# IB IL RS UNI(-2MBD)-PAC
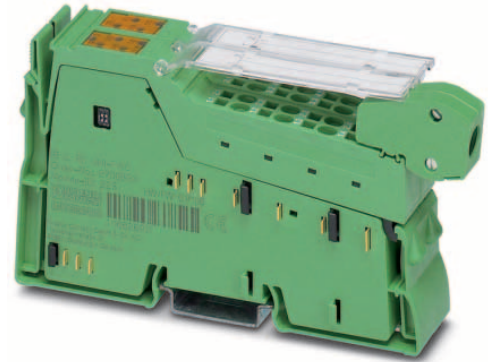
**Inline communication terminal
for serial data transmission**

Data sheet
8235_en_01

## 1    Description

The terminal is designed for use within an Inline station.

It is used to operate standard I/O devices with serial interfaces on a bus system.

**Features**

– A serial input and output channel in RS-232, RS-422 and RS-485 format
– Various protocols supported
– Transmission speed can be set up to 250000 baud
– Number of data bits, stop bits and parity can be set
– Parameterization via process data or PCP
– Process data width can be set via DIP switches: 7, 15 or 31 words
– Diagnostic and status indicators

| | |
|---|---|
| **i** | This data sheet is only valid in association with the IL SYS INST UM E user manual. |
| **i** | Make sure you always use the latest documentation.<br>It can be downloaded from the product at www.phoenixcontact.net/catalog. |

## 2    Table of contents

## 3 Ordering data

| Description | Type | Order No. | Pcs. / Pkt. |
|---|---|---|---|
| Inline communication terminal, for serial data transmission, complete with accessories (plug connector and labeling field), 1 serial input and output channel in RS-485/422 or RS-232 design | IB IL RS UNI-PAC | 2700893 | 1 |
| Inline communication terminal, for serial data transmission, complete with accessories (plug connector and labeling field), 1 serial input and output channel in RS-485/422 or RS-232 design, transmission speed 2 Mbaud | IB IL RS UNI-2MBD-PAC | 2701025 | 1 |

| Accessories | Type | Order No. | Pcs. / Pkt. |
|---|---|---|---|
| Connector set ( Plug/Adapter ) | IB IL AO/CNT-PLSET | 2732664 | 1 |
| Inline shield connector ( Plug/Adapter ) | IB IL SCN-6 SHIELD | 2726353 | 5 |
| Labeling field, width: 12.2 mm ( Marking ) | IB IL FIELD 2 | 2727501 | 10 |
| Insert strip, Sheet, white, Unlabeled, Can be labeled with: Office-Drucksysteme,  Plotter: Laser printer, Mounting type: Insert, Lettering field: 62 x 10 mm ( Marking ) | ESL 62X10 | 0809492 | 1 |

| Documentation | Type | Order No. | Pcs. / Pkt. |
|---|---|---|---|
| User manual, English, Automation terminals of the Inline product range | IL SYS INST UM E | - | - |
| Data sheet, English, INTERBUS addressing | DB GB IBS SYS ADDRESS | - | - |

## 4 Technical data

**Dimensions (nominal sizes in mm)**
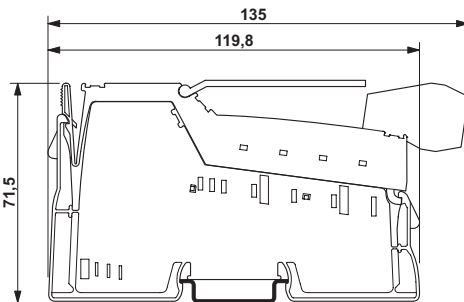


| Width | 24.4 mm |
|---|---|
| Height | 136.8 mm |
| Depth | 71.5 mm |

**General data**

| Color | green |
|---|---|
| Weight | 135 g |
| Mounting type | DIN rail |
| Ambient temperature (operation) | -25 °C ... 55 °C |
| Ambient temperature (storage/transport) | -25 °C ... 85 °C |
| Permissible humidity (operation) | 10 % ... 95 % (DIN EN 61131-2) |
| Permissible humidity (storage/transport) | 10 % ... 95 % (DIN EN 61131-2) |

## General data

| | |
|---|---|
| Air pressure (operation) | 70 kPa ... 106 kPa (up to 3000 m above sea level) |
| Air pressure (storage/transport) | 70 kPa ... 106 kPa (up to 3000 m above sea level) |
| Degree of protection | IP20 |
| Protection class | III, IEC 61140, EN 61140, VDE 0140-1 |

## Connection data

| | |
|---|---|
| Name | Inline connectors |
| Connection method | Spring-cage connection |
| Conductor cross section solid / stranded | 0.08 mm² ... 1.5 mm² |
| Conductor cross section [AWG] | 28 ... 16 |

## Interface Inline local bus

| | |
|---|---|
| Connection method | Inline data jumper |

## Interface RS-232, RS-485, RS-422

| | |
|---|---|
| Connection method | Spring-cage connection |
| Transmission physics | Copper |

## Transmission speed Inline local bus

| | |
|---|---|
| IB IL RS UNI-PAC | 500 kBit/s |
| IB IL RS UNI-2MBD-PAC | 2 MBit/s |

## Power consumption

| | |
|---|---|
| Main circuit supply $U_M$ | 24 V DC |
| I/O voltage range | 19.2 V DC ... 30 V DC (including all tolerances, including ripple) |
| Current consumption from $U_M$ | Typ. 15 mA |
| Current consumption from $U_M$ | max. 25 mA |
| Communications power $U_L$ | 7.5 V (via voltage jumper) |
| Current consumption from $U_L$ | Typ. 78 mA |
| Current consumption from $U_L$ | max. 90 mA |
| Power loss | max. 1.2 W (Within the permissible operating temperature) |

> **i** This terminal takes no current from the $U_{ANA}$ und $U_S$ potential jumpers.

## Electrical isolation/isolation of the voltage areas

| | |
|---|---|
| Serial interface/7.5 V supply (bus logic) | 500 V AC , 50 Hz , 1 min |
| Serial interface/24 V supply $U_M$ | 500 V AC , 50 Hz , 1 min |
| Serial interface/functional earth ground | 500 V AC , 50 Hz , 1 min |
| 7.5 V supply (bus logics) / functional earth ground | 500 V AC , 50 Hz , 1 min |
| 24 V supply ($U_M$)/functional earth ground | 500 V AC , 50 Hz , 1 min |

> **i** Electrical isolation of the logic level from the serial interface is ensured by the DC/DC converter.
> The serial interface control and data lines galvanically have the same potential. FE is a separate potential area.

## Error messages to the higher level control or computer system

None

## Approvals

For the latest approvals, please visit www.phoenixcontact.net/catalog.

---

# 5 Internal circuit diagram
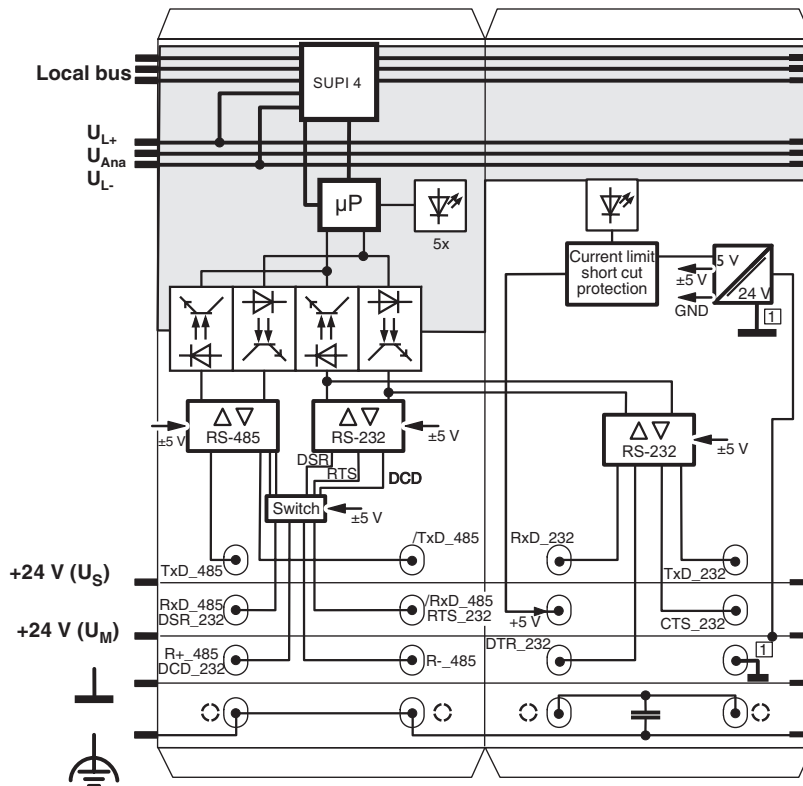


Figure 1    Internal wiring of the terminal points

Key:

| Symbol | Description |
|---|---|
| SUPI 4 | Protocol chip |
| µC | Microprocessor |
| ▽△ | RS-485/422/232 interface |
| ⧄ | Diagnostic and status indicators |
| | Optocoupler |
| | DC/DC converter with electrical isolation |
| | Electrically isolated area |
| ⊥ | Capacitor |
| 1⊥ | Ground, electrically isolated from ground of the communications power $U_L$ |

**i** Explanation for other used symbols has been provided in the IL SYS INST UM E user manual.

**i** There is a combination of resistors between R+_485 and R-_485 which serve as a termination resistor when being used as RS-485 or RS-422 interface.

# 6 Terminal point assignment
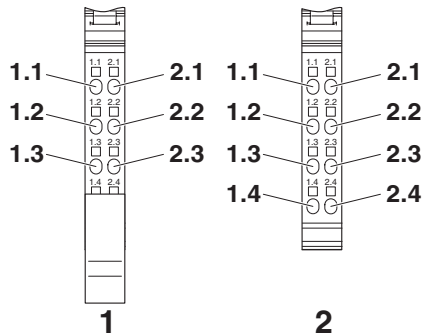


Figure 2    Terminal point assignment

## 6.1 RS-485/422 communication

| Terminal point | Signal | Description | |
|---|---|---|---|
| **Connector 1** | | | |
| 1.1 | TxD+ | Transmit data | positive |
| 2.1 | TxD- | Transmit data | negative |
| 1.2 | RxD+ | Receive data | positive |
| 2.2 | RxD- | Receive data | negative |
| 1.3 | R+ | Termination resistor | Positive pole |
| 2.3 | R- | Termination resistor | Negative pole |
| 1.4 / 2.4 | Shield | Shield connection, same potential as FE | |
| **Connector 2** | | | |
| The terminal points of this connector are not used. | | | |

## 6.2 RS-232 communication

| Terminal point | Signal | | Description |
|---|---|---|---|
| **Connector 1** | | | |
| 1.1 / 2.1 | - | | Not used |
| 1.2 | DSR | Data set ready | Connected device ready to operate; handshake signal; input |
| 2.2 | RTS | Request to send | Request to send; handshake signal; output |
| 1.3 | DCD | Data carrier detect | Connected device ready to operate; handshake signal; input |
| 2.3 | - | | Not used |
| 1.4 / 2.4 | - | | Not used |
| **Connector 2** | | | |
| 1.1 | RxD | | Serial data input |
| 2.1 | TxD | | Serial data output |
| 1.2 | +5 V DC | | Control output; internally wired to +5 V |
| 2.2 | CTS | Clear to send | Permission to send; connected device ready to receive; handshake signal; input |
| 1.3 | DTR | Data terminal ready | Startup request to the connected device; handshake signal; output |
| 2.3 | GND | | Ground for the serial interface |
| 1.4 / 2.4 | Shield | | Shield connection, capacitive to FE |

**i** Ensure that on connector 2, terminal point 1.2 (+5 V DC) is exclusively used to provide the 5 V signal for the CTS input (terminal point 2.2), in the event of communication without a handshake. In this case, insert a jumper between the terminal points.

## 7 Local status and diagnostic indicators
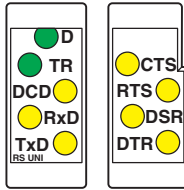


Figure 3     Local status and diagnostic indicators

| Designation | Color | Meaning |
|---|---|---|
| D | green | Diagnostics (bus and logic voltage) |
| TR | Green | PCP communication |
| DCD | Yellow | Data carrier detect |
| RxD | Yellow | Terminal receives data from the connected device |
| TxD | Yellow | Terminal transmits data to the connected device |
| CTS | Yellow | Clear to send |
| RTS | Yellow | Request to send |
| DSR | Yellow | Data set ready |
| DTR | Yellow | Data terminal ready |

> **i** For detailed information on diagnostics, please refer to the IL SYS INST UM E user manual.

**Function identification**

Orange

2 Mbps: White stripe in the vicinity of the D LED

# 8    Serial interfaces

## 8.1    RS-232

The RS-232 interface on the terminal represents some form of DTE (data termination equipment).

This means that connector 2 terminal point 2.1 (TxD) is always used to transmit and connector 2 terminal point 1.1 (RxD) is always used to receive.

According to the standard, some form of DCE (data communication equipment) should be connected to the RS-232 interface as a peer. DTE can also be connected.

Measuring the voltage between the connection points for the TxD and GND signals in idle state will determine whether the device to be connected to the RS-232 interface is a form of DTE or DCE. If the voltage measures approximately -5 V, the device is a form of DTE. If the voltage is approximately 0 V, the device is a form of DCE.

Example: When using a 25-pos. standard connector the voltage between pin 2 (TxD) and pin 7 (GND) must be measured.

### RS-232 module handshake signals

Any device with a RS-232 interface can be connected to the RS-232 interface on the terminal. Both the terminal and the device connected to the RS-232 interface can act as a transmitter and a receiver for data exchange. As errors can occur during data exchange if both devices transmit or receive simultaneously, the handshake is used as a procedure for the mutual signaling of clear to receive and clear to transmit.

The terminal supports the RTS, CTS, CTR, DSR and DCD handshake signals. Each uses one wire of the connecting cable.

The connecting signals are described from the point of view of the terminal, i.e., from the point of view of the DTE.

## 8.2    RS-485

In RS-485 operating mode, an RS-485 network with several devices can be created.

Use a twisted-pair, shared and shielded data line to connect the devices. Fit this data line with termination resistors at the two most remote points of the RS-485 network. Use the termination resistor integrated in the device when connecting the Inline terminal.

This operating mode supports half-duplex transmission. Make sure that only one device is transmitting data at a time.

To ensure a defined idle cable state, the terminal contains data line polarization.

## 8.3    RS-422

In RS-422 operating mode, a point-to-point connection can be established.

Use a twisted-pair, shared and shielded data line to connect the devices. Fit this data line with termination resistors at every device. Use the termination resistor integrated in the device when connecting the Inline terminal.

This operating mode supports full duplex transmission.

# 9 Connection examples

| Operating mode | Special feature | Shield connection | Notes | Required connectors |
|---|---|---|---|---|
| RS-485 | Inline terminal as network end point | Directly | Termination resistor required | Supplied connectors (connector set) |
| RS-485 | Inline terminal as network end point | With a capacitor | Termination resistor required | Supplied connectors (connector set) |
| RS-485 | Terminal in the network center | With a capacitor/directly | - | 2 shield connectors |
| RS-422 | Shield connected using a capacitor | With a capacitor | Termination resistor required | Supplied connectors (connector set) |

| **i** | The connector set consists of a shield connector and a standard connector. It is supplied as standard. |
|---|---|

**Please observe the following when wiring:**

– Always connect the shield between two devices on one side using a capacitor and on the other side directly to FE.

– Fit a termination resistor to the RS-485 network beginning and end.

– For an RS-422 point-to-point connection, the receive signals on every device must be fitted with termination resistors.

## 9.1 RS-485: Terminal as the network end point

| **i** | Connect the shield directly to FE. |
|---|---|
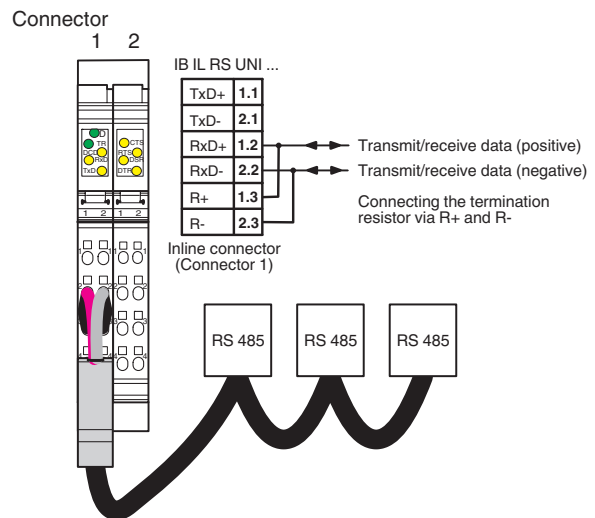


Figure 4    RS-485 interface wiring:
Terminal as end point of a network, shield or data cable directly connected to FE

### 9.2 RS-485: Terminal as the network end point

### 9.3 RS-485: Terminal in the network center

| i | Connect the shield to FE using a capacitor. |
|---|---|



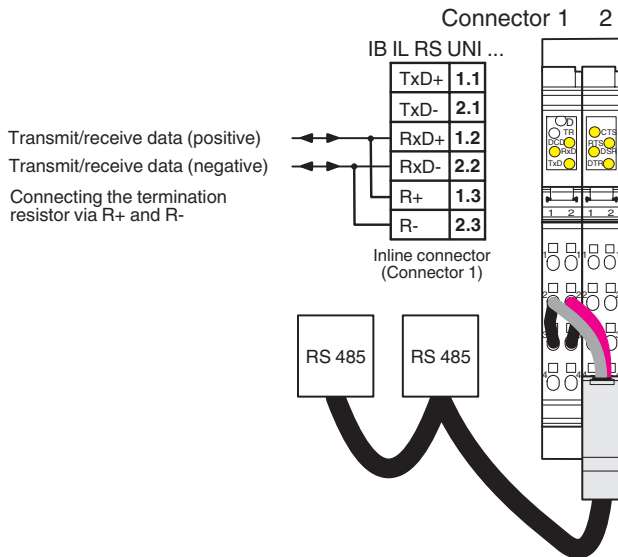Figure 5    RS-485 interface wiring:
Terminal as end point of a network, shield or
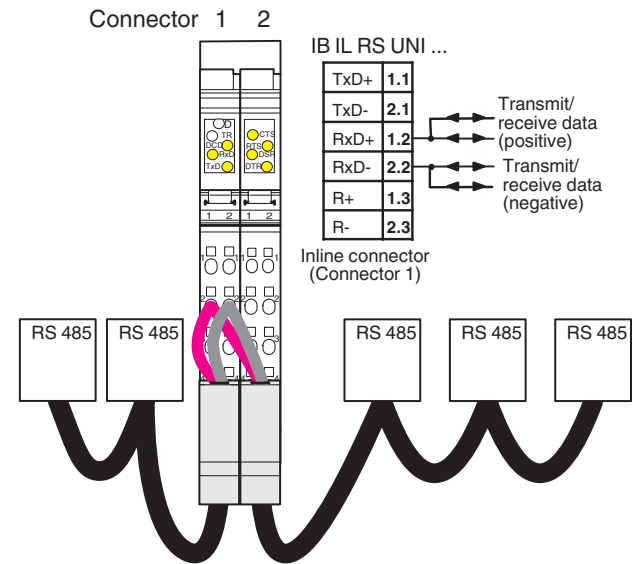data cable connected capacitively to FE



Figure 6    RS-485 interface wiring:
Terminal in the center of a network, data line
shield connected directly capacitively to FE

### 9.4 RS-422: Shield connected to the terminal using a capacitor
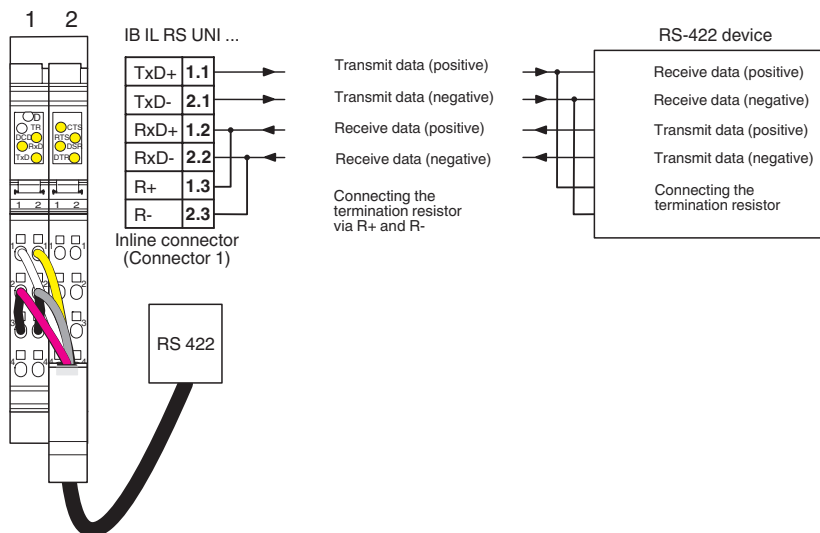


Figure 7    RS-422 interface wiring: Data line shield connected capacitively to FE

### 9.5 RS-232: With 4-wire handshake
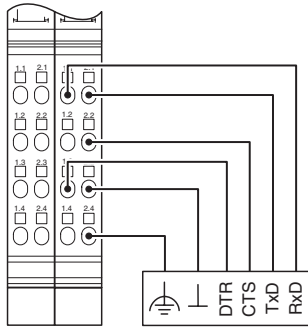


Figure 8    RS-232 interface wiring: data cable shield connected capacitively to FE

### 9.6 RS-232: Without handshake
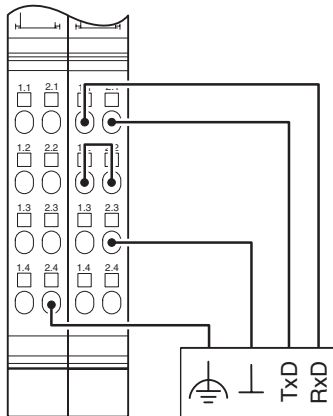


Figure 9    RS-232 interface wiring: data cable shield connected directly to FE

For wiring without handshake, insert a bridge between terminal points 1.2 (+5 V) and 2.2 (CTS).

# 10    Data storage and transmission

The terminal stores the received serial data in an intermediate buffer until it is requested from the serial interface by the bus controller board or the device. Serial data traffic can be managed using various protocols. The protocol used depends on the type of protocol supported by the peer.

### 10.1 Supported protocols

| Protocol | Receive memory | Transmit memory | Special features when receiving |
|---|---|---|---|
| Transparent | 4096 bytes | 1023 bytes | |
| End-to-end | 3 buffers with 330 bytes each | 1023 bytes (including end characters) | Two end characters are filtered out |
| XON/XOFF | 4096 bytes | 1023 bytes | Software handshake |

### 10.2 Transparent protocol

If the transparent protocol is used, serial data is transmitted in the same format it is received from the serial interface or from the bus side.

The transmit FIFO (First-In-First-Out memory) can store 1023 bytes (1 kbyte) and the receive FIFO can store 4096 bytes (4 kbytes). If the terminal receives another character after the 4095th character, the error pattern is stored in the receive FIFO. All other subsequent characters are ignored.

This protocol supports a CTS hardware handshake.

If the available space in the receive memory is less than 15 bytes, DTR is set to logic 0. As soon as more memory space becomes available again, DTR is set to logic 1.

### 10.3 End-to-end protocol

The serial data is conditioned for the end-to-end protocol.

If serial data is sent from the bus side, two additional characters, the first and second delimiters, are attached for transmission to the serial interface.

A block of serial data sent from the serial interface is only valid if the terminal has received the first and second delimiters. Only then the data can be read via the bus side. The delimiters are filtered out.

Unlike in the transparent protocol, the receive memory is not organized as a FIFO but as a buffer. There are 3 buffers available, each with 330 bytes. If the buffer size of 330 bytes is exceeded without the two delimiters being detected, the previous characters will be ignored and the buffer will be written again.

The transmit FIFO can store 1023 bytes (1 kbyte). The delimiters are attached to, and stored with, the data to be sent.

### 10.4 XON/XOFF protocol

This protocol operates in the same way as the transparent protocol, but uses a software handshake.

Data transmission with this protocol is controlled by the XON and XOFF characters. XON is preset to $11_{hex}$ and XOFF to $13_{hex}$.

If the terminal receives an XOFF, no more serial data will be sent until an XON is received.

The terminal itself will transmit an XOFF if the available space in the receive memory is less than 15 bytes. As soon as more memory space becomes available again, the terminal will transmit a single XON.

Transmission does not depend on the CTS input.

Transmission of serial data is not filtered. So any characters occurring with the code defined for XON and XOFF are transmitted and may trigger undesirable events at the receiver. When serial data is received, the XON and XOFF characters are filtered and are not available as data. Any characters with the XON or XOFF code are lost. Ensure that characters with these codes do not appear in the data stream.

## 11 Programming data/configuration data

### 11.1 Local bus (INTERBUS)

| ID code [hex] | DF | DF | DF |
|---|---|---|---|
| ID code (dec) | 223 | 223 | 223 |
| Length code [hex] | 07 | 0F | 1F |
| Process data channel | 112 bits | 240 bits | 496 bits |
| Input address space | 7 words | 15 words | 31 words |
| Output address space | 7 words | 15 words | 31 words |
| Parameter channel (PCP) | 1 word | 1 word | 1 word |
| Register length (bus) | 8 words | 16 words | 32 words |

### 11.2 Other bus systems or networks

> **i** For the programming data/configuration data of other bus systems, please refer to the corresponding electronic device data sheet (e.g., GSD, EDS).

## 12 Process data

The process image width of the terminal depends on the DIP switch positions and comprises 7, 15 or 31 data words in input and output direction respectively.
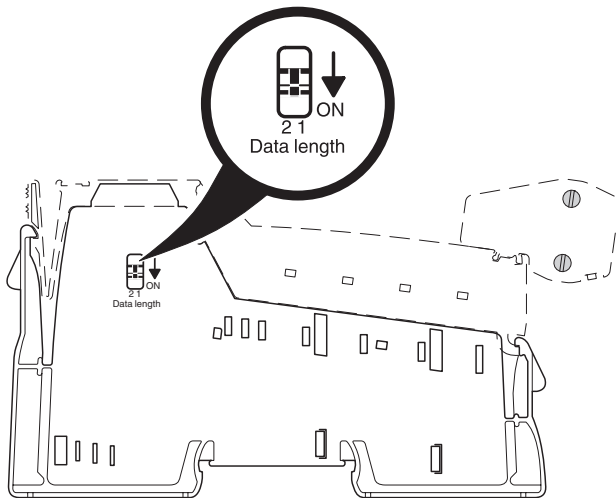
| Switch 2 | Switch 1 | Data width | User data when transmitting and reading |
|----------|----------|-----------|------------------------------------------|
| OFF | OFF | 7 words | 11 bytes |
| OFF | ON | 15 words | 27 bytes |
| ON | OFF | 31 words | 59 bytes |
| ON | ON | Reserved | Reserved |



For the assignment of the illustrated (byte.bit) view to your INTERBUS control or computer system, please refer to the DB GB IBS SYS ADDRESS data sheet.

Figure 10    DIP switches

**Example for the assignment of seven words of process data**

| Word | 0 | | 1 | | 2 | | 3 | | 4 | | 5 | | 6 | |
|------|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Byte in the Motorola format | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 |
| Byte in the Intel format | 1 | 0 | 3 | 2 | 5 | 4 | 7 | 6 | 9 | 8 | 11 | 10 | 13 | 12 |
| OUT | K/P | S | L | D | D | D | D | D | D | D | D | D | D | D |
| IN | K/P | S | L | D | D | D | D | D | D | D | D | D | D | D |

K/P    Command/parameter

S    Control bits (OUT) or status bits (IN)

L    Length: Number of characters to be written (OUT) or to be read (IN)

D    Data

The byte representation in the Motorola format, also called Big Endian (high order byte at starting address) corresponds to the INTERBUS standard representation. All byte representations in the data sheet have this format.

The byte representation in the Intel format is also called Little Endian (low order byte at starting address).

The command is used to determine the function. The actually transmitted data depends on the command.

# 13 Process data word 0

## 13.1 Control word

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|----|----|----|----|----|----|---|---|---|---|---|---|---|---|---|---|
| 0 | Command | | | OUT parameter | | | | Control bits | | | | | | | |

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| DTR | 0 | 0 | 0 | 0 | Reset transmit error | Reset receive error | 0 |

| DTR (if DTR control enabled) | |
|---|---|
| **Code [bin]** | **Meaning** |
| 0 | DTR = logic 0 |
| 1 | DTR = logic 1 |

ℹ The DTR signal can only be controlled when DTR control is enabled via process data, that is when the DTR control bit has been enabled in the configuration (see "Write configuration" command).

| Code [bin] | Meaning |
|---|---|
| **Reset transmit error** | |
| 0 | No action |
| 1 | Reset transmit error |

| Reset receive error | |
|---|---|
| **Code [bin]** | **Meaning** |
| 0 | No action |
| 1 | Reset receive error |

ℹ The listed protocols do not support transmit and receive errors.

**Commands**

| Code [bin] | Code [hex] | Command |
|---|---|---|
| 000 | 0 | Read number of characters received |
| 001 | 1 | Transmit characters |
| 010 | 2 | Store characters temporarily |
| 011 | 3 | OUT parameter = $0_{hex}$: Read characters |
| 011 | 3 | OUT parameter = $C_{hex}$: Read firmware version |
| 011 | 3 | OUT parameter = $D_{hex}$: Read configuration |
| 011 | 3 | OUT parameter = $E_{hex}$: Read counters |
| 100 | 4 | Write configuration |
| 101 | 5 | Toggling command 1: Transmit characters |
| 110 | 6 | Toggling command 2: Store characters temporarily |
| 111 | 7 | Toggling command 3: Read characters |

**Command toggling**

Command toggling is used to execute a command on a terminal again. In this way, a second command code is available for the same function.

This applies for the following commands:
– Transmit characters
– Store characters temporarily
– Read characters
– Read counters

Here, bit 14 is used for toggling. If, for example, you wish to transmit character strings in sequence, use command code $001_{hex}$ for the first transmission and then use $101_{hex}$ and $001_{hex}$ alternately.

### 13.2 Status word

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|----|----|----|----|----|----|---|---|---|---|---|---|---|---|---|---|
| St | Command | | | IN parameter | | | | Status bits | | | | | | | |

Reasons for an error bit set:

St      Error bit

– Invalid parameter for the specified command
– Failure of the I/O voltage

**Status bits**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| DSR | Transmit buffer not empty | Transmit buffer full | Receive buffer full | DCD | Transmit error | Receive error | Receive buffer not empty |

| DSR | |
|---|---|
| Code [bin] | Meaning |
| 0 | DSR = logic 0 |
| 1 | DSR = logic 1 |

| Transmit buffer not empty | |
|---|---|
| Code [bin] | Meaning |
| 0 | Empty |
| 1 | Not empty |

| Transmit buffer full | |
|---|---|
| Code [bin] | Meaning |
| 0 | Not full |
| 1 | Full |

| Receive buffer full | |
|---|---|
| Code [bin] | Meaning |
| 0 | Not full |
| 1 | Full |

| DCD | |
|---|---|
| Code [bin] | Meaning |
| 0 | DCD = logic 0 |
| 1 | DCD = logic 1 |

| Transmit error | |
|---|---|
| Code [bin] | Meaning |
| 0 | No error |
| 1 | Transmit error |

| Receive error | |
|---|---|
| Code [bin] | Meaning |
| 0 | No error |
| 1 | Receive error |

| Code [bin] | Meaning |
|---|---|
| Receive buffer not empty | |
| 0 | Empty |
| 1 | Not empty |

ℹ️ The listed protocols do not support transmit and receive errors.

# 14 Commands

### 14.1 "Read number of characters received" command

The command result is the number of characters that have been received but not yet read.

This command can be used to first reach a minimum number of characters before transmitting the "Read characters" command.

Process data assignment for the "Read number of characters received" command (process data width: 7 words)

| Word | 0 | | 1 | | 2 | | 3 | | 4 | | 5 | | 6 | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Byte | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 |
| OUT | $00_{hex}$ | xx | xx | xx | xx | xx | xx | xx | xx | xx | xx | xx | xx | xx |
| IN | $00_{hex}$ | Status bits | Read number of characters received | | xx | xx | xx | xx | xx | xx | xx | xx | xx | xx |

### 14.2 "Transmit characters" command

The transmit data located in the process data is stored in the transmit memory. From there the data is transmitted directly via the interface. Depending on the process data width a maximum of 11, 27 or 59 characters can be transmitted. Enter the number of characters to be transmitted in the third byte. If there are characters in the intermediate buffer, these are transmitted first. After the command has been executed successfully, the intermediate buffer is cleared.

Process data assignment for the "Transmit characters" command with 11 characters (Z1 - Z11; process data width: 7 words)

| Word | 0 | | 1 | | 2 | | 3 | | 4 | | 5 | | 6 | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Byte | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 |
| OUT | $10_{hex}$ | xx | $0B_{hex}$ | C1 | C2 | C3 | C4 | C5 | C6 | C7 | C8 | C9 | C10 | C11 |
| IN | $10_{hex}$ | Status bits | xx | xx | xx | xx | xx | xx | xx | xx | xx | xx | xx | xx |

Reasons for an error bit set:

– Byte 2 (number of characters to be transmitted)
  = 0 or > maximum user data length (11, 27 or 59 characters)

– "Neutral" interface type

### 14.3 "Store characters temporarily" command

The transmit data located in the process data is stored in an intermediate buffer, which can store up to 330 characters. No characters are transmitted. The "Transmit characters" command is used to transmit temporarily stored data. In this way, blocks of up to 330 characters can be transmitted. They are, for example, divided into 30 telegrams, each with 11 characters.

Reasons for an error bit set:

– Byte 2 (number of characters to be transmitted)
  = 0 or > maximum user data length (11, 27 or 59 characters)

– Not enough space in the intermediate buffer

### 14.4 "Read characters" command

Depending on the process data width set a maximum of 11, 27 or 59 characters can be read. The third byte (byte 2) contains the number of valid characters available in the input data.

Process data assignment for the "Read characters" command with eleven characters (C1 - C11)

| Word | 0 | | 1 | | 2 | | 3 | | 4 | | 5 | | 6 | |
|------|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Byte | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 |
| OUT | $30_{hex}$ | xx | xx | xx | xx | xx | xx | xx | xx | xx | xx | xx | xx | xx |
| IN | $30_{hex}$ | Status bits | $0B_{hex}$ | C1 | C2 | C3 | C4 | C5 | C6 | C7 | C8 | C9 | C10 | C11 |

Bit 7 in the third input byte (byte 2)

| Byte 2, bit 7 | Effect | Protocol |
|---------------|--------|----------|
| 0 | The characters read are the last ones in the block received. | E.g., end-to-end |
| 1 | There are still more characters to be read from the block received. | E.g., end-to-end |

### 14.5 "Read counters" command

This command can be used to read several counters. The counters are used for interface diagnostics.

Process data assignment for the "Read counters" command

| Word | 0 | | 1 | | 2 | | 3 | |
|------|---|---|---|---|---|---|---|---|
| Byte | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| OUT | $3E_{hex}$ | $00_{hex}$ | xx | xx | xx | xx | xx | xx |
| IN | $3E_{hex}$ | Status bits | Number of valid characters received | | Number of invalid characters received (parity, overrun or framing errors) | | Number of characters transmitted | |

### 14.6  "Write configuration" command

**Output words 0 to 6**

Process data assignment for the "Write configuration" command

| Word | 0 | | 1 | | 2 | | 3 | | 4 | | 5 | | 6 | |
|------|-----|-----|-----|------|---------|---------|------|------|------|------|-----|-----|-----|-----|
| Byte | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 |
| OUT | $40_{hex}$ | xx | T/P | B/DB | 1st del | 2nd del | ErrP | Uni1 | Uni2 | Uni3 | res. | res. | Tv | Tn |
| IN | $40_{hex}$ | Status bits | - | - | - | - | - | - | - | - | - | - | - | - |

**Element value range**

| i | The options in bold are default settings. |
|---|---|

**Byte 2 (T/P: interface type/protocol)**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------|------|-------|-------|----------|----|----|----|
| res. | DTR | Interface type | | Protocol | | | |

| DTR: DTR control | |
|---|---|
| **Code [bin]** | **Meaning** |
| **0** | **Automatic** |
| 1 | Can be controlled via process data |

| Interface type | |
|---|---|
| **Code [bin]** | **Meaning** |
| 00 | RS-232 |
| 01 | RS-485 |
| 10 | RS-422 |
| **11** | **Neutral** |

| i | Select an RS-xxx interface type. With the "neutral" default setting, no data is transmitted or received. When using the default setting, an error message is generated and the error bit is set. |
|---|---|

| Protocol | |
|---|---|
| **Code [hex]** | **Meaning** |
| **0** | **Transparent** |
| 1 | End-to-end |
| 2 | XON/XOFF |
| Other | Reserved |

**Byte 3 (D/DB: baud rate/data width)**

| Baud rate | |
|---|---|
| **Code [hex]** | **Wert [baud]** |
| 0 | 110 |
| 1 | 300 |
| 2 | 600 |
| 3 | 1200 |
| 4 | 1800 |
| 5 | 2400 |
| 6 | 4800 |
| **7** | **9600** |
| 8 | 15625 |
| 9 | 19200 |
| A | 38400 |
| B | 57600 |
| C | 115200 |
| D | 230400 |
| E | 250000 |
| F | Direct (Uni1 ... 3) |

i The specified baud rates of 110 baud to 250000 baud are adequate for most applications. However, you can freely choose the baud rate by direct programming. For this, use the $0F_{hex}$ baud rate code.

| Data width | | | |
|---|---|---|---|
| **Code [hex]** | **Meaning** | | |
| **-** | **Data bits** | **Parity** | **Stop bits** |
| 0 | 7 | Even | 1 |
| 1 | 7 | Odd | 1 |
| 2 | 8 | Even | 1 |
| 3 | 8 | Odd | 1 |
| 4 | 8 | Without | 1 |
| 5 | 7 | Without | 1 |
| 6 | 7 | Even | 2 |
| 7 | 7 | Odd | 2 |
| 8 | 8 | Even | 2 |
| 9 | 8 | Odd | 2 |
| A | 8 | Without | 2 |
| B | 7 | Without | 2 |
| C | 8 | Constantly at 0 | 1 |
| D | 8 | Constantly at 1 | 1 |
| E | 6 | Without | 1 |
| F | Directly (Uni1) | | |

i The specified combinations of data width, parity and stop bits are adequate for most applications. However, you can freely choose the combination by direct programming. For this, use the $0F_{hex}$ data width code.

**Byte 4 and 5 (1st del, 2nd del: 1st and 2nd delimiters)**

The 1st delimiter and 2nd delimiter contain the end characters for the end-to-end and dual buffer protocols.

When selecting the end-to-end protocol, the two delimiters can be adjusted.

Default settings:

| Delimiter | Default value | ASCII character |
|---|---|---|
| 1st del | $0D_{hex}$ | CR = carriage return |
| 2nd del | $0A_{hex}$ | LF = line feed |

**Byte 6 (ErrP: error pattern)**

The error pattern contains the character that is written to the FIFO if a character was received with errors (e.g., in the event of a parity error).

The error pattern is used for the following protocols:

– Transparent
– End-to-end
– XON/XOFF

| Error pattern | |
|---|---|
| **Code [hex]** | **Meaning** |
| **24** | **$** |
| xx | Any character |
| 00 | If a character is received with an error, no error pattern is stored. |
| FF | The invalid character is stored instead of the error pattern. |

**Byte 7 ... 9 (Uni1 ... Uni3: universal byte 1 ... 3)**

These bytes can be used as universal bytes for direct specification of the baud rate or the data width.

| Byte 7 | | | | | | | | Byte 8 | | | Byte 9 | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | 7 | ... | 0 | 7 | ... | 0 |
| Data width | | | Baud rate | | | | | | | | | | |

**Direct specification of baud rate**

If the provided baud rates do not correspond to the desired baud rate, the baud rate can be directly specified. For this, write the desired baud rate on the Uni1 to Uni3 fields in the form of a numeric value. Transfer is right-aligned. However, only the 2 lower bits of Uni1 are permitted.

The maximum value is 18 bits. The maximum value = $3FFFF_{hex}$ which corresponds to 262143 baud.

**Direct specification of the data width**

If the provided combinations of data width, parity and stop bits are inadequate, they can be directly specified. Use the Uni1 byte. Bit 1 and bit 0 are reserved.

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| Parity | | Enable parity | Stop bits | Data bits | | res. | res. |

| Parity | |
|---|---|
| **Code [bin]** | **Meaning** |
| **00** | **Odd** |
| 01 | Even |
| 10 | Constantly at 1 |
| 11 | Constantly at 0 |

| Enable parity | |
|---|---|
| **Code [bin]** | **Meaning** |
| **0** | **Disable** |
| 1 | Enable |

| Stop bits | |
|---|---|
| **Code [bin]** | **Meaning** |
| **0** | **1 stop bit** |
| 1 | 2 stop bits |

| Data bits | |
|---|---|
| **Code [bin]** | **Meaning** |
| **00** | **5 bits** |
| 01 | 6 bits |
| 10 | 7 bits |
| 11 | 8 bits |

Example for direct specification of the baud rate:

– Baud rate: 100000 baud
– $100000_{dec}$ = 01 86 $A0_{hex}$

| Word | 0 | | 1 | | 2 | | 3 | | 4 | | 5 | | 6 | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Byte | 1 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 |
| OUT | $40_{hex}$ | $xx_{hex}$ | T/P | B/DB | res. | res. | ErrP | Uni1 | Uni2 | Uni3 | res. | res. | Tv | Tn |
| OUT | $40_{hex}$ | $xx_{hex}$ | xx | $Fx_{hex}$ | 00 | 00 | xx | 01 | 86 | A0 | 00 | 00 | xx | xx |

**Byte 12 und 13 (Tv, Tn: lead time, lag time)**

These two times are used for the RS-232 protocol

Tv          Lead time
Tn          Lag time

The times are specified with a resolution of 1 ms. This allows for waiting times of 1 ms ... 255 ms.

When transmitting data via RS-232 the RTS signal is set. If Tv does not equal 0, transmission only starts after this time has elapsed. After all the data is sent and Tn does not equal 0, the RTS signal is only set to LOW after this time has elapsed.

# 15    PCP communication

## 15.1    General information

> **i** For information on PCP communication, please refer to the PCP user manuals (see Ordering data).

By default upon delivery, the terminal is configured according to the default settings (under configuration). The terminal can be configured using process data or PCP to suit your application.

In PCP mode, the terminal is configured with the "Config Table" object.

> **i** The programs IBS CMD (for standard controller boards) and PC WorX (for Controllers (ILC), Field Controllers (FC) and Remote Field Controllers (RFC)) are available for the configuration and parameterization of your INTERBUS system.
> For additional information, please refer to the documentation of the software used.

## 15.2 Object dictionary for PCP communication

| Index | Data type | A | L | Meaning | Object name | Rights |
|---|---|---|---|---|---|---|
| 0080$_{hex}$ | Record | 1 | 16 | Terminal configuration | Config Table | rd/wr |

| | | | |
|---|---|---|---|
| A | Number of elements | rd | Read access permitted |
| L | Length of an element in bytes | wr | Write access permitted |

## 15.3 "Config Table" object

Configure the terminal using this object.

Access this object via subindex 0, i.e., read or write the entire project.

The first twelve bytes correspond to byte 2 through byte 13 of the "Write configuration" command. The other four bytes are reserved.

| Byte | Meaning | Default value |
|---|---|---|
| 0 | Type, protocol | 00$_{hex}$ |
| 1 | Baud rate, data width | 72$_{hex}$ |
| 2 | 1st delimiter | 0D$_{hex}$ |
| 3 | 2nd delimiter | 0A$_{hex}$ |
| 4 | Error pattern | 24$_{hex}$ |
| 5 ... 7 | Uni1 ... Uni3 | 00 00 00$_{hex}$ |
| 8, 9 | Reserved | 00 00$_{hex}$ |
| 10 | Tv | 00$_{hex}$ |
| 11 | Tn | 00$_{hex}$ |
| 12 ... 15 | Reserved | 00 00 00 00$_{hex}$ |

If an invalid configuration is specified, a negative confirmation is generated with error message 08$_{hex}$, 00$_{hex}$ or xx30$_{hex}$. The low byte of the Additional_Error_Code is 30$_{hex}$ (value is out of range), the high byte contains the number of the affected byte.

单击下面可查看定价，库存，交付和生命周期等信息

>>Phoenix Contact(菲尼克斯)