



Nexto Series PROFINET Manual

MU214621 Rev. A

May 17th, 2024

No part of this document may be copied or reproduced in any form without the prior written consent of Altus Sistemas de Automação S.A. who reserves the right to carry out alterations without prior advice.

According to current legislation in Brazil, the Consumer Defense Code, we are giving the following information to clients who use our products, regarding personal safety and premises.

The industrial automation equipment, manufactured by Altus, is strong and reliable due to the stringent quality control it is subjected to. However, any electronic industrial control equipment (programmable controllers, numerical commands, etc.) can damage machines or processes controlled by them when there are defective components and/or when a programming or installation error occurs. This can even put human lives at risk.

The user should consider the possible consequences of the defects and should provide additional external installations for safety reasons. This concern is higher when in initial commissioning and testing.

The equipment manufactured by Altus does not directly expose the environment to hazards, since they do not issue any kind of pollutant during their use. However, concerning the disposal of equipment, it is important to point out that built-in electronics may contain materials which are harmful to nature when improperly discarded. Therefore, it is recommended that whenever discarding this type of product, it should be forwarded to recycling plants, which guarantee proper waste management.

It is essential to read and understand the product documentation, such as manuals and technical characteristics before its installation or use.

The examples and figures presented in this document are solely for illustrative purposes. Due to possible upgrades and improvements that the products may present, Altus assumes no responsibility for the use of these examples and figures in real applications. They should only be used to assist user trainings and improve experience with the products and their features.

Altus warrants its equipment as described in General Conditions of Supply, attached to the commercial proposals.

Altus guarantees that their equipment works in accordance with the clear instructions contained in their manuals and/or technical characteristics, not guaranteeing the success of any particular type of application of the equipment.

Altus does not acknowledge any other guarantee, directly or implied, mainly when end customers are dealing with third-party suppliers.

The requests for additional information about the supply, equipment features and/or any other Altus services must be made in writing form. Altus is not responsible for supplying information about its equipment without formal request.

These products use EtherCAT® technology (www.ethercat.org).

COPYRIGHTS

Nexto, MasterTool, Grano and WebPLC are the registered trademarks of Altus Sistemas de Automação S.A.

Windows, *Windows NT* and *Windows Vista* are registered trademarks of Microsoft Corporation.

OPEN SOURCE SOFTWARE NOTICE

To obtain the source code under GPL, LGPL, MPL and other open source licenses, that is contained in this product, please contact opensource@altus.com.br. In addition to the source code, all referred license terms, warranty disclaimers and copyright notices may be disclosed under request.

List of Contents

1. INTRODUCTION	1
Features of Nexto Controllers with PROFINET	1
Documents Related to this Manual	2
Visual Inspection	2
Technical Support	3
Warning Messages Used in this Manual	3
2. INTRODUCTION TO PROFINET	4
Open Standard for Industrial Ethernet	4
Conformance Classes	4
System Model and Device Classes	4
Device Model of an IO Device (IOD)	5
Station Names and IP Addresses (DCP Protocol)	6
Neighborhood Detection (LLDP Protocol)	7
Device Replacement without Engineering Tool	8
Media Redundancy (MRP Protocol)	8
PROFINET Switches	9
Diagnostics	10
I&M Data	10
Device Descriptions (GSDML Files)	10
Basic Communication Functions	11
Cyclic Data Exchange	11
Acyclic Data Exchange	12
Device/Network Diagnostics	12
IOPS and IOCS	12
3. INSTALLATION	14
Typical Topologies	14
Line Topology using Embedded Switches	14
Star Topology using Auxiliar Switch	14
Mixed Line and Star Topology using Auxiliar Switch	15
Ring Topology using Embedded Switches	15
Mixed Ring and Star Topology using Auxiliar Switch - IOC as Ring Manager	15
Mixed Ring and Star Topology using Auxiliar Switch - Switch as Ring Manager	16
Installation Guidelines for Copper Cables	16
Maximum Length of Copper Cables	16
Transmission Speed	16
Types of Copper Cables	17
RJ45 Connectors for Nexto Series Devices	17
Shielding and Equipotential Bonding	18
Routing Copper Cables	19
Installation Guidelines for Fiber Optic Cables	25
Using Optical Media Converters	26
Using Switches with Optical Ports	27
4. CONFIGURATION	28
Insert PROFINET Controller (IOC) below a Network Interface	28
Select the Network Interface	28
Configure Parameters of Selected Port(s)	28
Insert Ethernet Adapter	29

Insert Instance of the PROFINET Controller	31
Adjust Interval of Profinet_IOTask	32
Configure General Parameters of the PROFINET Controller	32
Install Missing GSDML Files in Device Repository	33
Insert PROFINET Devices (IODs) below the Profinet Controller (IOC)	35
Insert Instance of a PROFINET Device	35
Configure General Parameters of a PROFINET Device	36
Disable Instances of Not Controlled PROFINET Devices	38
Insert I/O Modules below IODs	39
Configure Additional Parameters of IODs	41
Configure Parameters of I/O Modules below IODs	42
Renaming Device Tree Objects of IODs and I/O Modules	43
Naming Variables of IODs and I/O Modules	45
Configure Topology	45
Basic Topology Configuration	46
Configure Media Redundancy with MRP Rings	48
Configure MRP Ring with ALTUS Controller	50
Recommendations for MRP Rings	51
Adjust Parameter "Data hold time" of IODs in the Ring	51
Open the Ring before some MRP Configuration Changes	51
Specific Recommendation for ALTUS Controllers in an MRP Ring	51
Online Functions for Supporting Configuration	52
Assign Station Name and IP Address to an IOD, and Blink LEDs of an IOD	52
Scan for Devices	55
Import Topology Connections	58
Reset to Factory Defaults	59
Auto-IP	60
I&M	61
Special Configuration Functions	61
Update Device after Updating GSDML File	61
5. DIAGNOSTICS	63
Introduction to PROFINET Diagnostics	63
Data Structures for a Diagnostic	63
Methods for Retrieving Diagnostics in Nexto Controllers	65
Visualization of Diagnostics using Mastertool Programming System	66
Device Communication Status for the User Application	70
Generic Library for Retrieving Diagnostics in the User Application	70
Retrieving Module Differences	71
Retrieving other Types of Diagnostics - Example for an I/O Module	72
Retrieving other Types of Diagnostics - Example for PROFINET Ports of a Remote Head	75
Controller Diagnostics	77
MRP Diagnostics	77
6. I&M DATA	79
Description of I&M Data Structures	79
I&M0	79
I&M1	80
I&M2	80
I&M3	80
Addressing I&M Data	80
Reading and Writing I&M Data for a Device in Mastertool Programming System	80
Using Function Blocks for Reading or Writing I&M Data	82
7. PERFORMANCE ANALYSIS FOR PROFINET CONTROLLERS	84

Main Factors that Affect CPU Consumption in a PROFINET Configuration	84
CPU Model.....	84
Interval of Profinet_IOTask and Send Clock	84
I/O Cycle of Devices	84
I/O Data Size	84
Number of Devices	84
MRP	85
Examples of CPU Consumption for CPUs Xpress	85
Examples of CPU Consumption for CPU NX3008.....	85

1. Introduction

Nexto series has controllers that support PROFINET networking. This manual covers several topics related to Nexto series equipment that support PROFINET:

- Features of Nexto controllers;
- Supported topologies and media redundancy;
- Installation;
- Configuration;
- Diagnostics;
- I&M data (identification and maintenance);
- Performance analysis.

Because PROFINET is an open and interoperable standard, third-party equipment can be used together with Nexto equipment in a PROFINET network.

A small introduction to PROFINET technology is presented in chapter **Introduction to PROFINET**. Users that are newcomers to PROFINET technology should read this chapter first, for understanding better this manual.

Features of Nexto Controllers with PROFINET

The following table lists all Nexto controllers that support PROFINET and their PROFINET related features.

Feature (note)	Controller Models			
	Xpress CPUs (XP300, XP315, XP325, XP340)	NX3003, NX3004, NX3005, NX3010	NX3020, NX3030	NX3008
Conformance Class (1)	A	A	A	A
Maximum number of PROFINET controller instances (2, 3, 4, 5)	1	1	1	2
PROFINET ports compatible with PROFINET	NET1	NET1	NET1 NET2	NET1 NET2 NET3
Transmission Speed of PROFINET Ports	100 Mbps	100 Mbps	100 Mbps	100 Mbps 1000 Mbs (only for NET1)
Maximum Number of IODs (6)	Depends on many factors	Depends on many factors	Depends on many factors	Depends on many factors
LLDP	Yes	Yes	Yes	Yes
Device replacement without engineering tool	Yes	Yes	Yes	Yes
Media Redundancy	None	None	None	None MRP client MRP manager

Table 1-1. Features of Nexto controllers with PROFINET

Notes in first column of previous table:

1. Although the controllers have conformance class A, they support all functions of conformance class B with one exception: SNMP.
2. A PROFINET controller instance may use:
 - A single Ethernet port;

- A pair of Ethernet ports (switched ports), only for model NX3008. This enables special topologies (line, ring with MRP).
3. A PROFINET controller instance in NX3008 can use:
 - NET1
 - NET2
 - NET2+NET3
 4. No more than two PROFINET controller instances can be installed in NX3008:
 - First instance in NET1;
 - Second instance in NET2 or NET2+NET3.
 5. Only a single instance of PROFINET controller can be installed in the other controllers (XP300, XP315, XP325, XP340, NX3003, NX3004, NX3005, NX3010, NX3020, NX3030).
 6. There is not an imposed limit for the number of IODs supported by each controller model. However, in practice, many factors influence on the number of IODs that can be connected to a controller. One of these factors, off course, is the CPU performance. But other factors are also important, like the I/O cycle of each device, the presence of MRP configuration, the number of I/O bytes inside each device, and so on. The chapter **Performance Analysis for PROFINET Controllers** shows some examples of CPU consumption for some controller models, and how it is affected by these factors.

Documents Related to this Manual

For additional information about the Nexto Series, other documents (manuals and technical characteristics) can be consulted in addition to this one. These documents are available in their latest revision at www.altus.com.br.

Each product has a document called Technical Characteristics (CE), which contains the characteristics of the product in question. Additionally, the product may have User Manuals (manual codes are quoted on CE).

The following documents are advised as a source of additional information:

- Technical characteristics (CE) of each product;
- Nexto Series User Manual;
- Nexto Xpress User Manual;
- Nexto Series CPUs User Manual;
- MasterTool IEC XE User Manual;
- MasterTool IEC XE Programming Manual.

Visual Inspection

Before proceeding with the installation, it is recommended to carry out a careful visual inspection of the equipment, verifying that there is no damage caused by transportation. Check that all components of your order are in perfect condition. In case of defects, inform the transport company and the nearest Altus representative or distributor.

CAUTION:

Before removing the modules from the packaging, it is important to discharge any static potentials accumulated in the body. To do this, touch (with your bare hands) any grounded metallic surface before handling the modules. This procedure ensures that static electricity levels supported by the module will not be exceeded.

It is important to record the serial number of each equipment received, as well as the software revisions, if any. This information will be necessary if you need to contact Altus Technical Support.

Technical Support

To contact Altus Technical Support in São Leopoldo, RS, call +55 51 3589-9500. To know the Altus Technical Support centers in other locations, see our URL (www.altus.com.br) or send an email to altus@altus.com.br.

If the equipment is already installed, have the following information ready when requesting assistance:

- The models of equipment used and the configuration of the installed system;
- The serial number of the CPU;
- The revision of the equipment and the firmware version, contained in the label affixed to the side of the product;
- Information about the CPU operation mode, obtained through the MasterTool programming system;
- The content of the application program (modules), obtained through the MasterTool programming system;
- The version of the programming system used.

Warning Messages Used in this Manual

In this manual, warning messages will have the following formats and meanings:

DANGER:

They report potential causes, which if not observed, lead to damage to physical integrity and health, property, environment and loss of production.

CAUTION:

They report configuration, application, and installation details that must be followed to avoid conditions that could lead to system failure and its related consequences.

ATTENTION:

They indicate important configuration, application or installation details to obtain the maximum operational performance of the system.

2. Introduction to PROFINET

This chapter provides a small introduction to PROFINET technology. It intends to help users to understand better this manual, especially those who are newcomers to PROFINET technology.

For users wanting more information about PROFINET technology, literature is available in the URL of PROFIBUS and PROFINET Organization (<https://www.profibus.com/pi-organization>). The document **PROFINET System Description- Technology and Application**, available in this URL, is a good point to start.

Open Standard for Industrial Ethernet

PROFINET is an open standard for industrial Ethernet, covering all the requirements of automation technology. It can be applied for production automation, process automation, or drives, and also supports functional safety using profile PROFIsafe over PROFINET.

PROFINET is switched Ethernet. To make out cost-effective and easy, many PROFINET devices already have a switch with two or more ports integrated into them. For instance, NX3008 has a pair of switched ports (NET2 + NET3), so the user does not need to buy external switches.

PROFINET is 100% Ethernet compatible as per IEEE standards and fulfills system requirements with its flexible topology. Line, ring, and star structures are easy to implement with copper and fiber-optic cables. For instance, NX3008 has a pair of switched ports (NET2 + NET3) that support ring structures using MRP as a media redundancy protocol.

Conformance Classes

As a broad standard, PROFINET offers a large number of functions. These functions are divided into Conformance Classes in a clear way. They provide a practical summary of the various different minimum property values.

There are three Conformance Classes (CC) that build upon one another as shown in the following figure.

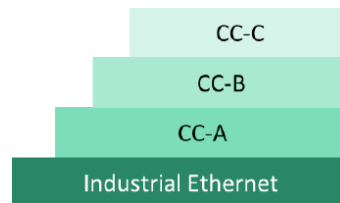


Figure 2-1. Conformance classes

Conformance Class A (CC-A) provides basic functions for PROFINET with real-time (RT) communication. All IT services can be used without restriction. Typical applications are found, for example, in business automation. Wireless communication is specified for this class.

Conformance Class B (CC-B) expands PROFINET to include network diagnostics using IT mechanisms and topology information of the network.

Conformance Class C (CC-C) describes the basic functions for devices with hardware-supported bandwidth reservation and synchronization Isochronous Real-Time (IRT) communication and is thus the basis for isochronous applications. Special switches and Ethernet ports are required for CC-C.

System Model and Device Classes

PROFINET follows the provider/consumer model for data exchange. This means that both the IO controller and IO device spontaneously send cyclic I/O data independently.

There are three device classes in a PROFINET network:

- IO controller (IOC): This is typically the Programmable Logic Controller (PLC) in which the automation program runs. The IO controller provides output data to the configured IO devices in its role as provider and is the consumer of input data.
- IO device (IOD): An IO device is a distributed IO field device connected to one or more IO controllers via PROFINET. The IO device is the provider of input data and the consumer of output data from the IO controller.
- IO supervisor (IOS): This can be a programming device (PG), personal computer (PC) or human-machine interface (HMI) device for commissioning or diagnostic purposes.

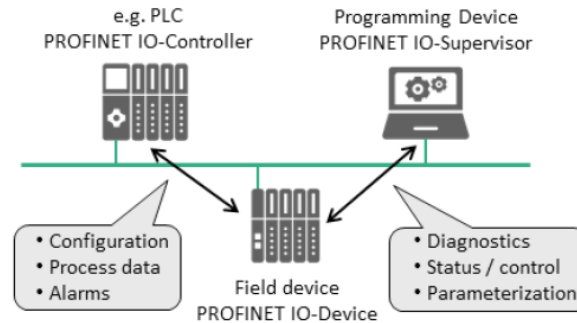


Figure 2-2. Device classes

A system unit contains at least one IO controller and one or more IO devices. IO supervisors are usually integrated only temporarily for commissioning or troubleshooting purposes.

Device Model of an IO Device (IOD)

The technical and functional options of all field devices are described using the device model, which is oriented toward a modular device, according the following figure.

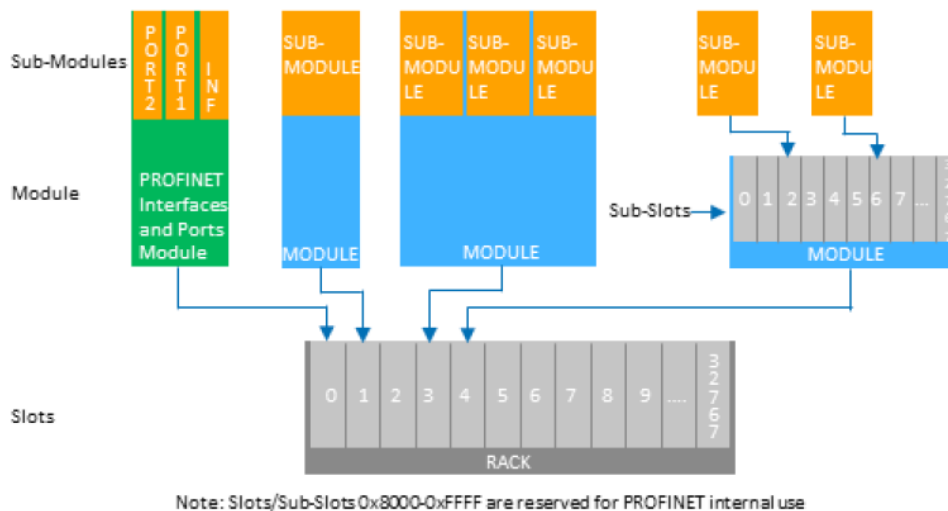


Figure 2-3. Device model of an IOD

PROFINET differentiates between **compact field devices**, in which the degree of expansion is already specified in the as-delivered condition and cannot be changed by the user, and **modular field devices**, in which the degree of expansion can be customized for a specific application when the system is configured. In its logical structure, a PROFINET field device is always modular in design. Modularity in the logical sense, however, does not require actual modularity in the electrical and mechanical design sense.

An IOD is usually comprised of a communication module with an Ethernet interface and (physical or virtual) I/O modules assigned to it. The assigned I/O modules handle the actual process data traffic. The access point for communication (Ethernet interface with data processing) is called the **DAP (Device Access Point)**.

The following structures are standardized for an IOD:

- The device model consists of **slots**, **subslots**, **modules**, **submodules** and **channels**.
- The **slot** designates the insertion slot of a module in an IO field device. A field device usually has two or more slots.
- A **module** is comprised of one or more **submodules** or provides available **subslots** into which **submodules** can be inserted.
- The modules themselves have no task other than to provide structuring. The actual inputs and outputs (**channels**) are implemented in its submodules. The granularity of the channels (bitwise, byte-wise, or word-wise division of IO data) is determined by the manufacturer. Acyclic services always address submodules. Therefore, a module always contains at least one submodule.

The **index** specifies the data within a submodule inserted into a slot/subslot which can be read or written acyclically using read/write services. For example, parameters can be written to a module, or manufacturer-specific module data can be read out on the basis of an index. Specific indexes are defined in the PROFINET standard. Additional indexes can be freely defined by the manufacturer.

The submodule is the owner of the user data, diagnostics, channels, actual configuration, records, and I&M data. Cyclic IO data of the submodule in the device is addressed by specifying the slot/subslot combination of the insertion slot. They can be freely defined by the manufacturer. For acyclic data communication via read/write services, an application can specify the data of the submodule to be addressed using slot, subslot and index.

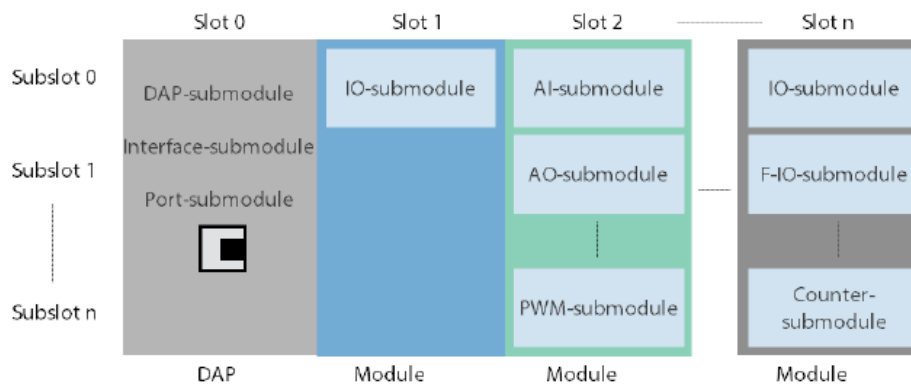


Figure 2-4. Addressing of IO data in PROFINET on the basis of slots and subslots

To avoid user profiles (e.g. for PROFIdrive, weighing and dosing) from having competing definitions, each of the profiles is assigned its own **Application Process Identifier (API)**.

Station Names and IP Addresses (DCP Protocol)

In a PROFINET system, each IOD receives a symbolic name (station name) which uniquely identifies the IOD within the IO system. In addition, the IOD needs an IP address. In the controller project, the user must inform both the station name and the IP address for each IOD.

During commissioning of IOD, the user must set the station name in a retentive way, so that the name is not lost in case of power-down of IOD. This name assignment can be done using an engineering tool, like the programming system of a controller or some generic tool (e.g.: Profinet Commander®). The DCP protocol (Discovery and Basic Configuration Protocol) is used by these engineering tools for setting the station name of the IOD.

Later, when the controller application starts, the controller uses the DCP protocol again, this time for finding the IOD searching by station name, and then assigning the corresponding IP address configured in the controller project for this IOD.

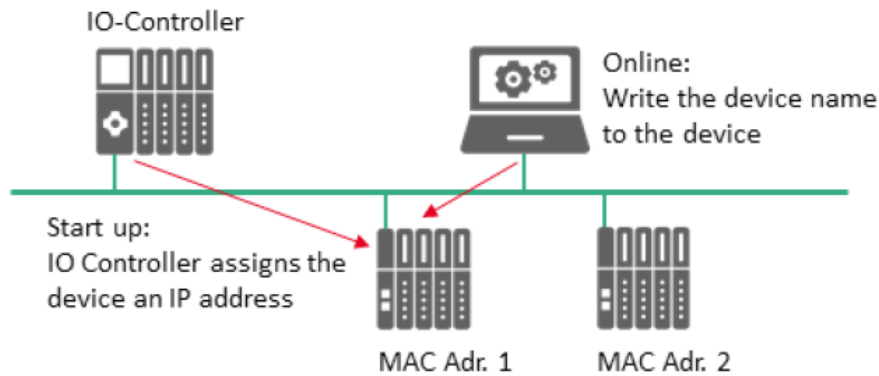


Figure 2-5. Station name and IP address assignment

Neighborhood Detection (LLDP Protocol)

In the controller project, the user may configure the neighbors connected to each PROFINET port of each IOD and IOC. This is normally referred to as "topology configuration". Basically, the user configures which station name and port number are connected to each port of a given IOD or IOC. For instance, in the following figure, the user informs that port001 of "alpha" is connected to port004 of "switch 1", and so on for all ports of all IODs and IOCs.

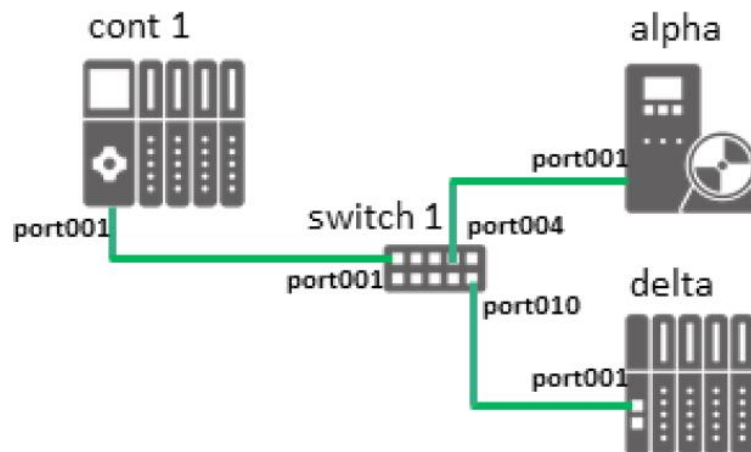


Figure 2-6. Neighborhood detection

IOCs and IODs then use the LLDP protocol (Link Layer Discovery Protocol) for checking if their connections are good, that is, if the right neighbor is connected to each port.

This methodology provides several advantages:

- Diagnostics are transmitted from IODs to IOC or IOS to report connection failures (missing connections or wrong connections).
- Device replacement without using an engineering tool for naming the IOD (see sections **Station Names and IP Addresses (DCP Protocol)** and **Device Replacement without Engineering Tool**).

Device Replacement without Engineering Tool

If all neighbors of an IOD are declared in the topology configuration of the controller project, it is possible to replace a defective IOD without using an engineering tool for naming the spare IOD being inserted.

The defective IOD can be replaced by a brand-new spare IOD with an empty station name (see section **Station Names and IP Addresses (DCP Protocol)**).

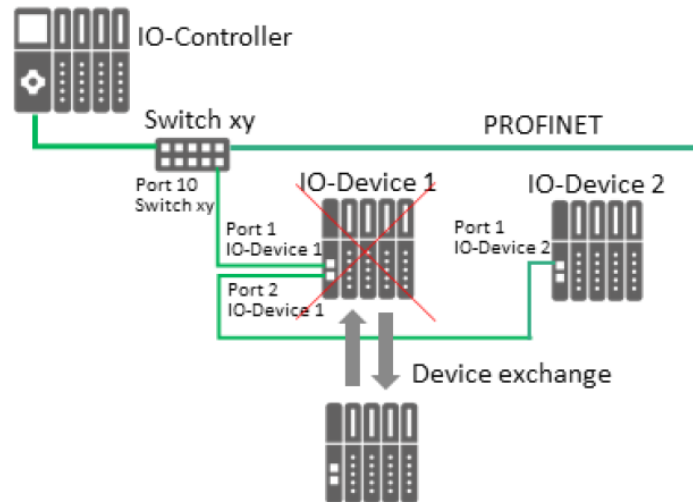


Figure 2-7. Device replacement without engineering tool

When the spare IOD is inserted in the place of the defective IOD, it receives the station name automatically from IOC, based on the configured topology. After this, IOC also adjusts the IP address.

ATTENTION:

This is an optional feature supported by IOCs from Nexto series. However, some third party IOCs and IODs may not support this feature.

Media Redundancy (MRP Protocol)

The MRP protocol allows the configuration of rings for improving the reliability of a PROFINET network. In case of failure of a connection between two devices, the typical reconfiguration time is less than 200 ms. Error-free operation of an automation system involves a Media Redundancy Manager (MRM) and several Media Redundancy Clients (MRC) arranged in a ring as shown in the following figure.

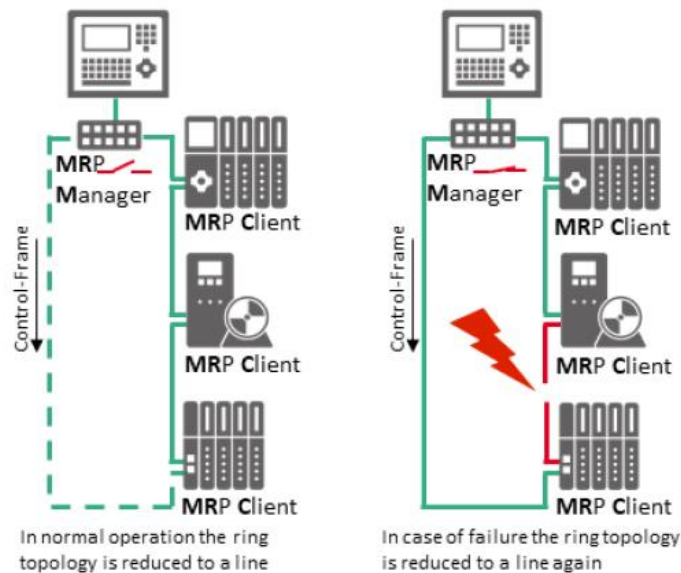


Figure 2-8. Working principle of MRP protocol

The task of a Media Redundancy Manager (MRM) is to check the functional capability of the configured ring structure. This is done by sending out cyclic test frames (control frames). As long as it receives all of its test frames back, the ring structure is intact. For all other frames, the redundancy manager logically opens the ring. Through this behavior, an MRM converts a ring structure into a line structure and thus prevents a loop (the circulation of frames).

A Media Redundancy Client (MRC) is a switch that acts only as a "passer" of frames and generally does not assume an active role. In order for it to be integrated into a ring, it must have at least two switch ports. If an MRC detects a failure like a link down, it may send a special MRP message to MRM informing about this failure, because this reduces the reconfiguration time.

The devices in a ring (MRP domain) can be configured with three different roles:

- Client (MRC)
- Manager (MRM)
- Manager-auto (alternates between MRC and MRM)

A further improvement of reliability can be achieved using several managers in the same ring, predicting the possibility of failure of the manager. In this case, all the possible managers must be configured as manager-auto. Only one of the devices configured as manager-auto will really work as a manager, and the remaining will work as clients.

PROFINET Switches

Normal switches can be used in a PROFINET network, but the usage of PROFINET certified switches supporting the LLDP protocol is strongly recommended because they offer several advantages:

- A PROFINET switch is an IOD, so it can be configured in the controller project (station name, topology, etc). The IOC will configure the switch at startup. A non-PROFINET switch must be configured by the user using different tools (e.g.: web-browser).
- As an IOD, it can send PROFINET diagnostics to the controller (for instance, wrong or missing connections detected using the LLDP protocol).
- If the switch supports MRP protocol, it can be used for attaching devices that do not support MRP to other devices connected in a ring.
- Connections between a non-PROFINET switch and other IODs cannot be configured in topology and therefore diagnostics for these connections will not be reported properly.

- IODs connected to a non-PROFINET switch cannot take the benefit described in the section **Device Replacement without Engineering Tool**.

The following figure shows an example where a 4-port switch (IOD4) is used for connecting two devices with a single port (IOD3 and IOS) to the other three devices in an MRP ring (IOC, IOD1, IOD2). The switch uses ports P1 and P2 for connection to the ring, and ports P3 and P4 for connecting to the devices which have a single port.

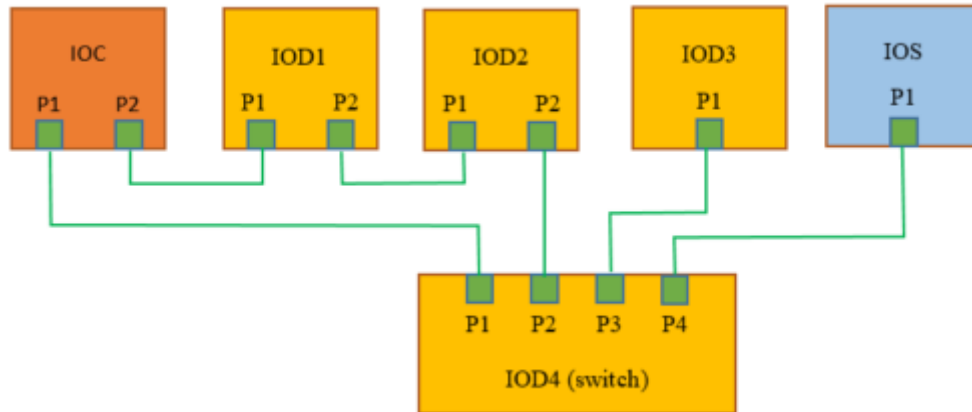


Figure 2-9. Example of usage of PROFINET switch

Diagnostics

PROFINET includes intelligent diagnostic concepts for field devices and networks. Acyclically transmitted diagnostic data provides important information on the status of the devices and communication and enables a user-friendly representation of the network.

This manual dedicates an entire chapter to diagnostics (see chapter **Diagnostics**). Inside this chapter, there is an introduction to PROFINET diagnostics (see section **Introduction to PROFINET Diagnostics**).

I&M Data

Identification and Maintenance (I&M) data supports unique identification of the devices, modules, and submodules and their versions. This identification information is an important basis for maintaining the system and for asset management.

This information is specified in the I&M data structures. The I&M functions are subdivided into six different blocks (I&M0 to I&M5) and can be addressed separately using their index.

This manual dedicates an entire chapter to I&M data (see chapter **I&M Data**).

Device Descriptions (GSDML Files)

To enable system engineering, the GSDML files of the field devices to be configured are required.

A GSDML file is a XML-based file with a "General Station Description" that describes the properties and functions of the PROFINET field device, including its modules and submodules. It contains all data relevant for engineering as well as for data exchange with the field device.

In particular, GSDML files enable the engineering of a PROFINET system or PROFINET application without the devices used having to be physically present.

The GSDML file is essential for a PROFINET field device, as engineering is impossible without it. Every manufacturer of a PROFINET field device must create an associated GSDML file.

The engineering tool obtains knowledge about the device using the GSDML data. The GSDML file is read into the engineering tool (e.g. programming device) once for this purpose. The field device can then be configured, for example, from the product catalog of the engineering tool.

Basic Communication Functions

The basic functions of Conformance Class A include:

- The cyclical exchange of I/O data with real-time properties.
- Acyclic data traffic for reading and writing need-based data (parameters and diagnostic data), including the identification and maintenance function (I&M) for reading out device information.
- A flexible alarm model for signaling device and network errors.

Cyclic Data Exchange

Cyclic I/O data is transmitted unacknowledged as real-time data between the provider and consumer in a parameterizable time frame. The update time can be specified individually for connections to the individual devices and is thus adapted to the requirements of the application. Different update times can be selected for the input and output data within the range of 4 ms to 512 ms for communication with IOCs of Nexto series.

For each IOD, there are two independent cyclic I/O data transmissions:

- From IOC to IOD, used for sending output data (provider = IOC, consumer = IOD);
- From IOD to IOC, used for sending input data (provider = IOD, consumer = IOC).

For each IOD, three parameters configure when I/O data is exchanged:

- Send clock: this is a basic cycle when any transmission can start;
- Reduction rate: the product between Send Clock and Reduction Rate determines the real update cycle of transmission.
- Phase: this is a number between 1 and Reduction Rate, and defines when the transmission will start inside the update cycle. The Mastertool programming system also allows to configure "-" instead of a number, and this means that it will configure the phase automatically for distributing the traffic equally between all IODs.

The following figure shows an example with three devices. For obtaining the results shown in this figure, the following configurations should be done for each device:

- IOD1: Send clock = 4 ms, Reduction rate = 1, Phase = 1 (update cycle = 4 ms);
- IOD2: Send clock = 4 ms, Reduction rate = 2, Phase = 1 (update cycle = 8 ms, at phase 1 of 2);
- IOD3: Send clock = 4 ms, Reduction rate = 2, Phase = 2 (update cycle = 8 ms, at phase 2 of 2);

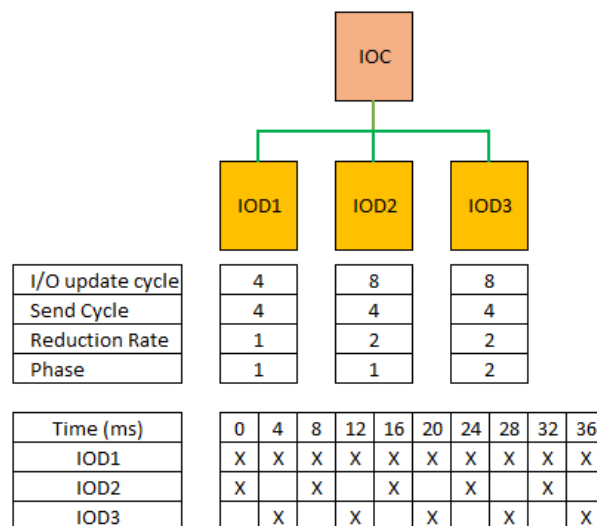


Figure 2-10. Example of cyclic I/O data exchange

Acyclic Data Exchange

Acyclic data exchange using the "record data CR" can be used for parameter configuration and other configuration of the IO devices or reading out status information. This is accomplished with read / write services based on UDP/IP, in which the data records are distinguished by the index.

In addition to the data records which are freely definable by device manufacturers, the following system data records are defined:

- Diagnostic information about the network and the devices can be read out by the user from any device at any time.
- Identification and Maintenance (I&M) data supports unique identification of the devices, modules, and submodules and their versions. This identification information is an important basis for maintaining the system and for asset management.

Device/Network Diagnostics

A status-based maintenance approach is currently gaining relevance for operation and maintenance. It is based on the capability of devices and components to determine their state and to communicate them using agreed-upon mechanisms. A system for reliable signaling of alarms and status messages by the IO devices to the IO controller was defined for PROFINET for this purpose.

This alarm concept covers both system-defined events (such as removal and insertion of modules) as well as signaling of faults that were detected in the controller technology utilized (e.g. defective load voltage or wire break). This is based on a state model which defines "good" and "defective" states as well as (optionally) the "maintenance required" and "maintenance demanded" pre-warning levels. A typical example of "maintenance required" is the loss of media redundancy. When a redundant connection is lost, "maintenance required" is signaled, but all nodes can still be reached.

Alarms are transmitted spontaneously from IOD to IOC only when some failure appears or disappears.

More information about diagnostics and alarms is provided in chapter **Diagnostics**, section **Introduction to PROFINET Diagnostics**.

IOPS and IOCS

IOPS and IOCS are status information transmitted between IOC and IOD together with cyclic data exchange (see section **Cyclic Data Exchange**).

Note that the IOC provides outputs and consumes inputs, while the IOD provides inputs and consumes outputs.

IOPS means "IO provider status" and is transmitted by the provider:

- The IOC sends one byte of IOPS together with outputs for each submodule that contains outputs;
- The IOD sends one byte of IOPS together with inputs for each submodule that contains inputs.

IOCS means "IO consumer status" and is transmitted by the consumer:

- The IOC sends one byte of IOCS together with inputs for each submodule that contains inputs;
- The IOD sends one byte of IOCS together with outputs for each submodule that contains outputs.

The following table shows the possible values for IOPS and IOCS.

Value	Meaning
128	Good The associated submodule data is valid and can be used.
0	Bad by subplot The associated submodule data is not valid and should not be used for process control. The submodule detected the issue.
32	Bad by slot The associated submodule data is not valid and should not be used for process control. The module detected the issue.
64	Bad by IO Device The associated submodule data is not valid and should not be used for process control. The IO Device detected the issue.
96	Bad by IO Controller The associated submodule data is not valid and should not be used for process control. The IO Controller (VIM) detected the issue.

Figure 2-11. Possible values for IOPS and IOCS

IOPS and IOCS transmitted from IOD to IOC normally allocate %I variables in the Nexto series IOCs (input image variables). These variables can be used as diagnostics for submodules containing inputs or outputs. For instance:

- If IOPS of a submodule containing inputs is different from 128 (good), this means that the value of these inputs cannot be totally trusted.
- If IOCS of a submodule containing outputs is different from 128 (good), this means that the physical value in these outputs may not correspond to the outputs transmitted by the IOC.

Therefore, it is important to test values of IOPS and IOCS received from each submodule of each IOD. Values different from 128 (good) indicate some problem that may require automatic safety actions and maintenance.

In the other hand, IOPS transmitted by the IOC can be used for switching the outputs of IODs to some special state, if IOPS is different from good. For instance, when the controller goes to STOP, it normally should send IOPS equal 96 (bad by controller), causing outputs going to safe state in IODs.

About IOCS transmitted by the IOC, it is not that useful.

3. Installation

This chapter describes some of the typical PROFINET topologies. After this, it presents guidelines for field installation using copper cables or fiber optics cables.

Typical Topologies

This section shows several typical topologies possible in a PROFINET network. Each topology consists of one controller (IOC), some field devices (IODs), and sometimes auxiliary switches.

Other topologies may be possible. In case of doubt, contact ALTUS support.

ATTENTION:

Some limits for topologies are pointed in tables of section **Features of Nexto Controllers with PROFINET**. For instance, there is a limited number of IODs per IOC.

Line Topology using Embedded Switches

The following figure shows a line topology, where IOD1 and IOD2 have embedded two port switches. IOC and IOD3 can have single ports because they are at the ends of the line topology.

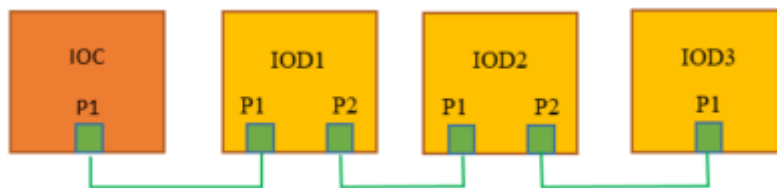


Figure 3-1. Line topology using embedded switches

Star Topology using Auxiliar Switch

The following figure shows a star topology, where IOC and most IODs have single ports. Only the auxiliary switch (IOD4) must have multiple ports. A non-certified PROFINET switch can be used, however, in this case, some advantages offered by PROFINET switches are lost (see section **PROFINET Switches**).

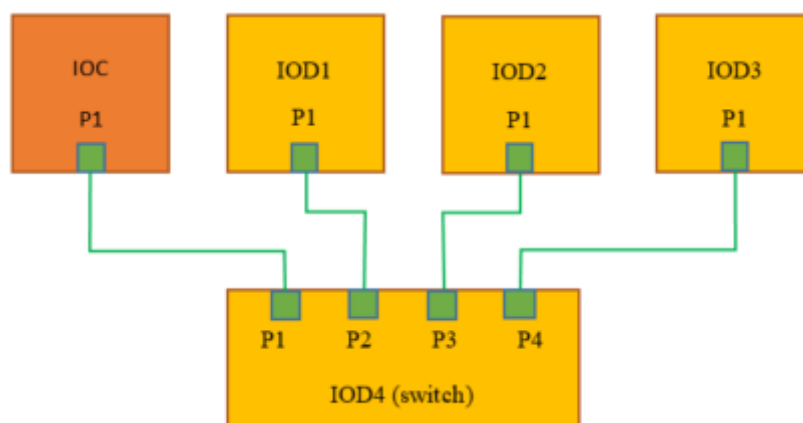


Figure 3-2. Star topology with auxiliary switch

Mixed Line and Star Topology using Auxiliar Switch

The following figure shows a mixed topology (line and star). IOD2 has an embedded 2 port switch. In addition, IOD4 is the auxiliary switch for the star topology. A non-certified PROFINET switch can be used, however, in this case, some advantages offered by PROFINET switches are lost (see section **PROFINET Switches**).

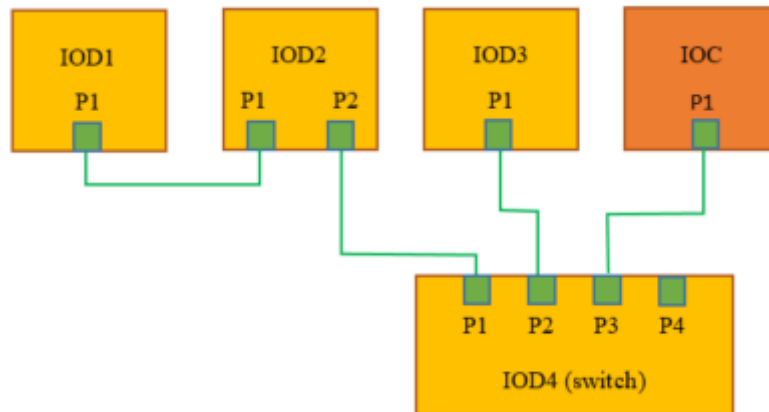


Figure 3-3. Mixed line and star topology with auxiliary switch

Ring Topology using Embedded Switches

The following figure shows a ring topology, where IOC and IODs have two switched ports capable of participating in an MRP ring. The IOC will typically play the role of MRP manager, while the IODs will play the role of MRP clients.

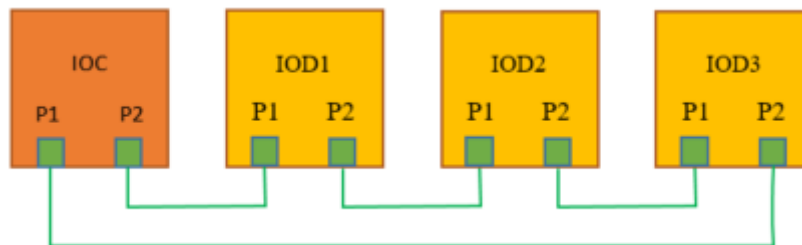


Figure 3-4. Ring topology using embedded switches

Mixed Ring and Star Topology using Auxiliar Switch - IOC as Ring Manager

The following figure shows a mixed ring and star topology. The IOC will typically play the role of MRP manager, while the IODs (IOD1, IOD2 and IOD4) will play the role of MRP clients. IOD3 has a single port, therefore it needs the auxiliary switch (IOD4) for connecting to this PROFINET topology. A certified PROFINET switch supporting MRP-client must be used.

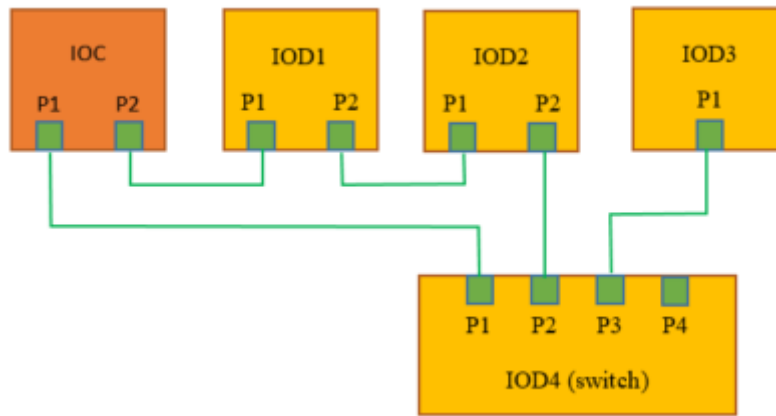


Figure 3-5. Mixed ring and star topology with auxiliary switch - IOC as ring manager

Mixed Ring and Star Topology using Auxiliar Switch - Switch as Ring Manager

The following figure shows a mixed ring and star topology. The switch (IOD4) will typically play the role of MRP manager, while other IODs (IOD1, IOD2, and IOD3) will play the role of MRP clients. IOC has a single port, therefore it needs the auxiliary switch (IOD4) for connecting to this PROFINET topology. A certified PROFINET switch supporting MRP-manager must be used.

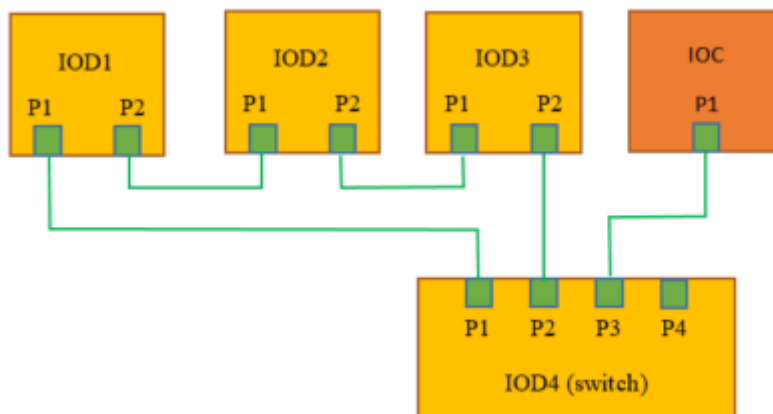


Figure 3-6. Mixed ring and star topology with auxiliary switch as ring manager

Installation Guidelines for Copper Cables

Copper cables are the cheaper way for connecting PROFINET devices. However, there are situations where fiber optic cables must be used instead of copper cables (see section **Installation Guidelines for Fiber Optic Cables**).

Maximum Length of Copper Cables

The maximum length of a copper cable is 100 meters. For longer lengths, a suitable fiber optic cable must be used instead.

Transmission Speed

PROFINET devices using copper cables must support at least 100 Mbps, full-duplex, with auto-crossover and auto-negotiation. Faster speeds are supported by some devices.

Some Nexto series devices can also operate at 1000 Mbps: NX3008 (only port NET1).

Types of Copper Cables

Selection of PROFINET copper cables products shall comply with the criteria for industrial machinery and equipment. These kinds of cables can be installed in challenging environment conditions (vibrations, EMI, extreme temperatures, humidity, dust, and more).

There are several types of PROFINET copper cables considering the installation environment. The appropriate cable must be selected considering factors like temperature, humidity, chemical contaminants, ultra-violet light resistance, flame resistance, rodent animals, crush and tread resistance, flexing or bending resistance, among others. The outer sheath and other characteristics may change according to the installation environment.

The following are examples of types of PROFINET copper cables:

- PROFINET PE cable (PolyEthylene sheath);
- PROFINET ground cable, with and additional PE sheath;
- Trailing cables;
- Festoon cables;
- Flame retardant non-corrosive cables (FRNC);

RJ45 Connectors for Nexto Series Devices

Nexto series devices have RJ45 female connectors for PROFINET connection, suitable for IP20 environment.

Consider the following requirements for selecting a matching RJ45 male connector for the cable:

- It shall comply with the criteria for industrial machinery and equipment.
- It must have a metal enclosure to be connected to the cable shield.
- It must provide a secure connection with the female connector in the device. This is important to avoid contact intermittency due to vibrations.
- It must match with the cable type (gauge).

The pin assignment is shown in the following figure (note that two color schemes are used - T568A and T568B):

Function 4 pair	Wire Colours	Wire Colours	Contact Assignment RJ-45
	T568A	T568B	
TD/RD 1	White/Orange	White/Green	3
	Orange	Green	6
TD/RD 2	White/Green	White/Orange	1
	Green	Orange	2
TD/RD 3	White/Blue	White/Blue	5
	Blue	Blue	4
TD/RD 4	White/Brown	White/Brown	7
	Brown	Brown	8

Figure 3-7. Pin assignment in RJ45 connectors

The following figure shows a schematic with a T568B color scheme. Note that shield must be connected in the metal enclosure of the RJ45 male connectors.

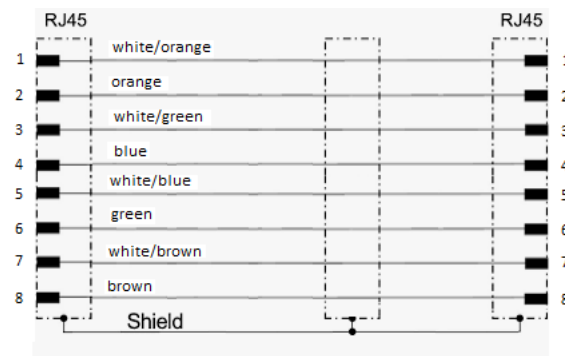


Figure 3-8. Schematic of a cable with two RJ45 connectors (T568B color scheme)

Shielding and Equipotential Bonding

To ensure the EMC (electromagnetic compatibility) of the PROFINET system, the entire communication route is shielded up to and into the device.

PROFINET operates with shielded copper cables whose shield is routed via the connector to the device. Metal connectors and device sockets are therefore used. In the device, the shield is connected low-impedance to the functional grounding.

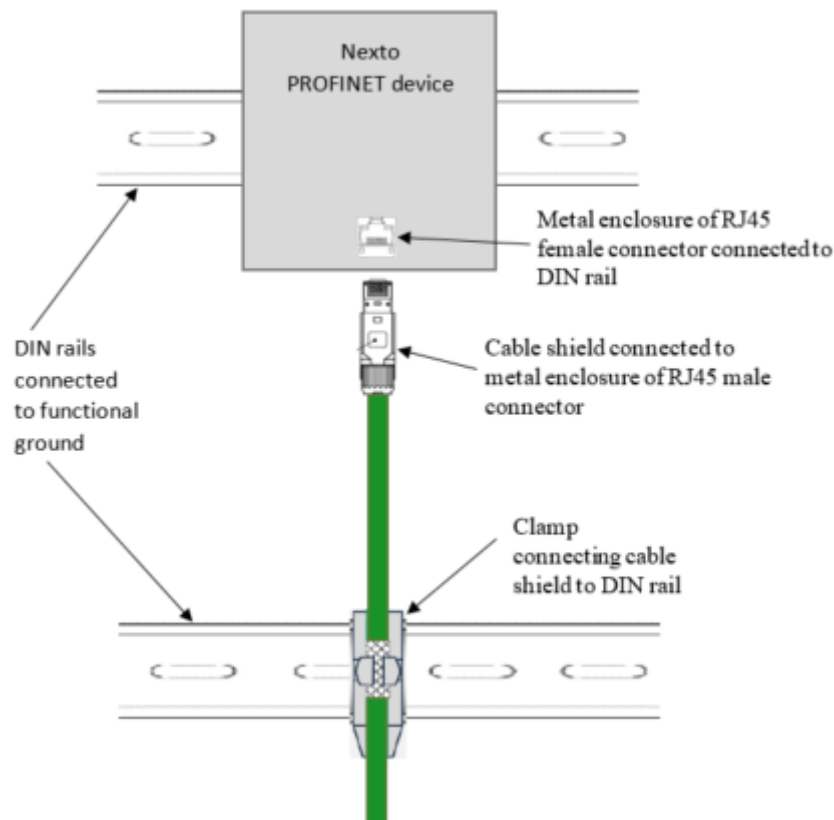


Figure 3-9. Shielding PROFINET communication route inside a cabinet

ATTENTION:

If a patch panel is used inside the cabinet, a shielded patch panel must be selected.

Potential differences can exist between functional grounds of PROFINET devices interconnected by a cable. These potential differences can lead to currents flowing through the cable shield and interfere with the communication as the cable shields are connected at both ends of the cable.

To prevent this, the installation should be provided with a low-resistance equipotential bonding system. Several types of equipotential bondings can be implemented. Some of them may combine functional grounding with protective grounding. The following figure shows some examples.

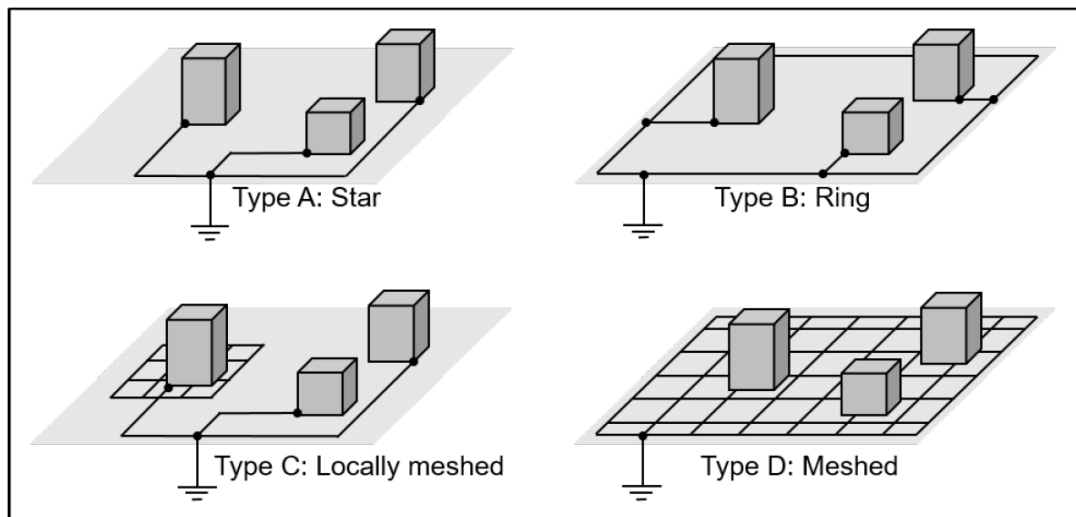


Figure 3-10. Examples of equipotential bonding systems

Routing Copper Cables

This section presents some additional hints for routing PROFINET copper cables. These hints aim to avoid electromagnetic interference, contact intermittency, and damages to the cable.

Minimize Pickup of Electromagnetic Interferences

In order to minimize pickup of electromagnetic interferences, lay PROFINET cables separately from another plant wiring (especially power supply cables). Parallel routing of PROFINET and other types of cables should be minimized and the distance between these different cables should be maximized.

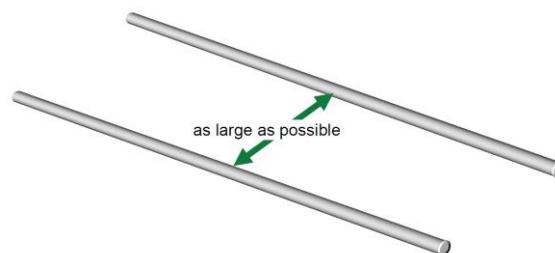


Figure 3-11. Maximize distance between parallel paths

Where cables in different categories have to cross, they should always cross at right angles. Try to avoid running cables in different categories in parallel even for short distances.

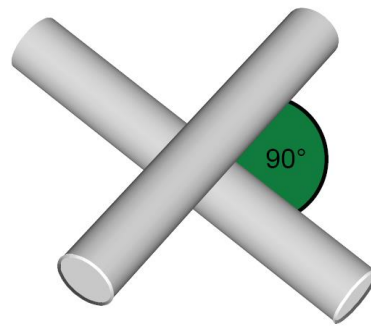


Figure 3-12. Cross PROFINET cables with other cables using right angles

In case there is not sufficient space to achieve the required spacing between cables of different categories, the cables must be installed in separate, metallic, conducting ducts. Each duct should only carry cables of the same category. These channels can be arranged directly side by side.

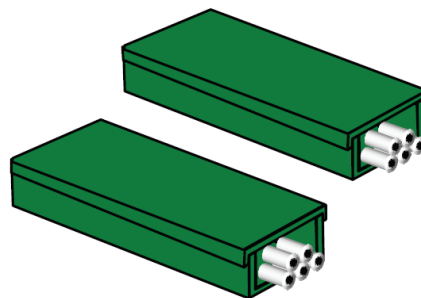


Figure 3-13. Metallic conducting ducts for different categories of cables

Some additional measures can be taken, like using shielded power supply cables.

If a common metallic cable duct is used for all categories, different cable categories must be separated by using metallic partitions. These partitions must be electrically connected to the channel over a large area.

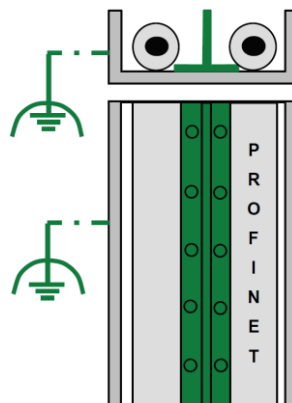


Figure 3-14. Partitions inside metallic conducting ducts for different categories of cables

Connect metallic conductive cable trays or ducts to the equipotential bonding system of the building.

Cable Routing Outside Buildings

It is recommended that fiber optic cables are used for PROFINET links outside buildings. If using fiber optic cables is not possible for some reason, initially consider the rules described in the section **Minimize Pickup of Electromagnetic Interferences**.

Furthermore, there are some additional rules.

Only use approved cables for installations outside buildings. This applies in particular to underground cables.

Lay the cables on metallic conductive cable trays. Connect the joints of the cable trays using a large conductive area. Connect the cable trays to the functional ground.

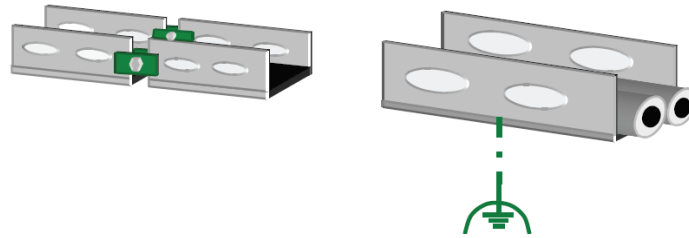


Figure 3-15. Metallic conductive cable trays between buildings

Mechanical Protection of Copper PROFINET Cables

Mechanical protection is intended to prevent wire breaks or short circuits on PROFINET cables and mechanical damage to the cable sheath and shield.

If the PROFINET cable cannot be laid in a cable tray, use a cable protection tube.

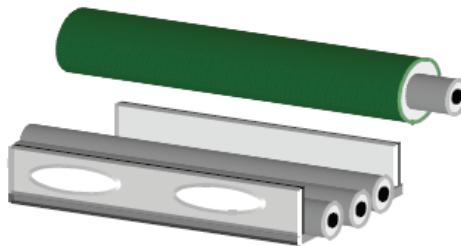


Figure 3-16. Cable tray or protection tube

In areas with heavy mechanical stress, lay the PROFINET cables in metal armored conduits. In areas with light to medium stress, plastic conduits can be used instead.

In areas where people can step or climb, PROFINET cables should be run in metal armored conduits or cable trays.

Laying Copper PROFINET Cables

During installation be aware that PROFINET cables only have limited mechanical resilience. Cables can thus be damaged or destroyed by excessive tension or pressure. Twisting or sharp bending (kinking) of the PROFINET cable can also have the same effect. The following notes will help you to avoid damage resulting from the installation of PROFINET cables.

Reel position

Store and transport the cable reel according to the picture (side view), so that the coiled cable does not entangle.

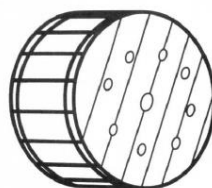


Figure 3-17. Correct position of cable reel

Temperature range for storage, installation and operation

The cable manufacturer usually specifies the minimum and maximum ambient temperatures for laying, operation and storage. The mechanical resilience of the cable will significantly decrease outside this temperature range. You will find these temperature specifications in the manufacturer datasheets. Some manufacturers even print the temperature specifications on the cable sheath. As soon as the cable is subject to mechanical stress caused by movement or during installation or when using it in drag chains, the temperature range is reduced significantly. Observe the manufacturer's instructions.

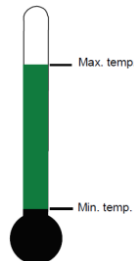


Figure 3-18. Observe temperature ranges during installation, operation, and storage

Tensile strength

The manufacturer specifies a maximum tensile strength for each cable type. The PROFINET cable can be damaged or even destroyed when this maximum tensile strength is exceeded. This is important because of the high mechanical stress when using drag chains or festoon attachment, or because of the tensile force generated when laying the cable. Please select from the table below the cable type best suited for each application.

PROFINET cable	Suited for
Non-flexible (solid core)	Stationary use without any motion
Flexible	Occasional motion or vibrations
Permanent movement	Special applications, e.g. permanent motion, vibrations or highly flexible

Table 3-1. PROFINET cables according tensile strength

Unreeling cable

Carefully unreel the PROFINET cable from the drum by hand, only. Do not apply force to pull it off.



Figure 3-19. Unreeling cable

Using pulling aid and protecting connectors

Use cable stockings when pulling in PROFINET cables. If the PROFINET cable has already been pre-assembled, protect the connector by using a plastic or metal tube to cover the connector.

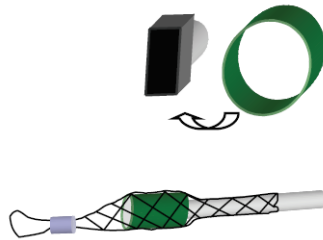


Figure 3-20. Pulling aid and connector protection

Attaching cable strain relief

For all cables that are subject to tensile stress, attach a strain relief about 30 cm from the connection point. The shield connector at the cabinet entry does not provide sufficient strain relief. As the insulation has been removed in order to allow contact with the cable shield, the cable is sensitive to strain and torsion. Assembly components for strain relief are available from various vendors.

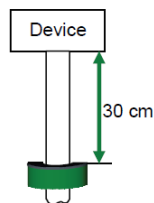


Figure 3-21. Cable strain relief

Pressure load

Do not squash the PROFINET cable, e.g. by walking or driving over it. Avoid excessive loading of PROFINET cables through pressure, e.g. caused by squashing due to improper attachment.



Figure 3-22. Avoid pressure load

Distortion

Distortion, and in particular, twisting can degrade the electrical properties of PROFINET cables. Therefore, do not distort or twist PROFINET cables whilst unreeling or laying.



Figure 3-23. Avoid distortion

Special distortion-resistant PROFINET cables can be obtained from some manufacturers. For PROFINET cables subject to frequent distortion, use flexible, distortion-resistant PROFINET cables. Robots are a typical application.

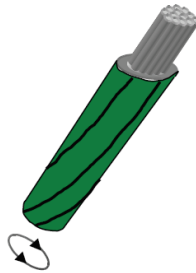


Figure 3-24. Flexible PROFINET cables (distortion-resistant cables)

Trailing and festoon cables

Use only those cables and attachments as trailing and festoon cables that have been approved for this application by the manufacturer. Appropriate components are available from various vendors.



Figure 3-25. Trailing and festoon cables

Keeping to the minimum bending radius

Keep to the minimum permissible bending radius. Falling below the minimum bending radius may damage the PROFINET cable. Please consult the manufacturer's data sheets for bending radii specifications.

When laying PROFINET cables they can be subjected to additional mechanical load caused by excessive tension. For that reason, larger bending radii are required during pulling than in the installed state. Pulling the PROFINET cable over a quoin can be particularly dangerous. It is therefore advisable to use guide pulleys. Use whenever possible ducts or cable channels with radii or chamfers That helps to prevent a kink of the cables.

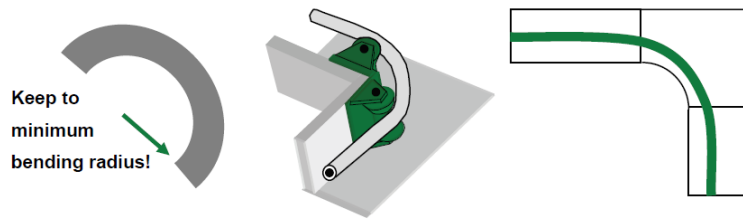


Figure 3-26. Keep to minimum bending radius

Avoiding loop formation

Always unreel the PROFINET cable straight from the drum. Never unwind the cable without rotating the drum, since that can cause looping or kinking of the cable. The cable drum should always be mounted so as to rotate as the cable is pulled from the drum. This helps to avoid the formation of loops and associated cable kinks.

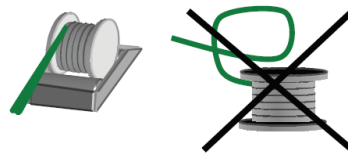


Figure 3-27. Avoiding loop formation

Avoiding sharp edges

Sharp edges may damage the PROFINET cable. You should therefore de-burr sharp edges – e.g. the cutting edges of cable ducts – using a de-burring tool or file.

Use plastic edge protectors to protect edges and angles. Use bends limiting parts in the direction of cable routing at the end of the duct or cable channel. That helps to prevent a kink of the cables.

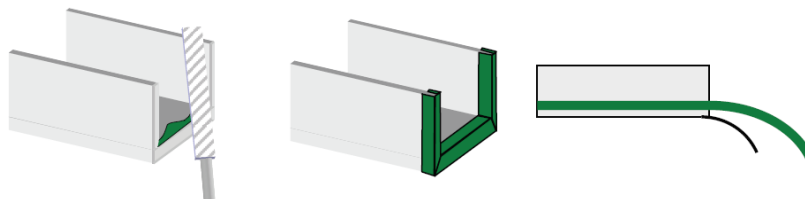


Figure 3-28. Avoiding sharp edges

Installation Guidelines for Fiber Optic Cables

In areas where electromagnetic interference fields or high potential differences can be anticipated, fiber-optic cables (FOCs) can be used for the connection of automation islands and systems. The use of fiber-optic cables eliminates electromagnetic influences and/or grounding-related equalizing currents through the shields of PROFINET copper cables.

Fiber-optic cables must also be used when the cable length exceeds 100 meters (maximum length supported by a copper cable).

The advantages of FOC transmission technology compared to copper cables are:

- FOCs generally span larger distances than copper cables;
- FOCs enable electrical separation between the connected plant sections;
- FOCs are immune to electromagnetic interference (EMI).

The following fiber types are available for selection:

- Plastic fibers plastic optical fiber (POF);
- Glass fiber polymer clad fiber (PCF);
- Glass fiber (multimode, single-mode);

With each fiber type, only a limited transmission distance can be achieved due to the respective damping and utilized wavelength of the optical signal. The following table shows some typical distances achieved by each technology. The table also shows the effect of additional connectors in the link (for instance, due to patch panels).

Fiber	No additional connectors	One additional connector	Two additional connectors
Plastic Optical Fiber (POF)	50 m	43.5 m	37 m
Plastic Cladded Fiber (PCF)	100 m	100 m	100 m
Multimode fiber	2000 m	2000 m	2000 m
Monomode fiber	14000 m	14000 m	14000 m

Table 3-2. Transmission distances for fiber optic technologies

Using Optical Media Converters

Nexto series devices do not have fiber optic interfaces, but media converters can be used, as the following figure shows.

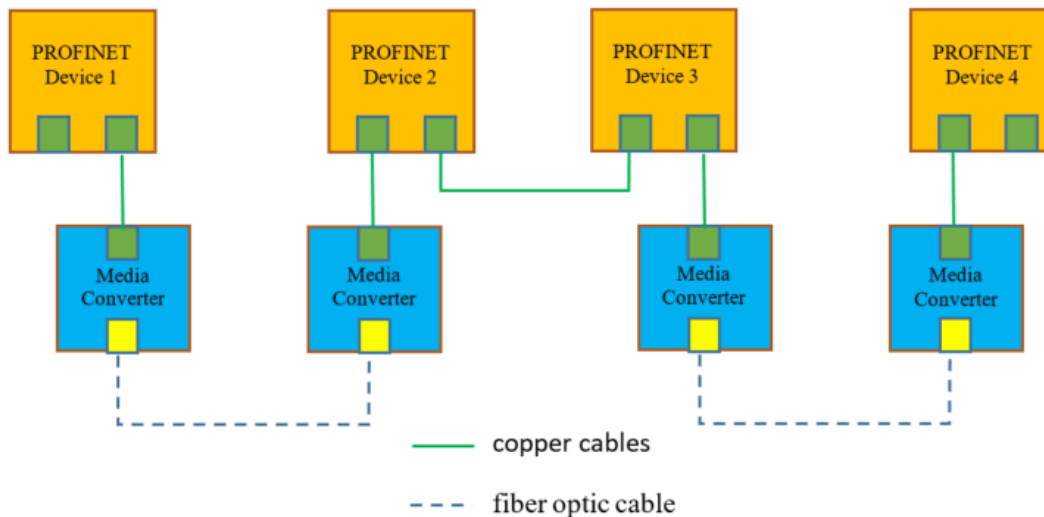


Figure 3-29. Connecting PROFINET devices optically with media converters

ATTENTION:

Contact ALTUS support for getting recommendations about approved models of media converters. It is very important to use a converter with low delay and that propagates link-down problems in the fiber optics to the RJ45 interface (link fault pass-through).

The copper cable connection between the device and the media converter must follow the guidelines presented in the section **Installation Guidelines for Copper Cables**.

From the point of view of configuration, media converters are transparent. In other words, media converters are not PROFINET devices and therefore do not appear in the configuration of the PROFINET network, described in chapter **Configuration**.

Using Switches with Optical Ports

It is also possible to connect different segments of a PROFINET network using PROFINET certified switches. The following figure shows an example.

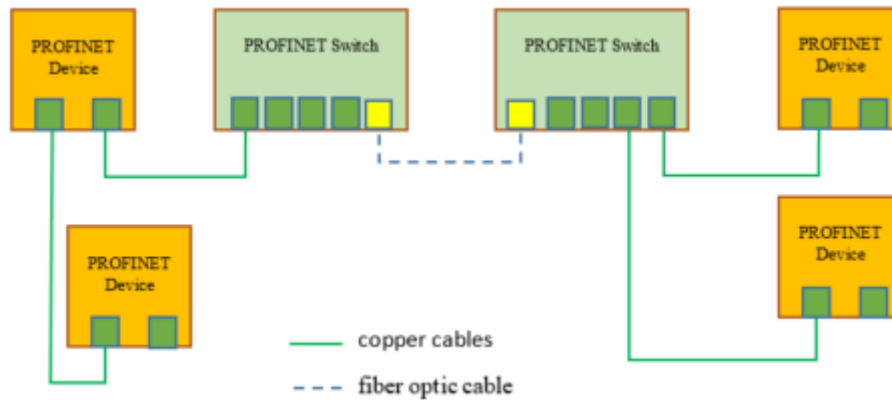


Figure 3-30. Connecting PROFINET devices optically with switches

ATTENTION:

If non-PROFINET certified switches are used, some important PROFINET features will not be available. For details, see section **PROFINET Switches**.

PROFINET switches are PROFINET devices, therefore they appear in the configuration of the PROFINET network, described in chapter **Configuration**.

4. Configuration

This chapter describes the configuration of a PROFINET network with the Mastertool Programming System for a Nexto controller (IOC). The list of Nexto controllers that support PROFINET is shown in Table 1-1.

Insert PROFINET Controller (IOC) below a Network Interface

Select the Network Interface

A PROFINET controller instance must be inserted below the network interface used by the controller as an interface to the PROFINET network. Some Nexto controllers have several network interfaces. For instance, NX3008 has three network interfaces: NET1 to NET3.

There are some rules for selecting an appropriate interface:

1. No more than one instance of PROFINET controller can be installed below a network interface (or pair of network interfaces in switch mode).
2. If the PROFINET controller does not participate in a topology that requires two ports (line or MRP ring), it must connect using a single Ethernet port, which must be configured in "Single" mode. More details about these topologies are given in the section **Typical Topologies**.
 - Possible single ports for XP300, XP315, XP325, XP340:
 - NET1 Possible single ports for NX3003, NX3004, NX3005 and NX3010:
 - NET1
 - Possible single ports for NX3020 and NX3030:
 - NET1
 - NET2
 - Possible single ports for NX3008:
 - NET1
 - NET2
3. If the PROFINET controller participates in a topology that requires two ports (line or MRP ring), it must connect using a pair of Ethernet ports, which must be configured in "Switch" mode. The loop protection mode can be "Disabled" (for line topology) or "MRP" (for ring topology). More details about these topologies are given in the section **Typical Topologies**. These special topologies can only be used with controller NX3008.
 - Possible pairs of ports for NX3008:
 - NET2+NET3

Configure Parameters of Selected Port(s)

After the network port(s) was(were) selected, some parameters must be configured for it(them).

Parameters for Single Port

Consider an example where only NET2 was selected for NX3008. In this case, the configurations are quite simple and are shown in the following figure.

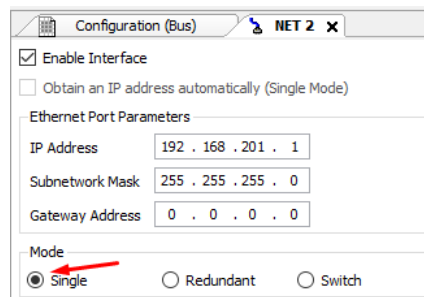


Figure 4-1. Example of parameters for single port

First, define the IP address, subnetwork mask, and gateway address for the port.

After this, select the Mode as "Single".

Parameters for Dual Ports

Consider an example where the pair of ports NET2+NET3 was selected for NX3008. In this case, the configurations must be done for NET2, as shown in the following figure. These configurations are inherited by NET3 (not necessary to configure again in NET3).

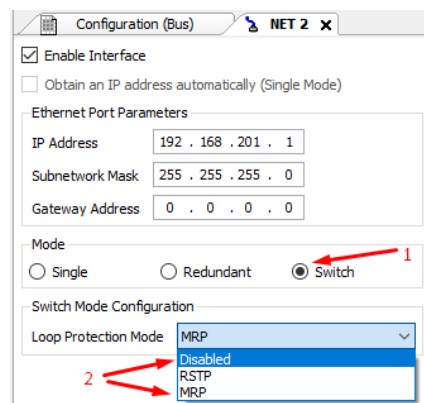


Figure 4-2. Example of parameters for dual port

First, define the IP address, subnetwork mask, and gateway address for the pair of ports.

After this, select the Mode as "Switch".

Finally, select the Loop Protection Mode, selecting one of the following options:

- Disabled, for a line topology (no loops). An example of this topology is shown in **Figure 3-1. Line topology using embedded switches.**
- MRP, for a ring topology. An example of this topology is shown in **Figure 3-4. Ring topology using embedded switches.** More details about MRP configuration can be found in the section **Configure Media Redundancy with MRP Rings.**

Notes:

- The mode "Redundant" cannot be used with PROFINET.
- The Loop Protection Mode "RSTP" cannot be used with PROFINET.

Insert Ethernet Adapter

An Ethernet adapter must be installed below the selected port. The following figure shows an example for inserting it below NET2 in NX3008 (this can be used when NET2 is used standalone in "Single" mode, or when it is paired with NET3 in "Switch" mode).

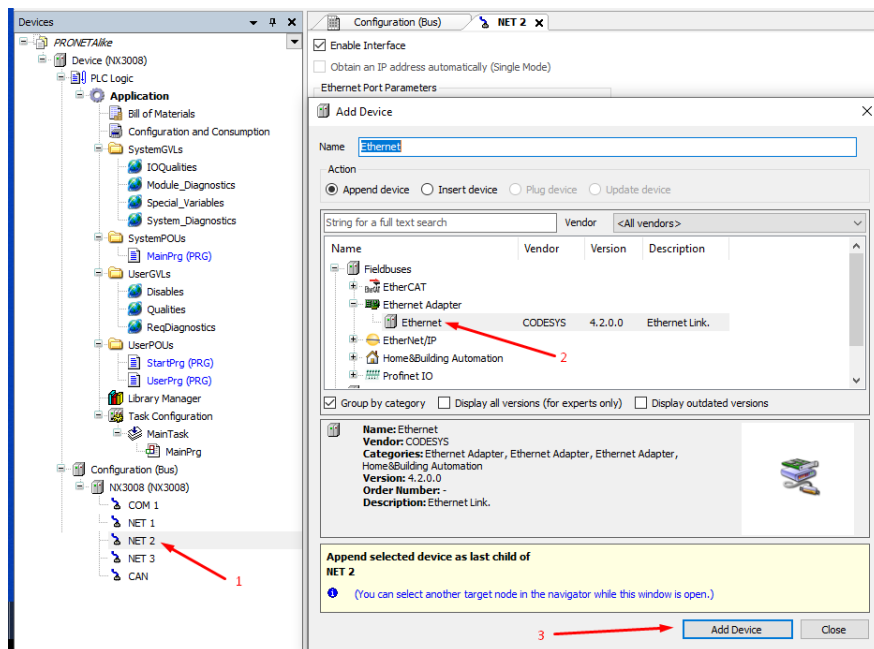


Figure 4-3. Inserting an Ethernet adapter

1. Right-click over NET2, and select "Add Device".
2. Select "Ethernet" below Fieldbuses / Ethernet Adapter.
3. Finally, press the button "Add Device".

After this, the device tree is modified as shown in the following figure.

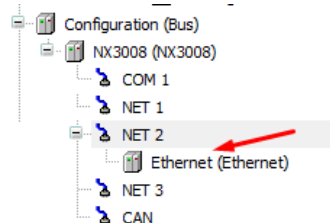


Figure 4-4. Device tree with Ethernet adapter

Insert Instance of the PROFINET Controller

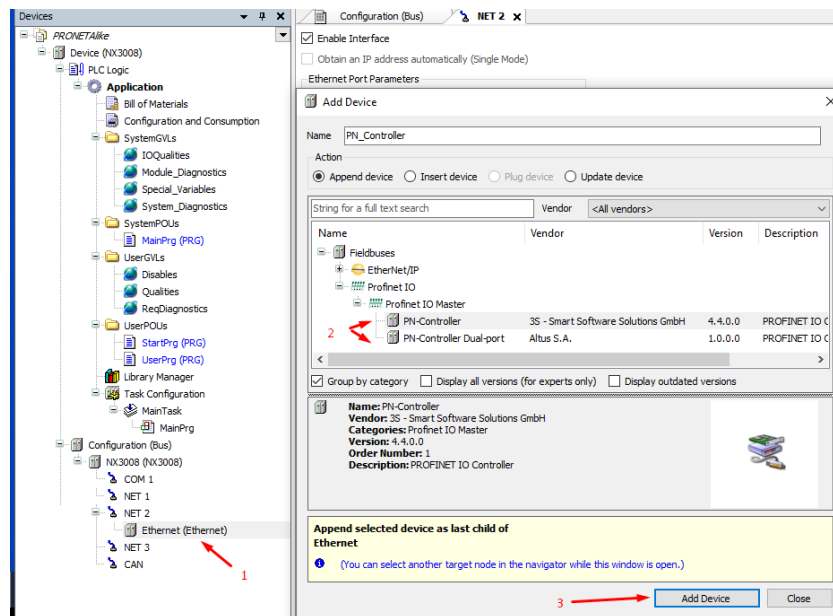


Figure 4-5. Inserting a PROFINET controller

1. Right-click on the Ethernet adapter inserted in the previous step, and select "Add Device".
2. After this, choose one of the following devices below Fieldbuses / PROFINET IO / PROFINET IO Master:
 - PN-Controller, if the selected port is not paired with another port (mode = Single).
 - PN-Controller Dual-port, if the selected port is paired with another port (mode = Switch).
3. Finally, press the button "Add Device".

After this, the device tree is modified as shown in the following figures.

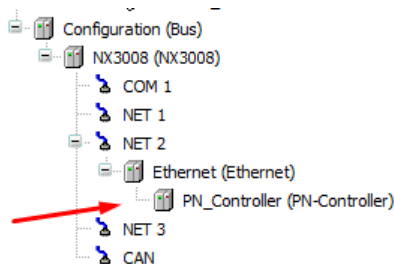


Figure 4-6. Device tree with PN-Controller

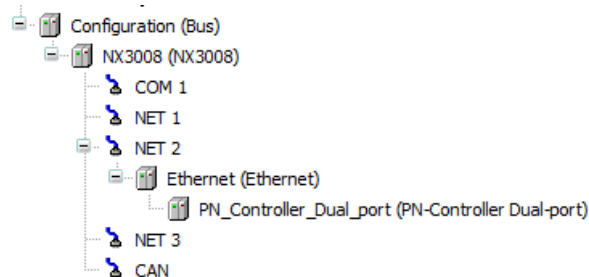


Figure 4-7. Device tree with PN-Controller Dual-port

Adjust Interval of Profinet_IOTask

After inserting a instance of a PROFINET controller in the previous step, the task Profinet_IOTask was also inserted in the device tree.

There are three options for the interval of Profinet_IOTask: 1, 2, or 4 ms. The default option is 4 ms. If possible, use 4 ms, because this reduces CPU consumption.

After double-clicking on Profinet_IOTask in the device tree, the following screen allows configuring the interval.

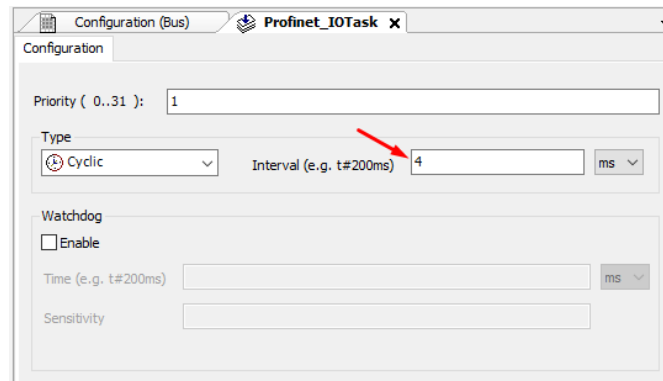


Figure 4-8. Configuration of interval of Profinet_IOTask.

Compilation errors will occur in the following situations:

1. If the value is different from 1, 2, or 4 ms.
2. If the value is greater than the update cycle of any IOD. The update cycle of an IOD is equal to "Send clock" multiplied by "Reduction Rate".
3. If the value is greater than "Send clock" and "Phase" is different from "-" or "1" for any IOD.

The meaning of the parameters "Send clock", "Reduction Rate" and "Phase" is explained in section **Cyclic Data Exchange**. Configuration of these parameters is explained in the section **Configure General Parameters of a PROFINET Device**.

Leave the checkbox "Watchdog" not marked, and "Priority" with value 1.

Configure General Parameters of the PROFINET Controller

The tab "General" of "PN_Controller" allows configuring several parameters shown in the following figure.

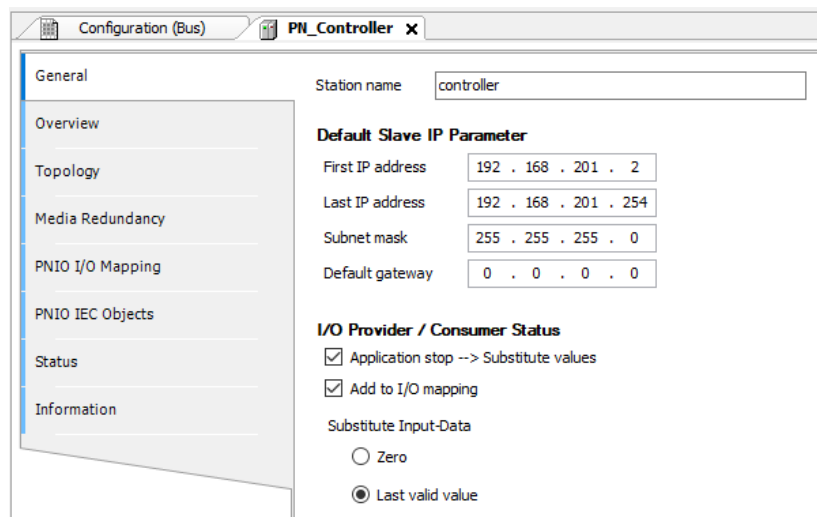


Figure 4-9. Configuration of General Parameters of PROFINET Controller

1. The **Station name** for the controller, by default, is "controller". The user can change this name. It may be necessary to change this name if another controller uses the same name in the same network. The following rules apply for creating a valid name:
 - The name is formed by one or more labels separated by ".";
 - The maximum size of each label is 63 characters;
 - The maximum size is 80 characters for the entire name (including labels and the "." separators);
 - The following characters may be used in labels:
 - "a ... z" (capital characters "A ... Z" can be used, but they will be internally converted to lowercase characters "a ... z");
 - "0 ... 9";
 - "-".
 - The character "-" cannot start or end a label;
 - The first label must start with "a ... z";
 - The first label must not have the form "port-xyz" or "port-xyz-abcde" with a, b, c, d, e, x, y, z = "0 ... 9";
 - Names must not have the form a.b.c.d with a, b, c, d = "0 ... 999".
 - Example of valid station name: "controller-1.machine-2.plant-3.vendor"
2. **First IP address** and **Last IP address** define a range for allocation of IP addresses for IODs connected to this controller. **Subnet mask** and **Default gateway** apply for addressing the subnetwork of these IODs.
3. The checkbox **Application stop --> Substitute values** is marked by default, and it is recommended to keep it marked. If the CPU of the controller goes to STOP, the controller sends IOPS equal to "bad by controller" to outputs of all IODs. This causes outputs to go to the safe state, or to some other state parameterized in the IOD (normally safe state). The meaning of IOPS is explained in section **IOPS and IOCS**.
4. The checkbox **Add to I/O mapping** is marked by default, and it is recommended to keep it marked. It causes the allocation of %I variables for IOPS and IOCS received from IODs. These IOPS/IOCS can be used as diagnostics. The meaning of IOPS and IOCS is explained in section **IOPS and IOCS**.
5. The radio-button **Substitute Input-Data** defines what to do with inputs of an IOD if this IOD gets disconnected:
 - If the option "Zero" is selected, the inputs are replaced by zero.
 - If the option "Last valid data" is selected, the inputs are frozen with the last value read while the IOD was connected.

Install Missing GSDML Files in Device Repository

It is necessary to install GSDML files of IODs in the device repository of the Mastertool Programming System, if you have not installed these GSDML files before.

For installing a GSDML file in the device repository, select the menu **Tools / Device Repository**.

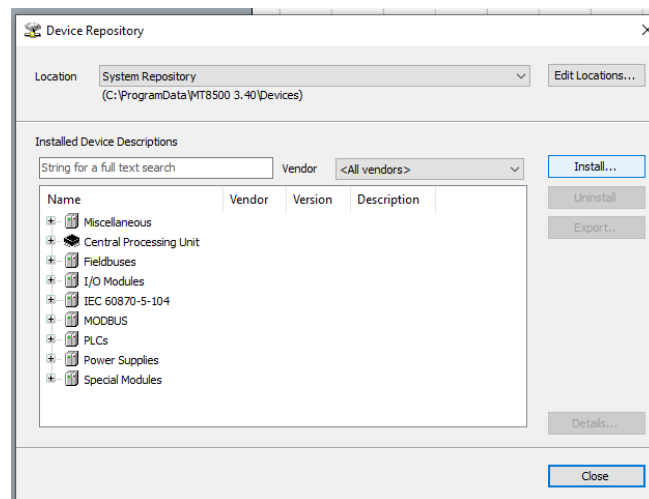


Figure 4-10. Menu Tools / Device Repository

After this, click on the "Install" button. This opens a browser window.

Navigate in this browser window until you find the GSDML file, select it, and click on the "Open" button. This starts the installation of the GSDML file. The following figure shows an example of an IOD from Phoenix®.

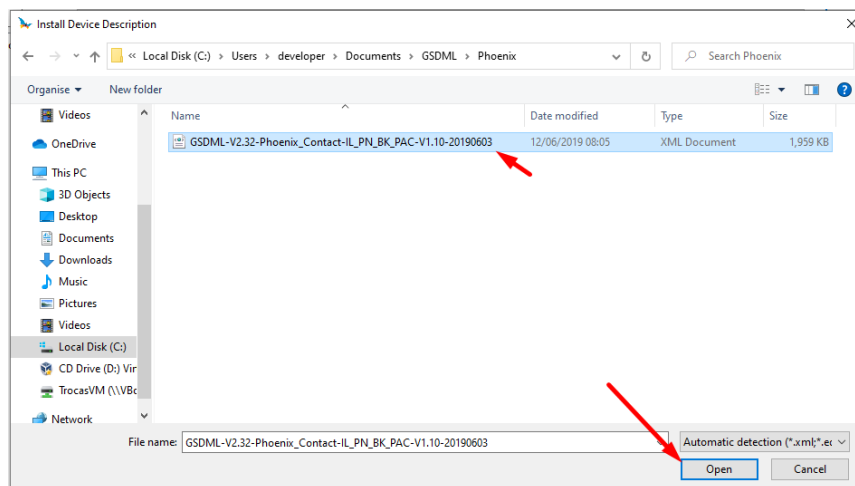


Figure 4-11. Select the GSDML file

Wait until the installation of GSDML file is complete. In the following figure, the message box informs all devices installed in the repository. Some GSDML files may contain descriptions for several devices. Finish the process by pressing the "Close" button.

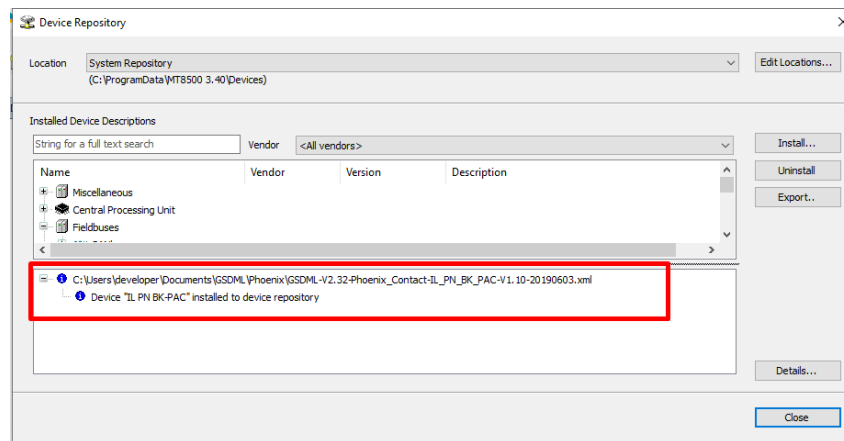


Figure 4-12. GSDML installation complete

Insert PROFINET Devices (IODs) below the Profinet Controller (IOC)

This step must be executed for all the IODs controlled by the IOC in the PROFINET network.

ATTENTION:

It is strongly recommended to use PROFINET-certified switches supporting LLDP in the network. Topology diagnostics are not available for connections between a non-PROFINET certified switch and PROFINET devices.

Insert Instance of a PROFINET Device

Right-click on "PN_Controller" and select "Add Device".

Navigate below "Profinet IO Slave" until you find the desired device. If you have not installed the GSDML file of this device, see the section **Install Missing GSDML Files in Device Repository**.

The following figure shows an example for selecting the device IL PN BK-PAC from Phoenix Contact®.

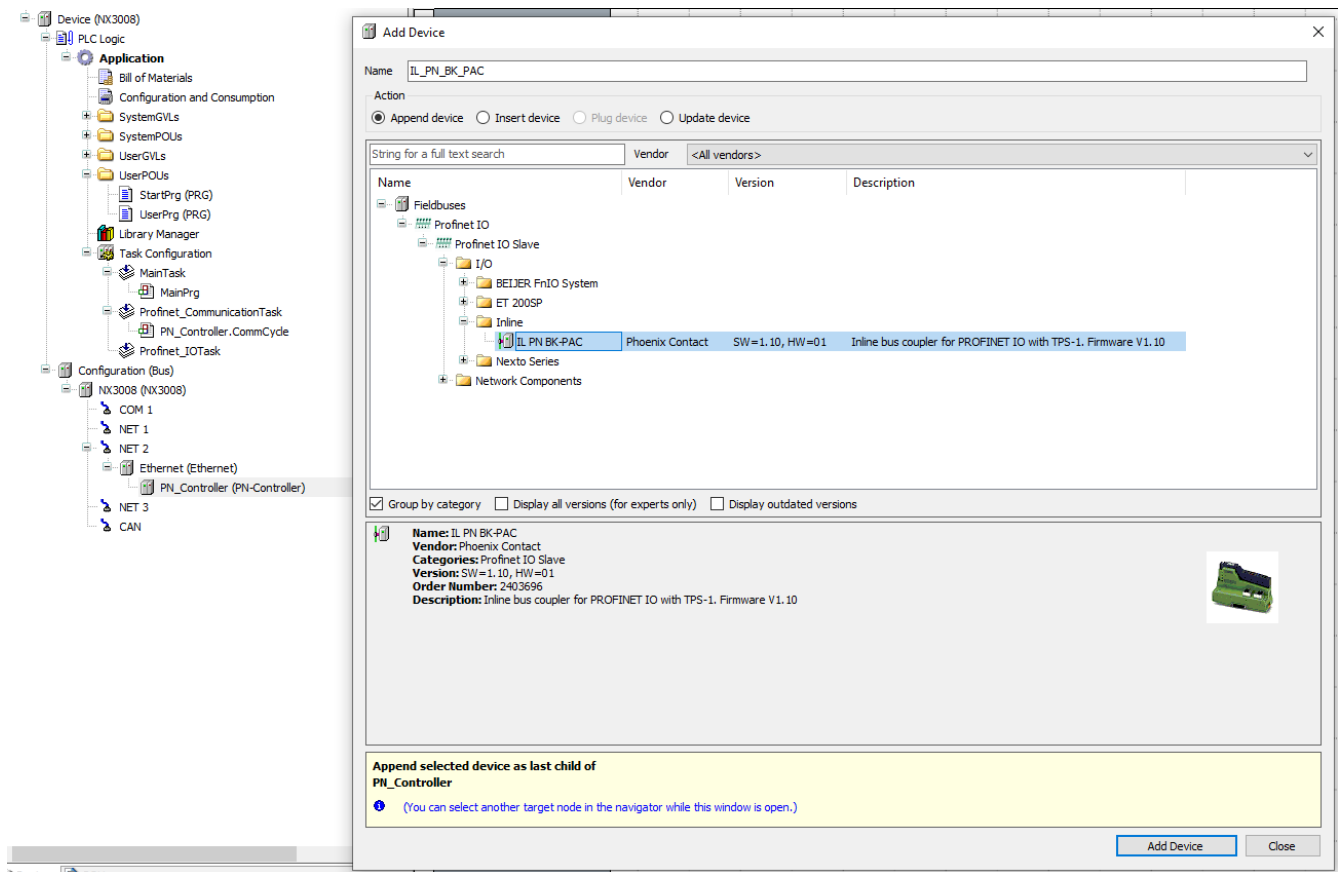


Figure 4-13. Inserting instance of a PROFINET device

After selecting the desired device, press the button "Add Device". As a result, the device is inserted below "PN_Controller" in the device tree.

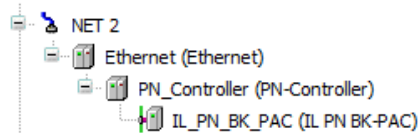


Figure 4-14. PROFINET device in device tree

Configure General Parameters of a PROFINET Device

Double-click on the device in the device tree and select the tab "General". The following figure shows an example for IL PN BK-PAC from Phoenix Contact®.

Configuration (Bus) IL_PN_BK_PAC x

General

Port data

IOxS

PNIO I/O Mapping

PNIO IEC Objects

Status

Information

Station name: IL-PN-BK-PAC

Station status:

IP Parameter

IP address: 192 . 168 . 201 . 2

Subnet mask: 255 . 255 . 255 . 0

Default gateway: 0 . 0 . 0 . 0

Communication

Send clock (ms): 1

Data hold time (ms): 12

Reduction ratio: 4

VLAN ID: 0

Phase: -

RT class: RT Class 1

Options

Fast Startup

Shared device

Settings

Set All Default Values Read All Values Write All Values

Parameters	Value	Data Type	Allowed Values	Description
Profinet				
Alarm Generation	on	Unsigned8		

Figure 4-15. General parameters of a PROFINET device

The parameters inside the red box are common for any model of IOD. Parameters below the red box are specific for each IOD (in the previous figure, they are specific for IL PN BK-PAC).

This section only discusses the common parameters inside the red box. Parameters below the red box are discussed in the section **Configure Additional Parameters of IODs**.

- Station name:** this is a unique name for the IOD. The following rules apply for creating a valid name:
 - The name is formed by one or more labels separated by ".";
 - The maximum size of each label is 63 characters;
 - The maximum size is 80 characters for the entire name (including labels and the "." separators);
 - The following characters may be used in labels:
 - "a ... z" (capital characters "A ... Z" can be used, but they will be internally converted to lowercase characters "a ... z");
 - "0 ... 9";
 - "-".
 - The character "-" cannot start or end a label;
 - The first label must start with "a ... z";
 - The first label must not have the form "port-xyz" or "port-xyz-abcde" with a, b, c, d, e, x, y, z = "0 ... 9";
 - Names must not have the form a.b.c.d with a, b, c, d = "0 ... 999".
 - Example of valid station name: "device-5.machine-2.plant-3.vendor"
- IP address, Subnet mask, and Default gateway:** these parameters are used for addressing the device and defining its subnetwork. The following rules apply:

- The device must be in the same subnetwork of the controller;
- A gateway may be necessary for accessing the device from another subnetwork. For instance, this type of access may be used by an engineering tool. If a gateway to another subnetwork is not necessary or does not exist, its address can be 0.0.0.0.

3. Send clock (ms), Reduction rate, Phase:

- The meaning of these parameters is explained in section **Cyclic Data Exchange**.
- Allowed values for "Send clock (ms)": 1, 2 or 4 ms.
- Allowed values for "Reduction rate": 1, 2, 4, 8, 16, 32, 64, 128, 256 and 512.
- Allowed values for "Phase": 1 ... "Reduction Rate", or the special value "-". The special value "-" makes the programming system calculate an appropriate phase number automatically, for distributing phases equally among the several devices.

Smaller values of "Send clock (ms)" and "Reduction rate" result in smaller response times for control functions. However, on the other hand, this also results in more CPU consumption. It is important to tune these parameters in a suitable manner, according to the response time requirements. Avoid using unnecessarily low values for obtaining a response time smaller than necessary. The user must know the worst-case response time required by each control loop.

- 4. Data hold time (ms):** According to section **Cyclic Data Exchange**, there are cyclic I/O data transmissions from IOC and IOD and vice-versa. Both IOC and IOD monitor the time between two consecutive received frames of I/O data. If this time exceeds the value of parameter "Data hold time (ms)", the connection between IOC and IOD is aborted, and a new connection must be established later. Some important notes about this parameter:
- If the data hold time (watchdog) expires and the connection is aborted, outputs of IOD typically change to a "safe state".
 - The minimum value allowed for "Watchdog (ms)" is 3 times the I/O update cycle ($3 * \text{"Send clock (ms)} * \text{"Reduction rate"}$).
 - In some situations a greater value must be used for this parameter. For instance, when redundant media like MRP ring is used, the failover time can be pretty big (e.g.: several tenths of a second). "Data hold time (ms)" must be greater than the worst-case failover time.
- 5. VLAN ID:** number between 0 and 4095 for VLAN type 802.1Q. For newer devices compliant with PROFINET specification V2.3 or newer, only "0" is permitted.
- 6. RT class:** this parameter will be fixed at "RT class 1" for all the PROFINET controllers supported by ALTUS.

Disable Instances of Not Controlled PROFINET Devices

This action is rarely necessary. However, it may be necessary when there are two or more controllers in the same network.

Normally all PROFINET devices inserted below "PN_Controller" are really controlled by this controller. However, there are situations where it is necessary to disable a PROFINET device, because this device must not be controlled by the controller.

Consider the example of the PROFINET network in the following figure that contains two controllers.

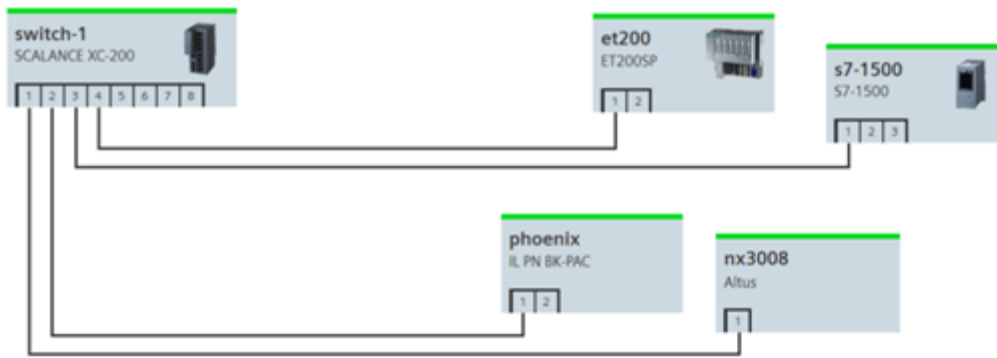


Figure 4-16. Example of PROFINET network with two controllers sharing connections to switch-1

This network has two controllers: s7-1500 and nx3008. Controller s7-1500 must control device et200, and controller nx3008 must control device phoenix. In addition, there is another device: switch-1.

Note that switch-1 must be controlled either by s7-1500 or by nx3008, but not by both controllers simultaneously. In this example, assume that switch-1 must be controlled by s7-1500.

Nevertheless, an instance of switch-1 must also be inserted in the project of nx3008, for configuring the topology connections between switch-1, phoenix, and nx3008 (see section **Configure Topology**). In this case, you must insert switch-1 in the project using the methodology described in the previous sections. However, after this, you must disable switch-1. In doing so, controller nx3008 will not try to control switch-1.

For disabling switch-1, right-click on SWITCH_1 in the device tree, and select "Disable Device". After this, the device tree will look like the following figure.

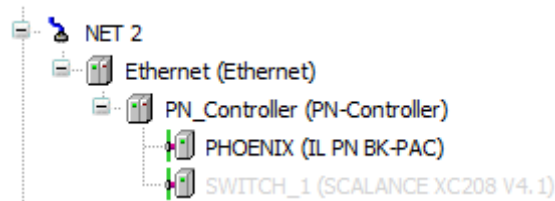


Figure 4-17. Example of device tree with disabled device (switch-1)

I/O, diagnostics and alarms of switch-1, in this example, are not received by controller nx3008, but are received by controller s7-1500.

Insert I/O Modules below IODs

This step must be executed for all the IODs which are modular. In this case, it is necessary to inform the I/O modules installed in each slot of the modular IOD.

First, we will describe an example for the device IL-PN-BK-PAC from Phoenix Contact®.

Right after adding IL-PN-BK-PAC below PN_Controller, it looks like the following figure.

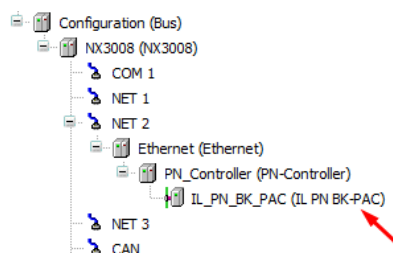


Figure 4-18. IL-PN-BK-PAC added in device tree

For adding an I/O module in the first slot, right-click over IL-PN-BK-PAC, and select "Add Device". After this, browse until you find the desired I/O module.

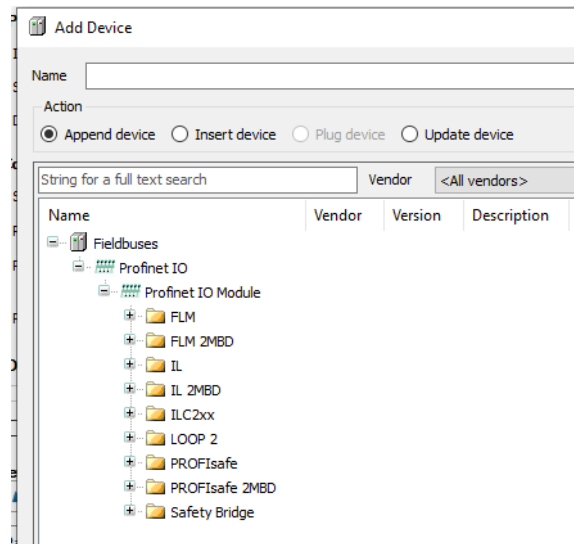


Figure 4-19. Adding an I/O module in next slot of IL-PN-BK-PAC

This process must be repeated for inserting additional I/O modules in the next slots of IL-PN-BK-PAC.

For removing an I/O module from IL-PN-BK-PAC, right-click on it and select "Delete".

Other devices may use a different approach. This happens, for instance, with device IM 155-6 PN/2 HF V4.2 from Siemens®.

Right after adding IM 155-6 PN/2 HF V4.2 below PN_Controller, it looks like the following figure.

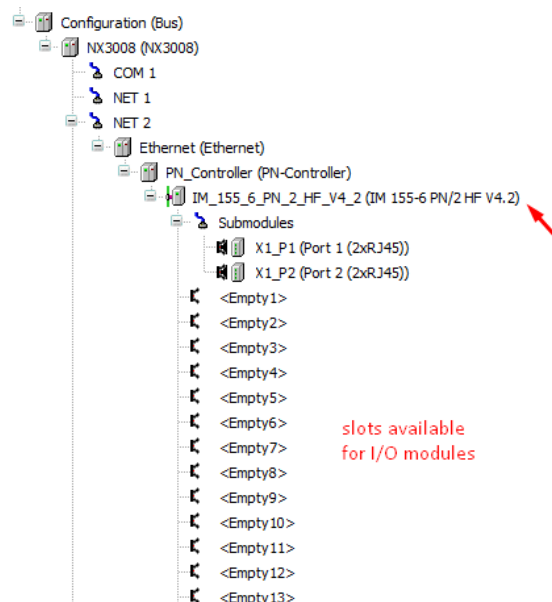


Figure 4-20. IM 155-6 PN/2 HF V4.2 added in device tree

After this, right-click over an empty slot and select "Plug Device". After this, browse until you find the desired I/O module.

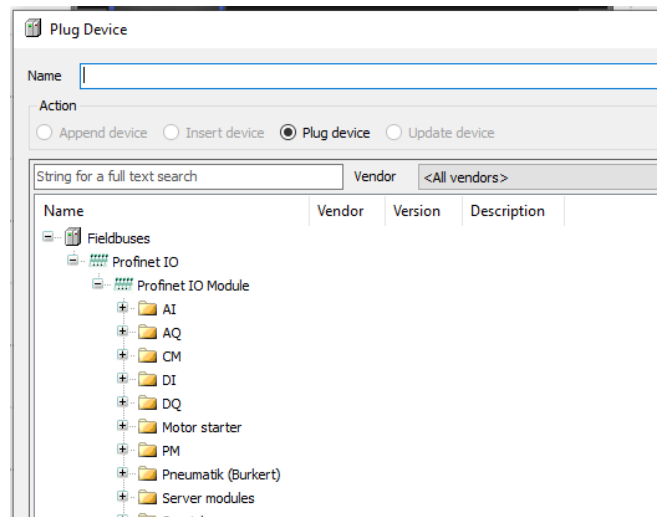


Figure 4-21. Adding an I/O module in a specific slot of IM 155-6 PN/2 HF V4.2

Repeat this procedure for installing all the desired I/O modules for IM 155-6 PN/2 HF V4.2.

For removing an I/O module from IM 155-6 PN/2 HF V4.2, right-click on it and select "Delete".

Configure Additional Parameters of IODs

Besides the common parameters previously described in the section **Configure General Parameters of a PROFINET Device**, some IODs may have additional parameters that need to be configured.

An example of the remote head "IM 155-6 PN/2 HF V4.2" from Siemens® is shown in the following figure. The red box encompasses the additional parameters, which are specific for this model of remote head. Consult the manual of the manufacturer for getting a description of these parameters.

Configuration (Bus) IM_155_6_PN_2_HF_V4_2 x

General Station name: ET200SP

Station status: []

IP Parameter

IP address: 192 . 168 . 201 . 3

Subnet mask: 255 . 255 . 255 . 0

Default gateway: 0 . 0 . 0 . 0

Communication

Send clock (ms): 1 [v] Data hold time (ms): 12 [v]

Reduction ratio: 4 [v] VLAN ID: 0 [v]

Phase: - [v]

RT class: RT Class 1 [v]

Options

Fast Startup

Shared device

Settings

[Set All Default Values] [Read All Values] [Write All Values]

Parameters	Value	Data Type	Allowed Values	Description
General head parameters				
Diagnostics: Undervoltage	0	Bit		
Configuration control				
Configuration control	0	Bit		Configuration control provides you with the c

Figure 4-22. Additional parameters of IODs

Configure Parameters of I/O Modules below IODs

For modular IODs, it may be necessary to configure the parameters of I/O modules installed in slots. The following figure shows an example of an I/O module DI 8x24VDC HF V2.0 (8 digital inputs) inserted below an IOD IM 155-6 PN/2 HF V4.2 from Siemens®. Description of these parameters can be obtained in the datasheet or manual of the I/O module provided by its manufacturer.

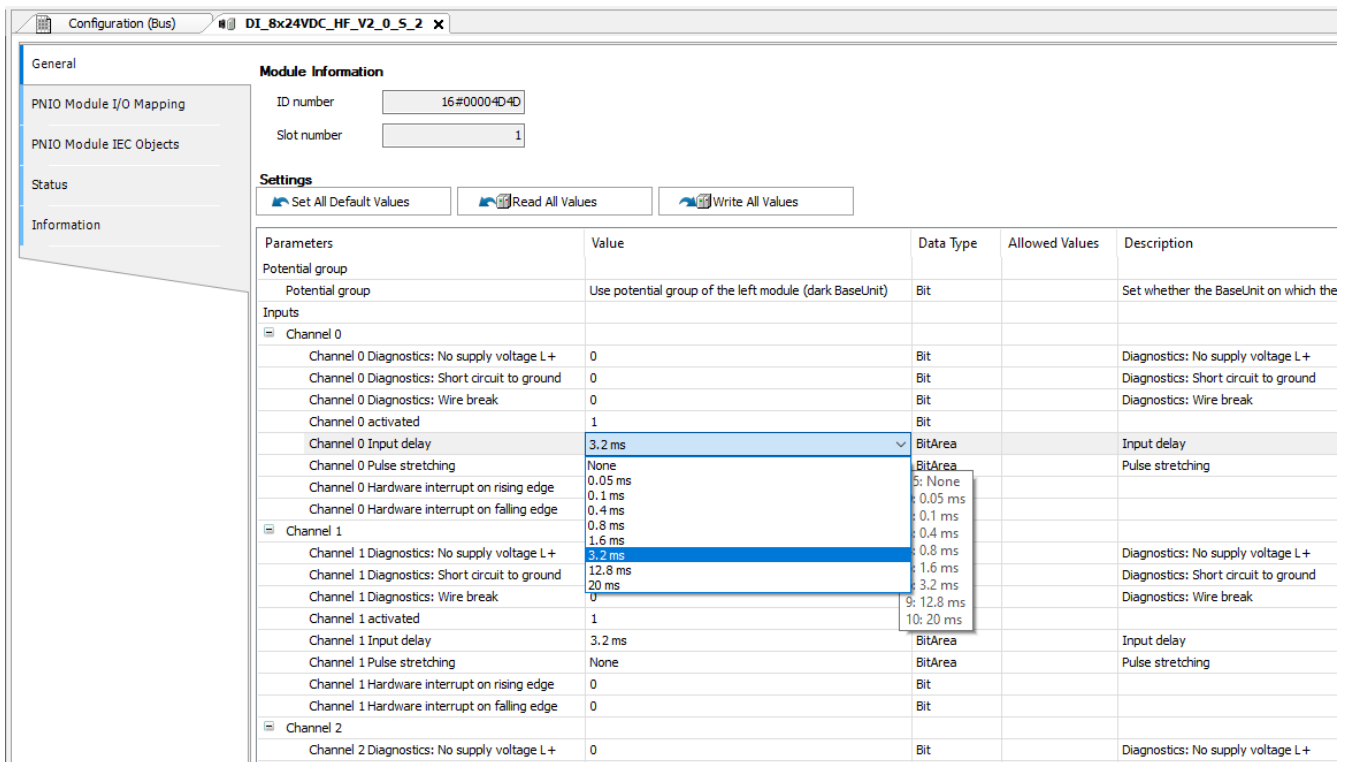


Figure 4-23. Parameters of I/O modules

Renaming Device Tree Objects of IODs and I/O Modules

This step is optional but is recommended for providing better project documentation.

Names of device tree objects may be referenced in the project (for instance, for calling diagnostic functions). Using meaningful names is better than using the default names created automatically by the programming system.

Right after inserting IODs and I/O modules, the device tree objects receive automatic names like IL_PN_BK_PAC, IL_PN_BK_PAC_1, IB_IL_24_DI_8_HD_ECO, IB_IL_24_DI_8_HD_ECO_1, etc. An example is shown in the following figure.

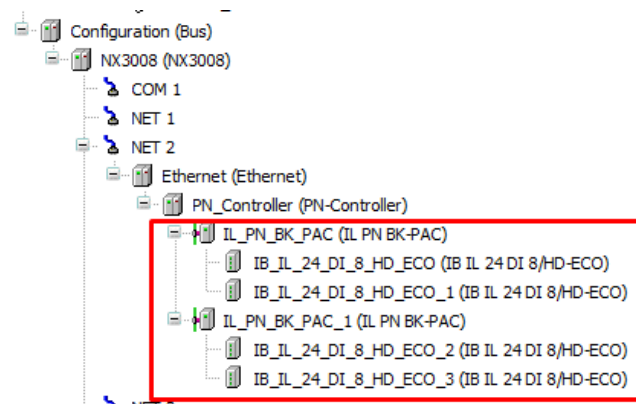


Figure 4-24. Initial names of device tree objects

For renaming a device tree object, right-click on it and select "Properties". The following figure shows an example for renaming the object IL_PN_BK_PAC to BOILER_01A.

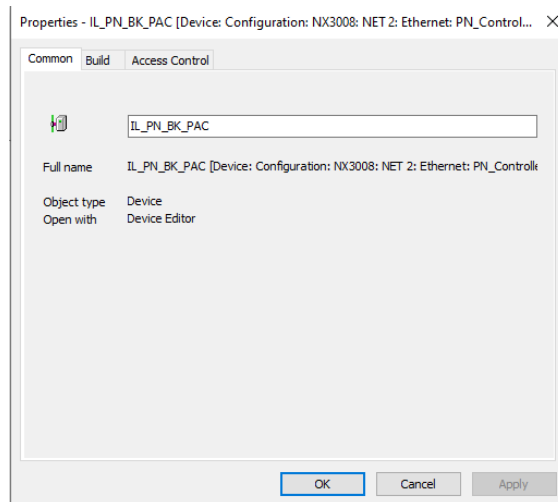


Figure 4-25. Renaming a device tree object

In the text box, change the name IL_PN_BK_PAC to BOILER_01A and click on the button OK. A message box will appear. Clicking on "Yes" allows to rename other references of IL_PN_BK_PAC that may already exist in the project. Clicking on "No" just renames the object in the device tree. Clicking on "Cancel" aborts renaming.

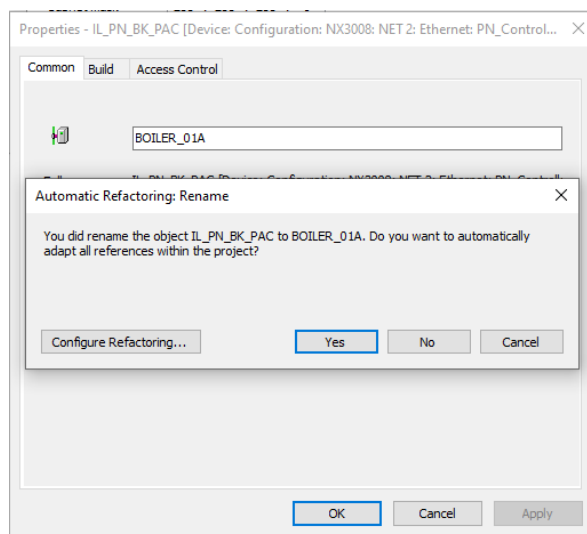


Figure 4-26. Message box for refactoring

In the end, the device tree will look like the following figure.

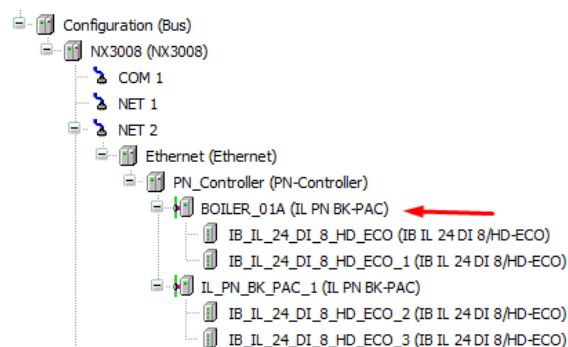


Figure 4-27. Device tree object renamed

Naming Variables of IODs and I/O Modules

This step is optional but is recommended for providing better project documentation.

In the IOD and I/O modules, %I and %Q variables are allocated for inputs, outputs, IOPS, and IOCS. You can name these variables for referencing them in the user application.

Double-click on the I/O module or in the IOD, and select the tab "PNIO Module I/O Mapping". Expand the column "Variable" and edit names in this column. The following figure shows an example for the 4 analog input module AI_BOILER_01A (IB IL AI 4/I/4-20-ECO) below the IOD BOILER_01A (IL PN BK-PAC).

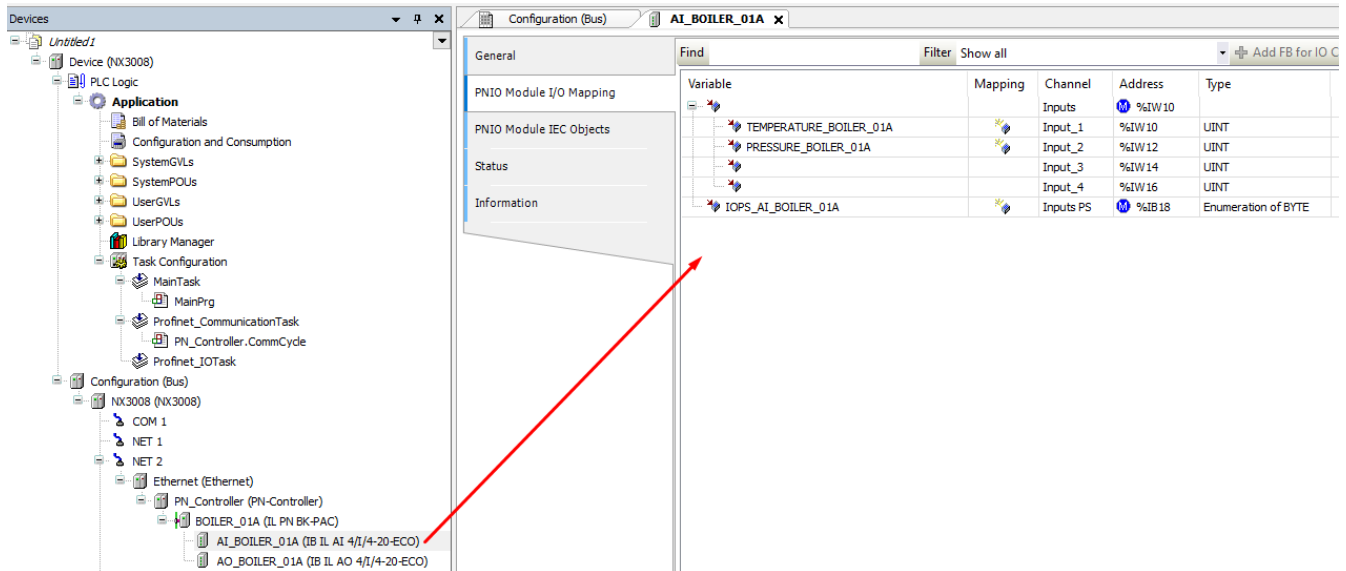


Figure 4-28. Example of I/O tags for an I/O module

The first and second analog inputs were named TEMPERATURE_01A and PRESSURE_01A. In addition, the IOPS of the module was named IOPS_AI_BOILER_01A.

Configure Topology

If a network contains only PROFINET-certified devices that support the LLDP protocol, configuring topology enables diagnostics about missing and wrong connections. In addition, configuring topology is mandatory if you want to use MRP rings. Another benefit of configuring topology is described in the section **Device Replacement without Engineering Tool**.

The following figure shows an example of topology that will be used in this section to illustrate topology configuration. In this example, there is an MRP ring formed by three IODs: switch-1, et200, and phoenix. In this ring, switch-1 is the MRP manager, and the two other IODs are MRP clients. The controller is not part of the MRP ring.

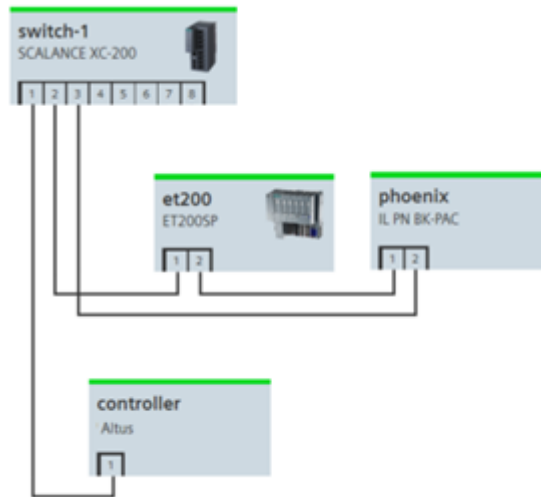


Figure 4-29. Example for topology configuration

Basic Topology Configuration

For starting the basic topology configuration double-click on "PN_Controller" and select the tab "Topology". Initially, the topology configuration will be empty, as shown in the following figure.

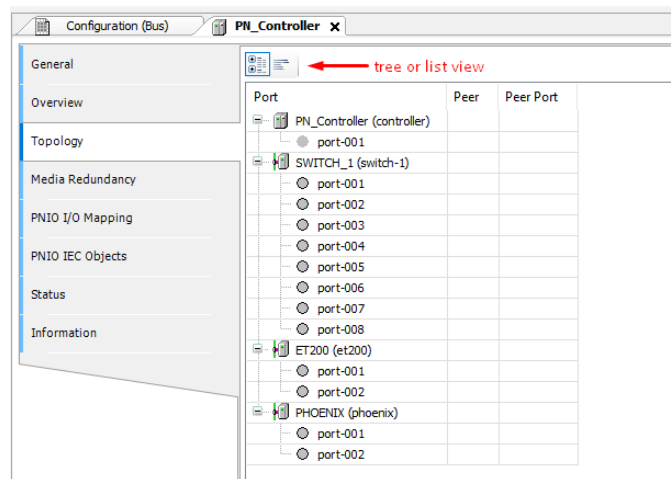


Figure 4-30. Initial screen with empty topology

It is possible to see the topology in tree view or list view by pressing the associated button. At the end of this section, after the final topology is configured, these two options will be presented.

Now, you need to insert all the connections between PROFINET devices. Let us start with the connection between the controller and switch-1. Double-click on column "Peer" of PN_Controller.port-001, and expand until selecting switch-1.port-001:

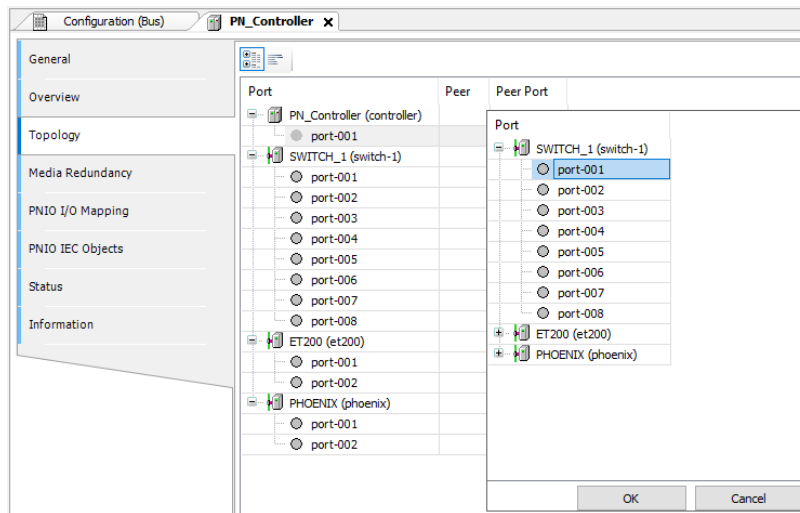


Figure 4-31. Inserting a connection

After pressing button OK, the following screen results:

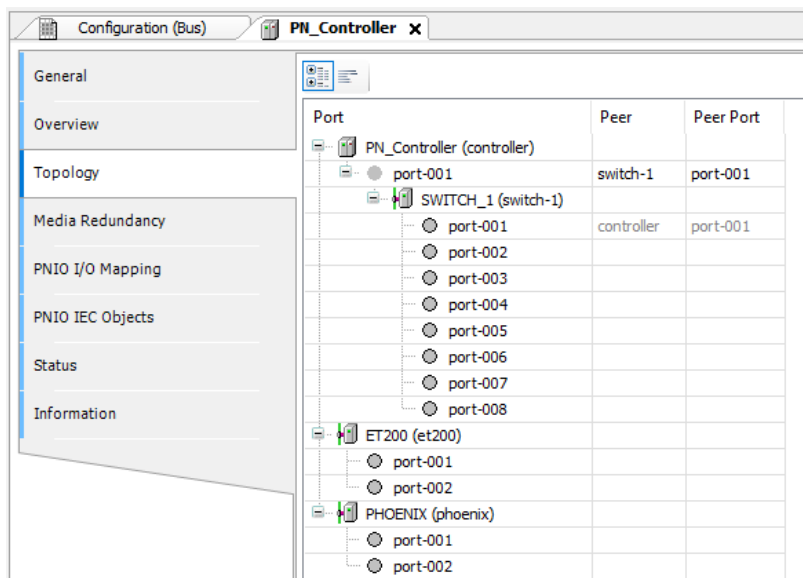


Figure 4-32. Connection inserted

Note that two entries appeared in the table: from controller.port-001 to switch-01.port-001, and vice-versa.

Now proceed to insert the remaining connections. After all the connections are inserted, the following screen results:

Port	Peer	Peer Port
PN_Controller (controller)		
port-001	switch-1	port-001
SWITCH_1 (switch-1)		
port-001	controller	port-001
port-002	et200	port-001
port-003	phoenix	port-002
port-004		
port-005		
port-006		
port-007		
port-008		
ET200 (et200)		
port-001	switch-1	port-002
port-002	phoenix	port-001
PHOENIX (phoenix)		
port-001	et200	port-002
port-002	switch-1	port-003

Figure 4-33. Screen with final topology (list view)

Port	Peer	Peer Port
PN_Controller (controller)		
port-001	switch-1	port-001
SWITCH_1 (switch-1)		
port-001	controller	port-001
port-002	et200	port-001
ET200 (et200)		
port-001	switch-1	port-002
port-002	phoenix	port-001
PHOENIX (phoenix)		
port-001	et200	port-002
port-002	switch-1	port-003
port-003	phoenix	port-002
port-004		
port-005		
port-006		
port-007		
port-008		

Figure 4-34. Screen with final topology (tree view)

Configure Media Redundancy with MRP Rings

This step is only necessary if an MRP ring exists (the previous example has an MRP ring).

Right after making the basic topology configuration shown in the previous figure, select the tab "Media Redundancy" in PN_Controller. The following screen will appear.

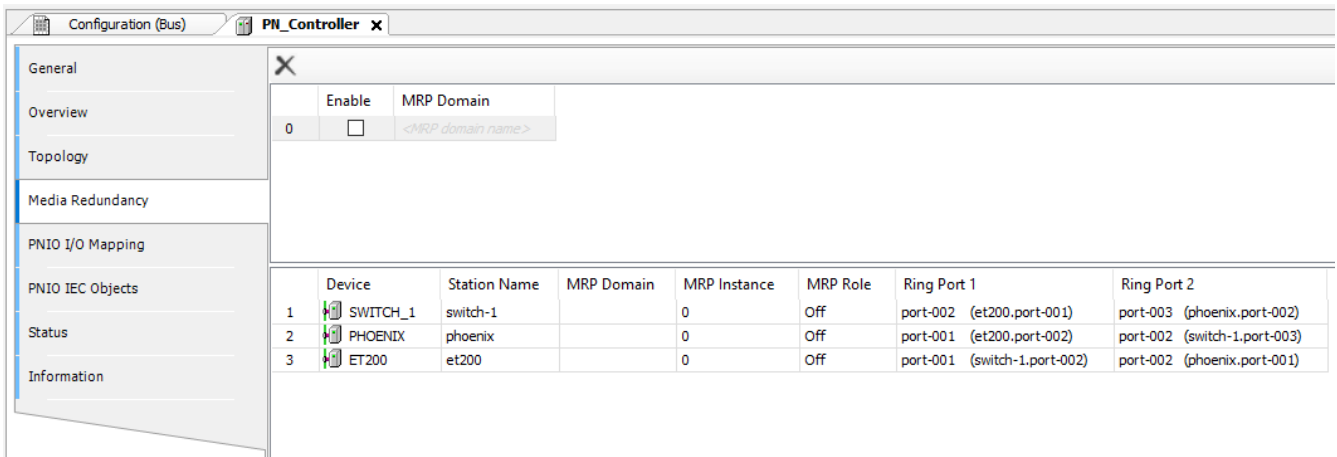


Figure 4-35. Initial state of Media Redundancy screen

Note that all devices that participate in the ring (MRP domain) are listed.

In the upper part of the screen, mark the checkbox "Enable" for the MRP domain. After this, the screen looks like the following figure.

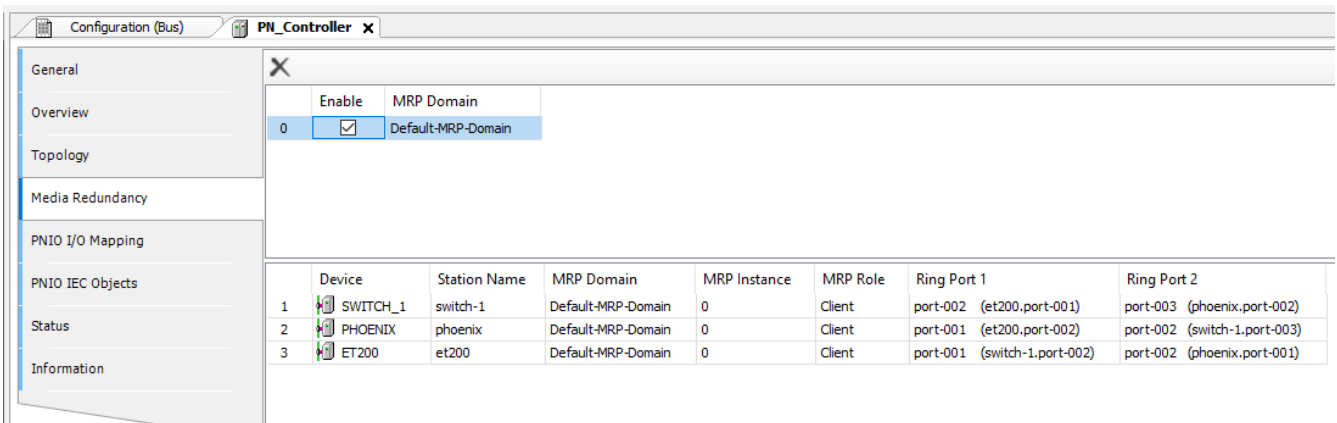


Figure 4-36. Media Redundancy screen after enabling the MRP domain

Finally, it is necessary to define the role of each device, in the column "MRP role". Only change switch-1 to "Manager", because the other devices are already configured correctly as "Client". The final screen is shown in the following figure.

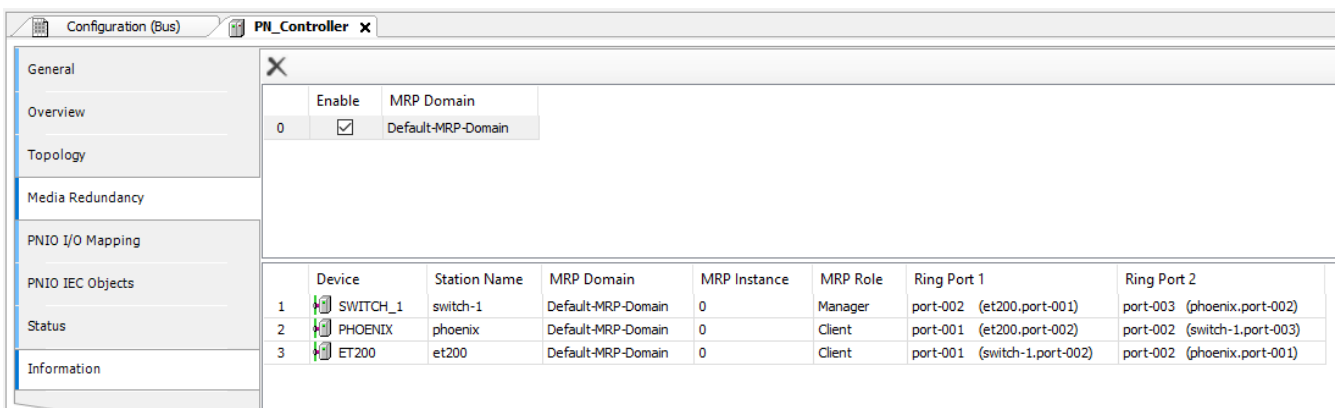


Figure 4-37. Final state of Media Redundancy screen

Configure MRP Ring with ALTUS Controller

Some ALTUS controllers like NX3008 can be installed in a MRP ring. In this case, it is necessary to use "PN-Controller Dual-port" as described in section **Insert Instance of the PROFINET Controller**. In addition, it is necessary to configure the dual-ports (for instance, NET2+NET3 of NX3008) in the switch mode with loop protection mode equal to MRP, as described in section **Parameters for Dual Ports**.

Finally, it is necessary to configure topology and media redundancy, as described in sections **Basic Topology Configuration** and **Configure Media Redundancy with MRP Rings**.

The following figure shows an example of MRP ring topology containing NX3008 as controller, and devices switch-1, et200 and phoenix.



Figure 4-38. Example of MRP ring with ALTUS controller

The topology configuration for the previous figure looks like the following figure:

Port	Peer	Peer Port
PN_Controller_Dual_port (controller)		
port-001	phoenix	port-002
PHOENIX (phoenix)		
port-001	et200	port-002
ET200 (et200)		
port-001	switch-1	port-002
SWITCH_1 (switch-1)		
port-001	controller	port-002
port-002	et200	port-001
port-003		
port-004		
port-005		
port-006		
port-007		
port-008		
port-002	phoenix	port-001
port-002	controller	port-001
port-002	switch-1	port-001

Figure 4-39. Topology configuration for the previous figure

The media redundancy configuration looks like the following figure when the controller is an MRP manager.

Device	Station Name	MRP Domain	MRP Instance	MRP Role	Ring Port 1	Ring Port 2
1 PN_Controller_Dual_port	controller	Default-MRP-Domain	0	Manager	port-001 (phoenix.port-002)	port-002 (switch-1.port-001)
2 SWITCH_1	switch-1	Default-MRP-Domain	0	Client	port-001 (controller.port-002)	port-002 (et200.port-001)
3 ET200	et200	Default-MRP-Domain	0	Client	port-001 (switch-1.port-002)	port-002 (phoenix.port-001)
4 PHOENIX	phoenix	Default-MRP-Domain	0	Client	port-001 (et200.port-002)	port-002 (controller.port-001)

Figure 4-40. Media redundancy configuration for the controller as MRP manager

The media redundancy configuration looks like the following figure when the controller is an MRP client (in this case, switch-1 was configured as the MRP manager).

Device	Station Name	MRP Domain	MRP Instance	MRP Role	Ring Port 1	Ring Port 2
1 PN_Controller_Dual_port	controller	Default-MRP-Domain	0	Client	port-001 (phoenix.port-002)	port-002 (switch-1.port-001)
2 SWITCH_1	switch-1	Default-MRP-Domain	0	Manager	port-001 (controller.port-002)	port-002 (et200.port-001)
3 ET200	et200	Default-MRP-Domain	0	Client	port-001 (switch-1.port-002)	port-002 (phoenix.port-001)
4 PHOENIX	phoenix	Default-MRP-Domain	0	Client	port-001 (et200.port-002)	port-002 (controller.port-001)

Figure 4-41. Media redundancy configuration for the controller as MRP client

Recommendations for MRP Rings

Adjust Parameter "Data hold time" of IODs in the Ring

In an MRP ring, there is a failover time needed to recover communication between the IOC and the IODs after a failure is detected (for instance, a broken connection). The failover time is normally lower than 200 ms for any kind of failure. However, the failover time can be bigger depending on specific features of MRP managers and clients connected in the ring.

The parameter "Data hold time" of each IOD connected in an MRP ring must be bigger than the worst-case failover time of the ring, to avoid disconnection of the IOD (see section **Configure General Parameters of a PROFINET Device**).

Open the Ring before some MRP Configuration Changes

For avoiding loops, open the ring before some configuration changes:

- Before downloading the first configuration with MRP.
- After downloading a configuration where the role of NX3008 changed from MRP manager to MRP client.

Specific Recommendation for ALTUS Controllers in an MRP Ring

If an ALTUS controller is connected to an MRP ring, some recommendations must be followed to avoid increased failover times:

- The average cycle of MainTask must not exceed 50% of MainTask interval.
- The average cycle of MainTask must not exceed 50 ms.

The average cycle of MainTask can be observed using the Mastertool programming (clicking in the **Task Configuration** object in the tree-view).

If the above recommendations cannot be followed, it may be necessary to increase the parameter "Data hold time" of each IOD connected in the MRP ring.

Online Functions for Supporting Configuration

The Mastertool programming system provides a collection of online tools for supporting PROFINET configuration. This section describes these functions.

These functions are executed by Mastertool through an ALTUS controller that supports PROFINET (see **Table 1-1. Features of Nexto controllers with PROFINET**). Before trying to execute these functions, it is necessary to download a project to the controller. This project must contain at least an instance of the PROFINET controller (see section **Insert PROFINET Controller (IOC) below a Network Interface**). After downloading this project to the controller, for executing these functions, you can log out from the controller, but you must keep active the network path to the controller.

In addition, these functions normally are executed over IODs. These IODs must also be online and connected to the same Ethernet network of the controller.

Some simpler functions, like assigning station names to IODs, can be executed by some generic tools (e.g.: Profinet Commander®, PRONETA®, etc) that only requires a computer, without needing the controller.

For describing how to use the online function in Mastertool, the following example of topology will be used. The station names of the three IODs are switch, et200, and phoenix.

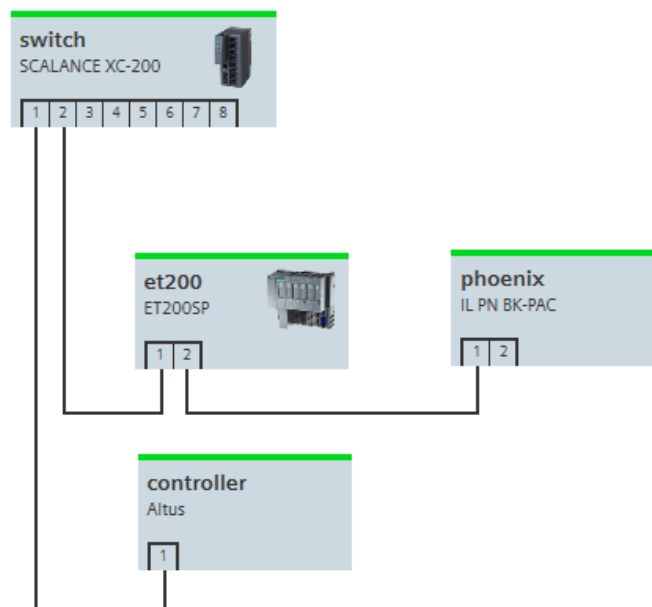


Figure 4-42. Example of a project for describing online functions

Assign Station Name and IP Address to an IOD, and Blink LEDs of an IOD

In the controller project, every IOD is designated by a station name, as described in the section **Configure General Parameters of a PROFINET Device**.

Consider the following facts about the station name of an IOD:

- If an IOD does not have a correct station name assigned to it, the IOC will not be able to connect to it. There is an exception for this situation, described in the section **Device Replacement without Engineering Tool**.

- A station name can be assigned to an IOD using an engineering tool, like the Mastertool programming system or some generic tool (e.g.: Profinet Commander®, PRONETA®, etc).
- A station name must be retentive so that it is not lost in case of power down.

This section describes how the Mastertool programming system can be used for assigning a station name and IP address to an IOD, and for blinking LEDs of an IOD for identifying it.

Before assigning a station name to an IOD, it is recommended to blink their LEDs, to be sure that you are naming the correct IOD (this could happen in a network where several IODs are still not named).

Consider the project topology of **Figure 4-42. Example of a project for describing online functions**. In this topology, assume that IODs et200 and phoenix are still not named and do not have IP addresses, and that IODs switch-1 is already named and has an IP addresses.

Initially, double-click on "PN_Controller" in the device tree, select the tab "Topology", and then press the refresh button for scanning the topology.

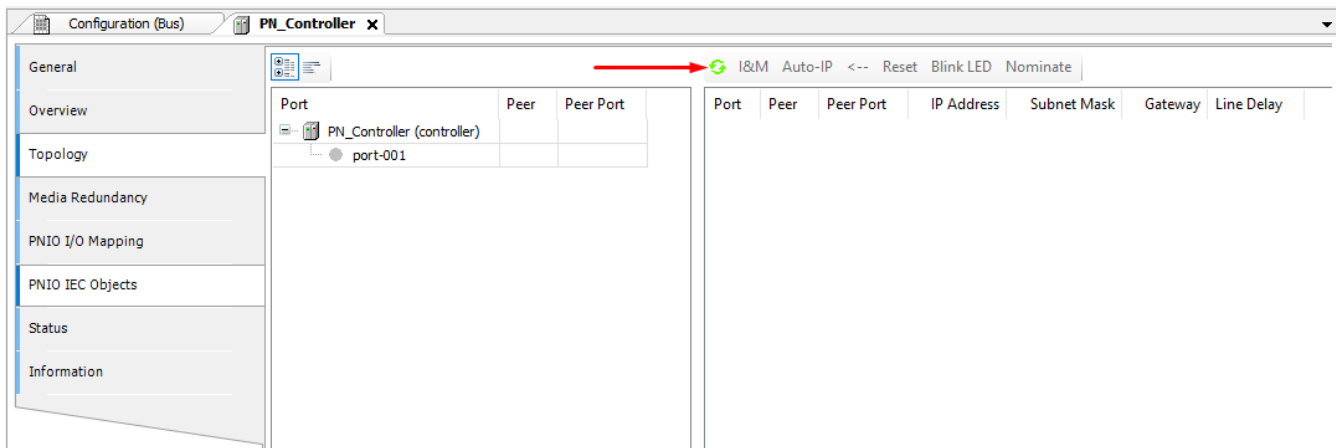


Figure 4-43. Scanning topology

After this, the following screen arises:

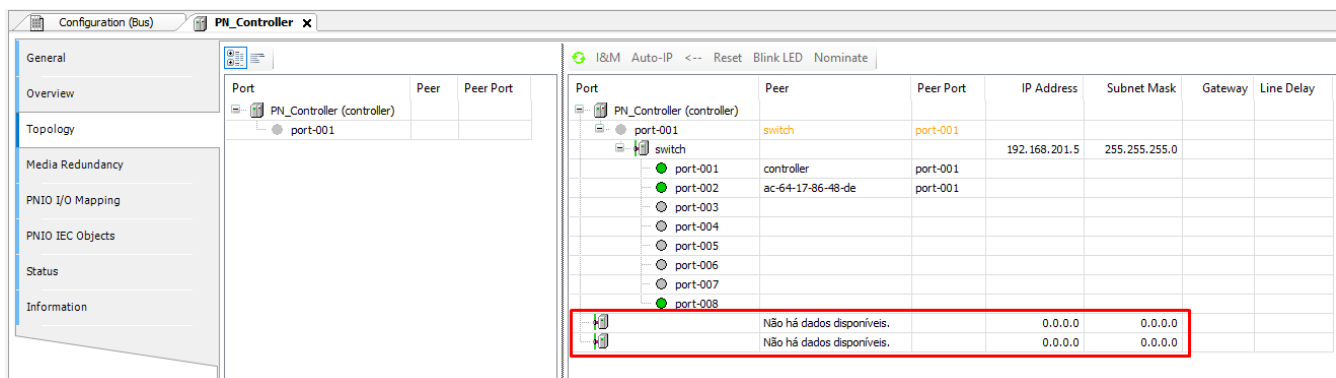


Figure 4-44. Scanned topology with two not named devices

Inside the red box, you see the two devices not named (et200 and phoenix). However, it is not possible to know which one is et200 and which one is phoenix. The solution for this dilemma is to use the "Blink LED" command and observe which device will blink the LEDs.

The following screen shows how to use the blink command. Click on the row of one of the not named devices, and then click on the "Blink LED" command. You will see some LEDs of the corresponding device blinking, so now you identified which is this device (in this case, it was the phoenix device). Clicking again on the "Blink LED" command, the LEDs stop blinking. Note that the LEDs that will blink in an IOD depend on this specific IOD.

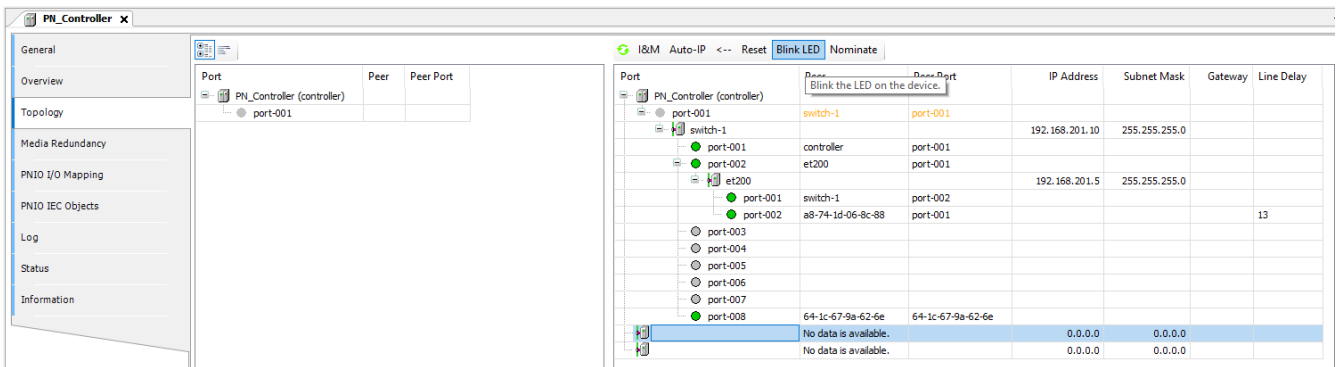


Figure 4-45. Scanning topology for naming and blinking devices

Now that you know that the device in the row selected in previous figure is phoenix, you can name it "phoenix" and set an IP address for it, according to the values configured in the project. Keep the row of this device selected, type the name "phoenix" in the port column, type the IP address and subnet mask, click outside the last edited column, and finally click on the "Nominate" command.

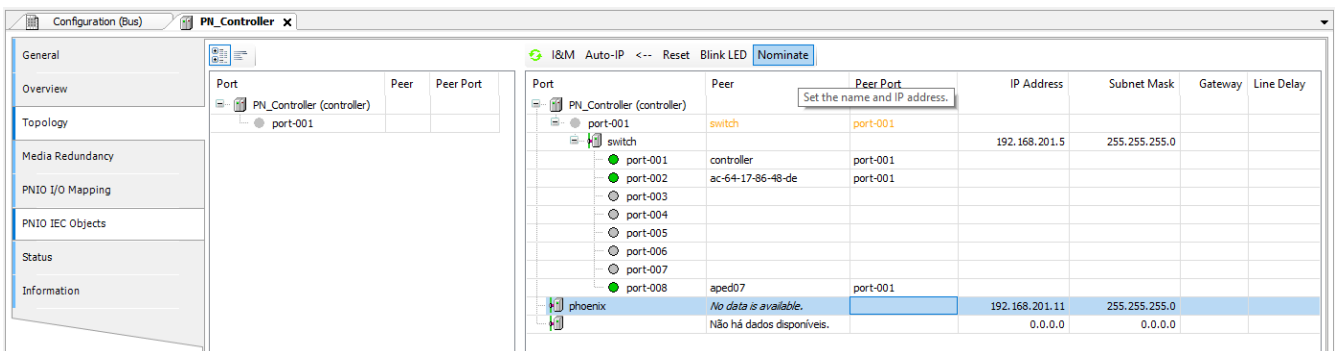


Figure 4-46. Setting station name and IP address of device

After this, the following screen is shown:

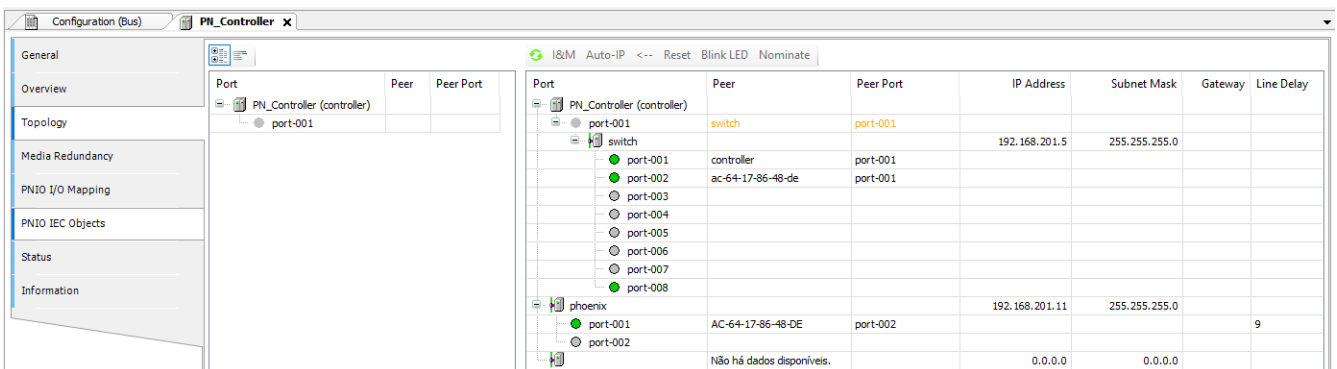


Figure 4-47. Name and IP address of device adjusted

Note that now the device phoenix has a name and IP and subnet addresses and that the screen also shows the connections between phoenix and the neighbor devices in the columns "Peer" and "Peer Port" (the neighbor at port-001 is et200 that is still not named, so the MAC address of et200 is shown in column "Peer").

Now you can repeat this procedure (blinking and naming) for device et200. After everything is complete, you will see a screen like the following.

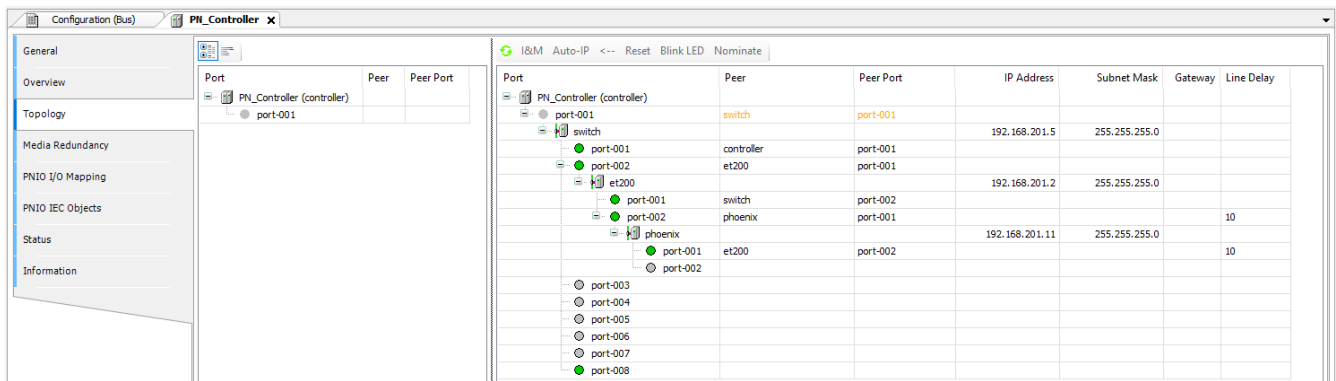


Figure 4-48. Final topology screen

ATTENTION:

When naming a device, it is not essential to set the IP and subnet addresses, because the controller will assign these addresses later if they are different from the project. However, there are some advantages when setting the IP and subnet addresses together with the station name. Without these addresses, some topology connections (columns "Peer" and "Peer Port") cannot be updated. In one of the following subsections, you will see that it is possible to import topology connections to the project.

Scan for Devices

This function reduces the configuration effort in some situations. It enables the addition of one or more devices and their I/O modules if these devices are online in the same Ethernet network of the controller. Therefore, some of the configuration tasks described in the following section will be executed automatically:

- **Insert PROFINET Devices (IODs) below the Profinet Controller (IOC);**

There are some important notes before using this function for copying IODs and their I/O modules to the project:

- Remember to install the GSDML file of these IODs (see section **Install Missing GSDML Files in Device Repository**);
- If possible, give station names and assign IP addresses for these IODs. In doing so, these station names and IP addresses will also be imported into the project. See section **Assign Station Name and IP Address to an IOD, and Blink LEDs of an IOD**.

Consider an example for using the function "Scan for Devices", based on the topology shown in **Figure 4-42. Example of a project for describing online functions**. Assume that device phoenix is not still in the project, that is, our project only has devices switch and et200. The initial state of the PROFINET device tree is shown in the following figure:

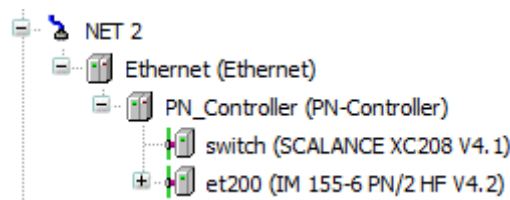


Figure 4-49. Initial state of PROFINET device tree (without phoenix)

Assume also that phoenix is connected correctly according to the topology shown in **Figure 4-42. Example of a project for describing online functions**, online, named, and with IP/subnet addresses. In this case, if you refresh the topology (tab "Topology" of "PN_Controller"), the following screen appears:

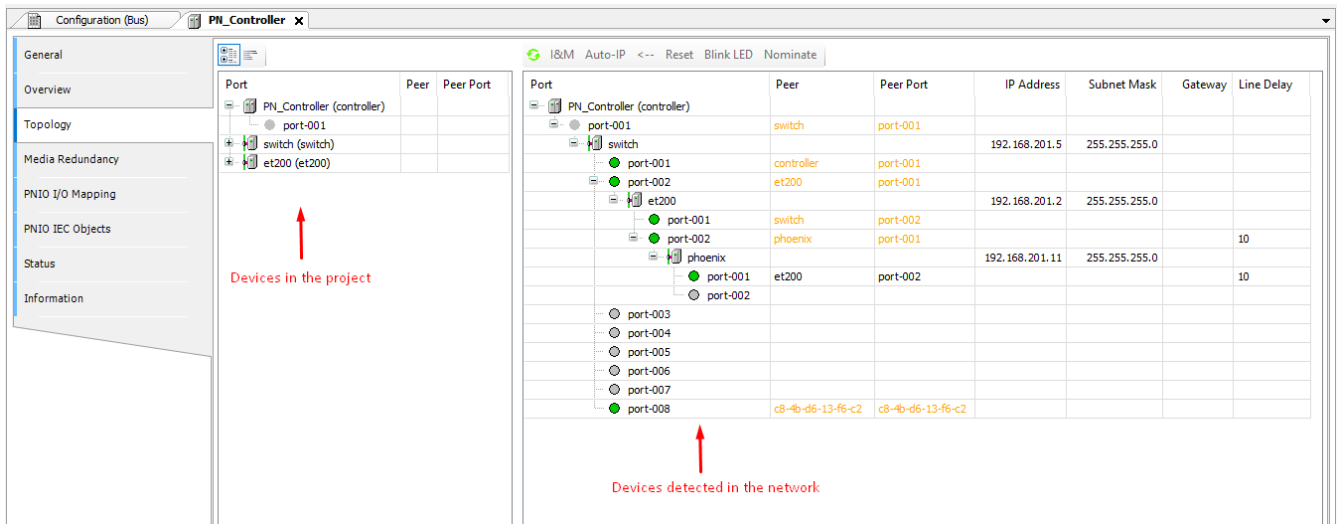


Figure 4-50. Topology scanned before inserting phoenix in the project

Now, let us execute the command "Scan for Devices". This is accomplished by right-clicking on "PN_Controller" in the device tree, and selecting the function "Scan for Devices". After this, the following pop-up screen appears:

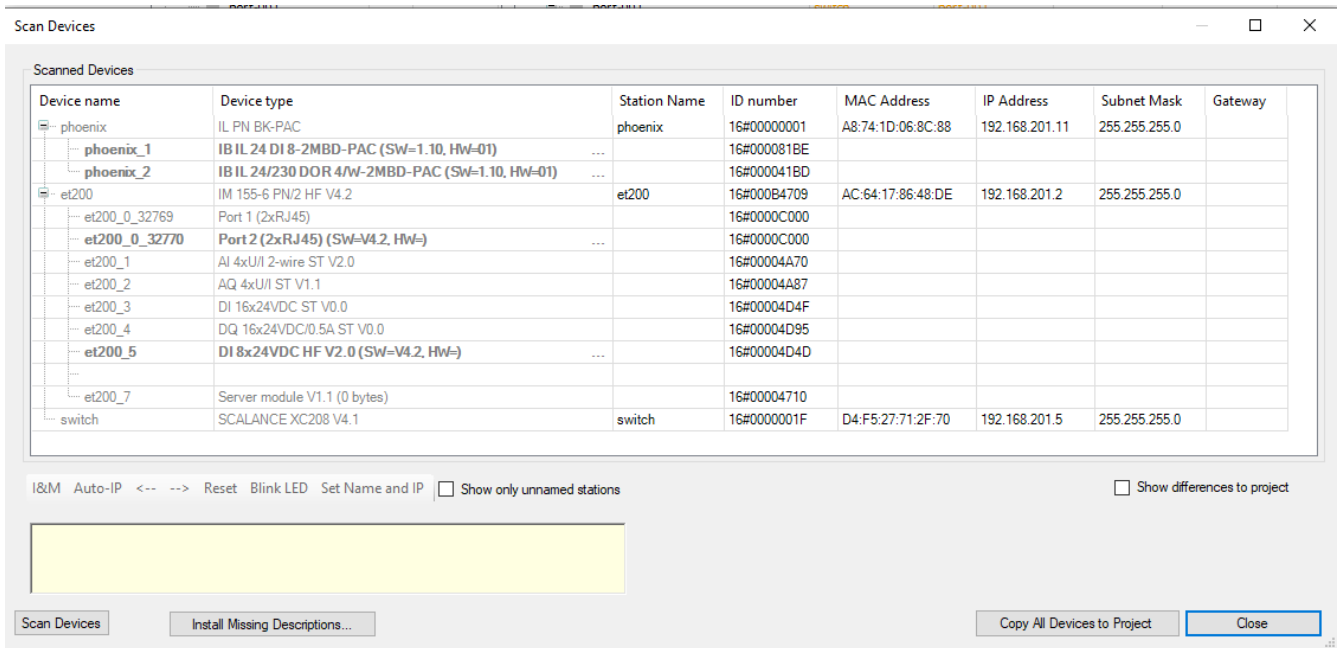


Figure 4-51. Pop-up screen with results of "Scan for Devices"

The screen shows all devices detected, including those already configured in the project. In our case, we want to copy only the device not still included in the project (phoenix). For copying only phoenix, click on the row where phoenix appears, and after this click on the button "Copy to Project", as shown in the following figure.

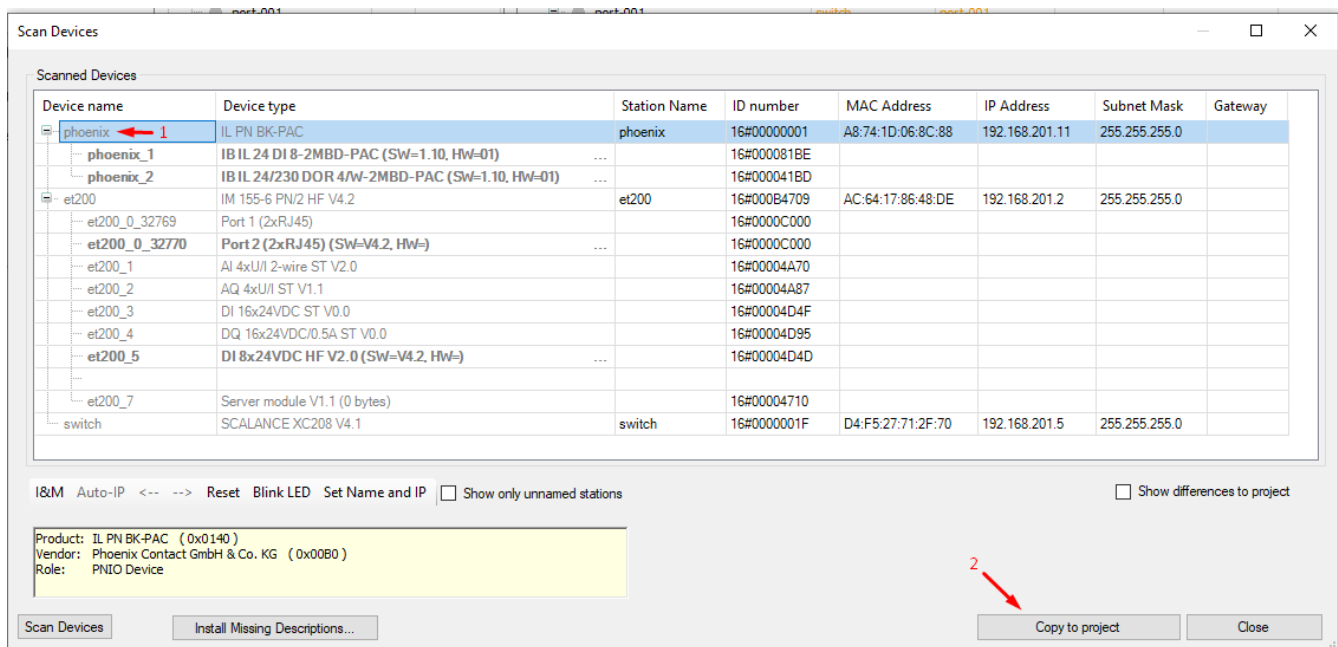


Figure 4-52. Copying device and I/O modules to the project

After pressing the button "Copy to project", the following PROFINET device tree results.

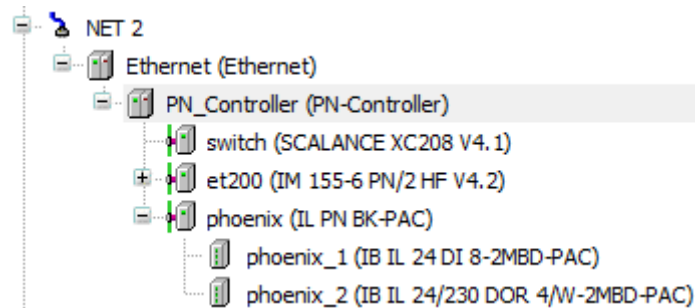


Figure 4-53. Device tree after copying phoenix and its I/O modules to the project

Now, the following configuration tasks are still necessary:

- Adjust some parameters of phoenix (see section **Configure General Parameters of a PROFINET Device**).
- Adjust additional parameters of phoenix (see section **Configure Additional Parameters of IODs**).
- Configure parameters of I/O modules below phoenix (see section **Configure Parameters of I/O Modules below IODs**).
- Optionally, rename device tree objects (see section **Renaming Device Tree Objects of IODs and I/O Modules**).
- Optionally, name variables of phoenix and their I/O modules (see section **Naming Variables of IODs and I/O Modules**).

It is also necessary to add topology connections between phoenix and other devices. For executing this task, there are two options:

- Use the manual method, described in section **Basic Topology Configuration**.
- Use an automatic method, described in the section **Import Topology Connections**.

Finally, if there are MRP rings (not the case in this example), it is necessary to update the tab "Media Redundancy" below "PN_Controller". See section **Configure Media Redundancy with MRP Rings**.

Import Topology Connections

This function will be demonstrated as a continuation of the example started in the previous section (**Scan for Devices**). After completing the example of the previous section, if you refresh the topology, the following screen appears:

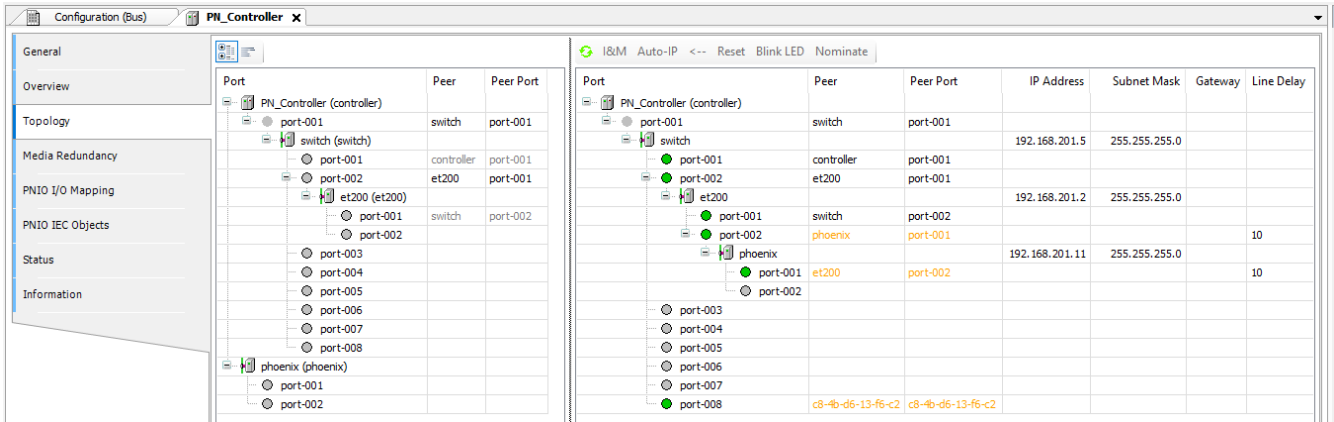


Figure 4-54. Topology scanned after inserting phoenix in the project

Now, it is possible to copy the topology connections of phoenix to the project. This can be accomplished by clicking on phoenix in the right pane, and after this clicking on the button "Import the port data" (<--).

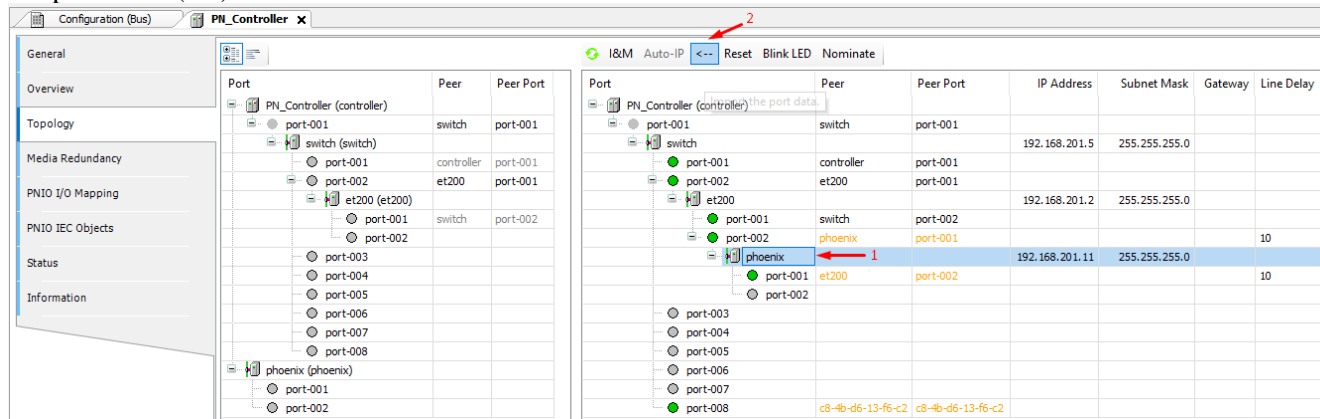


Figure 4-55. Importing topology connections of phoenix to the project

After executing this command, the updated topology screen is the following:

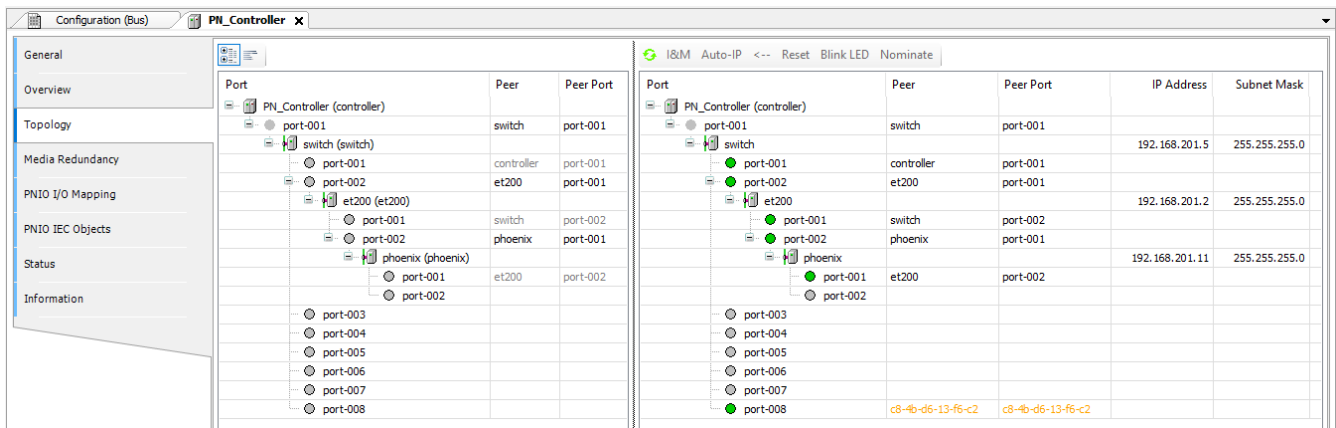


Figure 4-56. Final topology including phoenix connections

Reset to Factory Defaults

This function resets the following parameters of an IOD to factory defaults:

- Station name: empty name;
- IP address and subnet address: 0.0.0.0;
- I&M1, I&M2 and I&M3 (see chapter **I&M Data**): empty strings.

For instance, if you intend to stock an IOD for future use as a spare part, it is necessary to reset at least the station name. This enables the function described in the section **Device Replacement without Engineering Tool**.

In tab "Topology" of PN_Controller, refresh the topology. After this, select the row with the device that you want to reset. Finally, click on the command "Reset".

The following figure shows an example for resetting the IOD phoenix.

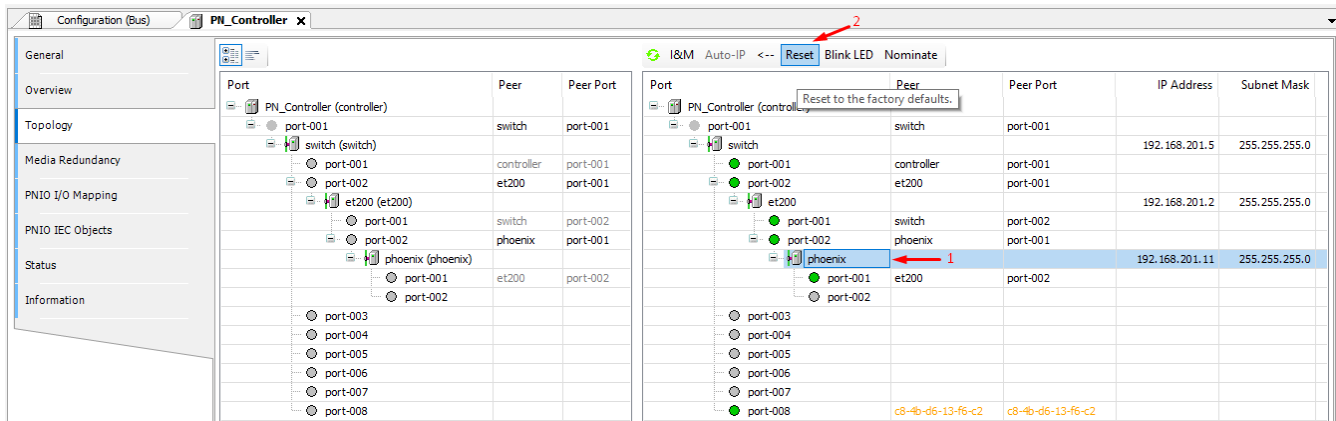


Figure 4-57. Executing reset to factory defaults

After executing the command and refreshing the topology:

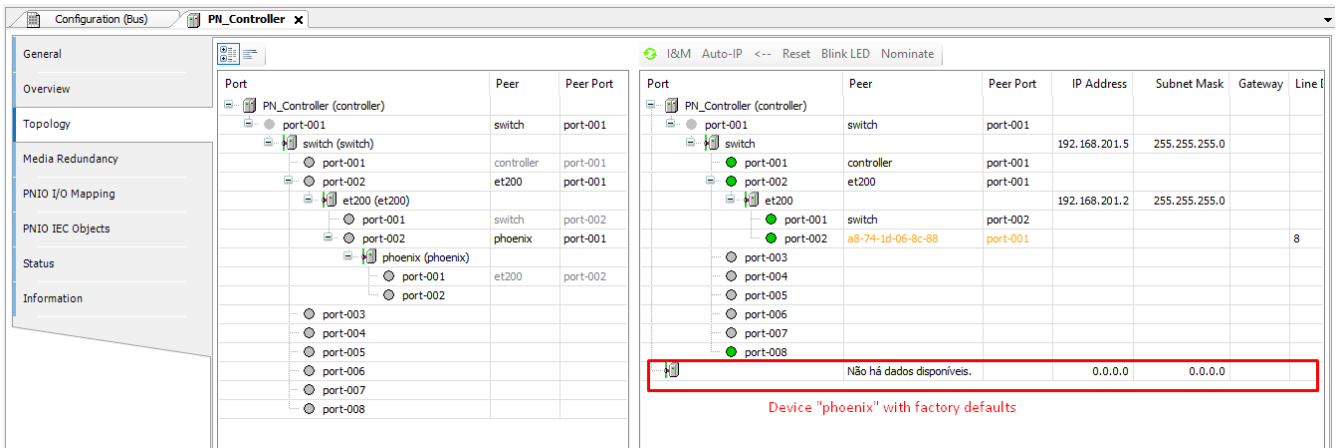


Figure 4-58. After resetting phoenix to factory defaults

Auto-IP

This command assigns an IP address and a subnet mask automatically to an IOD if this IOD already has a station name.

We could have used this command in the example of section **Assign Station Name and IP Address to an IOD, and Blink LEDs of an IOD**. In that example, we have assigned station names and IP addresses using the command "Nominate". But we could have used the command "Nominate" only for assigning station names and, later, use the command "Auto-IP" for assigning IP addresses and subnet masks.

Now let us see how to use this command and how it allocates an IP address and a subnet mask.

The following topology screen shows a situation where device phoenix is already named but does not have an IP address (all the other IODs have a name and an IP address). Select the row with phoenix and click on the command "Auto-IP".

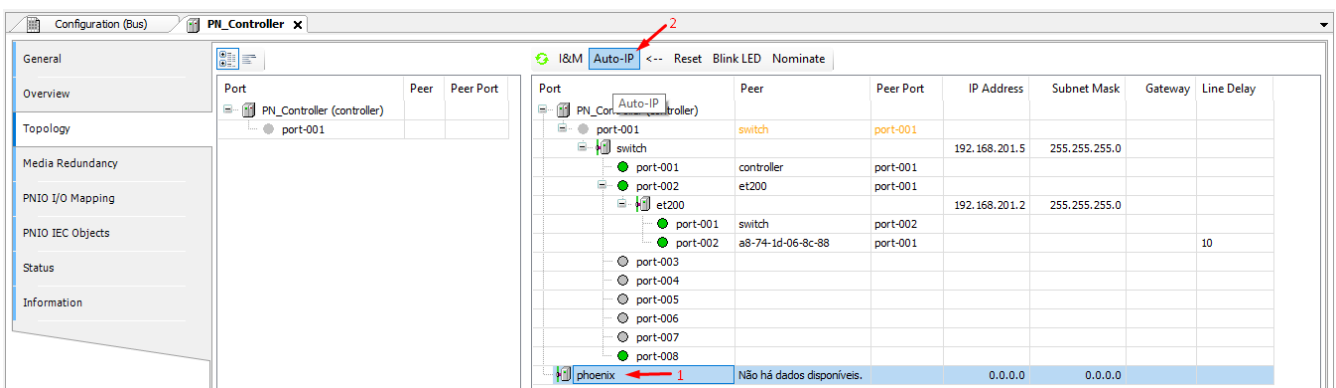


Figure 4-59. Assigning IP address to phoenix

After this, the following topology screen results:

Port	Peer	Peer Port	IP Address	Subnet Mask	Gateway	Line Delay
PN_Controller (controller)						
port-001	switch	port-001	192.168.201.5	255.255.255.0		
port-001	controller	port-001				
port-002	et200	port-001	192.168.201.2	255.255.255.0		
port-001	switch	port-002				10
port-002	phoenix	port-001	192.168.201.3	255.255.255.0		
port-001	et200	port-002				10
port-003						
port-004						
port-005						
port-006						
port-007						
port-008						

Figure 4-60. After assigning IP address to phoenix

In tab "General" of "PN_Controller", the range of addresses for IODs goes from "192.168.201.2" to "192.168.201.254", and the subnet mask is 255.255.255.0.

Default Slave IP Parameter	
First IP address	192 . 168 . 201 . 2
Last IP address	192 . 168 . 201 . 254
Subnet mask	255 . 255 . 255 . 0
Default gateway	0 . 0 . 0 . 0

Figure 4-61. Tab General of PN_Controller

Note that phoenix received the following addresses:

- IP = 192.168.201.3: because this is the smaller free address in the range (192.168.201.2 and 192.168.201.5 were already in use);
- Subnet mask = 255.255.255.0: the same subnet mask configured in tab General of PN_Controller.

I&M

This command is described in section **Reading and Writing I&M Data for a Device in Mastertool Programming System** of chapter **I&M Data**.

Special Configuration Functions

This section describes some functions that normally are not necessary but may become necessary in special situations.

Update Device after Updating GSDML File

In some situations, it may be necessary to update the GSDML file of a device that was already inserted in the project. A new version of GSDML may be released for correcting bugs or adding new features.

In this case, there are two options:

1. Remove the device and insert it again. This strategy may require a lot of work (insert the device and configure their parameters, insert I/O modules and configure their parameters). The work can be pretty big if many devices of this model exist in the project.
2. Update the device. This strategy typically requires much less work (for instance, configure new features).

The following procedure must be executed:

1. Instal the updated GSDML file. You can use the same procedure used for missing GSDML, described in the section **Install Missing GSDML Files in Device Repository**.
2. Right-click on each device described by this GSDML file, and select "Update Device".
3. Adjust those parameters affected by the modification in the GSDML file.

5. Diagnostics

This chapter describes how to retrieve PROFINET diagnostics with the controller user application using functions or function blocks available in diagnostic libraries. It also describes how to view these diagnostics using the Mastertool programming system.

ATTENTION:

Some diagnostics are shown in LEDs and LCD of devices and controllers. This manual does not describe how to interpret these types of diagnostics. For this purpose, consult the manual or datasheet of the device or controller.

PROFINET diagnostics originate from PROFINET devices (IODs) and can be destined to a PROFINET controller (IOC) or to a PROFINET supervisor (IOS).

Nexto Series controllers have generic libraries that can be used for retrieving diagnostics from devices of any manufacturer. Using these generic libraries demands some coding effort and a good understanding about PROFINET diagnostics concepts.

Introduction to PROFINET Diagnostics

This section provides an introduction about PROFINET diagnostics. It is important to read this section for retrieving diagnostics from devices using the generic diagnostic libraries.

Data Structures for a Diagnostic

Diagnosis Source

This data structure contains the following fields, which describe the source of a diagnostic inside a given device:

- API: the application process identifier (normally "0" when specific profiles are not used).
- Slot: slot number where the module is physically installed.
- Subslot: subslot of a submodule of the module. Most I/O modules have a single submodule which number is 1. Remote head modules have additional submodules for the network interfaces (example of numbers for these interfaces: 16#8001 for network port 1, 16#8002 for network port 2). Some I/O modules can also have more than one submodule.
- ChannelNumber: channel number inside a submodule (16#8000 means the entire submodule, for diagnostics that are common for the entire submodule).
- Accumulative:
 - TRUE means that source is a group of channels. In this case, ChannelNumber is the smaller number of a channel belonging to this group.
 - FALSE means that source is a single channel, or that source is the entire submodule (ChannelNumber = 16#8000).
- Direction:
 - In: the source is an input channel.
 - Out: the source is an output channel.
 - InOut: the source is a bidirectional channel.
 - ManufacturerSpecific: source is neither input nor output.

Diagnostic Severity

Severity describes how critical is the diagnostic. The following options exist:

- Normal operation.
- Advice: normal operation, but some advice is delivered to user.
- MaintenanceRequired: maintenance should be done in near future.

- MaintenanceDemanded: maintenance should be done as soon as possible.
- Fault: an immediate action is required because the channel is not working.

Diagnostic Information - Standard Format

There are two diagnosis information formats specified for PROFINET: standard format and USI format. The library function used for retrieving the diagnostic informs which format applies, so the user can decode the diagnostic properly.

This subsection describes the standard format, used by most devices.

Diagnostic information in standard format contains the following fields:

- ChannelErrorType (UINT): expresses the type of diagnostic.
 - Some codes are predefined by PROFINET standard (e.g.: 16#0001 = short circuit, 16#0006 = line break, etc), so that an IOS or programming system can display the diagnostic description without needing the GSDML file of device.
 - Other codes are reserved for manufacturers. In this case, an IOS or programming system needs the GSDML file of device for displaying the diagnostic description.
- ExtChannelErrorType (UINT): this optional field expresses a subtype for diagnosis. For instance, for a diagnostic which source is an Ethernet port of device head, ChannelErrorType = 16#8001 (remote port mismatch) means that this Ethernet port is not connected to the expected peer port. In this case, ExtChannelErrorType gives some additional information about this failure (16#8005 = no peer detected, 16#8000 = peer is a port of a wrong station, 16#8001 = peer is a wrong port of the correct station). ExtChannelErrorType has codes predefined by PROFINET standard, but also allows codes defined by manufacturer in the GSDML file.
- ExtChannelAddValue (UDINT): this optional field expresses additional information to the diagnosis. For instance, for a diagnostic where ChannelErrorType indicates "overtemperature", ExtChannelAddValue could indicate the temperature value.

Diagnosis Information - USI Format

There are two diagnosis information formats specified for PROFINET: standard format and USI format. The library function used for retrieving the diagnostic informs which format applies, so the user can decode the diagnostic properly.

USI format is normally used by gateways that integrate different fieldbusses.

This subsection describes the USI format. Diagnostic information in USI format contains the following fields:

- USI (UINT): UserStructureIdentifier - expresses the type of diagnosis.
- ManufacturerData: this is an array of bytes with manufactures specific diagnosis data. Its size is fixed in 8 bytes for diagnostics in USI format received by Nexto controllers.
- DataLength (UINT): informs the real size provided in ManufacturerData.

For decoding a diagnostic in USI format, the user must consult the user manual of the device.

Diagnosis Information - Module Differences

Module differences are managed differently from other diagnostics. Standard format and USI format are not suitable for reporting module differences.

Module differences indicate that the module found (or not found) in a slot is different from the module configured for this slot in the user application. The following module differences can be reported as diagnostics:

- Absent module: there is no module installed in a slot where a module is configured;
- Wrong module: the module installed in a slot is different and not compatible with the module configured for this slot;

- Substituted module: the module installed in the slot is different but compatible with the module configured for this slot.

Data structures for reporting this kind of diagnosis information just inform if the module is correct, absent, wrong or substituted. Diagnosis source is reported the same way as other diagnostics. Diagnosis severity is not reported, but controller normally assumes as "fault".

The methods for retrieving this type of diagnostics are different from the methods for retrieving the other types diagnostics (standard format and USI format), and different library functions are used for retrieving these diagnostics.

Methods for Retrieving Diagnostics in Nexto Controllers

The following methods for retrieving diagnostics exist in Nexto controllers.

Module Differences Reported at Connection Time

When the controller establishes connection with a device, this device reports differences between configured modules and real modules found (or not found) in the slots of device.

Module Differences Reported after Connection Time

If a device has been connected for some time, and a module is pulled from or plugged into a slot, this device may report a difference between the configured module for this slot and the real module being plugged into this slot, or pulled from this slot.

This method use alarms, which are very effective for reporting diagnostics, because they are fast and consume little bandwidth. Device only transmits alarms to the controller when a diagnostic changes state.

The following steps occur in case of pulling a module from a slot where some module is configured:

1. When the device observes that a module was pulled from the slot, it sends a "pull alarm" to controller.
2. When controller receives the "pull alarm", it sets diagnostic "no module" (absent module) for this slot.

The following steps occur in case of plugging a module into a slot where some module is configured:

1. When the device observes that a module was plugged, it sends a "plug alarm" to controller if the module is equal or compatible with configuration (equal module or substitute module). If the module is not equal and not compatible with configuration (wrong module), it sends a "plug wrong alarm" to controller.
2. If controller receives a "plug wrong alarm", it sets diagnostic "wrong module" for this slot.
3. If controller receives a "plug alarm", it sends parameterization to this module. After parameterization, the device informs if the plugged module is equal or substitute. If the module is substitute (compatible but not equal), the controller will set the "substitute module" diagnostic.

Automatic Polling of Diagnostics in Standard or USI Formats at Connection Time

When the controller establishes connection with a device, the device may have previously detected some diagnostics in standard or USI formats.

In this case, the device informs to controller that diagnostics are available. After this, the controller automatically issues a polling (read record function) for reading these diagnostics. After this, the device responds the polling informing with all active diagnostics in standard or USI formats.

Alarms in Standard or USI Formats Reported after Connection Time

If a device has been connected for some time, and a diagnostic in standard or USI format changes state (appears or disappears), the device sends an alarm to the controller. This method is very effective for reporting diagnostics, because it is fast and consumes little bandwidth.

When the controller receives an alarm informing that a diagnostic "appeared", it sets the diagnostic. When the controller receives an alarm informing that a diagnostic "disappeared", it resets the diagnostic.

User Requested Polling for Diagnostics in Standard or USI Formats

The user application of controller can request a polling (read record) of diagnostics in standard or USI formats, by calling a specific library function for this purpose.

This is possible but not very useful, because the previously described methods are sufficient for retrieving diagnostics and have the following advantages:

- User does not need to worry about calling functions cyclically for starting the pollings;
- The previously described methods use little bandwidth because they work in exceptions instead of cyclically (only at connection time and when diagnostics change state).

In theory, cyclic polling called by user application could be useful for compensating lost alarms. However, alarms have a strong acknowledgment scheme. Therefore, missing alarms is unlikely. When a device sends an alarm to a controller, this alarm is only deleted from the device after the controller sends an acknowledgment for it.

Visualization of Diagnostics using Mastertool Programming System

Diagnostics transmitted from devices to Nexto controller can be visualized using the Mastertool Programming system logged in the controller.

Observing the device tree, special marks indicate modules with problems:

- A red exclamation mark (❗) indicates that the module has active diagnostics;
- A red triangle mark (⚠) indicates that module is absent or wrong, or has communication failure;
- A black exclamation mark (⚫) indicates that the module had problems in the past, but now all these problems disappeared. For erasing the black exclamation mark, the user must press an "Acknowledge" button for clearing the alarm list of the module, as explained later in this section.

The next figure shows part of a device tree indicating that module AI4_01_GAS has active diagnostics (red exclamation mark).

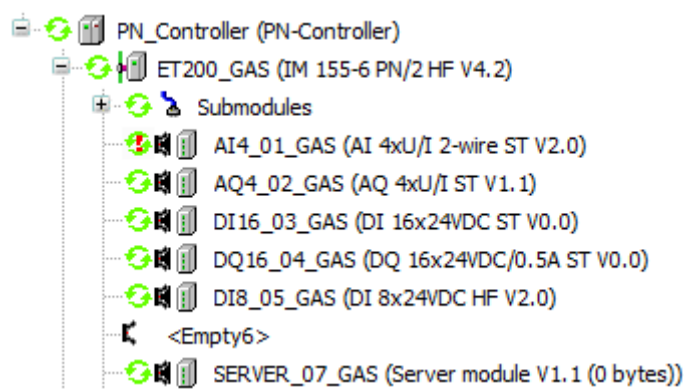


Figure 5-1. Device tree indicating active diagnostics in a module

If the user double-clicks on AI4_01_GAS in the device tree of the previous figure, a screen will open for this module. After selecting the "Status" tab in this screen, it will look like the following figure:

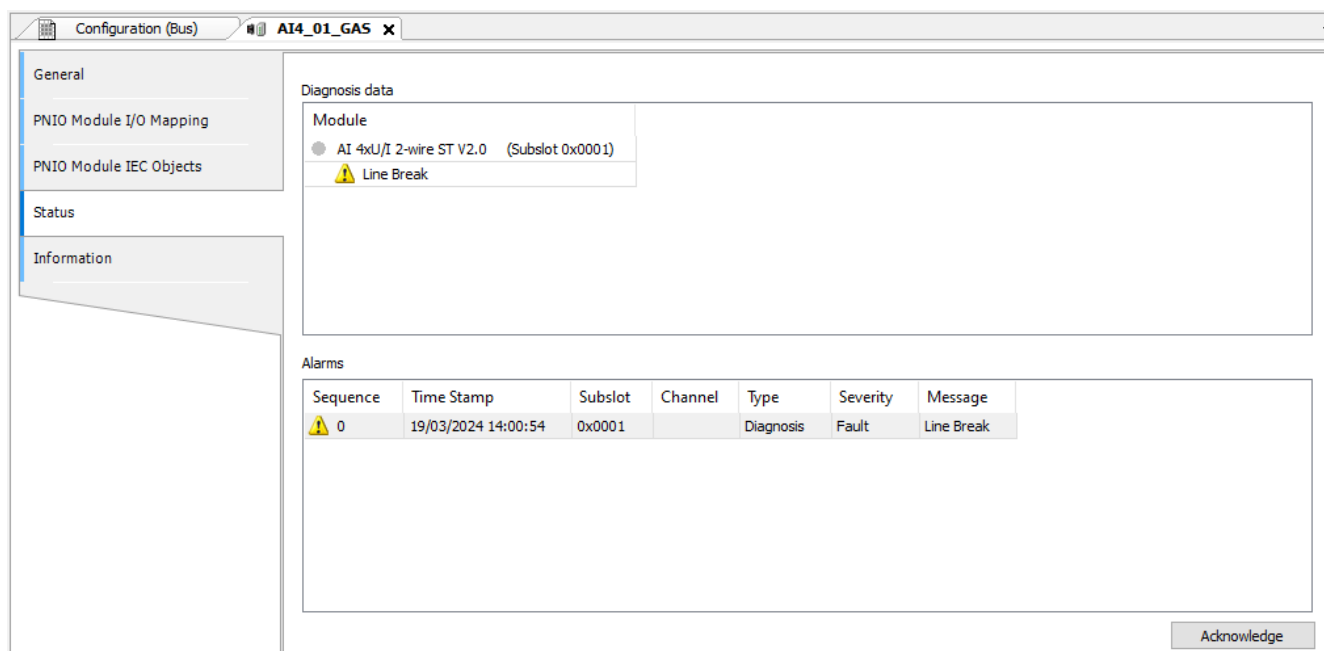


Figure 5-2. Status tab with active diagnostics and alarm history of a module

The previous figure shows:

- Diagnosis data in the upper side, that is, the currently active diagnostics for this module.
- An alarm history in the lower side, with timestamps indicating when alarms appeared and disappeared. The user can clear this alarm history by clicking on the button "Acknowledge".

The next figure shows part of a device tree indicating that module DI16_03_GAS is absent or wrong (red triangle mark).

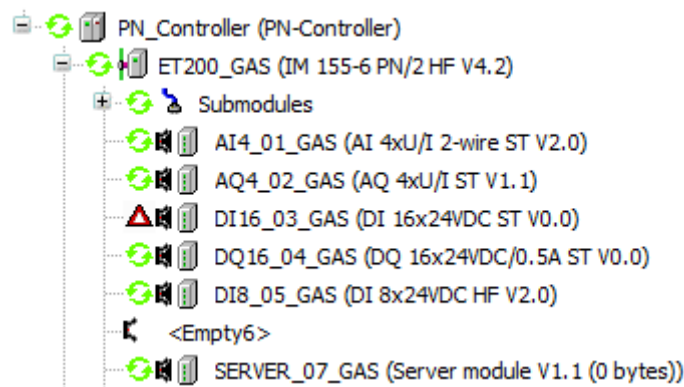


Figure 5-3. Device tree indicating absent or wrong module

If the user double-clicks on DI16_03_GAS in the device tree of the previous figure, a screen will open for this module. After selecting the "Status" tab in this screen, it will look like the following figure:

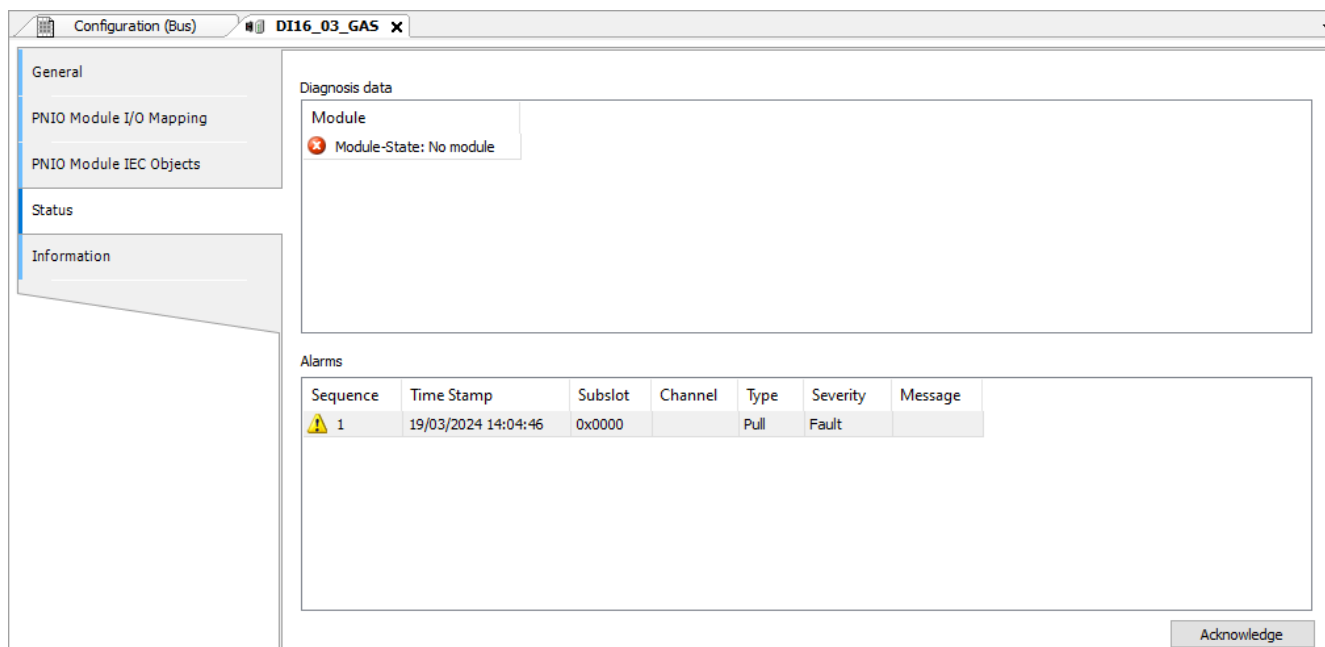


Figure 5-4. Status tab with active diagnostics and alarm history for absent or wrong module

In the previous figure, diagnosis data informs that module is absent. The alarms history shows when the pull alarm was received.

The next figure shows part of a device tree indicating that module DI16_03_GAS had problems in the past in which alarms were not acknowledged yet (black exclamation mark).

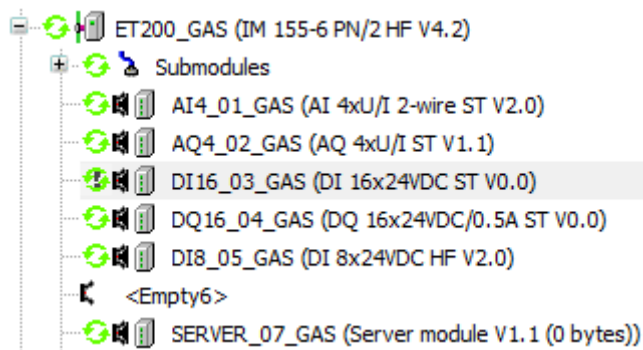


Figure 5-5. Device tree indicating no failure but unacknowledged alarms

If the user double-clicks on DI16_03_GAS in the device tree of the previous figure, a screen will open for this module. After selecting the "Status" tab in this screen, it will look like the following figure:

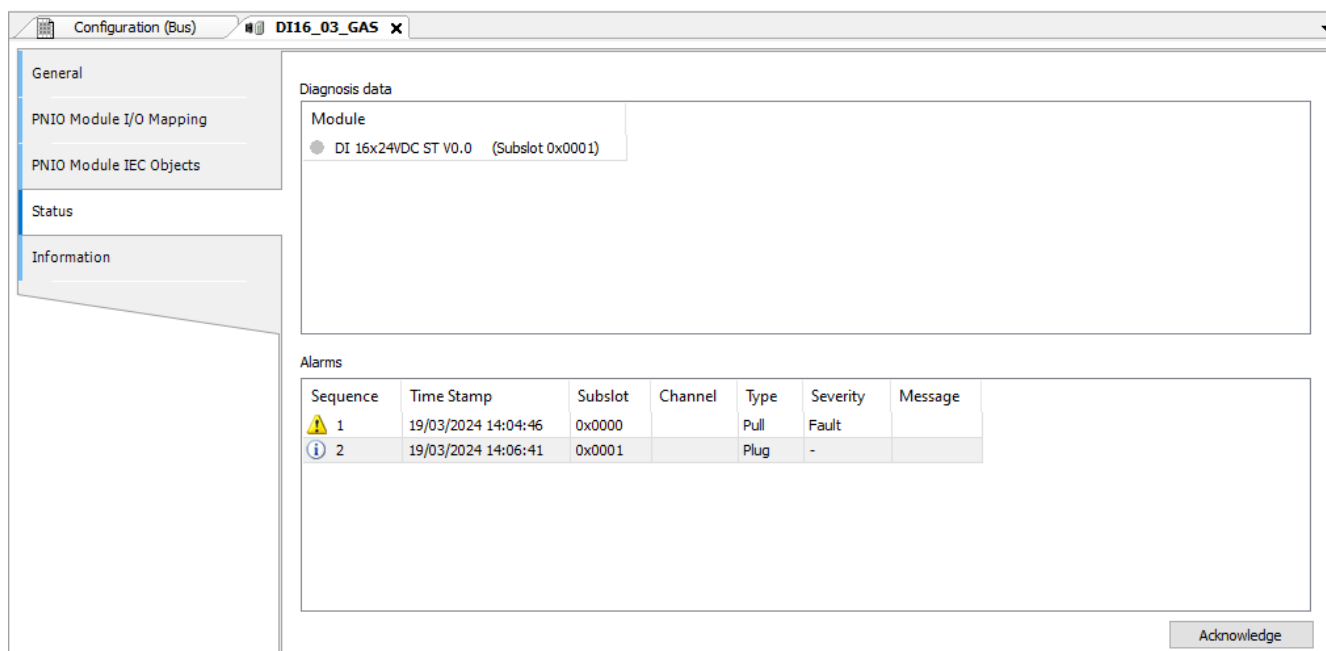


Figure 5-6. Status tab with active diagnostics and alarm history of a pulled module

In the previous figure, diagnosis data is empty. The alarms history shows when the module was removed (pull) and when it was inserted (plug).

If the user clicks on the "Acknowledge" button, two things will happen:

- The alarms history will be cleared;
- The black exclamation mark in the device tree on module DI16_03_GAS will be erased.

The next figure shows part of a device tree indicating that the entire remote station ET200_GAS has communication problems (red triangle mark in the remote head ET200_GAS and in all I/O modules installed in the remote station).

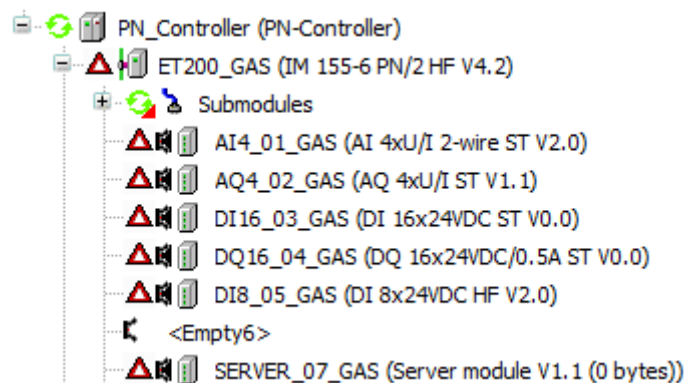


Figure 5-7. Device tree indicating communication failure in a remote station

If the user double-clicks on ET200_GAS in the device tree of the previous figure, a screen will open for this module. After selecting the "Status" tab in this screen, it will look like the following figure:

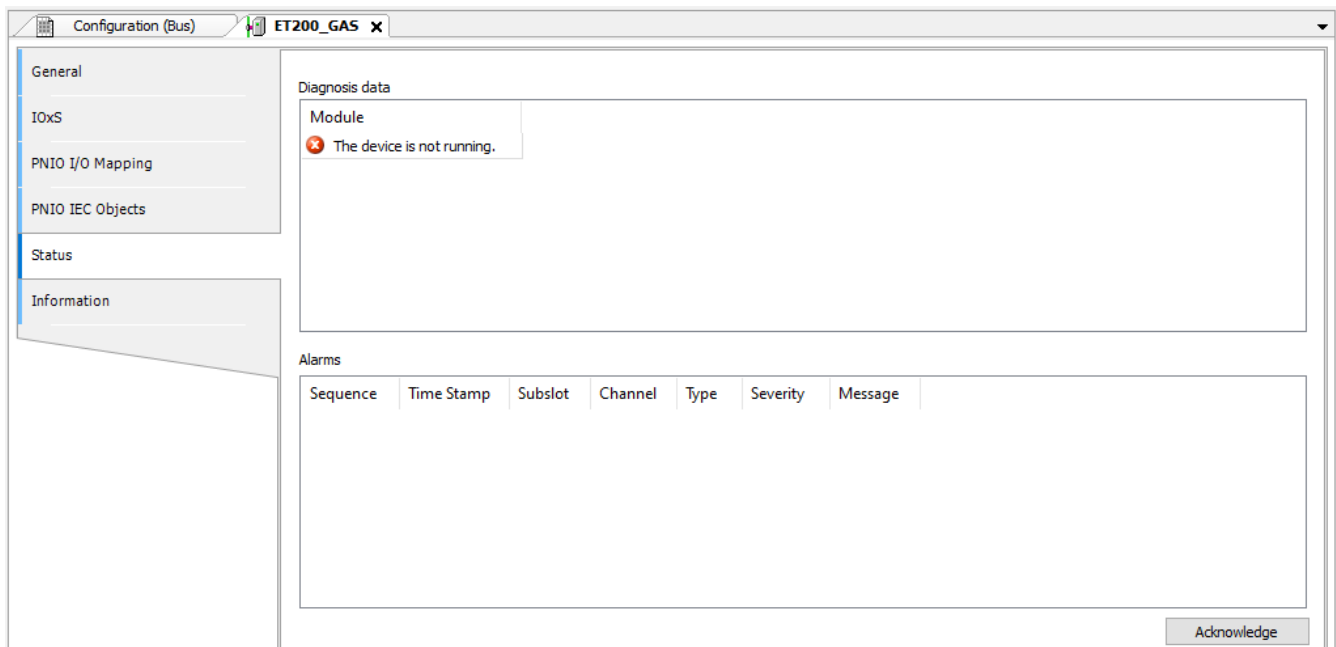


Figure 5-8. Status tab showing communication failure of a remote station

Device Communication Status for the User Application

A very important diagnostic that the user application must know is the communication status of each device. This diagnostic can be easily obtained, without needing to call functions or function blocks from any library.

For instance, the device tree in the following figure shows that device phoenix has communication problems, while all other devices have no communication problems.

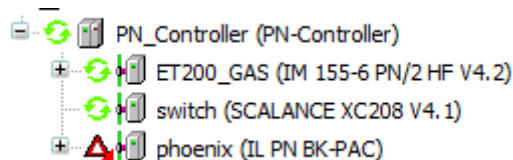


Figure 5-9. Device tree indicating communication failure in remote station phoenix

The field "xRunning" of a device indicates if it is connected and communicating successfully. For the example shown in the previous figure, the following figure shows the value of "xRunning" field for each device:

Watch 1			
Expression	Application	Type	Value
ET200_GAS.xRunning	Device.Application	BOOL	TRUE
switch.xRunning	Device.Application	BOOL	TRUE
phoenix.xRunning	Device.Application	BOOL	FALSE

Figure 5-10. Monitoring field xRunning of each device

Generic Library for Retrieving Diagnostics in the User Application

A generic library must be used for retrieving diagnostics of third-party devices. It is sufficient to use two function blocks of this generic library:

- ProfinetCommon.SubmoduleIterator: used for retrieving module differences (see section **Diagnosis Information - Module Differences**).
- ProfinetCommon.DiagnosisDataBuffer: used for retrieving the other types of diagnostics, in standard format or USI format (see sections **Diagnosis Information - Standard Format** and **Diagnosis Information - USI Format**).

Retrieving Module Differences

The following figure shows an example of a very simple function that can be used for detecting module differences in a device.

```

1  FUNCTION PN_GetModulesDiff
2  VAR_INPUT
3      Device : REFERENCE TO IoDrvProfinet.PNSlave;           // Reference to PN device
4      ModuleDiffStates : POINTER TO ProfinetCommon.SubmoduleState_IdentInfo; // Pointer to array with function results
5      MaxSlots : UINT;                                       // Maximum number of slots in device
6  END_VAR
7  VAR
8      submodule : ProfinetCommon.SubmoduleInfo;
9      SubModIterator : ProfinetCommon.SubmoduleIterator;
10     i : UINT;
11 END_VAR
12
13 // Initially assume as OK
14 FOR i := 0 TO MaxSlots - 1 DO
15     ModuleDiffStates[i] := ProfinetCommon.SubmoduleState_IdentInfo.OK;
16 END_FOR
17 // Check ModuleDiffStates only if device running
18 IF Device.xRunning THEN
19     SubModIterator.InitByID(Device.ID);
20     WHILE (SubModIterator.Next(submodule => submodule)) DO
21         IF (submodule.Slot < MaxSlots) THEN
22             ModuleDiffStates[submodule.Slot] := submodule.SubmoduleState.IdentInfo;
23         END_IF
24     END_WHILE
25 END_IF

```

Figure 5-11. Function for retrieving module differences

The following figure shows a program calling the function shown in the previous figure, for evaluating module differences inside remote station ET200_GAS. The function stores the results in array ET200_GAS_ModuleDiffState.

```

VerifyDiffModules x
1  PROGRAM VerifyDiffModules
2  VAR
3      ET200_GAS_ModuleDiffState : ARRAY [0 .. 7] OF ProfinetCommon.SubmoduleState_IdentInfo;
4  END_VAR
5
6  PN_GetModulesDiff(Device:= ET200_GAS, ModuleDiffStates:= ADR(ET200_GAS_ModuleDiffState[0]), MaxSlots:= 8);

```

Figure 5-12. Calling function for retrieving module differences

Some care must be taken when calling the function for avoiding exceptions in the controller's CPU:

- Define the array of ProfinetCommon.SubmoduleState_IdentInfo starting with index 0, and with the same number of elements passed in parameter MaxSlots (in this example, ARRAY [0 .. 7] is compatible with MaxSlots = 8).
- Pass address of index 0 of array of ProfinetCommon.SubmoduleState_IdentInfo (in this example, ADR(ET200_GAS_ModuleDiffState[0])).

The following figures show an example of monitoring the results produced by the function. In this example, slot 1 has a substitute module, slot 3 has a wrong module and slot 4 has an absent module.

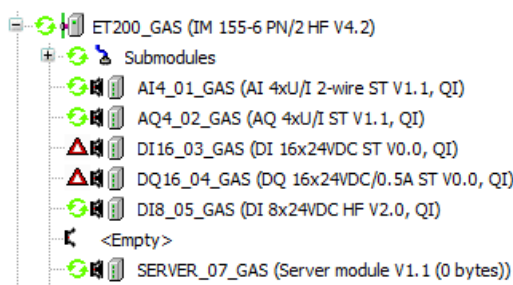


Figure 5-13. Device tree of remote station ET200_GAS with module differences

Note that the device tree does not show special marks for substitute modules (in this example, AI4_01_GAS at slot 1).

Expression	Type	Value
ET200_GAS_ModuleDiffState	ARRAY [0..7] OF SUBMODULESTATE_ID...	
ET200_GAS_ModuleDiffState[0]	SUBMODULESTATE_IDENTINFO	OK
ET200_GAS_ModuleDiffState[1]	SUBMODULESTATE_IDENTINFO	Substitute
ET200_GAS_ModuleDiffState[2]	SUBMODULESTATE_IDENTINFO	OK
ET200_GAS_ModuleDiffState[3]	SUBMODULESTATE_IDENTINFO	Wrong
ET200_GAS_ModuleDiffState[4]	SUBMODULESTATE_IDENTINFO	NoSubmodule
ET200_GAS_ModuleDiffState[5]	SUBMODULESTATE_IDENTINFO	OK
ET200_GAS_ModuleDiffState[6]	SUBMODULESTATE_IDENTINFO	OK
ET200_GAS_ModuleDiffState[7]	SUBMODULESTATE_IDENTINFO	OK

Figure 5-14. Monitored results for module differences

The following constants can be used for evaluating each result:

- OK module: ProfinetCommon.SubmoduleState_IdentInfo.OK
- Substitute module: ProfinetCommon.SubmoduleState_IdentInfo.Substitute
- Wrong module: ProfinetCommon.SubmoduleState_IdentInfo.Wrong
- Absent module: ProfinetCommon.SubmoduleState_IdentInfo.NoSubmodule

Retrieving other Types of Diagnostics - Example for an I/O Module

The other types of diagnostics can be retrieved using function block ProfinetCommon.DiagnosisDataBuffer.

For third-party devices, it is recommended to develop a function for each type of module. This section describes an example of function developed for retrieving diagnostics from a Siemens® digital input module (DI 8x24VDC HF - 6ES7131-6BF00-0CA0).

The first step is consulting the manual of the module for discovering which diagnostics are available.

This module has 3 diagnostics for each of the 8 digital inputs (ChannelNumber = 0 ... 7):

- Short circuit (ChannelErrorType = 16#0001).
- Wire break (ChannelErrorType = 16#0006).
- Supply voltage missing (ChannelErrorType = 16#0011).

In addition, the module has 3 diagnostics that apply to the whole module (ChannelNumber = 16#8000):

- Parameter assignment error (ChannelErrorType = 16#0010).
- Hardware interrupt lost (ChannelErrorType = 16#0016).
- Channel/component temporarily unavailable (ChannelErrorType = 16#001F).

The following figure shows a data structure created for storing the diagnostics of this module. Such a structure can be created adding a DUT object.

```

1  TYPE T_DIAG_DI8_24VDC_HF :
2  STRUCT
3      ParameterError : BOOL;           // Parameter assignment error for the module
4      HwIntLost : BOOL;                // Hardware interrupt lost error for the module
5      CompUnavailable : BOOL;         // Channel/component temporarily unavailable error for the module
6      WireBreak : ARRAY [0 .. 7] OF BOOL; // Wire break diagnostic for the 8 input channels
7      ShortCircuit : ARRAY [0 .. 7] OF BOOL; // Short circuit diagnostic for the 8 input channels
8      NoPower : ARRAY [0 .. 7] OF BOOL; // Supply voltage missing diagnostic for the 8 input channels
9  END_STRUCT
10 END_TYPE

```

Figure 5-15. Data structure for diagnostics of module DI 8x24VDC HF

The following figure shows a function that retrieves diagnostics from this module. When adding this POU of type function, use the previous structure (T_DIAG_DI8_24VCD_HF) as return type.

```

1  FUNCTION PN_GetDiags_DI8_24VDC_HF : T_DIAG_DI8_24VDC_HF
2  VAR_INPUT
3      Device : REFERENCE TO IODrvProfinet.PNSlave; // Reference to PN device
4      Slot : UINT; // Slot where DI 8x24Vdc HF is installed
5  END_VAR
6  VAR
7      diagBuffer : ProfinetCommon.DiagnosisDataBuffer;
8      i : DINT;
9      diag_DI8_24VDC_HF : T_DIAG_DI8_24VDC_HF;
10 END_VAR
11 // Reset all diagnostics for this module
12 SysMem.SysMemSet(pDest:= ADR(diag_DI8_24VDC_HF), udiValue:=0, udiCount:= SIZEOF(T_DIAG_DI8_24VDC_HF));
13 // Get diagnostic of the entire device
14 diagBuffer(xEnable := TRUE, ID := Device.ID, DiagnosisIndex := 0);
15 // Process diagnostics from the selected slot of the device
16 FOR i := 0 TO diagBuffer.DiagnosisCount -1 DO
17     diagBuffer(DiagnosisIndex := i);
18     // Check if some valid diagnostic is active for this slot and in standard format
19     IF NOT(diagBuffer.xError) AND diagBuffer.IsStandardFormat AND (Slot = diagBuffer.Source.Slot) THEN
20         //Check diagnostics for the whole module
21         IF diagBuffer.Source.ChannelNumber = 16#8000 THEN
22             CASE diagBuffer.Diagnosis.ChannelErrorType OF
23                 16#0010: diag_DI8_24VDC_HF.ParameterError := TRUE;
24                 16#0016: diag_DI8_24VDC_HF.HwIntLost := TRUE;
25                 16#001F: diag_DI8_24VDC_HF.CompUnavailable := TRUE;
26             END_CASE
27         END_IF
28         //Check input channel diagnostics
29         IF diagBuffer.Source.ChannelNumber < 8 THEN
30             CASE diagBuffer.Diagnosis.ChannelErrorType OF
31                 16#0001: diag_DI8_24VDC_HF.ShortCircuit[diagBuffer.Source.ChannelNumber] := TRUE;
32                 16#0006: diag_DI8_24VDC_HF.WireBreak[diagBuffer.Source.ChannelNumber] := TRUE;
33                 16#0011: diag_DI8_24VDC_HF.NoPower[diagBuffer.Source.ChannelNumber] := TRUE;
34             END_CASE
35         END_IF
36     END_IF
37 END_FOR
38 PN_GetDiags_DI8_24VDC_HF := diag_DI8_24VDC_HF;

```

Figure 5-16. Function for retrieving diagnostics of module DI 8x24VDC HF

The function has two input parameters:

- Reference to the device where the module is installed;
- Slot where the module is installed.

The following figure shows how this function can be called in a program for retrieving diagnostics of this module when installed in slot 5 of device ET200_GAS.

```

Verify3rdPartyModule x
1 PROGRAM Verify3rdPartyModule
2 VAR
3     Diag_DI8_05_GAS : T_DIAG_DI8_24VDC_HF;
4 END_VAR

1 Diag_DI8_05_GAS := PN_GetDiags_DI8_24VDC_HF(Device:= ET200_GAS, Slot:= 5);
    
```

Figure 5-17. Calling function for retrieving diagnostics of module DI 8x24VDC HF

The following figures show an example of monitoring results of diagnostics of this module, with wire-break failures in channels 2 and 6.

Sequence	Time Stamp	Subslot	Channel	Type	Severity	Message
3	16/08/2021 07:56:54	0x0001	2	Diagnosis	Fault	Line Break
4	16/08/2021 07:57:23	0x0001	6	Diagnosis	Fault	Line Break

Figure 5-18. Device tree and status tab of module DI 8x24VDC HF

Expression	Type	Value	A...	Comment
Diag_DI8_05_GAS	T_DIAG_DI8_24VDC_HF			
ParameterError	BOOL	FALSE		Parameter assignment error for the module
HwIntLost	BOOL	FALSE		Hardware interrupt lost error for the module
CompUnavailable	BOOL	FALSE		Channel/component temporarily unavailable error for the module
WireBreak	ARRAY [0..7] OF BOOL			wire break diagnostic for the 8 input channels
WireBreak[0]	BOOL	FALSE		
WireBreak[1]	BOOL	FALSE		
WireBreak[2]	BOOL	TRUE		
WireBreak[3]	BOOL	FALSE		
WireBreak[4]	BOOL	FALSE		
WireBreak[5]	BOOL	FALSE		
WireBreak[6]	BOOL	TRUE		
WireBreak[7]	BOOL	FALSE		
ShortCircuit	ARRAY [0..7] OF BOOL			short diagnostic for the 8 input channels
ShortCircuit[0]	BOOL	FALSE		
ShortCircuit[1]	BOOL	FALSE		
ShortCircuit[2]	BOOL	FALSE		
ShortCircuit[3]	BOOL	FALSE		
ShortCircuit[4]	BOOL	FALSE		
ShortCircuit[5]	BOOL	FALSE		
ShortCircuit[6]	BOOL	FALSE		
ShortCircuit[7]	BOOL	FALSE		
NoPower	ARRAY [0..7] OF BOOL			supply voltage missing diagnostic for the 8 input channels
NoPower[0]	BOOL	FALSE		
NoPower[1]	BOOL	FALSE		
NoPower[2]	BOOL	FALSE		
NoPower[3]	BOOL	FALSE		
NoPower[4]	BOOL	FALSE		
NoPower[5]	BOOL	FALSE		
NoPower[6]	BOOL	FALSE		
NoPower[7]	BOOL	FALSE		

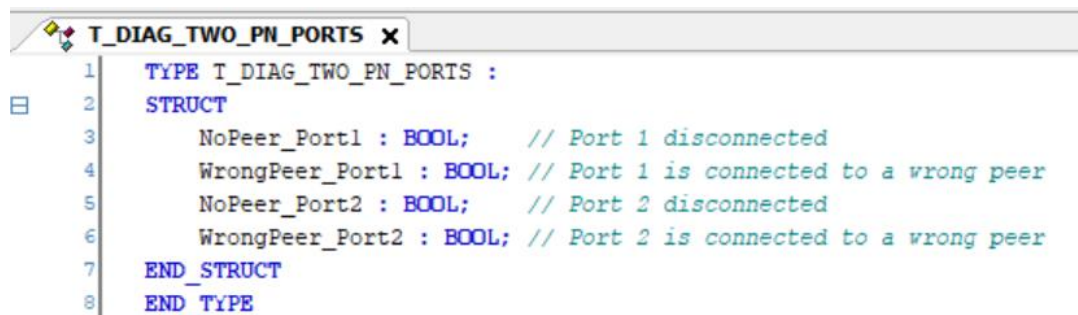
Figure 5-19. Monitored results of diagnostics of module DI 8x24VDC HF

Retrieving other Types of Diagnostics - Example for PROFINET Ports of a Remote Head

This section describes an example of a function developed for retrieving diagnostics from two switched PROFINET ports (X1_P1 and X1_P2) of remote head IM 155-6 PN/2 HF V4.2 from Siemens®.

Again, this function was implemented using function block ProfinetCommon.DiagnosisDataBuffer.

The following figure shows a data structure created for storing the diagnostics for these two PROFINET ports. Such a structure can be created by adding a DUT object. Basically, the diagnostics inform if a port is disconnected or connected to a wrong port.



```

1  TYPE T_DIAG_TWO_PN_PORTS :
2  STRUCT
3      NoPeer_Port1 : BOOL;    // Port 1 disconnected
4      WrongPeer_Port1 : BOOL; // Port 1 is connected to a wrong peer
5      NoPeer_Port2 : BOOL;    // Port 2 disconnected
6      WrongPeer_Port2 : BOOL; // Port 2 is connected to a wrong peer
7  END_STRUCT
8  END_TYPE

```

Figure 5-20. Data structure for diagnostics of two PROFINET ports

The following figure shows a function that retrieves these diagnostics from the remote head module. When adding this POU of type function, use the previous structure (T_DIAG_TWO_PN_PORTS) as return type.

Although this function was developed for remote head IM 155-6 PN/2 HF V4.2 from Siemens®, it probably also works for other remote heads with two switched ports.

Some codes used in this function are defined by PROFINET standard:

- Subslot = 16#8001 means port1, Subslot = 16#8002 means port2.
- ChannelErrorType = 16#8001 indicates a problem with the port. Additional information about the problem is provided by ExtChannelErrorType.
- ExtChannelErrorType = 16#8005 indicates no peer detected. Other codes indicates wrong peer (wrong station or wrong port of correct station).


```

PN_GetPortDiags x Configuration (XP)
1  FUNCTION PN_GetPortDiags : T_DIAG_TWO_PN_PORTS
2  VAR_INPUT
3      Device : REFERENCE TO IoDrvProfinet.PNSlave;           // Reference to PN device
4      Slot : UINT;                                         // Slot where remote head is installed
5  END_VAR
6  VAR
7      diagBuffer : ProfinetCommon.DiagnosisDataBuffer;
8      i : DINT;
9      diag_ports : T_DIAG_TWO_PN_PORTS;
10 END_VAR

1 // Reset all diagnostics
2 SysMem.SysMemSet(pDest:= ADR(diag_ports), udiValue:=0, udiCount:= SIZEOF(T_DIAG_TWO_PN_PORTS));
3 // Get diagnostic of the entire device
4 diagBuffer(xEnable := TRUE, ID := Device.ID, DiagnosisIndex := 0);
5 // Process diagnostics from the selected slot of the device
6 FOR i := 0 TO diagBuffer.DiagnosisCount -1 DO
7     diagBuffer(DiagnosisIndex := i);
8     // Check if some valid diagnostic is active for this slot and in standard format
9     IF NOT(diagBuffer.xError) AND diagBuffer.IsStandardFormat AND (Slot = diagBuffer.Source.Slot) THEN
10        //Check diagnostics with subslot Ethernet port 1
11        IF (diagbuffer.Source.Subslot = 16#8001) AND (diagbuffer.Diagnosis.ChannelErrorType = 16#8001) THEN
12            IF (diagbuffer.Diagnosis.ExtChannelErrorType = 16#8005) THEN
13                diag_ports.NoPeer_Port1 := TRUE;
14            ELSE
15                diag_ports.WrongPeer_Port1 := TRUE;
16            END_IF
17        END_IF
18        //Check diagnostics with subslot Ethernet port 2
19        IF (diagbuffer.Source.Subslot = 16#8002) AND (diagbuffer.Diagnosis.ChannelErrorType = 16#8001) THEN
20            IF (diagbuffer.Diagnosis.ExtChannelErrorType = 16#8005) THEN
21                diag_ports.NoPeer_Port2 := TRUE;
22            ELSE
23                diag_ports.WrongPeer_Port2 := TRUE;
24            END_IF
25        END_IF
26    END_IF
27 END_FOR
28 PN_GetPortDiags:= diag_ports;

```

Figure 5-21. Function for retrieving diagnostics of two PROFINET ports

The function has two input parameters:

- Reference to the device where the module is installed;
- Slot where the module is installed.

The following figure shows how this function can be called in a program for retrieving these diagnostics from device ET200_GAS. Note that remote head is installed at slot 0.

```

Verify3rdPartyPorts x
1  PROGRAM Verify3rdPartyPorts
2  VAR
3      DiagPorts : T_DIAG_TWO_PN_PORTS;
4  END_VAR

1  DiagPorts := PN_GetPortDiags(Device:= ET200_GAS, Slot:= 0);

```

Figure 5-22. Calling function for retrieving PROFINET port diagnostics of device ET200_GAS

The following figures show an example of monitoring results of port diagnostics of this remote head, with port 1 disconnected.

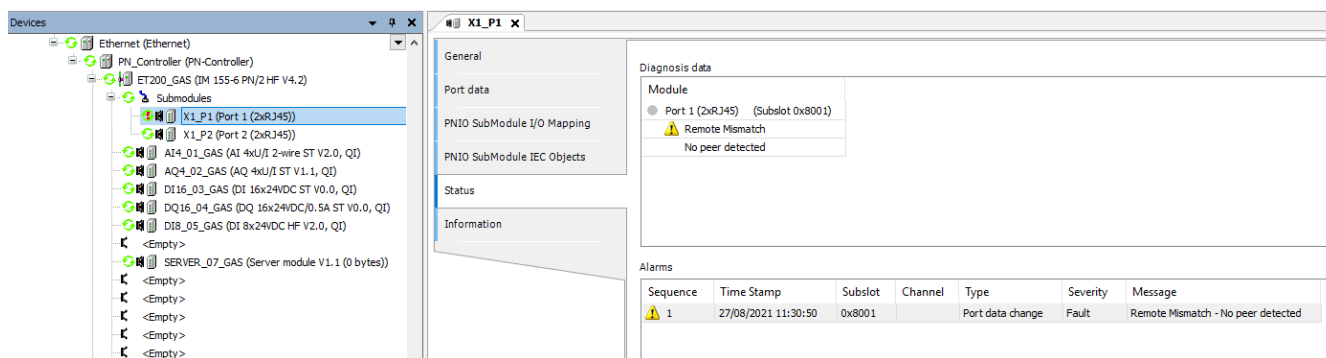


Figure 5-23. Device tree and status tab of port 1 of device ET200_GAS

Expression	Type	Value	Comment
DiagPorts	T_DIAG_TWO_PN_PORTS		
NoPeer_Port1	BOOL	TRUE	Port 1 disconnected
WrongPeer_Port1	BOOL	FALSE	Port 1 is connected to a wrong peer
NoPeer_Port2	BOOL	FALSE	Port 2 disconnected
WrongPeer_Port2	BOOL	FALSE	Port 2 is connected to a wrong peer

Figure 5-24. Monitored results of port diagnostics of device ET200_GAS

Controller Diagnostics

In tab "Status" of "PN_Controller", there is a screen with some diagnostics and statistics about the PROFINET controller.

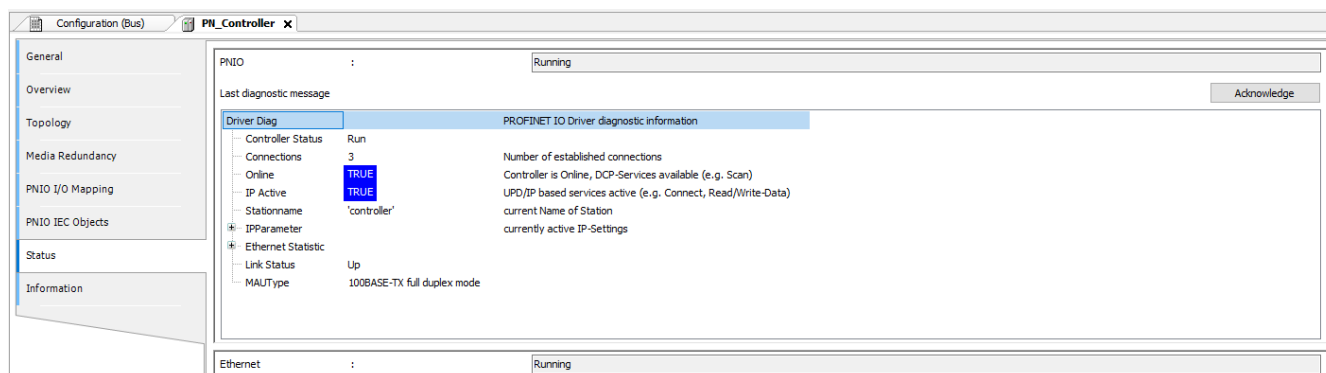


Figure 5-25. Status tab of PN_Controller

MRP Diagnostics

If the controller is connected to an MRP ring, there are diagnostics related to MRP in the GVL "System_Diagnostics".

The following figure shows an example of these diagnostics for the pair of ports NET2+NET3 of NX3008 configured in switch mode with MRP loop protection. For instance, if eRingState is OPEN, a maintenance action is required for closing the ring again.

Device.Application.System_Diagnostics			
Expression	Type	Value	
[-] DG_NX3008	T_DIAG_NX3008_1		
+ tSummarized	T_DIAG_SUMMARIZED		
[-] tDetailed	T_DIAG_DETAILED		
+ Target	T_DIAG_TARGET		
+ Hardware	T_DIAG_HARDWARE		
+ Exception	T_DIAG_EXCEPTION		
+ WebVisualization	T_DIAG_WEBVISUALIZATION		
+ RetainInfo	T_DIAG_RETAIN		
+ Reset	T_DIAG_RESET		
+ Thermometer	T_DIAG_THERMOMETER		
+ Serial	T_DIAG_SERIAL_SINGLE		
+ CAN	T_DIAG_CAN		
+ USB	T_DIAG_USB		
[-] Ethernet	T_DIAG_ETHERNET		
+ NET	ARRAY [1..3] OF T_DIAG_ETH_ADV...		
+ RSTP	ARRAY [0..0] OF T_DIAG_RSTP_BRI...		
[-] MRP	ARRAY [0..0] OF T_DIAG_MRP_RING		
[-] MRP[0]	T_DIAG_MRP_RING		
+ eConfiguredRingRole	ENUM_MRP_RING_ROLE	MANAGER	
+ eCurrentRingRole	ENUM_MRP_RING_ROLE	MANAGER	
+ eRingState	ENUM_MRP_RING_STATE	OPEN	
+ ePrimaryPort	ENUM_ETH_PORT_NAME	NET2	
+ eSecondaryPort	ENUM_ETH_PORT_NAME	NET3	
+ eManagerDiagnostics	ENUM_MRP_MANAGER_DIAGNOSTICS	OK	

Figure 5-26. MRP diagnostics

6. I&M Data

Identification and Maintenance (I&M) data supports unique identification of the devices, modules and submodules and their versions. This identification information is an important basis for maintaining the system and for asset management.

This information is specified in the I&M data structures. The I&M functions are subdivided into six different blocks (I&M0 to I&M5) and can be addressed separately using their index.

I&M0, I&M1, I&M2 and I&M3 are mandatory for all PROFINET devices.

I&M4 is necessary for devices following the common application profile PROFIsafe.

I&M5 is necessary if a submodule contains more than one function that must be identified. In this case, I&M5 data complement I&M0 data.

This manual will not describe I&M4 and I&M5.

Description of I&M Data Structures

I&M0

I&M0 provide some read-only parameters about the addressed device, module, or submodule:

- Vendor ID (type: WORD): this is a code that identifies the manufacturer. For instance, Siemens AG® has Vendor ID = 42. The list of all Vendor ID codes are registered at PROFIBUS and PROFINET Organization and can be found at the URL https://www.profibus.com/IM/Man_ID_Table.xml.
- Order ID (type: text with 20 characters): this is the part number of device, module or submodule.
- Serial Number (type: text with 16 characters): this is the serial number of device, module or submodule, used in production.
- Hardware Revision (type: WORD): this is a number indicating the hardware revision.
- Software Revision: this information indicates the software revision, using four fields:
 - Prefix (type: text with single character): this character may be:
 - “V” for an officially released version
 - “R” for Revision
 - “P” for Prototype
 - “U” for Under Test (Field Test)
 - “T” for Test Device
 - Functional Enhancement (type: BYTE): this field must be incremented when the software revision add new functions.
 - Bug Fix (type: BYTE): this field must be incremented when the software revision fixes a bug.
 - Internal Change (type: BYTE): this field must be incremented when the software revision corresponds to an internal change.
- Revision Counter (type UINT): A changed value of this parameter of a given module marks a change of hardware or of its parameters.
- Profile ID (type: WORD): Devices not following any application profile shall use 16#0000 (Non-Profile Device – preferred) or 0xF600 (Generic Device). PROFILE_ID are provided by PROFIBUS & PROFINET International (PI). PI provides and maintains a XML file (Profile_ID_Table) containing the assignment of PROFILE_IDs to profiles. It is available at the URL http://www.profibus.com/IM/Profile_ID_Table.xml.
- Profile Specific Type (type: WORD): In case a module follows a special profile (see Profile ID) this parameter offers further information about profile-specific details according to the respective definitions of the application profile. Devices/modules not following any application profile shall use 16#0000. For details about coding see IEC 61158-6-10 (PROFINET).

- I&M Version: This parameter indicates the implemented version of I&M functions, using two fields:
 - Version Major (type: BYTE): most significant byte.
 - Version Minor (type: BYTE): least significant byte.
- I&M Supported (byte WORD): This parameter indicates which I&M functions are supported through individual bits:
 - bit 0: must always be 0 (zero).
 - bit 1: indicates the I&M1 functions are supported at least for reading.
 - bit 2: indicates the I&M2 functions are supported at least for reading.
 - bit 3: indicates the I&M3 functions are supported at least for reading.
 - bit 4: indicates the I&M4 functions are supported at least for reading.
 - bit 5: indicates the I&M5 functions are supported at least for reading.
 - bits 6 to 15: for now must be 0 (zero). Reserved for future I&M functions.

I&M1

I&M1 provide some read-write parameters about the addressed device, module, or submodule:

- Function (type: text with 32 characters): describes the function executed by the device, module, or submodule in the application;
- Location (type: text with 22 characters): describes where the device, module, or submodule is installed.

I&M2

I&M2 provide some read-write parameters about the addressed device, module, or submodule:

- Date and Time (type: text with 16 characters): describes when the device, module, or submodule was installed. The format is "YYYY-MM HH:MM".

I&M3

I&M3 provide some read-write parameters about the addressed device, module, or submodule:

- Description (type: text with 54 characters): this is a free description of the device, module, or submodule.

Addressing I&M Data

When reading or writing I&M data, it is possible to address any submodule, by specifying device, slot, and subplot. Some IODs really have individual I&M data, at the module or submodule level.

Reading and Writing I&M Data for a Device in Mastertool Programming System

Mastertool Programming System has a screen that enables reading and writing I&M data at the device level. It is not possible to address modules or submodules.

Double-click on PN_Controller in the device tree, select the Topology tab and press the refresh button.

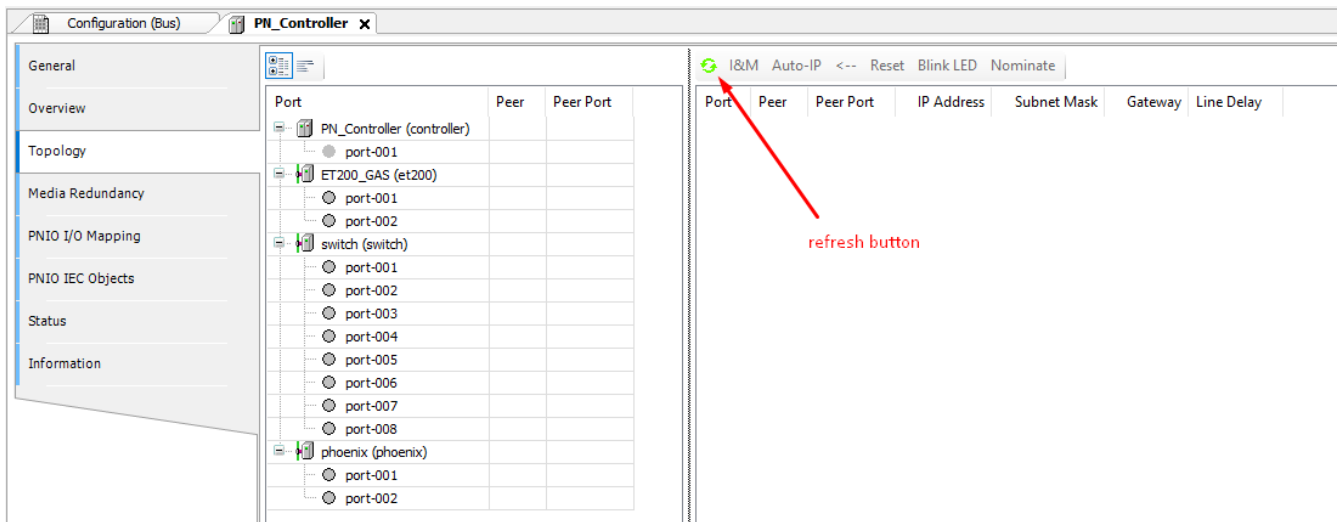


Figure 6-1. Refreshing topology screen

After this, the refreshed topology appears. For instance, it could be the following:

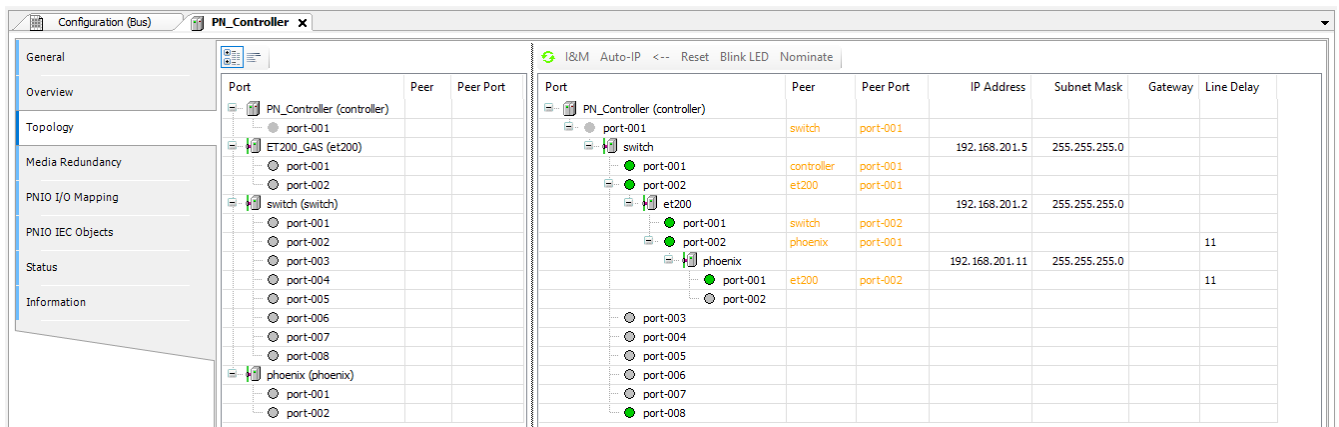


Figure 6-2. Refreshed topology screen

For accessing I&M data of a device, it must have a station name and an IP address. In this example, all devices have station name and an IP address.

Select the row of device that you want to access I&M data. For instance, et200. Then click on command "I&M".

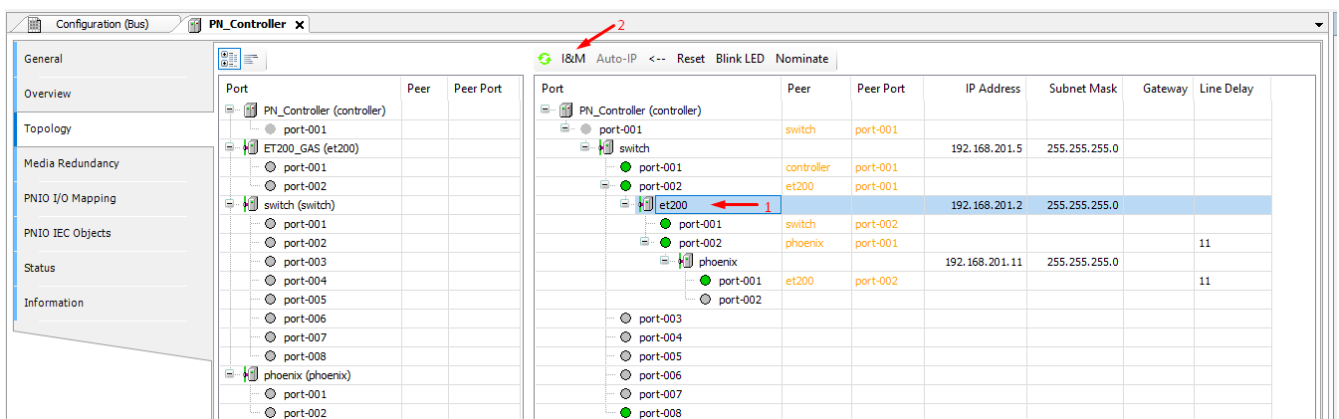


Figure 6-3. Starting I&M command

After clicking on the command "I&M", the following pop-up screen appears. The fields inside the red box can be written (I&M1, I&M2, and I&M3). The other fields are read-only (I&M0).

I&M Parameter	Value
Vendor ID	42
Order ID	6ES7 155-6AU01-0CN0
Serial Number	S C-LDDM33142019
Hardware Revision	2
Software Revision	V 4.2.2
Profile	
I&M Version	1.1
Location	
Function	
Date	
Description	

Manufacturer Info

[SIEMENS AG](#)

[Search product](#)

Hotline

Write I&M Close

Figure 6-4. Pop-up screen for reading and writing I&M data

For writing the writable values, enter some text in the "Value" column and, after this, press the button "Write I&M". The following figure shows an example:

I&M Parameter	Value
Vendor ID	42
Order ID	6ES7 155-6AU01-0CN0
Serial Number	S C-LDDM33142019
Hardware Revision	2
Software Revision	V 4.2.2
Profile	
I&M Version	1.1
Location	Control Room
Function	Fire detection
Date	2022-07-08 16:13
Description	Smoke and other detectors

Manufacturer Info

[SIEMENS AG](#)

[Search product](#)

Hotline

Write I&M Close

Figure 6-5. Pop-up screen after writing I&M data

Using Function Blocks for Reading or Writing I&M Data

Sometimes the user may want to use function blocks for reading or writing I&M data. This enables to address not only at device level, but also at the module level (slot) or at the submodule level (subslot).

Two function blocks can be used for this purpose:

- CommFB.RDREC: read a record acyclically;
- CommFB.WRREC: write a record acyclically.

The usage of these function blocks is explained at the online-help of Mastertool Programming System. The following hints explain some input parameters required by these function blocks:

- Example of ID parameter for read record or write record, specifying subslot 1 of slot 3 of device ET200_GAS:

```
ET200_GAS.GetID(API:=0, SLOT:=3, SubSlot:= 1);
```
- Indexes used for accessing I&M data with read record or write record:
 - I&M0: 16#AFF0
 - I&M1: 16#AFF1
 - I&M2: 16#AFF2
 - I&M3: 16#AFF3

7. Performance Analysis for PROFINET Controllers

The total percentual CPU consumption of a CPU can be observed in the diagnostic "tDetailed.Exception.byProcessorLoad" inside the GVL "System_Diagnostics". When a CPU is configured as a PROFINET controller, this function consumes a considerable percent of CPU.

Depending on the complexity and parameters of the PROFINET network, and on the other functions that must be executed in the same CPU, the processing power of the CPU may not be sufficient.

The objectives of this chapter are:

- List the main factors that affect CPU consumption in a PROFINET configuration.
- Give some hints for saving CPU consumption in a PROFINET configuration.
- Show the CPU consumption for some examples of PROFINET configurations, for two types of CPUs (NX3008 and Xpress).

Main Factors that Affect CPU Consumption in a PROFINET Configuration

The following subsections also contain some hints for saving CPU consumption in a PROFINET configuration.

CPU Model

The CPU models that support PROFINET controllers are listed in **Table 1-1. Features of Nexto controllers with PROFINET**.

Some CPU models have more processing power than others (for instance, NX3008 is faster than Xpress, and Xpress is a little bit faster than the other CPUs listed in the table).

Faster CPUs, of course, support bigger and more complex PROFINET configurations.

Interval of Profinet_IOTask and Send Clock

The interval of Profinet_IOTask can be 1 ms, 2 ms or 4 ms, but only for NX3008. For other CPUs (much slower than NX3008), this interval can only be 4 ms.

Even for NX3008, always try to configure 4 ms, unless some device needs an I/O cycle lower than 4 ms.

In addition, configure the "Send clock (ms)" parameter of all devices equal to the Profinet_IOTask interval.

I/O Cycle of Devices

The I/O cycle of a device is the product between the parameters "Send clock (ms)" and "Reduction rate" configured for this device. Lower I/O cycles consume more CPU.

Evaluate carefully which I/O cycle is really needed for each device. Try to keep the I/O cycle as big as possible for saving CPU. The I/O cycle really needed for a device depends on the required process response times.

I/O Data Size

Each device allocates certain amount of input bytes (%I variables) and output bytes (%Q variables) for exchanging I/O with the controller.

The CPU consumption increases with the amount of I/O bytes allocated.

Number of Devices

More devices increase the CPU consumption.

MRP

If the controller is connected to a MRP ring, the CPU consumption increases.

In addition, if the role of the controller is "MRP manager" it consumes more CPU than if the role of the controller is "MRP client".

Examples of CPU Consumption for CPUs Xpress

The following table lists tests that determine the CPU consumption added for executing the PROFINET management. The added CPU consumption appears in the rightmost column.

The tests were executed with XP340 firmware 1.14.23.0, and with Mastertool MT8500 3.61.

The Send clock parameter of all devices was always configured equal to the interval of Profinet_IOTask. For Xpress, the only possible value for the interval of Profinet_IOTask is 4 ms.

The column "Total I/O bytes" show the sum between input and output bytes exchanged between all devices and the controller.

Comparing different tests, it is possible to evaluate the effect of varying some of the main factors that influence CPU consumption, mentioned in section **Main Factors that Affect CPU Consumption in a PROFINET Configuration**.

Test	Profinet_IOTask (ms)	Number of Devices	Total I/O Bytes	I/O Cycle (ms)	%CPU added for PROFINET
	SendClock (ms)				
1	4	3	24	16	45,7%
2	4	3	24	32	42,2%
3	4	3	24	64	40,9%
4	4	3	24	128	39,6%
5	4	3	630	16	51,2%
6	4	3	630	32	48,2%
7	4	3	630	64	46,9%
8	4	3	630	128	45,1%
9	4	5	69	16	52,0%
10	4	5	69	32	48,9%
11	4	5	69	64	46,6%
12	4	5	69	128	45,9%
13	4	5	675	16	53,5%
14	4	5	675	32	52,5%
15	4	5	675	64	51,0%
16	4	5	675	128	50,1%
17	4	7	136	64	49,8%
18	4	7	136	128	48,6%
19	4	7	742	64	52,7%
20	4	7	742	128	51,7%
21	4	10	158	128	51,0%
22	4	10	158	256	50,4%
23	4	10	764	128	52,9%
24	4	10	764	256	52,6%

Table 7-1. Tests for CPU consumption of Xpress

Examples of CPU Consumption for CPU NX3008

The following table lists tests that determine the CPU consumption added for executing the PROFINET management. The added CPU consumption appears in the rightmost column.

The tests were executed with NX3008 firmware 1.14.23.0, and with Mastertool MT8500 3.61.

The Send clock parameter of all devices was always configured equal to the interval of Profinet_IOTask.

There are two types of devices used in the test:

- Group 1: devices which I/O cycle can be configured with smaller values, some of them starting at 1 ms.
- Group 2: switches which I/O cycle can be configured with higher values, starting at 128 ms.

For each group, the column "Total I/O bytes" show the sum between input and output bytes exchanged between all devices of this group and the controller.

Comparing different tests, it is possible to evaluate the effect of varying some of the main factors that influence CPU consumption, mentioned in section **Main Factors that Affect CPU Consumption in a PROFINET Configuration**.

7. Performance Analysis for PROFINET Controllers

Test	Profinet_IOTask (ms)	Group 1			Group 2 (switches)			MRP role of controller	%CPU added for PROFINET
	SendClock (ms)	Number of Devices	Total I/O Bytes	I/O Cycle (ms)	Number of Devices	Total I/O Bytes	I/O Cycle (ms)		
1	1	3	24	1				None	49,5%
2	1	3	24	2				None	47,0%
3	1	3	24	4				None	40,4%
4	2	3	24	2				None	34,9%
5	2	3	24	4				None	29,0%
6	4	3	24	4				None	22,0%
7	4	3	24	8				None	19,1%
8	4	3	24	16				None	16,6%
9	4	3	24	32				None	14,1%
10	4	3	24	64				None	13,2%
11	4	3	24	128				None	12,8%
12	4	3	24	256				None	12,4%
13	4	3	630	4				None	24,7%
14	4	3	630	8				None	22,4%
15	4	3	630	16				None	19,6%
16	4	3	630	32				None	17,2%
17	4	3	630	64				None	16,4%
18	4	3	630	128				None	16,0%
19	4	3	630	256				None	15,4%
20	4	6	91	4				None	27,4%
21	4	6	91	8				None	23,5%
22	4	6	91	16				None	20,6%
23	4	6	91	32				None	18,1%
24	4	6	91	64				None	16,5%
25	4	6	91	128				None	15,7%
26	4	6	91	256				None	14,9%
27	4	6	697	4				None	30,5%
28	4	6	697	8				None	26,5%
29	4	6	697	16				None	23,7%
30	4	6	697	32				None	21,2%
31	4	6	697	64				None	19,2%
32	4	6	697	128				None	18,0%
33	4	6	697	256				None	17,5%
34	4	7	136	4	3	22	128	None	30,1%
35	4	7	136	8	3	22	128	None	26,7%
36	4	7	136	16	3	22	128	None	23,9%
37	4	7	136	32	3	22	128	None	21,0%
38	4	7	136	64	3	22	128	None	19,3%
39	4	7	136	128	3	22	128	None	18,2%
40	4	7	136	256	3	22	256	None	17,1%
41	4	7	742	4	3	22	128	None	36,2%
42	4	7	742	8	3	22	128	None	29,1%
43	4	7	742	16	3	22	128	None	26,3%
44	4	7	742	32	3	22	128	None	24,0%
45	4	7	742	64	3	22	128	None	21,5%
46	4	7	742	128	3	22	128	None	20,6%
47	4	7	742	256	3	22	256	None	20,0%
48	4	7	136	8	3	22	128	Manager	34,0%
49	4	7	136	16	3	22	128	Manager	29,6%
50	4	7	136	32	3	22	128	Manager	26,9%
51	4	7	136	64	3	22	128	Manager	25,1%
52	4	7	136	128	3	22	128	Manager	23,5%
53	4	7	136	256	3	22	256	Manager	22,5%
54	4	7	136	8	3	22	128	Client	30,5%
55	4	7	136	16	3	22	128	Client	26,0%
56	4	7	136	32	3	22	128	Client	23,9%
57	4	7	136	64	3	22	128	Client	21,3%
58	4	7	136	128	3	22	128	Client	20,0%
59	4	7	136	256	3	22	256	Client	19,2%

Table 7-2. Tests for CPU consumption of NX3008