Rexroth IndraWorks 12VRS
Field Buses

R911334394
Edition 02

Application Manual

Title
Rexroth IndraWorks 12VRS
Field Buses

Type of Documentation
Application Manual

Document Typecode
DOK-IWORKS-FB******V12-AP02-EN-P

Internal File Reference
RS-d570f432e4f9359f0a6846a0000c9d99-2-en-US-7

Purpose of Documentation
This documentation describes potential field buses and IndraLogic 2G libraries which are used with the IndraLogic XLC, IndraMotion MLC and IndraMotion MTX systems.

This manual is the basis for the online help.

Record of Revision

Edition	Release Date	Notes
Edition 01	06.2011	First edition
Edition 02	02.2012	Additional function in 12V04

Liability
The specified data is intended for product description purposes only and shall not be deemed to be a guaranteed characteristic unless expressly stipulated in the contract. All rights are reserved with respect to the content of this documentation and the availability of the product.

Published by
Bosch Rexroth AG
Bgm.-Dr.-Nebel-Str. 2 • 97816 Lohr a. Main, Germany
Phone +49 (0)93 52 / 18-0 • Fax+49 (0) 9352 18 49 41
http://www.boschrexroth.com/
System IndraLogic & Basis Motion Logic BAn / HBu (vha/PiGe)

Note
This document has been printed on chlorine-free bleached paper.

Table of Contents

Page

Table of Contents

Page

Table of Contents

Page

Table of Contents

Page

Table of Contents

Page

Table of Contents

Page

1 Field Buses and Field Bus Communication

1.1 About this Documentation

1.1.1 Validity of the Documentation

Target group

This documentation is intended for users wanting to inform themselves about the available field buses of IndraLogic XLC, IndraMotion MLC or IndraMotion MTX.

The following is described

- Field buses and field bus communication
  - Terms and abbreviations
  - Field bus features
  - Transmission types
- Field bus support in the PLC user program
  - PROFIBUS DP, master und slave
  - PROFINET I/O, controller und device
  - EtherNet/IP, adapter and adapter via the Engineering interface
  - sercos III I/O, master
  - Inline I/Os
  - Onboard I/Os
- Device database, installation of additional devices and uninstallation of devices that are not required
- Mapping the Onboard, Inline and field bus inputs and outputs
- Field bus libraries
  - RIL_ProfibusDP_02.library / RIL_ProfibusDPSlave.library
  - RIL_ProfinetIO.library / RIL_ProfinetIODevice.library
  - RIL_EtherNetIPAdapter.library
  - RIL_MappingList.library
  - RIL_SERCOSIII.library
  - RIL_Inline.library

This documentation supports the user during the phases:

- Configuration of field bus components in IndraWorks
- Field bus application in the PLC user program

1.1.2 Required and Supplementing Documentations

The documents listed below contain additional information regarding this subject.

Rexroth IndraWorks 12VRS IndraLogic 2G Programming Instruction DOK-IWORKS-IL2GPRO*V12-APxx-EN-P, R911334390 This documentation describes the PLC programming tool IndraLogic 2G and its usage. It includes the basic usage, first steps, visualization, menu items and editors.
Rexroth IndraWorks 12VRS, Basic Libraries, IndraLogic 2G DOK-IL*2G*-BASLIB**V12-LIxx-EN-P, R911333835 This documentation describes the system-comprehensive PLC libraries.
Rexroth IndraMotion MLC 12VRS Technology Libraries DOK-MLC***-TF*LIB**V12-LIxx-EN-P, R911333868 This documentation describes the function blocks, functions and data types of the libraries "ML_TechInterface.library", "ML_TechMotion.library", "RMB_TechCam.library" and "ML_TechBase.library". It also includes libraries for the winder functionality, register controller functionality and CrossCutter functionality.

1.1.3 Information Representation

Safety Instructions

If there are safety instructions, they contain certain signal words (Danger, Warning, Caution, Notice) and if applicable, signal alert symbols (acc. to ANSI Z535.6-2006).

The signal word draws the attention to the safety instruction and indicates the risk potential.

The signal alert symbol (warning triangle with exclamation mark) positioned in front of the signal words Danger, Warning and Caution indicates hazards for individuals.

⚠ DANGER

In case of non-compliance with this safety instruction, death or serious injury **will** occur.

⚠ WARNING

In case of non-compliance with this safety instruction, death or serious injury **can** occur.

⚠ CAUTION

In case of non-compliance with this safety instruction, minor or moderate injury can occur.

NOTICE

In case of non-compliance with this safety instruction, material or property damage can occur.

Symbols Used

Note Notes are represented as follows:

☞ This is a note for the user.

Tip

Tips are represented as follows:

This is a tip for the user.

Names and Abbreviations

Term	Explanation
ENIP	EtherNet/IP
PNIO	PROFINET I/O

Fig. 1-1: Terms and abbreviations used

1.2 Field Buses and Field Bus Communication, Overview

This is a brief introduction to the functionalities of field bus communication and other varieties of communication (e.g. EtherNet communication). The respective functionality consists of the following properties:

- Cyclic data traffic
- Acyclic data traffic
- Representation in the engineering software (configuration, diagnostics, ...)
- Features in the PLC (function blocks and functions for diagnostics or acyclic communication)

The properties are described for each field bus and/or communication.

Overview

The following topics will be treated:

- PROFIBUS DP Master (see page 19)

  A PROFIBUS DP master can be activated on the onboard interface or on a connected function module (CFL01.1-TP).

- PROFIBUS DP Slave (see page 20)

  A PROFIBUS DP slave can be activated on the onboard interface or on a connected function module (CFL01.1-TP).

- PROFINET I/O Controller (see page 53)

  A PROFINET I/O controller can be activated on the onboard interface or on a connected function module (CFL01.1-TP).

- PROFINET I/O Device (see page 54)

  A PROFINET I/O device can be activated on the onboard interface or on a connected function module (CFL01.1-TP).

- EtherNet/IP Adapter (see page 91)

  or

  EtherNet/IP adapter (Engineering) (see page 101)

  An EtherNet/IP adapter can be activated on the onboard interface, on a connected function module (CFL01.1-TP) or on the Engineering interface.

- sercos III I/O (see page 105)

  A sercos III master interface for I/Os or field devices is only available via the onboard sercos III interface

- Inline I/Os (see page 121)

Inline I/O modules can be connected to the local inline bus of the control and using inline field bus couplers. Furthermore, inline block modules are available from Rexroth in the degree of protection IP20 and inline modules in protection class IP67.

- Onboard I/O (see page 131)

  The onboard I/O modules are a direct component of the control.

1.3 Terms and Abbreviations

The following table shows the field bus dependent designations for the master or slave interfaces of the field buses:

Terms for the field bus interfaces

Field bus	Master interface	Slave interface
PROFIBUS DP	Master	Slave
PROFINET I/O	Controller	Device
EtherNet/IP	Scanner	Adapter
sercos III	Master	Slave

Terms for data transfer mechanisms

Field bus	Cyclic transfer	Acyclic transfer
PROFIBUS DP	DP/V0	DP/V1
PROFINET I/O	RTC (real-time cyclic)	RTA (real-time acyclic)
EtherNet/IP	Implicit messaging (polled I/O)	Explicit messaging
sercos III	Cyclic data	Service channel + IP channel

The local inline bus at the control is a special feature. It is comparable with a field bus master with exactly one field bus slave to which I/O modules can be connected.

GSD file This is the device description with which the PROFIBUS DP slaves are made known to IndraWorks or another Engineering system.

GSDML file This is the device description with which the PROFINET I/O device is made known to IndraWorks or another Engineering system.

EDS file This is the device description with which the EtherNet/IP adapter is made known to IndraWorks or another Engineering system (EDS: Electronic Data Sheet).

SDDML file This is the device description with which the sercos III slaves are made known to IndraWorks or another engineering system according to an I/O profile (SDDML: sercos Device Description Markup Language).

1.4 Field Bus Features

1.4.1 Field Bus Master Features

Features of the field bus master interfaces

	PROFIBUS DP master	PROFINET I/O controller	sercos III master	Inline
Max. number of participants	125	25 / 128 [1]	64 / 32 [2]	64 modules
Max. number of cyclic input data for the master	3584 bytes	3072 bytes	--	128 bytes [3]
Max. number of cyclic output data for the master	3584 bytes	3072 bytes	--	128 bytes [3]
Max. number of cyclic input data per slave	244 bytes	1024 bytes	--	--
Max. number of cyclic output data per slave	244 bytes	1024 bytes	--	--
Max. telegram data per acyclic telegram	240 bytes	1092 bytes	1480 bytes	--

[1] 25 at a cycle time < 4 ms, 128 at a cycle time >= 4 ms

[2] Up to 16 (L25), 32 (L45), 64 (L65) drives and 32 I/O participants

[3] Inputs and outputs and PCP channel calculated together

1.4.2 Field Bus Slave Features

Features of the field bus slave interfaces

	PROFIBUS DP slave	PROFINET I/O device	EtherNet/IP adapter	EtherNet/IP adapter (Engineering)
Max. number of cyclic input data for the slave	244 bytes	1024 bytes	504 bytes	128 bytes
Max. number of cyclic output data for the slave	244 bytes	1024 bytes	504 bytes	128 bytes + 4 bytes header
Consistency of the cyclic data	128 bytes	128 bytes	128 bytes	128 bytes
Max. telegram data per acyclic telegram	240 bytes	1092 bytes	450 bytes	

1.5 Transfer Types

The transfer types can be divided into the following categories:

- Cyclic transfer channel
- Acyclic transfer channel

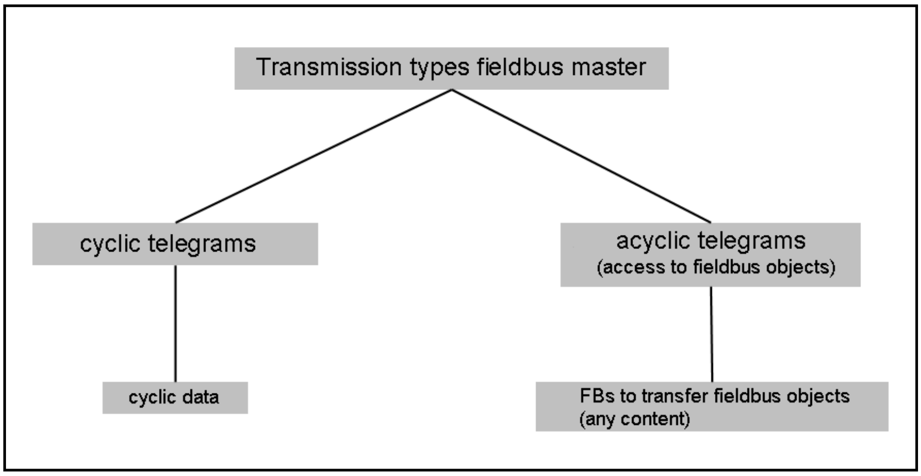

Transmission types fieldbus master
cyclic telegrams
acyclic telegrams
(access to fieldbus objects)
cyclic data
FBs to transfer fieldbus objects
(any content)

Fig.1-2: Overview of the transfer types for the field bus master

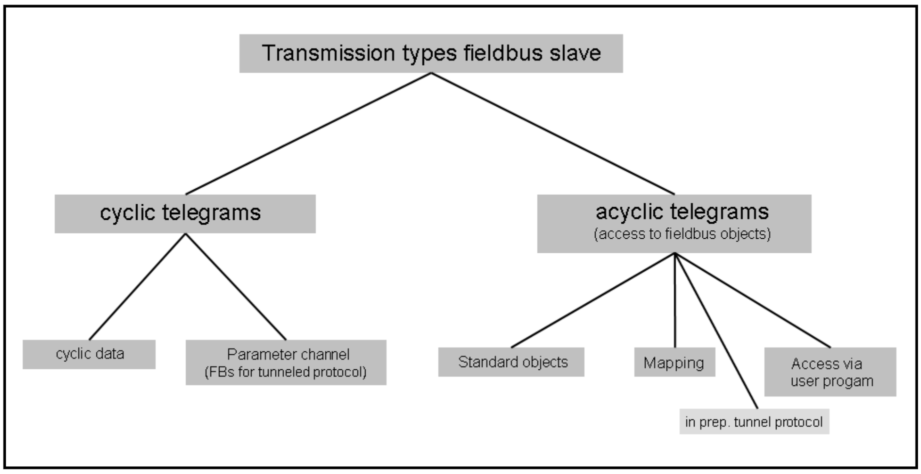


Fig.1-3: Overview of the transfer types for the field bus slaves

The field bus objects are described using the following elements:

- PROFIBUS: Slot (8 bits), Index (8 bits)
- PROFINET I/O: Slot (16 bits), Subslot (16 bits), Index (16 bits)
- EtherNet/IP: Class (8 bits), Instance (8/16/32 bits), Attribute (16 bits)
- sercos III: EIDN (32 bits) with parameter number (16 bits), Structure element SE (8 bits), Structure instance SI (8 bits)

2 Field Bus Support in the PLC User Program

2.1 Field Bus Support in the PLC User Program, Overview

The following support with regard to field buses is contained within the PLC user program:

1. Function Blocks for Field Bus Diagnostics, page 15
2. Function Blocks for Acyclic Data Transfer (on the Master Side), page 16
3. Function Blocks for the Mapping Concept (on the Slave Side), page 17
4. Function Blocks for Access from the User Program (on the Slave Side, only PROFIBUS DP Slave), page 17
5. Function Blocks for the Parameter Channel (on the Slave Side), page 17
6. Function Blocks for Access to Data Objects via the PCB Channel, page 17

2.2 Function Blocks for Field Bus Diagnostics

PROFIBUS DP master

The RIL_ProfibusDP_02 library contains the following function blocks for diagnostic purposes

- IL_DPMasterState, page 166, determines the state of a local bus master as well as of the configured slaves,
- IL_DPDeviceListGet, page 175, establishes a list of the currently available PROFIBUS DP masters,
- IL_DPDevInfoGet, page 177, retrieves information about the operating state of the affected PROFIBUS DP device,
- IL_DPPrjSlaveListGet, page 178, returns a list of configured slaves. The list contains all of the slaves contained in the master configuration file,
- IL_DPSlaveDiagListGet, page 179, establishes a lit of the slaves, reporting a diagnostics,
- IL_DPReadDiag, page 172, reads the diagnostic data of a slave from the PROFIBUS DP master (DPM1).

PROFIBUS DP slave

in preparation

PROFINET I/O controller

The RIL_ProfinetIO library contains the following function blocks for diagnostic purposes:

- IL_PNIOControllerState, page 198, for determining the basis state
- IL_PNIOControllerStateDetails, page 199 for determining the state of the PROFINET stacks

The library contains the following function blocks for remote diagnostic purposes:

- IL_PNIORemoteDeviceState, page 201, for determining the basic state of any device
- IL_PNIORemoteDeviceStateDetails, page 202, for determining the state of any device

PROFINET I/O device The RIL_ProfinetIODevice library contains the following diagnostic function blocks:

- IL_PNIODeviceState, page 221, for determining the state of a PROFINET device
- IL_PNIODeviceStateDetails, page 222, for determining the state of the PROFINET device stack
- IL_PNIODeviceStateDetailsXMAC, page 224, state of the PROFINET 2-port switch

sercos III (master) ### in preparation ###

EtherNet/IP adapter The RIL_EtherNetIPAdapter library includes the following function blocks for the EtherNet/IP adapter via the onboard or function module interface:

- IL_ENIPAdapterState, page 231, diagnostics: basic adapter state.
- IL_ENIPAdapterStateDetails, page 233, diagnostics: detailed status of the Ethernet/IP stack of the adapter.

The RIL_EtherNetIPAdapter library includes the following function block for the EtherNet/IP adapter via the Engineering interface:

- IL_Status, page 234, cyclic communication diagnostics

Inline The RIL_Inline.library library contains the following essential function blocks for Inline bus diagnostics:

- IL_INLNState, page 261 outputs the state of the local inline I/O
- IL_INLNStateDetails, page 262, outputs the detailed inline I/O diagnostics to allow finding the location and cause of the error
- IL_INLNModuleConfigList, page 265, determines the configured module equipment and the existing one
- IL_INLNReadCounter, page 269, determines the inline cycle counters
- IL_INLNClearCounter, page 270, resets the nline cycle counters

2.3 Function Blocks for Acyclic Data Transfer (on the Master Side)

PROFIBUS DP master The RIL_ProfibusDP_02 library contains the following function blocks for acyclic data transmission:

- IL_DPV1Read, page 169, reading V1 service
- IL_DPV1Write, page 170, writing V1 service

PROFINET I/O controller The RIL_ProfinetIO library contains the following function blocks for acyclic data transmission:

- IL_PNIOWriteRecord, page 211, acyclic writing, and
- IL_PNIOReadRecord, page 209, acyclic reading

sercos III (master) The RIL_SERCOSIII library contains the following function blocks for implementing the acyclic services (AcyclicCommunication):

- FB IL_SIIISvcWrite, page 256, writing a parameter via the sercos III service channel
- FB IL_SIIISvcRead, page 251, reading a parameter via the sercos III service channel.

2.4 Function Blocks for the Mapping Concept (on the Slave Side)

To allow acyclic access to field bus slaves (PROFIBUS DP slave, PROFINET I/O device and EtherNet/IP adapter), a mapping table is integrated in the field bus slaves.

A mapping to data objects of the control is stored in this mapping table to allow bus-specific acyclic field bus access.

This mapping (= addressing) rule is executed when an acyclic field bus access is made.

The library RIL_MappingList contains the following function blocks:

- IL_SlaveMapListInit, page 244, initializes and deletes the mapping table
- IL_SlaveMapListAddEntry, page 245, adds an entry to the mapping table

2.5 Function Blocks for Access from the User Program (on the Slave Side, only PROFIBUS DP Slave)

The functionality implements the acyclic DPV1 services READ and WRITE for the PROFIBUS DP slave. This allows access to data objects on slave application level.

The RIL_ProfibusDPSlave contains the following function blocks:

- FB IL_PBDPSlaveDPV1Polling, page 187, the polling FB is used to query the activity of a DPV1 service request. The call is made cyclically.
- FB IL_PBDPSlaveDPV1Response, page 189, the response FB is used to respond to an active DPV1 service request. The call is made in relation to the result of the polling FB.
- FB IL_PBDPSlaveDPV1GetWriteData, page 192, GetWriteData is used to copy the data of a DPV1 WRITE service request to the application object.

2.6 Function Blocks for the Parameter Channel (on the Slave Side)

Using its function block IL_ParameterChannel, the RIL_ParameterChannel library allows acyclic communication via the cyclic channel of a field bus connection.

2.7 Function Blocks for Access to Data Objects via the PCB Channel

The RIL_Inline implements the access to data objects via the PCB channel for Inline modules.

- FB IL_INLNPCPRead, page 272, reading PCB service,
- FB IL_INLNPCPWrite, page 273, writing PC service.

3 PROFIBUS DP

3.1 PROFIBUS DP Master

3.1.1 Terms and Abbreviations

Master The PROFIBUS field bus master is called **master**.

Slave The PROFIBUS field bus slaves are called **slaves**.

Device address The criterion for addressing a slave is the device address (or PROFIBUS address).

It can range from 1 to 125.

GSD file This is the device description with which the PROFIBUS slaves are made known to IndraWorks.

3.1.2 PROFIBUS DP Master Features

The current implementation of the PROFIBUS DP master includes the following functionalities:

- Bus master functionality in accordance with DIN EN 50170, Part 2
- Cyclic data traffic (DP/V0)
- Acyclic data traffic (DP/V1 class 1) via FBs
- Acyclic data traffic (DP/V1 class 2) for FDT/DTM communication
- Device description file import into the device database
- PROFIBUS slave-related connection status in the engineering software
- PROFIBUS master diagnostics via FB
- PROFIBUS slave diagnostics via FB
- Sync/Freeze
- Disabling slaves (at bus startup)

Future extensions are planned for the following functionalities:

- Alarms
- Disabling slaves in the PLC program

Function/Characteristic	Value
Max. number of slaves	125
Baud rate	9.6 kBit/s to 12 MBit/s
Max. amount of cyclic input data	3584 bytes
Max. amount of cyclic output data	3584 bytes
Max. amount of cyclic input data per slave	244 bytes
Max. amount of cyclic output data per slave	244 bytes
Max. length of consistent data blocks	244 bytes
Max. acyclic telegram data per slave / telegram	240 bytes
Number of simultaneous DP/V1 class 1 services per slave	1
Number of simultaneous DP/V1 class 2 connections	1

Fig.3-1: Technical data

3.2 PROFIBUS DP Slave

3.2.1 Terms and Abbreviations

Master The PROFIBUS DP field bus master is called **master**.

Slave The PROFIBUS DP field bus slaves are called **slaves**.

Device address The criterion for addressing a slave is the device address (or PROFIBUS address).

It can range from 1 to 125.

GSD file This is the device description with which the PROFIBUS slaves are made known to IndraWorks.

3.2.2 PROFIBUS DP Slave Features

The current implementation of the PROFIBUS DP slave includes the following functionalities:

- Slave functionality in accordance with DIN EN 50170, Part 2
- Cyclic data traffic (DP/V0)
- Acyclic data traffic (DP/V1 class 1) via FB
- Disabling slaves when starting up the PLC program

Future extensions are planned for the following functionalities:

- PROFIBUS DP slave diagnostics via FB
- Alarms
- Sync/Freeze

Function/Characteristic	Value
Max. number of I/O modules	16 (8 in each direction)
Baud rate	9.6 kBit/s to 12 MBit/s
Max. amount of cyclic input data	244 bytes
Max. amount of cyclic output data	244 bytes
Max. length of consistent data blocks	128 bytes

Fig.3-2: Technical data

3.3 Configuring PROFIBUS DP

3.3.1 Configuring PROFIBUS DP, Overview

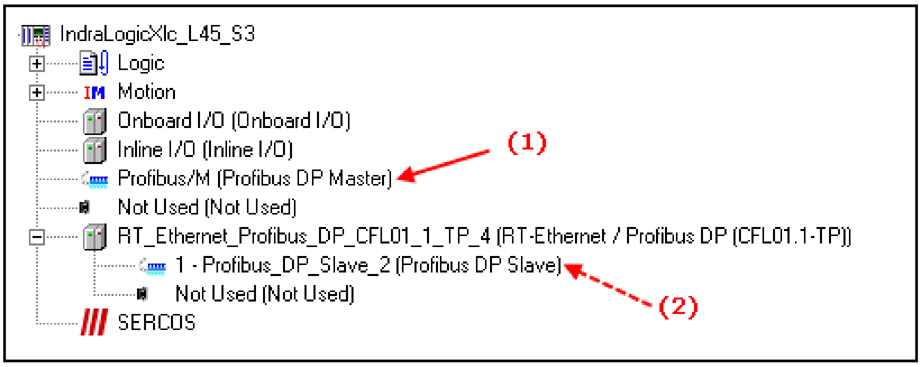
The following controls allow implementing PROFIBUS DP:

- IndraLogic XLC L25
  - With sercos III:
    PROFIBUS DP master / PROFIBUS DP slave, function modules
  - Without sercos III:
    PROFIBUS DP master / PROFIBUS DP slave, onboard
    PROFIBUS DP master / PROFIBUS DP slave, function modules
- IndraMotion MLC L25: PROFIBUS DP master / PROFIBUS DP slave, function modules
- IndraLogic XLC L45/L65 and IndraMotion MLC L45/L65
  PROFIBUS DP master / PROFIBUS DP slave, onboard

PROFIBUS DP master / PROFIBUS DP slave, function modules

The interface respectively available at the control is represented as a PROFIBUS DP object in the Project Explorer.

The PROFIBUS DP object can either be configured when creating the control or via the context menu item **Set device** as PROFIBUS DP master or PROFIBUS DP slave.


IndraLogicXlc_L45_S3
Logic
Motion
Onboard I/O (Onboard I/O)
Inline I/O (Inline I/O)
Profibus/M (Profibus DP Master)
Not Used (Not Used)
RT_Ethernet_Profibus_DP_CFL01_1_TP_4 (RT-Ethernet / Profibus DP (CFL01.1-TP))
1 - Profibus_DP_Slave_2 (Profibus DP Slave)
Not Used (Not Used)
SERCOS

(1) PROFIBUS DP, onboard, configured as master
(2) PROFIBUS DP, function module RT_EtherNet / Profibus_DP (CFL01.1-TP), configured as slave

Fig.3-3: PROFIBUS DP objects, still unassigned

PROFIBUS DP master

- *PROFIBUS DP master object*
  - DP parameters, page 23,
  - Field bus diagnostics, page 25, is only visible if the option "Show generic configuration editors" is enabled under **Tools ▸ Options ▸ IndraLogic 2G ▸ Device editor**.
  - PROFIBUS DP master configuration, page 27, this window is used for service purposes and is only visible if the option "Show generic configuration editors" is enabled under **Tools ▸ Options ▸ IndraLogic 2G ▸ Device editor**.
  - PROFIBUS DP Master I/O mapping, page 28,
  - Status, page 29,
  - Information, page 29.
- *PROFIBUS DP master object, adding PROFIBUS DP slaves*
  - Enabling/Disabling PROFIBUS DP Slaves, page 33,
  - DP parameters, page 34,
  - PROFIBUS DP configuration, page 36, this window is used for service purposes and is only visible if the option "Show generic configuration editors" is enabled under **Tools ▸ Options ▸ IndraLogic 2G ▸ Device editor**.
  - Status, page 36,
  - Information, page 37.
- *PROFIBUS DP master object, adding modules*
  - DP parameters, page 38,
  - DP module configuration, page 39, this window is used for service purposes and is only visible if the option "Show generic configuration editors" is enabled under **Tools ▸ Options ▸ IndraLogic 2G ▸ Device editor**.
  - DP_Module I/O mapping, page 39,

- Status, page 40,
- Information, page 40.

PROFIBUS DP slave

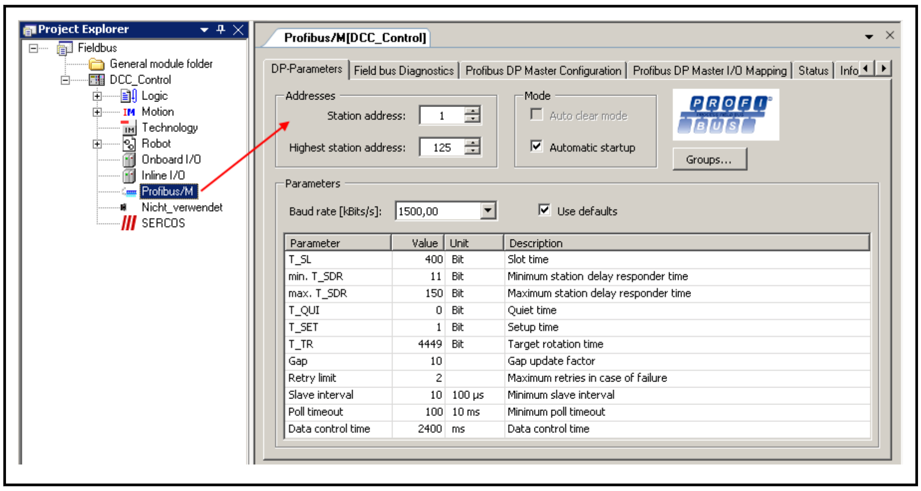
- *PROFIBUS DP slave object*
  - DP parameters, page 41,
  - PROFIBUS DP configuration, page 45, this window is used for service purposes and is only visible if the option "Show generic configuration editors" is enabled under **Tools ▸ Options ▸ IndraLogic 2G ▸ Device editor**.
  - PROFIBUS DP I/O mapping, page 42,
  - Field bus mapping, page 43,
  - Status, page 46,
  - Information, page 46.
- *PROFIBUS DP slave object, adding modules*
  - DP parameters, page 47,
  - DP module configuration, page 48, this window is used for service purposes and is only visible if the option "Show generic configuration editors" is enabled under **Tools ▸ Options ▸ IndraLogic 2G ▸ Device editor**.
  - DP modules I/O mapping, page 48,
  - Status, page 49,
  - Information, page 49.

3.3.2 PROFIBUS DP Master

Configuring the PROFIBUS DP Master, General Information

To open the editing window, double click on the PROFIBUS DP master object in the Project Explorer.

The dialogs will inform you about the configuration of the entire PROFIBUS and you can modify it if necessary.



Addresses

Station address: 1

Highest station address: 125

Mode

☐ Auto clear mode

☑ Automatic startup

Parameters

Baud rate [kBits/s]: 1500,00

☑ Use defaults

Parameter	Value	Unit	Description
T_SL	400	Bit	Slot time
min. T_SDR	11	Bit	Minimum station delay responder time
max. T_SDR	150	Bit	Maximum station delay responder time
T_QUI	0	Bit	Quiet time
T_SET	1	Bit	Setup time
T_TR	4449	Bit	Target rotation time
Gap	10		Gap update factor
Retry limit	2		Maximum retries in case of failure
Slave interval	10	100 µs	Minimum slave interval
Poll timeout	100	10 ms	Minimum poll timeout
Data control time	2400	ms	Data control time

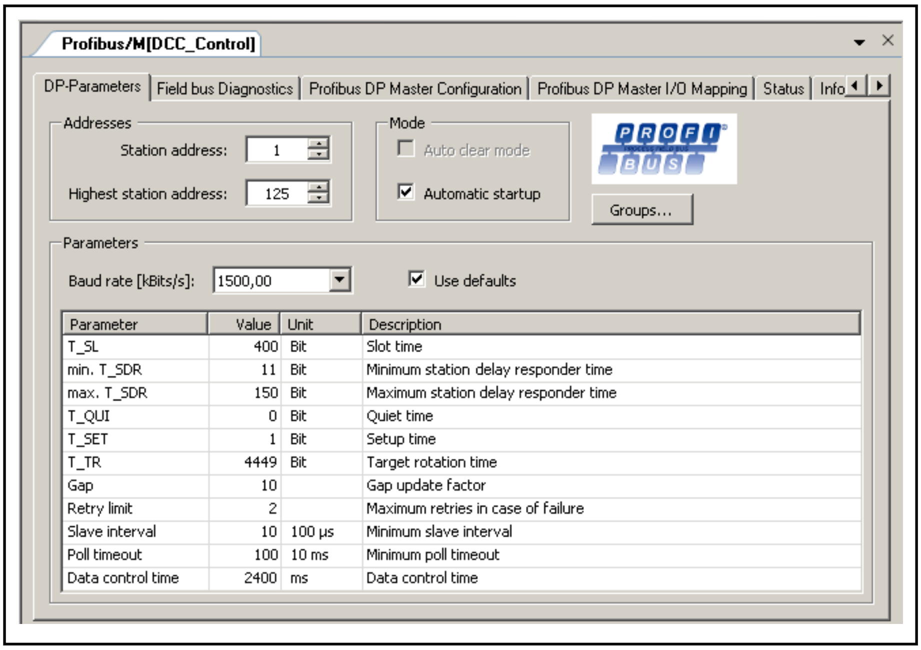
Fig.3-4: PROFIBUS DP master object

Register

- DP parameters, page 23,

- Field bus diagnostics, page 25, is only visible if the option "Show generic configuration editors" is enabled under **Tools ▸ Options ▸ IndraLogic 2G ▸ Device editor**.
- PROFIBUS DP master configuration, page 27, this window is used for service purposes and is only visible if the option "Show generic configuration editors" is enabled under **Tools ▸ Options ▸ IndraLogic 2G ▸ Device editor**.
- PROFIBUS DP Master I/O mapping, page 28,
- Status, page 29,
- Information, page 29.

Tab "DP parameters"



Addresses

Station address: 1

Highest station address: 125

Mode

☐ Auto clear mode

☑ Automatic startup

Parameters

Baud rate [kBits/s]: 1500,00

☑ Use defaults

Parameter	Value	Unit	Description
T_SL	400	Bit	Slot time
min. T_SDR	11	Bit	Minimum station delay responder time
max. T_SDR	150	Bit	Maximum station delay responder time
T_QUI	0	Bit	Quiet time
T_SET	1	Bit	Setup time
T_TR	4449	Bit	Target rotation time
Gap	10		Gap update factor
Retry limit	2		Maximum retries in case of failure
Slave interval	10	100 µs	Minimum slave interval
Poll timeout	100	10 ms	Minimum poll timeout
Data control time	2400	ms	Data control time

Fig.3-5: PROFIBUS DP master object: DP parameters

Parameters which have to be configured for each PROFIBUS DP master. In general, they are defined via the device description file but can be modified in the DP Parameters dialog.

Addresses:

- **Station address**: The valid range is 0 to 125.

  Each new devices that is added to the bus automatically has the next higher address assigned to it.

  Manual input is allowed; addresses are checked for duplicates.
- **Highest station address**: The highest station address on the bus is displayed. To keep the address range for searching for new active devices small, a low address value can be entered here.

Use the lowest addresses possible. High address values decrease bus performance!

Mode:

- **Auto clear mode:** is currently not supported.

- **Automatic startup:** When this option is activated, the master starts automatically; otherwise it has to be be started manually.

☞ Whether or not the setting for the "Automatic startup" option is evaluated depends on the driver.

Group properties:

- The "Groups..." button opens the 'Group Properties' dialog. The group properties refer to the slaves assigned to the master.

  Up to eight groups can be set up. For each group, you can set whether it is to operate in freeze mode and/or sync mode.

  The assignment of the slaves (see "Properties of DP Slaves", "Group assignment") to different groups can synchronize the data exchange from the master using a global control command.

  With a **freeze command** a master causes a slave or a group to 'freeze' the inputs in the current status and to transfer this data in the following data exchange.

  A **sync command** prompts the slaves to cycle through the data received by the master in the following data exchange synchronously with the next sync command at the outputs.

  To switch the freeze and sync options for a group on and off, left-click on the corresponding position in the table to place/remove a check next to the desired option. In addition, you can edit the group names here.

  Group names can be changed. To do this, select the name from the Group column and press the <space bar> to allow editing of the character string. To enable the Freeze and Sync option for a group, click on the checkbox in the corresponding column. If the checkbox is ticked, i.e., contains an "x", the property is enabled.

Parameters: These parameters describe the time behavior of the communication on the PROFIBUS.

- **Baud rate [kBits/s]**:

  The selection defined in the device description file is available here.

☞ In the baud rate selection field, only one baud rate [KB/sec.] can be set which all slaves support.

- **Use defaults**: When this option is enabled, the values in the parameter table are set to the default values based on the currently set baud rate and can **not** be modified.

For each parameter, the parameter table displays the unit and a short description.

The parameter values can be edited by double clicking on the respective value field (the option Preset: disabled).

Parameter [<Unit>]	Description
T_SL [bits[1]]	Slot time: Maximum amount of time that the master waits for the slave to respond after sending a request.
min. T_SDR [bits]	Minimum station delay responder time: Minimum response time after which a participant on the bus may respond.
max. T_SDR [bits]	Maximum station delay responder time: Maximum time period within which a slave must respond.
T_QUI [bits]	Quiet time: Quiet time to be taken into account when converting NRZ signals (Non Return to Zero) to other encoding (changeover time for repeater).
T_SET [bits]	Setup time: Time for setup
T_TR [bits]	Target rotation time: Token command changeover time, projected time frame in which a master is to receive the token. The result taken from the sum of the token holding times for all of the masters on the bus.
Gap	Gap update factor: Number of bus cycles after which another newly added, active station is searched for in the master's gap (address range from its own bus address to the address of the next active participant).
Retry limit	Maximum number of times the master tries to call again if it does not receive a valid response from the slave.
Slave interval [µs]	Time between two bus cycles in which a slave can process the master's request (time basis 100 µs). The value entered here must match the respective specifications in the slave's device description file.
Poll timeout [ms]	Maximum time after which the response of a master in a master-master communication must be obtained from the requester (DP_Master class 2, time basis 1 ms).
Data control time [ms]	Time in which the master informs the assigned slaves regarding its operating status. The master simultaneously monitors whether at least one user data exchange has occurred with the slaves within this period and updates the Data_Transfer_List.

Fig.3-6: Parameter table

Tab "Field Bus Diagnostics"

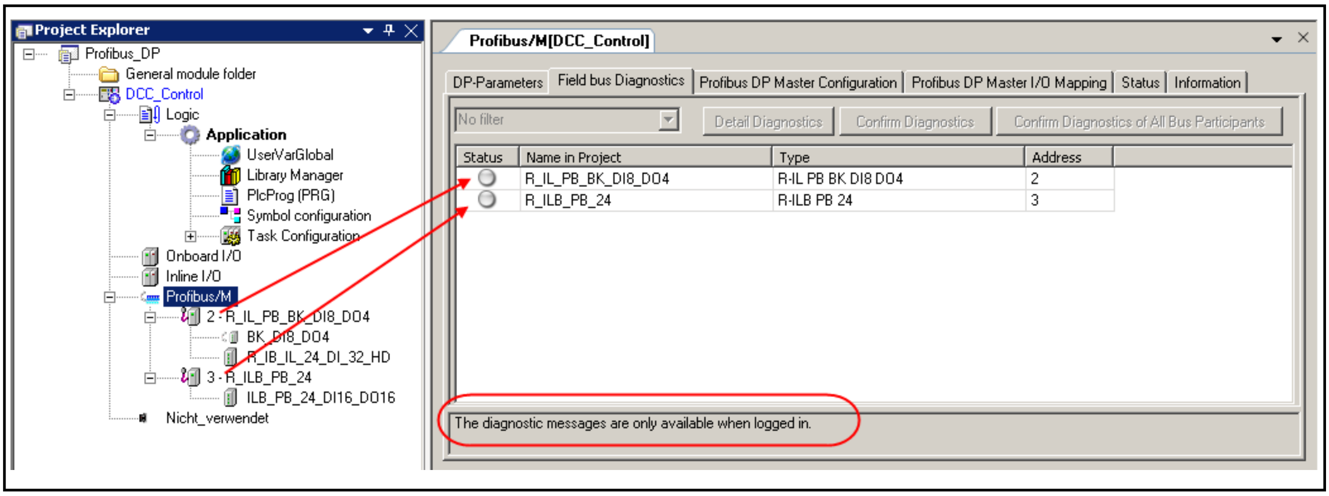
Based on the object node of the master, here: PROFIBUS DP master, a uniform diagnostics concept has been developed for the field buses of the IndraLogic XLC and IndraMotion MLC / MTX systems.

The "Field bus diagnostics" tab is only visible if the option "Show generic configuration editors" is enabled under **Tools ▸ Options ▸ IndraLogic 2G ▸ Device editor.**

The "Field bus Diagnostics" tab shows the modules applied to the field bus node in online mode.

[1] *Bit: Time unit for transferring a bit to the PROFIBUS*

PROFIBUS DP



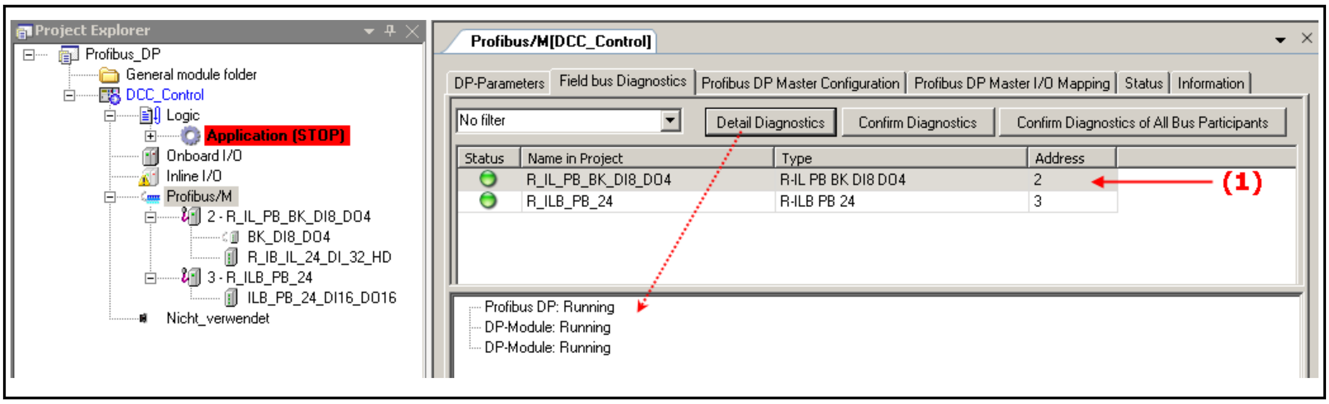
Status	Name in Project	Type	Address
○	R_IL_PB_BK_DI8_DO4	R-IL PB BK DI8 DO4	2
○	R_ILB_PB_24	R-ILB PB 24	3

The diagnostic messages are only available when logged in.

Fig.3-7: Field bus diagnostics, online control, not logged in

After login, the bus available at the node runs through a diagnostics cycle. The status LEDs change their color in the result of this cycle (here: green), i.e., both modules signal an error-free run.

After having been selected, a module **(1)** is available for detailed diagnostics.



Status	Name in Project	Type	Address
●	R_IL_PB_BK_DI8_DO4	R-IL PB BK DI8 DO4	2
●	R_ILB_PB_24	R-ILB PB 24	3

Profibus DP: Running
DP-Module: Running
DP-Module: Running

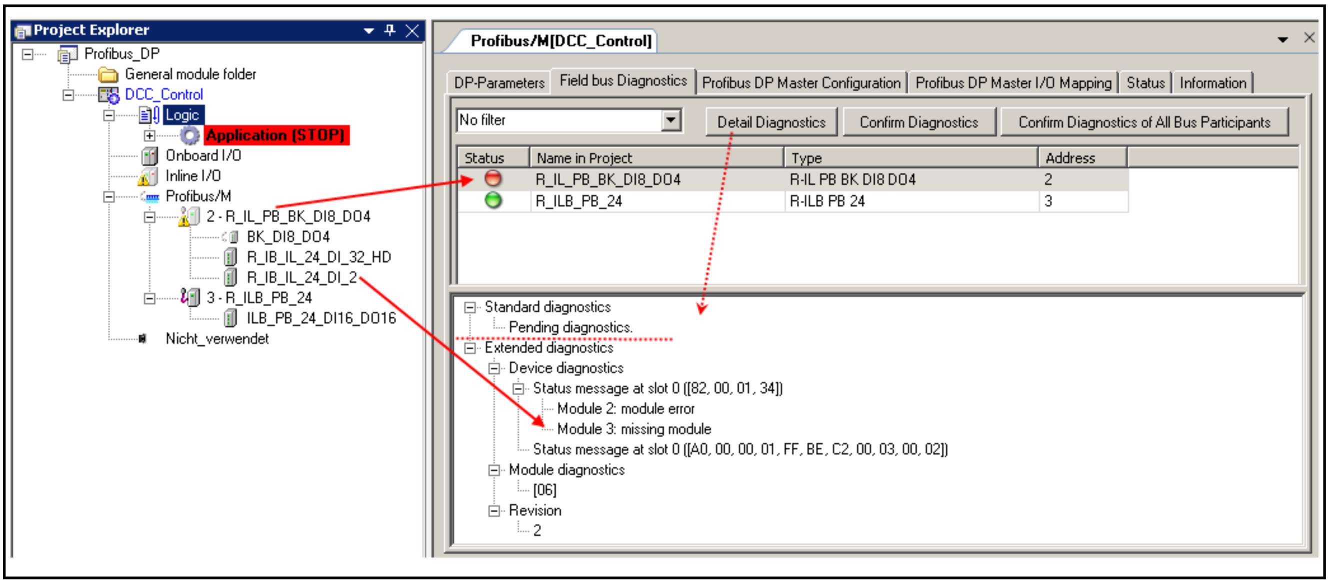
Fig.3-8: Field bus diagnostics, online control, logged in, error-free

Diagnostics in case of an error

Login is repeated with an additional terminal not available at the real control.

A yellow warning triangle appears at the module in question, indicating that a diagnostic message is present. The status LED of the module in question turns red. Clicking on "Detail Diagnostics" provides "Standard diagnostics" with brief information and "Extended diagnostics" with detailed specification of the error cause.

- Project Explorer, yellow warning triangle at the module: a diagnostic message is present.
- Detail Diagnostics: module 2, module error, shows the following error.
- Detail Diagnostics: module 3, module missing, indicates the module missing in reality.



```
Profibus_DP
    General module folder
    DCC_Control
        Logic
            Application [STOP]
        Onboard I/O
        Inline I/O
        Profibus/M
            2 - R_IL_PB_BK_DI8_DO4
                BK_DI8_DO4
                R_IB_IL_24_DI_32_HD
                R_IB_IL_24_DI_2
            3 - R_ILB_PB_24
                ILB_PB_24_DI16_DO16
        Nicht_verwendet
```

Status	Name in Project	Type	Address
🔴	R_IL_PB_BK_DI8_DO4	R-IL PB BK DI8 DO4	2
🟢	R_ILB_PB_24	R-ILB PB 24	3

```
Standard diagnostics
    Pending diagnostics.
Extended diagnostics
    Device diagnostics
        Status message at slot 0 ([82, 00, 01, 34])
            Module 2: module error
            Module 3: missing module
        Status message at slot 0 ([A0, 00, 00, 01, FF, BE, C2, 00, 03, 00, 02])
    Module diagnostics
        [06]
    Revision
        2
```

Fig.3-9: Field bus diagnostics, online control, logged in, with error

See also Tab "Status", page 29.

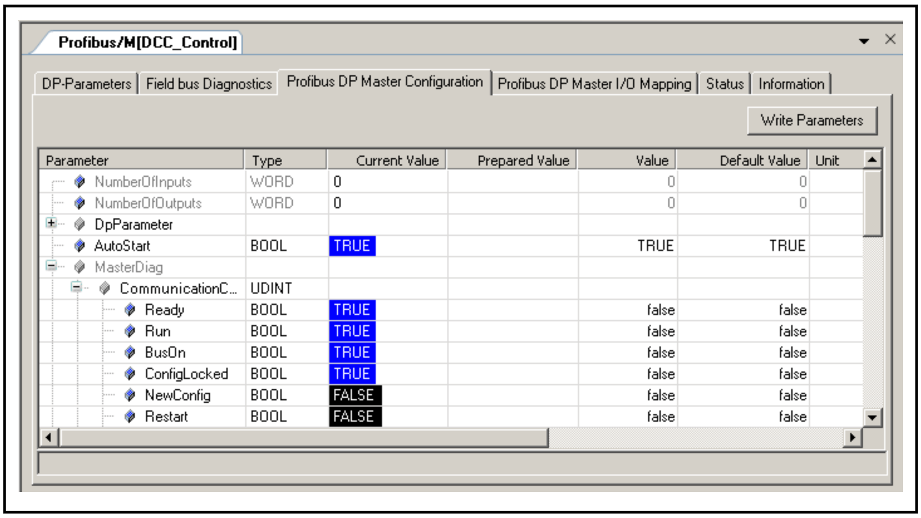
Confirm Diagnostics: Although there is an error, both the warning triangle and the red status LED are turned off for the selected module. The "Extended diagnostics" of the "Detail Diagnostics" remains in the text message. (The next diagnostic message can be processed...)

Confirm Diagnostics of All Bus Participants: Although there are errors, both the warning triangle and the red status LED are turned off for all modules.

Tab "PROFIBUS DP Master Configuration"

This window is used for service purposes and is only visible if in **Tools ▸ Options ▸ IndraLogic 2G ▸ Device Editor** the option "Display Generic Configuration Views" was enabled.

☞ Please clarify any possible modifications to the parameters that can be edited with the service team.



Parameter	Type	Current Value	Prepared Value	Value	Default Value	Unit
NumberOfInputs	WORD	0		0	0	
NumberOfOutputs	WORD	0		0	0	
DpParameter						
AutoStart	BOOL	TRUE		TRUE	TRUE	
MasterDiag						
CommunicationC...	UDINT					
Ready	BOOL	TRUE		false	false	
Run	BOOL	TRUE		false	false	
BusOn	BOOL	TRUE		false	false	
ConfigLocked	BOOL	TRUE		false	false	
NewConfig	BOOL	FALSE		false	false	
Restart	BOOL	FALSE		false	false	

Fig.3-10: PROFIBUS DP master object: PROFIBUS DP Master Configuration

This window contains information about the PROFIBUS DP master parameter set. The parameters marked in the figure are from the Tab DP parameters, page 23.

Window structure

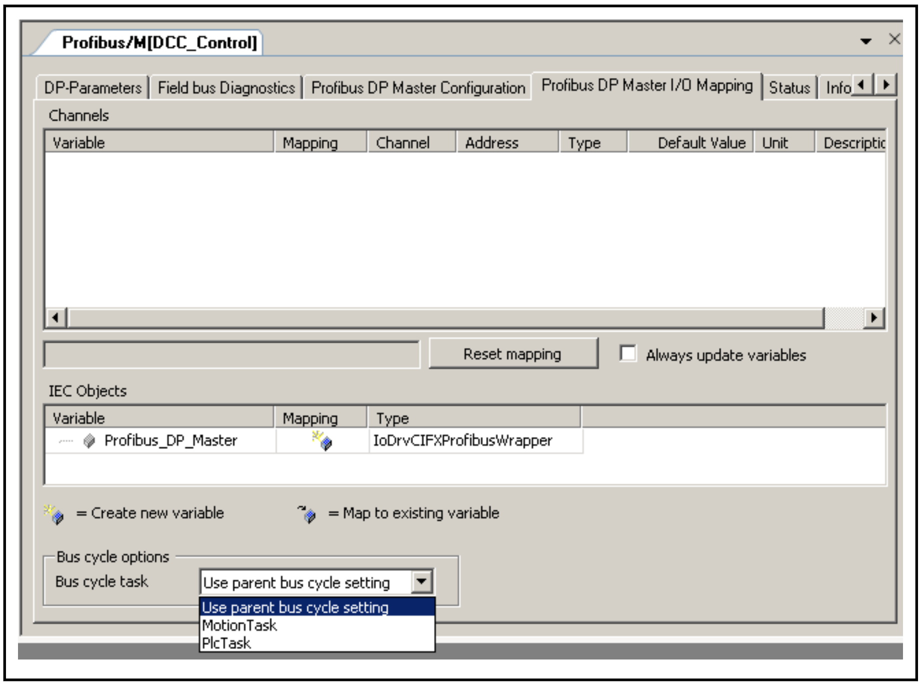
- **Parameters**: Parameter name from the device description file, cannot be edited.
- **Type**: Data type of the parameter, cannot be edited.
- **Value**: First, the standard value of the parameter is displayed, directly or as a specification of the corresponding symbolic name.

  If the parameter can be edited (this depends on the device description; parameters that cannot be edited are displayed in light gray), an input field or a selection list can be opened by double-clicking on the table field (or pressing the <space bar> in a previously selected field) where the value can be changed.

  Values are accepted with <Write parameter>.

  If the value is related to a file specification, the standard dialog for selected a file opens.
- **Default Value**: Defined value from the device description, cannot be edited.
- **Unit**: Unit for the value, e.g. "ms" for milliseconds, cannot be edited.
- **Description**: Short description of the parameter from the device description file, cannot be edited.

Tab "PROFIBUS DP Master I/O Mapping"



Variable	Mapping	Type
Profibus_DP_Master		IoDrvCIFXProfibusWrapper

Bus cycle task: Use parent bus cycle setting

Fig.3-11: PROFIBUS DP master object: I/O mapping

Channels: The "channels" area for the PROFIBUS DP master object is empty because it does not have any inputs/outputs to be mapped.

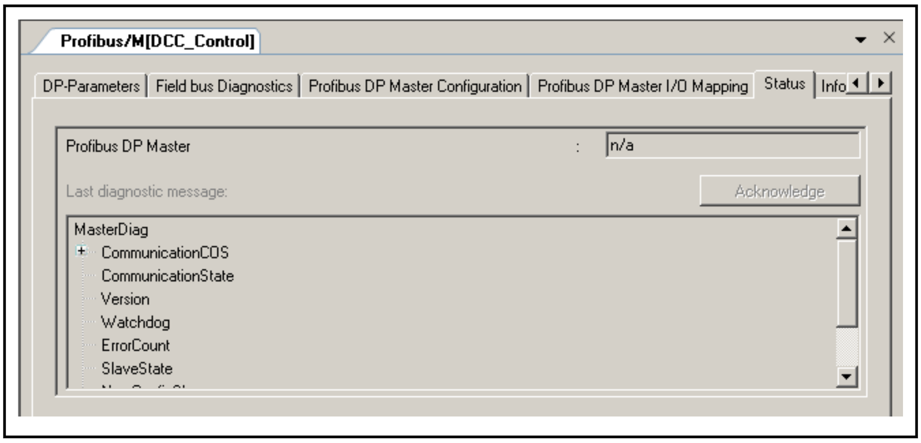
IEC objects: The data that belongs to the actual bus object can be addressed as a (global project) variable `Profibus_DP_Master_4` (PROFIBUS DP master object with the number 4) of type `IoDrvCIFXProfibusWrapper`.

Bus cycle task: By selecting a bus cycle task, the cycle of the mapping exchange for the PROFIBUS DP master can be connected to a particular task. In this task, it is useful to process the I/O data of the master as well.

Default setting: "Use parent bus cycle setting"

With this setting, the task setting made in "Bus cycle options" for the control (double click on the actual control in the Project Explorer, PLC settings) is accepted for the actual bus.

Tab "Status"



Profibus DP Master : n/a

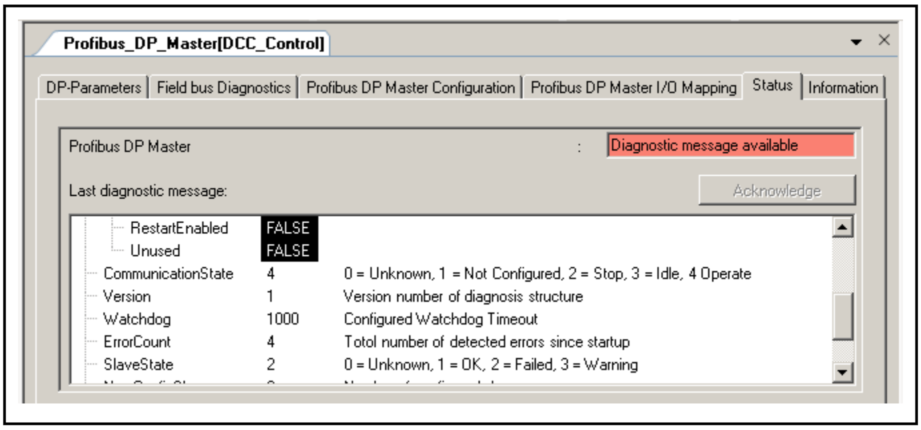
Last diagnostic message:

```
MasterDiag
 + CommunicationCOS
   CommunicationState
   Version
   Watchdog
   ErrorCount
   SlaveState
```

Fig.3-12: PROFIBUS DP master object: Status

The "Status" tab displays status information (e.g. "Running" (bus active), and "n/a" (no information available) and

specific diagnostic messages from the respective device and regarding the card used and the internal bus system.


Profibus_DP_Master[DCC_Control]
DP-Parameters
Field bus Diagnostics
Profibus DP Master Configuration
Profibus DP Master I/O Mapping
Status
Information
Acknowledge

Profibus DP Master : Diagnostic message available

Last diagnostic message:

```
      RestartEnabled   FALSE
      Unused           FALSE
  CommunicationState   4      0 = Unknown, 1 = Not Configured, 2 = Stop, 3 = Idle, 4 Operate
  Version              1      Version number of diagnosis structure
  Watchdog             1000   Configured Watchdog Timeout
  ErrorCount           4      Totol number of detected errors since startup
  SlaveState           2      0 = Unknown, 1 = OK, 2 = Failed, 3 = Warning
```

Fig.3-13: PROFIBUS DP master object: Status with present error (after confirmation)

In addition, the "Most recent diagnostic message" is displayed, which can be confirmed with "Acknowledge".

Tab "Information"

The window displays some general information from the device description file:

Name, Vendor, Categories, Version, Order number, Description, Image, if available.

3.3.3 PROFIBUS DP Master, Adding a Slave

PROFIBUS DP Master, Adding Slaves, General Information

The slaves are located in the "Periphery" library in the "ProfibusDP" folder.

Drag the required slaves from the library to the PROFIBUS DP master object.

In Project Explorer, slaves can also be added between existing slaves in this way.

Optionally, slaves can be added in the context menu via **Add ▸ Slave ▸ ...** of the PROFIBUS DP master object.

In this case, the new slave is added as the last slave under the PROFIBUS DP master object.

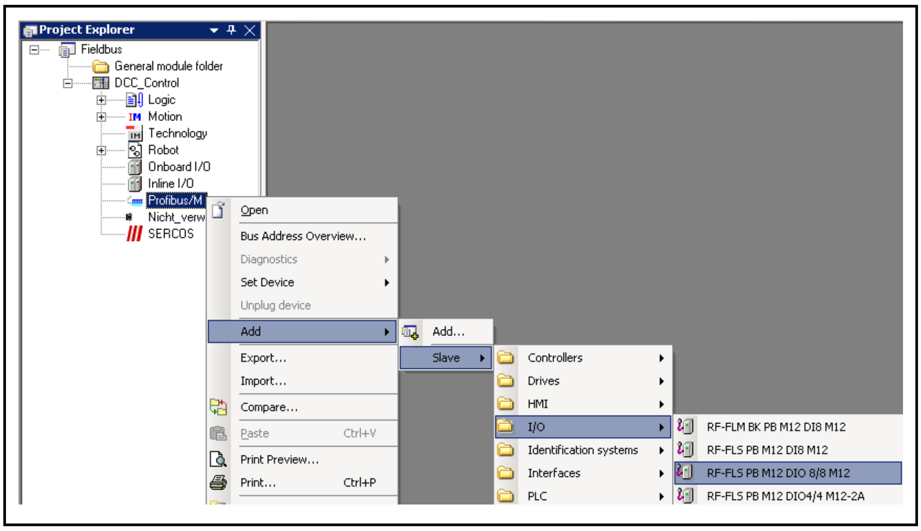


Fig.3-14: PROFIBUS DP master, adding a slave

If a required slave is not present in the library by default, it can be integrated into the library by importing its device description file using the main menu **Tools ▸ Device database...**.

See also PROFINET I/O, Reloading the Device Description File, page 89.

Slaves for connecting I/O modules

The PROFIBUS DP distinguishes two types of slaves for connecting I/O modules:

1. **Compact**: For compact slaves, the module structure is specified.

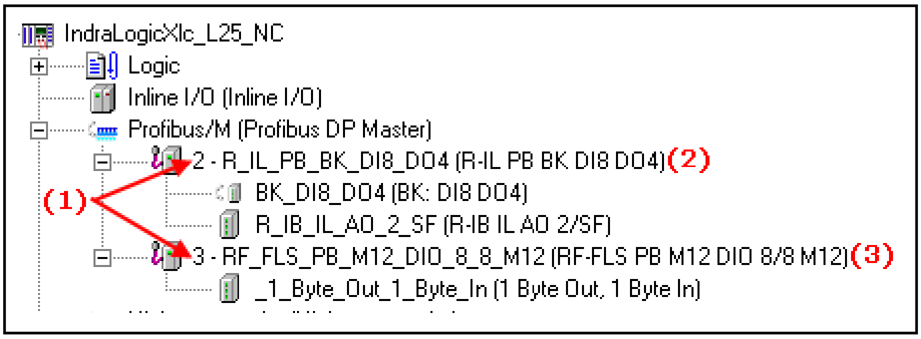
   After a slave has been added in Project Explorer, for compact slaves the modules below the slave object node are already present in their complete form. The terminals are not visible in the library.

2. **Modular**: The module structure of the slave is variable.

   The modules (terminals) can be arranged individually, although according to the device placement specifications.

   Immediately after the slave is added in Project Explorer, there are no subordinate I/O device levels for the slave.

   The modules have to be manually assigned for modular slaves. To add modules, see "Adding Modules", page 37.



IndraLogicXlc_L25_NC
Logic
Inline I/O (Inline I/O)
Profibus/M (Profibus DP Master)
2 - R_IL_PB_BK_DI8_DO4 (R-IL PB BK DI8 DO4)
BK_DI8_DO4 (BK: DI8 DO4)
R_IB_IL_AO_2_SF (R-IB IL AO 2/SF)
3 - RF_FLS_PB_M12_DIO_8_8_M12 (RF-FLS PB M12 DIO 8/8 M12)
_1_Byte_Out_1_Byte_In (1 Byte Out, 1 Byte In)

(1) Current bus addresses for the slaves
(2) Modular slave with assigned modules
(3) Compact slave

Fig.3-15: Slaves with I/O modules at the PROFIBUS DP master object

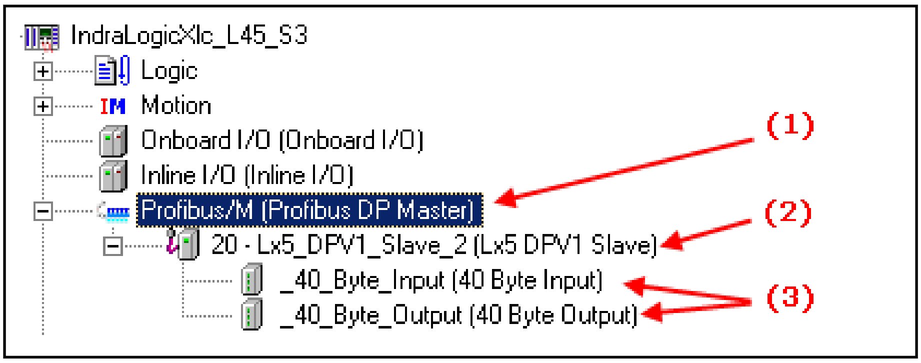
Slaves (PLC coupling modules) for connecting slave controls

To couple a control used as PROFIBUS DP slave, three PLC coupling modules are available under **Library** ▸ **Periphery** ▸ **ProfibusDP** ▸ **PLC** that allow coupling of the following controls:

- **L20 DPV1 slave** for the IndraLogic L20 DP control,
- **L40 DPV1 slave** for the IndraLogic L40 DP control,
- **Lx5 DPV1 slave** for the IndraLogic XLC L25/L45/L65 and IndraMotion MLC L25/L45/L65 controls.

When a PLC coupling module is added in the Project Explorer, a firmware component is enabled in the PROFIBUS DP master for connection to a PROFIBUS DP slave control via PROFIBUS.

Each of these PLC coupling modules has a selection of modules in the device library that release input and output variables in the mapping memory of the master for data exchange between the PROFIBUS DP master control and the PROFIBUS DP slave control.



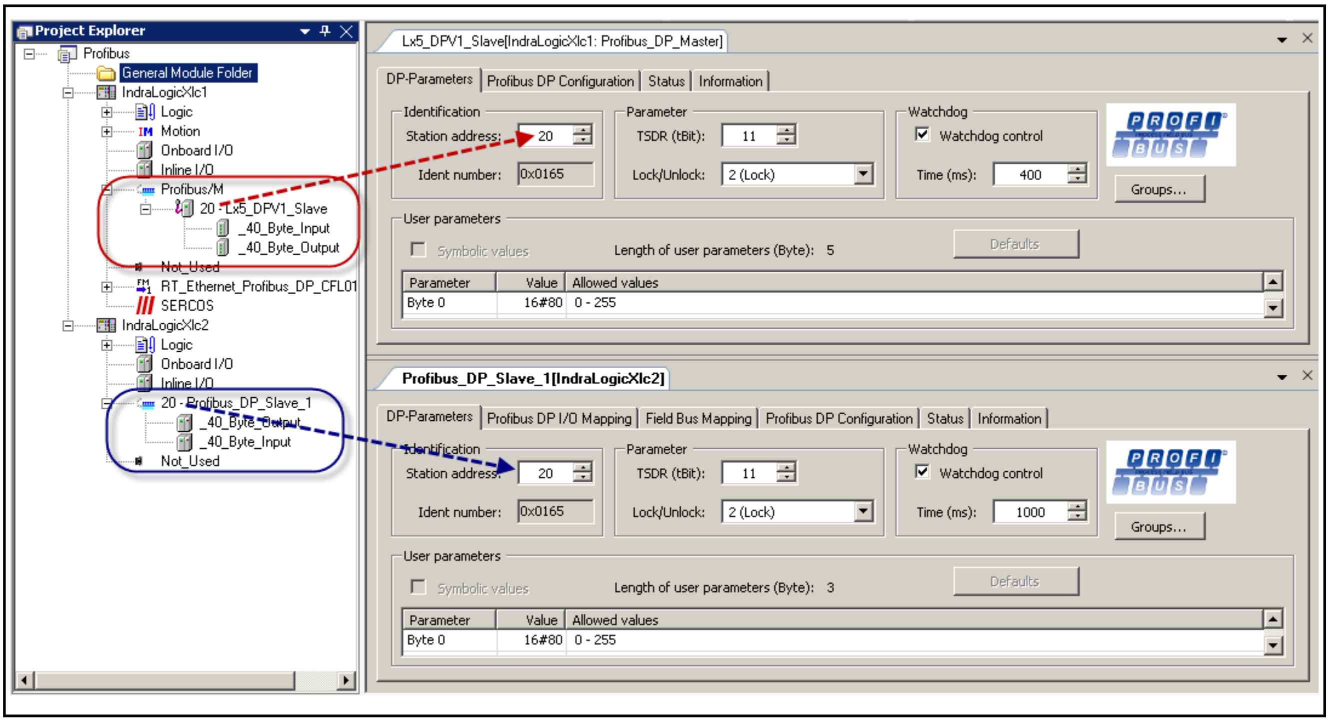
IndraLogicXlc_L45_S3
Logic
Motion
Onboard I/O (Onboard I/O)
Inline I/O (Inline I/O)
Profibus/M (Profibus DP Master)
20 - Lx5_DPV1_Slave_2 (Lx5 DPV1 Slave)
_40_Byte_Input (40 Byte Input)
_40_Byte_Output (40 Byte Output)

(1) PROFIBUS DP master object
(2) Slave for coupling onto a L25/L45/L65 control
(3) Memory modules for data exchanged between the controls

Fig.3-16: Slave with memory modules for coupling a PROFIBUS DP slave control

The figure below shows the coupling between two controls based on PROFIBUS DP.

PROFIBUS DP



Lx5_DPV1_Slave[IndraLogicXlc1: Profibus_DP_Master]

Identification
Station address: 20
Ident number: 0x0165

Parameter
TSDR (tBit): 11
Lock/Unlock: 2 (Lock)

Watchdog
☑ Watchdog control
Time (ms): 400

User parameters
□ Symbolic values
Length of user parameters (Byte): 5

Parameter	Value	Allowed values
Byte 0	16#80	0 - 255

Profibus_DP_Slave_1[IndraLogicXlc2]

Identification
Station address: 20
Ident number: 0x0165

Parameter
TSDR (tBit): 11
Lock/Unlock: 2 (Lock)

Watchdog
☑ Watchdog control
Time (ms): 1000

User parameters
□ Symbolic values
Length of user parameters (Byte): 3

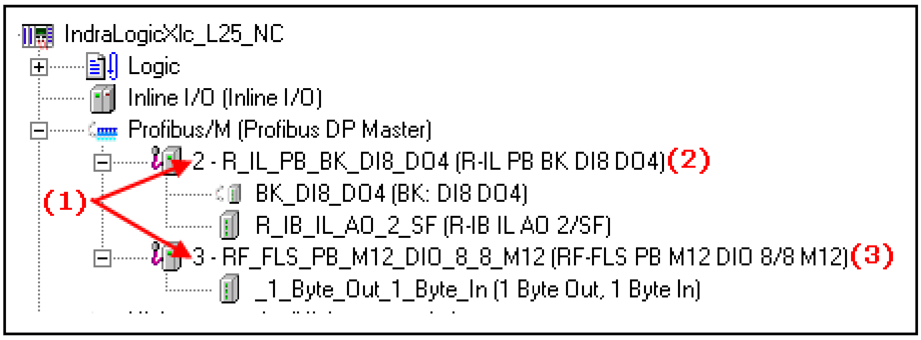
Parameter	Value	Allowed values
Byte 0	16#80	0 - 255

Fig.3-17: Computer coupling via PROFIBUS DP

Inputs to the PROFIBUS DP master side become outputs on the PROFIBUS DP slave side and vice versa. The station numbers at the coupling point are the same.

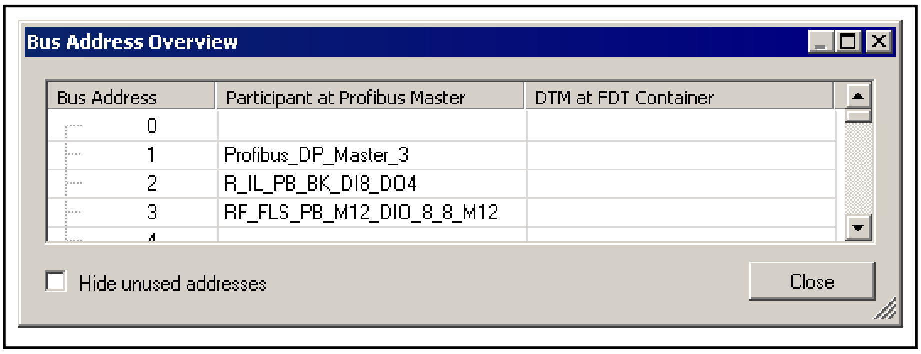
Overview of bus address...

To show the complete address assignment of all bus devices, open the context menu item **Bus address overview...** of the PROFIBUS DP master object.


IndraLogicXlc_L25_NC
Logic
Inline I/O (Inline I/O)
Profibus/M (Profibus DP Master)
2 - R_IL_PB_BK_DI8_DO4 (R-IL PB BK DI8 DO4)
(2)
BK_DI8_DO4 (BK: DI8 DO4)
R_IB_IL_AO_2_SF (R-IB IL AO 2/SF)
(1)
3 - RF_FLS_PB_M12_DIO_8_8_M12 (RF-FLS PB M12 DIO 8/8 M12)
(3)
_1_Byte_Out_1_Byte_In (1 Byte Out, 1 Byte In)

(1) Current bus addresses for the slaves
(2) Modular slave with assigned modules
(3) Compact slave

Fig.3-18: Slaves with I/O modules at the PROFIBUS DP master object



Bus Address Overview

Bus Address	Participant at Profibus Master	DTM at FDT Container
0		
1	Profibus_DP_Master_3	
2	R_IL_PB_BK_DI8_DO4	
3	RF_FLS_PB_M12_DIO_8_8_M12	
4		

☐ Hide unused addresses

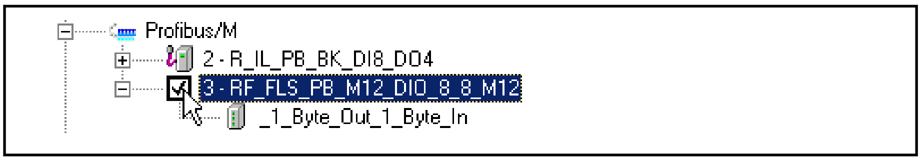
Close

Fig.3-19: Overview of bus addresses

Enabling / Disabling PROFIBUS DP Modules

The PROFIBUS DP modules configured at a control can be "enabled" and "disabled" in the Project Explorer. This applies both to I/O modules and to the complete PROFIBUS DP slave control.

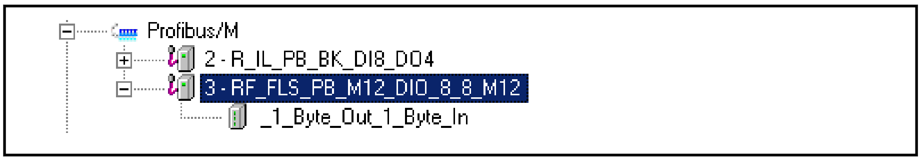
It can be achieved by selecting/deselecting the button appearing at the icon of the PROFIBUS DP module.



Profibus/M
2 - R_IL_PB_BK_DI8_DO4
☑ 3 - RF_FLS_PB_M12_DIO_8_8_M12
_1_Byte_Out_1_Byte_In

Fig.3-20: Enabling/disabling PROFIBUS DP modules

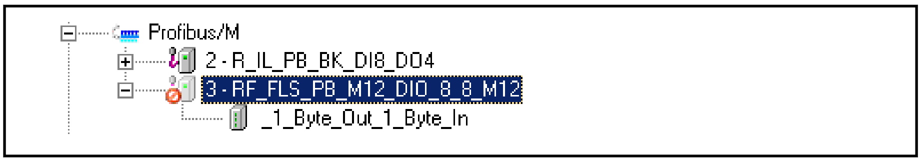
The module is enabled if the checkbox is ticked. It is taken into account in the diagnostics of the bus.



Profibus/M
2 - R_IL_PB_BK_DI8_DO4
3 - RF_FLS_PB_M12_DIO_8_8_M12
_1_Byte_Out_1_Byte_In

Fig.3-21: PROFIBUS DP with enabled modules

The module is disabled if the checkbox is unticked. It is not taken into account in the diagnostics of the bus.



Profibus/M
2 - R_IL_PB_BK_DI8_DO4
3 - RF_FLS_PB_M12_DIO_8_8_M12
_1_Byte_Out_1_Byte_In

Fig.3-22: PROFIBUS DP with disabled modules

If a new PROFIBUS DP module is created, it is enabled by default.

When loading the configuration, all enabled PROFIBUS DP modules are taken into account. Any possibly disabled PROFIBUS DP modules are ignored.

A module that is enabled but not present generates an error message in the diagnostics.

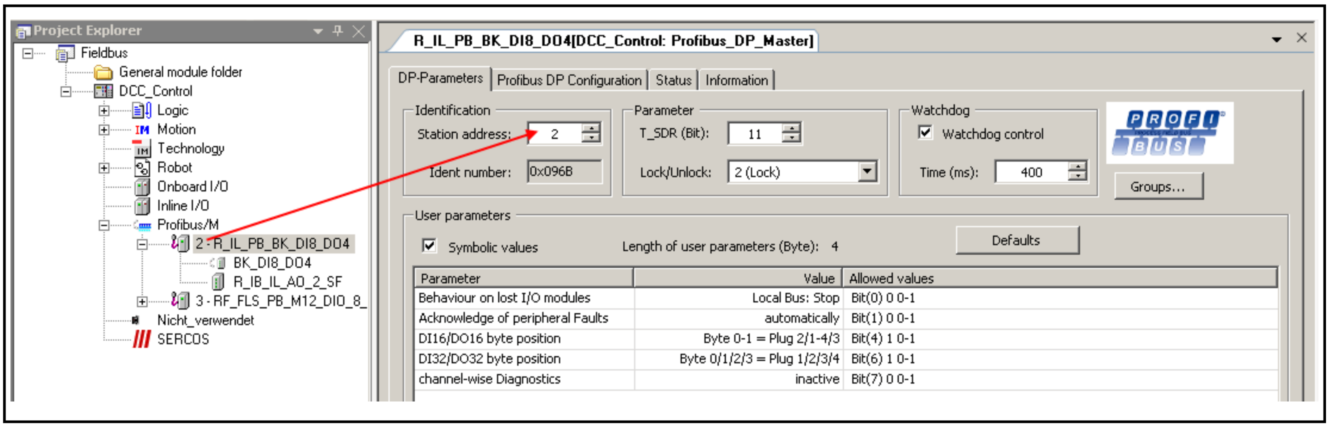
A module that is disabled and not present does not generate any error message in the diagnostics.

PROFIBUS DP Master Object, Configuring Slaves

To open the editing window, in the Project Explorer, double click on the desired slave.

PROFIBUS DP

The dialogs will inform you about the configuration of the entire slave and you can modify it if necessary.



Identification
Station address: 2
Ident number: 0x096B

Parameter
T_SDR (Bit): 11
Lock/Unlock: 2 (Lock)

Watchdog
☑ Watchdog control
Time (ms): 400

User parameters
☑ Symbolic values
Length of user parameters (Byte): 4

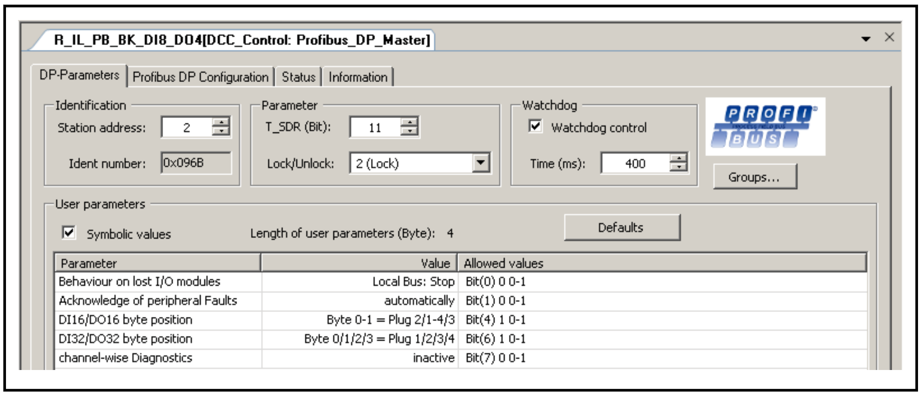
Parameter	Value	Allowed values
Behaviour on lost I/O modules	Local Bus: Stop	Bit(0) 0 0-1
Acknowledge of peripheral Faults	automatically	Bit(1) 0 0-1
DI16/DO16 byte position	Byte 0-1 = Plug 2/1-4/3	Bit(4) 1 0-1
DI32/DO32 byte position	Byte 0/1/2/3 = Plug 1/2/3/4	Bit(6) 1 0-1
channel-wise Diagnostics	inactive	Bit(7) 0 0-1

Fig.3-23: PROFIBUS DP master object, configuring slaves

Register

- DP parameters, page 34,
- PROFIBUS DP configuration, page 36, this window is used for service purposes and is only visible if the option "Show generic configuration editors" is enabled under **Tools ▸ Options ▸ IndraLogic 2G ▸ Device editor.**
- Status, page 36,
- Information, page 37.

Tab "DP Parameters"


R_IL_PB_BK_DI8_DO4[DCC_Control: Profibus_DP_Master]
DP-Parameters
Profibus DP Configuration
Status
Information
Groups...
Defaults

Identification
Station address: 2
Ident number: 0x096B

Parameter
T_SDR (Bit): 11
Lock/Unlock: 2 (Lock)

Watchdog
☑ Watchdog control
Time (ms): 400

User parameters
☑ Symbolic values
Length of user parameters (Byte): 4

Parameter	Value	Allowed values
Behaviour on lost I/O modules	Local Bus: Stop	Bit(0) 0 0-1
Acknowledge of peripheral Faults	automatically	Bit(1) 0 0-1
DI16/DO16 byte position	Byte 0-1 = Plug 2/1-4/3	Bit(4) 1 0-1
DI32/DO32 byte position	Byte 0/1/2/3 = Plug 1/2/3/4	Bit(6) 1 0-1
channel-wise Diagnostics	inactive	Bit(7) 0 0-1

Fig.3-24: PROFIBUS DP, adding slaves: DP parameters

In contrast to the DP parameters of a DP Master, those of a DP Slave are not a standard set, but are instead described individually for each device in the device description. The user can modify them in the DP parameters dialog.

Identification

- **Station address**: The address is entered automatically. It is to be adjusted according to the actual bus configuration. The address "126" is reserved for service and commissioning and should not be used.
- **Ident number**: A unique number that was assigned to this device type from the PNO (PROFIBUS user organization). This allows for a unique reference to exist between the DP Slave and the related device description file.

Parameter

- **T_SDR (Bit)**: Time Station Delay Responder. Response time which reflects the earliest point at which the slave may respond to the master.

  "Bit":

  Time unit for transferring a bit with PROFIBUS;

  Reciprocal value of the transfer rate; e.g., 1 bit at 12 MBaud=1/12.000.000 Bit/sec=83 ns.
- **Lock/Unlock**: Slave is blocked or released for other masters. Select one of the following options from the selection list:
  - **0 (T_SDR unlock)**: Min. T_SDR and slave-specific parameters can be overwritten.
  - **1 (Will be unlocked)**: Slave is released for other masters:
  - **2 (Lock)**: Slave is blocked for other masters; all parameters are accepted.
  - **3 (Unlock)**: Slave is released for other masters again.

Watchdog

- **Watchdog control** When this option is enabled, the monitoring time entered applies. If the slave is not addressed by the master within this time period, it returns to initialization status.
- **Time (ms)**: Monitoring time, relevant in the case that the "Watchdog control" option is active.

Groups

- The "Groups..." button opens the 'Group Properties' dialog.

  This dialog is used for assigning the slave to one or more of the eight possible groups.

  In contrast, the general group properties (sync mode and/or freeze mode) are defined when the master properties are configured (see 'DP parameters for the DP_Master', 'Group properties', page 23).

  You can also use the "Global Group Properties" button to access this dialog.

  The group(s) to which the slave was assigned are selected with a checkmark.

  A slave device can only be assigned to groups with properties that it supports. The related properties of the respective slave (sync mode / freeze mode) are displayed above the table. The modes supported by the device are selected with a checkmark.

User parameters:

- **User parameters** are, in addition to the basic DP parameters (see above), individual parameters of a DP Slave that are displayed here if they are defined in the device description file.

  For each parameter, the parameter table displays the parameter name, the real or the symbolic value (see below, "Symbolic value") and the permissible values that are also defined in the device description file. After using the mouse to click on the corresponding field, the parameter values can be edited in the "Value" column.
- **Symbolic values**: If symbolic names for the parameters are also specified in the device description file, this option can be activated here to display these symbolic values instead of the numeric values in the "Value" column.

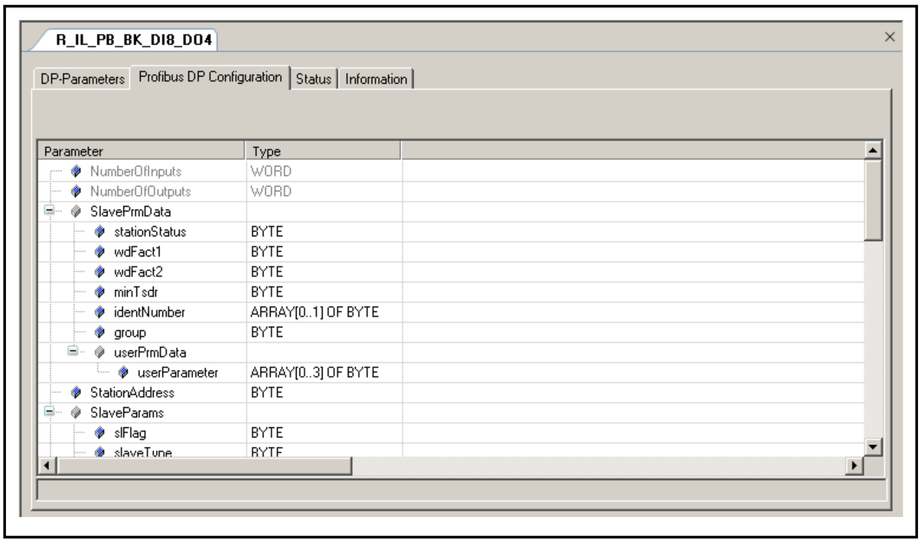
- **Length of the user parameters (bytes)**: Sum of the lengths of the user parameters defined in the device description file.
- **Defaults**:

  This button can be used to reset the values shown in the table to the standard setting.

Tab "PROFIBUS DP Configuration"

This window is used for service purposes and is only visible if in **Tools ▸ Options ▸ IndraLogic 2G ▸ Device Editor** the option "Display Generic Configuration Views" was enabled.

☞ Please clarify any possible modifications to the parameters that can be edited with the service team.


R_IL_PB_BK_DI8_DO4

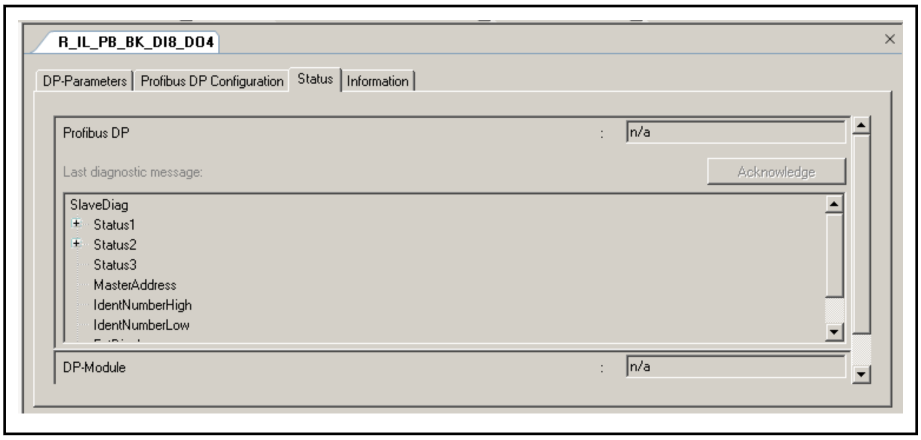
DP-Parameters | Profibus DP Configuration | Status | Information

Parameter	Type
NumberOfInputs	WORD
NumberOfOutputs	WORD
SlavePrmData	
stationStatus	BYTE
wdFact1	BYTE
wdFact2	BYTE
minTsdr	BYTE
identNumber	ARRAY[0..1] OF BYTE
group	BYTE
userPrmData	
userParameter	ARRAY[0..3] OF BYTE
StationAddress	BYTE
SlaveParams	
slFlag	BYTE
slaveType	BYTE

Fig.3-25: PROFIBUS DP, adding slaves: PROFIBUS DP Configuration

The dialog displays the module parameters in detail.

Tab "Status"


R_IL_PB_BK_DI8_DO4

DP-Parameters | Profibus DP Configuration | Status | Information

Profibus DP : n/a

Last diagnostic message:

SlaveDiag
+ Status1
+ Status2
Status3
MasterAddress
IdentNumberHigh
IdentNumberLow

DP-Module : n/a

Fig.3-26: PROFIBUS DP, adding slaves: Status

The "Status" tab displays status information (e.g., "Running" (bus active), and "n/a" (no information available).

Tab "Information"

The window displays some general information from the device description file:

Name, Vendor, Categories, Version, Order number, Description, Image, if available.

3.3.4 PROFIBUS DP Master, Adding Modules to the Slave

PROFIBUS DP Master, Adding Modules to the Slave, General Information

The modules that work with the respective slave are located in the "Peripherals" library in the "ProfibusDP" folder under the respective slave.

- I/O: I/O modules for modular slaves.
- PLC: Declaration of commonly used memory space for data exchange between the participating controls.

> ☞ I/O modules can only be added in modular structured slaves, page 30,.

Drag the required modules out of the library into the slave object.

New modules can also be added between existing modules in Project Explorer.

Optionally, modules can be added in the context menu via **Add** ▸ **Slave** ▸ **...** of the slave.

In this case, the new module is added as the last module under the slave.

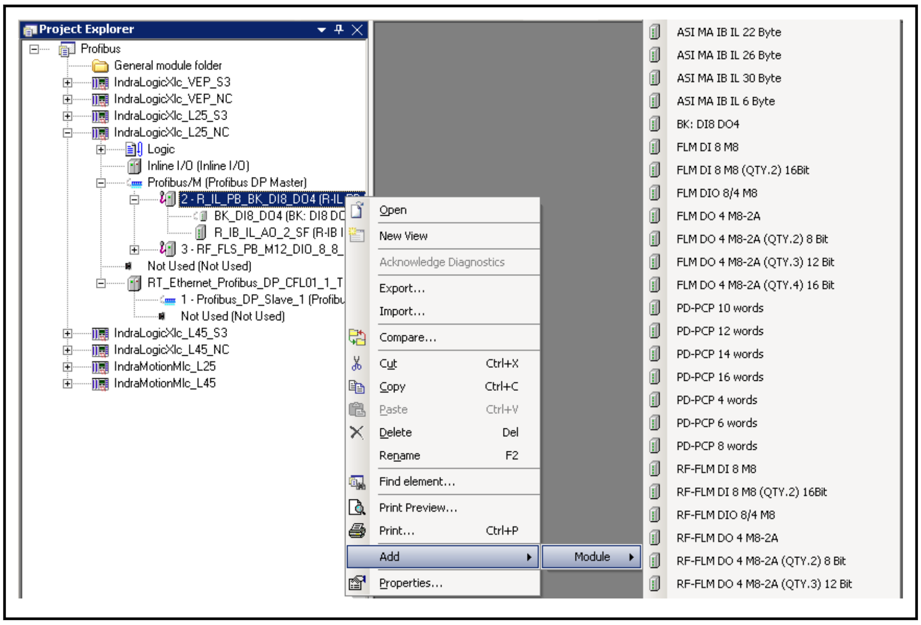


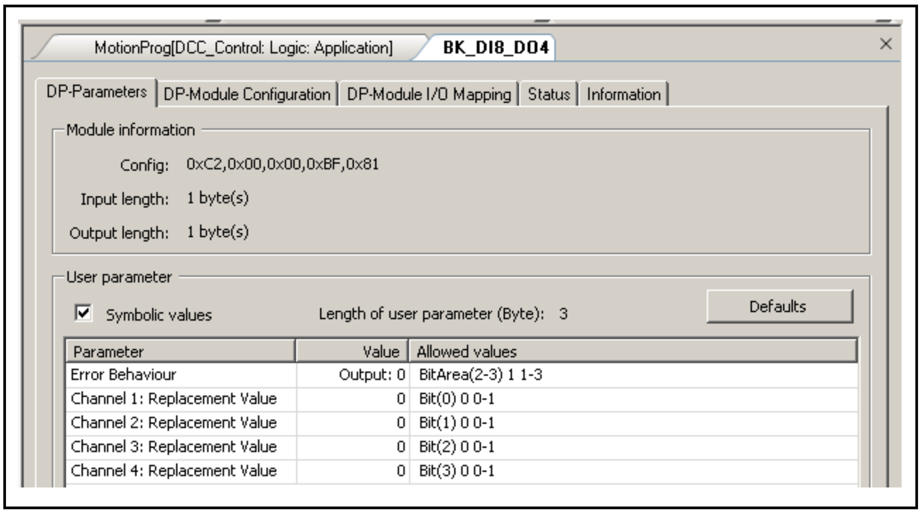
Fig.3-27: PROFIBUS DP master, adding modules to a slave

Register

- DP parameters, page 38,
- DP module configuration, page 39, this window is used for service purposes and is only visible if the option "Show generic configuration editors" is enabled under **Tools** ▸ **Options** ▸ **IndraLogic 2G** ▸ **Device editor**.
- DP_Module I/O mapping, page 39,

- Status, page 40,
- Information, page 40.

Tab "DP Parameters"



Module information

Config: 0xC2,0x00,0x00,0xBF,0x81

Input length: 1 byte(s)

Output length: 1 byte(s)

User parameter

☑ Symbolic values

Length of user parameter (Byte): 3

Parameter	Value	Allowed values
Error Behaviour	Output: 0	BitArea(2-3) 1 1-3
Channel 1: Replacement Value	0	Bit(0) 0 0-1
Channel 2: Replacement Value	0	Bit(1) 0 0-1
Channel 3: Replacement Value	0	Bit(2) 0 0-1
Channel 4: Replacement Value	0	Bit(3) 0 0-1

Fig.3-28: PROFIBUS DP master, slave modules: DP parameters

Module information

- The "Module information" panel of the "DP Parameters" tab of an input and output module in a PROFIBUS DP configuration describes the settings for the configuration (module description according to PROFIBUS DP standard) and the length of the input and output data (input length and output length) in bytes as they are defined in the device description file (GSD file).

User parameters

- **User parameters** are, in addition to the basic DP parameters (see above), individual parameters of a DP Slave that are displayed here if they are defined in the device description file.

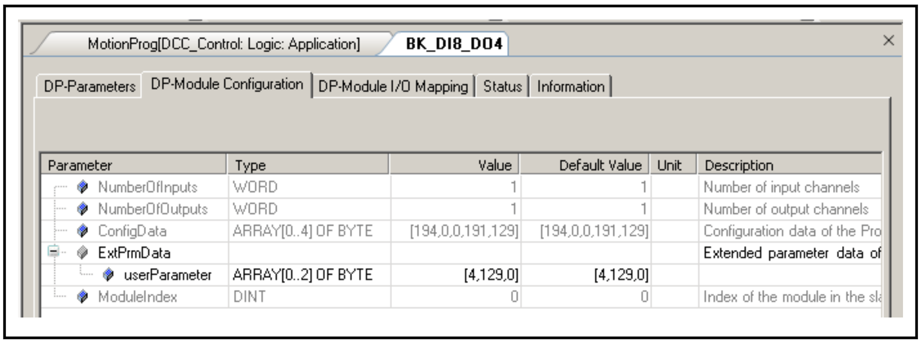
  For each parameter, the parameter table displays the parameter name, the real or the symbolic value (see below, "Symbolic value") and the permissible values that are also defined in the device description file. After using the mouse to click on the corresponding field, the parameter values can be edited in the "Value" column.
- **Symbolic values**: If symbolic names for the parameters are also specified in the device description file, this option can be activated here to display these symbolic values instead of the real values in the "Value" column.
- **Length of the user parameters (bytes)**: The length of the user parameters specifies the total length of all user parameters from the device description. After using the mouse to click on the respective field in the "Value" column, a parameter value can be edited.
- **Defaults**: This button can be used to reset the values shown in the table to the standard setting.

☞ Detailed information about the individual parameters can be found in the respective module description.

Tab "DP Module Configuration"

This window is used for service purposes and is only visible if in **Tools** ▸ **Options** ▸ **IndraLogic 2G** ▸ **Device Editor** the option "Display Generic Configuration Views" was enabled.

☞ Please clarify any possible modifications to the parameters that can be edited with the service team.



Parameter	Type	Value	Default Value	Unit	Description
NumberOfInputs	WORD	1	1		Number of input channels
NumberOfOutputs	WORD	1	1		Number of output channels
ConfigData	ARRAY[0..4] OF BYTE	[194,0,0,191,129]	[194,0,0,191,129]		Configuration data of the Pro
ExtPrmData					Extended parameter data of
userParameter	ARRAY[0..2] OF BYTE	[4,129,0]	[4,129,0]		
ModuleIndex	DINT	0	0		Index of the module in the sl

Fig.3-29: PROFIBUS DP master, slave modules: configuration

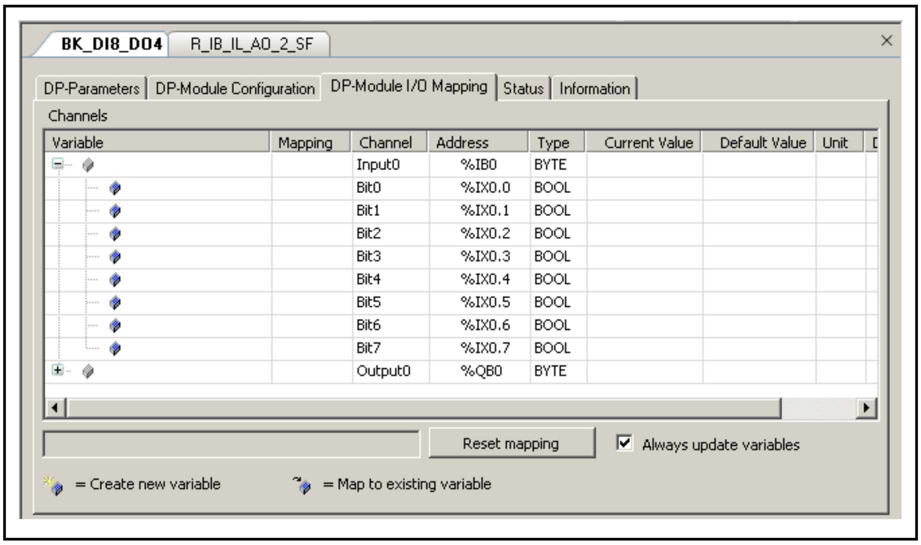
The dialog contains information on the position and size of the parameters.

Tab "DP Modules I/O Mapping"

The window is used to assign inline module inputs and outputs to variables that can be used as local or global variables in the individual POUs.

The current value of the variables is displayed in online mode.

This assignment is described in Mapping the Onboard, Inline and Field Bus Inputs and Outputs, page 139,.



Variable	Mapping	Channel	Address	Type	Current Value	Default Value	Unit	D
		Input0	%IB0	BYTE				
		Bit0	%IX0.0	BOOL				
		Bit1	%IX0.1	BOOL				
		Bit2	%IX0.2	BOOL				
		Bit3	%IX0.3	BOOL				
		Bit4	%IX0.4	BOOL				
		Bit5	%IX0.5	BOOL				
		Bit6	%IX0.6	BOOL				
		Bit7	%IX0.7	BOOL				
		Output0	%QB0	BYTE				

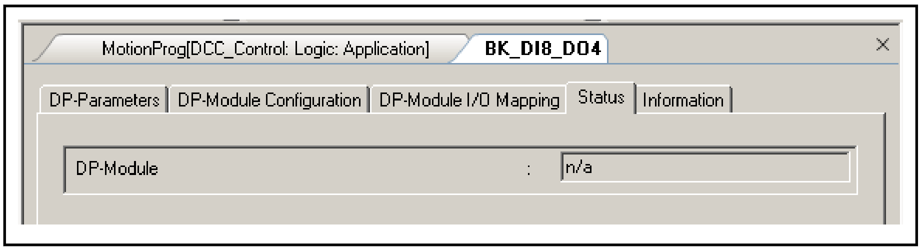
☑ Always update variables

Fig.3-30: PROFIBUS DP master, slave modules: I/O mapping

Reset mapping deletes the assignment made in the editor.

Always update variables If this option is enabled, all variables are updated in each cycle of the Bus Cycle Task, page 28, no matter whether they are used or not and whether they are mapped to an input or an output channel.

Tab "Status"



DP-Module : n/a

Fig.3-31: PROFIBUS DP master, slave modules: Status

- "DP modules":

  In online mode, the DP-Module area displays status information from the control (e.g. "Running", "Not running (n/a)").

Tab "Information"

The window displays some general information from the device description file:

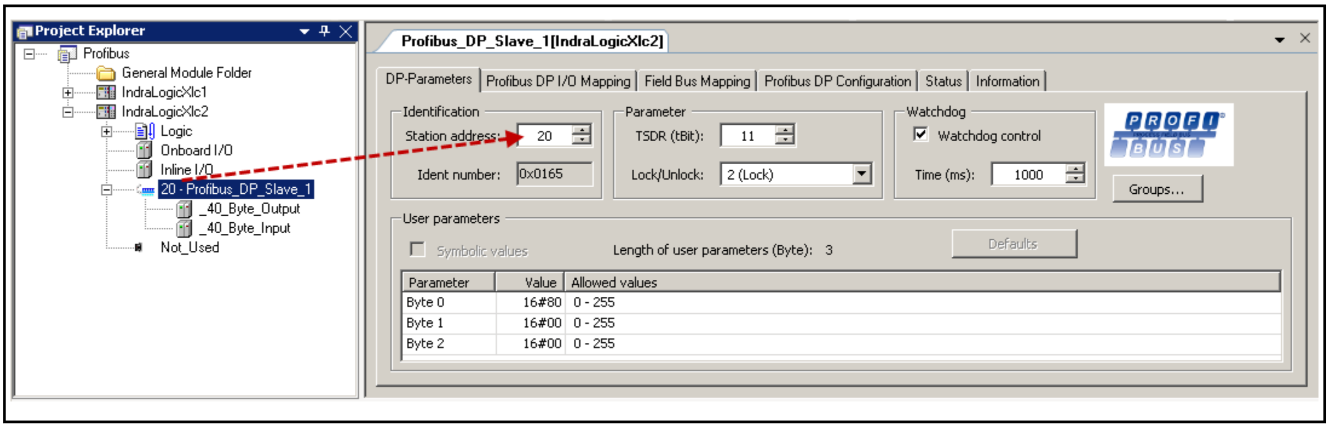
Name, Vendor, Categories, Version, Order number, Description, Image, if available.

3.3.5 PROFIBUS DP Slave

Configuring the PROFIBUS DP Slave, General Information

To open the editor window in the Project Explorer, double click on the PROFIBUS DP slave object.

The dialogs will inform you about the configuration of the entire slave, i.e the control itself as a slave, and you can modify it if necessary.



Identification
Station address: 20
Ident number: 0x0165

Parameter
TSDR (tBit): 11
Lock/Unlock: 2 (Lock)

Watchdog
☑ Watchdog control
Time (ms): 1000

User parameters
□ Symbolic values
Length of user parameters (Byte): 3

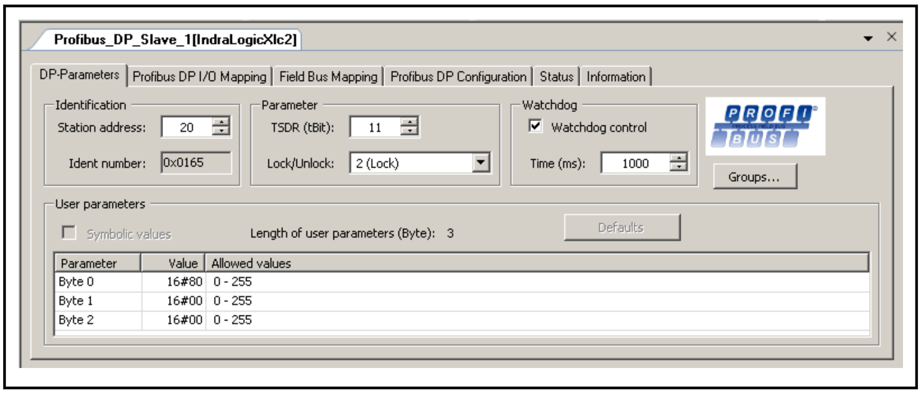
Parameter	Value	Allowed values
Byte 0	16#80	0 - 255
Byte 1	16#00	0 - 255
Byte 2	16#00	0 - 255

Fig.3-32: Configuring the PROFIBUS DP slave object

Register

- DP parameters, page 41,
- PROFIBUS DP I/O Mapping, page 42,
- Field bus mapping, page 43,
- PROFIBUS DP configuration, page 45, this window is used for service purposes and is only visible if the option "Show generic configuration editors" is enabled under **Tools ▸ Options ▸ IndraLogic 2G ▸ Device editor.**
- Status, page 46,
- Information, page 46.

Tab "DP Parameters"



Profibus_DP_Slave_1[IndraLogicXlc2]

Identification

Station address: 20

Ident number: 0x0165

Parameter

TSDR (tBit): 11

Lock/Unlock: 2 (Lock)

Watchdog

☑ Watchdog control

Time (ms): 1000

User parameters

□ Symbolic values

Length of user parameters (Byte): 3

Parameter	Value	Allowed values
Byte 0	16#80	0 - 255
Byte 1	16#00	0 - 255
Byte 2	16#00	0 - 255

Fig.3-33: PROFIBUS DP slave object: DP parameters

In contrast to the DP parameters of a DP Master, those of a DP Slave are not a standard set, but are instead described individually for each device in the device description. The user can modify them in the DP parameters dialog.

Identification

- Station address: The address is entered automatically. It is to be adjusted according to the actual bus configuration.

  The number selected is to be identical to the number of the slave object to be coupled to the PROFIBUS DP master.

- Ident number: A unique number that was assigned to this device type from the PNO (PROFIBUS user organization). This allows for a unique reference to exist between the DP Slave and the related device description file.

Parameter

- T_SDR (Bit): Time Station Delay Responder. Response time which reflects the earliest point at which the slave may respond to the master.

  Bit: Time unit for transferring a bit with PROFIBUS;

  Reciprocal value of the transfer rate; e.g., 1 bit at 12 MBaud=1/12.000.000 Bit/sec=83 ns.

- Lock/Unlock: Slave is blocked or released for other masters. Select one of the following options from the selection list:
  - 0 (T_SDR unlock): Min. T_SDR and slave-specific parameters can be overwritten.
  - 1 (Will be unlocked): Slave is released for other masters:
  - 2 (Lock): Slave is blocked for other masters; all parameters are accepted.
  - 3 (Unlock): Slave is released for other masters again.

Watchdog

- Watchdog control: When this option is enabled, the monitoring time entered applies. If the slave is not addressed by the master within this time period, it returns to initialization status.

- Time (ms): Monitoring time, relevant in the case that the "Watchdog control" option is active.

Groups:

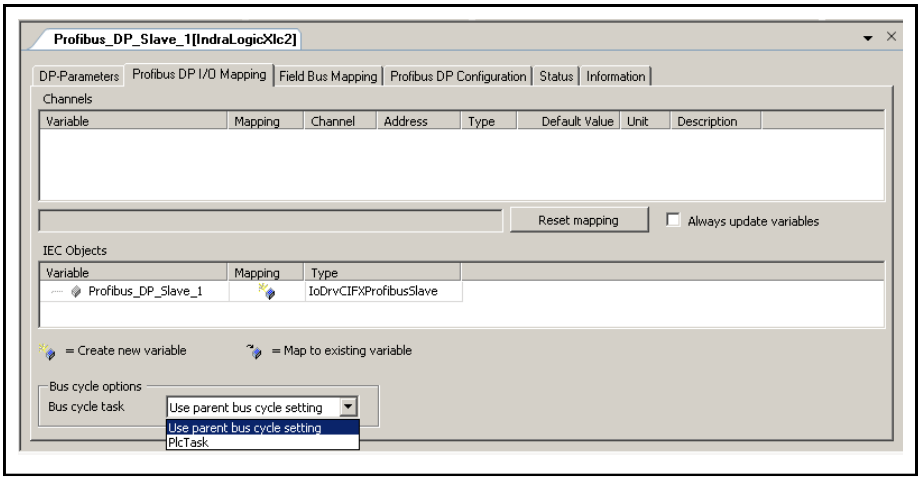
### projected ###

User parameters:

- **User parameters** are, in addition to the basic DP parameters (see above), individual parameters of a DP Slave that are displayed here if they are defined in the device description file.

  For each parameter, the parameter table displays the parameter name, the real or the symbolic value (see below, "Symbolic value") and the permissible values that are also defined in the device description file. After using the mouse to click on the corresponding field, the parameter values can be edited in the "Value" column.
- **Symbolic values**: If symbolic names for the parameters are also specified in the device description file, this option can be activated here to display these symbolic values instead of the numeric values in the "Value" column.
- **Length of the user parameters (bytes)**: Sum of the lengths of the user parameters defined in the device description file.
- **Defaults**: This button can be used to reset the values shown in the table to the standard setting.

Tab "PROFIBUS DP I/O Mapping"



Variable	Mapping	Channel	Address	Type	Default Value	Unit	Description

Variable	Mapping	Type
Profibus_DP_Slave_1		IoDrvCIFXProfibusSlave

Bus cycle task: Use parent bus cycle setting

Fig.3-34: PROFIBUS DP slave object: PROFIBUS DP I/O Mapping

Channels: The "Channels" panels for the PROFIBUS DP slave object is empty because it does not have any inputs/outputs to be mapped.

The "IEC Objects" area displays information on the PROFIBUS master driver, which you can call up in the statement section.

IEC objects: The data that belongs to the actual bus object can be addressed as a (global project) variable `Profibus_DP_Slave_1` (PROFIBUS DP slave object with number 1) of type `IoDrvCIFXProfibusSlave`.

Bus cycle task: By selecting a bus cycle task, the cycle of the mapping exchange for the PROFIBUS DP slave can be connected to a particular task. In this task, it is useful to process the I/O data of the slave as well.

Default setting: "Use parent bus cycle setting"

With this setting, the task setting made in "Bus cycle options" for the control (double click on the actual control in the Project Explorer, PLC settings) is accepted for the actual bus.

Tab "Field Bus Mapping"

Field bus mapping

Field bus mapping enables access to control variables from a parent field bus master.

The following field buses are supported here:

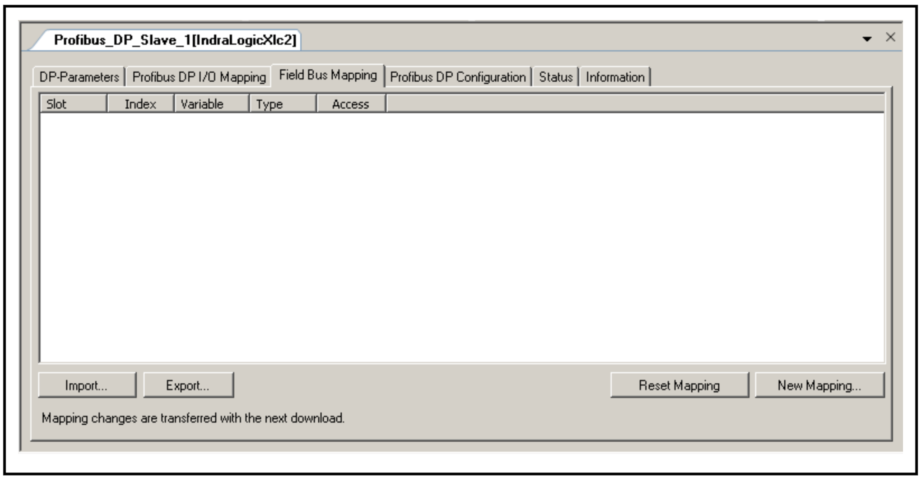
- PROFIBUS DP
- PROFINET I/O
- EtherNet/IP

To access a control variable from a parent master, an address must first be assigned to these variables (mapping). The parent master can use this address to access the control data using acyclic services.

To use field bus mapping, the control has to be configured accordingly as a bus device, e.g. as PROFIBUS DP slave.

Configuration

The "Field bus mapping" tab is located in the device editor of the respective device connection. Addresses can be assigned to individual variables here in a table.



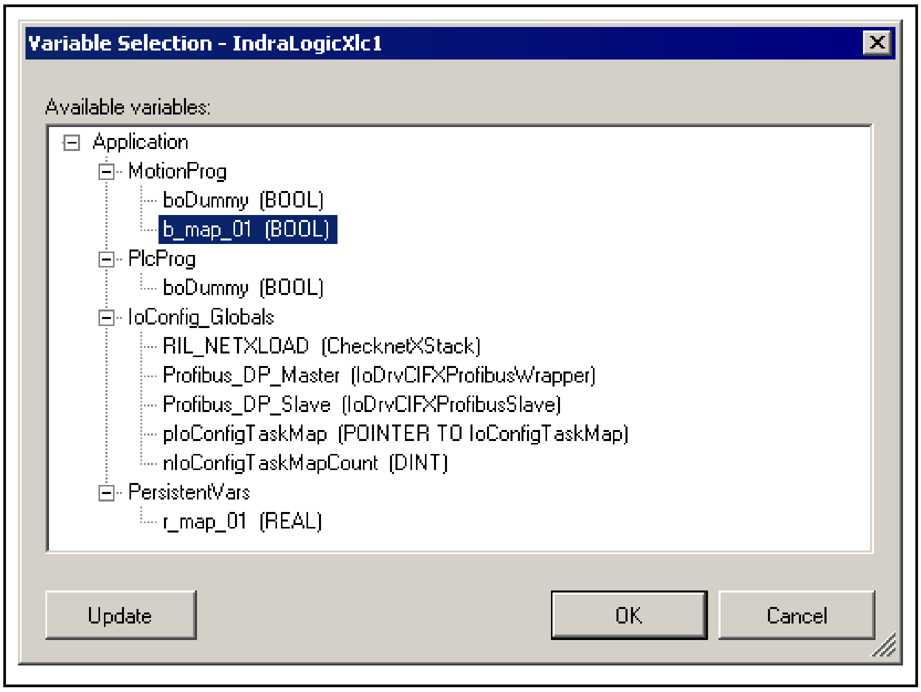
Mapping changes are transferred with the next download.

Fig.3-35: PROFIBUS DP slave: Field bus mapping

Creating new mapping

A new entry can be added using the "New mapping" button. A dialog opens fro selecting a variable.

Variables declared within the application can be selected:



Available variables:

```
Application
  MotionProg
    boDummy (BOOL)
    b_map_01 (BOOL)
  PlcProg
    boDummy (BOOL)
  IoConfig_Globals
    RIL_NETXLOAD (ChecknetXStack)
    Profibus_DP_Master (IoDrvCIFXProfibusWrapper)
    Profibus_DP_Slave (IoDrvCIFXProfibusSlave)
    pIoConfigTaskMap (POINTER TO IoConfigTaskMap)
    nIoConfigTaskMapCount (DINT)
  PersistentVars
    r_map_01 (REAL)
```

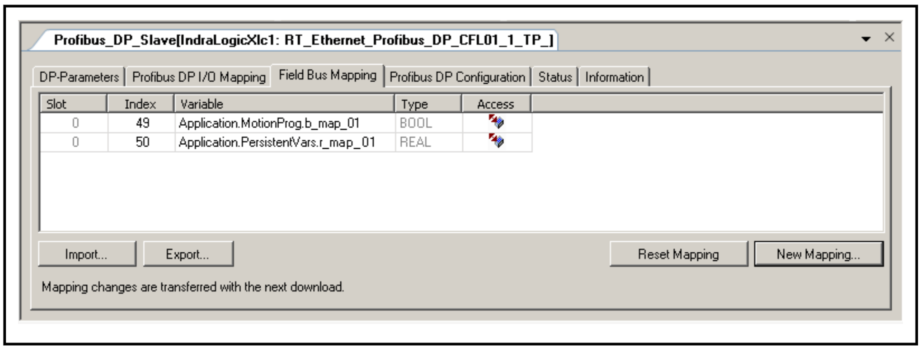
Fig.3-36: PROFIBUS DP slave: field bus mapping, selecting variables

If the selection is confirmed with "OK", a new entry is created in the table for the selected variable.

In addition, the variable is added to the symbol configuration.

The variable section only contains variables that were present at the most recent compilation. If new variables have been added to the PLC program since that time, they are only visible after a new compilation is performed. This can be performed with the "Update" button.

Address, variable and access can be changed afterward by clicking on the corresponding table cell. Gray columns cannot be edited.



Slot	Index	Variable	Type	Access
0	49	Application.MotionProg.b_map_01	BOOL	⇄
0	50	Application.PersistentVars.r_map_01	REAL	⇄

Mapping changes are transferred with the next download.

Fig.3-37: PROFIBUS DP slave: field bus mapping, variables accepted

The address can be specified in the first columns in the table, but the values in the grayed out fields cannot be changed.

The address of the PROFIBUS DP consists of slot and index. The slot is always 0. The index has to be within the range from 49 to 32767.

The Variable column displays the variables' instance path.

The Type column displays the variables' data type.

The Access column displays the access rights for the variables.

Reset mapping

The "Reset Mapping" button can be used to delete all of the table entries.

Import/Export

To save an existing mapping in a file, use the "Export…" button. It can be read out later using the "Import…" button.

This file is a simple text file.

For this reason, this file can be created manually without a previous export.

The export file contains the name for the field bus "EtherNet IP", "ProfinetI/O“, "ProfibusDP" and a table (see program listings).

The address is divided into several columns and consists of the following:

- ProfibusDP: Slot, index,
- ProfinetI/O: Slot, Subslot, Index,
- EthernetIP: Class, Instance, Attribute.

This table also includes the variable name, the data type and the access rights of the mapped variables.

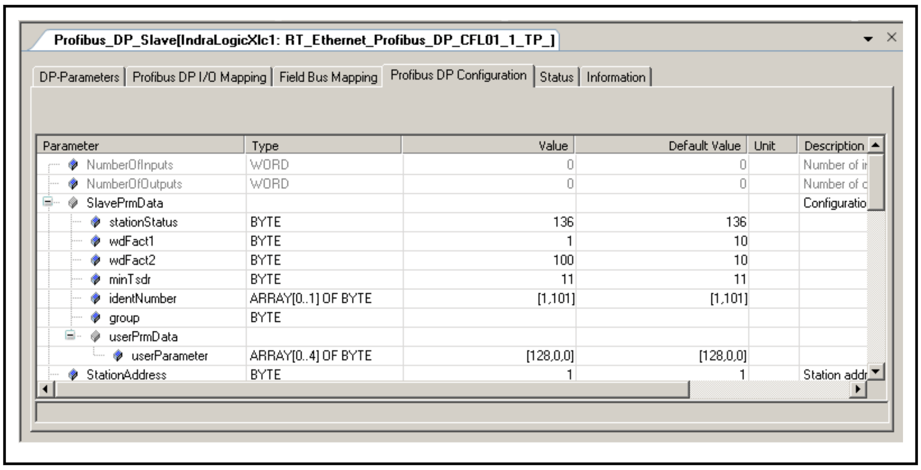
```
ProfibusDP
  # Slot    Index Variablename                          Datatyp Access right
     0      49    Application.MotionProg.b_map_01       BOOL    ReadWrite
     0      50    Application.PersistentVars.r_map_01   REAL    ReadWrite
```

Fig.3-38: Excerpt from the export file, separator TAB:

Tab "PROFIBUS DP Configuration"

This window is used for service purposes and is only visible if in **Tools** ▸ **Options** ▸ **IndraLogic 2G** ▸ **Device Editor** the option "Display Generic Configuration Views" was enabled.

☞ Please clarify any possible modifications to the parameters that can be edited with the service team.

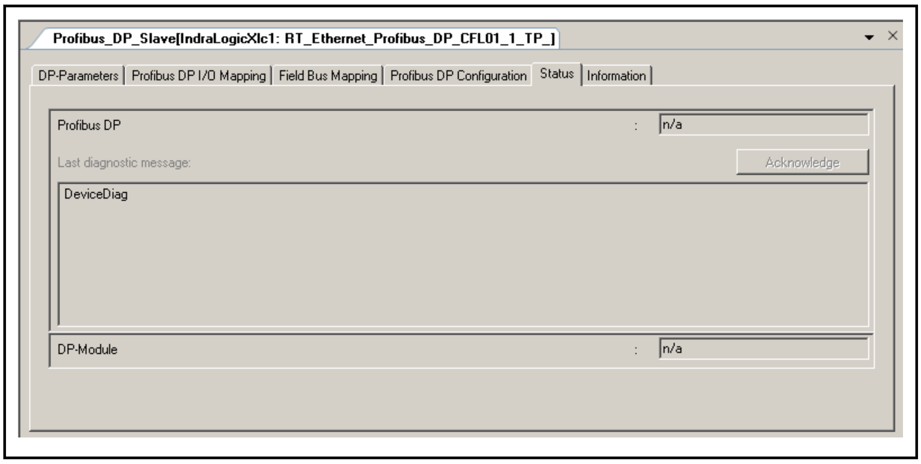

Profibus_DP_Slave[IndraLogicXlc1: RT_Ethernet_Profibus_DP_CFL01_1_TP_]
DP-Parameters
Profibus DP I/O Mapping
Field Bus Mapping
Profibus DP Configuration
Status
Information

Parameter	Type	Value	Default Value	Unit	Description
NumberOfInputs	WORD	0	0		Number of ir
NumberOfOutputs	WORD	0	0		Number of c
SlavePrmData					Configuratio
stationStatus	BYTE	136	136		
wdFact1	BYTE	1	10		
wdFact2	BYTE	100	10		
minTsdr	BYTE	11	11		
identNumber	ARRAY[0..1] OF BYTE	[1,101]	[1,101]		
group	BYTE				
userPrmData					
userParameter	ARRAY[0..4] OF BYTE	[128,0,0]	[128,0,0]		
StationAddress	BYTE	1	1		Station addr

Fig.3-39: PROFIBUS DP slave object: PROFIBUS DP Configuration

☞ When the bus is running, modified parameters can be transferred by clicking button "Write parameter".

Tab "Status"


Profibus_DP_Slave[IndraLogicXlc1: RT_Ethernet_Profibus_DP_CFL01_1_TP_]
DP-Parameters
Profibus DP I/O Mapping
Field Bus Mapping
Profibus DP Configuration
Status
Information
Acknowledge
DeviceDiag

Profibus DP : n/a

Last diagnostic message:

DP-Module : n/a

Fig.3-40: PROFIBUS DP slave object: Status

The "Status" tab displays status information (e.g. "Running" (bus active) and "n/a" (no information available)) and specific diagnostic messages from the respective device and regarding the card used and the internal bus system.

Tab "Information"

The window displays some general information from the device description file:

Name, Vendor, Categories, Version, Order number, Description, Image, if available.

3.3.6 PROFIBUS DP Slave, Adding Modules to the Slave

PROFIBUS DP Slave, Adding Modules to the Slave, General Information

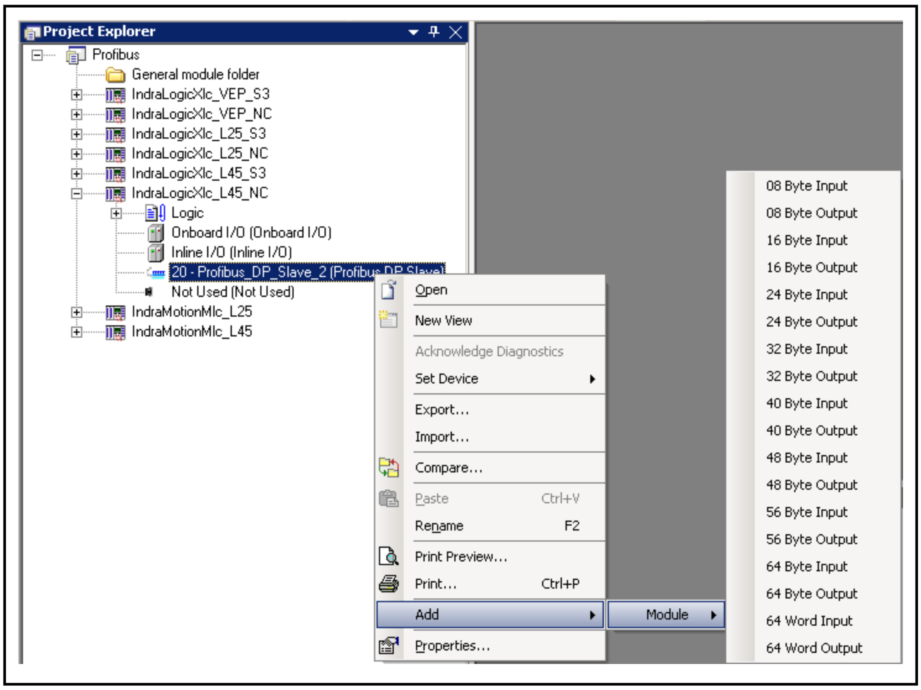


Fig.3-41: PROFIBUS DP slave object: adding modules

The interface between a PROFIBUS DP slave and PROFIBUS DP master is implemented as a common memory space. Its size can differ based on requirements.

For this reason, the various modules above are available.

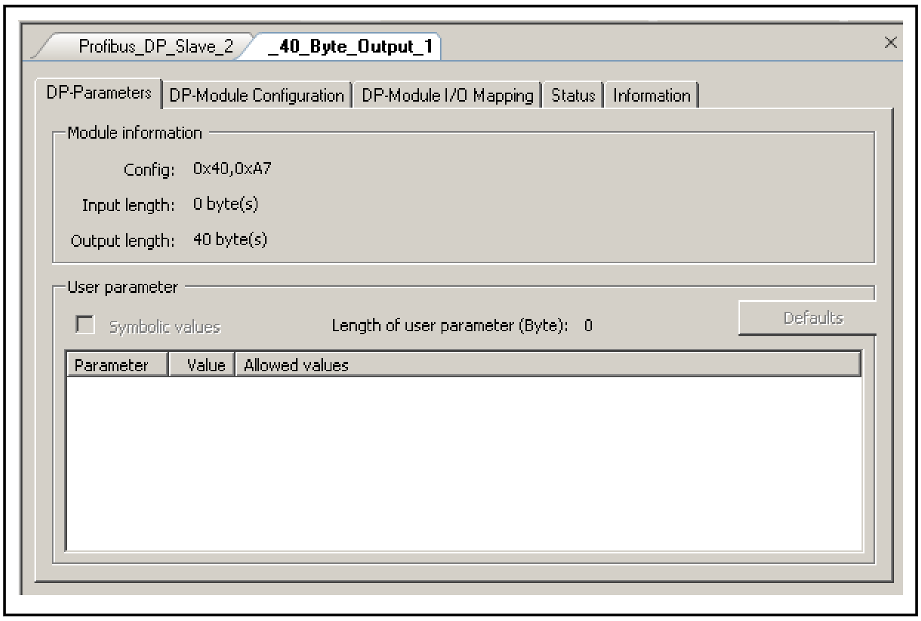
☞ Inputs to the PROFIBUS DP master side become outputs on the PROFIBUS DP slave side and vice versa.

The station numbers at the coupling point are the same.

Register

- DP parameters, page 47,
- DP module configuration, page 48, this window is used for service purposes and is only visible if the option "Show generic configuration editors" is enabled under **Tools ▸ Options ▸ IndraLogic 2G ▸ Device editor**.
- DP modules I/O mapping, page 48,
- Status, page 49,
- Information, page 49.

Tab "DP Parameters"



Config: 0x40,0xA7

Input length: 0 byte(s)

Output length: 40 byte(s)

□ Symbolic values

Length of user parameter (Byte): 0

Fig.3-42: PROFIBUS DP slave, slave modules: DP parameters

Module information

- The "Module information" panel of the "DP Parameters" tab of an input and output module in a PROFIBUS DP configuration describes the settings for the configuration (module description according to PROFIBUS DP standard) and the length of the input and output data (input length and output length) in bytes as they are defined in the device description file (GSD file).

User parameters:

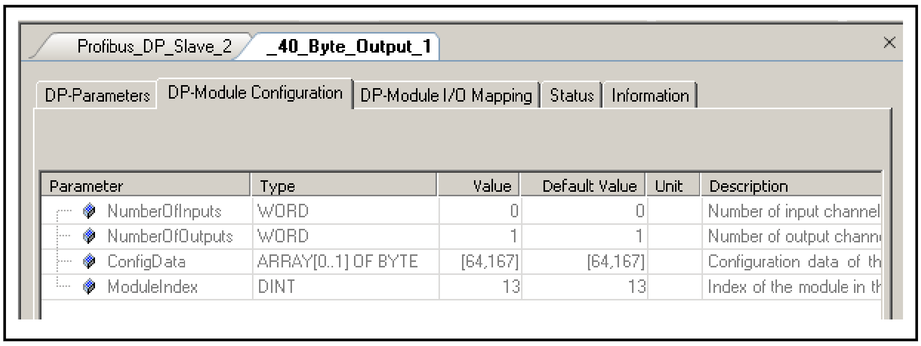
None.

☞ Detailed information about the individual parameters can be found in the respective module description.

Tab "DP Module Configuration"

This window is used for service purposes and is only visible if in **Tools** ▸ **Options** ▸ **IndraLogic 2G** ▸ **Device Editor** the option "Display Generic Configuration Views" was enabled.

☞ Please clarify any possible modifications to the parameters that can be edited with the service team.



Parameter	Type	Value	Default Value	Unit	Description
NumberOfInputs	WORD	0	0		Number of input channel
NumberOfOutputs	WORD	1	1		Number of output chann
ConfigData	ARRAY[0..1] OF BYTE	[64,167]	[64,167]		Configuration data of th
ModuleIndex	DINT	13	13		Index of the module in th

Fig.3-43: PROFIBUS DP slave, slave modules: configuration

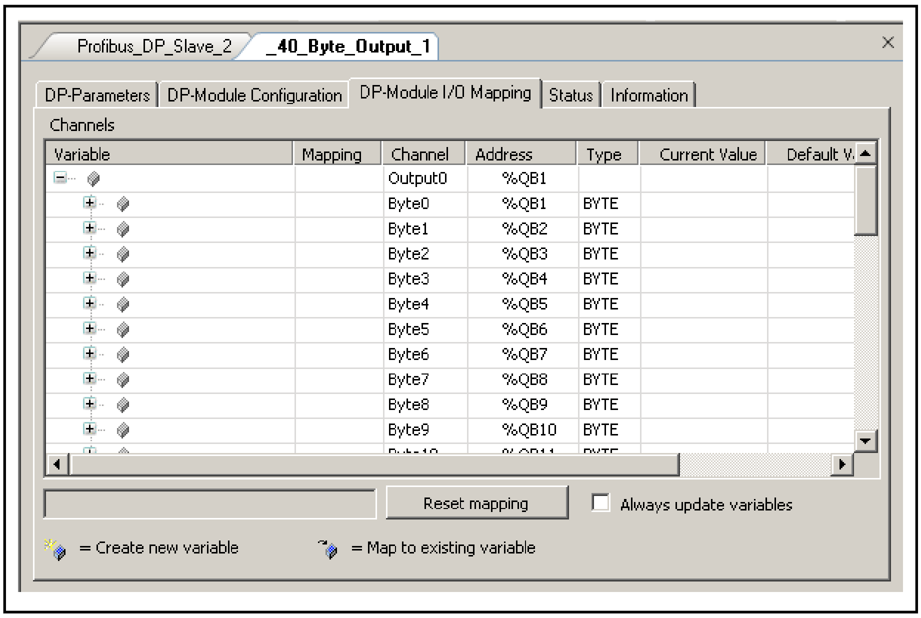
The dialog contains information on the position and size of the parameters.

Tab "DP Modules I/O Mapping"

The window is used to assign module inputs and outputs to variables that can be used as local or global variables in the individual POUs.

The current value of the variables is displayed in online mode.

This assignment is described in Mapping the Onboard, Inline and Field Bus Inputs and Outputs, page 139,.



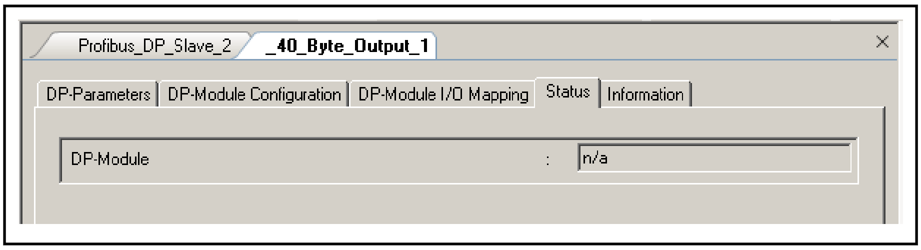
Variable	Mapping	Channel	Address	Type	Current Value	Default V
		Output0	%QB1			
		Byte0	%QB1	BYTE		
		Byte1	%QB2	BYTE		
		Byte2	%QB3	BYTE		
		Byte3	%QB4	BYTE		
		Byte4	%QB5	BYTE		
		Byte5	%QB6	BYTE		
		Byte6	%QB7	BYTE		
		Byte7	%QB8	BYTE		
		Byte8	%QB9	BYTE		
		Byte9	%QB10	BYTE		

Fig.3-44: PROFIBUS DP slave, slave modules: I/O mapping

Reset mapping: Deletes the assignment made in the editor.

Always update variables: If this option is enabled, all variables are updated in each cycle of the Bus Cycle Task, page 28,, no matter whether they are used or not and whether they are mapped to an input or an output channel.

Tab "Status"



DP-Module : n/a

Fig.3-45: PROFIBUS DP slave, slave modules: Status

"DP modules":

In online mode, the DP-Module area displays status information from the control (e.g. "Running", "Not running (n/a)").

Tab "Information"

The window displays some general information from the device description file:

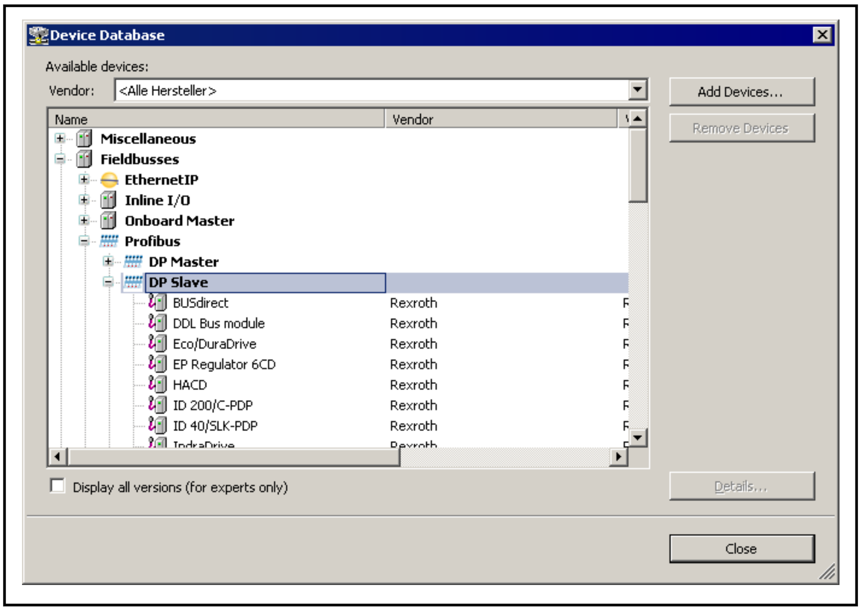
Name, Vendor, Categories, Version, Order number, Description, Image, if available.

3.3.7 PROFIBUS DP, Reloading the Device Description File

To add devices in IndraWorks, please carry out the following steps:

1. Click on **Tools ▸ Device database...** in the main menu to open the "Device database" dialog.
2. In the "Device database" dialog, click on the "Add devices" button.

   The "Install device description" file selection dialog opens:



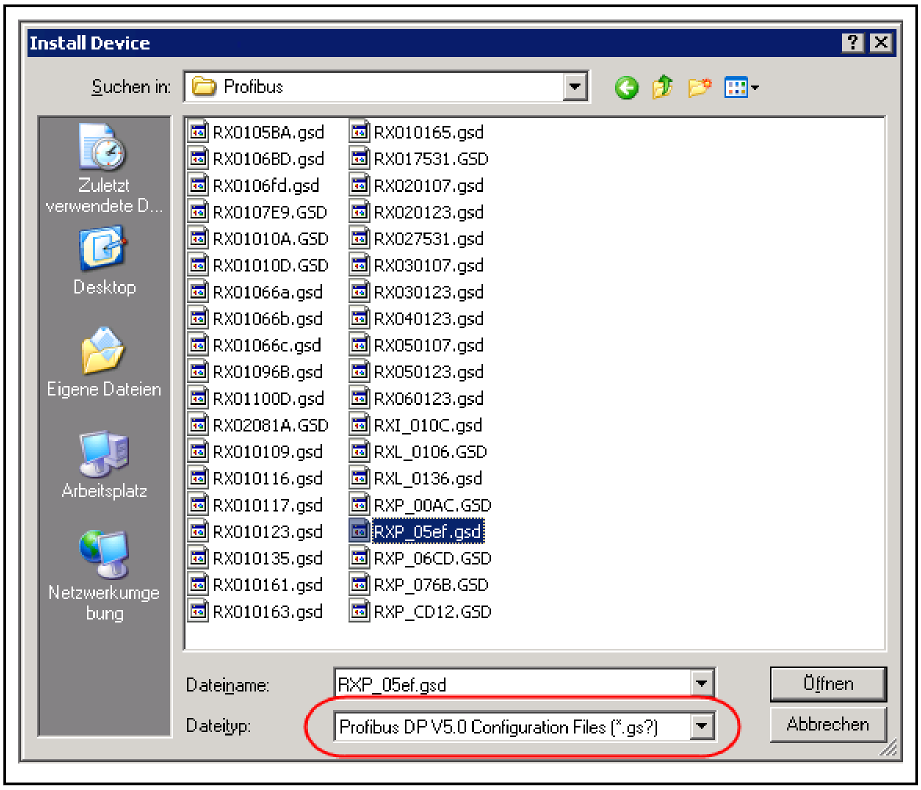
Available devices:
Vendor: <Alle Hersteller>

Name	Vendor
Miscellaneous	
Fieldbusses	
EthernetIP	
Inline I/O	
Onboard Master	
Profibus	
DP Master	
DP Slave	
BUSdirect	Rexroth
DDL Bus module	Rexroth
Eco/DuraDrive	Rexroth
EP Regulator 6CD	Rexroth
HACD	Rexroth
ID 200/C-PDP	Rexroth
ID 40/SLK-PDP	Rexroth
IndraDrive	Rexroth

☐ Display all versions (for experts only)

Fig.3-46: Device Database, add device

3. Select the device description files to be installed. Make sure that the correct file type is listed in the "File type" selection field.



RX0105BA.gsd
RX0106BD.gsd
RX0106fd.gsd
RX0107E9.GSD
RX01010A.GSD
RX01010D.GSD
RX01066a.gsd
RX01066b.gsd
RX01066c.gsd
RX01096B.gsd
RX01100D.gsd
RX02081A.GSD
RX010109.gsd
RX010116.gsd
RX010117.gsd
RX010123.gsd
RX010135.gsd
RX010161.gsd
RX010163.gsd
RX010165.gsd
RX017531.GSD
RX020107.gsd
RX020123.gsd
RX027531.gsd
RX030107.gsd
RX030123.gsd
RX040123.gsd
RX050107.gsd
RX050123.gsd
RX060123.gsd
RXI_010C.gsd
RXL_0106.GSD
RXL_0136.gsd
RXP_00AC.GSD
RXP_05ef.gsd
RXP_06CD.GSD
RXP_076B.GSD
RXP_CD12.GSD

Dateiname: RXP_05ef.gsd

Dateityp: Profibus DP V5.0 Configuration Files (*.gs?)

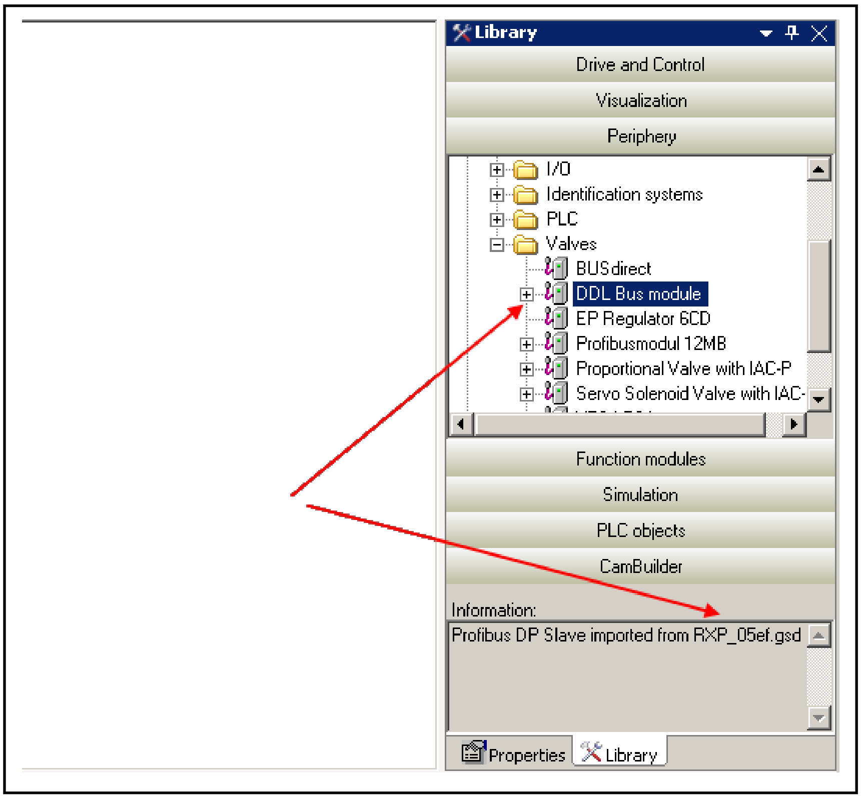
Fig.3-47: Installing device descriptions

4. Start the installation procedure by clicking on "Open".

   After completed installation, a dialog appears confirming that all device description files have been imported. The imported devices appear below "Available devices" and are highlighted there:

5. Confirm the "Install Device Descriptions" message with "OK".
6. Close the "Device database" dialog with "Close".

   The newly installed devices are also displayed in the library:



Profibus DP Slave imported from RXP_05ef.gsd

Fig.3-48: Imported device displayed in the library

For more details on importing device files, please refer to:

- Device Database, page 135.

4 PROFINET I/O

4.1 PROFINET I/O Controller

4.1.1 Terms and Abbreviations

Controller The PROFINET IO field bus master is called **controller**.

Device The PROFINET IO field bus slaves are called **devices**.

Station name The criterion for addressing a device is the device name (= station name).

GSDML file This is the device description with which the PROFINET IO devices are made known to IndraWorks.

4.1.2 PROFINET I/O Controller Features

The current implementation of the PROFINET I/O controller includes the following functionalities:

- Cyclic data traffic
- GSDML file import into the device database
- Bus scan of PROFINET I/O devices
- PROFINET I/O device-related connection status in the interface
- Diagnostics of the PROFINET I/O controller via FB
- Supported protocols
  - RTC (real-time cyclic) class 1
  - RTA (real-time acyclic)
  - DCP (discovery and configuration)

Future extensions are planned for the following functionalities

- PROFINET I/O device-related status via FB
- DHCP, SNMP

Function/Characteristic	Value
Max. number of devices, cycle time < 4 ms	25
Max. number of devices, cycle time >= 4 ms	128
Max. amount of modules per device	64
Max. number of submodules	32
Baud rate	100 Mbit/s
Auto negotiation/ autocrossing	Yes
Min. cycle time	1 ms
Max. amount of cyclic input data	3072 bytes
Max. amount of cyclic output data	3072 bytes
Max. amount of cyclic input data per device	1024 bytes
Max. amount of cyclic output data per device	1024 bytes
Max. acyclic telegram data per device / telegram	1392 bytes
Max. acyclic telegram data per device / request	4096 bytes

Fig.4-1: Technical data

4.2 PROFINET I/O Device

4.2.1 Terms and Abbreviations

Controller The PROFINET IO field bus master is called **controller**.

Device The PROFINET IO field bus slaves are called **devices**.

Station name The criterion for addressing a device is the device name (= station name).

GSDML file This is the device description with which the PROFINET IO devices are made known to IndraWorks.

4.2.2 PROFINET I/O Device Features

The current implementation of the PROFINET I/O device includes the following functionalities:

- Cyclic data traffic
- Acyclic data traffic
- PROFINET I/O device-related connection status in the interface
- Diagnostics of the PROFINET I/OO device via FB
- Supported protocols:
  - RTC (real-time cyclic) class 1
  - RTA (real-time acyclic)
  - DCP (discovery and configuration)
- MRP / Redundancy class 1 (optional) (Ring redundancy with a switching time of 200 ms in preparation)

Future extensions are planned for the following functionalities:

- Alarms
- DHCP, SNMP

Function/Characteristic	Value
Max. number at I/O modules	16 slots (independent of direction)
Max. number of submodules	1
Baud rate	100 Mbit/s
Auto negotiation/ autocrossing	Yes
Min. cycle time	1 ms
Max. amount of cyclic input data	1024 bytes
Max. amount of cyclic output data	1024 bytes
Max. length of consistent data blocks	128 bytes

Fig.4-2: Technical data

4.3 Configuring PROFINET I/O

4.3.1 Configuring PROFINET I/O, Overview

The following controls allow implementing PROFINET I/O:

- IndraLogic XLC L25
  - With sercos III:

PROFINET I/O controller / PROFINET I/O device, function modules

– Without sercos III:

PROFINET I/O controller / PROFINET I/O device, onboard

PROFINET I/O controller / PROFINET I/O device, function modules

- IndraMotion MLC L25: PROFINET I/O controller / PROFINET I/O device, function modules
- IndraLogic XLC L45/L65 and IndraMotion MLC L45/L65

  PROFINET I/O controller / PROFINET I/O device, onboard

  PROFINET I/O controller / PROFINET I/O device, function modules

The interface respectively available at the control is represented as a PROFINET I/O object in the Project Explorer.

The PROFINET I/O object can either be configured when creating the control or via the context menu item **Set device** as PROFINET I/O controller or PROFINET I/O device.

Installing and adding PROFINET I/O devices

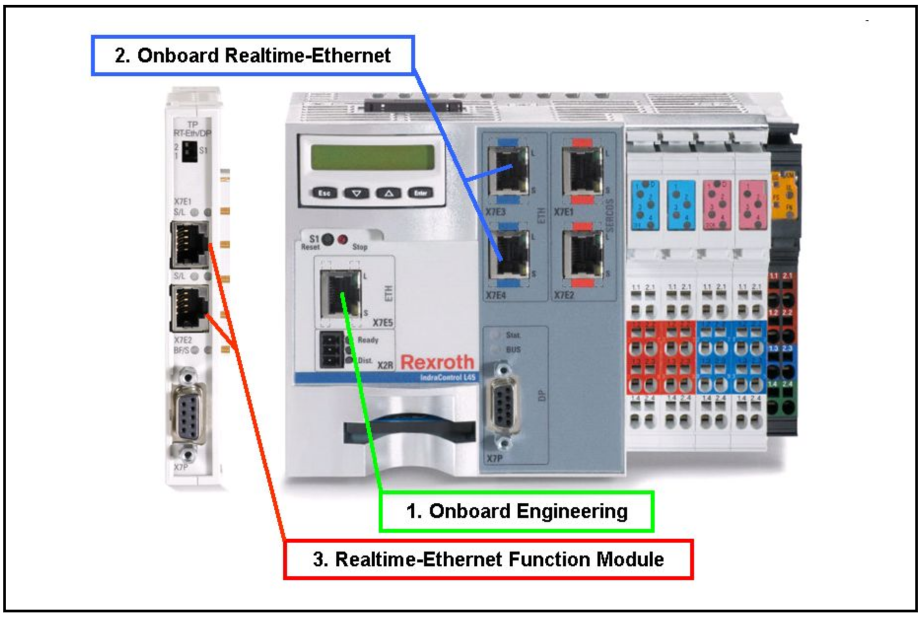

2. Onboard Realtime-Ethernet
1. Onboard Engineering
3. Realtime-Ethernet Function Module
Rexroth

Fig.4-3: IndraMotion MLC L65 with real-time EtherNet function module (CFL01.1-TP)

Interface 1, Onboard Engineering, is reserved for connecting a local computer or the enterprise network.

Interfaces 2 and 3 can each be operated either as a controller or as a device. Both connections for each interface are equivalent (switch functionality).

When being a PROFINET I/O controller with I/O blocks and PROFINET I/O drives, the control contains the simplest topology.

☞ It has to be ensured that a ring is only closed by a "redundancy manager" when PROFINET participants are wired.

If there is no such manager in a closed ring, there will be a "broadcast storm", with the result that the field bus fails and diagnostic messages, if any, will not be instrumental.

PROFINET I/O

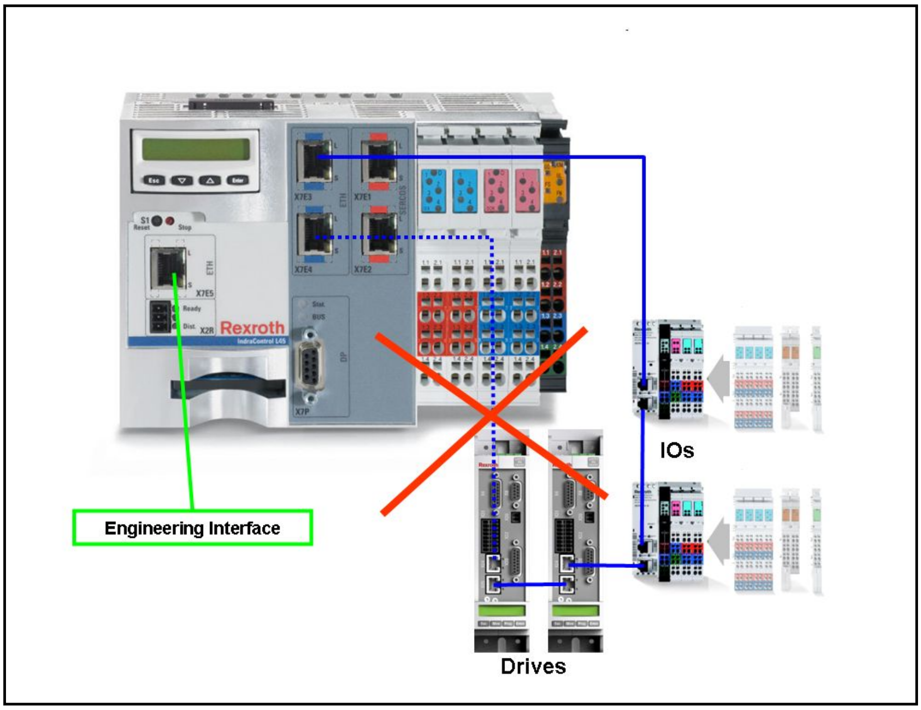
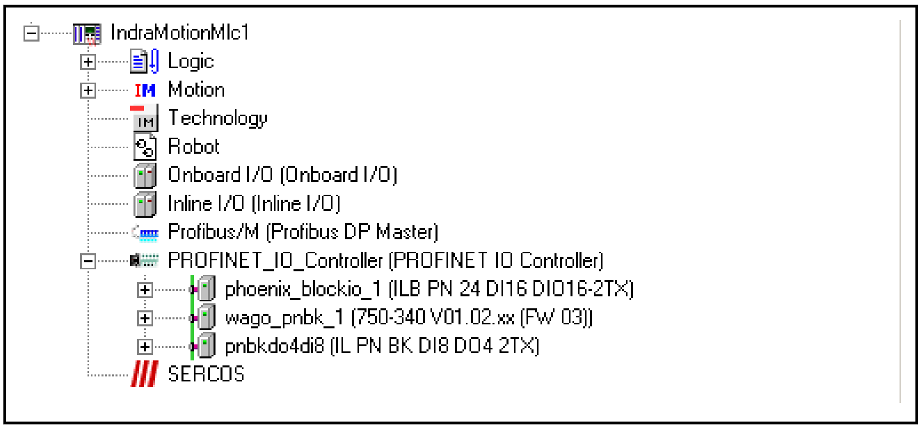


Fig.4-4: IndraMotion L65 as a PROFINET I/O controller with I/Os and drives



```
⊟ IndraMotionMlc1
   ⊞ Logic
   ⊞ Motion
     Technology
     Robot
     Onboard I/O (Onboard I/O)
     Inline I/O (Inline I/O)
     Profibus/M (Profibus DP Master)
   ⊟ PROFINET_IO_Controller (PROFINET IO Controller)
      ⊞ phoenix_blockio_1 (ILB PN 24 DI16 DIO16-2TX)
      ⊞ wago_pnbk_1 (750-340 V01.02.xx (FW 03))
      ⊞ pnbkdo4di8 (IL PN BK DI8 DO4 2TX)
     SERCOS
```

Fig.4-5: Example: PROFINET I/O configuration in Project Explorer (without drive)

The following topology shows a system consisting of a PROFINET I/O controller and a PROFNET I/O device, each with local drives and I/O blocks.

The left controller is an onboard controller with respect to its I/O blocks and via the controller function module relative to the second (and other possible) control(s).

The right control is an onboard controller with respect to its I/O blocks and via the device function module relative to the left control.

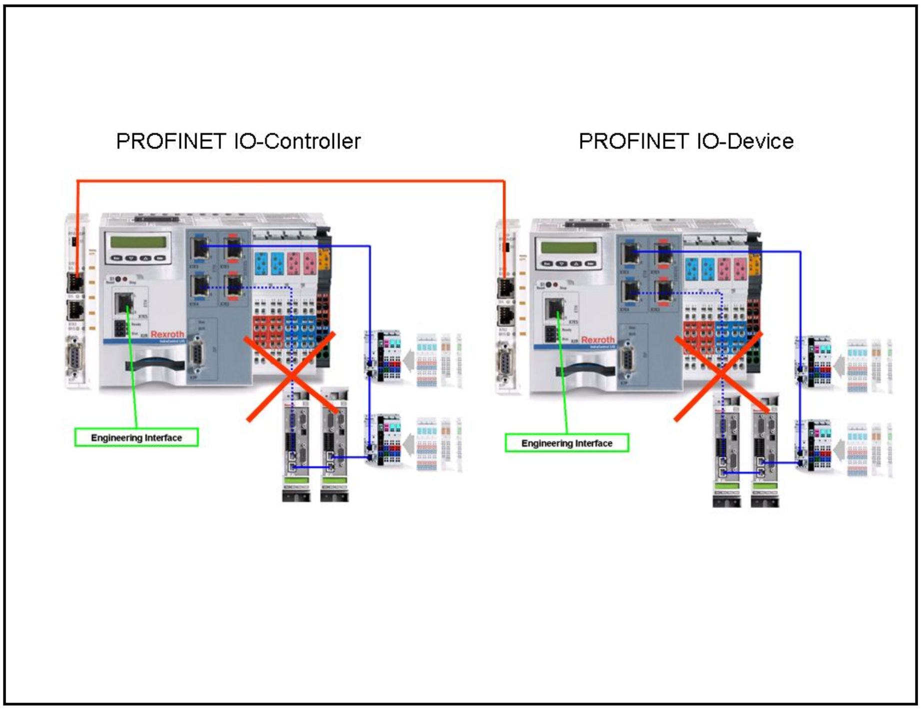
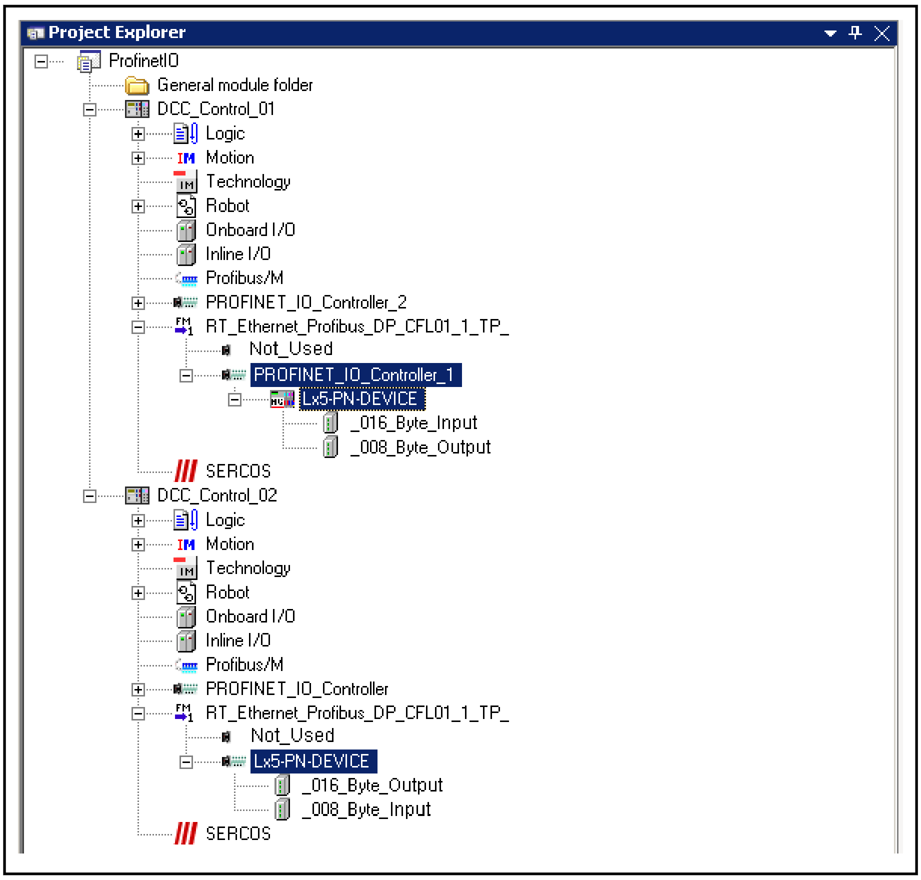


Fig.4-6: System with PROFINET I/O controller and PROFINET I/O device, each with local drives and I/O blocks



```
Project Explorer
⊟ ProfinetIO
    General module folder
  ⊟ DCC_Control_01
      ⊞ Logic
      ⊞ Motion
        Technology
      ⊞ Robot
        Onboard I/O
        Inline I/O
        Profibus/M
      ⊞ PROFINET_IO_Controller_2
      ⊟ RT_Ethernet_Profibus_DP_CFL01_1_TP_
            Not_Used
          ⊟ PROFINET_IO_Controller_1
              ⊟ Lx5-PN-DEVICE
                    _016_Byte_Input
                    _008_Byte_Output
        SERCOS
  ⊟ DCC_Control_02
      ⊞ Logic
      ⊞ Motion
        Technology
      ⊞ Robot
        Onboard I/O
        Inline I/O
        Profibus/M
      ⊞ PROFINET_IO_Controller
      ⊟ RT_Ethernet_Profibus_DP_CFL01_1_TP_
            Not_Used
          ⊟ Lx5-PN-DEVICE
                _016_Byte_Output
                _008_Byte_Input
        SERCOS
```

Fig.4-7: Example: PROFINET I/O configuration with 2 MLC in the Project Explorer

The desired topology can be transferred to the IndraWorks project after the required controls have been created with the desired periphery (context menu **Set Device**) either

- manually, using drag and drop from the device library or using the context menu **Add** or
- if the control system is already available, the online PROFINET I/O controller for controllers with **Search Devices**.

To do this, see

- Scanning for devices, page 84, (scanning).
- Reload device description file, page 89.

The device editor for the PROFINET I/O configuration can be opened by double-clicking on a PROFINET I/O object in the project tree.

PROFINET I/O controller

- PNIO master parameters, page 60
- Field bus diagnostics, page 62, is only visible if the option "Show generic configuration editors" is enabled under **Tools ▸ Options ▸ IndraLogic 2G ▸ Device editor**.
- PROFINET I/O controller configuration, page 63, this window is used for service purposes and is only visible if the option "Show generic configuration editors" is enabled under **Tools ▸ Options ▸ IndraLogic 2G ▸ Device editor**.
- PROFINET I/O controller I/O mapping, page 64
- Status, page 65
- Information, page 65

PROFINET I/O, module for coupling a PROFINET I/O device

- PNIO Parameters, page 67
- PNIO identification, page 68
- PNIO configuration, page 70, this window is used for service purposes and is only visible if the option "Show generic configuration editors" is enabled under **Tools ▸ Options ▸ IndraLogic 2G ▸ Device editor**.
- Status, page 70
- Information, page 70.

PROFINET I/O device

- PNIO parameters, page 71
- PNIO identification, page 74
- PNIO configuration, page 74, this window is used for service purposes and is only visible if the option "Show generic configuration editors" is enabled under **Tools ▸ Options ▸ IndraLogic 2G ▸ Device editor**.
- PNIO I/O Mapping, page 76
- Field bus mapping, page 76
- Status, page 79
- Information, page 79

PROFINET I/O modules

- PNIO Parameters, page 80
- PNIO module configuration, page 81, this window is used for service purposes and is only visible if the option "Show generic configuration

editors" is enabled under **Tools ▸ Options ▸ IndraLogic 2G ▸ Device editor**.

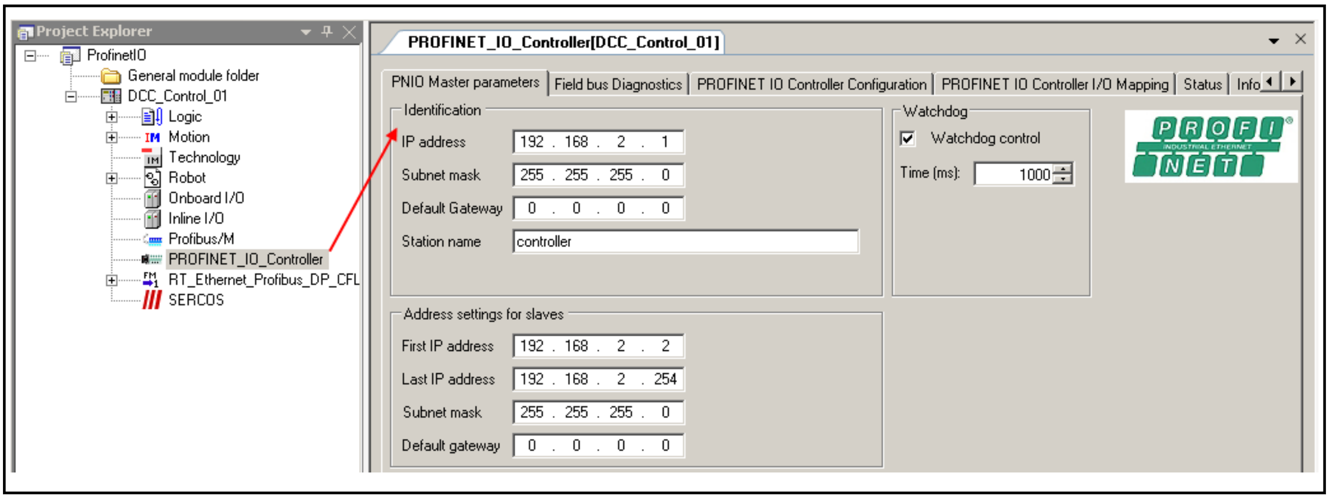
- PNIO modules I/O mapping, page 82
- Status, page 82
- Information, page 82

4.3.2 PROFINET I/O Controller

Configuring a PROFINET I/O Controller Object, General Information

To open the editing window in the Project Explorer, double click on the PROFINET I/O controller object.

The dialogs will inform you about the configuration of the entire PROFINET I/O bus and you can modify it if necessary.



Identification

IP address: 192 . 168 . 2 . 1

Subnet mask: 255 . 255 . 255 . 0

Default Gateway: 0 . 0 . 0 . 0

Station name: controller

Watchdog

☑ Watchdog control

Time (ms): 1000

Address settings for slaves

First IP address: 192 . 168 . 2 . 2

Last IP address: 192 . 168 . 2 . 254

Subnet mask: 255 . 255 . 255 . 0

Default gateway: 0 . 0 . 0 . 0

Fig.4-8: PROFINET I/O controller object

To a PROFINET I/O controller, the following can be assigned:

- I/O modules, page 79,

  and/or
- Modules for coupling a PROFINET I/O Device, page 65.

.

When the module for coupling a PROFINET I/O device is added in the Project Explorer, a firmware component in the controller is activated for connection to a PROFINET I/O device via PROFINET.

Tabs (PROFINET I/O controller)

- PNIO master parameters, page 60
- Field bus diagnostics, page 62,
- PROFINET I/O controller configuration, page 63, this window is used for service purposes and is only visible if the option "Show generic configuration editors" is enabled under **Tools ▸ Options ▸ IndraLogic 2G ▸ Device editor**.
- PROFINET I/O controller I/O mapping, page 64
- Status, page 65
- Information, page 65.

Tab "PNIO Master Parameters"

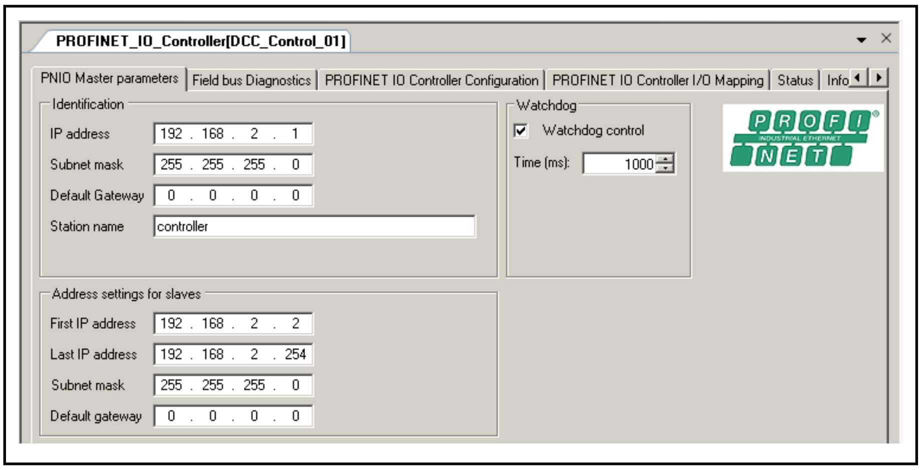
A "PROFINET I/O controller" is identified by an IP address and a subnet mask (the gateway address is optional). In addition, a station name must be specified and a watchdog control can be defined.

☞ The station name has to consist of the following characters:

Lower case letters'a' to 'z', digits '0' to '9' and the special character, hyphen '-'.

Upper case letters 'A' to 'Z' are permissible, but are internally converted into lower case letters.

The station names are not case sensitive.


PROFINET_IO_Controller[DCC_Control_01]
PNIO Master parameters
Field bus Diagnostics
PROFINET IO Controller Configuration
PROFINET IO Controller I/O Mapping
Status
Info
PROFI INDUSTRIAL ETHERNET NET

Identification
IP address: 192 . 168 . 2 . 1
Subnet mask: 255 . 255 . 255 . 0
Default Gateway: 0 . 0 . 0 . 0
Station name: controller

Watchdog
☑ Watchdog control
Time (ms): 1000

Address settings for slaves
First IP address: 192 . 168 . 2 . 2
Last IP address: 192 . 168 . 2 . 254
Subnet mask: 255 . 255 . 255 . 0
Default gateway: 0 . 0 . 0 . 0

Fig.4-9: PROFINET I/O controller: PNIO master parameters

Identification

The following addresses must be configured in accordance with the current environment to identify the controller.

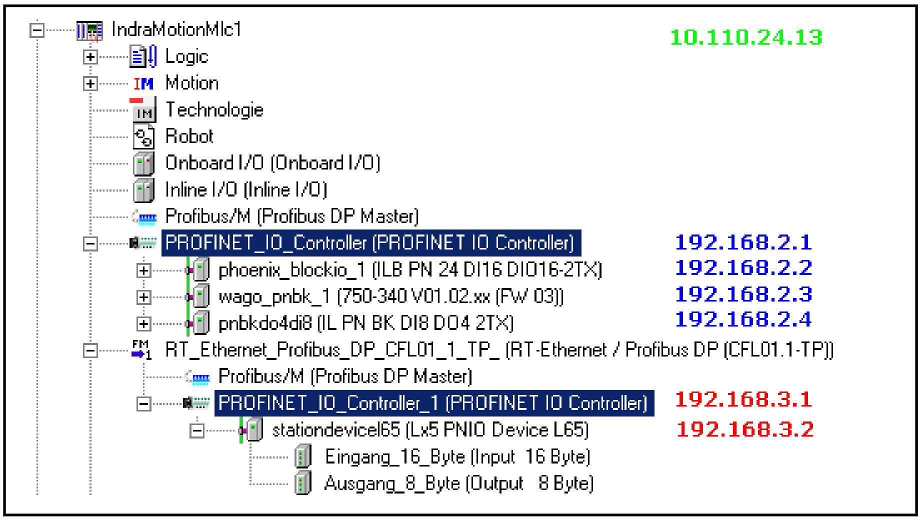
- IP address: default setting "192.168.2.1"

☞ In this section, to prevent doubled assignments when adjusting the IP address, also note the current address settings for devices (see Address settings for devices, page 61) that are added to the configuration in a scan procedure!

- Subnet mask: Default "255.255.255.0"
- Default gateway: Default gateway "0.0.0.0"
- Station name: Default "controller"

☞ Each device needs a unique station name in the subnet because it is required for certain functionalities in network mode!

Example: IP addresses in accordance with scanning



```
IndraMotionMlc1                                                        10.110.24.13
    Logic
    Motion
    Technologie
    Robot
    Onboard I/O (Onboard I/O)
    Inline I/O (Inline I/O)
    Profibus/M (Profibus DP Master)
    PROFINET_IO_Controller (PROFINET IO Controller)                    192.168.2.1
        phoenix_blockio_1 (ILB PN 24 DI16 DIO16-2TX)                   192.168.2.2
        wago_pnbk_1 (750-340 V01.02.xx (FW 03))                        192.168.2.3
        pnbkdo4di8 (IL PN BK DI8 DO4 2TX)                              192.168.2.4
    RT_Ethernet_Profibus_DP_CFL01_1_TP_ (RT-Ethernet / Profibus DP (CFL01.1-TP))
        Profibus/M (Profibus DP Master)
        PROFINET_IO_Controller_1 (PROFINET IO Controller)              192.168.3.1
            stationdevicel65 (Lx5 PNIO Device L65)                     192.168.3.2
                Eingang_16_Byte (Input  16 Byte)
                Ausgang_8_Byte (Output  8 Byte)
```

Fig.4-10: IP addresses in accordance with scanning the controller

The control is located in three networks that are independent of each other:

- Green (top): Enterprise network (Engineering interface),
- Blue (middle): PNIO, with its own I/O blocks,
- Red (bottom): PNIO, to the devices

Although the networks of the two controllers are independent of each other, a different subnet, e.g., 192.168.**3**.1.. should be used on the second controller to avoid possible future problems.

Watchdog

- When the "Watchdog" option is enabled, the monitoring time entered applies. That means, when this time has run out and the master was not active, a device-specific reaction results, e.g. an error message.
- Time (ms): Monitoring time, relevant in the case that the "Watchdog control" option is active. Possible values: 0 to 65535.

The defaults for the watchdog control are taken from the device description.

Address settings for devices

The address settings for devices are only used if devices found when scanning the hardware (**Project ▸ Search for Devices..**) are found, mapped in the Project Explorer and no address definitions have been selected in the scan dialog.

The values defined in this dialog are automatically transferred to the devices added (see PNIO identification, page 74) and for the IP address the next respective free address in the defined range is set. The defaults are taken from the device description file. They can be edited by setting the cursor in the respective input field:

☞ To prevent doubled assignments, when adjusting these addresses note the current IP address for the controller!

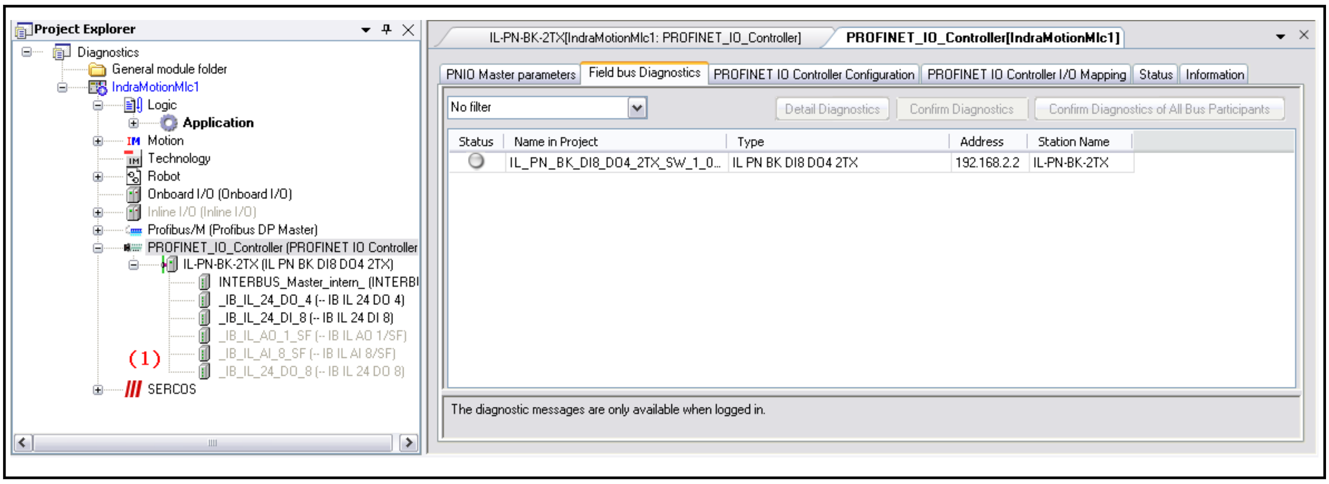
- First IP address: default setting "192.168.2.2"
- Last IP address: default setting "192.168.2.254"
- Subnet mask: Default "255.255.255.0"
- Default gateway: Default gateway "0.0.0.0"

Tab "Field Bus Diagnostics"

Based on the object node of the Master (here: PROFINET I/O controller), a uniform diagnostics concept has been developed for the field buses of the IndraLogic XLC and IndraMotion MLC / MTX systems.

The tab is only visible if the option "Show generic configuration editors" is enabled under **Tools** ▸ **Options** ▸ **IndraLogic 2G** ▸ **Device editor.**

The "Field bus Diagnostics" tab shows the modules applied to the field bus node in online mode.


Project Explorer
Diagnostics
General module folder
IndraMotionMlc1
Logic
Application
Motion
Technology
Robot
Onboard I/O (Onboard I/O)
Inline I/O (Inline I/O)
Profibus/M (Profibus DP Master)
PROFINET_IO_Controller (PROFINET IO Controller
IL-PN-BK-2TX (IL PN BK DI8 DO4 2TX)
INTERBUS_Master_intern_ (INTERBI
_IB_IL_24_DO_4 (-- IB IL 24 DO 4)
_IB_IL_24_DI_8 (-- IB IL 24 DI 8)
_IB_IL_AO_1_SF (-- IB IL AO 1/SF)
_IB_IL_AI_8_SF (-- IB IL AI 8/SF)
_IB_IL_24_DO_8 (-- IB IL 24 DO 8)
(1)
SERCOS
IL-PN-BK-2TX[IndraMotionMlc1: PROFINET_IO_Controller]
PROFINET_IO_Controller[IndraMotionMlc1]
PNIO Master parameters
Field bus Diagnostics
PROFINET IO Controller Configuration
PROFINET IO Controller I/O Mapping
Status
Information
No filter
Detail Diagnostics
Confirm Diagnostics
Confirm Diagnostics of All Bus Participants

Status	Name in Project	Type	Address	Station Name
○	IL_PN_BK_DI8_DO4_2TX_SW_1_0...	IL PN BK DI8 DO4 2TX	192.168.2.2	IL-PN-BK-2TX

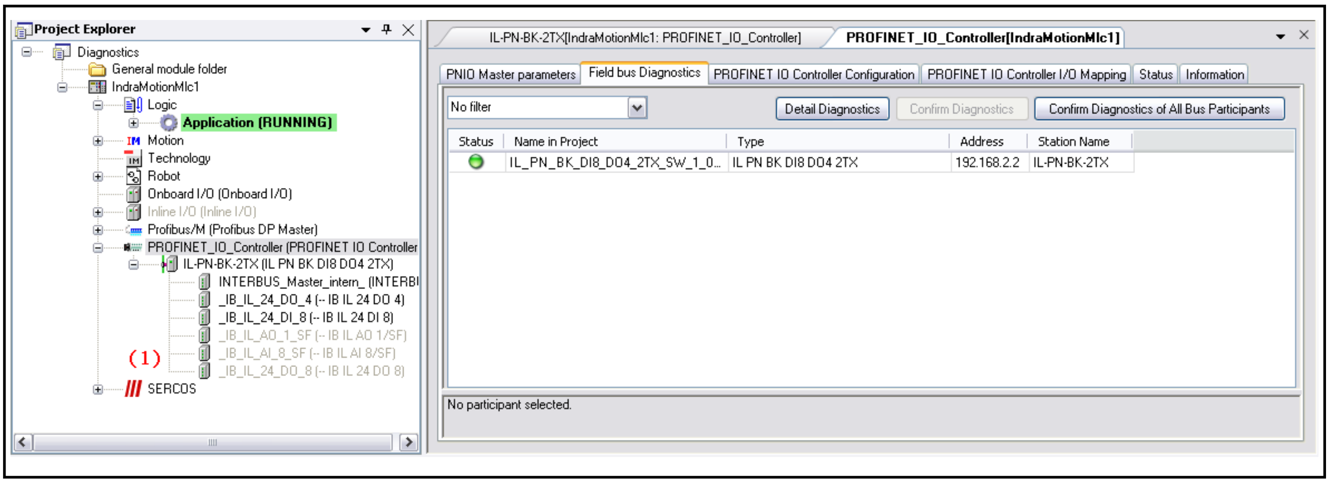
The diagnostic messages are only available when logged in.

Fig.4-11: Field bus diagnostics, online control, not logged in

The terminals (1) are disabled and excluded from the diagnostics by selecting **Properties** ▸ **Compile** ▸ **Exclude from compilation.**

After login, the bus available at the node runs through a diagnostics cycle. The status LEDs change their color in the result of this cycle (here: green), i.e., the module signals an error-free run.

After having been selected, a module (click on the line) is available for detailed diagnostics.


Project Explorer
Diagnostics
General module folder
IndraMotionMlc1
Logic
Application (RUNNING)
Motion
Technology
Robot
Onboard I/O (Onboard I/O)
Inline I/O (Inline I/O)
Profibus/M (Profibus DP Master)
PROFINET_IO_Controller (PROFINET IO Controller
IL-PN-BK-2TX (IL PN BK DI8 DO4 2TX)
INTERBUS_Master_intern_ (INTERBI
_IB_IL_24_DO_4 (-- IB IL 24 DO 4)
_IB_IL_24_DI_8 (-- IB IL 24 DI 8)
_IB_IL_AO_1_SF (-- IB IL AO 1/SF)
_IB_IL_AI_8_SF (-- IB IL AI 8/SF)
_IB_IL_24_DO_8 (-- IB IL 24 DO 8)
(1)
SERCOS
IL-PN-BK-2TX[IndraMotionMlc1: PROFINET_IO_Controller]
PROFINET_IO_Controller[IndraMotionMlc1]
PNIO Master parameters
Field bus Diagnostics
PROFINET IO Controller Configuration
PROFINET IO Controller I/O Mapping
Status
Information
No filter
Detail Diagnostics
Confirm Diagnostics
Confirm Diagnostics of All Bus Participants

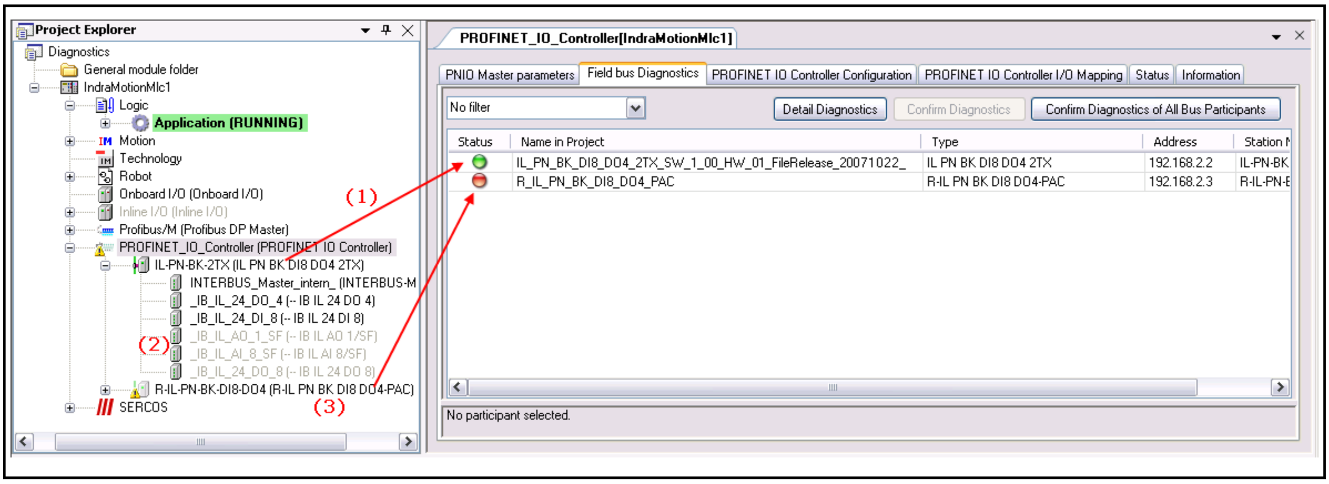
Status	Name in Project	Type	Address	Station Name
●	IL_PN_BK_DI8_DO4_2TX_SW_1_0...	IL PN BK DI8 DO4 2TX	192.168.2.2	IL-PN-BK-2TX

No participant selected.

Fig.4-12: Field bus diagnostics, online control, logged in, error-free

Login is repeated with an additional coupler not available at the real control.

A yellow warning triangle appears at the module in question, indicating that a diagnostic message is present. The status LED of the module in question turns red. Clicking on "Detail Diagnostics" provides "Standard diagnostics" with brief information and "Extended diagnostics" with detailed specification of the error cause.


Project Explorer
Diagnostics
General module folder
IndraMotionMlc1
Logic
Application [RUNNING]
Motion
Technology
Robot
Onboard I/O (Onboard I/O)
Inline I/O (Inline I/O)
Profibus/M (Profibus DP Master)
PROFINET_IO_Controller (PROFINET IO Controller)
IL-PN-BK-2TX (IL PN BK DI8 DO4 2TX)
INTERBUS_Master_intern_ (INTERBUS-M
_IB_IL_24_DO_4 (-- IB IL 24 DO 4)
_IB_IL_24_DI_8 (-- IB IL 24 DI 8)
_IB_IL_AO_1_SF (-- IB IL AO 1/SF)
_IB_IL_AI_8_SF (-- IB IL AI 8/SF)
_IB_IL_24_DO_8 (-- IB IL 24 DO 8)
R-IL-PN-BK-DI8-DO4 (R-IL PN BK DI8 DO4-PAC)
SERCOS
(1)
(2)
(3)
PROFINET_IO_Controller[IndraMotionMlc1]
PNIO Master parameters
Field bus Diagnostics
PROFINET IO Controller Configuration
PROFINET IO Controller I/O Mapping
Status
Information
No filter
Detail Diagnostics
Confirm Diagnostics
Confirm Diagnostics of All Bus Participants

Status	Name in Project	Type	Address	Station N
🟢	IL_PN_BK_DI8_DO4_2TX_SW_1_00_HW_01_FileRelease_20071022_	IL PN BK DI8 DO4 2TX	192.168.2.2	IL-PN-BK
🔴	R_IL_PN_BK_DI8_DO4_PAC	R-IL PN BK DI8 DO4-PAC	192.168.2.3	R-IL-PN-E

No participant selected.

Fig.4-13: Field bus diagnostics, online control, logged in, with error

(1) The bus coupler with existing terminals or terminals that are "excluded from compilation" (2) works without errors.

(3) The bus coupler is not existing and generates an error message.

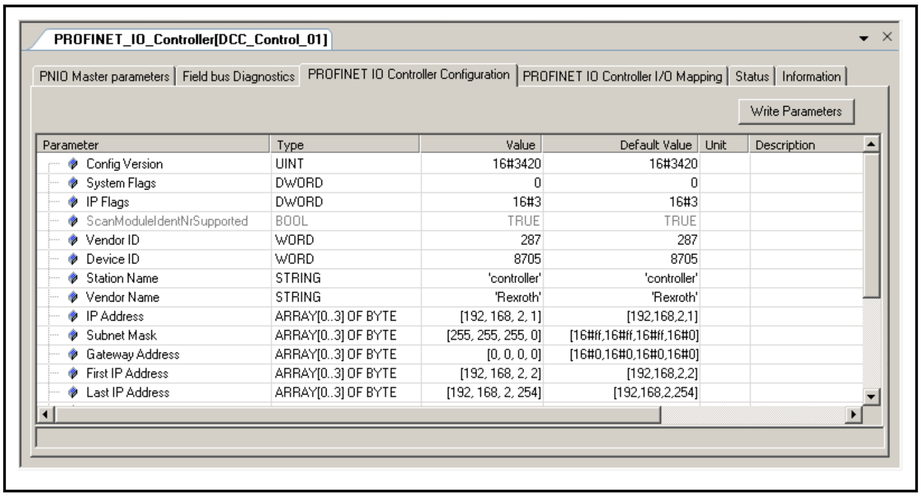
Confirm Diagnostics: Although there is an error, both the warning triangle and the red status LED are turned off for the selected module. The "Extended diagnostics" of the "Detail Diagnostics" remains in the text message. (The next diagnostic message can be processed...)

Confirm Diagnostics of All Bus Participants: Although there are errors, both the warning triangle and the red status LED are turned off for all modules.

Tab "PROFINET I/O Controller Configuration"

This window is used for service purposes and is only visible if in **Tools ▸ Options ▸ IndraLogic 2G ▸ Device Editor** the option "Display Generic Configuration Views" was enabled.

☞ Please clarify any possible modifications to the parameters that can be edited with the service team.


PROFINET_IO_Controller[DCC_Control_01]
PNIO Master parameters
Field bus Diagnostics
PROFINET IO Controller Configuration
PROFINET IO Controller I/O Mapping
Status
Information
Write Parameters

Parameter	Type	Value	Default Value	Unit	Description
Config Version	UINT	16#3420	16#3420		
System Flags	DWORD	0	0		
IP Flags	DWORD	16#3	16#3		
ScanModuleIdentNrSupported	BOOL	TRUE	TRUE		
Vendor ID	WORD	287	287		
Device ID	WORD	8705	8705		
Station Name	STRING	'controller'	'controller'		
Vendor Name	STRING	'Rexroth'	'Rexroth'		
IP Address	ARRAY[0..3] OF BYTE	[192, 168, 2, 1]	[192,168,2,1]		
Subnet Mask	ARRAY[0..3] OF BYTE	[255, 255, 255, 0]	[16#ff,16#ff,16#ff,16#0]		
Gateway Address	ARRAY[0..3] OF BYTE	[0, 0, 0, 0]	[16#0,16#0,16#0,16#0]		
First IP Address	ARRAY[0..3] OF BYTE	[192, 168, 2, 2]	[192,168,2,2]		
Last IP Address	ARRAY[0..3] OF BYTE	[192, 168, 2, 254]	[192,168,2,254]		

Fig.4-14: PROFINET I/O controller: PROFINET I/O controller configuration (online)

This window contains information about the PROFINET I/O controller parameter set.

Window structure

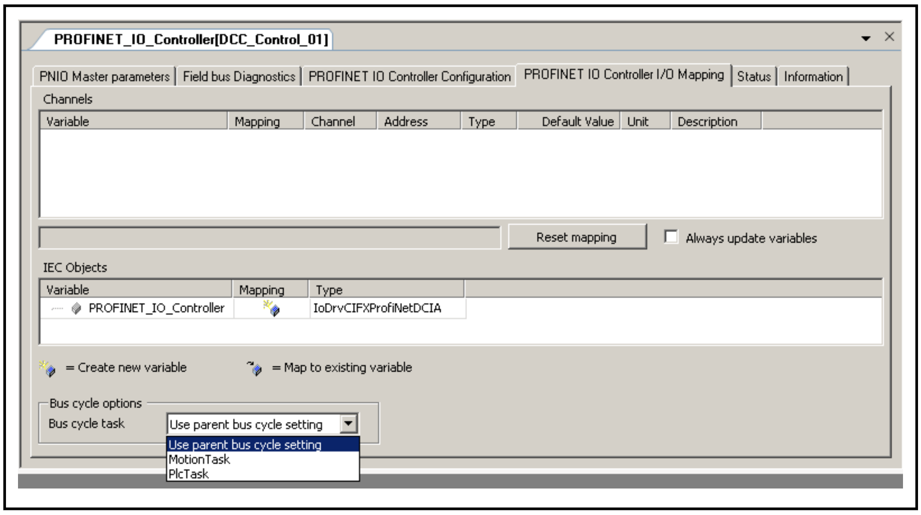
- **Parameters**: Parameter name from the device description file, cannot be edited.
- **Type**: Data type of the parameter, cannot be edited.
- **Value**: First, the standard value of the parameter is displayed, directly or as a specification of the corresponding symbolic name.

  If the parameter can be edited (this depends on the device description; parameters that cannot be edited are displayed in light gray), an input field or a selection list can be opened by double-clicking on the table field (or pressing the <space bar> in a previously selected field) where the value can be changed.

  Values are accepted with <Write parameter>.

  If the value is related to a file specification, the standard dialog for selected a file opens.
- **Default Value**: Defined value from the device description, cannot be edited.
- **Unit**: Unit for the value, e.g. "ms" for milliseconds, cannot be edited.
- **Description**: Short description of the parameter from the device description file, cannot be edited.

Tab "PROFINET I/O Controller I/O Mapping"


PROFINET_IO_Controller[DCC_Control_01]
PNIO Master parameters
Field bus Diagnostics
PROFINET IO Controller Configuration
PROFINET IO Controller I/O Mapping
Status
Information
Reset mapping
□ Always update variables
= Create new variable
= Map to existing variable
Use parent bus cycle setting
MotionTask
PlcTask

Channels

Variable	Mapping	Channel	Address	Type	Default Value	Unit	Description

IEC Objects

Variable	Mapping	Type
PROFINET_IO_Controller		IoDrvCIFXProfiNetDCIA

Bus cycle options

Bus cycle task: Use parent bus cycle setting

Fig.4-15: PROFINET I/O controller: PROFINET I/O controller I/O mapping

Channels: The upper section of the dialog is not used because the I/O mapping is done in the I/O blocks.

See PNIO modules I/O mapping, page 82

IEC objects: When the PROFINET I/O controller is defined, the libraries "IoDrvCIFXProfinet.library" and/or RIL_ProfinetIO.library automatically applied. In this way, the memory space required to implement the PROFINET I/O controller can be defined.

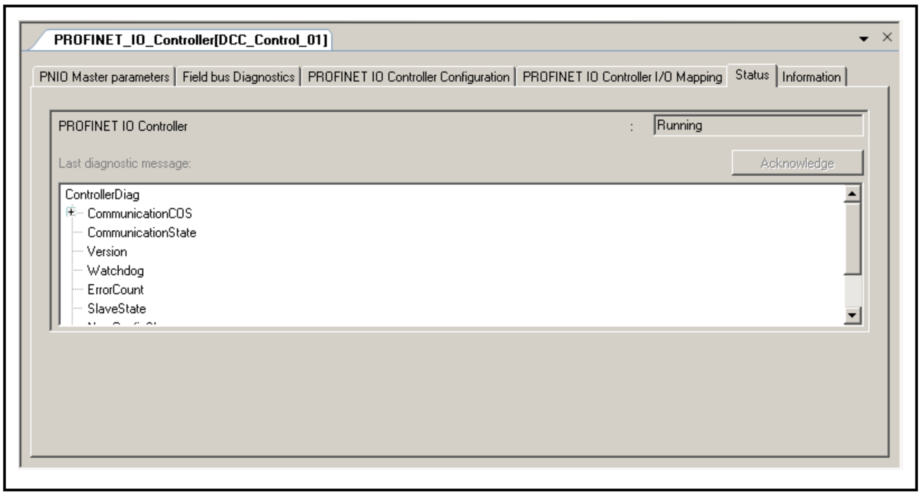
Bus cycle options

Bus cycle task: By selecting a bus cycle task, the cycle of the mapping exchange for the PROFINET I/O controller can be connected to a particular task. In this task, it is useful to process the I/O data of the master as well.

Default setting: "Use parent bus cycle setting"

With this setting, the task setting made in "Bus cycle options" for the control (double click on the actual control in the Project Explorer, PLC settings) is accepted for the actual bus.

Tab "Status"


PROFINET_IO_Controller[DCC_Control_01]
PNIO Master parameters
Field bus Diagnostics
PROFINET IO Controller Configuration
PROFINET IO Controller I/O Mapping
Status
Information
Acknowledge

PROFINET IO Controller : Running

Last diagnostic message:

ControllerDiag
CommunicationCOS
CommunicationState
Version
Watchdog
ErrorCount
SlaveState

Fig.4-16: PROFINET I/O controller: Status

The "Status" tab in the "PROFINET I/O Editor" displays status information (e.g., "Running", "Stopped") and specific diagnostic messages from the device. In addition, the diagnostic messages contained in the status flags are displayed in "Diag".

The output is a hexadecimal value determined by the set status flags (see the "Protocol Interface Manual" for PROFINET I/O).

See Diagnostics in the PROFINET I/O, page 82.

Tab "Information"

The window displays some general information from the device description file:

Name, Vendor, Categories, Version, Order number, Description, Image, if available.

4.3.3 PROFINET I/O Controller, Adding a Module for Coupling a PROFINET I/O Device

PROFINET I/O Controller, Adding a Module for Coupling a PROFINET I/O Device

To couple a control used as PROFINET I/O device, a device is available under **Library** ▸ **Periphery** ▸ **PROFINET I/O** ▸ **PLC** that allows coupling of the following controls:

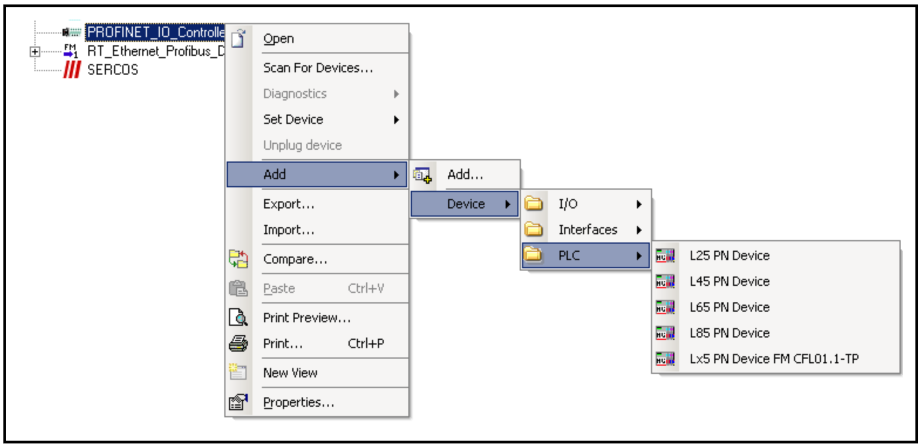


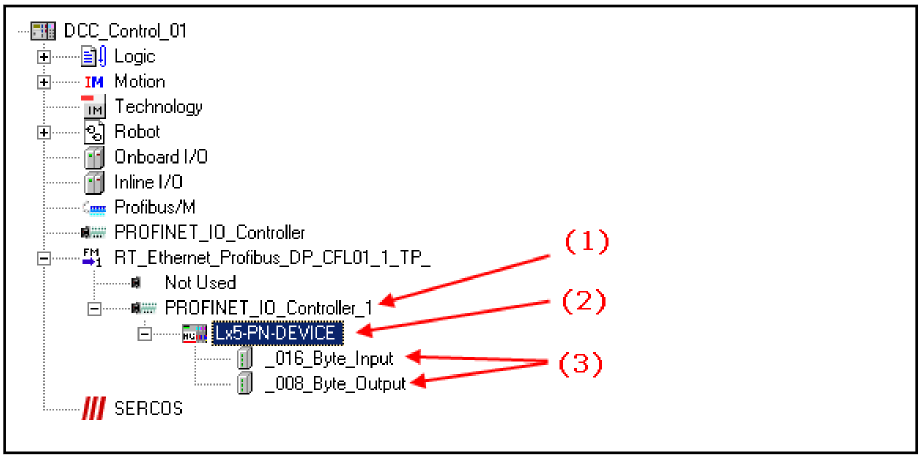
Fig.4-17: PROFINET I/O controller: selecting the module for coupling a PROFINET I/O device control

- **L25 PN device, ..., L85 PN device** for controls L25 ... L85, coupling via their onboard interfaces
- **Lx5 PN device FM CFL011.1-TP** for controls L25 ... L85, coupling via a function module "RT-Ethernet / Profibus DP (CFL01.1-TP)"

When the module for coupling a PROFINET I/O device control is added in the Project Navigator, a firmware component in the controller is activated for connection to a PROFINET I/O device via PROFINET.

Each of these modules has a selection of (sub)modules in the device library that release input and output variables in the mapping memory of the controller for data exchange between the PROFINET I/O controller and the PROFINET I/O device control.

(See also PROFINET I/O Modules, page 79.)



(1) PROFINET I/O controller
(2) Module for coupling
(3) Memory modules for data exchanged between the controls

Fig.4-18: PROFINET I/O controller with module for coupling a PROFINET I/O device via a function module

Tabs (module for coupling a PROFINET I/O device)

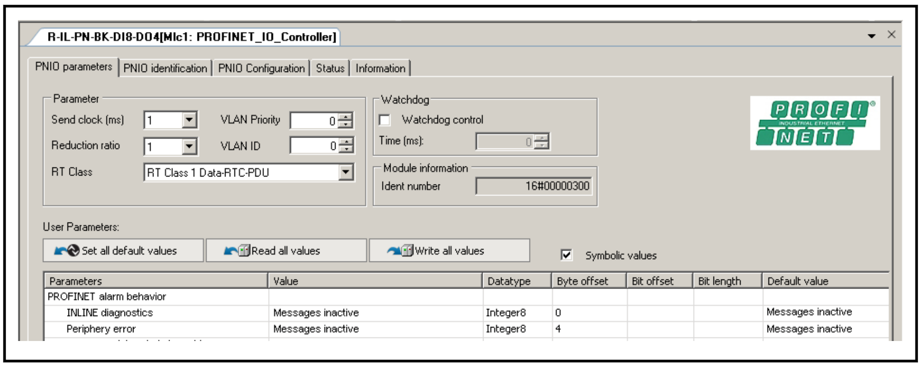
- PNIO Parameters, page 67
- PNIO identification, page 68
- PNIO configuration, page 70, this window is used for service purposes and is only visible if the option "Show generic configuration editors" is enabled under **Tools** ▸ **Options** ▸ **IndraLogic 2G** ▸ **Device editor**.

- Status, page 70
- Information, page 70.

Tab "PNIO Parameters"

To open the editor window in the Project Explorer, double-click on the PROFINET I/O module for coupling a PROFINET I/O device.

All settings in the dialog are dependent on the device description in terms of whether the settings can be edited here and which values are specified or possible.



Parameter
Send clock (ms): 1
Reduction ratio: 1
RT Class: RT Class 1 Data-RTC-PDU
VLAN Priority: 0
VLAN ID: 0

Watchdog
☐ Watchdog control
Time (ms): 0

Module information
Ident number: 16#00000300

User Parameters:

☑ Symbolic values

Parameters	Value	Datatype	Byte offset	Bit offset	Bit length	Default value
PROFINET alarm behavior						
INLINE diagnostics	Messages inactive	Integer8	0			Messages inactive
Periphery error	Messages inactive	Integer8	4			Messages inactive

Fig.4-19: Register: PNIO parameters

Parameter

These parameters describe the time response of the communication on the PROFINET.

The product of the two parameters

`Reduction Ratio` and

`Send clock`, the actual cycle time is calculated

`(t = Reduction Ratio * Send clock)`

based upon which the device transmits data:

- **Send clock (ms):** Sending time in milliseconds.
- **Reduction ratio:** Factor for calculating the cycle time from the sending time.
- **RT class:** Select the desired class (Real-time communication, page 73).

  Currently, RT_Class_1 is supported.
- **VLAN priority:** Priority of the device in the "Virtual Local Area Network" (0 to 7), if available.
- **VLAN ID:** Enter a value between 0 and 4095 for the VLAN type 802.1Q.

  For VLAN type ISL enter a value between 0 and 32767, if available.

Watchdog

- **Monitoring:** Monitoring control. If this option is active, the monitoring time entered applies (watchdog). If the slave does not obtain any signal from the master within this time period anymore, a device-specific response occurs, e.g. error message.
- **Time (ms):** Monitoring time, relevant in the case that the "Watchdog control" option is enabled. Possible values: 0 to 65535.

Module information

Ident number: This entry is contained in the device description file and helps to identify the device.

User parameters

- **Set all default values:** This command resets all selected settings to the default settings (see "Default value" column) of the GSDML file.

- **Read all values:** This command reads the current values from the device and updates the values in the editor.
- **Write all values:** This command writes the current editor values to the slave. Not all slaves support a parameter update during run mode. In this case, an error message is displayed.
- **Symbolic values:** a symbolic value specification can be activated. When it is deactivated, the numeric values are displayed.

Parameter table

In the parameter table, double-clicking on the respective value allows you to edit this value. Depending on the parameter this occurs with a selection list or by directly entering a value.

- **Parameters:** Name of the parameter of the parameter category (without value assignment)
- **Value:** Current parameter value
- **Data type:** Data type of parameter, e.g. "Bit"
- **Byte offset:** The parameters defined by the user are saved in the Variable Record Data (array of bytes) variables. The byte offset specifies the first valid byte.
- **Bit offset:** The parameters defined by the user are saved in the Variable Record Data (array of bytes) variables. The bit offset specifies the first valid bit, the byte identified by the byte offset.
- **Bit length:** Length of the information saved in the Variable Record Data (array of bytes) variables which contain the parameters defined by the user.
- **Default value:** Default value of the parameter.
- **Value range:** Specification as to what may be entered in the 'Value' column.

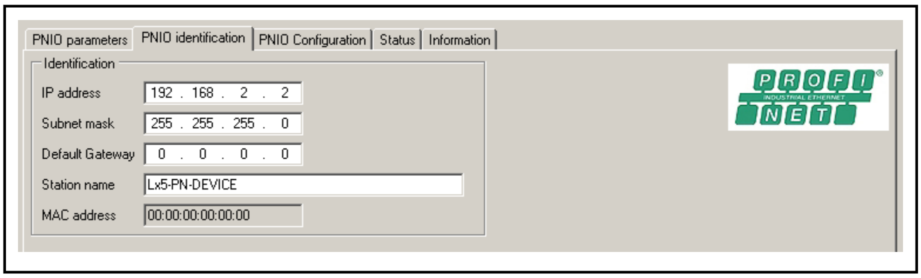
  Syntax:

  `<Parameter_Type>(<used bit>)<Basic value><Value range>`.

  Example:

  "BitArea (4-5) 0 0-2" means: This is a bit combination stored in bits 4 and 5 of the configuration byte. The base value is 0; the value can lie between 0 and 2.
- *Interfaces "User parameters" (only active online)*
  - **Set all default values**: Clicking this button replaces all the changed settings with the default settings from the GSDML file.
  - **Read all values**: Reading the current values of the device.
  - **Write all values**: Writing the current values from the following devices.

Tab "PNIO Identification"



Identification

IP address 192 . 168 . 2 . 2

Subnet mask 255 . 255 . 255 . 0

Default Gateway 0 . 0 . 0 . 0

Station name Lx5-PN-DEVICE

MAC address 00:00:00:00:00:00

Fig.4-20: Register: PNIO identification

The fields for "IP address", "Subnet mask", "Default gateway", "Station name" and if required "MAC address" contain default entries derived from the controller setting.

☞ The station name is used for selecting the PROFINET I/O device. Missing or deviating settings on the side of the device are ignored.

☞ The station name may consist of the following characters: Lower case letters'a' to 'z', digits '0' to '9' and the special character, hyphen '-'. Upper case letters'A' to 'Z' are permissible, but are internally converted into lower case letters. The station names are not case sensitive.

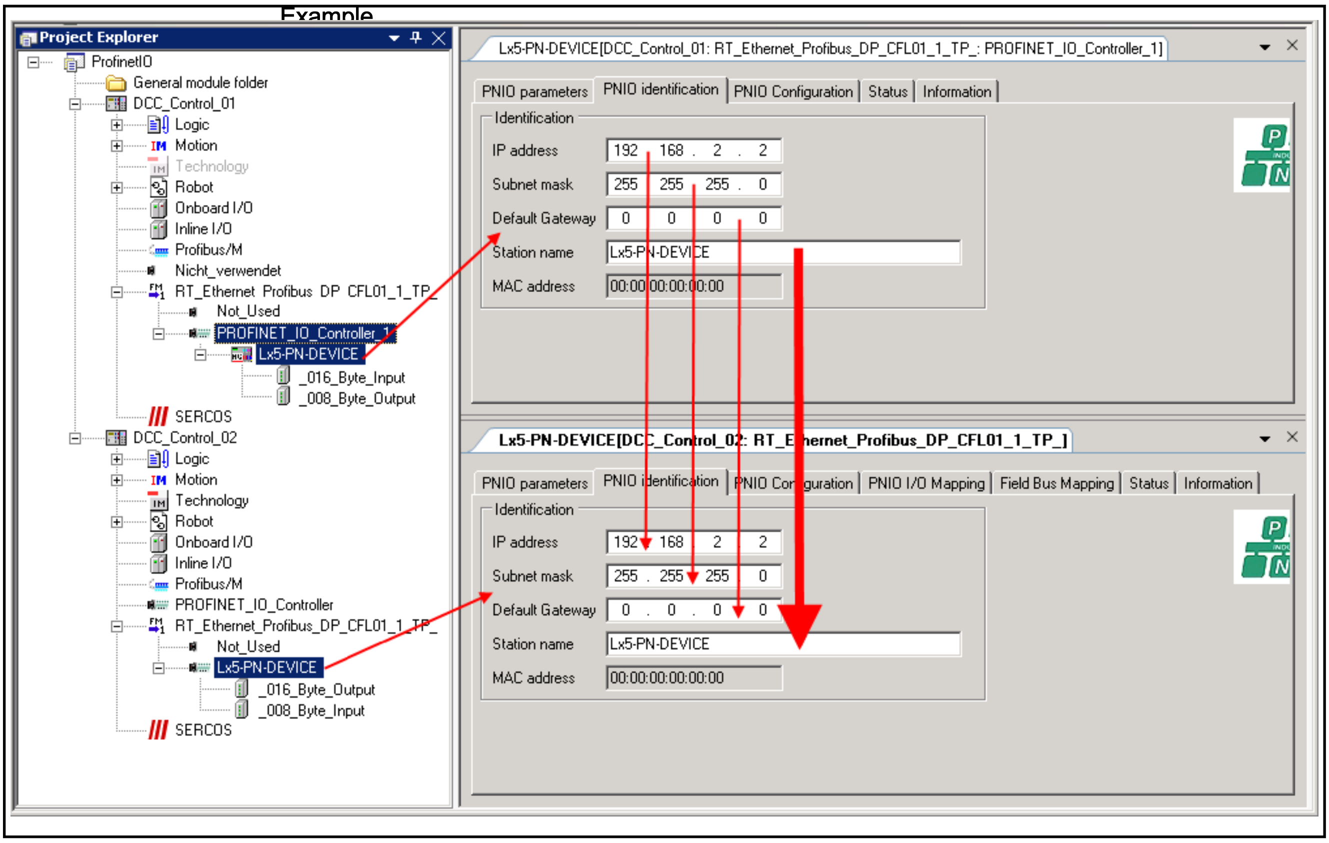
The device cannot be scanned. It has to be identified by a device name that is unique throughout the subnet.

Name and address settings of the associated module for coupling a PROFINET I/O device has to be applied, as shown in the following figure.

In the "MAC address" field, the MAC address is displayed following a network scan. This field cannot be edited.

If the PROFINET I/O device has not been inserted via the "Scan functionality", the data of the associated module for coupling a PROFINET I/O device has to be transferred as shown in the following figure; the name is the decisive factor.

Example



Lx5-PN-DEVICE[DCC_Control_01: RT_Ethernet_Profibus_DP_CFL01_1_TP_: PROFINET_IO_Controller_1]

Identification

IP address	192 . 168 . 2 . 2
Subnet mask	255 . 255 . 255 . 0
Default Gateway	0 . 0 . 0 . 0
Station name	Lx5-PN-DEVICE
MAC address	00:00:00:00:00:00

Lx5-PN-DEVICE[DCC_Control_02: RT_Ethernet_Profibus_DP_CFL01_1_TP_]

Identification

IP address	192 . 168 . 2 . 2
Subnet mask	255 . 255 . 255 . 0
Default Gateway	0 . 0 . 0 . 0
Station name	Lx5-PN-DEVICE
MAC address	00:00:00:00:00:00

Fig.4-21: Transferring the name for identifying from the controller to the device

☞ The data exchange between the controller and the device is performed using common memory ranges, where the outputs of one are inputs of the other and vice versa.

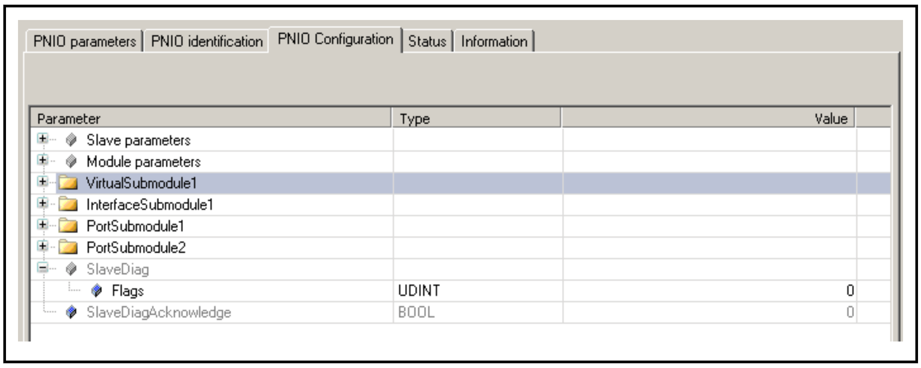
A check must be carried out and, if necessary, the device must be manually adjusted to the module of the controller.

See also "Scan functionality", page 84.

Tab "PNIO Configuration"

This window is used for service purposes and is only visible if in **Tools ▸ Options ▸ IndraLogic 2G ▸ Device Editor** the option "Display Generic Configuration Views" was enabled.

☞ Please clarify any possible modifications to the parameters that can be edited with the service team.



Parameter	Type	Value
Slave parameters		
Module parameters		
VirtualSubmodule1		
InterfaceSubmodule1		
PortSubmodule1		
PortSubmodule2		
SlaveDiag		
Flags	UDINT	0
SlaveDiagAcknowledge	BOOL	0

Fig.4-22: Register: PNIO configuration

☞ When the bus is running, modified parameters can be transferred by using "Write parameter".

Tab "Status"

The "Status" tab in the "PROFINET I/O device editor" displays status information (e.g. "Running", "Stopped") and specific diagnostic messages from the device. In addition, the diagnostic messages contained in the status flags are displayed in "Diag".

The output is a hexadecimal value determined by the set status flags (see the "Protocol Interface Manual" for the PROFINET I/O).

See Diagnostics in the PROFINET I/O, page 82.

Tab "Information"

The window displays some general information from the device description file:

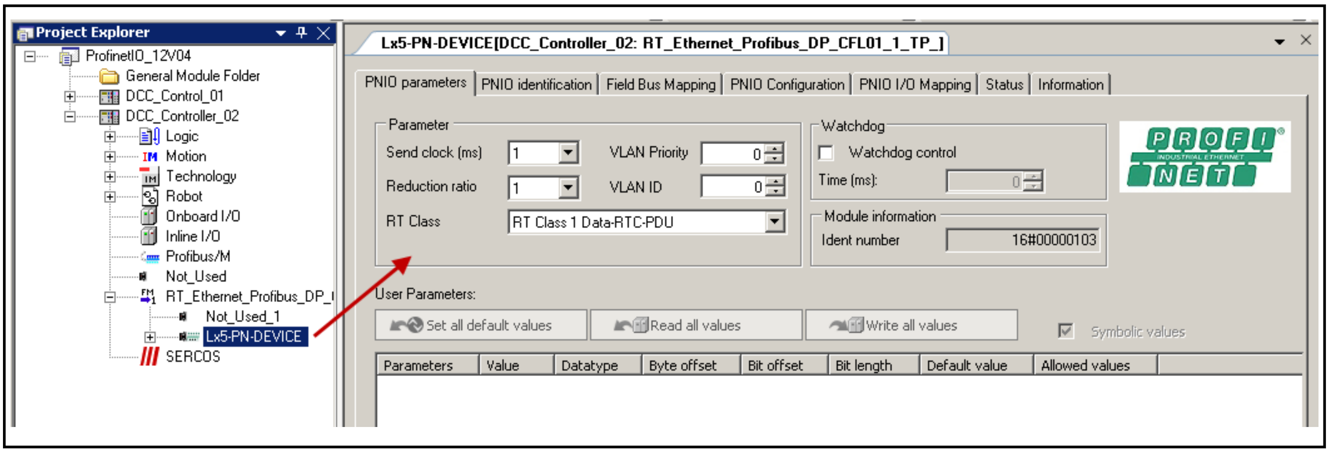
Name, Vendor, Categories, Version, Order number, Description, Image, if available.

4.3.4 PROFINET I/O Device

Configuring a PROFINET I/O Device Object, General Information

To open the editor window in the Project Explorer, double click on the PROFINET I/O device object.

The dialogs will inform you about the configuration of the device side of the PROFINET I/O bus and you can modify it if necessary.


Project Explorer
ProfinetIO_12V04
General Module Folder
DCC_Control_01
DCC_Controller_02
Logic
Motion
Technology
Robot
Onboard I/O
Inline I/O
Profibus/M
Not_Used
RT_Ethernet_Profibus_DP_I
Not_Used_1
Lx5-PN-DEVICE
SERCOS
Lx5-PN-DEVICE[DCC_Controller_02: RT_Ethernet_Profibus_DP_CFL01_1_TP_]
PNIO parameters | PNIO identification | Field Bus Mapping | PNIO Configuration | PNIO I/O Mapping | Status | Information
Set all default values
Read all values
Write all values
PROFI NET INDUSTRIAL ETHERNET

Parameter: Send clock (ms) 1; Reduction ratio 1; RT Class RT Class 1 Data-RTC-PDU; VLAN Priority 0; VLAN ID 0

Watchdog: □ Watchdog control; Time (ms): 0

Module information: Ident number 16#00000103

User Parameters: ☑ Symbolic values

Parameters	Value	Datatype	Byte offset	Bit offset	Bit length	Default value	Allowed values

Fig.4-23: PROFINET I/O device object

To a PROFINET I/O device, the following can be assigned:

- Memory modules, page 79,

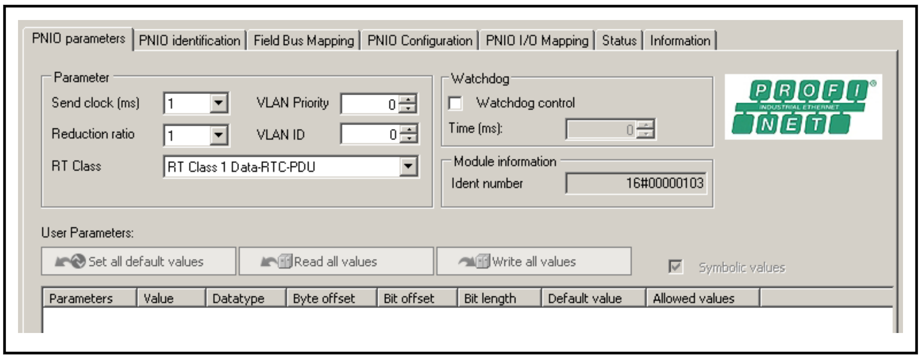
.

Tabs (PROFINET I/O device)

- PNIO parameters, page 71
- PNIO identification, page 74
- PNIO configuration, page 75, this window is used for service purposes and is only visible if the option "Show generic configuration editors" is enabled under **Tools ▸ Options ▸ IndraLogic 2G ▸ Device editor.**
- PNIO I/O mapping, page 76
- Field bus mapping, page 76
- Status, page 79
- Information, page 79

Tab "PNIO Parameters"

All settings in the dialog are dependent on the device description in terms of whether the settings can be edited here and which values are specified or possible.


PNIO parameters | PNIO identification | Field Bus Mapping | PNIO Configuration | PNIO I/O Mapping | Status | Information
Set all default values
Read all values
Write all values
PROFI NET INDUSTRIAL ETHERNET

Parameter: Send clock (ms) 1; Reduction ratio 1; RT Class RT Class 1 Data-RTC-PDU; VLAN Priority 0; VLAN ID 0

Watchdog: □ Watchdog control; Time (ms): 0

Module information: Ident number 16#00000103

User Parameters: ☑ Symbolic values

Parameters	Value	Datatype	Byte offset	Bit offset	Bit length	Default value	Allowed values

Fig.4-24: PROFINET I/O device: PNIO parameters

Parameter

These parameters describe the time response of the communication on the PROFINET. They are set by the PROFINET I/O controller while the connection is being established.

From the two parameters

`Reduction Ratio` and

`Send clock`, the actual cycle time is calculated

(`t = Reduction Ratio * Send clock`)

based upon which the device transmits data:

- Send clock (ms), sending time in milliseconds.
- Reduction ratio, factor for calculating the cycle time from the sending time.
- RT class:

  Select the desired class (real-time communication, page 73).

  Currently, RT_Class_1 is supported.
- VLAN priority, priority of the device in the "Virtual Local Area Network" (0 to 7, if available).
- VLAN ID, enter a value between 0 and 4095 for the VLAN type 802.1Q.

  For VLAN type ISL enter a value between 0 and 32767, if available.

Watchdog

If the "Watchdog" option is enabled, the monitoring time entered applies. If the PROFINET I/O device does not receive any more signals from the controller within this time period, a device-specific response occurs, e.g., error message.

- Time (ms), monitoring time, relevant in the case that the "Watchdog control" option is active. Possible values: 0 to 65535.

Module information

The ID number returns the identifier of the PROFINET I/O device type.

User parameters

Based on the device, there are either no setting possibilities or there are several setting possibilities for the user-specific parameters:

- **Symbolic values**:

  You can use the "Symbolic values" option to enable symbolic value display. When it is deactivated, the numeric values are displayed.
- **Length of the user parameters**:

  The total length of all user parameters is displayed in bytes.

In the parameter table you can edit the values by double-clicking on them. Depending on the parameter this occurs with a selection list or by directly entering a value.

- **Parameters**: Name of the parameter of the parameter category (without value assignment).
- **Value**: Current parameter value
- **Type of data**: Type of parameter data, e.g. "bit"
- **Byte offset**: The parameters defined by the user are saved in the "Record Data" (array of bytes) variables. The byte offset specifies the first valid byte.
- **Bit offset**: The parameters defined by the user are saved in the "Record Data" (array of bytes) variables. The bit offset specifies the first valid bit, the byte identified by the byte offset.
- **Bit length**: Length of the information saved in the "Record Data" (array of bytes) variables which contain the parameters defined by the user.
- **Default Value**: Default value of the parameter.
- **Permissible values**: Specification as to what may be entered in the "Value" column.

Syntax:

`<parameter type> (<used bits>) <base value> <permissible value range>`.

Example:

"BitArea (4-5) 0 0-2" means: This is a bit combination stored in bits 4 and 5 of the configuration byte. The base value is 0; the value can lie between 0 and 2.

- *Interfaces "User parameters" (only active online)*
  - **Set all default values**: Clicking this button replaces all the changed settings with the default settings from the GSDML file.
  - **Read all values**: Reading the current values of the device.
  - **Write all values**: Writing the current values from the following devices.

Real-time classes, PROFINET I/O

In order to provide better scaling for the communication options and therefore the determinism in PROFINET I/O, real-time classes were defined for data exchange. From the user perspective, this means both unsynchronized and synchronized communication. The details are taken care of automatically in the field devices. Real-time in PROFINET means that the priority of UDP/IP frames is reduced. This is necessary in order to prioritize the data flow in the switches so that RT frames are not delayed by UDP/IP frames.

PROFINET I/O differentiates the following classes in RT communication, although the difference is not in performance, but instead, in determinism.

RT_CLASS_1: Unsynchronized RT communication within a subnet.

No special addressing information is necessary for this communication. The target device can only be identified using "Dest. Addr".

In PROFINET I/O, unsynchronized communication within a subnet is the usual type of data transfer.

If the RT data traffic on a subnet (same network ID) can be limited, this version is the simplest. This communication path is standardized in parallel with UDP/IP communication and is implemented in every PROFINET I/O field device. The management information from UDP/IP and RPC is purposely not provided here. RT frames that are received are identified upon reception using the Ether type (16#8892) and are then forwarded to the RT path for processing. Industrial-strength standard switches can be used in this RT class.

RT_CLASS_2: RT_CLASS_2 frames can be transferred synchronously or asynchronously.

The asynchronous communication is considered to be the same here as RT_CLASS_1 communication.

In synchronized communication, the start of a bus cycle is defined for all devices. This determines exactly the time period in which field devices are allowed to transmit. For all of the field devices in RT_CLASS_2 that are involved in the communication, this is always at the start of the bus cycle. Switches suitable for PROFINET must support this synchronization during this communication. For this data transfer, which is designed for performance, special provisions for hardware must be made (EtherNet controller/switch with support for isochronicity).

RT_CLASS_3: Synchronized RT communication within a subnet.

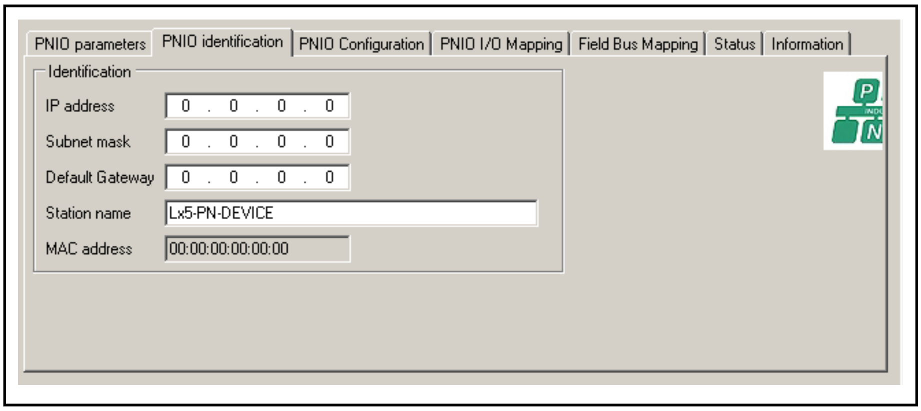
In synchronized RT_CLASS _3 communication, the process data is transmitted with high precision according to an exact sequence determined when the equipment is engineered (maximum allowed deviation from the start of a bus cycle is 1 µs). This data transfer functionality, optimized with the topology, is also known as IRT functionality (isochronous real time). In RT_CLASS_3 communication there is no waiting time. To take advantage of this data transfer procedure, designed for high performance, special provisions for hardware must be made (EtherNet controller/switch with support for isochronicity).

RT_CLASS_UDP: The unsynchronized communication across and among a variety of subnets requires addressing information from the target network (IP address).

This version is also known as RT_CLASS_UDP. Standard switches can be used in this RT class. For RT frames, achieving data cycles of 5 ms at 100 Mb/sec. in full duplex operation with VLAN tag is sufficient. This RT communication can be realized with all available standard network components.

(Citation: PROFINET Technology and Application, version April 2009, PROFIBUS User Organization e.V. (incorporated society), PROFIBUS & PROFINET International Support Center)

Tab "PNIO Identification"



Identification

IP address 0 . 0 . 0 . 0

Subnet mask 0 . 0 . 0 . 0

Default Gateway 0 . 0 . 0 . 0

Station name Lx5-PN-DEVICE

MAC address 00:00:00:00:00:00

Fig.4-25: PROFINET I/O devices: PNIO identification

The fields for "IP address", "Subnet mask", "Default gateway", "Station name" and if required "MAC address" contain default entries or are empty.

The device cannot be scanned. It has to be identified by a device name that is unique throughout the subnet.

The entries into these fields must be synchronized manually with the associated module for coupling a PROFINET I/O device of the PROFINET I/O controller, as shown in the figure below.

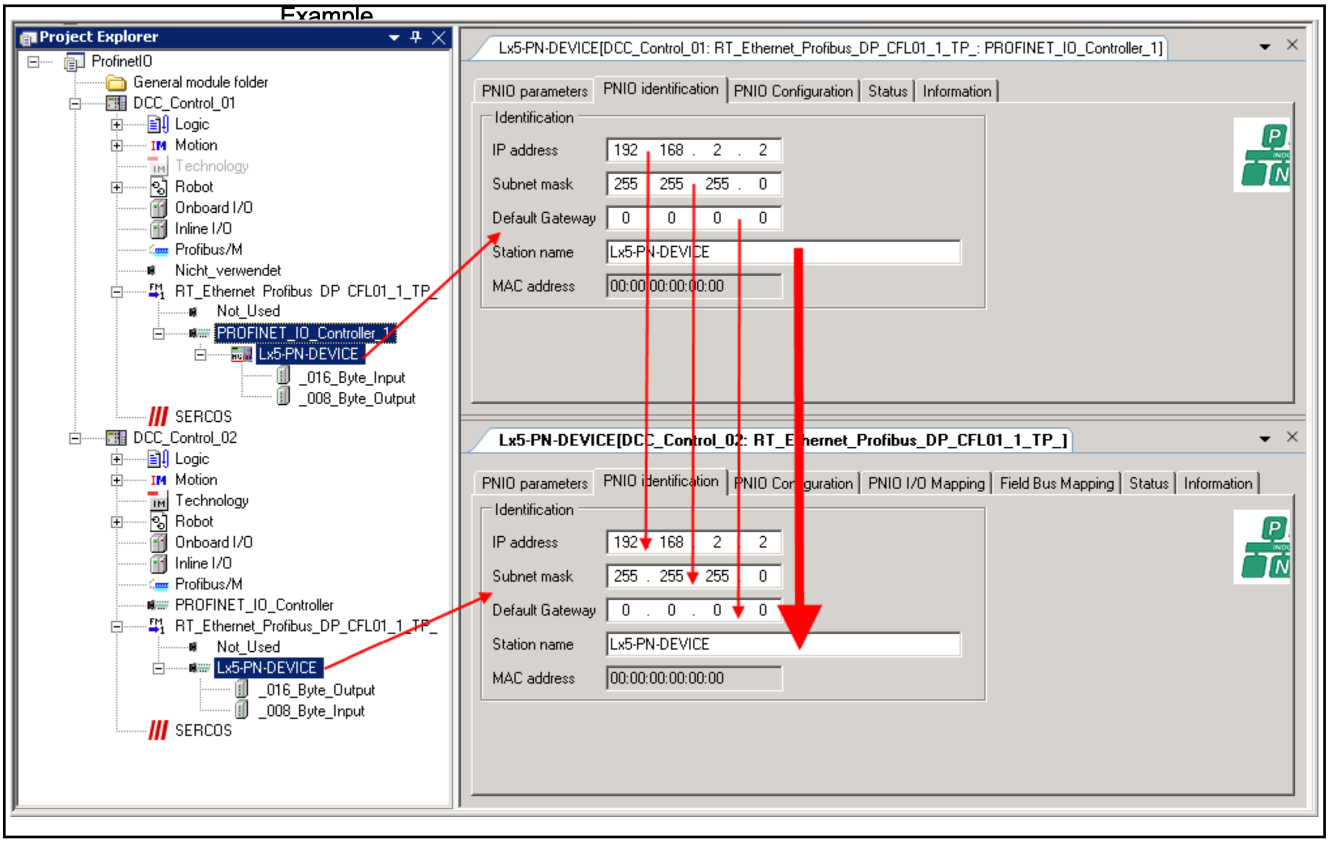
The station name is the decisive factor.

☞ The station name may consist of the following characters:

Lower case letters 'a’ to 'z’, digits '0’ to '9’ and the special character, hyphen '-’.

Upper case letters 'A' to 'Z' are permissible, but are internally converted into lower case letters. The station names are not case sensitive.

In the "MAC address" field, the MAC address is displayed following a network scan. This field cannot be edited.



IP address 192 . 168 . 2 . 2
Subnet mask 255 . 255 . 255 . 0
Default Gateway 0 . 0 . 0 . 0
Station name Lx5-PN-DEVICE
MAC address 00:00:00:00:00:00

IP address 192 . 168 . 2 . 2
Subnet mask 255 . 255 . 255 . 0
Default Gateway 0 . 0 . 0 . 0
Station name Lx5-PN-DEVICE
MAC address 00:00:00:00:00:00

Fig.4-26: Transferring the name for identifying from the controller to the device

☞ The data exchange between the controller and the device is performed using common memory ranges, where the outputs of one are inputs of the other and vice versa.

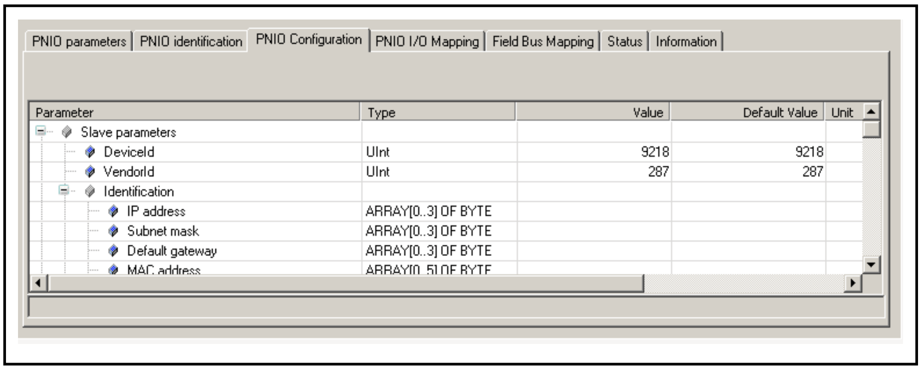
A check must be carried out and, if necessary, the device must be manually adjusted to the module for coupling a PROFINET I/O device of the controller.

See also "Scan functionality", page 84.

Tab "PNIO Configuration"

This window is used for service purposes and is only visible if in **Tools** ▸ **Options** ▸ **IndraLogic 2G** ▸ **Device Editor** the option "Display Generic Configuration Views" was enabled.

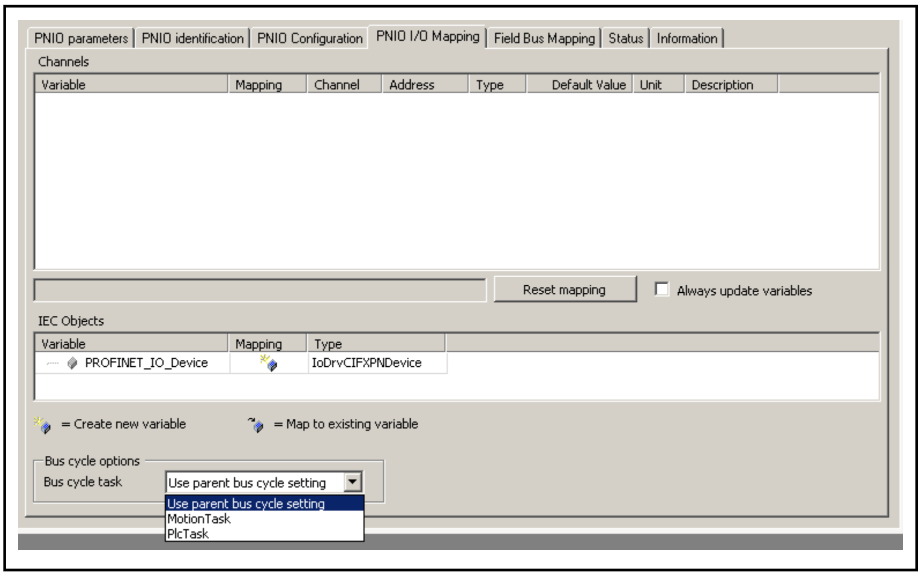
☞ Please clarify any possible modifications to the parameters that can be edited with the service team.



Parameter	Type	Value	Default Value	Unit
Slave parameters				
DeviceId	UInt	9218	9218	
VendorId	UInt	287	287	
Identification				
IP address	ARRAY[0..3] OF BYTE			
Subnet mask	ARRAY[0..3] OF BYTE			
Default gateway	ARRAY[0..3] OF BYTE			
MAC address	ARRAY[0..5] OF BYTE			

Fig.4-27: PROFINET I/O devices: PNIO configuration

Tab "PNIO I/O Mapping"



Channels

Variable	Mapping	Channel	Address	Type	Default Value	Unit	Description

□ Always update variables

IEC Objects

Variable	Mapping	Type
PROFINET_IO_Device		IoDrvCIFXPNDevice

= Create new variable
= Map to existing variable

Bus cycle options
Bus cycle task: Use parent bus cycle setting

Fig.4-28: PROFINET I/O devices: PNIO I/O mapping

Channels: The upper section of the dialog is not used because the I/O mapping is done in the I/O blocks.

See PNIO modules I/O mapping, page 82

IEC objects: When the PROFINET I/O controller is defined, libraries "IoDrvCIFXPNDevice.library" and/or RIL_PROFINETIODevice.library automatically applied. In this way, the memory space required to implement the PROFINET I/O device can be defined.

Bus cycle options: By selecting a bus cycle task, the cycle of the mapping exchange for the PROFINET I/O controller can be connected to a particular task. In this task, it is useful to process the I/O data of the controller as well.

Tab "Field Bus Mapping"

Field bus mapping

Field bus mapping enables access to control variables from a parent field bus master.

The following field buses are supported here:

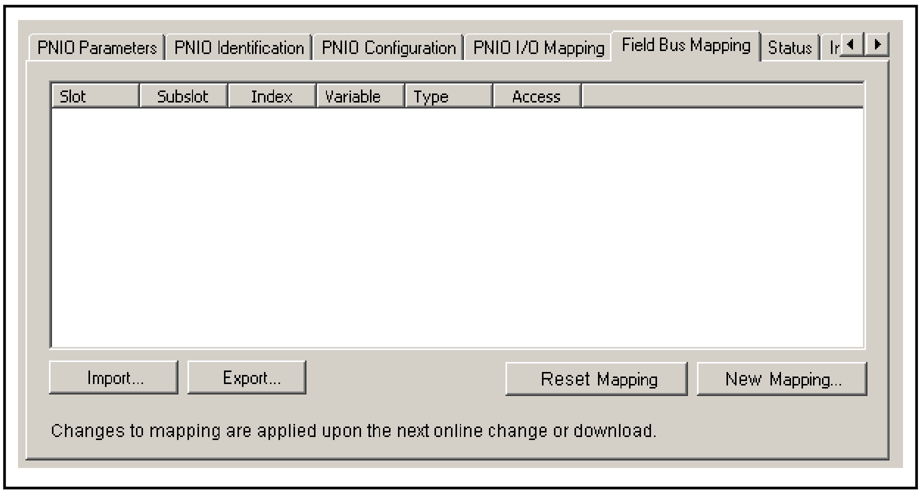
- PROFIBUS DP
- PROFINET I/O
- EtherNet/IP

To access a control variable from a parent master, an address must first be assigned to these variables (mapping). The parent master can use this address to access the control data using acyclic services.

To use field bus mapping, the control must be configured accordingly as a bus participant, e.g., as a PROFINET I/O device.

Configuration

The "Field bus mapping" tab is located in the device editor of the respective device connection. Addresses can be assigned to individual variables here in a table.



Changes to mapping are applied upon the next online change or download.

Fig.4-29: PROFINET I/O devices: Field bus mapping

Creating new mapping

A new entry can be added using the "New mapping" button. A dialog opens fro selecting a variable.

Variables declared within the application can be selected:

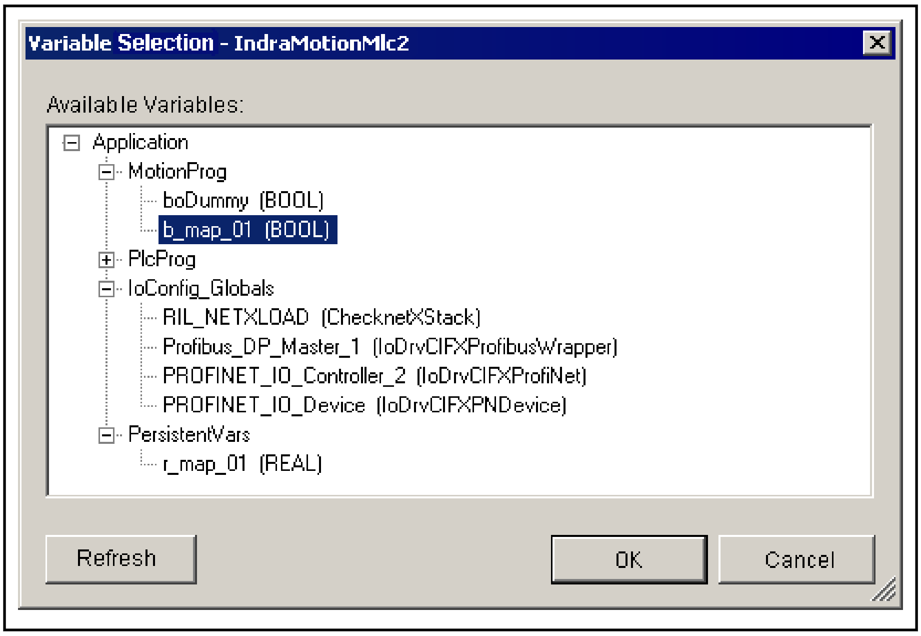

Variable Selection - IndraMotionMlc2
Available Variables:
Application
MotionProg
boDummy (BOOL)
b_map_01 (BOOL)
PlcProg
IoConfig_Globals
RIL_NETXLOAD (ChecknetXStack)
Profibus_DP_Master_1 (IoDrvCIFXProfibusWrapper)
PROFINET_IO_Controller_2 (IoDrvCIFXProfiNet)
PROFINET_IO_Device (IoDrvCIFXPNDevice)
PersistentVars
r_map_01 (REAL)
Refresh
OK
Cancel

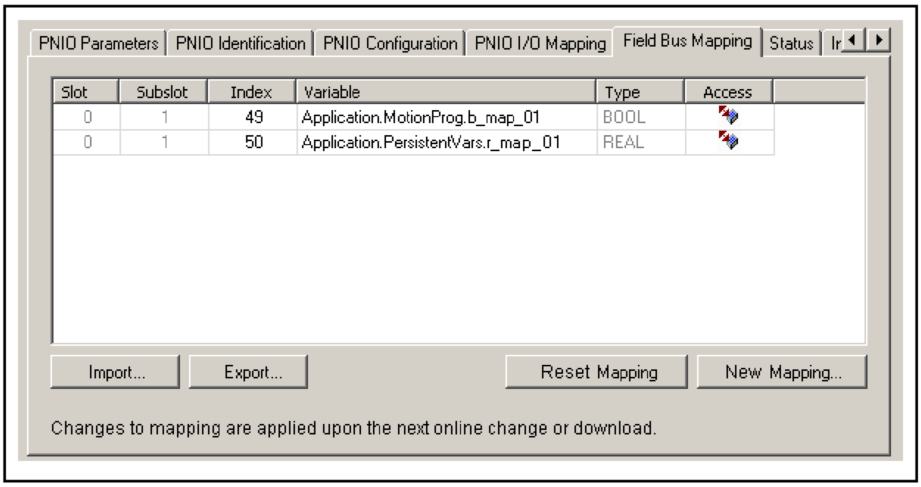
Fig.4-30: PROFINET I/O devices: field bus mapping, selecting variables

If the selection is confirmed with "OK", a new entry is created in the table for the selected variable.

In addition, the variable is added to the symbol configuration.

The variable section only contains variables that were present at the most recent compilation. If new variables have been added to the PLC program since that time, they are only visible after a new compilation is performed. This can be performed with the "Update" button.

Address, variable and access can be changed afterward by clicking on the corresponding table cell. Gray columns cannot be edited.



Slot	Subslot	Index	Variable	Type	Access
0	1	49	Application.MotionProg.b_map_01	BOOL	⇄
0	1	50	Application.PersistentVars.r_map_01	REAL	⇄

Changes to mapping are applied upon the next online change or download.

Fig.4-31: PROFINET I/O devices: field bus mapping, variables accepted

The address can be specified in the first columns in the table, but the values in the grayed out fields cannot be changed.

For PROFINET I/O the address consists of slot, subslot and index. Here slot is always 0 and subslot is always 1. Index must be in the range from 49 to 32767.

The Variable column displays the variables' instance path.

The Type column displays the variables' data type.

The Access column displays the access rights for the variables.

Reset mapping

The "Reset Mapping" button can be used to delete all of the table entries.

Import/Export

To save an existing mapping in a file, use the "Export…" button. It can be read out later using the "Import…" button.

This file is a simple text file.

For this reason, this file can be created manually without a previous export.

The export file contains the name for the field bus (EtherNet IP, PROFINETIO, ProfibusDP) and a table (see program listings).

The address is divided into several columns and consists of the following:

- ProfibusDP: Slot, index,
- PROFINETIO: Slot, Subslot, Index,
- EthernetIP: Class, Instance, Attribute.

This table also includes the variable name, the data type and the access rights of the mapped variables.

```
ProfinetIO
# Slot Subslot Index Variablename                        Datatyp Access right
  1    1       49    Application.MotionProg.b_map_01     BOOL    ReadWrite
  1    1       50    Application.PersistentVars.r_map_01 REAL    ReadWrite
```

Fig.4-32: Excerpt from the export file, separator TAB:

Tab "Status"

The "Status" tab in the "PROFINET I/O device editor" displays status information (e.g. "Running", "Stopped") and specific diagnostic messages from the device. In addition, the diagnostic messages contained in the status flags are displayed in "Diag".

The output is a hexadecimal value determined by the set status flags (see the "Protocol Interface Manual" for the PROFINET I/O).

See Diagnostics in the PROFINET I/O, page 82.

Tab "Information"

The window displays some general information from the device description file:

Name, Vendor, Categories, Version, Order number, Description, Image, if available.

4.3.5 PROFINET I/O Modules

PROFINET I/O Modules, General Information

The modules are located in the "Periphery" library in the "PROFINETIO" folder.

Drag the required modules out of the library into the PROFINET object.

Modules can also be added between existing modules in Project Explorer in this way.

Optionally, modules can be added in the context menu via **Add ▸ ...** of the PROFINET object.

In this case, the new module is added as the last module under the PROFINET object.

The dialog regarding the respective module appears after you double click on the module in the Project Explorer.

In PROFINET I/O, two types of modules are used:

- Modules that provide memory space for data exchange in the mapping memory of the PROFINET I/O controller or PROFINET I/O device.

  See also: PROFINET I/O controller, module for coupling a PROFINET I/O device, page 65.

  See also: PROFINET I/O device, page 75.

- I/O modules that accept the communication with the controlled object.

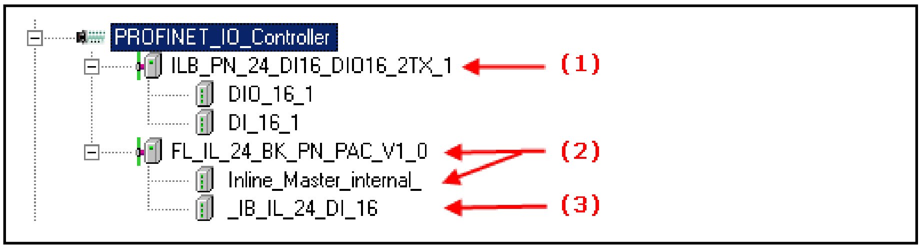
  PROFINET I/O distinguishes between two types of I/O modules:

  1. **Compact**: For a compact module, the module structure is specified.

     After a module has been added in Project Explorer, the modules below the module are already present in their complete, compact form. The terminals are not visible in the library.

  2. **Modular**: The module structure is variable.

     Based on a bus terminal (with its own existing I/Os, if required), terminals can be arranged individually - but according to the device placement specifications.



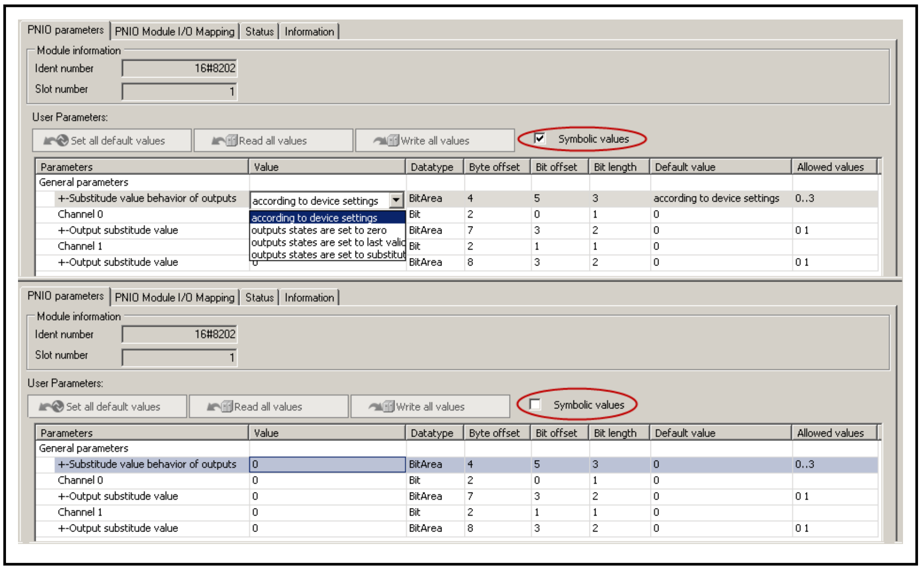
(1) Compact module with fixed module structure
(2) Module with inputs/outputs associated with the bus terminal
(3) Terminals that were assigned individually

Fig.4-33: I/O modules at the PROFINET I/O Controller

PROFINET I/O modules

- PNIO Parameters, page 80
- PNIO module configuration, page 81, this window is used for service purposes and is only visible if the option "Show generic configuration editors" is enabled under **Tools ▸ Options ▸ IndraLogic 2G ▸ Device editor.**
- PNIO modules I/O mapping, page 82
- Status, page 82
- Information, page 82

Tab "PNIO Parameters"



Module information

Ident number 16#8202

Slot number 1

User Parameters:

☑ Symbolic values

Parameters	Value	Datatype	Byte offset	Bit offset	Bit length	Default value	Allowed values
General parameters							
+-Substitude value behavior of outputs	according to device settings	BitArea	4	5	3	according to device settings	0..3
Channel 0		Bit	2	0	1	0	
+-Output substitude value		BitArea	7	3	2	0	0 1
Channel 1		Bit	2	1	1	0	
+-Output substitude value	0	BitArea	8	3	2	0	0 1

according to device settings
outputs states are set to zero
outputs states are set to last valid
outputs states are set to substitut

Module information

Ident number 16#8202

Slot number 1

User Parameters:

☐ Symbolic values

Parameters	Value	Datatype	Byte offset	Bit offset	Bit length	Default value	Allowed values
General parameters							
+-Substitude value behavior of outputs	0	BitArea	4	5	3	0	0..3
Channel 0	0	Bit	2	0	1	0	
+-Output substitude value	0	BitArea	7	3	2	0	0 1
Channel 1	0	Bit	2	1	1	0	
+-Output substitude value	0	BitArea	8	3	2	0	0 1

Fig.4-34: PROFINET I/O modules: PNIO parameters

Module information

- **ID number**: Identification of the module from the device description.
- **Slot number**: Position of the module under the device, starts with "1" for the first module and increases for each other module, resulting automatically from the current structure in the device tree.

User parameters

Based on the device, there are either no setting possibilities or there are several setting possibilities for the user-specific parameters:

- **Symbolic values**: You can use the "Symbolic values" option to enable symbolic value display. When it is deactivated, the numeric values are displayed.
- **Length of the user parameters**: The total length of all user parameters is displayed in bytes.

In the parameter table you can edit the values by double-clicking on them. Depending on the parameter this occurs with a selection list or by directly entering a value.

- **Parameters**: Name of the parameter of the parameter category (without value assignment).
- **Value**: Current parameter value
- **Type of data**: Type of parameter data, e.g. "bit"
- **Byte offset**: The parameters defined by the user are saved in the "Record Data" (array of bytes) variables. The byte offset specifies the first valid byte.
- **Bit offset**: The parameters defined by the user are saved in the "Record Data" (array of bytes) variables. The bit offset specifies the first valid bit, the byte identified by the byte offset.
- **Bit length**: Length of the information saved in the "Record Data" (array of bytes) variables which contain the parameters defined by the user.
- **Default Value**: Default value of the parameter.
- **Permissible values**: Specification as to what may be entered in the "Value" column.

  Syntax:

  `<parameter type> (<used bits>) <base value> <permissible value range>`.

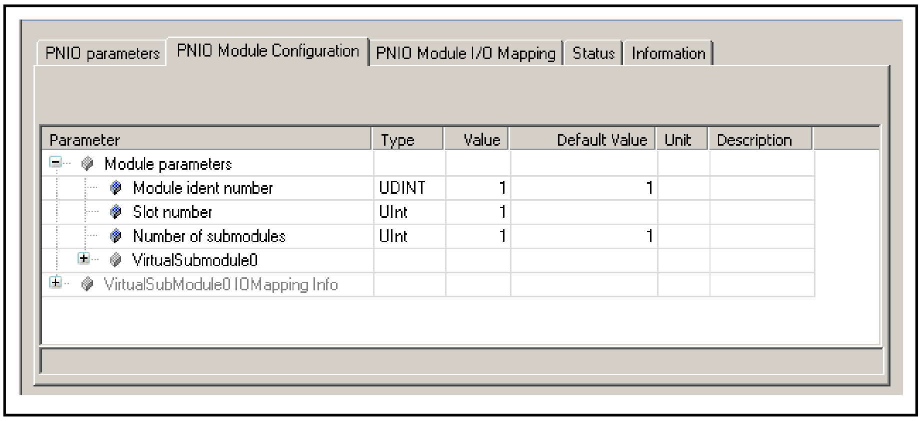
  Example:

  "BitArea (4-5) 0 0-2" means: This is a bit combination stored in bits 4 and 5 of the configuration byte. The base value is 0; the value can lie between 0 and 2.
- *Interfaces "User parameters" (only active online)*
  - **Set all default values**: Clicking this button replaces all the changed settings with the default settings from the GSDML file.
  - **Read all values**: Reading the current values of the device.
  - **Write all values**: Writing the current values from the following devices.

Tab "PNIO Module Configuration"

This window is used for service purposes and is only visible if in **Tools** ▸ **Options** ▸ **IndraLogic 2G** ▸ **Device Editor** the option "Display Generic Configuration Views" was enabled.

☞ Please clarify any possible modifications to the parameters that can be edited with the service team.



Parameter	Type	Value	Default Value	Unit	Description
Module parameters					
Module ident number	UDINT	1	1		
Slot number	UInt	1			
Number of submodules	UInt	1	1		
VirtualSubmodule0					
VirtualSubModule0 IOMapping Info					

Fig.4-35: PROFINET I/O modules: PNIO module configuration

The dialog contains information on the position and size of the parameters.

Tab "PNIO Module I/O Mapping"

The window is used to assign module inputs and outputs to variables that can be used as local or global variables in the individual POUs.

The current value of the variables is displayed in online mode.

This assignment is described in Mapping the Onboard, Inline and Field Bus Inputs and Outputs, page 139,.

Tab "Status"

The "Status" tab in the "PROFINET I/O device editor" displays status information (e.g., "Running", "Stopped") and specific diagnostic messages from the device.

See Diagnostics in the PROFINET I/O, page 82.

Tab "Information"

The window displays some general information from the device description file:

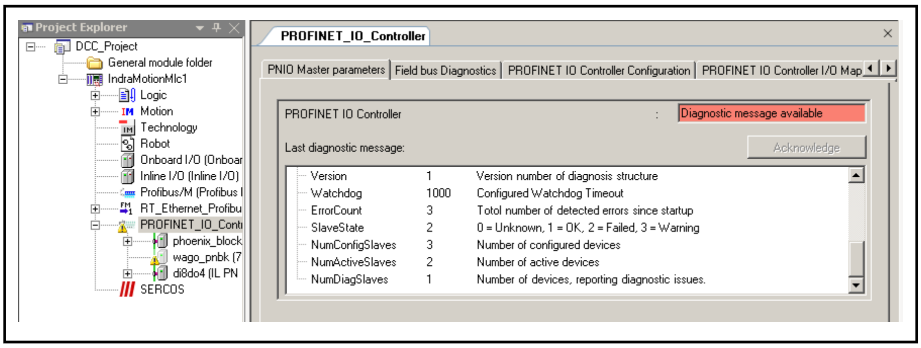
Name, Vendor, Categories, Version, Order number, Description, Image, if available.

4.3.6 Diagnostics in PROFINET I/O

Diagnostics - IndraWorks (interface)

In PROFINET I/O, diagnostics is supported by the IndraWorks interface as long as the bus is reached online.

This starts as soon as the PNIO I/O devices are scanned:



PROFINET IO Controller : Diagnostic message available

Last diagnostic message:

Version	1	Version number of diagnosis structure
Watchdog	1000	Configured Watchdog Timeout
ErrorCount	3	Totol number of detected errors since startup
SlaveState	2	0 = Unknown, 1 = OK, 2 = Failed, 3 = Warning
NumConfigSlaves	3	Number of configured devices
NumActiveSlaves	2	Number of active devices
NumDiagSlaves	1	Number of devices, reporting diagnostic issues.

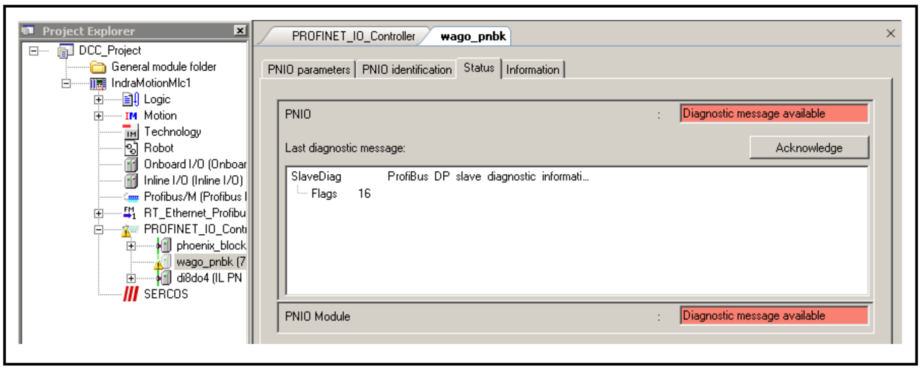
Fig.4-36: Diagnostic messages in PROFINET (PNIO controller)

A recognized device that functions without error has the usual icon in the Project Explorer. A device that is not recognized or recognized only partially also features a yellow triangle with an exclamation point.

The diagnostics are inherited from the cause "upwards" to the controller, i.e. the analysis should begin from the controller:

- SlaveState: 2 - with error
- NumConfigSlaves: 3 - number of configured devices
- NumActiveSlaves: 2 - number of active devices
- NumDiagSlaves: 1 - number of devices with diagnostic reports

Continuing with the bus terminal that displays a diagnostic:



PNIO : Diagnostic message available

Last diagnostic message:

SlaveDiag ProfiBus DP slave diagnostic informati...
Flags 16

PNIO Module : Diagnostic message available

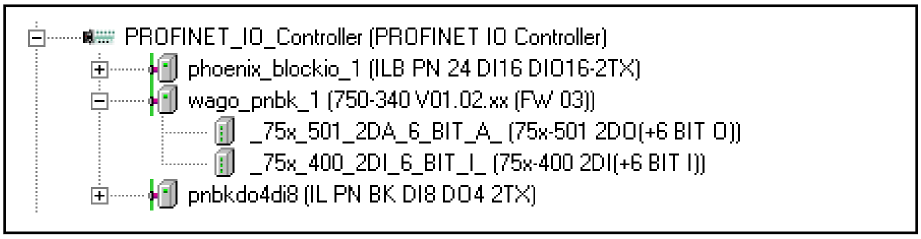
Fig.4-37: Diagnostic messages in PROFINET (PNIO bus terminal)

At the error location itself an error number that should be contained in the following table in output as diagnostic information.

BitNo	Value	Meaning
0	1	The device does not exist or does not respond to DCP ident requests.
1	2	The device is not ready
2	4	The device has a configuration error (e.g. its station name or IP address is used more than once in the network).
3	8	The device sends invalid responses (invalid response), e.g. DCP Set IP was not successful.
4	16	The device has a parameterization error (e.g. at bus startup, Connect_Request or Write_Record is rejected with an error code.
5	32	The device has been deactivated (by the user).
6	64	Diagnostic data are present.
7	128	The device sends the alarm "Diagnostic disappeared alarm".
8	256	The controller buffer for the diagnostic data was too small for the amount of diagnostic data sent by the device.
9	512	The controller buffer for diagnostic data was overwritten with new diagnostic data before the previous diagnostic data were read out from the controller.
10	1024	The diagnostic data requirement telegram is too small to accept the device's diagnostic data.
11	2048	The device reports the error ModuleDiffBlock while the connection is being established.

Fig.4-38: Error table, PROFINET I/O

When the bus terminal is scanned in, the I/O terminals are not actually accepted into the Project Explorer. They can be added manually.



```
PROFINET_IO_Controller (PROFINET IO Controller)
    phoenix_blockio_1 (ILB PN 24 DI16 DIO16-2TX)
    wago_pnbk_1 (750-340 V01.02.xx (FW 03))
        _75x_501_2DA_6_BIT_A_ (75x-501 2DO(+6 BIT O))
        _75x_400_2DI_6_BIT_I_ (75x-400 2DI(+6 BIT I))
    pnbkdo4di8 (IL PN BK DI8 DO4 2TX)
```

Fig.4-39: Bus terminals with (manually) added I/O terminals

4.4 Scanning PROFINET I/O Devices

4.4.1 Scanning Devices, Overview

The scanning command is provided to scan the hardware environment currently connected with the project.

Example: Determining the devices connected to a field bus.

- Switch the control online.
- Highlight the field bus onboard or on the function module.
- The command for scanning, page 84, is contained in the project menu and the in the field bus menu item.
- A device can only be properly detected if its device description file (containing manufacturer, device name, parameters) is existing in field-bus-specific format (PROFINET IO: GSDML file).

  If this file is missing for the actual participant, scanning can only be partially performed.

  The device description file must be reloaded and the scanning repeated (Reload device description file, page 89).

4.4.2 Scanning for Devices

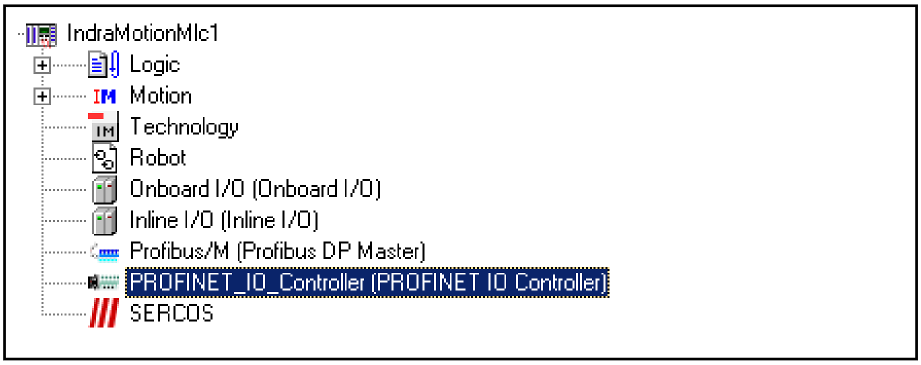
This command is used to scan the hardware that is currently controlled, i.e. to determine its structure, to display it in a dialog and to make it available to the user for transfer into the Project Explorer.

The scan functionality is created differently for different network types:

Its implementation can be fixed in the control (in a dialog of the wizard for creating the control) or it can be provided as a field bus menu item (onboard or function module).

In either case, to execute scan functions, a connection to the control is automatically established and then closed again.

The PLC gateway must also be configured and the control must be running.

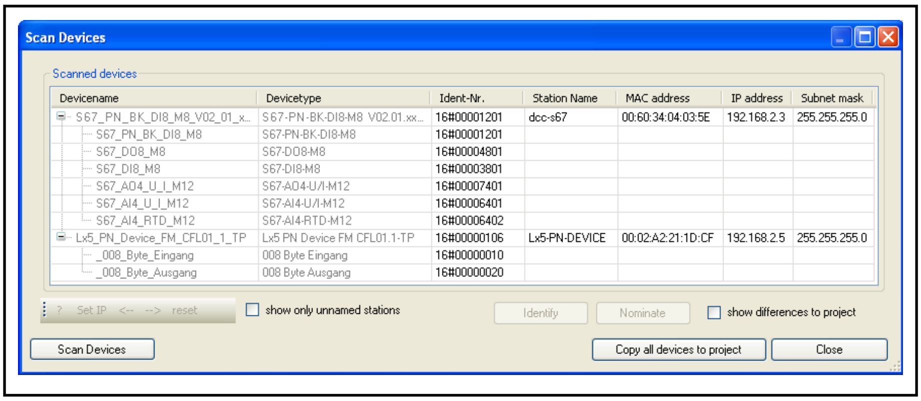


IndraMotionMlc1
- Logic
- Motion
- Technology
- Robot
- Onboard I/O (Onboard I/O)
- Inline I/O (Inline I/O)
- Profibus/M (Profibus DP Master)
- PROFINET_IO_Controller (PROFINET IO Controller)
- SERCOS

Fig.4-40: Highlighting the field bus to be scanned (online)

The scanning operation can be started via **Project ▸ Scan Devices** or **<Field bus> ▸ Scan Devices**.

The dialog lists the devices found in the current search and the related modules in the **Available devices** window.



Scan Devices

Scanned devices

Devicename	Devicetype	Ident-Nr.	Station Name	MAC address	IP address	Subnet mask
S67_PN_BK_DI8_M8_V02_01_x...	S67-PN-BK-DI8-M8 V02.01.xx...	16#00001201	dcc-s67	00:60:34:04:03:5E	192.168.2.3	255.255.255.0
S67_PN_BK_DI8_M8	S67-PN-BK-DI8-M8	16#00001201				
S67_DO8_M8	S67-DO8-M8	16#00004801				
S67_DI8_M8	S67-DI8-M8	16#00003801				
S67_AO4_U_I_M12	S67-AO4-U/I-M12	16#00007401				
S67_AI4_U_I_M12	S67-AI4-U/I-M12	16#00006401				
S67_AI4_RTD_M12	S67-AI4-RTD-M12	16#00006402				
Lx5_PN_Device_FM_CFL01_1_TP	Lx5 PN Device FM CFL01.1-TP	16#00000106	Lx5-PN-DEVICE	00:02:A2:21:1D:CF	192.168.2.5	255.255.255.0
_008_Byte_Eingang	008 Byte Eingang	16#00000010				
_008_Byte_Ausgang	008 Byte Ausgang	16#00000020				

□ show only unnamed stations

□ show differences to project

Fig.4-41: Dialog "Scan Devices"

The following functions are available in any case:

- The **Device name** and **Device type** for the devices found are displayed in the 'Available devices' window.

  Depending on the device type, the station address, the identification number and the status can be displayed.

- The <Scan devices> button can be used to start a new search. If the option **Show only differences to the project** is selected, the only devices shown are those that are not yet displayed in the device tree in the project.

- The <Copy into project> button can be used to accept the devices selected in the window, or if nothing is selected, all of the devices found, as entries in the device tree in the project.

  Depending on the device type, the standard dialog to add a device can also be opened. The device to be copied can be selected in the dialog (e.g. for Profibus).

PROFINET I/O

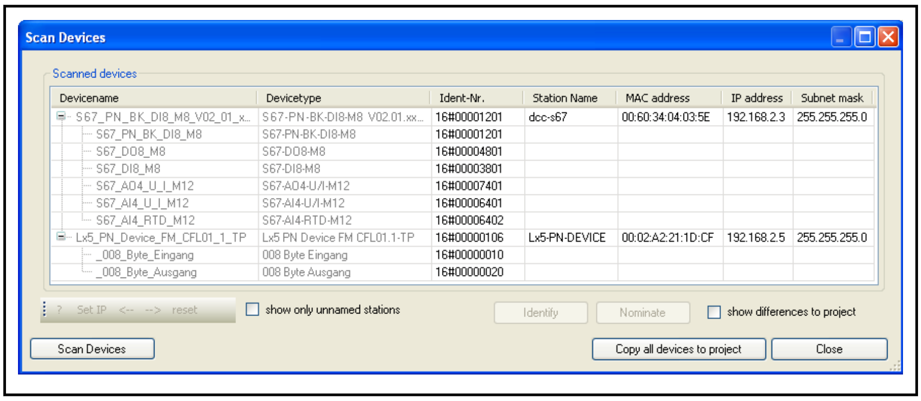
If values are changed in the list of available devices (e.g. IP address change), the corresponding value is displayed in italics.

This new value is only changed in the editor but not the device.

After the value has been loaded to the device, it is not shown in italics anymore.

Values displaying differences between project and available devices are displayed in orange.

Functions for PROFINET I/O devices:



Devicename	Devicetype	Ident-Nr.	Station Name	MAC address	IP address	Subnet mask
S67_PN_BK_DI8_M8_V02_01_x...	S67-PN-BK-DI8-M8 V02.01.xx...	16#00001201	dcc-s67	00:60:34:04:03:5E	192.168.2.3	255.255.255.0
S67_PN_BK_DI8_M8	S67-PN-BK-DI8-M8	16#00001201				
S67_DO8_M8	S67-DO8-M8	16#00004801				
S67_DI8_M8	S67-DI8-M8	16#00003801				
S67_AO4_U_I_M12	S67-AO4-U/I-M12	16#00007401				
S67_AI4_U_I_M12	S67-AI4-U/I-M12	16#00006401				
S67_AI4_RTD_M12	S67-AI4-RTD-M12	16#00006402				
Lx5_PN_Device_FM_CFL01_1_TP	Lx5 PN Device FM CFL01.1-TP	16#00000106	Lx5-PN-DEVICE	00:02:A2:21:1D:CF	192.168.2.5	255.255.255.0
_008_Byte_Eingang	008 Byte Eingang	16#00000010				
_008_Byte_Ausgang	008 Byte Ausgang	16#00000020				

? Set IP <-- --> reset
□ show only unnamed stations
Identify
Nominate
□ show differences to project
Scan Devices
Copy all devices to project
Close

Fig.4-42: Dialog "Scan Devices" for PROFINET I/O

The scanning functionality in a PROFINET I/O bus system can scan devices as well as modules. The following functions are available in addition to the general functions:

- For each device, the **station name**, **MAC address**, **IP address** and **subnet mask** are also displayed in other columns in the 'Available devices' window.

☞ Each device must have a **station name** because it is required for certain functionalities in network operation!

- If the filter **Show only unnamed stations** is selected, the only devices listed are those that do not yet have a station name.
- The **station name** can be changed in the dialog (editing by double-clicking on the field).

  The **Baptize** interface has to be used when the device entry is selected to inform the bus system about the new name.

☞ After the scanned data has been changed and the <Baptize> function has been used, "Scan Devices" has to be used again. The new device data is displayed.

- The entries for **Device name**, **IP address** and **Subnet mask** can also be edited or created in the dialog before the device settings are applied to the configuration in the project.

  When the device is applied with no defined addresses, the default settings defined in the PNIO Master Parameter Dialog, page 60, of the device editor will be used.

  The final definition of the addresses can still be made for each device in its configuration dialog PNIO identification (page 74)

- To identify a device in the hardware listed in the 'Available devices' window, select the entry in the dialog and click on the <Identify> button. The device should react with a blinking signal (often "Link ok").

  If the device is a control (CML L25, L45, L65, L85), information is displayed on the control.
- The **module ID number** is only displayed in the list, if

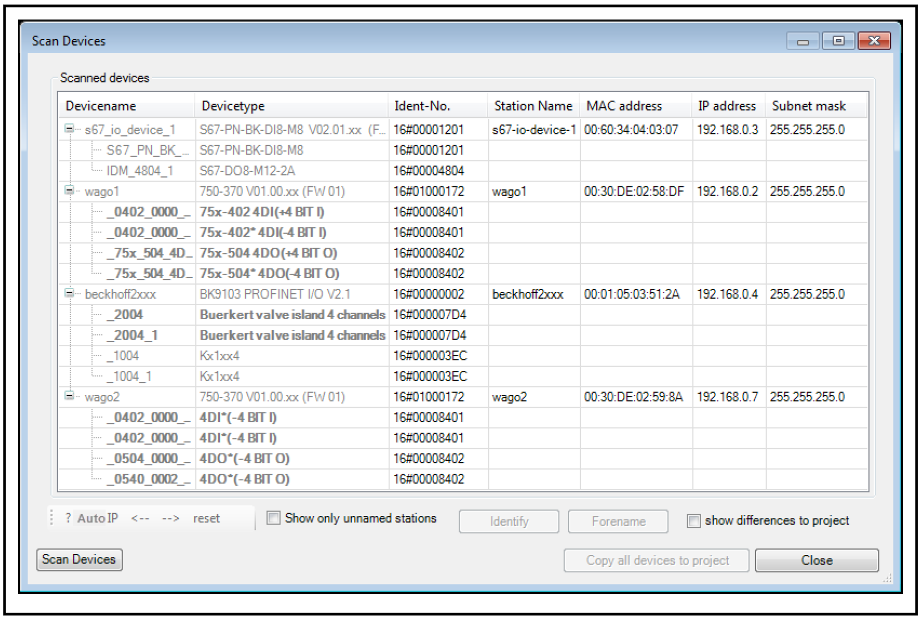
  1. it is provided by the device,
  2. if device and controller are part of the same logic network and
  3. if IP address and network mask of the device are set correctly.

  If the ID number is not available, a device cannot be "copied" to the project if it cannot be identified by other available information.

Auto-IP is used to set selected devices which do not have a valid IP configuration so that they match the controller settings and there will be no conflicts with other devices.

Please note the option to adjust the configuration settings of a PROFINET I/O device with the corresponding device adjust (page 88).

Show difference to project: This option opens the dialog in which the found devices and modules are compared to the configured devices.


Scan Devices

Scanned devices

Devicename	Devicetype	Ident-No.	Station Name	MAC address	IP address	Subnet mask
s67_io_device_1	S67-PN-BK-DI8-M8 V02.01.xx (F...	16#00001201	s67-io-device-1	00:60:34:04:03:07	192.168.0.3	255.255.255.0
S67_PN_BK_...	S67-PN-BK-DI8-M8	16#00001201				
IDM_4804_1	S67-DO8-M12-2A	16#00004804				
wago1	750-370 V01.00.xx (FW 01)	16#01000172	wago1	00:30:DE:02:58:DF	192.168.0.2	255.255.255.0
_0402_0000_...	**75x-402 4DI(+4 BIT I)**	16#00008401				
_0402_0000_...	**75x-402* 4DI(-4 BIT I)**	16#00008401				
_75x_504_4D...	**75x-504 4DO(+4 BIT O)**	16#00008402				
_75x_504_4D...	**75x-504* 4DO(-4 BIT O)**	16#00008402				
beckhoff2xxx	BK9103 PROFINET I/O V2.1	16#00000002	beckhoff2xxx	00:01:05:03:51:2A	192.168.0.4	255.255.255.0
_2004	**Buerkert valve island 4 channels**	16#000007D4				
_2004_1	**Buerkert valve island 4 channels**	16#000007D4				
_1004	Kx1xx4	16#000003EC				
_1004_1	Kx1xx4	16#000003EC				
wago2	750-370 V01.00.xx (FW 01)	16#01000172	wago2	00:30:DE:02:59:8A	192.168.0.7	255.255.255.0
_0402_0000_...	**4DI*(-4 BIT I)**	16#00008401				
_0402_0000_...	**4DI*(-4 BIT I)**	16#00008401				
_0504_0000_...	**4DO*(-4 BIT O)**	16#00008402				
_0540_0002_...	**4DO*(-4 BIT O)**	16#00008402				

? Auto IP <-- --> reset
□ Show only unnamed stations
Identify
Forename
□ show differences to project
Scan Devices
Copy all devices to project
Close

Fig.4-43: Dialog "Show differences to project"

The devices highlighted in green are identical whereas the devices highlighted in red are only listed in the overview of found or configured devices.

- Using the commands "Copy (before)" and "Copy (after)", the highlighted devices can be copied to the configured devices before and after the selected devices.
- A configured device cannot be replaced by a found device by executing the "Replace with" command. For this action, the concerned devices are highlighted.

Load ? I&M data: This command calls the I&M data (Identification and Maintenance) from the device and displays then in a dialog.

Set IP: This command automatically sets a valid IP address if the address scanned by the device is invalid.

<-- Assign online device: The settings of the configured device are assigned to the scanned device using this command.

--> Assign configured device: The settings of the configured device are assigned to the scanned device using this command.

Reset: Resetting the device settings to the default settings.

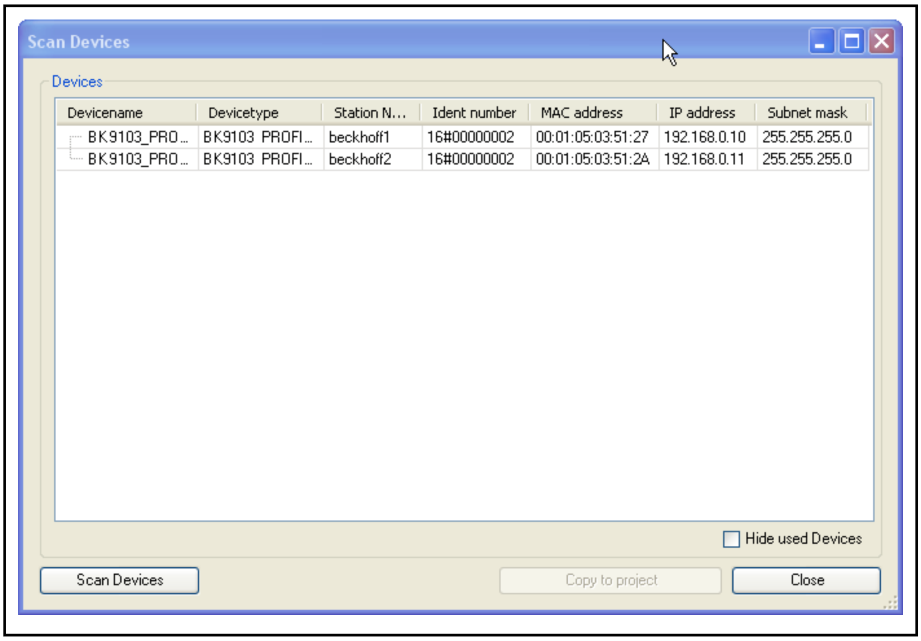
4.4.3 PROFINET I/O, Configuration Adjustment

☞ Please not that this functionality is only available for PROFINET I/O bus systems!

By using this command, the configuration settings of a PROFINET I/O slave device defined in the project can be easily synchronized with the settings of a device (same device and manufacturer ID) in the connected hardware. Consequently, the local settings are overwritten with the hardware settings.

Execute the following steps:

- Select communication settings.
- Select the device object in the device tree in offline mode and the command "Configuration adjustment" in the context menu.
- The dialog "Scan devices" is opened and the search for matching devices in the hardware is started. The found devices are listed. The devices already used in the project (identified by means of the station name) can be hidden with "Hide used devices".
- Scanning can be restarted using the "Scan device" interface.



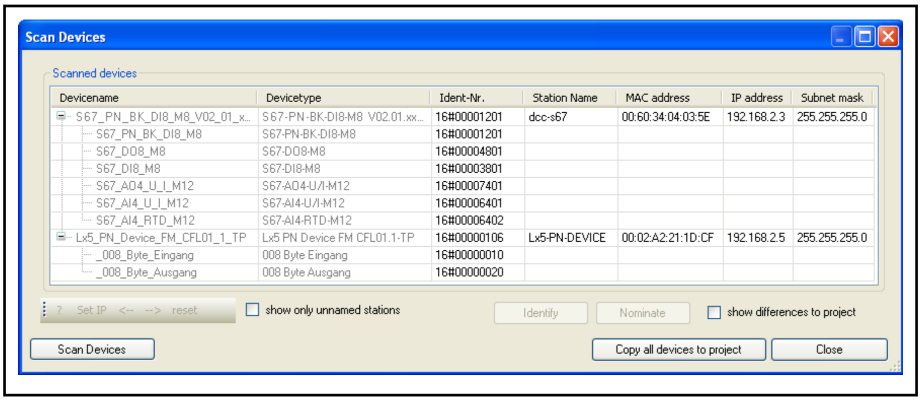
Devicename	Devicetype	Station N...	Ident number	MAC address	IP address	Subnet mask
BK9103_PRO...	BK9103 PROFI...	beckhoff1	16#00000002	00:01:05:03:51:27	192.168.0.10	255.255.255.0
BK9103_PRO...	BK9103 PROFI...	beckhoff2	16#00000002	00:01:05:03:51:2A	192.168.0.11	255.255.255.0

Fig.4-44: Dialog "Scan devices" for offline configuration adjustment, example

To overwrite the device configuration settings in the project with the settings of the hardware device, select the the device from the list and click on "Copy to project". Subdevice objects are not removed.

4.4.4 PROFINET I/O, Reloading the Device Description File

The user can extend the existing group of device description files if modules are used that are not among the standard scope. For example, see the following scan result.


Scan Devices
Scanned devices
? Set IP <-- --> reset
□ show only unnamed stations
Identify
Nominate
□ show differences to project
Scan Devices
Copy all devices to project
Close

Devicename	Devicetype	Ident-Nr.	Station Name	MAC address	IP address	Subnet mask
S67_PN_BK_DI8_M8_V02_01_x...	S67-PN-BK-DI8-M8 V02.01.xx...	16#00001201	dcc-s67	00:60:34:04:03:5E	192.168.2.3	255.255.255.0
S67_PN_BK_DI8_M8	S67-PN-BK-DI8-M8	16#00001201				
S67_DO8_M8	S67-DO8-M8	16#00004801				
S67_DI8_M8	S67-DI8-M8	16#00003801				
S67_AO4_U_I_M12	S67-AO4-U/I-M12	16#00007401				
S67_AI4_U_I_M12	S67-AI4-U/I-M12	16#00006401				
S67_AI4_RTD_M12	S67-AI4-RTD-M12	16#00006402				
Lx5_PN_Device_FM_CFL01_1_TP	Lx5 PN Device FM CFL01.1-TP	16#00000106	Lx5-PN-DEVICE	00:02:A2:21:1D:CF	192.168.2.5	255.255.255.0
_008_Byte_Eingang	008 Byte Eingang	16#00000010				
_008_Byte_Ausgang	008 Byte Ausgang	16#00000020				

Fig.4-45: Scan results; the device description file is missing for the middle device

To add devices in IndraWorks, please carry out the following steps:

1. Click on **Tools ▸ Device database...** in the main menu to open the "Device database" dialog.
2. In the "Device database" dialog, click on the "Add devices" button.

   The "Install device description" file selection dialog opens:

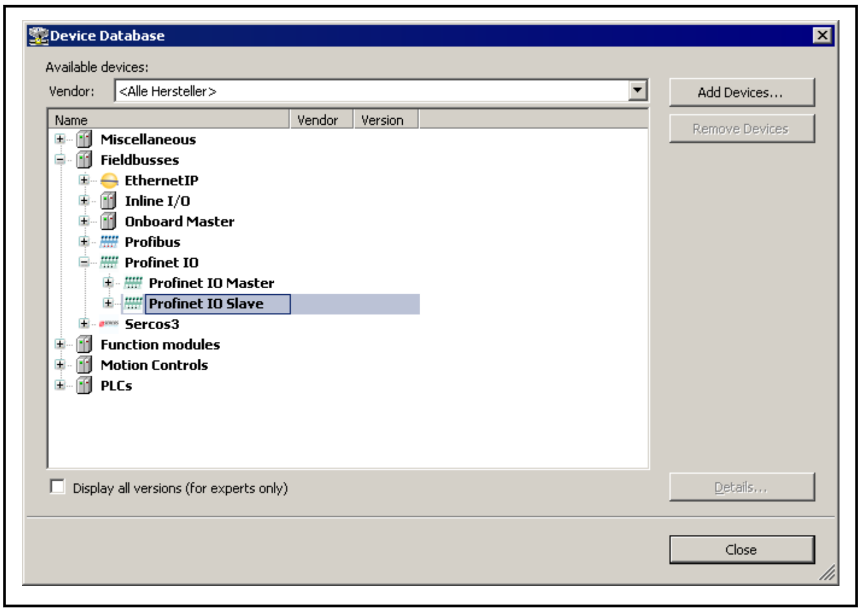
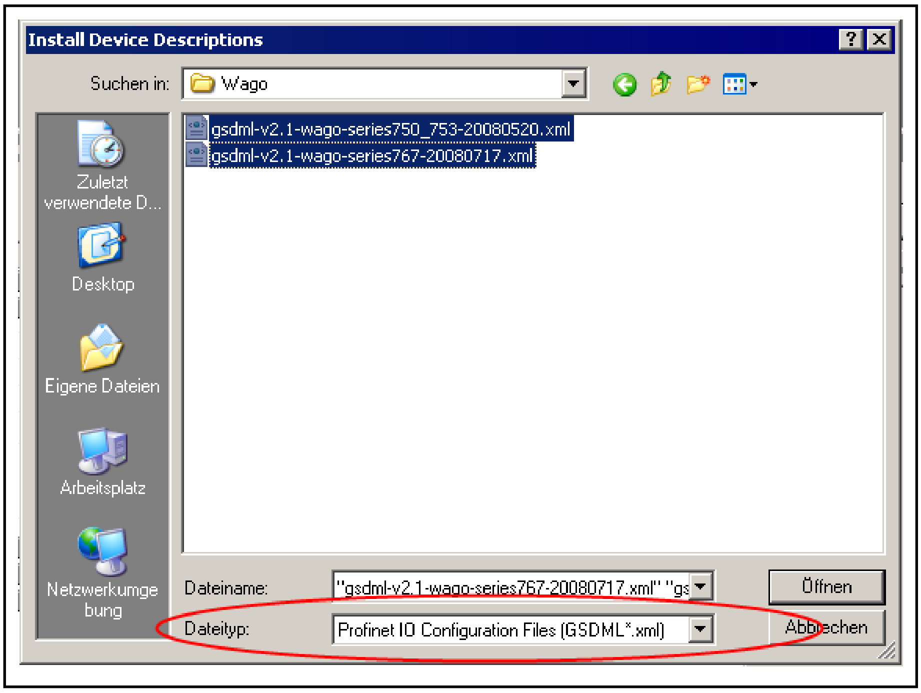


Fig.4-46: Device Database, add device

3. Select the device description files to be installed. Make sure that the correct file type is listed in the "File type" selection field.



Suchen in: Wago

gsdml-v2.1-wago-series750_753-20080520.xml
gsdml-v2.1-wago-series767-20080717.xml

Dateiname: "gsdml-v2.1-wago-series767-20080717.xml" "gs

Dateityp: Profinet IO Configuration Files (GSDML*.xml)

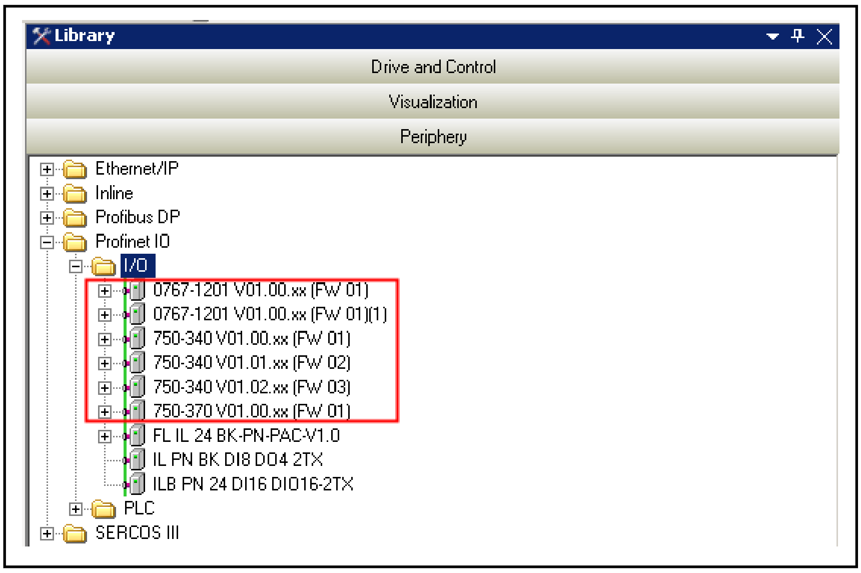
Fig.4-47: Install Device Descriptions dialog

4. Start the installation procedure by clicking on "Open".

   After completed installation, a dialog appears confirming that all device description files have been imported. The imported devices appear below "Available devices" and are highlighted there:

5. Confirm the "Install Device Descriptions" message with "OK".
6. Close the "Device database" dialog with "Close".

   The newly installed devices are also displayed in the library:



```
Ethernet/IP
Inline
Profibus DP
Profinet IO
  I/O
    0767-1201 V01.00.xx (FW 01)
    0767-1201 V01.00.xx (FW 01)(1)
    750-340 V01.00.xx (FW 01)
    750-340 V01.01.xx (FW 02)
    750-340 V01.02.xx (FW 03)
    750-370 V01.00.xx (FW 01)
    FL IL 24 BK-PN-PAC-V1.0
    IL PN BK DI8 DO4 2TX
    ILB PN 24 DI16 DIO16-2TX
  PLC
SERCOS III
```

Fig.4-48: Imported devices displayed in the library

For more details on importing device files, please refer to:

- Device Database, page 135.

5 EtherNet/IP

5.1 EtherNet/IP Adapter

5.1.1 Terms and Abbreviations

Scanner	The EtherNet/IP field bus master is called **scanner**.
Adapter	The EtherNet/IP field bus slaves are called **adapters**.
Originator	The device that establishes the connection (usually the scanner) is called the **originator**.
Target	The device to which the connection is established is a **Target**.
O→T	Means Originator to Target.
T→O	Means Target to Originator.
RPI	**RPI** is the "requested packet interval“. This value designates the send or response cycle time for a connection. Typically identical RPIs are selected for all connections.
EDS file	This is the device description with which the EtherNet/IP adapter is made known to IndraWorks (EDS: Electronic Data Sheet).
ODVA	Open DeviceNet Vendor Association www.odva.org
CIP	Common Industrial Protocol

5.1.2 EtherNet/IP Adapter Features

The current implementation of the adapter includes the following functionalities:

- Cyclic communication (implicit messaging)
- Diagnostic of the adapter
- Acyclic communication in accordance with the mapping concept

Future extensions are planned for the following functionalities:

- CIP sync services
- TAGs

Function/Characteristic	Value
Max. amount of input data	480 bytes
Max. amount of output data	480 bytes
IO connection	1 explicit owner, up to 2 listeners
IO connection type	Cyclic, min. 2 ms

Function/Characteristic	Value
Explicit messages	*CIP standard services:* • Set_Attribute_Single • Set_Attributes_All • Get_Attribute_Single • Get_Attribute_All *Rockwell-specific services:* • ReadDataTable • WriteDataTable • ReadFragmentedData • WriteFragmentedData
UCMM	Supported
Max. number of user-specific objects	20
Max. number of connections	8 explicit and implicit connections
DCHP	Supported
BOOTP	Supported
Baud rate	10 and 100 Mbit/s

Fig.5-1: Technical data

Supported CIP classes

CIP classes are contained in the CIP specification of the ODVA.

They describe the properties of the objects, irrespective of the physical interface, e.g., EtherNet, CAN (volume 1).

The physical interface is described in another specification.

This is volume 2 for EtherNet/IP, which describes how EtherNet/IP is adapted to CIP.

Use is made of classes 1, 2, 4 - which are described in volume 1 ("Common Industrial Protocol").

The classes 16#F5 and 16#F6 of volume 2 (EtherNet/IP adaptation of CIP) are supported.

A "Vendor Specific Object" 0xC7 is defined for acyclic data communication.

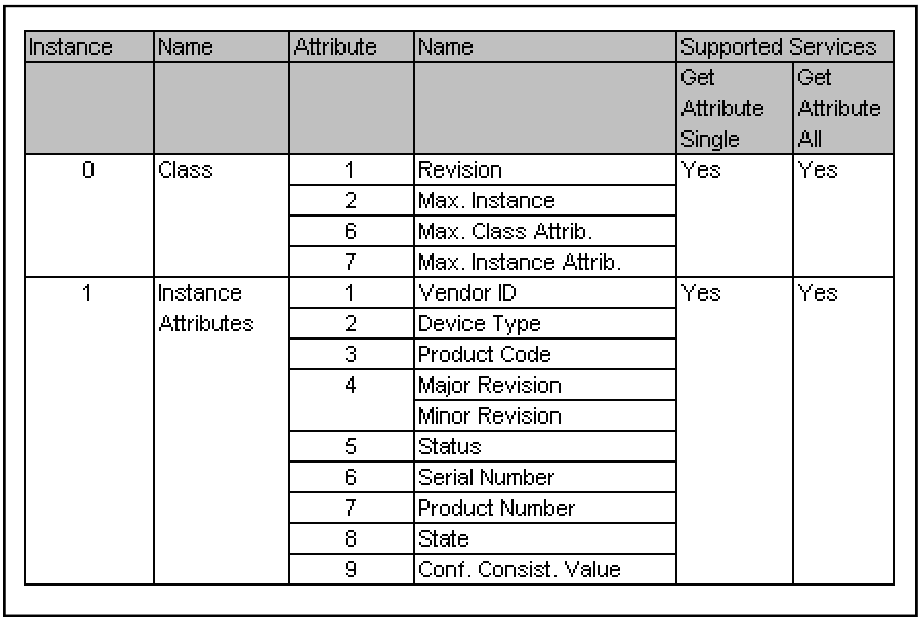
The chapters below contain a detailed description of the classes listed in the following table.

Class	Name
16#01	Identity Object
16#02	Message Router Object
16#04	Assembly Object
16#F5	TCP/IP Interface Object
16#F6	EtherNet Link Object
16#C7	Vendor Specific Object

Fig.5-2: Supported CIP classes

Indentity Object (16#01)

The Identity class serves to provide general adapter information uniquely identifying the adapter.



Instance	Name	Attribute	Name	Supported Services	
				Get Attribute Single	Get Attribute All
0	Class	1	Revision	Yes	Yes
		2	Max. Instance		
		6	Max. Class Attrib.		
		7	Max. Instance Attrib.		
1	Instance Attributes	1	Vendor ID	Yes	Yes
		2	Device Type		
		3	Product Code		
		4	Major Revision		
			Minor Revision		
		5	Status		
		6	Serial Number		
		7	Product Number		
		8	State		
		9	Conf. Consist. Value		

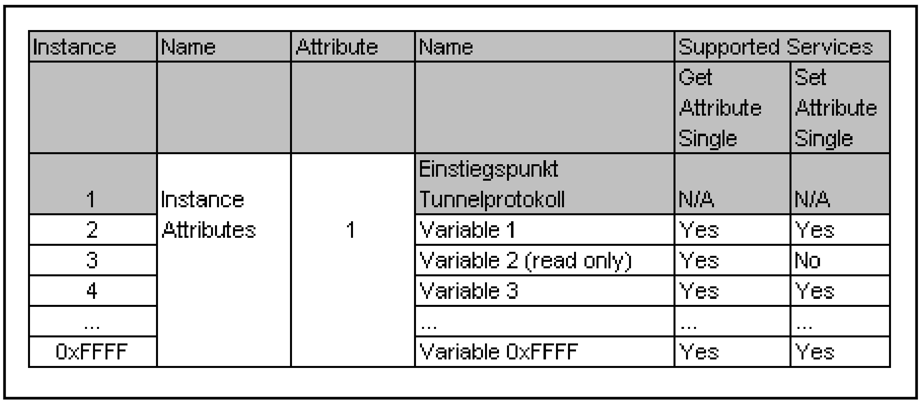
Fig.5-3: "Identity Object" class: Class attribute (instance = 0) and instance attribute (instance = 1)

Message Router Object (16#02)

The "Message Router Object" provides connection points in the form of classes or instances, which a client can use to address services (read, write). These messages can be sent from the client to the adapter both based on the connection (connected) and without connection (unconnected).

No services are supported for the Message Router Object (Predefined Standard Object 16#02).

A "Vendor Specific Object" 16#C7 is defined on the Message Router for acyclic communication. This object supports the Get/Set_Attribute_Single services. Various "Vendor Specific Services" can be defined for this object (e.g., the Read/Write DataTable, Read/Write Fragmented Data services).

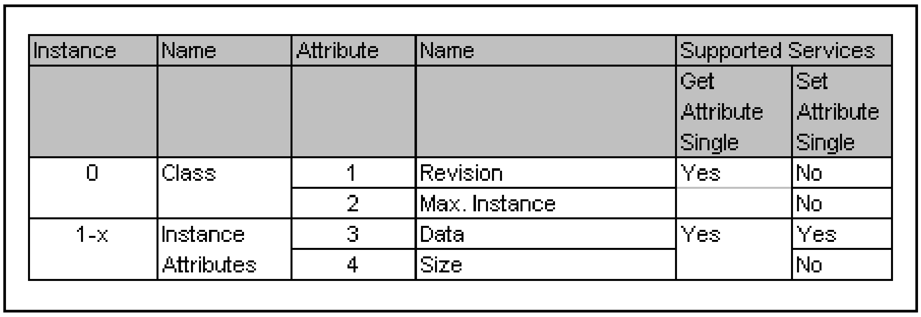


Instance	Name	Attribute	Name	Supported Services	
				Get Attribute Single	Set Attribute Single
1	Instance Attributes	1	Einstiegspunkt Tunnelprotokoll	N/A	N/A
2			Variable 1	Yes	Yes
3			Variable 2 (read only)	Yes	No
4			Variable 3	Yes	Yes
...			...	...	...
0xFFFF			Variable 0xFFFF	Yes	Yes

Fig.5-4: Class "Vendor Specific Object (16#C7)": Only instance attribute (instance = 1, 2, 3, …, 16#FFFF)

Assembly Object (16#04)

The Assembly class can be used to combine several objects, even if they are different. Such objects can, e.g., be input and output data. The manufacturer-specific instances are used to provide these objects in various arrangements. This results in an efficient way of exchanging process data.



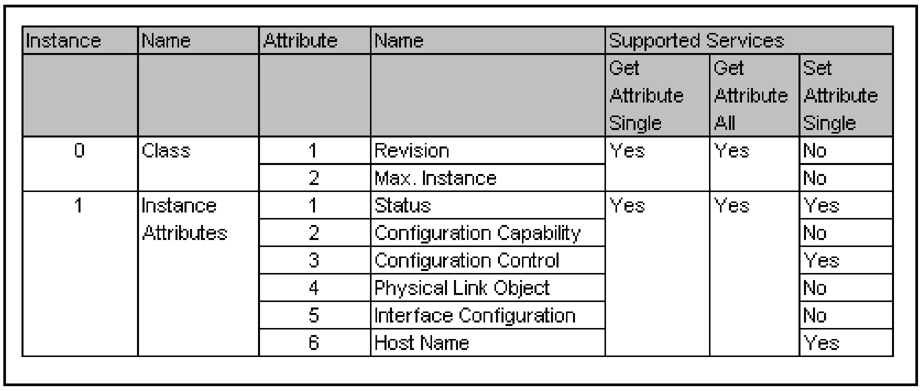
Instance	Name	Attribute	Name	Supported Services	
				Get Attribute Single	Set Attribute Single
0	Class	1	Revision	Yes	No
		2	Max. Instance		No
1-x	Instance Attributes	3	Data	Yes	Yes
		4	Size		No

Fig.5-5: Class "Assembly Object": Class attribute (instance = 0) and instance attribute (instance = 1)

TCP/IP Interface Object (16#F5)

The "TCP/IP Interface Object" provides the setup for configuring the TCP/IP network interface of an adapter.

Examples of configurable objects are IP address, network screen and gateway address of the adapter.

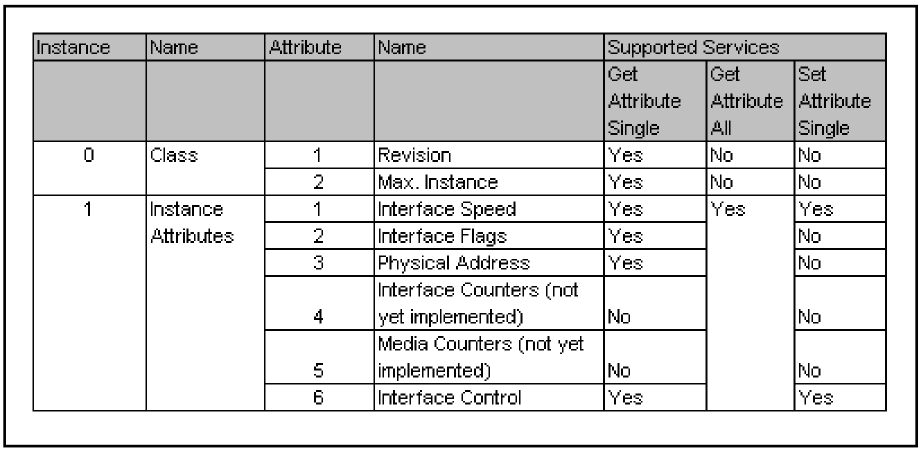


Instance	Name	Attribute	Name	Supported Services		
				Get Attribute Single	Get Attribute All	Set Attribute Single
0	Class	1	Revision	Yes	Yes	No
		2	Max. Instance			No
1	Instance Attributes	1	Status	Yes	Yes	Yes
		2	Configuration Capability			No
		3	Configuration Control			Yes
		4	Physical Link Object			No
		5	Interface Configuration			No
		6	Host Name			Yes

Fig.5-6: Class "TCP/IP Interface Object": Class attribute (instance = 0) and instance attribute (instance = 1)

EtherNet Link Object (16#F6)

The "EtherNet Link Object" contains link-specific counter and state information for a communication interface of the EtherNet type.



Instance	Name	Attribute	Name	Supported Services		
				Get Attribute Single	Get Attribute All	Set Attribute Single
0	Class	1	Revision	Yes	No	No
		2	Max. Instance	Yes	No	No
1	Instance Attributes	1	Interface Speed	Yes	Yes	Yes
		2	Interface Flags	Yes		No
		3	Physical Address	Yes		No
		4	Interface Counters (not yet implemented)	No		No
		5	Media Counters (not yet implemented)	No		No
		6	Interface Control	Yes		Yes

Fig.5-7: Class "EtherNet Link Object": Class attribute (instance = 0) and instance attribute (instance = 1)

5.1.3 EtherNet/IP Adapter Features (Engineering)

The current implementation of the adapter (Engineering) includes the following functionalities:

- Cyclic communication (implicit messaging)
- Diagnostic of the adapter

Function/Characteristic	Value
Max. amount of input data	128 bytes[1)]
Max. amount of output data	128 bytes
IO connection	1 explicit owner, 1 listener
IO connection type	Cyclic, min. 5 ms
Explicit messages	Get_Attribute Set_Attribute
UCMM	Supported
Max. number of user-specific objects	-
Max. number of connections	1 explicit and 1 implicit connection
DCHP	-
BOOTP	-
Baud rate	10 and 100 Mbit/s

Fig.5-8: Technical data

5.2 Configuring the EtherNet/IP Adapter

5.2.1 Configuring the EtherNet/IP Adapter, Overview

An EtherNet/IP adapter can be used in the following controls:

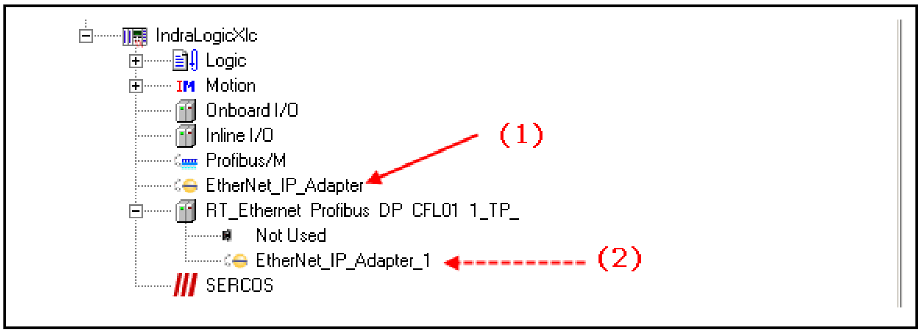
- IndraLogic XLC L25
  - With sercos III:
    EtherNet/IP adapter, function modules
  - Without sercos III:
    EtherNet/IP adapter, onboard
    EtherNet/IP adapter, function modules
- IndraMotion MLC L25: EtherNet/IP adapter, function modules
- IndraLogic XLC L45/L65 and IndraMotion MLC L45/L65
  EtherNet/IP adapter, onboard
  EtherNet/IP adapter, function modules

The engineering software available at the control is represented as an EtherNet/IP adapter object in the Project Explorer.

The object can either be configured as EtherNet/IP adapter when creating the control or via the context menu item **Set device**.

When an "EtherNet/IP adapter" is selected, library **IoDrvCIFXEIPAdapter** is automatically applied.

[1)] *The data width of the coupling area is preset to 8 bytes input data and 8 bytes output data plus a 4byte Run Header. It can be changed via the Set Device context menu.*



(1) EtherNet/IP adapter, onboard

(2) EtherNet/IP adapter, function module

Fig.5-9: EtherNet/IP adapter objects, still unassigned

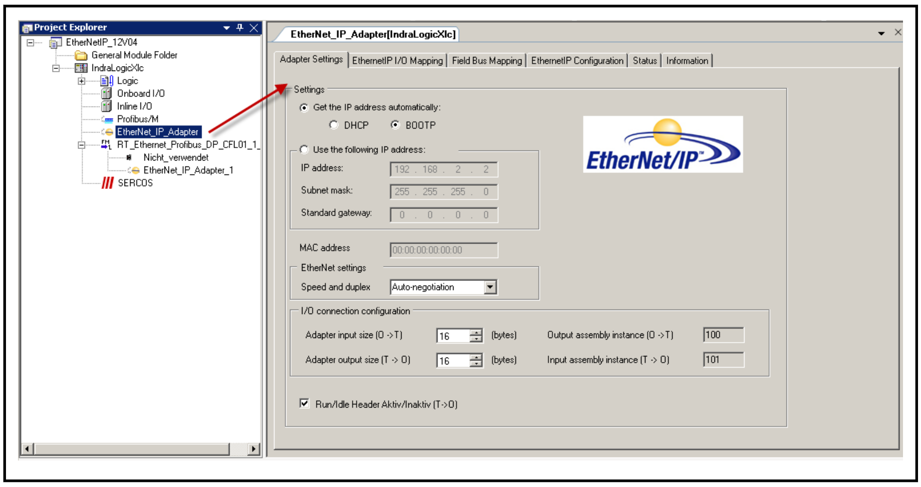
EtherNet/IP adapter

- Adapter settings, page 96,
- EtherNetIP mapping, page 97,
- User Parameter, ### projected ###,
- Field bus mapping, page 98
- EtherNetIP configuration, page 100, this window is used for service purposes and is only visible if the option "Show generic configuration editors" is enabled under **Tools ▸ Options ▸ IndraLogic 2G ▸ Device editor.**
- Status, page 101,
- Information, page 101.

5.2.2 Configuring the EtherNet/IP Adapter

"Adapter Settings" Tab

In the Adapter Settings tab, the target system settings of the EtherNet/IP adapters are selected.



Settings

● Get the IP address automatically:

○ DHCP ● BOOTP

○ Use the following IP address:

IP address: 192 . 168 . 2 . 2

Subnet mask: 255 . 255 . 255 . 0

Standard gateway: 0 . 0 . 0 . 0

MAC address 00:00:00:00:00:00

EtherNet settings

Speed and duplex Auto-negotiation

I/O connection configuration

Adapter input size (O ->T)	16	(bytes)	Output assembly instance (O ->T)	100
Adapter output size (T -> O)	16	(bytes)	Input assembly instance (T -> O)	101

☑ Run/Idle Header Aktiv/Inaktiv (T->O)

Fig.5-10: Configuring the EtherNet/IP adapter

IP address The IP-address of the EtherNet/IP adapters can be automatically obtained:

- **DHCP** - Get IP configuration of a DHCP server automatically.
- **BOOTP** - Get IP configuration of a BOOTP server automatically.

Optionally, the IP address can be specified manually via "Use the following IP address". The values for IP address, subnet mask and gateway address have to manually selected.

☞ If BOOTP/DHCP is enabled, the adapter attempts to reach the BOOTP/DHCP server for 56 seconds. If the adapter does not receive any IP configuration from a BOOTP/DHCP server, it applies the manually set values for IP address, subnet mask and gateway address.

The MAC address of the EtherNet/IP adapter is automatically read and is only displayed.

Ethernet settings

The velocity and the duplex of the Ethernet interface can be set under "Speed and Duplex". Possible values are:

- Auto-negotiation
- 10 Mbit/half duplex
- 10 MBit/full duplex
- 100 Mbit/half duplex
- 100 MBit/full duplex.

MAC address of netX interface: Initial "00:00:00:00:00:00", the real MAC address is only displayed when the user is logged in. The MAC address is empty again after the user logged out.

I/O connection configuration

In the EtherNet/IP, the input/output assembly instances have to be defined as follows, according to the scanner (originator):

- **Adapter input size (O -> T)** - Length of input data in byte
- **Adapter output size (T -> O)** - Length of output data in byte
- **Output assembly instance (O -> T)** - Display of output assembly instance
- **Input assembly instance (T -> O)** - Display of input assembly instance.

The specified length of the input and output data influences the setting options in the "EthernetIP I/O Mapping" tab.

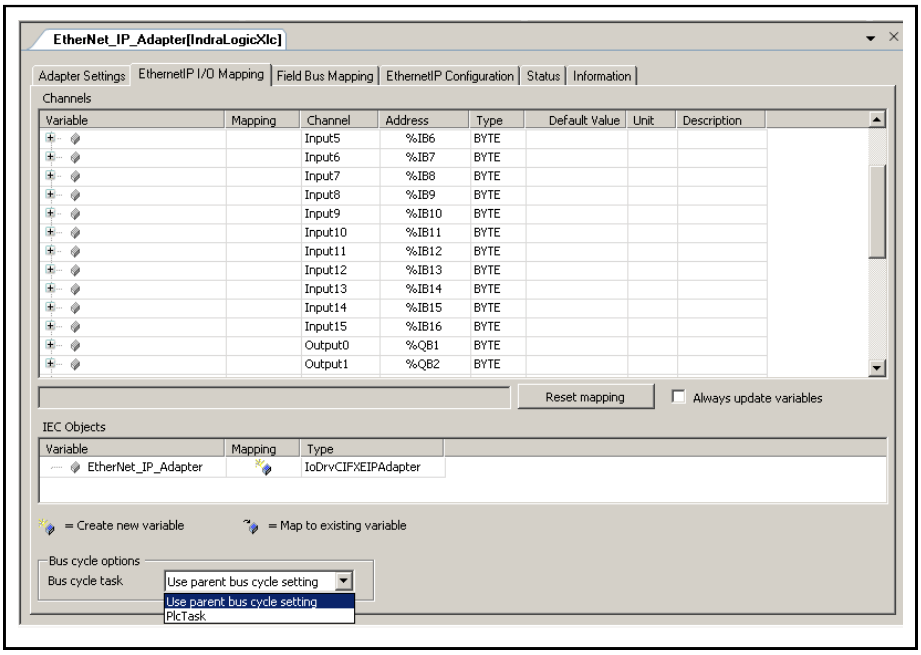
Header

Run/idle header (4 Byte) from adapter to scanner.

☞ This option has to be selected (Run/idle header active) if the scanner configuration has to be selected in the EDS file.

Tab "EtherNetIP I/O Mapping"

The tab permits mapping of variables for data exchange with the scanner.



EtherNet_IP_Adapter[IndraLogicXlc]

Channels

Variable	Mapping	Channel	Address	Type	Default Value	Unit	Description
		Input5	%IB6	BYTE			
		Input6	%IB7	BYTE			
		Input7	%IB8	BYTE			
		Input8	%IB9	BYTE			
		Input9	%IB10	BYTE			
		Input10	%IB11	BYTE			
		Input11	%IB12	BYTE			
		Input12	%IB13	BYTE			
		Input13	%IB14	BYTE			
		Input14	%IB15	BYTE			
		Input15	%IB16	BYTE			
		Output0	%QB1	BYTE			
		Output1	%QB2	BYTE			

☐ Always update variables

IEC Objects

Variable	Mapping	Type
EtherNet_IP_Adapter		IoDrvCIFXEIPAdapter

Bus cycle options

Bus cycle task: Use parent bus cycle setting

Fig.5-11: EtherNet/IP adapter: EtherNetIP I/O Mapping, subsequent to establishing the connection

Channels: The upper part of the dialog provides the inputs and outputs for communication between scanner and adapter that are specified in the configuration of the connection.

The window is used to assign module inputs and outputs to variables that can be used as (local or) global variables in the individual POUs.

The current value of the variables is displayed in online mode.

This assignment is described in Mapping the Onboard, Inline and Field Bus Inputs and Outputs, page 139,.

IEC objects: When the EtherNet/IP adapter is defined, libraries "IoDrvCIFXEIPAdapter.library" and/or RIL_EtherNetIPAdapter.library are automatically applied. In this way, the memory space required to implement the EtherNet/IP adapter can be defined.

Bus cycle options: By selecting a bus cycle task, the cycle of the mapping exchange for the EtherNet/IP adapter can be connected to a particular task. In this task, it is useful to process the IO data of the adapter as well.

Tab "Field Bus Mapping"

Field bus mapping

Field bus mapping enables access to control variables from a parent field bus master.

The following field busses are supported here:

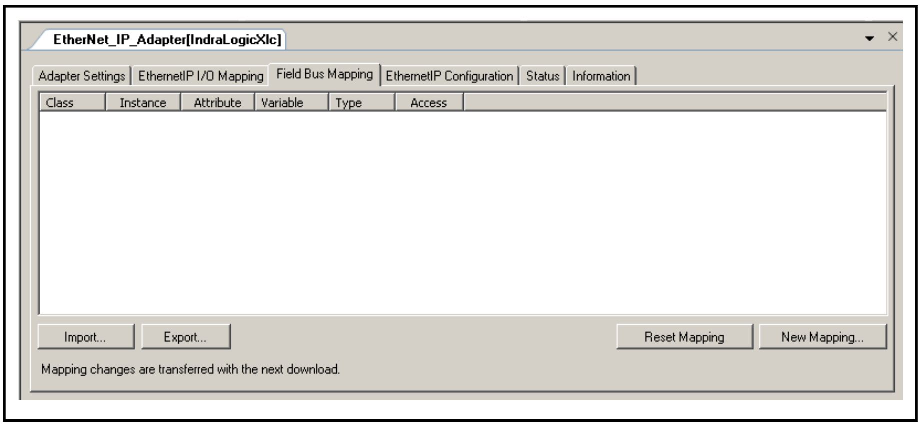
- PROFIBUS DP
- PROFINET IO
- EtherNet/IP

To access a control variable from a parent master, an address must first be assigned to these variables (mapping). The parent master can use this address to access the control data using acyclic services.

To use field bus mapping, the control must be configured accordingly as a bus participant, e.g., as a EtherNet/IP adapter.

configuration

The "Field bus mapping" tab is located in the device editor of the respective device connection. Addresses can be assigned to individual variables here in a table.



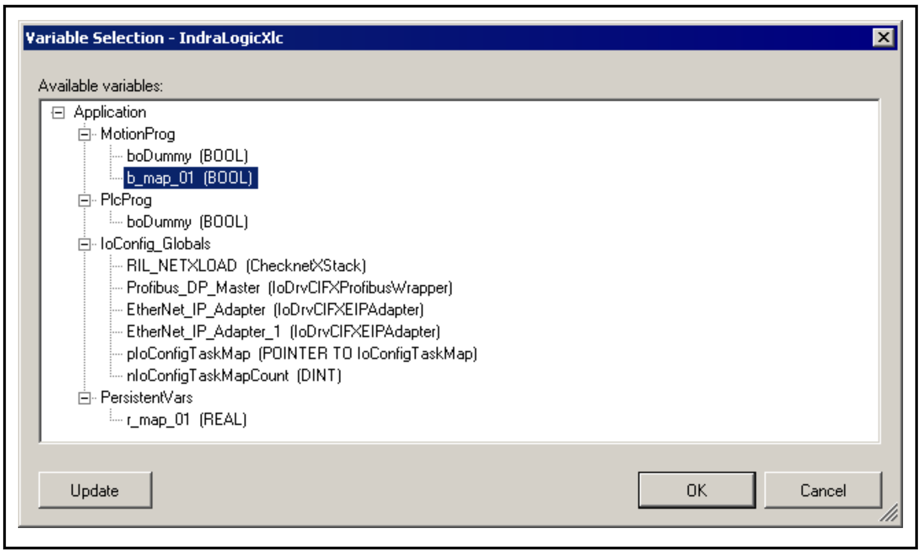
Mapping changes are transferred with the next download.

Fig.5-12: EtherNet/IP adapter: Field bus mapping

Creating new mapping

A new entry can be added using the "New mapping" button. A dialog opens fro selecting a variable.

Variables declared within the application can be selected:



Available variables:

```
Application
  MotionProg
    boDummy (BOOL)
    b_map_01 (BOOL)
  PlcProg
    boDummy (BOOL)
  IoConfig_Globals
    RIL_NETXLOAD (ChecknetXStack)
    Profibus_DP_Master (IoDrvCIFXProfibusWrapper)
    EtherNet_IP_Adapter (IoDrvCIFXEIPAdapter)
    EtherNet_IP_Adapter_1 (IoDrvCIFXEIPAdapter)
    pIoConfigTaskMap (POINTER TO IoConfigTaskMap)
    nIoConfigTaskMapCount (DINT)
  PersistentVars
    r_map_01 (REAL)
```

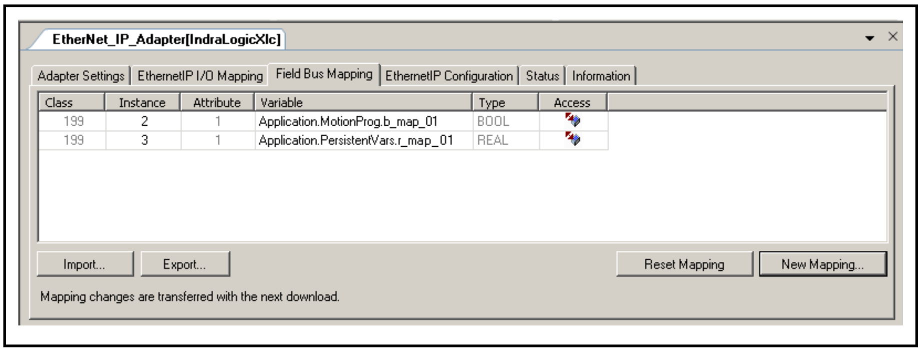
Fig.5-13: EtherNet/IP adapter: field bus mapping, selecting variables

If the selection is confirmed with "OK", a new entry is created in the table for the selected variable.

In addition, the variable is added to the symbol configuration.

The variable section only contains variables that were present at the most recent compilation. If new variables have been added to the PLC program since that time, they are only visible after a new compilation is performed. This can be performed with the "Update" button.

Address, variable and access can be changed afterward by clicking on the corresponding table cell. Gray columns cannot be edited.



Class	Instance	Attribute	Variable	Type	Access
199	2	1	Application.MotionProg.b_map_01	BOOL	⇄
199	3	1	Application.PersistentVars.r_map_01	REAL	⇄

Mapping changes are transferred with the next download.

Fig.5-14: EtherNet/IP adapter: field bus mapping, variables accepted

The address can be specified in the first columns in the table, but the values in the grayed out fields cannot be changed.

For EtherNet/IP, the address consists of class, instance and attribute. In this case, the class is always 199; the instance must be within a range from 2 to 65535; and the attribute is always 1.

The Variable column displays the variables' instance path.

The Type column displays the variables' data type.

The Access column displays the access rights for the variables.

Reset mapping

The "Reset Mapping" button can be used to delete all of the table entries.

Import/Export

To save an existing mapping in a file, use the "Export…" button. It can be read out later using the "Import…" button.

This file is a simple text file.

For this reason, this file can be created manually without a previous export.

The export file contains the name for the field bus "EtherNet IP", "ProfinetIO“, "ProfibusDP" and a table (see program listings).

The address is divided into several columns and consists of the following:

- ProfibusDP: Slot, index,
- ProfinetIO: Slot, Subslot, Index,
- EthernetIP: Class, Instance, Attribute.

This table also includes the variable name, the data type and the access rights of the mapped variables.

```
EthernetIP
# Class Inst Attr Variablename                        Datatyp  Access rights
   199   2    1    Application.MotionProg.b_map_01     BOOL     ReadWrite
   199   3    1    Application.PersistentVars.r_map_01 REAL     ReadWrite
```

Fig.5-15: Excerpt from the export file, separator TAB:

Tab "EtherNetIP Configuration"

The "EtherNetIP Configuration" tab displays the communication parameters of the EtherNet/IP adapter.

Some of the parameters (in black) can be changed at this point.

This window is used for service purposes is only visible if the option "Show generic configuration editors" is enabled under **Tools ▸ Options ▸ IndraLogic 2G ▸ Device editor**.

☞ Please clarify any other modifications that might possibly be made in the parameters that can be edited with the Service team.

Tab "Status"

The "Status" tab in the "EtherNet/IP adapter device editor" displays status information (e.g., "Running", "Stopped") and specific diagnostic messages from the device. In addition, the diagnostic messages contained in the status flags are displayed in "Diag".

Tab "Information"

The window displays some general information from the device description file:

Name, Vendor, Categories, Version, Order number, Description, Image, if available.

5.3 Configuring the EtherNet/IP Adapter (Engineering)

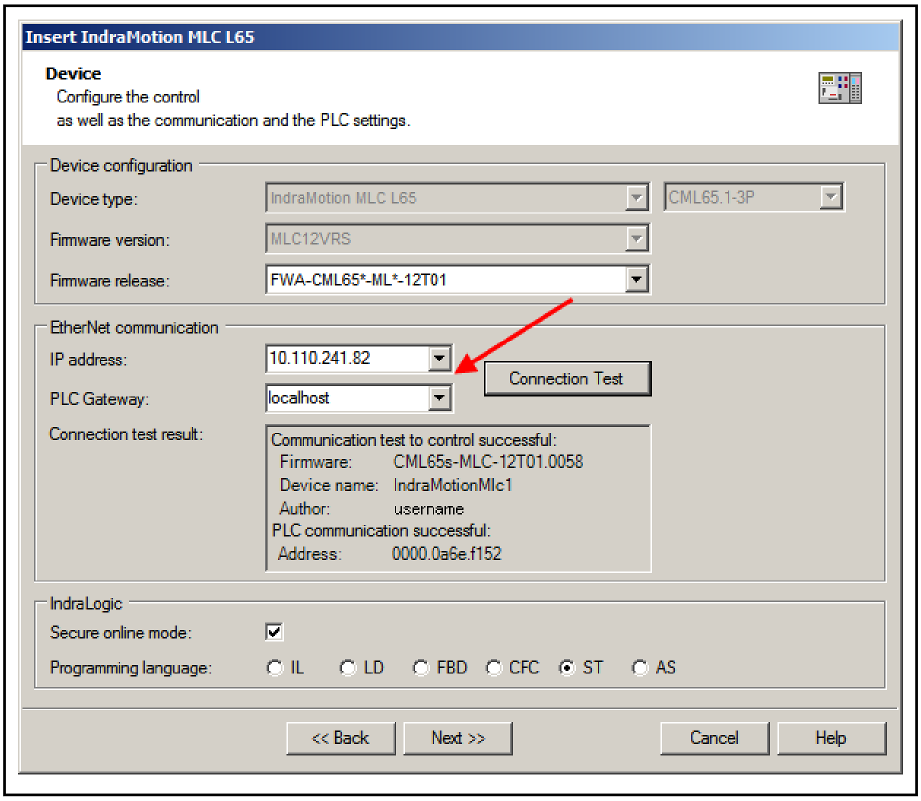
5.3.1 Configuring the EtherNet/IP Adapter (Engineering), Overview

EtherNet adapters provide dialogs for configuring an EtherNet link ("EtherNet Adapter") for a TCP/IP network using the engineering software of the control.

The following controls allow the implementation of the EtherNet/IP adapter (Engineering):

- IndraLogic XLC L25/L45/L65
- IndraMotion MLC L25/L45/L65

When the control is created in the Project Explorer, the respective EtherNet Engineering interface available at the control is configured such that it assumes the double function of engineering software and EtherNetI/P adapter interface. The IP address / gateway setting is accepted. The required port configuration is made without any action on the user's part.



Insert IndraMotion MLC L65

Device

Configure the control
as well as the communication and the PLC settings.

Device configuration

Device type: IndraMotion MLC L65 CML65.1-3P

Firmware version: MLC12VRS

Firmware release: FWA-CML65*-ML*-12T01

EtherNet communication

IP address: 10.110.241.82

PLC Gateway: localhost

Connection test result:

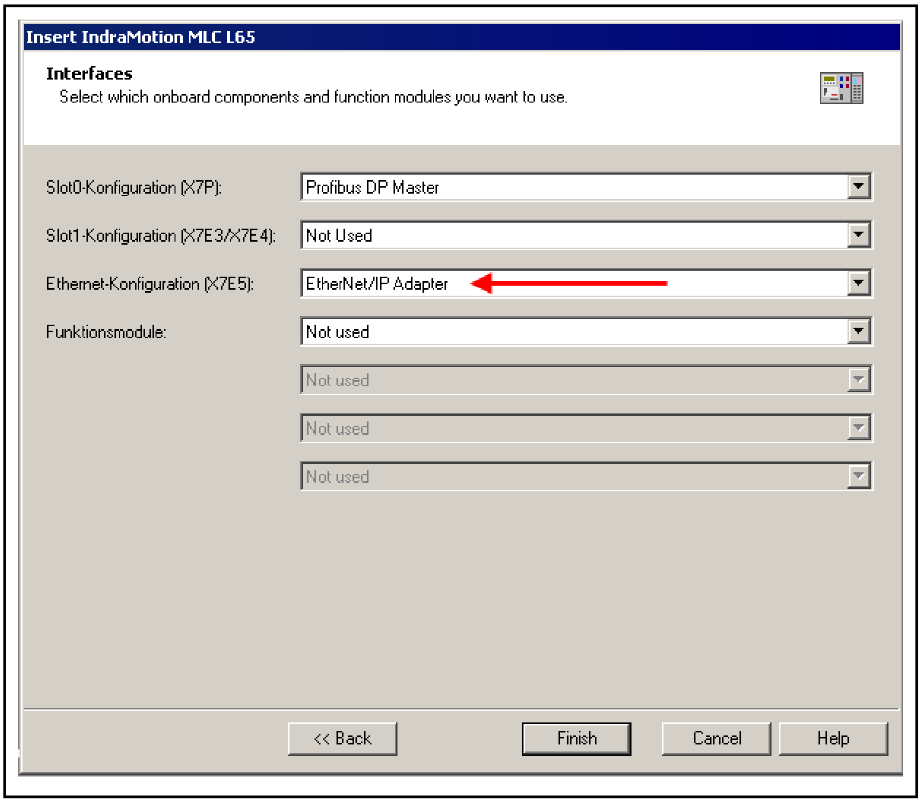
Communication test to control successful:
Firmware: CML65s-MLC-12T01.0058
Device name: IndraMotionMlc1
Author: username
PLC communication successful:
Address: 0000.0a6e.f152

IndraLogic

Secure online mode: ☑

Programming language: ○ IL ○ LD ○ FBD ○ CFC ● ST ○ AS

Fig.5-16: Dialog: configuration of the control, IP address / PLC gateway



Insert IndraMotion MLC L65

Interfaces

Select which onboard components and function modules you want to use.

Slot0-Konfiguration (X7P): Profibus DP Master

Slot1-Konfiguration (X7E3/X7E4): Not Used

Ethernet-Konfiguration (X7E5): EtherNet/IP Adapter

Funktionsmodule: Not used

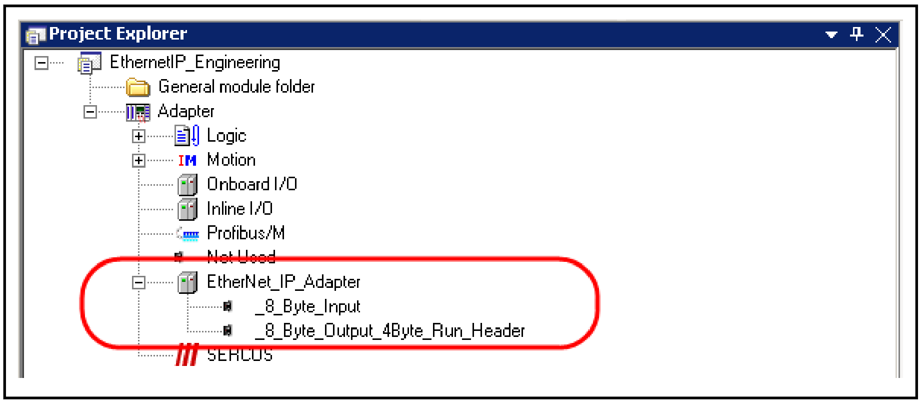
Not used

Not used

Not used

Fig.5-17: Dialog: Configuring the control and enabling the EtherNet/IP adapter (Engineering) functionality

After the control configuration is successfully enabled and complete, the preconfigured folder appears in the Project Explorer.



Project Explorer

- EthernetIP_Engineering
  - General module folder
  - Adapter
    - Logic
    - Motion
    - Onboard I/O
    - Inline I/O
    - Profibus/M
    - Not Used
    - EtherNet_IP_Adapter
      - _8_Byte_Input
      - _8_Byte_Output_4Byte_Run_Header
    - SERCOS

Fig.5-18: EtherNet/IP adapter (Engineering)

The interface between an EtherNet/IP adapter (Engineering) and the EtherNet/IP scanner is executed as common memory space. This space is preconfigured with 8 bytes inputs and 8 bytes outputs plus a 4 byte header.

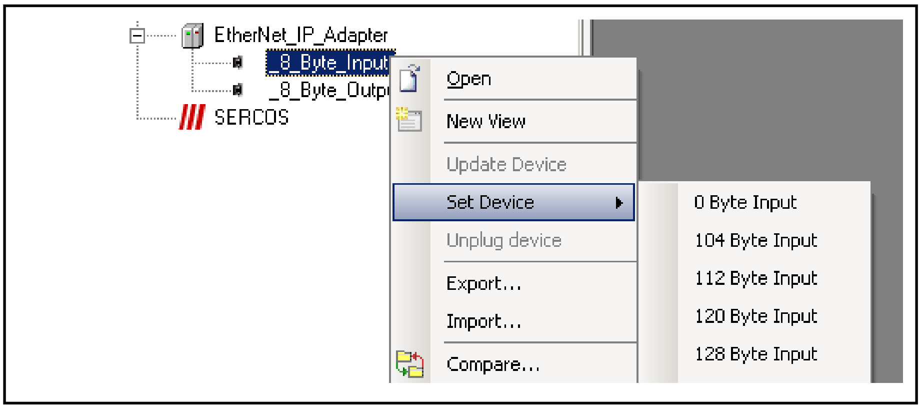


Fig.5-19: EtherNet/IP adapter (Engineering) , changing the size of the coupling area

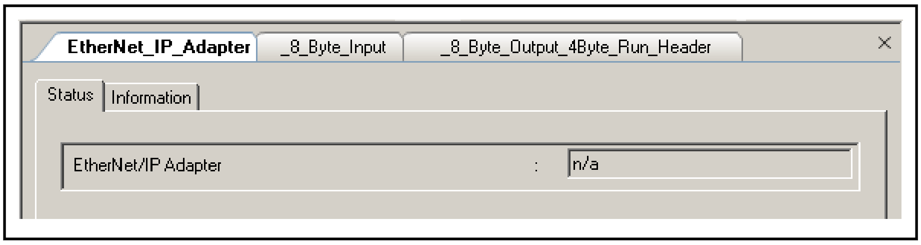
☞

1. Inputs on the EtherNet/IP scanner side become outputs on the EtherNet/IP adapter side and vice versa.
2. If the input and output ranges are set to 0 bytes, cyclic communication is not possible.

EtherNet/IP adapter (Engineering)

- *EtherNet/IP adapter (Engineering), object*
  - Status, page 103,
  - Information, page 103.
- *EtherNet/IP adapter (Engineering), modules*
  - EtherNet/IP I/O mapping, page 103,
  - Information, page 104.

5.3.2 Tab "Status" of the Adapter



EtherNet/IP Adapter : n/a

Fig.5-20: EtherNet/IP adapter (Engineering): adapter status

In online mode, the tab displays status information from the control (e.g., "Running", "Not running (n/a)").

5.3.3 Tab "Information" of the Adapter

The window displays some general information from the device description file:

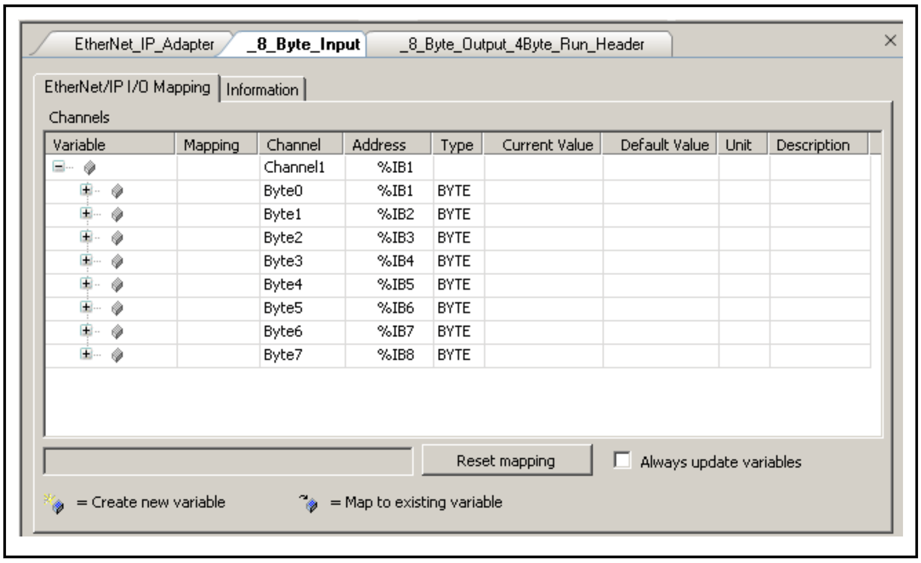
Name, Vendor, Categories, Version, Order number, Description, Image, if available.

5.3.4 Tab "EtherNet/IP I/O Mapping" of the Modules

The window is used to assign module inputs and outputs to variables that can be used as local or global variables in the individual POUs.

The current value of the variables is displayed in online mode.

This assignment is described in Mapping the Onboard, Inline and Field Bus Inputs and Outputs, page 139,.



Variable	Mapping	Channel	Address	Type	Current Value	Default Value	Unit	Description
		Channel1	%IB1					
		Byte0	%IB1	BYTE				
		Byte1	%IB2	BYTE				
		Byte2	%IB3	BYTE				
		Byte3	%IB4	BYTE				
		Byte4	%IB5	BYTE				
		Byte5	%IB6	BYTE				
		Byte6	%IB7	BYTE				
		Byte7	%IB8	BYTE				

Fig.5-21: EtherNet/IP adapter (Engineering): I/O mapping

Reset mapping: Deletes the assignment made in the editor.

Always update variables: If this option is enabled, all variables are updated in each cycle (see bus cycle options, bus cycle task of the control), no matter whether they are used or not and whether they are mapped to an input or an output channel.

5.3.5 Tab "Information" of the Modules

The window displays some general information from the device description file:

Name, Vendor, Categories, Version, Order number, Description, Image, if available.

6 sercos III I/O

6.1 Terms and Abbreviations

Master	The sercos III field bus master is called **master**.
Slave	The sercos III field bus slaves are called **slaves**.
sercos address	The criterion for addressing a slave is the sercos address.
SDDML file	This is the device description with which the sercos III slaves (typically with an I/O profile) are made known to IndraWorks.

6.2 sercos III I/O Master Features

The current implementation of the sercos III I/O master contains the following functionalities:

- Cyclic data traffic
- SDDML file import into the device database
- Bus scan of sercos III slaves
- Remote address allocation
- Automatic configuration of the devices in the device tree (offline/online adjustment of the devices in the project)
- sercos III slave related connection state in the interface
- Diagnostics for the sercos III master and slaves via FB
- Acyclic data transfer (service channel or IP channel) via FB

Function/Characteristic	Value
Max. number at I/O slaves, IndraLogic XLC/ IndraMotion MLC	32
Max. number at I/O slaves, IndraMotion MTX	64
Baud rate	100 Mbit/s
Auto negotiation/ autocrossing	Yes
Min. cycle time	L65: 250 µs L45: 500 µs L25, VEP: 1 ms
Max. amount of cyclic input data	1500 bytes
Max. amount of cyclic output data	1500 bytes
Max. amount of cyclic input data per slave	1500 bytes
Max. amount of cyclic output data per slave	1500 bytes
Max. amount of modules per slave	61
Max. acyclic telegram data per slave / telegram (MTU)	1500 bytes

Fig.6-1: Technical data

6.3 Configuring sercos III I/Os

6.3.1 Configuring sercos III I/Os, Overview

sercos III is an IEC-compliant, open system universal bus for EtherNet-based real-time communication. As a universal bus, sercos III has communication channels and device profiles for all established automation applications.

sercos III IOs can be used in all controls that have the sercos object (onboard) node (function of a sercos master).

The master can be extended with slaves and modules so that they can be configured later, depending on the device description file.

sercos III master

- Bus diagnostics, page 106.

sercos III I/O slave

- sercos III slave, page 111,
- sercos III configuration, page 112, this window is used for service purposes and is only visible if the option "Show generic configuration editors" is enabled under **Tools ▸ Options ▸ IndraLogic 2G ▸ Device editor.**
- Status, page 113
- Information, page 113

sercos III I/O modules

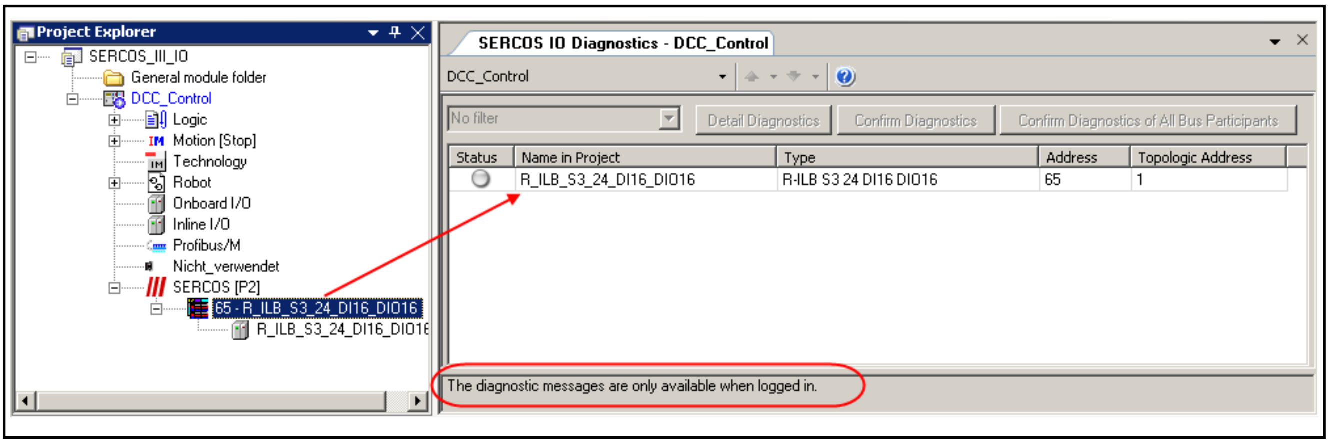
- sercos III module, page 115,
- Function groups, page 115,
- Special tab User-defined parameters, page 116,
- sercos III module configuration, page 117, this window is used for service purposes and is only visible if the option "Show generic configuration editors" is enabled under **Tools ▸ Options ▸ IndraLogic 2G ▸ Device editor.**
- sercos III modules I/O mapping, page 117,
- Information, page 118

6.3.2 sercos III Master

Based on the object node of the master (here: sercos III master), a uniform diagnostics concept has been developed for the field buses of the IndraLogic XLC and IndraMotion MLC / MTX systems.

Double-click on the sercos III master node in the Project Navigator to open the diagnostic window for the sercos III bus.

The "Field bus Diagnostics" tab shows the modules applied to the field bus node in online mode.



Status	Name in Project	Type	Address	Topologic Address
○	R_ILB_S3_24_DI16_DIO16	R-ILB S3 24 DI16 DIO16	65	1

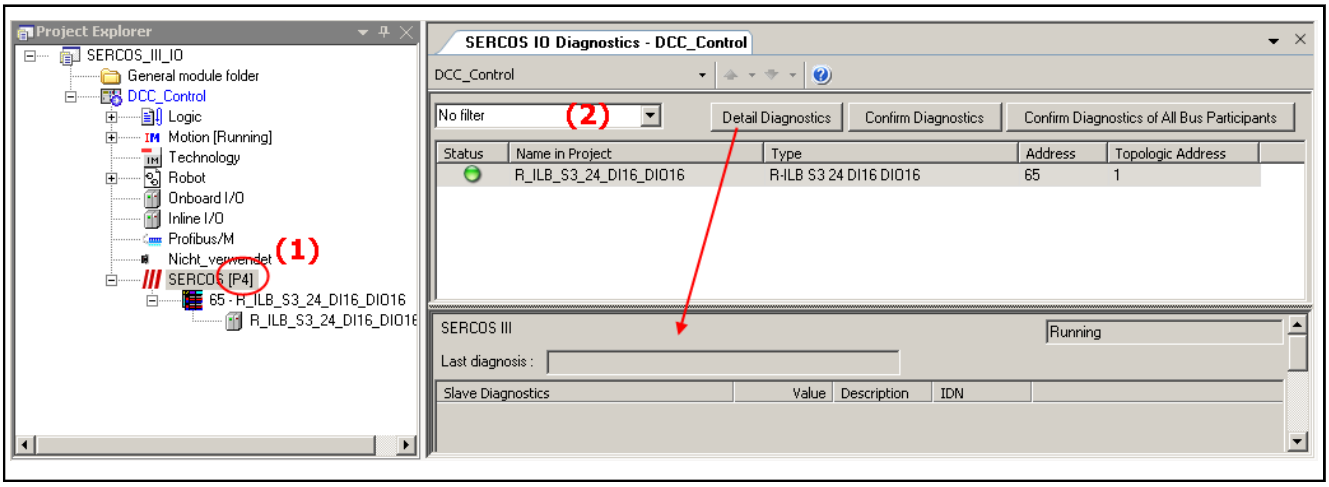
The diagnostic messages are only available when logged in.

Fig.6-2: Field bus diagnostics, control not logged in

The terminals that are disabled and excluded from the diagnostics by selecting **Properties ▸ Compile ▸ Exclude from compilation**.

After login, the bus available at the node runs through a diagnostics cycle. The status LEDs change their color in the result of this cycle (here: green), i.e., the module signals an error-free run.

After having been selected, a module (click on the line) is available for detailed diagnostics.



Status	Name in Project	Type	Address	Topologic Address
●	R_ILB_S3_24_DI16_DIO16	R-ILB S3 24 DI16 DIO16	65	1

SERCOS III: Running

Last diagnosis :

Slave Diagnostics	Value	Description	IDN

(1) sercos state [P0], [P2], [P4]
(2) Filter: only modules without errors / only modules with error
Fig.6-3: Field bus diagnostics, online control, logged in, error-free

After the login, the control has to be switched to online mode, the sercos bus switches to parameterization mode [P2]. To ensure that the transmission can be completed without errors, the sercos state has to be switched from [P2] to [P4] (context menu of the sercos III master **sercos state**).

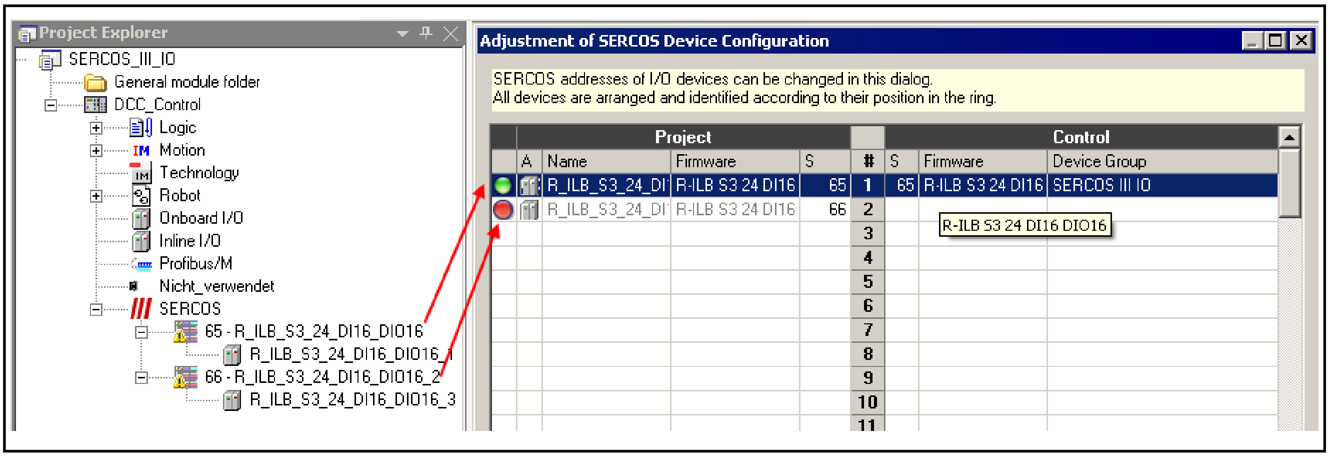
Bus errors

Login is repeated with an additional coupler not available at the real control.

The status LEDs turn red, thus indicating a bus problem.

When trying to go online, the dialog for adjusting the sercos device configuration opens, where the error is displayed.

sercos III I/O

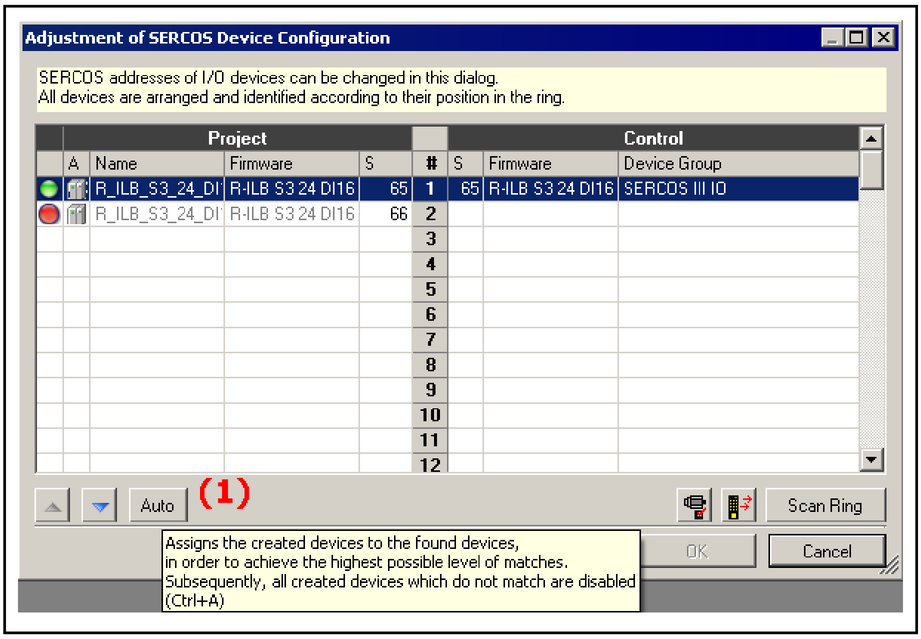


Adjustment of SERCOS Device Configuration

SERCOS addresses of I/O devices can be changed in this dialog.
All devices are arranged and identified according to their position in the ring.

	Project					Control		
	A	Name	Firmware	S	#	S	Firmware	Device Group
🟢		R_ILB_S3_24_DI	R-ILB S3 24 DI16	65	1	65	R-ILB S3 24 DI16	SERCOS III IO
🔴		R_ILB_S3_24_DI	R-ILB S3 24 DI16	66	2			
					3			
					4			
					5			
					6			
					7			
					8			
					9			
					10			
					11			

Fig.6-4: Adjustment dialog with display of the additional coupler



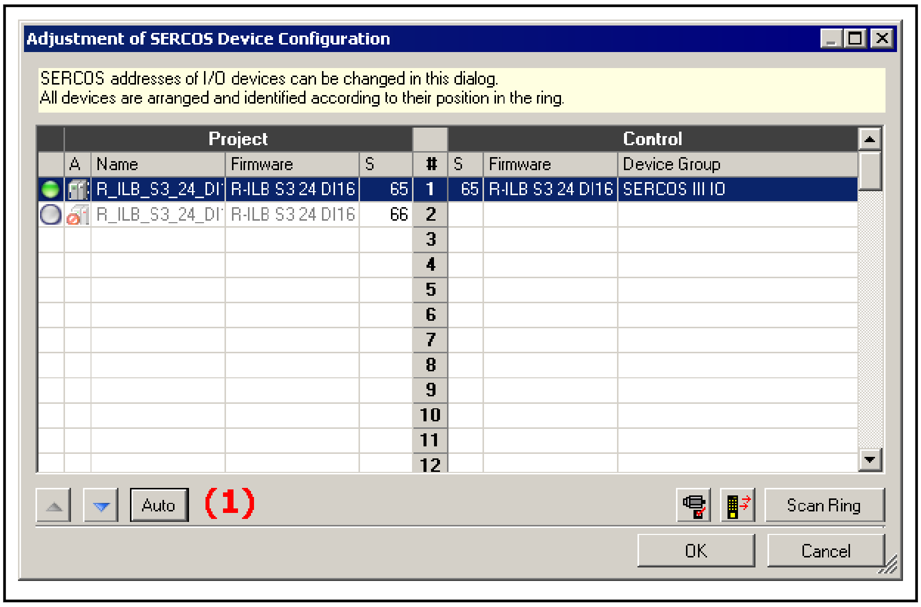
Adjustment of SERCOS Device Configuration

SERCOS addresses of I/O devices can be changed in this dialog.
All devices are arranged and identified according to their position in the ring.

	Project					Control		
	A	Name	Firmware	S	#	S	Firmware	Device Group
🟢		R_ILB_S3_24_DI	R-ILB S3 24 DI16	65	1	65	R-ILB S3 24 DI16	SERCOS III IO
🔴		R_ILB_S3_24_DI	R-ILB S3 24 DI16	66	2			
					3			
					4			
					5			
					6			
					7			
					8			
					9			
					10			
					11			
					12			

Assigns the created devices to the found devices,
in order to achieve the highest possible level of matches.
Subsequently, all created devices which do not match are disabled
(Ctrl+A)

Fig.6-5: Adjustment dialog with "Auto" (1) allowing deactivation of the bus coupler

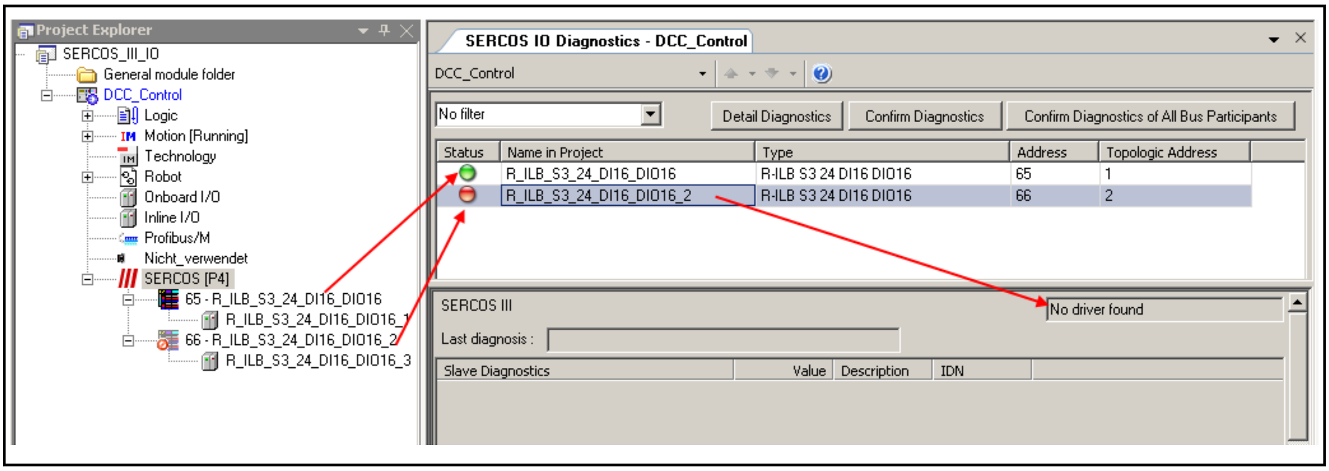


Adjustment of SERCOS Device Configuration

SERCOS addresses of I/O devices can be changed in this dialog.
All devices are arranged and identified according to their position in the ring.

Project				#	Control		
A	Name	Firmware	S	#	S	Firmware	Device Group
●	R_ILB_S3_24_DI	R-ILB S3 24 DI16	65	1	65	R-ILB S3 24 DI16	SERCOS III IO
○	R_ILB_S3_24_DI	R-ILB S3 24 DI16	66	2			
				3			
				4			
				5			
				6			
				7			
				8			
				9			
				10			
				11			
				12			

Fig.6-6: Adjustment dialog, bus coupler deactivated



Status	Name in Project	Type	Address	Topologic Address
●	R_ILB_S3_24_DI16_DIO16	R-ILB S3 24 DI16 DIO16	65	1
●	R_ILB_S3_24_DI16_DIO16_2	R-ILB S3 24 DI16 DIO16	66	2

SERCOS III
No driver found
Last diagnosis :

Slave Diagnostics	Value	Description	IDN

Fig.6-7: sercos III bus online in [P4] with deactivated module

Confirm Diagnostics: Although there is an error, both the warning triangle and the red status LED are turned off for the selected module. The "Extended diagnostics" of the "Detail Diagnostics" remains in the text message. (The next diagnostic message can be processed...)

Confirm Diagnostics of All Bus Participants: Although there are errors, both the warning triangle and the red status LED are turned off for all modules.

6.3.3 Adding a Slave

The slaves are located in the "Periphery" library in the "sercos III" folder.

Drag and drop the required slaves from the library to the sercos object.

In Project Explorer, slaves can also be added between existing slaves in this way.

If a required slave is not contained in the library by default, it can be integrated into the library by importing its device description file using the main menu **Tools ▸ Device database...**.

sercos III I/O

Slaves for connecting I/O modules

The sercos III I/O differentiates two types of slaves for connecting I/O modules:

1. **Compact**:

   For compact slaves, the module structure is specified.

   After a slave has been added in Project Explorer, for compact slaves the modules below the slave object node are already present in their complete form. The modules are not visible in the library.

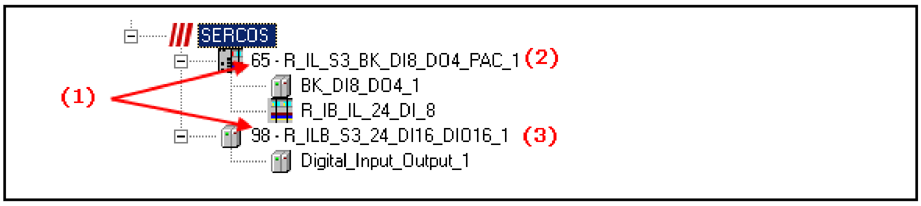
2. **Modular**:

   The module structure of the slave is variable.

   In addition to a fixed portion, in the figure below "BK_DI8_DO4_1" with 8 digital inputs and 4 digital outputs, further modules can be added.

   The modules can be arranged as desired, but according to the fitting specification.

   To add modules, see Adding modules, page 114.

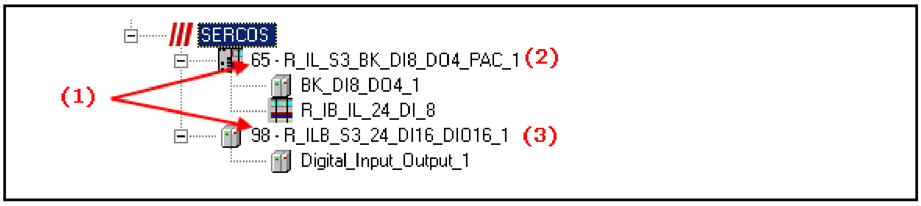


(1) Current bus addresses for the slaves
(2) Modular slave with assigned modules
(3) Compact slave

Fig.6-8: Slaves with I/O modules on the sercos III master object

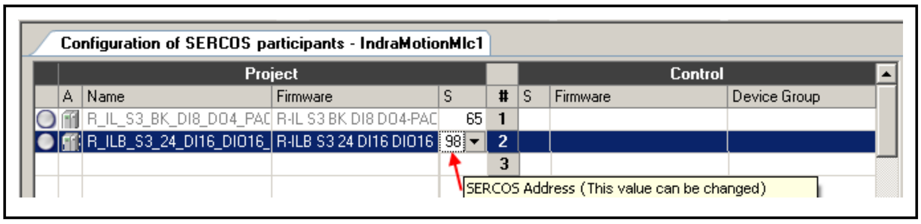
Overview of sercos addresses...

To show the complete address assignment of all bus devices, open the context menu item **sercos Device Configuration...** of the sercos master object.



(1) Current bus addresses for the slaves
(2) Modular slave with assigned modules
(3) Compact slave

Fig.6-9: Slaves with I/O modules on the sercos III master object



Configuration of SERCOS participants - IndraMotionMlc1

	Project					Control		
	A	Name	Firmware	S	#	S	Firmware	Device Group
○		R_IL_S3_BK_DI8_DO4_PAC	R-IL S3 BK DI8 DO4-PAC	65	1			
○		R_ILB_S3_24_DI16_DIO16_	R-ILB S3 24 DI16 DIO16	98	2			
					3			

SERCOS Address (This value can be changed)

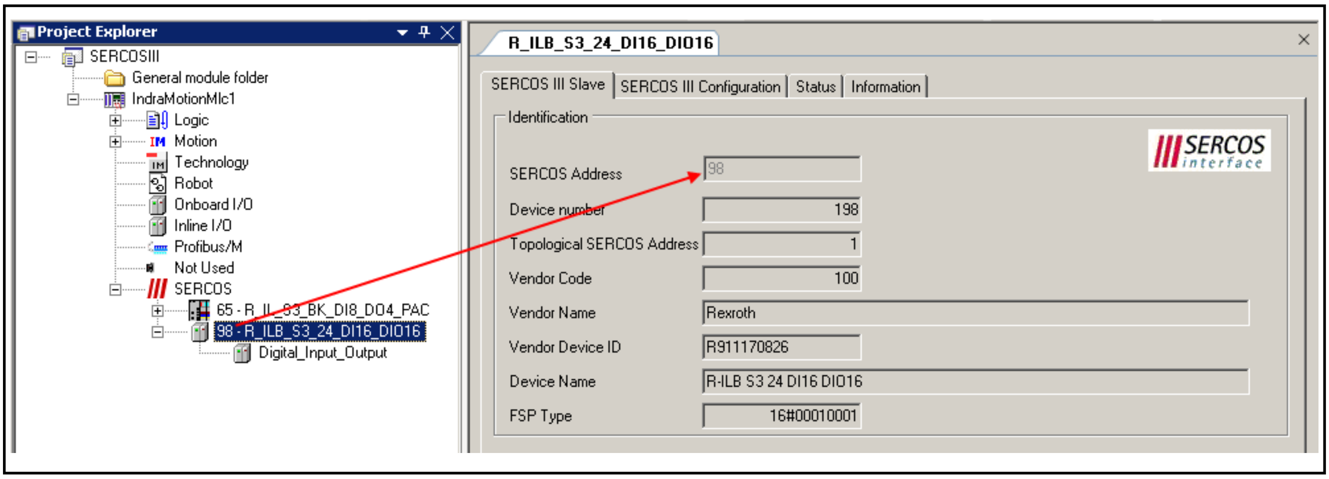
Fig.6-10: Overview of sercos addresses

6.3.4 sercos III I/O Slave

General

To open the editing window in the Project Explorer, double click on the sercos III object.

The dialogs will inform you about the configuration of the entire slave and you can modify it if necessary.



SERCOS Address: 98
Device number: 198
Topological SERCOS Address: 1
Vendor Code: 100
Vendor Name: Rexroth
Vendor Device ID: R911170826
Device Name: R-ILB S3 24 DI16 DIO16
FSP Type: 16#00010001

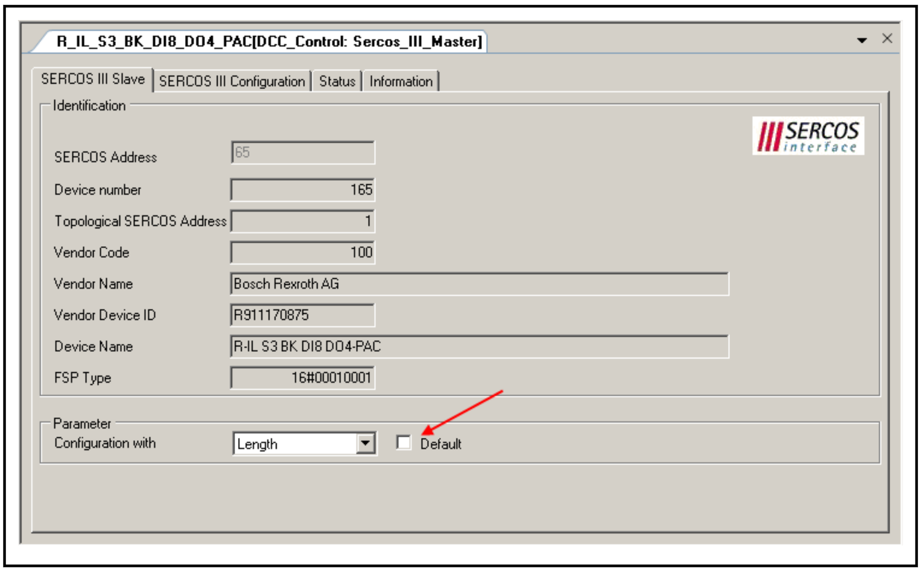
Fig.6-11: Configuring a sercos III slave object

Tab (sercos III I/O slave)

- sercos III slave, page 111,
- sercos III configuration, page 112, this window is used for service purposes and is only visible if the option "Show generic configuration editors" is enabled under **Tools ▸ Options ▸ IndraLogic 2G ▸ Device editor**.
- Status, page 113
- Information, page 113

Tab "sercos III Slave"

All settings in this dialog are dependent on the device description in terms of whether they can be edited here and which values are specified or possible.



SERCOS Address: 65
Device number: 165
Topological SERCOS Address: 1
Vendor Code: 100
Vendor Name: Bosch Rexroth AG
Vendor Device ID: R911170875
Device Name: R-IL S3 BK DI8 DO4-PAC
FSP Type: 16#00010001
Configuration with: Length
□ Default

Fig.6-12: sercos III slave: sercos III slave

Identification

In the "Identification" section you will find specific information regarding the sercos III slave, which uniquely identifies the slave.

sercos address:	Bus-specific address of the sercos III slave
Device number:	Logical address of the sercos III slave

sercos III I/O

Topological sercos address:	Topological address of the sercos III slave List position in "Configuration of sercos participants", column "#"
Vendor code (Vendor Code):	Vendor number of the sercos III slave
Vendor Name (Vendor Name):	Vendor name of the sercos III slave, e.g., Rexroth
Device ID (Vendor Device ID):	Identification number for the sercos III slave specified by the vendor
Device name:	Description of the sercos III slave by the vendor
FSP Type:	This number defines the device-specific functions of the sercos III slave, e.g., device properties.

Parameter

The "Parameter" panel provides setting options affecting the configuration or the behavior of the slaves.

These are settings which should only be made by sercos experts.

The settings can be changed when the checkbox (Default) is unticked.

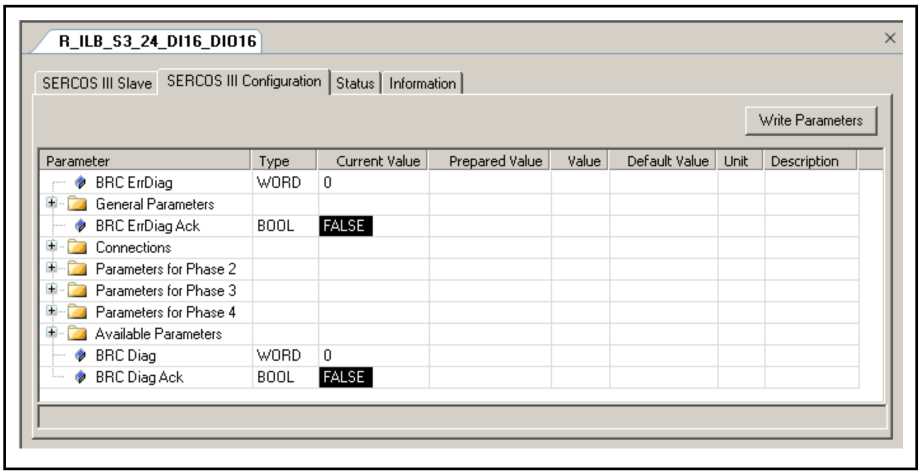
Configuration with: Select whether the sercos connection of a slave is achieved via Length or via IDN.

This option is only available for devices which support the I/O profile and a variable connection configuration.

Tab "sercos III Configuration"

This window is used for service purposes is only visible if the option "Show generic configuration editors" is enabled under **Tools** ▸ **Options** ▸ **IndraLogic 2G** ▸ **Device editor**.

☞ Please clarify any possible modifications to the parameters that can be edited with the service team.

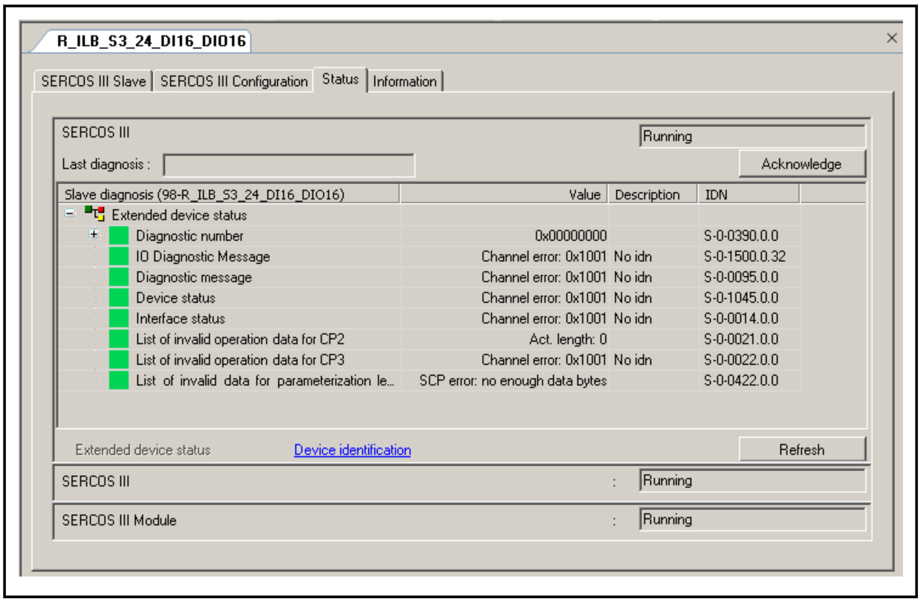


Parameter	Type	Current Value	Prepared Value	Value	Default Value	Unit	Description
BRC ErrDiag	WORD	0					
+ General Parameters							
BRC ErrDiag Ack	BOOL	FALSE					
+ Connections							
+ Parameters for Phase 2							
+ Parameters for Phase 3							
+ Parameters for Phase 4							
+ Available Parameters							
BRC Diag	WORD	0					
BRC Diag Ack	BOOL	FALSE					

Fig.6-13: sercos III slave: sercos III configuration (online)

☞ When the bus is running, modified parameters can be transferred by clicking button **Write parameter**.

Tab "Status"



SERCOS III: Running

Last diagnosis :

Slave diagnosis (98-R_ILB_S3_24_DI16_DIO16)	Value	Description	IDN
Extended device status			
🟩 Diagnostic number	0x00000000		S-0-0390.0.0
🟩 IO Diagnostic Message	Channel error: 0x1001	No idn	S-0-1500.0.32
🟩 Diagnostic message	Channel error: 0x1001	No idn	S-0-0095.0.0
🟩 Device status	Channel error: 0x1001	No idn	S-0-1045.0.0
🟩 Interface status	Channel error: 0x1001	No idn	S-0-0014.0.0
🟩 List of invalid operation data for CP2	Act. length: 0		S-0-0021.0.0
🟩 List of invalid operation data for CP3	Channel error: 0x1001	No idn	S-0-0022.0.0
🟩 List of invalid data for parameterization le...	SCP error: no enough data bytes		S-0-0422.0.0

Extended device status

SERCOS III : Running

SERCOS III Module : Running

Fig.6-14: sercos III slave: Status (online)

The "Status" tab displays status information (e.g. "Running" (bus active) and "n/a" (no information available)) and specific diagnostic messages from the respective device and regarding the card used and the internal bus system.

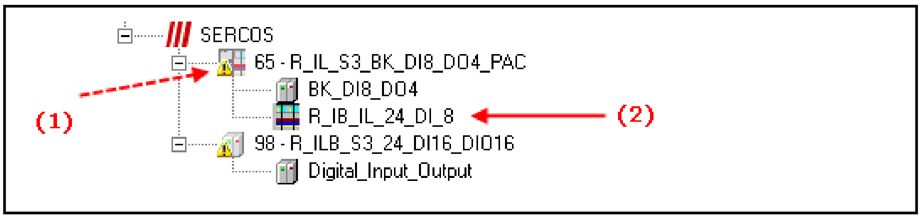
☞ Modifications in the bus are only offline

Prerequisites for the sercos III bus to go online:

1. Configuration of the sercos participants in the project and control match with respect to type, sequence and sercos address.
2. All modules run without errors.

   This can be determined for the actual slave in the "Status" window by clicking on "Device identification" and "Extended device identification".
3. sercos state: P4

Slaves for which there are errors (starting with the first one) appear in the project tree with a yellow warning triangle and an error tool tip.



(1) Slaves with errors
(2) Module is only in the project, not on the control
Fig.6-15: sercos III bus, error message in the Project Explorer

Tab "Information"

The window displays some general information from the device description file:

Name, Vendor, Categories, Version, Order number, Description, Image, if available.

6.3.5 Adding Modules to the Slave

The modules that work with the respective slave are located in the "Peripherals" library in the "sercos III" folder under the respective slave.

> ☞ I/O modules can only be added in modular structured slaves, page 110,.

Drag the required modules out of the library into the slave object.

New modules can also be added between existing modules in Project Explorer.

Optionally, modules can be added in the context menu via **Add** ▸ **Slave** ▸ ... of the slave.

In this case, the new module is added as the last module under the slave.

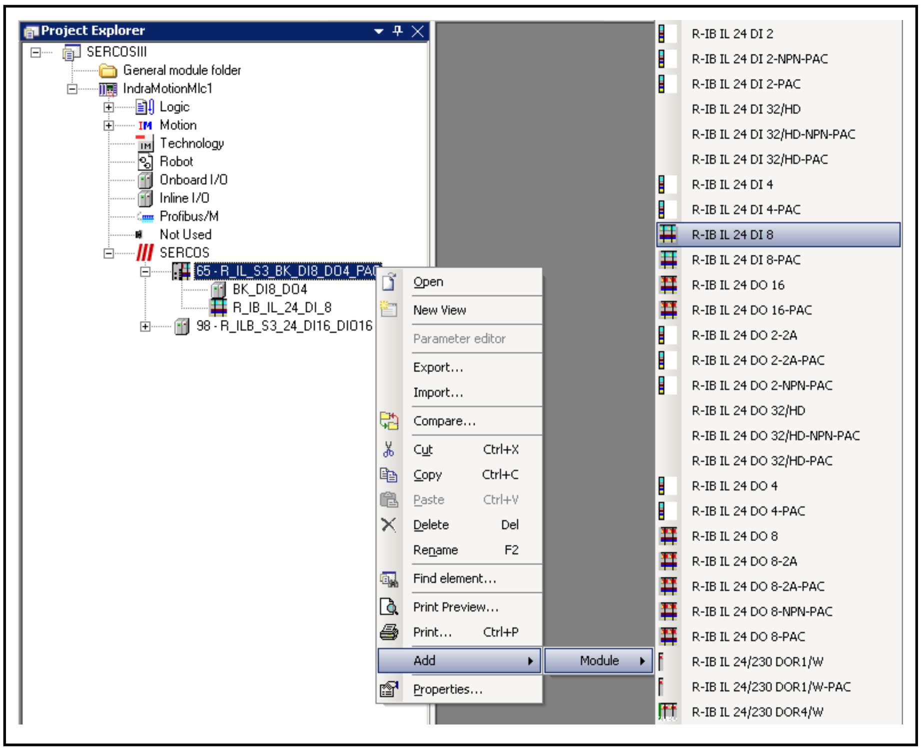


Fig.6-16: sercos III, adding modules to the slave

6.3.6 sercos III I/O Modules

General

To open the editing window, in the Project Explorer, double click on the desired module.

The dialogs will inform you about the configuration of the module and you can modify it if necessary.

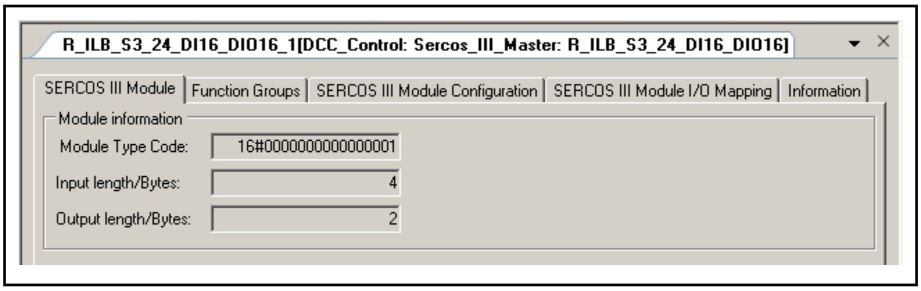
The settings in all tabs of the dialog are specified by the module's device description file. In this file, the value with which the setting is preset is determined and whether or not it can be edited.

Tab (sercos III I/O modules)

- sercos III module, page 115,

- Function groups, page 115,
- Special tab User-defined parameters, page 116,
- sercos III module configuration, page 117, this window is used for service purposes and is only visible if the option "Show generic configuration editors" is enabled under **Tools ▸ Options ▸ IndraLogic 2G ▸ Device editor.**
- sercos III modules I/O mapping, page 117,
- Information, page 118,

Tab "sercos III Module"


R_ILB_S3_24_DI16_DIO16_1[DCC_Control: Sercos_III_Master: R_ILB_S3_24_DI16_DIO16]
SERCOS III Module
Function Groups
SERCOS III Module Configuration
SERCOS III Module I/O Mapping
Information

Module information

Module Type Code: 16#0000000000000001

Input length/Bytes: 4

Output length/Bytes: 2

Fig.6-17: sercos III module: sercos III module

Module information

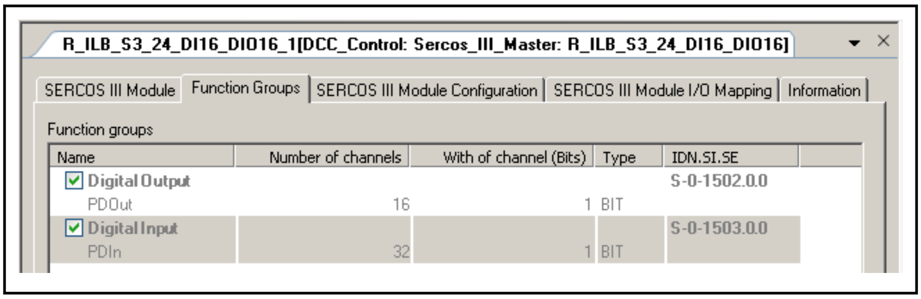
Module type code: Each module has a unique module type identification from the respective vendor. Depending on the vendor, this number can be a hexadecimal number or an order number.

Input length/Bytes: Specifies the input length of the module in bytes

Output length/Bytes: Specifies the output length of the module in bytes

Tab "Function Groups"

The "Function Groups" tab provides information on the inputs and outputs of the function group. None of the settings in this dialog can be edited.


R_ILB_S3_24_DI16_DIO16_1[DCC_Control: Sercos_III_Master: R_ILB_S3_24_DI16_DIO16]
SERCOS III Module
Function Groups
SERCOS III Module Configuration
SERCOS III Module I/O Mapping
Information

Function groups

Name	Number of channels	With of channel (Bits)	Type	IDN.SI.SE
☑ Digital Output				S-0-1502.0.0
PDOut	16	1	BIT	
☑ Digital Input				S-0-1503.0.0
PDIn	32	1	BIT	

Fig.6-18: sercos III module: Function Groups

Name: Name of the channel, cannot be edited.

Number of channels: Number of supported channels for this module.

Channel width (bits): Bit size of the individual channels, cannot be edited

Type: Number of structure elements (SE) of the I/O function groups. Depends on the I/O functions

IDN.SI.SE:

- IDN Identification number, e.g. 15xx

- SI (Structure Instance) The SI number is identical with the slide-in number of the module.

  Module 1 has SI number 1

  For fixed modules, module 1 has SI number 0
- SE (Structure Element) Number of structure elements (SE) of the I/O function groups. Depends on the I/O functions

☞ Function groups (in case of I/O devices), drive inputs and outputs (in case of drives):

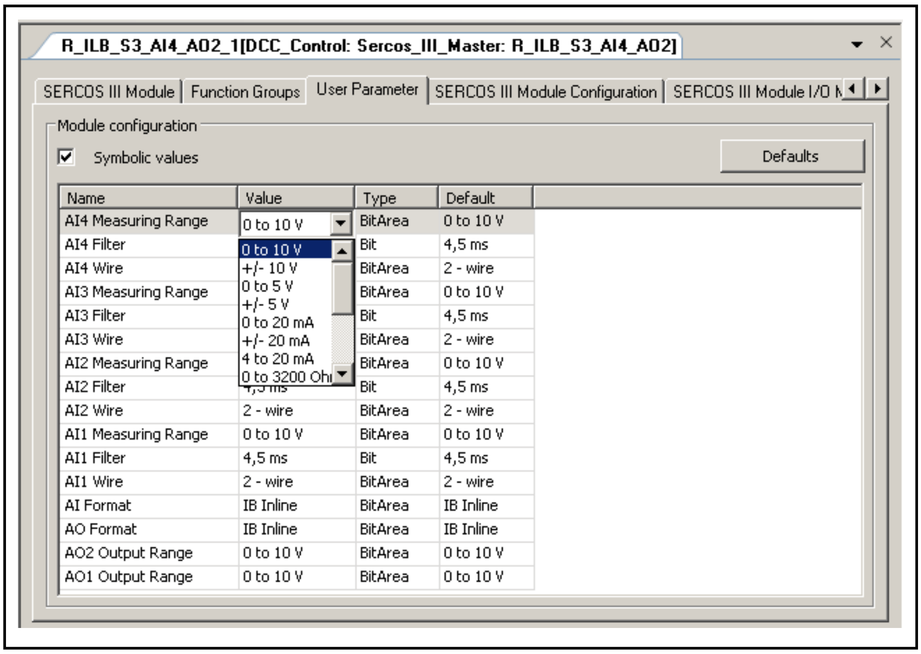
The "Drive inputs and outputs" tab shows the sercos parameters which are configured in the cyclic connection of a drive.

The "Add" and "Edit" buttons open a dialog which allows adding parameters to the configuration or editing parameters.

The "Input configuration" or "Output configuration" dialog can be used to specify the parameters which are to be added to the configuration. By confirming the selection with OK, the parameter is added to the input or output configuration.

Special Tab "User Parameters"

This tab is used for presetting parameters of the module and is only displayed for those modules which require or allow this presetting.



Module configuration

☑ Symbolic values

Name	Value	Type	Default
AI4 Measuring Range	0 to 10 V	BitArea	0 to 10 V
AI4 Filter		Bit	4,5 ms
AI4 Wire		BitArea	2 - wire
AI3 Measuring Range		BitArea	0 to 10 V
AI3 Filter		Bit	4,5 ms
AI3 Wire		BitArea	2 - wire
AI2 Measuring Range		BitArea	0 to 10 V
AI2 Filter	4,5 ms	Bit	4,5 ms
AI2 Wire	2 - wire	BitArea	2 - wire
AI1 Measuring Range	0 to 10 V	BitArea	0 to 10 V
AI1 Filter	4,5 ms	Bit	4,5 ms
AI1 Wire	2 - wire	BitArea	2 - wire
AI Format	IB Inline	BitArea	IB Inline
AO Format	IB Inline	BitArea	IB Inline
AO2 Output Range	0 to 10 V	BitArea	0 to 10 V
AO1 Output Range	0 to 10 V	BitArea	0 to 10 V

Fig.6-19: sercos III module: User-defined Parameters

Module configuration

The example shows a module with four analog inputs and two analog outputs.

The value range in which the input information is to be expected is selected for analog input AI4.

A selection window based on the enumeration data type is offered for each "Value".

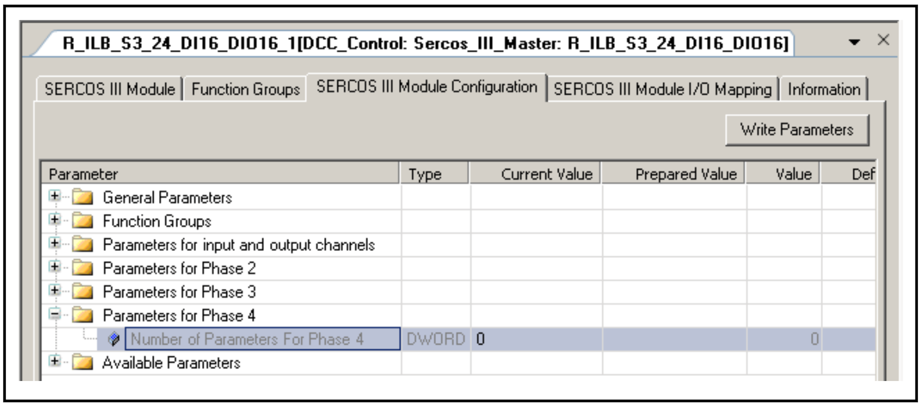
Symbolic values: defines whether the value is displayed in plaintext (0 to 10 V) or as an integer (0).

Default: resets all parameters to their initial value.

Tab "sercos III Module Configuration"

This window is used for service purposes and is only visible if in **Tools** ▸ **Options** ▸ **IndraLogic 2G** ▸ **Device Editor** the option "Display Generic Configuration Views" was enabled.

☞ Please clarify any possible modifications to the parameters that can be edited with the service team.


R_ILB_S3_24_DI16_DIO16_1[DCC_Control: Sercos_III_Master: R_ILB_S3_24_DI16_DIO16]

SERCOS III Module
Function Groups
SERCOS III Module Configuration
SERCOS III Module I/O Mapping
Information
Write Parameters

Parameter	Type	Current Value	Prepared Value	Value	Def
General Parameters					
Function Groups					
Parameters for input and output channels					
Parameters for Phase 2					
Parameters for Phase 3					
Parameters for Phase 4					
Number of Parameters For Phase 4	DWORD	0		0	
Available Parameters					

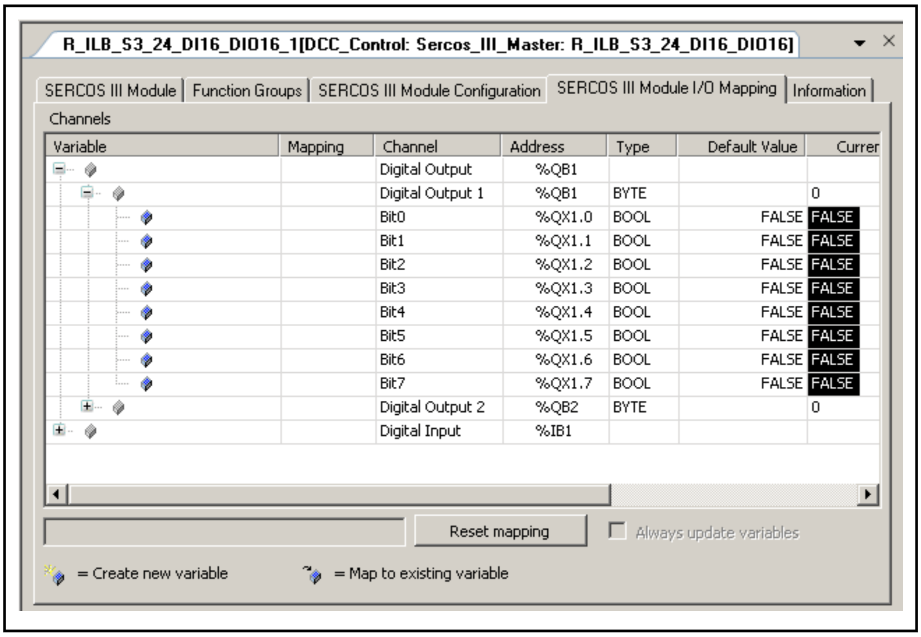
Fig.6-20: sercos III module: sercos III module configuration

Tab "sercos III Modules I/O Mapping"

The window is used to assign sercos III module inputs and outputs to variables that can be used as local or global variables in the individual POUs.

The current value of the variables is displayed in online mode.

This assignment is described in Mapping the Onboard, Inline and Field Bus Inputs and Outputs, page 139,.


R_ILB_S3_24_DI16_DIO16_1[DCC_Control: Sercos_III_Master: R_ILB_S3_24_DI16_DIO16]

SERCOS III Module
Function Groups
SERCOS III Module Configuration
SERCOS III Module I/O Mapping
Information
Channels

Variable	Mapping	Channel	Address	Type	Default Value	Currer
		Digital Output	%QB1			
		Digital Output 1	%QB1	BYTE		0
		Bit0	%QX1.0	BOOL	FALSE	FALSE
		Bit1	%QX1.1	BOOL	FALSE	FALSE
		Bit2	%QX1.2	BOOL	FALSE	FALSE
		Bit3	%QX1.3	BOOL	FALSE	FALSE
		Bit4	%QX1.4	BOOL	FALSE	FALSE
		Bit5	%QX1.5	BOOL	FALSE	FALSE
		Bit6	%QX1.6	BOOL	FALSE	FALSE
		Bit7	%QX1.7	BOOL	FALSE	FALSE
		Digital Output 2	%QB2	BYTE		0
		Digital Input	%IB1			

Fig.6-21: sercos III module: sercos III Module I/O Mapping

Reset mapping

Deletes the assignment made in the editor.

Always update variables

If this option is enabled, all variables are updated in each bus cycle, whether they are used or not no matter if they are mapped on an input or an output channel.

6.3.7 Tab "Information"

The window displays some general information from the device description file:

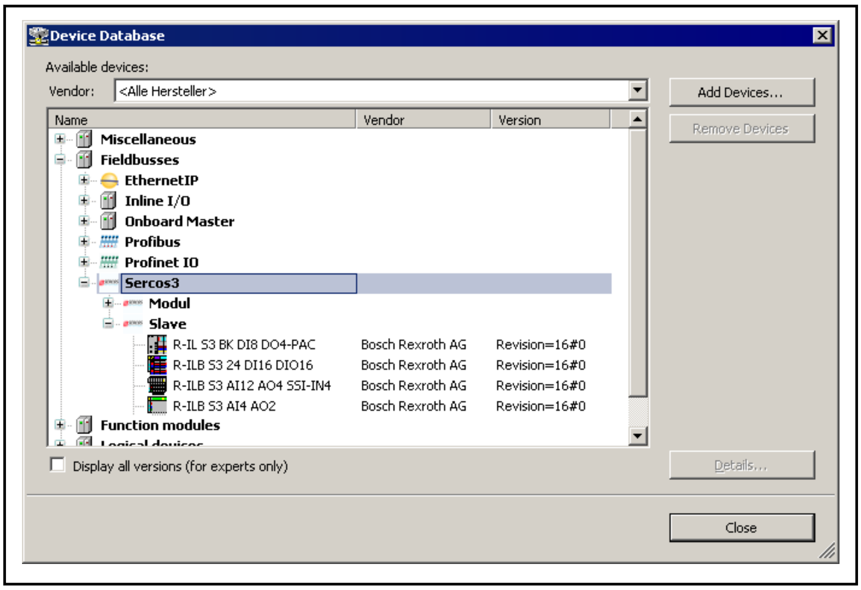
Name, Vendor, Categories, Version, Order number, Description, Image, if available.

6.3.8 sercos III, Reloading the Device Description File

To add devices in IndraWorks, please carry out the following steps:

1. Click on **Tools ▸ Device database...** in the main menu to open the "Device database" dialog.
2. In the "Device database" dialog, click on the "Add devices" button.

   The "Install device description" file selection dialog opens:



Device Database

Available devices:

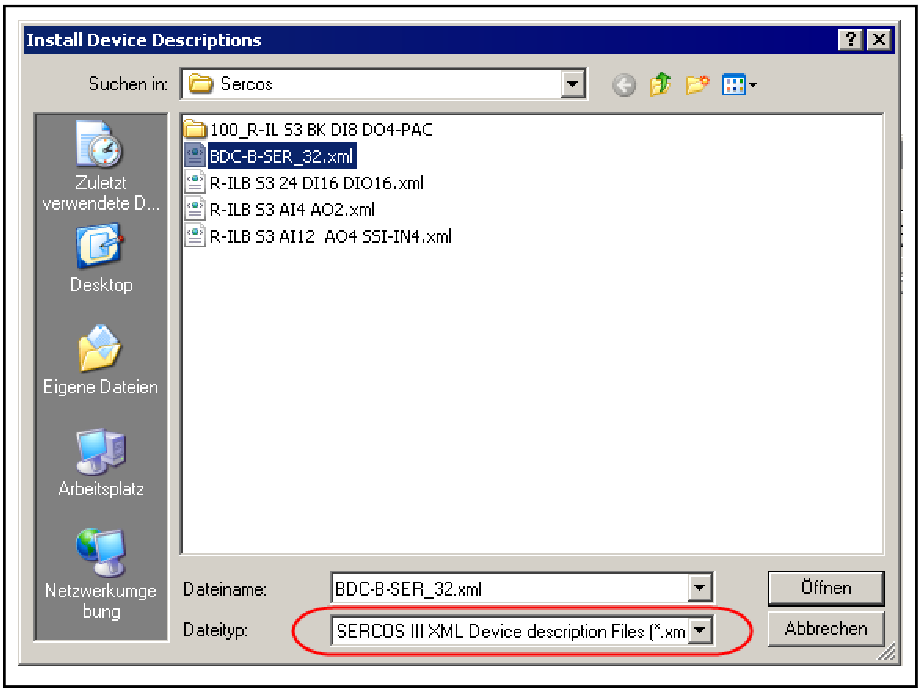
Vendor: <Alle Hersteller>

Name	Vendor	Version
Miscellaneous		
Fieldbusses		
EthernetIP		
Inline I/O		
Onboard Master		
Profibus		
Profinet IO		
Sercos3		
Modul		
Slave		
R-IL S3 BK DI8 DO4-PAC	Bosch Rexroth AG	Revision=16#0
R-ILB S3 24 DI16 DIO16	Bosch Rexroth AG	Revision=16#0
R-ILB S3 AI12 AO4 SSI-IN4	Bosch Rexroth AG	Revision=16#0
R-ILB S3 AI4 AO2	Bosch Rexroth AG	Revision=16#0
Function modules		
Logical devices		

□ Display all versions (for experts only)

Fig.6-22: Device Database, add device

3. Select the device description files to be installed. Make sure that the correct file type is listed in the "File type" selection field.



100_R-IL S3 BK DI8 DO4-PAC
BDC-B-SER_32.xml
R-ILB S3 24 DI16 DIO16.xml
R-ILB S3 AI4 AO2.xml
R-ILB S3 AI12 AO4 SSI-IN4.xml

Dateiname: BDC-B-SER_32.xml

Dateityp: SERCOS III XML Device description Files (*.xm

Fig.6-23: Install Device Descriptions dialog

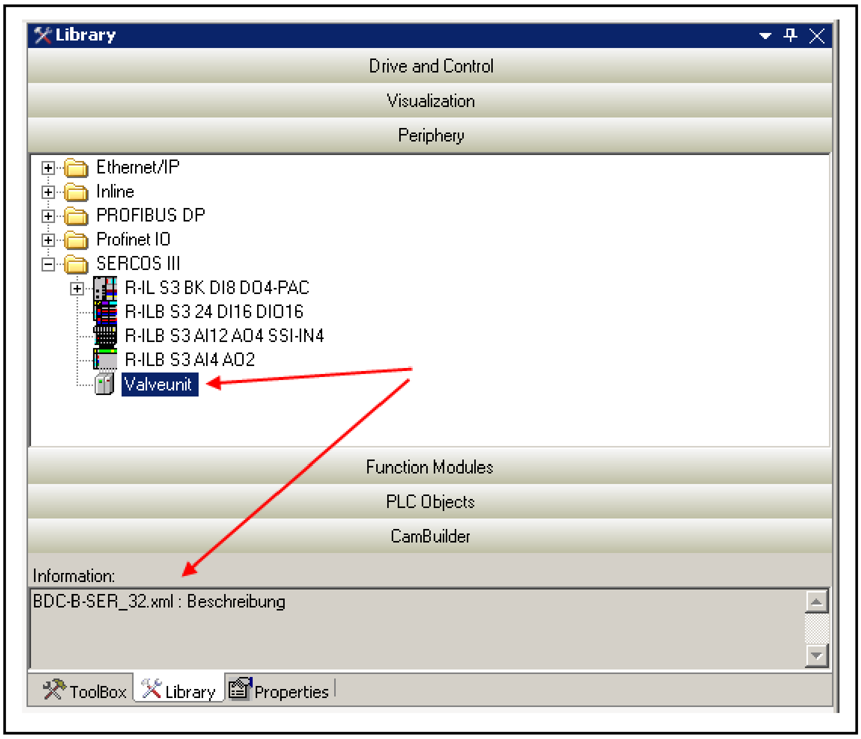
4. Start the installation procedure by clicking on "Open".

   After completed installation, a dialog appears confirming that all device description files have been imported. The imported devices appear below "Available devices" and are highlighted there:

5. Confirm the "Install Device Descriptions" message with "OK".
6. Close the "Device database" dialog with "Close".

   The newly installed devices are also displayed in the library:

sercos III I/O



BDC-B-SER_32.xml : Beschreibung

Fig.6-24: Imported devices displayed in the library

For more details on importing device files, please refer to:

- Device Database, page 135.

7 Inline I/Os

7.1 Features of the Inline I/Os

The Inline I/Os are classified into the following groups:

- Bus couplers for inline modules
- Inline modules
- Inline block modules

Bus couplers for inline modules

The following bus couplers are available for the inline modules:

- Inline modules on the PROFIBUS bus coupler (R-IL PB BK DI8 DO4-PAC)
- Inline modules on the PROFINET IO bus coupler (R_IL PN BK DI8 DO4-PAC)
- Inline modules on the sercos III bus coupler (R-IL S3 BK DI8 DO4-PAC)

Inline modules

The inline I/O modules can be connected the following couplers.

- Local inline bus on the control
- Inline modules on the bus couplers

The following I/O types are available:

- Digital input modules
- Digital output modules
- Analog input modules
- Analog output modules
- Relay modules
- Modules for temperature measurement
- Counter modules
- Modules with incremental encoder inputs
- Modules with SSI encoder inputs
- PWM modules
- Modules for serial communication (RS232, RS422, RS485)

Inline block modules

In addition to the bus couplers in connection with inline modules, the so-called block modules are also available. They contain a bus coupler and several IOs in a particularly compact and affordable design.

Block modules are available for the following busses:

- PROFIBUS digital IO
- sercos III digital IO
- sercos III analog IO

Related documentation

The documents listed below contain additional information regarding this subject.

Title	Identification
Automation Terminals of the Rexroth Inline Product Family	DOK-CONTRL-ILSYSINS***-AW01-EN-P
Rexroth Inline Bus Coupler for sercos III With Digital Inputs and Outputs	DOK-CONTRL-ILS3BKDI8DO-KB02-EN-P
Rexroth Inline Bus Coupler for PROFIBUS-DP With Digital Inputs and Outputs	DOK-CONTRL-ILPBBKDI8DO-AW02-EN-P

Inline I/Os

Title	Identification
Rexroth Inline Terminals with 2 Digital Inputs	DOK-CONTRL-ILDI2******-KB01-EN-P
Rexroth Inline Terminals with 2 Digital Inputs, Non-switching	DOK-CONTRL-ILDI2*NPN**-KB01-EN-P
Rexroth Inline Terminals with 2 Digital Inputs and DESINA Diagnostics	DOK-CONTRL-ILEDI2*DES*-KB01-EN-P
Rexroth Inline Terminals with 4 Digital Inputs	DOK-CONTRL-ILDI4******-KB01-EN-P
Rexroth Inline Terminals with 8 Digital Inputs	DOK-CONTRL-ILDI8******-KB01-EN-P
Rexroth Inline Terminals with 16 Digital Inputs	DOK-CONTRL-ILDI16*****-KB01-EN-P
Rexroth Inline Terminals with 16 Digital Inputs, Non-switching	DOK-CONTRL-ILDI16*NPN*-KB01-EN-P
Rexroth Inline Terminals with 16 Digital Inputs	DOK-CONTRL-ILDI32/HD**-KB01-EN-P
Rexroth Inline Terminals with 32 Digital Inputs, Non-switching	DOK-CONTRL-ILDI32*NPN*-KB01-EN-P
Rexroth Inline Terminals with 2 Digital Outputs	DOK-CONTRL-ILDO2*2A***-KB01-EN-P
Rexroth Inline Terminals with 2 Digital Outputs, Non-switching	DOK-CONTRL-ILDO2*NPN**-KB01-EN-P
Rexroth Inline Terminals with 4 Digital Outputs	DOK-CONTRL-ILDO4******-KB01-EN-P
Rexroth Inline Terminals with 8 Digital Outputs	DOK-CONTRL-ILDO8******-KB01-EN-P
Rexroth Inline Terminals with 8 Digital Outputs, Non-switching	DOK-CONTRL-ILDO8*NPN**-KB01-EN-P
Rexroth Inline Terminals with 8 Digital Outputs	DOK-CONTRL-ILDO8*2A***-KB01-EN-P
Rexroth Inline Terminals with 16 Digital Outputs	DOK-CONTRL-ILDO16*****-KB01-EN-P
Rexroth Inline Terminals with 32 Digital Outputs	DOK-CONTRL-ILDO32/HD**-KB01-EN-P
Rexroth Inline Terminals with 32 Digital Outputs, Non-switching	DOK-CONTRL-ILDO32*NPN*-KB01-EN-P
Rexroth Inline Terminals with 2 Analog Input Channels	DOK-CONTRL-ILDO4******-KB01-EN-P
Rexroth Inline Terminals with 2 Analog Input Channels	DOK-CONTRL-ILAI2/SF230-KB01-EN-P
Rexroth Inline Terminals with 8 Analog Input Channels	DOK-CONTRL-ILAI8/IS***-KB02-EN-P
Rexroth Inline Terminals with 8 Analog Input Channels	DOK-CONTRL-ILAI8/SF***-KB02-EN-P
Rexroth Inline Terminals with 4 Analog Difference Input Channels	DOK-CONTRL-ILDO4******-KB01-EN-P
Rexroth Inline Terminals with Two Inputs for Thermo Elements	DOK-CONTRL-ILTEMP2UTH*-KB01-EN-P
Rexroth Inline Terminals with Two Inputs for Temperature Sensors	DOK-CONTRL-ILTEMP2RTD*-KB01-EN-P
Rexroth Inline Terminals with 2 Analog Inputs for Strain Gauges	DOK-CONTRL-ILSGI2/F***-KB01-EN-P
Rexroth Inline Terminals with 1 Analog Output	DOK-CONTRL-ILAO1/SF***-KB01-EN-P
Rexroth Inline Terminals with 2 Analog Outputs	DOK-CONTRL-ILAO2/SF***-KB01-EN-P
Rexroth Inline Terminals with 2 Analog Voltage Outputs	DOK-CONTRL-ILAO2/U/BP*-KB01-EN-P
Rexroth Inline Terminals with a Relay Changeover Contact	DOK-CONTRL-ILDOR1/W***-KB01-EN-P
Rexroth Inline Terminals with 4 Relay Changeover Contacts	DOK-CONTRL-ILDOR4/W***-KB01-EN-P
Rexroth Inline Terminals for Absolute Encoders with SSI Interfaces	DOK-CONTRL-ILSSIIN****-KB01-EN-P
Rexroth Positioning Terminals for Absolute Encoders	DOK-CONTRL-ILSSI******-AW01-EN-P
Rexroth Inline Terminals for Incremental Encoders	DOK-CONTRL-ILINC*IN***-KB02-EN-P
Rexroth Inline Counter Terminals	DOK-CONTRL-ILCNT******-KB01-EN-P
Rexroth Inline Terminals for Pulse Width and Frequency Modulation	DOK-CONTRL-ILPWM/2****-KB01-EN-P

Title	Identification
Rexroth Inline Terminals for Serial Data Transfer	DOK-CONTRL-ILRS232*P**-KB01-EN-P
Rexroth Inline Terminals for Serial Data Transfer	DOK-CONTRL-ILRS485*P**-KB02-EN-P
Rexroth Inline Extension Terminals for Extending the Inline Local Bus	DOK-CONTRL-ILLSKIP****-KB01-EN-P
Rexroth Inline Branch Terminals for Fieldline Modular Coupling	DOK-CONTRL-ILFLM******-KB01-EN-P
Rexroth Inline Power Supply Terminals for Supplying the Logic Voltage	DOK-CONTRL-ILPWRIN/R**-KB01-EN-P
Rexroth Inline Power Supply Terminals	DOK-CONTRL-ILPWRIN/2F*-KB01-EN-P
Rexroth Inline Power Supply Terminals	DOK-CONTRL-ILPWRIN****-KB01-EN-P
Rexroth Inline Segment Terminals	DOK-CONTRL-ILSEG/*****-KB01-EN-P
Rexroth Inline Segment Terminals	DOK-CONTRL-ILSEG/F****-KB01-EN-P
Rexroth Inline Segment Terminals	DOK-CONTRL-ILSEG/F*D**-KB01-EN-P

Fig.7-1: Inline modules

Not every I/O module can be used on every bus coupler or on the inline bus.

7.2 Configuring the Inline I/Os

7.2.1 Inline Object and Inline Modules, Overview

The controls

- IndraLogic XLC L25/L45/L65
- IndraMotion MLC L25/L45/L65

allow the locally available I/O units to be extended by arranging inline modules on the right side of the control.

The inline I/O object is to be extended in the Project Explorer with the desired inline modules.

Inline I/O object

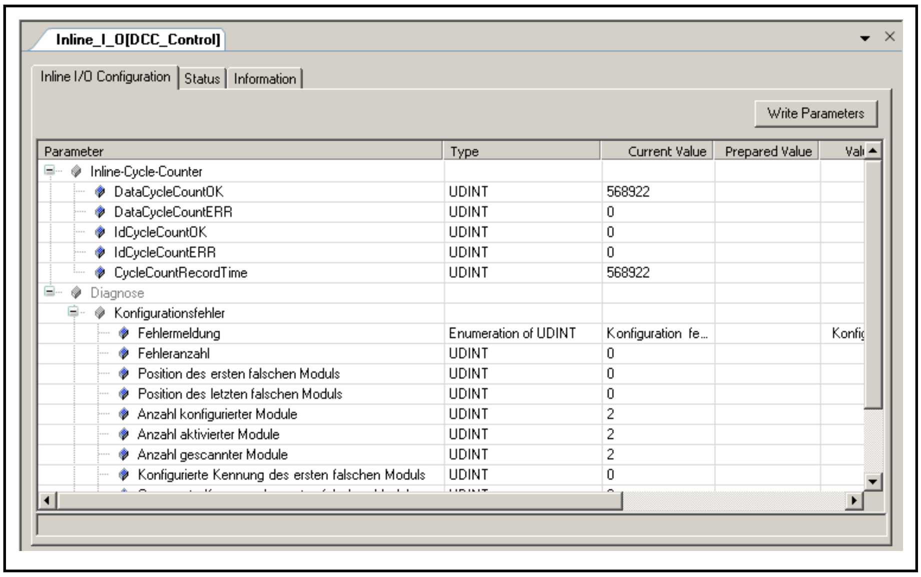
- Inline I/O Configuration, page 124
- Status, page 125 and
- Information, page 127

Inline I/O modules added to the "inline I/O object" (example)

- Adding Inline modules, page127,
- Inline modules I/O mapping, page 128,
- Status, page 129 and
- Information, page 129

7.2.2 Inline I/O object

Tab "Inline I/O Configuration"



Parameter	Type	Current Value	Prepared Value	Val
Inline-Cycle-Counter				
DataCycleCountOK	UDINT	568922		
DataCycleCountERR	UDINT	0		
IdCycleCountOK	UDINT	0		
IdCycleCountERR	UDINT	0		
CycleCountRecordTime	UDINT	568922		
Diagnose				
Konfigurationsfehler				
Fehlermeldung	Enumeration of UDINT	Konfiguration fe...		Konfig
Fehleranzahl	UDINT	0		
Position des ersten falschen Moduls	UDINT	0		
Position des letzten falschen Moduls	UDINT	0		
Anzahl konfigurierter Module	UDINT	2		
Anzahl aktivierter Module	UDINT	2		
Anzahl gescannter Module	UDINT	2		
Konfigurierte Kennung des ersten falschen Moduls	UDINT	0		

Fig.7-2: Inline object: Inline I/O configuration, Online

The window contains information regarding the inline cycle counters and the diagnostics of the inline bus.

Window structure

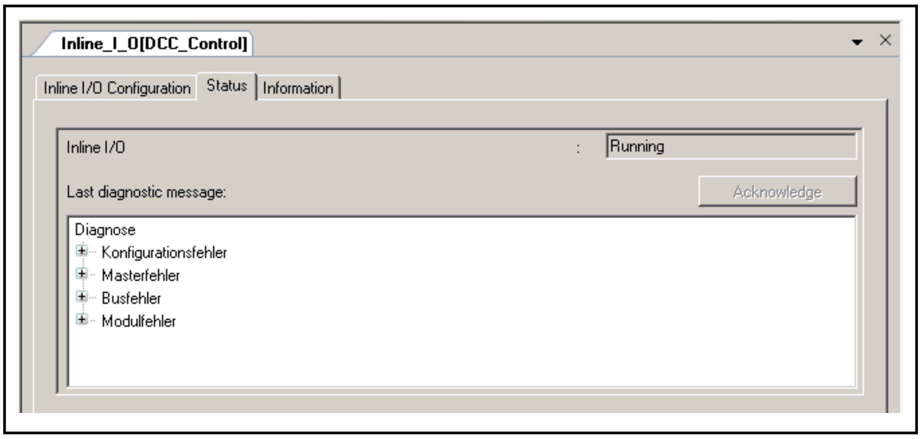
- **Parameters**: Parameter name from the device description file, cannot be edited.
- **Type**: Data type of the parameter, cannot be edited.
- **Value**: First, the standard value of the parameter is displayed, directly or as a specification of the corresponding symbolic name.

  If the parameter can be edited (this depends on the device description; parameters that cannot be edited are displayed in light gray), an input field or a selection list can be opened by double-clicking on the table field (or pressing the <space bar> in a previously selected field) where the value can be changed.

  Values are accepted with <Write parameter>.

  If the value is related to a file specification, the standard dialog for selected a file opens.
- **Default Value**: Defined value from the device description, cannot be edited.
- **Unit**: Unit for the value, e.g. "ms" for milliseconds, cannot be edited.
- **Description**: Short description of the parameter from the device description file, cannot be edited.

Tab "Status"



Inline I/O : Running

Last diagnostic message:

Diagnose
- Konfigurationsfehler
- Masterfehler
- Busfehler
- Modulfehler

Fig.7-3: Inline object: Status (online)

The window displays the status of the entire inline bus.

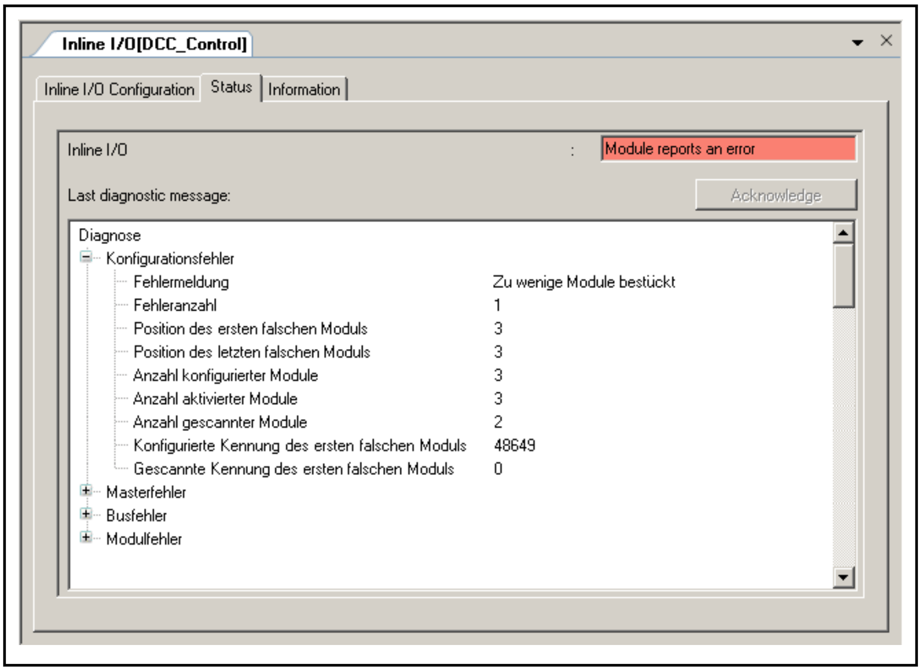
Offline: n/a

Online: "Running", "Not running (n/a)"

In addition, the "Most recent diagnostic message" is displayed, which can be confirmed with "Acknowledge".

Diagnostics in case of an error

In case of an error, a detailed diagnostic is transmitted. Here, the third module is missing at the real control.



Inline I/O : Module reports an error

Last diagnostic message:

Diagnose

Konfigurationsfehler

Fehlermeldung	Zu wenige Module bestückt
Fehleranzahl	1
Position des ersten falschen Moduls	3
Position des letzten falschen Moduls	3
Anzahl konfigurierter Module	3
Anzahl aktivierter Module	3
Anzahl gescannter Module	2
Konfigurierte Kennung des ersten falschen Moduls	48649
Gescannte Kennung des ersten falschen Moduls	0

Masterfehler

Busfehler

Modulfehler

Fig.7-4: Detailed error message in the I/O inline bus

1. **Configuration error**:

   In this case of error, the configuration of the modules in the project does not match with the modules that are physically present on the bus.

   Possible messages are:

   - Configuration has no errors
   - Too many modules loaded

    i.e. there are more modules on the bus than there are configured in the project.

  - Too few modules loaded

    I.e., there are less modules on the bus than there are configured in the project. The modules that are not fitted are shown with a yellow warning triangle in the project tree.

  - Modules loaded improperly

    i.e. there are other modules on the bus than those configured in the project. The modules that are improperly configured are shown in the project tree with a yellow warning triangle.

  - Other configuration errors

Additionally, further information is provided:

- Number of errors
- Position of the first and final improperly configured module
- Number of configured modules in the project
- Number of modules activated in the project, see also Enabling / Disabling Inline Modules, page 127
- Number of modules present on the bus
- For the first improperly configured module:
  - Identification of the module in the project
  - Identification of the module on the bus

1. **Master errors**: These are internal errors associated with the inline master.

   Possible messages are:

   - Master has no errors
   - General master error

A error number is delivered here as additional information.

1. **Bus errors**: If these errors occur, the bus is not running, e.g. because a module has failed.

Possible messages are:

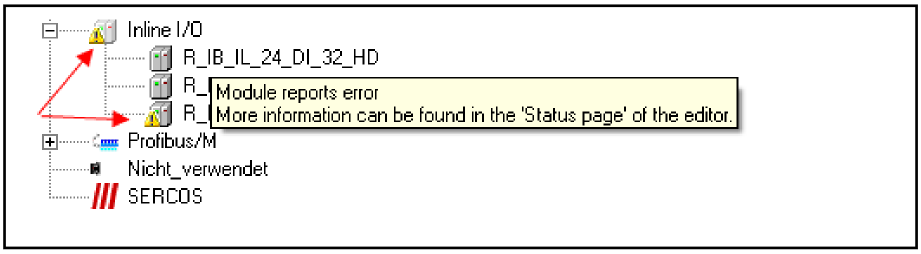
- The bus has no errors
- General bus error

The module position at which the error occurred and the number of modules on the bus are indicated. Starting with the module at this position, all of the modules are shown in the project tree with a yellow warning triangle.

1. **Module error**: At least one module reports an error (e.g. short circuit). The bus continues to run. Possible messages are:

   - Modules have no errors
   - Module specific error

The position of the first module with an error is indicated. Modules for which there are errors also appear in the project tree with a yellow warning triangle and an error tool tip.



Inline I/O
R_IB_IL_24_DI_32_HD
R_
R_
Module reports error
More information can be found in the 'Status page' of the editor.
Profibus/M
Nicht_verwendet
SERCOS

Fig.7-5: I/O inline bus, error message in the Project Explorer

Tab "Information"

The window displays some general information from the device description file:

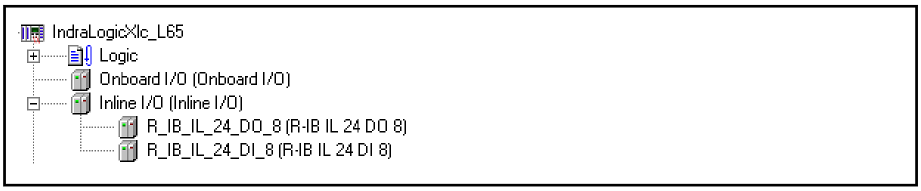
Name, Vendor, Categories, Version, Order number, Description, Image, if available.

7.2.3 Adding Inline Modules

All of the inline modules available for the respective control are located in the "Periphery" library in the "Inline" folder.

Drag the required inline modules out of the library into the respective "Inline I/O" object. New inline modules can also be added between existing inline modules in Project Explorer.

Alternatively, you can add I/O modules in the context menu using **Add ▸ Module ▸ <Inline module>** for the "Inline I/O" object. The new module is added as the last module below "Inline I/O".

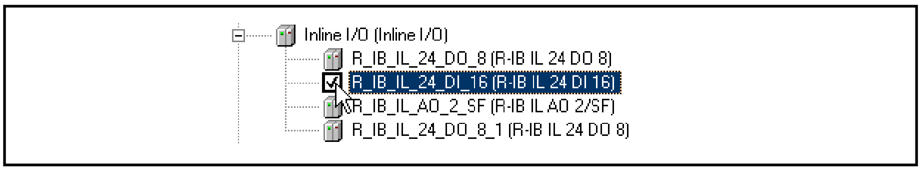


IndraLogicXlc_L65
Logic
Onboard I/O (Onboard I/O)
Inline I/O (Inline I/O)
R_IB_IL_24_DO_8 (R-IB IL 24 DO 8)
R_IB_IL_24_DI_8 (R-IB IL 24 DI 8)

Fig.7-6: Inline object with two modules (example)

7.2.4 Enabling / Disabling Inline Modules

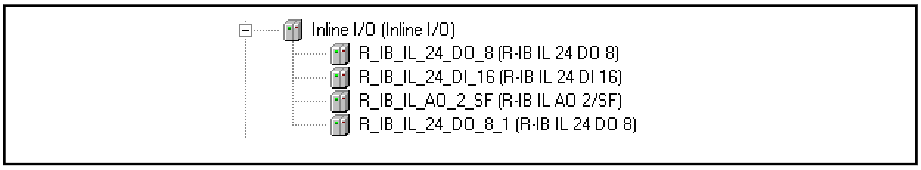
The Inline modules configured at a control can be "enabled" or "disabled" in the Project Explorer. This is carried out by selecting/deselecting the button appearing at the icon of the Inline module.



Inline I/O (Inline I/O)
R_IB_IL_24_DO_8 (R-IB IL 24 DO 8)
☑ R_IB_IL_24_DI_16 (R-IB IL 24 DI 16)
R_IB_IL_AO_2_SF (R-IB IL AO 2/SF)
R_IB_IL_24_DO_8_1 (R-IB IL 24 DO 8)

Fig.7-7: Enabling/disabling Inline modules

The module is enabled if the checkbox is ticked. It is taken into account in the diagnostics of the bus.

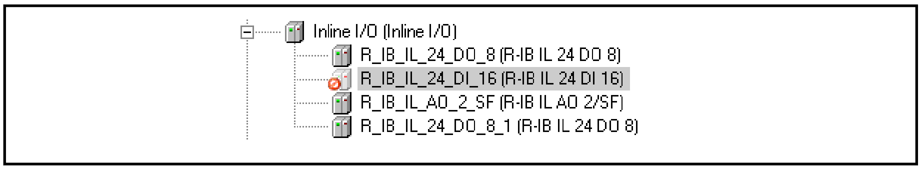


Inline I/O (Inline I/O)
R_IB_IL_24_DO_8 (R-IB IL 24 DO 8)
R_IB_IL_24_DI_16 (R-IB IL 24 DI 16)
R_IB_IL_AO_2_SF (R-IB IL AO 2/SF)
R_IB_IL_24_DO_8_1 (R-IB IL 24 DO 8)

Fig.7-8: Inline bus with enabled modules

The module is disabled if the checkbox is unticked. It is not taken into account in the diagnostics of the bus.

Inline I/Os



```
Inline I/O (Inline I/O)
    R_IB_IL_24_DO_8 (R-IB IL 24 DO 8)
    R_IB_IL_24_DI_16 (R-IB IL 24 DI 16)
    R_IB_IL_AO_2_SF (R-IB IL AO 2/SF)
    R_IB_IL_24_DO_8_1 (R-IB IL 24 DO 8)
```

Fig.7-9: Inline bus with disabled modules

If a new Inline module is created, it is enabled by default.

When loading the configuration, all enabled Inline modules are considered. Inline modules switched to passive mode are ignored.

☞ A module that is enabled but not present generates an error message in the diagnostics.

A module that is disabled and not present does not generate any error message in the diagnostics.

7.2.5 Configuring Inline Modules

General

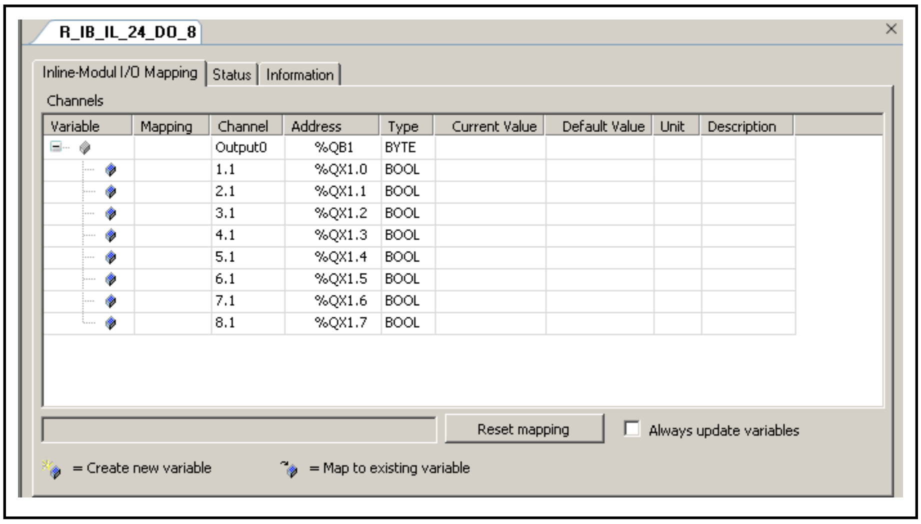
In Project Explorer double-click on the inline module that you wish to configure.

The inline module editor contains three tabs that you can open by clicking on them.

Tab:

- Inline Module I/O Mapping, page 128
- Status, page 129 and
- Information, page 129

Tab "Inline Module I/O Mapping"



R_IB_IL_24_DO_8

Inline-Modul I/O Mapping | Status | Information

Channels

Variable	Mapping	Channel	Address	Type	Current Value	Default Value	Unit	Description
		Output0	%QB1	BYTE				
		1.1	%QX1.0	BOOL				
		2.1	%QX1.1	BOOL				
		3.1	%QX1.2	BOOL				
		4.1	%QX1.3	BOOL				
		5.1	%QX1.4	BOOL				
		6.1	%QX1.5	BOOL				
		7.1	%QX1.6	BOOL				
		8.1	%QX1.7	BOOL				

Fig.7-10: Inline module: Inline , module I/O mapping

The window is used to assign inline module inputs and outputs to variables that can be used as local or global variables in the individual POUs.

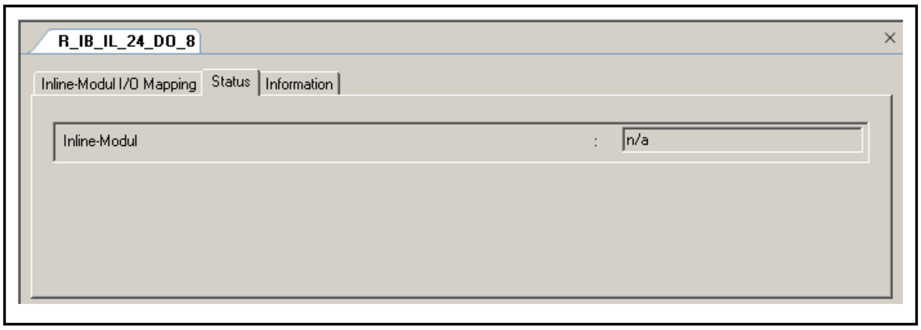
The current value of the variables is displayed in online mode.

This assignment is described in Mapping the Onboard, Inline and Field Bus Inputs and Outputs, page 139,.

Reset mapping: Deletes the assignment made in the editor.

Always update variables: If this option is enabled, all variables are updated in each cycle of the bus cycle task of the control (double click on the actual control in the Project Explorer, PLC settings), no matter whether they are used or not and whether they are mapped on an input or an output channel.

Tab "Status"



Inline-Modul : n/a

Fig.7-11: Inline module: Status

The window displays the status of the entire actual module.

Offline: n/a

Online: "Running", "Not running (n/a)"

Tab "Information"

The window displays some general information from the device description file:

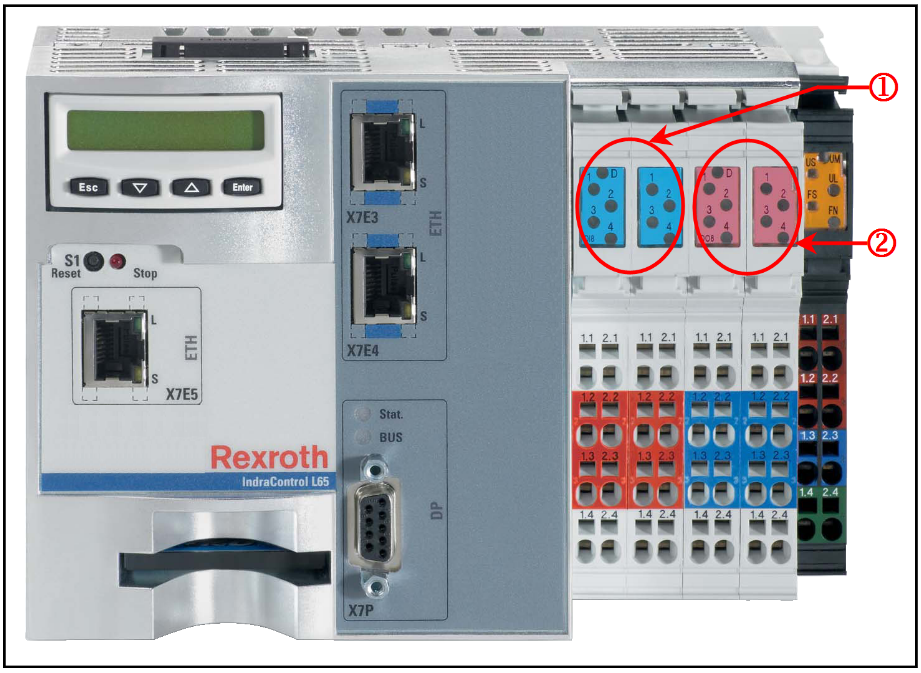
Name, Vendor, Categories, Version, Order number, Description, Image, if available.

8 Onboard I/Os

8.1 Configuring the Onboard I/Os

The IndraLogic XLC L45/L65 and IndraMotion MLC L45/L65 controls each feature eight fast interruptible digital inputs and outputs.

☞ For information on the interrupt capability, please refer to the description of the Task Editor in documentation "Rexroth IndraWorks 11VRS IndraLogic 2G, PLC Programming System", DOK-CONTRL-IL2GPRO*V11-AP01-EN-P.



① Inputs
② Outputs

Fig.8-1: Example: IndraLogic XLC L65 control

The inputs and outputs available from left to right in positions 1 to 4 are assigned to the LEDs and bit addresses according to the following table:

		Inputs								Outputs							
	Slot	1				2				3				4			
	Status LED	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4
Byte-Bit view	Byte	IX0.0 – 0.7 (default)								QX0.0 – 0.7 (default)							
	Bit	0	1	2	3	4	5	6	7	0	1	2	3	4	5	6	7
Module	Terminal connection (Signal)	1.1	2.1	1.4	2.4	1.1	2.1	1.4	2.4	1.1	2.1	1.4	2.4	1.1	2.1	1.4	2.4
	Terminal connection (24 V)	1.2	2.2	1.3	2.3	1.2	2.2	1.3	2.3	-	-	-	-	-	-	-	-
	Terminal connection (last ground)	-	-	-	-	-	-	-	-	1.2	2.2	1.3	2.3	1.2	2.2	1.3	2.3

Fig.8-2: Default address assignment for inputs and outputs

These inputs and outputs are configured using the onboard editor.

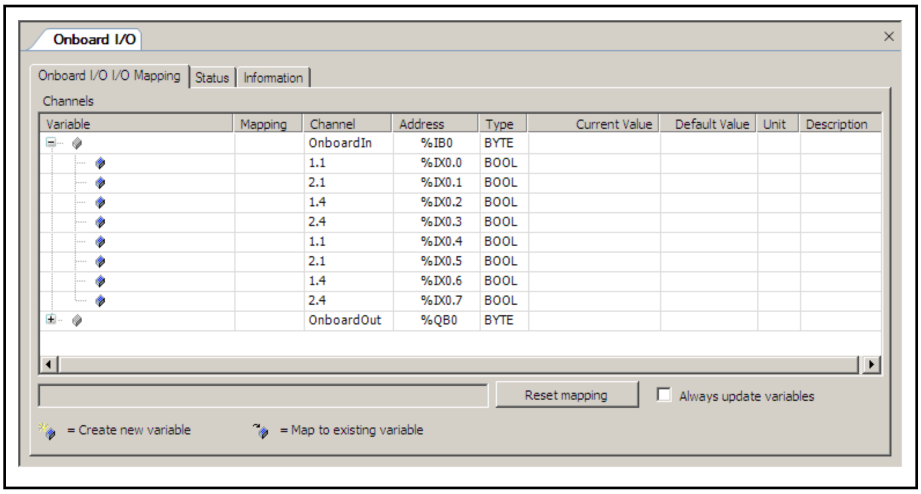
To do this, in the Project Explorer, double click on the "Onboard I/O" object.

The onboard editor contains three tabs that you can open by clicking on them.

Register:

- Onboard I/O I/O mapping, page 132
- Status, page 132 and
- Information, page 133

8.2 Register "Onboard I/O I/O Mapping"



Variable	Mapping	Channel	Address	Type	Current Value	Default Value	Unit	Description
		OnboardIn	%IB0	BYTE				
		1.1	%IX0.0	BOOL				
		2.1	%IX0.1	BOOL				
		1.4	%IX0.2	BOOL				
		2.4	%IX0.3	BOOL				
		1.1	%IX0.4	BOOL				
		2.1	%IX0.5	BOOL				
		1.4	%IX0.6	BOOL				
		2.4	%IX0.7	BOOL				
		OnboardOut	%QB0	BYTE				

Fig.8-3: Onboard I/O: Onboard I/O I/O mapping:

The window is used to assign onboard inputs and outputs to variables that can be used as local or global variables in the individual POUs.

The current value of the variables is displayed in online mode.

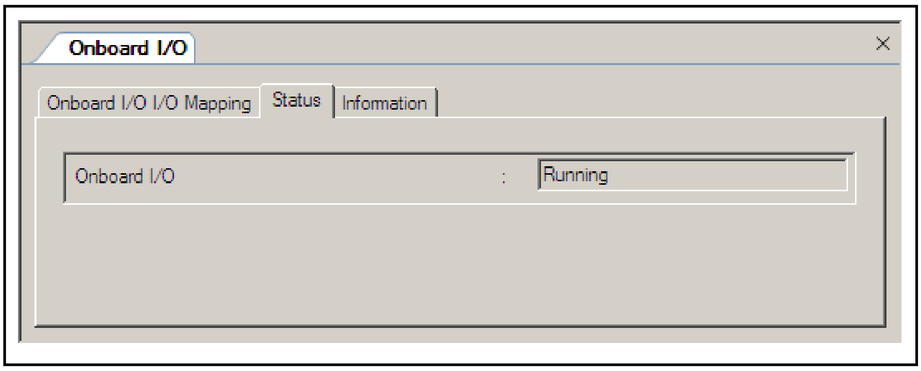
Reset mapping

Deletes the assignment made in the editor.

Always update variables

If this option is enabled, all variables are updated in each cycle of the bus cycle task, whether they are used or not no matter if they are mapped on an input or an output channel.

8.3 Register "Status"



Onboard I/O : Running

Fig.8-4: Onboard I/O: Status

The window displays the bus state.

Offline: n/a

Online: "Running", "Not running (n/a)"

8.4 Register "Information"

The window displays some general information from the device description file:

Name, Vendor, Categories, Version, Order number, Description, Image, if available.

9 Device database

9.1 Device Database, Overview

The device database is an extension of the device library (IndraWorks, right side).

The device database is a database for device descriptions installed on the local system to make them available for projects in IndraWorks. Properties (e.g. parameters, vendor, device name, etc.) for the respective device(s) are stored in the device description files.

There are different formats for different devices:

- PROFIBUS DP: *.gs?,
- PROFINET IO: GSDML*.xml,
- EtherNet/IP: EDS*.xml,
- sercos III IO: SDDML-Datei, *.xml.

The device database is installed along with the IndraWorks installation.

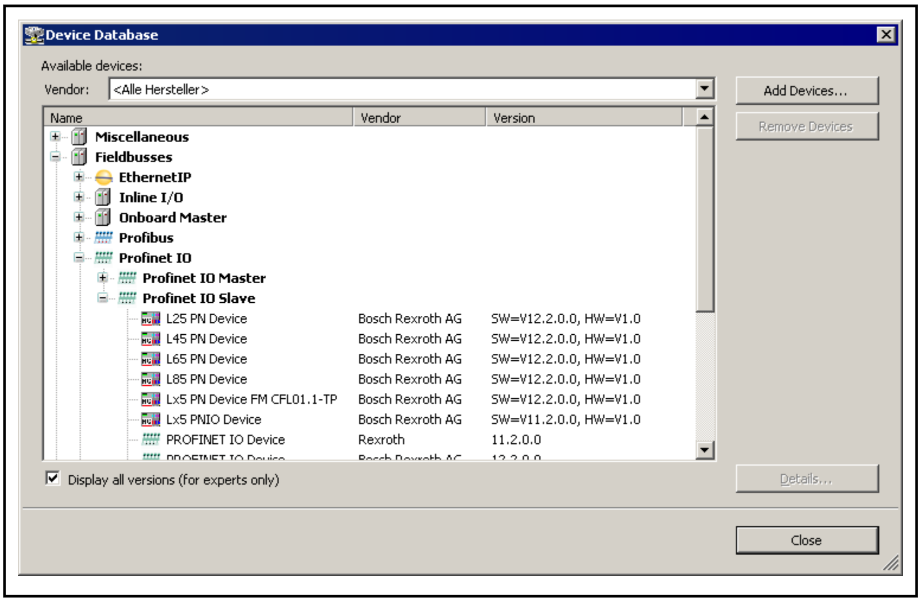
Installing additional devices and uninstalling unneeded devices can be done in the device database itself.

9.2 Managing Devices

9.2.1 Device Database, Dialog

Icon:

The "Device database" command can be used to install and uninstall device description files. In the main menu click on **Tools** ▸ **Device Database...**to open the "Device Database" dialog:



Available devices:

Vendor: <Alle Hersteller>

Name	Vendor	Version
Miscellaneous		
Fieldbusses		
EthernetIP		
Inline I/O		
Onboard Master		
Profibus		
Profinet IO		
Profinet IO Master		
Profinet IO Slave		
L25 PN Device	Bosch Rexroth AG	SW=V12.2.0.0, HW=V1.0
L45 PN Device	Bosch Rexroth AG	SW=V12.2.0.0, HW=V1.0
L65 PN Device	Bosch Rexroth AG	SW=V12.2.0.0, HW=V1.0
L85 PN Device	Bosch Rexroth AG	SW=V12.2.0.0, HW=V1.0
Lx5 PN Device FM CFL01.1-TP	Bosch Rexroth AG	SW=V12.2.0.0, HW=V1.0
Lx5 PNIO Device	Bosch Rexroth AG	SW=V11.2.0.0, HW=V1.0
PROFINET IO Device	Rexroth	11.2.0.0
PROFINET IO Device	Bosch Rexroth AG	12.2.0.0

☑ Display all versions (for experts only)

Fig.9-1: Device database

Available devices

In the "Available devices" area, the currently installed devices are listed.

The devices have a hierarchical order, i.e., controls are arranged in the topmost level, followed by function modules, integrated interfaces, etc., in the levels underneath.

The list can be limited by selected filter "Vendor".

In addition, devices can be added and removed in the "Available devices" panel.

Usually, the devices shown match your IndraWorks version. However, it is also possible to display a complete overview by activating the "Display all version" checkbox.

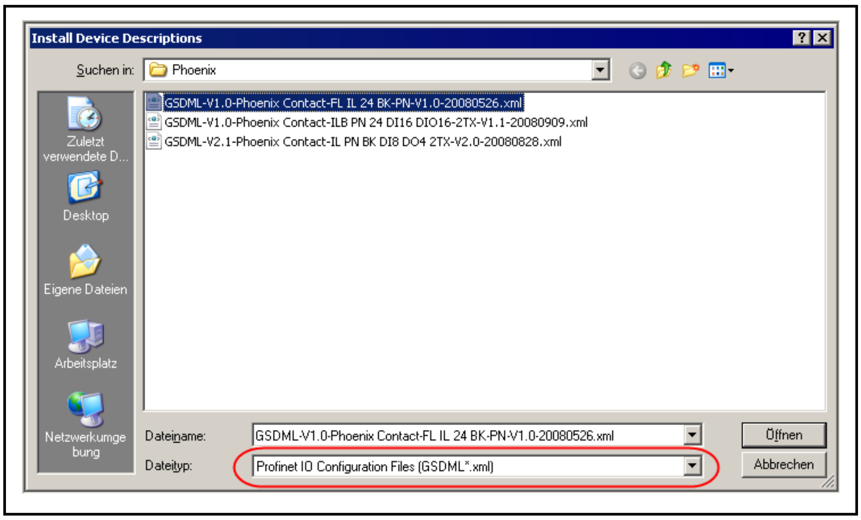
After having selected a device, click on "Details" to obtain additional information.

9.2.2 Add Devices

To add devices in IndraWorks, please carry out the following steps:

1. In the main menu click on **Tools ▸ Device Database...**to open the "Device Database" dialog.
2. In the "Device database" dialog, click on the "Add devices" button.

   The "Install device description" file selection dialog opens:



Install Device Descriptions

Suchen in: Phoenix

GSDML-V1.0-Phoenix Contact-FL IL 24 BK-PN-V1.0-20080526.xml
GSDML-V1.0-Phoenix Contact-ILB PN 24 DI16 DIO16-2TX-V1.1-20080909.xml
GSDML-V2.1-Phoenix Contact-IL PN BK DI8 DO4 2TX-V2.0-20080828.xml

Dateiname: GSDML-V1.0-Phoenix Contact-FL IL 24 BK-PN-V1.0-20080526.xml

Dateityp: Profinet IO Configuration Files (GSDML*.xml)

Fig.9-2: Installing device descriptions

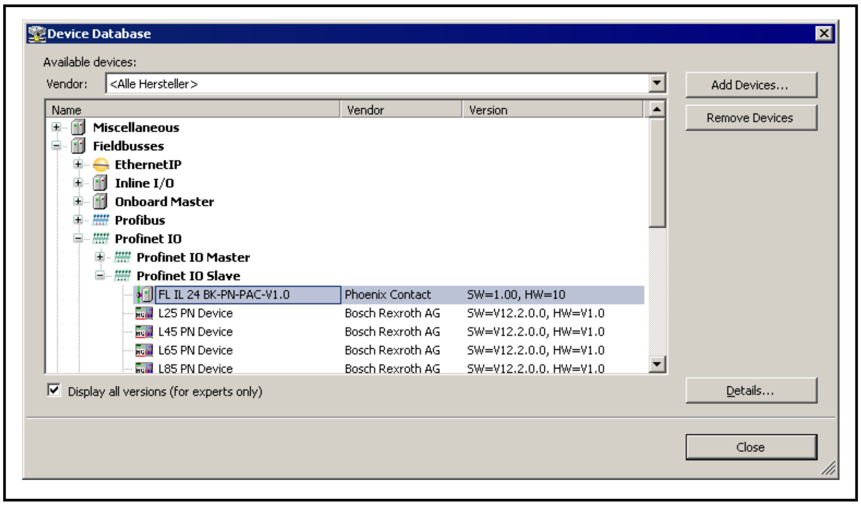
3. Select the desired device descriptions that you wish to install.

☞ Make sure that the correct file type is listed in the "File type" selection field.

4. Start the installation procedure by clicking on "Open".

   After completed installation, the imported devices appear under "Available devices" and are highlighted there:

Device database



Device Database

Available devices:

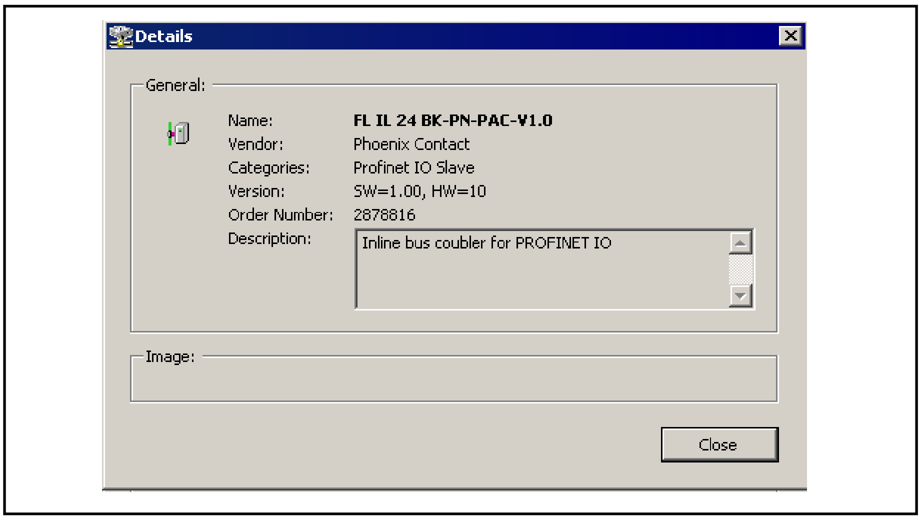
Vendor: <Alle Hersteller>

Name	Vendor	Version
Miscellaneous		
Fieldbusses		
EthernetIP		
Inline I/O		
Onboard Master		
Profibus		
Profinet IO		
Profinet IO Master		
Profinet IO Slave		
FL IL 24 BK-PN-PAC-V1.0	Phoenix Contact	SW=1.00, HW=10
L25 PN Device	Bosch Rexroth AG	SW=V12.2.0.0, HW=V1.0
L45 PN Device	Bosch Rexroth AG	SW=V12.2.0.0, HW=V1.0
L65 PN Device	Bosch Rexroth AG	SW=V12.2.0.0, HW=V1.0
L85 PN Device	Bosch Rexroth AG	SW=V12.2.0.0, HW=V1.0

☑ Display all versions (for experts only)

Fig.9-3: Device descriptions installed

5. Select the desired device and click on "Details" to obtain additional information.



Details

General:

Name: FL IL 24 BK-PN-PAC-V1.0
Vendor: Phoenix Contact
Categories: Profinet IO Slave
Version: SW=1.00, HW=10
Order Number: 2878816
Description: Inline bus coubler for PROFINET IO

Image:

Fig.9-4: Details on the device description file installed

6. Close the "Device database" dialog with "Close".

   The newly installed devices are also displayed in the library:

Device database

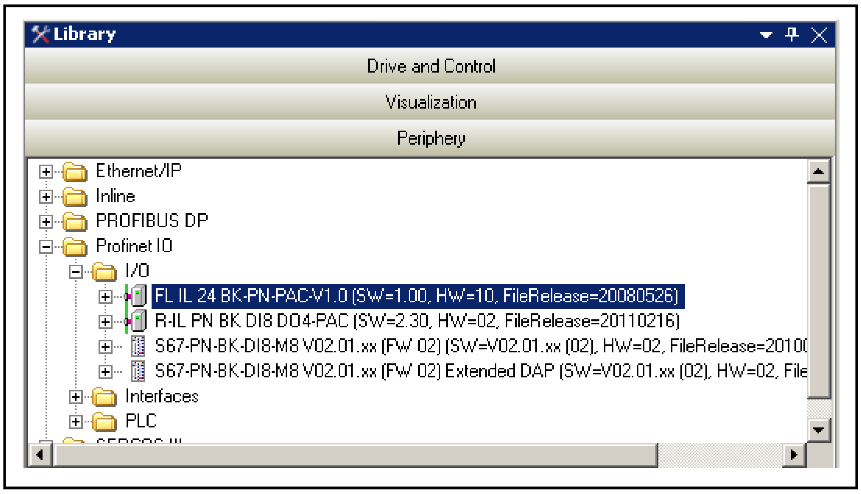

Library
Drive and Control
Visualization
Periphery
Ethernet/IP
Inline
PROFIBUS DP
Profinet IO
I/O
FL IL 24 BK-PN-PAC-V1.0 (SW=1.00, HW=10, FileRelease=20080526)
R-IL PN BK DI8 DO4-PAC (SW=2.30, HW=02, FileRelease=20110216)
S67-PN-BK-DI8-M8 V02.01.xx (FW 02) (SW=V02.01.xx (02), HW=02, FileRelease=2010
S67-PN-BK-DI8-M8 V02.01.xx (FW 02) Extended DAP (SW=V02.01.xx (02), HW=02, File
Interfaces
PLC

Fig.9-5: Imported device displayed in the library

9.2.3 Remove Devices

To remove devices from IndraWorks, please carry out the following steps:

1. In the main menu click on **Tools** ▸ **Device Database...**to open the "Device Database" dialog.
2. In the device tree, highlight the devices that you wish to remove.
3. Click on the "Remove devices" button.

   The "Confirm Device Removal" dialog opens:

4. Click"Yes" to remove the highlighted device.

   The device is removed from the device database and from the library.

5. Close the "Device database" dialog with "Close".

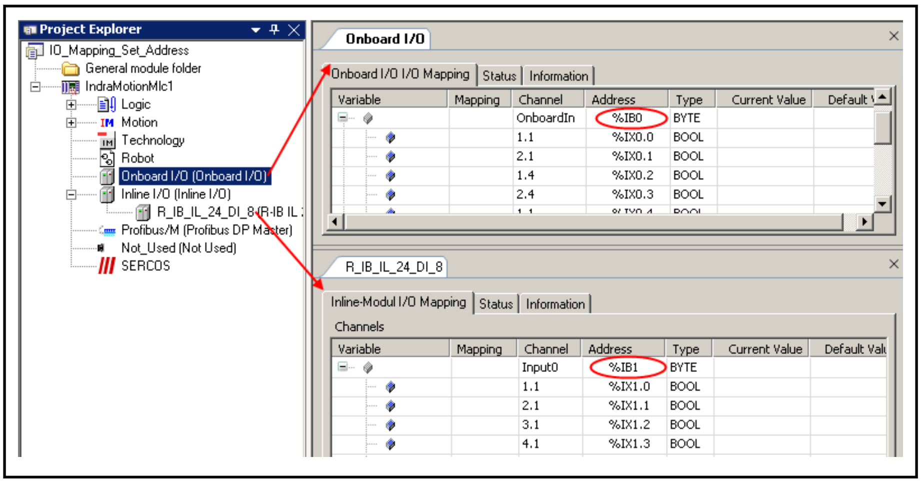
10 Mapping the Onboard, Inline and Field Bus Inputs and Outputs

These explanations apply equally to the following bus systems:

- OnBoard
- Inline
- PROFIBUS DP
- PROFINET IO
- Ethernet/IP
- sercos III IO

The module (the onboard bus, if present) starts with the address "%IB0" or "%IW0" for the inputs and analog "%QB0" or "%QW0" for the outputs.

The other modules follow accordingly, even if they already belong to the subsequent bus in the Project Explorer.



Variable	Mapping	Channel	Address	Type	Current Value	Default V
		OnboardIn	%IB0	BYTE		
		1.1	%IX0.0	BOOL		
		2.1	%IX0.1	BOOL		
		1.4	%IX0.2	BOOL		
		2.4	%IX0.3	BOOL		
		1.1	%IX0.4	BOOL		

Variable	Mapping	Channel	Address	Type	Current Value	Default Valu
		Input0	%IB1	BYTE		
		1.1	%IX1.0	BOOL		
		2.1	%IX1.1	BOOL		
		3.1	%IX1.2	BOOL		
		4.1	%IX1.3	BOOL		

Fig. 10-1: Address assignment beyond the bus limit

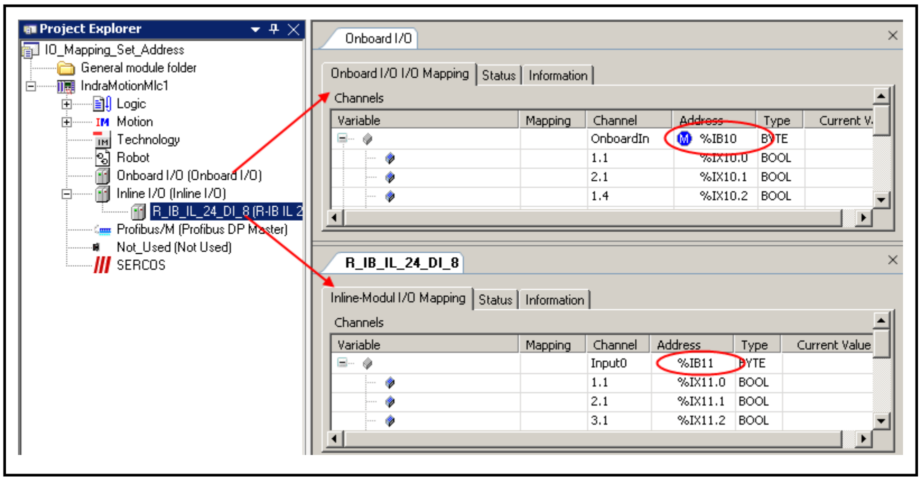
However, users can still assign the addresses as desired.

Clicking in the respective address field allows the address for the current module to be edited. The others follow suit.

An address modified in this way is identified by the white "M" on a blue background.

The other modules follow accordingly, even if they already belong to the subsequent bus in the Project Explorer.

Mapping the Onboard, Inline and Field Bus Inputs and Outputs



Variable	Mapping	Channel	Address	Type	Current V...
		OnboardIn	%IB10	BYTE	
		1.1	%IX10.0	BOOL	
		2.1	%IX10.1	BOOL	
		1.4	%IX10.2	BOOL	

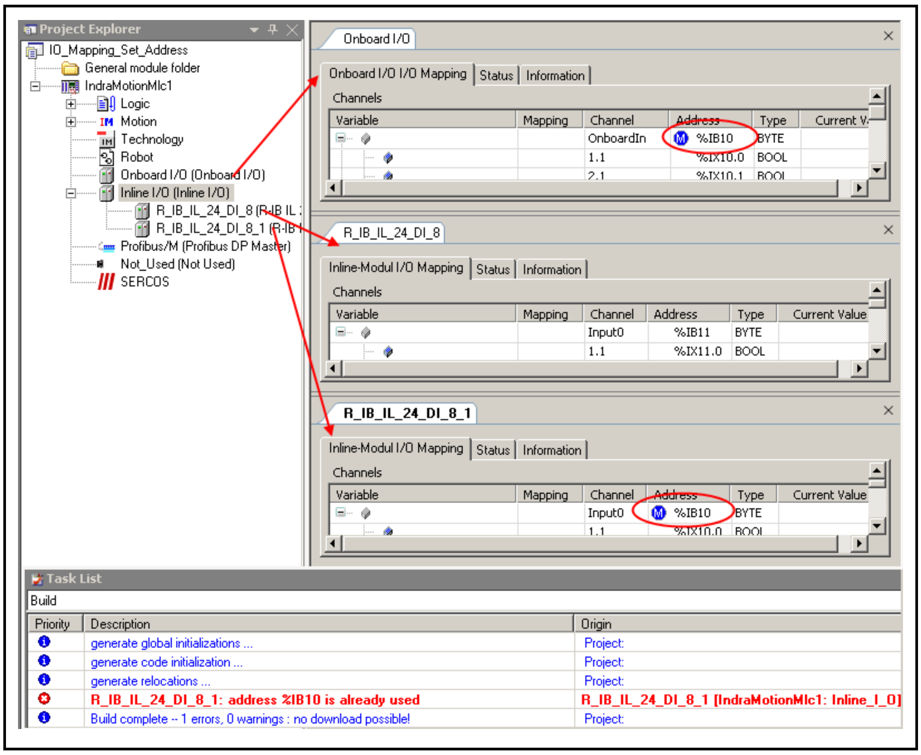
Variable	Mapping	Channel	Address	Type	Current Value
		Input0	%IB11	BYTE	
		1.1	%IX11.0	BOOL	
		2.1	%IX11.1	BOOL	
		3.1	%IX11.2	BOOL	

Fig. 10-2: Modified basic address

☞ With multiple mapping, there is a risk that the address ranges could overlap.

This is not detected until the **code is generated**.

The error message contains only the second assigned position....



Variable	Mapping	Channel	Address	Type	Current V...
		OnboardIn	%IB10	BYTE	
		1.1	%IX10.0	BOOL	
		2.1	%IX10.1	BOOL	

Variable	Mapping	Channel	Address	Type	Current Value
		Input0	%IB11	BYTE	
		1.1	%IX11.0	BOOL	

Variable	Mapping	Channel	Address	Type	Current Value
		Input0	%IB10	BYTE	
		1.1	%IX10.0	BOOL	

Priority	Description	Origin
ℹ	generate global initializations ...	Project:
ℹ	generate code initialization ...	Project:
ℹ	generate relocations ...	Project:
⊗	R_IB_IL_24_DI_8_1: address %IB10 is already used	R_IB_IL_24_DI_8_1 [IndraMotionMlc1: Inline_I_O]
ℹ	Build complete -- 1 errors, 0 warnings : no download possible!	Project:

Fig. 10-3: Mapping contains an error, addresses used more than once

☞ I/O modules of the same type on the respective slot, in the same sequence, are the prerequisite for automatic address generation results in the same configuration and then to the connection of the correct inputs and outputs.

Please check this carefully if the control is exchanged.

Mapping local variables:

- Variables:

  This column displays the input or output module. The plus and minus symbols allow you to switch between bit and byte display.

  For each absolute address, a symbolic address can be assigned (double-click on the respective field).

  For Map to existing variable the complete path must be entered, i.e.

  `<ApplicationName> <ProgramName> <VariableName>`.

  Example (figure below):

  `Application.MotionProg.x_in_1`,

  `Application.MotionProg.x_in_2`

  or

  `Application.MotionProg.x_out_1`,

  `Application.MotionProg.x_out_2`.

- Mapping:

  The icon is displayed in the Mapping column.

  Then the address value is displayed with a strike through, i.e. the %Qx.x-/ or %Ix.x address.

- Type:

  Byte address are identified with "BYTE" and bit address with "BOOL".

- Current value:

  Physical status of the input/output. The status is displayed only in the diagnostic mode for communication between IndraWorks and the control.

- Unit:

  Unit for the parameter value, e.g. "ms" for milliseconds.

- Description:

  Enter a comment regarding an address here.

Mapping global variables:

- Variables:

  This column displays the input or output module. The plus and minus symbols allow you to switch between bit and byte display.

  For each absolute address, a symbolic address can be assigned (double-click on the respective field).

  When mapping, only the name of the variable can be entered, i.e.

  `<Variable name>`.

  Example (figure below):

  `x_in_3` or

  `x_out_3`, `x_out_4`.

- Mapping:

  The icon is displayed in the Mapping column.

The variable is entered as VAR_GLOBAL in the "IoConfig_Globals_Mapping" list.

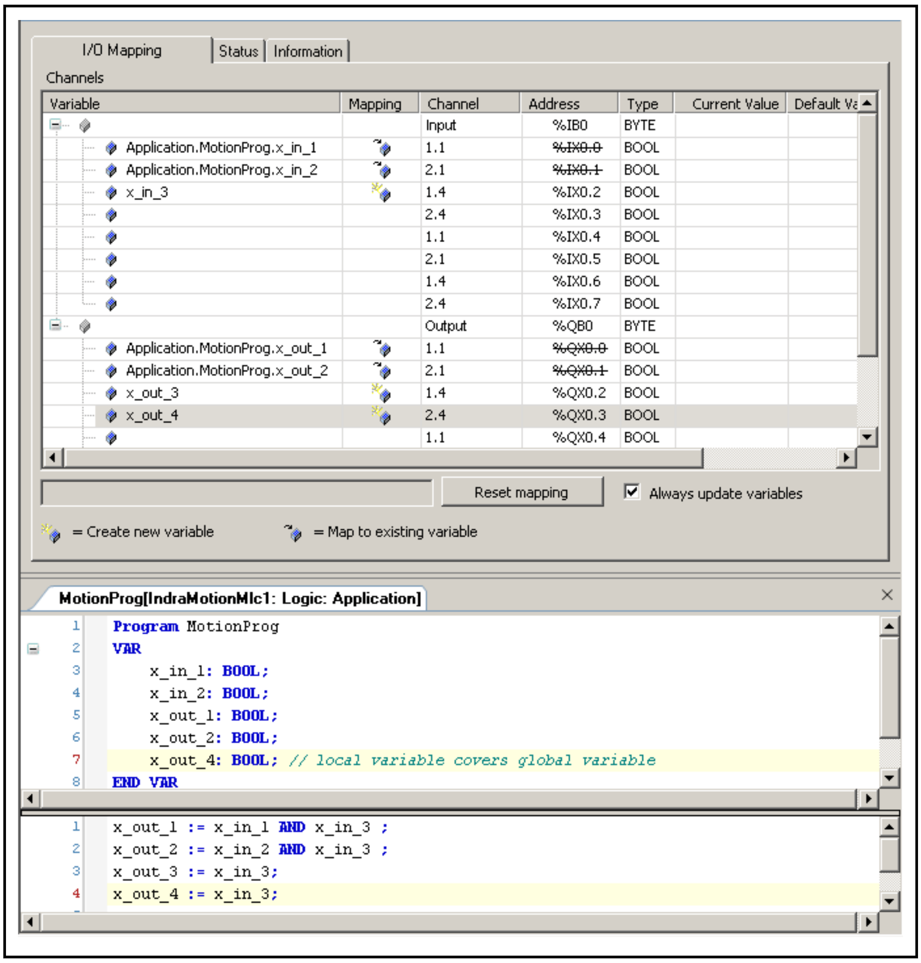
- **Type:**

  Byte address are identified with "BYTE" and bit address with "BOOL".
- **Current value:**

  Physical status of the input/output. The status is displayed only in the diagnostic mode for communication between IndraWorks and the control.
- **Unit:**

  Unit for the parameter value, e.g. "ms" for milliseconds.
- **Description:**

  Enter a comment regarding an address here.



Variable	Mapping	Channel	Address	Type	Current Value	Default Va
		Input	%IB0	BYTE		
Application.MotionProg.x_in_1	↷	1.1	~~%IX0.0~~	BOOL		
Application.MotionProg.x_in_2	↷	2.1	~~%IX0.1~~	BOOL		
x_in_3	✳	1.4	%IX0.2	BOOL		
		2.4	%IX0.3	BOOL		
		1.1	%IX0.4	BOOL		
		2.1	%IX0.5	BOOL		
		1.4	%IX0.6	BOOL		
		2.4	%IX0.7	BOOL		
		Output	%QB0	BYTE		
Application.MotionProg.x_out_1	↷	1.1	~~%QX0.0~~	BOOL		
Application.MotionProg.x_out_2	↷	2.1	~~%QX0.1~~	BOOL		
x_out_3	✳	1.4	%QX0.2	BOOL		
x_out_4	✳	2.4	%QX0.3	BOOL		
		1.1	%QX0.4	BOOL		

Reset mapping
☑ Always update variables
✳ = Create new variable
↷ = Map to existing variable
MotionProg[IndraMotionMlc1: Logic: Application]

```
Program MotionProg
VAR
    x_in_1: BOOL;
    x_in_2: BOOL;
    x_out_1: BOOL;
    x_out_2: BOOL;
    x_out_4: BOOL; // local variable covers global variable
END_VAR
```

```
x_out_1 := x_in_1 AND x_in_3 ;
x_out_2 := x_in_2 AND x_in_3 ;
x_out_3 := x_in_3;
x_out_4 := x_in_3;
```

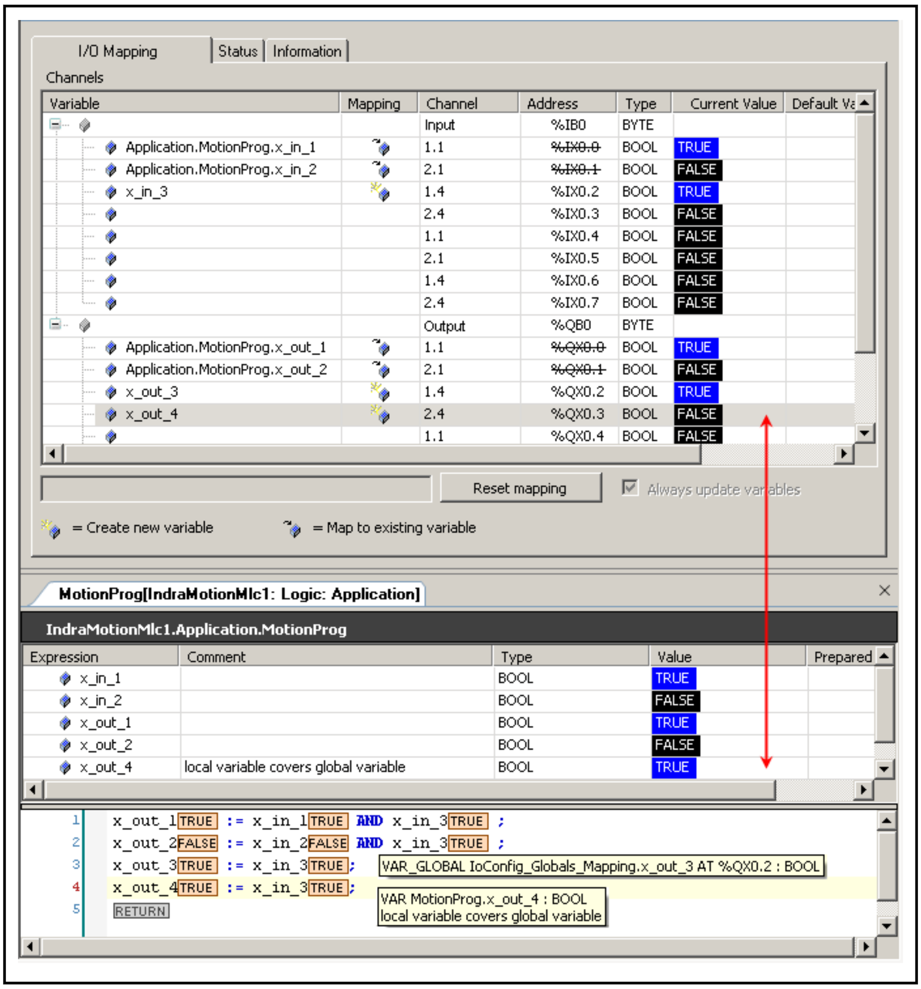
Fig.10-4: IO mapping, variable declaration and example program (offline)

The declaration section of the POU "MotionProg" contains

- the local POU variables,
- the variables that are to be mapped as inputs and outputs (without AT construction!).

☞ If you declare a local variable of the same name, as with the global list the local variable will be used!

Example: `x_out_4`.



Variable	Mapping	Channel	Address	Type	Current Value	Default Va
		Input	%IB0	BYTE		
Application.MotionProg.x_in_1	↷	1.1	~~%IX0.0~~	BOOL	TRUE	
Application.MotionProg.x_in_2	↷	2.1	~~%IX0.1~~	BOOL	FALSE	
x_in_3	✳	1.4	%IX0.2	BOOL	TRUE	
		2.4	%IX0.3	BOOL	FALSE	
		1.1	%IX0.4	BOOL	FALSE	
		2.1	%IX0.5	BOOL	FALSE	
		1.4	%IX0.6	BOOL	FALSE	
		2.4	%IX0.7	BOOL	FALSE	
		Output	%QB0	BYTE		
Application.MotionProg.x_out_1	↷	1.1	~~%QX0.0~~	BOOL	TRUE	
Application.MotionProg.x_out_2	↷	2.1	~~%QX0.1~~	BOOL	FALSE	
x_out_3	✳	1.4	%QX0.2	BOOL	TRUE	
x_out_4	✳	2.4	%QX0.3	BOOL	FALSE	
		1.1	%QX0.4	BOOL	FALSE	

Reset mapping
☑ Always update variables
✳ = Create new variable
↷ = Map to existing variable
MotionProg[IndraMotionMlc1: Logic: Application]
IndraMotionMlc1.Application.MotionProg

Expression	Comment	Type	Value	Prepared
x_in_1		BOOL	TRUE	
x_in_2		BOOL	FALSE	
x_out_1		BOOL	TRUE	
x_out_2		BOOL	FALSE	
x_out_4	local variable covers global variable	BOOL	TRUE	

```
x_out_1 TRUE  := x_in_1 TRUE  AND x_in_3 TRUE ;
x_out_2 FALSE := x_in_2 FALSE AND x_in_3 TRUE ;
x_out_3 TRUE  := x_in_3 TRUE ;   VAR_GLOBAL IoConfig_Globals_Mapping.x_out_3 AT %QX0.2 : BOOL
x_out_4 TRUE  := x_in_3 TRUE ;
RETURN                           VAR MotionProg.x_out_4 : BOOL
                                 local variable covers global variable
```

Fig. 10-5: IO mapping, variable declaration and example program (online)

Always update variables:

If this option is enabled, all variables are updated in each cycle of the bus cycle task, whether they are used or not no matter if they are mapped on an input or an output channel.

11 Field Bus Libraries

11.1 Basic Libraries, IndraLogic 2G - Overview

The **IndraLogic 2G field bus libraries** are described in the following sections, sorted by systems **XLC**, **MLC** and **MTX**:

Name	Description	XLC	MLC	MTX
RIL_ProfibusDP_02, page 159	PROFIBUS DP master V1 services, diagnostics, Sync and Freeze	(x)	(x)	(x)
RIL_ProfibusDPSlave, page 185	Profibus DP Slave V1 services, diagnostics	x	x	x
RIL_ProfinetIO, page 197	Diagnostic and communication serviced for PROFINET IO-Controller	x	x	x
RIL_ProfinetIODevice, page 220	Diagnostic and communication serviced for PROFINET IO-Device	x	x	x
RIL_EtherNetIPAdapter, page 230	Diagnostic and communication services for EtherNet/IP adapter and EtherNet/IP adapter (Engineering interface)	x	x	x
RIL_MappingList, page 242	Mapping table for acyclic accesses on the field bus slaves (DP slave, PNIO device and ENIP adapter)	x	x	
RIL_SERCOSIII, page 250	Diagnostic and communication services for sercos III	x	x	x
RIL_Inline, page 260	Diagnostic functions for Rexroth Inline modules	x	x	x

x Library exists/is available for the system
(x) Library is being processed
Fig. 11-1: IndraLogic 2G field bus libraries

11.2 Standard Interfaces at Function Blocks

11.2.1 Motivation

The majority of the function blocks disposes of an input for the activation and an output for displaying the correct processing.

In addition, an output is often necessary which indicates the processing time.

Additional outputs are defined for the indication of errors.

A uniform naming convention as well as an identical standard interface behavior, increases understanding, shortens the time needed for familiarization and reduces the workload of the support.

11.2.2 Function Block Types

With function blocks, it is possible to encapsulate complex tasks so they can be reused and to address them via defined interfaces. For this, the processing can be either state-controlled or edge-controlled. Furthermore, a difference is made between tasks which can be processed completely and tasks which require continuous intervention once they are started.

State-controlled

If a function block always repeats its task after switch-on whenever it has reached a defined state, this FB processes in a "state-controlled" way.

Edge-controlled

If a function block fulfills its task exactly once after being switched on, this FB processes in an "edge-controlled" way.

Terminating processing

If a block can definitively process its order and afterwards stop working, this is called "definitive processing".

Permanent processing

If a block can never conclude its order definitively but is continuously in engagement, this is defined as "continuous processing".

In combination, there are four FB types possible:

Control	Processing	Example
State-controlled (Enable)	Terminating (Done)	Permanent reading of a parameter. (If the reading is completed, the FB has completed its task. It repeats this task as long as the control input (Enable) remains set, e.g. MB_ReadParameter.
State-controlled (Enable)	Permanent (In….)	Terminable control. (If the control is active, it processes until the control input (Enable) is reset and thus deactivates the control.)
Edge-controlled (Execute)	Terminating (Done)	One-time writing of a parameter. (If the writing is completed, the FB has completed its task. A new edge is required at the control input (Execute) in order to start new writing.) e.g. MB_WriteParameter
Edge-controlled (Execute)	Permanent (In….)	Non-terminable control. (Once the control is started via the control input (Execute), it cannot be deactivated via the interfaces of this FB anymore.) e.g. MC_MoveVelocity

Fig.11-2: Overview on FB types

11.2.3 Function Block Types and their Interfaces

Inputs

In order to mark whether a function block processes state-controlled or edge-controlled, two different variable names are used for the inputs for the activation of the function block.

- Enable = state-controlled
- Execute = edge-controlled

Outputs

In order to mark whether a function block processes in a terminating way or permanent, different variable names are used for the outputs which indicate the processing state.

☞

- Done = Terminating processing
- In….. = continuous processing (e.g., InOperation, InSync, InVelocity, InGear, InTorque…)

The optional "Active" output can be used with terminating processing and permanent processing FBs behaving differently!

All outputs for the error identification (Error, ErrorID, ErrorIdent) can be found with terminating processing and permanent processing FBs behaving identically.

Standardized identifiers

Identifier	Description
Inputs	
Enable	Enable input of state-controlled FBs
Execute	Enable input of edge-controlled FBs. Other FBs can interrupt this.
ExecuteLock	Enable input of edge-controlled FBs. As long as this input is TRUE, the FB cannot be interrupted by others.
Outputs	
Active	Output marking the processing time

Identifier	Description
Done	Processing completed successfully and the data outputs are valid.
In……	Output signalizes that the FB is processing its task and that the data outputs are valid.
Shutdown (optional)	The output signalizes that the FB is currently reaching a defined final state (e.g.,it stops axes, releases resources, …). The FB must be called as long as necessary for "Shutdown" to become FALSE.
CommandAborted	Output indicates that the FB was interrupted (e.g. by another FB).
Error	Process completed with error
ErrorID	Output for rough error classification
ErrorIdent	Output for detailed error classification

Fig.11-3: Overview on standardized identifiers

11.2.4 Inputs and Outputs of State-controlled Function Blocks, Terminating Processing

Control/ Processing	I/O	Variable name	Description
State-controlled/terminating	E	Enable	The input variables are registered with a positive edge at "Enable". New input values do not become effective until there is the next positive edge at "Enable". "Enable" has to be TRUE as long as the FB is processed! If "Enable" is deleted, processing is interrupted and the "Done", "Active", "CommandAborted" and "Error" outputs are set to FALSE. If required, "Shutdown" signalizes that further signals are necessary for the FB to reach its final state defined.
	A	Done	If "Done" is TRUE, the function block has successfully completed its order and is then in a final state. Data outputs are valid now. "Active", "Error" and "CommandAborted" are FALSE! "Done" remains TRUE for exactly one PLC cycle. If "Enable" is still TRUE, the FB starts processing again.
	A	Active	If "Active" is TRUE, the function block is processing its actual task and is then in an intermediary state. Possible preprocessing are not marked with this output! "Done", "Error" and "CommandAborted" are FALSE!
	A	Shutdown (optional)	If a "Shutdown" output is available, the FB needs several cycles of calculation time when "Enable" is deactivated in order to reach a defined final state and to release the resources used. The FB must be called as long as necessary for "Shutdown" to become FALSE.
	A	CommandAborted	If "CommandAborted" is TRUE, the function block has been interrupted and is now in a final state. "Done", "Active" and "Error" are FALSE! "CommandAborted" remains TRUE until the control input "Enable" is deleted. An additional reset input is not necessary. If a "Shutdown" output is available, it becomes simultaneously TRUE with "CommandAborted“. In this case, the FB must be called as long as necessary for "Shutdown" to become FALSE.
	A	Error	If "Error" is TRUE, the function block has been interrupted due to an error and is now in a final state. "Error" remains TRUE until control input "Enable" is deleted. An additional reset input is not necessary. If a "Shutdown“ output is available, it becomes simultaneously TRUE with "Error“. In this case, the FB must be called as long as necessary for "Shutdown" to become FALSE.

Fig.11-4: I/O of state-controlled function blocks, terminating processing

☞ If specific inputs must be applied not only with an edge at "Enable" but cyclically as long as the function block is processed, this must be explicitly documented; it is not allowed to specify this by means of variable names.

☞ If a function block is provided with "Shutdown", the behavior of this FB when "Enable" is set to TRUE while "Shutdown" remains set (retrigger), must be precisely defined.

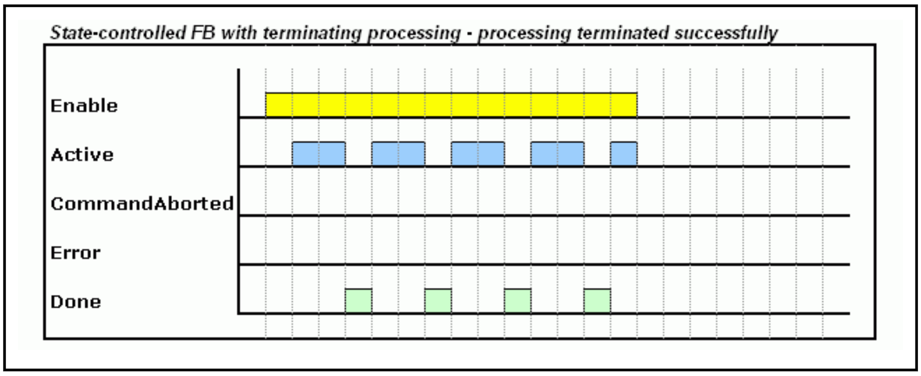


Fig. 11-5: I/O of state-controlled function blocks, definitive processing without "Shutdown". Processing successfully completed

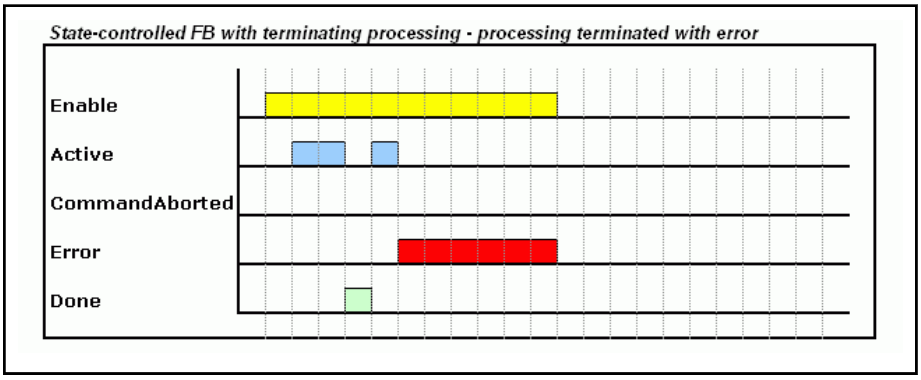


Fig. 11-6: I/O of state-controlled function blocks, definitive processing without "Shutdown". Processing completed with error

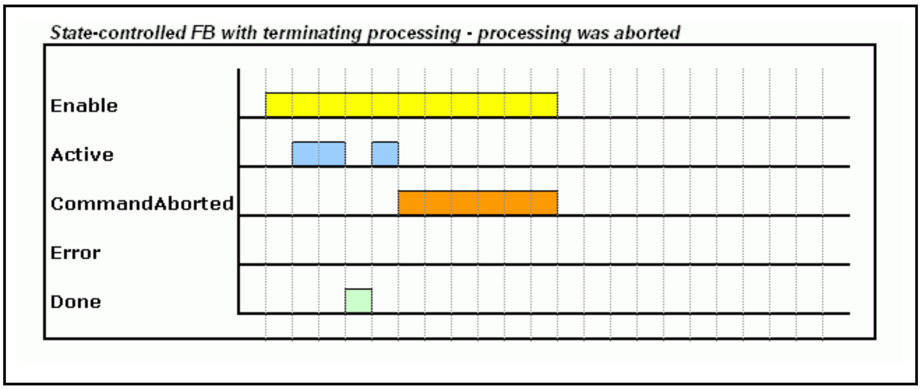


Fig. 11-7: I/O of state-controlled function blocks, definitive processing without "Shutdown". Processing interrupted

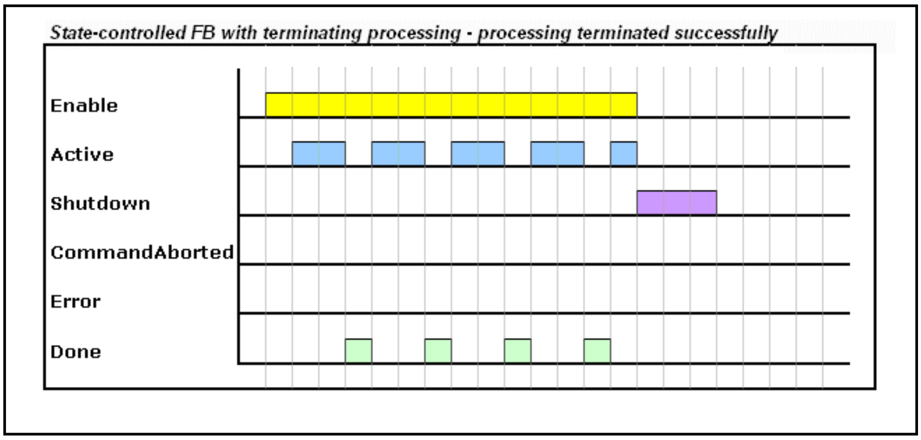


Fig.11-8: I/O of state-controlled function blocks, definitive processing with "Shutdown". Processing successfully completed.

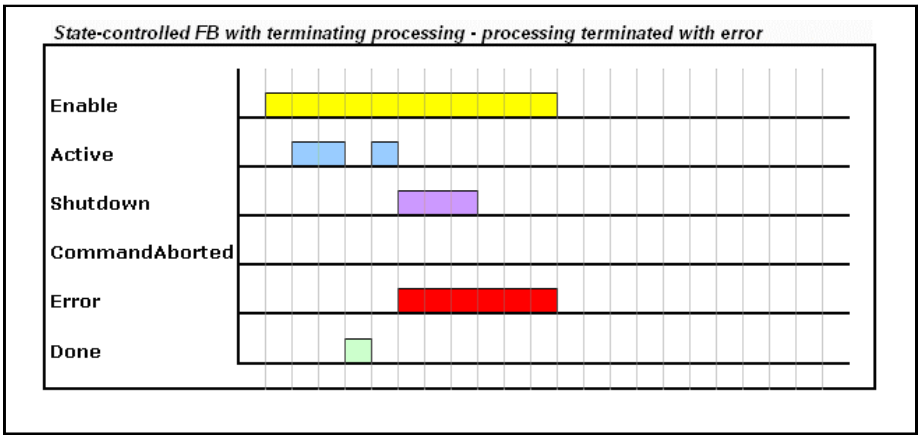


Fig.11-9: I/O of state-controlled function blocks, definitive processing with "Shutdown". Processing completed with errors.

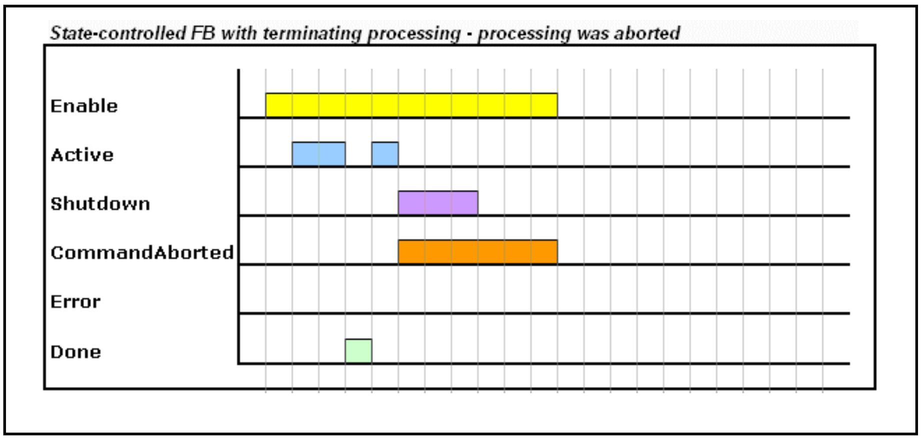


Fig.11-10: I/O of state-controlled function blocks, definitive processing with "Shutdown". Processing interrupted.

11.2.5 Inputs and Outputs of Edge-controlled Function Blocks, Definitive Processing

Control/ Processing	I/O	Name	Description
Edge-controlled/terminating	E	Execute	The input variables are registered with a positive edge at "Execute". New input values do not become effective until there is the next positive edge at "Execute". The change of edges at "Execute" is sufficient to start the FB. The state of "Execute" is irrelevant for further processing. With a new edge change during the processing, the previous task is discarded, the inputs are taken over again and the task is continued with the new values (retriggering, subsequent triggering). If a "Shutdown“ output is available, retriggering is perhaps not supported. The FB documentation brings clarity.
	A	Done	If "Done" is TRUE, the function block has successfully completed its order and is then in a final state. Data outputs are valid now. "Active", "Error" and "CommandAborted" are FALSE! If control input "Execute" is FALSE when the order is completed, "Done" remains TRUE for exactly one PLC cycle. If "Execute" is TRUE, "Done" remains TRUE until "Execute" is deleted. If a "Shutdown" output is available, it becomes simultaneously TRUE with "Done“. In this case, the FB must be called as long as necessary for "Shutdown" to become FALSE.
	A	Active	If "Active" is TRUE, the function block is processing its actual task and is then in an intermediary state. Possible preprocessing are not marked with this output! "Done", "Error", "Shutdown" and "CommandAborted" are FALSE!
	A	Shutdown (optional)	If a "Shutdown" output is available, the FB needs several cycles of calculation time after having completed its task (signalized via "Done") in order to reach a defined final state and to release the resources used. The FB must be called as long as necessary for "Shutdown" to become FALSE.
	A	CommandAborted	If "CommandAborted" is TRUE, the function block has been interrupted and is now in a final state. "Done", "Active" and "Error" are FALSE! If control input "Execute" is FALSE when the order is completed, "CommandAborted" remains TRUE for exactly one PLC cycle. If "Execute" is TRUE, "CommandAborted" remains TRUE until "Execute" is deleted. If a "Shutdown“ output is available, it becomes simultaneously TRUE with "CommandAborted“. In this case, the FB must be called as long as necessary for "Shutdown" to become FALSE.
	A	Error	If "Error" is TRUE, the function block has been interrupted due to an error and is now in a final state. If control input "Execute" is FALSE when the error occurs, "Error" remains TRUE for exactly one PLC cycle. If "Execute" is TRUE, "Error" remains TRUE until "Execute" is deleted. If a "Shutdown“ output is available, it becomes simultaneously TRUE with "Error“. In this case, the FB must be called as long as necessary for "Shutdown" to become FALSE.

Fig.11-11: I/O of state-controlled function blocks, terminating processing

☞ If certain inputs are applied not only with the 0/1 edge at "Execute", but cyclically as long as the FB is processed, this must be explicitly documented; it is not allowed to specify this by means of variable names!

☞ If a function block is provided with "Shutdown", the behavior of this FB when an edge is redetected at "Execute" while "Shutdown" remains set (retrigger), must be precisely defined.

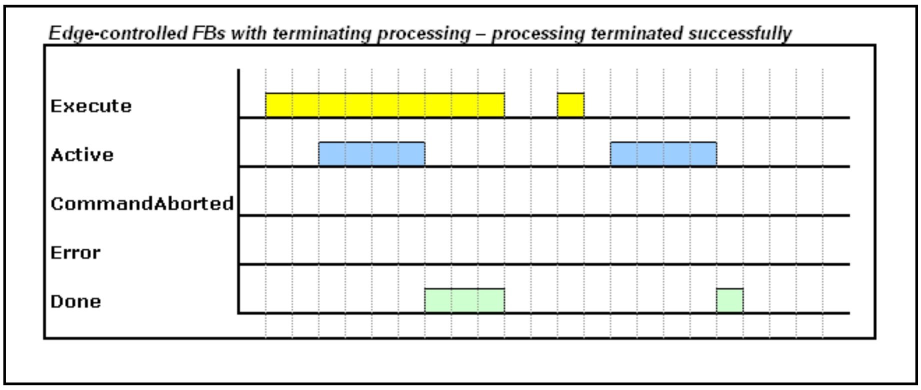

Edge-controlled FBs with terminating processing – processing terminated successfully
Execute
Active
CommandAborted
Error
Done

Fig.11-12: I/O of edge-controlled function blocks, definitive processing without "Shutdown". Processing successfully completed

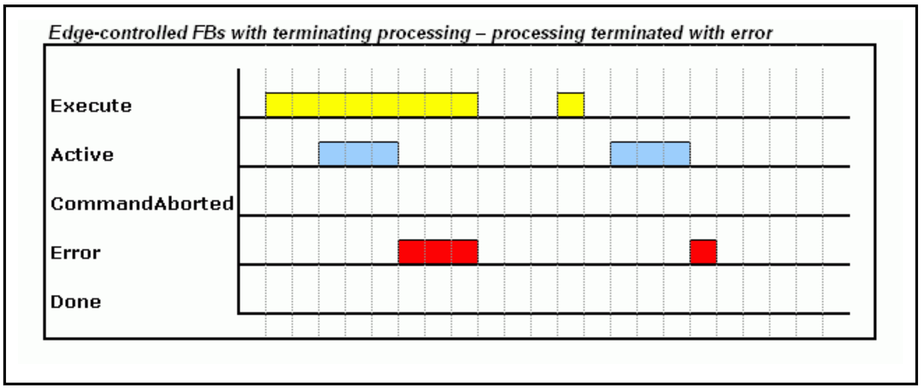

Edge-controlled FBs with terminating processing – processing terminated with error
Execute
Active
CommandAborted
Error
Done

Fig.11-13: I/O of edge-controlled function blocks, definitive processing without "Shutdown". Processing completed with error

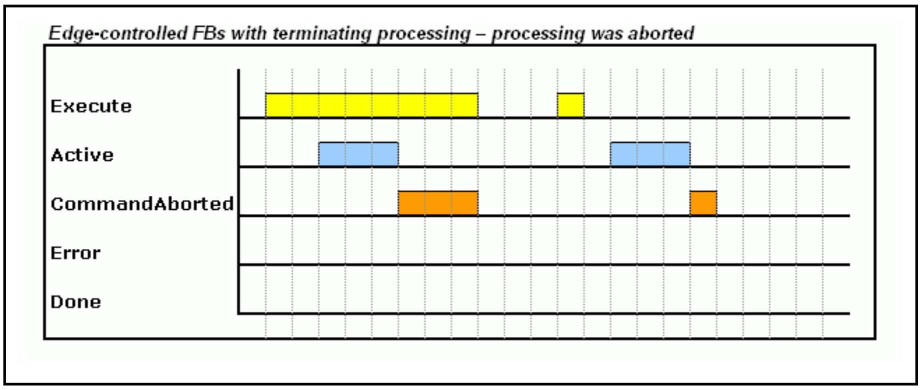

Edge-controlled FBs with terminating processing – processing was aborted
Execute
Active
CommandAborted
Error
Done

Fig.11-14: I/O of edge-controlled function blocks, definitive processing without "Shutdown". Processing interrupted

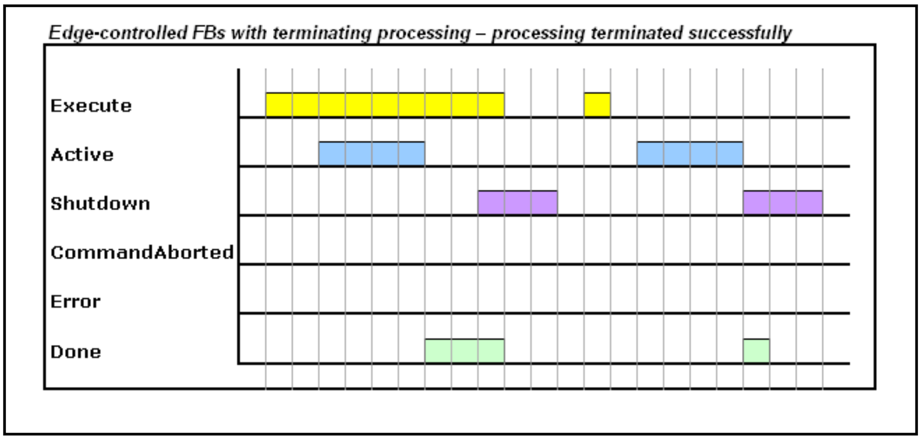


Fig. 11-15: I/O of edge-controlled function blocks, definitive processing with "Shutdown". Processing successfully completed

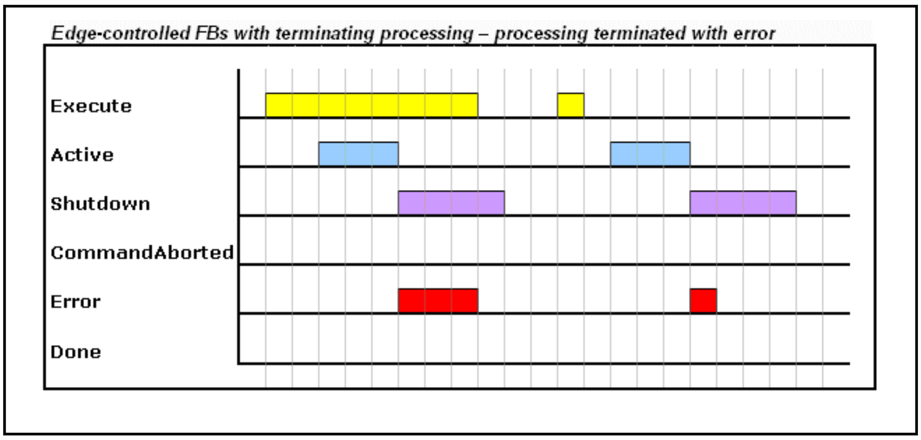


Fig. 11-16: I/O of edge-controlled function blocks, definitive processing with "Shutdown". Processing completed with error

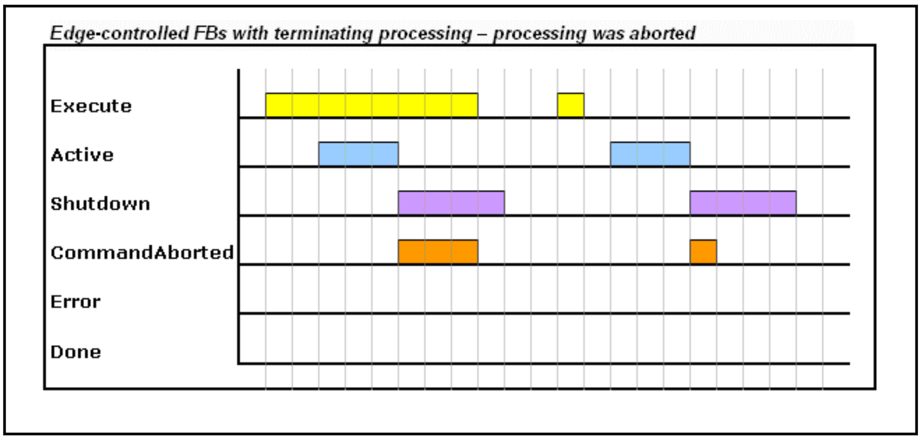


Fig. 11-17: I/O of edge-controlled function blocks, definitive processing with "Shutdown". Processing interrupted

11.2.6 Inputs and Outputs of State-controlled Function Blocks, Permanent Processing

Control/ Processing	I/O	Variable name	Description
State-controlled/ permanent	E	Enable	The input variables are registered with a positive edge at "Enable". New input values do not become effective until there is the next positive edge at "Enable". "Enable" has to be TRUE as long as the FB is processed! If "Enable" is deleted, processing is interrupted and the "In...", "Active", "CommandAborted" and "Error" outputs are set to FALSE. If required, "Shutdown" signalizes that further signals are necessary for the FB to reach its final state defined.
	A	In….	If "In…." is TRUE, the function block has reached its goal but is still in engagement in order to "keep" what it has achieved and is thus in a continuous final state. Data outputs are valid now. Since the FB remains in engagement,"Active" must also remain TRUE. "Error" and "CommandAborted" are FALSE! As long as "Enable" is TRUE, "In...." remains TRUE as well.
	A	Active	If "Active" is TRUE, the function block processes its actual task. Possible preprocessing are not marked with this output! Since the FB remains in continuous engagement, "Active" remains TRUE until the FB is switched off via control input "Enable" or until it is completed by "Error" or "CommandAborted". As long as "Active" is TRUE, "Error" or "CommandAborted" must be FALSE.
	A	Shutdown (optional)	If a "Shutdown" output is available, the FB needs several cycles of calculation time when "Enable" is deactivated in order to reach a defined final state and to release the resources used. The FB must be called as long as necessary for "Shutdown" to become FALSE.
	A	CommandAborted	If "CommandAborted" is TRUE, the function block has been interrupted and is now in a final state. "In...", "Active" and "Error" are FALSE! "CommandAborted" remains TRUE until control input "Enable" is deleted. An additional reset input is not necessary. If a "Shutdown" output is available, it becomes simultaneously TRUE with "CommandAborted“. In this case, the FB must be called as long as necessary for "Shutdown" to become FALSE.
	A	Error	If "Error" is TRUE, the function block has been interrupted due to an error and is now in a final state. "Error" remains TRUE until control input "Enable" is deleted. An additional reset input is not necessary. If a "Shutdown“ output is available, it becomes simultaneously TRUE with "Error“. In this case, the FB must be called as long as necessary for "Shutdown" to become FALSE.

Fig.11-18: I/O of state-controlled function blocks, terminating processing

☞ If specific inputs must be applied not only with an edge at "Enable" but cyclically as long as the function block is processed, this must be explicitly documented; it is not allowed to specify this by means of variable names.

☞ If a function block is provided with "Shutdown", the behavior of this FB when "Enable" is set to TRUE while "Shutdown" remains set (retrigger), must be precisely defined.

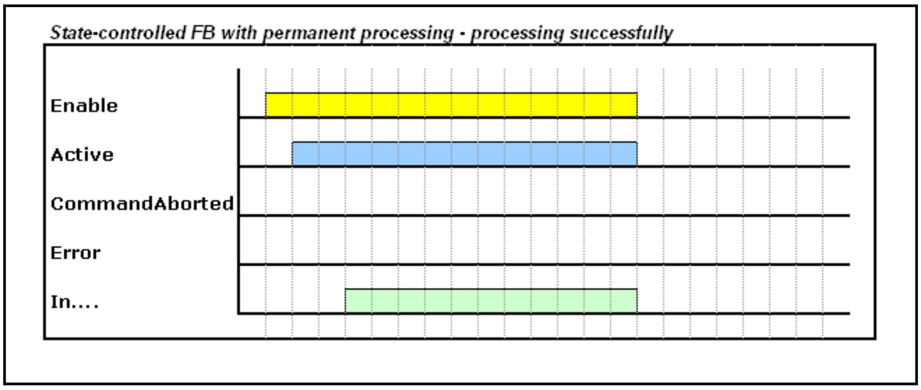

State-controlled FB with permanent processing - processing successfully
Enable
Active
CommandAborted
Error
In....

Fig.11-19: I/O of state-controlled function blocks, continuous processing without "Shutdown". Processing successful

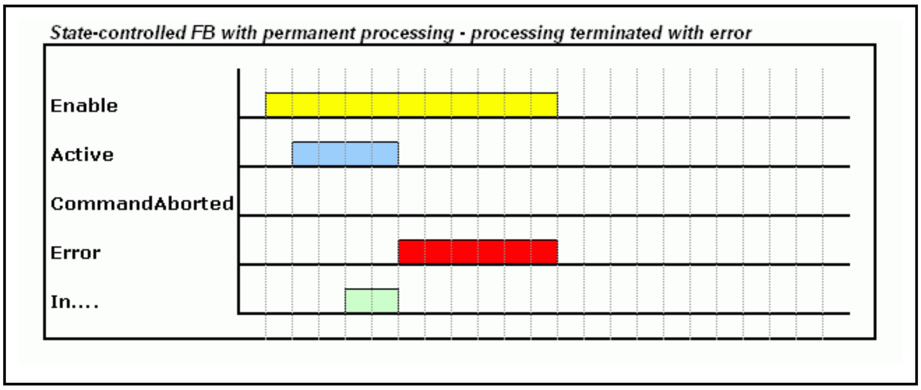

State-controlled FB with permanent processing - processing terminated with error
Enable
Active
CommandAborted
Error
In....

Fig.11-20: I/O of state-controlled function blocks, continuous processing without "Shutdown". Processing aborted with errors.

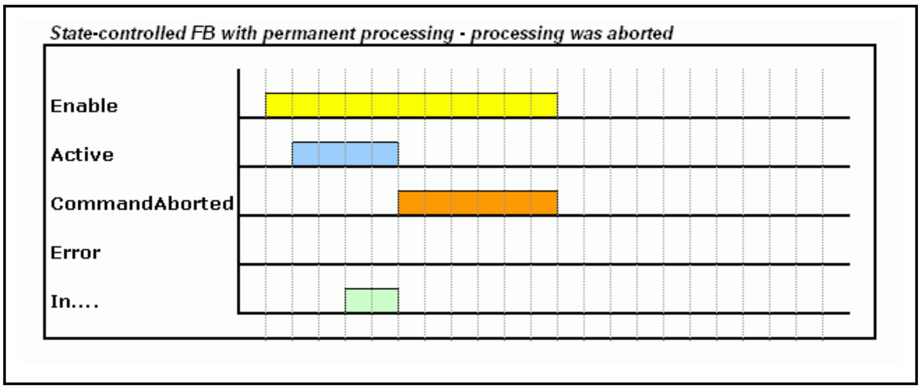

State-controlled FB with permanent processing - processing was aborted
Enable
Active
CommandAborted
Error
In....

Fig.11-21: I/O of state-controlled function blocks, continuous processing without "Shutdown". Processing interrupted

Field Bus Libraries

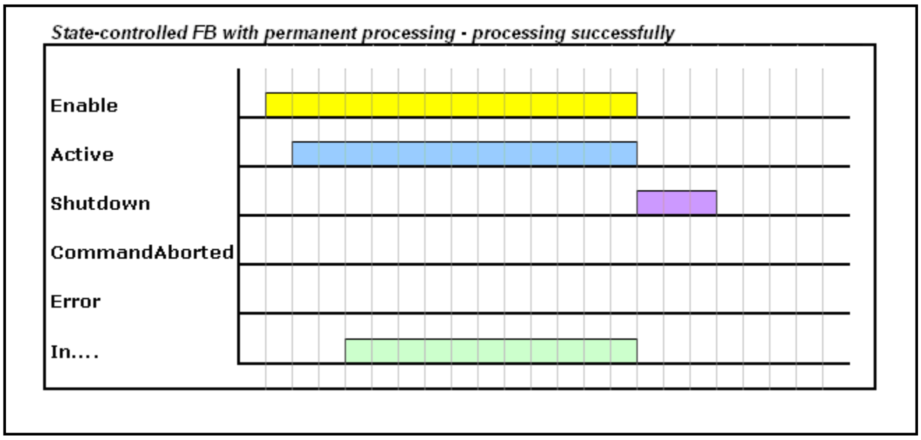


Fig.11-22: I/O of state-controlled function blocks, continuous processing with "Shutdown". Processing successful

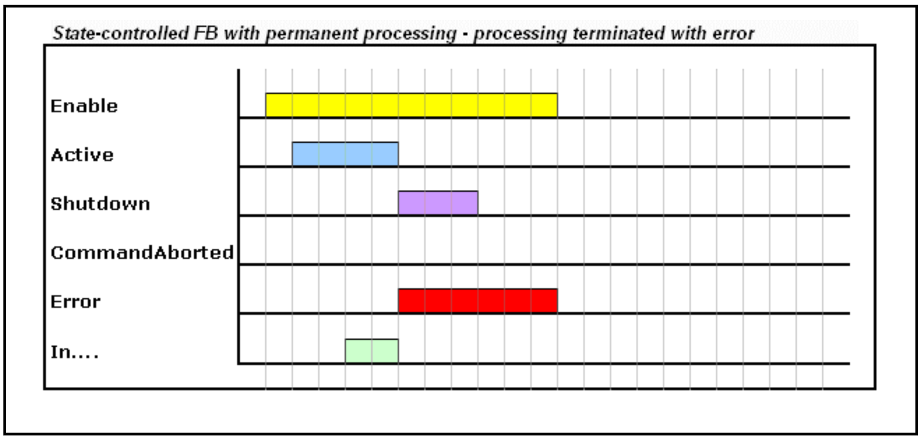


Fig.11-23: I/O of state-controlled function blocks, continuous processing with "Shutdown". Processing aborted with errors.

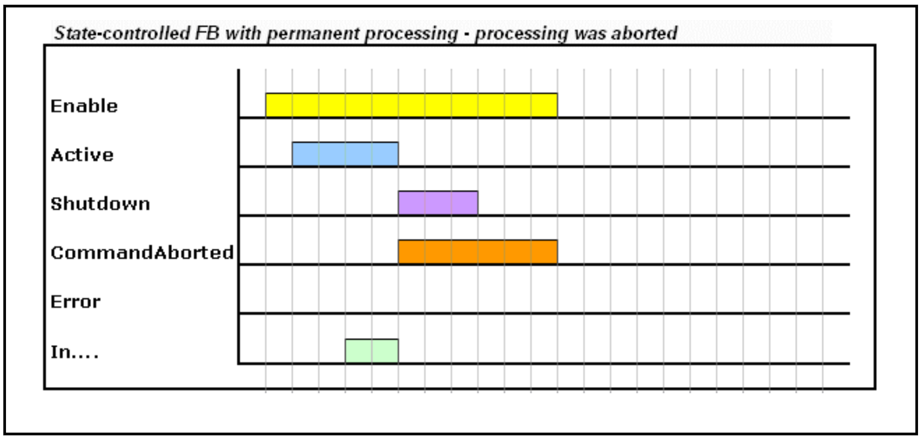


Fig.11-24: I/O of state-controlled function blocks, continuous processing with "Shutdown". Processing interrupted

11.2.7 Inputs and Outputs of Edge-controlled Function Blocks, Permanent Processing

Control/ Processing	I/O	Variable name	Description
Edge-controlled/ permanent	E	Execute	The input variables are registered with a positive edge at "Execute". New input values do not become effective until there is the next positive edge at "Execute". The change of edges at "Execute" is sufficient to start the FB. The state of "Execute" is irrelevant for further processing. With a new edge change during the processing, the previous task is discarded, the inputs are taken over again and the task is continued with the new values (retriggering, subsequent triggering). If the FB is provided with a "Shutdown“ output, retriggering is perhaps not allowed.
	A	In….	If "In…." is TRUE, the function block has reached its goal but is still in engagement in order to "keep" what it has achieved and is thus in a continuous final state. Data outputs are valid now. Since the FB remains in engagement,"Active" must also remain TRUE. This even applies when "In...." drops again. "Error" and "CommandAborted" are FALSE! If control input "Execute" is FALSE when the order is completed, "In..." remains TRUE for exactly one PLC cycle. If "Execute" is TRUE, "In..." remains TRUE until "Execute" is deleted.
	A	Active	If "Active" is TRUE, the function block processes its actual task. Possible preprocessing are not marked with this output! Since the FB remains in continuous engagement, "Active" remains TRUE until the FB is completed by "Error" or "CommandAborted". As long as "Active" is TRUE, "Error" or "CommandAborted" as well as "Shutdown" must be FALSE.
	A	Shutdown (optional)	If a "Shutdown" output is available, the FB needs several cycles of calculation time after having aborted its task ("CommandAborted" or "Error" = TRUE) in order to reach a defined final state and to release the resources used. The FB must be called as long as necessary for "Shutdown" to become FALSE.
	A	CommandAborted	If "CommandAborted" is TRUE, the function block has been interrupted and is now in a final state. "In...", "Active" and "Error" are FALSE! If control input "Execute" is FALSE when the order is completed, "CommandAborted" remains TRUE for exactly one PLC cycle. If "Execute" is TRUE, "CommandAborted" remains TRUE until "Execute" is deleted. If a "Shutdown" output is available, it becomes simultaneously TRUE with "CommandAborted“. In this case, the FB must be called as long as necessary for "Shutdown" to become FALSE.
	A	Error	If "Error" is TRUE, the function block has been interrupted due to an error and is now in a final state. If control input "Execute" is FALSE when the error occurs, "Error" remains TRUE for exactly one PLC cycle. If "Execute" is TRUE, "Error" remains TRUE until "Execute" is deleted. If a "Shutdown“ output is available, it becomes simultaneously TRUE with "Error“. In this case, the FB must be called as long as necessary for "Shutdown" to become FALSE.

Fig.11-25: I/O of edge-controlled function blocks, permanent processing

☞ If certain inputs are applied not only with the 0/1 edge at "Execute", but cyclically as long as the FB is processed, this must be explicitly documented; it is not allowed to specify this by means of variable names!

☞ If a function block is provided with "Shutdown", the behavior of this FB when an edge is redetected at "Execute" while "Shutdown" remains set (retrigger), must be precisely defined.

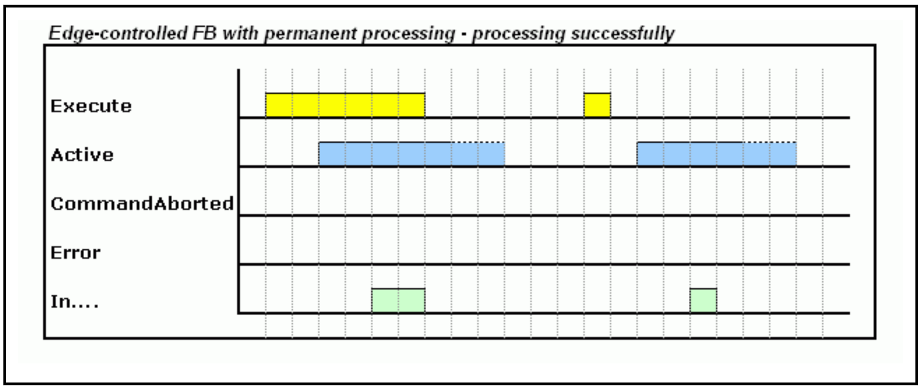


Fig.11-26: I/O of edge-controlled function blocks, continuous processing without "Shutdown". Processing successful

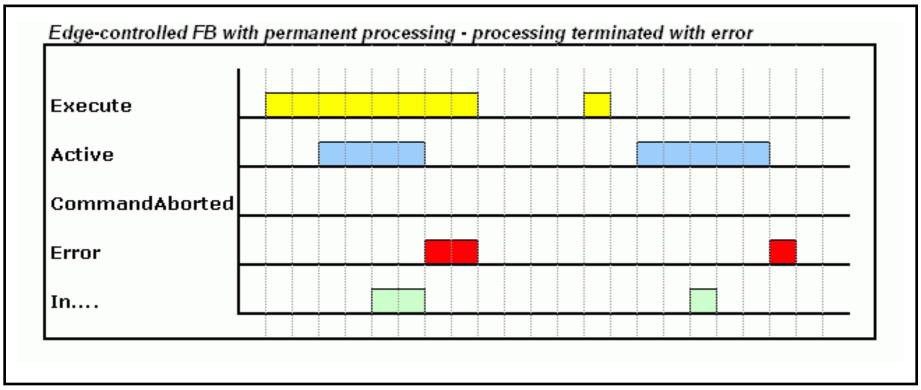


Fig.11-27: I/O of edge-controlled function blocks, continuous processing without "Shutdown". Processing aborted with errors.

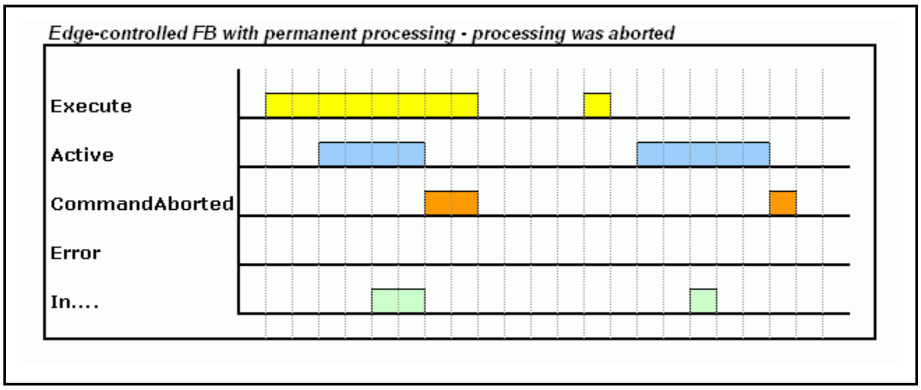


Fig.11-28: I/O of edge-controlled function blocks, continuous processing without "Shutdown". Processing interrupted

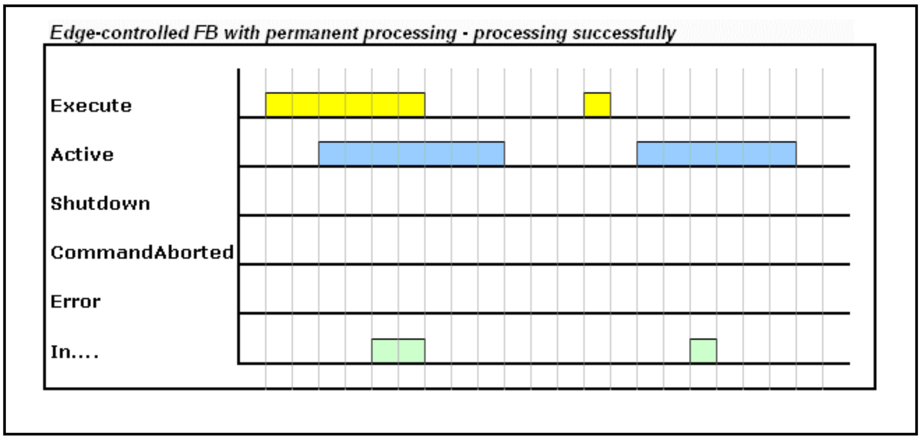


Fig.11-29: I/O of edge-controlled function blocks, continuous processing with "Shutdown". Processing successful

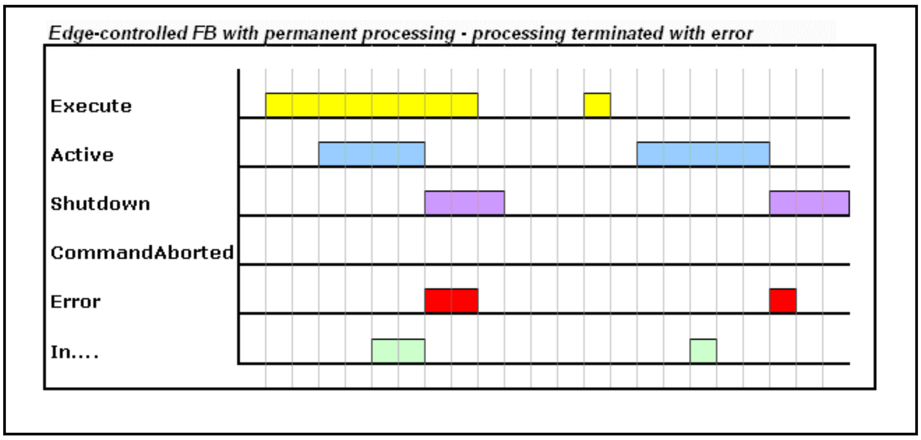


Fig.11-30: I/O of edge-controlled function blocks, continuous processing with "Shutdown". Processing aborted with errors.

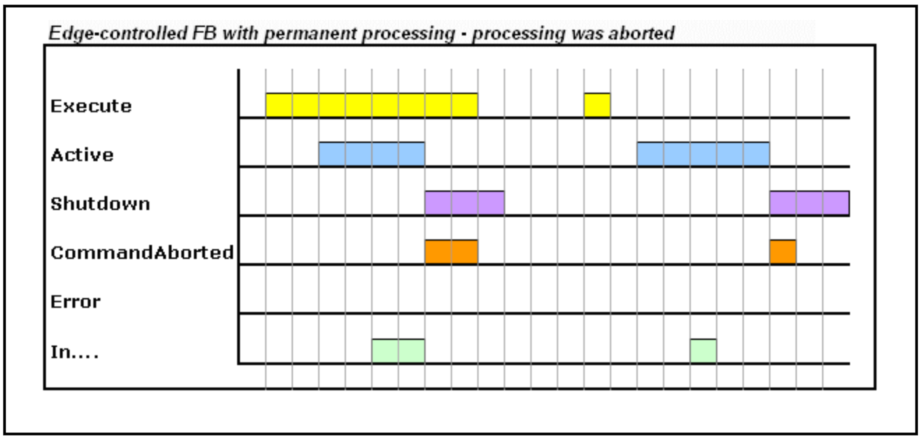


Fig.11-31: I/O of edge-controlled function blocks, continuous processing with "Shutdown". Processing interrupted

11.3 RIL_ProfibusDP_02.library

11.3.1 Overview

History

When using the function module DP Master and the existing onboard master, there are controls available for implementing systems with several DP Masters.

To do this, the RIL_ProfibusDP_02.library is created, which can distinguish multiple DP Masters.

RIL_ProfibusDP.lib version 01V01 is not suitable for being on systems with several DP masters. This library is extended to version 01V02 to ensure compatibility.

Overview of the function blocks and functions contained in the library

- FB IL_DPMasterState, page 166, determines the state of a local bus master as well as of the configured slaves;
- IL_DPBaudrateGet, page 176, reads the current baud rate;
- IL_DPDeviceListGet, page 175, list of active DP devices in the system;
- IL_DPDevInfoGet, page 177, reads the bus master information structure;
- IL_DPGetCfg, in preparation
- IL_DPIdent, page 168, function for assembling Profibus IDs;
- IL_DPPrjSlaveListGet, page 178, list of configured slaves;
- IL_DPReadDiag, Seite 172, reads the diagnostic data of a slave according to DP standard;
- IL_DPSlaveDiagListGet, Seite 179, list of current slave diagnostics;
- IL_DPSycFr, page 173, synchronizes/freezes the control command;
- IL_DPV1Read, page 169, reading V1 service;
- IL_DPV1Write, page 170, writing V1 service.

11.3.2 Data types

IL_BUSMASTER

Brief Description

The IL_BUSMASTER enumeration type of the RIL_FieldbusTypes.library serves to select the appropriate PROFIBUS DP master. In addition to the onboard real-time Ethernet interface, other controllers can also be available in an XLC/MLC/MTX system via function modules.

The PROFIBUS DP masters are distinguished on the basis of their ascending order in the configuration.

Indexes are assigned to the masters in the following order:

- Onboard master
- Function module 1
- Function module 2,
- Function module 3,
- Function module 4

☞ If, for example, the onboard master is not a Profibus master while a Profibus master is configured on the FM1, IL_BUSMASTER_0 is the value assigned to the master on the FM1.

Element	Value	Description
IL_BUSMASTER_0	0	First device
IL_BUSMASTER_1	1	Second device
IL_BUSMASTER_2	2	Third device

Element	Value	Description
IL_BUSMASTER_3	3	Fourth device
IL_BUSMASTER_4	4	Fifth device

Fig.11-32: IL_BUSMASTER enumeration type

Examples of addressing the various instances of the Profibus masters on a control:

Configured:	Addressed with:
Onboard: Profibus DP master	IL_BUSMASTER_0
FM1: Profibus DP slave	IL_BUSSLAVE_0
FM2: Profibus DP master	IL_BUSMASTER_1
FM3: Profibus DP master	IL_BUSMASTER_2

Slave Diagnostic Data according to Profibus DP Standard

Overview

Structure of Profibus-specific diagnostic information according to Profibus DP standard.

Slave diagnostic data are subdivided as follows:

- General part with a defined length of 6 bytes
- Extended diagnostics (slave-specific with variable length)

Offset	Type	Designation	Description
0	BYTE	Stationsstatus_1	See below
1	BYTE	Stationsstatus_2	See below
2	BYTE	Stationsstatus_3	See below
3	BYTE	Master_Add	Bus address of the master having parameterized the slave
4	WORD	Ident_Number	Ident_Number of the slave
6-243		Ext_Diag_Data	Extended diagnostics according to Profibus DP standard

Fig.11-33: Slave diagnostic data

Station state

The following description of station states 1 to 3 is an excerpt from the Profibus DP standard.

Bit	Designation	Description
7	Master_Lock	The DP slave has been parameterized by a different master. This bit is set by the DP master (class 1) if the address in octet 4 is unequal to 255 and unequal to its own address. The DP slave sets this bit to a defined value of zero.
6	Prm_Fault	This bit is set by the DP slave if the most recent parameter telegram contained an error, e.g., incorrect length, incorrect Ident_Number, invalid parameters.
5	Invalid_Slave_Response	This bit is set by the DP master as soon as an addressed DP slave receives an implausible answer. The DP slave sets this bit to a defined value of zero.

Bit	Designation	Description
4	Not_Supported	This bit is set by the DP slave as soon as a function has been requested which is not supported by this DP slave.
3	Ext_Diag	This bit is set by the DP slave. If the bit is set, a diagnostic entry must be present in the slave-specific diagnostic area (Ext_Diag_Data). If the bit is not set, a status message may be present in the slave-specific diagnostic area (Ext_Diag_Data). The meaning of this status message must be defined according to the specific application.
2	Cfg_Fault	This bit is set by the DP slave as soon as the configuration data which was the last to be received by the DP master does not correspond to the data determined by the DP slave.
1	Station_Not_Ready	This bit is set by the DP slave when the DP slave is not ready for data exchange yet.
0	Station_Non_Existent	This bit is set by the DP master if this DP slave cannot be reached via the bus. If this bit is set, the diagnostic bits contain the state of the previous diagnostic message or the initial value. The DP slave sets this bit to a defined value of zero.

Fig.11-34: Stationsstatus_1

Bit	Designation	Description
7	Deactivated	This bit is set by the DP master as soon as the DP slave has been labeled as inactive in the DP slave parameter set and has been removed from cyclic processing. The DP slave always sets this bit to zero.
6	reserved	-
5	Sync_Mode	This bit is set by the DP slave as soon as it has received the Sync control command. A change in these bits does not cause a diagnostic message. For this reason, these bits usually do not reflect the current state.
4	Freeze_Mode	This bit is set by the DP slave as soon as it has received the Freeze control command. A change in these bits does not cause a diagnostic message. For this reason, these bits usually do not reflect the current state.
3	WD_On (watchdog)	This bit is set by the DP slave as soon as its response monitoring is activated.
2	1	This bit is set by the DP slave to a defined value of 1.

Bit	Designation	Description
1	Stat_Diag (static diagnostics)	If the DP slave sets this bit, the DP master must retrieve diagnostic information until this bit is cleared again. For example, the DP slave sends this message if it cannot provide any valid user data.
0	Prm_Req	If the DP slaves sets this bit, the bit has to be parameterized and configured again. The bit remains set until parameterization has been completed. This bit is set by the DP slave.

Fig. 11-35: Stationsstatus_2

Bit	Designation	Description
7	Ext_Diag_Overflow	If this bit is set, the quantity of diagnostic information exceeds that specified in Ext_Diag_Data. For example, the DP slave sets this bit if the number of channel diagnostic messages is higher than that which the DP slave can enter in its transmission buffer; of the DP master sets this bit if the quantity of diagnostic information sent by the DP slave exceeds that which the DP master can accept to its diagnostic buffer.
6	reserved	-
5	reserved	-
4	reserved	-
3	reserved	-
2	reserved	-
1	reserved	-
0	reserved	-

Fig. 11-36: Stationsstatus_3

Bus Master Status Word

The bus master status word "BmState" provides an overview of the state of the bus master and the slaves at the field bus. For example, it specifies whether diagnosis is applied to at least one slave. Each bit that is set (TRUE) in "BmState" represents a status:

Bit	Status	Description
0	BMS_BMF	Bus master error: This bit indicates that there is a bus master error. In this case, the bus master error word contains more detailed information.
1	BMS_KSD	Classified slave diagnostics If this bit is set, at least one slave signals classified diagnostics. Bits 8 to 13 indicate the classified diagnostic message(s) that is/are set.
2	BMS_SD	Slave diagnostics: If this bit is set, at least one slave signals diagnostics.
3	-	reserved

Bit	Status	Description
4	-	reserved
5	-	reserved
6	-	reserved
7	BMS_AKTIV	Active detection: This bit must always have the value 1. If this is not the case, there is a fatal error in the software of the bus master.
8	BMS_SNE	One or more slaves cannot be reached via the bus.
9	BMS_SKF	One or more slaves signal configuration errors.
10	BMS_DPS	One or more slaves signal static diagnostics.
11	BMS_EXD	One or more slaves signal extended diagnostics.
12	BMS_SNB	One or more slaves are not ready for cyclic data exchange.
13	BMS_SF	One or more slaves signal a miscellaneous error.
14	-	reserved
15	-	reserved

Fig. 11-37: Status coding in "BmState"

Bus Master Error Word

The "BmError" bus master error word indicates serious errors which prevent operation at the field bus. Each bit that is set (TRUE) in "BmError" represents an error:

Bit	Error[1]	Description
0	IL_BMF_HW_ERR	Hardware fault
1	IL_BMF_MPS_ERR	Master parameter set (field bus configuration file) missing or defective
2	IL_BMF_BUS_ERR	Error at field bus (e.g., short-circuit)
3	IL_BMF_SW_ERR	System error in periphery driver (i.e., the driver software has detected a serious error)

Fig. 11-38: Error coding in "BmError"

DP_SLAVELIST, Array

Brief Description

The "DP_SLAVELIST" bit list (DP_BITLIST) has a defined length of 16 bytes (128 bits).

Type Declaration

```
TYPE  DP_SLAVELIST : ARRAY  [0..15] OF BYTE; END_TYPE
```

Bit List Coding

Each bit of the bit list is assigned to a bus address of the slave (Profibus: FDL address). For example, the lowest-order bit in the first array element (ARRAY[0]) is assigned to the Profibus user with address 0:

[1] *"IL_BMF_OK" indicates that there is no error*

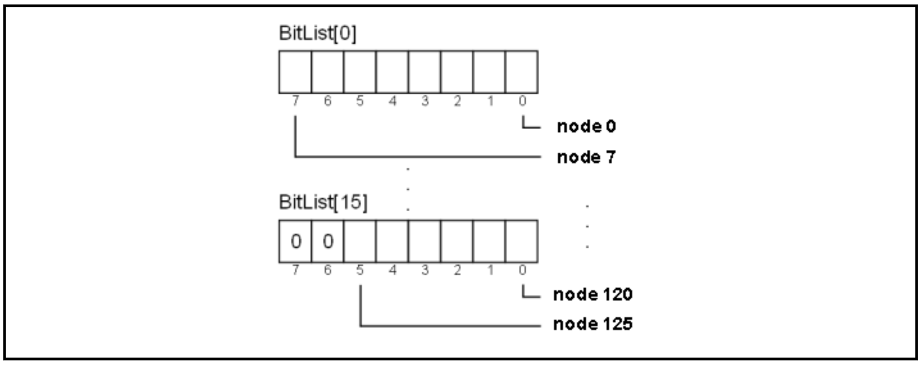


Fig. 11-39: Bit list coding

DP_DEVICELIST, Structure

Brief Description This data type comprises information with regard to a DP master:

Example *Program:*

```
TYPE  DP_DEVICELIST:
 STRUCT
 bMasterAdr         :  BYTE;  (* master addressing *)
 bMasterBusAdr      :  BYTE;  (* master bus address *)
 wMasterError       :  WORD;  (* cf. IL_DPDevInfoGet *)
 wMasterState       :  WORD;  (* cf. IL_DPDevInfoGet *)
 udBaudrate         :  UDINT; (* cf. IL_DPBaudrateGet *)
 dFirmwareVersion   :  DINT;  (* driver firmware version *)
 dHardwareVersion   :  DINT;  (* hardware version *)
 dAddInfo1          :  DINT;  (* 3S module identification *)
 dAddInfo2          :  DINT;  (* res *)
 END_STRUCT
END_TYPE

TYPE  DP_MASTERLIST: ARRAY [0..5] of DP_DEVICELIST;
  (* list for 6 masters *)
END_TYPE
```

Note: Any possible onboard slave is also included in this list.

DP_MASTERLIST, Array

Brief Description This data type comprises information with regard to 6 DP masters in maximum:

Beispiel *Program:*

```
TYPE  DP_DEVICELIST:
 STRUCT
 bMasterAdr         :  BYTE;  (* master addressing *)
 bMasterBusAdr      :  BYTE;  (* master bus address *)
 wMasterError       :  WORD;  (* cf. IL_DPDevInfoGet *)
 wMasterState       :  WORD;  (* cf. IL_DPDevInfoGet *)
 udBaudrate         :  UDINT; (* cf. IL_DPBaudrateGet *)
 dFirmwareVersion   :  DINT;  (* driver firmware version *)
 dHardwareVersion   :  DINT;  (* hardware version *)
 dAddInfo1          :  DINT;  (* 3S module identification *)
 dAddInfo2          :  DINT;  (* res *)
 END_STRUCT
END_TYPE

TYPE  DP_MASTERLIST: ARRAY [0..5] of DP_DEVICELIST;
  (* list for 6 masters *)
END_TYPE
```

DP_MASTERINFO, Structure

Brief Description This data type comprises information about the state of the bus master.

Example *Program:*

```
TYPE  DP_MASTERINFO: (*DP_DEVICEINFO:*)
 STRUCT
 BmState    :  WORD;      (* bus master status *)
 BmError    :  WORD;      (* bus master error word *)
 END_STRUCT
END_TYPE
```

11.3.3 Selecting the DP Master

Addressing

The DP masters are distinguished on the basis of their order in the DP configuration.

0 .. n(5): number of the DP master in ascending order of the configuration. The only items counted are the DP masters.

Usage

The functions and function blocks which must access a certain instance of a master receive the "Master" input parameter of type BYTE.

The functions and function blocks which comply with the Profibus Guideline 2182 have an "ID" DWORD parameter which is interpreted as a slot handle. A master selection byte is reserved in this slot handle. Function DP_SLOT can be used to generate the ID parameter.

The functions and function blocks which use the "Ident" DWORD parameter can generate this parameter using function IL_DPIdent.

Byte	Contents	Description
0	MASTER	ID of DP system: Labeling of the DP master (or of the onboard slave)
1	SEGMENT	Number of the DP segment (0)
2	STATION	Number of the DP slave (bus address)
3	SLOT	Number of the slot in the slave

Fig. 11-40: Slot handle: "ID" parameter

11.3.4 IL_DPMasterState

Brief Description

The DP master state FB is used to determine the state of a local bus master and the configured slaves. The result is displayed in the State output parameter, in bit code.

Assignment: Target system/library

Target system	Library
XLC, MLC, MTX	RIL_ProfibusDP_02.library
XLC, MLC, MTX	RIL_ProfibusDP_02.compiled-library

Fig. 11-41: Reference table of function block IL_DPMasterState

Interface Description

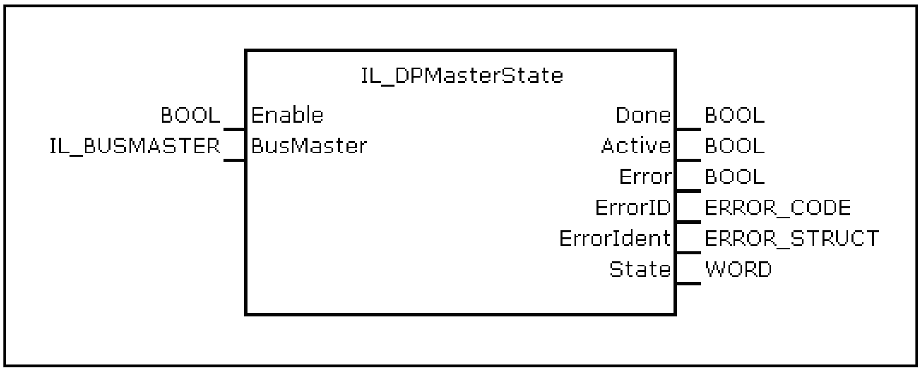


Fig. 11-42: Function block IL_DPMasterState, interface

I/O type	Name	Data type	Description
VAR_INPUT	Enable	BOOL	Processing enabled for function block (continuous, state-controlled)
	BusMaster	IL_BUSMASTER	Instance of the master in the order of the configuration tree. If there is only one Profibus DP master available, the value of this master is always IL_BUSMASTER_0. See also IL_BUSMASTER Profibus DP, page 160.
VAR_OUTPUT	Done	BOOL	Processing completed without error, output data valid
	Active	BOOL	Processing not yet completed, output data invalid
	Error	BOOL	Processing completed with error, output data invalid
	ErrorID	ERROR_CODE	Standardized rough classification of the error
	ErrorIdent	ERROR_STRUCT	Function-block-specific detailed information on the error
	State	WORD	Master diagnostic state (see elements of the State output)

Fig.11-43: FB IL_DPMasterState I/O interface

The "State" output provides a rough diagnostic classification in binary code.

The bits are contained in the RIL_FieldbusTypes.library as global "Fieldbus-States" constants.

Output "State" (WORD)

Bit no.	Description	Description
0	IL_FBM_BUS_OFF	Field bus not ready
1	IL_FBM_MASTER_FAILURE	Bus master failure
2	IL_FBM_SLAVE_NOT_REACHABLE	One or more field bus slaves cannot be reached (slave not reachable)
3	IL_FBM_SLAVE_DIAGNOSIS	One or more field bus slaves signal diagnoses (slave diag)
4	IL_FBM_SLAVE_ERROR	One or more field bus slaves signal errors (slave error)

Fig.11-44: Elements of output State

Min./max. and default values of the inputs

Name	Type	Min. value	Max. value	Default value	Effective
Enable	BOOL			FALSE	Continuous
BusMaster	IL_BUSMASTER	IL_BUSMASTER_0	IL_BUSMASTER_4	IL_BUSMASTER_0	TRUE state of enable

Fig.11-45: Min./max. and default values of the IL_DPMasterState inputs

Functional Description

- The IL_DPMasterState is called effectively and the BusMaster parameter is applied if Enable = TRUE.
- If Enable = FALSE, the call does not have any effect. In this case, all output parameters are set to '0'.
- Initially, the result of the call is shown in output parameters Done, Active and Error. If Active = FALSE, the call is completed and the other parameters must be evaluated.
- If Done = TRUE and Error = FALSE, no error occurred during the call.

- If Done = FALSE and Error = TRUE, errors occurred during the call. In this case, parameters ErrorID and ErrorIdent describe the error cause.
- After the call has been completed without errors, the effective State output parameter is valid and can be evaluated by the calling instance.
- If errors occurred during the call, the effective State output parameter is invalid.
- If the output parameter is valid, State shows the state of the bus master instance and the configured slaves in bit-encoded format.
- State = 0 indicates that the state of the bus system is error-free.
- If Error = TRUE, errors occurred during the call. In this case, ErrorID and ErrorIdent define the error cause. Output parameter State is invalid.

Error Handling

ErrorID	Additional1	Additional2	Description
INPUT_RANGE_ERROR	16#1001	BusMaster instance	Invalid BusMaster instance (value range)
DEVICE_ERROR	16#2001	BusMaster instance	Bus master instance not available
ACCESS_ERROR	16#2002	ERROR_CODE	Error during access

Fig. 11-46: Error codes of function block IL_DPMasterState

11.3.5 IL_DPIdent

Brief Description

This function assembles an ident handle from various components.

Library	Range
RIL_ProfibusDP_02	

Fig. 11-47: IL_DPIdent library assignment

Interface Description

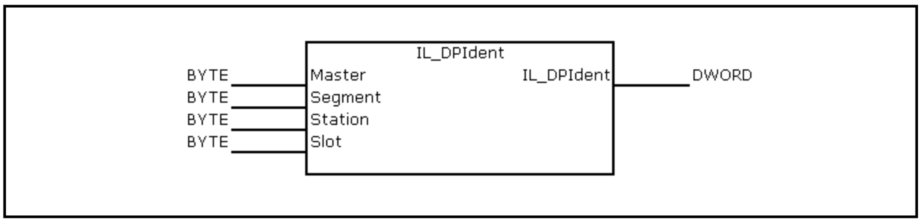


Fig. 11-48: IL_DPIdent structure

	Name	Type	Description
VAR_INPUT	Master	BYTE	DP master identification (see above)
	Segment	BYTE	Number of the DP segment (0)
	Station	BYTE	Number of the DP slave (bus address): If the command is to apply for only one special slave, then the bus address of the slave must be entered here (0..125). Applicable to IL_DPSycFr only: However, if the command is to be entered for all slaves of a group, the global address (= 127) must be entered here.

	Name	Type	Description
	Slot	BYTE	Number of the slot within the slave (according to the slave specification) (value range: 0...254).
Function value		DWORD	Ident handle

Fig. 11-49: IL_DPIdent interface

Functional Description The 4 byte values are used to form the "Ident" DWORD. This DWORD is required as an input parameter for the following function blocks.

11.3.6 IL_DPV1Read

Brief Description The "IL_DPV1Read" function block is used for DPV1 read access. Data exchange on the Profibus DP is acyclic. A pointer (POINTER) must be addressed to define a target area for the process data to be read.

Library	Area
RIL_ProfibusDP_02	

Fig. 11-50: IL_DPV1Read library assignment

Interface Description

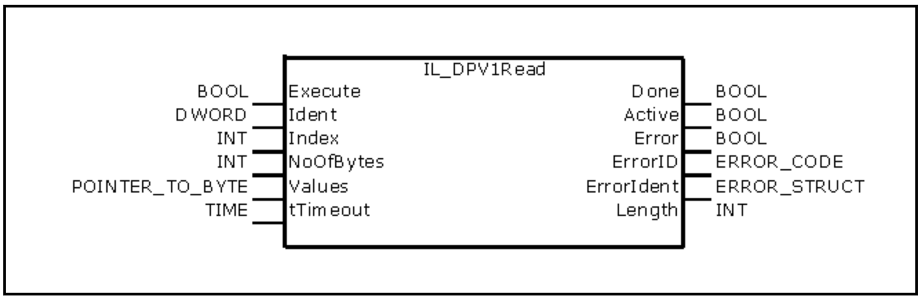


Fig. 11-51: FB IL_DPV1Read

	Name	Type	Description
VAR_INPUT	Execute	BOOL	Function activation; interruption of an activated function block is not possible.
	Ident	DWORD	Ident handle (see IL_DPIdent function)
	Index	INT	Process data index (field number)
	NoOfBytes	INT	Maximum length of the data to be read; number of bytes available on the "Values" pointer
	Values	POINTER TO BYTE	Pointer to the data buffer for the target data
	tTimeout	TIME	Timeout on waiting for data
VAR_OUTPUT	Done	BOOL	Done message (successful)
	Active	BOOL	Activity display
	Error	BOOL	Done message (unsuccessful)
	ErrorID	ERROR_CODE	See chapter "ErrorID / ERROR_CODE " on page 180
	ErrorIdent	ERROR_STRUCT	Error structure from PB_DP_TABLE ErrorTable
	Length	INT	Length of the data read in bytes

Fig. 11-52: IL_DPV1Read interface

Signal-Zeit-Diagramm

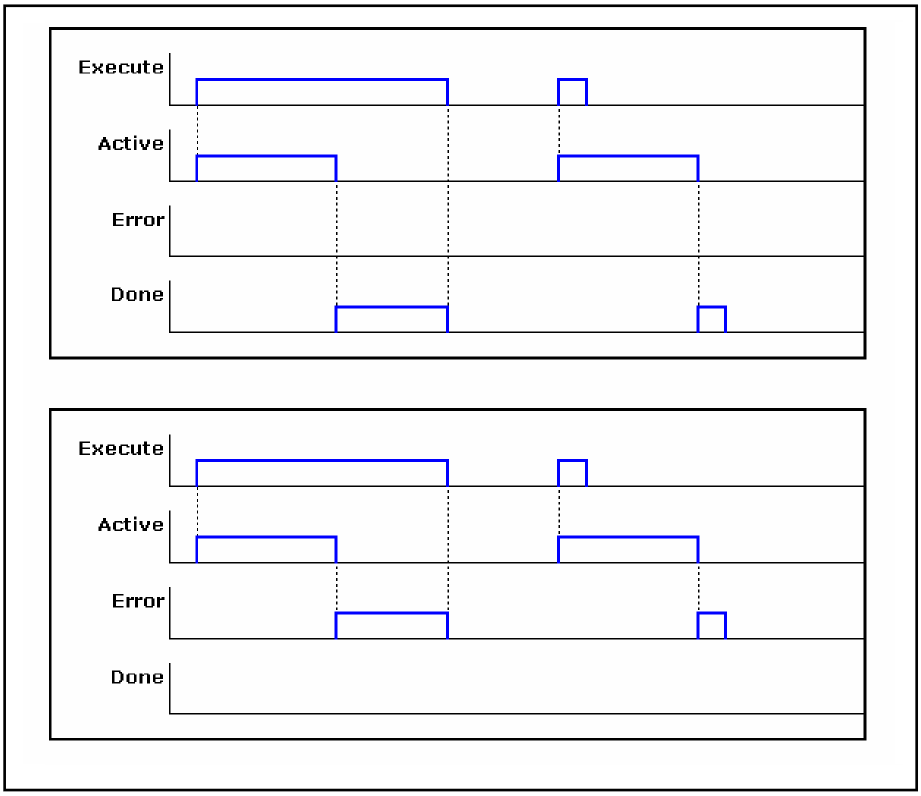


Fig. 11-53: IL_DPV1Read signal time diagram

Functional Description

The master (class 1) accesses a DP-V1 slave. It reads the data record of the slave. This data record is addressed through the slave address, the slot and the index. Addressing with slot and index and data interpretation are slave-specific and can be found in the documentation of the particular slave. The function execution time is dependent on the bus load and the set baud rate, among other factors.

This function is only available for slaves participating in the DP bus cycle.

11.3.7 IL_DPV1Write

Brief Description

The "IL_DPV1Write" function block is used for DPV1 write access. Data exchange on the Profibus DP is acyclic. A pointer (POINTER) must be addressed to deliver the process data to be written.

Library	Area
RIL_ProfibusDP_02	

Fig. 11-54: IL_DPV1Write library assignment

Interface Description

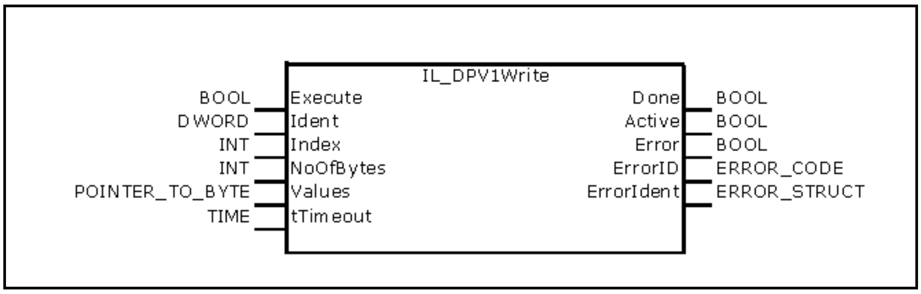


Fig. 11-55: FB IL_DPV1Write

	Name	Type	Description
VAR_INPUT	Execute	BOOL	Function activation; interruption of an activated function block is not possible.
	Ident	DWORD	Ident handle (see IL_DPIdent function)
	Index	INT	Process data index (field number)
	NoOfBytes	INT	Maximum length of the data to be written; number of bytes available on the "Values" pointer
	Values	POINTER TO BYTE	Pointer to the data buffer for the data
	tTimeout	TIME	Timeout on waiting for response
VAR_OUTPUT	Done	BOOL	Done message (successful)
	Active	BOOL	Activity display
	Error	BOOL	Done message (unsuccessful)
	ErrorID	ERROR_CODE	See chapter "ErrorID / ERROR_CODE " on page 180
	ErrorIdent	ERROR_STRUCT	Error structure from PB_DP_TABLE ErrorTable

Fig. 11-56: Interface signals: IL_DPV1Write

Signal Time Diagram

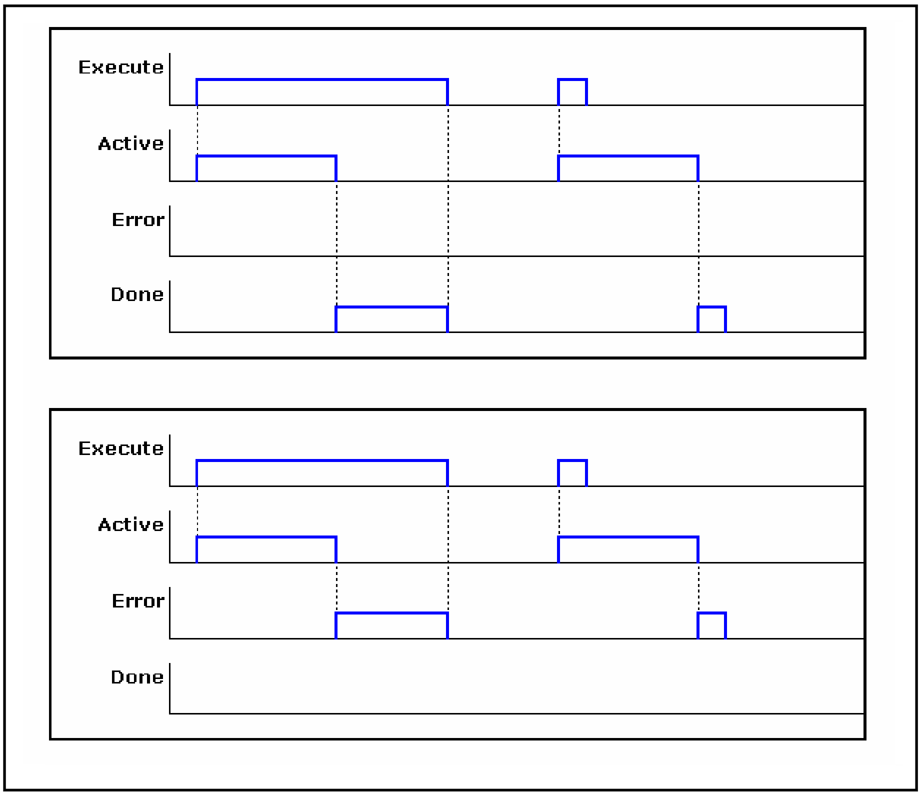


Fig. 11-57: IL_DPV1Write signal time diagram

Functional Description

The master (class 1) accesses a DP-V1 slave. It reads the data record of the slave. This data record is addressed through the slave address, the slot and the index. Addressing with slot and index and data interpretation are slave-specific and can be found in the documentation of the particular slave. The function execution time is dependent on the bus load and the set baud rate, among other factors.

This function is only available for slaves participating in the DP bus cycle.

11.3.8 IL_DPReadDiag

Brief Description

The "IL_DPReadDiag" function block is used by the DP master (DPM1) to read the diagnostic data of a slave. The data buffer of the diagnostic data must be provided to address it via a pointer (POINTER).

Library	Area
RIL_ProfibusDP_02	

Fig. 11-58: IL_DPReadDiag library assignment

Interface Description

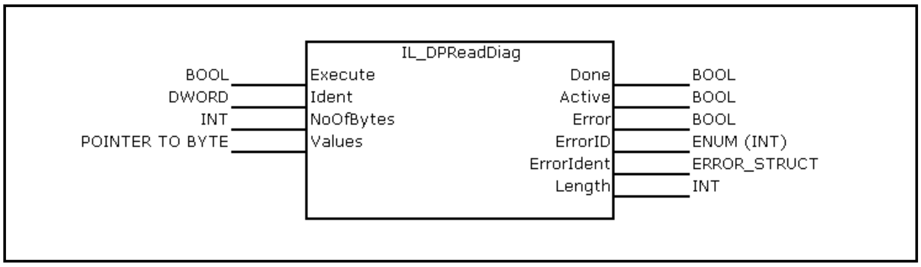


Fig. 11-59: IL_DPReadDiag structure

	Name	Type	Description
VAR_INPUT	Execute	BOOL	Function activation; interruption of an activated function block is not possible.
	Ident	DWORD	Ident handle (see IL_DPIdent function)
	NoOfBytes	INT	Maximum length of the data to be read; number of bytes available on the "Values" pointer
	Values	POINTER TO BYTE	Pointer to data buffers of slave diagnostics data according to Profibus DP standard
VAR_OUTPUT	Done	BOOL	Done message (successful)
	Active	BOOL	Activity display
	Error	BOOL	Done message (unsuccessful)
	ErrorID	ERROR_CODE	See chapter "ErrorID / ERROR_CODE " on page 180
	ErrorIdent	ERROR_STRUCT	Error structure from PB_DP_TABLE ErrorTable
	Length	INT	Length of the diagnostic data in bytes

Fig. 11-60: IL_DPReadDiag interface

Signal Time Diagram

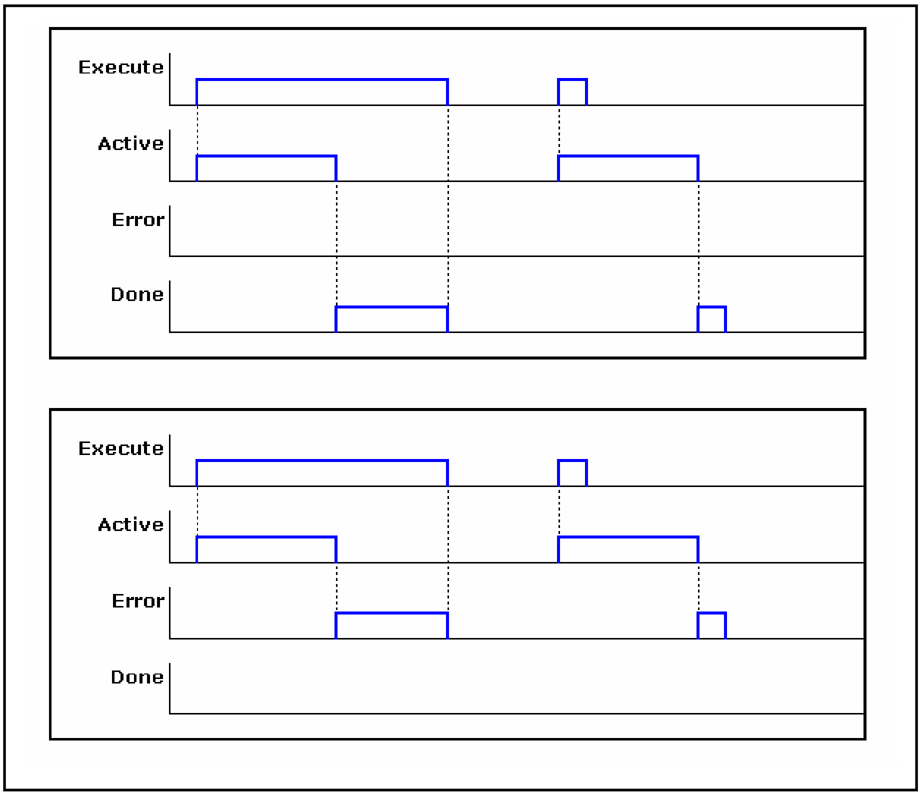


Fig. 11-61: IL_DPReadDiag signal time diagram

Functional Description

The master reads the diagnostic information from the slaves and saves it in relation to the particular slave. Using request bits, the slave triggers the diagnostic request of the master in cyclic telegram traffic. The slave is responsible for the content of the diagnostic data. The present function does not trigger any telegram traffic at the Profibus. It just accesses the diagnostic information provided by the master.

11.3.9 IL_DPSycFr

Brief Description

The "IL_DPSycFr" function block can be used to implement control commands for synchronization of inputs and outputs.

Profibus DP provides the possibility that a master sends what is called a "global control telegram" to a group of slaves. The global control telegram contains a control command.

- Using the **freeze** control command, all slaves of the addressed group are storing the current **input data** at the same time (synchronize inputs).
- Using the **sync** control command, all slaves of the addressed group are applying the current **output data** at the same time (synchronize outputs).

Library	Range
RIL_ProfibusDP_02	

Fig. 11-62: IL_DPSycF library assignment

Interface Description

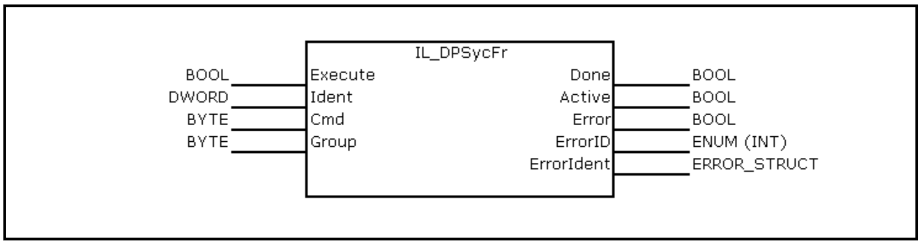


Fig. 11-63: DPSycFr structure

	Name	Type	Description
VAR_INPUT	Execute	BOOL	Function activation; interruption of an activated function block is not possible.
	Ident	DWORD	Ident handle; see IL_DPIdent (slot is irrelevant and should be 0)
	Cmd	BYTE	Control command (see above)
	Group	BYTE	Selects one or more groups to which the command refers. Each bit is assigned to a group.
VAR_OUTPUT	Done	BOOL	Done message (successful)
	Active	BOOL	Activity display
	Error	BOOL	Done message (unsuccessful)
	ErrorID	ERROR_CODE	See chapter "ErrorID / ERROR_CODE " on page 180
	ErrorIdent	ERROR_STRUCT	Error structure from PB_DP_TABLE ErrorTable

Fig. 11-64: IL_DPSycF interface

Signal Time Diagram

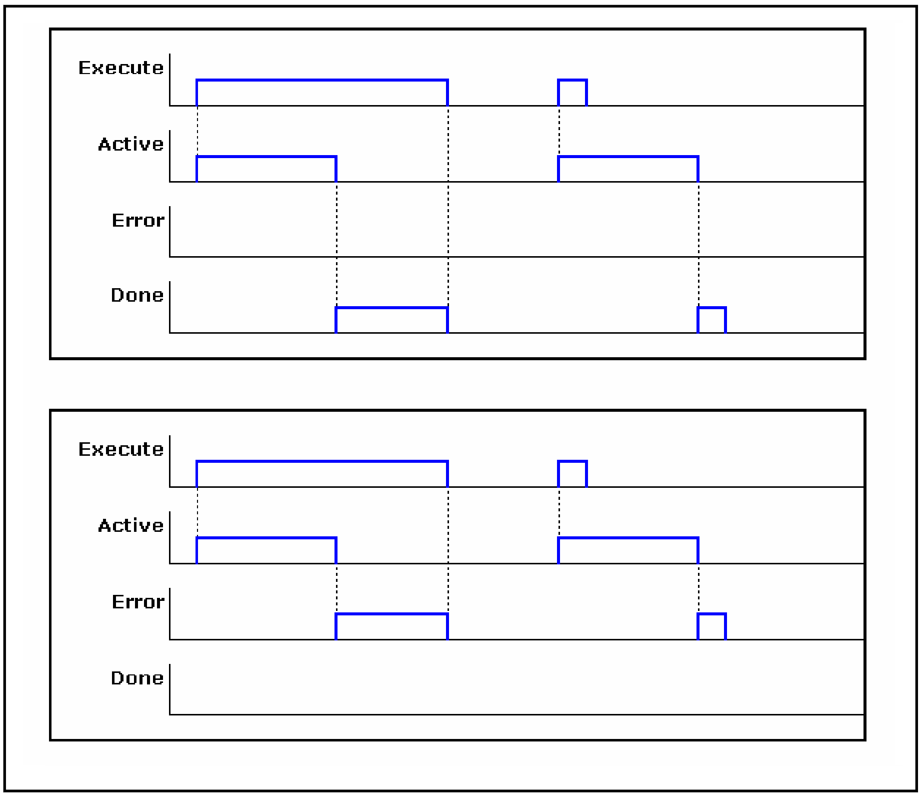


Fig. 11-65: IL_DPSycFr signal time diagram

Functional Description

Example: To address all slaves of a group, the value of the ID is 16#007f0000.

The ID is formed from the various components by means of the "IL_DPIdent" function.

☞ Outputs can only be synchronized if all slaves received the current output data before having received the sync command.

To achieve this, call the "IL_DPSycFr" function block from the same PLC task from which the output data of the slaves is written.

In a PLC task, the output data for the synchronized slaves is written first. Then, the SYNC command is started with "IL_DPSycF". As long as the Sync command is not completed yet (BUSY), the output data must not be modified.

If sync or freeze is used in the IndraWorks project explorer, an assignment of the groups at the master and the corresponding slaves must be set. For more information, please refer to the IndraWorks documentation or online help.

Possible control commands:

IL_DP_CMD_UNFREEZE	16#04	Freeze mode release
IL_DP_CMD_FREEZE	16#08	command Freeze
IL_DP_CMD_UNSYNC	16#10	Sync mode release
IL_DP_CMD_SYNC	16#20	command Sync

Fig. 11-66: Possible IL_DPSycF control commands

11.3.10 IL_DPDeviceListGet

Brief Description

The "IL_DPDeviceListGet" function block determines a list of the currently available DP masters.

Library	Area
RIL_ProfibusDP_02	

Fig. 11-67: IL_DPDeviceListGet library assignment

Interface Description

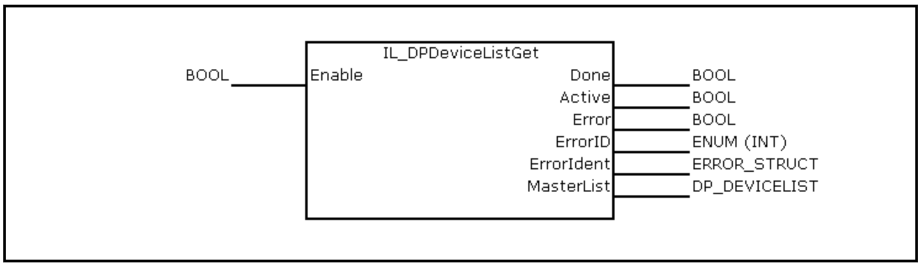


Fig. 11-68: IL_DPDeviceListGet structure

	Name	Type	Description
VAR_INPUT	Enable	BOOL	Function release
VAR_OUTPUT	Done	BOOL	Done message (successful)
	Active	BOOL	Activity display
	Error	BOOL	Done message (unsuccessful)

	Name	Type	Description
	ErrorID	ERROR_CODE	See chapter "ErrorID / ERROR_CODE " on page 180
	ErrorIdent	ERROR_STRUCT	Error structure from PB_DP_TABLE ErrorTable
	MasterList	DP_DEVICELIST	List of available DP masters

Fig. 11-69: IL_DPDeviceListGet interface

Signal Time Diagram

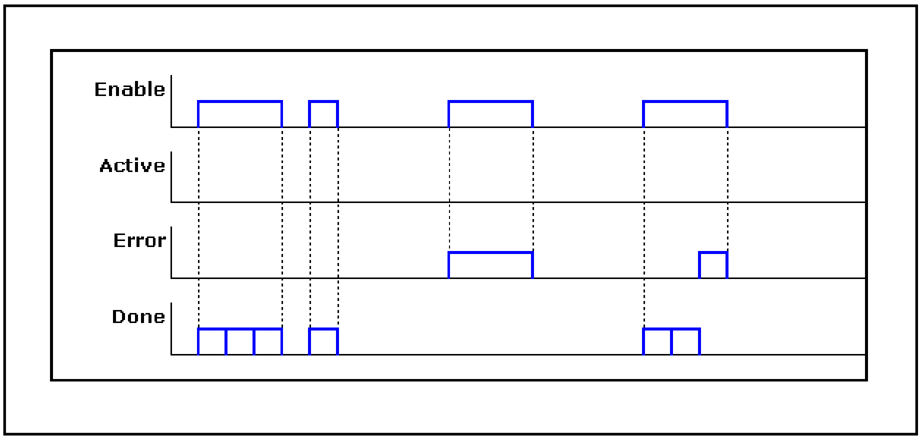


Fig. 11-70: IL_DPDeviceListGet signal time diagram

Functional Description

This function is used to display the master instances active and addressable at the PLC. This also allows the user to monitor whether the configuration loaded is appropriate and whether it has been detected correctly.

11.3.11 IL_DPBaudrateGet

Brief Description

The "IL_DPBaudrateGet" function block determines the baud rate of the connected field bus. The baud rate is specified in bits per second.

Library	Area
RIL_ProfibusDP_02	

Fig. 11-71: Library assignment

Interface Description

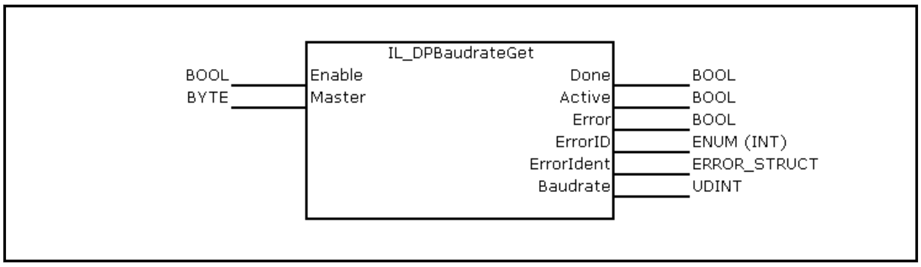


Fig. 11-72: IL_DPBaudrateGet structure

	Name	Type	Description
VAR_INPUT	Enable	BOOL	Function release
	Master	BYTE	MasterID (see above)
VAR_OUTPUT	Done	BOOL	Done message (successful)
	Active	BOOL	Activity display
	Error	BOOL	Done message (unsuccessful)
	ErrorID	ERROR_CODE	See chapter "ErrorID / ERROR_CODE " on page 180

	Name	Type	Description
	ErrorIdent	ERROR_STRUCT	Error structure from PB_DP_TABLE ErrorTable
	Baud rate	UDINT	Baud rate in 1/s

Fig. 11-73: IL_DPBaudrateGet interface

Signal Time Diagram

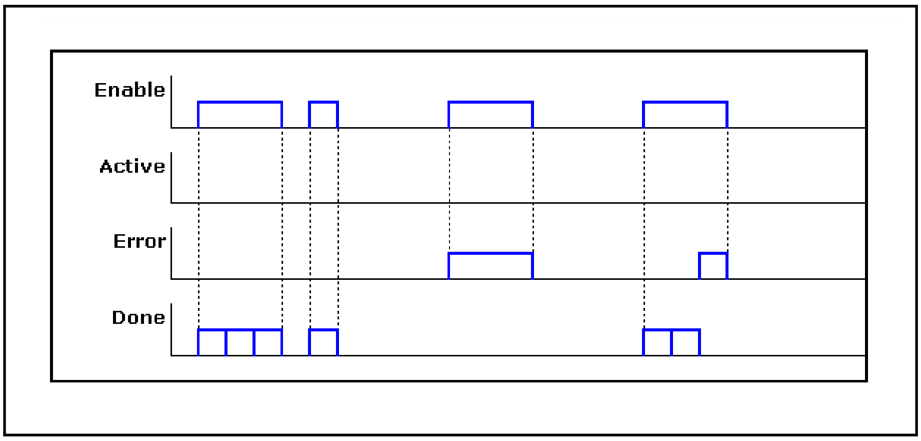


Fig. 11-74: IL_DPBaudrateGet signal time diagram

Functional Description

This function is used to determine the operating state of the PB master that has been addressed. "BmState" and "BmError" must be encoded with 1.3.2 and 1.3.3.

11.3.12 IL_DPDevInfoGet

Brief Description

The "IL_DPDevInfoGet" is used to obtain information about the operating state of the particular PB device.

Library	Range
RIL_ProfibusDP_02	

Fig. 11-75: IL_DPDevInfoGet library assignment

Interface Description

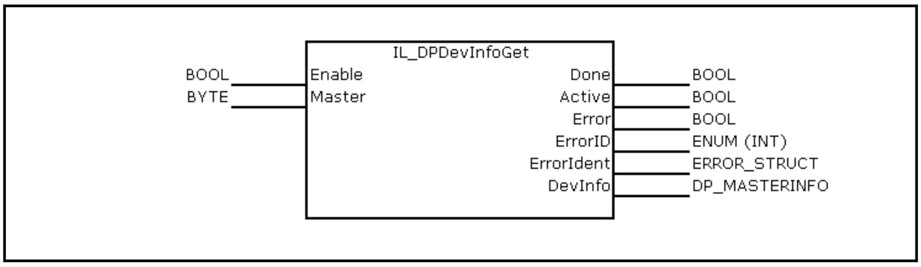


Fig. 11-76: IL_DPDevInfoGet structure

	Name	Type	Description
VAR_INPUT	Enable	BOOL	Function release
	Master	BYTE	MasterID (see above)
VAR_OUTPUT	Done	BOOL	Done message (successful)
	Active	BOOL	Activity display
	Error	BOOL	Done message (unsuccessful)
	ErrorID	ERROR_CODE	See chapter "ErrorID / ERROR_CODE " on page 180

	Name	Type	Description
	ErrorIdent	ERROR_STRUCT	Error structure from PB_DP_TABLE ErrorTable
	DevInfo	DP_MASTERINFO	DP_MASTERINFO

Fig.11-77: IL_DPDevInfoGet interface

Signal Time Diagram

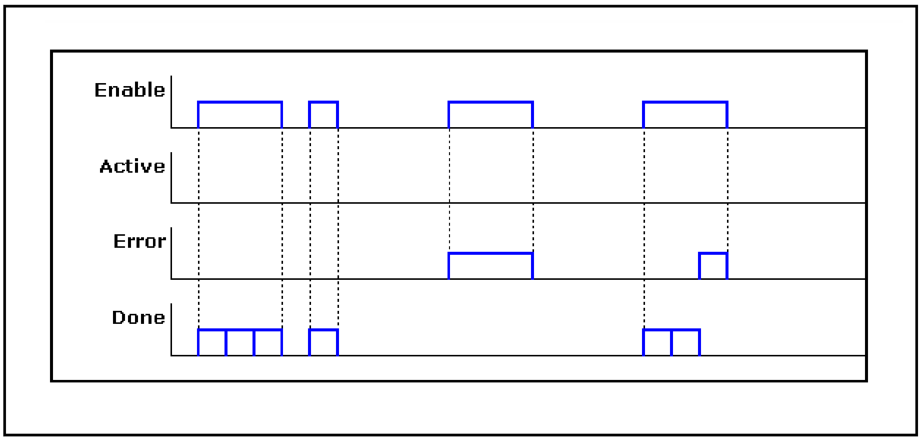


Fig.11-78: IL_DPDevInfoGet signal time diagram

11.3.13 IL_DPPrjSlaveListGet

Brief Description

The "IL_DPPrjSlaveListGet" function block supplies the list of projected slaves. The list contains all slaves available in the master configuration file.

Library	Area
RIL_ProfibusDP_02	

Fig.11-79: IL_DPDevInfoGet library assignment

Interface Description

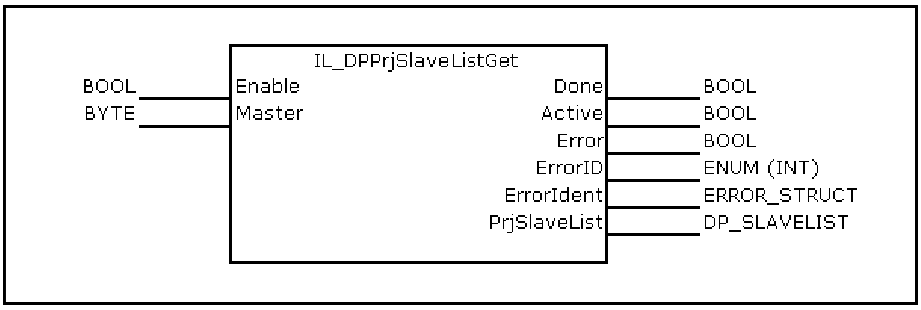


Fig.11-80: IL_DPPrjSlaveListGet structure

	Name	Type	Description
VAR_INPUT	Enable	BOOL	Function release
	Master	BYTE	MasterID (see above)

	Name	Type	Description
VAR_OUTPUT	Done	BOOL	Done message (successful)
	Active	BOOL	Activity display
	Error	BOOL	Done message (unsuccessful)
	ErrorID	ERROR_CODE	See chapter "ErrorID / ERROR_CODE " on page 180
	ErrorIdent	ERROR_STRUCT	Error structure from PB_DP_TABLE ErrorTable
	PrjSlaveList	DP_SLAVELIST	Bit list with set bit for every projected slave

Fig. 11-81: IL_DPPrjSlaveListGet interface

Signal Time Diagram

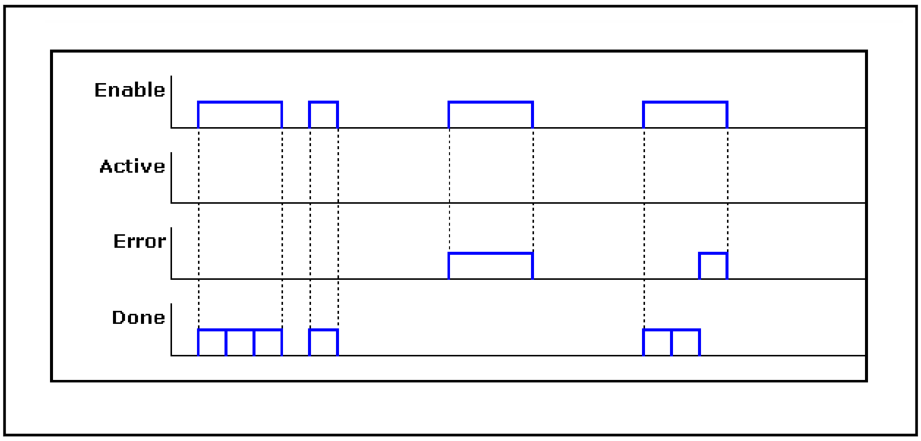


Fig. 11-82: IL_DPPrjSlaveListGet signal time diagram

Functional Description

The bit list is used to set a bit for each projected slave of a master. The function is based on the configuration data available.

11.3.14 IL_DPSlaveDiagListGet

Brief Description

The "IL_DPSlaveDiagListGet" supplies the list of slaves indicating diagnostics.

Library	Area
RIL_ProfibusDP_02	

Fig. 11-83: IL_DPSlaveDiagListGet library assignment

Interface Description

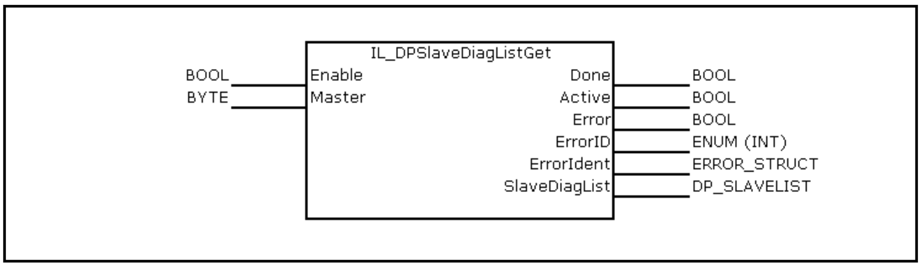


Fig. 11-84: IL_DPSlaveDiagListGet structure

	Name	Type	Description
VAR_INPUT	Enable	BOOL	Function release
	Master	BYTE	MasterID (see above)
VAR_OUTPUT	Done	BOOL	Done message (successful)

	Name	Type	Description
	Active	BOOL	Activity display
	Error	BOOL	Done message (unsuccessful)
	ErrorID	ERROR_CODE	See chapter "ErrorID / ERROR_CODE " on page 180
	ErrorIdent	ERROR_STRUCT	Error structure from PB_DP_TABLE ErrorTable
	SlaveDiagList	DP_SLAVELIST	Bit list with set bit for every slave indicating diagnostics

Fig. 11-85: IL_DPSlaveDiagListGet interface

Signal Time Diagram

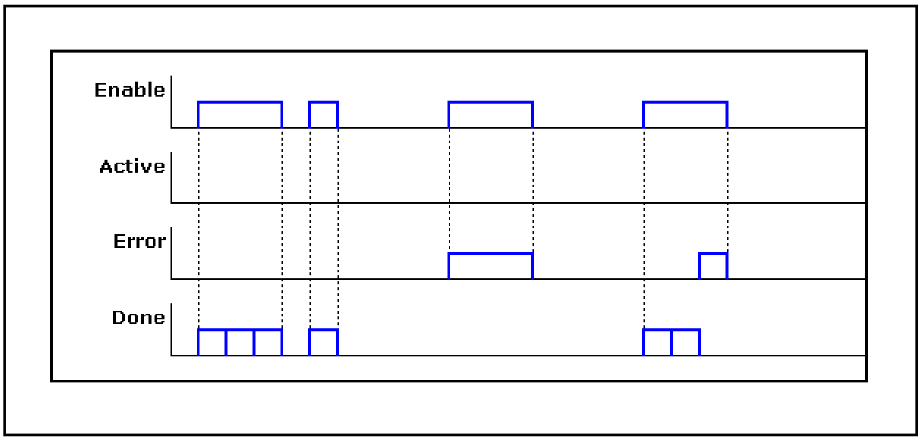


Fig. 11-86: IL_DPSlaveDiagListGet signal time diagram

Functional Description

The "SlaveDiagList" is used to set a bit for each slave indicating diagnostics. This allows reading selected diagnostics. This function block does not trigger any DP services. It is used to evaluate data provided in the master.

11.3.15 Error IDs

ErrorID / ERROR_CODE

Library	Range
RIL_CommonTypes.lib	Datatypes of POU diagnosis - includes

Fig. 11-87: ERROR_CODE, defined in RIL_CommomTypes

Enumerator	Code	Description
NONE_ERROR	16#0000	No error code available
INPUT_INVALID_ERROR	16#0001	Invlaid input assignment
COMMUNICATION_ERROR	16#0002	Communication error
RESOURCE_ERROR	16#0003	Source not available
ACCESS_ERROR	16#0004	Faulty or invalid access to data
STATE_MACHINE_ERROR	16#0005	Invalid state machine value
INPUT_RANGE_ERROR	16#0006	The value of one or more inputs is outside of the defined limits
CALCULATION_ERROR	16#0007	Calculation error
DEVICE_ERROR	16#0008	Drive error
OTHER_ERROR	16#7FFE	Undefined error (assignment to any of the other IDs not possible)
SYSTEM_ERROR	16#7FFF	System error

Fig.11-88: Possible ErrorIDs

ErrorIdent

Structure

ErrorIdent is a structure comprising three elements. Its default value is 0.

- Error_Table: PB_DP_TABLE (16#0130) Profibus DP error
- ErrorAdditional1: see below
- ErrorAdditional1: see below

ErrorAdditional1

ErrorAdditional1 is used for a superordinate distinction by error source. There are the following error sources:

16#0001: Onboard Profibus Device

16#0002: FunctionModule (FM) Profibus Device

16#0003: netXDevice

16#0100: Function not supported

16#0101: Device(Master) not found

16#0102: Slave not configured

16#0103: IO-driver not ready

16#0104: Timeout-Error

16#0107: Parameter-Error

ErrorAdditional2

ErrorAdditional2 also comprises 4 bytes for "Onboard Profibus Device".

The meaning of the bytes is as follows:

Byte no.	Meaning	Description
Byte 3	Error_Source	Distinctionn by error origin: 16#00 Profibus (slave) 16#10 Masterstack 16#20 Profibus FDL layer
Byte 2	Error_Code_DP Error_Code_OB Error_Code_FDL	For Error_Source = 16#00 For Error_Source = 16#10 For Error_Source = 16#20
Byte 1	AddInfo_1	Reserved
Byte 0	AddInfo_2	Reserved

Fig.11-89: Onboard Profibus coding

ErrorAdditional2 also comprises 4 bytes for "FunctionModule (FM) Profibus Device".

The meaning of the bytes is as follows:

Byte no.	Meaning	Description
Byte 3	FM_Error_Code	FM master error code
Byte 2	Error_Code_DP	See tables below
Byte 1	Error_Code_1	DP-user-specific
Byte 0	AddInfo_1	Reserved

Fig.11-90: Function module coding

FM_Error_Code

The following error description corresponds to the error response definitions of the Hilscher Profibus DP master.

FM_Error_Code	Error
16#02	The slave does not provide any memory or buffer for this service.
16#03	The slave does not support any DPV1 services.
16#09	The slave did not transmit any data.
16#11	The slave did not respond/is not applied to the bus.
16#12	The DP master is not applied to the ProfiBus (check cabling)
16#19	The slave does not comply with DPV1.
16#36	The slave rejected the access. Evaluate Error_Code_DP!
16#81	DPV1 is not configured on the master.
16#82	The slave did not respond with plausible parameters.
16#83	Another service already in progress; parallel services not allowed.
16#84	Data capacity exceeds configured size.
16#85	Wrong parameter in request.

16#9a	Unknown command
16#F0	Invalid state

Fig.11-91: ErrorCode function module

Error_Code_OB

Error_Code_OB	Error
16#11	Invalid order parameters
16#23	RequestList full
16#25	SemTake error
16#31	Unallowed call
16#32	Invalid call parameters
16#33	Invalid data length
16#34	Faulty call state
16#35	Slave not configured
16#36	Slave configured but not in cyclic mode
16#61	DPV1 request to non-DPV! slave
16#62	The slave does not respond within timeout.
16#63	DPV1 telegram format error
16#64	Order was withdrawn.
16#65	Pertinent RQB not found.
16#66	Invalid parameter
16#67	Unknown AMPRO2 opcode

Fig.11-92: Onboard ErrorCode

Error_Code_FDL

Error_Code_FDL	Error
16#61	FE: format error in a request APDU
16#62	NI: service not implemented
16#63	AD: access denied
16#64	EA: area to large (up/download)
16#65	LE: data block length too large (up/download)
16#66	RE: format error in a request APDU
16#67	IP: invalid parameter
16#68	SC: sequence conflict
16#69	SE: sequence error
16#6A	NE: area non-existent
16#60	No slave found

Error_Code_FDL	Error
16#6B	DI: data incomplete
16#6C	NC: master parameter set not compatible

Fig.11-93: Field bus data link layers (FDL) ErrorCode

Error_Code_DP

Aufbau

The meaning of "Error_Code_DP" corresponds to that of "Error_Code_1" described in the DPV1 standard. Bits 4..7 of the error byte constitute the "Error_Class", while Bits 0..3 constitute the "Error_Code".

<table>
<tr><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td><td>Meaning</td></tr>
<tr><td colspan="4"></td><td colspan="4">————————————</td><td>Error Code</td></tr>
<tr><td colspan="4">————————————</td><td colspan="4"></td><td>Error Class</td></tr>
</table>

Fig.11-94: Error Code DP

Error_Class	Meaning	Error_Code
0 to 9	reserved[2]	
10	Application	0 = read error 1 = write error 2 = module failure 3 to 7 = reserved[3] 8 = version conflict 9 = feature not supported 10 to 15 = user specific
11	Access	0 = invalid index 1 = write length error 2 = invalid slot 3 = type conflict 4 = invalid area 5 = state conflict 6 = access denied 7 = invalid range 8 = invalid parameter 9 = invalid type 10 to 15 = user specific

[2] *reserved Values are intended to be passed unchanged to the user.*

[3] *reserved Values are intended to be passed unchanged to the user.*

Error_Class	Meaning	Error_Code
12	resource	0 = read constrain conflict 1 = write constrain conflict 2 = resource busy 3 = resource unavailable 4 to 7 = reserved[4)] 8 to 15 = user specific
13 to 15	User specific	

Fig. 11-95: DP Error Code

Additional Info: User(Slave) specific

11.4 RIL_ProfibusDPSlave.library

11.4.1 General

The functionality describe in this chapter implements the acyclic READ and WRITE DPV1 services for Profibus DP slaves. This allows access to data objects at slave application level.

To implement the READ service, the function requires combined use of two function blocks (FBs).

FB IL_PBDPSlaveDPV1Polling is used to cyclically poll the activity of a DPV1 service request. If active, a READ request is checked by the user program for permissibility and answered by calling FB IL_PBDPSlaveDPV1Response.

The particular data object is accessed such that the parameters transferred to FB IL_PBDPSlaveDPV1Response are a pointer to the data object and the valid data length.

The data is copied by the FB.

A WRITE service requires combined use of three function blocks.

FB IL_PBDPSlaveDPV1Polling is used to cyclically poll the activity of a DPV1 service request. If active, a WRITE request is checked by the user program for permissibility. If necessary, the user data is copied to the target object by calling FB IL_PBDPSlaveGetWriteData. To achieve this, a pointer to the data object and the valid data length are transferred as parameters to FB IL_PBDPSlaveGetWriteData.

The data is copied by the FB.

The response is again given by calling FB IL_PBDPSlaveDPV1Response.

The maximum data length per access is 240 bytes (limited by the V1 channel of Profibus DP). Data volumes exceeding this maximum size can, for example, be achieved by fragmented into multiple fieldbus objects of 240 bytes each.

[4)] *reserved Values are intended to be passed unchanged to the user.*

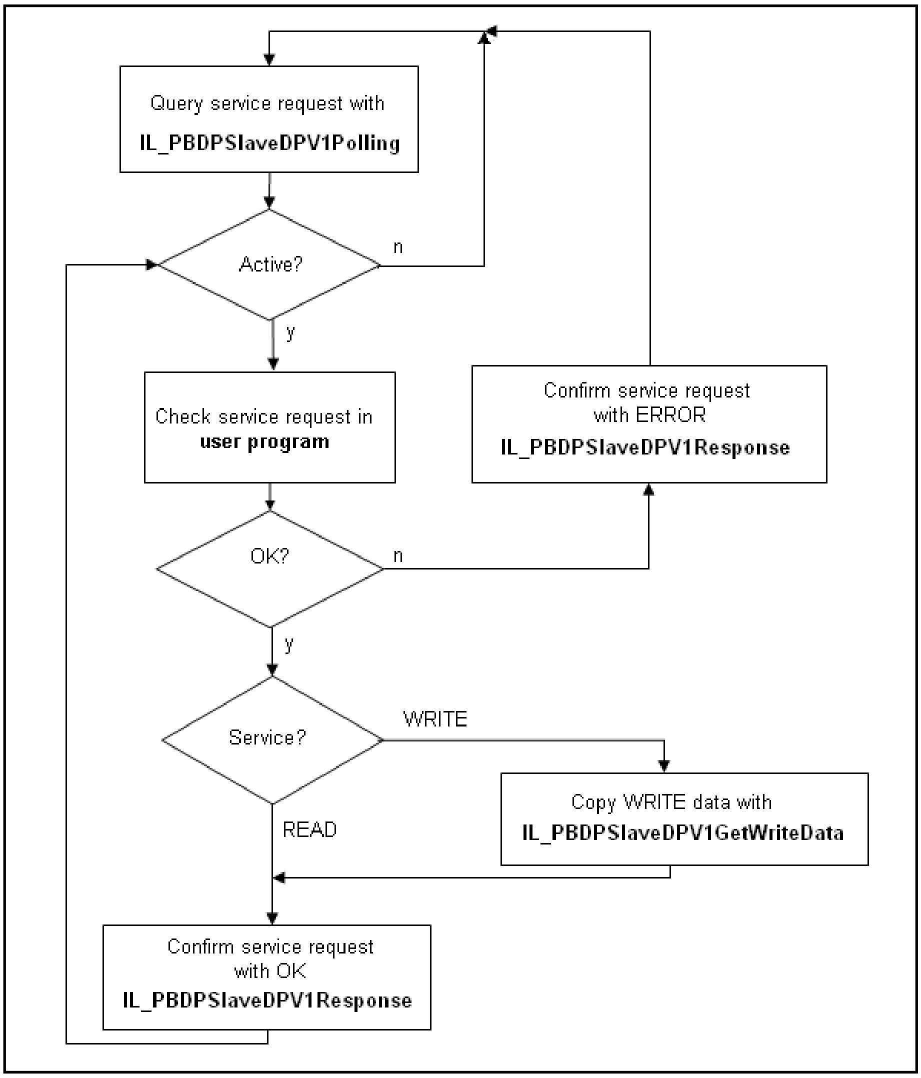


Fig.11-96: Cooperation of the function blocks of the RIL_ProfibusDPSlave library

Target systems

The library can be used with the following systems:

Target assembly	Remark
CML65	Onboard / function modules
CML45	Onboard / function modules
CML25	Onboard / function modules
VEP	Onboard
…	

Fig.11-97: Target systems

The library contains the following components:

- FB IL_PBDPSlaveDPV1Polling page 187, the polling FB is used to poll the activity of a DPV1 service request. The call is made cyclically.
- FB IL_PBDPSlaveDPV1Response page 189, the response FB is used to respond to an active DPV1 service request. The call is made in relation to the result of the polling FB.

- FB IL_PBDPSlaveDPV1GetWriteData page 192, FB GetWriteData is used to copy the data of a DPV1 WRITE service request to the application object.

- IL_FBUS_SLAVE_RESULT page 196, serves to encode an identifier of the result of the data transfer in the V1 channel.
- IL_FBUS_SLAVE_SERVICE page 195, serves to represent a service identifier of the DPV1 channel.

Selecting the slave

The IL_BUSSLAVE enumeration type of the RIL_FieldbusTypes.library serves to select the appropriate Profibus DP slave.

See also: IL_BUSSLAVE page 195.

11.4.2 IL_PBDPSlaveDPV1Polling

Brief description

The DPV1 polling FB is used to poll the activity of a DPV1 service request. The FB indicates that a service request is active by output parameters Service, Slot, Index, ServiceID, and DataLength.

Parameter Service defines the type of access (read or write) or is idle while no service request is active.

Parameters Slot and Index define the data object to be accessed.

ServiceID is an internal Ident parameter which must be transferred unchanged when FBs Response and GetWriteData are transferred.

DataLength specifies the access length, i.e., the number of data to be written or the maximum number of data to be read.

The user program is intended to check an active service request for permissibility and to respond to it by calling FB IL_PBDPSlaveDPV1Response (see below).

Assignment: target system / library

Target system	Library
XLC, MLC, MTX	RIL_ProfibusDPSlave.library
XLC, MLC, MTX	RIL_ProfibusDPSlave.compiled-library

Fig. 11-98: FB IL_PBDPSlaveDPV1Polling reference table

Interface description

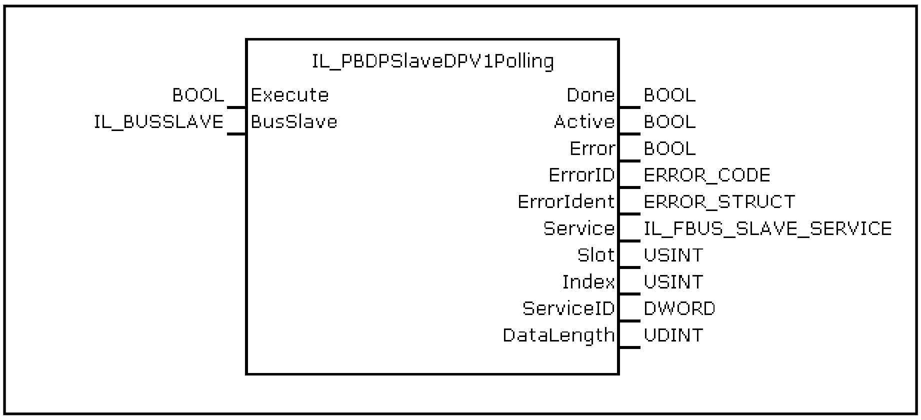


Fig. 11-99: FB IL_PBDPSlaveDPV1Polling interface

I/O type	Name	Data type	Description
VAR_INPUT	Enable	BOOL	Processing of FB enabled
	BusSlave	IL_BUSSLAVE	Instance of the slave in the order of the configuration tree. If only one Profibus DP slave is available, the value of this slave is always IL_BUSSLAVE_0.
VAR_OUTPUT	Done	BOOL	Processing completed without error, output data valid
	Active	BOOL	Processing not completed yet, output data is invalid
	Error	BOOL	Processing completed with error
	ErrorID	ERROR_CODE	Description of the diagnostics in case of error
	ErrorIdent	ERROR_STRUCT	Detailed diagnostics
	Service	IL_FBUS_SLAVE_SERVICE	Service identifier (read, write or idle)
	Slot	USINT	Address parameter Slot
	Index	USINT	Address parameter Index
	ServiceID	DWORD	Ident parameter ServiceID
	DataLength	UDINT	Data length (possible data lengths are ≤ 240 bytes)

Fig.11-100: FB IL_PBDPSlaveDPV1Polling I/O interface

Minimum / maximum values and default values of the inputs

Name	Type	Min. value	Max. value	Default value	Effective
Enable	BOOL			FALSE	Continuously
BusSlave	IL_BUSSLAVE	IL_BUSSLAVE_0	IL_BUSSLAVE_4	IL_BUSSLAVE_0	TRUE state of enable

Fig.11-101: Min/max values and default values of the IL_PBDPSlaveDPV1Polling inputs

Functional description

- When there is a rising edge at "Execute", IL_PBDPSlaveDPV1Polling is called with the call being effective and parameter BusSlave is applied.
- If Execute = FALSE, the call does not have any effect. In this case, all output parameters are set to '0'.
- Initially, the result of the call is shown in output parameters Done, Active and Error. If Active = FALSE, the call is completed and the other parameters must be evaluated.
- If Done = TRUE and Error = FALSE, no error occurred during the call.
- If Done = FALSE and Error = TRUE, errors occurred during the call. In this case, parameters ErrorID and ErrorIdent describe the error cause.
- If no error occurred during the call, the effective output parameters Service, Slot, Index, ServiceID and DataLength are valid and can be evaluated by the calling instance.
- If errors occurred during the call, the effective output parameters are invalid.
- If output parameters are valid, Service shows the request type (READ, WRITE oder IDLE). If the type is READ or WRTE, Slot and Index define the object to be accessed and DataLength defines the access length.

- If Service = IDLE, there is no active service request. In this case, a treatment by the user program is not required.

Error handling

If Error = TRUE, errors occurred during the call.

The errors of the function block can be found in error table **PB_DP_TABLE** (ERROR_TABLE = 16#0130).

In this case, ErrorID and ErrorIdent (Additional1 and Additional2) define the error cause.

ErrorID	Additional1	Additional2	Description
INPUT_RANGE_ERROR	16#1001	BusSlave instance	Invalid BusSlave instance
STATE_MACHINE_ERROR	16#1011	0	Repeated polling of an active service
SYSTEM_ERROR	16#1021	0	Slave interface not available
SYSTEM_ERROR	16#1022	Interface error	Slave interface error

Fig. 11-102: FB IL_PBDPSlaveDPV1Polling error codes

11.4.3 IL_PBDPSlaveDPV1Response

Brief description

The DPV1 Response FB is used to respond to an active DPV1 service request that was detected with **IL_PBDPSlaveDPV1Polling** beforehand (see above). Depending on whether or not the request is permissible, the FB is called with different parameters ValueAdr, SizeOfValue and Result.

If the **READ** service request is permissible, a pointer to the data object is transferred to the FB in parameter ValueAdr and the corresponding data length in parameter SizeOfValue. The data is copied by the Response FB.

When the **WRITE** service is active, the data is copied by FB **IL_PBDPSlaveGetWriteData** (see below) before the Response FB is called. In this case, parameter ValueAdr is of no relevance.

The response to an impermissible request is given with ValueAdr = 0, SizeOfValue = 0 and a corresponding error identifier in Result.

Assignment: target system / library

Target system	Library
XLC, MLC, MTX	RIL_ProfibusDPSlave.library
XLC, MLC, MTX	RIL_ProfibusDPSlave.compiled-library

Fig. 11-103: FB IL_PBDPSlaveDPV1Response reference table

Interface description

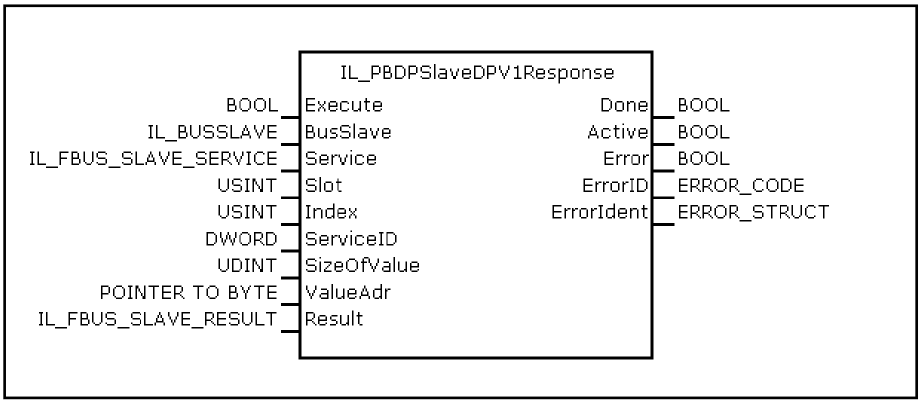


Fig.11-104: IFB L_PBDPSlaveDPV1Response interface

I/O type	Name	Data type	Description
VAR_INPUT	Execute	BOOL	Processing of FB enabled
	BusSlave	IL_BUSSLAVE	Instance of the slave in the order of the configuration tree. If only one Profibus DP slave is available, the value of this slave is always IL_BUSSLAVE_0.
	Service	IL_FBUS_SLAVE_SERVICE	Identifier of the service to be completed. This value is compared with the active service.
	Slot	USINT	Address parameter Slot of the service to be completed. This value is compared with the active Slot.
	Index	USINT	Address parameter Index of the service to be completed. This value is compared with the active Index.
	ServiceID	DWORD	Ident parameter ServiceID of the service for which the data is to be copied (parameter of IL_PBDPSlaveDPV1Polling). This value is compared with the active ServiceID.
	SizeOfValue	UINT	Maximum size of the data range of ValueAdr
	ValueAdr	POINTER TO BYTE	Pointer to data
	Result	IL_FBUS_SLAVE_RESULT	Result of the access (is returned to the requesting Profibus master via the V1 response)
VAR_OUTPUT	Done	BOOL	Processing completed without error, output data valid
	Active	BOOL	Processing not completed yet, output data is invalid
	Error	BOOL	Processing completed with error
	ErrorID	ERROR_CODE	Description of the diagnostics in case of error
	ErrorIdent	ERROR_STRUCT	Detailed diagnostics

Fig.11-105: FB IL_PBDPSlaveDPV1Response I/O interface

Minimum / maximum values and default values of the inputs

Name	Type	Min. value	Max. value	Default value	Effective
Execute	BOOL			FALSE	Continuously
BusSlave	IL_BUSSLAVE	IL_BUSSLAVE_0	IL_BUSSLAVE_4	IL_BUSSLAVE_0	Rising edge at Execute

Name	Type	Min. value	Max. value	Default value	Effective
Service	IL_FBUS_SLAVE_SERVICE	IL_SLAVE_SERVICE_IDLE	IL_SLAVE_SERVICE_WRITE	IL_SLAVE_SERVICE_IDLE	Rising edge at Execute
Slot	USINT	0	255	0	Rising edge at Execute
Index	USINT	0	255	0	Rising edge at Execute
ServiceID	DWORD	0	16#FFFFFFFF	0	Rising edge at Execute
SizeOfValue	UDINT	0 (1)	240	0	Rising edge at Execute
ValueAdr	POINTER TO BYTE		n.def.	0	Rising edge at Execute
Result	IL_FBUS_SLAVE_RESULT	IL_SLAVE_RESULT_OK	n.def.	IL_SLAVE_RESULT_OK	Rising edge at Execute

Fig.11-106: Min/max values and default values of the IL_PBDPSlaveDPV1Response inputs

Functional description

- When there is a rising edge at "Execute", IL_PBDPSlaveDPV1Response is called with the call being effective and the input parameters are applied.
- If Execute = FALSE, the call does not have any effect. In this case, all output parameters are set to '0'.
- Input parameters Service, Slot and Index define the data object having been accessed. It is imperative that these input parameters are identical with the corresponding output parameters of **IL_PBDPSlaveDPV1Polling**.
- ServiceID is the ident parameter of the active service. It is imperative that this parameter is identical with the corresponding output parameter of **IL_PBDPSlaveDPV1Polling**.
- In case of a WRITE service, SizeOfValue is identical with DataLength of **IL_PBDPSlaveDPV1Polling**, but may also be smaller in case of a READ service.
- ValueAdr is a pointer to the data object addressed and is '0' in case of an invalid service request.
- Result defines an error event during access to the requested data object.
- Initially, the result of the call is shown in output parameters Done, Active and Error.

  If Active = FALSE, the call is completed and the other parameters must be evaluated.
- If Done = TRUE and Error = FALSE, no error occurred during the call.
- If Done = FALSE and Error = TRUE, errors occurred during the call.

  In this case, parameters ErrorID and ErrorIdent describe the error cause.

Error handling

If Error = TRUE, errors occurred during the call.

The errors of the function block can be found in error table **PB_DP_TABLE** (ERROR_TABLE = 16#0130).

In this case, ErrorID and ErrorIdent (Additional1 and Additional2) define the error cause.

ErrorID	Additional1	Additional2	Description
INPUT_RANGE_ERROR	16#2001	BusSlave instance	Invalid BusSlave instance
INPUT_RANGE_ERROR	16#2002	Service	Invalid service
INPUT_RANGE_ERROR	16#2003	Slot	Invalid Slot
INPUT_RANGE_ERROR	16#2004	Index	Invalid Index
INPUT_RANGE_ERROR	16#2005	ServiceID	Invalid ServiceID
INPUT_RANGE_ERROR	16#2006	SizeOfValue	Invalid SizeOfValue
INPUT_RANGE_ERROR	16#2007	ValueAdr	Invalid ValueAdr
INPUT_RANGE_ERROR	16#2008	Result	Invalid Result
STATE_MACHINE_ERROR	16#2011	0	No service active
COMMUNICATION_ERROR	16#2012	Timeout [ms]	Service not processed in time
SYSTEM_ERROR	16#2021	0	Slave interface not available
SYSTEM_ERROR	16#2022	Interface error	Slave interface error

Fig. 11-107: FB IL_PBDPSlaveDPV1Response error codes

11.4.4 IL_PBDPSlaveDPV1GetWriteData

Brief description

The DPV1 GetWriteData FB is used to copy the user data to the appropriate target object in case of an active and permissible DPV1 WRITE service request that was detected with **IL_PBDPSlaveDPV1Polling** beforehand (see above).

To achieve this, a pointer to the data object is transferred to the FB in parameter ValueAdr and the corresponding data length in parameter SizeOfValue.

Assignment: target system / library

Target system	Library
XLC, MLC, MTX	RIL_ProfibusDPSlave.library
XLC, MLC, MTX	RIL_ProfibusDPSlave.compiled-library

Fig. 11-108: FB IL_PBDPSlaveDPV1GetWriteData reference table

Interface description

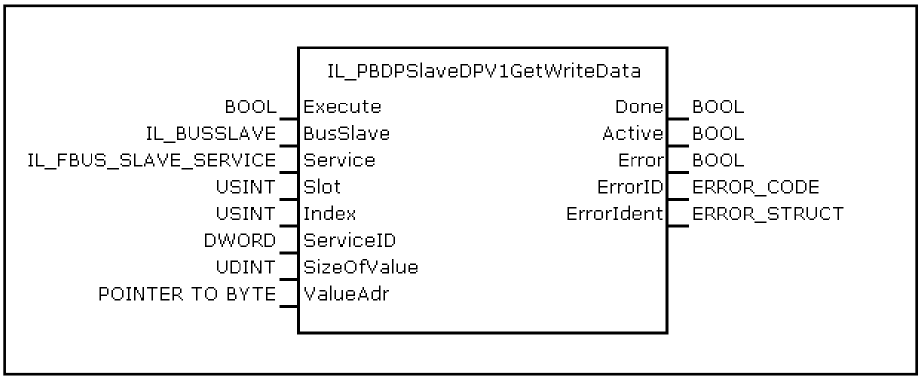


Fig. 11-109: FB IL_PBDPSlaveDPV1GetWriteData interface

I/O type	Name	Data type	Description
VAR_INPUT	Execute	BOOL	Enables processing of the function block (once, edge-triggered)
	BusSlave	IL_BUSSLAVE	Instance of the slave in the order of the configuration tree. If only one Profibus DP slave is available, the value of this slave is always IL_BUSSLAVE_0.
	Service	IL_FBUS_SLAVE_SERVICE	Identifier of the service for which the data is to be copied. This value is compared with the active service.
	Slot	USINT	Address parameter Slot of the service for which the data is to be copied. This value is compared with the active Slot.
	Index	USINT	Address parameter Index of the service for which the data is to be copied. This value is compared with the active Index.
	ServiceID	DWORD	Ident parameter ServiceID of the service for which the data is to be copied (parameter of IL_PBDPSlaveDPV1Polling). This value is compared with the active ServiceID.
	SizeOfValue	UINT	Maximum size of the data range of ValueAdr
	ValueAdr	POINTER TO BYTE	Pointer to data
VAR_OUTPUT	Done	BOOL	Processing completed without error, output data valid
	Active	BOOL	Processing not completed yet, output data is invalid
	Error	BOOL	Processing completed with error
	ErrorID	ERROR_CODE	Description of the diagnostics in case of error
	ErrorIdent	ERROR_STRUCT	Detailed diagnostics

Fig. 11-110: FB IL_PBDPSlaveDPV1GetWriteData I/O interface

Minimum / maximum values and default values of the inputs

Name	Type	Min. value	Max. value	Default value	Effective
Execute	BOOL			FALSE	Continuously
BusSlave	IL_BUSSLAVE	IL_BUSSLAVE_0	IL_BUSSLAVE_4	IL_BUSSLAVE_0	Rising edge at Execute
Service	IL_FBUS_SLAVE_SERVICE	IL_SLAVE_SERVICE_IDLE	IL_SLAVE_SERVICE_WRITE	IL_SLAVE_SERVICE_IDLE	Rising edge at Execute
Slot	USINT	0	255	0	Rising edge at Execute
Index	USINT	0	255	0	Rising edge at Execute
ServiceID	DWORD	0	16#FFFFFFFF		Rising edge at Execute
SizeOfValue	UDINT	0 (1)	240	0	Rising edge at Execute
ValueAdr	POINTER TO BYTE	n.def.	n.def.	0	Rising edge at Execute

Fig. 11-111: Min/max values and default values of the IL_PBDPSlaveDPV1GetWriteData inputs

Functional description

- When there is a rising edge at "Execute", IL_PBDPSlaveDPV1GetWriteData is called with the call being effective and the input parameters are applied.

- If Execute = FALSE, the call does not have any effect. In this case, all output parameters are set to '0'.
- If Execute = TRUE (constantly), the output parameters remain as they are.
- Input parameters Service, Slot and Index define the data object having been accessed. It is imperative that these input parameters are identical with the corresponding output parameters of **IL_PBDPSlaveDPV1Polling**.
- ServiceID is the ident parameter of the active service. It is imperative that this parameter is identical with the corresponding output parameter of **IL_PBDPSlaveDPV1Polling**.
- In case of a WRITE service, SizeOfValue is identical with DataLength of **IL_PBDPSlaveDPV1Polling**, but may also be smaller in case of a READ service.
- ValueAdr is a pointer to the target object.
- Initially, the result of the call is shown in output parameters Done, Active and Error. If Active = FALSE, the call is completed and the other parameters must be evaluated.
- If Done = TRUE and Error = FALSE, no error occurred during the call.
- If Done = FALSE and Error = TRUE, errors occurred during the call. In this case, parameters ErrorID and ErrorIdent describe the error cause.

Error handling If Error = TRUE, errors occurred during the call.

The errors of the function block can be found in error table **PB_DP_TABLE** (ERROR_TABLE = 16#0130).

In this case, ErrorID and ErrorIdent (Additional1 and Additional2) define the error cause.

ErrorID	Additional1	Additional2	Description
INPUT_RANGE_ERROR	16#3001	BusSlave instance	Invalid BusSlave instance
INPUT_RANGE_ERROR	16#3002	Service	Invalid service
INPUT_RANGE_ERROR	16#3003	Slot	Invalid Slot
INPUT_RANGE_ERROR	16#3004	Index	Invalid Index
INPUT_RANGE_ERROR	16#3005	ServiceID	Invalid ServiceID
INPUT_RANGE_ERROR	16#3006	SizeOfValue	Invalid SizeOfValue
INPUT_RANGE_ERROR	16#3007	ValueAdr	Invalid ValueAdr
STATE_MACHINE_ERROR	16#3011	0	No service active
COMMUNICATION_ERROR	16#3012	Timeout [ms]	Service not processed in time
SYSTEM_ERROR	16#3021	0	Slave interface not available
SYSTEM_ERROR	16#3022	Interface error	Slave interface error

Fig.11-112: FB IL_PBDPSlaveDPV1GetWriteData error codes

11.4.5 IL_BUSSLAVE

Brief description

The IL_BUSSLAVE enumeration type serves to select the particular Profibus DP slave. In addition to the onboard Profibus interface, other slaves can also be available in an MLC system via function modules.

The Profibus DP slaves are distinguished on the basis of their ascending order in the configuration. Indexes are assigned to the devices in the following order:

- Onboard slave
- Function module 1
- Function module 2
- Function module 3
- Function module 4

If, for example, the onboard slave is not a Profibus slave while a Profibus slave is configured on the FM1, IL_BUSSLAVE_0 is the value assigned to the slave on the FM1.

Element	Value	Description
IL_BUSSLAVE_0	0	First device
IL_BUSSLAVE_1	1	Second device
IL_BUSSLAVE_2	2	Third device
IL_BUSSLAVE_3	3	Fourth device
IL_BUSSLAVE_4	4	Fifth device

Fig. 11-113: Elements of the IL_BUSSLAVE enumeration type

Examples of addressing the various instances of the Profibus slaves on a control:

Configured:	Addressed with:
Onboard: Profibus DP Slave	IL_BUSSLAVE_0
FM1: Profibus DP Master	IL_BUSMASTER_0
FM2: Profibus DP Slave	IL_BUSSLAVE_1
FM3: Profibus DP Slave	IL_BUSSLAVE_2

Fig. 11-114: Example 1 of IL_BUSSLAVE enumeration type

11.4.6 IL_FBUS_SLAVE_SERVICE

Brief description

The IL_FBUS_SLAVE_SERVICE enumeration type serves to represent a service identifier of the DPV1 channel. This indicates the request state currently existing in the V1 channel.

The bits are contained in the "RIL_FIELDBUS_TYPES" library in the form of defines.

Element	Value	Description
IL_SLAVE_SERVICE_IDLE	0	No active request
IL_SLAVE_SERVICE_READ	1	A read request is present
IL_SLAVE_SERVICE_WRITE	2	A write request is present

Fig. 11-115: Elements of data type IL_FBUS_SLAVE_SERVICE

11.4.7 IL_FBUS_SLAVE_RESULT

Brief description

The IL_FBUS_SLAVE_RESULT enumeration type serves to encode an identifier of the result of the data transfer in the V1 channel. This allows the user to specify whether data transfer was faultless and which error code is returned to the requesting Profibus master via the V1 channel.

The bits are contained in the "RIL_FIELDBUS_TYPES" library in the form of defines.

Element	Value	Description
IL_SLAVE_RESULT_OK	16#00	No error during access
IL_SLAVE_RESULT_INVALID_SLOT	16#B2	Faulty Slot
IL_SLAVE_RESULT_INVALID_INDEX	16#B0	Faulty Index
IL_SLAVE_RESULT_ACCESS_DENIED	16#B6	Access not allowed
IL_SLAVE_RESULT_READ_ONLY	16#BA	Write access not allowed
IL_SLAVE_RESULT_READ_ERROR	16#A0	Error during reading (with respect to the PLC application)
IL_SLAVE_RESULT_WRITE_ERROR	16#A1	Error during writing (with respect to the PLC application)
IL_SLAVE_RESULT_WRITE_LENGTH_ERROR	16#B1	Write access with excessive data length (the number of data to be written exceeds the number allowed for the data object)
IL_SLAVE_RESULT_READ_LENGTH_ERROR	16#BB	Read access with insufficient data length (the number of data to be read is less than the number required for the data object)
IL_SLAVE_RESULT_RESOURCE_BUSY	16#C2	Data access not possible at present
IL_SLAVE_RESULT_USER_ERROR_0	16#F0	Meaning is user-defined
IL_SLAVE_RESULT_USER_ERROR_1	16#F1	Meaning is user-defined
IL_SLAVE_RESULT_USER_ERROR_2	16#F2	Meaning is user-defined
IL_SLAVE_RESULT_USER_ERROR_3	16#F3	Meaning is user-defined
IL_SLAVE_RESULT_USER_ERROR_4	16#F4	Meaning is user-defined
IL_SLAVE_RESULT_USER_ERROR_5	16#F5	Meaning is user-defined
IL_SLAVE_RESULT_USER_ERROR_6	16#F6	Meaning is user-defined
IL_SLAVE_RESULT_USER_ERROR_7	16#F7	Meaning is user-defined
IL_SLAVE_RESULT_USER_ERROR_8	16#F8	Meaning is user-defined
IL_SLAVE_RESULT_USER_ERROR_9	16#F9	Meaning is user-defined
IL_SLAVE_RESULT_USER_ERROR_10	16#FA	Meaning is user-defined
IL_SLAVE_RESULT_USER_ERROR_11	16#FB	Meaning is user-defined
IL_SLAVE_RESULT_USER_ERROR_12	16#FC	Meaning is user-defined
IL_SLAVE_RESULT_USER_ERROR_13	16#FD	Meaning is user-defined
IL_SLAVE_RESULT_USER_ERROR_14	16#FE	Meaning is user-defined
IL_SLAVE_RESULT_USER_ERROR_15	16#FF	Meaning is user-defined

Fig.11-116: Elements of data type IL_FBUS_SLAVE_RESULT

11.5 RIL_ProfinetIO.library

11.5.1 General

This library describes the IEC program options allowing access to the netX-PROFINET I/O controller.

Implementation of the library is based on RIL_UTILITIES.

The IL_EnableDoneBase and IL_ExecuteDoneBase methods are used and extended or overwritten.

Target systems

The library can be used with the following systems:

Target assembly	Remark
CML65	Onboard / 4 CFL01_1_TP
CML45	Onboard / 4 CFL01_1_TP
CML25	Onboard / 4 CFL01_1_TP

Fig.11-117: Target systems

The library contains the following components:

Function blocks for diagnostic purposes

- IL_PNIOControllerState, page 198, for determining the basic state
- IL_PNIOControllerStateDetails, page 199, for determining the state of the Profinet stack

Function blocks for remote diagnostic purposes

- IL_PNIORemoteDeviceState, page 201, for determining the basic state of any device desired
- IL_PNIORemoteDeviceStateDetails, page 202, for determining the basic state of any device desired

Function blocks for determining lists

- IL_PNIOGetConfigDeviceNameList, page 206, for determining the list of configured devices
- IL_PNIOGetDiagDeviceNameList, page 207, for determining the list of devices which send diagnostic messages

Function blocks for implementing the following acyclic services:

- IL_PNIOWriteRecord, page 211, acyclic writing
- IL_PNIOReadRecord, page 209, acyclic reading

Selecting the controller

Addressing: The PROFINET I/O controllers are distinguished on the basis of their ascending order in the configuration.

Counting starts with the 0th controller, i.e., if there is only one configured controller, its controller instance is always 0.

See also: IL_BUSMASTER, page 214.

Addressing the device

Device name: PROFINET uses the "IdentStr" device name to address the devices.

The device name must be entered in the configuration. Formally, this name is a DNS name.

That means that the character set comprises lower case letters and '-'. ('-' neither at the beginning nor at the end), cf. RFC 1034 (DNS).

11.5.2 IL_PNIOControllerState

Brief Description

This function block returns a basic diagnosis of the PROFINET IO controller. If a diagnostic message is present, additional FBs can be used to read details on the diagnostic messages (e.g., IL_PNIOControllerStateDetails, page 199, and IL_PNIORemoteDeviceState, page 201,).

Target system	Library
MLC 10VRS and higher	RIL_ProfinetIO.compiled-library
MTX 10VRS and higher	RIL_ProfinetIO.compiled-library
IndraLogic L65 10VRS and above	RIL_ProfinetIO.compiled-library

Fig. 11-118: FB IL_PNIOControllerState reference table

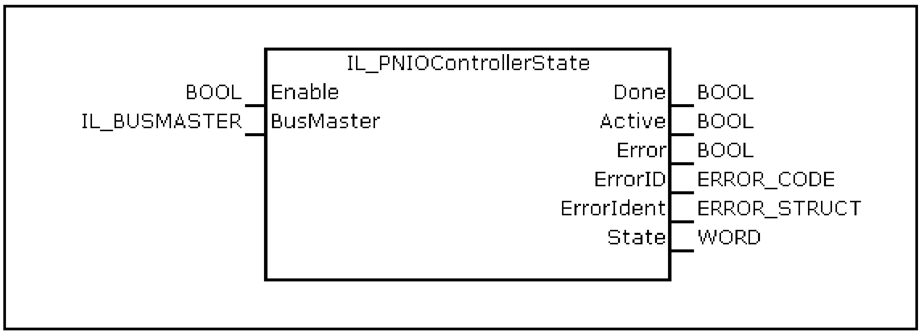


Fig. 11-119: FB IL_PNIOControllerState interface

	Name	Type	Comment
VAR_INPUT	Enable	BOOL	Enables processing of the function block (permanent, level-controlled)
	BusMaster	IL_BUSMASTER	Instance of the controller in the order of the configuration tree. If only one PROFINET IO controller is available, the value of this controller is always IL_BUSMASTER_0.
VAR_OUTPUT	Done	BOOL	Processing completed without error, output data valid
	Active	BOOL	Processing active
	Error	BOOL	Processing completed with error
	ErrorID	ERROR_CODE	Describes the diagnosis in the event of an error (from RIL_CommonTypes)
	ErrorIdent	ERROR_STRUCT	Detailed diagnosis (ErrorTable PROFINETIO_TABLE)
	State	WORD	State of the PROFINET IO controller, see below.

Fig. 11-120: FB IL_PNIOControllerState I/O interface

The "State" output provides a rough diagnostic classification in binary code.

The bits are contained in the RIL_FieldbusTypes.library as global "Fieldbus-States" constants.

Output "State" (WORD)

Bit no.	Description	Description
0	IL_FBM_BUS_OFF	Field bus not ready
1	IL_FBM_MASTER_FAILURE	Bus master failure
2	IL_FBM_SLAVE_NOT_REACHABLE	One or more field bus slaves cannot be reached (slave not reachable)
3	IL_FBM_SLAVE_DIAGNOSIS	One or more field bus slaves signal diagnoses (slave diag)
4	IL_FBM_SLAVE_ERROR	One or more field bus slaves signal errors (slave error)

Fig. 11-121: Elements of output State

Min./max. and default values of the inputs

Name	Type	Min. value	Max. value	Default value	Effective
Enable	BOOL			FALSE	Continuous
BusMaster	IL_BUSMASTER	IL_BUSMASTER_0	IL_BUSMASTER_4	IL_BUSMASTER_0	Rising edge at Enable

Fig. 11-122: Minimum / maximum values and default values of the IL_PNIOControllerState inputs

Functional Description

The function block retrieves state information about a PROFINET controller and returns it as bit string in "State". The function block is completely processed in one PLC cycle.

Error Handling

The function block uses the error table **PROFINETIO_TABLE** (ERROR_TABLE = 16#0200).

It can generate the following error messages in Additional1 and Additional2:

ErrorID	Additional1	Additional2	Remarks
SYSTEM_ERROR	16#0101	internal use	Controller not found in Configurationtable
COMMUNICATION_ERROR	16#0003	*PROFINET IO Controller Diagnosis* ● see General Error Codes, BusMaster , page 216, ● see General Error Codes, BusMaster , page 216.	

Fig. 11-123: Error codes FB IL_PNIOControllerState

11.5.3 IL_PNIOControllerStateDetails

Brief Description

This function block determines the detailed state of the Profinet controller stack. It provides information on

- state of the communication stack,
- the state of the bus communication,
- error counters,
- an overview on configured and active adapters reporting a diagnostics.

Target system	Library
MLC 10VRS and higher	RIL_ProfinetIO.compiled-library
MTX 10VRS and higher	RIL_ProfinetIO.compiled-library
IndraLogic L65 10VRS and above	RIL_ProfinetIO.compiled-library

Fig.11-124: FB IL_PNIOControllerStateDetails reference table

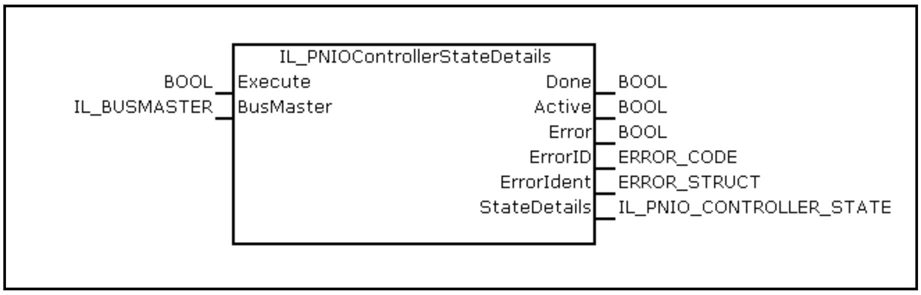


Fig.11-125: FB IL_PNIOControllerStateDetails interface

	Name	Type	Comment
VAR_INPUT	Execute	BOOL	Processing of the function block enabled (once, edge-controlled)
	BusMaster	IL_BUSMASTER	Instance of the controller in the order of the configuration tree. If only one PROFINET IO controller is available, the value of this controller is always IL_BUSMASTER_0.
VAR_OUTPUT	Done	BOOL	Processing completed without error, output data valid
	Active	BOOL	Processing not yet completed, output data invalid
	Error	BOOL	Processing completed with error
	ErrorID	ERROR_CODE	Describes the diagnosis in the event of an error (from RIL_CommonTypes)
	ErrorIdent	ERROR_STRUCT	Detailed diagnosis (ErrorTable PROFINETIO_TABLE)
	StateDetails	IL_PNIO_CONTROLLER_STATE	Detailed information about the state of the controller stack See also IL_PNIO_CONTROLLER_STATE, page 214

Fig.11-126: FB IL_PNIOControllerStateDetails I/O interface

Min./max. and default values of the inputs

Name	Type	Min. value	Max. value	Default value	Effective
Execute	BOOL			FALSE	Continuous
BusMaster	IL_BUSMASTER	IL_BUSMASTER_0	IL_BUSMASTER_4	IL_BUSMASTER_0	Rising edge at "Execute"

Fig.11-127: Minimum / maximum values and default values of the IL_PNIOControllerStateDetails inputs

Functional Description

The function block provides current state information of the PROFINET controller.

The comparison of the following values is of particular interest for evaluation of the bus system: number of configured devices (NumOfConfigSlaves), number of devices present at the PROFINET (NumOfActiveSlaves) and number of devices with present diagnostic message (NumOfDiagSlaves).

Error Handling

The function block uses error table **PROFINETIO_TABLE** (ERROR_TABLE = 16#0200).

It can generate the following error messages in Additional1 and Additional2:

ErrorID	Additional1	Additional2	Remarks
SYSTEM_ERROR	16#0101	internal use	Controller not found in Configurationtable
COMMUNICATION_ERROR	16#0003	*PROFINET IO Controller Diagnosis* • see General Error Codes, BusMaster , page 216, • see General Error Codes, BusMaster , page 216.	

Fig. 11-128: Error codes FB IL_PNIOControllerStateDetails

11.5.4 IL_PNIORemoteDeviceState

Brief Description

This function block determines the basic state of a device connected to the controller.

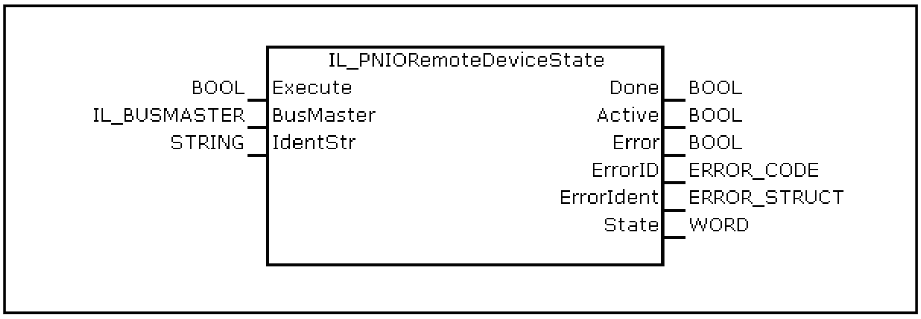


Fig. 11-129: IL_PNIORemoteDeviceState

	Name	Type	Comment
VAR_INPUT	Execute	BOOL	Activates the function block; if activated, a function block cannot be interrupted.
	BusMaster	IL_BUSMASTER	Instance of the controller in the order of the configuration tree. Allowed values 0..4; default: 0
	IdentStr	STRING	Station name of the IO device to be accessed.
VAR_OUTPUT	Done	BOOL	Finished message (successful)
	Active	BOOL	Activity display
	Error	BOOL	Processing completed with error, output data invalid
	ErrorID	ERROR_CODE	Standardized rough classification of the error
	ErrorIdent	ERROR_STRUCT	Function-block-specific detailed information on the error
	State	WORD	State information of the slave at the field bus, see below

Fig. 11-130: Function block IL_PNIORemoteDeviceState

Output "State" (WORD)

Bit no.	Description	Description
0	IL_RBS_SLAVE_NOT_REACHABLE	Device not existent (bus slave state – slave not existent)
1	IL_RBS_SLAVE_NOT_READY	Device not ready(bus slave state – slave not ready)
2	IL_RBS_SLAVE_CONFIG_ERROR	Device configuration error (bus slave state – slave configuration error)
3	IL_RBS_SLAVE_DIAGNOSIS	Device diagnosis available (bus slave state – slave diagnoses available)
4	IL_FBS_SLAVE_ERROR	General device error

Min./max. and default values of the inputs

Name	Type	Min. value	Max. value	Default value	Effective
Execute	BOOL			FALSE	Continuous
BusMaster	IL_BUSMASTER	IL_BUSMASTER_0	IL_BUSMASTER_4	IL_BUSMASTER_0	Rising edge at "Execute"
IdentStr	STRING			’’	Rising edge at "Execute"

Fig. 11-131: Minimum / maximum values and default values of the IL_PNIOControllerState inputs

Functional Description

The bit-encoded state string of the PROFINET device is evaluated. The IL_PNIORemoteDeviceStateDetails service can be used for further information. The device is selected by specifying the PROFINET device in the "IdentStr" variable. The devices must be configured and active so that they can be observed.

Error Handling

The function block uses error table PROFINETIO_TABLE (ErrorIdent.Table = 16#0200). It can generate the following error messages in Additional1 and Additional2:

ErrorID	Additional1	Additional2	
INPUT_RANGE_ERROR	16#0001	1	IdentStr invalid
SYSTEM_ERROR	16#0101	internal use	Controller not found in Configurationtable
INPUT_RANGE_ERROR	16#0106	0	Device IdentStr not configured

Fig. 11-132: Error codes FB IL_PNIORemoteDeviceState

11.5.5 IL_PNIORemoteDeviceStateDetails

Brief Description

The function block determines the detailed PROFINET IO state of a device configured and connected to the controller.

The PROFINET data structures are summarized in general diagnostic information.

Target system	Library
MLC 12VRS and higher	RIL_ProfinetIO.compiled-library
IndraLogic L65 12VRS and above	RIL_ProfinetIO.compiled-library

Fig. 11-133: Reference table of FB IL_PNIORemoteDeviceStateDetails

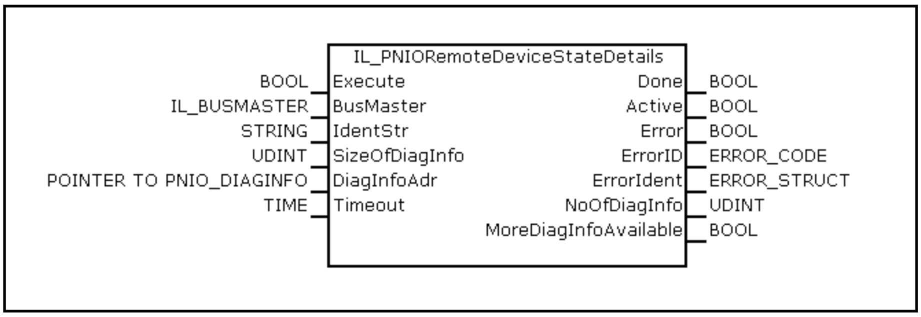


Fig. 11-134: Interface of FB IL_PNIORemoteDeviceStateDetails

	Name	Type	Comment
VAR_INPUT	Execute	BOOL	Processing of the function block enabled (once, edge-controlled)
	BusMaster	IL_BUSMASTER	Instance of the controller in the order of the configuration tree. If only one PROFINET IO controller is available, the value of this controller is always IL_BUSMASTER_0.
	IdentStr	STRING	Identification string of the field bus participant
	SizeOfDiagInfo	UDINT	Maximum size of the data range of "DiagInfoAdr". Must be an integer multiple of SIZEOF(PNIO_DIAGINFO).
	DiagInfoAdr	POINTER TO PNIO_DIAGINFO	Points to the data storage for the evaluated diagnostic data of the device. This should be a pointer to an array of type PNIO_DIAGINFO, page 204,.
	Timeout	TIME	Timeout monitoring of the function block in ms. T#0ms = Timeout monitoring not active
VAR_OUTPUT	Done	BOOL	Processing completed without error, output data valid
	Active	BOOL	Processing not yet completed, output data invalid
	Error	BOOL	Processing completed with error
	ErrorID	ERROR_CODE	Describes the diagnosis in the event of an error (from RIL_CommonTypes)
	ErrorIdent	ERROR_STRUCT	Detailed diagnosis (ErrorTable PROFINETIO_TABLE)
	NoOfDiagInfo	UDINT	Number of evaluations entered in DiagInfoAdr
	MoreDiagInfoAvailable	BOOL	There is additional diagnostic information that does not have any place in the data structure provided.

Fig. 11-135: FB IL_PNIORemoteDeviceStateDetails I/O interface

Min./max. and default values of the inputs

Name	Type	Min. value	Max. value	Default value	Effective
Execute	BOOL			FALSE	Continuous
BusMaster	IL_BUSMASTER	IL_BUSMASTER_0	IL_BUSMASTER_4	IL_BUSMASTER_0	Rising edge at "Execute"

Name	Type	Min. value	Max. value	Default value	Effective
IdentStr	STRING			’’	Rising edge at "Execute"
SizeOfDiagInfo	UDINT	0	n.def.	0	Rising edge at "Execute"
DiagInfoAdr	POINTER TO PNIO_DIAGINFO			0	Rising edge at "Execute"
Timeout	TIME	T#0ms	T#3600s	T#2s	Rising edge at "Execute"

Fig. 11-136: Minimum / maximum values and default values of the IL_PNIOControllerStateDetails inputs

Functional Description

The state of the device is determined by acyclic read accesses.

The number of accesses required may vary depending on the state. This may have an effect on the flow rate of other acyclic accesses to this device. More than one diagnostic messages may be present at a time. More than one diagnostic messages may be present particularly with device which provide a channel-granular diagnosis (1 diagnosis per IO point).

Output parameter "MoreDiagInfoAvailable" indicates that only a part of the present diagnostic messages has place in the entries provided.

Error Handling

The function block uses error table **PROFINETIO_TABLE** (ERROR_TABLE = 16#0200).

It can generate the following error messages in Additional1 and Additional2:

ErrorID	Additional1	Additional2	
INPUT_RANGE_ERROR	16#0001	0	Parametererror
INPUT_RANGE_ERROR	16#0001	1	IdentStr invalid
INPUT_RANGE_ERROR	16#0001	2	DiagInfoAdr not set
INPUT_RANGE_ERROR	16#0001	3	SizeOfDiagInfo not set
INPUT_RANGE_ERROR	16#0001	4	SizoOfDiagInfo is not a multiple of size of PNIO_DIAGINFO
SYSTEM_ERROR	16#0101	internal use	Controller not found in Configurationtable
COMMUNICATION_ERROR	16#0104	0	timeout
COMMUNICATION_ERROR	16#0106	0	Device ‘IdentStr’ not configured
COMMUNICATION_ERROR	16#0003	*PROFINET IO Controller Diagnosis* ● see General Error Codes, BusMaster , page 216, ● see General Error Codes, BusMaster , page 216.	

Fig. 11-137: Error codes FB IL_PNIORemoteDeviceStateDetails

11.5.6 PNIO_DIAGINFO

Brief Description

The data structure contains a summary of the PROFINET state and diagnostic information. It consists of the type (DiagType), the location (slot, subslot, channel) and, if necessary, an error type of the event.

Structure element	Value	Description
DiagType	PNIO_DIAGTYPE	Event type
Slot	UINT	Slot where the event has occurred.
Subslot	UINT	Subslot where the event has occurred.
Channel	UINT	Channel where the event has occurred. Here, a channel is the individual IO point which may deliver a diagnosis (e.g., cable break).
ChannelErrorType	PNIO_CHANNELERROR	Event or diagnosis

Fig. 11-138: Structure of PNIO_DIAGINFO

11.5.7 PNIO_DIAGTYPE

Brief Description

The PNIO_DIAGTYPE enumeration type serves to label the events that can be diagnosed.

☞ Surplus modules may be present for PROFINET devices. These modules do not generate any diagnostic message.

Element	Value	Description
NO_DIAGNOSIS	0	No event/diagnosis
DIAGNOSIS_APPEARS	1	There is a diagnostic message. Evaluate PNIO_CHANNELERROR
MODULE_MISSING	2	The module designated by Slot/Subslot is configured but not present at the device
MODULE_WRONG	3	The module designated by Slot/Subslot does not correspond to the configured module
SUBMODULE_STATE	4	The module designated by Slot/Subslot delivers the state filed in ChannelErrorType

Fig. 11-139: Enumeration type PNIO_DIAGTYPE

11.5.8 PNIO_CHANNELERROR

Brief Description

Enumeration type PNIO_ CHANNELERROR generates the ChannelErrorType according to IEC 61158-6-10 V23 6.2.7.2.

There may be additional manufacturer-specific values.

Element	Value	Description
ChanErr_Unknown	0	Unknown error
ChanErr_ShortCircuit	1	Short circuit
ChanErr_Undervoltage	2	Voltage too low
ChanErr_Overvoltage	3	Voltage too high
ChanErr_Overtemperature	4	Overtemperature
ChanErr_Overload	5	Overload
ChanErr_LineBreak	6	Cable Break
ChanErr_UpperLimit	7	Value range exceeded
ChanErr_LowerLimit	8	Value range fallen below

Element	Value	Description
ChanErr_Error	9	General error
ChanErr_SimulationActive	10	Simulation mode

Fig. 11-140: Enumeration type PNIO_CHANNELERROR

11.5.9 IL_PNIOGetConfigDeviceNameList

Brief Description

The function block reads the names of all configured PROFINETIO devices.

The user must provide a STRING ARRAY for accepting the DeviceNames.

These device names can be used to activate input "IdentStr".

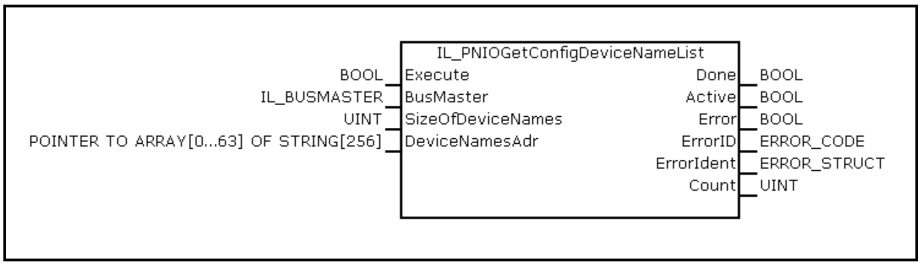


Fig. 11-141: IL_PNIOGetConfigDeviceNameList

	Name	Type	Comment
VAR_INPUT	Execute	BOOL	Processing of the function block enabled (once, edge-controlled)
	BusMaster	IL_BUSMASTER	Instance of the field bus master in the order of the configuration tree.
	SizeOfDeviceNames	UINT	Size of array DeviceNamesAdr in bytes
	DeviceNamesAdr	POINTER TO ARRAY [0..63] OF STRING(255)	Pointer to an array of strings where the configured device names are entered. The size of this array must be transferred in parameter "SizeOfDeviceNames"
VAR_OUTPUT	Done	BOOL	Processing completed without error, output data valid
	Active	BOOL	Processing not yet completed, output data invalid
	Error	BOOL	Processing completed with error, output data invalid
	ErrorID	ERROR_CODE	Standardized rough classification of the error
	ErrorIdent	ERROR_STRUCT	Detailed diagnosis (ErrorTable PROFINETIO_TABLE)
	Count	UINT	Number of valid device names in DeviceNamesAdr

Fig. 11-142: Function block IL_PNIOGetConfigDeviceNameList

Min./max. and default values of the inputs

Name	Type	Min. value	Max. value	Default value	Effective
Execute	BOOL			FALSE	Continuous
BusMaster	IL_BUSMASTER	IL_BUSMASTER_0	IL_BUSMASTER_4	IL_BUSMASTER_0	Rising edge at "Execute"

Name	Type	Min. value	Max. value	Default value	Effective
SizeOfDevice-Names	UINT	0	n.def.	0	Rising edge at "Execute"
DeviceNamesAdr	POINTER TO ARRAY [0..63] OF STRING[256]			0	Rising edge at "Execute"

Fig. 11-143: Min./max. values and default values of the L_PNIOGetConfigDeviceNameList inputs

Functional Description

The function block searches the opened IO configuration for the devices which are located below the selected PROFINET controller.

Deactivated devices are not included in the list. These device names can be used as "IdentStr" input parameters for other function blocks.

Error Handling

The function block uses error table **PROFINETIO_TABLE** (ERROR_TABLE = 16#0200).

It can generate the following error messages in Additional1 and Additional2:

ErrorID	Additional1	Additional2	
INPUT_RANGE_ERROR	16#0001	0	DeviceNamesAdr: list is too short
INPUT_RANGE_ERROR	16#0001	1	DeviceNamesAdr not set
INPUT_RANGE_ERROR	16#0001	2	SizeOfDeviceNames not set
SYSTEM_ERROR	16#0101	internal use	Controller not found in Configurationtable

Fig. 11-144: Error codes FB IL_PNIOGetConfigDeviceNameList

11.5.10 IL_PNIOGetDiagDeviceNameList

Brief Description

This function block reads the names of all PROFINETIO devices that deliver a diagnostic message.

The user must provide a STRING ARRAY for accepting the device names.

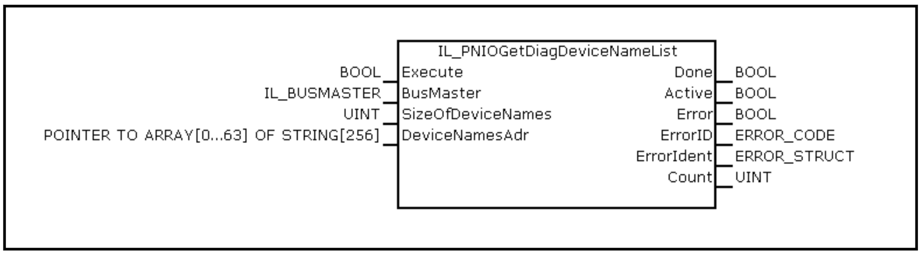


Fig. 11-145: IL_PNIOGetDiagDeviceNameList

	Name	Type	Comment
VAR_INPUT	Execute	BOOL	Processing of the function block enabled (once, edge-controlled)
	BusMaster	IL_BUSMASTER	Instance of the field bus master in the order of the configuration tree.
	SizeOfDeviceNames	UINT	Size of array DeviceNamesAdr in bytes

	Name	Type	Comment
	DeviceNamesAdr	POINTER TO ARRAY [0..63] OF STRING(255)	Pointer to an array of strings where the DeviceNames of the devices which signal diagnosis are entered. The size of this array must be transferred in parameter "SizeOfDeviceNames".
VAR_OUTPUT	Done	BOOL	Processing completed without error, output data valid
	Active	BOOL	Processing not yet completed, output data invalid
	Error	BOOL	Processing completed with error, output data invalid
	ErrorID	ERROR_CODE	Standardized rough classification of the error
	ErrorIdent	ERROR_STRUCT	Function-block-specific detailed information on the error
	Count	UINT	Number of devices with active diagnosis in DeviceNamesAdr

Fig.11-146: Function block IL_PNIOGetDiagDeviceNameList

Min./max. and default values of the inputs

Name	Type	Min. value	Max. value	Default value	Effective
Execute	BOOL			FALSE	Continuous
BusMaster	IL_BUSMASTER	IL_BUSMASTER_0	IL_BUSMASTER_4	IL_BUSMASTER_0	Rising edge at "Execute"
SizeOfDeviceNames	UINT	0	n.def.	0	Rising edge at "Execute"
DeviceNamesAdr	POINTER TO ARRAY [0..63] OF STRING[256]			0	Rising edge at "Execute"

Fig.11-147: Min./max. values and default values of the IL_PNIOGetDiagDeviceNameList inputs

Functional Description

All PROFINET devices currently signaling diagnosis are determined from the configuration table.

The device name is transferred.

☞ This list does not include any device with other errors.

Error Handling

The function block uses error table **PROFINETIO_TABLE** (ERROR_TABLE = 16#0200).

It can generate the following error messages in Additional1 and Additional2:

ErrorID	Additional1	Additional2	
INPUT_RANGE_ERROR	16#0001	0	DeviceNamesAdr: list i s too short
INPUT_RANGE_ERROR	16#0001	1	DeviceNamesAdr not set

ErrorID	Additional1	Additional2	
INPUT_RANGE_ERROR	16#0001	2	SizeOfDeviceNames not set
SYSTEM_ERROR	16#0101	internal use	Controller not found in Configurationtable

Fig.11-148: Error codes FB IL_PNIOGetDiagDeviceNameList

11.5.11 IL_PNIOReadRecord

Brief Description

This function blocks reads acyclic data from devices.

The devices are addressed via their Profinet names in "IdentStr".

Target system	Library
MLC 10VRS and higher	RIL_ProfinetIO.compiled-library
MTX 10VRS and higher	RIL_ProfinetIO.compiled-library
IndraLogic L65 10VRS and above	RIL_ProfinetIO.compiled-library

Fig.11-149: FB IL_PNIOReadRecord reference table

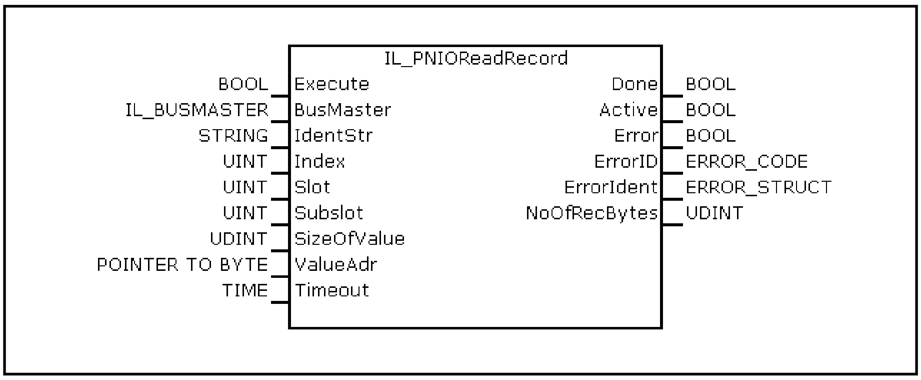


Fig.11-150: FB IL_PNIOReadRecord interface

	Name	Type	Comment
VAR_INPUT	Execute	BOOL	Processing of the function block enabled (once, edge-controlled)
	BusMaster	IL_BUSMASTER	Instance of the field bus master in the order of the configuration tree.
	IdentStr	STRING	Identification string of the field bus participant
	Index	UINT	Index of the addresses data set. Indices 16#8000 to 16#FFFF are reserved for PROFINET. Indices 0 to 16#7FFF are user- or application-specific.
	Slot	UINT	Addressing of the module slot. If the device is a PROFINET device, a slot addresses a logical or a physical module (for example, an I/O module).
	Subslot	UINT	Addressing of the submodule. A module can consist of several logical or physical submodules. At least the submodule with number 1 is present in each module (slot).
	SizeOfValue	UDINT	Size of the data range addressed with ValueAdr.

	Name	Type	Comment
	ValueAdr	POINTER TO BYTE	Initial address of the data range
	Timeout	TIME	Timeout monitoring of the function block in ms. T#0ms = Timeout monitoring not active
VAR_OUTPUT	Done	BOOL	Processing completed without error, output data valid
	Active	BOOL	Processing not yet completed, output data invalid
	Error	BOOL	Processing completed with error, output data invalid
	ErrorID	ERROR_CODE	Standardized rough classification of the error
	ErrorIdent	ERROR_STRUCT	Function-block-specific detailed information on the error
	NoOf RecBytes	UDINT	Number of bytes received in the data range

Fig.11-151: FB IL_PNIOReadRecord I/O interface

Min./max. and default values of the inputs

Name	Type	Min. value	Max. value	Default value	Effective
Execute	BOOL			FALSE	Continuous
BusMaster	IL_BUSMASTER	IL_BUSMASTER_0	IL_BUSMASTER_4	IL_BUSMASTER_0	Rising edge at "Execute"
IdentStr	STRING			’’	Rising edge at "Execute"
Index	UINT	0	16#ffff	0	Rising edge at "Execute"
Slot	UINT	1	16#7fff	0	Rising edge at "Execute"
Subslot	UINT	1	16#8fff	0	Rising edge at "Execute"
SizeOfValue	UDINT	0	n.def.	0	Rising edge at "Execute"
ValueAdr	POINTER TO BYTE	n.def.	n.def.	0	Rising edge at "Execute"
Timeout	TIME	T#0ms	T#3600s	T#2s	Rising edge at "Execute"

Fig.11-152: Minimum / maximum values and default values of the IL_PNIOReadRecord inputs

Functional Description

The device is addressed via the PROFINET device name in "IdentStr". It must be part of the configuration of the controller. The data set on the PROFINET device is addressed via the slot-subslot combination as well as via the index.

Slot and subslot address the various logical or physical modules of a device. The index serves to select the information type. The range from 16#8000 to 16#ffff is assigned by IEC 61158-6-10 V23. The value range from 0 to 16#7fff can be assigned the device manufacturer. For more information, please refer to the documentation of the particular device.

For example, the I&M0 data of a device can be addressed with index 16#AFF0 as well as slot 0 and subslot 1.

The data is requested by the PROFINET IO device in the maximum allowed length (SizeOfValue). The number of valid data bytes actually delivered by the device is returned in NoOfRecBytes.

Error Handling

The function block uses error table **PROFINETIO_TABLE** (ERROR_TABLE = 16#0200).

It can generate the following error messages in Additional1 and Additional2:

ErrorID	Additional1	Additional2	
INPUT_RANGE_ERROR	16#0001	0	ValueAdr is not set
INPUT_RANGE_ERROR	16#0001	1	SizeOfValue is not set
INPUT_RANGE_ERROR	16#0001	2	IdentStr invalid
SYSTEM_ERROR	16#0101	internal use	Controller not found in Configurationtable
COMMUNICATION_ERROR	16#0104	0	timeout
STATE_MACHINE_ERROR	16#0105	16#0001 16#0002	statemachine error
COMMUNICATION_ERROR	16#0003	*PROFINET IO Controller Diagnosis* ● see General Error Codes, BusMaster , page 216, ● see Special Error Codes, PROFINET I/O Controller , page 218.	
COMMUNICATION_ERROR	16#0004	*PROFINET IO Status* ● see Special Error Codes, PROFINET I/O State , page 218,	

Fig.11-153: Error codes FB IL_PNIOReadRecord

11.5.12 IL_PNIOWriteRecord

Brief Description

This function blocks writes acyclic data to devices.

The devices are addressed via their Profinet name. The devices must be available in the configuration.

Target system	Library
MLC 10VRS and higher	RIL_ProfinetIO.compiled-library
MTX 10VRS and higher	RIL_ProfinetIO.compiled-library
IndraLogic L65 10VRS and above	RIL_ProfinetIO.compiled-library

Fig.11-154: FB IL_PNIOWriteRecord reference table

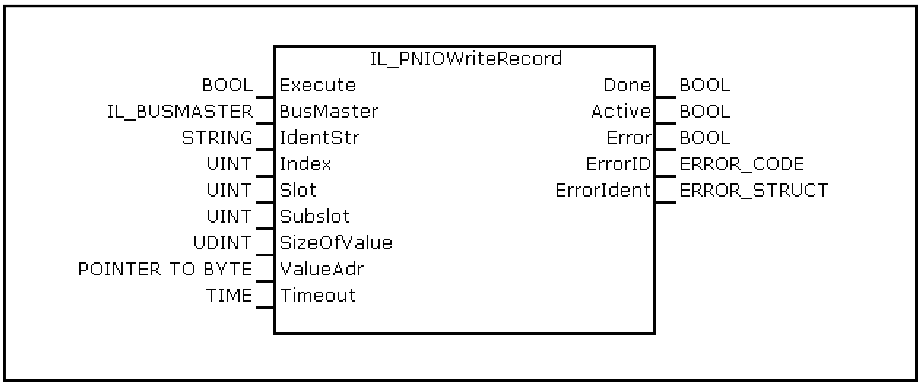


Fig.11-155: FB IL_PNIOWriteRecord interface

	Name	Type	Comment
VAR_INPUT	Execute	BOOL	Processing of the function block enabled (once, edge-controlled)
	BusMaster	IL_BUSMASTER	Instance of the controller in the order of the configuration tree. If only one PROFINET IO controller is available, the value of this controller is always IL_BUSMASTER_0.
	IdentStr	STRING	Station name of the IO device to be accessed.
	Index	UINT	Index of the addresses data set. Indices 16#8000 to 16#FFFF are reserved for Profinet. Indices 0 to 16#7FFF are user- or application-specific.
	Slot	UINT	Addressing of the module slot. In a modular PNIO device, Slot is used to address the physical modules. In a compact PNIO device, Slot addresses a logic function or a virtual module.
	Subslot	UINT	Addressing of the submodule. Subslot can be used to address the physical interfaces of the submodules of a module. In general, Subslot is the second structure level of a PNIO device.
	SizeOfValue	UDINT	Maximum size of the data range of "ValueAdr"
	ValueAdr	POINTER TO BYTE	Pointer to the data storage
	Timeout	TIME	Timeout for abortion of the function.
VAR_OUTPUT	Done	BOOL	Processing completed without error, output data valid
	Active	BOOL	Processing not yet completed, output data invalid
	Error	BOOL	Processing completed with error
	ErrorID	ERROR_CODE	Describes the diagnosis in the event of an error (from RIL_CommonTypes)
	ErrorIdent	ERROR_STRUCT	Detailed diagnosis (ErrorTable PROFINETIO_TABLE)

Fig.11-156: FB IL_PNIOWriteRecord I/O interface

Min./max. and default values of the inputs

Name	Type	Min. value	Max. value	Default value	Effective
Execute	BOOL			FALSE	Continuous
BusMaster	IL_BUSMASTER	IL_BUSMASTER_0	IL_BUSMASTER_4	IL_BUSMASTER_0	Rising edge at "Execute"
IdentStr	STRING			’’	Rising edge at "Execute"
Index	UINT	0	16#ffff	0	Rising edge at "Execute"
Slot	UINT	1	16#7fff	0	Rising edge at "Execute"
Subslot	UINT	1	16#8fff	0	Rising edge at "Execute"
NoOfBytes	UDINT	0	n.def.	0	Rising edge at "Execute"
ValueAdr	POINTER TO BYTE	n.def.	n.def.	0	Rising edge at "Execute"
Timeout	TIME	T#0ms	T#3600s	T#2s	Rising edge at "Execute"

Fig.11-157: Minimum / maximum values and default values of the IL_PNIOWriteRecord inputs

Functional Description

The device is addressed via the PROFINET device name. It must be part of the configuration of the controller. The data set on the PROFINET device is addressed via the slot-subslot combination as well as via the index.

Slot and subslot address the various logical or physical modules of a device. The index serves to select the information type. The range from 16#8000 to 16#ffff is assigned by IEC 61158-6-10 V23. The value range from 0 to 16#7fff can be assigned the device manufacturer. For more information, please refer to the documentation of the particular device.

The specified data range "ValueAdr" is transferred to the PROFINET device in its complete size "SizeOfValue". The reaction of the device to exceeded ranges is manufacturer-specific.

Error Handling

The function block uses error table **PROFINETIO_TABLE** (ERROR_TABLE = 16#0200).

It can generate the following error messages in Additional1 and Additional2:

ErrorID	Additional1	Additional2	
INPUT_RANGE_ERROR	16#0001	0	ValueAdr is not set
INPUT_RANGE_ERROR	16#0001	1	SizeOfValue is not set
INPUT_RANGE_ERROR	16#0001	2	IdentStr is not set
SYSTEM_ERROR	16#0101	internal use	Controller not found in Configurationtable
COMMUNICATION_ERROR	16#0104	0	timeout
STATE_MACHINE_ERROR	16#0105	0	statemachine error

ErrorID	Additional1	Additional2	
COMMUNICATION_ERROR	16#0003	*PROFINET IO Controller Diagnosis* ● See General Error Codes, BusMaster , Seite 216, ● See Special Error Codes, PROFINET I/O Controller , page 218.	
COMMUNICATION_ERROR	16#0004	*PROFINET IO Status* ● See Special Error Codes, PROFINET I/O State , page 218,	

Fig.11-158: Error codes FB IL_PNIOWriteRecord

11.5.13 IL_BUSMASTER

Brief Description

The IL_BUSMASTER enumeration type of the RIL_FieldbusTypes.library serves to select the appropriate PROFINET IO controller. In addition to the onboard real-time Ethernet interface, other controllers can also be available in an XLC/MLC system via function modules.

The PROFINET IO controllers are distinguished on the basis of their ascending order in the configuration.

Indexes are assigned to the controllers in the following order:

- Onboard master
- Function module 1
- Function module 2,
- Function module 3,
- Function module 4

☞ If, for example, the onboard master is not a PROFINET IO controller while a PROFINET IO controller is configured on the FM1, IL_BUSMASTER_0 is the value assigned to the controller on the FM1.

Element	Value	Description
IL_BUSMASTER_0	0	First PN controller
IL_BUSMASTER_1	1	Second PN controller
IL_BUSMASTER_2	2	Third PN controller
IL_BUSMASTER_3	3	Fourth PN controller
IL_BUSMASTER_4	4	Fifth PN controller

Fig.11-159: IL_BUSMASTER enumeration type

11.5.14 IL_PNIO_CONTROLLER_STATE

Brief Description

Data structure IL_PNIO_CONTROLLER_STATE contains diagnostic information about

- state of the communication stack,
- the state of the bus communication,
- error counters,
- an overview on configured and active adapters reporting a diagnostics.

Structure element	Type	Description
CommunicationCOS	DWORD	Status of the communication stack, see below
CommunicationState	IL_FBUS_COMMUNICATION_STATE	Status of the communication Refer to IL_FBUS_COMMUNICATION_STATE on page 215.
Version	UINT	Version of the data structure of the diagnostic information
Watchdog	UINT	Watchdog timeout between PLC and field bus connection in ms
ErrorCount	UDINT	Error counter. Counts the number communication errors that have occurred (both internal errors and communication errors)
SlaveState	IL_FBUS_SLAVE_STATE	Indicates whether the PROFINET IO controller has an active connection to all configured slaves. Refer to IL_FBUS_SLAVE_STATE on page 216.
NumOfConfigSlaves	UDINT	Number of configured slaves
NumOfActiveSlaves	UDINT	Number of active slaves, i.e., slaves to which the master has established a communication connection.
NumOfDiagSlaves	UDINT	Number of slaves with an active diagnostics

Fig. 11-160: Data structure IL_PNIO_CONTROLLER_STATE

Structure element CommunicationCOS

Bit no.	Description	Description
0	COMM_COS_READY	Protocol stack has been started and waits for configuration
1	COMM_COS_RUN	Protocol stack has been configured and waits for bus connection
2	COMM_COS_BUS_ON	Protocol stack may communicate on the bus
3	COMM_COS_CONFIG_LOCKED	Bus configuration is locked and cannot be changed
4	COMM_COS_CONFIG_NEW	New bus configuration is available but not initialized yet
5	COMM_COS_RESTART_REQUIRED	Communication restart required
6	COMM_COS_RESTART_REQUIRED_ENABLE	Communication restart possible
		FALSE assigned to remaining ones

The state of the communication stack is binary encoded in output CommunicationCOS.

The bits are contained in the RIL_FIELDBUS_TYPES library in the form of defines (PNIO_Defines).

11.5.15 IL_FBUS_COMMUNICATION_STATE

Brief Description

The IL_FBUS_COMMUNICATION_STATE enumeration type is used to show the status of the field bus communication.

Element	Value	Description
COMM_STATE_NOT_CONFIGURED	16#00000001	Communication is not configured
COMM_STATE_STOP	16#00000002	Communication is in "Stop" state
COMM_STATE_IDLE	16#00000003	Communication is in "Idle" state
COMM_STATE_OPERATE	16#00000004	Communication in cyclic operation

Fig.11-161: IL_FBUS_COMMUNICATION_STATE enumeration type

11.5.16 IL_FBUS_SLAVE_STATE

The IL_FBUS_SLAVE_STATE enumeration type shows the status of the communication relations to the slaves. That means that it indicates whether the field bus master has an active connection to all configured slaves.

Element	Value	Description
SLAVE_STATE_OK	16#00000001	Communication to all active slaves available
SLAVE_STATE_FAILED	16#00000002	Communication error to at least one slave

Fig.11-162: IL_FBUS_SLAVE_STATE enumeration slave

11.5.17 General Error Codes, BusMaster

The error codes shown apply to all bus masters, i.e., they are bus-independent.

The error codes are signaled as ERROR_STRUCT, Additional2 with ERROR_STRUCT, Additional1= 16#0003.

Additional2	
16#800A0001	Invalid pointer (NULL) passed to driver
16#800A0002	No board with the given name / index available
16#800A0003	No channel with the given index available
16#800A0004	Invalid handle passed to driver
16#800A0005	Invalid parameter
16#800A0006	Invalid command
16#800A0007	Invalid buffer size
16#800A0008	Invalid access size
16#800A0009	Function failed
16#800A000A	File could not be opened
16#800A000B	File size is zero
16#800A000C	Insufficient memory to load file
16#800A000D	File checksum compare failed
16#800A000E	Error reading from file
16#800A000F	Invalid file type
16#800A0010	Invalid file name
16#800A0011	Driver function not available

Additional2	
16#800A0012	Given buffer is too short
16#800A0013	Failed to map the memory
16#800A0014	No more entries available
16#800B0001	Driver not initialized
16#800B0002	Driver init state error
16#800B0003	Driver read state error
16#800B0004	Command is active on device
16#800B0005	General error during download
16#800B0006	Wrong driver version
16#800B0030	CIFx driver is not running
16#800B0031	Failed to initialize the device
16#800B0032	Channel not initialized (xOpenChannel not called)
16#800B0033	IOControl call failed
16#800B0034	Driver was not opened
16#800C0010	Dual port memory not accessable (board not found)
16#800C0011	Device not ready (ready flag failed)
16#800C0012	Device not running (running flag failed)
16#800C0013	Watchdog test failed
16#800C0015	Error in handshake flags
16#800C0016	Send mailbox is full
16#800C0017	Send packet timeout
16#800C0018	Receive packet timeout
16#800C0019	No packet available
16#800C001A	Mailbox too short
16#800C0020	Reset command timeout
16#800C0021	COM-flag not set
16#800C0022	I/O data exchange failed
16#800C0023	I/O data exchange timeout
16#800C0024	Unknown I/O exchange mode
16#800C0025	Device function failed
16#800C0026	DPM size differs from configuration
16#800C0027	Unknown state mode
16#800C0028	Output port already in use
16#800C0029	Configuration locking timeout
16#800C002A	Configuration unlocking timeout
16#800C002B	Set HOST state timeout

Additional2	
16#800C002C	Clear HOST state timeout
16#800C002D	Timeout during channel initialization
16#800C002E	Set Bus ON Timeout
16#800C002F	Set Bus OFF Timeout

Fig.11-163: General error codes of BusMaster

11.5.18 Special Error Codes, PROFINET I/O Controller

The error codes shown apply to PROFINET I/O controllers.

The error codes are signaled as ERROR_STRUCT, Additional2 with ERROR_STRUCT, Additional1= 16#0003.

Additional2	
16#C0000145	No Ethernet cable plugged in
16#C0000180	The field bus is OFF
16#C00C0060	The acyclic service is acknowledged negatively by the device.
16#C00C0061	The acyclic service failed. The RPC layer detected an error in the received package.
16#C00C0062	The acyclic service failed. An internal error has occurred.
16#C0140041	Invalid characters in the device name

Fig.11-164: Special error codes of the PROFINET I/O controller

11.5.19 Special Error Codes, PROFINET I/O State

The error codes represented are the special error codes of the PROFINET I/O state.

These codes are taken from IEC 61158-6-10, chapter 6.2.5, Coding section related to PNIOStatus.

The error codes represented are signaled in error code Additional2 at a value of 16#0004 in error code Additional1.

The 32-bit value consists of four 4-bit values.

PROFINET I/O state coding

Additional2 screen	Meaning	
16#FF000000	**ErrorCode**	Error code for negative responses
16#00FF0000	**ErrorDecode**	Selection of further decoding
16#0000FF00	**ErrorCode1**	Error codes 1
16#000000FF	ErrorCode2	Error codes 2

Selecting the values for **ErrorCode**. There are other defined values which are, however, not occurring in this context.

ErrorCode	Meaning	
16#DE	ReadResponse	Negative Read.response
16#DF	WriteResponse	Negative Write.response

Selecting the values for **ErrorDecode**. There are other defined values which are, however, not occurring in this context.

ErrorDecode	Meaning	
16#80	PNIORW	For read and write services
16#81	PNIO	For other services

Meaning of **ErrorCode1** with **ErrorDecode "PNIORW"**.

In this case, the meaning of ErrorCode2 is user-specific and will not be explained in more detail.

ErrorCode1

ErrorClass (decimal) Bits 7–4	Meaning	ErrorCode (decimal) Bits 3–0
0 to 9	Not specified	Reserved
10	Application	0 = read error 1 = write error 2 = module failure 3,4,5,6 = not specified 7 = busy 8 = version conflict 9 = feature not supported 10 = user specific 1 …. 15 = User-specific 6
11	Access	0 = invalid index 1 = write length error 2 = invalid slot / subslot 3 = type conflict 4 = invalid area / API 5 = state conflict 6 = access denied 7 = invalid range 8 = invalid parameter 9 = invalid type 10 = backup 11 = User-specific 7 … 15 = User-specific 11

ErrorClass (decimal) Bits 7–4	Meaning	ErrorCode (decimal) Bits 3–0
12	Resource	0 = read constrain conflict 1 = write constrain conflict 2 = resource busy 3 = resource unavailable 4,5,6,7 = not specified 8 = User-specific 12 …. 15 = User-specific 19
13 to 15	User specific	User specific

Fig. 11-165: Selecting the coding of PNIOStatus.ErrorCode

11.6 RIL_ProfinetIODevice.library

11.6.1 General

The controls can be operated as field bus slaves. This special field bus slave is referred to as ProfinetIODevice in the case of ProfinetIO. The present library describes the interfaces of this PronetIODevice to the IEC user program.

Implementation of the library is based on RIL_UTILITIES. The IL_EnableDoneBase and IL_ExecuteDoneBase methods are used and extended or overwritten.

Target systems

The library can be used with the following systems:

Target assembly	Remark
CML65	Onboard / 4 CFL01_1_TP
CML45	Onboard / 4 CFL01_1_TP
CML25	Onboard / 4 CFL01_1_TP

Fig. 11-166: Target systems

The library contains the following components:

Diagnostic function blocks

- IL_PNIODeviceState, page 221, for determining the state of a Profinet device;
- IL_PNIODeviceStateDetails, page 222, for determining the state of the Profinet device stack;
  - using data type
    IL_PNIO_DEVICE_STATE, page 226;
- IL_PNIODeviceStateDetailsXMAC, page 224, state of the Profinet 2-port switch;
  - using data types
    IL_PNIO_DEVICE_XMAC, page 227; and
  - IL_PNIO_DEVICE_XMAC_PORT, page 228.

Addressing the device

Addressing: The Profinet I/O devices are distinguished on the basis of their ascending order in the configuration.

Counting starts with the 0th device, i.e., if there is only one configured device, its device instance is always 0.

See also IL_BUSSLAVE, page 225.

11.6.2 IL_PNIODeviceState

Brief Description

This function block returns a basic diagnosis of the PROFINET I/O device. If a diagnostic message is present, details on the diagnoses can be read via additional FBs (e.g., IL_PNIODeviceStateDetails).

Assignment: Target system/library

Target system	Library
MLC 11VRS and higher	RIL_ProfinetIODevice.compiled-library
MTX 11VRS and above	RIL_ProfinetIODevice.compiled-library
IndraLogic L65 11VRS and above	RIL_ProfinetIODevice.compiled-library

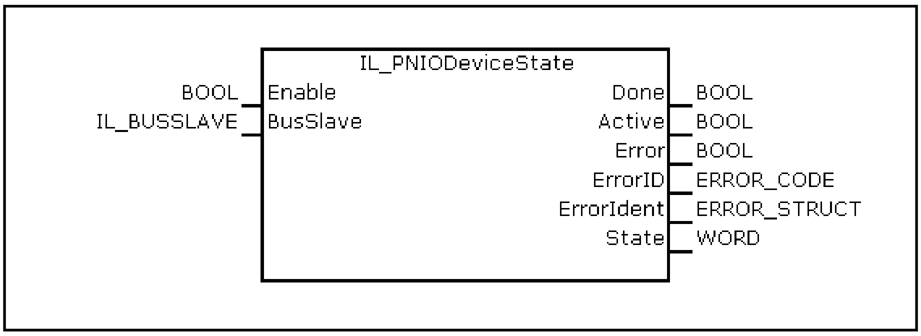


Fig.11-167: IL_PNIODeviceState

	Name	Type	Comment
VAR_INPUT	Enable	BOOL	Enables processing of the function block (permanent, level-controlled)
	BusSlave	IL_BUSSLAVE	Instance of the field bus slave in the order of the configuration tree.
VAR_OUTPUT	Done	BOOL	Processing completed without error, output data valid
	Active	BOOL	Processing not yet completed, output data invalid
	Error	BOOL	Processing completed with error, output data invalid
	ErrorID	ERROR_CODE	Standardized rough classification of the error
	ErrorIdent	ERROR_STRUCT	Function-block-specific detailed information on the error
	State	WORD	State information of the local field bus slave

Fig.11-168: Function block IL_PNIODeviceState

The "State" output provides a rough diagnostic classification in binary code.

The bits are contained in the RIL_FieldbusTypes.library as global "FieldbusStates" constants.

Output "State"

Bit no.	Designation	Description
0	IL_FBS_BUS_OFF	No field bus communication
2	IL_FBS_CONFIG_ERROR	Faulty configuration by the field bus master
4	IL_FBS_SLAVE_ERROR	Device error
5	IL_FBS_UNUSED_MODULE_INFO	Information: Unused I/O module; the I/O modules are not operated by the controller.

Min./max. and default values of the inputs

Name	Type	Min. value	Max. value	Default value	Effective
Enable	BOOL			FALSE	Continuous
BusSlave	IL_BUSSLAVE	IL_BUSSLAVE_0	IL_BUSSLAVE_4	IL_BUSSLAVE_0	Rising edge at Enable

Fig.11-169: Minimum / maximum values and default values of the IL_PNIODeviceState inputs

Functional Description The function block retrieves state information about the PROFINET device and returns it as bit string in "State". The function block is completely processed in one PLC cycle.

Error handling The function block uses error table **PROFINETIO_TABLE** (ERROR_TABLE = 16#0200). It can generate the following error messages in Additional1 and Additional2:

ErrorID	Additional1	Additional2	
SYSTEM_ERROR	16#0101	internal use	Device not found in configuration table

Fig.11-170: Error codes FB IL_PNIODeviceState

11.6.3 IL_PNIODeviceStateDetails

Brief Description This function block determines the detailed status of the PROFINET device stack. It provides information on

- the state of the communication stack,
- the state of the bus communication,
- the previous error values.

Assignment: Target system/library

Target system	Library
MLC 11VRS and higher	RIL_ProfinetIODevice.compiled-library
MTX 11VRS and above	RIL_ProfinetIODevice.compiled-library
IndraLogic L65 11VRS and above	RIL_ProfinetIODevice.compiled-library

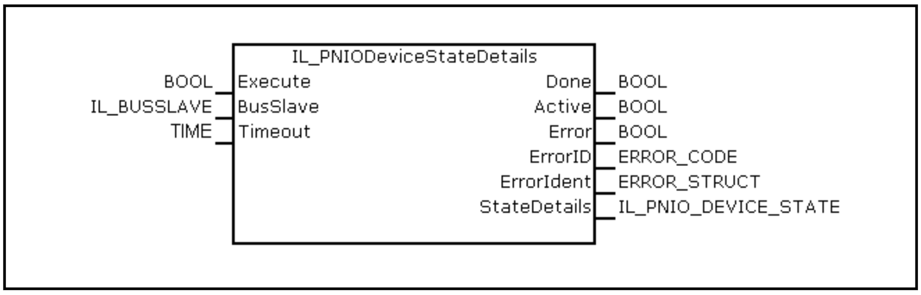


Fig. 11-171: IL_PNIODeviceStateDetails

	Name	Type	Comment
VAR_INPUT	Execute	BOOL	Processing of the function block enabled (once, edge-controlled)
	BusSlave	IL_BUSSLAVE	Instance of the field bus slave in the order of the configuration tree.
	Timeout	TIME	Timeout monitoring of the function block in ms. T#0ms = Timeout monitoring not active
VAR_OUTPUT	Done	BOOL	Processing completed without error, output data valid
	Active	BOOL	Processing not yet completed, output data invalid
	Error	BOOL	Processing completed with error, output data invalid
	ErrorID	ERROR_CODE	Standardized rough classification of the error
	ErrorIdent	ERROR_STRUCT	Function-block-specific detailed information on the error
	StateDetails	IL_PNIO_DEVICE_STATE	Detailed information about the status of the device stack

Fig. 11-172: Function block IL_PNIODeviceStateDetails

Min./max. and default values of the inputs

Name	Type	Min. value	Max. value	Default value	Effective
Execute	BOOL			FALSE	Continuous
BusSlave	IL_BUSSLAVE	IL_BUSSLAVE_0	IL_BUSSLAVE_4	IL_BUSSLAVE_0	Rising edge at Execute
Timeout	TIME	T#0ms	T#3600s	T#2s	Rising edge at "Execute"

Fig. 11-173: Minimum / maximum values and default values of the IL_PNIODeviceStateDetails inputs

Functional Description

When activated, this function block determines the current state of the device once.

Error Handling

The function block uses error table **PROFINETIO_TABLE** (ERROR_TABLE = 16#0200). It can generate the following error messages in Additional1 and Additional2:

ErrorID	Additional1	Additional2	
SYSTEM_ERROR	16#0101	Internal use	Device not in configuration table
COMMUNICATION_ERROR	16#0104	0	Timeout

ErrorID	Additional1	Additional2	
STATE_MACHINE_ERROR	16#0105	0	Error in the state machine
COMMUNICATION_ERROR	16#0003	*PROFINET IO device diagnosis* ● Refer to General Error Codes, Profinet I/O Device on page 228. ● Refer to Special Error Codes, Profinet I/O Device on page 230.	

Fig.11-174: Error codes of function block IL_PNIODeviceStateDetails

11.6.4 IL_PNIODeviceStateDetailsXMAC

Brief Description

This function block determines the detailed status of the Profinet Switch on the netX© communication controller. It provides information on each Ethernet port about

- its connectivity and
- the statistics counter of that port.

Assignment: Target system/library

Target system	Library
MLC 11VRS and higher	RIL_ProfinetIODevice.compiled-library
MTX 11VRS and above	RIL_ProfinetIODevice.compiled-library
IndraLogic L65 11VRS and above	RIL_ProfinetIODevice.compiled-library

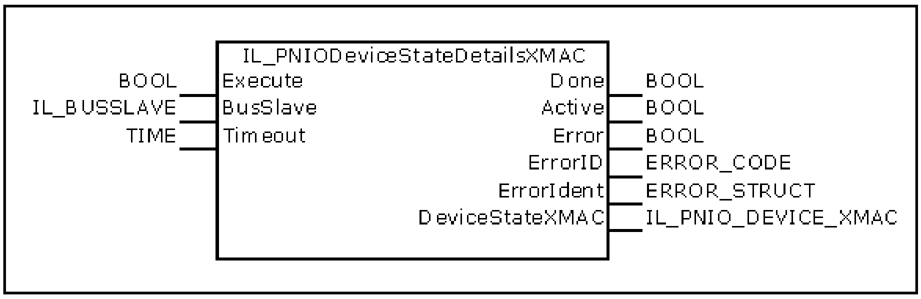


Fig.11-175: IL_PNIOGetDeviceXMACState

	Name	Type	Comment
VAR_INPUT	Execute	BOOL	Processing of the function block enabled (once, edge-controlled)
	BusSlave	IL_BUSSLAVE	Instance of the field bus slave in the order of the configuration tree.
	Timeout	TIME	Timeout monitoring of the function block in ms. T#0ms = Timeout monitoring not active
VAR_OUTPUT	Done	BOOL	Processing completed without error, output data valid
	Active	BOOL	Processing not yet completed, output data invalid
	Error	BOOL	Processing completed with error, output data invalid
	ErrorID	ERROR_CODE	Standardized rough classification of the error
	ErrorIdent	ERROR_STRUCT	Function-block-specific detailed information on the error
	DeviceStateXMAC	IL_PNIO_DEVICE_XMAC	Detailed state information about the two ports of the communication controller

Fig.11-176: Function block IL_PNIOGetDeviceXMACState

Min./max. and default values of the inputs

Name	Type	Min. value	Max. value	Default value	Effective
Execute	BOOL			FALSE	Continuous
BusSlave	IL_BUSSLAVE	IL_BUSSLAVE_0	IL_BUSSLAVE_4	IL_BUSSLAVE_0	Rising edge at Execute
Timeout	TIME	T#0ms	T#3600s	T#2s	Rising edge at "Execute"

Fig. 11-177: Minimum / maximum values and default values of the IL_PNIODeviceStateDetailsXMAC inputs

Functional Description

When activated, the function block once determines the current statistical information of the two Ethernet ports of the communication controller.

Error Handling

The function block uses error table **PROFINETIO_TABLE** (ERROR_TABLE = 16#0200). It can generate the following error messages in Additional1 and Additional2:

ErrorID	Additional1	Additional2	
SYSTEM_ERROR	16#0101	Internal use	Controller not in configuration table
COMMUNICATION_ERROR	16#0104	0	Timeout
STATE_MACHINE_ERROR	16#0105	0	Error in the state machine
COMMUNICATION_ERROR	16#0003	*PROFINET IO device diagnosis* ● Refer to General Error Codes, Profinet I/O Device on page 228. ● Refer to Special Error Codes, Profinet I/O Device on page 230.	

Fig. 11-178: Error codes of function block IL_PNIODeviceStateDetailsXMAC

11.6.5 IL_BUSSLAVE

Brief Description

The IL_BUSSLAVE enumeration type of the RIL_FieldbusTypes.library serves to select the appropriate field bus slave. In addition to the onboard real-time Ethernet interface, other devices can also be available in an MLC system via function modules.

The devices are distinguished on the basis of their ascending order in the configuration. Indexes are assigned to the devices in the following order:

- Onboard
- Function module 1
- Function module 2
- Function module 3
- Function module 4

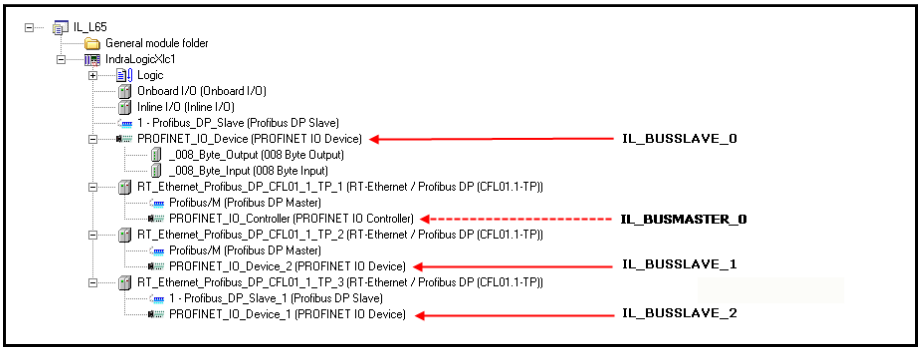
Element	Value	Description
IL_BUSSLAVE_0	0	First device
IL_BUSSLAVE_1	1	Second device
IL_BUSSLAVE_2	2	Third device
IL_BUSSLAVE_3	3	Fourth device
IL_BUSSLAVE_4	4	Fifth device

Fig. 11-179: Elements of the IL_BUSSLAVE enumeration type

Examples of addressing the various instances of the PROFINET IO devices on a control:

Configured:	Addressed with:
Onboard: PROFINET IO device	IL_BUSSLAVE_0
FM1: PROFINET IO Controller	IL_BUSMASTER_0
FM2: PROFINET IO device	IL_BUSSLAVE_1
FM3: PROFINET IO device	IL_BUSSLAVE_2

Fig.11-180: Enumeration type IL_BUSSLAVE, example



- IL_L65
  - General module folder
  - IndraLogicXlc1
    - Logic
    - Onboard I/O (Onboard I/O)
    - Inline I/O (Inline I/O)
    - 1 - Profibus_DP_Slave (Profibus DP Slave)
    - PROFINET_IO_Device (PROFINET IO Device) ← IL_BUSSLAVE_0
      - _008_Byte_Output (008 Byte Output)
      - _008_Byte_Input (008 Byte Input)
    - RT_Ethernet_Profibus_DP_CFL01_1_TP_1 (RT-Ethernet / Profibus DP (CFL01.1-TP))
      - Profibus/M (Profibus DP Master)
      - PROFINET_IO_Controller (PROFINET IO Controller) ← IL_BUSMASTER_0
    - RT_Ethernet_Profibus_DP_CFL01_1_TP_2 (RT-Ethernet / Profibus DP (CFL01.1-TP))
      - Profibus/M (Profibus DP Master)
      - PROFINET_IO_Device_2 (PROFINET IO Device) ← IL_BUSSLAVE_1
    - RT_Ethernet_Profibus_DP_CFL01_1_TP_3 (RT-Ethernet / Profibus DP (CFL01.1-TP))
      - 1 - Profibus_DP_Slave_1 (Profibus DP Slave)
      - PROFINET_IO_Device_1 (PROFINET IO Device) ← IL_BUSSLAVE_2

Fig.11-181: Enumeration type IL_BUSSLAVE, example

11.6.6 IL_PNIO_DEVICE_STATE

Brief Description

Data structure IL_PNIO_DEVICE_STATE contains the state information of the PNIO device stack.

Structure element	Type	Description
PnsState	UDINT	State of the PROFINET IO device task
LastRslt	UDINT	Previous error code. The meaning of the error can be learned from the error table.
LinkState	UDINT	Link state of the network connection
ConfigState	UDINT	Configuration status
CommunicationState	UDINT	Network status of the communication channel
CommunicationError	UDINT	Current error code of the communication channel. The meaning of the error can be learned from the error table.

Fig.11-182: Data structure IL_PNIO_DEVICE_STATE

The structure elements can have the following values:

PnsState

The 'PnsState' element of data structure IL_PNIO_DEVICE_STATE describes the status of the PNIO device via bit code. Multiple bits can be set at the same time.

Bit	Value	
7	16#00000080	Network Communication is enabled
6	16#00000040	Network Communication is allowed
5	16#00000020	Module 0 an Submodule 1 are plugged
4	16#00000010	Module 0 is plugged

Bit	Value	
3	16#00000008	At least one API (Application ProzessIndentifier) is present
2	16#00000004	Reserved
1	16#00000002	Profinet Stack is started
0	16#00000001	Device Information is set

LinkState

The 'LinkState' element of data structure IL_PNIO_DEVICE_STATE describes the status of the network connection.

Value	
0	No information available
1	Physical link works correctly
2	Low speed of physical link
3	No physical link present

ConfigState

Value	
0	Not configured
1	Configured with DBM Files
2	Error during configuration with DBM Files
3	Configured by application
4	Configuration by application is running
5	Error during configuration by application
6	Configured with warmstart-parameters
7	Configuration with warmstart-parameters is running
8	Error during configuration with warmstart-parameters

CommunicationState

The 'CommunicationState' element of data structure IL_PNIO_DEVICE_STATE describes the status of communication of the PNIO device.

Value	Designation	
0	UNKNOWN	Communication status unknown
1	OFFLINE	No PN communication
2	STOP	No PN communication possible or allowed
3	IDLE	Reserved
4	OPERATE	PN communication allowed (conclusions about cyclic communication cannot be drawn)

11.6.7 IL_PNIO_DEVICE_XMAC

Brief description

The structure contains the status and statistics information of the subordinate Profinet 2-port switch.

A structure of type IL_PNIO_DEVICE_XMAC_PORT, page 228, is filed for each available port.

Type	Designation	Description
ARRAY [0..1] of IL_PNIO_DEVICE_XMAC_PORT	XmacPort[]	State of the Profinet IO device task

Fig.11-183: Data structure IL_PNIO_DEVICE_XMAC

Assignment of pin names to the ports with indexes 0 and 1 in field aXmacDiag[]:

Control	Port 0	Port 1
CML65, CML45	X7E3	X7E4
CFL01_1_TP	X7E1	X7E2

11.6.8 IL_PNIO_DEVICE_XMAC_PORT

Brief description The structure contains the status and statistics information of a switch port.

Element	Type	
FramesTransmittedOk	UDINT	count of frames that are successfully transmitted
SingleCollisionFrames	UDINT	count of frames that are involved into a single collision
MultipleCollisionFrames	UDINT	count of frames that are involved into more that one collisions
LateCollisions	UDINT	later than 512 bit times into the transmitted packet
LinkDownDuringTransmission	UDINT	count of the times that a frame was transmitted during link down
UtxUnderflowDuringTransmission	UDINT	utx fifo underflow at transmission time
TxFatalErrors	UDINT	wrong tpu error code, shall always be zero
FramesReceivedOk	UDINT	count of frames that are successfully received
FrameCheckSequenceErrors	UDINT	count of frames that are an integral number of octets in length and do not pass the FCS check
AlignmentErrors	UDINT	count of frames that are not an integral number of octets in length and do not pass the FCS check
FrameTooLongErrors	UDINT	count of frames that are received and exceed the maximum permitted frame size
RuntFramesReceived	UDINT	count of frames that have a length between 42..63 bytes and a valid CRC
CollisionFragmentsReceived	UDINT	count of frames that are smaller that 64 bytes and have a invalid CRC
FramesDroppedDueLowResource	UDINT	no empty pointer available at indication time
FramesDroppedDueUrxOverflow	UDINT	urx fifo overflow at indication time
RxFatalErrors	UDINT	wrong rpu error code, shall always be zero

Fig.11-184: STRUCT IL_PNIO_DEVICE_XMAC_PORT

11.6.9 General Error Codes, Profinet I/O Device

The error codes presented are applicable to devices. These error codes are general error codes, i.e., they are independent of any bus.

The error codes are signaled as ERROR_STRUCT, Additional2 with ERROR_STRUCT, **Additional1= 16#0003.**

Additional2	
16#800A0001	Invalid pointer (NULL) passed to driver
16#800A0002	No board with the given name / index available
16#800A0003	No channel with the given index available
16#800A0004	Invalid handle passed to driver
16#800A0005	Invalid parameter
16#800A0006	Invalid command
16#800A0007	Invalid buffer size
16#800A0008	Invalid access size
16#800A0009	Function failed
16#800A000A	File could not be opened
16#800A000B	File size is zero
16#800A000C	Insufficient memory to load file
16#800A000D	File checksum compare failed
16#800A000E	Error reading from file
16#800A000F	Invalid file type
16#800A0010	Invalid file name
16#800A0011	Driver function not available
16#800A0012	Given buffer is too short
16#800A0013	Failed to map the memory
16#800A0014	No more entries available
16#800B0001	Driver not initialized
16#800B0002	Driver init state error
16#800B0003	Driver read state error
16#800B0004	Command is active on device
16#800B0005	General error during download
16#800B0006	Wrong driver version
16#800B0030	CIFx driver is not running
16#800B0031	Failed to initialize the device
16#800B0032	Channel not initialized (xOpenChannel not called)
16#800B0033	I/OControl call failed
16#800B0034	Driver was not opened
16#800C0010	Dual port memory not accessible (board not found)
16#800C0011	Device not ready (ready flag failed)
16#800C0012	Device not running (running flag failed)
16#800C0013	Watchdog test failed
16#800C0015	Error in handshake flags

Additional2	
16#800C0016	Send mailbox is full
16#800C0017	Send packet timeout
16#800C0018	Receive packet timeout
16#800C0019	No packet available
16#800C001A	Mailbox too short
16#800C0020	Reset command timeout
16#800C0021	COM-flag not set
16#800C0022	I/O data exchange failed
16#800C0023	I/O data exchange timeout
16#800C0024	Unknown I/O exchange mode
16#800C0025	Device function failed
16#800C0026	DPM size differs from configuration
16#800C0027	Unknown state mode
16#800C0028	Output port already in use
16#800C0013	Configuration locking timeout
16#800C002A	Configuration unlocking timeout
16#800C002B	Set HOST state timeout
16#800C002C	Clear HOST state timeout
16#800C002D	Timeout during channel initialization
16#800C002E	Set Bus ON Timeout
16#800C002F	Set Bus OFF Timeout

Fig. 11-185: General bus slave error codes

11.6.10 Special Error Codes, Profinet I/O Device

The error codes shown apply to PROFINET I/O devices.

The error codes are signaled as ERROR_STRUCT, Additional2 with ERROR_STRUCT, Additional1= 16#0003.

Additional2	
16#C0300001	Invalid Command
16#C0300004	PROFINET IO Device Setup failed

Fig. 11-186: Profinet IO Devices, special error codes

11.7 RIL_EtherNetIPAdapter.library

11.7.1 General

The controls can be operated as field bus slaves. This special field bus slave is called EtherNet/IP adapter in the context of EtherNet/IP.

The present library describes the interfaces of the EtherNet/IP adapter to the IEC user program. These interfaces allow diagnosing the EtherNet/IP participants.

The IL_EnableDoneBase and IL_ExecuteDoneBase methods are used and extended or overwritten.

Two interface types are supported:

- EtherNet/IP adapter via the Engineering interface and
- EtherNet/IP adapter via the onboard or function module interface.

Target systems

The library can be used with the following systems:

Target assembly	Remark
CML65	Engineering/Onboard/function modules
CML45	Engineering/Onboard/function modules
CML25	Engineering/Onboard/function modules
VEP	Onboard

Fig. 11-187: Target systems

The library includes the following function blocks for the EtherNet/IP adapter via the onboard or function module interface:

- IL_ENIPAdapterState, page 231, Diagnostics: basic adapter state.
- IL_ENIPAdapterStateDetails, page 233, Diagnostics: detailed status of the Ethernet/IP stack of the adapter.

The library includes the following function block for the EtherNet/IP adapter via the Engineering interface:

- IL_Status, page 234, Diagnostics of cyclic communication.

Structures an enumeration types

- IL_ENIP_ADAPTER_STATE, page 237, structure containing diagnostic information on the state of the communication stack, the state of the bus communication, the error counter as well as an overview of configured, active and diagnostics reporting slaves.

Selecting the slave

The IL_BUSSLAVE enumeration type of the RIL_FieldbusTypes.library serves to select the appropriate EtherNet/IP adapter.

See also: IL_BUSSLAVE, page 236.

11.7.2 IL_ENIPAdapterState

Brief Description

This function block returns a basic diagnosis of the EtherNet/IP adapter.

This simple diagnostics can be cyclically queried in a polling.

Assignment: Target system/library

Target system	Library
MLC 11VRS and higher	RIL_EtherNetIPAdapter.compiled-library
IndraLogic L65 11VRS and above	RIL_EtherNetIPAdapter.compiled-library
MTX 11VRS and above	

Fig. 11-188: Reference table of FB IL_ENIPAdapterState

Interface Description

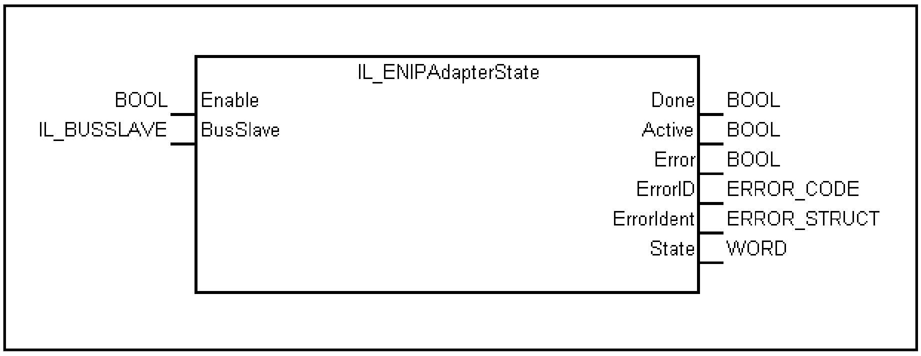


Fig.11-189: Interface of FB IL_ENIPAdapterState

I/O type	Name	Data type	Description
VAR_INPUT	Enable	BOOL	Enables processing of the function block (permanent, level-controlled)
	BusSlave	IL_BUSSLAVE	Instance of the adapter in the order of the configuration tree. If only one EtherNet/IP adapter is available, the value of this adapter is always IL_BUSSLAVE_0.
VAR_OUTPUT	Done	BOOL	Processing completed without error, output data valid
	Active	BOOL	Processing active
	Error	BOOL	Processing completed with error
	ErrorID	ERROR_CODE	Describes the diagnosis in the event of an error (from RIL_CommonTypes)
	ErrorIdent	ERROR_STRUCT	Detailed diagnosis (ErrorTable ETHERNETIP_TABLE)
	State	WORD	State of the EtherNet/IP adapter, see below

Fig.11-190: I/O interface of FB IL_ENIPAdapterState

The "State" output provides a rough diagnostic classification in binary code.

The bits are contained in the RIL_FieldbusTypes.library as global "Fieldbus-States" constants.

Output "State"

Bit no.	Description	Description
0	IL_FBS_BUS_OFF	No field bus communication
2	IL_FBS_CONFIG_ERROR	Faulty configuration by the field bus master
4	IL_FBS_SLAVE_ERROR	Device error

Min./max. and default values of the inputs

Name	Type	Min. value	Max. value	Default value	Effective
Enable	BOOL			FALSE	Continuous
BusSlave	IL_BUSSLAVE	IL_BUSSLAVE_0	IL_BUSSLAVE_4	IL_BUSSLAVE_0	Rising edge at Enable

Fig.11-191: Min./max. and default values of the IL_ENIPAdapterState inputs

Functional Description

The function block returns the state information of the EtherNet/IP adapter as bit string in "State".

The simple diagnostics is queried cyclically. If this function block signals a diagnostic message, function block IL_ ENIPAdapterStateDetails, page 233, should be used to retrieve detailed diagnostics.

Error Handling

The function block uses the **error table ETHERNET_IP_TABLE** (ERROR_TABLE = 16#0151). It can generate the following error messages in Additional1 and Additional2:

ErrorID	Additional1	Additional2	
SYSTEM_ERROR	16#0101	Internal use	Adapter not found in the configuration table

Fig. 11-192: FB IL_ENIPAdapterState error codes

11.7.3 IL_ENIPAdapterStateDetails

Brief Description

This function block determines the detailed status of the EtherNet/IP adapter stack. It provides information on

- state of the communication stack,
- the state of the bus communication,
- the previous error values.

Assignment: Target system/library

Target system	Library
MLC 11VRS and higher	RIL_EtherNetIPAdapter.compiled-library
IndraLogic L65 11VRS and above	RIL_EtherNetIPAdapter.compiled-library
MTX 11VRS and above	

Fig. 11-193: Reference table of FB IL_ENIPAdapterStateDetails

Interface Description

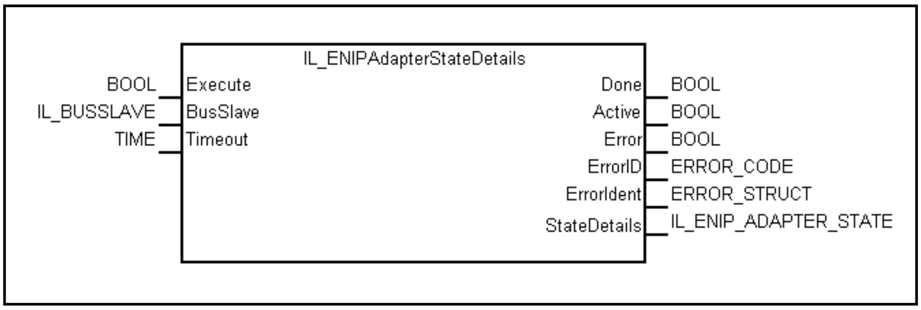


Fig. 11-194: Interface of FB IL_ENIPAdapterStateDetails

I/O type	Name	Data type	Description
VAR_INPUT	Execute	BOOL	Processing of the function block enabled (once, edge-controlled)
	BusSlave	IL_BUSSLAVE	Instance of the field bus slave in the order of the configuration tree.
	Timeout	TIME	Timeout monitoring of the function block in ms. T#0ms = Timeout monitoring not active
VAR_OUTPUT	Done	BOOL	Processing completed without error, output data valid
	Active	BOOL	Processing not yet completed, output data invalid

I/O type	Name	Data type	Description
	Error	BOOL	Processing completed with error, output data invalid
	ErrorID	ERROR_CODE	Standardized rough classification of the error
	ErrorIdent	ERROR_STRUCT	Function-block-specific detailed information on the error
	AdapterState	IL_ENIP_ADAPTER_STATE	Detailed information about the status of the adapter stack

Fig.11-195: I/O interface of FB IL_ENIPAdapterStateDetails

Min./max. and default values of the inputs

Name	Type	Min. value	Max. value	Default value	Effective
Execute	BOOL			FALSE	Continuous
BusSlave	IL_BUSSLAVE	IL_BUSSLAVE_0	IL_BUSSLAVE_4	IL_BUSSLAVE_0	Rising edge at Execute
Timeout	TIME	T#0ms	T#3600s	T#2s	Rising edge at "Execute"

Fig.11-196: Min./max. and default values of the IL_ENIPAdapterStateDetails inputs

Functional Description

When activated, this function block determines the current state of the adapter once.

Error Handling

The function block uses error table **ETHERNETIP_TABLE** (ERROR_TABLE = 16#0151).

It can generate the following error messages in Additional1 and Additional2:

ErrorID	Additional1	Additional2	
SYSTEM_ERROR	16#0101	Internal use	Scanner not in configuration table
COMMUNICATION_ERROR	16#0104	0	Timeout
STATE_MACHINE_ERROR	16#0105	0	Error in the state machine
COMMUNICATION_ERROR	16#0003	*EtherNet/IP adapter diagnostics* ● General Error Codes, EtherNet/IP Adapter , page 237,and ● Special Error Codes, EtherNet/IP Adapter , page 239.	

Fig.11-197: FB IL_ENIPAdapterStateDetails error codes

11.7.4 IL_Status

Brief Description

This function block returns a basic diagnosis of the EtherNet/IP adapter (Engineering port).

This basic diagnostics can be cyclically queried in a polling.

Assignment: Target system/library

Target system	Library
XLC	RIL_EtherNetIPAdapter.compiled-library
MLC	RIL_EtherNetIPAdapter.compiled-library
MTX	

Fig.11-198: Reference table of FB IL_ENIPAdapterStateDetails

Interface Description

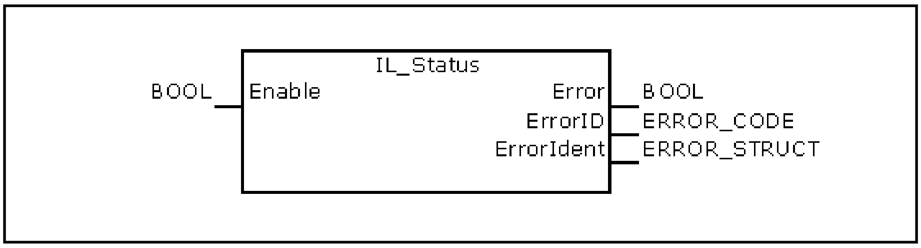


Fig. 11-199: IL_Status

	Name	Type	Comment
VAR_INPUT	Enable	BOOL	Enables processing of the function block (permanent, level-controlled)
VAR_OUTPUT	Error	BOOL	Processing completed with error
	ErrorID	ERROR_CODE	Describes the diagnosis in the event of an error (from RIL_CommonTypes)
	ErrorIdent	ERROR_STRUCT	Detailed diagnosis (ErrorTable ETHERNETIP_TABLE)

Fig. 11-200: Interface of IL_Status

Min./max. and default values of the inputs

Name	Type	Min. value	Max. value	Default value	Effective
Enable	BOOL			FALSE	Continuous

Fig. 11-201: Min./max. and default values of the IL_Status inputs

Functional Description

When Enable is TRUE, the state of the cyclic EtherNet/IP communication is queried whenever the function block is called.

If an error is detected in cyclic communication, the state of output "Error" becomes TRUE. The error message in Additional1 and Additional2 indicates the cause of the communication error.

Error Handling

The function block uses error table **ETHERNETIP_TABLE** (ERROR_TABLE = 16#0151).

It can generate the following error messages in Additional1 and Additional2:

ErrorID	Additional1	Additional2	
COMMUNICATION_ERROR	16#10000001	16#00000000	NoCyclicCom: The scanner has not yet established cyclic communication with the IndraLogic-EtherNet/IP adapter.
COMMUNICATION_ERROR	16#10000002	16#00000000	Idle: The EtherNet/IP scanner has started cyclic communication but has set the Idle flag in the cyclic channel. In this way, the scanner signals that its output image is invalid.

ErrorID	Additional1	Additional2	
COMMUNICATION_ERROR	16#10000003	16#00000000	Timeout: The cyclic communication time monitoring function signals an error.
COMMUNICATION_ERROR	16#10000004	16#00000000	Closed: Cyclic communication was actively completed by the EtherNet/IP scanner.

Fig. 11-202: FB IL_ENIPAdapterStateDetails error codes

Example The following example shows the function block IL_Status is to be used.

Example

```
(*************************************************
** Variables for IL_Status
**************************************************)
PROGRAM PLC_PRG
VAR
  fbDiag:         IL_Status;
 diCtrValidInput: DINT;
  ...
END_VAR;
  ...
(*************************************************
** Program
**************************************************)
fbDiag(Enable:=TRUE);
IF(Diag.Error = TRUE) THEN
  (*Insert error treatment here*)
...
  fbDiag(Enable:=FALSE ); (*Reset error (Enable*)
ELSE
  (*Valid data: Insert input and output data processing here*)
  diCtrValidInput := diCtrValidInput +1;
END_IF
...
```

11.7.5 IL_BUSSLAVE

Brief Description The IL_BUSSLAVE enumeration type of the RIL_FieldbusTypes.library serves to select the appropriate field bus slave. In addition to the onboard real-time Ethernet interface, other EtherNet/IP adapters can also be available in a control via function modules.

The devices are distinguished on the basis of their ascending order in the configuration. Indexes are assigned to the devices in the following order:

- Onboard
- Function module 1
- Function module 2
- Function module 3
- Function module 4

Element	Value	Description
IL_BUSSLAVE_0	0	First device
IL_BUSSLAVE_1	1	Second device
IL_BUSSLAVE_2	2	Third device
IL_BUSSLAVE_3	3	Fourth device
IL_BUSSLAVE_4	4	Fifth device

Fig. 11-203: Elements of the IL_BUSSLAVE enumeration type

11.7.6 IL_ENIP_ADAPTER_STATE

Brief Description The IL_ENIP_ADAPTER_STATE data structure contains the status information of the EtherNet/IP adapter stack.

Structure element	Type	Description
LastGRC	UDINT	Latest CIP General Error Code page 239
LastERC	UDINT	Latest Extended Error Code page 241
CurrentConnection	UDINT	Established connections
CommunicationState	UDINT	Network state of communication channel, see below
CommunicationError	UDINT	Current error code of the communication channel. The meaning of the error can be learned from the error table, see below.

Fig. 11-204: IL_ENIP_ADAPTER_STATE data structure

Element "CommunicationState" of data structure IL_ENIP_ADAPTER_STATE describes the state of communication of the EtherNet/IP adapter.

Structure element "CommunicationState"

Value	Description	Description
16#00000000	UNKNOWN	Communication state unknown
16#00000001	NOT_CONFIGURED	Communication channel was not configured
16#00000002	STOP	Stop
16#00000003	IDLE	Idle
16#00000004	OPERATE	EtherNet/IP communication

Element "CommunicationError" of data structure IL_ENIP_ADAPTER_STATE describes the current error code of the communication channel.

Structure element "CommunicationError"

Value	Description	Description
16#00000000	SUCCESS	Communication channel running
16#C0000140	NETWORK FAULT	Network fault
16#C0000141	CONNECTION CLOSED	Communication was closed
16#C0000142	CONNECTION TIMED OUT	The communication time monitoring function signals an error.
16#C0000143	LONELY NETWORK	Lonely network
16#C0000144	DUPLICATE NODE	Duplicate nodes
16#C0000145	CABLE DISCONNECT	Cable was pulled off

11.7.7 General Error Codes, EtherNet/IP Adapter

The error codes presented are applicable to ENIP adapters. These error codes are general error codes, i.e., they are independent of any bus.

The error codes are signaled as ERROR_STRUCT, Additional2 with ERROR_STRUCT, **Additional1= 16#0003**.

Additional2	
16#800A0001	Invalid pointer (NULL) passed to driver
16#800A0002	No board with the given name / index available
16#800A0003	No channel with the given index available
16#800A0004	Invalid handle passed to driver
16#800A0005	Invalid parameter
16#800A0006	Invalid command
16#800A0007	Invalid buffer size
16#800A0008	Invalid access size
16#800A0009	Function failed
16#800A000A	File could not be opened
16#800A000B	File size is zero
16#800A000C	Insufficient memory to load file
16#800A000D	File checksum compare failed
16#800A000E	Error reading from file
16#800A000F	Invalid file type
16#800A0010	Invalid file name
16#800A0011	Driver function not available
16#800A0012	Given buffer is too short
16#800A0013	Failed to map the memory
16#800A0014	No more entries available
16#800B0001	Driver not initialized
16#800B0002	Driver init state error
16#800B0003	Driver read state error
16#800B0004	Command is active on device
16#800B0005	General error during download
16#800B0006	Wrong driver version
16#800B0030	CIFx driver is not running
16#800B0031	Failed to initialize the device
16#800B0032	Channel not initialized (xOpenChannel not called)
16#800B0033	IOControl call failed
16#800B0034	Driver was not opened
16#800C0010	Dual port memory not accessable (board not found)
16#800C0011	Device not ready (ready flag failed)
16#800C0012	Device not running (running flag failed)
16#800C0013	Watchdog test failed
16#800C0015	Error in handshake flags

Additional2	
16#800C0016	Send mailbox is full
16#800C0017	Send packet timeout
16#800C0018	Receive packet timeout
16#800C0019	No packet available
16#800C001A	Mailbox too short
16#800C0020	Reset command timeout
16#800C0021	COM-flag not set
16#800C0022	I/O data exchange failed
16#800C0023	I/O data exchange timeout
16#800C0024	Unknown I/O exchange mode
16#800C0025	Device function failed
16#800C0026	DPM size differs from configuration
16#800C0027	Unknown state mode
16#800C0028	Output port already in use
16#800C0029	Configuration locking timeout
16#800C002A	Configuration unlocking timeout
16#800C002B	Set HOST state timeout
16#800C002C	Clear HOST state timeout
16#800C002D	Timeout during channel initialization
16#800C002E	Set Bus ON Timeout
16#800C002F	Set Bus OFF Timeout

Fig. 11-205: General error codes of BusMaster

11.7.8 Special Error Codes, EtherNet/IP Adapter

This section presents the special error codes of the EtherNet/IP adapter.

The error codes are signaled as ERROR_STRUCT, Additional2 with ERROR_STRUCT, **Additional1= 16#0003**.

Additional2	
16#C0590001	Invalid command
16#C0590004	Configuration of TCP/IP failed
16#C0590009	Invalid offset for I/O data

Fig. 11-206: EtherNet/IP adapter, special error codes

11.7.9 CIP, General Error Codes

This section introduces the CIP general error codes.

Code	
16#01	A connection-related service failed along the connection path
16#02	Resources needed for the object to perform the requested service were unavailable

Code	
16#03	See status code 0x20 which is the preferred value to use for this condition.
16#04	A path segment error has been encountered. Evaluation of the supplied path information failed.
16#05	The path references an unknown object class, instance or structure element causing the abort of path processing.
16#06	Only a part of the expected data could be transferred.
16#07	The messaging connection was lost.
16#08	The requested service has not been implemented or has not been defined for this object class or instance.
16#09	Detection of invalid attribute data
16#0A	An attribute in the Get_Attribute_List or Set_Attribute_List response has a status not equal to 0.
16#0B	The object is already in the mode or state which has been requested by the service
16#0C	The object is not able to perform the requested service in the current mode or state
16#0D	It has been tried to create an instance of an object which already exists.
16#0E	It has been tried to change an non-modifiable attribute.
16#0F	A check of permissions or privileges failed.
16#10	The current mode or state of the device prevents the execution of the requested service.
16#11	The data to be transmitted in the response buffer requires more space than the size of the allocated response buffer.
16#12	The service specified an operation that is going to fragment a primitive data value, i.e., half a REAL data type.
16#13	The service did not supply all required data to perform the specified operation.
16#14	An unsupported attribute has been specified in the request.
16#15	More data than was expected was supplied by the service.
16#16	The specified object does not exist in the device.
16#17	Fragmentation sequence for this service is not currently active for this data.
16#18	The attribute data of this object has not been saved prior to the requested service.
16#19	The attribute data of this object could not be saved due to a failure during the storage attempt.
16#1A	The service request packet was too large for transmission on a network in the path to the destination. The routing device was forced to abort the service.
16#1B	The service response packet was too large for transmission on a network in the path from the destination. The routing device was forced to abort the service.
16#1C	The service did not supply an attribute in a list of attributes that was needed by the service to perform the requested behavior.
16#1D	The service returns the list of attributes containing status information for invalid attributes.
16#1E	An embedded service caused an error.
16#1F	A vendor specific error has occurred. This error should only occur when none of the other general error codes can be applied correctly.
16#20	A parameter which was associated with the request was invalid. The parameter does not meet the requirements of the CIP specification and/or the requirements defined in the specification of an application object.
16#21	An attempt was made to write to a write-once medium for the second time, or to modify a value that cannot be changed after being established once.

Code	
16#22	An invalid reply is received. Possible causes can for instance be among others a reply service code not matching the request service code or a reply message shorter than the expectable minimum size.
16#25	The key segment (i.e., the first segment in the path) does not match the destination module. More information about which part of the key check failed can be derived from the object specific status.
16#26	Path cannot be routed to an object due to lacking information or too much routing data has been included.
16#27	It has been attempted to set an attribute which may not be set in the current situation.
16#28	The Member ID specified in the request is not available within the specified class/ instance or attribute.
16#29	A request to modify a member which cannot be modified has occurred.

Fig. 11-207: CIP general error code

11.7.10 Advanced Error Codes for Connection Manager

This section introduces the extended error codes for the Connection Manager.

Code	
16#0100	Connection already in use
16#0103	Transport type not supported
16#0106	More than one guy configuring
16#0107	Trying to close inactive connection
16#0108	Unsupported connection type
16#0109	Connection size mismatch
16#0110	Connection unconfigured
16#0111	Unsupportable RPI
16#0113	Connection Manager out of connections
16#0114	Vendor ID or Product Code mismatch
16#0115	Product Type mismatch
16#0116	Revision mismatch
16#0117	Nonexistent instance number
16#0118	Bad config instance number
16#0119	No controlling connection opened
16#011A	Application out of connections
16#0203	Using a timed out connection
16#0204	Unconnected Send timed out
16#0205	Unconnected Send parameter error
16#0301	No buffer memory available
16#0302	Insufficient bandwidth left
16#0303	Out of gen screeners
16#0304	Not configured to send RT data
16#0305	Signature does not match signature store in CCM

Code	
16#0306	CCM is not responding to request
16#0311	Nonexistent port
16#0312	Invalid link address in path
16#0315	Invalid segment in path
16#0316	Path & conn not equal in close
16#0317	Net segment not present or bad
16#0318	Link address to self invalid
16#0319	Resources in secondary unavailable
16#031D	Redundant connection mismatch
16#0320	Vendor specific: read write access fail
16#2105	Vendor specific: Access beyond end of the requested tag
16#2107	Vendor specific: Data type used in request does not match target tag's data type

Fig.11-208: Advanced error codes for Connection Manager

11.8 RIL_MappingList.library

11.8.1 General

To allow acyclic access to fieldbus slaves (DP slave, PNIO device and ENIP adapter), a mapping table is implemented on the fieldbus slaves.

A mapping to data objects of the control is stored in this mapping table to allow bus-specific acyclic fieldbus access.

This mapping (= addressing) rule is executed when an acyclic fieldbus access is made.

It is planned to create the mapping table on the user interface via a configuration tool and to load it to the control.

As long as this configuration tool is not available, the mapping table can be filled with the RIL_MappingList.library from the IEC program and edited.

PLC addresses used to access the user data and assigned to fieldbus-specific (object) accesses are filed in the mapping table

These addresses may be

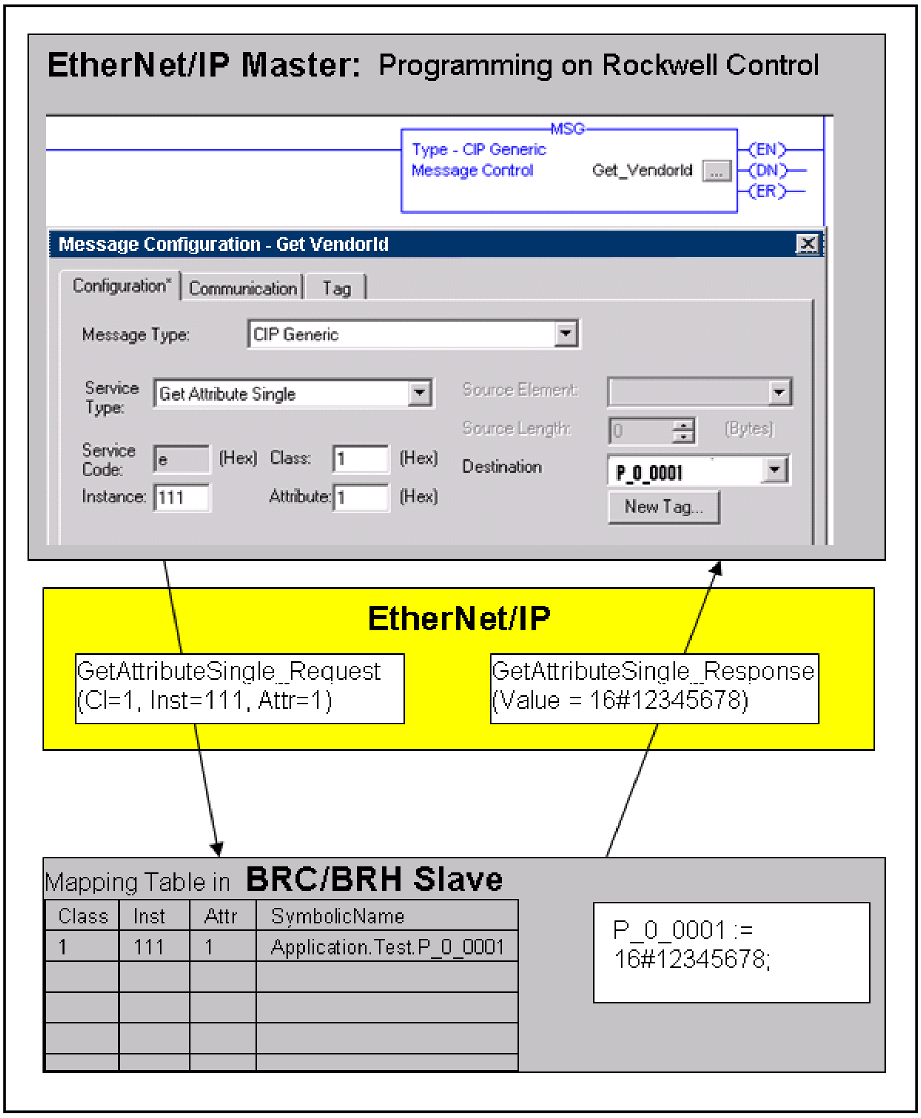
- symbolic PLC operands, or
- absolute memory addresses.

The mapping table is filled with the contents of the IO configuration (final configuration, depending on the mapping configuration tool of the user interface) or from the IEC program via the function block calls described herein.

The mapping table exists once for each fieldbus slave.

The fieldbus addresses can be used as desired within the value ranges allowed by the particular fieldbus.

The number of possible entries in the mapping table is presently limited to 256.


EtherNet/IP Master: Programming on Rockwell Control
MSG
Type - CIP Generic
Message Control
Get_VendorId
EN
DN
ER
Message Configuration - Get VendorId
Configuration*
Communication
Tag
New Tag...
EtherNet/IP
Mapping Table in BRC/BRH Slave

Message Type: CIP Generic

Service Type: Get Attribute Single

Service Code: e (Hex) Class: 1 (Hex)

Instance: 111 Attribute: 1 (Hex)

Source Element:

Source Length: 0 (Bytes)

Destination P_0_0001

GetAttributeSingle_Request
(Cl=1, Inst=111, Attr=1)

GetAttributeSingle_Response
(Value = 16#12345678)

Class	Inst	Attr	SymbolicName
1	111	1	Application.Test.P_0_0001

```
P_0_0001 :=
16#12345678;
```

Fig. 11-209: Example access via Ethernet/IP

Target systems

The library can be used with the following systems:

Target assembly	Remark
CML65	Onboard / function modules
CML45	Onboard / function modules
CML25	Onboard / function modules
VEP	Onboard
…	

Fig. 11-210: Target systems

The library contains the following components:

- FB IL_SlaveMapListInit page 244, initializing and deleting the mapping table.
- FB IIL_SlaveMapListAddEntry page 245, adding an entry to the mapping table.

- IL_FIELDBUSTYPE page 249, serves to select the type of the particular fieldbus.

  IL_FIELDBUSOBJECT page 249, contains the addressing of the various fieldbus objects.
- IL_ADDRESSTYPE page 250, serves to select the type of the variable to be mapped.

Selecting the slave

The IL_BUSSLAVE enumeration type of the RIL_FieldbusTypes.library serves to select the appropriate fieldbus slave.

See also: IL_BUSSLAVE page 248.

11.8.2 IL_SlaveMapListInit

Brief description

This function block initializes and deletes the entire mapping table.

This ensures that there are no history entries.

If the RIL_MappingList library is used, the list should be initialized before access entries are added, to avoid interactions with other methods for creating access entries.

This function block can likewise be used to disable outside access via the fieldbus again.

Assignment: target system / library

Target system	Library
MLC 11VRS and above	RIL_MappingList.compiled-library
MTX 11VRS and above	RIL_MappingList.compiled-library
IndraLogic L65 11VRS and above	RIL_MappingList.compiled-library

Fig. 11-211: FB IL_SlaveMapListInit reference table

Interface description

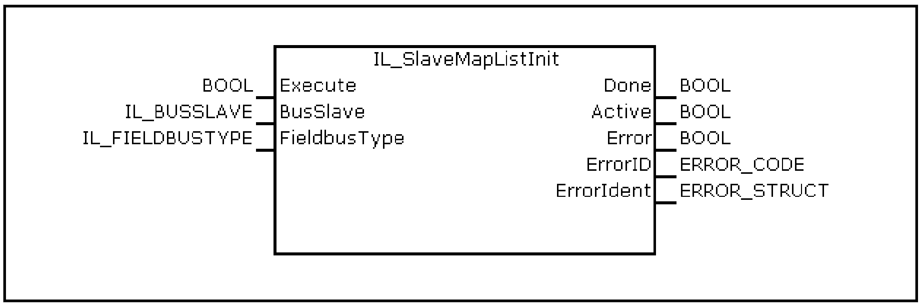


Fig. 11-212: FB IL_SlaveMapListInit interface

I/O type	Name	Data type	Description
VAR_INPUT	Execute	BOOL	Enables processing of the function block (once, edge-triggered)
	BusSlave	IL_BUSSLAVE	Instance of the fieldbus slave in the order of the configuration tree. If only one slave/device/adapter is available, the value of this slave/device/adapter is always IL_BUSSLAVE_0.
	FieldbusType	IL_FIELDBUSTYPE	Selects the fieldbus type (Profinet IO, Profibus DP, EtherNet/IP)
VAR_OUTPUT	Done	BOOL	Processing completed without error, output data valid
	Active	BOOL	Processing active

I/O type	Name	Data type	Description
	Error	BOOL	Processing completed with error
	ErrorID	ERROR_CODE	Describes the diagnosis in the event of an error (from RIL_CommonTypes)
	ErrorIdent	ERROR_STRUCT	Detailed diagnosis (ErrorTable NO_TABLE_USED)

Fig.11-213: IL_SlaveMapListInit I/O interface

Minimum, maximum and default values of the inputs

Name	Type	Min. value	Max. value	Default value	Effective
Execute	BOOL			FALSE	Continuously
BusSlave	IL_BUSSLAVE	IL_BUSSLAVE_0	IL_BUSSLAVE_4	IL_BUSSLAVE_0	Rising edge at Execute
FieldbusType	IL_FIELDBUS-TYPE	IL_DP_SLAVE	IL_ENIP_ADAPT-ER	IL_DP_SLAVE	Rising edge at Execute

Fig.11-214: Minimum, maximum and default values of the IL_SlaveMapListInit inputs

Functional description

Function block IL_SlaveMapListInit serves to delete an existing mapping list.

A mapping list may have been created through the following actions:

- The mapping table was generated via the user interface.
- The mapping table was generated earlier by the PLC program.

The FB must be called in the following cases:

- if the mapping table is not generated via the user interface,
- if it is intended to remove entries from the mapping table (i.e., complete deletion of the table).

Error handling

The function block uses error table **NO_TABLE_USED 16#0000**. It can generate the following error messages in Additional1 and Additional2:

ErrorID	Additional1	Additional2	
SYSTEM_ERROR	16#0101	16#0000 16#0001	BusSlave not found in configuration table
STATE_MACHINE_ERROR	16#0105	0	State machine error
INPUT_RANGE_ERROR	16#0108	0	Fieldbus type not supported

Fig.11-215: Fehlercodes FB IL_SlaveMapListInit

11.8.3 IL_SlaveMapListAddEntry

Brief description

This function block adds an entry to the mapping table. The entry is permanently associated with the fieldbus object address. A fieldbus object address can be assigned only once.

Assignment: target system / library

Target system	Library
MLC 11VRS and above	RIL_MappingList.compiled-library
MTX 11VRS and above	RIL_MappingList.compiled-library
IndraLogic L65 11VRS and above	RIL_MappingList.compiled-library

Fig.11-216: FB IL_SlaveMapListAddEntry reference table

Interface description

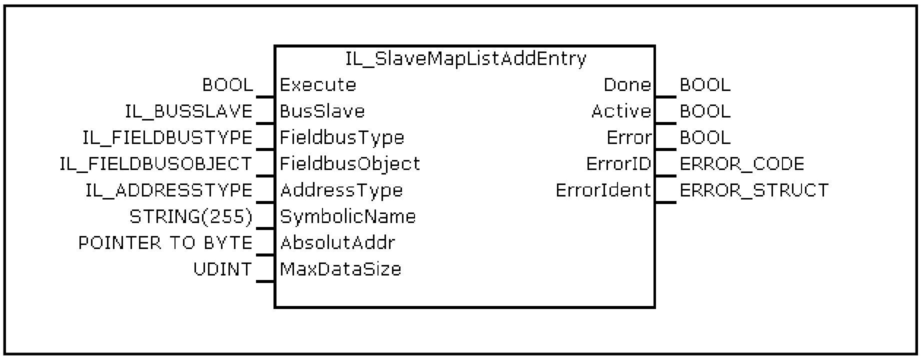


Fig.11-217: FB IL_SlaveMapListAddEntry interface

I/O type	Name	Data type	Description
VAR_INPUT	Execute	BOOL	Enables processing of the function block (once, edge-triggered)
	BusSlave	IL_BUSSLAVE	Instance of the fieldbus slave in the order of the configuration tree. If only one slave/device/adapter is available, the value of this slave/device/adapter is always IL_BUSSLAVE_0.
	FieldbusType	IL_FIELDBUSTYPE	Selects the fieldbus type (Profinet IO, Profibus DP, EtherNet/IP)
	FieldbusObject	IL_FIELDBUSOBJECT	Fieldbus-specific object address
	AddressType	IL_ADDRESSTYPE	Addressing type.
	SymbolicName	STRING(255)	Symbolic name of a variable from the IEC program. The symbol must comply with pertinent IEC programming rules. To ensure uniqueness, task and POU must perhaps also be specified.
	AbsoluteAddr	POINTER TO BYTE	Pointer to an absolutely addressed data area
	MaxDataSize	UDINT	Maximum allowed byte length for access to the object. This parameter can be used to limit the read/write access to the number of bytes specified. If the input address type = FBT_SYMBOL_IEC (_RO), this limit may also be below the size of the object addressed via SymbolicName.
VAR_OUTPUT	Done	BOOL	Processing completed without error, output data valid
	Active	BOOL	Processing not completed yet, output data is invalid
	Error	BOOL	Processing completed with error
	ErrorID	ERROR_CODE	Describes the diagnosis in the event of an error (from RIL_CommonTypes)
	ErrorIdent	ERROR_STRUCT	Detailed diagnosis (ErrorTable NO_TABLE_USED)

Fig.11-218: IL_SlaveMapListAddEntry I/O interface

Minimum, maximum and default values of the inputs

Name	Type	Min. value	Max. value	Default value	Effective
Execute	BOOL			FALSE	Continuously
BusSlave	IL_BUSSLAVE	IL_BUSSLAVE_0	IL_BUSSLAVE_4	IL_BUSSLAVE_0	Rising edge at Execute
FieldbusType	IL_FIELDBUSTYPE	IL_DP_SLAVE	IL_ENIP_ADAPTER	IL_DP_SLAVE	Rising edge at Execute
FieldbusObject	IL_FIELDBUSOBJECT	Depending on the fieldbus	Depending on the fieldbus		Rising edge at Execute
AddressType	IL_ADRESSTYP	FBT_SYMBOL_IEC	FBT_ADR_IEC_RO	FBT_SYMBOL_IEC	Rising edge at Execute
SymbolicName	STRING(255)	n.a.	n.a.	”	Rising edge at Execute
AbsoluteAddr	POINTER TO BYTE	n.a.	n.a.	0	Rising edge at Execute
MaxDataSize	UDINT	0	Depending on the fieldbus	0	Rising edge at Execute

Fig.11-219: Minimum, maximum and default values of the IL_SlaveMapListAddEntry

Functional description

Example: Entry of the symbolic read access to IEC variable 'wrvar' via fieldbus index 16#1001 to slot and subslot 1 for a PN device. Write access is disabled. Symbol 'wrvar' must be entered in the symbol table.

Extract from the program

```
fbMapAdd: IL_SlaveMapListAddEntry;
wrvar:    DINT;

fbMapAdd.FieldbusType:=           IL_FIELDBUSTYPE.IL_PNIO_DEVICE;
fbMapAdd.BusSlave:=               IL_BUSSLAVE_0;
fbMapAdd.FieldbusObject.Index:=   16#1001;
fbMapAdd.FieldbusObject.Slot:=    1;
fbMapAdd.FieldbusObject.SubSlot:= 1;
fbMapAdd.MaxDataSize:=            SIZEOF(wrvar);
fbMapAdd.AddressType:=  IL_ADDRESSTYPE.IL_SYMBOL_IEC_RO; //read only
fbMapAdd.SymbolicName:=           'Application.MappingTest.wrvar';

fbMapAdd  (Execute:= TRUE);
```

Example: Entry of the absolutely addressed read and write access to IEC variable 'wrvar' via fieldbus index 16#1002 to slot and subslot 1 for a PN device.

Extract from the program

```
fbMapAdd: IL_SlaveMapListAddEntry;
wrvar:    DINT;

fbMapAdd.Execute:=                TRUE;
fbMapAdd.FieldbusType:=           IL_FIELDBUSTYPE.IL_PNIO_DEVICE;
fbMapAdd.FieldbusObject.Index:=   16#1002;
fbMapAdd.FieldbusObject.Slot:=    1;
fbMapAdd.FieldbusObject.SubSlot:= 1;
fbMapAdd.MaxDataSize:=            SIZEOF(wrvar);
fbMapAdd.AddressType:=            IL_ADDRESSTYPE.IL_ADDR_IEC;
fbMapAdd.AbsolutAddr:=            ADR(wrvar);

fbMapAdd  (Execute:= TRUE);
```

Error handling

The function block uses error table NO_TABLE_USED 16#0000. It can generate the following error messages in Additional1 and Additional2:

ErrorID	Additional1	Additional2	
SYSTEM_ERROR	16#0101	16#0000 16#0001	BusSlave not found in configuration table
STATE_MACHINE_ERROR	16#0105	0	statemachine error
INPUT_RANGE_ERROR	16#0108	0	Fieldbustype not supported
INPUT_RANGE_ERROR	16#0109	0	Addresstype not supported
SYSTEM_ERROR	16#010A	0	Mappingtable is filled up No more entry available
SYSTEM_ERROR	16#010B	0	Entry is already used
INPUT_RANGE_ERROR	16#010C	0	Symbol in SymbolicName not found
INPUT_RANGE_ERROR	16#010D	0	FieldbusObject not valid
INPUT_RANGE_ERROR	16#010E	0	FieldbusObject is reserved
INPUT_RANGE_ERROR	16#010F	0	AbsoluteAddr not valid
INPUT_RANGE_ERROR	16#0110	0	MaxDataSize is zero
INPUT_RANGE_ERROR	16#0111	0	MaxDataSize is too large
INPUT_RANGE_ERROR	16#0112	0	Index already reserved
INPUT_RANGE_ERROR	16#0113	0	Slot already reserved
INPUT_RANGE_ERROR	16#0114	0	Subslot already reserved
INPUT_RANGE_ERROR	16#0115	0	Class already reserved
INPUT_RANGE_ERROR	16#0116	0	Instanz already reserved
INPUT_RANGE_ERROR	16#0117	0	Attribute already reserved

Fig.11-220: Error codes FB IL_SlaveMapListAddEntry

11.8.4 IL_BUSSLAVE

Brief description

The IL_BUSSLAVE enumeration type of the RIL_FieldbusTypes.library serves to select the appropriate fieldbus slave.

In addition to the onboard real-time Ethernet interface, other controllers can also be available in a control via function modules.

The devices are distinguished on the basis of their ascending order in the configuration. Indexes are assigned to the devices in the following order:

- Onboard
- Function module 1
- Function module 2
- Function module 3
- Function module 4

If, for example, the onboard real-time Ethernet interface is a PROFINET IO controller and a PROFINET IO device is configured on FM1, the value assigned to the device on the FM1 is IL_BUSSLAVE_0.

Element	Value	Description
IL_BUSSLAVE_0	0	First device
IL_BUSSLAVE_1	1	Second device

Element	Value	Description
IL_BUSSLAVE_2	2	Third device
IL_BUSSLAVE_3	3	Fourth device
IL_BUSSLAVE_4	4	Fifth device

Fig. 11-221: Elements of the IL_BUSSLAVE enumeration type

11.8.5 IL_FIELDBUSTYPE

Brief description

Enumeration type IL_FIELDBUSTYPE serves to select the type of the appropriate fieldbus.

At present, fieldbus slaves are available for

- Profibus DP,
- Profinet IO,
- Ethernet/IP.

Element	Value	Description
IL_DP_SLAVE	0	Profibus DP
IL_PNIO_DEVICE	1	Profinet IO
IL_ENIP_ADAPTER	2	Ethernet/IP

Fig. 11-222: Elements of enumeration type IL_FIELDBUSTYPE

11.8.6 IL_FIELDBUSOBJECT

Brief description

Data structure IL_FIELDBUSOBJECT contains the addressing of the various fieldbus objects.

	Type	Required for ProfibusDP IL_DP_SLAVE	Required for ProfinetIO IL_PNIO_DEVICE	Required for EthernetIP IL_ENIP_ADAPTER	Description
Slot	WORD	Yes	Yes		Slot number
Index	WORD	Yes	Yes		Index
SubSlot	WORD		Yes		Subslot number
Class	WORD			Yes	Class
Instance	WORD			Yes	Instance
Attribute	WORD			Yes	Attribute

Fig. 11-223: Interface description of data structure IL_FIELDBUSOBJECT

Addressing of the fieldbus objects for ProfibusDP:

	Type	Min. value	Max. value	Default value	Description
Slot	WORD	0	254		Slot number
Index	WORD	0	254		Index

Fig. 11-224: IL_FIELDBUSOBJECT for ProfibusDP

Addressing of the fieldbus objects for ProfinetIO:

	Type	Min. value	Max. value	Default value	Description
Slot	WORD	0	16#7FFF		Slot number
Index	WORD	0	16#7FFF		Index (16#8000 to 16#ffff are reserved)
SubSlot	WORD	1	16#8FFF		Subslot number

Fig. 11-225: IL_FIELDBUSOBJECT for ProfinetIO

Addressing of the fieldbus objects for EtherNet/IP:

	Type	Min. value	Max. value	Default value	Description
Class	WORD	0	16#FFFF		
Instance	WORD	1	16#FFFF		
Attribute	WORD	1	16#FFFF		

Fig. 11-226: IL_FIELDBUSOBJECT for EtherNet/IP

11.8.7 IL_ADDRESSTYPE

Brief description Enumeration type IL_ADDRESSTYPE serves to select the type of the variables to be mapped.

Element	Value	Description
IL_SYMBOL_IEC	0	Symbolic IEC variable name; parameter SymbolicName is evaluated. Read and write access allowed
IL_ADDR_IEC	1	Absolute IEC address; parameter AbsoluteAddr is evaluated. Read and write access allowed
IL_SYMBOL_IEC_RO	2	Symbolic IEC variable name; parameter SymbolicName is evaluated. Read access allowed only
IL_ADDR_IEC_RO	3	Absolute IEC address; parameter AbsoluteAddr is evaluated. Read access allowed only

Fig. 11-227: Elements of enumeration type IL_ADDRESSTYPE

11.9 RIL_SERCOSIII.library

11.9.1 General

The RIL_SercosIII library is the interface between the PLC programming environment and the sercos III devices. For example, this interface allows diagnosing the sercos III IO devices or acyclically exchanging sercos parameters.

Target systems The library can be used with the following systems:

Target assembly	Remark
CML65	Onboard
CML45	Onboard

Target assembly	Remark
CML25	Onboard
VEP	Onboard

Fig. 11-228: Target systems

The library contains the following components:

Function blocks for implementing the following acyclic services (AcyclicCommunication):

- FB IL_SIIISvcRead page 251, reading a parameter via the sercos III service channel.
- FB IL_SIIISvcWrite page 256, writing a parameter via the sercos III service channel.

Diagnostics

- In preparation

Utilities

- FUN IL_SIIIElementsToIdn page 259, combining the individual elements of an IDN to form a MB_IDN value.

  Using data types:

  - IL_SIII_ELEMENT page 260, describes the sercos elements of a parameter,
  - IL_SIII_PARAM_TYPE page 260, distinguishes between standard IDN and product-specific IDN.

Selecting the master

The IL_BUSMASTER enumeration type of the RIL_FieldbusTypes.library serves to select the appropriate sercos III master.

At present, only the onboard master is supported.

See also: IL_BUSMASTER page 259.

11.9.2 IL_SIIISvcRead

Brief Description

Function block IL_SIIISvcRead can be used to read parameters of a slave device via the sercos III service channel.

Assignment: target system / library

Target system	Library
MLC	RIL_SercosIII.compiled-library
MTX	RIL_SercosIII.compiled-library
XLC	RIL_SercosIII.compiled-library
MLD	RIL_SercosIII.lib (IndraLogic 1.x)

Fig. 11-229: Reference table of the IL_SIIISvcRead function block

Interface Description

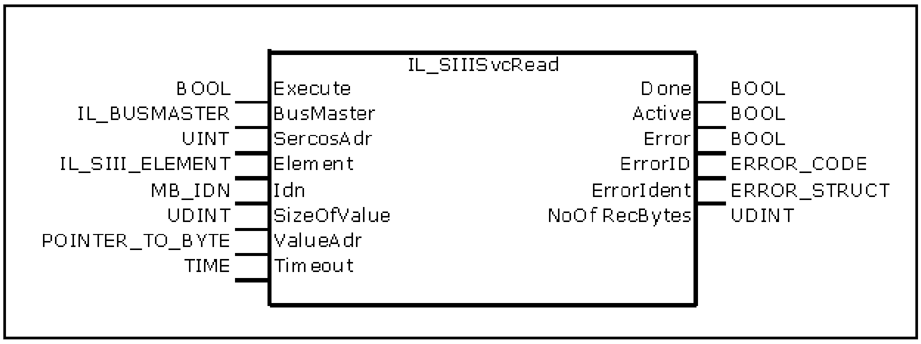


Fig.11-230: IL_SIIISvcRead function block

I/O type	Name	Data type	Description
VAR_INPUT	Execute	BOOL	Processing of the function block enabled (once, edge-controlled)
	BusMaster	IL_BUSMASTER	Master ID. If simultaneously operated on the control, multiple masters are distinguished by the BusMaster input variable. If operated alone, a single master has master ID = IL_BUSMASTER_0.
	SercosAdr	UINT	sercos III address of the slave device.
	Element	IL_SIII_ELEMENT	sercos data block element. This input is used to decide whether the name, the attribute, the unit, the minimum value, the maximum value or the date of a parameter will be written. With sercos data block element 1 (Element := IL_STATUS;), the service channel "Data Status" is read.
	Idn	MB_IDN	IDN of the parameter. See function IL_SIIIElementsToIdn page 259.
	SizeOfValue	UDINT	Size in bytes of the buffer or the variables provided for data reception (SIZEOF(Value)).
	ValueAdr	POINTER TO BYTE	Address of the buffer or the variables provided for data reception
	Timeout	TIME	Timeout monitoring of the function block in ms. T#0ms = Timeout monitoring not active.
VAR_OUTPUT	Done	BOOL	Processing completed without errors, output data is valid.
	Active	BOOL	Processing active, output data is invalid.
	Error	BOOL	Processing completed with errors.
	ErrorID	ERROR_CODE	Diagnostics description in case of error.
	ErrorIdent	ERROR_STRUCT	Detailed diagnostics
	NoOfRecBytes	UDINT	Number of bytes copied to the ValueAdr buffer.

Fig.11-231: IL_SIIISvcRead interface

Min./max. and default values of the inputs

Name	Type	Min. value	Max. value	Default value	Effective
Execute	BOOL			FALSE	Continuous
BusMaster	USINT	IL_BUSMASTER_0	IL_BUSMASTER_0	IL_BUSMASTER_0	Rising edge at "Execute"
SercosAdr	UINT	0 (MLD: locale axis) 1 (other systems)	511	0	Rising edge at "Execute"
Element	IL_SIII_ELEMENT	1	7	7	Rising edge at "Execute"
Idn	MB_IDN	0	16#FFFFFFFF	0	Rising edge at "Execute"
SizeOfValue	UDINT	0	16#FFFF	0	Rising edge at "Execute"
ValueAdr	POINTER TO BYTE	n.def.	n.def.	0	Rising edge at "Execute"
Timeout	TIME	T#0s	T#60 min	T#2s	Rising edge at "Execute"

Fig.11-232: Minimum / maximum values and default values of the IL_SIIISvcRead inputs

Error Handling

This function block uses error table **SERCOS_TABLE** (ERROR_TABLE = 16#0010) for representing errors of the sercos service channel.

In addition, the following errors of the function block can be found in error table **F_RELATED_TABLE** (ERROR_TABLE = 16#0170).

ErrorID	Additional1	Additional2	
INPUT_RANGE_ERROR	16#00000002	16#00000001	The input value of parameter "BusMaster" is unequal to IL_BUSMASTER_0 and outside of the input value range.
INPUT_RANGE_ERROR	16#00000002	16#00000002	The input value of parameter "SercosAdr" is 0 or >511. MLD: The input value of parameter "SercosAdr" > 511.
INPUT_RANGE_ERROR	16#00000002	16#00000003	The input value of parameter "Element" is <> 7.
INPUT_RANGE_ERROR	16#00000002	16#00000004	The input value of the parameter "Idn“ is > 16#FFFFFFFF
INPUT_RANGE_ERROR	16#00000002	16#00000006	The input value of parameter "ValueAdr" is 0.
INPUT_RANGE_ERROR	16#00000002	16#00000007	The input value of parameter "Timeout" is >T#60 min.
STATE_MACHINE_ERROR	16#00000006	16#00000001	Error in the state machine.
OTHER_ERROR	16#00000015	16#00000001	A time monitoring error was detected.

Fig.11-233: Error codes of function block IL_SIIISvcRead in F_RELATED_TABLE := 16#0170

Example

Reading single parameters and data function block elements

When reading single parameters and data function block elements, the data type of the variables at the "ValueAdr" input **should** fit to the data size of the element to be read.

In the following example, the sercos parameter S-0-1002.0.0, sercos cycle time, is read.

All sercos data block elements of the parameter are read.

Auxiliary structures

```
TYPE LIST_STRING:
STRUCT
 uiCurLength: UINT;
 uiMaxLength: UINT;
 strValue:    STRING;
END_STRUCT
END_TYPE
```

Variable definition

```
VAR
 fbSIIISvcRead:                  IL_SIIISvcRead;

 uiServiceChannelStatus_E1:      UINT;
 tSercosCycleTimeName_E2:        LIST_STRING;
 dwSercosCycleTime_Attribute_E3: DWORD;
 tSercosCycleTime_Unit_E4:       LIST_STRING;
 udiSercosCycleTime_Min_E5:      UDINT;
 udiSercosCycleTime_Max_E6:      UDINT;
 udiSercosCycleTime_Value_E7:    UDINT;
END_VAR
```

Reading data status of the service channel

```
 fbSIIISvcRead.Execute:=     TRUE;
 fbSIIISvcRead.SercosAdr:=   65;
 fbSIIISvcRead.Element:=     IL_STATUS;
 fbSIIISvcRead.Idn:=         IL_SIIIElementsToIdn(IL_S_PARAM,0,1002,0,0);
 fbSIIISvcRead.SizeOfValue:= SIZEOF(uiServiceChannelStatus_E1);
 fbSIIISvcRead.ValueAdr:=    ADR(uiServiceChannelStatus_E1);

 fbSIIISvcRead();
```

Reading parameter name

```
 fbSIIISvcRead.Execute:=     TRUE;
 fbSIIISvcRead.SercosAdr:=   65;
 fbSIIISvcRead.Element:=     IL_NAME;
 fbSIIISvcRead.Idn:=         IL_SIIIElementsToIdn(IL_S_PARAM,0,1002,0,0);
 fbSIIISvcRead.SizeOfValue:= SIZEOF(tSercosCycleTimeName_E2);
 fbSIIISvcRead.ValueAdr:=    ADR(tSercosCycleTimeName_E2);

 fbSIIISvcRead();

IF fbSIIISvcRead.Done = TRUE THEN
 tSercosCycleTimeName_E2.strValue[tSercosCycleTimeName_E2.uiCurLength]:=0;
END_IF
```

Reading parameter attribute

```
 fbSIIISvcRead.Execute:=     TRUE;
 fbSIIISvcRead.SercosAdr:=   65;
 fbSIIISvcRead.Element:=     IL_ATTRIBUTE;
 fbSIIISvcRead.Idn:=         IL_SIIIElementsToIdn(IL_S_PARAM,0,1002,0,0);
 fbSIIISvcRead.SizeOfValue:= SIZEOF(dwSercosCycleTime_Attribute_E3);
 fbSIIISvcRead.ValueAdr:=    ADR(dwSercosCycleTime_Attribute_E3);

 fbSIIISvcRead();
```

Reading parameter unit

```
 fbSIIISvcRead.Execute:=     TRUE;
 fbSIIISvcRead.SercosAdr:=   65;
 fbSIIISvcRead.Element:=     IL_UNIT;
 fbSIIISvcRead.Idn:=         IL_SIIIElementsToIdn(IL_S_PARAM,0,1002,0,0);
 fbSIIISvcRead.SizeOfValue:= SIZEOF(tSercosCycleTime_Unit_E4);
 fbSIIISvcRead.ValueAdr:=    ADR(tSercosCycleTime_Unit_E4);

 fbSIIISvcRead();

IF fbSIIISvcRead.Done = TRUE THEN
 tSercosCycleTimeUnit_E4.strValue[tSercosCycleTimeUnit_E4.uiCurLength]:=0;
END_IF
```

Reading parameter minimum value

```
fbSIIISvcRead.Execute:=     TRUE;
fbSIIISvcRead.SercosAdr:=   65;
fbSIIISvcRead.Element:=     IL_MINVALUE;
fbSIIISvcRead.Idn:=         IL_SIIIElementsToIdn(IL_S_PARAM,0,1002,0,0);
fbSIIISvcRead.SizeOfValue:= SIZEOF(udiSercosCycleTime_Min_E5);
fbSIIISvcRead.ValueAdr:=    ADR(udiSercosCycleTime_Min_E5);

fbSIIISvcRead();
```

Reading parameter maximum value

```
fbSIIISvcRead.Execute:=     TRUE;
fbSIIISvcRead.SercosAdr:=   65;
fbSIIISvcRead.Element:=     IL_MAXVALUE;
fbSIIISvcRead.Idn:=         IL_SIIIElementsToIdn(IL_S_PARAM,0,1002,0,0);
fbSIIISvcRead.SizeOfValue:= SIZEOF(udiSercosCycleTime_Max_E6);
fbSIIISvcRead.ValueAdr:=    ADR(udiSercosCycleTime_Max_E6);

fbSIIISvcRead();
```

Reading parameter date

```
fbSIIISvcRead.Execute:=     TRUE;
fbSIIISvcRead.SercosAdr:=   65;
fbSIIISvcRead.Element:=     IL_OPDATA;
fbSIIISvcRead.Idn:=         IL_SIIIElementsToIdn(IL_S_PARAM,0,1002,0,0);
fbSIIISvcRead.SizeOfValue:= SIZEOF(udiSercosCycleTime_Value_E7);
fbSIIISvcRead.ValueAdr:=    ADR(udiSercosCycleTime_Value_E7);

fbSIIISvcRead();
```

After reading has been successfully completed, a value of 100,000 is in "ValueAdr" for example.

Parameter S-0-1002 has three decimal places. See attribute.

Thus, the sercos cycle time is 1,000 µs.

☞ If a parameter with the display format FLOAT is used, the input parameter of "ValueAdr" should be provided with the data type REAL.

For a parameter with a decimal value with decimal places, the data type Integer has to be used and the decimal places have to be considered.

Reading lists

In the following example, the sercos parameter S-0-17.0.0, IDN list of all operation data, is read.

When reading lists, it has to be considered that the actual length of a list and the maximum length are part of the list.

2 bytes	2 bytes	<ActualLength> bytes
Actual length	Maximum-Length	User data

Auxiliary structures

```
TYPE LIST_DINT:
 STRUCT
  uiCurLength: UINT;
  uiMaxLength: UINT;
  adiValue:    ARRAY[0..10000] OF DINT;
 END_STRUCT
END_TYPE
```

Variable definition

```
VAR
 fbSIIISvcRead:        IL_SIIISvcRead;
 tListOfIdn_Value_E7: LIST_DINT;
END_VAR
```

Reading parameter date from S-0-0017.0.0 list

```
 fbSIIISvcRead.Execute:=     TRUE;
 fbSIIISvcRead.SercosAdr:=   65;
 fbSIIISvcRead.Element:=     IL_OPDATA;
 fbSIIISvcRead.Idn:=         IL_SIIIElementsToIdn(IL_S_PARAM,0,17,0,0);
 fbSIIISvcRead.SizeOfValue:= SIZEOF(tListOfIdn_Value_E7);
 fbSIIISvcRead.ValueAdr:=    ADR(tListOfIdn_Value_E7);

 fbSIIISvcRead();
```

11.9.3 IL_SIIISvcWrite

Brief Description

Function block IL_SIIISvcWrite can be used to write parameters of a slave device via the sercos III service channel.

Assignment: target system / library

Target system	Library
MLC	RIL_SercosIII.compiled-library
MTX	RIL_SercosIII.compiled-library
XLC	RIL_SercosIII.compiled-library
MLD	RIL_SercosIII.lib (IndraLogic 1.x)

Fig.11-234: Reference table of the IL_SIIISvcWrite function block

Interface Description

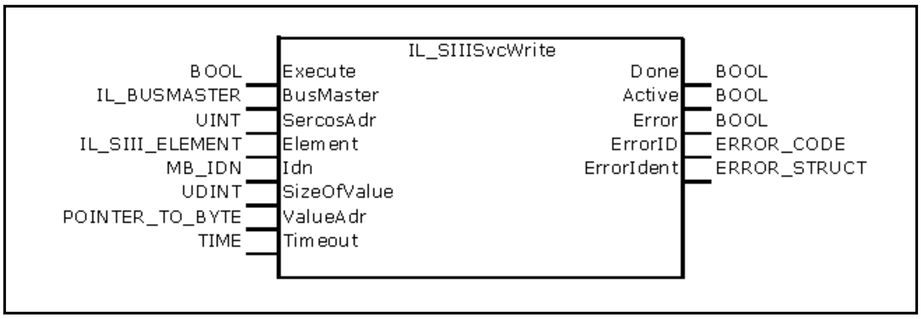


Fig.11-235: IL_SIIISvcRead function block

I/O type	Name	Data type	Description
VAR_INPUT	Execute	BOOL	Processing of the function block enabled (once, edge-controlled)
	BusMaster	IL_BUSMASTER	Master ID. If simultaneously operated on the control, multiple masters are distinguished by the BusMaster input variable. If operated alone, a single master has master ID = IL_BUSMASTER_0.
	SercosAdr	UINT	sercos III address of the slave device.
	Element	IL_SIII_ELEMENT	sercos data block element. Only IL_OPDATA is currently supported.
	Idn	MB_IDN	IDN of the parameter. See function IL_SIIIElementsToIdn page 259.
	SizeOfValue	UDINT	Number of bytes to be written via the service channel.

I/O type	Name	Data type	Description
	ValueAdr	POINTER TO BYTE	Address of the buffer or the variables provided for sending data
	Timeout	TIME	Timeout monitoring of the function block in ms. T#0ms = Timeout monitoring not active.
VAR_OUTPUT	Done	BOOL	Processing completed without errors, output data is valid.
	Active	BOOL	Processing active, output data is invalid.
	Error	BOOL	Processing completed with errors.
	ErrorID	ERROR_CODE	Diagnostics description in case of error.
	ErrorIdent	ERROR_STRUCT	Detailed diagnostics

Fig. 11-236: IL_SIIISvcWrite interface

Min./max. and default values of the inputs

Name	Type	Min. value	Max. value	Default value	Effective
Execute	BOOL			FALSE	Continuous
BusMaster	BYTE	IL_BUSMASTER_0	IL_BUSMASTER_0	IL_BUSMASTER_0	Rising edge at "Execute"
SercosAdr	UINT	0 (MLD: locale axis) 1 (other systems)	511	0	Rising edge at "Execute"
Element	IL_SIII_ELEMENT	7	7	7	Rising edge at "Execute"
Idn	MB_IDN	0	16#FFFFFFFF	0	Rising edge at "Execute"
SizeOfValue	UDINT	0	16#FFFF	0	Rising edge at "Execute"
ValueAdr	POINTER TO BYTE	n.def.	n.def.	0	Rising edge at "Execute"
Timeout	TIME	T#0s	T#60 min	T#2s	Rising edge at "Execute"

Fig. 11-237: Minimum / maximum values and default values of the IL_SIIISvcWrite inputs

Error Handling

This function block uses error table **SERCOS_TABLE** (ERROR_TABLE = 16#0010) for representing errors of the sercos service channel.

In addition, the following errors of the function block can be found in error table **F_RELATED_TABLE** (ERROR_TABLE = 16#0170).

ErrorID	Additional1	Additional2	
INPUT_RANGE_ERROR	16#00000002	16#00000001	The input value of parameter "BusMaster" is unequal to IL_BUSMASTER_0 and outside of the input value range.
INPUT_RANGE_ERROR	16#00000002	16#00000002	The input value of parameter "SercosAdr" is 0 or >511.
INPUT_RANGE_ERROR	16#00000002	16#00000003	The input value of parameter "Element" is <> 7

ErrorID	Additional1	Additional2	
INPUT_RANGE_ERROR	16#00000002	16#00000004	The input value of the parameter "Idn“ is > 16#FFFFFFFF
INPUT_RANGE_ERROR	16#00000002	16#00000005	The input value of parameter "SizeOfValue" is 16#FFFF.
INPUT_RANGE_ERROR	16#00000002	16#00000006	The input value of parameter "ValueAdr" is 0.
INPUT_RANGE_ERROR	16#00000002	16#00000007	The input value of parameter "Timeout" is >T#60 min.
STATE_MACHINE_ERROR	16#00000006	16#00000001	Error in the state machine.
OTHER_ERROR	16#00000015	16#00000001	A time monitoring error was detected.

Fig.11-238: Error codes of function block IL_SIIISvcWrite in F_RELATED_TABLE := 16#0170

Example

Writing single parameters

In the following example, sercos parameter S-0-0099.0.0, Reset command, is written.

Variable definition

```
VAR
 uiDummy:    UINT := 3;
 fbSIIISvcWrite: IL_SIIISvcWrite;
END_VAR
```

Writing parameter S-0-0099.0.0

```
 fbSIIISvcWrite.Execute:=     TRUE;
 fbSIIISvcWrite.SercosAdr:=   0;
 fbSIIISvcWrite.Element:=   IL_OPDATA;
 fbSIIISvcWrite.Idn:=         IL_SIIIElementsToIdn(IL_S_PARAM,0,99,0,0);
 fbSIIISvcWrite.SizeOfValue:= SIZEOF(uiDummy);
 fbSIIISvcWrite.ValueAdr:=    ADR(uiDummy);

 fbSIIISvcWrite ();
```

Writing lists

In the following example, the sercos parameter S-0-1020.0.0, IP-address, is written.

When writing lists, it has to be considered that the actual length of a list and the maximum length are part of the list. Thus, four bytes have to be added to the actual length of the list (see example "fbSIIISvcWrite.SizeOfValue").

2 bytes	2 bytes	<ActualLength> bytes
Actual length	Maximum-Length	User data

Auxiliary structures

```
TYPE LIST_USINT:
 STRUCT
  uiCurLength: UINT;
  uiMaxLength: UINT;
  ausiValue:   ARRAY[0..3] OF USINT;
 END_STRUCT
END_TYPE
```

Variable definition

```
VAR
 fbSIIISvcWrite:          IL_SIIISvcWrite;
 // IndraLogic 2G Syntax
 tIpAddress: LIST_USINT:= (uiCurLength:=4, ausiValue:=[192,186,73,1]);
 (* IndraLogic 1x Syntax
 tIpAddress: LIST_USINT:= (uiCurLength:=4, ausiValue:=192,186,73,1);*)
END_VAR
```

Writing parameter S-0-1020.0.0

```
 fbSIIISvcWrite.Execute:=     TRUE;
 fbSIIISvcWrite.SercosAdr:=   65;
 fbSIIISvcWrite.Idn:=         IL_SIIIElementsToIdn(IL_S_PARAM,0,1020,0,0);
 fbSIIISvcWrite.SizeOfValue:= tIpAddress.uiCurLength + 4;
 fbSIIISvcWrite.ValueAdr:=    ADR(tIpAddress);

 fbSIIISvcWrite ();
```

11.9.4 IL_SIIIElementsToIdn

Brief Description

Function block IL_SIIIElementsToIdn combines the individual elements of an IDN in an MB_IDN value.

Assignment: target system / library

Target system	Library
MLC	RIL_SercosIII.compiled-library
MTX	RIL_SercosIII.compiled-library
XLC	RIL_SercosIII.compiled-library
MLD	RIL_SercosIII.lib (IndraLogic 1.x)

Fig. 11-239: Reference table of the IL_SIIIElementsToIdn function

Interface Description

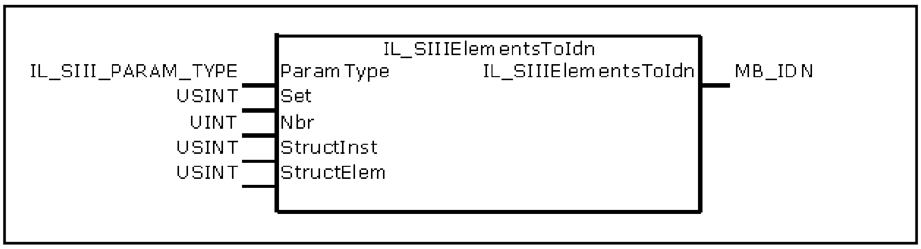


Fig. 11-240: IL_SIIIElementsToIdn function

I/O type	Name	Data type	Description
VAR_INPUT	ParamType	IL_SIII_PARAM_TYPE	Standard (S) or product-specific (P) IDN
	Set	USINT	Parameter set
	Nbr	UINT	Data block number
	StructInst	USINT	Structure instance
	StructElem	USINT	Structure element
Return value	IL_SIIIElementsToIdn	MB_IDN	sercos IDN. If the input parameter is invalid, IL_SIIIElementsToIdn = 0

Fig. 11-241: IL_SIIIElementsToIdn interface

11.9.5 IL_BUSMASTER

Brief description

The IL_BUSMASTER enumeration type of the RIL_FieldbusTypes.library serves to select the appropriate sercos III master.

At present, only the onboard master is supported.

Element	Value	Description
IL_BUSMASTER_0	0	OnBoard master
IL_BUSMASTER_1	1	Not supported
IL_BUSMASTER_2	2	Not supported
IL_BUSMASTER_3	3	Not supported
IL_BUSMASTER_4	4	Not supported

Fig.11-242: Enumeration type IL_BUSMASTER

11.9.6 IL_SIII_ELEMENT

Brief Description

The IL_SIII_ELEMENT enumeration type describes the sercos elements of a parameter.

Name	Value	Description
IL_STATUS	1	Service channel "Data Status"
IL_NAME	2	Name
IL_ATTRIBUTE	3	Attributes
IL_UNIT	4	Unit
IL_MINVALUE	5	Maximum value
IL_MAXVALUE	6	Minimum value
IL_OPDATA	7	Date

Fig.11-243: Elements of enumeration type IL_SIII_ELEMENT

11.9.7 IL_SIII_PARAM_TYPE

Brief Description

The IL_SIII_PARAM_TYPE distinguishes between standard IDN and product-specific IDN.

Name	Value	Description
IL_S_PARAM	0	Standard IDN
IL_P_PARAM	1	Product-specific IDN

Fig.11-244: Elements of enumeration type IL_SIII_PARAM_TYPE

11.10 RIL_Inline.library

11.10.1 RIL_Inline.library, General

The function blocks described below serve for simplified user-compatible output of local Inline I/O diagnostic messages.

Target systems

The library can be used with the following systems:

Target assembly	Remark
CML65	IndraLogic 2G
CML45	IndraLogic 2G

The library contains the following function blocks:

Function blocks for Inline bus diagnostics

- IL_INLNState, page 261, outputs the state of the local Inline IO;
- IL_INLNStateDetails, page 262, outputs the detailed Inline IO diagnostics in order to find the location and cause of the error;
- IL_INLNModuleConfigList, page 265, determines the configured module equipment and the existing one;
- IL_INLNReadCounter, page 269, determines the Inline cycle counters;
- IL_INLNClearCounter, Seite 270, resets the Inline cycle counters.

Function blocks for access to data objects of Inline modules via the PCP channel

- IL_INLNPCPRead, page 272, reading PCP service;
- IL_INLNPCPWrite, page 273, writing PCP service.

11.10.2 IL_INLNState

Brief Description

The function block outputs the state of the local Inline I/O system. The Inline error source is indicated via the "State" bit string.

Target system	Library
XLC 12VRS and higher	RIL_Inline.compiled-library
MLC 12VRS and higher	RIL_Inline.compiled-library
MTX 12VRS and higher	RIL_Inline.compiled-library

Fig. 11-245: Reference table of function block IL_INLNState

Interface Description

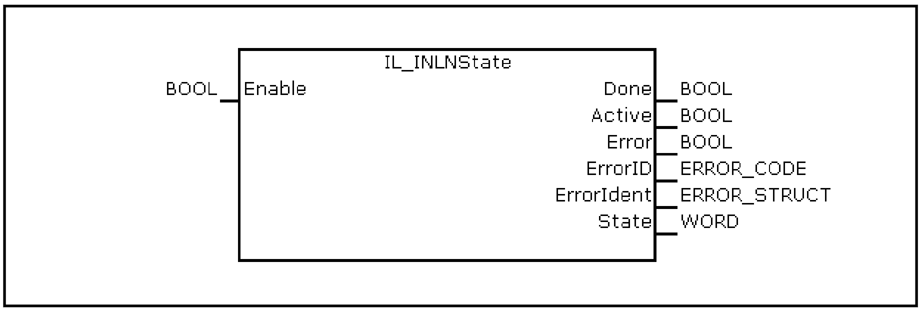


Fig. 11-246: Function block IL_INLNState, interface

I/O type	Name	Data type	Description
VAR_INPUT	Enable	BOOL	Processing enabled for function block (continuous, state-controlled)
VAR_OUTPUT	Done	BOOL	Processing completed without error, output data valid
	Active	BOOL	Processing not yet completed, output data invalid
	Error	BOOL	Processing completed with error, output data invalid
	ErrorID	ERROR_CODE	Standardized rough classification of the error
	ErrorIdent	ERROR_STRUCT	Function-block-specific detailed information on the error
	State	WORD	Inline diagnostic state (see below)

Fig. 11-247: I/O interface of FB IL_INLNState

The "State" output provides a rough diagnostic classification in binary code.

Output "State"

Bit no.	Description	Description
0	IL_INLN_BUS_FAILURE	Inline bus error
1	IL_INLN_MASTER_FAILURE	Inline master stack error
2	IL_INLN_MODULE_ERROR	Inline module error
3	IL_INLN_CONFIG_ERROR	Inline configuration error

Functional Description

The function block outputs the state of the local Inline I/O system. The Inline error source is indicated via the "State" bit string. The local Inline system is free from errors when all bits are reset.

If the operating mode of the local Inline I/O system is essential for the application, the function block should be cyclically called in the application program and branch to an error handling option in the event of an error.

For further information about the location and cause of the error, please refer to function block IL_INLNStateDetails, page 262,.

Error Handling

The function block generates error messages according to error table INLINEIO_TABLE, ERROR_TABLE 16#0190, page 275.

11.10.3 IL_INLNStateDetails

Brief Description

The function block outputs the detailed Inline diagnostics about a data structure in order to determine the location and cause of the error.

Target system	Library
XLC 12VRS and higher	RIL_Inline.compiled-library
MLC 12VRS and higher	RIL_Inline.compiled-library
MTX 12VRS and higher	RIL_Inline.compiled-library

Fig. 11-248: Reference table of function block IL_INLNStateDetails

Interface Description

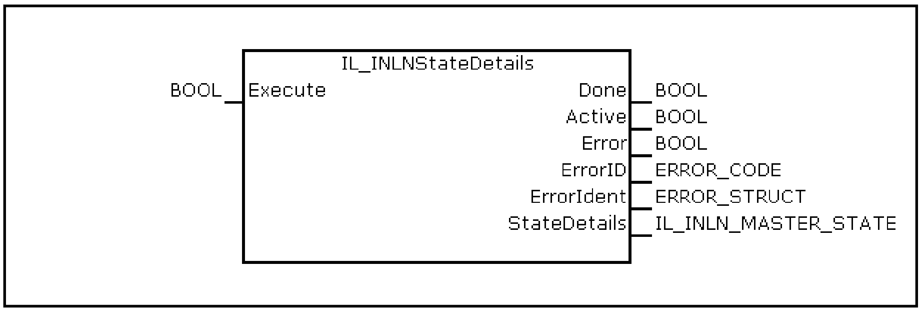


Fig. 11-249: Function block IL_INLNStateDetails, interface

I/O type	Name	Data type	Description
VAR_INPUT	Execute	BOOL	Processing of the function block enabled (once, edge-controlled)
VAR_OUTPUT	Done	BOOL	Processing completed without error, output data valid
	Active	BOOL	Processing not yet completed, output data invalid
	Error	BOOL	Processing completed with error, output data invalid
	ErrorID	ERROR_CODE	Standardized rough classification of the error

I/O type	Name	Data type	Description
	ErrorIdent	ERROR_STRUCT	Function-block-specific detailed information on the error
	StateDetails	IL_INLN_MASTER_STATE	Detailed Inline diagnostic state See IL_INLN_MASTER_STATE, page 264

Fig. 11-250: I/O interface of FB IL_INLNStateDetails

Functional Description

The function block outputs detailed diagnostics about the local Inline system using output data structure IL_INLN_MASTER_STATE, page 264.

Depending on the error sources, further diagnostic information can be read via data structure IL_INLN_MASTER_STATE.

Bus errors

Bus errors (e.g., loose contact or short-circuit on bus line ...) are signaled via bit **IL_ INLN_BUS_FAILURE** = TRUE, FB IL_INLNState, output "State".

In this case, element **BusFailurePosition** shows the position of the defective Inline module. After the error has been corrected, the control must be restarted.

Fatal stack errors

Fatal stack errors (e.g., error in firmware, internal timeout elapsed ...) are signaled via bit **IL_ INLN_MASTER_FAILURE** = TRUE, FB IL_INLNState, output "State".

In this case, element **MasterFailureCode** shows the error number and can be interpreted by Service / Development. After the error has been corrected, the control must be restarted.

Module errors

Module errors (e.g., short-circuit at the outputs of an Inline disk ...) are signaled via bit **IL_ INLN_MODULE** = TRUE, FB IL_INLNState, output "State".

Here, bit strings **ModuleErrorPosition01_32** and **ModuleErrorPosition33_64** can be used to determine the modules involved. Further information about the error cause can only be provided directly via the LEDs of the Inline disk (LED codes). Details are described in the Inline documentation. All modules without error continue working without error; the error is cleared automatically as soon as the error cause has been eliminated (the control does not have to be rebooted).

Element **StoredModuleErrorCount** is used to register every error that occurs.

Configuration error

Configuration errors (e.g., wrong disk inserted, insufficient equipment ...) are signaled via bit **IL_ INLN_CONFIG_ERROR** = TRUE, FB IL_INLNState, output "State".

In this case, element **ConfigErrorCount** shows the number of current configuration errors. Output **NoOfWrongConfigModules** shows the number of improperly configured modules.

Elements **ConfigErrorFirstErrorPosition**…**ConfigErrorLastErrorPosition** show the position of the defective modules.

In addition, outputs **NoOfConfiguredModules** and **NoOfScannedModules** can be used to display the number of configured modules that have been scanned by the hardware.

The complete local Inline I/O system is no longer updated.

- If caused by an improperly inserted Inline disk, the error should be repaired as follows: Switch off the control, replace the Inline disk, switch on the control.
- If caused by an improperly configured Inline disk, the error should be repaired as follows:

  Correct the configuration, log in the application (the control does not have to be restarted)

Element **StoredConfigErrorCount** is used to register every error that occurs.

Error Handling

The function block generates error messages according to error table INLINEIO_TABLE, ERROR_TABLE 16#0190, page 275.

11.10.4 IL_INLN_MASTER_STATE

Structure IL_INLN_MASTER_STATE contains detailed information about the diagnostics of the Inline bus.

Name	Type	Description
Relevant elements if **IL_INLN_BUS_FAILURE** = TRUE		
BusFailurePosition	USINT	Position with errors[1] at the local Inline I/O system.
Relevant elements if **IL_INLN_MASTER_FAILURE** = TRUE		
MasterFailureCode	DWORD	Internal system error number (interpretation by Service / Development)
Relevant elements if **IL_INLN_MODULE_ERROR** = TRUE		
ActModuleErrorCount	USINT	Current module error counter. Is incremented after the error has been detected and decremented after the error has been repaired.
StoredModuleErrorCount	USINT	Stored module error counter. Is incremented whenever an error is detected and set back only in case of a PLC download, reset or restart of the control.
ModuleErrorPosition	ARRAY [1..8] OF BYTES	Bit string for module errors at positions[1] 1…64. A defective module is indicated by a set bit in the bit string.
Relevant elements if **IL_INLN_CONFIG_ERROR** = TRUE		
ConfigErrorCount	USINT	Current configuration error counter. Is incremented after the error has been detected and decremented after the error has been eliminated.
StoredConfigErrorCount	USINT	Stored configuration error counter. Is incremented whenever an error is detected and set back only in case of a PLC download, reset or restart of the control.
ConfigErrorFirstErrorPosition	USINT	First position[1] of a spread area which contains Inline modules whose configuration is different from the actually inserted modules
ConfigErrorLastErrorPosition	USINT	Last position[1] of a spread area which contains Inline modules whose configuration is different from the actually inserted modules
NoOfConfiguredModules	USINT	Number of configured modules
NoOfActivatedModules	USINT	Number of activated modules

Name	Type	Description
NoOfScannedModules	USINT	Number of scanned modules
NoOfWrongConfigModules	USINT	Number of modules whose configuration is different from the actually inserted modules

1) Counting starting from the left with starting value 1 (example: position = 3 -> 3rd Inline disk from the left)

Fig.11-251: Elements of structure IL_INLN_MASTER_STATE

Functional Description

The "rough diagnostics" of the function block IL_INLNState, page 261, can set the four bits described in the table below, in order to signal the respective error.

Output "State" of function block IL_INLNState

Bit no.	Description	Description
0	IL_INLN_BUS_FAILURE	Inline bus error
1	IL_INLN_MASTER_FAILURE	Inline master stack error
2	IL_INLN_MODULE_ERROR	Inline module error
3	IL_INLN_CONFIG_ERROR	Inline configuration error

If one of these bits becomes active, the detailed information about the error is available in the "StateDetails" output variable of type IL_INLN_MASTER_STATE in the above-described sections of the structure, after a function block instance of IL_INLNStateDetails, page 262, has been called.

11.10.5 IL_INLNModuleConfigList

Brief Description

The function block outputs a table with the configured Inline modules and those that are fitted.

Target system	Library
XLC 12VRS and higher	RIL_Inline.compiled-library
MLC 12VRS and higher	RIL_Inline.compiled-library
MTX 12VRS and higher	RIL_Inline.compiled-library

Fig.11-252: Reference table of function block IL_INLNModuleConfigList

Interface Description

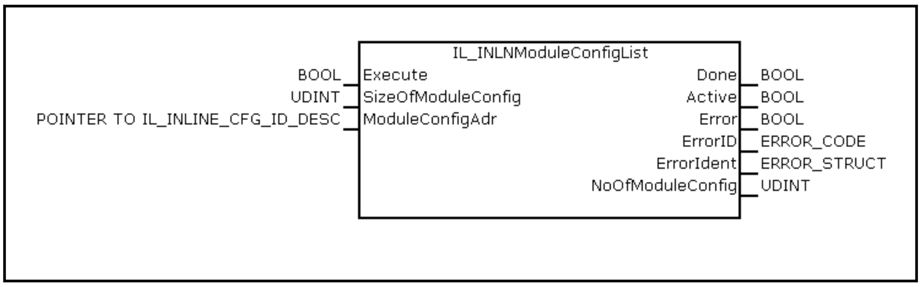


Fig.11-253: Function block IL_INLNModuleConfigList, interface

I/O type	Name	Data type	Description
VAR_INPUT	Execute	BOOL	Processing of the function block enabled (once, edge-controlled)
	SizeOfModuleConfig	UDINT	Size of the data range in bytes, which is addressed with ModuleConfigAdr

I/O type	Name	Data type	Description
	ModuleConfigAdr	POINTER TO IL_INLINE_CFG_ID_DESC	Initial address of the data range
VAR_OUTPUT	Done	BOOL	Processing completed without error, output data valid
	Active	BOOL	Processing not yet completed, output data invalid
	Error	BOOL	Processing completed with error, output data invalid
	ErrorID	ERROR_CODE	Standardized rough classification of the error
	ErrorIdent	ERROR_STRUCT	Function-block-specific detailed information on the error
	NoOfModuleConfig	WORD	Number of determined table entries of type IL_INLINE_CFG_ID_DESC, page 266

Fig. 11-254: I/O interface of FB IL_INLNModuleConfigList

Functional Description

The function block outputs a 2-column table in the data range addressed by ModuleConfigAdr. The first column shows the configured and the second column the fitted Inline modules with their Inline ID code. By comparing the ID codes in relation to the position, configuration and/or fitting errors can be immediately detected.

A table entry is defined by IL_INLINE_CFG_ID_DESC (two WORDs).

The data range to be provided for the table is to be declared as

```
ARRAY [1..n] OF IL_INLINE_CFG_ID_DESC
```

.

Error Handling

The function block generates error messages according to error table INLINEIO_TABLE, ERROR_TABLE 16#0190, page 275.

11.10.6 IL_INLINE_CFG_ID_DESC

Type definition

```
TYPE  IL_INLINE_CFG_ID_DESC:
    STRUCT
        ModuleIdCFG :WORD; // Module ID of configured modules
        ModuleIdSCN :WORD; // Module ID of scanned modules
    END_STRUCT
END_TYPE
```

Structure of a module ID

Module ID

Highbyte	Lowbyte
ID code	Length code

REXROTH Inline module	Parts number	ID code Hex	Length code Hex	ID code Dec
Digital inputs				
R-IB IL 24 DI 16	R911289290	BE	01	190
R-IB IL 24 DI 16-PAC	R911170752	BE	01	190
R-IB IL 24 DI 16-2MBD-PAC	R911170408	BE	01	190
R-IB IL 24 DI 32/HD	R911297188	BE	02	190
R-IB IL 24 DI 32/HD-PAC	R911170753	BE	02	190

REXROTH Inline module	Parts number	ID code Hex	Length code Hex	ID code Dec
R-IB IL 24 DI 32/HD-NPN-PAC	R911170405	BE	02	190
R-IB IL 24 DI 4	R911289287	BE	41	190
R-IB IL 24 DI 4-PAC	R911170750	BE	41	190
R-IB IL 24 EDI 2-DES	R911289292	BE	41	190
R-IB IL 24 DI 16-NPN-PAC	R911170404	BE	41	190
R-IB IL 24 DI 8	R911289288	BE	81	190
R-IB IL 24 DI 8-PAC	R911170751	BE	81	190
R-IB IL 24 DI 8-2MBD-PAC	R911170407	BE	81	190
R-IB IL 24 DI 2	R911289286	BE	C2	190
R-IB IL 24 DI 2-PAC	R911170767	BE	C2	190
R-IB IL 24 DI 2-NPN-PAC	R911170403	BE	C2	190
Infeed and segment terminals				
IB IL 24 SEG/F-D-PAC	R911170710	BE	C2	190
R-IB IL 24 PWR IN/2F-D-2MBD -PAC	R911170447	BE	C2	190
R-IB IL 24 SEG/F-D-2MBD -PAC	R911170448	BE	C2	190
Digital outputs				
R-IB IL 24 DO 16	R911289299	BD	01	189
R-IB IL 24 DO 16-PAC	R911170757	BD	01	189
R-IB IL 24 DO 16-2MBD-PAC	R911170415	BD	01	189
R-IB IL 24 DO 32/HD	R911297191	BD	02	189
R-IB IL 24 DO 32/HD-PAC	R911170768	BD	02	189
R-IB IL 24 DO 32/HD-NPN-PAC	R911170411	BD	02	189
R-IB IL 24 DO 4	R911289295	BD	41	189
R-IB IL 24 DO 4-PAC	R911170755	BD	41	189
R-IB IL 24 DO 4-2MBD-PAC	R911170413	BD	41	189
R-IB IL 24 DO 8	R911289297	BD	81	189
R-IB IL 24 DO 8-PAC	R911170756	BD	81	189
R-IB IL 24 DO 8-2A	R911289298	BD	81	189
R-IB IL 24 DO 8-2A-PAC	R911170759	BD	81	189
R-IB IL 24 DO 8-NPN-PAC	R911170410	BD	81	189
R-IB IL 24 DO 8-2MBD-PAC	R911170414	BD	81	189
R-IB IL 24 DO 2-2A	R911289294	BD	C2	189
R-IB IL 24 DO 2-2A-PAC	R911170754	BD	C2	189
R-IB IL 24 DO 2-NPN-PAC	R911170409	BD	C2	189
R-IB IL 24 DO 2-2A-2MBD-PAC	R911170412	BD	C2	189

REXROTH Inline module	Parts number	ID code Hex	Length code Hex	ID code Dec
Relay terminals				
R-IB IL 24/230 DOR 4/W	R911289302	BD	41	189
R-IB IL 24/230 DOR 4/W-PAC	R911170758	BD	41	189
R-IB IL 24/230 DOR 4/W-2MBD-PAC	R911170417	BD	41	189
R-IB IL 24/230 DOR 1/W	R911289301	BD	C2	189
R-IB IL 24/230 DOR 1/W-PAC	R911170769	BD	C2	189
Analog inputs				
R-IB IL AI 8/SF-PAC	R911308493	5F	02	95
R-IB IL AI 8/IS-PAC	R911308494	5F	02	95
R-IB IL AI 8/SF-2MBD-PAC	R911170430	5F	02	95
R-IB IL AI 2/SF	R911289306	7F	02	127
R-IB IL AI 2/SF-PAC	R911170784	7F	02	127
R-IB IL AI 2/SF-230-PAC	R911170425	7F	02	127
R-IB IL TEMP 2 RTD	R911289305	7F	02	127
R-IB IL TEMP 2 RTD-PAC	R911170785	7F	02	127
R-IB IL TEMP 2 UTH-PAC	R911170431	7F	02	127
R-IB IL SGI 2/F-PAC	R911170432	DF	03	223
R-IB IL SGI 2/F-2MBD-PAC	R911170433	DF	03	223
R-IB IL AI 4/EF-PAC	R911170426	DF	05	223
R-IB IL AI 4/EF-2MBD-PAC	R911170427	DF	05	223
R-IB IL TEMP 4/8 RTD-PAC	R911170428	DF	05	223
R-IB IL TEMP 4/8 RTD-2MBD-PAC	R911170429	DF	05	223
R-IB IL SGI 2/P-PAC	R911170434			
R-IB IL SGI 2/P-2MBD-PAC	R911170435			
Analog outputs				
R-IB IL AO 2/U/BP	R911289381	5B	02	91
R-IB IL AO 2/U/BP-PAC	R911170786	5B	02	91
R-IB IL AO 2/SF-PAC	R911170436	5B	02	91
R-IB IL AO 2/SF-2MBD-PAC	R911170437	5B	02	91
R-IB IL AO 1/SF	R911289303	7D	01	125
R-IB IL AO 1/SF-PAC	R911170787	7D	01	125
R-IB IL AO 4/8/U/BP-2MBD-PAC	R911170438	DF	05	223
Functional terminals				
R-IB IL INC-IN-PAC	R911308491	7F	02	127
R-IB IL CNT	R911289315	BE	02	191

REXROTH Inline module	Parts number	ID code Hex	Length code Hex	ID code Dec
R-IB IL CNT-PAC	R911170788	BE	02	191
R-IB IL CNT-2MBD-PAC	R911170439	BE	02	191
R-IB IL INC-PAC	R911308492	BE	02	191
R-IB IL SSI-PAC	R911308594	BE	02	191
R-IB IL PWM/2-PAC	R911170444	BE	02	191
R-IB IL TEMPCON UTH-PAC	R911308596	BE	02	191
R-IL BK DDL	BRP product	BE	04	191
R-IB IL RS232-PRO-PAC	R911170440	BE	06	191
R-IB IL RS232-PRO-2MBD-PAC	R911170441	BE	06	191
R-IB IL RS485/422-PRO-PAC	R911170442	BE	06	191
R-IB IL RS485/422-PRO-2MBD-PAC	R911170443	BE	06	191
Fieldline Modular M8				
RF-FLM DI 8 M8	R911170449	B2	81	178
RF-FLM DIO 8/4 M8	R911170450	B3	81	179

11.10.7 IL_INLNReadCounter

Brief Description

Function block IL_INLNReadCounter determines the Inline cycle counters and provides these at its outputs.

Target system	Library
XLC 12VRS and higher	RIL_Inline.compiled-library
MLC 12VRS and higher	RIL_Inline.compiled-library
MTX 12VRS and higher	RIL_Inline.compiled-library

Fig. 11-255: Reference table of function block IL_INLNReadCounter

Interface Description

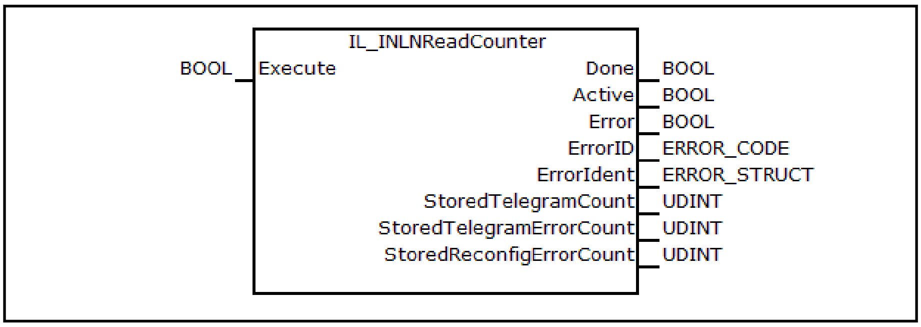


Fig. 11-256: Function block IL_INLNReadCounter, interface

I/O type	Name	Data type	Description
VAR_INPUT	Execute	BOOL	Processing of the function block enabled (once, edge-controlled)
VAR_OUTPUT	Done	BOOL	Processing completed without error, output data valid
	Active	BOOL	Processing not yet completed, output data invalid

I/O type	Name	Data type	Description
	Error	BOOL	Processing completed with error, output data invalid
	ErrorID	ERROR_CODE	Standardized rough classification of the error
	ErrorIdent	ERROR_STRUCT	Function-block-specific detailed information on the error
	StoredTelegram-Count	UDINT	Stored telegram counter. Is incremented with each telegram cycle and is set back only in case of a PLC download, reset or restart of the control.
	StoredTelegramErrorCount	UDINT	Stored telegram failure counter. Is incremented with each telegram failure and is set back only in case of a PLC download, reset or restart of the control.
	StoredReconfigErrorCount	UDINT	Stored reconfiguration error counter. Is incremented whenever an error is detected and set back only in case of a PLC download, reset or restart of the control.

Fig. 11-257: I/O interface of FB IL_INLNReadCounter

Functional Description

Function block IL_INLNReadCounter determines the Inline cycle counters and provides these at its outputs. The cycle counters can be reset via function block IL_INLNClearCounter, page 270,.

Telegram failures (e.g., due to EMC problems) can therefore be detected via output

`StoredTelegramErrorCount`

being > 0.

Element "StoredReconfigErrorCount" is used to register every reconfiguration error that has occurred by means of en error counter.

Error Handling

The function block generates error messages according to error table INLINEIO_TABLE, ERROR_TABLE 16#0190, page 275.

11.10.8 IL_INLNClearCounter

Brief Description

Function block IL_INLNClearCounter resets the Inline cycle counters to 0.

Target system	Library
XLC 12VRS and higher	RIL_Inline.compiled-library
MLC 12VRS and higher	RIL_Inline.compiled-library
MTX 12VRS and higher	RIL_Inline.compiled-library

Fig. 11-258: Reference table of function block IL_INLNClearCounter

Interface Description

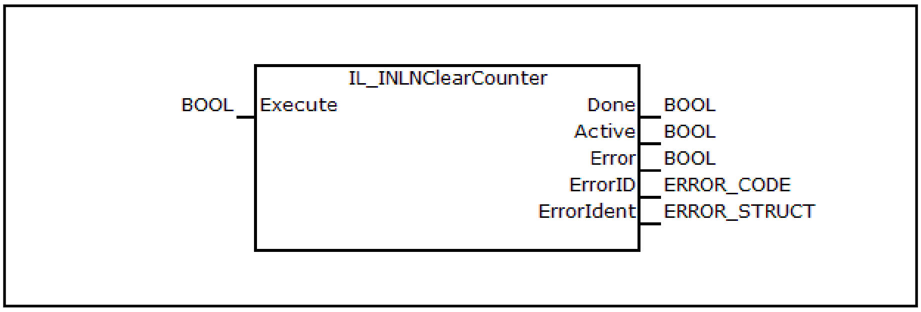


Fig. 11-259: Function block IL_INLNClearCounter, interface

I/O type	Name	Data type	Description
VAR_INPUT	Execute	BOOL	Processing of the function block enabled (once, edge-controlled)
VAR_OUTPUT	Done	BOOL	Processing completed without error, output data valid
	Active	BOOL	Processing not yet completed, output data invalid
	Error	BOOL	Processing completed with error, output data invalid
	ErrorID	ERROR_CODE	Standardized rough classification of the error
	ErrorIdent	ERROR_STRUCT	Function-block-specific detailed information on the error

Fig. 11-260: I/O interface of FB IL_INLNClearCounter

Error Handling The function block generates error messages according to error table INLINEIO_TABLE, ERROR_TABLE 16#0190, page 275.

11.10.9 PCP Data Transfer, General

The Inline system features two data channels in parallel.

- Process data channel
- Parameter data channel

Both data channels are integrated into the transmission protocol. Cyclic data traffic is controlled via the process data channel. The parameter data channel provides for parameterization of the modules and is mapped in the PCP.

The "PCP (Peripherals Communication Protocol)" driver transmits the parameter data to the PCP-based Inline component.

All devices in the Inline system are considered to be a single logic device. In each cycle, the entire process data and parameter data information is simultaneously transmitted to all devices in a summation frame. Based on the transmission position of the individual pieces of information in the summation frame, each device can apply the data intended for it.

If they want to deliver their own data, the devices must deposit the data at the same position because the summation frame is read back in the same order.

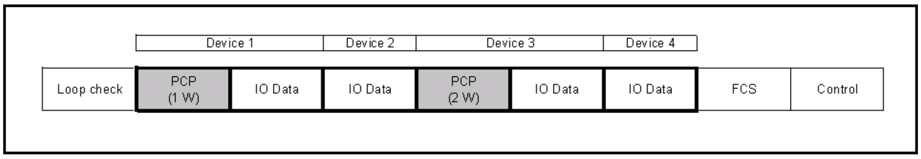


Fig. 11-261: Summation frame

Please note that the PCP data is each positioned at the beginning of the transmission to the particular modules.

11.10.10 IL_INLNPCPRead

Brief Description

Function block IL_INLNPCPRead performs a reading access to data objects of Inline modules via the PCP channel of the Inline IO.

Target system	Library
XLC 12VRS and higher	RIL_Inline.compiled-library
MLC 12VRS and higher	RIL_Inline.compiled-library
MTX 12VRS and higher	RIL_Inline.compiled-library

Fig.11-262: Reference table of function block IL_INLNPCPRead

Interface Description

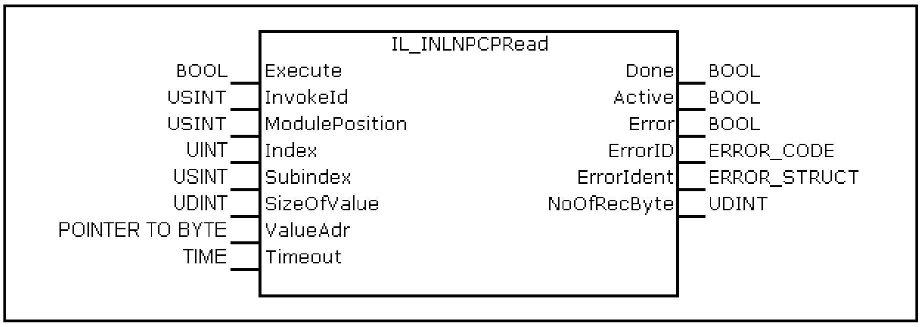


Fig.11-263: Function block IL_INLNPCPRead, interface

I/O type	Name	Data type	Description
VAR_INPUT	Execute	BOOL	Processing of the function block enabled (once, edge-controlled)
	InvokeId	USINT	Job number for parallel services
	ModulePosition	USINT	Position of the Inline module
	Index	UINT	Logical address of an object that is to be read
	SubIndex	USINT	Logical sub-address of an object element
	SizeOfValue	UDINT	(Size of the data range addressed with ValueAdr) Size of the data object to be read
	ValueAdr	POINTER TO BYTE	Initial address of the data range
	Timeout	TIME	Timeout monitoring of the function block in ms, T#0ms = Timeout monitoring not active
VAR_OUTPUT	Done	BOOL	Processing completed without error, output data valid
	Active	BOOL	Processing not yet completed, output data invalid
	Error	BOOL	Processing completed with error, output data invalid
	ErrorID	ERROR_CODE	Standardized rough classification of the error
	ErrorIdent	ERROR_STRUCT	Function-block-specific detailed information on the error
	NoOfRecBytes	UDINT	Number of bytes received in the data range

Fig.11-264: I/O interface of FB IL_INLNPCPRead

Min./max. and default values of the inputs

Name	Type	Min. value	Max. value	Default value	Effective
Execute	BOOL			FALSE	Continuous
InvokeId	USINT	0	255	0	Rising edge at "Execute"
ModulePosition	USINT	1	64	0	Rising edge at "Execute"
Index	UINT	0	65535	0	Rising edge at "Execute"
SubIndex	USINT	1	255	0	Rising edge at "Execute"
SizeOfValue	UDINT	0	nn	0	Rising edge at "Execute"
ValueAdr	POINTER TO BYTE	n.def.	n.def.	0	Rising edge at "Execute"
Timeout	TIME	0	nn	0	Rising edge at "Execute"

Functional Description

Function block IL_INLNPCPRead performs a reading access to data objects of Inline modules via the PCP channel of the Inline IO.

The module involved is addressed via ModulePosition. The data object is defined by the index and subindex as well as the size SizeOfValue.

The effectively read data length is shown in output NoOfRecBytes and can be less than the requested data length.

Error Handling

The function block generates error messages according to error table INLINEIO_TABLE, ERROR_TABLE 16#0190, page 275.

11.10.11 IL_INLNPCPWrite

Brief Description

Function block IL_INLNPCPWrite performs a writing access to data objects of Inline modules via the PCP channel of the Inline IO.

Target system	Library
XLC 12VRS and higher	RIL_Inline.compiled-library
MLC 12VRS and higher	RIL_Inline.compiled-library
MTX 12VRS and higher	RIL_Inline.compiled-library

Fig.11-265: Reference table of function block IL_INLNPCPWrite

Interface Description

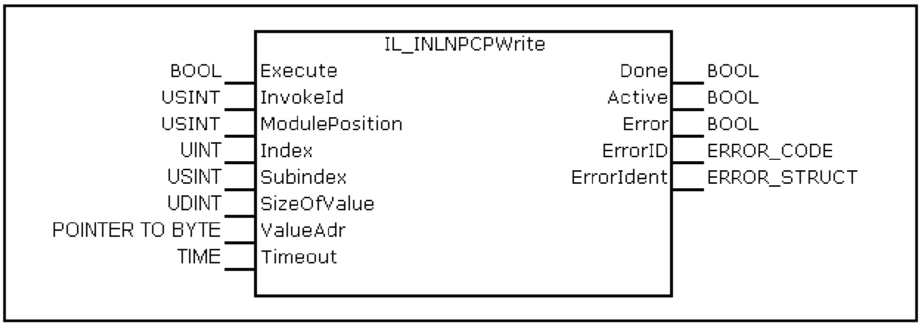


Fig.11-266: Function block IL_INLNPCPWrite, interface

I/O type	Name	Data type	Description
VAR_INPUT	Execute	BOOL	Processing of the function block enabled (once, edge-controlled)
	InvokeId	USINT	Job number for parallel services
	ModulePosition	USINT	Position of the Inline module
	Index	UINT	Logical address of an object that is to be written
	Subindex	USINT	Logical sub-address of an object element
	SizeOfValue	UDINT	(Size of the data range addressed with ValueAdr) Size of the data object to be written
	ValueAdr	POINTER TO BYTE	Initial address of the data range
	Timeout	TIME	Timeout monitoring of the function block in ms, T#0ms = Timeout monitoring not active
VAR_OUTPUT	Done	BOOL	Processing completed without error, output data valid
	Active	BOOL	Processing not yet completed, output data invalid
	Error	BOOL	Processing completed with error, output data invalid
	ErrorID	ERROR_CODE	Standardized rough classification of the error
	ErrorIdent	ERROR_STRUCT	Function-block-specific detailed information on the error

Fig.11-267: I/O interface of FB IL_INLNPCPWrite

Min./max. and default values of the inputs

Name	Type	Min. value	Max. value	Default value	Effective
Execute	BOOL			FALSE	Continuous
InvokeId	USINT	0	255	0	Rising edge at "Execute"
ModulePosition	USINT	1	64	0	Rising edge at "Execute"
Index	UINT	0	65535	0	Rising edge at "Execute"
SubIndex	USINT	1	255	0	Rising edge at "Execute"
SizeOfValue	UDINT	0	nn	0	Rising edge at "Execute"
ValueAdr	POINTER TO BYTE	n.def.	n.def.	0	Rising edge at "Execute"
Timeout	TIME	0	nn	0	Rising edge at "Execute"

Functional Description

Function block IL_INLNPCPWrite performs a writing access to data objects of Inline modules via the PCP channel of the Inline IO.

The module involved is addressed via ModulePosition. The data object is defined by the index and subindex as well as the size SizeOfValue.

Error Handling

The function block generates error messages according to error table INLINEIO_TABLE, ERROR_TABLE 16#0190, page 275.

11.10.12 INLINEIO_TABLE, ERROR_TABLE 16#0190

INLINEIO_TABLE - ERROR_TABLE, table number 16#190

Type definition

```
TYPE  ERROR_STRUCT :
    STRUCT
        Table        :ERROR_TABLE;
        Additional1 :DWORD;
        Additional2 :DWORD;
    END_STRUCT
END_TYPE
```

Description	Type	Description
Table	ERROR_TABLE	Indicates the "error table" from which the error numbers are entered in ErrorAdditional, here: 16#190.
Additional1	DWORD	Varies in its assignment, depending on ErrorTable, e.g.,: INLINE error...
Additional2	DWORD	If necessary, given as additional error information, depending on ErrorTable

Fig. 11-268: Error Struct

ErrorCode	ErrorIdent		Description
	Additional1	Additional2	
------ User errors ------			
INPUT_INVALID_ERROR	1311	[module position]	Invalid module position
INPUT_INVALID_ERROR	2000	0	Parameter error FB
------ Diagnostics ------			
NONE_ERROR	0	0	No error
ACCESS_ERROR	1310	0	Internal access error, diag module not ready
ACCESS_ERROR	1410	0	Internal access error, config module not ready
ACCESS_ERROR	1411	0	Internal access error, invalid module list
ACCESS_ERROR	1412	0	Internal access error, invalid module scan
ACCESS_ERROR	1413	0	Internal access error, invalid module position
ACCESS_ERROR	1414	0	Internal access error, no module entries available
ACCESS_ERROR	1420	[call position]	Internal access error, driver not ready
ACCESS_ERROR	1421	[call position]	Internal access error, execution error
DEVICE_ERROR	1330	0	Internal error, invalid diag type
DEVICE_ERROR	1331	0	Internal error, invalid list index

ErrorCode	ErrorIdent		Description
	Additional1	Additional2	
DEVICE_ERROR	1332	0	Internal error, ring buffer overflow
DEVICE_ERROR	1333	0	Internal error, module state conflict
DEVICE_ERROR	1334	0	Internal error, invalid buffer index
OTHER_ERROR	1188	0	Undefined error
OTHER_ERROR	1199	0	Undefined error
SYSTEM_ERROR	1177	0	Internal system error
SYSTEM_ERROR	2000	0	Internal system error FB
---- IL_PCPRead und IL_PCPWrite ----			
INPUT_INVALID_ERROR	1510	0	Module no PCP module
COMMUNICATION_ERROR	1512	0	Config data not available
COMMUNICATION_ERROR	1514	0	PCP module data not available
INPUT_INVALID_ERROR	1516	0	Module position invalid
DEVICE_ERROR	1518	0	No connection to PCP module
RESSOURCE_ERROR	1520	0	Too many PCP requests to one Module
RESSOURCE_ERROR	1522	0	Too many PCP requests total
ACCESS_ERROR	1525	0	PCP request canceled
ACCESS_ERROR	1530	0	Inline driver not ready
ACCESS_ERROR	1531	0	Internal access error
ACCESS_ERROR	1532	0	PCP driver not ready
STATE_MACHINE_ERROR	1550	0	Call sequence invalid
STATE_MACHINE_ERROR	1551	0	Request reference invalid
SYSTEM_ERROR	1552	0	Request type invalid
SYSTEM_ERROR	1553	0	Request type invalid
DEVICE_ERROR	1555	0	Request rejected
DEVICE_ERROR	1557	0	Communication timeout
DEVICE_ERROR	1559	0	Application timeout
DEVICE_ERROR	1561	0	Request state invalid
DEVICE_ERROR	1562	0	Request rejected
DEVICE_ERROR	1563	0	Communication reference invalid
DEVICE_ERROR	1564	0	Request reference invalid
COMMUNICATION_ERROR	1565	0	Communication reference invalid
COMMUNICATION_ERROR	1566	0	Service error in communication
COMMUNICATION_ERROR	1567	0	Data length invalid
COMMUNICATION_ERROR	1568	0	Communication reference invalid

ErrorCode	ErrorIdent		Description
	Additional1	Additional2	
COMMUNICATION_ERROR	1569	0	Service error in communication
COMMUNICATION_ERROR	1570	0	Data length invalid

Fig.11-269: Error table INLINEIO_TABLE := 16#0190

12 Service and Support

Our worldwide service network provides an optimized and efficient support. Our experts offer you advice and assistance should you have any queries. You can contact us **24/7**.

Service Germany

Our technology-oriented Competence Center in Lohr, Germany, is responsible for all your service-related queries for electric drive and controls.

Contact the **Service Helpdesk & Hotline** under:

Phone:	**+49 9352 40 5060**
Fax:	**+49 9352 18 4941**
E-mail:	service.svc@boschrexroth.de
Internet:	http://www.boschrexroth.com

Additional information on service, repair (e.g. delivery addresses) and training can be found on our internet sites.

Service worldwide

Outside Germany, please contact your local service office first. For hotline numbers, refer to the sales office addresses on the internet.

Preparing information

To be able to help you more quickly and efficiently, please have the following information ready:

- Detailed description of malfunction and circumstances resulting in the malfunction
- Type plate name of the affected products, in particular type codes and serial numbers
- Your contact data (phone and fax number as well as your email address)

Index

Index

Index

Notes

Rexroth
Bosch Group

Bosch Rexroth AG
Electric Drives and Controls
P.O. Box 13 57
97803 Lohr, Germany
Bgm.-Dr.-Nebel-Str. 2
97816 Lohr, Germany
Tel. +49 9352 18 0
Fax +49 9352 18 8400
www.boschrexroth.com/electrics