Helmholz
COMPATIBLE WITH YOU

TB20 – Serial interface RS232

Manual

Version 2 / 07/06/2015 for HW 1 & FW 1 and higher

Manual order no.: 960-400-7AA31/en

Sysyteme Helmholz **GmbH** | Hannberger Weg 2 | D-91091 Großenseebach | Germany

Phone +49 9135 7380-0 | Fax +49 9135 7380-110 | info@helmholz.de | www.helmholz.com

Notes

Hannberger Weg 2, 91091 Großenseebach, Germany

To download the latest version of this manual, please visit our website at www.helmholz.de.

We welcome all ideas and suggestions.

Trademarks

Revision Record

Version	Date	Change
1	2015-05-21	First version
2	2015-07-08	Description of commissioning improved, corrections of character delay times and baud rates.

Table of contents

1 General information

This operating manual applies only to devices, assemblies, software, and services of Systeme Helmholz GmbH.

1.1 Target audience for this manual

This description is only intended for trained personnel qualified in control and automation engineering who are familiar with the applicable national standards. For installation, commissioning, and operation of the components, compliance with the instructions and explanations in this operating manual is essential.

Configuration, execution, and operating errors can interfere with the proper operation of the TB20 devices and result in personal injury as well as material or environmental damage. Only suitably qualified personnel may operate the TB20 devices!

Qualified personnel must ensure that the application and use of the products described meet all the safety requirements, including all relevant laws, regulations, provisions, and standards.

1.2 Safety instructions

The safety notes must be observed in order to prevent harm to living creatures, material goods, and the environment. The safety notes indicate possible hazards and provide information about how hazardous situations can be prevented.

1.3 Note symbols and signal words in the manual

HAZARD

If the hazard warning is ignored, there is an imminent danger to life and health of people from electrical voltage.

WARNING

If the hazard warning is ignored, there is a probable danger to life and health of people from electrical voltage.

CAUTION

If the hazard warning is ignored, people can be injured or harmed.

ATTENTION

Draws attention to sources of error that can damage equipment or the environment.

NOTE

Gives an indication for better understanding or preventing errors.

1.4 Intended use

The TB20 I/O system is an open, modular, and distributed peripheral system designed to be mounted on 35 mm DIN rails.

Communication with a higher-level control system is via a bus system / network and a TB20 bus coupler. Up to 64 modules from the TB20 range can be set up on a bus coupler. The bus couplers support hot-swapping for replacing modules during ongoing operation.

All components are supplied with a factory hardware and software configuration. The user must carry out the hardware and software configuration for the conditions of use. Modifications to hardware or software configurations which are beyond the documented options are not permitted and nullify the liability of Systeme Helmholz GmbH.

The TB20 devices should not be used as the only means for preventing hazardous situations on machinery and equipment.

Successful and safe operation of the TB20 devices requires proper transport, storage, installation, assembly, installation, commissioning, operation, and maintenance.

The ambient conditions provided in the technical specifications must be adhered to.

The TB20 systems have protection rating of IP20 and must have a control box/cabinet fitted to protect against environmental influences in an electrical operating room. To prevent unauthorized access, the doors of control boxes/cabinets must be closed and possibly locked during operation.

TB20 devices can be equipped with modules that can carry dangerously high voltages. The voltages connected to the TB20 devices can result in hazards during work on the TB20 devices.

1.5 Improper use

The consequences of improper use may include personal injuries of the user or third parties as well as property damage to the control system, the product, or environment. Use TB20 devices only as intended!

1.6 Installation

1.6.1 Access restriction

The modules are open operating equipment and must only be installed in electrical equipment rooms, cabinets, or housings.

Access to the electrical equipment rooms, cabinets, or housings must only be possible using a tool or key, and access should only be granted to trained or authorized personnel.

1.6.2 Electrical installation

Observe the regional safety regulations.

TB20 devices can be equipped with modules that can carry dangerously high voltages. The voltages connected to the TB20 devices can result in hazards during work on the TB20 devices.

1.6.3 Protection against electrostatic discharges

To prevent damage through electrostatic discharges, the following safety measures are to be followed during assembly and service work:

- Never place components and modules directly on plastic items (such as polystyrene, PE film) or in their vicinity.
- Before starting work, touch the grounded housing to discharge static electricity.
- Only work with discharged tools.
- Do not touch components and assemblies on contacts.

1.6.4 Overcurrent protection

To protect the TB20 and the supply line, a slow-blowing 8 A line protection fuse is required.

1.6.5 EMC protection

To ensure electromagnetic compatibility (EMC) in your control cabinets in electrically harsh environments, the known rules of EMC-compliant configuration are to be observed in the design and construction.

1.6.6 Operation

Operate the TB20 only in flawless condition. The permissible operating conditions and performance limits must be adhered to.

Retrofits, changes, or modifications to the device are strictly forbidden.

The TB20 is an operating means intended for use in industrial plants. During operation, the TB20 can carry dangerous voltages. During operation, all covers on the unit and the installation must be closed in order to ensure protection against contact.

1.6.7 Liability

The contents of this manual are subject to technical changes resulting from the continuous development of products of Systeme Helmholz GmbH. In the event that this manual contains technical or clerical errors, we reserve the right to make changes at any time without notice. No claims for modification of delivered products can be asserted based on the information, illustrations, and descriptions in this documentation. Beyond the instructions contained in the operating manual, the applicable national and international standards and regulations also must be observed in any case.

1.6.8 Disclaimer of liability

Systeme Helmholz GmbH is not liable for damages if these were caused by use or application of products that was improper or not as intended.

Systeme Helmholz GmbH assumes no responsibility for any printing errors or other inaccuracies that may appear in the operating manual, unless there are serious errors about which Systeme Helmholz GmbH was already demonstrably aware.

Beyond the instructions contained in the operating manual, the applicable national and international standards and regulations also must be observed in any case.

Systeme Helmholz GmbH is not liable for damage caused by software that is running on the user's equipment which compromises, damages, or infects additional equipment or processes through the remote maintenance connection and which triggers or permits unwanted data transfer.

1.6.9 Warranty

Report any defects to the manufacturer immediately after discovery of the defect.

The warranty is not valid in case of:

- Failure to observe these operating instructions
- Use of the device that is not as intended
- Improper work on and with the device
- Operating errors
- Unauthorized modifications to the device

The agreements met upon contract conclusion under "General Terms and Conditions of Systeme Helmholz GmbH" apply.

2 System overview

2.1 General information

The TB20 I/O system is an open, modular, and distributed peripheral system designed to be mounted on 35-mm DIN rails.

It is made up of the following components:

- Bus couplers
- Peripheral modules
- Power and isolation modules
- Power modules

By using these components, you can build a custom automation system that is tailored to your specific needs and that can have up to 64 modules connected in series to a bus coupler. All components have a protection rating of IP20.

2.2 The components that make up the TB20 I/O system

2.2.1 Bus coupler

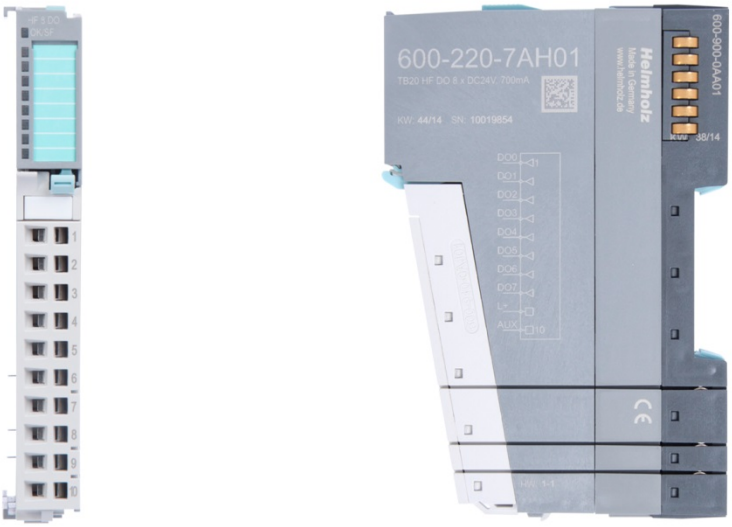
The system's bus coupler includes a bus interface and a power module. The bus interface is responsible for establishing a connection to the higher-level bus system and is used to exchange I/O signals with the automation system's CPU.

The power module is responsible for powering the coupler's electronics and all connected peripheral modules.

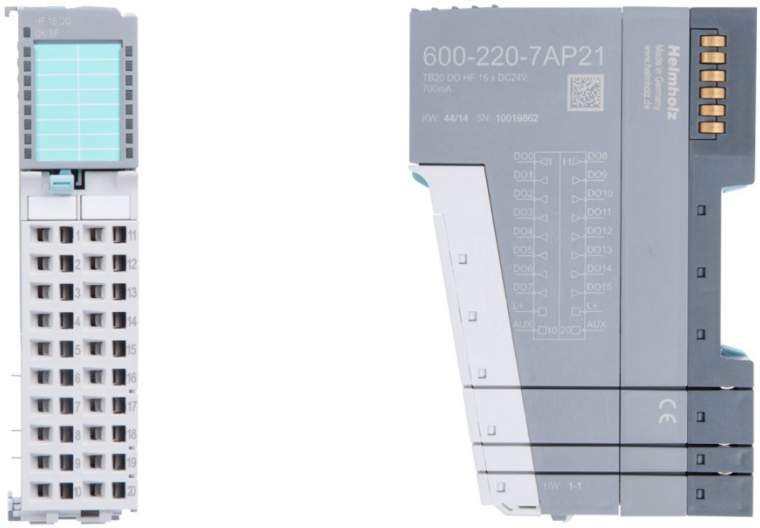
2.2.2 Peripheral modules

The system's peripheral modules are electronic components to which peripheral devices such as sensors and actuators can be connected. A variety of peripheral modules with different tasks and functions are available.

Example: Peripheral module with 10-terminal front connector



Example: Peripheral module with 20-terminal front connector



2.2.3 Power and isolation module

The system's bus coupler provides the supply voltage for the communications bus (5 V, top) and for external signals (24 V, bottom). These voltages are passed from module to module through the base modules.

Power and isolation modules make it possible to segment the power supply for external signals into individual power supply sections that are powered separately. Meanwhile, the communications bus' signals and supply voltage simply continue to be passed through, in contrast to the way they are handled by power modules (see section 2.2.4).

NOTE

Power and insulation modules have a lighter body color.

2.2.4 Power module

The system's bus coupler provides the supply voltage for the communications bus (5 V, top) and for external signals (24 V, bottom). These voltages are passed from module to module through the base modules.

Power modules make it possible to segment the power supply for both external signals and the communication bus into individual power supply sections that are powered separately.

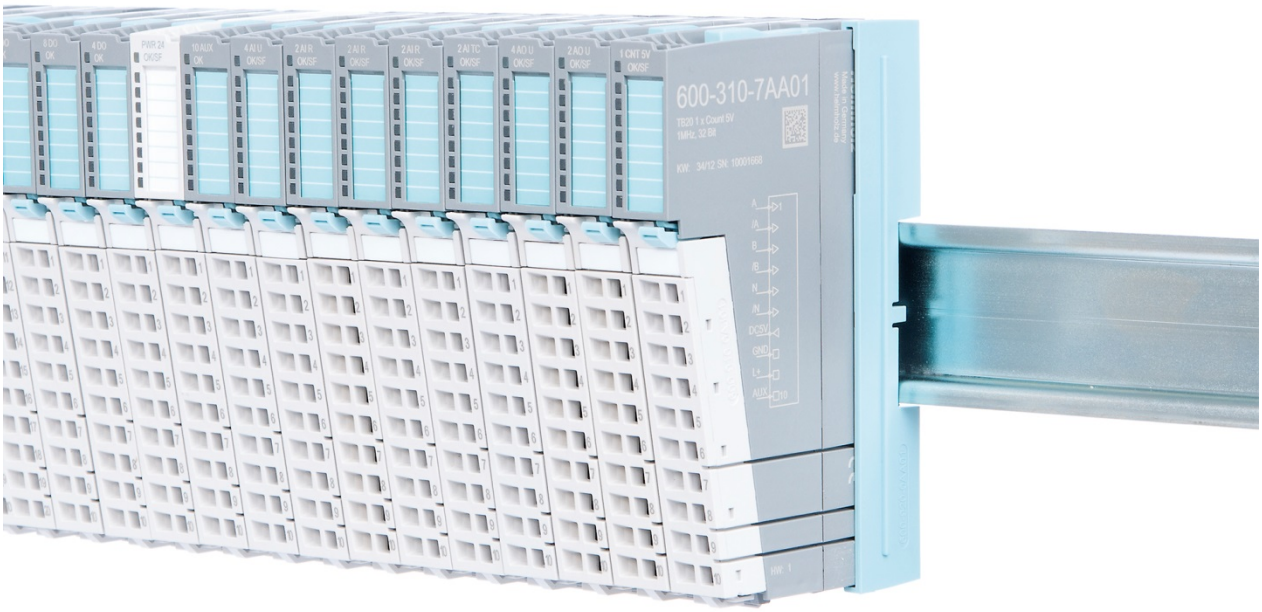
Power modules deliver all necessary power to the peripheral modules connected after them and, if applicable, all the way to the next power module or power and isolation module. A power module is required whenever the power supplied by the coupler alone is not sufficient, e.g., when there are a large number of modules with high power requirements. The "TB20 ToolBox" configuration program can be used to determine whether power modules are needed, as well as how many of them will be needed.

NOTE

Power modules have a lighter body color.

2.2.5 Final bus cover

The final bus cover protects the contacts on the last base module from accidental contact by covering the outer right-hand side of the base module.



2.2.6 Components in a module

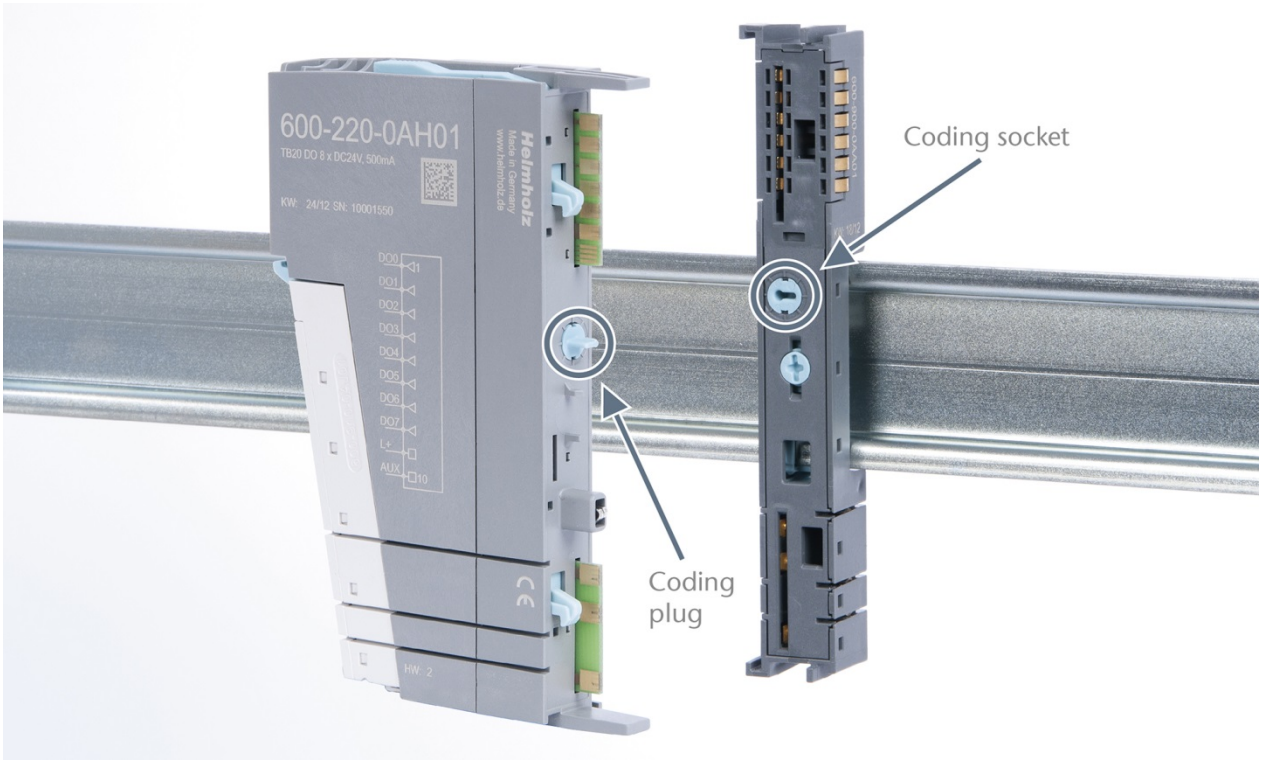
Each module consists of three parts:

- Base module
- Electronic module
- Front connector

2.2.7 Module coding

Electronic modules and base modules feature coding elements meant to prevent the wrong spare electronic modules from being plugged in during maintenance and repairs.

These coding elements consist of a coding plug on the electronic module and a coding socket on the base module (see following figure).



The coding plug and coding socket can each be in one of eight different positions. Each of these eight positions is factory-assigned to a specific type of module (Digital In, Digital Out, Analog In, Analog Out, Power) from the TB20 system. It will only be possible to plug an electronic module into a base module if the position of the coding plug and the position of the coding socket match. In the positions differ, the electronic module is mechanically blocked.

3 Installation and removal

TB20 modules can carry lethal voltage.

Before starting any work on TB20 system components, make sure to de-energize all components, as well as the cables supplying them with power! During work when the system is live, there is the risk of fatal electrocution!

Insulation must be carried out according to VDE 0100/IEC 364 and performed in accordance with applicable national standards. The TB20 IO system has protection rating IP20. If a higher protection rating is required, the system must be installed in a housing or control cabinet. In order to ensure safe operation, the ambient temperature must not exceed 60 °C.

3.1 Installation position

The TB20 I/O system can be installed in any position.

In order to achieve optimum ventilation and be able to use the system at the specified maximum ambient temperature, it will, however, be necessary to use a horizontal installation layout.

3.2 Minimum clearance

It is recommended to adhere to the minimum clearances specified when installing the coupler and modules. Adhering to these minimum clearances will ensure that:

- The modules can be installed and removed without having to remove any other system components
- There will be enough space to make connections to all existing terminals and contacts using standard accessories
- There will be enough space for cable management systems (if needed)

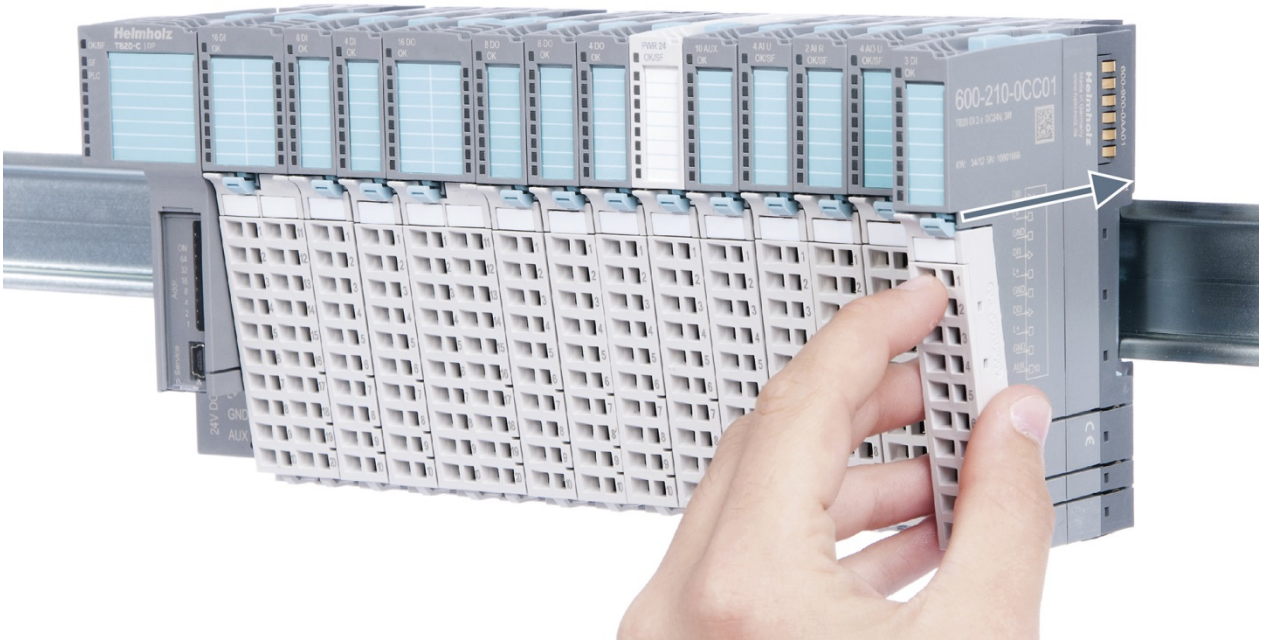
The minimum clearances for mounting TB20 components are: 30 mm on the top and on bottom and 10 mm on each side.

3.3 Installing and removing peripheral modules

3.3.1 Installation

Installing an assembled peripheral module

Place the assembled module on the DIN rail by moving it straight towards the rail. Make sure that the module engages the upper and lower guide elements of the previous module. Then push the upper part of the module towards the DIN rail until the rail fastener fastens into place on the inside snaps with a soft click.



Installing the individual parts of a peripheral module one after the other

Place the base module on the DIN rail from below in an inclined position. Then push the upper part of the base module towards the rail until the module is parallel to the rail and the rail fastener on the inside snaps into place with a soft click.

Place an electronic module with matching coding (see the “Module Coding” section on page 16) on the base module in a straight line from above and then gently push it into the base module until both modules are fully resting against each other and the module fastener snaps into place with a soft click.

Finally, place the front connector on the electronic module from below in an inclined position and then gently push it onto the electronic module until the front connector fastener snaps into place with a soft click.

3.3.2 Removal

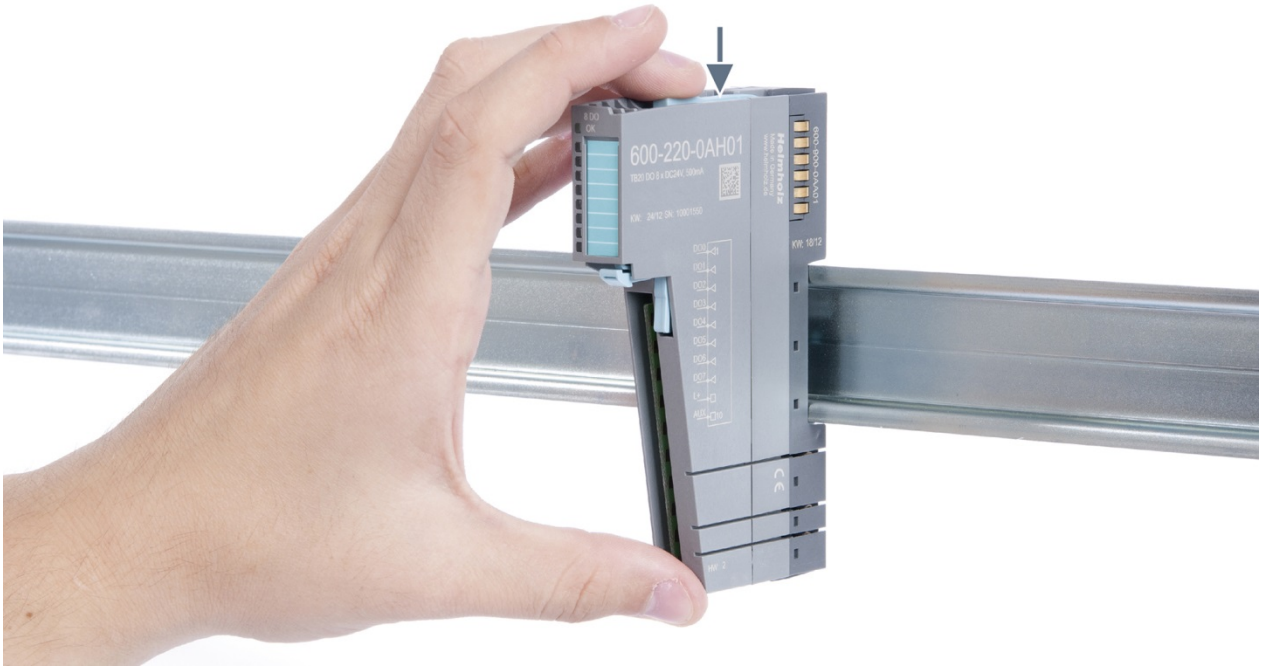
To remove a peripheral module, follow the four steps below:

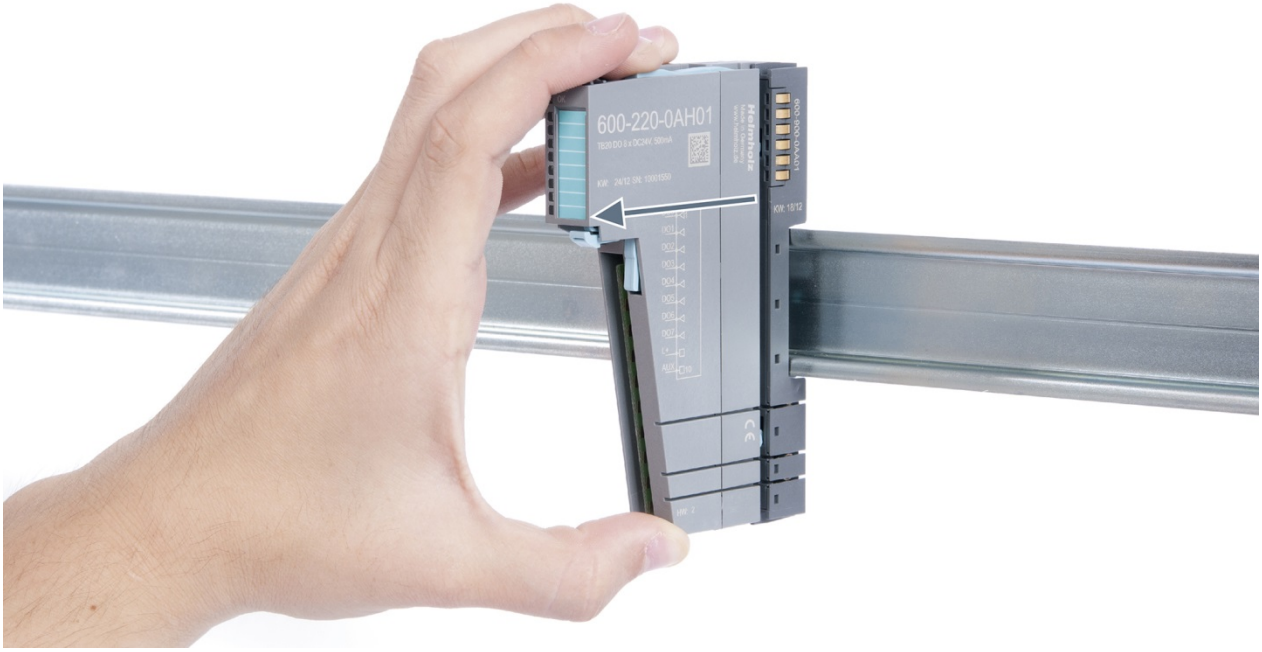
Step 1: Remove the front connector

To remove the front connector, push the tab above the front connector upwards (see the picture below). This will push out the front connector, after which you can pull it out.

Step 2: Remove the electronic module

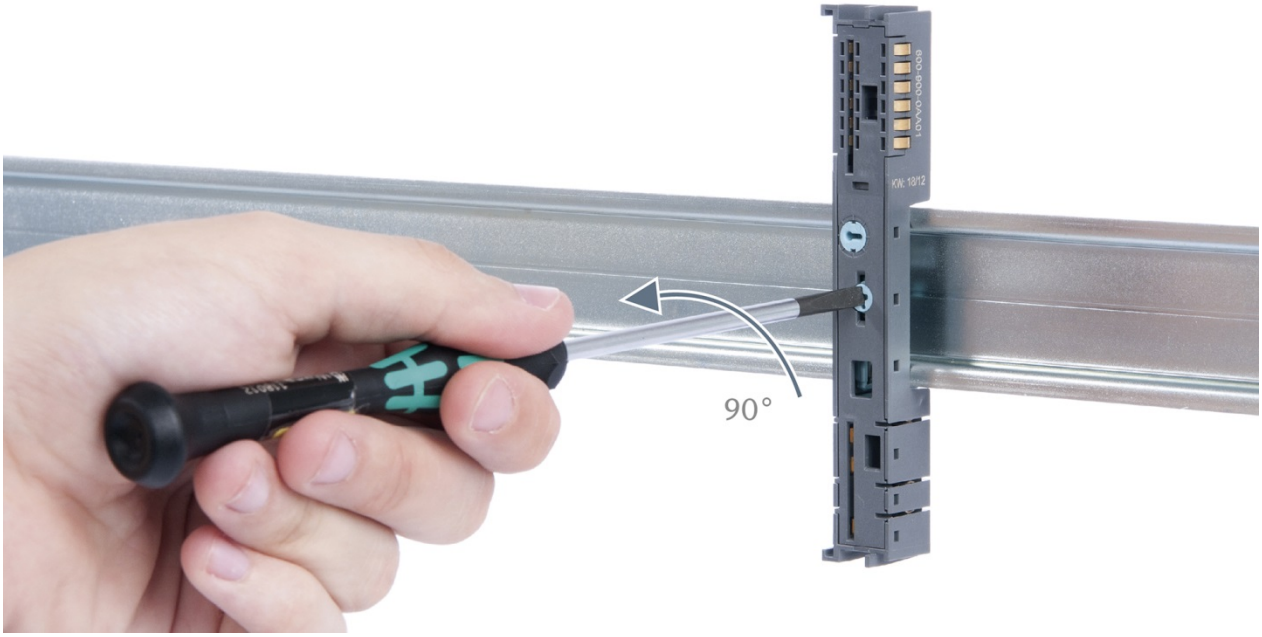
To remove the electronic module, use your middle finger to push on the lever from above and then use your thumb and index finger to pull out the electronic module while holding the lever down (see the picture below).





Step 3: Release the base module

Use a screwdriver to release the base module. by turning the locking mechanism 90° counterclockwise.



Step 4: Remove the base module

Remove the base module by pulling it towards you.

3.4 Replacing an electronic module

The procedure for replacing the electronic module on a peripheral module consists of four steps.

If you need to replace the electronic module while the system is running, make sure to take into account the general technical specifications for the bus coupler being used.

TB20 modules can carry lethal voltage.

Before starting any work on TB20 system components, make sure to de-energize all components, as well as the cables supplying them with power! During work when the system is live, there is the risk of fatal electrocution!

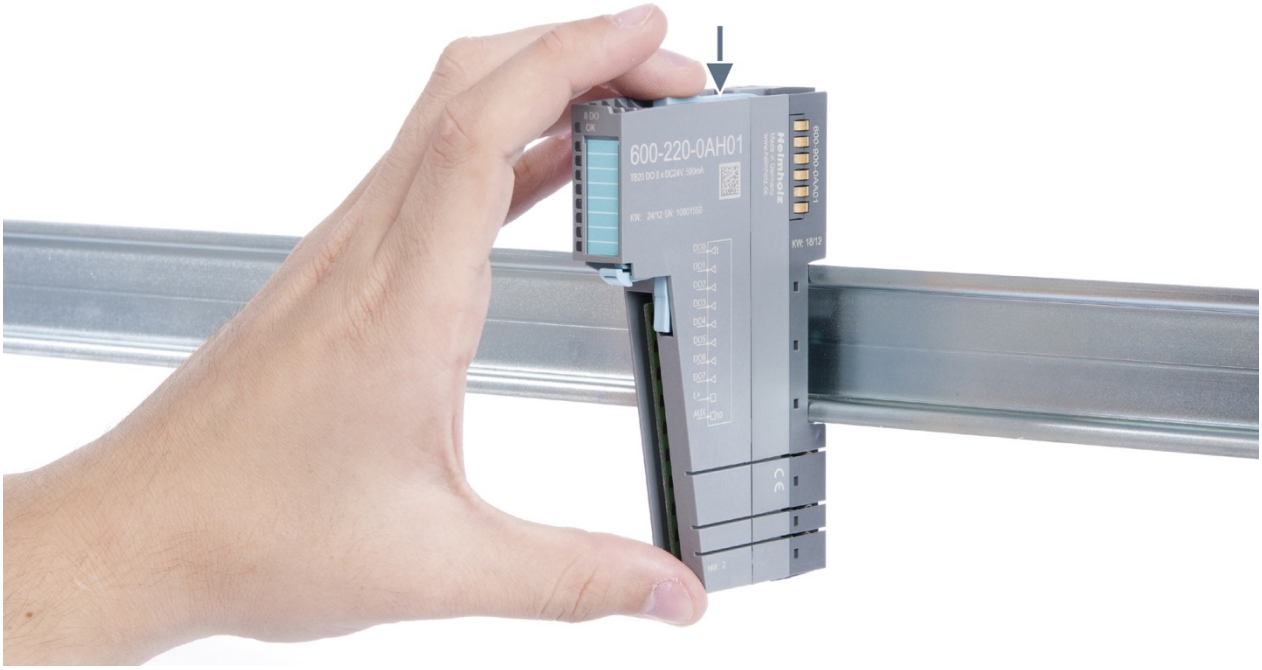
Note the wiring diagram of the system and switch off dangerous voltages before starting work!

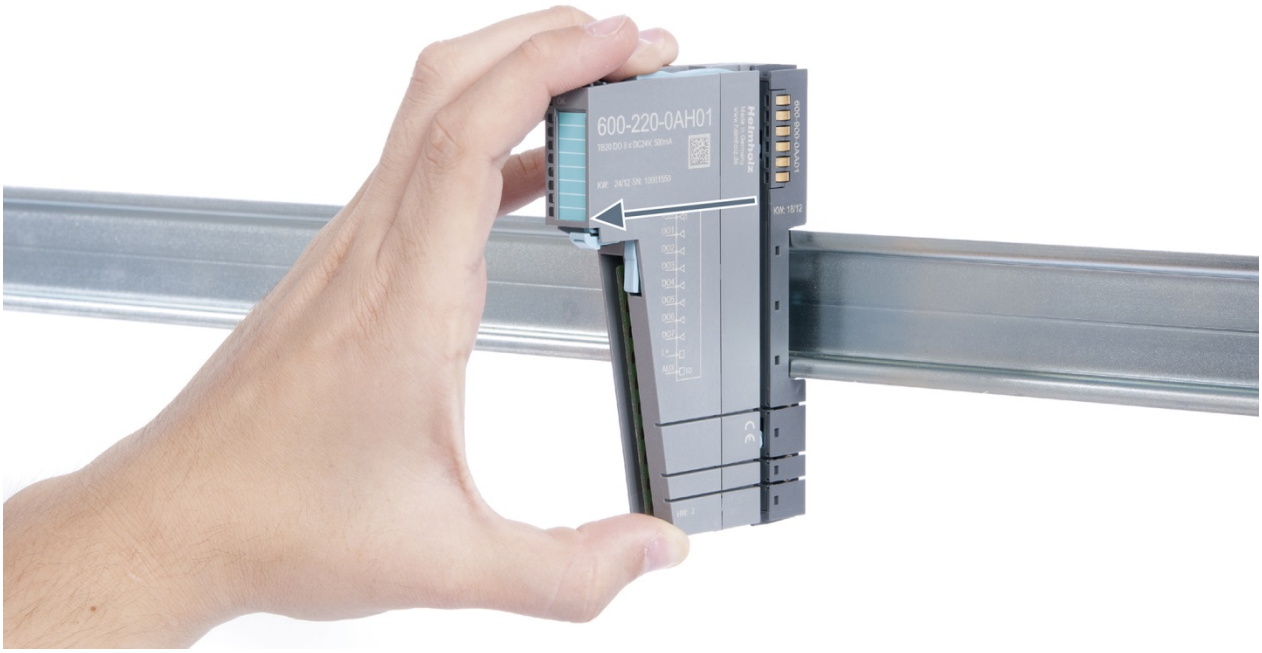
Step 1: Remove the front connector

To remove the front connector, push the tab above the front connector upwards (see the picture below). This will push out the front connector, after which you can pull it out.

Step 2: Remove the electronic module

To remove the electronic module, use your middle finger to push on the lever from above and then use your thumb and index finger to pull out the electronic module while holding the lever down (see the picture below).





Step 3: Plug in a new electronic module

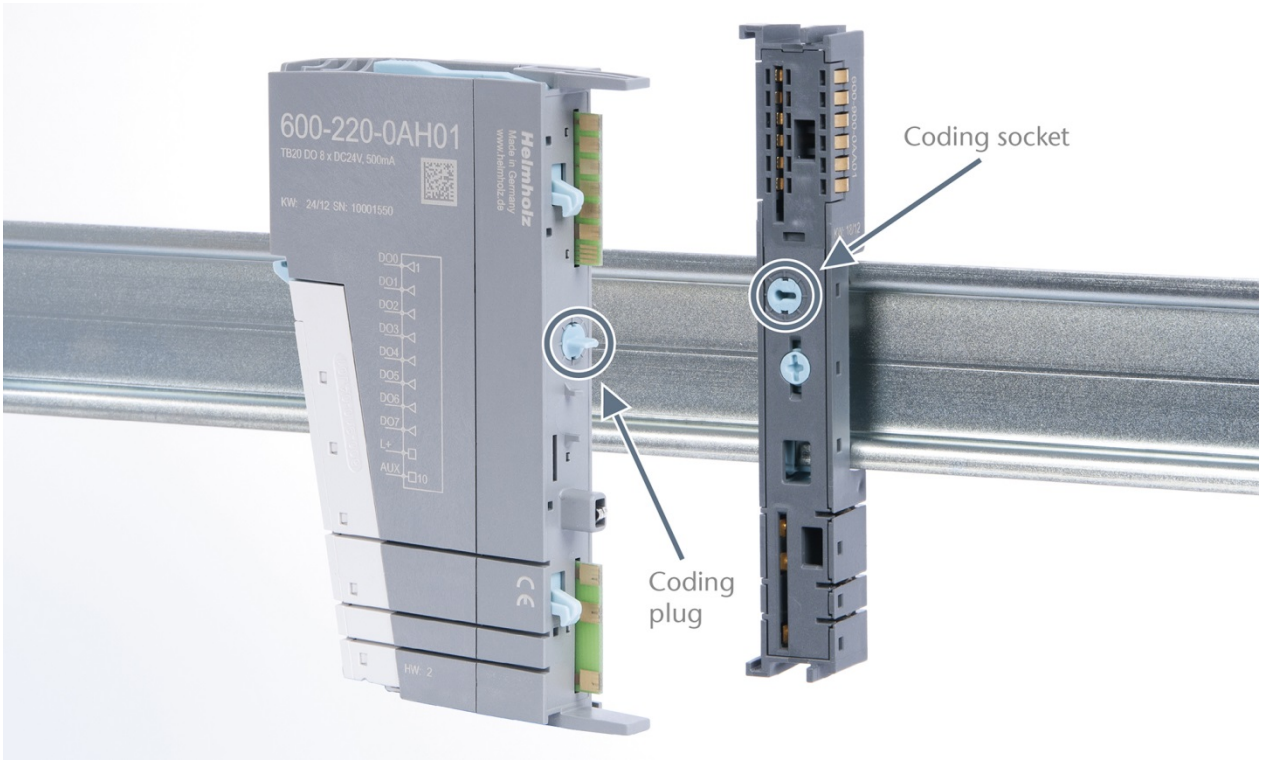
ATTENTION

The electronic module must be snapped into place on the base module with a single continuous movement. If the electronic module is not snapped into place firmly and straight on the base module, bus malfunctions may occur.

ATTENTION

If the electronic module cannot be plugged into the base module, check whether the coding elements on the electronic module and base module (see figure below) match. If the coding elements on the electronic module do not match those on the base module, you may be attempting to plug in the wrong electronic module.

For more information on coding elements, please consult section 2.2.7.



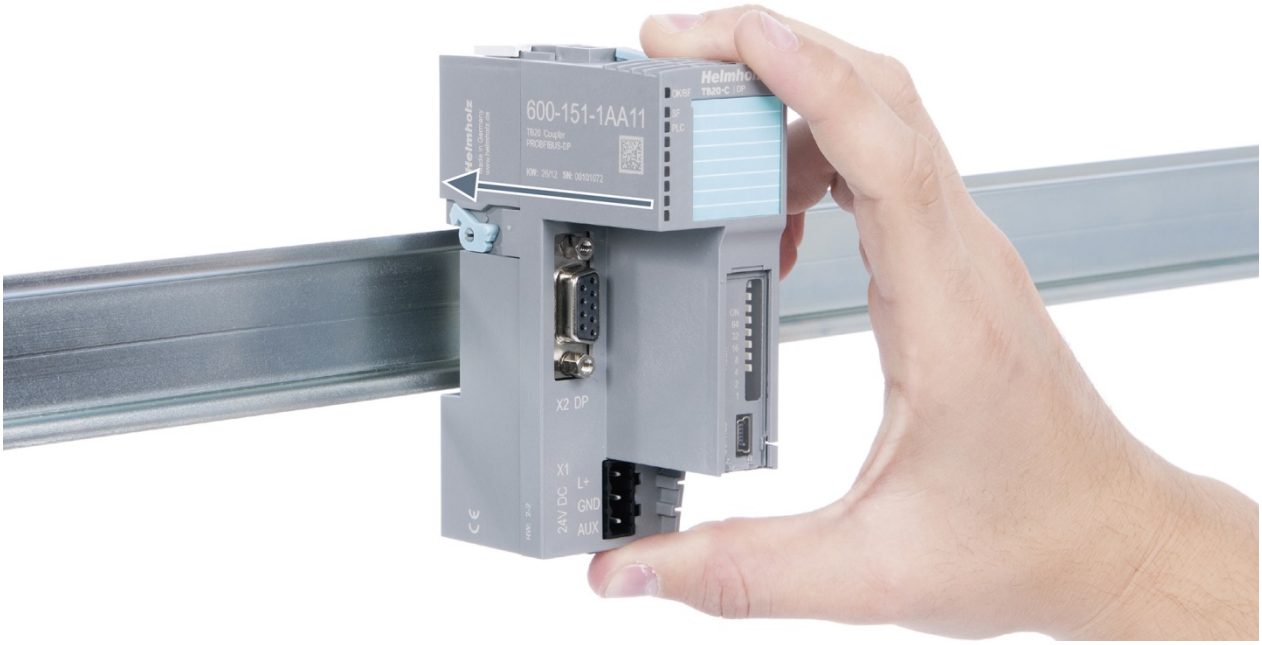
Step 4: Plug in the front connector

3.5 Installing and removing the coupler

3.5.1 Installation

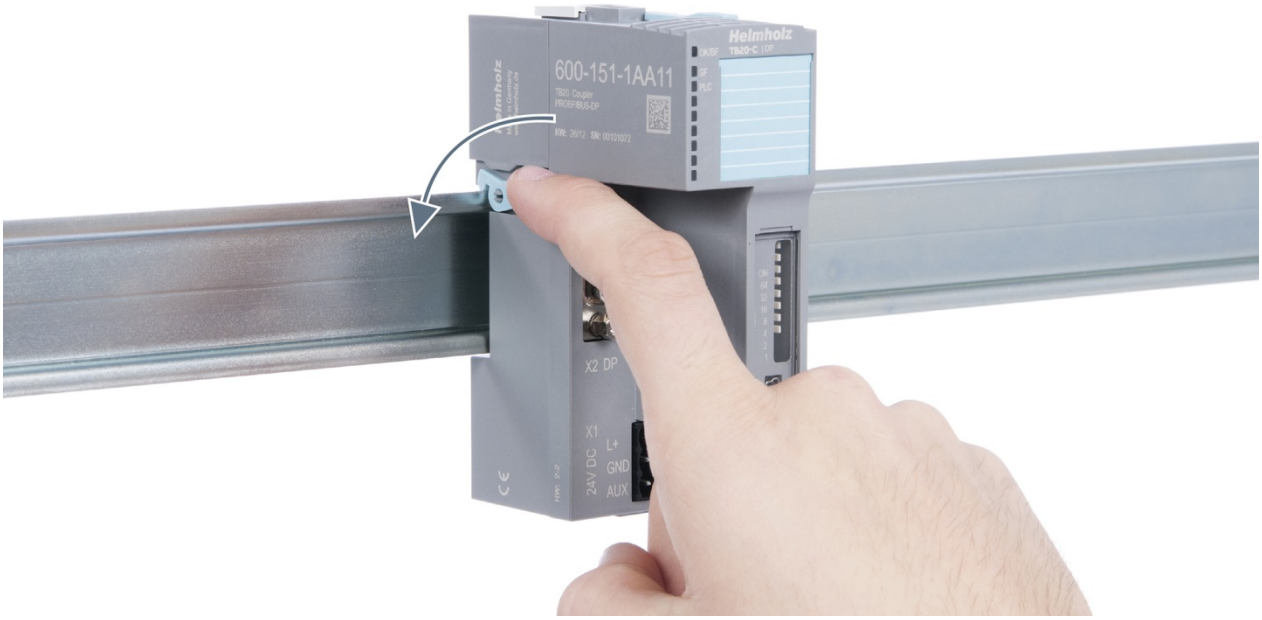
Step 1: Place the coupler on the DIN rail

Place the coupler, together with the attached base module, on the DIN rail by moving it straight towards the rail. Then push the coupler towards the rail until the base module's rail fastener snaps into place with a soft click.



Step 2: Secure the coupler on the DIN rail

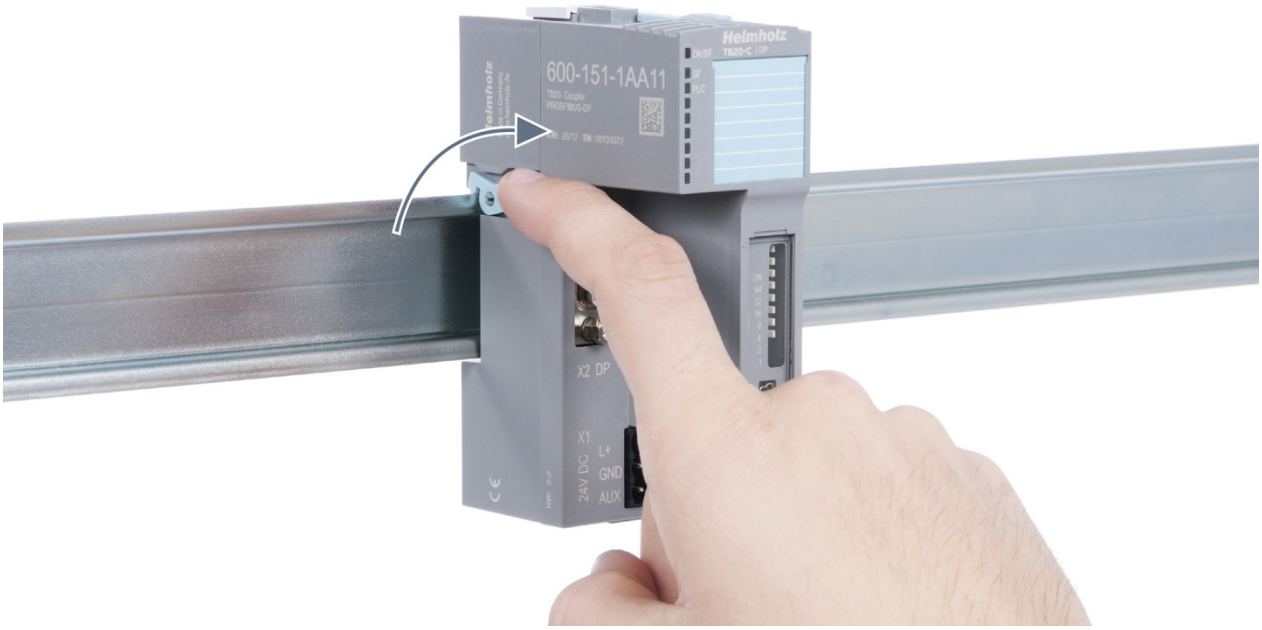
Use the locking lever on the left side of the coupler to lock the coupler into position on the DIN rail.



3.5.2 Removal

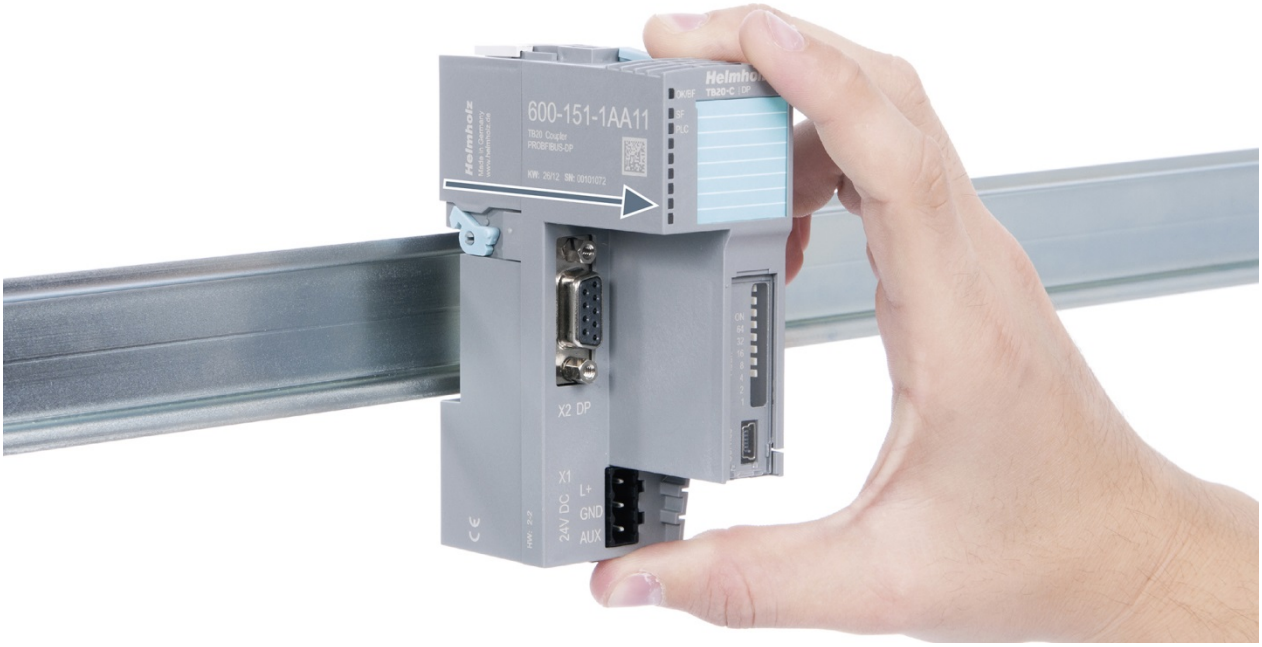
Step 1: Release the locking mechanism

Release the locking lever on the left side of the coupler in order to disengage it from the DIN rail.



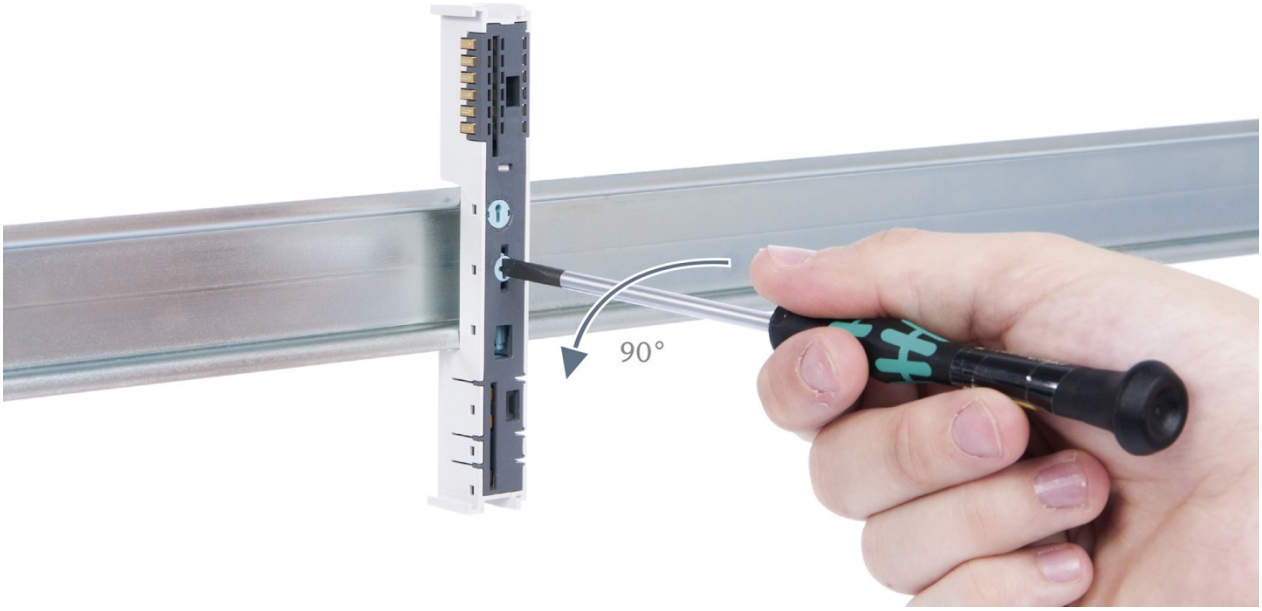
Step 2: Remove the coupler

Use your middle finger to push on the lever from above and use your thumb and index finger to pull out the coupler while holding the lever down.



Step 3: Release the base module

Use a screwdriver to release the base module



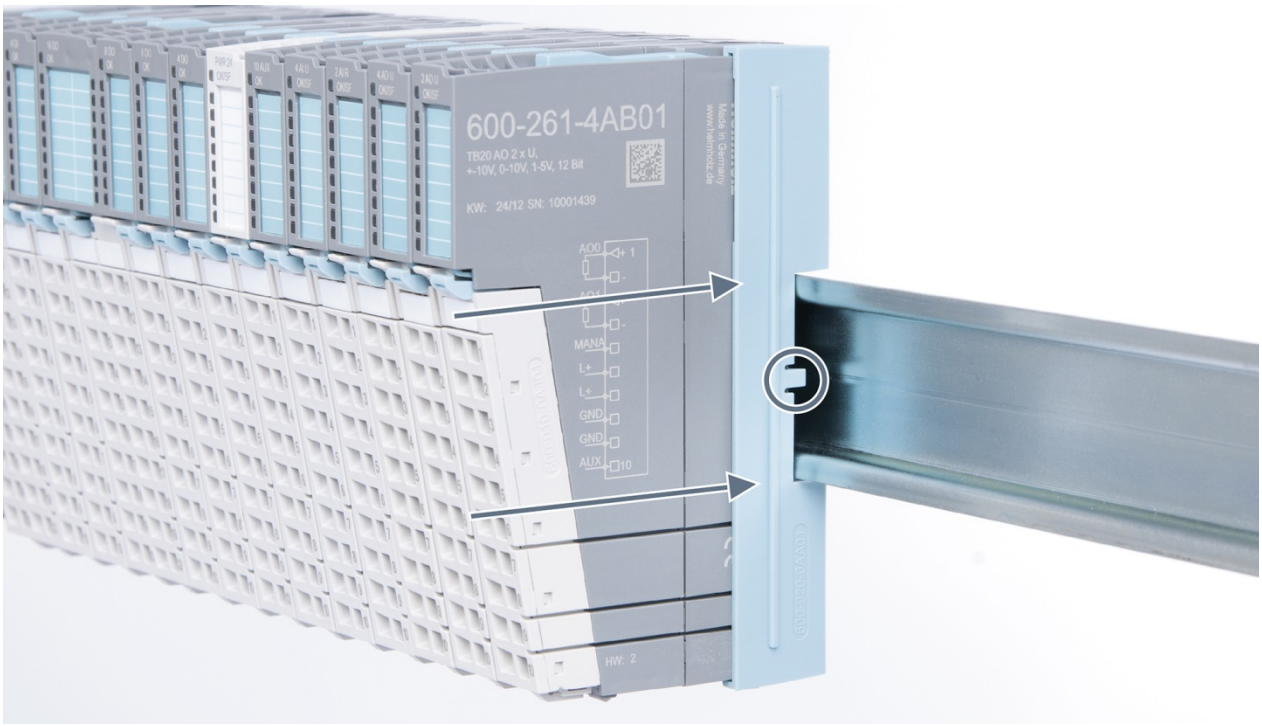
Step 4: Remove the base module

Remove the base module by pulling it towards you.

3.6 Installing and removing the final bus cover

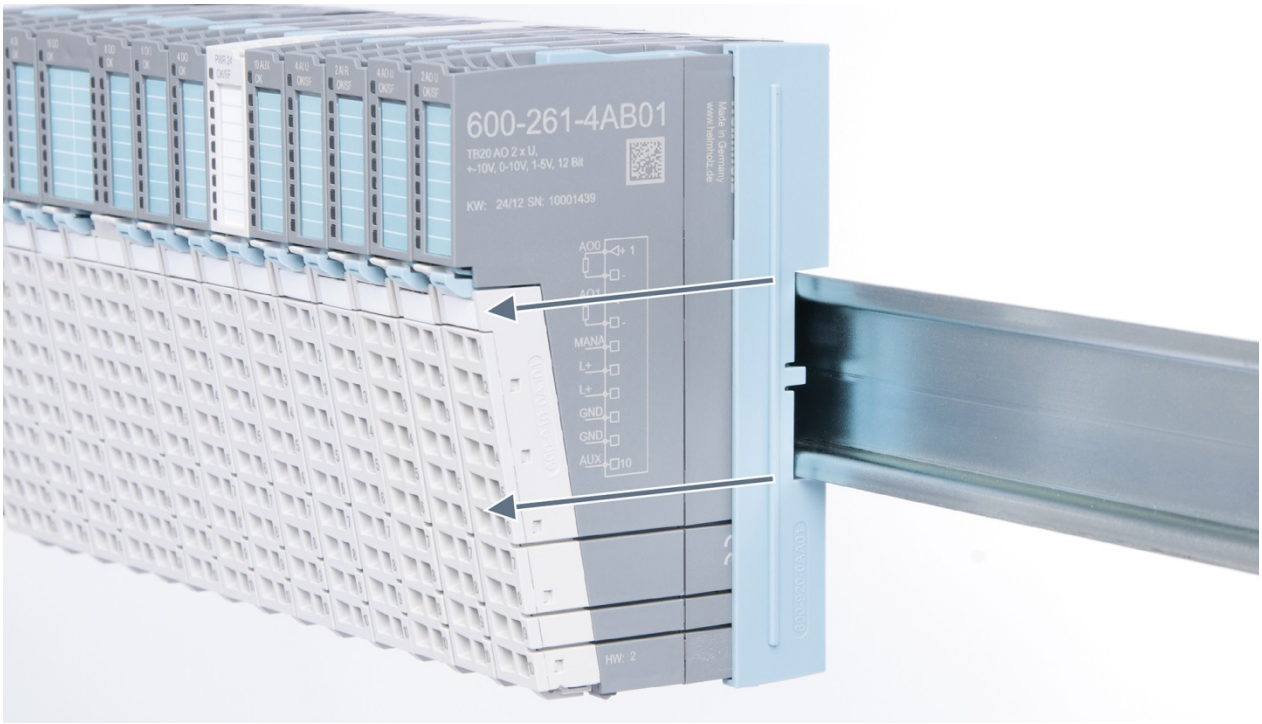
3.6.1 Installation

Slide the final bus cover onto the last module along the case, starting from the end with the front connector and moving towards the DIN rail, until the cover covers the base module's contacts and the tab snaps into place.



3.6.2 Removal

Pull the final bus cover along the module's case and away from the DIN rail in order to remove it from the module.



4 Configuration/wiring

4.1 EMC/safety/shielding

The TB20 IO system complies with EU Directive 2004/108/EC (“Electromagnetic Compatibility”).

One effective way to protect against disturbances caused by electromagnetic interference is to shield electric cables, wires, and components.

When putting together the system and routing of the required cables, make sure to fully comply with all standards, regulations, and rules regarding shielding (please consult the relevant guidelines and documents published by the PROFIBUS User Organization as well). All work must be done professionally!

Shielding faults can result in serious malfunctions, including the system’s failure.

To ensure electromagnetic compatibility (EMC) in your control cabinets in electrically harsh environments, the following EMC rules are to be observed in the design:

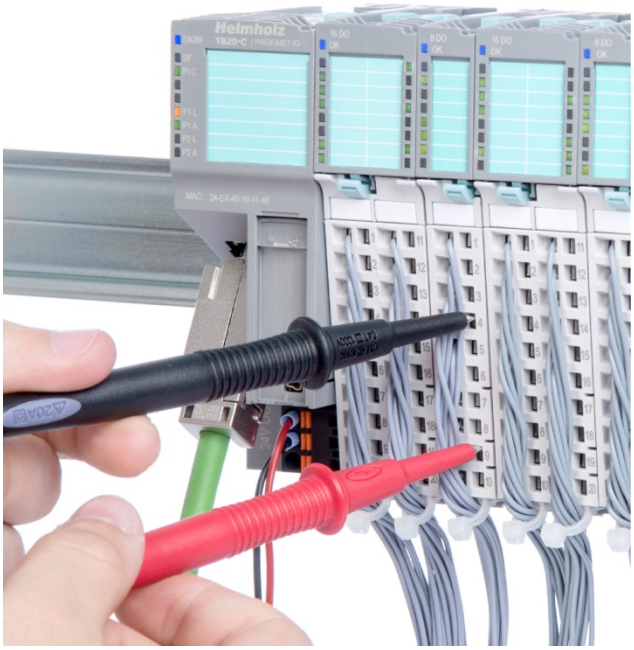
- All metal parts of the cabinet are to be connected with each other over a large area with good conductivity (no paint on paint). Where necessary, use contact washers or serrated washers.
- The cabinet door must be connected to the ground straps (top, middle, bottom) over as short a distance as possible.
- Signal cables and power cables are to be laid separated spatially by a minimum distance of 20 cm from each in order to avoid coupling paths.
- Run signal lines only from one level into the cabinet if possible.
- Unshielded cables in the same circuit (outgoing and incoming conductors) must be twisted if possible.
- Contactors, relays, and solenoid valves in the closet, or in adjacent cabinets if applicable, must be provided with quenching combinations; e.g., with RC elements, varistors, diodes.
- Do not lay wires freely in the closet; instead, run them as closely as possible to the cabinet housing or mounting panels. This also applies to reserve cables. These must be grounded on at least one end, and it is better if they are grounded on both ends (additional shielding effect).
- Unnecessary line lengths should be avoided. Coupling capacitances and inductances are kept low in this way.
- Analog signal lines and data lines must be shielded.

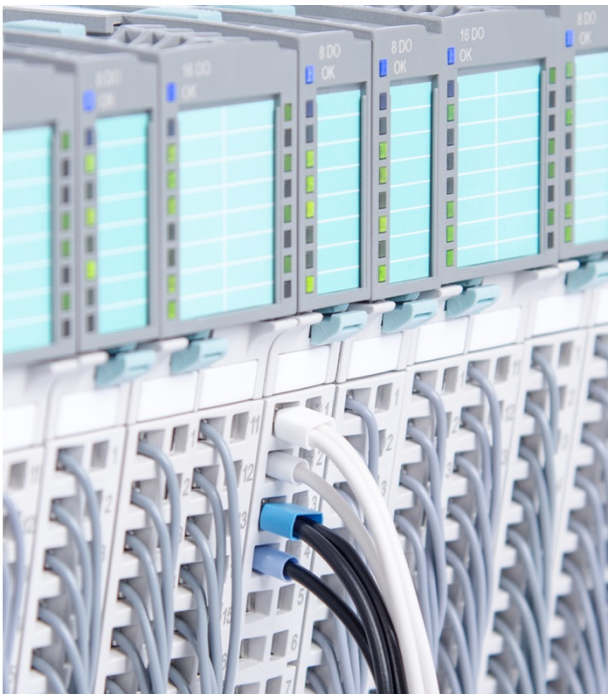
4.2 Front connector

The front connector's spring-clamp terminals are designed for a cross-sectional cable area of up to 1.5 mm² (16–22 AWG) with or without ferrules.

It is also possible, for example, to connect two 0.75 mm² wires to a single spring-type terminal, provided the maximum cross-sectional cable area of 1.5 mm² per terminal is not exceeded.

The cables can be attached to the underside of the front connector with a cable tie.





4.3 Wiring the coupler

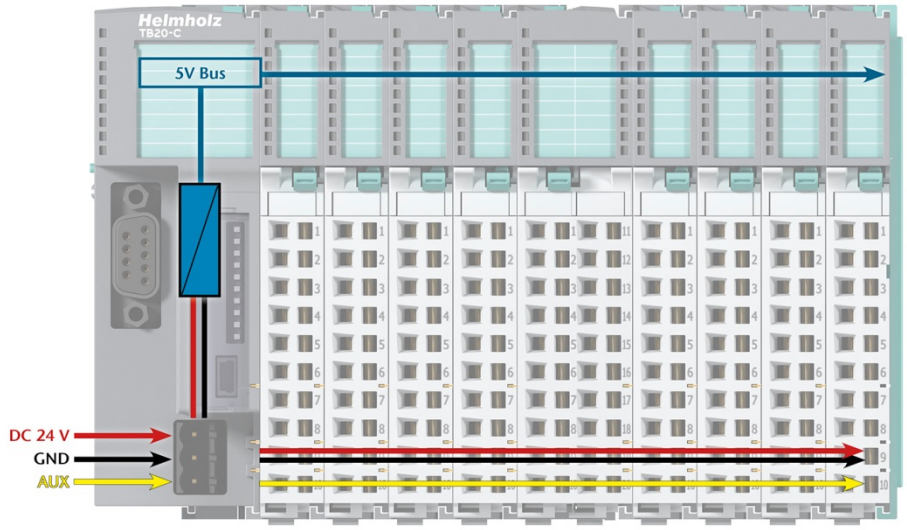
A power supply unit is integrated into the bus coupler. The power supply unit is responsible for powering the peripheral modules connected to the coupler.

In turn, it draws its own power from the three-pin connector on the front (24 VDC, GND, AUX).

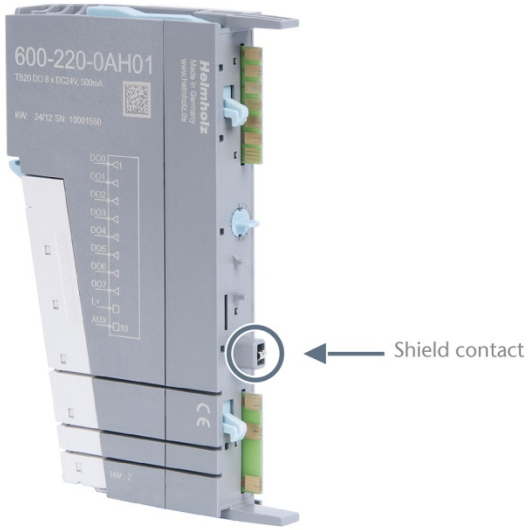
The 24 V connector is used to power two buses:

- The power bus used to power the I/O components (24 VDC, GND, AUX)
- The communications bus used to power the electronics in the peripheral modules

The AUX pin can be used to connect and use an additional voltage potential. Every peripheral module has an AUX terminal on its front connector (the bottommost terminal, i.e., terminals 10 and 20).

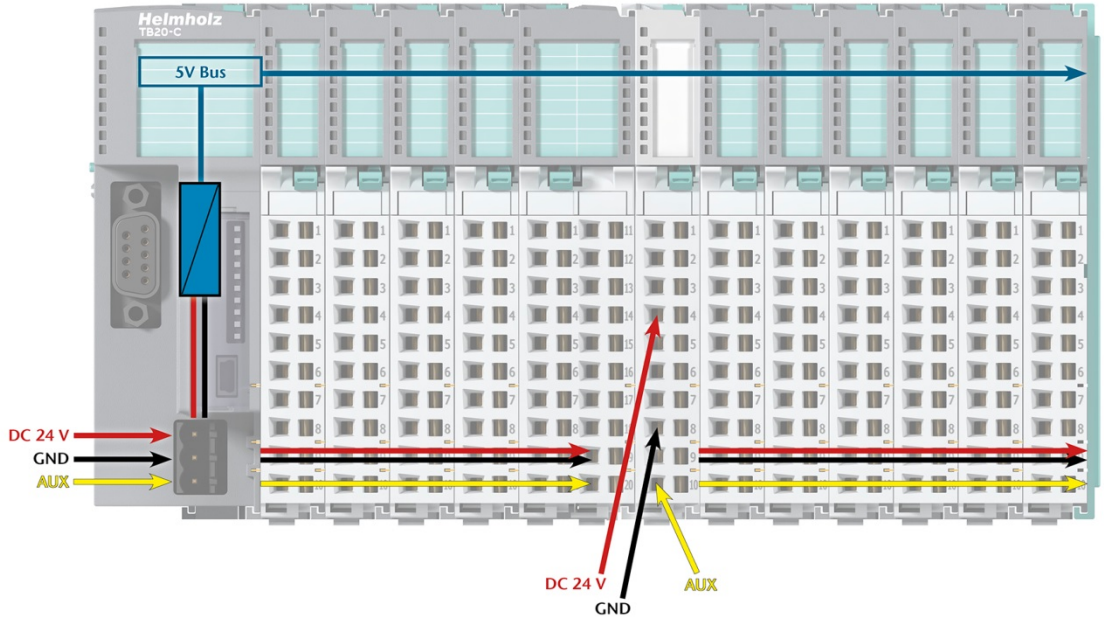


The coupler and the modules are grounded via the shield contact to the DIN rail. The DIN rail must be grounded. The surface of the DIN rail must be clean and conduct electricity well.



4.4 Using power and isolation modules

Power and isolation modules make it possible to segment the power supply for external signals (24 V, GND, AUX) into individual power supply sections that are powered separately.

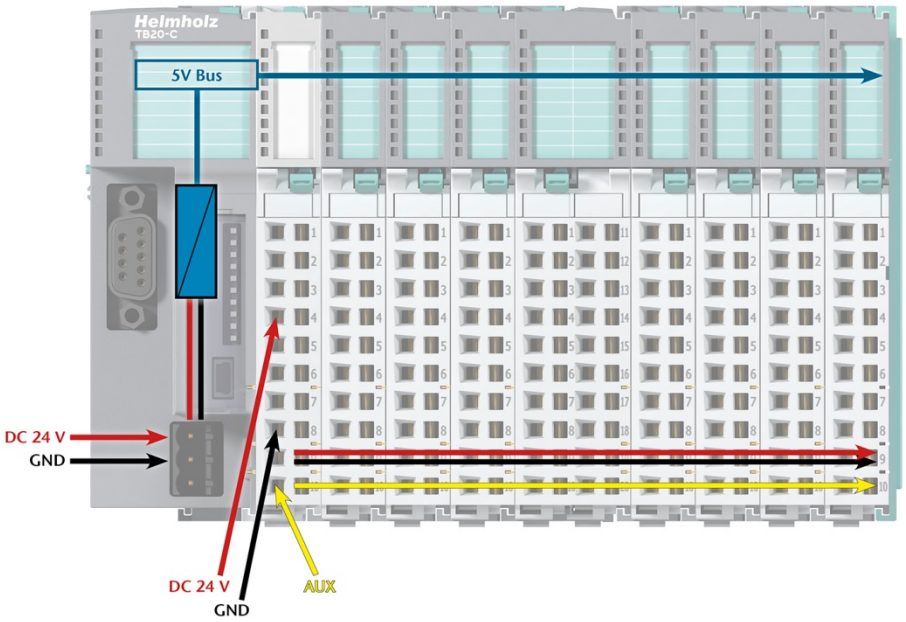


The order no. for the power and isolation module for 24 V signals is 600-710-0AA01.

Its electronic module and base module have the same light gray color as the front connector, ensuring that all power and isolation modules will stand out visually in the system and make it easy to clearly distinguish each individual power supply segment.

4.5 Separate power supply segments for the coupler and the I/O components

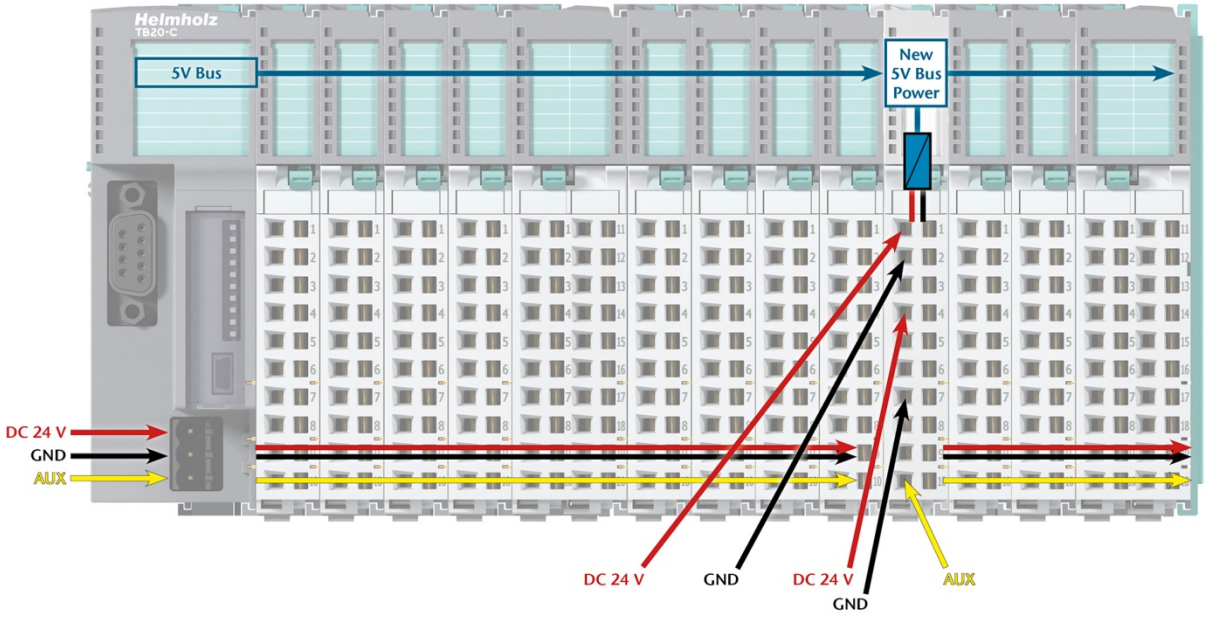
If the power supply for the coupler needs to be switched separately from the power supply for the I/O modules, a power and isolation module can be used right after the coupler.



4.6 Using power modules

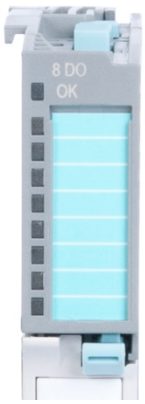
Power modules deliver all necessary power to the connected peripheral modules and, if applicable, all the way to the next power module or power and isolation module. Power modules must be used whenever the power supplied by the coupler alone is not sufficient, that is, when there are a large number of modules on the bus. The “TB20 ToolBox” parameter configuration and diagnosis program can be used to calculate a system’s total current draw.

24 VDC, GND, and AUX are fed into the terminals on the front, while the connected modules are powered through the base modules’ bus system.



The order no. for the power module is 600-700-0AA01. The electronic module of the power module is light gray like the front connector. The base module of the power module is light gray with a dark top part.

4.7 Function of the OK-LED



The topmost LED (OK-LED) on every module indicates the module's current system status.

Solid blue light:	The module is running (RUN)
Slowly flashing blue light:	The module is stopped (STOP); substitute values (if any) are being applied
Quickly flashing blue light:	The module is idle (IDLE); its parameters have not been configured yet
Solid red light:	The module is indicating a diagnostic error
Flashing red light:	The module is indicating a parameter assignment error

The red LED lights will only be shown on modules with configurable parameters or diagnosis capabilities.

4.8 Electronic nameplate

All of a TB20 module's important information can be found on its electronic nameplate. This information includes, for example, the corresponding module ID, module type, order number, unique serial number, hardware version, firmware version, and internal range of functionalities.

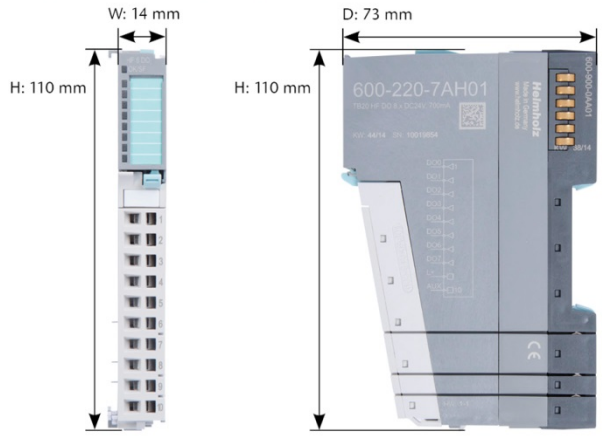
This information can be read in a number of ways, one of which is using the "TB20 ToolBox" configuration and diagnosis program. The modules' electronic nameplates not only make it possible to prevent configuration errors (setup), but also make maintenance (servicing) easier.

4.9 Fusing

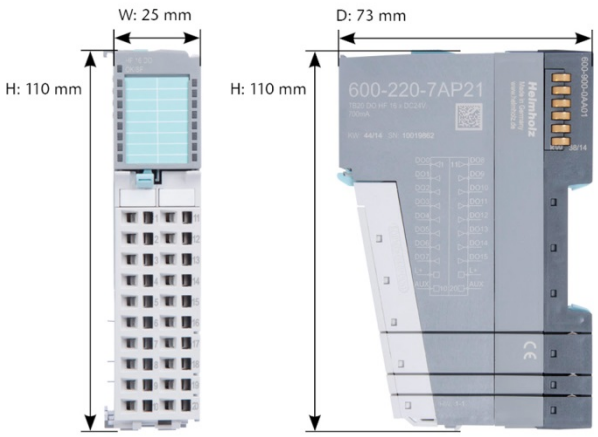
The TB20 coupler's and power modules' power supply must be externally fused with a slow-blowing fuse, maximum 8 A, appropriate for the required maximum current.

4.10 Dimensions

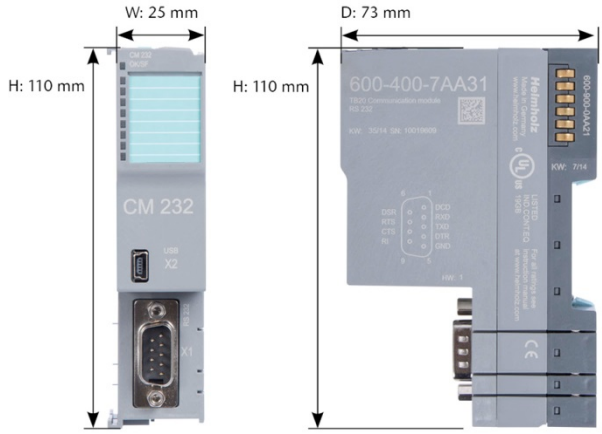
Module with standard width



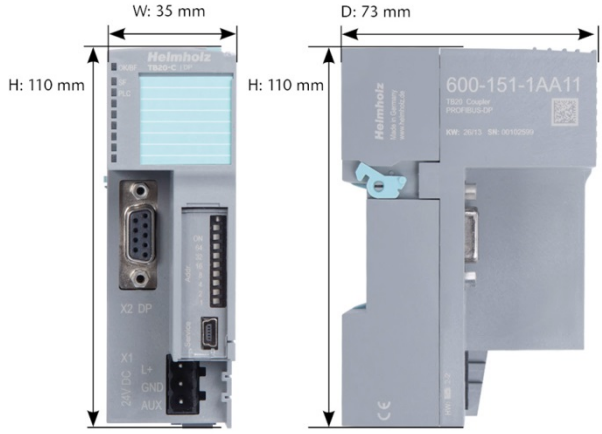
Module with double width

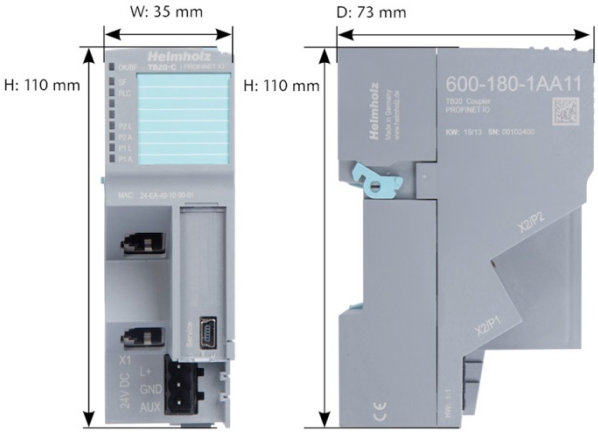


Communication Module



Bus Coupler





5 TB20 – RS232 serial interface

5.1 Characteristics

The RS232 serial interface is a module for the Helmholz fieldbus I/O system TB20. Communication with a Helmholz fieldbus coupler is via the backplane bus of the TB20 system.

Through the TB20 bus, parameter data and control commands are transmitted from the TB20 fieldbus coupler to the RS232 serial interface. The RS232 serial interface sends feedback data to the TB20 fieldbus coupler.

The RS232 serial interface allows for a point-to-point connection with various modules with serial interface. For communication with attached serial devices, the ASCII protocol is supported.

Serial communication

The RS232 serial interface handles the data transmission with the communication partner automatically. For bidirectional data traffic, the ASCII protocol is available. The modes half duplex and full duplex are possible.

Half-duplex ASCII driver
The data is transmitted alternately between the communication partners. A communication partner cannot transmit and receive simultaneously. Control characters for flow control are transmitted regardless of the transmit/receive operation.

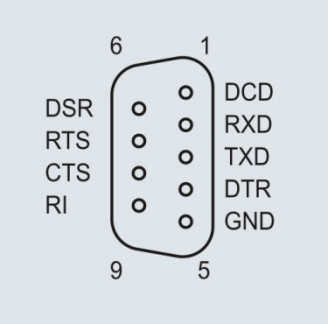
Full-duplex ASCII driver
The data is transmitted simultaneously in both directions between the communication partners. The communication partners must be equipped for simultaneous transmit/receive operation.

The RS232 serial interface can be configured using the software TB20-ToolBox.

5.1.1 Functions in the RS232 operating mode:

Communications module RS-232 serial interface	600-400-7AA31
Protocols	ASCII
Physical layer	RS 232
Terminal	Sub-D 9-pin
Transfer rates Procedure ASCII driver	110, 300, 600, 1200, 4800, 9600, 14400, 19200, 38400, 57600, 76800, 115200 baud
Data bits	7 data bits / 8 data bits
Stop bits	1 stop bit / 2 stop bits
Parity	None / Even / Odd / Custom
ASCII frame end detection	Terminator, number of characters, inter-character delay (1–65535 ms)
Frame length	1–224 characters
USB interface	Firmware update
· Protocol	USB 1.01 Device, Full Speed
· Terminal	Mini-USB

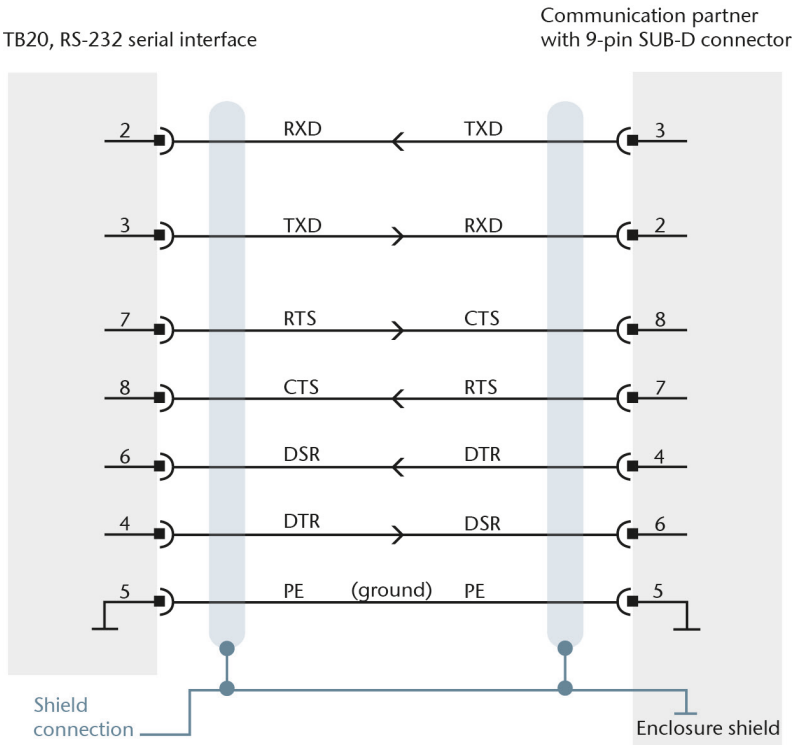
5.1.2 Pin assignment



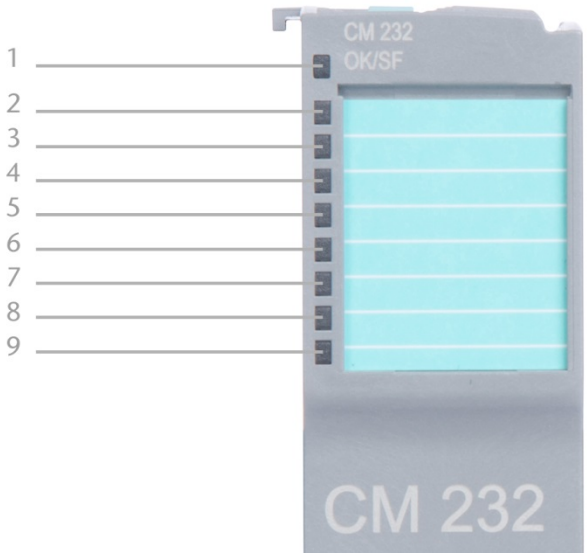
Pin	Name	Function	Direction	Description
1	DCD	Data Carrier Detect	Input	Carrier signal (modem)
2	RxD	Receive Data	Input	Receive Data Receiving line is held at logic “1” by the communication partner
3	TxD	Transmit Data	Output	Transmit Data The RS232 serial interface keeps the transmission line at logic “1” in the idle state
4	DTR	Data Terminal Ready	Output	ON = RS232 serial interface is ready for operation
5	GND	Signal Ground		Reference potential
6	DSR	Data Set Ready	Input	Communication partner ready for operation?
7	RTS	Request to send	Output	ON = RS232 serial interface ready to transmit OFF = does not transmit
8	CTS	Clear to send	Input	Communication partner ready to receive? The RS232 serial interface responds on RTS=ON.
9	RI	Ring indicator	Input	Ring indicator (modem)

5.1.3 Cable assignment of point-to-point connection

For the point-to-point connection to a RS-232 communication partner with 9-pin SUB-D terminal, a cable with the pin assignment in accordance with the following image is required.



5.1.4 LEDs of the RS232 serial interface



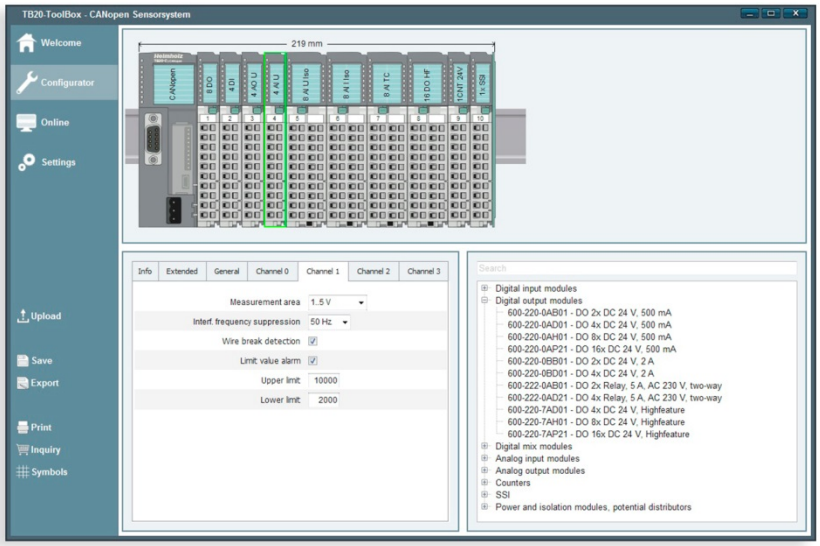
LED	Name	Display	Description
1	OK/SF LED	Solid blue light	The module is running (RUN)
		Slowly flashing blue light	The module is stopped (STOP)
		Quickly flashing blue light	The module is idle (IDLE); module's parameters have not been configured yet
		Solid red light	The module is indicating a diagnostic error
		Flashing red	The module is indicating a parameter assignment error
2	PLC	Quickly flashing green	Communication with PLC is taking place
3	TxD	Solid green light	Characters are being transmitted
4	RxD	Solid green light	Characters are being received
5	CTS	Solid green light	If CTS signal is high
6	RTS	Solid green light	If RTS signal is high
7	Reserved		
8	Reserved		
9	Reserved		

NOTE

IDLE mode (quickly flashing blue LED) indicates modules that have not been added to ongoing system operation by the coupler. One of the reasons that can cause this is an incorrect configuration (wrong module model in the slot).

6 Commissioning

6.1 TB20-ToolBox



In the TB20-ToolBox, positioning and configuration of the parameters of the components for the planning of a system is possible. Using the USB interface of the bus coupler, parameters can be configured for the RS232 serial interface.

6.2 Firmware updates

A firmware update can be requested from the support department of Helmholz if required (e-mail: support@helmholz.de).

6.3 Integrating the RS232 serial interface with the GSD file

The serial interface RS232 can be integrated into the project and parameterized with a GSD file (PROFIBUS) or GSDML file (PROFINET). The GSD and GSDML files can be downloaded in the download area under www.helmholz.de.

If the serial interface RS232 is used with a Simatic S7 SPS, the standard function blocks FB2 and FB3 can be used. Otherwise, the data exchange with the serial interface is described in chapter 8.

6.4 Using the serial interface RS232 with other PLCs

Depending upon the coupler type, the serial interface RS232 can be parameterized with the TB20 ToolBox or with coupler-specific methods. The data exchange with the serial interface RS232 is described in chapter 8.

7 Parameterization

7.1 ASCII protocol

7.1.1 Features

Using the ASCII protocol, data can be transferred to and from connected communication partners. The ASCII protocol includes only a bit transmission layer and no data link layer.

With the ASCII protocol, the complete transmission frame is passed from the TB20 master to the RS232 serial interface. The supplied data is not supplemented with control characters by the RS232 serial interface.

In the receive direction, the end criterion of a frame is defined by the parameterization of the module. The structure of the transmission frames may differ from that of the receive frames.

The ASCII protocol allows data with any structure to be sent and received.

All ASCII characters (visible and invisible) from 00 to FFh (for transmission with 8 data bits) or from 00 through 7Fh (for transmission with 7 data bits) can be transferred.

7.1.2 Transmitting data

During transmission, the number of payload data bytes to be transmitted and the payload data are transferred by the TB20 master. In the payload data, any required start and end characters may need to be included.

If the end criterion “Expiration of character delay time” needs to be configured with parameters, the RS232 serial interface keeps a pause between two frames during transmission.

Data may be transferred at any time to the RS232 serial interface. The RS232 serial interface begins with output only if a time greater than the parameterized character delay time has passed since the last frame was sent.

7.1.3 Receiving data

For receipt of data with the ASCII protocol, three different end criteria are supported. The end criterion specifies when a frame was received completely and is defined by parameterization. Supported end criteria are:

- Reception of the end character(s)
  At the end of the frame, there are one or two defined end characters.
- Reception of a fixed number of characters
  The length of the receive frames is always identical.
- Expiration of character delay time
  The frame has neither a fixed length nor defined end characters. The end of a frame is defined by a pause on the line (expiration of character delay time). The minimum values of the character delay time are dependent on the configured baud rate.

7.1.4 Minimum character delay time

The minimum delay time is dependent on the baud rate; see table:

Baud rate	Minimum character delay time
110	364 ms
300	130 ms
600	65 ms
1 200	32 ms
4 800	8 ms
9 600	4 ms
14 400	3 ms
19 200	2 ms
38 400	1 ms
57 600	1 ms
76 800	1 ms
115 200	1 ms

7.1.5 Receive buffer

The receive buffer of the RS232 serial interface is 4096 bytes. Parameterization of the RS232 defines whether the receive buffer is deleted at start-up (changing the operating state from STOP to RUN) and whether overwriting of data is to be prevented in the receive buffer. Whether the buffering of received frames is enabled or disabled is also defined.

The receive buffer of the RS232 serial interface is designed as a ring buffer:

- If several frames are stored in the receive buffer of the RS232 serial interface, the oldest frame is always transferred to the TB20 master.
- If only the most recent message frame is to be transferred to the master, the buffer overwrite protection must be canceled in the parameterization, and the parameter “Dynamic frames” must be disabled.

7.1.6 Reception error

If a reception error (parity error) is detected by the RS232 serial interface upon receipt of a frame, after expiration of the character delay time or the receipt of the end character the entire frame is discarded and a corresponding error is reported to the TB20 master.

If the end criterion “Receipt of a fixed number of characters” is set, expiration of the character delay time before reaching the parameterized number of characters generates a reception error, which is reported to the TB20 master.

7.1.7 RS232 escort lines

The RS232 serial interface supports the following escort lines:

- DCD (Input) Data carrier detect; data carrier recognized
- DTR (Output) Data terminal ready; RS232 serial interface is ready for operation
- DSR (Input) Data set ready; communication partner ready for operation
- RTS (Output) Request to send; RS232 serial interface is ready to transmit
- CTS (Input) Clear to send; communication partner can receive data from the RS232 serial interface (response to RTS = ON)

After the RS232 serial interface is switched on, the output signals are in the OFF state (inactive).

7.1.8 Control of the RS232 escort lines

The control of the control signals DTR/DSR and RTS/CTS is defined by the parameterization in the ToolBox or can be controlled by the user program of the bus master.

The following modes are supported:

- Automatic control of all escort lines
- Data flow control via RS232 escort lines (RTS/CTS)
- No control

7.1.9 Automatic control of the RS232 escort lines

NOTE

For automatic control of escort signals, only half-duplex operation is possible!

Automatic operation of the RS232 escort signals has the following functions:

- Once the RS232 serial interface is put into an operating mode with automatic use of the RS-232 escort signals through parameterization, it sets the RTS line to OFF and the DTR line to ON.
  Transmitting and receiving of frames is only possible after the DTR line is set to ON. As long as DTR remains set to OFF, no data is received on the RS232 interface. A transmit job is aborted with the corresponding error.
- When a transmit job is pending, RTS is set to ON, and with CTS = ON, the data is transmitted on the RS232 interface.
- If during transmission within the data output time the CTS line is not set to ON, or if CTS changes to OFF during the transmission process, the transmit job is aborted and an appropriate error is sent to the TB20 master.
- After the data is transmitted, the RTS line is set to OFF after expiration of the parameterized RTS removal time. The RS232 serial interface does not wait for CTS to change to OFF.

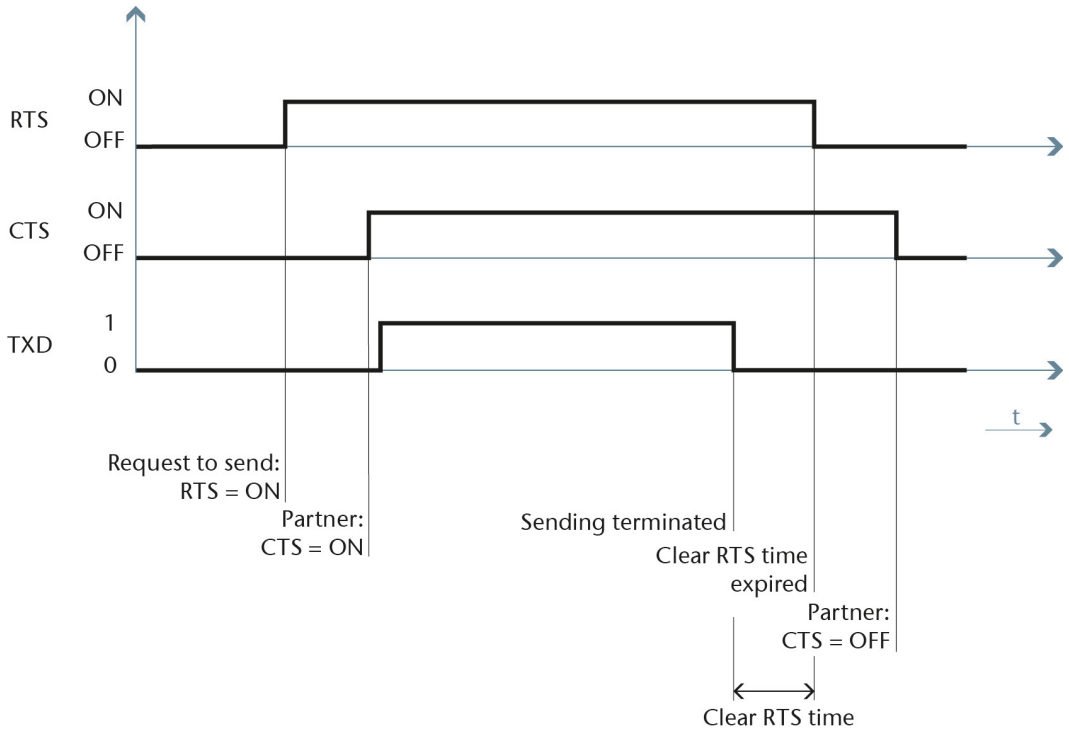
- Data can be received via the RS232 interface as soon as the DSR line is set to ON. If the receive buffer of the RS232 serial interface threatens to overflow, there is no reaction of the RS232 serial interface.
- If DSR changes from ON to OFF, both a running transmit job and the reception of data will be aborted and a corresponding error will be sent to the TB20 master.

7.1.10 Data flow control via RS232 escort lines (RTS/CTS)

The handshake procedure via RS232 escort lines controls the data flow between the RS232 serial interface and the communication partner. The handshake procedure makes it possible to prevent having data lost during transmission when the speed of data processing of the communication partner differs.

The data flow control via RS232 escort lines (RTS/CTS) is implemented in the RS232 serial interface as follows:

- If the RS232 serial interface is parameterized in the operating mode “Data flow control via RS232 escort lines,” the RS232 serial interface sets the RTS line to ON.
- If the receive buffer overflows with more than 4096 bytes or 250 frames, the RS232 serial interface sets the RTS line to OFF. If the communication partner nonetheless continues to transmit, a corresponding error is sent to the TB20 master and the received data of the last frame is discarded.
- If a frame is read by the bus master and the receive buffer is ready to receive, the RS232 serial interface sets the RTS line to ON.
- If the control signal CTS is set to OFF, the RS232 serial interface interrupts the transmission. If CTS is not set to ON after the configured time, the transmission is aborted and an error message is generated.
- If CTS is set back to ON within the waiting time for CTS ON, the remaining bytes of the previously aborted frame are sent.



7.1.11 Data flow control via XON and XOFF / software handshake

The handshake procedure controls the data flow between the RS232 serial interface and the communication partner. Sending characters makes it possible to prevent having data lost during transmission when the speed of data processing of the communication partner differs.

The data flow control via software handshake is implemented in the RS232 serial interface as follows:

- If the RS232 serial interface is configured in the operating mode with “Data flow control via XON and XOFF", the RS232 serial interface transmits the XON character.
- If the receive buffer overflows with more than 4096 bytes or 250 frames, the RS232 serial interface sends the XOFF character. If the communication partner nonetheless continues to transmit, a corresponding error is sent to the TB20 master and the received data of the last frame is discarded.
- If a frame is fetched by the bus master and the receive buffer is ready to receive, the RS232 serial interface transmits the XON character.
- If the RS232 serial interface receives the XOFF character, further transmission is aborted. If no XON is received after the configured time, the transmission is aborted and an error message is generated.
- If the XON character is sent within the waiting time, the remaining bytes of the previously aborted frame are sent.

7.2 ASCII configuration

Parameters for operating mode ASCII: 19 bytes

	7	6	5	4	3	2	1	0
Par 0	Operating mode: ASCII-8							
Par1	Reserved				Interface			
Par 2	Reserved			Dyn. frames	Data flow control			
Par 3	Default receive line	Baud rate						
Par 4	Data bits		Stop bits		Parity		Buffer at startup	Overwrite buffer
Par 5	XON character							
Par 6	XOFF characters							
Par 7-8	Waiting time on XON							
Par 9-10	Waiting time for CTS = ON							
Par 11-12	Waiting time for RTS = OFF							
Par 13	Frame end							
Par 14-15	Character delay time							
Par 16	End character 1							
Par 17	End character 2							
Par 18	Frame length							

Agreements are needed for the serial data transfer between the two communication partners.

The default settings are underlined.

Parameters	Description	Value range
Operating mode (UINT8)		1 = ASCII (8 byte)
Interface	Module 600-400-7AA31, only RS232 possible	1 = RS232
Dynamic frames	With dynamic frames, the RS232 serial interface can buffer several messages of different lengths.	1 = Yes / 0 = No
Data flow control	The transmission is synchronized through the data flow control if the communication partners are working at different speeds.	0 = None 1 = XON/XOFF 2 = RTS/CTS 3 = Automatic detection
Default reception line		For RS232: No meaning.
Baud rate	Transmission speed in bits per second	0 = 110 1 = 300 2 = 600 3 = 1200 4 = 4800

		5 = 9600 6 = 14400 7 = 19200 8 = 38400 9 = 57600 10 = 76800 11 = 115200
Data bits	Bits per character	1 = 7 data bits 2 = 8 data bits
Stop bits	Number of bits that mark the end of the information words.	1 = 1 stop bit 2 = 2 stop bits
Parity	Parity check is used to detect incorrectly transmitted information words. None: Data is sent without a parity bit. Odd: The parity bit is set; the total number of data bits with the signal state “1” is odd, including the parity bit. Even: The parity bit is set; the total number of data bits with the signal state “1” is even, including the parity bit. Any: Parity is not checked when data is received. When transmitting, it behaves like with the setting "Even".	0 = None 1 = Odd 2 = Even 3 = Any
Buffer at startup	The receive buffer is automatically cleared when the operating mode is changed from STOP to RUN.	0 = Delete the receive buffer upon startup 1 = Do not delete the receive buffer upon startup
Overwrite buffer	It is possible to prevent buffered frames from being overwritten when the RS232 serial interface receives a new frame; the receive buffer is not deleted. This prevents old received frames from being lost.	0 = Overwriting of the buffer 1 = Prevent overwriting of the buffer
XON / XOFF character:		0x00 to 0xFF with 8 data bits 0x00 to 0x7F with 7 data bits Default XON: 0x11, default XOFF: 0x13
Waiting time on XON		20-655350 in increments of 10 ms Default: 200 (10 * 200 = 2000 ms)
Waiting time for CTS = ON		20 to 655350 in increments of 10 ms Default: 200 (10 * 200 = 2000 ms)
Waiting time for RTS = OFF:		0 to 655350 in increments of 10 ms Default: 200 (10 * 200 = 2000 ms)
Frame end	Detecting the end of the reception frame Expiration of character delay time The frame end is recognized when the character delay time	0 = character delay time 1 = reception of the end character 2 = number of characters

	has expired. Reception of the end character The frame end is recognized when the end characters are received. Perception of a fixed number of characters The frame end is recognized by the frame length. All frames to be received must have the same length.	
Character delay time	Time that may elapse between the receipt of two characters	1–65535 ms Default: 4 ms
End character 1	The selected end characters limit the length of the frame Only takes effect with the end criterion “Receipt of the end character.”	0x00 to 0xFF with 8 data bits 0x00 to 0x7F with 7 data bits Default: 0x03
End character 2	The selected end characters limit the length of the frame Only takes effect with the end criterion “Receipt of the end character.”	0x00 to 0xFF with 8 data bits 0x00 to 0x7F with 7 data bits Default: 0x0
Frame length upon receipt	Frame length for data with a fixed number of characters.	1–224 characters Default: 100

8 Reference data for communication with bus masters

8.1 Data exchange between the master and the RS232 serial interface

To operate the RS232 serial interface behind a communications module, corresponding function modules must be provided in the master controller program.

The RS232 serial interface transmits data of 4, 8, or 32 bytes, input or output, with consistency over the entire length. The RS232 serial interface uses the 4-, 8-, or 32-byte input/output memory for data transmission to and from the master via a bus.

Inputs (data/feedback): n bytes. n = 8 or 32, depending on operating mode

	7	6	5	4	3	2	1	0
ID 0	Coordination byte							
ID 1	Input data byte 0							
ID 2	Input data byte 1							
ID ..	Input data byte ..							
ID ..	Input data byte ..							
ID n-1	Input data byte n-2							

Outputs (control interface): n bytes. n = 8 or 32, depending on operating mode

	7	6	5	4	3	2	1	0
ID 0	Coordination byte							
ID 1	Output data byte 0							
ID 2	Output data byte 1							
ID ..	Output data byte ..							
ID ..	Output data byte ..							
ID n-1	Output data byte n-2							

The master writes data to the inputs and outputs and reads data from the inputs and outputs:

- In the first byte of the output memory of the RS232 serial interface, the master releases a job to the RS232 serial interface.
- The RS232 serial interface transmits the job code in the input memory and thus accepts the job.
- The master exchanges data via segments 3, 7, or 31 bytes (as many segments as are required in accordance with the I/O size) until all data of the job is transferred.

The first byte of the segment is a coordination byte, which serves to synchronize the transfer of the segment between the master and the RS232 serial interface; see the figure below. The other bytes of the I/O memory contain data of the job.

Data exchange between the CPU and the RS232 serial interface:

Transmission of data from the master to the RS232 serial interface	
Byte	Content
0	Coordination byte
1	Data byte 0
2	Data byte 1
…	…
n	Data byte N

Transmission of the data from the RS232 serial interface to the master	
Byte	Content
0	Coordination byte
1	Data byte 0
2	Data byte 1
…	…
n	Data byte N

n = 3, 7, or 31, depending on the module variant of the RS232 serial interface

8.2 Coordination byte

The coordination byte (byte 0) synchronizes the data transfer between the master and the RS232 serial interface.

Table: Coordination byte

<table>
<tr><th>Byte segment</th><th colspan="9">Description</th></tr>
<tr><td rowspan="2">Job byte written by the master</td><td>Bit</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr>
<tr><td colspan="2">Reserve</td><td colspan="3">Job code</td><td>Fault</td><td colspan="3">Sequence number</td></tr>
<tr><td>Bit 7</td><td colspan="9">Reserved for special applications. For evaluations of the coordination byte, this bit needs to be hidden.</td></tr>
<tr><td>Job code</td><td colspan="9">Is set by the CPU to initiate a job.</td></tr>
<tr><td>Sequence number</td><td colspan="9">Send job: Is increased by the bus master to 1 when the bus master sends another segment to the RS232 serial interface
or
Receive job: Is taken each time from input byte 0 of the bus master when the bus master of the interface module receives a new segment in the correct order. Indicates the last valid sequence number when the error bit is set. (Value goes from 1 to 7).</td></tr>
<tr><td>Fault</td><td colspan="9">Is set by the bus master to indicate that a segment has not been received in the correct order. The sequence number field indicates the last valid sequence number.</td></tr>
<tr><td rowspan="2">Job byte written by the RS232 serial interface</td><td>Bit</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr>
<tr><td colspan="2">Reserve</td><td colspan="3">Job code</td><td>Fault</td><td colspan="3">Sequence number</td></tr>
<tr><td>Bit 7</td><td colspan="9">Reserved for special applications. For evaluations of the coordination byte, this bit needs to be hidden.</td></tr>
<tr><td>Job code</td><td colspan="9">Is taken over by the RS232 serial interface to acknowledge that the job was accepted.</td></tr>
<tr><td>Sequence number</td><td colspan="9">Send job: Is taken each time from output byte 0 of the RS232 serial interface if the module from the bus master receives a new segment in the correct order. Indicates the last valid sequence number when the error bit is set.
Receive job: Is increased by 1 by the module if the module sends another segment of the bus master (the value goes from 1 to 7).</td></tr>
<tr><td>Fault</td><td colspan="9">The error bit of the receiver is monitored by the sender for a segmented transaction. Reactions when the error bit is set:
Sender bus master (send job): The bus master sends the segments again, starting with the next segment after the number reported by the receiver.
Sender module (receive job): The RS232 serial interface interrupts transmission of the Rx frame to the user with error message 0x0551 in the status word. The RS232 serial interface waits for acknowledgment of the error message (idle). After completion of the current error sequence, the aborted Rx frame is again reported or made available for collection by the user.</td></tr>
</table>

8.3 Definitions of the job codes

The jobs are assigned to bits 4 to 6 in the coordination byte.

Table: Job codes

Bit	6	5	4	Hex. value	
	0	0	0	0_H	Idle
	0	0	1	1_H	Send
	0	1	0	2_H	Receive
	0	1	1	3_H	Read V.24 signal status
	1	0	0	4_H	Write V.24 signals
	1	0	1	5_H	Transfer parameters: Additional parameters can be set with this job.
	1	1	0	6_H	Reserved
	1	1	1	7_H	Job sent acknowledgment

Rules for writing job codes

For writing job codes in the coordination byte, the following rules apply, with which the bus master and the RS232 serial interface can synchronize the data transfers:

- Before the user program of the bus master can write a job code in the output coordination byte, it needs to see an idle code from the input coordination byte of the RS232 serial interface.
- Before the user program of the bus master can write the first segment in the output byte 1 ... n, it must see the acknowledgment code (that is, the assumed job code) in the input coordination byte of the assembly.
- If the user program sees acknowledgment codes that differ from those sent by the user program, it may not write to the output byte 0 ... n until it has again seen an idle code from the input coordination byte of the RS232 serial interface. This happens when two separate jobs are executed in the same cycle, both jobs see the idle code, and both write a different job code to the output byte. Because of the asynchronous cycle between the bus master cycle and the bus cycle, it is not ensured that the job will reach the RS232 serial interface first. Therefore, each job must be able to wait for the end of another job before it is processed.

Definitions of the status words

In the following examples of data transmissions, the module RS232 serial interface uses the bytes 1 and 2 for the status message in some responses to the bus master.

Reception status of the RS232 serial interface

When the RS232 serial interface is idle (job acknowledgment byte 0 = 00_H), it displays its reception status. The reception status is stored in bytes 1 and 2.

Table: Reception status

Status	Meaning
0000_H	No received message available
0001_H	Received message or receive frame available
$0B01_H$	The receive buffer is filled to more than 2/3rds.

Order of the bytes in the word

For data transfers between the bus master and the RS232 serial interface, the byte with the highest value is transmitted first in all 16-bit words (such as status and length).

8.3.1 Sequence for sending data from the bus master to the RS232 serial interface

The table is an example of sending a message from the bus master to the RS232 serial interface. The character string “helmholztb20seriellmodul” is sent. The latency time from one cycle until response with the sequence number arises when the CPU cycle of the bus master is almost equal to the bus cycle.

Bus master cycle	Bus master writes to RS232 serial interface	Bus master reads from RS232 serial interface

1.

User program reads the idle code

	Byte 0	1	2	3	4	5	6	7
←	00_H	$nnnn_H$		xx_H	xx_H	xx_H	xx_H	xx_H
	Ack. job	Status		No meaning				

Bus master writes the job for sending

Byte 0	1	2	3	4	5	6	7	
10_H	xx_H	xx_H	xx_H	xx_H	xx_H	xx_H	xx_H	→
Job	No meaning							

2.

User program continues to read the idle code

	Byte 0	1	2	3	4	5	6	7
←	00_H	$nnnn_H$	xx_H	xx_H	xx_H	xx_H	xx_H	xx_H
	Ack. job	Status		No meaning				

Bus master repeats the job for sending

Byte 0	1	2	3	4	5	6	7	
10_H	xx_H	xx_H	xx_H	xx_H	xx_H	xx_H	xx_H	→
Job	No meaning							

3.

User program reads the response from the RS232 serial interface

	Byte 0	1	2	3	4	5	6	7
←	10_H	$nnnn_H$	xx_H	xx_H	xx_H	xx_H	xx_H	xx_H
	Ack. job	Status		No meaning				

Bus master sends the first segment

Byte 0	1	2	3	4	5	6	7	
11_H	0016_H		‘h’	‘e’	‘l’	‘m’	‘h’	→
Job	Transmission length		Data					

4.

User program reads the response from the RS232 serial interface

	Byte 0	1	2	3	4	5	6	7
←	10_H	xx_H	xx_H	xx_H	xx_H	xx_H	xx_H	xx_H
	Ack. job	No meaning						

Bus master repeats the first segment

Byte 0	1	2	3	4	5	6	7	
11_H	‘o’	‘l’	‘z’	‘t’	‘b’	‘2’	‘0’	→
Job	Data							

Bus master cycle	Bus master writes to RS232 serial interface	Bus master reads from RS232 serial interface

5.

User program reads the response from the RS232 serial interface

	Byte 0	1	2	3	4	5	6	7
←	11_H	xx_H	xx_H	xx_H	xx_H	xx_H	xx_H	xx_H
	Ack. job	No meaning						

No error is displayed; the bus master sends the second segment

Byte 0	1	2	3	4	5	6	7	
12_H	's'	'e'	'r'	'i'	'e'	'l'	'l'	→
Job	Data							

6.

User program reads the response from the RS232 serial interface

	Byte 0	1	2	3	4	5	6	7
←	12_H	xx_H	xx_H	xx_H	xx_H	xx_H	xx_H	xx_H
	Ack. job	No meaning						

No error is displayed; the bus master sends the third segment

Byte 0	1	2	3	4	5	6	7	
13_H	'm'	'o'	'd'	'u'	'l'	xx_H	xx_H	→
Job	Data					No meaning		

7.

User program reads the response from the RS232 serial interface

	Byte 0	1	2	3	4	5	6	7
←	13_H	xx_H	xx_H	xx_H	xx_H	xx_H	xx_H	xx_H
	Ack. job	No meaning						

No error is displayed; the bus master sends the fourth segment

Byte 0	1	2	3	4	5	6	7	
14_H	'm'	'o'	'd'	'u'	'l'	xx_H	xx_H	→
Job	Data					No meaning		

8.

User program reads the response from the RS232 serial interface

	Byte 0	1	2	3	4	5	6	7
←	13_H	xx_H	xx_H	xx_H	xx_H	xx_H	xx_H	xx_H
	Ack. job	No meaning						

The bus master waits for acknowledgment after the fourth segment

Byte 0	1	2	3	4	5	6	7	
14_H	'm'	'o'	'd'	'u'	'l'	xx_H	xx_H	→
Job	Data					No meaning		

9.

User program reads the response from the RS232 serial interface

	Byte 0	1	2	3	4	5	6	7
←	14_H	xx_H	xx_H	xx_H	xx_H	xx_H	xx_H	xx_H
	Ack. job	No meaning						

The bus master does not receive a new job; the outputs remain the same.
The bus master waits for the last acknowledgment of the RS232 serial interface

Byte 0	1	2	3	4	5	6	7	
14_H	'm'	'o'	'd'	'u'	'l'	xx_H	xx_H	→
Job	Data					No meaning		

<table>
<tr><th>Bus master cycle</th><th>Bus master writes to RS232 serial interface</th><th>Bus master reads from RS232 serial interface</th></tr>
<tr><td>n.</td><td colspan="2">n bus master cycles later, the user program sees the response of the RS232 serial interface
<table>
<tr><td></td><td>Byte 0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td></tr>
<tr><td>←</td><td>74_H</td><td colspan="2">$nnnn_H$</td><td>xx_H</td><td>xx_H</td><td>xx_H</td><td>xx_H</td><td>xx_H</td></tr>
<tr><td></td><td>Ack. job</td><td colspan="7">No meaning</td></tr>
</table></td></tr>
<tr><td>-</td><td colspan="2">The CPU of the bus master writes the idle code to the job and terminates the job.
<table>
<tr><td>Byte 0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td></tr>
<tr><td>00_H</td><td>xx_H</td><td>xx_H</td><td>xx_H</td><td>xx_H</td><td>xx_H</td><td>xx_H</td><td>xx_H</td></tr>
<tr><td>Job</td><td colspan="7">No meaning</td></tr>
</table></td></tr>
</table>

8.3.2 Sequence for receiving data in the bus master from the RS232 serial interface

The table is an example of receiving a message in the bus master from the RS232 serial interface. The character string “helmholz” is received. The I/O memory is 8 bytes. The bus cycle is less than the CPU cycle of the bus master, so that no latency time is created in the module.

<table>
<tr><th>Bus master cycle</th><th colspan="8">Bus master writes to RS232 serial interface</th><th colspan="8">Bus master reads from RS232 serial interface</th></tr>
<tr><td>n</td><td colspan="16">For n-1 cycles, the CPU of the bus master reads the idle code of the RS232 serial interface until in cycle n the status is displayed that a received message is present.
←
Byte 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7
00_H | $nnnn_H$ | xx_H | xx_H | xx_H | xx_H | xx_H | xx_H
Ack. job | Status | No meaning
Status 0000_H = No received message available
0001_H = Received message available
$0B01_H$ = Receive buffer is more than 2/3rds full
The CPU of the bus master writes the job for receipt
Byte 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7
20_H | xx_H | xx_H | xx_H | xx_H | xx_H | xx_H | xx_H
Job | No meaning
→</td></tr>
<tr><td>n+1</td><td colspan="16">The user program reads the response of the RS232 serial interface, the RS232 serial interface acknowledges receipt, responds with the first segment, and increments the sequence number.
←
Byte 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7
21_H | 0006_H | ‘h’ | ‘e’ | ‘l’ | ‘m’ | ‘h’
Ack. job | Length | Data
The bus master reads the job for acknowledgment of the 1st segment
Byte 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7
21_H | xx_H
Job | No meaning
→</td></tr>
<tr><td>n+2</td><td colspan="16">The bus master reads the 2nd segment from the RS232 serial interface.
←
Byte 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7
22_H | ‘o’ | ‘l’$_H$ | ‘z’ | xx_H | xx_H | xx_H | xx_H
Ack. job | Data | No meaning
The bus master reads the job for acknowledgment of the 2nd segment
Byte 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7
22_H | xx_H | xx_H | xx_H | xx_H | xx_H | xx_H | xx_H
Job | No meaning
→</td></tr>
<tr><td>n+3</td><td colspan="16">After the first receive transaction is terminated, the RS232 serial interface returns to the idle state.
←
Byte 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7
00_H | xx_H | xx_H | xx_H | xx_H | xx_H | xx_H | xx_H
Ack. job | Status US | No meaning</td></tr>
<tr><td>-</td><td colspan="16">The bus master ends the job.</td></tr>
</table>

8.3.3 Sequence for reading the V.24 signal status

The table is an example of how the CPU reads the status of the V.24 signals from the RS232 serial interface. The I/O memory is 8 bytes.

<table>
<tr><th>Bus master cycle</th><th colspan="8">Bus master writes to RS232 serial interface</th><th colspan="8">Bus master reads from RS232 serial interface</th></tr>
<tr><td>1.</td><td colspan="16">The user program reads the idle code from the RS232 serial interface
← Byte 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7
← 00_H | $nnnn_H$ | xx_H | xx_H | xx_H | xx_H | xx_H
← Ack. job | Status | No meaning
The bus master writes the job for writing the V.24 signals
Byte 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7
30_H | xx_H | xx_H | xx_H | xx_H | xx_H | xx_H | xx_H →
Job | No meaning</td></tr>
<tr><td>2.</td><td colspan="16">The user program reads the response from the RS232 serial interface
← Byte 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7
← 31_H | $nnnn_H$ | xx_H | xx_H | xx_H | xx_H | xx_H
← Ack. job | Signals*) | No meaning
The bus master writes the acknowledgment and accepts the sequence number.
Byte 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7
31_H | xx_H | xx_H | xx_H | xx_H | xx_H | xx_H | xx_H →
Job | No meaning</td></tr>
<tr><td>3.</td><td colspan="16">After the first receive transaction is terminated, the RS232 serial interface returns to the idle state.
← Byte 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7
← 00_H | $nnnn_H$ | xx_H | xx_H | xx_H | xx_H | xx_H | xx_H
← Ack. job | Status | No meaning</td></tr>
<tr><td>-</td><td colspan="16">The bus master ends the job.</td></tr>
</table>

* Signal states

MSB	LSB							
00	0	0	0	DCD	CTS	RTS	DSR	DTR
	7	6	5	4	3	2	1	0

8.3.4 Sequence for writing V.24 signals

The table is an example of how the CPU writes V.24 signals to the RS232 serial interface. The I/O memory is 8 bytes.

<table>
<tr><th>Bus master cycle</th><th colspan="8">Bus master writes to RS232 serial interface</th><th>Bus master reads from RS232 serial interface</th></tr>
<tr><td>1.</td><td colspan="9">The user program reads the idle code from the RS232 serial interface
<table>
<tr><td></td><td>Byte 0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td></tr>
<tr><td>←</td><td>00_H</td><td colspan="2">$nnnn_H$</td><td>xx_H</td><td>xx_H</td><td>xx_H</td><td>xx_H</td><td>xx_H</td></tr>
<tr><td></td><td>Ack. job</td><td colspan="2">Status</td><td colspan="5">No meaning</td></tr>
</table>
The bus master writes the job for reading the V.24 signal status
<table>
<tr><td>Byte 0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td></td></tr>
<tr><td>10_H</td><td>xx_H</td><td>xx_H</td><td>xx_H</td><td>xx_H</td><td>xx_H</td><td>xx_H</td><td>xx_H</td><td>→</td></tr>
<tr><td>Job</td><td colspan="2">Signal states*)</td><td colspan="6">No meaning</td></tr>
</table></td></tr>
<tr><td>2.</td><td colspan="9">The user program reads the response from the RS232 serial interface
<table>
<tr><td></td><td>Byte 0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td></tr>
<tr><td>←</td><td>40_H</td><td>$nnnn_H$</td><td>xx_H</td><td>xx_H</td><td>xx_H</td><td>xx_H</td><td>xx_H</td><td>xx_H</td></tr>
<tr><td></td><td>Ack. job</td><td colspan="2">Status</td><td colspan="5">No meaning</td></tr>
</table>
The bus master writes the idle state to the job byte
<table>
<tr><td>Byte 0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td></td></tr>
<tr><td>00_H</td><td>xx_H</td><td>xx_H</td><td>xx_H</td><td>xx_H</td><td>xx_H</td><td>xx_H</td><td>xx_H</td><td>→</td></tr>
<tr><td>Job</td><td colspan="7">No meaning</td></tr>
</table></td></tr>
<tr><td>3.</td><td colspan="9">After the transaction is terminated, the RS232 serial interface returns to the idle state.
<table>
<tr><td></td><td>Byte 0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td></tr>
<tr><td>←</td><td>00_H</td><td>$nnnn_H$</td><td>xx_H</td><td>xx_H</td><td>xx_H</td><td>xx_H</td><td>xx_H</td><td>xx_H</td></tr>
<tr><td></td><td>Ack. job</td><td colspan="2">Status</td><td colspan="5">No meaning</td></tr>
</table></td></tr>
<tr><td>-</td><td colspan="9">The bus master ends the job.</td></tr>
</table>

* Signal states

MSB	LSB							
00	0	0	0	DCD	CTS	RTS	DSR	DTR
	7	6	5	4	3	2	1	0

8.4 Parameters for data flow control

The job code of the parameter transfer with the ASCII driver allows additional parameters to be set. The parameters depend on the data flow control used.

Parameter frame for data flow control with XON/XOFF

Byte	Description	Value range	Default value
1	Parameter block number	20_H	
2 and 3	Length	0004_H	0004_H
4	XON character	0 to 127 (7 data bits) 0 to 255 (8 data bits)	11 (DC1)
5	XOFF character	0 to 127 (7 data bits) 0 to 255 (8 data bits)	13 (DC3)
6 and 7	Waiting time for XON after XOFF	20 to 655350 in increments of 10 ms	200 (2000 ms)

Parameter frame for data flow control with RTS/CTS

Byte	Description	Value range	Default value
1	Parameter block number	21_H	
2 and 3	Length	0002_H	0002_H
4 and 5	Time for RTS = OFF after the transmission	20 to 655350 in increments of 10 ms	1 (10 ms)
6 and 7	Waiting time for CTS = ON after RTS = ON	20 to 655350 in increments of 10 ms	1 (10 ms)

Parameter frame for automatic use of RS232C escort signals

Byte	Description	Value range	Default value
1	Parameter block number	22_H	
2 and 3	Length	0004_H	0004_H
4 and 5	Time for RTS = OFF after the transmission	0 to 655350 in increments of 10 ms	1 (10 ms)
6 and 7	Waiting time for CTS = ON after RTS = ON	0 to 655350 in increments of 10 ms	1 (10 ms)

8.4.1 Example sequence for XON/XOFF

<table>
<tr><th>Bus master cycle</th><th>Bus master writes to RS232 serial interface</th><th>Bus master reads from RS232 serial interface</th></tr>
<tr><td>1.</td><td>User program reads the idle code
←
<table><tr><td>Byte 0</td><td>1</td><td>2</td><td>3</td></tr><tr><td>00_H</td><td colspan="2">$nnnn_H$</td><td>xx_H</td></tr><tr><td>Ack. job</td><td colspan="2">Status</td><td>No meaning</td></tr></table></td><td></td></tr>
<tr><td></td><td>Job:
Send parameter code (1 0 1 or 5H) plus sequence number 0
<table><tr><td>Byte 0</td><td>1</td><td>2</td><td>3</td></tr><tr><td>50_H</td><td>xx_H</td><td>xx_H</td><td>xx_H</td></tr><tr><td>Job</td><td colspan="3">No meaning</td></tr></table>→</td><td></td></tr>
<tr><td>2.</td><td>User program reads the response from the RS232 serial interface
←
<table><tr><td>Byte 0</td><td>1</td><td>2</td><td>3</td></tr><tr><td>50_H</td><td>xx_H</td><td>xx_H</td><td>xx_H</td></tr><tr><td>Ack. job</td><td colspan="3">No meaning</td></tr></table></td><td></td></tr>
<tr><td></td><td>The job has been accepted, the CPU of the bus master sends the first segment
<table><tr><td>Byte 0</td><td>1</td><td>2</td><td>3</td></tr><tr><td>51_H</td><td>20_H</td><td colspan="2">0004_H</td></tr><tr><td>Job</td><td>Data flow</td><td colspan="2">Transmission length</td></tr></table>Job: Continue parameter and increment sequence number
Data flow: Code for data flow parameter
→</td><td></td></tr>
<tr><td>3.</td><td>User program reads the response from the RS232 serial interface
←
<table><tr><td>Byte 0</td><td>1</td><td>2</td><td>3</td></tr><tr><td>51_H</td><td>xx_H</td><td>xx_H</td><td>xx_H</td></tr><tr><td>Ack. job</td><td colspan="3">No meaning</td></tr></table></td><td></td></tr>
<tr><td></td><td>No error is displayed; the CPU of the bus master sends the second segment
<table><tr><td>Byte 0</td><td>1</td><td>2</td><td>3</td></tr><tr><td>52_H</td><td>$0B_H$</td><td>$0D_H$</td><td>00_H</td></tr><tr><td>Job</td><td>DC1</td><td>DC3</td><td>Waiting time for XON after XOFF - MSB</td></tr></table>→</td><td></td></tr>
<tr><td>2.</td><td>User program reads the response from the RS232 serial interface
←
<table><tr><td>Byte 0</td><td>1</td><td>2</td><td>3</td></tr><tr><td>52_H</td><td>xx_H</td><td>xx_H</td><td>xx_H</td></tr><tr><td>Ack. job</td><td colspan="3">No meaning</td></tr></table></td><td></td></tr>
<tr><td></td><td>No error is displayed; the CPU of the bus master sends the third segment
<table><tr><td>Byte 0</td><td>1</td><td>2</td><td>3</td></tr><tr><td>53_H</td><td>$C8_H$</td><td>xx_H</td><td>xx_H</td></tr><tr><td>Job</td><td>Waiting time for XON after XOFF - LSB</td><td colspan="2">No meaning</td></tr></table>→</td><td></td></tr>
</table>

<table>
<tr><th>Bus master cycle</th><th colspan="4">Bus master writes to RS232 serial interface</th><th colspan="4">Bus master reads from RS232 serial interface</th></tr>
<tr><td>5.</td><td colspan="8">User program reads the response from the RS232 serial interface</td></tr>
<tr><td></td><td colspan="4"></td><td>Byte 0</td><td>1</td><td>2</td><td>3</td></tr>
<tr><td></td><td colspan="4"></td><td>53_H</td><td>xx_H</td><td>xx_H</td><td>xx_H</td></tr>
<tr><td></td><td colspan="4"></td><td>Ack. job</td><td colspan="3">No meaning</td></tr>
<tr><td></td><td colspan="8">Bus master repeats the third segment and waits for job end acknowledgment</td></tr>
<tr><td></td><td>Byte 0</td><td>1</td><td>2</td><td>3</td><td colspan="4"></td></tr>
<tr><td></td><td>53_H</td><td>$C8_H$</td><td>xx_H</td><td>xx_H</td><td colspan="4"></td></tr>
<tr><td></td><td>Job</td><td>Waiting time for XON after XOFF - LSB</td><td colspan="2">No meaning</td><td colspan="4"></td></tr>
<tr><td>6.</td><td colspan="8">User program reads the response from the RS232 serial interface</td></tr>
<tr><td></td><td colspan="4"></td><td>Byte 0</td><td>1</td><td>2</td><td>3</td></tr>
<tr><td></td><td colspan="4"></td><td>73_H</td><td colspan="2">xx_H</td><td>xx_H</td></tr>
<tr><td></td><td colspan="4"></td><td>Ack. job</td><td colspan="2">Status</td><td>No meaning</td></tr>
<tr><td></td><td colspan="8">The CPU of the bus master writes the idle code to the job and terminates the job.</td></tr>
<tr><td></td><td>Byte 0</td><td>1</td><td>2</td><td>3</td><td colspan="4"></td></tr>
<tr><td></td><td>00_H</td><td>xx_H</td><td>xx_H</td><td>xx_H</td><td colspan="4"></td></tr>
<tr><td></td><td>Job</td><td colspan="3">No meaning</td><td colspan="4"></td></tr>
</table>

9 Error handling

The serial interface module outputs an error as a response to the following conditions:

- If the send job is longer than 224 bytes, the RS232 serial interface responds with a job end acknowledgment and the status word contains the error code. The bus master writes an idle code to the job and terminates the job.
- If a receive job is sent to the RS232 serial interface and the received message contains an error, the RS232 serial interface takes over the receive job code with the sequence number zero and the status word contains the error code. The bus master writes an idle code to the job and terminates the job.
- If a receive job is sent to the RS232 serial interface and there is no received message available, the RS232 serial interface takes over the receive job code with the sequence number zero and the status word contains the value 0101_{H}. This is not an error condition, but it prevents the RS232 serial interface from being disabled in receive job mode and waits for a receive message so that transmission jobs can be executed. The bus master writes an idle code to the job and terminates the job.

Table: Error codes in the status word

Status word (hexadecimal)	Error description
0708_{H}	CTS = ON or XON – waiting period expired.
0806_{H}	The character delay time was exceeded.
$080A_{H}$	The receive buffer is full
$080C_{H}$	An error occurred during transmission
0810_{H}	A parity error occurred
0811_{H}	A character frame error occurred
0812_{H}	Additional characters were received after CTS was set to Off or XOFF was sent.
0818_{H}	CTS or DSR were switch to Off within a transmission process.
0850_{H}	The receive frame is greater than the parameterized length or greater than the maximum allowed length.
$0B01_{H}$	The receive buffer has reached a level greater than 2/3rds.

10 General technical specifications

Order no.	600-400-7AA31
Module type	RS-232
Electrically isolated from backplane bus	Yes
Cable length	15 m
Power dissipation	Max. 0.7 W
Power supply for modules	5 VDC, max. 130 A
Certifications	CE, UL pending
Noise immunity	DIN EN 61000-6-2 “EMC Immunity”
Interference emission	DIN EN 61000-6-2 “EMC Emission”
Vibration and shock resistance	DIN EN 60068-2-8:2008 “Vibration” DIN EN 60068-27:2010 “Shock”
Isolation voltage	500 V
Protection rating	IP 20
Relative humidity	95% r H without condensation
Installation position	Any
Permissible ambient temperature	0 °C to 60 °C
Transport and storage temperature	-20 °C to 80 °C
Dimensions (H x W x D)	110 mm x 25 mm x 73 mm
Weight	Approx. 120 g
Hot-pluggable	Yes

11 Spare parts

11.1 Base modules

11.1.1 14 mm width standard base module

The 14 mm standard base module is available in sets of five with order no. 600-900-9AA01.

11.1.2 25 mm width base module

The 25 mm standard base module is available in sets of five with order no. 600-900-9AA21.

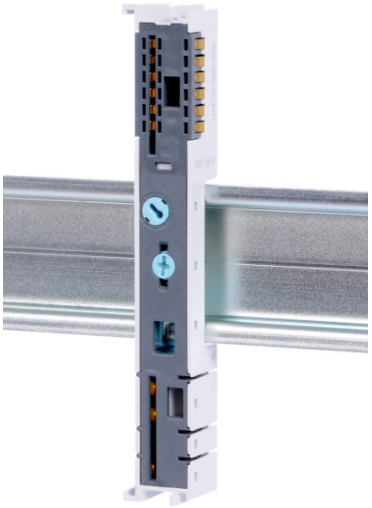
11.1.3 Power and isolation base module

The power and isolation base module is available in sets of five with order no. 600-900-9BA01.

11.1.4 Power base module

The power base module is available in sets of five with order no. 600-900-9CA01.

It can be used with the power module (600-700-0AA01) and with all bus couplers.



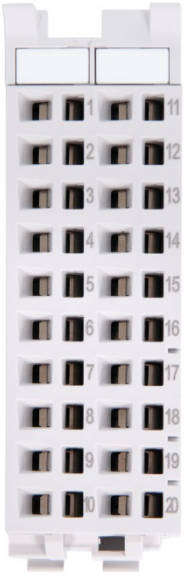
11.2 Front connector

11.2.1 10-terminal front connector

The 10-terminal front connector is available in sets of five with order No. 600-910-9AJ01.

11.2.2 20-terminal front connector

The 20-terminal front connector is available in sets of five with order No. 600-910-9AT21.



11.3 Electronic modules

To order spare electronic modules, simply use the order no. for the original product. Electronic modules are always sent as a complete assembly, including the corresponding base module and front connector.

11.4 Final bus cover

The final bus cover is available in sets of five with order no. 600-920-9AA01.