NOTICE

This document contains information about one or more ABB products and may include a description of or a reference to one or more standards that may be generally relevant to the ABB products. The presence of any such description of a standard or reference to a standard is not a representation that all of the ABB products referenced in this document support all of the features of the described or referenced standard. In order to determine the specific features supported by a particular ABB product, the reader should consult the product specifications for the particular ABB product.

ABB may have one or more patents or pending patent applications protecting the intellectual property in the ABB products described in this document.

The information in this document is subject to change without notice and should not be construed as a commitment by ABB. ABB assumes no responsibility for any errors that may appear in this document.

Products described or referenced in this document are designed to be connected, and to communicate information and data via a secure network. It is the sole responsibility of the system/product owner to provide and continuously ensure a secure connection between the product and the system network and/or any other networks that may be connected.

The system/product owners must establish and maintain appropriate measures, including, but not limited to, the installation of firewalls, application of authentication measures, encryption of data, installation of antivirus programs, and so on, to protect the system, its products and networks, against security breaches, unauthorized access, interference, intrusion, leakage, and/or theft of data or information.

ABB verifies the function of released products and updates. However system/product owners are ultimately responsible to ensure that any system update (including but not limited to code changes, configuration file changes, third-party software updates or patches, hardware change out, and so on) is compatible with the security measures implemented. The system/product owners must verify that the system and associated products function as expected in the environment they are deployed.

In no event shall ABB be liable for direct, indirect, special, incidental or consequential damages of any nature or kind arising from the use of this document, nor shall ABB be liable for incidental or consequential damages arising from use of any software or hardware described in this document.

This document and parts thereof must not be reproduced or copied without written permission from ABB, and the contents thereof must not be imparted to a third party nor used for any unauthorized purpose.

The software or hardware described in this document is furnished under a license and may be used, copied, or disclosed only in accordance with the terms of such license. This product meets the requirements specified in EMC Directive 2004/108/EC and in Low Voltage Directive 2006/95/EC.

TRADEMARKS

All rights to copyrights, registered trademarks, and trademarks reside with their respective owners.

Copyright © 2003-2016 by ABB.
All rights reserved.

Release: April 2016
Document number: 3BDS021515-600 A
# Table of Contents

## About This User Manual
- Intended User.................................................................................................................. 11
- How to Use this User Manual.......................................................................................... 12
- User Manual Conventions ............................................................................................... 12
  - Feature Pack ............................................................................................................... 13
  - Warning, Caution, Information, and Tip Icons .................................................... 13
- Terminology..................................................................................................................... 14
- Released User Manuals and Release Notes........................................................................ 17

## Section 1 - Introduction

## Section 2 - Functional Description
- PROFINET IO Basics........................................................................................................ 24
- PROFINET IO Device Model............................................................................................ 24
  - Modules and Submodules ....................................................................................... 25
  - Slots and Subslots .................................................................................................... 25
  - PROFINET IO and PROFIBUS .................................................................................. 27
- Naming Conventions........................................................................................................ 27
- Real-Time Communication .............................................................................................. 27
- Data Frame .................................................................................................................... 31
- PROFINET IO Data Exchange Quality .......................................................................... 33
- Multi Controller Access ................................................................................................. 34
- Redundancy .................................................................................................................... 35
  - Overview .................................................................................................................... 35
  - CI871 Redundancy .................................................................................................... 35
  - PROFINET IO Device Redundancy .......................................................................... 36
<table>
<thead>
<tr>
<th>Section Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ethernet Network Redundancy</td>
<td>36</td>
</tr>
<tr>
<td>Status Handling</td>
<td>36</td>
</tr>
<tr>
<td>Status Handling for CI871</td>
<td>37</td>
</tr>
<tr>
<td>Status Handling for PROFINET IO Devices</td>
<td>39</td>
</tr>
<tr>
<td>Alarms and Events</td>
<td>41</td>
</tr>
<tr>
<td>PNIO Diagnosis</td>
<td>42</td>
</tr>
<tr>
<td>Structure of a PNIO Alarm</td>
<td>45</td>
</tr>
<tr>
<td>Standard Status Alarm</td>
<td>45</td>
</tr>
<tr>
<td>Channel Diagnosis</td>
<td>45</td>
</tr>
<tr>
<td>Manufacturer Specific Diagnosis</td>
<td>46</td>
</tr>
<tr>
<td>Process Alarms</td>
<td>46</td>
</tr>
<tr>
<td>Sequence of Events (SOE)</td>
<td>47</td>
</tr>
<tr>
<td>Example of an SOE</td>
<td>49</td>
</tr>
<tr>
<td>Acyclic Data Communication</td>
<td>52</td>
</tr>
<tr>
<td>Configuring IOCommLib Function Block</td>
<td>56</td>
</tr>
<tr>
<td>Addressing in PROFINET IO</td>
<td>62</td>
</tr>
<tr>
<td>Hot Swap</td>
<td>62</td>
</tr>
</tbody>
</table>

**Section 3 - Configuration**

<table>
<thead>
<tr>
<th>Section Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hardware Library</td>
<td>63</td>
</tr>
<tr>
<td>Inserting CI871 PROFINET IO Controller Unit</td>
<td>63</td>
</tr>
<tr>
<td>Configuring CI871 PROFINET IO Controller Unit</td>
<td>64</td>
</tr>
<tr>
<td>Hardware Editor</td>
<td>64</td>
</tr>
<tr>
<td>Settings Tab</td>
<td>65</td>
</tr>
<tr>
<td>Connection Tab</td>
<td>68</td>
</tr>
<tr>
<td>Unit Status Tab</td>
<td>68</td>
</tr>
<tr>
<td>Inserting a PROFINET IO Device</td>
<td>69</td>
</tr>
<tr>
<td>Configuring PROFINET IO Device Unit</td>
<td>73</td>
</tr>
<tr>
<td>Configuration Options</td>
<td>73</td>
</tr>
<tr>
<td>Station Name</td>
<td>74</td>
</tr>
<tr>
<td>Basic Communication Settings</td>
<td>74</td>
</tr>
<tr>
<td>Cyclic Communication Options</td>
<td>76</td>
</tr>
<tr>
<td>Startup Parameters</td>
<td>77</td>
</tr>
</tbody>
</table>
# Table of Contents

- I/O Data ........................................................................................................... 78
- Complex Device Example MNS iS .............................................................. 79
- New Device Types ....................................................................................... 80

## Section 4 - Download and Online Mode

- Prerequisites .................................................................................................. 81
- Downloading Configuration Settings .......................................................... 81
  - Online Changes ....................................................................................... 81
  - Logfile .................................................................................................... 82
- Online Data .................................................................................................... 83
- Behavior of I/O and Communication ............................................................ 83
  - Insertion and Deletion of I/O Units ......................................................... 83
  - Connection Error .................................................................................... 83
  - System Error .......................................................................................... 84

## Section 5 - CI871 Web Server

- CI871 Web Server Login Prerequisite .......................................................... 85
  - Web Server Login .................................................................................. 86
  - Enable Javascript for Web Server ......................................................... 87
  - Reset Web Browser Security settings for Web Server ......................... 88
- CI871 Web Server Security .......................................................................... 89
- CI871 Web Server Interface ......................................................................... 92
  - CI871 Web Server Interface Menu Items ............................................. 93
    - Errormemory ....................................................................................... 94
    - Version Info ....................................................................................... 95
    - LifeList ............................................................................................... 96
    - Set Devicename ................................................................................. 97
    - I&M Data .......................................................................................... 99
    - Let Device blink ............................................................................... 100
    - Journal Buffer .................................................................................. 101
    - CPU Load ......................................................................................... 102
    - General Status .................................................................................. 103
- Service File .................................................................................................... 104
Table of Contents

Change Password .......................................................................................................... 105
Reset Default Password ............................................................................................. 106

Section 6 - Technical Data and Performance

Section 7 - Device Import Wizard
Importing GSD File....................................................................................................... 112
Create a New Hardware Library ................................................................................... 112
Start Wizard ................................................................................................................. 114
Configure Hardware Types ......................................................................................... 118
   Hardware Tree Context Menu .................................................................................. 118
   Hardware Types Selection ....................................................................................... 121
Input Tab ....................................................................................................................... 124
Byte Swapping .............................................................................................................. 127
Advanced ADV Settings with status .......................................................................... 131
   Range Parameter ..................................................................................................... 132
   Frame format ............................................................................................................. 135
To Modify I/O Channels ............................................................................................... 135
Deleting I/O Channel ................................................................................................. 138
Creating I/O Channels ............................................................................................... 139
Output Tab ..................................................................................................................... 142
Parameter Tab ............................................................................................................... 143
   Modifying Default Parameters .............................................................................. 143
Create Hardware Types ............................................................................................. 146
Append GSD files to a Hardware Library .................................................................... 149
Re-importing the GSD file ........................................................................................... 150

Section 8 - Controller/Controller Communication
PN/PN Coupler.............................................................................................................. 153
Data Transfer................................................................................................................ 153

Appendix A - CI871 Error Codes

Appendix B - CI871 TroubleShooting
Device Replacement ......................................................................................................161
Exchanging Device Positions ......................................................................................161
Exchanging IP Addresses of Devices .........................................................................162
Set Devicename for an Active Device ........................................................................162

Revision History
Updates in Revision Index A .......................................................................................163

Index
About This User Manual

Any security measures described in this User Manual, for example, for user access, password security, network security, firewalls, virus protection, etc., represent possible steps that a user of an 800xA System may want to consider based on a risk assessment for a particular application and installation. This risk assessment, as well as the proper implementation, configuration, installation, operation, administration, and maintenance of all relevant security related equipment, software, and procedures, are the responsibility of the user of the 800xA System.

This manual describes the PROFINET IO configuration in the 800xA control system using the communication interface CI871.

Some of the important topics described in this user manual are:

- PROFINET IO functionalities available with CI871.
- Hardware configuration with the Control Builder.
- Supervision and status visualization of the PROFINET IO.
- Commissioning and Diagnostics.

Intended User

This manual is intended for application engineers and design engineers who are planning the set up of a PROFINET IO system. The reader should be familiar with the hardware and software functionality of the 800xA system products. Added to this requirement, the user should have an adequate knowledge of PROFINET IO.
How to Use this User Manual

Section 1, Introduction gives an overview of PROFINET IO and how it is integrated with the controllers.

Section 2, Functional Description gives a detailed information on the PROFINET IO implementation.

Section 3, Configuration describes the PROFINET IO configuration in Control Builder.

Section 4, Download and Online Mode describes the download procedure and the system behavior in case of an error.

Section 5, CI871 Web Server describes how to get a detailed diagnostic information from the system in case of a serious PROFINET IO error and how to set the device name for PROFINET IO devices.

Section 6, Technical Data and Performance describes the guidelines for PROFINET IO configurations in 800xA with CI871.

Section 7, Device Import Wizard describes the procedure of parsing a General Station Description (GSD) file using a wizard and converting them into a hardware library.

Section 8, Controller/Controller Communication describes the communication between two controllers using PN/PN Coupler.

Appendix A, CI871 Error Codes describes the list of error codes used for acyclic communication.

Appendix B, CI871 TroubleShooting describes the workaround for PNIO devices.

For a list of documents related to the products described in this user manual, refer to Released User Manuals and Release Notes on page 17.

User Manual Conventions

Microsoft Windows conventions are normally used for the standard presentation of material when entering text, key sequences, prompts, messages, menu items, screen elements, etc.
Feature Pack

The Feature Pack content (including text, tables, and figures) included in this User Manual is distinguished from the existing content using the following two separators:

Feature Pack Functionality

<Feature Pack Content>

Feature Pack functionality included in an existing table is indicated using a table footnote (*):

*Feature Pack Functionality

Unless noted, all other information in this User Manual applies to 800xA Systems with or without a Feature Pack installed.

Warning, Caution, Information, and Tip Icons

This User Manual includes Warning, Caution, and Information where appropriate to point out safety related or other important information. It also includes Tip to point useful hints to the reader. The corresponding symbols should be interpreted as follows:

- Electrical warning icon indicates the presence of a hazard that could result in electrical shock.
- Warning icon indicates the presence of a hazard that could result in personal injury.
- Caution icon indicates important information or warning related to the concept discussed in the text. It might indicate the presence of a hazard that could result in corruption of software or damage to equipment/property.
- Information icon alerts the user to pertinent facts and conditions.
Tip icon indicates advice on, for example, how to design the project or how to use a certain function.

Although Warning hazards are related to personal injury, and Caution hazards are associated with equipment or property damage, it should be understood that operation of damaged equipment could, under certain operational conditions, result in degraded process performance leading to personal injury or death. Therefore, fully comply with all Warning and Caution notices.

## Terminology

A complete and comprehensive list of Terms is included in the *System 800xA, Engineering Concepts instruction* (*3BDS100972*). The listing includes terms and definitions that apply to the 800xA System where the usage is different from commonly accepted industry standard definitions and definitions given in standard dictionaries such as Webster’s Dictionary of Computer Terms. Terms that uniquely apply to this User Manual are listed in the following table.

<table>
<thead>
<tr>
<th>Term/Acronym</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AC 800M</td>
<td>ABB Controller 800M series, general purpose process controller series by ABB.</td>
</tr>
<tr>
<td>AC 800M Controller</td>
<td>Any controller constructed from the units and units connected to the AC 800M hardware platform.</td>
</tr>
<tr>
<td>CBA</td>
<td>Component Based Automation</td>
</tr>
<tr>
<td>Control Builder M</td>
<td>The programming tool for AC 800M. Control Builder Professional is integrated into System 800xA.</td>
</tr>
<tr>
<td>CEX-Bus</td>
<td>Communication Expansion Bus (for communication units).</td>
</tr>
<tr>
<td>Connector</td>
<td>A Connector is a coupling device used to connect the wire medium to a fieldbus device or to another wire segment.</td>
</tr>
<tr>
<td>DAP</td>
<td>Device Access Point</td>
</tr>
<tr>
<td>DCP</td>
<td>Discovery and Configuration Protocol</td>
</tr>
<tr>
<td>Term/Acronym</td>
<td>Description</td>
</tr>
<tr>
<td>-------------</td>
<td>-------------</td>
</tr>
<tr>
<td>Ethernet</td>
<td>Protected trademark of Xerox (since 1975).</td>
</tr>
<tr>
<td>Fieldbus</td>
<td>A Fieldbus is used to interconnect field devices, such as I/O modules, smart sensors, actuators, variable speed drives, PLCs, or small single loop devices, and to connect these devices to the 800xA system.</td>
</tr>
<tr>
<td>GSD File</td>
<td>General Station Description device communication database file for PROFINET IO devices.</td>
</tr>
<tr>
<td>GSDML</td>
<td>GSDML is the XML based language to describe the characteristics of PROFINET IO devices e.g. communication and module parameters.</td>
</tr>
<tr>
<td>Hot Removal</td>
<td>Units with hot removal support can be removed online, without any disturbance to other units connected to the CEX-Bus. This indicates that the unit can be removed online, if it becomes faulty.</td>
</tr>
<tr>
<td>Hot Swap</td>
<td>Units with hot swap (includes hot removal) support, can be replaced online, without any disturbance to other units connected to the CEX-Bus. In a redundant system, the backup unit can be replaced without any disturbances to the primary unit. This indicates that the unit can be replaced online, if it becomes faulty.</td>
</tr>
<tr>
<td>HWD File</td>
<td>Hardware Definition file is an ASCII readable file that describes the hardware unit. It is used by Control Builder.</td>
</tr>
<tr>
<td>Hardware Library</td>
<td>Library containing the hardware definition files.</td>
</tr>
<tr>
<td>IEC</td>
<td>International Electrotechnical Commission.</td>
</tr>
<tr>
<td>I&amp;M</td>
<td>Identification &amp; Maintenance Functions</td>
</tr>
<tr>
<td>IP</td>
<td>Internet Protocol</td>
</tr>
<tr>
<td>IRT</td>
<td>Isochronous Real-Time</td>
</tr>
</tbody>
</table>
### Terminology

<table>
<thead>
<tr>
<th>Term/Acronym</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ISP</td>
<td>Input Set as Predetermined. When the controller detects a communication failure with an input module, the application variables are set to predetermined values specified by ISP control.</td>
</tr>
<tr>
<td>LSB</td>
<td>Least Significant Byte</td>
</tr>
<tr>
<td>MAC Address</td>
<td>Media Access Control Address (Ethernet Address)</td>
</tr>
<tr>
<td>MNS /S</td>
<td>Motor Control Center</td>
</tr>
<tr>
<td>MSB</td>
<td>Most Significant Byte</td>
</tr>
<tr>
<td>Node</td>
<td>A computer that communicates with the network, for example the Internet, Plant, Control or I/O network. Each node typically has a unique node address with a format depending on the network it is connected to.</td>
</tr>
<tr>
<td>OSP</td>
<td>Output Set as Predetermined. When an I/O module locally detects communication failure with the controller, it automatically sets its output to the values specified by OSP control.</td>
</tr>
<tr>
<td>PDU</td>
<td>Processing Data Unit. A data packet passed across a network through telegrams.</td>
</tr>
<tr>
<td>PNIO</td>
<td>PROFINET IO</td>
</tr>
<tr>
<td>PN/PN Coupler</td>
<td>Transmits data between two PNIO Controllers.</td>
</tr>
<tr>
<td>PROFIBUS</td>
<td><strong>PROcess Fiel</strong> <strong>dBUS.</strong> PROFIBUS is a manufacturer-independent fieldbus standard for applications in manufacturing, process and building automation. The PROFIBUS family is composed of three types of protocol, each of which is used for different tasks. The three types of protocols are: PROFIBUS FMS, DP, and PA.</td>
</tr>
<tr>
<td>PROFIBUS DP</td>
<td>PROFIBUS DP is the communication protocol for Decentralized Peripherals. DP has the following versions: DP-V0, DP-V1, and DP-V2.</td>
</tr>
<tr>
<td>PROFIBUS PA</td>
<td>PROFIBUS for Process Automation</td>
</tr>
</tbody>
</table>
About This User Manual

Released User Manuals and Release Notes

A complete list of all User Manuals and Release Notes applicable to System 800xA is provided in System 800xA Released User Manuals and Release Notes (3BUA000263*).

<table>
<thead>
<tr>
<th>Term/Acronym</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PROFIBUS International (PI)</td>
<td>The international umbrella organization for PROFIBUS founded in 1995.</td>
</tr>
<tr>
<td>PROFIBUS User Organization e.V. (PNO)</td>
<td>The PNO is the trade body of manufacturers and users for PROFIBUS founded in 1989.</td>
</tr>
<tr>
<td>PROFINET</td>
<td>PROFINET is the Ethernet-based automation standard of PROFIBUS International.</td>
</tr>
<tr>
<td>PROFINET CBA</td>
<td>PROFINET standard for distributed automation system on an automation component basis.</td>
</tr>
<tr>
<td>PROFINET IO</td>
<td>PROFINET standard for simple distributed I/O and time-critical applications.</td>
</tr>
<tr>
<td>Redundancy</td>
<td>The existence of more than one capability of an item (system, equipment, or component) to perform its intended function.</td>
</tr>
<tr>
<td>Remote I/O</td>
<td>Input/Output units connected to a controller by a fieldbus.</td>
</tr>
<tr>
<td>RT</td>
<td>Real Time</td>
</tr>
<tr>
<td>TCP</td>
<td>Transmission Control Protocol/Internet Protocol</td>
</tr>
<tr>
<td>UDP</td>
<td>User Datagram Protocol</td>
</tr>
<tr>
<td>Unit</td>
<td>A hardware unit, with or without accommodated software.</td>
</tr>
<tr>
<td>USI</td>
<td>User Structure Identifier</td>
</tr>
<tr>
<td>XML</td>
<td>Extensible Markup Language</td>
</tr>
</tbody>
</table>
System 800xA Released User Manuals and Release Notes (3BUA000263*) is updated each time a document is updated or a new document is released. It is in pdf format and is provided in the following ways:

- Included on the documentation media provided with the system and published to ABB SolutionsBank when released as part of a major or minor release, Service Pack, Feature Pack, or System Revision.

- Published to ABB SolutionsBank when a User Manual or Release Note is updated in between any of the release cycles listed in the first bullet.

A product bulletin is published each time System 800xA Released User Manuals and Release Notes (3BUA000263*) is updated and published to ABB SolutionsBank.

For standards and commercially available PROFINET documentation, visit the PROFINET Web Site (http://www.profinet.com).
PROFINET is an open Fieldbus standard for applications in manufacturing and process automation. PROFINET technology is an international standard that is part of IEC 61158 and IEC 61784.

The two perspectives of PROFINET are:

- PROFINET IO, which is used to integrate simple distributed I/O and time-critical applications into Ethernet communication.
- PROFINET CBA, which is used to integrate distributed automation system into Ethernet communication.

The PROFINET integration into ABB System 800xA focuses on the I/O connectivity. Therefore, only the PROFINET IO technology is used for the integration.

PROFINET IO is based on IEEE 802.3. It supports a transmission speed of 100 Mbps with auto negotiation and auto crossover in a switched Ethernet network. PROFINET IO uses Ethernet as well as TCP, UDP, and IP as the basis for communications. It is designed to work with other IP-based protocols on the same network.

Communication in PROFINET IO has different levels of performance:

- The transmission of non time-critical parameters and configuration data occurs in the standard channel of PROFINET IO based on TCP/IP or UDP.
- The transmission of time-critical process data within the production facility, occurs in the Real Time (RT) channel, also described as soft real-time.
For challenging tasks, the hardware based communication channel Isochronous Real-Time (IRT) is defined. For example, IRT can be used in motion control applications and high performance applications in factory automation.

The PROFINET IO implementation in System 800xA supports only RT channel. There is no support for IRT.

When distributed I/O applications are connected for communication through PROFINET IO, the familiar I/O view of PROFIBUS is retained. The peripheral data from the field devices are periodically transmitted into the process model of the control system.

PROFINET IO describes a device model oriented to the PROFIBUS framework, which consists of places of insertion (slots) and groups of I/O channels (subslots). The technical characteristics of the field devices are described by the General Station Description (GSD) file, which is based on XML. The PROFINET IO engineering is performed in a way familiar to PROFIBUS. The distributed field devices are assigned to the controllers during configuration.

The PROFINET IO is interfaced to the AC 800M controller (under System 800xA) using the PROFINET IO module CI871.

Figure 1 shows a possible PROFINET IO installation with AC 800M controller.
PROFINET IO is configured using the Control Builder available under System 800xA. The configuration includes the planning of the hardware units in the hardware tree, specific configurations for the PROFINET IO communication interface CI871 and the PROFINET IO devices. The device specific configuration data is described within the GSD file provided by the device manufacturer.

To configure the PROFINET IO device within the Control Builder, the GSD file must be imported into a hardware library and inserted to the project using the Device Import Wizard.
Section 2  Functional Description

This section provides the functional description of the PROFINET IO integration with System 800xA.

This section contains:

- PROFINET IO Basics on page 24
- Multi Controller Access on page 34
- Redundancy on page 35
- Status Handling on page 36
- PNIO Diagnosis on page 42
- Sequence of Events (SOE) on page 47
- Acyclic Data Communication on page 52
- Addressing in PROFINET IO on page 62
- Hot Swap on page 62
PROFINET IO Basics

PROFINET IO (PNIO) integrates distributed I/O and field devices into the Ethernet Communication. The existing Fieldbus systems can be integrated into PNIO applications using an IO-proxy.

PROFINET IO Device Model

A PROFINET IO device model consists of:

- **Device**
  Represents the device by its name and IP address.

- **Device Access Point (DAP)**
  Represents the Fieldbus Communication Interface. It provides configuration options for cyclic communication.

- **I/O Modules**
  Carrying submodules 1...n.

- **Submodules**
  Carrying I/O data and startup parameters.

Figure 2 shows the PROFINET IO device model.

![Figure 2. PROFINET IO Device Model](image-url)
**Modules and Submodules**

A device can have one or more modules. Each module can have one or more submodules.

The modules do not carry any data like cyclic I/O data or parameter data. The submodules carry cyclic I/O data, parameter data and alarms/events. The submodules are used to functionally group the I/O data belonging to one module. This is useful in configuring complex devices. For example, for MNS iS, each motor starter is handled as a separate module. Each module can have several submodules. Each submodule provides a well defined subset of I/O data for the motor starter. In this way, all instances of motor starters can be configured individually (see Figure 30 on page 80), as an example for the configuration.

**Slots and Subslots**

The address position of a module is called a **slot**, and the address position of a submodule is called a **subslot**.

A **slot** is the physical place of insertion of a peripheral assembly (module) in a device. Within a slot, the subslots form the actual interface to the process (inputs/outputs). The data content of a subslot is always accompanied by status information, from which the validity of the data is derived. The index specifies the data within a slot/subslot that can be read or written acyclically through read/write services. For example, parameters can be written to a module or module data can be read out on the basis of an index.

The DAP is required to be placed in slot 0. Similar to the I/O modules a DAP can have one or more submodules.

**Figure 3** shows a representation the PROFINET IO device model in the hardware tree.
Figure 3. PROFINET IO Device Model, hardware tree presentation
PROFINET IO and PROFIBUS

The PROFINET IO device model is similar to the PROFIBUS framework. The IO data is accessed by configuring the device with its modules and submodules in the hardware tree. In PROFIBUS, the IO data is available on module level, whereas with PROFINET IO, the IO data is available on submodule level.

From an application level, PROFIBUS and PROFINET IO are compatible.

Naming Conventions

The names of the hardware-types and I/O channels are described within the GSD file of the device. However, if the information needed by 800xA is missing in GSD file, this must be added. This is automatically done by the Device Import Wizard during the device import.

Typically, the names of DAP and submodules are to be added. The following naming conventions are used by the Device Import Wizard:

- The name of the DAP gets the name of the device with the suffix _DAP.
- The name of the submodules gets the name of the module with the suffix _SUB (see Figure 25 on page 73).

An example for a device with specific names for the DAP and the submodules defined in the GSD file is MNS iS. For this device the specific names are used as shown in Figure 30 on page 80.

Real-Time Communication

PROFINET IO uses optimized communication channels for real-time communication based on Ethernet (layer 2) without TCP/UDP or IP information. The transmission of data in the network is also optimized in PROFINET IO.

To achieve an optimal result, the packets in PROFINET IO are prioritized according to IEEE 802.1Q through VLAN tagging. The priority for real-time data in PROFINET IO is six.

The network components use this priority to control the data flow between the devices. Figure 4 shows the PROFINET IO communication protocols in the ISO/OSI reference models.
The communication between the AC 800M controller and the PROFINET IO device is based on the following criteria:

- A logical connection needs to be established to set up an active communication for PROFINET IO data between the PNIO controller and the PNIO device. This connection between the PNIO controller and the PNIO device is called Application Relation (AR). AR is set up by the Context Management through UDP/IP and RPC. This process facilitates the establishment of Communication Relation (CR). Several ARs containing more than one CR can also be established for the data transmission. ARs are established and cleared depending on their corresponding CRs.
There are three different types of CRs:

- **Record Data CR** - These are first established between the controller and the device. They carry parameter settings for the subslots.

- **I/O Data CR** - Transmits I/O Data. Device configuration holds the number of CRs to be established. They carry sub-modules status information.

- **Alarm CR** - Transmits alarms to the controller. These are acyclic data which require acknowledgement within the stipulated time. The Controller defines the priority for the alarms based on which they are transmitted.

- A data frame contains a maximum of 1440 bytes of cyclic data handled by the I/O Communication Relation (IOCR).

The performance levels defined for the PROFINET IO real-time communication are:

- The transmission of non time-critical parameters and configuration data occurs in the standard channel of PROFINET IO based on TCP/IP or UDP.

- The transmission of time-critical process data within the production facility, occurs in the Real-Time (RT) channel.

**Figure 5** shows the PROFINET IO communication model.

![PROFINET IO Communication Model](image-url)
Once the system is setup, cyclic exchange of process signals and high priority alarms is carried out by the Real-Time Channel. I/O data are transferred cyclically between the PNIO Controller and the PNIO Device. An AR must contain at least one Input CR and one Output CR.

The limited size of 1440 bytes for the IOCR is due to the maximum size of an Ethernet frame. In order to overcome this limitation, PROFINET IO is defined to support more than one IOCR for each direction. The number of supported IOCRs for each direction depends on the device capability.

CI871 supports only one IOCR for each direction with a maximum size of 1440 bytes for input and 1440 bytes for output data.

For the cyclic communication of I/O data, each IOCR (each PNIO device with CI871) can have a separate update time configured that is different for input and output direction. Hence, cyclic communication is updated based on the requirements of the device or application as shown in Figure 6. The update times can be configured in the range of 1 ms up to 512 ms.

![Figure 6. Cyclic Communication with PNIO](image)
For information on configuring the settings for cyclic communication, refer to **Cyclic Communication Options** on page 76.

The maximum size of I/O data length for inputs and outputs and the supported update times depend on the device capabilities.

The cyclic communication is monitored by a watchdog timer for each IOCR on both the units – the PNIO controller and the PNIO device. The watchdog timer can be configured on CI871. For information on configuring the watchdog timer, refer to **Configuring CI871 PROFINET IO Controller Unit** on page 64. If the timeout occurs for the watchdog timer, the system activates fail safe state. For more information on communication errors, refer to **Connection Error** on page 83.

### Data Frame

PROFINET IO uses the Ethernet Version II Frame Format for the Real Time communication.

**Figure 7** describes the PROFINET IO communication using an Ethernet Frame.

**Figure 7. Structure of Ethernet Frame for PROFINET IO Communication**

**Table 1** describes the components in the Ethernet data frame
Table 1. Description of Ethernet data frame components

<table>
<thead>
<tr>
<th>Component</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Preamble</td>
<td>Seven bytes comprising an alternating sequence of 1 or 0 for synchronization of the receiver.</td>
</tr>
<tr>
<td>SFD</td>
<td>Start of frame delimiter (10101011)</td>
</tr>
<tr>
<td></td>
<td>The double 1 at the end of the byte identifies the beginning of the destination address of the data packet.</td>
</tr>
<tr>
<td>MAC</td>
<td>Ethernet Source and Destination Address.</td>
</tr>
<tr>
<td>EtherType</td>
<td>Type ID of the packet (0x8100).</td>
</tr>
<tr>
<td>VLAN</td>
<td>Transmission of data with priority.</td>
</tr>
<tr>
<td></td>
<td>The PROFINET IO frame gets the highest priority among other protocol frames.</td>
</tr>
<tr>
<td>EtherType</td>
<td>Type identification for the network protocol following in the data component: 0x8892: PROFINET IO</td>
</tr>
<tr>
<td>Frame ID</td>
<td>Identification of FrameType in PNIO</td>
</tr>
<tr>
<td>Data</td>
<td>PNIO: I/O data. The maximum size of I/O data in PROFINET IO is 1440 Bytes and minimum is 40 as this a prerequisite for Ethernet frame. In case the I/O of the PNIO device is less than 40 then padding byte will be added automatically.</td>
</tr>
<tr>
<td>Cycle Counter</td>
<td>The counter is incremented by the provider with each send clock. A consumer can detect overtaking processes using the counter value.</td>
</tr>
<tr>
<td>Data Status</td>
<td>Indicates whether the data is valid (Bit 1) or invalid (Bit 0).</td>
</tr>
<tr>
<td>Transfer Status</td>
<td>Bit 0-7: 0 Reserved</td>
</tr>
<tr>
<td>FCS Frame Check</td>
<td>32 Bit checksum. Cyclic redundancy check (CRC) for the complete ethernet frame.</td>
</tr>
<tr>
<td>Sequence</td>
<td></td>
</tr>
</tbody>
</table>
PROFINET IO Data Exchange Quality

PROFINET IO is based on the producer/consumer model. The I/O data that is exchanged between CI871 and the devices is checked for quality.

The data is produced by one component and consumed by another component:

- For input data, the PNIO device is the producer and PNIO controller (PLC) is the consumer.
- For output data, PNIO controller (PLC) is the producer and PNIO device is the consumer.

The producer or consumer definition is related to the submodule level where the PNIO data is defined. The data contains the following two attributes (as shown in Figure 8):

- IOPS - IO Provider Status (status of the data provider).
- IOCS - IO Consumer Status (status of the data consumer).

A submodule is a producer or a consumer of PNIO data. The IOPS or IOCS status indicates the quality of the PNIO data belonging to the same submodule.

IOPS and IOCS are sent with each IOCR in opposite directions. While the IOPS indicates the status of the produced data, IOCS indicates if the consumer is in a position to operate the data. IOPS is used by the input channels to indicate a channel error (in case the IOPS value is BAD).

![Figure 8. Attributes of PNIO Data for submodule](image)
**Multi Controller Access**

In Multi Controller Access (Shared Device), several controllers access the same device. PROFINET IO supports Multi Controller Access.

An example of Multi Controller Access device is a Motor Control Center like MNS iS. In MNS iS, 60 motor starters are modeled as one PNIO device through the Device Access Point MLink, as shown in Figure 9.

PNIO controllers establishes individual ARs to access the same device. The number of ARs that can be established to a PNIO device depends on the device capability.

Multiple controllers access the same device after the establishment of ARs, but only one PNIO controller is allowed to write data while the other controllers have only read access.

*Figure 9. MNS iS as Multi Controller Access device*
In Multi Controller Access using 800xA, each controller in 800xA should be configured with only those modules to which the controller requires communication. Parallel access to the same submodule should not be done as it results in access violation errors during configuration. If the submodules are locked by other controller/supervisor, the access violation errors are displayed as warning messages.

In case of MNS iS, the access limitation setup is done at the motor starter level. Each motor starter is configured to only one controller and in this way multiple motor starters get access to multiple controllers. In order to setup a cyclic communication, the DAPs of multiple controllers should be setup with the same Basic communication settings.

The configuration of the modules and submodules on each controller can be different.

### Redundancy

#### Overview

In general, the following levels of redundancy are defined:

- Redundancy of CI871.
- Redundancy of PROFINET IO device.
- Redundancy of Ethernet network.

CI871 supports only application redundancy. There is no system integrated functionality available to enable redundancy for CI871 or the PNIO devices.

#### CI871 Redundancy

The system integrated redundancy functionality for CI871 is not supported. If the CI871 needs to be used in a redundant mode, an application redundancy needs to be configured. This requires a second PROFINET IO configuration with active communication which includes a second CI871 and a PNIO device supporting this topology.

A voter functionality within the IEC 61131-3 application decides which of the two communication paths is active. The application must be configured to manage the error handling in case of communication interruption.
PROFINET IO Device Redundancy

There is no system integrated redundancy functionality for PROFINET IO devices. If the PNIO device needs to be used in a redundant mode, an application redundancy needs to be configured. For information on application redundancy configuration, refer to CI871 Redundancy on page 35.

Ethernet Network Redundancy

The Ethernet network redundancy is the ability of the network to survive a single cable failure in its switch-to-switch links. The network survives by providing an alternate data path when a cable fault occurs.

PROFINET IO is used to setup highly available networks through a ring topology. The ring topology protects the network against line breaks in a system or a failure of a network component. The structure of a PROFINET IO system is shown in Figure 1 on page 21.

The PROFINET IO cyclic communication settings on device level need to be adjusted when a ring topology for the Ethernet is used. The Watchdog factor of all the connected devices must be adjusted, so that the cyclic PROFINET IO communication withstands errors on the network. This is achieved by changing the Red. Ethernet recovery time on CI871. During an application download, the Watchdog factor is increased by a value as set for Red. Ethernet recovery time, for all connected devices. For more information, refer to Table 7: Configuration Settings for CI871.

Status Handling

The error and status information for the hardware and software of CI871 and PROFINET IO device is indicated by the corresponding unit status in the Control Builder in Online mode. Alarms/events are also generated based on the unit status. For more information on alarms and events, refer to Alarms and Events on page 41. Additionally, the unit status is accessible through the IEC 61131-3 application.
## Status Handling for CI871

Table 2 lists the CI871 specific unit status bits in ErrorsAndWarnings.

### Table 2. PROFINET IO ErrorsAndWarnings for CI871

<table>
<thead>
<tr>
<th>Bit</th>
<th>Status bit</th>
<th>Value</th>
<th>Status Text</th>
<th>Status Type</th>
<th>Alarm/Event</th>
<th>Severity</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>ConnectionDown</td>
<td>16#00000001</td>
<td>Connection down</td>
<td>Error</td>
<td>Alarm</td>
<td>High</td>
<td>Not supported</td>
</tr>
<tr>
<td>1</td>
<td>IoError</td>
<td>16#00000002</td>
<td>I/O error</td>
<td>Error</td>
<td>Alarm</td>
<td>Medium</td>
<td>Not supported</td>
</tr>
<tr>
<td>2</td>
<td>ModuleMissing</td>
<td>16#00000004</td>
<td>Module missing</td>
<td>Error</td>
<td>Alarm</td>
<td>High</td>
<td>Not supported</td>
</tr>
<tr>
<td>3</td>
<td>WrongModuleType</td>
<td>16#00000008</td>
<td>Wrong module type</td>
<td>Error</td>
<td>Alarm</td>
<td>High</td>
<td>Not supported</td>
</tr>
<tr>
<td>4</td>
<td>StatusChannelError</td>
<td>16#00000100</td>
<td>Channel error</td>
<td>Warning</td>
<td>Alarm</td>
<td>Medium</td>
<td>Not supported</td>
</tr>
<tr>
<td>5</td>
<td>IoWarning</td>
<td>16#00000200</td>
<td>I/O warning</td>
<td>Warning</td>
<td>Event</td>
<td>Low</td>
<td>Not supported</td>
</tr>
<tr>
<td>6</td>
<td>StatusUnderflow</td>
<td>16#00000040</td>
<td>Underflow</td>
<td>Warning</td>
<td>Alarm</td>
<td>Low</td>
<td>Not supported</td>
</tr>
<tr>
<td>7</td>
<td>StatusOverflow</td>
<td>16#00000080</td>
<td>Overflow</td>
<td>Warning</td>
<td>Alarm</td>
<td>Low</td>
<td>Not supported</td>
</tr>
<tr>
<td>8</td>
<td>StatusForced</td>
<td>16#00000100</td>
<td>Forced</td>
<td>Warning</td>
<td>Event</td>
<td>Low</td>
<td>Not supported</td>
</tr>
<tr>
<td>9</td>
<td>WatchdogTimeout</td>
<td>16#00000200</td>
<td>Watchdog timeout</td>
<td>Error</td>
<td>Alarm</td>
<td>High</td>
<td>Watchdog timeout</td>
</tr>
<tr>
<td>10</td>
<td>DeviceFailure</td>
<td>16#00000400</td>
<td>Device failure</td>
<td>Error</td>
<td>Alarm</td>
<td>High</td>
<td>Device failure</td>
</tr>
<tr>
<td>11</td>
<td>DeviceNotFound</td>
<td>16#00000800</td>
<td>Device not found</td>
<td>Error</td>
<td>Alarm</td>
<td>High</td>
<td>Device not found</td>
</tr>
<tr>
<td>12</td>
<td>WrongDeviceType</td>
<td>16#00001000</td>
<td>Wrong device type</td>
<td>Error</td>
<td>Alarm</td>
<td>High</td>
<td>Wrong device type</td>
</tr>
<tr>
<td>13</td>
<td>IOConnectError</td>
<td>16#00002000</td>
<td>I/O connection error</td>
<td>Error</td>
<td>Alarm</td>
<td>Medium</td>
<td>Not supported</td>
</tr>
<tr>
<td>14</td>
<td>IOConfigError</td>
<td>16#00004000</td>
<td>I/O configuration error</td>
<td>Error</td>
<td>Alarm</td>
<td>Medium</td>
<td>Not supported</td>
</tr>
<tr>
<td>15</td>
<td>HWConfigError</td>
<td>16#00008000</td>
<td>Hardware configuration error</td>
<td>Error</td>
<td>Alarm</td>
<td>High</td>
<td>Hardware configuration error</td>
</tr>
<tr>
<td>16</td>
<td>GeneralError</td>
<td>16#00100000</td>
<td>-</td>
<td>Error</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>17</td>
<td>GeneralWarning</td>
<td>16#00200000</td>
<td>-</td>
<td>Warning</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>18</td>
<td>RedWarningPrimary</td>
<td>16#00400000</td>
<td>Warning on primary unit</td>
<td>Warning</td>
<td>Event</td>
<td>Low</td>
<td>Not supported</td>
</tr>
<tr>
<td>19</td>
<td>RedWarningBackup</td>
<td>16#00800000</td>
<td>Warning on backup unit</td>
<td>Warning</td>
<td>Event</td>
<td>Low</td>
<td>Not supported</td>
</tr>
</tbody>
</table>
### Table 2. PROFINET IO ErrorsAndWarnings for CI871 (Continued)

<table>
<thead>
<tr>
<th>Bit</th>
<th>Status bit</th>
<th>Value</th>
<th>Status Text</th>
<th>Status Type</th>
<th>Alarm/Event</th>
<th>Severity</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>20</td>
<td>RedErrorBackup</td>
<td>16#00100000</td>
<td>Error on backup unit</td>
<td>Warning</td>
<td>Alarm</td>
<td>Medium</td>
<td>Not supported</td>
</tr>
<tr>
<td>21</td>
<td>Reserved</td>
<td>16#00200000</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Reserved</td>
</tr>
<tr>
<td>22</td>
<td>DeviceSpecific10</td>
<td>16#00400000</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Reserved</td>
</tr>
<tr>
<td>23</td>
<td>DeviceSpecific9</td>
<td>16#00800000</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Reserved</td>
</tr>
<tr>
<td>24</td>
<td>DeviceSpecific8</td>
<td>16#01000000</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Reserved</td>
</tr>
<tr>
<td>25</td>
<td>DeviceSpecific7</td>
<td>16#02000000</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Reserved</td>
</tr>
<tr>
<td>26</td>
<td>DeviceSpecific6</td>
<td>16#04000000</td>
<td>Communication problems due to flooding on Ethernet</td>
<td>Error</td>
<td>Event</td>
<td>High</td>
<td>The CI871 is flooded with too many Ethernet frames due to DoS attack. To protect itself, the CI871 has disabled the receiving of data until the flooding has stopped. This can cause communication interruptions on PROFINET.</td>
</tr>
<tr>
<td>27</td>
<td>DeviceSpecific5</td>
<td>16#08000000</td>
<td>PNIO Alarms blocked</td>
<td>Error</td>
<td>Event</td>
<td>High</td>
<td>The alarm handling on CI871 is blocked. Further alarms from the devices cannot be operated.</td>
</tr>
<tr>
<td>28</td>
<td>DeviceSpecific4</td>
<td>16#10000000</td>
<td>CEX watchdog expired on CI871</td>
<td>Error</td>
<td>Event</td>
<td>High</td>
<td>The CEX-Bus watchdog on CI871 was not triggered by the PM8xx processor module through the CEX-Bus.</td>
</tr>
<tr>
<td>29</td>
<td>DeviceSpecific3</td>
<td>16#20000000</td>
<td>Communication memory obtained too long</td>
<td>Warning</td>
<td>Event</td>
<td>High</td>
<td>Overload of the communication memory access. There is too much access from the application tasks to the PROFINET IO-data in the shared memory on the CI871 so that the CI871 cannot update the memory on time.</td>
</tr>
</tbody>
</table>
Section 2  Functional Description

Status Handling for PROFINET IO Devices

Table 2. PROFINET IO ErrorsAndWarnings for CI871 (Continued)

<table>
<thead>
<tr>
<th>Bit</th>
<th>Status bit</th>
<th>Value</th>
<th>Status Text</th>
<th>Status Type</th>
<th>Alarm/Event</th>
<th>Severity</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>30</td>
<td>DeviceSpecific2</td>
<td>16#40000000</td>
<td>Ethernet cable dropped</td>
<td>Error</td>
<td>Alarm</td>
<td>High</td>
<td>The Ethernet connector on CI871 is unplugged.</td>
</tr>
<tr>
<td>31</td>
<td>DeviceSpecific1</td>
<td>16#80000000</td>
<td>Hardware failure</td>
<td>Error</td>
<td>Event</td>
<td>High</td>
<td>The CI871 has identified a serious failure and cannot proceed execution.</td>
</tr>
</tbody>
</table>

Status Handling for PROFINET IO Devices

The CI871 handles status information for the connected PROFINET IO devices. If there is a failure on a device, an indication for the device or the specific I/O-unit appears in the hardware tree. For example see Figure 10: Unit Status for Wrong Module Type.

The error is identified by the CI871 because of missing communication or because the device itself has reported the error through an active PROFINET IO alarm to CI871. All errors are mapped to the specific unit status of the device, module or submodule.

Table 3. PROFINET IO ErrorsAndWarnings for PNIO Device, Modules, Submodules

<table>
<thead>
<tr>
<th>Bit</th>
<th>Status bit</th>
<th>Value</th>
<th>Status Text</th>
<th>Status Type</th>
<th>Alarm/Event</th>
<th>Severity</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>ConnectionDown</td>
<td>16#00000001</td>
<td>Connection down</td>
<td>Error</td>
<td>Alarm</td>
<td>High</td>
<td>No communication with the device. For the device and all connected modules and submodules ConnectionDown will be set.</td>
</tr>
<tr>
<td>1</td>
<td>IoError</td>
<td>16#00000002</td>
<td>I/O error</td>
<td>Error</td>
<td>Alarm</td>
<td>Medium</td>
<td>IO Error</td>
</tr>
<tr>
<td>2</td>
<td>ModuleMissing</td>
<td>16#00000004</td>
<td>Module missing</td>
<td>Error</td>
<td>Alarm</td>
<td>High</td>
<td>A configured module/submodule is physically missing.</td>
</tr>
<tr>
<td>3</td>
<td>WrongModuleType</td>
<td>16#00000008</td>
<td>Wrong module type</td>
<td>Error</td>
<td>Alarm</td>
<td>High</td>
<td>The configured module/submodule is of different type than the physical one.</td>
</tr>
</tbody>
</table>
### Table 3. PROFINET IO ErrorsAndWarnings for PNIO Device, Modules, Submodules (Continued)

<table>
<thead>
<tr>
<th>Bit</th>
<th>Status bit</th>
<th>Value</th>
<th>Status Text</th>
<th>Status Type</th>
<th>Alarm/Event</th>
<th>Severity</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>StatusChannelError</td>
<td>16#00000010</td>
<td>Channel error</td>
<td>Warning</td>
<td>Alarm</td>
<td>Medium</td>
<td>Channel Error</td>
</tr>
<tr>
<td>5</td>
<td>IoWarning</td>
<td>16#00000020</td>
<td>I/O warning</td>
<td>Warning</td>
<td>Event</td>
<td>Low</td>
<td>IO warning</td>
</tr>
<tr>
<td>6</td>
<td>StatusUnderflow</td>
<td>16#00000040</td>
<td>Underflow</td>
<td>Warning</td>
<td>Alarm</td>
<td>Low</td>
<td>Underflow</td>
</tr>
<tr>
<td>7</td>
<td>StatusOverflow</td>
<td>16#00000080</td>
<td>Overflow</td>
<td>Warning</td>
<td>Alarm</td>
<td>Low</td>
<td>Overflow</td>
</tr>
<tr>
<td>8</td>
<td>StatusForced</td>
<td>16#00000100</td>
<td>Forced</td>
<td>Warning</td>
<td>Event</td>
<td>Low</td>
<td>Forced</td>
</tr>
<tr>
<td>9</td>
<td>WatchdogTimeout</td>
<td>16#00000200</td>
<td>Watchdog timeout</td>
<td>Error</td>
<td>Alarm</td>
<td>High</td>
<td>Watchdog timeout</td>
</tr>
<tr>
<td>10</td>
<td>DeviceFailure</td>
<td>16#00000400</td>
<td>Device failure</td>
<td>Error</td>
<td>Alarm</td>
<td>High</td>
<td>Device failure</td>
</tr>
<tr>
<td>11</td>
<td>DeviceNotFound</td>
<td>16#00000800</td>
<td>Device not found</td>
<td>Error</td>
<td>Alarm</td>
<td>High</td>
<td>Device not found</td>
</tr>
<tr>
<td>12</td>
<td>WrongDeviceType</td>
<td>16#00010000</td>
<td>Wrong device type</td>
<td>Error</td>
<td>Alarm</td>
<td>High</td>
<td>Wrong device type</td>
</tr>
<tr>
<td>13</td>
<td>IOConnectError</td>
<td>16#00020000</td>
<td>I/O connection error</td>
<td>Error</td>
<td>Alarm</td>
<td>Medium</td>
<td>I/O connection error</td>
</tr>
<tr>
<td>14</td>
<td>IOConfigError</td>
<td>16#00040000</td>
<td>I/O configuration error</td>
<td>Error</td>
<td>Alarm</td>
<td>Medium</td>
<td>I/O configuration error</td>
</tr>
<tr>
<td>15</td>
<td>HWConfigError</td>
<td>16#00080000</td>
<td>Hardware configuration error</td>
<td>Error</td>
<td>Alarm</td>
<td>High</td>
<td>Hardware configuration error</td>
</tr>
<tr>
<td>16</td>
<td>GeneralError</td>
<td>16#00100000</td>
<td>-</td>
<td>Error</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>17</td>
<td>GeneralWarning</td>
<td>16#00200000</td>
<td>-</td>
<td>Warning</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>18</td>
<td>RedWarningPrimary</td>
<td>16#00400000</td>
<td>Warning on primary unit</td>
<td>Warning</td>
<td>Event</td>
<td>Low</td>
<td>Not supported</td>
</tr>
<tr>
<td>19</td>
<td>RedWarningBackup</td>
<td>16#00800000</td>
<td>Warning on backup unit</td>
<td>Warning</td>
<td>Event</td>
<td>Low</td>
<td>Not supported</td>
</tr>
<tr>
<td>20</td>
<td>RedErrorBackup</td>
<td>16#01000000</td>
<td>Error on backup unit</td>
<td>Warning</td>
<td>Alarm</td>
<td>Medium</td>
<td>Not supported</td>
</tr>
<tr>
<td>21</td>
<td>Reserved</td>
<td>16#00200000</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>Reserved</td>
</tr>
<tr>
<td>22</td>
<td>DeviceSpecific10</td>
<td>16#00400000</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>Reserved</td>
</tr>
<tr>
<td>23</td>
<td>DeviceSpecific9</td>
<td>16#00800000</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>Reserved</td>
</tr>
<tr>
<td>24</td>
<td>DeviceSpecific8</td>
<td>16#01000000</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>Reserved</td>
</tr>
</tbody>
</table>
Table 3. PROFINET IO ErrorsAndWarnings for PNIO Device, Modules, Submodules (Continued)

<table>
<thead>
<tr>
<th>Bit</th>
<th>Status bit</th>
<th>Value</th>
<th>Status Text</th>
<th>Status Type</th>
<th>Alarm/Event</th>
<th>Severity</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>25</td>
<td>DeviceSpecific7</td>
<td>16#02000000</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Reserved</td>
</tr>
<tr>
<td>26</td>
<td>DeviceSpecific6</td>
<td>16#04000000</td>
<td>Parameterization fault</td>
<td>Error</td>
<td>Alarm</td>
<td>High</td>
<td>Wrong, too less or too many parameters are written.</td>
</tr>
<tr>
<td>27</td>
<td>DeviceSpecific5</td>
<td>16#08000000</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Reserved</td>
</tr>
<tr>
<td>28</td>
<td>DeviceSpecific4</td>
<td>16#10000000</td>
<td>Locked by other controller /supervisor</td>
<td>Warning</td>
<td>Event</td>
<td>Medium</td>
<td>Indicates a change of parameter for a module/submodule. Will only be set if parameter was changed without download.</td>
</tr>
<tr>
<td>29</td>
<td>DeviceSpecific3</td>
<td>16#20000000</td>
<td>Diagnosis active</td>
<td>Warning</td>
<td>Event</td>
<td>Medium</td>
<td>HW-unit has active diagnosis.</td>
</tr>
<tr>
<td>30</td>
<td>DeviceSpecific2</td>
<td>16#40000000</td>
<td>Maintenance demanded</td>
<td>Warning</td>
<td>Event</td>
<td>Medium</td>
<td>Maintenance is demanded.</td>
</tr>
<tr>
<td>31</td>
<td>DeviceSpecific1</td>
<td>16#80000000</td>
<td>Maintenance required</td>
<td>Warning</td>
<td>Event</td>
<td>Medium</td>
<td>Maintenance is requested.</td>
</tr>
</tbody>
</table>

**Alarms and Events**

Alarms and Events are generated in the AC 800M controller for the PROFINET IO. For each alarm or event, a unit status bit in **ErrorsAndWarnings** or **ExtendedStatus** is defined within the hardware library of the device or CI871.

The source for an alarm or event is located in the device, the CI871 or the controller. In all the three cases, the time stamp for the alarm or event is generated in the AC 800M controller when the state of the corresponding unit status bit is changed.

The generated alarms and events are either of type system alarm or system event. The severity is defined through an attribute to the unit status bit.

For General status **Bit 22, ExtendedStatus**, the Controller Hardware object for CI871 display eA, indicating that CI871 supports Essential Automation (eA). For a detailed description, refer to **Appendix G, Hardware Units for Essential Automation**, of **AC 800M Controller Hardware (3BSE036351*)** manual.

For details on General status bit 22 ExtendedStatus, refer to **AC 800M Configuration (3BSE035980*)** manual.
PNIO Diagnosis

PROFINET IO supports integrated diagnosis that enables efficient fault localization and correction. For all the diagnostics sources, an indication is raised in the 800xA system which generates either a Unit Status or an alarm/event. The diagnosis is used to get status information from the PNIO devices and it indicates errors on process signal level during commissioning and runtime. The diagnosis is standardized and it is device specific as defined by the GSD files.

All standardized diagnostics in PROFINET IO system events are generated with textual presentation like:

- Connection down.
- Module missing/Wrong module type.
- Configuration error/Parameterization fault.
- Channel error/Line break.
- Upper limit value exceeded/Lower limit value exceeded.

With PROFINET IO, the following diagnosis information is available in the 800xA system:

- Unit Status.
- System Alarms/Events.
- Process Alarms.
- Signal Status.

Figure 10 shows an example of Unit Status for a remote I/O module having a wrong module type configured.
Figure 10. Unit Status for Wrong Module Type
Figure 11 shows an overview of PNIO diagnostics and its operations.

The PNIO alarms can be grouped in the following ways:

- Standard/Status alarms.
- Diagnosis alarms.
- Process alarms (Process alarms can generate SOE. For more information on SOE, refer to Process Alarms on page 46).

The alarms are sent event-driven by the device to the CI871. The CI871 interprets these alarms and triggers the Unit Status, system events or process alarms.
Structure of a PNIO Alarm

The following are the various PNIO alarm notifications and their syntax:

**Standard Status Alarm**

An example of Status-Alarm is:

```
Status change (Run, Stop, Ready) [true]
```

**Message** Provides a short description of the alarm type.

[true] Status of diagnosis
true: Appears
false: Disappears.

**Channel Diagnosis**

An example of Channel Diagnosis is:

```
Fiber optic mismatch Ch8 [true] E=32775 xE=32768 V=0xC8
```

**Message** Provides a short description of the alarm type.

**Ch** Channel Number.

[true] Status of diagnosis.
true: Appears.
false: Disappears.

**E** ChannelErrorType.

**xE** Extended ChannelErrorType.

Only shown in case of Extended Channel Diagnosis and Qualified Channel Diagnosis.

**V** ExtendedAddedValue.

Only shown in case of Extended Channel Diagnosis and Qualified Channel Diagnosis.

**Q** Qualified Channel Qualifier.

Only shown in case of Qualified Channel Diagnosis.
**Manufacturer Specific Diagnosis**

An example of Manufacturer Specific Diagnosis is:

Manufacturer Specific Ch8 USI=12345, Data

Manufacturer Specific is used as a fixed prefix.

- Ch: Channel Number.
- USI: User Structure Identifier.
- Data: Raw data, number of bytes.

**Process Alarms**

An example of Process Alarm is:

Process Alarm USI=12345, Data

Process Alarm is used as a fixed prefix.

- USI: User Structure Identifier.
- Data: Raw data, number of bytes.

USI and Data are device specific.
Sequence of Events (SOE)

Time stamped events are passed by PROFINET IO and the CI871 through the controller and are indicated in the AC 800M OPC Server in the Plant Explorer. The time stamping is done by the PNIO device itself. The SOE is supported using the ABB SOE profile.

The following are the definitions and functions of the ABB SOE Profile:

1. Alarms from the PNIO devices are converted into an External Event. These External Events are transferred through the AC 800M OPC Server and are indicated in the 800xA EventList with their corresponding source address.

2. The external event can be picked up in the IEC 61131-3 application by a Function block. The block treats the event like an alarm condition and converts it to process alarm.

3. The time synchronization of the devices is done externally and not by the CI871. It is the responsibility of the PNIO devices to receive time synchronization managed (through access to the central time master in the system). The devices defines the information to be time stamped.

4. The ABB SOE profile is handled as a process alarm on PROFINET IO with a vendor specific User Structure Identifier (USI).

5. The device deletes the SOE alarm only after receiving the acknowledgement from the controller. The controller sends the acknowledgement after operating the alarm.

It is recommended to configure function blocks as Alarm condition for process signals, only when the process values can be used as initial values in case of restart. Otherwise, the alarm state becomes uncertain.

Figure 12 displays the workflow of ABB SOE profile.
The various address levels of SOE events are:

- I/O channel
- Submodule
- Module
- Device
Example of an SOE

The maximum length of the message for a system event is 70 characters. The message includes a process alarm and some additional information, which includes:

- Message [40]
- Alarm ID
- Channel Number
- Monitored Value

There can be several alarms related to the same hardware unit without any relation to an I/O channel. There can also be several alarms related to the same I/O channel.

The AlarmIds are used to differentiate the various alarms generated at the same hardware level. The AlarmId is also used as part of SignalId because it is mandatory to have an additional address level for the alarms, irrespective of whether the alarm is related to the module level or channel level.

External events require a unique address to be mapped to the function blocks in the IEC 61131-3 application. The SignalId is one of the parameters used to configure the function blocks.

The SignalId consists of the hardware address, channel number and the AlarmId.

The following is the syntax for SignalId:

```
cc.ddd.mmmmm.sssss.ccccc.aaaaa
```

- \(c\) = CI module number (max 12)
- \(d\) = device address
- \(m\) = module address (slot)
- \(s\) = submodule address (subslot)
- \(c\) = channel number
- \(a\) = alarm identifier
Figure 13 displays the configuration of SignalId in a Function block.

![Diagram of SignalId configuration](image)

Figure 13. Example of an SOE with module related trigger

The value of the parameter SignalId is 3.1.1.2025 where:

- 3 CI module number
- 1 device address
- 1 module address
- 2025 alarm identifier

In case a device has an external event at the channel level, then the value of the parameter SignalId is 2.15.4.3.6.2025 where:

- 2 CI module number
- 15 device address
- 4 module address
- 3 submodule address
- 6 channel number
- 2025 alarm identifier

Devices supporting ABB SOE Profile are *MNS iS via MLink* and *UMC100 via PNQ22*. For both, the SOE events are typically indicated on Motor Starter level, which is the module level in PROFINET IO.
MNS iS:
The following are the examples of SOE along with their AlarmId for MNS iS at the Motor Starter level:

- Warning Main Switch Supervision (1006).
- Trip Main Switch Supervision (2019).
- Warning Thermal Overload (1006).

UMC100 via PNQ22
The following are the examples of SOE along with their AlarmId for UMC100 at the Motor Starter level:

- Thermal Overload Trip (0)
- PTC Temperature High (5)
- Num Starts Overrun (43)
- Cooling Time Running (45)

The message within an ExternalEvent differs for a channel and a hardware unit. If it is related to a channel, then the Channel numbers are shown as:

Overvoltage (12) Ch10 [true]

where 12 is the AlarmId and 10 is the Channel Number.

If the external event is for a hardware unit, then the message is shown as:

WARNING Thermal Overload (14) [true]

where 14 is the AlarmId.
Figure 14 shows a list of events displayed in the EventList.

![Event List Example]

Figure 14. An Example of EventList

**Acyclic Data Communication**

In addition to the normal cyclic communication, the AC 800M supports acyclic communication with the connected PNIO devices as well. With acyclic communication it is possible to Read/Write data on demand via IEC 61131-3 application. Typically wise with acyclic communication it is possible to have access to data that is normally not available via cyclic communication.

To get acyclic access with a PNIO device first the cyclic communication must be configured and up and running. Then the acyclic data is available via the function
blocks IOConnect, IORead and IOWrite as part of the library IOCommLib. Data records with up to 4 KBytes are supported.

Typical use cases for usage of acyclic communication:

- Drive status and parameterization
- Advanced Asset data of UMC100
- Advanced diagnosis support
- Advanced Controller/Controller communication
- HART Pass Through
- I&M data

**Drive status and parameterization:**
Depending on the application needs drive parameters can be read or written. Examples are window control limits, torque reference limits or start mode. **Figure 17** shows an example of acyclic communication with ABB Drives ACS880 via FENA-11.

**Advanced Asset Data of UMC100:**
UMC100 offers acyclic access to monitoring data like:

- Number of starts
- Number of trips
- Number of therm. Overload trips
- Operating hours
- Max. Current at Startup

**Advanced Diagnosis Support:**
If needed the diagnosis data of a PNIO device can be read on demand and made available to the IEC 61131 application. E.g. the complete diagnosis frame of UMC100 can be accessed and operated by IEC 61131 application.

**Advanced Controller/Controller communication:**
In addition to the standard cyclic communication with the 3rd party PLC via PN/PN-coupler or i-Device functionality, the acyclic communication can be used to
save cost. The benefit is that the number of acyclic frames and the size of each frame is bigger than available with the cyclic communication. This reduces e.g. the number of PN/PN-couplers in a large configuration.

**HART Pass Through:**

If a PNIO device has support for HART Pass Through functionality then it is possible to access the connected HART device via read/write services to e.g. upload the secondary HART variables into the controller.

**I&M Data:**

The Identification & Maintenance data is mandatory with PROFINET IO. For e.g. maintenance purpose it is possible to read the I&M data for each connected device to get following information:

- Hardware Revision
- Software Revision
- Serial Number
- Order Number
Figure 15 shows an example of data access using acyclic communication in IEC 61131-3 Application.

The read or write data on PROFINET is transferred in one frame. A single frame can have a length of up to 4 KBytes. In IEC 61131-3 application, the frame is split into several parameters. These parameters are of type extensible at the read or write function block. That is, the number of parameters must be defined as a property of the read and write function block. Up to 32 parameters \( SD[1..32], RD[1..32] \) are supported for each function block. Depending on the number and data types of the configured parameters, the PROFINET read or write frame is configured.

With standard datatypes, it is possible to get a PROFINET frame of 128 bytes (32x4bytes). To get more data transferred, use structured instead of standard datatypes for the parameters SD[1..32], RD[1..32].
Figure 16 shows an example of HART data access using acyclic communication in IEC 61131-3 Application.

For CI871 device performance with acyclic communication, refer Section 6, Technical Data and Performance.

![IEC 61131 Application](image)

**Figure 16. HART Data access using Acyclic Data Communication**

**Configuring IOCommLib Function Block**

For each PNIO device, IOConnect function block is first configured to setup the communication relation between CI871 and the PNIO device. The CI871 is defined by the CEX-position, which is the Channel parameter on the IOConnect function block. The Partner parameter on the IOConnect defines the device position number in the hardware tree below the related CI871.

The Id output parameter of the IOConnect is connected to all related IORead and IOWrite function blocks that is handled through the same communication relation.
The address must be configured on the **IORead** and **IOWrite** function blocks where the data on the device is read or written. The address is defined by the **VarName**.

The **VarName** parameter gets operated by the **IORead** and **IOWrite** only by a positive edge of the **En_C** parameter on the related **IOConnect**.

For PROFINET the **VarName** parameter is in the following format:

```xml
<HwTrPosM>.<HwTrPosSuM>:<Index>.<API>
```

For example: `1.1:B02F.14848`

- **Module** = 1
- **Submodule** = 1
- **Index** = B02F
- **API** = 14848 (PROFIdrive Profile)

Where:

- **HwTrPosM**: This is the Hardware Tree position of the module below the device. The value ranges from 0 to 65535.
- **HwTrPosSuM**: This is the Hardware Tree position of the sub-module below the module. The value ranges from 0 to 65535.
- **Index**: Index is a number for the address information of the data to be read/written on the submodule. The index is either defined by PROFINET for common data like I&M data or is device specific and described in the manual of the device.
  
  Definition is in hexadecimal and the range is from 0x0 to 0xFFFF (supports upper or lower case).
- **API**: API is an optional information and in most cases the default value 0 is used. API defines the used Application Process Identifier. For example, API for PROFIdrive is defined by 14848 or 0x3A00
  - For example: `1.2:0xAFF0`

  0xAFF0 is the index of the connected device on the module 1 and submodule 2. API value is zero, as the API value is not appended on the device.
For example: 1.0:13Fa.14848
13Fa is the index of the connected device on module 1 and sub-module 0. The API value is 14848.

Table 4, Table 5 and Table 6 explains the IOConnect, IORead, and IOWrite function block parameters used for acyclic communication:

**Table 4. IOConnect Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Data type</th>
<th>Direction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>En_C</td>
<td>bool</td>
<td>in</td>
<td>Enables the execution while True.</td>
</tr>
<tr>
<td>Channel</td>
<td>string[16]</td>
<td>in</td>
<td>The local communication channel to be used for connection.</td>
</tr>
<tr>
<td>Partner</td>
<td>string[16]</td>
<td>in</td>
<td>The remote communication partner/Identity/Node/Slave.</td>
</tr>
<tr>
<td>Valid</td>
<td>bool</td>
<td>out</td>
<td>True when output values are valid.</td>
</tr>
<tr>
<td>Error</td>
<td>bool</td>
<td>out</td>
<td>Indicates an error with True during one scan. Status parameter &lt;0.</td>
</tr>
</tbody>
</table>
| Status    | dint      | out       | Status code of last execution
  - Operation successful 1
  - pending 0
  - errors <0
  - warnings <0 |
| Id        | Comm_Channel | out | Communication channel, has to be connected to a FB of the type Read/Write. |
### Table 5. IORead Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Data type</th>
<th>Direction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Id</td>
<td>Comm_Channel</td>
<td>out</td>
<td>Communication channel, has to be connected to the Id- par. of the Connect- FB.</td>
</tr>
<tr>
<td>Req</td>
<td>bool</td>
<td>in</td>
<td>Requests one operation on positive edge, True.</td>
</tr>
<tr>
<td>VarName</td>
<td>string[40]</td>
<td>in</td>
<td>The name of the “variable n” in remote system, read after first invocation.</td>
</tr>
<tr>
<td>Ndr</td>
<td>bool</td>
<td>out</td>
<td>True when New Data has been received on each FB call after successful operation.</td>
</tr>
<tr>
<td>Error</td>
<td>bool</td>
<td>out</td>
<td>Indicates an error with True during one scan. Status parameter &lt;0.</td>
</tr>
<tr>
<td>Status</td>
<td>dint</td>
<td>out</td>
<td>Status code of last execution</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- Operation successful 1</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- pending 0</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- errors &lt;0</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- warnings &lt;0</td>
</tr>
<tr>
<td>RD[1]</td>
<td>AnyType</td>
<td>out</td>
<td>Read value of the variable “VarName”</td>
</tr>
<tr>
<td>RD[2]</td>
<td>AnyType</td>
<td>out</td>
<td>Read value of the variable “VarName”</td>
</tr>
<tr>
<td>RD[3]</td>
<td>AnyType</td>
<td>out</td>
<td>Read value of the variable “VarName”</td>
</tr>
<tr>
<td>RD[4]</td>
<td>AnyType</td>
<td>out</td>
<td>Read value of the variable “VarName”</td>
</tr>
</tbody>
</table>

### Table 6. IOWrite Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Data type</th>
<th>Direction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Id</td>
<td>Comm_Channel</td>
<td>out</td>
<td>Communication channel, has to be connected to the Id- par. of the Connect- FB.</td>
</tr>
<tr>
<td>Req</td>
<td>bool</td>
<td></td>
<td>Requests one operation on positive edge, True.</td>
</tr>
<tr>
<td>VarName</td>
<td>string[40]</td>
<td>in</td>
<td>The name of the “variable n” in remote system, read after first invocation.</td>
</tr>
</tbody>
</table>
For acyclic communication, the parameter Status at the function blocks IOConnect, IORead and IOWrite provides PROFINET specific error codes. The definition of the error codes is derived from the PROFINET standard. Errors are indicated by CI871 or the PNIO device.

The PROFINET specific error codes are mapped to the lowest byte of the protocol specific Status in the range of \(-7424 \text{ to } -7169\) or the PNIO device.

The range of \(-7160 \text{ to } -7165\) is used by CI871 specific error codes.

For the list of possible error codes, refer to, Appendix A, CI871 Error Codes.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Data type</th>
<th>Direction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Done</td>
<td>bool</td>
<td>out</td>
<td>True when requested operation is done successful.</td>
</tr>
<tr>
<td>Error</td>
<td>bool</td>
<td>out</td>
<td>Indicates an error with True, Status parameter &lt;0.</td>
</tr>
</tbody>
</table>
| Status    | dint      | out       | Status code of last execution  
• Operation successful 1  
• pending 0  
• errors <0  
• warnings <0 |
| SD[1]     | AnyType   | in        | In variable_to be transferred. |
| SD[2]     | AnyType   | in        | In variable_to be transferred. |
| SD[3]     | AnyType   | in        | In variable_to be transferred. |
| SD[4]     | AnyType   | in        | In variable_to be transferred. |
Figure 17. Example of an Acyclic Communication in IEC 61131-3 Application with ABB Drives ACS880 via FENA-11

<table>
<thead>
<tr>
<th>Sl. No.</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Connect FENA-11, setup communication relation</td>
</tr>
<tr>
<td>2</td>
<td>Write Read-Request for DC Voltage to FENA-11</td>
</tr>
<tr>
<td>3</td>
<td>Read DC Voltage from FENA-11</td>
</tr>
<tr>
<td>4</td>
<td>Write Limits:Minimum Speed to FENA-11</td>
</tr>
</tbody>
</table>

- To read data from an ABB Drive, a command **IOWrite** function block (**Drive_Write**) is written to the drive that defines which data to be read. In a second step the requested data can be read using **IORead** function block (**Drive_Read**).
The second `IOWrite` function block (`Drive_WritePara`) is used to write a new value for the `Minimum Speed` to the drive by use of the ABB Drive specific command structure.

**Addressing in PROFINET IO**

The following are the sequence of steps for addressing PNIO devices in 800xA:

- The devices need a symbolic name to establish a connection and communicate with the 800xA System.

- The Discovery Control Protocol (DCP) is used to assign a symbolic name to each PROFINET IO device in 800xA by use of the web server functionality of AC 800M. The IP address for the PNIO device is assigned in the Control Builder. For more information on how to assign symbolic name to the PNIO devices refer to [Set Devicename](#) on page 97.

- Configure the PNIO Device in Control Builder, using the steps described in [Inserting a PROFINET IO Device](#) on page 69.

After the system is configured, the Control Builder loads all information required for data exchange to the CI871, including the IP addresses of the connected PNIO devices. After address resolution, the configuration and parameters are transmitted to the PNIO devices.

A symbolic name should be assigned to the PNIO devices only once as it is stored in their nonvolatile memory.

**Hot Swap**

CI871 supports hot swap without any configuration changes. Faulty units can be replaced online, without disturbing other units connected to the CEX-Bus. After hot removal and the subsequent insertion, the new CI871 gets configured automatically.

During operation, if the non redundant CI871 is hot removed then all connected I/Os are set to bad, and Input Set as Predetermined (ISP) and Output Set as Predetermined (OSP) are activated.
Section 3  Configuration

Hardware Library

The hardware libraries are used to configure the CI871 and the hardware types of PROFINET IO devices. For more information about handling hardware libraries, refer to System 800xA Control AC 800M Configuration (3BSE035980*) Manual.

Inserting CI871 PROFINET IO Controller Unit

To insert a new CI871 PROFINET IO controller unit under an AC 800M controller:

1. Right-click the AC 800M controller and select **Insert Unit**. The Insert Unit dialog appears as shown in **Figure 18**.

![Figure 18. Dialog to Insert CI871](image-url)
2. Expand the library for CI871 under Connected Libraries and select the CI871 hardware type.

3. Select a position for the hardware unit in the Position drop-down list. The first available position is chosen by default. Redundancy cannot be enabled for CI871.

4. Enter a Name if required.

5. Click Insert to apply the current changes.

6. Click Close to close the dialog or proceed with inserting additional units.

The hardware unit is now included in the hardware tree, see Figure 19.

![Figure 19. CI871 in Hardware Tree](image)

Configuring CI871 PROFINET IO Controller Unit

Hardware Editor

Double-click on the CI871, or right-click and select Editor from the context menu to open the Hardware Editor. The Hardware Editor contains Settings, Connections and Unit Status tabs.
Settings Tab

![Figure 20. Settings for CI871](image)

**Table 7** describes the parameters available in the **Settings** tab of a CI871:

**Table 7. Configuration Settings for CI871**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Station name</td>
<td>Station name is the symbolic name for the CI871. The Station name must be unique for PROFINET IO. The same syntax applies for PROFINET IO devices. For more information, refer to <strong>Table 8: Basic Communication Settings for PROFINET IO Device</strong>.</td>
</tr>
<tr>
<td>IP address</td>
<td>IP address identifies the CI871 on the PROFINET. An IP address consists of four groups of numbers (0-255), separated by periods.</td>
</tr>
</tbody>
</table>
Table 7. Configuration Settings for CI871 (Continued)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Subnet mask</td>
<td>Subnet mask along with the IP address identifies the network to which the system is connected. An IP subnet mask consists of four groups of numbers (0-255), separated by periods.</td>
</tr>
<tr>
<td>Default gateway</td>
<td>IP address of the station connected to the PROFINET IO that acts as a router to other networks.</td>
</tr>
<tr>
<td>Red. Ethernet recovery time</td>
<td>The time it takes for a redundant Ethernet to switch the communication from the disturbed previous path A to the alternative path B. The Red. Ethernet recovery time is used to configure the watchdog time so that the cyclic PROFINET IO communication withstands errors on the network. For more information, refer to Ethernet Network Redundancy on page 36.</td>
</tr>
<tr>
<td>Watchdog factor</td>
<td>Defines the timeout used to monitor the receiving of data on side of CI871 and the device. The timeout is given by the calculation: WatchdogTime&lt;sub&gt;Ci871&lt;/sub&gt; = Watchdog factor x Update Time Inputs WatchdogTime&lt;sub&gt;Device&lt;/sub&gt; = Watchdog factor x Update Time Outputs. The value must be divisible by 3. If the entered value is not divisible by 3, Control Builder will automatically recalculate it and use the next possible greater value, that is divisible by 3.</td>
</tr>
<tr>
<td>Disable diagnosis alarms</td>
<td>Disables the operation of diagnosis alarms from PROFINET IO. Typically used during commissioning.</td>
</tr>
<tr>
<td></td>
<td><strong>Note</strong>: In some cases, alarms will still continue to show up due to the presence of other diagnostic functions which are enabled. Default: off</td>
</tr>
<tr>
<td>Disable process alarms (SOE)</td>
<td>Disables the operation of process alarms from PROFINET. Because SOE is also handled through process alarms, this flag disables the SOE functionality.</td>
</tr>
<tr>
<td></td>
<td>Default: off</td>
</tr>
</tbody>
</table>
Control Builder checks whether the Red. Ethernet recovery time or the Watchdog factor defines the greatest timeout. The greatest resulting timeout will be downloaded.

If the configured Watchdog factor defines the greatest timeout, then it will be used as configured only with the correction that it needs to be divisible by 3.

If the Red. Ethernet recovery time defines the greatest timeout, then the resulting watchdog time is given by the calculation:

\[
\text{WatchdogTime}_{\text{CI871}} = \text{Red. Ethernet recovery time} + 2 \times \text{Update Time Inputs}
\]

\[
\text{WatchdogTime}_{\text{Device}} = \text{Red. Ethernet recovery time} + 2 \times \text{Update Time Outputs}
\]
Also in the second case the Control Builder calculates based on the watchdog time, a resulting Watchdog factor that is divisible by 3.

The calculated and downloaded watchdog factor can only be seen in the message pane of Control Builder and the logfile PROFINET_Configuration.txt. PROFINET IO defines the maximum watchdog timeout as 1920 ms. Control Builder ensures that this limit is not exceeded.

**Connection Tab**

A connection to the UnitStatus on the CI871 can be made. For more information on connecting variables to I/O channels, refer to *System 800xA Control AC 800M Configuration (3BSE035980*) Manual.

**Unit Status Tab**

For more information on the Unit Status tab, refer to *System 800xA Control AC 800M Configuration (3BSE035980*) Manual.*
Inserting a PROFINET IO Device

To insert a PROFINET IO device under a PROFINET IO controller (CI871):

1. Right-click the CI871 and select **Insert Unit**. The Insert Unit dialog appears as shown in **Figure 21**.

![Insert Unit for CI871](image)

**Figure 21. Dialog to Insert Phoenix Interbus Proxy**

2. Under Connected Libraries, expand the library for the PROFINET IO device (for example, expand the **PXC_InterbusProxy** library as shown in **Figure 21**). The library can contain several devices of a device family. For example, the Phoenix Interbus Proxy has two device types. One proxy has an optical connection and the other has an electrical connection for the Interbus.

3. Select the required hardware type to be inserted as shown in **Figure 21**. Select a position for the hardware unit from the **Position** drop-down list. The first available position is chosen by default. If another position is needed, click the **Position** drop-down list to display the available positions and select the required position.
The position configured using Control Builder is the position in the hardware tree. Communication relations are configured through the Settings tab of each device.

Redundancy cannot be enabled for a PROFINET IO device.

4. Enter a Name if required.
5. Click Insert to apply the current changes.
6. Click Close to close the dialog or proceed inserting further units.

The hardware unit is now included in the hardware tree as shown in Figure 22.

Figure 22. Phoenix Interbus Proxy in Hardware Tree

If the PROFINET IO device is a modular one similar to the Phoenix Interbus Proxy, then add the I/O modules and additional submodules for the required functionality.
7. Right-click the device for which the I/O modules are to be inserted and select **Insert Unit**. The Insert Unit dialog appears as shown in **Figure 23**.

![Insert Unit for FL NP PND-4TX IB-LK V2.0](image)

*Figure 23. Insert I/O modules for Phoenix Interbus Proxy*

By default, the DAP module is configured at position 0. This module is required to setup the PROFINET IO communication. The name of the DAP module is identical with the name of the device itself with the extension **_DAP**.
8. When an I/O module is inserted, the submodules of that I/O module are automatically inserted as default, if defined in the GSD file. This feature is device specific and depends on the functionality of the specific I/O modules. If the device supports the configuration of several submodules below an I/O module, then the user can configure this depending on the functional needs.

9. To configure additional submodules, right-click the I/O module under which the submodules are to be inserted and select **Insert Unit**. Figure 25 shows an example of Phoenix device with I/O modules and submodules inserted in the hardware tree.

![Figure 24. PROFINET IO Device with I/O modules in Hardware Tree](image)

Modules/Submodules having the attribute **UsedInSlots** or **FixedInSlots** (as defined in the GSD file) are inserted automatically (For example in MNS iS). **UsedInSlots** is a default configuration but the Modules/Submodules can be deleted, for Modules/Submodules having the attribute **FixedInSlots** the configuration cannot be changed.
Configuring PROFINET IO Device Unit

Configuration Options

The hardware topology is configured by inserting the PROFINET IO device, the I/O modules and the submodules in the hardware tree. The inserted PROFINET IO device needs to be further configured.
The available configuration options are:

- Station name.
- Basic communication settings.
- Cyclic communication options.
- Startup parameters.
- I/O data.

**Station Name**

PROFINET IO devices require an unique station name to allow PROFINET IO communication. The station name is stored nonvolatile in the device and must be identical with the station name configured for the device in Control Builder. For more information, refer to Basic Communication Settings on page 74. To configure the station name on the device, refer to Set Devicename on page 97.

**Basic Communication Settings**

To allow the CI871 as the PROFINET IO controller to setup a communication to the assigned PROFINET IO device, the basic device specific communication settings such as Station name and IP address must be configured. Basic communication settings are defined on PROFINET IO device level.

To define the basic communication settings, double-click the device (or right-click the device and select Editor). The **Hardware Editor** dialog appears as shown in Figure 26. Select the **Settings** tab and enter the parameters as required.

*Figure 26. Basic Communication Settings for PNIO Device*
Table 8. Basic Communication Settings for PNIO Device

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Station name</td>
<td>The station name configured in Control Builder must be unique and identical to the station name stored nonvolatile in the device. This includes all PROFINET IO controller units and PROFINET IO device units sharing the same Ethernet.</td>
</tr>
<tr>
<td></td>
<td>The following syntax applies:</td>
</tr>
<tr>
<td></td>
<td>• String with 1 to 63 octets.</td>
</tr>
<tr>
<td></td>
<td>• One or more labels, separated by [.] A maximum of four labels can exist.</td>
</tr>
<tr>
<td></td>
<td>• Labels consist of [a...z] and [0...9] and [-].</td>
</tr>
<tr>
<td></td>
<td>• Labels do not start with [-].</td>
</tr>
<tr>
<td></td>
<td>• Labels do not end with [-].</td>
</tr>
<tr>
<td></td>
<td>• First label does not start with [0...9].</td>
</tr>
<tr>
<td></td>
<td>• Station-names do not have the form n.n.n.n with n = 0...999.</td>
</tr>
<tr>
<td></td>
<td>• The first label does not start with <code>port-xyz</code> or <code>port-xyz-abcde</code> with a,b,c,d,e, x, y, z = 0...9.</td>
</tr>
<tr>
<td></td>
<td>The PROFINET IO communication cannot be established if the station name has any capital letters.</td>
</tr>
<tr>
<td>IP address</td>
<td>The IP address is required to setup the CI871 communication with the PROFINET IO device. An IP address consists of four groups of numbers (0-255), separated by periods.</td>
</tr>
<tr>
<td></td>
<td>The IP address is downloaded during the startup of the device.</td>
</tr>
<tr>
<td></td>
<td>Communication with the I/O device is not possible, until the IP address is downloaded to the device and the PROFINET IO Communication with the CI871 is established. For example, if the device supports a Web server, then it can be used only after startup of the PROFINET IO communication.</td>
</tr>
<tr>
<td>Subnet mask</td>
<td>The Subnet mask along with the IP address identifies the network to which the system is connected. An IP Subnet mask consists of four groups of numbers (0-255), separated by periods.</td>
</tr>
<tr>
<td>Default gateway</td>
<td>IP address of the station connected to the PROFINET IO that acts as a router to other networks.</td>
</tr>
</tbody>
</table>
Control Builder generates a compilation error if it detects duplicate Station names and/or IP addresses that are only configured below CI871. That is, if there are several CI871 connected to one and/or several controller(s) all connecting to the same Ethernet network, then the complete configuration cannot be checked. For such kind of configurations, user must ensure that Station names and IP addresses are unique.

**Cyclic Communication Options**

With cyclic communication options specific cyclic update times can be configured. The update times are handled separately for inputs and outputs for the specific PNIO device. The configuration is defined at the DAP level (module position 0).

To configure the update times, double-click the DAP, (or right-click the DAP and select Editor). The Hardware Editor dialog appears, as displayed in Figure 27. Select the Settings tab and enter the parameters as required.

![Figure 27. Cyclic Communication Options for PNIO Device](image)
### Table 9. Cyclic Communication Options for PNIO Device

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Update Time Inputs</td>
<td>Defines the update time for the input data. The permitted possible values are device specific. The value range defined by PROFINET IO is 1–512 ms. Specific values are defined with 1, 2, 4, 8, 16, 32, 64, 128, 256, 512 ms. The default update time inputs is defined as 32 ms in Device Import Wizard.</td>
</tr>
<tr>
<td>Update Time Outputs</td>
<td>Defines the update time for the output data. The permitted possible values are device specific. The value range defined by PROFINET IO is 1–512 ms. Specific values are defined with 1, 2, 4, 8, 16, 32, 64, 128, 256, 512 ms. The default update time outputs is defined as 32 ms in Device Import Wizard.</td>
</tr>
</tbody>
</table>

### Startup Parameters

The startup parameters are device specific and are used to configure the behavior of the device at startup. Startup parameters are available only on submodule level. They are configured in the **Settings** tab through the Hardware Editor of Control Builder. Double-click the submodule that has to be configured (or right-click the submodule and select **Editor**). The **Hardware Editor** dialog appears as shown in Figure 28. Select the **Settings** tab and enter the parameters as required.
Figure 28 shows the Phoenix Interbus Proxy settings that has to be configured through the submodule below the DAP.

![Figure 28. Startup Parameters for Phoenix Interbus Proxy](image)

**I/O Data**

I/O data is read or written using different data types depending on the device functionality. Hardware units expose I/O channels that can be connected to IEC 61131-3 applications. I/O data is only available on submodule level. Right-click the submodule to connect variables to and select **Editor**. The **Hardware Editor** dialog appears. Select the **Connections** tab to set the channel assignments.
Figure 29 shows the channel assignments for an analog input module of the Phoenix Interbus Proxy.

![Channel Assignment](image)

**Figure 29. Channel Assignment**

## Complex Device Example MNS iS

Simple devices like Remote I/Os do not really benefit from the availability of the submodules with PROFINET IO. A Remote I/O typically has only one submodule below the module because the functionality of the I/O module is limited. But a complex device such as the Motor Control Center MNS iS provides a wider range of functionalities that demands the need for having more configuration options.

MNS iS is handled as one PNIO device having 60 motor starters. Each motor starter is handled as one I/O module. The functionality of each motor starter can be configured instance specific and is scalable. Subfunctions for each motor starter type are defined by different types of submodules below the I/O module. The submodules and by that the functionality of the motor starter, can be configured on instance level.

Figure 30 shows an example configuration of MNS iS. It shows two different configurations of the motor starter RevDol. The first RevDol has just the RevDol Control submodule configured, the second RevDol has Measurements, Thermal Overload and General Purpose IO configured.
New Device Types

The Device Import Wizard helps to integrate new device types, that is not provided with the delivered hardware libraries. The wizard parses the provided GSD files and converts them into hardware types which can be integrated into the Control Builder. For further information on importing GSD files, refer to Section 7, Device Import Wizard.

Figure 30. Usage of Submodules with example of MNS iS
Section 4  Download and Online Mode

This section describes the procedure how to download an application and process online data.

Prerequisites

Check the hardware configuration of the PROFINET IO devices and the PROFINET Ethernet installation. Ensure that no duplicate addresses are configured and switches support Full Duplex mode.

Before downloading to the AC 800M controller, ensure that the processor module and CI871 have the correct firmware loaded. For more information, refer to Control Builder documentation and online help.

Downloading Configuration Settings

In the Control Builder, click Download and Go Online to download all the configuration settings to the system. This includes the settings for the controller, the CI871, and the PROFINET IO devices.

Online Changes

With PROFINET IO in 800xA it is possible to insert and delete PNIO devices during runtime without interrupting the active communication and without interrupting the running application. Bumpless download is ensured by inserting a new device to an empty HW tree position. Moving a device in the HW tree causes an interruption of the active communication.
The following configuration changes cannot be downloaded online:

- Station Name.
- Basic Communication Settings.
- Cyclic Communication Options.
- Startup Parameters.
- Position of Device in Hardware Tree.

Any of these changes will stop the PROFINET IO communication, at least for a short while. This will cause the PNIO device to activate fail safe values. The Control Builder checks for the CPU load of the CI871 and prohibits the download in case of an overload situation.

Depending on the configuration change, Control Builder indicates a warning in the Compilation Summary window as shown in Figure 31. The warning states that the communication will be temporarily lost for the listed hardware units in case the download is continued.

![Compilation Summary](image)

**Figure 31. Warning: Loss of Communication during Download**

**Logfile**

The Control Builder creates a logfile `PROFINET_Configuration.txt` during download. The logfile is placed in the logfile directory, called LogFiles, of Control Builder. The logfile contains the result of the downloaded compilation for
the current and previous configurations up to a size of 10 Mb. The result of the current compilation is located at the end of the file.

If the file has exceeded the maximum size, then the PROFINET_Configuration.txt will be saved as PROFINET_Configuration1.txt and a new PROFINET_Configuration.txt will be created. Up to nine old files will be saved before the oldest one will be overwritten.

The logfile also contains internally calculated data which is not available in Control Builder. This logfile is required only for maintenance operations.

Online Data

When going online the hardware tree shows the current status for all hardware units in the form of an error or warning indication. For detailed information, double-click the specific hardware unit to open the Editor and select the Unit Status tab to access the online information. see Figure 10: Unit Status for Wrong Module Type as an example for a Remote I/O having a wrong module type configured.

Behavior of I/O and Communication

Insertion and Deletion of I/O Units

When the I/O units are inserted or deleted, the corresponding outputs are set to zero.

Connection Error

The PROFINET IO communication between CI871 and PNIO device is monitored by both units. When an error is detected, particular operations of the CI871 and the PROFINET IO device are activated.

If the CI871 detects loss of communication with the PROFINET IO device, then the error Connection down is indicated for the specific device. The AC 800M controller then stops copying all input data, and sets the input values to 0 and activates Input Set as Predetermined (ISP) control for the related variables (if supported by the device). During Connection down, the status for Error/Warning and Alarm/Event does not change. If the connection is re-established, then the status is updated.
If the PROFINET IO device detects a loss of communication with the CI871, it activates fail safe values.

Reconfiguration or a temporary error such as cable error can cause a connection error. If the error is corrected, then the connection is re-established automatically.

**System Error**

The communication between the PM8xx and the CI871 through the CEX-Bus is monitored by a **Watchdog**. If the CI871 Watchdog expires due to a communication error, then the corresponding CI871 switches into a safe state mode and disables all communication with PROFINET IO. The communication with all interrelated PROFINET IO devices get interrupted. To get back the CI871 to the normal operating mode, either reset the controller or perform a hot swap of the CI871.
Section 5  CI871 Web Server

The CI871 PROFINET IO web server interface provides additional feature through a web browser interface. These features are used during commissioning or maintenance.

CI871 Web Server Login Prerequisite

The web server interface can be accessed through web browsers such as Microsoft Internet Explorer or Mozilla Firefox. Before logging into web server, ensure that the following web browser settings are changed in Internet Options > Security tab.

- It is recommended to connect to web server from Windows 8 client machine. The browser setting changes in Internet Options > Security tab is only done to connect from a Windows 2012 R2 server.

- Ensure that correct browser settings for the connection (for example, no proxy server for the controller address) are set, and the controller must be accessible from the system.

- Ensure that TCP/IP forwarding is enabled on Connectivity Servers. For more information, refer to System 800xA Network Configuration (3BSE034463*).
Web Server Login

It is recommended to connect to web server from Windows 8 client machine. The browser setting changes in Internet Options > Security tab is only done to connect from a Windows 2012 R2 server.

To overcome unsuccessful login attempt in web server, open Internet Options and in the Security tab, and ensure that for all three zones (Internet, Local Intranet, and Trusted Sites), Enable Protected Mode (requires restarting Internet Explorer) check box is cleared as shown in sample Figure 32.

Figure 32. Internet Options - Security Tab, Zone Settings
Enable Javascript for Web Server

It is recommended to connect to web server from Windows 8 client machine. The browser setting changes in Internet Options > Security tab is only done to connect from a Windows 2012 R2 server.

To enable the javascript for using the web server, open Internet Options and click Custom Level in the Security tab, as shown in sample Figure 33.

Figure 33. Internet Options - Security Tab
In the Security Settings, for **Scripting** and **Scripting of Java applets**, select the **Enable** option as shown in sample Figure 34.

![Security Settings - Internet Zone](image)

**Figure 34. Custom Level Security Settings - Enable Javascript**

Ensure that the same settings are done for all three zones (**Internet**, **Local Intranet**, and **Trusted Sites**).

**Reset Web Browser Security settings for Web Server**

The web browser settings mentioned in the **Web Server Login** on page 86 and **Enable Javascript for Web Server** on page 87 must be retained to its original settings after logging out of web server. This reset is manually done by changing the settings in all the three zones (**Internet**, **Local Intranet**, and **Trusted Sites**).
**CI871 Web Server Security**

The web server is disabled by default. It can be enabled on request using the **Enable Web Server** option available in Control Builder. This option is visible only when the Control Builder is online and at least one of the modules CI854, CI860 and CI871 is configured.

For more information on accessing the enable web server option in Control Builder and general handling of web server in controller, refer to the *System 800xA Control AC 800M Configuration (3BSE035980*) Manual*

The login screen for authentication is displayed as shown in **Figure 35**. Enter the user name and password to login and access the home page of the web server. The default user name is **service** and the password is **ABB800xA**. To change the default password, refer to Change Password on page 105.

![Figure 35. CI871 Webserver Login Authentication](image-url)
The default password is **ABB800xA**. Replace this with a password that conforms with your organization's security policy at the first possible opportunity. Failure to replace the default password makes the system susceptible to unauthorized access.

Refer to the Password Security topic in *System 800xA Administration and Security (3BSE037410*)* for recommendations on establishing a password security scheme.

After firmware update to system version 6.0, the PM8xx controller password must be configured again, as the previously configured password is cleared and replaced with the new default password.

If three unsuccessful login attempts occur in a minute, the webserver is locked and can be logged in only after ten minutes.

After launching the web server, select the required CI871 from the **Cex slot** drop-down list as shown in Figure 36.

![Figure 36. Web Server Interface for CI871 Service Diagnosis](image)

The web server page is active only for 4 hours from the first login. A timer is set for 4 hours and during this limited session time, an information on the remaining time before the web server disables is shown in the browser.
When the limited activation time has expired and if the web server page is still open, a message *The web server is disabled, it can only be reenabled externally* is shown. To continue using the web server, enable it through Control Builder.

![Figure 37. Web server Disabled](image)

Table 10 lists the Alarms, Events and Warnings recorded in web server. These are displayed in Plant Explorer and Hardware Status of Control Builder.

**Table 10. Web server Alarms and Events**

<table>
<thead>
<tr>
<th>Message</th>
<th>Type</th>
<th>Severity</th>
</tr>
</thead>
<tbody>
<tr>
<td>The web server is enabled, but the password has not been changed from default.</td>
<td>Alarm</td>
<td>Medium</td>
</tr>
<tr>
<td>The web server has been enabled.</td>
<td>Event</td>
<td>Low</td>
</tr>
<tr>
<td>The web server has been disabled.</td>
<td>Event</td>
<td>Low</td>
</tr>
<tr>
<td>The web server timer has been reset.</td>
<td>Event</td>
<td>Low</td>
</tr>
<tr>
<td>The controller password has been changed from.</td>
<td>Event</td>
<td>Low</td>
</tr>
<tr>
<td>Successful login to the controller from &lt;IP&gt;.</td>
<td>Event</td>
<td>Low</td>
</tr>
<tr>
<td>Unsuccessful login attempt to the controller from &lt;IP&gt;.</td>
<td>Event</td>
<td>Medium</td>
</tr>
<tr>
<td>Unsuccessful attempt for controller password change from &lt;IP&gt;.</td>
<td>Event</td>
<td>Medium</td>
</tr>
<tr>
<td>The web server has been enabled.</td>
<td>Audit Trail</td>
<td>N/A</td>
</tr>
</tbody>
</table>
**Cl871 Web Server Interface**

The web server interface can be accessed through a web browser. To open the web server interface, enter the IP address of the controller in the address bar of the browser and press <Enter>.

The browser window has two frames. The left frame displays the menu items. Select the menu item on the left to display the respective contents in the content frame on the right.

The view of the menu is function specific. For certain menu items, only a simple view of the available data is displayed. Some menu items need an additional input before the corresponding data is displayed. An input dialog appears on the top of the content frame for these menu items.

*Figure 38. Functional overview of the Service Diagnosis*
CI871 Web Server Interface Menu Items

CI871 web server interface displays the following menu items:

- Errormemory
- Version Info
- LifeList
- Set Devicename
- Get I&M by name
- Get I&M by ARID
- Let Device blink
- Journal Buffer
- CPU Load
- Current Time
- General Status
- Debug...

Following chapters explains some of the important functions of the CI871 web server:
Errormemory

This information is relevant only for an ABB service engineer in case of an error. Figure 39 shows an example of Error Memory details.

Figure 39. Error Memory of CI871
Version Info

Select **Version Info** to display the version information of the firmware and the communication interface CI871 as shown in Figure 40.

<table>
<thead>
<tr>
<th>Part name</th>
<th>Date</th>
<th>Version</th>
</tr>
</thead>
<tbody>
<tr>
<td>CI871FW</td>
<td>2013-11-20</td>
<td>3.5.2</td>
</tr>
<tr>
<td>CPU Chip</td>
<td></td>
<td>0700</td>
</tr>
<tr>
<td>HW revision</td>
<td></td>
<td>CI871 A</td>
</tr>
<tr>
<td>FFPGA(15)</td>
<td></td>
<td>0.1/0 B0</td>
</tr>
<tr>
<td>Cirrus</td>
<td>2010-12-20</td>
<td>5.1.0014.0000</td>
</tr>
</tbody>
</table>

*Figure 40. Version Information Dialog*
LifeList

Select **LifeList** to list the devices that are connected to PROFINET IO as shown in Figure 41.

![Figure 41. LifeList](image)

The PROFINET IO devices are listed and sorted by name along with the MAC address and the IP address.

LifeList displays only the PROFINET IO devices which are connected to the PROFINET IO and which support for DCP protocol. These IO devices are independent of configuration and cyclic communication status.
Set Devicename

Select **Set Devicename** to set the symbolic name for the PROFINET IO device.

To set a unique station name for the PNIO device in the Control Builder, refer to **Station Name** on page 74.

Selecting this item, displays the parameter dialog **DCP Set Name**. Enter the Media Access Control (MAC) address of the PROFINET IO device to assign a new device name. Use the following syntax:

\[ <\text{MAC Address}> \text{ blank } <\text{Devicename}> \]

Check the LifeList to verify the available devices and MAC addresses.

Click **OK** to assign a new device name, as shown in **Figure 42**.

Verify if the device is displayed in the LifeList with the correct device name. For more information, refer to **LifeList** on page 96.

**StackAccess**

**DCP Set Name**

Parameter:

```
00:a0:45:04:1b:de devicename
```

**OK**

**DCP_Set-Operation successful done**

Target MAC-Address: 00:a0:45:04:1b:de

**Figure 42. Set Devicename Successful**
If an error occurs, a dialog containing the error details are displayed as shown in Figure 43.

**StackAccess**

**DCP Set Name**

Parameter:

00:a0:45:04:1b:de Devicename

OK

Label consists of other signs than [a-zA-Z],[0-9],[-]: Devicename

*Figure 43. Set Devicename with Error*
I&M Data

Select **Get I&M by name** to display the I&M data. Use the Identification and Maintenance (I&M) data to get more information for a specific PROFINET IO device. For example, important information such as the software and hardware revision of the device are displayed.

Enter the name of the device in the **Parameter** field and click **OK**. **Figure 44** shows an example of I&M data for a specific PROFINET IO device.

![I&M Data Dialog](image)

**Figure 44. I&M Data Dialog**
Let Device blink

I&M data are displayed for PROFINET IO devices that are successfully configured in CI871.

Specify the name of the device with the correct syntax. For more information on assigning station name, refer to Basic Communication Settings on page 74.

Let Device blink

Select Let Device blink to identify a PNIO Device in the network. To enable the PNIO Device to blink, select Let Device blink from the menu. Enter the MAC address of the device in the Parameter field and click OK. The PNIO Device with the corresponding MAC address will blink for a few seconds (4-5 seconds).

Figure 45. Let Device blink Dialog
Journal Buffer

Select **Journal Buffer**, to access the Journal Buffer. This information is relevant only for an ABB service engineer in case of an error. The Journal Buffer provides useful information in addition to the Error memory.

![Figure 46. Journal Buffer of CI871](image-url)
CPU Load

Select **CPU Load**, to display the current value of the CI871 CPU load. The value is displayed in percentage. **Figure 47** shows an example value of 15% for the CPU load.

![Example of CPU Load](image)

**Figure 47. CPU Load of CI871**
General Status

Select **General Status**, to display the current status of all the PNIO devices connected to the CI871 along with their IP address, Vendor ID, and Update time for inputs and outputs,

![General Status of CI871](image_url)

*Figure 48. General Status of CI871*
Service File

Select **Create service file**, to create a service file for maintenance action. The web server interface automatically compiles all the required information into a service file, in case of a support action. This file contains all the information that is accessible through the function list instead of device specific data.

Select the required CI871 modules for displaying the corresponding data. Select all the modules, to display the information relevant to all the modules.

![Image of CI Module Selection for Service File]

**Figure 49. CI Module Selection for Service File**

Click **Create**. A **File Download** window appears with the options to **Open** or **Save** the file to the hard disk.

- If the user has cleared the check box **Always ask before opening this type of file** during the previous file download then the service file is opened in Notepad.
- Select **Create service file** to generate the **Error Memory** information as a text file.
Change Password

The controller password should not be changed when several users are using the web server at the same time. If several web browser pages are connected to the web server, close all except for one before changing the password.

The Change Password page is used for changing the default password of the web server. The web server page displays a caution to change the default password, when a user login for the first time as shown in the Figure 50. The caution is displayed for every login, until the default password is changed to a new password.

After the controller password has been changed through web server page, a login window appears, you need to login again with the new password.

To change the password from default, select the Change password option available at the top left corner of the browser window. Enter the current and the new password and confirm the new password.

Figure 50. Web Server Change Password
A successful operation is confirmed with a message *Change password succeeded.* The new password is active immediately and stored nonvolatile in the AC 800M controller.

**Reset Default Password**

Use the IP Config Tool to reset the controller password to default password. Perform the following to reset the default password:

Ensure that the COM port of the **PM8xx** controller is connected to a COM port of the computer through a serial cable (If it is a **PM86x** controller, the port of the controller to be connected is the marked **COM4**.). In the IP Config tool select the COM port of the computer that is connected to the controller.

1. Open **IP Config** and select **Settings** menu.

   From the **Com Port** context menu, select the corresponding Com port of the computer. In this example, **Com1** is selected as shown in **Figure 51**. After selecting the COM port, click **Connect**.

   ![IP Config Tool - COM Port Settings](image)

   **Figure 51. IP Config Tool - COM Port Settings**
IP Config tool can be started only after a long reset of the controller. Later a fresh
download is performed, as the configuration is erased during reset.

2. In the Misc menu, click **Reset Controller Password**.

![Figure 52. IP Config Tool - Reset Controller Password](image)
3. A confirmation window, *The Controller password has been reset* appears.

![Figure 53. Reset Password - Confirmation Window](image-url)
Section 6  Technical Data and Performance

For PROFINET IO configurations in 800xA with CI871 the following dimensioning guidelines need to be taken into account:

- Up to 12 CI871 per AC 800M controller.
- Up to 126 PNIO devices per CI871.
- Up to 512 modules per PNIO device.
- One IOCR for each direction (Input and Output) per PNIO device, each IOCR up to 1440 bytes of I/O data.
- Update times down to 1 ms (only if CI871 has one device configured).
- For CPU-load calculation of CI871 the Ethernet frames for inputs and outputs need to be calculated. CI871 can handle as a maximum one frame per ms in each direction.
  Example 1: Update times for all devices is configured to 32 ms (default), then up to 32 devices can be connected to CI871.
  Example 2: Update times for all devices is configured to 8 ms, then up to 8 devices can be connected to CI871.
- Limitations for acyclic communication:
  CI871 can handle up to 25 transactions/second for a payload of 60 bytes per read/write operation causing a CPU load of about 1% on CI871 and 0,3% on PM866 for each transaction.
  The maximum framelenlength is defined by 4 KBytes for each service. Longer frames than 60 bytes cause a slightly increased CPU load.
The limitation for the CPU load of CI871 is checked by the system during download. If the system detects that there is an CPU overload, then it is indicated in the Compilation Summary window and the download is blocked. The CI871 may not function properly when there is an overload. The user can check the CPU load before and after download by use of the web server interface. Refer to CPU Load for information on how to read it. The limit for the CPU load is 100%. Up to that value the CI871 works stable without any problems or restrictions.

The CPU load for acyclic communication cannot be automatically checked by the system during download. CI871 is able to handle up to 10 acyclic transactions/second and also stable in a high load scenario, In case a higher acyclic communication load is required the user needs to take care that the resulting CPU load is in a safe area e.g. below 70%.
Section 7  Device Import Wizard

The functionality of a PROFINET IO Device is described in a General Station Description (GSD) file. The GSD file contains all the data that are relevant for engineering and data exchange with the PNIO Device.

This section describes the Device Import Wizard, which is used to parse the PROFINET IO GSD files and converts it into hardware library. The hardware library is integrated in Control Builder, which is required to configure the PNIO device in 800xA.

The Wizard supports the following capabilities along with the conversion of a GSD file to a hardware library:

1. Select DAPs, modules, and submodules to be included in the hardware library.
2. Select Device and their corresponding version.
3. Modify properties of I/O Devices (Names, Channels, Datatypes, Icons).
4. Modifying parameters.
5. Converting an Integer datatype to a RealIO (Also called ADV Format. For more details, refer to Advanced ADV Settings with status on page 131).

If it is not required to modify any settings other than the those provided in the GSD file then select the required hardware types and click the Next button available at the bottom of each window of the Device Import Wizard to complete it and insert the hardware library.

The hardware library is required to integrate and configure the PNIO Devices in the Control Builder. The wizard contains a PROFINET IO Parser which converts the GSD file into a hardware library. The wizard supports a re-import, in cases where the existing hardware library needs to be modified, so that the user can continue the import with the previous configured settings.
Importing GSD File

The following are the sequence of steps to be followed to convert a GSD file to hardware library:

1. Create a new hardware library.
2. Start the wizard.
3. Configure the hardware types.

Create a New Hardware Library

1. Open the Control Builder.
2. Select Libraries > Hardware.
3. Right-click Hardware and select New Library. The New Library pop-up window is displayed.

Figure 54. New Library Window
Enter a library name in the **Name** field and click **OK**. The user defined library is created under the hardware tree as shown in **Figure 55**.

**Figure 55. Created Hardware Library**

4. Expand the newly created library. Right-click on **Hardware types** and select **Insert/Replace Hardware Types**.

**Figure 56. Insert/Replace Hardware Types**

The Insert Hardware Definitions window appears as shown in **Figure 57**. Set the file type as PROFINET (GSDML*.xml) and select the GSD file to be converted and click **Open**.
Start Wizard

1. The Wizard welcome dialog is displayed as shown in Figure 58. This dialog displays the details of the GSD file and the device type which is imported.

*Figure 57. GSD File Selection Dialog*
2. Click Next. The PROFINET GSD file import - Device Information dialog is displayed. This dialog reads the information from the GSD file and displays the basic information about the device like IDs and schema version of the GSDML. The Start value for I/O channels helps the user to specify whether the channel numbering in Control Builder should begin with the number 1 or 0. The value selected in the Start value for I/O channels is displayed in the Control Builder hardware tree along with the name of the first channel. The consecutive numbers are displayed along with the names of the subsequent channels.

The number selected in the drop down should be identical to the channel number of the I/O module. To avoid any discrepancies between the I/O channel numbering and the number displayed in Control Builder, user has to check for the device manual (in case of a remote I/O it is the front cover) to select a 1 or 0 from the drop down.
Figure 59. Device Information Dialog
3. Click **Next**. The **PROFINET GSD file import - hardware types selection** dialog is displayed.

![PROFINET GSD file import - HW types selection](image)

*Figure 60. Device Import Wizard Hardware Type Selection*

The hardware types selection dialog displays all the module information for the PROFINET IO device which is collected from the GSD file. This is the main window of the Wizard where all the configuration is done.
Configure Hardware Types

Hardware Tree Context Menu

Figure 61 shows an item selected in the tree view along with a context menu.

![Context Menu Image]

Figure 61. Navigating Hardware Tree using Context Menu

Table 11 describes the list of options available in the context menu for the selected device object.

Table 11. Hardware Tree Context Menu

<table>
<thead>
<tr>
<th>Menu Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Select All</td>
<td>Helps to select the modules and submodules within the module.</td>
</tr>
<tr>
<td>Deselect All</td>
<td>Helps to deselect all the modules and submodules within the module.</td>
</tr>
<tr>
<td>Expand or Collapse</td>
<td>Based on the current state of the module, the context menu displays.</td>
</tr>
<tr>
<td></td>
<td>• Expand</td>
</tr>
<tr>
<td></td>
<td>Helps to display the expanded view of all the submodule within the module.</td>
</tr>
<tr>
<td></td>
<td>• Collapse</td>
</tr>
<tr>
<td></td>
<td>Makes all the submodules to be hidden within the module.</td>
</tr>
</tbody>
</table>
### Table 11. Hardware Tree Context Menu

<table>
<thead>
<tr>
<th>Menu Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Go to next selected HW type</td>
<td>Helps to navigate between the selected DAP/module/submodule within the tree view.</td>
</tr>
<tr>
<td>Go to previous selected HW type</td>
<td></td>
</tr>
<tr>
<td>GSDML view</td>
<td>Displays the XML view of the GSDML file of the selected modules or sub modules as shown in Figure 62.</td>
</tr>
<tr>
<td>Show selected items</td>
<td>Displays the tree view of only the selected modules and sub modules in a Selected modules window as shown in Figure 63. The Show selected items option can be used for large gsd files with a lot of hardware types. The Selected modules navigation window makes the navigation very easy and also user does not have to scroll (search) in a long list.</td>
</tr>
</tbody>
</table>
Click **Parent XML node** to select the parent XML node of the selected module.

The XML structure of the entire GSDML file is shown step by step, by clicking on Parent XML node.

![GSDML View](image)

*Figure 62. GSDML View*
The **Selected modules** window shows a tree view of already selected modules and submodules. It can only be used for navigation purposes (to select a module or submodule). Each module or submodule which is selected in the right window is also selected in the left tree view and vice versa. When a module or submodule is deselected in the left tree view it disappears in the right window and when selected, it also appears in the right window.

![Selected Modules](image)

**Figure 63. Selected Modules**

**Hardware Types Selection**

The **HW type selection** window contains a tree view on the left which lists the devices and their corresponding modules, and submodules. The right pane of the window contains the Input, Output, and Parameter tab. The display of contents in these tabs depends on the selection made in the tree view.

The highest level of Hardware tree are the devices. Each device contains a set of applicable modules. Each module contains a set of applicable submodules. The GSD file can contain several devices and multiple versions of a device as shown in **Figure 64**.
The hardware view is also used to navigate between the devices, modules, submodules. Selecting the check box (create a check mark) makes the device to be included in the hardware library. It is possible to modify the name and description for the device, module and submodule if the GSD file does not follow a proper naming convention or if the user wants to customize the names to make it user friendly. To modify the name or description, select the respective module or submodule and edit the text field for **Name, Description** in the **HW type information** pane as shown in Figure 65. To select the icon to be used for a particular submodule, use the **Icon** drop-down menu. The icon selected in this window is displayed in the Control Builder for each module.
Modules or Hardware types which are selected are indicated by a check mark and only these are included in the hardware library. The selections are indicated in various ways in the **HW types selection** navigation pane.

- Indicates that some of the underlying modules are selected.
- Indicates that all the underlying modules are selected.
- Indicates that none of the underlying modules are selected.

Some DAPs/modules are grayed out in the navigation pane and cannot be deselected. These are the modules which are defined as **FixedInSlots** in GSD file.

Click the + sign in the hardware tree to expand the modules as shown in **Figure 66**. If the sign changes to – it means the module cannot be further expanded.

![HW types selection](image)

**Figure 66. Device with Modules and Submodules Dialog**

There are various buttons in the window which helps to select the modules.
Click **Select All** to select all the modules in the GSD file.

Click **Select none** to clear all the selected modules. This prompts for a confirmation as shown in Figure 67.

![Figure 67. Select None Confirmation Dialog](image)

**Input Tab**

The Input and Output Tab (shown in Figure 68) displays the details of the default configuration available in the GSD file for the particular submodule selected in the hardware tree view. The input channels are created in the hardware library based on the configuration displayed in the Input tab.

If the user does not want to modify the default settings then select the required hardware types and click the **Next** button available at the bottom of the window (Figure 68) to proceed.

Channel numbers in the Control Builder depend on the order in which the I/O channels are displayed in the Input tab. If the user wants to change the numbering order then it is done by moving the I/O channels using a drag and drop option with the help of the mouse.
The PROFINET IO frame is displayed on the left of the tab. The Channel related details are available on the right of the tab. Figure 68 displays the Input tab for a particular submodule.

![Figure 68. Input Area](image)

The table on the right shows the textual description of the defined I/O channels. These details include:

- **Byte, bit** - displays the starting bit of the I/O channel. The allocation of Byte and bit can be modified with the help of the graphical display of the PNIO frame provided on the right pane of the input tab. The number of rows available in the graphical display depends on the maximum number of bytes for the particular submodule. For example, if the submodule can hold a maximum 4 bytes of data then the Input Tab is displayed as shown in Figure 69. Each bit in PNIO is configured twice for the IO channels. For more information on assigning the bits twice to the I/O channel, refer to Creating I/O Channels on page 139.
If a particular channel is selected in the Input tab then the corresponding bits are highlighted in the PNIO graphical display frame (Figure 69) with the datatype specific color as shown in Figure 70.

- **Length** - displays the number of bits allocated for each I/O channel.
- **Name** - displays the name of each I/O channel. The Input channels are created in the hardware library with the names displayed on this tab.
- **Data type conversion** - displays the copy function which is generated in the hardware library file for each I/O channel.
- **Features** - displays the specific features or properties of the I/O channel like byte swapping, ADV conversion. The availability of a specific feature is indicated by an icon and different datatypes have different features.

The byte swapping feature is available only for specific copy functions.
Byte Swapping

The Byte Swapping feature is used for swapping bytes for specific I/O channels. Channels which support byte swapping are displayed with an icon 🔄 in the features column of the Input tab. By default, the frame format used in PNIO is **Big endian** where the MSB is read first and followed by the LSB. There are some PNIO devices which deliver data where LSB is preceded by MSB, these are devices which follow the **Little endian** format. In such cases, use the Bytes swapping feature to swap the bytes which facilitates the Device Import Wizard to read the bytes in the correct order. For example, Figure 71 displays that the channel 2,7 supports byte swapping.

![Figure 71. Byte Swapping Feature](image-url)
To swap the bytes:

1. Double-click 🔄. The Frame format dialog is displayed.

![Frame format dialog]

*Figure 72. Frame Format*

2. Select the frame format to be applied. By default, **Big endian** is selected.

3. When **Big endian** is selected, the bytes are not swapped.

4. When **Little endian** is selected, the bytes are swapped.

5. Select **OK**. The feature icon changes to green, that is the byte swapping feature is applied to the I/O channel as shown in *Figure 73*. 
Figure 73. I/O Channel with Byte Swapping applied

Figure 74 displays how the frame structure is handled after Byte Swapping in case of I/O channel having Two Bytes.
Figure 74. Example of Byte Swapping in Two Bytes Frame

Figure 75 displays how the frame structure is handled after Byte Swapping in case of I/O channel having **Four Bytes**.

Figure 75. Example of Byte Swapping in Four Bytes Frame
Advanced ADV Settings with status

In PNIO, Integer value can be mapped to RealIO. The Advanced settings can be configured for channels having a data type value as \texttt{Int16+IOPS=>Real}.

The ADV feature dialog offers the various possibilities like:

- Assigning scaling parameters so that no scaling is required in the IEC 61131-3 Application.
- To modify the range, for example changing 0..20mA to 4..20mA which does not affect the IEC 61131-3 application.
- Byte swapping.

A sequence of steps is required to be followed to map the Integer value to a RealIO value.

\textbf{Figure 76} shows the default configuration available in the Wizard.

\textbf{Figure 76. I/O channel with Integer Datatype}

- Delete the channels present in the input tab displayed in \textbf{Figure 76}. For further information on deleting I/O channels, refer to Filtering I/O Channel on page 138.
- Create new channels with 16 bits with the datatype \texttt{Int16+IOPS=>Real}. For more information on creating channels, refer to Creating I/O Channels on page 139.
Right-click on the channel in the input tab and select **Advanced Edit**. A dialog containing the following tabs is displayed as shown in *Figure 78*:

- **Range Parameter**.
- **Frame format**.

**Range Parameter**

The Range Parameter tab contains all the parameters the device supports with their scaling parameters like datatype, Byte Offset, Value, and Description. Select the range parameter related to that configured channel and configure the connection inside HWD between the I/O Channel and the settings. By that the EnumRange datatype is created for the settings parameter (previously it was Enum datatype).
Figure 78. Advanced ADV Settings with status - Range Parameter

The Figure 78 displays some default configurations. For each EnumValue the user has to configure the digital value of the A/D-converter with minimum and maximum value. Typically these values are available in the user manual provided along with the device.
Figure 79 displays the Voltage measuring ranges ±5V and ±10V for an analog input module of Siemens ET 200S.

<table>
<thead>
<tr>
<th>Measuring range ±5 V</th>
<th>Measuring range ±10 V</th>
<th>Units</th>
<th>Decimal</th>
<th>Hexadecimal</th>
<th>Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>&gt; 5.8794</td>
<td>&gt; 11.7589</td>
<td></td>
<td>32767</td>
<td>7FFFh</td>
<td>Overflow</td>
</tr>
<tr>
<td>5.8794</td>
<td>11.7589</td>
<td></td>
<td>32511</td>
<td>7EFFh</td>
<td>Overshoot range</td>
</tr>
<tr>
<td>:</td>
<td>:</td>
<td></td>
<td>:</td>
<td>:</td>
<td></td>
</tr>
<tr>
<td>5,0002</td>
<td>10,0004</td>
<td></td>
<td>27649</td>
<td>6C01h</td>
<td></td>
</tr>
<tr>
<td>6.00</td>
<td>10.00</td>
<td></td>
<td>27648</td>
<td>6C00h</td>
<td>Nominal range</td>
</tr>
<tr>
<td>3.75</td>
<td>7.50</td>
<td></td>
<td>20736</td>
<td>5100h</td>
<td></td>
</tr>
<tr>
<td>:</td>
<td>:</td>
<td></td>
<td>:</td>
<td>:</td>
<td></td>
</tr>
<tr>
<td>-3.75</td>
<td>-7.50</td>
<td></td>
<td>-20736</td>
<td>A800h</td>
<td></td>
</tr>
<tr>
<td>-5.00</td>
<td>-10.00</td>
<td></td>
<td>-27648</td>
<td>5400h</td>
<td></td>
</tr>
<tr>
<td>-5,0002</td>
<td>-10,0004</td>
<td></td>
<td>-27640</td>
<td>03FFh</td>
<td>Underrange</td>
</tr>
<tr>
<td>:</td>
<td>:</td>
<td></td>
<td>:</td>
<td>:</td>
<td></td>
</tr>
<tr>
<td>-5,8796</td>
<td>-11,759</td>
<td></td>
<td>-32512</td>
<td>8100h</td>
<td></td>
</tr>
<tr>
<td>&lt; -5,8796</td>
<td>&lt; -11,759</td>
<td></td>
<td>-32766</td>
<td>9000h</td>
<td>Underflow</td>
</tr>
</tbody>
</table>

Figure 79. Voltage measuring ranges: ±5 to ±10

Figure 80 displays the settings entered for minimum and maximum values as per the Siemens ET200S device.

Figure 80. Min, max, overflow, and underflow applied
Frame format

Frame Format tab supports swapping of bytes. For more information on Byte Swapping, refer to Byte Swapping on page 127.

![Frame Format](image)

Figure 81. Advanced ADV Settings with status - Frame Format

To Modify I/O Channels

The I/O channel Name can be modified within the I/O area.

To rename a channel name, double-click the required row and modify the details.

To rename multiple I/O channel rows hold down the CTRL key or SHIFT key and select the required I/O channels as shown in Figure 82.
1. Right-click on the selected I/O channel rows and select **Rename (Batch job)** which displays a dialog as shown in **Figure 83**.

![Figure 82. Selected I/O channels for modification](image)

![Figure 83. I/O Channel Renaming dialog](image)

2. Enter the channel name in **Prefix**.
3. Specify the number to be added with the channel name in **Start with** and click **OK**.
If there is a duplication in I/O channel naming, then a pop-up message is displayed as shown in Figure 84.

![Collision Error Message]

**Figure 84. Collision Error Message**

To modify the name of an I/O channel, double-click the specific row and edit the name.

![I/O channels after modification]

**Figure 85. I/O channels after modification**
Deleting I/O Channel

To delete an I/O channel:

1. Right-click the required I/O channel row. This displays the context menu as shown in Figure 86.

![Figure 86. I/O channel selected for Deletion](image)

2. Select **Delete** from the context menu. The I/O channel slots become empty as displayed in Figure 87.

![Figure 87. Deleted I/O Channels](image)

To delete a set of I/O channels, press **CTRL** key or **SHIFT** key and select the required number of rows on the right. Right-click the selected rows and select **Delete**.
Creating I/O Channels

I/O channels can be created for the unused channels and within the I/O area.

To create I/O channels:

1. Select the empty bits or select the blue triangle to select the entire unused byte. A context menu with the permissible data types for the selected I/O channel is displayed as shown in Figure 88.

Figure 88. I/O Channel Creation

PROFINET IO contains different types of copy functions. A set with IOPS and a set without IOPS. By default IOPS is used, which helps to indicate the channel errors. For more details on IOPS, refer to PROFINET IO Data Exchange Quality on page 33.

The following are the datatypes supported by AC 800M:

- RealIO.
- DintIO.
- DWORDIO.
• BOOLIO.

2. Select the required data type conversion from the context menu. The created channels are displayed at the bottom of the channel details table on the right.

```
<table>
<thead>
<tr>
<th>Input</th>
<th>Output</th>
<th>Parameter</th>
</tr>
</thead>
<tbody>
<tr>
<td>7 6 5 4 3 2 1 0</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
```

The I/O area positions are used exactly for one channel. Each byte/bit in the PNIO frame can be accessed only through one copy function. But, there is one exception to this rule. The I/O channels created using the data type **Bit Boolean** can be assigned with another data type also.

Each bit can be assigned twice to the I/O channels in the following ways:

- A bit can act as a single entity and can be assigned as **1 Bit => Bool**.
- The same bit can be grouped along with other bits to form a datatype which can be used for some other copy function.

**Figure 89. I/O Channels created**

The I/O area positions are used exactly for one channel. Each byte/bit in the PNIO frame can be accessed only through one copy function. But, there is one exception to this rule. The I/O channels created using the data type **Bit Boolean** can be assigned with another data type also.

Each bit can be assigned twice to the I/O channels in the following ways:

- A bit can act as a single entity and can be assigned as **1 Bit => Bool**.
- The same bit can be grouped along with other bits to form a datatype which can be used for some other copy function.

**Figure 90** shows an example of an 8 bit input channel where each bit is assigned with an input channel **1 Bit => Bool** and the same 8 bits are grouped together and assigned to an input channel with a different datatype. A grouped channel gives a better performance than the single boolean channels. There is also a disadvantage from engineering perspective as different boolean I/O channels do not get a unique name.
User can access the cyclic data on PNIO level only once as a separate boolean channel or as a grouped channel. For example, in the Figure 90 user has the option to access the data either through a 8 single boolean channel or through a DINT channel.

User is not provided with any error or warning if both channels are accessed or configured in the Control Builder. An error is prompted to the user only during download in the compilation summary window.

The support to access three channels through single boolean channel and the other five through DINT is not available in PNIO.
Output Tab

The Output tab provides the details of the output channels for the selected submodule. The bit and byte details are available on the left. The Channel details are available on the right. Figure 91 displays the Output tab for a particular submodule.

![Image of Output Tab](image)

Figure 91. Output Area

The details available in the Output tab are identical to the details available in the Input tab.

IOPS, Real+Status, and ADV Format are handled for Input channels only and excluded for Output channels.

For additional information on the I/O channel details and other functionalities, refer to Input Tab on page 124.
Parameter Tab

The Parameter tab displays the parameters available for a PROFINET IO device. These parameters are defined in the GSD file and are used for configuring the PROFINET IO devices.

Under normal circumstances it is not necessary to modify default parameter settings. If the GSD file is not correct then some modifications are required.

Modifying Default Parameters

Execute the following steps to modify the PROFINET IO device parameters:

1. Select the Parameter tab. The parameters described in the GSD file are displayed as a tree view as shown in Figure 92.

![Figure 92. Parameter Tab]
2. Select the required parameter in the tree view. The settings for this parameter are displayed on the right as shown in Figure 93.

![Diagram showing Parameter Tab with Details](image)

*Figure 93. Parameter Tab with Details*

3. Modify the parameter settings as required. Parameters can be
   - Changed.
   - Deleted.
   - Added.
To configure Parameters, right-click on the parameter tree to show the context menu and display the options available at each level as shown in Figure 94.

![Parameter - Context Menu](image)

**Figure 94. Parameter - Context Menu**

User can configure the parameter using the options available in the context menu. However, depending on the type of parameter selected some of the user control fields are disabled for editing.
Create Hardware Types

1. Click Next when necessary configurations are done. The comments dialog is displayed as shown in Figure 95.

![Figure 95. Comments Dialog](image)

By default the comments dialog contains the timestamp of the last modified date. If required, additional notes can also be added which is used as logbook for future re-import.
2. Click **Next**. The Conversion results are displayed. The dialog provides a summary of all the errors and warnings encountered during the conversion process.

![Device Import Wizard]

*Figure 96. Conversion Results Dialog*

One can continue Wizard even if the conversion results displays any warnings, but in case of any errors the device might not work properly.
3. Select **View/Print HWD** to view the content of HWD file.

![HWD Viewer dialog](image)

**Figure 97. View/Print HWD dialog**

4. Click **Finish**. The configured HWD file as added to the hardware library.

The configured hardware library is displayed in the Control Builder as shown in **Figure 98**.
Append GSD files to a Hardware Library

User can import multiple GSD files into a hardware library. To insert additional GSD files into a hardware library:

1. Right-click on **Hardware types** and select **Insert/Replace Hardware Types** as described in **Step 4 in Create a New Hardware Library** on page 112.
2. User will be prompted with a dialog as shown in Figure 99.

![Import File dialog](image)

*Figure 99. Importing multiple GSD files into a hardware library*

3. Select **Yes** to proceed and append the GSD into the hardware library.

   User can select only one GSD file for import. Multiple selection of GSD files is not supported.

### Re-importing the GSD file

After successful creation of the hardware library it might be required to add additional modules or modify/append a few parameters or settings to the existing configuration. When a parsed GSD file is imported into the Wizard it retains all the configuration that were set during the last conversion process.

To re-import the GSD file right-click on the hardware library and select **Properties>Files** to display the dialog as shown in Figure 100.

Select the GSD file for which re-import needs to be done and click on the **Wizard** button. Click the **Wizard** button to display the Device Import Wizard. These are the same navigation windows as described in **Start Wizard** on page 114. User can add additional modules or configure settings if required during this process.

In case the user deselects the modules/submodules that were already imported, then the modules/submodules will be deleted from the hardware library if they are not configured as any instances.
Figure 100. Hardware library showing multiple GSD files and option for reimport
Section 8  Controller/Controller Communication

This section describes the process of data transmission between two PROFINET IO Controllers.

PN/PN Coupler

The PN/PN Coupler enables fast, simple, deterministic I/O data coupling between two PNIO Controllers.

The PN/PN Coupler can be used for fast peer to peer communication between two controllers irrespective of their device functionality. The data can be transferred between PNIO networks (see Figure 101) which are configured with different controllers like AC 800M and Siemens S7 300. The maximum size of the data which can be transferred is 1044 bytes input data and 1044 bytes output data with an update time of 1 ms. If the transfer of data exceeds the limit (1044 byte) then the user has to configure more than one PN/PN coupler.

For acyclic communication, the PN/PN coupler supports up to 64 KBytes of data transfer, since it supports up to 16 slots each with 4 KBytes. This acyclic data record transfer exceeds the cyclic I/O data transfer of 1 KBytes input and output data.
The GSD file for the PN/PN Coupler can be used to integrate the PN/PN Coupler using the Device Import Wizard.

**Data Transfer**

The PN/PN coupler works as a device between the PNIO networks. Transfer of data from one PNIO network to the other PNIO network is done through the PN/PN Coupler. Figure 102 shows process of data transfer internally between the I/O modules having 4 bytes and 2 bytes by a PN/PN Coupler.
PNIO Controller 1 has to have a module with 2 bytes to hold the data transferred by the module of PNIO controller 2. Similarly PNIO controller 2 has to have a module with 4 bytes to hold the data transferred by PNIO controller 1 as shown in Figure 103.
Figure 103. Module specification for data transfer
Appendix A CI871 Error Codes

The error codes in Table 12 and Table 13 are used for acyclic communication:

*Table 12. Error Codes related to CI871*

<table>
<thead>
<tr>
<th>Error Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-7160</td>
<td>Data frame too long, more than 4096 bytes used</td>
</tr>
<tr>
<td>-7161</td>
<td>PNIO Stack cannot process the read or write request or CI871 received invalid message type in the request from the controller</td>
</tr>
<tr>
<td>-7162</td>
<td>Timeout occurred while waiting for a response from CI871</td>
</tr>
<tr>
<td>-7163</td>
<td>Invalid response from CI871, wrong transactionID, index or response type</td>
</tr>
<tr>
<td>-7164</td>
<td>Response from CI871 contains an implausible internal error code</td>
</tr>
<tr>
<td>-7165</td>
<td>Communication overload, too many transactions active</td>
</tr>
</tbody>
</table>
The error codes explained in the following table are defined by PROFINET standard and belong to the PNIO device:

The term User specific mentioned in the table is device related and is documented in the device specific manual.

*Table 13. Error Codes from PNIO Device*

<table>
<thead>
<tr>
<th>Error Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-7424</td>
<td>No PROFINET specific error code</td>
</tr>
<tr>
<td>-7264</td>
<td>Read error</td>
</tr>
<tr>
<td>-7263</td>
<td>Write error</td>
</tr>
<tr>
<td>-7262</td>
<td>Module failure</td>
</tr>
<tr>
<td>-7261</td>
<td>Not specified</td>
</tr>
<tr>
<td>-7260</td>
<td>Not specified</td>
</tr>
<tr>
<td>-7259</td>
<td>Not specified</td>
</tr>
<tr>
<td>-7258</td>
<td>Not specified</td>
</tr>
<tr>
<td>-7257</td>
<td>Busy</td>
</tr>
<tr>
<td>-7256</td>
<td>Version conflict</td>
</tr>
<tr>
<td>-7255</td>
<td>Feature not supported</td>
</tr>
<tr>
<td>-7254</td>
<td>User specific 1</td>
</tr>
<tr>
<td>-7253</td>
<td>User specific 2</td>
</tr>
<tr>
<td>-7252</td>
<td>User specific 3</td>
</tr>
<tr>
<td>-7251</td>
<td>User specific 4</td>
</tr>
<tr>
<td>-7250</td>
<td>User specific 5</td>
</tr>
<tr>
<td>-7249</td>
<td>User specific 6</td>
</tr>
<tr>
<td>-7248</td>
<td>Invalid index</td>
</tr>
<tr>
<td>-7247</td>
<td>Write length error</td>
</tr>
</tbody>
</table>
Table 13. Error Codes from PNIO Device (Continued)

<table>
<thead>
<tr>
<th>Error Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-7241</td>
<td>Invalid range</td>
</tr>
<tr>
<td>-7240</td>
<td>Invalid parameter</td>
</tr>
<tr>
<td>-7239</td>
<td>Invalid type</td>
</tr>
<tr>
<td>-7238</td>
<td>Backup</td>
</tr>
<tr>
<td>-7237</td>
<td>User specific 7</td>
</tr>
<tr>
<td>-7236</td>
<td>User specific 8</td>
</tr>
<tr>
<td>-7235</td>
<td>User specific 9</td>
</tr>
<tr>
<td>-7234</td>
<td>User specific 10</td>
</tr>
<tr>
<td>-7233</td>
<td>User specific 11</td>
</tr>
<tr>
<td>-7232</td>
<td>Read constrain conflict</td>
</tr>
<tr>
<td>-7231</td>
<td>Write constrain conflict</td>
</tr>
<tr>
<td>-7230</td>
<td>Resource busy</td>
</tr>
<tr>
<td>-7229</td>
<td>Resource unavailable</td>
</tr>
<tr>
<td>-7228</td>
<td>Not specified</td>
</tr>
<tr>
<td>-7227</td>
<td>Not specified</td>
</tr>
<tr>
<td>-7226</td>
<td>Not specified</td>
</tr>
<tr>
<td>-7225</td>
<td>Not specified</td>
</tr>
<tr>
<td>-7224</td>
<td>User specific 12</td>
</tr>
</tbody>
</table>
### Table 13. Error Codes from PNIO Device (Continued)

<table>
<thead>
<tr>
<th>Error Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-7223</td>
<td>User specific 13</td>
</tr>
<tr>
<td>-7222</td>
<td>User specific 14</td>
</tr>
<tr>
<td>-7221</td>
<td>User specific 15</td>
</tr>
<tr>
<td>-7220</td>
<td>User specific 16</td>
</tr>
<tr>
<td>-7219</td>
<td>User specific 17</td>
</tr>
<tr>
<td>-7218</td>
<td>User specific 18</td>
</tr>
<tr>
<td>-7217</td>
<td>User specific 19</td>
</tr>
<tr>
<td>-7216-7169</td>
<td>User specific</td>
</tr>
</tbody>
</table>
Appendix B  CI871 TroubleShooting

Device Replacement

If a device is showing an error and must be exchanged, following information is taken into account:

1. Exchange the device with another one of same device type with same or compatible Software and Hardware revision. All this information can be accessed by I&M functionality.

2. If the device is used in an Ethernet network that does not have support for LLDP, then the newly inserted device must get the station name configured owned by the exchanged one. To configure the station name on the device, refer to Set Devicename on page 97. After having configured the station name the device get's automatically configured by CI871 and communication starts up.

3. If the device is used in an Ethernet network that has support for LLDP, then the newly inserted device gets automatically the station name configured by the adjacent switches. Subsequently the device is automatically configured by CI871 and communication starts up. Manual configuration is not required.

Network with LLDP support means that the switch(es) the device is connected to must have support for LLDP. Additionally the device itself must have support for LLDP as well.

Exchanging Device Positions

Exchanging the positions in the hardware tree of two PROFINET devices does not work. The communication for the device with the lower tree address does not start. A reset of CI871 must be done to overcome this situation.

Workaround: Exchange the positions in two steps.
Exchanging IP Addresses of Devices

Exchanging the IP addresses of PNIO devices causes a configuration conflict since the IP addresses are temporarily used twice in the system. The devices do not start cyclic communication.

**Workaround:** Exchange the IP addresses in two steps.

Set Devicename for an Active Device

**Set Devicename** is only done for devices not being in active cyclic communication. Otherwise it might occur that the device gets a short communication interruption with a succeeding Connection Down indication.
Revision History

This section provides information on the revision history of this User Manual.

The revision index of this User Manual is not related to the 800xA 6.0.2 System Revision.

The following table lists the revision history of this User Manual.

<table>
<thead>
<tr>
<th>Revision Index</th>
<th>Description</th>
<th>Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>-</td>
<td>Published for 800xA 6.0</td>
<td>August 2014</td>
</tr>
<tr>
<td>A</td>
<td>Published for 800xA 6.0.2</td>
<td>April 2016</td>
</tr>
</tbody>
</table>

Updates in Revision Index A

The following table shows the updates made in this User Manual for 800xA 6.0.2.

<table>
<thead>
<tr>
<th>Updated Section/Sub-section</th>
<th>Description of Update</th>
</tr>
</thead>
<tbody>
<tr>
<td>Appendix B CI871 TroubleShooting</td>
<td>Information for device replacement updated.</td>
</tr>
</tbody>
</table>
Updates in Revision Index A
Index

GSD File 21

H
Hardware Editor 64
Hardware Library 63
Hardware Status 36
HART Pass Through 54
Hot Swap 62

I
I/I Channel
   Creation 139
I/O Channel
   Deletion 138
   Renaming 135
I/O-Data 78
I&M Data 54, 99
IEC 61158 19
IEC 61784 19
Input Tab 124
Inserting
   CI871 63
   PNIO Device 69
IP 19
IP Address 65, 75
IRT 20
Isochronous Real-Time 20
ISP 62
ISP Control 83

J
Journal Buffer 101

L
Little endian 128
Logfile 82
Login 89

M
Manufacturer Specific Diagnosis 46
MLink 34
MNS iS 50, 79
modifying Default Parameters 143
Module 24
Multi Controller Access 34

N
Names 27

O
Online Data 83
Output Tab 142

P
Parameter Tab 143
Password 105
PN/PN Coupler 153
   Data Transfer 154
PNQ22 50 to 51
Process Alarms 46
PROFIBUS 20
PROFINET 19
PROFINET CBA 19
PROFINET IO 19

R
Read 53, 56 to 57
Real-Time 20
Real-Time Communication 27
Red. Ethernet Recovery Time 66
Redundancy 35
   Ethernet 36
Reimporting 150
Reset Default Password 106
RT 19, 29
Index

S
Security 85 to 88
Sequence of Events 47
Settings CI871 65
Settings Tab CI871 65
Shared Device 34
Startup Parameters 77
Station Name 65, 74 to 75
Status Handling 36
CI871 37
PROFINET IO Devices 39
Submodule 24
Subnet Mask 66
Subnet mask 75
System Event 45

T
TCP/IP 19, 29

U
UDP 29
UDP/IP 19
UMC100 50
Unit Status 36, 68
Update Time 30
Inputs 77
Outputs 77
Update Times 76
USI 46

W
Watchdog 31
Watchdog Factor 66
Web Interface 85
Write 53, 56
Contact us

www.abb.com/800xA
www.abb.com/controlsystems

Copyright © 2016 ABB.
All rights reserved.