Manual

WAGO-I/O-SYSTEM 750

PFC200 CS 2ETH CAN

750-8203(/xxx-xxx)

PLC - PFC200 Controller

Version 1.1.0, valid from SW-Version 02.02.12(03)

WAGO Kontakttechnik GmbH & Co. KG

Hansastraße 27
D-32423 Minden

Phone: +49 (0) 571/8 87 – 0
Fax: +49 (0) 571/8 87 – 1 69

E-Mail: info@wago.com

Web: http://www.wago.com

Technical Support

Phone: +49 (0) 571/8 87 – 5 55
Fax: +49 (0) 571/8 87 – 85 55

E-Mail: support@wago.com

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.

E-Mail: documentation@wago.com

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

Table of Contents

1 Notes about this Documentation

Note

Always retain this documentation!
This documentation is part of the product. Therefore, retain the documentation during the entire service life of the product. Pass on the documentation to any subsequent user. In addition, ensure that any supplement to this documentation is included, if necessary.

1.1 Validity of this Documentation

This documentation is only applicable to the controller “PFC200 CS 2ETH CAN” (750-8203) and the variants listed in the table below.

Table 1: Variants

Item Number/Variant	Designation
750-8203	PFC200 CS 2ETH CAN
750-8203/025-000	PFC200 CS 2ETH CAN/T

Note

Documentation Validity for Variants
Unless otherwise indicated, the information given in this documentation applies to listed variants.

This documentation is only applicable from SW-Version 02.02.12(03).

1.2 Copyright

This Manual, including all figures and illustrations, is copyright-protected. Any further use of this Manual by third parties that violate pertinent copyright provisions is prohibited. Reproduction, translation, electronic and phototechnical filing/archiving (e.g., photocopying) as well as any amendments require the written consent of WAGO Kontakttechnik GmbH & Co. KG, Minden, Germany. Non-observance will involve the right to assert damage claims.

1.3 Symbols

⚠ DANGER

Personal Injury!
Indicates a high-risk, imminently hazardous situation which, if not avoided, will result in death or serious injury.

⚠ DANGER

Personal Injury Caused by Electric Current!
Indicates a high-risk, imminently hazardous situation which, if not avoided, will result in death or serious injury.

⚠ WARNING

Personal Injury!
Indicates a moderate-risk, potentially hazardous situation which, if not avoided, could result in death or serious injury.

⚠ CAUTION

Personal Injury!
Indicates a low-risk, potentially hazardous situation which, if not avoided, may result in minor or moderate injury.

NOTICE

Damage to Property!
Indicates a potentially hazardous situation which, if not avoided, may result in damage to property.

NOTICE

Damage to Property Caused by Electrostatic Discharge (ESD)!
Indicates a potentially hazardous situation which, if not avoided, may result in damage to property.

Note

Important Note!
Indicates a potential malfunction which, if not avoided, however, will not result in damage to property.

Information

Additional Information:
Refers to additional information which is not an integral part of this documentation (e.g., the Internet).

1.4 Number Notation

Table 2: Number Notation

Number code	Example	Note
Decimal	100	Normal notation
Hexadecimal	0x64	C notation
Binary	'100' '0110.0100'	In quotation marks, nibble separated with dots (.)

1.5 Font Conventions

Table 3: Font Conventions

Font type	Indicates
italic	Names of paths and data files are marked in italic-type. e.g.: *C:\Programme\WAGO-I/O-CHECK*
Menu	Menu items are marked in bold letters. e.g.: **Save**
>	A greater-than sign between two names means the selection of a menu item from a menu. e.g.: **File** > **New**
Input	Designation of input or optional fields are marked in bold letters, e.g.: **Start of measurement range**
“Value”	Input or selective values are marked in inverted commas. e.g.: Enter the value “4 mA” under **Start of measurement range**.
[Button]	Pushbuttons in dialog boxes are marked with bold letters in square brackets. e.g.: **[Input]**
[Key]	Keys are marked with bold letters in square brackets. e.g.: **[F5]**

2 Important Notes

This section includes an overall summary of the most important safety requirements and notes that are mentioned in each individual section. To protect your health and prevent damage to devices as well, it is imperative to read and carefully follow the safety guidelines.

2.1 Legal Bases

2.1.1 Subject to Changes

WAGO Kontakttechnik GmbH & Co. KG reserves the right to provide for any alterations or modifications that serve to increase the efficiency of technical progress. WAGO Kontakttechnik GmbH & Co. KG owns all rights arising from the granting of patents or from the legal protection of utility patents. Third-party products are always mentioned without any reference to patent rights. Thus, the existence of such rights cannot be excluded.

2.1.2 Personnel Qualifications

All sequences implemented on WAGO-I/O-SYSTEM 750 devices may only be carried out by electrical specialists with sufficient knowledge in automation. The specialists must be familiar with the current norms and guidelines for the devices and automated environments.

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

2.1.3 Use of the WAGO-I/O-SYSTEM 750 in Compliance with Underlying Provisions

Fieldbus couplers, fieldbus controllers and I/O modules found in the modular WAGO-I/O-SYSTEM 750 receive digital and analog signals from sensors and transmit them to actuators or higher-level control systems. Using programmable controllers, the signals can also be (pre-) processed.

The devices have been developed for use in an environment that meets the IP20 protection class criteria. Protection against finger injury and solid impurities up to 12.5 mm diameter is assured; protection against water damage is not ensured. Unless otherwise specified, operation of the devices in wet and dusty environments is prohibited.

Operating the WAGO-I/O-SYSTEM 750 devices in home applications without further measures is only permitted if they meet the emission limits (emissions of interference) according to EN 61000-6-3. You will find the relevant information in the section “Device Description” > “Standards and Guidelines” in the manual for the used fieldbus coupler/controller.

Appropriate housing (per 94/9/EG) is required when operating the WAGO-I/O-SYSTEM 750 in hazardous environments. Please note that a prototype test certificate must be obtained that confirms the correct installation of the system in a housing or switch cabinet.

2.1.4 Technical Condition of Specified Devices

The devices to be supplied ex works are equipped with hardware and software configurations, which meet the individual application requirements. WAGO Kontakttechnik GmbH & Co. KG will be exempted from any liability in case of changes in hardware or software as well as to non-compliant usage of devices.

Please send your request for modified and new hardware or software configurations directly to WAGO Kontakttechnik GmbH & Co. KG.

2.2 Safety Advice (Precautions)

For installing and operating purposes of the relevant device to your system the following safety precautions shall be observed:

⚠ DANGER

Do not work on devices while energized!
All power sources to the device shall be switched off prior to performing any installation, repair or maintenance work.

⚠ DANGER

Install the device only in appropriate housings, cabinets or in electrical operation rooms!
The WAGO-I/O-SYSTEM 750 and its components are an open system. As such, install the system and its components exclusively in appropriate housings, cabinets or in electrical operation rooms. Allow access to such equipment and fixtures to authorized, qualified staff only by means of specific keys or tools.

NOTICE

Do not use in telecommunication circuits!
Only use devices equipped with ETHERNET or RJ-45 connectors in LANs. Never connect these devices with telecommunication networks.

NOTICE

Replace defective or damaged devices!
Replace defective or damaged device/module (e.g., in the event of deformed contacts), since the long-term functionality of device/module involved can no longer be ensured.

NOTICE

Protect the components against materials having seeping and insulating properties!
The components are not resistant to materials having seeping and insulating properties such as: aerosols, silicones and triglycerides (found in some hand creams). If you cannot exclude that such materials will appear in the component environment, then install the components in an enclosure being resistant to the above-mentioned materials. Clean tools and materials are imperative for handling devices/modules.

NOTICE

Clean only with permitted materials!
Clean soiled contacts using oil-free compressed air or with ethyl alcohol and leather cloths.

NOTICE

Do not use any contact spray!
Do not use any contact spray. The spray may impair contact area functionality in connection with contamination.

NOTICE

Do not reverse the polarity of connection lines!
Avoid reverse polarity of data and power supply lines, as this may damage the devices involved.

NOTICE

Avoid electrostatic discharge!
The devices are equipped with electronic components that may be destroyed by electrostatic discharge when touched. Please observe the safety precautions against electrostatic discharge per DIN EN 61340-5-1/-3. When handling the devices, please ensure that environmental factors (personnel, work space and packaging) are properly grounded.

2.3 Special Use Conditions for ETHERNET Devices

If not otherwise specified, ETHERNET devices are intended for use on local networks. Please note the following when using ETHERNET devices in your system:

- Do not connect control components and control networks to an open network such as the Internet or an office network. WAGO recommends putting control components and control networks behind a firewall.
- Limit physical and electronic access to all automation components to authorized personnel only.
- Change the default passwords before first use! This will reduce the risk of unauthorized access to your system.
- Regularly change the passwords used! This will reduce the risk of unauthorized access to your system.
- If remote access to control components and control networks is required, use a Virtual Private Network (VPN).
- Regularly perform threat analyses. You can check whether the measures taken meet your security requirements.
- Use “defense-in-depth” mechanisms in your system's security configuration to restrict the access to and control of individual products and networks.

3 Device Description

The controller 750-8203(PFC200 CS 2ETH CAN) is an automation device that can perform control tasks of a PLC. It is suitable for mounting on a DIN rail and stands out on account of its various interfaces.

This controller can be used for applications in mechanical and systems engineering, in the processing industry and in building technology.

You can connect all available I/O modules of the WAGO-I/O-SYSTEM 750 (750 and 753 Series) to the controller, enabling it to internally process analog and digital signals from the automation environment, or to supply these signals to other devices via one of the available interfaces.

Automation tasks can be executed in all IEC 61131-3-compatible languages with the programming system CODESYS 2.3 (WAGO-I/O-PRO).
The implementation of the CODESYS task processing is optimized with real-time extensions in order to provide maximal performance for automation tasks. For visualization, Web visualization is also available in addition to the development environment.

The controller provides a physical 256 Mbyte program memory (flash), a 256 Mbyte data memory (RAM) and a 128 kbyte remanent memory (retain, NVRAM). The memory capacities may not be able to be utilized fully on account of internal administration.

The file system on the internal memory provides 64 Mbyte for applications. Files may also be stored on a removable memory card, or on an internal RAM disk.

The controller provides a 16 Mbyte program memory, a 64 Mbyte data memory and a 128 kbyte remanent memory (retain and flag variables) in an integrated NVRAM for IEC-61131-3 programming on CODESYS applications.

Two ETHERNET interfaces and an integrated, configurable switch enable line topology wiring for:

- In line topology with a common MAC address and IP address for both interfaces.
- Two separate networks with a common MAC address and an IP address for each interface.

Both of these interfaces support:

- 10Base-T / 100Base-TX
- Full/Half duplex
- Autonegotiation
- Auto-MDI(X)

The following fieldbus circuits are implemented for exchange of process data:

- MODBUS TCP
- MODBUS UDP
- CANopen Master/Slave

In the controller, all input signals from the sensors are combined. After connecting the controller, all of the I/O modules on the bus node are detected and a local process image is created from these. Analog and specialty module data is sent via words and/or bytes; digital data is sent bit by bit.

Note

No direct access from fieldbus to the process image for I/O modules!
Any data that is required from the I/O module process image must be explicitly mapped in the CODESYS program to the data in the fieldbus process image and vice versa! Direct access is not possible!

Fieldbus configuration can be performed using the CODESYS 2.3 controller configuration.

A Web-based management system (WBM) is also available as a configuration aid. This system includes various dynamic HTML pages from which, among other things, information about configuration and the status of the controller can be called up. The WBM is already stored in the device and is presented and operated using an Internet browser. You can also save your own HTML pages in the implemented file system, or call up programs directly.

In the controller's initial state, the installed firmware is based on Linux®, with special real-time extensions of the RT-Preempt patch. In addition, the following application programs are also installed on the controller, along with a number of different auxiliary programs:

- a SNMP server/client
- a Telnet server
- a FTP, FTPS server
- a SSH server/client
- a Web server
- a NTP client
- a BootP and DHCP daemon
- a CODESYS Runtime Environment

Based on IEC-61131-3 programming, data processing takes place on site in the controller. The logical process results can be output directly to the actuators or transmitted via a connected fieldbus to the higher level controller.

➔ **Note**

Memory card is not included in the scope of delivery!
Note, the controller is delivered without memory card.
To use a memory card, you must order one separately. The controller can also be operated without memory card expansion, the use of a memory card is optional.

➔ **Note**

Only use recommended memory cards!
Use only the SD memory card available from WAGO (item No. 758-879/000-001) as it is suitable for industrial applications subjected to environmental extremes and was developed for use in the controller.
Compatibility with other commercially available storage media cannot be guaranteed.

3.1 View

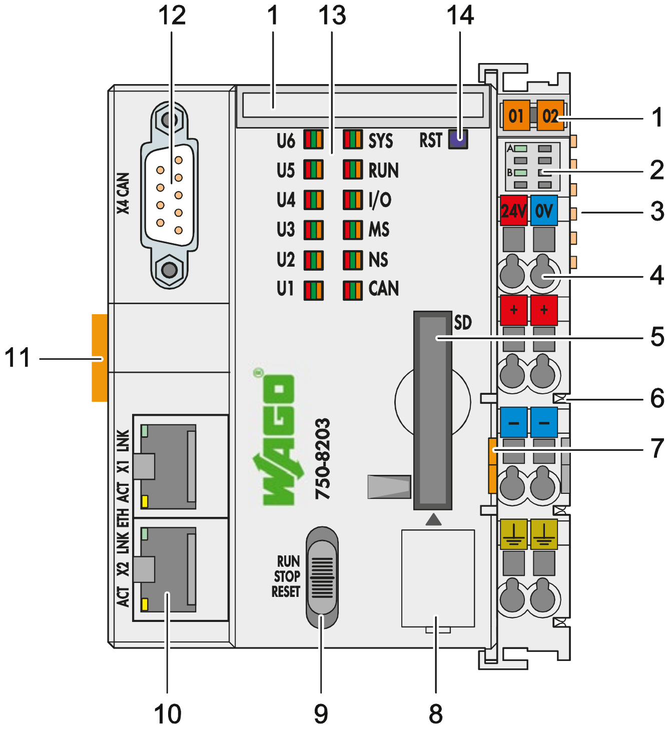


Figure 1: View of device

Table 4:Legend for figure “Device view”

Item	Description	See section
1	Marking Options (Mini-WSB)	---
2	LED Indicators – Power Supply	“Indicating elements” > “Indicating element power supply”
3	Data contacts	“Connections” > “Data contacts/Internal data bus”
4	CAGE CLAMP® Connections for Power Supply	“Connections” > “CAGE CLAMP® connections”
5	Slot for memory card	“Memory card slot”
6	Power contacts for power supply of down-circuit I/O modules	“Connections” > “Power contacts/ Field-side supply”
7	Releasing strap	“Mounting” > “Inserting and Removing Device”
8	Service Interface (behind the flap)	“Connections” > “Service interface”
9	Mode selector switch	“Operating elements” > “Mode selector switch”
10	ETHERNET Connections	“Connections” > “Network connections ETHERNET – X1, X2”

11	Safe Locking Feature	“Mounting” > “Inserting and Removing Device”
12	Fieldbus Connection – CANopen	“Connections” > “CANopen – X4 Fieldbus Connection”
13	LED Indicators – System	“Indicating elements” > “Indicating elements Fieldbus/System”
14	Reset button (in hole)	“Operating elements” > “Reset button”

3.2 Connectors

3.2.1 Data Contacts/Internal Bus

NOTICE

Do not place the I/O modules on the gold spring contacts!
Do not place the I/O modules on the gold spring contacts in order to avoid soiling or scratching!

NOTICE

Ensure that the environment is well grounded!
The devices are equipped with electronic components that may be destroyed by electrostatic discharge. When handling the devices, ensure that the environment (persons, workplace and packing) is well grounded. Avoid touching conductive components, e.g. data contacts.

Communication between the controller and the I/O modules and system power supply for the I/O modules is provided via the internal data bus, which consists of 6 data contacts designed as self-cleaning gold spring contacts.

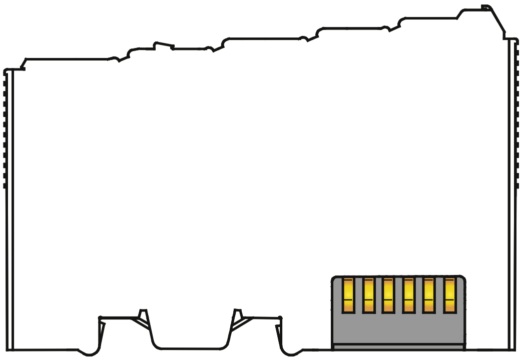


Figure 2: Data contacts

3.2.2 Power Jumper Contacts/Field Supply

⚠ CAUTION

Risk of injury due to sharp-edged blade contacts!
The blade contacts are sharp-edged. Handle the I/O module carefully to prevent injury.

The controller 750-8203is equipped with 3 self-cleaning power contacts for transferring of the field-side power supply to down-circuit I/O modules. These contacts are designed as spring contacts.

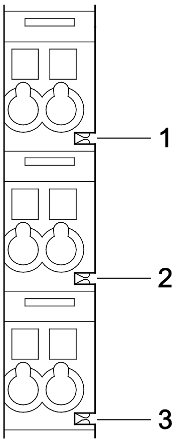


Figure 3: Power Jumper Contacts

Table 5: Legend for Figure “Power Jumper Contacts”

Contact	Type	Function
1	Spring contact	Potential transmission (U_V) for field supply
2	Spring contact	Potential transmission (0 V) for field supply
3	Spring contact	Potential transmission (ground) for field supply

NOTICE

Do not exceed maximum current via power jumper contacts!
The maximum current to flow through the power jumper contacts is 10 A. Greater currents can damage the contacts.
When configuring your system, ensure that this current is not exceeded. If exceeded, insert an additional supply module.

3.2.3 CAGE CLAMP® Connectors

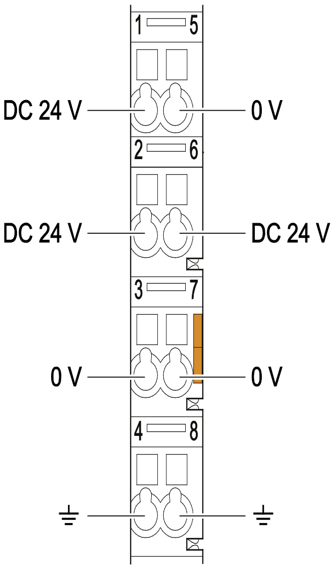


Figure 4: CAGE CLAMP® connections

Table 6: Legend for figure “CAGE CLAMP® connections”

Contact	Description	Description
1	24 V	System power supply voltage +24 V
2	+	Field-side power supply voltage U_V
3	-	Field-side power supply voltage 0 V
4	Ground	Field-side power supply voltage, ground
5	0 V	System power supply voltage 0 V
6	+	Field-side power supply voltage U_V
7	-	Field-side power supply voltage 0 V
8	Ground	Field-side power supply voltage, ground

Note

Observe supplementary power supply regulations for use in shipbuilding!
Observe supplementary power supply regulations for shipbuilding and the supply voltage in Section “Connect Devices” > … > “Supplementary Power Supply Regulations”!

3.2.4 Service Interface

The service interface is located behind the flap.

The Service interface is used for communication with WAGO-I/O-*CHECK* and WAGO-I/O-*PRO* and for firmware download.

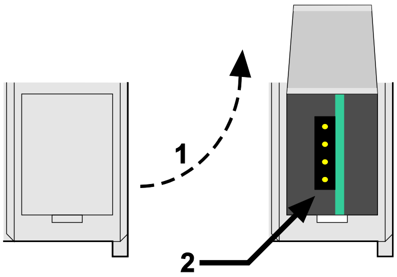


Figure 5: Service interface, (closed and open flap)

Table 7: Service interface

Number	**Description**
1	Open flap
2	Service interface

NOTICE

Device must be de-energized!
To prevent damage to the device, unplug and plug in the communication cable only when the device is de-energized!

The connection to the 4-pin header under the cover flap can be realized via the communication cables with the item numbers750-920 and 750-923 or via the WAGO radio adapter with the item number 750-921.

3.2.5 Network Connections – X1, X2

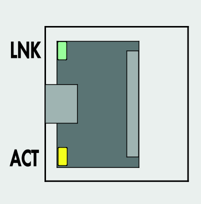


Figure 6: Network connections – X1, X2

Table 8: Legend for Figure "Network Connections – X1, X2"

Contact	Signal	Description
1	TD +	Transmit Data +
2	TD -	Transmit Data -
3	RD +	Receive Data +
4	NC	Not assigned
5	NC	Not assigned
6	RD -	Receive Data -
7	NC	Not assigned
8	NC	Not assigned

3.2.6 CANopen – X4 Fieldbus Connection

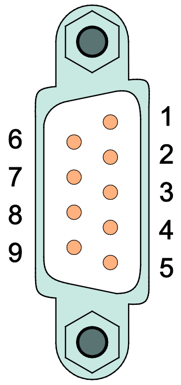


Figure 7: CANopen – X4 fieldbus connection

Table 9: Legend for Figure “CANopen – X4 Fieldbus Connection”

Contact	Signal	Description
1	-	Not used
2	CAN_L	CAN Signal Low
3	GND	Ground
4	-	Not used
5	Drain Shield	Shield termination
6	-	Not used
7	CAN_H	CAN Signal High
8	-	Not used
9	CAN_V+	Not used

DC/DC converters and optocouplers in the fieldbus interface provide electrical isolation between the CANopen bus system and the electronics.

The cable shield must be applied to the CAN shield. This is terminated to ground in devices with 1 MΩ (DIN rail contact). A low-impedance connection of the shielding to ground is possible only from the outside (e.g., by a supply module). We recommend using central ground contacts for the entire CANopen bus line shielding.

To minimize reflection at the end of the line, the CANopen line must be terminated at both ends by a cable termination.

Note

Attention - bus termination!

The CANopen bus segment must be terminated at both ends!
No more than 2 terminations per bus segment may be used!
Terminations may not be used in stub and branch lines!
Operation without proper termination of the CANopen network may result in transmission errors.

Note

Observe permissible resistor power loss!
For normal operation, 1/4Watt resistors are sufficient. In the event of a short circuit (24V power supply to a bus line), the resistor is subjected to a power loss of (short-circuit output current from transceiver * power supply voltage). The resistor must be designed to withstand this power loss level.

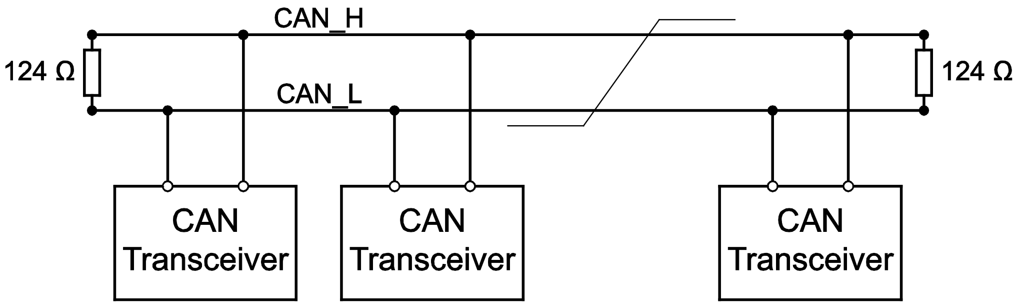


Figure 8: CANopen standard bus termination

3.3 Display Elements

3.3.1 Power Supply Indicating Elements

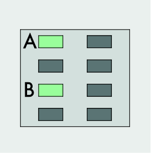


Figure 9: Power supply indicating elements

Table 10: Legend for Figure “Power Supply Indicating Elements”"

Description	Color	Description
A	Green/off	Status of system power supply voltage
B	Green/off	Status of field-side power supply voltage

3.3.2 Fieldbus/System Indicating Elements

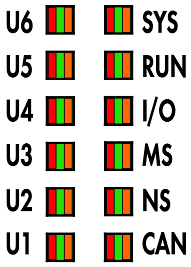


Figure 10: Indicating elements for fieldbus/system

Table 11: Legend for figure "Fieldbus/System Indicating Elements"

Description	Color	Description
SYS	Red/Green/ Orange/Off	System status
RUN	Red/Green/ Orange/Off	PLC program status
I/O	Red/Green/ Orange/Off	Internal data bus status
CG	Without function	
NS	Without function	
CAN	Red/Green/ Orange/Off	CANopen status
U6	Green/Red/ Orange/Off	User LED 6, programmable using the function blocks of the "WagoLibLed.lib" library.
U5	Green/Red/ Orange/Off	User LED 5, programmable using the function blocks of the "WagoLibLed.lib" library.
U4	Red/Green/ Orange/Off	User LED 4, programmable using the function blocks of the "WagoLibLed.lib" library.
U3	Red/Green/ Orange/Off	User LED 3, programmable using the function blocks of the "WagoLibLed.lib" library.
U2	Red/Green/ Orange/Off	User LED 2, programmable using the function blocks of the "WagoLibLed.lib" library.
U1	Red/Green/ Orange/Off	User LED 1, programmable using the function blocks of the "WagoLibLed.lib" library.

3.3.3 Memory Card Indicating Elements

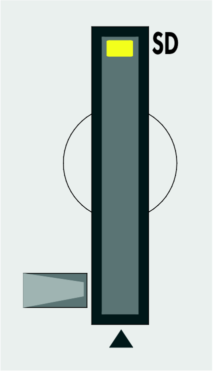


Figure 11: Indicating elements, memory card slot

Table 12: Legend for figure “Indicating Elements, memory card slot”

Description	Color	Description
SD	Yellow/Off	Memory card status

3.3.4 Network Indicating Elements

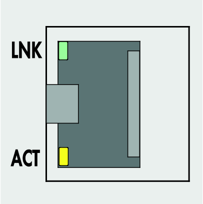


Figure 12: Indicating elements, RJ-45 jacks

Table 13: Legend for figure “Indicating elements, RJ-45 jacks”

Description	Color	Description
LNK	Green/Off	ETHERNET connection status
ACT	Yellow/Off	ETHERNET data exchange

3.4 Operating Elements

3.4.1 Operating Mode Switch

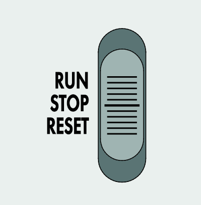


Figure 13: Mode selector switch

Table 14: Mode selector switch

Item	Activation	Function
RUN	Latching	**Normal mode** CODESYS 2 application runs.
STOP	Latching	**Stop** CODESYS 2 application stopped.
RESET	Spring-return	**Reset warm start** or **Reset cold start** (based on the duration of activation, see Section "Starting" > "Initiating Reset Functions")

Other functions can also be initiated using the reset button.

3.4.2 Reset Button

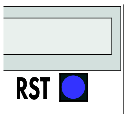


Figure 14: Reset button

The Reset button can be actuated using a suitable object (e.g., pen) through a hole in the enclosure.

Depending on the position of the mode selector switch, the Reset button can be used to initiate different functions: Software reset, factory reset or fixed IP address.

Information about these functions are given in the Section “Starting” > “Initiating Reset Functions”.

3.5 Slot for Memory Card

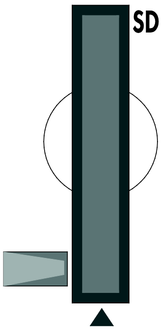


Figure 15: Memory card slot

The memory card is locked in the enclosure by a push/push mechanism. Inserting and removing the card is described in the Section “Service” > “Inserting and Removing the Memory Card”.
The memory card is protected by a cover flap, which can also be sealed.

Note

Memory card is not included in the scope of delivery!
Note, the controller is delivered without memory card.
To use a memory card, you must order one separately. The controller can also be operated without memory card expansion, the use of a memory card is optional.

Note

Only use recommended memory cards!
Use only the SD memory card available from WAGO (item No. 758-879/000-001) as it is suitable for industrial applications subjected to environmental extremes and was developed for use in the controller.
Compatibility with other commercially available storage media cannot be guaranteed.

3.6 Schematic Diagram

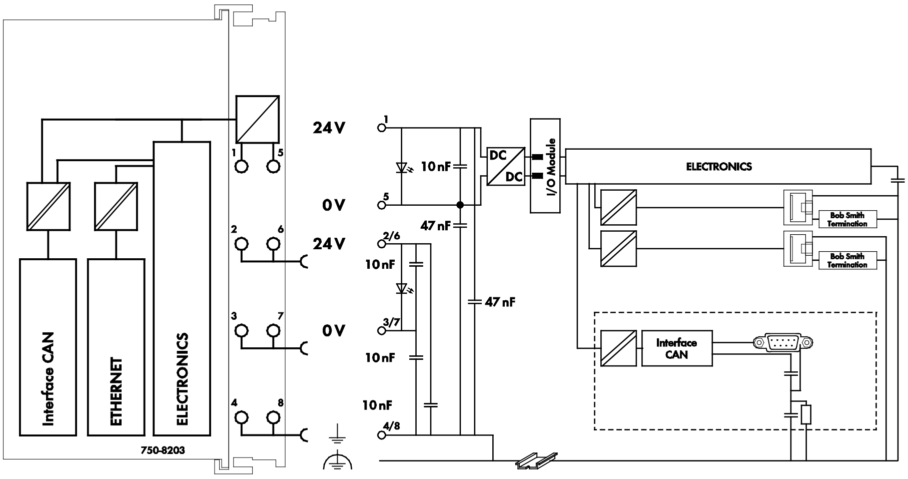


Figure 16: Schematic diagram

3.7 Technical Data

3.7.1 Device Data

Table 15: Technical data, device data

Width	75 mm
Height (from upper edge of DIN 35 rail)	65 mm
Length	100 mm
Weight	190 g

3.7.2 System Data

Table 16: Technical Data – System Data

CPU	Cortex A8, 600 MHz
Operating System	Real-time Linux® 3.6 (with RT Preemption Patch)
Main memory (RAM)	256 MB
Internal memory (flash)	256 MB
Non-volatile memory (Retain, NVRAM)	128 KB
Memory card slot	Push-push mechanism, sealing cover lid
Type of memory card	SD and SDHC up to 32 Gbytes (All guaranteed properties are valid only in connection with the WAGO 758-879/000-001 memory card.)

3.7.3 Power supply

Table 17: Technical Data – Power Supply

Power supply	24 VDC (-25 % … +30 %)
Max. input current (24 V)	550 mA
Total current for I/O modules (5V)	1700 mA
Isolation	500 V system/supply

3.7.4 Clock

Table 18: Technical data – Clock

Drift - system clock (25 °C)	20 ppm
Drift - RTC (25 °C)	3 ppm
Buffer time RTC	30 days

3.7.5 Programming

Table 19: Technical Data – Programming

Programming	WAGO-I/O-PRO V2.3
IEC 61131-3	IL, LD, FBD, ST, FC
Program memory (Flash)	16 MB
Data memory (RAM)	64 MB
Non-volatile memory (Retain + Flag, NVRAM)	128 KB

3.7.6 Internal data bus

Table 20: Technical Data – Internal Data Bus

Number of I/O modules (per node)	64
with bus extension	250
Input and output process image (max.)	1000 words

3.7.7 ETHERNET

Table 21: Technical Data – ETHERNET

ETHERNET	2 x RJ-45 (switched or separated mode)
Transmission medium	Twisted Pair S-UTP, 100 Ω, Cat 5, 100 m maximum cable length
Baud rate	10/100 Mbit/s; 10Base-T/100Base-TX
Protocols	DHCP, DNS, SNTP, FTP, FTPS, SNMP, HTTP, HTTPS, SSH, MODBUS (TCP, UDP)
MODBUS input and output process image, max.	1000 words, also with MODBUS access to the flag area (see Section "MODBUS" > … > "Flag Area")

Note

No direct access from fieldbus to the process image for I/O modules!
Any data that is required from the I/O module process image must be explicitly mapped in the CODESYS program to the data in the fieldbus process image and vice versa! Direct access is not possible!

3.7.8 CANopen

Table 22: Technical Data – CANopen

CANopen input and output process image max.	2000 words

Note

No direct access from fieldbus to the process image for I/O modules!
Any data that is required from the I/O module process image must be explicitly mapped in the CODESYS program to the data in the fieldbus process image and vice versa! Direct access is not possible!

3.7.9 Connection Type

Table 23: Technical Data – Field Wiring

Wire connection	CAGE CLAMP®
Cross section	0.08 mm² … 2.5 mm², AWG 28 … 14
Stripped lengths	8 mm … 9 mm / 0.33 in

Table 24: Technical Data – Power Jumper Contacts

Power jumper contacts	Spring contact, self-cleaning
Voltage drop at $I_{max.}$	< 1 V/64 modules

Table 25: Technical Data – Data Contacts

Data contacts	Slide contact, hard gold plated, self-cleaning

3.7.10 Climatic Environmental Conditions

Table 26: Technical Data – Climatic Environmental Conditions

Operating temperature range	0 °C … 55 °C
Operating temperature range for components with extended temperature range (750-xxx/025-xxx)	-20 °C … +60 °C
Storage temperature range	-25 °C … +85 °C
Storage temperature range for components with extended temperature range (750-xxx/025-xxx)	-40 °C … +85 °C
Relative humidity	Max. 5 % … 95 % without condensation
Resistance to harmful substances	Acc. to IEC 60068-2-42 and IEC 60068-2-43
Maximum pollutant concentration at relative humidity < 75 %	$SO_2 \leq 25$ ppm $H_2S \leq 10$ ppm
Special conditions	Ensure that additional measures for components are taken, which are used in an environment involving: – dust, caustic vapors or gases – ionizing radiation

3.8 Approvals

The following approvals have been granted to the controller “PFC200 CS 2ETH CAN” (750-8203):

Conformity Marking	
$_{C}UL_{US}$	UL508

The following ship approvals have been granted to the basic version of the controller “PFC200 CS 2ETH CAN” (750-8203):

GL (Germanischer Lloyd)	Cat. A, B, C, D (EMC 1)

3.9 Standards and Guidelines

The controller “PFC200 CS 2ETH CAN” (750-8203) fulfills the following EMC standards:

EMC CE-Immunity to interference	acc. to EN 61000-6-2: 2005
EMC CE-Emission of interference	acc. to EN 61000-6-3: 2007 + A1: 2011

4 Function Description

4.1 Network Configuration

ETHERNET interfaces X1 and X2 of the controller can be operated in Switch mode or as separate network interfaces.

Switch mode can be switched ON or OFF at run-time.

Switch mode is ON by default and during initial startup. Configuration mode is set to “DHCP”.

For interface X1, a fixed IP address can be set.
Setting a fixed IP address has no effect on the mode previously set.

4.1.1 Operation in Switch Mode

For operation in Switch mode, the TCP/IP settings such as the IP address or subnet mask apply for both X1 and X2.

When switching to Switch mode, the X1 settings are applied as a new common configuration for X1 and X2.
The device is then no longer accessible via the IP address previously set for X2. For CODESYS applications that use X2 for communication must take this into account.

4.1.2 Operation with Separate Network Interfaces

When operating with separate network interfaces, both ETHERNET interfaces can be configured and used separately.
Note that two interfaces still have the same MAC address. Therefore, they must not be operated in the same network segment.

When switching to operating with separate interfaces, interface X2 is initialized with the setting values last valid for it. The connections on the X1 interface persist.

When operating with separate interfaces and fixed IP address, the device can still be accessed via interface X2 via the regular IP address.

5 Mounting

5.1 Installation Position

Along with horizontal and vertical installation, all other installation positions are allowed.

Note

Use an end stop in the case of vertical mounting!
In the case of vertical assembly, an end stop has to be mounted as an additional safeguard against slipping.

WAGO order no. 249-116	End stop for DIN 35 rail, 6 mm wide
WAGO order no. 249-117	End stop for DIN 35 rail, 10 mm wide

5.2 Overall Configuration

The maximum total length of a fieldbus node without fieldbus coupler/controller is 780 mm including end module. The width of the end module is 12 mm. When assembled, the I/O modules have a maximum length of 768 mm.

Examples:

- 64 I/O modules with a 12 mm width can be connected to a fieldbus coupler/controller.
- 32 I/O modules with a 24 mm width can be connected to a fieldbus coupler/controller.

Exception:

The number of connected I/O modules also depends on the type of fieldbus coupler/controller is used. For example, the maximum number of stackable I/O modules on one PROFIBUS DP/V1 fieldbus coupler/controller is 63 with no passive I/O modules and end module.

NOTICE

Observe maximum total length of a fieldbus node!
The maximum total length of a fieldbus node without fieldbus coupler/controller and without using a 750-628 I/O Module (coupler module for internal data bus extension) may not exceed 780 mm.
Also note the limitations of individual fieldbus couplers/controllers.

Note

Increase the total length using a coupler module for internal data bus extension!

You can increase the total length of a fieldbus node by using a 750-628 I/O Module (coupler module for internal data bus extension). For such a configuration, attach a 750-627 I/O Module (end module for internal data bus extension) after the last I/O module of a module assembly. Use an RJ-45 patch cable to connect the I/O module to the coupler module for internal data bus extension of another module block.

This allows you to segment a fieldbus node into a maximum of 11 blocks with maximum of 10 I/O modules for internal data bus extension.

The maximum cable length between two blocks is five meters.

More information is available in the manuals for the 750-627 and 750-628 I/O Modules.

5.3 Mounting onto Carrier Rail

5.3.1 Carrier Rail Properties

All system components can be snapped directly onto a carrier rail in accordance with the European standard EN 50022 (DIN 35).

NOTICE

Do not use any third-party carrier rails without approval by WAGO!
WAGO Kontakttechnik GmbH & Co. KG supplies standardized carrier rails that are optimal for use with the I/O system. If other carrier rails are used, then a technical inspection and approval of the rail by WAGO Kontakttechnik GmbH & Co. KG should take place.

Carrier rails have different mechanical and electrical properties. For the optimal system setup on a carrier rail, certain guidelines must be observed:

- The material must be non-corrosive.
- Most components have a contact to the carrier rail to ground electro-magnetic disturbances. In order to avoid corrosion, this tin-plated carrier rail contact must not form a galvanic cell with the material of the carrier rail which generates a differential voltage above 0.5 V (saline solution of 0.3 % at 20°C).
- The carrier rail must optimally support the EMC measures integrated into the system and the shielding of the I/O module connections.
- A sufficiently stable carrier rail should be selected and, if necessary, several mounting points (every 20 cm) should be used in order to prevent bending and twisting (torsion).
- The geometry of the carrier rail must not be altered in order to secure the safe hold of the components. In particular, when shortening or mounting the carrier rail, it must not be crushed or bent.
- The base of the I/O components extends into the profile of the carrier rail. For carrier rails with a height of 7.5 mm, mounting points are to be riveted under the node in the carrier rail (slotted head captive screws or blind rivets).
- The medal springs on the bottom of the housing must have low-impedance contact with the DIN rail (wide contact surface is possible).

5.3.2 WAGO DIN Rail

WAGO carrier rails meet the electrical and mechanical requirements shown in the table below.

Table 27: WAGO DIN Rail

Order number	Description
210-113 /-112	35 x 7.5; 1 mm; steel yellow chromated; slotted/unslotted
210-114 /-197	35 x 15; 1.5 mm; steel yellow chromated; slotted/unslotted
210-118	35 x 15; 2.3 mm; steel yellow chromated; unslotted
210-198	35 x 15; 2.3 mm; copper; unslotted
210-196	35 x 7.5; 1 mm; aluminum; unslotted

5.4 Spacing

The spacing between adjacent components, cable conduits, casing and frame sides must be maintained for the complete fieldbus node.

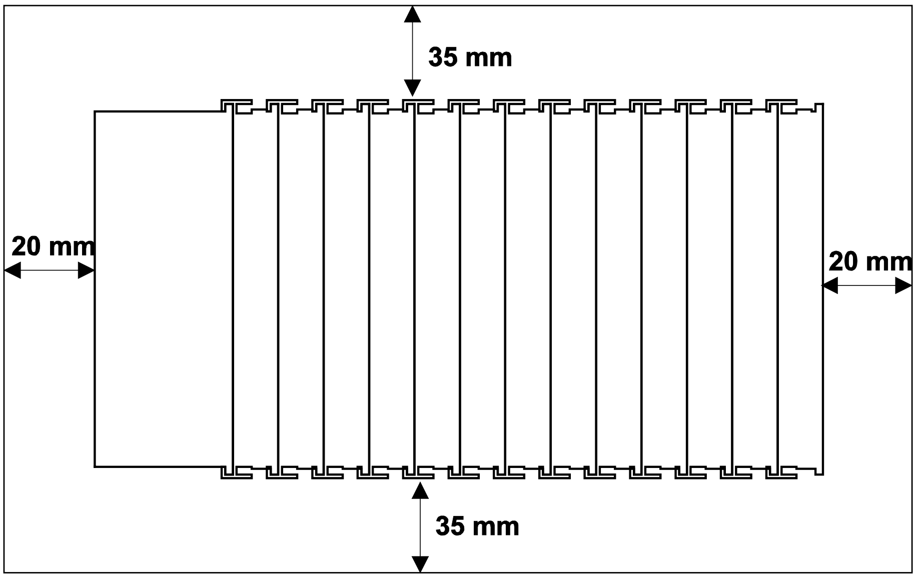


Figure 17: Spacing

The spacing creates room for heat transfer, installation or wiring. The spacing to cable conduits also prevents conducted electromagnetic interferences from influencing the operation.

5.5 Mounting Sequence

Fieldbus couplers/controllers and I/O modules of the WAGO-I/O-SYSTEM 750 are snapped directly on a carrier rail in accordance with the European standard EN 50022 (DIN 35).

The reliable positioning and connection is made using a tongue and groove system. Due to the automatic locking, the individual devices are securely seated on the rail after installation.

Starting with the fieldbus coupler/controller, the I/O modules are mounted adjacent to each other according to the project design. Errors in the design of the node in terms of the potential groups (connection via the power contacts) are recognized, as the I/O modules with power contacts (blade contacts) cannot be linked to I/O modules with fewer power contacts.

⚠ CAUTION

Risk of injury due to sharp-edged blade contacts!
The blade contacts are sharp-edged. Handle the I/O module carefully to prevent injury.

NOTICE

Insert I/O modules only from the proper direction!
All I/O modules feature grooves for power jumper contacts on the right side. For some I/O modules, the grooves are closed on the top. Therefore, I/O modules featuring a power jumper contact on the left side cannot be snapped from the top. This mechanical coding helps to avoid configuration errors, which may destroy the I/O modules. Therefore, insert I/O modules only from the right and from the top.

Note

Don't forget the bus end module!
Always plug a bus end module 750-600 onto the end of the fieldbus node! You must always use a bus end module at all fieldbus nodes with WAGO-I/O-SYSTEM 750 fieldbus couplers/controllers to guarantee proper data transfer.

5.6 Inserting Devices

NOTICE

Perform work on devices only if they are de-energized!
Working on energized devices can damage them. Therefore, turn off the power supply before working on the devices.

5.6.1 Inserting the Fieldbus Coupler/Controller

1. When replacing the fieldbus coupler/controller for an already available fieldbus coupler/controller, position the new fieldbus coupler/controller so that the tongue and groove joints to the subsequent I/O module are engaged.

2. Snap the fieldbus coupler/controller onto the carrier rail.

3. Use a screwdriver blade to turn the locking disc until the nose of the locking disc engages behind the carrier rail (see the following figure). This prevents the fieldbus coupler/controller from canting on the carrier rail.

With the fieldbus coupler/controller snapped in place, the electrical connections for the data contacts and power contacts (if any) to the possible subsequent I/O module are established.

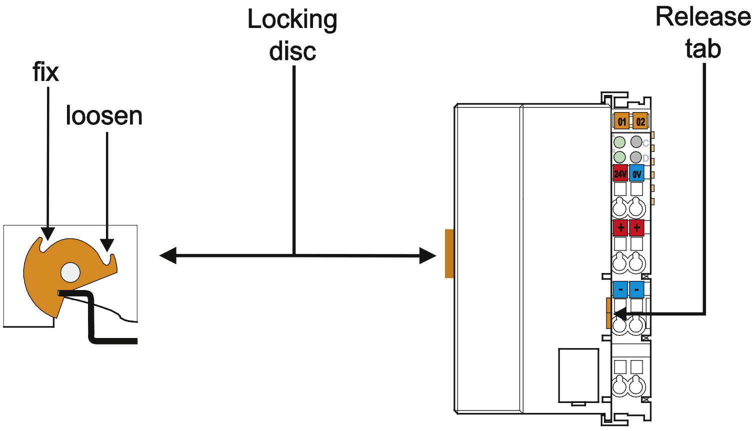


Figure 18: Release Tab of Controller

5.6.2 Inserting the I/O Module

1. Position the I/O module so that the tongue and groove joints to the fieldbus coupler/controller or to the previous or possibly subsequent I/O module are engaged.

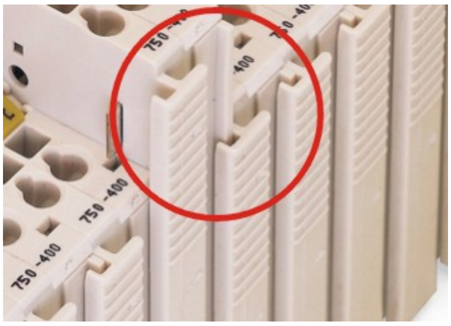


Figure 19: Insert I/O Module (Example)

2. Press the I/O module into the assembly until the I/O module snaps into the carrier rail.

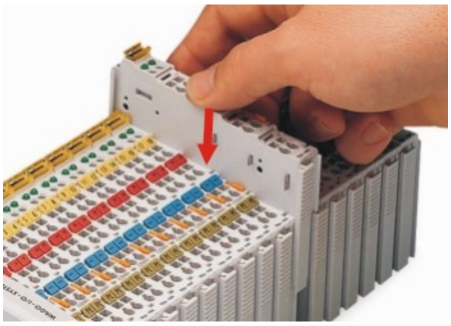


Figure 20: Snap the I/O Module into Place (Example)

With the I/O module snapped in place, the electrical connections for the data contacts and power jumper contacts (if any) to the fieldbus coupler/controller or to the previous or possibly subsequent I/O module are established.

6 Connect Devices

6.1 Connecting a Conductor to the CAGE CLAMP®

The WAGO CAGE CLAMP® connection is appropriate for solid, stranded and finely stranded conductors.

Note

Only connect one conductor to each CAGE CLAMP®!
Only one conductor may be connected to each CAGE CLAMP®.
Do not connect more than one conductor at one single connection!

If more than one conductor must be routed to one connection, these must be connected in an up-circuit wiring assembly, for example using WAGO feed-through terminals.

Exception:
If it is unavoidable to jointly connect 2 conductors, then you must use a ferrule to join the wires together. The following ferrules can be used:

Length:	8 mm
Nominal cross section $_{max.}$:	1 mm^2 for 2 conductors with 0.5 mm^2 each
WAGO product:	216-103 or products with comparable properties

1. For opening the CAGE CLAMP® insert the actuating tool into the opening above the connection.

2. Insert the conductor into the corresponding connection opening.

3. For closing the CAGE CLAMP® simply remove the tool. The conductor is now clamped firmly in place.

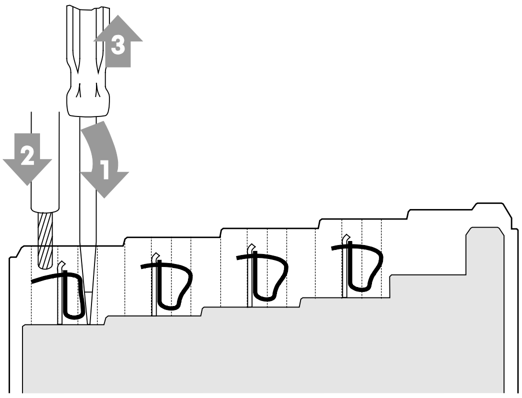


Figure 21: Connecting a Conductor to a CAGE CLAMP®

6.2 Power Supply Concept

6.2.1 Supplementary Power Supply Regulations

The WAGO-I/O-SYSTEM 750 can also be used in shipbuilding or offshore and onshore areas of work (e. g. working platforms, loading plants). This is demonstrated by complying with the standards of influential classification companies such as Germanischer Lloyd and Lloyds Register.

Filter modules for 24 V supply are required for the certified operation of the system.

Table 28: Filter Modules for 24 V Supply

Order No.	Name	Description
750-626	Supply Filter	Filter module for system supply and field supply (24 V, 0 V), i. e. for fieldbus coupler/controller and bus power supply (750-613)
750-624	Supply Filter	Filter module for the 24 V field supply (750-602, 750-601, 750-610)

Therefore, the following power supply concept must be absolutely complied with.

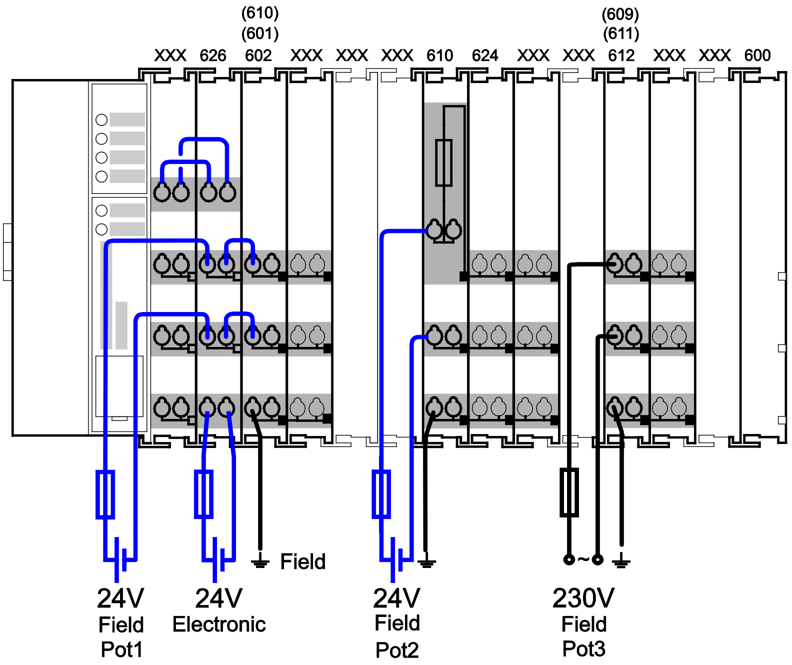


Figure 22: Power Supply Concept

➔ **Note**

Use a supply module for equipotential bonding!
Use an additional 750-601/ 602/ 610 Supply Module behind the 750-626 Filter Module if you want to use the lower power jumper contact for equipotential bonding, e.g., between shielded connections and require an additional tap for this potential.

7 Commissioning

7.1 Switching On the Controller

Before switching on the controller ensure that you

- have properly installed the controller
(see section “Installation”),
- have connected all required data cables (see section “Connections”) to the corresponding interfaces and have secured the connectors by their attached locking screws,
- have connected the electronics and field-side power supply
(see section “Connections”),
- have mounted the end module (750-600)
(see Section “Installation”),
- have performed appropriate potential equalization at your machine/system (see System Description for 750-xxx) and
- have performed shielding properly (see System Description for 750-xxx).

To switch on both the controller and the connected I/O modules, switch on your power supply unit.

Starting of the controller is indicated by a brief green flashing of all LEDs. After a few seconds the SYS LED will signal successful boot-up of the controller. The CODESYS 2.3 runtime system is started simultaneously.

Once the entire system has been successfully started, the SYS and I/O LEDs light up green.

If there is no executable IEC-61131-3 program stored on the controller, or if the RUN/STOP switch is set to STOP, the RUN LED will light up red.

7.2 Determining the IP Address of the Host PC

To ensure that the host PC can communicate with the controller via ETHERNET, both devices must be located in the same subnet.

To determine the IP address of the host PC (with the Microsoft Windows® operating system) using the MS DOS prompt, proceed as follows:

1. Open the MS DOS prompt window.
   To do this, enter the command “cmd” in the input field under **Start** > **Execute…** > **Open:** (Windows® XP) or **Start** > **Search programs/files** (Windows® 7) and then click **[OK]** or press **[Enter]**.

2. In the MS DOS prompt enter the command “ipconfig” and then press **[Enter]**.

3. The IP address, subnet mask and standard gateway, including the appropriate parameters, are displayed.

7.3 Setting an IP Address

In the controller's initial state the following IP addresses are active for the ETHERNET interface (Port X1 and Port X2):

Table 29: Default IP Addresses for ETHERNET Interfaces

Ethernet interface	**Default setting**
X1/X2	Dynamic assignment of IP address using “Dynamic Host Configuration Protocol” (DHCP)

Adapt IP addressing for your specific system structure to ensure that the PC and the controller can communicate with one another using one of the available configuration tools (WBM, “WAGO Ethernet Settings”, CBM) (see section “Configuration”).

Example for incorporating the controller (192.168.2.17) into an existing network:
If the IP address of your host PC is 192.168.1.2, for example, then the controller must be on the same subnet. That is, with the net mask **255.255.255**.0, the first three digits of the controller must match those of your PC. This yields the following address range for the controller:

Table 30: Network Mask 255.255.255.0

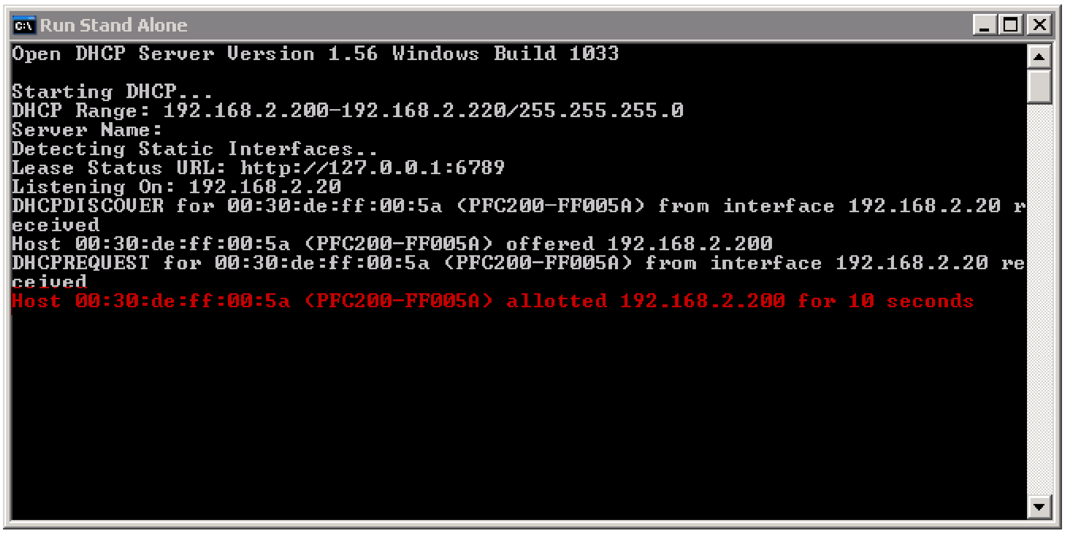
Host PC	**Subnet address range for the controller**
192.168.1.2	**192.168.1**.3 … **192.168.1**.254

7.3.1 Assigning an IP Address using DHCP

The PFC200 can obtain dynamic IP addresses from a server (DHCP/BootP).
In contrast to fixed IP addresses, dynamically assigned addresses are not stored permanently. Therefore, a BootP or DHCP server must be available each time the controller is restarted.

If an IP address has been assigned by means of DHCP (default setting), it can be determined through the settings and the output of the specific DHCP server.

In the example figure shown here, the corresponding output of “Open DHCP” is presented.



```
Open DHCP Server Version 1.56 Windows Build 1033

Starting DHCP...
DHCP Range: 192.168.2.200-192.168.2.220/255.255.255.0
Server Name:
Detecting Static Interfaces..
Lease Status URL: http://127.0.0.1:6789
Listening On: 192.168.2.20
DHCPDISCOVER for 00:30:de:ff:00:5a (PFC200-FF005A) from interface 192.168.2.20 r
eceived
Host 00:30:de:ff:00:5a (PFC200-FF005A) offered 192.168.2.200
DHCPREQUEST for 00:30:de:ff:00:5a (PFC200-FF005A) from interface 192.168.2.20 re
ceived
Host 00:30:de:ff:00:5a (PFC200-FF005A) allotted 192.168.2.200 for 10 seconds
```

Figure 23: “Open DHCP”, example figure

In conjunction with the DNS server associated with DHCP, the device can be reached using its host name.
This name consists of the prefix “PFC200-“ and the last six places of the MAC address (in the example shown here: “00:30:DE:FF:00:5A”). The MAC address of the device can be printed on the label on the side of the device.

The host name of the device in the example shown here is thus “PFC200-FF005A”.

7.3.2 Changing an IP Address using “WAGO Ethernet Settings”

The Microsoft Windows® application “WAGO Ethernet Settings” is a software used to identify the controller and configure network settings.

Note

Observe the software version!

To configure the controller use at least Version 5.4.2.3 dated July 20, 2013 in “WAGO Ethernet Settings”!

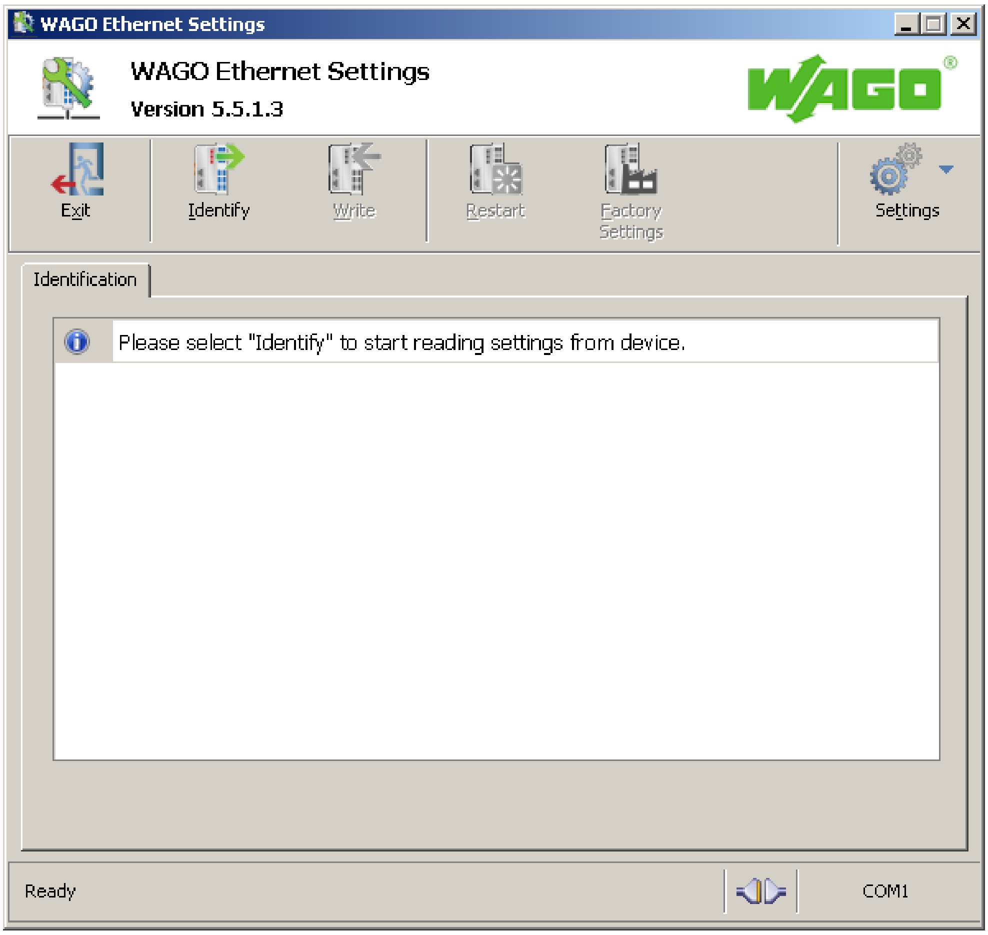
You can use WAGO communication cables or WAGO radio adapters or even the IP network for data communication.

1. Switch off the power supply to the controller.

2. Connect the 750-920 communication cable to the Service interface on the controller and to a serial interface of your PC.

3. Switch the power supply to the controller on again.

4. Start the “WAGO Ethernet Settings” program.



WAGO Ethernet Settings

WAGO Ethernet Settings
Version 5.5.1.3

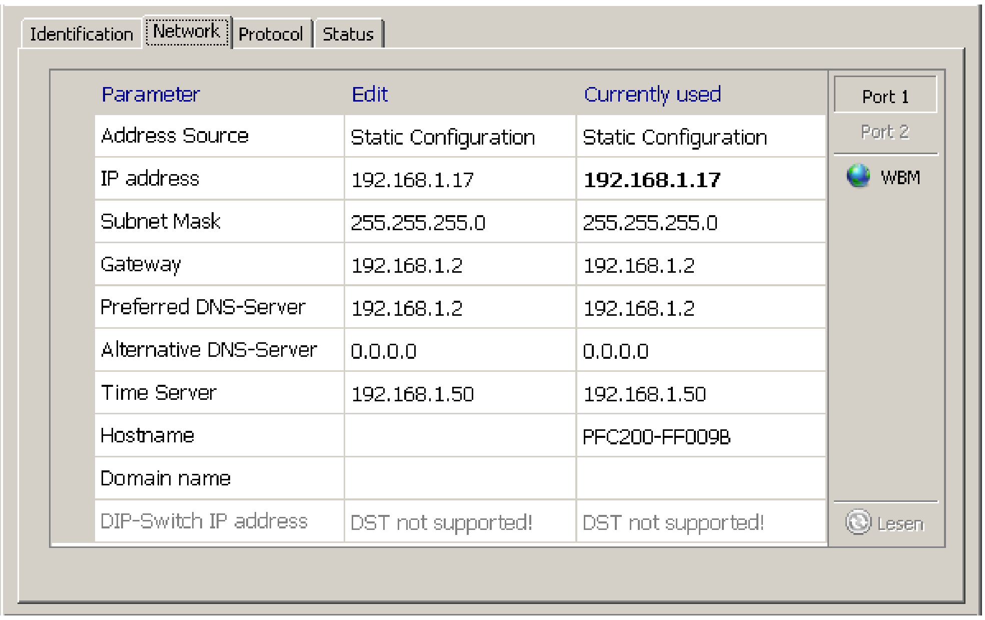
Please select "Identify" to start reading settings from device.

Ready

Figure 24: "WAGO Ethernet Settings" – Starting screen

5. Click **[Identify]** to read in and identify the connected PFC200.

6. Select the “Network” tab:



Parameter	Edit	Currently used
Address Source	Static Configuration	Static Configuration
IP address	192.168.1.17	**192.168.1.17**
Subnet Mask	255.255.255.0	255.255.255.0
Gateway	192.168.1.2	192.168.1.2
Preferred DNS-Server	192.168.1.2	192.168.1.2
Alternative DNS-Server	0.0.0.0	0.0.0.0
Time Server	192.168.1.50	192.168.1.50
Hostname		PFC200-FF009B
Domain name		
DIP-Switch IP address	DST not supported!	DST not supported!

Figure 25: “WAGO Ethernet Settings” – “Network” tab

7. To assign a fixed address, select “Static configuration” on the “Source” line under “Input”. DHCP is normally activated as the default setting.

8. In the column “Input” enter the required IP address and, if applicable, the address of the subnet mask and of the gateway.

9. Click on **[Write]** to accept the address in the PFC200. (When you click the **[Write]** button “WAGO Ethernet Settings” will automatically restart your controller. This action therefore requires about 30 seconds.)

10. You can now close “WAGO Ethernet Settings”, or make other changes directly in the Web-based Management system as required. To do this, click on **[WBM]** at the right in the window.

7.4 Testing the Network Connection

Carry out a ping network function to check whether you can reach the controller at the IP address you have assigned in the network.

1. Open the MS DOS prompt window.
   To do this, enter the command “cmd” in the input field under **Start** > **Execute…** > **Open:** (Windows® XP) or **Start** > **Search programs/files** (Windows® 7) and then click **[OK]** or press **[Enter]**.

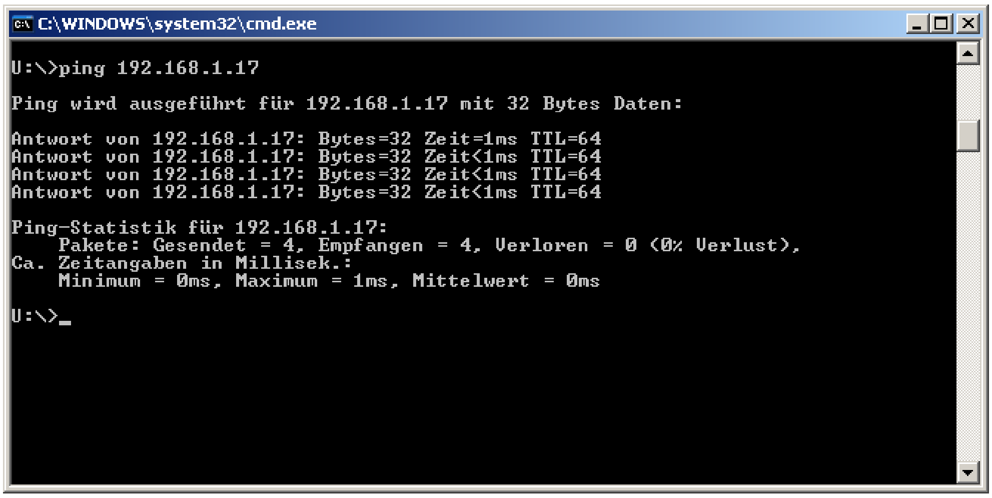
2. In the MS DOS window, enter the command “ping” and the IP address of the controller (for example, `ping 192.168.1.17`) and then press **[Enter]**.

Note

Host entries in the ARP table!

It may also be useful to delete the current host entries in the ARP table with the command “arp -d *” before executing the “ping” command (as administrator in Windows® 7). This ensures that older entries will not impair the success of the “ping” command.

3. Your PC sends out a query that is answered by the controller. This reply appears in the MS DOS prompt window. If the error message “Timeout” appears, the controller has not responded properly. You then need to check your network settings.



```
U:\>ping 192.168.1.17

Ping wird ausgeführt für 192.168.1.17 mit 32 Bytes Daten:

Antwort von 192.168.1.17: Bytes=32 Zeit=1ms TTL=64
Antwort von 192.168.1.17: Bytes=32 Zeit<1ms TTL=64
Antwort von 192.168.1.17: Bytes=32 Zeit<1ms TTL=64
Antwort von 192.168.1.17: Bytes=32 Zeit<1ms TTL=64

Ping-Statistik für 192.168.1.17:
    Pakete: Gesendet = 4, Empfangen = 4, Verloren = 0 (0% Verlust),
Ca. Zeitangaben in Millisek.:
    Minimum = 0ms, Maximum = 1ms, Mittelwert = 0ms

U:\>_
```

Figure 26: Example of a function test

4. If the test is completed successfully, close the MS DOS window.

7.5 Shutdown/Restart

Switch off the power supply to shut down the controller.

To perform a controller restart press and hold the Reset ALL button for more than 7 seconds until all of the LEDs go out.
The controller then performs a restart.
As an alternative, you can switch off the controller and switch it back on again.
A controller restart is signaled by all LEDs briefly lighting up green.

7.6 Initiating Reset Functions

You can initiate various reset functions using the mode selector switch and the Reset button (RST).

7.6.1 Warm Start Reset

The CODESYS 2 application is reset on a warm start reset.
This corresponds to the CODESYS 2 IDE command “Reset”.

To perform a warm start reset, set the mode selector switch to "Reset" and hold it there for two to seven seconds.
Execution of the reset is signaled by the red “RUN LED” briefly going out when the mode selector switch is released.

7.6.2 Cold Start Reset

On a cold start reset the CODESYS 2 application is reset and the memory containing the retain variables is cleared.
This corresponds to the CODESYS 2 IDE command “Reset (Cold)”.

To perform a cold start reset, set the mode selector switch to “Reset” and hold it there for more than seven seconds.
Execution of the reset is signaled after seven seconds by the “RUN” LED going out for an extended period. You can then release the mode selector switch.

7.6.3 Software Reset

The controller is restarted on a software reset.

To perform a software reset, set the mode selector switch to RUN or STOP and then press the Reset button (RST) for one to eight seconds.
All LEDs will light up briefly in green to signal reset completion.

7.6.4 Setting a Fixed IP Address

This procedure sets the IP address for the X1 interfaces to the fixed address “192.168.1.17”.
When the switch is enabled, the fixed address is also used for interface X2.
When the switch is disabled, the original address setting for interface X2 is not changed.
No reset is performed.

To make this setting, set the mode selector switch to STOP and press and hold the Reset button (RST) for longer than eight seconds.
Execution of the setting is signaled by the “SYS” LED flashing orange.

To cancel this setting, perform a software reset or switch off the controller and then switch it back on.

7.6.5 Factory Reset

A factory reset restores the device to its initial state on delivery. The controller is then restarted.

Note

Any subsequently installed firmware functions are overwritten!
Firmware functions installed after initial controller startup are overwritten by a factory reset, and the firmware is reset to the version marked on the device.
After factory reset, some of the functions described in this manual may no longer be available.
To restore the current operating status, you will need:

- a backup of the currently programmed application,
- the currently installed firmware,
- the manual's version that is applicable to the factory settings.

If you have any questions, feel free to contact our WAGO Support.

To perform a factory reset, set the mode selector switch to "RESET" and press the Reset button (RST) for 1 … 8 seconds. Briefly release the Reset button (RST) (< 1 second) and press it again until the "CAN" LED lights up red. If the "CAN" LED lights up red, release the mode selector switch and the Reset button.
After the first 1 … 8 seconds, the controller reboots (all LEDs light up orange) and after another 3 seconds, the "Factory reset" process begins. The process is indicated by all LEDs lighting up red in succession.

Alternatively, you can also initiate a factory reset when you switch on the controller. Hold the mode selector switch at "RESET" for at least 3 seconds when you switch on the controller and hold the Reset button (RST) down until the "CAN" LED lights up red. If the "CAN" LED lights up red, release the mode selector switch and the Reset button.

Note

Do not interrupt the reset process!
If you release the Reset button (RST) too soon (after the Reset process), the controller switches to production mode (indicated by a green "CAN" LED). If this happens, switch the controller off and then back on again.

Note

Do not switch off!
Do not interrupt, i.e., switch off the controller, during the factory reset process! In other words, do not switch off the controller while this process is in progress!

7.7 Users and Passwords

Several groups of users are provided in the controller which can be used for various services.

Standard passwords are set for all users. We strongly recommend changing these passwords on startup!

Note

Change passwords
Standard passwords are documented in these instructions and therefore do not offer adequate protection! Change the passwords to meet your particular needs.

7.7.1 Services and Users

All password-protected services and their associated users are listed in the following table.

Service	WBM		Linux®			SNMP user
	admin	user	root	admin	user	
Web Based Management (WBM)	X	X				
Linux® console			X	X	X	
Console Based Management (CBM)			X	X		
CODESYS				X		
Telnet			X	X	X	
FTP			X	X	X	
FTPS			X	X	X	
SSH			X	X	X	
SNMP						X

7.7.2 WBM Group

WBM has its own user administration system. The users in this system are isolated from the other user groups in the system for security reasons.

Detailed information about this is given in the Section “WBM User Administration”.

Table 31: WBM Users

Users	Permissions	Default Password
admin	All (administrator)	wago
user	Supported to a limited extent:	user
guest	Display only	---

7.7.3 Linux User Group

The Linux® users group include the actual users of the operating system, which is likewise used by most services.

The passwords for these users must be configured through a terminal connection via SSH/RS-232.

Table 32: Linux® Users

User	Special Feature	Home Directory	Default Password
root	Super user	/root	wago
admin	CODESYS user	/home/admin	wago
user	Normal user	/home/user	user

7.7.4 SNMP User Group

The SNMP service manages its own users. In its initial state, no users are stored in the system.

7.8 Configuration

The following methods are available for configuring the PFC200:

- Access to the Web-based management system via the PC using an Internet browser (“Configuration using Web-Based Management (WBM)”)
- Access to the “Console-Based Management” system (CBM) via the PC using a terminal program (via ETHERNET) (“Configuration using a Terminal Program”)
- Access via the CODESYS PLC program using the WagoConfigToolLIB.lib library (“Appendix” > “WagoConfigToolLIB.lib”)
- Access via the PC using “WAGO Ethernet Settings” (“Configuration using 'WAGO Ethernet Settings'”).

The CBM basically provides the same parameters for configuration of the PFC200 as the WBM, except for parameters which cannot be represented in a useful manner in a terminal window.
For an explanation of these parameters refer to the sections starting at “'Information' Page”.

7.8.1 Configuration using Web-Based Management (WBM)

The implemented HTML pages (from here on referred to as “pages”) for Web-Based Management configure the PFC200. Proceed as follows for access the WBM using an Internet browser:

1. Connect the PFC200 to the ETHERNET network via the ETHERNET interface X1.

2. To access the pages, enter the controller's IP address in the address line of your browser followed by “/wbm”, e.g., “http://192.168.1.17/wbm”. Note that the PC and the PFC200 must be located within the same subnet (see Section “Setting an IP Address”). If you do not know the IP address and cannot determine it, switch the controller to the pre-set address “192.168.1.17” using the “Fixed IP address” function (see Section “Initiate Reset Functions” > “Set Fixed IP Address”).

If you have installed a DHCP server on your PC and would like to access WBM through DHCP, use the other interface. You can find detailed information about this in the section “Assigning an IP Address Using DHCP”.

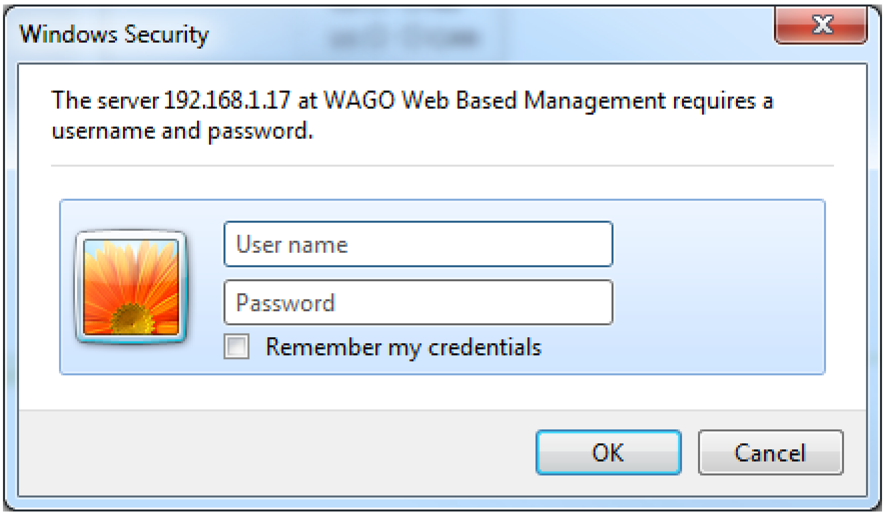
Note

Displaying the PFC200 Start Page
If the PFC200 does not display the start page, ensure that your Internet browser settings permit bypassing the proxy server for local addresses. Also check whether your PC is located in the same subnet as the PFC200.

Note

Take usage by the CODESYS program into account
If the PFC200 is at capacity due to a CODESYS program, this may result in slower processing in the WBM. As a result, timeout errors are sometimes reported under some circumstances. It is therefore important to stop the CODESYS application prior to performing complicated configuration using WBM.

Some pages of the WBM are password protected. The first time you select an item from the navigation bar, the password query appears:


Windows Security

The server 192.168.1.17 at WAGO Web Based Management requires a username and password.

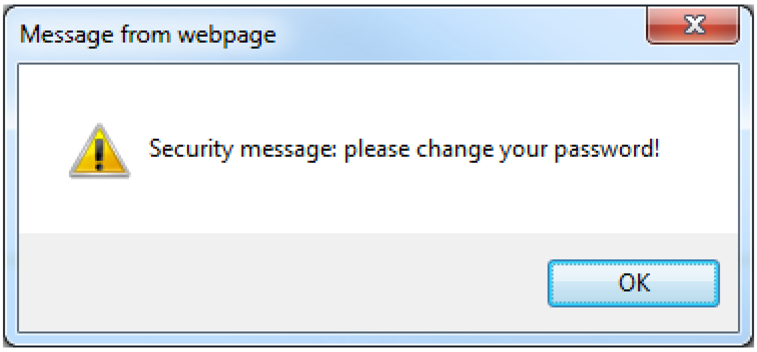
Figure 27: Entering authentication

7.8.1.1 WBM User Administration

To allow settings to be made only by a select number of users, limit access to WBM functions through User Administration.

Note

Change passwords
Standard passwords are documented in these instructions and therefore do not offer adequate protection. Change the passwords to meet your particular needs, see Section “'Administration - Users' Page”.
If you do not change the passwords, a warning is displayed on every requested web page after logging in.


Message from webpage

Security message: please change your password!

Figure 28: Password reminder

Table 33: User Settings in the Initial State

Users	Password
user	user
admin	wago

Note

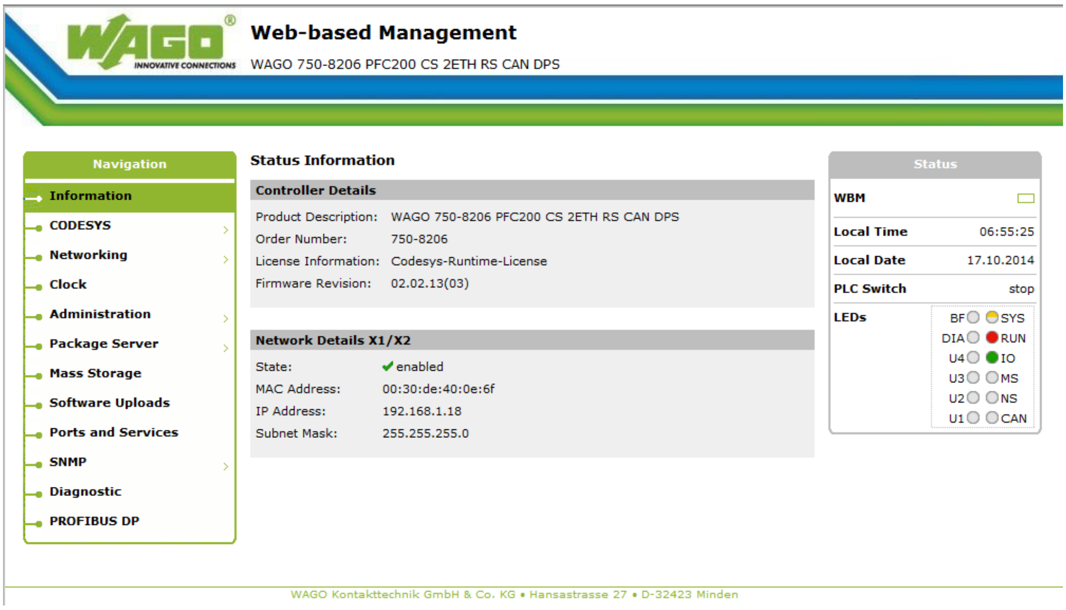
Observe access rights
Users in WBM are authorized exclusively for access to Web pages. User administration for controller applications is configured separately.

Access to the WBM pages is as follows:

Table 34: Access Rights for WBM Pages

Navigation	WBM page	User
Information	Status Information	---
CODESYS		
– Information	CODESYS Information	---
– General Configuration	CODESYS Configuration	user, admin
– WebVisu	CODESYS WebVisu	---
Networking		
– Host/Domain Name	Configuration of Network Parameters	user, admin
– TCP/IP	TCP/IP Configuration	user, admin
– Ethernet	Configuration of Ethernet Parameters	user, admin
Clock	Configuration of Date and Time	user, admin
Administration		
– Users	Configuration of the users for the Web-based Management	admin
– Create Image	Create bootable Image	admin
– Reboot	Reboot Controller	admin
Package Server		
– Firmware Backup	Firmware Backup	admin
– Firmware Restore	Firmware Restore	admin
– System Partition	System Partition	admin
Mass Storage	Mass Storage	admin
Software Uploads	Software Uploads	admin
Ports and Services		
– Network Services	Configuration of Network Services	user, admin
– NTP Client	Configuration of NTP Client	user, admin
– CODESYS Services	Configuration of the CODESYS Services	user, admin
– SSH	SSH Client Settings	user, admin
– TFTP	TFTP Server	user, admin
SNMP		
– General Configuration	Configuration of SNMP parameter	admin
– v1/v2c	Configuration of SNMP parameter	admin
– v3	Configuration of SNMP v3 Users	admin
Diagnostic	Diagnostic Information	---

7.8.1.2 General Information about the Page



Web-based Management
WAGO 750-8206 PFC200 CS 2ETH RS CAN DPS

Status Information

Controller Details

Product Description:	WAGO 750-8206 PFC200 CS 2ETH RS CAN DPS
Order Number:	750-8206
License Information:	Codesys-Runtime-License
Firmware Revision:	02.02.13(03)

Network Details X1/X2

State:	✓ enabled
MAC Address:	00:30:de:40:0e:6f
IP Address:	192.168.1.18
Subnet Mask:	255.255.255.0

Status

WBM	□
Local Time	06:55:25
Local Date	17.10.2014
PLC Switch	stop
LEDs	BF ○ ● SYS DIA ○ ● RUN U4 ○ ● IO U3 ○ ○ MS U2 ○ ○ NS U1 ○ ○ CAN

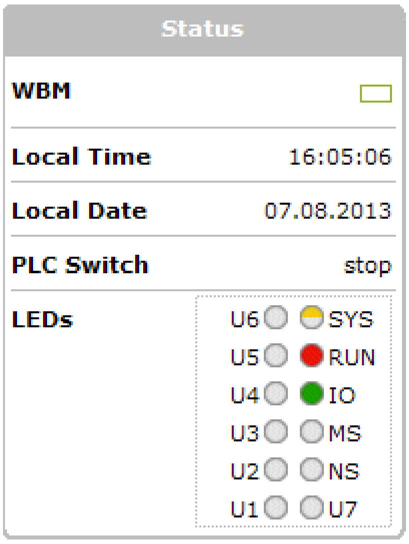
WAGO Kontakttechnik GmbH & Co. KG • Hansastrasse 27 • D-32423 Minden

Figure 29: WBM browser window (example)

The device name is displayed in the header of the browser window.

The navigation tree is shown at the left edge of the browser window. You can use this navigation tree to go to the individual pages and, where provided, subpages included in these pages.

A status area with the following elements is displayed at the right edge:



Status

WBM	□
Local Time	16:05:06
Local Date	07.08.2013
PLC Switch	stop
LEDs	U6 ○ ● SYS U5 ○ ● RUN U4 ○ ● IO U3 ○ ○ MS U2 ○ ○ NS U1 ○ ○ U7

Figure 30: WBM status information (example)

- WBM status:
Here, you can recognize whether the WBM is currently communicating with the device in the background, i.e., one or more queries have been sent and the browser is waiting on a response. Movement is then visible in the graphic. This occurs when data are read on initial call-up of the page, when

the user has sent off a change form or when data is reloaded automatically in cycles, e.g., the contents of the status area.

- Local Time:
  Local time on the device

- Local Date:
  Local date on the device

- PLC Switch:
  Setting of the mode selector switch

- LEDs:
  Here, the the status of the device LEDs is indicated. All of the LEDs are symbolized by a diagram and are labeled with their particular designation (e.g., SYS, RUN, etc.). The following colors may be used:

  - gray:
  LED is off

  - full color (green, red, yellow, orange):
  The LED is activated in the particular color

  - half color:
  The LED is flashing in the corresponding color. The other half of the surface is then either gray or also colored. The latter case indicates that the LED is flashing sequentially in different colors.

  A tool tip containing more detailed information opens as long as the cursor is positioned over an LED. The text that is displayed also contains the message that put the LED into its current status. The time of the message is also shown.
  The statuses displayed in the WBM will not always correspond at the precise time to those on the PFC200. Data has a runtime during transmission and can only be queried at a certain interval. The time period between two queries is 30 seconds.

The contents of the individual pages and sub-pages are explained in the following sections.

7.8.1.3 “Status Information” Page

The following tables explain the parameters listed on this page:

7.8.1.3.1 “Controller Details” Group

Table 35: WBM “Status Information” page – “Controller Details” group

Parameters	Explanation
Product Description	Controller identification
Order Number	Item number of the controller
License Information	Notification that the CODESYS runtime system is available
Firmware Revision	Firmware status

7.8.1.3.2 “Network Details (Xn)” Group(s)

If the switch is enabled, one group (“Network Details”) is shown for both connections.
If the switch is disabled, one group (“Network Details X1” / “Network Details X2”) is shown for each connection.

Table 36: WBM “Status Information” page – “Network Details (Xn)” group(s)

Parameters	Explanation
State	Status of the ETHERNET interface (enabled/disabled)
Mac Address	MAC address identifies and addresses the controller
IP Address	Current IP address of the controller
Subnet Mask	Current subnet mask of the controller

7.8.1.4 “CODESYS Configuration” Page

The settings for the boot project created in CODESYS are shown on the “CODESYS Configuration” page.

7.8.1.4.1 “General Configuration” Group

Table 37: WBM “CODESYS Configuration” page – “General Configuration” group

Display Fields	Explanation	
CODESYS Version	Select which CODESYS version you want to activate.	
	None	CODESYS is not active.
	2	CODESYS version 2 is active.
(Boot Device)	Select the storage medium the boot project will be stored on.	
	Memory Card	The boot project is stored on the memory card.
	Internal Flash	The boot project is stored in the internal flash memory.

Note

“Internal Flash” is not always available!
The boot project can only be stored in the internal flash memory when the operating system has been started from the internal flash.
If the operating system has been started from the memory card, the “Internal Flash” option is not available.

Click **[Submit]** to apply change. The change is effective immediately.

Note

Restart the controller after changing this setting!
Restart the controller after changing the setting to ensure that any files still open are switched over correctly.

7.8.1.5 “CODESYS Information” Page

All information about the PLC program created in CODESYS is provided on the “CODESYS Information” page.

7.8.1.5.1 “CODESYS” Group

Table 38: WBM “CODESYS – Information” page – “CODESYS” group

Display Fields	Explanation	
Version	The version of the CODESYS runtime system currently active is shown (if the runtime system is disabled, “None” is displayed and the subsequent fields of this group are disabled).	
Version Number	The version number of the CODESYS Web server is shown. This field is only visible when CODESYS is enabled.	
CODESYS State	The operating status of CODESYS is shown. This field is only visible when CODESYS is enabled.	
	STOP	PLC program is not executed.
	RUN	PLC program is executed.
Number of Tasks	The number of tasks in the CODESYS program is shown. This field is only visible when CODESYS is enabled.	

7.8.1.5.2 “Project Details” Group

Table 39: WBM “CODESYS Information” page – “Project Details” group

Display Fields	Explanation
Date	Display of project information that the programmer entered in the PLC program (in CODESYS under Project > Project Information...). The information only appears in an executed PLC program. Descriptive texts up to 1024 characters long are given under “Description.”
Title	
Version	
Author	
Description	

7.8.1.5.3 "Task n" Group(s)

One dedicated group is displayed for each task when the PLC program is executed. As a rule, only the group title is displayed with the task number, the task name and the task ID.

Click **[+]** to expand the group and display the following information.

Table 40: WBM “CODESYS Information” page – “Task n” group(s)

Display Field	**Explanation**
Cycle count	Number of task cycles since system start
Cycle time (µsec)	Currently measured task cycle time
Cycle time min (µsec)	Minimum task cycle time since system start
Cycle time max (µsec)	Maximum task cycle time since system start
Cycle time avg (µsec)	Average task cycle time since system start
Status	Task status (e.g., RUN, STOP)
Mode	Task execution mode (e.g., in cycles)
Priority	Set task priority
Interval (msec)	Set task interval

To hide this information, click **[-]**.

7.8.1.6 “CODESYS WebVisu” Page

The settings for the Web visualization created in CODESYS are shown on the “CODESYS WebVisu” page.

7.8.1.6.1 “Webserver Configuration” Group

Table 41: WBM “CODESYS WebVisu” page – “Webserver Configuration” group

Display Fields	Explanation	
CODESYS Webserver State	The status (enabled/disabled) of the CODESYS Web server is displayed here.	
Default Webserver	Choose whether the Web-based Management or CODESYS Web visualization should be displayed when only entering the IP address of the controller.	
	Web-based Management	The Web-based Management is displayed.
	CODESYS WebVisu	The CODESYS Web visualization is displayed.

Click **[Submit]** to apply change. The change is effective immediately.

In its default setting, the WBM is called up when only entering the IP address.

To update the display after switching, enter the IP address again in the address line of the Web browser.

To display the Web visualization, the CODESYS Web server must be enabled (in WBM under “Ports and Services” -> “CODESYS Services”) and there must be a suitably configured CODESYS application.

Regardless of the default Web browser setting, the WBM can be called up at any time with “http://<IP address>/wbm” and the Web visualization with “http://<IP address>/webvisu”.

You can obtain additional information on CODESYS Web visualization in the chapter with the same name.

Note

Possible error messages when calling up the Web visualization
The “500 – Internal Server Error” message indicates that the CODESYS Web server is not enabled.
A page with the header “WebVisu not available” means that no CODESYS application has been loaded in the controller using Web visualization.

7.8.1.7 “Configuration of Network Parameters” Page"

The settings for the general TCP/IP parameters are found on the “Configuration of Network Parameters” page.

7.8.1.7.1 “Hostname” Group

Table 42: WBM “Configuration of Network Parameters” page – “Hostname” group

Parameters	Explanation
Currently used	If you have selected dynamic assignment of an IP address via DHCP, the name of the host currently being used is displayed.
Configured	Enter the host name of the PC that will be used after a controller restart.

Click **[Submit]** to apply the change. The change becomes effective after the next controller reboot.

7.8.1.7.2 “Domain Name” Group

Table 43: WBM “Configuration of Network Parameters” page – “Domain Name” group

Parameters	Explanation
Domain Name	Set the domain name here.

Click **[Submit]** to apply change. The change is effective immediately.

7.8.1.8 “TCP/IP Configuration” Page

The TCP/IP settings for the ETHERNET interfaces are shown on the “TCP/IP configuration” page.

7.8.1.8.1 “Switch Configuration” Group

Table 44: WBM “TCP/IP Configuration” page – “Switch Configuration” group

Parameters	Explanation	
Interfaces	Enable or disable the switch.	
	Switched	Both interfaces are operated with one IP address.
	Separated	Each interface is operated with its own IP address.

Click **[Submit]** to apply change. The change is effective immediately.

7.8.1.8.2 “IP Address (Xn)” Group(s)

If the switch is enabled, one group (“IP Address”) is shown for both connections. If the switch is disabled, one group (“IP Address X1” / “IP Address X2”) is shown for each connection.

Table 45: WBM “TCP/IP Configuration” page – “IP Address (Xn)” group(s)

Parameters	Explanation	
Configuration Type	Select a static or dynamic IP address.	
	Static IP	Static IP addressing
	DHCP	Dynamic IP addressing
	BootP	Dynamic IP addressing
IP Address	Here, enter a static IP address. This is active if “Static IP” is activated in the **Configuration Type** field.	
Subnet Mask	Enter the subnet mask here. This is active if “Static IP” is activated in the **Configuration Type** field.	

Click **[Submit]** to apply changes. The change is effective immediately.

7.8.1.8.3 “Default Gateway” Group

Table 46: WBM “TCP/IP Configuration” page – “Default Gateway” group

Parameters	Explanation	
State This field is only visible when the switch is enabled.	Specify if you will use the standard gateway. The controller uses the standard gateway when the target address is located outside of its own network.	
	Disabled	The standard gateway is not used.
	Enabled	The standard gateway is used.
Interface This field is only visible when the switch is disabled.	Choose which connection you will use for the standard gateway. The controller uses the standard gateway when the target address is located outside of its own network.	
	None	The standard gateway is not used.
	X1	The standard gateway is used for connection X1.
	X2	The standard gateway is used for connection X2.
Gateway	Enter the address of the standard gateway here.	

Note

Set a maximum of one standard gateway!
If the standard gateway is specified via the DHCP server, a maximum of one standard gateway can be set in “Separated” mode for all combinations (“DHCP”/“Static”, “DHCP"/“DHCP”, etc.).

Click **[Submit]** to apply change. The change is effective immediately.

7.8.1.8.4 “DNS Server” Group

Table 47: WBM “TCP/IP Configuration” page – “DNS Server” group

Parameters	Explanation
DNS Server 1, 2, …	The addresses of the defined DNS servers are displayed here. If no server has been defined, “Configured: None” is displayed.
New server IP	Add additional DNS addresses here.

Click **[Delete]** to remove the selected DNS server. The change is effective immediately.

Click **[Add]** to add the entered DNS server. The change is effective immediately.

7.8.1.9 “Configuration of ETHERNET Parameters” Page

The ETHERNET TCP/IP settings are shown on the “Configuration of ETHERNET” page.

7.8.1.9.1 “Interface Xn” Groups

One group (“Interface X1” / “Interface X2”) is displayed for each connection.

Table 48: WBM “Configuration of ETHERNET Parameters” page – “Interface Xn” groups

Parameters	**Explanation**	
Enabled	You can enable or disable the interface.	
Autonegotiation on	When Autonegotiation is activated, the connection modalities are negotiated automatically with the peer devices.	
Speed/Duplex	Select the transmission speed and the duplex method:	
	10 Mbit half-duplex	Information can only be sent or received.
	100 Mbit half-duplex	
	10 MBit full-duplex	Information can be sent and received simultaneously.
	100 Mbit full-duplex	

Click **[Submit]** to apply changes. The change is effective immediately.

7.8.1.10 “Configuration of Time and Date” Page

The settings for date and time are shown on the “Configuration of Time and Date” page.

7.8.1.10.1 “Date on Device” Group

Table 49: WBM “Configuration of Time and Date” page – “Date on Device” group

Parameters	Explanation
Local	Set date here.

Click **[Change date]** to apply change. The change is effective immediately.

7.8.1.10.2 “Time on Device” Group

Table 50: WBM “Configuration of time and date” page – “Time on Device” group

Parameters	Explanation
Local	Set local time.
UTC	Set GMT time.
12 h format	For switching between 12-hour and 24-hour time display.

Click **[Change time]** to apply change to the time. The change is effective immediately.

Click **[Change format]** to apply change to the time format. The change is effective immediately.

7.8.1.10.3 "Time Zone" Group

Table 51: WBM "Configuration of time and date" page – "Time Zone" group

Parameters	Explanation	
Time Zone	Specify the appropriate time zone for your location. Default setting:	
	AST/ADT	"Atlantic Standard Time", Halifax
	EST/EDT	"Eastern Standard Time", New York, Toronto
	CST/CDT	"Central Standard Time", Chicago, Winnipeg
	MST/MDT	"Mountain Standard Time", Denver, Edmonton
	PST/PDT	"Pacific Standard Time", Los Angeles, Whitehouse:
	GMT/BST	"Greenwich Mean Time", GB, P, IRL, IS, …
	CET/CEST	"Central European Time", B, DK, D, F, I, CRO, NL, …
	EET/EEST	"East European Time", BUL, FI, GR, TR, …
	CST	"China Standard Time"
	JST	"Japan/Korea Standard Time"

Click **[Change]** to apply time zone change. The change is effective immediately.

7.8.1.10.4 "TZ String" Group

Table 52: WBM "Configuration of time and date" page – "TZ String" group

Parameters	Explanation
TZ String	Enter your time zone if it is not available from the "Time Zone" parameter list. An overview of all time zones is available at http://home.tiscali.nl/~t876506/TZworld.html with information about editing the TZ string in Linux® at http://www.minix-vmd.org/pub/Minix-vmd/1.7.0/wwwman/man5/TZ.5.html.

Click **[Change]** to apply the change. The change is effective immediately.

7.8.1.11 “Configuration of the Users for the Web-based Management” Page

The settings for user administration are shown on this page.

7.8.1.11.1 “Change Password for Selected User” Group

Table 53: WBM “Configuration of the users for the Web-based Management” page – “Change Password for selected user” group

Parameters	Explanation
Select User	Select the user (“user” or “admin”) for new password assignment.
New Password	Enter the new password for the user selected under “Select User”. The following ASCII characters for passwords are valid: a … z, A … Z, 0 … 9 and spaces. These special characters are also valid:]!"#$%&'()*+,./:;<=>?@[\^_`{|}~-
Confirm password	Enter the new password again for confirmation.

Click **[Change Password]** to apply change. The change is effective immediately.

Note

Observe the valid characters for WBM passwords!
If WBM passwords with invalid characters are set outside the WBM system (e.g. via CBM), then accessing the WBM pages is no longer possible!

Note

Observe access rights
Authorized WBM users only have access to the Web pages. User administration for controller applications is configured separately.

7.8.1.12 “Create Bootable Image” Page

You can create a bootable image on the “Create Bootable Image” page.

7.8.1.12.1 "Create Bootable Image from Active Partition (<Active Partition>" Group

The active partition that boot-up was performed from is displayed in brackets in the heading.

Table 54: WBM “Create Bootable Image” page – “Create bootable image from active partition” group

<table>
<tr><th>Parameters</th><th colspan="3">Explanation</th></tr>
<tr><td rowspan="4">Destination</td><td colspan="3">The possible destination partition that an image will be saved to is displayed. Depending on which medium has been booted, the following destination is available for selection after boot-up for the image to be generated:</td></tr>
<tr><td>System was booted from</td><td></td><td>Target partition for “bootable image”</td></tr>
<tr><td>Memory Card</td><td>→</td><td>Internal Flash</td></tr>
<tr><td>Internal Flash</td><td>→</td><td>Memory Card</td></tr>
</table>

Once the destination has been determined and output, it is then checked and the results of this check are displayed below the settings:

- Free space on the target device:
  If the available memory is less than 5 %, a warning is displayed. You can still start the copy process despite the warning. If the available space is definitively too low, a corresponding message is displayed and copying cannot be started.

- Device being used by CODESYS:
  If the device is being used by CODESYS a warning is displayed. Although it is not recommended, you can still start the copying procedure despite this warning.

Click **[Start Copy]** to start the copying procedure. If the outcome of the test is positive, copying begins immediately. If errors have been detected, a corresponding message is displayed and copying is not started. If warnings have been issued, these are displayed again and you must then confirm that you still wish to continue.

Note

Remove the memory card write protection!
Because write access to the memory card is possible during the boot process, the memory card cannot be write protected when creating the image and during operation.

7.8.1.13 “Reboot Controller” Page

The settings for the system reboot are shown on the “Reboot Controller” page.

7.8.1.13.1 “Reboot Controller” Group

Click the **[Reboot]** button to reboot the system.

Note

Account for boot-up time!
The boot process takes time. You cannot access the PFC200 while this is occurring.

7.8.1.14 “Firmware Back-up” Page

The settings for a firmware backup are shown on the “Firmware Backup” page.

Select the packages to be restored in the **Packages** group. Select the corresponding entries.

Note

Only one package may be copied to the network!
If you have specified “Network” as the storage location, only one package may be selected for each storing process.

In the selection box **Destination**, select the target storage location.

Note

No backup of the memory card!
Backup from the memory card to the internal flash memory is not possible.

To activate the automatic update function mark check box **Activate “auto update feature**”.

Note

Account for backup time
Generation of backup files can take several minutes. Stop the CODESYS program before you start the backup procedure to help shorten the time required.

To begin the backup procedure, click the **[Submit]** button.

7.8.1.15 “Firmware Restore” Page

The settings for restoring the firmware are shown on the “Firmware Restore” page.

In the **Source** selection box, select the storage location.

Select the packages to be restored in the **Packages** group. Select the corresponding entries.

In the **CODESYS backup file** input field, enter the name of the backup file for the CODESYS project or click the **[Browse]** button to select the file in Explorer.

In the **Settings backup file** input field, enter the name of the backup file for the settings or click the **[Browse]** button to select the file in Explorer.

In the **System backup file** input field, enter the name of the backup file for the system data or click the **[Browse]** button to select the file in Explorer.

Note

Reset by restore
A reset is performed when the system or settings are restored by CODESYS!

To start the restore procedure, click the **[Submit]** button.

7.8.1.16 “System Partition” Page

The settings for specifying the partition that the system will be started from are shown on the “System Partition” page.

7.8.1.16.1 “Current Active Partition” Group

The partition currently in use is displayed here.

7.8.1.16.2 “Set Inactive Partition Active” Group

Click **[Activate Partition]** to start the system from a different partition at the next controller reboot.

Note

Ensure bootable partition!
A functional firmware backup must be present in the boot partition!

Note

Remove the memory card write protection!
Because write access to the memory card is possible during the boot process, the memory card cannot be write protected when operated as the active partition.

7.8.1.17 “Mass Storage” Page

A group containing information about the storage volume is displayed for each storage volume that is found, along with an additional group for formatting (when this is possible).

The group title contains the designation for the storage volume (“SD card” or “Internal Flash”) and, if this storage volume is also the active partition, the text “Active Partition”.

7.8.1.17.1 “<Device Name>” Group(s)

Table 55: WBM “Mass Storage” page – “<Device Name>” group

Parameters	Explanation
Device	The name of the storage volume in the operating system file system is displayed here.
Volume name	The name of the storage volume is displayed here.

7.8.1.17.2 “<Device Name> - FAT Format” Group(s)

Table 56: WBM “Mass Storage” page – “<Device Name>” group

Parameters	Explanation
Volume Name	Specify the name for the storage volume when formatted.

Note

Data are deleted!
Any data stored in the storage volume is deleted during formatting!

To format the specified storage volume, click **[Start Formatting]**.

7.8.1.18 “Software Uploads” Page

The settings for a device update are shown on the “Software uploads” page.

7.8.1.18.1 “Upload New Software” Group

Table 57: WBM “Software Uploads” page – “Upload New Software” group

Parameters	Explanation
(Package File)	You can select fieldbus software, program licenses and update scripts, for example, for transfer from a PC to the controller.

To select a file on the PC, click the **[Browse]** button.

To transfer the selected file to the controller, click **[Start Upload]** button.

7.8.1.18.2 “Activate New Software” Group

Table 58: WBM “Software Uploads” page – “Activate New Software” group

Parameters	Explanation
(Package File)	The file name of the software package to be transferred is displayed here, with a button on the right for activation. If no new, uploaded software package is available on the controller, the message “No upload file exists” is displayed.

To activate the uploaded software package, click the **[Activate]** button. The process starts immediately.
The file with the software package is deleted on completion of the installation procedure.

7.8.1.19 “Configuration of Network Services” Page

The settings for various services are shown on the “Configuration of Network Services” page.

7.8.1.19.1 “Telnet” Group

Table 59: WBM “Configuration of Network Services” page – “Telnet” group

Parameters	Explanation
enabled/disabled	Indicates if the Telnet service is enabled.

Click **[Disable]**/**[Enable]** to change the status of the service. The change is effective immediately.

7.8.1.19.2 “FTP” Group

Table 60: WBM “Configuration of Network Services” page – “FTP” group

Parameters	Explanation
enabled/disabled	Indicates if the FTP service is enabled.

Click **[Disable]**/**[Enable]** to change the status of the service. The change is effective immediately.

7.8.1.19.3 “FTPS” Group

Table 61: WBM “Configuration of Network Services” page – “FTPS” group

Parameters	Explanation
enabled/disabled	It is displayed if the FTPS service is enabled.

Click **[Disable]**/**[Enable]** to change the status of the service. The change is effective immediately.

7.8.1.19.4 “HTTP” Group

Table 62: WBM “Configuration of Network Services” page – “HTTP” group

Parameters	Explanation
enabled/disabled	It is displayed if the HTTP service is enabled.

Click **[Disable]**/**[Enable]** to change the status of the service. The change is effective immediately.

7.8.1.19.5 “HTTPS” Group

Table 63: WBM “Configuration of Network Services” page – “HTTPS” group

Parameters	Explanation
enabled/disabled	It is displayed if the HTTPS service is enabled.

Click **[Disable]**/**[Enable]** to change the status of the service. The change is effective immediately.

7.8.1.20 “Configuration of NTP Client” Page

The settings for the NTP service are shown on the “Configuration of NTP Client” page.

7.8.1.20.1 “NTP Client” Group

Table 64: WBM “Configuration of NTP Client” page – “NTP Client” group

Parameters	**Explanation**
Service enabled	Activate/deactivate time update.
Port	Enter the port number for the NTP access (default setting: 123)
Time Server	Enter the IP address of the time server.
Update Time (sec)	Specify the time server query cycle.

Click the **[Submit]** button to apply the changes. The change is effective immediately.

7.8.1.21 “Configuration of the CODESYS Services” Page

The settings for various CODESYS services are shown on the “Configuration of the CODESYS Services” page.

7.8.1.21.1 “CODESYS Webserver” Group

Table 65: WBM “Configuration of the CODESYS Services” page – “CODESYS Webserver” group

Parameters	Explanation
Current State	The CODESYS Web server status is displayed.

Click **[Disable]**/**[Enable]** to change the status of the service. The change is effective immediately.

7.8.1.21.2 “Communication” Group

Table 66: WBM “Configuration of the CODESYS Services” page – “Communication” group

Parameters	Explanation
Current State	The CODESYS status is displayed.
Port Number	The CODESYS port number is displayed.

Click **[Disable]**/**[Enable]** to change the status of the service. The change is effective immediately.

Click **[Change]** to change the port number. The change is effective immediately.

7.8.1.21.3 “Port Authentication” Group

Table 67: WBM “Configuration of the CODESYS Services” page – “Port Authentication” group

Parameters	Explanation
Current State	Displays if port authentication is activated. If authentication is “enabled”, a password is required to access the port.

Click **[Disable]**/**[Enable]** to activate/deactivate port authentication. The change is effective immediately.

7.8.1.21.4 “Port Authentication Password” Group

Table 68: WBM “Configuration of the CODESYS Services” page – “Port Authentication Password” group

Parameters	Explanation
New Password	Specify the password for port authentication.
Confirm password	Enter the new password again for confirmation.

Click **[Change]** to change the password. The change is effective immediately.

7.8.1.22 “SSH Client Settings” Page

The settings for the SSH service are shown on the “SSH Client Settings” page.

7.8.1.22.1 “SSH Client” Group

Table 69: WBM “SSH Client Settings” page – “SSH Client” group

Parameters	Explanation
Current state	You can enable/disable the SSH client.
Port Number	Specify the port number.
Allow root login	You can enable or inhibit root access.
Allow password login	Activate or deactivate the password query function here.

Click **[Disable]**/**[Enable]** to change the status of the service. The change is effective immediately.

Click the **[Submit]** button to apply the other changes. The change is effective immediately.

7.8.1.23 “TFTP Server” Page

The settings for the TFTP service are shown on the “TFTP Server” page.

7.8.1.23.1 “TFTP Server” Group

Table 70: WBM “TFTP Server” page – “TFTP Server” group

Parameters	Explanation
Current state	Activate or deactivate the TFTP server.
Download directory	Specify the path for downloading the server directory.

Click **[Disable]**/**[Enable]** to change the status of the service. The change is effective immediately.

Click the **[Submit]** button to apply the other changes. The change is effective immediately.

7.8.1.24 “Configuration of SNMP parameter” Page

The general settings for SNMP are shown on the “Configuration of SNMP parameter” page.

7.8.1.24.1 “General SNMP Configuration” Group

Table 71: WBM “Configuration of SNMP parameter” page – “General SNMP Configuration” group

Parameters	**Explanation**
Current state: enabled/disabled	Here, the actual state of the SNMP service is displayed.
Name of device	Enter the device name (sysName) here.
Description	Enter the device description (sysDescription) here.
Physical location	Enter the device location (sysLocation) here.
Contact	Enter the email contact address (sysContact) here.

Click **[Disable/Enable]** to change the stateof the SNMP service. Click **[Change]** to apply any further changes. The changes do not become effective until after the next software or hardware reset.

7.8.1.25 "Configuration of SNMP Parameter" Page

The general settings for SNMP v1/v2c are shown on the "Configuration of SNMP parameter" page.

7.8.1.25.1 "SNMP v1/v2c Manager Configuration" Group

Table 72: WBM "Configuration of SNMP parameter" page – "SNMP v1/v2c Manager Configuration" group

Parameters	Explanation
Protocol enabled	It is displayed the SNMP protocol for v1/v2c is activated. The local community name is deleted when the protocol is deactivated.
Local Community Name	Here, specify the community name for the SNMP manager configuration. The community name can establish relationships between SNMP managers and agents who are respectively referred to as "Community" and who control identification and access between SNMP participants. The community name can be up to 32 characters long and may not include spaces. To use the SNMP protocol, a valid community name must always be specified.

Click **[Change]** to apply changes. The changes do not become effective until after the next software or hardware reset.

7.8.1.25.2 "Actually Configured Trap Receivers" Group(s)

Table 73: WBM "Configuration of SNMP parameter" page – "Actually Configured Trap Receivers" group

Parameters	Explanation
Count	The number of configured trap receivers is displayed.

7.8.1.25.3 “Trap Receiver n” Group(s)

A dedicated group with the following information is displayed for each trap receiver:

Table 74: WBM “Configuration of SNMP parameter” page – “Trap Receiver n” group

Parameters	Explanation
IP Address	The IP address of the trap receiver (management station) is displayed.
Community Name	The community name for the trap receiver configuration is displayed here. The community name can be evaluated by the trap receiver.
Version	The SNMP version, via which the traps are sent, is displayed here: v1 or v2c (traps higher than v3 are displayed in a separate form).

Click **[Delete]** to delete the trap receiver. The changes do not become effective until after the next software or hardware reset.

7.8.1.25.4 “Add new Trap Receiver” Group

Table 75: WBM “Configuration of SNMP parameter” page – “Add new Trap Receiver” group

Parameters	Explanation
IP Address	Enter the IP address of the new trap receiver (management station).
Community Name	Here, specify the community name for the new trap receiver configuration. The community name can be evaluated by the trap receiver. The community name can be up to 32 characters long and may not include spaces.
Version	Specify the SNMP version that will send the traps: v1 or v2c (traps higher than v3 are configured in a separate form).

Click **[Add]** to add a new trap receiver. The changes do not become effective until after the next software or hardware reset.

7.8.1.26 “Configuration of SNMP v3 Users” Page

The general settings for SNMP v3 are shown on the “Configuration of SNMP v3 Users” page.

7.8.1.26.1 “Actually Configured v3 Users” Group(s)

Table 76: WBM “Configuration of SNMP v3” page – “Actually Configured v3 Users” group

Parameters	Explanation
Count	The number of configured v3 users is displayed.

7.8.1.26.2 “v3 User n” Group(s)

A group with the following information is displayed for each user:

Table 77: WBM “Configuration of SNMP v3 Users” page – “v3 User n” group(s)

Parameters	Explanation
Security Authentication Name	The user name is displayed.
Authentication Type	The authentication type for the SNMP v3 packets is displayed here. Possible values: - Use no authentication (“None”) - Message Digest 5 (“MD5”) - Secure Hash Algorithm (“SHA”)
Authentication Key (min. eight char.)	The authentication key is displayed.
Privacy	The encryption algorithm for the SNMP message is displayed here. Possible values: - No encryption (“None”) - Data Encryption Standard (“DES”) - Advanced Encryption Standard (“AES”)
Privacy Key (min. eight char.)	The key for encryption of the SNMP message is displayed here. If nothing is displayed here, the “authentication key” is automatically used.
Notification Receiver IP	The IP address of a trap receiver for v3 traps is displayed here. If no v3 traps are to be sent for this user, this field remains blank.

Click **[Delete]** to delete the user. The changes do not become effective until after the next software or hardware reset.

7.8.1.26.3 “Add New v3 User” Group

Table 78: WBM “Configuration of SNMP v3 Users” page – “Add new v3 User” group

Parameters	Explanation
Security Authentication Name	Enter the user name here. This name must be unique; a pre-existing user name is not accepted when entered here. The security authentication name can have a maximum 32 characters, without any spaces.
Authentication Type	Specify the authentication type for the SNMP v3 packets. Possible values: - Use no authentication (“None”) - Message Digest 5 (“MD5”) - Secure Hash Algorithm (“SHA”)
Authentication Key (min. eight char.)	Specify the authentication key here. This authentication key must have between eight and 32 characters, without any spaces.
Privacy	Specify the encryption algorithm for the SNMP message here. Possible values: - No encryption (“None”) - Data Encryption Standard (“DES”) - Advanced Encryption Standard (“AES”)
Privacy Key (min. eight char.)	Enter the key for encryption of the SNMP message here. If nothing is specified here, the “authentication key” is automatically used. The privacy key must have between eight and 32 characters, without any spaces.
Notification Receiver IP	Specify an IP address for a trap receiver for v3 traps here. If no v3 traps are to be sent for this user, this field remains blank.

Click **[Add]** to add a new user. The changes do not become effective until after the next software or hardware reset.

7.8.1.27 “Diagnostic Information” Page

The settings for displaying diagnostics messages are shown on the “Diagnostic Information” page.

Table 79: WBM “Diagnostic Information” page

Parameters	Explanation
Read all notifications	Activate display of all messages.
Read only the last n	Activate display of only the last n messages. You also specify the number of messages to be displayed.
Automatic refresh cycle (sec)	Select the check box to enable cyclic refresh. Enter the cycle time in seconds when a cyclic refresh is performed. If cyclic refresh is enabled, the button label changes to “Stop”.

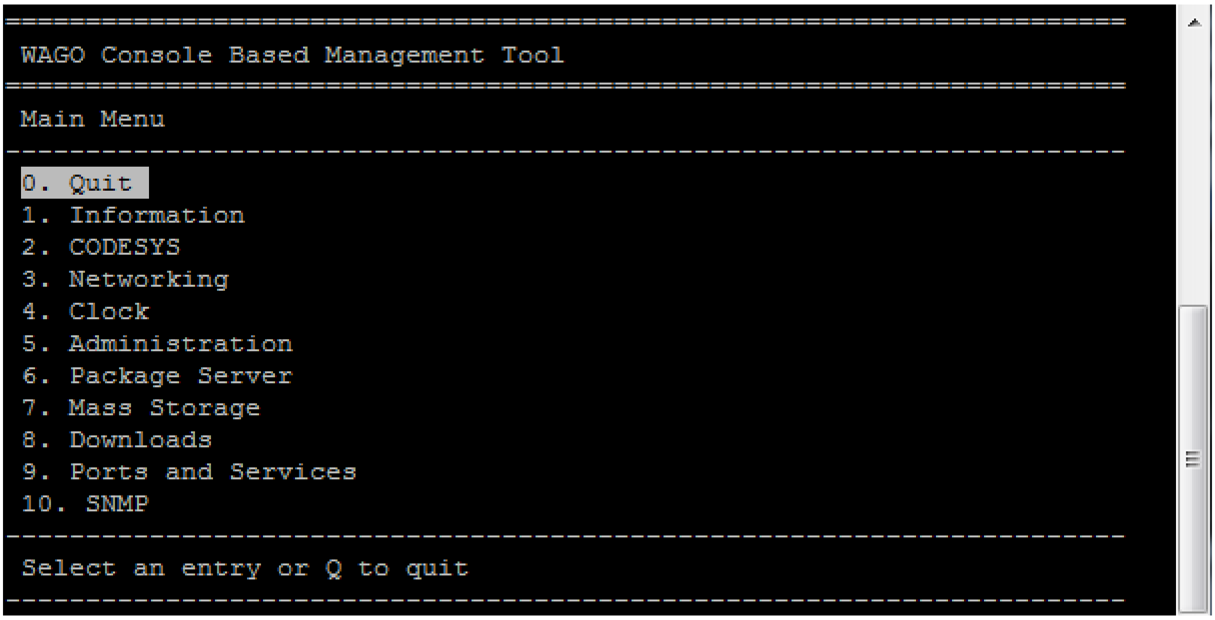
To refresh the display or to enable cyclic refresh, click the **[Refresh]** button. The button is only visible if cyclic refresh is disabled.

To stop cyclic refresh again, click the **[Stop]** button. The button is only visible if cyclic refresh is enabled.

The messages are displayed below the settings.

7.8.2 Configuration using a Terminal Program (CBM)

You can configure the PFC200 via ETHERNET using ssh via the CBM. To call up the CBM log in to the Linux® console and enter the command “cbm” (case-sensitive!).



```
======================================================================
 WAGO Console Based Management Tool
======================================================================
 Main Menu
----------------------------------------------------------------------
 0. Quit
 1. Information
 2. CODESYS
 3. Networking
 4. Clock
 5. Administration
 6. Package Server
 7. Mass Storage
 8. Downloads
 9. Ports and Services
 10. SNMP
----------------------------------------------------------------------
 Select an entry or Q to quit
----------------------------------------------------------------------
```

Figure 3: Accessing the CBM using ssh (example)

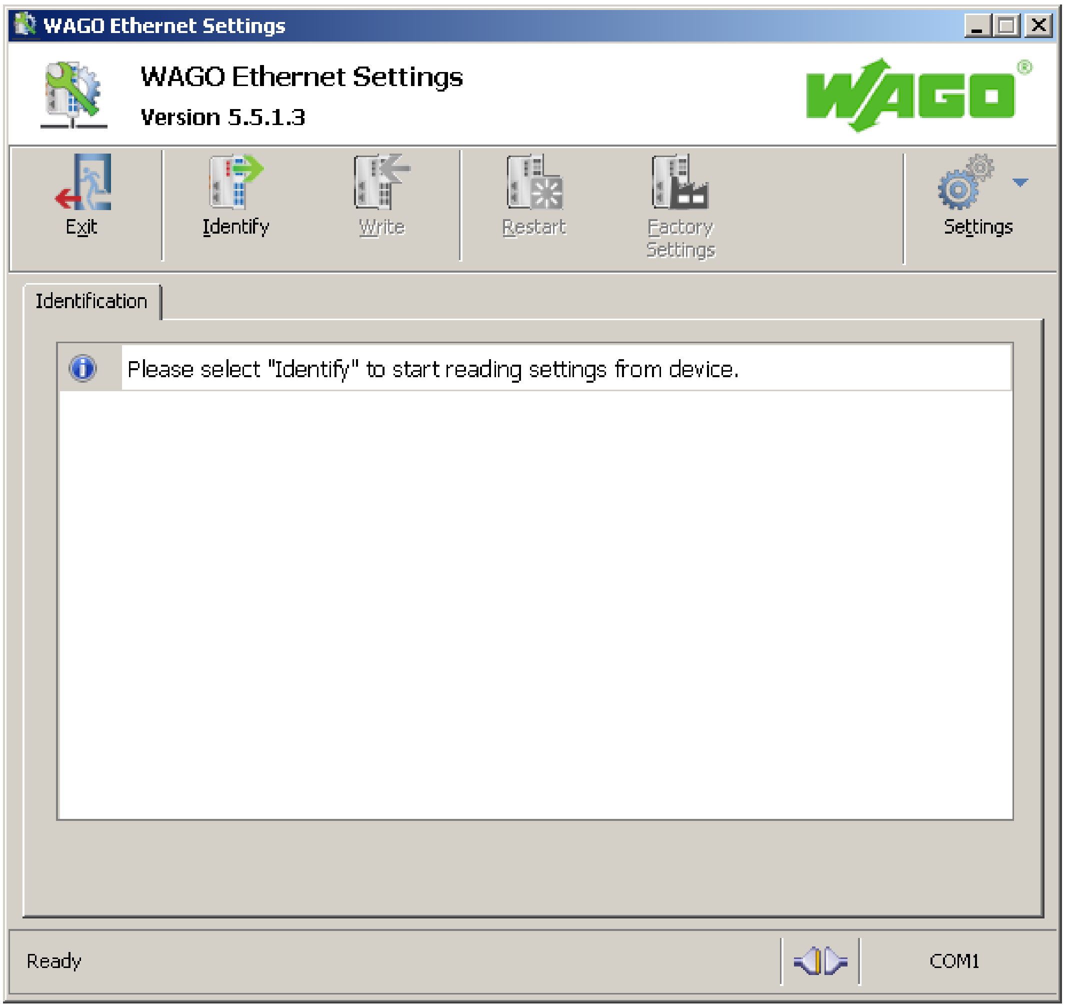
7.8.3 Configuration using “WAGO ETHERNET Settings”

The “WAGO ETHERNET Settings” program enables you to read system information about your PFC200, make network settings and enable/disable the Web server.

Note

Observe the software version!
To configure the controller, use at least Version 5.4.2.3 dated July 30, 2013 or newer in “WAGO ETHERNET Settings”!

You must select the correct COM port after starting “WAGO ETHERNET Settings”.

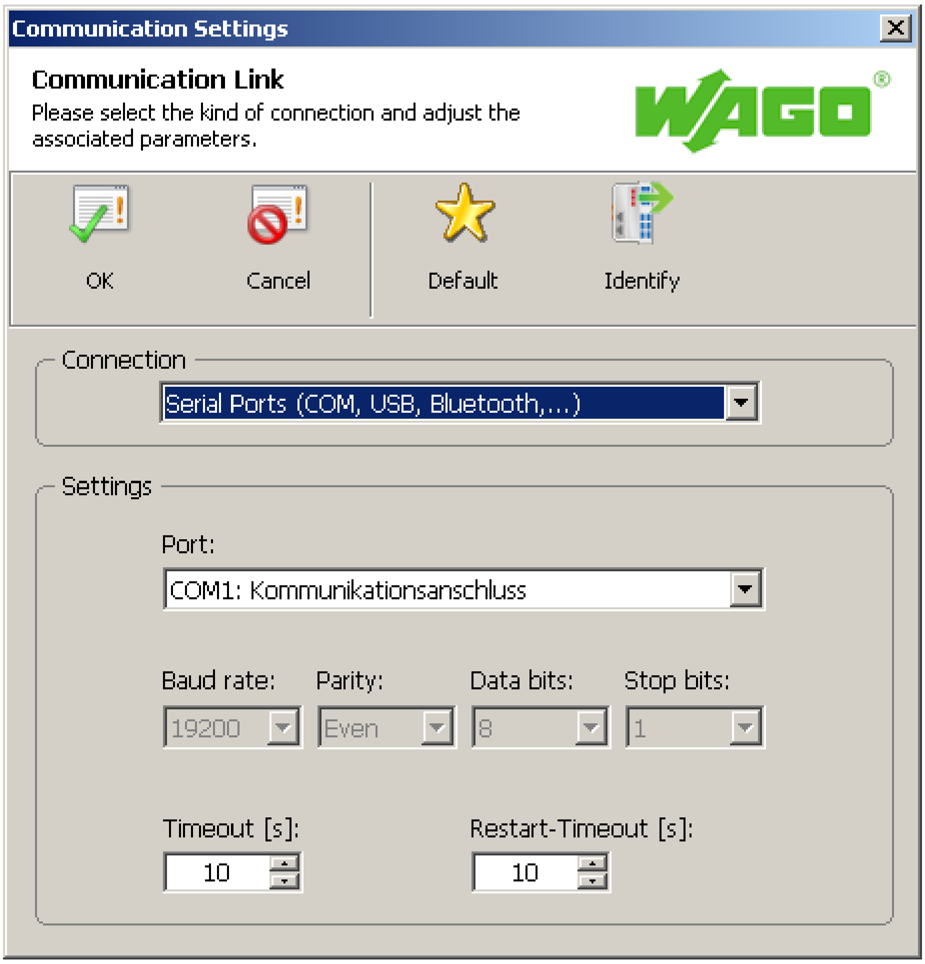


Please select "Identify" to start reading settings from device.

Figure 31: “WAGO ETHERNET Settings” – Start screen

For this, click “Settings” and then “Communication”.

In the “Communication settings” window that then opens, adapt the settings to your needs.


Communication Settings

Communication Link

Please select the kind of connection and adjust the associated parameters.

Connection: Serial Ports (COM, USB, Bluetooth,...)

Settings

Port: COM1: Kommunikationsanschluss

Baud rate: 19200

Parity: Even

Data bits: 8

Stop bits: 1

Timeout [s]: 10

Restart-Timeout [s]: 10

Figure 32: “WAGO ETHERNET Settings” – Communication link

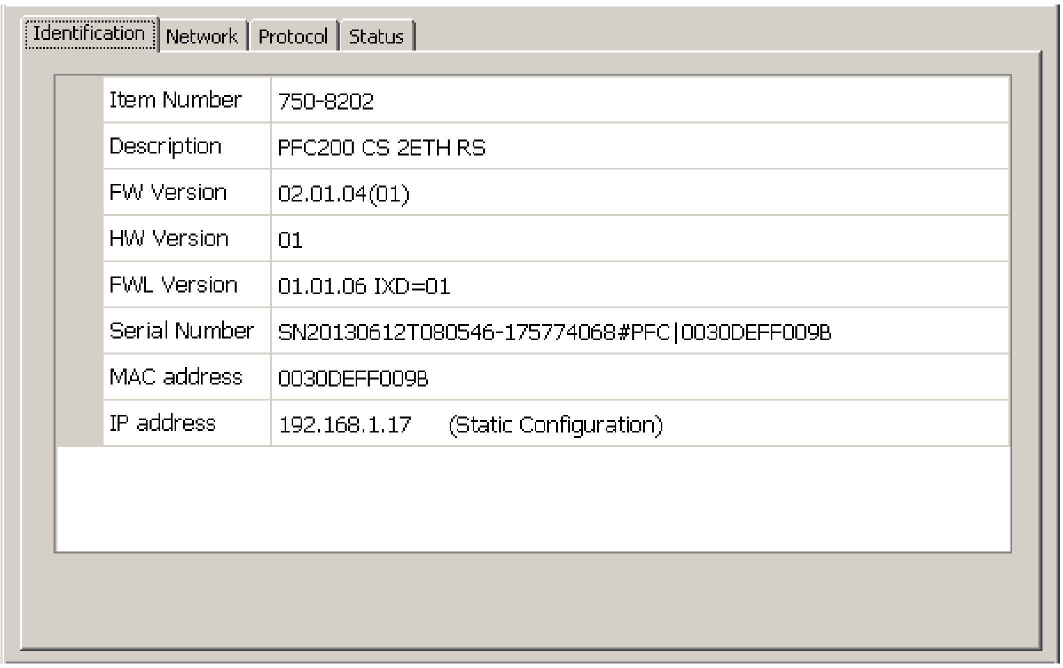
Once you have configured “WAGO ETHERNET Settings” and have clicked **[OK]**, connection to the PFC200 is established automatically.

If “WAGO ETHERNET Settings” has already been started with the correct parameters, you can establish connection to the PFC200 by clicking **[Identify]**.

7.8.3.1 Identification Tab

An overview of the connected device is given here.

Besides some fixed values — e.g., item No., MAC address and firmware version — the currently used IP address and the configuration method are also shown here.



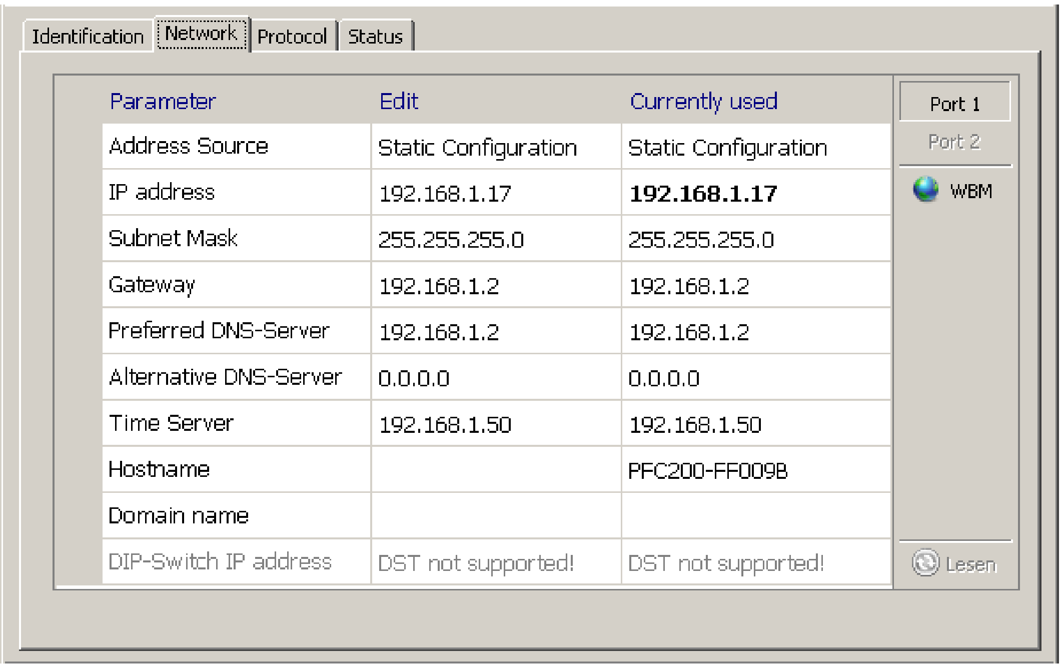
Item Number	750-8202
Description	PFC200 CS 2ETH RS
FW Version	02.01.04(01)
HW Version	01
FWL Version	01.01.06 IXD=01
Serial Number	SN20130612T080546-175774068#PFC|0030DEFF009B
MAC address	0030DEFF009B
IP address	192.168.1.17 (Static Configuration)

Figure 33: “WAGO ETHERNET Settings” – Identification tab (example)

7.8.3.2 Network Tab

This tab is used to configure network settings.

Values can be changed in the “Input” column, while the parameters in use are shown in the “Currently in use” column.



Parameter	Edit	Currently used
Address Source	Static Configuration	Static Configuration
IP address	192.168.1.17	**192.168.1.17**
Subnet Mask	255.255.255.0	255.255.255.0
Gateway	192.168.1.2	192.168.1.2
Preferred DNS-Server	192.168.1.2	192.168.1.2
Alternative DNS-Server	0.0.0.0	0.0.0.0
Time Server	192.168.1.50	192.168.1.50
Hostname		PFC200-FF009B
Domain name		
DIP-Switch IP address	DST not supported!	DST not supported!

Figure 34: “WAGO ETHERNET Settings” – Network tab

Source
Specify how the PFC200 will determine its IP address: Static, via DHCP or via BootP.

IP address, subnet mask, gateway
Specify the specific network parameters for static configuration.

Preferred DNS server, alternative DNS server
Enter the IP address (when required) for an accessible DNS server when identifying network names.

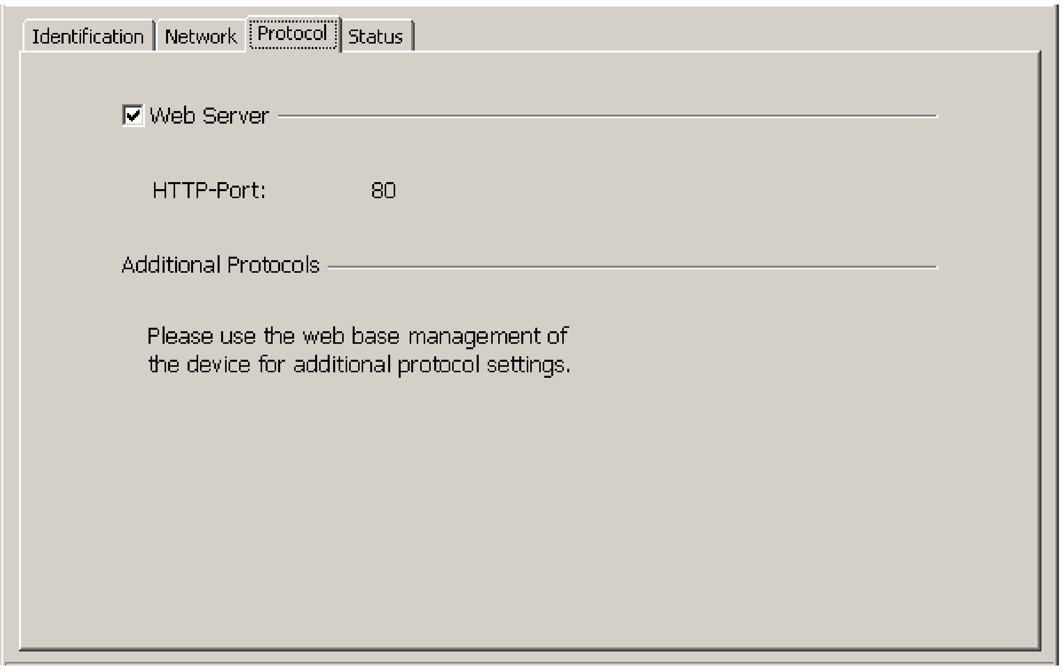
Time server
Specify the IP address for a time server if setting the PFC200's system time via NTP.

Host name
The host name of the PFC200 is displayed here. In the controller's initial state, this name is composed of the string “PFC200” and the last three bytes of the MAC address.
This standard value is also used whenever the chosen name in the “Input” column is deleted.

Domain name

The current domain name is displayed here. This setting can be automatically overwritten with dynamic configurations, e.g., DHCP.

7.8.3.3 Protocol Tab



☑ Web Server

HTTP-Port: 80

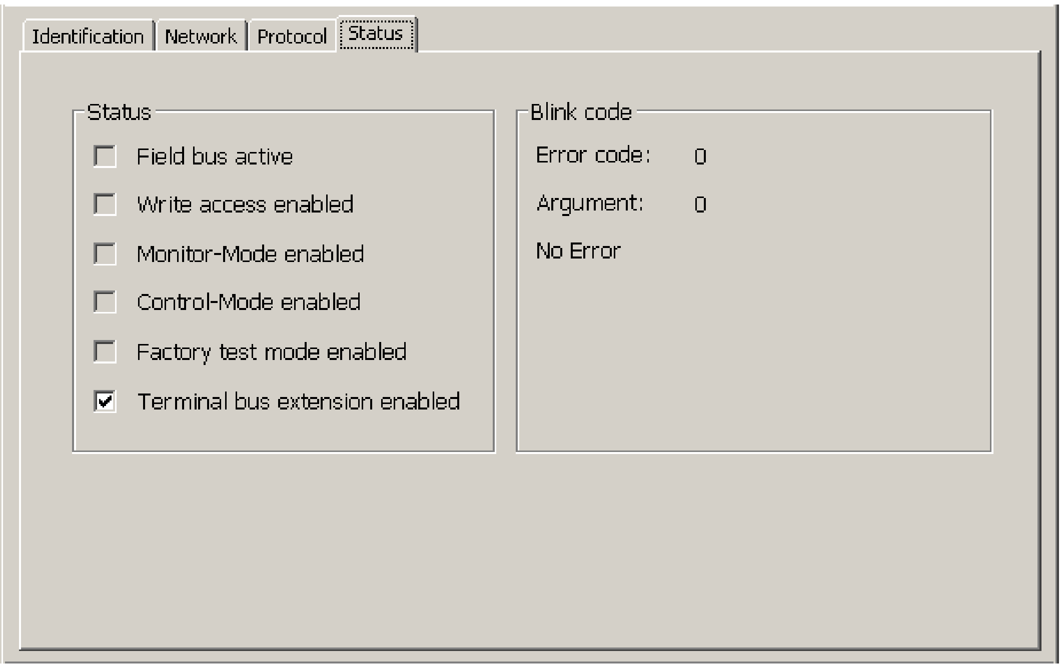
Additional Protocols

Please use the web base management of the device for additional protocol settings.

Figure 35: “WAGO ETHERNET Settings” – Protocol tab

You can enable or disable the Web server.

7.8.3.4 Status Tab



Status

□ Field bus active

□ Write access enabled

□ Monitor-Mode enabled

□ Control-Mode enabled

□ Factory test mode enabled

☑ Terminal bus extension enabled

Blink code

Error code: 0

Argument: 0

No Error

Figure 36: “WAGO ETHERNET Settings” – Status tab

General information about the device status is displayed here.

The **Bus extension** check box has no function for the PFC200, i.e., the bus extension is always active.

8 Run-time System CODESYS 2.3

8.1 Installing the CODESYS 2.3 Programming System

The WAGO target files must also be included for the installation of CODESYS. These contain all device-specific information for the WAGO 750/758 product series.

Proceed as described below to install the CODESYS 2.3 programming software on the fieldbus controller.

1. Insert the “WAGO-I/O-*PRO*” CD into your computer 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.

8.2 First Program with CODESYS 2.3

This section uses an example to explain the relevant steps required for the creation of a CODESYS project. It is intended as a set of quick start instructions and does not address the full functional range of CODESYS 2.3.

Note

Additional information

For a detailed description of the full range of functions, refer to the “Manual for PLC Programming using CODESYS 2.3” manual available on the “WAGO-I/O-*PRO*” (759-911) CD.

8.2.1 Start the CODESYS Programming System

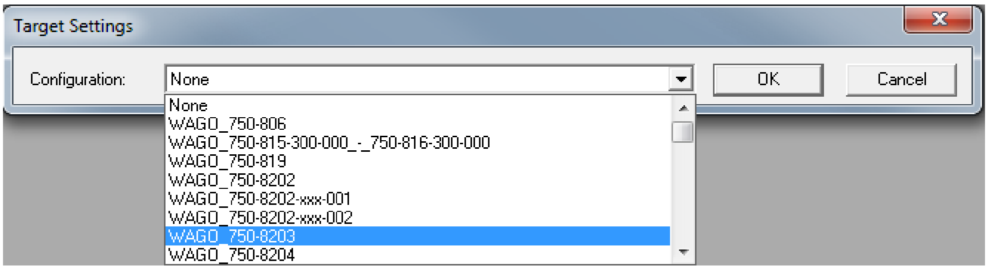
Start CODESYS by double clicking on the CODESYS pictogram on your desktop using the Start menu in your operating system. To do this, click on the “Start” button and choose **Programs** >
WAGO Software > **CODESYS** > **CODESYS V2.3**.

8.2.2 Creating a Project and Selecting the Target System

1. In the menu bar click on **File** and select **New**. The “Target system settings” window then opens. Here, all available target systems that can be programmed with CODESYS 2.3 are listed.

2. Open the selection box in the “Target system settings” window and select the fieldbus controller you are using. In the example shown here this is the PFC200 CS 2ETH CAN “WAGO_750-8203”.

3. Click on **[OK]**. The “Target system settings” configuration window then opens.

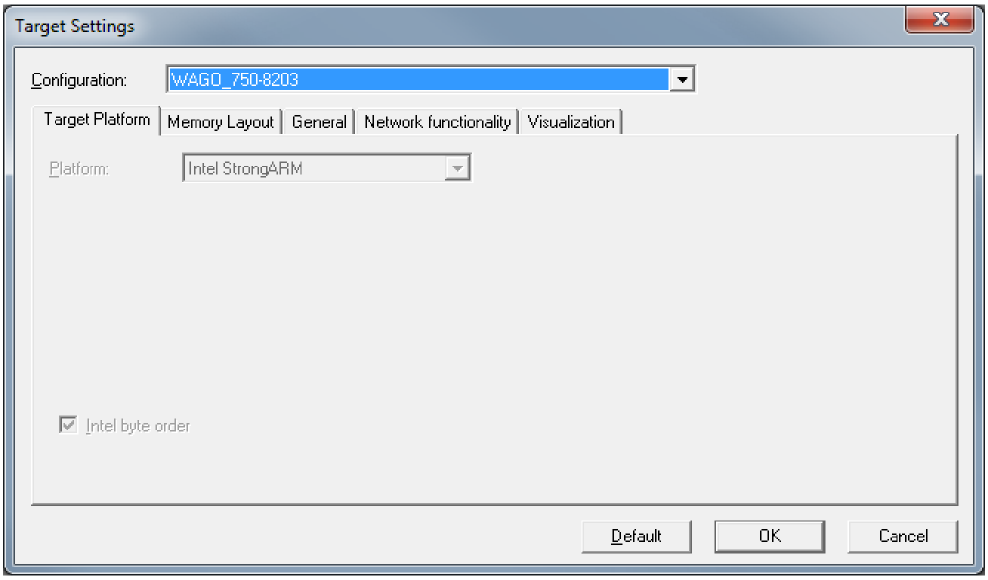


Target Settings

Configuration: None

Figure 37: Target system settings (1)

4. To accept the default configuration for the fieldbus controller click **[OK]**. The "New component" window opens.



Target Settings

Configuration: WAGO_750-8203

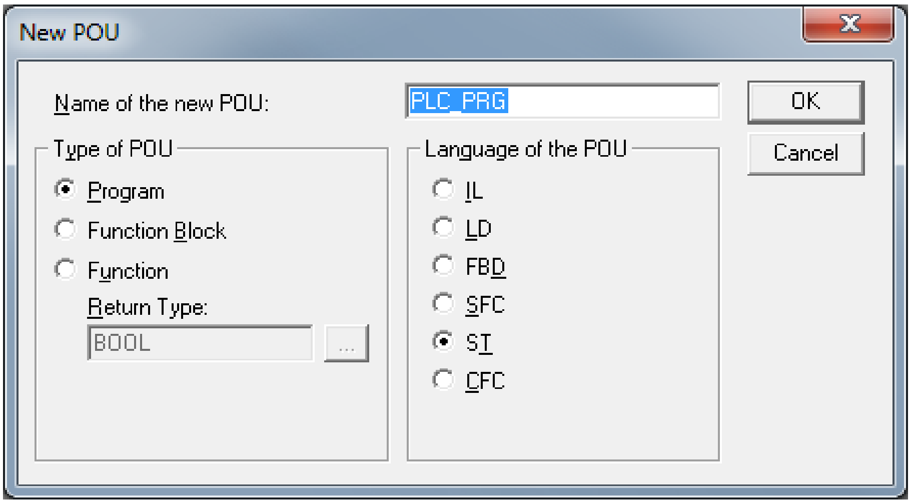
Platform: Intel StrongARM

☑ Intel byte order

Figure 38: Target system settings (2)

5. In this “New component” window create a new program function block. In the example shown here, the new function block “PLC_PRG” is created in the “ST” programming language.

6. Click on **[OK]** to create the project. The programming interface opens.



New POU

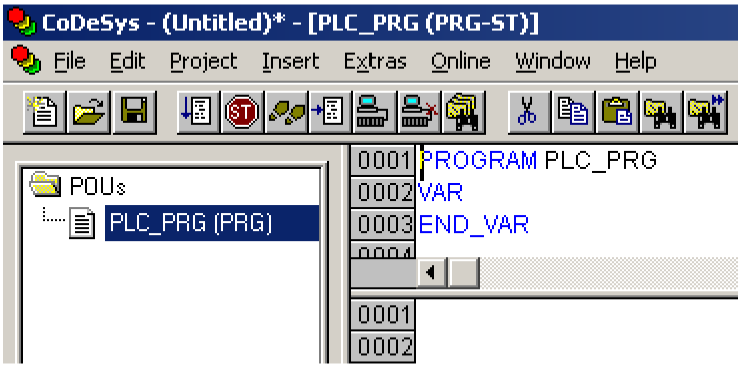
Name of the new POU: PLC_PRG

Type of POU
- ● Program
- ○ Function Block
- ○ Function

Return Type: BOOL

Language of the POU
- ○ IL
- ○ LD
- ○ FBD
- ○ SFC
- ● ST
- ○ CFC

Figure 39: Creating a new function block



CoDeSys - (Untitled)* - [PLC_PRG (PRG-ST)]

```
0001 PROGRAM PLC_PRG
0002 VAR
0003 END_VAR
```

Figure 40: Programming interface with the PLC_PRG program module

8.2.3 Creating the PLC Configuration

Note

Procedure for Creating the PLC Configuration

The procedure explained in this section describes the PLC configuration for the I/O modules connected to the controller.
Information about the controller function for any fieldbuses connected to the system is given in the section on the specific fieldbus.

The PLC configuration is used to configure the fieldbus controller, along with the connected I/O modules and to declare variables for accessing the inputs and outputs of the I/O modules.

1. Click on the “Resources” tab.

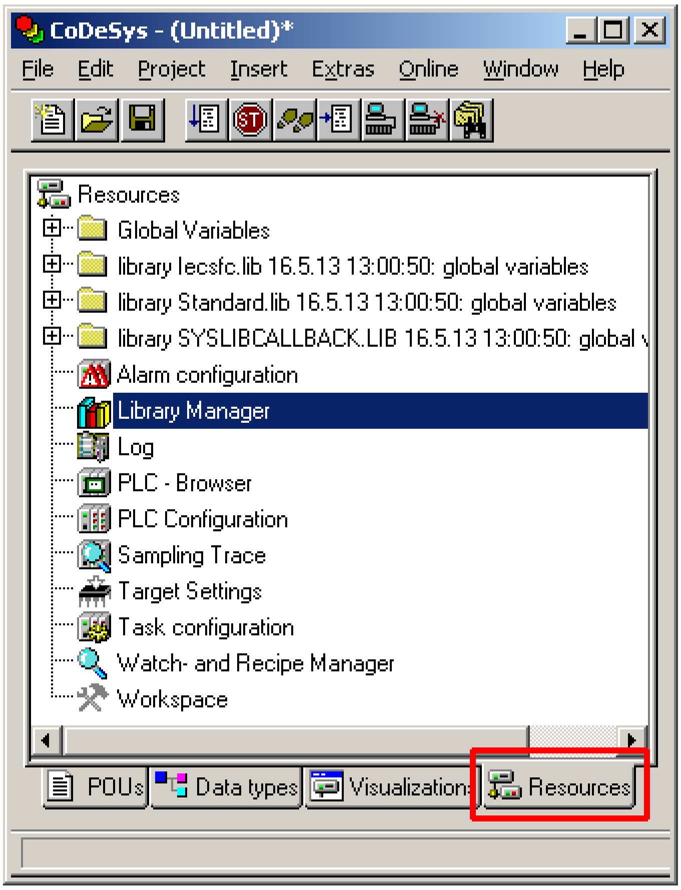

CoDeSys - (Untitled)*
File Edit Project Insert Extras Online Window Help
Resources
Global Variables
library Iecsfc.lib 16.5.13 13:00:50: global variables
library Standard.lib 16.5.13 13:00:50: global variables
library SYSLIBCALLBACK.LIB 16.5.13 13:00:50: global v
Alarm configuration
Library Manager
Log
PLC - Browser
PLC Configuration
Sampling Trace
Target Settings
Task configuration
Watch- and Recipe Manager
Workspace
POUs
Data types
Visualization
Resources

Figure 41: “Resources” tab

2. In the left window double-click on “PLC configuration”. The PLC configuration for the controller opens.

3. Right-click on the entry “K-Bus[FIX]” and then select “Edit” in the contextual menu. The “configuration” dialog window then opens.

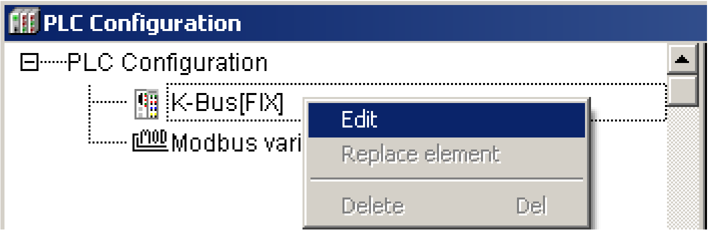


Figure 42: Control configuration – Edit

4. There are three options for accepting the topology for the I/O modules connected to the fieldbus controller. The simplest way is to scan in the topology using WAGO-I/O-*CHECK*.
To do this, click on the “Start WAGO-I/O-*CHECK* and scan” button.

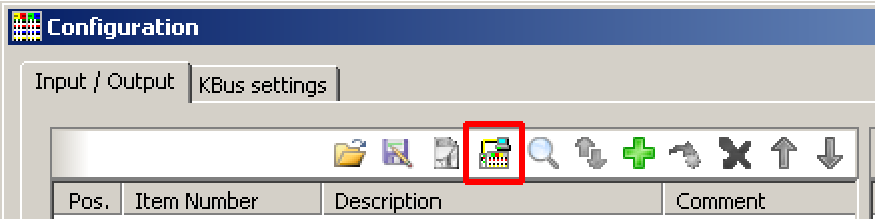


Figure 43: “Start WAGO-I/O-*CHECK* and scan” button

Note

Ensure proper installation of WAGO-I/O-*CHECK*!

This function requires that the latest version of WAGO-I/O-*CHECK* be installed and the IP address set under “Online > Communication parameters”, as otherwise communication will not be possible.

5. WAGO-I/O-*CHECK* is started.

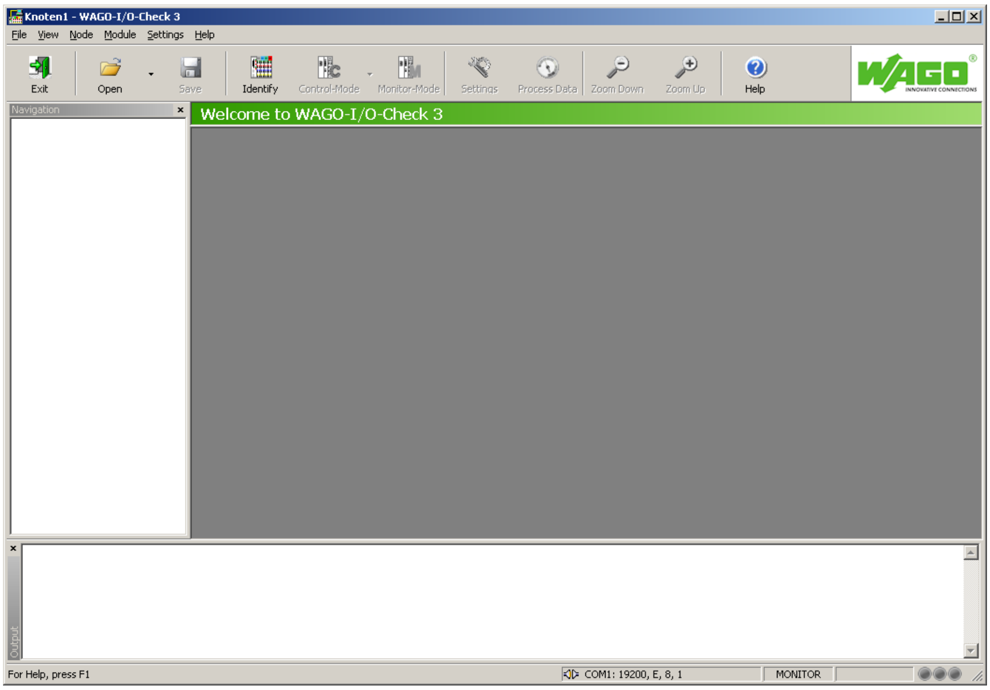


Figure 44: WAGO-*I/O-CHECK* – Starting screen

6. To connect to the controller and read in the module configuration, click **[Identify]**.

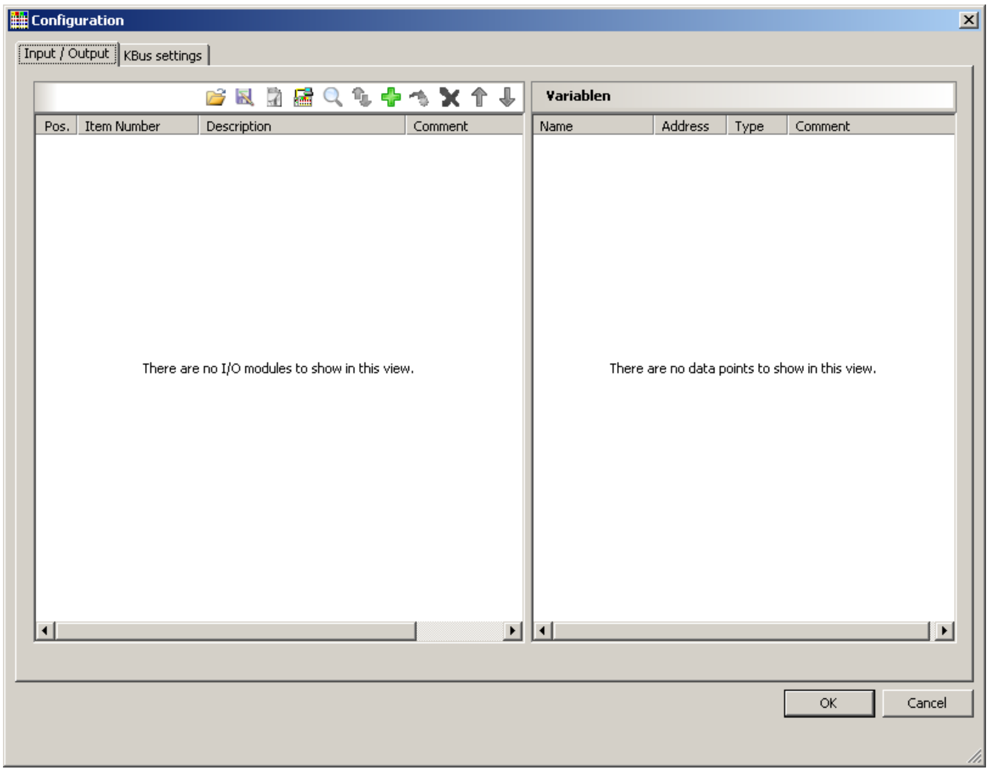
7. If this action is successful click **[Save]** and exit WAGO-I/O-*CHECK*.

8. The detected I/O modules then appear in the configuration window.

Note

Passive I/O Modules

Remember that passive I/O modules, such as a power supply module (750-602) or end module (750-600) will not be shown in the I/O configurator.



There are no I/O modules to show in this view.

There are no data points to show in this view.

Figure 45: I/O Configurator empty

9. You can use the **[Add]** button to add new I/O modules to manually define or change the configuration.

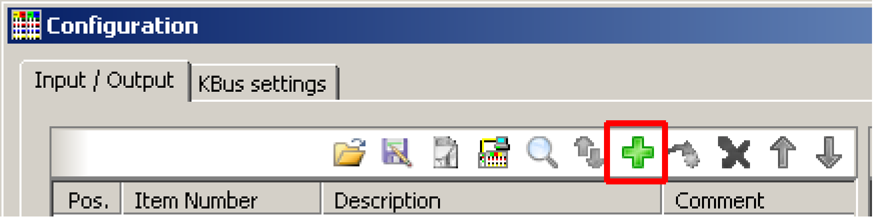


Figure 46: “Add I/O modules” button

10. You can select a module in the new “Module selection” window that then appears.

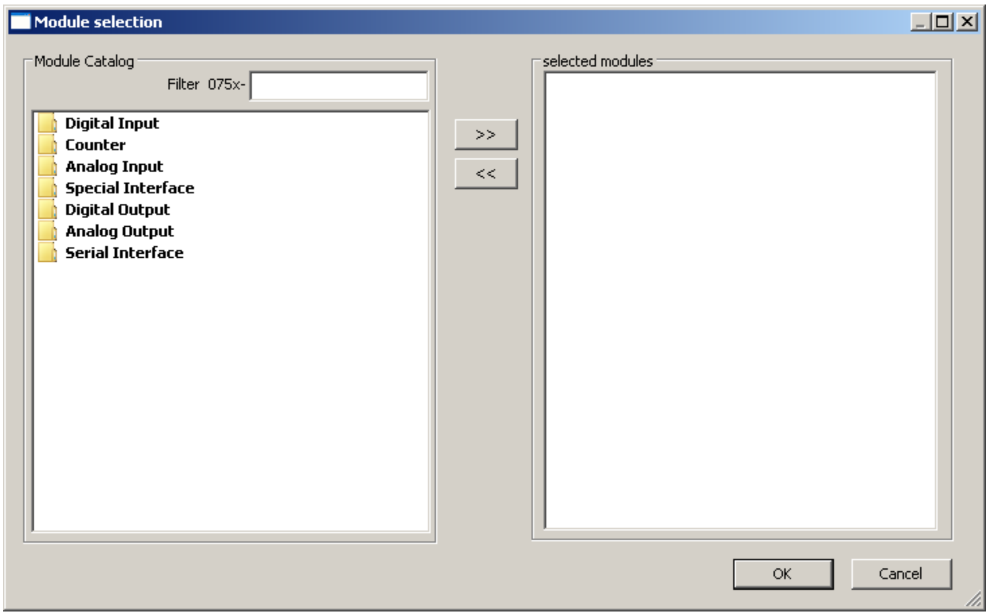
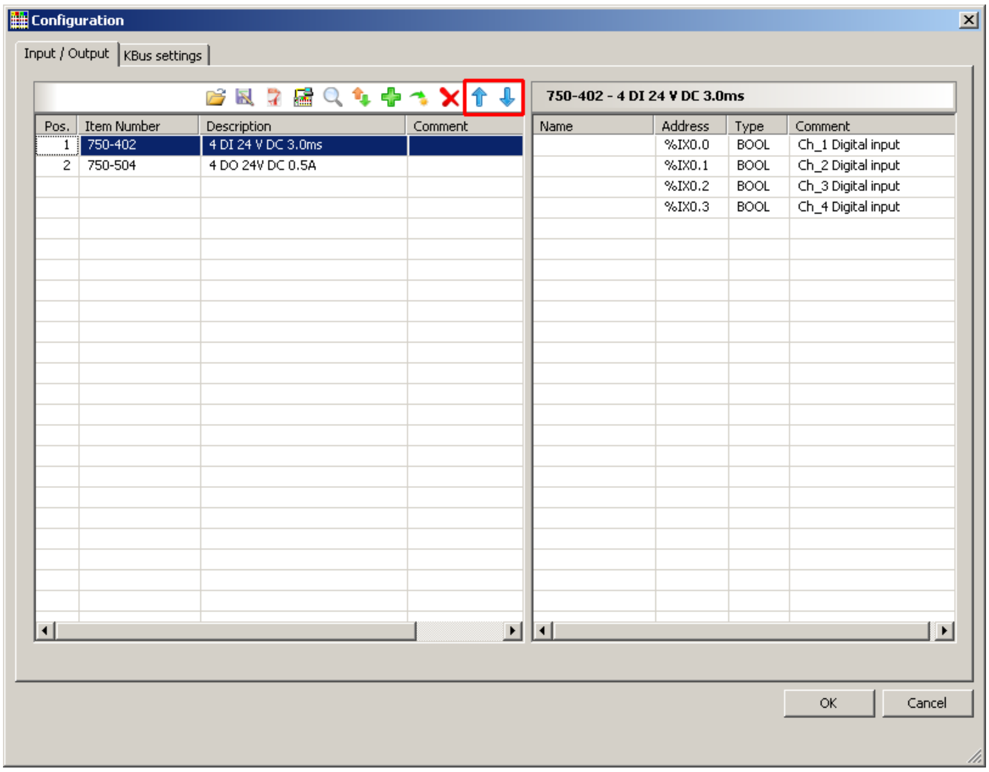


Figure 47: “Module selection” window

11. You can change the position of an I/O module by marking it and then using the arrow buttons at the right edge of the window to move it up or down.



Pos.	Item Number	Description	Comment
1	750-402	4 DI 24 V DC 3.0ms	
2	750-504	4 DO 24V DC 0.5A	

750-402 - 4 DI 24 V DC 3.0ms

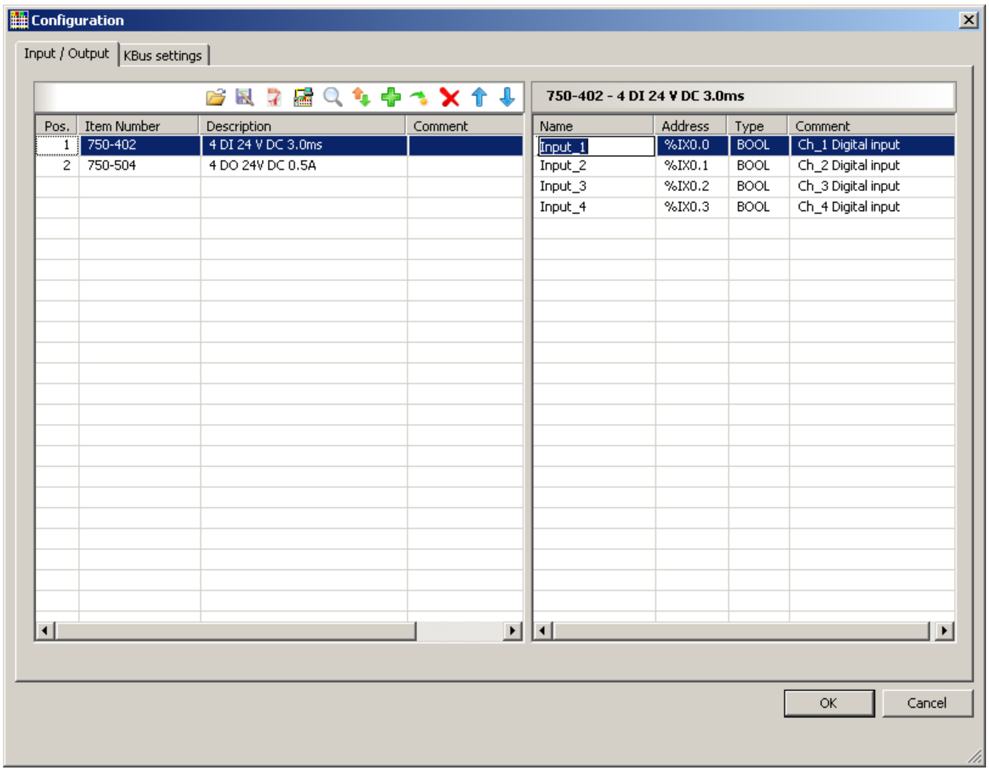
Name	Address	Type	Comment
	%IX0.0	BOOL	Ch_1 Digital input
	%IX0.1	BOOL	Ch_2 Digital input
	%IX0.2	BOOL	Ch_3 Digital input
	%IX0.3	BOOL	Ch_4 Digital input

Figure 48: I/O Configurator with defined I/O modules

12. Use **[Import configuration from file]** to add a configuration imported previously using WAGO-I/O-*CHECK*.

13. To close the I/O Configurator, click **[OK]**.

14. The individual inputs and outputs of the selected I/O module are displayed in the right half of the configuration window.
Here, you can declare a dedicated variable in the “Name” column for each input and output, e.g., “Output_1”, “Output_2”, “Input_1”, “Input_2”.

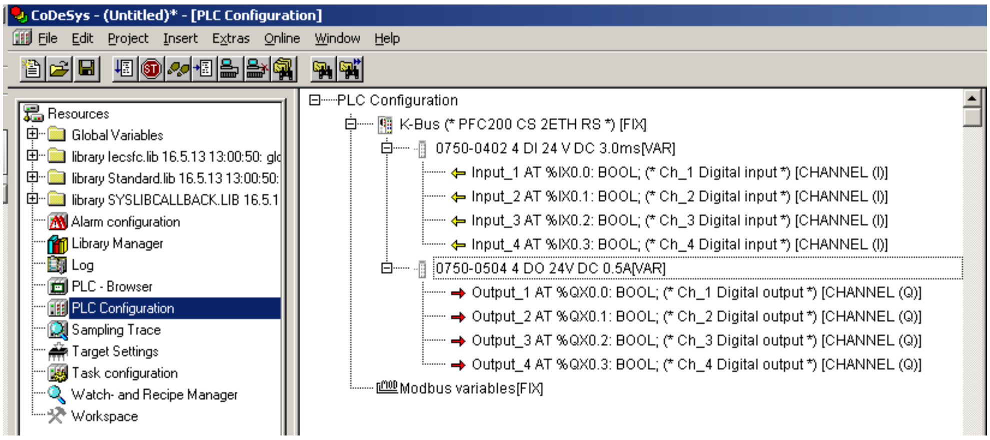


Pos.	Item Number	Description	Comment
1	750-402	4 DI 24 V DC 3.0ms	
2	750-504	4 DO 24V DC 0.5A	

Name	Address	Type	Comment
Input_1	%IX0.0	BOOL	Ch_1 Digital input
Input_2	%IX0.1	BOOL	Ch_2 Digital input
Input_3	%IX0.2	BOOL	Ch_3 Digital input
Input_4	%IX0.3	BOOL	Ch_4 Digital input

Figure 49: Variable declaration

15. The added I/O modules appear in the control configuration under “K-Bus[FIX]” with their associated fixed addresses and, where applicable, their previously set variable name.

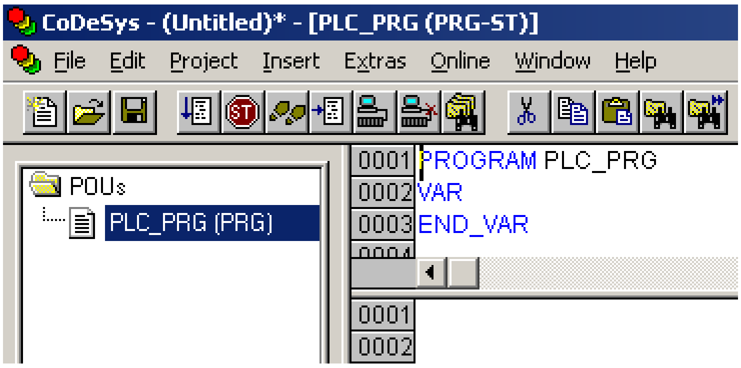


```
PLC Configuration
  K-Bus (* PFC200 CS 2ETH RS *) [FIX]
    0750-0402 4 DI 24 V DC 3.0ms[VAR]
      Input_1 AT %IX0.0: BOOL; (* Ch_1 Digital input *) [CHANNEL (I)]
      Input_2 AT %IX0.1: BOOL; (* Ch_2 Digital input *) [CHANNEL (I)]
      Input_3 AT %IX0.2: BOOL; (* Ch_3 Digital input *) [CHANNEL (I)]
      Input_4 AT %IX0.3: BOOL; (* Ch_4 Digital input *) [CHANNEL (I)]
    0750-0504 4 DO 24V DC 0.5A[VAR]
      Output_1 AT %QX0.0: BOOL; (* Ch_1 Digital output *) [CHANNEL (Q)]
      Output_2 AT %QX0.1: BOOL; (* Ch_2 Digital output *) [CHANNEL (Q)]
      Output_3 AT %QX0.2: BOOL; (* Ch_3 Digital output *) [CHANNEL (Q)]
      Output_4 AT %QX0.3: BOOL; (* Ch_4 Digital output *) [CHANNEL (Q)]
  Modbus variables[FIX]
```

Figure 50: Control configuration: I/O modules with their associated addresses

8.2.4 Editing the Program Function Block

To edit the PLC_PRG program function block, go to the “Function block” tab and double-click on the PLC_PRG program module.


CoDeSys - (Untitled)* - [PLC_PRG (PRG-ST)]
File Edit Project Insert Extras Online Window Help
POUs
PLC_PRG (PRG)

```
0001 PROGRAM PLC_PRG
0002 VAR
0003 END_VAR
0004
```

Figure 51: Program function block

The following example illustrates the editing of the program function block. To do this, an input is assigned to an output:

1. Press **[F2]** to open the Input assistant, or right click and select “Input assistant” from the contextual menu.

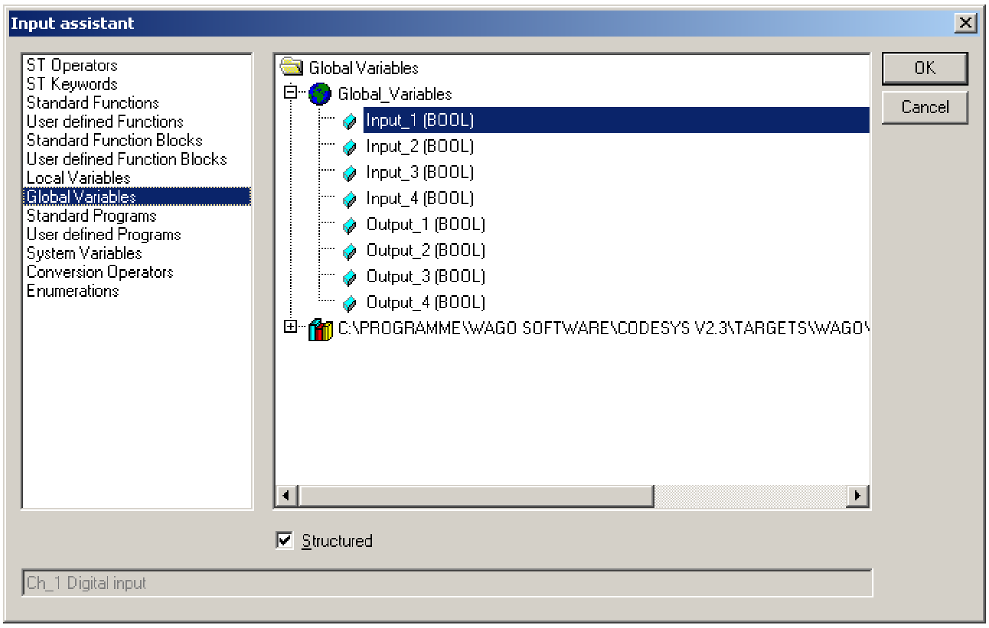
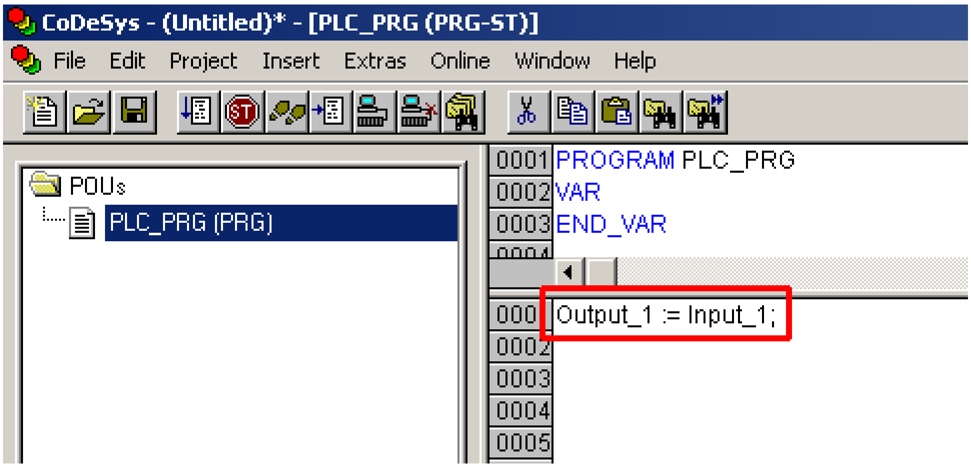


Figure 52: Input assistant for selecting variables

2. Under “Global variables” select the previously declared variable “Output_1” and click **[OK]** to add it.

3. Enter the allocation “=” behind the variable name.

4. Repeat Step 2 for the “Input_1” variable.


CoDeSys - (Untitled)* - [PLC_PRG (PRG-ST)]
File Edit Project Insert Extras Online Window Help
POUs
PLC_PRG (PRG)

```
PROGRAM PLC_PRG
VAR
END_VAR
```

```
Output_1 := Input_1;
```

Figure 53: Example of an allocation

5. To compile, click on **Project > Compile all** in the menu bar.

8.2.5 Loading and Running the PLC Program in the Fieldbus Controller (ETHERNET)

Requirement:

- The simulation is deactivated (**Online > Simulation**).
- The PC is linked to the controller via ETHERNET. Refer to Section “Device Description” > …> “ETHERNET – X1, X2 Network Connection”.

Proceed as follows:

1. In the menu bar click on **Online** and select **Communication parameters …**. The "Communication Parameters" window opens.
2. To select a communication link click on **[New …]** in the “Communication Parameters” window. A window opens in which you can define a communication link.

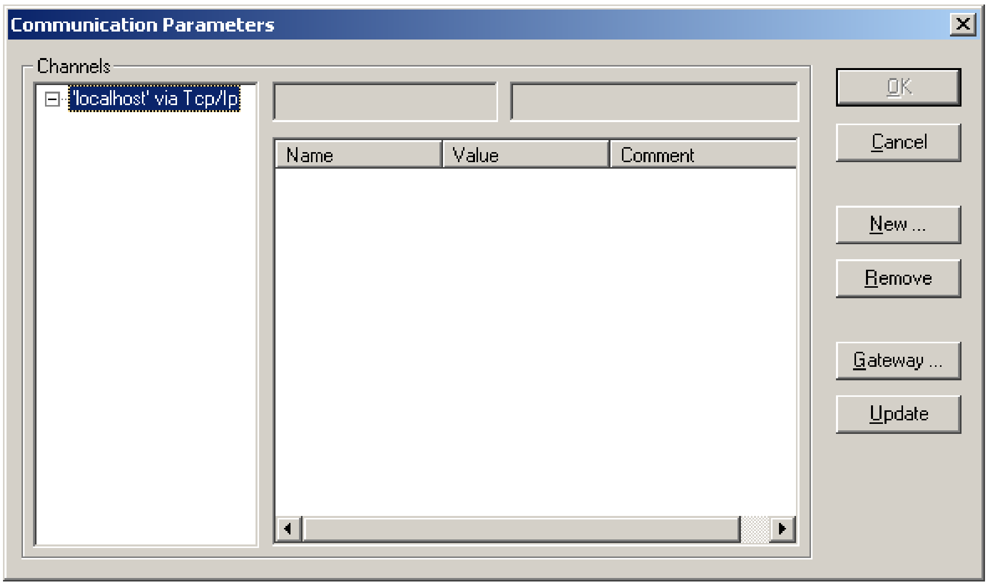
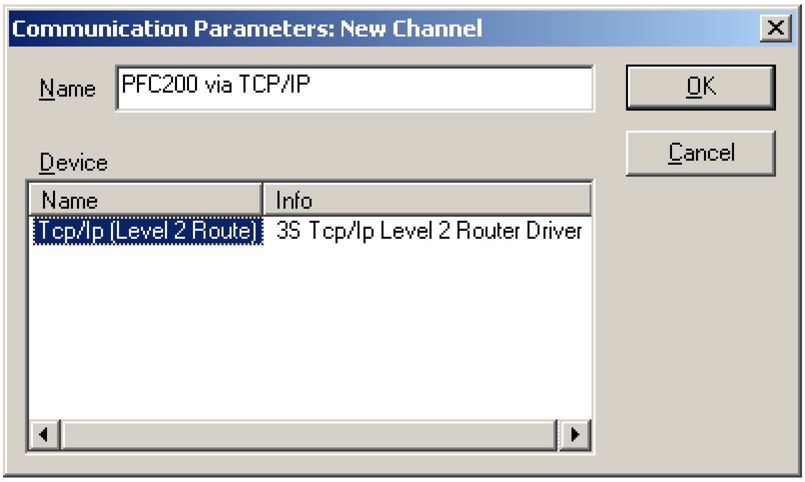


Figure 54: Creating a communication link – Step 1

3. In the “Name” field enter a designation for your fieldbus controller and then click on “Tcp/Ip (Level 2 Route)”. Then click **[OK]**.


Communication Parameters: New Channel

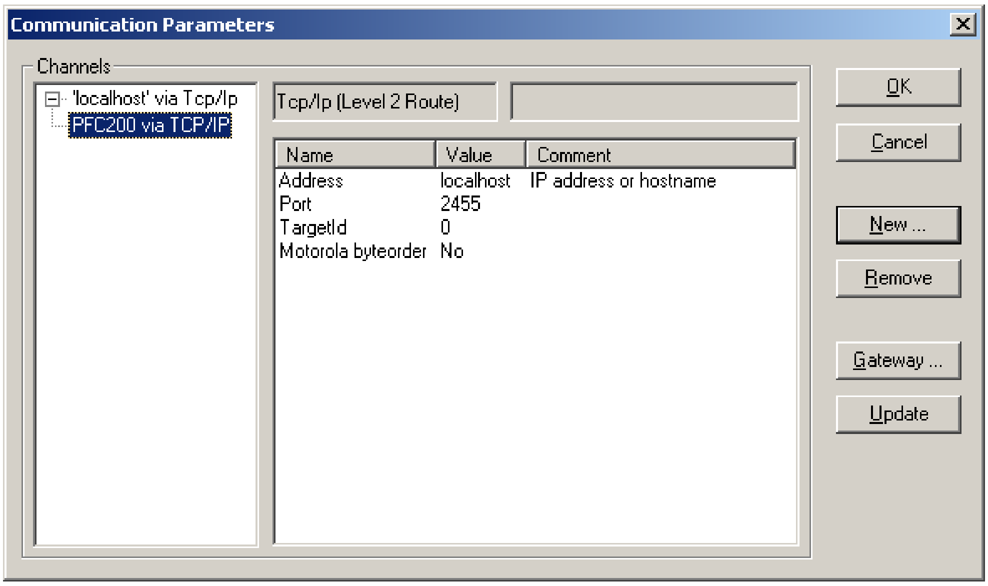
Name: PFC200 via TCP/IP

Device

Name	Info
Tcp/Ip (Level 2 Route)	3S Tcp/Ip Level 2 Router Driver

Figure 55: Creating a communication link – Step 2

4. In the “Communication Parameters” window enter the **IP address of your fieldbus controller** in the “Address” field and then press Enter. To close the window click on **[OK]**.
   Zum Auswählen eines bereits angelegten Feldbuscontrollers selektieren Sie diesen im linken Fenster, und klicken Sie anschließend auf **[OK]**.


Communication Parameters

Channels
- 'localhost' via Tcp/Ip
  - PFC200 via TCP/IP

Tcp/Ip (Level 2 Route)

Name	Value	Comment
Address	localhost	IP address or hostname
Port	2455	
TargetId	0	
Motorola byteorder	No	

Figure 56: Creating a communication link – Step 3

5. Transfer the PLC program by clicking on **Online** in the menu bar and select **Login**.

6. Ensure that the Run/Stop switch for the fieldbus controller is set to “Run”.

7. Start the PLC program by clicking on **Online** > **Start** in the menu bar.

8.2.6 Creating a Boot Project

Create a boot project to ensure that the PLC program starts automatically again after a fieldbus controller restart. In the menu bar select **Online** > **Create boot project**. You must be logged in to CODESYS to use this function.

Note

Automatic loading of the boot project

In addition, you can load the boot project automatically when starting the fieldbus controller. Click on the “Resources” tab and open “Target system settings”. Select the “General” tab and “Load boot project automatically”.

If a boot project (DEFAULT.PRG and DEFAULT.CHK) is present under */home/codesys* and the “Run/Stop” switch of the fieldbus controller is set to “Run”, the fieldbus controller automatically starts with the processing of the PLC program. The PLC program is not started if the switch is set to “Stop”.

If a PLC program is running in the fieldbus controller, a PLC task starts with the reading of the fieldbus data (only with fieldbus controllers and fieldbus connection), the integrated input and output data and the I/O modules. The output data changed in the PLC program is updated after the PLC task is processed. A change in operating mode (“Stop/Run”) is only carried out at the end of a PLC task. The cycle time includes the time from the start of the PLC program to the next start. If a larger loop is programmed within a PLC program, the task time is prolonged accordingly. The inputs and outputs are updated during processing. These updates only take place at the end of a PLC task.

8.3 Syntax of Logical Addresses

Access to individual memory elements according to IEC 61131-3 is possible using only the following special symbols:

Table 80: Syntax of logical addresses

Item	Prefix	Description	Notes:
1	%	Starts the absolute address	-
2	I	Input	
	Q	Output	
	M	Flag	
3	X	Single bit	Data width
	B-	Byte (8 bits)	
	W	Word (16 bits)	
	D	Double word (32 bits)	
4		Address	

Two examples:

Addressing by word	%QW27 (28th word)
Addressing by bit	%IX1.9 (10th bit in word 2)

Enter the character string of the absolute address without empty spaces. The first bit of a word has an address of 0.

8.4 Creating Tasks

Set the time response and the priority of individual tasks in the task configuration.

Note

Watchdog

In an application program without task configuration, there is no watchdog that monitors the cycle time of the application program (PLC_PRG).

Create a task as follows:

1. Open the task configuration by double-clicking on the “Task configuration” module in the “Resources” tab.

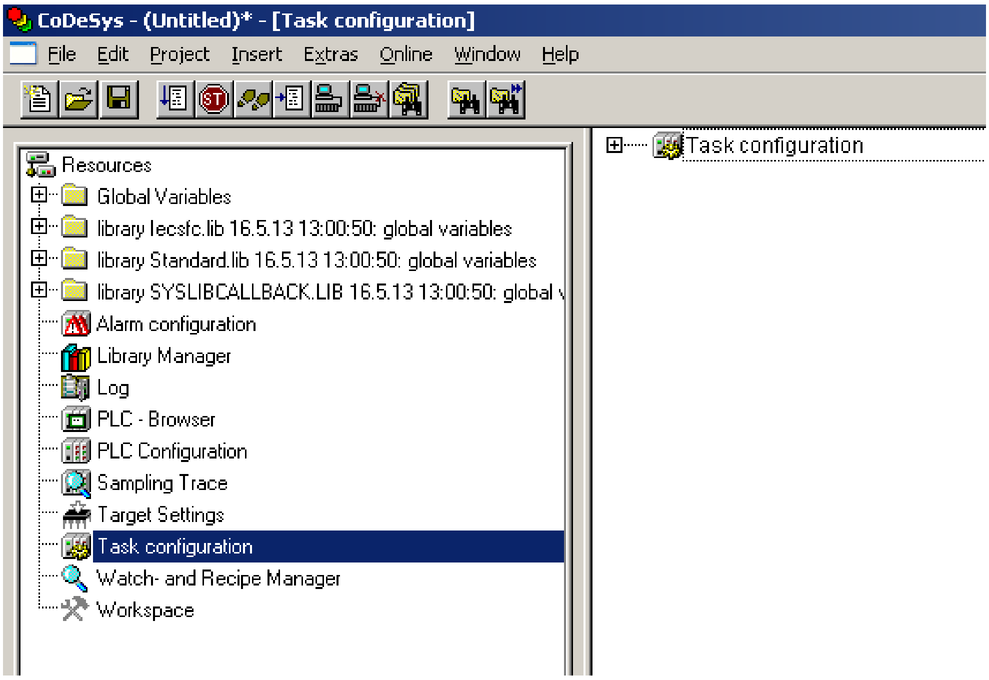

CoDeSys - (Untitled)* - [Task configuration]
File Edit Project Insert Extras Online Window Help
Resources
Global Variables
library Iecsfc.lib 16.5.13 13:00:50: global variables
library Standard.lib 16.5.13 13:00:50: global variables
library SYSLIBCALLBACK.LIB 16.5.13 13:00:50: global v
Alarm configuration
Library Manager
Log
PLC - Browser
PLC Configuration
Sampling Trace
Target Settings
Task configuration
Watch- and Recipe Manager
Workspace
Task configuration

Figure 57: Task configuration

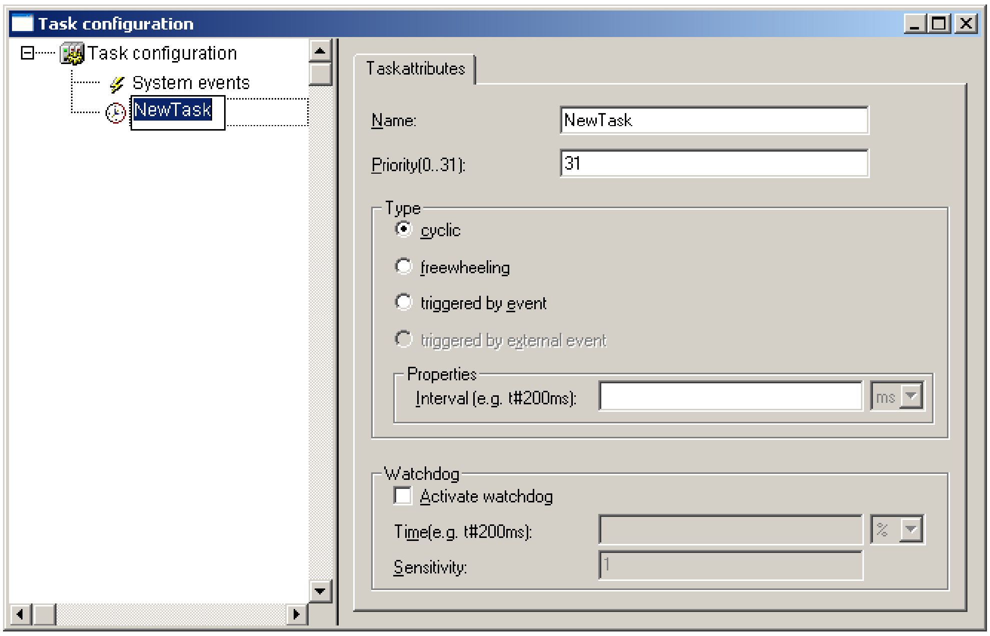
2. To create a task right-click on “Task configuration” and in the contextual menu select “Attach task”.

3. To assign a new name to the task (e.g. PLC_Prog), click on “New Task”. Then select the type of task. In this example, this is the “cyclic” type.

Note

Observe the cycle time!

The minimum cycle time for I/O-based tasks is 2 milliseconds (ms)!



Task configuration

- Task configuration
  - System events
  - NewTask

Taskattributes

Name: NewTask

Priority(0..31): 31

Type
- ● cyclic
- ○ freewheeling
- ○ triggered by event
- ○ triggered by external event

Properties
Interval (e.g. t#200ms): ms

Watchdog
- □ Activate watchdog

Time(e.g. t#200ms): %

Sensitivity: 1

Figure 58: Changing task names 1

4. Add the program module PLC_PRG that you have just created (see Section “Editing the Program Modules”). To do this, right-clock on the “Clock” symbol and in the contextual menu select “Attach program call-up”. Then, click the **[...]** button and **[OK]**.

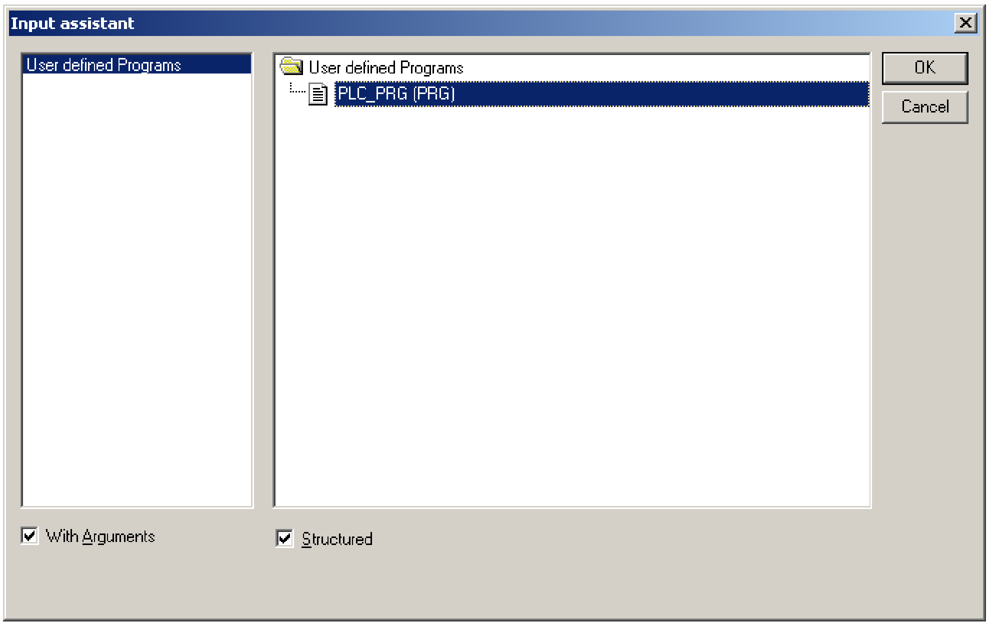
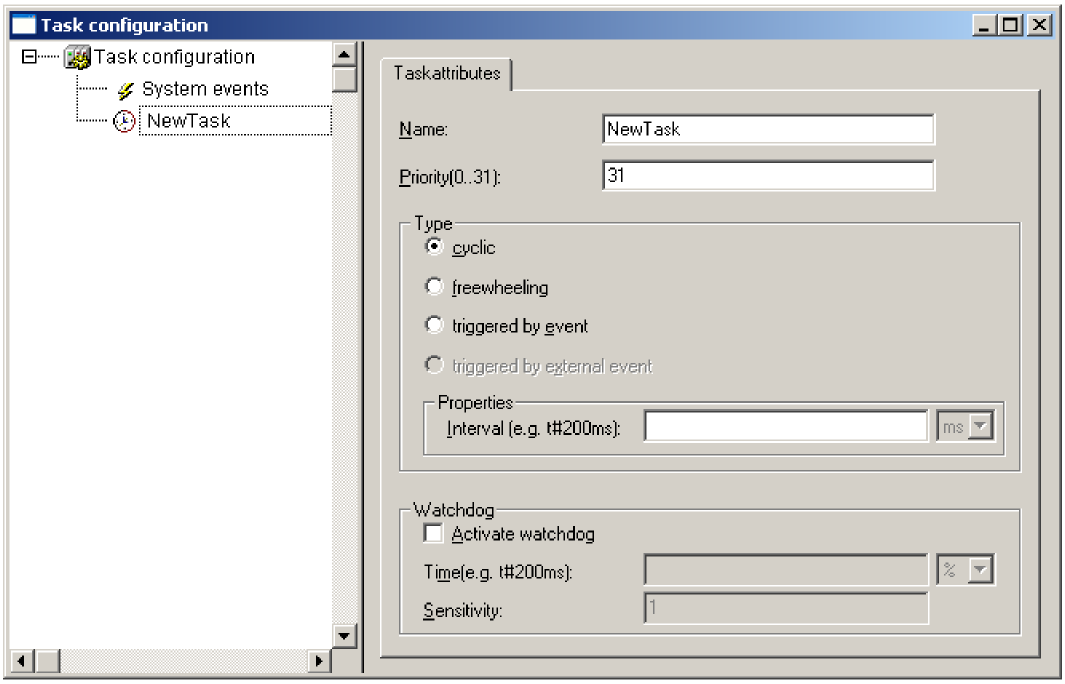


Figure 59: Call-up to add to the program module

5. Compile the example program by selecting **Project > Rebuild all** in the context menu.

8.4.1 Cyclic Tasks

You can assign a priority for each task in order to establish the task processing sequence.



Name: NewTask

Priority(0..31): 31

Type
● cyclic
○ freewheeling
○ triggered by event
○ triggered by external event

Properties
Interval (e.g. t#200ms): ms

Watchdog
□ Activate watchdog
Time(e.g. t#200ms): %
Sensitivity: 1

Figure 60: Cyclic task

Note

Order of Task Processing
The priorities given below do not specifiy the order of task processing. The tasks start in an arbitrary order.

Priority 0 … 5:
Important arithmetic operations and synchronized access to I/O module process images are to be carried out as tasks with the highest priorities 0 … 5. These tasks are processed fully according to priority and correspond to Linux® RT priorities -79 through -74.

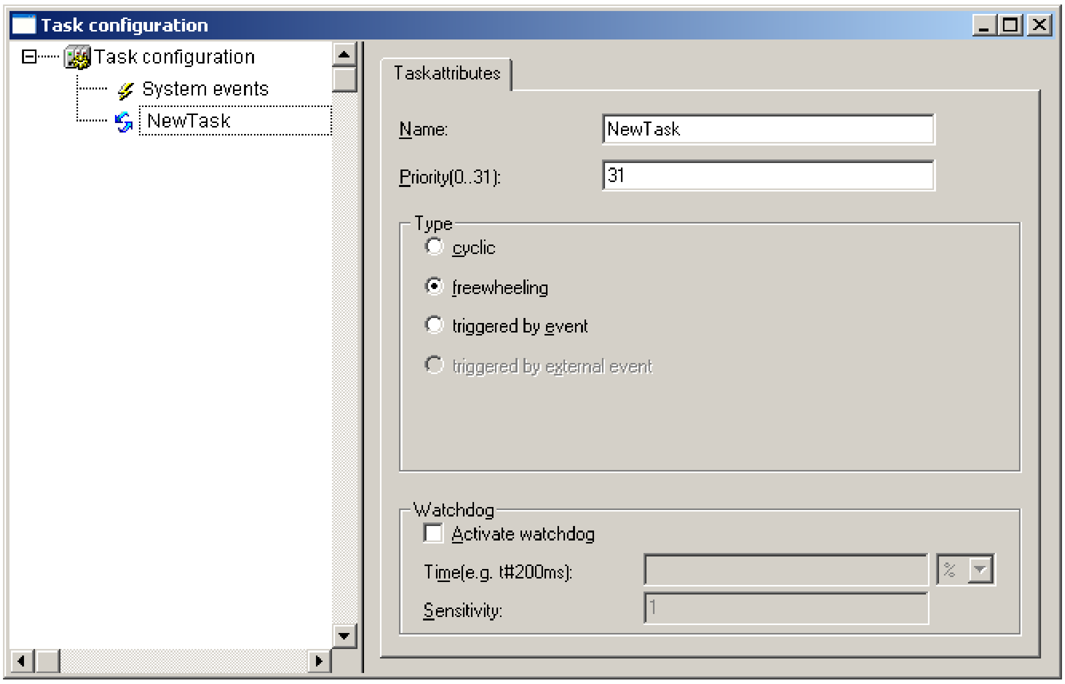
Priority 6 … 20:
Real-time access, such as access to ETHERNET and the file system, to fieldbus data and to the RS-232 interface (when available) are to be carried out as tasks with average priorities 6 … 20. These tasks are processed fully according to priority and correspond to Linux® RT priorities -40 through -26.

Priority 21 … 31:
Applications such as long-lasting arithmetic operations and non-real-time-relevant access to ETHERNET and the file system, to fieldbus data and the RS-232 interface (when provided) are to be carried out as tasks with the lowest priorities 21 … 31. No prioritiy distinction is made between tasks of priorities 21 … 31.

These tasks all receive the same computing time from the operating system ("Completely Fair Scheduler" procedure).

8.4.2 Freewheeling Tasks

So-called freewheeling tasks are not processed in cycles. Their processing depends solely on the current capacity of the system. The input field "Priority (0 … 31)" is provided for freewheeling tasks without a function. These tasks are handled as tasks with priority 21 … 31.



Name: NewTask
Priority(0..31): 31
Type
○ cyclic
● freewheeling
○ triggered by event
○ triggered by external event
Watchdog
□ Activate watchdog
Time(e.g. t#200ms):
Sensitivity: 1

Figure 61: Freewheeling task

Note

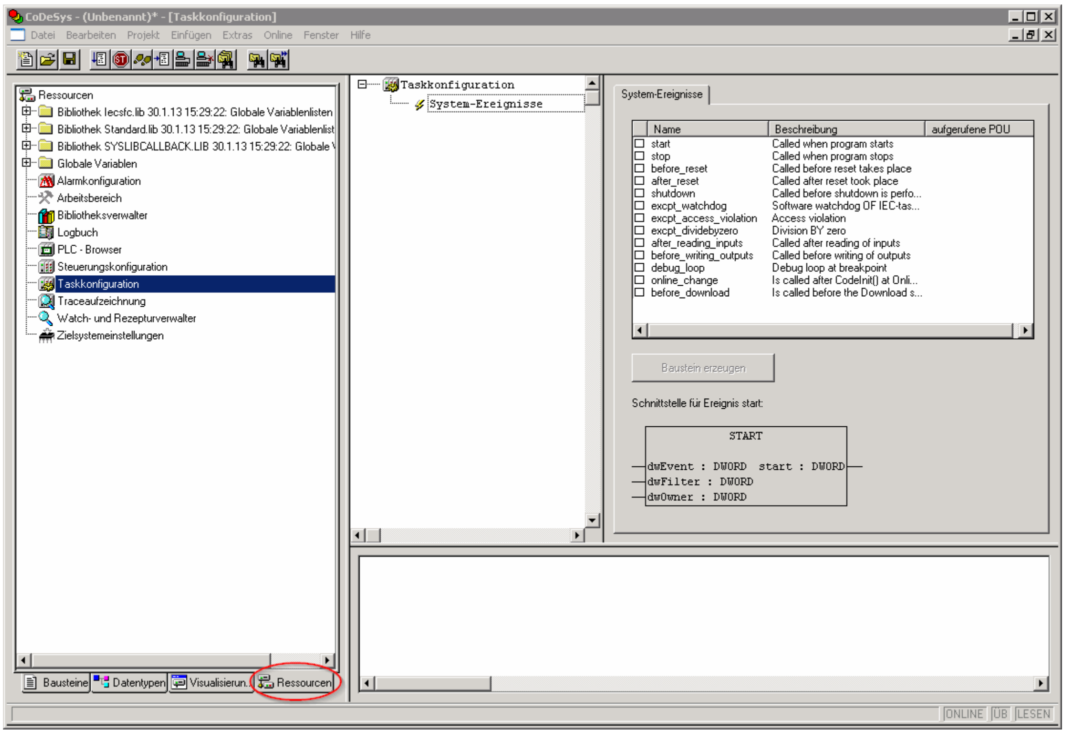
PLC-PRG as Freewheeling Task without Task Configuration

If you do not perform any task configuration, the program PLC_PRG is carried out with the lowest priority at an interval of 10 ms. The runtime of "freewheeling tasks" is not monitored by a CODESYS watchdog.

8.5 System Events

Event tasks can be used in the CODESYS task configuration in addition to cyclical tasks. Event tasks call up certain events in the device.

To activate events and define a program to be called up, open the window "Task configuration" in the "Resources" tab in the CODESYS development environment.


CoDeSys - (Unbenannt)* - [Taskkonfiguration]
Datei Bearbeiten Projekt Einfügen Extras Online Fenster Hilfe
Ressourcen
Bibliothek Iecsfc.lib 30.1.13 15:29:22: Globale Variablenlisten
Bibliothek Standard.lib 30.1.13 15:29:22: Globale Variablenlist
Bibliothek SYSLIBCALLBACK.LIB 30.1.13 15:29:22: Globale V
Globale Variablen
Alarmkonfiguration
Arbeitsbereich
Bibliotheksverwalter
Logbuch
PLC - Browser
Steuerungskonfiguration
Taskkonfiguration
Traceaufzeichnung
Watch- und Rezepturverwalter
Zielsystemeinstellungen
Bausteine
Datentypen
Visualisierun...
Ressourcen
Taskkonfiguration
System-Ereignisse
System-Ereignisse
Baustein erzeugen
ONLINE
ÜB
LESEN

	Name	Beschreibung	aufgerufene POU
□	start	Called when program starts	
□	stop	Called when program stops	
□	before_reset	Called before reset takes place	
□	after_reset	Called after reset took place	
□	shutdown	Called before shutdown is perfo...	
□	excpt_watchdog	Software watchdog OF IEC-tas...	
□	excpt_access_violation	Access violation	
□	excpt_dividebyzero	Division BY zero	
□	after_reading_inputs	Called after reading of inputs	
□	before_writing_outputs	Called before writing of outputs	
□	debug_loop	Debug loop at breakpoint	
□	online_change	Is called after CodeInit() at Onli...	
□	before_download	Is called before the Download s...	

Schnittstelle für Ereignis start:

```
                START

—dwEvent : DWORD  start : DWORD—
—dwFilter : DWORD
—dwOwner : DWORD
```

Figure 62: CODESYS – System events

Note

Do not set debug points in the event handlers!

Debug points in event handlers can lead to unforeseeable errors and must therefore not be set!

The following events can be activated:

Table 81: Events

Name	Description
start	The event is called directly after the user program starts.
stop	The event is called directly after the user program stops.
before_reset	The event is called directly before the user program is reset.
after_reset	The event is called directly after the user program is reset.
shutdown	The event is called directly before the user program is shutdown.
excpt_watchdog	The event is called if a task watchdog is recognized.
excpt_access_violation	The event is called if a memory access error to an invalid memory area is recognized. (incorrect pointer, invalid array index, invalid data descriptor)
excpt_dividebyzero	The event is called if a division by zero is recognized.
after_reading_inputs	The event is triggered after reading all of the inputs independent of the user program.
before_writing_outputs	The event is triggered before writing all of the outputs independent of the user program.
debug_loop	This event is triggered at every task call, if a breakpoint was reached in this task and the processing of this task is therefore blocked.
online_change	This event is called up after initialization of the program on an online change.
before_download	This event is always called up before a download from the IDE to the device takes place.

Note

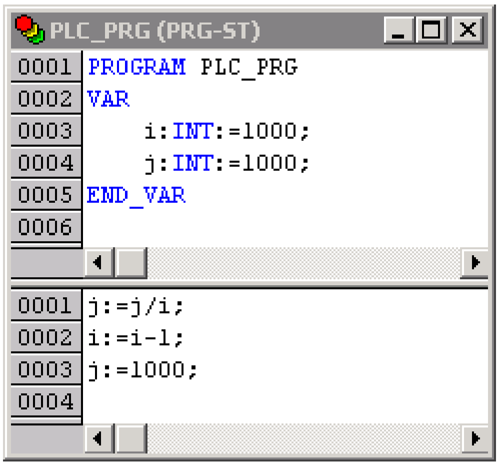
Application stops on a non-defined event handler!

If “excpt” events occur in the system and an event handler has not been defined, the application goes into the "Stop" status.

8.5.1 Creating an Event Handler

The example here is provided to illustrate how to define and use an event handler. The event handler “excpt_dividebyzero” is used in this example.

First, a program is generated in the PLC_PRG- module which provokes division by 0.

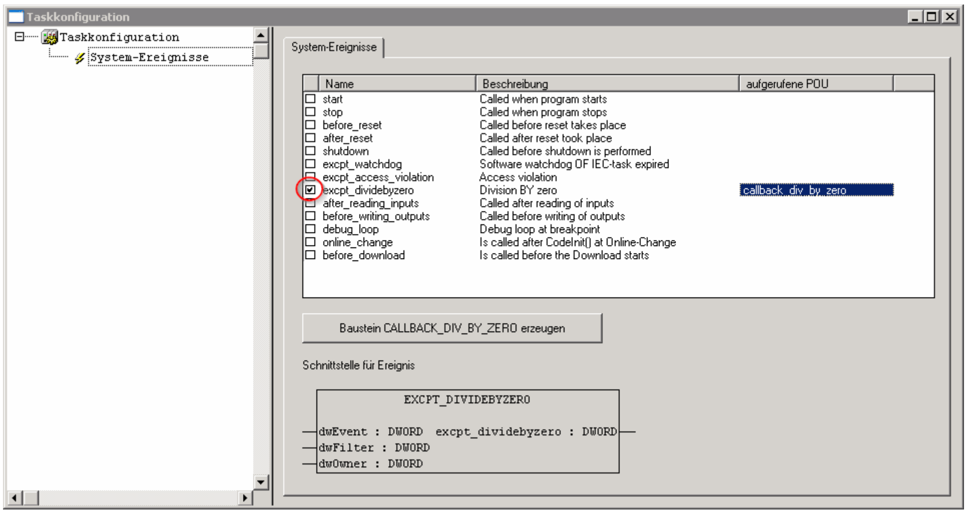


```
PROGRAM PLC_PRG
VAR
     i:INT:=1000;
     j:INT:=1000;
END_VAR

```

```
j:=j/i;
i:=i-1;
j:=1000;

```

Figure 63: CODESYS program provokes division by “0”

After this, the system event “excpt_dividebyzero” is activated in the Task Configurator and the name of the event handler to be generated is entered in the column “Called POU”.



	Name	Beschreibung	aufgerufene POU
□	start	Called when program starts	
□	stop	Called when program stops	
□	before_reset	Called before reset takes place	
□	after_reset	Called after reset took place	
□	shutdown	Called before shutdown is performed	
□	excpt_watchdog	Software watchdog OF IEC-task expired	
□	excpt_access_violation	Access violation	
☑	excpt_dividebyzero	Division BY zero	callback_div_by_zero
□	after_reading_inputs	Called after reading of inputs	
□	before_writing_outputs	Called before writing of outputs	
□	debug_loop	Debug loop at breakpoint	
□	online_change	Is called after CodeInit() at Online-Change	
□	before_download	Is called before the Download starts	

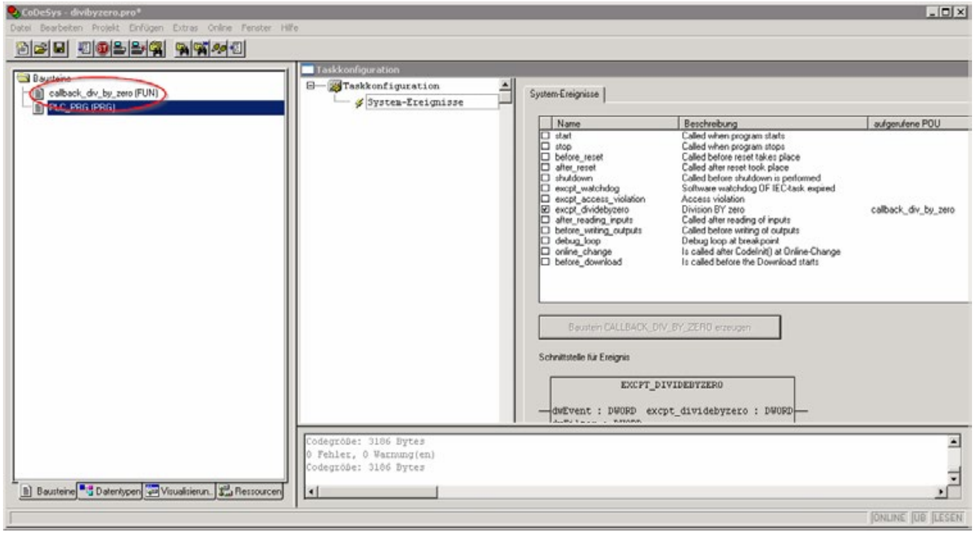
Baustein CALLBACK_DIV_BY_ZERO erzeugen
Schnittstelle für Ereignis

```
EXCPT_DIVIDEBYZERO

dwEvent : DWORD  excpt_dividebyzero : DWORD
dwFilter : DWORD
dwOwner : DWORD
```

Figure 64: CODESYS – Creating and activating an event handler

To generate the event handler, click **[Generate CALLBACK_DIV_BY_ZERO function block]**.

A new function having the defined name then appears in the “Function blocks” tab.


CoDeSys - divibyzero.pro*
Datei Bearbeiten Projekt Einfügen Extras Online Fenster Hilfe
Bausteine
callback_div_by_zero (FUN)
PLC_PRG (PRG)
Bausteine
Datentypen
Visualisierun...
Ressourcen
Taskkonfiguration
Taskkonfiguration
System-Ereignisse
System-Ereignisse
Baustein CALLBACK_DIV_BY_ZERO erzeugen
Schnittstelle für Ereignis
ONLINE
ÜB
LESEN

	Name	Beschreibung	aufgerufene POU
☐	start	Called when program starts	
☐	stop	Called when program stops	
☐	before_reset	Called before reset takes place	
☐	after_reset	Called after reset took place	
☐	shutdown	Called before shutdown is performed	
☐	excpt_watchdog	Software watchdog OF IEC-task expired	
☐	excpt_access_violation	Access violation	
☑	excpt_dividebyzero	Division BY zero	callback_div_by_zero
☐	after_reading_inputs	Called after reading of inputs	
☐	before_writing_outputs	Called before writing of outputs	
☐	debug_loop	Debug loop at breakpoint	
☐	online_change	Is called after CodeInit() at Online-Change	
☐	before_download	Is called before the Download starts	

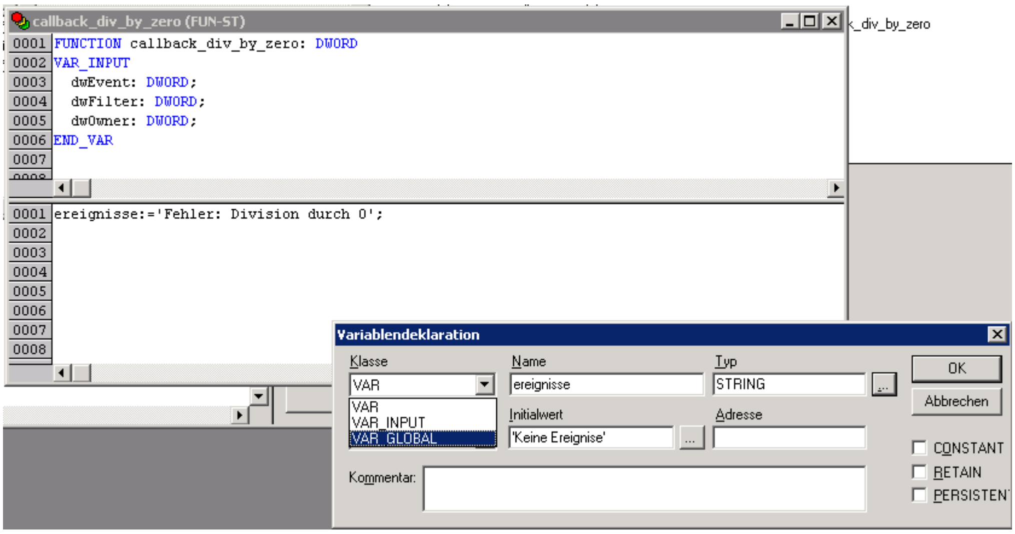
```
EXCPT_DIVIDEBYZERO
dwEvent : DWORD  excpt_dividebyzero : DWORD
```

```
Codegröße: 3186 Bytes
0 Fehler, 0 Warnung(en)
Codegröße: 3186 Bytes
```

Figure 65: CODESYS – New module has been generated

Handling for the event that has occurred is now programmed in this new function.

In the example here, the event is documented in a global variable.



```
FUNCTION callback_div_by_zero: DWORD
VAR_INPUT
  dwEvent: DWORD;
  dwFilter: DWORD;
  dwOwner: DWORD;
END_VAR
```

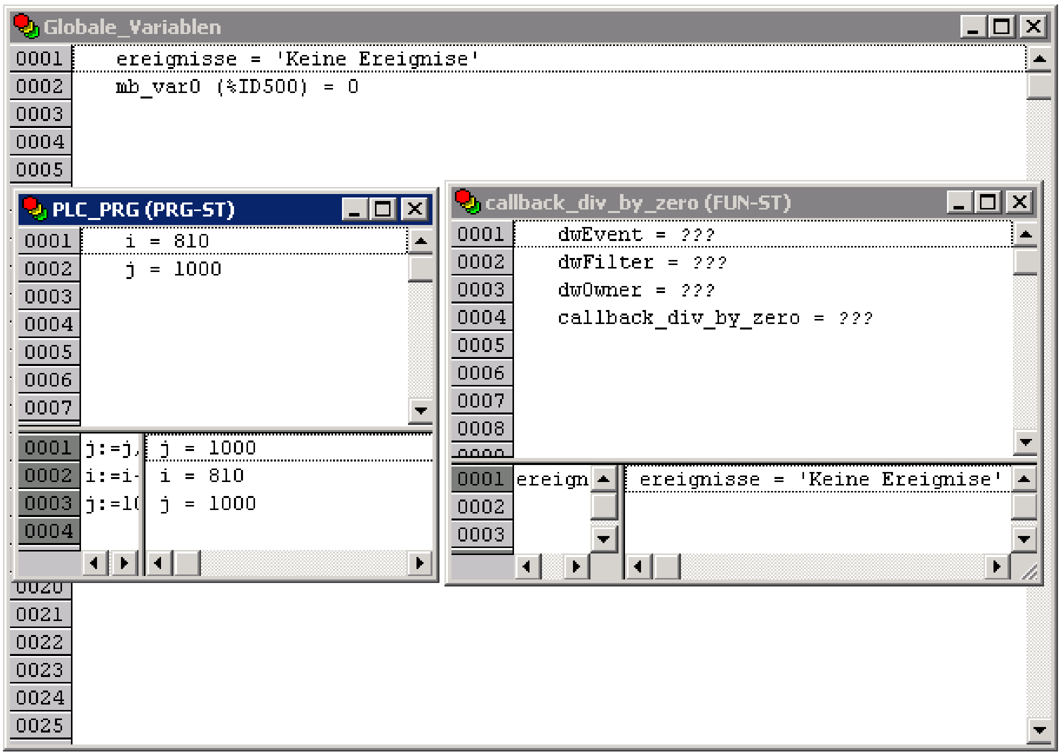
```
ereignisse:='Fehler: Division durch 0';
```

Klasse: VAR_GLOBAL
Name: ereignisse
Typ: STRING
Initialwert: 'Keine Ereignisse'
Adresse:
Kommentar:
☐ CONSTANT
☐ RETAIN
☐ PERSISTENT

Figure 66: CODESYS – Enter the event in a global variable

The newly created project is now supported and can be loaded to the controller.

After startup, the value of the “Events” variable changes only when counter “i” reaches the value 0, meaning that division by 0 has been performed.



```
ereignisse = 'Keine Ereignise'
mb_var0 (%ID500) = 0
```

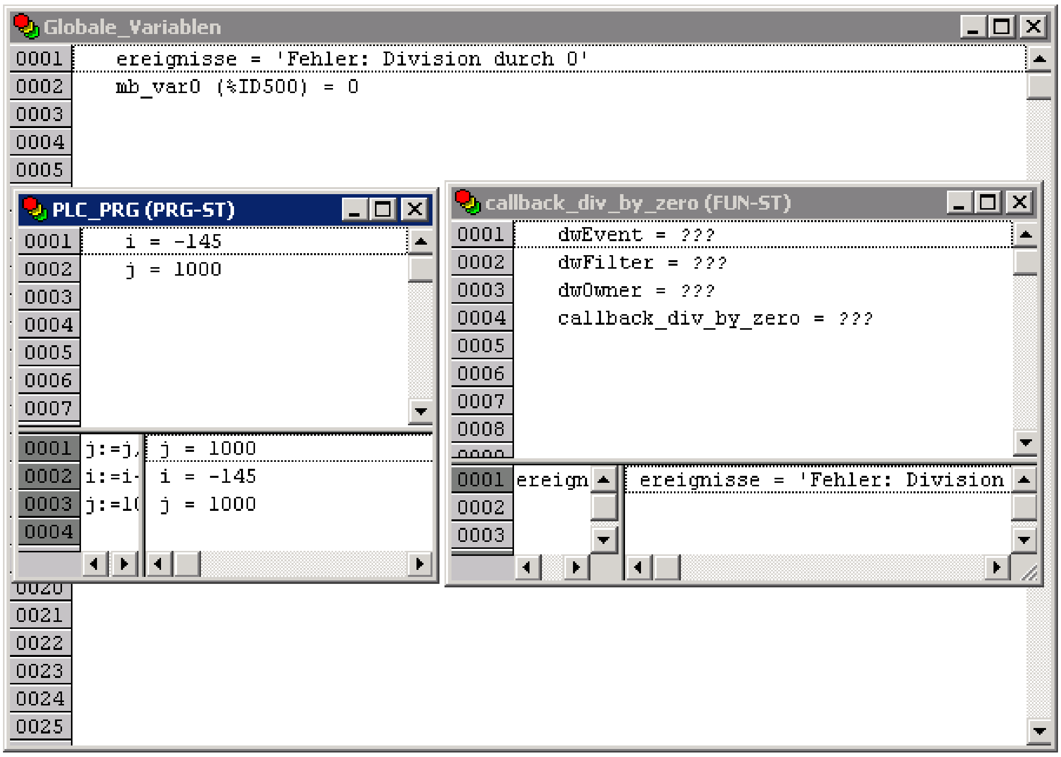
```
i = 810
j = 1000
```

```
j:=j, j = 1000
i:=i- i = 810
j:=1( j = 1000
```

```
dwEvent = ???
dwFilter = ???
dwOwner = ???
callback_div_by_zero = ???
```

```
ereign ereignisse = 'Keine Ereignise'
```

Figure 67: CODESYS – Variable contents prior to division by “0”



```
ereignisse = 'Fehler: Division durch 0'
mb_var0 (%ID500) = 0
```

```
i = -145
j = 1000
```

```
j:=j, j = 1000
i:=i- i = -145
j:=1( j = 1000
```

```
dwEvent = ???
dwFilter = ???
dwOwner = ???
callback_div_by_zero = ???
```

```
ereign ereignisse = 'Fehler: Division
```

Figure 68: CODESYS – Variable contents after division by “0” and call-up of the event handler

8.6 Process Images

A process image is a memory area in which the process data is stored in a defined sequence and consists of the I/O modules attached to the internal bus, the PFC variables, the bit memory address area and the slaves attached to the fieldbus.

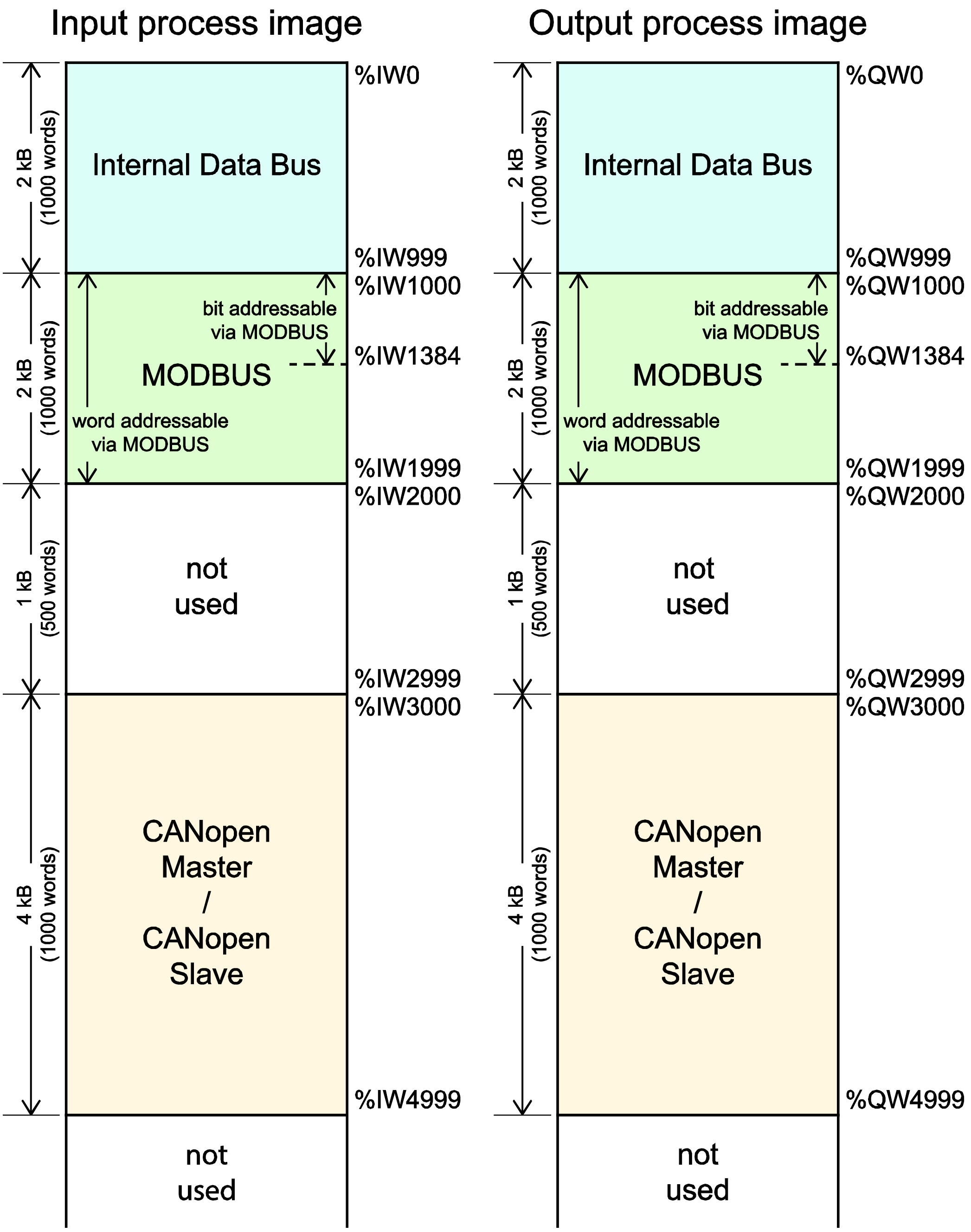


Figure 69: Process image

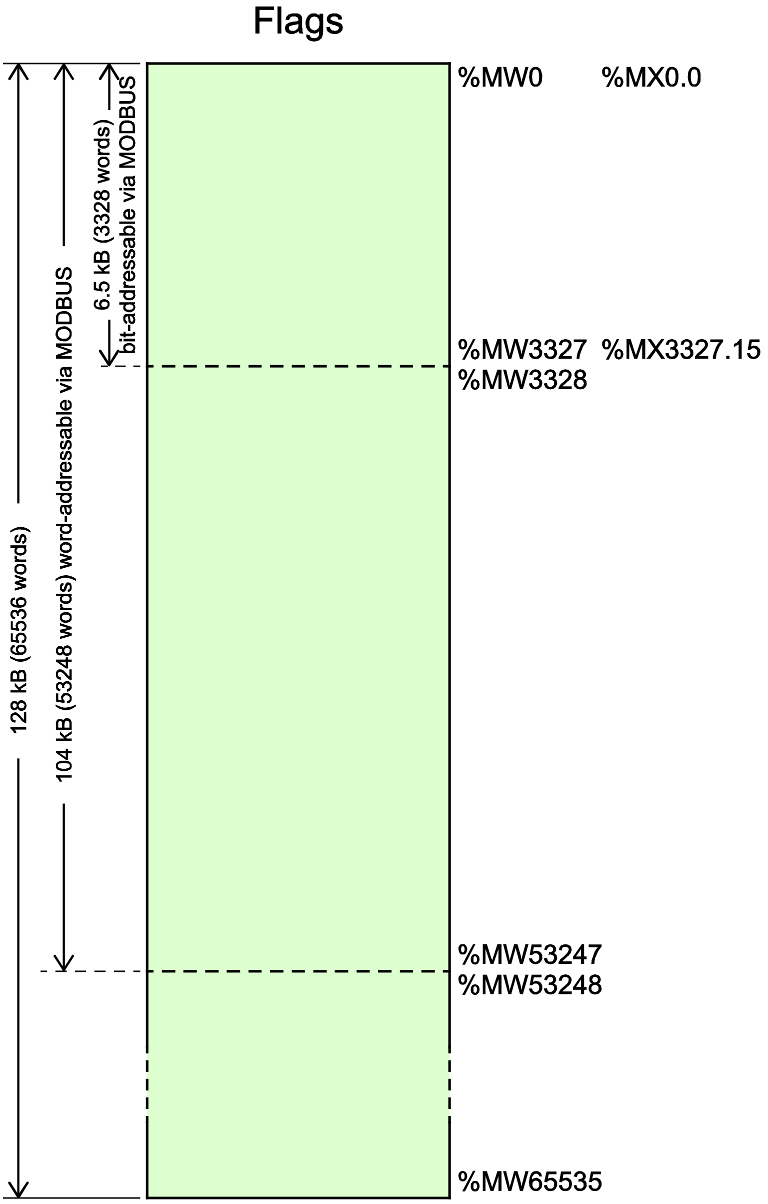


Figure 70: Flag area

8.6.1 Process Images for I/O Modules Connected to the Controller

After starting the fieldbus controller, it automatically detects all connected I/O modules.
The analog input and output data is stored first word by word in the process image. Subsequent to this, come the digital input and output data bits combined to form words.

The size and structure of the process image for the I/O modules connected to the system are described in the appendix.

Note

I/O Module Data Width

The data width of an I/O module is between 0 and 48 bytes.

Note

I/O Module Process Data

Check the I/O module process data whenever you add or remove the modules to/from the fieldbus controller. Changing the I/O module topology results in an adjustment of the process image, as the process data addresses also change.

8.6.2 Process Image for Slaves Connected to the Fieldbus

The size and structure of the process image for the slaves connected to the system are described in the section for the specific fieldbus.

Note

No direct access from fieldbus to the process image for I/O modules!

Any data that is required from the I/O module process image must be explicitly mapped in the CODESYS program to the data in the fieldbus process image and vice versa! Direct access is not possible!

8.7 Access to Process Images of the Input and Output Data via CODESYS 2.3

The following tables describe the possibilities with which you can access the address ranges of the process image for the inputs and outputs connected to the internal data bus.

Table 82: Access to the Process Images of the Input and Output Data – Internal Data Bus

Memory area	**Description**	**Access via PLC**	**Logical Address Space**
Internal data bus input process image	Map of the local input modules (internal data bus, I/O module 1 to 1 bis 64*) in the RAM.	Read	Word %IW0 to %IW999
			Byte %IB0 to %IB1999
Internal data bus output process image	Map of local output modules (internal data bus, I/O module 1 to 64*) in the RAM.	Read/ Write	Word %QW0 to %QW999
			Byte %QB0 to %QB1999

* The use of up to 250 I/O modules is possible with the WAGO internal data bus extension modules.

Table 83: Access to the Process Images of the Input and Output Data – MODBUS

Memory area	Description	Access via PLC	Logical Address Space
MODBUS input process image	MODBUS input variables, addressed by word via MODBUS	Read	Word %IW1000 to %IW1999
			Byte %IB2000 to %IB3999
	MODBUS input variables, addressed by bit via MODBUS	Read	Bit %IX1000.0 …%IX1000.15 to %IX1384.0 … %IX1384.15
MODBUS output process image	MODBUS output variables, addressed by word via MODBUS	Read/ Write	Word %QW1000 to %QW1999
			Byte %QB2000 to %QB3999
	MODBUS output variables, addressed by bit via MODBUS	Read/ Write	Bit %QX1000.0 … %QX1000.15 to %QX1384.0 … %QX1384.15

Table 84: Access to the Process Images of the Input and Output Data – CANopen

Memory area	Description	Access via PLC	Logical Address Space
CANopen input process image	CANopen master or CANopen slave input variables	Read	Word %IW3000 to %IW4999
			Byte %IB6000 to %IB9999
CANopen output process image	CANopen master or CANopen slave output variables	Read/ Write	Word %QW3000 to %QW4999
			Byte %QB6000 to %QB9999

Table 85: Access to the Process Images of the Input and Output Data – Flags

Memory area	Description	Access via PLC	Logical Address Space
Flag variables	Total of 128 kB remanent memory (65536 words).	Read/ Write	%MW0 to %MW65535
	104 kB addressed by word via MODBUS (53248 words)	Read/ Write	Word (MODBUS) %MW0 to %MW3327
	6.5 kB addressed by bit via MODBUS (3328 words).	Read/ Write	Bit (MODBUS) %MX0.0 … %MX0.15 to %MX3327.0 … %MX3327.15
Retain variables	Retain memory addressed by symbols in the NVRAM: 128 kB	Read/ Write	-

* The use of up to 250 I/O modules is possible with the WAGO internal data bus extension modules.

The total size of the memory for flag and retain variables is 128 kB (131060 bytes). The size of these two sections can be customized as required, provided the total (permissible) size is not exceeded.

If you are using bit-oriented addressing, remember that the basic address is word-based. The bits are addressed from 0 to 15.

8.8 Addressing Example

The following addressing example clarifies the access to the process image:

Table 86: Arrangement of the I/O modules for the addressing example

Fieldbus controller	**750-400**	**750-554**	**750-402**	**750-504**	**750-454**	**750-650**	**750-468**	**750-600**
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**

Table 36: Addressing example

I/O module		**Input data**		**Output data**		**Description**
Type	C*					
750-400	1		%IX8.0			**2DI, 24 V, 3 ms:** 1. Digital input module with a data width of 2 bits. As the analog input modules already occupy the first 8 words of the input process image, the 2 bits occupy the lowest-value bits of the 8th word.
	2		%IX8.1			
750-554	1			%QW0		**2AO, 4 – 20 mA:** 1. Analog output module with a data width of 2 words. This module occupies the first 2 words in the output process image.
	2			%QW1		
750-402	1		%IX8.2			**4DI, 24 V:** 2. Digital input module with a data width of 4 bits. These are added to the 2 bits of the 750-400 module and stored in the 8th word of the input process image.
	2		%IX8.3			
	3		%IX8.4			
	4		%IX8.5			
750-504	1				%QX4.0	**4DO, 24 V:** 1. Digital output module with a data width of 4 bits. As the analog output module already occupies the first 4 words of the output process image, the 4 bits occupy the lowest-value bits of the 4th word.
	2				%QX4.1	
	3				%QX4.2	
	4				%QX4.3	
750-454	1	%IW0				**2AI, 4 – 20 mA:** 1. Analog input module with a data width of 2 words. This module occupies the first 2 words in the input process image.
	2	%IW1				
750-650	1	%IW2				**RS-232, C 9600/8/N/1:** The serial interface module is an analog input and output module, which displays 2 words both in the input process image and in the output process image.
		%IW3				
				%QW2		
				%QW3		
750-468	1	%IW4				**4AI, 0 – 10 V S.E:** 2. Analog input module with a data width of 4 words. As the 750-454 and 750-650 analog input and output modules already occupy the first 4 words of the input process image, the 4 words of this I/O module are added behind the others.
	2	%IW5				
	3	%IW6				
	4	%IW7				

Table 36: Addressing example

I/O module		Input data		Output data		Description
Type	**C***					
750-600						**End module** The passive 750-600 end module does not transmit any data.

▨ Analog input and output modules

□ Digital input and output modules

*C: Number of the input/output

8.9 Internal Data Bus Synchronization

The internal data bus cycle and the CODESYS task cycle are optimally automatically synchronized: This depends on the number of I/O modules connected and the fastest CODESYS task cycle set in the fieldbus controller. The synchronization cases described below can therefore take place.

In this chapter, CODESYS task denotes only tasks within CODESYS that contain an access to the internal data bus. Tasks that do not access the internal data bus are not synchronized in the same way as described below. For this, see Section “Creating Tasks.”

8.9.1 Case 1: CODESYS Task Interval Set Smaller than the I/O Module Cycle

Execution of the CODESYS tasks is synchronized with internal data bus cycle time.

The CODESYS task is processed in parallel to the internal data bus cycle. The CODESYS task interval is extended to the internal data bus cycle time. This is necessary so that each CODESYS task is started with new input data from the internal data bus and the output values are also set at the module after each CODESYS task.

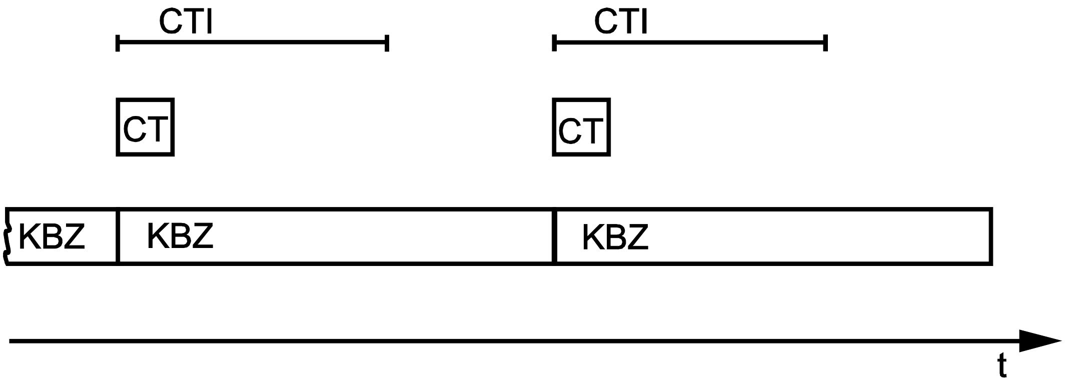


Figure 71: Internal data bus synchronization 01

CTI: CODESYS Task Interval
CT: CODESYS Task that accesses the I/O module of the internal data bus
KBZ: Internal Data Bus Cycle

Example:
CODESYS task interval (CTI): 100 µs
I/O module cycle (KBZ): 2000 µs
Result: Matching of the CODESYS task interval to the I/O module cycle of 2000 µs.

8.9.2 Case 2: CODESYS Task Interval Smaller than Twice the Internal Data Bus Cycle

Execution of the internal data bus is synchronized with the set CODESYS task interval.

At the end of the CODESYS task, the internal bus cycle starts, which is processed synchronously with the fastest CODESYS task. This ensures that when starting each CODESYS Task, current input data are available from the internal data bus and the output values of each CODESYS task are also output to the I/O modules.

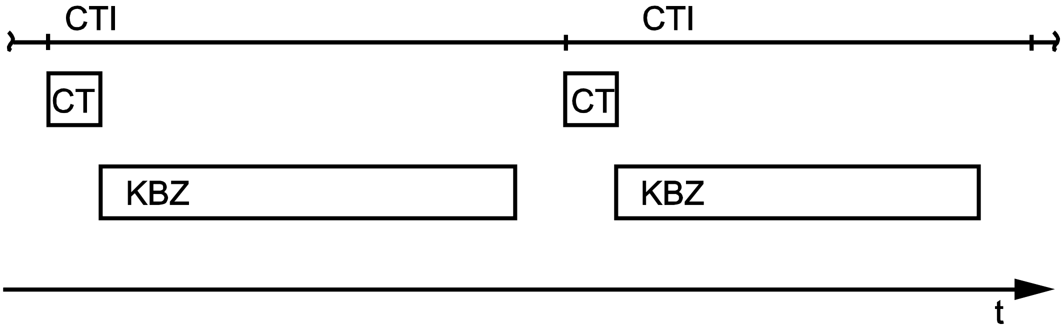


Figure 72: I/O module synchronization 02

CTI: CODESYS Task Interval
CT: CODESYS Task that accesses the I/O module of the internal data bus
KBZ: Internal Data Bus Cycle

Example:
CODESYS task interval (CTI): 2500 µs
Internal data bus cycle (KBZ): 2000 µs
Result: Execution of the internal data bus cycle every 2500 µs.

8.9.3 Case 3: CODESYS Task Interval Greater than Twice the Internal Data Buc Cycle

The I/O data from the internal data bus are refreshed once prior to the CODESYS task and once after the CODESYS task.

Prior to processing the CODESYS task, the internal data bus cycle is executed, which provides the current input data for the CODESYS task. After execution of the CODESYS task, an additional internal data bus cycle is started, which provides the output data to the I/O modules.

This ensures that at the start of every CODESYS task, current input data are available from the internal data bus and the output data from each CODESYS task are quickly output to the I/O modules. This prevents processing of internal data bus cycles that would unnecessarily use a great deal of computing time on the CPU.

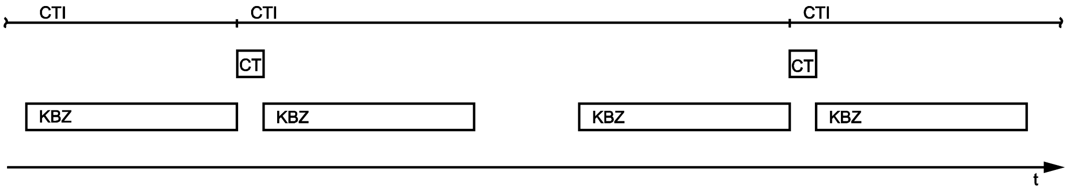


Figure 73: I/O module synchronization 03

CTI: CODESYS Task Interval
CT: CODESYS Task that accesses the I/O module of the internal data bus
KBZ: Internal Data Bus Cycle

Example:
CODESYS task interval (CTI): 500 µs
Internal data bus cycle (KBZ): 2000 µs
Result: Execution of the internal data bus cycle 2000 µs prior to the CODESYS task and once directly after the CODESYS task.

8.9.4 Case 4: CODESYS Task Interval Greater than 10 ms

Synchronization takes place as in case 3; however, the output modules would be reset to their default state after 150 ms without an internal data bus cycle. This reliably prevents the execution of an internal data bus cycle after at least every 10 ms.

The I/O data from the internal data bus are refreshed once before the CODESYS task and once after the CODESYS task and an additional internal data bus cycle is also executed every 10 ms.

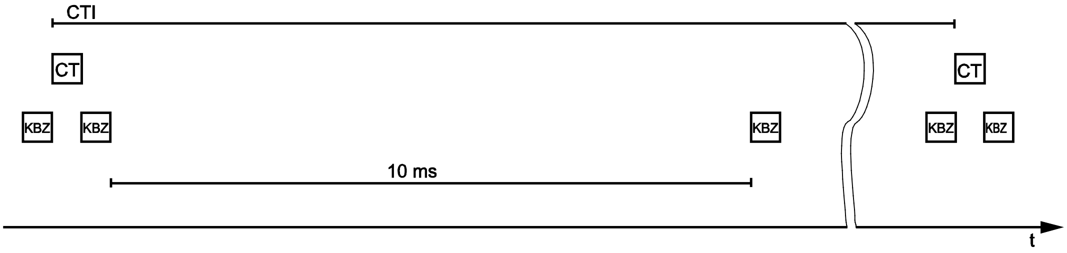
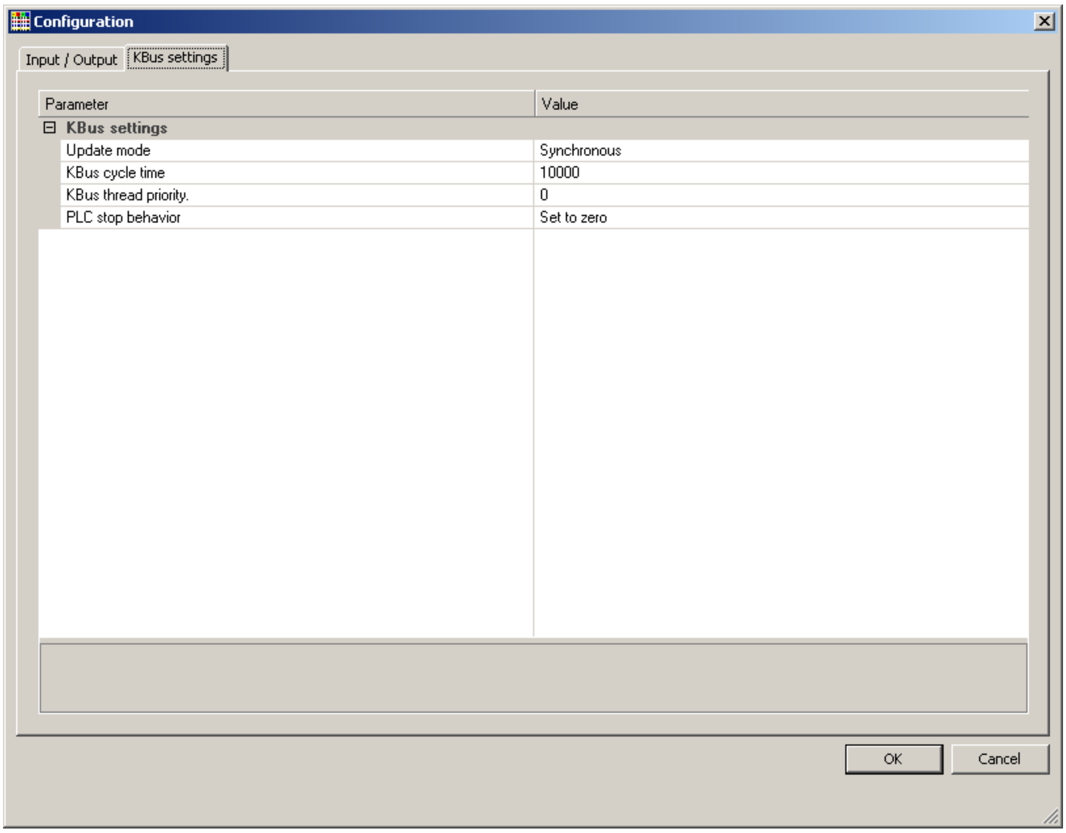


Figure 74: Internal data bus synchronization 04

CTI:	CODESYS Task Interval
CT:	CODESYS task that accesses the I/O module of the internal data bus
KBZ:	Internal data bus cycle

Example:
CODESYS task interval (CTI): 150000 µs
Internal data bus cycle (KBZ): 2000 µs
Result: Execution of the internal data bus cycle 2000 µs prior to the CODESYS task, once directly after the CODESYS task and 10 ms after the previous internal data bus cycle.

8.9.5 Internal Data Bus Configuration



Parameter	Value
KBus settings	
Update mode	Synchronous
KBus cycle time	10000
KBus thread priority.	0
PLC stop behavior	Set to zero

Figure 75: Internal data bus settings

Table 87: Internal Data Bus Settings

Parameter	Explanation	
Update Mode	The Update mode is used to configure how the internal data bus process data is to be updated (refreshed).	
	Asynchronous	In the asynchronous update mode process data are refreshed in cycles at a definable interval.
	Synchronous*	In the synchronous update mode the process data are synchronized with the most rapid CODESYS task that accesses the internal data bus.
Internal Data Bus Cycle Time	The update interval for the internal data bus is set by the cycle time. This setting is effective only in the asynchronous mode.	
	1000 µs	Minimum value 1 millisecond
	10000 µs*	Default value 10 milliseconds
	50000 µs	Maximum value 50 milliseconds
Internal Data Bus Thread Priority	This value indicates the priority for the internal data bus thread. This setting is effective only in the asynchronous mode. This priority is equivalent to the priority of the cyclic CODESYS tasks (see Section “Cyclic Tasks”). This setting is effective only in the asynchronous mode.	
	0*	Highest priority
	15	Lowest priority
PLC stop response	Specifies the response of the internal data bus outputs when the PLC application stops.	
	Hold last value	The output states are retained.
	Set to zero*	Outputs are set to zero.

* Default setting

8.9.5.1 Effect of Update Mode on CODESYS Tasks

8.9.5.1.1 Asynchronous Update Mode

In the asynchronous update mode there is no direct influence on CODESYS task behavior.

Note

Internal data bus “freeze” on priority conflicts!

In the asynchronous update mode there is a risk of the internal data bus “freezing”, as the internal data bus thread operates at the same priority as the IEC tasks. The internal data bus thread must therefore use a priority higher than that of the IEC task to prevent this from occurring.

8.9.5.1.2 Synchronous Update Mode

In the synchronous update mode the runtime behavior of CODESYS tasks can be influenced by the internal data bus. The minimum task interval that can then be achieved depends on the duration of an internal data bus cycle. The duration of an internal data bus cycle, on the other hand, is based on the I/O modules connected to the bus. As a rule of thumb: The shorter the internal data bus structure, the shorter the cycle time and digital modules are faster than analog or complex ones.

In the event of an internal data bus error, the CODESYS tasks are blocked until the error is rectified, i.e., when an internal data bus cycle has been successfully executed again.

Note

No call-up of internal data bus status when internal data bus errors are present!

If an internal data bus error has occurred, it is not possible to call up the internal data bus status using KBUS_ERROR_INFORMATION (mod_com.lib) while in the synchronous update mode.

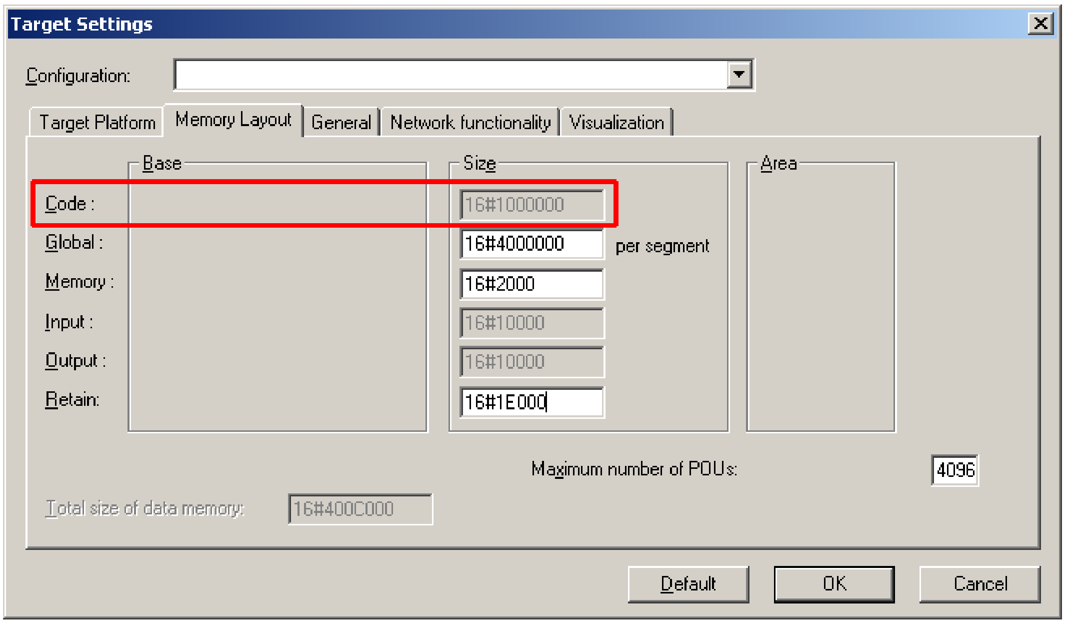
8.10 Memory Settings in CODESYS

The list below illustrates the standard memory allocation of the PFC200:

• Program memory:	16 Mbyte (max.)
• Data memory:	64 Mbytes
• Input data:	64 kbytes
• Output data:	64 kbytes
• Flags:	24 kbytes
• Retain:	104 kbytes
• Function block limitation:	12 * 4096 bytes = 48 kbytes

8.10.1 Program Memory

The program memory (also code memory) cannot be configured and is limited to a maximum of 16 Mbytes. The memory space actually available is based on the scope of installed applications.



Target Settings

Configuration:

Memory Layout

	Base	Size	Area
Code :		16#1000000	
Global :		16#4000000 per segment	
Memory :		16#2000	
Input :		16#10000	
Output :		16#10000	
Retain:		16#1E000	

Maximum number of POUs: 4096

Total size of data memory: 16#400C000

Figure 76: Program memory

8.10.2 Data Memory and Function Block Limitation

The data memory is set for 64 Mbytes in the controller's initial state.

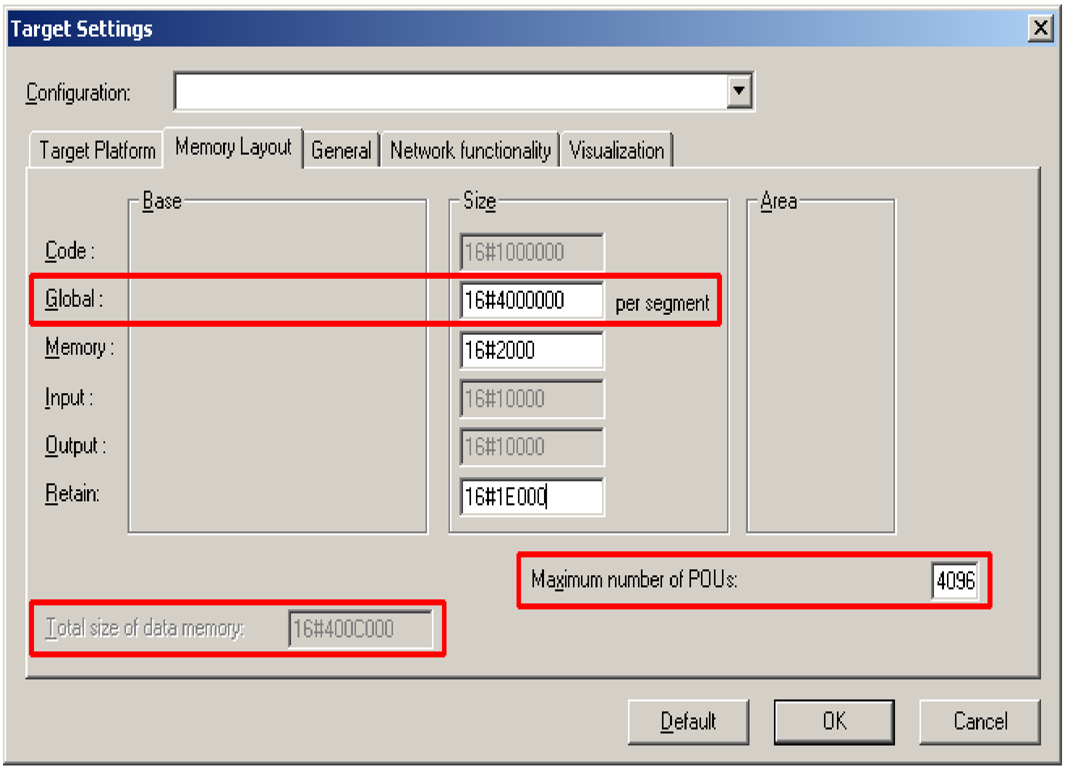
This set value has already been requested after a successfuly program download in the system and can be utilized to the full extent.

Together with the data memory to be used by the application, memory is required for the individual program function blocks in the system.

The size of the administration space is calculated from the function block limitation * 12 (i.e., normally 4096 *12).

The sum from the global data memory and function block limitation yields the actual size of the main memory required in the system for data.

This value should not exceed the value specified for “Size of entire data memory”.



Configuration:

Code : 16#1000000

Global : 16#4000000 per segment

Memory : 16#2000

Input : 16#10000

Output : 16#10000

Retain: 16#1E000

Maximum number of POUs: 4096

Total size of data memory: 16#400C000

Figure 77: Data memory and function block limitation

8.10.3 Remanent Memory

A total of 128 kbytes of remanent memory is available for the IEC-61131 application.

The remanent section is subdivided into the flag area (memory) and the retain area.

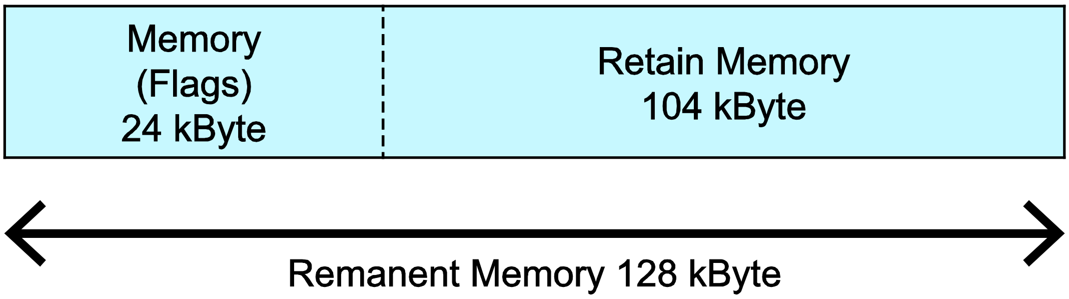
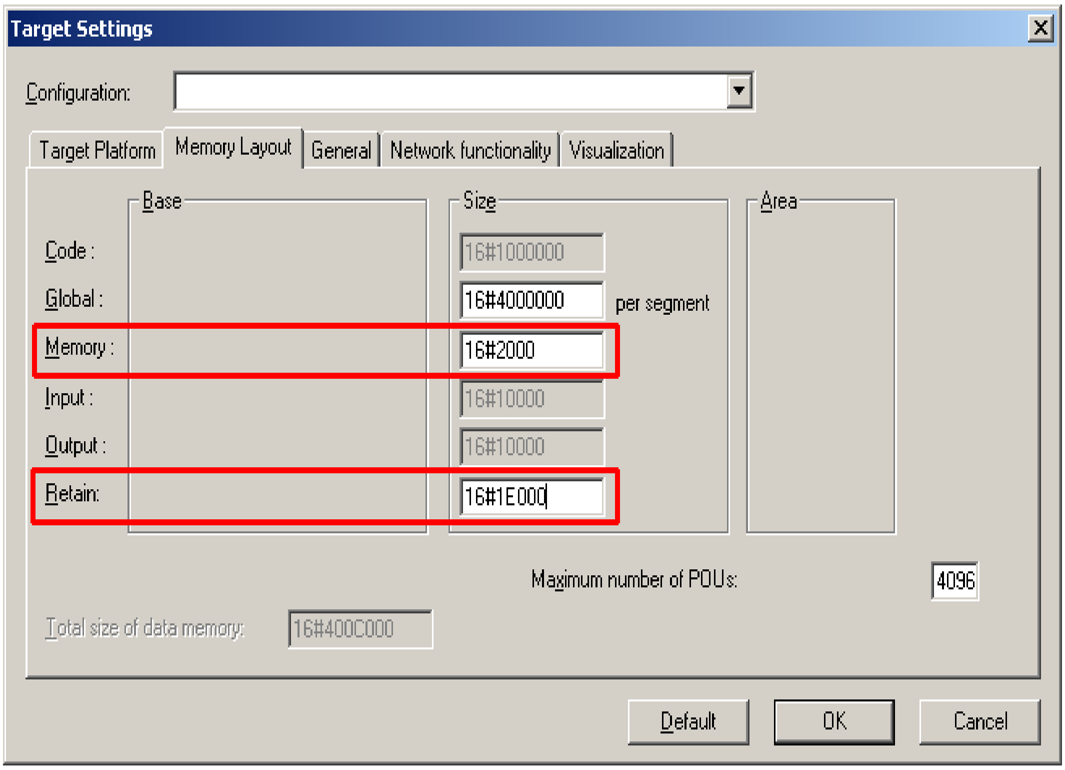


Figure 78: Remanent main memory

The breakdown of the flag and retain variables can be customized as required.

It must be ensured, however, that the sum from Memory + Retain does not exceed the maximum value of 128 kbytes (0x20000).



Configuration:

Code : 16#1000000

Global : 16#4000000 per segment

Memory : 16#2000

Input : 16#10000

Output : 16#10000

Retain: 16#1E000

Maximum number of POUs: 4096

Total size of data memory: 16#400C000

Figure 79: Flag and retain memory

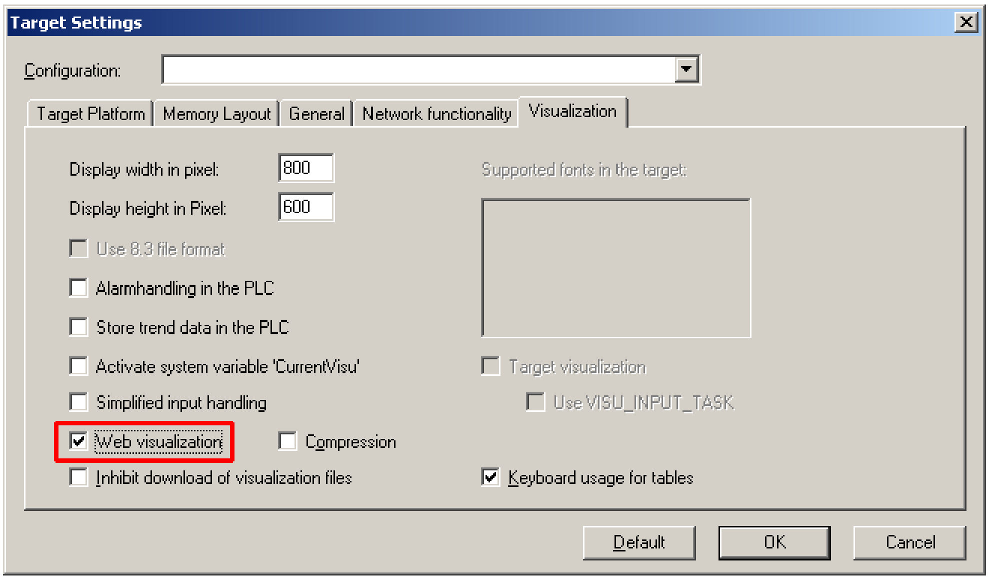
8.11 CODESYS Visualization

CODESYS Web visualization is based on Java technology. All Java programs require a Java runtime environment (JRE), which must be installed on the host PC along with an Internet browser. An applet is stored in the file system of a Web server and is accessible to browsers via an HTML page.

You create all visualization types (HMI and Web visualization) with the same CODESYS graphic editor. Select the visualization type in the “Target system settings” window. A description file in XML format is generated from the information for each of these pages. You can find these files in the subfolder *“visu”* of the CODESYS installation path. The HTML home page “webvisu.htm” and the Java archive “webvisu.jar” in the applet (webvisu.class) are also saved there in a compressed format.

Once you have selected a visualization type, the following steps must be performed to execute the technique:

1. Click the “Resources” tab and open the “Target system settings.” Specify whether you wish to have visualization displayed as a “Web visualization” using an Internet browser.



Configuration:

Display width in pixel: 800

Display height in Pixel: 600

☐ Use 8.3 file format

☐ Alarmhandling in the PLC

☐ Store trend data in the PLC

☐ Activate system variable 'CurrentVisu'

☐ Simplified input handling

☑ Web visualization ☐ Compression

☐ Inhibit download of visualization files

Supported fonts in the target:

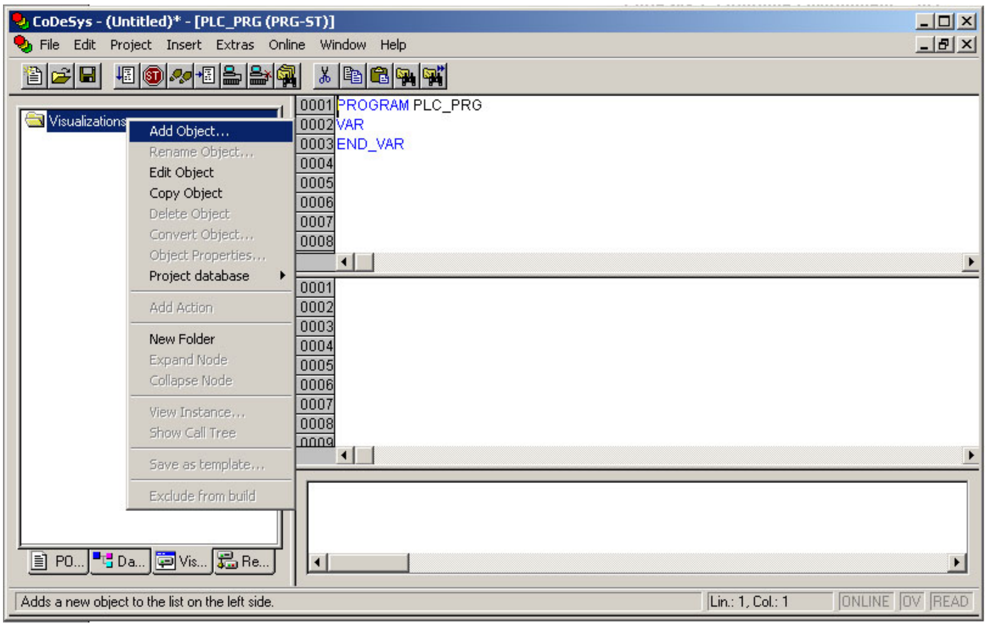
☐ Target visualization

☐ Use VISU_INPUT_TASK

☑ Keyboard usage for tables

Figure 80: Selecting the visualization technique in the target system settings

2. Generate a start page for the visualization. Right-click the “Visualization” folder in the “Visualization” tab. Select **Add object** ... from the contextual menu. The “New visualization” dialog box opens.



```
PROGRAM PLC_PRG
VAR
END_VAR
```

Adds a new object to the list on the left side.

Figure 81: Creating the PLC_VISU starting visualization

3. In the “New visualization” dialog window, enter the name **PLC_VISU** for the start visualization. This page is then displayed as the start page upon system startup.

4. Activate the CODESYS Web server in the WBM on the “Ports and Services – CODESYS Services” page in the “CODESYS Webserver” group.

5. Activate the http service in the WBM on the “Ports and Services – Network Services” page in the “HTTP” group.

If you transfer the PLC program to the controller (**Online > Login**) and start the program (**Online > Start**), enter one of the following lines in the address line of the Web browser for online visualization:

- “https://<IP address of the controller>/webvisu”, preferred method (http can also be used instead of https)

- “https://<IP address of the controller>”, if the default Web server in the WBM has been set to“WebVisu” (http can also be used instead of https)

- “http://<IP address of the controller>:8080/webvisu.htm”

You can also have Web visualization displayed via the WBM (see Section “`CODESYS - WebVisu´” Page).

Information

Frequently Asked Questions
Additional information (FAQs) on CODESYS Web visualization is also provided in the Section “Frequently Asked Questions about CODESYS Web Visualization” and in the online Help function for CODESYS 2.3.

8.11.1 Limits of CODESYS Visualization

The controller supports the “WebVisu” visualization type integrated into CODESYS. Technological limitations can be caused by the visualization type used.

Compared to “HMI”, Web visualization on the controller is performed within significantly narrower physical limits. Whereas “HMI” can access almost unlimited resources on a desktop PC, the following limitations must be observed when using Web visualization:

Adapting to the File System

The overall size of the PLC program, visualization files, bitmaps, log files, configuration files, etc. must fit into the file system.

Process Data Memory

Web visualization uses its own protocol for exchanging process data between applet and control.
The controller transfers process data with ASCII coding. The pipe symbol (“|”) is used to separate two process values. Therefore, the space requirement for a process data variable in the process data memory is dependent not only on the data type, but also on the process value itself. Thus, a variable of the “WORD” type occupies between one byte for the values 0 through 9 and five bytes for values from 10000 and greater. The selected format (ASCII + |) only permits a rough estimate of the space requirement for the individual process data in the process data buffer. If the size of the ASCII coded process data is exceeded, Web visualization no longer works as expected.

Computer Performance/Processor Time

The controller is based on a real-time operating system. This means that high-priority processes (e.g., PLC program) interrupt or block lower priority processes. The Web server responsible for Web visualization is among these lower priority processes.

Note

Processor Time
Make sure when configuring tasks, that there is sufficient processor time available for all processes.

Network Load

The controller's CPU processes both the PLC program and network traffic. ETHERNET communication demands that each received telegram is processed, regardless of whether it is intended for the controller or not.

A significant reduction of the network load can be achieved by using switches instead of hubs.

There is no measure against broadcast telegrams that can be used on the controller, however. These can only be curtailed by the sender, or blocked with configurable switches that have a broadcast limitation. A network monitor such as “wireshark” (www.wireshark.com) provides an overview of the current load in your network.

8.11.2 Eliminating Errors in CODESYS Web Visualization

If you are experiencing problems when working with the CODESYS Web visualization, use the following table to find the solution. If you cannot eliminate the problem, please contact WAGO support.

Table 88: Errors and remedies

Error	Solution
Internet Explorer reports the error "APPLET NOT INITIATED"	Close all Internet Explorer windows and restart. If the error persists, this indicates a missing or damaged file. Using FTP, check if the entire Java archive "webvisu.jar" is available in the "/PLC" folder of the controller. The original file can be found in the installation path of CODESYS (usually under *C:\Programme\WAGO Software\CODESYS V2.3\Visu\webvisu.jar*). If necessary, replace the damaged file using FTP or force the download of all files in CODESYS with **Purge All** > Compile **All** > **Log in**.
Web visualization is not displayed	Have you installed the JRE? Check the firewall settings, e.g., if port 8080 is open.
Web visualization "freezes". Web visualization stops after an extended period of time.	The call-up intervals selected in the task configuration are too small. As a result, the Web server of the controller — which is executed with a low priority — does not receive sufficient computer time, if any at all. If no (explicit) task configuration has been provided, the PLC_PRG is (implicitly) executed as a free running task with Priority 1. This significantly limits the Web server's computing time. Always provide a task configuration when using Web visualization. In doing so, the call-up interval should not exceed three times the average execution time. When determining the execution time, ensure that the PLC program has reached a "steady state." When determining the execution time, ensure that the PLC program is not "steady state."
Web visualization cannot be loaded into the controller	Not all files may fit into the controller's file system. Delete any unneeded data (e.g., via FTP).
Bitmap is not displayed	If the name of an image file contains umlauts, the Web server cannot interpret these image names.
Java console reports: "Class not found"	The JRE does not find the entry point for the class "webvisu.class" in the Java archive "WebVisu.jar". The Java archive is probably incomplete. Delete "WebVisu.jar" from the Java cache and/or deactivate the cache. In this case, the controller requests the archive (applet) again. If the problem persists, reload the project into the controller.
Web visualization is static, all process values are "0"	Process data communication has failed. If Web visualization is operated over a proxy server, then a SOCKS proxy is also necessary for process data exchange in addition to the actual HTTP proxy.

8.11.3 FAQs about CODESYS Web Visualization

How can I optimize the applet for special screen resolutions?

In order to optimize the Web visualization for display on a device with a fixed resolution, proceed as follows:
In the “Target system settings”, enter the pixel width and height in the tab “Visualization”. When the visualization is created, the visible area is highlighted in gray. However, the actual pixel width and height of the Web visualization is defined by the attributes “Height” and “Width” of the HTML APPLET tag in the “webvisu.htm” file. Do not forget to also adapt these parameters to the existing resolution.

Which JRE should I use?

Java2 standard edition Version 1.5.0 (J2SE1.5.0_06) or higher is recommended. This is available free of charge at www.oracle.com.
Microsoft's MSJVM3810 was also tested. For PDAs, there are runtime environments available from other manufacturers (JamaicaVM, CrEme, etc.). Please consider that for the Web visualization, these solutions can behave differently within their scope of services (e.g., stability) than those mentioned above.

Should the Java Cache be used?

This depends on the situation. After a standard installation, the cache is enabled. If the cache is enabled, the JRE uses it to store applets and Java archives. If the Web visualization is called up a second time, it requires considerably less time to start because the applet (approx. 250 kb) does not need to be reloaded via the network, but is already available in the cache. This is especially useful when network connections are slow.

Note:
The Java archives may not be completely transferred into the cache due to network failures. In this case, the cache must be cleared manually or disabled.

Why does the visualization element “TREND” in the Web visualization only work “Online”?

The following settings must be selected for visualization projects: **Resources** tab **> Target system settings**.
Activate “Web visualization” and “Trend data recording within control”. Otherwise, the trend data is stored on the hard drive of the CODESYS development PC. This makes a permanent connection between the controller and the CODESYS gateway necessary. If this connection is interrupted, this may lead to the controller behaving unpredictably.

In the TREND configuration dialog, you can choose between “Online” and “History” operating modes. The controller only supports the “Online” operating mode for visualization projects since it is not possible to configure the maximum size (quota) of the trend files (*.trd). Uncontrolled expansion of trend files can lead to unpredictable controller behavior.

In most cases, the use of the “HISTOGRAM” visualization element is the better choice, as this gives full control over the time and number of measurements and thus the amount of memory required.

What needs to be observed when the visualization element “ALARM TABLE” is used in the Web visualization?

The status of this component is best described as “Add-On”, i.e., an extra that is free of charge and not warrantied.

The following settings must be selected for visualization projects: **Resources** tab **> Target system settings**.
Activate “Web visualization” (checkmark) and “Alarm handling within control”. Otherwise, the alarm data is processed on the CODESYS development PC. This makes a permanent connection between the controller and the CODESYS gateway necessary. If this connection is interrupted, this may lead to the controller behaving unpredictably.

9 MODBUS

9.1 General

MODBUS is a non-vendor-specific, open fieldbus standard for a wide range of applications in production and process automation. The MODBUS communications protocol is based on a master/slave or client/server architecture that uses function codes for execution of individual MODBUS services, which have reading or writing access to individual or multiple elements of the MODBUS data model simultaneously.

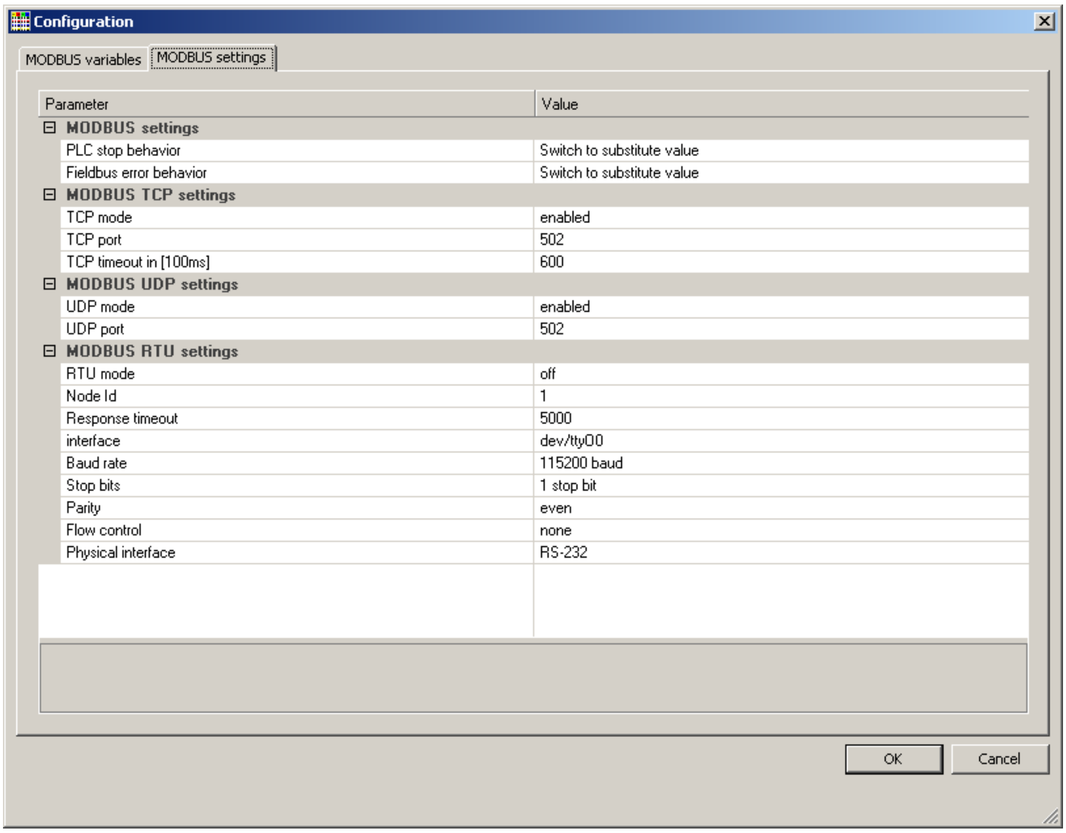
9.2 Features

The MODBUS slave implemented in the PFC200 has the following features:

- 3 modes: MODBUS TCP, MODBUS UDP and MODBUS RTU, which can be run independently of one another simultaneously
- Each mode can be configured
- 10 supported MODBUS services (Function Codes): FC1 to FC6, FC15, FC16, FC22, FC23
- Data exchange via 1000 registers in each of the local MODBUS process images
- 768-byte sector that can be addressed by bits in each local MODBUS process image
- Access to a 104-kB flag sector (total of 53248 registers/words, 3328 bits of which can be addressed)
- 28 Information and configuration registers
- Up to 1000 TCP connections
- MODBUS communications monitoring using programmable watchdogs
- Configurable response on PLC stop
- Configurable response on disruption of MODBUS communication

9.3 Configuration

All of the MODBUS operating modes are configured using the CODESYS control system configuration.



Parameter	Value
⊟ **MODBUS settings**	
PLC stop behavior	Switch to substitute value
Fieldbus error behavior	Switch to substitute value
⊟ **MODBUS TCP settings**	
TCP mode	enabled
TCP port	502
TCP timeout in [100ms]	600
⊟ **MODBUS UDP settings**	
UDP mode	enabled
UDP port	502
⊟ **MODBUS RTU settings**	
RTU mode	off
Node Id	1
Response timeout	5000
interface	dev/ttyO0
Baud rate	115200 baud
Stop bits	1 stop bit
Parity	even
Flow control	none
Physical interface	RS-232

Figure 82: CODESYS control system configuration - MODBUS settings

The MODBUS slave configuration is composed of four basic parameter groups:

- MODBUS settings,
- MODBUS TCP settings,
- MODBUS UDP settings,
- MODBUS RTU settings.

A detailed description of all the parameter groups is given in the following sections.

9.3.1 MODBUS Settings

The “MODBUS settings” group contains the following configuration parameters:

Table 89: MODBUS Settings

Parameters	Explanation	
PLC stop response	Response of the MODBUS slave when the control system has halted (control system in STOP state)	
	No data exchange	No data exchange possible. MODBUS requests will always be answered by the exception response "ILLEGAL FUNCTION" (0x81).
	Switch to substitute value*	Data exchange possible. Substitute values (0) are provided for MODBUS read requests and the values accepted unchanged in the local MODBUS process image for write requests, without passing these on to the control system.
	Hold last value	Data exchange possible. The last frozen values are provided for MODBUS read requests and the values accepted unchanged in the MODBUS process image for write requests, without passing these on to the control system.
Fieldbus error response	Response of the MODBUS slave on detected fieldbus errors (interruption of communication).	
	No data exchange	No data exchange possible.
	Switch to substitute value*	Data exchange possible. Substitute values (0) are supplied from the MODBUS process image for PLC read functions; for write access the values are accepted unchanged in the MODBUS process image without passing them on to the MODBUS master.
	Hold last value	Data exchange possible. The previously frozen values are supplied from the MODBUS process image for PLC read functions; for write access the values are accepted unchanged in the MODBUS process image without passing them on to the MODBUS master.

* Default setting

9.3.2 MODBUS TCP Settings

The “MODBUS TCP Settings” contains the following configuration parameters for the “MODBUS TCP” mode:

Table 90: MODBUS TCP Settings

Parameters	Explanation	
TCP mode	Enable for the MODBUS TCP mode	
	Off	Operation not permitted
	Active*	Operation possible
TCP port	Port number for the TCP link	
	1	Minimum port number
	502*	MODBUS standard port
	65535	Maximum port number
TCP time-out	Time-out for a TCP link	
	1	100 ms (1 x 100 ms)
	600*	60 seconds (600 x 100 ms)
	65535	1 h 49 min 13 s 500 ms (65535 x 100 ms)

* Default setting

9.3.3 MODBUS UDP Settings

The “MODBUS UDP Settings” group contains the following configuration parameters for the “MODBUS UDP” mode:

Table 91: MODBUS UDP Settings

Parameters	Explanation	
UDP mode	Enable for the MODBUS UDP mode	
	Off	Operation not permitted
	Active*	Operation possible
UDP port	Port number for the UDP link	
	1	Minimum port number
	502*	MODBUS standard port
	65535	Maximum port number

* Default setting

9.4 Data Exchange

MODBUS data exchange is performed in cycles or acyclically using MODBUS services. The type and number of usable MODBUS services depends on the area that is addressed. There are generally four MODBUS-relevant address areas in the PFC200:

- **MODBUS input process image** (MODBUS Input) – is an area in the PIO (PIO = Output Process Image), in which data from the PLC is provided in cycles exclusively for MODBUS Read services.
- **MODBUS output process image** (MODBUS Output) – is an area in the PII (PII = Input Process Image), in which MODBUS Write services provide data for cyclic reading by the PLC. MODBUS Read services are also acceptable in this area.
- **MODBUS flag area** – is an area, in which both MODBUS Read and Write services can be executed.
- **MODBUS register** – is an area, in which the WAGO-specific information and configuration registers are contained. Only MODBUS register services may be executed in this area.

9.4.1 Process Image

The main data interfaces between the PLC and the MODBUS slave are the local MODBUS process images in the PLC address area based on IEC 61131. The MODBUS input process image (MODBUS Input) is in the PIO and the MODBUS output process image (MODBUS Output) in the PII. Data memory blocks of 2 kB (1000 registers/word) are available for each local MODBUS input and output process image. The first 768 bytes of each of these data blocks are also provided for executing bit services.

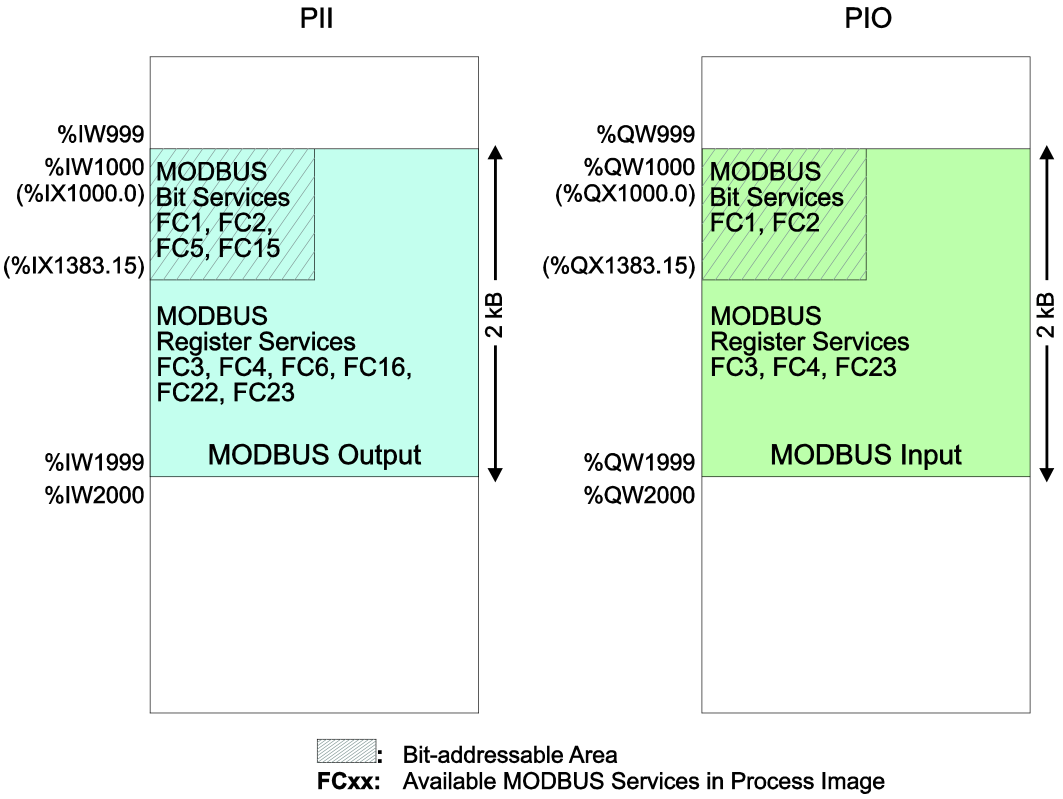


Figure 83: MODBUS process image

As no direct access to the I/O modules is provided by the fieldbus, data can be exchanged via this interface between the PLC and MODBUS for processing in the control system (PLC). Using this data in the individual I/O modules connected to the PLC can then be performed by the application.

9.4.2 Flag Area

MODBUS can also exchange data and fieldbus variables with the PLC via the flag area. Caution is urged, however, when using data and/or variables in this area that is accessed by both MODBUS and the PLC. This “conflicting” access is not protected from either side and could result in data inconsistency.

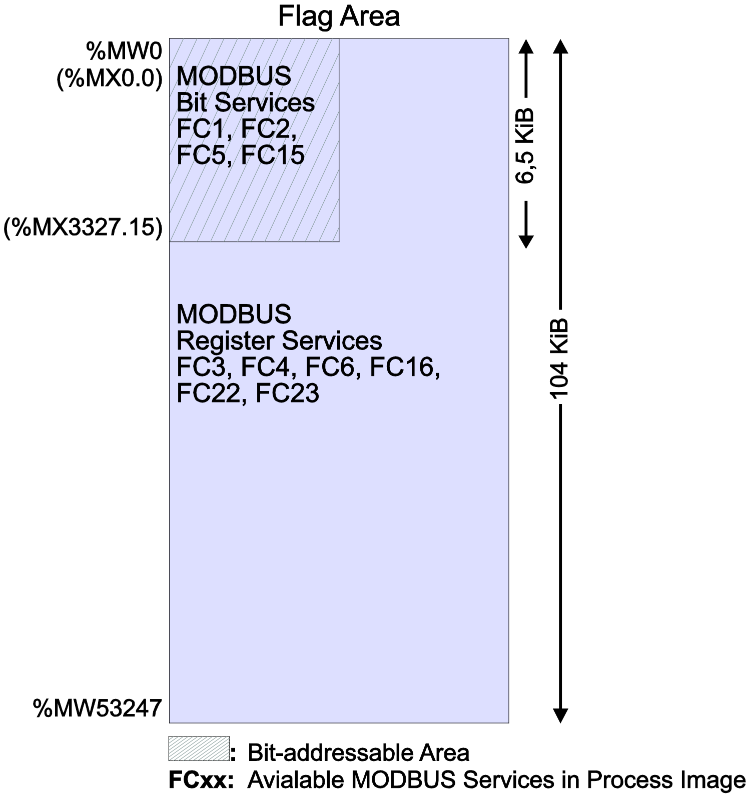


Figure 84: Flag area

9.4.3 MODBUS Registers

WAGO-specific registers are implemented in the last MODBUS-relevant address area; this simplifies the reading of certain system and MODBUS information, as well as configuration.

The MODBUS address area reserved for these registers ranging from the MODBUS starting address of 4096 (0x1000) up to the MODBUS end address of 12287 (0x2FFF), without any allocation to the IEC 61131 address area. These registers can be queried using the register read services FC3, FC4 and FC23 and with the register write services FC6, FC16, FC22 and FC23. A detailed description of the individual registers is given in the Section “WAGO MODBUS Registers”.

9.4.4 MODBUS Mapping

9.4.4.1 MODBUS Mapping for Write Bit Services FC1, FC2

The table below outlines the mapping for the MODBUS-reading, bit-oriented services:

- FC1 – Read Single Coil,
- FC2 – Read Discrete Inputs.

Table 92: MODBUS Mapping for Read Bit Services FC1, FC2

MODBUS Address (hexadecimal values in parentheses)	IEC 61131 Address	Description
0 … 6143 (0x0000 … 0x17FF)	%IX1000.0 … %IX1383.15	MODBUS Output: 6144 PFC input bit variables in the first 384 registers/words (768 bytes) of the 2kB MODBUS output process image in the PII. Note: In this area, the read bit services return the content from the bit-addressed PII.
6144 … 12287 (0x1800 … 0x2FFF)	%QX1000.0 … %QX1383.15	MODBUS Input: 6144 PFC output bit variables in the first 384 registers/words (768 bytes) of the 2 kB MODBUS-input process image in the PIO.
12288 … 65535 (0x3000 … 0xFFFF)	%MX0.0 … %MX3327.15	Flag area: 53248 bit flags (6.5 kB) in the bit-addressed flag area

9.4.4.2 MODBUS Mapping for Write Bit Services FC5, FC15

The table below outlines the mapping for the MODBUS-writing, bit-oriented services:

- FC5 – Write Single Coil
- FC15 – Write Multiple Coils

Table 93: MODBUS mapping for write bit services FC5, FC15

MODBUS Address (hexadecimal values in parentheses)	IEC 61131 Address	Description
0 … 6143 (0x0000 … 0x17FF)	%IX1000.0 … %IX1383.15	MODBUS Output: 6144 PFC input bit variables in the first 384 registers/words (768 bytes) of the 2kB MODBUS output process image in the PII.
6144 … 12287 (0x1800 … 0x2FFF)	~~%QX1000.0 …~~ ~~%QX1383.15~~	MODBUS Output: MODBUS-only area for bit-oriented write access. Bit-based write services for this area are acknowledged by the MODBUS slave with the MODBUS exception code “ILLEGAL DATA ADDRESS” (0x02).
12288 … 65535 (0x3000 … 0xFFFF)	%MX0.0 … %MX3327.15	Flag area: 53248 bit flags (6.5 kB) in the bit-addressed flag area

9.4.4.3 MODBUS Mapping for Read Register Services FC3, FC4, FC23

The table below outlines the mapping for the MODBUS-reading, register-oriented services:

- FC3 – Read Holding Registers,
- FC4 – Read Input Registers,
- FC23 – Read/Write Multiple Registers

Table 94: MODBUS Mapping for Read Register Services FC3, FC4, FC23

MODBUS Address (hexadecimal values in parentheses)	IEC 61131 Address	Description
0 … 999 (0x0000 … 0x03E7)	%IW1000 … %IW1999	MODBUS Output: 1000 PFC input registers/words in the 2 kB MODBUS output process image in the PII. Note: In this area, the read register services return the content from the PII.
1000 … 1999 (0x03E8 … 0x07CF)	%QW1000 … %QW1999	MODBUS Input: 1000 PFC output registers/words in the 2 kB MODBUS input process image in the PIO. Note on FC23: Only the Read portion of this service can be executed.
2000 … 4095 (0x07D0 … 0x0FFF)		Inhibited to MODBUS-only area for register-oriented read access. Register-based read services for this area are acknowledged by the MODBUS slave with the MODBUS exception code "ILLEGAL DATA ADDRESS" (0x02).
4096 … 12287 (0x1000 … 0x2FFF)	No IEC 61131 address	Information and configuration registers: Not all MODBUS addresses in this range are valid. Valid MODBUS addresses are described in the Section "WAGO MODBUS Registers". Access to invalid addresses are acknowledged by the MODBUS slave with the MODBUS exception code "ILLEGAL DATA ADDRESS" (0x02). Note on FC23: The Write portion of this service can only be executed for registers that data can be written to.

Table 94: MODBUS Mapping for Read Register Services FC3, FC4, FC23

MODBUS Address (hexadecimal values in parentheses)	**IEC 61131 Address**	**Description**
12288 … 65535 (0x3000 … 0xFFFF)	%MW0 … %MW53247	Flag Area: 53248 register/word flags (104 kB) in the flag area

9.4.4.4 MODBUS Mapping for Write Register Services FC6, FC16, FC22, FC23

The table below outlines the mapping for MODBUS-writing, register-oriented services.

- FC6 – Write Single Register,
- FC16 – Write Multiple Registers,
- FC22 – Mask Write Register,
- FC23 – Read/Write Multiple Registers.

Table 95: MODBUS mapping for write register services FC6, FC16, FC22, FC23

MODBUS Address (hexadecimal values in parentheses)	IEC 61131 Address	Description
0 … 999 (0x0000 … 0x03E7)	%IW1000 … %IW1999	MODBUS Output: 1000 PFC input registers/words in the 2 kB MODBUS output process image in the PII.
1000 … 1999 (0x03E8 … 0x07CF)	No access to: %QW1000 … %QW1999	MODBUS Output: Inhibited MODBUS area for register-oriented write access. Register-oriented write services in this area are acknowledged by the MODBUS slave with the MODBUS exception code “ILLEGAL DATA ADDRESS” (0x02).
2000 … 4095 (0x07D0 … 0x0FFF)		Inhibited MODBUS area for register-oriented write access. Register-oriented write services in this area are acknowledged by the MODBUS slave with the MODBUS exception code “ILLEGAL DATA ADDRESS” (0x02).
4096 .. 12287 (0x1000 … 0x2FFF)	No IEC 61131 address	Information and Configuration Registers: Not all MODBUS addresses in this area are valid and not all registers can be written to. Valid MODBUS addresses are described in the Section “WAGO MODBUS Registers”. Access to invalid addresses are acknowledged by the MODBUS slave with the MODBUS exception code “ILLEGAL DATA ADDRESS” (0x02).

Table 95: MODBUS mapping for write register services FC6, FC16, FC22, FC23

MODBUS Address (hexadecimal values in parentheses)	IEC 61131 Address	Description
12288 … 65535 (0x3000 … 0xFFFF)	%MW0 … %MW53247	Flag Area: 53248 register/word flags (104 kB) in the flag area

9.5 WAGO MODBUS Register

System and MODBUS data can be read and some MODBUS parameters configured using the WAGO MODBUS registers. The following table lists all of the WAGO MODBUS registers.

Table 96: WAGO MODBUS Registers

MODBUS Address		Data Length in Words	Access	Description
Dec.	Hex.			
4130	0x1022	1	ro	Number of registers in the MODBUS input process image in the PIO
4131	0x1023	1	ro	Number of registers in the MODBUS output process image in the PII
4132	0x1024	1	ro	Number of bits in the MODBUS input process image in the PIO
4133	0x1025	1	ro	Number of bits in the MODBUS output process image in the PII
4136	0x1028	1	ro	IP configuration: BootP(1), DHCP(2) or fixed, coded IP address(4)
4138	0x102A	1	ro	Number of established TCP connections
4144	0x1030	1	r/w	MODBUS TCP Timeout (Changes apply only to new connections)
4145	0x1031	3	ro	MAC ID of the ETHERNET interface (eth0)
4151	0x1037	1	r/w	MODBUS TCP response delay
4160	0x1040	1	ro	PLC status
4352	0x1100	1	wo	Watchdog Command
4353	0x1101	1	ro	WatchdogStatus
4354	0x1102	1	rw	Watchdog Timeout (configuration register)
4355	0x1103	1	rw	Watchdog Config (configuration register)
8192	0x2000	1	ro	0x0000 (constant)
8193	0x2001	1	ro	0xFFFF (constant)
8194	0x2002	1	ro	0x1234 (constant)
8195	0x2003	1	ro	0xAAAA (constant)
8196	0x2004	1	ro	0x5555 (constant)

Table 96: WAGO MODBUS Registers

MODBUS Address		Data Length in Words	Access	Description
Dec.	Hex.			
8197	0x2005	1	ro	0x7FFF (constant)
8198	0x2006	1	ro	0x8000 (constant)
8199	0x2007	1	ro	0x3FFF (constant)
8200	0x2008	1	ro	0x4000 (constant)
8208	0x2010	1	ro	Revision (Firmware Index)
8209	0x2011	1	ro	Series code
8210	0x2012	1	ro	Device code
8211	0x2013	1	ro	Major Firmware Version
8212	0x2014	1	ro	Minor Firmware Version
8213	0x2015	1	ro	MBS Version

The WAGO MODBUS registers are described in more details in the following sections.

9.5.1 Process Image Properties

9.5.1.1 Register 0x1022 – Number of Registers in the MODBUS Input Process Image

This register contains the number of registers available in the MODBUS input process image (MODBUS input).

9.5.1.2 Register 0x1023 – Number of Registers in the MODBUS Output Process Image

This register contains the number of registers available in the MODBUS output process image (MODBUS output).

9.5.1.3 Register 0x1024 – Number of Bits in the MODBUS Input Process Image

This register contains the number of bits available in the MODBUS input process image (MODBUS input).

9.5.1.4 Register 0x1025 – Number of Bits in the MODBUS Output Process Image

This register contains the number of bits available in the MODBUS output process image (MODBUS output).

9.5.2 Network Configuration

9.5.2.1 Register 0x1028 – IP Configuration

This register contains information about the set IP configuration.
Possible values:

1 = BootP
2 = DHCP
4 = Fixed IP address

9.5.2.2 Register 0x102A – Number of Established TCP Connections

This register supplies the number of established TCP connections.
The maximum number of MODBUS TCP connections is 1000.

9.5.2.3 Register 0x1030 – MODBUS TCP Socket Timeout

This register contains the timeout value for the TCP sockets.
This value is given in units of 100ms (ticks). A new value is accepted only for new connections which have not yet been established. In the event of any changes, pre-established connections will continue to operate using the previously set timeout value.

9.5.2.4 Register 0x1031 – MAC Address for ETHERNET-Interface 1 (eth0)

This register provides the MAC address for the first ETHERNET interface (eth0).
MAC may also provide a partial result.

9.5.2.5 Register 0x1037 - MODBUS TCP Response Delay

This register saves the value of the MODBUS response delay.
This value is specified in ms units. The maximum delay is 32 ms, default value is 0 ms (no delay). Transmission of the response to a MODBUS request is delayed from the time of processing (read and/or write register values) by the time set. In the meantime, incoming requests can only be processed when the previous response is sent. For MODBUS UDP, this applies to all requests and for MODBUS TCP, for each connection. The actual length of time between a MODBUS request and the associated response depends on the number of parallel requests overall system utilization; it is always greater than the response delay set. Changes to the response delay become effective immediately for each subsequent request.

9.5.3 PLC Status Register

Register 0x1040 provides the status (state) that the controller is currently in. Possible values:

1 = PLC running − PLC status is RUNNING.
2 = PLC stopped − PLC status is STOPPED.

9.5.4 MODBUS Watchdog

The MODBUS watchdog monitors ongoing MODBUS communication. In the event that the configured monitoring time (see “Watchdog Timeout” Register) is exceeded without one single valid MODBUS query (trigger) being received from a MODBUS slave, the watchdog reacts and initiates the “Watchdog Timeout” reaction (previously configured in the “Watchdog Config” register). The watchdog must be continuously triggered by the MODBUS master to ensure that the set monitoring time does not expire. Triggering is performed by receiving any arbitrary MODBUS query from the entire volume of services supported by the MODBUS slave. These MODBUS-slave-supported services are listed in the Section “MODBUS Mapping”. The Explicit Trigger Mode is an exception to this rule and is elucidated in detail in the description of the “MODBUS Config” register (0x1103).

9.5.4.1 Register 0x1100 – Watchdog Command

This register receives commands for the MODBUS watchdog.
The following commands are accepted:

Table 97: Watchdog Commands

Value	Name		Explanation
0x5555	WATCHDOG_START		Starts the watchdog
	Error-free response		Watchdog has been successfully started, or restarted, and has the status “Running”.
	Error response	ILLEGAL FUNCTION (0x01)	Watchdog time has already expired. The watchdog must be reset with the command WATCHDOG_RESET.
		ILLEGAL DATA VALUE (0x03)	Watchdog not configured, i.e., the “Watchdog Timeout” register (0x1102) contains the value 0.
0x55AA	WATCHDOG_STOP		Stops the watchdog
	Error-free response		Watchdog was stopped successfully and has the status “Stopped”.
	Error response	ILLEGAL FUNCTION (0x01)	Watchdog monitoring time has already expired. The watchdog must then first be reset by the command WATCHDOG_RESET.
		ILLEGAL DATA VALUE (0x03)	Watchdog not configured, i.e., the “Watchdog Timeout” register (0x1102) contains the value 0.
0xAAAA	WATCHDOG_RESET		Resets the watchdog after a timeout
	Error-free response		Watchdog was reset successfully and has the status “Stopped” or “Unconfigured” when the “Watchdog Timeout” register (0x1102) contains the value 0.
	Error response	ILLEGAL DATA VALUE (0x03)	Watchdog does not have the status “Expired”.

The watchdog is only started by the command WATCHDOG_START. A prerequisite for successful startup is a valid timeout value in the “Watchdog Timeout” register (0x1102) and the status “Stopped” or “Running” (for restart), which the watchdog must have (see “Watchdog Status” register 0x1101). A response of ILLEGAL_DATA_VALUE indicates that the watchdog has not yet been configured (status “Unconfigured”, timeout value is “0”). If the watchdog has the status “Expired”, the WATCHDOG_START command is also rejected by the ILLEGAL_FUNCTION acknowledgement.

A running watchdog can be stopped by the command WATCHDOG_STOP. A stop request received several times in a row does not have any impact on the behavior of the watchdog and is therefore not acknowledged with an error response. By contrast, a stop command received when the watchdog has the status “Unconfigured” is rejected with the acknowledgement ILLEGAL_DATA_VALUE; if the watchdog has the status “Expired” this command is rejected by the exception code ILLEGAL_FUNCTION.

If the watchdog monitoring time has expired, the watchdog can only be reset by the command WATCHDOG_RESET. The watchdog cannot be restart until this command is executed. Resetting the watchdog does not necessarily restart it automatically, however. To restart the watchdog after a reset, the command WATCHDOG_START must be sent. Possible reactions to expiring of the watchdog monitoring time (timeout) are described in the “Watchdog Config” register (0x1102). If the command WATCHDOG_RESET is received when the watchdog has a status other than “Expired”, this command is rejected with the exception code ILLEGAL_DATA_VALUE.

9.5.4.2 Register 0x1101 – Watchdog Status

This register provides the current status of the MODBUS watchdog.
The watchdog can have any of the four following statuses:

Table 98: Watchdog Status

Value	Name	Explanation
0xFFFF	WATCHDOG_UNCONFIGURED	Watchdog not configured, i.e., the watchdog time-out register (0x1102) contains the value 0.
0x0000	WATCHDOG_STOPPED	Watchdog is not active (not started).
0x0001	WATCHDOG_RUNNING	Watchdog is active (started).
0x0002	WATCHDOG_EXPIRED	Watchdog monitoring time has expired.

If the “Watchdog Timeout” register (0x1102) contains the value 0, the Watchdog has not been configured − WATCHDOG_UNCONFIGURED. If the timeout value remains unchanged (0), the Watchdog cannot leave this state. After a configuration, the Watchdog switches to “Stopped” and can now be started.

The watchdog can only be started by the command WATCHDOG_START. When the watchdog is started, it assumes the status “Running” - WATCHDOG_RUNNING. Configuration changes in this state are no longer possible.

When the watchdog expires (status WATCHDOG_EXPIRED), the registers “Watchdog Status” (0x1101), “Watchdog Timeout” (0x1102) and “Watchdog Config” (0x1103) are the only registers that can be read. Access to other registers, with the exception of write access to the “Watchdog Command” register (0x1100), and bit access is acknowledged by the error ILLEGAL FUNCTION.

The configuration registers can only be written in the watchdog statuses WATCHDOG_UNCONFIGURED, WATCHDOG_STOPPED or WATCHDOG_EXPIRED. Only read access is permitted for these registers in the status WATCHDOG_RUNNING.

9.5.4.3 Register 0x1102 – Watchdog Timeout

This configuration register contains the watchdog timeout value. As the basic unit is 100 ms, the timeout value must be a multiple of 100 ms. Thus, the minimum timeout value can be set to 100 ms, while the maximum value can be set to 6553.5 seconds. If this value is 0, the watchdog cannot be started and will have the status

“Unconfigured”. The watchdog timeout register can be rewritten to the states “Unconfigured”, “Stopped” or “Expired”. Access to this register is read only while the watchdog is active.

9.5.4.4 Register 0x1103 – Watchdog Config

This register contains the configuration parameters for the watchdog.
The following parameters can be configured:

Table 99: Watchdog Configuration

Bit	Name/Bit Identifier	Explanation	
0	Explicit triggering EXPLICIT_TRIGGER_ONLY	Controls explicit triggering	
		0	All supported MODBUS queries are considered watchdog triggers (default setting). There is a special rule for the “Watchdog Status” register. See the description for bit 1 EXPLICIT_TRIGGER_ON_STATUS_REG.
		1	Explicit Trigger Mode − only the command WATCHDOG_START (0x5555), sent to the “Watchdog Command” register (0x1100), or reading of the “Watchdog Status” registers (0x1101), when Bit 1 EXPLICIT_TRIGGER_ON_STATUS_REG has been set accordingly, are considered valid trigger events for the MODBUS watchdog.
1	Triggering by access to the watchdog status register TRIGGER_ON_STATUS_REG	Controls triggering by read-only access to the “Watchdog Status” register	
		0	Read access to the “Watchdog Status” register is not considered a trigger event (default setting).
		1	Read access to the “Watchdog Status” register is considered a trigger event.
2	Close-down of established TCP connections CLOSE_ALL_TCP_CONNECTIONS	Closes down all established TCP connections	
		0	All established connections remain open.
		1	All established connections are closed (default setting).

The individual options are activated when the specific bit, or bit combination, is set.

The “Watchdog Config” register can be rewritten to the statuses “Unconfigured”, “Stopped” or “Expired”. Access to this register is read only while the watchdog is active.

9.5.5 MODBUS Constants Register

Registers 0x2000 … 0x2008 provide constants based on the table “WAGO MODBUS Register”. It is possible to read all of the constants, or a consecutive portion of them, at once.

9.5.6 Electronic Nameplate

Registers 0x2010 to 0x2015 contain information from the electronic nameplate. It is possible to read the entire nameplate, or a consecutive portion of it, all at once.

9.5.6.1 Register 0x2010 – Revision (Firmware Index)

This register provides the consecutive revision index (firmware index) for the PFC200.

Example: 5 for Version 5.

9.5.6.2 Register 0x2011 – Series Designator

This register provides the designation for the WAGO series (Series Code) of the PFC200.

Example: 750 for WAGO-I/O SYSTEM 750.

9.5.6.3 Register 0x2012 – Device ID

This register provides the device ID (WAGO Item No.) of the controller.

Example: 8206.

9.5.6.4 Register 0x2013 – Major Firmware Version

This register provides the major part for the firmware version.

9.5.6.5 Register 0x2014 – Minor Firmware Version

This register provides the minor part for the firmware version.

9.5.6.6 Register 0x2015 – MBS Version

This register provides the version of the MODBUS slave library. The high byte contains the major version number and the low byte, the minor version number.

Example:
0x010A => Major version number = 1, Minor version number = 10.

9.6 Diagnostics

9.6.1 Diagnostics for the MODBUS Master

The status of the PLC, or of the control system, can be queried by the MODBUS master by reading the WAGO-specific register 0x1040 – “PLC Status” using MODBUS services FC3 (Read Holding Registers) or FC4 (Read Input Registers). The WAGO-specific register 0x1040 – “PLC Status” is explained in the Section “PLC Status Registers”.

The status of the MODBUS Watchdog can be requested using a register service (FC3 or FC4) with a query to the WAGO-specific register 0x1101 – “Watchdog Status Register”. Information about this is given in the Section “MODBUS Watchdog”.

The MODBUS service “Get Communication Event Counter” (FC11) is not supported in the current MODBUS slave Version V1.0.

9.6.2 Diagnostics for the Runtime System

Diagnostics for the MODBUS slaves can be executed by integrating the CODESYS library “BusDiag.lib” via the runtime system. The required function block, “DiagGetBusState() indicates the status of the fieldbus (here MODBUS) and is located in this library. Details about this function block are provided both in this document and in the online Help function for CODESYS.

9.6.3 Diagnostics for the Error Server

The MODBUS slave also supports the error service implemented in the PFC and generates diagnostic messages, which are stored permanently (in a file), or temporarily (in the RAM) and can be displayed directly via the WBM client. The following diagnoses are generated by the MODBUS slave:

Table 100: Diagnostics for the Error Server

Diagnostics ID	Diagnostic text	Method of saving	Explanation
0x00090000	Modbus Slave library loaded	Temporary	MODBUS slave library has been successfully loaded.
0x00090001	Modbus Slave library closed	Temporary	MODBUS slave library has been successfully unloaded.
0x00090002	Modbus Slave TCP started	Temporary	MODBUS slave successfully started in TCP mode.
0x00090003	Modbus Slave TCP start failed	Permanent	Starting the MODBUS slave in the TCP mode failed.
0x00090004	Modbus Slave TCP terminated	Temporary	MODBUS slave TCP mode successfully terminated.
0x00090005	Modbus Slave UDP started	Temporary	MODBUS slave successfully started in UDP mode.
0x00090006	Modbus Slave UDP start failed	Permanent	Starting the MODBUS slave in UDP mode failed.

Table 100: Diagnostics for the Error Server

Diagnostics ID	Diagnostic text	Method of saving	Explanation
0x00090007	Modbus Slave UDP terminated	Temporary	MODBUS slave UDP mode successfully terminated.
0x00090008	Modbus Slave RTU started	Temporary	MODBUS slave successfully started in the RTU mode.
0x00090009	Modbus Slave RTU start failed	Permanent	Starting the MODBUS slave in RTU mode failed.
0x0009000A	Modbus Slave RTU terminated	Temporary	MODBUS slave RTU mode successfully terminated.
0x0009000B	Modbus Slave data exchange started by PLC	Temporary	MODBUS slave data exchange started.
0x0009000C	Modbus Slave data exchange stopped by PLC	Temporary	MODBUS slave data exchange stopped.
0x0009000F	Modbus Slave PLC watchdog timer expired	Permanent	Monitoring time for controller (PLC) expired.
0x00090100	Modbus Slave common configuration failed.	Permanent	MODBUS slave configuration failed.
0x00090101	Modbus Slave TCP configured successfully.	Temporary	MODBUS slave TCP configuration completed successfully.
0x00090102	Modbus Slave TCP configuration failed.	Permanent	MODBUS slave TCP configuration failed.
0x00090103	Modbus Slave UDP configured successfully	Temporary	MODBUS slave UDP configuration completed successfully.
0x00090104	Modbus Slave UDP configuration failed.	Permanent	MODBUS slave UDP configuration failed.
0x00090105	Modbus Slave RTU configured successfully.	Temporary	MODBUS slave RTU configuration completed successfully.
0x00090106	Modbus Slave RTU configuration failed	Permanent	MODBUS slave RTU configuration failed.
0x00090107	Port for Modbus Slave RTU operation not free.	Permanent	Serial port for MODBUS slave RTU configuration already occupied.

Table 100: Diagnostics for the Error Server

Diagnostics ID	Diagnostic text	Method of saving	Explanation
0x00090108	Modbus Slave RTU configuration in RS-485 mode failed.	Permanent	MODBUS slave RTU configuration for the RS-485 mode has failed.
0x00090200	Modbus Slave Watchdog activated.	Temporary	MODBUS watchdog activated.
0x00090201	Modbus Slave Watchdog deactivated.	Temporary	MODBUS watchdog deactivated.
0x00090202	Modbus Slave Watchdog Timer expired.	Permanent	MODBUS watchdog monitoring time expired.
0x00090203	Modbus Slave has terminated all established TCP connections.	Permanent	All MODBUS TCP connections terminated due to timeout.
0x00090300	Modbus Slave: obtaining system resource failed	Permanent	Request for system resources by the MODBUS slave has failed.
0x00090301	Modbus Slave: processing system resource failed.	Permanent	Access to system resources by the MODBUS slave has failed.

10 CANopen Master and Slave

Based on IEC 61131-3 programming, data processing occurs on site in the controller. The process results can be output directly to the actuators, or transmitted via the bus.

Process data is exchanged with PDOs and SDOs. The controller supports 512 TX PDOs and 512 RX PDOs and SDOs to send process data via the CANopen fieldbus.

In the local process image, a range of 4000 bytes serves as each input and output range for data exchange via the CANopen interface. This range is situated between the addresses 6000 to 9999. Direct access to the I/O modules via the fieldbus is not provided.

All entries of the object directory can be mapped as required to the RX PDOs and TX PDOs. The complete input and output process image can be transmitted via SDOs.

10.1 Object Directory

All communication objects and all user objects are compiled in the object directory. The figure below provides a rough overview of this:

Table 101: Overview of addresses in the object directory

Index Range	Use
0000	Not used
0001-009F	Data types
00A0-0FFF	Reserved (addresses used for other services)
1000-1FFF	Communication profile
2000-5FFF	Vendor-specific range
6000-9FFF	Up to eight standardized device profiles
A000-AFFF	Process images from IEC 61131 devices
B000-BFFF	Process images from CANopen gateways acc. CiA 302-7
C000-FFFF	Reserved

The objects, which are made available by the controller, are described below.

10.2 Communications Profile

0x1000 Device Type

The stack responds on the bus as the DS-405 device (IEC 61131-3 programmable device), regardless of being configured as the master or slave. As direct access to the I/O modules via the bus is prohibited, the bits for information about inputs and outputs are 0.

Entry 0x000191 = DS 405 for master and slave

0x1001 Error Register

This entry contains an 8-bit item of information about the error status. At present, bit 4 is used specifically for communication and bit 5 for the device profile. Bit 0 is set for each error.

0x1003 Pre-defined Error Field

This entry contains the list of accumulated errors which were signaled in error register 0x1001. Sub-index 0 contains the number of entries. If a new error occurs, it is added to sub-index 1 and all existing errors are moved down one sub-index. A maximum of 20 error entries is supported. If more than 20 errors occur, the error at sub-index 20 is overwritten. By writing a “0” into sub-index 0, the complete error memory is deleted.

Standard values: 0 in all entries

0x1005 COB IB Sync

This objects defines the COB ID for the synchronization message.

Default: 0x80

0x1006 Communication Cycle Period

The duration of the synchronization cycle given in µs, or 0 for cyclic synchronization. Internal resolution is 1 ms. If this value is 0, SYNC monitoring does not occur.

Default: 0

0x1008 Manufacturer Device Name

This object specifies the device name.

Entry: Item No. for the PFC200, e.g., “750-8206”

0x1009 Manufacturer Hardware Version

Entry: “V 1.0” or higher

0x100A Manufacturer Software Version

Entry: “V 1.00” or higher

0x100C Node Guarding 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 “Guarding Time.”

Default: 0 (Node guarding disabled)

0x100D Life Time Factor

The “Life Time Factor” is part of the node guarding protocol. The NMT slave checks whether it was queried within the node lifetime (guarding time multiplied by the lifetime factor). If not, the slave must assume that the NMT master is no longer in normal operation; it then initiates a “life guarding event”.

Default: 0 (Node guarding off)

0x1012h COB-ID Time Stamp Object

The time stamp object enables every device's clock on the bus to be synchronized. The ID for this object is indicated here. Although the synchronization signal is not evaluated by the runtime, it may be used with library functions.

Default: 0x100 (Time Stamp Consumer)

0x1014h Emergency COB ID

An emergency message is transmitted in the event of CANopen device errors. The ID for this object is indicated here.

Default: 0x80 + Device ID

0x1015h Emergency Inhibit Time

This object specifies the minimum time that must elapse before another emergency object is sent. An entry equal to zero disables delayed sending. One time unit amounts to 100µs.

Default: 0

0x1016h Consumer Heartbeat Time

This entry can be used for monitoring of other devices on the bus. A check is made to determine whether each module defined in this object has generated 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. If the time is 0, monitoring is deactivated. The number of devices to be monitored is entered in index 0, the heartbeat time is entered in ms in the bottom 16 bits and the ID of the bus device in the 8 bits above that.

Default:

Index 0: 0 (currently still 127 = Number of possible entries)

All other entries are 0 (this function is not yet supported by the CAN master in Firmware 1.0).

0x1017h Producer Heartbeat Time

This object defines the time (in milliseconds) between two transmitted heartbeat messages. No heartbeat is sent if the time is set to 0.

Default: 0

0x1200, 0x1201 Server SDO Parameter Channels

The communication parameters for an SDO as the server are entered here. Two server SDO channels are supported.

0x1280 … 0x128E Client SDO Parameter Channels

The communication parameters for an SDO transfer as the client are entered here. 16 client SDO channels are supported.

0x1018h Identity

This object specifies the device being used. The manufacturer ID contains a unique number for each vendor. WAGO has been assigned an ID of 33. The device description reflects the family of products used.

The Rev. No. contains a specific CANopen behavior. The Major Rev. No. contains the CANopen functionality. If the functionality is changed, the Major Rev. No. is increased. You can use the Minor Rev. No. to distinguish between different versions with the same CANopen behavior.

Sub-index 0 No. of entries:	4
Sub-index 1 vendor ID:	33
Sub-index 2 product_code:	e.g., 8206 for 750-8206
Sub-index 3 revision_number:	0x00010001 or higher
Sub-index 4 serial_number	corresponds to the last 4 bytes of the MAC address.

0x1029h Error Behavior

This object defines how the slave responds in the event of an error.

Sub-Index Sub-index 0 No. of entries: 1

Sub-Index 1 Communication Error:

1	No change (Standard)
0	Change from operational to preoperational
2	Change to stop

0x1F51 Program Control

The status of the PLC can be read out using this object. Writing is prohibited.

Entries: 0 = Stop 1 = Run 2 = Reset 3 = Clear

10.2.1 Master Configuration

These objects are only available at the bus end when the master has been configured.

0x102A NMT Inhibit Time

This object indicates the minimum time that must elapse before another NMT telegram is sent. An entry equal to zero deactivates delayed sending. One unit of time is 100 µs.

Default: 0

0x1F80 NMT Start-up

This object contains the configuration bits for the master status. If automatic startup is deactivated, the master can be started by writing of 0x1F to this object.

0x1F81 … 0x1F8A Slave Configuration

The configured slaves are entered in these lists. All of the entries are checked when the master is started and transferred to the slaves.

0x1F81 NMT Slave-Assignment

Sub-Index 0: 127 = Number of possible entries

Sub-Index 1 … 127	Bit 0: Slave available
	Bit 2: Slave required for start
	Bit 3: Slave reset performed on start
	Bit 8 … 15: Guard Retry Factor
	Bit 16 … 31: Guard Time

0x1F82 Request-NMT

Sub-Index 0: 127 = Number of possible entries

Sub-Index = Master Node ID NMT state of the master

0x1F84 Device Type Identification

Sub-Index 0: 127 = Number of possible entries

Sub-Index 1 … 127: Slave device type

0x1F85 Vendor Identification

Sub-Index 0: 127 = Number of possible entries

Sub-Index 1 … 127: Slave device type (not used by default)

0x1F86 Product Code

Sub-Index 0: 127 = Number of possible entries

Sub-Index 1 … 127: Slave device type (not used by default)

0x1F87 Revision number

Sub-Index 0: 127 = Number of possible entries

Sub-Index 1 … 127: Slave device type (not used by default)

0x1F88 Serial Number

Sub-Index 0: 127 = Number of possible entries

Sub-Index 1 … 127: Slave device type (not used by default)

0x1F89 Boot Time

Time in ms between the start of slaves and operational readiness of all slaves.

Default: 0 = deactivated

0x1F8A Restore Configuration

Sub-Index 0: 127 = Number of possible entries

Sub-Index 1 … 127: Bit 0 = 1 Send restore configuration to slave on start

10.3 Data Exchange

Process data exchange occurs via the communication objects with the CANopen fieldbus controller.

Each object consists of a CAN telegram with a maximum of 8 bytes process data and a COB (Communication Object Identifier) ID that is unique within the network.

These communication objects transmit data, trigger events, signal error statuses, etc.

The parameters required for the communication objects, as well as CANopen device parameters and data are stored in an object directory.

10.3.1 Controller Communication Objects

The PFC200 supports the following communication objects:

512 Tx-PDOs for process data exchange from input data of the fieldbus node

512 Rx-PDOs for process data exchange from output data of the fieldbus node

Synchronization objects (SYNC) for network synchronization

Emergency objects (EMCY)

Network management objects

- Module Control Protocols
- Error Control Protocols
- Boot-up Protocol

10.3.2 Fieldbus-Specific Addressing

The CODESYS variable for the CAN bus (%QB6000 … %QB9999 and %IB6000 … %IB9999) are mapped to an object directory after configuring the CAN interface as a master or slave (initialization). A CANopen fieldbus device uses the 16-bit indices and 8-bit sub-indices of the object directory to address data via PDOs or SDOs and to access the data. The position of the data in the process image is therefore not directly significant for the CANopen user at the fieldbus end.

The variables entered into the object directory are distinguished by data type (Integer8, Unsigned8, Boolean, Integer16, etc.) and by input/output. Access via PDOs can be either for reading or writing. Direct access via SDO can be read-only.

As 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 a similar manner.

Note

Observe the direction of data flow!
The IEC 61131-3 input variables are defined from the perspective of the CAN bus. These are output variables from the perspective of the PFC. Accordingly, the IEC 61131-3 output variables are input variables for the PFC.

This table provides an overview of the indices of "IEC 61131-3" variables.

Table 102: Indexing of "IEC 61131-3" Variable Data in the Object Directory

Data Type	**IEC 61131-3 Output Variables**	**IEC 61131-3 Input Variables**
	Index	
Integer8	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
Integer40	0xA2C0	0xA740
Unsigned48	0xA300	0xA780
Integer48	0xA340	0xA7C0
Unsigned56	0xA380	0xA800
Integer56	0xA3C0	0xA840
Integer64	0xA400	0xA880
Unsigned64	0xA440	0xA8C0

Using the associated indices for data types with a data width of 1 byte (Integer8, Unsigned8 and Boolean), read-only byte-by-byte access is possible from the fieldbus to data in the controller memory.

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 PFC output variable data can be accessed in a word-by-word manner using the index for Integer16 (0xA0C0) or for Unsigned16 (0xA100), three bytes can be accessed using index 0xA140 for Integer24, etc.

Example:

The first three bytes of the PFC output data for the data type integer or unsigned are accessed from the fieldbus:

Table 103: Fieldbus access to PFC output data

Access	PFC Output Data	Reading with Index (Integer / Unsigned)	Sub-Index
By byte (with Integer8 / Unsigned8)	Byte 6000	(0xA000 / 0xA040)	1
	Byte 6001	(0xA000 / 0xA040)	2
	Byte 6002	(0xA000 / 0xA040)	3
By word (with Integer16 / Unsigned16)	Word 3000 (Byte 6000/6001)	(0xA0C0 / 0xA100)	1
	Word 3001 (Byte 6002/6003)	(0xA0C0 / 0xA100)	2
3 Bytes (with Integer24 / Unsigned24)	Bytes 6000 … 6002	(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 fieldbus variables (byte 6000 to byte 9999) as a function of the data width.

The indexing indicated in the tables continues up to the respective maximum index and sub-index.

Note

Observe the direction of data flow!
The PFC output variables are defined from the perspective of the controller; from the perspective of the CAN fieldbus these are input variables. Accordingly, the PFC input variables for IEC 61131-3 access are output variables for the fieldbus.

Thus: IEC 61131-3 input variable = PFC output variable

PFC input variable = IEC 61131-3 input variable.

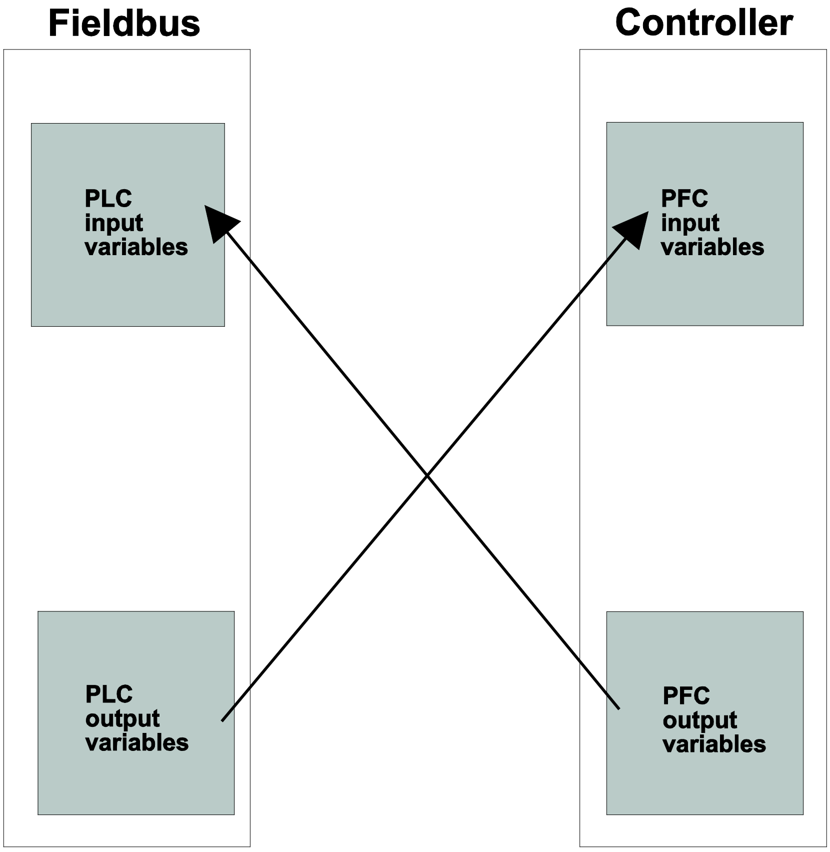


Figure 85: Correlation between “IEC 61131-3” variables and PFC variables

10.3.3 Examples for the Definition of PFC Fieldbus Variables

The examples below show the allocation of several definitions for PFC variables with different data types to the associated object directory entries.

10.3.3.1 CODESYS Access to PFC Variables

Table 104: Examples for CODESYS access to PFC variables

Data Type of PFC Variables	**PFC Input Variables**		**PFC Output Variables**	
	Definition based on IEC 61131-3	Index/sub-index	Definition based on IEC 61131-3	Index/sub-index
Unsigned8	InByte0 AT %IB6000: BYTE;	0xA4C0/1	OutByte0 AT %QB6000: BYTE;	0xA040 /1
	InByte0 AT %IB6001: BYTE;	0xA4C0/2	OutByte0 AT %QB6001: BYTE;	0xA040 /2
Integer16	InInt0 AT %IW3000: INT;	0xA540 /1	OutInt0 AT %QW3000: INT;	0xA0C0/1
	InInt1 AT %IW3001: INT;	0xA540 /2	OutInt1 AT %QW3001: INT;	0xA0C0/2
Unsigned16	InWord0 AT %IW3000: WORD;	0xA580 /1	OutWord0 AT %QW3000: WORD;	0xA100 /1
	InWord0 AT %IW3001: WORD;	0xA580 /2	OutWord0 AT %QW3001: WORD;	0xA100 /2
Unsigned32	InDWord0 AT %ID1500: DWORD;	0xA680 /1	OutDWord0 AT %QD1500: DWORD;	0xA200 /1
	InDWord0 AT %ID1501: DWORD;	0xA680 /2	OutDWord0 AT %QD1501: DWORD;	0xA200 /2

10.3.3.2 Maximum Indices

The maximum indices and sub-indices are yielded from the memory size of the fieldbus controller at 4000 bytes and the corresponding data width for the data types.

The table below provides an overview of the maximum indices and sub-indices of the IEC 61131-3 variables.

Table 105: Maximum indices and sub-indices for “IEC 61131-3” variables

Data Type	**IEC 61131-3 Input Variables**		**IEC 61131-3 Output Variables**	
	Max. Index	Max. sub-index	Max. Index	Max. sub-index
Integer8	0xA00F	0xFF	0xA487	0xFF
Unsigned8	0xA04F	0xFF	0xA4C7	0xFF
Boolean	0xA08F	0xFF	0xA507	0xFF
Integer16	0xA0C7	0xFF	0xA543	0xFF
Unsigned16	0xA107	0xFF	0xA583	0xFF
Integer24	0xA145	0x55	0xA5C0	0x55
Unsigned24	0xA185	0x55	0xA600	0x55
Integer32	0xA1C3	0xFF	0xA643	0xFF
Unsigned32	0xA203	0xFF	0xA683	0xFF
Float32	0xA243	0xFF	0xA6C3	0xFF
Unsigned40	0xA283	0x33	0xA703	0x33
Integer40	0xA2C3	0x33	0xA743	0x33
Unsigned48	0xA302	0xAA	0xA780	0xAA
Integer48	0xA342	0xAA	0xA7C0	0xAA
Unsigned56	0xA382	0x49	0xA802	0x49
Integer56	0xA3C2	0x49	0xA842	0x49
Integer64	0xA401	0xFF	0xA880	0xFF
Unsigned64	0xA441	0xFF	0xA8C0	0xFF

Example:

514 bytes of output variables are addressed by word by the data type Unsigned16.

Addressing of 257 data words then occurs with:

- Index 0xA580, sub-index 1 to 255
- Index 0xA581, sub-index 1 and 2.

Table 106: Example of “IEC 61131-3” output variables

Index	**Sub-Index**	**Contents**	**Description**
0xA580	1	D1 *)	1st output variable block
	2	D2 *)	2nd output variable block
	...	...	...
	255	D255 *)	255th output variable block
0xA581	1	D256 *)	256th output variable block
	2	D257 *)	257th output variable block

*) D1 = Data word output variable 1, D255 = Data word output variable 255, etc.

10.3.4 CANopen Master Control Configuration

Note

Calling up the addresses or the symbolic name of the inputs and outputs
Addresses or symbolic names of the inputs and outputs have to be called up explicitly, otherwise the process image is not updated. Alternatively, you can also create an array of max. 240 bytes at the memory addresses IB%6000 or QB%6000. This array has to be called up in the PLC program.

An application must be configured in CODESYS before it can access the connected CAN network.

10.3.4.1 Selecting the Master

To add the CANopen Master to the control configuration, right-click “COS unused[Slot]” and select “Replace element -> CANopen Master”.

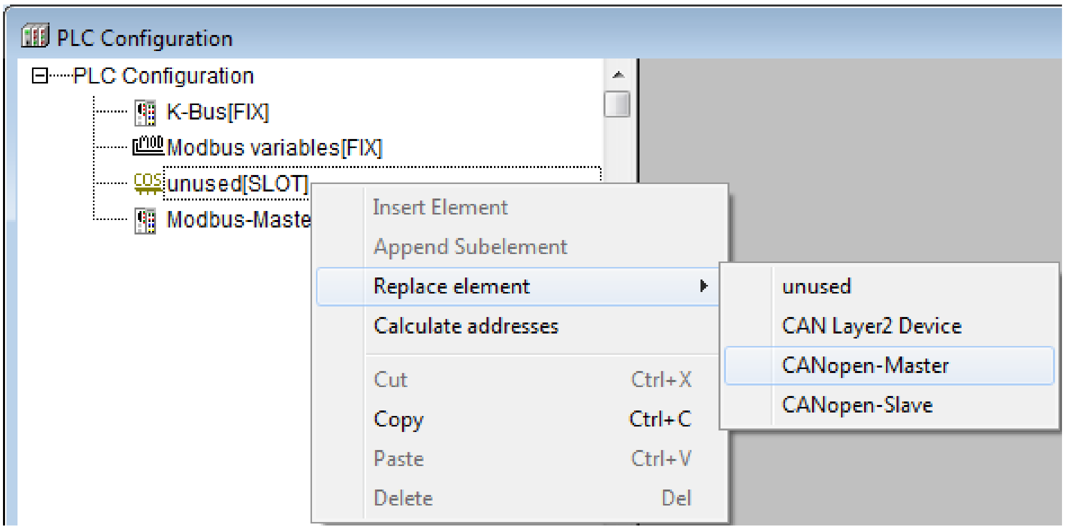
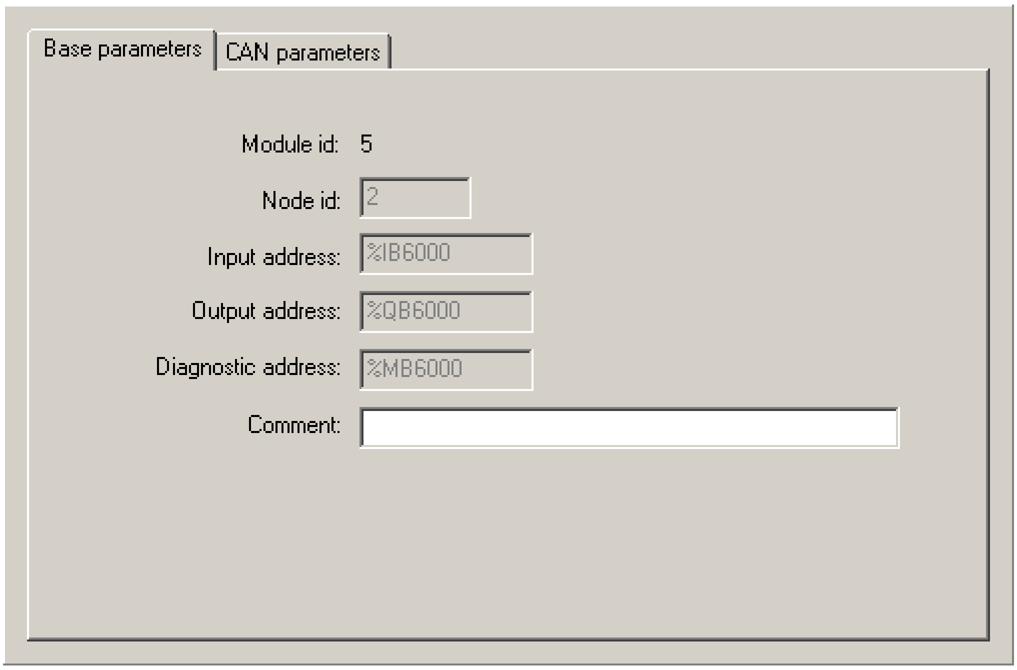


Figure 86: Adding the CANopen master

10.3.4.2 Setting the Master Parameters

No input should be made on the Basic parameters tab.



Module id: 5

Node id: 2

Input address: %IB6000

Output address: %QB6000

Diagnostic address: %MB6000

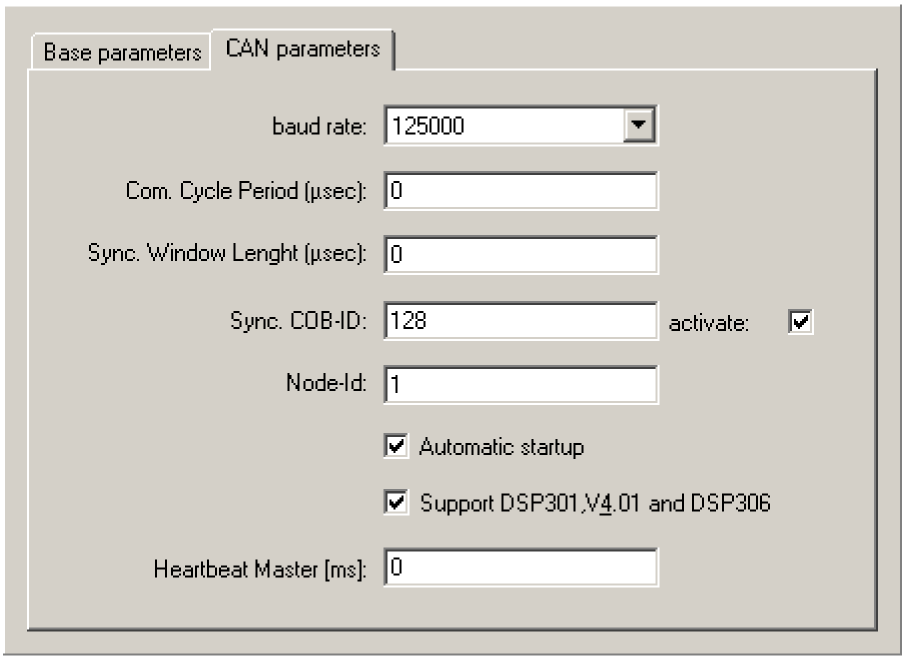
Comment:

Figure 87: Master basic parameters tab

Table 107: Description of Basic Parameters

Parameters	**Explanation**
Module ID	Parameters that use the runtime system CODESYS.
Node ID	
Input address	
Output address	
Diagnostic address	
Comment	Input field for comments.

Normally, you must only check the baud rate on the CAN parameters tab.



baud rate: 125000

Com. Cycle Period (µsec): 0

Sync. Window Lenght (µsec): 0

Sync. COB-ID: 128 activate: ☑

Node-Id: 1

☑ Automatic startup

☑ Support DSP301,V4.01 and DSP306

Heartbeat Master [ms]: 0

Figure 88: Master CAN Parameters tab

Table 108: Description of the CAN parameters

Parameters	Explanation
Baud rate	Specify the baud rate that will apply to the transfer on the CAN bus (default setting: 125000 baud).
Com. Cycle Period (µsec)	Specify the time interval (in µsec) that the synchronization notification will be sent by the controller. Smallest time interval: 1000 µsec
Sync. Window Length (µsec)	Not currently implemented.
Sync. COB-ID	You can enable or disable sending of synchronization notification by the controller. Default setting: COB ID 128 (0x80).
Node ID	Station address (node ID) of the controller on the CAN bus.
Start automatically	When you mark this check box, the controller automatically sets the CAN master and slaves to the “Operational” based on the defined parameters. If you do not mark this check box, starting can be performed using the CIA405NMT library command.

Table 108: Description of the CAN parameters

Parameters	Explanation
Support ... DSP 301	If you mark this check box, modular CAN slaves as well as additional extensions adhering to the DSP301 V3.01 and DSP 306 standards are supported by the control configuration.
Heartbeat Master	If the “Heartbeat Generation” option is enabled, the CAN device transmits heartbeats in ms intervals specified in “Heartbeat Producer Time”. Heartbeat consumption is not currently implemented.

10.3.4.3 Adding Slaves

To select one (or several) CANopen slaves, right-click on the CANopen master and select “Attach subelements”. In this example, the 750-337 was selected as the slave.

Note

EDS Files
The EDS files for current components of the WAGO-I/O-SYSTEM are integrated in the target files for the controller. The associated EDS files are required for incorporation of non-WAGO devices. For this, click “Tools” > “Add configuration file” in the menu bar.

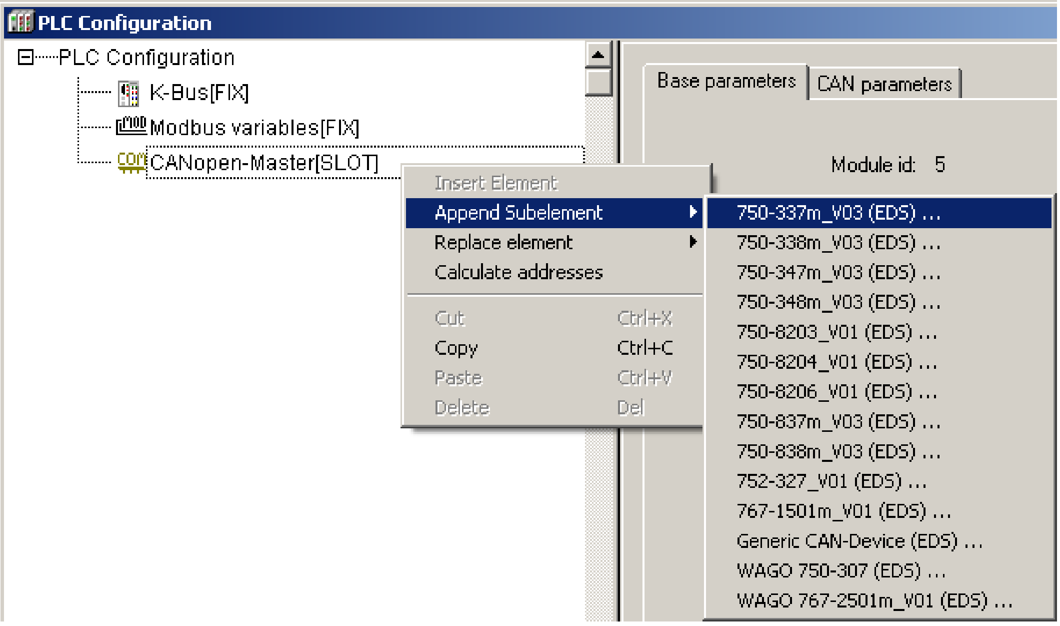
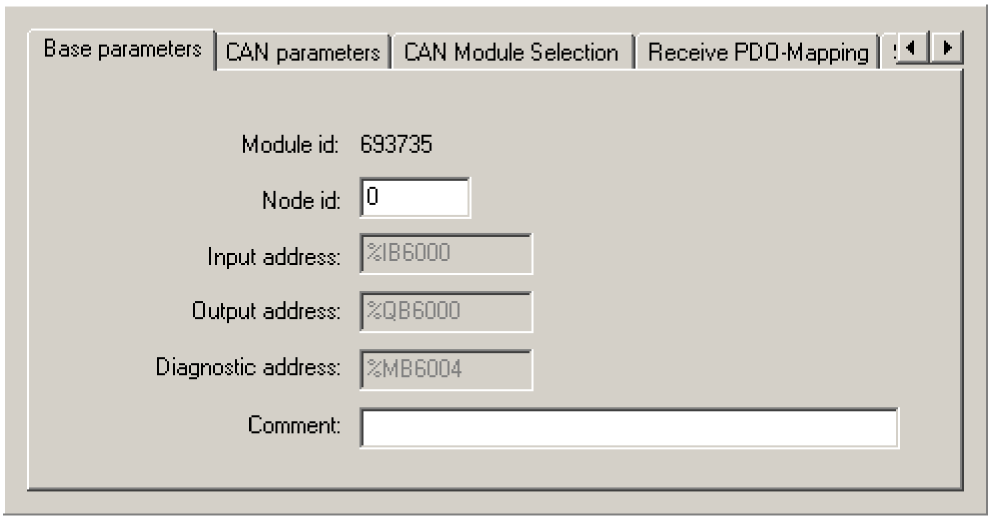


Figure 89: Adding a CANopen slave

No input must be made on the Basic parameters tab.



Module id: 693735

Node id: 0

Input address: %IB6000

Output address: %QB6000

Diagnostic address: %MB6004

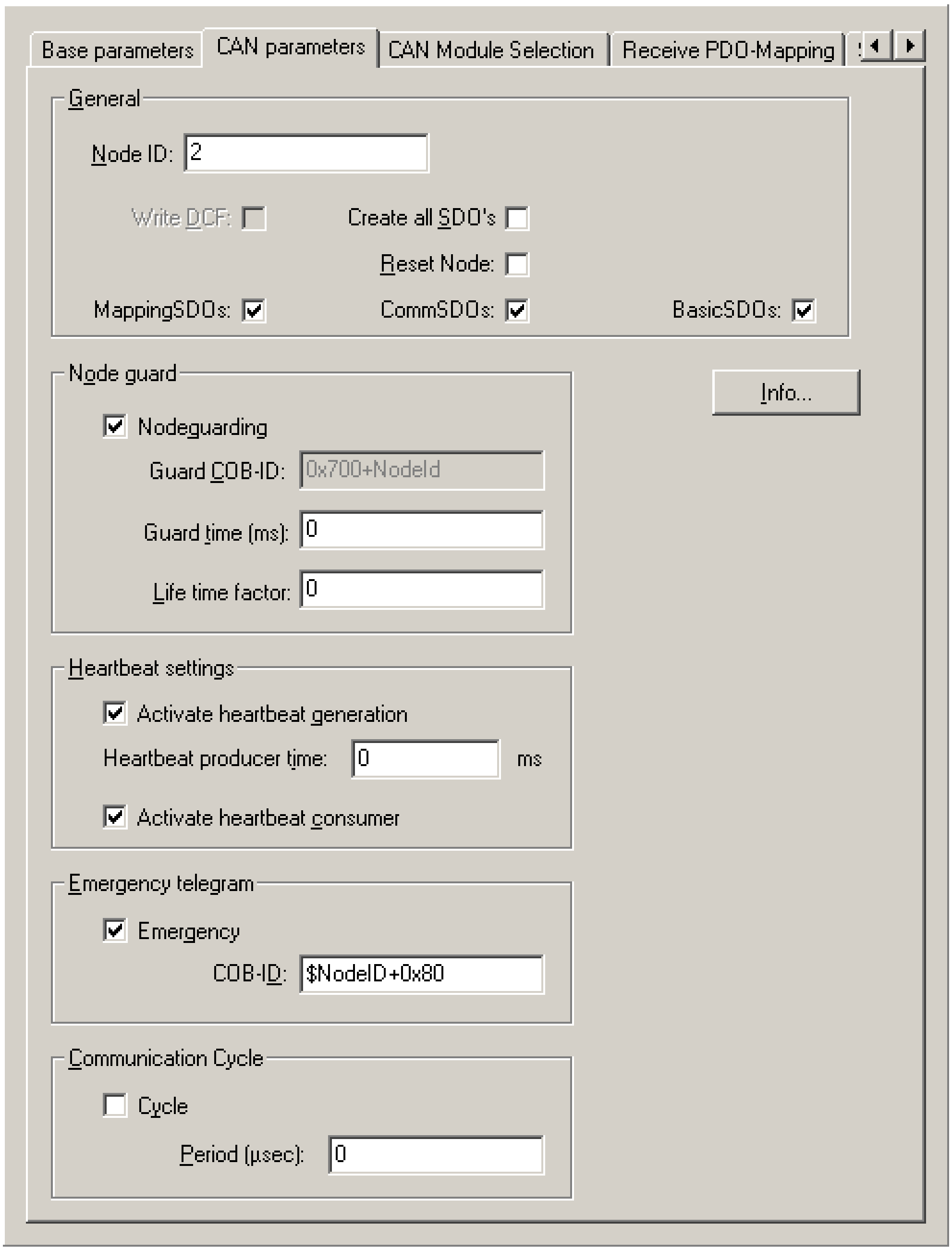
Comment:

Figure 90: Basic parameters tab

Table 109: Description of Basic Parameters

Parameters	**Explanation**
Module ID	Recognition of the slave.
Node ID	Node number of the slave used in the CODESYS runtime environment.
Input address	Starting address for the input data: The address space always begins at %IB 6000 and is assigned automatically.
Output address	Starting address for the output data: The address space always begins at %QB 6000 and is assigned automatically.
Diagnostic address	Memory area for internal diagnostic processing.
Comment	Input field for comments.

Normally, you must only check the Node ID on the CAN parameters tab.



General

Node ID: 2

Write DCF: □ Create all SDO's □

Reset Node: □

MappingSDOs: ☑ CommSDOs: ☑ BasicSDOs: ☑

Node guard

☑ Nodeguarding

Guard COB-ID: 0x700+NodeId

Guard time (ms): 0

Life time factor: 0

Heartbeat settings

☑ Activate heartbeat generation

Heartbeat producer time: 0 ms

☑ Activate heartbeat consumer

Emergency telegram

☑ Emergency

COB-ID: $NodeID+0x80

Communication Cycle

□ Cycle

Period (µsec): 0

Figure 91: CAN parameters tab

Table 110: Description of the CAN parameters

Parameters	Explanation
General	
Node ID	The node ID (1–126) is the station address under which the controller communicates with the slave on the CAN network.
Write DCF	Currently not implemented.
Generate all SDOs	When this check box is marked, SDOs are generated for all objects in the EDS file. In addition, the corresponding options must be activated. If the node-guarding objects are to be written, for example, the checkbox for the “Node-Guarding” option must also be marked. If the checkbox is not marked, SDOs are only generated for the objects in which the default values deviate from the EDS file.
Reset node	If you activate this option, the slave is reset by the controller (receives a “reset node”) before the configuration is sent to the slave. This function is not currently implemented.
Mapping SDOs	Activate or deactivate each of the three SDO ranges of the slave configuration here.
Comm SDOs	Mapping SDOs: Objects 0x1600 … 0x1620 Objects 0x1A00 … 0x1A20 Comm SDOs: Objects 0x1400 … 0c1420 Objects 0x1800 … 0x1820 Basic SDOs: Objects 0x100C … 0x1017
Basic SDOs	
Node Guard	
Node-Guarding	With Node-Guarding enabled, the slave monitors the PFC for any potential disruption of fieldbus communication.
Guard COB ID	Default setting: 0x700 + Node ID.
Guard Time (ms)	Under“Guard Time” specify the interval at which the PFC expects to receive “Confirmation” from the slave.
Lifetime factor	In the field “Lifetime factor” (>= 2) specify the multiplier for the “Guard time”. If the time yielded from “Guard time” x "Lifetime factor” “Node lifetime”) has expired, the slave is brought into the predefined state. This function is deactivated by “0”.

Table 110: Description of the CAN parameters

Parameters	Explanation
Heartbeat Settings	
Activate heartbeat generation	If the “Heartbeat Generation” option is enabled, the CAN device transmits heartbeats in ms intervals specified in “Heartbeat Producer Time”. This function is disabled by “0”.
Heartbeat producer time (ms)	
Activate heartbeat consumption	If the “Heartbeat Consumption” option is enabled, the CAN device monitors the heartbeat of the master. Only “Heartbeat” or “Node-Guarding” can be used for monitoring.
Emergency Telegram	
Emergency	If you mark this checkbox, the slave sends error and status messages that are stored as emergency messages to the diagnostic address in the flag area. These error and status messages are read out using “BusDiag.lib”. If you do not mark this checkbox, SDO 0x1014 is not transmitted to the slave. The default setting for the slave would then still apply.
COB-ID	Default: Node ID + 0x80
Communication Cycle	
Cycle	These functions are currently not implemented.
Period (µsec)	

Click **[Info …]** to display the parameters “FileInfo” and “DeviceInfo” from the EDS file.

You can now select the installed input and output modules in the CAN module selection tab. In the example shown here, one 8-bit input and output module each.

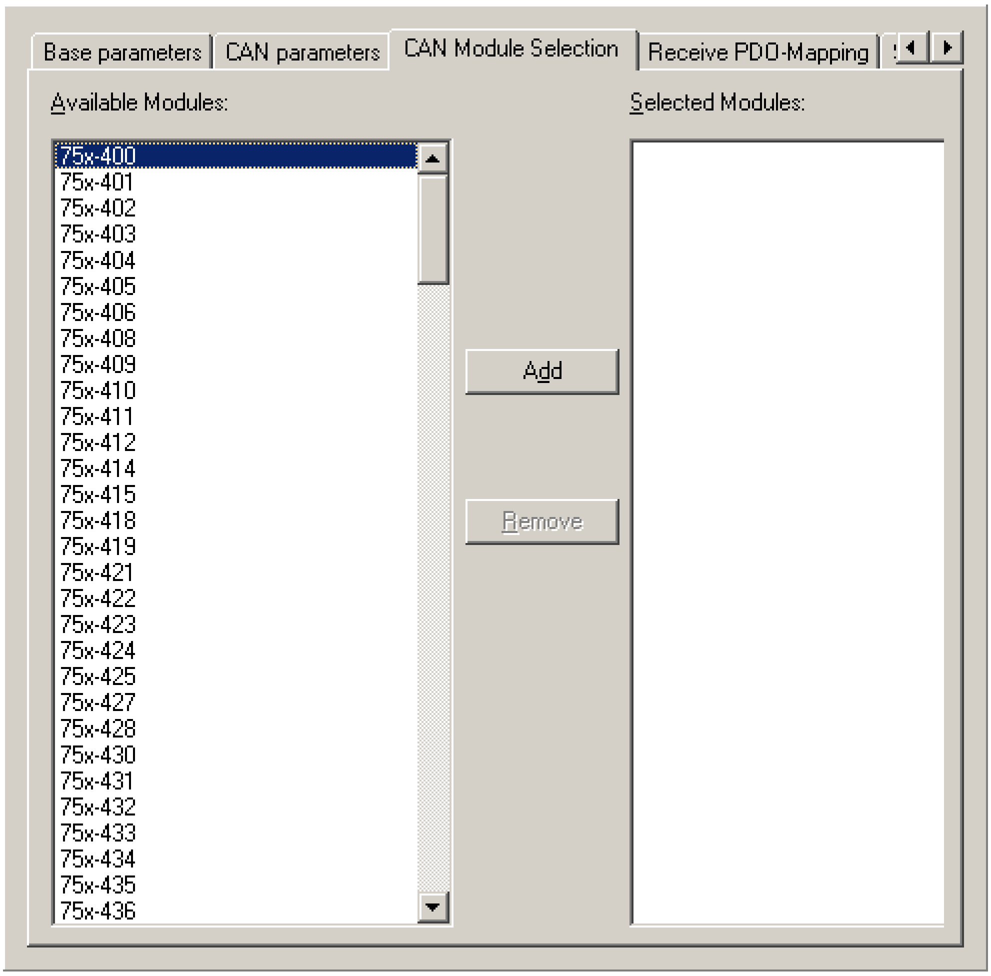


Figure 92: CAN module selection tab

10.3.4.4 Configuring the Slave PDOs

Now, configure the receiving PDOs for the slave (output from the perspective of the controller) and the send PDOs. The PDOs for the module have already been created by the Configurator. The eight bits have been placed in the first PDO.

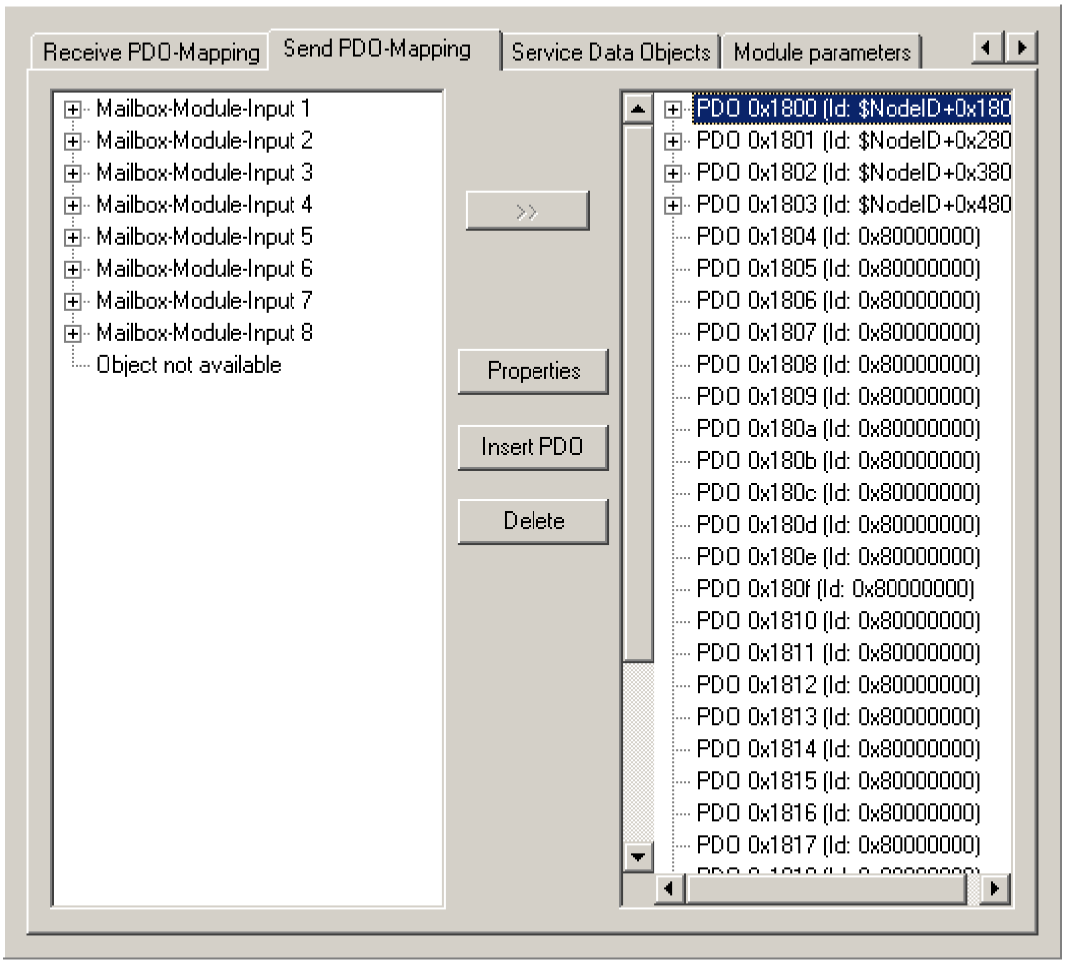
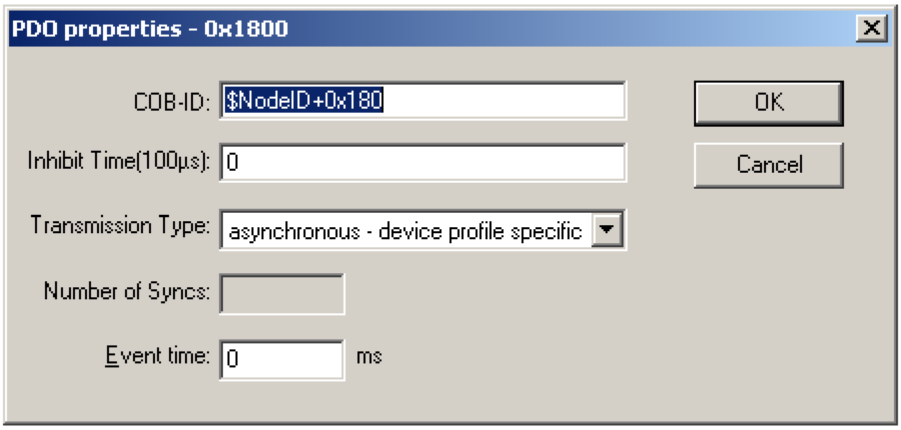

Receive PDO-Mapping
Send PDO-Mapping
Service Data Objects
Module parameters
Mailbox-Module-Input 1
Mailbox-Module-Input 2
Mailbox-Module-Input 3
Mailbox-Module-Input 4
Mailbox-Module-Input 5
Mailbox-Module-Input 6
Mailbox-Module-Input 7
Mailbox-Module-Input 8
Object not available
>>
Properties
Insert PDO
Delete
PDO 0x1800 (Id: $NodeID+0x180
PDO 0x1801 (Id: $NodeID+0x280
PDO 0x1802 (Id: $NodeID+0x380
PDO 0x1803 (Id: $NodeID+0x480
PDO 0x1804 (Id: 0x80000000)
PDO 0x1805 (Id: 0x80000000)
PDO 0x1806 (Id: 0x80000000)
PDO 0x1807 (Id: 0x80000000)
PDO 0x1808 (Id: 0x80000000)
PDO 0x1809 (Id: 0x80000000)
PDO 0x180a (Id: 0x80000000)
PDO 0x180b (Id: 0x80000000)
PDO 0x180c (Id: 0x80000000)
PDO 0x180d (Id: 0x80000000)
PDO 0x180e (Id: 0x80000000)
PDO 0x180f (Id: 0x80000000)
PDO 0x1810 (Id: 0x80000000)
PDO 0x1811 (Id: 0x80000000)
PDO 0x1812 (Id: 0x80000000)
PDO 0x1813 (Id: 0x80000000)
PDO 0x1814 (Id: 0x80000000)
PDO 0x1815 (Id: 0x80000000)
PDO 0x1816 (Id: 0x80000000)
PDO 0x1817 (Id: 0x80000000)

Figure 93: PDO mapping tab

Table 111: Receiving and Sending Description for PDO Mapping

Parameters	Explanation
Insert PDO	Depending on the I/O modules selected for the CANopen slave, the corresponding CANopen objects appear on the “Receive PDO Mapping” (PFC → slave) and “Send PDO Mapping” (slave → PFC) tabs. Using these tabs, you can change the “Default Mapping” described in the EDS file. Click on the button **[Add PDO]** to match the PDOs to the I/O module topology. The PDO properties window opens for defining certain properties for the PDO. More information about this is provided under “Properties”. In order to assign one of the PDOs an object from the left window, mark both the corresponding object and the corresponding PDO and then click [>>]. Then the object will be added below the PDOs in the right window. The first 64 digital and the first 12 analog inputs and outputs are then assigned automatically to the PDOs 1–4.
Remove	Click **[Remove]** to remove the item currently marked in the right window from the configuration.
Features	A dialog box with information about the PDO properties opens (see next page).

Further configuration of the PDOs is also possible using the “Properties” button.



PDO properties - 0x1800

COB-ID: $NodeID+0x180

Inhibit Time(100µs): 0

Transmission Type: asynchronous - device profile specific

Number of Syncs:

Event time: 0 ms

Figure 94: PDO properties window

Table 112: Description of the PDO Properties window

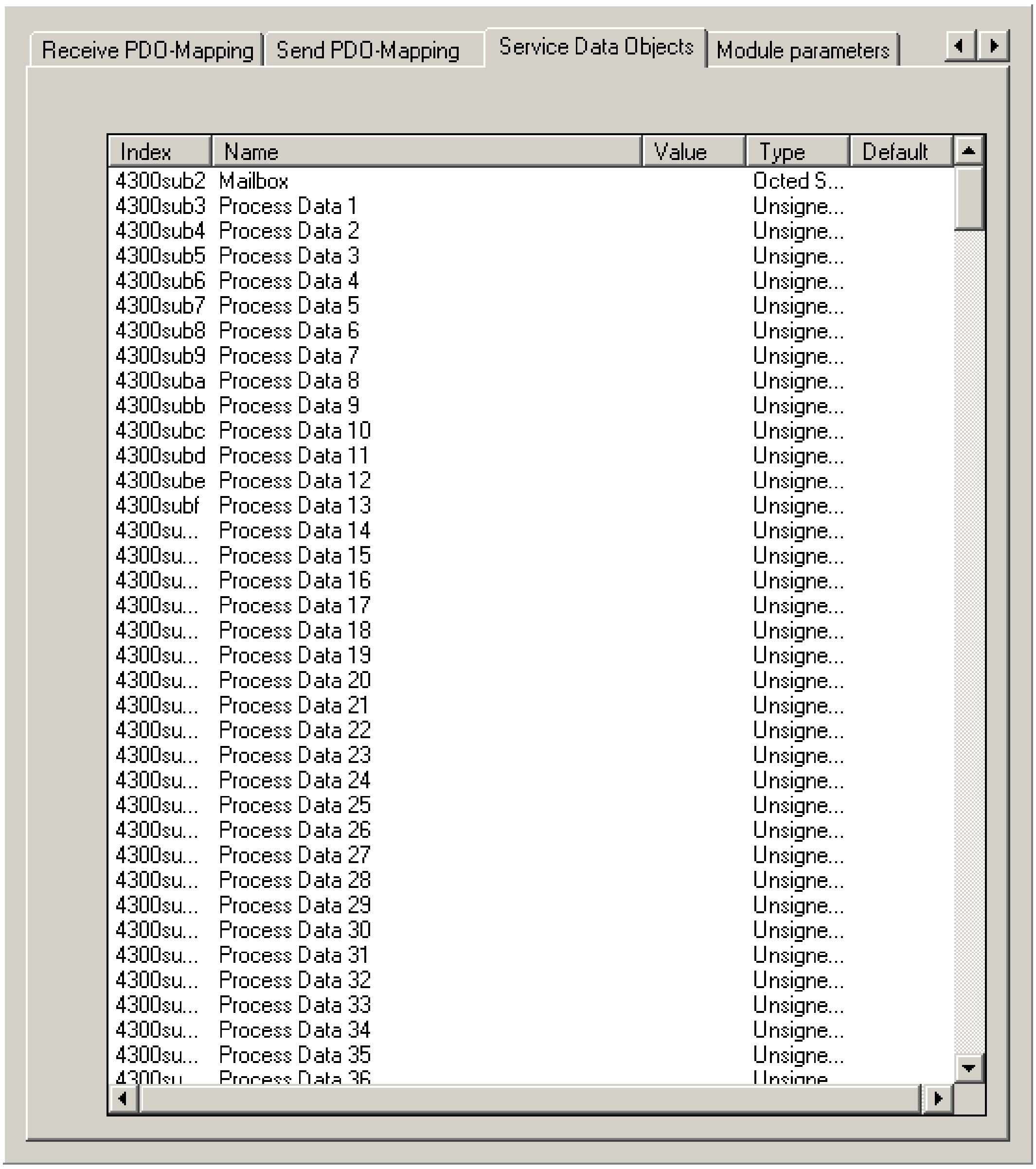
Parameters	Explanation
COB-ID	CAN Identifier
Inhibit Time (* 100 µs)	Here, specify the time span of a PDO to reduce communication incidence; this is the time that must pass before it can be sent again. This value is not used for synchronous transmission. This value is insignificant for a receive PDO.
Transmission Type	Here, select the transmission mode for the PDO: acyclic-synchronous: (transmission type 0) The PDO is transmitted synchronously, but not periodically. For receive PDOs, the transmission types 0–240 are handled the same way. cyclic-synchronous: (transmission type 1–240) The PDO is transmitted synchronously, whereby “Number of Syncs” specifies the number of synchronization messages that lie between two transmissions of the PDO. For receive PDOs, the transmission types 0–240 are handled the same way. synchronous - only RTR: (transmission type 252) The PDO is updated after a synchronization message, but not sent. It is only transmitted with an explicit inquiry “Remote Transmission Request” (not implemented). asynchronous - only RTR: (transmission type 253) The PDO is only updated and transmitted with an explicit inquiry “Remote Transmission Request” (not implemented). asynchronous-vendor-specific: (transmission type 254) The PDO is only transmitted after particular events. asynchronous-device-profile-specific: (transmission type 255) The PDO is only transmitted after specific events.
Number of Syncs	Depending on the “transmission type,” this field can be edited to enter the number of synchronization messages from 1–240. This value is insignificant for a receive PDO.

Table 112: Description of the PDO Properties window

Parameters	Explanation
Event Time	Depending on the “transmission type”, enter the time span (in ms) that should elapse between two transmissions of the PDO. This value is insignificant for a receive PDO.

10.3.4.5 Configuring the Service Data Objects

Service data objects can also be configured in addition to the configuration performed on the previous tabs.



Index	Name	Value	Type	Default
4300sub2	Mailbox		Octed S...	
4300sub3	Process Data 1		Unsigne...	
4300sub4	Process Data 2		Unsigne...	
4300sub5	Process Data 3		Unsigne...	
4300sub6	Process Data 4		Unsigne...	
4300sub7	Process Data 5		Unsigne...	
4300sub8	Process Data 6		Unsigne...	
4300sub9	Process Data 7		Unsigne...	
4300suba	Process Data 8		Unsigne...	
4300subb	Process Data 9		Unsigne...	
4300subc	Process Data 10		Unsigne...	
4300subd	Process Data 11		Unsigne...	
4300sube	Process Data 12		Unsigne...	
4300subf	Process Data 13		Unsigne...	
4300su...	Process Data 14		Unsigne...	
4300su...	Process Data 15		Unsigne...	
4300su...	Process Data 16		Unsigne...	
4300su...	Process Data 17		Unsigne...	
4300su...	Process Data 18		Unsigne...	
4300su...	Process Data 19		Unsigne...	
4300su...	Process Data 20		Unsigne...	
4300su...	Process Data 21		Unsigne...	
4300su...	Process Data 22		Unsigne...	
4300su...	Process Data 23		Unsigne...	
4300su...	Process Data 24		Unsigne...	
4300su...	Process Data 25		Unsigne...	
4300su...	Process Data 26		Unsigne...	
4300su...	Process Data 27		Unsigne...	
4300su...	Process Data 28		Unsigne...	
4300su...	Process Data 29		Unsigne...	
4300su...	Process Data 30		Unsigne...	
4300su...	Process Data 31		Unsigne...	
4300su...	Process Data 32		Unsigne...	
4300su...	Process Data 33		Unsigne...	
4300su...	Process Data 34		Unsigne...	
4300su...	Process Data 35		Unsigne...	
4300su	Process Data 36		Unsigne	

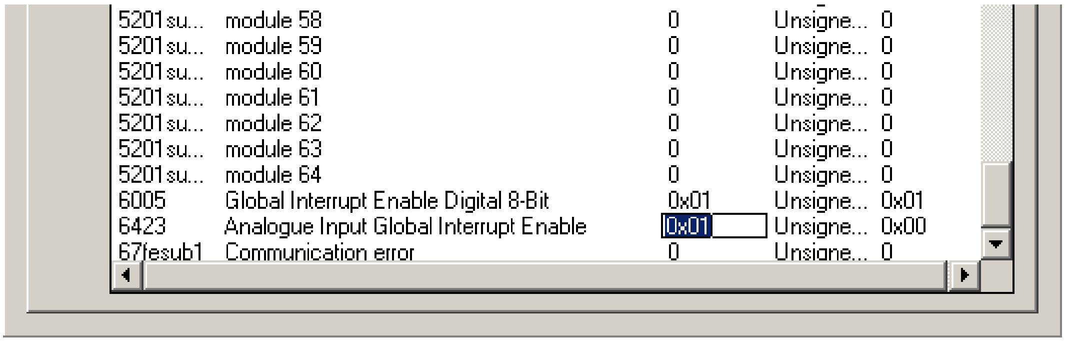
Figure 95: Service data objects tab

All objects of the EDS file are listed here, which range from index 0x2000 to 0x9FFF and can be described.

For each object, the index, name, value, type and default are specified.

The value of the objects can be changed. To do this, mark the field in question in the “Value” column and overwrite the value with your input, then press **[Enter]**. On initialization of the CAN bus, the set values will be transmitted to the slaves as SDOs.

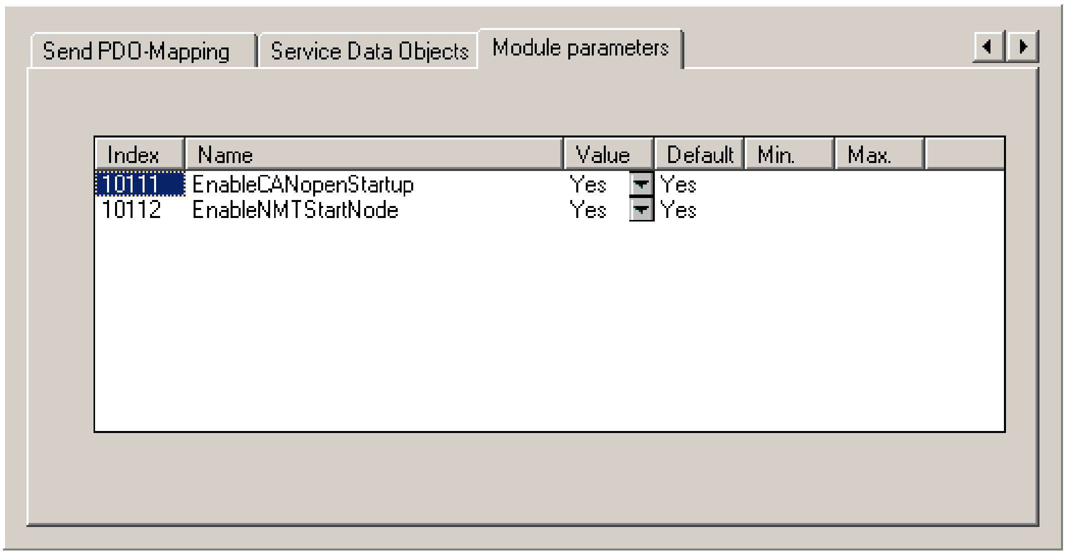
An example here is the activation of sending PDOs on a change in analog values, which is normally deactivated:



5201su...	module 58	0	Unsigne...	0
5201su...	module 59	0	Unsigne...	0
5201su...	module 60	0	Unsigne...	0
5201su...	module 61	0	Unsigne...	0
5201su...	module 62	0	Unsigne...	0
5201su...	module 63	0	Unsigne...	0
5201su...	module 64	0	Unsigne...	0
6005	Global Interrupt Enable Digital 8-Bit	0x01	Unsigne...	0x01
6423	Analogue Input Global Interrupt Enable	0x01	Unsigne...	0x00
67fesub1	Communication error	0	Unsigne...	0

Figure 96: Adapting SDOs

The starting parameters for the slave can now be set on the “Module parameters” tab.



Index	Name	Value	Default	Min.	Max.
10111	EnableCANopenStartup	Yes	Yes		
10112	EnableNMTStartNode	Yes	Yes		

Figure 97: “Module parameters” tab

Table 113: Description of the module parameters (slave)

Parameters	Explanation
EnableCANopenStartup	Yes (default): During the boot-up phase of the CANopen network, all basic SDO frames are sent to the selected CANopen slave. No: With this setting, no SDO frames are sent to the CANopen slave.
EnableNMTStartNode	Yes (default): During the boot-up phase of the CANopen network, the NMT command “Start remote node” is sent to the selected CANopen slave (communication connection is established). No: With this setting, no start command is transmitted. The CANopen slave can be started at any time using the “Start remote node” command. Note: to do this, deactivate the parameter “Start automatically”:

10.3.5 CANopen Slave Control Configuration

An application must be configured in CODESYS before it can access the connected CAN network.

To add the CANopen slave to the control configuration, right-click “COS unused[Slot]” and select “Replace element -> CANopen slave”.

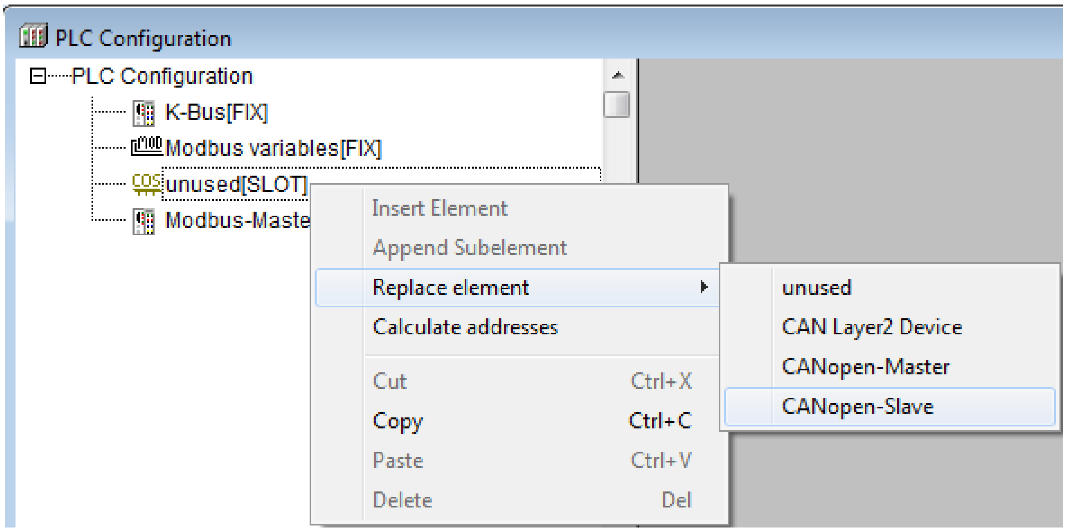


Figure 98: Attaching a CANopen slave

Right-click on CANopen Slave and select “Edit” to configure the slave.

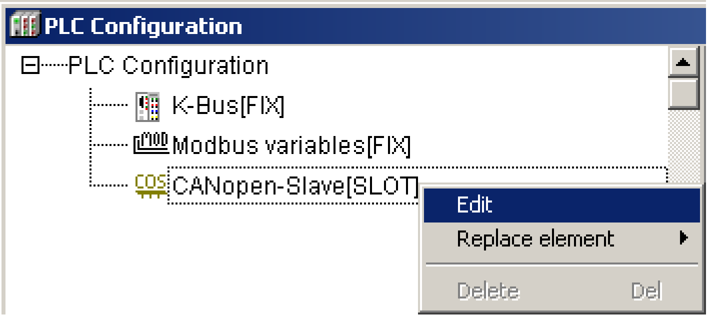
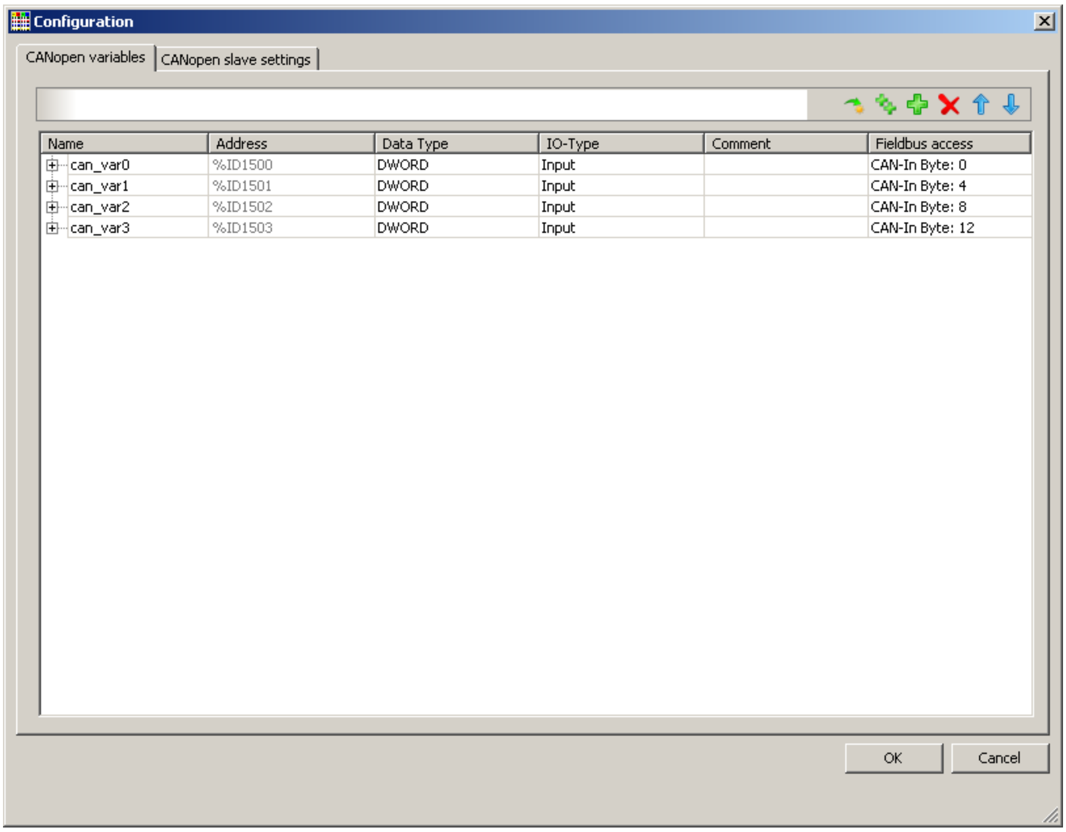


Figure 99: Configuring a CANopen slave

10.3.5.1 CANopen Variables Configuration



Name	Address	Data Type	IO-Type	Comment	Fieldbus access
can_var0	%ID1500	DWORD	Input		CAN-In Byte: 0
can_var1	%ID1501	DWORD	Input		CAN-In Byte: 4
can_var2	%ID1502	DWORD	Input		CAN-In Byte: 8
can_var3	%ID1503	DWORD	Input		CAN-In Byte: 12

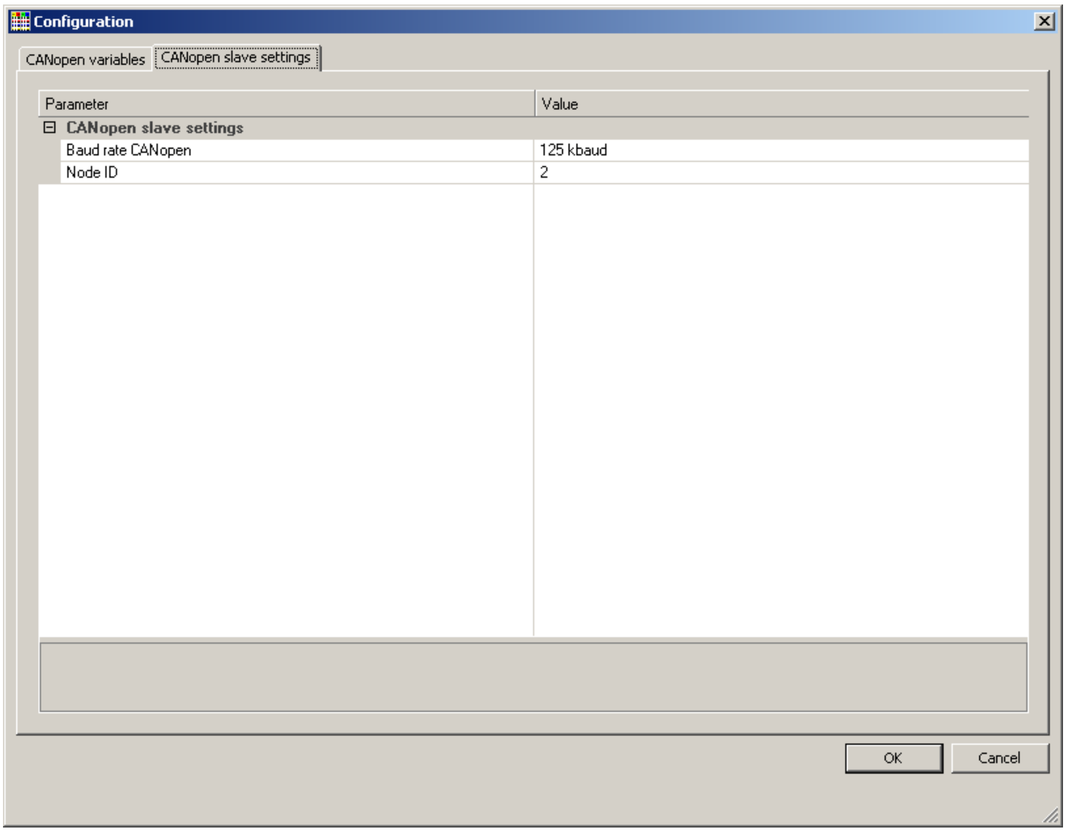
Figure 100: Configuration of the CANopen slave variables

CANopen variables can be added by clicking the **[+]** (Add) button. After this, you must specify the data type and communication direction (I/O type).

Table 114: Description of the CANopen Slave Variables Window

Parameters	**Explanation**
Name	You can edit the name for the variable or for its bits.
Address	Output of the CODESYS address in the input or output area, based on data type.
Data Type	The following data types may be used: BOOL, BYTE ,WORD, DWORD, SINT, INT, DINT, USINT, UINT, UDINT, REAL, BYTE(Array) The byte array can be extended by adding bytes until the required size is achieved. The maximum length of the array is eight bytes.
Comment	Input field for a comment
I/O type	Input or output
Fieldbus access	Output of the address offset in bytes, relative to the beginning of the CAN data range

10.3.5.2 Configuring of CANopen Parameters



Parameter	Value
⊟ **CANopen slave settings**	
Baud rate CANopen	125 kbaud
Node ID	2

Figure 101: Configuration of CANopen slave parameters

Table 115: Description of CANopen Slave Settings

Parameters	**Explanation**
Baud rate	Specify the baud rate to apply for the transfer on the CAN bus (default setting: 125000 baud).
Node ID	PFC's Node ID on the CAN bus.

10.4 Fieldbus Coupler Diagnostics

This section requires substantial knowledge of the CODESYS programming tool. It only describes the procedure to create diagnostics using the fieldbus master.

Configured slaves (e.g., a fieldbus coupler or a fieldbus controller) are required for diagnostics in fieldbus networks.

10.4.1 DiagGetBusState() and DiagGetState()

To evaluate the diagnostics, you need the following function blocks from the BusDiag.lib library:

- DiagGetBusState() for bus diagnostics
  This function block provides general information on each connected slave (e.g., number of slaves).

- DiagGetState() for participant diagnostics
  This function block provides detailed information on each slave (e.g., information on diagnostics).

10.4.2 Creating Diagnostics 7 in CODESYS 2.3

In order to execute bus diagnostics or device diagnostics for the slaves, it is necessary to integrate the BusDiag.lib library into CODESYS. This library contains the necessary function blocks DiagGetBusState() for bus diagnostics and DiagGetState() for participant diagnostics.

Integrate the BusDiag.lib library into CODESYS as described below:

1. Click the “Resources” tab.

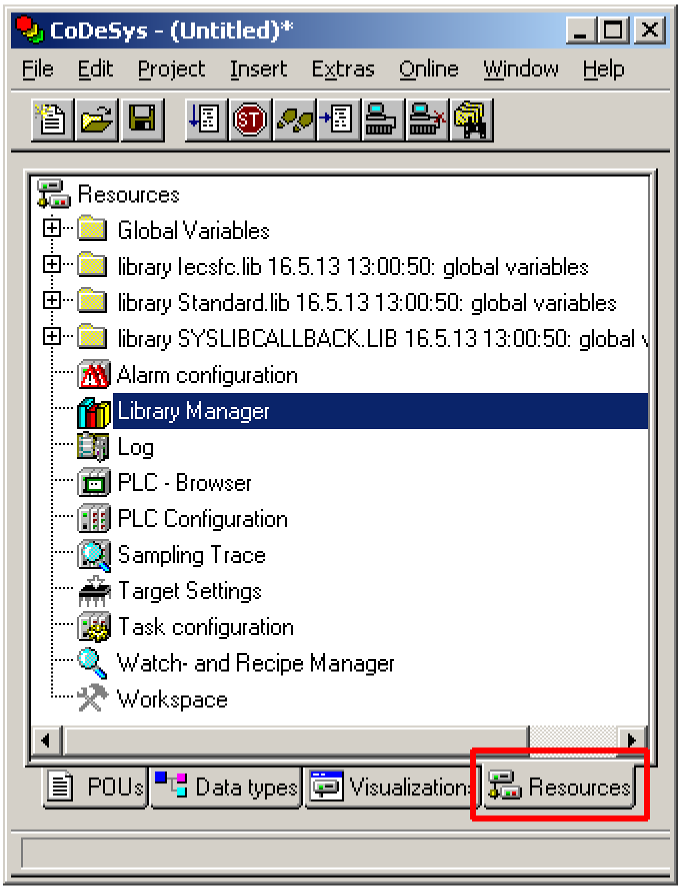
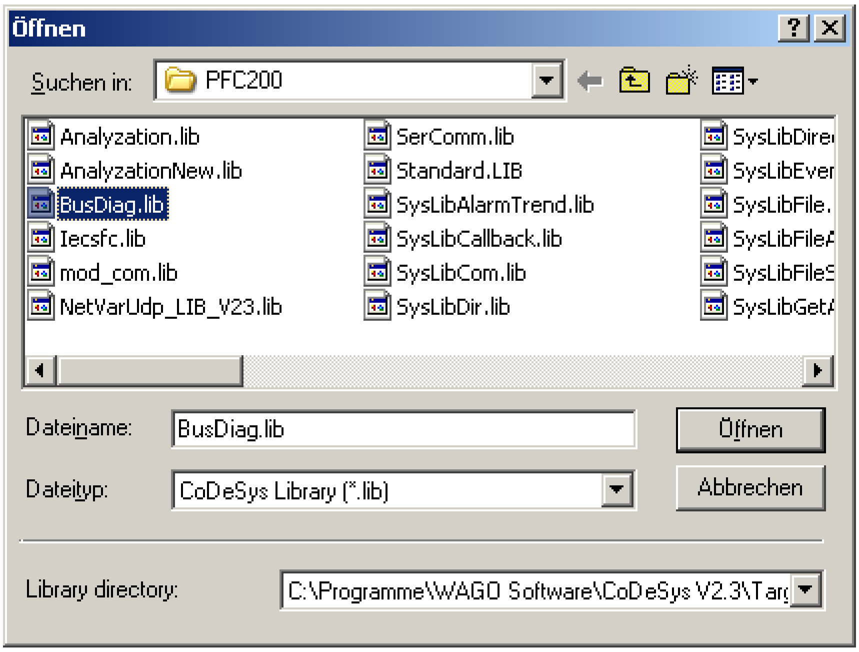


Figure 102: “Resources” tab

2. In the left column, double-click “Library Manager”.

3. In the menu bar, click on **Insert** > **Additional library**. The “Open” dialog window then opens. Select the “BusDiag.lib” and click **[Open]** to add it to the project.



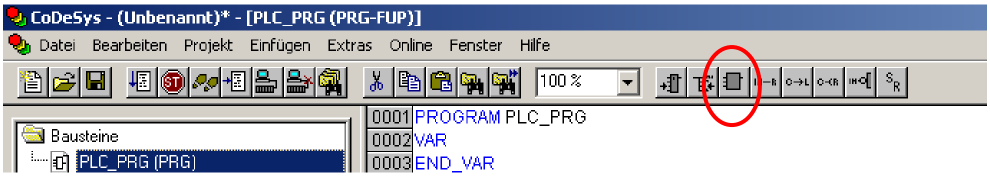
Dateiname: BusDiag.lib

Dateityp: CoDeSys Library (*.lib)

Library directory: C:\Programme\WAGO Software\CoDeSys V2.3\Targ

Figure 103: “Open” dialog window

4. In the menu bar, click the “Box” symbol.



```
0001 PROGRAM PLC_PRG
0002 VAR
0003 END_VAR
```

Figure 104: Module symbol in the menu bar; FUP programming language

5. Press **[F2]** on your keyboard. The “Input Assistant” dialog window opens. Click the “Standard function block” option and select the function block DiagGetBusState().

6. Create an entity for the function block DiagGetBusState(). Enter a name above the function block.
In the example shown here this is “GeneralBusInformation”.

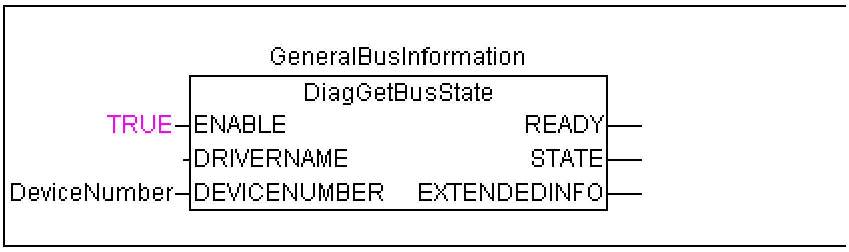


Figure 105: Entity for the function block DiagGetBusState() in FUP

7. Call up the function block DiagGetBusState() for the slave diagnostics from the library BusDiag.lib.

8. Create an entity for the function block DiagGetState().
Shown in this example is “DiagnosticsNode”.

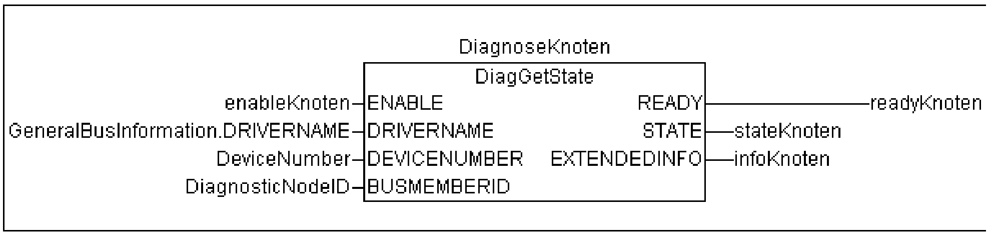
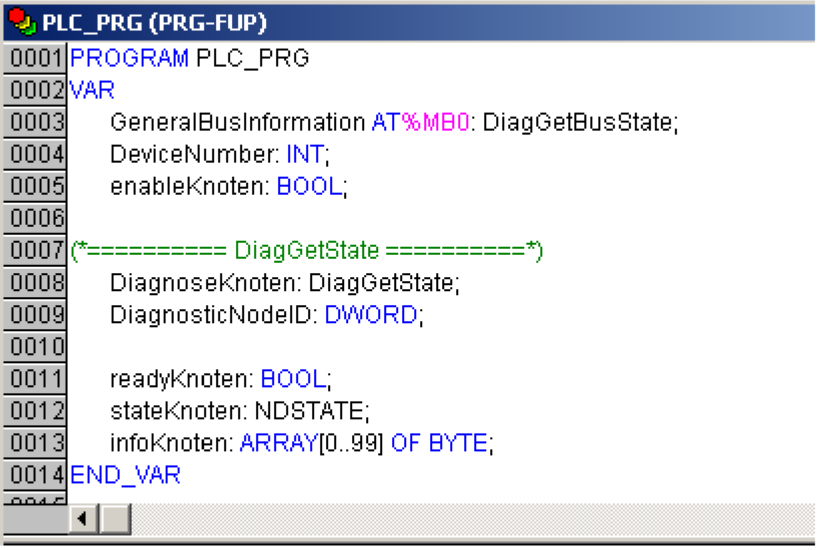


Figure 106: Function block DiagGetState() in FUP

In this example, both function blocks are called up during the program sequence. In order to not prolong the cycle times during the program sequence, do not set the input “ENABLE” of DiagGetState() to “TRUE” until you have conducted diagnostics.

10.4.3 Calling Up the Diagnostics Function Block

Call up the function block as shown in the following figure.



```
PROGRAM PLC_PRG
VAR
    GeneralBusInformation AT%MB0: DiagGetBusState;
    DeviceNumber: INT;
    enableKnoten: BOOL;

(*========== DiagGetState ==========*)
    DiagnoseKnoten: DiagGetState;
    DiagnosticNodeID: DWORD;

    readyKnoten: BOOL;
    stateKnoten: NDSTATE;
    infoKnoten: ARRAY[0..99] OF BYTE;
END_VAR
```

Figure 107: Offline view of the variable window in CODESYS

10.4.4 Executing a Bus Diagnosis using DiagGetBusState()

To perform bus diagnostics, proceed as follows:

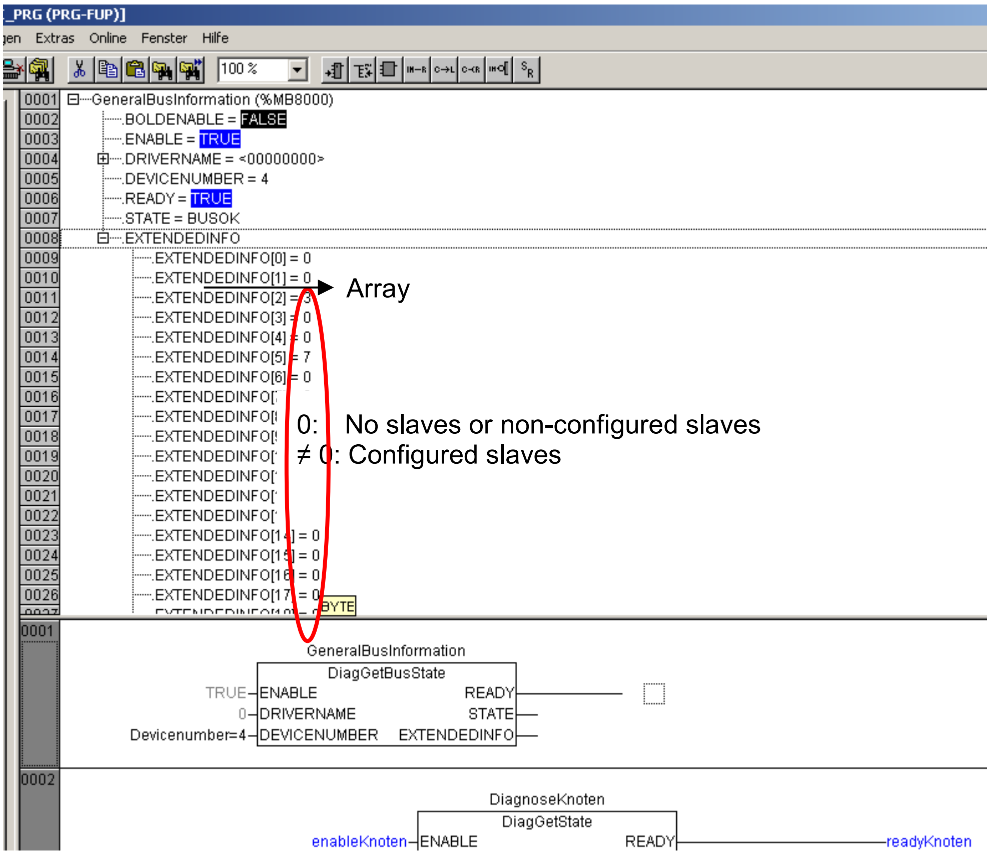
1. Log into CODESYS. To do this, click in the menu bar on **Online** > **Login**. Information about the variables is then displayed in the Variable window (Online view).

2. To start the PLC program, click **Online** > **Start** in the menu bar. Starting this program also calls up the DiagGetBusState() function block and outputs diagnostics information to the EXTENDEDINFO array.

In the online view of the Variable window, the array EXTENDEDINFO provides information on the status of the slaves. An entry is reserved in the array for each slave. The slave address is allocated to the array index. In this example, it is the slaves with the station addresses 2 and 5 that store the diagnostics information. If the device is configured as a slave, only the information for its own address is available.

Note

Display of Diagnostics Information
The diagnostic information is only displayed for the duration of one program cycle. If the diagnostic information should be available for longer, a suitable program must be written.



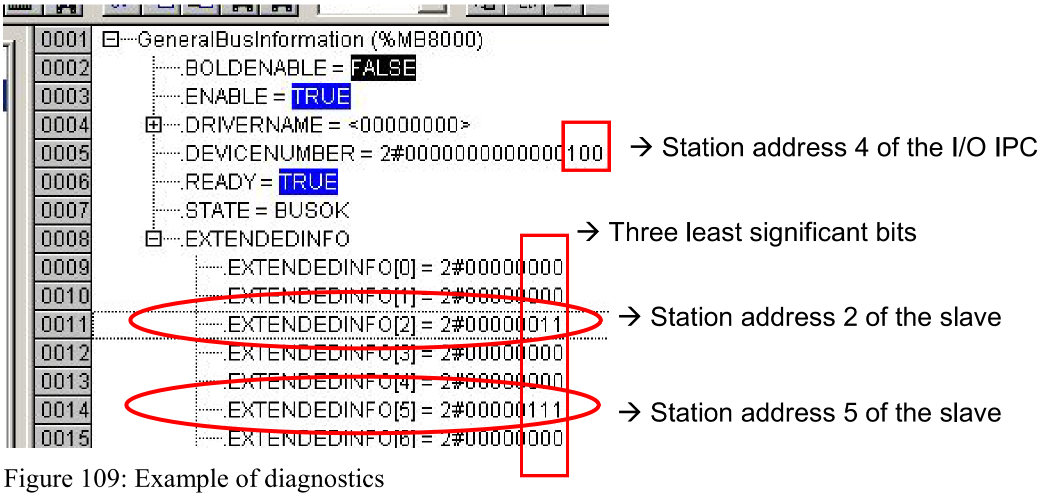
```
_PRG (PRG-FUP)]
jen  Extras  Online  Fenster  Hilfe
100 %
0001 ⊟GeneralBusInformation (%MB8000)
0002     .BOLDENABLE = FALSE
0003     .ENABLE = TRUE
0004   ⊞.DRIVERNAME = <00000000>
0005     .DEVICENUMBER = 4
0006     .READY = TRUE
0007     .STATE = BUSOK
0008   ⊟.EXTENDEDINFO
0009         .EXTENDEDINFO[0] = 0
0010         .EXTENDEDINFO[1] = 0
0011         .EXTENDEDINFO[2] = 3
0012         .EXTENDEDINFO[3] = 0
0013         .EXTENDEDINFO[4] = 0
0014         .EXTENDEDINFO[5] = 7
0015         .EXTENDEDINFO[6] = 0
0016         .EXTENDEDINFO[
0017         .EXTENDEDINFO[
0018         .EXTENDEDINFO[
0019         .EXTENDEDINFO[
0020         .EXTENDEDINFO[
0021         .EXTENDEDINFO[
0022         .EXTENDEDINFO[
0023         .EXTENDEDINFO[14] = 0
0024         .EXTENDEDINFO[15] = 0
0025         .EXTENDEDINFO[16] = 0
0026         .EXTENDEDINFO[17] = 0   BYTE

0001
                       GeneralBusInformation
                          DiagGetBusState
              TRUE─ENABLE                  READY─
                 0─DRIVERNAME              STATE─
     Devicenumber=4─DEVICENUMBER    EXTENDEDINFO─

0002
                                  DiagnoseKnoten
                                   DiagGetState
                     enableKnoten─ENABLE          READY──────readyKnoten
```

Array

0: No slaves or non-configured slaves
≠ 0: Configured slaves

Figure 108: Online view of the variable window (top window) in FUP

3. The binary code facilitates the evaluation of the individual diagnostic bits. You can have the diagnostic information of the array EXTENDEDINFO displayed as binary code. To do this, right click in the variable window and select **binary**.



```
0001 ⊟GeneralBusInformation (%MB8000)
0002     .BOLDENABLE = FALSE
0003     .ENABLE = TRUE
0004   ⊞.DRIVERNAME = <00000000>
0005     .DEVICENUMBER = 2#0000000000000100
0006     .READY = TRUE
0007     .STATE = BUSOK
0008   ⊟.EXTENDEDINFO
0009         .EXTENDEDINFO[0] = 2#00000000
0010         .EXTENDEDINFO[1] = 2#00000000
0011         .EXTENDEDINFO[2] = 2#00000011
0012         .EXTENDEDINFO[3] = 2#00000000
0013         .EXTENDEDINFO[4] = 2#00000000
0014         .EXTENDEDINFO[5] = 2#00000111
0015         .EXTENDEDINFO[6] = 2#00000000
```

→ Station address 4 of the I/O IPC

→ Three least significant bits

→ Station address 2 of the slave

→ Station address 5 of the slave

Figure 109: Example of diagnostics

4. Compare the three lowest value bits of the diagnostic information of the slaves with station addresses 2 and 5 with the bits from the following table:

Table 116: Diagnostics Information Bits

2nd bit		1st bit		0 bit	
1	0	1	0	1	0
Diagnostics information is present at the slave.	No diagnostics information is present at the slave.	Slave is active.	Slave is inactive.	Slave projected.	Slave not designed.

- The slave with station address 2 delivers the value 011. This means that the slave is configured and is active.

- The slave with station address 5 delivers the value 111. This means that the slave is configured, is active and that error information is available for this slave. Subscriber diagnostics must be performed to evaluate this error information. Refer to Section “Performing Subscriber Diagnostics using DiagGetState()” for more information about this.

Note

Diagnostic information
If READY = TRUE, then STATE indicates the current bus status with one of the following values:

BUSOK: All configured slaves exchanging data with the DP master.

BUSFAULT: One or more configured slaves is not exchanging data with the DP master
BUSNOTCOMMUNICATION: All configured slaves are not exchanging data with the DP master.

10.4.5 Performing Subscriber Diagnostics using DiagGetState()

If the bus diagnostics have revealed that an I/O module contains diagnostic information, then perform participant diagnostics on the corresponding slave. Proceed as follows:

1. Call up the function block DiagGetState() by setting the input ENABLE to “True.”

2. Specify the slave that provides the diagnostic information at the input variable BUSMEMBERID. In this example, it is the slave with the fieldbus address 5.

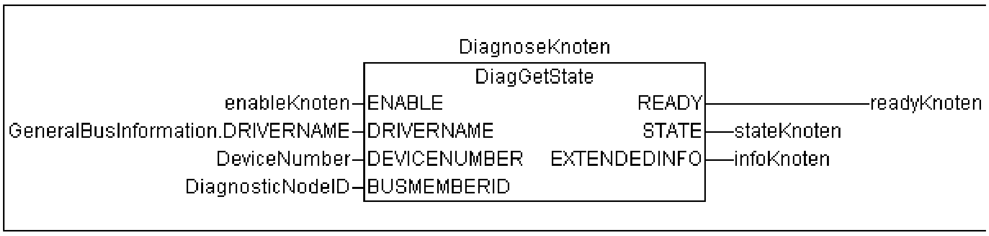


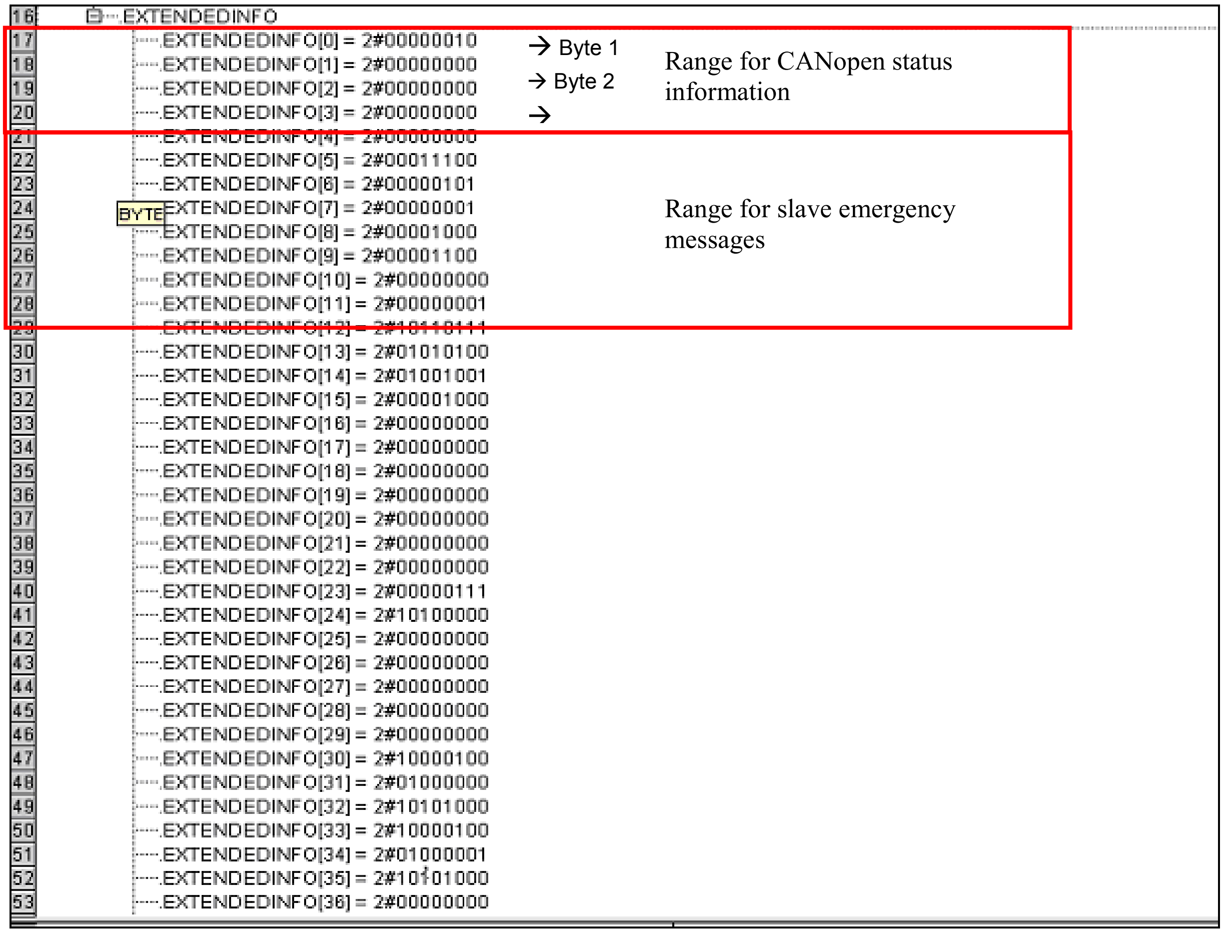
Figure 110: DiagGetState() diagnostic call

- DRIVERNAME:
  The input parameter DRIVERNAME is specified through the instance data of the function block DiagGetBusState.

- DEVICENUMBER:
  The DEVICENUMBER is device-specific and the variable "DeviceNumber" must be adapted accordingly based on the information given in the Section "Appendix" > … > "BusDiag.lib".

10.4.6 Evaluating the CANopen Diagnosis (Emergency Messages)

The array elements [0] to [3] listed in the illustration below are reserved for the CANopen status information in bytes. The emergency messages of the slaves are stored starting with array element [4].



```
EXTENDEDINFO
   .EXTENDEDINFO[0] = 2#00000010
   .EXTENDEDINFO[1] = 2#00000000
   .EXTENDEDINFO[2] = 2#00000000
   .EXTENDEDINFO[3] = 2#00000000
   .EXTENDEDINFO[4] = 2#00000000
   .EXTENDEDINFO[5] = 2#00011100
   .EXTENDEDINFO[6] = 2#00000101
   .EXTENDEDINFO[7] = 2#00000001
   .EXTENDEDINFO[8] = 2#00001000
   .EXTENDEDINFO[9] = 2#00001100
   .EXTENDEDINFO[10] = 2#00000000
   .EXTENDEDINFO[11] = 2#00000001
   .EXTENDEDINFO[12] = 2#10110111
   .EXTENDEDINFO[13] = 2#01010100
   .EXTENDEDINFO[14] = 2#01001001
   .EXTENDEDINFO[15] = 2#00001000
   .EXTENDEDINFO[16] = 2#00000000
   .EXTENDEDINFO[17] = 2#00000000
   .EXTENDEDINFO[18] = 2#00000000
   .EXTENDEDINFO[19] = 2#00000000
   .EXTENDEDINFO[20] = 2#00000000
   .EXTENDEDINFO[21] = 2#00000000
   .EXTENDEDINFO[22] = 2#00000000
   .EXTENDEDINFO[23] = 2#00000111
   .EXTENDEDINFO[24] = 2#10100000
   .EXTENDEDINFO[25] = 2#00000000
   .EXTENDEDINFO[26] = 2#00000000
   .EXTENDEDINFO[27] = 2#00000000
   .EXTENDEDINFO[28] = 2#00000000
   .EXTENDEDINFO[29] = 2#00000000
   .EXTENDEDINFO[30] = 2#10000100
   .EXTENDEDINFO[31] = 2#01000000
   .EXTENDEDINFO[32] = 2#10101000
   .EXTENDEDINFO[33] = 2#10000100
   .EXTENDEDINFO[34] = 2#01000001
   .EXTENDEDINFO[35] = 2#10101000
   .EXTENDEDINFO[36] = 2#00000000
```

BYTE
→ Byte 1
→ Byte 2
→
Range for CANopen status information
Range for slave emergency messages

Figure 111: Online view of the EXTENDEDINFO array in the binary representation

The CANopen status information and the slaves' emergency messages are described on the following pages.

Description of the Function Block's Diagnostic Information DiagGetState.EXTENDEDINFO for CANopen

The EXTENDEDINFO array contains the following status information:

Byte 0

Bit 0:	Slave entered
Bit 1:	Slave entered and configured
Bit 2:	Slave configuration invalid
Bit 3:	Diagnosis: Emergency event active
Bit 4:	Slave status “Operational”
Bit 5:	Slave status “Stop”
Bit 6:	Slave status “Preoperational”
Bit 7:	Erroneous configuration structure (from Master)

Byte 1

Bit 0:	Configuration structure incompatible with slave
Bit 1:	Slave device detection error

Byte 2

Bit 0:	An emergency message is present in the list

Byte 3

Not used

Byte 4 … 11

Last emergency message in the list

10.5 Data Exchange between Simple CAN Subscribers and PFC200 in the CANopen Network

The EDS file “Generic CAN device”, which has been reduced to the essentials, has been created to simplify the control configuration for adding CAN Layer2 device. The EDS file contains 16 send and receive PDOs, each of which has 8x1-byte entries. You only have to deactivate the configuration and monitoring telegrams typical of CANopen for these subscribers.

You can also execute the control configuration with any EDS file for CANopen.

1. To add the CANopen Master to the control configuration, right-click “COS unused[Slot]” and select “Replace element -> CANopen Master”.

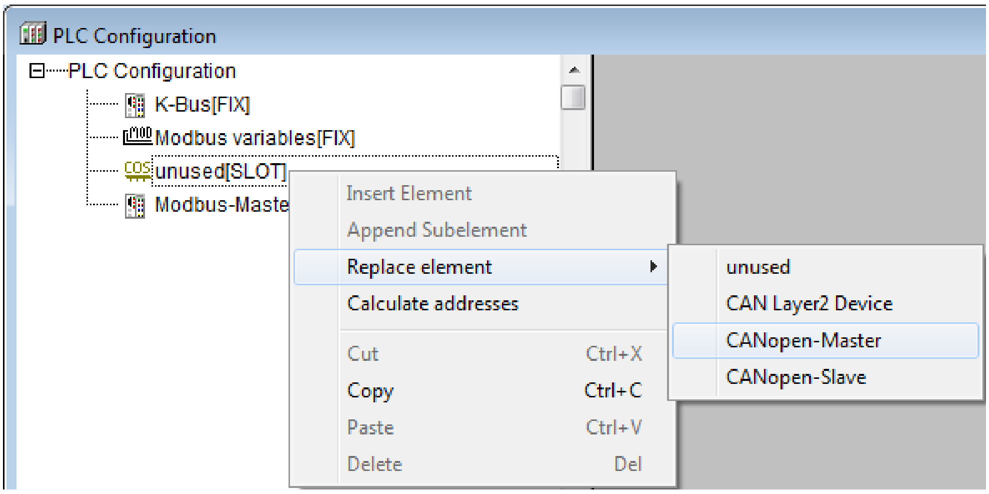
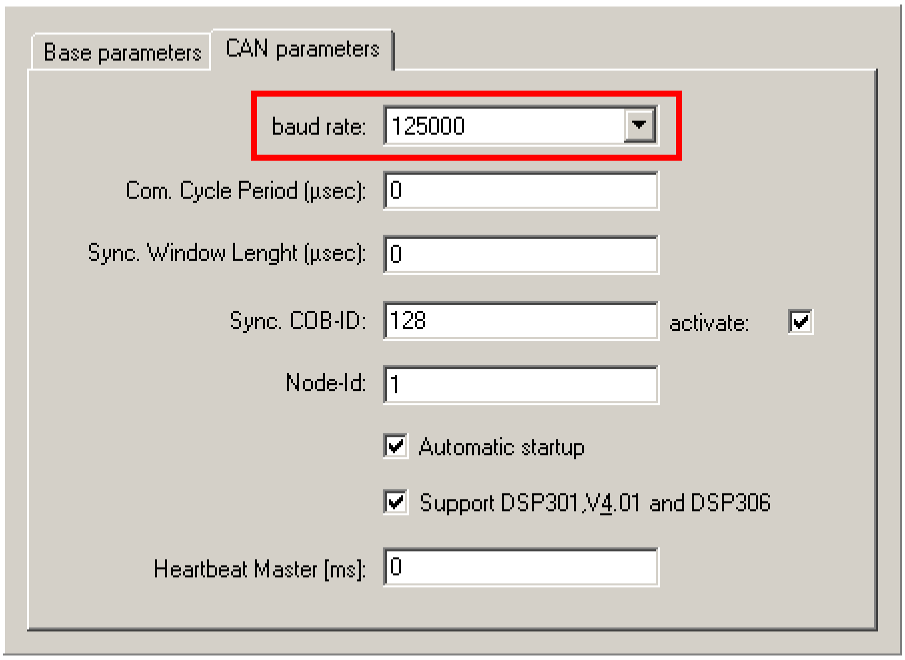


Figure 112: Attaching the CANopen master

2. In the “CAN Parameters” tab, select the required baud rate.


Base parameters | CAN parameters

baud rate: 125000

Com. Cycle Period (µsec): 0

Sync. Window Lenght (µsec): 0

Sync. COB-ID: 128 activate: ☑

Node-Id: 1

☑ Automatic startup

☑ Support DSP301,V4.01 and DSP306

Heartbeat Master [ms]: 0

Figure 113: Setting the baud rate

3. To add a slave, right-click **CANopen Master[SLOT]** and select **Attach sub-element > Generic CAN Device (EDS) …** in the contextual menu.

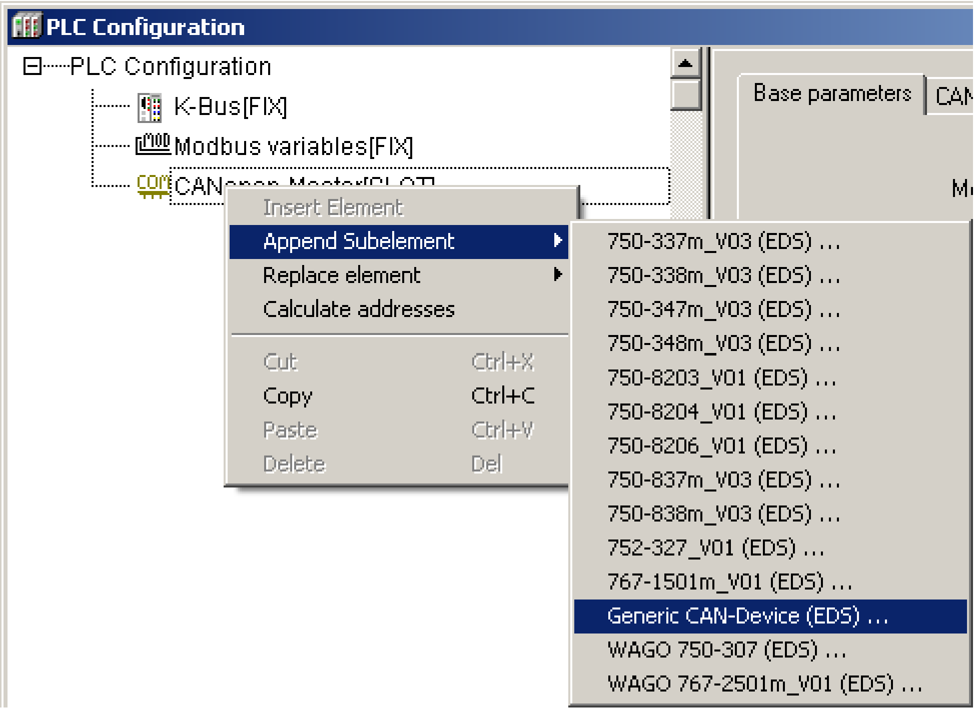
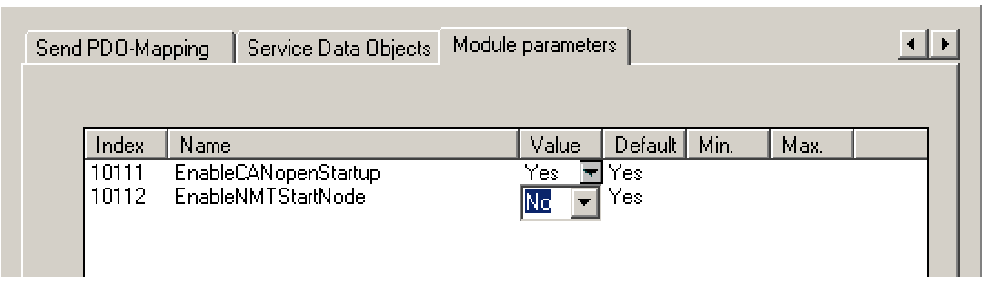


Figure 114: EDS file "Generic CAN Device"

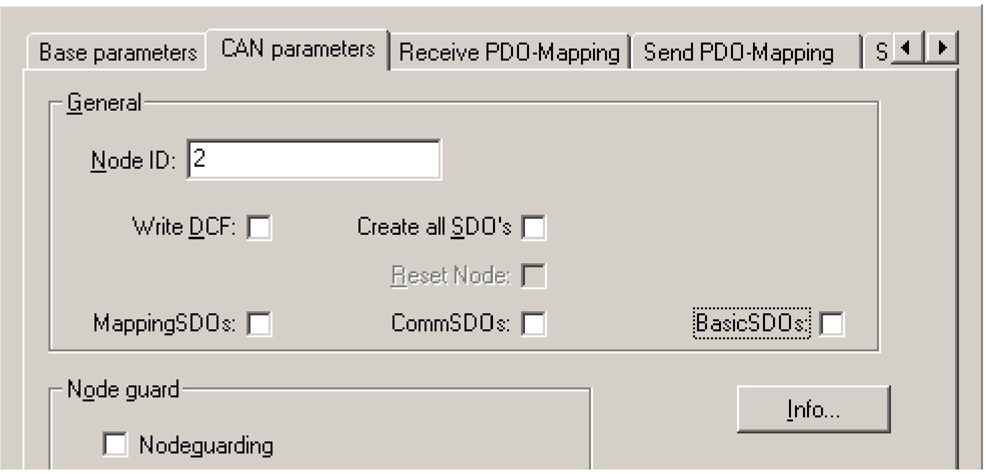
4. Open the **Module parameters** tab of the slave. For communication with simple CAN Layer2 devices, deactivate transmission of the configuration to the slave with **EnableCANopenStartup** (= "No").



Index	Name	Value	Default	Min.	Max.
10111	EnableCANopenStartup	Yes	Yes		
10112	EnableNMTStartNode	No	Yes		

Figure 115: "Module parameters" tab

5. Open the **CAN parameters** tab of the slave. Deactivate the parameters **CommSDO**, **MappingSDO**, **Basic SDO** and **Nodeguarding**.



General

Node ID: 2

Write DCF: □ Create all SDO's □

Reset Node: □

MappingSDOs: □ CommSDOs: □ BasicSDOs: □

Node guard

□ Nodeguarding

Figure 116: “CAN parameters“ tab

6. You can now use the commands of the CAN Layer2 library (see Section “CODESYS Libraries” > … > “WagoCANLayer2_01.lib”) to access the devices.
To configure the CAN frames for CAN Layer2 devices, refer to the Section “CANopen Master and Slaves” > … > “Configuring the slave PDOs”.

10.6 Data Exchange between CAN Subscribers and the PFC200 in a CAN Layer2 Network

If there are only CAN Layer2 devices in the network, the CANopen functions are not necessary. A separate device has been defined that provides only the CAN Layer2 basic functions.

1. To add the CAN Layer2 device to the control configuration, right-click “COS unused[Slot]” and select “Replace element -> CAN Layer2 Device”.

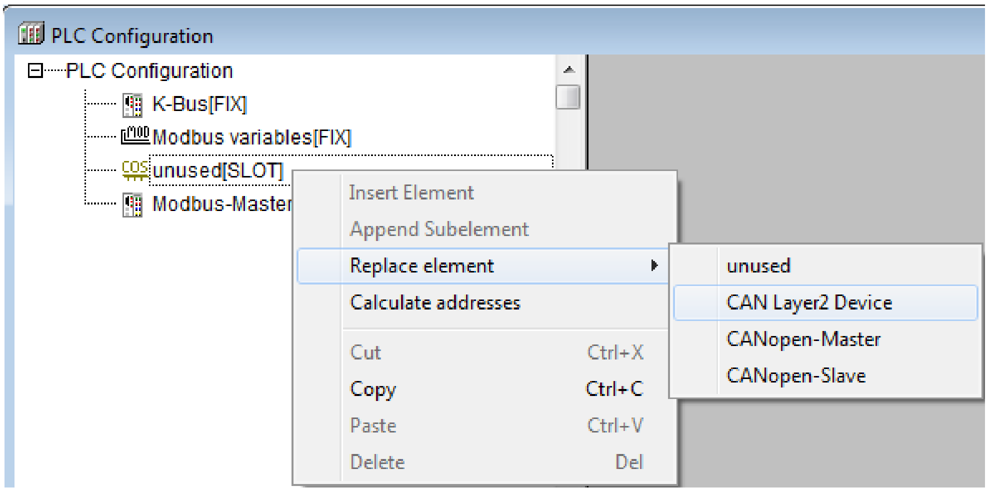


Figure 117: Attaching the CAN Layer2 device

2. The CAN interface can now be opened with the function blocks from the WagoCANLayer2_02.lib, the CAN LED set and the data exchanged.

11 Diagnostics

11.1 Operating and Status Messages

The following tables contain descriptions of all operating and status messages for the controller which are indicated by LEDs.

11.1.1 Power Supply Indicating Elements

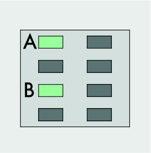


Figure 118: Power supply indicating elements

Table 117: Legend for Figure “Power Supply Indicating Elements”"

Description	Color	Description
A	Green/off	Status of system power supply voltage
B	Green/off	Status of field-side power supply voltage

Table 118: Field-Side Supply Diagnostics

Status	Explanation	Solution
Green	24V field-side supply voltage present	---
Off	No 24V field-side supply voltage present	Switch on the power supply. Check the supply voltage.

Table 119: System Power Supply Diagnistics

Status	Explanation	Solution
Green	24V system power supply voltage present	---
Off	No 24V system power supply voltage present	Switch on the power supply. Check the supply voltage.

11.1.2 Fieldbus/System Indicating Elements

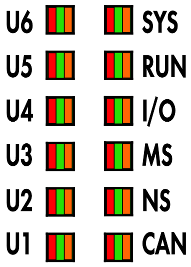


Figure 119: Indicating elements for fieldbus/system

Table 120: Diagnostics SYS LED

Status	Explanation	Solution
Green	System start completed without errors	---
Orange	Usage > Limit 1 The system is at full capacity; real-time response can no longer be guaranteed.	Try to reduce the load on the system: - Change the CODESYS program. - End any fieldbus communication that is not essential, or reconfigure the fieldbuses. - Remove any non-critical tasks from the RT area.
Orange flashing	The IP address was temporarily set to a defined value via the FIC IP function with the RST button.	Connect to the device via the standard address (192.168.1.17), or restart the device to restore the set value.
Red	Usage > Limit 2 The system is overloaded; real-time response is no longer guaranteed.	Try to reduce the load on the system: - Change the CODESYS program. - End any fieldbus communication that is not essential, or reconfigure the fieldbuses. - Remove any non-critical tasks from the RT area.

Table 121: Diagnostics RUN LED

Status	**Explanation**	**Solution**
Green	PLC program has the status “Run”.	---
Green flashing	PLC program at a debug point.	Resume the program in the linked IDE (Integrated Development Environment) using “Single step” or “Start”. If the connection has been interrupted, set the Run/Stop switch to “Stop” and then back to “Run” to enable the program to continue.
Green/red flashing	PLC is at a debug point and the Run/Stop switch has been set to “Stop”.	Set the Run/Stop switch to “Run” to enable the program to continue.
Red	No PLC-program loaded or PLC program has the status “Stop”.	Load the PLC program. Set the Run/Stop switch to “Run” to start the current program.

Table 122: Diagnostics I/O LED

Status	Explanation	Solution
Green	Data cycle on the internal data bus, normal operating status.	---
Orange flashing	Startup phase; the internal data bus is being initialized. The startup phase is indicated by rapid flashing for about 1 ... 2 seconds.	Wait until initialization has been completed.
Red	A hardware fault is present.	Contact WAGO Support.
Red flashing (2 Hz)	An error which may be able to be eliminated is present.	First, try to eliminate the error by switching the device (power supply) off and then back on. Check the entire node structure for any errors. If you cannot eliminate the error, contact WAGO Support.
Red flashing (flashing sequence)	An internal data bus error is present.	An explanation of the flashing sequence is given in the section "Diagnostics Messages (I/O LEDs)".
Off	A library was not loaded, or a library function was not called up.	Restart the device. If you cannot eliminate the error, contact WAGO Support.

Table 123: Diagnostics CAN LED

Status	Explanation	Solution
Off	The CAN interface has not been configured	---
Alternating red 50 ms / green 50 ms	Configuration in progress	---
Alternating red 200 ms / green 200 ms	Configuration invalid	Check the configuration in the CODESYS Configurator.
Green 200 ms / off 800 ms	The CANopen interface has the status “Stop”.	---
Green 200 ms / off 200 ms	The CANopen interface has the status “Preoperational”.	---
Green	The CANopen interface has the status “Operational”.	---
Red	The CANopen interface has the status “Bus Off” (short-circuit or other major fault).	Check the bus connections and the baud rate.
Error in the status “Preoperational”		
1* red flashing / 2 * green flashing	“Bus Warning Level” exceeded.	Check the wiring for the CAN bus.
2* red flashing / 2 * green flashing	“Guarding Error”, slave incorrectly configured or not available.	Check the slaves and the configuration.
3* red flashing / 2 * green flashing	“Sync Error”	Change the time interval for the synchronization message.
Error in the status “Operational”		
1* red flashing / green 800 ms on	“Bus Warning Level” exceeded.	Check the wiring for the CAN bus.
2* red flashing / green 800 ms on	“Guarding Error”	Check the slaves and the configuration.
3* red flashing / green 800 ms on	“Sync Error”	Change the time interval for the synchronization message.

11.2 Diagnostics Messages (I/O LED)

11.2.1 Flashing Sequence

A diagnosis (fault/error) is always displayed as three flashing sequences in a cyclic manner:

1. The first flashing sequence (flickering) initiates reporting of the fault/error.

2. After a short break (approx. 1 second), the second flashing sequence starts. The number of blink pulses indicates the **error code**, which describes the type of error involved.

3. After a further break the third flashing sequence is initiated. The number of blink pulses indicates the **error argument**, which provides an additional description of the error, e.g., which of the I/O modules connected to the controller exhibits an error.

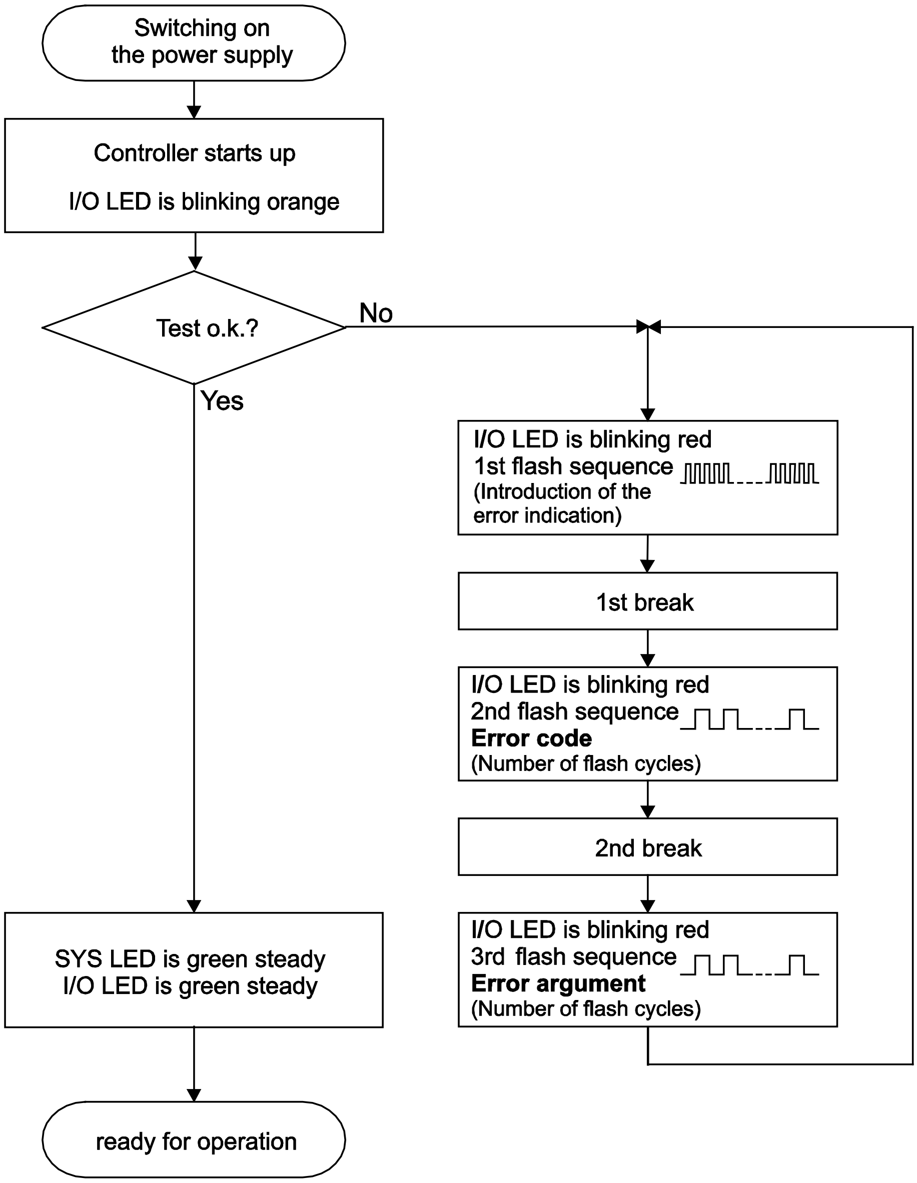


Figure 120: Flashing sequence process diagram

11.2.2 Example of a Diagnostics Message Indicated by a Blink Code

The example below illustrates the representation of a diagnostics message via the blink code. A data error is indicated on the internal data bus, caused by the removal of an I/O module located at the 6th position of the bus node.

Initiation of the Start Phase

1. The I/O LED flashes for 1 cycle at about 10 Hz (10 flashes/second).

2. This is followed by a pause of about one second.

Error Code 4: Data Error in the Internal Data Bus

3. The I/O LED flashes for 4 cycles of about 1Hz.

4. This is followed by a pause of about 1 second.

Error Argument 5: I/O Module at the 6th Slot

5. The I/O LED flashes for 5 cycles at 1 Hz.
   This indicates that a disruption has occurred at the I/O module downcircuit of the 5th I/O module.

6.The blink code starts flickering when the start phase is initiated again. If there is only one error, this process is repeated.

11.2.3 Meaning of Blink Codes and Procedures for Troubleshooting

This section describes the diagnostics presented as blink codes via the I/O LEDs.

If the diagnostics cannot be cleared by the measured specified for them, contact WAGO support. Be ready to explain to them the blink code that is displayed.

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

Table 124: Overview of Error Codes

Error code	Explanation
1	Hardware and configuration error
2	Not used
3	Internal data bus protocol error
4	Physical error on the internal data bus
5	Internal data bus initialization error
6	Design error in the node configuration
7	Not used
8	Not used
9	CPU exception error

Table 125: Error Code 1, Explanation of Blink Codes and Procedures for Troubleshooting

Error Argument	Cause	Solution
--	Invalid parameter checksum for internal data bus interface	- Switch off the power to the controller and replace it. - After this, switch the power back on.
1	Internal buffer overflow (max. amount of data exceeded) during inline code generation	- Switch off the power to the controller. - Reduce the number of I/O modules. - Switch the power back on.
2	Data type of the I/O module(s) is not supported	- Update the controller firmware. If this error persists, there is an error in the I/O module. Determine this error as follows: - Switch off the power supply. - Place the end module in the middle of the I/O modules connected to the system. - Switch the power back on. - If the I/O flashes red switch off the power supply again and place the end module in the middle of the first half of the I/O modules (toward the controller). - If the LED is no longer flashing, switch off the power supply and place the end module in the middle of the second half of the I/O modules (away from the controller). - Switch the power back on. - Repeat this procedure until you establish which I/O module is defective. Then replace that module.
3	Unknown module type of the flash program memory	- Switch off the power to the controller and replace it. - After this, switch the power back on.
4	Error occurred while writing to the flash memory	- Switch off the power to the controller and replace it. - After this, switch the power back on.
5	Error occurred while erasing a flash sector	
6	The I/O module configuration after an internal data bus reset differs from the one after the last controller startup.	- Restart the controller by first switching off the power supply and then switching it back on, or by pressing the Reset button on the controller.
7	Error occurred while writing to the serial EEPROM	- Switch off the power to the controller and replace it. - After this, switch the power back on.

Table 125: Error Code 1, Explanation of Blink Codes and Procedures for Troubleshooting

Error Argument	Cause	Solution
8	Invalid hardware/ firmware combination	
9	Invalid checksum in the serial EEPROM	
10	Fault when initializing the serial EEPROM.	
11	Error occurred while reading from the serial EEPROM	- Switch off the power supply to the controller and reduce the number of I/O modules. - After this, switch the power back on.
12	Time to access the serial EEPROM exceeded	- Switch off the power to the controller and replace it. - After this, switch the power back on.
14	Maximum number of gateway or mailbox modules exceeded.	- Switch off the power to the controller. - Reduce the number of gateway or mailbox modules. - After this, switch the power back on.

Table 126: Error Code 3, Explanation of Blink Codes and Procedures for Troubleshooting

Error Argument	Cause	Solution
--	Internal data bus communication error; defective I/O module cannot be identified	If a power supply module (e.g., 750-602) is connected to the controller, ensure that this module functions properly (see Section “LED Signaling”). If the supply module does not exhibit any errors/faults, the I/O module is defective. Identify the defective I/O module as follows: - Switch off the power supply. - Place the end module in the middle of the I/O modules connected to the system. - Switch the power back on. - If the I/O LED continues to flash red switch off the power supply again and place the end module in the middle of the first half of the I/O modules (toward the controller). If only one I/O module is left and the LED continues to flash, either this module or the controller internal data bus interface is defective. Replace the defective module or the controller. - If the LED is no longer flashing, switch off the power supply and place the end module in the middle of the second half of the I/O modules (away from the controller). - Switch the power back on. - Repeat this procedure until you establish which I/O module is defective. Then replace that module.

Table 127: Error Code 4, Explanation of Blink Codes and Procedures for Troubleshooting

Error Argument	Cause	Solution
--	Maximum permissible number of I/O modules exceeded.	- Switch off the power to the controller. - Reduce the number of I/O modules to an acceptable value. - Switch the power back on.
n*	Internal data bus disruption after the nth process data module.	- Switch off the power to the controller. - Replace the (n+1)th process data module. - Switch the power back on. I/O modules that do not provide any data are ignored (e.g., supply module without diagnostics).

Table 128: Error Code 5, Explanation of Blink Codes and Procedures for Troubleshooting

Error Argument	Cause	Solution
n*	Register communication error during internal data bus initialization	- Switch off the power to the controller. - Replace the (n+1)th process data module. - Switch the power back on. I/O modules that do not provide any data are ignored (e.g., supply module without diagnostics).

Table 129: Error Code 6, Explanation of Blink Codes and Procedures for Troubleshooting

Error Argument	Cause	Solution
5	Maximum size of the process image exceeded	- Switch off the power supply to the controller and reduce the number of I/O modules. - Switch the power back on.

Table 130: Error Code 9, Explanation of Blink Codes and Procedures for Troubleshooting

Error Argument	Cause	Solution
1	Invalid program statement	Malfunction of the program sequence. - Please contact WAGO Support.
2	Stack overflow	Malfunction of the program sequence. - Please contact WAGO Support.
3	Stack underflow	Malfunction of the program sequence. - Please contact WAGO Support.
4	Invalid event (NMI)	Malfunction of the program sequence. - Please contact WAGO Support.

12 Service

12.1 Inserting and Removing the Memory Card

12.1.1 Inserting the Memory Card

1. Use an actuating tool or a screwdriver to open the transparent cover flap by flipping it upwards. The points of application for the tool are marked by arrows.

2. Hold the memory card such that the contacts are visible on the right side and the sloping edge is at the top, as depicted in the figure below.

3. Insert the memory card in this position into the slot of the fieldbus controller.

4. Push the memory card all the way in. When you let go, the memory card will move back a little and then snap in.

5. Close the cover flap by flipping it down and pushing it in until it snaps into place.

6. You can seal the closed flap through the hole in the enclosure next to the flap.

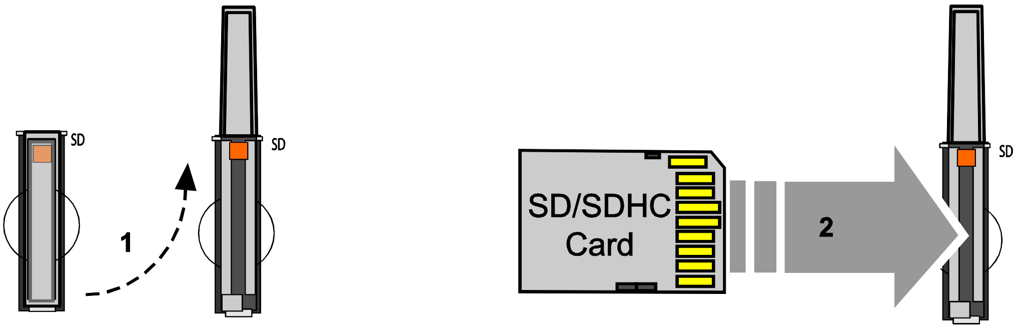


Figure 121: Inserting the memory card

12.1.2 Removing the Memory Card

1. First, remove any seal that may be in place.

2. Use an actuating tool or a screwdriver to open the transparent cover flap by flipping it upwards. The points of application for the tool are marked by arrows.

3. To remove the memory card you must first push it into the slot. This releases the mechanical locking mechanism.

4. As soon as you let go of the memory card, the memory card is pushed out a bit and you can remove it.

5. Close the cover flap by flipping it down and pushing it in until it snaps into place.

13 Removal

⚠ CAUTION

Risk of injury due to sharp-edged blade contacts!
The blade contacts are sharp-edged. Handle the I/O module carefully to prevent injury.

13.1 Removing Devices

NOTICE

Perform work on devices only if they are de-energized!
Working on energized devices can damage them. Therefore, turn off the power supply before working on the devices.

13.1.1 Removing the Fieldbus Coupler/Controller

1. Use a screwdriver blade to turn the locking disc until the nose of the locking disc no longer engages behind the carrier rail.

2. Remove the fieldbus coupler/controller from the assembly by pulling the release tab.

Electrical connections for data or power contacts to adjacent I/O modules are disconnected when removing the fieldbus coupler/controller.

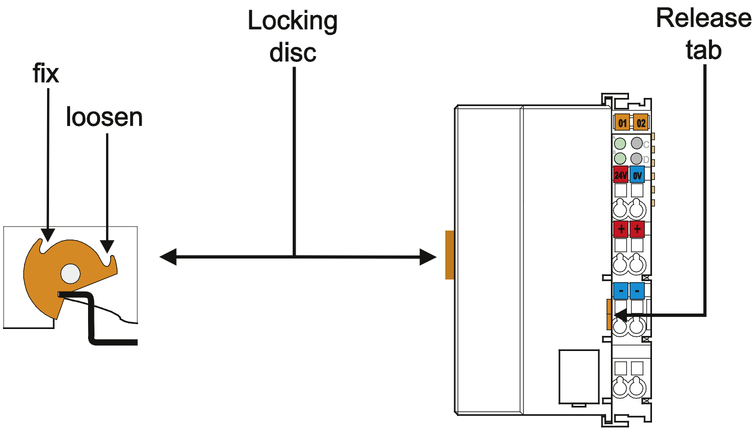


Figure 122: Release Tab of Controller

Note

Do not take the controller enclosure apart!
The enclosure sections are firmly joined. The feed-in section with the CAGE CLAMP® connections cannot be separated from the other enclosure section.

13.1.2 Removing the I/O Module

1. Remove the I/O module from the assembly by pulling the release tab.

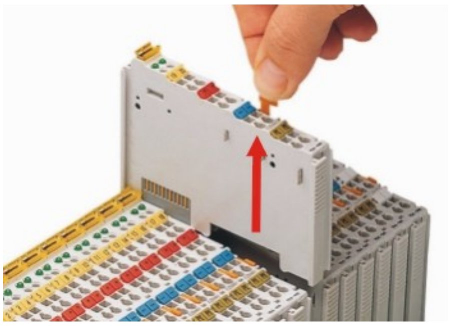


Figure 123: Removing the I/O Module (Example)

Electrical connections for data or power jumper contacts are disconnected when removing the I/O module.

Note

Do not take the controller enclosure apart!

The enclosure sections are firmly joined. The feed-in section with the CAGE CLAMP® connections cannot be separated from the other enclosure section.

14 Appendix

14.1 Structure of Process Data for the I/O Modules

The process image for the I/O modules on the internal data bus is built up word-by-word in the PFC200 controller (with word alignment). The internal mapping method for data greater than one byte conforms to Intel formats.

The following section describes the representation for WAGO-I/O SYSTEM 750 (750 and 753 Series) I/O modules in the process image, as well as the configuration of the process values.

NOTICE

Equipment damage due to incorrect address!

To prevent any damage to the device in the field you must always take the process data for all previous byte or bit-oriented I/O modules into account when addressing an I/O module at any position in the fieldbus node.

Note

No direct access from fieldbus to the process image for I/O modules!

Any data that is required from the I/O module process image must be explicitly mapped in the CODESYS program to the data in the fieldbus process image and vice versa! Direct access is not possible!

14.1.1 Digital Input Modules

Digital input modules supply one bit of data per channel to specify the signal state for the corresponding channel. These bits are mapped into the Input Process Image.

Some digital modules have an additional diagnostic bit per channel in the Input Process Image. The diagnostic bit is used for detecting faults that occur (e.g., wire breaks and/or short circuits).

When analog input modules are also present in the node, the digital data is always appended after the analog data in the Input Process Image, grouped into bytes.

14.1.1.1 1 Channel Digital Input Module with Diagnostics

750-435

Table 131: 1 Channel Digital Input Module with Diagnostics

Input Process Image							
Bit 7	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**
						Diagnostic bit S 1	Data bit DI 1

14.1.1.2 2 Channel Digital Input Modules

750-400, -401, -405, -406, -410, -411, -412, -427, -438, (and all variations),
753-400, -401, -405, -406, -410, -411, -412, -427

Table 132: 2 Channel Digital Input Modules

Input Process Image							
Bit 7	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**
						Data bit DI 2 Channel 2	Data bit DI 1 Channel 1

14.1.1.3 2 Channel Digital Input Module with Diagnostics

750-419, -421, -424, -425,
753-421, -424, -425

Table 133: 2 Channel Digital Input Module with Diagnostics

Input Process Image							
Bit 7	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**
				Diagnostic bit S 2 Channel 2	Diagnostic bit S 1 Channel 1	Data bit DI 2 Channel 2	Data bit DI 1 Channel 1

14.1.1.4 2 Channel Digital Input Module with Diagnostics and Output Process Data

750-418,
753-418

The digital input module supplies a diagnostic and acknowledge bit for each input channel. If a fault condition occurs, the diagnostic bit is set. After the fault condition is cleared, an acknowledge bit must be set to re-activate the input. The diagnostic data and input data bit is mapped in the Input Process Image, while the acknowledge bit is in the Output Process Image.

Table 134: 2 Channel Digital Input Module with Diagnostics and Output Process Data

Input Process Image							
Bit 7	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**
				Diagnostic bit S 2 Channel 2	Diagnostic bit S 1 Channel 1	Data bit DI 2 Channel 2	Data bit DI 1 Channel 1

Output Process Image							
Bit 7	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**
				Acknowledge-ment bit Q 2 Channel 2	Acknowledge-ment bit Q 1 Channel 1	0	0

14.1.1.5 4 Channel Digital Input Modules

750-402, -403, -408, -409, -414, -415, -422, -423, -428, -432, -433, -1420, -1421, -1422, -1423
753-402, -403, -408, -409, -415, -422, -423, -428, -432, -433, -440

Table 135: 4 Channel Digital Input Modules

Input Process Image							
Bit 7	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**
				Data bit DI 4 Channel 4	Data bit DI 3 Channel 3	Data bit DI 2 Channel 2	Data bit DI 1 Channel 1

14.1.1.6 8 Channel Digital Input Modules

750-430, -431, -436, -437, -1415, -1416, -1417, -1418
753-430, -431, -434

Table 136: 8 Channel Digital Input Modules

Input Process Image							
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Data bit DI 8 Channel 8	Data bit DI 7 Channel 7	Data bit DI 6 Channel 6	Data bit DI 5 Channel 5	Data bit DI 4 Channel 4	Data bit DI 3 Channel 3	Data bit DI 2 Channel 2	Data bit DI 1 Channel 1

14.1.1.7 8 Channel Digital Input Module PTC with Diagnostics and Output Process Data

750-1425

The digital input module PTC provides via one logical channel 2 byte for the input and output process image.

The signal state of PTC inputs DI1 … DI8 is transmitted to the fieldbus coupler/controller via input data byte D0.
The fault conditions are transmitted via input data byte D1.

The channels 1 … 8 are switched on or off via the output data byte D1. The output data byte D0 is reserved and always has the value “0“.

Table 137: 8 Channel Digital Input Module PTC with Diagnostics and Output Process Data

Input Process Image															
Input Byte D0								**Input Byte D1**							
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Signal status DI 8 Channel 8	Signal status DI 7 Channel 7	Signal status DI 6 Channel 6	Signal status DI 5 Channel 5	Signal status DI 4 Channel 4	Signal status DI 3 Channel 3	Signal status DI 2 Channel 2	Signal status DI 1 Channel 1	Wire break/ short circuit DB/KS 8 Channel 8	Wire break/ short circuit DB/KS 7 Channel 7	Wire break/ short circuit DB/KS 6 Channel 6	Wire break/ short circuit DB/KS 5 Channel 5	Wire break/ short circuit DB/KS 4 Channel 4	Wire break/ short circuit DB/KS 3 Channel 3	Wire break/ short circuit DB/KS 2 Channel 2	Wire break/ short circuit DB/KS 1 Channel 1

Output Process Image															
Output Byte D0								**Output Byte D1**							
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	0	0	0	0	0	0	0	DI Off 8 Channel 8 0: Channel ON 1: Channel OFF	DI Off 7 Channel 7 0: Channel ON 1: Channel OFF	DI Off 6 Channel 6 0: Channel ON 1: Channel OFF	DI Off 5 Channel 5 0: Channel ON 1: Channel OFF	DI Off 4 Channel 4 0: Channel ON 1: Channel OFF	DI Off 3 Channel 3 0: Channel ON 1: Channel OFF	DI Off 2 Channel 2 0: Channel ON 1: Channel OFF	DI Off 1 Channel 1 0: Channel ON 1: Channel OFF

14.1.1.8 16 Channel Digital Input Modules

750-1400, -1402, -1405, -1406, -1407

Table 138: 16 Channel Digital Input Modules

Input Process Image															
Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Data bit DI 16 Channel 16	Data bit DI 15 Channel 15	Data bit DI 14 Channel 14	Data bit DI 13 Channel 13	Data bit DI 12 Channel 12	Data bit DI 11 Channel 11	Data bit DI 10 Channel 10	Data bit DI 9 Channel 9	Data bit DI 8 Channel 8	Data bit DI 7 Channel 7	Data bit DI 6 Channel 6	Data bit DI 5 Channel 5	Data bit DI 4 Channel 4	Data bit DI 3 Channel 3	Data bit DI 2 Channel 2	Data bit DI 1 Channel 1

14.1.2 Digital Output Modules

Digital output modules use one bit of data per channel to control the output of the corresponding channel. These bits are mapped into the Output Process Image.

Some digital modules have an additional diagnostic bit per channel in the Input Process Image. The diagnostic bit is used for detecting faults that occur (e.g., wire breaks and/or short circuits). For modules with diagnostic bit is set, also the data bits have to be evaluated.

When analog output modules are also present in the node, the digital image data is always appended after the analog data in the Output Process Image, grouped into bytes.

14.1.2.1 1 Channel Digital Output Module with Input Process Data

750-523

The digital output modules deliver 1 bit via a process value Bit in the output process image, which is illustrated in the input process image. This status image shows "manual mode".

Table 139: 1 Channel Digital Output Module with Input Process Data

Input Process Image							
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
						not used	Status bit “Manual Operation“

Output Process Image							
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
						not used	controls DO 1 Channel 1

14.1.2.2 2 Channel Digital Output Modules

750-501, -502, -509, -512, -513, -514, -517, -535, (and all variations),
753-501, -502, -509, -512, -513, -514, -517

Table 140: 2 Channel Digital Output Modules

Output Process Image							
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
						controls DO 2 Channel 2	controls DO 1 Channel 1

14.1.2.3 2 Channel Digital Input Modules with Diagnostics and Input Process Data

750-507 (-508), -522,
753-507

The digital output modules have a diagnostic bit for each output channel. When an output fault condition occurs (i.e., overload, short circuit, or broken wire), a diagnostic bit is set. The diagnostic data is mapped into the Input Process Image, while the output control bits are in the Output Process Image.

Table 141: 2 Channel Digital Input Modules with Diagnostics and Input Process Data

Input Process Image							
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
						Diagnostic bit S 2 Channel 2	Diagnostic bit S 1 Channel 1

Output Process Image							
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
						controls DO 2 Channel 2	controls DO 1 Channel 1

750-506,
753-506

The digital output module has 2-bits of diagnostic information for each output channel. The 2-bit diagnostic information can then be decoded to determine the exact fault condition of the module (i.e., overload, a short circuit, or a broken wire). The 4-bits of diagnostic data are mapped into the Input Process Image, while the output control bits are in the Output Process Image.

Table 142: 2 Channel Digital Input Modules with Diagnostics and Input Process Data 75x-506

Input Process Image							
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
				Diagnostic bit S 3 Channel 2	Diagnostic bit S 2 Channel 2	Diagnostic bit S 1 Channel 1	Diagnostic bit S 0 Channel 1

Diagnostic bits S1/S0, S3/S2: = ‘00’	standard mode
Diagnostic bits S1/S0, S3/S2: = ‘01’	no connected load/short circuit against +24 V
Diagnostic bits S1/S0, S3/S2: = ‘10’	Short circuit to ground/overload

Output Process Image							
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
				not used	not used	controls DO 2 Channel 2	controls DO 1 Channel 1

14.1.2.4 4 Channel Digital Output Modules

750-504, -516, -519, -531,
753-504, -516, -531, -540

Table 143: 4 Channel Digital Output Modules

Output Process Image							
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
				controls DO 4 Channel 4	controls DO 3 Channel 3	controls DO 2 Channel 2	controls DO 1 Channel 1

14.1.2.5 4 Channel Digital Output Modules with Diagnostics and Input Process Data

750-532

The digital output modules have a diagnostic bit for each output channel. When an output fault condition occurs (i.e., overload, short circuit, or broken wire), a diagnostic bit is set. The diagnostic data is mapped into the Input Process Image, while the output control bits are in the Output Process Image.

Table 144: 4 Channel Digital Output Modules with Diagnostics and Input Process Data

Input Process Image							
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
				Diagnostic bit S 4 Channel 4	Diagnostic bit S 3 Channel 3	Diagnostic bit S 2 Channel 2	Diagnostic bit S 1 Channel 1

Diagnostic bit S = ‘0’	no Error
Diagnostic bit S = ‘1’	overload, short circuit, or broken wire

Output Process Image							
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
				controls DO 4 Channel 4	controls DO 3 Channel 3	controls DO 2 Channel 2	controls DO 1 Channel 1

14.1.2.6 8 Channel Digital Output Module

750-530, -536, -1515, -1516
753-530, -534

Table 145: 8 Channel Digital Output Module

Output Process Image							
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
controls DO 8 Channel 8	controls DO 7 Channel 7	controls DO 6 Channel 6	controls DO 5 Channel 5	controls DO 4 Channel 4	controls DO 3 Channel 3	controls DO 2 Channel 2	controls DO 1 Channel 1

14.1.2.7 8 Channel Digital Output Modules with Diagnostics and Input Process Data

750-537

The digital output modules have a diagnostic bit for each output channel. When an output fault condition occurs (i.e., overload, short circuit, or broken wire), a diagnostic bit is set. The diagnostic data is mapped into the Input Process Image, while the output control bits are in the Output Process Image.

Table 146: 8 Channel Digital Output Modules with Diagnostics and Input Process Data

Input Process Image							
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Diagnostic bit S 8 Channel 8	Diagnostic bit S 7 Channel 7	Diagnostic bit S 6 Channel 6	Diagnostic bit S 5 Channel 5	Diagnostic bit S 4 Channel 4	Diagnostic bit S 3 Channel 3	Diagnostic bit S 2 Channel 2	Diagnostic bit S 1 Channel 1

Diagnostic bit S = ‘0’	no Error
Diagnostic bit S = ‘1’	overload, short circuit, or broken wire

Output Process Image							
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
controls DO 8 Channel 8	controls DO 7 Channel 7	controls DO 6 Channel 6	controls DO 5 Channel 5	controls DO 4 Channel 4	controls DO 3 Channel 3	controls DO 2 Channel 2	controls DO 1 Channel 1

14.1.2.8 16 Channel Digital Output Modules

750-1500, -1501, -1504, -1505

Table 147: 16 Channel Digital Output Modules

Output Process Image															
Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
controls DO 16 Channel 16	controls DO 15 Channel 15	controls DO 14 Channel 14	controls DO 13 Channel 13	controls DO 12 Channel 12	controls DO 11 Channel 11	controls DO 10 Channel 10	controls DO 9 Channel 9	controls DO 8 Channel 8	controls DO 7 Channel 7	controls DO 6 Channel 6	controls DO 5 Channel 5	controls DO 4 Channel 4	controls DO 3 Channel 3	controls DO 2 Channel 2	controls DO 1 Channel 1

14.1.2.9 8 Channel Digital Input/Output Modules

750-1502, -1506

Table 148: 8 Channel Digital Input/Output Modules

Input Process Image							
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Data bit DI 8 Channel 8	Data bit DI 7 Channel 7	Data bit DI 6 Channel 6	Data bit DI 5 Channel 5	Data bit DI 4 Channel 4	Data bit DI 3 Channel 3	Data bit DI 2 Channel 2	Data bit DI 1 Channel 1

Output Process Image							
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
controls DO 8 Channel 8	controls DO 7 Channel 7	controls DO 6 Channel 6	controls DO 5 Channel 5	controls DO 4 Channel 4	controls DO 3 Channel 3	controls DO 2 Channel 2	controls DO 1 Channel 1

14.1.3 Analog Input Modules

The analog input modules provide 16-bit measured data and 8 control/status bits per channel.

The PFC200 controller only uses the 8 control/status bits internally for configuration/parameterization (e.g., via WAGO-I/O-*CHECK*).

Therefore, only the 16-bit measurement values for each channel are in Intel format and are mapped by word in the input process image for the PFC200 controller.

When digital input modules are also present in the node, the analog input data is always mapped into the Input Process Image in front of the digital data.

Information

Information on the structure of control and status bytes
For detailed information on the structure of a particular I/O module's control/status bytes, please refer to that module's manual. Manuals for each module can be found on the Internet at www.wago.com.

14.1.3.1 1 Channel Analog Input Modules

750-491, (and all variations)

Table 149: 1 Channel Analog Input Modules

Input Process Image			
Offset	**Byte Destination**		**Description**
	High Byte	**Low Byte**	
0	D1	D0	Measured Value U_D
1	D3	D2	Measured Value U_{ref}

14.1.3.2 2 Channel Analog Input Modules

750-452, -454, -456, -461, -462, -465, -466, -467, -469, -472, -474, -475, 476, -477, -478, -479, -480, -481, -483, -485, -492, (and all variations),
753-452, -454, -456, -461, -465, -466, -467, -469, -472, -474, -475, 476, -477, 478, -479, -483, -492, (and all variations)

Table 150: 2 Channel Analog Input Modules

Input Process Image			
Offset	**Byte Destination**		**Description**
	High Byte	**Low Byte**	
0	D1	D0	Measured Value Channel 1
1	D3	D2	Measured Value Channel 2

14.1.3.3 4 Channel Analog Input Modules

750-450, -453, -455, -457, -459, -460, -468, (and all variations),
753-453, -455, -457, -459

Table 151: 4 Channel Analog Input Modules

Input Process Image			
Offset	**Byte Destination**		**Description**
	High Byte	**Low Byte**	
0	D1	D0	Measured Value Channel 1
1	D3	D2	Measured Value Channel 2
2	D5	D4	Measured Value Channel 3
3	D7	D6	Measured Value Channel 4

14.1.3.4 3-Phase Power Measurement Module

750-493

The above Analog Input Modules have a total of 9 bytes of user data in both the Input and Output Process Image (6 bytes of data and 3 bytes of control/status). The following tables illustrate the Input and Output Process Image, which has a total of 6 words mapped into each image.
Word alignment is applied.

Table 152: 3-Phase Power Measurement Module

Input Process Image			
Offset	**Byte Destination**		**Description**
	High Byte	**Low Byte**	
0	-	S0	Status byte 0
1	D1	D0	Input data word 1
2	-	S1	Status byte 1
3	D3	D2	Input data word 2
4	-	S2	Status byte 2
5	D5	D4	Input data word 3

Output Process Image			
Offset	**Byte Destination**		**Description**
	High Byte	**Low Byte**	
0	-	C0	Control byte 0
1	D1	D0	Output data word 1
2	-	C1	Control byte 1
3	D3	D2	Output data word 2
4	-	C2	Control byte 2
5	D5	D4	Output data word 3

14.1.3.5 8 Channel Analog Input Modules

750-451

Table 153: 8 Channel Analog Input Modules

Input Process Image			
Offset	**Byte Destination**		**Description**
	High Byte	**Low Byte**	
0	D1	D0	Measured Value Channel 1
1	D3	D2	Measured Value Channel 2
2	D5	D4	Measured Value Channel 3
3	D7	D6	Measured Value Channel 4
4	D9	D8	Measured Value Channel 5
5	D11	D10	Measured Value Channel 6
6	D13	D12	Measured Value Channel 7
7	D15	D14	Measured Value Channel 8

14.1.4 Analog Output Modules

The analog output modules provide 16-bit output values and 8 control/status bits per channel.

The PFC200 controller only uses the 8 control/status bits internally for configuration/parameterization (e.g., via WAGO-I/O-*CHECK*).

Therefore, only the 16-bit measurement values for each channel are in Intel format and are mapped by word in the output process image for the PFC200 controller.

When digital output modules are also present in the node, the analog output data is always mapped into the Output Process Image in front of the digital data.

Information

Information on the structure of control and status bytes
For detailed information on the structure of a particular I/O module's control/status bytes, please refer to that module's manual. Manuals for each module can be found on the Internet at www.wago.com.

14.1.4.1 2 Channel Analog Output Modules

750-550, -552, -554, -556, -560, -562, 563, -585, (and all variations),
753-550, -552, -554, -556

Table 154: 2 Channel Analog Output Modules

Output Process Image			
Offset	**Byte Destination**		**Description**
	High Byte	**Low Byte**	
0	D1	D0	Output Value Channel 1
1	D3	D2	Output Value Channel 2

14.1.4.2 4 Channel Analog Output Modules

750-553, -555, -557, -559,
753-553, -555, -557, -559

Table 155: 4 Channel Analog Output Modules

Output Process Image			
Offset	**Byte Destination**		**Description**
	High Byte	**Low Byte**	
0	D1	D0	Output Value Channel 1
1	D3	D2	Output Value Channel 2
2	D5	D4	Output Value Channel 3
3	D7	D6	Output Value Channel 4

14.1.5 Specialty Modules

WAGO has a host of Specialty I/O modules that perform various functions. With individual modules beside the data bytes also the control/status byte is mapped in the process image.

The control/status byte is required for the bidirectional data exchange of the module with the higher-ranking control system. The control byte is transmitted from the control system to the module and the status byte from the module to the control system.
This allows, for example, setting of a counter with the control byte or displaying of overshooting or undershooting of the range with the status byte.

The control/status byte always is in the process image in the Low byte.

Information

Information to the structure of the Control/Status byte
For detailed information about the structure of a particular module’s control/status byte, please refer to that module’s manual. Manuals for each module can be found on the Internet under: www.wago.com.

14.1.5.1 Counter Modules

750-404, (and all variations except of /000-005),
753-404, (and variation /000-003)

The above Counter Modules have a total of 5 bytes of user data in both the Input and Output Process Image (4 bytes of counter data and 1 byte of control/status). The counter value is supplied as 32 bits. The following tables illustrate the Input and Output Process Image, which has a total of 3 words mapped into each image. Word alignment is applied.

Table 156: Counter Modules 750-404, (and all variations except of /000-005),
753-404, (and variation /000-003)

Input Process Image			
Offset	Byte Destination		Description
	High Byte	Low Byte	
0	-	S	Status byte
1	D1	D0	Counter value
2	D3	D2	

Output Process Image			
Offset	Byte Destination		Description
	High Byte	Low Byte	
0	-	C	Control byte
1	D1	D0	Counter setting value
2	D3	D2	

750-404/000-005

The above Counter Modules have a total of 5 bytes of user data in both the Input and Output Process Image (4 bytes of counter data and 1 byte of control/ status). The two counter values are supplied as 16 bits. The following tables illustrate the Input and Output Process Image, which has a total of 3 words mapped into each image. Word alignment is applied.

Table 157: Counter Modules 750-404/000-005

Input Process Image			
Offset	**Byte Destination**		**Description**
	High Byte	**Low Byte**	
0	-	S	Status byte
1	D1	D0	Counter Value of Counter 1
2	D3	D2	Counter Value of Counter 2

Output Process Image			
Offset	**Byte Destination**		**Description**
	High Byte	**Low Byte**	
0	-	C	Control byte
1	D1	D0	Counter Setting Value of Counter 1
2	D3	D2	Counter Setting Value of Counter 2

750-638,
753-638

The above Counter Modules have a total of 6 bytes of user data in both the Input and Output Process Image (4 bytes of counter data and 2 bytes of control/status). The two counter values are supplied as 16 bits. The following tables illustrate the Input and Output Process Image, which has a total of 4 words mapped into each image. Word alignment is applied.

Table 158: Counter Modules 750-638, 753-638

Input Process Image			
Offset	**Byte Destination**		**Description**
	High Byte	**Low Byte**	
0	-	S0	Status byte von Counter 1
1	D1	D0	Counter Value von Counter 1
2	-	S1	Status byte von Counter 2
3	D3	D2	Counter Value von Counter 2

Output Process Image			
Offset	**Byte Destination**		**Description**
	High Byte	**Low Byte**	
0	-	C0	Control byte von Counter 1
1	D1	D0	Counter Setting Value von Counter 1
2	-	C1	Control byte von Counter 2
3	D3	D2	Counter Setting Value von Counter 2

14.1.5.2 Pulse Width Modules

750-511, (and all variations /xxx-xxx)

The above Pulse Width modules have a total of 6 bytes of user data in both the Input and Output Process Image (4 bytes of channel data and 2 bytes of control/ status). The two channel values are supplied as 16 bits. Each channel has its own control/status byte. The following table illustrates the Input and Output Process Image, which has a total of 4 words mapped into each image. Word alignment is applied.

Table 159: Pulse Width Modules 750-511, /xxx-xxx

Input and Output Process			
Offset	**Byte Destination**		**Description**
	High Byte	**Low Byte**	
0	-	C0/S0	Control/Status byte of Channel 1
1	D1	D0	Data Value of Channel 1
2	-	C1/S1	Control/Status byte of Channel 2
3	D3	D2	Data Value of Channel 2

14.1.5.3 Serial Interface Modules with alternative Data Format

750-650, (and the variations /000-002, -004, -006, -009, -010, -011, -012, -013),
750-651, (and the variations /000-001, -002, -003),
750-653, (and the variations /000-002, -007),
753-650, -653

Note

The process image of the / 003-000-variants depends on the parameterized operating mode!
With the freely parameterizable variations /003 000 of the serial interface modules, the desired operation mode can be set. Dependent on it, the process image of these modules is then the same, as from the appropriate variation.

The above Serial Interface Modules with alternative data format have a total of 4 bytes of user data in both the Input and Output Process Image (3 bytes of serial data and 1 byte of control/status). The following table illustrates the Input and Output Process Image, which have a total of 2 words mapped into each image. Word alignment is applied.

Table 160: Serial Interface Modules with alternative Data Format

Input and Output Process Image				
Offset	**Byte Destination**		**Description**	
	High Byte	**Low Byte**		
0	D0	C/S	Data byte	Control/status byte
1	D2	D1	Data bytes	

14.1.5.4 Serial Interface Modules with Standard Data Format

750-650/000-001, -014, -015, -016
750-653/000-001, -006

The above Serial Interface Modules with Standard Data Format have a total of 6 bytes of user data in both the Input and Output Process Image (5 bytes of serial data and 1 byte of control/status). The following table illustrates the Input and Output Process Image, which have a total of 3 words mapped into each image. Word alignment is applied.

Table 161: Serial Interface Modules with Standard Data Format

Input and Output Process Image				
Offset	**Byte Destination**		**Description**	
	High Byte	**Low Byte**		
0	D0	C/S	Data byte	Control/status byte
1	D2	D1	Data bytes	
2	D4	D3		

14.1.5.5 Data Exchange Module

750-654, (and the variation /000-001)

The Data Exchange modules have a total of 4 bytes of user data in both the Input and Output Process Image. The following tables illustrate the Input and Output Process Image, which has a total of 2 words mapped into each image.
Word alignment is applied.

Table 162: Data Exchange Module

Input and Output Process Image			
Offset	**Byte Destination**		**Description**
	High Byte	**Low Byte**	
0	D1	D0	Data bytes
1	D3	D2	

14.1.5.6 SSI Transmitter Interface Modules

750-630 (and all variations)

Note

The process image of the / 003-000-variants depends on the parameterized operating mode!
The operating mode of the configurable /003-000 I/O module versions can be set. Based on the operating mode, the process image of these I/O modules is then the same as that of the respective version.

The above SSI Transmitter Interface modules have a total of 4 bytes of user data in the Input Process Image, which has 2 words mapped into the image.
Word alignment is applied.

Table 163: SSI Transmitter Interface Modules

Input Process Image			
Offset	Byte Destination		Description
	High Byte	Low Byte	
0	D1	D0	Data bytes
1	D3	D2	

14.1.5.7 Incremental Encoder Interface Modules

750-631/000-004, -010, -011

The above Incremental Encoder Interface modules have 5 bytes of input data and 3 bytes of output data. The following tables illustrate the Input and Output Process Image, which have 4 words into each image. Word alignment is applied.

Table 164: Incremental Encoder Interface Modules 750-631/000-004, --010, -011

Input Process Image				
Offset	Byte Destination		Description	
	High Byte	Low Byte		
0	-	S	not used	Status byte
1	D1	D0	Counter word	
2	-	-	not used	
3	D4	D3	Latch word	

Output Process Image				
Offset	Byte Destination		Description	
	High Byte	Low Byte		
0	-	C	not used	Control byte
1	D1	D0	Counter setting word	
2	-	-	not used	
3	-	-	not used	

750-634

The above Incremental Encoder Interface module has 5 bytes of input data (6 bytes in cycle duration measurement mode) and 3 bytes of output data. The following tables illustrate the Input and Output Process Image, which has 4 words mapped into each image. Word alignment is applied.

Table 165: Incremental Encoder Interface Modules 750-634

Input Process Image				
Offset	Byte Destination		Description	
	High Byte	Low Byte		
0	-	S	not used	Status byte
1	D1	D0	Counter word	
2	-	(D2) *)	not used	(Periodic time)
3	D4	D3	Latch word	

*) If cycle duration measurement mode is enabled in the control byte, the cycle duration is given as a 24-bit value that is stored in D2 together with D3/D4.

Output Process Image				
Offset	Byte Destination		Description	
	High Byte	Low Byte		
0	-	C	not used	Control byte
1	D1	D0	Counter setting word	
2	-	-	not used	
3	-	-		

750-637

The above Incremental Encoder Interface Module has a total of 6 bytes of user data in both the Input and Output Process Image (4 bytes of encoder data and 2 bytes of control/status). The following table illustrates the Input and Output Process Image, which have 4 words mapped into each image. Word alignment is applied.

Table 166: Incremental Encoder Interface Modules 750-637

Input and Output Process Image			
Offset	Byte Destination		Description
	High Byte	Low Byte	
0	-	C0/S0	Control/Status byte of Channel 1
1	D1	D0	Data Value of Channel 1
2	-	C1/S1	Control/Status byte of Channel 2
3	D3	D2	Data Value of Channel 2

750-635,
753-635

The above Digital Pulse Interface module has a total of 4 bytes of user data in both the Input and Output Process Image (3 bytes of module data and 1 byte of control/status). The following table illustrates the Input and Output Process Image, which have 2 words mapped into each image. Word alignment is applied.

Table 167: Digital Pulse Interface Modules 750-635

Input and Output Process Image				
Offset	Byte Destination		Description	
	High Byte	Low Byte		
0	D0	C0/S0	Data byte	Control/status byte
1	D2	D1	Data bytes	

14.1.5.8 DC-Drive Controller

750-636

The DC-Drive Controller maps 6 bytes into both the input and output process image. The data sent and received are stored in up to 4 input and output bytes (D0 ... D3). Two control bytes (C0, C1) and two status bytes (S0/S1) are used to control the I/O module and the drive.

In addition to the position data in the input process image (D0 … D3), it is possible to display extended status information (S2 … S5). Then the three control bytes (C1 … C3) and status bytes (S1 … S3) are used to control the data flow.

Bit 3 of control byte C1 (C1.3) is used to switch between the process data and the extended status bytes in the input process image (Extended Info_ON). Bit 3 of status byte S1 (S1.3) is used to acknowledge the switching process.

Table 168: DC-Drive Controller 750-636

Input Process Image				
Offset	Byte Destination		Description	
	High Byte	Low Byte		
0	S1	S0	Status byte S1	Status byte S0
1	D1*) / S3**)	D0*) / S2**)	Actual position*) / Extended status byte S3**)	Actual position (LSB) / Extended status byte S2**)
2	D3*) / S5**)	D2*) / S4**)	Actual position (MSB) / Extended status byte S3**)	Actual position*) / Extended status byte S4**)

*) ExtendedInfo_ON = ‘0’.
**) ExtendedInfo_ON = ‘1’.

Output Process Image				
Offset	Byte Destination		Description	
	High Byte	Low Byte		
0	C1	C0	Control byte C1	Control byte C0
1	D1	D0	Setpoint position	Setpoint position (LSB)
2	D3	D2	Setpoint position (MSB)	Setpoint position

14.1.5.9 Stepper Controller

750-670

The Stepper controller RS422 / 24 V / 20 mA 750-670 provides the fieldbus coupler 12 bytes input and output process image via 1 logical channel. The data to be sent and received are stored in up to 7 output bytes (D0 … D6) and 7 input bytes (D0 … D6), depending on the operating mode.

Output byte D0 and input byte D0 are reserved and have no function assigned.

One I/O module control and status byte (C0, S0) and 3 application control and status bytes (C1 ... C3, S1 ... S3) provide the control of the data flow.

Switching between the two process images is conducted through bit 5 in the control byte (C0 (C0.5). Activation of the mailbox is acknowledged by bit 5 of the status byte S0 (S0.5).

Table 169: Stepper Controller RS 422 / 24 V / 20 mA 750-670

Input Process Image				
Offset	Byte Destination		Description	
	High Byte	Low Byte		
0	reserved	S0	reserved	Status byte S0
1	D1	D0	Process data*) / Mailbox**)	
2	D3	D2		
3	D5	D4		
4	S3	D6	Status byte S3	Process data*) / reserved**)
5	S1	S2	Status byte S1	Status byte S2

*) Cyclic process image (Mailbox disabled)
**) Mailbox process image (Mailbox activated)

Output Process Image				
Offset	Byte Destination		Description	
	High Byte	Low Byte		
0	reserved	C0	reserved	Control byte C0
1	D1	D0	Process data*) / Mailbox**)	
2	D3	D2		
3	D5	D4		
4	C3	D6	Control byte C3	Process data*) / reserved**)
5	C1	C2	Control byte C1	Control byte C2

*) Cyclic process image (Mailbox disabled)
**) Mailbox process image (Mailbox activated)

14.1.5.10 RTC Module

750-640

The RTC Module has a total of 6 bytes of user data in both the Input and Output Process Image (4 bytes of module data and 1 byte of control/status and 1 byte ID for command). The following table illustrates the Input and Output Process Image, which have 3 words mapped into each image. Word alignment is applied.

Table 170: RTC Module 750-640

Input and Output Process Image				
Offset	Byte Destination		Description	
	High Byte	Low Byte		
0	ID	C/S	Command byte	Control/status byte
1	D1	D0	Data bytes	
2	D3	D2		

14.1.5.11 DALI/DSI Master Module

750-641

The DALI/DSI Master module has a total of 6 bytes of user data in both the Input and Output Process Image (5 bytes of module data and 1 byte of control/status). The following tables illustrate the Input and Output Process Image, which have 3 words mapped into each image. Word alignment is applied.

Table 171: DALI/DSI Master Module 750-641

Input Process Image				
Offset	Byte Destination		Description	
	High Byte	Low Byte		
0	D0	S	DALI Response	Status byte
1	D2	D1	Message 3	DALI Address
2	D4	D3	Message 1	Message 2

Output Process Image				
Offset	Byte Destination		Description	
	High Byte	Low Byte		
0	D0	C	DALI command, DSI dimming value	Control byte
1	D2	D1	Parameter 2	DALI Address
2	D4	D3	Command extension	Parameter 1

14.1.5.12 DALI Multi-Master Module

753-647

The DALI Multi-Master module occupies a total of 24 bytes in the input and output range of the process image.

The DALI Multi-Master module can be operated in "Easy" mode (default) and "Full" mode. "Easy" mode is used to transmit simply binary signals for lighting control. Configuration or programming via DALI master module is unnecessary in "Easy" mode.

Changes to individual bits of the process image are converted directly into DALI commands for a pre-configured DALI network. 22 bytes of the 24-byte process image can be used directly for switching of electronic ballasts (ECG), groups or scenes in "Easy" mode. Switching commands are transmitted via DALI and group addresses, where each DALI and each group address is represented by a 2-bit pair.

The structure of the process data is described in detail in the following tables.

Table 172: Overview of Input Process Image in the "Easy" Mode

Input process image				
Offset	**Byte designation**		**Note**	
	High byte	**Low byte**		
0	-	S	res.	Status, activate broadcast Bit 0: 1-/2-button mode Bit 2: Broadcast status ON/OFF Bit 1,3-7: -
1	DA4…DA7	DA0…DA3	Bitpaar für DALI-Adresse DA0: Bit 1: Bit set = ON Bit not set = OFF Bit 2: Bit set = Error Bit not set = No error Bit pairs DA1 … DA63 similar to DA0.	
2	DA12…DA15	DA8…DA11		
3	DA20…DA23	DA16…DA19		
4	DA28…DA31	DA24…DA27		
5	DA36…DA39	DA32…DA35		
6	DA44…DA47	DA40…DA43		
7	DA52…DA55	DA48…DA51		
8	DA60…DA63	DA56…DA59		
9	GA4…GA7	GA0…GA3	Bit pair for DALI group address GA0: Bit 1: Bit set = ON Bit not set = OFF Bit 2: Bit set = Error Bit not set = No error Bit pairs GA1 … GA15 similar to GA0.	
10	GA12…GA15	GA8…GA11		
11	-	-	Not in use	

DA = DALI address
GA = Group address

Table 173: Overview of the Output Process Image in the "Easy" Mode“

Output process image				
Offset	**Byte designation**		**Note**	
	High byte	**Low byte**		
0	-	S	res.	Broadcast ON/OFF and activate: Bit 0: Broadcast ON Bit 1: Broadcast OFF Bit 2: Broadcast ON/OFF/dimming Bit 3: Broadcast short ON/OFF Bits 4 … 7: reserved
1	DA4…DA7	DA0…DA3	Bit pair for DALI address DA0: Bit 1: short: DA switch ON long: dimming, brighter Bit 2: short: DA switch OFF long: dimming, darker Bit pairs DA1 … DA63 similar to DA0.	
2	DA12…DA15	DA8…DA11		
3	DA20…DA23	DA16…DA19		
4	DA28…DA31	DA24…DA27		
5	DA36…DA39	DA32…DA35		
6	DA44…DA47	DA40…DA43		
7	DA52…DA55	DA48…DA51		
8	DA60…DA63	DA56…DA59		
9	GA4…GA7	GA0…GA3	Bitpaar für DALI-Gruppenadresse GA0: Bit 1: short: GA switch ON long: dimming, brighter Bit 2: short: GA switch OFF long: dimming, darker Bit pairs GA1 … GA15 similar to GA0.	
10	GA12…GA15	GA8…GA11		
11	Bit 8…15	Bit 0…7	Switch scene 0…15	

DA = DALI address
GA = Group address

14.1.5.13 LON® FTT Module

753-648

The process image of the LON® FTT module consists of a control/status byte and 23 bytes of bidirectional communication data that is processed by the WAGO-I/O-*PRO* function block "LON_01.lib". This function block is essential for the function of the LON® FTT module and provides a user interface on the control side.

14.1.5.14 EnOcean Radio Receiver

750-642

The EnOcean radio receiver has a total of 4 bytes of user data in both the Input and Output Process Image (3 bytes of module data and 1 byte of control/status). The following tables illustrate the Input and Output Process Image, which have 2 words mapped into each image. Word alignment is applied.

Table 174: EnOcean Radio Receiver 750-642

Input Process Image				
Offset	**Byte Destination**		**Description**	
	High Byte	**Low Byte**		
0	D0	S	Data byte	Status byte
1	D2	D1	Data bytes	

Output Process Image				
Offset	**Byte Destination**		**Description**	
	High Byte	**Low Byte**		
0	-	C	not used	Control byte
1	-	-	not used	

14.1.5.15 MP Bus Master Module

750-643

The MP Bus Master Module has a total of 8 bytes of user data in both the Input and Output Process Image (6 bytes of module data and 2 bytes of control/status). The following table illustrates the Input and Output Process Image, which have 4 words mapped into each image. Word alignment is applied.

Table 175: MP Bus Master Module 750-643

Input and Output Process Image				
Offset	Byte Destination		Description	
	High Byte	Low Byte		
0	C1/S1	C0/S0	extended Control/ Status byte	Control/status byte
1	D1	D0	Data bytes	
2	D3	D2		
3	D5	D4		

14.1.5.16 *Bluetooth*® RF-Transceiver

750-644

The size of the process image for the *Bluetooth*® module can be adjusted to 12, 24 or 48 bytes.
It consists of a control byte (input) or status byte (output); an empty byte; an overlay able mailbox with a size of 6, 12 or 18 bytes (mode 2); and the *Bluetooth*® process data with a size of 4 to 46 bytes.
Thus, each *Bluetooth*® module uses between 12 and 48 bytes in the process image. The sizes of the input and output process images are always the same.

The first byte contains the control/status byte; the second contains an empty byte.

Process data attach to this directly when the mailbox is hidden. When the mailbox is visible, the first 6, 12 or 18 bytes of process data are overlaid by the mailbox data, depending on their size. Bytes in the area behind the optionally visible mailbox contain basic process data. The internal structure of the *Bluetooth*® process data can be found in the documentation for the *Bluetooth*® 750-644 RF Transceiver.

The mailbox and the process image sizes are set with the startup tool WAGO-I/O-*CHECK*.

Table 176: Bluetooth® RF-Transceiver 750-644

Input and Output Process Image				
Offset	Byte Destination		Description	
	High Byte	Low Byte		
0	-	C0/S0	not used	Control/status byte
1	D1	D0	Mailbox (0, 3, 6 or 9 words) and Process data (2-23 words)	
2	D3	D2		
3	D5	D4		
...	...	...		
max. 23	D45	D44		

14.1.5.17 Vibration Velocity/Bearing Condition Monitoring VIB I/O

750-645

The Vibration Velocity/Bearing Condition Monitoring VIB I/O has a total of 12 bytes of user data in both the Input and Output Process Image (8 bytes of module data and 4 bytes of control/status). The following table illustrates the Input and Output Process Image, which have 8 words mapped into each image.
Word alignment is applied.

Table 177: Vibration Velocity/Bearing Condition Monitoring VIB I/O 750-645

Input and Output Process Image				
Offset	Byte Destination		Description	
	High Byte	Low Byte		
0	-	C0/S0	not used	Control/status byte (log. Channel 1, Sensor input 1)
1	D1	D0	Data bytes (log. Channel 1, Sensor input 1)	
2	-	C1/S1	not used	Control/status byte (log. Channel 2, Sensor input 2)
3	D3	D2	Data bytes (log. Channel 2, Sensor input 2)	
4	-	C2/S2	not used	Control/status byte (log. Channel 3, Sensor input 1)
5	D5	D4	Data bytes (log. Channel 3, Sensor input 3)	
6	-	C3/S3	not used	Control/status byte (log. Channel 4, Sensor input 2)
7	D7	D6	Data bytes (log. Channel 4, Sensor input 2)	

14.1.5.18 KNX/EIB/TP1 Module

753-646

The KNX/TP1 module appears in router and device mode with a total of 24-byte user data within the input and output area of the process image, 20 data bytes and 2 control/status bytes. Even though the additional bytes S1 or C1 are transferred as data bytes, they are used as extended status and control bytes. The opcode is used for the read/write command of data and the triggering of specific functions of the KNX/EIB/TP1 module. Word-alignment is used to assign 12 words in the process image. Access to the process image is not possible in router mode. Telegrams can only be tunneled.

In device mode, access to the KNX data can only be performed via special function blocks of the IEC application. Configuration using the ETS engineering tool software is required for KNX.

Table 178: KNX/EIB/TP1 Module 753-646

Input Process Image				
Offset	**Byte Destination**		**Description**	
	High Byte	**Low Byte**		
0	-	S0	not used	Status byte
1	S1	OP	extended Status byte	Opcode
2	D1	D0	Data byte 1	Data byte 0
3	D3	D2	Data byte 3	Data byte 2
4	D5	D4	Data byte 5	Data byte 4
5	D7	D6	Data byte 7	Data byte 6
6	D9	D8	Data byte 9	Data byte 8
7	D11	D10	Data byte 11	Data byte 10
8	D13	D12	Data byte 13	Data byte 12
9	D15	D14	Data byte 15	Data byte 14
10	D17	D16	Data byte 17	Data byte 16
11	D19	D18	Data byte 19	Data byte 18

Output Process Image				
Offset	**Byte Destination**		**Description**	
	High Byte	**Low Byte**		
0	-	C0	not used	Control byte
1	C1	OP	extended Control byte	Opcode
2	D1	D0	Data byte 1	Data byte 0
3	D3	D2	Data byte 3	Data byte 2
4	D5	D4	Data byte 5	Data byte 4
5	D7	D6	Data byte 7	Data byte 6
6	D9	D8	Data byte 9	Data byte 8
7	D11	D10	Data byte 11	Data byte 10
8	D13	D12	Data byte 13	Data byte 12
9	D15	D14	Data byte 15	Data byte 14
10	D17	D16	Data byte 17	Data byte 16
11	D19	D18	Data byte 19	Data byte 18

14.1.5.19 AS-interface Master Module

750-655

The length of the process image of the AS-interface master module can be set to fixed sizes of 12, 20, 24, 32, 40 or 48 bytes.
It consists of a control or status byte, a mailbox with a size of 0, 6, 10, 12 or 18 bytes and the AS-interface process data, which can range from 0 to 32 bytes.

The AS-interface master module has a total of 6 to maximally 24 words data in both the Input and Output Process Image. Word alignment is applied.

The first Input and output word, which is assigned to an AS-interface master module, contains the status / control byte and one empty byte.
Subsequently the mailbox data are mapped, when the mailbox is permanently superimposed (Mode 1).

In the operating mode with suppressible mailbox (Mode 2), the mailbox and the cyclical process data are mapped next.
The following words contain the remaining process dat.

The mailbox and the process image sizes are set with the startup tool WAGO-I/O-*CHECK*.

Table 179: AS-interface Master Module 750-655

Input and Output Process Image				
Offset	**Byte Destination**		**Description**	
	High Byte	**Low Byte**		
0	-	C0/S0	not used	Control/status byte
1	D1	D0	Mailbox (0, 3, 5, 6 or 9 words)/ Process data (0-16 words)	
2	D3	D2		
3	D5	D4		
...	...	...		
max. 23	D45	D44		

14.1.6 System Modules

14.1.6.1 System Modules with Diagnostics

750-610, -611

The modules provide 2 bits of diagnostics in the Input Process Image for monitoring of the internal power supply.

Table 180: System Modules with Diagnostics 750-610, -611

Input Process Image							
Bit 7	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**
						Diagnostic bit S 2 Fuse	Diagnostic bit S 1 Fuse

14.1.6.2 Binary Space Module

750-622

The Binary Space Modules behave alternatively like 2 channel digital input modules or output modules and seize depending upon the selected settings 1, 2, 3 or 4 bits per channel. According to this, 2, 4, 6 or 8 bits are occupied then either in the process input or the process output image.

Table 181: Binary Space Module 750-622 (with Behavior Like 2 Channel Digital Input)

Input and Output Process Image							
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
(Data bit DI 8)	(Data bit DI 7)	(Data bit DI 6)	(Data bit DI 5)	(Data bit DI 4)	(Data bit DI 3)	Data bit DI 2	Data bit DI 1

14.2 CODESYS Libraries

Additional functions for the controller 750-8203are provided using libraries.

14.2.1 General Libraries

This section contains general CODESYS libraries supported by the controller750-8203.

14.2.1.1 CODESYS System Libraries

All of the functions of the CODESYS system libraries listed below are supported.

Table 182: CODESYS System Libraries

Library	Function	C/IEC 61131
Analyzation.lib	Analysis of boolean expressions	C and IEC 61131
AnalyzationNew.lib	Analysis of boolean expressions	C and IEC 61131
Iecsfc.lib	Provision of implicit variables in the SFC (sequential function chart)	IEC 61131
NetVarUdp_LIB_V23.lib	Implementation for network variables	IEC 61131
Standard.LIB	Offers various standard functions	C
SysLibAlarmTrend.lib	Supports alarm and trend tasks	IEC 61131
SysLibCallback.lib	For installing call-back handlers and event handlers	C
SysLibDir.lib	For accessing directories	C
SysLibDirect.lib	Access to variables using indices	C
SysLibEvent.lib	Handling of events in the system	C
SysLibFileStream.lib	File handling using ANSI-C functions	C
SysLibGetAddress.lib	Returns addresses and the size of memory segments	C
SysLibIecTasks.lib	Administration of IEC tasks	C
SysLibMem.lib	Memory administration	C
SysLibPlcCtrl.lib	Control of the PLC from outside the PLC program	C
SysLibProjectInfo.lib	Reading out of information about the CODESYS project	C
SysLibSem.lib	Handling of semaphores	C
SysLibSockets.lib	Socket handling	C
SysLibSocketsAsync.lib	Socket handling, asynchronous	C
SysLibStr.lib	String functions	C
SysLibTasks.lib	Administration of tasks	C
SysLibTime.lib	Administration of real-time clock	C
SysLibVisu.lib	Dynamic visualization	C

Table 182: CODESYS System Libraries

Library	Function	C/IEC 61131
SysTaskInfo.lib	Evaluation of task information in the Online mode	IEC 61131
Util.lib	Various logical operations	IEC 61131
Util_no_Real.lib	Various logical operations	IEC 61131

Additional information about the libraries is given in the online Help function for CODESYS-IDE.

14.2.1.2 SysLibFile.lib

The controller 750-8203supports the following function blocks of the “SysLibCom.lib” library:

- SysFileClose
- SysFileCopy
- SysFileDelete
- SysFileEOF
- SysFileGetPos
- SysFileGetSize
- SysFileGetTime
- SysFileOpen
- SysFileRead
- SysFileRename
- SysFileSetPos
- SysFileWrite

Note

Ensure that files are saved!

Files are not reliably saved on the data medium until you call up the “SysFileClose” function block!

Additional information about this is given in the online Help function for CODESYS-IDE.

14.2.1.3 SysLibFileAsync.lib

The controller 750-8203supports the following function blocks of the “SysLibCom.lib” library:

- SysFileCloseAsync
- SysFileCopyAsync
- SysFileDeleteAsync
- SysFileEOFAsync
- SysFileGetPosAsync
- SysFileGetSizeAsync
- SysFileGetTimeAsync
- SysFileOpenAsync

- SysFileReadAsync
- SysFileRenameAsync
- SysFileSetPosAsync
- SysFileWriteAsync

Note

Ensure that files are saved!

Files are not reliably saved to the data medium until you call up the "SysFileCloseAsync" function block.

Additional information about this is given in the online Help function for CODESYS-IDE.

14.2.1.4 SysLibRtc.lib

The controller 750-8203supports the following function blocks of the "SysLibRtc.lib" library:

- SysRtcGetHourMode
- SysRtcGetTime
- SysRtcSetTime

Additional information about this is given in the online Help function for CODESYS-IDE.

14.2.1.5 BusDiag.lib

The controller 750-8203 supports the following function blocks of the "BusDiag.lib" library:

- DiagGetBusState
- DiagGetState

The document containing a description of this library and the function blocks it includes is available for download on the Internet at www.wago.com.

The values for the input variables "DEVICE NUMBER" of the "DiagGetBusState" and "DiagGetState" functions are based on the particular device and bus system and are as follows for the controller 750-8203:

Table 183: Input Variable "DEVICE NUMBER"

Bus System	Value
Internal data bus	0
MODBUS	1
CANopen	2

14.2.1.6 mod_com.lib

The controller 750-8203supports the following function blocks of the "mod_com.lib" library:

- ADD_PI_INFORMATION
- CRC16
- FBUS_ERROR_INFORMATION
- GET_DIGITAL_INPUT_OFFSET
- GET_DIGITAL_OUTPUT_OFFSET
- KBUS_ERROR_INFORMATION
- MOD_COM_VERSION
- PI_INFORMATION
- SET_DIGITAL_INPUT_OFFSET
- SET_DIGITAL_OUTPUT_OFFSET
- SLAVE_ADDRESS

The document containing a description of this library and the function blocks it includes is available for download on the Internet at www.wago.com.

14.2.1.7 SerComm.lib

The controller 750-8203supports the following function blocks of the "SerComm.lib" library:

- SERCOMM
- SERCOMM_VERSION

The document containing a description of this library and the function blocks it includes is available for download on the Internet at www.wago.com.

14.2.1.8 WagoConfigToolLIB.lib

The following table shows call-ups that allow you to configure and parameterize the controller from the PLC program or Linux® via the "ConfigToolFB" function block (see parameter "stCallString"). In addition to WBM and the CBM, this is another variant to configure the controller for operational requirements.

The configuration directory for this under Linux® is: **`/etc/config-tools/`**

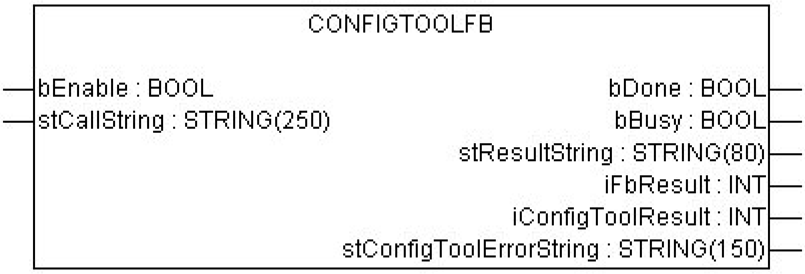

CONFIGTOOLFB
bEnable : BOOL
stCallString : STRING(250)
bDone : BOOL
bBusy : BOOL
stResultString : STRING(80)
iFbResult : INT
iConfigToolResult : INT
stConfigToolErrorString : STRING(150)

Figure 124: Graphical Representation of the "ConfigToolFB" Function Block

Table 184: Description of the Configuration Scripts for "Information"

Parameters	Status	Call-Up	Output/Input	Effective
Controller Details: Identifies various information about the controller				
Product Description	read	get_coupler_details product-description	Product description	Immediately
Order Number	read	get_coupler_details order-number	Item number of the controller	Immediately
Firmware Revision	read	get_coupler_details firmware-revision	Firmware version of the controller	Immediately
Licence Information	read	get_coupler_details license-information	CODESYS license details	Immediately
Network Details X1: Identifies the parameters currently used for the ETHERNET interface X1/X2 in "switched" mode or for the ETHERNET interface X1 in "separated" mode				
State	read	get_actual_eth_config X1 state	Status of the interface. Possible return values: - enabled - disabled	Immediately
Mac Address	read	get_actual_eth_config X1 mac-address	Display of the MAC address	Immediately
IP Address	read	get_actual_eth_config X1 ip-address	Display of current IP address	Immediately
Subnet Mask	read	get_actual_eth_config X1 subnet-mask	Display of the current subnet mask	Immediately
Network Details X2: Identifies the parameters currently used for the ETHERNET interface X2 in "separated" mode				
See "Network Details X1". When calling these up, replace "X1" with "X2" (in "separated" mode only).				

Table 185: Description of the Configuration Scripts for “CODESYS”

Parameters	Status	Call-Up	Output/Input	Effective
Information				
CODESYS Web-Server Version	read	get_coupler_details codesys-Web-Server-version	CODESYS Web server version	Immediately
Project Details				
Date	read	get_rts_info project date	Display of the project information specified in CODESYS (Menu > Project > Project Information)	Immediately
Title	read	get_rts_info project title		Immediately
Version	read	get_rts_info project version		Immediately
Author	read	get_rts_info project author		Immediately
Description	read	get_rts_info project description		Immediately
CODESYS State				
State	read	get_rts_info state	Display of the CODESYS status (RUN or STOP)	Immediately
Boot Project Location				
Boot Project Location	read	get_rts3scfg_value PLC Files	Reading of the storage location of the boot project Possible return values: - HOME:// - CARD://	Immediately
	write	change_rts_config area=PLC Files=<value>	Change the storage location for the boot project. Possible entries for <Value>: - HOME:// - CARD://	

Table 186: Description of the Configuration Scripts for “Networking - Host-/Domain-Name”

Parameters	Status	Call-Up	Output/Input	Effective
Host Name				
Host Name	read	get_coupler_details hostname	Display of the host name. The return value is blank when /etc/hostname is empty. For details see the parameter “Actual Hostname”.	Immediately
	write	change_hostname hostname=<String>	Changing the host name. Input a host name for <String>.	After restart
Actual Hostname	read	get_coupler_details actual-hostname	The actual host name (if /etc/hostname is empty, an unique host name is generated from the MAC address)	Immediately
Domain Name				
Domain Name	read	get_coupler_details domain-name	Display of domain name	Immediately
	write	edit_dns_server domain-name=<String>	Change the domain name. Enter the domain name for <String>.	

Table 187: Description of the Configuration Scripts for “Networking - TCP/IP”

Parameters	Status	Call-Up	Output/Input	Effective
IP Address X1: Identifies the parameters currently used for the ETHERNET interface X1/X2 in “switched” mode or for the ETHERNET interface X1 in “separated” mode				
Type of IP address configuration	read	get_eth_config X1 config-type	Method by which the interface receives its IP address: Possible return values: - static (set statically) - dhcp (per DHC) - bootp (per BootP)	Immediately

	write	config_interfaces
		interface=X1
		config-type=<value>
		state=enabled
			Activate the method by which the interface receives its IP address.
			Possible entries for <Value>:
			- static (set statically)
			- dhcp (per DHC)
			- bootp (per BootP)	

IP address	read	get_eth_config X1 ip-address	Address set for using a static IP address (static IP).	Immediately

	write	config_interfaces
		interface=X1
		ip-address=<value>
			Change IP address for static IP. <Value> must have an IP address with the format “Number.Number.Number. Number”.	

Subnet Mask	read	get_eth_config X1 subnet-mask	Subnet mask set for using a static IP address (Static IP).	Immediately

	write	config_interfaces
		interface=X1
		subnet-mask=<value>
			Change subnet mask for static IP. <Value> must have an IP address with the format “Number.Number.Number. Number”.	

IP Address X2: Identifies the parameters currently used for the ETHERNET interface X2 in “separated” mode				
See “IP Address X1”. When calling these up, replace “X1” with “X2” (in “separated” mode only).				

Table 187: Description of the Configuration Scripts for “Networking - TCP/IP”

Parameters	Status	Call-Up	Output/Input	Effective
Default Gateway				
Default Gateway	read	get_coupler_details default-gateway	Display of the standard gateway address.	Immediately
	read	get_eth_config X1 default-gateway	Display of the standard gateway address. Use of “X2” leads to the same result because the value is always written for both interfaces at the same time.	Immediately
	write	config_default_gateway interface=<value> default-gateway-value=<value>	Select the interface you want to use as the standard gateway. Possible entries for interface: - X1 - X2 - none (no standard gateway selected) Default-gateway-value is an IP address with the format “Number.Number.Number. Number”.	Immediately
DNS server				
DNS Server 1	read	get_dns_server 1	DNS server address with the consecutive number 1.	Immediately
	write/ change	edit_dns_server dns-server-nr=1 change=change dns-server-name=<value>	Set the address of the DNS server with 1 as the consecutive number. <Value> is an IP address with the format “Number.Number.Number. Number”.	
	write/ delete	edit_dns_server dns-server-nr=1 delete=delete	Delete the DNS server with the consecutive number 1.	
DNS Server 2 … n	See “DNS Server 1”. Change the server number each time you call this up (ascending order).			Immediately
Add DNS server	write	edit_dns_server add=add dns-server-name=<value>	Add additional DNS addresses here. <Value> is an IP address with the format “Number.Number.Number. Number”.	Immediately

Table 188: Description of the Configuration Scripts for “Networking - ETHERNET”

Parameters	Status	Call-Up	Output/Input	Effective
Switch Configuration				
Interface Mode	read	get_dsa_mode	Query the switch configuration: Possible return values: - 0 = „switched“ mode - 1 = „separated“ mode	Immediately
	write	set_dsa_mode -v <value>	Set the switch configuration: Possible entries for <value>: - 0 = „switched“ mode - 1 = „separated“ mode	
Interface X1				
Port State	read	get_eth_config X1 state	Query the port state: Possible return values: - enabled - disabled	Immediately
	write	config_ethernet port=X1 state=enabled	Activate port: enabled	
		config_ethernet port=X1 state=disabled	Deactivate port: disabled	
Autonegotiation	read	get_eth_config X1 autoneg	Query the status of the autonegotiation function: Possible return values: - on - off	Immediately
	write	config_ethernet port=X1 autoneg=on	Activate the autonegotiation function: on	
		config_ethernet port=X1 autoneg=off speed=<value> duplex=<value>	Deactivate the autonegotiation function: off Note: You must also indicate the speed and duplex value when you deactivate the autonegotiation function. Possible entries for speed: - 10M - 100M Possible entries for duplex: - half - full	
Speed and Duplex Settings	read	get_eth_config X1 speed	Display of ETHERNET speed	Immediately
	read	get_eth_config X1 duplex	Display of the Duplex mode	
	write	config_ethernet port=X1 autoneg=off speed=<value> duplex=<value>	Change the ETHERNET speed and the Duplex mode. Possible entries for speed: - 10M - 100M Possible entries for duplex: - half - full	
Interface X2				
See “Interface X1”. When calling these up, replace “X1” with “X2”.				

Table 189: Description of the Configuration Scripts for “CODESYS”

Parameters	Status	Call-Up	Output/Input	Effective
Transmission Mode X1				
Autonegotiation	read	get_eth_config X1 autoneg	Query the status of the autonegotiation function. Possible return values: - on - off	Immediately
	write	config_ethernet port=X1 autoneg=on	Activate the autonegotiation function: on	
		config_ethernet port=X1 autoneg=off speed-duplex=<value>	Deactivate the autonegotiation function: off Note: You must also indicate the speed and duplex value when you deactivate the autonegotiation function. Possible entries for <Value>: - 10-half - 10-full - 100-half - 100-full	
Speed and Duplex Settings	read	get_eth_config X1 speed	Display of ETHERNET speed	Immediately
	read	get_eth_config X1 duplex	Display of Duplex mode	
	write	config_ethernet port=X1 autoneg=off speeed-duplex=<value>	Change the ETHERNET speed and the Duplex mode. Possible entries for <Value>: - 10-half - 10-full - 100-half - 100-full	
Transmission Mode X2				
See “Transmission Mode X1”. When calling, replace “X1” with “X2”.				

Table 190: Description of the Configuration Scripts for “NTP”

Parameters	Status	Call-Up	Output/Input	Effective
Configuration Data				
State	read	get_ntp_config state	Query the status of the NTP server. Possible return values: - enabled - disabled	Immediately
	write	config_sntp state=<value>	Possible entries for <Value>: - enabled - disabled	
Port	read	get_ntp_config port	Port number of the NTP server	Immediately
	write	config_sntp port=<value>	Enter the port number for <Value>.	
Time Server	read	get_ntp_config time-server	Query the IP address of the Time server.	Immediately
	write	config_sntp time-server=<value>	Enter the IP address of the Time server. The <value> can contain an IP address in the format “Number.Number.Number. Number” or a domain name as a string.	
Update Time (seconds)	read	get_ntp_config update-time	Query the polling cycle of the time server.	Immediately
	write	config_sntp update-time=<value>	Specify the time-server's query cycle (in s) for <value>.	

Table 191: Description of the Configuration Scripts for “Clock”

Parameters	Status	Call-Up	Output/Input	Effective
Clock				
Time and Date				
Date on device, local	read	get_clock_data date-local	Local time and date	Immediately

	write	config_clock
		type=local
		date=<Datum>
			Change date. The format for <date> is: DD.MM.YYYY	

Time on device, UTC	read	get_clock_data time-utc	Time/UTC	Immediately
	write	config_clock type=utc time=<Time>	Change time, based on UTC time. The format for <time> is: hh:mm:ss xx	
Time on device, local	read	get_clock_data time-local	Time/local time	Immediately

	write	config_clock
		type=local
		time=<Time>
			Change time, based on local time. The format for <time> is: hh:mm:ss xx	

12-Hour-Format	read	get_clock_data display-mode	Presentation format either as 12 or 24-hour format: Possible return values: - 12-hour-format - 24-hour-format	Immediately
	write	config_clock _ display_mode display-mode=<value>	Set the presentation format for the time. Possible entries for <Value>: - 12-hour-format - 24-hour-format	
Time Zone				
TZ-String	read	get_clock_data tz-string	Currently set time zone – original TZ string as stored in the operating system.	Immediately
	write	config_timezone tz-string=<String>	Change TZ string directly. Example of <String>: CET-1CEST, M3.5.0/2,M10.5.0/3	

Table 192: Description of the Configuration Scripts for “Administration”

Parameters	Status	Call-Up	Output/Input	Effective
Administration				
Reboot Controller				
-	write	start_reboot	Restart the controller.	immediately

Table 193: Description of Configuration Scripts for “Package Server”

Parameters	Status	Call-Up	Output/Input	Valid
Firmware Update				
Medium for active partition	read	get_filesystem_data active-partition-medium	Specifies the medium for the active partition (memory card, internal flash).	Right away
Create firmware backup	write	firmware_backup package-settings=<Value1> package-codesys=<Value2> package-system=<Value3> device-medium=<Value4> auto-update=<Value5>	Generates a backup of the selected packet on the specified medium. Parameter: <Value1> = 1, if “Settings” packet is to be selected. <Value2> = 1, if the “CODESYS Project” packet is to be selected. <Value3> = 1, if the “System” packet is to be selected. <Value4> = Target medium for saving the backup. (memory card, internal flash) <Value5> = 1, if Auto Update is to be activated. Parameters, which are not to be set (1) can either be set to 0 or omitted completely.	Right away

Table 194: Description of Configuration Scripts for “Ports and Services”

Parameters	Status	Call-Up	Output/Input	Valid
Port				
Telnet				
Telnet Port	read	get_port_state telnet	Read the status of the Telnet server. Possible return values: - enabled - disabled	Right away
	write	config_port port=telnet state=<Value>	Possible entries for <Value>: - enabled - disabled	
FTP				
FTP Port	read	config_ssl ftp-status	Read the status of the FTP server. Possible return values: - enabled - disabled	Right away
	write	config_port port=ftp state=<Value>	Possible entries for <Value>: - enabled - disabled	
FTPS				
FTPS Port	read	Config_ssl ftps-status	Read the status of the FTPS port. Possible return values: - enabled - disabled	Right away
	write	config_port port=ftps state=<Value>	Activate/Deactivate FTPS. Possible entries for <Value>: - enabled - disabled	

Table 194: Description of Configuration Scripts for “Ports and Services”

Parameters	Status	Call-Up	Output/Input	Valid
HTTP				
HTTP Port	read	Config_ssl http-status	Read the status of the HTTP port. Possible return values: - enabled - disabled	Right away
	write	config_port port=http state=<Value>	Activate/Deactivate HTTP. Possible entries for <Value>: - enabled - disabled	
HTTPS				
HTTPS Port	read	Config_ssl https-status	Read the status of the HTTPS port. Possible return values: - enabled - disabled	Right away
	write	config_port port=https state=<Value>	Activate/Deactivate HTTPS. Possible entries for <Value>: - enabled - disabled	

Table 194: Description of Configuration Scripts for "Ports and Services"

Parameters	Status	Call-Up	Output/Input	Valid
SSH				
SSH	read	get_ssh_config state	Read the status of the SSH port. Possible return values: - enabled - disabled	Right away
	read	get_ssh_config root-access-state	Indicates whether logon as root is permitted. Possible return values: - enabled - disabled	
	read	get_ssh_config password-request-state	Indicates whether authentication by password (instead of PKI key files) is permitted. Possible return values: - enabled - disabled	
	read	get_ssh_config port-number	Specifies the SSH port	
	write	config_ssh state=<Value>	Activate/Deactivate SSH service. Possible entries for <Value>: - enabled - disabled	
	write	config_ssh port-number=<Value>	Set the SSH port	
	write	config_ssh root-access-state-value=<Value>	Permit/Prohibit logon as root. Possible entries for <Value>: - enabled - disabled	
	write	config_ssh password-request-state-value=<Value>	Permit/Prohibit authentication by password. Possible entries for <Value>: - enabled - disabled	
TFTP				
TFTP	read	get_tftp_config state	Read the status of the TFTP port. Possible return values: - enabled - disabled	Right away
	read	get_tftp_config download-dir	Read the TFTP main directory.	
	write	config_tftp state=<Value>	Activate/Deactivate TFTP port. Possible entries for <Value>: - enabled - disabled	
	write	config_tftp download-dir=<Value>	Set the TFTP main directory.	

Table 194: Description of Configuration Scripts for “Ports and Services”

Parameters	Status	Call-Up	Output/Input	Valid
CODESYS				
CODESYS Webserver Port	read	get_port_state codesys-webserver	Read status of the CODESYS Web server. Possible return values: - enabled - disabled	Right away
	write	config_port port=codesys-webserver state=<Value>	Enable/disable the CODESYS Web server. Possible entries for <Value>: - enabled - disabled	
CODESYS Port	read	get_rts3scfg_value PLC DisableTcpIp Programming	Query the status of the value for “DisableTcpIpProgramming” in the CODESYS configuration. Possible return values: - YES: CODESYS port not being used. - NO: CODESYS port in use	Right away
	write	change_rts_config area=PLC disable-tcpip=<Value>	Possible entries for <Value>: - YES: CODESYS port not being used. - NO: CODESYS port in use.	
CODESYS Port Number	read	get_rts3scfg_value PLC TcpIpPort	Value set in the CODESYS configuration for the TCP/IP port.	Right away
	write	change_rts_config area=PLC TcpIpPort=<Value>	Change the CODESYS port number. Enter the TCP/IP port number for <Value>.	
CODESYS Authentication	read	get_rts3scfg_value PASSWORD USEPWD	Read the status of the CODESYS access password query. Possible return values: - 1 - 0	Right away
	write	change_rts_config area=PASSWORD USEPWD=<Value>	Activate/Deactivate CODESYS access password query. Possible entries for <Value>: - 1 - 0	
Change CODESYS Authentication Password	write	config_linux_user user=admin new-password=<Value> confirm-password=<Value>	Change the CODESYS access password	Right away

Table 195: Description of configuration scripts for “SNMP”

Parameters	Status	Call-Up	Output/Input	Valid
General SNMP information parameters				
Name of device	read	get_snmp_data device-name	Specifies the SNMP parameter “sysName”.	Right away
	write	config_snmp device-name=<Value>	Change the SNMP parameter “sysName” (<Value> = string). *	After restart
Description	read	get_snmp_data description	Specifies the SNMP parameter “sysDescr”.	Right away
	write	config_snmp description=<Value>	Change the SNMP parameter “sysDescr” (<Value> = string). *	After restart
Physical location	read	get_snmp_data physical-location	Specifies the SNMP “sysLocation” parameter.	Right away
	write	config_snmp physical-location=<Value>	Change the SNMP parameter “sysLocation” (<Value> = string). *	After restart
Contact	read	get_snmp_data contact	Specifies the SNMP “sysContact” parameter.	Right away
	write	config_snmp contact=<Value>	Change the SNMP parameter “sysContact” (<Value> = string).	After restart
* When entering values, the blank characters must be filled by either “+” or “%20”. If this is not done, the input is not recognized as a coherent string.				
SNMP Manager configuration for v1 and v2c				
Protocol status	read	get_snmp_data v1-v2c-state	Outputs the status of the SNMP protocol for v1/v2c as a string. Possible return values: - enabled - disabled	Right away
Local Community Name	read	get_snmp_data v1-v2c-community-name	Specifies the community name set for v1/v2c/	Right away
Protocol Status/Community Name	write	config_snmp v1-v2c-state=<Value1> v1-v2c-community-name=<Value2>	Activates/deactivates the v1/v2c protocol (<Value1> = enabled or disabled) and assigns a community name. (<Value2> = string without spaces, min. 1, max. 32 characters). Note: No community name is required for deactivation. Activation is only possible by entering a community name. A community name can only be saved when the protocol is activated.	After restart

Table 195: Description of configuration scripts for “SNMP”

Parameters	Status	Call-Up	Output/Input	Valid
SNMP Trap Receiver Configuration for v1 and v2c Any number of trap receivers can be configured. A trap receiver that has been set up is always active; the data set must be completely deleted to deactivate it.				
IP address of a trap receiver	read	get_snmp_data v1-v2c-trap-receiver-address <Nummer>	Specifies the IP address of the trap receiver that the controller is to send the v1 or v2 traps to. The <number> parameter enables consecutive reading of related data from the individually configured trap receiver for a short period of time (without interim changing of the data). This is a consecutive number that is not connected to the data. If the number is not included, the data of the first receiver are read.	Right away
Community Name	read	get_snmp_data v1-v2c-trap-receiver-community-name <Nummer>	Specifies the community name that the SNMP agent of the controller sends in the Trap Header. Parameter <number> see section “IP Address of a Trap Receiver”.	Right away
Trap version	read	get_snmp_data v1-v2c-trap-receiver-version <Nummer>	Specifies the SNMP version (“v1” or “v2c”) via which the SNMP agent sends the traps to the associated trap receiver address. Parameter <number> see section “IP Address of a Trap Receiver”.	Right away

Creating/ deleting a trap receiver	write	config_snmp
		v1-v2c-trap-receiver-edit=<Value1>
		v1-v2c-trap-receiver-address=<Value2>
		v1-v2c-trap-receiver-community-name=<Value3>
		v1-v2c-trap-receiver-version=<Value4>
			Create a new trap receiver (value1=add) or delete an already configured trap receiver (value1=delete).
			Other parameters:
			<Value2> = IP address (number.number.number.number) that the controller is to send the traps to.
			<Value3>: Community string (string), which the controller enters in the trap header.
			<Value4>: SNMP version, via which the traps are sent (v1 or v2c).
			Note:
			All parameters must also be entered when deleting a trap receiver, as this is the only means to uniquely identify the data set.
				After restart

Table 195: Description of configuration scripts for “SNMP”

Parameters	Status	Call-Up	Output/Input	Valid
Configuration of SNMP v3 Any number of SNMP v3 users can be created. A user that has been set up is always active; the complete data set must be deleted to deactivate a user.				
Authentication Name	read	get_snmp_data v3-auth-name <Nummer>	Specifies the user name for the v3 user. The <number> parameter enables consecutive reading of the related data from the individually configured trap receiver for a short period of time (without interim changing of the data). This is a consecutive number that is not connected to the data. If the number is not included, the data of the first user are read.	Right away
Authentication encryption type	read	get_snmp_data v3-auth-type <Number>	Specifies the type of encryption that the v3 user uses (none, MD5, or SHA). Parameter <number> see “Authentication Name”.	Right away
Authentication key	read	get_snmp_data v3-auth-key <Nummer>	Specifies the key string for authentication. Parameter <number> see “Authentication Name”.	Right away
Privacy encryption type	read	get_snmp_data v3-privacy <number>	Specifies the type of privacy encryption for the v3 user (none, DES, or AES). Parameter <number> see “Authentication Name”.	Right away
Privacy key	read	get_snmp_data v3-privacy-key <number>	Specifies the key string for privacy. If nothing is entered, the SNMP agent uses the “Authentication Key”. Parameter <number> see “Authentication Name”.	Right away
Trap receiver address	read	get_snmp_data v3-notification-receiver <number>	IP address of an SNMP manager that the agent traps for this v3 user are sent to. If nothing is entered here, no traps are sent for this user. Parameter <number> see “Authentication Name”.	Right away

Table 195: Description of configuration scripts for “SNMP”

Parameters	Status	Call-Up	Output/Input	Valid

Add new v3-User	write	config_snmp
		v3-edit=add
		v3-auth-name=<Value1>
		v3-auth-type=<Value2>
		v3-auth-key=<Value3>
		v3-privacy=<Value4>
		v3-privacy-key=<Value5>
		v3-notification-receiver=<Value6>
			Creating a new v3 user.
			v3-auth-name: User name, string without spaces, maximum of 32 characters. This must be a new, unique user name.
			
			Parameters:
			User name (<Value1> = string)
			Encryption method. (<Value2> = none, MD5 or SHA).
			Key string for authentication, (<Value3> = String with at least eight and a maximum of 32 characters)
			Privacy encryption method (<Value4> = none, DES or AES).
			Privacy key string (<Value5> = String with at least eight and a maximum of 32 characters), can also be blank; in this case the authentication key will be used.
			The IP address of a trap receiver is transmitted as the notification receiver (<Value6> = number.number.number.number) . This parameter is not required if no v3 traps are to be sent.
				After restart

Delete v3 user	write	config_snmp
		v3-edit=delete
		v3-auth-name=<Value>
			Deleting a v3 user that has been set up.
			Because the doubled allocation of the same user name is prevented when creating a user, the name is sufficient to uniquely identify a data set (<Value> = string).
				After restart

14.2.1.9 WagoLibCpuUsage.lib

The controller 750-8203supports the following function blocks of the “WagoLibCpuUsage.lib” library:

- CPU_Usage

The document containing the description of the library and the function block it includes is available for download on the Internet at www.wago.com.

14.2.1.10 WagoLibDiagnosticIDs.lib

The controller 750-8203supports the following function blocks of the “WagoLibDiagnosticIDs.lib” library:

- DIAGNOSTIC_SEND_ID
- DIAGNOSTIC_SET_TEXT_FOR_ID

The document containing the description of the library and the function block it includes is available for download on the Internet at www.wago.com.

14.2.1.11 WagoLibLed.lib

The controller 750-8203supports the following function blocks of the “WagoLibLed.lib” library:

- LED_SET_STATIC
- LED_SET_BLINK
- LED_SET_FLASH
- LED_SET_ERROR
- LED_RESET_ERROR
- LED_RESET_ALL_ERRORS
- LED_GET_STATE
- LED_GET_STATE_ASYNC

The document containing a description of this library and the function blocks it includes is available for download on the Internet at www.wago.com.

14.2.1.12 WagoLibNetSnmp.lib

The controller 750-8203supports the following function blocks of the “WagoLibNetSnmp.lib” library:

- snmpGetValueCustomOID_INT32
- snmpGetValueCustomOID_STRING
- snmpGetValueCustomOID_UINT32
- snmpRegisterCustomOID_INT32
- snmpRegisterCustomOID_STRING
- snmpRegisterCustomOID_UINT32
- snmpSetValueCustomOID_INT32
- snmpSetValueCustomOID_STRING
- snmpSetValueCustomOID_UINT32

The document containing the description of the library and the function block it includes is available for download on the Internet at www.wago.com.

14.2.1.13 WagoLibNetSnmpManager.lib

The controller 750-8203supports the following function blocks of the “WagoLibNetSnmpManager.lib” libraries:

- SNMPM_DINT_TO_TLV
- SNMPM_UDINT_TO_TLV
- SNMPM_STRING_TO_TLV
- SNMPM_TLV_TO_DINT
- SNMPM_TLV_TO_UDINT
- SNMPM_TLV_TO_STRING
- SNMPM_GET
- SNMPM_GET_V3

- SNMPM_SET
- SNMPM_SET_V3

The document containing a description of this library and the function blocks it includes is available for download on the Internet at www.wago.com.

14.2.1.14 WagoLibSSL.lib

The controller 750-8203supports the following function blocks of the "WagoLibSSL.lib" library:

- SSL_CTX
- SSL_CTX_load_verify_locations
- SSL_CTX_sess_set_cache_size
- SSL_CTX_set_client_CA_list
- SSL_CTX_set_method
- SSL_CTX_use_certificate_file
- SSL_CTX_use_PrivateKey_file
- SSL_free
- SSL_get_error
- SSL_Hndshk_Accept
- SSL_Hndshk_Connect
- SSL_load_client_CA_file
- SSL_read
- SSL_shutdown
- SSL_write

The document containing a description of this library and the function blocks it includes is available for download on the Internet at www.wago.com.

14.2.1.15 WagoLibTerminalDiag.lib

The controller 750-8203supports the following function blocks of the "WagoLibTerminalDiag.lib" library:

- GET_TERMINALDIAG

The document containing a description of this library and the function blocks it includes is available for download on the Internet at www.wago.com.

14.2.2 Libraries for a CANopen and CANLayer2 Link

This section contains libraries supported by the controller 750-8203for linking with CANopen and CANLayer2.

14.2.2.1 WagoCANLayer2_02.lib

The controller 750-8203supports the following function blocks of the “WagoCANLayer2_02.lib” library:

- CAN_ERROR_INFO
- CAN_LAYER2_VERSION
- CAN_RX_11BIT_FRAME
- CAN_RX_29BIT_FRAME
- CAN_TX_11BIT_FRAME
- CAN_TX_29BIT_FRAME
- CAN_OPEN
- CAN_CLOSE
- CAN_RESET
- CAN_SET_LED

The document containing a description of this library and the function blocks it includes is available for download on the Internet at www.wago.com.

14.2.2.2 WagoCANopen_02.lib

The controller 750-8203supports the following functions blocks of the “WagoCANopen_02.lib” library:

- CIA405_GET_KERNEL_STATE
- CIA405_GET_LOCAL_NODE_ID
- CIA405_RECV_EMCY
- CIA405_RECV_EMCY_DEV
- CIA405_GET_STATE
- CIA405_RECV_EMCY_DEV
- CIA405_NMT
- CANOPEN_VERSION
- NMT_GUARD_ERROR
- NMT_GUARD_ERROR_DEV
- CIA405_SDO_WRITE4
- CIA405_SDO_READ4
- CIA405_SDO_WRITEXX
- CIA405_SDO_READXX

The document containing a description of this library and the function blocks it includes is available for download on the Internet at www.wago.com.

List of Figures

List of Tables

WAGO Kontakttechnik GmbH & Co. KG
Postfach 2880 • D-32385 Minden
Hansastraße 27 • D-32423 Minden

Phone:	+49/5 71/8 87 – 0
Fax:	+49/5 71/8 87 – 1 69
E-Mail:	info@wago.com
Internet:	http://www.wago.com