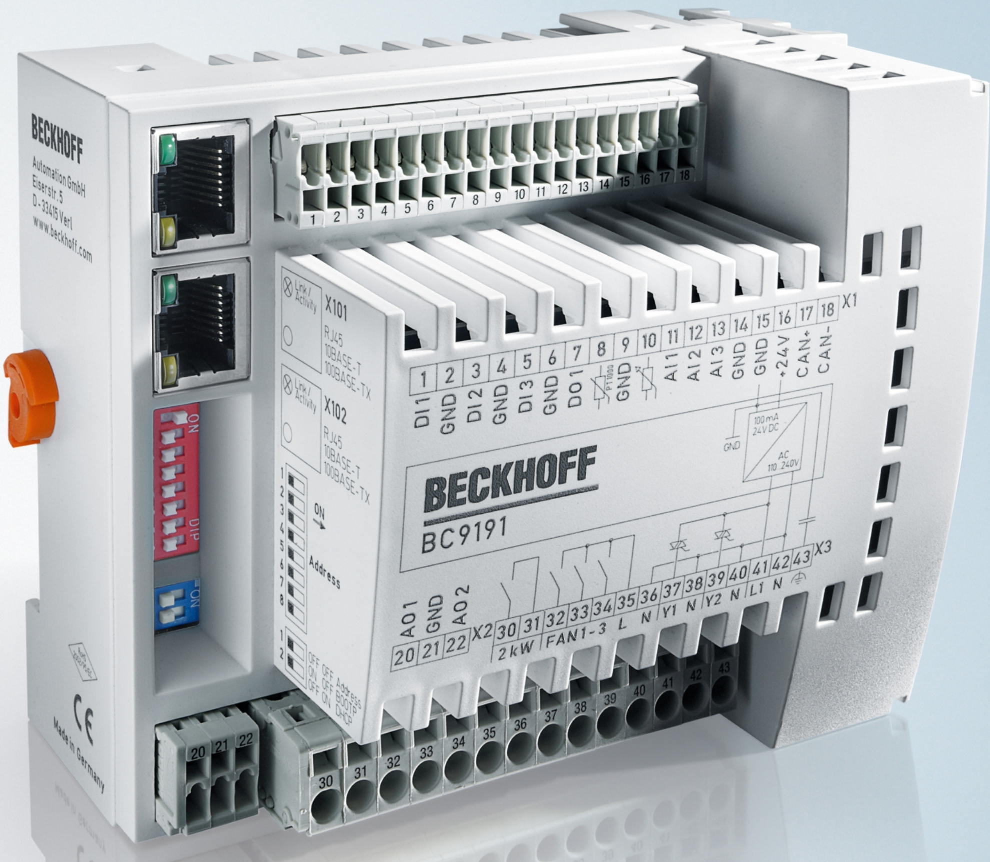
BECKHOFF New Automation Technology

Documentation | EN

BC9191 and BC9191-0100

Room Controller for Buiding Automation



2022-10-12 | Version: 3.1.0

Table of contents

1 Foreword

1.1 Notes on the documentation

Intended audience

This description is only intended for the use of trained specialists in control and automation engineering who are familiar with the applicable national standards.
It is essential that the documentation and the following notes and explanations are followed when installing and commissioning these components.
It is the duty of the technical personnel to use the documentation published at the respective time of each installation and commissioning.

The responsible staff must ensure that the application or use of the products described satisfy all the requirements for safety, including all the relevant laws, regulations, guidelines and standards.

Disclaimer

The documentation has been prepared with care. The products described are, however, constantly under development.

We reserve the right to revise and change the documentation at any time and without prior announcement.

No claims for the modification of products that have already been supplied may be made on the basis of the data, diagrams and descriptions in this documentation.

Trademarks

Beckhoff®, TwinCAT®, TwinCAT/BSD®, TC/BSD®, EtherCAT®, EtherCAT G®, EtherCAT G10®, EtherCAT P®, Safety over EtherCAT®, TwinSAFE®, XFC®, XTS® and XPlanar® are registered trademarks of and licensed by Beckhoff Automation GmbH. Other designations used in this publication may be trademarks whose use by third parties for their own purposes could violate the rights of the owners.

Patent Pending

The EtherCAT Technology is covered, including but not limited to the following patent applications and patents: EP1590927, EP1789857, EP1456722, EP2137893, DE102015105702 with corresponding applications or registrations in various other countries.

EtherCAT® is registered trademark and patented technology, licensed by Beckhoff Automation GmbH, Germany.

Copyright

1.2 Safety instructions

Safety regulations

Please note the following safety instructions and explanations!
Product-specific safety instructions can be found on following pages or in the areas mounting, wiring, commissioning etc.

Exclusion of liability

All the components are supplied in particular hardware and software configurations appropriate for the application. Modifications to hardware or software configurations other than those described in the documentation are not permitted, and nullify the liability of Beckhoff Automation GmbH & Co. KG.

Personnel qualification

This description is only intended for trained specialists in control, automation and drive engineering who are familiar with the applicable national standards.

Description of instructions

In this documentation the following instructions are used.
These instructions must be read carefully and followed without fail!

⚠ DANGER

Serious risk of injury!

Failure to follow this safety instruction directly endangers the life and health of persons.

⚠ WARNING

Risk of injury!

Failure to follow this safety instruction endangers the life and health of persons.

⚠ CAUTION

Personal injuries!

Failure to follow this safety instruction can lead to injuries to persons.

NOTE

Damage to environment/equipment or data loss

Failure to follow this instruction can lead to environmental damage, equipment damage or data loss.

Tip or pointer

This symbol indicates information that contributes to better understanding.

1.3 Notes on information security

The products of Beckhoff Automation GmbH & Co. KG (Beckhoff), insofar as they can be accessed online, are equipped with security functions that support the secure operation of plants, systems, machines and networks. Despite the security functions, the creation, implementation and constant updating of a holistic security concept for the operation are necessary to protect the respective plant, system, machine and networks against cyber threats. The products sold by Beckhoff are only part of the overall security concept. The customer is responsible for preventing unauthorized access by third parties to its equipment, systems, machines and networks. The latter should be connected to the corporate network or the Internet only if appropriate protective measures have been set up.

In addition, the recommendations from Beckhoff regarding appropriate protective measures should be observed. Further information regarding information security and industrial security can be found in our https://www.beckhoff.com/secguide.

Beckhoff products and solutions undergo continuous further development. This also applies to security functions. In light of this continuous further development, Beckhoff expressly recommends that the products are kept up to date at all times and that updates are installed for the products once they have been made available. Using outdated or unsupported product versions can increase the risk of cyber threats.

To stay informed about information security for Beckhoff products, subscribe to the RSS feed at https://www.beckhoff.com/secinfo.

1.4 Documentation issue status

Version	Comment
3.1.0	• Links updated
3.0.0	• Migration • Chapter *Notes on ESD protection* added • Chapter *Disposal* added • Chapter *Notes on information security* added • Safety instructions adapted to IEC 82079-1 • New title page
2.0.0	• Chapter *Programming* and *Parameterization and commissioning / IP address* added
1.1.0	• Chapter *BC9191 in master/slave mode* extended
1.0.0	• First release

Firmware BC9191

The label under the coupler will tell you which firmware is installed on the Bus Coupler in delivery state.

Example:
1416 **SW:3.4** HW:6.1 0000
The firmware in the example is 3.4.

Firmware	Description
3.4	Modbus UDP implemented
3.3	FW 3.3 was not used
3.2	Optimizations DHCP
3.1	First firmware version from series delivery BC9191-0100, Modbus TCP : Client functionality implemented
3.0	First firmware version from series delivery BC9191

2 Product overview

2.1 Introduction

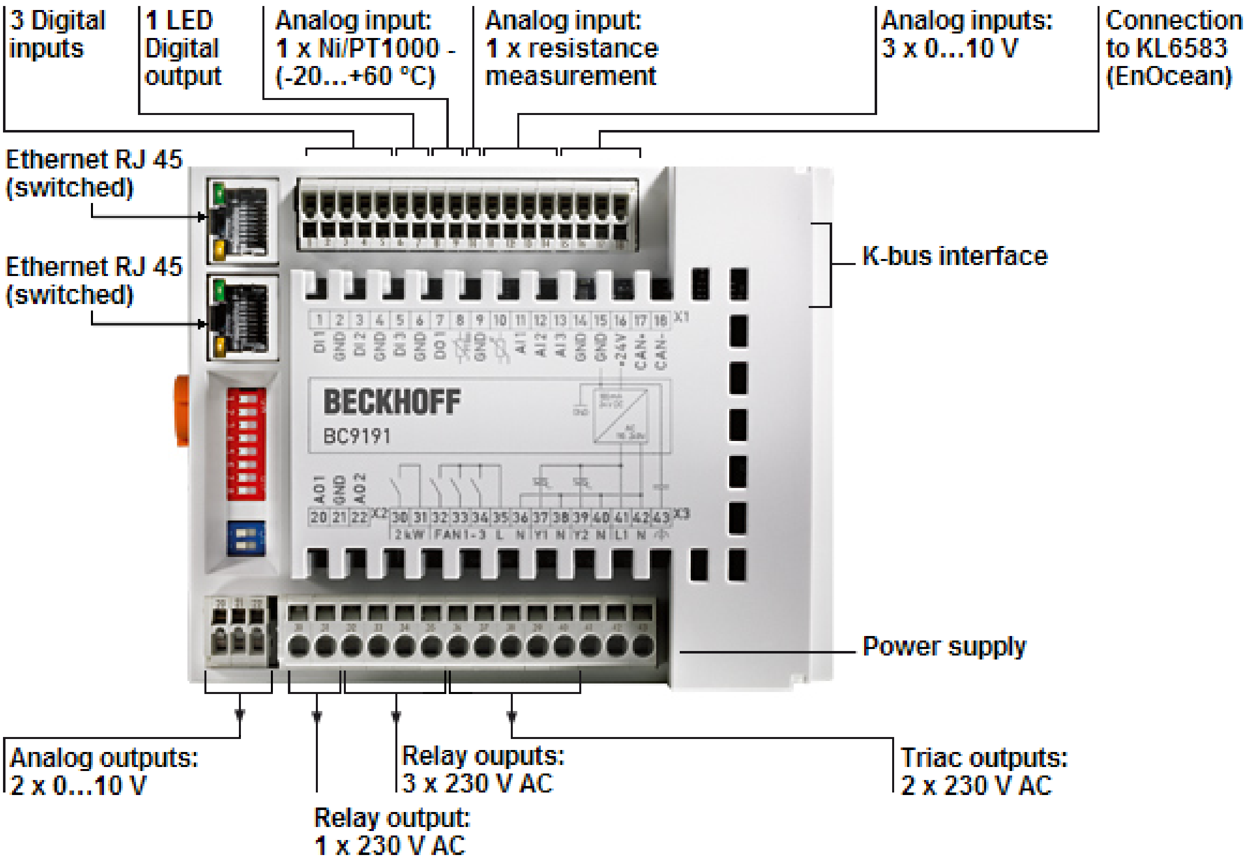


Fig. 1: BC9191

The BC9191 is a room controller in a compact design with decentralized intelligence.

The integrated digital and analog connections already cover the typical standard functionalities of room temperature control.
The K-bus interface enables extensions from the entire signal spectrum of the Beckhoff Bus Terminal system.

Two Ethernet interfaces are available for communication with a higher-level master computer or other BC9191s.

Furthermore, an additional sub-bus is available with both variants of the BC9191:

Variants of the BC9191

The BC9191 is available in two variants. These differ by the sub-bus, the memory extension and the cycle time.

- The variant BC9191 [▶ 23] provides a CAN bus for connecting the KL6583 EnOcean transceiver.
- The variant BC9191-0100 [▶ 25] provides an RS485 interface for the efficient connection of room control units.

Bus Terminal Controller	BC9191	BC9191-0100
Sub-Bus	EnOcean KL6853	RS485 / Modbus RTU / DMX
PLC memory	48 kbyte	128 kbyte
Data memory	32 kbyte	128 kbyte
PLC cycle time (for 1000 commands without I/O cycle and K-bus)	approx. 0.9 ms	approx. 0.7 ms

2.2 Extension of the BC9191 with Bus Terminals

Power supply of additional Bus Terminals and termination of the K-bus

The BC9191 does not have power contacts. Additionally inserted K-bus terminals must therefore be supplied with voltage by a power supply terminal.
If a further potential group is required, a further power supply terminal must be provided as in the example.
The K-bus must be terminated with the KL9010 end terminal. If the BC9191 is used without further terminals, do not use an end terminal, but insert the K-bus cover included in the scope of delivery.

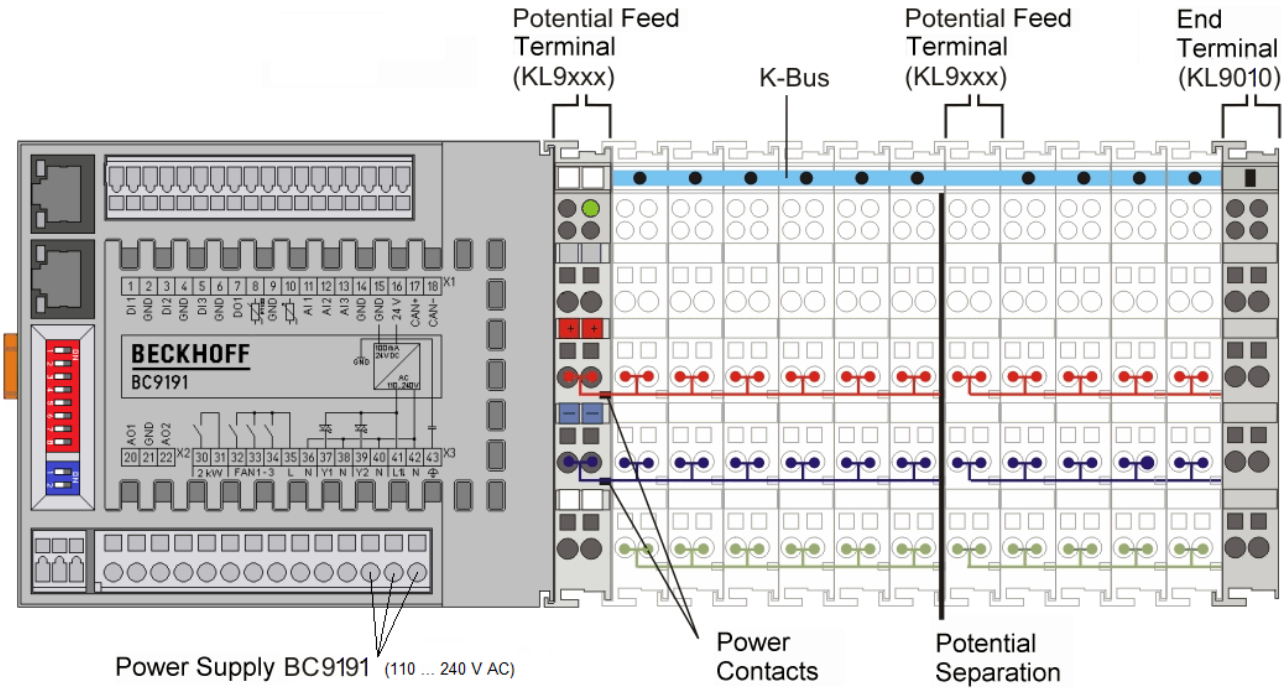


Fig. 2: Extension of the BC9191 with Bus Terminals

2.3 Technical data

Bus Terminal Controller	BC9191	BC9191-0100
Number of Bus Terminals	64	
Maximum fieldbus byte number	512 bytes input, 512 bytes output	
Digital inputs	3, for connecting potential-free contacts (e.g. for window contact, dew point sensor, occupancy sensor)	
Digital outputs	1 x LED output, constant current 10 mA , max. 24 V (e.g. for LED occupancy signal) 1 x 230 V_{AC}, 10 A, relay (e.g. for air heater) 3 x 230 V_{AC}, 1 A, relay (e.g. for 3-stage fan) 2 x 230 V_{AC}, 1 A, triac (e.g. for valves for heating and cooling)	
Analog inputs	1 x PT/Ni1000 (PT1000: -20...+60 °C, NI1000: -15...+55 °C) for air temperature measurement 1 x resistance measurement for setpoint setting (0...10 kOhm) 3 x 0...10 V (e.g. for CO_2 sensor, air quality sensor, light sensor)	
Analog outputs	2 x 0...10 V (e.g. air volume controller)	
Sub-bus systems	1 x K-bus 1 x system bus for KL6583 (EnOcean)	1 x K-bus 1 x RS485 (e.g. Modbus-RTU/ DMX)
Dielectric strength	500 V (supply voltage/fieldbus)	
Supply voltage	110 ... 240 V_{AC}	
Current consumption	typically 80 mA (230 V), max. 220 mA (85 V)	
Power loss	max. 15 W, typically 8 W (without 24 V_{DC} consumers and without further bus terminals)	
K-bus power supply	max. 200 mA	
24V_{DC} power supply	max. 100 mA	
Weight	approx. 350 g	
Permissible ambient temperature range during operation	0 °C ... +55 °C	
Permissible ambient temperature range during storage	-25 °C ... +85 °C	
Permissible relative humidity	max. 95 %, no condensation	
Vibration / shock resistance	conforms to EN 60068-2-6 / EN 60068-2-27	
EMC immunity / emission	conforms to EN 61000-6-2 / EN 61000-6-4	
Correct installation position	vertically on a mounting rail mounted horizontally on the wall (see chapter Installation position [▸ 17])	
Protection class	IP20	
Approvals/markings*	CE UKCA, EAC	

*) Real applicable approvals/markings see type plate on the side (product marking).

PLC	BC9191	BC9191-0100
Programming/configuration	with TwinCAT via Ethernet (TwinCAT 2.11 Build 2034 or higher, no TC3)	
Program memory	48 kbyte (minus task-configuration minus POUs during online change)	128 kbyte (minus task-configuration minus POUs during online change)
Data memory	32 kbyte	128 kbyte
Source code memory	256 kbyte	
Remanent flags	2 kbyte	
Persistent data	1000 bytes	
INPUT	2 kbyte	
OUTPUT	2 kbyte	
FLAG	4 kbyte	
max. variable size	16 kbyte	
Max. POUs	Limited by memory	
PLC cycle time for 1,000 commands (without I/O cycle, K-bus)	approx. 0.9 ms	approx. 0.7 ms
Number of runtime systems	1	
Programming languages	IEC 6-1131-3 (IL, LD, FBD, ST, SFC)	
Online Change	yes	

Fieldbus	BC9191	BC9191-0100
Fieldbus	Ethernet	
Ethernet connection	2 x RJ45 (internal switch)	
Auto-crossing	yes	
Data transfer rate	10/100 Mbaud	
Protocols	Modbus TCP/IP, Ethernet TCP/IP, Beckhoff ADS via TCP and UDP	
Data transfer medium	4 x 2 twisted-pair copper cable; category 3 (10 Mbaud), category 5 (100 Mbaud)	
Distance between modules	100 m (distributor hub to BC9191) Line: max. 100 m between two BC9191	
Topology	- star wiring - line wiring (20 x BC9191 in line maximum)	
Number of I/O modules	only limited by the IP address range	
Number of I/O points	depending on the higher-level controller	

Connections	BC9191	BC9191-0100
Number of ADS connections via TCP/IP	5	
Number of ADS connections via UDP/IP	1	
Number of ModbusTCP connections	3	
SMTP (Simple Mail Transfer Protocol)	yes	
SNTP (Simple Network Time Protocol)	yes	
BootP/DHCP	yes/yes	
UDP socket connections	3	
TCP socket connections	3	

Try using as few TCP/IP connections as possible

The number of TCP/IP connections should be reduced to a minimum. The fewer TCP/IP connections you use, the more time the BC9191 has for its actual task. If possible, use communication via ADS instead of TCP/IP or UDP/IP communication.

Communicate with "reasonable" time intervals for TCP/IP and for UDP/IP connections.
Example:
The task time of the BC9191 is 20 ms. In this case, you should not communicate with the BC9191 more often than every 40 ms.

2.4 The Beckhoff Bus Terminal system

Up to 256 Bus Terminals, with 1 to 16 I/O channels per signal form

The Bus Terminal system is the universal interface between a fieldbus system and the sensor / actuator level. A unit consists of a Bus Coupler as the head station, and up to 64 electronic series terminals, the last one being an end terminal. Up to 255 Bus Terminals can be connected via the K-Bus extension. For each technical signal form, terminals are available with one, two, four or eight I/O channels, which can be mixed as required. All the terminal types have the same mechanical construction, so that difficulties of planning and design are minimized. The height and depth match the dimensions of compact terminal boxes.

Decentralized wiring of each I/O level

Fieldbus technology allows more compact forms of controller to be used. The I/O level does not have to be brought to the controller. The sensors and actuators can be wired decentrally, using minimum cable lengths. The controller can be installed at any location within the plant.

Industrial PCs as controllers

The use of an Industrial PC as the controller means that the operating and observing element can be implemented in the controller's hardware. The controller can therefore be located at an operating panel, in a control room, or at some similar place. The Bus Terminals form the decentralized input/output level of the controller in the control cabinet and the subsidiary terminal boxes. The power sector of the plant is also controlled over the bus system in addition to the sensor/actuator level. The Bus Terminal replaces the conventional series terminal as the wiring level in the control cabinet. The control cabinet can have smaller dimensions.

Bus Couplers for all usual bus systems

The Beckhoff Bus Terminal system unites the advantages of a bus system with the possibilities of the compact series terminal. Bus Terminals can be driven within all the usual bus systems, thus reducing the controller parts count. The Bus Terminals then behave like conventional connections for that bus system. All the performance features of the particular bus system are supported.

Mounting on standardized mounting rails

The installation is standardized thanks to the simple and space-saving mounting on a standardized mounting rail (EN 60715, 35 mm) and the direct wiring of actuators and sensors, without cross connections between the terminals. The consistent labelling scheme also contributes.

The small physical size and the great flexibility of the Bus Terminal system allow it to be used wherever a series terminal is also used. Every type of connection, such as analog, digital, serial or the direct connection of sensors can be implemented.

Modularity

The modular assembly of the terminal strip with Bus Terminals of various functions limits the number of unused channels to a maximum of one per function. The presence of two channels in one terminal is the optimum compromise of unused channels and the cost of each channel. The possibility of electrical isolation through potential feed terminals also helps to keep the number of unused channels low.

Display of the channel state

The integrated LEDs show the state of the channel at a location close to the sensors and actuators.

K-Bus

The K-Bus is the data path within a terminal strip. The K-Bus is led through from the Bus Coupler through all the terminals via six contacts on the terminals' side walls. The end terminal terminates the K-Bus. The user does not have to learn anything about the function of the K-Bus or about the internal workings of the terminals and the Bus Coupler. Many software tools that can be supplied make project planning, configuration and operation easy.

Potential feed terminals for isolated groups

The operating voltage is passed on to following terminals via three power contacts. You can divide the terminal strip into arbitrary isolated groups by means of potential feed terminals. The potential feed terminals play no part in the control of the terminals, and can be inserted at any locations within the terminal strip.

Up to 64 Bus Terminals can be used in a terminal block, with optional K-Bus extension for up to 256 Bus Terminals. This count does include potential feed terminals, but not the end terminal.

Bus Couplers for various fieldbus systems

Various Bus Couplers can be used to couple the electronic terminal strip quickly and easily to different fieldbus systems. It is also possible to convert to another fieldbus system at a later time. The Bus Coupler performs all the monitoring and control tasks that are necessary for operation of the connected Bus Terminals. The operation and configuration of the Bus Terminals is carried out exclusively by the Bus Coupler. Nevertheless, the parameters that have been set are stored in each Bus Terminal, and are retained in the event of voltage drop-out. Fieldbus, K-Bus and I/O level are electrically isolated.

If the exchange of data over the fieldbus is prone to errors or fails for a period of time, register contents (such as counter states) are retained, digital outputs are cleared, and analog outputs take a value that can be configured for each output when commissioning. The default setting for analog outputs is 0 V or 0 mA. Digital outputs return in the inactive state. The timeout periods for the Bus Couplers correspond to the usual settings for the fieldbus system. When converting to a different bus system it is necessary to bear in mind the need to change the timeout periods if the bus cycle time is longer.

The interfaces

A Bus Coupler has six different methods of connection. These interfaces are designed as plug connectors and as spring-loaded terminals.

3 Installation

3.1 Instructions for ESD protection

NOTE

Destruction of the devices by electrostatic discharge possible!

The devices contain components at risk from electrostatic discharge caused by improper handling.

- Please ensure you are electrostatically discharged and avoid touching the contacts of the device directly.
- Avoid contact with highly insulating materials (synthetic fibers, plastic film etc.).
- Surroundings (working place, packaging and personnel) should by grounded probably, when handling with the devices.
- Each assembly must be terminated at the right hand end with a KL9010 bus end terminal, to ensure the protection class and ESD protection.

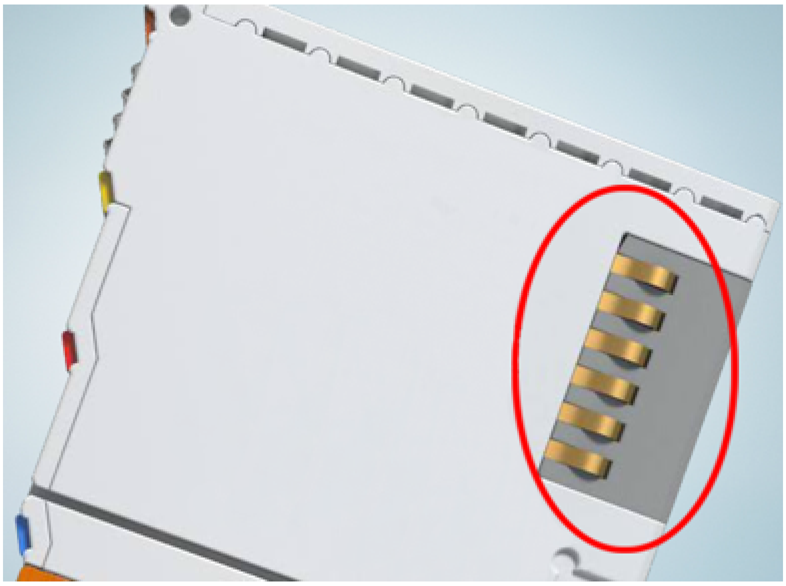


Fig. 3: Spring contacts of the Beckhoff I/O components

3.2 Assembly

3.2.1 Dimensions

The BC9191 and the BC9191-0100 are characterized by their low physical volume. The dimensions given apply to both variants.

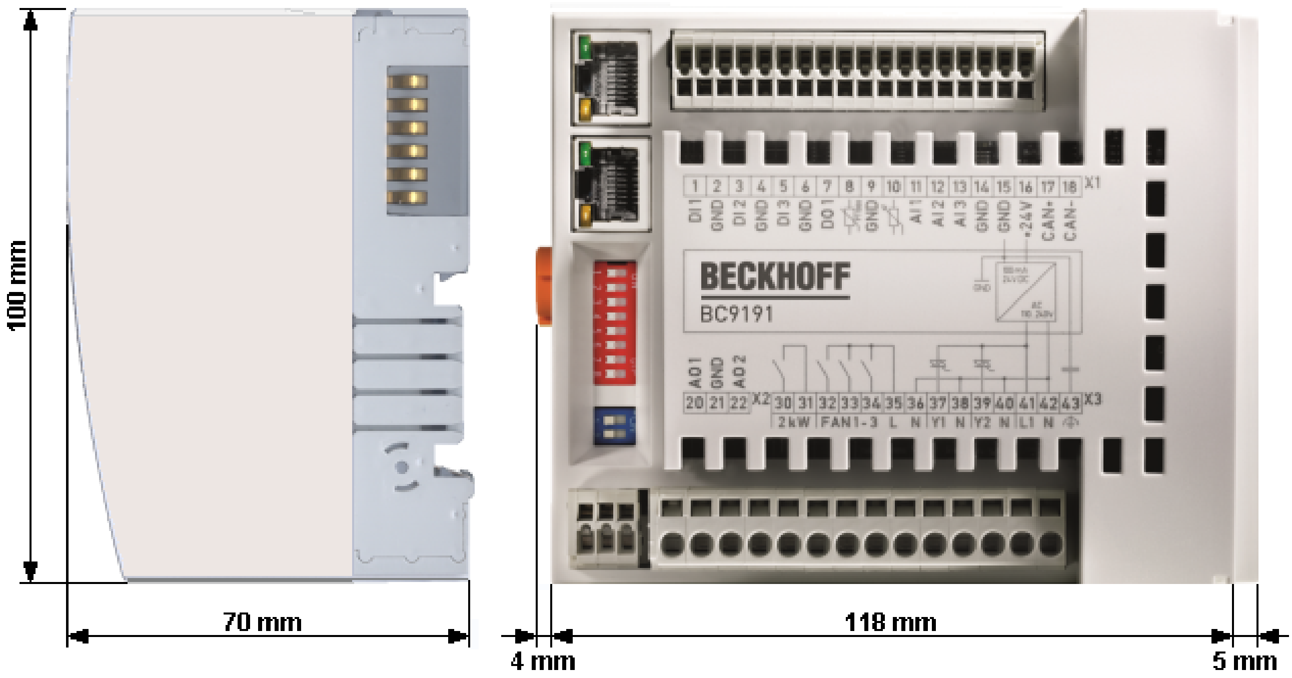


Fig. 4: Dimensions of the BC9191

If the BC9191 or BC9191-0100 is extended by Bus Terminals, the total width is made up of the width of the Bus Terminal Controller, the width of the Bus Terminals used and the KL9010 bus end terminal. The right-hand cover (5 mm) is then omitted. Depending on design, the Bus Terminals are 12 mm or 24 mm wide. The depth of the Bus Terminals of 68 mm is increased by the front wiring by approx. 5 mm to 10 mm depending on the wire diameter.

3.2.2 Installation position and minimum distances

NOTE

Mounting specifications

To ensure optimum function of the BC9191, observe the following specifications when installing it

- Installation position
- Minimum distances
- Ambient temperature

Installation position

Mount the mounting rail horizontally so that the connection surfaces of the snapped-on BC9191 face forward (see figure *Minimum distances*).
The BC9191 is then ventilated from bottom to top, which enables optimum cooling of the electronics by convection. "From below" is relative to the acceleration of gravity.

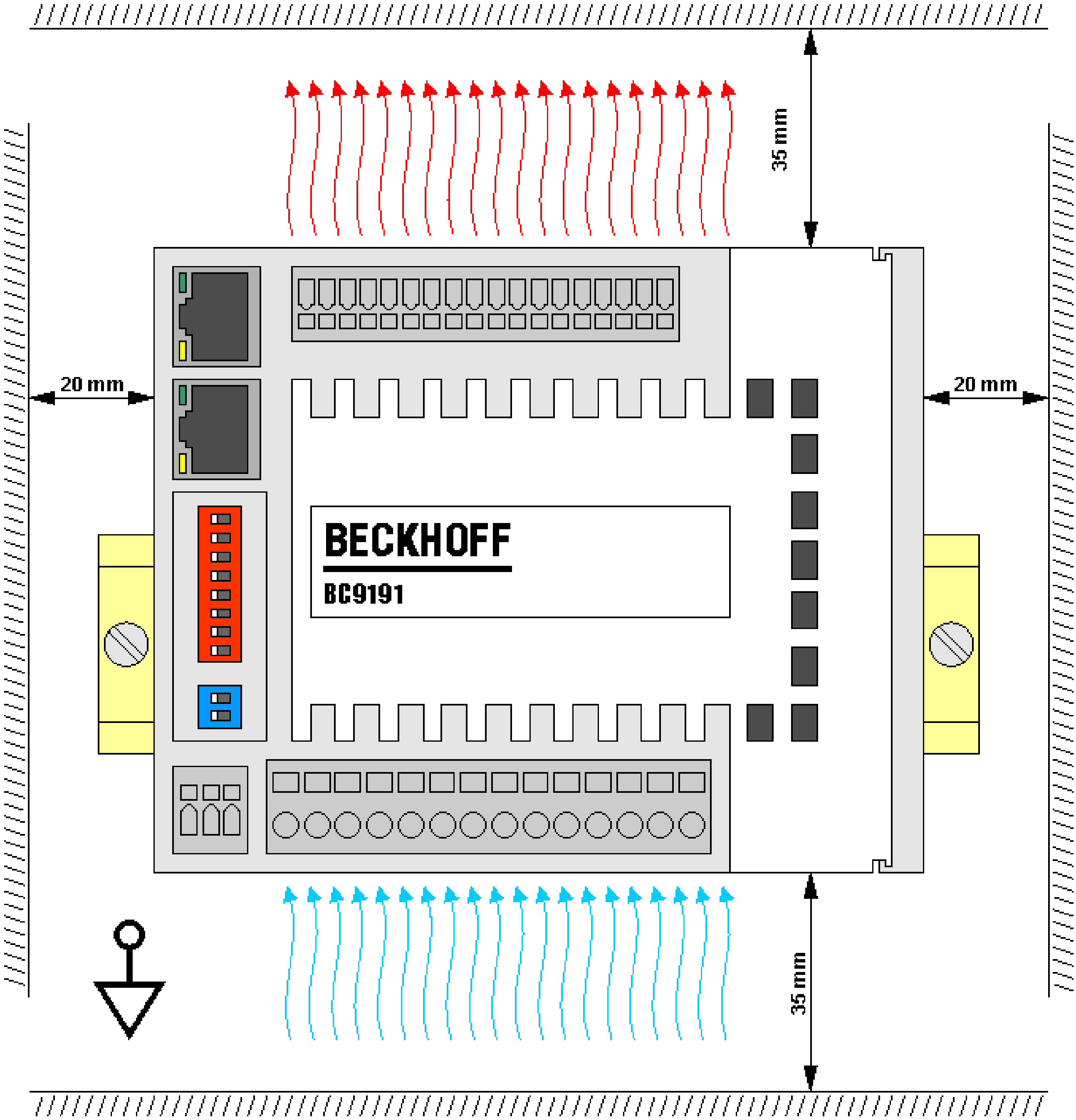


Fig. 5: BC9191 - Minimum distances

Minimum distances

When mounting, adhere to the minimum distances to other components and the walls of the control cabinet or terminal box employed as specified in the illustration above.

Ambient temperature

Ensure adequate ventilation so that the permissible ambient temperature range for the BC9191 is maintained in the control cabinet or terminal box employed (see Technical data [▸ 12])!

3.2.3 Installation on mounting rails

⚠ WARNING

Risk of injury through electric shock and damage to the device!

Bring the BC9191 and the Bus Terminals into a safe, de-energized state before starting installation, disassembly or wiring!

Assembly

The BC9191 is snapped onto standard 35 mm mounting rails (DIN rails according to EN 60715) by applying slight pressure:

1. Press the Bus Terminal Controller onto the mounting rail.
2. With slight pressure, two latching lugs on the right-hand side then automatically engage in the mounting rail.
3. On the left side, insert a screwdriver into the upper notch of the orange latch and engage the latch with a twisting motion.
4. Bus Terminals are now installed. To do this, plug the components together with tongue and groove and push the terminals against the mounting rail until the latch audibly engages on the mounting rail. If you first snap the terminals onto the mounting rail and then push them next to each other without the tongue and groove interlocking, no functional connection will be established! When correctly assembled, no significant gap should be visible between the housings.
5. If no Bus Terminals are attached, plug the K-bus cover with tongue and groove onto the BC9191 and push it against the mounting rail until the latch audibly engages on the mounting rail.

Nature and source of the danger

The locking mechanism of the terminals and couplers protrudes into the profile of the mounting rail. When installing the components, make sure that the locking mechanism doesn't come into conflict with the fixing bolts of the mounting rail. For fastening mounting rails with a height of 7.5 mm under the terminals and couplers, use flat fastening components such as countersunk head screws or blind rivets.

Disassembly

The BC9191 is secured to the mounting rail on the left-hand side by an orange rotary latch and on the right-hand side by a pull latch, which must be released for removal:

1. Insert a screwdriver into the lower recess of the orange rotary latch and release it by pressing on the screwdriver. The retaining lug of the orange rotary latch then releases the mounting rail.
2. Lever the unlatching hook on the right-hand side of the BC9191 upwards with a screwdriver. An internal mechanism pulls the two latching lugs from the DIN rail back into the terminal module,
3. The lock with the mounting rail is now released and the Bus Terminal Controller can be pulled from the mounting rail without excessive force.

Connections within a Bus Terminal block

The electric connections between the Bus Terminal Controller and the Bus Terminals are automatically realized by joining the components:

- The six spring contacts of the K-bus deal with the transfer of the data and the supply of the Bus Terminal electronics.
- The power contacts deal with the supply for the field electronics and thus represent a supply rail within the Bus Terminal block. The power contacts of the BC9191 must be supplied via potential supply terminals (KL9xxx).

Power contacts

During the design of a Bus Terminal block, the pin assignment of the individual Bus Terminals must be taken account of, since some types (e.g. analog Bus Terminals or digital 4-channel Bus Terminals) do not or not fully loop through the power contacts. Power supply terminals (KL91xx, KL92xx and EL91xx, EL92xx) interrupt the power contacts and thus represent the start of a new supply rail.

3.3 Wiring

3.3.1 Internal power supply, GND, potential groups, insulation test and PE

Internal power supply

The BC9191 has an integrated wide-range power supply unit:

- The AC input voltage may be in the range of 110 V_{AC} to 240 V_{AC}.
- The integrated power supply unit generates all internally required voltages and also the electrically isolated voltage 24 V_{DC} (pin X1/16).
- All GND connections are internally interconnected. To avoid interference, GND is capacitively connected to PE and the spring contacts for the mounting rail.
- The 24 V_{DC} at pin X1/16, which is electrically isolated from the mains, can be loaded with max. 100 mA.

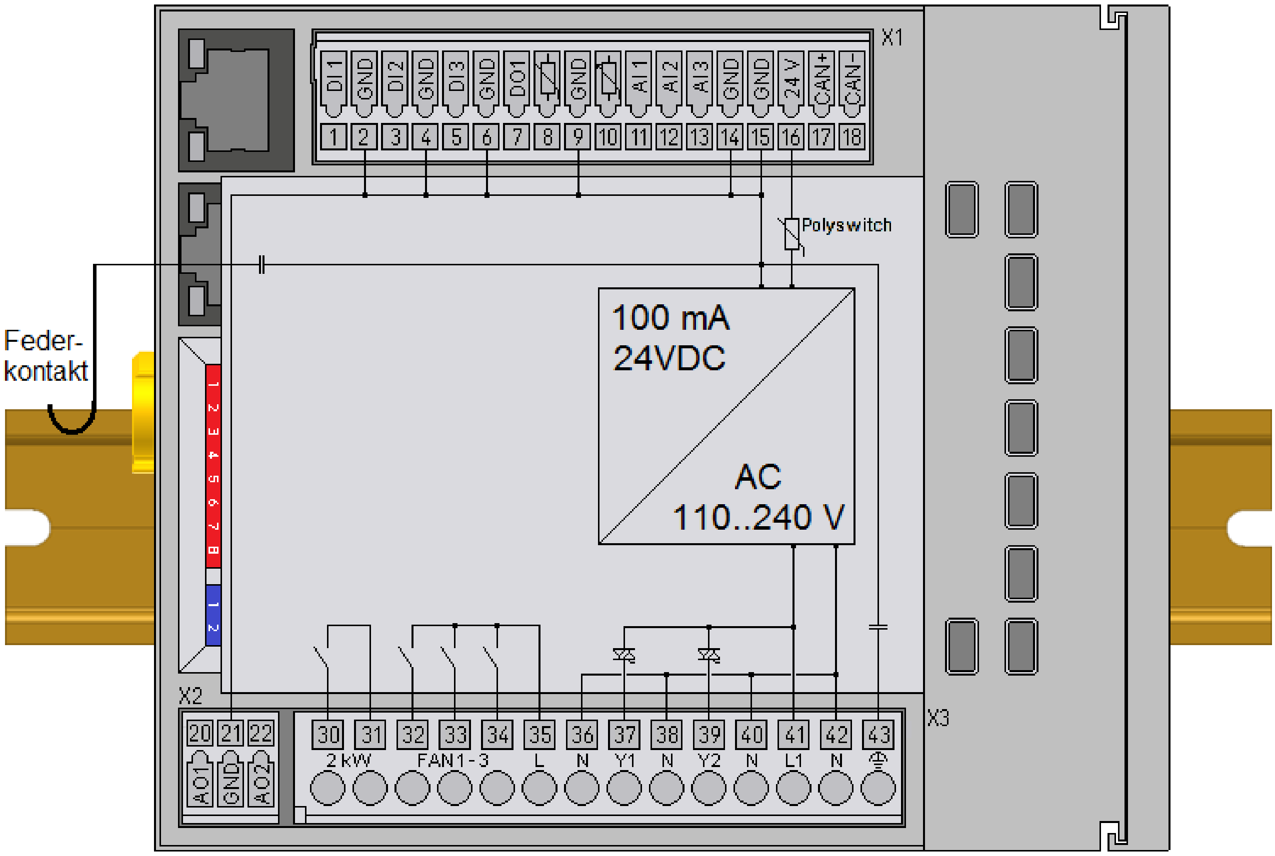


Fig. 6: BC9191 - Schematic diagram of the internal power supply

Potential groups

The BC9191 does not have power contacts. The power contacts of connected Bus Terminals must therefore be supplied by a power supply terminal. If an additional potential group is required, an additional power supply terminal must be provided, as in the example.

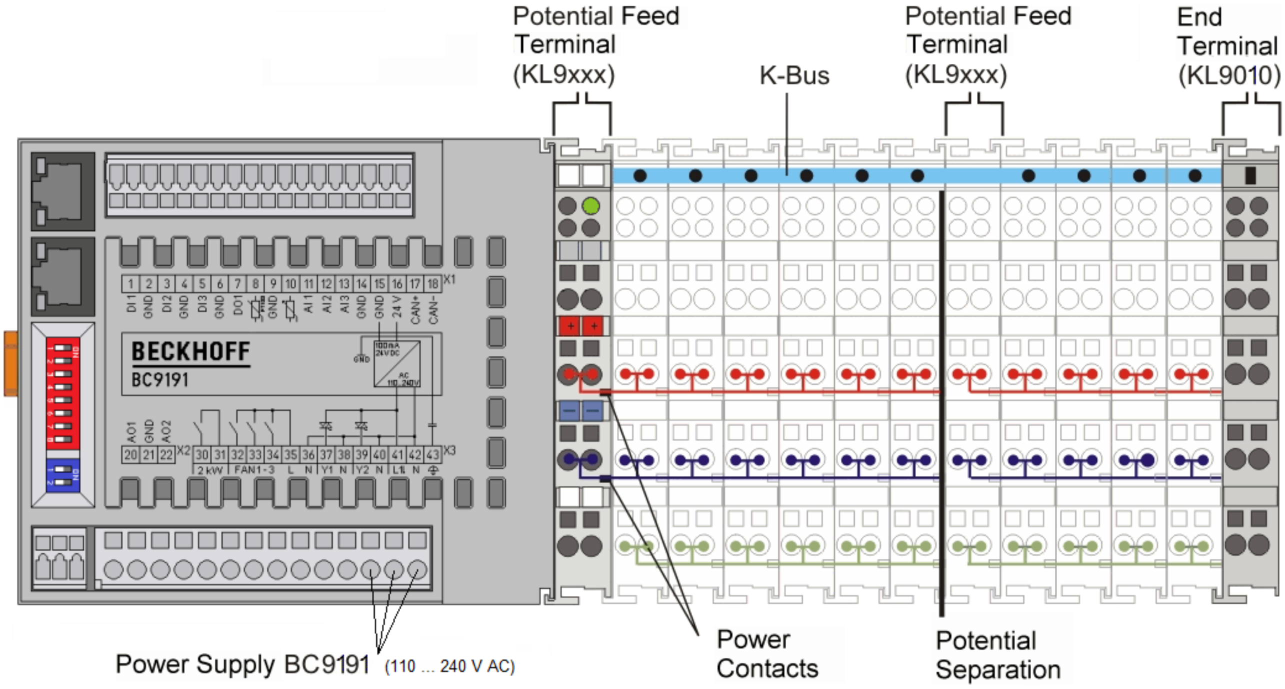


Fig. 7: BC9191 - Supply of the power contacts of connected Bus Terminals

Power contacts

The connection between Bus Coupler / Bus Terminal Controller and Bus Terminals is realized automatically by latching the components. The transfer of the data and the supply voltage for the intelligent electronics in the Bus Terminals is performed by the K-bus. The supply of the field electronics is performed through the power contacts. Plugging together the power contacts creates a supply rail. Since some Bus Terminals (e.g. analog Bus Terminals or digital 4-channel Bus Terminals) do not loop through these power contacts or not completely, the pin assignments of the Bus Terminals must be observed.

The power supply terminals interrupt the power contacts, and represent the start of a new supply rail.

Insulation testing

The power contact labeled PE can be used as a protective earth. The contact is leading for safety reasons when mated and can dissipate short circuit currents up to 125 A.

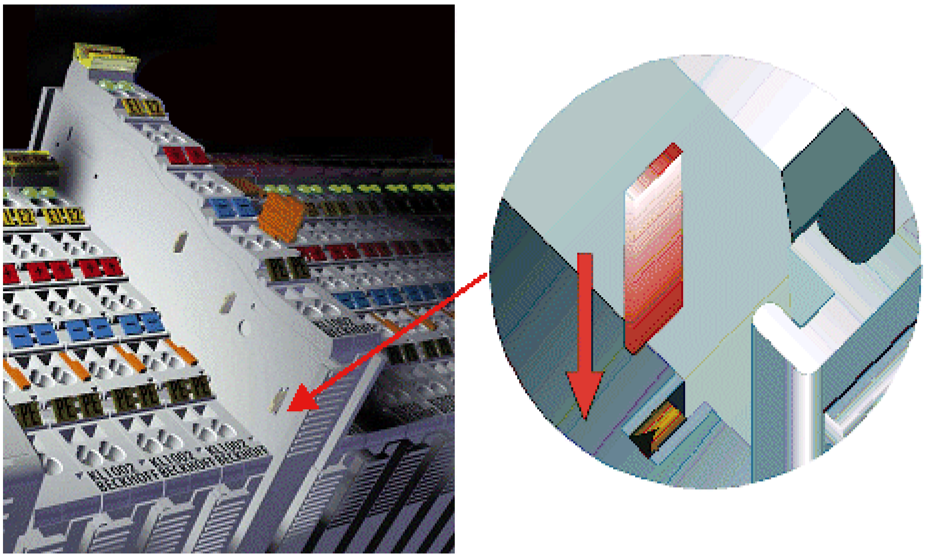


Fig. 8: Power contact on the left

It should be noted that, for reasons of electromagnetic compatibility, the PE contacts are capacitively coupled to the mounting rail. This can both lead to misleading results and to damaging the terminal during insulation testing (e.g. breakdown of the insulation from a 230 V power consuming device to the PE conductor). The PE supply line at the Bus Coupler / Bus Terminal Controller must be disconnected for an insulation test. In order to uncouple further supply locations for the purposes of testing, the power supply terminals can be pulled at least 10 mm out from the connected group of other terminals. In that case, the PE conductors do not have to be disconnected.

The PE power contact must not be used for other potentials.

3.3.2 BC9191 - Terminal strip X1

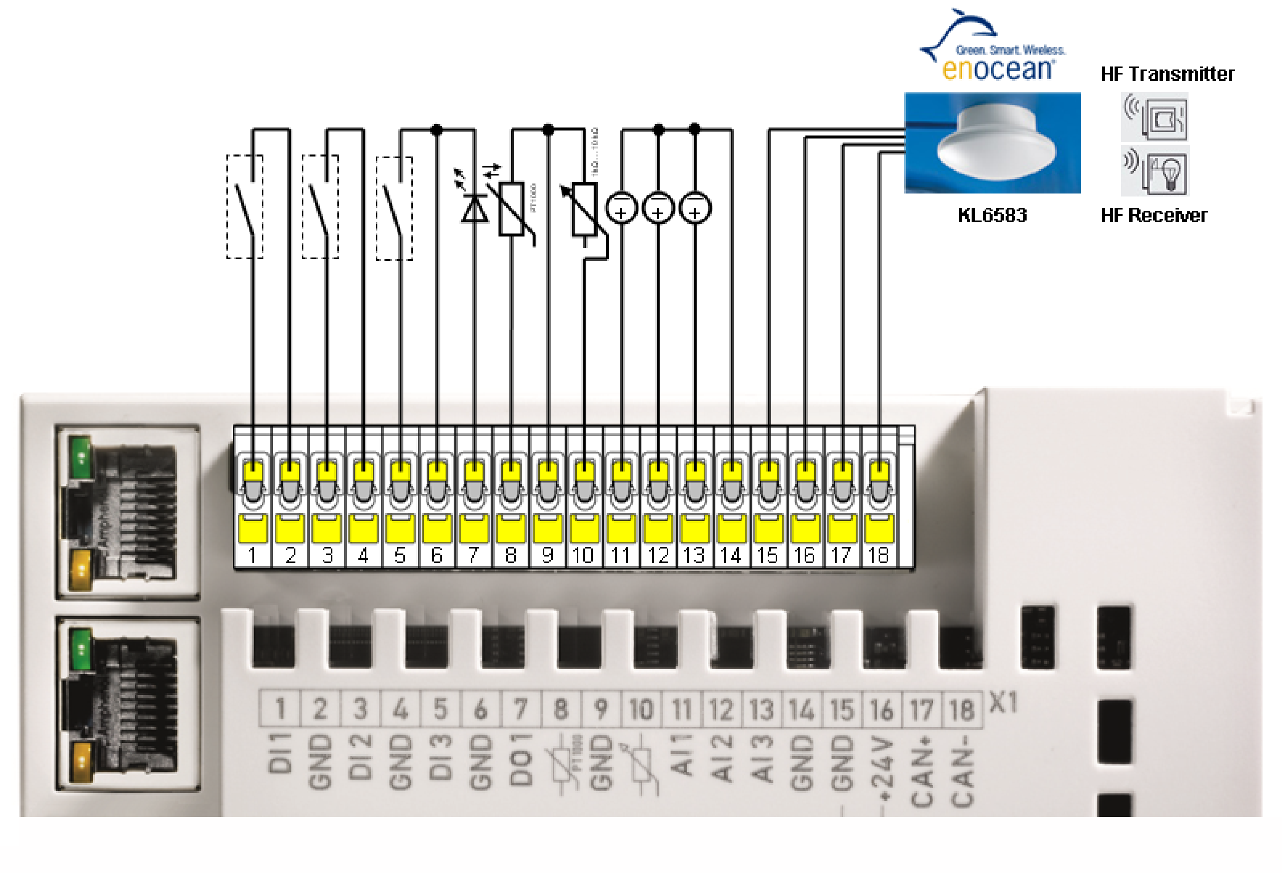


Fig. 9: BC9191 - Terminal strip X1 with interface for enocean

Terminal point	Designation	Connection	Comment
1	DI 1	Input for switching contact 1	For potential-free contact, e.g. window contact
2	GND	GND	GND for switching contact 1
3	DI 2	Input for switching contact 2	For potential-free contact, e.g. dew point
4	GND	GND	GND for switching contact 2
5	DI 3	Input for switching contact 3	For potential-free contact, e.g. occupancy sensor
6	GND	GND	GND for switching contact 3 and LED
7	DO1	LED output	Output for LED, e.g. display on room control unit1 with constant current 10 mA , max. 24 V
8	PT1000	Input PT/NI 1000	Connection for PT/NI 1000
9	GND	GND	GND for PT/NI 1000 and set value generator
10	Potentiometer	Potentiometer	Connection for setpoint generator (potentiometer 1 kΩ ...10 kΩ)
11	AI 1	Analog input 1	Analog input 0…10 V (also usable as 24 V digital input)
12	AI 2	Analog input 2	Analog input 0…10 V (also usable as 24 V digital input)
13	AI 3	Analog input 3	Analog input 0…10 V (also usable as 24 V digital input)
14	GND	GND	0 V potential of the analog input signals
15	GND	GND	GND of the data cable for KL6583 (shield of the data cable should be connected to PE)
16	+24 V	+24 V direct voltage	max. 100 mA for supply of KL6583 (EnOcean) or external operating device. Internal PolySwitch fuse that can be reset after triggering by switching off the BC9191 or disconnecting the inadmissible consumer.
17	CAN+	CAN +	Data cable for KL6583 (EnOcean)
18	CAN+	CAN -	

See also the EnOcean application example on the BC9191:
https://infosys.beckhoff.com/content/1033/tcplclibenocean/11985604235.html

Potential-free contacts for the digital inputs at terminal strip X1

The digital inputs of the BC9191 expect potential-free contacts at the connections.
24 V signals must not be used.

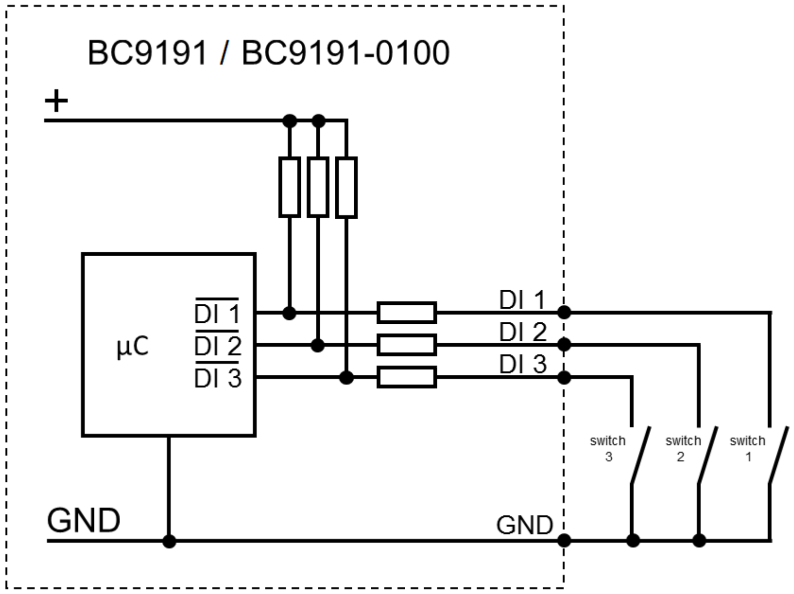


Fig. 10: Block diagram of the integrated digital inputs

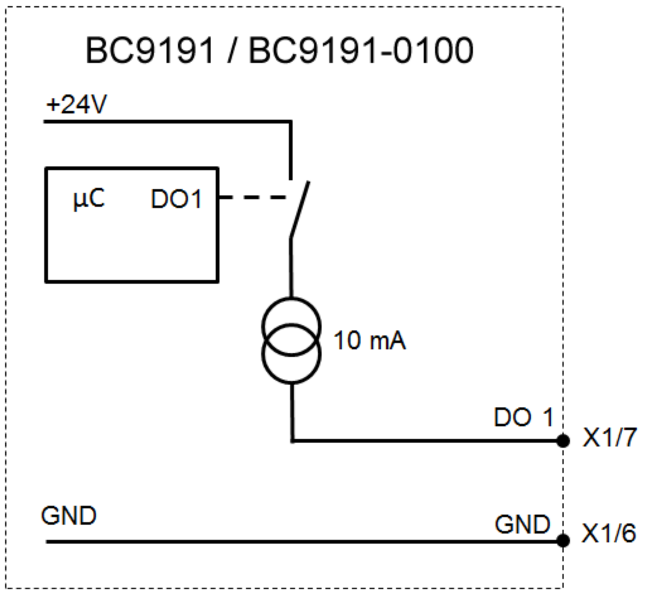


Fig. 11: Block diagram LED output

3.3.3 BC9191-0100 - Terminal strip X1

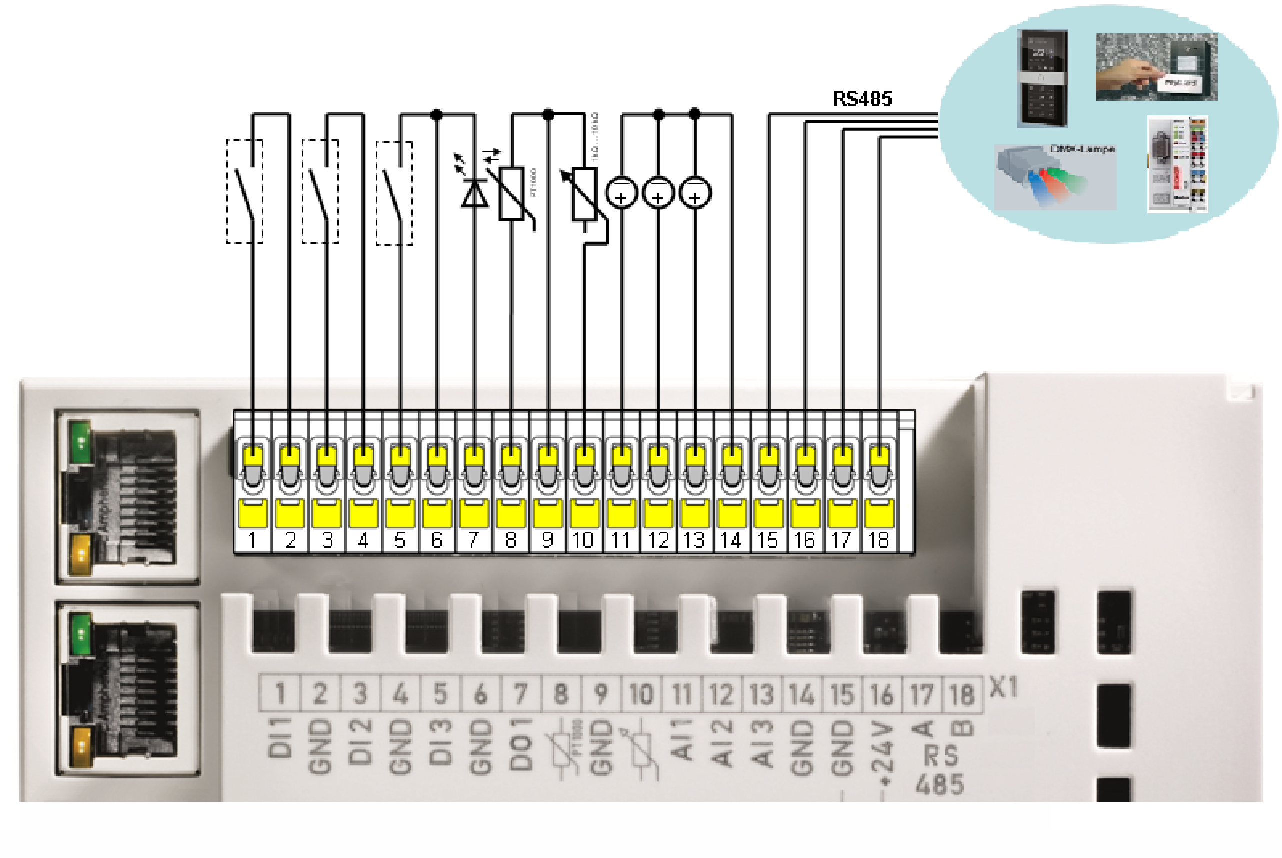


Fig. 12: BC9191-0100 - Terminal strip X1 with RS485 interface

Terminal point	Designation	Connection	Comment
1	DI 1	Input for switching contact 1	For potential-free contact, e.g. window contact
2	GND	GND	GND for switching contact 1
3	DI 2	Input for switching contact 2	For potential-free contact, e.g. dew point
4	GND	GND	GND for switching contact 2
5	DI 3	Input for switching contact 3	For potential-free contact, e.g. occupancy sensor
6	GND	GND	GND for switching contact 3 and LED
7	DO1	LED output	Output for LED, e.g. display on room control unit1 with constant current 10 mA , max. 24 V
8	PT1000	Input PT/NI 1000	Connection for PT/NI 1000
9	GND	GND	GND for PT/NI 1000 and set value generator
10	Potentiometer	Potentiometer	Connection for setpoint generator (potentiometer 1 kΩ ...10 kΩ)
11	AI 1	Analog input 1	Analog input 0…10 V (also usable as 24 V digital input)
12	AI 2	Analog input 2	Analog input 0…10 V (also usable as 24 V digital input)
13	AI 3	Analog input 3	Analog input 0…10 V (also usable as 24 V digital input)
14	GND	GND	0 V potential of the analog input signals
15	GND	GND	GND of the RS485 (shield of the bus cable should be connected to PE)
16	+24 V	+24 V direct voltage	max. 100 mA for supply of KL6583 (EnOcean) or external operating device. Internal PolySwitch fuse that can be reset after triggering by switching off the BC9191 or disconnecting the inadmissible consumer.
17	RS485	RS485/A	RS485 data cable, e.g. Modbus RTU
18	RS485	RS485/B	

See also application example DMX master with BC9191-0100:
https://infosys.beckhoff.com/content/1033/tcplclibdmx/11977733643.html

Potential-free contacts for the digital inputs at terminal strip X1

The digital inputs of the BC9191 expect potential-free contacts at the connections.
24 V signals must not be used.

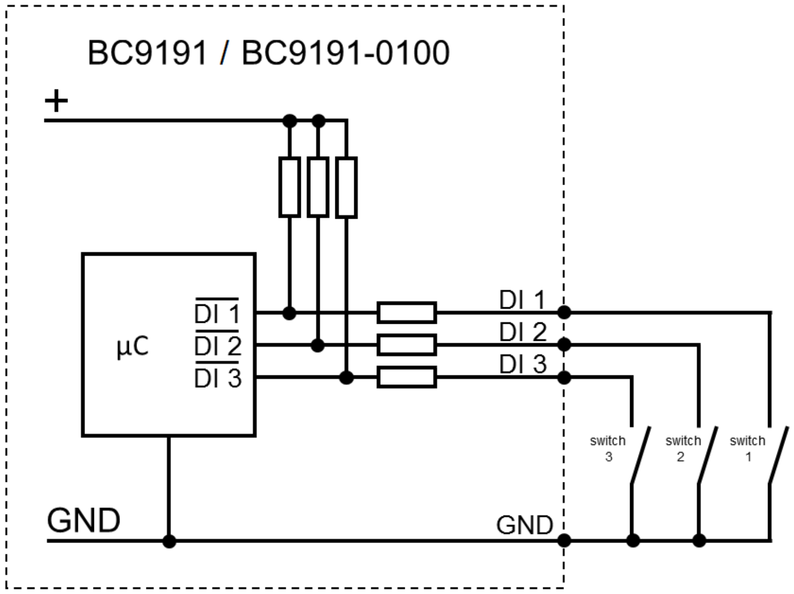


Fig. 13: Block diagram of the digital inputs

The digital output DO1 is intended for direct connection of an LED (10 mA).

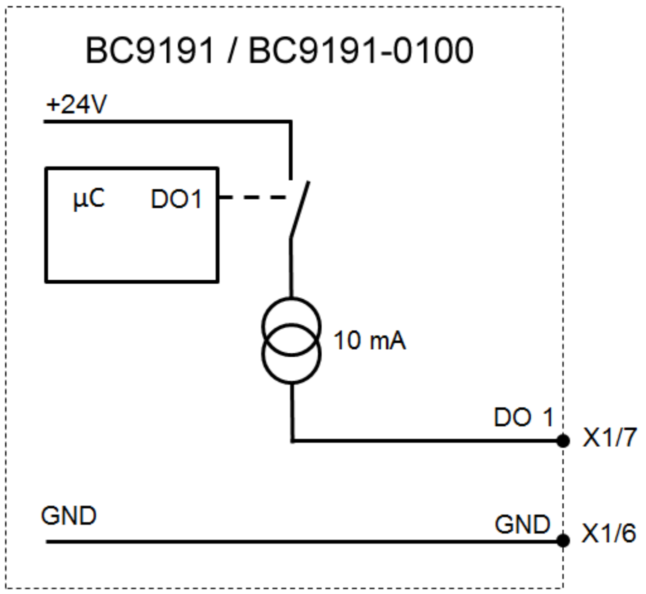


Fig. 14: Block diagram LED output

3.3.4 Terminal strips X2 and X3

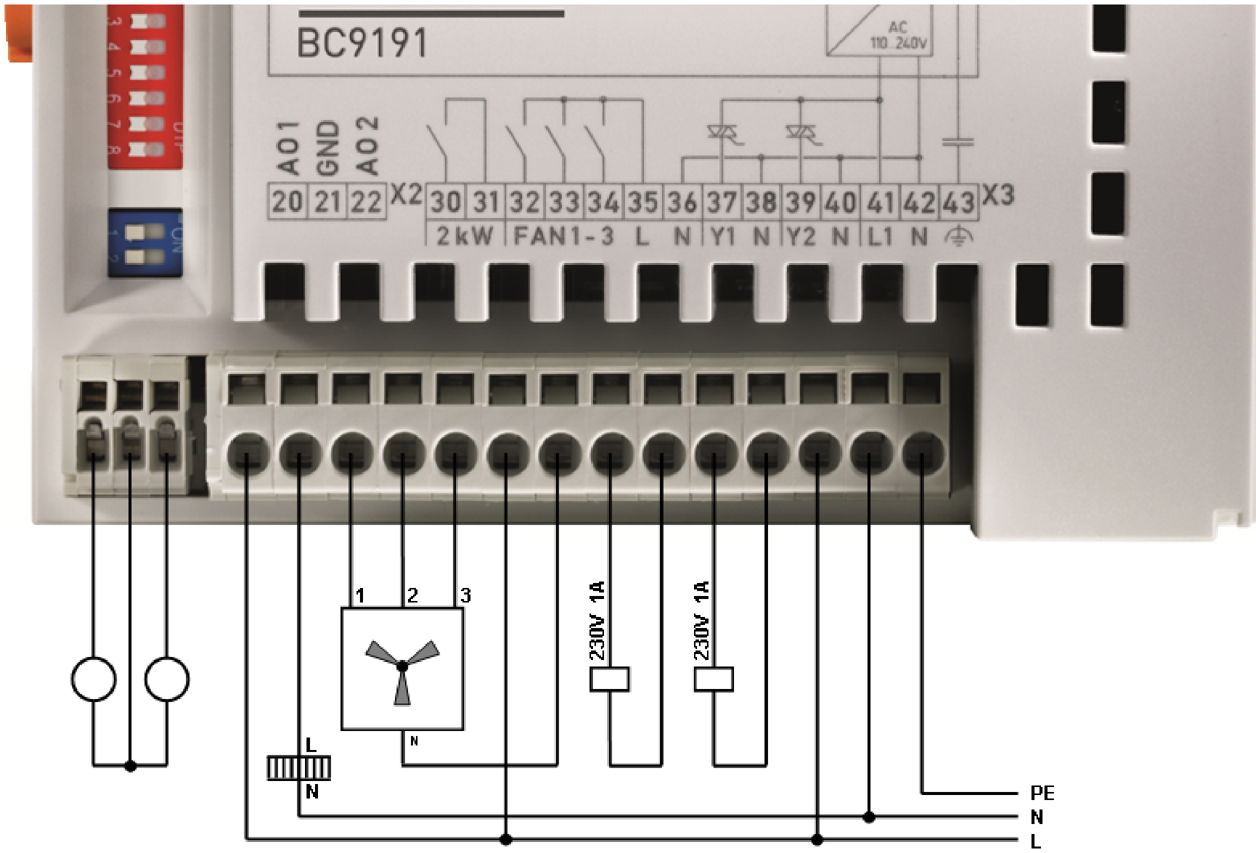


Fig. 15: BC9191 and BC9191-0100 - Terminal strips X2 and X3

Power supply

AC voltage connection

The BC9191 Bus Terminal Controller uses an integrated power supply to generate the supply voltage (Us) of 24 V_{DC} required for operation.

The connection for the integrated wide-range power supply unit is made via connections 41 (L), 42 (N) and 43 (PE) of terminal strip X3.
This supply voltage supplies the electronics of the BC9191 as well as the electronics of the connected Bus Terminals via the K-bus.
It is electrically isolated from the field level voltage.

Power contacts supply (Up)

If the BC9191 is expanded with Bus Terminals, these must be supplied with Up via a power supply terminal.

Terminal strip X2

Terminal point	Designation	Connection	Comment
20	AO 1	Analog output 1	Analog output 0...10 V, e.g. air volume control
21	GND	GND	GND for analog outputs
22	AO 2	Analog output 2	Analog output 0...10 V, e.g. air volume control

Both analog outputs can be loaded with max. 10 mA. A connected actuator should have an internal resistance greater than 1 kOhm.

Terminal strip X3

Terminal point	Designation	Connection	Comment
30	2 kW	K1	Potential-free relay, 230 V~, 10 A, e.g. for electrical auxiliary heating
31	2 kW	K2	Potential-free relay, 230 V~, 10 A, e.g. for electrical auxiliary heating
32	FAN 1	Fan Speed 1	Relay output, 230 V~, e.g. low fan speed
33	FAN 2	Fan Speed 2	Relay output, 230 V~, e.g. medium fan speed
34	FAN 3	Fan Speed 3	Relay output, 230 V~, e.g. high fan speed
35	L	Phase	Supply for relay, fan speeds 1 to 3
36	N	Neutral conductor	Neutral conductor, e.g. for fan
37	Y1	Triac output 1	Triac output 1, 230 V~, max 1 A, switches at zero crossing, e.g. "heating" valve
38	N	N	Neutral conductor for triac output 1
39	Y2	Triac output 2	Triac output 2, 230 V~, max 1 A, switches at zero crossing, e.g. "cooling" valve
40	N	N	Neutral conductor for triac output 2
41	L1	L1	Power supply connection 230 V~
42	N	Neutral conductor	Power supply connection N
43	FE	Functional earth	Ground connection, capacitively connected to GND

3.3.5 Ethernet topologies

BK9100, BC9100, BC9120, BC9191

These Bus Couplers and Bus Terminal controllers have an internal triple switch with one internal and two external ports. The internal switch enables the simple construction of a linear topology. A maximum of 20 BK9100/BC91x0/BC9191 can be connected in series in a physical line. However the distance between two Ethernet devices may not exceed 100 m. The maximum overall line length is therefore 2 km. No further switches may be included in this line.

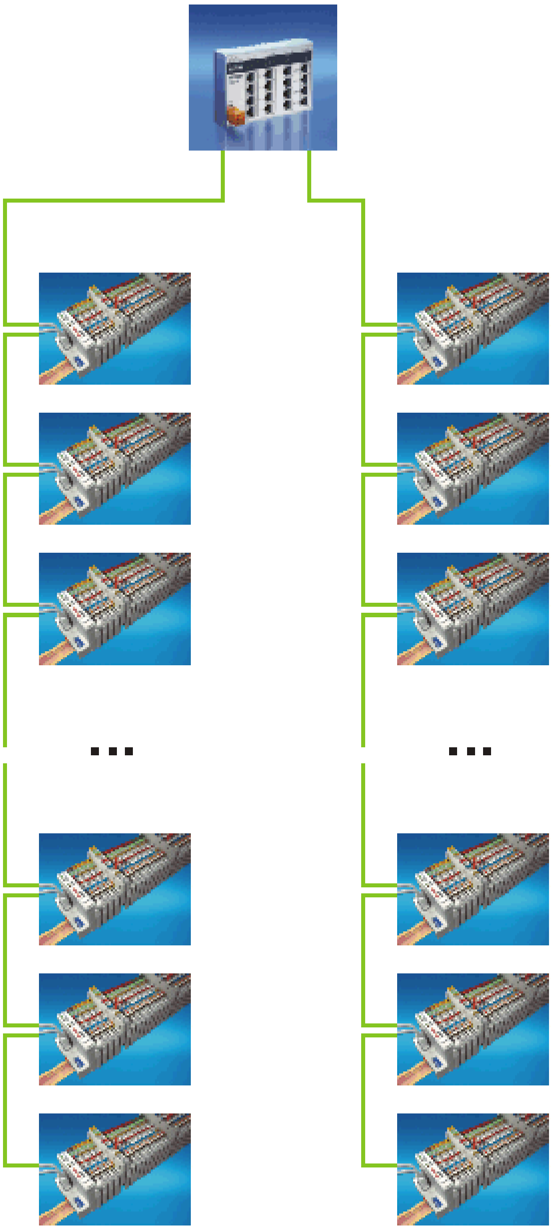


Fig. 16: Ethernet layout in linear topology

Of course, the construction of a classic star topology is also possible with these Bus Couplers and Bus Terminal controllers.

3.3.6 Ethernet connection

The connection to the Ethernet bus is made via an RJ45 plug (a Western plug).

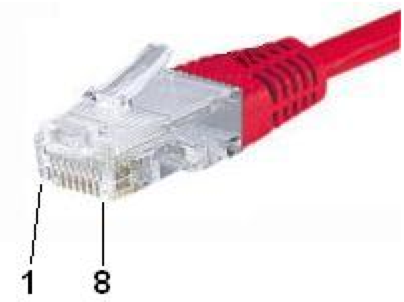


Fig. 17: RJ45 plug

Pin assignment of the RJ45 plug

PIN	Signal	Description
1	TD +	Transmit +
2	TD -	Transmit -
3	RD +	Receive +
4	-	reserved
5	-	reserved
6	RD -	Receive -
7	-	reserved
8	-	reserved

Direct connection between PC with Ethernet card and BC9191

The BC9191 supports auto-crossing. A crossed or an uncrossed Ethernet cable can be used to connect a PC directly to the BC9191. The internal switch in the BC9191 detects this automatically.

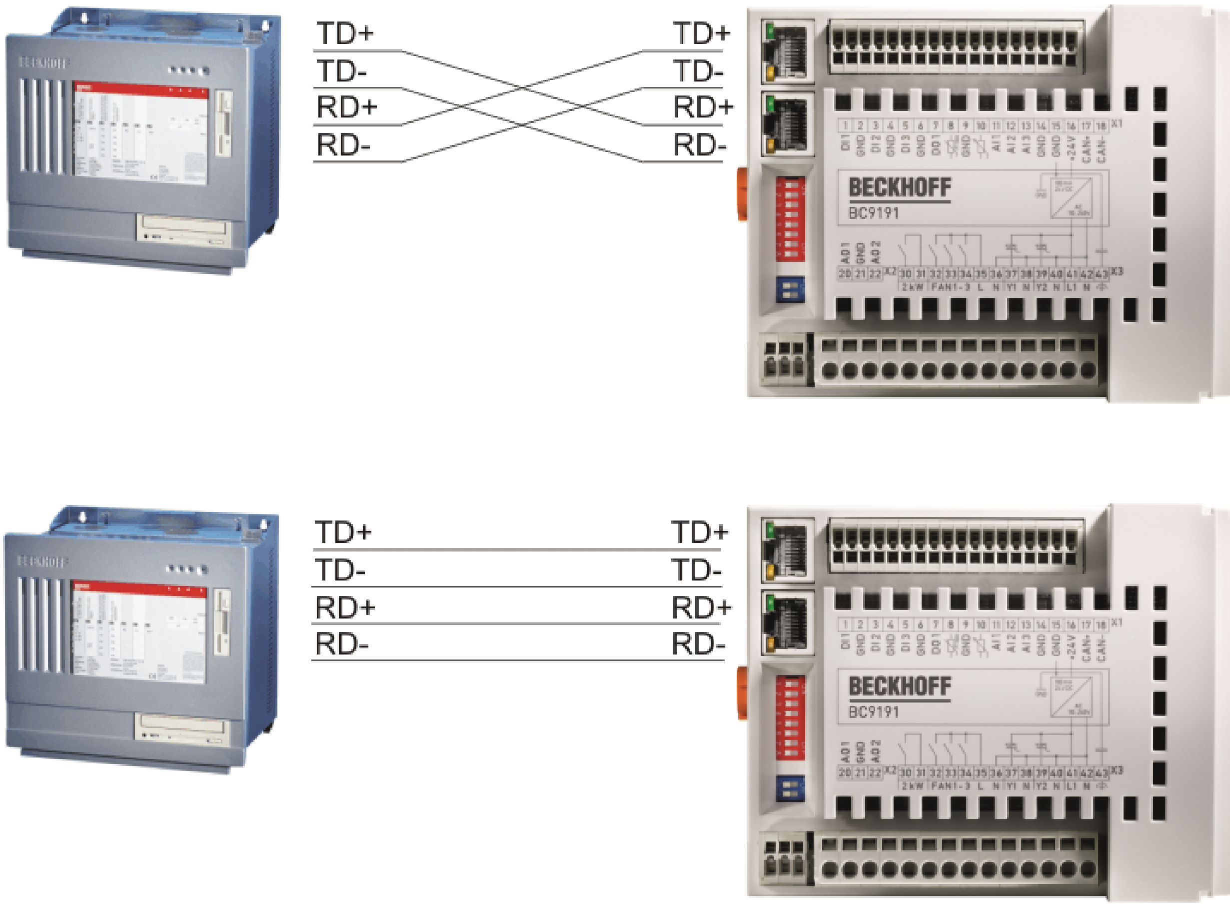


Fig. 18: BC9191 - Auto-crossing

3.3.7 Ethernet cable

Transmission standards

10Base5

The transmission medium for 10Base5 consists of a thick coaxial cable ("yellow cable") with a max. transmission speed of 10 Mbaud arranged in a line topology with branches (drops) each of which is connected to one network device. Because all the devices are in this case connected to a common transmission medium, it is inevitable that collisions occur often in 10Base5.

10Base2

10Base2 (Cheaper net) is a further development of 10Base5, and has the advantage that the coaxial cable is cheaper and, being more flexible, is easier to lay. It is possible for several devices to be connected to one 10Base2 cable. It is frequent for branches from a 10Base5 backbone to be implemented in 10Base2.

10BaseT

Describes a twisted pair cable for 10 Mbaud. The network here is constructed as a star. It is no longer the case that every device is attached to the same medium. This means that a broken cable no longer results in failure of the entire network. The use of switches as star couplers enables collisions to be reduced. Using full-duplex connections they can even be entirely avoided.

100BaseT

Twisted pair cable for 100 Mbaud. It is necessary to use a higher cable quality and to employ appropriate hubs or switches in order to achieve the higher data rate.

10BaseF

The 10BaseF standard describes several optical fiber versions.

Short description of the 10BaseT and 100BaseT cable types

Twisted-pair copper cable for star topologies, where the distance between two devices may not exceed 100 meters.

UTP

Unshielded twisted pair
This type of cable belongs to category 3, and is not recommended for use in an industrial environment.

S/UTP

Screened/unshielded twisted pair (screened with copper braid)
Has an overall shield of copper braid to reduce influence of external interference. This cable is recommended for use with Bus Couplers.

FTP

Foiled shielded twisted pair (screened with aluminium foil)
This cable has an outer screen of laminated aluminium and plastic foil.

S/FTP

Screened/foiled-shielded twisted pair (screened with copper braid and aluminium foil)
Has a laminated aluminium screen with a copper braid on top. Such cables can provide up to 70 dB reduction in interference power.

STP

Shielded twisted pair
Describes a cable with an outer screen, without defining the nature of the screen any more closely.

S/STP

Screened/shielded twisted pair (wires are individually screened)
This identification refers to a cable with a screen for each of the two wires as well as an outer shield.

ITP

Industrial Twisted-Pair
The structure is similar to that of S/STP, but, in contrast to S/STP, it has only one pair of conductors.

3.4 Disposal

Products marked with a crossed-out wheeled bin shall not be discarded with the normal waste stream. The device is considered as waste electrical and electronic equipment. The national regulations for the disposal of waste electrical and electronic equipment must be observed.

4 Parameterization and Commissioning

4.1 Start-up behavior

When the BC9191 is switched on it checks its state, configures the K-bus, creates a configuration list based on the connected Bus Terminals and starts its local PLC.

When the BC9191 starts up, the diagnostic LEDs light up and flash.
In an error-free state, only the following green LEDs should light up after approx. 15 seconds:

- PLC LED
- Power LED Vs
- Power LED Vp
- TC/DC LED, only if TwinCAT configuration is active
- K-bus RUN LED, only if the BC 9191 has been extended with Bus Terminals

The position of the LEDs or which diagnostics LED flashes in the event of an error can be found in the chapter Diagnostics LEDs [▶ 136].

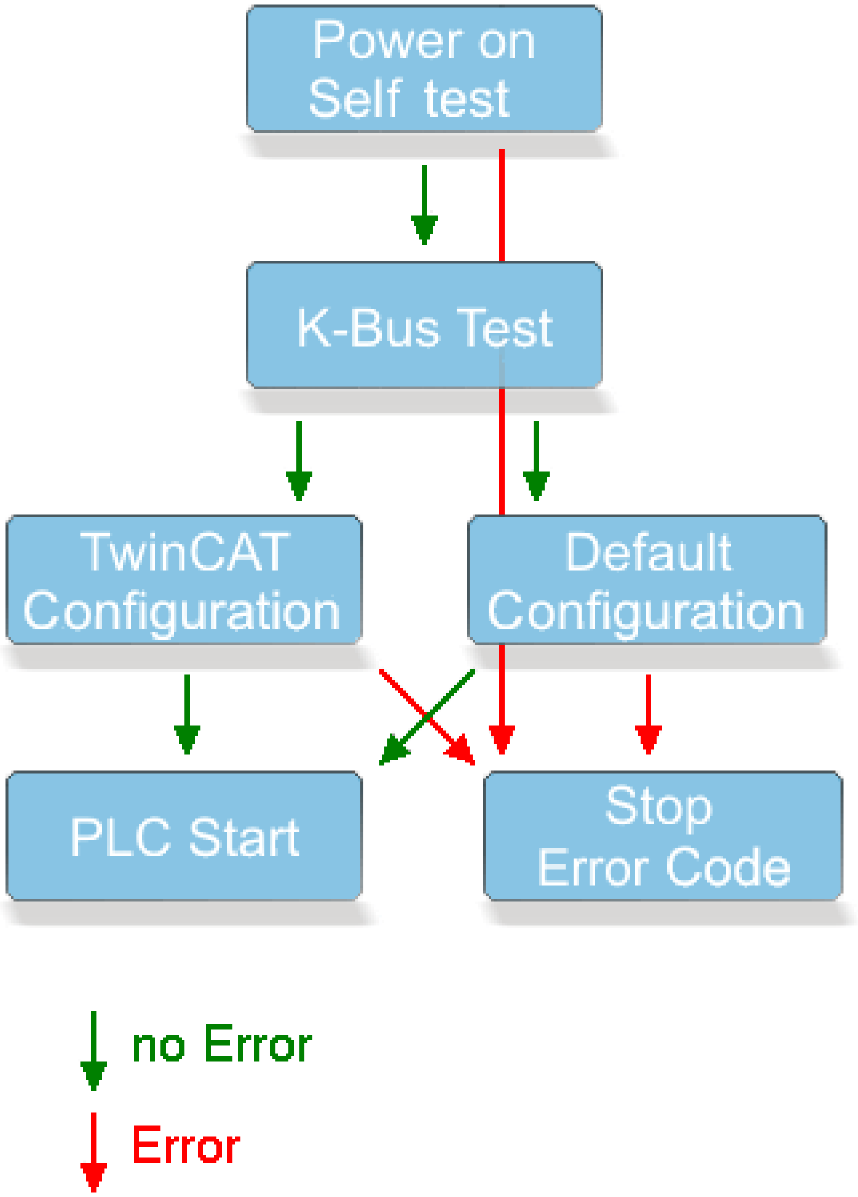


Fig. 19: Start-up behavior of the BC9191

4.2 DIP switch

The BC9191 and BC9191-0100 feature an 8-position and a 2-position DIP switch. These DIP switches have different functions.

1. 8-position DIP switch (red DIP switch)
    - for setting the IP address
    - for the configuration mode
2. 2-position DIP switch (blue DIP switch)
    - for the addressing mode

Setting the Ethernet IP address by DIP switch

The red 8-position DIP switch sets the address of the last byte of the IP address.

The factory-preset default IP address is set to 172.16.21.xxx, where xxx represents the DIP switch setting.

The last byte of the IP address is set with the 8-position DIP switch on the basis of the binary number system.

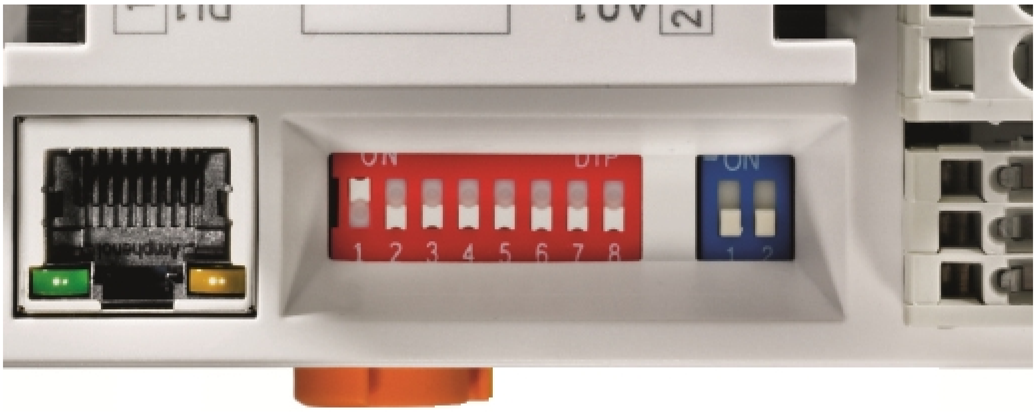


Fig. 20: Example of Ethernet IP address 172.16.21.1

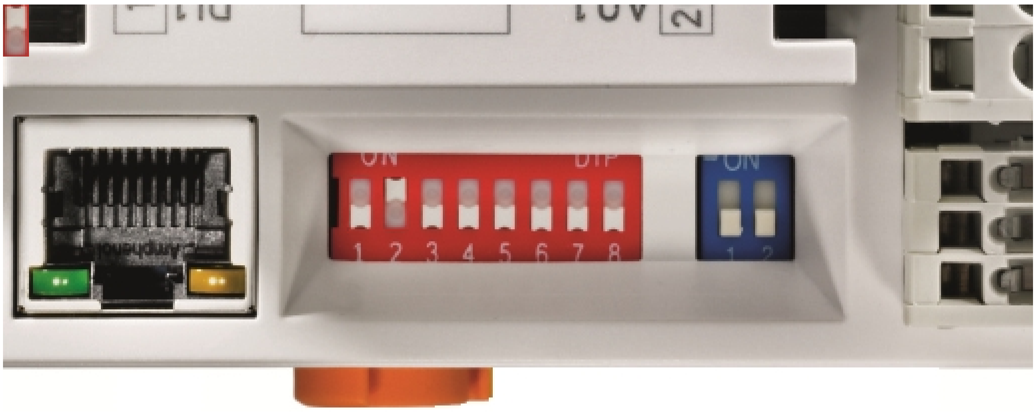


Fig. 21: Example of Ethernet IP address 172.16.21.2

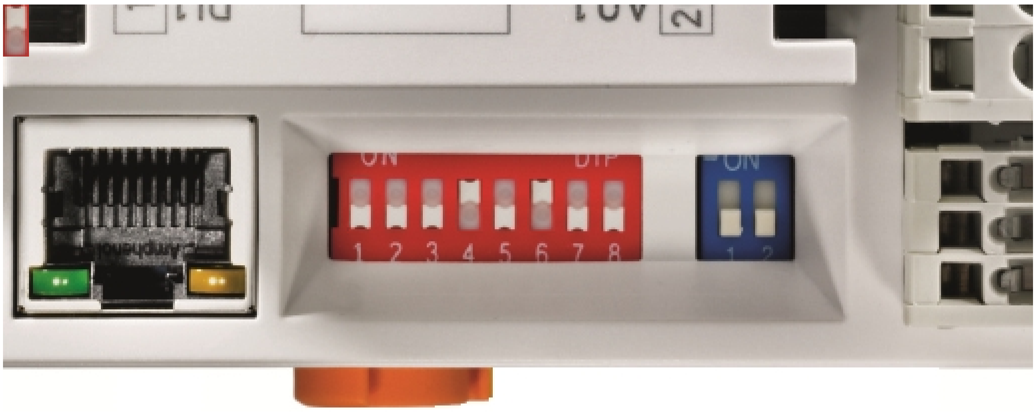


Fig. 22: Example of Ethernet IP address 172.16.21.40

The IP address is applied by restarting the BC9191.

Addressing mode

Blue 2-position DIP switch

Left switch position: off, right switch position: on.

- DIP switches 1-2 select the IP address mode (blue DIP switch in the picture).

DIP1	DIP2	Meaning
off	off	IP address 172.16.21.xxx (xxx corresponds to DIP switch 1-8), subnet mask 255.255.0.0, default gateway 0.0.0.0
on	off	BootP (DIP 1-8 all off), BootP & Safe (DIP 1-8 all on)
off	on	DHCP
on	on	Configuration via the TwinCAT System Manager

Configuration mode, reset to factory settings

The following configurations can be made without configuration software using the DIP switch and the end terminal (KL9010).
This mode is active if only one end terminal (KL9010) is plugged on the BC9191. Otherwise, the normal settings apply.

- Switch off the BC9191, and plug in just the end terminal (KL9010).
- Set the DIP switches 1 to 8 to the desired function in the table.
  The position of the blue dual DIP switch has no influence.

Requirements

Setting	Meaning / function	DIP1	DIP2	DIP3	DIP4	DIP5	DIP6	DIP7	DIP8
255	Restore factory settings	ON	ON	ON	ON	ON	ON	ON	ON
254	Delete boot project	OFF	ON	ON	ON	ON	ON	ON	ON
253	Delete TwinCAT Config	ON	OFF	ON	ON	ON	ON	ON	ON

- Switch the BC9191 on again. Following the successfully executed function the *Error* LED lights up and the *I/O RUN* and *I/O ERR* LEDs flash alternately.
- After that you must switch the BC9191 off and remove the KL9010 again.
- Subsequently you can connect Bus Terminals again if necessary and proceed as usual.

The *restore factory settings* function also deletes the standard room automation program included in the delivery state of the BC9191.

4.3 IP address

4.3.1 IP address - overview

The IP address can be set using four different procedures, and these will be described in more detail below.

Procedure	Explanation	DIP switch 1/2 (blue)	Necessary components
DIP switch	Addressing via DIP switch [▶ 34]	1 = OFF / 2 = OFF	none
KS2000	Addressing via KS2000 [▶ 36]	1 = OFF / 2 = OFF	PC with network and KS2000 configuration software
TwinCAT	Addressing via TwinCAT System Manager [▶ 37]	1 = ON / 2 = ON	PC with network and TwinCAT
BootP	Addressing via BootP server [▶ 37]	1 = ON / 2 = OFF	BootP server
DHCP	Addressing via DHCP server [▶ 39]	1 = OFF / 2 = ON	DHCP server
Local IP address	Local IP address [▶ 39]	1 = ON / 2 = OFF or 1 = OFF / 2 = ON	If no BootP or DHCP server responds or is available

4.3.2 Configuration with KS2000

Using the KS2000 configuration software (as from version 3.2.8) you can set the TCP/IP address via a dialog box or write directly into the registers. DIP switches 1 and 2 in blue should both be OFF (0) before switching on.

In the KS2000 dialog box it is possible to:

- change the name of the controller
- change the first three bytes of the IP address The last byte of the IP address is defined by DIP switches 1 to 8.
- change the settings of subnet mask, default gateway and DNS server.

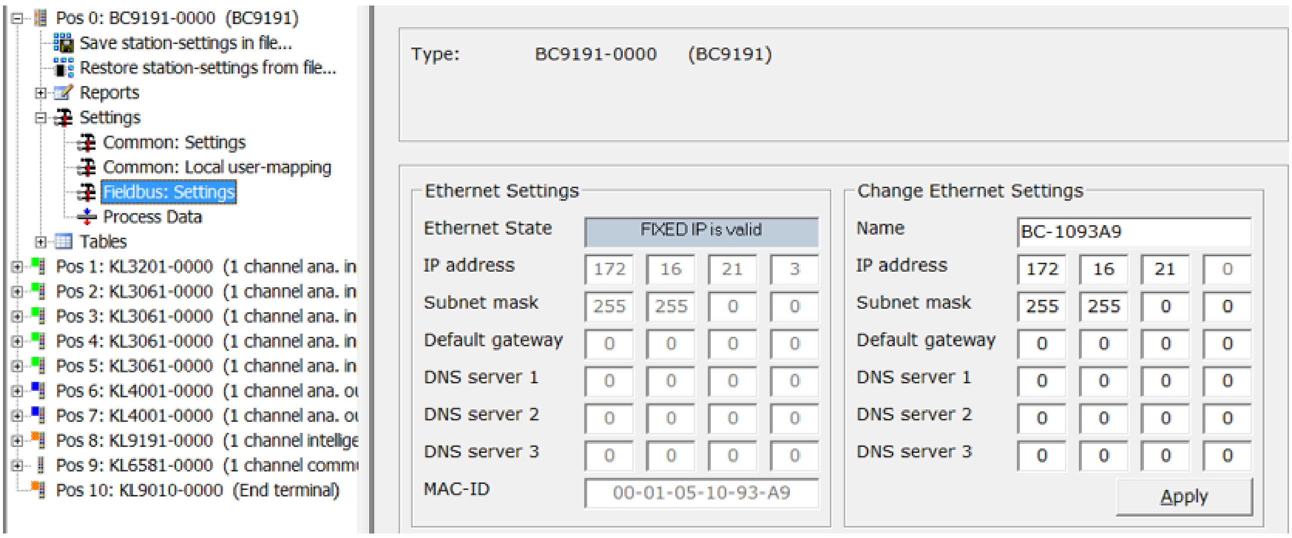


Fig. 23: IP address setting with the KS2000 configuration software

Table 100

Register	High byte	Low byte
0	IP-Byte 2	IP-Byte 1
1	Not used	IP-Byte 3

Default

Byte	Default value (hex)	Default value (dec)
1	0xAC	172_{dec}
2	0x10	16_{dec}
3	0x11	17_{dez}
4	(DIP switch)	(0 to 255_{dec})

4.3.3 Address Configuration via TwinCAT System Manager

An operable ADS connection is necessary in order to set the IP address through the System Manager. This can be done via Ethernet (see chapter Finding the Bus Terminal Controller with the TwinCAT System Manager).

On the BC9191, switches 1 and 2 of the 2-pin blue DIP switch must be set to ON to enable IP addressing via the System Manager.
The settings made via the System Manager are then applied.

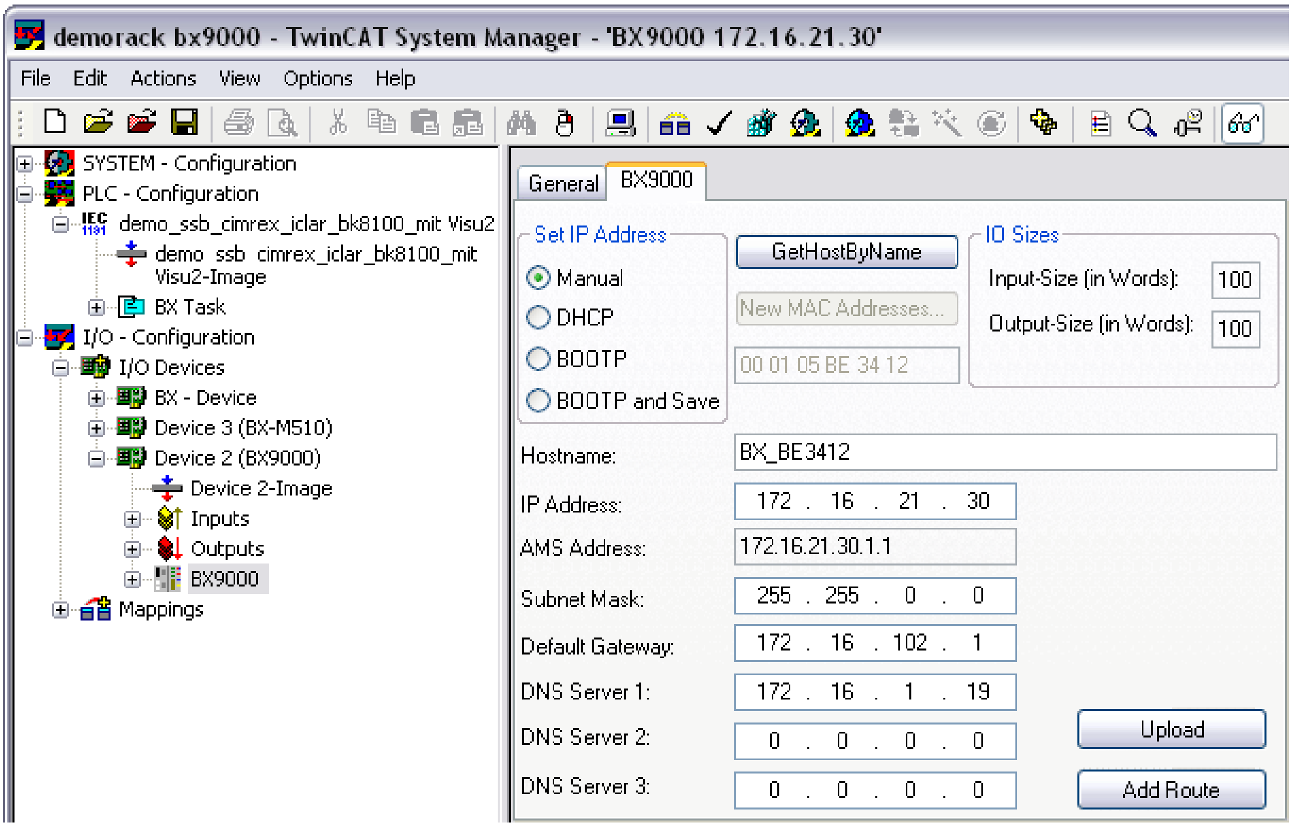


Fig. 24: Address Configuration via TwinCAT System Manager

You can read the current settings with Upload. If the Bus Terminal Controller was configured offline, the *Add Route* button can be used to establish a connection to the Bus Terminal Controller. Edit the required settings, activate the configuration  and restart your Bus Terminal Controller with the green TwinCAT icon (shortcut [Ctrl] [F4]). If the Bus Terminal Controller has been assigned a new IP address, you have to re-enter the new route with the new IP address (see chapter Finding the Bus Terminal Controller with the TwinCAT System Manager).

4.3.4 Setting the address via BootP server

With the BC9191, the assignment of the IP address via a BootP server is activated via the DIP switch.

IP address save modes

BootP & Save

With *BootP & Save*, the IP address issued by the BootP server is stored on the BX, and the BootP service is not queried again at the next cold start.
The address can be cleared again by reactivating the manufacturers' settings (using the KS2000 software or by DIP switch and end terminal [▶ 34]).

BootP

With BootP, the IP address assigned by the BootP server is only valid until the Bus Terminal Controller is switched off. At the next restart, the BootP server has to issue a new IP address for the Bus Terminal Controller.
The address is retained during a software reset of the Bus Terminal Controller.

Beckhoff BootP server

Beckhoff supply a BootP server for Windows 98, ME, NT4.0, NT2000 and XP. You will find the installation version in the *Unsupported Utilities* folder on the Beckhoff *Software Products* CD, or on the internet under: https://download.beckhoff.com/download/software/TwinCAT/TwinCAT2/Unsupported_Utilities/TcBootP_Server/.

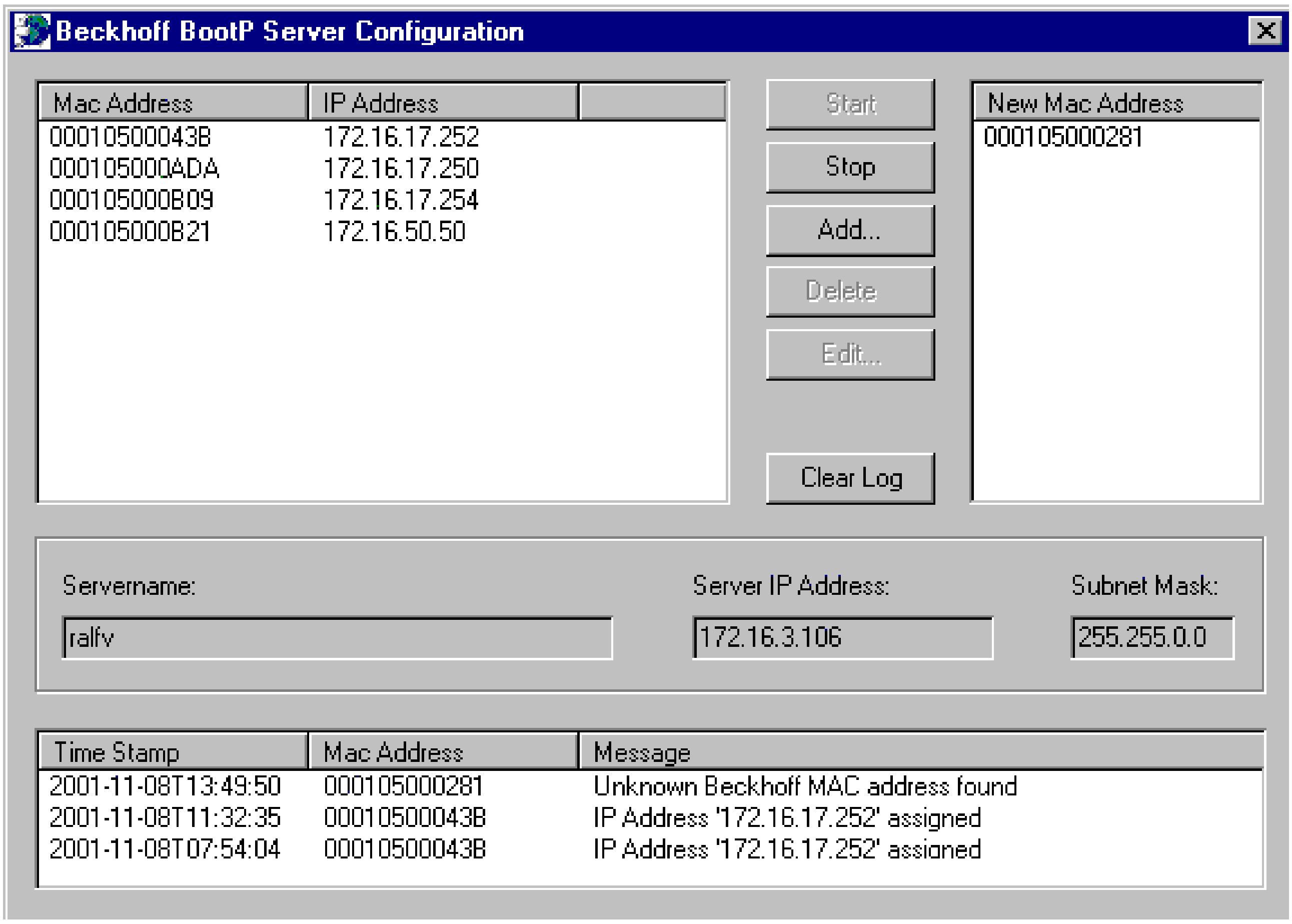


Fig. 25: Configuration of the BootP server

As soon as the BootP server has started, the *New MAC Address* window shows all the Beckhoff nodes that are working in BootP mode and still have not received an IP address. The assignment of the MAC-ID to IP address is made with the [<<] button. Successful assignment is displayed in the log window.
To start the BootP server automatically when your PC boots, it is only necessary to provide a shortcut in the Windows autostart folder. Include the */Start* parameter in the shortcut (.../TcBootPDlg.exe/start).

4.3.5 Setting the address via DHCP server

With the BC9191, the assignment of the IP address via a DHCP server is activated via the dip switch of the Bus Terminal Controller (see chapter DIP switch [▶ 34]).

For the BC9191, switch 1 of the 2-pin blue DIP switch must be set to OFF and switch 2 to ON.

If DHCP is active, the Bus Terminal Controller is automatically assigned an IP number by the DHCP server. The DHCP server must know the MAC ID of the Bus Terminal Controller for this.

The DNS name is formed from the type and the last 3 byte of the MAC ID. The MAC ID is given on the production label of the Bus Terminal Controller.

Example for BC9191

- MAC ID: 00-01-05-01-02-03
- DNS name: BC_010203

4.3.6 Auto IP address

Auto IP address is activated if no DHCP or BootP server is found. This can take several minutes. The Auto IP address is formed as follows:

In the event of a timeout IP address 169.254.[MAC_05].[MAC06] is generated. If MAC_05 is 0 it is set to 1.

3 ARP probes are then sent. If no response is received a gratuitous ARP is sent and the IP is saved.

If the IP address already exists, the values for [MAC_05] and [MAC06] [MAC_04] are added up, and another attempt is made.

4.3.7 Subnet mask

The subnet mask is subject to the control of the network administrator, and specifies the structure of the subnet.

Small networks without a router do not require a subnet mask. The same is true if you do not use registered IP numbers. A subnet mask can be used to subdivide the network with the aid of the mask instead of using a large number of network numbers.

The subnet mask is a 32-bit number:

- Ones in the mask indicate the subnet part of an address space.
- Zeros indicate that part of the address space which is available for the host IDs.

Description	Binary representation	Decimal representation
IP address	10101100.00010000.00010001.11001000	172.16.17.200
Subnet mask	11111111.11111111.00010100.00000000	255.255.20.0
Network ID	10101100.00010000.00010000.00000000	172.16.16.0
Host ID	00000000.00000000.00000001.11001000	0.0.1.200

Standard subnet mask

Address class	Standard subnet mask (decimal)	Standard subnet mask (hex)
A	255.0.0.0	FF.00.00.00
B	255.255.0.0	FF.FF.00.00
C	255.255.255.0	FF.FF.FF.00

Subnets and host number

Neither subnet 0 nor the subnet consisting only of ones may be used. Neither host number 0 nor the host number consisting only of ones may be used!
If the IP address is set using the KS2000 configuration software, it is necessary for the subnet mask also to be changed with the KS2000 configuration software.
Under BootP or DHCP the subnet mask is transmitted also by the server.

4.3.8 Testing the IP address

To test the IP address you can use the *Ping* command in a Windows prompt.

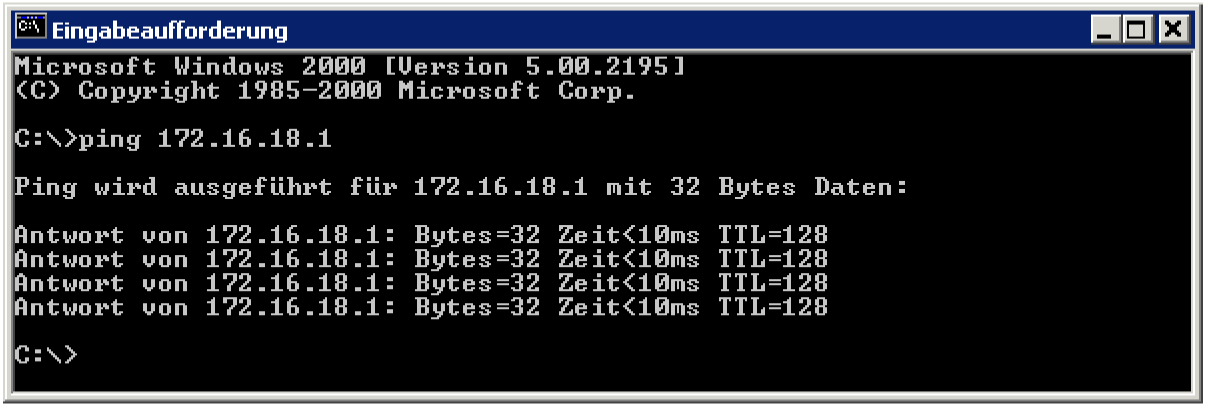


Fig. 26: Testing the IP address using the Ping command

4.3.9 Reading the MAC-ID

Proceed as follows to read the MAC-ID:

- Change the IP address of your PC to 172.16.x.x. and SubNetMask to 255.255.0.0
  In the delivery condition of the BC9191, the IP address is 172.16.21.255 if DIP switches 1 to 8 are set to ON.
- Start the DOS window
- Send a *ping >ip-address<* to the IP address 172.16.21.255
- Read the MAC-ID with *arp -a*.

4.3.10 Security settings

The security setting causes the Bus Terminal Controller to accept and process TCP/IP or UPD/IP telegrams only from certain TCP/IP devices. If a TCP/IP device is not included in the table, the Bus Terminal controller refuses connections with this device. UDP telegrams from devices that are not entered in the table are rejected. In delivery state the table is empty, i.e. all devices have access to the Bus Coupler.

KS2000 dialog

From KS2000 version 4.3.0.39 security table entries can be made via dialog.

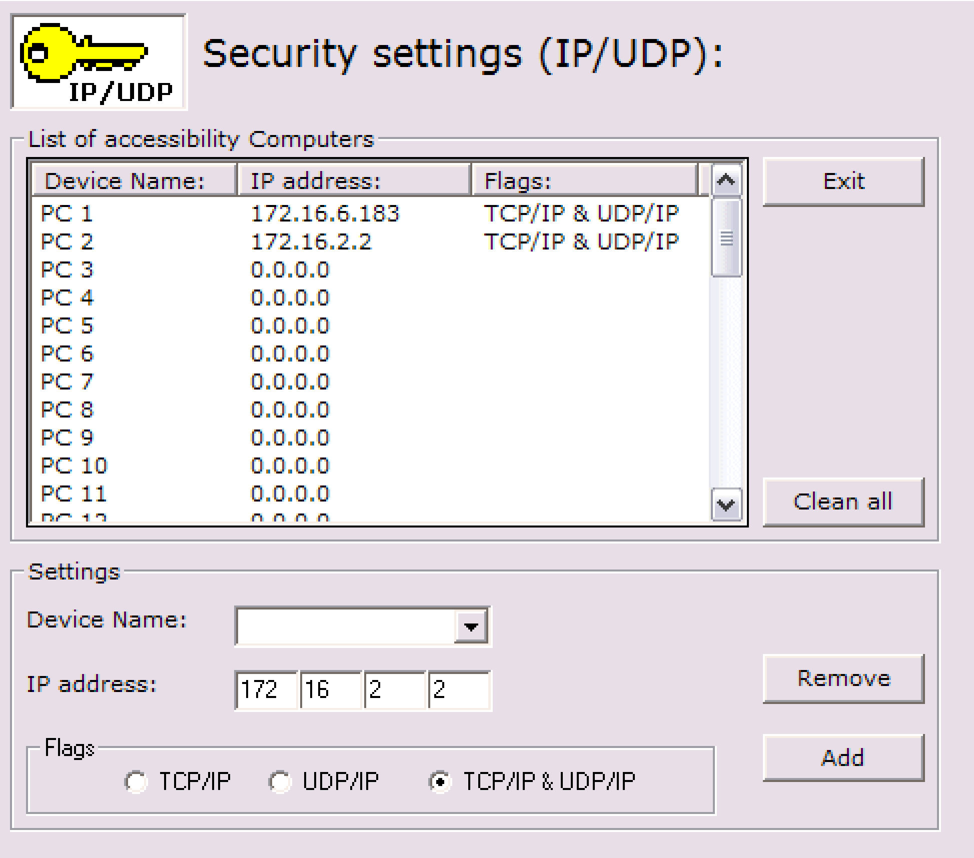


Fig. 27: Security settings

4.4 BC configuration

4.4.1 Overview

Configuration types

The BC9191 Bus Terminal Controllers can be configured in two different ways: DEFAULT CONFIG or TwinCAT CONFIG.

DEFAULT-CONFIG

Fixed addressing BC9191 in the DEFAULT CONFIG:

input/output	Designation	Address	Data type
Digital_Input_1	DI 1	IX 30.0	Bool
Digital_Input_12	DI 2	IX 30.1	Bool
Digital_Input_3	DI 3	IX 30.2	Bool
Analog_PT_1000	PT 1000	IW 2	Integer
Analog_Setpoint	Setpoint	IW 6	Integer
Analog_Input_1	AI 1	IW 10	Integer
Analog_Input_2	AI 2	IW 14	Integer
Analog_Input_3	AI 3	IW 18	Integer
Analog_Output_1	AO 1	QW22	Integer
Analog_Output_2	AO 2	QW26	Integer
Digital_LED_Output	DO 1	QX 30.0	Bool
Relay_Output_2KW	2 kW2	QX 30.1	Bool
Relay_FAN_1	FAN 1	QX 30.2	Bool
Relay_FAN_2	FAN 2	QX 30.3	Bool
Relay_FAN_3	FAN 3	QX 30.4	Bool
TRIAC_Y1_Output	Y1	QX 30.5	Bool
TRIAC_Y2_Output	Y2	QX 30.6	Bool
KL6581.InData	CAN	IB32 … IB43	12 bytes
KL6581.OutData	CAN	QB32 … QB43	12 bytes
Further connected terminals start with the addresses IB44 or Qb44			

Fixed addressing BC9191-0100 in the DEFAULT-CONFIG:

input/output	Designation	Address	Data type
Digital_Input_1	DI 1	IX 30.0	Bool
Digital_Input_12	DI 2	IX 30.1	Bool
Digital_Input_3	DI 3	IX 30.2	Bool
Analog_PT_1000	PT 1000	IW 2	Integer
Analog_Setpoint	Setpoint	IW 6	Integer
Analog_Input_1	AI 1	IW 10	Integer
Analog_Input_2	AI 2	IW 14	Integer
Analog_Input_3	AI 3	IW 18	Integer
Analog_Output_1	AO 1	QW22	Integer
Analog_Output_2	AO 2	QW26	Integer
Digital_LED_Output	DO 1	QX 30.0	Bool
Relay_Output_2KW	2 kW2	QX 30.1	Bool
Relay_FAN_1	FAN 1	QX 30.2	Bool
Relay_FAN_2	FAN 2	QX 30.3	Bool
Relay_FAN_3	FAN 3	QX 30.4	Bool
TRIAC_Y1_Output	Y1	QX 30.5	Bool
TRIAC_Y2_Output	Y2	QX 30.6	Bool
KL6041.InData	RS 485	IB32 … IB55	23 bytes
KL6041.OutData	RS 485	QB32 … QB55	23 bytes
Further connected terminals start with the addresses IB 56 or QB 56			

TWINCAT-CONFIG

In the TWINCAT CONFIG the integrated inputs/outputs and connected Bus Terminals can be freely linked (TwinCAT System Manager file required). The configuration is transferred to the BC9191 with the System Manager via ADS.

For the TwinCAT Config (TC file) you need a PC with TwinCAT 2.10 build 1322 or higher.

OThe System Manager of the TwinCAT 2 program can be used to parameterize the following properties:

- Variable I/O mapping
- PLC settings
- K-bus settings

The configuration can be transferred to the BCx9191 via ADS protocol.

The TwinCAT configuration can be used to link variables, I/Os and data.

In addition, the TwinCAT configuration can be used to parameterize special behavior, for example whether data are preserved or set to "0" in the event of an FK-Bus error.
The internal clock of the BC9191 can be set via a tab in the System Manager.

4.4.2 Finding the Bus Terminal Controller with the TwinCAT System Manager

An operable ADS connection is necessary in order to set the IP address through the System Manager. With the BC9191 and BC9191-0100, this can only be done via Ethernet.

A functioning Ethernet connection is necessary for the ADS connection via Ethernet. You can test the IP connection with the PING command. By default the Bus Terminal Controller is set to 172.16.21.xxx with the subnet mask 255.255.0.0. Set your PC to the same network class, for instance 172.16.200.100 (sub-net mask 255.255.0.0).

Now use PING to test whether a connection exists:

Now start the TwinCAT System Manager and look for the Bus Terminal Controller using the button highlighted in the image or press F8:

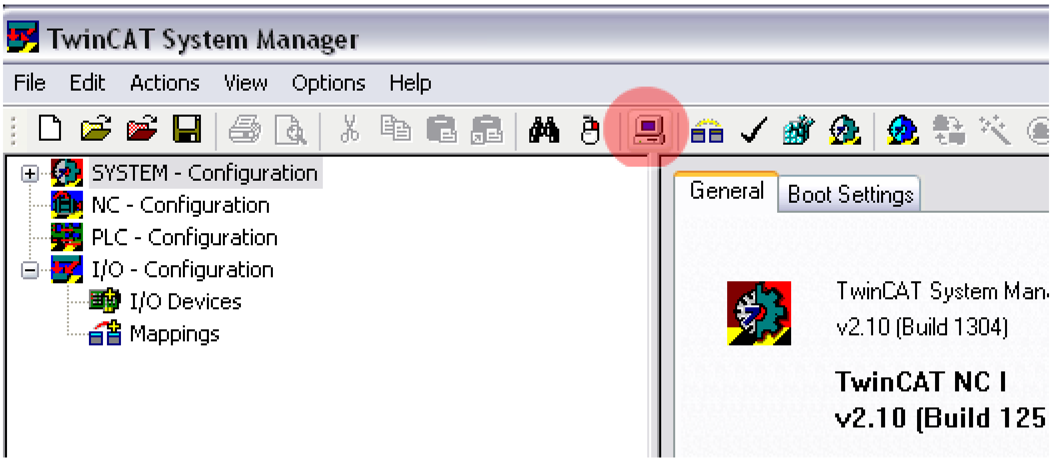


Fig. 28: Finding the Bus Terminal Controller

Now look for the Bus Terminal Controller via Ethernet:

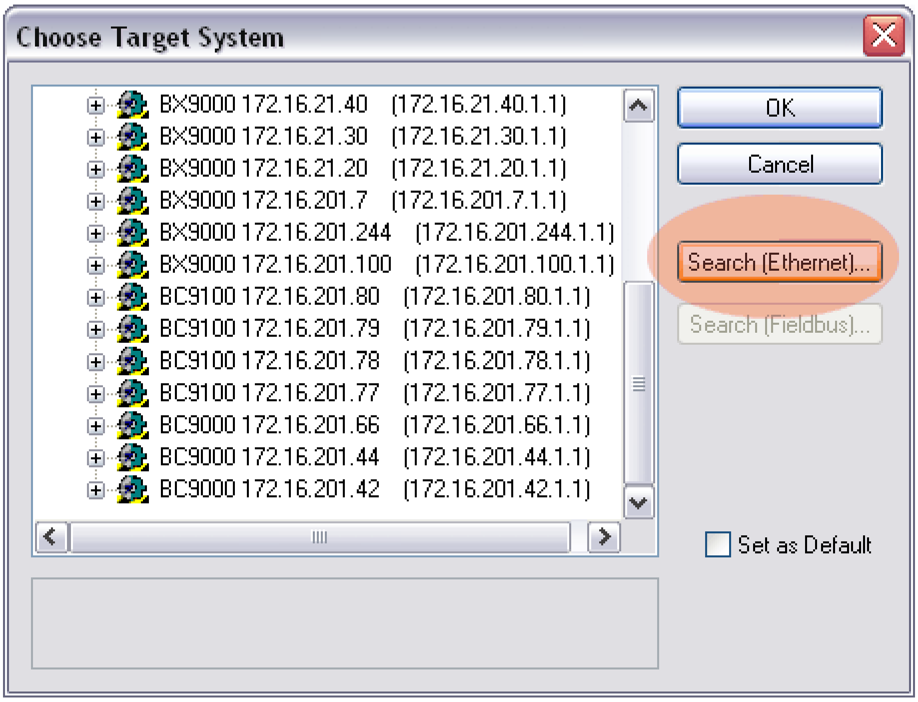


Fig. 29: Choose Target System

Now find your Bus Terminal Controller via Broadcast Search. If you have several Bus Terminal Controllers in your network, you can distinguish them by means of the name.
The name is composed of "BX_" or "BC_" and the last three bytes of the MAC ID. The MAC ID can be found on the underside of the housing of the BC9191.
Example: MAC-ID: 00-01-05-00-1D-C3, then the default name is BC_001DC3.

If the Broadcast Search fails to find a device, check the Ethernet connection.

If the Bus Terminal Controller was addressed via DHCP, you can include the Bus Terminal Controller in your connection via the button *Add Route* in the *Host Name* dialog.

If the Bus Terminal Controller was addressed manually or via BootP, select the assigned *IP address* and establish the connection via the button *Add Route*.

Acknowledge the password query dialog without making an entry. No password is required for Bus Terminal Controllers. Bus Terminal controllers do not offer password support.

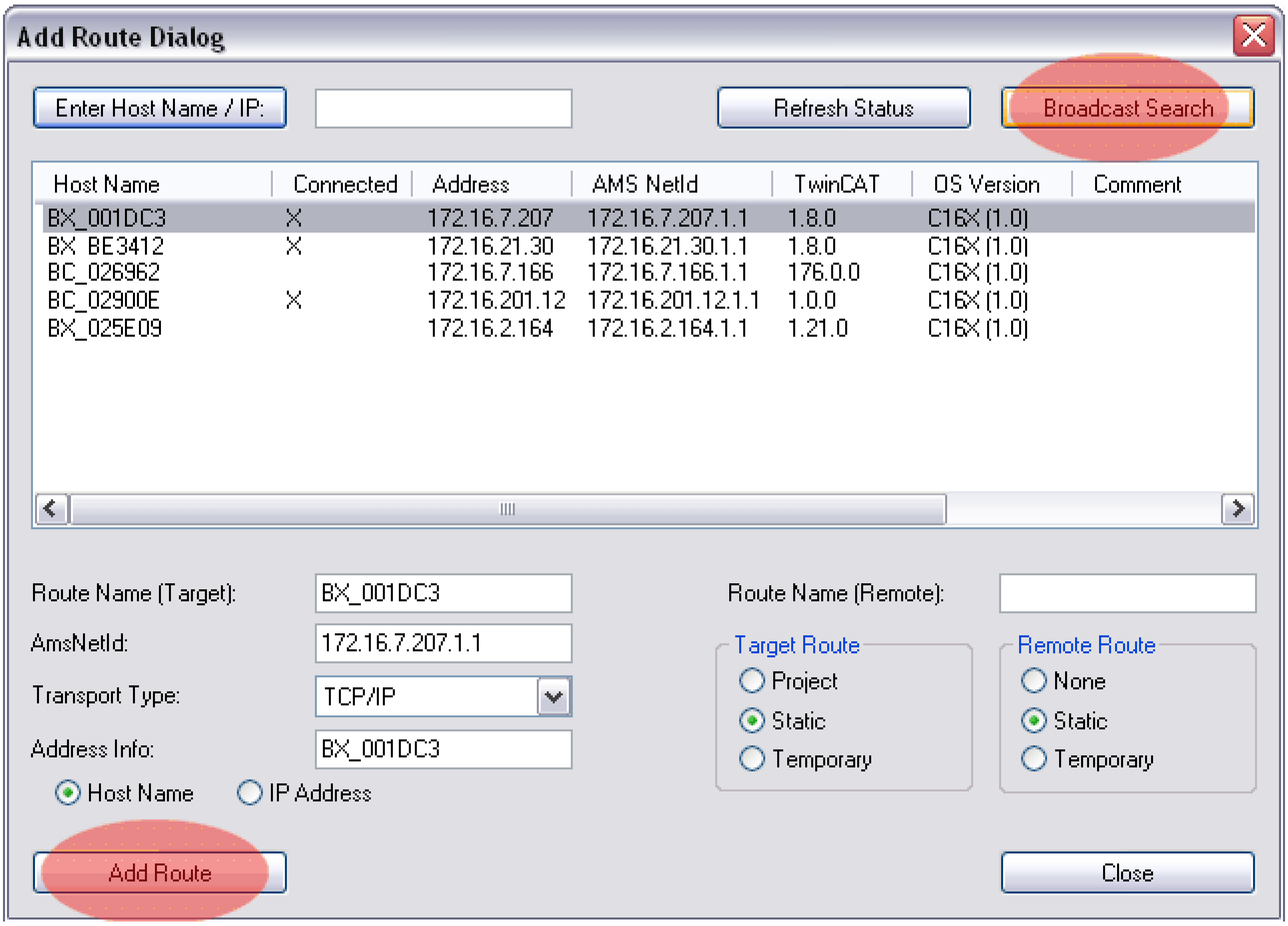


Fig. 30: Add Route Dialog

Now select the Bus Terminal Controller you want to connect to and scan the devices connected to it. The Bus Terminal Controller must be in Config mode (Shift-F4).

4.4.3 Creating a TwinCAT configuration

In order to configure a Bus Terminal Controller of the BCxx50, BCxx20, BXxx00 or BC9191 series, create a BX file in the System Manager. To simplify matters, files for the basic units have already been prepared. Open the corresponding Bus Terminal Controller with *New from Template*.

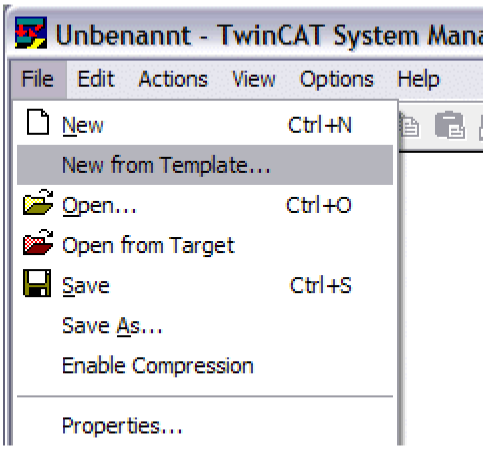


Fig. 31: Creating a TwinCAT configuration

Select the corresponding Bus Terminal Controller.

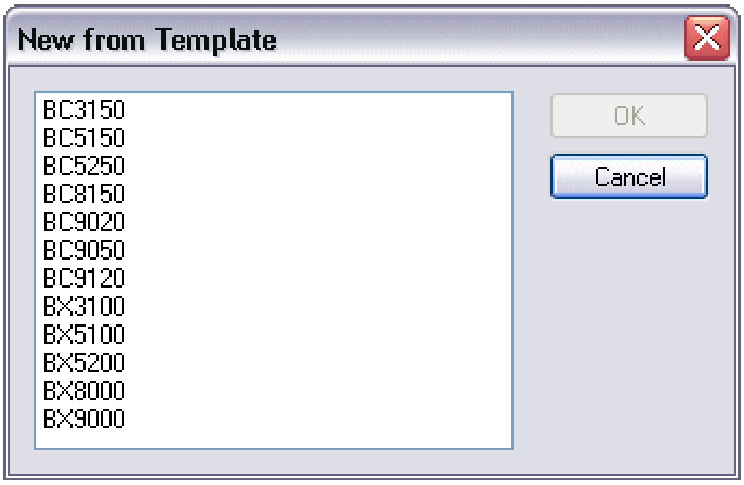


Fig. 32: Selecting the Bus Terminal Controller

All Bus Terminal Controller components are now available:

- Fieldbus interface
- K-bus interface
- PLC program [▸ 57]
- SSB (only Bus Terminal Controllers of the BX series)

Please refer to the relevant chapter for device configuration.

4.4.4 Downloading a TwinCAT configuration

The TwinCAT configuration is loaded into the Bus Terminal Controller via ADS protocol.

Serial ADS protocol

(all Bus Terminal Controllers of the BXxx00 and BCxx50 series)

Enter the serial ADS connection, as described in the chapter Serial ADS .

ADS protocol via the fieldbus

(BC3150, BC5150, BC9x20, BC9050, BX3100, BX5100, BX9000, BC9191 only)

A prerequisite is that TwinCAT operates as master and is engaged in data exchange, i.e. the physical and fieldbus configuration must be complete, and data exchange must take place between the master (e.g. fieldbus master card) and the Bus Terminal Controller.

Choose Target System

Select the Bus Terminal Controller onto which the configuration is to be loaded. Use the function key F8 to open the dialog for downloading your file to the corresponding device.

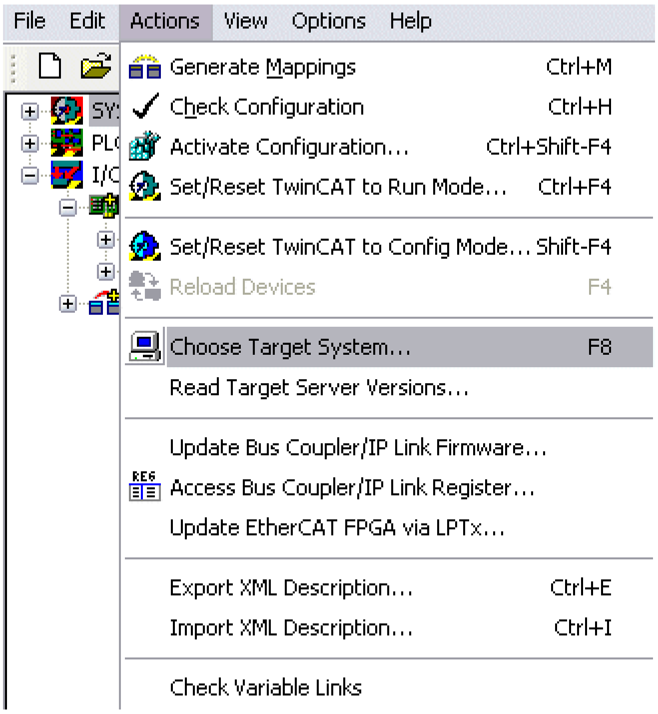


Fig. 33: Downloading a TwinCAT configuration

Select the corresponding Bus Terminal Controller.

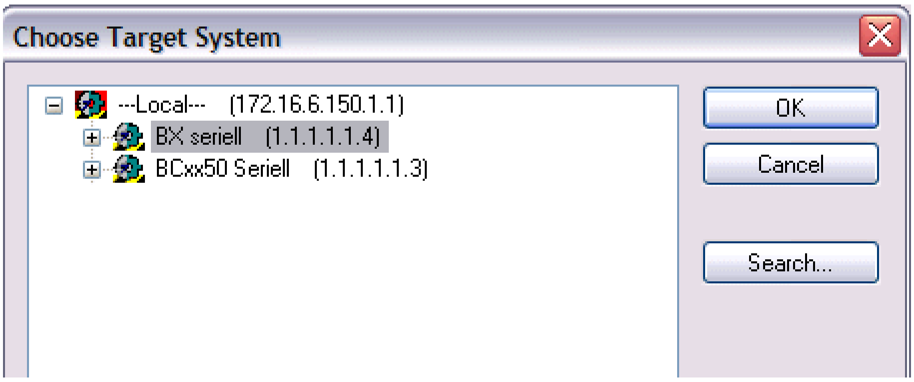


Fig. 34: Selecting the Bus Terminal Controller

The state of the Bus Terminal Controller is shown at the bottom right of the System Manager.

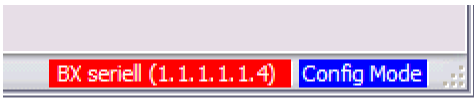


Fig. 35: State of the Bus Terminal Controller

In *Config mode / FreeRun* the configuration can now be downloaded to Bus Terminal Controller. If the Bus Terminal Controller is in *Stop mode*, ADS communication is not yet activated. In this case, it is not possible to download the configuration.

To activate the TwinCAT configuration select Ctrl+Shift+F4 or *Activate Configuration*.

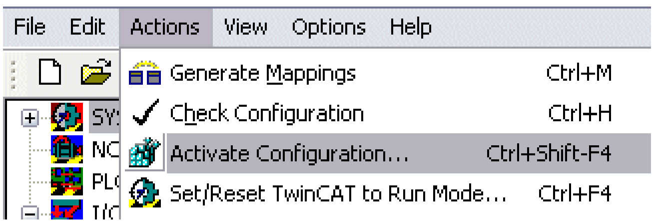


Fig. 36: Activating the TwinCAT configuration

The current configuration is loaded onto the Bus Terminal Controller. The display will show *Store Config*, and the BUS and I/O LED will flash. Once the configuration is successfully loaded onto Bus Terminal Controller, *TwinCAT Config* should appear in the display of a BXxx00. The corresponding program can now be transferred to the Bus Terminal Controller (program-download via the fieldbus) [▸ 73].

4.4.5 Uploading a TwinCAT configuration

The TwinCAT configuration is loaded into the Bus Terminal Controller via ADS protocol.

Serial ADS protocol

(all Bus Terminal Controllers of the BCxx50, BCxx20 and BXxx00 series)

Enter the serial ADS connection, as described in the chapter Serial ADS .

ADS protocol via the fieldbus

(BC3150, BC5150, BC9x20, BC9050, BX3100, BX5100, BX9000, BC9191 only)

A prerequisite is that TwinCAT operates as master and is engaged in data exchange, i.e. the physical and fieldbus configuration must be complete, and data exchange must take place between the master (e.g. fieldbus card) and the Bus Terminal Controller.

Choose Target System

Select the Bus Terminal Controller onto which the configuration is to be loaded. Use the function key [F8] to open the dialog for downloading your file to the corresponding device.

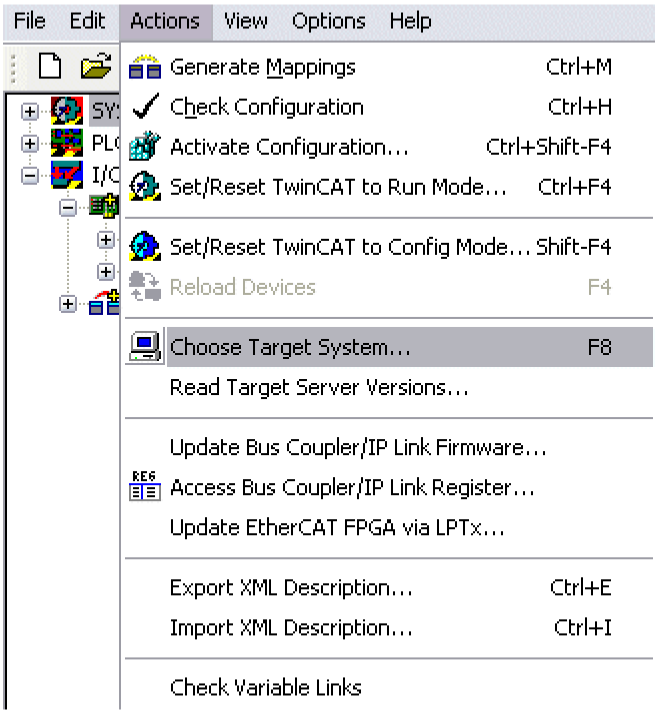


Fig. 37: Choose Target System

Select the corresponding Bus Terminal Controller.

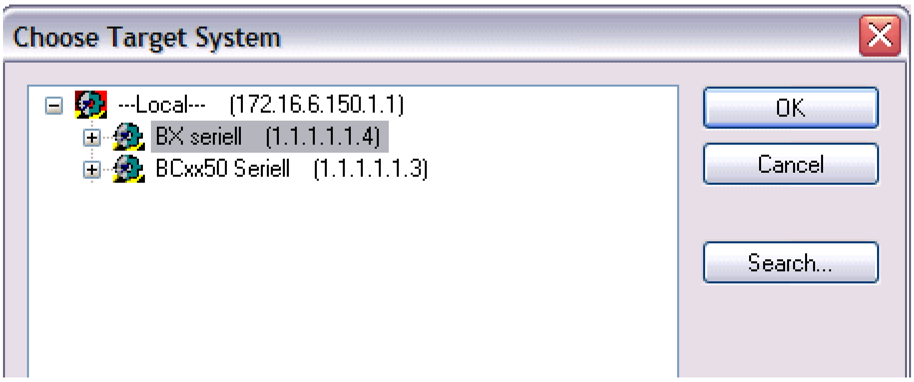


Fig. 38: Selecting the Bus Terminal Controller

The state of the Bus Terminal Controller is shown at the bottom right of the System Manager.

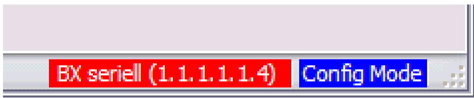


Fig. 39: State of the Bus Terminal Controller

Click on the red folder. The TwinCAT configuration will now be uploaded.

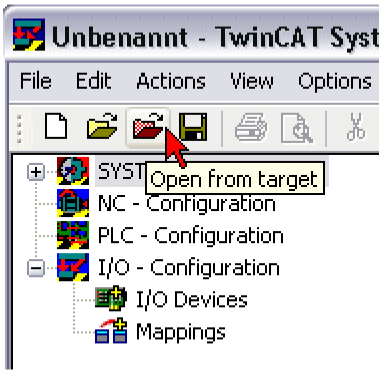


Fig. 40: Uploading the TwinCAT configuration

4.4.6 Resources in the Bus Terminal Controller

The memory resources assigned in the Bus Terminal Controller are shown in the System Manager in the *Resources* tab of the Bus Terminal Controller.

Mapping code

The mapping code is required for calculating the TwinCAT configuration (see Figure *Memory for the code mapping*). The percentages are added here. In the example from Fig. *Memory for code mapping*, 8% of the memory is allocated to the mapping calculation.

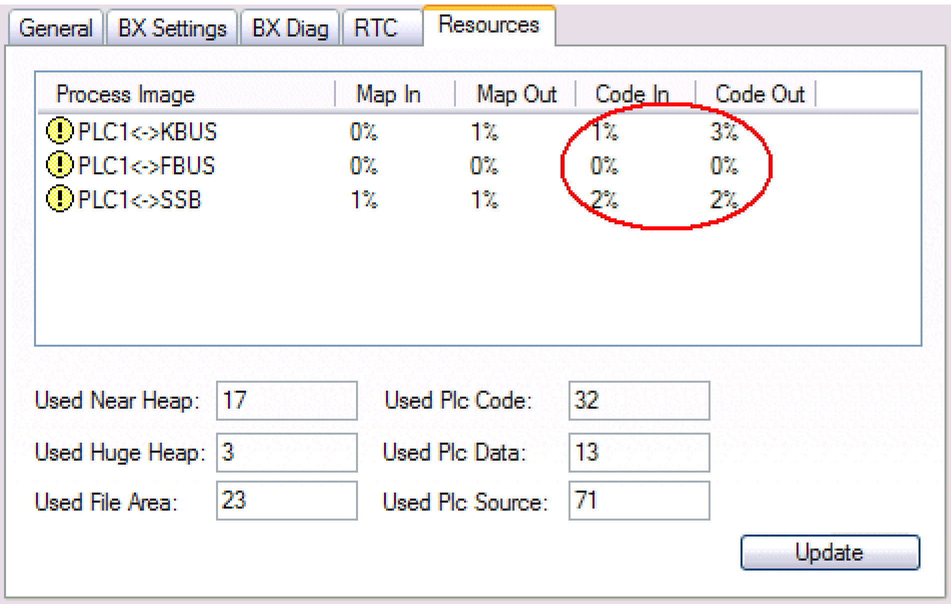


Fig. 41: Memory for code mapping

Data memory mapping

Data memory for mapping. The values are to be considered individually, i.e. each value can be up to 100%.

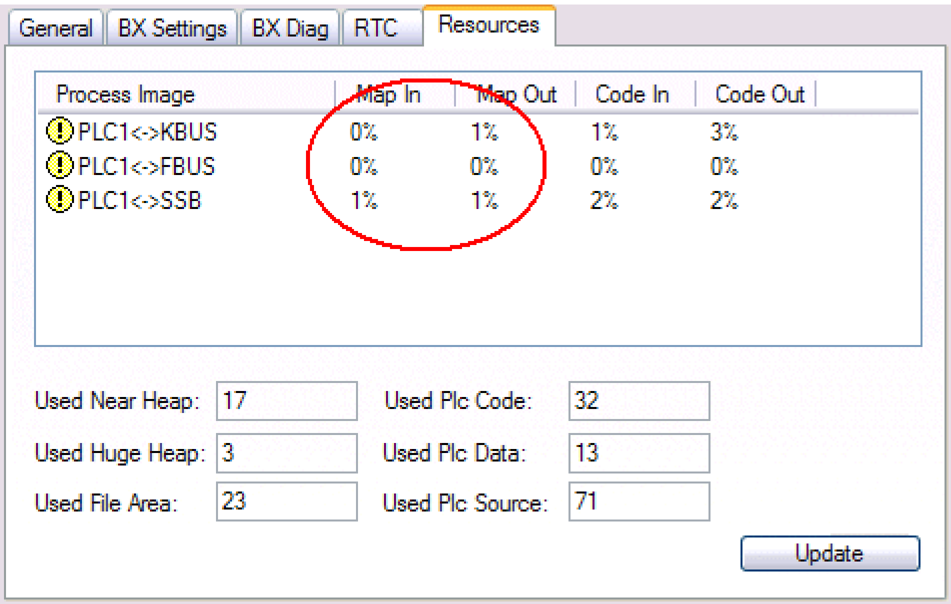


Fig. 42: Data memory mapping

Used code and data memory

Fig. *Code and data memory* (1) "Used PLC code" in %.
Fig. *Code and data memory* (2) "Used PLC data" in %.
Fig. *Code and data memory* (3) "Used PLC source" in %.

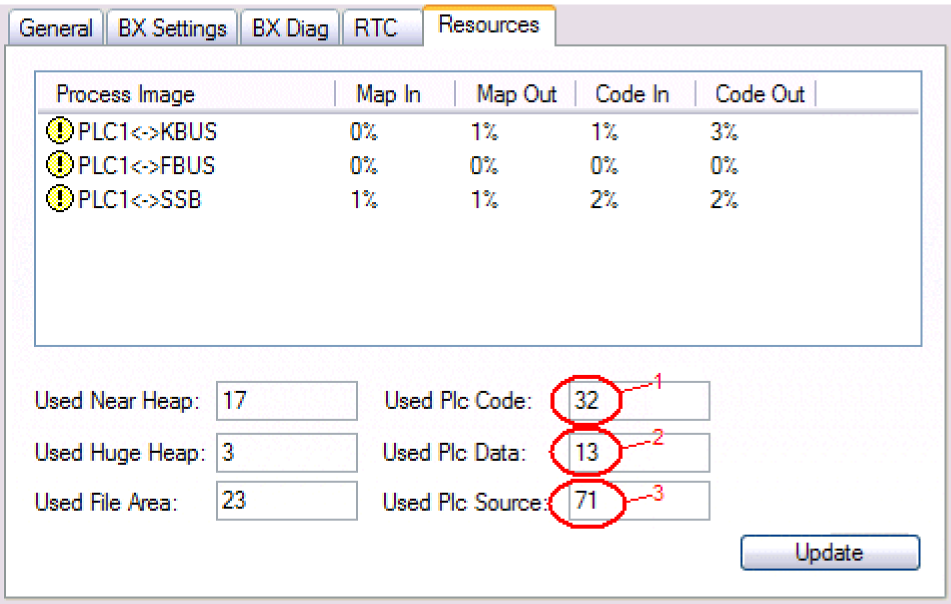


Fig. 43: Code and data memory

Other memory

Fig. *Other Memory* (1) "Used Near Heap" is required for the COM interface and SSB. % values.
Fig. *Other Memory* (2) "Used Huge Heap" is required for the ADS communication. % values. This value should be less than 30 %.
Fig. *Other Memory* (3) "Used File Area" is required for the TwinCAT configuration, the TSM file and the 16 kbyte flash access. % values.

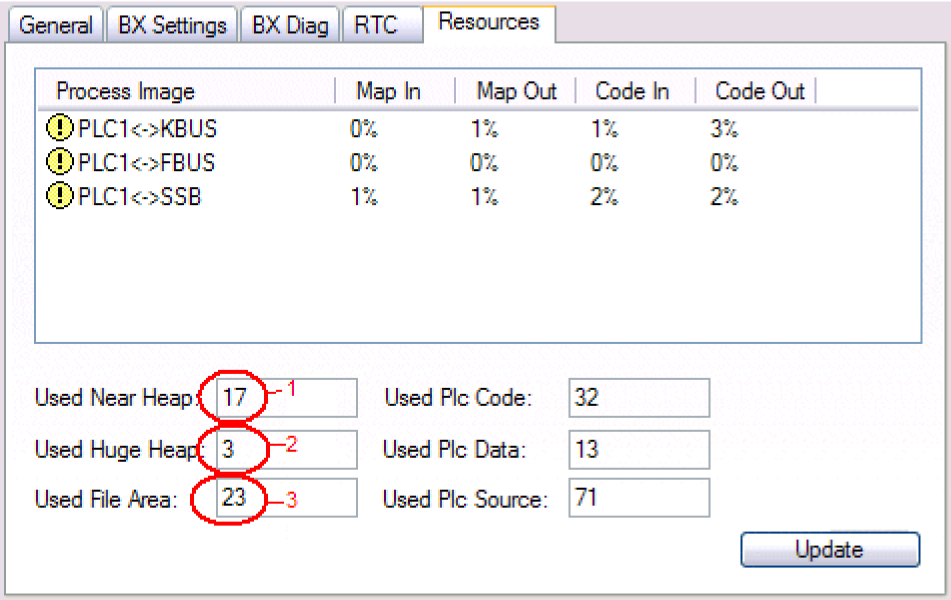


Fig. 44: Other memory

4.4.7 Ethernet

4.4.7.1 TwinCAT as master PLC

A higher-level controller (PC or CX with TwinCAT) can operate as a master PLC (control system) for the Bus Terminal Controller (BC9x20, BC9050, BC9191 or BX9000). It queries the PLC variables depending on the set task time of the Bus Terminal Controller. This enables the higher-level controller to receive data from the Bus Terminal Controller or to send data to it. The following communication options are supported:

- ADS TCP, cyclic or acyclic from the PLC using the ADS READ and WRITE function blocks
- ADS UDP, cyclic
- ModbusTCP (with TC Modbus client)
- Bus Terminal Controller sends or reads data from the PLC using the ADS READ and WRITE function blocks.

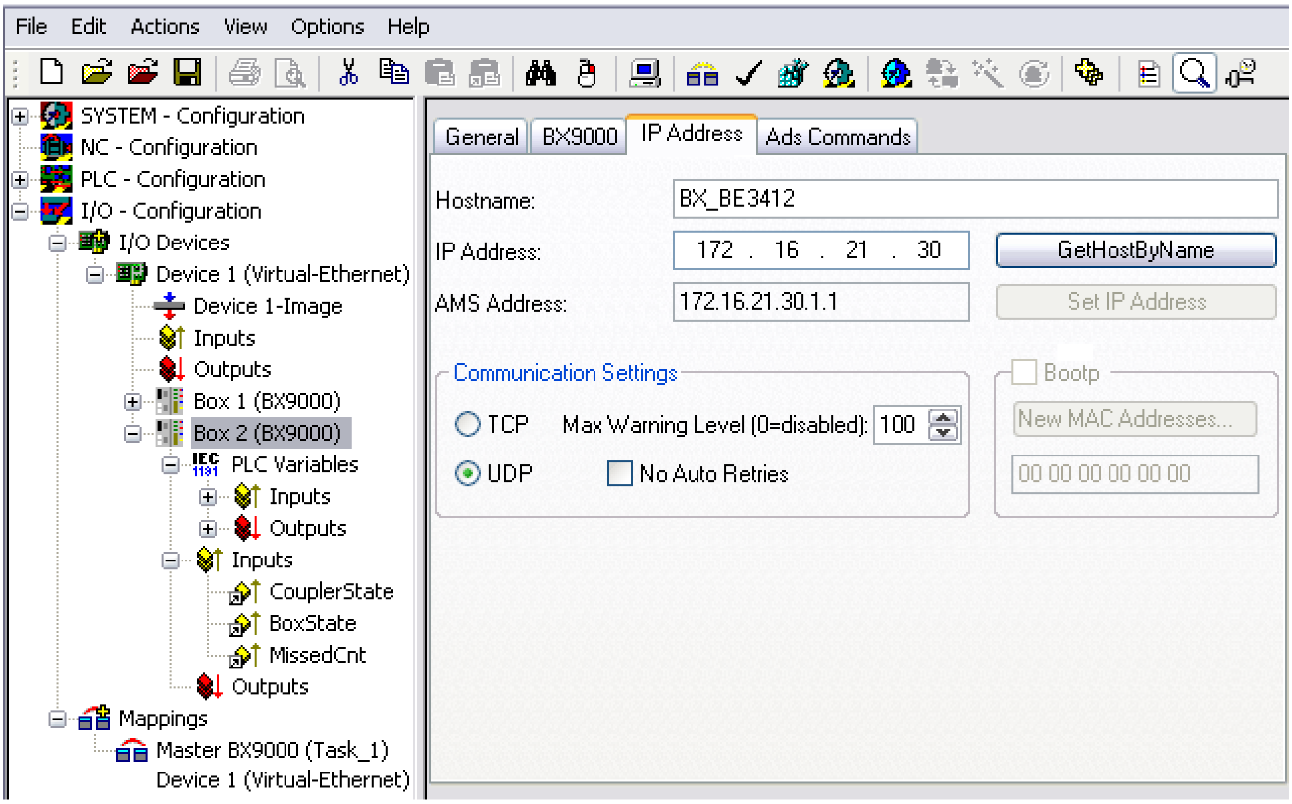


Fig. 45: Communication settings

GetHostByName: This function enables the IP address to be found based on the name (this only works if the IP address of the Bus Terminal Controller was assigned via DHCP)

PLC Variables: Data for the cyclic data connection. This must be linked into at least one task. 256 words of input or output is the maximum. Should more data be required for the transfer, these can be read or written acyclically via the flag area of the Bus Terminal Controller.

Diagnostic Data:

Coupler State: Should always be zero. "1" is set if, for example, the K-bus reports an error
BoxState: see Comment in the dialog
MissedCnt: Should not count up if possible. Because TwinCAT operates in real time, but neither TCP nor UDP are real-time protocols, is not impossible that the counter will increase under certain circumstances. The counter is incremented every time that data that is been transmitted at the beginning of the task has not yet returned by the time the task starts again.

The task time should be set in the following way

For ADS TCP, cyclic

Measure the required PLC time of the Bus Terminal Controller, add 20 to 30% and set the task time of the Bus Terminal Controller accordingly. Now take three times this task time; the result corresponds to the task time on your master controller.

Example:
Measured PLC time: 5 ms
Set a task time of 7 ms for the Bus Terminal Controller and
set a task time of 3 x 7 ms = 21 ms for the Master PLC.

If you operate several Bus Terminal Controllers below the Master PLC, the Bus Terminal Controller with the slowest PLC time sets the task time for the Master PLC.

For ADS UDP, cyclic

Measure the required PLC time of the Bus Terminal Controller, add 20 to 30% and set the task time of the Bus Terminal Controller accordingly. Now take two times this task time; the result corresponds to the task time on your master controller.

Example:
Measured PLC time: 5 ms
Set a task time of 7 ms for the Bus Terminal Controller and
set a task time of 2 x 7 ms = 14 ms for the Master PLC.

If you operate several Bus Terminal Controllers below the Master PLC, the Bus Terminal Controller with the slowest PLC time sets the task time for the Master PLC.

For ModbusTCP, cyclic

Measure the required PLC time of the Bus Terminal Controller, add 20 to 30% and set the task time of the Bus Terminal Controller accordingly. Now take two times this task time; the result corresponds to the task time on your master controller.

Example:
Measured PLC time: 5 ms
Set a task time of 7 ms for the Bus Terminal Controller and
set a task time of 2 x 7 ms = 14 ms for the Master PLC.

If you operate several Bus Terminal Controllers below the Master PLC, the Bus Terminal Controller with the slowest PLC time sets the task time for the Master PLC.

Different PLC cycle times

If the Bus Terminal Controllers in your system require different cycle times for their local PLC processing, you can also adjust the time after which the Master PLC queries each individual Bus Terminal Controller individually.

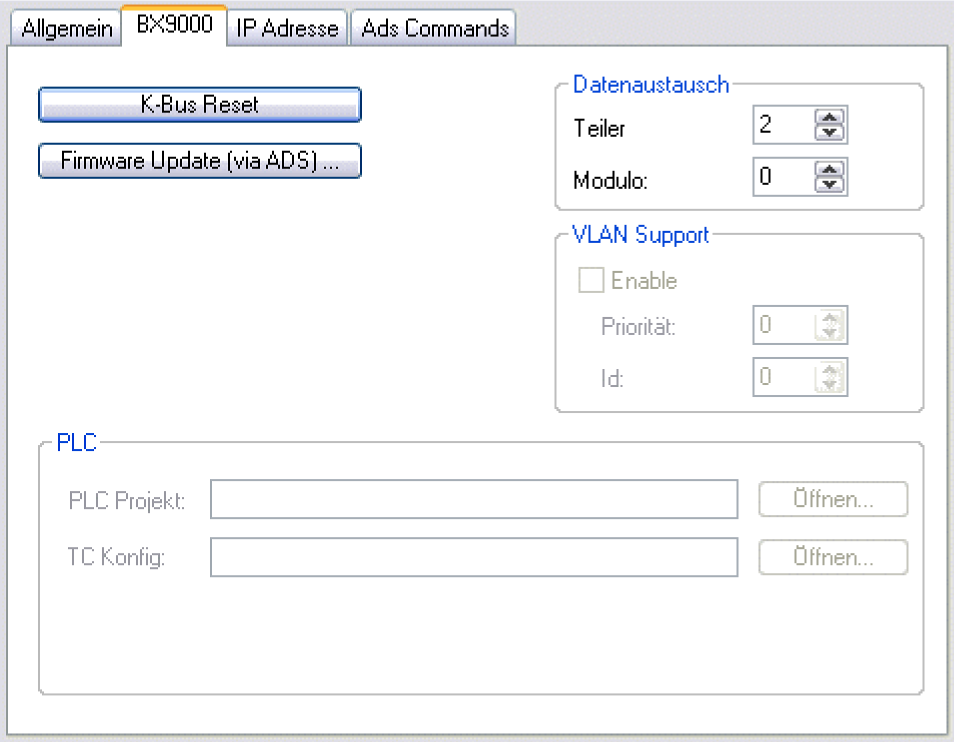


Fig. 46: Individual adjustment of the time after which the Master PLC queries the individual Bus Terminal Controller

Divider

Use the divider for this purpose. The divider takes as its basis the cycle time of the higher-level Master PLC, for example 10 ms. If the divider is now set to 2, a telegram is sent to the Bus Terminal Controller every 2 x 10 ms, i.e. every 20 ms.

Modulo

Modulo can be used to set the timing for the higher-level Master PLC.

Example:
Divider 3, Modulo 0, means that a telegram is sent after the 1st task cycle and then after every 3rd task cycle.
If the Modulo is set to 1, a telegram is only sent after the 2nd task cycle and then after every 3rd task cycle + 1.

In systems with many Ethernet nodes this enables the number of Ethernet packets to be distributed more evenly, which results in more uniform network load and avoids network load peaks.

4.4.8 K-bus

Cover and end terminal

Unlike the operation of a BC or BX series Bus Terminal Controller without integrated I/Os, no Bus Terminal or end terminal has to be plugged on the BC9191!

- BC9191 without further Bus Terminals => only insert K-bus cover
- BC9191 with further Bus Terminals => termination of the K-bus with the KL9010 end terminal

BX Settings tab

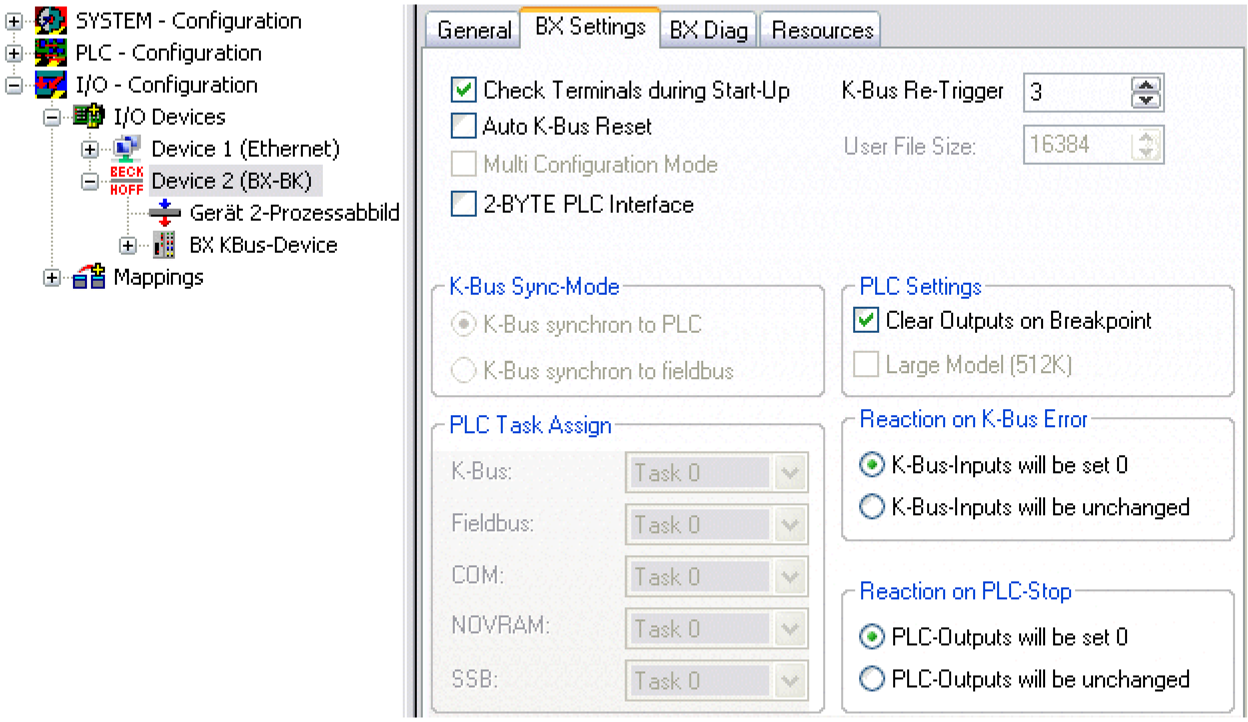


Fig. 47: BX Settings tab

Check Terminals during Start-up

When a boot project is created, the current Bus Terminal configuration is stored. The connected Bus Terminals are checked when the Bus Terminal Controller restarts. If this option is selected, the Bus Terminal Controller does not enter into data exchange. The PLC project will not be started.

Auto K-Bus Reset

Once a K-bus error has been rectified, the Bus Terminal Controller automatically resumes the data exchange.

⚠ CAUTION
Once a K-Bus error has been rectified, the outputs become active again immediately! Ensure that the outputs are reactivated immediately and that analog outputs retain their programmed value, if this is not dealt with in your PLC project.

Clear Outputs on Breakpoint

If breakpoints are set in PLC Control, the K-Bus is no longer processed, i.e. the outputs are set to a safe state (zero).

K-Bus Sync Mode

Writing and reading of the Bus Terminals can take place synchronously with task 1 or the fieldbus.

K-Bus Re-Trigger

If the processor is busy dealing with the PLC project or the SSB, the K-Bus cannot be processed for a certain amount of time. This leads to triggering of the Bus Terminal watchdog and dropping of the outputs. The Bus Terminal Controller is set such that the K-bus watchdog is re-triggered 3 times after 85 ms. The K-Bus watchdog would then be activated.
K-Bus Re-Trigger 0: 100 ms

K-Bus Re-Trigger 1: 2 x 85 ms = 170 ms
K-Bus Re-Trigger 2: 3 x 85 ms = 255 ms
K-Bus Re-Trigger 3: 4 x 85 ms = 340 ms

Reaction on K-Bus Error

In the event of a K-Bus error, the K-Bus inputs are set to "0" or retain their last state.

Response on PLC-Stop

The user can set the behavior of the fieldbus output data in the event of the PLC project being stopped. The master will use these data as input data. In the event of a PLC stop, the data can be set to "0" or remain unchanged.

BX Diag tab

Display of the cycle time for task 1, K-bus, fieldbus processing and the SSB load.

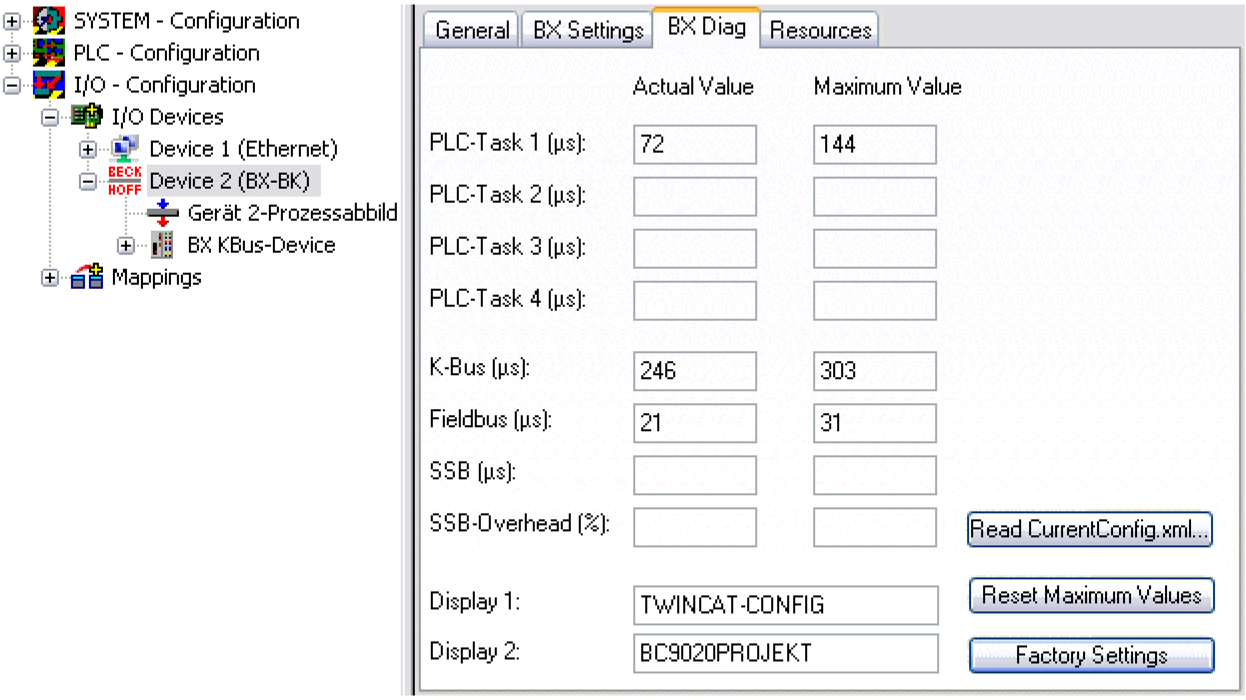
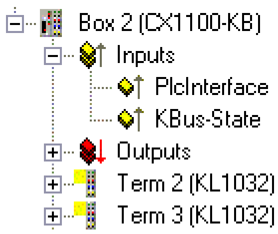


Fig. 48: BX Diag tab

Factory Settings: the Bus Terminal Controller is set to its delivery. These settings are reactivated via Restart System or by switching the system off and on again (display shows DEFAULT-CONFIG).
Reset Maximum Values: resets the maximum values

K-Bus variables



PLC interface: Not supported (only included for moving CX or BX projects)

K-bus state: see Diagnostics

4.4.9 PLC

4.4.9.1 Inserting a PLC project

For variable mapping, configuration has to be specified in the system manager. This is where the link between PLC and hardware is specified. The variables can process and link bit, byte, word or data structures. Automatic addressing via the System Manager is possible, but should be checked for offset.

Word alignment, byte orientation

With data structures, ensure that the Bus Terminal Controller saves the data in word alignment and the System Manager operates byte-oriented (see Data structures [▸ 72])

A valid project has to be compiled and saved in PLC Control. These data are saved as a *.tpy file. For inserting a PLC project, right-click on *PLC - Configuration*. Select your current PLC project.

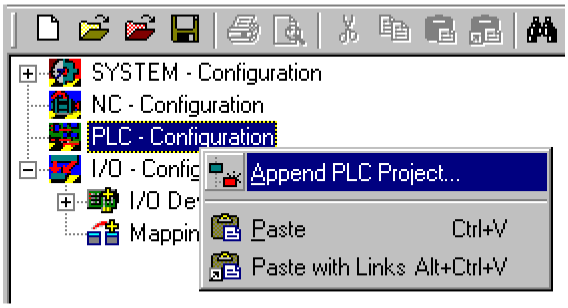


Fig. 49: Selecting the PLC project

Link the PLC variable with the hardware (e.g. digital Bus Terminal).

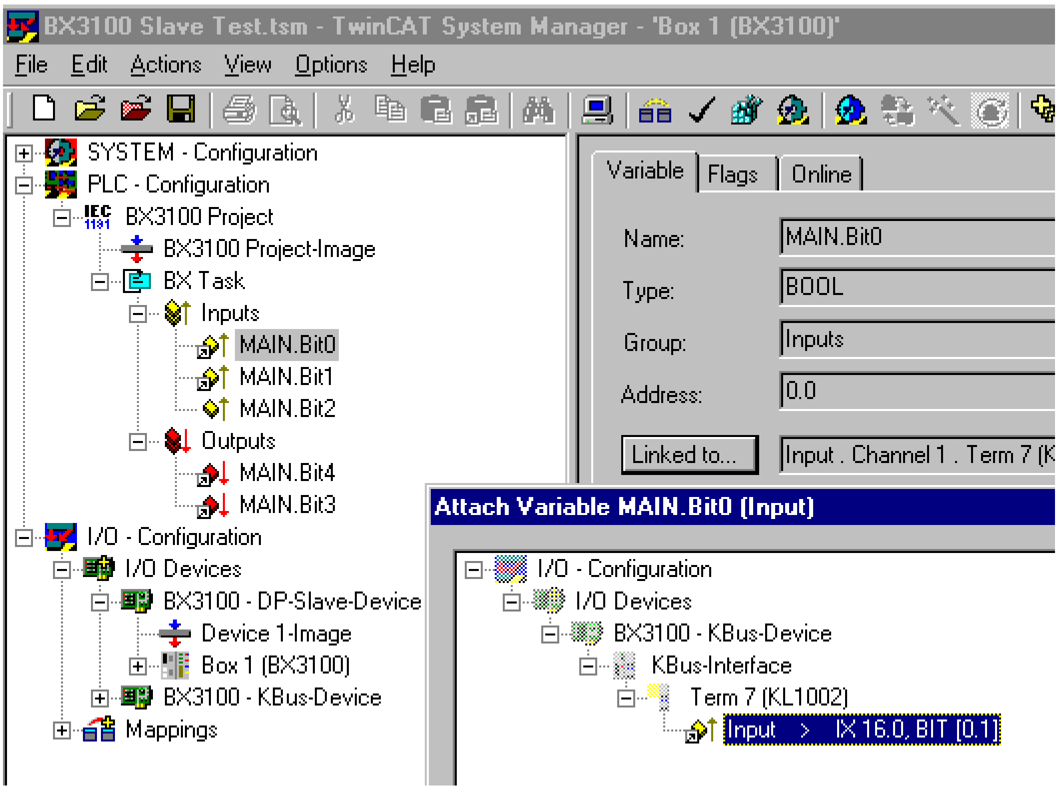


Fig. 50: Connecting PLC variable and hardware

Once all links have been created, activate the configuration *Actions/Activate Configuration* (Ctrl+Shift+F4) and start TwinCAT *Set/Reset TwinCAT to Run Mode*. Ensure that you have selected the correct target system (bottom right in the System Manager window).

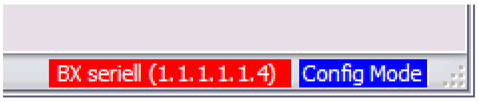


Fig. 51: Target system display

4.4.9.2 Measuring the PLC cycle time

The task time is set in PLC Control. The default setting is 20 ms.

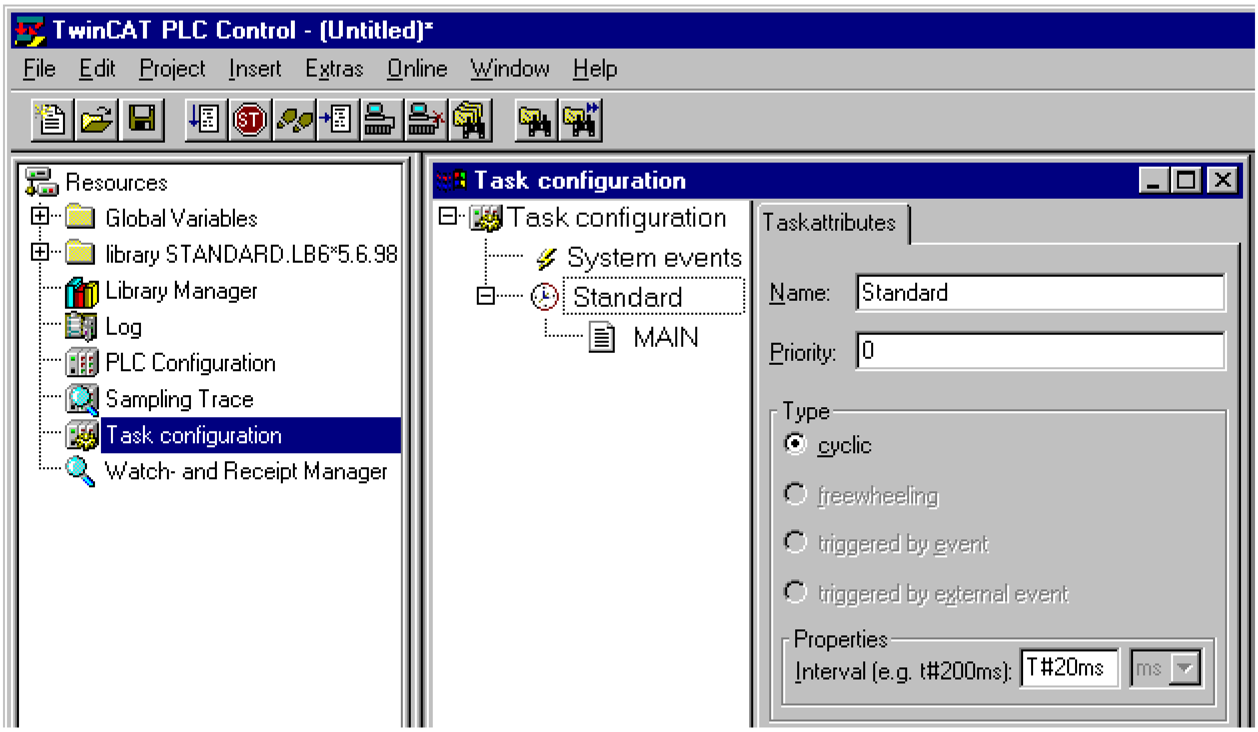


Fig. 52: Setting the task time

In the default setting, the PLC program is called every 20 ms, as long as the general cycle time is less than 20 ms. To determine the load of your system, the PLC cycle time can be measured in the System Manager. In order to ensure trouble-free operation, the set task time should be 20-30 % higher than the measured total cycle time. A precise cycle time breakdown can be found under K-Bus tab description. The total cycle time is displayed with the TcBase library (see TcBase.lbx or TcBaseBCxx50.lbx).

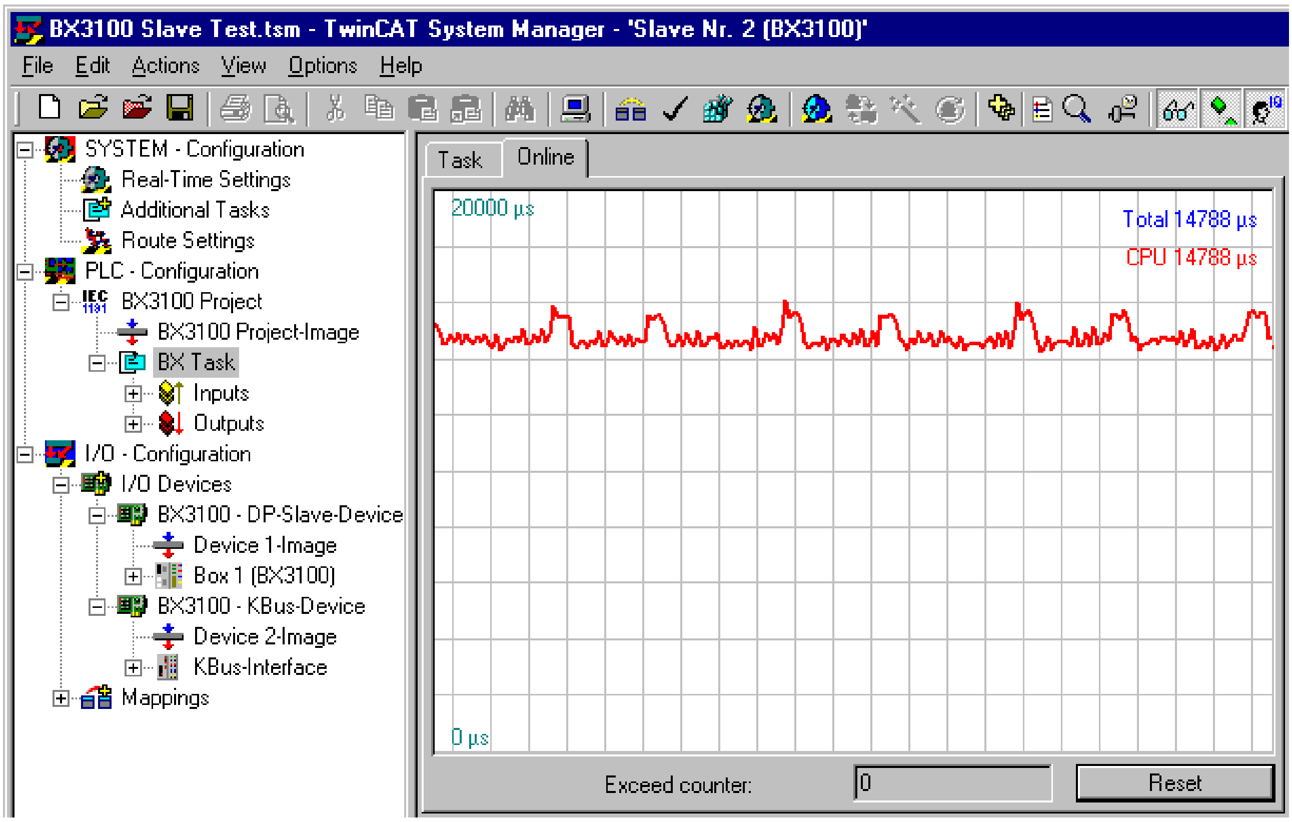


Fig. 53: Displaying the PLC cycle time

4.5 Configuration of the integrated I/Os

4.5.1 Analog I/Os

4.5.1.1 PT1000 / NI1000 / NI1000Tk5000 analog input

A temperature sensor can be directly connected to connections 8 and 9 of terminal strip X1.

The type of the temperature sensor is set to PT1000 in the standard and can be changed by means of register R32 or the KS2000 software.

Temperature sensor	Measuring range	Output of the process data	Register entry (bin / hex)
PT1000	-20 °C ... + 60 °C	in 1/10 degree increments. -20 °C = -200$_{dec}$ 0 °C = 0$_{dec}$ +20 °C = +200$_{dec}$ +40 °C = +400$_{dec}$ +60 °C = +600$_{dec}$	2# 0010 0000 0000 0000 16# 2000
NI1000	-15 °C ... +55 °C		2# 0101 0000 0000 0000 16# 5000
NI1000Tk5000	-20 °C ... +60 °C		2# 0111 0000 0000 0000 16# 7000
Resistance (direct)	910 Ω to 1,330 Ω	in 1/10 ohm increments. 910 Ω = 9,100$_{dec}$ 1,000 Ω = 10,000$_{dec}$ 1,330 Ω = 13,300$_{dec}$	2# 1110 0000 0000 0000 16# E000

4.5.1.2 Potentiometer input

A setpoint potentiometer can be directly connected to connections 9 and 10 of terminal strip X1 for specifying the room temperature.

It is recommended to use a potentiometer with a nominal value of 10 kΩ.

The change of the resistance value from 0 to 10 kΩ is represented in the process value as 0 to 10000 steps.

The resolution is 10 bits.

4.5.1.3 Analog inputs (10 V)

3 analog inputs with a resolution of 10 bits are available at connections 11, 12 and 13 of terminal strip X1.

Input voltage	Process value
0 V	0_{dec}
5 V	16383_{dec}
10 V	32767_{dec}

4.5.1.4 Analog outputs (10 V)

2 analog outputs with a resolution of 10 bits are available at connections 20 and 22 of terminal strip X2. Connection 21 is the common ground.

Process value	Output voltage
0_{dec}	0 V
16383_{dec}	5 V
32767_{dec}	10 V

4.6 KS2000

4.6.1 KS2000 - Introduction

The KS2000 configuration software permits configuration, commissioning and parameterization of bus couplers, of the affiliated bus terminals and of Fieldbus Box Modules. The connection between bus coupler / Fieldbus Box Module and the PC is established by means of the serial configuration cable or the fieldbus.

Fig. 54: KS2000 configuration software

Configuration

You can configure the Fieldbus stations with the Configuration Software KS2000 offline. That means, setting up a terminal station with all settings on the couplers and terminals resp. the Fieldbus Box Modules can be prepared before the commissioning phase. Later on, this configuration can be transferred to the terminal station in the commissioning phase by means of a download. For documentation purposes, you are provided with the breakdown of the terminal station, a parts list of modules used and a list of the parameters you have modified. After an upload, existing fieldbus stations are at your disposal for further editing.

Parameterization

KS2000 offers simple access to the parameters of a fieldbus station: specific high-level dialogs are available for all bus couplers, all intelligent bus terminals and Fieldbus Box modules with the aid of which settings can be modified easily. Alternatively, you have full access to all internal registers of the bus couplers and intelligent terminals. Refer to the register description for the meanings of the registers.

Commissioning

The KS2000 software facilitates commissioning of machine components or their fieldbus stations: Configured settings can be transferred to the fieldbus modules by means of a download. After a *login* to the terminal station, it is possible to define settings in couplers, terminals and Fieldbus Box modules directly *online*. The same high-level dialogs and register access are available for this purpose as in the configuration phase.

The KS2000 offers access to the process images of the bus couplers and Fieldbus Box modules.

- Thus, the coupler's input and output images can be observed by monitoring.
- Process values can be specified in the output image for commissioning of the output modules.

All possibilities in the *online mode* can be used in parallel with the actual fieldbus mode of the terminal station. The fieldbus protocol always has the higher priority in this case.

4.6.2 Representation of the integrated inputs and outputs

4.6.2.1 Configuration of the integrated inputs and outputs

The integrated inputs and outputs, as well as the sub-bus of the BC9191, are displayed as normal terminals for configuration in the KS2000 configuration software.

The configuration is then performed in the same way as for the respective standard Bus Terminal.

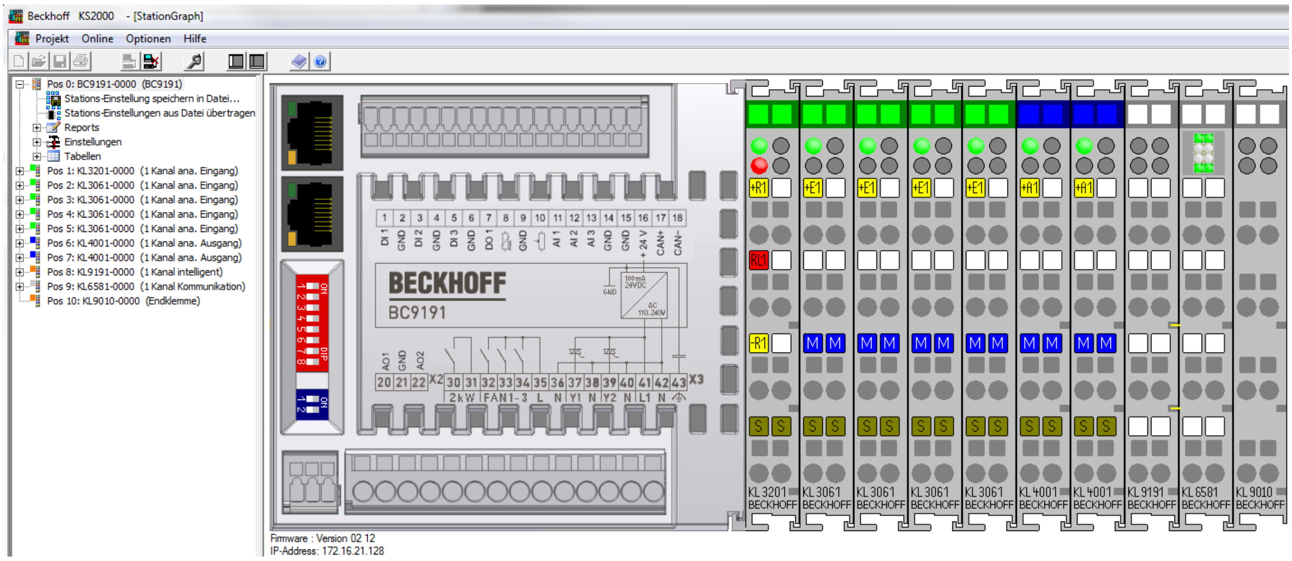


Fig. 55: Display of the BC9191 in the KS2000 software

5 Programming

5.1 TwinCAT PLC

The Beckhoff TwinCAT Software System turns any compatible PC into a real-time controller with a multi-PLC system, NC axis control, programming environment and operating station. The TwinCAT programming environment is also used for programming the BC/BX. If you have TwinCAT PLC (Windows NT4/2000/XP) installed, you can use the fieldbus connection or the serial port for downloading and debugging software.

TwinCAT I/O or TwinCAT PLC can also be used as the Ethernet Master (host), in order to exchange process data with the Bus Terminal Controller. TwinCAT provides you with the System Manager as a configuration tool, as well as the drivers and the ADS protocol.

Bus Terminal Controllers of the BCxx50, BCxx20 and BXxx00 series

These 2nd-generation Bus Terminal Controllers are configured with the TwinCAT System Manager and programmed with TwinCAT PLC Control. TwinCAT PLC must be installed for these couplers (Windows NT4, Windows 2000, Windows XP).

Programming and program transfer

- via the serial interface
- via the fieldbus interface [▶ 73] (only for Bus Terminal controllers for PROFIBUS, CANopen and Ethernet)

Online change

The Bus Terminal Controllers of the BX series and the BCxx50 support online change. This means that the PLC program is replaced with a new program without interrupting the program. The switch-over to the new program occurs after the task is completed. This means that two versions of the PLC program have to be stored. 512 kbyte are available, which therefore have to be divided by two, leaving 256 kbyte for the actual PLC program. In addition, several kbyte are required for task configuration etc. During an online change, dynamic data are stored in memory. Should a program approach the memory limit (program size greater than 240 kbyte), the online change may no longer work, even though the program may still be written to the BX after "Rebuild all".

When is online change not available?

Online change is not available under certain conditions,.

- Inserting of a new library
- Changing the task setting
- "Rebuild all"
- Controller memory limit is almost reached (PLC program greater than 90%)

5.2 TwinCAT PLC - Error codes

Error type	Description
PLC compiler error	Maximum number of POUs (...) exceeded
PLC compiler error	Out of global data memory ...

Error POUs

For each function block one POU (process object unit) is created. 256 function blocks are available by default.

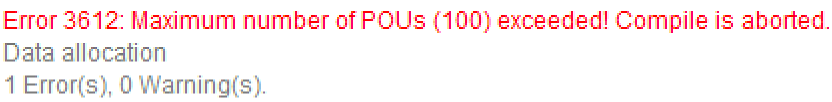


Fig. 56: Maximum number of POUs exceeded

If libraries are integrated this value may be insufficient. In this case, the number of POUs should be increased.

To this end, open in PLC Control under Projects/Options...

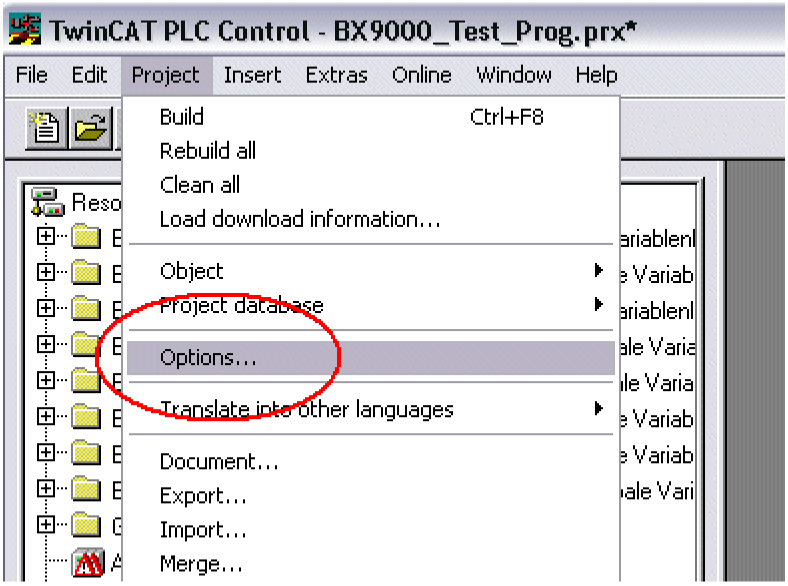


Fig. 57: Menu path Projects / Options / Controller Settings

...the controller settings.

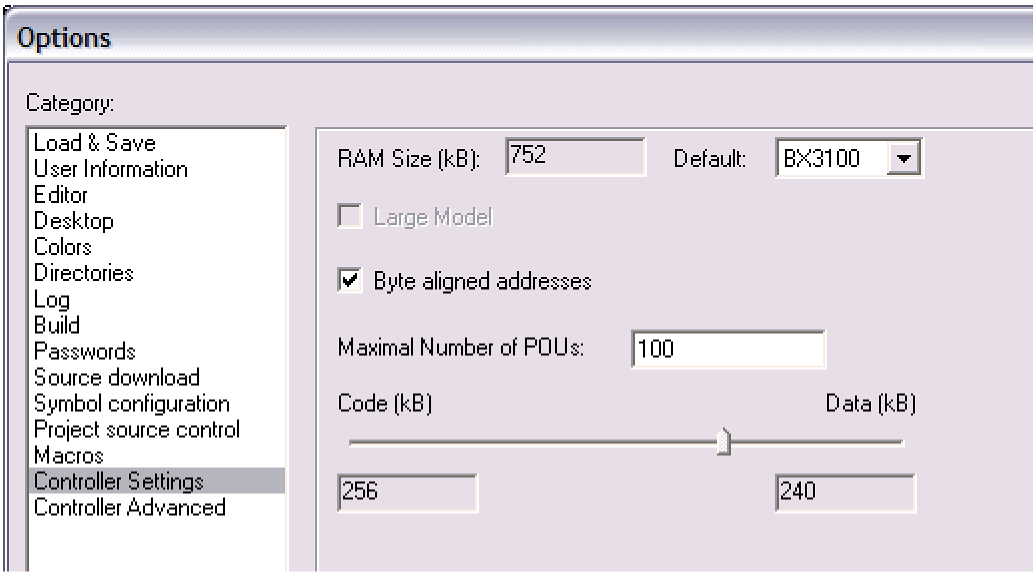


Fig. 58: Controller settings

Changing these settings will deactivate online changes.

Global memory error

Fig. 59: Global memory insufficient

2 x 16 kbyte of data are available by default. If large data quantities are to be used, this range should be increased. A maximum of 14 data segments are possible for the BX.

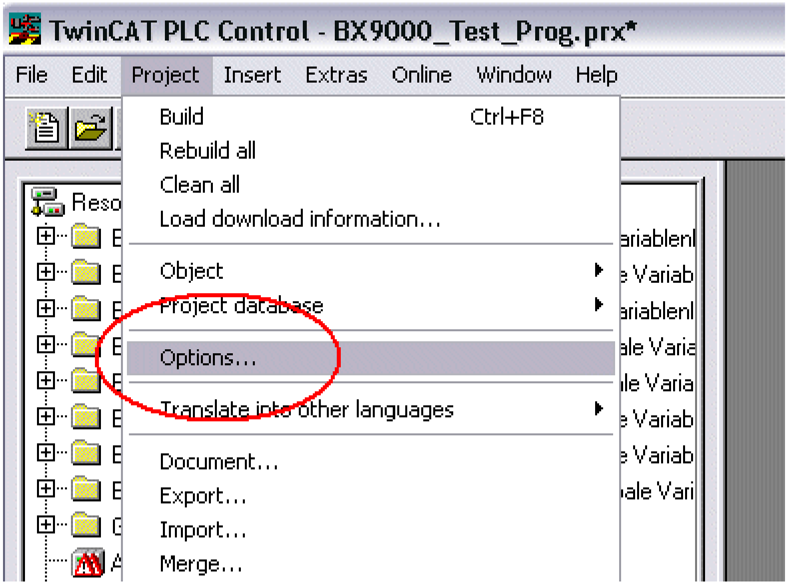


Fig. 60: Menu path Projects / Options / Build

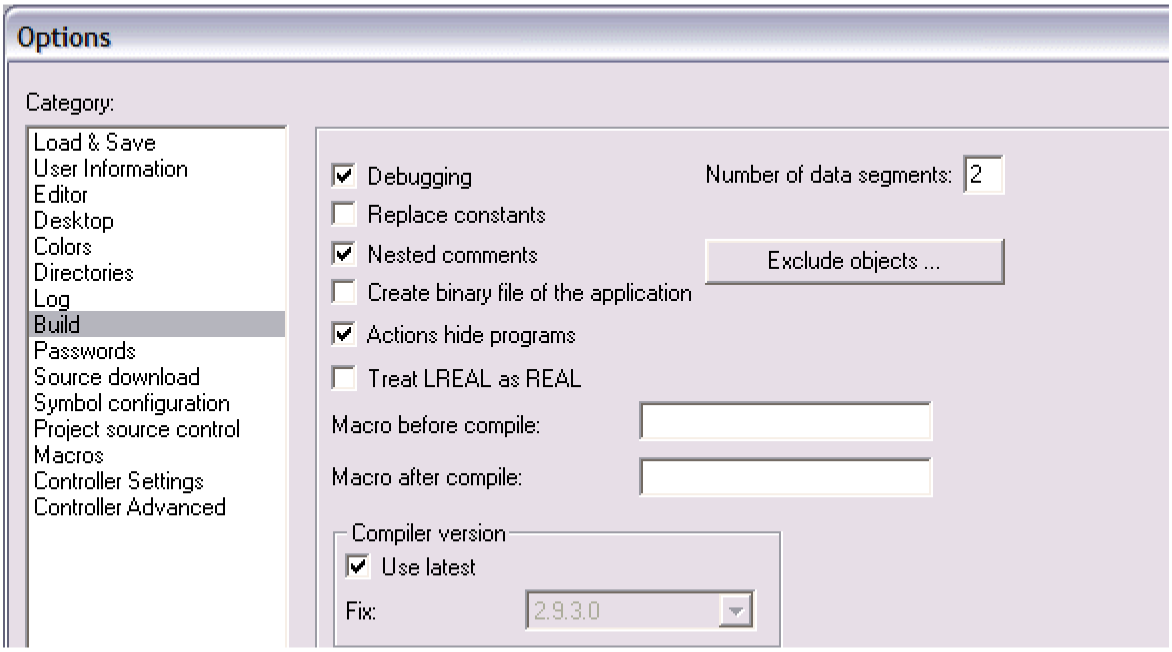


Fig. 61: Build

5.3 Remanent data

2000 bytes of remanent data are available on the BC9191 and the BX controller. These data are declared as VAR RETAIN in PLC Control.

Example

```
VAR RETAIN
    Test     :BOOL;
    Count    :INT;
END_VAR
```

Retain data are located between VAR RETAIN and END_VAR. These data are stored in a NOVRAM and are consistent across the whole 2 kbyte range. The RETAIN data are stored in the NOVRAM after each cycle. For 2 kbyte approx. 2 ms are required (for 1 kbyte approx. 1 ms). The variables can be configured locally or globally. Allocated variables (%MB, %QB, %IB) cannot be used as remanent data.

Do not use VAR_RETAIN in function blocks

VAR_RETAIN should not be used in function blocks. All FB data are copied into the retain memory. This leads to an unnecessary increase in cycle time, and the retain memory is filled with unnecessary data.

Do not use variables with address as remanent data

Variables that have been assigned an address (%MB, %QB, %IB) must not be used as remanent data.

Example for remanent data in the function block

This should be avoided, if possible, since all the data of a function block, in which even just a single remanent bit is found, are stored by default. A program sample can be found below.

Function block test (no program code required - in ST semicolon is sufficient)

```
FUNCTION_BLOCK Test
VAR_INPUT
END_VAR
VAR_OUTPUT
END_VAR
VAR
END_VAR
VAR_IN_OUT
    Counter    :INT;
END_VAR
```

MAIN program

```
PROGRAM MAIN
VAR
    fb_Test:Test;
END_VAR
VAR RETAIN
    iCounter1:INT;
END_VAR
```

```
fb_Test(Counter:=iCounter1);
```

5.4 Persistent data

The Bus Terminal Controller has 1000 bytes of persistent data available. In contrast to the retain data, these are not deleted, even with a new project, a PLC reset or a new download.

In order to use the persistent data, these must first be activated once with a function block from the PLC.

Secondly, the variables should reside in the allocated flag area. Here you can choose where the persistent data reside. 4 kbytes of allocated flags are available, of which 1000 bytes can be declared as persistent data.

Example

```
VAR
    Test AT %MX1000   :BOOL;
    Count AT %MB1002  :INT;
END_VAR
```

The **Persistent_Data** function block can be used to specify the start address and the length (in bytes) from which the data are to be persistent.

The input variable *WriteOffset* is used to specify the byte offset of the flag area, *WriteSize* is used for the length in bytes.

The function block can be found in the TcSystemBX.lbx library. Should this not be available, it can be downloaded from this documentation (see Libraries).

Example values

WriteOffset 1000
WriteSize 10

All data in the range %MB1000 - %MB1009 are then persistent. The variable type is irrelevant.

Like the retain data, the data are copied to the NOVRAM and are therefore writeable in each cycle.

Persistent data from firmware 1.17

Persistent data is supported for all BX controllers from firmware 1.17 or higher.

Parameters are valid immediately

The parameters only have to be written once, after which they are valid immediately. These data are stored permanently.
Activation of the factory setting deletes everything, including the persistent data.

Sample Program

Click on the link

(https://infosys.beckhoff.com/content/1033/bc9191/Resources/3207307659/.zip)

to download a sample program.

5.5 Allocated flags

4 kbyte of allocated flags are available. They can be used to assign different variable types to the same address, e.g. for converting strings to bytes. Data can also be placed here that can be read or written via ADS by the controller.

Allocated variables are not remanent data

For the Bus Terminal Controllers of the BX series and the BCxx50 the allocated variables are **not** saved as remanent data.

Reading/writing of allocated flags via ADS

The flags may also be read via the controller and ADS. In PROFIBUS, the DPV-1 services are used for this purpose, in CANopen SDO communication is used.
The AmsNetID can be obtained from the System Manager, or it can be displayed via the Bus Terminal Controller menu.
The PLC port number is 800.

Index group	Meaning	Index offset (value range)
0x4020	Flag (only BXxxx0)	0..4096

Example

BX program

```
VAR
    Flag_01 AT %MB0: WORD;
END_VAR
```

TwinCAT PC/CX master program

```
VAR
    fbADRSREAD: ADSREAD;
    Flag_M: WORD;
END_VAR
```

```
fbADRSREAD(
    NETID:='172.16.3.0.2.3' ,   (* AMSNetId BX *)
    PORT:=800 ,                 (* 800 - PLC *)
    IDXGRP:=16#4020 ,           (* 0x4020hex falgs *)
    IDXOFFS:=0 ,                (* byte offset *)
    LEN:=2 ,                    (* Lenght byte *)
    DESTADDR:=ADR(Merker) ,
    READ:=TRUE ,
    TMOUT:=t#1s );
IF NOT fbADRSREAD.BUSY THEN
    fbADRSREAD(READ:=FALSE);
END_IF
```

5.6 Local process image in delivery state (default config)

The process image of the Bus Terminal Controller consists of input, output and flag area. In addition, there are unallocated data without fixed address. They are created without specifying an address. For these variable types the memory allocation is as follows:

- BCxx50 48 kbyte,
- BC9x20, BC9191 128 kbyte and
- BXxx00 256 kbyte.

The maximum size of a variable or structure (array) is 16 kbyte. For the allocated data 2048 bytes of input data and 2048 bytes of output data are available. The Bus Terminal Controller has 4 kbyte of memory allocated for the flag area.

In the delivery state (default configuration) of the BX/BCxx50, fixed addresses are allocated for all connected Bus Terminals. The data for Ethernet communication start from address offset 1000_{dec}. The length of the Ethernet data depends on how much data has been configured; on the BX9000 it has a maximum length of 1000 bytes.

Inputs	Outputs
Bus Terminal %IB0 ...	Bus Terminal %QB0 ...
Ethernet DATA (PLC variables) %IB1000 ...(Modbus TCP/ADS-TCP/ADS-UDP)	Ethernet DATA (PLC variables) %QB1000 ... (Modbus TCP/ADS-TCP/ADS-UDP)
... %IB2047 maximum	... %QB2047 maximum

Addressing of the connected Bus Terminals

The default setting is for all the connected Bus Terminals to be assigned to the local process image.
Mapping within the Bus Terminal Controller is carried out according to the following rule:
First all the complex Bus Terminals, in the sequence they are physically inserted, followed by the digital Bus Terminals which are filled to a byte. The default mapping of the complex Bus Terminals is:

- complete evaluation
- Intel format
- Word alignment

Example structure

Bus Terminal Controller: 1 x BCxx50, BCxx20 or BXxx00
Position 1: 1 x KL1012
Position 2: 1 x KL1104
Position 3: 1 x KL2012
Position 4: 1 x KL2034
Position 5: 1 x KL1501
Position 6: 1 x KL3002
Position 7: 1 x KL4002
Position 8: 1 x KL6001
Position 9: 1 x KL9010

Table 1: Process image

Bus Terminal	Position	Input image	Output image	Size
KL1501	5	%IB0...%IB5	%QB0...%QB5	6 bytes
KL3002	6	%IB6...%IB13	%QB6...%QB13	8 bytes
KL4002	7	%IB14...%IB21	%QB14...%QB21	8 bytes
KL6001	8	%IB22...%IB29	%QB22...%QB29	6 bytes
KL1012	1	%IX30.0...%IX30.1	-	2-bit
KL1104	2	%IX30.1...%IX30.5	-	4-bit
KL2012	3	-	%QX30.0...%IX30.1	2-bit
KL2034	4	-	%QX30.2...%IX30.5	4-bit
KL9010	9	-	-	-

5.7 Mapping the Bus Terminals

The precise assignment of the byte-oriented Bus Terminals may be found in the configuration guide for the particular bus terminal. This documentation can be found on the Internet at http://www.beckhoff.de.

Byte oriented Bus Terminals	Bit oriented Bus Terminals
KL15x1	KL10xx, KL11xx, KL12xx, KL17xx, KM1xxx
KL25xx	KL20xx, KL21xx, KL22xx, KL26xx, KL27xx, KM2xxx
KL3xxx	
KL4xxx	
KL5xxx	
KL6xxx	
KL7xxx	
KL8xxx	
	KL9110, KL9160, KL9210, KL9260

5.8 Local process image in the TwinCAT configuration

The TwinCAT configuration (TwinCAT CONFIG) enables free mapping between fieldbus, K-bus and PLC variables. Variables can be linked independent of their address via the System Manager.

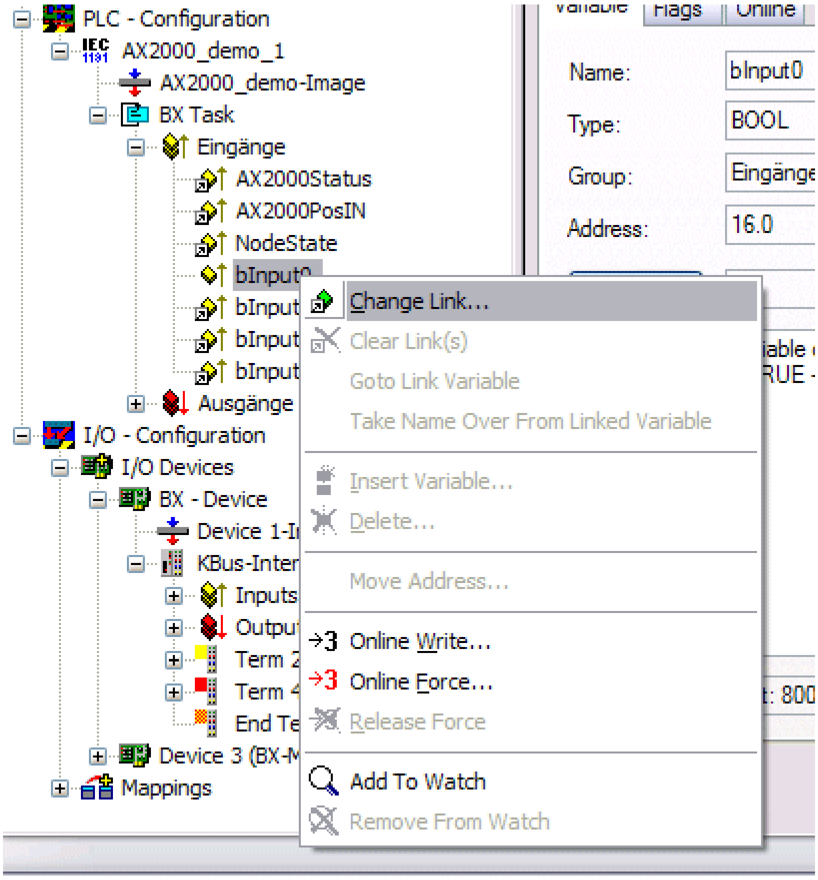


Fig. 62: Changing variable links

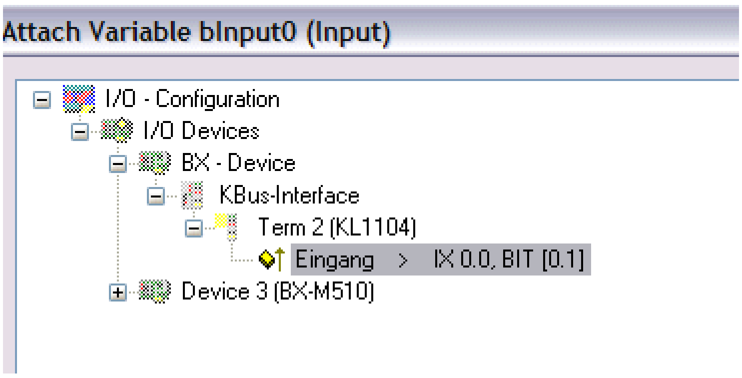


Fig. 63: Linking a variable with an input

In the default configuration all Bus Terminals are assigned fixed addresses. If a Bus Terminal is inserted, the whole address range may be shifted. The TwinCAT configuration enables allocated variables to be linked to a Bus Terminal, as required. This is parameterized in the System Manager, and the configuration is then downloaded to the Bus Terminal Controller (see TwinCAT configuration). It is also possible to upload an existing TwinCAT configuration.

5.9 Creating a boot project

The following memory resources are available for generating the boot project

- approx. 250 kbyte flash on the Bus Terminal controllers of the BX series;

- approx. 48 kbyte flash on the Bus Terminal controllers of the BCxx50 series.

PLC Control

After logging into TwinCAT PLC Control, a boot project can be created.

- Opening a PLC project
- Selecting the target system (or selection the serial interface)
- Logging into the BX/BCxx50
- Creating a boot project (Online\Create boot project)

The PLC LED lights up green once a valid boot project is available on the BX/BCxx50.

In the Bus Terminal controllers of the BX series, the PLC LED flashes orange while boot project is created. The PLC LED lights up orange if no boot project is available on the BX.

Deleting a boot project

The boot project can be deleted from the Bus Terminal Controller. The following steps must be followed:

- Opening the project
- Logging into the Bus Terminal Controller
- Deleting the boot project (Online\Delete boot project)

The PLC LED lights up orange when the boot project is deleted.

Using the current project as boot project

After an online change the old project is still shown as boot project. To use the current project (after the online change) as the boot project, the boot project has to be recreated.

Bypassing the start of the boot project*

With the Bus Terminal controllers of the BX series, starting of the boot project during booting can be prevented by pressing the Navi button. This does not delete the boot project. The project is reloaded when the Bus Terminal Controller is rebooted.

* from version 0.85

5.10 Communication between TwinCAT and BX/BCxx50

For transferring data from TwinCAT to the Bus Terminal Controller, it makes sense to organize the data in a structure. Please note the following to account for the differences in data management on the two systems.

- If two different data types are sent in sequence (e.g. byte and INT), the following variable is set to the next even address offset
- Boolean variables should never be allocated individually within a structure, since they would invariably occupy 1 byte. Boolean expressions should always be masked in a byte or word.

Example 1: A structure on the BX/BCxx50 and on the PC

Variable	BX/BCxx50 memory	PC memory (TwinCAT)
Byte	%..B0	%..B0
INT (1)	%..B2	%..B1
INT (2)	%..B4	%..B3

Due to the fact that another variable type (INT) follows the first byte, in the BX/BCxx50 it was assigned the next free even address. In order to achieve the same data structure on both systems, a dummy byte has to be inserted in the PC project (see example 2).

Example 2: A structure on the BX/BCxx50 and on the PC with the same memory allocation

Variable	BX/BCxx50 memory	PC memory (TwinCAT)
Byte	%..B0	%..B0
Byte (dummy)	%..B1 (not necessarily required, since the system deals with this itself if the variable does not exist)	%..B1
INT (1)	%..B2	%..B2
INT (2)	%..B4	%..B4

Data structure

```
Type PB_Data
STRUCT
    wVar_1:WORD;
    iValue_1:INT;
    iValue_2:INT;
    iValue_3:INT;
END_STRUCT
END_TYPE
```

Creating a variable structure

```
VAR_Global
    strData_Out AT %QB1000:PB_Data; (*PLC Variables *)
    bInput_01 AT %IX0.0:BOOL; (* Input from a terminal *)
END_VAR
```

Small programming example

```
strData_Out.wVar_1.0:=bInput_01;
```

Do not use real values in a mixed data structure

A mixed data structure should not contain real values. If this is nevertheless the case, the high and low words must be swapped in the BX/BCxx50 or in the TwinCAT master project. It is better to use an array of Real values or to transfer the Real values individually.

Larger fieldbus data blocks

You can transfer larger fieldbus data blocks, in order to have a reserve for your structure. Disadvantage: These reserves are then transferred with each fieldbus telegram, resulting in overload of the fieldbus communication.

5.11 Program transfer

5.11.1 Program Transfer via Ethernet

TwinCAT offers a facility for transferring the user program to the Bus Terminal Controller over the fieldbus. The BC/BX can be selected as the target system in PLC Control, after saving in the registry and restarting the TwinCAT system. The TwinCAT-level TwinCAT PLC is necessary.

Minimum requirements:

- TwinCAT 2.10 build 1251

Initializing the Bus Terminal Controllers

Possibility 1: If you use TwinCAT as a polling PLC.
The coupler must first be made known to the system before it can be selected in PLC Control.
Enter the Bus Terminal Controller in the System Manager, specify type, quantity and size of the fieldbus variables and link them with a task. Save your settings and activate the configuration. Then start the TwinCAT system and the cyclic task.

Possibility 2: If you only use TwinCAT for programming or configuring:
Click the TwinCAT icon, and open the features. You can enter the BX9000 under the AMS router.
Name: variable
AMS Net Id: IP address plus ".1.1"
IP address: IP address of the BX9000
Transport type: TCP/IP

Now start TwinCAT, either in the configuration mode (the blue TwinCAT icon) or in the RUN mode (the green TwinCAT icon)

TwinCAT System Manager

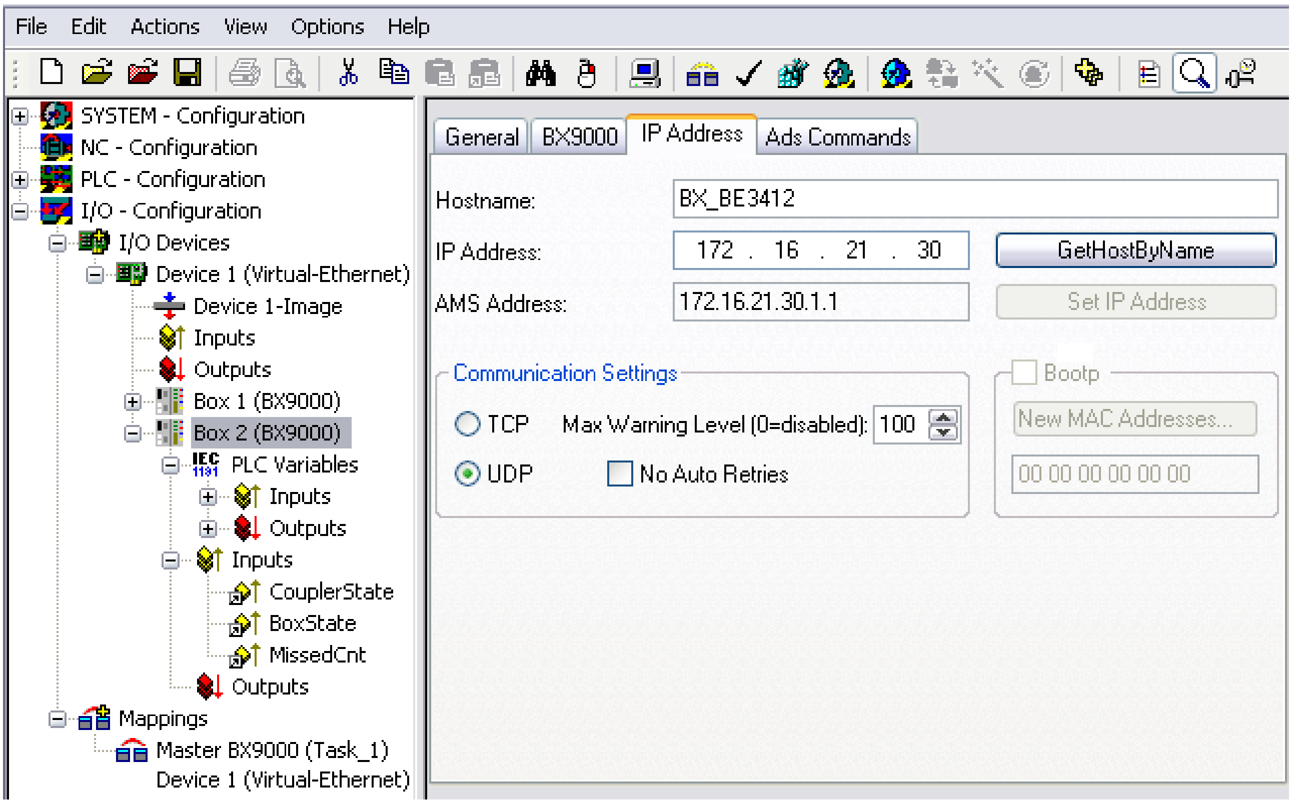


Fig. 64: IP address of the BX9000 in the TwinCAT System Manager

PLC Control

When TwinCAT PLC Control is restarted, TwinCAT asks for the target platform, i.e. the device on which the user program is later to run. TwinCAT offers two target platforms as controller, the PC or the Bus Terminal Controller.

Two options are available to you for transmission to the Bus Terminal Controller:

- AMS for BCxx00 (Bus Terminal Controller without online change)
- AMS for BCxx50 and BX (Bus Terminal Controller with online change)
- BC serial – the serial cable for communication via the RS232 interface of the PC and the programming interface of the Bus Terminal Controller
- For the BC9191, select "BCxx50 or BX via AMS"

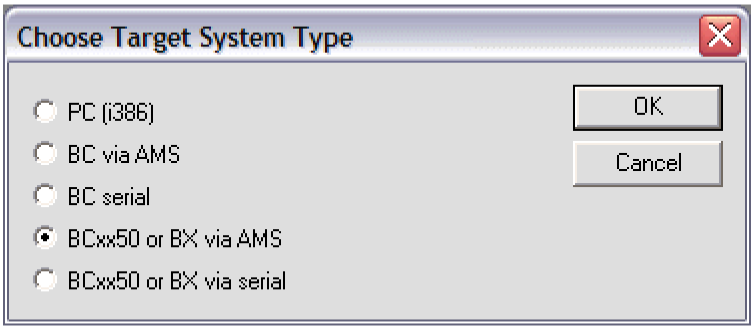


Fig. 65: Selecting the data transfer route - AMS

After your program has been created, select the target system under the *Online* toolbar. TwinCAT must be running to do this. In the sample, this is the Ethernet card with Box 1 and the Run-Time 1 of the Bus Terminal Controller.

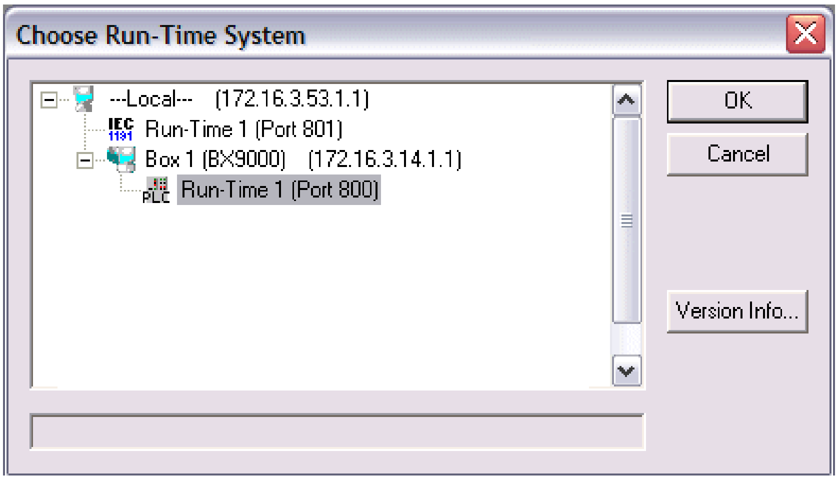


Fig. 66: Choose Target System

5.11.2 Up- and downloading of programs

The Bus Terminal Controller has a memory for the source code. It can be used for storing the program, the task configuration, and the libraries. Should the memory be insufficient, the source code may be stored without task configuration and libraries. This takes up significant less memory space!

General settings

The timing of the source code download to the target system can be specified via Edit/Options. Open the options menu.

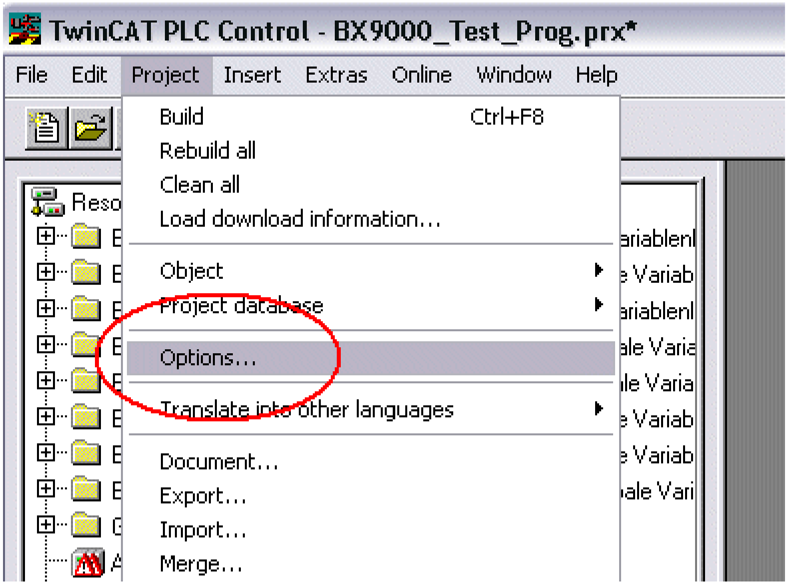


Fig. 67: Opening the options menu

Select Source Download.

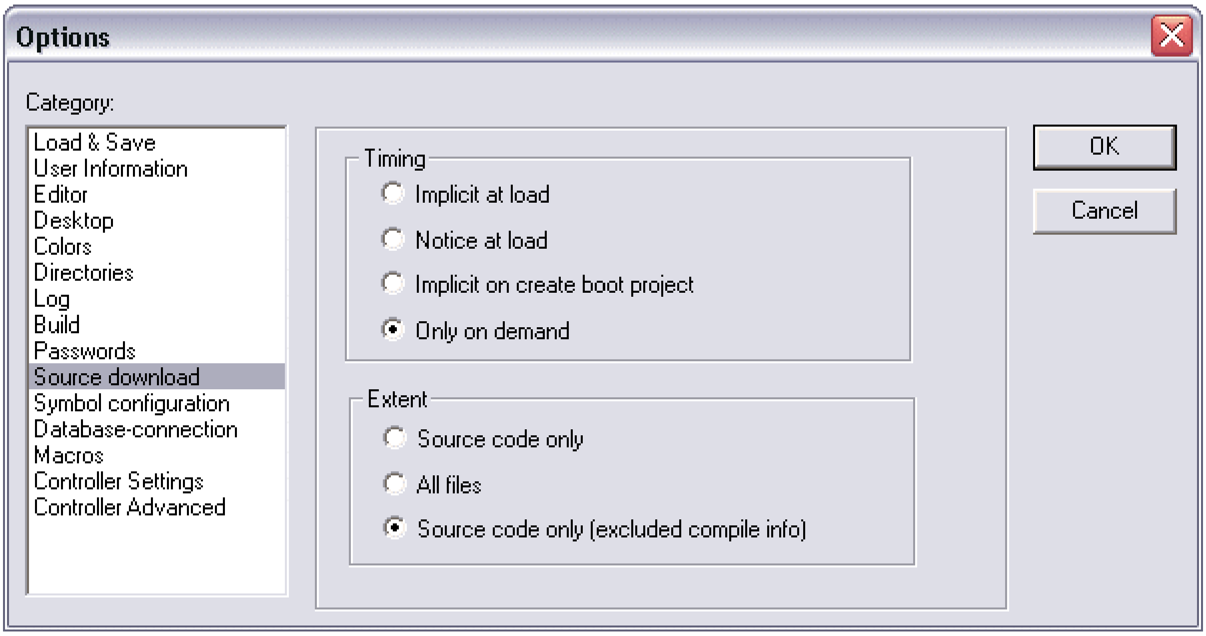


Fig. 68: Selecting Source Download

Here you can set which parts of the source code are to be downloaded to the Bus Terminal Controller, and when.

Source code only: the prx file with information on the online change is transferred. Login via online change is possible (the PLC does not stop).
All files: as *Source code only*, plus all required libraries.
Source code only (compile info excluded): only the prx file is transferred. Login is only possible when the PLC stops.

Which option you can use depends on the size of your projects.

Downloading a program

The source code can be transferred to the target system on request. This requires the user to be logged in with his program. Under Online/Source code download the program code can now be transferred to the Bus Terminal Controller.

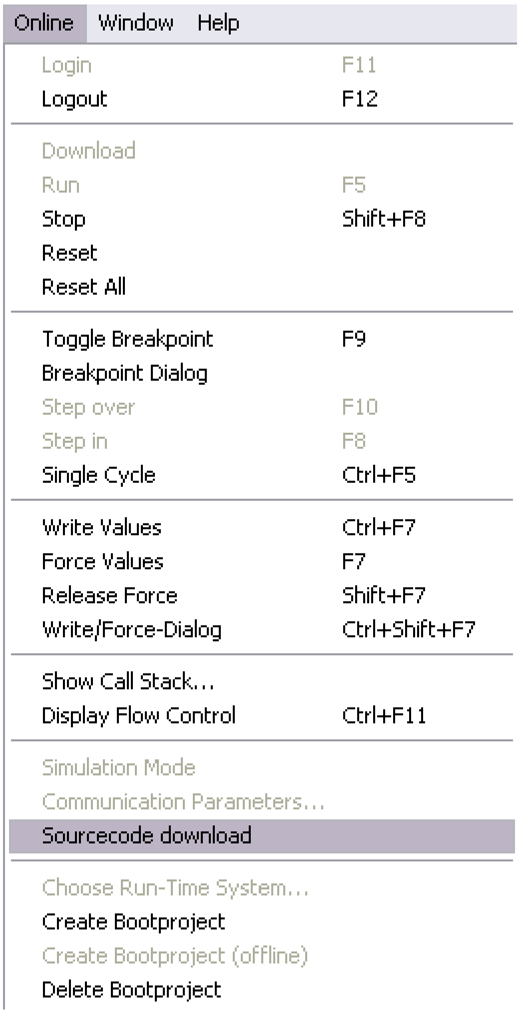


Fig. 69: Downloading the program code

After a short delay, a window will open that indicates the download progress.

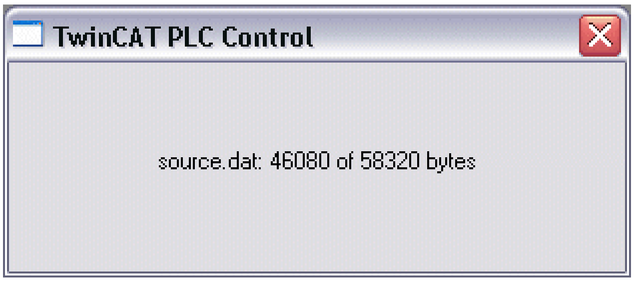


Fig. 70: Download progress

Uploading a program

For uploading the program code again, open a new file in PLC Control. Then click on the PLC button.

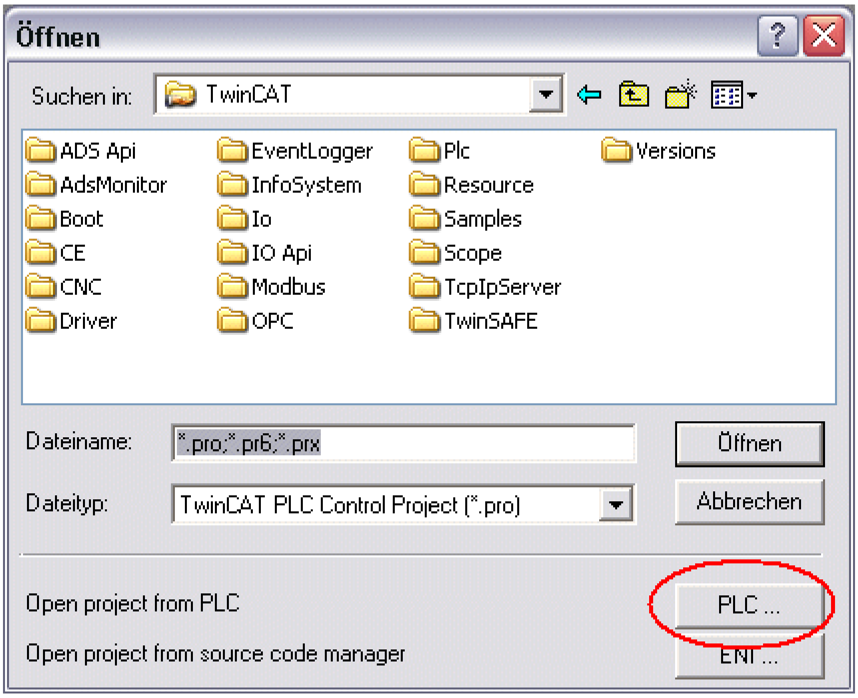


Fig. 71: Uploading a program

Select the data transfer route:

- *BCxx50 or BX via AMS*, if you are connected to the Bus Terminal Controller via the fieldbus, or
- *BCxx50 or BX via serial*, if you are connected to the Bus Terminal Controller via the serial interface.

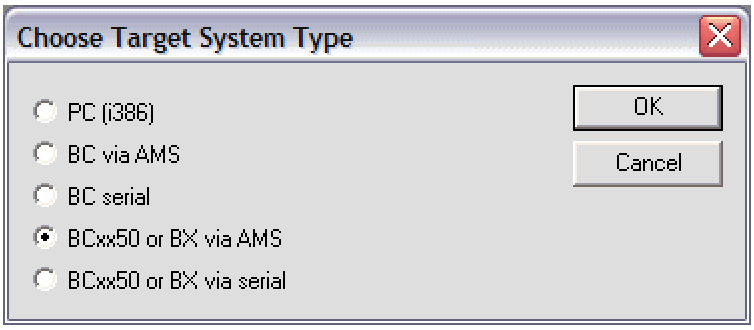


Fig. 72: Selecting the data transfer route

Then select the device and confirm with OK.

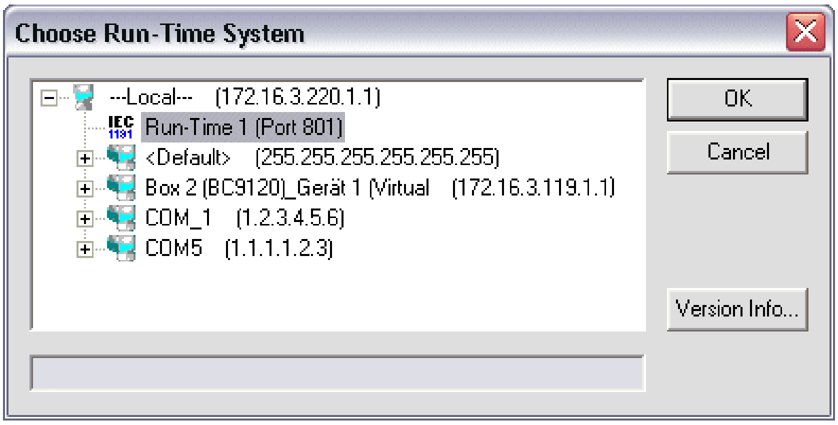


Fig. 73: Selecting the device

The source code will now be uploaded.

Password

You can protect your project with a password (in PLC Control Project/Options/Passwords).

5.12 Libraries

5.12.1 Libraries overview

The TwinCAT 2 automation software provides various libraries for the BC9191 and BC9191-0100 (see Beckhoff Information System). These libraries are also used for the Bus Terminal Controllers (Bus Couplers with PLC functionality) of the BC9050, BC9020, BC9120 and BX9000 series.

Download and documentation

If libraries are not installed by default, perform a download, extract the library from the .zip file and copy it into the TwinCAT directory TwinCAT\PLC\LIB. KS2000KS2000KS2000

- Standard.lbx (installed by default in TwinCAT 2.11)
- TcBaseBCxx50.lbx (installed by default in TwinCAT 2.11)
  Documentation: https://infosys.beckhoff.com/content/1033/bc9xx0/3740793995.html
- TcSystemBCxx50.lbx (installed by default in TwinCAT 2.11)
  Documentation: https://infosys.beckhoff.com/content/1033/bc9xx0/3740791947.html
- TcBaseBX9000.lbx
  Documentation: https://infosys.beckhoff.com/content/1033/bx9000/3207183627.html
  Download: https://infosys.beckhoff.com/content/1033/bx9000/Resources/zip/3207318539.zip
- TcSystemBX9000.lbx
  Documentation: https://infosys.beckhoff.com/content/1033/bx9000/3207213707.html
  Download: https://infosys.beckhoff.com/content/1033/bx9000/Resources/zip/3207385995.zip

Further libraries with useful functions or communication blocks can also be found in the Beckhoff Information System:
https://infosys.beckhoff.com/content/1033/tcinfosys/12706086667.html

When using the libraries, firmware 3.1 or higher is recommended for the BC9191 and BC9191-0100.

Use the library that matches the firmware

Always use the latest libraries in conjunction with the latest BC firmware. If you update the firmware of your Bus Terminal Controller, please also update the libraries. Copy the new libraries into the LIB folder, remove them from your project and re-insert them.

6 PLC program

6.1 Fan controller with room temperature control

The BC9191 is supplied with a pre-installed PLC program. This OLC program can be used for a three-stage fan controller with room temperature control (heating/cooling) for a 4-pipe system.

The factory-installed PLC code for TwinCAT 2 is also available for download as a sample program in a ZIP file (see Documentation and Downloads / Configuration files):

BC9191: www.beckhoff.de/BC9191

BC9191-0100: www.beckhoff.de/BC9191-0100

Application

With the pre-installed standard air conditioning application it is possible to attain the highest energy efficiency standard of EN15232 Class A. The standard application contains the basic room air conditioning functions according to VDI 3813.

The application contains basic function blocks and basic functions such as:

- a local temperature measurement with smoothing function and a compensation value,
- a local setpoint value shift/setpoint correction
- a scaling function
- the control of analog actuators
- a PI controller with input via the proportional band
- occupancy detection and occupancy signal
- Window monitoring
- local overriding of the fan stages is possible
- load optimization is possible

Furthermore, room air conditioning functions are implemented, such as:

- energy level selection
- function selection
- Setpoint determination block for the different energy levels
- 3-stage fan control (FanCoil)

Notes

- The pre-installed standard air conditioning application is mapped as a default configuration, i.e. the TwinCAT System Manager is not required for the linking of the PLC variables with the I/O level.
- The Index Groups 16#4020 and 16#4021 can be accessed for both reading and writing from a higher-level controller, e.g. BMS or floor controller, by ADS or Modbus TCP. This is accounted for in the application.
- Parameters of individual subprograms are saved persistently. See also PersistentDataGlobalState.

Main_1 (main program)

All subprograms are called in the main program of the standard application. The connections and communication between the program blocks are given in the main program by the connecting lines. All the program blocks can therefore be seen in a single functioning context.

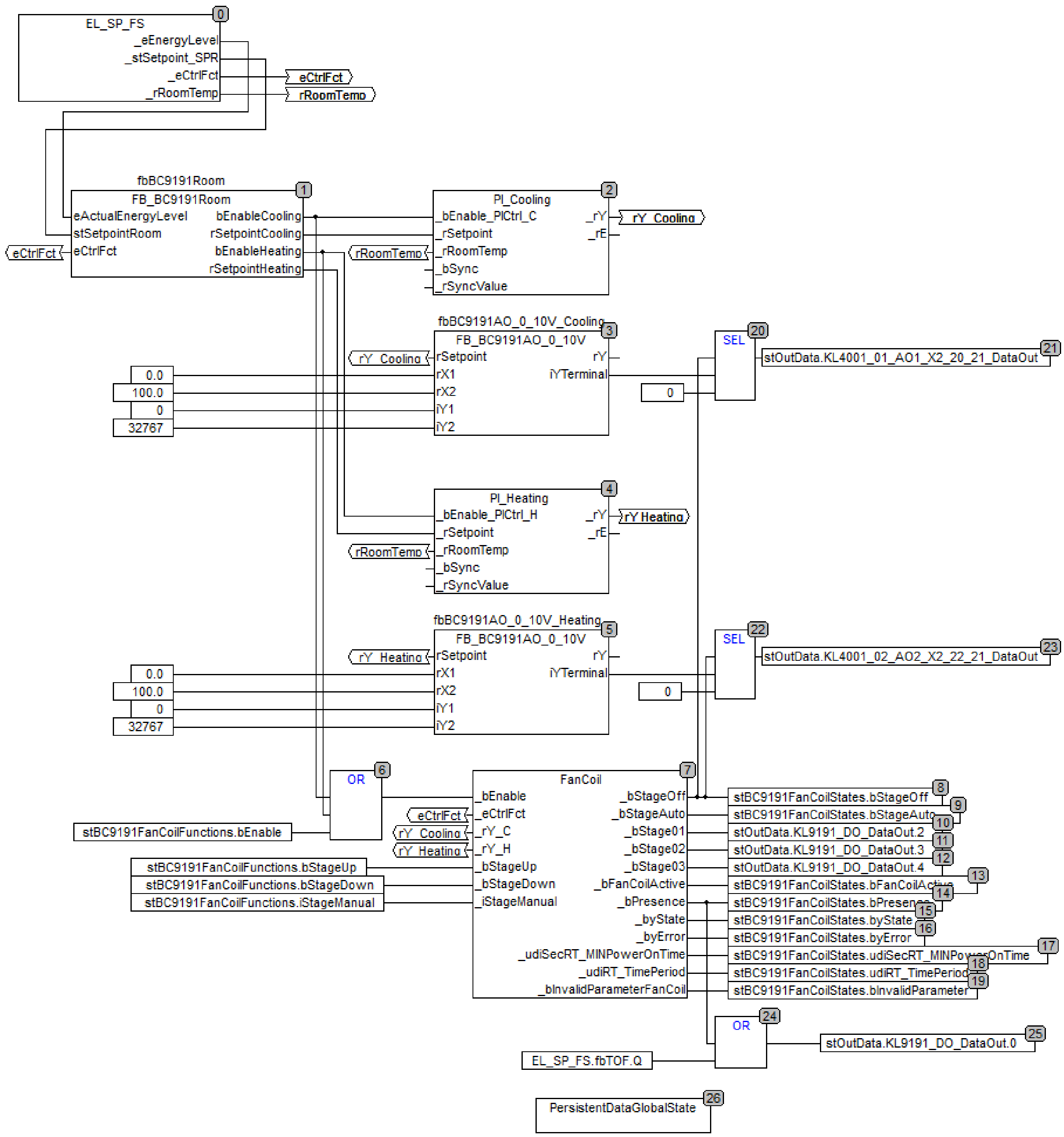


Fig. 74: Fan control - main program

Connection from block 0 to block 1

The EnergyLevelSelection_SetpointGeneration_FunctionSelection subprogram (EL_SP_FS) transmits the information regarding the energy level, the setpoints of the four energy levels for the heating and cooling operation, the control function (heating/cooling/OFF) and the current room temperature (to blocks 2 and 4).

Connection from block 1 to block 2

FB_BC9191Room enables the cooling function with the valid/calculated energy level setpoint.

Connection from block 1 to block 4

FB_BC9191Room enables the heating function with the valid/calculated energy level setpoint.

Connection from block 1 to block 6

If the heating or cooling function is activated, it enables the fan coil on block 7.

Connection from block 2 to block 3

The cooling controller transmits its control value (0..100%) to the control valve. This control value is scaled to the output signal with the value range 0..32767; block 3.

Connection from block 4 to block 5

The heating controller transmits its control value (0..100%) to the control valve. This control value is scaled to the output signal with the value range 0..32767; block 5.

Block 7

3-stage fan coil, which switches the stages automatically depending on the control signal from the controller.

System diagram

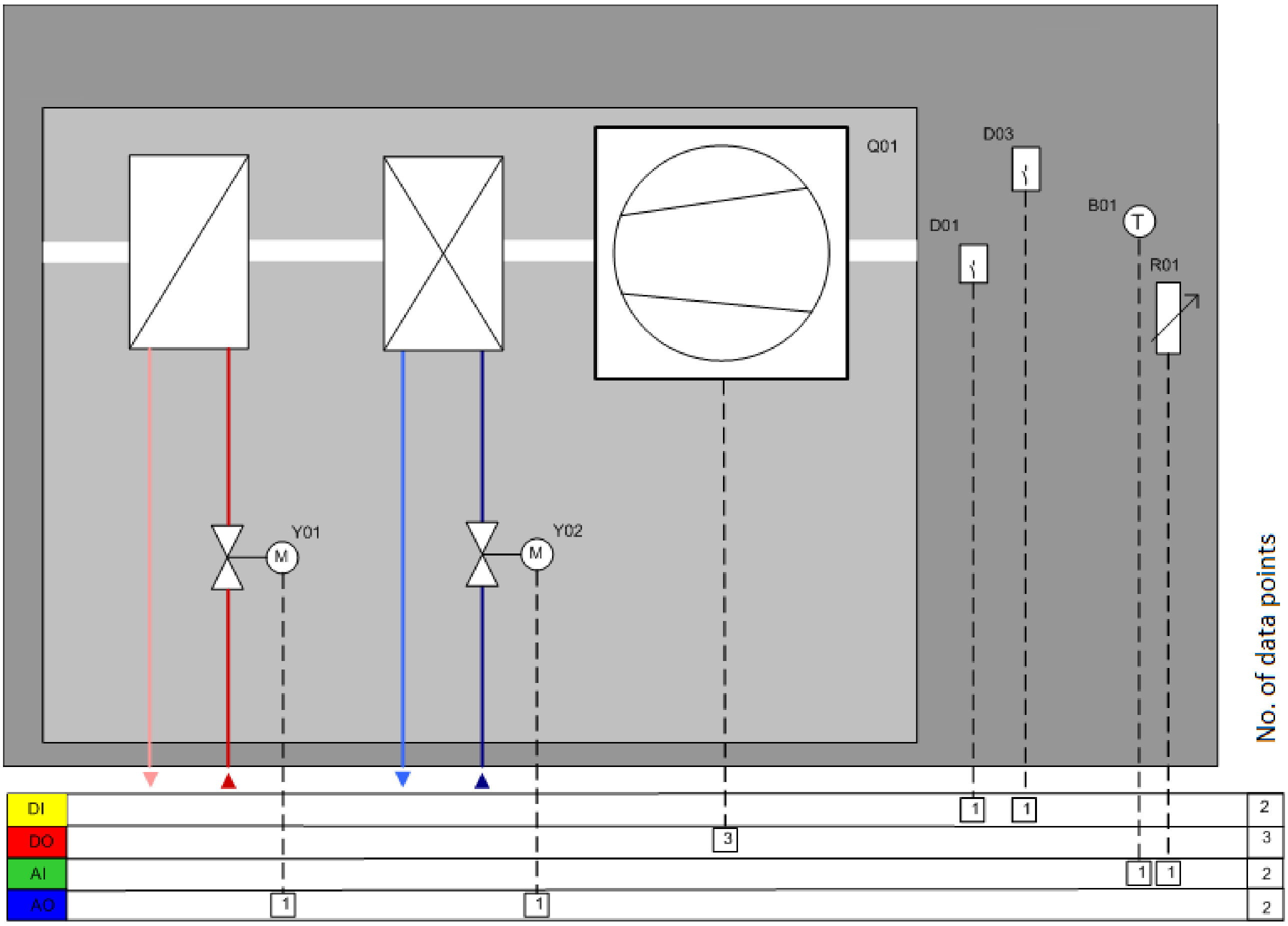


Fig. 75: Fan control - system diagram

Data point list

DP	Name	Type	Program block / function block	Variable	Feature	Terminal strip
Y01	Heating valve	AA	PI_Heating / fb-BC9191AO_0_10V_Heating	stOutData.KL4001_02_AO2_X2_22_21_DataOut	mandatory	X2 Contact X2:22; X2:21
Y02	Cooling valve	AA	PI_Cooling / fb-BC9191AO_0_10V_Cooling	stOutData.KL4001_01_AO1_X2_20_21_DataOut	mandatory	X2 Contact X2:20; X2:21
Q1	3-stage fan coil	DA	FanCoil	stOutData.KL9191_DO_DataOut.2 stOutData.KL9191_DO_DataOut.3 stOutData.KL9191_DO_DataOut.4	binding binding binding	X3 Contact X3:32; X3:35 Contact X3:33; X3:35 Contact X3:34; X3:35
D01	Window contact	DE	EnergyLevel	stInData.KL9191_DI_DataIn.0	mandatory	X1 Contact X1:1; X1:2
D03	Occupancy sensor	DE	EnergyLevel	stInData.KL9191_DI_DataIn.2	mandatory	X1 Contact X1:5; X1:6
B01	Room temperature	AE PT/NI 1000 NI1000Tk5000	EL_SP_FS / fb-BC9191TemperatureSensor	stInData.KL3201_AI_RTD_X1_8_9_DataIn stInData.KL3201_AI_RTD_X1_8_9_State	binding binding	X1 Contact X1:8; X1:9 Contact X1:8; X1:9
R01	local set value generator	AE 0 - 10,000 Ω	EL_SP_FS / fb-BC9191Poti / F_BC9191Scale	stInData.KL3061_01_AI_Poti_X1_10_9_DataIn stInData.KL3061_01_AI_Poti_X1_10_9_State	binding binding	X1 Contact X1:10; X1:9 Contact X1:10; X1:9

Key

AE: Analog input
AA: Analog output
DE: Digital input
DA: Digital output

6.2 Description of the POUs

6.2.1 Description of the internal function blocks and subprograms

6.2.1.1 FB_BC9191EnergyLevel

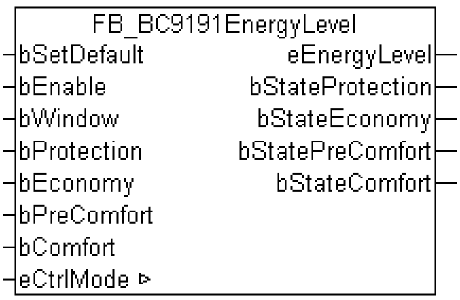


Fig. 76: Function block FB_BC9191EnergyLevel

Application

This function block is for the adaptation of the supply of energy for the use of the building. The type of room utilization is set by the BMS. The longer a building or room is not used, the further the energy level can be lowered. The energy level currently selected by the function block is transferred to the room temperature controller.

Protection:

This operation mode is activated in the case of long absence times e.g. during works holidays or also when a window is open. The energy level is very low and serves only to protect the building from damage caused by frost or overheating.

Economy:

The Economy energy level is used for the economy mode. Economy mode is activated, for example, at night by a timer switch schedule.

PreComfort:

The PreComfort energy level is for an unused room which, however, can be occupied again shortly. The standby mode is frequently activated by a timer schedule.

Comfort:
If the room is occupied, it is in Comfort mode. Comfort mode can be activated by a timer switch schedule or by presence recognition.

VAR_INPUT

```
Name                    : Type
```

```
bSetDefault             : BOOL;
bEnable                 : BOOL;
bWindow                 : BOOL;
bProtection             : BOOL;
bEconomy                : BOOL;
bPreComfort             : BOOL;
bComfort                : BOOL;
```

bSetDefault: If the variable is TRUE, the default values of the VAR_IN_OUT variables are adopted.

bEnable: The function block is activated by a TRUE at this input.

bWindow: The window contact is connected to this input. TRUE means that the window is OPEN. FALSE means that the window is CLOSED.

bProtection: Protection mode is activated with the input bProtection. Protection mode is active if the input is TRUE.

bEconomy: Economy mode is activated with the input bEconomy. Economy mode is active if the input is TRUE.

bPreComfort: The Pre-comfort level is activated with this input. The Pre-comfort level is active if the input is TRUE.

bComfort: The Comfort level is activated with this input if the room is occupied.

VAR_OUTPUT

```
Name                  : Type
```

```
eEnergyLevel          : E_BC9191EnergyLevel;
bStateProtection      : BOOL;
bStateEconomy         : BOOL;
bStatePreComfort      : BOOL;
bStateComfort         : BOOL;
```

eEnergyLevel [▸ 85]**:** This output contains the current energy level.

bStateProtection: The state of the *bProtection* input is relayed to the outside in the operation modes *eEnergyLevel_AUTO_I* and *eEnergyLevel_AUTO_II*.

bStateEconomy: The state of the *bEconomy* input is relayed to the outside in the operation modes *eEnergyLevel_AUTO_I* and *eEnergyLevel_AUTO_II*.

bStatePreComfort: The state of the *bPreComfort* input is relayed to the outside in the operation modes *eEnergyLevel_AUTO_I* and *eEnergyLevel_AUTO_II*.

bStateComfort: The state of the *bComfort* input is relayed to the outside in the operation modes *eEnergyLevel_AUTO_I* and *eEnergyLevel_AUTO_II*.

VAR_IN_OUT

Name	: Type	Persistent	Default
eCtrlMode	: E_BC9191EnergyLevel [▸ 85];	X	eEnergyLevel_AUTO_I

eCtrlMode: Using this ENUM the operation mode can be preselected from the building management level.

6.2.1.2 E_BC9191EnergyLevel

```
TYPE E_BC9191EnergyLevel:
(
     eEnergyLevel_AUTO_I       := 0,
     eEnergyLevel_Protection   := 1,
     eEnergyLevel_Economy      := 2,
     eEnergyLevel_PreComfort   := 3,
     eEnergyLevel_Comfort      := 4,
     eEnergyLevel_AUTO_II      := 5
);
END_TYPE
```

eEnergyLevel_AUTO_I

Automatic mode I means that in this operation mode it is only possible to switch to the next higher energy level if the lower energy levels are already TRUE. In this operation mode the states of the inputs *bProtection*, *bEconomy*, *bPreComfort* and *bComfort* are evaluated for the selection of the energy level. Please see diagram.

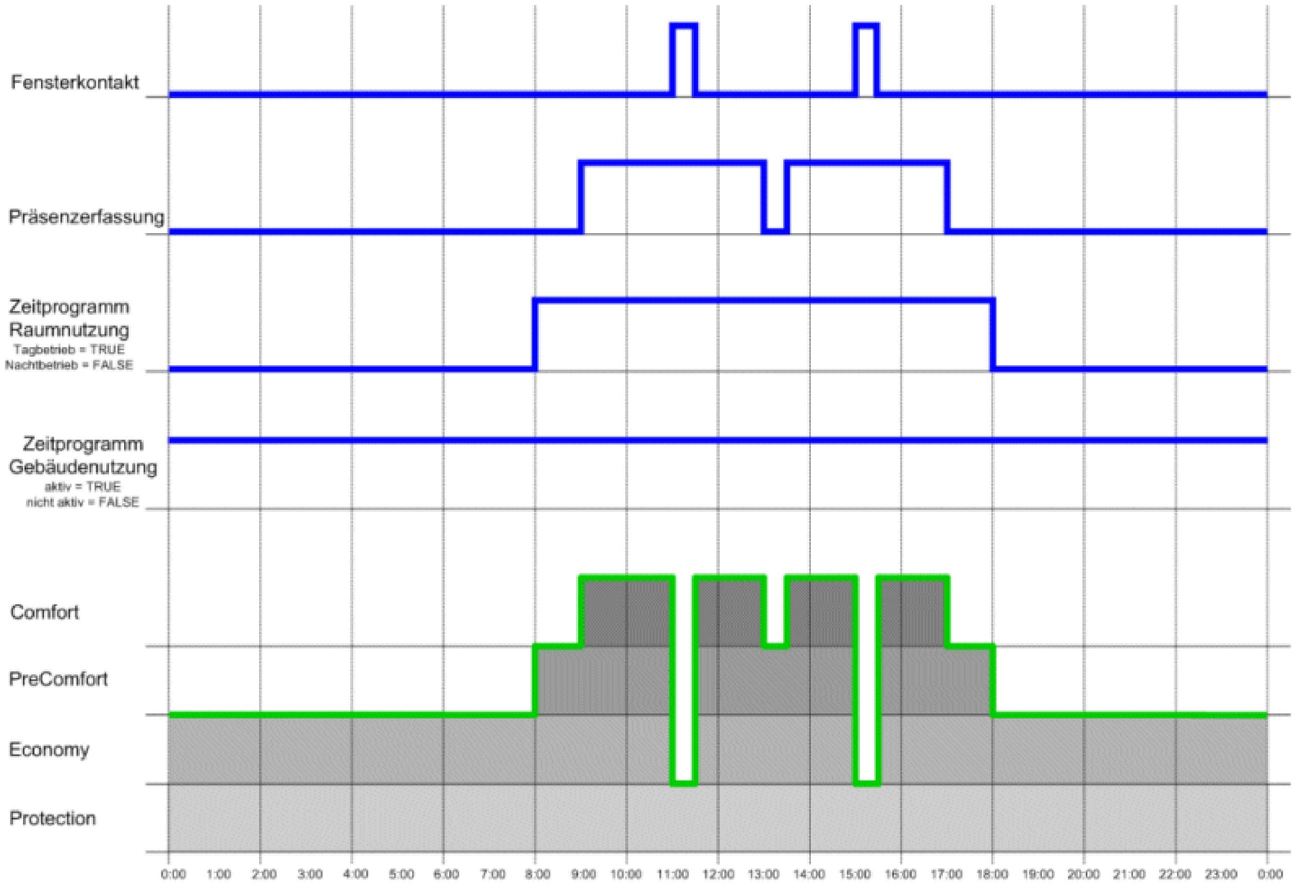


Fig. 77: Assignment of the energy level in automatic mode-I

eEnergyLevel_Protection

In this operation mode, which can be manually set by the building management level, the room/zone/area is maintained in a minimum energy state. This operation mode is purely a Protection mode.

eEnergyLevel_Economy

In this operation mode, which can be manually set by the building management level, the room/zone/area is maintained in economy mode. This operation mode is to be set in case of lengthy periods of absence or at night.

eEnergyLevel_PreComfort

In this operation mode, which can be manually set by the building management level, the room/zone/area is maintained in Pre-comfort mode. This operation mode is to be set in case of short periods of absence.

eEnergyLevel_Comfort

In this operation mode, which can be manually set by the building management level, the room/zone/area is maintained in comfort mode. This operation mode is to be set in the case of presence.

eEnergyLevel_AUTO_II

Automatic mode II means that in this operation mode the energy level switches directly from the Protection, Economy or PreComfort energy level to the Comfort energy level on recognition of presence. Please see diagram.

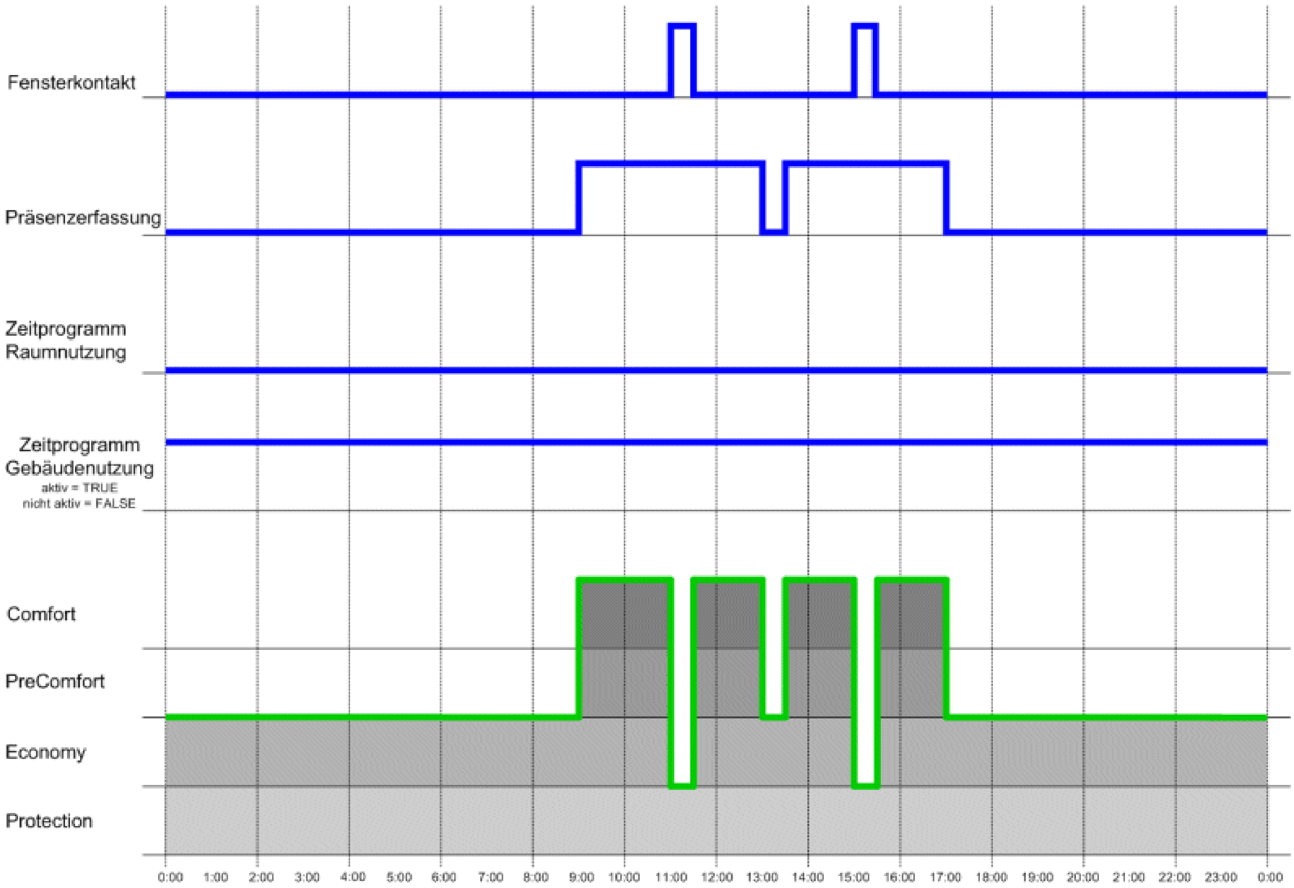


Fig. 78: Assignment of the energy level in automatic mode-II

6.2.1.3 FB_BC9191FanCoil

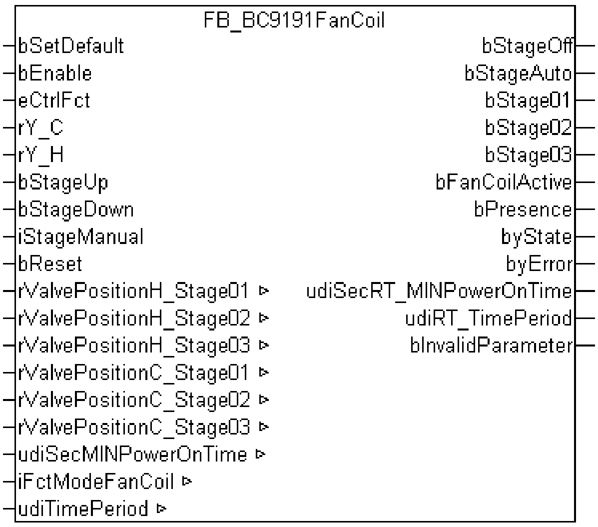


Fig. 79: Function block FB_BC9191FanCoil

Application

This function block maps a 3-stage fan. The three stages are switched via the control value (0...100%) of the heating or cooling controller. Furthermore there is a possibility to manually override the fan controller via the *iStageManual* or *bStageUp* or *bStageDown* input. A minimum switch-on time can be set via the *udiSecMINPowerOnTime* input, which is then valid for each stage.

VAR_INPUT

```
Name                        : Type
bSetDefault                 : BOOL;
bEnable                     : BOOL;
eCtrlFct                    : E_BC9191CtrlFct [▶ 90];
rY_C                        : REAL;
rY_H                        : REAL;
bStageUp                    : BOOL;
bStageDown                  : BOOL;
iStageManual                : INT;
bReset                      : BOOL;
```

bSetDefault: If the variable is TRUE, the default values of the VAR_IN_OUT variables are adopted.

bEnable: The function block is activated if the variable *bEnable* is TRUE. No fan stage is activated with a FALSE.

eCtrlFct: This input is connected to the *eCtrlFct* output of the **FB_BC9191FctSelection**. This information is important in order to know whether the plant is in heating or cooling mode. In automatic mode, the fan stages are only activated if

- the system is in heating mode and according to the control deviation the request for heating is active or
- the system is in cooling mode and according to the control deviation the request for cooling is active.

rY_C: Input for the control value from the cooling controller.

rY_H: Input for the control value from the heating controller.

bStageUp: Local adjustment of the fan stage, stepwise increase by push button.

bStageDown: Local adjustment of the fan stage, stepwise decrease by push button.

iStageManual: The manual fan stage can/is set centrally via this input.
iStageManual: = 0 corresponds to fan stage OFF
iStageManual: = 1 corresponds to fan stage AUTO
iStageManual: = 2 corresponds to fan stage01 active
iStageManual: = *3*, corresponds to fan stage02 active
iStageManual: = *4*, corresponds to fan stage03 active

bReset: Acknowledge input in case of a fault or an incorrect parameter.

VAR_OUTPUT

```
Name                          : Type
bStageOff                     : BOOL;
bStageAuto                    : BOOL;
bStage01                      : BOOL;
bStage02                      : BOOL;
bStage03                      : BOOL;
bFanCoilActive                : BOOL;
bPresence                     : BOOL;
byState                       : BYTE;
byError                       : BYTE;
udiSecRT_MINPowerOnTime       : UDINT;
udiRT_TimePeriod              : UDINT;
bInvalidParameter             : BOOL;
```

bStageOff: TRUE, fan stages are switched off.

bStageAuto: TRUE, fan controller is in automatic mode.

bStage01: TRUE, fan stage01 active.

bStage02: TRUE, fan stage02 active.

bStage03: TRUE, fan stage03 active.

bFanCoilActive: TRUE if one of the three fan stages is active. This output can be/is used to enable controllers in order to avoid a build up of heat or cold.

bPresence: TRUE means that presence was detected via the *bStageUp*, *bStageDown* or *iStageManual* inputs.

byState: Indicates the state of the fan controller.
byState.0:= Function block is activated
byState.3:= Manual fan stage setting is active
byState.4:= bReset
byState.5:= Fan stage01 active
byState.6:= Fan stage02 active
byState.7:= Fan stage03 active

byError: Output of the errors as byte.
byError.1:= bInvalidParameter

udiSecRT_MINPowerOnTime: Indicates the remaining time of the minimum switch-on time.

udiRT_TimePeriod: Indicates the remaining time of the manual override.

bInvalidParameter: Indicates that an incorrect input parameter is present. *bInvalidParameter* must be acknowledged with *bReset*.

VAR_IN_OUT

```
Name                          : Type     Persistent      Default
rValvePositionH_Stage01     : REAL;      X               0          %
rValvePositionH_Stage02     : REAL;      X              21          %
rValvePositionH_Stage03     : REAL;      X              76          %
rValvePositionC_Stage01     : REAL;      X               0          %
rValvePositionC_Stage02     : REAL;      X              21          %
rValvePositionC_Stage03     : REAL;      X              76          %
udiSecMINPowerOnTime        : UDINT;     X             120         sec
iFctModeFanCoil             : INT;       X               3
udiTimePeriod               : UDINT;     X              60  min
```

rValvePositionH_Stage01: Limit value of the control value of the heating controller from which fan stage01 is switched on.

rValvePositionH_Stage02: Limit value of the control value of the heating controller from which fan stage02 is switched on.

rValvePositionH_Stage03: Limit value of the control value of the heating controller from which fan stage03 is switched on.

rValvePositionC_Stage01: Limit value of the control value of the cooling controller from which fan stage01 is switched on.

rValvePositionC_Stage02: Limit value of the control value of the cooling controller from which fan stage02 is switched on.

rValvePositionC_Stage03: Limit value of the control value of the cooling controller from which fan stage03 is switched on.

udiSecMINPowerOnTime: Minimum switch-on time that a fan must run for in a stage before switching to a different stage or switching off. Input in seconds (e.g. 120 corresponds to 120 s).

iFctModeFanCoil: The user has the possibility to activate the fan controller for heating mode or cooling mode or both modes via the valence of this variable. Valid values are 1, 2 or 3. Other values are invalid and bInvalidParameter is set to TRUE.

Cooling	Heating	Valence
0	1	1 (= fan controller active in heating mode)
1	0	2 (= fan controller active in cooling mode)
1	1	3 (= fan controller active in heating mode and cooling mode)

udiTimePeriod: Timeframe during which the manual override is active in case of presence. Specified in minutes

6.2.1.4 E_BC9191CtrlFct

```
TYPE E_BC9191CtrlFct:
(
     eCtrlFct_Off         := 0,
     eCtrlFct_Heating     := 1,
     eCtrlFct_Cooling     := 2
);
END_TYPE
```

eCtrlFct_Off

Function selection OFF active.

eCtrlFct_Heating

Function selection for heating operation active.

eCtrlFct_Cooling

Function selection for cooling operation active.

6.2.1.5 FB_BC9191FctSelection

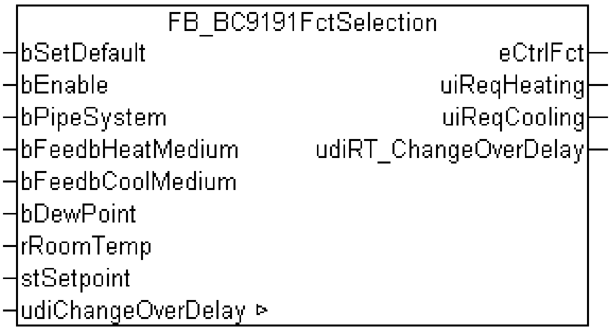


Fig. 80: Function block FB_BC9191FctSelection

Application

This function block is for enabling room heating or room cooling. It can be used for 2-wire pipe systems (changeover) or 4-wire pipe systems.
In the case of a 4-wire pipe system the changeover from heating to cooling operation takes place automatically on the basis of a comparison of the setpoint for the room temperature with the actual value for the room temperature.

Sketch:

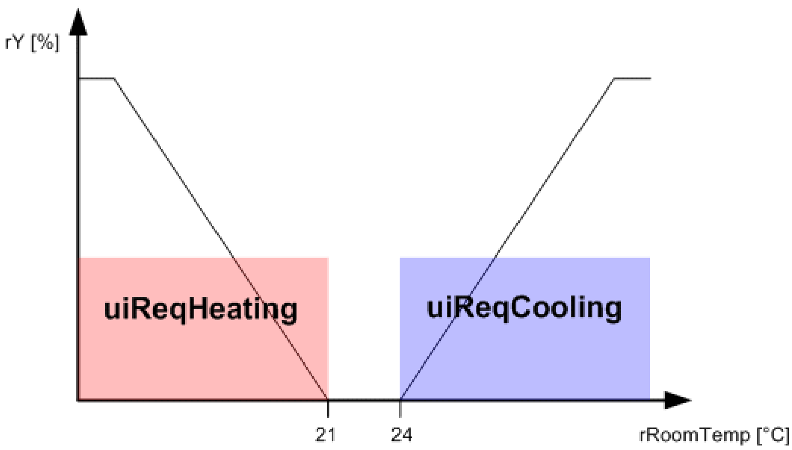


Fig. 81: Switching between heating and cooling mode

In the case of a 2-wire pipe system the heating operation or cooling operation may only be enabled when heating or cooling medium is present. The room temperature controller gets this information from the primary units.

In both 2-wire and 4-wire pipe systems the changeover between heating operation and cooling operation can be delayed by a timer. The input variable *udiChangeOverDelay* must be greater than zero for this.

The following tables describe the interrelationship between the inputs and the eCtrlFct output of the FB_BC9191FctSelection function block.

In the 2-wire pipe system

bEnable	bPipeSystem	bFeedbHeatMedium	bFeedbCoolMedium	interim result	bDewPoint	eCtrlFct
0	0	0	0	OFF	TRUE / FALSE	OFF
1	0	0	0	Heating	TRUE / FALSE	Heating
1	0	0	1	Cooling	TRUE	OFF
					FALSE	Cooling
1	0	1	0	Heating	TRUE / FALSE	Heating
1	0	1	1	Heating	TRUE / FALSE	Heating

In the 4-wire pipe system

bEnable	bPipeSystem	T_Room ≤ Tsetpoint	T_Room > Tsetpoint	interim result	bDewPoint	eCtrlFct
0	1	0	0	OFF	TRUE / FALSE	OFF
1	1	0	1	Cooling	TRUE	OFF
					FALSE	Cooling
1	1	1	0	Heating	TRUE / FALSE	Heating
1	1	1	1	Heating	TRUE / FALSE	Heating

VAR_INPUT

```
Name                     : Type
bSetDefault              : BOOL;
bEnable                  : BOOL;
bPipeSystem              : BOOL;
bFeedbHeatMedium         : BOOL;
bFeedbCoolMedium         : BOOL;
bDewPoint                : BOOL;
rRoomTemp                : REAL;
stSetpoint               : ST_BC9191SetpointRoom [▸ 97];
```

bSetDefault: If the variable is TRUE, the default values of the VAR_IN_OUT variables are adopted.

bEnable: TRUE enables the function block. If FALSE the function block is disabled and *eCtrlFct := eCtrlFct_Off*.

bPipeSystem: FALSE means that a 2-wire pipe system is present. TRUE means that a 4-wire pipe system is present.

bFeedbHeatMedium: Signal from the power generation or distribution that heating medium is available.

bFeedbCoolMedium: Signal from the power generation or distribution that cooling medium is available.

bDewPoint: The dew point sensor is connected to this input. If this is triggered the cooling control function is deactivated and *eCtrlFct:= eCtrlFct_Off* is set.

rRoomTemp: This input variable transfers the current room temperature to the function block.

stSetpoint: STRUCTURE containing the setpoints for the individual energy levels.

VAR_OUTPUT

```
Name                     : Type
eCtrlFct                 : E_BC9191CtrlFct [▶ 90];
uiReqHeating             : UINT;
uiReqCooling             : UINT;
udiRT_ChangeOverDelay    : UDINT;
```

eCtrlFct: This output contains the current control function.

uiReqHeating: Is 1 if the room/zone requests heating energy. It is 0 if there is no heating requirement.

uiReqCooling: Is 1 if the room/zone requests cooling energy. It is 0 if there is no cooling requirement.

udiRT_ChangeOverDelay: Indicates the time remaining until the active control function is changed over.

VAR_IN_OUT

```
Name                   : Type          Persistent      Default
udiChangeOverDelay     : UDINT;            x           0
```

udiChangeOverDelay: Changeover time between the control functions. Must be specified in seconds. If the input is greater than 0 it will always be observed. The variable must be 0 if there is to be no changeover time between the control functions.

6.2.1.6 FB_BC9191SetpointRoom

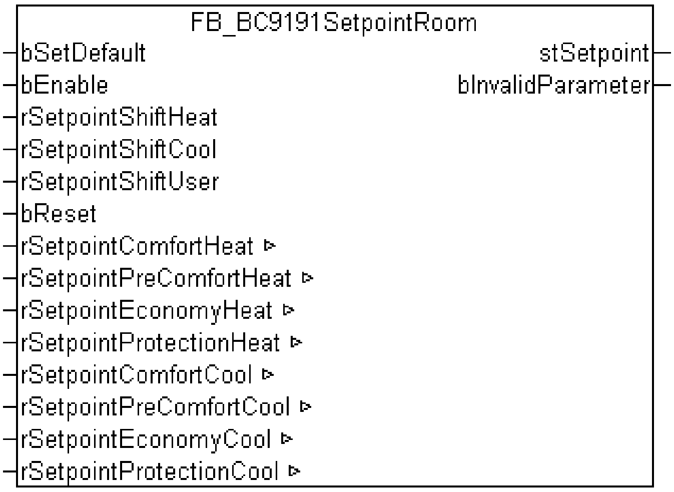


Fig. 82: Function block FB_BC9191SetpointRoom

Application

The function block FB_BC9191SetpointRoom assigns setpoints for cooling and heating operation to each of the energy levels Protection, Economy, PreComfort and Comfort. In connection with the function block FB_BC9191EnergyLevel the room temperature controllers are assigned an energetically optimum setpoint in accordance with the use of the room and the selected heating or cooling operation.

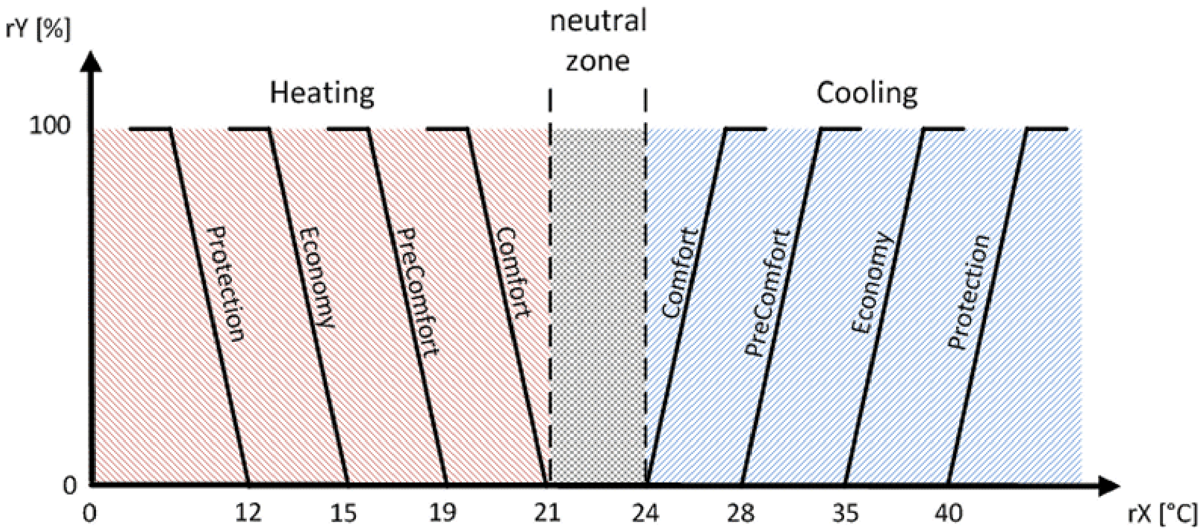


Fig. 83: Assignment of energy levels

The resulting setpoint for the different energy levels is made up of:

1. the base setpoint value
2. the local setpoint value shift (not in the case of the Protection setpoints)
3. the central setpoint value shift (not in the case of the Protection setpoints)

The local shift due to a room setpoint generator and also the remote adjustment of the setpoints by a building management system only affect the Comfort and PreComfort energy levels.

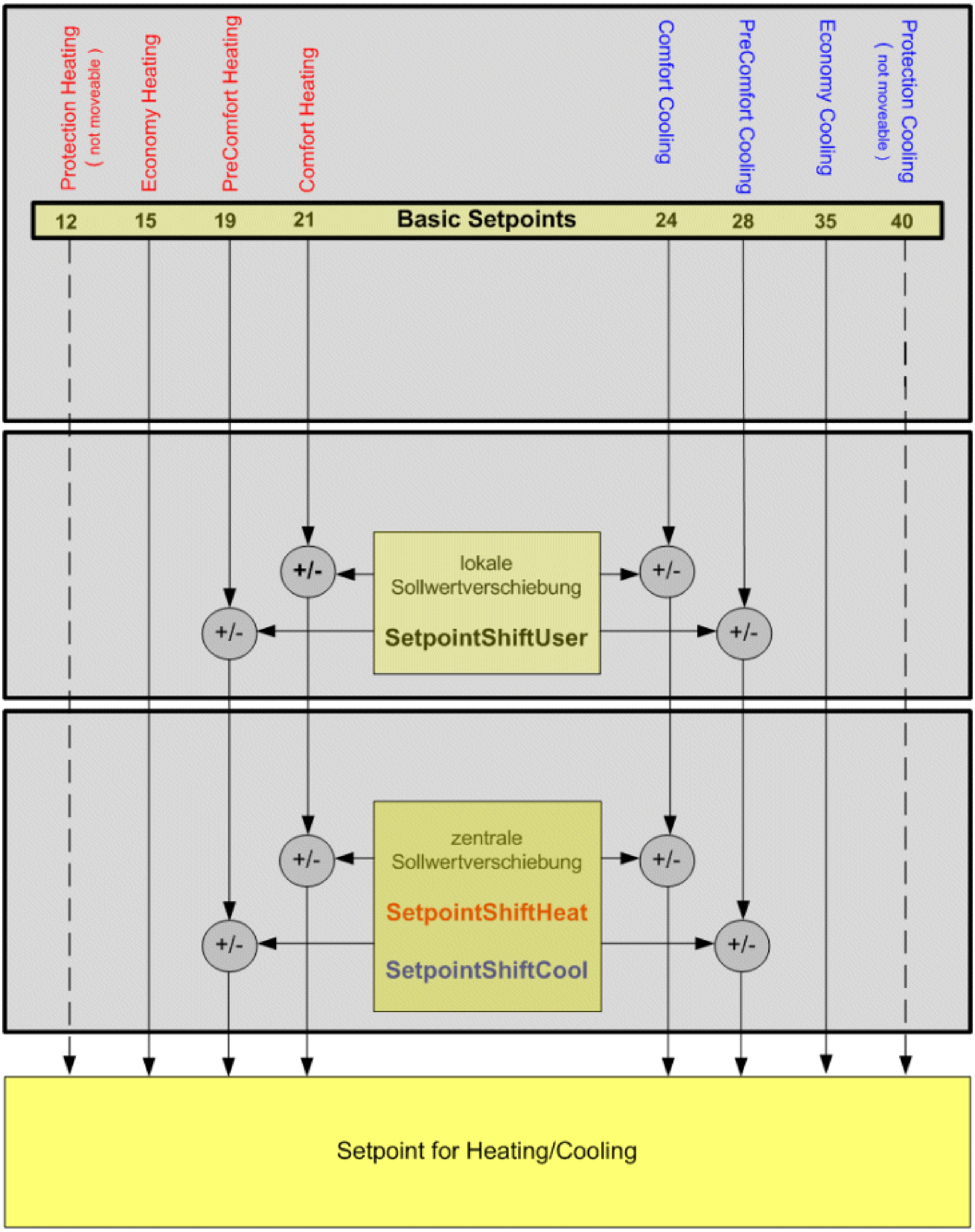


Fig. 84: Mode of operation Setpoint shift

VAR_INPUT

```
Name                    : Type

bSetDefault             : BOOL;
bEnable                 : BOOL;
rSetpointShiftHeat      : REAL;
rSetpointShiftCool      : REAL;
rSetpointShiftUser      : REAL;
bReset                  : BOOL;
```

bSetDefault: If the variable is TRUE, the default values of the VAR_IN_OUT variables are adopted.

bEnable: The function block is activated if the variable *bEnable* is TRUE.

rSetpointShiftHeat: The rSetpointShiftHeat variable is used for the adaptation of the ComfortHeating setpoint of the building management system.

If the ComfortHeating setpoint is raised, then the setpoints for the ComfortCooling and PreComfortCooling are also increased.
Example:

Energy levels		Protection-Heating	Economy-Heating	PreComfortHeating	ComfortHeating	Comfort-Cooling	PreComfort-Cooling	Economy-Cooling	ProtectionCooling
Base setpoint value [°C]		12	15	19	21	24	28	35	40
SetpointShiftHeat [K]	+3	-	-	-	+3	+3	+3	-	-
Resulting setpoint [°C]		12	15	19	24	27	31	35	40

A lowering of the ComfortHeating setpoint affects only the ComfortHeating and PreComfortHeating setpoints.
Example:

Energy levels		Protection-Heating	Economy-Heating	PreComfortHeating	ComfortHeating	Comfort-Cooling	PreComfort-Cooling	Economy-Cooling	Protection-Cooling
Base setpoint value [°C]		12	15	19	21	24	28	35	40
SetpointShiftHeat [K]	-3	-	-	-3	-3	-	-	-	-
Resulting setpoint [°C]		12	15	16	18	24	28	35	40

rSetpointShiftCool: The rSetpointShiftCoolvariable is used for the adaptation of the ComfortCooling setpoint of the building management system.

If the ComfortCooling setpoint is raised, then the setpoint for the PreComfortCooling is also raised.
Example:

Energy levels		Protection-Heating	Economy-Heating	PreComfortHeating	ComfortHeating	Comfort-Cooling	PreComfort-Cooling	Economy-Cooling	Protection-Cooling
Base setpoint value [°C]		12	15	19	21	24	28	35	40
SetpointShiftCool [K]	+3	-	-	-	-	+3	+3	-	-
Resulting setpoint [°C]		12	15	19	21	27	31	35	40

A lowering of the ComfortCooling setpoint only affects the ComfortCooling. The PreComfortCooling is not changed.
Example:

Energy levels		Protection-Heating	Economy-Heating	PreComfortHeating	ComfortHeating	Comfort-Cooling	PreComfort-Cooling	Economy-Cooling	Protection-Cooling
Base setpoint value [°C]		12	15	19	21	24	28	35	40
SetpointShiftCool [K]	-3	-	-	-	-	-3	-	-	
Resulting setpoint [°C]		12	15	19	21	21	28	35	40

If the setpoint of the PreComfort energy level is shifted beyond the setpoint of the Economy level, then the setpoint of the Economy level adopts the value of the PreComfort level.
Example:

Energy levels		**Protection-Heating**	**Economy-Heating**	**PreCom-fortHeat-ing**	**Com-fortHe-ating**	**Comfort-Cooling**	**PreComfort-Cooling**	**Economy-Cooling**	**Protection-Cooling**
Base setpoint value [°C]		12	15	19	21	24	28	35	40
SetpointShift-Cool [K]	+8	-	-	-	-	+8	+8	-	-
Resulting set-point [°C]		12	15	19	21	32	36	36	40

rSetpointShiftUser: The variable rSetpointShiftUser is used for local setpoint shifting of the user. A positive value of rSetpointShiftUser affects the setpoint of ComfortHeating, ComfortCooling and PreComfortColling.
Example:

Energy levels		**Protection-Heating**	**Economy-Heating**	**PreCom-fortHeat-ing**	**Com-fortHe-ating**	**Comfort-Cooling**	**PreComfort-Cooling**	**Economy-Cooling**	**Protection-Cooling**
Base setpoint value [°C]		12	15	19	21	24	28	35	40
SetpointShif-tUser [K]	+3	-	-	-	+3	+3	+3	-	-
Resulting set-point [°C]		12	15	19	24	27	31	35	40

A negative value of rSetpointShiftUser affects the setpoints of PreComfortHeating, ComfortHeating and ComfortCooling.
Example:

Energy levels		**Protection-Heating**	**Economy-Heating**	**PreCom-fortHeat-ing**	**Com-fortHe-ating**	**Comfort-Cooling**	**PreComfort-Cooling**	**Economy-Cooling**	**Protection-Cooling**
Base setpoint value [°C]		12	15	19	21	24	28	35	40
SetpointShif-tUser [K]	-3	-	-	-3♠	-3	-3	-	-	-
Resulting set-point [°C]		12	15	16	18	21	28	35	40

bReset: Acknowledge input in case of a fault or an incorrect parameter.

VAR_OUTPUT

```
Name                    : Type
stSetpoint              : ST_BC9191SetpointRoom [▸ 97];
bInvalidParameter       : BOOL;
```

stSetpoint: Structure containing the setpoints for all energy levels.

bInvalidParameter: Indicates that an incorrect input parameter is present. *bInvalidParameter* must be acknowledged with *bReset*.

VAR_IN_OUT

```
Name                       : Type      Persistent    Default     Range
rSetpointComfortHeat       : REAL;         X           21.0
rSetpointPreComfortHeat    : REAL;         X           19.0
rSetpointEconomyHeat       : REAL;         X           15.0
rSetpointProtectionHeat    : REAL;         X           12.0
rSetpointComfortCool       : REAL;         X           24.0
rSetpointPreComfortCool    : REAL;         X           28.0
rSetpointEconomyCool       : REAL;         X           35.0
rSetpointProtectionCool    : REAL;         X           40.0
```

rSetpointComfortHeat: Setpoint for the Comfort heating energy level.

rSetpointPreComfortHeat: Setpoint for the PreComfort heating energy level.

rSetpointEconomyHeat: Setpoint for the Economy heating energy level.

rSetpointProtectionHeat: Setpoint for the Protection heating energy level.

rSetpointComfortCool: Setpoint for the Comfort cooling energy level.

rSetpointPreComfortCool: Setpoint for the PreComfort cooling energy level.

rSetpointEconomyCool: Setpoint for the Economy cooling energy level.

rSetpointProtectionCool: Setpoint for the Protection cooling energy level.

6.2.1.7 ST_BC9191SetpointRoom

```
TYPE ST_BC9191SetpointRoom :
STRUCT
  stSetpointRoom_ComfortHeat       : REAL:= 21.0;
  stSetpointRoom_PreComfortHeat    : REAL:= 19.0;
  stSetpointRoom_EconomyHeat       : REAL:= 15.0;
  stSetpointRoom_ProtectionHeat    : REAL:= 12.0;
  stSetpointRoom_ComfortCool       : REAL:= 24.0;
  stSetpointRooom_PreComfortCool   : REAL:= 28.0;
  stSetpointRoom_EconomyCool       : REAL:= 35.0;
  stSetpointRoom_ProtectionCool    : REAL:= 40.0;
END_STRUCT
END_TYPE
```

The values in the structure are defined with the preset values.

6.2.1.8 FB_BC9191PICtrl

Simple PI controller

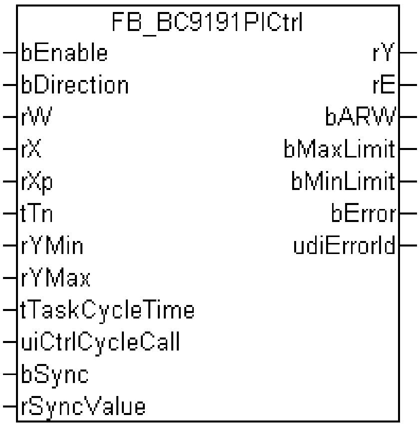


Fig. 85: Function block FB_BC9191PICtrl

This PI controller does not work directly with a gain factor K_p, but instead calculates this internally from the so-called proportional band (input *rXp*) in relation to the control value limits (*rYmin* and *rYmax*), from which K_p is then determined internally.

Setting via the proportional band

The adjustment of the gain factor K_p of a controller often harbors the difficulty for the user that there is no size reference to the application. If a heating controller normally operates within the two-figure range, then a flow rate controller can accept values in the five-figure range. It therefore makes sense to represent the K_p factor in such a way that it has a reference to the possible control deviation and change of control value. The P-part of the controller is regarded for the dimensioning of the K_p factor.
The equation for this is:

- Control value = control deviation x gain factor → $Y = E * K_p$

this relationship also applies to the changes in the control deviation and the control value:

- Change in control value = change in control deviation • gain factor → $\Delta Y = \Delta E * K_p$

Referenced to the minimum and maximum value of the control value, Y_{min} and Y_{max}:

- $Y_{max} - Y_{min} = (E(Y_{max}) - E(Y_{min})) * K_p$

This difference, $E(Y_{max})-E(Y_{min})$, is called the proportional band (Xp). Transposed, the equation is then:

- $K_p = (Y_{max} - Y_{min}) / X_p$

The following diagram clarifies the functional interrelationship:

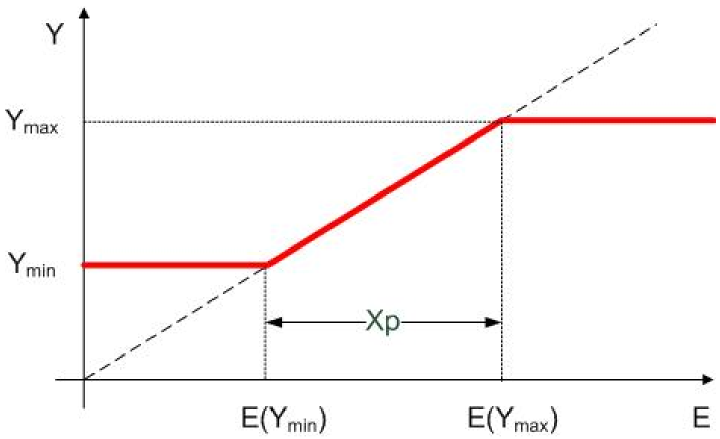


Fig. 86: Functional correlation Xp

The proportional band X_p therefore indicates the size of the range of the control deviation that leads to an output of Y_{min} to Y_{max} from the controller.

A smaller X_p leads to a steeper function and thus to an increase in the Kp factor. However, the control deviation limit values $E(Y_{max})$-$E(Y_{min})$ are shifted:

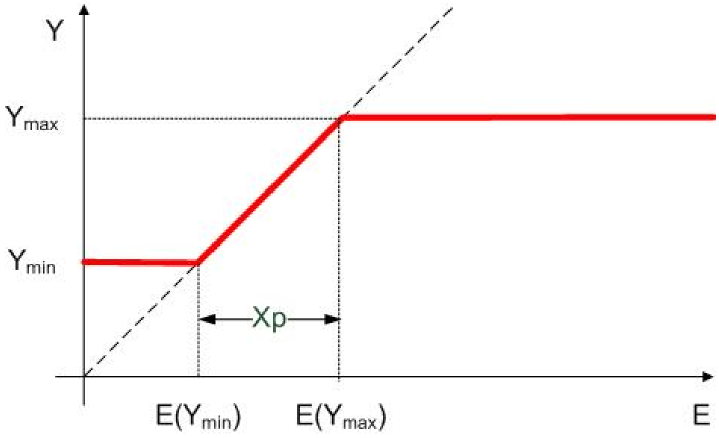


Fig. 87: Functional correlation small Xp

Functional description

Step response of a simple PI controller, where the control gain has no influence on the integral component. Response of output Y(t) to a control deviation jump by E: when the control deviation jumps by E, output Y first jumps to KP*E by the proportional component and then grows by a further E in each interval TN.
Note: The controller is designed in such a way that the controller starts at 0, i.e. without the KP*E jump, after a PLC reset or restart.

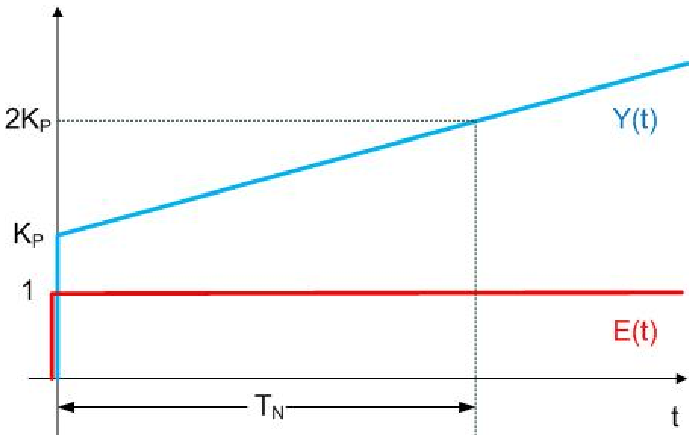


Fig. 88: Step response PI controller

Basic function

A TRUE signal at the *bEnable* input activates the function block. The internal control algorithm is now processed. The input value *uiCtrlCycleCall* specifies the number of PLC cycles after which the internal control algorithm is processed. If *uiCtrlCycleCall* =1, then the new calculation takes place in each PLC cycle; if, conversely, it is set to 100, then a new calculation of the output values takes place only every 100 PLC cycles. The PLC cycle time is also accounted for in the control value calculation. An incorrect input value leads to incorrect calculation.
The inputs *rW* (setpoint), *rX* (actual value), *rXp* (proportional band) and *rTn* (integral action time) are the input variables of the PI controller. They are used in each calculation cycle for the determination of the output values rY (control value) and *rE* (control deviation). The control value can additionally be limited by the inputs *rYMin* and *rYMax*.

Anti-Reset-Windup (ARW)

If the controller “runs” into this limit, then the I-component is held internally at the final value. If this were not done, then the I-component could adopt very large values without hindrance during the limit case, which would first have to be eliminated again in case of reversal of the direction of action of the controller. This function is called “Anti-Reset-Windup” (ARW). The output *bARW* is set if this function is active.

Special case: Tn=0 as cut-off of the I-component

From the above step response diagram it can be seen that the influence of the I-component becomes all the weaker, the larger the integral action time T_n is set. As the integral action time approaches infinity, the influence of the I-component is virtually zero. Conversely, an increasingly smaller integral action time allows the influence of the I-component to grow; at T_n=0 the control value would approach infinity. However, this special case is used to **cut off** the I-component. This is an internally formed exception, since the integral action time belongs directly to the I-component and should also figuratively result in cut-off due to the zero entry.

Synchronization

A positive edge on *bSync* sets the controller output *rY* directly to *rSyncValue*, provided that the controller has been activated by a TRUE signal on *bEnable*. If this is not the case the positive edge on *bSync* is ignored.

Error case/block not activated

If the controller is incorrectly parameterized processing is stopped, the *bError* output is set and the corresponding error ID is output at *udiErrorID*. The function block is also stopped if the input *bEnable* is not set. In both cases the outputs are set as follows:

rY	0.0
rE	0.0
bARW	FALSE
bMaxLimit	FALSE
bMinLimit	FALSE

VAR_INPUT

```
bEnable          : BOOL;
bDirection       : BOOL;
rW               : REAL;
rX               : REAL;
rXp              : REAL;
tTn              : TIME;
rYMin            : REAL;
rYMax            : REAL;
tTaskCycleTime   : TIME;
uiCtrlCycleCall  : UINT;
bSync            : BOOL;
rSyncValue       : REAL;
```

bEnable: The function block is active as long as this input is TRUE. A FALSE signal, on the other hand, stops the processing.

bDirection: The control direction of the controller can be changed with the parameter *bDirection*. If *bDirection* is TRUE, the direct control direction for cooling operation of the controller is active.
If *bDirection* is FALSE, the indirect control direction of the controller is activated for heating operation.

rW: setpoint.

rX: actual value.

rXp: proportional band.

tTn: integral action time in seconds. The I-component corrects the residual control deviation following correction of the P-part. The smaller tTi is set, the faster the controller corrects. The control loop becomes unstable if the time is too short. Larger tTi-times must be entered in order to reduce the integration component. The integral action time should be selected to be longer than the stroke time of the valve or damper drive. A zero value at this input deactivates the I-component.

rYMin: Lower limit of the control value.

rYMax: Upper limit of the control value.

tTaskCycleTime: Cycle time of the task in which the function block runs.

uiCtrlCycleCall: Call cycle of the function block as a multiple of the cycle time.
Example: *tTaskCycleTime* = 20 ms, *uiCtrlCycleCall* =10 -> The control algorithm is called every 200 ms. Thus the outputs are also updated only every 200 ms.

bSync,rSyncValue: A rising edge at this input sets the (internal) I-component such that *rSyncValue* is output at the control value output. If the I-component is deactivated by *tTn*=0ms, however, then this command is ignored.

VAR_OUTPUT

```
rY          : REAL;
rE          : REAL;
rEMin       : REAL;
bEMax       : REAL;
bARW        : BOOL;
bMaxLimit   : BOOL;
bMinLimit   : BOOL;
bError      : BOOL;
udiErrorId  : UDINT;
```

rY: Control value.

rE: Control deviation (setpoint/actual value).

rEMin: Lower control deviation limit value, which results from the input proportional band.

rEMax: Upper control deviation limit value, which results from the input proportional band.

bARW: Anti-Reset-Windup function is active.

bMaxLimit: The control value has reached its upper limit value.

bMinLimit: The control value has reached its lower limit value.

bError: This output is switched to TRUE if the parameters entered are erroneous.

udiErrorId: Contains the error code if the values entered should be erroneous.

6.2.2 ADS access of a higher-level controller

The building automation room controller supports the Beckhoff ADS and Modbus TCP protocols.
The TCP port number for the ADS protocol is 48898. The UDP port number for the ADS protocol is 48899.
The TCP port number of Modbus TCP has been standardized to the value 502

Access via the Beckhoff ADS protocol is described in this part of the documentation. ADS is the abbreviation for **A**utomation **D**evice **S**pecification and describes a device and fieldbus independent interface. The Beckhoff ADS protocol is based on the TCP/IP or UDP/IP protocol. It allows the user within the Beckhoff system to use almost any connecting route to communicate with all the connected devices and to parameterize them. ADS function blocks can be used in TwinCAT PLC Control for this. The function blocks are contained in the *TcSystem.lib* library.

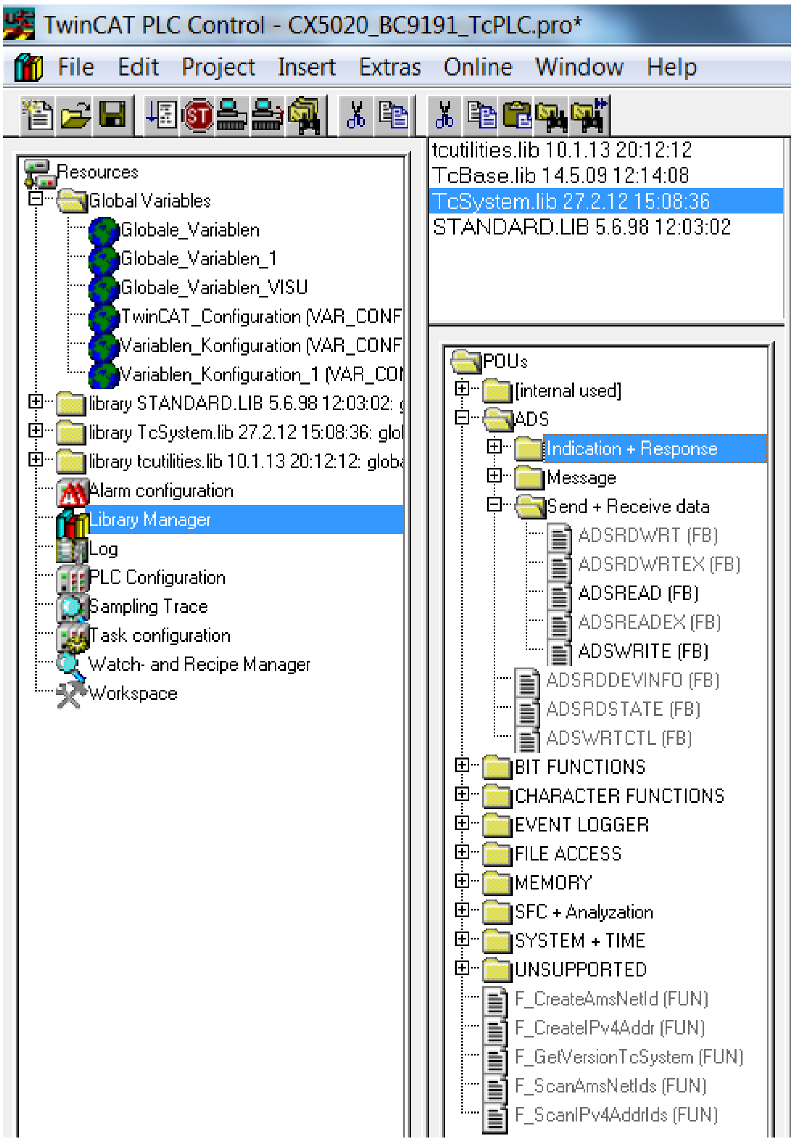


Fig. 89: TcSystem.lib library

Structure of the ADS protocol

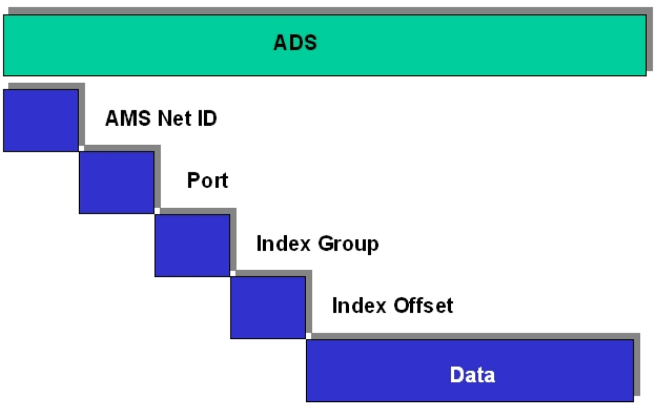


Fig. 90: Structure of the ADS protocol

AMSNetID

The AMSNetID provides a reference to the device that is to be addressed. This is created from the set TCP/IP address and an additional 2 bytes. These additional 2 bytes consist of "1.1" and cannot be changed. Example from a BC9191:

IP address 172.16.21.10
AMSNetID 172.16.21.10.1.1

Port number

The port number distinguishes sub-elements in the connected device.
Port 800: local process data (e.g. BC9191)

Index group

The index group distinguishes different data within a port.
0x4020 flag
0x4021 flag bit

Index offset (value range)

Indicates the offset, from which reading or writing the byte is to start.
Flag 0...4095
Flag bit 0...32760

Sample flag: Port 800 / Index Group 16#4020 / Byte Offset 100

```
10:
    fbADSWriteEL(    NETID:= stAMSNetId_Target,
                     PORT:= 800 ,
                     IDXGRP:= 16#4020,
                     IDXOFFS:= 100,
                     LEN:=SIZEOF(stBC9191EnergyLevelParameters.eCtrlMode) ,
                     SRCADDR:=ADR(stBC9191EnergyLevelParameters.eCtrlMode),
                     WRITE:=TRUE ,
                     TMOUT:=t#2s );
    IF NOT fbADSWriteEL.BUSY THEN
        IF fbADSWriteEL.ERR THEN
            bError:= fbADSRead.ERR;
            udiErrorId:= fbADSRead.ERRID;
            iStepWriteELParameter:= 1111;
        END_IF
        iStepWriteELParameter:=20;
        fbADSWriteEL(Write:= FALSE);
    END_IF
```

Fig. 91: Sample: writing flags with the FB ADSWrite

Sample Flag Bit: Port 800 / Index Group 16#4021 / Bit Offset 792

```
20:
    fbADSWriteEL(    NETID:= stAMSNetId_Target,
                     PORT:= 800 ,
                     IDXGRP:= 16#4021,
                     IDXOFFS:= 792,
                     LEN:=SIZEOF(bSetParameterEnergyLevel) ,
                     SRCADDR:=BITADR(bSetParameterEnergyLevel),
                     WRITE:=TRUE ,
                     TMOUT:=t#2s );
    IF NOT fbADSWriteEL.BUSY THEN
        IF fbADSWriteEL.ERR THEN
            bError:= fbADSRead.ERR;
            udiErrorId:= fbADSRead.ERRID;
            iStepWriteELParameter:= 1111;
        END_IF
        iStepWriteELParameter:=0;
        fbADSWriteEL(Write:= FALSE);
    END_IF
```

Fig. 92: Sample: writing flag bits with the FB ADSWrite

6.2.3 BC9191 in Master /Slave mode

Requirements

- the basic application program is installed on the BC9191 (slave) and the default configuration is active
- an application that takes over the control and regulation algorithms of all connected BC9191 in master/slave mode is installed on the higher-level controller – in this example a CX5020 (master). A TwinCAT configuration is active on the master controller.

Basic information

On delivery (default configuration) of the BC9191, the onboard I/Os have fixed addresses. The data for Ethernet communication start at the address offset 1000dec; this is a performance feature of the BC9191. This performance feature is used for the master/slave communication.

Inputs	Outputs
Bus Terminal %IB0 ...	Bus Terminal %QB0 ...
Ethernet DATA (PLC variables) %IB1000 ...(Modbus TCP/ADS-TCP/ADS-UDP)	Ethernet DATA (PLC variables) %QB1000 ... (Modbus TCP/ADS-TCP/ADS-UDP)
... %IB2047 maximum	... %QB2047 maximum

Detailed description of the TwinCAT configuration of the master controller

- generate a new TSM file and save it under MasterSlave_TcSM.tsm
- Append Device ==> virtual Ethernet interface

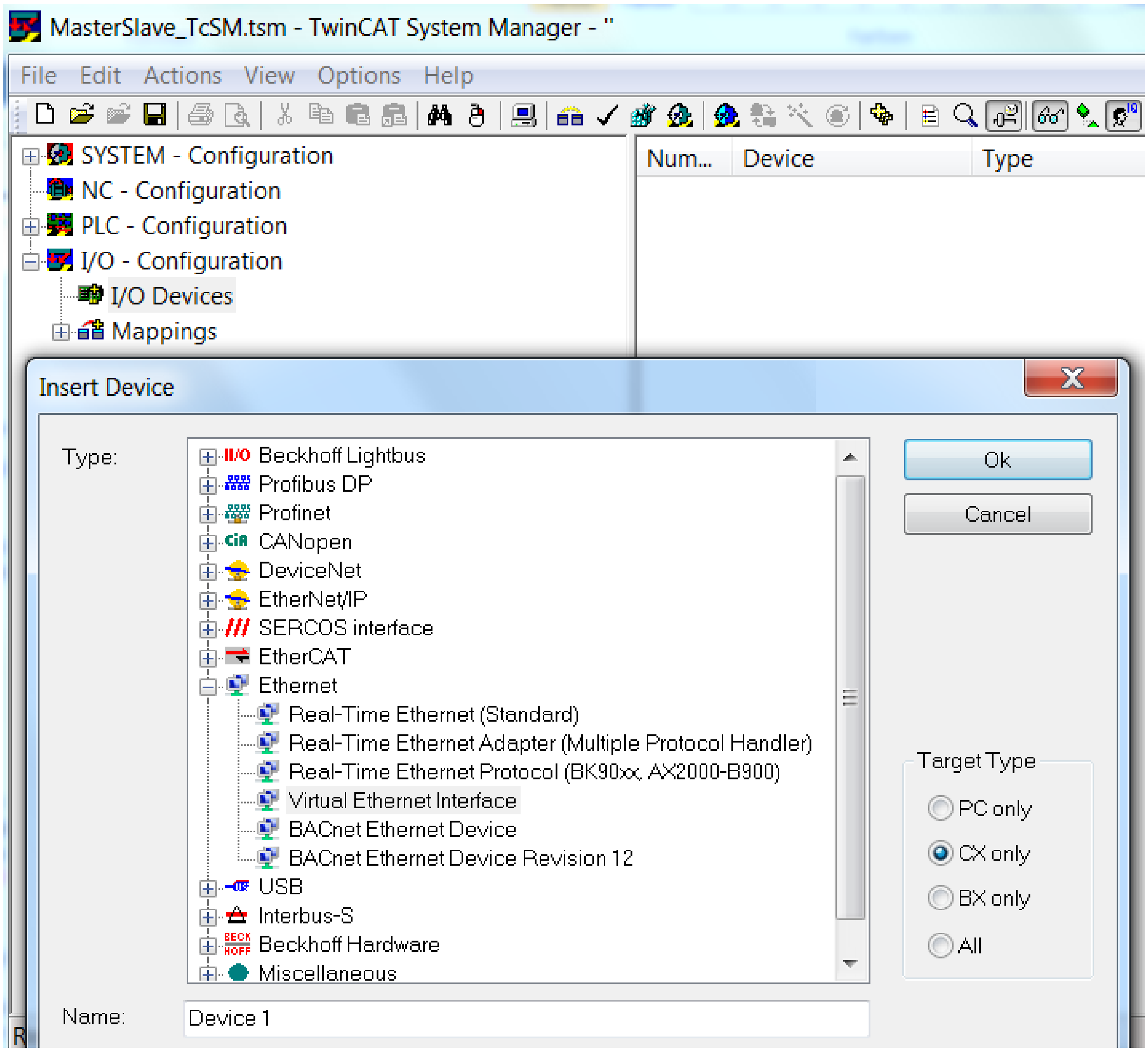


Fig. 93: Selection of the virtual Ethernet interface

- Append Box ==> select BC9191 and confirm with OK /==> do not open a template, confirm with the Cancel button instead

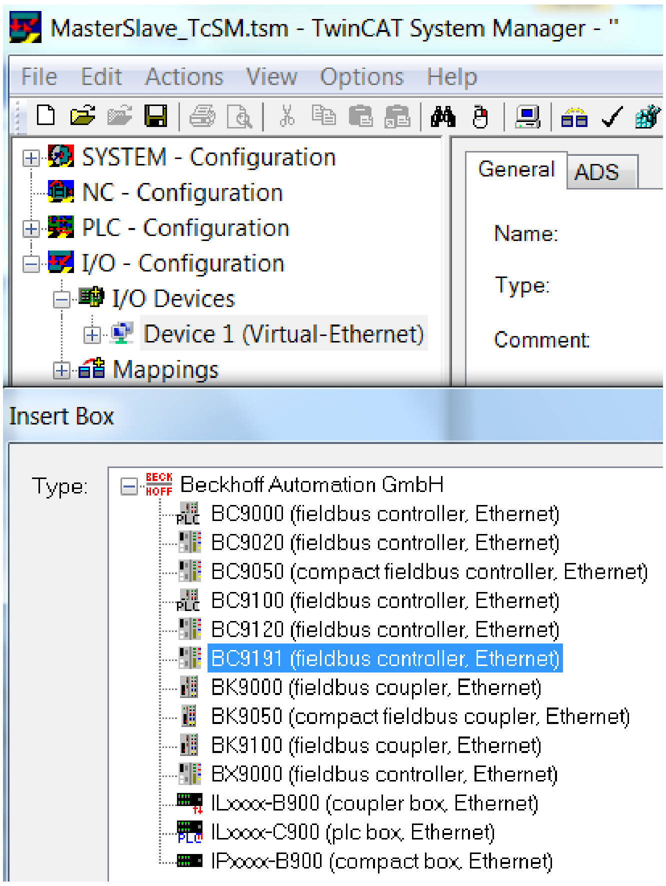


Fig. 94: Add BC9191 as BOX

- Add a PLC project so that the self-defined variable types are available.

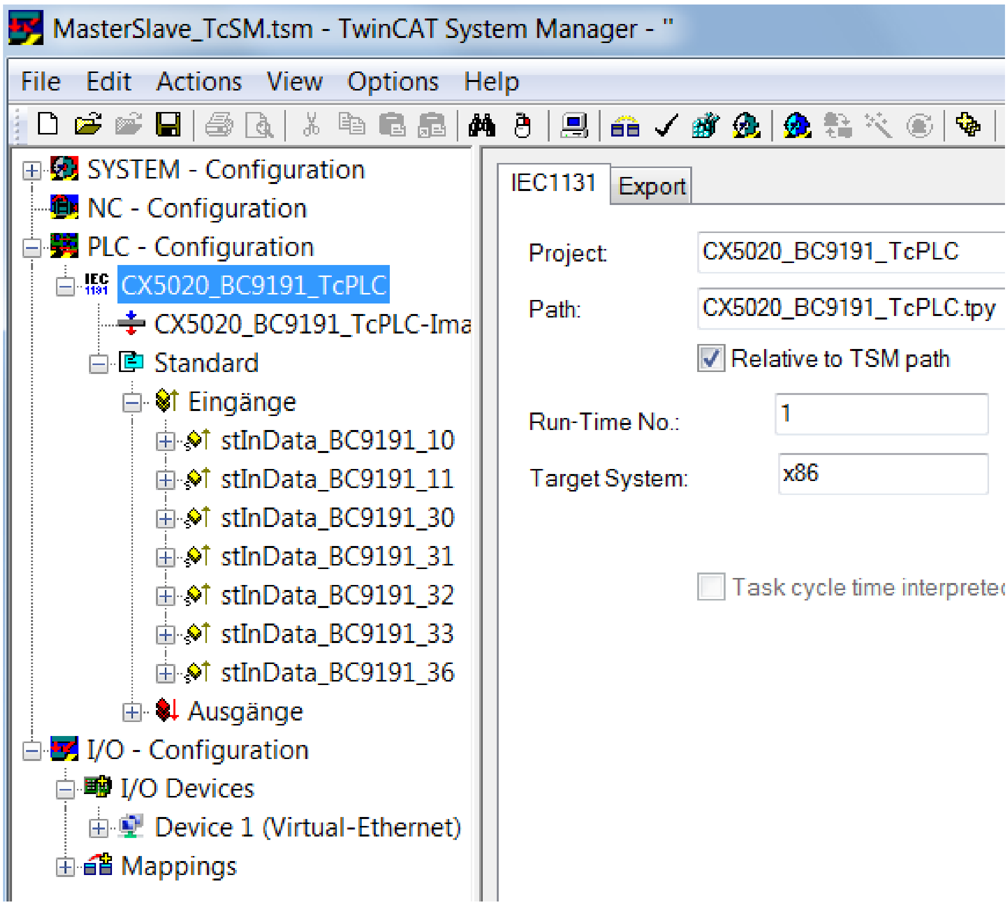


Fig. 95: Add PLC project

- insert PLC variables (for Ethernet communication) ==> inputs and outputs of the type ST_BC9191InData/ST_BC9191OutData under Box1 BC9191
- Edit a unique name, e.g. stInData_BC9191_10 and **stOutData** _BC9191_*10* (reference to the **name of the structure** and to the BC9191 (e.g. 172.16.21.*10*))

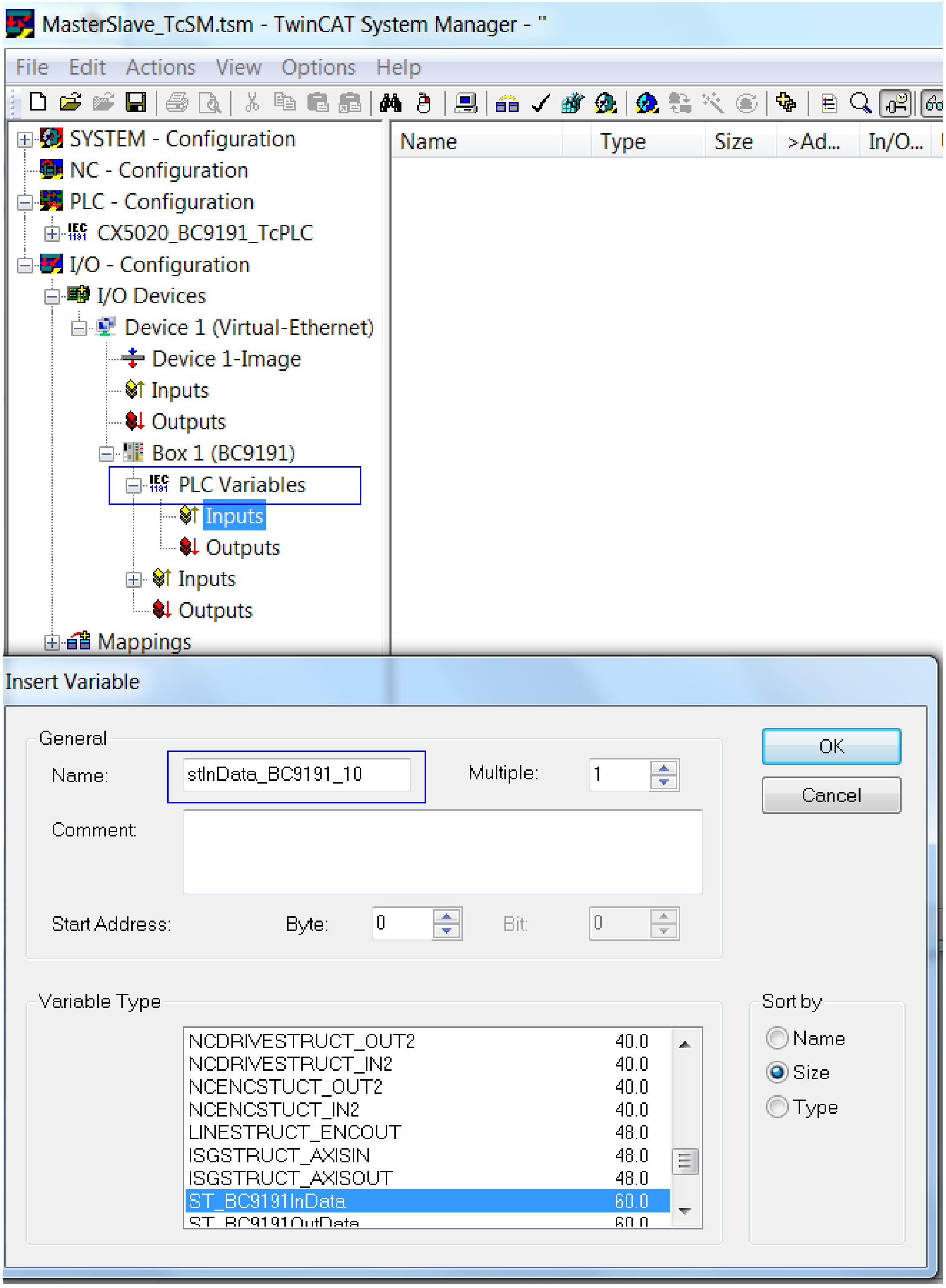


Fig. 96: Set assignable name

- Parameterization of Box 1 (BC9191)
- Declare a unique /assignable name e.g. BC9191_10
- Make communication settings in the IP address tab ==> adjust UDP and set the IP address

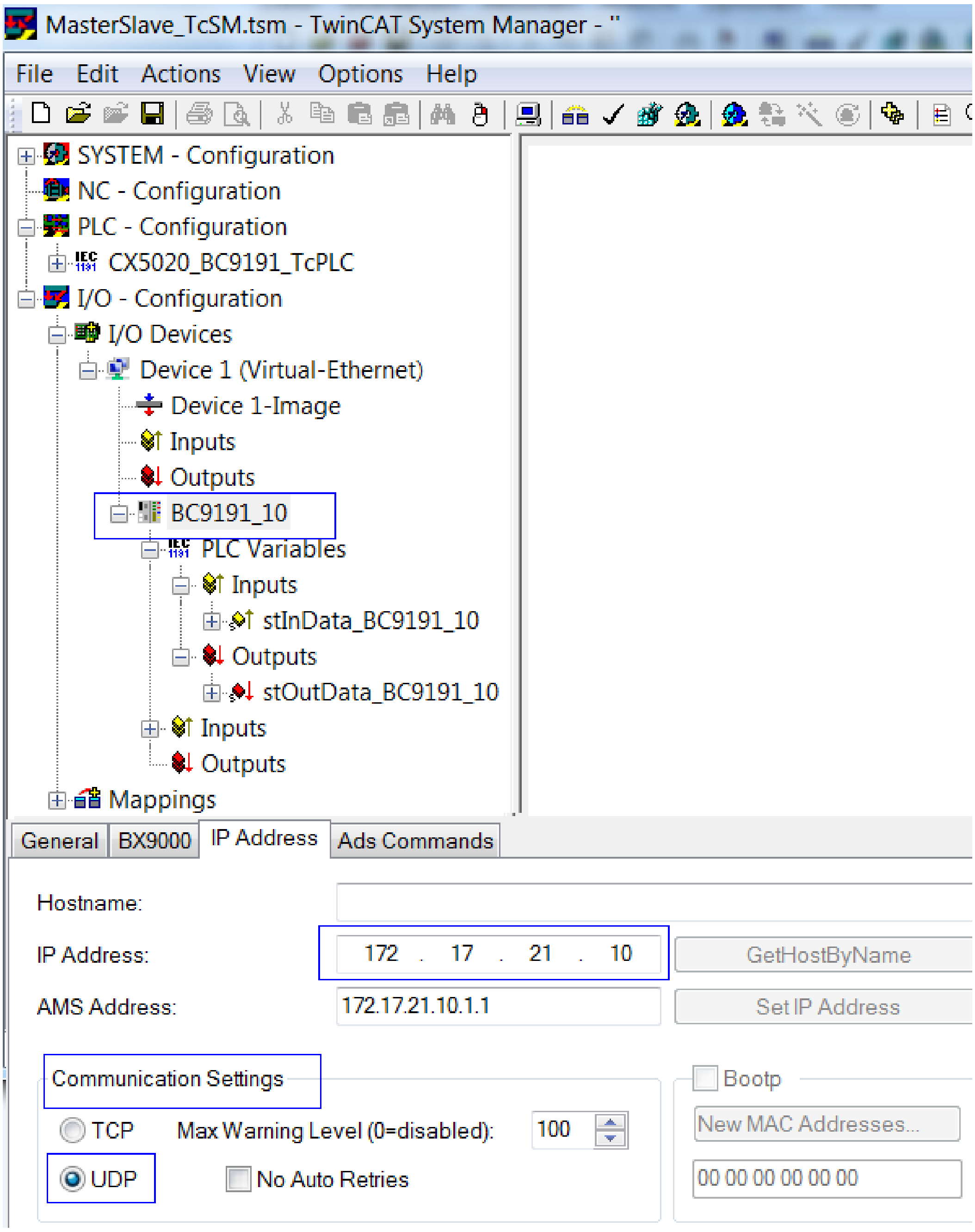


Fig. 97: Set communication setting to UDP

- If the parameterization of Box1 BC9191 is complete, it can be marked and copied as necessary.
- The respective name, the IP address and the names of the additional inputs and outputs must then be edited.

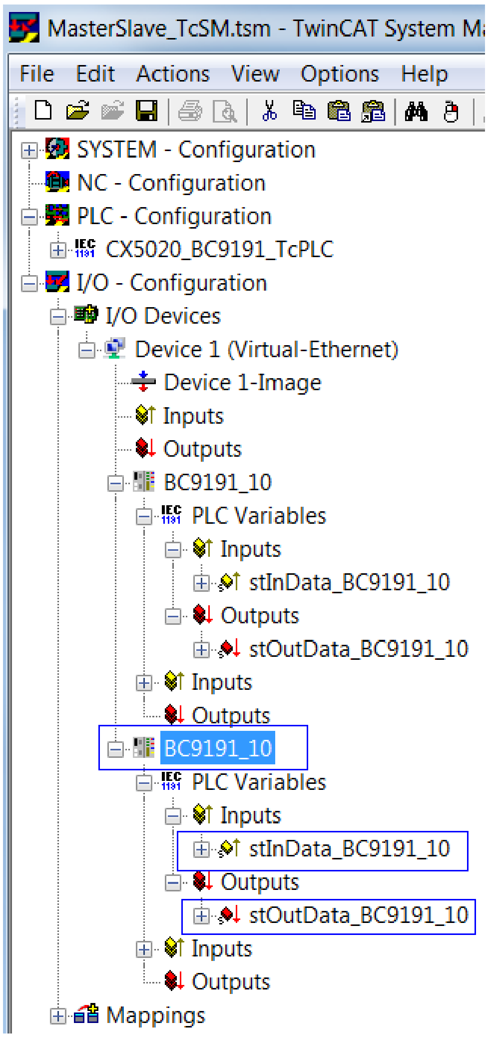


Fig. 98: Edit name of inputs and outputs

- Once the parameterization of the added BC9191s is complete, the inputs and outputs of the PLC task are linked to the inserted PLC variables

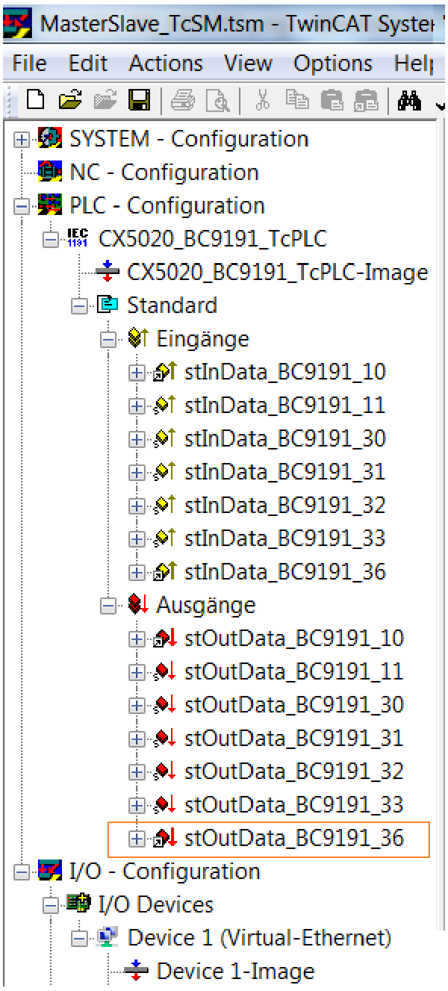


Fig. 99: Linking the structure of the PLC task with the PLC variables

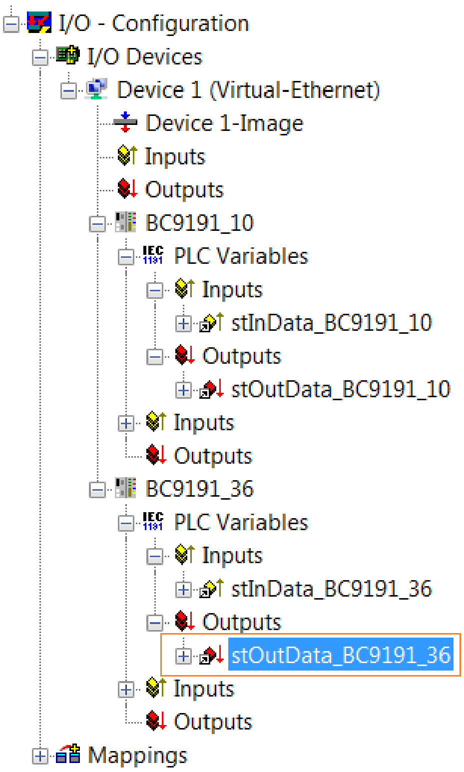


Fig. 100: Linking the structure of the PLC task with the PLC variables

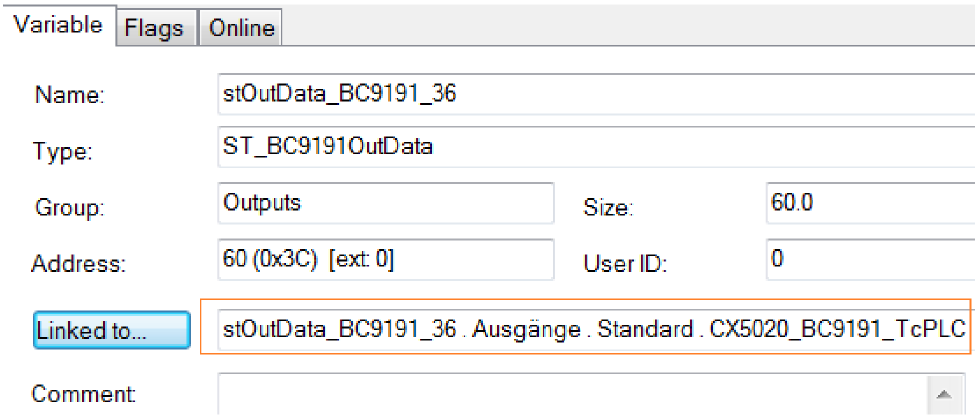


Fig. 101: Display of the successful variable mapping

Detailed description of the TwinCAT PLC program of the master controller

(CX5020_BC9191_TcPLC.pro as ZIP file) (https://infosys.beckhoff.com/content/1033/bc9191/Resources/3440596747/.zip)

(MasterSlave_TcSM.tsm as ZIP file) (https://infosys.beckhoff.com/content/1033/bc9191/Resources/3440598923/.zip)

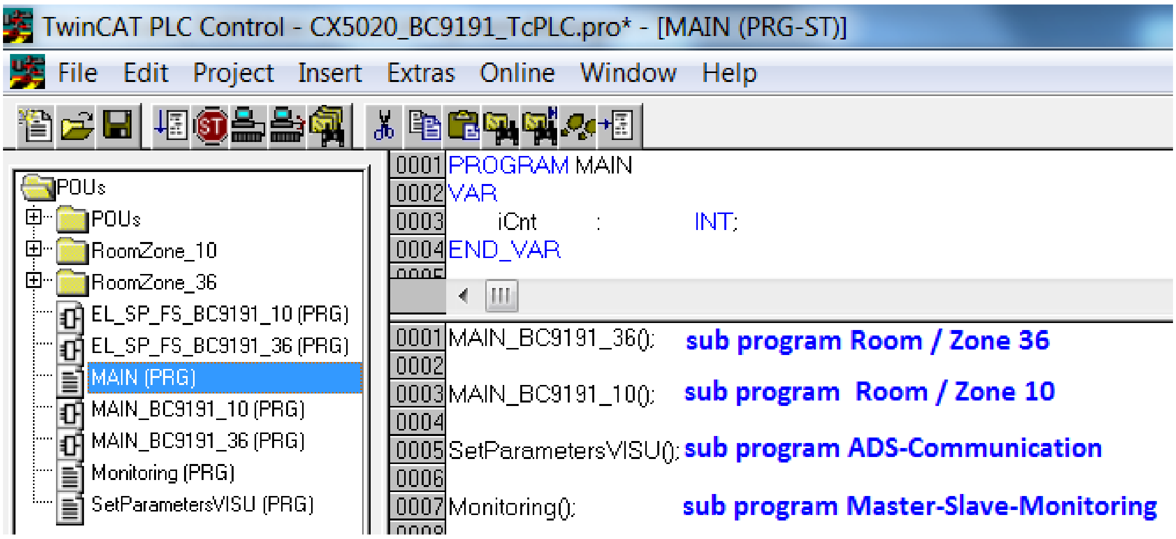


Fig. 102: Sample program with CX5020

- Depending on how many BC9191s are integrated in the master/slave application, it is necessary to declare stInData_BC9191 and stOutData_BC9191 several times globally as allocated variables

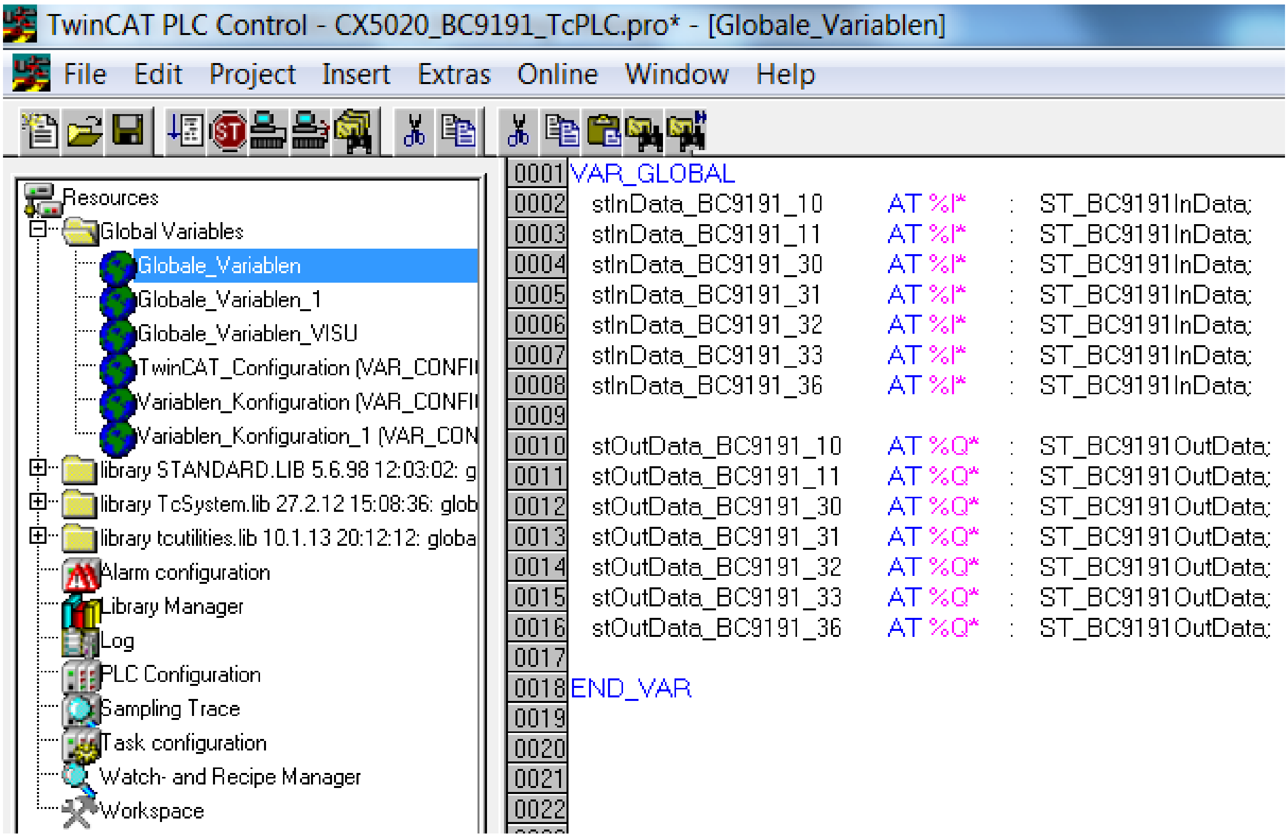


Fig. 103: Definition of BC9191 structures as global variables (CX5020)

- For the Ethernet communication used in the master/slave communication it is necessary to declare the PLC variables on the BC9191 side from the address offset 1000_{dec}

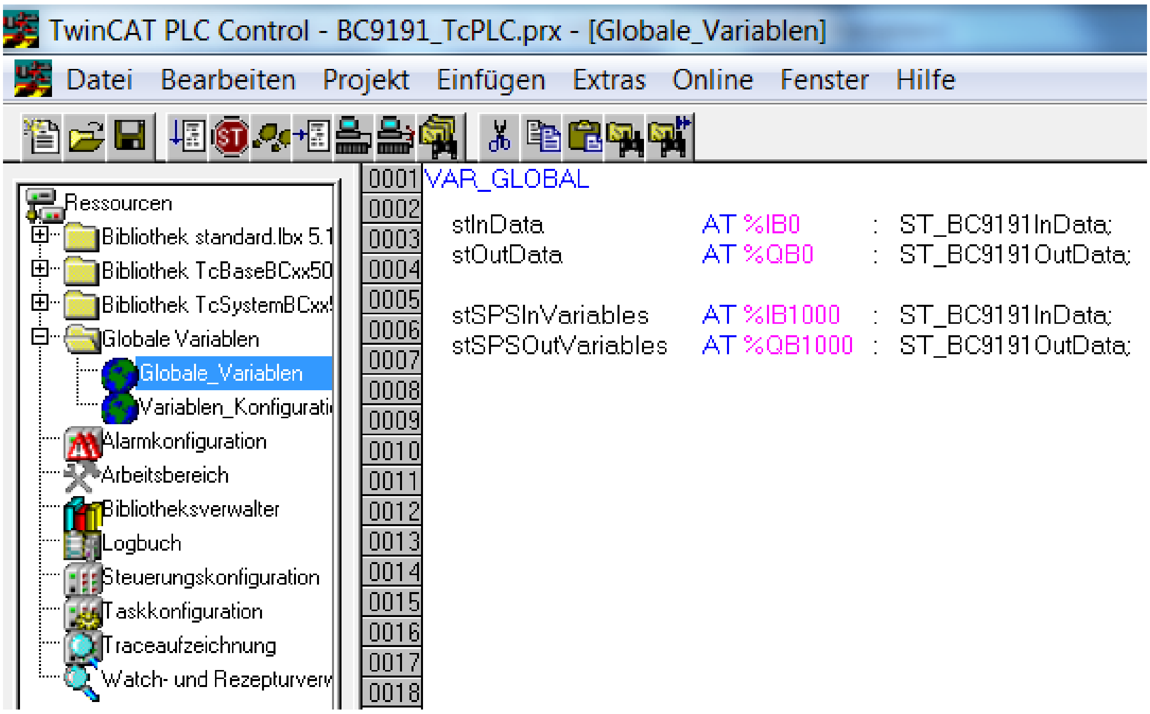


Fig. 104: BC9191 structures are placed on IB1000 and QB1000 for master/slave communication

- The master/slave communication is monitored in the Monitoring () subprogram.
- If the connection between master and slave is interrupted, the value of the structure variable .BC9191_uiMonitoring falls to 0. In this case, the room/zone is no longer controlled centrally, but locally by the BC9191. The control functions are then taken over by the local PLC of the BC9191.

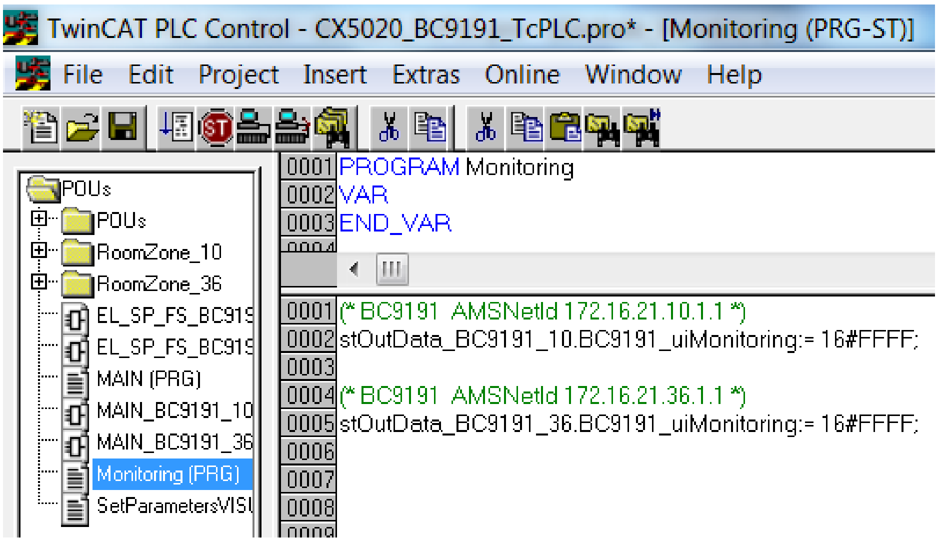


Fig. 105: Diagnosis of the connection via the variable BC9119_uiMonitoring

The master application contains a visualization. On the basis of the VISU and communication via Beckhoff ADS it is possible to execute read and write commands with the connected BC9191s
IndexGroup 16#4020 and 16#4021 ==> IndexOffset, see figure

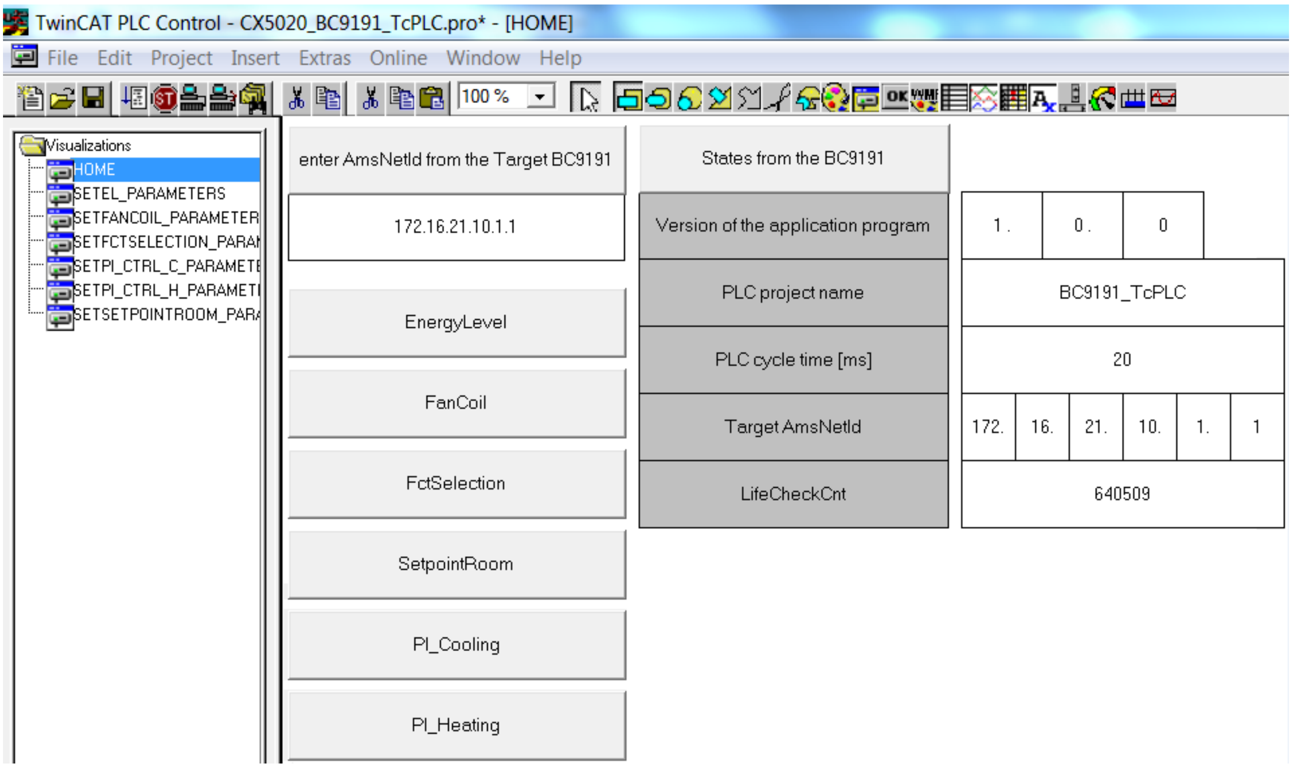


Fig. 106: Visualization of the master application (CX5020)

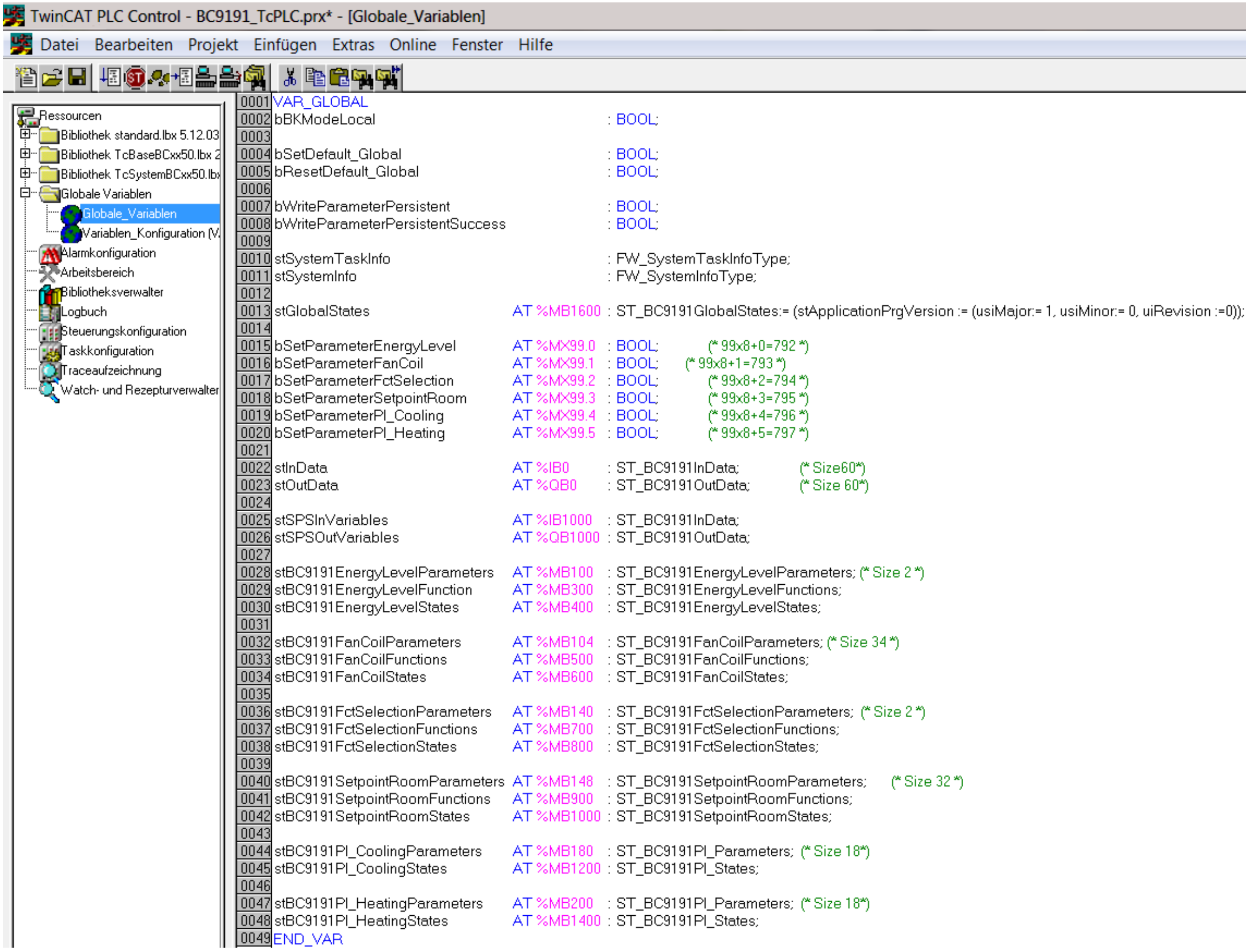


Fig. 107: Display of Global Variables (BC9191)

NOTES

- The BC9191 (C165) and CX/PC (x86) have different memory alignments; this must be taken into account for data structures that are exchanged between the two control platforms. ==> See also the way the **ST_BC9191InData** and **ST_BC9191OutData** are structured.
  BC (165) ==> Data structures have a WORD (2 bytes) memory alignment
  PC / CX (x86) ==> Data structures have a BYTE (1 byte) memory alignment
  CX (ARM) ==> Data structures have a DWORD (4 bytes) memory alignment
- So that variables of the data type REAL are represented correctly by a PC on a BC9191 via the network by Beckhoff ADS, the variables of the data type REAL must be converted to the correct format. That also has to be done if the PC is accessed from the BC9191. The conversion, i.e. the swapping of the HI and LOW word of a REAL variable can be done on the BC9191 or on the PC side with the aid of the function F_SwapRealEx.

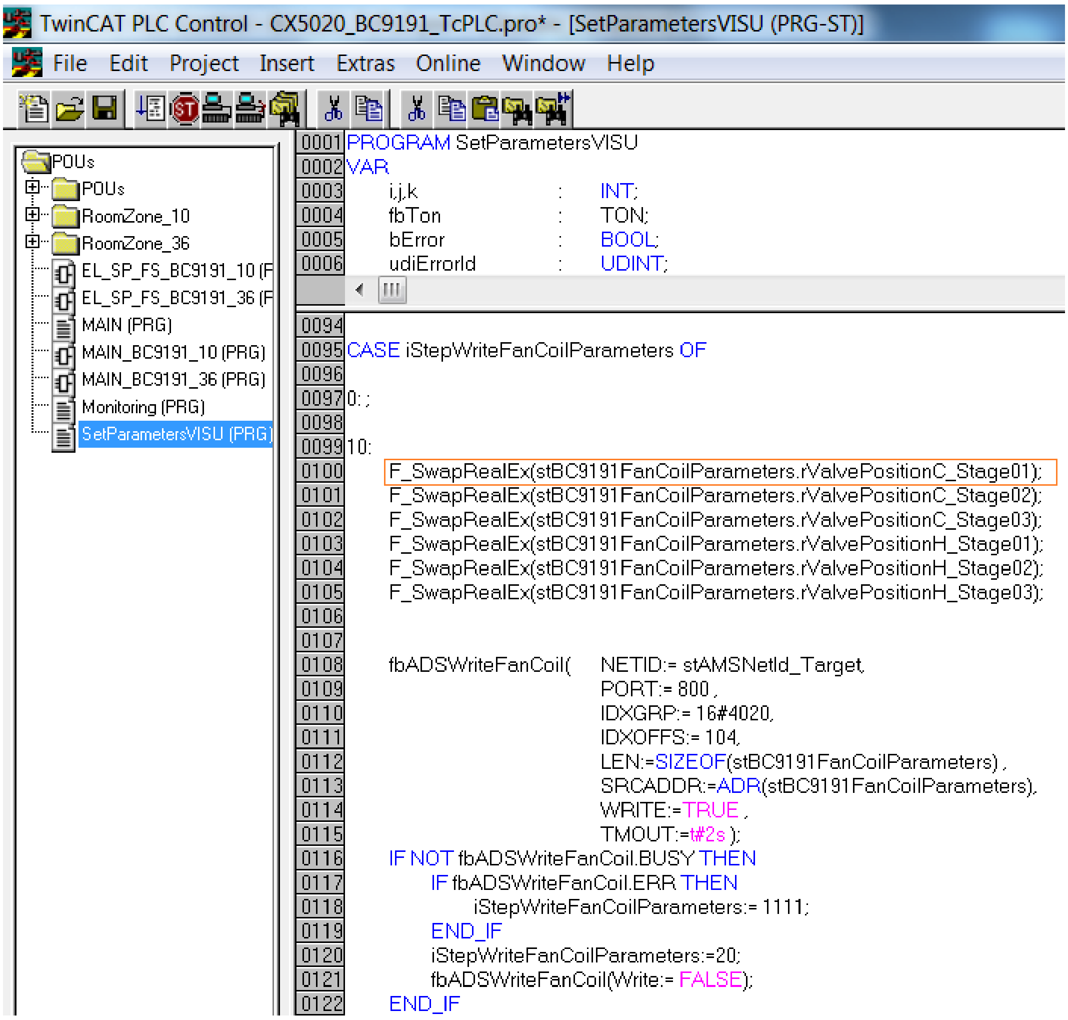


Fig. 108: Swap HI and LOW word

The PLC code, which runs locally on a BC9191 in the case of a communication interruption, is mapped once more on the master controller for each BC9191 and normally actively controls and regulates.

Blue and orange marked program sections belong functionally together.

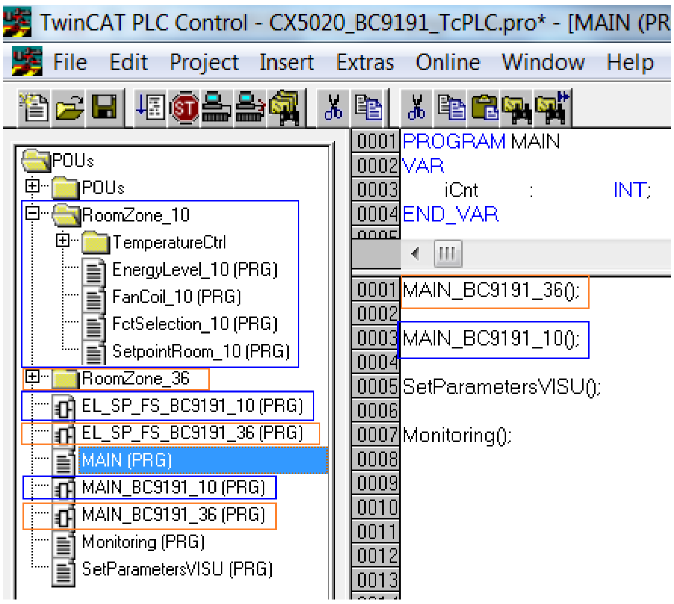


Fig. 109: Assignment of two BC9191 in the master configuration

In the individual programs that refer to an explicit BC9191, the assignment of the physical inputs and outputs must be made in the corresponding structures *and* **stInData_BC9191_xx stOutData_BC9191_xx** .

Example for the BC9191_10

The assignment to the physical inputs takes place in the EL_SP_FS_BC9191_10(PRG) program section. (stInData_BC9191_10 : ST_BC9191InData;)
The assignment to the physical outputs takes place in the program part MAIN_BC9191_10(PRG). (stOutData_BC9191_10 : ST_BC9191OutData;)

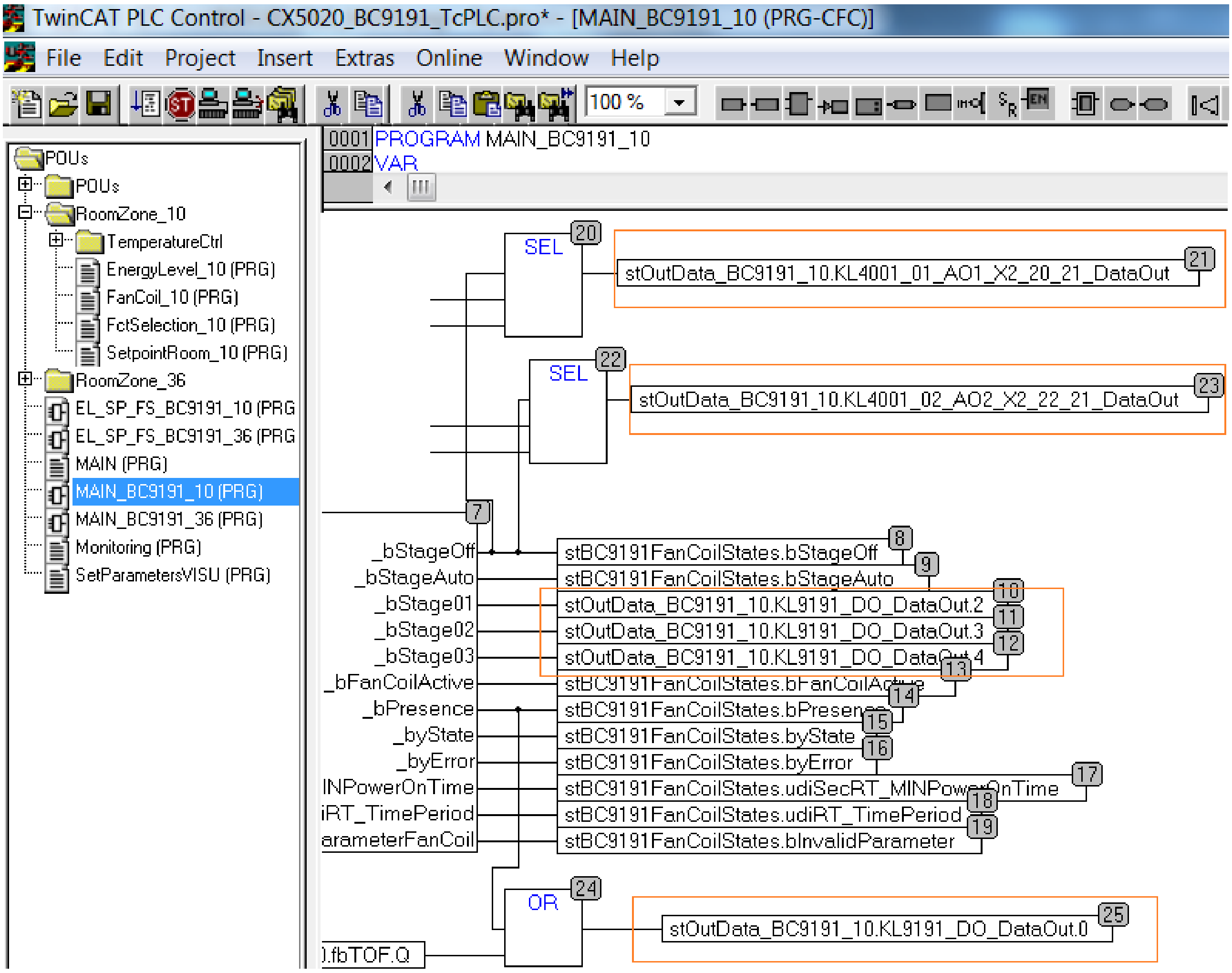


Fig. 110: Assignment of the physical inputs

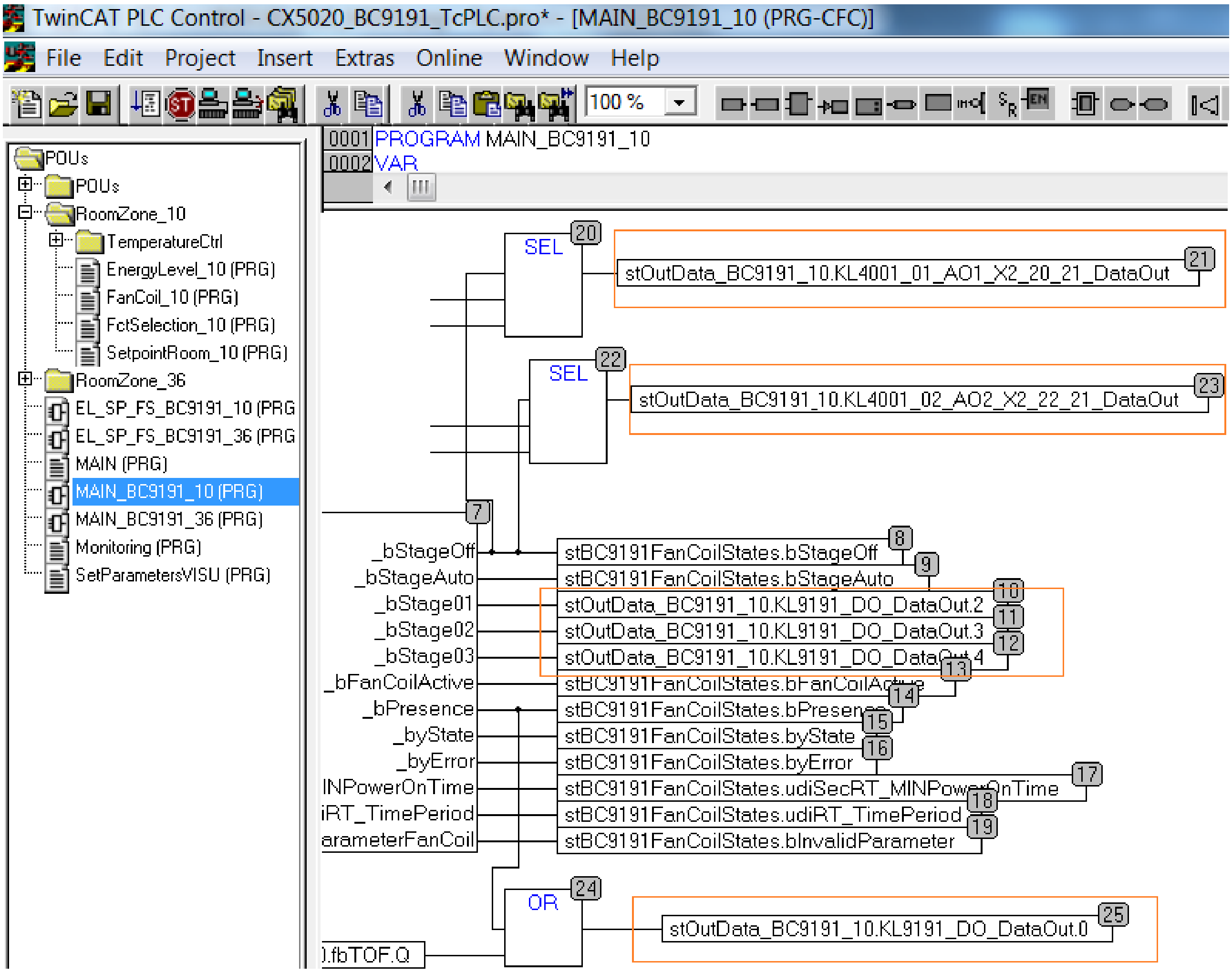


Fig. 111: Assignment of the physical outputs

7 Ethernet

7.1 Introduction to the system

7.1.1 Ethernet

Ethernet was originally developed by DEC, Intel and XEROX (as the "DIX" standard) for passing data between office devices. The term nowadays generally refers to the *IEEE 802.3 CSMA/CD* specification, published in 1985. Because of the high acceptance around the world this technology is available everywhere and is very economical. This means that it is easy to make connections to existing networks.

There are now a number of quite different transmission media: coaxial cable (10Base5), optical fiber (10BaseF) or twisted pairs (10BaseT) with screen (STP) or without screen (UTP). Coaxial cable (10Base5), optical fiber (10BaseF) or twisted pairs (10BaseT) with screen (STP) or without screen (UTP).

Ethernet transmits Ethernet packets from a sender to one or more receivers. This transmission takes place without acknowledgement, and without the repetition of lost packets. To achieve reliable data communication, there are protocols, such as TCP/IP, that can run on top of Ethernet.

MAC-ID

The sender and receiver of Ethernet packets are addressed by means of the MAC-ID. The MAC-ID is a 6 byte identification code unique to every Ethernet device in the world. The MAC-ID consists of two parts. The first part (i.e. the first 3 bytes) is a manufacturer identifier. The identifier for Beckhoff is 00 01 05. The next 3 bytes are assigned by the manufacturer and implement a unique serial number. The MAC-ID can, for example, be used for the BootP protocol in order to set the TCP/IP number. This involves sending a telegram containing the information such as the name or the TCP/IP number to the corresponding node. You can read the MAC-ID with the KS2000 configuration software.

The Internet Protocol (IP)

The internet protocol (IP) forms the basis of this data communication. IP transports data packets from one device to another; the devices can be in the same network, or in different networks. IP here looks after the address management (finding and assigning MAC-IDs), segmentation and routing. Like the Ethernet protocol, IP does not guarantee that the data is transported - data packets can be lost, or their sequence can be changed.

TCP/IP was developed to provide standardized, reliable data exchange between any numbers of different networks. TCP/IP was developed to provide standardized, reliable data exchange between any numbers of different networks. Although the term is often used as if it were a single concept, a number of protocols are layered together: z. B. IP, TCP, UDP, ARP and ICMP.

Transmission Control Protocol (TCP)

The Transmission Control Protocol (TCP) which runs on top of IP is a connection-oriented transport protocol. It includes error detection and handling mechanisms. Lost telegrams are repeated.

User Datagram Protocol (UDP)

UDP is connectionless transport protocol. It provides no control mechanism when exchanging data between sender and receiver. This results in a higher processing speed than, for example, TCP. Checking whether or not the telegram has arrived must be carried out by the higher-level protocol.

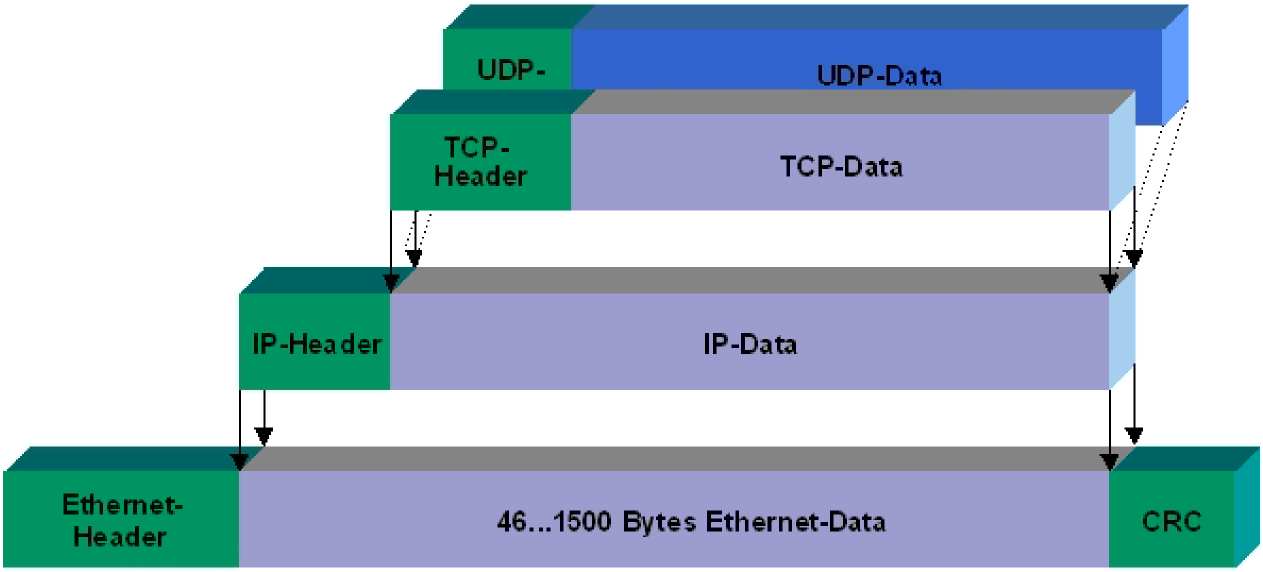


Fig. 112: User Datagram Protocol (UDP)

Protocols running on top of TCP/IP and UDP/IP

The following protocols can run on top of TCP/IP or UDP:

- ADS
- ModbusTCP

Both of these protocols are implemented in parallel on the Bus Coupler, so that no configuration is needed to activate the protocols.

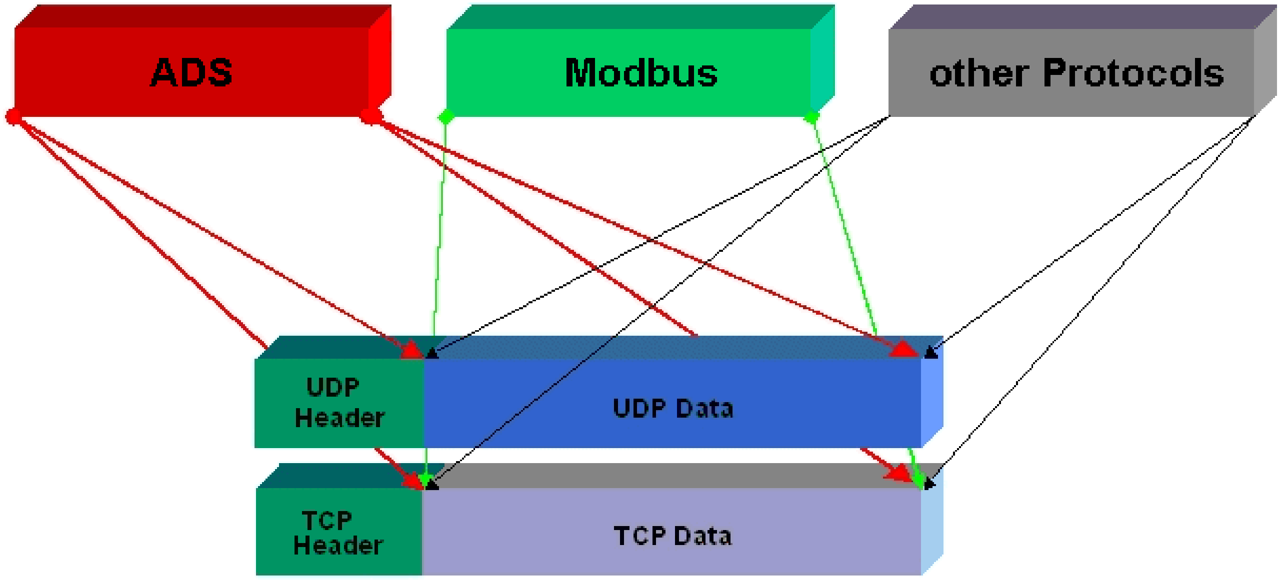


Fig. 113: Protocols running on top of TCP/IP and UDP/IP

ADS can be used on top of either TCP or UDP, but ModbusTCP is always based on TCP/IP.

7.2 Modbus TCP

7.2.1 ModbusTCP Protocol

The Ethernet protocol is addressed by means of the MAC-ID. The user does not normally need to be concerned about this address. The IP number has a length of 4 bytes, and must be parameterized by the user on the Bus Coupler and in the application. In ModbusTCP, the TCP port is set to 502. The UNIT can be freely selected under ModbusTCP, and does not have to be configured by the user.

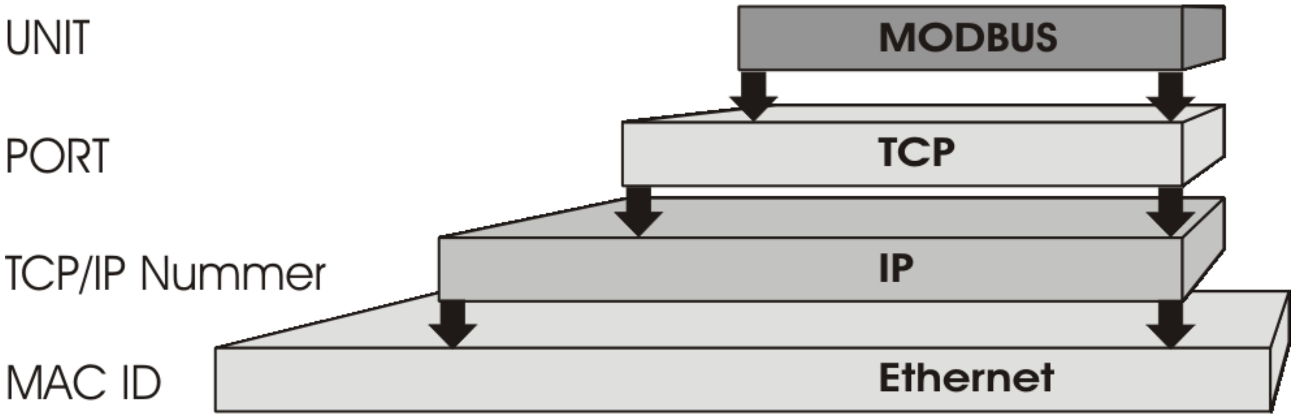


Fig. 114: ModbusTCP Protocol

TCP port number

The TCP port number for ModbusTCP has been standardised to 502.

Modbus-Unit

The unit is returned by the slave.

ModbusTCP Protocol

Byte	Name	Description
0	Transaction identifier	is returned by the slave
1	Transaction identifier	is returned by the slave
2	Protocol identifier	always 0
3	Protocol identifier	always 0
4	Length field	0 (if the message is less than 256 bytes in length)
5	Length field	Number of following bytes
6	UNIT identifier	returned by the slave
7	Modbus	Modbus protocol beginning with the function follows

7.2.2 ModbusTCP diagnostic

7.2.2.1 ModbusTCP diagnostic

See Modbus diagnostic function [▶ 122]

See ModbusTCP error answers [▶ 122]

7.2.2.2 ModbusTCP slave error answer (BK9000, BX/BC9xx0, BC9191, IP/ ILxxxx-B/C900, EK9000)

When the user sends the slave either a request or information that the coupler does not understand, the slave responds with an error report. This response contains the function and the error code. 0x80 is added to the value returned by the function.

Code	Name	Meaning
1	ILLEGAL FUNCTION	Modbus function not implemented
2	ILLEGAL DATA ADDRESS	Invalid address or length
3	ILLEGAL DATA VALUE	Invalid parameters - Diagnostic functions - Wrong register
4	SLAVE DEVICE ERROR	Watchdog or K-bus error EK9000: E-bus error
6	SLAVE DEVICE BUSY	Output data is already been received from another IP device

7.3 Modbus TCP Functions

7.3.1 Read holding register (Function 3)

The *Read holding register* function can be used to read the input and output words and the registers. Inputs from offset 0 - 0xFF and outputs from offset 0x800 - 0x8FF, and for controllers (BC, BX) the flag area from offset 0x4000.

In this example the first two analog outputs (or two output words) are read. The analog outputs (or output words) start at offset 0x800. The length indicates the number of channels (or words) to be read.

Query

Byte name	Sample
Function code	3
Start address high	8
Start address low	0
Count high	0
Count low	2

The fieldbus coupler answers with byte count 4, i.e. 4 bytes of data are returned. The query was for two analog channels, and these are distributed over two words. In the analog output process image, the first channel has the value 0x3FFF, while the second channel has the value 0x0.

Response

Byte name	Sample
Function code	3
Byte count	4
Data 1 high byte	63
Data 1 low byte	255
Data 2 high byte	0
Data 2 low byte	0

7.3.2 Read input register (Function 4)

The function *Read input register* reads the inputs on a word basis.

In this example the first two analog inputs (or the first two input words) are read. The analog inputs (or input words) start at an offset of 0x0000. The length indicates the number of words to be read. A KL3002, for example, has two words of input data. Therefore, the length to be entered at *Number low* is two.

Query

Byte name	Sample
Function code	4
Start address high	0
Start address low	0
Count high	0
Count low	2

The fieldbus coupler answers with byte count 4, i.e. four bytes of data are returned. The query was for two analog channels, and these are now distributed over 2 words. In the analog input process image, the first channel has the value 0x0038, while the second channel has the value 0x3F1B.

Response

Byte name	Sample
Function code	4
Byte count	4
Data 1 high byte	0
Data 1 low byte	56
Data 2 high byte	63
Data 2 low byte	11

7.3.3 Preset single register (Function 6)

The function *Preset singles register* can be used to access the output or flag process image (only for controllers) and the Modbus TCP interface.

Function 6 writes the first output word. The outputs start at an offset of 0x0800. Here again the offset always describes a word. This means offset 0x0803 refers to the fourth word in the output process image.

Query

Byte name	Sample
Function code	6
Start address high	8
Start address low	0
Data high	63
Data low	255

The Fieldbus Coupler replies with the same telegram and confirmation of the received value.

Response

Byte name	Sample
Function code	6
Start address high	8
Start address low	0
Data high	63
Data low	255

7.3.4 Preset single register (Function 16)

The *Preset multiple register* function can be used to write a number of outputs. The first two analog output words are written in this example. The outputs start at an offset of 0x0800. Here the offset always describes a word. Offset 0x0003 writes to the fourth word in the output process image. The length indicates the number of words, and the *Byte count* is formed from the combination of all the bytes that are to be written.

Sample: 4 words - correspond to a byte count of 8

The data bytes contain the values for the analog outputs. In this example, two words are to be written. The first word is to receive the value 0x7FFF, and the second word is to receive the value 0x3FFF.

Query

Byte name	Sample
Function code	16
Start address high	8
Start address low	0
Length high	0
Length low	2
Byte count	4
Data 1 byte 1	127
Date 1 byte 2	255
Date 2 byte 1	63
Data 2 byte 2	255

Response

The coupler replies with the start address and the length of the transmitted words.

Byte name	Sample
Function code	16
Start address high	8
Start address low	0
Length high	0
Length low	2

7.3.5 Read / write registers (Function 23)

A number of analog outputs can be written and a number of analog inputs read with one telegram using the *Read / write registers* function. In this example the first two analog output words are written, and the first two analog inputs are read. The analog outputs start at offset 0x0800, while the inputs start at offset 0x0000. Here the offset always describes a word. Offset 0x0003 writes to the fourth word in the output process image. The length indicates the number of words, and the *Byte count* is formed from the combination of all the bytes that are to be written. Sample: 4 words - correspond to a byte count of 8

The data bytes contain the values for the analog outputs. In this example, two words are to be written. The first word is to receive the value 0x3FFF, and the second word is to receive the value 0x7FFF.

Query

Byte name	Sample
Function code	23
Read start address high	0
Read start address low	0
Read length high	0
Read length low	2
Write start address high	8
Write start address low	0
Write length high	0
Write length low	2
Byte count	4
Data 1 high	63
Data 1 low	255
Data 2 high	127
Data 2 low	255

Response

The coupler replies with the start address and the length of the bytes to be transferred in *Byte count*. The data information follows. In this example the first word contains 0x0038 while the second word contains 0x3F0B.

Byte name	Sample
Function code	23
Byte count	4
Data 1 high	0
Data 1 low	56
Data 2 high	63
Data 2 low	11

7.4 ADS-Communication

7.4.1 ADS-Communication

The ADS protocol (ADS: Automation Device Specification) is a transport layer within the TwinCAT system. It was developed for data exchange between the different software modules, for instance the communication between the NC and the PLC. This protocol enables communication with other tools from any point within the TwinCAT. If communication with other PCs or devices is required, the ADS protocol can use TCP/IP as a basis. Within a networked system it is thus possible to reach all data from any point.

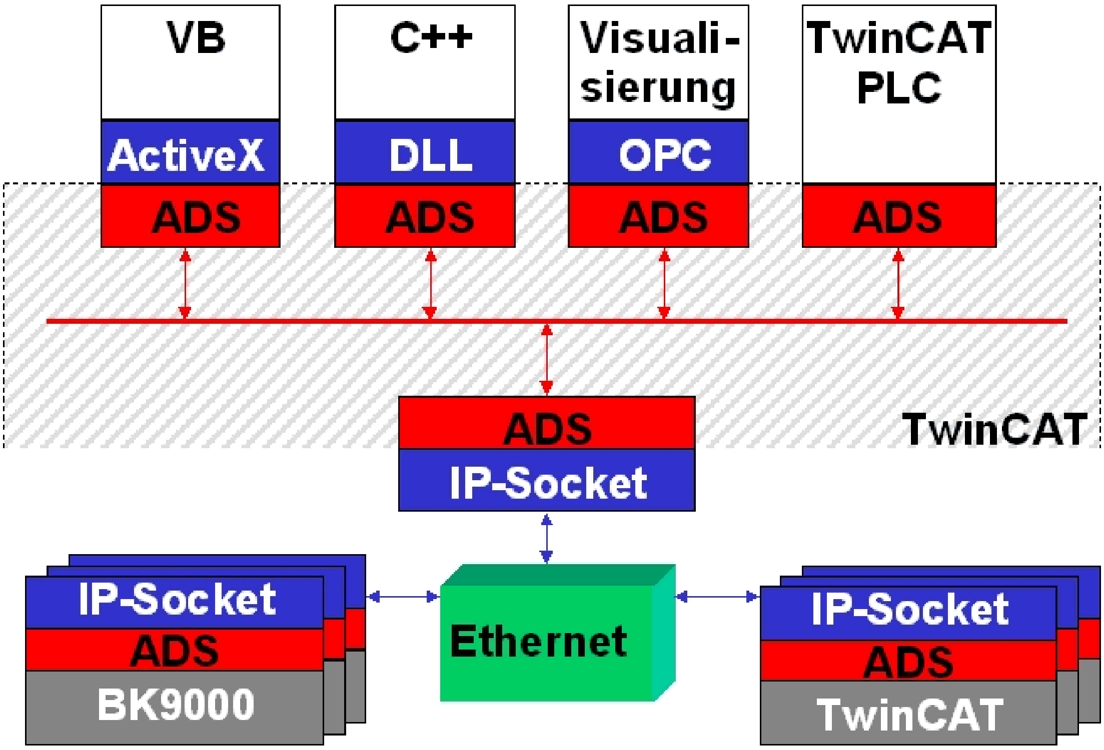


Fig. 115: The ADS protocol as a transport layer within the TwinCAT system

The ADS protocol runs on top of the TCP/IP or UDP/IP protocols. It allows the user within the Beckhoff system to use almost any connecting route to communicate with all the connected devices and to parameterize them. Outside the Beckhoff system a variety of methods are available to exchange data with other software tools.

Software interfaces

ADS-OCX

The ADS-OCX is an Active-X component. It offers a standard interface to, for instance, Visual Basic, Delphi, etc.

ADS-DLL

You can link the ADS-DLL (DLL: Dynamic Link Library) into your C program.

OPC

The OPC interface is a standardised interface for communication used in automation engineering. Beckhoff offer an OPC server for this purpose.

7.4.2 ADS protocol

The ADS functions provide a method for accessing the Bus Coupler information directly from the PC. ADS function blocks can be used in TwinCAT PLC Control for this. The function blocks are contained in the *TcSystem.lib* library. It is also equally possible to call the ADS functions from AdsOCX, ADSDLL or OPC. It is possible to access all the data through ADS port number 300, and to access the registers of the Bus Coupler and Bus Terminals through ADS port number 100.

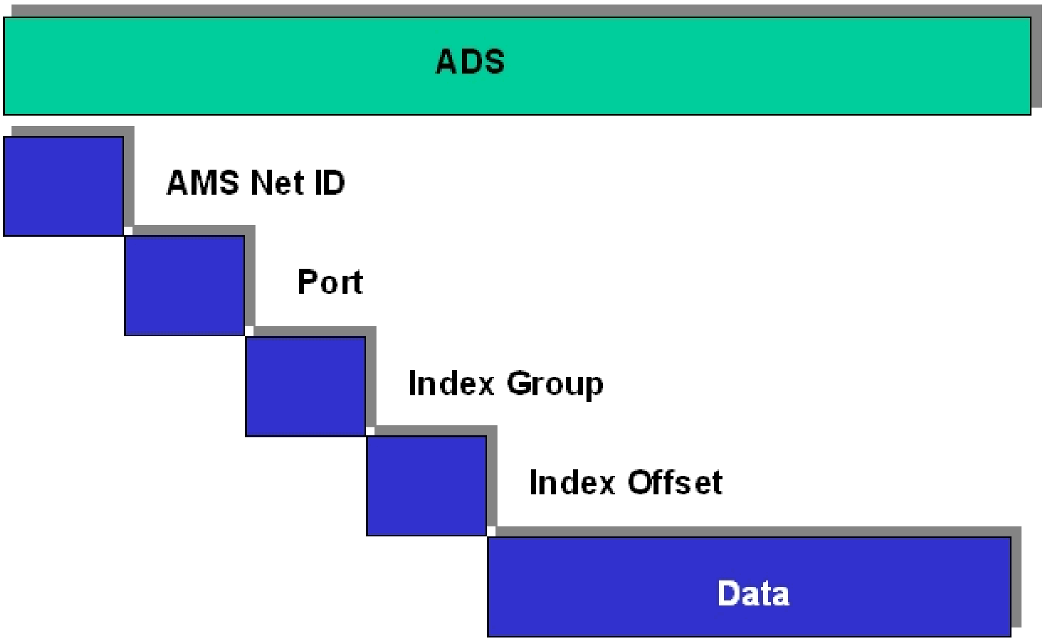


Fig. 116: Structure of the ADS protocol

AMSNetID

The AMSNetID provides a reference to the device that is to be addressed. This is created from the set TCP/IP address and an additional 2 bytes. These additional 2 bytes consist of "1.1", and cannot be altered.
Sample:
IP address 172.16.17.128
AMSNetID 172.16.17.128.1.1

Port number

The port number distinguishes sub-elements in the connected device.
Port 100: register access
Port 300: fieldbus process data
Port 800: local process data (BC90x0, C900 only)

Index group

The index group distinguishes different data within a port.

Index offset

Indicates the offset, from which reading or writing the byte is to start.

Len

Gives the length of the data, in bytes, that is to be read or written.

TCP port number

The TCP port number for the ADS protocol is 48898 or 0xBF02.

7.4.3 ADS services

Local Process Image PLC Task 1 Port 800/801

Data can be read from and written to the local process image. If it is necessary for outputs to be written, it is important to ensure that they are not used by the local PLC, because the local controller will overwrite these values. The data are not associated with a watchdog, and therefore must not be used for outputs that would have to be switched off in the event of a fault.

Index group	Meaning	Index offset (value range)
0xF020	Inputs	0...2047
0xF021	Bit inputs	0...16376
0xF030	Outputs	0...2047
0xF031	Bit outputs	0...16376
0x4020	Flags	0...4095
0x4021	Flag bit	0...32760

ADS services

AdsServerAdsState

Data type (read only)	Meaning
String	Start - the local PLC is running Stop - the local PLC is in stop mode

AdsServerDeviceState

Data type (read only)	Meaning
INT	0 - Start - the local PLC is running 1 - Stop - the local PLC is in stop mode

AdsServerType

Data type (read only)	Meaning
String	BX PLC Server

ADSWriteControl

Data type (write only)	Meaning
NetID	Net ID of the Ethernet Controller*
Port	800
ADSSTATE	5 - RUN / 6 - STOP
DEVSTATE	0
LEN	0
SRCADDR	0
WRITE	positive edge starts the function block
TMOUT	e.g.: t#1000 ms

* BC9050, BC9020, BC9120, BC9191, BX9000

Register Access Port 100

On the Bus Terminal Controllers of the BX series, and on the BCxx50/xx20 and BC9191 , the ADS port number for register communication is fixed at 100.

Index group	Index offset (value range)		Meaning
	Hi-Word	Low Word	
0 [READ ONLY]	0...127	0...255	Bus Coupler register High-word table number of the Bus Coupler Low-word register number of the table
1...255	0...3	1...255	Bus Terminal registers High-word channel number Low-word register number of the Bus Terminal

Timeout of the ADS function block

When reading the register, the time out for the ADS block has to be set to a time longer than 1 second.

Setting the password

When writing to the registers, the password has to be set (see the documentation for the particular Bus Terminal).

7.4.4 Examples

7.4.4.1 AMS routing table

The AMS routing table can be used for two functions:

1. If the AMS Net ID does not match the TCP/IP address, the link between the two non-matching numbers can be entered here.
   The AMS Net ID matches the TCP/IP address if the first 4 bytes are identical and the AMS Net ID ends with ".1.1".
2. Only AMS Net IDs stored in the table can access the BC9000 via ADS.

Please ensure that all ADS devices with which communication is required are entered in the table. If the table is empty (default), all ADS devices can access the BC9000.

Processing the AMS routing table using the KS2000 configuration software

From KS2000 version 4.3.0.39 entries in the AMS routing table can be made via dialog.

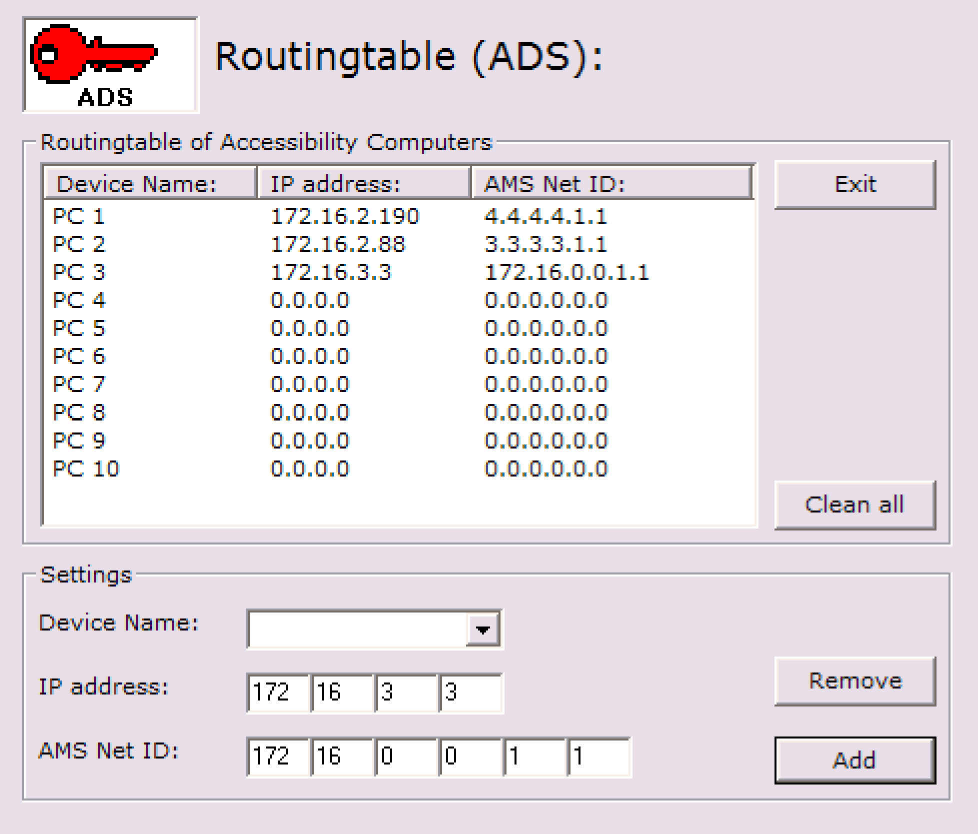


Fig. 117: Displaying the AMS routing table using the KS2000 configuration software

Entering values in the routing table via ADS

The AMS routing table can be loaded into the BC9000 via ADS. Ensure that the first entry is that of the sender, since this setting becomes effective immediately and may otherwise prevent further entries in the AMS routing table.

Download (https://infosys.beckhoff.com/content/1033/bc9191/Resources/6744240267/.zip)

7.4.4.2 Example of communication between several BC9000

ADS-Communication

Data can be sent directly from one BC9000 to another BC9000 using the ADS function blocks from the TcADSBC.lb6 library. ADSREADEX can be used to read data from another BC9000, ADSWRITE can be used for writing. A maximum of 3 BC9000 can be accessed at the same time. In order to communicate with further BC9000s, the connection to at least one BC9000 has to be closed via ADSCLOSE.

Make sure that there are only a maximum of three connections from one BC9000 to other BC9000s. You should reserve the fourth connection for programming.

These connections may also be established by another remote station.

Example for a functioning ADS connection

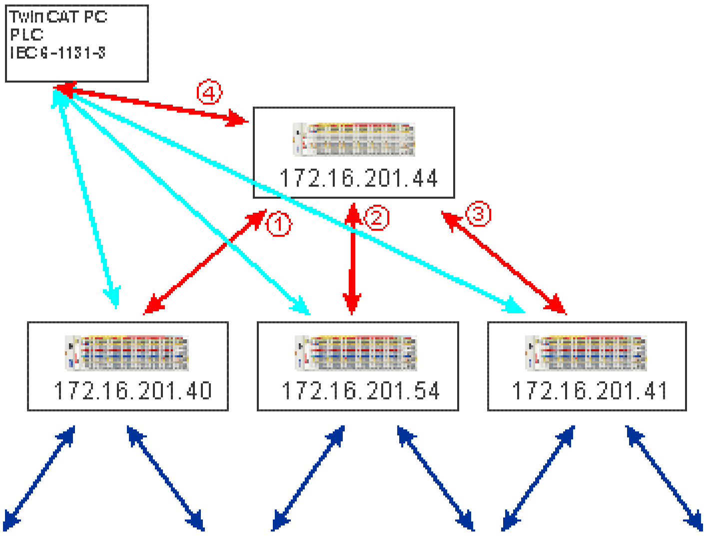


Fig. 118: Functioning ADS connection

Sample program for connecting BC9000 to BC9000

Sample program in ST Master: (https://infosys.beckhoff.com/content/1033/bc9191/Resources/3440601099/.zip)

Sample program in ST Slave: (https://infosys.beckhoff.com/content/1033/bc9191/Resources/3440603275/.zip)

Cross communication only between BC9000 and BC9000

Please note that a BK9000 only allows write access from a master, and that the BK9000 has to be polled regularly at 1 second intervals. It is therefore recommended to use cross communication only between BC9000 and BC9000 and not to BK9000.

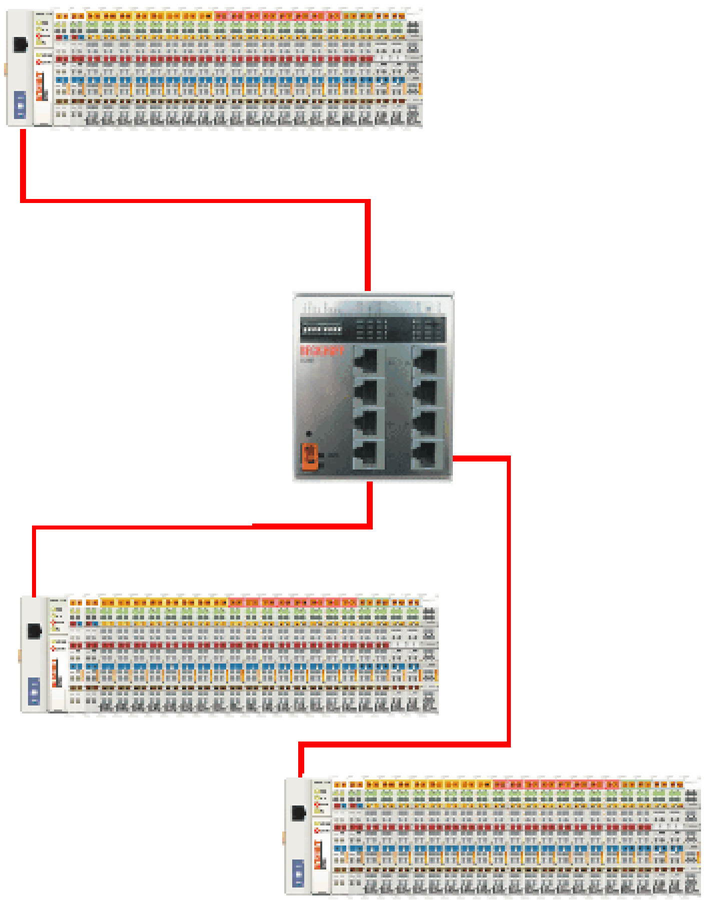


Fig. 119: Communication between several BC9000 - topology

7.4.4.3 Example communication from BC9000 to a TwinCAT PC system

In some applications polling operation (master-slave principle) is not wanted. The ADS function blocks of the BC9000 enable both cross communication (from BC9000 to BC9000) and communication to the higher-level controller (TwinCAT PC system). This means that the BC9000 application can decide when data are sent to the TwinCAT PC system. Any fault information, for example, is transferred immediately to the TwinCAT PC.

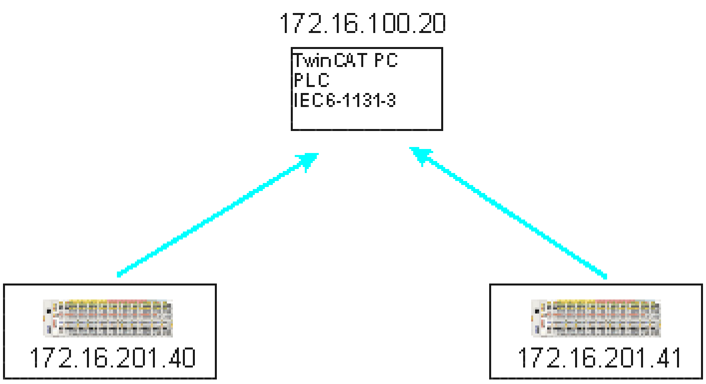


Fig. 120: Communication from the BC9000 to the higher-level controller (TwinCAT PC system)

Requirement:
TwinCAT 2.9 Build 1025
The IP address matches the AMS Net ID plus ".1.1"

An >ipconfig< command issued in a Dos window can be used to find out the IP address of your PC.

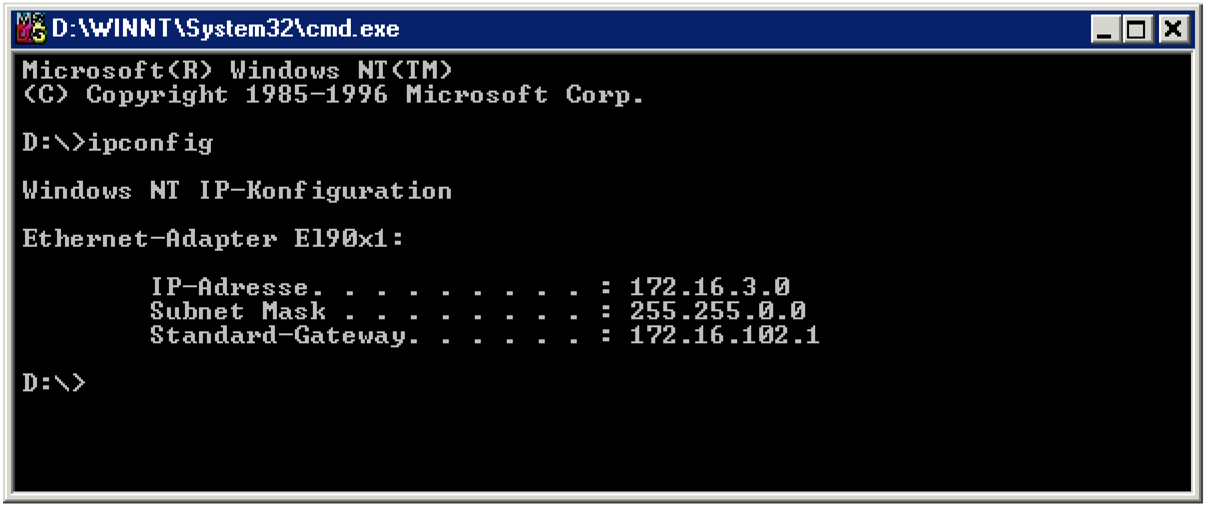


Fig. 121: The command *ipconfig*

The AMS Net ID is displayed under Properties/AMS router.

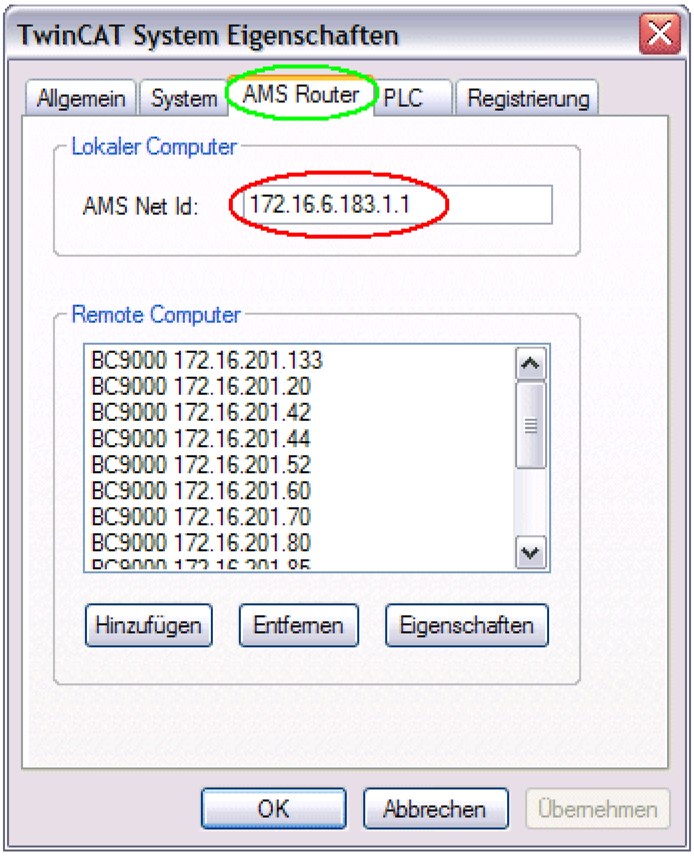


Fig. 122: Display of the AMS Net ID under Properties/AMS Router

If the AMS Net ID does not match the TCP/IP address, the link can be entered in the routing table (table 2).

Example for AMS Net ID OK

AMS Net ID: 172.16.100.10.1.1
TC/IP: 172.16.100.10

Required library: AdsBC.lb6

Example programs for a connection between BC9000 and TwinCAT PC

Sample program in ST Master:
(https://infosys.beckhoff.com/content/1033/bc9191/Resources/3440601099/.zip)

Sample program in ST Slave:
(https://infosys.beckhoff.com/content/1033/bc9191/Resources/3440603275/.zip)

7.4.4.4 Example: Changing the Ethernet Coupler settings via ADS

Requirement:
For this example a BC9000 is required. Please note two local sockets are required for reading or writing!

Required library:
AdsBC.lb6
TcPlcUlititiesBC.lb6
PlcSystemBC.lb6
PlcHelperBC.lb6
TcUtilitiesBC9000.lb6

Description:
The function block enables setting of various parameters (see sample). A more detailed description of the function block can be found in the chapter *Library*.

Sample:

Sample program in ST:
(https://infosys.beckhoff.com/content/1033/bc9191/Resources/4351425291/.zip)

8 Error handling and diagnostics

8.1 Diagnostics

Ethernet State

In many cases it is important to know whether the communication with the higher-level master is still OK. To this end, link the *FieldbusState* variable with your PLC program.

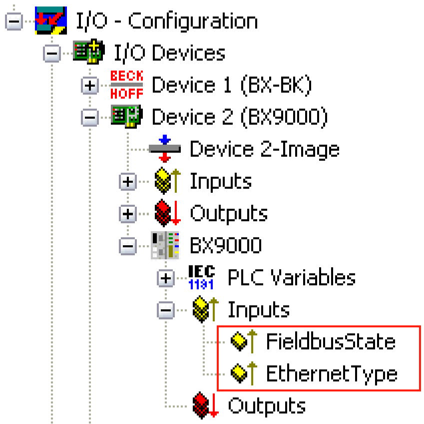


Fig. 123: Ethernet fieldbus state

Fieldbus State

Error number	Description	Remedy
0	No error	-
1	Watchdog error	Communication interrupted

Ethernet Type

Here it is possible to determine which Ethernet protocol is accessing the PLC variables, and thereby triggering the watchdog (for example, the data in the Default Config area starting from addresses %IB1000 and %QB1000).

Diagnosis number	Description	Remedy
0x0000	No protocol is accessing the PLC variables	-
0x0001	ADS TCP	Communication via ADS TCP/IP
0x0002	ADS UDP	Communication via ADS UDP/IP
0x0010	ModbusTCP	Communication via Modbus TCP/IP

Reading fieldbus state by ADS

The fieldbus status can be read through ADSREAD in the default configuration or in the TwinCAT configuration (under development).

Parameter ADSREAD function block	Description
NetID	local – empty string
Port	
IndexGroup	
IndexOffset	
LEN	

State of the K-bus

An internal bus or Bus Terminal error is indicated in the K-bus state. A more precise fault description can be obtained via a function block (in preparation). To this end, link the *K-bus-State* variable with your PLC program.

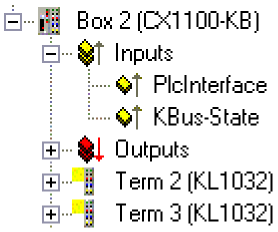


Fig. 124: State of the K-bus

Error bit	Description	Error type
0	No error	No ERROR
Bit 0	K-bus error	ERROR
Bit 2	K-bus is re-triggered	NOTE

Reading K-bus state by ADS

In default or TwinCAT configuration the fieldbus state can be read via ADSREAD.

Parameter ADSREAD function block	Description
NetID	local – empty string
Port	1
IndexGroup	16#0006
IndexOffset	16#000C_9000
LEN	1

8.2 Diagnostic LEDs

The BC9191 and BC9191-0100 have LEDs for status display.

Diagnostic LEDs BC9191 and BC9191-0100 up to HW 4

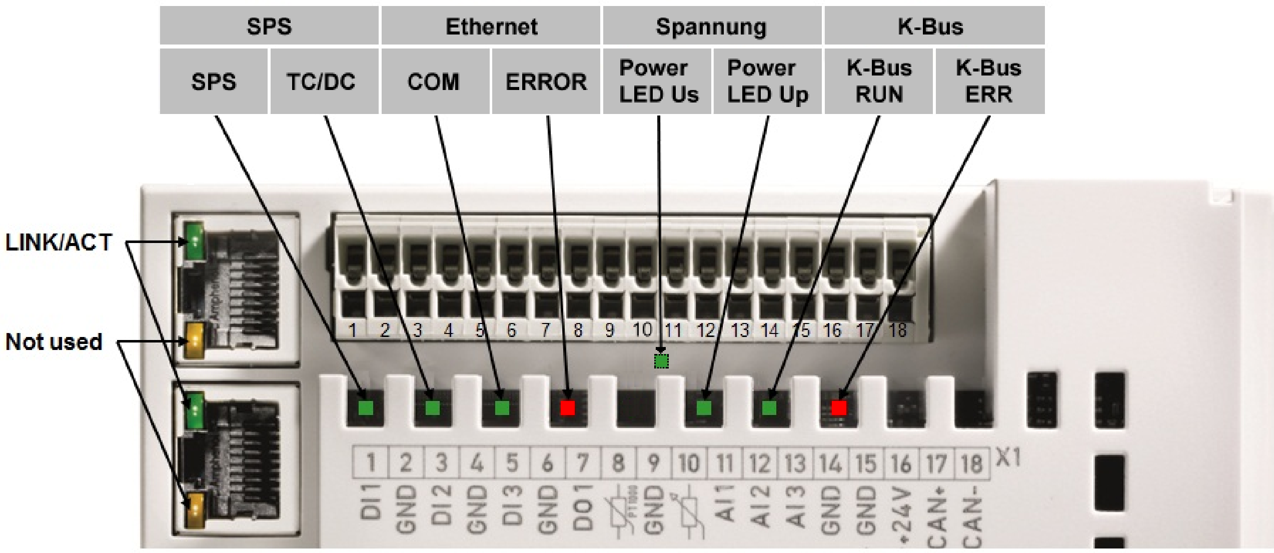


Fig. 125: LEDs BC9191 to HW 4

Diagnostic LEDs BC9191 and BC9191-0100 from HW 5

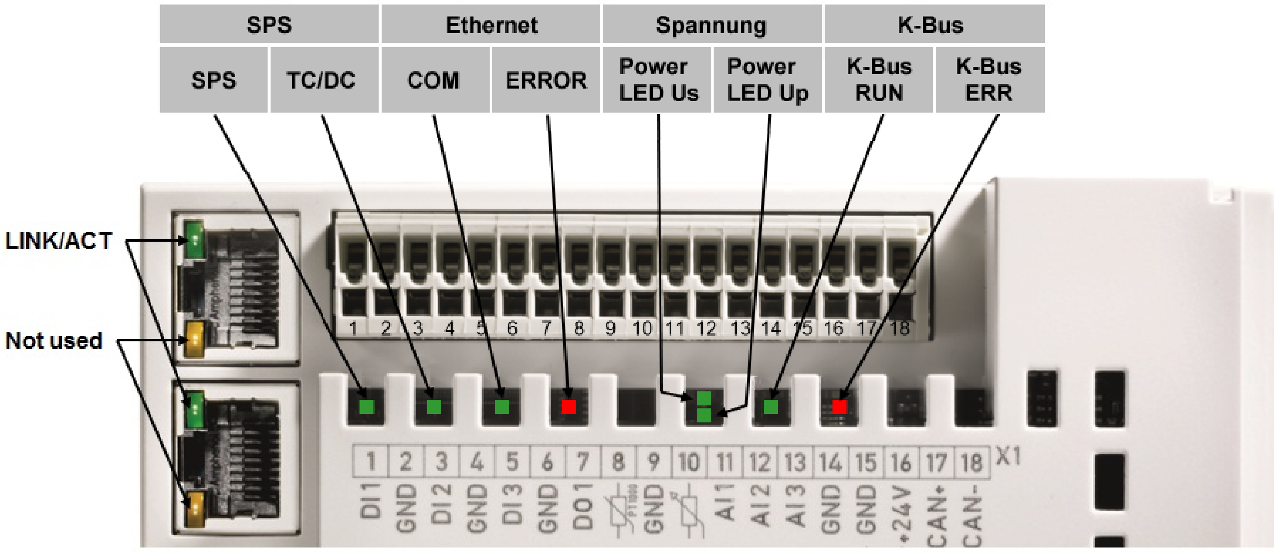


Fig. 126: LEDs BC9191 from HW 5

LEDs for PLC diagnostics

LED	Meaning
PLC	**on** - PLC running, **flashing** - cycle time exceeded, **off** - cycle time permanently exceeded or PLC stopped
TC/DC	**on** - TwinCAT configuration active, **off** - Default configuration active, **flashing** TwinCAT configuration faulty

LEDs for Ethernet diagnostics

LED	Meaning
LINK/ACT (RJ45)	**on** - LINK available, **flashing** - LINK available and communication
COM	Communication with controller available
ERROR	**flashing** - DHCP or BootP active. Waiting for an IP address. **short flashing** - the source code of the PLC program is loaded and is being checked. No boot project can be created during this check time.

LEDs for power supply diagnostics

LED	Meaning
Power LED Vs	**on** - internal power supply 5 V_{DC} okay (for hardware version 3 and 4 only visible with unplugged connector X1)
Power LED Vp	**on** - 24 V_{DC} power supply for external devices okay

LEDs for K-bus diagnostics

LED	Meaning
K-bus RUN	**on** or **flashing** - K-bus running
K-bus ERR	**flashing** (see error code) [▶ 138]

Error code for K-bus diagnostics

Error code	Error argument	Description	Remedy
-	flashing continuously	EMC problems	• Check power supply for undervoltage or overvoltage peaks • Implement EMC measures • If a K-bus error is present, it can be localized by a restart (by switching the BC9191 off and then on again)
1	0	EEPROM checksum error	Enter factory settings with the KS2000 configuration software
	1	Code buffer overflow	Insert fewer Bus Terminals. Too many entries in the table for the programmed configuration
	2	Unknown data type	Software update required for the BC9191
2	-	reserve	-
3	0	K-bus command error	• No Bus Terminal inserted • One of the Bus Terminals is defective; halve the number of Bus Terminals attached and check whether the error is still present with the remaining Bus Terminals. Repeat until the defective Bus Terminal is located.
4	0	K-bus data error, break behind the Bus Coupler	Check whether the n+1 Bus Terminal is correctly connected; replace if necessary
	n	Break behind Bus Terminal n	Check whether the KL9010 bus end terminal is connected
5	n	K-bus error in register communication with Bus Terminal n	Exchange the nth Bus Terminal
6	0	Error at initialization	Exchange Bus Coupler
	1	Internal data error	Perform a hardware reset on the BC9191 (switch off and on again)
	2	DIP switch changed after a software reset	Perform a hardware reset on the BC9191 (switch off and on again)
	3	IP address already assigned	Check whether the IP address already exists in the network.
7	0	Note: cycle time was exceeded	Warning: the set cycle time was exceeded. This note (flashing of the LED) can only be reset by rebooting the BC9191. Short-term remedy: increase cycle time
9	0	Checksum error in Flash program	Transmit program to the BC again
	1	Incorrect or faulty library implemented	Remove the faulty library
10	n	Bus Terminal n is not consistent with the configuration that existed when the boot project was created	Check the nth Bus Terminal. The boot project must be deleted if the insertion of an nth Bus Terminal is intentional.
14	n	nth Bus Terminal has the wrong format	Start the BC9191 again, and if the error occurs again then exchange the Bus Terminal.
15	n	Number of Bus Terminals is no longer correct	Start the BC9191 again. If the error occurs again, restore the manufacturers setting using the KS2000 configuration software
16	n	Length of the K-bus data is no longer correct	Start the Bus Coupler again. If the error occurs again, restore the manufacturers setting using the KS2000 configuration software

8.3 General errors

No data exchange after replacing a bus coupler

You have exchanged the Ethernet Bus Coupler and set the same TCP/IP number, but data is not exchanged.

Each Ethernet device has its own and unique MAC ID. This number is saved when connecting to the Ethernet node and stored in a table. This table contains the correspondences between the MAC-ID and the TCP/IP number. You must clear this memory and you can do this by entering the command >arp -d< and the TCP/IP number of the Bus Coupler in the DOS window.
Example: >arp -d 172.16.17.203<
If the DHCP or BootP protocol is activated, the MAC ID of the new Bus Coupler must be set on the DHCP or BootP server after the coupler change.

Communication errors when online *

After a period in the online state (logged in via Ethernet/AMS) the message *Communication error - logging out always occurs*.

The data traffic through the Ethernet interface is jamming.

Remedy:

- reduce the level of data communication.
- Stop the cyclical data traffic, or lengthen the task time.
- Reduce the number of windows open in the online representation.
- Log in via the serial interface.

* BC9000, C900 only

8.4 ADS diagnostics

Status inputs

It is possible to monitor the BK/BC9000, B/C900 Bus Coupler's communication in the System Manager. Each Bus Coupler has status inputs that can be found in the hardware tree.

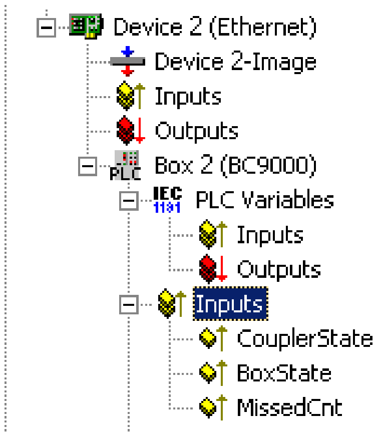


Fig. 127: Status and diagnostic inputs

Coupler state

Value	Meaning
0x0000	No error
0x0001	Bus Terminal error; there is a K-bus error
0x0002	Configuration error; the parameterized configuration does not match the actual configuration

MissedCnt

Ideal configuration:
The task time is always longer than the Ethernet propagation delay. An Ethernet telegram is transmitted at the beginning of the task, and it returns it to the PC again after a period of time, t_{Eth}. If the time t_{Eth} is always smaller than the task time that has been set, the value in the *MissedCnt* counter remains constant.

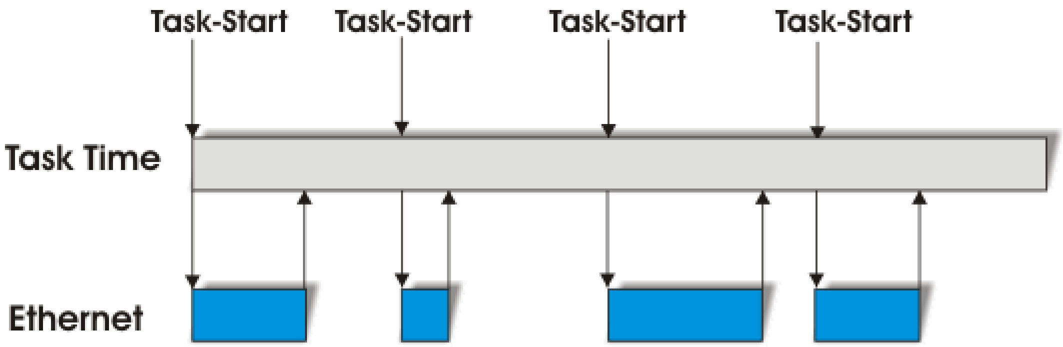


Fig. 128: Optimal Ethernet runtime

If, after the task time has elapsed, an Ethernet telegram has still not arrived at the PC, only reaching it after the next task has started, then TwinCAT will continue to work with the old input data. In addition, the *MissedCnt* counter is incremented.

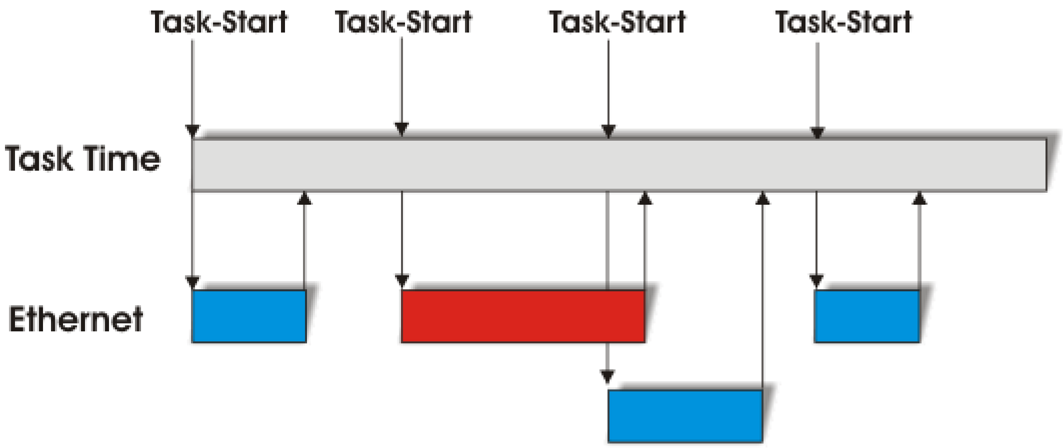


Fig. 129: Long Ethernet propagation delay with incrementing MissedCnt

This can have the following causes:

- The chosen task time is too short. Set
  - 100 ms or more for TCP,
  - 20 ms or more for UDP.
- Too many collisions in the network: use switches instead of hubs!
- The bus load is too high: change to 100 Mbaud!
- You are logged in to the BC9000, C900: this consumes additional processor power in the Bus Terminal Controller, which lengthens the response time.

The two following diagnostic inputs have different meanings, depending on the transmission type.

TCP/IP diagnosis

BoxState

Value	Meaning
0x0000	No error
0x0001	No current inputs

MissedCnt

Value	Meaning
0xyyyy	Number of telegrams that have not returned in time for the start of the task. This value should remain almost constant. If the value keeps rising, the task time should be lengthened.

UDP/IP diagnosis

BoxState

Value	Meaning
0x0000	No error
0x0001	No current inputs
0x0002	Outputs are written as zero
0xxxzz	xx - warning level. The value here is incremented by one each time the data is not received in time. When data is exchanged correctly, the value is decremented by one. When the maximum Warning Level (default 100) is reached, null is written to the output data, and it is only enabled again for the normal process image when the Warning Level has reached null.

MissedCnt

Value	Meaning
0xyyyy	Number of telegrams that have not returned in time for the start of the task. This value should remain almost constant. If the value keeps rising, the task time should be lengthened.

9 Appendix

9.1 BC9191 - First steps

Necessary components

- BC9191 or BC9191-0100
- Programming PC with TwinCAT 2 and Ethernet interface
- Wiring material
- A normal Ethernet cable between PC and BC9191

DIP switch settings

Set DIP switch 1 to ON, leave all other DIP switches OFF. A fixed IP address is used. Also set the double DIP switch to OFF. After setting the DIP switches, restart the BC9191.
The BC9191 then has the IP address 172.16.21.1

Set an appropriate IP address on your programmer PC. For example: 172.16.100.100 (SubNetMask 255.255.0.0).

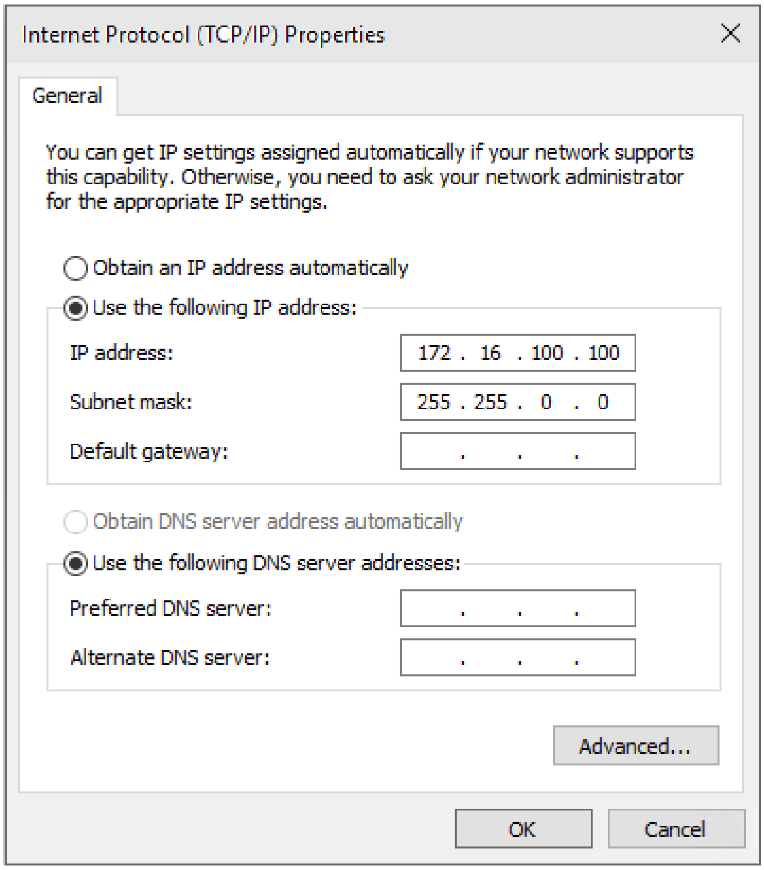


Fig. 130: Fixed IP address of the programmer PC

Use the >Ping< command to check whether TCP/IP communication is available. Open the prompt (DOS box) and enter the following: Ping 172.16.21.1. If the ping is successful you can continue. If the ping is not answered the communication is faulty. Then check the Ethernet connection:

- proper Ethernet cable
- the DIP switch of the controller is switched on correctly
- the IP address is changed on the programmer PC.

Finding the Bus Terminal Controller with the TwinCAT System Manager

Start the System Manager. Select a new target system.

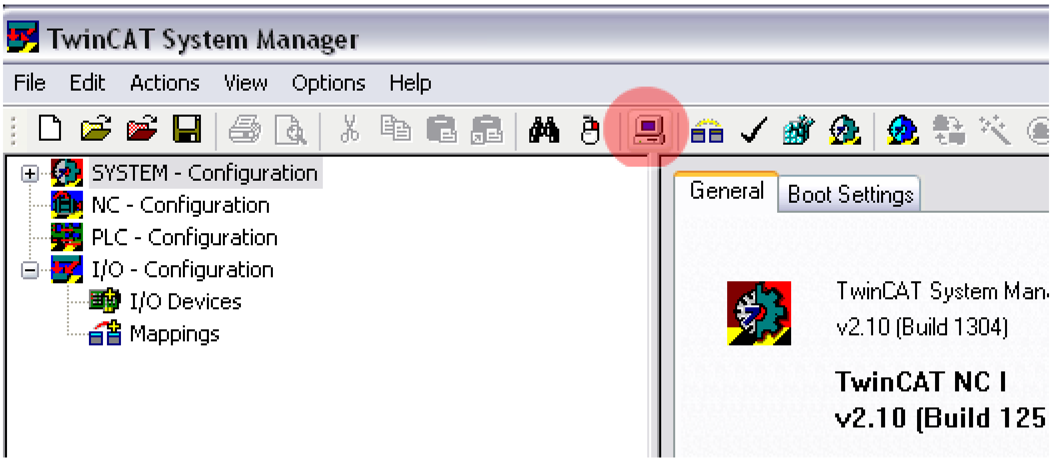


Fig. 131: System Manager

Click *Search (Ethernet)...*

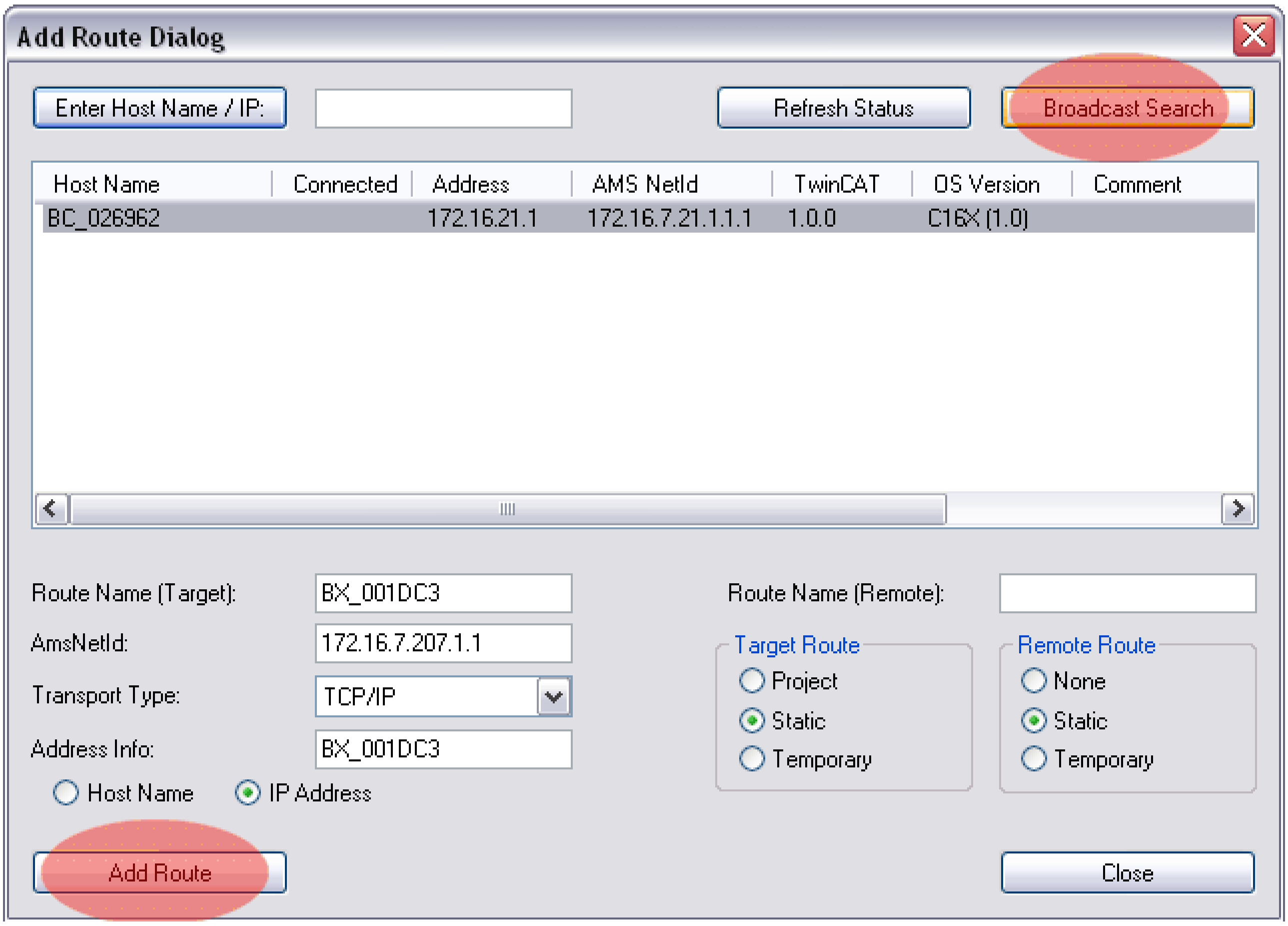


Fig. 132: Finding the Bus Terminal Controller with the TwinCAT System Manager

Then click **Broadcast Search**. The Bus Terminal Controller now appears in the window. The host name is formed from the Bus Terminal Controller plus the last 3 bytes of the MAC ID and is therefore unique in your system. The MAC ID is printed on a label below the Bus Terminal Controller. Set the options button to *IP Address* and click *Add Route*. The Connected field should show a cross or "X". If a password query appears confirm with OK (leave blank). The password function is not implemented in the Bus Terminal Controller.

You can now select the Bus Terminal Controller. Click on the Bus Terminal Controller and confirm with OK.

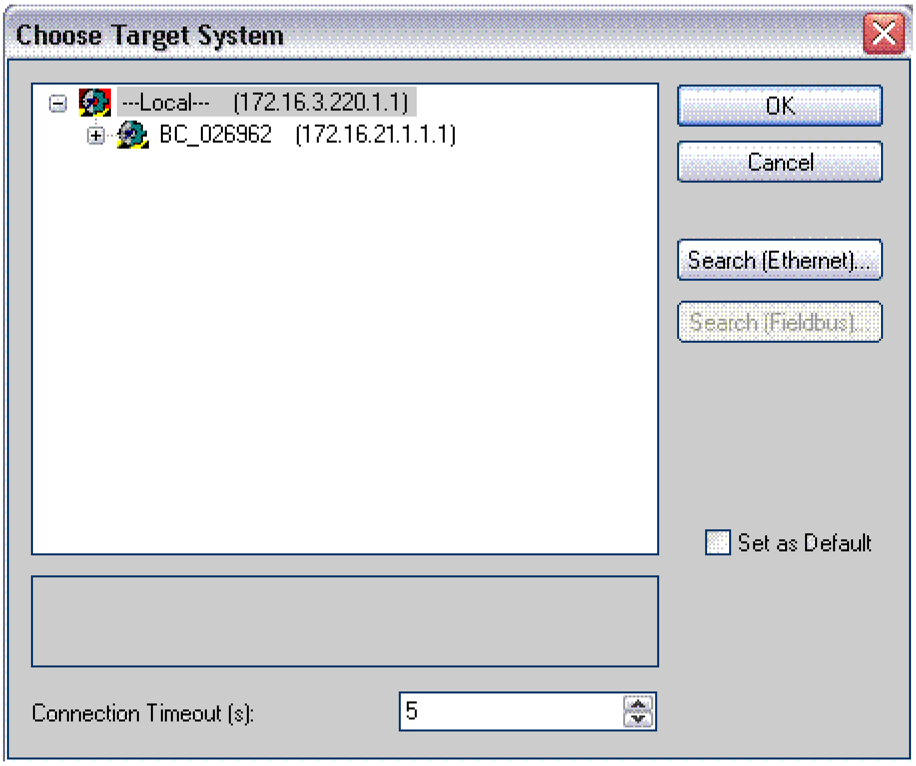


Fig. 133: Choose Target System

Your Bus Terminal Controller with name and IP address is shown in red at bottom left in the System Manager. The field next to it should be blue and show Config Mode. If this is the case you can now scan the device. Right-click on *I/O Devices* then **Scan Devices**.

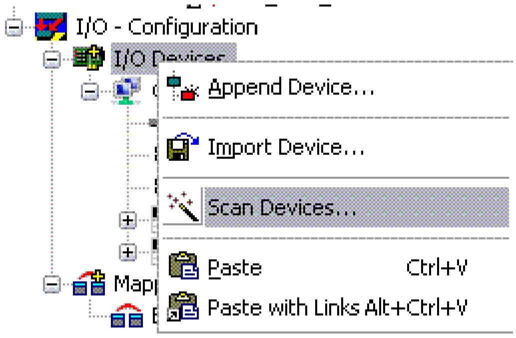


Fig. 134: Scan Devices

The System Manager should find 2 devices. The K-bus interface "Device 1 (BX-BK)" and the Ethernet interface "Device 2 (BC9191-0100)".

BC9191 in the System Manager

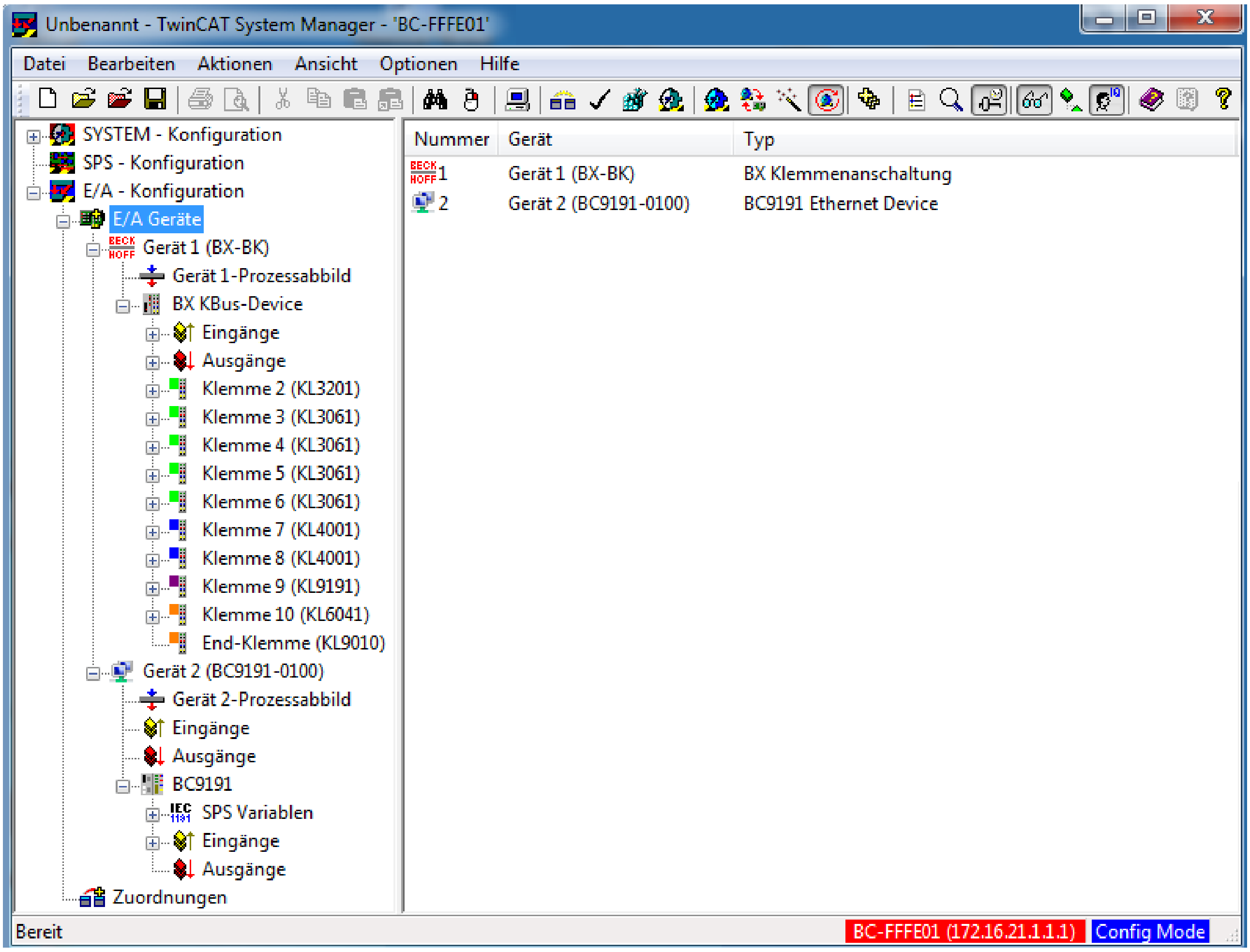


Fig. 135: Display of the BX9191 in the System Manager

Under device 1 (BX-BK) you can configure the terminal connection and the integrated periphery.

Under device 2 (BC9191-0100) you can make the settings for the Ethernet interface and the PLC variables.

If you do not want to use the pre-installed PLC program for a three-stage fan controller (see chapter 6), you can insert your own program created with TwinCAT 2 PLC Control under PLC Configuration.

9.2 General operating conditions

The following conditions must be met in order to ensure flawless operation of the fieldbus components.

Environmental conditions

Operation

The components may not be used without additional protection in the following locations:

- in difficult environments, such as where there are corrosive vapors or gases, or high dust levels
- in the presence of high levels of ionizing radiation

Condition	Permissible range
Permissible ambient temperature during operation	0 °C ... +55 °C
Installation position	Vertical on mounting rail mounted horizontally on the wall
Vibration resistance	conforms to EN 60068-2-6
Shock resistance	conforms to EN 60068-2-27
EMC immunity	conforms to EN 61000-6-2
Emission	conforms to EN 61000-6-4

Transport and storage

Condition	Permissible range
Permissible ambient temperature during storage	-25 °C ... +85 °C
Relative humidity	95 %, no condensation
Free fall	up to 1 m in the original packaging

Protection classes and types

Condition	Permissible range
Protection class in accordance with IEC 536 (VDE 0106, Part 1)	A protective conductor connection to the profile rail is necessary!
Protection class conforms to IEC 529	IP20 (protection against contact with a standard test finger)
Protection against foreign objects	Less than 12 mm in diameter
Protection against water	no protection

Component identification

Every supplied component includes an adhesive label providing information about the product's approvals. Example for the BK2000 Bus Coupler:

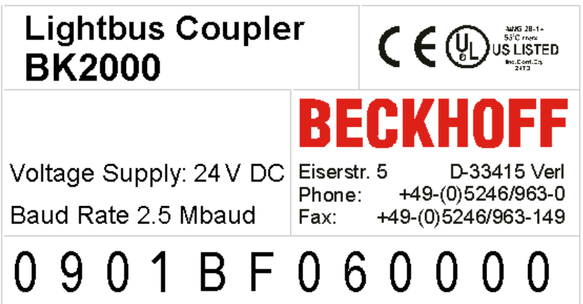


Fig. 136: BK2000 - label

The following information is printed on the label:

Printed item	Meaning for this label
Precise product identification	BK2000 Lightbus Coupler
Supply voltage	24 V_{DC}
Data transfer rate	2.5 Mbaud
Vendor	Beckhoff Automation GmbH
CE mark	Conformity mark
UL mark	Mark for UL approval. UL stands for the Underwriters Laboratories Inc., the leading certification organization for North America, based in the USA. C = Canada, US = USA, LISTED 22ZA (the test results can be inspected under this entry)
Production identification	From left to right, this sequence of characters indicates the week of production (2 characters), the year of production (2 characters), the software version (2 characters) and hardware version (2 characters), along with any special indications (4 characters). In this case the device is a BK2000 - produced in calendar week 9 - of 2021 - with firmware version BF - in hardware version 6 - without special designation

9.3 Test standards for device testing

EMC

EMC immunity

EN 61000-6-2

EMC emission

EN 61000-6-4

Vibration / shock resistance

Vibration resistance

EN 60068-2-6

Shock resistance

EN 60068-2-27, EN 60068-2-29

9.4 Bibliography

TCP/IP

TCP/IP (German)
Configuration and operation of a TCP/IP network
by Kevin Washburn, Jim Evans
Publisher: ADDISON-WESLEY Longmann Verlag

TCP/IP (english)
Illustrated, Volume1 The Protocols
by W. Richard Stevens
Publisher: ADDISON-WESLEY Longmann Verlag

Modbus/TCP

http://www.modbus.org

TwinCAT

Beckhoff Information System:
https://infosys.beckhoff.com

9.5 List of Abbreviations

ADS

Automation Device Specification

IP20

Bus Terminal protection class

IPC

Industrial PC

I/O

Inputs and outputs

K-bus

Terminal bus

KS2000

Configuration software for Bus Terminals, Bus Couplers, Bus Terminal Controllers, Fieldbus Box modules, etc.

PE

The PE power contact can be used as a protective earth.

TwinCAT

The Windows Control and Automation Technology

9.6 Support and Service

Beckhoff and their partners around the world offer comprehensive support and service, making available fast and competent assistance with all questions related to Beckhoff products and system solutions.

Beckhoff's branch offices and representatives

Please contact your Beckhoff branch office or representative for local support and service on Beckhoff products!

The addresses of Beckhoff's branch offices and representatives round the world can be found on her internet pages: https://www.beckhoff.com

You will also find further documentation for Beckhoff components there.

Beckhoff Support

Support offers you comprehensive technical assistance, helping you not only with the application of individual Beckhoff products, but also with other, wide-ranging services:

- support
- design, programming and commissioning of complex automation systems
- and extensive training program for Beckhoff system components

Hotline:	+49 5246 963 157
Fax:	+49 5246 963 9157
e-mail:	support@beckhoff.com

Beckhoff Service

The Beckhoff Service Center supports you in all matters of after-sales service:

- on-site service
- repair service
- spare parts service
- hotline service

Hotline:	+49 5246 963 460
Fax:	+49 5246 963 479
e-mail:	service@beckhoff.com

Beckhoff Headquarters

Beckhoff Automation GmbH & Co. KG

Huelshorstweg 20
33415 Verl
Germany

Phone:	+49 5246 963 0
Fax:	+49 5246 963 198
e-mail:	info@beckhoff.com
web:	https://www.beckhoff.com

List of figures

More Information:
www.beckhoff.com/BC9191

Beckhoff Automation GmbH & Co. KG
Hülshorstweg 20
33415 Verl
Germany
Phone: +49 5246 9630
info@beckhoff.com
www.beckhoff.com