie-databus" (IED V1.1) when using the Databus.

Configure the Time Behavior

The following table lists the two mandatory parameters for the configuration of the timing behavior of the connector.

Parameter	Details
RT Task Cycle [us]	Defines the cycle time of the real-time task. It reads the process image INPUT as record.
MQTT Oversampling Factor	Number of recorded cycles are put together to one MQTT topic - records.

You can configure the timing in the application configuration. The following figure shows the timing behavior:



The MQTT publishing interval can be calculated as follows:

$$[\text{MQTT publishing interval}] = [\text{RT task cycle}] \times [\text{Oversampling}]$$

In fact, the application internally uses more parameters.

Timing Configuration Rules and Hints

7.1 Configuration using four files

The timing configuration rules and hints are as follows:

- Adjust the RT cycle time as small as needed. Small values lead to high CPU and communication load.
- Adjust the MQTT Oversampling Factor to a resulting MQTT publishing interval of approximately 250 milliseconds (or greater). Short publishing intervals lead to high load of the Databus.
- The MQTT publishing interval has influence on the delay when using the PN Data Record Read function. The publishing of Data Record Read responses is connected to the publishing interval.

Example for 10 milliseconds recording cycle

Parameter	Value	Comment
RtTaskCycleUs	10000	Read PROFINET Input Data in 10 milliseconds cycle. For example, the cycle time of the real-time thread is 10 milliseconds.
MqttOversamplingFactor	25	Compose one MQTT message from 25 sets of PROFINET input data.

Note

The resulting MQTT publishing interval is $10 \text{ [ms]} \times 25 = 250 \text{ [ms]}$.

Example for 1 millisecond recording cycle

Parameter	Value	Comment
RtTaskCycleUs	1000	Read PROFINET input data in 1 millisecond cycle. For example, the cycle time of the real-time thread is 10 milliseconds.
MqttOversamplingFactor	250	Compose one MQTT message from 250 sets of PROFINET input data.

Note

The resulting MQTT publishing interval is $1 \text{ [ms]} \times 250 = 250 \text{ [ms]}$.

How to achieve high-speed recording with MQTT?

To manage the MQTT load, several PROFINET cycles (cycles of the real-time thread) are composed to one MQTT message = MQTT oversampling. So, one MQTT message contains several records as follows:

- You can configure independently:
 - In what cycle time the PN IO data should be recorded.
 - What MQTT oversampling should be used (how many PN IO records in one MQTT message).
- A typical configuration could be:
 - Record PROFINET input data every 5 milliseconds.
 - Publish the data every 250 milliseconds (includes 50 records of PN input data).

The PROFINET configuration is performed with TIA Portal. The edge app PROFINET IO Connector is the PROFINET controller (named "PN Driver") which controls the remote PN IO stations. You can configure PN iDevices as well for data exchange with other PLCs.

Tag Definitions

The PROFINET IO Connector supports tag definitions.

You can use the PROFINET IO Connector without the defined tags (just providing the raw PROFINET IO data). To do so, you can use the following two options:

1. Do not provide any tag definition file.
2. Provide a tag definition file without any defined tag

When the PROFINET IO Connector can read minimum one valid tag definition, it uses the tags. When the PROFINET IO Connector cannot read minimum one valid tag definition, it provides PNIO raw data. This may happen due to different reasons such as faulty JSON format or unavailability of the tag file.

Note

- In the current version of PROFINET IO Connector, the tag definition file name is fixed to `pn_hs_adpt_tagdefs.json`. You cannot change its name in the app configuration.
- Only in "legacy mode" user can provide a separate tag definition file. With the Common Configurator it is part of the configuration UI and also part of the one JSON config file.

Tag Definition File

The 'Tag Definition' file should be compatible with the JSON of the UI of SIMATIC S7 Connector.

The following example shows the corresponding 'Tag Definition' file:

```
{
  "message": "TAG_DEFINITION",
  "data": [{
    "tags": [
      {"id": 101, "name": "Toggle0", "address": "%I0.0", "dataType": "SimpleTagTypeBool"},
      {"id": 102, "name": "Toggle1", "address": "%I0.1", "dataType": "SimpleTagTypeBool"},
    ]
  }]
}
```

7.1 Configuration using four files

```

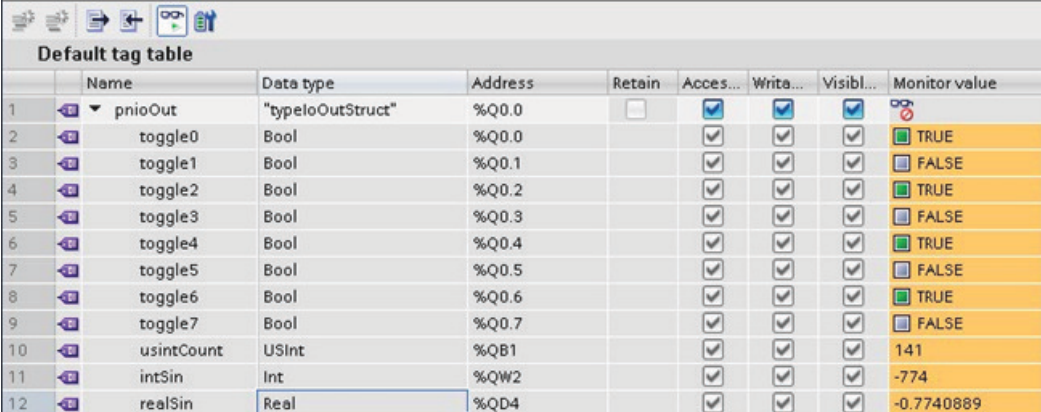
    {"id":103,"name":"Toggle2", "address":"%I0.2","dataType":"SimpleTagTypeBool"},
    {"id":104,"name":"Toggle3", "address":"%I0.3","dataType":"SimpleTagTypeBool"},
    {"id":105,"name":"Toggle4", "address":"%I0.4","dataType":"SimpleTagTypeBool"},
    {"id":106,"name":"Toggle5", "address":"%I0.5","dataType":"SimpleTagTypeBool"},
    {"id":107,"name":"Toggle6", "address":"%I0.6","dataType":"SimpleTagTypeBool"},
    {"id":108,"name":"Toggle7", "address":"%I0.7","dataType":"SimpleTagTypeBool"},
    {"id":109,"name":"usintCount","address":"%IB1", "dataType":"SimpleTagTypeUSInt"},
    {"id":110,"name":"intSin", "address":"%IW2", "dataType":"SimpleTagTypeInt"},
    {"id":111,"name":"realSin", "address":"%ID4", "dataType":"SimpleTagTypeReal"}
  }
}
}

```

The following table describes the properties of the tag definition:

Property	Comment
id	This is the connection for a specific tag between tag definition and metadata. The metadata refers to this id.
name	Symbolic name of the tag.
address	Input address according to the TIA Portal.
dataType	Type of the tag.

The following image depicts the TIA Portal example for the above 'Tag Definition' file:

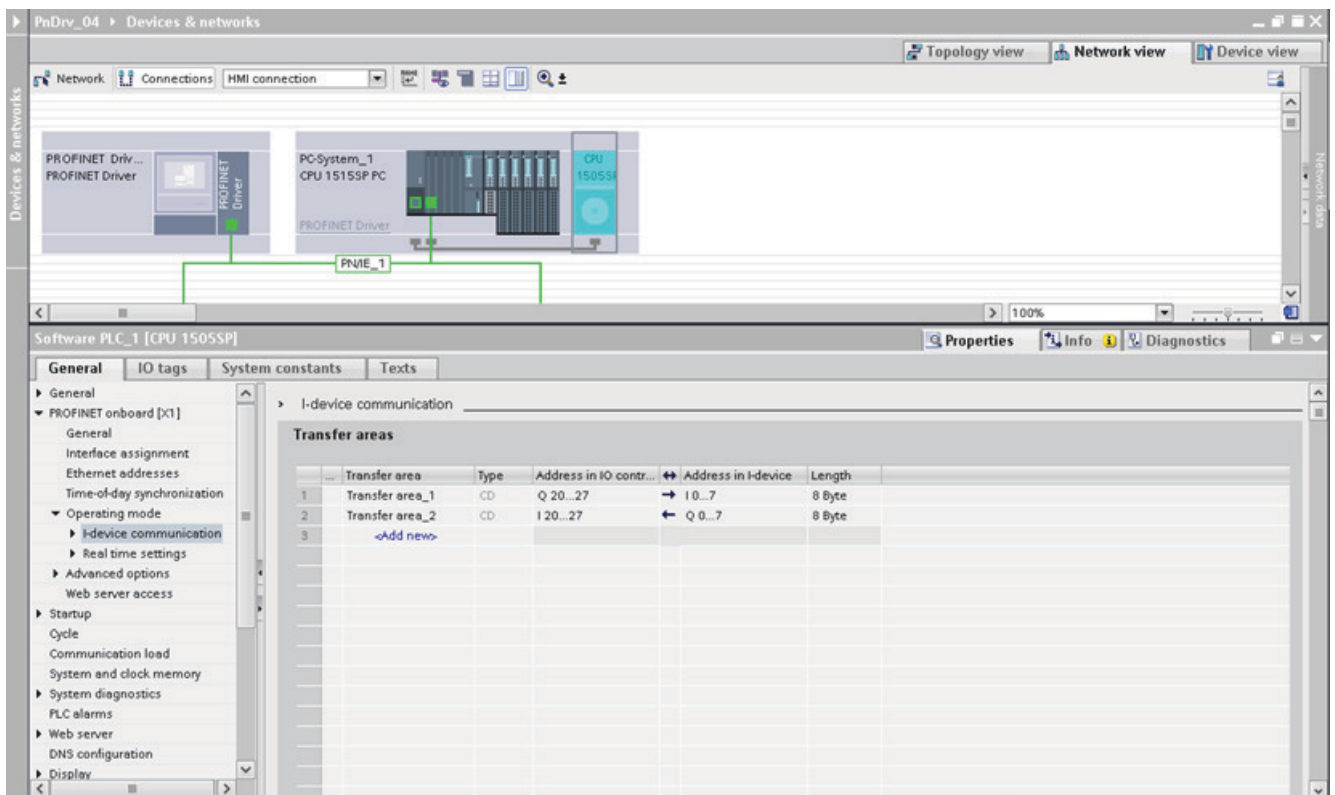


	Name	Data type	Address	Retain	Acces...	Write...	Visibl...	Monitor value
1	pnioOut	"typeloOutStruct"	%Q0.0	☐	☒	☒	☒	
2	toggle0	Bool	%Q0.0		☒	☒	☒	TRUE
3	toggle1	Bool	%Q0.1		☒	☒	☒	FALSE
4	toggle2	Bool	%Q0.2		☒	☒	☒	TRUE
5	toggle3	Bool	%Q0.3		☒	☒	☒	FALSE
6	toggle4	Bool	%Q0.4		☒	☒	☒	TRUE
7	toggle5	Bool	%Q0.5		☒	☒	☒	FALSE
8	toggle6	Bool	%Q0.6		☒	☒	☒	TRUE
9	toggle7	Bool	%Q0.7		☒	☒	☒	FALSE
10	usintCount	USInt	%QB1		☒	☒	☒	141
11	intSin	Int	%QW2		☒	☒	☒	-774
12	realSin	Real	%QD4		☒	☒	☒	-0.7740889

When you connect a PN iDevice to PROFINET IO Connector, the input/output transfer is processed as follows:

PROFINET IO Connector	ET200SP
%I0.0	Input Module
%Q0.0	Output Module

PROFINET IO Connector	PN iDevice
%I0.0	Output Transfer Area
%Q0.0	Input Transfer Area



Supported Data Types for Tag Definitions

The supported data types for tag definitions are as follows:

- SimpleTagTypeBool
- SimpleTagTypeByte
- SimpleTagTypeWord
- SimpleTagTypeDWord
- SimpleTagTypeLWord
- SimpleTagTypeSInt
- SimpleTagTypeInt
- SimpleTagTypeDInt
- SimpleTagTypeLInt
- SimpleTagTypeUSInt
- SimpleTagTypeUInt
- SimpleTagTypeUDInt
- SimpleTagTypeULInt
- SimpleTagTypeReal
- SimpleTagTypeLReal

Resulting Metadata

Depending on the tag definition, the PROFINET IO Connector generates the corresponding metadata.

The metadata can be used to parse the Binary Payload of the high frequency Read PN IO Data. To get a tag data out of the Binary Payload, the information "ioDataPointAddr" is required along with its submodule location. For example, IO Data Point index, Byte Offset, and Bit Offset.

We cannot use the original S7 address syntax because we do not provide an IO address. We provide the address within the IO Data Point. For example, PN submodule.

The following table shows the examples for IO-Data-Point Addresses:

Address	Definition
S0A5.1	Submodule Index 0, Offset inside Submodule Byte 5, Bit 1
S2A0	Submodule Index 2, Offset inside Submodule Byte 0

The metadata property sequence is used to provide the hash value for specific metadata and binary payload. In this way, the consumer has the chance to check the match of both.

For the tag definition example above the PROFINET IO Connector would create metadata as shown below:

```
{
  "seq": 3141217411, # hash as connection to thy cyclic payload
  "connections":[
    {
      "name":"profinetxdriver4933",
      "type":"pn",
      "dataPoints":[
        {
          "name":"default",
          "topic":"ieVdVbVsimaticVv1Vpnhs1VdpVr",
          "publishType":"timeseries", # can be 'bulk' or 'timeseries'
          "dataPointDefinitions":[
            {
              "name":"Toggle0", # name from tag definition file
              "id":"101", # reference to tag definition file
              "dataType":"Bool", # data type
              "ioDataPointAddr":"S0A0.0", # address information to parse the raw data format
              (binary payload)
              "acquisitionCycleInMs":"500",
              "acquisitionMode":"CyclicOnContinuous"
            },
            {
              "name":"Toggle1",
              "id":"102",
              "dataType":"Bool",
              "ioDataPointAddr":"S0A0.1",
              "acquisitionCycleInMs":"500",
              "acquisitionMode":"CyclicOnContinuous"
            },
            ...
          ]
        }
      ]
    }
  ]
}
```

7.2 Configuration of Databus and IIH Essentials apps

7.2.1 Configuration of Databus

The PROFINET IO Connector publishes data to the Databus. You must create a user who is allowed to publish topics.

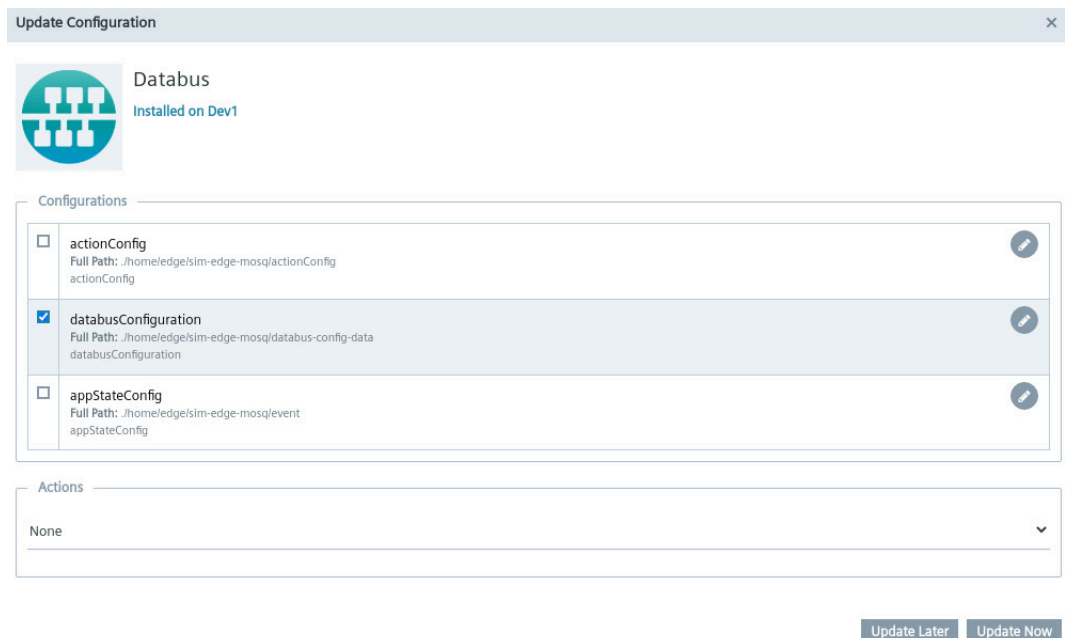
Prerequisite

- You must be logged into "Industrial Edge Management".
- The PROFINET IO Connector and Databus apps must be installed.

Procedure

To configure Databus, follow these steps:

1. Launch "IEM" and navigate to "My Installed Apps" > Databus > Update Configuration" section. "Update Configuration" window appears.



2. Select "databusConfiguration".
3. Click "Update Now".
The configuration is now updated.

Configuration Tasks

The configuration tasks consist of the following two sections:

- Create a user
- Create the MQTT topic(s)

Create a user

You will need to create a username and password.

Depending on the version of the PROFINET IO Connector application you are using, add these credentials to one of the following:

- For PN IO Connector version $\geq$ 2.0.0
Add them in the 'profinet-conn-config.json' file within the Common Configurator.
- For PN IO Connector version $<$ 2.0.0:
Insert them into the 'pn_hs_adpt_credentials.xml' file.

Create MQTT topics

The MQTT topics are defined in the app configuration file. The topics to be used are defined based on the version of the PN IO Connector application you are using:

- For PN IO Connector version $\geq$ 2.0.0:
Topics for the application are defined in the configuration file named "profinet-conn-config.json."
- For PN IO Connector version $<$ 2.0.0:
Topics for the application are defined in the user app configuration file named "pn_hs_adpt_appconfig.xml."

Since the topics may vary between different versions, it is essential to provide the databus user access to a variety of topics to ensure compatibility. For instance, you can use a wildcard topic structure like "ie/+/j/simatic/v1/pnhs1/#" to grant access to a range of relevant topics.

For specific topic configurations, refer to the default MQTT topics for:

- For Controller Configuration, refer step no. 6 of Add data sources (Page 76).
- For Device Configuration, refer step no. 6 of Add data sources (Page 82).

For practical examples of JSON configuration for both Controller and Device, you can find detailed guidance in the following resources:

- For Controller Configuration example, refer Example of JSON Configuration for Controller (Page 67).
- For Device Configuration example, refer Example of JSON Configuration for Device (Page 71).

Ensure that you configure the topics accurately based on your specific requirements and the version of PN IO Connector you are using. Proper topic configuration is essential for the seamless operation of your PROFINET application.

Figure below shows an example of a user along with corresponding topics and access rights:

Configuration for axburt6vm Import Export

User View Topic View

Add User Delete User Add Topic Delete Topic

User	☐	Topic	Access Rights
Search Username		Search Topic	Search Permission
edge	☐	ie/m/j/simatic-v1.0/pnhs1/dp/r	Publish and Subscribe
	☐	ie/d/b/simatic-v1.0/pnhs1/dp/r	Publish and Subscribe

Note

You should verify if the topics are actually stored in the configurator. Launch the configurator again and check if all topics are still there.

7.2.2 Configuration of IIH Essentials

The PROFINET IO Connector provides data to be stored in IIH Essentials. For this use case, some configuration of the IIH Essentials is required.

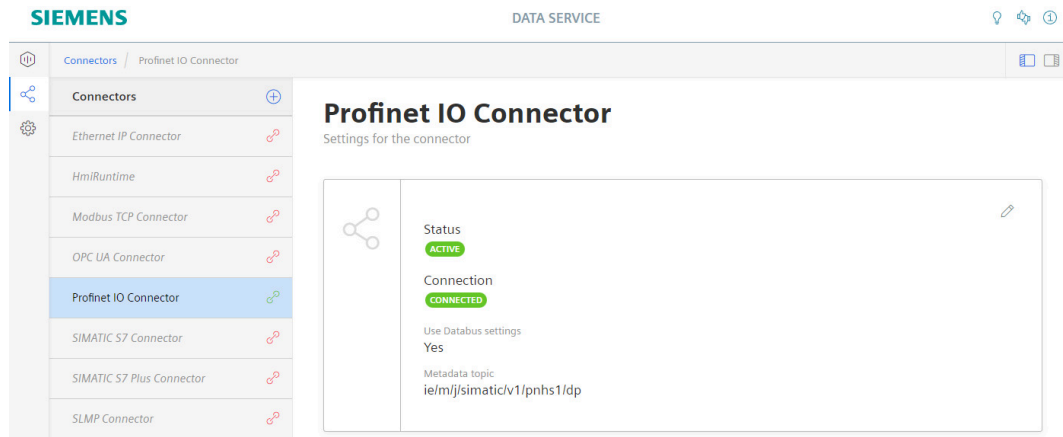
IIH Essentials subscribes the metadata of the PROFINET IO Connector to know the possible tags provided by the connector. When the metadata is read, IIH Essentials configurator offers to select specific tags to be stored. These stored tags can be processed by other applications.

IIH Essentials Connection

You must configure the metadata topic of the PROFINET IO Connector. The metadata topic you need to configure can be found in the following locations:

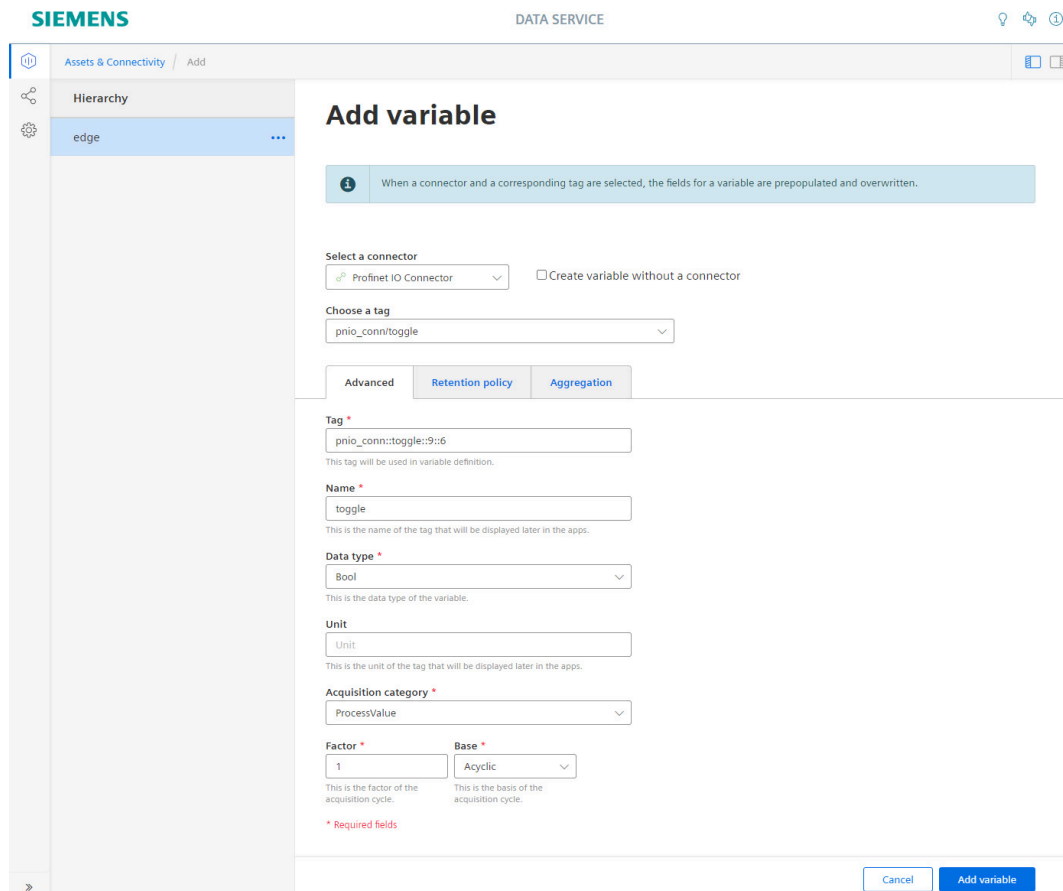
- For Controller Configuration, refer step no. 6 of Add data sources (Page 76).
- For Device Configuration, refer step no. 6 of Add data sources (Page 82).

A successful connection will appear as shown in the image below (IIH Essentials also known as Data Service):



IIH Essentials Tags

When IIH Essentials has received the metadata from PROFINET IO Connector successfully, the connection is marked green and you can select tags for storing as follows:



7.3 PROFINET Configuration in TIA Portal

7.3.1 Overview

The PROFINET IO Connector can be configured in TIA Portal in two ways:

1. Configure it as a PROFINET controller. For more information, refer Configuring PROFINET Driver as Controller (Page 55).
2. Configure it as a PROFINET device. For more information, refer Configuring PROFINET Driver as Device (Page 62).

Note

You can configure only one mode at any given time.

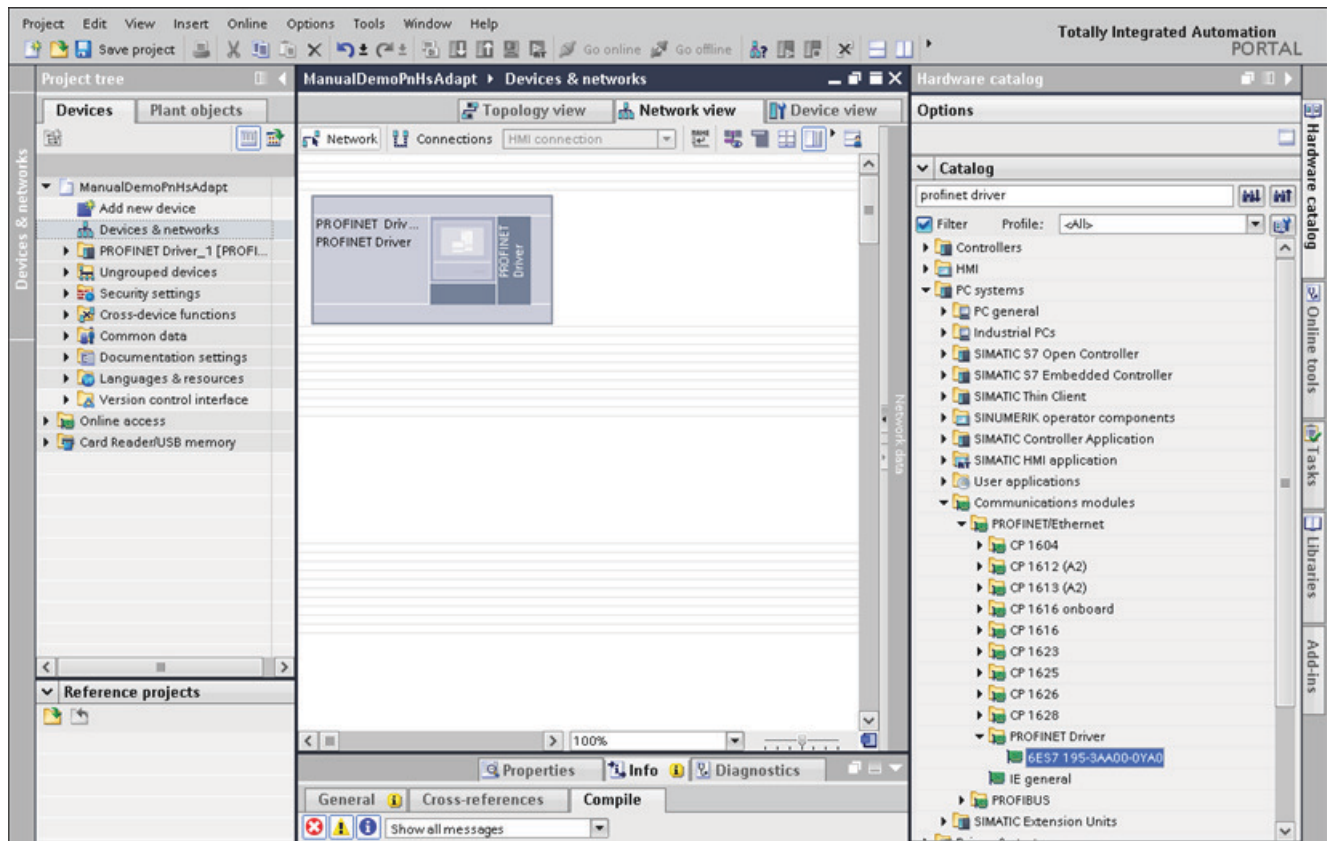
7.3.2 Configuring PROFINET Driver as Controller

Follow below steps to configure PROFINET Driver as a Controller:

1. Launch Siemens TIA Portal software and open your TIA Portal project.
2. Click on "Devices & Network" section within the TIA Portal.

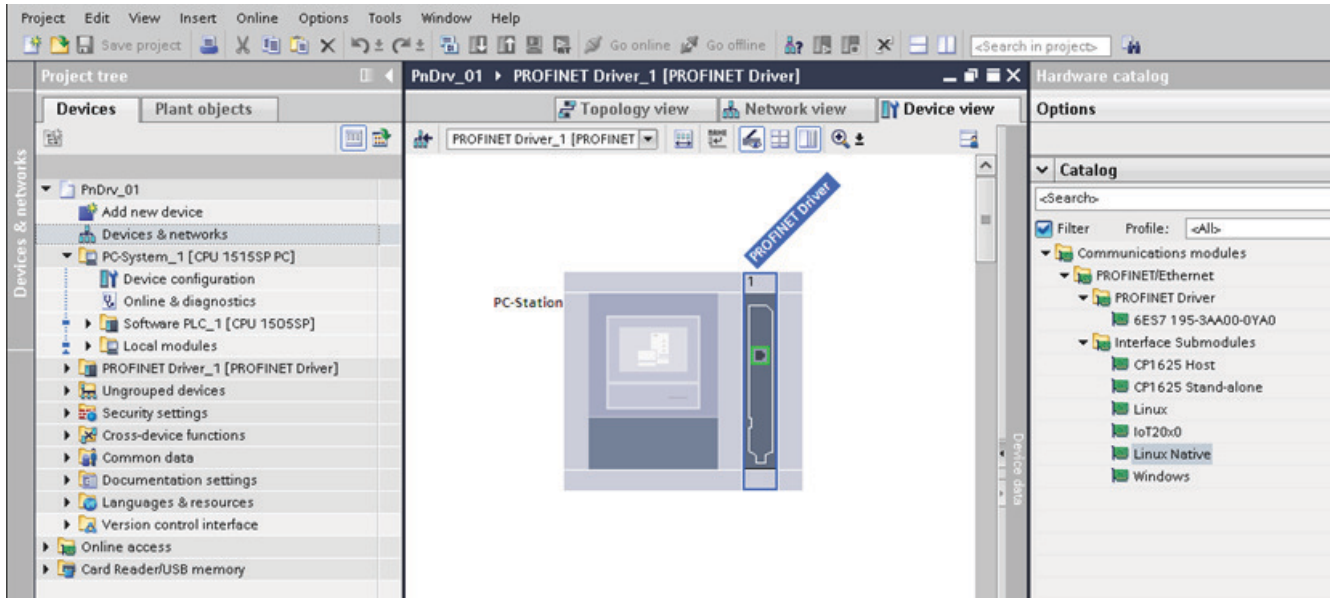
7.3 PROFINET Configuration in TIA Portal

3. Navigate to "PC Systems > Communications modules > PROFINET/Ethernet > PROFINET Driver" under Catalog section.
The page is displayed as follows:



4. Double-click on PROFINET Driver MLFB number to add new driver.
The project now contains a PC station with prepared PROFINET Driver.
5. Switch to the "Device view" from "Network View".

6. Add 'Linux Native' communication interface to the PROFINET Driver.



7. Configure PROFINET Network.

To configure the required PN devices and assign them to the PN Driver as a controller, follow these steps:

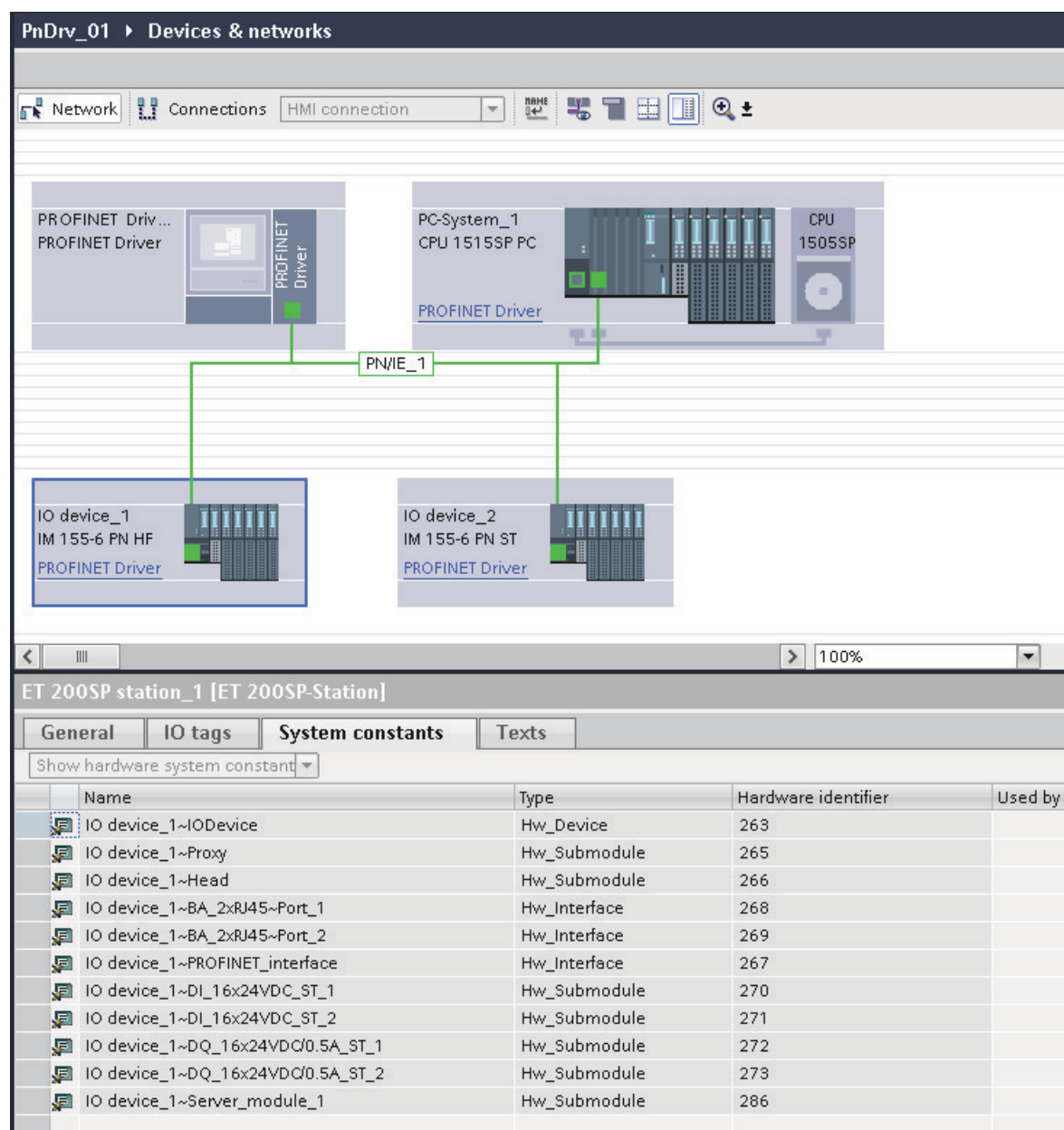
- Launch "Devices & Networks" if not already open.

- Navigate to "Network view > Properties > General > Overview of addresses (PROFINET IO-addresses)."

The screenshot shows the TIA Portal interface with the 'Network view' selected. The network topology displays a PC-System_1 (CPU 1515SP PC) connected to two IO devices (IM 155-6 PN HF and IM 155-6 PN ST) via a PROFINET network. The 'Properties' window is open for the 'PN/IE_1 [Industrial Ethernet]' network, showing the 'General' tab with the 'Overview of addresses' section. The table below lists the addresses for the IO devices.

Type	Addr. fr...	Addr. to	Size	Module	R...	S...	Device name
O	0	127	128 Bytes	Transfer area_1	0	1...	Software PLC_1 [CPU 1
I	0	127	128 Bytes	Transfer area_2	0	1...	Software PLC_1 [CPU 1
I	128	129	2 Bytes	DI 16x24VDC ST_1	0	1	IO device_1 [IM 155-6
I	130	131	2 Bytes	DI 16x24VDC ST_2	0	2	IO device_1 [IM 155-6
O	128	129	2 Bytes	DQ 16x24VDC/0.5A ST_1	0	3	IO device_1 [IM 155-6
O	130	131	2 Bytes	DQ 16x24VDC/0.5A ST_2	0	4	IO device_1 [IM 155-6

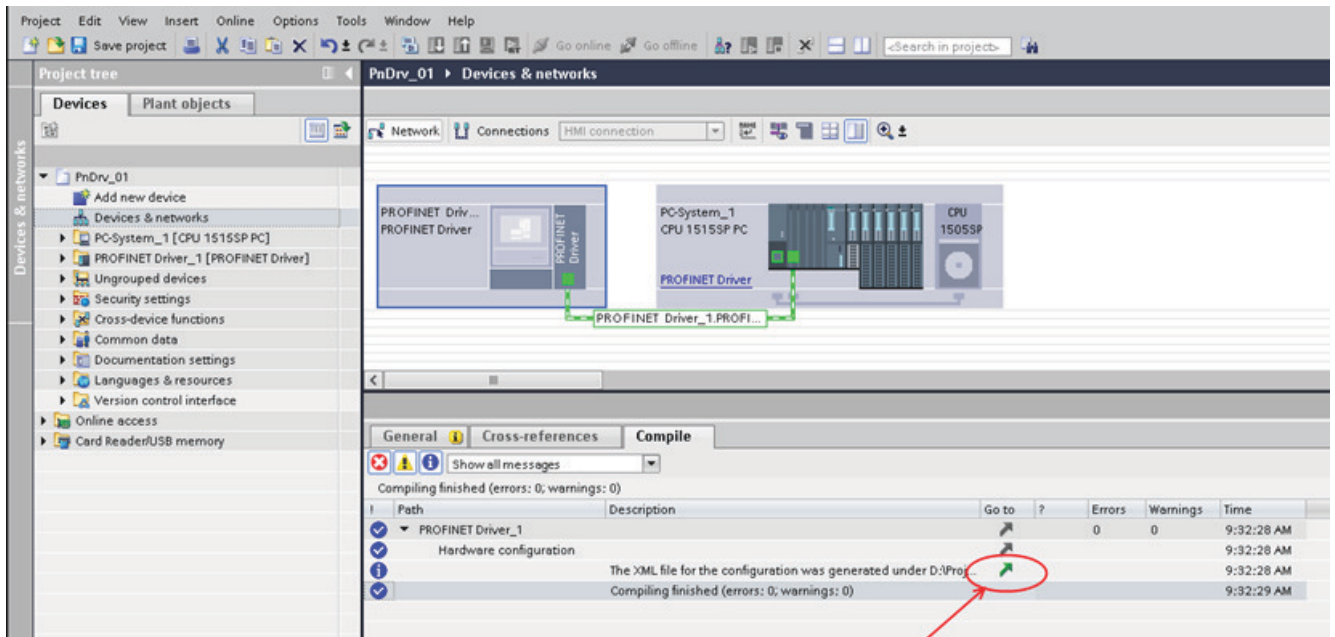
This information is important for the later processing of the cyclic PROFINET IO data, provided by the PROFINET IO Connector.
When using 'PROFINET Data Record Read' and 'PROFINET Data Record Write' functions, you will also need the logical address (LADDR) of the submodules. You can find these values in the system constants in the TIA Portal as shown in the below image:



8. PROFINET Configuration File

When you create a TIA Portal project with the PN Driver as a controller, an XML configuration file is generated during the compilation of the project.

This XML configuration file must be provided to the PROFINET IO Connector application for proper communication.



9. Activate Value Status

You have the option to activate the value status for IO modules. This provides additional diagnosis information for the application.

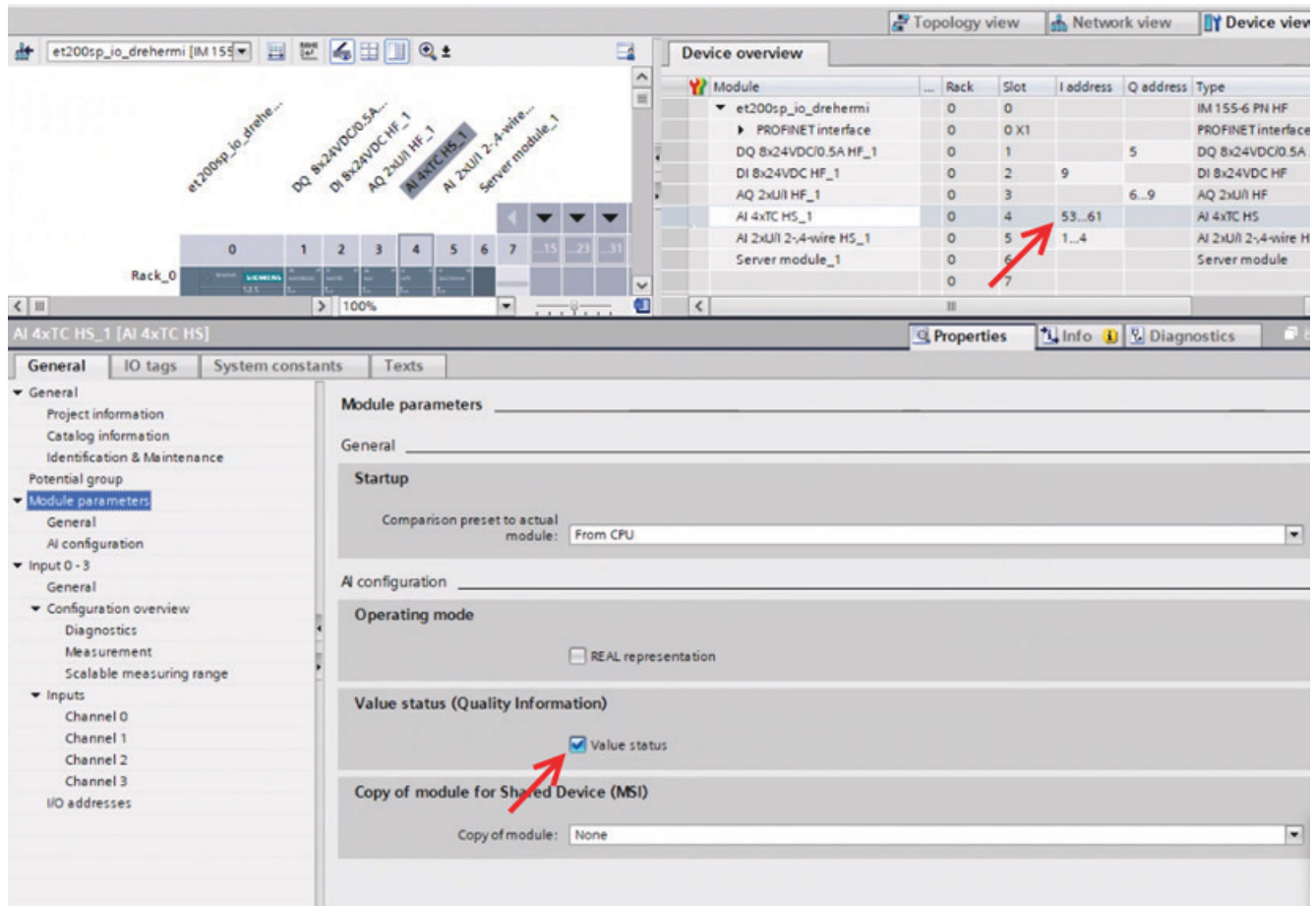
Note

Activating the 'Value status' increases the IO memory size.

To activate the 'Value status,' follow these steps:

- Launch "Device view" in the TIA Portal.
- Navigate to "Properties > General > Module parameters > Value status (Quality Information)".

- Enable the value status check box as shown in the below image:



7.3.3 Configuring PROFINET Driver as Device

Follow below steps to configure PROFINET driver as a device:

Step 1: Download the GSD Form

1. Navigate to the PROFINET IO Connector V2.1
2. Download the GSDML-V2.42-Siemens-PROFINET-IO-Connector-IOD-20230630.xml file from V2.1.0 of the link: SIOS (<https://support.industry.siemens.com/cs/us/en/view/109813215>).

Step 2: Copy GSD File to Project

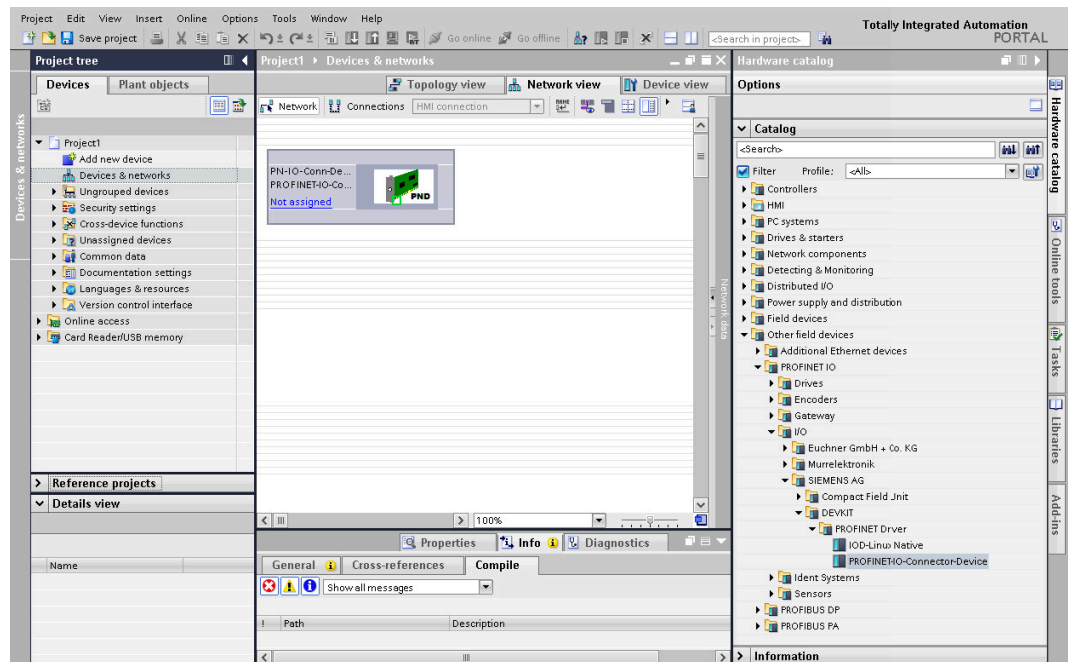
1. Open your TIA Portal project.
2. Locate the "AdditionalFiles" folder within your project directory (e.g., .../Project1/AdditionalFiles/GSD/).
3. Copy the downloaded GSDML-V2.42-Siemens-PROFINET-IO-Connector-IOD-20230630.xml file into the "GSD" subfolder.

Step 3: Load GSD in TIA Portal

1. Launch TIA Portal.
2. Go to the "Options" menu.
3. Select "Manage general station description files (GSD)".
4. Locate the GSDML file you copied in Step 2.
5. Select the GSDML file and click "Install".

Step 4: Configure PROFINET IO Connector

1. Go to TIA Portal and select the "Devices & Network" view.
2. In the Catalogue, navigate to "Other field devices" > "PROFINET IO" > "I/O" > "SIEMENS AG" > "DEVKIT" > "PROFINET Driver".



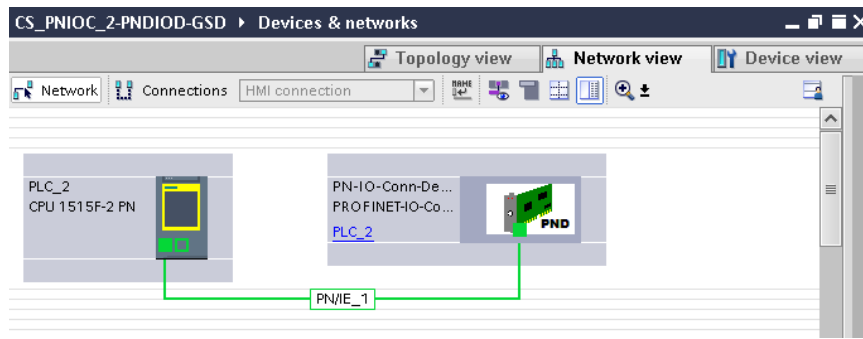
3. Double-click on "PROFINET-IO-Connector-Device" to add it to your project.

Step 5: Configure IP Address and Subnet

Set the IP address and subnet for the PROFINET IO Connector. Ensure it matches your controller's network configuration.

Step 6: Assign the Device to a Controller

Assign the PROFINET IO Connector to the appropriate controller in your project.



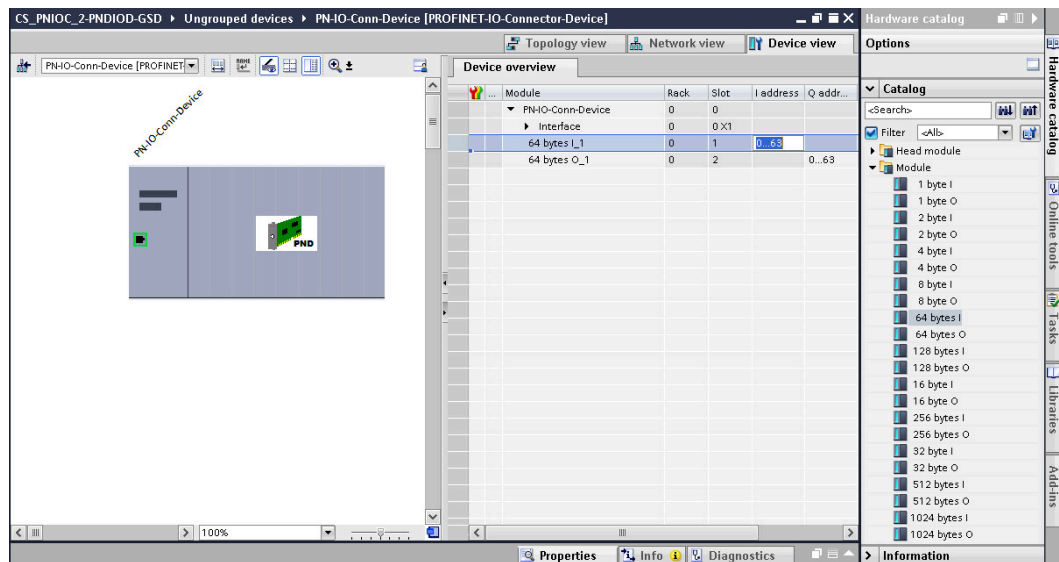
Information on IP Address configuration can be found here (Page 124).

Note

Configure module sizes

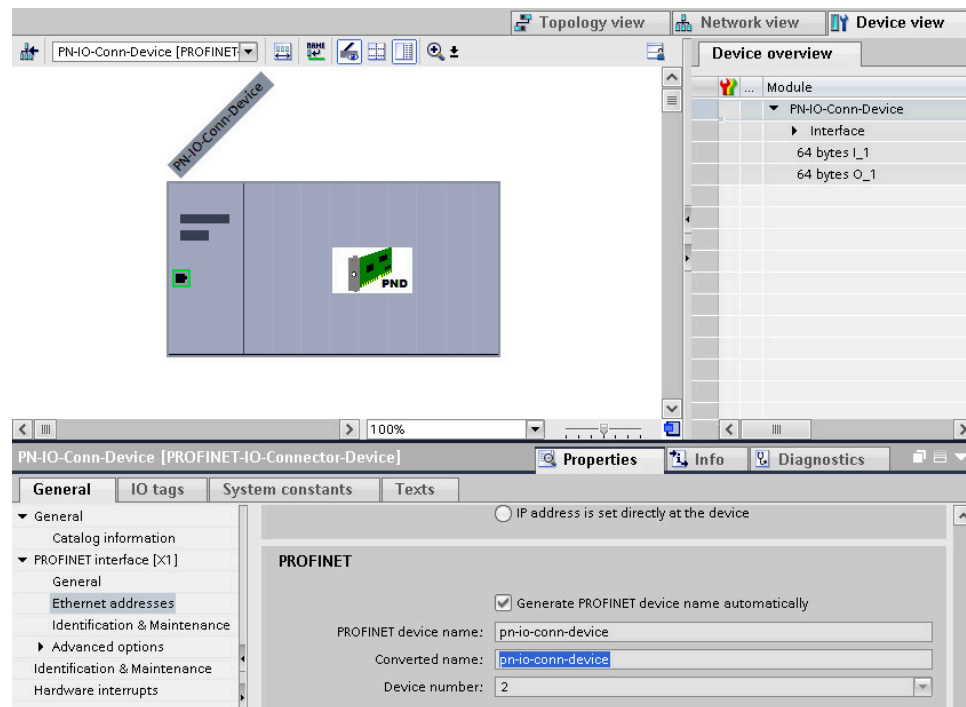
After the device is assigned to a controller, the addresses of the modules can be configured. In device mode, the PROFINET IO Connector offers two slots: one for an input module and one for an output module. These slots are optional, so there is no requirement to define both an input and an output module. It depends on your needs whether you configure only one input, one output, or both modules.

For each module, the user can choose a size ranging from 1 to 1436 bytes or select 0 to indicate no module. This module-size is important for later app-configuration.



Step 7: Set the PROFINET Name

1. The PROFINET name is important for device communication with the controller.
2. To change the name, click on the device in the Device View, navigate to "Properties" > "General" > "PROFINET Interface [X1]" > "Ethernet addresses."



3. Update the PROFINET name as needed.

Step 8: Note the PROFINET Device Name

Make a note of the PROFINET device name you've configured. You'll need this exact name later when configuring the application.
Failure to use the correct name may prevent a successful connection.

Note

The 'Converted name' is the correct one needed, not the 'PROFINET device name' from the image shown above.

7.4 Configuring PROFINET IO Connector using JSON file

7.4.1 Configuring PROFINET IO Connector as PN controller

7.4.1.1 Introduction

Prerequisites

- You must configure the PN Driver as Controller in TIA Portal. For more information, refer Configuring PROFINET Driver as Controller (Page 55).

Process

You can configure the PROFINET IO Connector as controller using a single JSON file. Perform below steps to configure using JSON file:

1. Download the JSON file from PROFINET IO Connector or create a new configuration file based on the example mentioned in Example of JSON Configuration for Controller (Page 67). For more information on how to download JSON file, refer Downloading the JSON file (Page 67). This JSON file contains the details of user credentials, application setting, app configuration, and data points.
2. Update the file as necessary and then upload the JSON file to the PROFINE IO Connector. The JSON file format is easy to read, configure and edit the details. For more information on how to configure using JSON file, refer Configure PROFINET IO Connector using JSON file (Page 68).

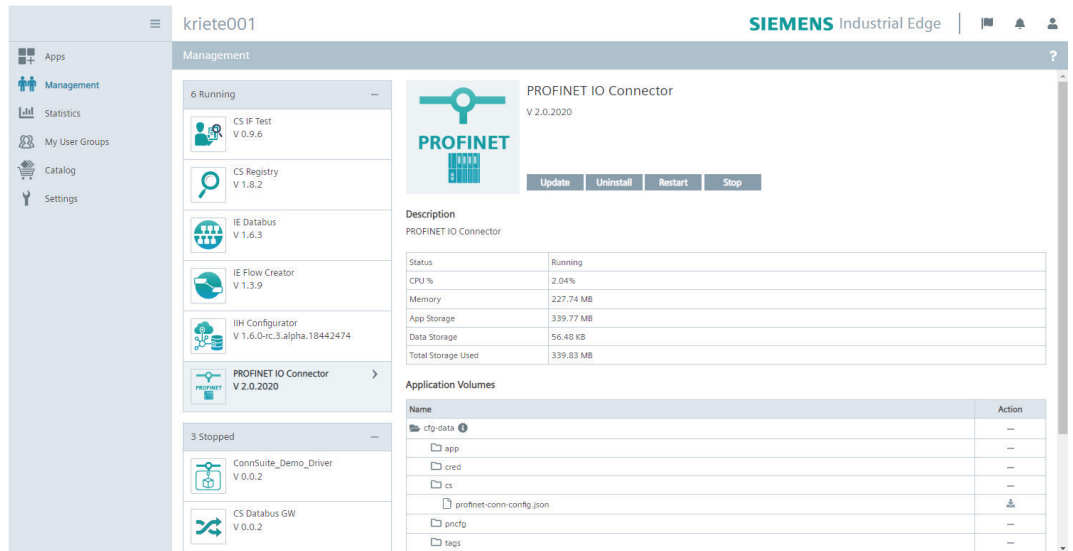
Note

If you are using a JSON file format for configuration, any modifications made to the other four files will not impact the functionality of the PROFINET IO Connector.

7.4.1.2 Downloading the JSON file

To download JSON file, follow the below steps:

1. Open Industrial Edge Device and go to "Management".
2. Click "PROFINET IO Connector". The PROFINET IO Connector page opens, as shown below:



3. In the "Application Volumes" section, click the download icon beside cs/Profinet-conn-config.json. The JSON file will be downloaded to the selected folder.

7.4.1.3 Example of JSON Configuration for Controller

Example of a JSON file is shown below:

```
{
  "configs": [{
    "$schema": "https://siemens.com/connectivity_suite/schemas/
profinet/1.0.0/config.json",
    "config": {
      "connections": [{
        "parameters": {
          "profinet_device_config": {
            "filename": "Station_1.PN-
Driver_1.PNDriverConfiguration.xml",
            "content": "PE..." # Base64-coded PROFINET
Configuration from TIA Portal
          },
          "rt_task_cycle_us": 4000,
          "mqtt_oversampling_factor": 100,
          "mqtt_publish_binary": false,
          "mqtt_topic_pn_rd_metadata": "ie/m/j/simatic/v1/pnhs1/
dp",
          "mqtt_topic_pn_rd_data_bin": "ie/d/b/simatic/v1/
pnhs1/dp/r/pnioc/default",
          "mqtt_topic_pn_rd_data_json": "ie/d/j/simatic/v1/
```

```

pnhs1/dp/r/pnioc/default",
    "mqt_topic_pn_wr_data": "ie/d/j/simatic/v1/
pnhs1/dp/w/pnioc/default",
    "mqt_topic_pn_status": "ie/s/j/simatic/v1/pnhs1/
status",
    "mqt_topic_pn_rcd_read_req": "ie/d/j/simatic/v1/
pnhs1/rcd/r/pnioc/default/req",
    "mqt_topic_pn_rcd_read_rsp": "ie/d/j/simatic/v1/
pnhs1/rcd/r/pnioc/default/rsp",
    "mqt_topic_pn_rcd_write_req": "ie/d/j/simatic/v1/
pnhs1/rcd/w/pnioc/default/req",
    "mqt_topic_pn_rcd_write_rsp": "ie/d/j/simatic/v1/
pnhs1/rcd/w/pnioc/default/rsp"
    },
    "name": "pnioc",
    "datapoints": [{
        "address": {
            "address_string": "%IB0"
        },
        "name": "tag001",
        "data_type": "USInt"
    },
    {...} # further datapoints
    ]
    }
    ],
    "databus_config": {
        "db_service_name": "ie-databus:1883",
        "username": "edge",
        "password": "*****"
    }
    }
    ]
    }

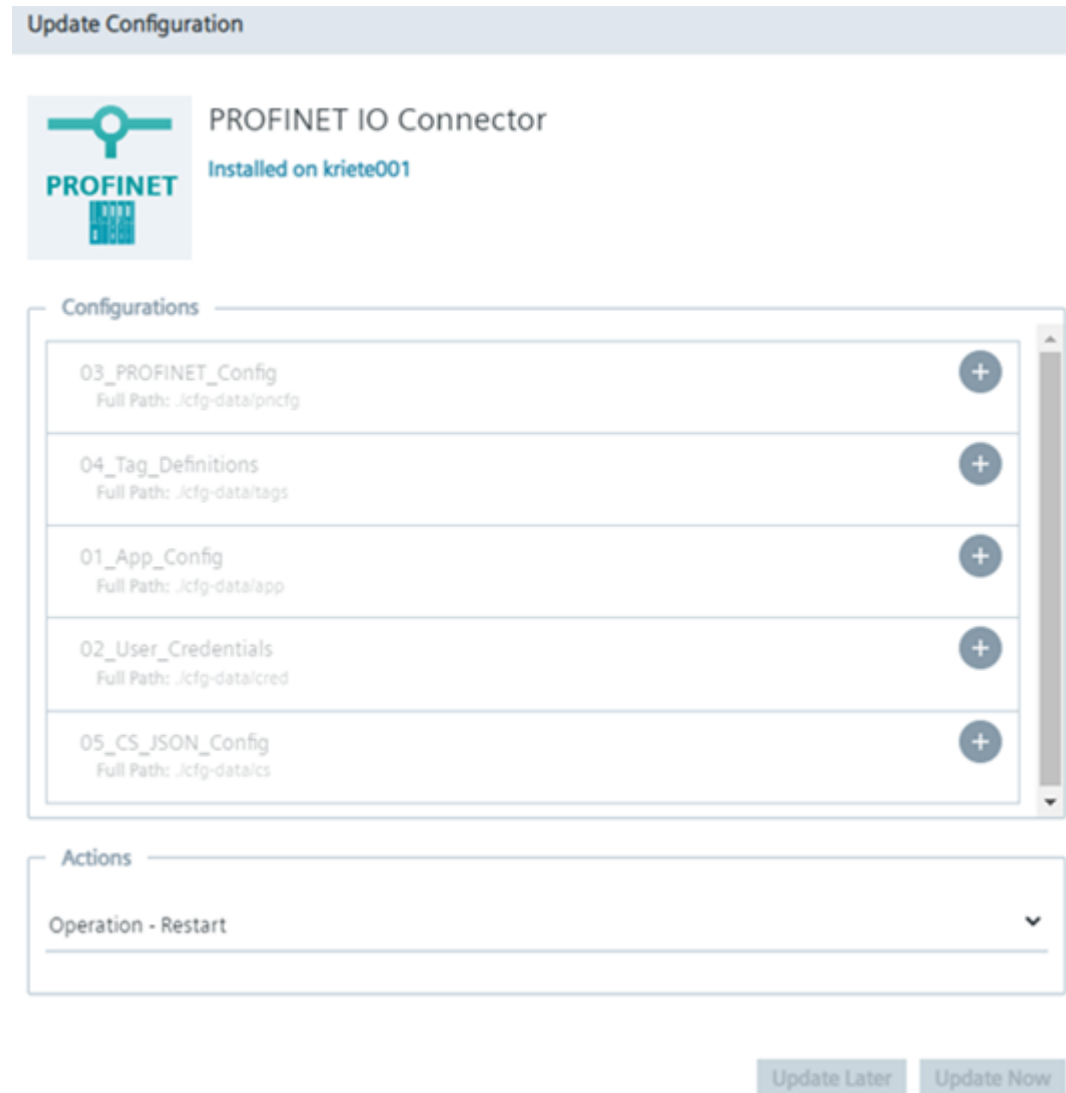
```

7.4.1.4 Configure PROFINET IO Connector using JSON file

To Configure PROFINET IO Connector follow the below steps:

1. Open Industrial Edge Management and go to "My Installed Apps".
2. Open the PROFINET IO Connector application. The PROFINET IO Connector details page opens.

3. Click "Update Configuration". A pop up appears as shown below:



4. Click the plus icon beside 05_CS_JSON_Config and upload the JSON file. You can ignore the remaining four files.
5. Click "Update Now".

The updated datasource will be displaying in the datasource table.

Note

User needs to restart the Connector to activate the updated configuration, select "Operation – Restart" from "Actions" drop down.

Deleting the JSON file

If you want to use four file configuration method for configuring datasources, you can delete the JSON file.

7.4 Configuring PROFINET IO Connector using JSON file

Prerequisites

JSON file must be configured in the Industrial Edge Management.

Procedure

To delete a JSON file, follow below steps:

1. Open Industrial Edge Management and go to "My Installed Apps".
2. Open the PROFINET IO Connector application. The PROFINET IO Connector details page opens.
3. Click "Delete Configuration". A pop-up opens displaying the available files, if JSON file is uploaded earlier it will be displayed.
4. Select the JSON file.
5. From the "Actions" drop down, select "Operation-Restart", and click the "Delete Now" button. The JSON file will be deleted.

7.4.2 Configuring PROFINET IO Connector as PN Device

7.4.2.1 Introduction

Prerequisite

- You must configure the PN Driver as Device in TIA Portal. For more information, refer Configuring PROFINET Driver as Device (Page 62).

Process

You can configure the PROFINET IO Connector as device using a single JSON file. Perform below steps to configure using JSON file:

1. Download the JSON file from PROFINET IO Connector or create a new configuration file based on the example as mentioned in Example of JSON Configuration for Device (Page 71). For more information on how to download JSON file, refer Downloading the JSON file (Page 67). This JSON file contains the details of user credentials, application setting, app configuration, and data points.
2. Update the file as necessary and then upload the JSON file to the PROFINET IO Connector. The JSON file format is easy to read, configure and edit the details. For more information on how to configure using JSON file, refer Configure PROFINET IO Connector using JSON file (Page 68).

Note

If you are using a JSON file format for configuration, any modifications made to the other four files will not impact the functionality of the PROFINET IO Connector.

7.4.2.2 Example of JSON Configuration for Device

Example of a JSON file is shown below:

```
{
  "configs": [{
    "$schema": "https://siemens.com/connectivity_suite/schemas/
profinet_device/1.0.0/config.json",
    "config": {
      "connections": [{
        "parameters": {
          "input_length": 8,
          "output_length": 256,
          "rt_task_cycle_us": 500000,
          "mqtt_oversampling_factor": 1,
          "mqtt_publish_binary": false,
          "mqtt_topic_pn_rd_metadata": "ie/m/j/simatic/v1/pnhs1/
dp",
          "mqtt_topic_pn_rd_data_bin": "ie/d/b/simatic/v1/
pnhs1/dp/r/pn-io-conn-device/default",
          "mqtt_topic_pn_rd_data_json": "ie/d/j/simatic/v1/
pnhs1/dp/r/pn-io-conn-device/default",
          "mqtt_topic_pn_wr_data": "ie/d/j/simatic/v1/
pnhs1/dp/w/pn-io-conn-device/default",
          "mqtt_topic_pn_status": "ie/s/j/simatic/v1/pnhs1/
status"
        },
        "name": "pn-io-conn-device",
        "datapoints": [{
          "address": {
            "address_string": "S1A0"
          },
          "name": "tag001",
          "data_type": "Int"
        },
        {...} # further datapoints
      ]
    }
  ]
},
"databus_config": {
  "db_service_name": "ie-databus:1883",
  "username": "edge",
  "password": "*****"
}
}
```

7.5 Configuring PROFINET IO Connector using Common Configurator

7.5.1 Introduction to Common Configurator

You can streamline the process of configuring the PROFINET IO Connector by using the Common Configurator, reducing the time and effort required for setup. This simplifies the task by presenting a well-organized interface that guides you through the necessary steps, ensuring that the connector is correctly configured to establish reliable and efficient communication between the PLCs and other apps in your IED.

Common Configurator fetches Profinet connector data through the Get Data tab and publishes the JSON data using the Databus.

Prerequisite

Following apps must be installed in the IED to configure using Common Configurator:

- Common Configurator
- IIH Registry Service

User interface of Common Configurator

To reach PROFINET IO Connector page follow the below steps:

1. Open Industrial Edge Device > Apps > Common Configurator.
2. Click "Get Data" > "Connector Configuration".
The Common Configurator page opens displaying all the available connectors.

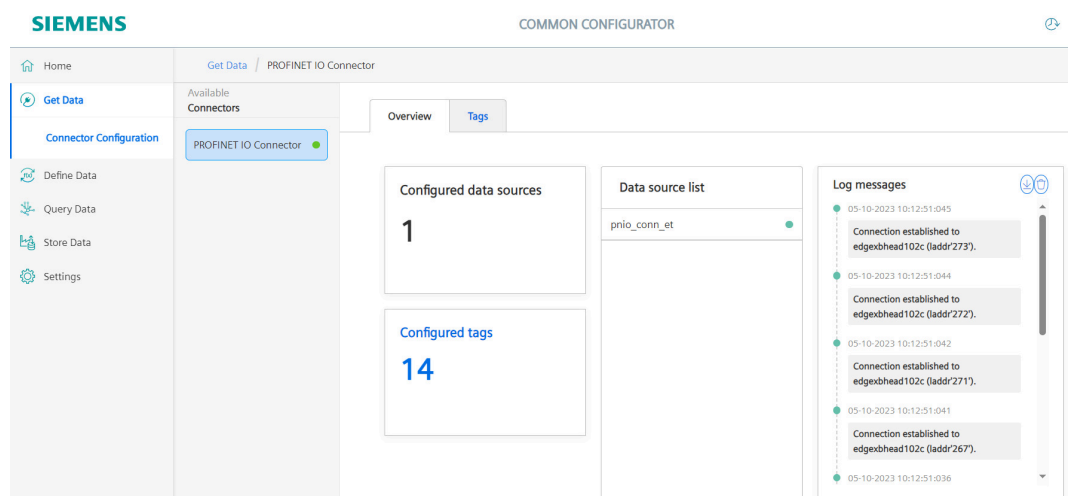


- Load configuration: To retrieve the connector configuration from the last deployment and revert any modifications made in the Common Configurator for all connectors.
 - Backup: To download all available connector configurations.
 - Restore: To import and restore downloaded configuration.
3. Select "PROFINET IO Connector".
PROFINET IO Connector page opens displaying the below tabs:

- Overview
- Tags

Overview

When you open PROFINET IO Connector, by default you will be in the "overview" tab as displayed below:



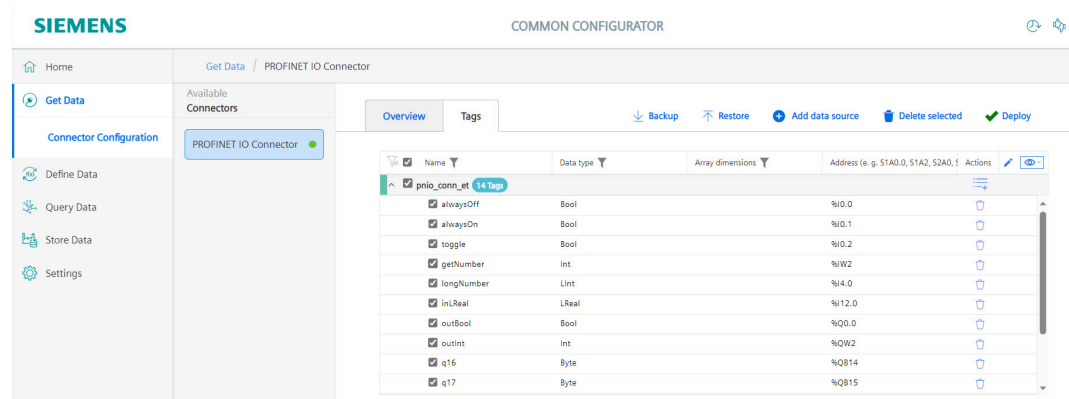
- Configured data sources- displays the number of data sources that are configured to PROFINET IO Connector.

7.5 Configuring PROFINET IO Connector using Common Configurator

- Tags configured- displays the total number of tags configured among all the data sources.
- Data source List- displays the list of datasources that are configured to PROFINET IO Connector.
- Log Messages- displays the status of the log.

Tags

Open PROFINET IO Connector, click "Tags" tab, the tag tab details are displayed as below:

**Note**

When you perform add, edit, or delete action on a datasource or a tag, you must click "Deploy" button to save the changes on the Industrial Edge Runtime of the PROFINET IO Connector.

See also

Prerequisites (Page 25)

7.5.2 Connecting the Common Configurator to Databus

You must insert the credentials for the Databus in the Common Configurator. The Common Configurator passes them to the PROFINET IO Connector.

The connector needs the credentials to connect to the Databus to be able to send and receive the data from it.

For connecting Common Configurator to Databus, follow the below steps:

1. Open Industrial Edge Device and go to "Apps" page.
The "Apps" page opens displaying all the available applications.
2. Click the "Common Configurator" application.
The "Common Configurator" page opens displaying available connectors.

3. Navigate to "Settings" > "Databus credentials".
The page opens displaying "Databus Credentials" details, and by default you will be navigated to the "Data Publisher settings" tab.

The screenshot shows the Siemens Common Configurator interface. On the left is a navigation menu with options: Home, Get Data, Define Data, Query Data, Store Data, Settings (selected), General, Databus Credentials (selected), OPC UA Server, Backup & Restore, and Language. The main area is titled 'Databus credentials'. It contains a 'Databus Service name' field with the value 'ie-databus:1883'. Below this are two tabs: 'Data Publisher settings' and 'Data Subscriber settings'. The 'Data Publisher settings' tab is active, showing a 'User name' field with the value 'edge' and a 'Password' field with masked characters. A 'Save' button is located at the bottom of the form.

4. In the "Databus Service Name" text field, enter the databus name. User can define the Databus service name. For more information about the naming, refer Name of the MQTT broker (Page 43).
5. In the "Username" text field enter the username.
6. In the "Password" text field, enter the password.
When you create a user in Databus, you define a username and password for the user. You must specify this username and password combination in the "User name" and "Password" text fields. Using these credentials, you can connect Common Configurator to Databus.
7. Click "Save".
8. Click "Data Subscriber settings".
"Data Subscriber settings" page opens.
9. Enter the same username and password and click "Save".
Common Configurator is now connected Databus.

Note

You must deploy the configuration in "Get Data" to save the databus credentials changes on the Industrial Edge Runtime of the PROFINET IO Connector. The deploy process is explained in the following chapters.

7.5.3 Configuring PROFINET IO Connector as a PN Controller

7.5.3.1 Overview

This chapter provides instructions on configuring the PROFINET IO Connector to function as a PROFINET Controller, allowing it to establish a connection with IO Devices.

Prerequisites

- You must configure the PN Driver as Controller in TIA Portal. For more information, refer Configuring PROFINET Driver as Controller (Page 55).
- Connect the Common Configurator to the Databus. For more information, refer Connecting the Common Configurator to Databus (Page 74).

7.5.3.2 Configuring data sources

Add data sources

A datasource is a field device which provides the data. You can configure, edit, and delete the datasources.

To configure a data source, follow the below steps:

1. Open Industrial Edge Device and go to "Apps" page.
The "Apps" page displays all the available applications.
2. Click the "Common Configurator" application.
The "Common Configurator" page opens, displaying available connectors.
3. Click "PROFINET IO Connector" tab.
"Overview" page of PROFINET IO Connector will be displayed.
4. Click "Tags" tab.
The tags details are displayed.

5. Click "Add Datasource" in the menu bar.
The "Add Datasource" dialog box is displayed as follows:

SIEMENS COMMON CONFIGURATOR

Home / Get Data / PROFINET IO Connector / Add Datasource

Get Data

Connector Configuration

Available Connectors

PROFINET IO Connector

Add Data source

Communication protocol *

PROFINET IO Controller

☒ Add manually ☐ Add from file

Name *

PROFINET Device configuration file * Select file

Real time task cycle in microseconds (e. g. 5000 or 1000)

10000

Oversampling factor for MQTT (e. g. 100 or 20)

50

☐ MQTT publishing mode (unchecked=json, checked=binary)

MQTT topic: read metadata

iel/m/j/simatic/v1/pnhs1/dp

MQTT topic: cyclic read data (binary)

iel/d/b/simatic/v1/pnhs1/dp/rl-<ConnectionNam

« Collapse

7.5 Configuring PROFINET IO Connector using Common Configurator

6. Update the following fields as per requirement, if you have not entered any values, default value will be updated as shown below:

Field Name	Default	Details
PLC Type*	None	The only selectable option is "PROFINET IO".
Name*	None	Name of the datasource.
PROFINET Configuration File*	None	Select the XML file from TIA Portal. For more details refer PROFINET Configuration in TIA Portal (Page 55).
Realtime task cycle	10000	Defines the cycle time of the real-time task. It reads the process image INPUT as record.
Oversampling factor for MQTT	50	Number of recorded records that are put together to one MQTT topic.
MQTT publish mode	json	Publish data as json file format or binary payload.
MQTT topic: read metadata	ie/m/j/simatic/v1/pnhs1/dp	MQTT topic for publishing the "Cyclic PN IO-Data Read" metadata.
MQTT topic: Cyclic read data (binary)	ie/d/b/simatic/v1/pnhs1/dp/r/<Connection_Name>/default	MQTT topic for publishing the "Cyclic PN IO-Data Read" data as BINARY payload.
MQTT topic: cyclic read data (json)	ie/d/j/simatic/v1/pnhs1/dp/r/<Connection_Name>/default	MQTT topic for publishing the "Cyclic PN IO-Data Read" data as JSON payload.
MQTT topic: write data	ie/d/j/simatic/v1/pnhs1/dp/w/<Connection_Name>/default	MQTT topic for subscribing the "PN IO-Data Write" commands.
MQTT topic: status	ie/s/j/simatic/v1/pnhs1/status	MQTT topic for publishing "PN Status".
MQTT topic: record read request	ie/d/j/simatic/v1/pnhs1/rcd/r/<Connection_Name>/default/req	MQTT topic for subscribing the "PN Data Record READ" request.
MQTT topic: record read response	ie/d/j/simatic/v1/pnhs1/rcd/r/<Connection_Name>/default/rsp	MQTT topic for publishing the "PN Data Record READ" response.
MQTT topic: record write request	ie/d/j/simatic/v1/pnhs1/rcd/w/<Connection_Name>/default/req	MQTT topic for subscribing the "PN Data Record WRITE" request.
MQTT topic: record write response	ie/d/j/simatic/v1/pnhs1/rcd/w/<Connection_Name>/default/rsp	MQTT topic for publishing the "PN Data Record WRITE" response.

* Indicates that all the fields are mandatory.

Note

The <Connection_Name> refers to the 'Name' parameter mentioned in the table above.

7. Click "Save".
The data source is added and displayed in "Datasource" table.
8. Click "Deploy"
The changes are deployed to the runtime.

Note

Only one datasource should be deployed at a time. When multiple datasources are deployed simultaneously, the configuration becomes invalid, and the application will fail to establish a connection.

Edit data sources

To edit a datasource, follow the below steps:

1. Open PROFINET IO Connector.
The PROFINET IO Connector "Overview" page is displayed.
2. Click the "Tags" tab.
The tag details are displayed.
3. Click the icon  under "Actions" column on the datasource.
A pop-up menu appears.
4. Click "Edit Datasource".
The "Edit Datasource" dialogue box is displayed.
5. Modify the required changes.
6. Click "Save".
The datasource is modified and displayed in the "Datasource" table.

Delete data sources

To delete a datasource, follow the below steps:

1. Open PROFINET IO Connector.
The PROFINET IO Connector "Overview" page is displayed.
2. Click the "Tags" tab. The tag details are displayed.
3. Click the icon  under "Actions" column on the datasource.
A pop-up menu appears.
4. Click "Delete data source".
A pop up message appears to confirm the action.
5. Click "Delete".
The data source is deleted form the "Data source" table.

User can also delete the data sources by following the below steps:

1. Open PROFINET IO Connector and got to "Tags".
2. Select the data sources that you want to delete.

3. Click "Delete Selected" in the upper right corner.
A pop up message appears to confirm the action.
4. Click "Delete".
The data sources are deleted from the "Data source" table.

7.5.3.3 Configuring tags

Add tags

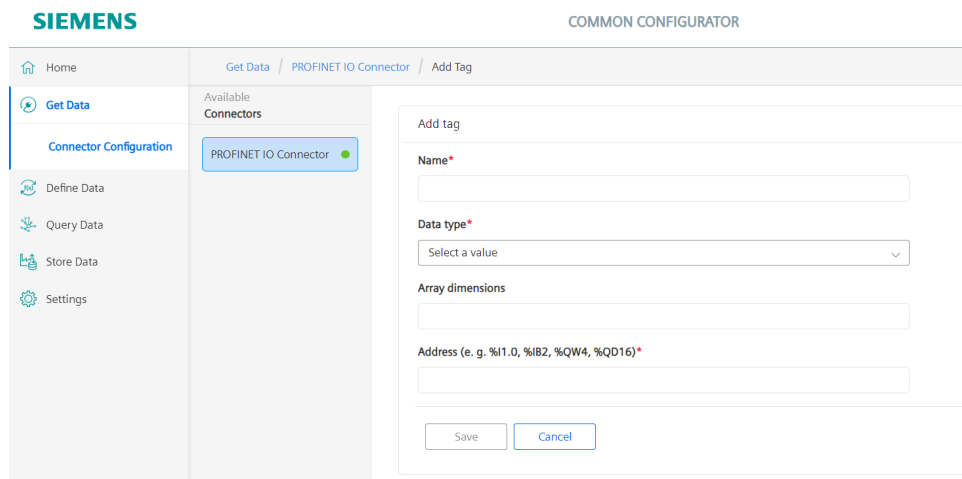
You can add a data point or tag from a data source in the PROFINET IO Connector. Datapoints are referred as Tags.

Prerequisite

- PROFINET IO Connector must be running
- A data source must be available

To add a tag to a datasource, follow the below steps:

1. Open PROFINET IO Connector.
The PROFINET IO Connector "Overview" page will be displayed.
2. Click the "Tags" tab. The tags details are displayed.
3. Click the icon  under "Actions" column on the data source.
A pop-up menu appears.
4. Click "Add Tag".
The "Add Tag" dialog box is displayed based on the selected data source as follows:



SIEMENS COMMON CONFIGURATOR

Home / Get Data / PROFINET IO Connector / Add Tag

Connector Configuration

Available Connectors: PROFINET IO Connector

Add tag

Name*

Data type*

Array dimensions

Address (e. g. %I1.0, %IB2, %QW4, %QD16)*

5. Update the following fields:

Filed name	Details
Name	Name of the tag.
Data type	Specifies the data point type. For more information about data types refer Data types (Page 43).
Array Dimensions	Not in use.
Address	Defines the address of the data point in the controller or on the server.

*Indicates that all the fields are mandatory.

Note

The datatype and address must match the TIA Portal hardware configuration.

6. Click "Save".
Tag is added to the data source.

Edit tags

To edit tags, follow the below steps:

1. Open PROFINET IO Connector.
The PROFINET IO Connector "Overview" page will be displayed.
2. Click the "Tags" tab.
The tags details are displayed.
3. Click the icon  on the right side of "Actions" tab.
Tag details with text fields will be displayed.
4. Select the required tag, and modify the changes in the required text fields.
5. Click on the icon .
The data point is modified and displayed in the "Data source" table.

Delete tags

To delete a tag, follow the below steps:

1. Open PROFINET IO Connector.
The PROFINET IO Connector "Overview" page appears.
2. Click the "Tags" tab.
The tags details appears.
3. Click the icon  on the right side of the tag that you want to delete.
A pop up message appears to confirm your action.
4. Click "Delete".
The tag is deleted from the data source.

You can also delete a tag by following the below steps:

1. In the "Tags" tab, select a tag or multiple tags that you want to delete.
2. Click "Delete Selected" in the upper right corner.
The selected tag will be deleted.

7.5.4 Configuring PROFINET IO Connector as a PN Device

7.5.4.1 Overview

This chapter provides instructions on configuring the PROFINET IO Connector to function as a PROFINET device, allowing it to establish a connection with a controller.

Prerequisites

- You must configure the PN Driver as Device in TIA Portal. For more information, refer [Configuring PROFINET Driver as Device \(Page 62\)](#).
- Connect the Common Configurator to the Databus. For more information, refer [Connecting the Common Configurator to Databus \(Page 74\)](#).

7.5.4.2 Configuring data sources

Add data sources

A datasource is a field device which provides the data. You can configure, edit, and delete the datasources.

To configure a data source, follow the below steps:

1. Open Industrial Edge Device and go to "Apps" page.
The "Apps" page displays all the available applications.
2. Click the "Common Configurator" application.
The "Common Configurator" page opens, displaying available connectors.
3. Click "PROFINET IO Connector" tab.
"Overview" page of PROFINET IO Connector will be displayed.
4. Click "Tags" tab.
The tags details are displayed.

- Click "Add Datasource" in the menu bar.
The "Add Datasource" dialog box is displayed as follows:

The screenshot shows the Siemens Common Configurator interface. The top bar includes the Siemens logo and the title 'COMMON CONFIGURATOR'. Below this is a breadcrumb trail: 'Home' / 'Get Data' / 'PROFINET IO Connector' / 'Add Datasource'. The left sidebar contains a 'Get Data' section with a 'Connector Configuration' sub-section, and a list of options: 'Define Data', 'Query Data', 'Store Data', and 'Settings'. The main area is titled 'Available Connectors' and lists 'PROFINET IO Connector' with a green status indicator. To the right, the 'Add Data source' configuration panel is displayed. It includes a 'Communication protocol' dropdown set to 'PROFINET IO Device', radio buttons for 'Add manually' (selected) and 'Add from file', a 'Name' text field, dropdowns for 'Slot 1 INPUT length' and 'Slot 2 OUTPUT length' (both set to 'Select value'), a 'Real time task cycle in microseconds' text field set to '10000', an 'Oversampling factor for MQTT' text field set to '1', a checkbox for 'MQTT publishing mode' (unchecked), and an 'MQTT topic' text field set to 'read metadata'.

6. Update the following fields as per requirement, if you have not entered any values, default value will be updated as shown below:

Field Name	Default	Details
PLC Type*	None	The only selectable option is "PROFINET IO".
Name*	None	Name of the datasource.
Input length	None	The length of the INPUT slot configured in TIA Portal.
Output length	None	The length of the OUTPUT slot configured in TIA Portal.
Realtime task cycle	10000	Defines the cycle time of the real-time task. It reads the process image INPUT as record.
Oversampling factor for MQTT	1	Number of recorded records that are put together to one MQTT topic.
MQTT publish mode	json	Publish data as json file format or binary payload.
MQTT topic: read metadata	ie/m/j/simatic/v1/pnhs1/dp	MQTT topic for publishing the "Cyclic PN IO-Data Read" meta-data.
MQTT topic: Cyclic read data (binary)	ie/d/b/simatic/v1/pnhs1/dp/r/<Connection_Name>/default	MQTT topic for publishing the "Cyclic PN IO-Data Read" data as BINARY payload.
MQTT topic: cyclic read data (json)	ie/d/j/simatic/v1/pnhs1/dp/r/<Connection_Name>/default	MQTT topic for publishing the "Cyclic PN IO-Data Read" data as JSON payload.
MQTT topic: write data	ie/d/j/simatic/v1/pnhs1/dp/w/<Connection_Name>/default	MQTT topic for subscribing the "PN IO-Data Write" commands.
MQTT topic: status	ie/s/j/simatic/v1/pnhs1/status	MQTT topic for publishing "PN Status".

* Indicates that all the fields are mandatory.

Note

The <Connection_Name> refers to the 'Name' parameter mentioned in the table above. This 'Name' must precisely match the converted PROFINET device name configured in TIA Portal.

7. Click "Save".
The data source is updated and displayed in "Datasource" table.
8. Click "Deploy"
The changes are deployed to the runtime.

Note

Only one datasource should be deployed at a time. When multiple datasources are deployed simultaneously, the configuration becomes invalid, and the application will fail to establish a connection.

Note

Refer the following sections for edit data source, delete data sources and managing data points:

- For edit data source, refer Edit data source (Page 79).
 - For delete data source, refer Delete data source (Page 79).
 - For Configuring tags, refer Configure tags (Page 80).
-

Working with PROFINET IO Connector

The PROFINET IO Connector offers the following different main functionalities:

- High-speed recording of PROFINET IO-Data (cyclic read)
- Write PROFINET IO-Data
- Read PROFINET Data Records
- Write PROFINET Data Records

The above tasks require specific MQTT topics and payloads as described in the following sections.

- Cyclic Read PROFINET IO Data
 - PROFINET IO Read - Metadata
 - PROFINET IO Read - Binary Payload
 - PROFINET IO Read - JSON Payload
- PROFINET IO Write
- PROFINET Data Record Write
- PROFINET Data Record Read

Note

Advanced features of PROFINET IO Connector

Besides the core functionality of the PROFINET IO Connector described in this manual, the driver offers additional advanced features like Profinet status information and Profinet events. These advanced features are subject to change in the upcoming version of the PROFINET IO Connector (change in format definition, functional range, and so on). Therefore, this manual does not provide detailed information about these features. You must not use these advanced features in productive applications.

8.1 Cyclic Read PROFINET IO Data

8.1.1 MQTT Payload for Cyclic Read PROFINET IO Data

The PROFINET IO Connector allows you to collect the data from the PROFINET network in a high time-resolution and provide this data to the Databus for processing by other apps. An important function is to store the data in IIH Essentials. IIH Essentials is a central data source for several other apps.

The cyclic data is provided in a raw Data Format (binary payload) when highest time-resolution is needed (to focus on performance).

The PROFINET IO Connector and Cyclic Read PROFINET IO Data support JSON as data format.

Note

The overhead due to the data format JSON is approximately factor 10. This means the number of bytes sent through the network is approximately 10 times greater than binary format.

Therefore, for higher time-resolution with big number IO data, the binary format is a better choice. Additionally, the parsing of JSON data needs more CPU power than reading the raw Data Format.

The information like structure of the data is used to process the raw Data Format (binary payload). This is described in the metadata. The content of the metadata depends on the tag definitions. When no tags are defined, the metadata looks different than with defined tags.

8.1.2 PROFINET IO Read - Metadata

The MQTT topic of the Metadata for Cyclic Read PROFINET IO Data is as given in the below links:

- For Controller Configuration, refer step no. 6 of Add data sources (Page 76).
- For Device Configuration, refer step no. 6 of Add data sources (Page 82).

The structure of the payload depends on the PROFINET configuration. When the PROFINET IO Connector starts, it analyzes the PN config to build the structure of the binary or json payload.

Note

The structure of the metadata depends on if the optional tag definition file is provided or not.

Remarks on properties

The following remarks apply to all configuration options:

pubTopic

The topic contains the MQTT topic. When the data is published as JSON, it contains the corresponding JSON topic. When the data is published exclusively in binary format (raw format), it contains the Binary topic. When the data is published in binary and JSON, it contains the JSON topic.

The following table describes the content of metadata 'dataPoints\topic' depending on the type of published data:

Publish JSON	Publish Binary	Topic in metadata
-	active	Binary topic
active	-	JSON topic
active	active	JSON topic

publishType

Depending on the format (JSON or Binary) and the number of published PN cycles in one MQTT message, the publishType has different values as described in the following table:

publishType	Description
bulk	One message includes only one PN cycle data and the format is JSON.
timeseries	One message includes several PN cycles (records), and the format is JSON.
binarytimeseries	One message includes one or more PN cycles in Binary format.

Metadata Read PN IO-Data - No Tags Defined

It represents the setup of all PN submodules and their IO memory size when tags are not defined. The property id is used as PN IO address as displayed in the TIA Portal project.

The metadata describes the content of one record in the binary payload.

An example for Metadata JSON (no tags defined) is as follows:

```
{
  "seq":0,                                # sequence of MQTT message
  "hashVersion":2106496482, # hash to refer in 'PN IO-Data Read'
  "connections":[
    {
      "name":"profinetxadriv4933", # PN name of the controller
      "type":"pn",                  # type is fixed to 'pn'
      "dataPoints":[
        {
          "name":"default",
          "topic":"ie\\d\\b\\simatic\\v1\\pnhs1\\dp\\r\\", #
          "pubTopic":"ie\\d\\j\\simatic\\v1\\pnhs1\\dp\\w\\", #
          "publishType":"timeseries", # alternatively: 'bulk' or
          "dataPointDefinitions":[
            {
              "name":"stat1_slot1", # generated name
              "id":"0",             # IO memory address
              "dataType":"Byte",    # when no tags defined,
```

8.1 Cyclic Read PROFINET IO Data

```

always 'array of Byte'
    "valueRank":1,                # one-dimensional array
    "arrayDimensions":[8],        # 8 bytes from this IP
point
    "acquisitionCycleInMs":"2",   # acquisition time in [ms]
    "acquisitionMode":"CyclicOnContinuous" # continuously
recording
    },
    {
        "name":"stat3_slot2",    # generated name
containing station and slot number
        "id":"8",                # IO memory address (shown
in TIA Portal)
        "dataType":"Byte",
        "valueRank":1,
        "arrayDimensions":[1],   # this IO-Point provides
only a single byte
        "acquisitionCycleInMs":2, # acquisition time in [ms]
        "acquisitionMode":"CyclicOnContinuous"
    },

```

Metadata Read PN IO-Data - With Tags

The metadata offers the connection between the PROFINET view (for example, stations, modules, and so on) and the variables when tags are defined (tag definition file). The metadata includes the needed information to be able to analyze the raw data (Binary payload).

The metadata covers both Read and Write functions.

- Read: For analyzing the binary payload (raw data format).
- Write: To analyze what tags can be written.

An example for Metadata JSON with defined tags is as follows:

```

{
    "seq":0,                # sequence number of the MQTT message
    "hashVersion":688894030, # hash as reference to the 'PN IO-Data
Read' data
    "connections":[
        {"name":"profinetxdriver4933", # PN name of controller (see
TIA Portal)
        "type":"pn",                # fixed to 'pn'
        "dataPoints":[
            {"name":"default",
            "topic":"ie/d/b/simatic/v1/pnhs1/dp/r/connection1/default/", # topic to publish read PN IO-Data (app config file)
            "pubTopic":"ie/d/j/simatic/v1/pnhs1/dp/w/", #
subscribed topic for WRITE (app config file)
            "publishType":"timeseries", # alternatively: 'bulk' or
'binarytimeseries' *2)
            "dataPointDefinitions":[
                # --- Example for READ tag ---
                {"name":"Digital1",                # name of the tag

```

```

        "id": "101",                                # ID of the tag
from tag definition file
        "dataType": "Byte",                          # data type of
the tag
        # "valueRank": -1,                          # default value
-1 is not published
        # "arrayDimensions": null,                  # currently no
arrays are supported
        "dataPointAddr": "S0A0",                    # address inside
binary RAW payload
        "acquisitionCycleInMs": "2",                # acquisition
cycle in [ms]
        "acquisitionMode": "CyclicOnContinuous",    # continuously
recording
        # "accessMode": "r"                          # default "r" not
published
    },
    # --- Example for WRITE tag ---
    { "name": "Bit1",
      "id": "201",
      "dataType": "Bool",
      "accessMode": "w"                               # access mode 'w' (Write)
    },

```

The following table lists the properties of tags metadata:

Property	Description
id	Displays the tag id in the tag definition file.
name	Defines the tag name in the tag definition file.
dataPointAddr	Describes the tag information stored in the raw data.
acquisitionCycleInMs	Defines the update ratio (for example, 2.000 milliseconds).
valueRank	Describes the information for array items (according to OPC UA: -1 = scalar, 1 = 1 dim array, 2 = 2 dim array, ...)
arrayDimensions	Describes the information for array items (for example, size of every dimension, array of 'valueRank' items).
accessMode	Describes the access mode 'w', 'r', 'rw'. 'r' is the default value (if not configured).

IO-Data Point Address in Raw Data

The metadata includes the defined tags and their data types. To be able to analyze the Binary payload, it provides the index of the input submodule and the byte offset. These values correspond to the metadata. This needed information (IO Data Point index, Byte Offset, Bit Offset) as one single additional property is the 'dataPointAddr'.

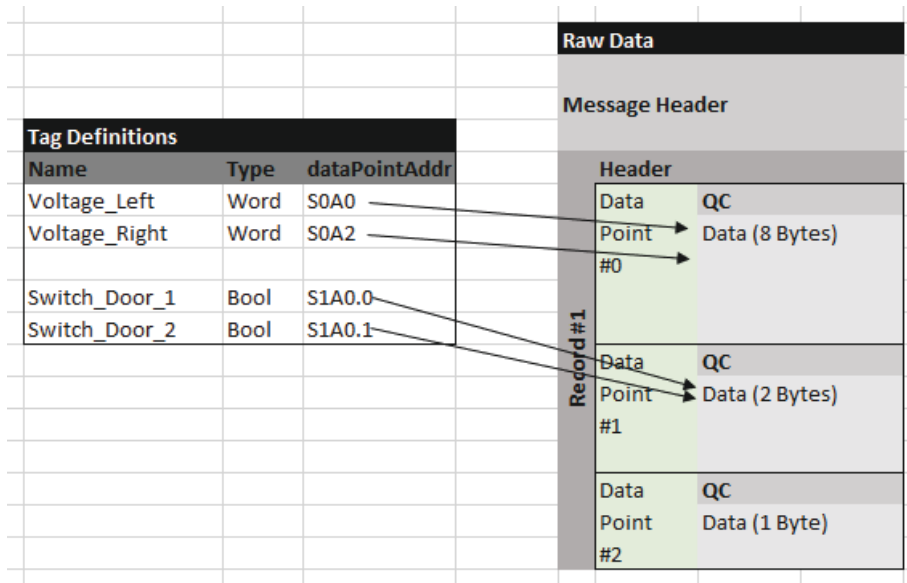
You cannot use the original S7 address syntax as an IO address is not provided. You can get only the address inside one IO Data Point (submodule) specific for the raw data format of the Binary payload.

8.1 Cyclic Read PROFINET IO Data

The following table provides some examples for IO-Data-Point Addresses:

Address in Raw Data	Description
S0A5.1	Submodule Index 0, Offset inside Submodule Byte 5, Bit 1
S2A0	Submodule Index 2, Offset inside Submodule Byte 0

The following image depicts the principle of the tag definition and its dataPointAddress:



Support of Arrays

Note

The current version of PROFINET IO Connector does not support array types.

The OPC UA has following properties to describe array types:

- **valueRank**: Information for array items: -1 = scalar, 1 = 1 -dim array, 2 = 2 dim array, ...
- **arrayDimensions**: Information for array items: size of every dimension, array of 'valueRank' items.

Examples of Array Items:

- "valueRank":-1, "arrayDimensions": null > scalar value
- "valueRank":2, "arrayDimensions": [3,3] > 3x3 array

8.1.3 PROFINET IO Read - Binary Payload

The raw data format (Binary payload) contains one or more records as described in the metadata. The application parameter MQTT oversampling factor defines the number of records provided with one payload.

The MQTT topic of the Raw Data of Cyclic Read PN IO-Data is given in the below links:

The raw data format (Binary payload) is not affected if tags are defined, or the tags are not defined. The MQTT topic of the raw data for Cyclic Read PN IO-Data is given in the below links:

- For Controller Configuration, refer step no. 6 of Add data sources (Page 76).
- For Device Configuration, refer step no. 6 of Add data sources (Page 82).

Note

The maximum size of a single binary message is limited by the internal buffer of the PROFINET IO Connector, with a maximum size of 264 KB.

The following image shows the principle structure of the raw data format (Binary payload):

8.1 Cyclic Read PROFINET IO Data

	Offset	Type	Len	Meaning	Remark
Msg Head	0	uint32_t	4	MQTT type / version	Identify this type of MQTT payload
	4	uint32_t	4	magic number	Detect the payload header
	8	uint32_t	4	Metadata sequence number	match binary payload and metadata
	12	uint16_t	2	size of a single record in bytes	
	14	uint16_t	2	total number of records in this message	
	16	uint32_t	4	Acquisition cycle in [100 ns] = [0,1 us]	Acquisition cycle - Time distance between two records
	20	uint64_t	8	timestamp in [100 ns] = [0,1 us] of the last record in the message	
	28	uint16_t	2	Message Counter - incremented for each send MQTT message	Detect missing MQTT telegrams
	30	uint16_t	2	Feature Flags: each bit enables another feature (e.g. additional fields in the header of the record) Bit 0=1 => each record has its own timestamp Bit 1=1 => each record has its own record counter Bit 2=1 => each IO point provides the PN sequence cnt.	
	32	uint16_t	2	Number of IO points in one record	
	34	uint16_t	2 - 2p	Array of length-values of all IO points of one record	
Record #1	Header	uint64_t	8	optional: timestamp in [100 ns] = [0,1 us]	
		uint16_t	2	optional: Record Counter - incremented for each record	Detect missing records, e.g. internal FiFo overflow
	Data Point #1 Header	uint8_t	1	Quality IO Point #1	0xC0 = good, 0x00 = bad
		uint16_t	2	optional: PN Sequence Counter - directly from Profinet field "PN sequence counter"	Detect missing PN telegrams Attention! The PNDrv API does not provide the PN sequence counter!!
	Data		1 - n	data IO Point #1	
	Data Point #2 Header	uint8_t	1	Quality IO Point #2	0xC0 = good, 0x00 = bad
		uint16_t	2	optional: PN Sequence Counter - directly from Profinet field "PN sequence counter"	Detect missing PN telegrams Attention! The PNDrv API does not provide the PN sequence counter!!
	Data		1 - n	data IO Point #2	
Record #2	Header	uint64_t	8	optional: timestamp in [100 ns] = [0,1 us]	
		uint16_t	2	optional: Record Counter - incremented for each record	Detect missing records, e.g. internal FiFo overflow
	Data Point #1 Header	uint8_t	1	Quality IO Point #1	0xC0 = good, 0x00 = bad
		uint16_t	2	optional: PN Sequence Counter - directly from Profinet field "PN sequence counter"	Detect missing PN telegrams Attention! The PNDrv API does not provide the PN sequence counter!!
	Data		1 - n	data IO Point #1	
	Data Point #2 Header	uint8_t	1	Quality IO Point #2	0xC0 = good, 0x00 = bad
		uint16_t	2	optional: PN Sequence Counter - directly from Profinet field "PN sequence counter"	Detect missing PN telegrams Attention! The PNDrv API does not provide the PN sequence counter!!
	Data		1 - n	data IO Point #2	

8.1.4 PROFINET IO Read - JSON Payload

The PROFINET IO Connector supports JSON format. You can send the Cyclic Read PN IO-Data as JSON additionally or alternatively. The performance is reduced due to the big overhead of JSON compared to Binary format.

There is a separate MQTT topic where the PROFINET IO Connector publishes the JSON data. The default MQTT topic for JSON payload of Cyclic Read PN IO-Data is as follows:

For Controller Configuration, refer step no. 6 of Add data sources (Page 76).

For Device Configuration, refer step no. 6 of Add data sources (Page 82).

The structure of the JSON payloads depends on the oversampling activated or deactivated and whether the tags are defined or not defined.

This leads to four different formats of the JSON payload as follows:

- no tags defined, no oversampling (Page 96)
- no tags defined, oversampling activated (Page 96)
- with tags, no oversampling (Page 97)
- with tags, oversampling activated (Page 98)

When tags are defined (tag definition file), the JSON payload contains all defined tags as typed values.

Note

JSON supports only a limited number of data types, for example, number and string.

This leads to the following implementation:

- Bits are displayed as "1-Bit-Values" with value '1' or '0'.
 - Floats are displayed in exponential format.
 - All other numbers are displayed as decimal values.
-

Size of JSON messages

To prevent an overload for the Databus, the maximum size of one sent JSON message is limited in the PROFINET IO Connector. The maximum size of a JSON message is limited to 512 KB.

You have to consider the needed size according your configuration. You can estimate the size of the JSON message with this formula:

$$\text{<aprox. JSON msg len>} = \text{<number of tags>} \times \text{<oversampling factor>} \times \text{<40 byte>}$$

The length of JSON message must be <480 KB.

Depending on the datatype you can estimate the needed characters for one tag approx. 40 bytes, e.g. {"id": "111", "val": 6.751120e-01, "qc": 3}.

Example for JSON size

If you have defined 50 tags in sum and your oversampling factor is 100, you can expect a JSON size of approx.:

8.1 Cyclic Read PROFINET IO Data

50 tags x 100 oversampling factor x 40 byte = 200.000 bytes.

You can check the current size of your JSON message in the log file of the PROFINET IO Connector, e.g.:

PN Read: size 1st cyclic JSON payload is '62683' bytes

8.1.4.1 JSON Payload (no tags defined, no oversampling)

This JSON payload contains only one set of PN data of one PROFINET cycle. As no tags are defined, the raw data of all PROFINET modules is an array of byte as follows:

```
{
  "seq": 10,                                # sequence number of
the MQTT message                             #
  "mdHashVer": 2106496482,                  # hash to reference to
metadata                                     #
  "ts": "2020-12-14T07:21:04.236435Z",      # timestamp of this
PROFINET cycle                             #
  "vals": [                                  # array of IO-Data
points of this PN cycle                     #
    {                                        # IO-Data Point 1 -----
      "id": "0",                            # PROFINET address
      (see TIA Portal)
      "val": [3,208,252,85,191,112,104,168], # IO-Data (array of
byte)                                       #
      "qc": 3                              # quality code
    },
    {                                        # IO-Data Point 2 -----
      "id": "8",                            # PROFINET address
      (see TIA Portal)
      "val": [0],                          # IO-Data (only one
byte for this IO Data Point)              #
      "qc": 3                              # quality code
    }
  ]
}
```

8.1.4.2 JSON Payload (no tags defined, oversampling activated)

Some records of 'PN cycles' are provided when oversampling is activated. As no tags are defined, the raw data of all PROFINET modules is an array of byte as follows:

```
{
  "seq": 4,                                # sequence number of
the MQTT message                             #
  "mdHashVer": 1729890178,                  # hash to reference
to metadata                                  #
  "records": [                              # array of records
(PN cycles)                                #
    {                                        # Data Record No. 1
-----
      "ts": "2020-12-14T07:22:43.016719Z", # timestamp of this
```



```

PROFINET cycle
    "rseq": 14,                                # record sequence
number (recorded PN cycle)
    "vals": [                                  # array of IO-Data
points of this PN cycle
    { "id": "0",                               # PROFINET address
(see TIA Portal)
    "val": [235,51,2,134,63,37,106,59], # IO-Data (array of
byte)
    "qc": 3                                   # quality code
    },
    { "id": "8", "val": [0], "qc": 3
    }
    ],
    {                                           # Data Record No. 2 ----
    "ts": "2020-12-14T07:22:43.516716Z", # timestamp of this
PROFINET cycle
    "rseq": 15,                                # record sequence number
(recorded PN cycle)
    "vals": [                                  # array of IO-Data
points of this PN cycle
    { "id": "0", "val": [203,51,1,144,62,204,142,207], "qc":
3 },
    { "id": "8", "val": [0], "qc": 3 }
    ]
    }
]
}

```

8.1.4.3 JSON Payload (with tags, no oversampling)

This JSON payload contains only one set of PN data of one PROFINET cycle. When the tags are defined, the payload includes typed values of all defined tags as follows:

```

{
    "seq":1,                                    # sequence number of
the MQTT message
    "mdHashVer":299652349,                    # hash to reference
to metadata
    "ts":"2020-12-11T07:26:05.329254Z",      # timestamp of this
PROFINET cycle
    "vals":[                                  # array of IO-Data
points of this PN cycle
    { "id":"101","val":1,"qc":3},              # typed value incl.
quality code
    { "id":"111","val":6.751120e-01,"qc":3}   # typed value incl.
quality code
    ]
}

```

8.1.4.4 JSON Payload (with tags, with oversampling)

Some records of 'PN cycles' are provided when oversampling is activated. And, when tags are defined, the payload includes typed values of all defined tags as follows:

```
{
  "seq": 7,                                # sequence number of the
MQTT message
  "mdHashVer": 688894030,                  # hash to reference to
metadata
  "records": [{                             # array of records (PN
cycles)
    "ts": "2020-12-14T07:25:56.245946Z",   # timestamp of this
PROFINET cycle
    "rseq": 20,                             # record sequence number
(recorded PN cycle)
    "vals": [{                             # array of IO-Data points
of this PN cycle
      "id": "101",                          # id from 'tag definition
file'
      "val": 0,                             # ATTENTION! Boolean
values are shown as number 0/1
      "qc": 3                              # quality code
    },
    { "id": "109",                          # id from 'tag definition
file'
      "val": 244,                           # typed value of this tag
      "qc": 3                              # quality code
    },
    { "id": "111",                          # id from 'tag definition
file'
      "val": -8.871680e-01,                 # typed value of this tag
      "qc": 3                              # quality code
    }
  ]
},
{                                           # Data Record No. 2
  "ts": "2020-12-14T07:25:56.745946Z",   # timestamp of this
PROFINET cycle
  "rseq": 21,                             # record sequence number
(recorded PN cycle)
  "vals": [                               # array of IO-Data points
of this PN cycle
    { "id": "101", "val": 1, "qc": 3 },
    { "id": "109", "val": 244, "qc": 3 },
    { "id": "111", "val": -7.175437e-01, "qc": 3 }
  ]
}]
}
```

8.2 PROFINET IO Write

In the current version, there is an option to write IO-Data via PROFINET.

In this version, the smallest unit which can be accessed by PROFINET IO-Data Write is a PROFINET submodule. For example, the size of the provided raw data must fit the size of this submodule. In case, an ET200SP 8 DO module is used, you must write the complete byte. You cannot access a single digital output.

There is a separate MQTT topic in which the PROFINET IO Connector is subscribed. It is configured in the 'app config file'. The default MQTT topic for PROFINET IO-data Write is as given in the below links:

- For Controller Configuration, refer step no. 6 of Add data sources (Page 76).
- For Device Configuration, refer step no. 6 of Add data sources (Page 82) .

MQTT Payload for PROFINET IO-Data Write without Defined Tags

If a tag definition file is not provided, the PROFINET IO Connector expects array of byte (raw data format) while writing at a PROFINET submodule.

An example for JSON payload for PROFINET IO-Data Write is as follows:

```
{ "seq":1,          # sequence number of the MQTT message (not processed
  by the app)
  "vals":[
    { "id":"0",    # PROFINET IO-address of the submodule to be
      written
      "val":[175,254,56,40,4,85,0,0] # byte array of raw data to be
      written at this submodule
    },
    { "id":"8",    # PROFINET IO-address of the submodule to be
      written
      "val":[0,0] # byte array of raw data to be written at this
      submodule
    }
  ]
}
```

MQTT Payload for PROFINET IO-Data Write with Defined Tags

If a tag definition file is provided, the PROFINET IO Connector expects tags together with its values. You must also comply with the limited data type support of JSON (Boolean, integer, and double). The PROFINET IO Connector verifies the data type according to the tag definition file.

Note

JSON supports only a limited number of data types, for example, number and string.

This leads to the following implementation:

- Bits are shown as "1-Bit-Values" with value '1' or '0'.
 - Floats are shown in exponential format.
 - All other numbers are shown as decimal values.
-

An example for JSON payload for PROFINET IO-Data Write is as follows:

```
{ "seq":1,                # sequence number of the MQTT message
  (not processed by the app)
  "vals":[
    { "id":"201",          # tag id according the tag definition file
      "val":128            # value of this tag
    },
    { "id":"202",          # tag id according the tag definition file
      "val":-9.975023e-01 # value of this tag
    },
    { "id":"203",          # tag id according the tag definition file
      "val":1              # BOOL as 0/1 - Not as true/false!
    }
  ]
}
```

Note

The PROFINET IO Connector does not support 'write' array variables when tags are defined.

Error Feedback

If you observe any irregularity with 'write' commands, then you can check the log file and analyze the issue. You can download the log file from the Industrial Edge Device.

8.3 PROFINET Data Record Write

Note

Applicability

The Record WRITE functionality is only for PROFINET IO Connector as controller to write records to connected devices. When using the connector in device mode, this feature is not available. In this case, record write requests can be send from the controller (e. g. a PLC) to a device.

The PROFINET IO Connector offers the function PN Data Record Write to PROFINET IO devices. The PROFINET IO Connector forwards the raw data to the device. Therefore, the client must provide an array of byte as data to be written to the device.

In most cases, only a one-to-one connection happens. For example, only one client is connected to the PROFINET IO Connector.

Interface for Client

This client interface is based on MQTT:

- Client publishes the Data Record Write request at a specific topic
- The PROFINET IO Connector (pn-hs) subscribes this topic
- pn-hs executes the Data Record Write request to the selected device
- The result of the Write operation is returned in a separate MQTT topic (success and failure)

Additionally, the PROFINET IO Connector creates a user log entry in case of an error.

Limits of this approach

- No per-client communication
- Every client can request Data Record Write commands
- Every client gets all responses of all Write requests

MQTT Topic

One MQTT topic is used for the Write request and one for the response. These topics are defined in the below link:

For Controller Configuration, refer step no. 6 of Add data sources (Page 76) .

Topic	Function
Write request	pn-hs subscribes this topic for Data Record Write requests.
Write response	pn-hs publishes response to Data Record Write requests to this topic (success and failure).

MQTT Payload for Data Record Write Request

The Data Record Write request requires the mandatory information as follows:

- The logical address LADDR of this submodule
- The data record index
- The raw data as byte array

The payload for a PN Data Record WRITE Request appear as follows:

```
{ "seq": 1, "reqs": [
  { "laddr": 261, "index": 45040, "val": [42, 127, 255, 12] },
  { "laddr": 261, "index": 45041, "val": [31, 66, 128, 255] }
]}
```

One message can include more than one requests. Every request contains LADDR, the data record index, and the byte array with the raw data.

Note

- Both parameters LADDR and index are provided as decimal number in the JSON. In PROFINET manuals, the index is typically provided as the hexadecimal number. For example, Index I&M3 Data Record = aff3 (hex) = 45043 (dec).
 - The byte array contains a list of byte values. The valid range is 0 to 255. If any value falls outside this range, the PROFINET IO Connector will generate an error.
-

Response Message

The PROFINET IO Connector sends a response for every PN Data Record Write request (success and failure) to the defined topic.

Positive Response

In case of success, the response message contains the timestamp of the acknowledge from the PN IO-Device for writing the data record and the "status":0. One response message can contain multiple write-response information. The following image depicts an example for positive response for PN Data Record Write request:

```
{
  "seq": 14, # sequence number of MQTT message
  "rsps": [
    {
      "ts": "2021-05-10T06:25:44.412357Z", # timestamp of ack. from
device
      "laddr": 261, # logical address
      "index": 45043, # index (decimal!)
      "status": 0 # status of operation
    }
  ]
}
```

Negative Response

There are a lot of possible reasons for failures, for example, request false PROFINET logical address (LADDR), false index, device is not connected, and so on. In the PROFINET IO Connector, there are various modules where the failure may appear. The source of the failure is coded in the highest nibble of the 32-bit status code. The following table describes the coding failure reason in status code:

Code	Description
0x0100'0000	Error calling PN-Driver API
0x0200'0000	Error in PN-Driver callback function
0x0300'0000	JSON parsing PN Data Record Read request
0x0400'0000	JSON parsing PN Data Record Write request

The response contains the timestamp, the error code, and an error description. The error description explains the meaning of the status code and may contain additional information depending on the type of error. The following image depicts an example for negative response for PN Data Record Write request:

```
{
  "seq": 18, // message sequence counter
  "rsps": [
    {
      "ts": "2021-05-10T08:43:00.482791Z",
      "laddr": 123,
      "index": 45041, // hex. aff1
      "status": 67108870, // hex. 0400'0006 - JSON parser error
      "errorDescription": "PN Data-Record Write Req: Invalid Byte
value!"
    },
    {
      "ts": "2021-05-10T08:43:00.482814Z",
      "laddr": 123,
      "index": 45041, // hex. aff1
      "status": 67108875, // hex. 0400'000B - JSON parser error
      "errorDescription": "PN Data-Record Write Req: Incomplete job
request!"
    },
    {
      "ts": "2021-05-10T08:43:00.482866Z",
      "laddr": 123,
      "index": 45045, // hex. aff5
      "status": 16777476, // hex. 0100'0104 - PN-Driver API Call
error
      "errorDescription": "ERROR PN Data Record Write: code '260'
descr 'parameter address is wrong'"
    },
    {
      "ts": "2021-05-10T08:43:00.487333Z",
      "laddr": 277,
      "index": 45099, // hex. b02b
      "status": 33554446, // hex. 0200'000E - PN-Driver Callback
      "errorDescription": "ERROR PN Data Record WRITE Cbf: code
```

8.3 PROFINET Data Record Write

```
'0xdf-80-b0-0a' descr 'IODWriteRes PNIOBW Access invalid index'"
    }
]
}
```

In case of any failure, an entry in the user log is created. You can analyze the issues in the user log file. You can download the log file using Industrial Edge Management.

See also

Add data sources (Page 82)

<https://www.profibus.com/> (<https://www.profibus.com/>)

8.4 PROFINET Data Record Read

Note

Applicability

The Record READ functionality is only for PROFINET IO Connector as controller to read records from connected devices. When using the connector in device mode, this feature is not available. In this case, record read requests can be send from the controller (e. g. a PLC) to a device.

The PROFINET IO Connector offers the functionality PN Data Record Read from the PROFINET IO devices. The PROFINET IO Connector provides the raw data, and the analysis of the record data must be done by the client.

In most cases, only a one-to-one connection happens. For example, only one client is connected to the PROFINET IO Connector.

Interface for Client

This client interface is based on MQTT:

- Client publishes the Data Record Read request at a specific topic
- The PROFINET IO Connector (pn-hs) subscribes this topic
- pn-hs executes the Data Record Read request to the selected device
- pn-hs publishes the Read data to a specific MQTT topic (response)
- In case of failure, the pn-hs creates a user log entry
- The result of the Read operation is returned in a separate MQTT topic (success and failure). For success, the record is returned as array of byte. For failure, the status and error description are returned.

Additionally, the PROFINET IO Connector creates a user log entry in case of an error.

Limits of this approach

- No per-client communication
- Every client can request Data Record Read commands
- Every client can subscribe the MQTT topic and can see the data from other clients as well

MQTT Topics

You need two new MQTT topics: one for the request and the other for the response (read data). These topics are defined in below link:

For Controller Configuration, refer step no. 6 of Add data sources (Page 76).

Topic	Function
Read request	pn-hs subscribes this topic for Data Record Read re-quests.
Read response	pn-hs publishes the response at this topic (success and failure).

MQTT Payload for PN Data Record Read Request

For a Data Record Read request, you need the following data:

- The PROFINET logical address LADDR of the submodule
- The Data Record index (unsigned 32 bit)

The payload for a PN Data Record Read request may look as follows:

```
{ "seq": 1, "reqs": [
  { "laddr": 261, "index": 45040 },
  { "laddr": 261, "index": 45041 }
] }
```

One message can include more than one request. Every request contains the PROFINET logical address (LADDR) of the device and Data Record index.

Note

All parameters LADDR, index, and the byte array are provided as decimal number in the JSON. In PROFINET manuals, the index is typically provided as hexadecimal number. For example, Index I&M0 Data Record = aff0 (hex) = 45040 (dec).

MQTT Payload for Data Record Read Response

The PROFINET IO Connector sends a response for every PN Data Record Read request (success and failure) to the defined topic.

Positive Response

In case of success, the request contains the data record as an array of byte. The message contains the following information:

- Timestamp when the record was received by the driver
- The PROFINET logical address LADDR of the submodule
- The Data Record index (unsigned 32 bit)
- The "status": 0
- The raw data as byte array

Below shows is an example for positive response for PN Data Record Read request:

```
{
  "rsps": [
    {
      "ts": "2021-05-10T11:49:44.259308Z",
      "laddr": 280,    # logical address
      "index": 45042, # index (decimal value!)
      "status": 0,    # status 'ok'
      "val":
    [0, 34, 0, 18, 1, 0, 32, 32, 32, 32, 32, 32, 32, 32, 32, 32, 32, 32, 32, 32]
    ]
  ]
}
```

Negative Response

In the PROFINET IO Connector, there are various modules where the failure may appear. The source of the failure is coded in the highest nibble of the 32-bit status code. The following table describes the coding failure reason in status code:

Code	Description
0x0100'0000	Error calling PN-Driver API
0x0200'0000	Error in PN-Driver callback function
0x0300'0000	JSON parsing PN Data Record Read request
0x0400'0000	JSON parsing PN Data Record Write request

The response contains the timestamp, the error code, and an error description. The error description explains the meaning of the status code and may contain additional information depending on the type of error. Below is an example of a negative response for a PN Data Record Read request:

```
{
  "seq": 2,
  "rsps": [
    {
      "ts": "2021-05-10T11:55:29.516266Z",
      "laddr": 263,          // logical address
      "status": 50331653, // hex. 0300'0005 - JSON parse error
      "errorDescription": "PN Data-Record Read Req: Incomplete job
request!"
    },
    {
      "ts": "2021-05-10T11:55:29.516409Z",
      "laddr": 162,          // logical address
      "index": 201,          // index (decimal value!)
      "status": 16777476, // hex 0100'0104 - PN-Driver API Call
error
      "errorDescription": "ERROR PN Data Record READ: code '260'
descr 'parameter address is wrong'"
    },
    {
      "ts": "2021-05-10T11:49:44.144417Z",
      "laddr": 259,          // logical address
      "index": 45039,        // index (decimal value!)
      "status": 33554441, // hex. 0200'0009 - Error signaled by
Callback
      "errorDescription": "ERROR PN Data Record READ Cbf:
code '779' descr 'address unknown in configuration, check your
configuration'"
    },
    {
      "ts": "2021-05-10T11:49:44.147298Z",
      "laddr": 260,          // logical address
      "index": 45039,        // index (decimal value!)
      "status": 33554442, // hex. 0200'000A - Error signaled by
Callback
      "errorDescription": "ERROR PN Data Record READ Cbf: code
```

8.4 PROFINET Data Record Read

```
'0xde-80-b2-08' descr 'IODReadRes PNIOBW Access invalid slot or  
subslot''  
}  
]  
}
```

In case of any failure, an entry in the user log is created. You can analyze the issues in the user log file. You can download the log file using Industrial Edge Management.

See also

Add data sources (Page 82)

Troubleshooting

There are several options for troubleshooting. The available options are as follows:

- Check the web UI of IEM
- Check the web UI of IED
- Analyze log file of application
- Analyze the app statistics
- Analyze the log messages of the Common Configurator (for V2.1.0 and above)
- Utilize version info of the four configuration files

The information about troubleshooting is described in the following chapters:

- Application Log Files (Page 110)
- Application Statistics (Page 112)
- Tips and Tricks - Typical Issues (Page 116)

9.1 Application log files

Download Log File

When you want to check possible issues of the PROFINET IO Connector, you can download the log files.

To download the log files from the PROFINET IO Connector, launch "Apps" and navigate to "IED > Apps > PROFINET IO Connector > Download Logs".

Typical Successful Start-up

An example for typical start-up procedure log file is as follows (the time stamps are removed to get better overview):

```
PROFINET High-Speed Adapter '0.9.1.20 R' started.
Configuration file '../config/pn_hs_adpt_appconfig.xml' read.
Config version: 'V0.9 26.06.2020 09:40'
Configuration file '../config/pn_hs_adpt_credentials.xml' read.
Config version: 'V0.9 19.06.2020 07:43'
Start PN Controller with config '../config/Station_1.PN
Driver_1.PNDriverConfiguration.xml'
Use docker container IF 'eth1' as PROFINET interface
Device Info Dev '0' IP '192.168.152.1' Name 'profinetxadriver4933'
Process Image Input: '20' bytes from '5' submodules -> net size
'35' bytes
Connect MQTT client 'pn_hs_adpt_pub' to broker 'mymosquitto'
Try to connect to MQTT broker 'mymosquitto' ...
Start MQTT publisher with oversampling '2' topic 'ie/d/b/simatic-
v1.0/pnhs1/dp/r'.
MQTT publisher connected to broker.
Start PN-Controller with cycle time '1000'ussec, MQTT reduction
factor '2000' -> acq.cycle '20000000' [0.1us]
ALARM 'Device return' (GOING ) device '1' slot '0' subslot '0'
1st successful read of PNIO data!
Device Info Dev '1' IP '192.168.152.5' Name 'softwarexaplcxb1857c'
```

In the log file you can see:

- The version of the running application
- What configuration files are used (incl. version)
- What interface is used for PN communication
- The IP address and name of the PN controller (this app)
- The size of the process image
- The name of the used MQTT topic
- The cycle times
- The name and IP of all PN Devices, connected to the controller

Version of App

In the beginning of the logfile, you can see the detailed version info of the PROFINET IO Connector application as follows:

```
PROFINET High-Speed Adapter '0.3.1.21 R' started.
```

Version of Configuration Files

Both configuration files (app config and user credentials) include a version (Tag VersionInfo). The only function of this version is to be printed in the log file. It enables you to check the version of the config files that the running application is using.

```
Configuration file '../config/pn_hs_adpt_appconfig.xml' read.
```

```
Config version: 'V0.9 26.06.2020 09:40'
```

```
Configuration file '../config/pn_hs_adpt_credentials.xml' read.
```

```
Config version: 'V0.9 19.06.2020 07:43'
```

9.2 Application statistics

Note

This feature is only applicable to versions prior to V2.0.0. For more information on different features depending on the version, refer Diagnostic features by version (Page 125).

The PROFINET IO Connector collects internal statistics. You can activate in the configuration file to print out this information in the user log file in a cyclic way (for example, every 10 seconds).

```
<StatisticMonCycleSec>10</StatisticMonCycleSec>    <!-- OPTIONAL
parameter: cycle time of 'statistic monitor' in [sec] -->
```

Note

- To count all the events internally, the Unsigned 32-bit Integer values are used. Therefore, they overrun regularly.
- To reduce the size of the statistics output, most values are only printed on change. For example, if the counter values are not changed, these are not printed out.

In Tips and Tricks - Typical Issues (Page 116), you can find hints how to take advantage of the statistics during commissioning.

Content of Statistics

The following table describes the comments for each section:

Section	Comment
PROFINET	General overview about the status of the configured PROFINET modules
PN Alarms	Counter for all received PROFINET alarms
PNIO cbf	Counter for general PROFINET callback calls (for example, device activated)
PNcbfRcdR	Counter PROFINET callback calls for PN-Data-Record Read
PNcbfRcdW	Counter PROFINET callback calls for PN-Data-Record Write
PNIO READ	Counter PROFINET IO-Data Read access
PNIO WRIT	Counter PROFINET IO-Data Write access
RT-Thread	Statistics for the real-time thread (PN Controller)
PNRD FIFO	Counter of internal FIFO for the recorded PN IO-Data Read
PnRcdRspF	Counter of FIFO for PN-Data-Record Read Responses
MQTT clnt	Counter of MQTT client functionality
MQTT Thrd	Statistics for the MQTT thread
PnIoWr FI	Counter for the internal FIFO for PN IO-Data Write
MQTT	Additional counter for MQTT
JP PNIO W	Counter JSON Parser for 'PN IO-Data Write'
JP PnRcdR	Counter JSON Parser for 'PN Data-Record Read'

Section	Comment
JP PnRcdW	Counter JSON Parser for 'PN Data-Record Write'
PnRcdWrk	Counter 'PN Data-Record Worker' handling all the PN Data-Record jobs

Statistics of PROFINET Controller

The running PROFINET Controller produces a lot of statistics information as follows:

- How many PNIO modules are configured in TIA Portal project in total?
- How many of the configured modules are currently online?
- Number of alarms Device Return
- Number of alarms Device Failure
- Number of alarms Plug
- Number of alarms Pull
- Number of alarms Submodule Return
- Number of alarms Pull Module

Note

The number of modules total and modules online may be a little bit confusing.

When a module has got input and output bytes, it will get two LADDR values. But for the number of modules, it counts only one. Therefore, the number of modules total is smaller than the number of LADDR values in the TIA Portal project.

This is typical for PN iDevices. They have got transfer area memory including input and output bytes.

Statistics of MQTT Publisher

The MQTT publisher produces a lot of statistics information, for instance:

- Number of successful connect to broker
- Number of disconnect from broker
- Number of published MQTT messages
- Number of application messages
- Number of application warnings
- Number of application errors
- Number of publish calls when not connected
- Number of errors returned by publish call
- Number of publish retain calls when not connected
- Number of errors returned by publish retain call

Statistics of Internal FIFOs

Several FIFOs (queues) are used to transfer data between real-time and non-real-time tasks. The statistics help to find possible issues. It includes for instance counters for:

- Write index in the internal FIFO
- Read index in the internal FIFO
- How many data records are currently stored in the FIFO?
- Total number of records processed by the FIFO
- How many records are lost, due to buffer overflow?

Example Output

If activated, the log entries may look like as follows:

```
PPROFINET: modules total'9' online'9'
PN Alarms: DevRet'2' DevFail'0' Plug'0' Pull'0' SubRet'0' PullMod'0'
PNIO cbf : MdCh'1' DevAct'0' Diag'2' Subm'1' DevDiag'0'
PNcbfRcdR: cbf'45' ok'22' Len0'7' RespErr'6' Err'10' AddE'0'
PNcbfRcdW: cbf'4' ok'1' RespErr'1' Err'2' AddE'0'
PNIO READ: ok'135' ERR'0' int'0' Rem OK'123' BAD'12'
PNIO WRIT: ok'0' err'0' Rem OK'0' BAD'0'
PNIO WRIT: ERROR False addr'0' False len'0' Not writable'0'
RT-Thread: cycles'45' exec (us)'4' min'3' max'46' avg'3'
RT-Thread: Jitter (us)'1' min'-5' max'6' avg'0'
PNRD FIFO: wr-idx'41' rd-idx'40' RECORDS stored'1' free'2183'
processed'41' lost'0'
PnRcdRspF: PUT ok'22' overfl'0' GET ok'22' empty'6'
PnRcdRspF: PUT err'0' GET err'0'
PnRcdRspF: wr-idx'4016' rd-idx'4016' Bytes: stored'0' free'65536'
proc'4016' lost'0'
MQTT clnt: cycles'49' PUB IO-RD bin'20' IO-RD json'20' RCD-RD-
Rsp'3' PN-Alrm'2' PN stat'2'
MQTT clnt: Err task overflow '0' Err JSON compose '0'
MQTT Thrd: wait (us)'500000' min'499695' max'500000' avg'499971'
MQTT Thrd: exec (us)'0' min'0' max'305' avg'22'
PnIoWr FI: PUT ok'0' overfl'0' GET ok'0' empty'90'
PnIoWr FI: PUT err'0' GET err'0'
PnIoWr FI: wr-idx'0' rd-idx'0' Bytes: stored'0' free'65536' proc'0'
lost'0'
MQTT      : pub'47' pub retain'1'
MQTT      : conn'1' discon'0' pub'48' rcvd'2' warn'0' err'0'
MQTT      : pub nc'0' pub err'0' pub.r nc'0' pub.r err'0'
JP PNIO W: calls'0' valid'0'
JP PNIO W: arrlen'0' Unexparr'0' UnexpJTyp'0' UnexpJItem'0'
byteRange'0' incompl'0' typenotsup'0'
JP PNIO W: skipped prop'0' array'0'
JP PnRcdR: calls'1' valid'54'
JP PnRcdR: Unexparr'0' incompl'0' typenotsup'0'
JP PnRcdR: skipped prop'1'
JP PnRcdW: calls'1' valid'5'
```

```
JP PnRcdW: arrlen'0' Unexparr'0' UnexpJTyp'0' UnexpJItem'0'
byteRange'0' incompl'0' typenotsup'0'
JP PnRcdW: skipped prop'1' array'0'
PnRcdWrk : cycles'244' RCD-RD(app)'21' RCD-RD(mqtt)'24' RCD-WR'4'
limit'8'
PnRcdWrk : Err rcd-rd(app)'0' rcd-rd(mqtt)'30' rcd-wr'1'
```

9.3 Tips and tricks - Typical issues

General Performance

In the IEM, you only get a very general overview about the CPU load of an app. The PROFINET IO Connector is a relatively complex connector. Its performance depends on the several factors. It is important to get a detailed picture about possible bottlenecks.

How to fix?

The output of the statistics monitor gives a lot of useful information.

- When the PN alarm 'Device Failure' (DevFail) is increasing, the reason could be the PN wiring or a massive overload of the PROFINET IO Connector.
- When the execution time of the real-time thread comes close to the required recording time resolution (RT-Thread: exec (us) avg '3'), the performance is not good enough.
- When the internal FIFOs signal lost data (FIFO: lost '0', PnRcdRspF: lost '0', PnIoWr FI: lost '0'), the amount of data traffic is too high.
- When the JSON parser signals errors (JP PNIO W, JP PnRcdR, JP PnRcdW), a client is sending erroneous messages.

IEM: Unable to Start Application

You see the following information:

- IEM: Installation Failed
- IED: Unable to start application

How to fix?

The prerequisites are not fulfilled. The network environment at the IED is not installed.

LOG: MQTT Connect Failed

You see the following info in the log file:

```
Try to connect to MQTT broker 'mosquitto_broker'...
ERROR MQTT - mosquitto_connect failed
ERROR MQTT - unable to initialize mosquitty
```

How to fix?

The Databus application is not installed or is not running.

LOG: MQTT Publisher Disconnected from Broker

You see the following info in the log file:

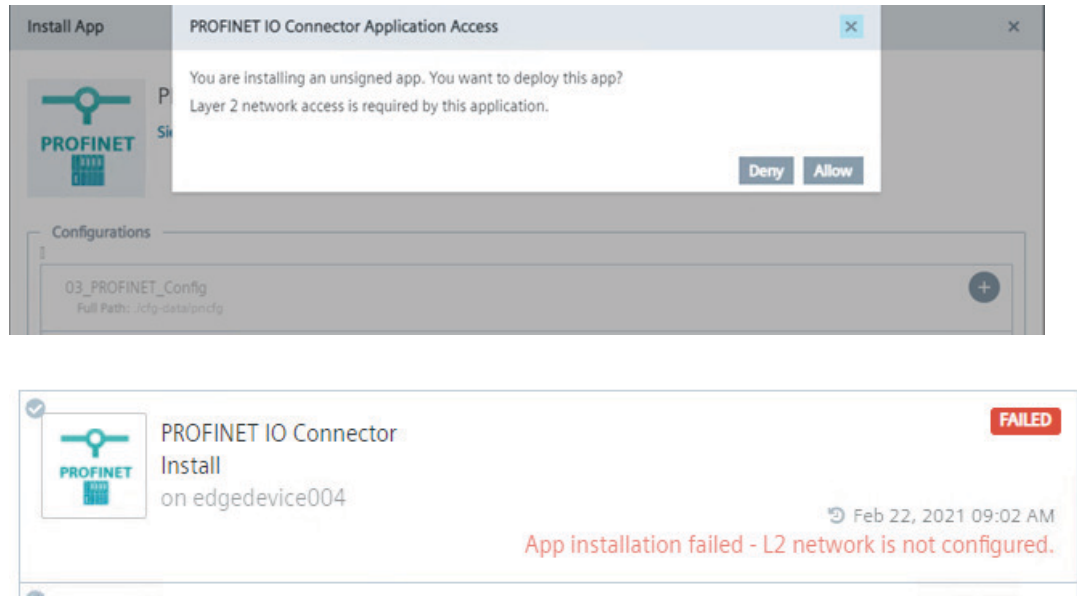
```
MQTT publisher connected to broker.
MQTT publisher disconnected from broker - code '5'.
```

How to fix?

You have not configured the user and topic in the Databus application.

Installation Fail

When you try to install the PROFINET IO Connector, the IED requires the Layer-2-Access to be activated for apps. If the Layer-2-Access is not activated, the following error messages are displayed in your Industrial Management System (IEM):



How to fix for the versions below 1.3?

You must onboard your IED again. During onboarding procedure, you must activate the Layer-2-Access for apps.

How to fix for the V1.3 onwards?

For the V1.3 and onwards, no need to onboard IED again. Only change the network setting under IED > "Settings" > "Connectivity" > "LAN Network".

PROFINET Device does not Connect

In the log file, you find the entry as follow:

```
P:PROFINET: '0' of '12' configured modules are online.
```

How to fix?

You can check the following points in this case:

- Is the PROFINET device name correct? (may be not unique?)
- Are the IP addresses in one subnet?
- Are all IP addresses unique?
- Is the Ethernet connection fine?

Note

You can use the TIA Portal or PRONETA to search for devices. With these tools the device name and IP address can be checked.

PROFINET Device Connects and Disconnects

In the log file, you find the entry as follow:

```
ALARM 'Device return' (GOING ) device '1' slot '0' subslot '0'  
ALARM 'Device failure' (COMING) device '1' slot '0' subslot '0'
```

How to fix?

You can check the following points in this case:

- What is the overall CPU load of the IED? (very high CPU load can influence the PROFINET functionality)
- Are all PROFINET device names unique in the network?

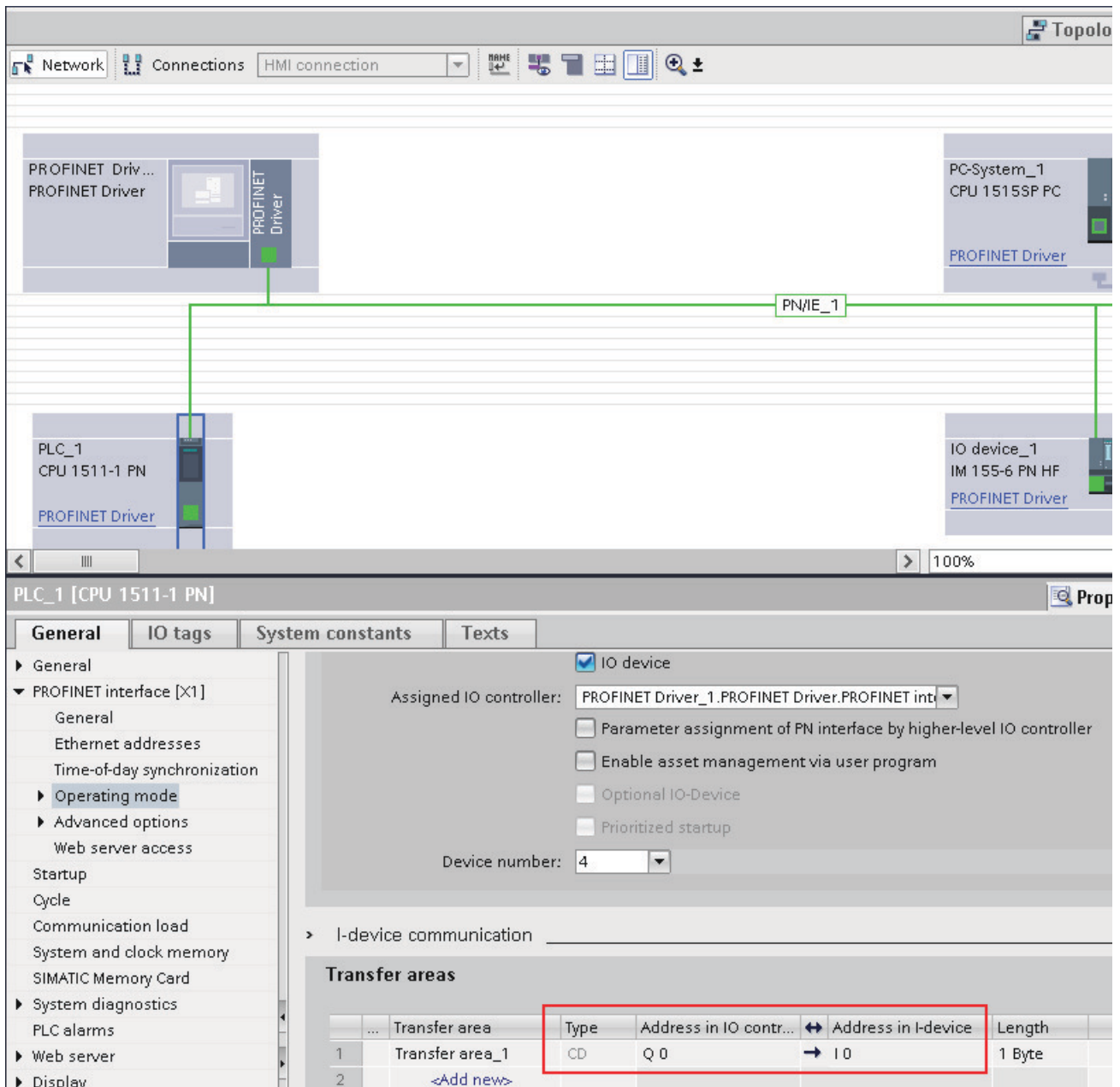
No PROFINET IO Data Read from Connected PROFINET iDevice

When the network is fine and app log displays 'Alarm Device Return' but the PROFINET IO Data is not transferred.

When you use tag definitions, you can find an error entry in the log file as 'tag definition wrong'.

How to fix?

When you configure transfer area for a PROFINET iDevice in the TIA Portal, the default setting is Input (I) for the iDevice. You must change it to Output (Q) as follows:



Only few Values are send via MQTT

If you experience the following behavior with the PROFINET IO Connector:

- No values are published to the MQTT broker although a device is connected and the PROFINET IO Connector is running, and after some minutes the first ten values are published at once and no further messages are published.

It is likely that the network does not meet the PROFINET requirements. For more information, refer Prerequisites (Page 25). The Connector may not always detect the problem, and the log file may not show any errors.

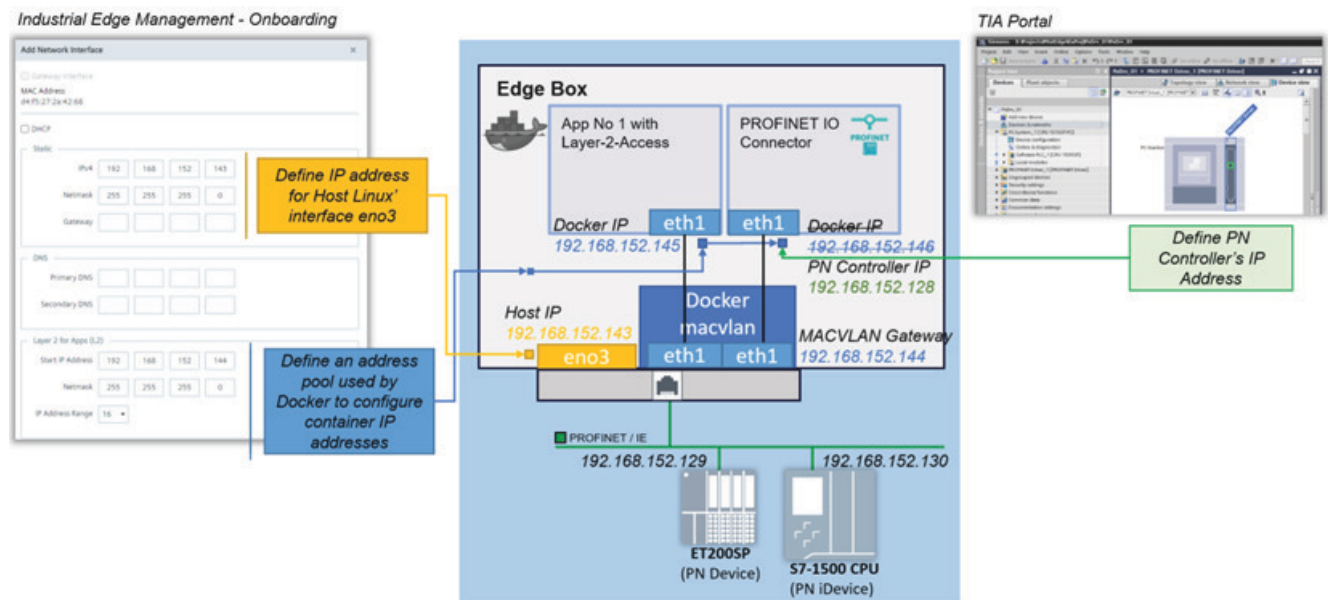
9.3 Tips and tricks - Typical issues

To troubleshoot this issue, reduce the network traffic. Make sure the interface for the PROFINET communication (Layer-2-access) is not used for connection to the IEM. Connect the device directly to the IED where the PROFINET IO Connector is installed. Now the connection should work as expected. The rest of the network should be attached step by step to see which component causes the error.

9.4 Configuring Layer-2-Access

9.4.1 Introduction

The PROFINET IO Connector needs a specific network configuration for the specific functionality of a PROFINET IO Controller. This network configuration is called as Layer-2-Access. It is mandatory for the operation of the PROFINET IO Connector. The following image depicts the network settings for Layer-2-Access:



One physical Ethernet network interface (gray box in the above image) is used for several virtual (logical) network interfaces. This port is shared by real and virtual interfaces, each having its own IP and MAC address.

The 'eno3' interface is part of the host operating system. It is controlled by the Industrial Edge operating system.

The 'eth1' interface of the Docker containers is externally visible as own network interface in the network with its IP address and also its own MAC address. The IP address which Docker assigns to the PROFINET IO Connector interface 'eth1' is later overwritten by the configured PN Controller IP address which has been set in the TIA Portal.

9.4.2 How to configure Layer-2-Access?

During the IED onboarding, you must configure the following:

- Assign an IP address for the Ethernet Interface of the Linux Host (eno3).
(It could be static, DHCP, or empty if you do not need it for host access to that network.)
- Provide an IP address range which is used by Docker for the layer 2 network.
(The macvlan bridge and the Docker containers attached to the layer 2 Docker network.)
(The 1st address 192.168.152.144 is reserved to avoid routing to a gateway.)

Note

With IED V1.2 and later version, you can change the layer-2-configuration after onboarding as well.

During the layer-2-configuration, it is advisable to follow these recommendations:

- The blue and the green IP addresses, as depicted in the above image, must belong to the same IP subnet.
- The yellow IP address, as depicted in the above image, is recommended to configure the same IP subnet as well. Technically, it may belong to a different IP subnet.
- All IP addresses must be unique (no double usage of same IP is allowed).
- You cannot use DHCP for the layer 2 interface because Docker and PROFINET do not fully support DHCP.
- You can use DHCP for the host interface 'eno3', but you must ensure that the IP addresses and IP ranges which are used for the statically configured network interfaces are excluded from the IP address range which the DHCP server uses.
- You must ensure that in the whole network all PROFINET device names are unique. Even same names in different IP subnet may cause issues. The handling of the PROFINET device names does not depend on IP addresses. LLDP is used for this purpose.
- The IP addresses of the external PROFINET devices are not covered by this configuration. You must ensure that these have their own unique IP addresses outside the range of the layer-2-configuration.

IP Address Range

The layer-2-configuration requires an IP address range. The possible range is defined by the start IP address and the 0-values of the binary address. The following table describes the examples for start address and resulting ranges:

Start IP Address	Binary Format (last number)	Resulting Address Range
192.168.152.0	bin 0000 0000	256
192.168.152.4	bin 0000 0100	4
192.168.152.8	bin 0000 1000	8
192.168.152.6	bin 0000 0110	2

The following image depicts the concept for a 64-bit subnet:

Layer 2: Possible ranges (length of range and position) for a subnet with 64 addresses (CIDR/26)	
Range Length	
64	64
32	32 32
16	16 16 16 16
8	8 8 8 8 8 8 8 8
4	4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4
2	2 2
1st address of range	0 2 4 6 8 10 12 14 16 18 20 22 24 26 28 30 32 34 36 38 40 42 44 46 48 50 52 54 56 58 60 62

Layer-2-Configuration Advanced Settings

When a large address range is not available for the Layer-2-Configuration, you can use the Layer-2-Network advanced settings. You can define the IP address for the docker internal

gateway and can exclude specific IP addresses from the configured IP address range as follows:

The screenshot shows the 'L2 Network Advanced Settings' window. At the top, the 'Gateway' is configured with the IP address 190.99.80.52. Below this, the 'IP Addresses for Layer 2' section is expanded, showing a list of four IP addresses: 190.99.80.52 (Gateway), 190.99.80.53, 190.99.80.54, and 190.99.80.55, each with a green checkmark indicating it is selected. An 'Apply' button is located to the right of this list. Below the IP list, the 'Secondary DNS' field is empty. The 'Layer 2 (L2) for Apps (Optional)' section is also expanded, showing the 'Start IP Address' as 190.99.80.52, the 'Netmask' as 255.255.0.0, and the 'IP Address Range' set to 4. An 'Advanced Settings' button is located to the right of these fields. At the bottom of the window, there are two warning messages: 'Removing the L2 network will stop the functioning of apps that use it.' and 'Changing the network configuration will invalidate the current browser tab.' An 'Update' button is located at the bottom right.

- The Layer-2-Configuration IP address range is from 192.168.152.0 to 192.168.152.7.
- The IP address 192.168.152.6 is used by an external and not changeable network component.
- The Netmask is 255.255.255.0.

9.4.3 PROFINET Addresses

In TIA Portal, you must configure the following PROFINET configuration:

- IP Address of the PROFINET IO Controller (the PROFINET IO Connector)
(This PN controller IP replaces the Docker IP during app startup.)
- IP Addresses of all the PROFINET Devices and iDevices

Note

The IP addresses of all PROFINET components must be located outside the configured Layer-2-Configuration IP address range.

9.5 Diagnostic features by version

The PROFINET IO Connector offers different features for diagnosis depending on the version as given in below table:

Version	Feature
V1.X.X and below	PN status and PN alarms
V2.0.0	Configurable in .json with Common Configurator. For more information, refer Configuring PROFINET IO Connector Using Common Configurator (Page 72).
V2.1.X and above	Status and information

PN status and PN alarms

PN status provides information for all configured PN devices. PN alarm provides event messages.

Configuration

The application publishes the information via MQTT to specific topics. The topics can be configured in the `pn_hs_adpt_appconfig.xml`

Example

Example configurations for the topics are as given below:

```
<MqttTopicPnStatus>ie/d/j/simatic/v1/pnhs1/status/r</MqttTopicPnStatus>
```

```
<MqttTopicPnAlarm>ie/d/j/simatic/v1/pnhs1/ev/r</MqttTopicPnAlarm>
```

Status and information

The information related to the PROFINET IO Connector is split between two topics: the status topic and the info topic. The status topic provides an interface as defined in Siemens Cloud's Industrial Edge System Apps Documentation for Developers (V1.2.2). This interface offers details about the application and its connections.

The additional live information of the PROFINET IO Connector is published to the info-topic.

The status messages are more clearly described and standardized, providing the status of the connector and the connections so that other applications can interpret and display relevant information.

Configuration

The status topic can be configured in the config json file as well as with the Common Configurator. For more information, refer Configuring PROFINET IO Connector Using Common Configurator (Page 72).

The info-topic is not configurable: "ie/s/j/simatic/v1/pnhs1/info"

Example

For specific topic configurations, refer to the default MQTT topics for:

- For Controller Configuration, refer step no. 6 of Add data sources (Page 76).
- For Device Configuration, refer step no. 6 of Add data sources (Page 82).

For practical examples of JSON configuration for both Controller and Device, you can find detailed guidance in the following resources:

- For Controller Configuration example, refer Example of JSON Configuration for Controller (Page 67).
- For Device Configuration example, refer Example of JSON Configuration for Device (Page 71).

Status values

Since the status message is defined, there are following possible values for the connector and connection:

Connector	Description
Good	Connector is running properly
Bad	Run without config - no functionality
Available	Connector just started up
Unavailable	Connector has stopped

Connection	Description
Good	Connection is established
Stopped	Connection is not established
Bad	Connection has error

Example

Defined status message of the PROFINET IO Connector is as follows:

```
{
  "seq":3,
  "ts":"2023-01-11T07:02:15.1232539Z",
  "connector":{
    "status":"good",
    "reason":"ok"
  },
  "connections":[
    {
      "name":"plcxbld0ed (station'1' slot'1' laddr'261')",
      "status":"good",
      "reason":"ok"
    },{
      "name":"unknown (station'2' slot'0' laddr'267')",    # unknown
      "status":"stopped",
      "reason":"ok"
    }
  ],
}
```

```
    ...  
  ]  
}
```

The following info message gives additional details in controller mode:

```
{  
  "pn-controller":{  
    "name":"profinetxadriverv4933",  
    "order-no":"6ES7195-3AA00-0YA0",  
    "sw-version":"PROFINET-IO-Connector V2.1.0.0-R",  
    "ip-addr":"192.168.4.102",          # ip address of the IED layer-2-  
access interface  
    "ip-addr-num":[192,168,4,102]  
  },  
  "stations":[  
    {  
      # the following station is connected therefore information  
like name, ip-address is known  
      "name":"plcxbld0ed",  
      "station-no":1,  
      "ip-addr":"192.168.4.10",        # ip address of a connected PLC  
      "ip-addr-num":[192,168,4,10],  
      "submodules":[  
        {  
          "slot-no":1,  
          "laddr":261,  
          "order-no":"6ES7 510-1DJ01-0AB0 ",  
          "sn":"S C-K3L994142018",  
          "pn-addr-out":[0,3],  
          "status":"online",  
          "status-num":3  
        },  
        ...  
      ]  
    }, {  
      # the second station is not connected -> no information about  
name, etc.  
      "name":"",  
      "station-no":2,  
      "ip-addr":"",  
      "ip-addr-num":[0,0,0,0],  
      "submodules":[  
        {  
          "slot-no":1,  
          "laddr":271,  
          "order-no":"",  
          "sn":"",  
          "pn-addr-in":[32,32],  
          "status":"offline",  
          "status-num":2  
        },  
        ...  
      ]  
    }  
  ]  
}
```

```
    ]
  }
]
```

The following info message gives additional details in device mode:

```
{
  "pn-device": {
    "name": "pn-io-conn-device",
    "sw-version": "PROFINET-IO-Connector V2.1.0.0-R"
  },
  "controllers": [
    {
      "ip-addr": "192.168.4.10"      # ip address of the connected
IO controller
    }
  ]
}
```


Changes

10.1 Breaking changes

During the development there were several breaking changes with incompatible modifications. They are listed in the table below:

Version	Date	Change
V0.4	23.11.2020	Several PN Diagnosis info via MQTT topics (optional)
V0.4	26.10.2020	Cyclic Read PN IO-Data ready for DataService
V0.4	07.10.2020	Change of Metadata for PN IO-Data READ
V0.4	07.10.2020	Change of Format of Binary Payload (PN IO-Data READ)
V0.4	01.09.2020	Massive extension/change of app configuration file
V1.1	29.06.2021	Complete changed processing of tag definition file (fixed name, etc.)
V2.0	14.09.2022	One json config instead of four different config files (some parameters can not be configured anymore)
V2.1	08.09.2023	Tag ids has been removed (now internally calculated)

Changelog

Here are Release Notes of different versions listed:

Version	Change
V2.0	Introduce new single configuration file The app is now configurable with Common Configurator
V2.1	Add IO device mode Add status topic

