# Migration Guide

# PROFINET IO-Device

## Migration from V3.x to V4.x and V5.x

**Hilscher Gesellschaft für Systemautomation mbH**

**www.hilscher.com**

DOC190604MG01EN | Revision 1 | English | 2019-08 | Released | Public

# Table of contents

# 1 Introduction

## 1.1 About this document

This document describes the major differences between

- the PROFINET IO Device stack V3.x,
- the PROFINET IO Device stack V4, and
- the PROFINET IO Device stack V5.

This document gives an overview about required modifications to existing application when migrating to a newer PROFINET IO Device stack generation.

Within this document the following short forms are used:

- PROFINET IO Device stack/firmware V3.x = PNSv3
- PROFINET IO Device stack/firmware V4.x = PNSv4
- PROFINET IO Device stack/firmware V5.x = PNSv5

This document covers the following migration scenarios for NXLFW:

- PNSv3 to PNSv4
- PNSv4 to PNSv5

## 1.2 List of revisions

| Rev | Date | Name | Chapter | Revision |
|---|---|---|---|---|
| 1 | 2019-08-23 | BMe | all | Created |

*Table 1: List of revisions*

## 1.3 Intended reader

This document is suitable for software developers with the following background:

- Knowledge of the PROFINET communication system
- Knowledge of the Hilscher PROFINET IO Device stack V3.x API
- Knowledge of the netX dual-port memory interface and communication

# 1.4 References to documents

**Documents related to PROFINET IO Device stack/firmware V3.x**

[1] Hilscher Gesellschaft für Systemautomation mbH: Protocol API, PROFINET IO RT/IRT Device, V3.14, Revision 19, English, 2019.

[2] Hilscher Gesellschaft für Systemautomation mbH: Migration Guide, PROFINET IO Device, Migration from V3.x to V3.14, Revision 10, English, 2019.

[3] Hilscher Gesellschaft für Systemautomation mbH: Packet API, netX Dual-Port Memory, Packet-based services netX 10/50/51/52/100/500, Revision 3, DOC161001API03EN, English, 2019. Same document as reference [7].

[4] Hilscher Gesellschaft für Systemautomation mbH: Protocol API, TCP/IP, Packet Interface, V2.4, Revision 14, DOC050201API14EN, English, 2017.

**Documents related to PROFINET IO Device stack/firmware V4.x**

[5] Hilscher Gesellschaft für Systemautomation mbH: Protocol API, PROFINET IO Device, V4.5.0, Revision 3, DOC171101API03EN, English, 2019.

[6] Hilscher Gesellschaft für Systemautomation mbH: Migration Guide, PROFINET IO Device, Migration from V4.x to V4.5, Revision 2, DOC171102MG02EN, English, 2019.

[7] Hilscher Gesellschaft für Systemautomation mbH: Packet API, netX Dual-Port Memory, Packet-based services netX 10/50/51/52/100/500, Revision 3, DOC161001API03EN, English, 2019. Same document as reference [3].

[8] Hilscher Gesellschaft für Systemautomation mbH: Protocol API, Socket Interface, Packet Interface, Revision 5, DOC140401API05EN, English, 2019. Same as reference [12].

**Documents related to PROFINET IO Device stack/firmware V5.x**

[9] Hilscher Gesellschaft für Systemautomation mbH: Protocol API, PROFINET IO Device, V5.1.0, Revision 1, DOC190103API01EN, English, 2019.

[10] Hilscher Gesellschaft für Systemautomation mbH: Packet API, netX Dual-Port Memory, Packet-based services netX 90/4000/4100, Revision 3, DOC190301API03EN, English, 2019.

[11] Hilscher Gesellschaft für Systemautomation mbH: Application note, Fragmentation of packets, Revision 1, DOC181003AN01EN, English, 2019.

[12] Hilscher Gesellschaft für Systemautomation mbH: Protocol API, Socket Interface, Packet Interface, Revision 5, DOC140401API05EN, English, 2019. Same as reference [8].

**Documents related to each PROFINET IO Device stack/firmware**

[13] Hilscher Gesellschaft für Systemautomation mbH: Dual-Port Memory Interface Manual, netX based Products, Revision 16, DOC060302DPM16EN, English, 2019.

[14] PROFIBUS International: Technical Specification for PROFINET IO: Application Layer protocol for decentralized periphery, Version 2.3Ed2MU3, March 2016, Order No. 2.722, English.

[15] Hilscher Gesellschaft für Systemautomation mbH: Protocol API, Ethernet, Packet interface, V4.4.0, Revision 9, DOC060901API09EN, English, 2017.

*Table 2: References to documents*

# 2 Migration paths

## 2.1 Overview migrations paths

Figure 1 shows the migration path and steps.

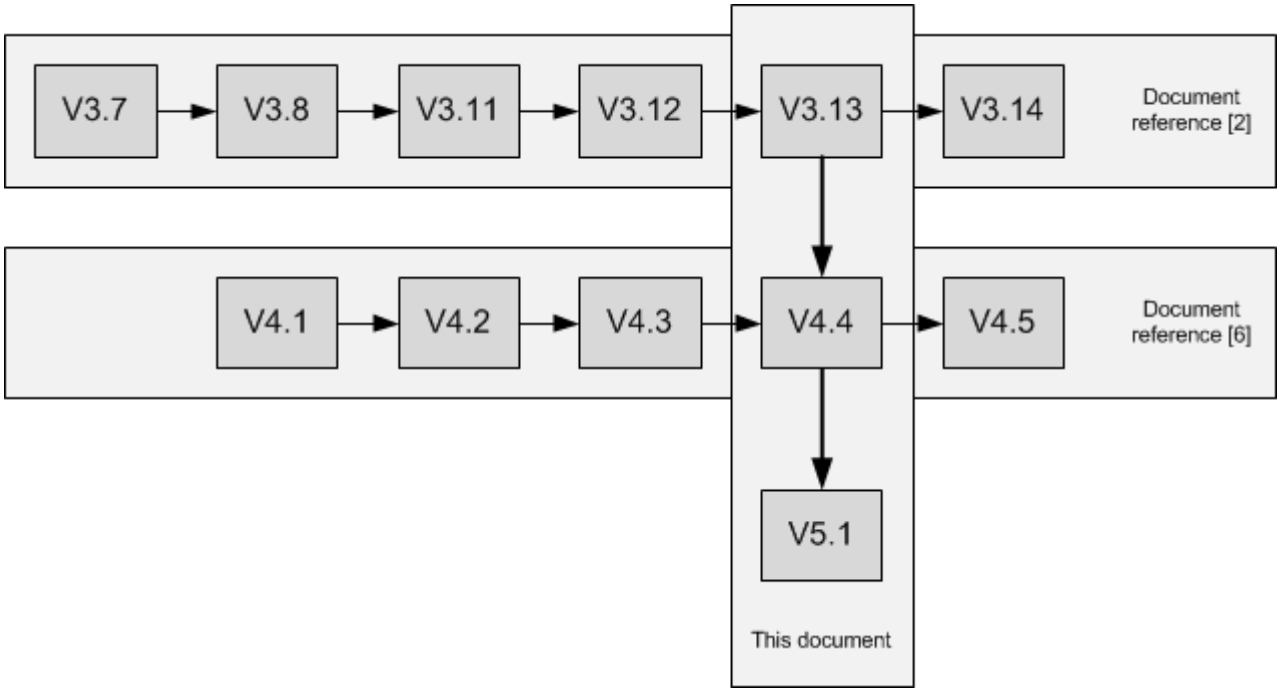



*Figure 1: Overview migrations paths*

The migration from NXLFW PNSv3 to NXLFW PNSv5 is covered by two steps

1. first, check the changes introduced with PNSv4 and
2. afterwards, check the changes introduced with PNSv5.

This document shows migration based on V3.13.0.x to V4.4.0.x and V4.4.0.x to V5.1.0.x.

The migration within a specific PROFINET IO Device stack/firmware generation is described in

- Migration from V3.x to V3.14, see reference [2],
- Migration from V4.x to V4.5, see reference [6].

**Migration to another firmware/stack generation**

Do the following steps to migrate from V3.11.0.2 to V4.5.0.1:

1. Migrate from V3.11 to V3.12 to V3.13 as documented in reference [2].
2. Migrate from V3.13 to V4.4 as documented in section *Changes of API and features between V3.x to V4.x* (page 9).
3. Migrate from V4.4 to V4.5 as documented in reference [6].

## 2.2 Overview stack generation

The following figures provide an overview about the stack/firmware generations including PROFINET features and netX chip type for several use cases.

| Stack generation | Status | PROFINET specification | NXLOM | NXLFW | Remark |
|---|---|---|---|---|---|
| V5.x | RELEASED | V2.35 | ✗ | ✓ | new generation for netX90/4x00 |
| V4.x | RELEASED | V2.35 | ✗ | ✓ | |
| V3.5 to V3.x | MAINTAINED | V2.34 | ✓ | ✓ | feature freeze<br>netload approved |

*Figure 2: Overview stack generation*

| Profinet Feature | Status V3.x | Status V4.x | Status V5.x |
|---|---|---|---|
| IRT | ✓ | ✓ | ✓ |
| SystemRedundancy S2 | ✗ | ✗ * | ✗ ** |
| AssetManagment | ✓ | ✓ | ✓ |
| I&M5 | ✓ | ✓ | ✗ ** |
| Profienergy ASE | ✓ | ✓ | ✗ ** |

* planned for future release (except netX52)

** planned for future release

*Figure 3: Overview stack generation and features*

| PROFINET IO-Device Version | netX 10 | netX 50 | netX 51 | netX 52 | netX 100 | netX 500 | netX90 | netX4x00 |
|---|---|---|---|---|---|---|---|---|
| V5.x | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✓ | ✓ |
| V4.x | ✗ | ✗ | ✓ | ✓ | (✓) * | (✓) * | ✗ | ✗ |
| V3.x | ✗ | ✓ | ✓ | ✗ | ✓ | ✓ | ✗ | ✗ |

*Figure 4: Overview stack generation and netX chip type*

Note: * NXLFW netX100/500 for PNS V4 in preparation. Will be available with next upcoming release V4.5.0.0

### Migration path that are not possible

- NXLOM PNSv3 to NXLOM PNSv4/v5 because NXLOM is available for PNSv3 only
- NXLFW PNSv3 to NXLFW PNSv4/v5 for netX 50 because PNSv4/v5 do not support netX 50
- NXLFW PNSv3 to NXLFW PNSv5 for netX 51/100/500 because PNSv5 do not support netX 51/100/500

# 3 Structural differences of V3/V4/V5

## 3.1 System services

A major difference between the stack/firmware generations is the used operating system (see section *Overview stack generation* page 6):

- PNSv3 and PNSv4 are designed to support the operating system rcX only
- PNSv5 is designed to support support the operating system eCos only

As a consequence, the system services available in PNSv3 are available in PNSv4 as well.

Some system services supported by PNSv3/PNSv4 are not available in PNSv5.

## 3.2 Protocol stack structure

PNSv3 is a “stand-alone” protocol stack. It has no shared component with PNSv4 or PNSv5.

PNSv4 and PNSv5 however share a basic PROFINET component called “PROFINET IO Core”. Therefore, similar behavior for many different topics exists between PNSv4 and PNSv5 while PNSv3 behaves differently. The next chapters cover the details.

## 3.3 Packet fragmentation on Communication Channel

Relevant for migration to PNSv5: Due to the different operating system used in PNSv5, the DPM handling for packet fragmentation is implemented differently. As a consequence, PNSv5 uses an other packet fragmentation mechanism for Communication Channel services. For a description of this packet fragmentation mechanism, see reference [11].

# 4 Changes of API and features between V3.x to V4.x

## 4.1 PROFINET Packet API

The packet API of the PNSv4 is mostly identical to the packet API of PNSv3 except the differences described below. Nevertheless, the behavior of the protocol stack was changed for some packets to fulfill resource requirements or to simplify usage. The following sections describe these changes.

This section may not be complete. Please check the Protocol API Manual for PNSv4 details for the specific services (reference [5]).

### 4.1.1 Configuring the IO-Device stack

#### Set Configuration Service

The field `ulTotalConfigPckLen` is not evaluated by the stack. Nevertheless, it is strongly recommended to set the value of this field to the total length of the data part of the packet including the field itself.

#### Configure Signal service

PNSv4 does not support byte order swapping of process data. Therefore, this request is not supported. The stack confirms to application with the status “Unknown command”.

#### Set OEM Parameter service

All parameter types can only be changed in unconfigured state.

The parameter types 9, 14 and 15 are not supported.

#### Set Isochronous Mode service

This service is not supported by PNSv4.

### 4.1.2 Connection establishment

#### AR Check service

The indication packet was extended with additional fields.

#### Connect Request Done service

The Connect Response on the network is not sent until the application sends the Connect Request Done Response to the PROFINET IO Device V4 protocol stack. In contrast, PROFINET IO Device V3 protocol stack did not wait for the response.

This service can be used as trigger in the application to start adjusting the submodule configuration if desired.

#### Parameter End service

Setting application ready for a specific AR is not possible anymore. If the `fSetApplReady` flag is set to `True`, all ARs will be set by the stack to application ready.

**Application Ready service**

The device handle argument of set Application Ready service is ignored. There is no AR-specific application ready service anymore. All pending ARs will be set by the stack to application ready. (relevant for Shared Device ARs). This does not affect future ARs, which must be set to Application Ready when they are established.

**Store Remanent Data service**

If configured, the protocol stack will send remanent data to the application instead of storing it within the on-board flash. The service might occur at different places and the application always must respond.

### 4.1.3 Acyclic events indicated by the stack

**General**

The protocol stack has implemented a packet timeout supervision for the application behavior. If the application does not send the required response to an indication within the defined time period, the protocol stack will generate a User Error Indication.

In contrast to PNSv3, the PNSv4 implementation will NOT perform any additional action. It waits until the response is received.

As a consequence, applications that do not respond to an indication will leave the stack stalled and the whole device may no longer work as expected.

**Save Station Name service**

The DCP Set Station Name response is sent to the bus after the application responds to the Save Station Name service or after a timeout of approx. 900 ms to fulfil certification requirements.

**Save IP Address service**

The DCP Set IP Address response is sent to the bus after the application responds to the Save IP Address service or after a timeout of approx. 900 ms to fulfil certification requirements.

**Reset Factory Settings service**

The DCP Set Reset Factory response is sent to the bus after the application responds to the Reset Factory Settings service or after a timeout of approx. 900 ms to fulfil certification requirements.

**Alarm Indication service**

This service is used to indicate incoming alarms to the application. The alarm acknowledge will be generated after the application responds to the Alarm Indication.

**Release Received service**

This service is not implemented in the PNSv4. In order to distinguish an AR Release by the Controller from an AR abort due to an error, the PNIO Status field of Abort Service is to be used. In case of AR Release by the Controller, the PNIO status field will be set to value PNIO_E_RTA_PROTOCOL_RELEASE_IND_RECEIVED (0xCF81FD0F).

### 4.1.4 Acyclic events requested by the application

**Process Alarm service**

This service is still supported but consider using the Send User Alarm service instead.

The application should not send more than one alarm request at a time to the protocol stack. After the protocol stacks confirms the process alarm then the next Process Alarm Service can be activated.

Please note that the protocol stack has only limited capability of queuing process alarms. If the application sends more than one alarm request at a time, the stack has to buffer the packet. Therefore, the stack uses the DPM Mailbox packet buffers. Each pending Process Alarm Service will occupy one packet buffer, but these buffers are used for other application initiated services as well. Thus it is strongly recommended to implement process alarm queuing at application level.

**Diagnosis Alarm service**

This service is not used anymore. The PROFINET IO specification defines that diagnosis alarms are to be sent when a diagnosis appears and when a diagnosis disappears. The protocol stack will automatically initiate the corresponding alarms if needed. If this packet is sent to the protocol stack it will not result in any action of the protocol stack. It will be responded with status code TLR_S_OK / SUCCESS_HIL_OK for backwards compatibility.

**Plug Submodule/Extended Plug Submodule service**

In contrast to version V3.x protocol stack, this service will be responded immediately to the application and then communicate with the IO-Controller. Any other requests associated with a Plug Submodule will occur afterwards, esp. handling of alarms will be done afterwards.

**Add Generic Channel Diagnosis service**

Generally, using this kind of diagnosis is strongly discouraged by the PI Diagnosis Guideline. They should be used in justified exceptions to this rule only.

Generic Diagnoses contain application specific amount of user data. Thus, the memory must be allocated dynamically at runtime and can not be predicted. In turn, the successful creation of a generic diagnosis entry depends on the following runtime properties:

- Available dynamic memory
- Memory Fragmentation due to runtime memory allocation

Therefore, there can be no guarantee for the success of this service under all operating conditions. It is strongly recommended to not use this service. Applications requiring this service should be tested carefully regarding this scenario.

**Return of Submodule Alarm service**

This service is not required anymore and it will not trigger any action of the stack. Since PNSv4 handling regarding this functionality will be performed by the stack automatically. The associated request packet will be returned to application with status TLR_S_OK / SUCCESS_HIL_OK always for backwards compatibility.

**Get Xmac Diagnosis service**

This service is available since version V4, but uses a new format. The information is not intended to be used by the application. This packet is for support purposes only and is only used to help debugging Ethernet problems.

# 4.2 Tag list

## 4.2.1 Quantity structure

From the general protocol stack structure, the PNSv4 stack is different compared to PNSv3. This has impact to some parameters that need to be known upfront of starting the protocol stack.

Therefore, the following parameters are part of the tag list of all PNSv4-based firmware and need to be set to match the application requirements.

- Number of useable submodule
- Minimum size of RPC buffers
- Number of diagnosis elements that can be added by application

Note that these parameters influence the memory requirements of the firmware.

When migrating from PNSv3 to PNSv4 on netX 51 or netX 100/500 note that by default SharedDevice is now set to a total of 4 IO ARs. PNSv3 had a default of 2 IO ARs.

## 4.2.2 Protocol stack behavior

Some tag list parameters that exist in PNSv3-based firmware no longer exist in PNSv4-based firmware. Instead, these parameters can be modified via the mailbox at firmware runtime (prior setting Bus ON).

| **Tag list parameter in PNSv3 firmware** | **Corresponding service in PNSv4 firmware** | **Description** |
|---|---|---|
| “Profinet Features”.”IO Supervisor communication accepted” | PNS_IF_SET_OEM_PARAMETERS_TYPE_12 | If the firmware should accept IO Supervisor connections |
| “Profinet Features”.”IRT Communication accepted” | PNS_IF_SET_OEM_PARAMETERS_TYPE_11 | If the firmware should accept IRT connections |
| “Profinet Features”.”MinDeviceInterval” | PNS_IF_SET_OEM_PARAMETERS_TYPE_13 | The fastest cycle time the firmware should accept |
| “DPM Behaviour”.”COM State Legacy Mode” | Not available | The legacy mode for DPM CommunicationState is no longer supported |
| “Interrupt: RX Timer” | Not available | Due to the internal firmware structure, it is no longer required to choose between “deterministic IO exchange” and “DPM Watchdog functionality in netload situations” |

## 4.3 Limitations and dropped features

The following limitations and unsupported features exist when migrating from PNSv3 to PNSv4

- Swapping the byte order of cyclic process data is not supported. Cyclic process data is always provided in network byte order (big endian).
- No loadable firmware offers an integrated webserver
- “old raw Ethernet API” is not available. This “old” API was only available in PROFINET IO Device stack V3 up to 3.10.0.0.
- Fiber Optic hardware is not supported.
- Support of “One-port mode” is not available. Using a PNSv4-based firmware, it is not possible to build a single-port device when using switch-based firmware (which is the default firmware).
- Legacy “Warmstart” configuration service is not supported. Use SetConfiguration service instead.

## 4.4 IRT

PNSv4-based firmware will generate the hardware sync signal for every red phase of the network, even if in the specific phase the netX itself does not send/receive data.

In contrast, the PNSv3-based firmware generates this signal only for red phases where the netX actively sends or receives data.

This difference only has an impact if the sync signal is used by the application or external hardware.

This difference has no impact if the netX cycle time is equal to the so called SendClock which is default behavior for IRT networks.

## 4.5 Information and Maintenance handling (I&M)

By default, PNSv4 firmware only supports I&M0 to I&M3 while I&M4 is only well defined in safety applications.

In contrast, PNSv3 firmware supported I&M0 to I&M4 by default.

It is possible to change the default behavior of PNSv4 firmware using the Set OEM Parameter service.

## 4.6 Structure of the PROFINET IO-Device stack

Figure 5 shows the internal structure of the layers which together represent the PROFINET IO Device stack V4.

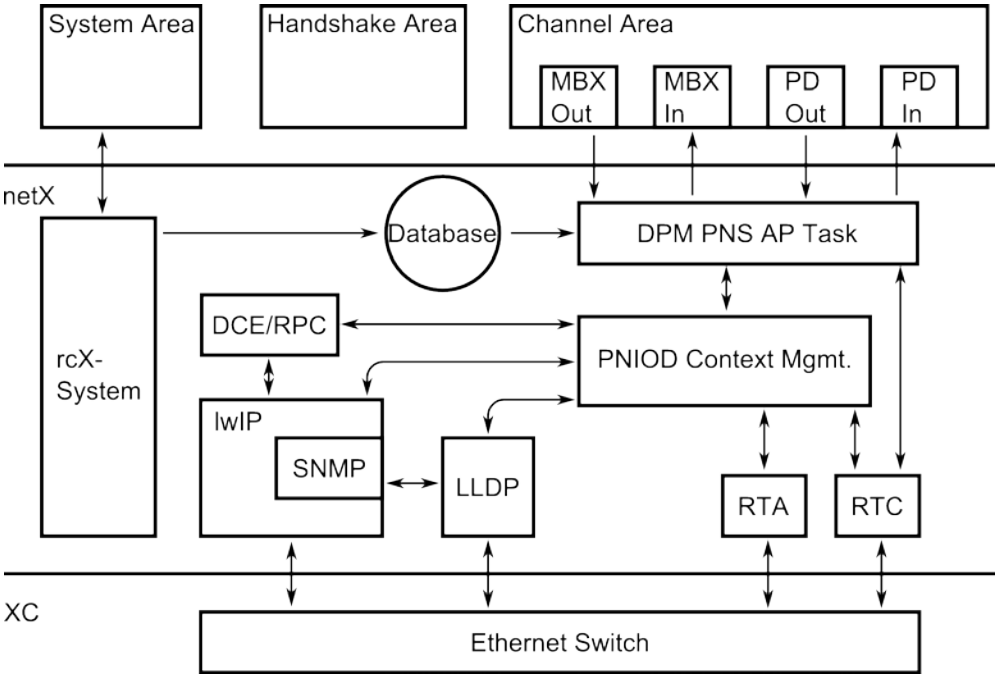



*Figure 5: Layer structure of PROFINET IO-Device stack V4*

## 4.7 TCP IP Packet API

The PNSv3 uses the TCP/IP stack implementation of Hilscher. For a description, see reference [4]. Using loadable firmware the API was available on DPM Channel 1 in the so-called “Ethernet Interface Channel”.

In contrast, the PNSv4 uses a different TCP/IP stack implementation with a completely different API. The API for TCP/IP services (so-called Socket API) available in PNSv4. For a description, see reference [8]. The availability and usability of the API depends on the netX chip used to execute PNSv4 firmware.

If used on netX52 the API is available directly in DPM Channel 0.

If used on netX51 or netX100/500 the API is available in DPM Channel 1 in so-called “Ethernet Interface Channel”.

## 4.8 RAW Ethernet API (NDIS)

The PROFINET IO Device stack V3.x offered an Ethernet API (NDIS) which needs to be activated in tag list of corresponding firmware. For a description of the API, see reference [15]. Using loadable firmware the API was available on DPM Channel 1 in so-called “Ethernet Interface Channel”.

PNSv4 offers the same Ethernet API (NDIS) which needs to be activated in the tag list of the corresponding firmware.

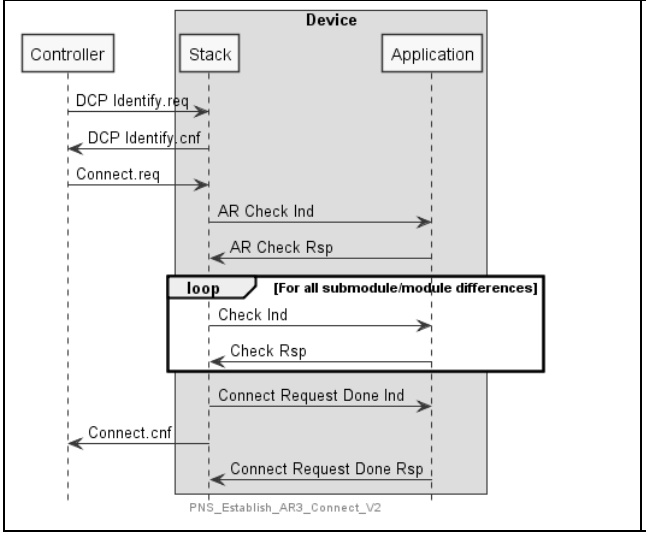
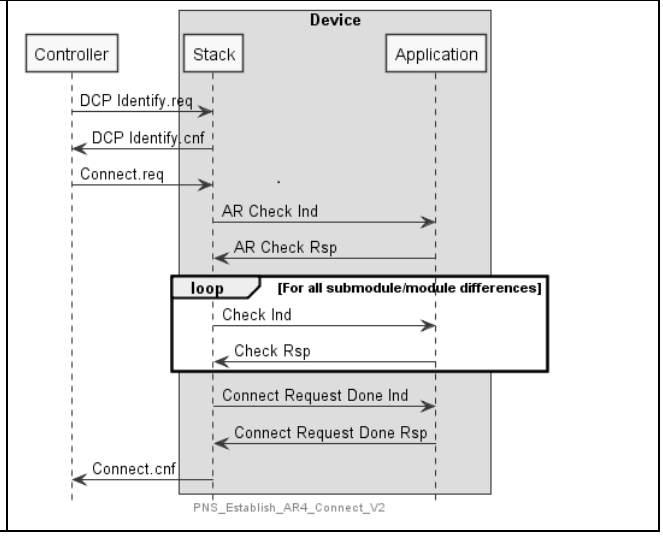
This functionality is not available for netX 52.

The OEM Mode has been removed, use Ethernet interface instead.

## 4.9 Behavior during connection establishment

The sequence of packets when a connection is established is identical to PNSv3 with the following behavior changes:

- Only one RPC request is processed by a stack at a time. That means that only one of the following services can be active at a time. The application must respond to the service before another service will be indicated.
  - AR Check Indication, Check Indication Service and Connect Request Done Service: As these requests are all related to RPC Connect, these services always occur as a whole for an AR, e.g. it can never happen that the application receives an AR Check Indication for another AR until the Connect Request Done Service has been responded
  - Read Record Service
  - Write Record Service
  - Parameter End Service
- The RPC Connect Response will be sent after Connect Request Done Service has been responded by the application. Thus the application might collect all Check Indications Data and adapt the Module/Submodule Configuration on Connect Request Done Indication. This is different to PNSv3 versions where the Connect Request Done Indication is sent after RPC Connect Response.

| PNSv3 | PNSv4 (and PNSv5) |
|---|---|
|  |  |

## 4.10 Linkable object NXLOM

Accessing the protocol stack is only possible with access via DPM of loadable firmware. Linkable Object is not support by PNSv4.

# 5 Changes of APIs and features between V4.x to V5.x

## 5.1 PROFINET Packet API

The packet API of PNSv5 is mostly identical to the packet API of PNSv4 except the differences described below. The behavior of PNSv5 is comparable to PNSv4.

This section may not be complete. Please check the Protocol API Manual for PNSv5 for details for the specific services (reference [9]).

Services that are marked as “no longer supported” for PNSv4 when migrating from PNSv3, are not supported by PNSv5 as well.

### 5.1.1 Configuring the IO-Device stack

#### Warmstart service and “old” Set Configuration service

The already legacy Warmstart service (command code 0x1F6A) and the “old” Set Configuration service (command code 0x1FC0) are no longer supported.
Use PNS_IF_SET_CONFIGURATION_REQ (0x1FE2) instead.

#### Set Port MAC Address service

This service is no longer supported. The port MAC addresses are either defined in the FDL (Flash Device Label) of the hardware or have to be provided to the system via newly defined Device Data Provider API function. For details, see reference [10].

#### Set OEM Parameter service

All parameter types can only be changed in unconfigured state. The parameter types 1, 2, 4, 9, 14, and 15 are not supported.

#### Load Remanent Data service

The remanent data is no longer handled by the protocol stack. Instead the remanent data is handled by a different, generic firmware part. Thus, the service to load remanent data is no longer PROFINET-stack-specific. Reference [9] describes the remanent data handling. Reference [10] describes the packet API.

### 5.1.2 Connection establishment

#### Store Remanent Data service

The remanent data is no longer handled by the protocol stack. Instead the remanent data is handled by a different, generic firmware part. Thus, the service to load remanent data is no longer PROFINET-stack-specific. Reference [9] describes the remanent data handling. Reference [10] describes the packet API.

### 5.1.3 Acyclic Events indicated by the stack

**Reset Factory Settings Indication service**

The service indicates receipt of a DCP Set Frame with command “Reset Factory Settings”. The service is generated by PNSv5 after the remanent data has been stored. In PNSv3 and PNSv4 the service was first generated and afterwards the remanent data has been stored.

**Store Remanent Data service**

In case the host application stores remanent data, PNSv5 uses a different service (by terms of packet command and fragmentation mechanism used) compared to PNSv4. Instead of PNS_IF_STORE_REMANENT_DATA_IND service the generic HIL_STORE_REMANENT_DATA_IND service is issued.

**Read I&M Data service**

PNSv5 does not support I&M5 dataset at all. Thus the corresponding indication is not generated.

### 5.1.4 Acyclic events requested by the application

**Process Alarm service**

This service is no longer supported. Use the Send User Alarm Service instead.

**Diagnosis Alarm service**

This service is no longer supported. The protocol stack will automatically initiate the corresponding alarms if needed.

**Return of Submodule Alarm service**

This service is no longer supported. Handling this specific alarm type will be performed by the stack automatically.

## 5.2 Limitations and dropped features

Features no longer supported by PNSv4 are not supported by PNSv5 as well. In addition PNSv5 does not support

- PROFIenegy ASE (*)
- Asset management (*)
- I&M5 (*)

Note (*): Planned for future release

## 5.3 Tag list

### 5.3.1 Protocol and firmware stack behavior

In contrast to PNSv3 and PNSv4 where behavior of remanent data (stored by firmware or stored by application) can be changed at runtime using the Set Configuration service, PNSv5 cannot change this at runtime.

For PNSv5, the decision needs to be made upfront and the configuration is done in tag list.

| **Corresponding service in PNSv4 firmware** | **Taglist parameter in PNSv5 firmware** | **Description** |
|---|---|---|
| “PNS_IF_SET_CONFIGURATION_REQ”.tData.tDeviceParameters.ulSystemFlags. PNS_IF_SYSTEM_DISABLE_STORE_REMANENT_ENABLED | “Remanent Data Responsibility”.”Remanent Data stored by Host” | If the host should store remanent data |

## 5.4 Information and Maintenance handling (I&M)

PNSv5 (V5.1.0.0) does not support I&M5 dataset at all. This feature is planned for future release.

## 5.5 PROFIenergy ASE

PNSv5 (V5.1.0.0) does not support PROFIenergy. This feature is planned for future release.

## 5.6 IRT

For netX 90, new PHY parameters have to be used. As a result,

- PNSv5 uses a new DAP and
- a drop-in replacement (for netX 51/52) is NOT possible.

In addition, the GSDML file has to be adapted.

## 5.7 Structure of the PROFINET IO-Device stack

Figure 6 shows the internal structure of the layers which together represent the PROFINET IO Device stack V5.

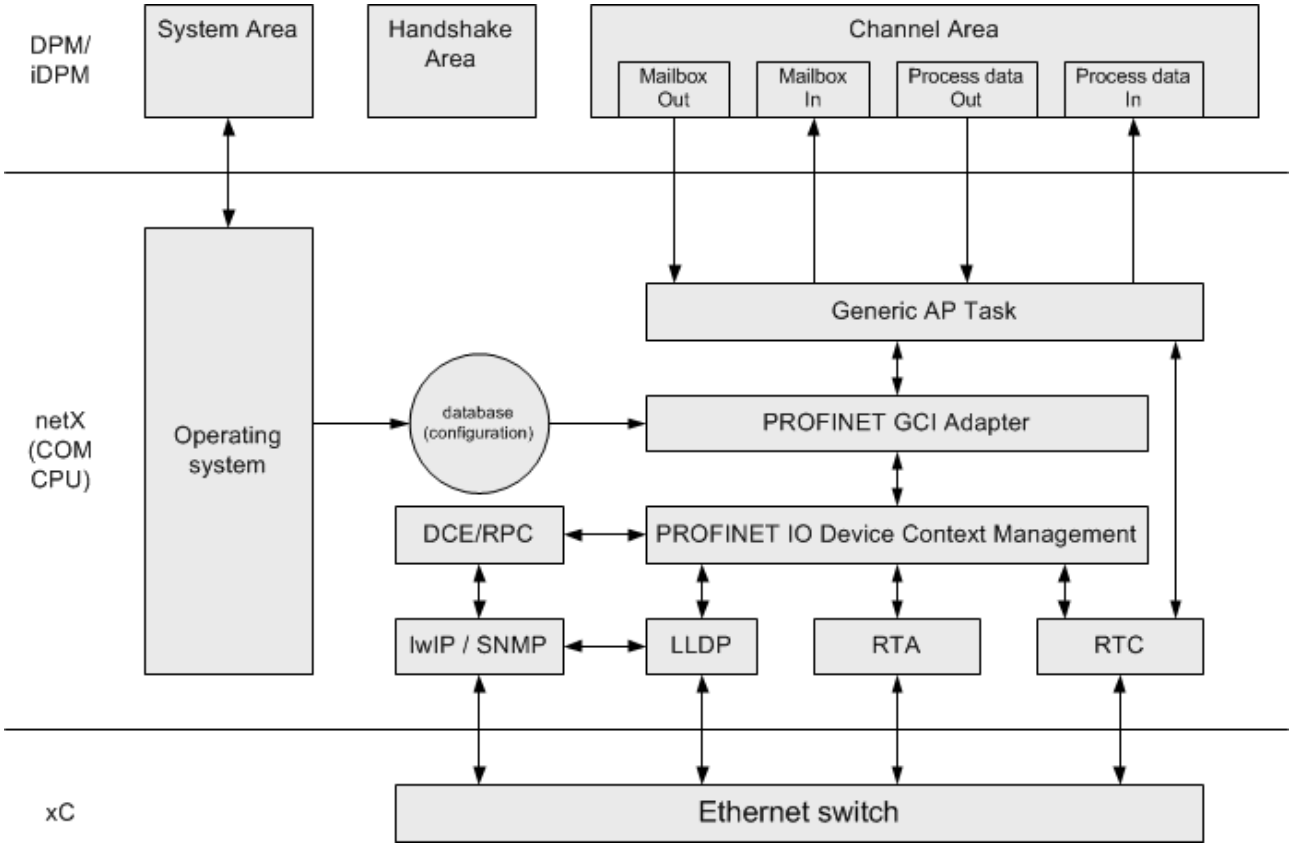



*Figure 6: Layer structure of PROFINET IO-Device stack V5*

## 5.8 TCP IP Packet API

PNSv4 and PNSv5 use the same TCP/IP stack and offer the same TCP/IP service API (so-called Socket API). For a description, see reference [12]. However, PNSv5 always offers this functionality in DPM Channel 1 while some PNSv4 firmware (netX52-based firmware) offer these services in DPM channel 0.

In PNSv4 the DPM Channel 1 identifies itself as so-called “Ethernet Interface Channel”. In PNSv5 the DPM Channel 1 identifies itself as so-called “Network Services Channel”.

# 6 Appendix

## 6.1 List of tables



## 6.2 List of figures

# 6.3 Contacts

### Headquarters

**Germany**
Hilscher Gesellschaft für
Systemautomation mbH
Rheinstrasse 15
65795 Hattersheim
Phone: +49 (0) 6190 9907-0
Fax: +49 (0) 6190 9907-50
E-Mail: info@hilscher.com

**Support**
Phone: +49 (0) 6190 9907-99
E-Mail: de.support@hilscher.com

### Subsidiaries

**China**
Hilscher Systemautomation (Shanghai) Co. Ltd.
200010 Shanghai
Phone: +86 (0) 21-6355-5161
E-Mail: info@hilscher.cn

**Support**
Phone: +86 (0) 21-6355-5161
E-Mail: cn.support@hilscher.com

**France**
Hilscher France S.a.r.l.
69500 Bron
Phone: +33 (0) 4 72 37 98 40
E-Mail: info@hilscher.fr

**Support**
Phone: +33 (0) 4 72 37 98 40
E-Mail: fr.support@hilscher.com

**India**
Hilscher India Pvt. Ltd.
Pune, Delhi, Mumbai
Phone: +91 8888 750 777
E-Mail: info@hilscher.in

**Italy**
Hilscher Italia S.r.l.
20090 Vimodrone (MI)
Phone: +39 02 25007068
E-Mail: info@hilscher.it

**Support**
Phone: +39 02 25007068
E-Mail: it.support@hilscher.com

**Japan**
Hilscher Japan KK
Tokyo, 160-0022
Phone: +81 (0) 3-5362-0521
E-Mail: info@hilscher.jp

**Support**
Phone: +81 (0) 3-5362-0521
E-Mail: jp.support@hilscher.com

**Korea**
Hilscher Korea Inc.
Seongnam, Gyeonggi, 463-400
Phone: +82 (0) 31-789-3715
E-Mail: info@hilscher.kr

**Switzerland**
Hilscher Swiss GmbH
4500 Solothurn
Phone: +41 (0) 32 623 6633
E-Mail: info@hilscher.ch

**Support**
Phone: +49 (0) 6190 9907-99
E-Mail: ch.support@hilscher.com

**USA**
Hilscher North America, Inc.
Lisle, IL 60532
Phone: +1 630-505-5301
E-Mail: info@hilscher.us

**Support**
Phone: +1 630-505-5301
E-Mail: us.support@hilscher.com