Manual

AUTOMATION

WAGO SPEEDWAY 767

CANopen, 8DI, 24VDC

767-2501

Programmable Fieldbus Coupler

Version 2.1.1

WAGO®
INNOVATIVE CONNECTIONS

Introduction

Every conceivable measure has been taken to ensure the accuracy and completeness of this documentation. However, as errors can never be fully excluded, we always appreciate any information or suggestions for improving the documentation. Please feel free to e-mail us at documentation@wago.com.

Service and Technical Support

Additional information regarding this and other products (e.g., data sheets) is available on our website: www.wago.com.

If you can not eliminate errors or faults using the measures described in this manual we will be glad to assist you further. Contact us at:

AUTOMATION Support
Phone: +49 571 887 555
Fax: +49 571 887 8555
E-mail: support@wago.com

Additional Support

The Seminar and Training department offers useful seminars to support you in the use of WAGO products. For more information, visit our website or call +49 571 887-327, or send an e-mail to training@wago.com.

Trademarks

We wish to point out that the software and hardware terms as well as the trademarks of companies used and/or mentioned in this manual are generally protected by trademark or patent.

Table of Contents

1 Notes about these Operating Instructions

The fieldbus coupler shall only be installed and operated in conjunction with these operating instructions and the system description.

⚠ WARNING

Observe release notes!
Please note that, within the SPEEDWAY system, a function is provided **without restriction** only if all system's components have the same system-wide firmware release.Therefore, always observe the appropriate release notes on products used.

NOTICE

Supply layout!
In addition to these operating instructions, you will need the "WAGO SPEEDWAY 767, System Description and Information" manual, which can be downloaded at www.wago.com. There you will find information regarding supply layout, etc.

Note

The operating instructions is part of the product and must be kept during the entire service life of the fieldbus coupler. The instructions must be passed to any subsequent owner or user of the fieldbus coupler. In addition, it must be ensured that any supplement to the instructions is also included with the instructions.

1.1 Validity of these Operating Instructions

These operating instructions are only applicable to the positive-switching WAGO SPEEDWAY 767 Series programmable fieldbus coupler, item number 767-2501. In this manual, the programmable fieldbus coupler will be referred to hereafter as simply the fieldbus coupler.

1.2 Explanation of Symbols

⚠ DANGER

Warning of physical injury
Indicates a direct hazard with a high level of risk which leads to death or serious physical injury if not avoided.

⚠ DANGER

Warning of physical injury from electric current
Indicates a direct hazard with a high level of risk which leads to death or serious physical injury if not avoided.

⚠ WARNING

Warning of physical injury
Indicates a possible hazard with a moderate level of risk, which may lead to death or (serious) physical injury if not avoided.

⚠ CAUTION

Warning of physical injury
Indicates possible hazards with a low level of risk, which could lead to minor or moderate physical injuries if not avoided.

NOTICE

Warning of damage to equipment
Indicates a possible hazard that could lead to equipment damage if not avoided.

NOTICE

Warning of damage to equipment from electrostatic discharge
Indicates a possible hazard that could lead to equipment damage if not avoided.

Note

Please note
Indicates possible malfunction, which does not lead to equipment damage if it is not avoided.

Information

Reference to additional information
Indicates other sources of information which are not an integral part of this documentation, such as the Internet.

2 Safety

2.1 Use in Compliance with Underlying Provisions

The CANopen fieldbus coupler provides digital process data from digital and analog I/O modules. The data is gathered by the fieldbus coupler and made available to a higher-level controller for further processing.

The fieldbus coupler shall not be used to control safety-related functions; i.e., emergency-off devices shall not be operated with this fieldbus coupler.

The fieldbus coupler shall only be used as a unit or in combination with I/O modules from the WAGO SPEEDWAY 767 Series.

The fieldbus coupler was developed for applications requiring IP 67 (NEMA type 6, 6P) protection.

The fieldbus coupler is expandable by a maximum of 64 I/O modules from the WAGO SPEEDWAY 767 Series.

Applications other than those described in this manual are not permitted.

2.2 Personnel Qualification

All sequences implemented on the fieldbus coupler may only be carried out by electrical specialists with sufficient knowledge in automation. The specialists must be familiar with the current standards and guidelines for the fieldbus coupler and automation environment.

All changes to the controller should always be carried out by qualified personnel with sufficient sufficient skills in PLC programming.

2.3 Basic Safety Information

This section contains a summary of the most important warnings, which are also repeated in the individual sections. They serve as a protection to your health and a protection from equipment damage on the 767 Series components (fieldbus coupler and the I/O modules connected to it). Read and adhere to the following safety precautions before using the fieldbus coupler.

⚠ DANGER

Electric voltage!
Operate the 767 Series components exclusively with 24VDC PELV (Protective Extra Low Voltage) or SELV (Safety Extra Low Voltage) voltage sources. Failure to comply may result in electric shock.

⚠ CAUTION

Hot connection sockets!
Even when taking into account derating, high surface temperatures on the metallic connection sockets and on the enclosure can arise during operation. If the 767 Series component has been in operation, allow it to cool off before moving it.

NOTICE

The highest current carrying capacity of the supply contacts is 4A!
Always observe the maximum current carrying capacity per supply line (U_{LS}, U_A) for each 767 Series component and the overall power consumption for all 767 Series component. Neither of these values shall exceed 4A since an increase in current causes the contacts to overheat and damages the 767 Series components. Information regarding the power demand of each 767 Series component can be found in the corresponding data sheet, which is available from www.wago.com.

NOTICE

Exposed connections!
If connections have not been closed with protective caps, liquid or dirt can penetrate the module and ruin it. Close all unused connections with protective caps to ensure that IP67 degree of protection is provided.

- Disconnect the power supply from the system on which you wish to mount the 767 Series component.
- Always keep the cover cap of the DIP switch closed.
- Observe the appropriate accident prevention regulations for your system during assembly, startup, maintenance and repairs.
- The operating instructions of the 767 Series components and the system description must be readily available at the workplace.

- Observe the exact positioning (coding) between plug and socket.
- The 767 Series component shall not come in contact with substances having seeping and insulating properties. Otherwise, additional measures shall be taken for the devices, such as installation of an enclosure that is resistant to the above-mentioned substance properties.
- Electronic components fulfilling the ESD requirements according to the IEC 61000-6-2 are integrated in the 767 Series component. Since, under unfavorable circumstances, higher voltages may also occur due to electrical charge in the field, discharge must be ensured before performing work on the 767 system.
- Observe the correct layout of the potential equalization.
- Keep all cables a sufficient distance away from electromagnetic sources of interference in order to maintain a high level of interference resistance of the 767 system against electromagnetic emissions. Use only shielded cables at the necessary locations, and always observe the appropriate standards for EMC-suitable installations.
- For the power supply and for the S-BUS, use only pre-assembled WAGO system cables, so the specified characteristics of the technical data can be achieved.
- Replace defective or damaged 767 Series component (e.g., deformed connections), else function disruptions can occur in the respective fieldbus stations or nodes.
- When laying any cables, make sure that you do not lay them within the shear range of movable machine parts.
- For each activity, observe the corresponding personnel qualification in Section 2.2.
- Observe the marking on the front and rear side of the 767 Series component.

2.4 Safety Equipment

All products of series 767 are designed according to the IP 67 safety class. This consists of, among other things, complete touch protection of electric voltages and currents – even when wet.

2.5 Technical Condition of the 767 Series components

If any change is made to the 767 Series components or software and firmware without the written approval of WAGO Kontakttechnik GmbH & Co. KG, all liability claims are nullified. Parameterization shall only be carried out within the described scope (Section "Parameterization").

2.6 Notes on Operation

When integrating the 767 Series components in your machine or system, all the currently applicable norms, regulations and guidelines shall be observed during all activities. The emergency stop equipment shall remain effective in all operating modes of the system and machine.

For protection from electromagnetic interferences

- Connect your system to protective earth (PE), and
- Ensure that the cable routing and the installation of the fieldbus cable, S-BUS cable, supply cable, and sensor cable are correct.

The following elements for 24V supply shall be present:

- Outer lightning protection on buildings
- Inner lightning protection of supply lines and signal lines
- Safe electrical separation of low voltage 24VDC through PELV (Protective Extra Low Voltage) or SELV (Safety Extra Low Voltage) voltage sources

3 Information on CANopen

CAN ("Controller Area Network") was developed in the mid-1980s for data transmission in automobiles.

CANopen describes the application level based on the CAN bus system. As a standard application level, CANopen is defined by specification DS 301 of CiA ("CAN in Automation"). The CANopen Network Management enables a simplified start of the CAN bus, which can be expanded within the approved range.

CAN is a multi-master bus system. Contrary to other fieldbus systems, the bus devices connected to the CAN are not addressed; instead, the telegrams are identified. The users are always allowed to send telegrams when there is data traffic on the CAN bus. Bus conflicts are solved because the telegrams are assigned a specific priority. This priority is determined by the COB ID ("Communication Object Identifier") and is unambiguously assigned to a communication object. The smaller the identifier, the higher the priority.

The CANopen specification (DS 301) defines the technical and functional characteristics of CANopen. The CANopen specification allows for additional device profiles, in which the objects and communication parameters of devices in a particular application area are specified. The fieldbus coupler supports device profile DS-401 ("Device Profile for Generic I/O Modules"). The flash signals RUN and ERR LED on the fieldbus coupler correspond to the supplementary CANopen specification "DR303 Part 3 Indicator Specification."

The CANopen protocol implemented in a device is designated as a CANopen application. Most adjustments to the CANopen protocol can be found in the manufacturer-specific area of the object directory and within the supported error messages ("Emergency").

3.1 Network Topology

The CAN network is set up as a linear topology with terminating resistors (120 Ohm). In the CAN network, all devices are wired in parallel. This requires the CAN bus cable to be looped through without disruption.

All devices in the CAN network communicate with the same baud rate. The bus structure (disconnected from power supply) enables devices to be added and removed in a non-reactive manner; it also enables step-by-step startup of the CAN bus. Later upgrades have no effect on CANopen devices that are already in operation. If a device malfunctions or if a new device is added to the CAN bus, it is automatically detected.

The maximum number of physical devices on the CAN network depends on the input impedance of the CAN bus drivers being used. The CAN bus driver utilized in the fieldbus coupler allows a total of 120 CANopen devices to be connected.

Star repeaters enable branching in linear structures and thus the setup of hierarchal network structures. If repeaters are used, the maximum possible quantity of 127 CANopen devices can be achieved, or the network reach (bus length) can be enlarged. However, when using repeaters the internal delay of the repeaters should be taken into account when calculating the network reach. Another advantage of the repeaters is error suppression. Without repeaters, all communication in the CAN network would fail if a short circuit arose between CAN_High and CAN_Low. Using repeaters would cause the error to only have an effect on the subsegment that is disconnected by the repeaters. The rest of the network would continue to function without constraint. In addition, repeaters can also enable the implementation of subsegments with fiber optic cable. This illustrates the advantage of isolating high-frequency interference signals, through which subsegments can be conducted even in environments with electrical interference.

3.2 Physical Transmission Technology

CANopen is based on CAN, which is standardized as a communication medium in the ISO 11898 specification.

3.2.1 Wiring with Symmetrical Copper Cable

Signal transmission through symmetrical copper cable ("twisted pair") is established with a transfer rate of 125 KBit/s – 1 Mbit/s. STP ("shielded twisted pair") is to be used as a transmission medium.

For wiring with shielded copper cable (3 x 0.25 mm²), the corresponding plug with the CAN_High, CAN_Low and CAN_GND connections is to be used. CAN_High and CAN_Low are two physically different bus signal levels. CAN_GND is the collective reference potential.

The cable's shielding can be placed on the CAN_SHLD connection.

Each CANopen node forms the V_{Diff} differential voltage from the CAN_High and CAN_Low bus signal levels as follows: $V_{Diff} = V_{CAN\_High} - V_{CAN\_Low}$.
The differential signal transmission offers the benefit of insensitivity to common mode interferences and potential differences between the CANopen nodes.

If the bus signal levels are found to be recessive, a voltage of 2.5 V is present between CAN_Low and CAN_GND, as well as between CAN_High and CAN_GND. The differential voltage amounts to 0 V.

If the bus signal levels are found to be dominant, a voltage of 1.5 V is present between CAN_Low and CAN_GND, and a voltage of 3.5 V between CAN_High and CAN_GND. The differential voltage amounts to approx. 2 V.

Check to ensure correct installation of the terminating resistors by using an ohmmeter to measure resistance. Please see the following figure:

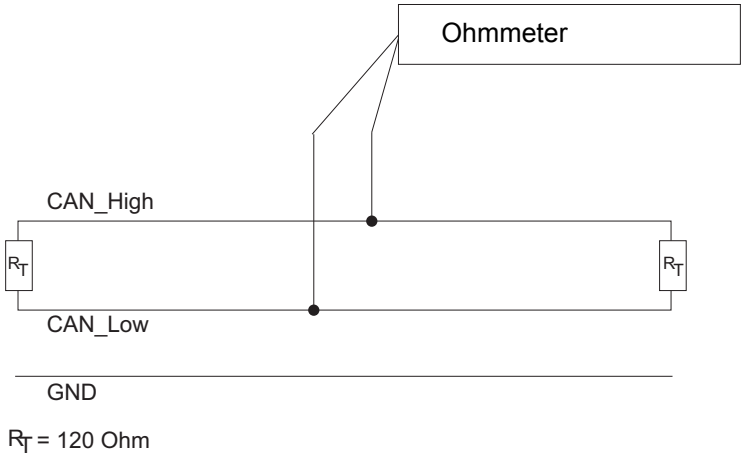


Figure 1: Measuring principle to check CAN bus before startup

Table 1: Measured values

Measurement Between:	Measured Value	Description
GND & CAN_L	Infinite	Ok
	0	Short circuit between CAN_H & CAN_L
GND & CAN_H	Infinite	Ok
	0	Short circuit between CAN_H & CAN_L
CAN_L & CAN_H	Approx. 60 Ω	Ok; two terminating resistors in CAN network
	Approx. 120 Ω	Only one terminating resistor in CAN network
	< 50 Ω	More than two terminating resistors in CAN network

Note

Additional information and documents on "CAN in Automation (CiA)" are available online at www.can-cia.de.

3.2.2 Maximum Bus Lengths in CAN Networks

The bus lengths are mainly limited by the signal propagation delay and must therefore be adjusted to the baud rate:

Table 2: Maximum bus lengths dependent on preset baud rate

Baud Rate	Bus Length
1 Mbit/s	30 m
800 KBit/s	50 m
500 KBit/s	100 m
250 KBit/s	250 m
125 KBit/s	500 m
(50 KBit/s	1000 m

3.3 Communication

In CANopen, the communication objects transmit data, trigger events, signal errors, etc. Each communication object is assigned a unique COB ID ("Communication Object Identifier") in the network.

The communication objects require these parameters as well as CANopen device parameters and data. This is all stored in an object directory. Which and how many entries in the object directory the fieldbus coupler supports is dependent on the connected I/O modules.

Communication objects exist for network management (NMT), synchronization (SYNC) or error messages (EMCY). Service data objects (SDO) enable read and write access to the object directory. Real-time data are transferred via process data objects (PDO). There are two types of PDO:

- RxPDO (Receiving PDO)
  An RxPDO is a process data object that is received by the CANopen device. It contains process data output by the device.

- TxPDO (Sending PDO)
  A TxPDO is a process data object that is sent by the CANopen device. It contains process data read by the device from its inputs.

4 Fieldbus Coupler Description

The fieldbus coupler serves to integrate I/O modules of series 767 into a CAN network. The bus driver utilized in the fieldbus coupler allows a total of 120 CANopen devices to be connected. Up to 64 I/O modules can then be connected to these devices.

The fieldbus coupler supports the following communication objects for data transmission:

- 32 TxPDO
  These transmit process data from the input modules.
- 32 RxPDO
  These transmit process data from the output modules.
- 2 Server SDO
  The SDO transfers configuration data in and out of the object directory of the fieldbus coupler.
- Synchronization object (SYNC)
  The SYNC communication object synchronizes the process data objects and sets and reads the process data from the CANopen devices.
- Emergency object (EMCY)
  This object indicates errors that have occurred on a CANopen device.
- Network management objects
  The network management objects control the CAN network. The following protocols belong to these objects: "Module Control Protocols," "Error Control Protocols," and "Bootup Protocol."

Summary of fieldbus coupler properties:

- Eight digital inputs, 24VDC
- Modular and extendable by up to 64 external I/O modules
- USB interface for service purposes, as well as configuration
- Lockable display panel for the selection of the station address (DIP switch)

Detailed information on the properties of the fieldbus coupler is available in Section 4.9.

4.1 Connections

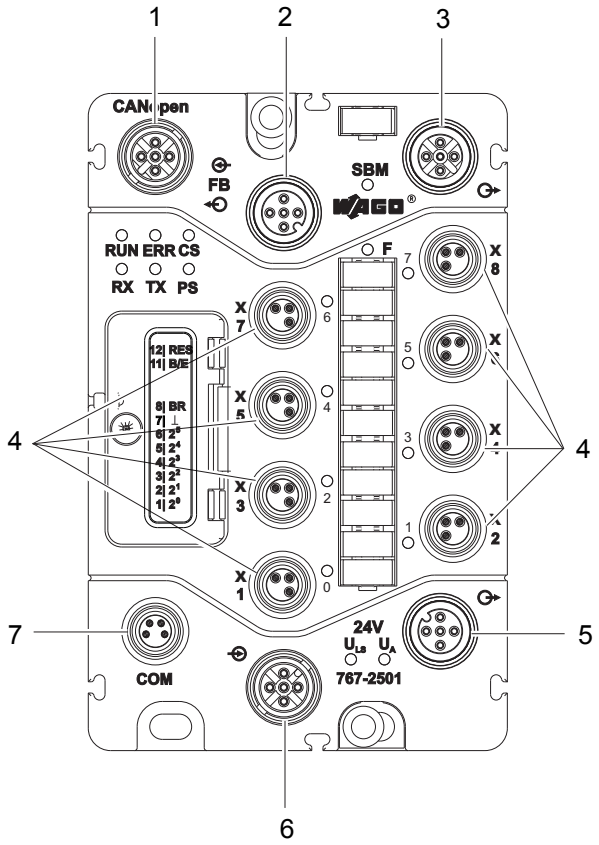


Figure 2: Identification of connections

Position	Description	Function
1	CANopen input, M12 plug, A-coded	Physical connections for the integration of the fieldbus coupler into the CAN network.
2	CANopen output M12 socket, A-coded	
3	S-BUS output M12 socket, B-coded	Physical connection for connecting I/O modules to the S-BUS and for closing the S-BUS.
4	Digital inputs 1 – 8 M8 socket	For connecting digital sensors (e.g., initiators or limit switches).
5	Supply output M12 socket, A-coded	Use of the system and/or field supply for the following I/O module.
6	Supply input M12 plug, A-coded	Infeed of system and field power supply
7	USB interface M8 socket, 4 poles; A-coded	Configuration independent of fieldbus, configuring and diagnosing entire fieldbus station and updating device software.

4.2 Marking Possibilities and Fasteners

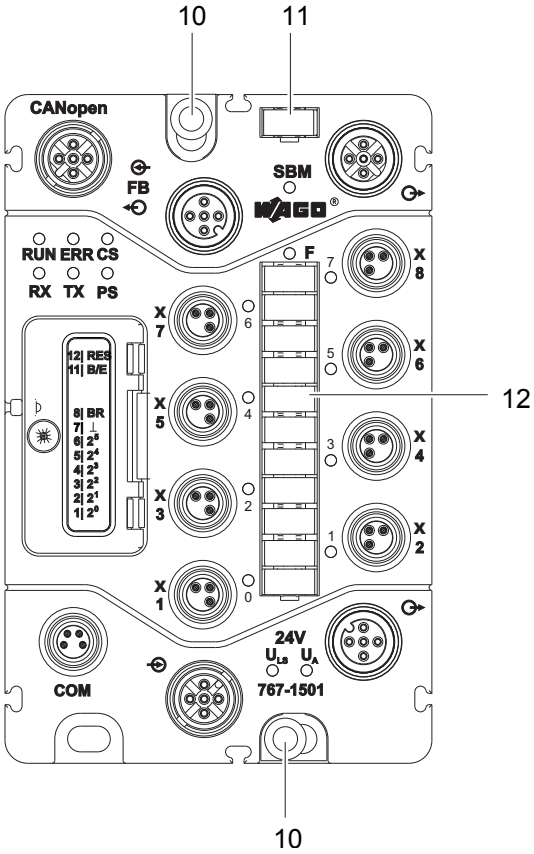


Figure 3: Identification of possibilities for marking and fastening

Position	Description	Function
10	Mounting holes for M4 screws	For fastening and grounding the fieldbus coupler.
11	Module marking plate	For identification of the fieldbus coupler within the CAN network.
12	Marker strips	For designation of digital inputs.

4.3 LEDs and Operating Elements

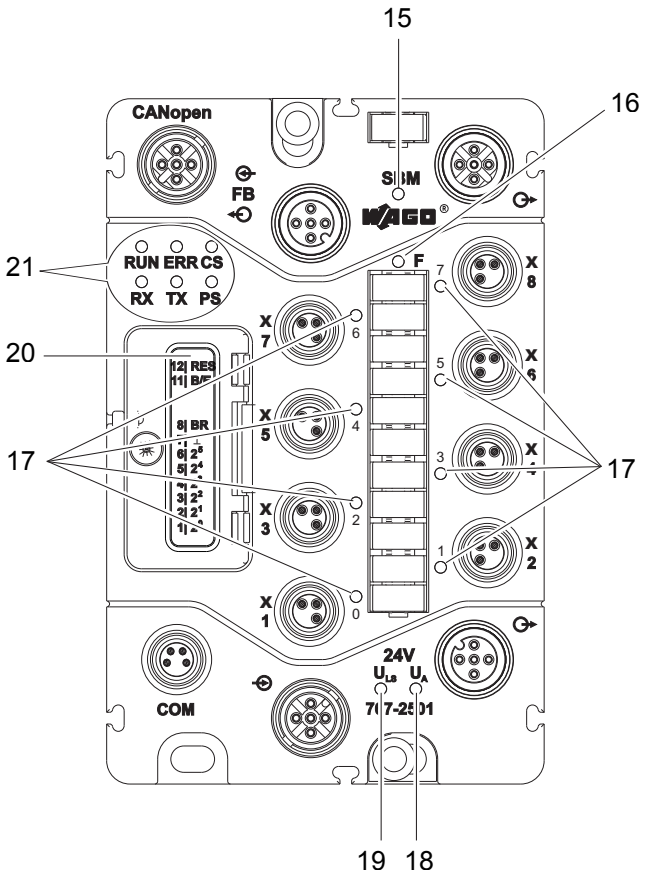


Figure 4: Identification of LEDs

Position	LED/ Operating element	Color	Meaning
15	SBM	Green, red	Status of S-BUS Master
16	F	Red	Diagnostic information of digital inputs is available.
17	LEDs 0 to 7	Yellow	Input signal pending.
18	U_A	Green	Actuator supply is present.
19	U_{LS}	Green	Logic supply and sensor supply are present.
20	Address and Operating Mode Switch (DIP Switch)	-	Sets the CANopen station address. Setting the baud rate. Executing a hardware reset.

Position	LED/ Operating element	Color	Meaning
21	CS	Red/green	Fieldbus coupler status
	RUN	Red/green	Status of CANopen application
	ERR	Red/green	
	RX	Red	
	TX	Red	
	PS	Red/green	Program status of run-time system

Detailed information regarding the LEDs can be found starting in Section 16.1.

4.4 Address and Operating Mode Switch (DIP Switch)

Use the DIP switch to set the CANopen station address of the fieldbus coupler. The fieldbus coupler is available at this address via the CAN network. Moreover, the DIP switch makes it possible to set the baud rate and execute a hardware reset of the fieldbus coupler. A transparent cover protects the DIP switch. The assignment of values is as follows:

Table 3: Default settings of DIP switch

Switch	1	2	3	4	5	6	7	8	9	10	11	12
Binary value/ Functions	2^0	2^1	2^2	2^3	2^4	2^5	2^6	Baud	Baud	Run/ Stop	Boot/ Execute	Reset
Switch Setting	Off	Off	Off	Off	Off	Off	Off	Off	Off	Off	Off	Off

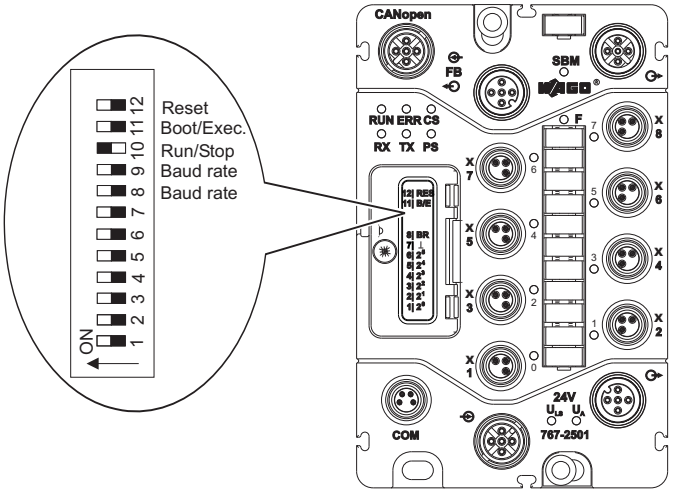


Figure 5: DIP switch. All switches are set on "Off" at delivery.

Table 4: Explanation of DIP switch

Switch	**On**	**Off**
1 … 7	Use this switch to set the CANopen station address of the fieldbus coupler. Station addresses in the range from 1 to 127 are permitted. Address 0 is reserved.	
8, 9	The fieldbus coupler's baud rate can be set via switches 8 and 9. You have the choice between 125kbit/s, 500kbit/s, 1Mbit/s and automatic baud rate detection. Detailed information on setting baud rates can be found in Section 7.2.	
10	This switch is used to specify whether the boot project is to be automatically executed when the fieldbus coupler is restarted.	
11	This DIP switch must always be set to „Off“.	
12	This switch is used to initiate a hardware reset of the fieldbus coupler. You can override this status by returning the switch to "Off."	Normal operation.

Detailed information can be found starting in Section 7.1.

4.5 Labeling and Symbols on the Back Side

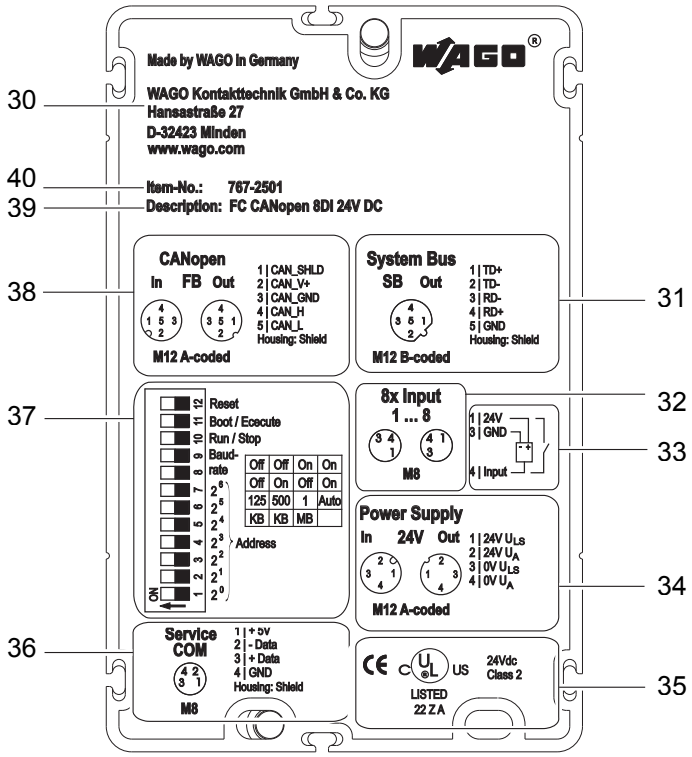


Figure 6: Marking and symbols

Position	Description
30	Manufacturer's mailing address
31	Connection assignment of the S-BUS
32	Connection assignment of digital inputs
33	Connection example
34	Connection assignment of supply input and output
35	Information on approvals and CE marks
36	Connection assignment of USB interface
37	Labeling and assignment of DIP switch
38	Connection assignment of CAN bus (input and output)
39	Unique identification of fieldbus coupler
40	Item number

4.6 Lateral Marking

On the side of the fieldbus coupler is a label, which gives information that would prove useful in the case of a complaint:

- BA: Work order number (50)
- SN: Serial number (50)
- Manufacturing number (51)

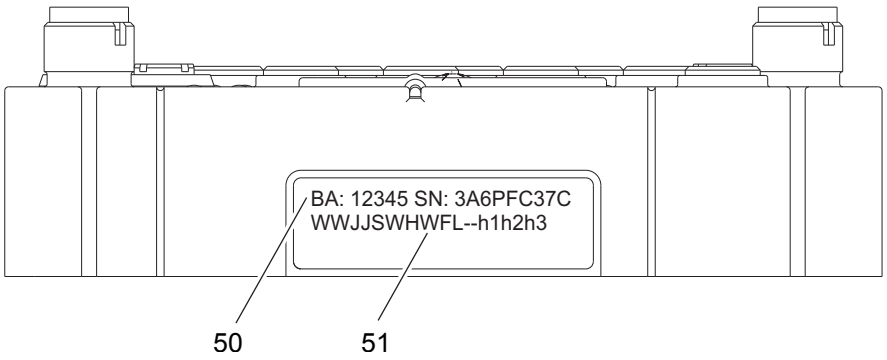


Figure 7: Label on the fieldbus coupler

Table 5: Describtion of manufacturing number

Abbreviation	Description
WW	Week of production
JJ	Year of production
FW	Firmware release index When updating the firmware, please note that the firmware release index may not be conformed to the printed firmware release index on the side of the fieldbus coupler. The “Electronic Type Label” (section 9.1) shows the actual fimware release index.
HW	Hardware release index
FL	Firmware loader release index
h1h2h3	Internal manufacturer information

4.7 Block diagram

The following schematic representation provides an overview of the power supply and principle of operation of the power supply connections, as well as the digital inputs of the fieldbus coupler (see also chapters "Connecting the Supply Cable" and "Connecting the Sensor Cable").

Please note that the common power supply of the sensors is distributed to all connections (X1 – X8, Pin 1) of the fieldbus coupler.

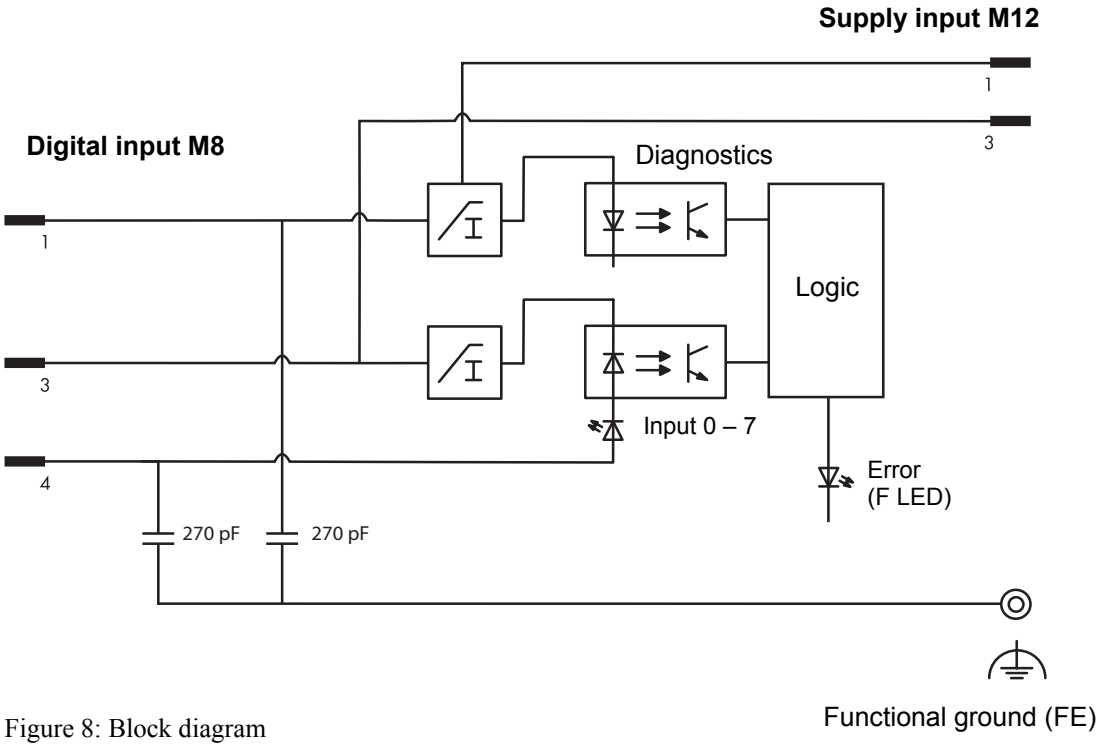


Figure 8: Block diagram

4.8 Dimensions

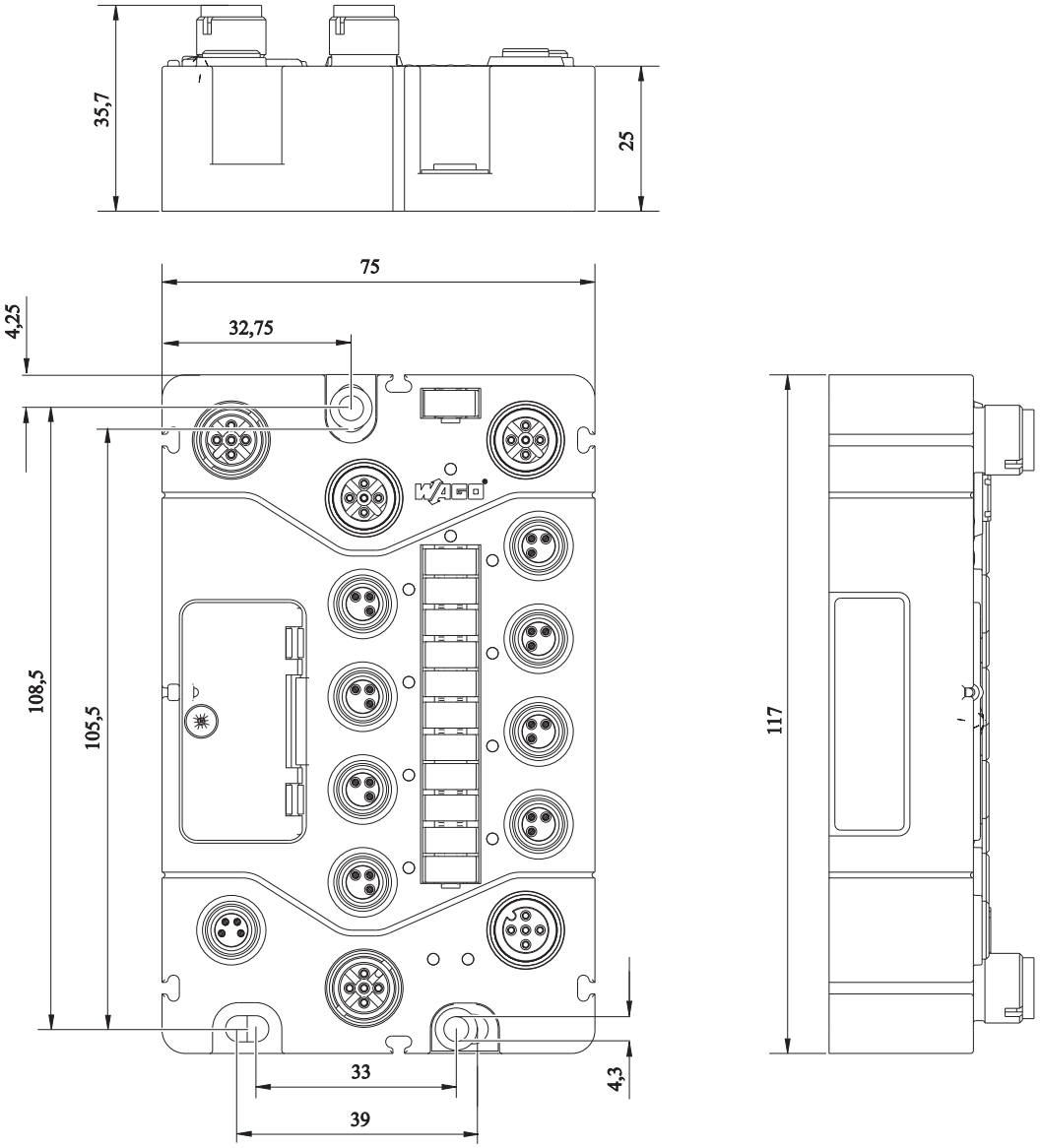


Figure 9: Dimensions of fieldbus coupler in mm

4.9 Technical Data

Table 6: Technical data

Fieldbus	
Device type	CANopen slave
Connection type	M12 connectors, A coded, 5 poles
Baud rate	125/ 500/ 1000 kBits
Transmission medium	Copper cable
Station address	1-127 (adjustable via operation panel)
Protocols	CANopen per DS 301 V4.01
Additional data	Per device profile DS 401 V2.0
Programming	
CoDeSys 3	Development system for programming and visualization according to IEC61131-3
Module supply	
Connection type	M12 connectors, A coded, 4 poles*
Current carrying capacity of the supply connections	Maximum 8A (U_{LS}: 4A, U_A: 4A)
Supply voltage Logic and sensor voltage U_{LS} Actuator voltage U_A **	 DC 24 V (-25 % … +30 %) DC 24 V (-25 % … +30 %)
Supply current Logic and sensor current I_{LS} Actuator current I_A	 Typically 85mA + sensors (max. 400mA) 5mA
Protection	Reverse voltage protection for U_{LS} + U_A Short-circuit protection for sensor supply
Digital inputs	
Number of inputs	8
Connection type	M8 connectors, 3 poles
Wire connection	2- to 3-wire
Input Filter	Parametrizable
Input characteristic	Type 1, acc. to IEC 61131-2
Signal voltage 0	DC -30V … +5V
Signal voltage 1	DC +15V … +30V
Input wiring	High-side switching
Input voltage	24VDC (-30VDC < U_{IN} < +30VDC)
Input current	Typically 2.8mA
Cable length, unshielded	≤ 30m
Wrong connection of inputs	No effect
S-BUS	
Number of expendable modules	64
Connection type	Shielded M12 connector, B coded, 5 poles

* Derating should be observed

** Also required for supply transmission

Table 6: Technical data

Isolation	
Channel - Channel	No
U_{LS}, U_A, S-BUS, fieldbus	500VDC each
Service	
Type	USB standard 1.1
Connection type	M8 connectors, 4 poles
Standards and approvals	
UL 508	
Conformity marking	CE
Configurable functions	
Fieldbus coupler	See section parameterization
Digital inputs Input filter (per channel) Inversion (per channel) Online simulation (per channel) (per module)	 0.1/ 0.5/ 3 /15 /20ms/ filter off on/off Lock/unlock; simulation value: 0/1 diagnostics
I/O diagnostics	
Per module	Short circuit/wire break of sensor supply
Per module	Undervoltage (U_{LS} + U_A)
Process image	
Input process image	512 bytes
Output process image	512 bytes
Input variables	512 bytes
Output variables	512 bytes
Program memory	1024 kbytes
Data memory	256 kbytes
Remanent memory	32 kbytes (20 Kbytes retain, 12 Kbytes flag)
Indicators	
RUN: Fieldbus coupler initialization	LED (green/red)
RX: CANopen receiver buffer	LED (red)
ERR: CANopen bus error	LED (green/red)
TX: CANopen transmit buffer	LED (red)
CS: Fieldbus coupler status	LED (green/red)
0 ... 7 : Input signal status	LED (yellow)
F: Error status	LED (red)
U_{LS} + U_A: Supply status	LED (green)
SBM: Status of S-BUS Master	LED (green/red)
Indicators	Non-storing
General information	
Dimensions (mm) W x H x L	75 x 35.7 x 117
Weight	Approx. 330 g

5 Mounting the Fieldbus Coupler

The fieldbus coupler can be fastened directly to your system using screws. It can also be mounted on a carrier rail using an adapter (WAGO accessory) or fastened to a profile rail using a surface mounting profile (WAGO accessory).

For mounting on a flat surface, WAGO offers spacers to assist in the mounting process that can be inserted between the 767 Series component. This helps by providing sufficient mounting distance for compact direct mounting, as well as eliminating gaps where dirt could accumulate. A cable tie can be fastened through each of two mounts in the spacer, which together serve to relieve strain from the sensor and actuator cables.

5.1 Information on Mounting

The following information shall always be observed:

- Disconnect the power supply from the system before you start with installation.
- The maximum diameter of the drill hole of the fieldbus coupler's mounting holes is not to exceed 4 mm. Otherwise, there may be no full contact with the fieldbus coupler's PE socket and a problem-free shielding is not possible.
- The cover cap of the DIP switch must be closed and bolted.
- To protect the fieldbus coupler from tensile forces that may arise, do not bridge spaces with it.
- Screw down the fieldbus coupler down only on flat contact surfaces to protect it from warping.
- Ensure that the connectors are not soiled during installation. Dirt and other such substances damage the contacts, allowing corrosion to develop.
- To avoid damaging the fieldbus coupler, do not mount it in shear areas of moving devices.
- Arrange for a sufficient potential equalization in your system.
- Use all mounting holes to mount the fieldbus coupler to your system so all FG (function ground) connections lie on a ground potential.

5.2 Tools and Accessories Required for Mounting

Depending on the mounting type, the following tools are required for installation:

- A screwdriver for M4 fixing screws
- Drilling machine to pre-drill the mounting holes for the fieldbus coupler to be mounted to the system and, if applicable, for the imperforated carrier rail.
- M4 thread cutter (bottoming tap or hand tap set)

The WAGO accessories listed below are required for mounting. The associated item numbers can be found in the "Accessories" section.

- Carrier rail adapter, including fixing screws and perforated or imperforated carrier rails (DIN 35 x 7.5) according to EN 60715, also available from WAGO.

  or

- Profile adapter, including fixing screws
- Spacer (optional)

Three M4x12 screws are required for direct mounting of the fieldbus coupler. The length of the screw shaft is to be selected according to the mounting type.

Bore measurements

When fastening the 767 Series component without a threaded hole, the clearance hole must not be wider than 4mm so as to ensure safe contact of the FG (functional ground) connections.

5.3 Direct Mounting on Your System

Mount the fieldbus coupler directly on a level surface of your system, without using WAGO accessories. Direct mounting of the fieldbus coupler is to be carried out as follows:

1. Disconnect the power supply from those devices on which you wish to mount the fieldbus coupler.

2. Mark the drill holes using the hole drilling template printed on the packaging. You can also hold the fieldbus coupler in the desired position and mark the drill holes. Ensure that there is sufficient space around the 767 Series component to enable you to connect all cable without problems.

Note

We recommend using WAGO spacers for compact direct mounting. If these are used, the additional distance from the second 767 component is to be noted (see Section 5.7).

3. Fasten the fieldbus coupler to the grounded frame of your system or to another grounding point with the M4x12 screws via the three mounting holes.

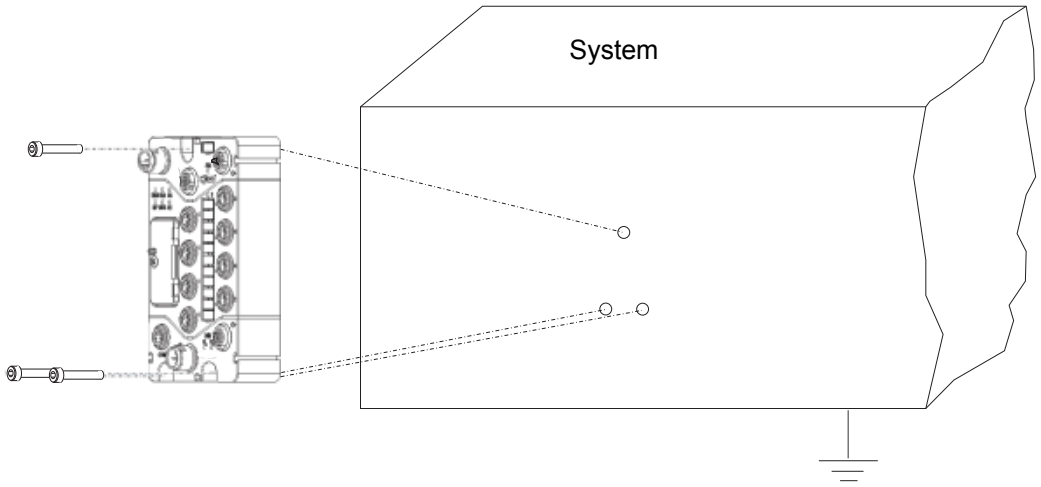


Figure 10: Mounting the fieldbus coupler on the grounded system

5.4 Mounting on a Carrier Rail (only with WAGO Accessories)

5.4.1 Fastening the Carrier Rail Adapter to the Fieldbus Coupler

A carrier rail adapter is required to mount the fieldbus coupler on carrier rails.

Screw together the fieldbus coupler and carrier rail adapter using the M4 threaded screws provided, as shown in the figure below.

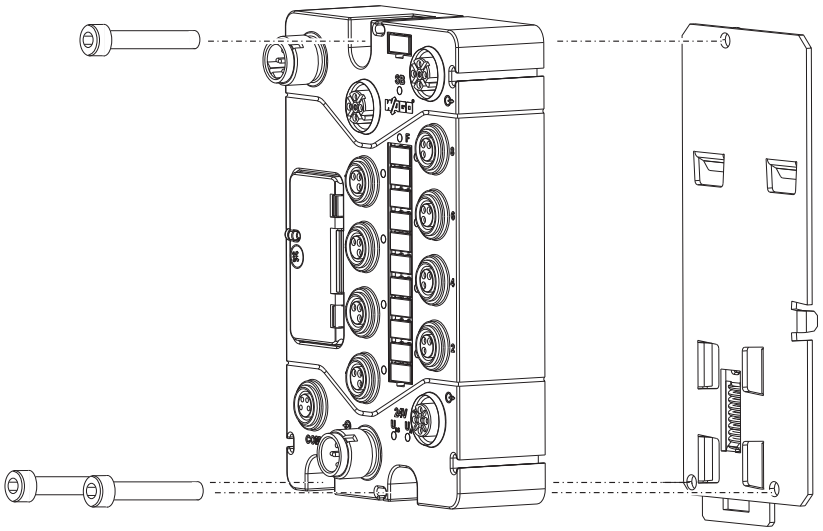


Figure 11: Fastening the fieldbus coupler to the carrier rail adapter

5.4.2 Fastening the Fieldbus Coupler with Carrier Rail Adapter to a Carrier Rail

In order to provide a clear representation, the carrier rail adapter in the figure below is shown without the fieldbus coupler.

When mounting the fieldbus coupler to a carrier rail (DIN rail 35 x 7.5) using a carrier rail adapter, proceed as follows:

1. Disconnect the power supply from those devices on which you wish to mount the fieldbus coupler.

2. Set the fieldbus coupler onto the edge of the carrier rail (61) with the two notches (60).

3. Press the undersurface against the lower carrier rail edge until the latch (62) locks in place.

Note

When mounting the rail vertically or if shock or vibration loading should occur, the use of end stops (item no.: 249-116 or 249-117) for stabilization is required. For more information, see the "Technical Data" section in the manual "WAGO SPEEDWAY 767, System Description and Notes."

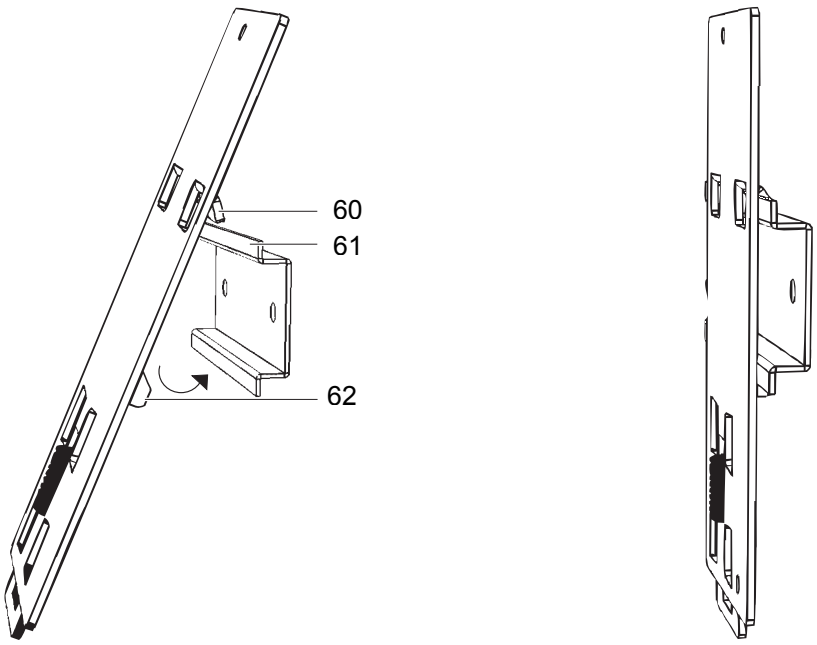


Figure 12: Mounting the carrier rail adapter

5.5 Mounting on a Profile Rail (only with WAGO Accessories)

5.5.1 Fastening the Profile Adapter to the Fieldbus Coupler

Aside from using carrier rail adapters to fasten the fieldbus coupler, you also have the option to fasten it to a profile rail using the profile adapter and nuts, provided that this mounting type is supported by your system. You are to supply the necessary nuts.

Screw together the fieldbus coupler and the profile adapter using the M4 threaded screws provided, as shown in the figure below.

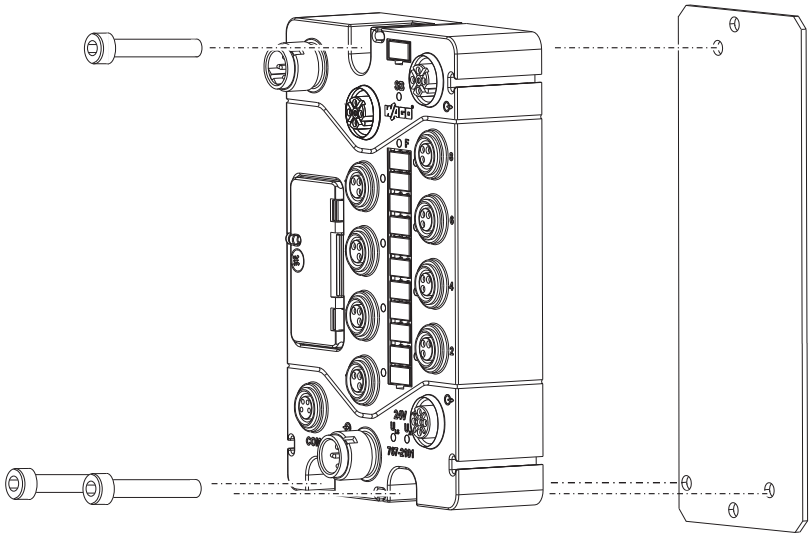


Figure 13: Fastening the fieldbus coupler to the profile adapter

5.5.2 Fastening the Fieldbus Coupler with Profile Adapter to a Profile Rail

To fasten the fieldbus coupler to a profile rail of your system, two nuts are required with one screw each (length of screw threads must be compatible with your system).

1. Disconnect the power supply from those devices on which you wish to mount the fieldbus coupler.

2. Insert the two screws into the holes above and beneath the fastened fieldbus coupler on the profile adapter.

3. Fasten an appropriate nut on each of these screws.

4. Insert the profile adapter with the attached fieldbus coupler into the profile rail of your system. Position it and tighten the screws.

5.6 Replacing the Marking Spaces

The module marker card and marking strip are attached when delivered. The protective cover is to be removed when labeling the marking strip. To do this, proceed as follows:

1. Press the slot screwdriver (maximum slot width: 3mm) into the small opening under the marking strip cover (12) and lever it up.
2. Remove the marking strip cover.
3. Mark the marking strip with a waterproof pen.
4. Reinsert the marking strip cover and press it firmly in place.

If the marker card (11) must be replaced, proceed in accordance with the step sequence described previously. New module marker cards can be obtained through WAGO.

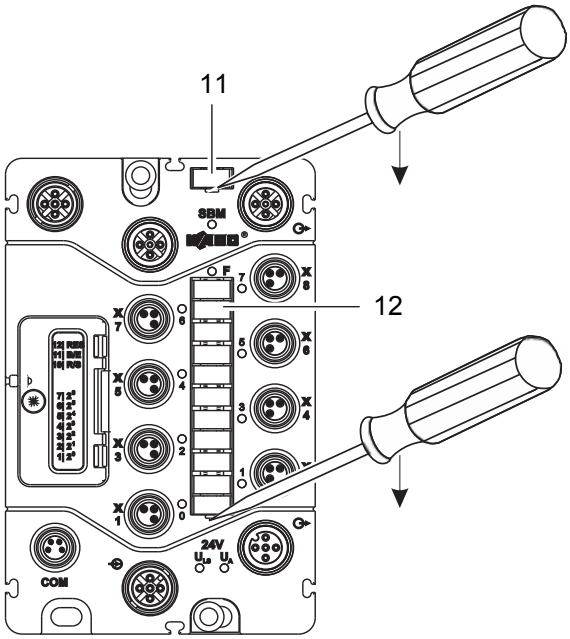


Figure 14: Replacing the marking spaces

5.7 Mounting the Spacer in the Case of Compact Arrangement

By using the spacer, a sufficient mounting distance can be achieved when directly mounting the 767 Series components, and gaps can be eliminated where otherwise dirt and other substances could accumulate. In addition, it is possible to optimize the cable routing of the sensors and actuators. For this purpose, two fastening lugs each are included on the spacer for cable ties.

1. Disconnect the power supply from those devices on which you wish to mount the fieldbus coupler.

2. The spacer can only be inserted into the appropriate openings of the fieldbus coupler from the bottom. To bind both components, place the fieldbus coupler on the spacer or push the spacer from the bottom into the coupler.

3. Fasten the components on a flat surface by fastening the fieldbus coupler to the grounded frame of your system or to another grounding point with three M4 screws via the mounting holes.

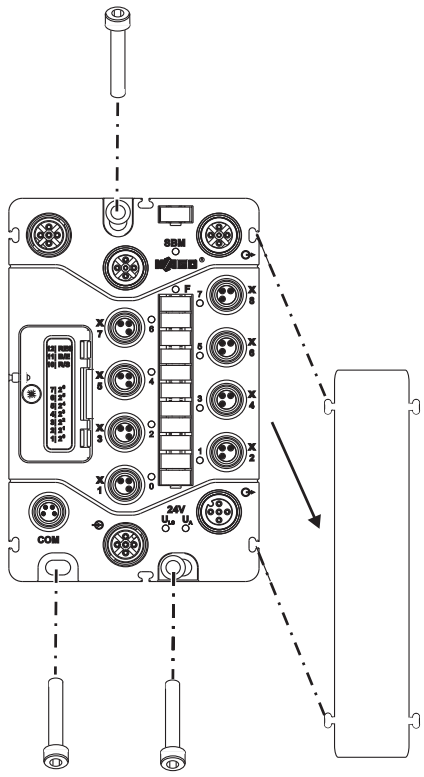


Figure 15: Attaching a spacer

4. When attaching 767 Series component, only one 767 component connected with a spacer can be attached and screwed on to the preceding component due to the mounting direction. The last 767 component is fastened without a spacer.

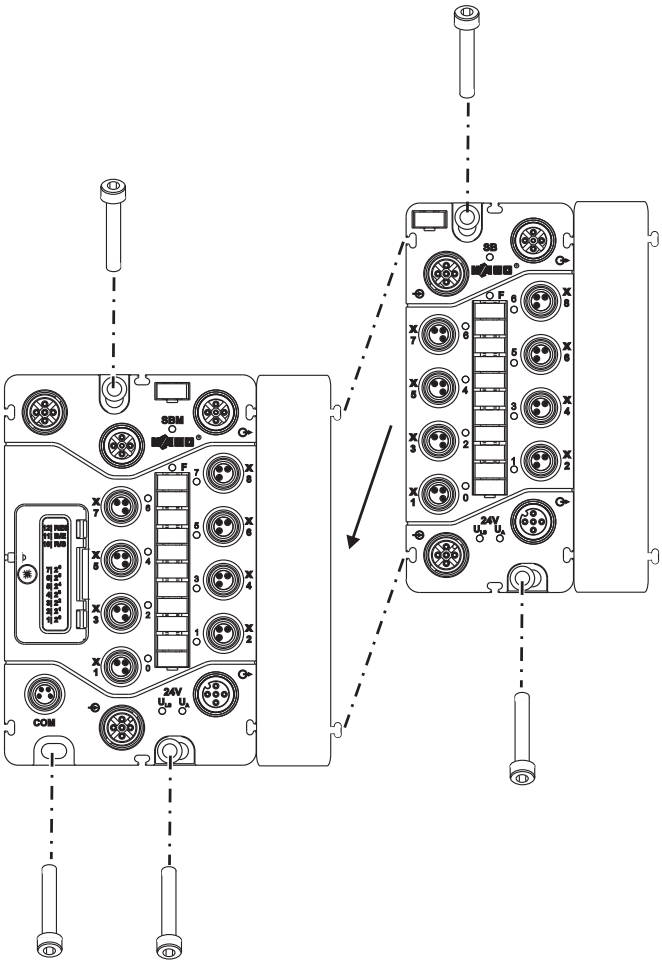


Figure 16: Attaching another 767 component to the fieldbus coupler

6 Connection to the Data and Supply Cable

6.1 Notes

DANGER

Electric voltage!
Operate the 767 Series components exclusively with 24VDC PELV (Protective Extra Low Voltage) or SELV (Safety Extra Low Voltage) voltage sources. Failure to comply may result in electric shock.

NOTICE

The highest current carrying capacity of the supply contacts is 4A!
Always observe the maximum current carrying capacity per supply line (U_{LS}, U_A) for each 767 component and the overall power consumption for all 767 Series component. Neither of these values shall exceed 4A since an increase in current causes the contacts to overheat and damages the 767 Series components. Information regarding the power demand of each 767 component can be found in the corresponding data sheet, which is available at www.wago.com.

NOTICE

Exposed connections!
If connections have not been closed with protective caps, liquid or dirt can penetrate the module and ruin it. Close all unused connections with protective caps to ensure that IP67 degree of protection is provided.

- The connectors must be disconnected from the power supply when screws are tightened.
- Tighten the connectors by hand only. Using mechanical tools can cause the threads to strip. In such a case, the fieldbus coupler must be replaced.

  Tightening torques for the connectors: 0.6 Nm
- Observe the exact positioning (coding) between plug and socket.
- For both power supply and S-BUS, use only pre-assembled WAGO system cables so the specified characteristics of the technical data can be achieved.
- Keep all cables a sufficient distance away from electromagnetic sources of interference in order to maintain a high level of interference resistance of the 767 system against electromagnetic emissions.
- Observe the minimum bending radiuses of the WAGO system cable. For more information, see the technical data at www.wago.com.

- When laying all cable, ensure that you do not lay it in shear areas of moving machine parts.
- The cable shield of the fieldbus cable must be connected with functional ground on both ends over a wide area.
- Observe the correct layout of the potential equalization.
- Do not use drop lines under any circumstances. This can lead to amplified line reflections and signal distortions, which greatly impair the transmission quality.
- Check for proper installation of the CAN line terminators (if applicable).

6.2 Required Accessories

The WAGO accessories listed below are required for connecting the data and supply cable. The associated item numbers can be found in the "Accessories" section.

- M12 CAN terminator in the IP 67 safety class
- S-BUS M12 terminator, IP 67
- S-BUS and supply cables, pre-assembled on both ends, IP 67
- USB cable pre-assembled on both ends, IP 67

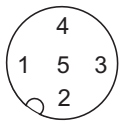
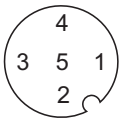
6.3 Connecting the CAN Cable

CANopen assists in the communication between a higher-level controller and the fieldbus coupler.

If you are not using a pre-assembled CAN cable, a shielded M12 socket with the IP 67 degree of protection is to be connected to this.

The following table outlines the assignment of the CANopen connections.

Table 7: CANopen connection assignment

Connection		Contact	Description
IN 	OUT 1 		CAN_SHILD
		2	CAN_V+
		3	CAN_GND
		4	CAN_H
		5	CAN_L

6.3.1 Connecting the Fieldbus Coupler to a CAN Network

To connect the fieldbus coupler to the CAN network, proceed as follows:

1. Disconnect the power supply from those devices on which you have mounted the fieldbus coupler.
2. Connect the fieldbus coupler with the CAN network by plugging the socket of the CAN cable (W) into the IN connection ⊖ (1).
3. Tighten the socket using the knurled-head screw.
4. Connect the CAN terminator to the OUT connection ⟲ (2) and tighten it.

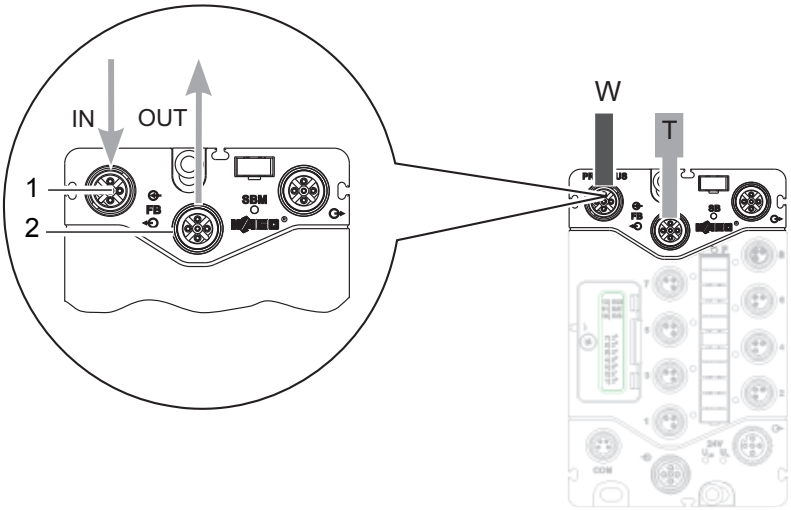


Figure 17: CAN connected to a fieldbus coupler

6.3.2 Connecting Several Fieldbus Couplers Within a CAN Network

Up to 120 fieldbus couplerss can be connected to the CAN network on the line topology. To connect several fieldbus couplers, proceed as follows:

1. Disconnect the power supply from those devices on which you have mounted the fieldbus coupler.
2. Connect the first fieldbus coupler with the CAN network by plugging the socket of the CAN cable (W) into the IN connection ⊖ (1).
3. Tighten the socket using the knurled-head screw.
4. To connect two fieldbus couplers together, attach the CAN cable (W1) to the OUT connection ⊖ (2) of the first fieldbus coupler and the IN connection ⊖ (1) of the second, as shown in the figure below.
5. Tighten the plugs and sockets using the knurled-head screw.
6. Attach the CAN terminator (T) to the OUT connection ⊖ (2) of the last fieldbus coupler, as shown in the image, and tighten it.

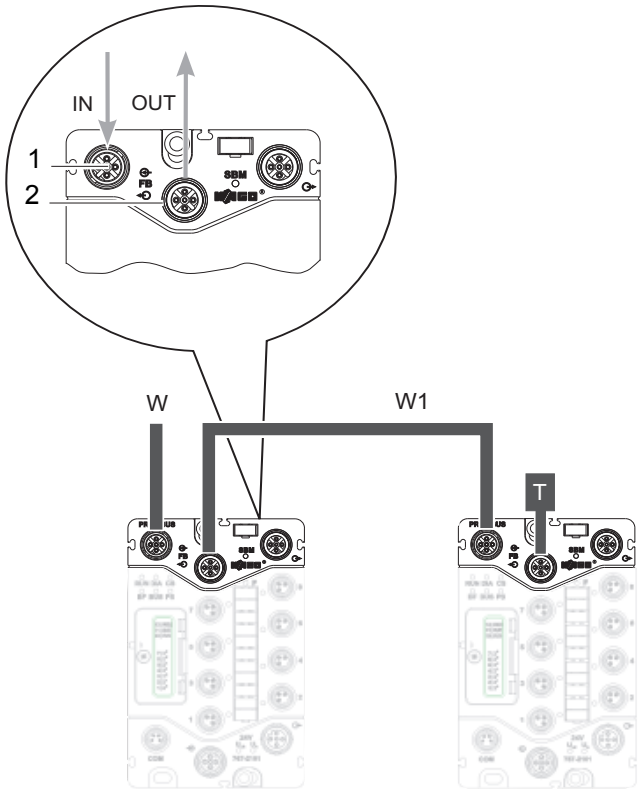


Figure 18: Two fieldbus couplers within a CAN network

6.4 Connecting the S-BUS Cable

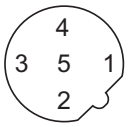
The S-BUS is used for communication between the fieldbus coupler and the connected I/O modules.

Requirement:

- A WAGO S-BUS cable pre-assembled on both ends is readily available to you. This is necessary for optimal signal transmission.
- The S-BUS terminator is available to you, which is necessary for communication.

The following table outlines the assignment of the S-BUS connection.

Table 8: S-BUS connection assignment

Connection	Contact	Description
OUT 	1	TD +
	2	TD -
	3	RD -
	4	RD +
	5	0VDC
	Connecting thread	Shielded

To connect the S-BUS cable to fieldbus couplers and I/O modules, proceed as follows:

1. Disconnect the power supply from those devices on which you have mounted the fieldbus coupler.

2. Connect the S-BUS cable (S1) with the OUT connection ⟳ (3) of the fieldbus coupler and the IN connection ⟲ (8) of the subsequent I/O module. For example, if two I/O modules have been connected to the fieldbus coupler, connect the S-BUS cables (S1, S2) to the associated IN and OUT connections, as shown in the figure below.

3. Tighten the plugs and sockets using the knurled-head screws.

4. Attach the S-BUS terminator (T) to the last I/O module as shown in the figure and tighten it.

If you do not connect any I/O modules to the S-BUS, screw the S-BUS terminator (T) onto the S-BUS OUT output ⟳ (3) and the CAN terminator (T1) onto the OUT output ⟲ (2) of the fieldbus coupler.

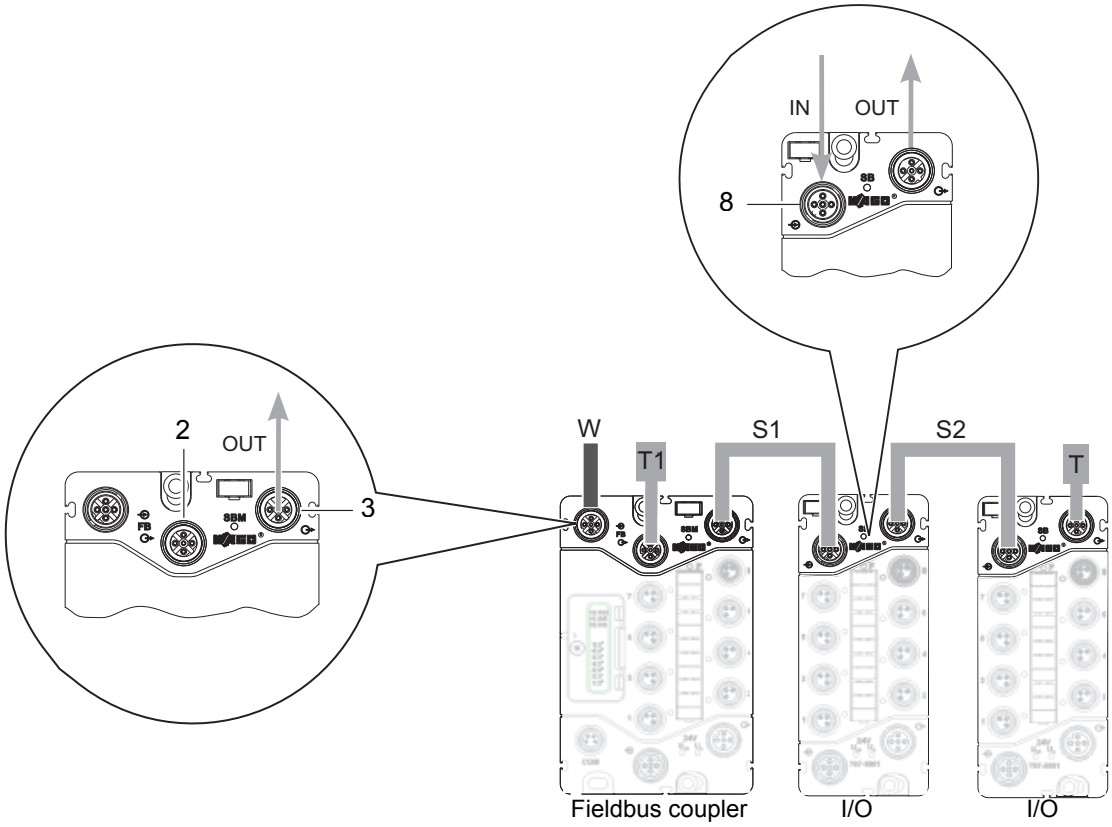


Figure 19: S-BUS is connected to one fieldbus coupler and two I/O modules

6.5 Connecting the Supply Cable

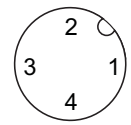
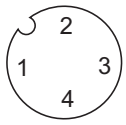
The supply cable provides power to the fieldbus coupler and the connected I/O modules.

Requirement:

- A supply cable for +24VDC and 0VDC must be connected to the supply input (6 in the figure on the next page) via appropriate fuses.
- The WAGO supply cable pre-assembled on both ends must be available (K1 and K2 in the figure on the next page).

The following table outlines the assignment of the supply connections:

Table 9: Supply connection assignment

Connection		Contact	Description
IN 	OUT 	1	24VDC U_{LS}
		2	24VDC U_A
		3	0V U_{LS}
		4	0V U_A

NOTICE

The highest current carrying capacity of the supply contacts is 4 A!
Always observe the maximum current carrying capacity per supply line (U_{LS}, U_A) for each 767 component and the overall power consumption for all 767 components. Neither of these values shall exceed 4A since an increase in current causes the contacts to overheat and damages the 767 components. Information regarding the power demand of each 767 component can be found in the corresponding data sheet, which can be downloaded at www.wago.com.

To connect the supply cable to the fieldbus coupler and I/O modules, proceed as follows:

1. Disconnect the power supply from those devices on which you have mounted the fieldbus coupler.

2. Connect the power supply cable (K0) with the fieldbus coupler by plugging the socket in the IN connection ⊖ (6) of the fieldbus coupler.

3. Tighten the socket using the knurled-head screw.

4. Connect the power supply transmission cable (K1) with the OUT connection ⊖ (5) of the fieldbus coupler and the IN connection ⊖ (8) of the subsequent I/O module.
   For example, if two I/O modules have been connected to the fieldbus coupler, connect the power supply transmission cables (K1, K2) to the associated IN and OUT connections as shown in the figure below.

5. Tighten the plugs and sockets using the knurled-head screw.

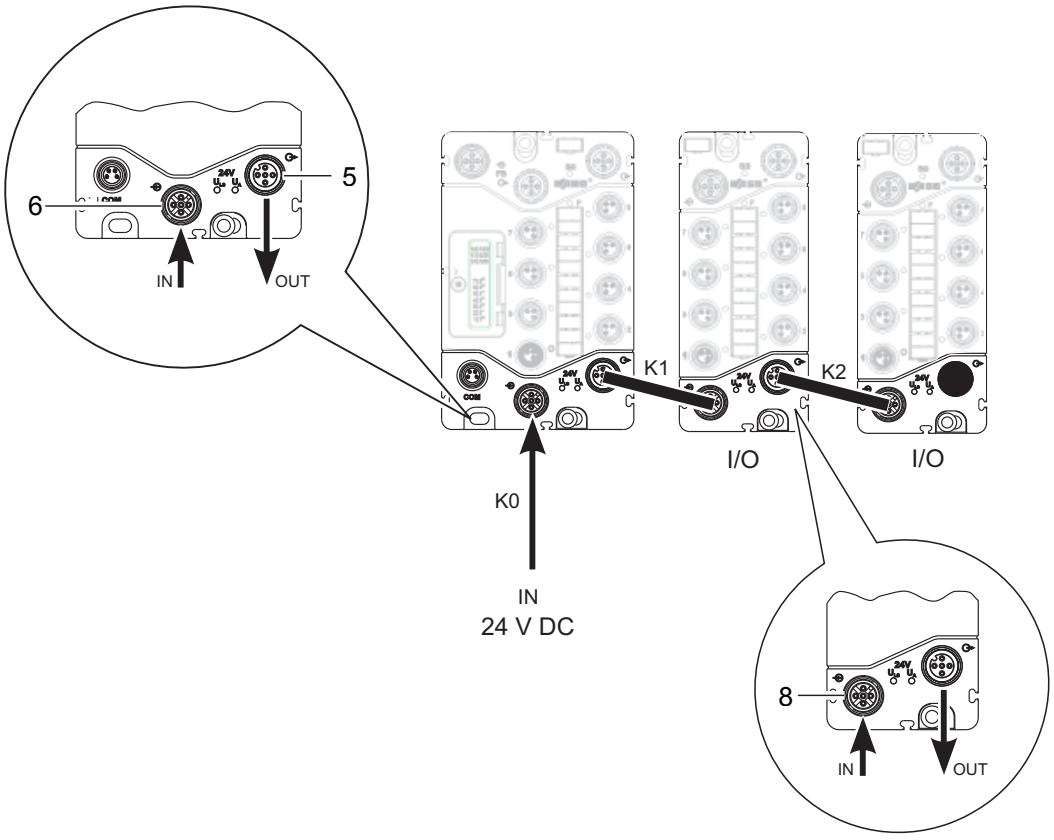


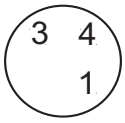
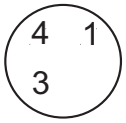
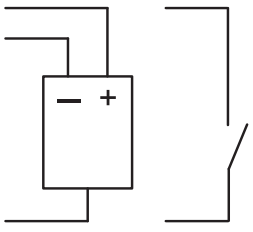
Figure 20: Supply cable connected with fieldbus coupler and I/O modules

6.6 Connecting the Sensor Cable

The sensor cables provide power to the connected sensors and transmit the sensor signals.

When using cables that have not been pre-assembled, make sure that these cables are equipped with M8 plugs rated to IP67. The following table outlines the assignment of the sensor connections:

Table 10: Connection assignment of digital inputs

Connection		Connection diagramm
IN	IN	1: 24 V 3: 0 V U_{LS}
 X1, X3, X5, X7	 X2, X4, X6, X8	 4: Input

NOTICE

The highest current carrying capacity of the supply contacts is 4A!
Ensure that the sensors from the U_{LS} supply line are supplied with power. The sensor's power consumption is to be taken into consideration when determining the present power demand for the U_{LS} supply line.

NOTICE

The total maximum power consumption of the sensors must not exceed 400mA (50mA/channel).
Please note that the combined power consumption of all connected sensors is not to exceed 400mA. The distribution of power among the existing connections is depending on the individual power requirements of the sensors.

To connect the sensors to the digital inputs (X1 – X8), proceed as follows:

1. Disconnect the power supply from those devices on which you have mounted the fieldbus coupler.

2. Insert the sensor cable plug into a digital input socket (4) of the fieldbus coupler, and tighten it via knurled-head screw.

3. Screw a protective cap on all unused ports to ensure that IP67 degree of protection is provided.

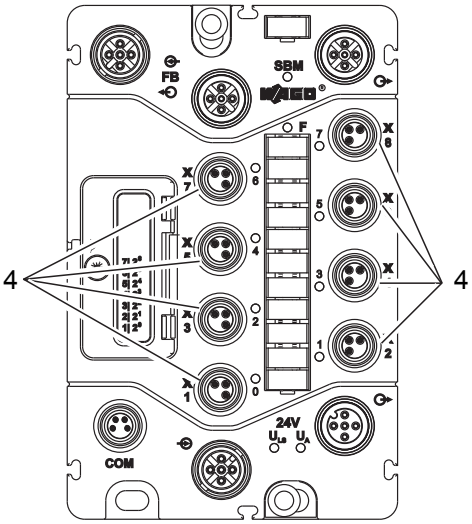


Figure 21: Digital inputs

6.7 Connecting the USB Cable

The fieldbus coupler's USB connection can be used to either configure the fieldbus coupler with an FDT/DTM frame application or to use general service functions. To use the USB connection, the USB driver "WAGO_767_USB_DRIVER_Setup" (item no.: 759-922) must be installed on your computer. This driver can be obtained on the Internet at www.wago.com. The USB driver is also included on the CD-ROMs "WAGOframe" (item no.: 759-370) and „CoDeSys 3“ (item no.: 759-915).

After installing the WAGO USB driver, the following COM ports are supplied by the USB connection:

- **I/O-Service**
  This COM port is needed to configure the fieldbus coupler using WAGOframe.

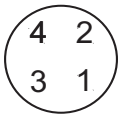
- **I/O-Programming**
  This COM port is required for serial communication between the CoDeSys 3 development environment and the fieldbus coupler.

Note

When using the USB connection, a faulty USB data transmission or an outage can cause communication disruptions. This can be due to additional or inappropriate USB distributors (hubs) or to USB cables that are too long (max. 5 m) or inappropriate. One should therefore refrain from using additional devices if possible.

When using a USB cable that has not been pre-assembled, M8 plugs with the IP 67 degree of protection are to be connected. The following table outlines the assignment of the USB connection:

Table 11: USB interface assignment

Connection	Contact	Description
IN 	1	+ 5V
	2	- Data
	3	+ Data
	4	0VDC
	Connecting thread	Shielded

1. Connect your PC with the fieldbus coupler by inserting the USB cable plug into the fieldbus coupler's USB connection COM (7) and tightening it.

2. Switch on the fieldbus coupler. Please refer to Section 7.2 for more information. After the fieldbus coupler has been switched on, your operating system detects a new device and installs the USB driver.

Note

During installation of the driver, an MS Windows message may appear several times informing you that the Windows Logo Test failed. Ignore this message and click **[Continue with installation]**.

3. When the USB connection is not in use, screw a protective cap on it to comply with the IP 67 degree of protection.

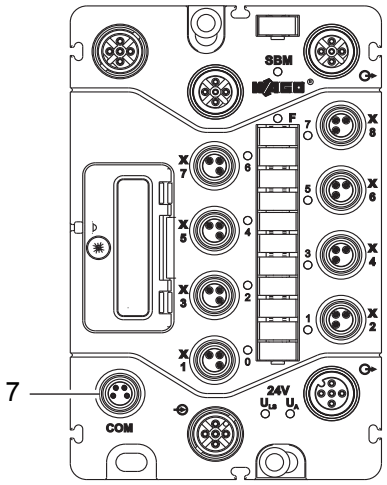


Figure 22: USB interface

7 Start-Up

NOTICE

Exposed connections!
If connections have not been closed with protective caps, liquid or dirt can penetrate the module and ruin it. Close all unused connections with protective caps to ensure that IP67 degree of protection is provided.

Before starting up the 767 node, ensure that the following requirements are met:

- The fieldbus coupler is mounted properly (see Section 5).
- All necessary supply and sensor lines, as well as the S-BUS and CAN cable, are securely fastened onto the appropriate connections (see Section 6).
- The CAN and S-BUS terminators are fastened (see Section 6).
- The power supply of the higher-level controller is switched on (see associated manual).
- An appropriate potential equalization is implemented in your system.
- The shielding is carried out properly.

7.1 Setting the CANopen Station Address

To integrate the fieldbus coupler in the CAN network, the fieldbus coupler must be assigned an IP address that is unique across the network using DIP switches. The address can be any number between 1 and 127. Use switches 1 – 7 to set the address.

Table 12: Default settings of DIP switch

Switch	1	2	3	4	5	6	7	8	9	10	11	12
Binary value/ Functions	2^0 (1)	2^1 (2)	2^2 (4)	2^3 (8)	2^4 (16)	2^5 (32)	2^6 (64)	Baud	Baud	Run/ Stop	Boot/ Execute	Reset
Switch Setting	Off	Off	Off	Off	Off	Off	Off	Off	Off	Off	Off	Off

Requirement:
The fieldbus coupler shall not be connected to the power supply.

To set the station address via the DIP switches, proceed as follows:

1. Open the cover by unscrewing the M3 screw with a screwdriver.
2. Set the CANopen station address by switching the switches 1 – 7 on ("On") or off ("Off") accordingly. With the seven available switches, you can set addresses from 1 – 127. Address 0 is reserved.
3. Close the cover and screw it securely in place to maintain degree of protection IP 67.

Note

The CANopen station address is only active when the fieldbus coupler is connected to the power supply.

In the following example, switches 3, 5 and 6 are switched on. Thus, the CANopen station address of the fieldbus coupler in this example is 52 ($2^2 + 2^4 + 2^5 = 4 + 16 + 32 = 52$).

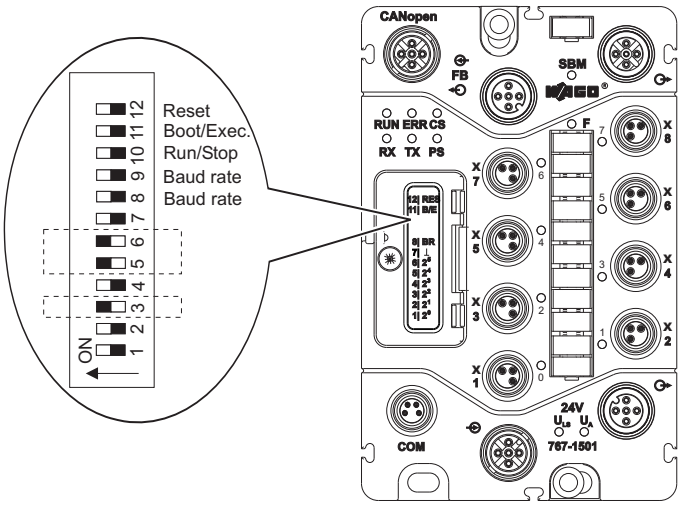


Figure 23: DIP switch set for station address 52

7.2 Baud Rate Setting

The fieldbus coupler's baud rate can be set via switches 8 and 9. You may choose between 125 KBit/s, 500 Kbit/s, 1 Mbit/s and automatic baud rate detection. The assignment of the switch position to the baud rate is shown in the table below.

Table 13: Assignment of switch position to baud rate

Switch	Switch Setting			
8	Off	On	Off	On
9	Off	Off	On	On
Baud rate	125 Kbit/s	500 Kbit/s	1 Mbit/s	Automatic baud rate detection

Automatic baud rate detection for the fieldbus coupler detects only the following baud rates: 1 MBit/s, 800 Kbit/s, 500 Kbit/s, 250 Kbit/s, 125 Kbit/s, 100 Kbit/s, 50 Kbit/s, 20 Kbit/s and 10 Kbit/s.

If automatic baud rate detection is turned on, the ERR LED flickers.

To use automatic baud rate detection, the CAN bus must be shared with a second device that uses one of the above baud rates. Depending on which baud rate is used, approximately 3 to 20 telegrams are necessary to detect the baud rate. When the fieldbus coupler discovers a valid baud rate, it sends the Bootup telegram. In isolated cases, the fieldbus coupler will not detect a baud rate. This occurs when the second CANopen device receives no acknowledgement of the telegram it has sent and shuts off its CAN controller. In this case, a third CANopen device with a firmly set baud rate is required on the CAN bus.

Note

The baud rate is only active when the fieldbus coupler is connected to the power supply.

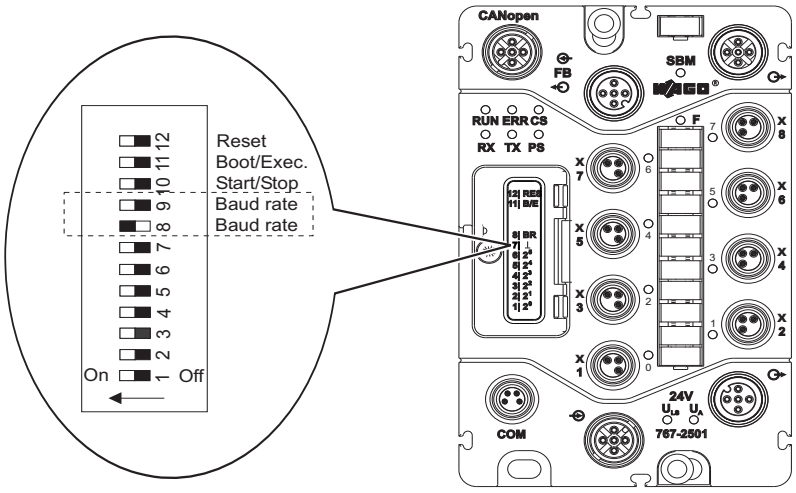


Figure 24: DIP switches set for a 500kbit/s baud rated

7.3 Switching On the Fieldbus Coupler

The fieldbus coupler starts running when the power supply is switched on. The fieldbus coupler and all connected I/O modules are ready for operation after the initialization phase. The fieldbus coupler uses the green, blinking RUN LED to signal operational readiness. If the fieldbus coupler is not connected to the CAN network, the ERR LED flashes red briefly.

Note

No 767 components are to be added or removed during operation, as this causes a disruption in the 767 node.

8 Configuration

This section contains all information required for configuration of the 767 components for CANopen.

The process data objects (PDO) are required in order to exchange process data between the higher-level controller and the fieldbus coupler. Using these assembly instances, you can write and read the inputs and outputs of the 767 components. The service data object (SDO) is necessary to determine which process data the PDO should transfer. Use this to configure the PDO.

Each PDO consists of a CAN telegram with a maximum of 8 bytes of process data and a COB ID ("Communication Object Identifier") that is unique in the network. Data transfer is carried out via these communication objects. The communication objects require these parameters and data. This is all stored in an object directory.

The object directory of a CANopen device consists of 16-bit indices and 8-bit sub-indices. All data from a CANopen device is stored here, as well as its communication parameters and mapping information.

8.1 Process Data

After starting up the fieldbus coupler, it automatically identifies all connected I/O modules. The fieldbus coupler uses this to create a local process image (input and output data range). For each I/O module, one index is created for the input data (index 0x5300 to 0x5340) and one for the output data (index 0x5400 to 0x5440). The process data of the I/O module is stored in the indices in a byte-by-byte manner.

In addition, the process data of the I/O module is stored in indices 0x6000 through 0x67FE per device profile DS401.

8.2 Fieldbus Variables

The CANopen fieldbus variables are available for data exchange between the control program and CANopen. This involves two memory areas of 512 bytes each. One memory area contains the fieldbus input variables and can be reached via CANopen objects 0xA000 through 0xA440. These objects can only be read by CANopen and mapped in TxPDOs. The PLC program can be utilized to gain write access to this area.
The other memory area contains fieldbus output variables and can be reached via CANopen objects 0xA480 through 0xA8C0. CANopen can be used to read and write these objects and to map them in RxPDOs. The PLC program can be used to gain read access to this area.

The variables entered into the object directory are separated by data type (Integer8, Unsigned8, Boolean, Integer16, etc.) and by input/output.

Because CANopen does not transfer data by bits, the variable data is combined from a Boolean data type to bytes and assigned to the corresponding index; Boolean input variable data is assigned to index 0xA080, Boolean output variable data to index 0xA500.
Variable data that has a data width of 1 byte or more is assigned to the corresponding indices in an analog manner.

Note

The IEC-61131-3 input variables are defined from the standpoint of the CAN fieldbus. In other words, the IEC-61131-3 input variables are classified as output variables from the standpoint of the fieldbus coupler. Accordingly, the IEC-61131-3 output variables are classified as input variables for the fieldbus coupler.

The following table provides an overview of the indices of the IEC-61131-3 variables:

Table 14: Indexing the IEC-61131-3 variable data in the object directory

Data Type	Output Variables of IEC 61131-3	Input variables of IEC 61131-3
	Index	
Integer 8	0xA000	0xA480
Unsigned8	0xA040	0xA4C0
Boolean	0xA080	0xA500
Integer16	0xA0C0	0xA540
Unsigned16	0xA100	0xA580
Integer24	0xA140	0xA5C0
Unsigned24	0xA180	0xA600
Integer32	0xA1C0	0xA640
Unsigned32	0xA200	0xA680
Float32	0xA240	0xA6C0
Unsigned40	0xA280	0xA700

Table 14: Indexing the IEC-61131-3 variable data in the object directory

Data Type	Output Variables of IEC 61131-3	Input variables of IEC 61131-3
	Index	
Integer40	0xA2C0	0xA740
Unsigned48	0xA300	0xA780
Integer48	0xA340	0xA7C0
Unsigned56	0xA380	0xA800
Integer56	0xA3C0	0xA840
Integer64	0xA400	0xA880
Unsigned64	0xA440	0xA8C0

Each index has a maximum of 256 sub-indices (sub-index 0-255).

The number of data entries is given in sub-index 0, and the data is stored in blocks in the subsequent sub-indices.
The size of the blocks depends on the data width of the associated data type.

Table 15: Sub-indexing the IEC 61131-3 variable data in the object directory

Sub-Index	Table of Contents
0	Number of data blocks
1	First data block with data width of corresponding data type
2	Second data block with data width of corresponding data type
...	...

Note

A detailed description of setting the default configuration can be found in Section 11.

Example:
The following IEC-61131-3 variables have been defined:
1. 11 Boolean input variables
2. 5 Integer24 output variables.

Table 16: Indexing the data from the 11 Boolean input variables

Index	Sub-Index	Content	Description
0xA080	0	2	Number of Boolean 8-bit input blocks
	1	D8 D7 D6 D5 D4 D3 D2 D1[1]	1st Boolean input block
	2	0 0 0 0 0 D11 D10 D9[1]	2nd Boolean input block

*) D1 = data bit input variable 1, D2 = data bit input variable 2, etc.

Table 17: Indexing the data from the 5 Integer24 output variables

Index	Sub-Index	Content	Description
0xA5C0	0	5	Number of 3-byte output blocks
	1	D1[2]	1st output block
	2	D2[2]	2nd output block
	3	D3[2]	3rd output block
	4	D4[2]	4th output block
	5	D5[2]	5th output block

D1 = 3 bytes of data from output variable 1, D2 = 3 bytes of data from output variable 2, etc.

[1] D1 = data bit input variable 1, D2 = data bit input variable 2, etc.
[2] D1 = 3 bytes of data from output variable 1, D2 = 3 bytes of data from output variable 2, etc.

8.2.1 Accessing the Fieldbus Variables from the Fieldbus

From the fieldbus, the data in the fieldbus coupler memory can be accessed in a byte-by-byte manner through the respective indices for data types with a data width of 1 byte (Integer8, Unsigned8 and Boolean). The sub-index is utilized to select a specific byte.

In contrast, when the indices for larger data blocks are used, several bytes can be accessed simultaneously. For example, the described fieldbus output variable data can be accessed in a word-by-word manner using the index for Integer16 (0xA0C0) or for Unsigned16 (0xA100), 3 bytes can be accessed using index 0xA140 for Integer24, etc.

Example:
The fieldbus output data bytes 0, 1 and 2 are accessed by the fieldbus coupler with the Integer/Unsigned data type:

Table 18: Access with Integer data type, such as Unsigned

Access	Fieldbus Output Data	Reading with Index (Integer/Unsigned)	Sub-Index
Byte by byte (with Integer8/Unsigned8)	Byte 0: Byte 1: Byte 2:	(0xA000/0xA040) (0xA000/0xA040) (0xA000/0xA040)	1 2 3
Word by word (with Integer16/Unsigned16)	Word 0 (Byte 0/1): Word 1 (Byte 2/3):	(0xA0C0/0xA100) (0xA0C0/0xA100)	1 2
3 bytes (with Integer24/Unsigned24)	Bytes 0 - 2:	(0xA140/0xA180)	1

The following tables give an overview of addressing data with different data widths. In this case, the corresponding indexing is assigned to the memory space for the variable data in the fieldbus coupler (bytes 0 through bytes 1), depending on the data width. The indexing indicated in the tables continues up to the respective maximum index and sub-index.

Note

The fieldbus output variables are defined from the standpoint of the fieldbus coupler. In other words, this involves input variables from the standpoint of the fieldbus. Accordingly, the fieldbus input variables for IEC-61131-3 access are classified as output variables from the fieldbus:
IEC-61131-3 input variable = fieldbus output variable
fieldbus input variable = IEC 61131-3 output variable

8.2.1.1 Access Addressing of Fieldbus Output Variables

Memory space for fieldbus output variables in fieldbus coupler (Offset)	Addressing data with a data width of															
	1 byte		1 word		3 bytes		1 double word		5 bytes		6 bytes		7 bytes		8 bytes	
	Index	Sub-Index	Index	Sub-Index	Index	Sub-Index	Index	Sub-Index	Index	Sub-Index	Index	Sub-Index	Index	Sub-Index	Index	Sub-Index
0	A000 A040 A040 A080 *	LSB 1 MSB	A0C0 A100	LSB 1 MSB	A140 A180	LSB 1 MSB	A1C0 A200 A240	LSB 1 MSB	A280 A2C0	LSB 1 MSB	A300 A340	LSB 1 MSB	A380 A3C0	LSB 1 MSB	A400 A440	LSB 1 MSB
1	A000 A040 A080 *	2														
2	A000 A040 A080 *	3	A0C0 A100	2												
3	etc.				A140 A180	2										
4	. . .		etc.				A1C0 A200 A240	2								
5									A280 A2C0	2						
6			. . .		etc.						A300 A340	2				
7													A380 A3C0	2		
8							etc.								A400 A440	2
9					. . .											
10									etc.							
11																
12							. . .				etc.					
13																
14													etc			
15									. . .							
16															etc.	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
510			Max. index	Max. sub-index	Max. index	Max. sub-index	Max. index	Max. sub-index	Max. index	Max. sub-index	Max. index	Max. sub-index	Max. index	Max. sub-index	Max. index	Max. sub-index
511	Max. index	Max. sub-index														

* Up to 8 Boolean values can be stored in one byte (1 sub-index); i.e., there is no bit dissolution for each sub-index.

8.2.1.2 Access Addressing of Fieldbus Input Variables

Memory space for fieldbus input variables in fieldbus coupler (Offset)	Addressing data with a data width of															
	1 byte		1 word		3 bytes		1 double word		5 bytes		6 bytes		7 bytes		8 bytes	
	Index	Sub-Index	Index	Sub-Index	Index	Sub-Index	Index	Sub-Index	Index	Sub-Index	Index	Sub-Index	Index	Sub-Index	Index	Sub index
0	A480 A4C0 A500 *	LSB 1 MSB	A540 A580	LSB 1 MSB	A5C0 A600	LSB 1 MSB	A640 A680 A6C0	LSB 1 MSB	A700 A740	LSB 1 MSB	A780 A7C0	LSB 1 MSB	A800 A840	LSB 1 MSB	A880 A8C0	LSB 1 MSB
1	A480 A4C0 A500 *	2														
2	A480 A4C0 A500 *	3	A540 A580	2												
3	etc.				A5C0 A600	2										
4	• • •		etc.				A640 A680 A6C0	2								
5									A700 A740	2	A780 A7C0	2	A800 A840	2	A880 A8C0	2
6			. . .		etc.											
7																
8							etc.									
9					. . .											
10									etc.		etc.					
11																
12							. . .									
13																
14																
15									. . .							
16																
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
510			Max. index	Max. sub-index	Max. index	Max. sub-index	Max. index	Max. sub-index	Max. index	Max. sub-index	Max. index	Max. sub-index	Max. index	Max. sub-index	Max. index	Max. sub-index
511	Max. index	Max. sub-index														

* Up to 8 Boolean values can be stored in one byte (1 sub-index); i.e., there is no bit dissolution for each sub-index.

8.3 EDS Files

The EDS files (standard and modular) contain the characteristics of the fieldbus coupler and information regarding its communication capabilities. An EDS file is imported and installed by the corresponding configuration software.

The EDS files can be obtained at www.wago.com. When installing the respective EDS file, please refer to the information provided in the documentation of the configuration software you are using.

When operating the fieldbus coupler on a WAGO-I/O-IPC (CANopen master), ensure that the sequence of the selected modules in the CoDeSys 2.3 controller configuration is consistent with the topology of your 767 node when using the modular EDS file. Otherwise, the assignment of the sub-indices will not match the data from the I/O modules, causing this data to be stored in the PDOs in the wrong order.

In the upper part of the window labeled "Available Modules," the item-oriented modules (I/O modules) "767-xxxx" of the modular EDS file is available. These modules are based on indices 0x6000 through 0x67FE. Enter the desired modules into the "Selected Modules" window according to the topology of your 767 node.

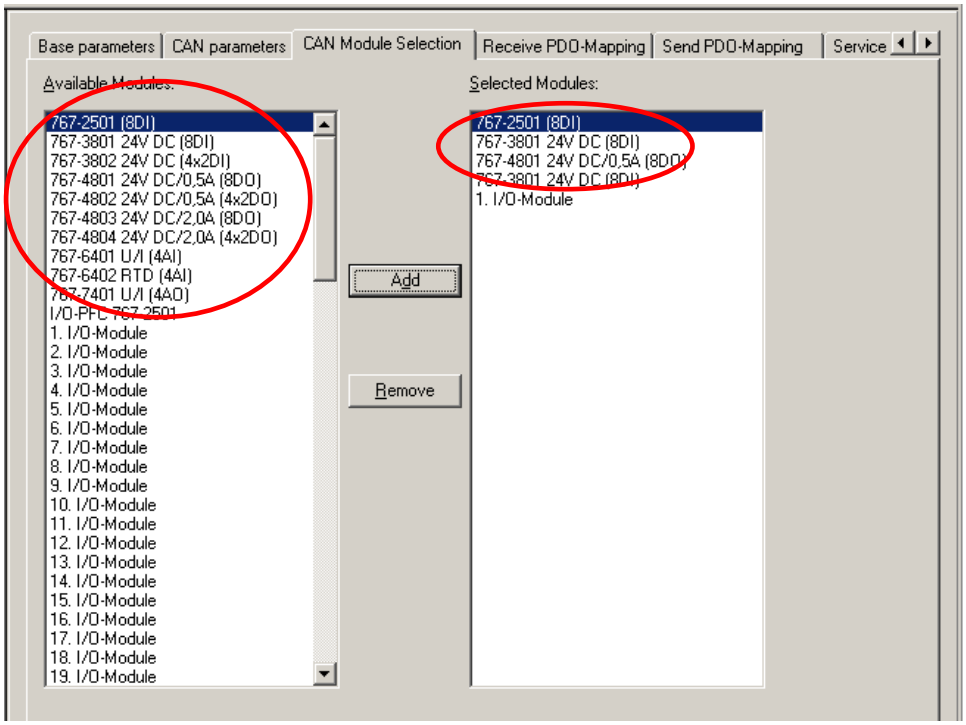


Figure 25: Example of CoDeSys controller configuration 1
(fieldbus coupler as slave on WAGO-I/O-IPC)

In the lower part of the window labeled "Available Modules," the modules "1st Module" up to "64th Module" are listed by slot. These modules are based on indices 0x5300 through 0x5340 and 0x5400 through 0x5440. The data structures of complex modules only located in this indices and not additionally in the indices 0x6000 – 0x67F. The modules with complex data structures include:

- 767-5801 in operating modes 4 and 5 (use of the counter)
- 767-5802 in operating modes 4 and 5 (use of the counter)

Enter the desired modules into the "Selected Modules" window following the item-oriented modules.

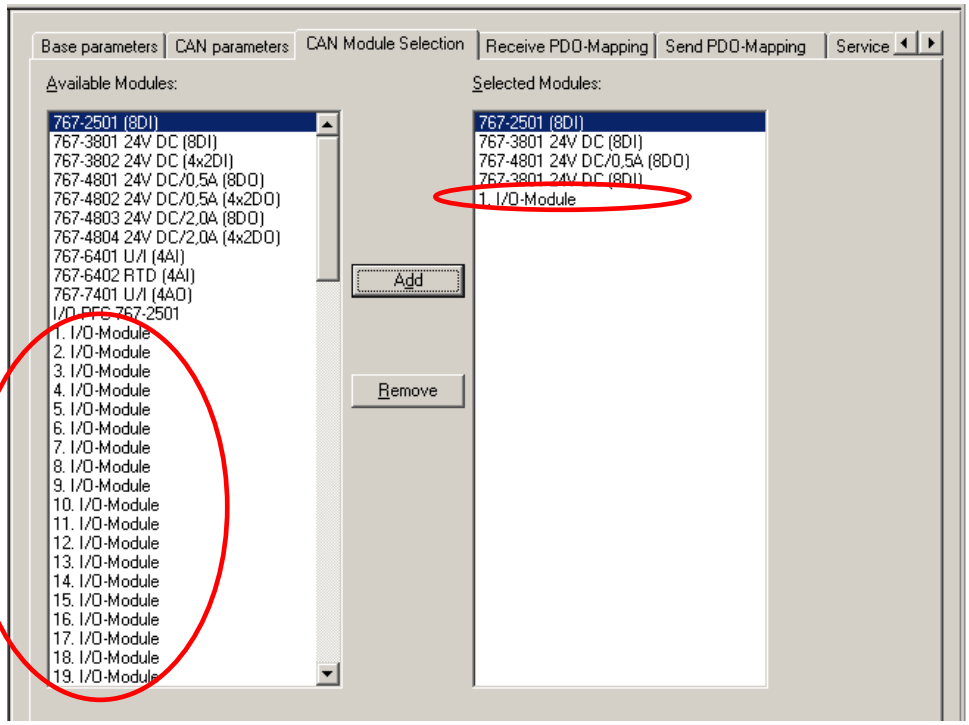


Figure 26: Example of CoDeSys controller configuration 2
(fieldbus coupler as slave on WAGO-I/O-IPC)

8.4 PDO Mapping

Mapping (configuring) the process data objects (PDO), determines which data the fieldbus coupler is exchanging with other CANopen devices on the CAN bus using the PDO. Mapping is carried out separately for each PDO. A mapping entry references the data in the object directory in which the process data of the I/O modules is stored.

If you are not using the configuration you have saved, the PDOs are assigned the default configuration per the DS 401 device profile. The default configuration is dependent on the type and quantity of connected I/O modules.

There are two options for PDO mapping:

- Default PDO mapping
  The default PDO mapping is the configuration per the DS 401 device profile after start-up of the fieldbus coupler, if no separate configuration has been saved.

- Application-specific mapping
  An application-specific mapping is a default PDO mapping modified on the basis of your specifications. It can be saved and used outside of operation.

⚠ WARNING

Data Consistency!
When distributing process data to various PDOs, data inconsistency can occur. This data can lead to a malfunction of your system, and personnel can be injured. Data consistency is only guaranteed for data inside a PDO.

Note

An object entry can only be inserted in a maximum of 3 different PDOs. If it is inserted more frequently, an SDO Abort message is sent during PDO validation with code "0800 0021": "Data cannot be transferred or stored to the application because of local control (see Section 13.2.5). The object is not taken into account in the PDO.

8.4.1 Default PDO Mapping

After switching on the fieldbus coupler, the coupler configure the first four PDOs with the process data of the I/O modules. The digital data is entered into the first PDO, the analog data into PDOs 2 – 4. No mapping is created for all other PDOs.

Example:

1st RxPDO:
The process data of the digital output modules are combined into bytes from blocks of 8 bits. The first RxPDO contains the data from the first eight digital output blocks.

If no digital output modules are connected to the fieldbus coupler, sub-index 0 contains the value 0; i.e., no mapping is available for this PDO.

Table 19: 1st RxPDO mapping

Index	Sub-Index	Description	Default Setting
0x1600	0	Number of available sub-indices (without sub-index 0)	0: no digital output block entered into PDO 1 … 8: digital output blocks entered into PDO
	1	1st digital output block	0x6200 01 08
	2	2nd digital output block	0x6200 02 08
	…	…	…
	8	8th digital output block	0x6200 08 08

2nd RxPDO:
The second RxPDO contains the process data from the first four outputs, which have a data width of 16 bits each. These outputs may be linked to different I/O modules. The important component is a data width of 16 bits per output. The data width of an I/O module channel can be found in the corresponding manual.

If no 16 bit wide outputs are connected to the fieldbus coupler, sub-index 0 contains the value 0; i.e., no mapping is available for this PDO.

Table 20: 2nd RxPDO

Index	Sub-Index	Description	Default Setting
0x1601	0	Number of available sub-indices (without sub-index 0)	0: no 16 bit wide output entered into PDO. 1 … 4: 16 bit wide outputs entered into PDO.
	1	1st analog output channel, data width: 16 bits	0x6411 01 10
	2	2nd analog output channel, data width: 16 bits	0x6411 02 10
	3	3rd analog output channel, data width: 16 bits	0x6411 03 10
	4	4th analog output channel, data width: 16 bits	0x6411 04 10

3rd RxPDO:
The third RxPDO contains the process data from outputs 5 – 8, which have a data width of 16 bits each.

If no more than four 16 bit wide outputs are connected to the fieldbus coupler, sub-index 0 contains the value 0; i.e., no mapping is available for this PDO.

Table 21: 3rd RxPDO

Index	Sub-Index	Description	Default Setting
0x1602	0	Number of available sub-indices (without sub-index 0)	0: no 16 bit wide output entered into PDO. 1 … 4: 16 bit wide outputs entered into PDO.
	1	5th analog output channel, data width: 16 bits	0x6411 05 10
	2	6th analog output channel, data width: 16 bits	0x6411 06 10
	3	7th analog output channel, data width: 16 bits	0x6411 07 10
	4	8th analog output channel, data width: 16 bits	0x6411 08 10

4th RxPDO:
The fourth RxPDO contains the process data from outputs 9 – 12, which have a data width of 16 bits each.

If no more than eight 16 bit wide outputs are connected to the fieldbus coupler, sub-index 0 contains the value 0; i.e., no mapping is available for this PDO.

Table 22: 4th RxPDO

Index	Sub-Index	Description	Default Setting
0x1603	0	Number of available sub-indices (without sub-index 0)	0: no 16 bit wide output entered into PDO. 1 … 4: 16 bit wide outputs entered into PDO.
	1	9th analog output channel, data width: 16 bits	0x6411 09 10
	2	10th analog output channel, data width: 16 bits	0x6411 0A 10
	3	11th analog output channel, data width: 16 bits	0x6411 0B 10
	4	12th analog output channel, data width: 16 bits	0x6411 0C 10

1st TxPDO:
The process data of the digital inputs are combined into bytes from blocks of 8 bits. The first TxPDO contains the data from the first eight digital input blocks.

If no digital inputs are connected to the fieldbus coupler, sub-index 0 contains the value 0; i.e., no mapping is available for this PDO.

Table 23: 1st TxPDO

Index	Sub-Index	Description	Default Setting
0x1A00	0	Number of available sub-indices (without sub-index 0)	0: no digital input block entered into PDO. 1 … 8: digital input blocks entered into PDO.
	1	1st digital input block	0x6000 01 08
	2	2nd digital input block	0x6000 02 08
	…	…	…
	8	8th digital input block	0x6000 08 08

2nd TxPDO:
The second TxPDO contains the data from the first four inputs, which have a data width of 16 bits each. These channels may be linked to different I/O modules. The important component is a data width of 16 bits per channel. The data width of an I/O module channel can be found in the corresponding manual.

If no analog inputs are connected to the fieldbus coupler, sub-index 0 contains the value 0; i.e., no mapping is available for this PDO.

Table 24: 2nd TxPDO

Index	Sub-Index	Description	Default Setting
0x1A01	0	Number of available sub-indices (without sub-index 0)	0: no 16 bit wide input entered into PDO. 1 … 4: 16 bit wide inputs entered into PDO.
	1	1st analog input channel, data width: 16 bits	0x6401 01 10
	2	2nd analog input channel, data width: 16 bits	0x6401 02 10
	3	3rd analog input channel, data width: 16 bits	0x6401 03 10
	4	4th analog input channel, data width: 16 bits	0x6401 04 10

3rd TxPDO:
The third TxPDO contains the process data from analog inputs 5 – 8, which have a data width of 16 bits each.

If no more than four 16 bit wide inputs are connected to the fieldbus coupler, sub-index 0 contains the value 0; i.e., no mapping is available for this PDO.

Table 25: 3rd TxPDO

Index	Sub-Index	Description	Default Setting
0x1A02	0	Number of available sub-indices (without sub-index 0)	0: no 16 bit wide input entered into PDO. 1 … 4: 16 bit wide inputs entered into PDO.
	1	5th analog input channel, data width: 16 bits	0x6401 05 10
	2	6th analog input channel, data width: 16 bits	0x6401 06 10
	3	7th analog input channel, data width: 16 bits	0x6401 07 10
	4	8th analog input channel, data width: 16 bits	0x6401 08 10

4th TxPDO:
The fourth TxPDO contains the process data from analog inputs 9 – 12, which have a data width of 16 bits each.

If no more than eight analog inputs are connected to the fieldbus coupler, sub-index 0 contains the value 0; i.e., no mapping is available for this PDO.

Table 26: 4th TxPDO

Index	Sub-Index	Description	Default Setting
0x1A03	0	Number of available sub-indices (without sub-index 0)	0: no 16 bit wide input entered into PDO. 1 … 4: 16 bit wide inputs entered into PDO.
	1	9th analog input channel, data width: 16 bits	0x6401 09 10
	2	10th analog input channel, data width: 16 bits	0x6401 0A 10
	3	11th analog input channel, data width: 16 bits	0x6401 0B 10
	4	12th analog input channel, data width: 16 bits	0x6401 0C 10

8.4.2 Application-Specific Mapping

In contrast to the default mapping, an application-specific PDO mapping gives you the option to specify which data is to be transferred by the PDO. This requires that the fieldbus coupler is in its pre-operational state. For more information, see Section 12.

The following example illustrates the procedure of an application-specific mapping with a fieldbus coupler set at the CANopen station address 8. The PDO communication parameters are also set in this example.

Using TxPDO 2, the third and fifth analog channels of an input module with the data width of 16 bits, as well as the first digital input block (8 bit) are to be transferred. The CAN identifier (COB ID) 0x432 is used for the transfer. The PDO transfer is carried out synchronously with every third SYNC object. The default COB ID is used for the SDO.

1. First deactivate the PDO mapping by setting the number of mapping objects in index 0x1A01, sub-index 0 ("Transmit PDO Mapping Parameter") to 0.

Table 27: Deactivate PDO mapping

	COB ID	Data
Send	608	0x2F 01 1A 00 00 xx xx xx
Receive	588	0x60 01 1A 00 xx xx xx xx

2. Enter the index, sub-index and object length of the application object into the mapping parameter structure of the TxPDO (index 0x1A01). A maximum of 8 bytes of data can be assigned for each PDO.

Table 28: Mapping parameters

Application Object	Index	Sub-Index
3rd analog input channel	0x6401	3
5th analog input channel	0x6401	5
1st digital input module	0x6000	1

3. This requires the following structure to be attained in the mapping parameters of the second TxPDO.

Table 29: TxPDO mapping parameter structure of TxPDO, index 0x1A01

Sub-Index	Application Object		Object Length (in bits)
	Index	Sub-Index	
0	3		
1	0x6401	1	0x10
2	0x6401	2	0x10
3	0x6000	3	0x08

Note

First enter the mapping parameter into sub-index 1 … 8, then enter the number of valid sub-indices into sub-index 0.

These objects are stored using SDO transmissions:

Table 30: Insert 3rd analog input channel

	COB ID	Data/Structure of CANopen Telegram
Send	0x608	0x23 01 1A 01 10 03 01 64: 23 Download 011A Index (Little Endian) 01 Sub-index 10 Data width of analog channel 03 Sub-index where 3rd analog channel is in "Manufacturer Device Profile" 00 64 Index (Little Endian) where 3rd analog channel is in Manufacturer Device Profile
Receive	0x588	0x60 01 1A 01 xx xx xx xx 60 OK 011A Index (Little Endian) 01 Sub-index

Table 31: Insert 5th analog input channel

	COB ID	Data
Send	0x608	0x23 01 1A 02 10 05 01 64
Receive	0x588	0x60 01 1A 02 xx xx xx xx

Table 32: Insert 1st digital input module

	COB ID	Data
Send	0x608	0x23 01 1A 03 08 01 00 60
Receive	0x588	0x60 01 1A 03 xx xx xx xx

Table 33: Number of mapping objects = 3 (enter into sub-index 0)

	COB ID	Data
Send	0x608	0x2F 01 1A 00 03 xx xx xx
Receive	0x588	0x60 01 1A 00 xx xx xx xx

To change the communication parameters, proceed as follows:

4. Deactivate the PDO that you wish to configure. In the example, this is TxPDO2. To do this, write the value 0x80000000 into the object with index 0x1801, sub-index 01 ("Transmit PDO Communication Parameter").

Table 34: Deactivate PDO

	COB ID	Data
Send	608	0x23 01 18 01 00 00 00 80
Receive	588	0x60 01 18 01 xx xx xx xx

5. Write the communication parameters into the object with index 0x1801, sub-index 1 through 3 ("Transmit PDO Communication Parameter"). The "Transmission Type" is 3 (synchronous transmission with every third SYNC object).

Table 35: Entering "Communication Parameters"

TxPDO Communication Parameter, Index 0x1801		
Sub-Index	**Value**	**Description**
0	3	Number of supported sub-indices
1	32 04 00 00 (0x432)	COB ID of PDO
2	3	"Transmission Type"
3	0	"Inhibit Time"

Table 36: Sub-index 3: "Inhibit Time" = 0

	COB ID	Data
Send	0x608	0x2B 01 18 03 00 00 xx xx
Receive	0x588	0x60 01 18 03 xx xx xx xx

Table 37: Sub-index 2: "Transmission Type" = 3

	COB ID	Data
Send	0x608	0x2F 01 18 02 03 xx xx xx
Receive	0x588	0x60 01 18 02 xx xx xx xx

Table 38: Sub-index 1: Setting COB ID of PDO to 0x432 and from invalid to valid

	COB ID	Data
Send	0x608	0x23 01 18 01 32 04 00 00
Receive	0x588	0x60 01 18 01 xx xx xx xx

8.5 Enabling the Analog Input Data

In the default setting, the PDO transfer of analog input data is switched off. Therefore, this data is only sent once by the fieldbus coupler and is not subsequently updated. The analog input data should be enabled so that the data can be used via the PDO. To do this, proceed as follows:

1. Switch the fieldbus coupler to its pre-operational state. For more information, see Section 12.1.2.

2. Set object 0x6423 to TRUE (= 1). The analog input data can be transferred (see Section 19.3.12).

Access via SDOs is currently possible.

8.6 Inputs on the CANopen Fieldbus Coupler

The fieldbus coupler has eight digital inputs, which resemble an 8 DI input module (767-3801) from the perspective of the CANopen application. Its process data and diagnostic data are accessible in indices 0x5300 and 0x5400 of the object directory. A description of the indices can be found in Sections 10.2.5 and 10.2.6.

8.7 Delayed I/O Module Start-Up

The topology of a 767 node can cause I/O modules to be delayed in starting up. So that the CANopen application is not blocked on the fieldbus coupler as a result of this delay, it starts independent of the S-BUS. If the S-BUS is not ready, the CANopen application sends the error message "No process data available," which is displayed as follows:

"Error Code":	0xFF00
"Error Register":	0x81
"Additional Code":	00 06 00 00 00

Because the process data of the I/O modules is unavailable, you cannot access their indices in the object directory; i.e., indices 0x5300 through 0x5340, 0x5400 through 0x5440, 0x5202 and the 0x6xxx indices cannot be accessed. The following error message ("Abort Code") is sent in the SDO reply telegram: 0x0800 0022 ("Data cannot be transferred or stored to the application because of the present device state"). If a configuration is saved, it is not loaded. Rather, a default configuration is set.

As soon as the S-BUS is available, the indices are created, the saved configuration is loaded (if applicable) and the outgoing error message "No process data available" is sent.

Detailed information on error codes can be found in Section 10.

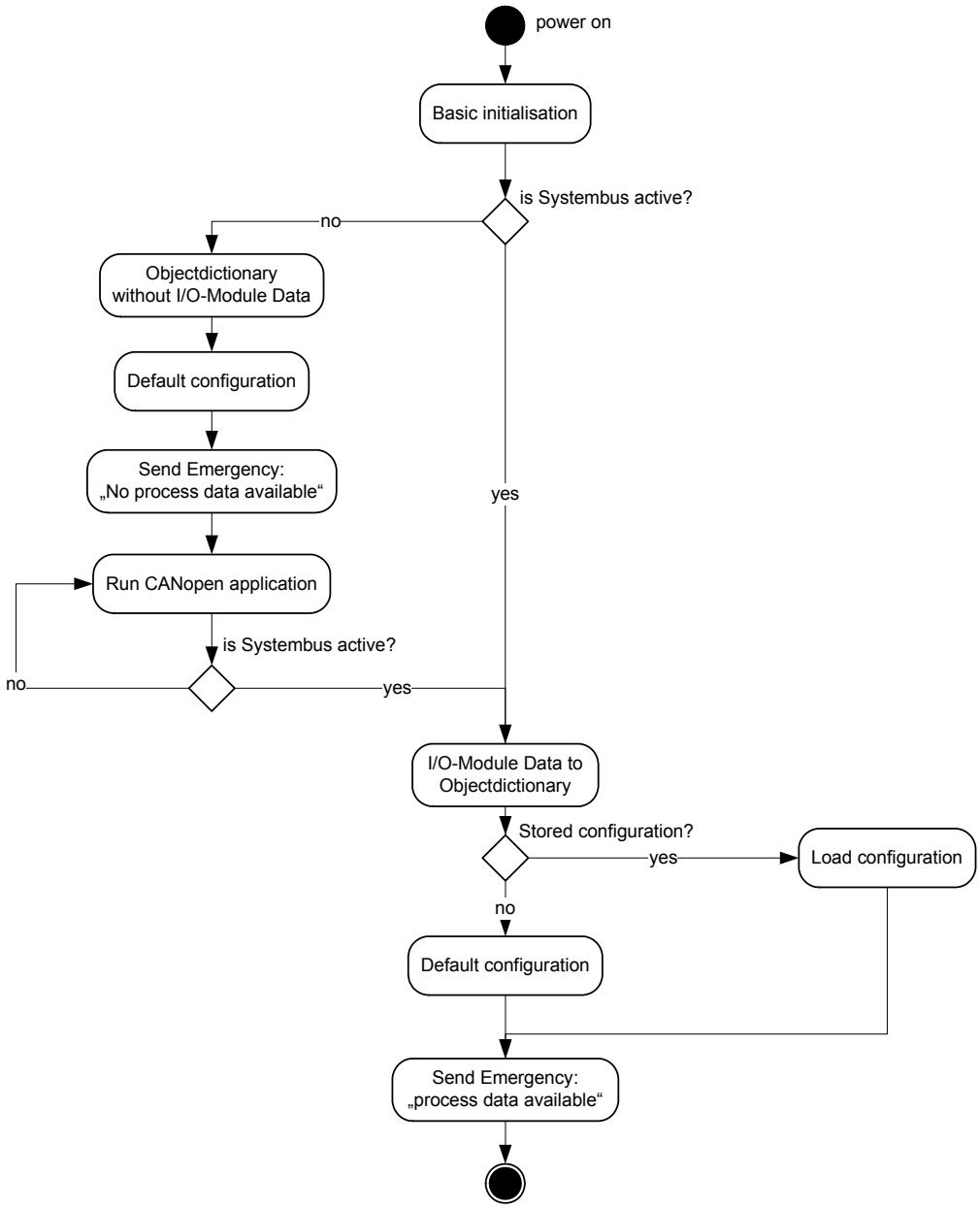


Figure 27: Fieldbus coupler response to delayed start-up

8.8 S-BUS Disruption During Continuous Operation

If a disruption of the S-BUS occurs during continuous operation, the output modules output the process value 0 (additional information can be found in the corresponding I/O module documentation). The CANopen application on the fieldbus coupler sends the error message (Emergency) "No process data available," which is displayed as follows:

"Error Code":	0xFF00
"Error Register":	0x81
"Additional Code::	00 06 00 00 00

Because the process data of the I/O modules is unavailable, you cannot access their indices in the object directory; i.e., indices 0x5300 to 0x5340, 0x5400 to 0x5440 and the 0x6xxx indices cannot be accessed. The following error message ("Abort Code") is sent in the SDO reply telegram: 0x0800 0022 ("Data cannot be transferred or stored to the application because of the present device state"). If a configuration is saved, it is not loaded. Rather, a default configuration is set.

So that the process value 0 is output to the outputs of the I/O modules when the S-BUS restarts, the CANopen application switches to its pre-operational state and the CANopen process image is deleted.

The S-BUS restarts when the disruption has been cleared, as long as the I/O module configuration has not been modified. In this case, the CANopen application sends the outgoing error message "No process data available."

Detailed information on error codes can be found in Section 10.

8.9 Activating Data Exchange

To activate the process data exchange, switch the fieldbus coupler to operational mode via the "Start Remote Node" command. Details on this command can be found in Section 13.7.1. The PDO is active in operational mode, and the CANopen devices can use the PDO to exchange data.

9 Diagnostics

This section contains all information required to analyze the CANopen diagnostic information of the 767 components. CANopen offers the possibility of reporting error states in the form of emergencies.

9.1 Diagnostic Messages from I/O Modules

I/O modules provide various diagnostic information, such as errors (e.g., internal hardware malfunctions), warnings (e.g., short circuit or wire break) and messages (e.g., measuring range overflow). The individual information of an I/O module is combined into bytes, resulting in the dianostic value. The diagnostic value always lies behind the process values in the input data index for I/O modules (index 0x5300 through 0x5340).

Diagnostic information is acknowledged using diagnostic acknowledgement. A CANopen device can reset the diagnostic information of the I/O modules by writing the diagnostic acknowledgement in the fieldbus coupler. This functionality must be configured in the I/O module. Additional information can be found in the manuals for the I/O modules in the section on process images. The diagnostic acknowledgement and the diagnostic value are identically structured.

Example:
An 8 DI module (767-3801) has 8 digital inputs (1 byte) and supplies a diagnostic value of 1 byte. The 8 DI module is located at the 9th position behind the fieldbus coupler (the fieldbus coupler begins at position 0).

The fieldbus coupler supplies the data for the input module in index 0x5309. The process data is stored in sub-index 1, the diagnostic value in sub-index 2. The diagnostic acknowledgement is stored in the indices of the input module's output data (index 0x5400 through 0x5440) behind the process data.

Both diagnostic information and diagnostic acknowledgements can be transferred via the PDO.

Note

Information on the structure of the diagnostic values and the diagnostic acknowledgement can be found in the manuals for the I/O modules in the section on process images.

10 Error Messages (Emergency)

Error messages are given when an error situation arises or is resolved in the fieldbus coupler. The structure and meaning of the entries in the emergency object are illustrated in the table below.

After an error has been resolved, an "Emergency Object" is also sent ("Error Code" = 0x0000). The "Error Register" and "Manufacturer-Specific Error Code" function according to the table below.

After restarting the fieldbus coupler, an "Emergency Object" is sent if the default settings are involved. There can be two reasons for this:

- No settings have been saved (index 0x1010).
- The saved setting was rejected by the fieldbus coupler because the node structure (type and number of connected I/O modules) has changed since that last start-up.

The following table lists all error codes. The "Manufacturer-Specific Error Code" consists of 5 bytes and is represented in the table as a hex value with an arrangement of low bytes to high bytes. On the second byte of the "Manufacturer-Specific Error Code," the error message can be clearly identified.

Table 39: Description of error messages

Byte	0, 1	2	3 – 7	
Name	"Error Code"	"Error Register"	"Manufacturer - Specific Error Code"	Description
	0x5000	0x81	00 01 00 00 00	Modified HW configuration after switching on power supply or after hardware reset. The fieldbus coupler was newly initialized because either a saved configuration is unavailable or the fieldbus coupler is not consistent with the connected I/O modules.
	0x5000	0x11	00 01 00 00 00	"CAN Controller Overrun". A receiving telegram was overwritten by another.
	0x8100	0x81	00 04 00 00 00	The duration between two sync telegrams is longer than the "Communication_Cycle_Period."
	0x8110	0x11	00 01 00 00 00	Internal receive memory overrun; state change is defined as in object 0x67FE. The outputs are connected as defined in the error mode/value objects.

Table 39: Description of error messages

Byte	0, 1	2	3 – 7	
Name	"Error Code"	"Error Register"	"Manufacturer - Specific Error Code"	Description
	0x8110	0x11	00 02 00 00 00	Internal send memory overrun; state change is defined as in object 0x67FE. The outputs are connected as defined in the error mode/value objects.
	0x8120	0x11	00 03 00 00 00	CAN driver in "Error Passive Mode"
	0x8130	0x11	00 04 00 00 00	The duration between two Node-Guarding Telegrams is longer than the "Guard_Time" multiplied by the "Life_Time_Factor."
	0x8130	0x11	00 05 KK 00 00	The duration between two heartbeat telegrams is longer than configured. KK: CANopen device that caused the timeout.
	0x8210	0x81	00 05 SS II NN	PDO was sent with a smaller number of bytes than was configured in "Communication Profile." The PDO data was rejected; i.e., the outputs remain unchanged. SS: Target value; configured value (e.g., in index 0x1600, sub-index 0). II: Actual value – number of bytes sent NN: Number of PDOs (1 … 32)
	0x8220	0x81	00 08 SS II NN	PDO was sent with a larger number of bytes than was configured in the "Communication Profile." Only the first n-data is used (n = total length configured in object directory). SS: Target value; configured value (total length of all valid, configured objects in bytes) II: Actual value; number of bytes sent NN: Number of PDOs (1 … 32)
	0xFF00	0x81	00 06 00 00 00	No process data available; see Section 8.7.

Table 39: Description of error messages

Byte	0, 1	2	3 – 7	
Name	"Error Code"	"Error Register"	"Manufacturer - Specific Error Code"	Description
	0xFF00	0x81	CC 07 PP 00 NN	Diagnostic message CC: "Diagnosis Classification," "Error" (1)/"Warning" (2)/ "Notification" (4)/"Outgoing Diagnosis" (0) PP: Slot of I/O module or fieldbus coupler 0 … 64, 255. 0: Digital inputs of fieldbus coupler. 1 .. 64 Diagnostic messages from IO modules. 255: Diagnostic messages from fieldbus coupler. NN: Number of current diagnostic messages
	0xFF00	0x81	00 09 EG EC EA	Fieldbus coupler CS LED flash signals: see Section 16.3. EG: Error group EC: Error code EA: Error argument

11 Object Directory

The object directory contains all configuration information and process data from the fieldbus coupler. The directory is organized in a table and includes three different types of CANopen objects:

- **Communication Profile Area (Index 0x1000 – 0x1FFF)**
  This area includes all parameters that are relevant for CANopen communication. The structure is specified by DS 301 and is identical for all CANopen devices. For more information, see Section 11.1.

- **Manufacturer-Specific Profile Area (Index 0x2000 – 0x5FFF)**
  Manufacturers can implement their own company-specific objects in this area. For more information, see Section **19.2**.

- **Standardized Device Profile Area (Index 0x6000 – 0x9FFF)**
  This area includes all objects that are supported by a specific device profile. The fieldbus coupler supports device profile DS-401 ("Device Profile for Generic I/O Modules"). For more information, see Section **19.3**.

Every entry in the directory is identified by a 16-bit index. You can access the object via this index. Up to 65536 indices are possible. If an object is composed of several components, the components can be addressed by an 8-bit sub-index. Up to 256 sub-indices are possible.

If the index consists only of simple variables (UNSIGNED8, UNSIGNED16, etc.), the sub-index will always be zero. For objects structured with more complexity (arrays, records, etc.), sub-index 0 indicates the maximum number of the subsequent sub-indices. The data is coded in the subsequent sub-indices.

Each entry consists of

- an object name that describes the function of the object,
- a data type attribute that defines the data type of the entry
- an access attribute that indicates whether the entry is only allowed to be read or written or both.

Table 40: Structure of CANopen object directory

Index (hexadecimal)	Object
0x0000	Not used
0x0001 – 0x001F	Static data types
0x0020 – 0x003F	Complex data types
0x0040 – 0x005F	Manufacturer-specific data types
0x0060 – 0x007F	Profile-specific static data types
0x0080 – 0x009F	Profile-specific complex data types
0x00A0 – 0x0FFF	Reserved
0x1000 – 0x1FFF	Communication profile (DS 301)
0x2000 – 0x5FFF	Manufacturer-specific parameters
0x6000 – 0x9FFF	Parameters from the standardized device profiles

The object directory is dependent on the connected I/O modules.

11.1 Communication Profile Area

The following table lists all the communication profile objects that are supported by the fieldbus coupler.

The detailed objects from the fieldbus coupler are covered in Section 19.1.

Table 41: Communication profile objects

Index	Name	Data Type	Description	Information
0x1000	Device type	Unsigned32	Device profile	For more information, see Section 19.1.1.
0x1001	Error register	Unsigned8	Internal error	For more information, see Section 19.1.2.
0x1003	Pre-defined error field	Array Unsigned 32	Save the last 20 errors that have occurred	For more information, see Section 19.1.3.
0x1005	COB ID SYNC message	Unsigned32	COB ID for the synchronization object	For more information, see Section 19.1.4.
0x1006	Communication cycle period	Unsigned32	Maximum duration between two SYNC telegrams	For more information, see Section 19.1.5.
0x1008	Manufacturer device name	Visible string	Device name	For more information, see Section 19.1.6.
0x1009	Manufacturer hardware version	Visible string	Hardware version	For more information, see Section 19.1.7.
0x100A	Manufacturer software version	Visible string	Software version	For more information, see Section 19.1.8.
0x100C	Guard time	Unsigned16	Monitoring time for the life guarding protocol	For more information, see Section 19.1.9.
0x100D	Life time factor	Unsigned8	"Life Time Factor"	For more information, see Section 19.1.10.
0x1010	Store parameters	Array Unsigned 32	Parameters to store the configuration	For more information, see Section 19.1.11.
0x1011	Restore default parameters	Array Unsigned 32	Parameter to restore the default configuration	For more information, see Section 19.1.12.
0x1014	COB ID emergency object	Unsigned32	COB ID for the emergency object	For more information, see Section 19.1.13.
0x1015	Inhibit time EMCY	Unsigned32	Duration between two EMCY telegrams	For more information, see Section 19.1.14.
0x1016	Consumer heartbeat time	Array Unsigned 32	Heartbeat monitoring time	For more information, see Section 19.1.15.
0x1017	Producer heartbeat time	Unsigned16	Duration between two generated heartbeat telegrams	For more information, see Section 19.1.16.

Table 41: Communication profile objects

Index	Name	Data Type	Description	Information
0x1018	Identity object	Record identity	Device information	For more information, see Section 19.1.17.
0x1029	Error behavior	Array Unsigned 8	State change in the case of error	For more information, see Section 19.1.18.
0x1200 through 0x1201	Server SDO parameter	Record SDO parameter	Parameter for server SDO	For more information, see Section 19.1.19.
0x1400 through 0x141F	Receive PDO communication parameter	Record PDO parameter	Communication parameter for receive PDO	For more information, see Section 19.1.21.
0x1600 through 0x161F	Receive PDO mapping parameter	Record PDO mapping	Mapping parameter for receive PDO	See Section 19.1.22.
0x1800 through 0x181F	Transmit PDO communication parameter	Record PDO parameter	Communication parameter for the send PDO	See Section 19.1.23.
0x1A00 through 0x1A1F	Transmit PDO mapping parameter	Record PDO mapping	Mapping parameter for the send PDO	See Section 19.1.24.

12 CANopen State Diagram

The following state diagram illustrates the individual communication states and possible transitions of CAN communication.

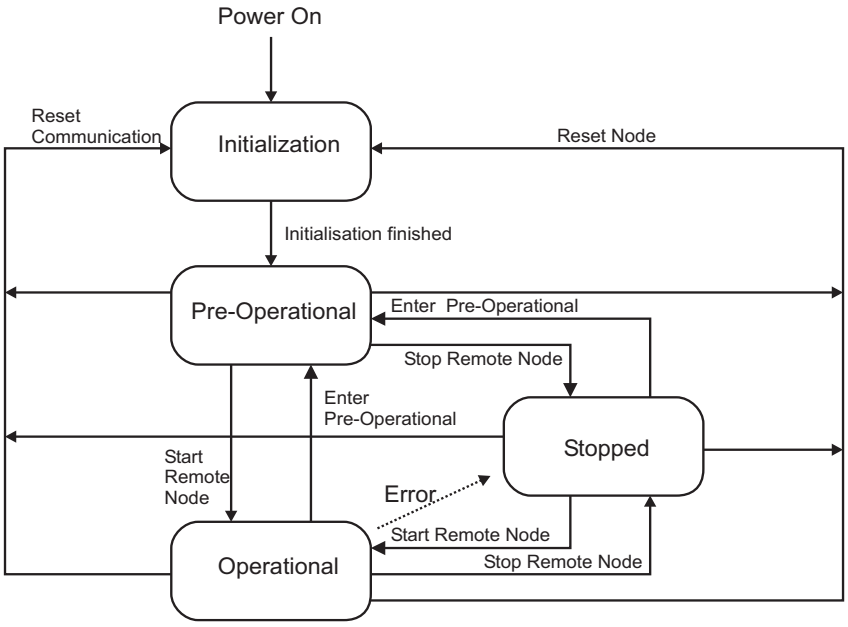


Figure 28: CANopen state diagram

12.1.1 Initialization

After a power-on or restart, the fieldbus coupler is automatically in the initialization state. In this state, the fieldbus coupler starts the S-BUS. The process image is created and the object directory initialized using the connected I/O modules and a saved configuration (if applicable). If no errors were detected during the initialization phase, the fieldbus coupler automatically switches to its pre-operational state. If errors were detected, the fieldbus coupler is switched to the stopped state and an appropriate blink code is emitted.

During initialization, the RUN and ERR LEDs light up red. If initialization and the switch to the pre-operational state were completed successfully, the RUN LED flashes green and the ERR LED goes out. If errors were detected, the RUN LED goes out and the ERR LED lights up red. The CS LED uses red flash sequences to indicate the error type.

12.1.2 Pre-Operational

In this state, communication can take place via SDOs. Communication via the PDO is not possible. Entries in the object directory can be read and written with the SDO. This makes it possible to configure the fieldbus coupler with, for example, a CANopen configuration tool.

Switching from the pre-operational to the operational state is carried out with the NMT service "Start_Remote_Node." In the pre-operational state, the CS LED lights up green and the RUN LED flashes.

12.1.3 Operational

In this state, communication takes place via SDOs and PDOs. Configuration is not possible in this state. For example, changing the COB ID of a valid PDO is not allowed. A detailed description can be found in the corresponding entries in the object directory.

Switching from the operational to the pre-operational state is carried out with the NMT service "Enter_Pre_Operational_State." In the operational state, the CS LED and the RUN LED light up green.

12.1.4 Stopped

If an error arises, the fieldbus coupler switches to the stopped state. This occurs when the NMT service "Stop_Remote_Node" was received or when an error (e.g., S-BUS was disrupted in continuous operation) arises.

This state makes it impossible to communicate using SDOs or PDOs. Only the NMT services and the "Node Guarding"/"Heartbeat" (if activated) are executed.

The stopped state can be exited via the NMT services "Start_Remote_Node_Indication," "Enter_Pre_Operational_State" and "Reset_Node." In the stopped state, the ERR LED lights up red.

13 CANopen Communication Objects

Communication with the devices takes place by means of the communication objects. This section describes the detailed structure of the communication objects and their use.

13.1 Process Data Object (PDO)

This protocol is used to transfer data to and from the fieldbus coupler without protocol overhead. PDOs consist of the COB ID and the data field. A PDO contains no additional protocol information. The data content is determined by the mapping parameters, and the transfer type is determined by the communication parameters. The data telegrams consist of a maximum of 8 bytes.

Distinction is made between RxPDO (receive PDO) and TxPDO (send PDO).

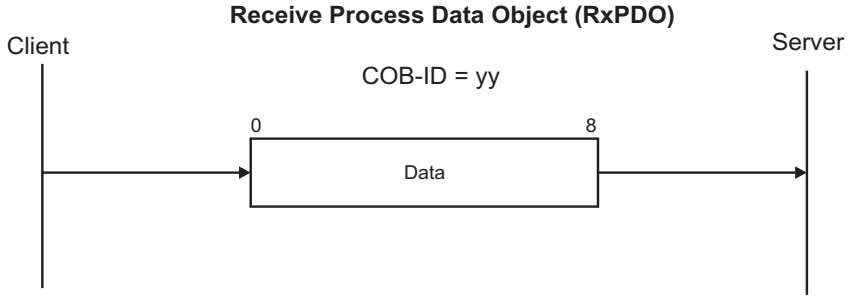


Figure 29: RxPDO

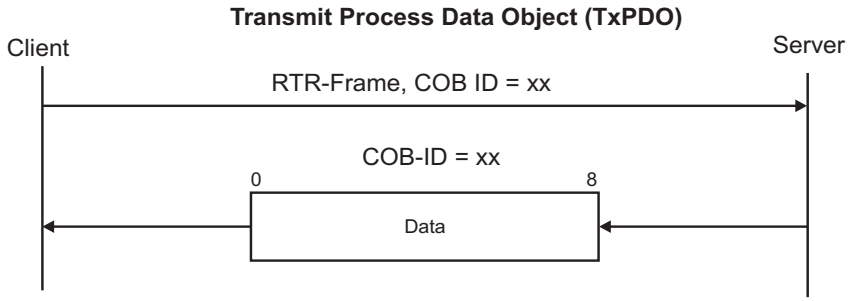


Figure 30: TxPDO

Data transfer via PDO is only possible in the operational state.

When switching to the operational state, all TxPDOs with transfer types 254 and 255 are sent once.

Distinctiveness of TxPDO "Transmission Types" 254 and 255

If the "Transmission Type" for a TxPDO is set to the values 254 or 255, the TxPDO is sent by the fieldbus coupler in an event-driven manner. Device profile DS401 ("Device profile for generic I/O modules"), in which the various events are described, distinguishes between events based on digital input values and events based on analog input values. Object 0x6423 ("Analog Input Global Interrupt Enable") is used to comprehensively switch the event control for analog input values on or off. This is not activated in the delivery condition of the fieldbus coupler. When starting up the fieldbus coupler, a TxPDO is sent with each modification of the analog input.

Because analog process values tend to make noise, a high load can result from TxPDOs on the CAN bus. By extending the "Inhibit Time" in the TxPDO communication parameters, the load on the CAN bus can be reduced. The threshold value monitoring (objects 0x6421, 0x6424 and 0x6425) and the delta functions (objects 0x6426, 0x6427 and 0x6428) offer additional options for reducing the bus load.

13.2 Service Data Object (SDO)

Use the SDO to read and write the entries in the object directory, through which you can fully configure a CANopen device. The default SDO is initialized with a low-priority COB ID. If the transferred data exceeds 4 bytes, it must be allocated across several telegrams.

A protocol overhead is required for the transfer. The protocol overhead must contain the "Command Specifier," the index and the sub-index of the entry that is to be read/written.

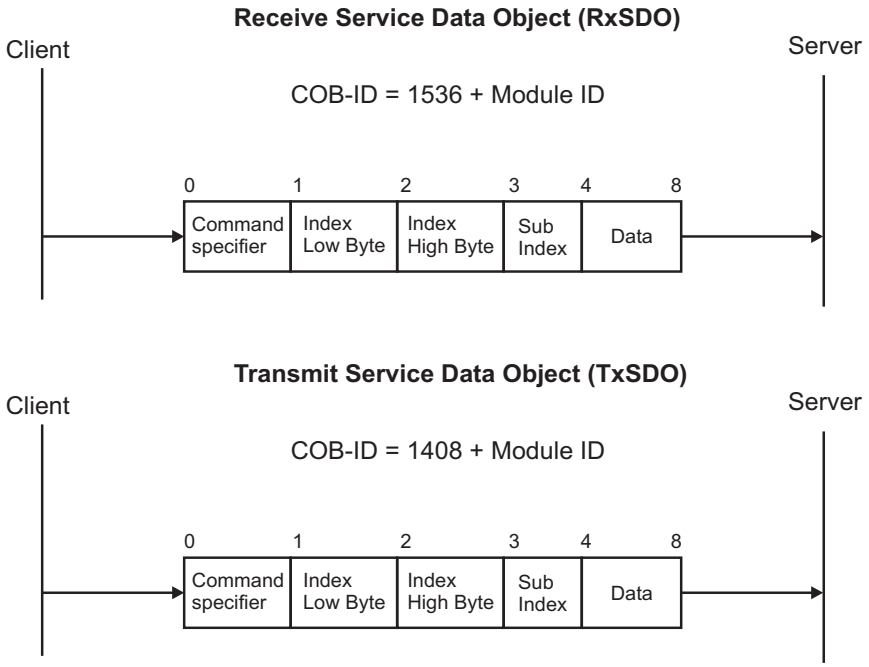


Figure 31: RxSDO and TxSDO

13.2.1 Initiate SDO Download

This protocol is used to initiate data transfer from the master to the fieldbus coupler. If the transfer contains a maximum of 4 bytes of data, this data is included in the transfer inside the protocol.

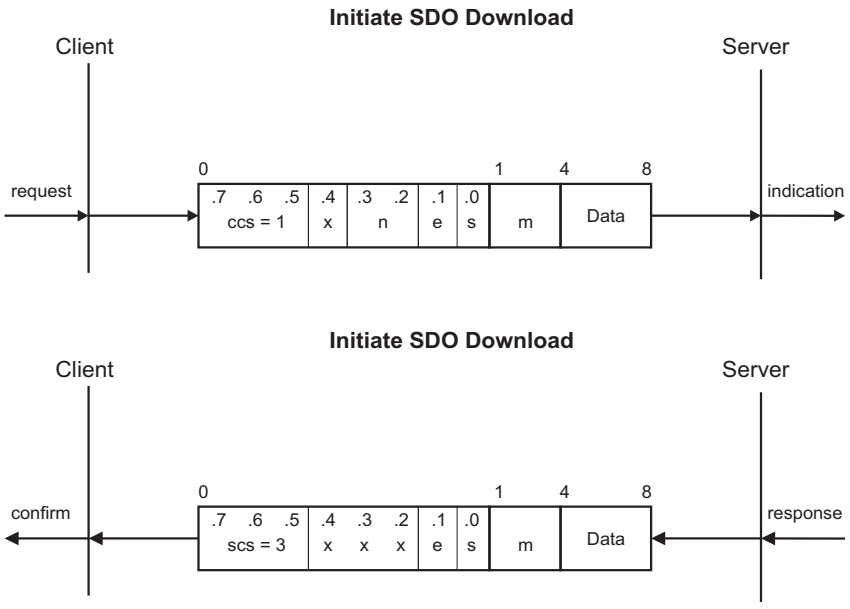


Figure 32: Initiate SDO Download

Table 42: Description of "Initiate SDO Download" protocol

Abbreviation	Term	Description
ccs:	Client command specifier	1: Initiate download request
scs:	Server command specifier	3: Initiate download response
n:	Only valid when e = 1 and s = 1; otherwise 0.	When n is valid, it indicates the number of bytes that contain no data. Example: 3 data bytes, e = 1 and s = 1, n = 4 - 3 = 1
e:	Transfer type	0: Normal transfer Number of bytes to be written >= 5 bytes
		1: Expedited transfer Number of bytes to be written < 5 bytes
s:	Size indicator	0: "Data set size" is not displayed
		1: "Data set size" is displayed s is always 1
m:	Multiplexor	Index and sub-index of object directory: Index, low-byte: byte #1 Index, high-byte: byte #2 Sub-Index: byte #3
d:	Data	e = 0, s = 0: d is reserved for another use by CANopen
		e = 0, s = 1: d contains the number of bytes for the download
		Byte 4 contains the LSB; byte 7 contains the MSB.
		e = 1: d contains the data that is to be transferred
X:	-	Not used; always 0
reserved:	-	Reserved for another use by CANopen

13.2.2 Download SDO Segment

If more than 4 bytes of data are to be transferred, use this protocol. Data transfer begins after the "Initiate SDO Download Protocol," which initiates the transfer, is processed.

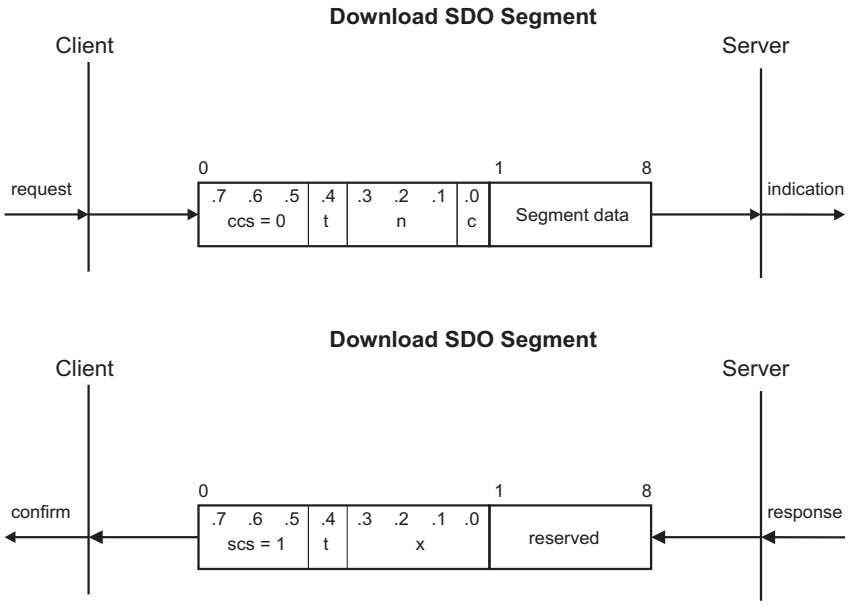


Figure 33: Download SDO Segment

Table 43: Description of the "Download SDO Segment" protocol

Abbreviation	Term	Description
ccs:	Client command specifier	0: "Download segment request"
scs:	Server command specifier	1: "Download segment response"
seg-data	Contains the data that is to be transferred.	The application determines the importance of the data.
n:		Indicates the number of bytes that contain no data. n is 0 when no segment size is indicated.
c:	Indicates whether additional data should be downloaded.	0: Data is not yet available for download.
		1: No more data is available for download.
t:	Toggle bit	This bit must be inverted for each segment for which a download takes place. The first segment sets the toggle bit to 0. The toggle bit is identical in both the request and reply telegrams.
X:	-	Not used. Always 0.
reserved:	-	Reserved for another use by CANopen.

13.2.3 Initiate SDO Upload

This protocol is used to initiate data transfer from the fieldbus coupler to the master. If the transfer contains a maximum of 4 bytes of data, this data is included in the transfer inside the protocol.

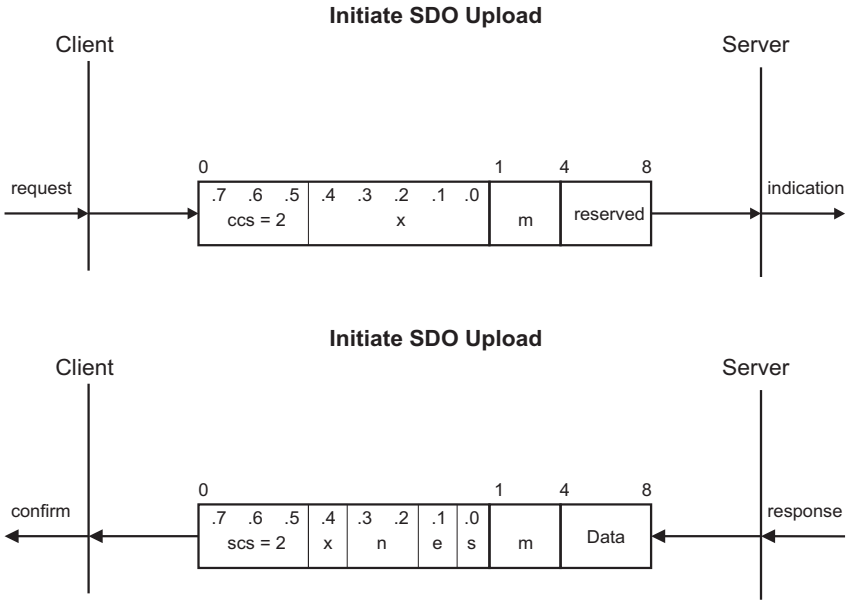


Figure 34: Initiate SDO Upload

Table 44: Description of "Initiate SDO Upload" protocol

Abbreviation	Term	Description
ccs:	Client command specifier	2: "Initiate upload request"
scs:	Server command specifier	2: "Initiate upload response"
n:	Only valid when e = 1 and s = 1; otherwise 0.	When n is valid, it indicates the number of bytes in d that contain no data. The bytes [8 – n, 7] contain no segment data.
e:	Transfer type	0: "Normal transfer," number of bytes to be written >= 5 bytes.
		1: "Expedited transfer" Number of bytes to be written < 5 bytes.
s:	Size indicator	0: The number of bytes that are to be transferred is not indicated.
		1: The number of bytes that are to be transferred is indicated (depending on number of bytes).

Table 44: Description of "Initiate SDO Upload" protocol

Abbreviation	Term	Description
m:	Multiplexor	Index and sub-index of object directory: Index, low-byte: byte #1 Index, high-byte: byte #2 Sub-Index: byte #3
d:	Data	e = 0, s = 0: d is reserved for another use by CANopen.
		e = 0, s = 1: d contains the number of bytes available for upload.
		Byte 4 contains the LSB; byte 7 contains the MSB.
		e = 1: d contains the data that is to be transferred.
X:	-	Not used. Always 0.

13.2.4 Upload SDO Segment

If more than 4 bytes of data are to be transferred, use this protocol. It is sent following the "Initiate Upload Protocol."

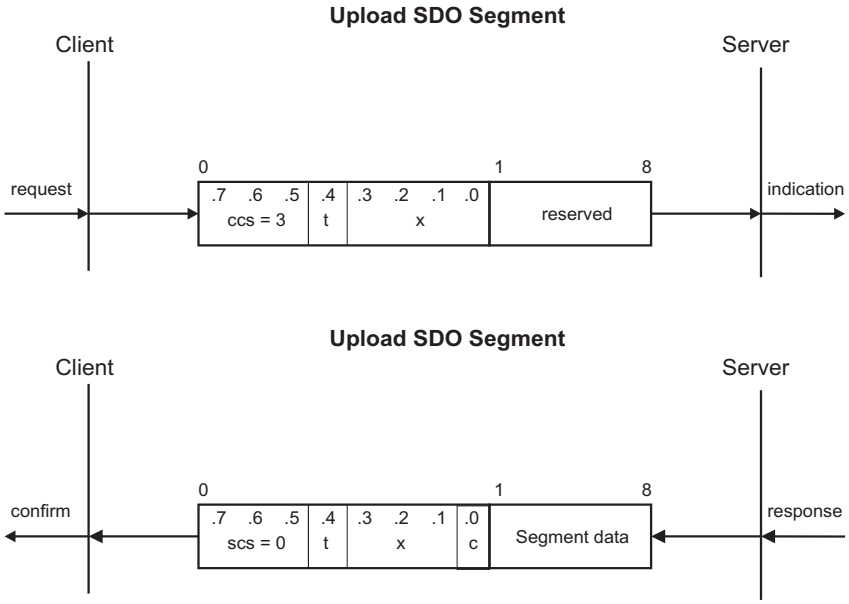


Figure 35: Upload SDO Segment

Table 45: Description of the "Upload SDO Segment" protocol

Abbreviation	Term	Description
ccs:	Client command specifier	3: Download segment request
scs:	Server command specifier	0: Download segment response
t:	Toggle bit	This bit must change for each segment for which an upload takes place. For the first segment, the toggle bit is set to 0. The toggle bit is identical in both the request and reply.
c:	Indicates whether additional segments are available for upload	0: Additional segments available for upload.
		1: No additional segments available for upload.
seg-data	Contains the data that is to be transferred.	The application determines the importance of the data.
n:		Indicates the number of bytes that contain no data. Bytes [8 – n, 7] contain no data. N is 0 when the segment size is not given.
X:	-	Not used; always 0
reserved:	-	Reserved for another use by CANopen.

13.2.5 Abort SDO Transfer

Use this protocol if an SDO transfer is to be aborted.

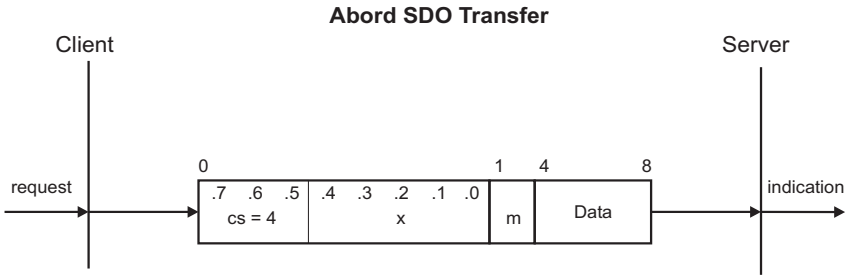


Figure 36: Abort SDO Transfer

Table 46: Description of the Abort SDO protocol

Abbreviation	Term	Description
cs:	Command specifier	4: "Abort domain transfer"
m:	Multiplexor	Index and sub-index of object directory
X:	-	Not used; always 0
Data	4-byte abort code	Information on the cause of the abort.

Structure of the supported abort domain transfer messages

Table 47: Description of error in bytes

Byte	Description
0	Command specifier; 0x80
1	Index, byte 0 (low-byte)
2	Index, byte 1 (high-byte)
3	Sub-Index
4	Abort code, byte 0 (low-byte)
5	Abort code, byte 1
6	Abort code, byte 2
7	Abort code, byte 3 (high-byte)

The following errors are coded in UNSIGNED32 format via the "Additional Code," the "Error Code" and the "Error Class":

Table 48: Error messages arising from the Abort SDO protocol

Byte 7 – byte 4 Abort Code	Description
0503 0000	Toggle bit not alternated
0504 0000	SDO protocol timed out
0504 0001	Client/server command specifier not valid or unknown
0504 0002	Invalid block size (block mode only)
0504 0003	Invalid sequence number (block mode only)
0504 0004	CRC error (block mode only)
0504 0005	Out of memory
0601 0000	Unsupported access to an object
0601 0001	Attempt to read a write-only object

Table 48: Error messages arising from the Abort SDO protocol

Byte 7 – byte 4 Abort Code	Description
0601 0002	Attempt to write a read-only object
0602 0000	Object does not exist in the object dictionary
0604 0041	Object cannot be mapped to the PDO
0604 0042	The number and length of the objects to be mapped would exceed PDO length
0604 0043	General parameter incompatibility reason
0604 0047	General internal incompatibility in the device
0606 0000	Access failed due to an hardware error
0607 0010	Data type does not match, length of service parameter does not match
0607 0012	Data type does not match, length of service parameter too high
0607 0013	Data type does not match, length of service parameter too low
0609 0011	Sub-index does not exist
0609 0030	Value range of parameter exceeded (only for write access)
0609 0031	Value of parameter written too high
0609 0032	Value of parameter written too low
0609 0036	Maximum value is less than minimum value
0800 0000	General error
0800 0020	Data cannot be transferred or stored to the application
0800 0021	Data cannot be transferred or stored to the application because of local control
0800 0022	Data cannot be transferred or stored to the application because of the present device state
0800 0023	Object dictionary dynamic generation fails or no object dictionary is present (e.g. object dictionary is generated from file and generation fails because of a file error)

13.2.6 Examples of SDOs

Four examples of SDOs are listed below in hexadecimal format. These examples show the operation of SDOs on the CAN telegram level and can prove helpful when the SDO protocol is to be implemented on a CAN PC card.

Explanation of the following telegram formats:

M -> BK:
A request telegram is sent from the master to the fieldbus coupler.

BK -> M:
A reply telegram is sent from the fieldbus coupler to the master.

D:
Data frame

R:
RTR frame

Data Bytes 0 – 7:
Up to eight data bytes can be transferred in a CAN telegram. The individual bytes are separated from each other by blank spaces. Entries with value "XX" are insignificant but must be present. The values should be set to 0 for better understanding. The "DD" values in the reply telegram of the fieldbus coupler contain configuration-dependent data.

Reading Index 0x1000, Sub-Index 0; "Device Type"

Index 0x1000 has a data width of 4 bytes. The "Expedited Transfer Mode" SDO is used for transfer.

Table 49: Read index 0x1000

Direction	COB ID	Frame Type	Data Bytes 0 – 7
M->BK	0x601	D	0x40 00 10 00 XX XX XX XX
BK->M	0x581	D	0x43 00 10 00 91 01 DD 00

Explanation of data in reply telegram of fieldbus coupler:

Data bytes 4 and 5: 91 01 sequence low-byte, high-byte turn:
0x0191 = 401 "Device Profile Number"

Data bytes 6 and 7: DD 00 sequence low-byte, high-byte turn

Reading Index 0x1008, Sub-Index 0; "Manufacturer Device Name"

Index 0x1008 has a data width of 8 bytes. The "Normal Transfer Mode" is used for transfer. In this case, three telegrams are sent for each CANopen device.

Table 50: Read index 0x1008

Direction	COB ID	Frame Type	Data Bytes 0 – 7
M->BK	0x601	D	0x40 08 10 00 XX XX XX XX
BK->M	0x581	D	0x41 08 10 00 08 00 00 00
M->BK	0x601	D	0x60 XX XX XX XX XX XX XX
BK->M	0x581	D	0x01 37 36 37 2D 32 35 30
M->BK	0x601	D	0x70 XX XX XX XX XX XX XX
BK->M	0x581	D	0x1D 31 00 00 00 00 00 00

In the first reply telegram of the fieldbus coupler, the amount of data to be transferred is communicated to the master (0x00000008 byte). In the second and third telegram, the fieldbus coupler supplies the item number in ASCII format (hexadecimal format): "767-2501."

Reading Index 0x5300, Sub-Index 1; Input Data of Fieldbus Coupler

The process value of the fieldbus coupler's digital inputs is in index 0x5300, sub-index 1. The 8 bits are combined into a byte and can be read in sub-index 1.

In this example, the process value of the fieldbus coupler's digital inputs is read:

Table 51: Read index 0x5300

Direction	COB ID	Frame Type	Data Bytes 0 – 7
M->BK	0x601	D	0x40 00 53 01 XX XX XX XX
BK->M	0x581	D	0x4F 00 53 01 02 XX XX XX

The reply telegram of the fieldbus coupler contains the process value of the digital inputs in the 5th byte. In this example, the 2nd bit is set.

Describing Index 0x6200, Sub-Index 1;
First Digital Output Block, 8 Bit

The output values of the digital output modules are stored in index 0x6200. Every 8 bits are assigned to a group and can be read and written starting in sub-index 1. In this example, value 0xFF is written into the outputs of the first digital input block (8-bit).

Table 52: Describe index 0x6200

Direction	COB ID	Frame Type	Data Bytes 0 – 7
M->BK	0x601	D	0x2F 00 62 01 FF XX XX XX
BK->M	0x581	D	0x60 00 62 01 XX XX XX XX

The outputs of the digital output modules are set.

13.3 Node Guarding

"Node Guarding" starts for the fieldbus coupler when the first Remote Transmit Request telegram (RTR) is received with the COB ID for "Node Guarding" (0x700+moduleID). If the fieldbus coupler receives no corresponding telegram, it does not monitor the "Node Guarding."

In the delivery status, "Node Guarding" is turned off because a 0 is entered in the corresponding indices (0x100C = "Guard Time"; 0x100D = "Life Time Factor").

The NMT master prompts the fieldbus coupler at regular time intervals. This period is called the "Guard Time" (index 0x100C). The internal state of the fieldbus coupler can be found in the reply telegram.

If an RTR request arrives and the "Guard Time" has not been set, the "Node Guarding" is not monitored; however, the fieldbus coupler does respond with its internal state.

The states are coded as follows:

Node State	Value
Pre-operational	127
Operational	5
Stopped	4

The "Life Time" is the product of the guard time (index 0x100C) and the life time factor (index 0x100D).

"Node Guarding" Failure

If the fieldbus coupler does not receive a node guarding telegram within the "Life Time," it sends the emergency telegram with the error code

"Error Code": 0x8130,
"Error Register": 0x11,
"Manufacturer-Specific Error Code": 0x00 04 00 000 0.

The I/O module outputs are set according to the settings in objects 0x6206, 0x6207, 0x6443 and 0x6444. The fieldbus coupler switches to the CANopen state according to the configuration in object 0x1029 or 0x67FE.

As soon as the node guarding protocol resumes, an emergency telegram ("Error Code": 0x0000; "Error Register": 0x11; "Manufacturer-Specific Error Code": 00 04 00 000 0) is transmitted to indicate that "Node Guarding" is active again. The outputs and the state of the fieldbus coupler remain unchanged.

Only the node guarding protocol or the heartbeat protocol may be used. The heartbeat protocol is always used if the "Heartbeat Producer Time" is configured.

13.4 Heartbeat Monitoring

This protocol makes it possible to carry out module monitoring without using RTR frames.

The heartbeat generator generates a telegram in a cyclical manner (time interval defined in object 0x1017) from the fieldbus coupler, in which it transfers the state of the I/O module. The transfer begins after configuration of object 0x1017. The telegram can be analyzed by one or several "Heartbeat Consumers" (object 0x1016). Up to five I/O modules can be monitored simultaneously. The monitoring begins with the first arrival of a heartbeat telegram (for each monitored I/O module separately).

"Heartbeat" Failure:

A blink code is emitted if no corresponding heartbeat telegram is received within the configured time (object 0x1016). An emergency telegram is likewise transmitted ("Error Code": 0x8130; "Error Register": 0x11; "Manufacturer-Specific Error Code:: 0x00 05 KK 00 00; KK: station address). The outputs are set according to objects 0x6206, 0x6207, 0x6443 and 0x6444, and the fieldbus coupler enters the defined state according to object 0x1029 or the identical object 0x67FE.

As soon as the heartbeat protocol resumes, an emergency telegram ("Error Code": 0x0000; "Error Register": 0x11; "Manufacturer-Specific Error Code": 0x00 05 KK 00 00) is transmitted to indicate that heartbeat monitoring is active again. The outputs and the state of the fieldbus coupler remain unchanged. If several I/O modules are monitored, the blink code for the heartbeat failure only stops when the last heartbeat telegram resumes.

Only the node guarding protocol or the heartbeat protocol may be used. The heartbeat protocol is always used if the "Heartbeat Producer Time" is configured.

13.5 Synchronization Object, SYNC

This object enables the synchronization of all network devices. Through the corresponding PDO configuration, you can cause the network device to process its data or update its outputs upon arrival of a SYNC object.

Sending the SYNC object in a cyclical manner ensures that all network devices process data simultaneously.

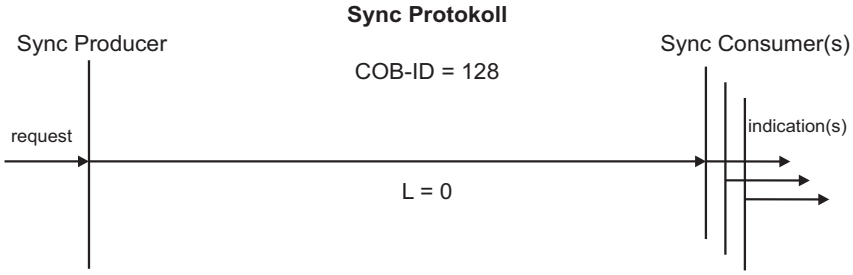


Figure 37: Sync protocol

13.5.1 SYNC Monitoring

If the "Communication Cycle Period" value does not equal 0, and if the fieldbus coupler is in the operational state, monitoring takes place at the first arrival of a SYNC telegram.

Failure of the SYNC Telegram:

A blink code is emitted if no SYNC telegram is received within the monitoring time ("Communication Cycle Period"). A state change does not take place. An emergency telegram is likewise transmitted ("Error Code": 0x8100; "Error Register": 0x81; "Manufacturer-Specific Error Code": 00 04 00 00 00). If the master causes a state change to come about, the failure of the SYNC telegram is indicated nonetheless.

The LEDs indicate the normal operational state only after the SYNC telegram has been received again in the operational state. Another emergency telegram ("Error Code": 0x0000; "Error Register": 0x81; "Manufacturer-Specific Error Code": 00 04 00 000 0) is transmitted to indicate that the SYNC monitoring is functioning once more.

13.6 Emergency Object (EMCY)

Emergency objects are triggered by an internal error situation; e.g., a S-BUS disruption or an I/O module error. The fieldbus coupler then sends an emergency object to all connected CANopen devices (broadcast) to notify them of the error. The CANopen devices can then take any necessary error correction measures.

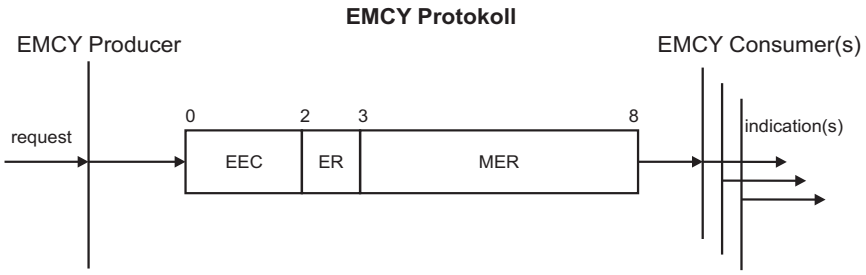


Figure 38: EMCY protocol

13.7 Network Management Protocol

The NMT master can use these protocols to control the state of the NMT slave. The initialization, pre-operational, operational and stopped states are defined. It is possible to convert all CANopen devices or each one individually to another state using a command.

13.7.1 Start Remote Node

This service is used to convert the NMT slave (fieldbus coupler) to the operational state.

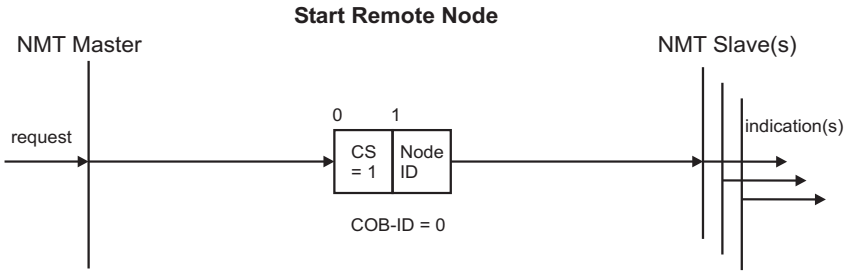


Figure 39: Start remote node

Node ID = 0: All available CANopen devices are converted to the operational state.

13.7.2 Stop Remote Node

This service is used to convert the NMT slave (fieldbus coupler) to the stopped state.

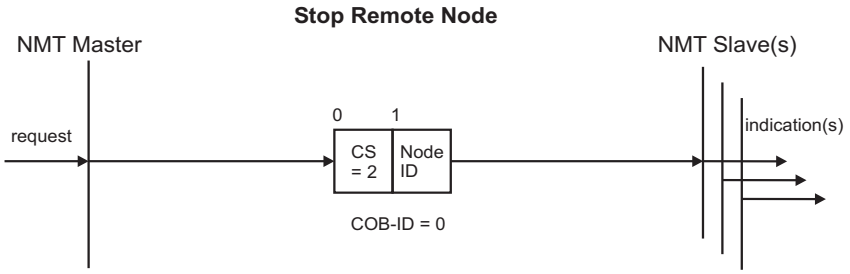


Figure 40: Stop remote node

Stop Remote Node

Node ID = 0: All available CANopen devices are converted to the stopped state.

13.7.3 Enter Pre-Operational

This service is used to convert the NMT slave (fieldbus coupler) to the pre-operational state.

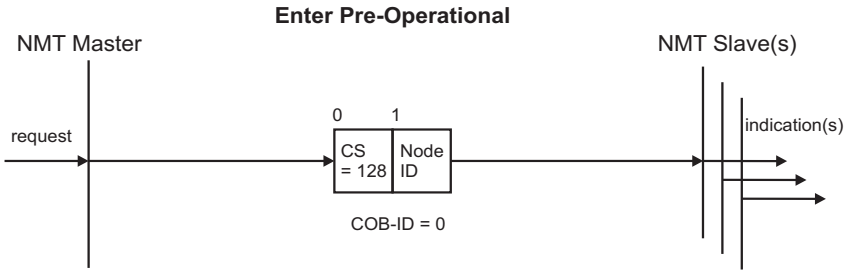


Figure 41: Enter pre-operational

Node ID = 0: All available CANopen devices are converted to the pre-operational state.

13.7.4 Reset Node

This service is used to reset the NMT slave (fieldbus coupler).

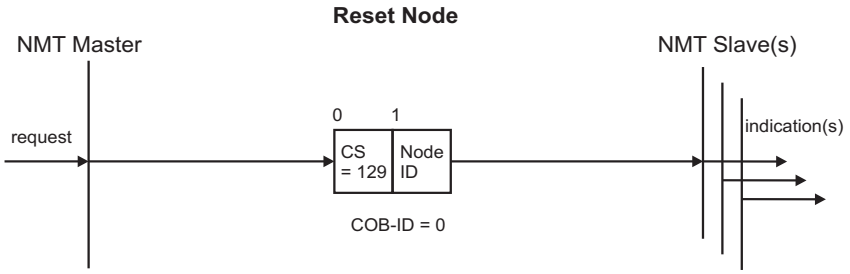


Figure 42: Reset node

Node ID = 0: All available CANopen devices are reset.

13.8 Error Control Protocols

This protocol can assist in detecting errors in the network. In this way, the master can test whether a CANopen device is still in the designated state or if it has been switched to a different state (e.g., by a reset).

13.8.1 Node Guarding Protocol

By means of the "Node Guarding," the NMT slave is prompted via an RTR frame in a cyclical manner to send its current state. An additional bit switch determines whether the NMT slave is still functioning correctly.

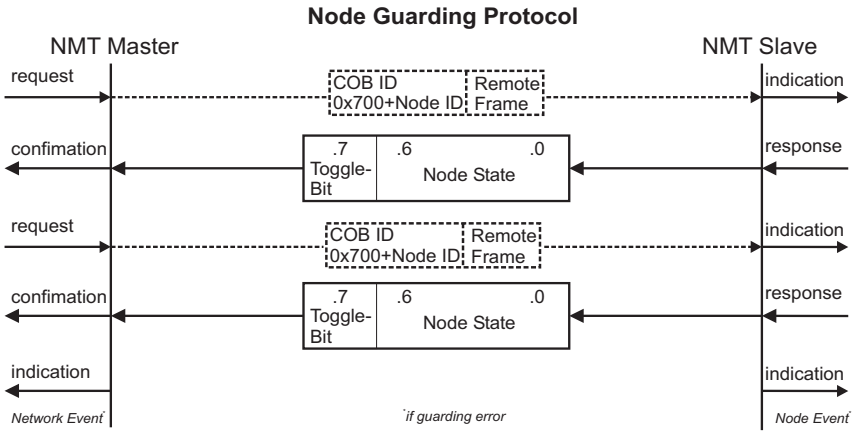


Figure 43: Node guarding protocol

13.8.2 Heartbeat Protocol

This protocol enables monitoring without RTR frames. A heartbeat generator generates a heartbeat telegram in a cyclical manner that is received by all. In the heartbeat telegram, the current state of the generator is coded.

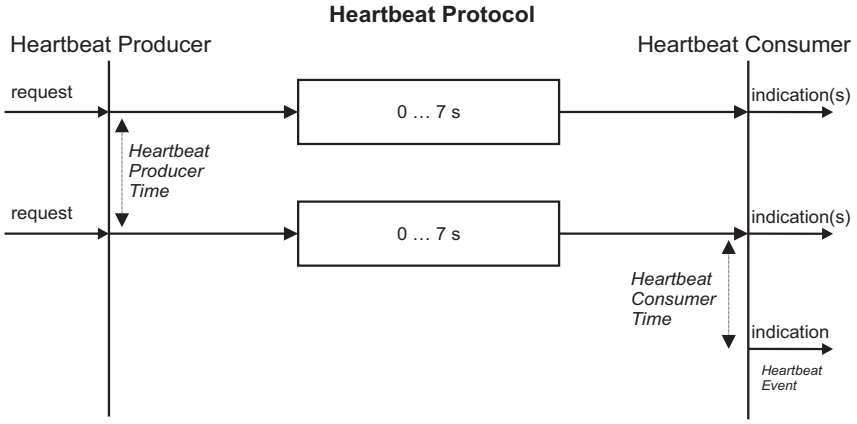


Figure 44: Heartbeat protocol

13.8.3 Bootup Protocol

This protocol indicates that the NMT slave has switched from the initialization state to the pre-operational state. The "Bootup Protocol" is executed following a hardware or software restart or following the "Reset Node" service.

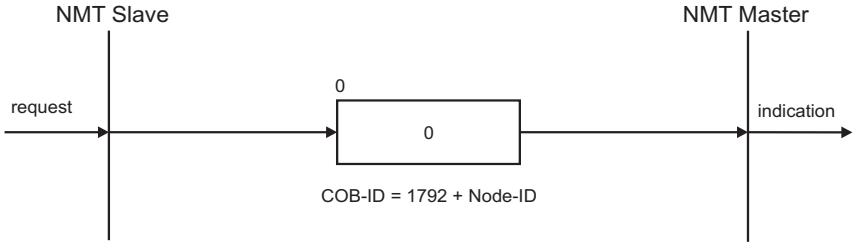


Figure 45: Bootup protocol

14 File System/User Administration

Two partitions of the file system are available to the user including a RAM disk and a partition in Flash memory.

Table 53: Summary of file system partitions

Partition	Format	Type	Size	Use
R:\	FAT 12	RAM disk	1MB	-
P:\	FAT 12	Flash disk	1MB	CoDeSys programm

The R:\ partition is designed as nonpermanent RAM disk and can be used to temporarily store data. These files are lost when the system is restarted. To save files permanently, use partitions "P:\".
CoDeSys development environment uses the P:\ partition to save program-relevant data.

User Administration

Access to the fieldbus coupler’s data via CoDeSys is password protected. In the default status, the following users and access authorizations are defined:

Table 54: User administration

User name	Password	Rights
admin	wago	Full access
user	user	Limited access
guest	guest	Read access

Use WBM and WAGOframe to change the passwords.

15 CoDeSys 3 Runtime Environment

This section contains information on configuring the gateway and using CoDeSys (WAGO-OEM Setup, item no. 759-915) to create a project.

15.1 Installing the CoDeSys Programming System

CoDeSys 3 is based on Microsoft Plug-In technology, whereby different versions (profiles) of CoDeSys can be installed in one directory. In the first installation, the Microsoft Framework 2.0 and all plug-ins associated with this profile are installed. For each subsequent installation, merely a new profile is created and any new plug-ins registered.

The following WAGO-specific expansions are also included in the installation:

- Project templates
- Device description
- Libraries
- USB drivers

The WAGO project templates already contain the most important elements of an application program and save time when creating a new project. The device descriptions contain all device-specific information for the WAGO 767 product series.

To install the CoDeSys 3 programming software on the fieldbus coupler, please proceed as follows.

1. Insert the CoDeSys 3 CD into your CD-ROM drive.

2. To install the programming system, follow the instructions that appear on your screen. A successful installation is indicated by a CoDeSys icon on your desktop.

15.2 Editing the Gateway Configuration for CoDeSys 3

Requirement: The fieldbus coupler must be switched on and your PC must be connected with the USB interface. For more information, see Section 6.7.

To edit the gateway for serial communication, proceed as follows:

1. Double-click on the "Remove Hardware" icon in the taskbar to display the existing COM ports.

Figure 46: "Remove Hardware" icon in the taskbar

Assigned COM ports are not displayed unless the "Display Device Components" selection box has been selected.

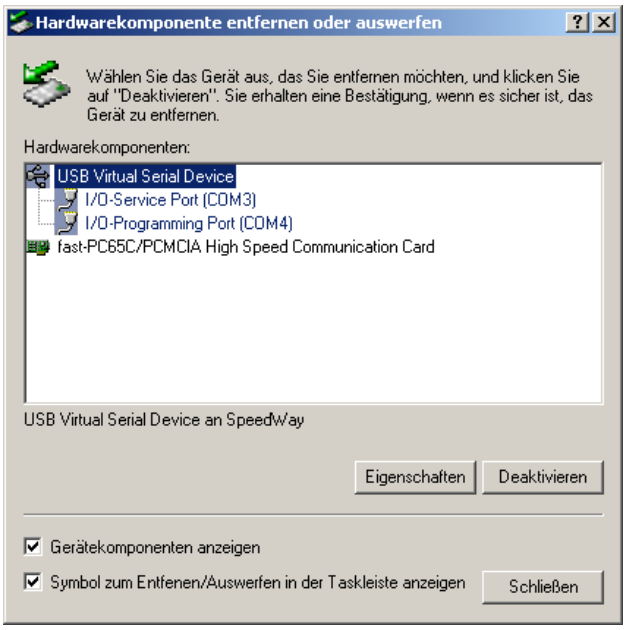


Figure 47: "Remove or Eject Hardware Components" dialog box

2. Note the COM port of the "I/O-Programming" gateway. This is COM port 4 in this example.

3. Close the dialog box by clicking **[Close]**.

Note

This COM port number remains constant for the fieldbus coupler currently connected and for the USB interface currently being used. When you replace the fieldbus coupler or the USB slot on your PC, the COM port also changes.

After installing CoDeSys, you must edit the ComPort= entry in the "Gateway.cfg" configuration file to conform to the local requirements.

To do this, open the "Edit Gateway.cfg" link that was created on the desktop during CoDeSys installation.

Figure 48: Edit Gateway.cfg

If this link is not available on your desktop, the configuration file can typically be found under **Programs > 3S > CoDeSys > GatewayPLC**. Open the Gateway.cfg configuration file with an editor.

1. Prefer to section `[CmpBlkDrvSimpleCom]` for `ComPort=`.

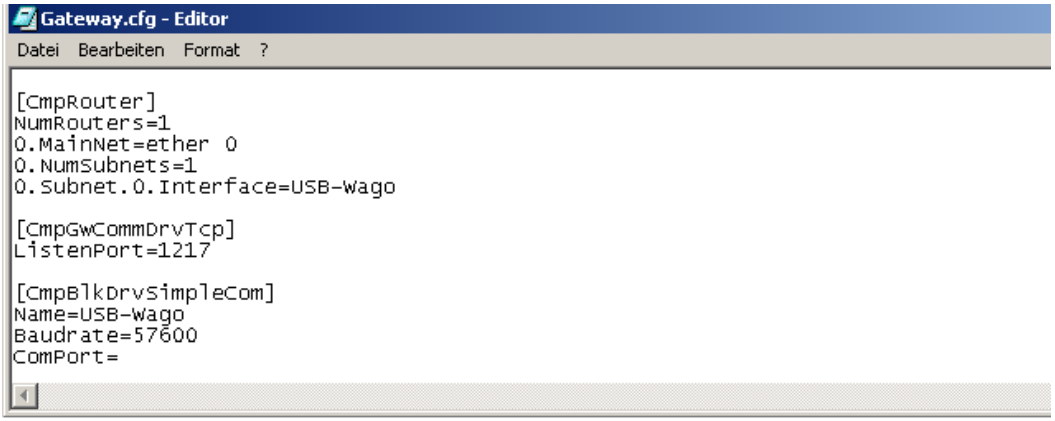


Figure 49: "Gateway.cfg" configuration file

2. Enter the COM port number of the connected fieldbus coupler after `ComPort=`. In this example, the number is 4.

   [CmpBlkDrvSimpleCom]
   Name=USB WAGO
   Baudrate=57600
   ComPort=**4**

3. Overwrite the modified configuration file as follows: Select **File > Save As...**. The **Save File As** dialog box opens. In the **File Type** field, select "All files" to maintain the ".cfg" file ending. Click **[Save]** to overwrite.

15.3 Programming with CoDeSys 3

This Section uses an example of the WAGO project template to explain the relevant steps in creating a CoDeSys project.

This Section is meant to be a quick start guide and does not cover the entire scope of CoDeSys functions. A detailed description of all features can be found on the CoDeSys CD-ROM (759-915), obtainable from WAGO.

15.3.1 Start the CoDeSys Programming System

To start CoDeSys, double-click on the CoDeSys icon on your desktop or navigate to the program via your operating system's Start menu as follows:
Start > **Programs** > **3S CoDeSys** > **CoDeSys** > **CoDeSys V3**.

15.3.2 Creating a CoDeSys Project

In the WAGO-specific CoDeSys installation, a project template already exists in which the settings are selected in such a way that a PLC program can be developed on the fieldbus coupler.

1. In the menu bar, click on **File** and select **New Project...**. The "New Project" dialog box opens.

2. To open the WAGO project template, click on the "WAGO" folder in the "Categories" field.

3. Click on the "767-2501 Project" icon in the "Templates" field.

4. In the "Name" input field, enter a project name.

5. In the "Location" input field, indicate the location where you wish to save the project by clicking on the **[...]** button.

6. Click **[OK]** to confirm your entry.

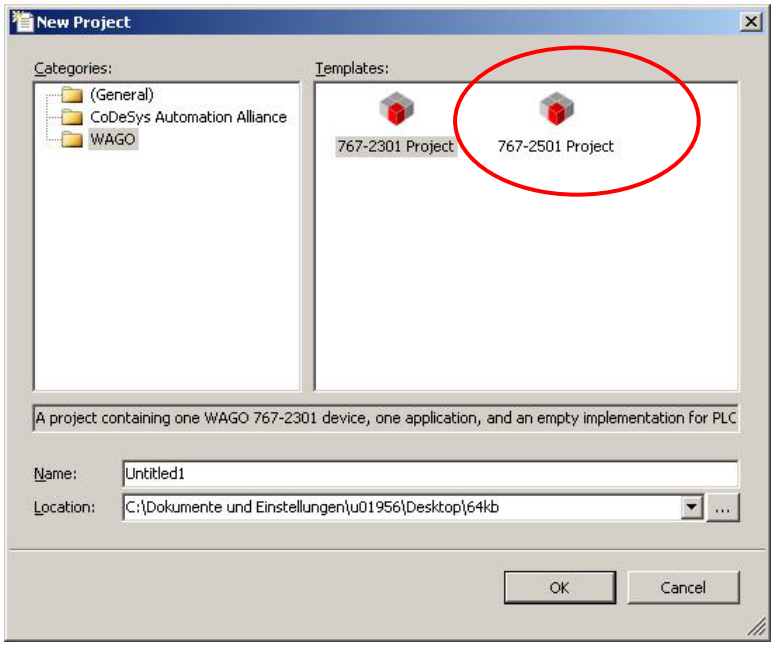


Figure 50: "New Project" dialog box

You can achieve the same results if you begin with a blank project and then manually add the necessary objects.

15.3.3 Adding Library Management

In the library management, all libraries are listed whose function blocks can be used in your project.

You can also use other library management objects to more precisely define the application area of the libraries. For the sake of simplicity, we will confine ourselves for this example to a general library management. If needed, use the online assistance or the manual for CoDeSys 3.

1. In the "POUs" window, right-click on the project name that you have assigned. In this example, the name is "FirstProject".

2. In the context menu, select **Add Object...**. The "Add Object" dialog box opens.

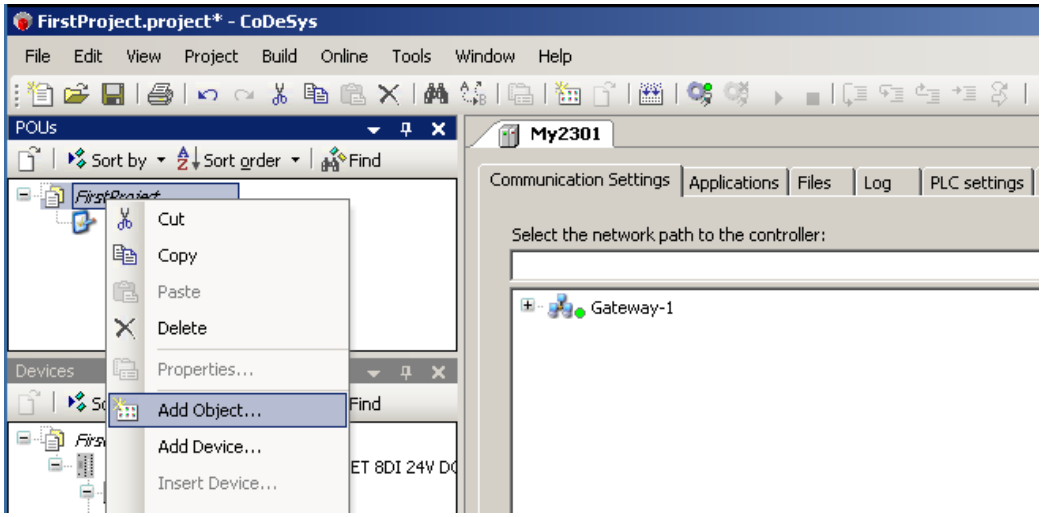


Figure 51: "Add Object" dialog box

3. In the "Add Object" dialog box, click on the "Library Manager" option.

4. Click **[Open]** to add the library manager.

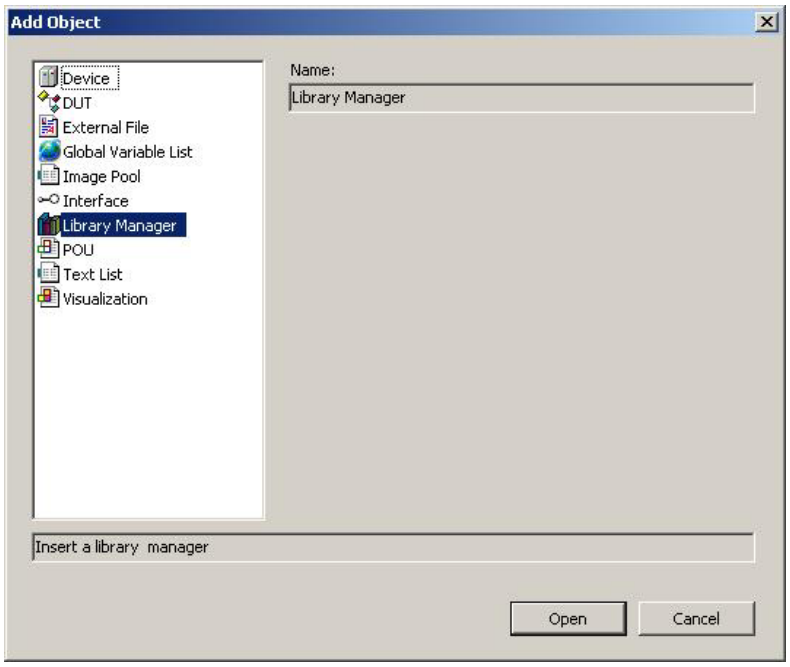


Figure 52: "Add Object" dialog box with the "Library Manager" selection

5. The library management opens.

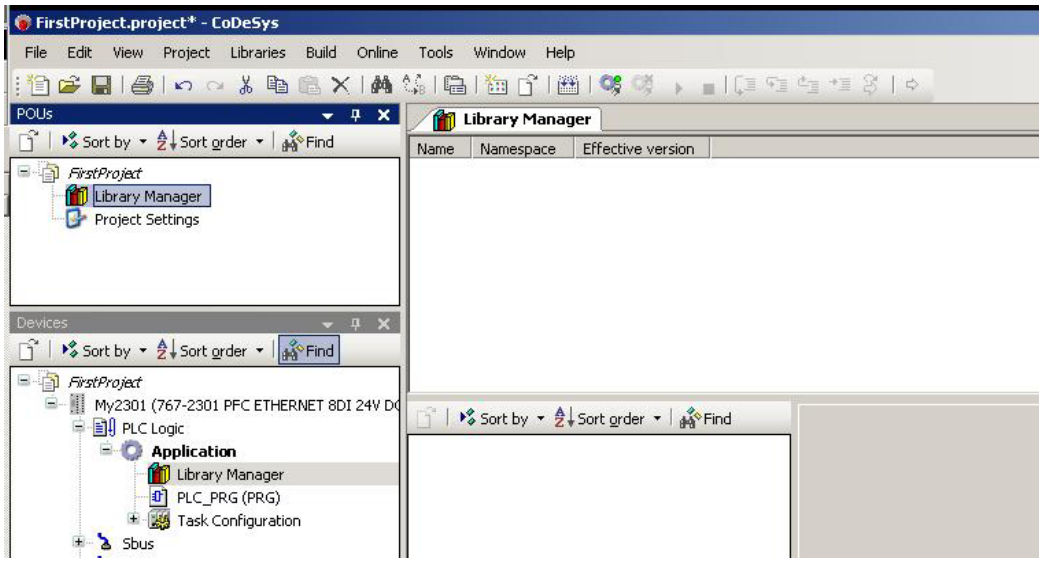


Figure 53: "Library Manager" view

15.3.4 Adding Libraries

In order to use the function blocks of a library in the PLC program, you must add this library in the library management.

Note

In order to use a function block whose library has not yet been referenced, this step is carried otu during program processing.

To add a reference to the "Standard" library in the library management, proceed as follows:

1. Click on **Libraries** in the menu bar and select **Add Library**. The "Add Library" dialog box opens.

2. In the "Add Library" dialog box, click on "Common" and select the "Standard" library.

3. Click **[OK]** to submit your selection.

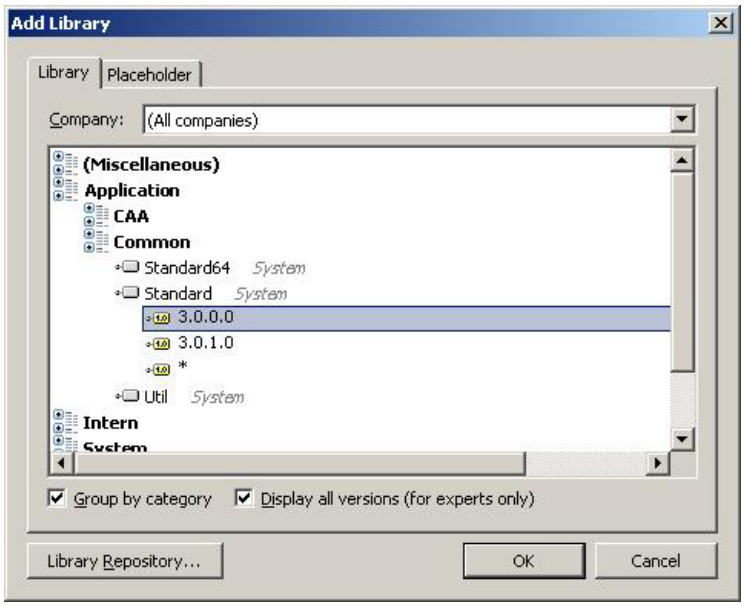


Figure 54: "Add Library" dialog box with the "Standard" selection

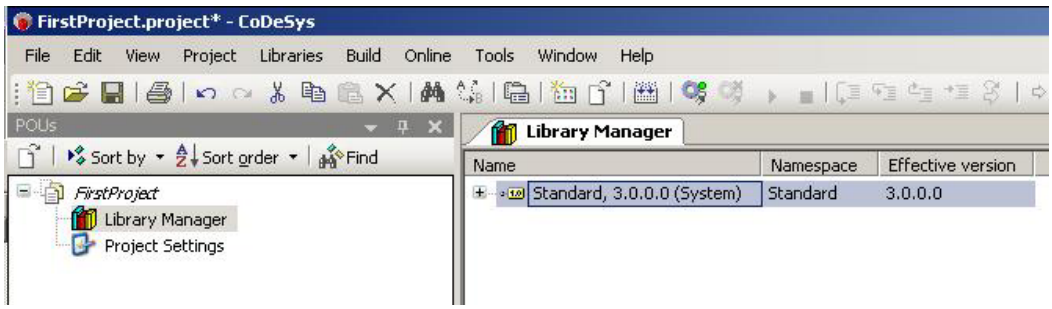


Figure 55: "Library Manager" view with the "Standard 3.0.0.0" library.

15.3.5 Creating a Control Program

A control program (POU) needs to be created for the project. POU stands for "Program Organizational Unit". This control program is created in an IEC 6131-3 programming language (AWL, FUP, etc.)

In the WAGO project template, a PLC program already exists with the name PLC_PRG (PRG). This program will be used for this description.

1. Right-click on the project name that you have assigned. In this example, the name is "FirstProject".

2. In the context menu, select **Add Object...**. The **"Add Object" dialog box** opens.

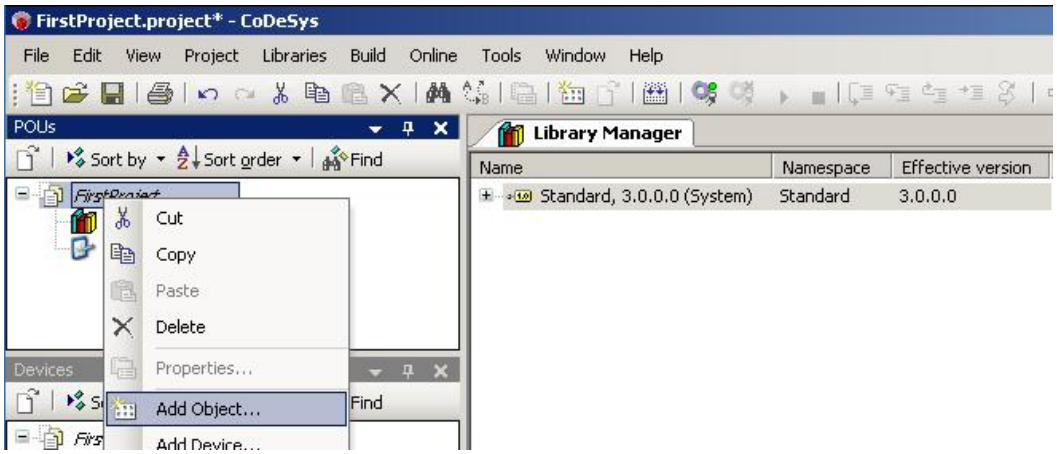


Figure 56: Creating a control program

3. In the "Add Object" dialog box, click on the "POU" object.

4. In the "Name" field, enter a name for your control program; e.g., FirstProgram.

5. In the "Type" field, select the **Program** option.

6. From the "Implementation Language" menu, select a programming language in which the PLC program should be created.

7. Click **[Open]** to confirm your selection.

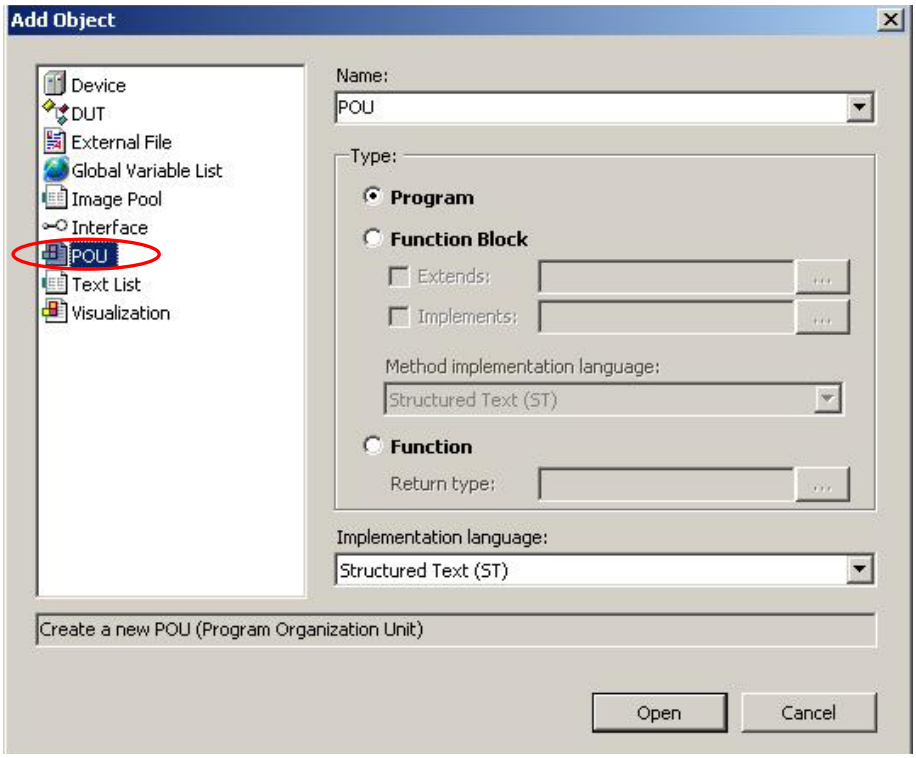


Figure 57: "Add Object" dialog box with the "POU" selection

8. To open the editor of your selected programming language, double-click on your program names. For further questions on Semantik, use the online assistance or the manual for CoDeSys 3.x.

15.3.6 Creating a Task Configuration

The task configuration is used to configure all tasks of the controller.

A task configuration (named "Task Configuration") already exists in the WAGO project template. This Section is therefore only necessary if you wish to add another controller in your project. This section is only necessary if you wish to create a new project not based on the WAGO project template.

A task management object is created as follows:

1. Right-click on "Application" in the "Devices" window. In the context menu, select "Add Object". The "Add Object" dialog box appears.

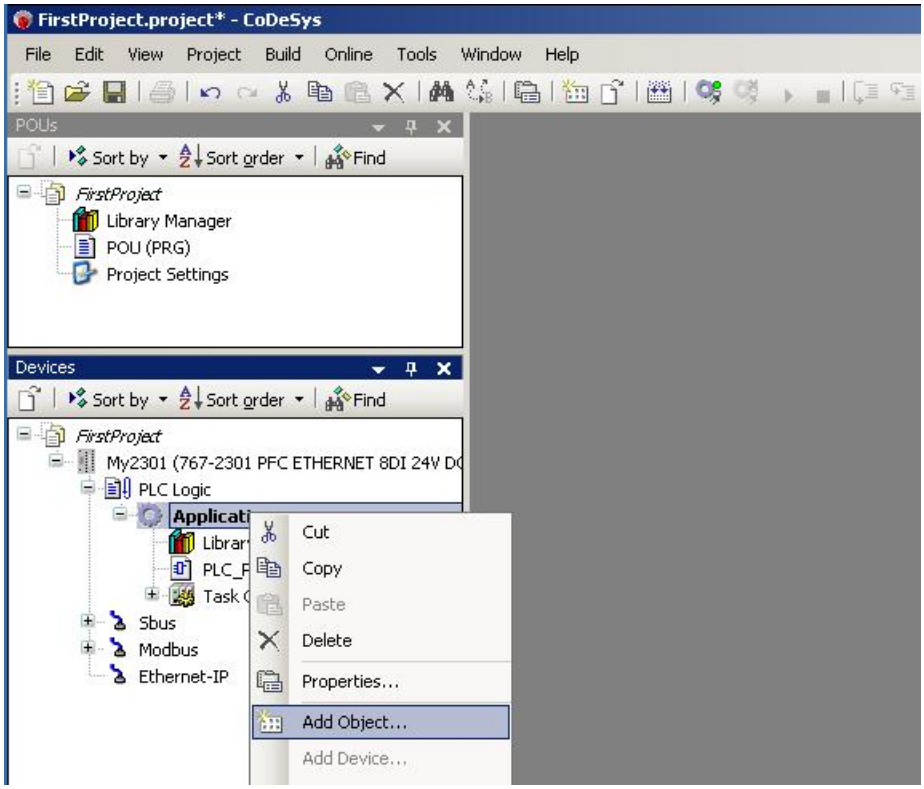


Figure 58: Creating a task management object

2. In the "Add Object" dialog box, select the "Task Configuration" object type.

3. Click **[Open]** to confirm your selection.

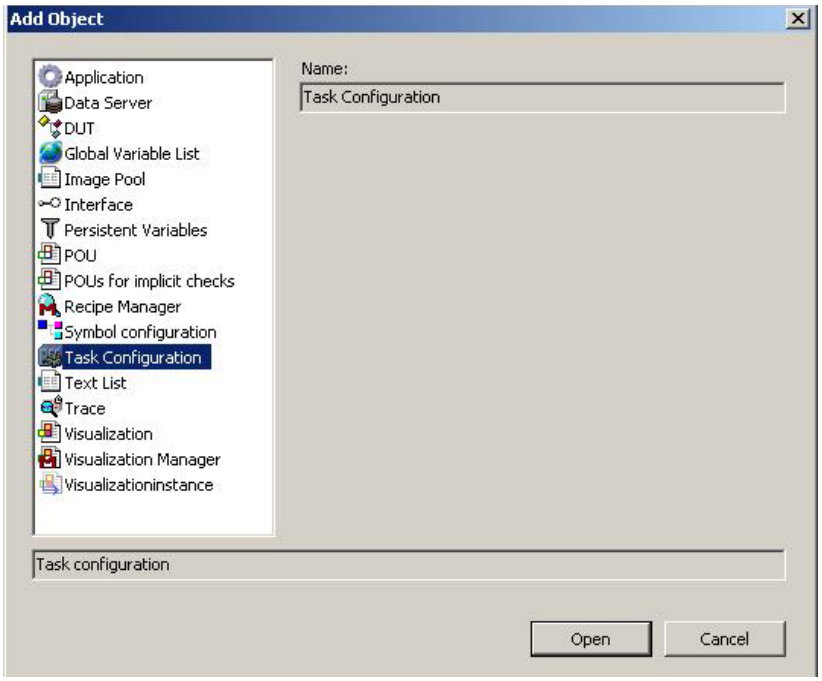


Figure 59: "Add Object" dialog box with the "Task Configuration" selection

15.3.6.1 Creating a Task

Create the task that will later access programs which are to be executed by the fieldbus coupler. A task can trigger program access in either a cyclical or an event-oriented manner.

Divide your automation task into several tasks with differing cycle times in order to complete automation tasks at different times.

1. To create a task, right-click on "Task Configuration" in the "Devices" window.

2. In the context menu, select **Add Object...**. The "Add Object" dialog box appears.

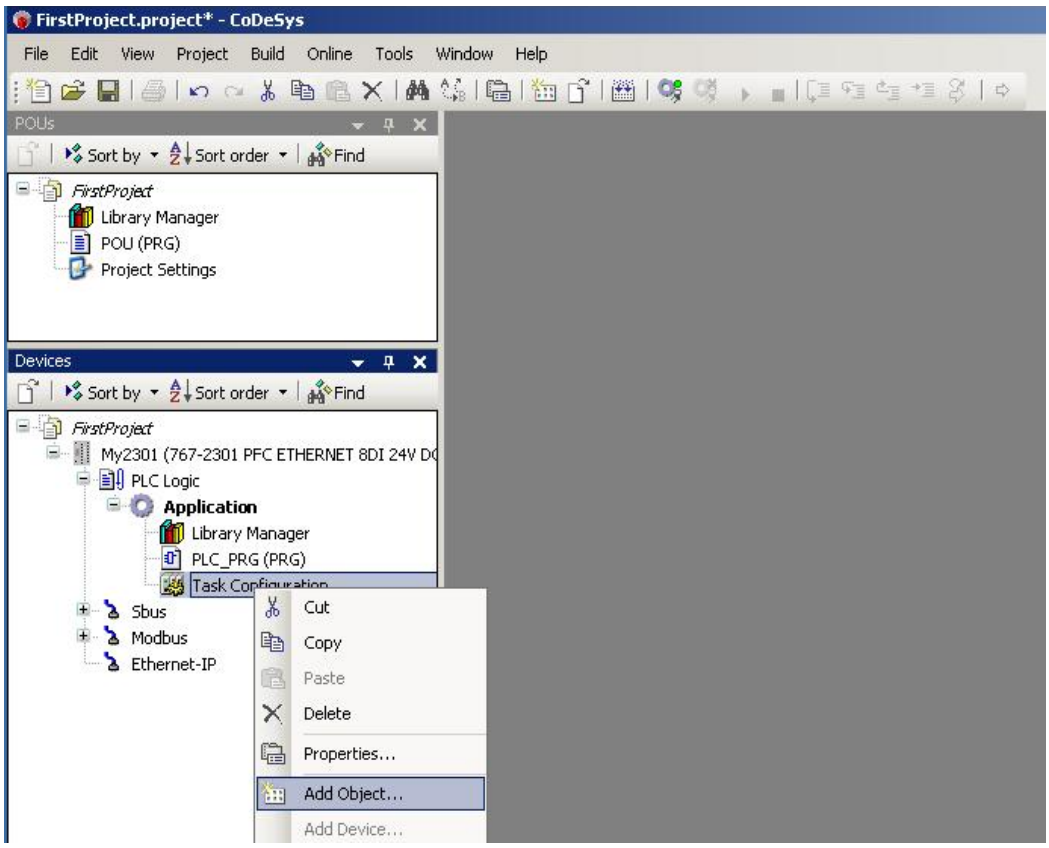


Figure 60: Creating a task

Note

The fieldbus coupler controller provides integrated monitoring of the CPU load. In case of an overload, the current application is closed. This exception is recorded in the "Log" tab under "My2501." The entry reads: "*Exception* WatchdogApplication<TaskName>."

3. Rename the task by assigning it a different name in the "Name" input field. In this example, the name is "myTask".

4. Click **[Open]** to submit your entry.

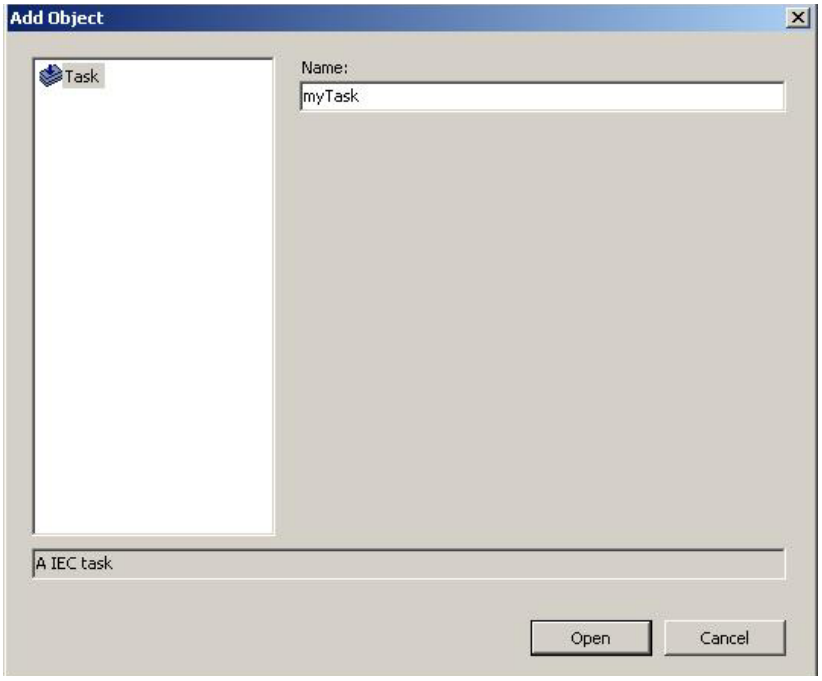


Figure 61: "Add Object" dialog box

15.3.6.2 Assignment between POU and Task

In the WAGO project template, the "PLC_PRG" POU is already added to a call task. A POU can be executed in several controllers. Ensure that the environmental parameters (e.g., comprehensive variables) of the various controllers are defined identically. The following steps are only relevant if you want to add another POU as call.

1. To open the task configuration input window, right-click on "myTask" in the "Device" pane.

2. Add a program to the task by clicking on the **[Add POU]** button. The "Input Assistance" dialog box appears.

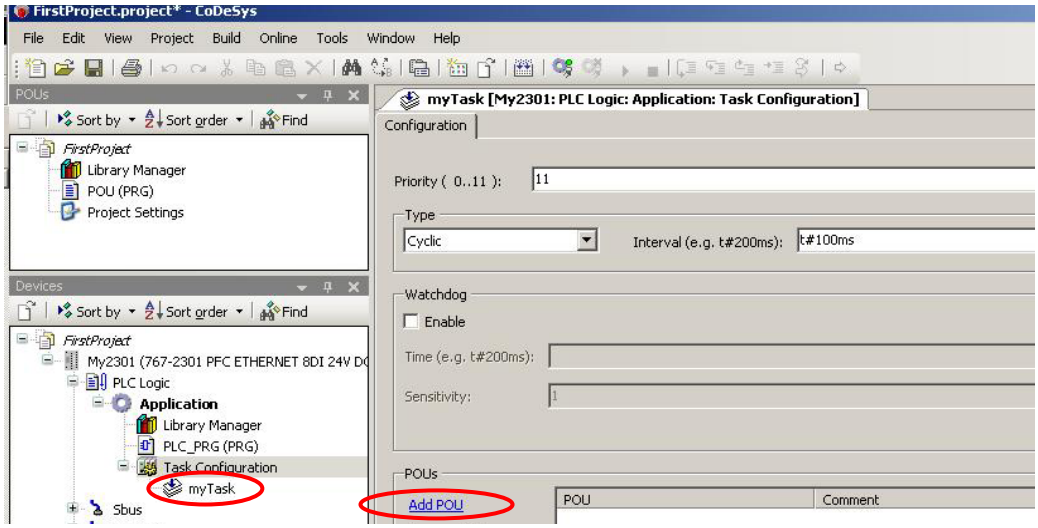


Figure 62: Assignment between POU and task

3. In the "Input Assistance" dialog box, select the "Program (Project)" option. In the "Elements" field, select the POU that should be added.

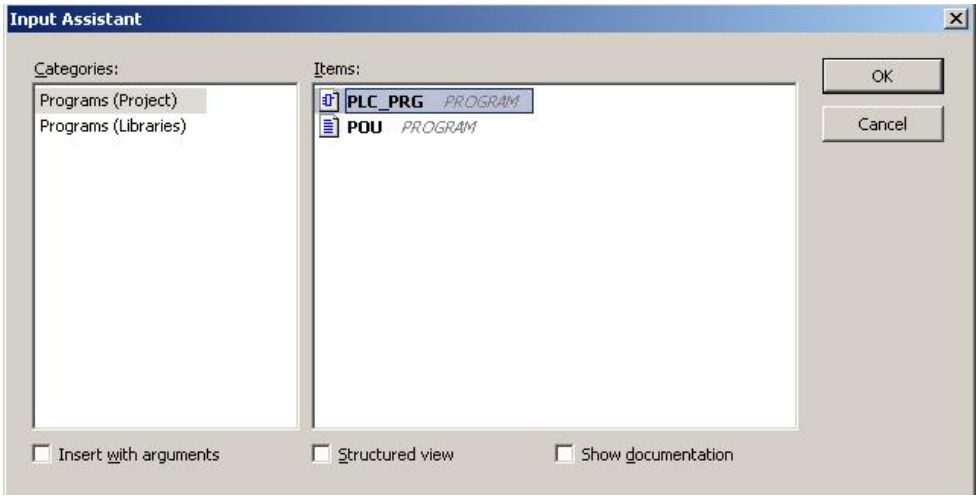


Figure 63: "Input Assistance" dialog box

4. Click **[OK]** to submit your entry.

15.3.7 Selecting the Communication Setting

Select here the controller (fieldbus coupler) on which your current project is to be executed. Connect the fieldbus coupler to your PC. The USB cable can be used for this (see Section 6.7).

Communication between CoDeSys and the fieldbus coupler is established by the CoDeSys gateway. The gateway is installed together with the installation of CoDeSys. The gateway icon appears in the system tray.

Figure 64: System tray on your desktop

1. Double-click on your device description in the "Devices" window. The controller description window appears. The gateway can be seen in the "Communication Settings" tab.

2. Browse the network for the fieldbus coupler by selecting "Gateway" and clicking on **[Scan network]**.

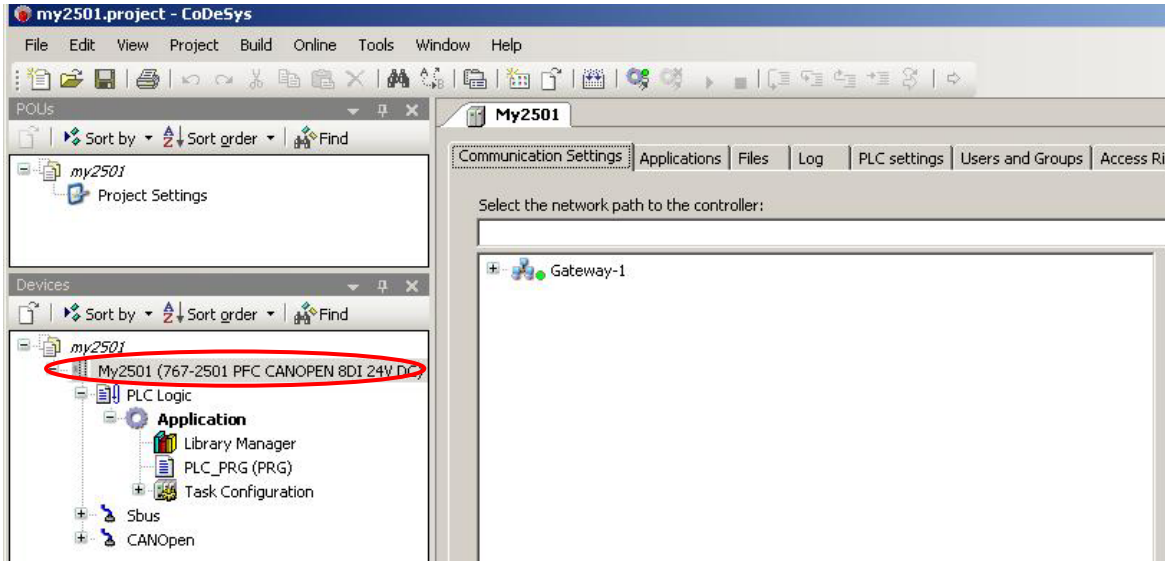


Figure 65: "Communication Settings" tab

3. Select the fieldbus coupler and click on **[Set active path]** to assign this to the configured fieldbus coupler.

15.3.8 Loading a PLC Program

CoDeSys is capable of managing several fieldbus couplers (controllers) in a project. Only one controller at a time can be loaded and started. Thus, you must define which application should be loaded. For the sake of simplicity, this example has only one application and one controller. Thus, the application in this example is always the "active application".

To load a PLC program, proceed as follows:

1. In the "Device" pane, right-click on the "Application" that you want to load, and select **Set Active Application** from the context menu.

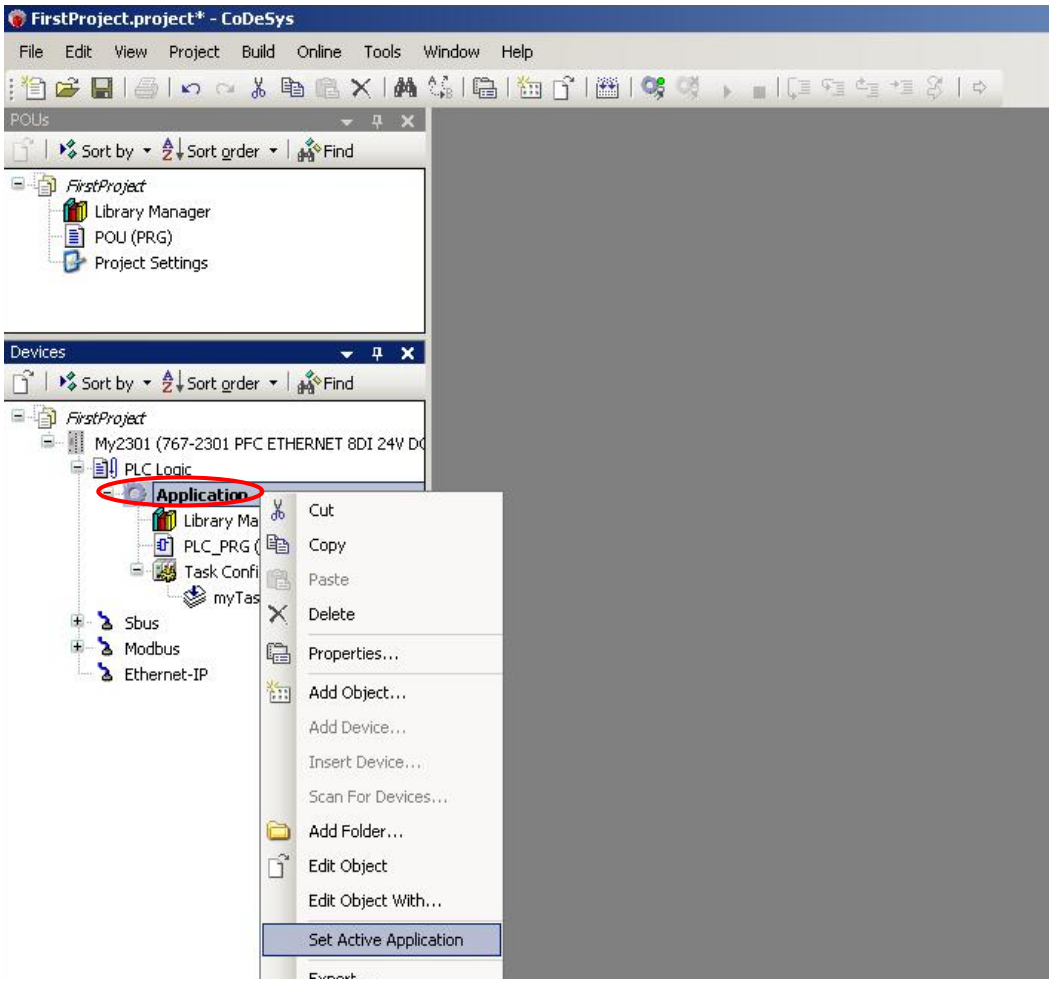


Figure 66: Loading a PLC program

The "Decode" and "Load" functions work on the "active application" of a controller. All programs that are assigned to the application by a task are decoded and loaded. Programs that are not assigned to a task are not decoded or loaded.

2. To load, click on **Online > Login** from the menu bar. This automatically generates the binary code for the controller. The following confirmation prompt appears. Click **[Yes]** to confirm.

Figure 67: Confirmation prompt

Note

To load the program, it must be decoded without errors. If errors exist in the program, click on the error message and correct the error. If needed, use the online assistance or the manual for CoDeSys 3.

3. To start the application that you have just loaded into the controller, select **Online > Start** from the menu bar.

In the online mode of the editor, you can view the online data of the PLC program and change the data, if needed. You can also disrupt the operation of the PLC program via breakpoints and proceed through the PLC program in individual steps. Additional information in this regard can be found in the online assistance or in the CoDeSys 3 manual.

4. Select **Online > Logout** from the menu bar.

5. To retain your intact application in the fieldbus coupler, select **Online > Create boot application for "Application"…**.

15.3.9 Adding I/O Modules

To access the process data of the I/O modules, you must first insert the I/O modules being used into the controller configuration. This does not require that you configure the entire physical structure of the 767 node. Simply establish the desired I/O modules in the CoDeSys project.

To attach the I/O modules, please proceed as follows:

1. In the "Devices" window, right-click on "Sbus" and select **Add Device...** from the context menu. The "Add Device" dialog box opens

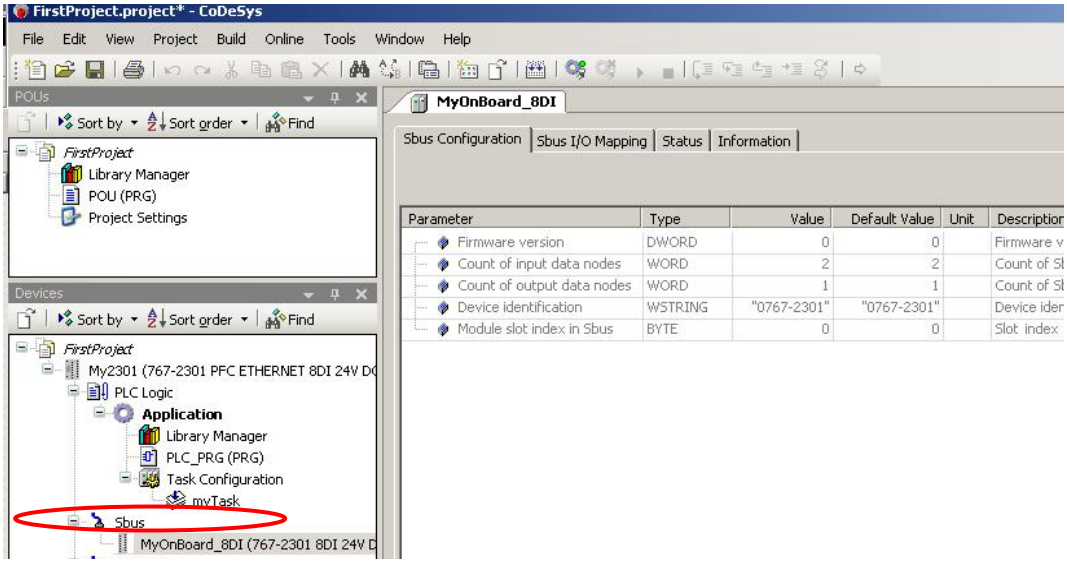


Figure 68: Adding I/O modules

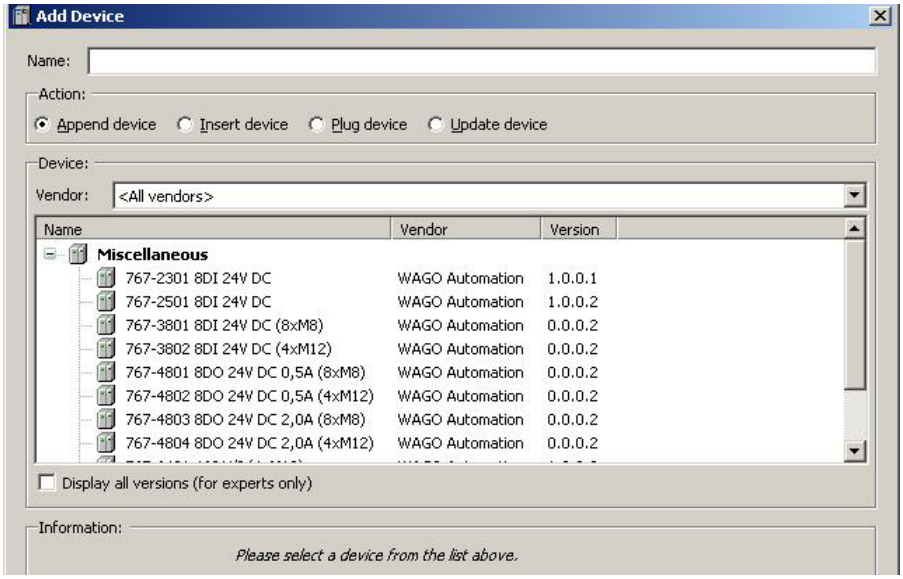


Figure 69: "Attach Device" dialog box

2. Choose the I/O module that you wish to configure by selecting the respective I/O module and clicking on **[Add Device]**. Click **[Close]** to close the dialog box.

3. To clearly define the inserted I/O module, you must assign a clear number to the "Module Slot Index in Sbus" parameter in the "SBUS Configuration" tab. This number indicates the position of the I/O module in the 767 node. For example, the number 4 is entered to access the fourth I/O module in the 767 node.

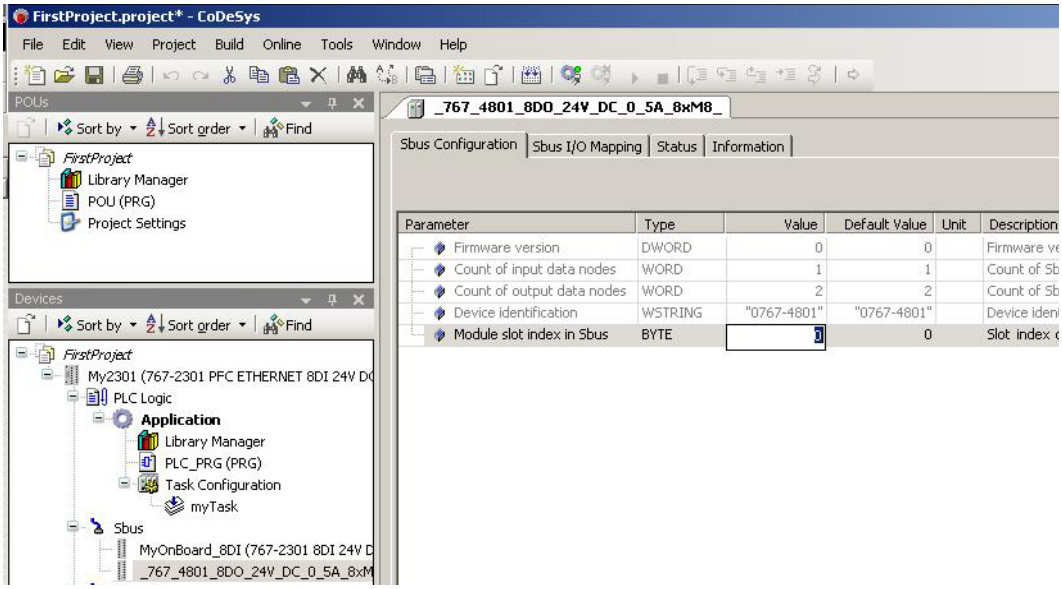


Figure 70: Selection of I/O modules in "Attach Device" dialog box

15.3.9.1 Assignment Between the Process Data of the I/O Modules and the Variables of the Control Program

In order to access the process data of the I/O modules, you must connect them with a variable. You have two options:

- Link pre-defined variables with the I/O modules
- Define new variables

Linking pre-defined variables with the fieldbus variables

1. To link variables, click on the "SBUS I/O Image" tab and open the "Output Channels" tree.

2. Double-click in the "Mapping" pane to choose between the options "Map to Existing Variable" and "Create New Variable." Select "Map to Existing Variable."

3. In the "Variable" pane, enter the name and path of a variable that you have created in CoDeSys. The following variable is contained in the WAGO project template: Application.PLC_PRG.iCount.

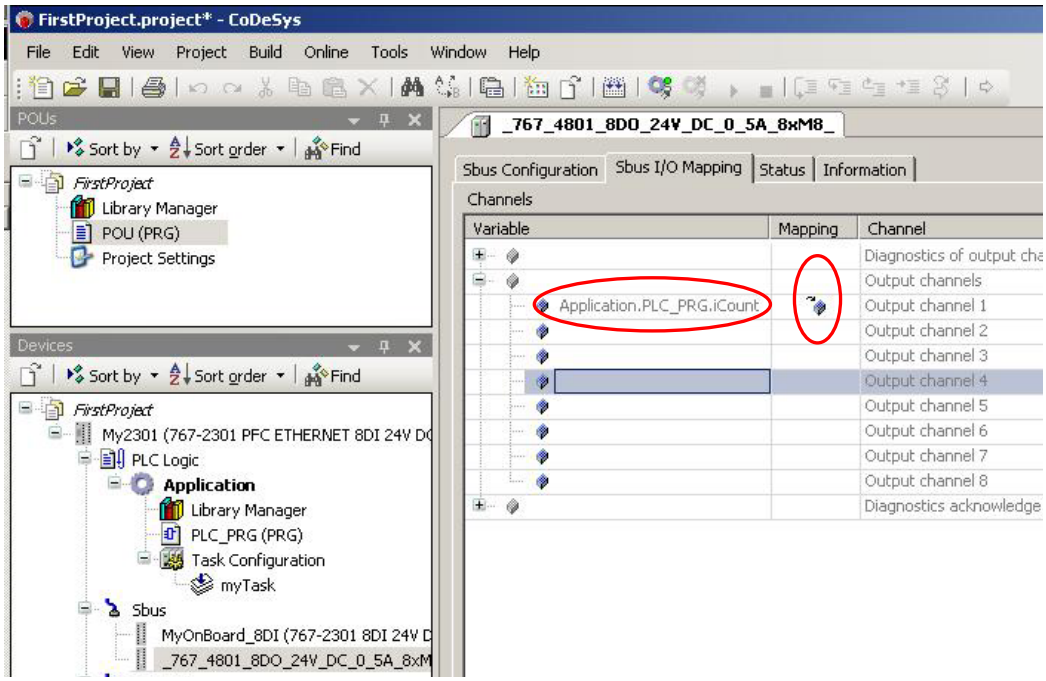


Figure 71: Link variables used in the program using inputs and outputs

Defining New Variables

1. To define variables, click on the "SBUS I/O Image" tab and open the "Output Channels" tree.

2. Double-click on the icon in the "Mapping" pane to choose between the options "Map to Existing Variable" and "Create New Variable." Select "Create New Variable."

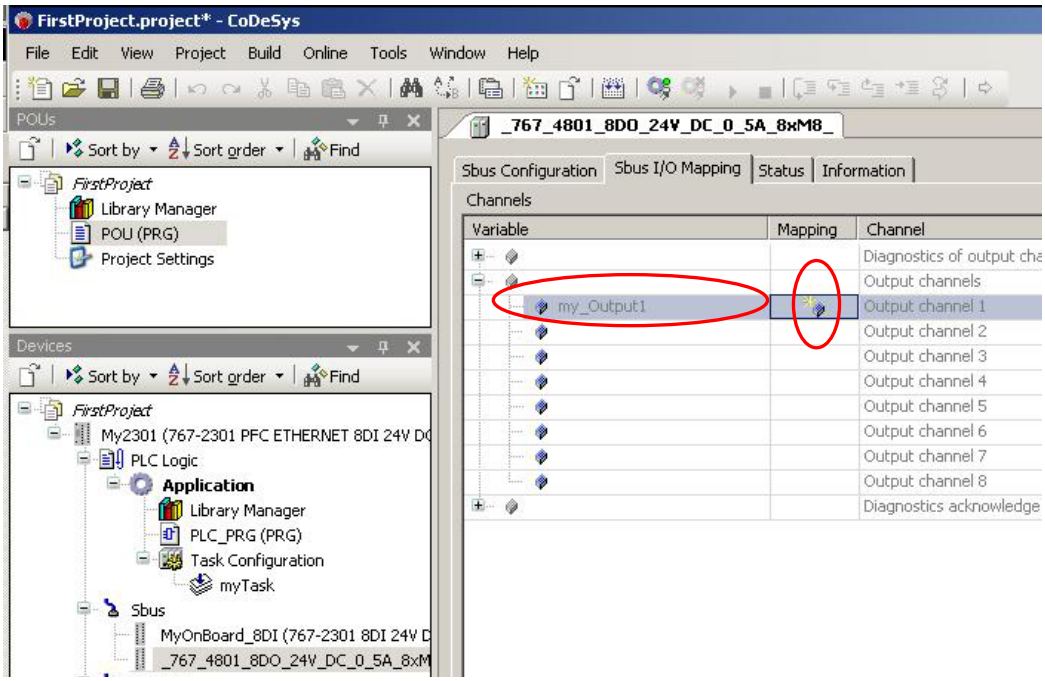


Figure 72: Define variables used in the program using inputs and outputs

3. Enter a name for the data area in the "Variable" pane. By designating the name, you have defined a new variable. This variable can now be used in CoDeSys.

15.3.10 Control Program access to the CANopen Fieldbus Variables

The CANopen fieldbus variables are available for data exchange between the control program and CANopen. This involves two memory areas of 512 bytes each. One memory area contains the fieldbus input variables and can be reached via CANopen objects 0xA000 through 0xA440. These objects can only be read by CANopen and mapped in TxPDOs. The PLC program can be utilized to gain write access to this area.
The other memory area contains fieldbus output variables and can be reached via CANopen objects 0xA480 through 0xA8C0. CANopen can be used to read and write these objects and to map them in RxPDOs. The PLC program can be used to gain read access to this area.

Access to the memory area by the PLC program of the fieldbus variables occurs via device objects. These are to be created in CoDeSys.
Each device object represents an 8-, 16- or 32-bit value within the CANopen fieldbus input variables or fieldbus output variables. Using a bit offset (which you specify), you can access any address in the memory area of the fieldbus variables. CANopen objects are assigned to this memory area.

To add a device object, please proceed as follows:

1. In the "Devices" window, right-click on "CANopen" and select **Add Object...** from the context menu. The "Add Object" dialog box opens.

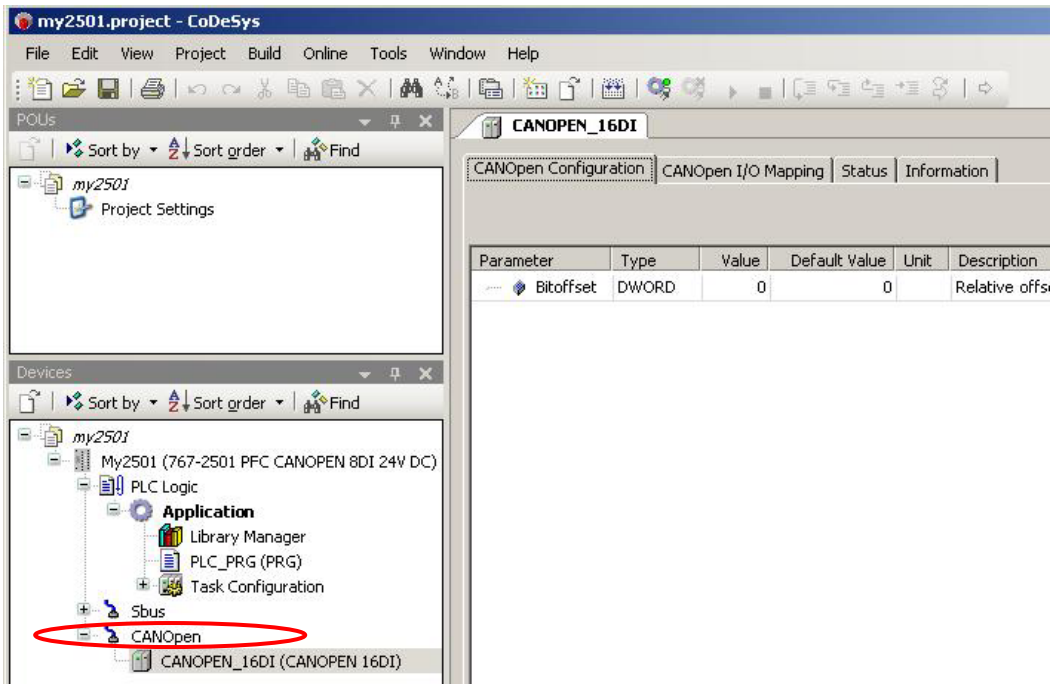


Figure 73: Adding device objects

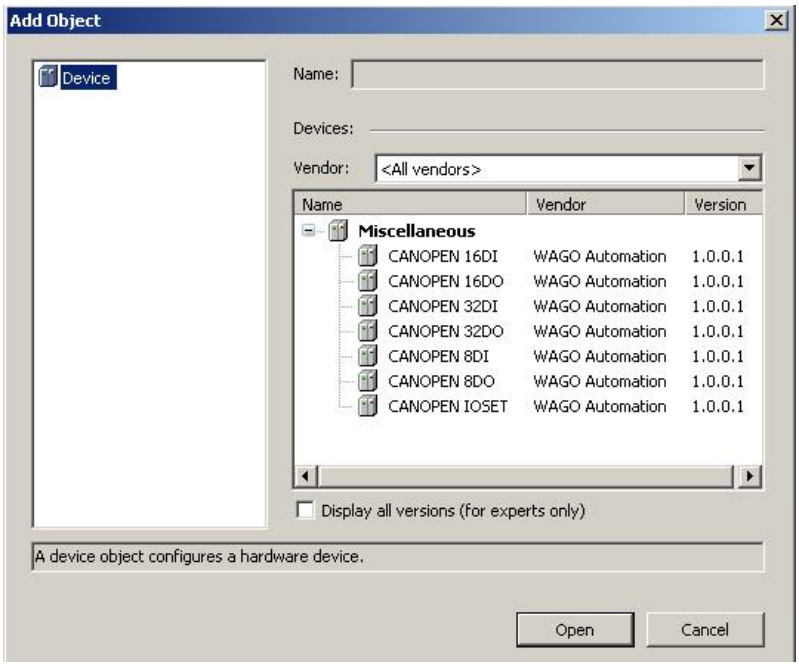


Figure 74: "Add Object" dialog box

2. Select here a device object for the CANopen fieldbus variable according to bit width and input or output direction. Select the corresponding device object and click **[Open]**.

3. Enter the bit offset of a device object in the "Value" pane. In this example, the bit offset is 20. This means that the device object lies on bit offset 20 of the memory area of the CANopen fieldbus output variables.

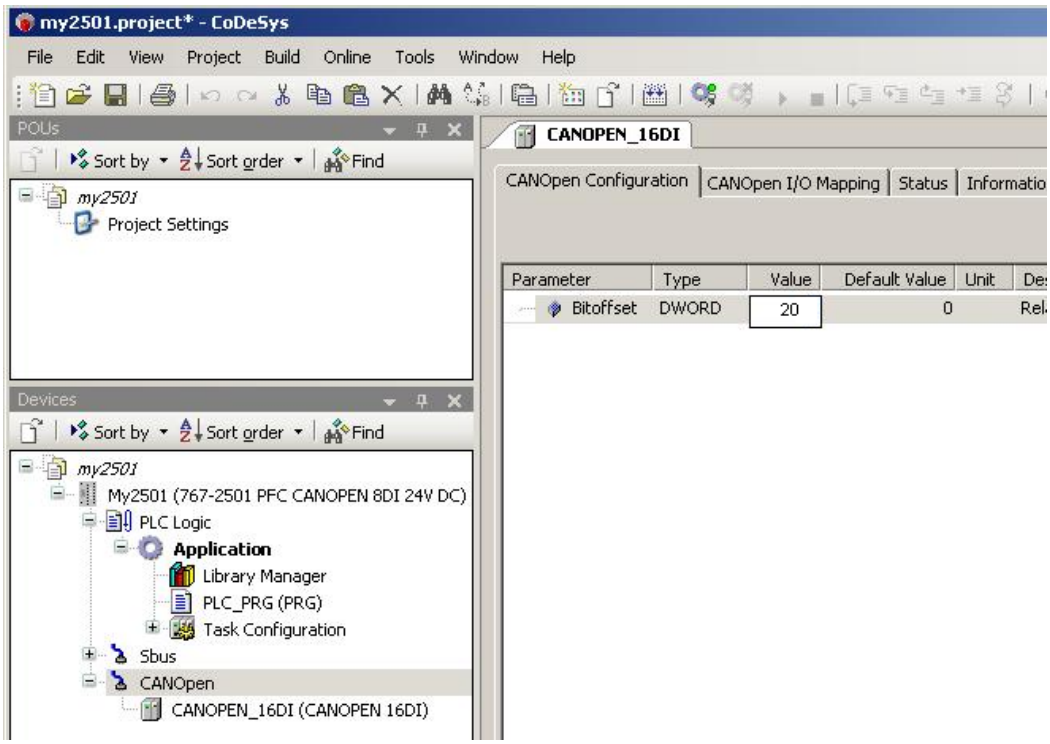


Figure 75: Assigning a bit offset

15.3.10.1 Assignment Between Device Objects and the Variables of the Control Program

In order to access the device objects in the PLC program, you must connect them with a variable. You have two options:

- Link variables used in the program using inputs and outputs of individual device objects.
- Define variables for the inputs and outputs of individual device objects.

Linking variables used in the program using inputs and outputs of individual device objects.

1. To link variables, click on the "CANopen I/O Image" tab.

2. In the "Variable" pane, enter the name and path of a variable that you have created in CoDeSys. The following variable is contained in the WAGO project template: Application.PLC_PRG.iCount.

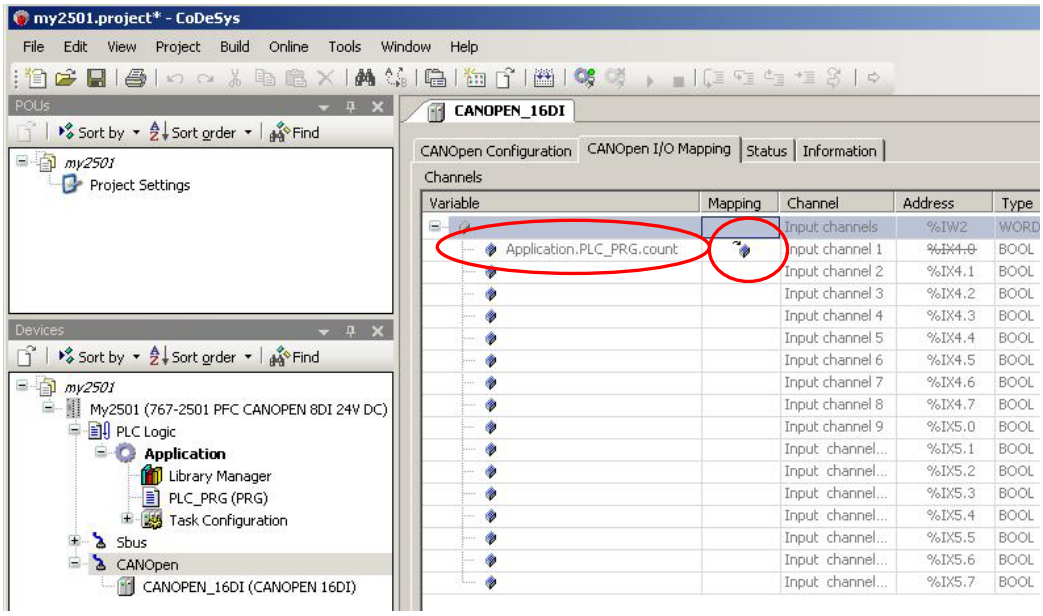


Figure 76: Linking variables used in the program using inputs and outputs

3. Double-click in the "Mapping" pane to choose between the options "Map to Existing Variable" and "Create New Variable." Select "Map to Existing Variable."

Defining variables for the inputs and outputs of individual device objects

1. To define variables, click on the "CANopen I/O Image" tab.

2. Enter a name for the data area in the "Variable" pane. By designating the name, you have defined a new variable. This variable can now be used in a CoDeSys program.

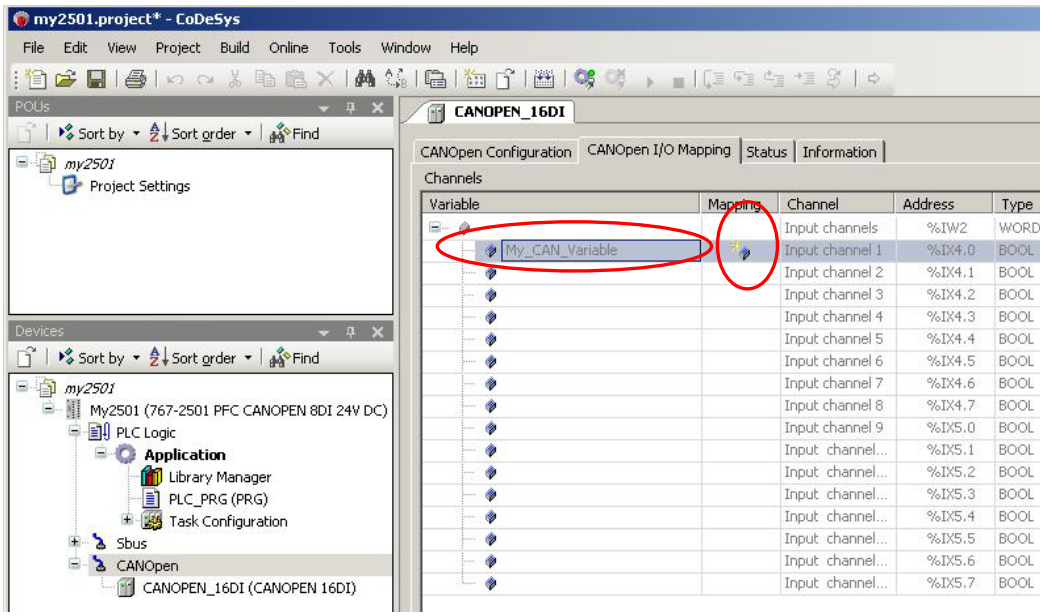


Figure 77: Defining variables used in the program using inputs and outputs

3. Double-click on the icon in the "Mapping" pane to choose between the options "Map to Existing Variable" and "Create New Variable." Select "Create New Variable."

15.3.11 Creating a Boot Project and Executing it at Fieldbus Coupler Start

To create a boot project, proceed as follows:

1. Click **Online** in the menu bar and select **Log in to "Application"…**.

2. Click **Online** in the menu bar and select **Create boot application for "Application"…**.

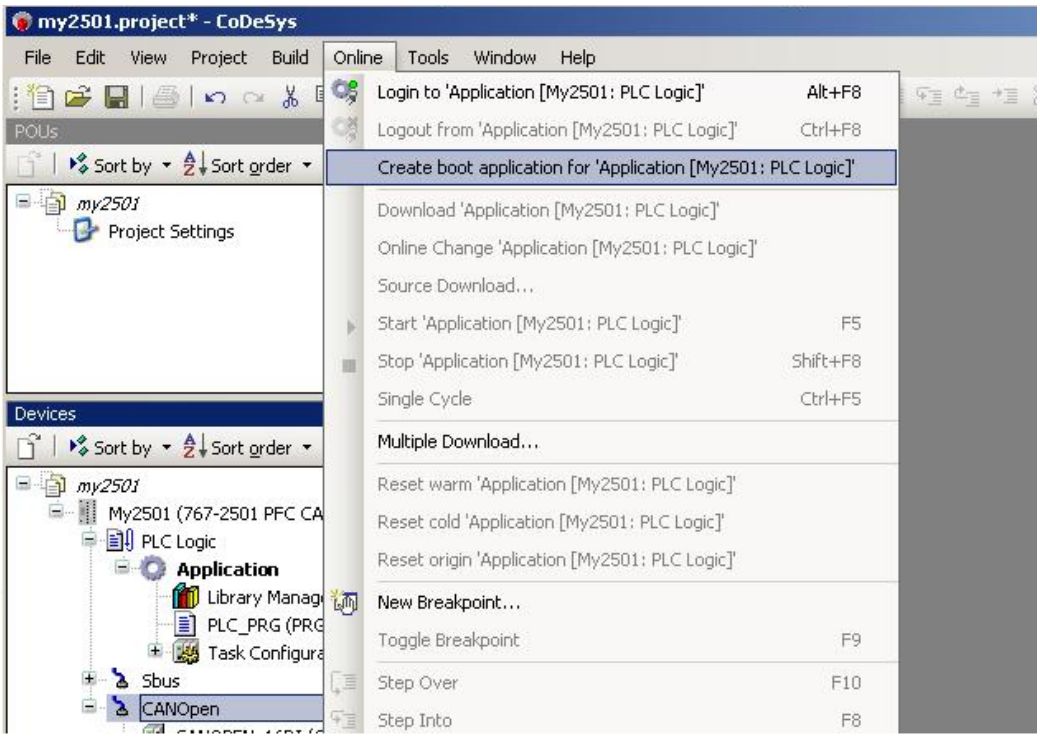


Figure 78: Creating a boot project

1. To monitor whether the boot project has been written into the controller, double-click on your device description in the "Devices" pane. Select the "Files" tab.

2. Click the refresh icon in the "Runtime" window. When the files *Application.app* and *Application.crc* appear, your program is in the controller as a boot project.

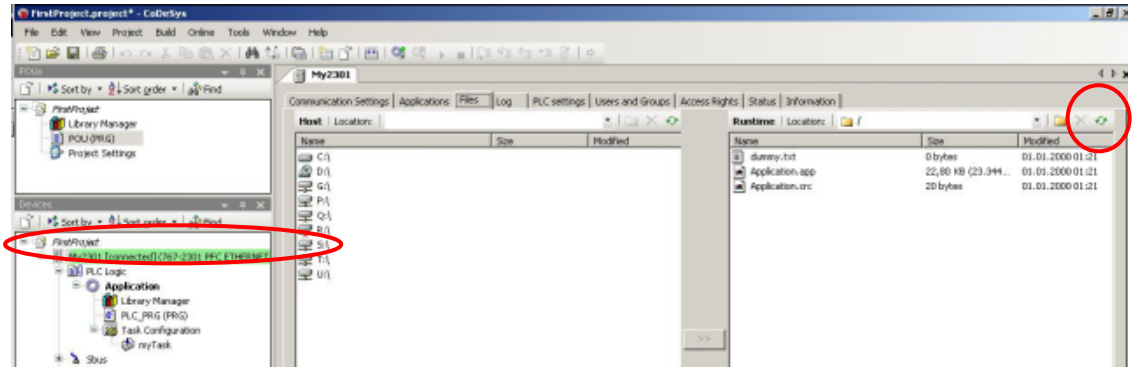


Figure 79: CoDeSys file explorer

Set the switch 10 (Run/Stop) to "ON" (Run) to enable the PLC program to run automatically after each restart.

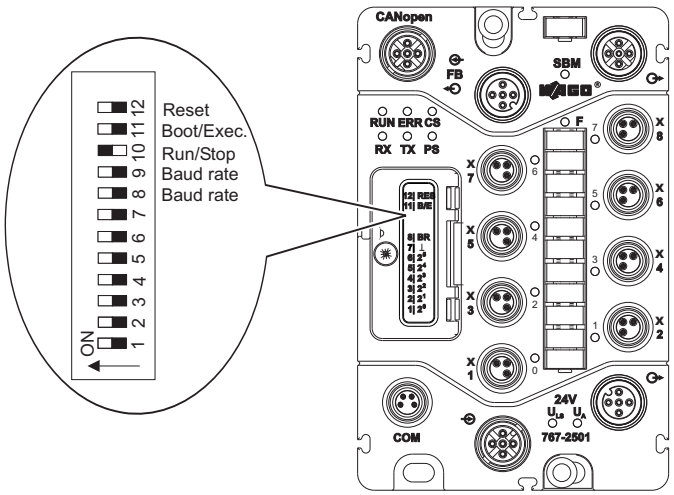


Figure 80: Run/Stop switch (10)

15.4 Notes on CoDeSys Functions

Table 55: Notes on CoDeSys functions

	Description
File system, P drive (1 MB)	The overall size of the PLC program, visualization files, bitmaps, log files, configuration files, etc. may not exceed 1 MB.
Maximum number of IEC tasks	This is limited to 12.
Free-running tasks	Free-running tasks are not supported.
Computer performance/processor time	The fieldbus coupler is based on a real-time operating system with pre-emptive multitasking. High-priority processes such as the PLC program will interrupt or eliminate low-priority processes.
Network load	Because the fieldbus coupler processor also processes CANopen telegrams, its load is dependent on it. You can reduce the fieldbus coupler's load by limiting CANopen communication.
Multi Application	Independent PLC applications can be operated in the fieldbus coupler via "Multi Application" function. Each application has its own memory area and can be changed or loaded independently. For example, this allows lighting control and temperature regulation to be automated independently from each other.

16 Diagnostic and Status Information

For the local diagnostic, the fieldbus coupler contains various LEDs that display the operating state of both the fieldbus coupler and the S-BUS. This information can also be displayed with WAGOframe. For more information, see Section 16.4.

16.1 CANopen Status Messages via LED Signals

The following table lists the CANopen status messages that are signalized by the LEDs (21) on the fieldbus coupler. Information regarding remedies of certain causes is also provided.

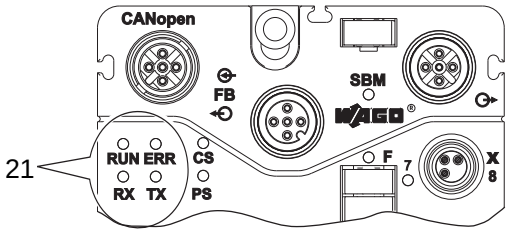


Figure 81: LEDs which display status messages

Table 56: Status messages of the RX and TX LEDs

LED	Color	Cause	Remedy/information
RX	Red	The CAN receiver buffer is full. Data loss can be anticipated.	Check the configuration of the CAN bus. Downsize the COB ID of the TxPDO.
TX	Red	The CAN sending buffer is full. Data loss can be anticipated.	Check the configuration of the CAN bus. Send fewer PDOs per time unit to the fieldbus coupler.

The following tables contain the descriptions of the CANopen-specific LEDs "ERR" "RUN."

If CANopen-specific errors arise, they are signaled via the ERR LED. The errors are classified according to priority. The ERR LED uses a "red, flickering" blink sequence to signal an error with the lowest priority; a red, glowing ERR LED signals an error with the highest priority.
If several errors exist simultaneously, the one with highest priority is always displayed. If this error no longer exists, the one with the next-highest priority is displayed.

Table 57: Status messages of the ERR LED

LED	Color/status	Cause	Remedy/information
ERR	Red, flickering	Automatic baud rate detection is active. A valid baud rate has not yet been detected.	A second CANopen device must be included in the network that sends telegrams with a valid baud rate.
	Red, blinking	Invalid fieldbus coupler configuration. An error arose during initialization of a CANopen application.	Restart the fieldbus coupler. If the error persists, contact WAGO support.
	Single red flash	At least one of the CAN controller error counters has reached or exceeded the warning level. This is caused by too many error telegrams on the CAN bus.	Check whether the fieldbus coupler is connected with at least one other CANopen device on the CAN bus. The same baud rate must be set for all CANopen devices.
	Double red flash (double flash)	A guarding or heartbeat error has occurred.	Check the configuration of the CAN bus.
	Triple red flash (trible flash)	The SYNC message was not received within the configured "Communication Cycle Period."	Check the configuration of the CANopen devices.
	Red	The CAN controller is in the "Bus Off" state. At least one of the CAN controller error counters has reached its maximum value.	Check whether the fieldbus coupler is connected with at least one other CANopen device on the CAN bus. The same baud rate must be set for all CANopen devices.

Table 58: Status messages of the RUN LED

LED	Color	Cause	Remedy/information
RUN	Off	Off	If no other LEDs light up, power is not being supplied to the unit. Turn on the power supply.
	Green, blinking (blinking)	Pre-operational	The fieldbus coupler is in the pre-operational state (see Section 12.1.2)
	Single green flash (single flash)	Stopped	The fieldbus coupler is in the stopped state (see Section 12.1.4)
	Green	Operational	The fieldbus coupler is in the operational state (see Section 12.1.3)

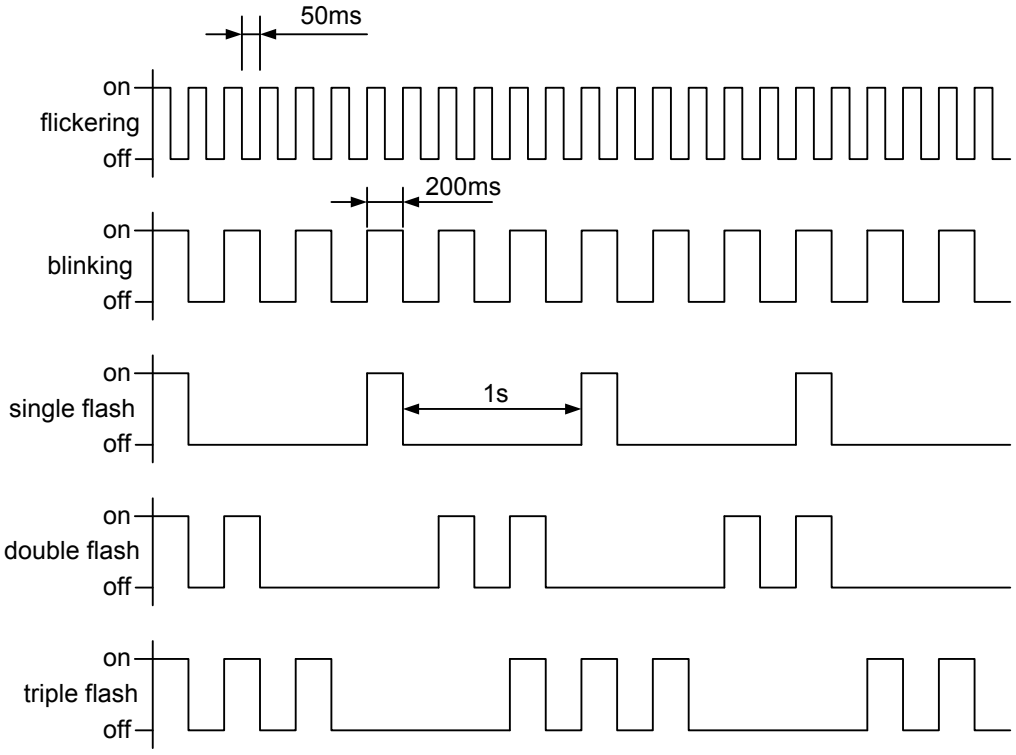


Figure 82: Graphical representation of blink sequences

16.2 Operational Messages of the Fieldbus Coupler via LED Signals

The following table lists the operational messages that are indicated via LEDs. Information regarding remedies of certain causes is also provided.

Note

Use the diagnostic overview (Section 17.7.3) to disable specific diagnostics (see F-LED). In this case, the LED is disabled (off).

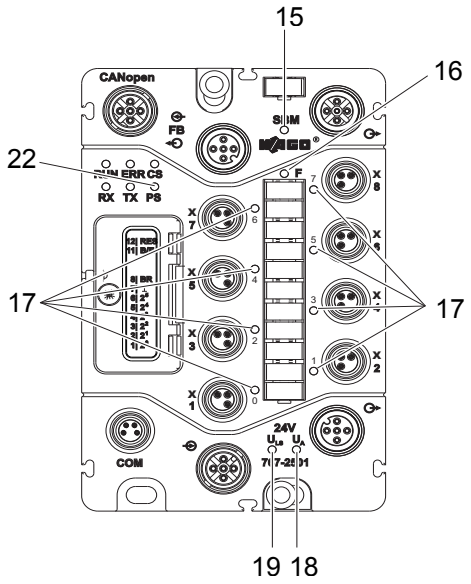


Figure 83: LEDs which display operational messages

Table 59: Operational messages of the fieldbus coupler

Position	LED	Color	Cause	Solution
15	SBM	Green flashing	S-BUS is being started.	-
		Green	S-BUS is functioning without problems.	-
		Red	Disruption on the S-BUS.	Check whether all components are connected to the S-BUS. Also check the S-BUS connection and S-BUS cable.
16	F	Red	Group error. At least one diagnostic message is available on the digital inputs.	Check the power supply of the connected sensors
17	I/O	Yellow	Input signal pending.	-

Table 59: Operational messages of the fieldbus coupler

Position	LED	Color	Cause	Solution
18	U_A	Green	Actuator supply is present.	-
		Off	Actuator supply is not present.	Connect the power supply and check the voltage level, if applicable.
19	U_{LS}	Green	Logic supply and sensor supply are present.	-
		Off	Logic supply and sensor supply are not present.	Connect the power supply and check the voltage level, if applicable.
22	PS	Off	There is no PLC program (boot project) in the controller	-
		Green	A PLC program (boot project) is in the controller and is being executed ("RUN" mode)	-
		Red	A PLC program (boot project) is in the controller, but is not being executed ("STOP" mode)	-

16.3 Error Messages from the Fieldbus Coupler via LED Signals

Error messages or warnings from the coupler are displayed in WAGOframe. They are also displayed via the CS LED (23) as a blink code.

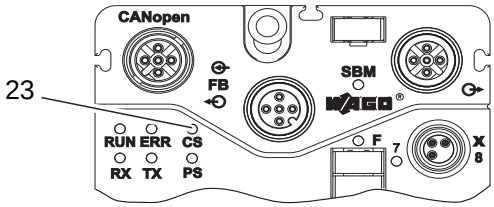


Figure 84: Indicator of blink codes via CS LED

A disruption is always displayed as three blinking sequences in a cyclic manner:

1. The first blinking sequence indicates the **group number.**
   The starting point in identifying the error is the group number, from which the error groups can be determined. The number of blink pulses indicates the specific error group.
   Example:
   Group number 1: Group of S-BUS errors.

2. After a short pause, the second blinking sequence appears and indicates the **error code**. The number of blink pulses indicates the specific error code that describes the type of error.
   Example:
   Error code 5: S-BUS terminator not attached to last I/O module.

3. After another pause, the third blinking sequence appears and indicates the error argument. The number of blink pulses indicates the **error argument**, which provides supplemental descriptions of the error; for example, on which of the 767 components an error is present.
   Example:
   Entry 0 … 63: In which of the connected I/O modules this error has arisen. For example, if "5" is indicated in the blink code, the error arose in the fifth I/O module (the "0" denotes the digital inputs of the fieldbus coupler)

The group number, error code and error argument are displayed as blink codes that are to be converted into numbers. The blink code can display numbers with single, double or triple digits. A zero is always expressed in four cycles of 20 Hz; zeroes preceding a digit(s) are ignored.

16.3.1 Progression of Blink Sequence

The following table outlines the progression of the blink codes over time. If there is no number in the second position ("1" in the number "10") or third position ("1" in the number "100") for the group number, error code or error argument, the subsequent pause does not take place; instead, the next sequence is initiated (designated in ***bold and cursive*** font in the following table).

Table 60: Overview of the blink codes

Description	Frequency	Additional explanations
Start sequence blinks	10 cycles with 12.5Hz each (12.5 times per second)	Initiation of start phase
Pause	1s	
Group number		
Group number (third (100) position)	Repeats: 0.5s on and 0.5s off	Repeats according to group number
Pause	2s	-
Group number (second (10) position)	Repeats: 0.5s on and 0.5s off	Repeats according to group number
Pause	2s	-
Group number (first (1) position)	Repeats: 0.5s on and 0.5s off	Repeats according to group number
Pause	***2s***	-
Blinks	***40ms***	Initiation of error code
Pause	***2s***	
Error code		
Error code (third (100) position)	Repeats: 0.5s on and 0.5s off	Repeats according to error code
Pause	2s	-
Error code (second (10) position)	Repeats: 0.5s on and 0.5s off	Repeats according to error code
Pause	2s	-
Error code (first (1) position)	Repeats: 0.5s on and 0.5s off	Repeats according to error code
Pause	***2s***	-
Blinks	***40ms***	Initiation of error argument
Pause	***2s***	
Error argument		
Error argument (third (100) position)	Repeats: 0.5s on and 0.5s off	Repeats according to error argument
Pause	2s	-
Error argument (second (10) position)	Repeats: 0.5s on and 0.5s off	Repeats according to error argument
Pause	2s	-
Error argument (first (1) position)	Repeats: 0.5s on and 0.5s off	Repeats according to error argument
Pause	4s	-
Progression restarts when start sequence blinks		

16.3.2 Example of an Error Message via Blink Code

The following example clarifies an error message as indicated by the blink code. A S-BUS error is displayed when the software update for the sixth I/O module fails.

Initialization:

1. The CS LED begins with the initiation of the start phase: quick flashing of about 1s.

2. A one-second pause occurs.

Group number 1: S-BUS error

3. The first position (1) blinks once: 0.5s on and off.

4. The initiation of the error code follows with a pause of two seconds, 40ms of blinking and another pause of two seconds.

Error code 13: software update for I/O module failed

5. The second position (10) blinks once the 10th position.

6. A two-second pause occurs.

7. The first position (1) blinks 3 times the 1th position.

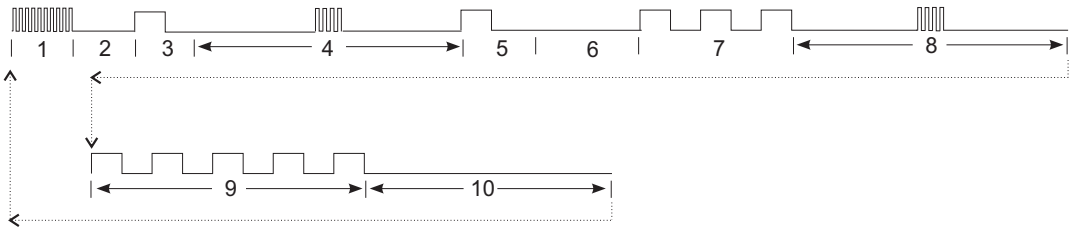
8. The initiation of the error argument follows with a pause of two seconds, 40ms of blinking and another pause of two seconds.

Error argument 5: I/O module in the sixth slot

9. The first position (1) blinks 5 times the 1th position.

10. A four-second pause occurs.

The blink code starts when the start phase is initiated. This process is repeated if only one error is present; if more are present, the next pending error is processed.



16.3.3 Meaning of the blink codes and procedures for troubleshooting

In this Section, all errors and warnings are listed that are given by the CS LED and BUS LED.

The errors and warnings are divided into the following error groups:

Table 61: List of error groups

Group number	Name	Status Indication
1	S-BUS error	CS LED
2	S-BUS warnings	
3	RTS error	
4	-	
5	General internal hardware errors	
6	General internal hardware warnings	
7	General software errors	
8	General software warnings	
9	-	-
10	-	-
11	CANopen-specific errors	CS LED
12	CANopen-specific warnings	
13	Firmware loader error	
14	Error with firmware download	

If the subsequent errors and warnings are not corrected with the indicated measures, please contact the WAGO AUTOMATION Support. Be ready to give them the blink code that is displayed.

Phone: +49 571 887 555
Fax: +49 571 887 8555
E-mail: support@wago.com

Table 62: Group number 1: S-BUS error

Error code	Error argument	Cause	Correction
1	0 … 65	Error during S-BUS initialization.	Check the cable for damage. Perform a restart by disconnecting the power supply and then reconnecting it.
2			
3			
4			
5		Error during S-BUS initialization. The last I/O module could not be initialized.	Check the cables to and from the last I/O module. Make sure that the S-BUS terminator is plugged into the last I/O module or are not connected to the fieldbus coupler as 64 I/O modules.
6	0 … 64	Error when starting S-BUS. An S-BUS disruption exists on the I/O module that is located in front of the I/O module indicated by the error argument.	Check the cable for damage. Perform a restart by disconnecting the power supply and then reconnecting it.
	255	The "Restart" function (see "S-BUS Master" in Section 17.7.1) is deactivated; a short disruption occurred on the S-BUS. This could not be localized since the disruption on the S-BUS no longer existed during investigation of the disruption point.	
7	0 … 63	Error when starting S-BUS.	
8			
9			
10			
11	1	A disruption in the S-BUS has arisen.	Check whether the S-BUS cable is connected properly.
			Check the S-BUS cable for damages.
12	1	Switching to software update mode is not possible.	Perform a restart by disconnecting the power supply and then reconnecting it.
13	0 … 63	Software update for I/O module failed.	
14	1	Authentication failed.	Perform a restart by disconnecting the power supply and then reconnecting it.
	2		

Table 62: Group number 1: S-BUS error

Error code	Error argument	Cause	Correction
15	0 … 63	Current software is defective. S-BUS operation only possible in recovery mode to load new software.	Perform a firmware update of the I/O modules.
16	1	Error when starting the S-BUS.	Perform a restart by disconnecting the power supply and then reconnecting it.
	2		
	3	Error when stopping the S-BUS	
17	0	Error at the digital inputs of the fieldbus coupler	
18	0 … 63	Communication test with I/O module failed	
19	255	Error when initializing the S-BUS	Perform a restart by disconnecting the power supply and then reconnecting it. If you cannot eliminate the error, please contact WAGO support.
…			
23			
24	1		

Table 63: Group number 2: S-BUS warnings

Error code	Error argument	Cause	Correction
1	1	Cycle time of S-BUS cannot be maintained	Specify a higher cycle time
4	255	Firmware update of I/O module	The modules switch to the mode for updating the firmware. In this mode, no data is exchanged (process data).

Table 64: Group number 3: RTS error

Error code	Error argument	Cause	Correction
1	1	Initialization failed when starting the 767 components.	Perform a restart by disconnecting the power supply and then reconnecting it.
1	2	Boot project cannot be loaded.	Reduce the amount of data in the file system. Diminish the project size.
1	4	Checksum not ok. Error in retention memory or boot project.	Reload the boot project to the controller via CoDeSys.
1	5	A reference is used in the PLC program that does not exist in the target.	Use the correct device description file. Delete the parts of the program that contain the unresolved reference.
1	6	Communication error (serial, USB, Ethernet)	Perform a restart by disconnecting the power supply and then reconnecting it.
			Check the data cables for correct fit and for damages.
1	8	Configuration error	Contact WAGO support.
1	9	Incorrect network mask or incorrect port is being used.	Check the network settings.
1	10	System resources are depleted.	Perform a restart by disconnecting the power supply and then reconnecting it.
1	11	Configuration of PLC application exists twice or is incorrect.	Check the project in use.
1	13	Error of SPEEDWAY driver.	Check the log file in CoDeSys for additional information on the error.
1	14	Error of I/O driver.	Check the project in use.

Table 64: Group number 3: RTS error

Error code	Error argument	Cause	Correction
2	1	Error in PLC program	Check the PLC program.
2	2	CoDeSys has identified an overload or the watchdog time has been exceeded.	Check the timeouts used in the project. Reduce the amount of PLC code.
2	3	The project being used is not appropriate to the device.	Compile the project for the device being used.
2	4	Communication disruption	Perform a restart by disconnecting the power supply and then reconnecting it.
2	5	Error in the scheduler	

Table 65: Group number 5: general internal hardware errors

Error code	Error argument	Cause	Correction
1	1	Error when accessing flash memory	Perform a restart by disconnecting the power supply and then reconnecting it.
	2		
2	1	EEPROM error	
	2		
	3	Real-time clock (RTC) is defective	-
	4		
3	1	RAM error	Perform a restart by disconnecting the power supply and then reconnecting it.
	2		
	3		
4	1	Error in internal coprocessor	
	2		
5	1		
	2		
6	1	Combination of hardware and software not allowed	Perform a firmware update on the fieldbus coupler
7	255	Internal fault	These blink codes assist the support team in further error investigation. Please be ready to give them the group number, error code and error argument.
…			
18			

Table 66: Group number 6: general internal hardware warnings

Error code	Error argument	Cause	Correction
1	1	Fieldbus coupler has not been connected to power supply for six days (back-up capacitor of RTC empty)	Set the real-time clock via WAGOframe

Table 67: Group number 7: general software errors

Error code	Error argument	Cause	Correction
1	0 – 254	Internal fault	Perform a restart by disconnecting the power supply and then reconnecting it. If you cannot eliminate the error, please contact WAGO support.
2	1 – 7		
3	1 – 4		
4	1		
6	1 – 10		
7	1		
9	1	Internal fault	Perform a restart by disconnecting the power supply and then reconnecting it. If you cannot eliminate the error, please contact WAGO support.
	2		
	3		
	4		
	5		
	10	Error in the parameter database of the fieldbus coupler.	Select the checkbox "Create nominal system configuration" in the SPH-DTM to save the current parameters of the I/O modules again to the file system of the fieldbus coupler (parameter database is created). If you cannot eliminate the error, please contact WAGO support.
	11	Parameter database version is not supported.	
	12	The number of I/O modules does not match the number entered in the parameter database (I/O module removed or added).	
	13	Error in the parameter database of the fieldbus coupler.	
	14	Internal fault	Perform a restart by disconnecting the power supply and then reconnecting it. If you cannot eliminate the error, please contact WAGO support.
	15		
	16		
	20	Order number(s) for I/O module(s) differ(s) from that saved in the parameter database (e.g., 767-6401 instead of 767-6402).	Replace the I/O module by the I/O module with the correct order number. The "Bus address of the first faulty component" parameter appears in the SPH-DTM. This shows you the I/O module to be replaced (e.g., "Bus address 3" refers to the 3rd I/O module on the fieldbus coupler).
9	21	The firmware release index from I/O module(s) does not match that saved in the parameter database.	Update the firmware of the I/O module affected.

Table 67: Group number 7: general software errors

Error code	Error argument	Cause	Correction
	22	The hardware release index from I/O module(s) does not match that saved in the parameter database.	Update the parameter database in the fieldbus coupler for the I/O module affected. Enter the position of the respective I/O module in the "Bus address of I/O modul to update" parameter.
	23	Internal fault	Perform a restart by disconnecting the power supply and then reconnecting it. If you cannot eliminate the error, please contact WAGO support.
	24		
	25	Saved parameters cannot be transferred to the I/O module (communication error).	
	30	Error in the parameter database of the fieldbus coupler.	Select the checkbox "Create nominal system configuration" in the SPH-DTM to save the current parameters of the I/O modules again to the file system of the fieldbus coupler (parameter database is created). If the problem persists, contact WAGO support.
	31	Internal fault	Perform a restart by disconnecting the power supply and then reconnecting it. If you cannot eliminate the error, please contact WAGO support.
	32		
	33		
	34		
	40		

Table 68: Group number 8: general software warnings

Error code	Error argument	Cause	Correction
2	2	Automatic parameterization of the I/O modules is not permitted.	Select the "Do not overwrite I/O module parameters" checkbox in the SPH. Disable this option if you want to allow automatic parameterization for all I/O modules or to update the parameter database.

Table 69: Group number 11: CANopen-specific errors

Error code	Error argument	Cause	Correction
1	1 … 7	Error in the CANopen application.	Perform a restart by disconnecting the power supply and then reconnecting it.
2	1	Error during initialization of the CANopen application.	
	2	Error during write access to EEPROM.	
	3	Error during read access to EEPROM.	
	4	Error during write access to flash memory.	
	5	Error during read access to flash memory.	
3	1	Maximum number of entries in object directory reached.	Reduce the number of connected I/O modules.
	2	The emergency send memory is full. Telegrams can be overwritten.	Check whether the fieldbus coupler is connected to at least one additional CANopen device on the CAN bus. All CANopen devices must have the same baud rate setting.

Table 70: Group number 12: CANopen-specific warnings

Error code	Error argument	Cause	Correction
1	1	Error during initialization of the S-BUS. The CANopen application does not have access to the process data of the I/O modules. Corresponding entries in the object directory are missing or are set to zero.	Check the cable for damage. Perform a restart by disconnecting the power supply and then reconnecting it.

Table 71: Group number 13: firmware loader error

Error code	Error argument	Cause	Correction
1	1	No firmware in fieldbus coupler	Perform a firmware update using the USB interface
	2	Firmware checksum error	
	3		
2	1	USB communication disrupted	Perform a restart by disconnecting the power supply and then reconnecting it.
	2		
	3		
	4		
	5		
	6		
	7		
3	1	EEPROM error or incompatible firmware	Perform a restart by disconnecting the power supply and then reconnecting it. Perform a firmware update using the USB interface
	2		
	3		
	4		
	5		
	6		
	7		
4	1	Internal fault	
	2		
5	1		
	2		
6	1	Access to flash memory not possible	
	2		
	3		
	4	Wrong firmware version	
	5	Internal fault	
7	1	Error in firmware file	
	2		
	3		
	4		
	5		
	6		
	7		
8	1	Real-time clock (RTC) is defective	-
	2		
	3		

Table 71: Group number 13: firmware loader error

Error code	Error argument	Cause	Correction
9	1	Code processor error	Perform a restart by disconnecting the power supply and then reconnecting it.
	2		
	3		
	4		
	5		
	6		
	7		
	8		
	9		
10	1		
	2		
	3		
	4		
	5		
	6		
	7		
	8		
	9		
11	1	Error when accessing Interrupt Controller	
	2		
12	1	SDRAM error	

Table 72: Group number 14: error with firmware download

Error code	Error argument	Cause	Correction
1	0	Switching to software update mode is not possible	Perform a restart by disconnecting the power supply and then reconnecting it.
2	0 … 64	Software update for I/O module failed	
3	1	Internal fault	These blink codes assist the support team in further error investigation. Please be ready to give them the group number, error code and error argument.
	2		
	3		
	4		

16.4 Readout of Blink Codes using WAGO DTMs

Error messages or warnings from the fieldbus coupler are also displayed in WAGOframe under the "Blink code" entry. For this to function, the WAGOframe software (or another FDT border application) and the WAGO DTMs must be installed on your PC. For more information, see Section 17. An overview of the meaning of the blink codes can be found in Section 16.3.3.

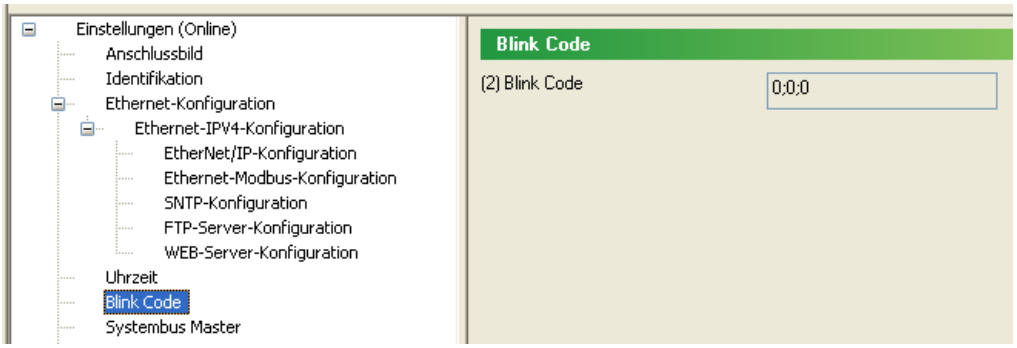


Figure 85: Example of the blink code display under the "Blink code" parameter in WAGOframe

17 Parameterization via FDT/DTM

This Section explains device parameterization with the help of an FDT border application (FDT/DTM) using WAGOframe.

⚠ DANGER

Changing parameters!
When parameters are incorrectly modified with an FDT/DTM border application (e.g. WAGOframe), machine components could be placed in a dangerous state and personnel and machines could be at risk. Before changing the parameters, ensure that the machine components are in a safe and defined state and switch off the higher-level controller.
Also ensure before start-up that no personnel remain in the danger area of the machine components.

FDT is an abbreviation for "Field Device Tool". This refers to an application that can be utilized for parameterization of the fieldbus devices independent of the fieldbus being used. For this purpose, the application needs supplementation in the form of software components, which establish the communication with the individual devices and supply the adjustable parameters. These software components are called DTM (Device Type Manager) and produced by the device manufacturers.

FDT/DTM represents an open concept in which the individual components of various manufacturers work together. The concept thus reduces the number of proprietary, manufacturer-specific software solutions and cultivates a uniform operating concept inside a comprehensive operating program.

For the parameterization of a 767 node, an appropriate DTM is available for each 767 component. Use this DTM to parameterize the 767 components. The 767 components can be parameterized either online or offline. The offline mode enables the parameterization of a 767 component that is not yet present. In the offline mode, first store the parameter in a project and later transfer it to the 767 components.

In the online mode, there is a direct connection between the display and the connected 767 components. If a 767 component is in the online mode, its name is displayed in **bold** and ***cursive*** font in the network window.

17.1 Installing the FDT/DTM Components

The following sections deal exclusively with the WAGOframe FDT/DTM frame application.

When using another FDT/DTM frame application, install only the USB driver (item no.: 759-922), the Service Interface DTM (item no.: 759-371) and the WAGO DTM (item no.: 759-361) on your computer. These can be obtained on the Internet at www.wago.com.

Information on configuring the fieldbus coupler using WAGO DTM can be obtained in Section 17.7.

1. Insert the "WAGOframe" CD into your CD-ROM drive.

2. The start screen appears. Select the desired installation language. A product selection window opens. If Autostart is not activated, open the "Language.htm" file.

Figure 86: Start screen of WAGOframe CD

3. To install the necessary 767 programs, click on the appropriate link. A window opens displaying the corresponding installation programs.

Figure 87: Window for product selection 1

4. To enable you to use the devices of the 767 series, install the FDT/DTM components shown below by clicking on the appropriate link. The "File Download" dialog box opens. To install the programs, click **[Open]**.

   - USB driver, 759-922 (installation not required when CoDeSys 3 – item no. 759-915 – is installed on your PC)
   - WAGOframe (installation is only necessary if an FDT/DTM border application has been installed on your PC), 759-370
   - WAGO service interface DTM, 759-371
   - DTM for the fieldbus coupler and I/O modules, 759-361
   - DTM for the system update, 759-362

Figure 88: Window for product selection 2

5. If all necessary 767 programs have been installed, connect your PC with the coupler via the USB cable. For more information, please see Section 6.7.

17.2 Starting WAGOframe

1. Start the FDT border application by double-clicking on the WAGOframe logo on your desktop.

Figure 89: WAGOframe logo

2. You can also start WAGOframe from the start menu of your operating system by clicking on "Start" and selecting **Programs** > **WAGO Software** > **WAGOframe**.

3. When the "Device Selection Wizard" opens, select "Expert Mode". The "Point to Point Mode" only applies to the configuration of directly connected devices with no S-BUS (such as WAGO Jumpflex). Click **[Next]**.

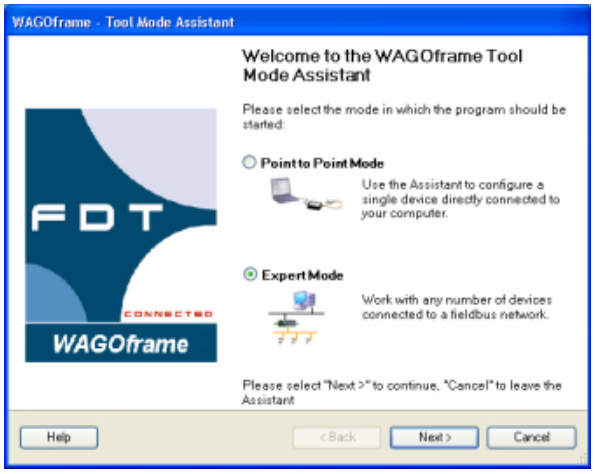


Figure 90: "Device Selection Wizard"

4. When opened for the first time, the "Query WAGOframe" dialog box appears. Click **[Yes]** to automatically set up the device catalog on your PC. After the device catalog has been updated, all 767 components for which a DTM is installed are listed.

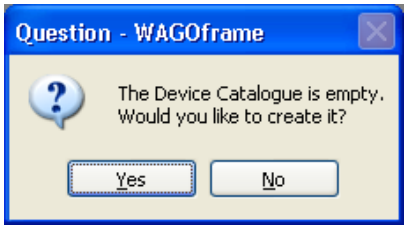


Figure 91: "Query WAGOframe" dialog box

17.3 Expansion of Device Catalog to include 767 Components

WAGOframe automatically detects newly installed DTMs at the next start-up. To expand the device catalog, proceed as follows:

1. To update the device catalog, navigate to the menu bar and select **View > Device Catalog**.

2. In the "Query WAGOframe" dialog box, click **[Yes]** to update the device catalog.

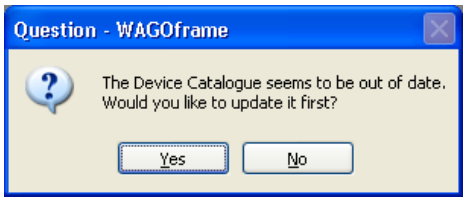


Figure 92: "Query WAGOframe" dialog box

After updating, the device drivers for the 767 components appear in the device catalog.

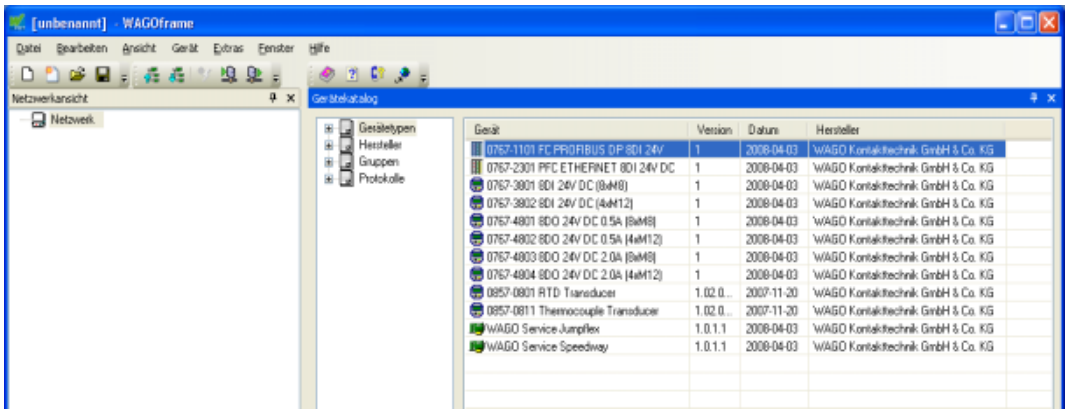


Figure 93: View of device catalog

17.4 Setting Up Network Manually

n order to work with the 767 components, the exact node structure topology must be copied into the "Network View" of WAGOframe.

Note

Depending on the application, you can create the network manually (this Section), automatically (see Section 17.6.7) or using the Lifelist (Section 17.6.8).

17.4.1 Adding the Communication DTM

So that you can parameterize the fieldbus coupler and the I/O modules connected with it, establish a connection to your PC via the USB interface.

1. Right-click on "Network" in the "Network View" pane.
2. In the context menu, select **Add…**. The "Add..." dialog box opens.

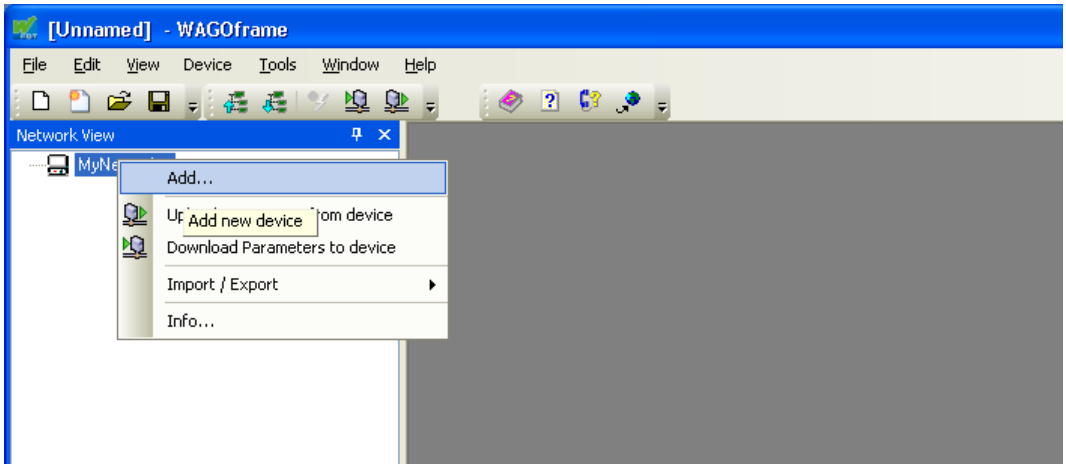


Figure 94: Adding the communication DTM

3. In the "Add" dialog box, select "WAGO Service Speedway".

4. Click **[OK]** to confirm your selection.

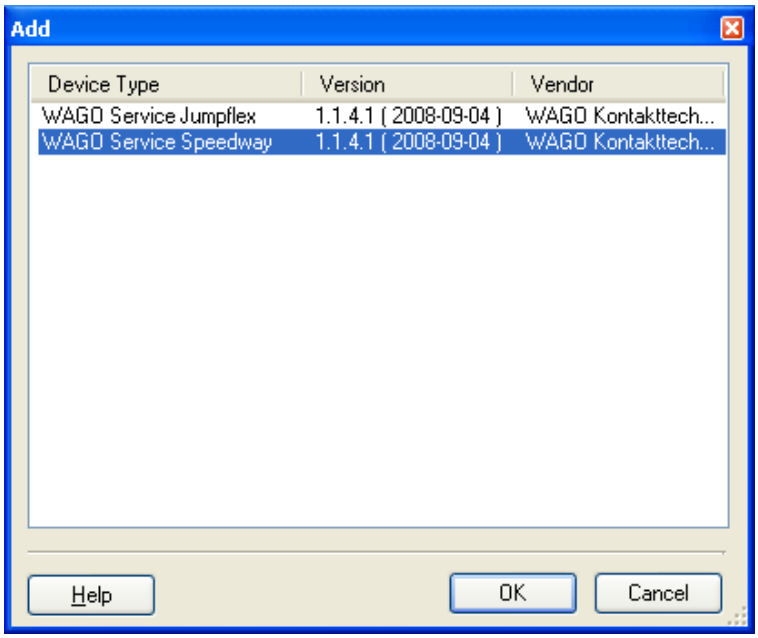


Figure 95: Selecting the communication DTM

17.4.2 Selecting the Communications Interface for WAGOframe

Requirement: The fieldbus coupler must be switched on and your PC must be connected with the USB interface. For more information, see Section 6.7.

1. Double-click on the "WAGO Service Speedway" device driver in the "Network View" pane. The DTM for the interface configuration opens.

2. Select the COM port that you use from the Interface selection box. If the list of available interfaces is empty, check whether the coupler is switched on and connected to your PC via the USB cable.
   If you establish the connection via **[Go online]** later, the COM port is checked. If it is incorrect, it is automatically selected.

3. Click **[Apply]** and then **[Close]**.

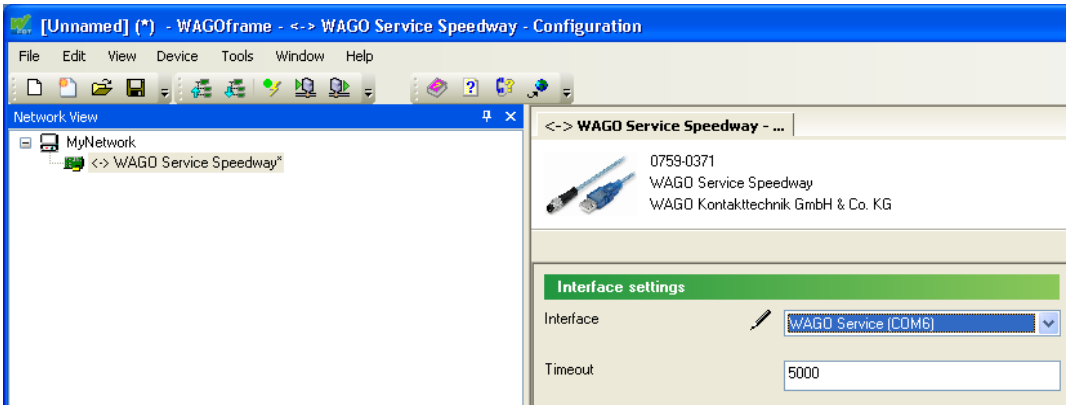


Figure 96: The DTM for the interface configuration

Note

This COM port number remains constant for the fieldbus coupler currently connected. When connecting a different fieldbus coupler to the PC, the COM port also changes. In this case, "WAGO Service Speedway" attempts to select the new COM port automatically. If several fieldbus couplers are connected to the PC, select the correct COM port.
The "WAGO SPEEDWAY Portmapper" is used to specify a constant COM port. The software is contained on the "WAGOframe" CD-ROM.

17.4.3 Adding a Fieldbus Coupler

To incorporate the fieldbus coupler that you use or the programmable coupler into WAGOframe, proceed as follows:

1. Right-click on the "WAGO Service Speedway" device driver in the "Network View" pane.

2.In the context menu, select **Add…**. The "Add" dialog box opens.

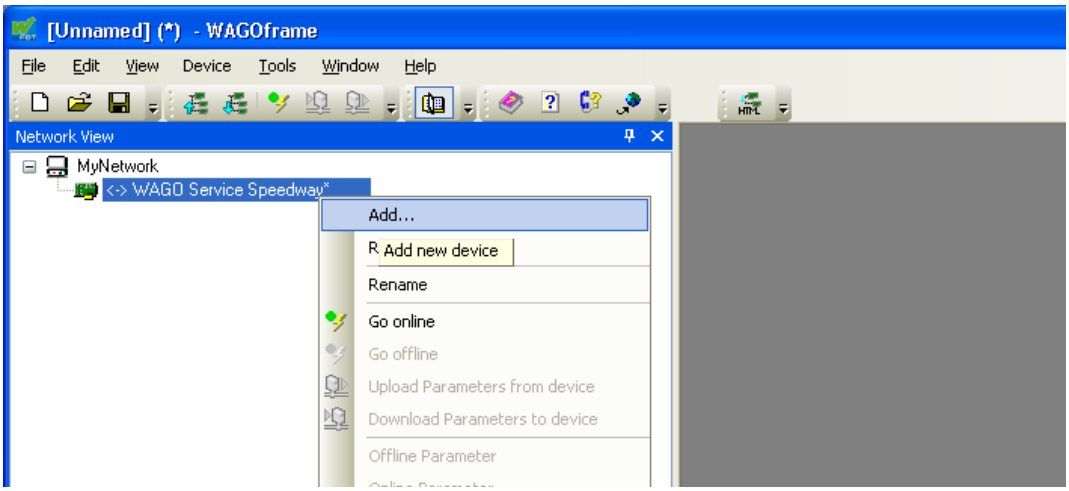


Figure 97: Adding a coupler

3. In the "Add" dialog box, select the coupler that you use.

4. Click **[OK]** to submit your selection.

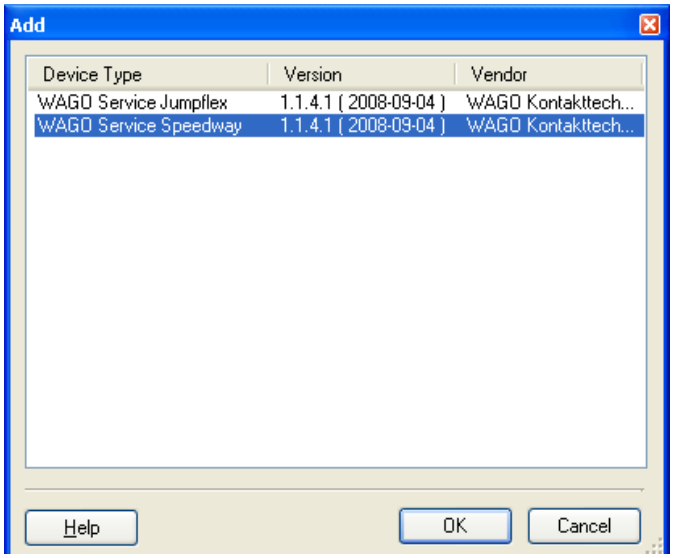


Figure 98: Adding the coupler

17.4.4 Adding the I/O Modules

To incorporate the I/O modules that you use into WAGOframe, proceed as follows:

1. Right-click on the "<Speedway:> 0767-xxxx" device driver in the "Network View" pane.

2. In the context menu, select **Add...**. The "Add" dialog box opens.

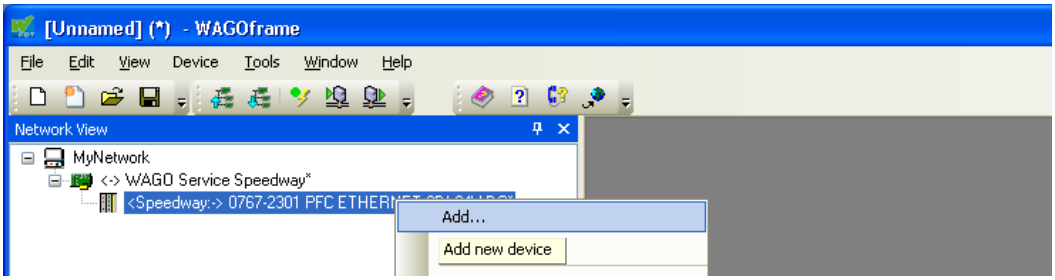


Figure 99: Adding the I/O modules

3. In the "Add" dialog box, select the I/O module's device type.

4. Click **[OK]** to confirm your selection.

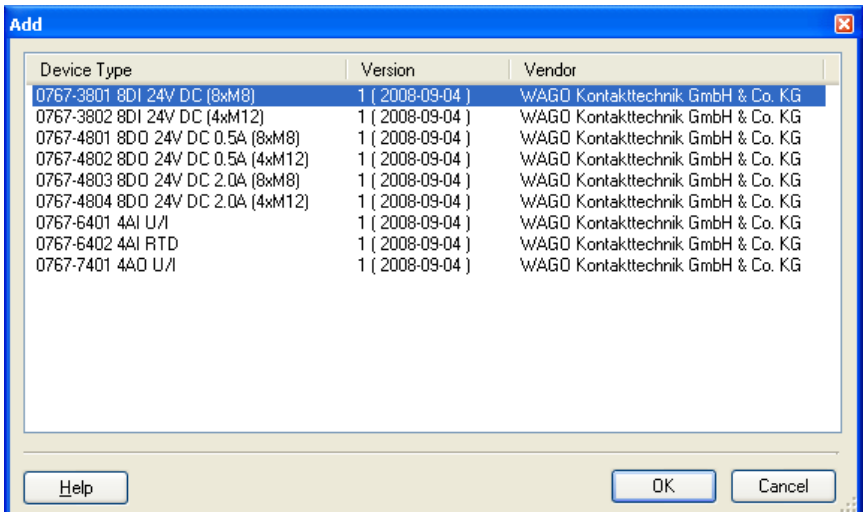


Figure 100: Selecting an I/O module

5. Repeat steps one through four until the module arrangement is concordant with your fieldbus node. In our example, two I/O modules were added.

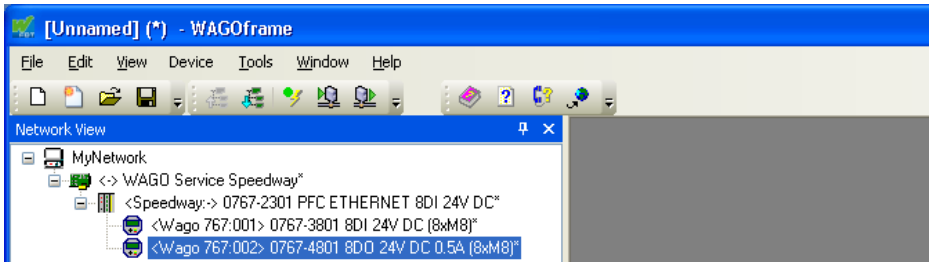


Figure 101: Two added I/O modules

17.5 Online and Offline Configuration

The options of online and offline configuration are available for configuring the 767 components. The offline mode enables the parameterization of a 767 component that is not yet present. In the offline mode, first store the parameters in a project and later transfer them to the 767 components. In the online mode, there is a direct connection between the display and the connected 767 components. If a 767 component is in the online mode, its name is displayed in ***bold and italic*** font in the network window.

17.5.1 Offline Configuration

Requirement:
To configure the fieldbus coupler in offline mode, the network structure of the 767 node must have been transferred to WAGOframe (see Section 17.4).

For offline configuration, proceed as follows:

1. Right-click on the "<Speedway:> 0767-xxxx" device driver of the fieldbus coupler in the "Network View" pane.

2. In the context menu, select **Offline Parameter**. The configuration interface opens with the fieldbus coupler parameters. Details on these parameters can be found in Section 17.7.

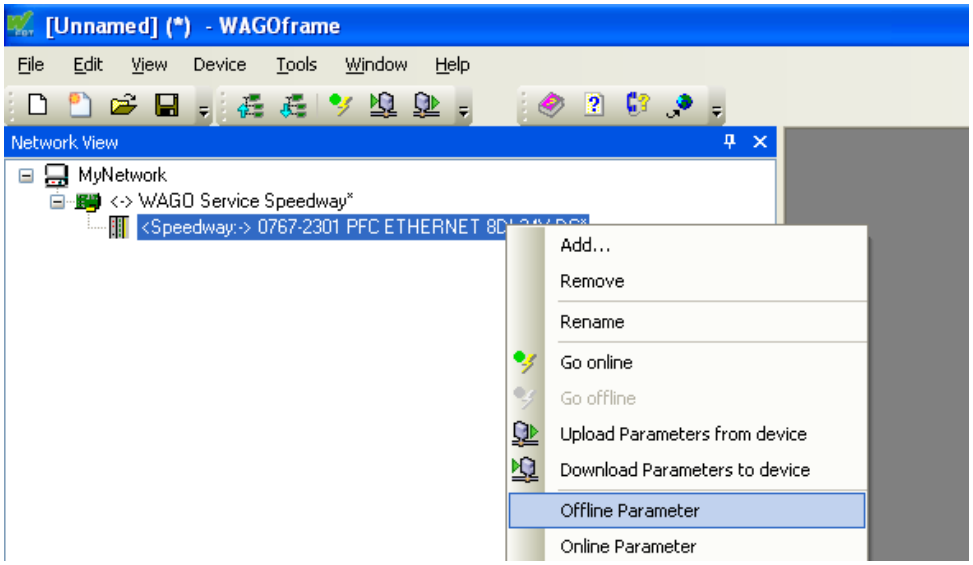


Figure 102: Opening the configuration interface (offline)

3. To configure the I/O modules, right-click on one of the I/O modules in the "Network View."

4. In the context menu, select **Offline Parameter**. The configuration interface opens with the corresponding I/O module parameters. Information on these parameters can be found in the respective I/O module manual.

5. Save the configuration in a project and then transfer it later to the respective 767 component. To transfer parameters, right-click on the fieldbus coupler or on the respective I/O module and select **Download Parameters to device** from the context menu.

17.5.2 Online Configuration

Requirement:
To configure the fieldbus coupler in online mode, the network structure of the 767 node must have been transferred to WAGOframe (see Section 17.4). You can also utilize the "Set Up Network" option as described in Section 17.6.7.

For offline configuration, proceed as follows:

1. Right-click on the "<Speedway:> 0767-xxxx" device driver in the "Network View" pane.

2. In the context menu, select **Go online**. When the progress bar reaches 100%, a connection to the fieldbus coupler has been established. Repeat this step for each desired I/O module.

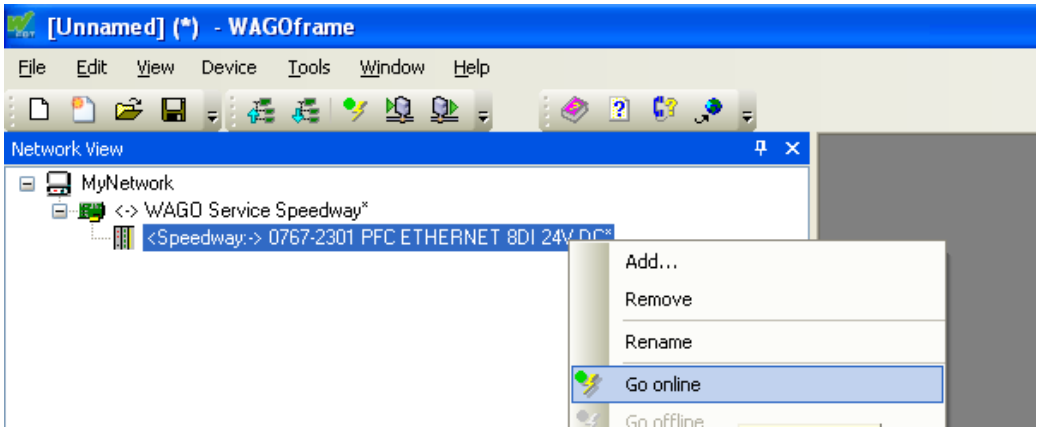


Figure 103: Setting up a connection to the fieldbus coupler

3. In the context menu, select **Online Parameter**. The configuration interface opens with the fieldbus coupler parameters. Details on these parameters can be found in Section 17.7.

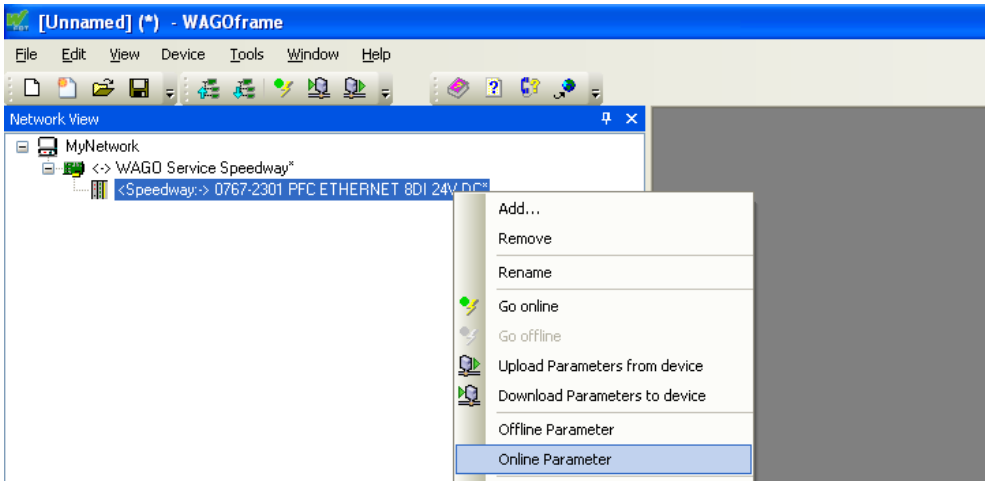


Figure 104: Opening the configuration interface (online)

4. To configure the I/O modules, right-click on one of the I/O modules in the "Network View."

5. In the context menu, select **Online Parameter**. The configuration interface opens with the corresponding I/O module parameters. Information on these parameters can be found in the respective I/O module manual.

17.6 The "Additional Functions" and "Scan" Selections

The "Additional Functions" selection from the context menu provides the following additional functionalities in addition to the possibility of parameterization:

- Change bus address
  Here you can change the S-BUS addresses for individual I/O modules.

- I/O owner assignment
  Here you can specify to which process image (e.g., fieldbus or runtime system) an I/O module should belong.

- Diagnostics setup
  Here you can enable or disable synchronous diagnostics and diagnostic confirmation of the I/O modules, as well as the integrated digital inputs of the fieldbus coupler.

- Service page
  You can use the Service page to restore the default state for selected 767 components.

- User administration
  In the user administration, you can change the preset passwords (see Section 14) for the **guest, user** and **admin** users. In addition, you can reset all passwords to their default state.

- File system
  Here you administer the file system of the fieldbus coupler.

The "Scan" selection from the context menu offers the following options:

- Set up network
  Add all 767 components connected to the PC by the USB cable to the network view.

- Life list
  Here you can search for connected 767 components.

17.6.1 Changing the Bus Address

If you wish to simply configure select I/O modules instead of the entire topology, add these I/O modules manually to the Network View. For these I/O modules to be accessible at the correct bus address, you must assign them the correct addresses. To do so, please proceed as follows:

1. Right-click on the coupler in the "Network View" pane.

2. In the context menu, select **Additional Functions** > **Change bus addresses**. A window opens with a list of the bus nodes.

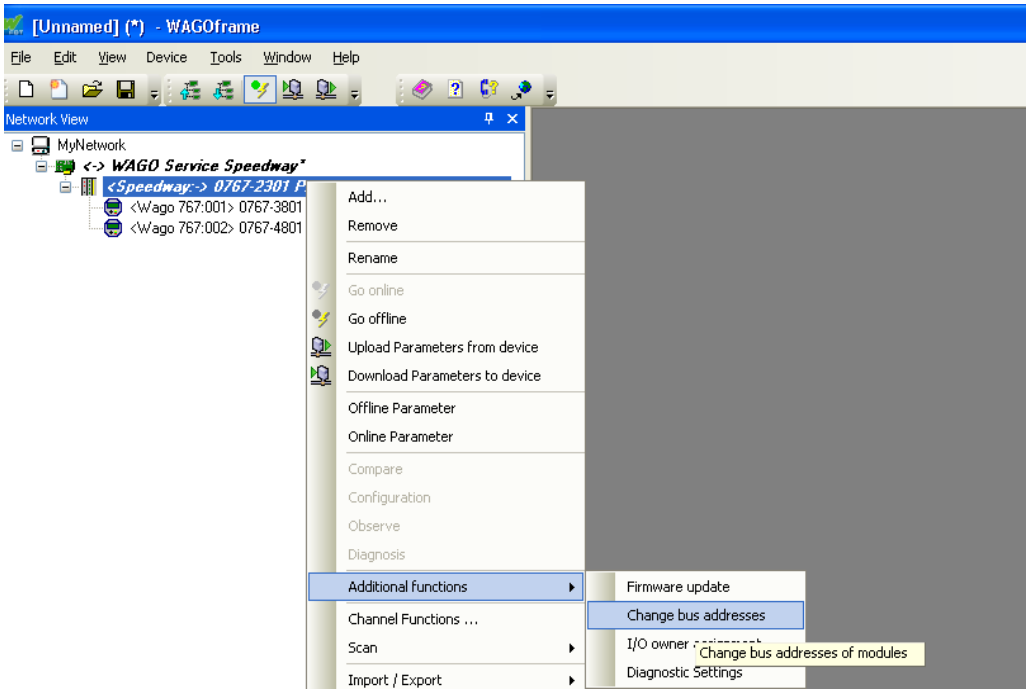


Figure 105: Opening the "List of Bus Nodes" window

3. Now select an I/O module from the "List of Bus Nodes" to which you wish to assign a new bus address. The current address is displayed in the **New Bus Address** field. Enter here the desired new address and click **[Apply]**.

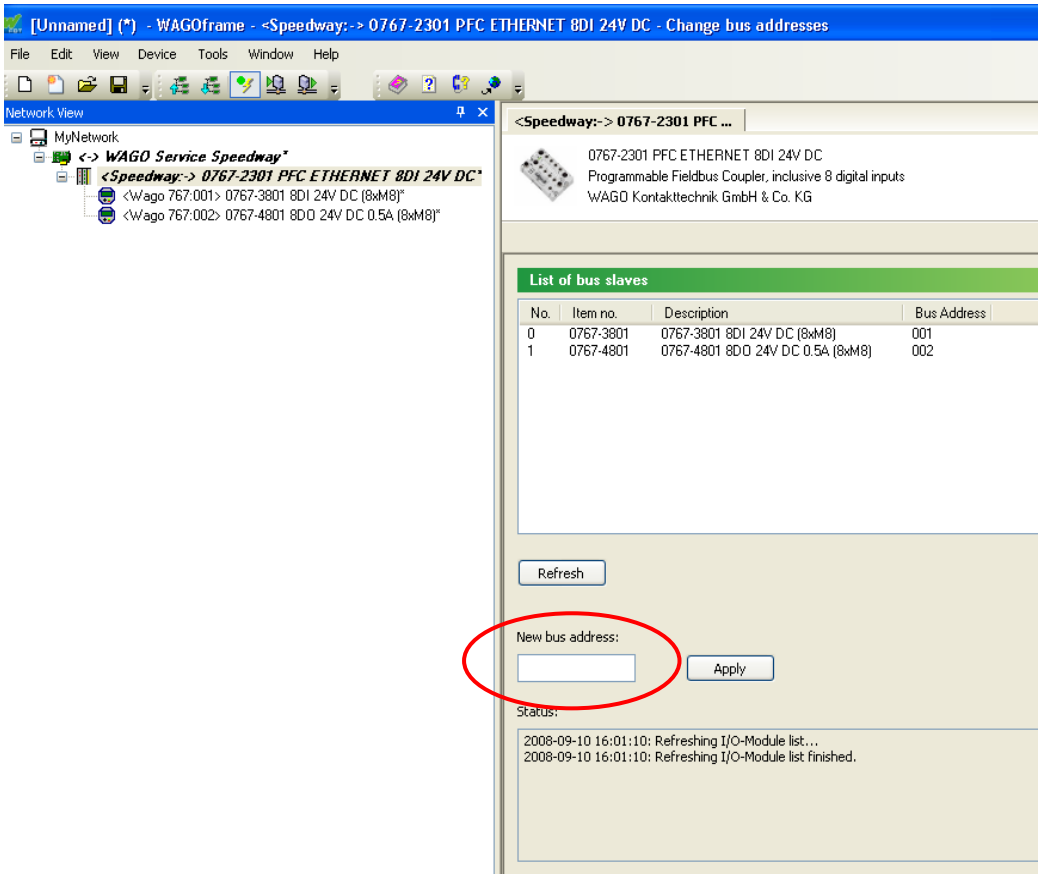


Figure 106: Assigning new bus addresses for the I/O modules

17.6.2 Assigning the I/O Owner

To configure the assignment of an I/O module to a fieldbus or to the runtime environment proceed as follows:

1. Right-click on the coupler in the "Network View" pane. In the context menu, select **Additional Functions** > **I/O owner assignment**. A window opens with a list of the bus nodes.

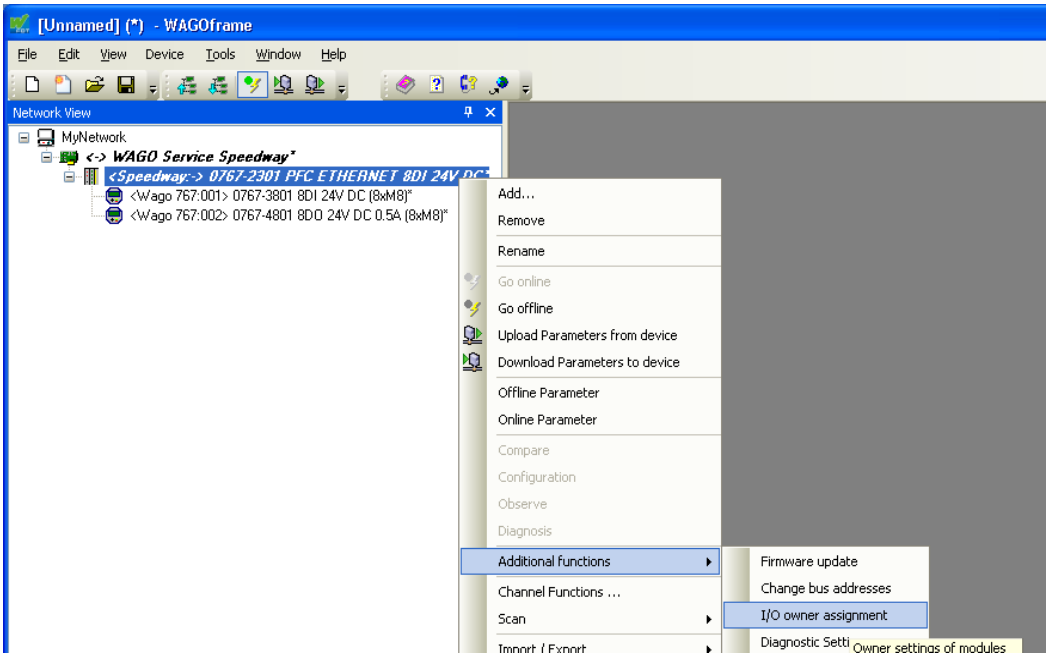


Figure 107: Opening the "Assign Owner" window

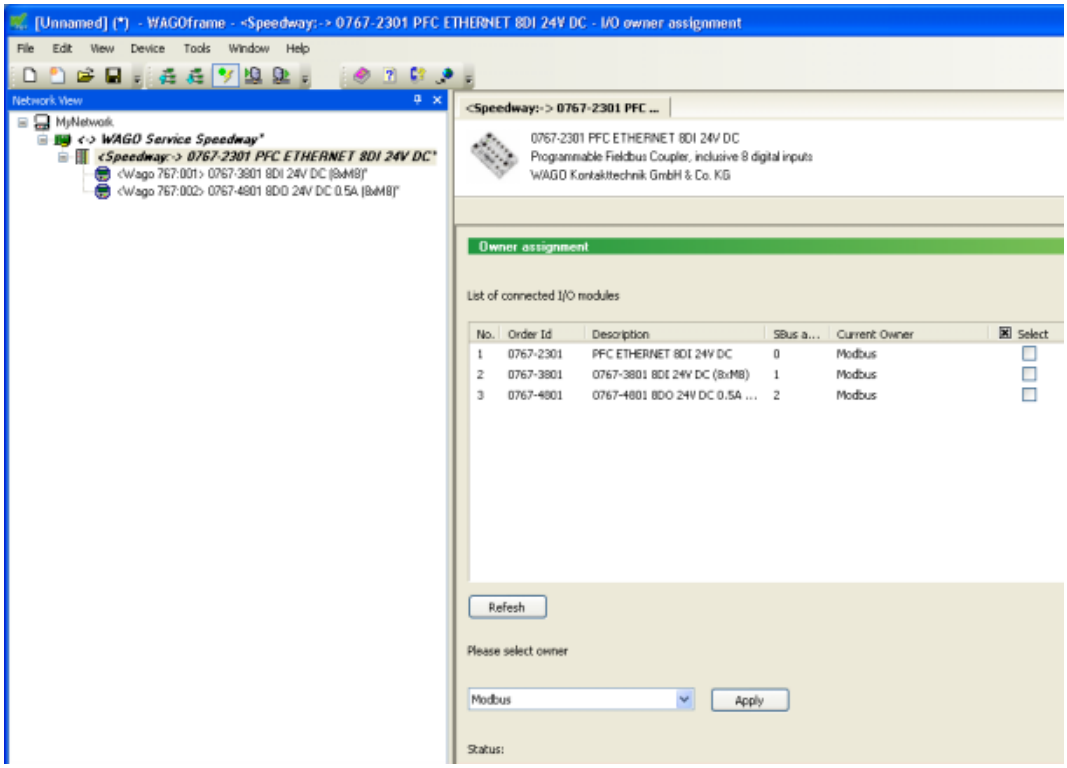


Figure 108: "Assign Owner" window

2. Select the I/O module that you wish to configure via the "Select" selection box.

3. Select the new user (fieldbus or runtime system) for the selected I/O modules from the selection box. Click **[Apply]** to confirm.

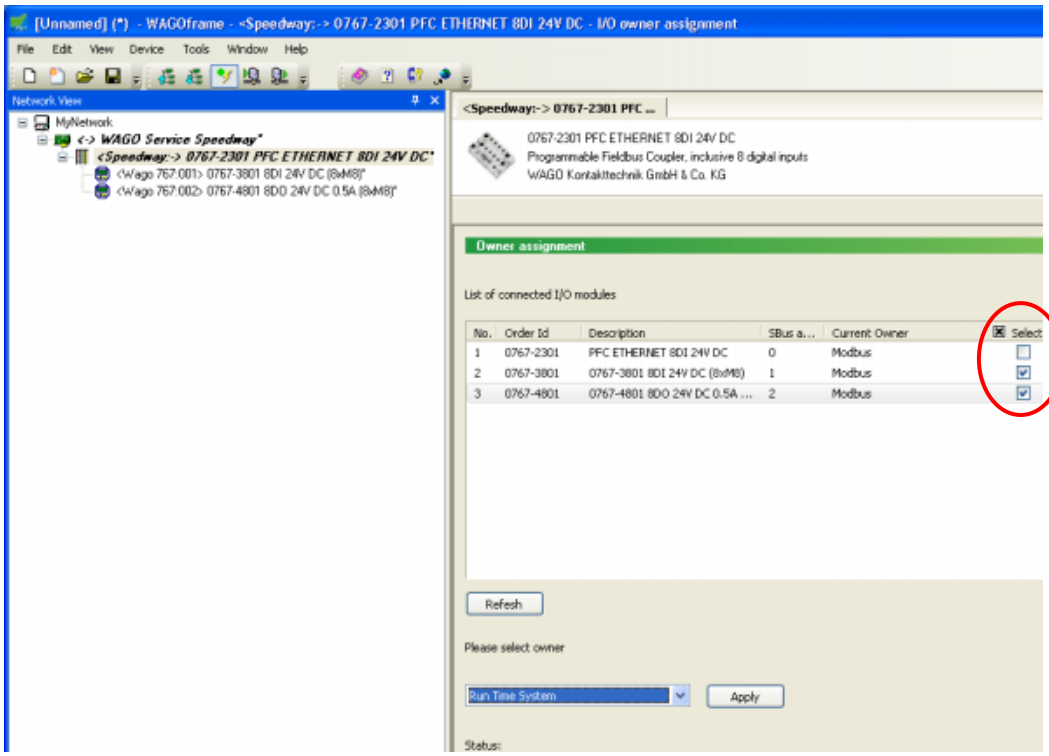


Figure 109: Selecting a new user for the selected I/O modules

Note

If you want to apply the settings, click **[Yes]** in the window. The 767 node is restarted.

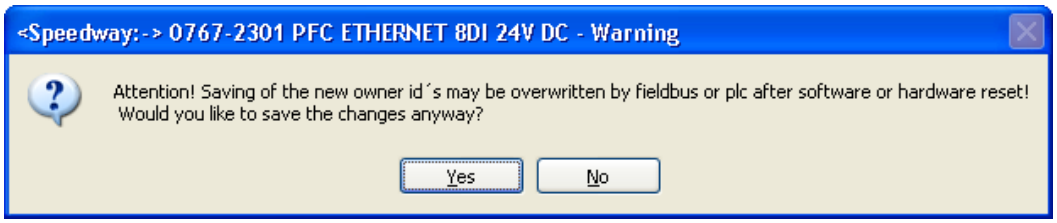


Figure 110: Warning

4. If the changes were successfully carried out, they can be seen in the updated list.

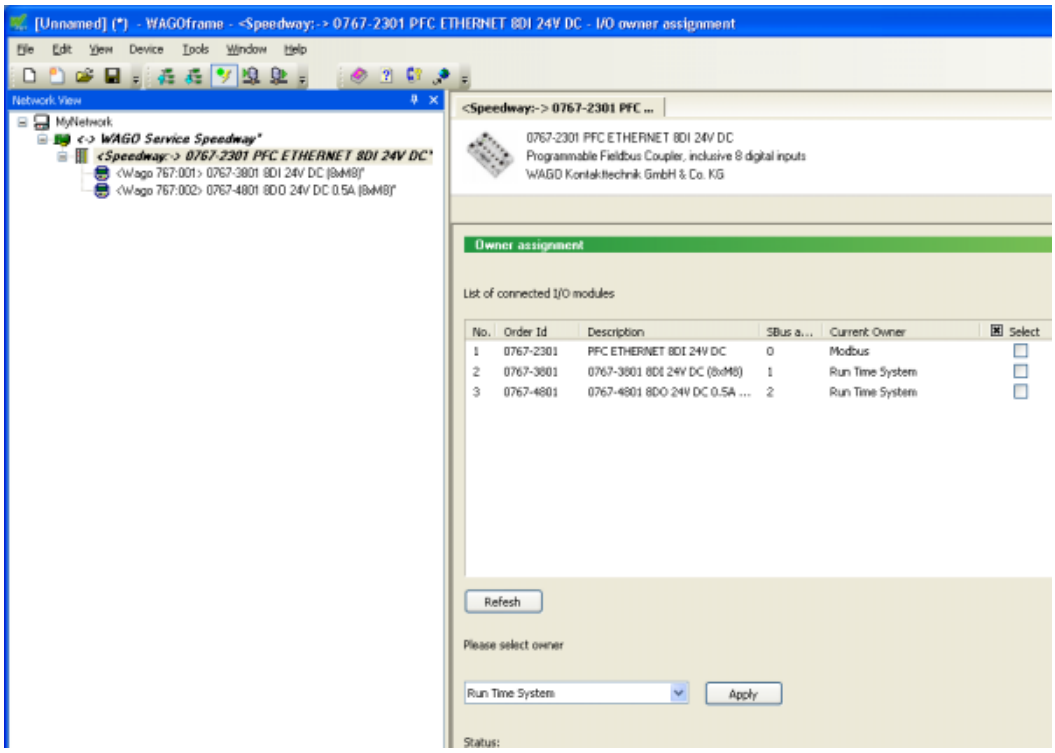


Figure 111: Runtime system chosen as new user for the selected I/O modules

Note

Because configuring the fieldbus or runtime environment accurately sets the assignation, you will normally not need to change the assignments. Configuration is only necessary if the assignment is incorrect.

17.6.3 Diagnostic Setup

Using diagnostic setup, you can activate or deactivate the synchronous diagnostics and the diagnostic confirmation of the I/O modules and couplers.

1. Right-click on the coupler in the "Network View" pane. In the context menu, select **Additional Functions** > **Diagnostic Settings**. A window opens with a list of the connected 767 components.

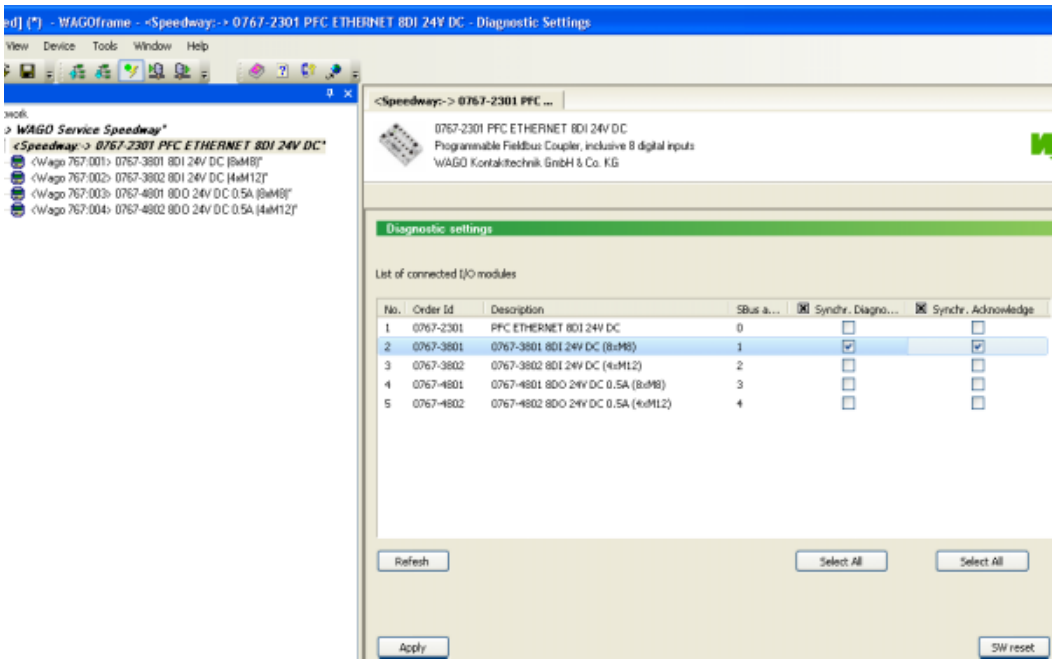


Figure 112: List of connected 767 components

2. Select the synchronous diagnostics and the diagnostic confirmation for the 767 component according to your request by activating or deactivating the selection boxes.

⚠ CAUTION

767 Node Restarting!
Restarting a 767 node can, depending on the parameterization that you performed, place machine components in dangerous circumstances and endanger personnel and machines.
Before pressing **[OK]** to restart, ensure that this will not pose a danger to the machine components.

3. Click **[Apply]** to apply your diagnostic settings. The 767 node is restarted automatically.

4. The diagnostic settings are active when the following text appears in the status indicator: "Application of selected diagnostic settings complete".

17.6.4 Service Page

The Service page is used to restore the default state for selected 767 components.

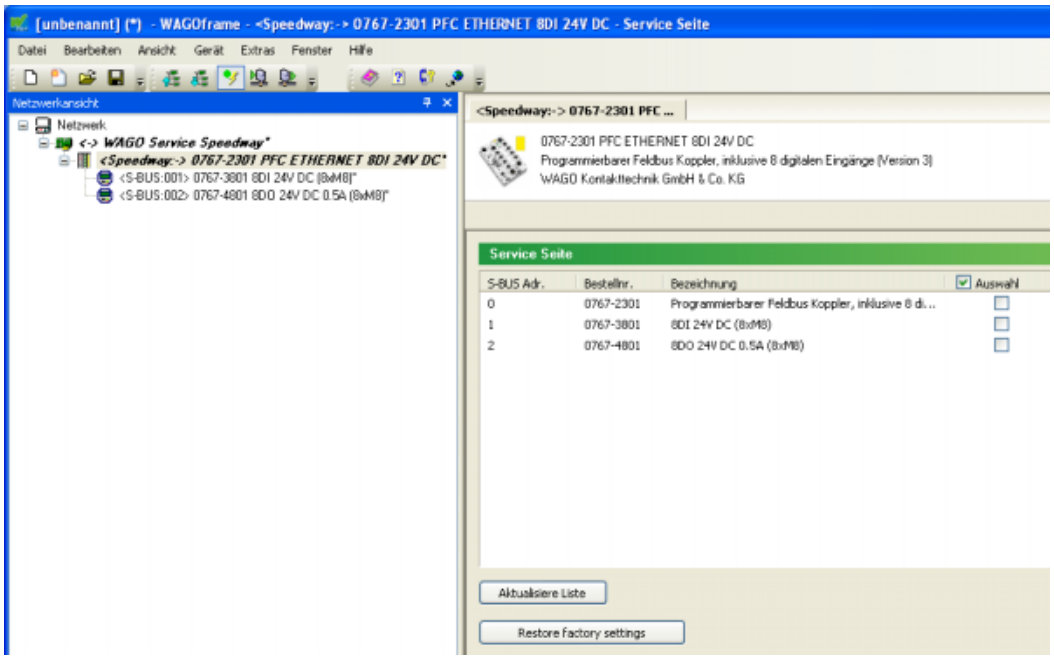


Figure 113: Service page

Table 73: Service page

Parameter/button	Description
S-BUS Addr.	Displays the physical positioni of the devices in the 767 node. 0: Fieldbus coupler 1: First I/O module connected to the fieldbus coupler 2: Second I/O module connected to the fieldbus coupler ...
Item No.	Item number of a 767 component
Designation	Designation of a 767 component
Selection	Selection of the 767 components that should be reset to the default state.
[Update list]	Update the 767 components displayed in the DTM after changing the physical topology (e.g., I/O module removed or added).
[Restore factory]	Reset the selected 767 components to the default state.

17.6.5 User Administration

Use the user administration to change the preset passwords (see Section 14) for the users **guest, user** and **admin** or set the changed passwords back to the default.

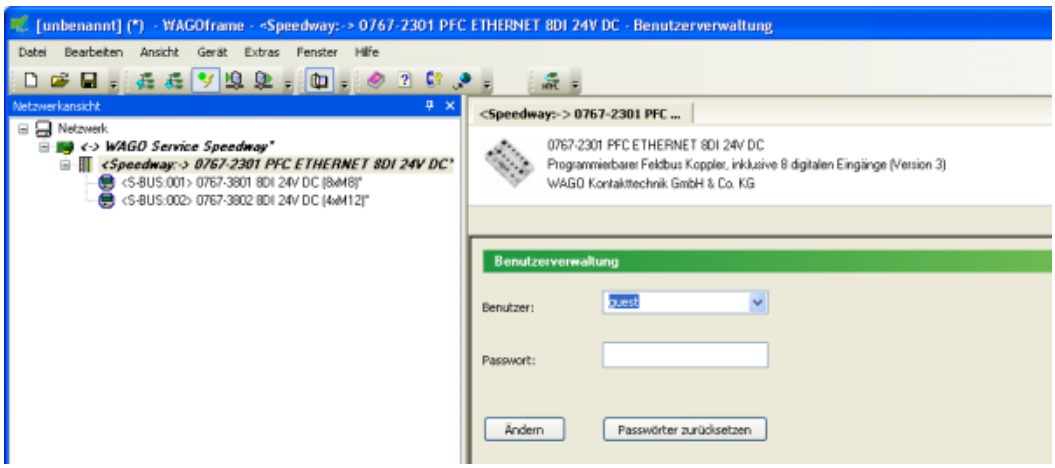


Figure 114: User administration

Table 74: User administration

Parameter/button	Description
User	Select a user whose password should be changed.
Password	Enter a password for the selected user. Only ASCII characters are permitted.
[Change]	Save the new password for the selected user. A dialog appears. Enter **admin** to confirm the new password and click **[OK]**.
[Reset password]	Reset the password to the default state. This must be performed by a **superuser** with a special password. Obtain the password indicating the **ID** listed in the dialog for WAGO Support.

Figure 115: Prompt to reset passwords

17.6.6 File System

Use the file system function to administer the file system on the fieldbus coupler. You can format drives, extract data from the firmware, create or delete directories, upload, download and delete files.

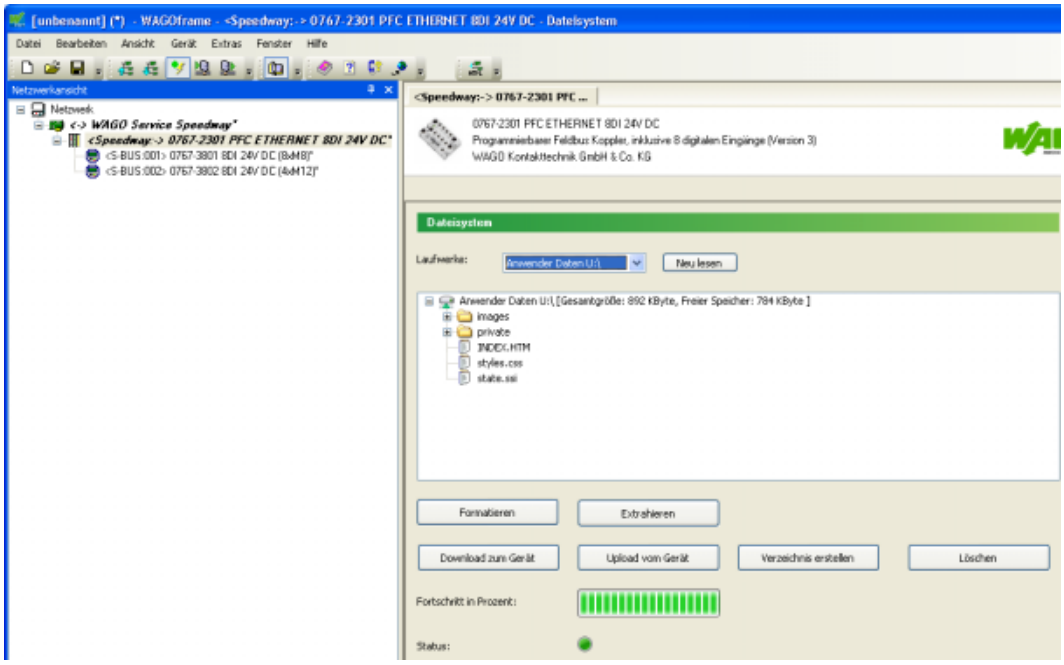


Figure 116: File system

Table 75: File system

Parameter/button	Description
Drives	Select the fieldbus coupler drive: U:\ (WBM pages) P:\ (CoDeSys programm)
[Reread]	Reread the drive currently selected.
[Format]	Format the drive selected under "Drives". All files are deleted.
[Extract]	Extract files from the firmware (WBM, system settings, etc.). **Note:** Any changes made to the existing files may be overwritten when files are extracted again!
[Download to device]	Copy files from the file system of the PC to the fieldbus coupler. **Without selection:** Data is saved to the top level of the file system. **Selected folder:** Data is saved to the selected folder.
[Upload from device]	Copy files from the fieldbus coupler to the PC (e.g., to edit them).
[Create directory]	Create a new director (folder) **Without selection:** The new directory is created at the top level of the file system. **Selected folder:** The new directory is created in the selected folder of the file system.
[Delete]	Delete selected files or directories (folders). You can only delete empty directories (folders).
State	Status display of an operation: Green: The selected procedure is completed. Yellow: The selected procedure is active. Red: Error in the action just executed.

17.6.7 Creating a Network

This involves inserting all 767 components into the topology that are connected to the USB connection.

1. Right-click on a coupler in the "Network View" pane or directly on the "WAGO Service Speedway" DTM.

2. In the context menu, select **Scan** > **Create Network**.
   If the "WAGO Service Speedway" DTM is selected, all components connected to the 767 nodes will be carried over into the topology. If your coupler is selected, only the I/O modules will be attached to the coupler.

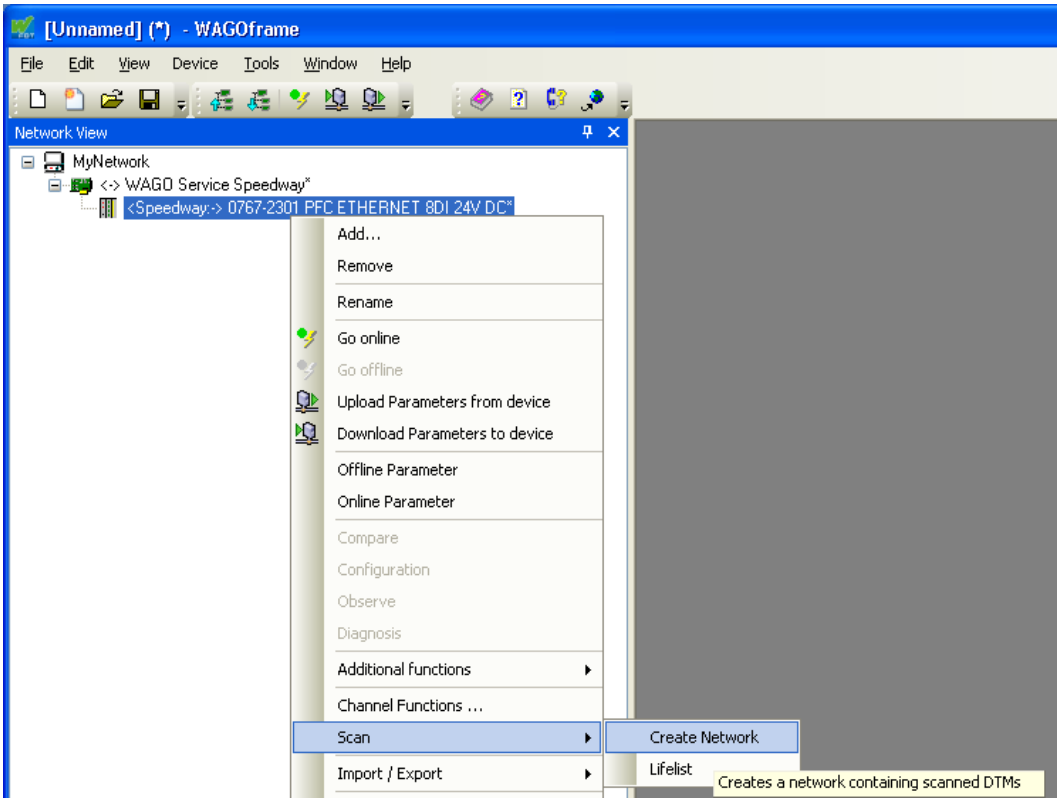


Figure 117: Setting up the network (on I/O modules connected to the fieldbus coupler)

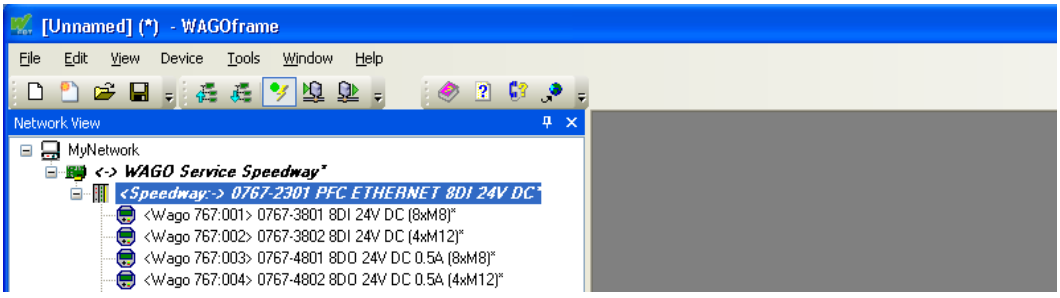


Figure 118: Components connected to the 767 node

17.6.8 Life List

The life list is used to search for all devices connected to the S-BUS in order to select them for configuration directly.

You have the option to use the "WAGO Service Speedway" device driver to search for the connected fieldbus coupler or to use the device driver for the fieldbus coupler to search for I/O modules connected to it.

To search for fieldbus couplers or I/O modules, proceed as follows:

1. Right-click on the respective device driver in the "Network View" pane. In this example, the device driver for the fieldbus coupler is selected to search for the I/O modules connected to it.

2. In the context menu, select **Scan > Life List**. The life list opens.

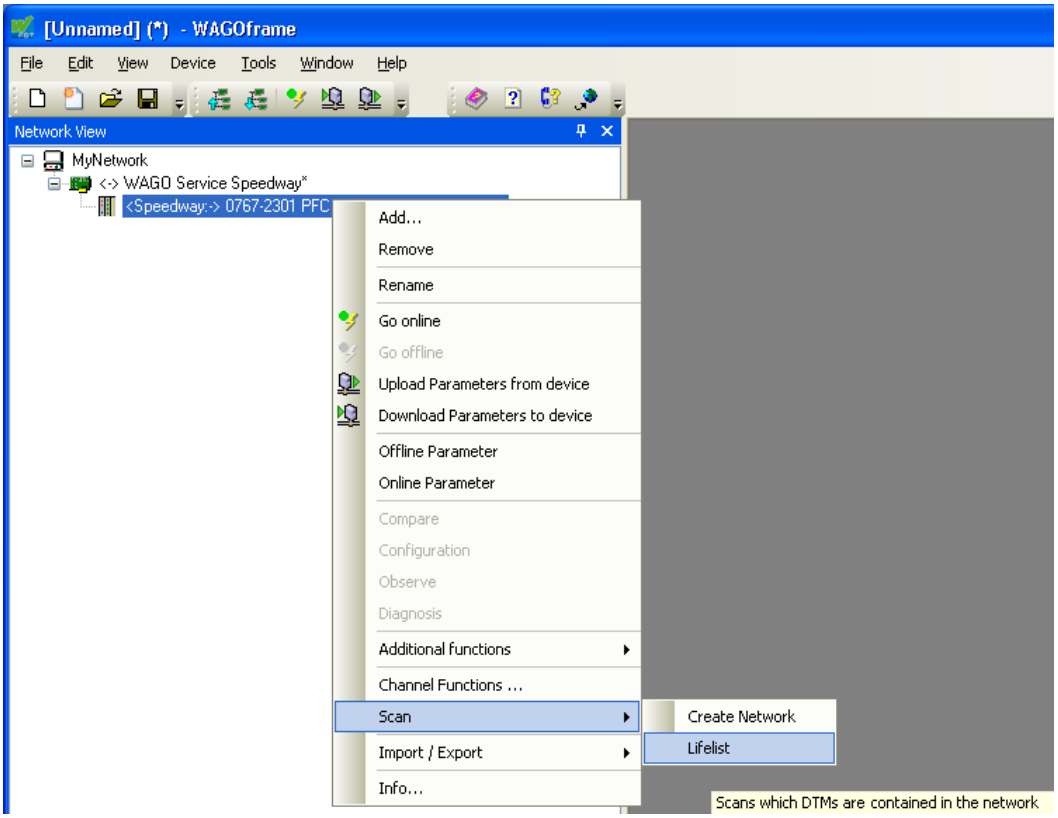


Figure 119: Searching for connected I/O modules using the life list

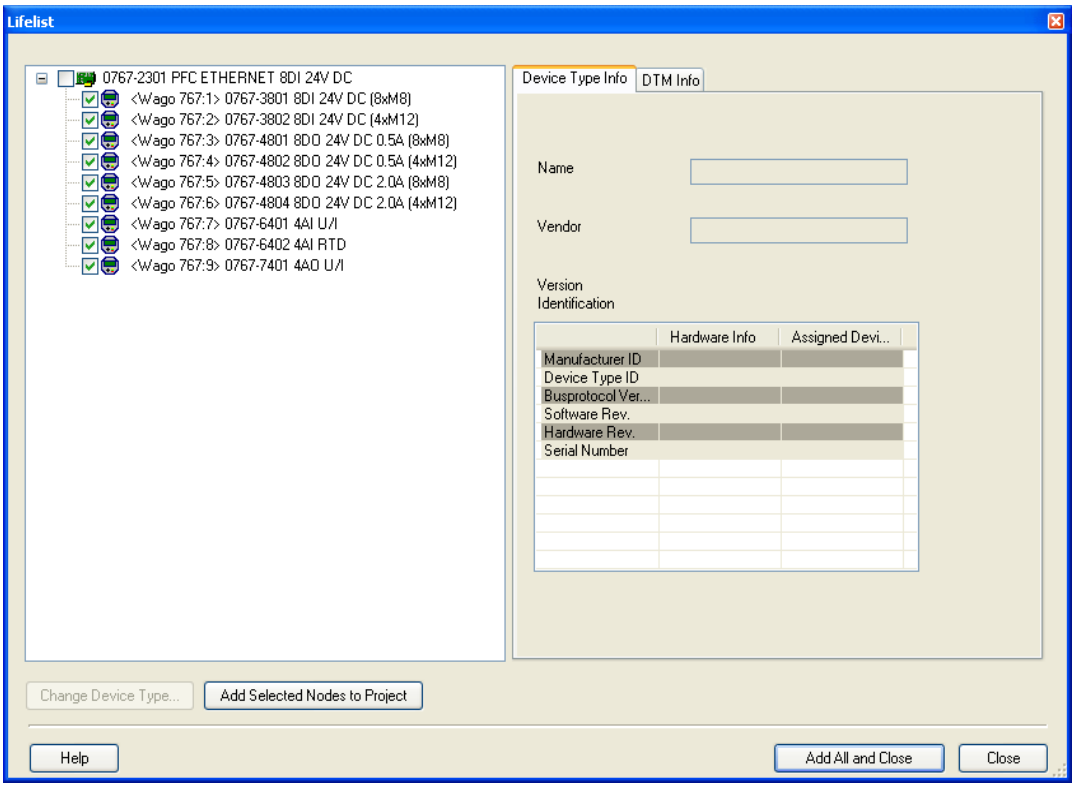


Figure 120: Connected I/O modules of the selected fieldbus coupler

3. Now select from the list those devices that you wish to carry over into the "Network View" for configuration. Click **[Add Selected Nodes to Project]** and then **[Close]**.

17.6.9 System Update

Firmware update for the 767 Series components is performed via system update. To ensure that the fieldbus node remains consistent and executable after updating the firmware, system update must be performed for both fieldbus coupler and connected I/O modules.

NOTICE

System update!
Before updating, observe the following measures to prevent any possible damage to the 767 system:
- The power supply must not be disconnected while updating!
- To exclude any interferences by the fieldbus, the fieldbus cable must be disconnected before updating!

Requirement:
- You have installed the DTM (759-370) WAGOframe
- You have installed the DTM (759-371) WAGO service interface
- You have installed the DTM (759-922) USB driver
- You have installed the DTM (759-362) system update.
- Update packages are available for the connected 767 Series components.

System Update Procedure

To perform a system update for each 767 Series component, please complete the following steps:

1. Read 767 components' parameters and save them on your PC.
2. Update 767 Series components' firmware.
3. Rewrite parameters from your PC to the 767 Series components.
4. Set parameters to valid and finish the procedure.

17.6.9.1 Adding the DTM System Update

To add the DTM system update to WAGOframe, please proceed as follows:

1. In "Network View", right-click on the "WAGO Service Speedway" drevice driver.

2. In the **Add...** context menu, select "Add". The "Add" dialog box opens.

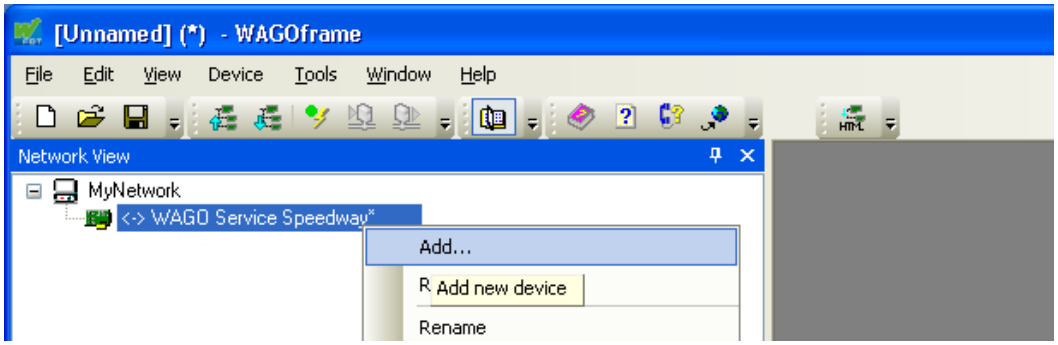


Figure 121: Adding the DTM system update

3. In the "Add" dialog box, select the DTM **0767 System Update**.

4. Click **[OK]** to confirm your selection.

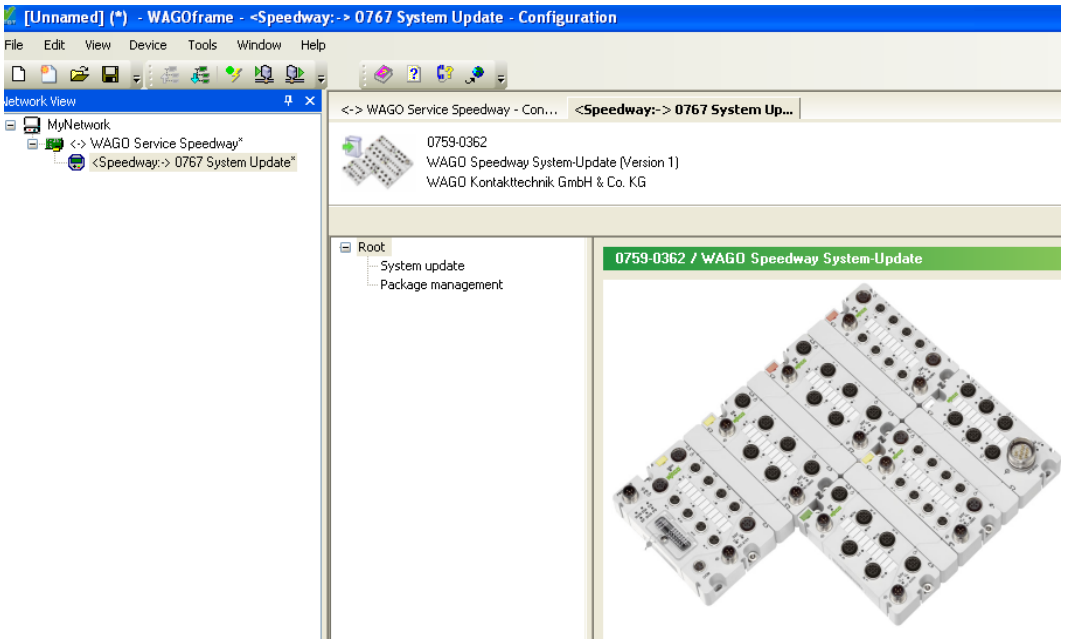


Figure 122: DTM system update

17.6.9.2 Go online to 767 Nodes using Update DTM

The firmware can only be updated when a communication connection exists between Update DTM and 767 node. Please proceed as follows:

1. In the "Network View", right-click on the "<Speedway:> 0767 System Update" device driver.

2. In the context menu, select **Go online**. When the progress bar displays 100% and the entry is displayed in ***bold italics***, the communication connection is established.

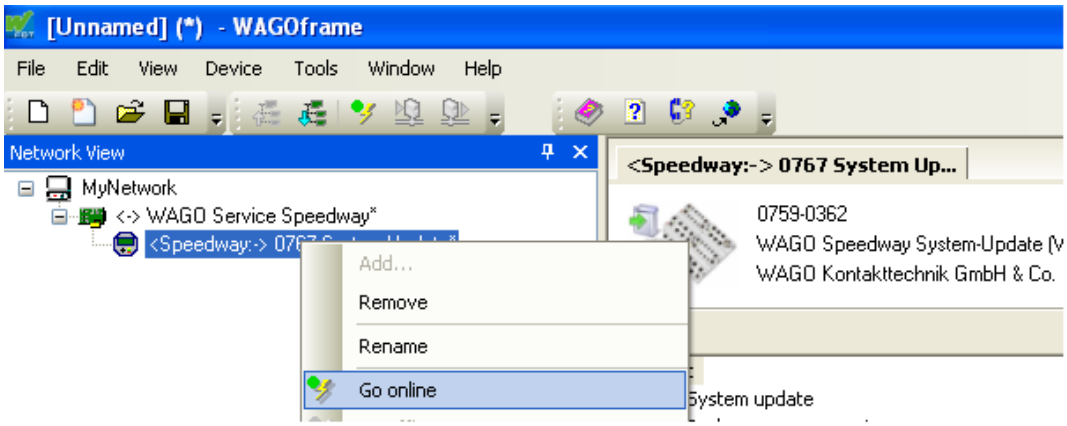


Figure 123: Go online to 767 node

17.6.9.3 Updating the 767 Components

The current firmware is available from WAGO Support. Send an e-mail with the subject "Current Speedway Firmware" and the item number of the respective 767 components to: support@wago.com.

Import firmware packages

To use the received firmware packages, import them into the DTM system update. Please proceed as follows:

1. Save the received firmware packages with the "*.wup" extension to any directory on your PC.

2. Open the DTM user interface by double clicking **"0767 System Update"** in the network view.

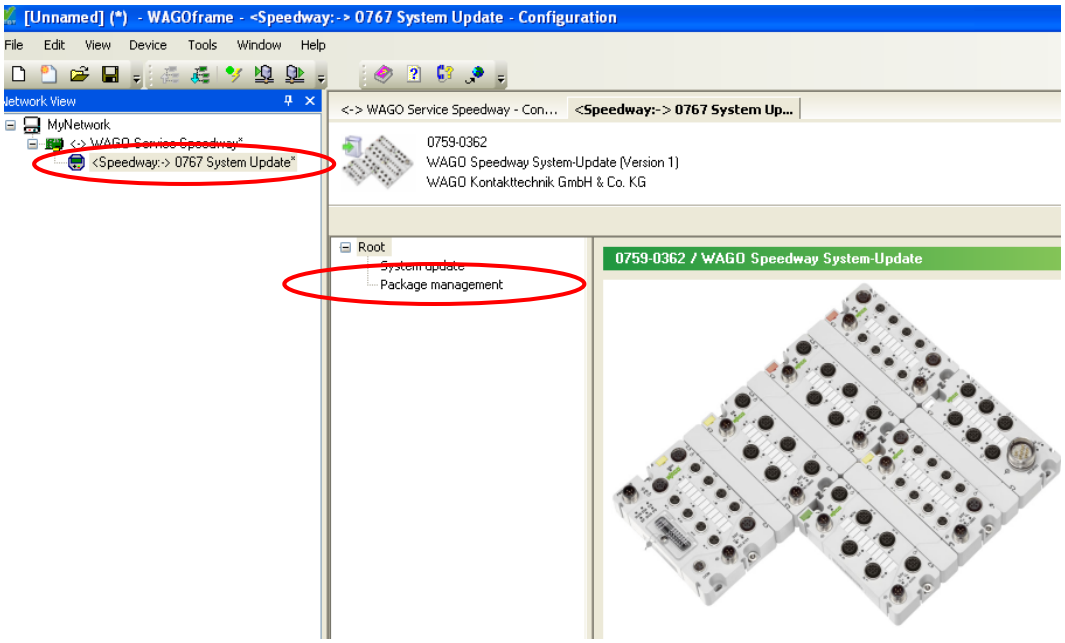


Figure 124: Package management 1

3. Click "Package Management" in the left window of WAGOframe.

4. To import the received firmware files, click **[Import]**. In the window that opens, go to the directory where you have saved the firmware files and select the file to be used. Click **[Open]** to apply the files.

Delete firmware packages

To maintain a clear "Package Management" interface, you can remove unneeded update packages from the view. Please proceed as follows:

1. Select the required firmware files in the right window.

2. Click **[Delete]** to remove the selected firmware packages.

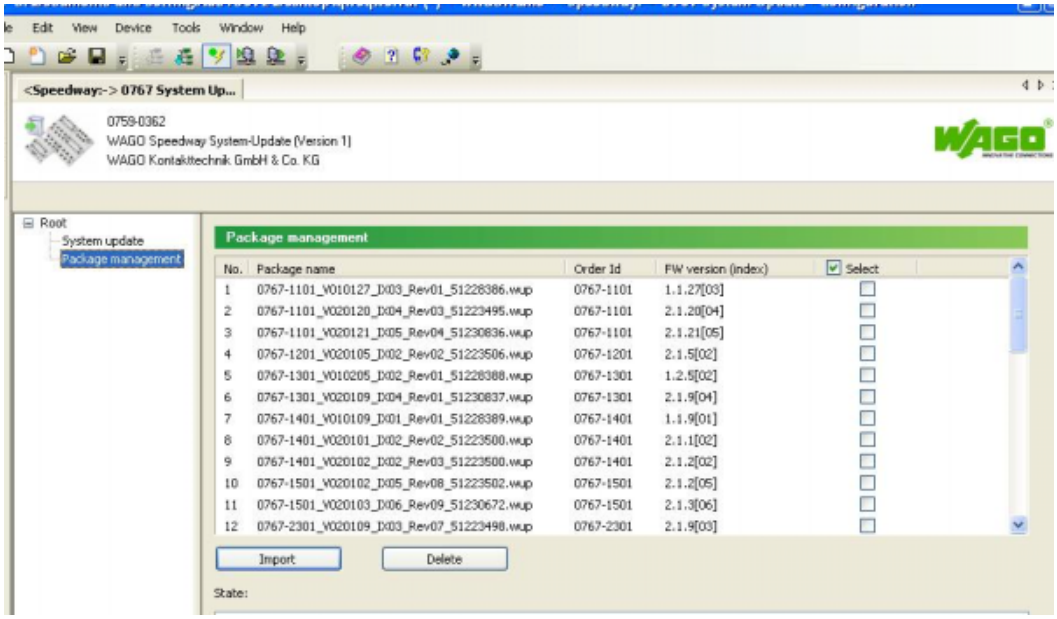


Figure 125: Package management 2

System update

Note

When updating the firmware of the fieldbus coupler, the saved module parameters may be overwritten. Therefore, check your existing configuration after updating the firmware.

Perform the system update here. The module settings you made normally remain unchanged. Otherwise, a corresponding warning message appears. If you still want to update the firmware, the 767 components are returned to their default state.

1. Click "System Update" in the left window.

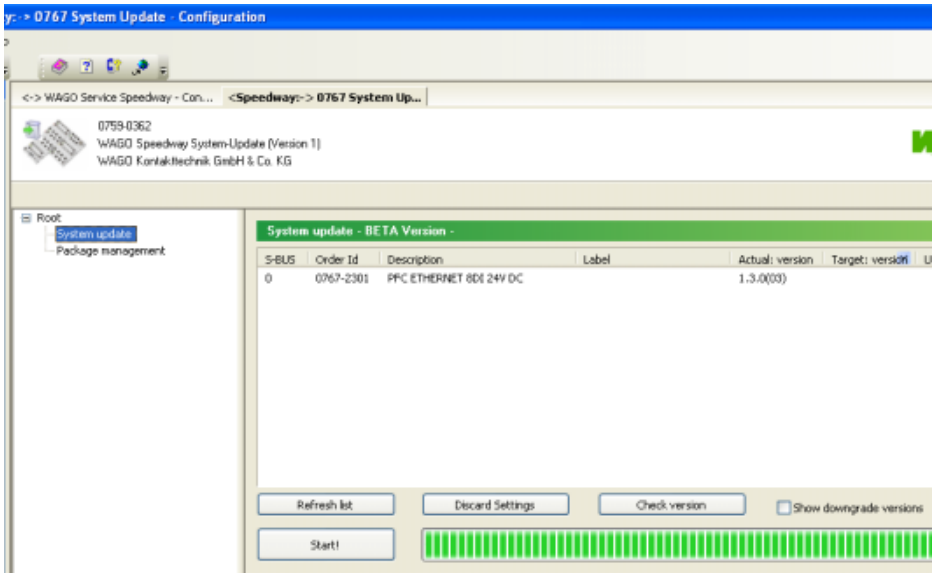


Figure 126: System update 1

2. The fieldbus coupler with all connected I/O modules are listed in the right window. All 767 components that can be updated are pre-selected. If the pre-selection is incorrect or if specific 767 components should not be updated, deselect them ("Update?" column).
   "Actual: Version": Firmware currently present on the device
   "Target: Version": Version of the firmware that should be loaded into the 767 components. If multiple "Target" versions can be selected, select the one relevant to you.

3. Click **[Start!]** to update the system. The 767 components are marked in yellow while being updated.

Note

During the firmware updating process, the fieldbus coupler disconnects each of its COM ports. A PC equipped with Windows 2000 will detect this, and a Windows message appears. This is not an error message. Confirm the message by pressing **[OK]**.

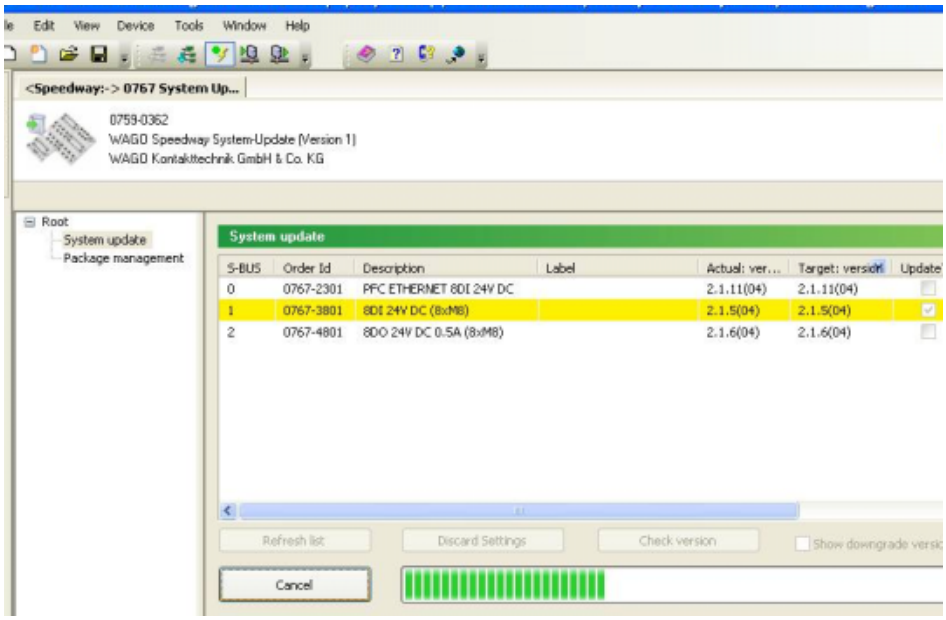


Figure 127: System update 2

Table 76: Buttons

Button	Description
[Update List]	Use this function to read the node design and the view updates.
[Discard Settings]	Discard the selections and settings performed by you.
Display downgrade versions	If this checkbox is selected, the versions to downgrade a device are also displayed in the list of target versions.
[Start!]/[Abort]	Start/abort system udpate.

4. If the system update is finished, the updated 767 components are displayed in green (see figure).

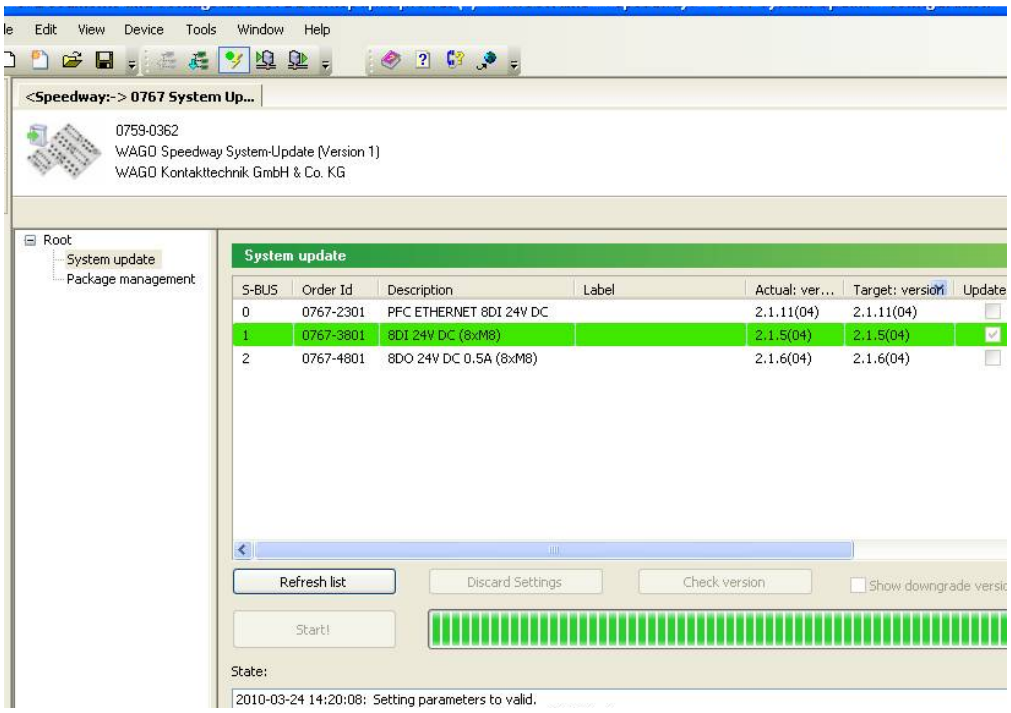


Figure 128: System update 3

During system update, all relevant information is stored on your PC. Update can be repeated subsequently if the system update fails (components are displayed in red). In this case, original parameters remain unchanged.

If the update for a device fails, please contact WAGO Support.

17.7 Parameterization

All parameters listed here can be set using WAGOframe (or another FDT/DTM border application) for the fieldbus coupler. If you use a fieldbus for the parameterization, only certain parameters are configurable, depending on fieldbus type.

The following Sections contain information on the parameters and their descriptions.

To open the parameterization user interface (DTM) of a 767 component, double-click on the appropriate 767 component in the "Network View". The parameterization user interface can also be opened by right-clicking on **Offline Parameter** or **Online Parameter** in the context menu.

If several parameterization user interfaces are open, select one via the corresponding tabs. If you switch a 767 component from the online mode to the offline mode or vice versa, close the parameterization user interface and reopen it.

Depending on the selection of parameterization user interfaces, various buttons are at your disposal:

Table 77: DTM buttons

Button	Description
[Read] *(Online mode only)*	Reads and displays the paremeters found in the 767 components.
[Write] *(Online mode only)*	Writes the modified values to the 767 components.
[Close] *(Online and offline mode)*	Closes the parameterization user interface (DTM).
[Apply] *(Offline mode only)*	Applies the entries in the project. Please note that the project should also be subsequently saved (**File > Save**).
[Help] *(Online and offline mode)*	Opens the online help for an entry that has been previously selected in the DTM (e.g., digital inputs, global setting).
	Shows/hides parameter overview.
	Displays the product data sheet. A PDF reader must be installed on your PC.
	Opens the DTM online help.

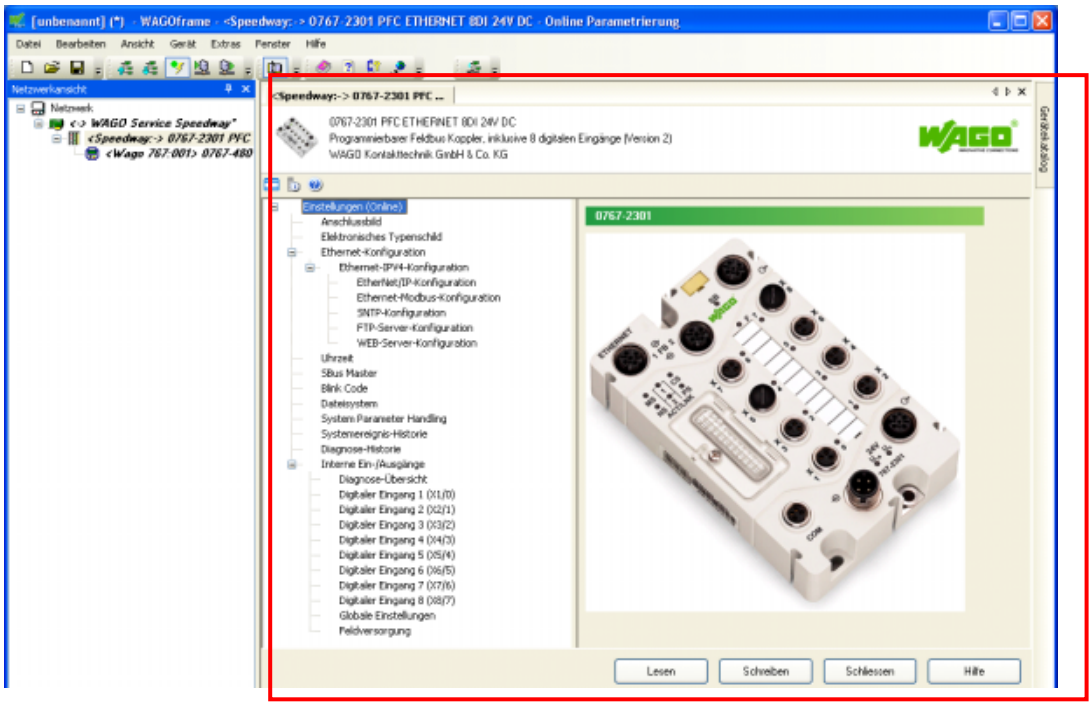


Figure 129: Example of an opened DTM with the available parameters

17.7.1 General Parameters

Electronic Type Label

Table 78: Information about the fieldbus coupler

Parameter	Description
Vendor	Manufacturer
Release index	SW.HW.FL FW: Actual firmware release index When updating the firmware, please note that the firmware release index may not be conformed to the printed firmware release index on the side of the fieldbus coupler. HW: Hardware release index FL: Firmware loader release index
Firmware revision	General information on the module
Order number	
Description	
Serial number	
Date of production	
Designation	Electronic marking field (max. 40 characters)

Date/Time

Table 79: Time settings

Parameter	Description
Daylight savings time	Activate or deactivate daylight savings time here.
Time zone	Select your time zone here.
Actual time	Set the date and time here.
UTC	Display of world time.

To transfer the current local time time from your PC to the fieldbus coupler, click **[Apply PC time]** and then **[Write]**.

DIP switch

Display of the switch positions of the existing DIP switches on the fieldbus coupler

.

S-BUS Master

Table 80: S-BUS-Master

Parameter	Description
Max. restarts	In the input field, enter how often the fieldbus coupler should start up again following a S-BUS disruption. 0x0: No restart 0x1 – 0xFFFFFFFE: Number of restarts 0xFFFFFFFF: Endless restarts
Actual baudrate	Actual baudrate display
Actual S-Bus cycle time	Actual S-Bus cycle time display

Blink code

Table 81: Display of errors

Parameter	Description
Blink code	Display of blink code (error group, error code and error argument).

A list of blink codes and error descriptions, as well as information on error correction, can be found in Section 16.3.3 above.

System Parameter Handling (SPH)

The SPH identifies changes to the configuration of a 767 node and to the automatic configuration of the I/O modules. When a configured I/O module must be replaced due to a defect, you do not need to reconfigure the new I/O module. The stored parameters are automatically transferred to the new I/O module.

Requirement: The exchanged I/O module must have the same item number as the previous one.

To activate the SPH function, proceed as follows:

1. Select "Create nominal system configuration" to store the currently set parameters of the I/O modules in the file system of the fieldbus coupler (parameter database is created).

2. If you consider the firmware status of the new I/O module relevant, select the "FW/HW Release index check I/O modules" checkbox. This requires that the exchanged I/O module has the same firmware and hardware status as the defective I/O module.
   If the firmware status is deviant, a blink code is emitted in the "Error Code" field and on the CS LED of the fieldbus coupler. In this case, the firmware should be updated. For more information, see Section 17.6.9.

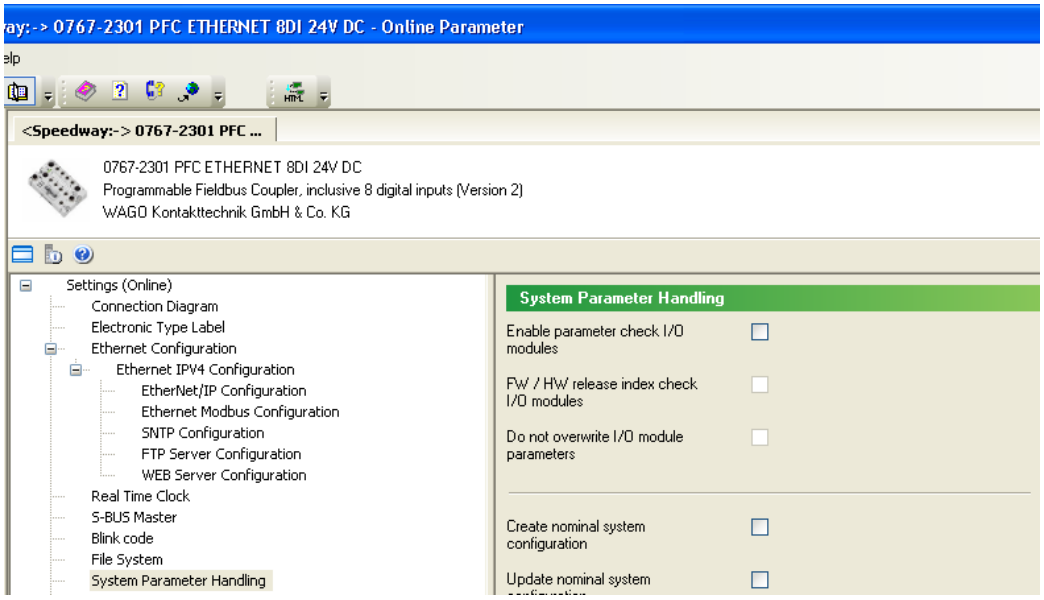


Figure 130: System parameter handling 1

3. Select "Enable parameter check I/O modules" to enable automatic configuration. The dialog box below appears.
   Click **[YES]** if you have not created a database yet.
   Click **[NO]** if you have already created a database (step 1). Otherwise, you will overwrite the existing database.

The stored parameters are now compared with the current parameters at each start-up. If this process reveals that parameters differ, these parameters are automatically overwritten by the stored values in the I/O module.

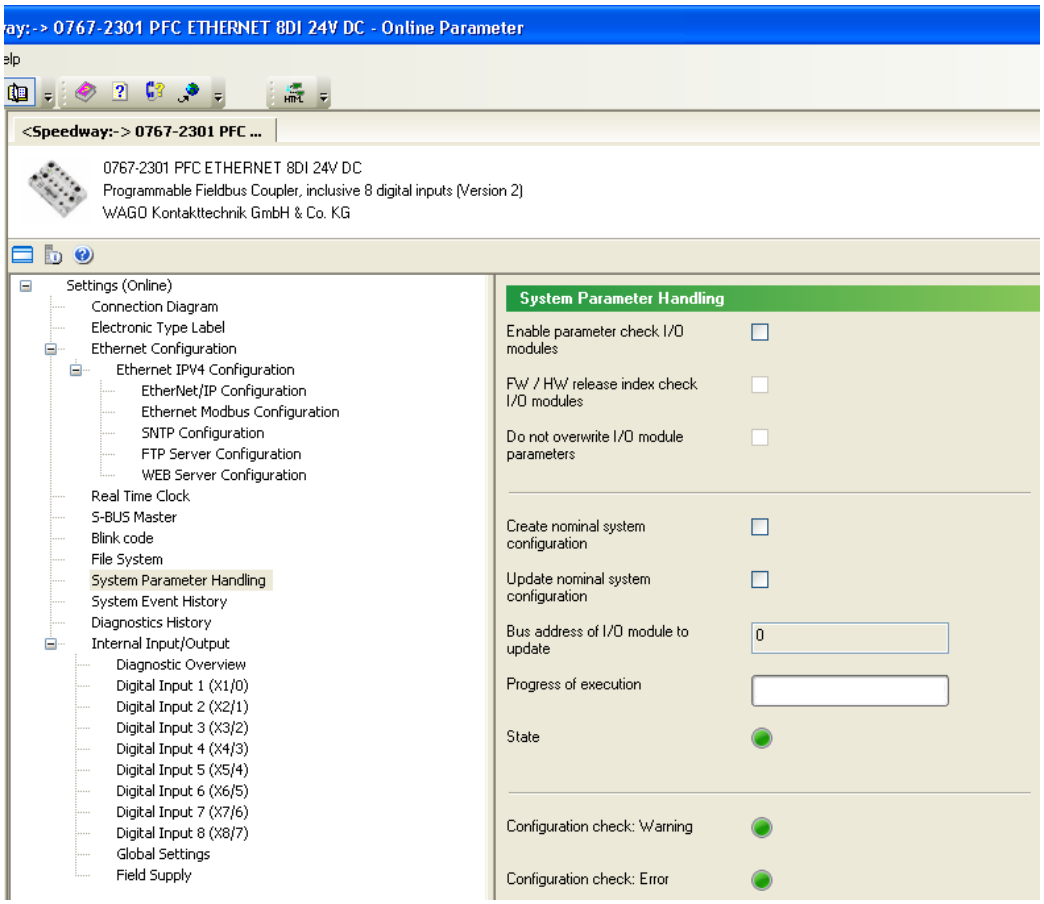


Figure 131: System parameter handling 2

Table 82: Overview of adjustable parameters for "System Parameter Handling"

Parameter	Description
Enable parameter check I/O modules	Select this checkbox to enable the parameter check for the I/O modules and for the integrated inputs of the fieldbus coupler.
FW/HW release index check I/O modules	Select this checkbox to check the FW/HW release index of the IO modules. The checkbox is only activated when "Enable parameter check I/O modules" is checked.
Do not overwrite I/O module parameters	Select this checkbox to prevent an automatic overwrite of the I/O module parameters. The checkbox is only activated when "Enable parameter check I/O modules" is checked.
Create nominal system configuration	Select this checkbox to create the parameter database.
Update nominal system configuration	Select this checkbox to update the system configuration for an I/O module. To do this, select the address of the I/O module from the "Bus address of I/O module to update" field. To update the I/O area of the coupler, select address 0.
Bus address of I/O module to update	Enter here the bus address of the I/O module that is to be updated. The checkbox is only activated when "Enable parameter check I/O modules" is checked. The first I/O module on the fieldbus coupler has bus address 1, the second has bus address 2, etc.
Progress of execution	The progress of the "Create nominal system configuration" and "Update nominal system configuration" parameters is shown here.

Table 82: Overview of adjustable parameters for "System Parameter Handling"

Parameter	Description
State	The status of the "Create nominal system configuration" and "Update nominal system configuration" parameters is shown here: Green: Procedure OK Yellow Procedure active Red Error
Configuration check: Warning	The warning message is displayed when the "I/O module parameter not overwritten" parameter is activated, but a change between stored and current parameters has been detected. Green: No warning during configuration check Red: Warning during configuration check
Configuration check: Error	An error message is shown here when an error is discovered during the configuration check or the automatic configuration (see Section 16.3.3 above): Green: No error during configuration check Red: Error during configuration check
Error code	The error code is displayed here for the configuration check error or warning that has occurred. The display is only activated when a warning or error has arisen.
Bus address of first failed component	The bus address of the first failed I/O module is displayed here. The display is only activated when a warning or error has arisen. If the error cannot be localized, the address "65535" (hex.: 0xFFFF) is displayed.

System Event History

The system event history contains a display of all system events from the fieldbus coupler. These events are also displayed as a blink code via the CS LED (see Section 16.3.3 above). The system events are permanently saved in the fieldbus coupler.

Table 83: Overview of system event history

Parameter	Description
Memory overflow	The status of the circular buffer is displayed here. Green: Circular buffer OK Red: Circular buffer overflow (old events are overwritten)
Entries	All events are displayed in the table. No.: Ongoing number of events Error: Notification type (Incoming: incoming notification Outgoing: outgoing notification) Error code: Error code (see Section 16.3.3) Data: Additional information on the error code Counter: Event counter (increases with repeated occurrence of identical events) Time stamp: Occurrence of events in local time of fieldbus coupler
Clear entries	Select this checkbox to delete all events.

Diagnostics History

All diagnostics of the fieldbus coupler and I/O modules are stored in the diagnostics history. The collective diagnostics are not permanently saved.

Table 84: Overview of diagnostics history

Parameter	Description
Entries	The entries of the diagnostics history are displayed in the table. The last chronological entry is always first in the table. ID: Current number of diagnostic entry S-BUS address: Bus address of the fieldbus coupler/I/O module Param. add.: Parameter address Time stamp: Occurrence of diagnostics in local time of fieldbus coupler Data: Additional information on the diagnostics
Clear entries	Input here the number of entries to be deleted. The oldest entries are deleted first. When the quantity entered is greater than the current number of entries, all entries are deleted.
Stop update	Stop the cyclic reading of the diagnostic messages.
ID of last entry	The consecutive number of the last entry in the history is displayed here.
Maximum size of history	The maximum size of the diagnostics history is displayed here in bytes.
Current history size	The current size of the diagnostics history is displayed here in bytes.

17.7.2 CANopen-Specific Parameters

CANopen Configuration

Table 85: Overview of CANopen-specific parameters

Parameter	Description
CiA vendor ID	Registered manufacturer identification for WAGO Kontakttechnik GmbH & Co. KG assigned by "CAN in Automation": 0x21 (33)
CANopen serial number	Current WAGO serial number
S-BUS error reaction	The response of the fieldbus coupler to an S-BUS error appears: - Set input values to 0 (sets the process value of the input modules to 0)
Process Data access on S-Bus error allowed	Select here whether access to 53xxh, 54xxh and 6xxxh process data objects is permitted via CANopen in the case of an S-BUS disruption.
CANopen node ID	The current node address is displayed here.
Baud rate	The current baud rate is displayed here.
Product code	1501

17.7.3 Parameters of Inputs/Diagnostic Overview

Diagnostic Overview

The currently pending diagnostics existing on the module are displayed here. In this view of the DTM, you can enable simulation of the diagnostics, as well as disable transmission of the diagnostics. When disabling transmission, make sure that the display behavior of each LED changes that indicates the specific diagnostics (Section 16.2). The diagnostic overview is only available in online mode.

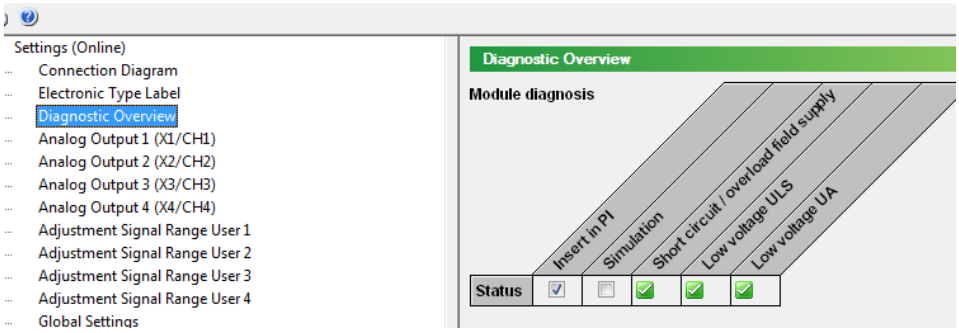


Figure 132: Example of the diagnostic overview of a module (information may differ from the actual module)

Table 86: Diagnostics setup

Parameter	Description
Insert in PI	Displays whether there is a diagnostics: X mark: There is a diagnostic message. Check mark: There is no diagnostic message.
Simulation	Displays whether there is a diagnostics: X mark: There is a diagnostic message. Check mark: There is no diagnostic message.
Status	Displays whether there is a diagnostics: X mark: There is a diagnostic message. Check mark: There is no diagnostic message.

Table 87: Information about existing module diagnostics

Modul Diagnostics	
Diagnostics	**Description**
Short circuit/overload field supply	The device has detected a short circuit or overload of the field supply (only possible when field supply is switched on).
Low voltage ULS	If an undervoltage of the logic or sensor supply (U_{LS}) of < 18 V occurs on the device, a corresponding diagnostic message is generated and the F-LED illuminates. The substitute value strategy that you have parameterized is used on the digital inputs.
Low voltage UA	If an undervoltage of the actuator supply (U_A) of < 18 V occurs on the device, a corresponding diagnostic message is generated and the F-LED illuminates. The undervoltage of the actuator supply has no functional effect on the device.

Parameters of Inputs

Table 88: Overview of adjustable parameters for the digital inputs

Parameter	Description
Designation	Electronic marking field (max. 40 characters) to give the channel of the fieldbus coupler a meaningful name, e.g. "Light switch in section A".
Input value	The current input signal is displayed here. If the "Simulation Input Value" parameter has been selected, select here the input value that is to be simulated: **If simulation is switched off*** **Checkbox unselected:** Input value (0) **Checkbox selected:** Input value (1) **If simulation is switched on** **Checkbox unselected:** Input is simulated with value 0 **Checkbox selected:** Input is simulated with value 1
Signal inverting	You can invert the currently pending input signal here. **Checkbox unselected*:** Input signal is reproduced in the process image in the same manner as at the input **Checkbox selected:** Input signal is reproduced in the process image in an inverted manner
Substitute Value Strategy	This releases the substitute value or the last input value in cases such as a fieldbus interruption. You have the following options: - Switch to substitute value* - Retain last value**
Substitute Value	Enter here the process value that is used in case of an error. In the case of an error (fieldbus interruption), this value is used with the "Switch to Substitute Value" substitute value strategy. **Checkbox unselected:** 0* **Checkbox selected:** 1
Filter time	Set the input filter for the measured signals here. You have the following options: - none - 0.1 ms - 0.5 ms - 3 ms* - 15 ms - 20 ms

Table 88: Overview of adjustable parameters for the digital inputs

Parameter	Description
Simulation input value	This simulation can be used to simulate input values. **Checkbox unselected*:** The value at the inputs is carried over into the process image. **Checkbox selected:** The simulation value is inserted into the process data independent of the input value. The "Input Value" parameter is selected.

* Default setting

17.7.4 Global Settings

Table 89: Overview of parameters for the entire fieldbus coupler

Parameter	Description
Simulation diagnostic	If the checkbox is selected, you can simulate a low voltage diagnostic. To generate a low voltage diagnostic, one or both of the two checkboxes "Low voltage ULS" and "Low voltage UA" must be selected. *Default setting: unselected*
Low voltage ULS	In the case of an undervoltage of the logic and sensor supply (ULS) or the actuator supply (UA), the corresponding diagnostic is displayed here.
Low voltage UA	

17.7.5 Parameters of Field Supply

Table 90: Overview of adjustable parameters for the field supply

Parameter	Description
Enable field supply	Switch on the field supply (24VDC) for the sensors here. *Default setting: selected*
Autorestart delay	In the event of a short circuit, the sensor supply is switched off for a parameterizable amount of time. Here, enter the delay time (in 100-ms increments); after this time, the sensor supply is restarted. If the short circuit still exists, the process is repeated.
Simulation diagnostic	The simulation can be used to simulate a short circuit. *Default setting: unselected*
Short circuit/overload	If simulation is deactivated, the respective error is displayed upon emergence. If simulation is activated, you can simulate one of the errors by selecting the appropriate parameter.

18 Maintenance and Service

This section contains information on maintenance and service.

18.1 Updating the Firmware

Information on updating the firmware can be found in Section 17.6.9.

18.2 Replacing the Fieldbus Coupler

To replace the fieldbus coupler, e.g., to change variants, proceed as described in the following sections.

18.2.1 Disconnecting the Cables

Before removing the connectors, clean the fieldbus coupler to ensure that no dirt or other material comes in contact with the connections. This can lead to damage of the contacts.

To unplug the cables, proceed as follows:

1. Disconnect the power supply from those devices on which you have mounted the fieldbus coupler.

⚠ CAUTION

Hot connection sockets!
Even when taking into account derating, high surface temperatures on the metallic connection sockets and on the enclosure can arise during operation. If the 767 Series component has been in operation, allow it to cool before touching or use gloves.

2. Unscrew all screw connections and remove the cables.

18.2.2 Removing the Fieldbus Coupler from your System

To remove the fieldbus coupler from the framework of your system, proceed as follows:

1. Disconnect the power supply from those devices on which you have mounted the fieldbus coupler.

2. Release the fieldbus coupler from the framework of your system by unscrewing the M4 screws.

18.2.3 Removing the Fieldbus Coupler from the Carrier Rail

In order to keep a clear representation, the mounting rail adapter in the figure below (B, C) is shown without the fieldbus coupler.

If the fieldbus coupler is mounted on a mounting rail, proceed with the removal as follows:

1. Disconnect the power supply from those devices on which you have mounted the fieldbus coupler.

2. To remove the fieldbus coupler, press down the release eyebolt of the mounting rail adapter using a slot screwdriver (B) and remove it from the mounting rail (C).

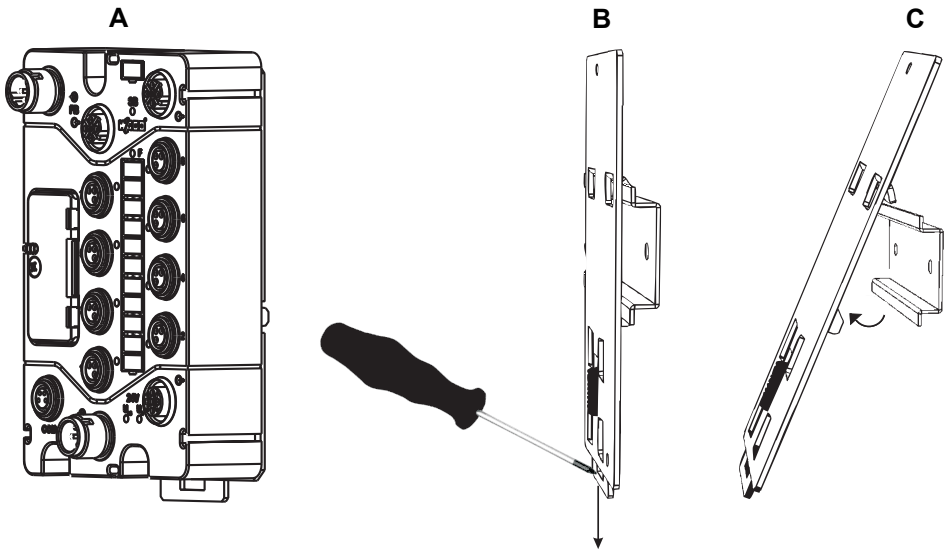


Figure 133: Removing the fieldbus coupler (with the mounting rail adapter) from the mounting rail

18.2.4 Removing the Fieldbus Coupler from the Profile Adapter

If the fieldbus coupler is mounted on a profile adapter, proceed with the removal as follows:

1. Disconnect the power supply from the device on which you have mounted the fieldbus coupler before attempting to remove.

2. Unscrew the screws on which the nuts are fastened and remove the fieldbus coupler from the profile rail of your system.

3. Unscrew the screws that connect the fieldbus coupler with the profile adapter.

18.2.5 Connecting a New Fieldbus Coupler

To connect the fieldbus coupler, proceed as described in Sections 5 through 7.

18.3 Disposal

Do not dispose of the 767 components in the household waste; observe the laws which apply to them. You can also contact a certified waste management company.

19 Appendix

19.1 CANopen Objects of the Fieldbus Coupler

19.1.1 Object 0x1000, Device Type

Object 0x1000 indicates the implemented device profile. The CANopen fieldbus coupler implemented the "Device Profile for Generic I/O Modules" (device profile no. 401). In addition, the value in index 0x1000 indicates which type of I/O modules are connected.

Table 91: Object 0x1000

Index	Sub-Index	Name	Data Type	Attribute	Default Value
0x1000	0	Device type	Unsigned32	RO	-

This object is structured as follows:

MSB			LSB
0000.0000	0000.**4321** **1** = 1 When at least one digital input module is connected. **2** = 1 When at least one digital output module is connected. **3** = 1 When at least one analog input module is connected. **4** = 1 When at least one analog output module is connected.	"Device Profile" Number 0x01 (high-byte)	"Device Profile Number" 0x91 (low-byte)

19.1.2 Object 0x1001, Error Register

Internal errors are represented in this register. This is a part of the emergency telegram.

Table 92: Object 0x1001

Index	Sub-Index	Name	Data Type	Attribute	Default Value
0x1001	0	Error register	Unsigned8	RO	-

If an error has arisen, bit 0 is always set. Additional bits that are set specify the error more precisely.

The object is structured as follows:

Table 93: Explanation of the object

Bit	Description
0	General error
1	Current
2	Voltage
3	Temperature
4	Communication
5	Device profile specific
6	Reserved
7	Manufacturer-specific

19.1.3 Object 0x1003, Pre-Defined Error Field

Sub-index 0 contains the number of currently stored errors. A maximum of 20 error entries is possible. If a new error arises, it is always entered into sub-index 1. all previous errors are shifted by one sub-index. If more than 20 errors occur, the error that was stored first is deleted.

Table 94: Object 0x1003

Index	Sub-Index	Name	Data Type	Attribute	Default Value
0x1003	0	Number of errors	Unsigned8	RW	0
	1	Standard error field	Unsigned32	RO	-
	…	…	…	…	…
	20	Standard error field	Unsigned32	RO	-

The object is structured as follows:

Bit 31	Bit 16	Bit 5	Bit 0
Additional information		Error code	

The additional information is consistent with the first 2 bytes of the "Manufacturer-Specific Error Code" from the emergency telegram. The error code is identical to the one in the emergency telegram. By writing a 0 into sub-index 0, the complete error memory is deleted.

19.1.4 Object 0x1005, COB-ID SYNC Message

The object defines the COB ID for the synchronization telegram.

Table 95: Object 0x1005

Index	Sub-Index	Name	Data Type	Attribute	Default Value
0x1005	0	COB-ID SYNC	Unsigned32	RW	0x00000080

This object is structured as follows:

Bit 31	Bit 11	Bit 10	Bit 0
Reserved (always 0)		COB ID	

19.1.5 Object 0x1006, Communication Cycle Period

The object defines the maximum time (in seconds) for two consecutive SYNC telegrams. The internal release is 2 ms. If the value is 0, SYNC monitoring does not take place.

Table 96: Object 0x1006

Index	Sub-Index	Name	Data Type	Attribute	Default Value
0x1006	0	Communication cycle period	Unsigned32	RW	0

19.1.6 Object 0x1008, Manufacturer Device Name

The object specifies the name of the fieldbus coupler.

Table 97: Object 0x1008

Index	Sub-Index	Name	Data Type	Attribute	Default Value
0x1008	0	Manufacturer device name	Visible string	RO	767-2501

19.1.7 Object 0x1009, Manufacturer Hardware Version

The object specifies the current hardware version of the fieldbus coupler.

Table 98: Object 0x1009

Index	Sub-Index	Name	Data Type	Attribute	Default Value
0x1009	0	Manufacturer hardware version	Visible string	RO	Current hardware version

19.1.8 Object 0x100A, Manufacturer Software Version

The object specifies the current software version of the fieldbus coupler.

Table 99: Object 0x100A

Index	Sub-Index	Name	Data Type	Attribute	Default Value
0x100A	0	Manufacturer software version	Visible string	RO	Current software version

19.1.9 Object 0x100C, Guard Time

The object specifies the "Guarding Time" in milliseconds. An NMT master requests the state of the NMT slave in a cyclical manner. The time between two requests is the "Guard Time."

Table 100: Object 0x100C

Index	Sub-Index	Name	Data Type	Attribute	Default Value
0x100C	0	Guard time	Unsigned16	RW	0

19.1.10 Object 0x100D, Life Time Factor

The "Life Time Factor" is a part of the "Node Guarding Protocol." The NMT slave (fieldbus coupler) checks whether there was communication between between them within the "Node Life Time" ("Guard Time" multiplied by the "Life Time Factor"). If this was not the case, the NMT slave must assume that the NMT master is unavailable. It then triggers a "Life Guarding Event." If the "Node Life Time" is 0, monitoring does not take place.

Table 101: Object 0x100D

Index	Sub-Index	Name	Data Type	Attribute	Default Value
0x100D	0	Life time factor	Unsigned8	RW	0

19.1.11 Object 0x1010, Store Parameters

This object makes it possible to permanently save the configuration you have carried out. To do this, write the "save" signature (lower-case letters ASCII - MSB – 0x65 76 61 73 - LSB) into index 0x1010, sub-index 1. The save process runs in the background and lasts approximately three seconds.

After saving, the SDO reply telegram is sent. During the save process, it is still possible to communicate via the SDO. An error message only appears if another attempt to save is made while the previous command is not yet completed. Likewise, it is impossible to save when "Restore" is still active.

The error message "Modified HW Configuration" is no longer sent when the fieldbus coupler is switched on and a saved configuration is available that matches the connected I/O modules.

Table 102: Object 0x1010

Index	Sub-Index	Name	Data Type	Attribute	Default Value
0x1010	0	Maximum supported sub-index	Unsigned8	RO	1
	1	Save all parameters	Unsigned32	RW	1

Note

If only the station address is changed via the DIP switch after saving a configuration, the saved configuration continues to be used. This means that all module-ID-specific entries in the object directory (objects that depend on the station address and include the "rw" attribute) report with the older values (e.g., "Emergency ID," etc.).

19.1.12 Object 0x1011, Restore Default Parameters

This object makes it possible to reset the saved configuration to the default setting. Sub-indices 2 and 3 are not supported.

The load command is processed in the background and lasts for approximately three seconds. Upon completion, an SDO reply telegram is sent. SDOs can be used to communicated during execution. An error message only appears if another attempt is made to send a load command when the previous command has not yet been completed. Likewise, it is impossible to release a load command when the save command has not been completed.

Table 103: Object 0x1011

Index	Sub-Index	Name	Data Type	Attribute	Default Value
0x1011	0	Maximum supported sub-index	Unsigned8	RO	4
	1	Set all parameters to default values	Unsigned32	RW	1
	2	-	Unsigned32	RW	0
	3	-	Unsigned32	RW	0
	4	Set all parameters to default values once	Unsigned32	RW	1

19.1.12.1 Sub-Index 1 - Permanent Creation of Default Parameters

By writing the "load" signature (lower-case letters ASCII - MSB 0x64 0x61 0x6F 0x6C LSB) into index 0x1011, sub-index 1, the factory default settings are loaded after fieldbus coupler start-up and at each restart (up to the next SAVE command).

19.1.12.2 Sub-Index 4 – One-Time Creation of Default Parameters

By writing the "load" signature (lower-case letters ASCII - MSB 0x64 0x61 0x6F 0x6C LSB) into index 0x1011, sub-index 4, the factory default settings are loaded once after fieldbus coupler start-up. The saved configuration is imported after each additional start-up. This can be used in the development phase to obtain a comparison between the saved and default configurations.

Example:

To replace the saved configuration once with the default configuration, proceed as follows:

1. Write the "load" signature (ASCII - MSB 0x64 0x61 0x6F 0x6C LSB) into index 0x1011, sub-index 4.

2. Restart the fieldbus coupler by disconnecting the power supply and then reconnecting it.

3. The fieldbus coupler starts using the default values.
   In this state, reusing the "load" command (index 0x1011, sub-index 4) is not allowed. In order to use the save configuration, the fieldbus coupler must be restarted.

19.1.13 Object 0x1014, COB ID Emergency Object

The object defines the COB ID for the EMCY telegram telegram.

Table 104: Object 0x1014

Index	Sub-Index	Name	Data Type	Attribute	Default Value
0x1014	0	COB ID EMCY	Unsigned32	RW	0x80+ModuleID

Structure:
In order to enter a new COB ID, bit 31 must first be set to 1; changing a valid COB ID (bit 31 = 0) is prohibited per DS301.

The object is structured as follows:

Bit 31	Bit 30	Bit 11	Bit 10	Bit 0
0/1 Valid/invalid	Reserved (always 0)		COB ID	

19.1.14 Object 0x1015, Inhibit Time Emergency Object

This object specifies the minimum time that must elapse before another emergency object is sent. To disable delayed sending, set the "Inhibit Time" to 0.

If delayed sending is enabled, the entries are entered into a send memory. The send memory contains 20 entries. If this number is exceeded, an emergency is sent immediately that reports the exceeded limit. One time unit amounts to 100µs.

Table 105: Object 0x1015

Index	Sub-Index	Name	Data Type	Attribute	Default Value
0x1015	0	Inhibit time EMCY	Unsigned16	RW	0

Example:
Min. time interval between two EMCY: 30 ms
Index 0x1015 = 300 = 0x12C

19.1.15 Object 0x1016, Consumer Heartbeat Time

This entry enables monitoring of up to five I/O modules. It also checks whether each I/O module that is defined in this object has created a "Heartbeat" within the set time. If the set time has been exceeded, a heartbeat event is triggered. The "Heartbeat Time" is entered in milliseconds. For value 0, monitoring is disabled.

Table 106: Object 0x1016

Index	Sub-Index	Name	Data Type	Attribute	Default Value
0x1016	0	Maximum I/O modules that can be monitored	Unsigned8	RO	5
	1	1st heartbeat time entry	Unsigned32	RW	0
	2	2nd heartbeat time entry	Unsigned32	RW	0
	3	3rd heartbeat time entry	Unsigned32	RW	0
	4	4th heartbeat time entry	Unsigned32	RW	0
	5	5th heartbeat time entry	Unsigned32	RW	0

The object is structured as follows:

	MSB		LSB
Bit	31-24	23-16	15-0
Value	Reserved	Module ID	Heartbeat time
Data Type	-	Unsigned8	Unsigned16

19.1.16 Object 0x1017, Producer Heartbeat Time

The object defines the time (in milliseconds) between two sent heartbeat telegrams. If the time is 0, no "Heartbeat" is sent. As soon as a value other than 0 is entered, the heartbeat transfer begins.

Table 107: Object 0x1017

Index	Sub-Index	Name	Data Type	Attribute	Default Value
0x1017	0	Producer heartbeat time	Unsigned16	RW	0

19.1.17 Object 0x1018, Identity Object

The object specifies the CANopen device that is being used. The manufacturer ID contains a unique number for each manufacturer; WAGO's is 33. The second part of the item number is the device description. The revision number contains a specific CANopen behavior, and the major revision number contains the CANopen functionality. If this functionality is modified, the major revision number is increased. The minor revision number can be used to distinguish between different versions that have the same CANopen behavior.

Table 108: Object 0x1018

Index	Sub-Index	Name	Data Type	Attribute	Default Value
0x1018	0	Maximum supported entries	Unsigned8	RO	4
	1	Manufacturer ID	Unsigned32	RO	33
	2	Device description	Unsigned32	RO	2501
	3	Revision number	Unsigned32	RO	Current version number
	4	Serial number	Unsigned32	RO	Current serial number

The object is structured as follows:

Bit 31	Bit 16	Bit 5	Bit 0
Major rev. #		Minor rev. #	

19.1.18 Object 0x1029, Error Behavior

This object is used to determine which to state the I/O module switches in the case of a communication error (e.g., "Node Guarding" failure).

Table 109: Object 0x1029

Index	Sub-Index	Name	Data Type	Attribute	Default Value
0x1029	0	Maximum supported sub-index	Unsigned8	RO	1
	1	Communication error	Unsigned8	RW	0

The object is structured as follows:

Table 110: Object 0x1029

Communication Error	Action
0	Switches to the pre-operational state (only if the fieldbus coupler is in the operation state)
1	No state change
2	Switch to stopped state

19.1.19 Object 0x1200 – 0x1201, Server SDO

This object is used to access entries in the object directory. The second SDO is not active in the delivery state. It is not allowed to change the COB IDs of the second SDO if they are active (bit 31 = 0).

Table 111: Object 0x1200 – 0x1201

Index	Sub-Index	Name	Data Type	Attribute	Default Value
0x1200 through 0x1201	0	Maximum supported entries	Unsigned8	RO	2
	1	COB ID Client server (Rx)	Unsigned32	Index 0x1200 RO Index 0x1201 RW	Index 0x1200 0x600+ModuleID Index 0x1201 0x80000000
	2	COB ID Server client (Tx)	Unsigned32	Index 0x1200 RO Index 0x1201 RW	Index 0x1200 0x580+ModuleID Index 0x1201 0x80000000

Structure of sub-indices 1 and 2:

Bit 31	Bit 30	Bit 11	Bit 10	Bit 0
0/1 valid/invalid		Reserved (always 0)	COB ID	

19.1.20 Object 0x1280 – 0x128F, Client SDO

This object allows SDO master operation for the fieldbus coupler, which supports 16 client SDOs and transmits one SDO client. Another SDO client can not be transmitted before transmission is complete.

Table 112: Object 0x1280 – 0x128F

Index	Sub-Index	Name	Data Type	Attribute	Default Value
0x1280 to 0x128F	0	Max. supported Entries	Unsigned8	RO	3
	1	COB-ID Client -> Server (Tx)	Unsigned32	RW	0x80000000
	2	COB-ID Server -> Client (Rx)	Unsigned32	RW	0x80000000
	3	Module ID of the SDO server	Unsigned8	RW	0

Design of the COB-ID:

Bit 31	Bit 30	Bit 11	Bit 10	Bit 0
0/1 valid/invalid		reserved (always 0)	COB-ID	

19.1.21 Object 0x1400 – 0x141F, Receive PDO Communication Parameter

This object is used to set the communication parameters of the RxPDO. Five RxPDOs are supported. The default COB IDs from the first four PDOs are preassigned per DS301. All other PDOs are disabled. If not all default PDOs are used (e.g., fewer I/O modules are connected), the unused default PDOs are also disabled.

Table 113: Object 0x1400 – 0x141F

Index	Sub-Index	Name	Data Type	Attribute	Default Value
0x1400 through 0x141F	0	Maximum supported entries	Unsigned8	RO	2
	1	COB ID	Unsigned32	RW	Index 0x1400 0x200+ModuleID Index 0x1401 0x300+ModuleID Index 0x1402 0x400+ModuleID Index 0x1403 0x500+ModuleID Index 0x1404 0x80000000
	2	Transfer type	Unsigned8	RW	255

Structure of sub-index 1, COB ID:

Bit 31	Bit 30	Bit 29 – Bit 11	Bit 10 – Bit 0
0/1 Valid/invalid	0/1 RTR allowed/not allowed	Reserved (always 0)	COB ID

In order to enter a new COB ID, bit 31 must first be set to 1; changing a valid COB ID (bit 31 = 0) is prohibited per DS301.

One data transfer mode (transmission type in index "Communication Parameter") can be defined for each PDO. Digital and analog inputs are transferred as standard in the "Change of Value" (COV) procedure. The following table explains the type of transfer depending on the set "Transmission Type."

Table 114: "Transmission Type"

Trans-mission Type	**PDO Transmission**						
	Cyclic	Acyclic	Synchron ous	Asynchrono us	RTR only	TxPDO (inputs)	RxPDO (outputs)
0		X	X			COV transfer at each SYNC	After each SYNC, outputs are set as demanded by the last received PDO.
Trans-mission Type	**PDO Transmission**						
1 – 240	X		X			Transfer for each x-th SYNC (x = 1 through 240)	After each SYNC, outputs are set as demanded by the last received PDO.
241 – 251	Reserved						
252			X		X	Data is newly imported but not sent at each SYNC. Request via RTR	Not supported
253				X	X	Request via RTR	COV
254				X		COV*	COV
255				X		COV*	COV

* The data is sent in the interval of the set "Inhibit Time"

19.1.22 Object 0x1600 – 0x161F, Receive PDO Mapping Parameter

This object is used to determine which data is sent via the Tx-PDO. Sub-index 0 contains the number of objects that are valid for the PDO.

Table 115: Object 0x1600 – 0x161F

Index	Sub-Index	Name	Data Type	Attribute	Default Value
0x1600 through 0x161F	0	Number of objects inserted	Unsigned8	RW	-
	1 through 8	1st object through 8th object	Unsigned32	RW	-

Objects 1 – 8 are structured as follows:

Bit 31	Bit 16	Bit 15	Bit 8	Bit 7	Bit 0
Index		Sub-Index		Object length	

Index: Index of object to be transferred
Sub index: Sub-index of object to be transferred
Object length: Size of object in bits
Because a maximum of 8 bytes can be transferred in one PDO, the sum of the valid object lengths shall not exceed 64 (8 bytes x 8 bits).

19.1.23 Object 0x1800 – 0x181F, Transmit PDO Communication Parameter

This object is used to set the communication parameters of the TxPDO. Five TxPDOs are supported. The default COB IDs from the first four PDOs are preassigned per DS301. All other PDOs are disabled. If not all default PDOs are used (e.g., fewer I/O modules are connected), the unused default PDOs are also disabled.

Table 116: Object 0x1800 – 0x181F

Index	Sub-Index	Name	Data Type	Attribute	Default Value
0x1800 through 0x181F	0	Maximum supported entries	Unsigned8	RO	5
	1	COB ID	Unsigned32	RW	Index 0x1800 0x180+ModuleID Index 0x1801 0x280+ModuleID Index 0x1802 0x380+ModuleID Index 0x1803 0x480h+ModuleID Index 0x1804 0x80000000
	2	Transfer type	Unsigned8	RW	255
	3	"Inhibit Time"	Unsigned16	RW	Index 0x1800 0 Index 0x1801 – 1804 100
	4	Reserved	Unsigned8	RW	0
	5	"Event Timer" *)*	Unsigned16	RW	0

Structure of sub-index 1, COB ID:

Bit 31	Bit 30	Bit 29 Bit 11	Bit 10 Bit 0
0/1 Valid/invalid	0/1 RTR allowed/not allowed	Reserved (always 0)	COB ID

In order to enter a new COB ID, bit 31 must first be set to 1; changing a valid COB ID (bit 31 = 0) is prohibited per DS301.

* The "Event Timer" can only be used for transfer types 254/255.

One data transfer mode (transmission type in index "Communication Parameter") can be defined for each PDO. Digital and analog inputs are transferred as standard after the "Change of Value" (COV) procedure. The following table explains the type of transfer depending on the set "Transmission Type."

Table 117: "Transmission Type"

Trans-mission Type	**PDO Transmission**						
	Cyclic	Acyclic	Synchron ous	Asynchrono us	RTR only	TxPDO (inputs)	RxPDO (outputs)
0		X	X			COV transfer at each SYNC	After each SYNC, outputs are set as demanded by the last received PDO.
Trans-mission Type	**PDO Transmission**						
1 – 240	X		X			Transfer for each x-th SYNC (x = 1 through 240)	After each SYNC, outputs are set as demanded by the last received PDO.
241 – 251	Reserved						
252			X		X	Data is newly imported but not sent at each SYNC. Request via RTR	Not supported
253				X	X	Request via RTR	COV
254				X		COV*	COV
255				X		COV*	COV

The "Inhibit Time" specifies the minimum time interval between two consecutive PDOs with identical COB IDs. One time unit amounts to 100 µs. The transferred value is internally rounded down to the nearest millisecond.

In order to enter a new value, the COB ID must be set to invalid (bit 31 = 1); entering a new time in a valid COB ID (bit 31 = 0) is prohibited per DS301.

* The data is sent in the interval of the set "Inhibit Time"

Example:

If a PDO is to be send no more frequently than every 30 ms, set sub-index 3, "Inhibit Time," to 300 (0x12C).

The "Event Timer" specifies the time in which a PDO is sent, even if no change has been made to the PDO. The time should be entered in milliseconds, and the timer is started each time an event (change of PDO data) occurs.

Note

The "Event Timer" can only be used for transfer types 254/255.

If the time is shorter than the "Inhibit Time," another event is generated after it elapses.

Note

An object entry can only be entered in a maximum of three different PDOs.

19.1.24 Object 0x1A00 – 0x1A1F, Transmit PDO Mapping Parameter

This object is used to determine which data is sent via the PDO. Sub-index 0 contains the number of objects that are valid for the PDO.

Table 118: Object 0x1A00 – 0x1A1F

Index	Sub-Index	Name	Data Type	Attribute	Default Value
0x1A00 through 0x1A1F	0	Number of objects inserted	Unsigned8	RW	-
	1 through 8	1st object through 8th object	Unsigned32	RW	-

Sub-index 0 contains the number of valid sub-indices.

Objects 1 – 8 are structured as follows:

Bit 31	Bit 16	Bit 15	Bit 8	Bit 7	Bit 0
Index		Sub-Index		Object length	

Index: Index of object to be transferred
Sub index: Sub-index of object to be transferred
Object length: Size of object in bits
Because a maximum of 8 bytes can be transferred in one PDO, the sum of the valid object lengths shall not exceed 64 (8 bytes x 8 bits).

19.2 Manufacturer-Specific Profile Area, Object 0x2000 – 0x5FFF

The data from the I/O module is reproduced in the "Manufacturer-Specific Area" profile. The I/O module data corresponding to device profile DS 401 exists in indices 0x5300 - 0x5340 (input data) and 0x5400 - 0x5440 (output data), as well as indices 0x6000, 0x6200, 0x6401, 0x6411.

19.2.1 Object 0x37F1, Fieldbus Coupler CS LED Flash Signals

The last 16 flash signals of the CS LED are stored in this index. Sub-index 0 contains the number of currently stored entries. If no entry is contained, a 0 is displayed. A new error is stored in sub-index 1, and all other errors are shifted down by one sub-index entry. The last error is automatically removed from the list. Writing a 0 on sub-index 0 deletes the list.

The object is structured by the Unsigned32 data type as follows:

Byte 4	Byte 3	Byte 2	Byte 1
0x00	Error group	Error code	Error argument

Table 119: Object 0x37F0

Sub-Index	Name	Data Type	Attribute	Description
0	Maximum supported Sub-Index	Unsigned8	RW	Number of supported sub-indices
1	1st error entry	Unsigned32	RO	Last error that occurred
2	2nd error entry	Unsigned32	RO	Next to last error
…	…	…	…	…
16	16th error entry	Unsigned32	RO	Oldest error

19.2.2 Object 0x37F5, Fieldbus Coupler Diagnostics

The diagnostic value of the fieldbus coupler is stored in this index. Sub-index 0 contains the number of used sub-indices. This corresponds to the length (in bytes) of the diagnostic value. In the initialization phase, the fieldbus coupler identifies the length of the diagnostic value and enters it.

Beginning with sub-index 1, the diagnostic value is entered into the following sub-indices of the fieldbus coupler in a byte-by-byte manner. The low byte of the diagnostic value is in sub-index 1.

Table 120: Object 0x37F5

Sub-Index	Name	Data Type	Attribute	Description
0	Maximum supported Sub-Index	Unsigned8	RO	Number of supported sub-indices
1	1st diagnostic byte	Unsigned8	RO	1st diagnostic byte (low byte) of the fieldbus coupler
2	2nd diagnostic byte	Unsigned8	RO	2nd diagnostic byte of the fieldbus coupler
…	…	…	…	…
n	nth diagnostic byte	Unsigned8	RO	Last diagnostic byte (high byte) of the fieldbus coupler

19.2.3 Object 0x37F6, Fieldbus Coupler Diagnostic Acknowledgement

This object contains the diagnostic acknowledgement of the fieldbus coupler. Sub-index 0 contains the number of used sub-indices. This corresponds to the length (in bytes) of the diagnostic acknowledgement. In the initialization phase, the fieldbus coupler identifies the length of the diagnostic acknowledgement and enters it into sub-index 0.

Beginning with sub-index 1, enter the diagnostic acknowledgement into the following sub-indices in a byte-by-byte manner. The low byte of the diagnostic acknowledgment is in sub-index 1.

Table 121: Object 0x37F6

Sub-Index	Name	Data Type	Attribute	Description
0	Maximum supported Sub-Index	Unsigned8	RO	Number of supported sub-indices
1	1st diagnostic acknowledgement byte	Unsigned8	RO	1st diagnostic acknowledgement byte (low byte) of the fieldbus coupler
2	2nd diagnostic acknowledgement byte	Unsigned8	RO	2nd diagnostic acknowledgement byte of the fieldbus coupler
…	…	…	…	…
n	nth diagnostic acknowledgement byte	Unsigned8	RO	Last diagnostic acknowledgement byte (high byte) of the fieldbus coupler

19.2.4 Object 0x5000, Read Input Process Image

This object allows the entire input process image to be read via the SDO. Because access via the SDO is slow, time-critical data should only be transferred via the PDO.

Table 122: Object 0x5000

Index	Sub-Index	Name	Data Type	Attribute	Default Value	Description
0x5000	0	Number of relevant bytes in the input process image	Unsigned16	RO	-	Number of relevant bytes in the input process image
	1	1st input block	Octet_String	RO	-	Lower 255 bytes from the 512-byte large input process image
	2	2nd input block	Octet_String	RO	-	Upper 255 bytes from the 512-byte large input process image. Only available if more than 255 bytes of input data exist.

19.2.5 Object 0x5001, Write Output Process Image

Allows the SDO to be used to write the entire output process image as a domain. Because access via the SDO is slow, time-critical data should only be transferred via the PDO.

Table 123: Object 0x5001

Index	Sub-Index	Name	Data Type	Attribute	Default Value	Description
0x5001	0	Number of relevant bytes in the output process image	Unsigned16	RO	-	Number of relevant bytes in the output process image
	1	1st output block	Octet_String	RW	-	Lower 255 bytes from the 512-byte large output process image
	2	2nd output block	Octet_String	RW	-	Upper 255 bytes from the 512-byte large output process image. Only available if more than 255 bytes of output data exist.

19.2.6 Object 0x5200, Coupler Configuration Object

Table 124: Object 0x5200

Index	Sub-Index	Name	Data Type	Attribute	Default Value	Description
0x5200	0	Maximum supported sub-index	Unsigned8	RO	-	Maximum supported sub-index
	1	PDO processing	Unsigned16	RW	0	Specifies processing for the received PDO 0: The data from the last PDO is valid. 1: Consistent data output; i.e., as soon as a PDO arrives twice, an S-BUS cycle is carried out.
	2	Disable flashing indicator "Warning Level Reached" on the TX and RX LEDs	Unsigned8	RW	0	Enable/disable the "Warning Level" indicator" 0: "Warning Level" is displayed 1: Warning level indicator is suppressed
	3	Disable I/O module diagnostics comprehen-sively via emergency telegrams	Unsigned8	RW	0	Send diagnostics acc. to classification level 0: "Error" (I/O module): diagnostic telegrams (default) 1: "Error" and "Warning" (I/O module): diagnostic telegrams 2: "Error," "Warning" and "Notification" (I/O module): diagnostic telegrams 0xFF: Do not send diagnostic telegrams to I/O module
	4	Disable fieldbus coupler error messages comprehensivel y via emergency telegrams	Unsigned8	RW	0	0: Send diagnostic telegrams of fieldbus coupler 1: Do not send diagnostic telegrams of fieldbus coupler
	5	Reset flashing indicator of RX and RX LEDs.	Unsigned8	RW	0	By writing a 0 on this sub-index, the currently displayed flash signals of the RX, TX, and ERR LEDs are deleted. If the error recurs, it is flashed again.

19.2.7 0x5202 Module Configuration Object

The object contains the structure of the 767 node. A sub-index always refers to the fieldbus coupler or an I/O module. The description involves the hexadecimal value of the fieldbus coupler's item number without a slash ("/") or a hyphen ("-"). There are only as many sub-indices as there are I/O modules (including fieldbus coupler). A default setting does not exist.

Index	Sub-Index	Name	Data Type	Attribute	Description
0x5202	0	Maximum supported Sub-Index	Unsigned8	RO	Maximum supported sub-index. Number of I/O modules including fieldbus couplers.
	1	Module description	Visible String	RO	Sub-index 1 describes the fieldbus coupler: always "0767-1501"
	2	Module description	Visible String	RO	1st I/O module; e.g., "0767-3801"
	3	Module description	Visible String	RO	2nd I/O module
	…	…	…	…	…
	65	Module description	Visible String	RO	64th I/O module

19.2.8 Object 0x5300 – 0x5340, Input Data of I/O Modules

An index in the range of 0x5300 to 0x5340 is assigned to each input module. The input data from the first I/O module are in index 0x5301, from the second I/O module in index 0x5302, etc. Even if an I/O module contains no input data, the index still exists (sub-index 0/value 0). The input data from the digital inputs on the fieldbus coupler are in index 0x5300.

The input data (e.g., digital or analog) are composed of input process values and the diagnostic values of the I/O module. Beginning at sub-index 1, the input process values are sorted consecutively into the sub-indices in a byte-by-byte manner, followed by the diagnostic values. The input data is organized in such a way that the low byte is entered first, then the high byte.

Note

The data width of an I/O module can be found in the corresponding manual.

Table 125: Object 0x5300 – 0x5340

Sub-Index	Name	Data Type	Attribute	Description
0	Maximum supported Sub-Index	Unsigned8	R	Number of supported sub-indices
1	1st input byte	Unsigned8	R	1st input byte of I/O module
2	2nd input byte	Unsigned8	R	2nd input byte of I/O module
…	…	…	…	…
255	255th input byte	Unsigned8	R	255th input byte of I/O module

If an I/O module contains 2 bits of input data, they are located in sub-index 1, bits 0 and 1. The other bits are set to 0.

If an I/O module contains 10 bits of input data, the first 8 bits are located in sub-index 1 and the remaining 2 bits in sub-index 2, bits 0 and 1. The other bits are set to 0.

Example of Classifying Analog Input Data into Index 0x5301:

An analog I/O module (767-6401) with four inputs (each with 16 bits and 3 diagnostic bytes) occupies 11 sub-indices:

Sub-index 1 contains the low byte of the 1st input channel

Sub-index 2 contains the high byte of the 1st input channel

Sub-index 3 contains the low byte of the 2nd input channel

Sub-index 4 contains the high byte of the 2nd input channel

Sub-index 5 contains the low byte of the 3rd input channel

Sub-index 6 contains the high byte of the 3rd input channel

Sub-index 7 contains the low byte of the 4th input channel

Sub-index 8 contains the high byte of the 4th input channel

Sub-index 9 contains the first diagnostic byte

Sub-index 10 contains the second diagnostic byte

Sub-index 11 contains the third diagnostic byte

⚠ WARNING

Data Consistency!
When distributing process data to various PDOs, data inconsistency can occur. This data can lead to a malfunction of your system, and personnel can be injured. Data consistency is only guaranteed for data inside a PDO.

Example:

If the low byte of a 16-bit wide channel is transferred into PDO 1 and the high byte into PDO 2, the data consistency is not given.

19.2.9 Object 0x5400 – 0x5440, Output Data of the I/O Modules

An index in the range of 0x5400 to 0x5440 is assigned to each output module. The input data from the first I/O module are in index 0x5401, from the second I/O module in index 0x5402, etc. Even if an I/O module contains no input data, the index still exists; however, in this case sub-index 0 contains the value 0. The diagnostic acknowledgement of the digital inputs on the fieldbus coupler is in index 0x5400.

The output data (e.g., digital of analog) are composed of output process values and the diagnostic acknowledgement of the I/O module. Beginning at sub-index 1, the output process values are sorted consecutively into the sub-indices in a byte-by-byte manner, followed by the diagnostic acknowledgement. The output data is arranged in such a way that the low byte is entered first, then the high byte (Little Endian).

Note

The data width of an I/O module can be found in the corresponding manual.

Table 126: Object 0x5400 – 0x5440

Sub-Index	Name	Data Type	Attribute	Description
0	Maximum supported Sub-Index	Unsigned8	R	Number of supported sub-indices
1	1st output byte	Unsigned8	R	1st output byte of I/O module
2	2nd output byte	Unsigned8	R	2nd output byte of I/O module
…	…	…	…	…
255	255th output byte	Unsigned8	R	255th output byte of I/O module

If an I/O module contains 2 bits of output data, they are located in sub-index 1, bits 0 and 1. The other bits are set to 0.

If an I/O module contains 10 bits of output data, the first 8 bits are located in sub-index 1 and the remaining 2 bits in sub-index 2, bit positions 0 and 1. The other bits are set to 0.

Example of Classifying Analog Output Data into Index 0x5401:

An analog I/O module (767-7401) with four outputs (each with 16 bits and 1 diagnostic bytes) occupies 9 sub-indices:

Sub-index 1 contains the low byte of the 1st output channel

Sub-index 2 contains the high byte of the 1st output channel

Sub-index 3 contains the low byte of the 2nd output channel

Sub-index 4 contains the high byte of the 2nd output channel

Sub-index 5 contains the low byte of the 3rd output channel

Sub-index 6 contains the high byte of the 3rd output channel

Sub-index 7 contains the low byte of the 4th output channel

Sub-index 8 contains the high byte of the 4th output channel

Sub-index 9 contains the diagnostic acknowledgement (1 byte)

⚠ WARNING

Data Consistency!
When distributing process data to various PDOs, data inconsistency can occur. This data can lead to a malfunction of your system, and personnel can be injured. Data consistency is only guaranteed for data inside a PDO.

Example:

If the low byte of a 16-bit wide channel is entered into PDO 1 and the high byte into PDO 2, the data consistency is not given.

19.3 Standard Device Profile Area – DS 401, from Object 0x6000

The fieldbus coupler supports the standard device profile "Device Profile for Generic I/O Modules." The following table lists all the objects of the standard profile DS401 that are supported by the fieldbus coupler:

Table 127: Standard device profile

Index	Name	Data Type	Description	Information
0x6000	Read input 8 Bit	Array Unsigned 8	Data from digital input modules	For more information, see Section 19.3.1.
0x6005	Global Interrupt Enable Digital 8 Bit	Unsigned8	Global enabling of transmission of digital input blocks (8 bit)	For more information, see Section 19.3.2.
0x6006	Interrupt Mask Any Change 8 Bit	Array Unsigned 8	Enabling of transmission at each change of digital input blocks (8 bit)	For more information, see Section 19.3.3.
0x6007	Interrupt Mask Low-to-High 8 Bit	Array Unsigned 8	Enabling to send TxPDO if positive pulse edge arises on the digital input block (8 bit)	For more information, see Section 19.3.4.
0x6008	Interrupt Mask High-to-Low 8 Bit	Array Unsigned 8	Enabling to send TxPDO if negative pulse edge arises on the digital input block (8 bit)	For more information, see Section 19.3.5.
0x6200	Write Output 8 Bit	Array Unsigned 8	Data from digital output modules	For more information, see Section 19.3.6.
0x6206	Error Mode Output 8-Bit	Array Unsigned 8	Enabling of defined error values from digital output blocks (8 bit)	For more information, see Section 19.3.7.
0x6207	Error Value Output 8-Bit	Array Unsigned 8	Defined error values from digital output blocks (8 bit)	For more information, see Section 19.3.8.
0x6401	Read analog input 16-Bit	Array Unsigned 16	Data from analog input modules (16 bit)	For more information, see Section 19.3.9.
0x6411	Write Analog Output 16-Bit	Array Unsigned 16	Data from analog output modules (16 bit)	For more information, see Section 19.3.10.
0x6421	Analog Input Trigger Selection	Array Unsigned 8	Determine trigger frequency for analog input data (16 bit)	For more information, see Section 19.3.11.

Table 127: Standard device profile

Index	Name	Data Type	Description	Information
0x6423	Analog Input Global Interrupt Enable	Boolean	Global enabling of transmission of analog input data (16 bit)	For more information, see Section 19.3.12.
0x6424	Analog Input Interrupt Upper Limit Integer	Array Unsigned 16	Transmission of 16-bit input data when threshold value has been exceeded	For more information, see Section 19.3.13.
0x6425	Analog Input Interrupt Lower Limit Integer	Array Unsigned 16	Transmission of 16-bit input data when threshold value has been undershot	For more information, see Section 19.3.14.
0x6426	Analog Input Interrupt Delta Unsigned	Array Unsigned 16	Transmission when the 16-bit input data has changed by at least the delta value	For more information, see Section 19.3.15.
0x6427	Analog Input Interrupt Negative Delta Unsigned	Array Unsigned 16	Transmission when the 16-bit input data has decreased by at least the delta value	For more information, see Section 19.3.16.
0x6428	Analog Input Interrupt Positive Delta Unsigned	Array Unsigned 16	Transmission when the 16-bit input data has increased by at least the delta value	For more information, see Section 19.3.20.
0x6443	Analog Output Error Mode	Array Unsigned 8	Enabling for defined error values of 16-bit output data	For more information, see Section 19.3.18.
0x6444	Analog Output Error Value Integer	Array Unsigned 16	Value in the case of error in the 16-bit output data	For more information, see Section 19.3.19.
0x67FE	Error Behavior	Array Unsigned 8	State change in the case of error	For more information, see Section 19.3.20.

19.3.1 Object 0x6000, Digital Inputs

This object contains the process data of the digital input modules. Sub-index 1 contains the first eight digital input channels, beginning with the fieldbus coupler. Only process values are concerned. Diagnostic values are not considered.

Table 128: Object 0x6000

Index	Sub-Index	Name	Data Type	Attribute	Default Value
0x6000	0	Number of digital input blocks	Unsigned8	RO	-
	1	1st input block	Unsigned8	RO	-
	2	2nd input block	Unsigned8	RO	-
	…	…	…	…	…
	64	64th input block	Unsigned8	RO	-

19.3.2 Object 0x6005, Global Interrupt Enable Digital, 8 Bit

Use this object to control the transmission of digital input data with the PDO. If the value = 1, the transmission is generally enabled; it depends only on objects 0x6006 … 0x6008 and the PDO's transmission type. If the value = 0, no transmission of digital input data takes place, irrespective of objects 0x6006 … 0x6008.

Table 129: Object 0x6005

Index	Sub-Index	Name	Data Type	Attribute	Default Value
0x6005	0	Global Interrupt Enable Digital 8 Bit	Unsigned8	RW	1

19.3.3 Object 0x6006, Interrupt Mask Any Change, 8 Bit

This object is used to determine which digital input channel sends its data when a change occurs. This requires that the transmission (object 0x6005 = 1) is enabled.

Table 130: Object 0x6006

Index	Sub-Index	Name	Data Type	Attribute	Default Value
0x6006	0	Number of digital input blocks	Unsigned8	RO	-
	1	Mask of 1st input block	Unsigned8	RW	255
	2	Mask of 2nd input block	Unsigned8	RW	255
	....	....	....	....	....
	64	Mask of 64th input block	Unsigned8	RW	255

0 = transmission in the case of change disabled (per channel)
1 = transmission in the case of change enabled (per channel)

Example:

Sub-index 0 = 1, sub-index 1 contains value 65 (0x41); i.e.,
only channels 1 and 7 transmit data in the case of a change.

19.3.4 Object 0x6007, Interrupt Mask Low-to-High, 8 Bit

This object is used to determine which digital input channel sends its data when a positive pulse edge occurs. This requires that the transmission (object 0x6005 = 1) is enabled. This object has the OR link for object 0x6006.

Table 131: Object 0x6007

Index	Sub-Index	Name	Data Type	Attribute	Default Value
0x6007	0	Number of digital input blocks	Unsigned8	RO	-
	1	Mask of 1st input block	Unsigned8	RW	0
	2	Mask of 2nd input block	Unsigned8	RW	0
	....	....	....	....	....
	64	Mask of 64th input block	Unsigned8	RW	0

0 = transmission in the case of positive pulse edge disabled (per channel)
1 = transmission in the case of positive pulse edge enabled (per channel)

Example:

Index 0x6006, sub-index 0 = 1, sub-index 1 contains value 65 (0x41)
Index 0x6007, sub-index 0 = 1, sub-index 1 contains value 33 (0x21)
Channels 1 and 7 transmit data at each change
Channel 6 only transmits in the case of a positive pulse edge

19.3.5 Object 0x6008, Interrupt Mask High-to-Low, 8 Bit

This object is used to determine which digital input channel sends its data when a negative pulse edge occurs. This requires that the transmission (object 0x6005 = 1) is enabled. This object has the OR link for object 0x6006.

Table 132: Object 0x6008

Index	Sub-Index	Name	Data Type	Attribute	Default Value
0x6008	0	Number of digital input blocks	Unsigned8	RO	-
	1	Mask of 1st input block	Unsigned8	RW	0
	2	Mask of 2nd input block	Unsigned8	RW	0
	....	....	....	....	....
	64	Mask of 64th input block	Unsigned8	RW	0

0 = transmission in the case of negative pulse edge disabled (per channel)
1 = transmission in the case of negative pulse edge enabled (per channel)

Example:

Index 0x6006, sub-index 0 = 1, sub-index 1 contains value 65 (0x41)
Index 0x6008, sub-index 0 = 1, sub-index 1 contains value 33 (0x21)
Channels 1 and 7 transmit data at each change
Channel 6 only transmits in the case of a negative pulse edge

19.3.6 Object 0x6200, Digital Outputs

This object contains the process data of the digital output modules. Sub-index 1 contains the first eight digital output channels, beginning with the fieldbus coupler. Only process values are concerned. Diagnostic acknowledgements are not considered.

Table 133: Object 0x6200

Index	Sub-Index	Name	Data Type	Attribute	Default Value
0x6200	0	Number of digital output blocks	Unsigned8	RO	-
	1	1st output block	Unsigned8	RW	0
	2	2nd output block	Unsigned8	RW	0
	....	....	....	....	....
	64	64th output block	Unsigned8	RW	0

19.3.7 Object 0x6206, Error Mode Output, 8 Bit

This object is used to specify whether the outputs are to switch to a defined error mode (see object 0x6207) if an error should occur (e.g., fieldbus coupler switches to stopped state, "Node Guarding" has failed, etc.). If the error is corrected, the outputs remain in their states. The set error mode for the output channels remains in effect.

Table 134: Object 0x6206

Index	Sub-Index	Name	Data Type	Attribute	Default Value
0x6206	0	Number of digital output blocks	Unsigned8	RO	-
	1	Mask of 1st output block	Unsigned8	RW	255
	2	Mask of 2nd output block	Unsigned8	RW	255
	…	…	…	…	…
	64	Mask of 64th output block	Unsigned8	RW	255

0 = outputs remain unchanged (per channel)
1 = outputs are switched to a defined error state (per channel)

19.3.8 Object 0x6207, Error Value Output, 8 Bit

This object is used to define the values that the outputs are to adopt if an error should occur. This requires that the corresponding bit is set in object 0x6206.

Table 135: Object 0x6207

Index	Sub-Index	Name	Data Type	Attribute	Default Value
0x6207	0	Number of digital output blocks	Unsigned8	RO	-
	1	Mask of 1st output block	Unsigned8	RW	0
	2	Mask of 2nd output block	Unsigned8	RW	0
	…	…	…	…	…
	64	Mask of 64th output block	Unsigned8	RW	0

0 = output at 0 (per channel)
1 = output at 1 (per channel)

Example:

Index 0x6206, sub-index 0 = 1, sub-index 1 contains value 65 (0x41)
Index 0x6207, sub-index 0 = 1, sub-index 1 contains value 33 (0x21)
Channel 1 is set to 1, channel 7 is set to 0. All other output channels remain unchanged if an error occurs.

19.3.9 Object 0x6401, Analog Inputs, 16 Bit

This object contains the process data of the analog input modules. Sub-index 1 contains the first analog input channel, beginning with the fieldbus coupler.

Table 136: Object 0x6401

Index	Sub-Index	Name	Data Type	Attribute	Default Value
0x6401	0	Number of analog input channels (16 bit)	Unsigned8	RO	-
	1	1st channel	Unsigned16	RO	-
	…	…	…	…	…
	254	254th channel	Unsigned16	RO	-

19.3.10 Object 0x6411, Analog Outputs, 16 Bit

This object contains the process data of the analog output modules. Sub-index 1 contains the first analog output channel, beginning with the fieldbus coupler.

Table 137: Object 0x6411

Index	Sub-Index	Name	Data Type	Attribute	Default Value
0x6411	0	Number of analog output channels (16 bit)	Unsigned8	RO	-
	1	1st channel	Unsigned16	RW	0
	…	…	…	…	…
	254	254th channel	Unsigned16	RW	0

19.3.11 Object 0x6421, Analog Input Interrupt Trigger Selection

Use this object to create a PDO transmission requirement. The transmission requires a 1 to be entered in object 0x6423, thus enabling general transmission.

Table 138: Object 0x6421

Index	Sub-Index	Name	Data Type	Attribute	Default Value
0x6421	0	Number of analog input channels (16 bit)	Unsigned8	RO	-
	1	Trigger 1st channel	Unsigned8	RW	7
	…	…	…	…	…
	254	Trigger 254th channel	Unsigned8	RW	7

Structure of sub-indices 1 … 254:

Table 139: Object 0x6421

Bit	Transmission Requirement	Configuration Sub-Index
0	Threshold value exceeded	0x6424
1	Threshold value undershot	0x6425
2	Change in input value larger than delta value from last transmission	0x6426
3	Reduction in input value by more than delta value from last transmission	0x6427
4	Increase in input value by more than delta value from last transmission	0x6428
5 through 7	Reserved	-

19.3.12 Object 0x6423, Analog Input Global Interrupt Enable

Use this object to control the transmission of analog input data with the PDO. If the value = 1, transmission is generally enabled; it depends only upon object 0x6421 and the PDO transmission type. If the value = 0, no transmission of analog input data takes place, irrespective of object 0x6421.

Table 140: Object 0x6423

Index	Sub-Index	Name	Data Type	Attribute	Default Value
0x6423	0	Global Interrupt Enable Analog 16Bit	Unsigned8	RW	0

In the default configuration, transmission of analog input data is switched off.

19.3.13 Object 0x6424, Analog Input Interrupt Upper Limit Integer

Threshold value monitoring is possible with this object if configured in object 0x6423. If an input value is greater than or equal to the specified threshold value, monitoring takes place for as long as no additional trigger requirements (e.g., object 0x6426) are set.

Table 141: Object 0x6424

Index	Sub-Index	Name	Data Type	Attribute	Default Value
0x6424	0	Number of analog input channels (16 bit)	Unsigned8	RO	-
	1	Threshold value of 1st channel	Integer 32	RW	0
	…	…	…	…	…
	254	Threshold value of 254th channel	Integer 32	RW	0

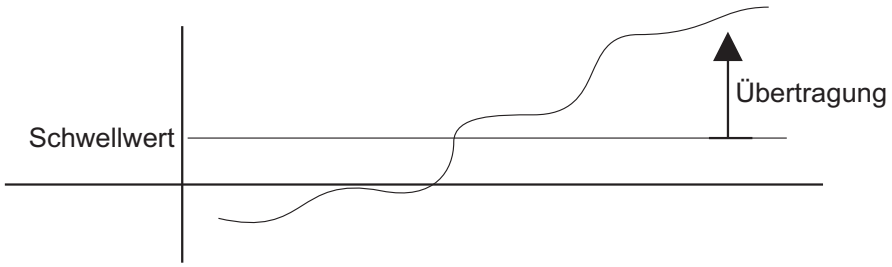


Figure 134: Upper threshold value

19.3.14 Object 0x6425, Analog Input Interrupt Lower Limit Integer

Threshold value monitoring is possible with this object if configured in object 0x6423. If an input value size is less than or equal to the specified threshold value, monitoring takes place for as long as no additional trigger requirements (e.g., object 0x6426) are set.

Table 142: Object 0x6425

Index	Sub-Index	Name	Data Type	Attribute	Default Value
0x6425	0	Number of analog input channels (16 bit)	Unsigned8	RO	-
	1	Threshold value of 1st channel	Integer 32	RW	0
	…	…	…	…	…
	254	Threshold value of 254th channel	Integer 32	RW	0

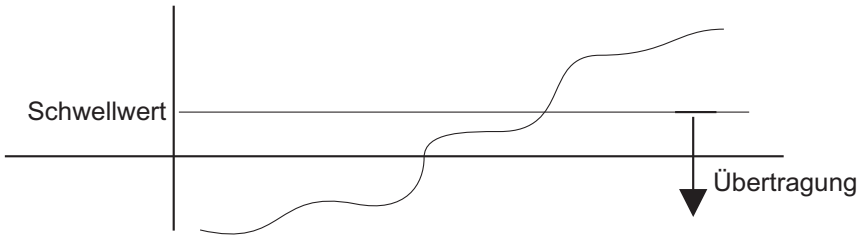


Figure 135: Lower threshold value

19.3.15 Object 0x6426, Analog Input Interrupt Delta Unsigned

By defining this object, the value that is to be retransmitted must be larger or smaller than the previously sent value by at least the delta value.
This object can, for example, be linked with object 0x6424 so that transmission only takes place after both the set threshold value and the delta function are achieved.

Table 143: Object 0x6426

Index	Sub-Index	Name	Data Type	Attribute	Default Value
0x6426	0	Number of analog input channels (16 bit)	Unsigned8	RO	-
	1	Delta value of 1st channel	Unsigned32	RW	0
	…	…	…	…	…
	254	Delta value of 254th channel	Unsigned32	RW	0

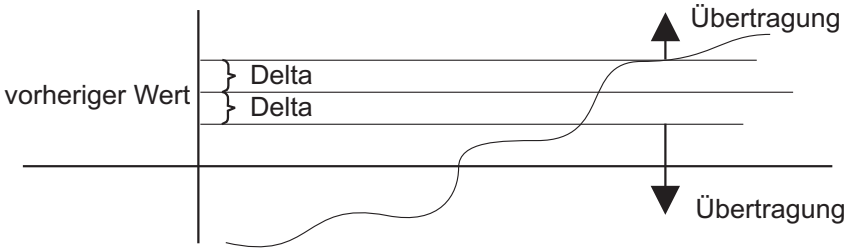


Figure 136: Delta value

19.3.16 Object 0x6427, Analog Input Interrupt Negative Delta Unsigned

By defining this object, the value that is to be retransmitted must be smaller than the previously sent value by at least the delta value.

Table 144: Object 0x6427

Index	Sub-Index	Name	Data Type	Attribute	Default Value
0x6427	0	Number of analog input channels (16 bit)	Unsigned8	RO	-
	1	Delta value of 1st channel	Unsigned32	RW	0
	…	…	…	…	…
	254	Delta value of 254th channel	Unsigned32	RW	0

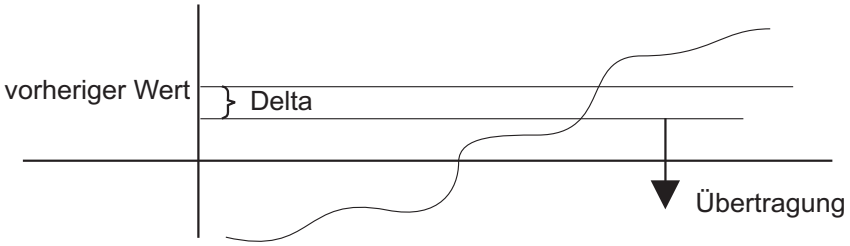


Figure 137: Lower delta value

19.3.17 Object 0x6428, Analog Input Interrupt Positive Delta Unsigned

By defining this object, the value that is to be retransmitted must be larger than the previously sent value by at least the delta value.

Table 145: Object 0x6428

Index	Sub-Index	Name	Data Type	Attribute	Default Value
0x6428	0	Number of analog input channels (16 bit)	Unsigned8	RO	-
	1	Delta value of 1st channel	Unsigned32	RW	0
	…	…	…	…	…
	254	Delta value of 254th channel	Unsigned32	RW	0

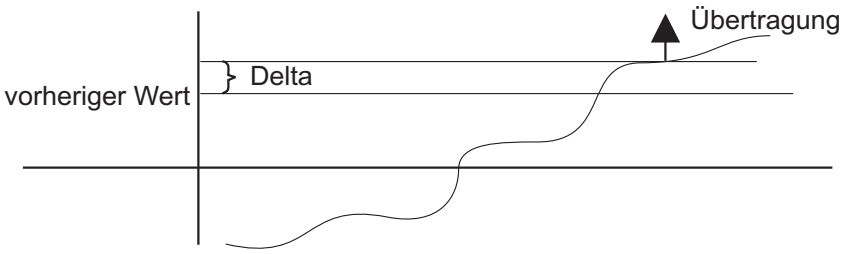


Figure 138: Upper delta value

19.3.18 Object 0x6443, Analog Output Error Mode

This object is used to specify whether the outputs are to switch to a defined error mode (see object 0x6444) if an error should occur (e.g., fieldbus coupler switches to stopped state, "Node Guarding" has failed, etc.). If the error is corrected, the outputs remain in their states. The set error mode for the output channels remains in effect.

Analog outputs not present with object 0x6444 are always set to 0 in the case of an error.

Table 146: Object 0x6443

Index	Sub-Index	Name	Data Type	Attribute	Default Value
0x6443	0	Number of analog output channels (16 bit)	Unsigned8	RO	-
	1	Error mode of 1st channel	Unsigned8	RW	1
	…	…	…	…	…
	254	Error mode of 254th channel	Unsigned8	RW	1

0 = output remains unchanged
1 = output switches to a defined error mode

19.3.19 Object 0x6444, Analog Output Error Value Integer

This object is used to define the values that the outputs are to adopt if an error should occur. This requires that the corresponding bit is set in object 0x6443.

Table 147: Object 0x6444

Index	Sub-Index	Name	Data Type	Attribute	Default Value
0x6444	0	Number of analog output channels (16 bit)	Unsigned8	RO	-
	1	Error value of 1st channel	Integer 32	RW	0
	…	…	…	…	…
	254	Error value of 254th channel	Integer 32	RW	0

19.3.20 Object 0x67FE, Error Behavior

This object is an identical copy of the "Error Behavior" object 0x1029. The functionality of both objects is the same. Its purpose consists of maintaining compatibility.

Table 148: Object 0x67FE

Index	Sub-Index	Name	Data Type	Attribute	Default Value
0x67FE	0	Maximum supported sub-index	Unsigned8	RO	1
	1	Communication error	Unsigned8	RW	0

19.3.21 CANopen Fieldbus Variables

In the index range of 0xA000 – 0xA8C0, the fieldbus variables of the control program (IEC 61131-3) are entered in the fieldbus coupler.

NOTICE

The fieldbus input variables are defined from the standpoint of CANopen; from the standpoint of the control program, these device objects are classified as the "output" type. Accordingly, the fieldbus output variables are classified as the "input" type for the control program.

The following table specifies the CANopen objects that are supported by the fieldbus coupler and that can be used to access the control program, with the help of the device objects.

Table 149: Objects of CANopen fieldbus variables

Index	Name	Data Type	Description	Information
0xA000	Integer8 input variables	Array Integer 8	Data from 8-bit Integer input variables	Please see Section 19.3.22
0xA040	Unsigned8 input variables	Array Unsigned 8	Data from 8-bit Unsigned input variables	Please see Section 19.3.23
0xA080	Boolean input variables	Boolean	Data from Boolean input variables	Please see Section 19.3.24
0xA0C0	Integer16 input variables	Array Integer 16	Data from 16-bit Integer input variables	Please see Section 19.3.25
0xA100	Unsigned16 input variables	Array Unsigned 16	Data from 16-bit Unsigned input variables	Please see Section 19.3.25

Table 149: Objects of CANopen fieldbus variables

Index	Name	Data Type	Description	Information
0xA140	Integer24 input variables	Array Integer 24	Data from 24-bit Integer input variables	
0xA180	Unsigned24 input variables	Array Unsigned 24	Data from 24-bit Unsigned input variables	
0xA1C0	Integer32 input variables	Array Integer 32	Data from 32-bit Integer input variables	
0xA200	Unsigned32 input variables	Array Unsigned 32	Data from 32-bit Unsigned input variables	
0xA240	Float32 input variables	Array Floating Point 32	Data from 32-bit Floating Point input variables	
0xA280	Unsigned40 input variables	Array Unsigned 40	Data from 40-bit Unsigned input variables	
0xA2C0	Integer40 input variables	Array Integer 40	Data from 40-bit Integer input variables	
0xA300	Unsigned48 input variables	Array Unsigned 48	Data from 48-bit Unsigned input variables	
0xA340	Integer48 input variables	Array Integer 48	Data from 48-bit Integer input variables	
0xA380	Unsigned56 input variables	Array Unsigned 56	Data from 56-bit Unsigned input variables	Please see Section 19.3.26
0xA3C0	Integer56 input variables	Array Integer 56	Data from 56-bit Integer input variables	
0xA400	Integer64 input variables	Array Integer 64	Data from 64-bit Integer input variables	
0xA440	Unsigned64 input variables	Array Unsigned 64	Data from 64-bit Unsigned input variables	
0xA480	Integer8 output variables	Array Integer 8	Data from 8-bit Integer output variables	
0xA4C0	Unsigned8 output variables	Array Unsigned 8	Data from 8-bit Unsigned output variables	
0xA500	Boolean output variables	Boolean	Data from Boolean output variables	
0xA540	Integer16 output variables	Array Integer 16	Data from 16-bit Integer output variables	
0xA580	Unsigned16 output variables	Array Unsigned 16	Data from 16-bit Unsigned output variables	
0xA5C0	Integer24 output variables	Array Integer 24	Data from 24-bit Integer output variables	

Table 150: Objects of CANopen fieldbus variables

Index	Name	Data Type	Description	Information
0xA600	Unsigned24 output variables	Array Unsigned 24	Data from 24-bit Unsigned output variables	Please see Section 19.3.26
0xA640	Integer32 output variables	Array Integer 32	Data from 32-bit Integer output variables	
0xA680	Unsigned32 output variables	Array Unsigned 32	Data from 32-bit Unsigned output variables	
0xA6C0	Float32 output variables	Array Floating Point 32	Data from 32-bit Floating Point output variables	
0xA700	Unsigned40 output variables	Array Unsigned 40	Data from 40-bit Unsigned output variables	
0xA740	Integer40 output variables	Array Integer 40	Data from 40-bit Integer output variables	
0xA780	Unsigned48 output variables	Array Unsigned 48	Data from 48-bit Unsigned output variables	
0xA7C0	Integer48 output variables	Array Integer 48	Data from 48-bit Integer output variables	
0xA800	Unsigned56 output variables	Array Unsigned 56	Data from 56-bit Unsigned output variables	
0xA840	Integer56 output variables	Array Integer 56	Data from 56-bit Integer output variables	
0xA880	Integer64 output variables	Array Integer 64	Data from 64-bit Integer output variables	
0xA8C0	Unsigned64 output variables	Array Unsigned 64	Data from 64-bit Unsigned output variables	

19.3.22 Object 0xA000, Integer8 IEC-61131-1 Input Variables

This object contains the process data of the Integer8 input variables. Sub-index 1 contains the first 8 bits of input data, sub-index 2 the next, etc.

Table 151: Object 0x67FE

Index	Sub-Index	Name	Data Type	Attribute	Default value
0xA000	0	Number of input data blocks	Unsigned8 R	O	-
	1	1st input data block	Integer 8	RO	-
	2	2nd input data block	Integer 8	RO	-
...	....		....		....
0xA002	2	512th input data block	Integer 8	RO	-

19.3.23 Object 0xA040, Unsigned8 IEC-61131-1 Input Variables

This object contains the process data of the Unsigned8 input variables. Sub-index 1 contains the first 8 bits of input data, sub-index 2 the next, etc.

Table 152: Object 0x67FE

Index	Sub-Index	Name	Data Type	Attribute	Default value
0xA040	0	Number of input data blocks	Unsigned8 R	O	-
	1	1st input data block	Unsigned8	RO	-
	2	2nd input data block	Unsigned8	RO	-
...	....		....		....
0xA042	2	512th input data block	Unsigned8	RO	-

19.3.24 Object 0xA080, Boolean IEC-61131-1 Input Variables

This object contains the process data of the Boolean input variables. Sub-index 1 contains the first 8 bits of input data, which are combined into one byte. Sub-index 2 contains the next, etc.

Table 153: Object 0x67FE

Index	Sub-Index	Name	Data Type	Attribute	Default value
0xA080	0	Number of input data blocks	Unsigned8 R	O	-
	1	1st input data block	Boolean	RO	-
	2	2nd input data block	Boolean	RO	-
...	....		....		....
0xA082	2	512th input data block	Boolean	RO	-

19.3.25 Object 0xA0C0, Integer16 IEC-61131-1 Input Variables

This object contains the process data of the Integer16 input variables. Sub-index 1 contains the first 16 bits of input data, sub-index 2 the next, etc.

Table 154: Object 0x67FE

Index	Sub-Index	Name	Data Type	Attribute	Default value
0xA0C0	0	Number of input data blocks	Unsigned8 R	O	-
	1	1st input data block	Integer 16	RO	-
	2	2nd input data block	Integer 16	RO	-
... .		....	....	....	....
0xA0C1	1	256th input data block	Integer16	RO	-

19.3.26 Object 0xA101 through 0xA8C0 Input and Output Variables per IEC 61131-1

All of the following objects are structured analogously to the ones previously described.
Each index has a maximum of 256 sub-indices (sub-index 0-255). The number of data entries is given in sub-index 0, and the data is stored in blocks in the subsequent sub-indices.
The size of the blocks depends on the data width of the associated data type.

Sub-Index	Table of Contents
0	Number of data blocks
1	First data block with data width of corresponding data type
2	Second data block with data width of corresponding data type
... ...	

The maximum indices and sub-indices are a result of the memory size of the fieldbus controller with 512 bytes, as well as the respective data width of the data types.

19.4 Available Libraries

After installing WAGO CoDeSys, the following libraries are available to you for the fieldbus coupler:

Table 155: Libraries

Name	Description
Config	Access to firmware version, version of FW components, WAGO_Object_Lib
Errors	-
CANopen	Synchronous access on extended WAGO CANopen functionalities
CANopenAsync	Asynchronous access on extended WAGO CANopen functionalities
CiA405	Asynchronous access on CiA405 CANopen functionality

19.5 Accessories

Here you will find a list of the most important accessory components that you need to begin operating the fieldbus coupler. The mounting rails, spacers, etc., as well as the entire accessories program from the AUTOMATION field, can be found in the Internet as usual or in the "Components for Automation" catalog.

19.5.1 S-BUS Cable, Assembled on Both Ends

Table 156: S-BUS cable, assembled on both ends, B coded

Item	Length	Item number	PU
M12 socket, M12 plug, straight	0.2m	756-1305/0060-0002	1
M12 socket, M12 plug, straight	0.3m	756-1305/0060-0003	
M12 socket, M12 plug, straight	0.5m	756-1305/0060-0005	
M12 socket, M12 plug, straight	1.0m	756-1305/0060-0010	
M12 socket, M12 plug, straight	2.0m	756-1305/0060-0020	
M12 socket, M12 plug, straight	5.0m	756-1305/0060-0050	
M12 socket, M12 plug, straight	10.0m	756-1305/0060-0100	
M12 socket, M12 plug, straight	20.0m	756-1305/0060-0200	
M12 socket, M12 plug, straight	50.0m	756-1305/0060-0500	
M12 socket, M12 plug, angular	0.2m	756-1306/0060-0002	
M12 socket, M12 plug, angular	0.3m	756-1306/0060-0003	
M12 socket, M12 plug, angular	0.5m	756-1306/0060-0005	
M12 socket, M12 plug, angular	1.0m	756-1306/0060-0010	
M12 socket, M12 plug, angular	2.0m	756-1306/0060-0020	
M12 socket, M12 plug, angular	5.0m	756-1306/0060-0050	
M12 socket, M12 plug, angular	10.0m	756-1306/0060-0100	
M12 socket, M12 plug, angular	20.0m	756-1306/0060-0200	
M12 socket, M12 plug, angular	50.0m	756-1306/0060-0500	

19.5.2 S-BUS Terminator and USB Cable

Table 157: S-BUS terminator (B coded) and USB cable

Column heading	Item number	PU
S-BUS terminator M12 plug, straight	756-9409/0060-0000	1
USB communication cable	756-4101/0042-0030	

19.5.3 Supply Cable, Assembled on Both Ends

Table 158: Supply cable, assembled on both ends, A coded

Item	Length	Item number	PU
M12 socket, M12 plug, straight	0.2m	756-3105/0040-0002	1
M12 socket, M12 plug, straight	0.3m	756-3105/0040-0003	
M12 socket, M12 plug, straight	0.5m	756-3105/0040-0005	
M12 socket, M12 plug, straight	1.0m	756-3105/0040-0010	
M12 socket, M12 plug, straight	2.0m	756-3105/0040-0020	
M12 socket, M12 plug, straight	5.0m	756-3105/0040-0050	
M12 socket, M12 plug, straight	10.0m	756-3105/0040-0100	
M12 socket, M12 plug, straight	20.0m	756-3105/0040-0200	
M12 socket, M12 plug, angular	0.2m	756-3106/0040-0002	
M12 socket, M12 plug, angular	0.3m	756-3106/0040-0003	
M12 socket, M12 plug, angular	0.5m	756-3106/0040-0005	
M12 socket, M12 plug, angular	1.0m	756-3106/0040-0010	
M12 socket, M12 plug, angular	2.0m	756-3106/0040-0020	
M12 socket, M12 plug, angular	5.0m	756-3106/0040-0050	
M12 socket, M12 plug, angular	10.0m	756-3106/0040-0100	
M12 socket, M12 plug, angular	20.0m	756-3106/0040-0200	

19.5.4 CAN Cable Assembled on One End

Table 159: CAN cable, assembled on one end, A coded

Item	Length	Item number	PU
M12 socket, straight, open-ended	2m	756-1401/0060-0020	1
M12 socket, straight, open-ended	5m	756-1401/0060-0050	
M12 socket, straight, open-ended	10m	756-1401/0060-0100	
M12 socket, straight, open-ended	20m	756-1401/0060-0200	
M12 socket, angular, open-ended	2m	756-1402/0060-0020	
M12 socket, angular, open-ended	5m	756-1402/0060-0050	
M12 socket, angular, open-ended	10m	756-1402/0060-0100	
M12 socket, angular, open-ended	20m	756-1402/0060-0200	
M12 plug, straight, open-ended	2m	756-1403/0060-0020	
M12 plug, straight, open-ended	5m	756-1403/0060-0050	
M12 plug, straight, open-ended	10m	756-1403/0060-0100	
M12 plug, straight, open-ended	20m	756-1403/0060-0200	
M12 plug, angular, open-ended	2m	756-1404/0060-0020	
M12 plug, angular, open-ended	5m	756-1404/0060-0050	
M12 plug, angular, open-ended	10m	756-1404/0060-0100	
M12 plug, angular, open-ended	20m	756-1404/0060-0200	

19.5.5 CAN Cable Assembled on Both Ends

Table 160: CAN cable, assembled on both ends, A coded

Item	Length	Item number	PU
M12 socket, straight M12 plug, straight	2m	756-1405/0060-0020	1
M12 socket, straight M12 plug, straight	5m	756-1405/0060-0050	
M12 socket, straight M12 plug, straight	10m	756-1405/0060-0100	
M12 socket, straight M12 plug, straight	20m	756-1405/0060-0200	
M12 socket, angular M12 plug, angular	2m	756-1406/0060-0020	
M12 socket, angular M12 plug, angular	5m	756-1406/0060-0050	
M12 socket, angular M12 plug, angular	10m	756-1406/0060-0100	
M12 socket, angular M12 plug, angular	20m	756-1406/0060-0200	

19.5.6 CAN Accessories

Table 161: CANopen accessories

Column heading	Item number	PU
M12 plug, straight, can be preassembled Spring clamp technology	756-9207/0060-0000	1
M12 plug, angular, can be preassembled Spring clamp technology	756-9211/0060-0000	
M12 socket, straight, can be preassembled Spring clamp technology	756-9208/0060-0000	
M12 plug, angular, can be preassembled Spring clamp technology	756-9210/0060-0000	
M12 terminating plug, straight	756-9209/0060-0000	

19.5.7 Carrier Rail Adapters and Profile Adapters

Table 162: Carrier rail adapters and profile adapters

Column heading	Item number	PU
Carrier rail adapter	On request	1
Profile adapter	On request	

19.5.8 Protective Caps

Table 163: Protective caps for unassigned connections sockets

Column heading	Item number	PU
M8 protective caps for connection sockets	756-8101	1
M12 protective caps for connection socket	756-8102	
M12 protective caps for connection plugs	755-809	
M23 protective caps for connection plugs	755-837	

List of Figures

List of Tables

WAGO Kontakttechnik GmbH & Co. KG
Postfach 2880 D-32385 Minden
Hansastraße 27 D-32423 Minden
Phone: 05 71/8 87 – 0
Fax: 05 71/8 87 – 1 69
E-mail: info@wago.com

Web: http://www.wago.com