RESI-xxx-SIO
RESI-xxx-ETH

Our series of intelligent IO modules based on MODBUS protocol and ASCII text protocol for building automation and industrial automation.

Content

1 Our portfolio

We offer the following IO products:

1.1 ULTRA SLIM serial IO modules

Our ultra slim IO modules are extreme small modules and offer various IOs, always in combination with a RS232 and RS485 interface.
The dimension of those IO modules is very slim:
- Only 17.5x90x58mm (WxHxD) in size

Those IO modules offer the following protocols:
- a MODBUS/RTU slave protocol
- a simple ASCII text protocol

The modules support the following baud rates:
- from 300bd up to 256000bd
- none, even and odd parity
- one or two stop bits.

All our modules are designed for use with 12 to 48Vdc power supplies, so they offer a broad range of applications. The serial interface is always galvanically insulated from the IOs on the module. The modules are designed for mounting on a DIN EN50022 rail.

Figure: Sample of an ultra slim IO module with serial interface

1.2 ULTRA SLIM Ethernet IO modules

Our ultra slim IO modules are extreme small modules and offer various IOs, always in combination with an Ethernet interface.
The dimension of those IO modules is very slim:
- Only 35.8x90x58mm (WxHxD) in size

Those IO modules offer various protocols:
- MODBUS/TCP server protocol
- MODBUS/RTU slave protocol via Ethernet
- simple ASCII text protocol via socket

The Ethernet interface offers
- RJ45 interface mit 10MBit/100MBit
- support of AUTO - MDIX

All our modules are designed for use with 12 to 48Vdc power supplies, so they offer a broad range of applications. The serial interface is always galvanically insulated from the IOs on the module. The modules are designed for mounting on a DIN EN50022 rail.

Figure: Sample of an ultra slim IO module with Ethernet interface

1.3 BIGIO serial IO modules

Our BIGIO modules are extreme compact modules with many IOs, always in combination with a serial RS485 interface. We offer two different housings depending on the amount of IOs implemented in the IO module.
The dimension of the XT8 IO modules is:

- 142,3x110x62mm (WxHxD) in size

The dimension of the XT12 IO modules is:

- 213x110x62mm (WxHxD) in size

Those IO modules offer various protocols:

- MODBUS/TCP server protocol
- MODBUS/RTU slave protocol via Ethernet
- simple ASCII text protocol via socket

The Ethernet interface offers

- RJ45 interface mit 10MBit/100MBit
- support of AUTO - MDIX

All our modules are designed for use with 12 to 48Vdc power supplies, so they offer a broad range of applications. The serial interface is always galvanically insulated from the IOs on the module. The modules are designed for mounting on a DIN EN50022 rail. but the modules offer also a wall mounting option.

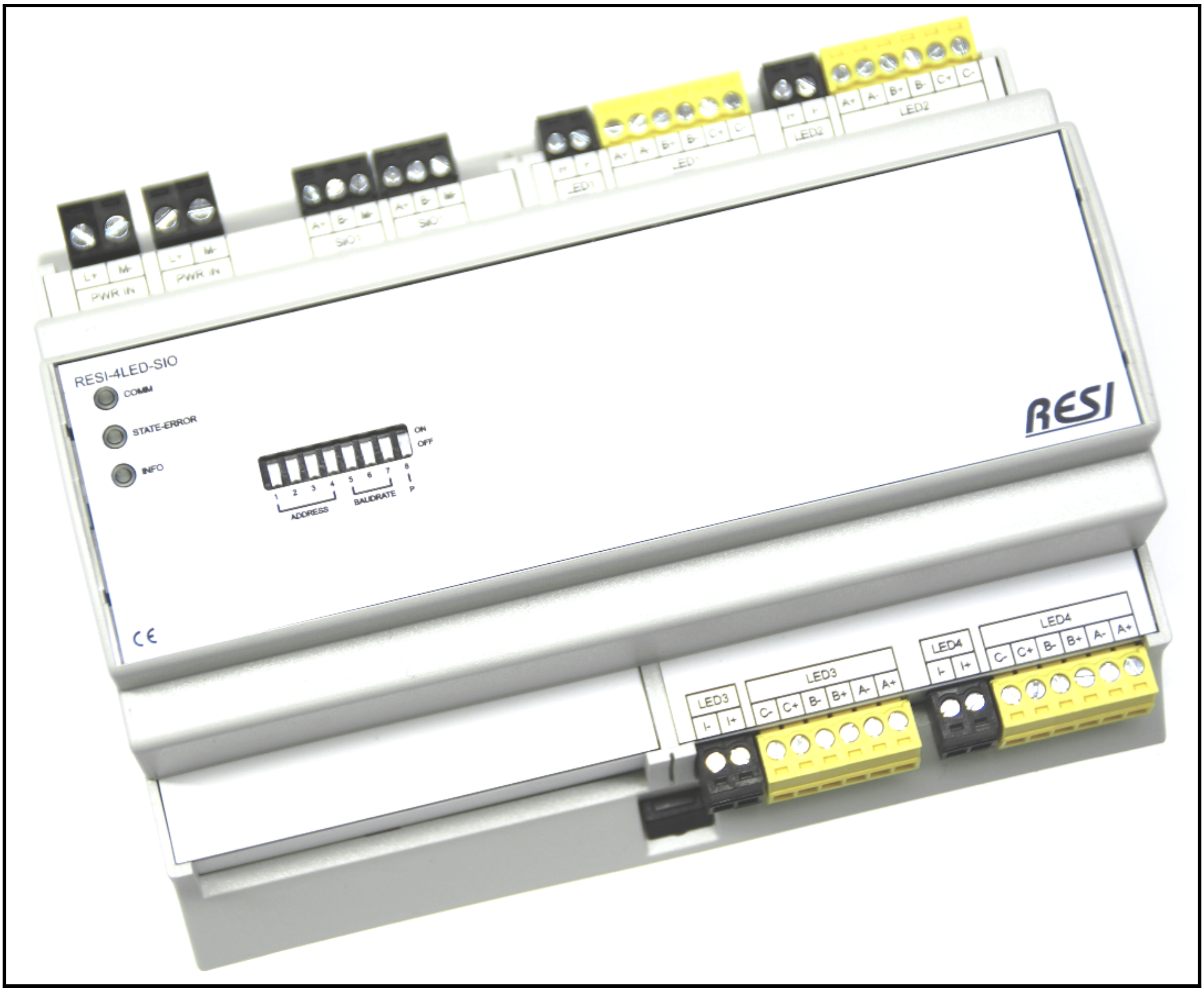


Figure: Sample of a XT8 IO module with serial interface

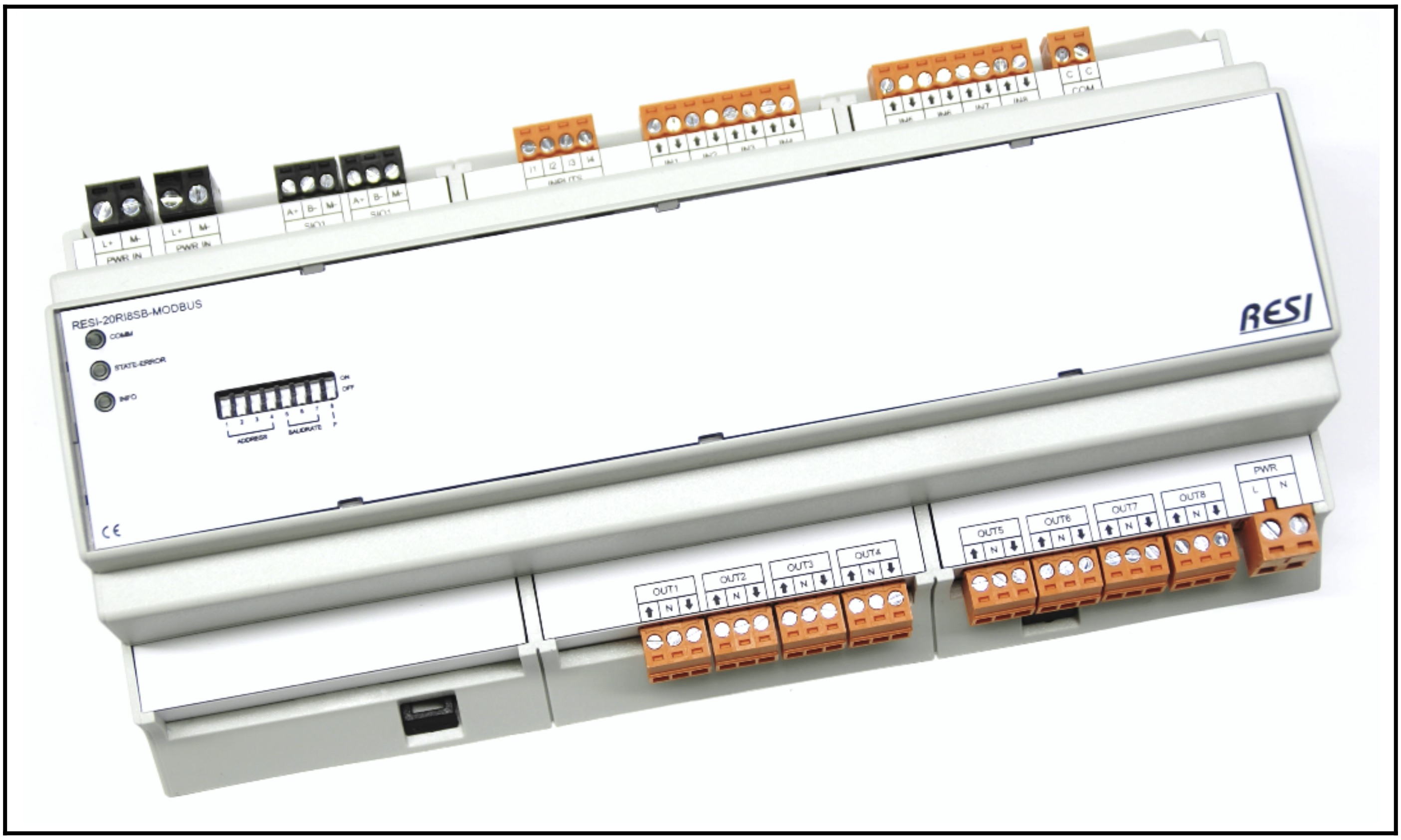


Figure: Sample of a XT12 IO module with serial interface

2 Declaration of conformity

2.1 CE

All products have passed the CE tests for environmental specifications when shielded cables are used for external wiring. We recommend the use of shielded cables.

2.2 Safety instructions

Danger to life through electrical current!

Only skilled personal trained in electro-engineering should perform the described steps in the following chapters. Please observe the country specific rules and standards. Do not perform any electrical work while the device is connected to power.

Pay attention to the following rules:

1. Disconnect the system from power
2. Secure the system against automatic power on
3. Check that the system is de-energized
4. Cover other energized parts of the system

IMPORTANT HINT: Before you start with the installation and the initial setup of the device, you have to read this document and the attached installation guide and the actual manual for the device very carefully. You have to follow all the herein given information very accurate!

- ❑ Only authorized and qualified personnel are allowed to install and setup the device!
- ❑ The connection of the device must be done in de-energized state!
- ❑ Do not perform any electrical work while the device is connected to power!
- ❑ Disable and secure the system against any automatic restart or power on procedure!
- ❑ The device must be operated with the defined voltage level!
- ❑ Supply voltage jitters must not exceed the technical specifications and tolerances given in the technical manuals for the product. If you do not obey this issue, the proper performance of the device cannot be guaranteed. This can lead to fail functions of the device and in worst case to a complete breakdown of the device!
- ❑ You have to obey the current EMC regulations for wiring!
- ❑ All signal, control and supply voltage cables must be wired in a way, that no inductive or capacitive interference or any other severe electrical noise disturbance may interfere with the device. Wrong wiring can lead to a malfunction of the device!
- ❑ For signal or sensor cables you have to use shielded cables, to avoid damages through induction!
- ❑ You have to obey and to apply the current safety regulations given by the ÖVE, VDE, the countries, their control authorities, the TÜV or the local energy supply company!
- ❑ Obey country-specific laws and standards!
- ❑ The device must be used for the intended purpose of the manufacturer!
- ❑ No warranties or liabilities will be accepted for defects and damages resulting from improper or incorrect usage of the device!
- ❑ Subsequent damages, which results from faults of this device, are excluded from warranty and liability!
- ❑ Only the technical data, wiring diagrams and operation instructions, which are part to the product shipment are valid!
- ❑ The information on our homepage, in our data sheets, in our manuals, in our catalogs or published by our partners can deviate from the product documentation and is not necessarily always actual, due to constant improvement of our products for technical progress!
- ❑ In case of modification of our devices made by the user, all warranty and liability claims are lost!
- ❑ The installation has to fulfil the technical conditions and specifications (e.g. operating temperatures, power supply, …) given in the devices documentation!
- ❑ Operating our device close to equipment, which do not comply with EMC directives, can influence the functionality of our device, leading to malfunction or in worst case to a breakdown of our device!
- ❑ Our devices must not be used for monitoring applications, which solely serve the purpose of protecting persons against hazards or injury, or as an emergency stop switch for systems or machinery, or for any other similar safety-relevant purposes!
- ❑ Dimensions of the enclosures or enclosures accessories may show slight tolerances on the specifications provided in these instructions!
- ❑ Modifications of this documentation is not allowed!

In case of a complaint, only complete devices returned in original packing will be accepted!

3 Our Portfolio

Here you find a list of all available IO modules:

3.1 Digital IO modules

PRODUCT	DESCRIPTION	HOUSING TYPE
RESI-2RI-SIO	2xdigital inputs for 12-250Vac/dc signals	ULTRA SLIM 17.5mm
RESI-2RI-ETH	2xdigital inputs for 12-250Vac/dc signals	ULTRA SLIM 35.8mm
RESI-4DI-SIO	4xdigital inputs for 12-48Vdc signals	ULTRA SLIM 17.5mm
RESI-4DI-ETH	4xdigital inputs for 12-48Vdc signals	ULTRA SLIM 35.8mm
RESI-14RI-SIO	14xdigital inputs for 12-250Vac/dc signals	BIGIO XT8 142.3mm
RESI-48RI-SIO	48xdigital inputs for 12-250Vac/dc signals	BIGIO XT12 213mm
RESI-32DI-SIO	32xdigital inputs for 12-48Vdc signals	BIGIO XT8 142.3mm
RESI-64DI-SIO	64xdigital inputs for 12-48Vdc signals	BIGIO XT12 213mm
RESI-1RO-SIO	1xrelay output with max. 230Vac,30Vdc, 8A and NO+NC contacts	ULTRA SLIM 17.5mm
RESI-1RO-ETH	1xrelay output with max. 230Vac,30Vdc, 8A and NO+NC contacts	ULTRA SLIM 35.8mm
RESI-2RO-SIO	2xrelay output with max. 230Vac,30Vdc, 8A and NO contacts	ULTRA SLIM 17.5mm
RESI-2RO-ETH	2xrelay output with max. 230Vac,30Vdc, 8A and NO contacts	ULTRA SLIM 35.8mm
RESI-2SSR-1A-SIO	2xsolid state relay outputs with max. 250Vac, 250Vdc, 1A and NO contacts	ULTRA SLIM 17.5mm
RESI-2SSR-1A-ETH	2xsolid state relay outputs with max. 250Vac, 250Vdc, 1A and NO contacts	ULTRA SLIM 35.8mm
RESI-2SSR-6A-SIO	2xsolid state relay outputs with max. 60Vac, 60Vdc, 6A and NO contacts	ULTRA SLIM 17.5mm
RESI-2SSR-6A-ETH	2xsolid state relay outputs with max. 60Vac, 60Vdc, 6A and NO contacts	ULTRA SLIM 35.8mm
RESI-4DO-SIO	4xdigital outputs with max. 2-32Vdc, 300mA	ULTRA SLIM 17.5mm
RESI-4DO-ETH	4xdigital outputs with max. 2-32Vdc, 300mA	ULTRA SLIM 35.8mm
RESI-8CO-SIO	8xrelay output with max. 230Vac,30Vdc, 8A and NO+NC contacts	BIGIO XT8 142.3mm
RESI-8COBI-SIO	8xbistable relay output with max. 230Vac,30Vdc, 8A and NO+NC contacts	BIGIO XT8 142.3mm
RESI-10SSR-1A-SIO	10xsolid state relay outputs with max. 250Vac, 250Vdc, 1A and NO contacts	BIGIO XT8 142.3mm
RESI-10SSR-6A-SIO	10xsolid state relay outputs with max. 60Vac, 60Vdc, 6A and NO contacts	BIGIO XT8 142.3mm
RESI-30DO-SIO	30xdigital outputs with max. 2-32Vdc, 300mA	BIGIO XT8 142.3mm
RESI-60DO-SIO	60xdigital outputs with max. 2-32Vdc, 300mA	BIGIO XT12 213mm

PRODUCT	DESCRIPTION	HOUSING TYPE
RESI-S16DI8PO-SIO	16xdigital inputs for 12-48Vdc signals 8xbistable power relais max. 250Vac, 16A, 200µF	BIGIO XT8 142.3mm
RESI-S8PO-SIO	8xbistable power relais max. 250Vac, 16A, 200µF	BIGIO XT8 142.3mm
RESI-16RI8PO-SIO	16xdigital inputs for 10-250Vac/dc signals 8xbistable power relais max. 250Vac, 16A, 200µF	BIGIO XT12 213mm
RESI-8PO-SIO	8xbistable power relais max. 250Vac, 16A, 200µF	BIGIO XT12 213mm
RESI-10RI4SB-SIO	10xdigital inputs for 10-250Vac/dc signals 8xrelais max. 250Vac, 6A, $AgSNO_2$ contacts	BIGIO XT8 142.3mm
RESI-4SB-SIO	8xrelais max. 250Vac, 6A, $AgSNO_2$ contacts	BIGIO XT8 142.3mm
RESI-20RI8SB-SIO	20xdigital inputs for 10-250Vac/dc signals 16xrelais max. 250Vac, 6A, $AgSNO_2$ contacts	BIGIO XT12 213mm
RESI-8SB-SIO	16xrelais max. 250Vac, 6A, $AgSNO_2$ contacts	BIGIO XT12 213mm
RESI-10RI8RO-SIO	10xdigital inputs for 10-250Vac/dc signals 8xrelais max. 250Vac, 6A, $AgSNO_2$ contacts	BIGIO XT8 142.3mm
RESI-8RO-SIO	8xrelais max. 250Vac, 6A, $AgSNO_2$ contacts	BIGIO XT8 142.3mm
RESI-20RI16RO-SIO	20xdigital inputs for 10-250Vac/dc signals 16xrelais max. 250Vac, 6A, $AgSNO_2$ contacts	BIGIO XT12 213mm
RESI-16RO-SIO	16xrelais max. 250Vac, 6A, $AgSNO_2$ contacts	BIGIO XT12 213mm

3.2 Analog IO modules

PRODUCT	DESCRIPTION	HOUSING TYPE
RESI-4AIU-SIO	4xanalog inputs for -10..+10Vdc signals, 16 bit, ±0.1%	ULTRA SLIM 17.5mm
RESI-4AIU-ETH	4xanalog inputs for -10..+10Vdc signals, 16 bit, ±0.1%	ULTRA SLIM 35.8mm
RESI-12AIU-SIO	12xanalog inputs for -10..+10Vdc signals, 16 bit, ±0.1%	BIGIO XT8 142.3mm
RESI-4AOU-SIO	4xanalog outputs for -10..+10Vdc signals, 16 bit, ±0.1%	ULTRA SLIM 17.5mm
RESI-4AOU-ETH	4xanalog outputs for -10..+10Vdc signals, 16 bit, ±0.1%	ULTRA SLIM 35.8mm
RESI-12AOU-SIO	12xanalog outputs for -10..+10Vdc signals, 16 bit, ±0.1%	BIGIO XT8 142.3mm
RESI-2AIU2AOU-SIO	2xanalog inputs for 0..+10Vdc signals, 12 bit, ±0.5% 2xanalog outputs for 0..+10Vdc signals, 12 bit, ±0.5%	ULTRA SLIM 17.5mm
RESI-2AIU2AOU-ETH	2xanalog inputs for 0..+10Vdc signals, 12 bit, ±0.5% 2xanalog outputs for 0..+10Vdc signals, 12 bit, ±0.5%	ULTRA SLIM 35.8mm

3.3 Temperature IO modules

PRODUCT	DESCRIPTION	HOUSING TYPE
RESI-2RTD-SIO	2xinputs for RTD temperature sensors (PT100, PT1000, NI1000, NI120,…) ±0.1% 2-wire, 3-wire and 4 wire connection	ULTRA SLIM 17.5mm
RESI-2RTD-ETH	2xinputs for RTD temperature sensors (PT100, PT1000, NI1000, NI120,…) ±0.1% 2-wire, 3-wire and 4 wire connection	ULTRA SLIM 35.8mm
RESI-8RTD-SIO	8xinputs for RTD temperature sensors (PT100, PT1000, NI1000, NI120,…) ±0.1% 2-wire, 3-wire and 4 wire connection	BIGIO XT8 142.3mm
RESI-8RTD2-SIO	8xinputs for RTD temperature sensors (PT100, PT1000, NI1000, NI120,…) ±0.1% 2-wire connection	BIGIO XT8 142.3mm

3.4 IO modules for light systems

PRODUCT	DESCRIPTION	HOUSING TYPE
RESI-1LED-SIO	1x3 PWM outputs for LED stripes <60Vdc, <5A per PWM channel	ULTRA SLIM 17.5mm
RESI-1LED-ETH	1x3 PWM outputs for LED stripes <60Vdc, <5A per PWM channel	ULTRA SLIM 35.8mm
RESI-4LED-SIO	4x3 PWM outputs for LED stripes <60Vdc, <5A per PWM channel	BIGIO XT8 142.3mm
RESI-DMX-SIO	1xDMX512 master interface for one DMX universe	ULTRA SLIM 17.5mm
RESI-DMX-ETH	1xDMX512 master interface for one DMX universe	ULTRA SLIM 35.8mm
RESI-DALI-SIO	1xDALI master interface for 64 DALI ballasts	ULTRA SLIM 17.5mm
RESI-DALI-ETH	1xDALI master interface for 64 DALI ballasts	ULTRA SLIM 35.8mm

3.5 Special IO modules

PRODUCT	DESCRIPTION	HOUSING TYPE
RESI-1S0-SIO	1xS0 impulse input for smart meter with S0 interface	ULTRA SLIM 17.5mm
RESI-1S0-ETH	1xS0 impulse input for smart meter with S0 interface	ULTRA SLIM 35.8mm
RESI-2S0-SIO	2xS0 impulse input for smart meter with S0 interface	ULTRA SLIM 17.5mm
RESI-2S0-ETH	2xS0 impulse input for smart meter with S0 interface	ULTRA SLIM 35.8mm
RESI-1EGYDCS-SIO	1xDC metering with external shunt, DC voltage: 0..100Vdc, max. 255A shunt	ULTRA SLIM 17.5mm
RESI-1EGYDCS-ETH	1xDC metering with external shunt, DC voltage: 0..100Vdc, max. 255A shunt	ULTRA SLIM 35.8mm
RESI-1EGYDC-SIO	1xDC metering with external hall sensor, DC voltage: 0..100Vdc, max. 80A	ULTRA SLIM 17.5mm
RESI-1EGYDC-ETH	1xDC metering with external hall sensor, DC voltage: 0..100Vdc, max. 80A	ULTRA SLIM 35.8mm

3.6 MBUS IO modules

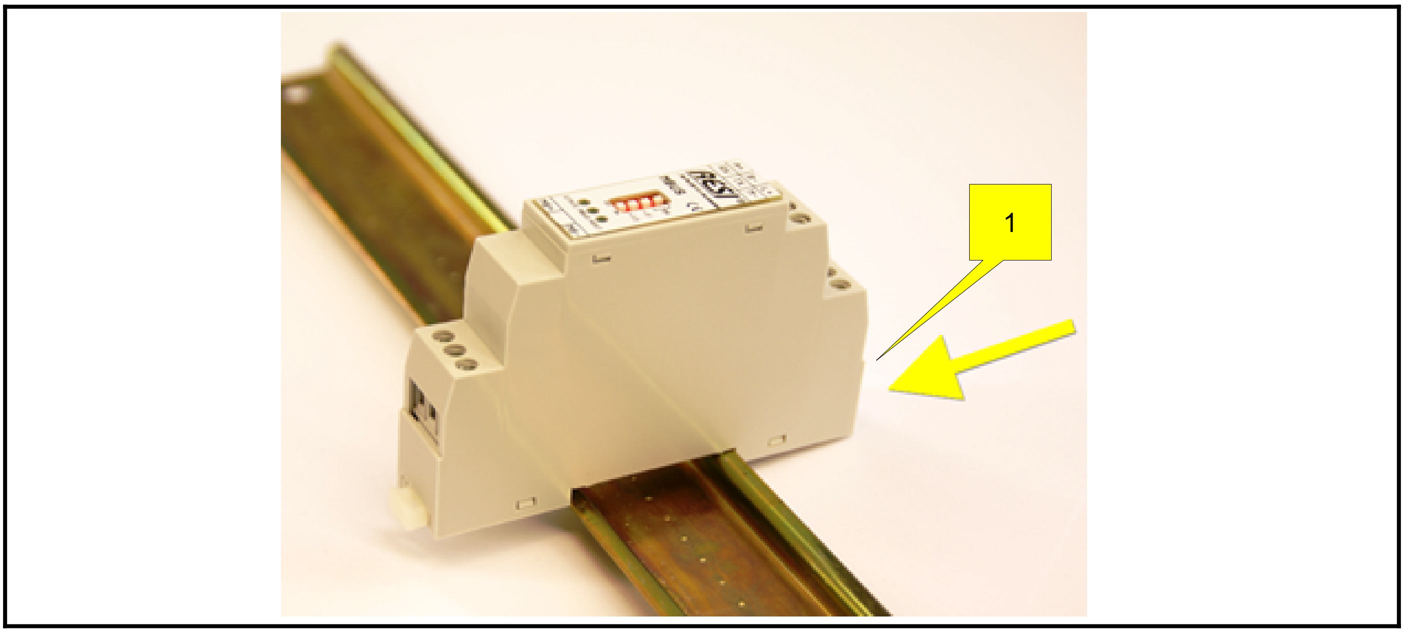
PRODUCT	DESCRIPTION	HOUSING TYPE
RESI-MBUS2-SIO	MBUS master to read data from 2 smart meter with MBUS interface	ULTRA SLIM 17.5mm
RESI-MBUS8SIO	MBUS master to read data from 8 smart meter with MBUS interface	ULTRA SLIM 17.5mm
RESI-MBUS24-SIO	MBUS master to read data from 24 smart meter with MBUS interface	ULTRA SLIM 17.5mm
RESI-MBUS48-SIO	MBUS master to read data from 48 smart meter with MBUS interface	ULTRA SLIM 17.5mm
RESI-MBUS64-SIO	MBUS master to read data from 64 smart meter with MBUS interface	ULTRA SLIM 17.5mm
RESI-MBUS2-ETH	MBUS master to read data from 2 smart meter with MBUS interface	ULTRA SLIM 35.8mm
RESI-MBUS8-ETH	MBUS master to read data from 8 smart meter with MBUS interface	ULTRA SLIM 35.8mm
RESI-MBUS24-ETH	MBUS master to read data from 24 smart meter with MBUS interface	ULTRA SLIM 35.8mm
RESI-MBUS48-ETH	MBUS master to read data from 48 smart meter with MBUS interface	ULTRA SLIM 35.8mm
RESI-MBUS64-ETH	MBUS master to read data from 64 smart meter with MBUS interface	ULTRA SLIM 35.8mm

4 Mounting

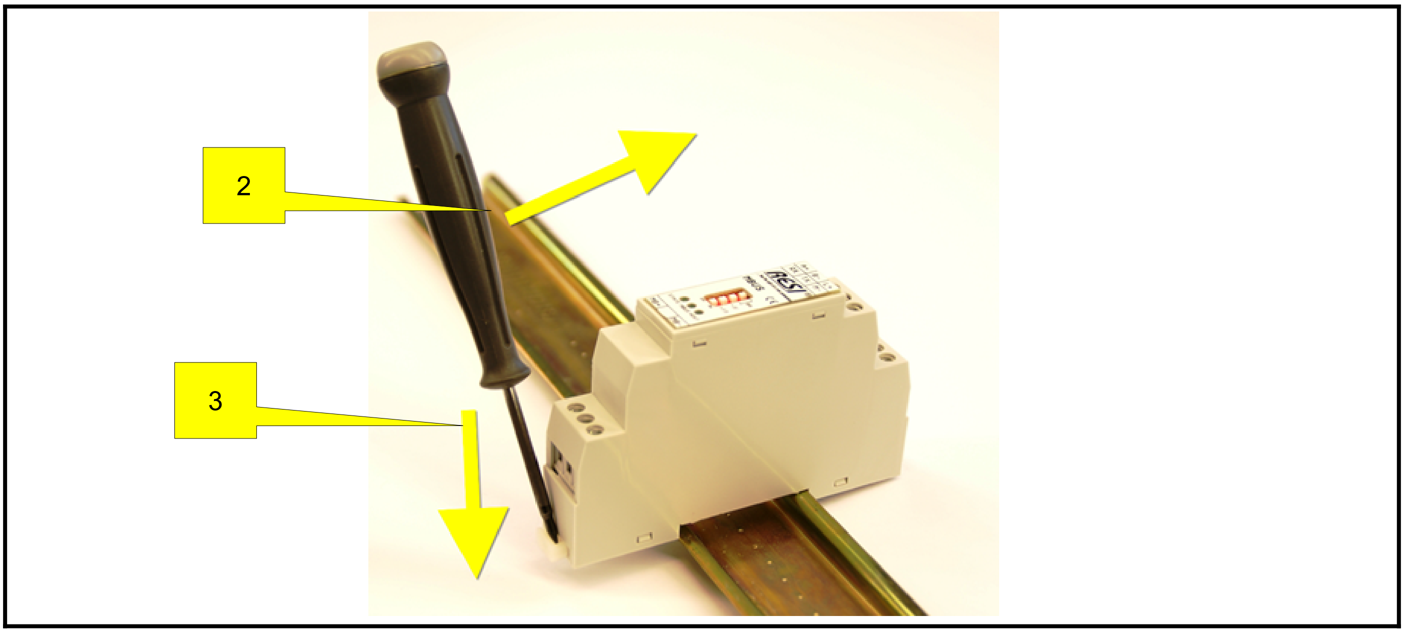
Here you will find the different mounting options for our modules

4.1 Mounting for ULTRA SLIM IOs

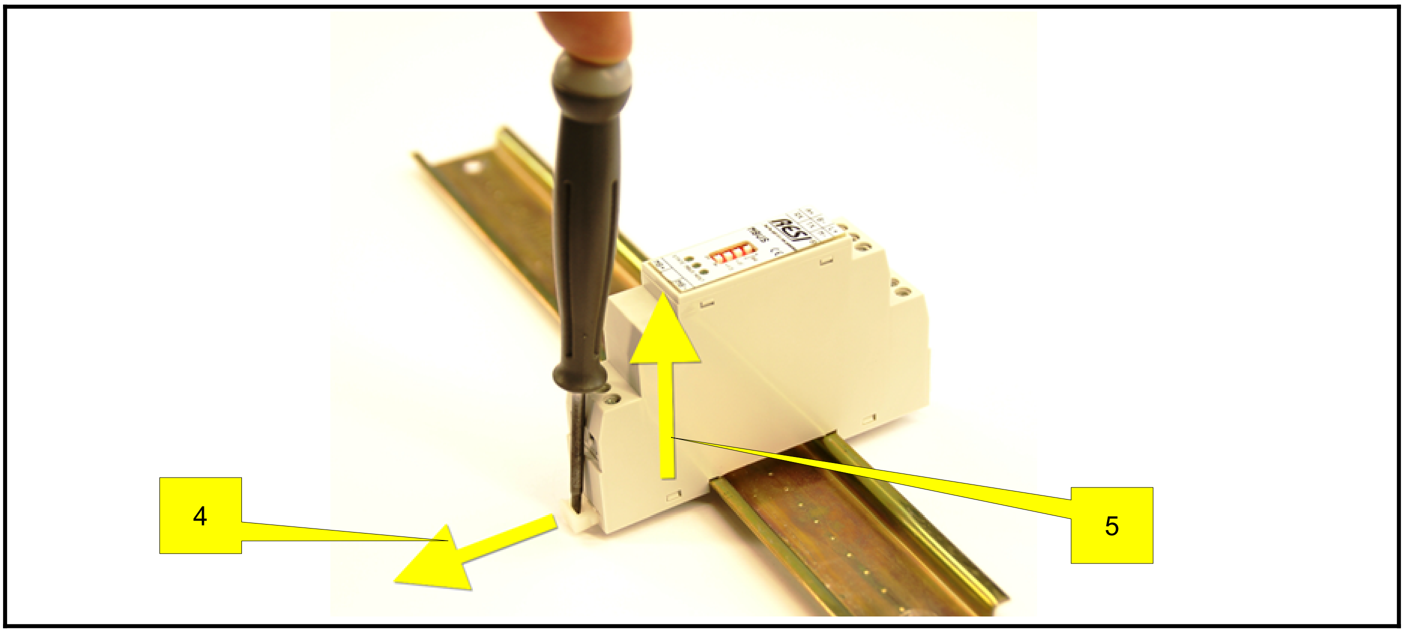
Our IO modules are designed for mounting on a 35mm DIN-EN50022 rail.
At first, put the modules with the top side on the DIN rail (1).



Then open the clamp lever on the bottom side with a screw driver (2) and press the device on the DIN rail (3). Release the clamp lever. The module is now placed correctly on the DIN rail.



To dismount the module from the DIN rail first open the clamp lever with a screwdriver on the bottom side (4). Hold the clamp lever opened while you lift the module from the DIN rail (5). Then remove the module from the bar with while pulling it on the top side.

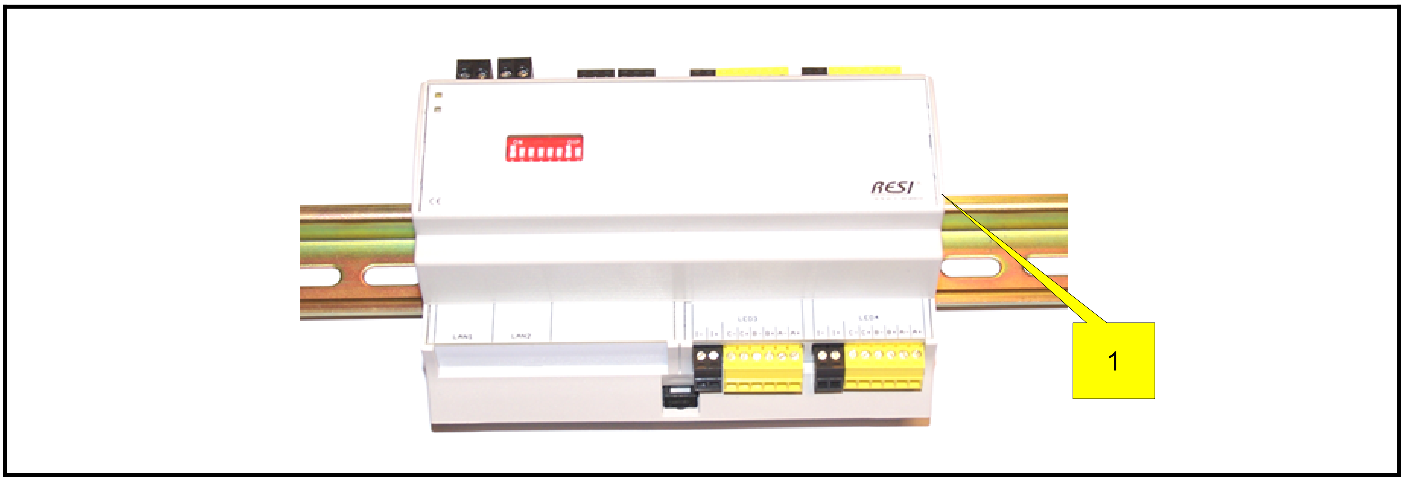


4.2 Mounting for BIG IOs XT8 or XT12

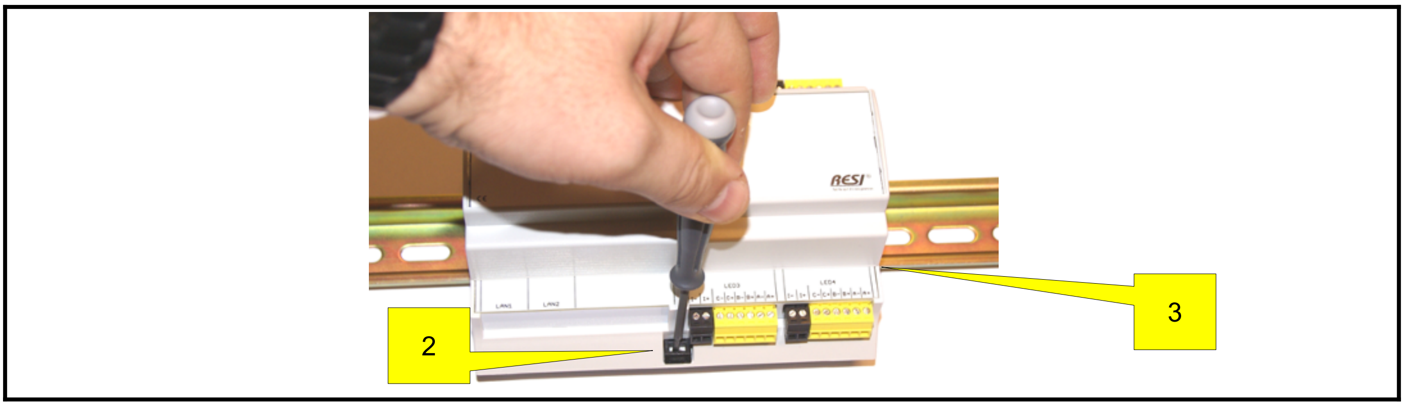
Our BIG IO modules are designed for mounting onto a 35mm DIN-EN50022 rail or for wall mounting. Please not, that in the following mounting description we use only symbolic photos of our IO modules.

4.2.1 Mounting on a DIN EN50022 rail

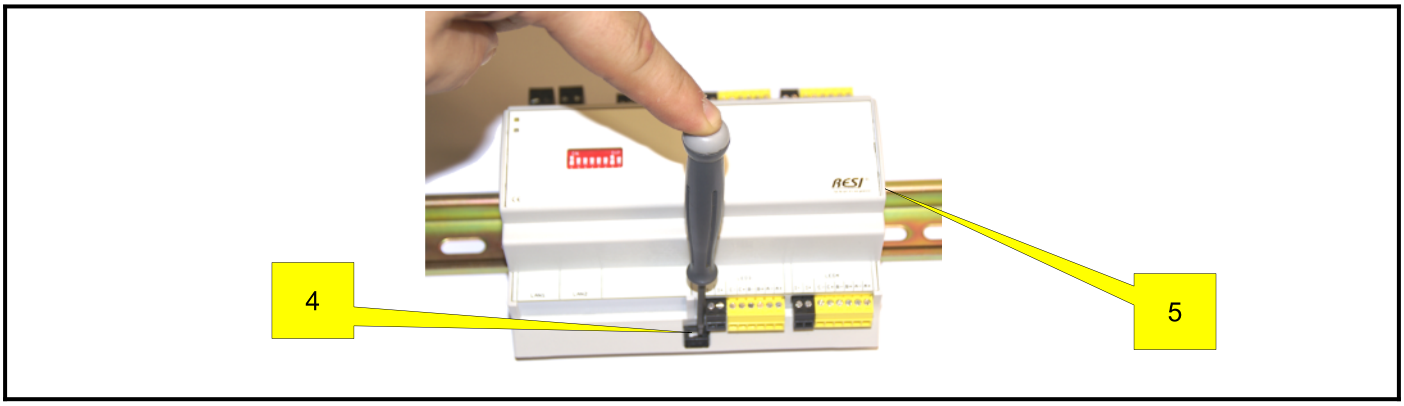
First snap in the top part of the module into the DIN rail (1). The bottom part of the module is not snapped into the DIN rail at this moment.



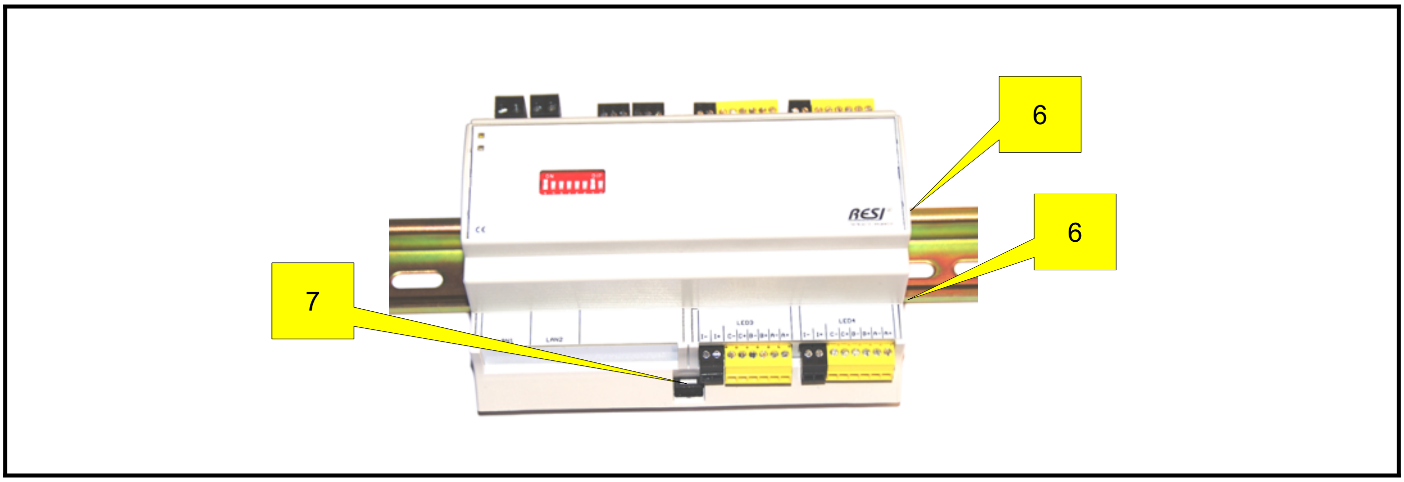
Then open the black hook with a screw driver (2). Now press the module with the opened hook onto the DIN rail until both sides of the module snap into the DIN rail (3). Release the screw driver now. The hook snaps into the DIN rail and the module is now mounted correctly onto the DIN rail.



To remove the module from the DIN rail, you must open the hook with a screwdriver first. (4). Afterwards tilt the bottom side of the module upwards with the open hook (5). Now remove the module slightly from the DIN rail with the top side, to completely hang out the module from the DIN rail.

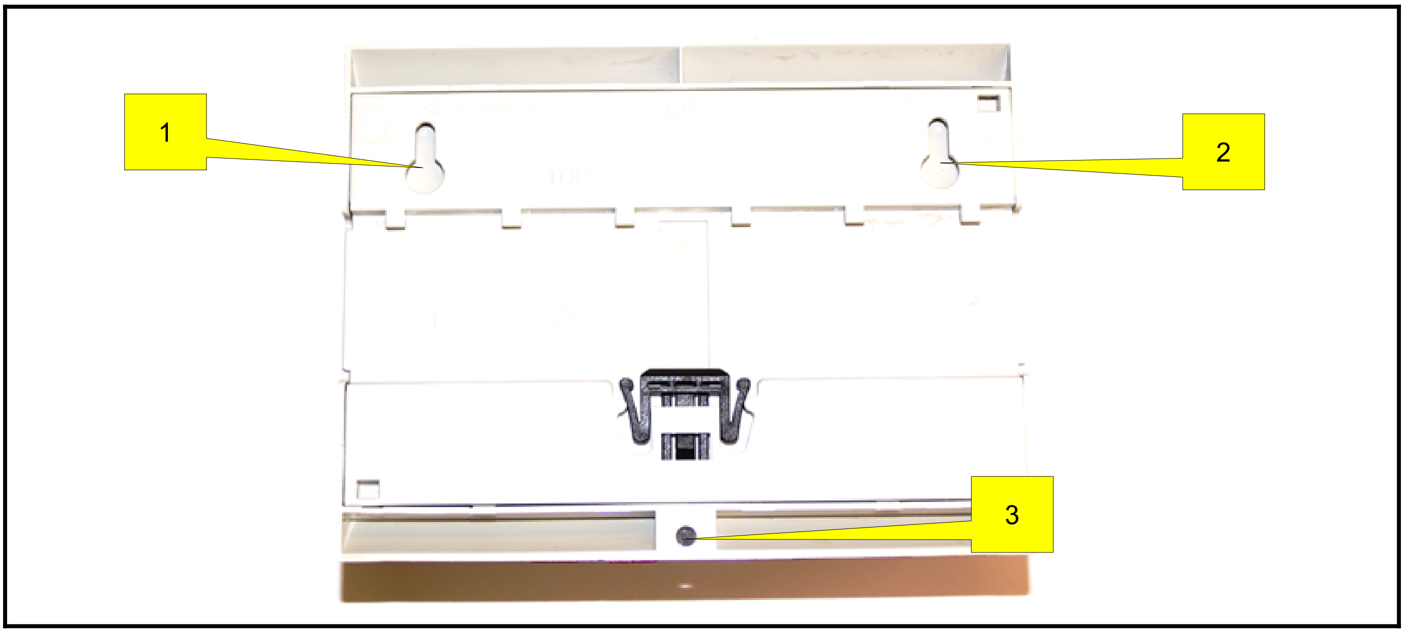


The module is correctly mounted, if the module has snapped into the DIN rail on both sides of the housing (6) and if the hook has snapped in too (7).

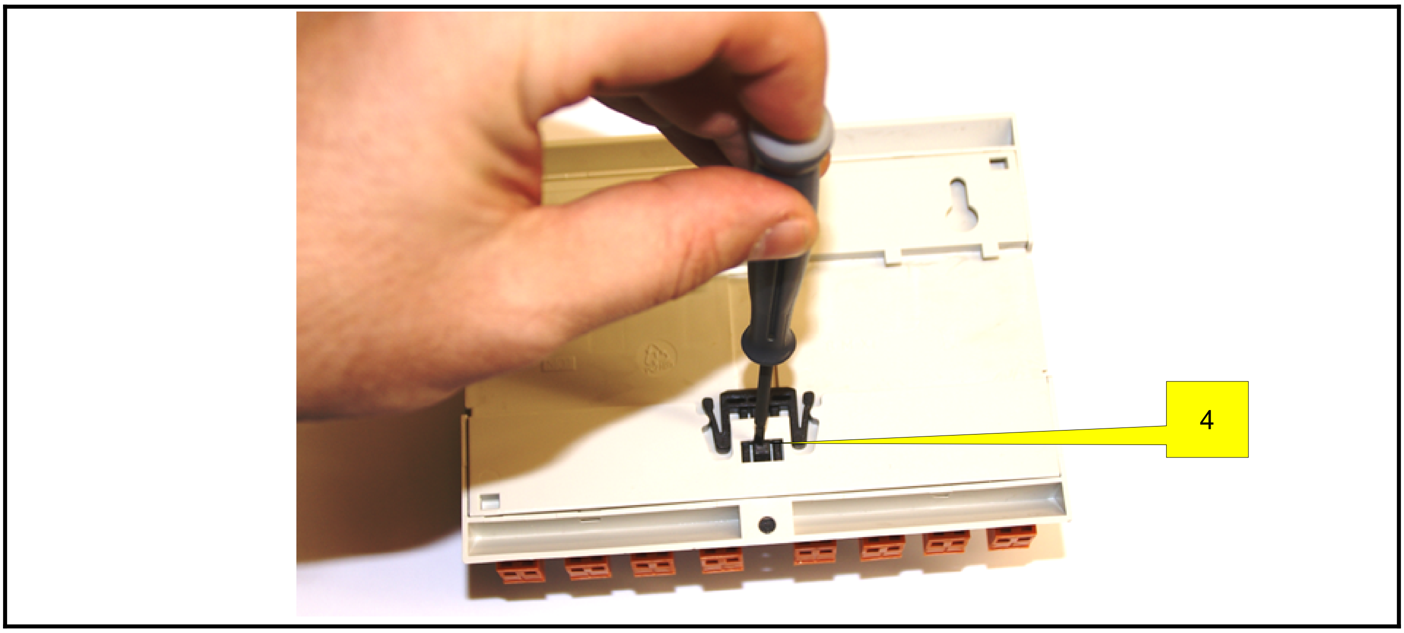


4.2.2 Mounting onto a wall

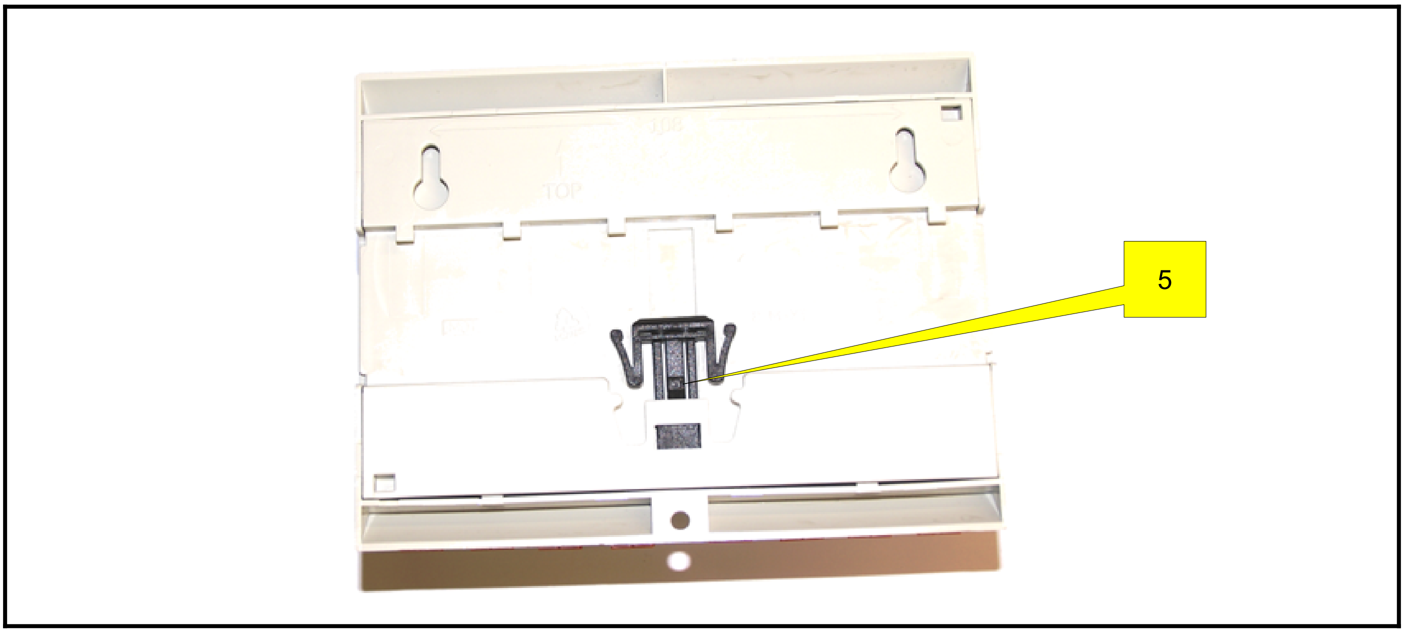
Our modules can also be mounted onto a wall. Turn over the module as shown in the picture below:

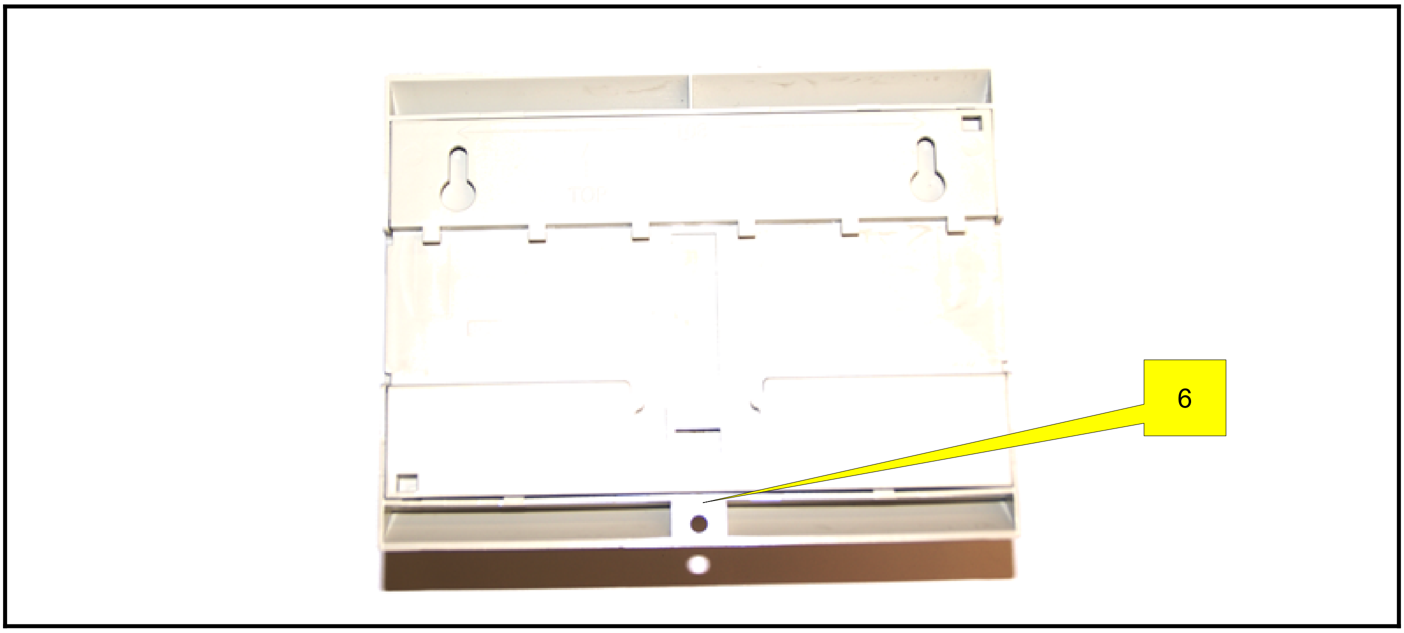


You will notice, that there are two holes for wall hooks or screws on the top side of the housing. (1) and (2). On the bottom side you will notice a small hole for a screw to fix the housing on the wall from the front (3). But first we have to remove the hook, which blocks the screw hole in the housing.



Press carefully the screwdriver onto the hook to open the lock (4) and pull back the hook to the inner side of the housing bottom to remove the hook. If the hook is not snapped into the housing, you can remove the hook by hand (5) and the screw hole for fixing the housing with a screen from the front side of the housing (6).





Now fix two wall hooks or screws into the wall. Use a center to center distance of 108mm between those two screws or hooks. The screw head must be bigger than 4mm but also smaller than 8mm to fix the housing onto the wall like a picture frame. If the housing in mounted onto the wall, you can fix the housing with a secure screw through the hole in the bottom housing from the front. But your screw must be smaller than 4mm to fit into this hole and the screw head must be bigger than 4mm to press the housing onto the wall.

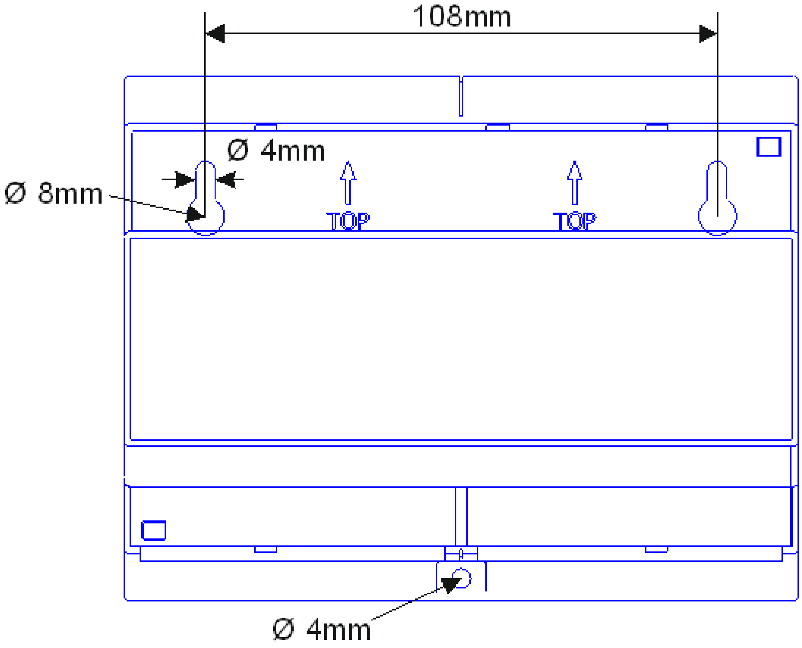


Illustration: Bottom view of the module with holes for XT8 wall mounting

5 General technical data

In this section you will find all technical data which is common to all IO modules. In the specific sections of the individual IO modules you will find only the differences and extensions to this standard description.

5.1 Basic technical data

Power supply	
Supply voltage	12-48 V = +/- 10%
Voltage LED indicator	Yes
Power consumption	see individual technical data for specific IO module
Serial interface	
(only for serial ULTRA SLIM IOs and BIG IOs)	
Protocols	MODBUS/RTU slave or ASCII text protocol
Type	RS232 or RS485 for ULTRA SLIM IOs
	RS485 for BIG IOs
Baud rate	300 to 256000bd
Data bits	8 bits
Parity	none, even or odd
Stop bits	1 or 2 bits
LED indicator	Yes
Ethernet interface	
(only for Ethernet ULTRA SLIM IOs)	
Protocols	MODBUS/TCP Server
	ASCII Text socket
	MODBUS/RTU over Ethernet
Type	Ethernet
Cable connection	via RJ 45 socket
LED indicator	Yes
General	
Storage temperature	-20...85 °C
Operating temperature	0...55 °C
Humidity	25...90% r.H. non-condensing
Protection class	IP20 (EN 60529)
Dimensions LxWxH	see section Dimension
Weight	see individual technical data for specific IO module
Installation	on DIN EN50022 rail for ULTRA SLIM IOs
	on DIN EN50022 rail and on wall for BIG IOs
Approvals	
CE conformity	Yes

5.2 Serial ULTRA SLIM IOs: basic terminals

The serial ULTRA SLIM IOs come in a housing with integrated clamps. All IO modules offer the following terminals:

L+, M-	Power supply:	
	L +:	12-48 V =
	M-:	mass
A, B, M-	RS485 ASCII or MODBUS/RTU interface	
	A +:	RS485 DATA + signal
	B-:	RS485 DATA signal
	M-:	RS485 ground signal
TX, RX, M-	RS232 ASCII or MODBUS/RTU interface	
	TX +:	RS232 transmit signal
	RX-:	RS232 receive Signal
	M-:	RS232 ground signal
Terminal type USLIM	Cable cross section:	max. 2.5 mm², max. 14AWG
	Screw:	M3
	Tightening torque:	max. 0.5Nm, max. 4.5 Lb-in

5.3 Ethernet ULTRA SLIM IOs: basic terminals

The Ethernet ULTRA SLIM IOs come in a housing with integrated clamps. All IO modules offer the following terminals:

L+, M-	Power supply:	
	L +:	12-48 V =
	M-:	mass
ETHERNET	RJ45 connector	
	Ethernet connection	10M/100Mbit adaptive
	supports AUTO–MDIX	
Terminal type USLIM	Cable cross section:	max. 2.5 mm², max. 14AWG
	Screw:	M3
	Tightening torque:	max. 0.5Nm, max. 4.5 Lb-in

5.4 Serial BIG IOs: basic terminals

The serial BIG IOs come in a housing with removable clamps. All IO modules offer the following terminals:

L+, M-	Power supply via two separated plug-in 2-pin terminal blocks.	
	For daisy chain IN and OUT power supply of many modules	
	Pin 1:	L+: 12-48 V=
	Pin 2:	M-: Ground
	Terminal type:	RM5
SIO1	RS485 ASCII or MODBUS/RTU serial interface IN	
	Pin 1:	A+: RS485 DATA+ signal
	Pin 2:	B-: RS485 DATA- signal
	Pin 3:	GND: RS485 ground signal
	Terminal type:	RM3.5
SIO2	RS485 ASCII or MODBUS/RTU serial interface OUT	
	Pin 1:	A+: RS485 DATA+ signal
	Pin 2:	B-: RS485 DATA- signal
	Pin 3:	GND: RS485 ground signal
	Terminal type:	RM3.5
Terminal type RM5	Cable cross section:	max. 2.5 mm², max. 14AWG
	Screw:	M3
	Tightening torque:	max. 0.5Nm, max. 4.43 Lb-in
Terminal type RM3.5	Cable cross section:	max. 1.5 mm², max. 16AWG
	Screw:	M2
	Tightening torque:	max. 0.2Nm, max. 1.77 Lb-in

5.5 Ethernet BIG IOs: basic terminals

The serial BIG IOs come in a housing with removable clamps. All IO modules offer the following terminals:

L+, M-	Power supply via two separated plug-in 2-pin terminal blocks.	
	For daisy chain IN and OUT power supply of many modules	
	Pin 1:	L+: 12-48 V=
	Pin 2:	M-: Ground
	Terminal type:	RM5
ETH	Ethernet interface	
	RJ45	
Terminal type RM5	Cable cross section:	max. 2.5 mm², max. 14AWG
	Screw:	M3
	Tightening torque:	max. 0.5Nm, max. 4.43 Lb-in

6 Power supply

All of our IO modules support 12-48Vdc external power supply (±10%). The power cables should be selected according to the length of the power lines and the number of modules connected. When implementing a network with long cables, the use of thicker wire is more suitable due to the limitation of DC voltage drop. Furthermore, long wires can also cause interference with communication wires. All modules use onboard switching regulators to sustain efficiency over the 12..48Vdc input range. So the actual drawn current can be assumed to be inversely proportional to the DC voltage.

6.1 Power supply for serial ULTRA SLIM IO modules

The following drawings show the correct power supply for all of our serial SLIMIO products:

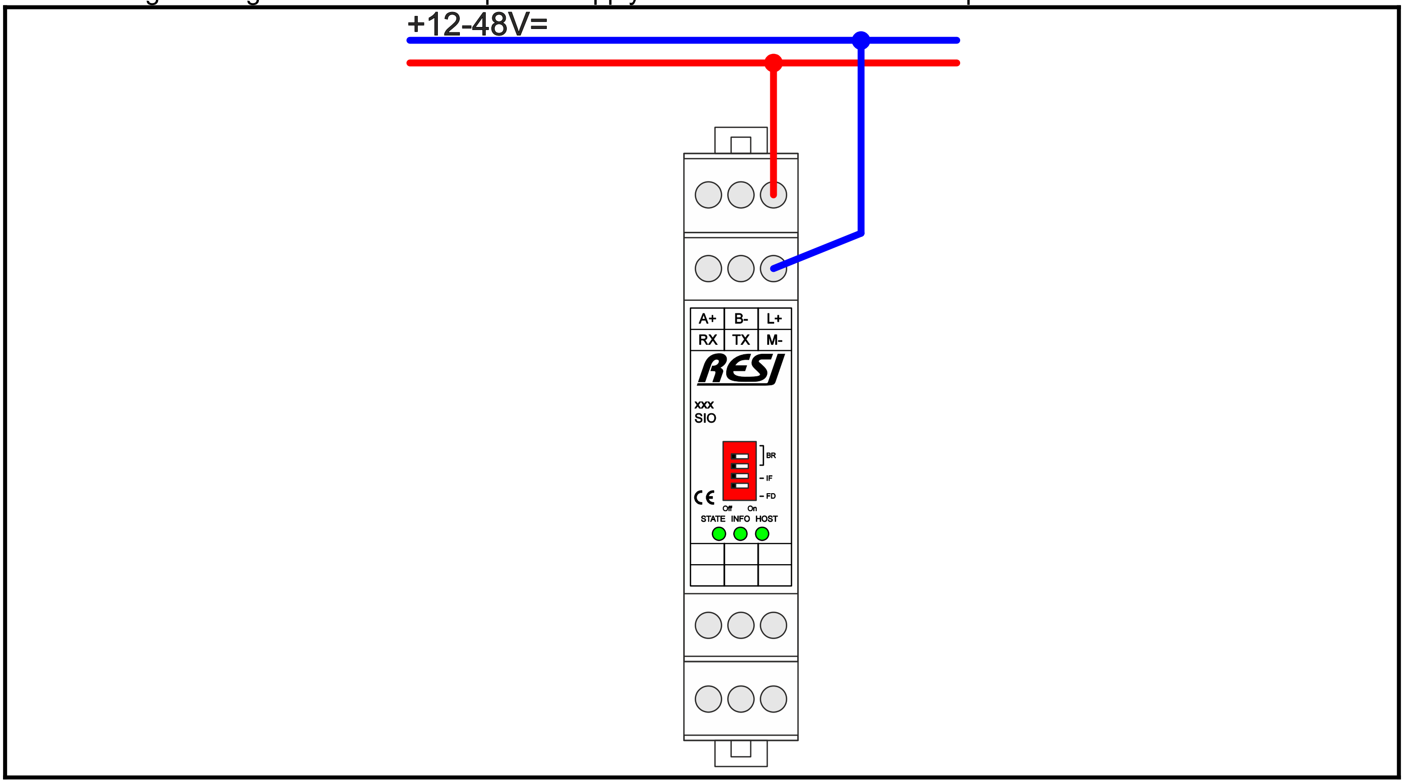


Figure: Power supply for our serial ULTRA SLIM IO modules

6.2 Power supply for Ethernet ULTRA SLIM IO modules

The following drawings show the correct power supply for all of our Ethernet SLIMIO products:

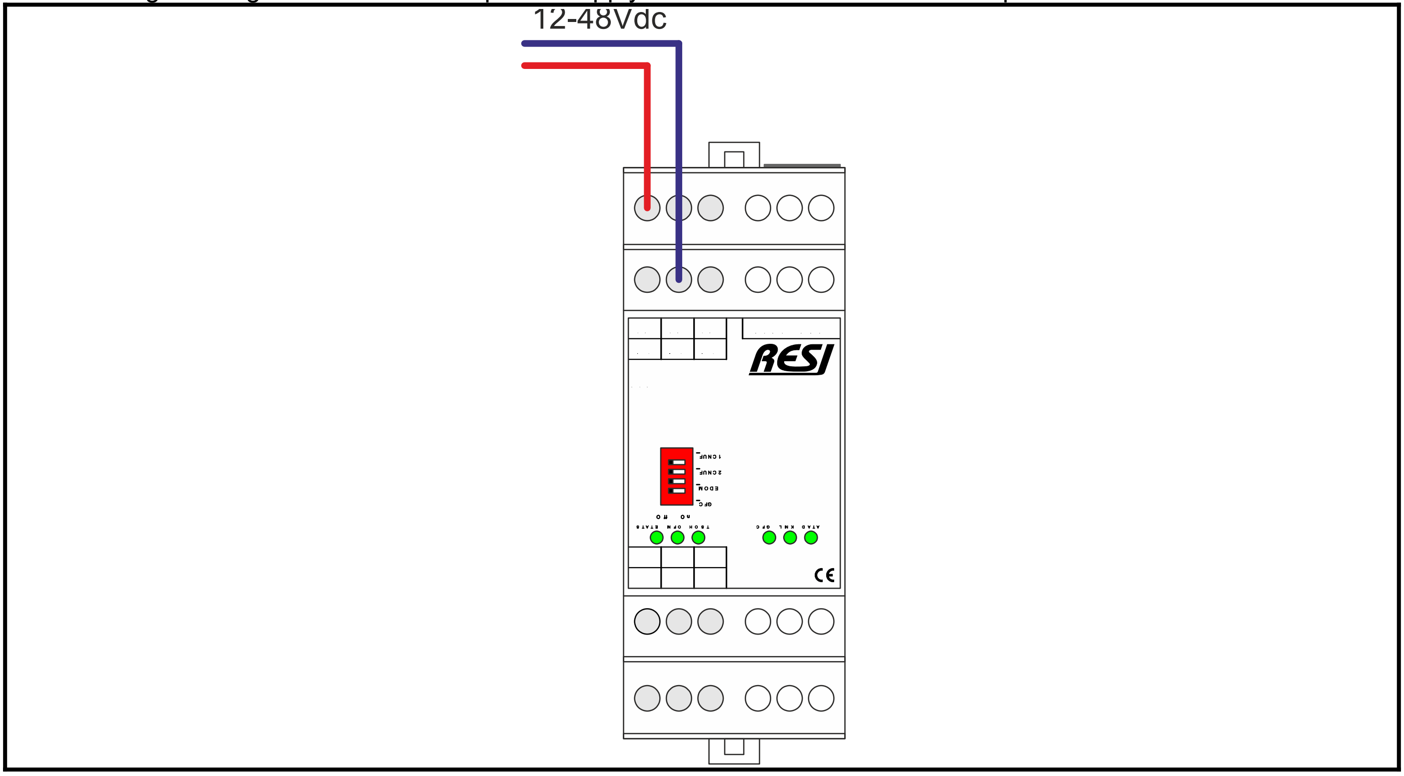


Figure: Power supply for our Ethernet ULTRA SLIM IO modules

6.3 Power supply for BIGIO XT8 modules

The following drawings show the correct power supply for all of our BIGIO XT8 products:

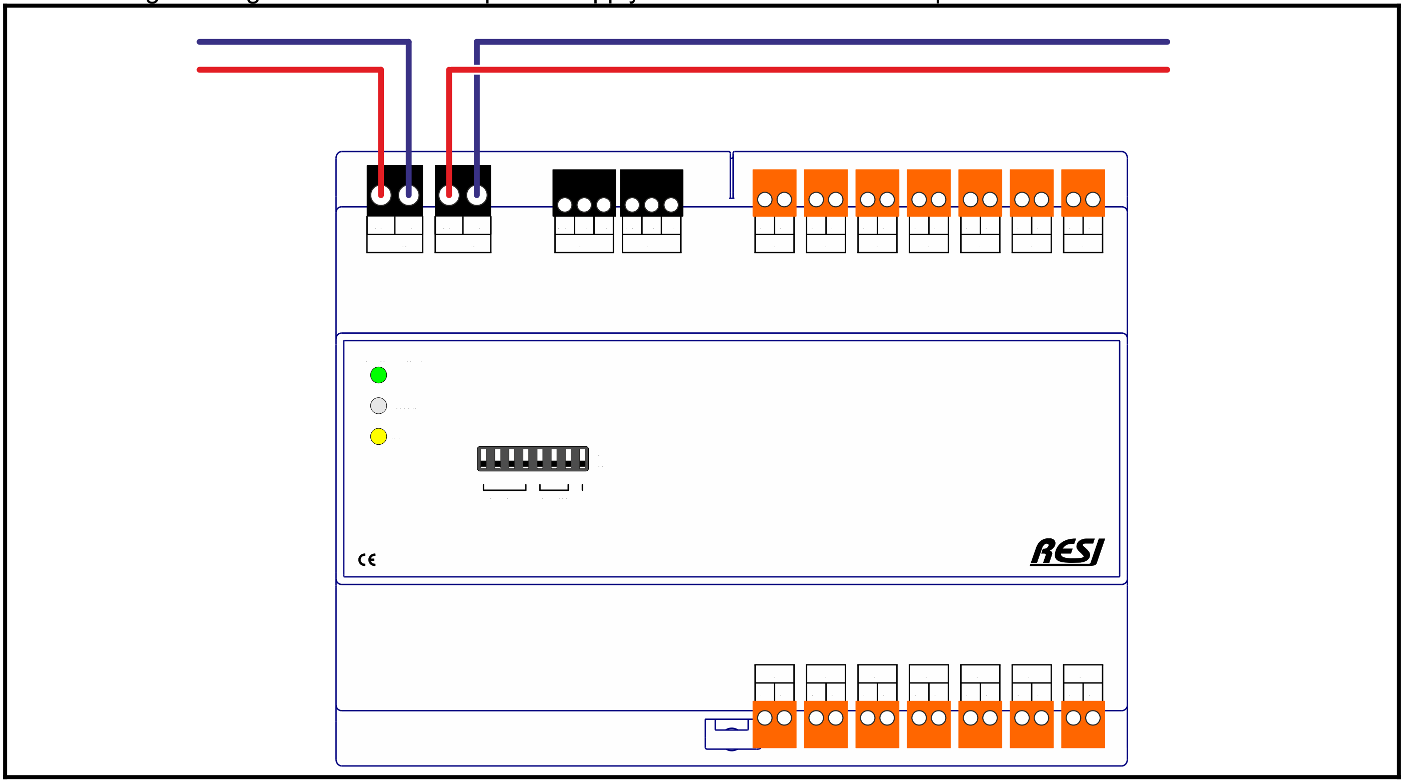


Figure: Power supply for our BIGIO XT8 modules

6.4 Power supply for BIGIO XT12 modules

The following drawings show the correct power supply for all of our BIGIO XT12 products:

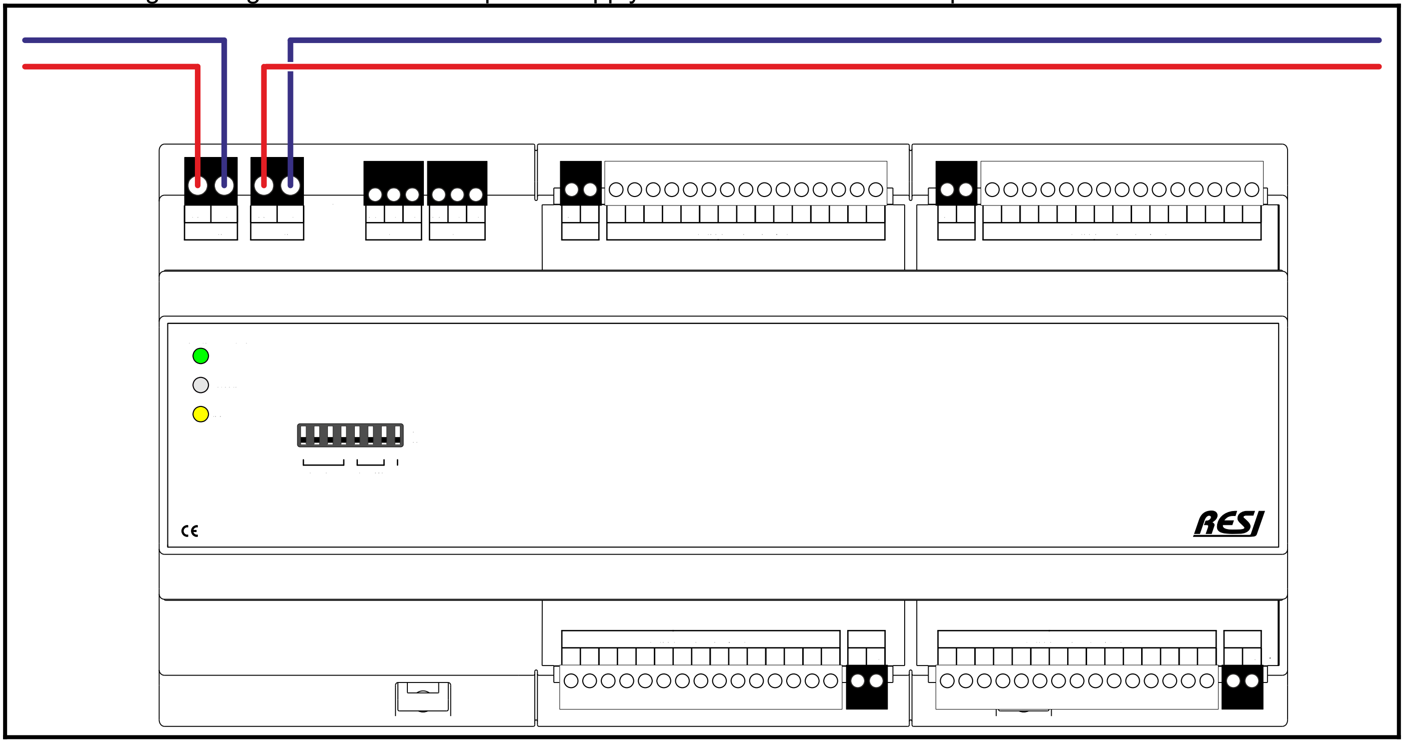


Figure: Power supply for our BIGIO XT12 modules

7 Serial connection

Our serial ULTRASLIM IO modules offer a RS232 or RS485 interface. Our serial BIGIO modules offer only a RS485 interface. The following drawings show the correct connection of the serial bus.

7.1 Serial connection for ULTRA SLIM IO modules

The following drawings show the correct serial connection of the RS232 or the RS485 for all of our serial SLIMIO products:

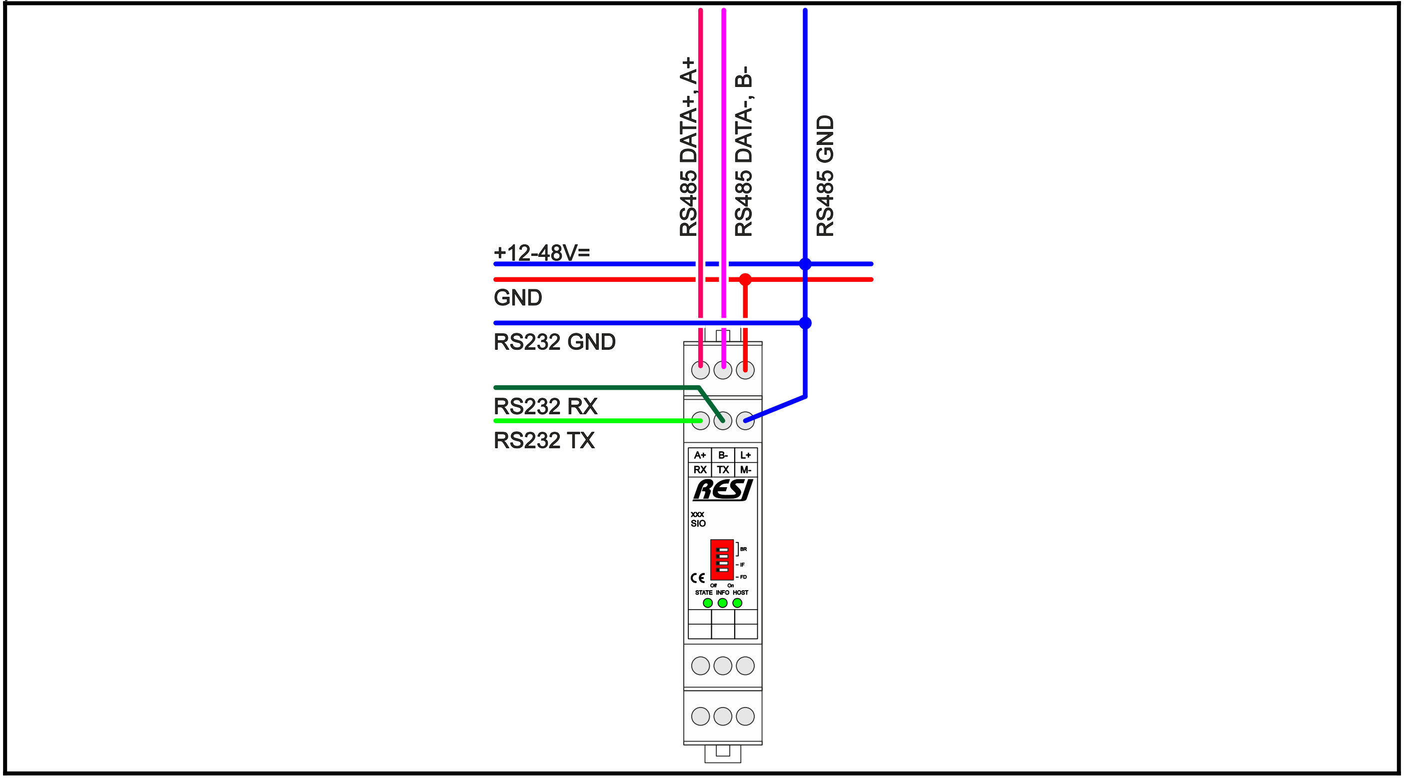


Figure: Serial connection for RS232 or RS485 for our serial ULTRA SLIM IO modules

7.2 Serial connection for BIGIO XT8 modules

The following drawings show the correct serial connection of the RS485 for all of our serial BIGIO products:

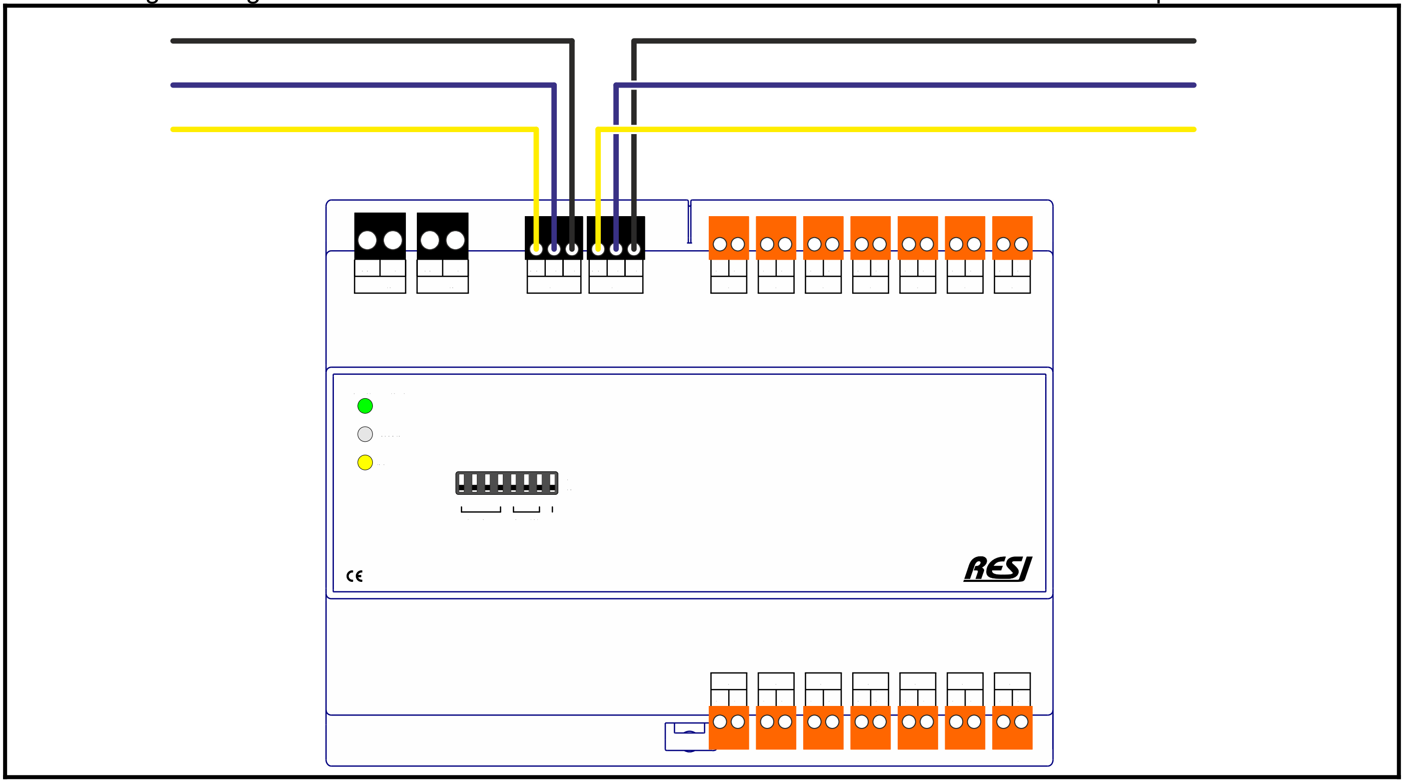


Figure: Serial connection for RS485 for our serial BIG IO modules

7.3 Serial connection for BIGIO XT12 modules

The following drawings show the correct serial connection of the RS485 for all of our serial BIGIO products:

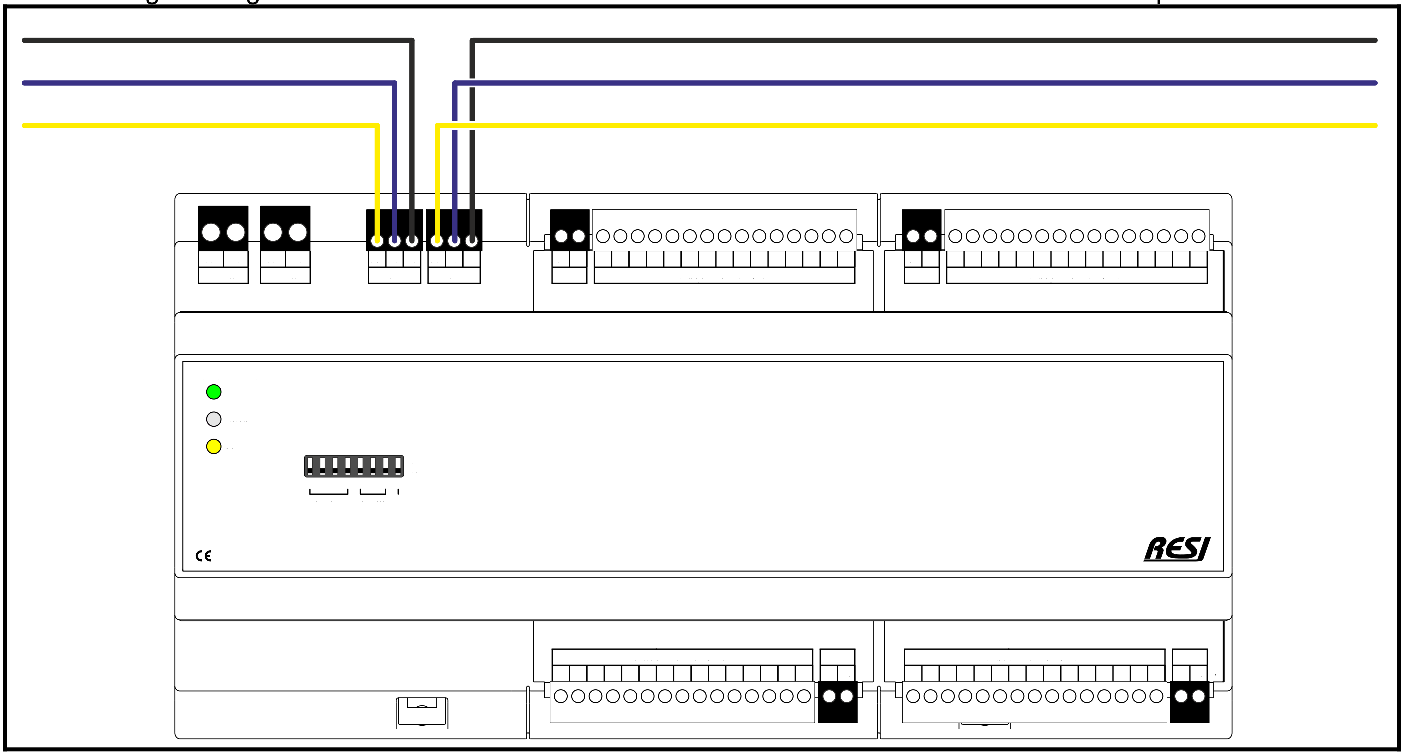


Figure: Serial connection for RS485 for our serial BIG IO modules

7.4 RESI-xxx-SIO SERIAL PROTOCOL

As mentioned our modules support either MODBUS/RTU or a simple ASCII text protocol.

7.4.1 MODBUS/RTU protocol

All of our serial IO modules communicate with a host system with the MODBUS/RTU slave protocol (RS232 or RS485 variants) or with the MODBUS/TCP server protocol (Ethernet version).

For communication via ASCII texts, ASCII messages with a special start character # (0x23, 35dec) and special end characters (0x0d, 13dec or CARRIAGE RETURN) are sent from the host to the module. The module also sends its responses with this special start and end character. See the ASCII command description below. In ASCII mode you can communicate with or without a bus number.

The following MODBUS functions are available for communication via MODBUS/RTU or MODBUS/TCP:

READ COIL STATUS (function code: 1)
READ INPUT STATUS (function code: 2)
READ HOLDING REGISTER (function code: 3)
READ INPUT REGISTER (function code: 4)
FORCE SINGLE COIL (function code: 5)
PRESET SINGLE REGISTER (function code: 6)
FORCE MULTIPLE COILS (function code: 15)
PRESET MULTIPLE REGISTERS (function code: 16)

Note:
The functions READ HOLDING REGISTER and PRESET MULTIPLE REGISTERS are limited to max. 125 registers limited per request! The functions READ INPUT STATUS, READ COIL STATUS and FORCE MULTIPLE COILS are limited to 2000 coils or inputs (bits) per data frame.

7.4.1.1 HOWTO map values to MODBUS registers

MODBUS is an international standard for communication between host systems like PLCs, DDCs or Industrial PCs and peripheral components or sensors.

More details about the MODBUS standard and the MODBUS protocol can be found here:
*http://en.wikipedia.org/wiki/**Modbus***
http://www.modbus.org/

You can find a documentation about this in the internet called “PI_MBUS_300.pdf”, which describes the MODBUS protocol pretty good.

There are three different MODBUS protocol versions available:
MODBUS/TCP: Used for communication with TCP/IP systems
MODBUS/RTU: A binary version of the MODBUS protocol
MODBUS/ASCII: An ASCII text based version of the protocol

To communicate, our RESI-xxx-SIO converters have either a RS232 interface to communicate 1 to 1, which means one MODBUS/RTU master (your host system) can talk to exact one MODBUS/RTU slave, or a RS485 to offer a one to many communication. Here one MODBUS/RTU master can communicate with a maximum of 255 MODBUS/RTU slaves. In older host systems the limit is 32 slaves. This depends on the capabilities of the RS485 driver IC in the host system. Our converters are able to use 256 communication partners on a RS485 line.

Our RESI-xxx-ETH converters can communicate with MODBUS/TCP protocol. A MODBUS/TCP system consists out of one TCP server which is in fact our gateway and at least one to n MODBUS/TCP clients. This will be your host. Our converters can connect only to one TCP client at a time.

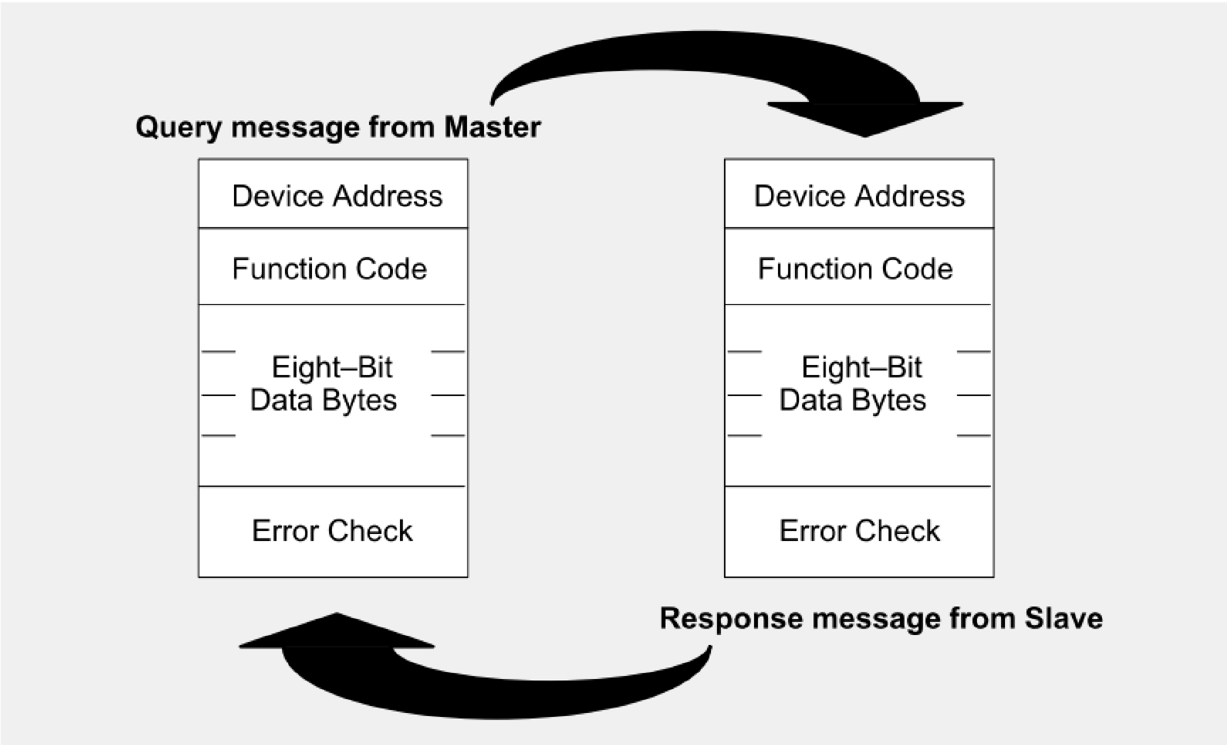
To communicate the converters use an Ethernet interface.

MODBUS unit:
The MODBUS protocol demands a unique address of a MODBUS slave to address this special slave. This address is called MODBUS unit. The range of this address is from 0 to 255. Usually 0 is not used in applications. We use 0 for broadcast functions.

7.4.1.2 MODBUS query response cycle

MODBUS is a master slave protocol. This means, the master (your host system) has to send a protocol to a specific MODBUS slave (one of our converters), then this specific slave answers to the master, and then the master asks the next slave. The address of the slave is the so-called device address or unit address, which we mentioned before. See the below graphic, how a basic MODBUS request and response cycle looks like.

The Query–Response Cycle



7.4.1.3 MODBUS/RTU telegram structure

A MODBUS/RTU protocol frame consists out of the following fields:

START: There is no specific start character, so a pause of four character timings depending on the baud rate of your communication must be established. This means at least for four characters, that there must be no communication on the serial line!

ADDRESS: This is the unit address of the slave, the master wants to talk to. It's a number between 0 and 255.

FUNCTION: This defines the type of data communication, the master wants to handle with the slave. Refer to the next pages for a detailed description of the functions.

DATA: This is a block of individual data bytes.

CRC CHECK: This is the checksum, to let the master and slave check, if the received protocol is correct and without communication errors.

END: Same as the start condition. Again there must not be communicated for at least 4 character times on the serial line.

IMPORTANT HINT: If there is more than one MODBUS slave on a serial line, the pausing of the START and END sequence are essential to re synchronize the slaves in case of data loss. If the host doesn't keep this gaps, communication with the slaves can be corrupted or impossible!

START	ADDRESS	FUNCTION	DATA	CRC CHECK	END
T1–T2–T3–T4	8 BITS	8 BITS	*n* x 8 BITS	16 BITS	T1–T2–T3–T4

7.4.1.4 MODBUS commands

The MODBUS standard defines many available commands . But not all systems handle the complete spectrum of telegrams. Our converter handles only all telegrams necessary for using holding and INPUT registers.

We support
03 READ HOLDING REGISTER
04 READ INPUT REGISTER
06 PRESET SINGLE REGISTER
16 PRESET MULTIPLE REGISTER

IMPORTANT HINT: All other protocols are ignored by our converters.

So what are HOLDING REGISTERs ?
According to the MODBUS standard, a MODBUS/RTU slave can hold up to 65535 HOLDING registers. Each holding register is a 16 bit register, capable for integer values between 0 and 65535 or in hexadecimal from 0x0000 to 0xFFFF. A MODBUS/RTU master system can read and write the contents of those registers.

IMPORTANT HINT:
A MODBUS/RTU master can read and write into this registers with a 16 bit index, called the
starting address. The problem is the definition of the starting address. A 16 bit value can store the values from 0 to 65535. But according the MODBUS standard the registers are numbered from 1 to 65536. So, if the MODBUS standard talks about register 1, an index of 0 must be used as start address in the telegram. You have to check carefully, how this index is interpreted by the manufacturers' documentation.

Code	Name
01	Read Coil Status
02	Read Input Status
03	Read Holding Registers
04	Read Input Registers
05	Force Single Coil
06	Preset Single Register
07	Read Exception Status
08	Diagnostics
09	Program 484
10	Poll 484
11	Fetch Comm. Event Ctr.
12	Fetch Comm. Event Log
13	Program Controller
14	Poll Controller
15	Force Multiple Coils
16	Preset Multiple Registers
17	Report Slave ID
18	Program 884/M84
19	Reset Comm. Link
20	Read General Reference
21	Write General Reference

Whenever you get a description of registers for a MODBUS device, the first question to solve is: How is the enumeration of the registers done?! Does the author use base=0, then he talks about the real start index of the telegram. Does the author mean base=1, conforming to naming conventions of the MODBUS consortium, then you have to subtract 1 before using this address in your telegrams.

IMPORTANT HINT:
If we display a holding register address like 4x00009 in our tool, we assume base=1 conforming to the standard. So your host system has to send the start index 00008 decimal to read out the correct register.

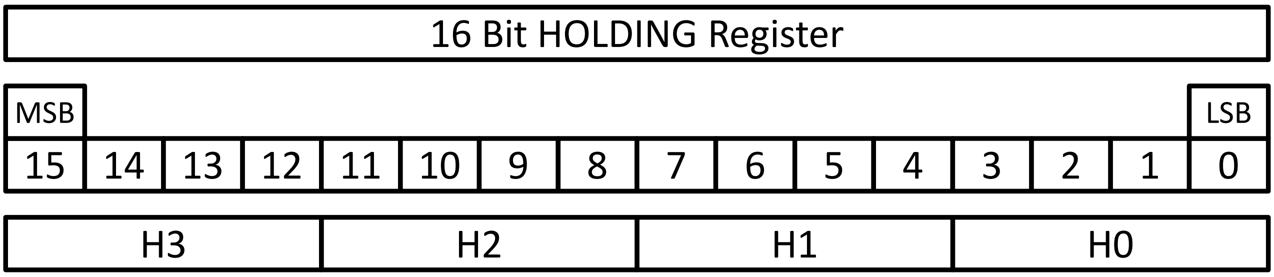
Start Index (Base=0)	MODBUS Register (Base=1)	Description
0	1	The first holding register
1	2	The second holding register
2	3	The third holding register
…	…	
65534	65535	The penultimate holding register
65535	65536	The last holding register

7.4.1.5 MODBUS 16 bit holding register structure

Here we give a brief introduction, how to build the contents of a MODBUS holding register, and how a hexadecimal writing of a 16 bit register looks like. We assume, that the user is familiar to hexadecimal and binary number systems and also how a computer stores data into its internal memory.

For more details consult the internet:
http://en.wikipedia.org/wiki/Hexadecimal
http://en.wikipedia.org/wiki/Binary_number

Usually a hexadecimal digit describes 4 bits. So we can group the 16 bits into 4 hexadecimal digits named H3,H2,H1,H0. This means eg. the hexadecimal number 0xABCD stands for H3=A, H2=B, H1=C, H0=D.



0xA=1010 binary, 10 dec, 0xB=1011,11 dec, 0xC=1100,12 dec and 0xD=1101, 13 dec. So the resulting binary number is 1010101111001101b or 43981 decimal.

See this graphical explanation, how the number is stored:

MSB															LSB
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	0	1	0	1	0	1	1	1	1	0	0	1	1	0	1

A	B	C	D

7.4.1.6 MODBUS big vs. least significant byte order

Now the first problem for a host system arises:
If we take the 16 bit number 0xABCD, we have to use 2 bytes to store this value internally. There are two concurrent versions of how to store this value in the RAM:

INTEL byte order, Little endian systems store the least significant byte first. So a memory map for 0xABCD look like:

Memory address 0	CD
Memory address 1	AB

MOTOROLA byte order, Big endian systems store the most significant byte first. So a memory map for 0xABCD look like:

Memory address 0	AB
Memory address 1	CD

Consult the internet for more details about this storage system.

http://en.wikipedia.org/wiki/Endianness

7.4.1.7 MODBUS storing large data into 16 bit registers

After years, the market found out, that the capabilities of storing only 16 bit numbers into one holding register is not enough for many applications. The most common solution to store more than 16 bit values into holding registers is to use more than one register to hold the value. For storing e.g. a 32 bit value, we use two consecutive 16 bit holding registers, for storing a 32 bit float value we also use also two consecutive 16 bit registers!

We want to store the 32 bit integer value 0x12345678 into two consecutive holding registers starting at 4x00020. The memory map of the holding registers look like:

		16 bit value
Start Index 19	Holding Register 4x00020	0x1234
Start Index 20	Holding Register 4x00021	0x5678

But again, we can also store the reverse word order into two consecutive registers. Then the result looks like this:

		16 bit value
Start Index 19	Holding Register 4x00020	0x5678
Start Index 20	Holding Register 4x00021	0x1234

So none of the above mentioned orders is better than the other. It depends only on the programmer, how the 32 bit value is treated.

Be aware, that both systems (host and converter) have to treat the 32 bit value in the same way. Otherwise you will read out wrong data! We will discuss this issue later in combination with 32 bit float numbers.

Our converter uses the second described way to store 32 bit values. We follow the little endian strategy of INTEL systems and store 0x5678 into the first HOLDING register, and then we store 0x1234 in the consecutive register.

7.4.1.8 MODBUS datatypes in our converters

Our converter supports the following data types for storing values into MODBUS registers.

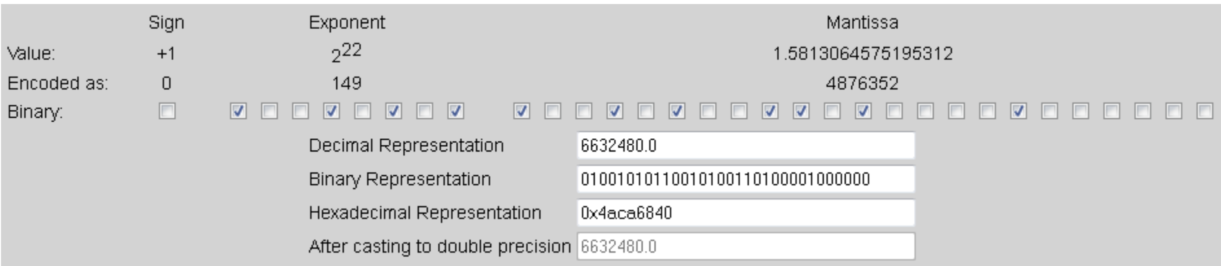
16 bit signed binary: This is an integer number between -32767..0..+32768 or 0x0000 to 0xFFFF hex. This number needs exactly one HOLDING register.

32 bit singed binary: This is an integer number between -2,147,483,647..0..+2,147,483,648 or 0x00000000 to 0xFFFFFFFF hex. This number needs two consecutive holding registers. We store the least significant word first. The serial number 2544082 is in hex 0x26D1D2. This leads to the following HOLDING register layout:

		16 bit value
Start Index 0	Holding Register 4x00001	0xD1D2 or 53714 dec
Start Index 1	Holding Register 4x00002	0x0026 or 38 dec

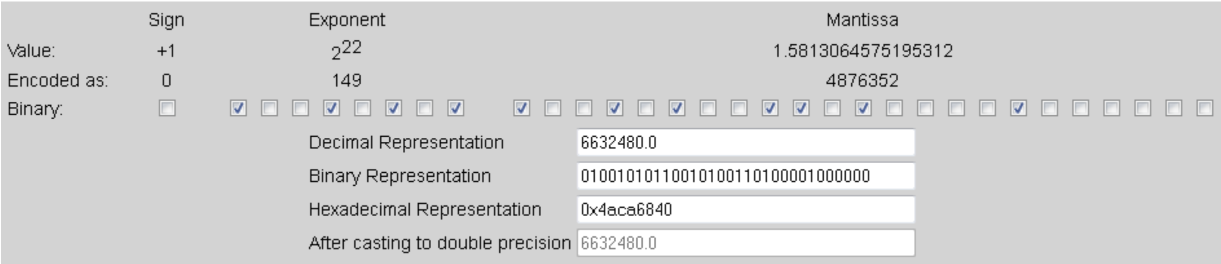
32 bit IEEE floating point: This is a float number using 32 bit. As before, this float needs two consecutive holding registers. We store the least significant word first. The energy value of 6632480,00 is defined in 32 bit hex with 0x4ACA6840. This leads to the following HOLDING register layout. For more details search in the internet or consult http://en.wikipedia.org/wiki/IEEE_floating_point or try out some float values and their hexadecimal representation under http://www.h-schmidt.net/FloatConverter/IEEE754.html

		16 bit value
Start Index 0	Holding Register 4x00001	0x6840
Start Index 1	Holding Register 4x00002	0x4ACA



32 bit IEEE floating point inverse: This is a float number using 32 bit. Again this float needs two consecutive holding registers. We store the least significant word first. The energy value of 6632480,00 is in 32 bit hex 0x4ACA6840. This means the following HOLDING register layout. For more details search in the internet or consult http://en.wikipedia.org/wiki/IEEE_floating_point or try out some float values and their hexadecimal representation under http://www.h-schmidt.net/FloatConverter/IEEE754.html

		16 bit value
Start Index 0	Holding Register 4x00001	0x4ACA
Start Index 1	Holding Register 4x00002	0x6840



IMPORTANT HINT:
32 bit floats are very tricky! Eg. The value 3,5351799 is represented internally as 0x40624063. But the reverse word order (if the host reads out the wrong register indexes or the host corrupts the word order) 0x40634062 leads to the float number 3,5508046. So this error in your software is very hard to find! Be very cautious, which register indexes you read and how the word order of the two registers are interpreted.

32 bit date&time: This is a compressed format using 32 bit. Again the least significant word is stored into the first register. The structure of the 32 bits are:
Bits 0..7: minute
Bits 8..15: hour
Bits 16..20: day
Bits 21..24: month
Bits 25..31: year
The current date & time “07.04.00 01:13” is represented hexadecimal with 0x0087010d (8847628dec) and stored as followed:

		16 bit value
Start Index 0	Holding Register 4x00001	0x010D
Start Index 1	Holding Register 4x00002	0x0087

7.4.1.9 MODBUS datatype storage and common pitfalls

In general MODBUS uses 16 bit wide registers. So if you use only datatypes, which needs also only one register, the mapping is easy. But as soon as you use datatypes, e.g. UINT32, which need two or more MODBUS registers, you can map the values in different ways.

We do a simple sample. We want to store the 32 bit unsigned integer value in hexadecimal 0x12345678 in MODBUS holding registers starting with index 4x00010. The mapping can be done in two different ways:

MODBUS Register	Storage of UINT32 datatype
4x00010 I:9	The high word of the 32 bit value 0x12345678 is stored in the first 16 bit wide MODBUS register. This means the value 0x1234 is stored here.
4x00011 I:10	The low word of the 32 bit value 0x12345678 is stored in the second 16 bit wide MODBUS register. This means the value 0x5678 is stored here.

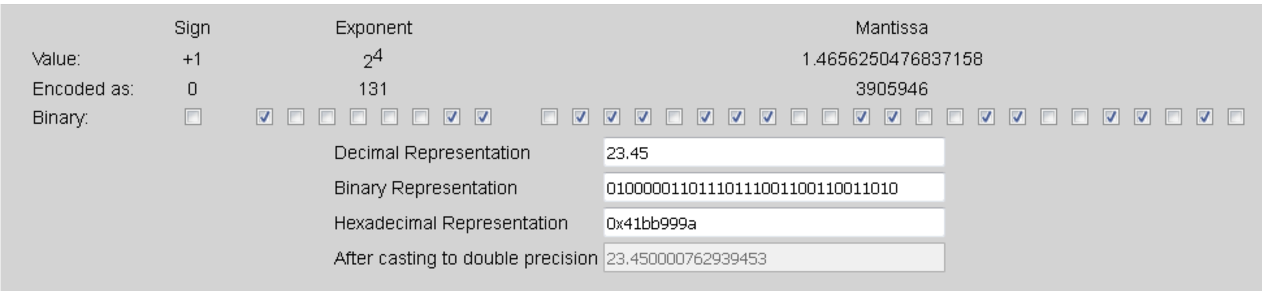
But it is only one possibility, that we store the high word in the first MODBUS register. With the same right, we can define to store the low word in the first register, and the high word in the second.

The result will look like this:

MODBUS Register	Storage of UINT32R datatype
4x00010 I:9	The low word of the 32 bit value 0x12345678 is stored in the first 16 bit wide MODBUS register. This means the value 0x5678 is stored here.
4x00011 I:10	The high word of the 32 bit value 0x12345678 is stored in the second 16 bit wide MODBUS register. This means the value 0x1234 is stored here.

More complicated is the storage of a FLOAT32 value into two consecutive holding registers. We use a standard room temperature e.g. 23,45 °C as a value, we want to store it into two registers.

First we have to translate this value into a valid IEE754 float value. Therefore we use a perfect site in the internet (http://www.h-schmidt.net/FloatConverter/IEEE754.html):



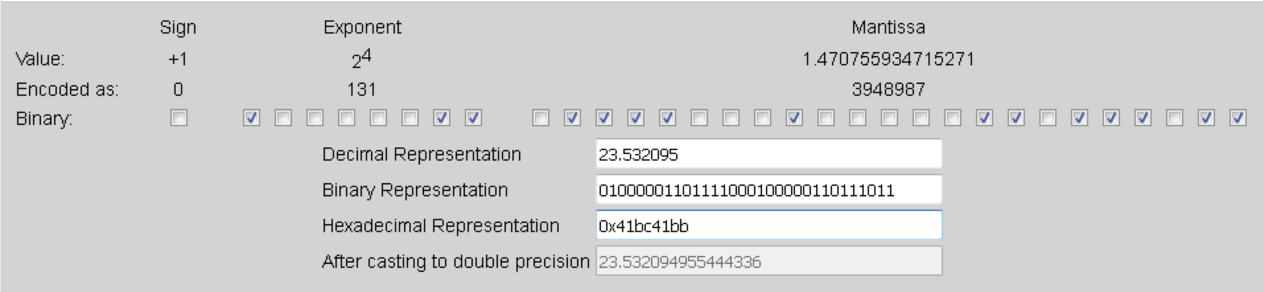
We enter the value 23.45 and we get a 32 bit hexadecimal representation of the float value. It is the number 0x41BB999A. Now we store this value in the same way, we have stored the UINT32 value into two registers:

MODBUS Register	Storage of FLOAT32 datatype
4x00010 I:9	The high word of the 32 bit float value 0x41BB999A is stored in the first 16 bit wide MODBUS register. This means the value 0x41BB is stored here.
4x00011 I:10	The low word of the 32 bit float value 0x41BB999A is stored in the second 16 bit wide MODBUS register. This means the value 0x999A is stored here.

But we can also use the reverse notation:

MODBUS Register	Storage of FLOAT32R datatype
4x00010 I:9	The low word of the 32 bit float value 0x41BB999A is stored in the first 16 bit wide MODBUS register. This means the value 0x999A is stored here.
4x00011 I:10	The high word of the 32 bit float value 0x41BB999A is stored in the second 16 bit wide MODBUS register. This means the value 0x41BB is stored here.

Now we show a common pitfall in writing and reading more than one MODBUS register and rebuilding a value. We use a different float value. In hexadecimal it is 0x41BC41BB. Again we use the online converter:

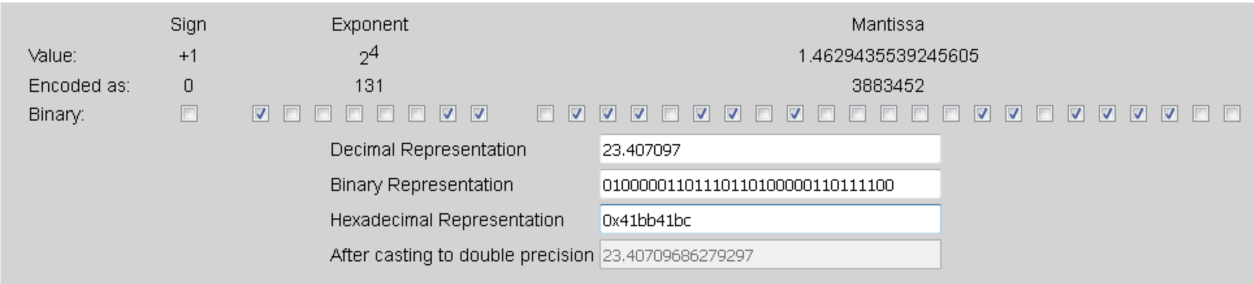


You notice, the float value is 23.532095.

Now we store it with HIGH word first into two registers:

MODBUS Register	**Storage of FLOAT32 datatype**
4x00010 I:9 HIGH WORD	The high word of the 32 bit float value 0x41BC41BB is stored in the first 16 bit wide MODBUS register. This means the value 0x41BC is stored here.
4x00011 I:10 LOW WORD	The low word of the 32 bit float value 0x41BC41BB is stored in the second 16 bit wide MODBUS register. This means the value 0x41BB is stored here.

But now we make a very big mistake, we read the two registers and restore the hexadecimal value in our host software in the reverse word order. First low word, then high word. The result is the 32 bit value 0x41BB41BC instead the correct value 0x41BC41BB. Then we convert this into an IEE754 float value.



The result is 23.407097. This is not far away from the original number of 23.532095! So this massive software error can be undiscovered for a long time. Only if the reverse float value generates numbers which are physically not possible for the measured signal, this error is discovered!

7.4.1.10 MODBUS data type table

The following table shows, how more complex data types are stored in successive 16 bit holding or input registers within the MODBUS registers:

MODBUS DATATYPE	SIZE	WORD ORDER	DESCRIPTION
UINT16	16 bits 1 register	none	Defines a 16 bit unsigned integer value in the range of 0 to 65535 or 0x0000 to 0xFFFF
SINT16	16 bits 1 register	none	Defines a 16 bit signed integer value in the range of -32768 to +32767 or 0x8000 to 0x7FFF
UINT32	32 bits 2 register	0:High Word 1:Low Word	Defines a 32 bit unsigned integer value in the range of 0 to 4.294.967.295 or 0x00000000 to 0xFFFFFFFF
SINT32	32 bits 2 register	0:High Word 1:Low Word	Defines a 32 bit signed integer value in the range of −2.147.483.648 to +2.147.483.647or 0x80000000 to 0x7FFFFFFF
UINT32R	32 bits 2 register	0:Low Word 1:High Word	Defines a 32 bit unsigned integer value in the range of 0 to 4.294.967.295 or 0x00000000 to 0xFFFFFFFF with reverse word order
SINT32R	32 bits 2 register	0:Low Word 1:High Word	Defines a 32 bit signed integer value in the range of −2.147.483.648 to +2.147.483.647or 0x80000000 to 0x7FFFFFFF with reverse word order
FLOAT32	32 bits 2 register	0:High Word 1:Low Word	Defines a 32 bit float value in the range of $\pm 1.4 \cdot 10^{-45}$ to $\pm 3.403 \cdot 10^{38}$. A mantissa of 23 bits and an exponent of 8 bits are used. The value can store 7 to 8 digits after the comma.
FLOAT32R	32 bits 2 register	0:Low Word 1:High Word	Defines a 32 bit float value in the range of $\pm 1.4 \cdot 10^{-45}$ to $\pm 3.403 \cdot 10^{38}$. A mantissa of 23 bits and an exponent of 8 bits are used. The value can store 7 to 8 digits after the comma. The two 16 bit words are stored in reverse order.
DOUBLE64	64 bits 4 register	0:Highest Word 1:Higher Word 2:Lower Word 3:Lowest Word	Defines a 64 bit float value in the range of $\pm 4.24 \cdot 10^{-324}$ to $\pm 1{,}798 \cdot 10^{308}$. A mantissa of 52 bits and an exponent of 11 bits are used. The value can store 15 to 16 digits after the comma.
DOUBLE64R	64 bits 4 register	0:Lowest Word 1:Lower Word 2:Higher Word 3:Highest Word	Defines a 64 bit float value in the range of $\pm 4.24 \cdot 10^{-324}$ to $\pm 1{,}798 \cdot 10^{308}$. A mantissa of 52 bits and an exponent of 11 bits are used. The value can store 15 to 16 digits after the comma. The four 16 bit words are stored in reverse order.

7.4.1.11 MODBUS table

COILS (1x) & INPUTS (2x)

The module holds internally a list of 1 bit coil and input register. Those registers can be read by the host with the function READ COIL STATUS (function code: 1). If the register can also be modified by the host, the host can use the functions FORCE SINGLE COIL (function code: 5) and FORCE MULTIPLE COILS (function code: 15).

In addition the SAME registers are also readable over the function READ INPUT STATUS (function code: 2). This is for host systems, which do not support all MODBUS/RTU functions properly.

The MODBUS convention defines 65535 possible coils with the notation 1x00001 to 1x65536. Inputs are usually noted with 2x00001 to 2x65536. Please refer the software MODBUS POLL as a sample for this notation. Internally in the MODBUS/RTU frames an index notation is used, which starts with 0 and ends with 65535. So we decided to note in the following document a register with: 1x00100 for the coil 100, 2x00100 as a hint, that you can read this register also as the input 100, and in addition also the real index of the protocol index 99 with the notation I:99.

HOLDING REGISTER (3x) & INPUT REGISTER (4x)

The module holds internally a list of 16 bit wide holding register. Those registers can be read by the host with the function READ HOLDING REGISTER (function code: 3). If the register can also be modified by the host, the host can use the functions PRESET SINGLE REGISTER (function code: 6) and PRESET MULTIPLE REGISTERS (function code: 16).

In addition the SAME holding registers are also readable over the function READ INPUT REGISTER (function code: 4). This is for host systems, which do not support all MODBUS/RTU functions properly.

The MODBUS convention defines 65535 possible holding register with the notation 4x00001 to 4x65536. Input register are usually noted with 3x00001 to 3x65536. Please refer the software MODBUS POLL as a sample for this notation. Internally in the MODBUS/RTU frames an index notation is used, which starts with 0 and ends with 65535. So we decided to note in the following document a register with: 4x00100 for the holding register 100, 3x00100 as a hint, that you can read this register also as the input register 100, and in addition also the real index of the protocol index 99 with the notation I:99.

SOFTWARE RESET						
RESET	1x06001 2x06001 I:6000	0,0x00 B:00		N/A:NO CHANGE	BIT R/W	NO
Performs a software reset, whenever 1 is written to this register. If the host writes to this register 1, the module executes a soft reset (reboot).						
RESET	3x06001 4x06001 I:6000	0,0x0000 B:00 00		N/A:NO CHANGE	UINT16 R/W	NO
Performs a software reset, whenever 1 is written to this register. If the host writes to this register 1, the module executes a soft reset (reboot).						
CONVERTER STATUS						
CONVERTER STATUS	3x06002 4x06002 I:6001	0,0x0000 B:00 00			UINT16 R/O	
Current status of the converter						
DIP SWITCH						
DIP SWITCH	3x10010 4x10010 I:10009	65,0x0041 B:00 41			UINT16 R/O	
Returns the current setting of the Dip switches. For ULTRA SLIM IOs: The current value of the DIP switches: Bit 0: DIP Switch 1 (=0:OFF, =1:ON) Bit 1: DIP Switch 2 (=0:OFF, =1:ON) Bit 2: DIP Switch 3 (=0:OFF, =1:ON) Bit 3: DIP Switch 4 (=0:OFF, =1:ON) For BIG IOs: The current value of the DIP switches: Bit 0: DIP Switch 1 (=0:OFF, =1:ON) Bit 1: DIP Switch 2 (=0:OFF, =1:ON) Bit 2: DIP Switch 3 (=0:OFF, =1:ON) Bit 3: DIP Switch 4 (=0:OFF, =1:ON) Bit 4: DIP Switch 5 (=0:OFF, =1:ON) Bit 5: DIP Switch 6 (=0:OFF, =1:ON) Bit 6: DIP Switch 7 (=0:OFF, =1:ON) Bit 7: DIP Switch 8 (=0:OFF, =1:ON)						
PRODUCT DATA						
HW_GROUP	3x65201 4x65201 I:65200	4096,0x1000 B:10 00			UINT16 R/O	
This is the group of hardware of the current product						
SW_GROUP	3x65202 4x65202 I:65201	37,0x0025 B:00 25			UINT16 R/O	
This is the group of software of the current product						
SW_VERSION	3x65203 4x65203 I:65202	4352,0x1100 B:11 00			UINT16 R/O	
		SW VERSION:1.1.0				
This is the current software version of the firmware						
SW_AUTHOR	3x65204 4x65204 I:65203	18771,0x4953 B:49 53			UINT16 R/O	
This is the current software author of the firmware						

MODBUS SETTINGS						
UNIT_ID	3x65222 4x65222 I:65221	1,0x0001 B:00 01		N/A:NO CHANGE	UINT16 R/W	NO
		UNIT ID:1				
f the host reads this register, the current programmed unit ID is returned. All values above unit ID 255 define also the unit ID 255. If the host write a new value into this register, the new value will be stored in the FLASH as the new unit ID. The new unit ID is activated after a power off/power on cycle or a software reboot of the module. The host can execute a reboot in writing to the register RESET SYSTEM. NOTE:DIP switch 4 must set to OFF to activate this unit ID, otherwise the unit ID is 255.						
HINT:This settings will be active after you repower or reset your device !!						
BAUD_RATE	3x65223 4x65223 I:65222	57600,0x0000E100 B:00 00 E1 00	38400	38400	UINT32 R/W	NO
		57600Bd		ENTER BAUD RATE		
This is the current configured baud rate in the FLASH For ULTRA SLIM IOs RESI-xxx-SIO: This baudrate is only used, if DIP switch mode DIP1=ON+DIP2=ON (BR) (default is 57600bd) For BIG IOs RESI-xxx-SIO: This baudrate is only used, if DIP switch mode DIP7=ON (PARAMETER) (default is 57600bd)						
Valid baud rates are: 300bd, 600bd, 900bd, 1200bd, 2400bd, 4800bd, 9600bd, 19200bd, 38400bd, 57600bd, 115200bd, 128000bd 230400bd, 250000bd, 256000bd						
HINT:This settings will be active after you repower or reset your device !!						
PARITY	3x65225 4x65225 I:65224	0,0x0000 B:00 00		N/A:NO CHANGE	UINT16 R/W	NO
		NO PARITY		SELECT PARITY		
If the register is read out, the currently set parity of the serial interface is returned. Writing a value to this register will change the new parity in FLASH. This will only take effect after a restart of the module. This can be triggered by writing to the RESET SYSTEM register. Parity values are 0: no parity 1: even parity 2: odd parity						
STOP BITS	3x65226 4x65226 I:65225	1,0x0001 B:00 01		N/A:NO CHANGE	UINT16 R/W	NO
		ONE STOPBIT		SELECT STOPBITS		
f the register is read out, the currently set number of stop bits of the serial interface is returned. Writing a value to this register will change the new number of stop bits in the FLASH. This will only take effect after a restart of the module. This can be triggered by writing to the RESET SYSTEM register. Values for stop bits are 1: one stop bit 2: two stop bits						

7.4.2 ASCII protocol

All of our IO modules communicate with very simple ASCII commands. The following special characters are used in this description:

stands for the **hash sign** ASCII character 35dec or 0x23
: stands for the **colon** ASCII characters 58dec or 0x3A
= stands for the **equal sign** with the ASCII code 61ec or 0x3D
- stands for the **minus sign** with the ASCII code 45dec or 0x2D
, stands for the **comma** with the ASCII code 44dec or 0x2C
<CR> or **CR** stands for the **CARRIAGE RETURN** ASCII character 13dec or 0x0D. This is shown as CR in the following.
<SP> or ☐ stands for **SPACE**. This is the space in ASCII code 32dec or 0x20. The space is shown as , hereinafter.
In the following **<ADR>** is used for the **bus address**. This can be transmitted in decimal or hexadecimal and is separated from the following command with a comma (ASCII characters 44dec or 0x2C). Hexadecimal numbers always start with 0x. Only the ASCII characters '0' - '9' 48dec to 57dec, 0x30-0x39 and 'A' to 'F', 65dec to 70dec, 0x41-0x46 may be used. Each module always responds to broadcast address 0 and its own bus address. An external DIP switch can be used to quickly switch between the fixed bus address 255 and the programmed bus address. See the DIP switch description.

7.4.2.1 COMMUNICATION SEQUENCE

In principle, the IO module does not send any characters by itself. Communication always starts from the host. If only one IO module is used on a bus line (e.g. with RS232 interface), there is no need for a bus address in the protocol. In RS485 mode, however, several modules can be connected on an RS485 line. Then a bus address is absolutely necessary for communication.

The command structure looks like this:

The host sends a command or a command with parameters without a bus address:
#<command><CR> or
#<command>:<parameter><CR>

The module responds when it feels addressed with the telegram:
#<respond><CR>

The host sends the following to the module with the bus address:
#<ADR>,<command><CR> or
#<ADR>,<command>:<parameter><CR>

The module then replies with:
#<ADR>,<reply><CR>

The bus address is in the range from 1dec to 255dec or 0x00 to 0xFF hexadecimal. The setting is made using our free configuration software MODBUSConfigurator or our free LIBRE OFFICE® based configurator.

There are two spellings for each command. A long version and a short version, so that you have to send less. For example, you can query the software version with the VERSION command or with the VER command.

7.4.2.2 Example: Query VERSION

This command provides the current type of the module.

Long host version:
#VERSION<CR> or
#<ADR>,VERSION<CR>

Short host version:
#VER<CR> or
#<ADR>,VER<CR>

Reply:
#VERSION:<HIGH>.<MED>.<LOW><CR> or
#<ADR>,VERSION:<HIGH>,<MED>,<LOW><CR>

<HIGH>.<MED>.<LOW> represents the current software version, e.g. 3.0.0

Examples:
#VERSION$_{CR}$
#VERSION:3.0.0$_{CR}$

With broadcast address in decimal and long version:
#0,VERSION$_{CR}$
#0,VERSION:3.0.0$_{CR}$

With broadcast address in hexadecimal and short version:
#0x00,VER$_{CR}$
#0x00,VERSION:3.0.0$_{CR}$

With bus address 255 in decimal
#255,VER$_{CR}$
#255,VERSION:3.0.0$_{CR}$

With bus address 255 in hexadecimal
#0xFF,VERSION$_{CR}$
#0xFF,VERSION:3.0.0$_{CR}$

With bus address 43 in decimal
#43,VER$_{CR}$
#43,VERSION:3.0.0$_{CR}$

With bus address 43 in hexadecimal
#0x2B,VER$_{CR}$
#0x2B,VERSION:3.0.0$_{CR}$

7.4.2.3 Example: Query module TYPE

This command provides the current type of the module.

Long host version:
#TYPE<CR> or
#<ADR>,TYPE<CR>

Host short version:
#TYP<CR> or
#<ADR>,TYP<CR>

Respond:
#TYPE:<TYP><CR> or
#<ADR>,TYPE:<TYP><CR>

<TYP> represents the current type of the module. A RESI-2RI-SIO is shown as an example

Examples:
#TYPE$_{CR}$
#TYPE:RESI-2RI-SIO$_{CR}$

#255,TYP$_{CR}$
#255,TYPE:RESI-2RI-SIO$_{CR}$

7.4.2.4 Table of all ASCII commands

In this list you will find all possible ASCII commands. Only the version including the bus address is listed here. It has already been explained that this can also be omitted. If an argument has the addition dec, it is returned as a decimal number. If an argument has the addition hex, a hexadecimal number is returned. Many commands return both the decimal and the hexadecimal representation. The host can thus choose which number conversion he would like to carry out.

Please refer to the description of individual products for more details about the available ASCII commands.

ASCII COMMANDS				
GET VERSION	**ASCII READ COMMAND**	#VERSION<CR> #VER<CR> Result: #VERSION:<VersionHi>,<VersionMed>,<VersionLo><CR>	ASCII	
	TX	#VERSION<CR>		
	RX	#1,VERSION:1.1.0<CR>		
		Current SW version:1.1.0		
Returns the version number of the module VersionHi: Version number high (1..255) VersionMed: Version number medium (1..255) VersionLo: Version number low (1..255)				
GET TYPE	**ASCII READ COMMAND**	#TYPE<CR> #TYP<CR> Result: #TYPE:<Type><CR>	ASCII	
	TX	#TYPE<CR>		
	RX	TYPE:RESI-S8RO-SIO<CR>		
		Current module type:RESI-S8RO-SIO		
Returns the current module type				
GET OWNER	**ASCII READ COMMAND**	#OWNER<CR> #OWN<CR> Result: #OWNER:<Owner><CR>	ASCII	
	TX	#OWNER<CR>		
	RX	OWNER:RESI<CR>		
		Current owner:RESI		
Returns the current owner of the module				
GET CREATOR	**ASCII READ COMMAND**	#CREATOR<CR> #CRE<CR> Result: #CREATOR:<Creator><CR>	ASCII	
	TX	#CREATOR<CR>		
	RX	#1,CREATOR:DI HC SIGL,MSC<CR>		
		Current creator:DI HC SIGL,MSC		
Returns the current creator of the module				
GET COPYRIGHT	**ASCII READ COMMAND**	#COPYRIGHT<CR> #COPY<CR> Result: #COPYRIGHT:<Copyright><CR>	ASCII	
	TX	#COPYRIGHT<CR>		
	RX	#1,COPYRIGHT:2015-20 BY RESI AND DI HC SIGL,MSC WWW.RESI.CC<CR>		
		Current copyright:2015-20 BY RESI AND DI HC SIGL,MSC WWW.RESI.CC		
Returns the current copyright of the module				
GET DIP SWITCH	**ASCII READ COMMAND**	#GET DIP<CR> #GDIP<CR> Result: #GDIP:<DIPSwitchDec>,<DIPSwitchHex><CR>	ASCII	
	TX	#GET DIP<CR>		
	RX	#1,GDIP:65,0x41<CR>		
		Current DIP SWITCH settings:01000001		
Returns the current setting of the Dip switches as decimal number and as hexadecimal number. DIPSwitchDec DIPSwitchHex The current value of the DIP switches: Bit 0: DIP Switch 1 (=0:OFF, =1:ON) Bit 1: DIP Switch 2 (=0:OFF, =1:ON) Bit 2: DIP Switch 3 (=0:OFF, =1:ON) Bit 3: DIP Switch 4 (=0:OFF, =1:ON) Bit 4: DIP Switch 5, if available (=0:OFF, =1:ON) Bit 5: DIP Switch 6, if available (=0:OFF, =1:ON) Bit 6: DIP Switch 7, if available (=0:OFF, =1:ON) Bit 7: DIP Switch 8, if available (=0:OFF, =1:ON)				

MODBUS INTERFACE				
SET MODBUS ADDRESS	**ASCII WRITE COMMAND**	#SET MODBUS ADDRESS:<UNITID><CR> #SETMBADR:<UNITID><CR> Result: #OK<CR>	ASCII	NO
	UNITID	1		
	TX	#SET MODBUS ADDRESS:1<CR>		
	RX	N/A		
Redefines the unit ID of the module. This change will affect the MODBUS/RTU communication immediately. As a Unit IO you can use the values 0dec to 255dec. HINT: The new settings are activated after a system reboot or power off on cycle!				
SET MODBUS BAUDRATE	**ASCII WRITE COMMAND**	#SET MODBUS BAUDRATE:<BAUD><CR> #SETMBBAUD:<BAUD><CR> Result: #OK<CR>	ASCII	NO
	BAUD	57600:57600BD		
	TX	#SET MODBUS BAUDRATE:57600<CR>		
	RX	N/A		
Sets a new baud rate in the FLASH For ULTRA SLIM IOs RESI-xxx-SIO: This baudrate is only used, if DIP switch mode DIP1=ON+DIP2=ON (BR) (default is 57600bd) For BIG IOs RESI-xxx-SIO: This baudrate is only used, if DIP switch mode DIP7=ON (PARAMETER) (default is 57600bd) The following baudrates are allowed: 300bd, 600bd, 900bd, 1200bd, 2400bd, 4800bd, 9600bd, 19200bd, 38400bd, 57600bd, 115200bd, 128000bd 230400bd, 250000bd, 256000bd HINT: The new setup parameters will be active after a restart of the module.				
SET MODBUS PARITY	**ASCII WRITE COMMAND**	#SET MODBUS PARITY:<PARITY><CR> #SETMBPAR:<PARITY><CR> Result: #OK<CR>	ASCII	NO
	PARITY	NONE:NO PARITY		
	TX	#SET MODBUS PARITY:NONE<CR>		
	RX	N/A		
Sets a new parity for the serial interface. MBParity: NONE: no parity EVEN: even parity ODD: odd parity HINT: The new setup parameters will be active after a restart of the module.				
SET MODBUS STOPS	**ASCII WRITE COMMAND**	#SET MODBUS STOP:<STOPBIT><CR> #SETMBSTOP:<STOPBIT><CR> Result: #OK<CR>	ASCII	NO
	STOPBIT	ONE:ONE STOPBIT		
	TX	#SET MODBUS STOP:ONE<CR>		
	RX	N/A		
Sets a new amount of stop bits for the serial interface. MBStops ONE: one stop bit TWO: two stop bits HINT: The new setup parameters will be active after a restart of the module.				
SET MODBUS PARAMS	**ASCII WRITE COMMAND**	#SET MODBUS PARAMS:<UNITID>,<BAUD>,<PARITY>,<STOPBIT><CR> #SETMBPARAMS:<UNITID>,<BAUD>,<PARITY>,<STOPBIT><CR> Result: #OK<CR>	ASCII	YES
	UNITID	1		
	BAUD	57600:57600BD		
	PARITY	NONE:NO PARITY		
	STOPBIT	ONE:ONE STOPBIT		
	TX	#SET MODBUS PARAMS:1,57600,NONE,ONE<CR>		
	RX	N/A		
Sets all parameters for serrial interface				

GET MODBUS ADDRESS	**ASCII READ COMMAND**	#GET MODBUS ADDRESS<CR> #GMBADR<CR> Result: #GMBADR:<MBUnitDec>,<MBFLASHDec>,<MBUnitHex>,<MBFLASHHex><CR>	ASCII	
	TX	#GET MODBUS ADDRESS<CR>		
	RX	#1,GMBADR:1,1,0x1,0x1<CR>		
		Current MODBUS unit ID for DIP4=OFF:1,1,0x1,0x1		
Shows the current used MODBUS/RTU or ASCII unit address and shows also the stored unit address in the FLASH memory, which is only used if the DIP switch for the bus address is set to 0.MBUnitDecMBUnitHexThe current used MODBUS/RTU unit or ASCII address for communicationMBFLASHDecMBFLASHHexThe internal stored MODBUS/RTU unit address or ASCII address from the FLASH memory, if the DIP switch DIP3 is OFF.				
GET MODBUS BAUDRATE	**ASCII READ COMMAND**	#GET MODBUS BAUDRATE<CR> #GMBBAUD<CR> Result: #GMBBAUD:<BaudRate><CR>	ASCII	
	TX	#GET MODBUS BAUDRATE<CR>		
	RX	#1,GMBBAUD:57600<CR>		
		Current baudrate for DIP1+2=ON:57600		
This is the current configured baud rate in the FLASH For ULTRA SLIM IOs RESI-xxx-SIO: This baudrate is only used, if DIP switch mode DIP1=ON+DIP2=ON (BR) (default is 57600bd) For BIG IOs RESI-xxx-SIO: This baudrate is only used, if DIP switch mode DIP7=ON (PARAMETER) (default is 57600bd) The following baudrates are allowed: 300bd, 600bd, 900bd, 1200bd, 2400bd, 4800bd, 9600bd, 19200bd, 38400bd, 57600bd, 115200bd, 128000bd 230400bd, 250000bd, 256000bd				
GET MODBUS PARITY	**ASCII READ COMMAND**	#GET MODBUS PARITY<CR> #GMBPAR<CR> Result: #GMBPAR:<MBParity><CR>	ASCII	
	TX	#GET MODBUS PARITY<CR>		
	RX	#1,GMBPAR:NONE<CR>		
		Current parity:NONE		
Shows the current configured parity of the serial interface. MBParity NONE: no parity EVEN: even parity ODD: odd parity				
GET MODBUS STOP	**ASCII READ COMMAND**	#GET MODBUS STOP<CR> #GMBSTOP<CR> Result: #GMBSTOP:<MBStop><CR>	ASCII	
	TX	#GET MODBUS STOP<CR>		
	RX	#1,GMBSTOP:ONE<CR>		
		Current stopbit(s):ONE		
Shows the current configured parity of the serial interface. MBParity NONE: no parity EVEN: even parity ODD: odd parity				
GET MODBUS PARAMS	**ASCII READ COMMAND**	#GET MODBUS PARAMS<CR> #GMBPARAMS<CR> Result: #GMBPARAMS:<MBUnitDec>,<MBFLASHDec>,<MBUnitHex>,<MBFLASHHex>, <MBBaudrateDec>,<MBBaudrateHex>,<MBParity>,<MBStops><CR>	ASCII	
	TX	#GET MODBUS PARAMS<CR>		
	RX	#1,GMBPARAMS:1,0x1,1,0x1,57600,0xE100,NONE,ONE<CR>		
		Current MODBUS unit ID used:1		
		Current MODBUS unit ID in FLASH:1		
		Current baudrate in FLASH:57600		
		Current parity in FLASH:NONE		
		Current stopbit(s) in FLASH:ONE		
Returns the complete settings for serial interface				

SYSTEM COMMANDS				
RESET	**ASCII WRITE COMMAND**	#RESET<CR> #RST<CR> Result: #OK<CR>	ASCII	NO
	TX	#RESET<CR>		
	RX	N/A		
Executes a software reset (Reboot) of the module.				
FACTORY RESET	**ASCII WRITE COMMAND**	#FACTORY RESET<CR> #FRST<CR> Result: #OK<CR>	ASCII	NO
	TX	#FACTORY RESET<CR>		
	RX	N/A		

7.5 RESI-xxx-SIO SERIAL PARAMETERS

Normally you select the serial parameters via DIP switch for fast use of the modules. But in special cases you will need a different setup for the serial interface. Please find all information here, how you can change the serial setup via MODBUS/RTU or ASCII commands.

HINT: This commands are only valid for the ULTRA SLIM IOs with serial RS232 or RS485 interface and for the BIG IOs with RS485 interface.

7.5.1 ULTRA SLIM IOs: Howto change the UnitID of the IO module

When DIP switch #4:FD is set to ON, the module always communicates with the Unit ID 255. When you switch this DIP switch to OFF, the module will use the internal Unit ID from the FLASH memory.

You can set this Unit ID either with this MODBUS register:

UNIT_ID	3x65222 4x65222 I:65221	1,0x0001 B:00 01		N/A:NO CHANGE	UINT16 R/W	NO
		UNIT ID:1				
f the host reads this register, the current programmed unit ID is returned. All values above unit ID 255 define also the unit ID 255. If the host write a new value into this register, the new value will be stored in the FLASH as the new unit ID. The new unit ID is activated after a power off/power on cycle or a software reboot of the module. The host can execute a reboot in writing to the register RESET SYSTEM. NOTE:DIP switch 4 must set to OFF to activate this unit ID, otherwise the unit ID is 255.						

or you use this ASCII command:

SET MODBUS ADDRESS	**ASCII WRITE COMMAND**	#SET MODBUS ADDRESS:<UNITID><CR> #SETMBADR:<UNITID><CR> Result: #OK<CR>	ASCII	NO
	UNITID	1		
	TX	#SET MODBUS ADDRESS:1<CR>		
	RX	N/A		
Redefines the unit ID of the module. This change will affect the MODBUS/RTU communication immediately. As a Unit IO you can use the values 0dec to 255dec. HINT: The new settings are activated after a system reboot or power off on cycle!				

After you changed the Unit ID you have to restart the module to make the changes effective. You can also use the MODBUS register for resetting the module:

SOFTWARE RESET						
RESET	1x06001 2x06001 I:6000	0,0x00 B:00		N/A:NO CHANGE	BIT R/W	NO
Performs a software reset, whenever 1 is written to this register. If the host writes to this register 1, the module executes a soft reset (reboot).						
RESET	3x06001 4x06001 I:6000	0,0x0000 B:00 00		N/A:NO CHANGE	UINT16 R/W	NO
Performs a software reset, whenever 1 is written to this register. If the host writes to this register 1, the module executes a soft reset (reboot).						

Or you use the ASCII command:

SYSTEM COMMANDS				
RESET	**ASCII WRITE COMMAND**	#RESET<CR> #RST<CR> Result: #OK<CR>	ASCII	NO
	TX	#RESET<CR>		
	RX	N/A		
Executes a software reset (Reboot) of the module.				

Remember, only if the DIP Switch #4 FD=OFF, you can use your new UnitID. Otherwise the module communicates with UnitID 255.

7.5.2 ULTRA SLIM IOs: Howto change the parity+stopbits of the IO module

Usually the IO module communicates with no parity and one stopbit. But you can change this behaviour:
You can set the parity and the stop bits with this MODBUS register:

PARITY	3x65225 4x65225 I:65224	0,0x0000 B:00 00		N/A:NO CHANGE	UINT16 R/W	NO
		NO PARITY		SELECT PARITY		
If the register is read out, the currently set parity of the serial interface is returned. Writing a value to this register will change the new parity in FLASH. This will only take effect after a restart of the module. This can be triggered by writing to the RESET SYSTEM register. Parity values are 0: no parity 1: even parity 2: odd parity						
STOP BITS	3x65226 4x65226 I:65225	1,0x0001 B:00 01		N/A:NO CHANGE	UINT16 R/W	NO
		TWO STOPBITS		SELECT STOPBITS		
f the register is read out, the currently set number of stop bits of the serial interface is returned. Writing a value to this register will change the new number of stop bits in the FLASH. This will only take effect after a restart of the module. This can be triggered by writing to the RESET SYSTEM register. Values for stop bits are 1: one stop bit 2: two stop bits						

or you use this ASCII command:

SET MODBUS PARITY	**ASCII WRITE COMMAND**	#SET MODBUS PARITY:<PARITY><CR> #SETMBPAR:<PARITY><CR> Result: #OK<CR>	ASCII	NO
	PARITY	NONE:NO PARITY		
	TX	#SET MODBUS PARITY:NONE<CR>		
	RX	N/A		
Sets a new parity for the serial interface. MBParity: NONE: no parity EVEN: even parity ODD: odd parity HINT: The new setup parameters will be active after a restart of the module.				
SET MODBUS STOPS	**ASCII WRITE COMMAND**	#SET MODBUS STOP:<STOPBIT><CR> #SETMBSTOP:<STOPBIT><CR> Result: #OK<CR>	ASCII	NO
	STOPBIT	ONE:ONE STOPBIT		
	TX	#SET MODBUS STOP:ONE<CR>		
	RX	N/A		
Sets a new amount of stop bits for the serial interface. MBStops ONE: one stop bit TWO: two stop bits HINT: The new setup parameters will be active after a restart of the module.				

After you changed the two settings you have to restart the module to make the changes effective. You can also use the MODBUS register for resetting the module:

SOFTWARE RESET						
RESET	1x06001 2x06001 I:6000	0,0x00 B:00		N/A:NO CHANGE	BIT R/W	NO
Performs a software reset, whenever 1 is written to this register. If the host writes to this register 1, the module executes a soft reset (reboot).						
RESET	3x06001 4x06001 I:6000	0,0x0000 B:00 00		N/A:NO CHANGE	UINT16 R/W	NO
Performs a software reset, whenever 1 is written to this register. If the host writes to this register 1, the module executes a soft reset (reboot).						

Or you use the ASCII command:

SYSTEM COMMANDS				
RESET	**ASCII WRITE COMMAND**	#RESET<CR> #RST<CR> Result: #OK<CR>	ASCII	NO
	TX	#RESET<CR>		
	RX	N/A		
Executes a software reset (Reboot) of the module.				

Remember, now the Module uses ALWAYS the new parity and stop bit setting in all communication modes!

7.5.3 ULTRA SLIM IOs: Howto change the baud rate of the IO module

Usually the IO module communicates with baud rates selected by the two DIP switches #1+#2. This will be:

#1	#2	
OFF	OFF	9600 baud
ON	OFF	19200 baud
OFF	ON	38400 baud
ON	ON	57600 baud or the new defined BAUDRATE from FLASH

But you can change the baud rate used with DIP switch setting ON,ON:
You can set the baud rate with this MODBUS register:

BAUD_RATE	3x65223 4x65223 I:65222	57600,0x0000E100 B:00 00 E1 00	38400	38400	UINT32 R/W	NO
		57600Bd		ENTER BAUD RATE		
This is the current configured baud rate in the FLASH For ULTRA SLIM IOs RESI-xxx-SIO: This baudrate is only used, if DIP switch mode DIP1=ON+DIP2=ON (BR) (default is 57600bd) For BIG IOs RESI-xxx-SIO: This baudrate is only used, if DIP switch mode DIP7=ON (PARAMETER) (default is 57600bd)						
Valid baud rates are: 300bd, 600bd, 900bd, 1200bd, 2400bd, 4800bd, 9600bd, 19200bd, 38400bd, 57600bd, 115200bd, 128000bd 230400bd, 250000bd, 256000bd						

or you use this ASCII command:

SET MODBUS BAUDRATE	**ASCII WRITE COMMAND**	#SET MODBUS BAUDRATE:<BAUD><CR> #SETMBBAUD:<BAUD><CR> Result: #OK<CR>	ASCII	NO
	BAUD	57600:57600BD		
	TX	#SET MODBUS BAUDRATE:57600<CR>		
	RX	N/A		
Sets a new baud rate in the FLASH For ULTRA SLIM IOs RESI-xxx-SIO: This baudrate is only used, if DIP switch mode DIP1=ON+DIP2=ON (BR) (default is 57600bd) For BIG IOs RESI-xxx-SIO: This baudrate is only used, if DIP switch mode DIP7=ON (PARAMETER) (default is 57600bd) The following baudrates are allowed: 300bd, 600bd, 900bd, 1200bd, 2400bd, 4800bd, 9600bd, 19200bd, 38400bd, 57600bd, 115200bd, 128000bd 230400bd, 250000bd, 256000bd HINT: The new setup parameters will be active after a restart of the module.				

After you changed the two settings you have to restart the module to make the changes effective. You can also use the MODBUS register for resetting the module:

SOFTWARE RESET						
RESET	1x06001 2x06001 I:6000	0,0x00 B:00		N/A:NO CHANGE	BIT R/W	NO
Performs a software reset, whenever 1 is written to this register. If the host writes to this register 1, the module executes a soft reset (reboot).						
RESET	3x06001 4x06001 I:6000	0,0x0000 B:00 00		N/A:NO CHANGE	UINT16 R/W	NO
Performs a software reset, whenever 1 is written to this register. If the host writes to this register 1, the module executes a soft reset (reboot).						

Or you use the ASCII command:

SYSTEM COMMANDS				
RESET	**ASCII WRITE COMMAND**	#RESET<CR> #RST<CR> Result: #OK<CR>	ASCII	NO
	TX	#RESET<CR>		
	RX	N/A		
Executes a software reset (Reboot) of the module.				

Remember, now the Module uses ALWAYS the new baud rate, if you set the two DIP switches #1+#2 to ON,ON!

7.5.4 BIG IOs: Howto change the UnitID of the IO module

When DIP switches #1-#4:ADDRESS are all set to OFF, the module always communicates with the Unit ID from the FLASH. Otherwise the module uses the UnitIDs 1..15.

You can set this Unit ID either with this MODBUS register:

UNIT_ID	3x65222 4x65222 I:65221	1,0x0001 B:00 01		N/A:NO CHANGE	UINT16 R/W	NO
		UNIT ID:1				
f the host reads this register, the current programmed unit ID is returned. All values above unit ID 255 define also the unit ID 255. If the host write a new value into this register, the new value will be stored in the FLASH as the new unit ID. The new unit ID is activated after a power off/power on cycle or a software reboot of the module. The host can execute a reboot in writing to the register RESET SYSTEM. NOTE:DIP switch 4 must set to OFF to activate this unit ID, otherwise the unit ID is 255.						

or you use this ASCII command:

SET MODBUS ADDRESS	**ASCII WRITE COMMAND**	#SET MODBUS ADDRESS:<UNITID><CR> #SETMBADR:<UNITID><CR> Result: #OK<CR>	ASCII	NO
	UNITID	1		
	TX	#SET MODBUS ADDRESS:1<CR>		
	RX	N/A		
Redefines the unit ID of the module. This change will affect the MODBUS/RTU communication immediately. As a Unit IO you can use the values 0dec to 255dec. HINT: The new settings are activated after a system reboot or power off on cycle!				

After you changed the Unit ID you have to restart the module to make the changes effective. You can also use the MODBUS register for resetting the module:

SOFTWARE RESET						
RESET	1x06001 2x06001 I:6000	0,0x00 B:00		N/A:NO CHANGE	BIT R/W	NO
Performs a software reset, whenever 1 is written to this register. If the host writes to this register 1, the module executes a soft reset (reboot).						
RESET	3x06001 4x06001 I:6000	0,0x0000 B:00 00		N/A:NO CHANGE	UINT16 R/W	NO
Performs a software reset, whenever 1 is written to this register. If the host writes to this register 1, the module executes a soft reset (reboot).						

Or you use the ASCII command:

SYSTEM COMMANDS				
RESET	**ASCII WRITE COMMAND**	#RESET<CR> #RST<CR> Result: #OK<CR>	ASCII	NO
	TX	#RESET<CR>		
	RX	N/A		
Executes a software reset (Reboot) of the module.				

Remember, only if the DIP Switches #1-#4:ADDRESS are all OFF, you can use your new UnitID. Otherwise the module communicates with the selected UnitID.

7.5.5 BIG IOs: Howto change the parity+stopbits of the IO module

Usually the IO module communicates with no parity and one stopbit. What kind of parity and stop bit setting the IO module is using, is defined by DIP switch #8: PARAMETER.
If this DIP switch is set to OFF, the IO module ALWAYS uses no parity and one stopbit!
If this DIP switch is set to ON, the IO module will use the settings from the FLASH memory!

You can set the parity and the stop bits with this MODBUS register:

PARITY	3x65225 4x65225 I:65224	0,0x0000 B:00 00		N/A:NO CHANGE	UINT16 R/W	NO
		NO PARITY		SELECT PARITY		
If the register is read out, the currently set parity of the serial interface is returned. Writing a value to this register will change the new parity in FLASH. This will only take effect after a restart of the module. This can be triggered by writing to the RESET SYSTEM register. Parity values are 0: no parity 1: even parity 2: odd parity						
STOP BITS	3x65226 4x65226 I:65225	1,0x0001 B:00 01		N/A:NO CHANGE	UINT16 R/W	NO
		TWO STOPBITS		SELECT STOPBITS		
f the register is read out, the currently set number of stop bits of the serial interface is returned. Writing a value to this register will change the new number of stop bits in the FLASH. This will only take effect after a restart of the module. This can be triggered by writing to the RESET SYSTEM register. Values for stop bits are 1: one stop bit 2: two stop bits						

or you use this ASCII command:

SET MODBUS PARITY	**ASCII WRITE COMMAND**	#SET MODBUS PARITY:<PARITY><CR> #SETMBPAR:<PARITY><CR> Result: #OK<CR>	ASCII	NO
	PARITY	NONE:NO PARITY		
	TX	#SET MODBUS PARITY:NONE<CR>		
	RX	N/A		
Sets a new parity for the serial interface. MBParity: NONE: no parity EVEN: even parity ODD: odd parity HINT: The new setup parameters will be active after a restart of the module.				
SET MODBUS STOPS	**ASCII WRITE COMMAND**	#SET MODBUS STOP:<STOPBIT><CR> #SETMBSTOP:<STOPBIT><CR> Result: #OK<CR>	ASCII	NO
	STOPBIT	ONE:ONE STOPBIT		
	TX	#SET MODBUS STOP:ONE<CR>		
	RX	N/A		
Sets a new amount of stop bits for the serial interface. MBStops ONE: one stop bit TWO: two stop bits HINT: The new setup parameters will be active after a restart of the module.				

After you changed the two settings you have to restart the module to make the changes effective. You can also use the MODBUS register for resetting the module:

SOFTWARE RESET						
RESET	1x06001 2x06001 I:6000	0,0x00 B:00		N/A:NO CHANGE	BIT R/W	NO
Performs a software reset, whenever 1 is written to this register. If the host writes to this register 1, the module executes a soft reset (reboot).						
RESET	3x06001 4x06001 I:6000	0,0x0000 B:00 00		N/A:NO CHANGE	UINT16 R/W	NO
Performs a software reset, whenever 1 is written to this register. If the host writes to this register 1, the module executes a soft reset (reboot).						

Or you use the ASCII command:

SYSTEM COMMANDS				
RESET	**ASCII WRITE COMMAND**	#RESET<CR> #RST<CR> Result: #OK<CR>	ASCII	NO
	TX	#RESET<CR>		
	RX	N/A		
Executes a software reset (Reboot) of the module.				

Remember, now the Module uses the new parity and stop bit setting only, if DIP switch #8=ON!

7.5.6 BIG IOs: Howto change the baud rate of the IO module

Usually the IO module communicates with baud rates selected by the three DIP switches #5+#6+#7. This will be:

DIP #7:BR2	DIP #6:BR1	DIP #5:BR0	MODBUS/RTU or ASCII baud rate
OFF	OFF	OFF	4800bd
OFF	OFF	ON	9600bd
OFF	ON	OFF	19200bd
OFF	ON	ON	38400bd
ON	OFF	OFF	57600bd
ON	OFF	ON	115200bd
ON	ON	OFF	230400bd
ON	ON	ON	256000bd

This baud rates and the parity NONE and ONE stop bit are used, if the DIP switch #8 is set to OFF.
But you can change the baud rate used with DIP switch #8 PARAMETER setting to ON. Remember, that you will also use the configured parity and stop bits from the FLASH memory!

You can set the baud rate with this MODBUS register:

BAUD_RATE	3x65223 4x65223 I:65222	57600,0x0000E100 B:00 00 E1 00	38400	38400	UINT32 R/W	NO
		57600Bd		ENTER BAUD RATE		
This is the current configured baud rate in the FLASH For ULTRA SLIM IOs RESI-xxx-SIO: This baudrate is only used, if DIP switch mode DIP1=ON+DIP2=ON (BR) (default is 57600bd) For BIG IOs RESI-xxx-SIO: This baudrate is only used, if DIP switch mode DIP7=ON (PARAMETER) (default is 57600bd)						
Valid baud rates are: 300bd, 600bd, 900bd, 1200bd, 2400bd, 4800bd, 9600bd, 19200bd, 38400bd, 57600bd, 115200bd, 128000bd 230400bd, 250000bd, 256000bd						

or you use this ASCII command:

SET MODBUS BAUDRATE	**ASCII WRITE COMMAND**	#SET MODBUS BAUDRATE:<BAUD><CR> #SETMBBAUD:<BAUD><CR> Result: #OK<CR>	ASCII	NO
	BAUD	57600:57600BD		
	TX	#SET MODBUS BAUDRATE:57600<CR>		
	RX	N/A		
Sets a new baud rate in the FLASH For ULTRA SLIM IOs RESI-xxx-SIO: This baudrate is only used, if DIP switch mode DIP1=ON+DIP2=ON (BR) (default is 57600bd) For BIG IOs RESI-xxx-SIO: This baudrate is only used, if DIP switch mode DIP7=ON (PARAMETER) (default is 57600bd) The following baudrates are allowed: 300bd, 600bd, 900bd, 1200bd, 2400bd, 4800bd, 9600bd, 19200bd, 38400bd, 57600bd, 115200bd, 128000bd 230400bd, 250000bd, 256000bd HINT: The new setup parameters will be active after a restart of the module.				

After you changed the two settings you have to restart the module to make the changes effective. You can also use the MODBUS register for resetting the module:

SOFTWARE RESET						
RESET	1x06001 2x06001 I:6000	0,0x00 B:00		N/A:NO CHANGE	BIT R/W	NO
Performs a software reset, whenever 1 is written to this register. If the host writes to this register 1, the module executes a soft reset (reboot).						
RESET	3x06001 4x06001 I:6000	0,0x0000 B:00 00		N/A:NO CHANGE	UINT16 R/W	NO
Performs a software reset, whenever 1 is written to this register. If the host writes to this register 1, the module executes a soft reset (reboot).						

Or you use the ASCII command:

SYSTEM COMMANDS				
RESET	**ASCII WRITE COMMAND**	#RESET<CR> #RST<CR> Result: #OK<CR>	ASCII	NO
	TX	#RESET<CR>		
	RX	N/A		
Executes a software reset (Reboot) of the module.				

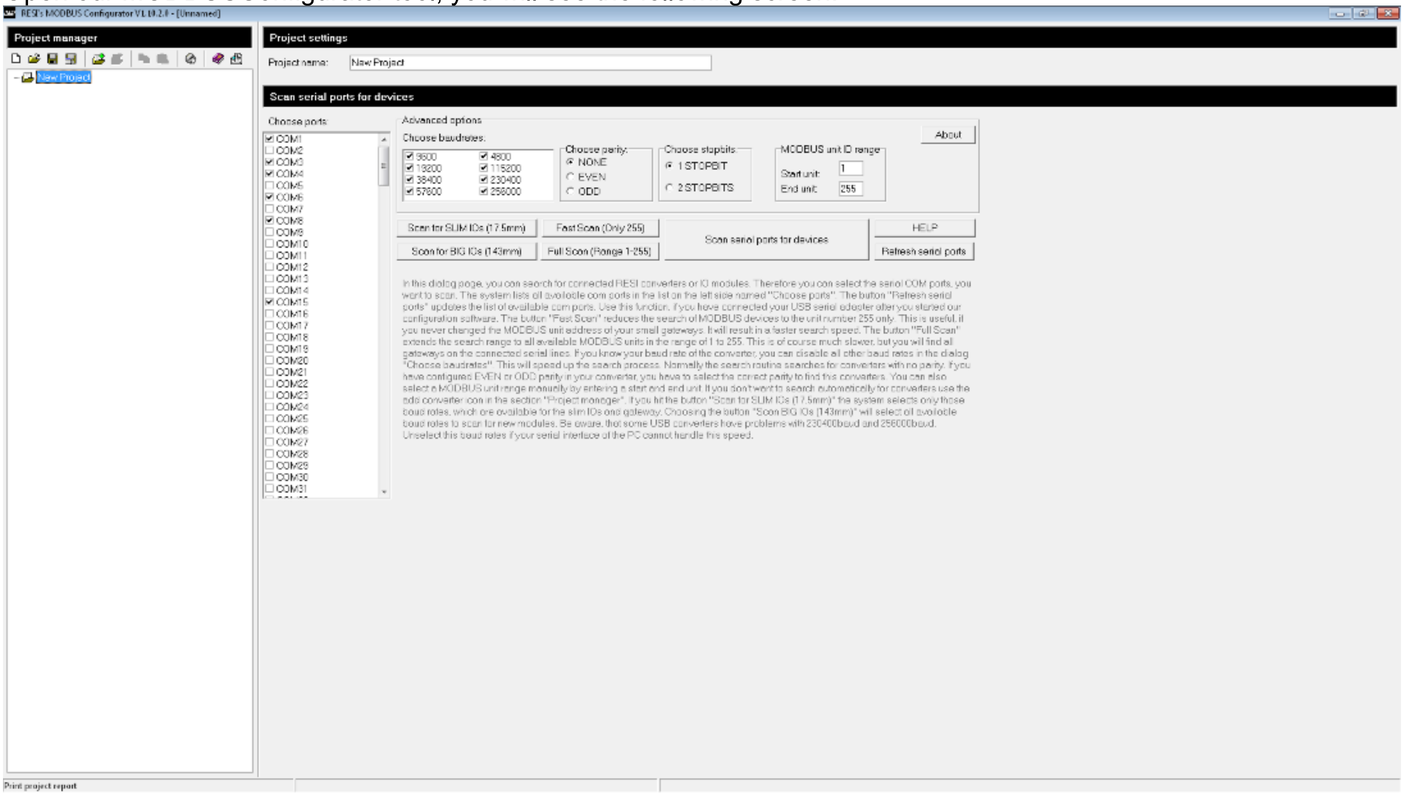
Remember, now the Module uses ALWAYS the new baud rate, the parity and the stop bits, if you set the DIP switch #8 to ON! If you set the DIP switch #8 to OFF, you will use the baud rate defined by DIP switch #5-#7 and the parity is always NONE and the stop bit is always ONE.

7.6 RESI's MODBUS Configurator

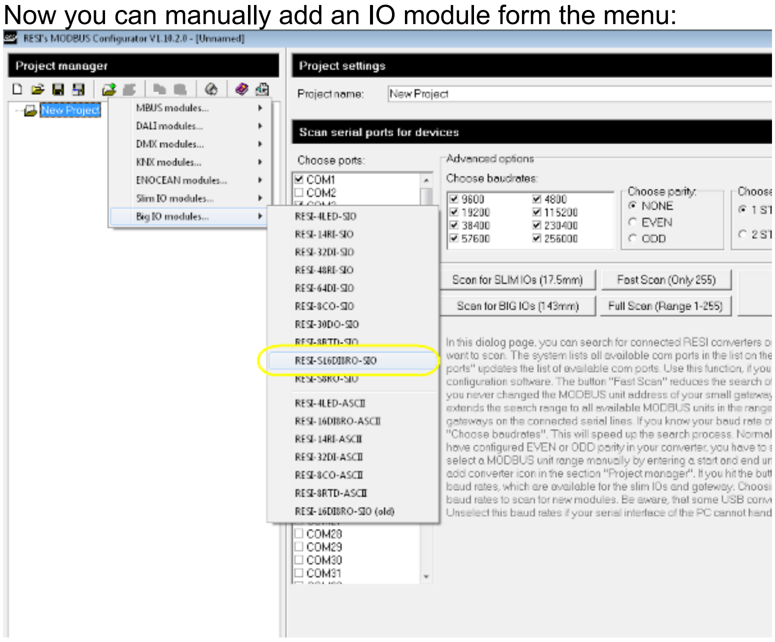
Almost all of our products can be used together with our MODBUSConfigurator software tool. You can configure and test the modules.

7.6.1 HOWTO manually establish a serial connection to the module

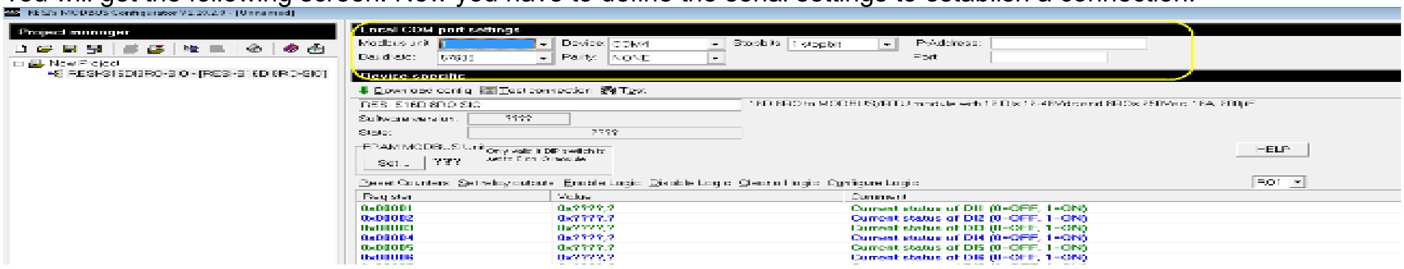
Open our MODBUSConfigurator tool, you will see the following screen:



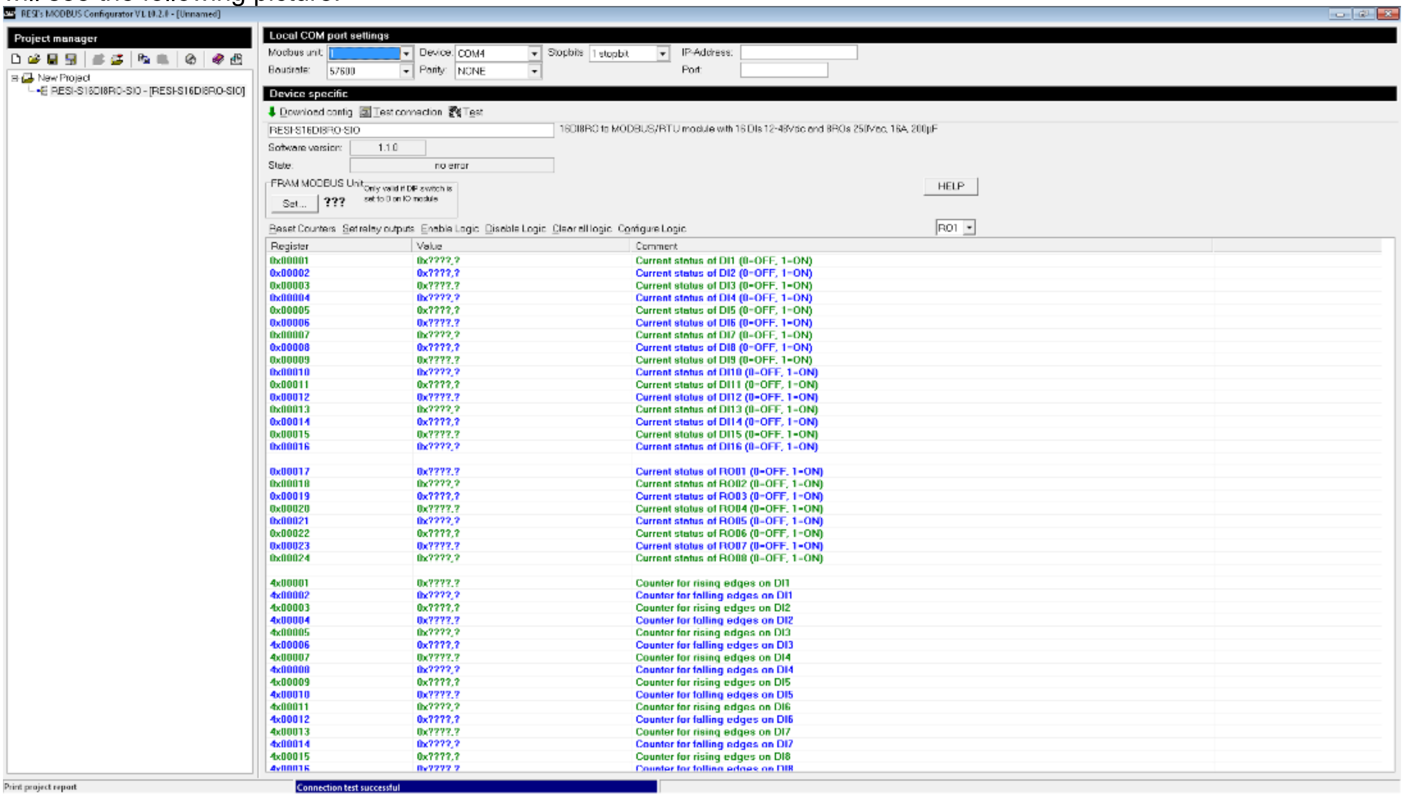
Now you can manually add an IO module form the menu:



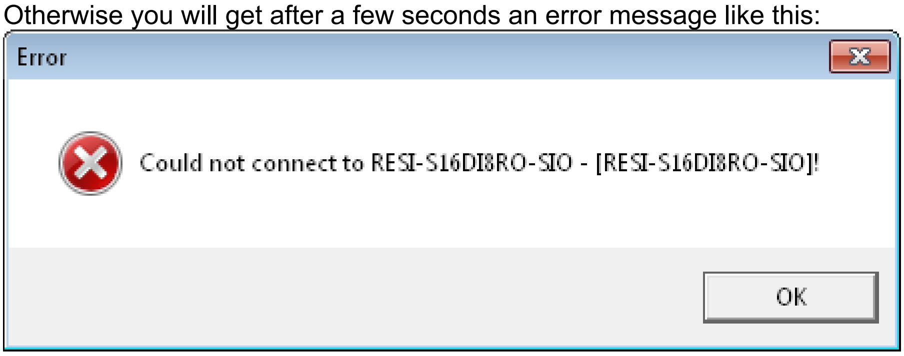
You will get the following screen. Now you have to define the serial settings to establish a connection:



Test the connection by pressing the button "Test connection". If you have successfully established the connection, you will see the following picture:

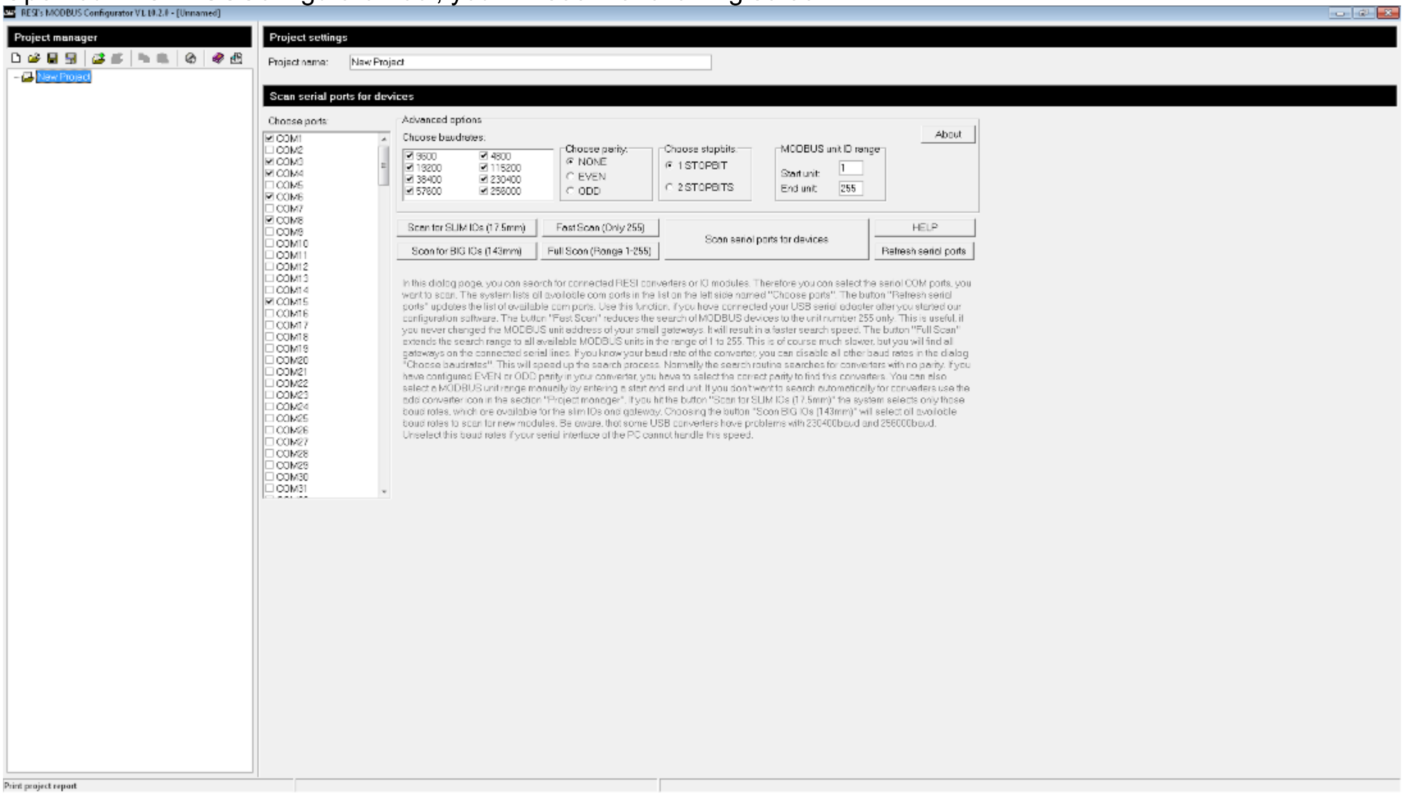


Otherwise you will get after a few seconds an error message like this:

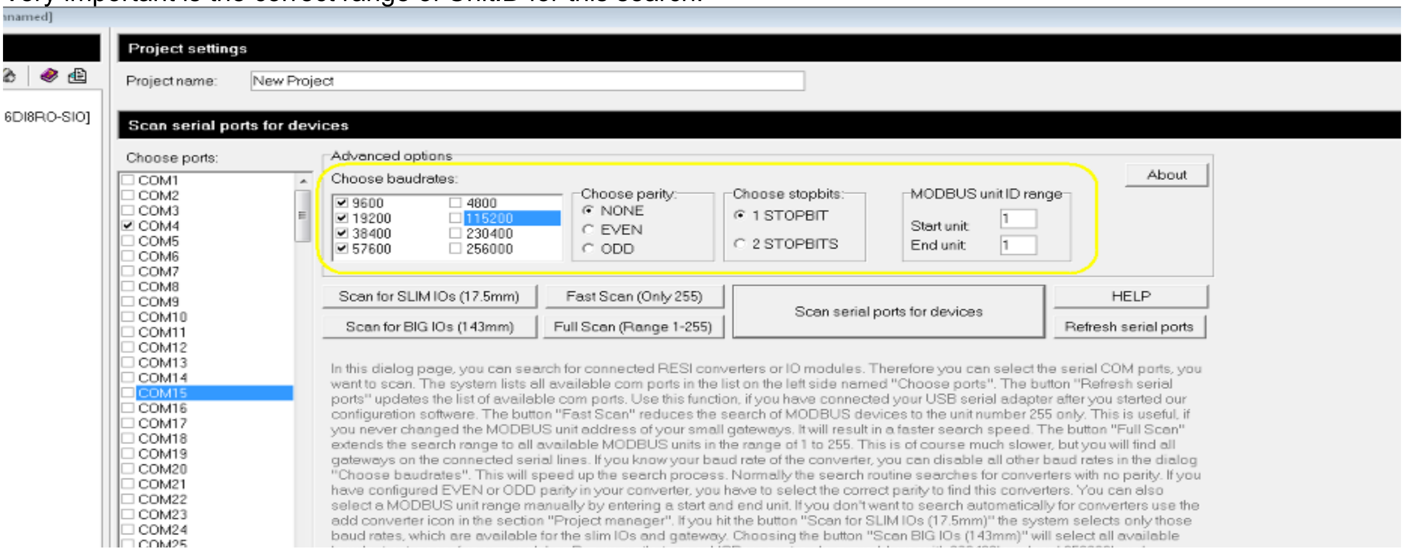


7.6.2 HOWTO search for serial modules

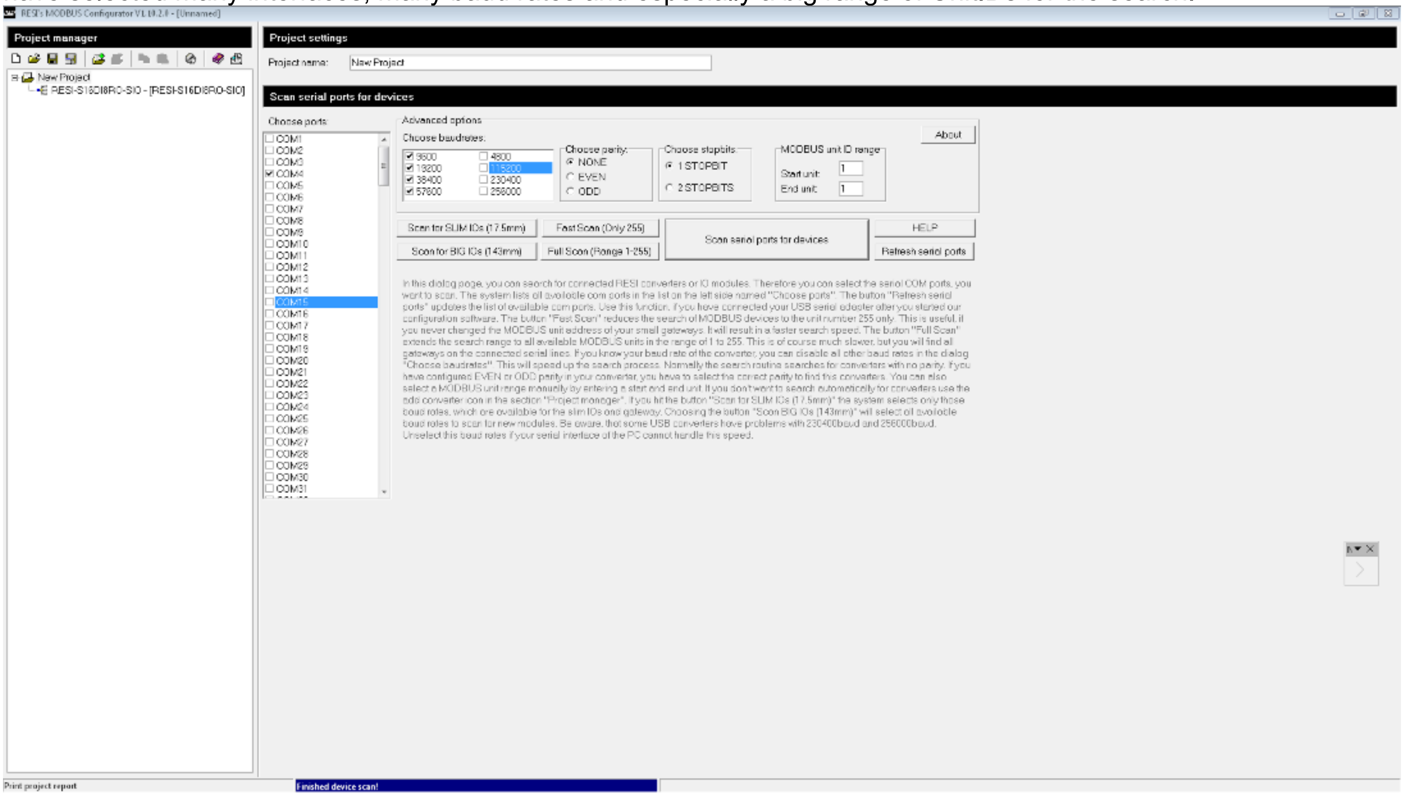
Open our MODBUSConfigurator tool, you will see the following screen:



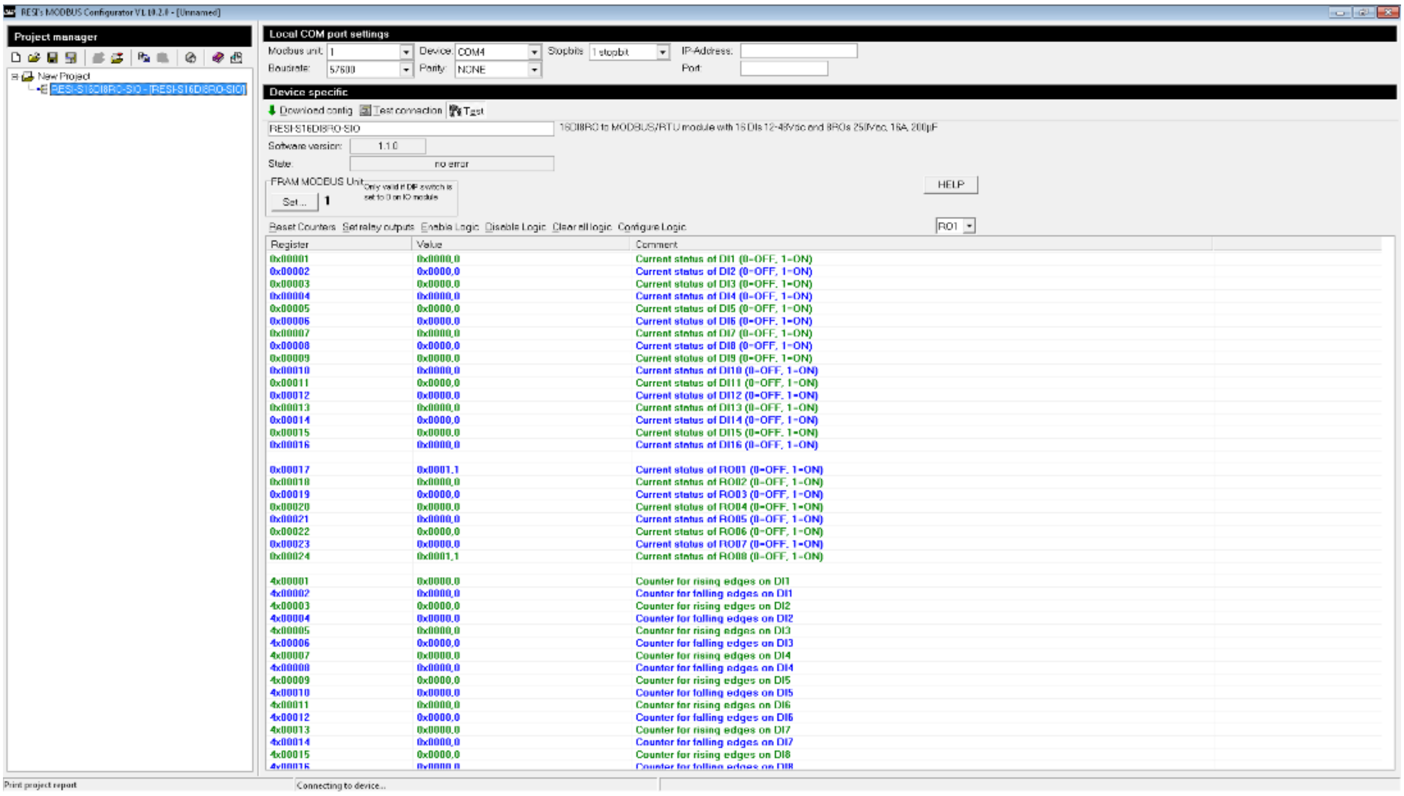
Select the correct serial interfaces, the correct baud rates, parity and stop bits for an automatic search for IO modules. Very important is the correct range of UnitID for this search:



Then click on the button "Scan serial ports for devices" to start the automatic search. This can last for minutes, if you have selected many interfaces, many baud rates and especially a big range of UnitIDs for the search.



Now select the module from the project tree and activate the test mode, you will see all actual values of your module on the screen:



You can also use now commands from the command bar to control the connected IO module.

8 Ethernet connection

Our Ethernet ULTRA SLIM IO or Ethernet BIG IO modules offer an Ethernet interface.

8.1 Ethernet connection for ULTRA SLIM IO modules

The following drawings show the correct Ethernet connection for all of our SLIMIO products:

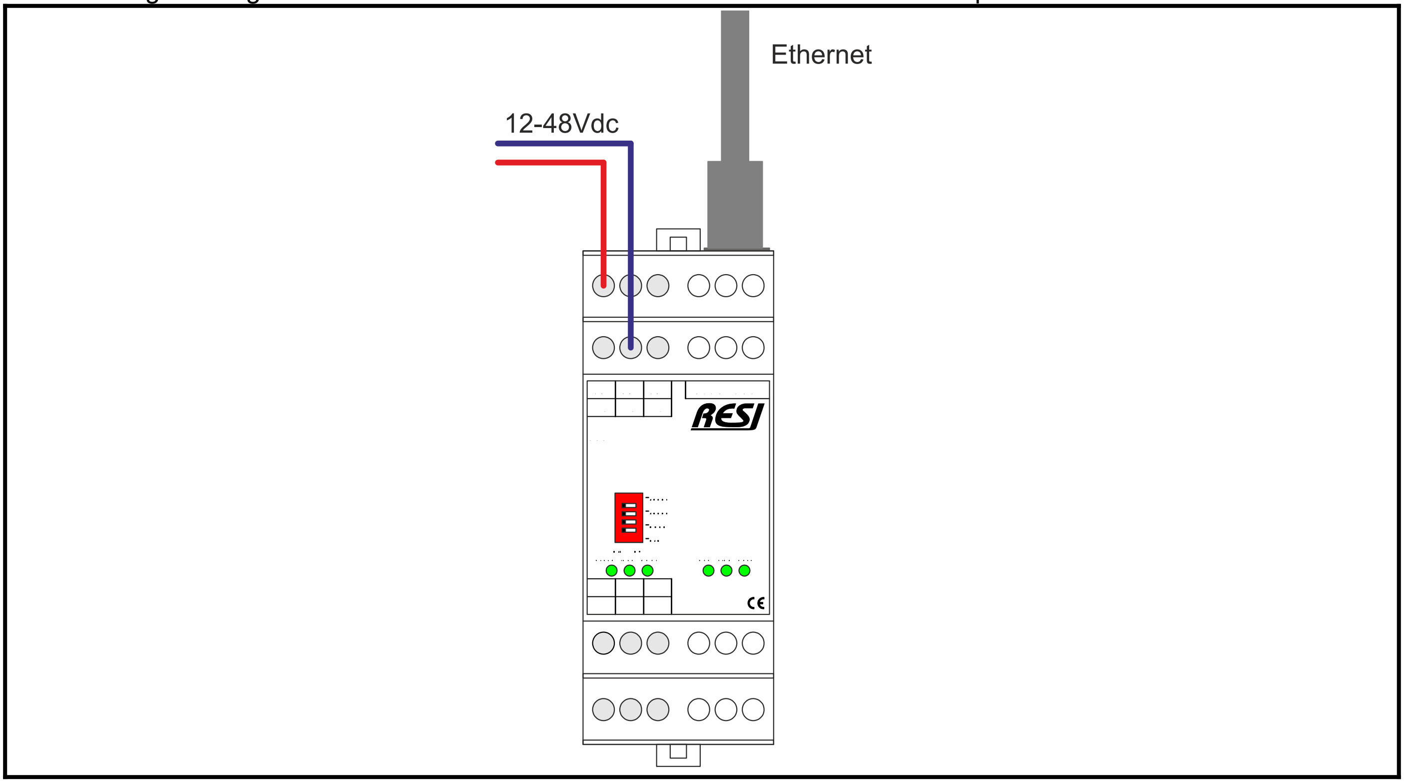


Figure: Ethernet connection for our ULTRA SLIM IO modules

8.2 Ethernet connection for BIG IO modules

The following drawings show the correct Ethernet connection for all of our BIGIO products:

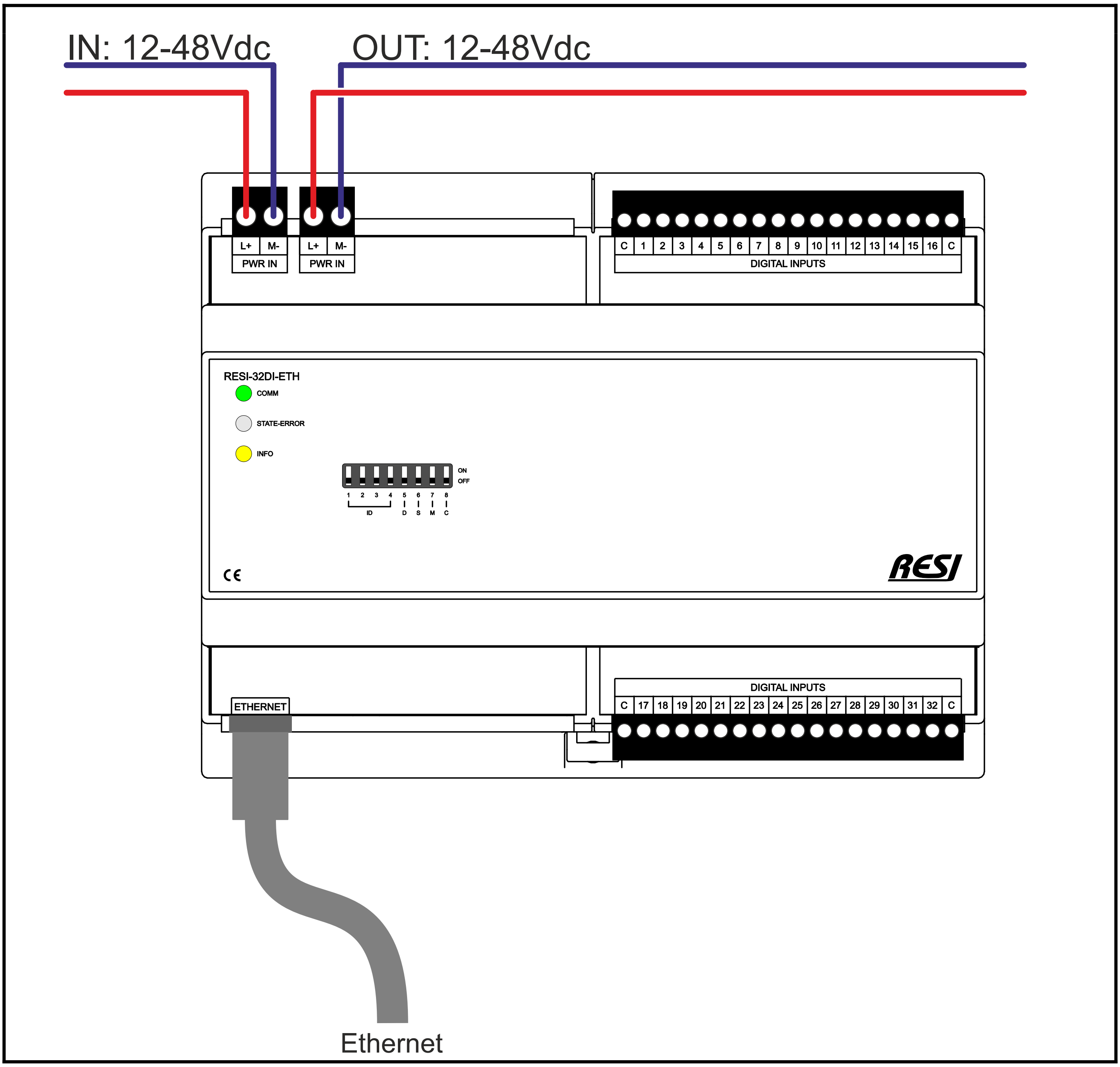


Figure: Ethernet connection for our BIG IO modules

8.3 RESI-xxx-ETH OPERATING MODES

The gateway basically supports two different operating modes:

- **TRANSPARENT MODE:** Bidirectional, transparent gateway between Ethernet socket data and IO module. All data arriving at the Ethernet socket are processed directly by the IO module. All data from the IO module is forwarded directly to the Ethernet socket. This mode is required for the ASCII protocol.

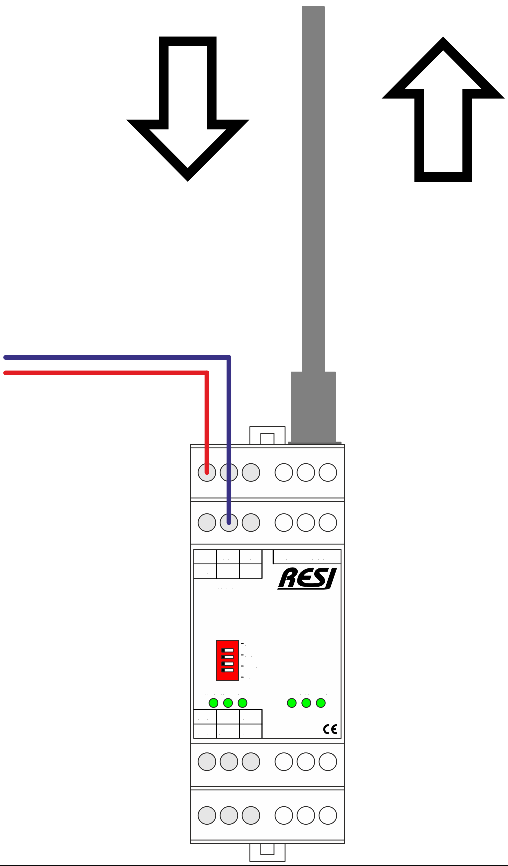


Figure: TRANSPARENT MODE for the RESI-xxx-ETH module

■ In this mode you can also use the **MODBUS/RTU protocol via Ethernet** to communicate with the IO module.

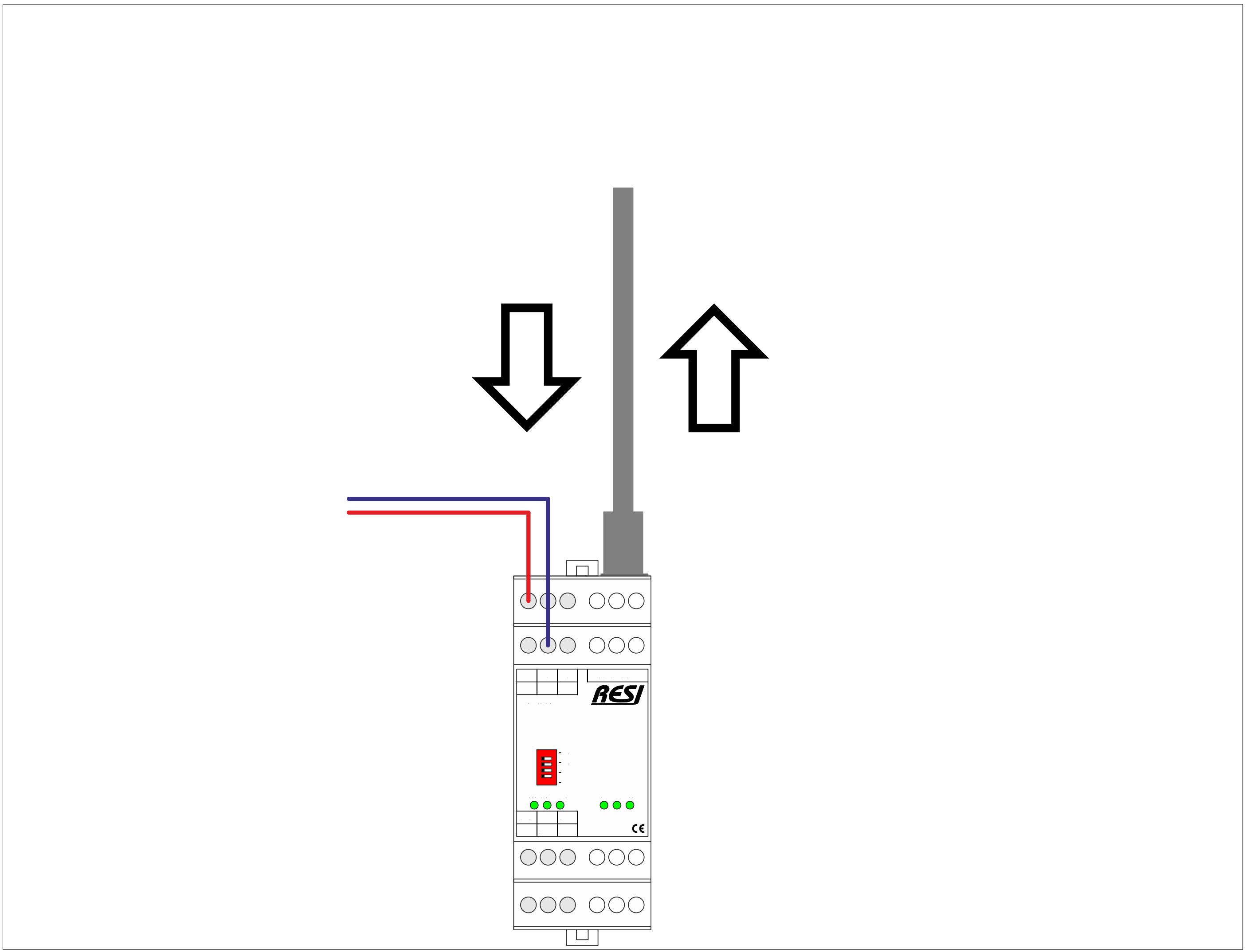


Figure: MODBUS/RTU via ETHERNET MODE for the RESI-xxx-ETH module

- **MODBUS/TCP server**: The module is a MODBUS/TCP server. A host with MODBUS/TCP master protocol can communicate directly with the IO module connected via Ethernet.

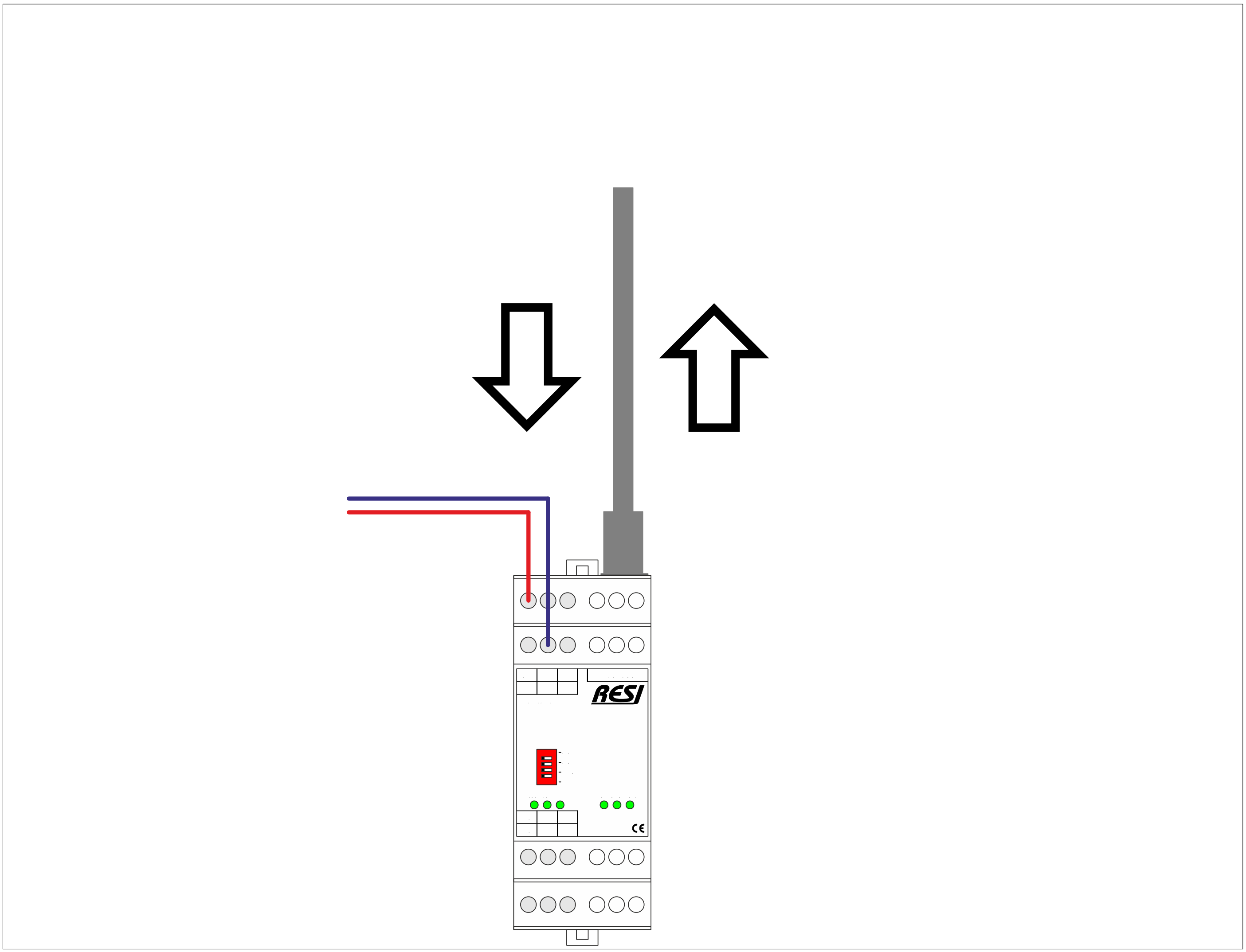


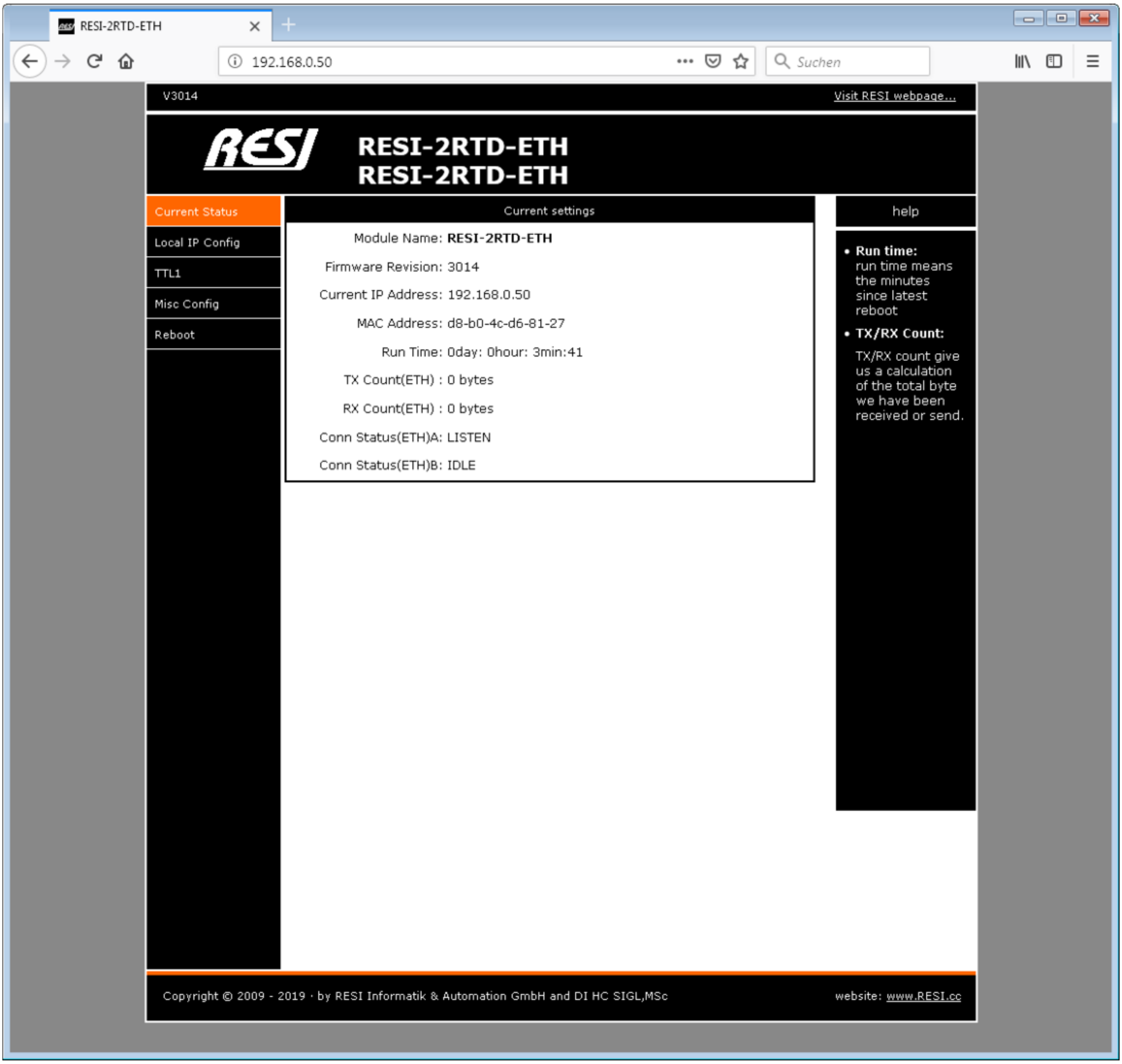
Figure: MODBUS/TCP SERVER MODE for the RESI-xxx-ETH gateway

8.4 RESI-xxx-ETH WEB CONFIGURATION

All of our RESI-xxx-ETH gateways have an integrated web server that configures basic access to the Ethernet interface and reads out MODBUS/TCP. To do this, open an Internet Explorer and enter the configured IP address of the selected gateway.

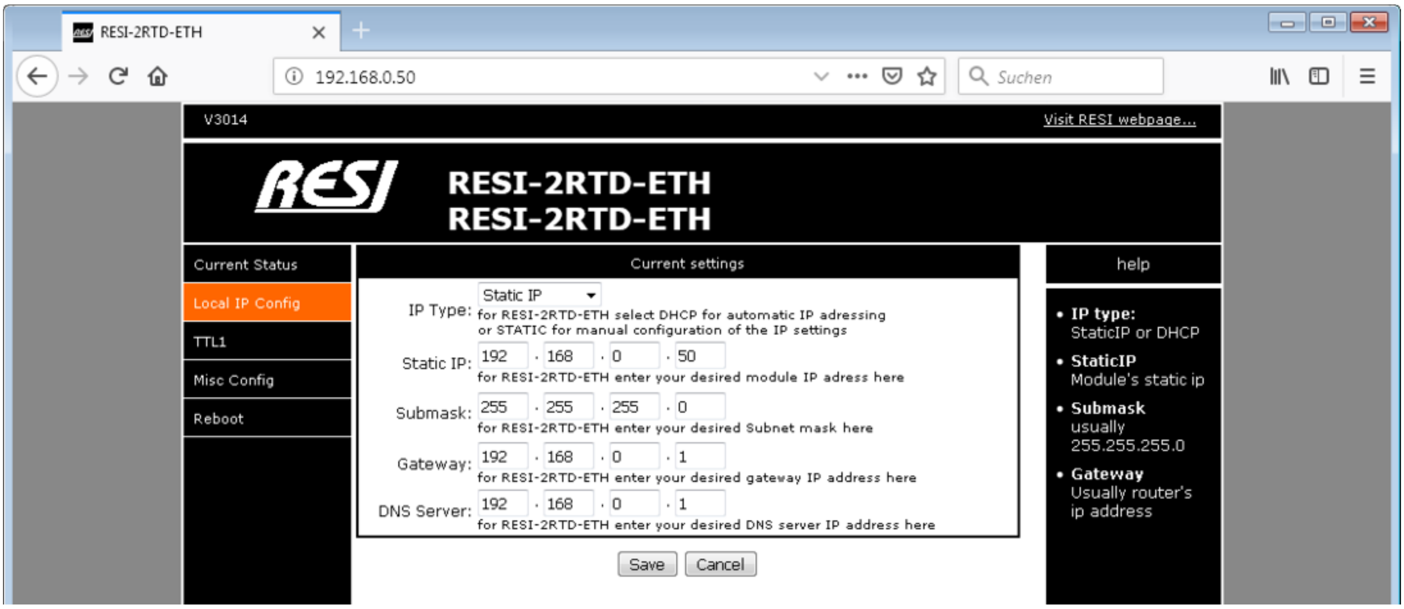
HINT: Please check the individual section of each IO module for the standard IP settings of our specific module.
The default user name is RESI and the default password is RESI.

You should see the following page:



8.4.1 How to set up the IP address

Select the "Local IP Config" page. Use the following mask to edit the IP settings:

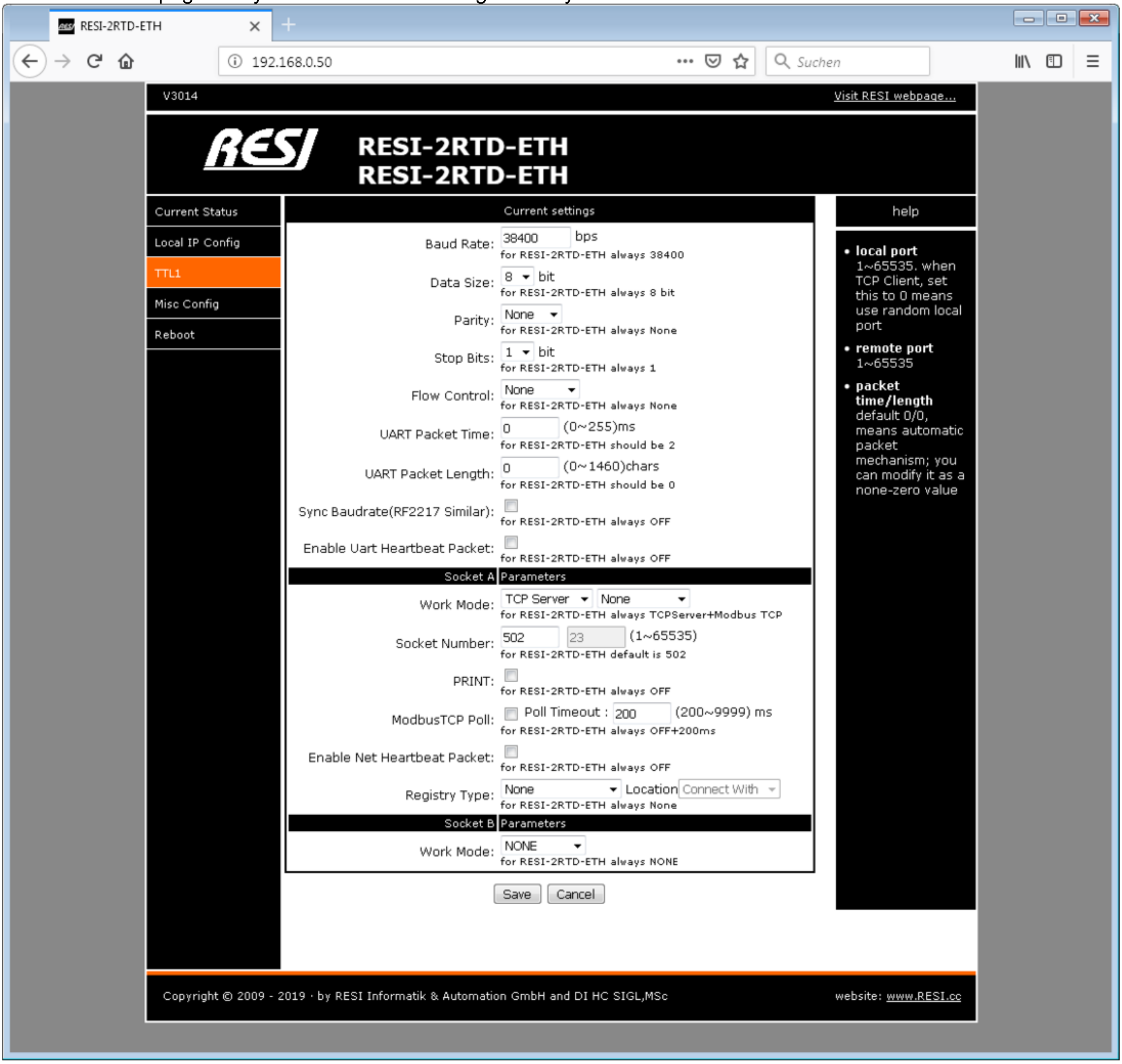


- **IP type**: Selection between STATIC IP for a static IP address or DHCP mode for an automatic assignment of the IP address.
- **Static IP**: Choose your desired IP address in IPv4 format
- **Submask**: Select your desired subnet mask in IPv4 format
- **Gateway**: Select your desired gateway IP address in IPv4 format
- **DNS server**: Select your desired DNS server IP address in IPv4 format

Click SAVE to save your data. But don't forget to restart the device for the new IP settings to take effect. If you have problems, set the CFG DIP switch to ON and restart the device. Wait for more than 30 seconds. The gateway resets to the factory settings with the IP standard settings defined above. Don't forget to set the CFG DIP switch back to OFF afterwards.

8.4.2 How to change the socket number

Select the TTL1 page and you will see the following view in your web browser.



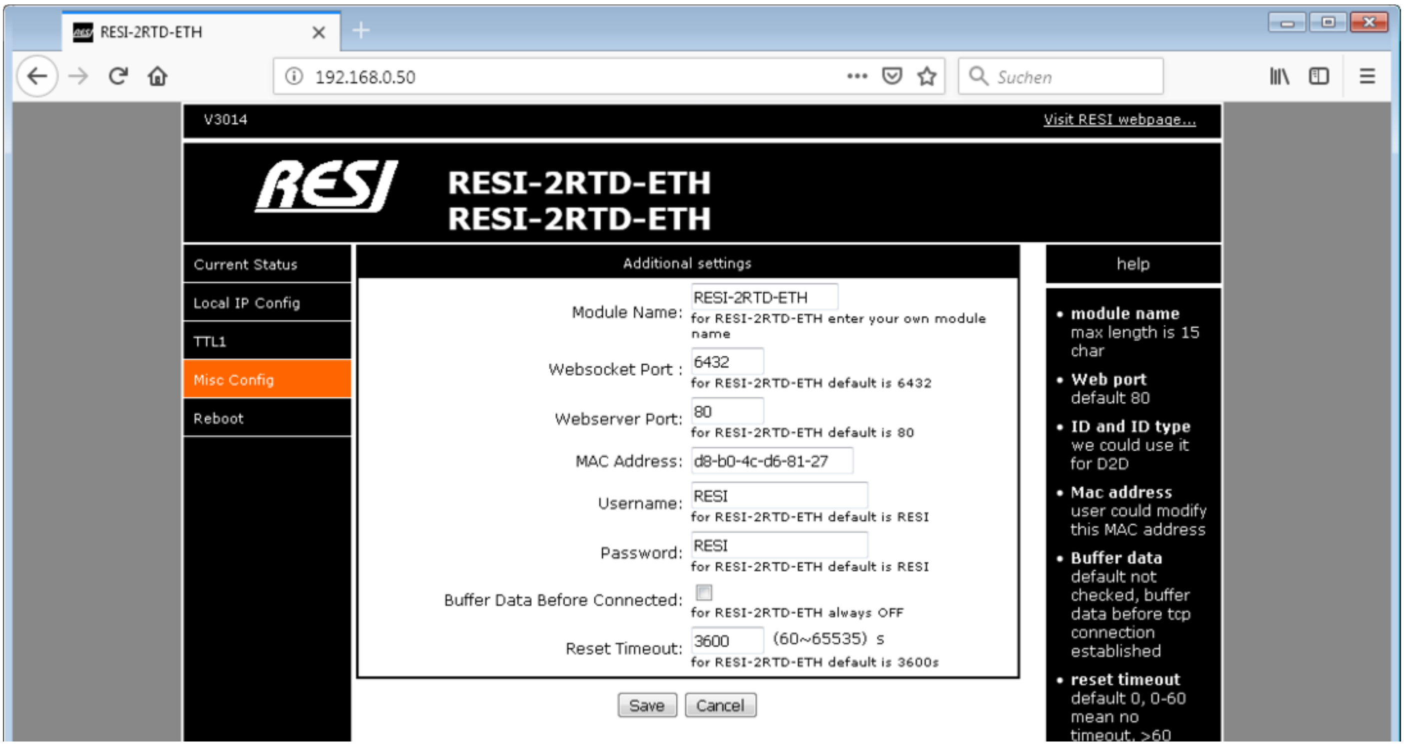
NOTE: Do not change the TTL communication parameters (e.g. baud rate, ...). You can lose the connection to the gateway!

- **Work mode**: Here you can select TCP Server/none if you want to communicate in transparent mode. All incoming data on the socket are forwarded directly to the IO module. If you want to use the internal MODBUS / TCP to MODBUS/RTU converter, you have to select TCP Server/ModbusTCP. If you select TCP-Server/none, you can also communicate with the MODBUS/RTU protocol over Ethernet or use the ASCII protocol.
- **Socket number**: Here you can select the desired socket number that you want to use for the Ethernet connection. The default value for our converters is 1024, for MODBUS/TCP 502.

Please leave the rest of the parameters unchanged. They are only suitable for experts!

8.4.3 How to change username and password

If you select the Misc config page, you will see the currently configured user name and password. You will also see the current module name.

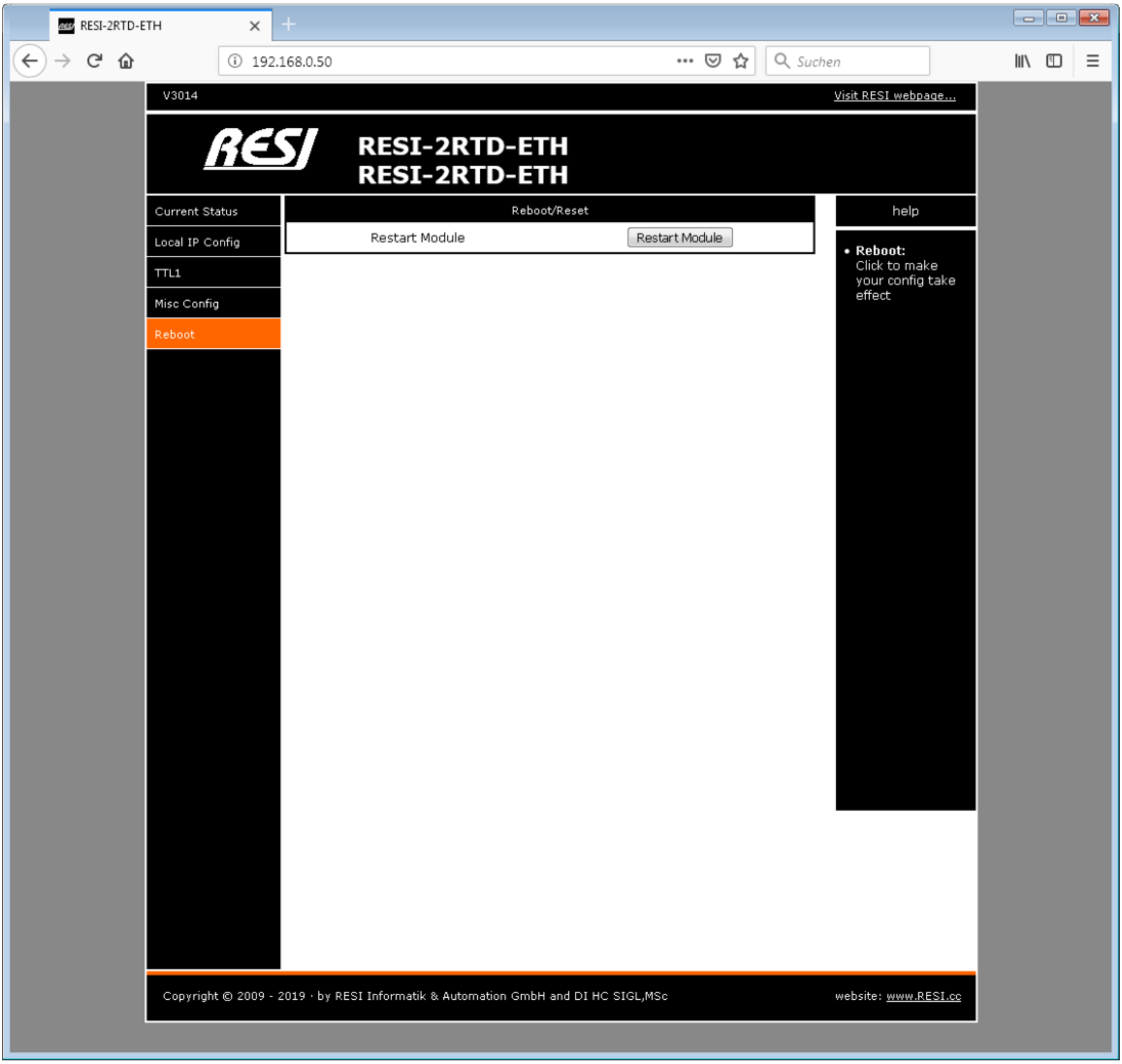


- Module name: Here you can enter a new module name. It is used for better identification if you have more than one gateway in your network.
- Username: Here you can enter a new user name for accessing the web configuration.
- Password: Here you can enter a new password for accessing the web configuration.

Don't forget to save the new settings with the SAVE button!
Please leave the rest of the parameters unchanged. These are only for experts!

8.4.4 How to restart the gateway via Ethernet

First select the Reboot page. Then select the Restart Module button to restart the software.

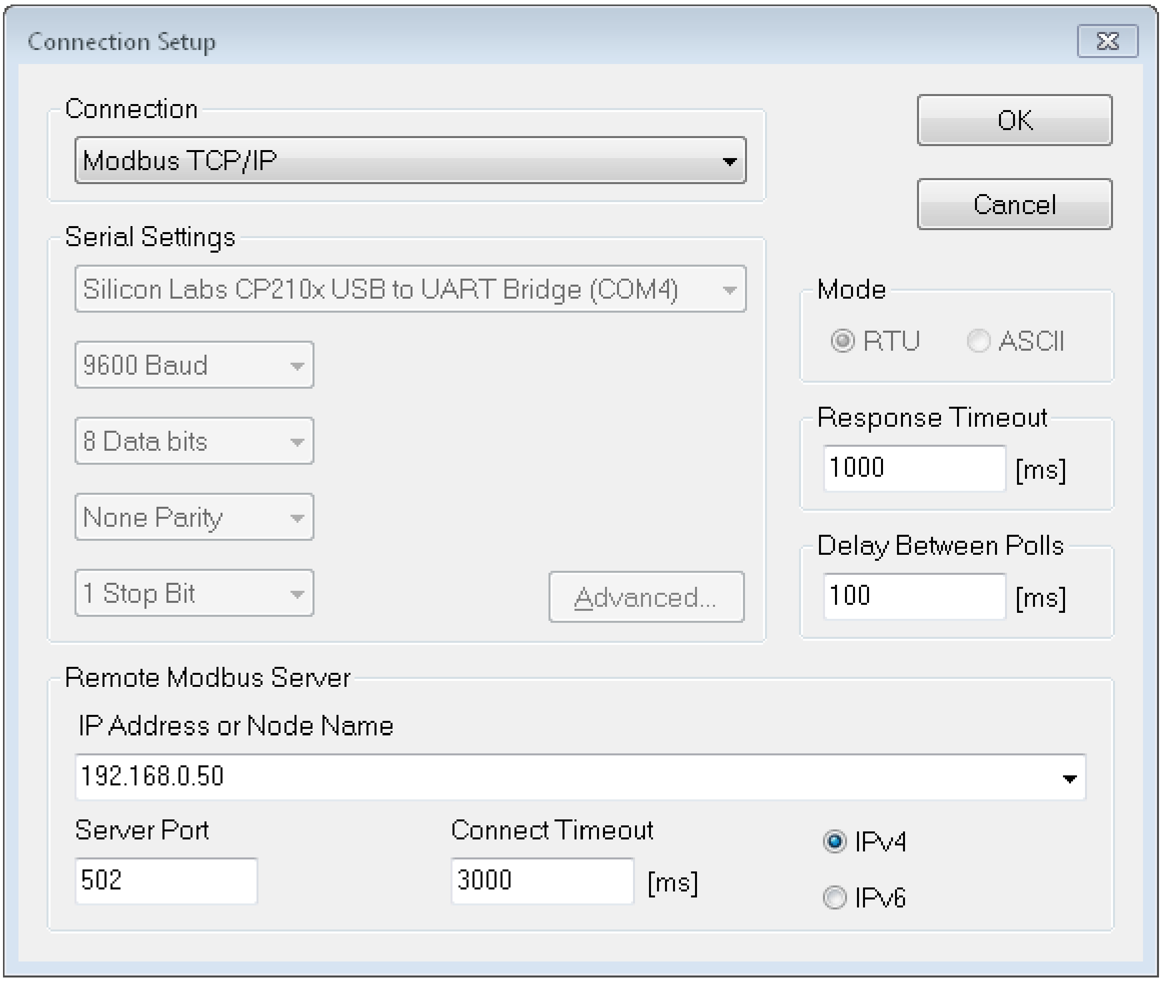


8.4.5 How to select the MODBUS / TCP server mode

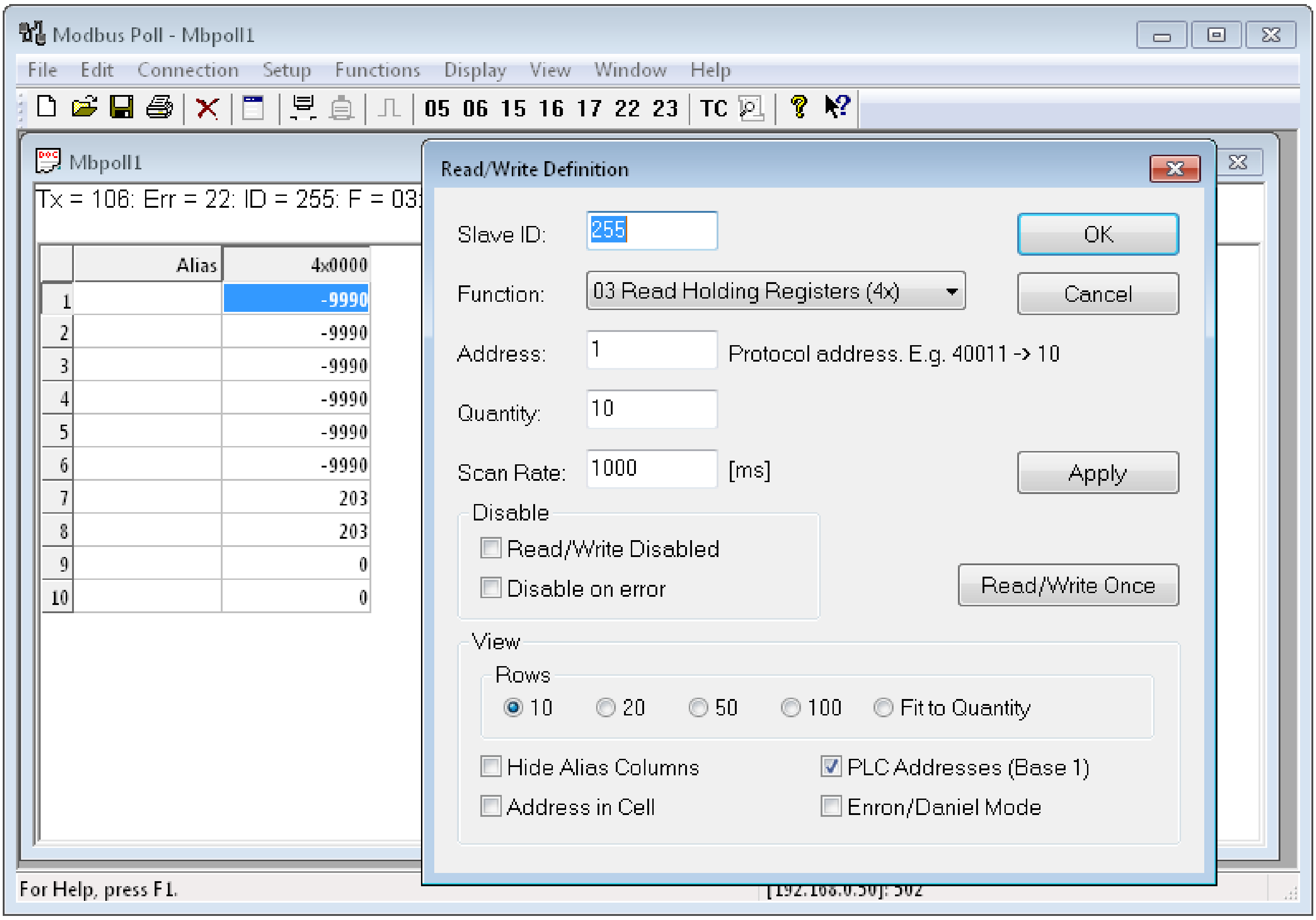
A gateway can be switched to one of the following states very quickly:

1. Activate DIP switch 4: CFG
2. Wait for about 30 seconds. The gateway LEDs flash very quickly
3. Deactivate all DIP switches

Now you have reset the factory settings to the IP standard settings and selected the MODBUS/TCP server mode.
To test your MODBUS/TCP server functionality, use the MODBUS Poll software shown here:



Now select the area of the MODBUS-Holding register you want to display. Select the function Setup/Read-Write Definition .. and configure the following parameters. After you click OK, the updated values are displayed.

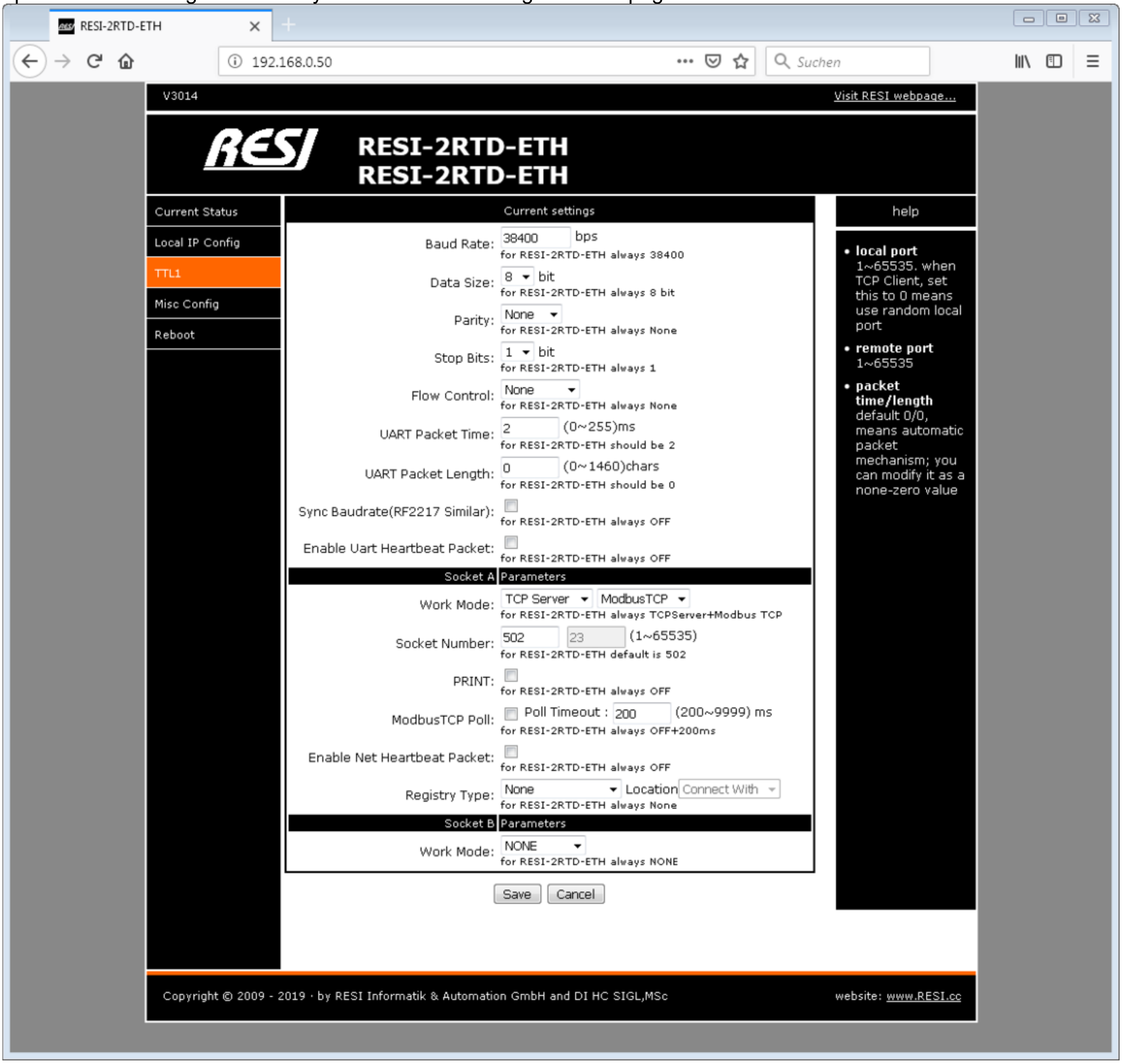


8.4.6 How to select the TRANSPARENT or MODBUS/RTU via ETHERNET mode

A gateway can be switched to one of the following states very quickly:

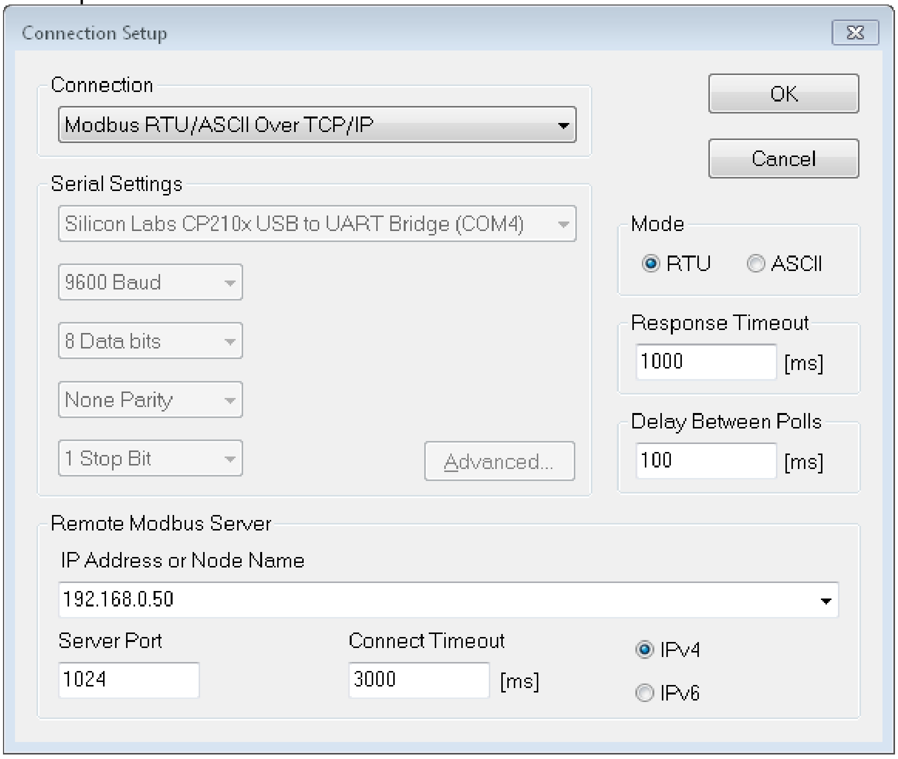
1. Activate DIP switch 4: CFG
2. Wait for about 30 seconds. The gateway LEDs flash very quickly
3. Deactivate all DIP switches

Now you have reset the factory settings to the IP standard settings and selected the MODBUS/TCP server mode. Now open the Web configuration with your browser and navigate to the page shown below:

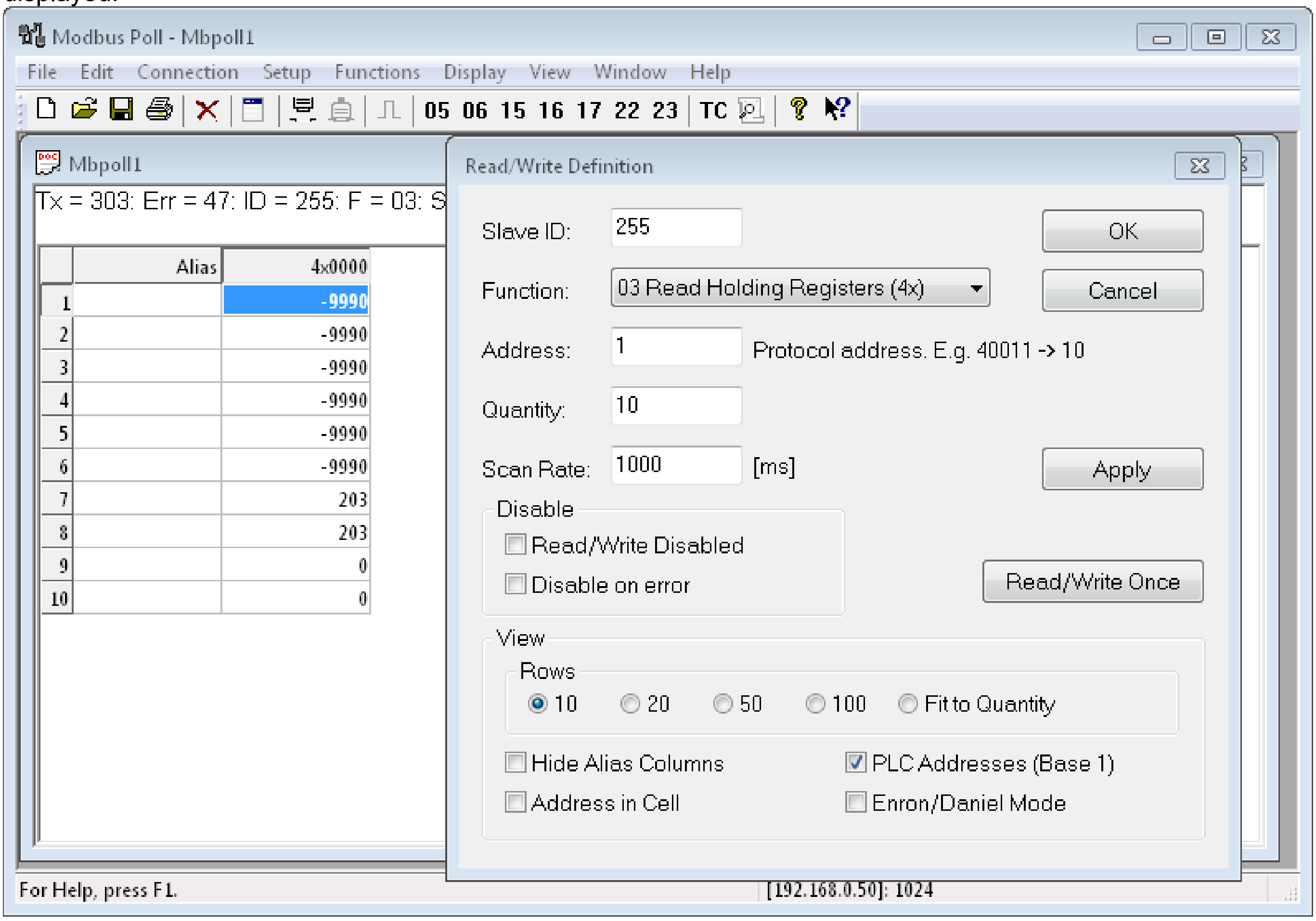


Now change the **Work Mode** from **MODBUS/TCP** to **None** and adapt the socket number to your needs. (e.g. 1024). Click SAVE and restart the module with the RESTART button. Now the module works in TRANSPARENT mode.

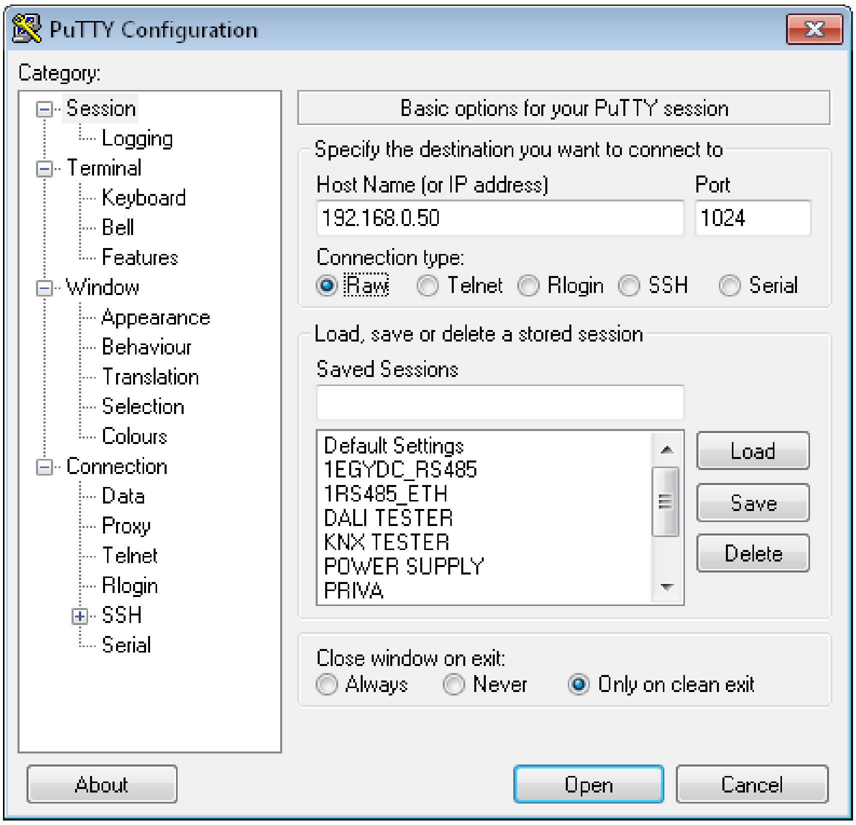
Now open the MODBUS Poll software to test the MODBUS/RTU via the Ethernet mode:



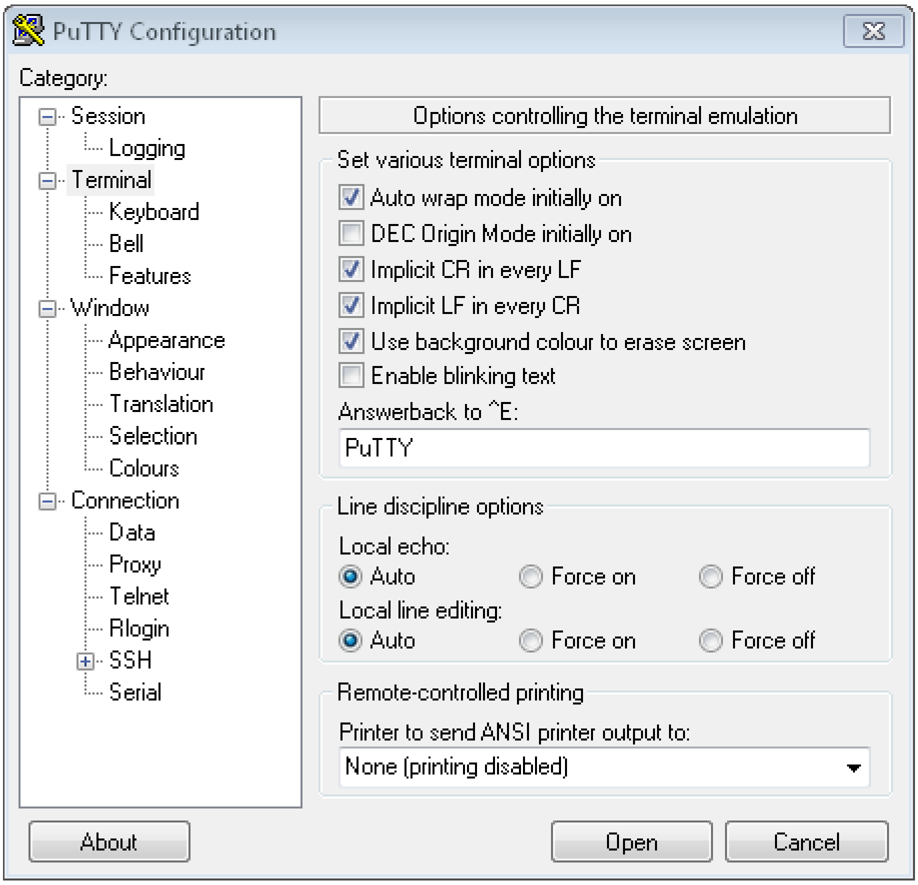
After you have established a connection, set the MODBUS read parameters to your needs. Select the function Setup / Read-Write Definition .. and configure the following parameters. If successful, the following values should be displayed:



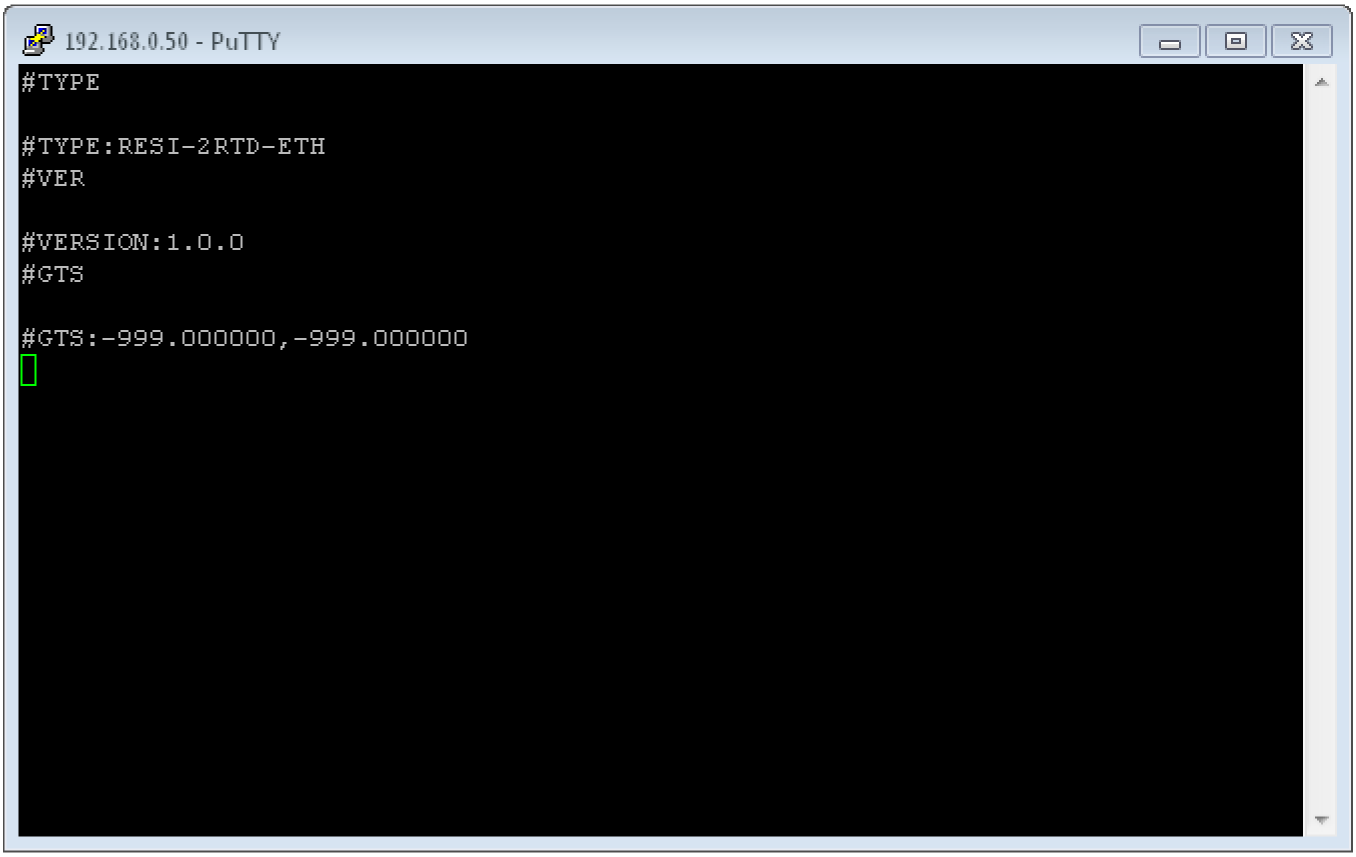
To test the ASCII protocol, use the freeware tool putty to establish a socket connection to the module. Configure your IP settings as follows:



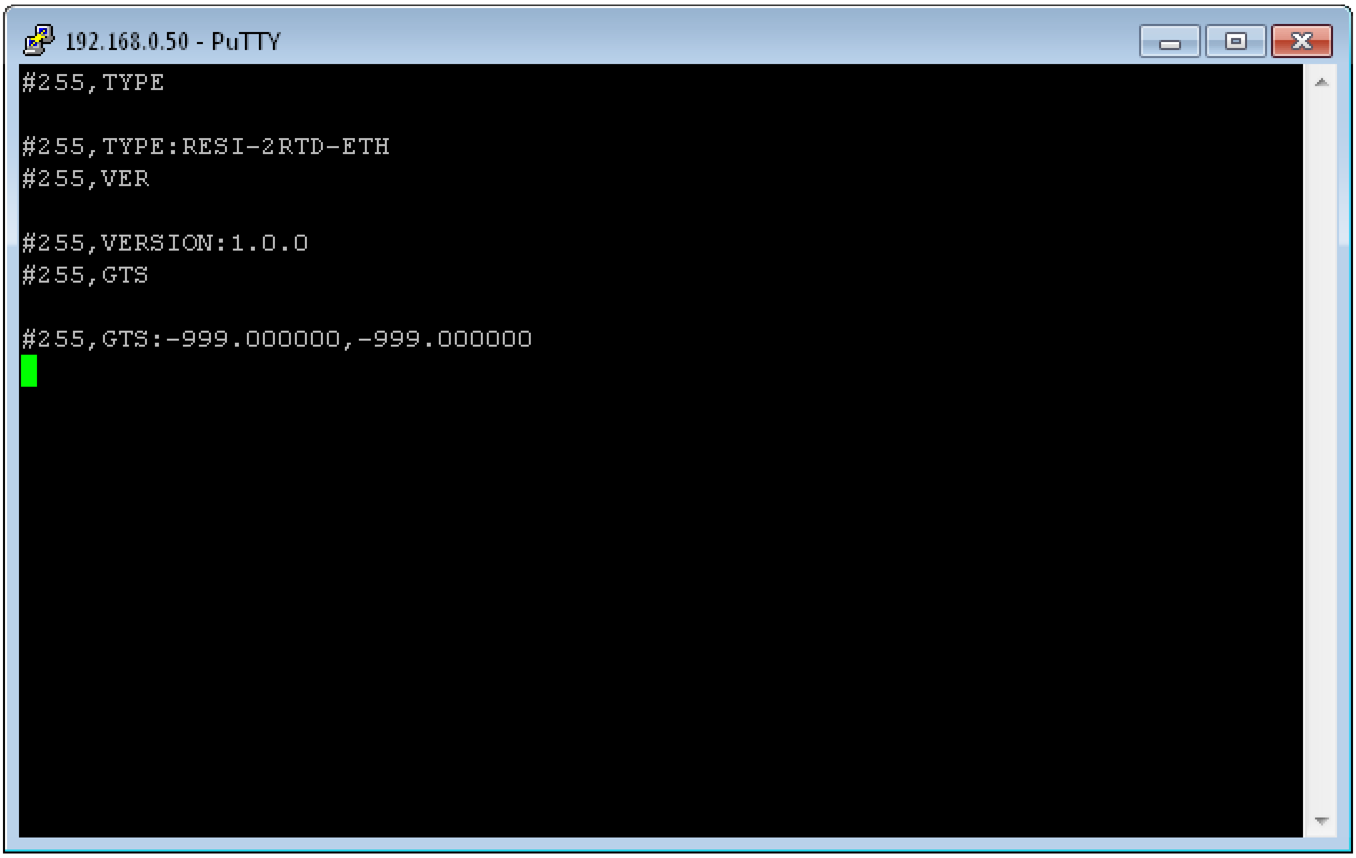
Then we have to change the behavior of the PUTTY terminal emulation. Click Terminal in the tree on the left and change the settings to the parameters shown below:



Click Open to establish a socket connection to the module. Enter the first command #TYPE <CR>. The IO module responds with the current module type.



You can also use the UnitID of the IO module in this protocol:

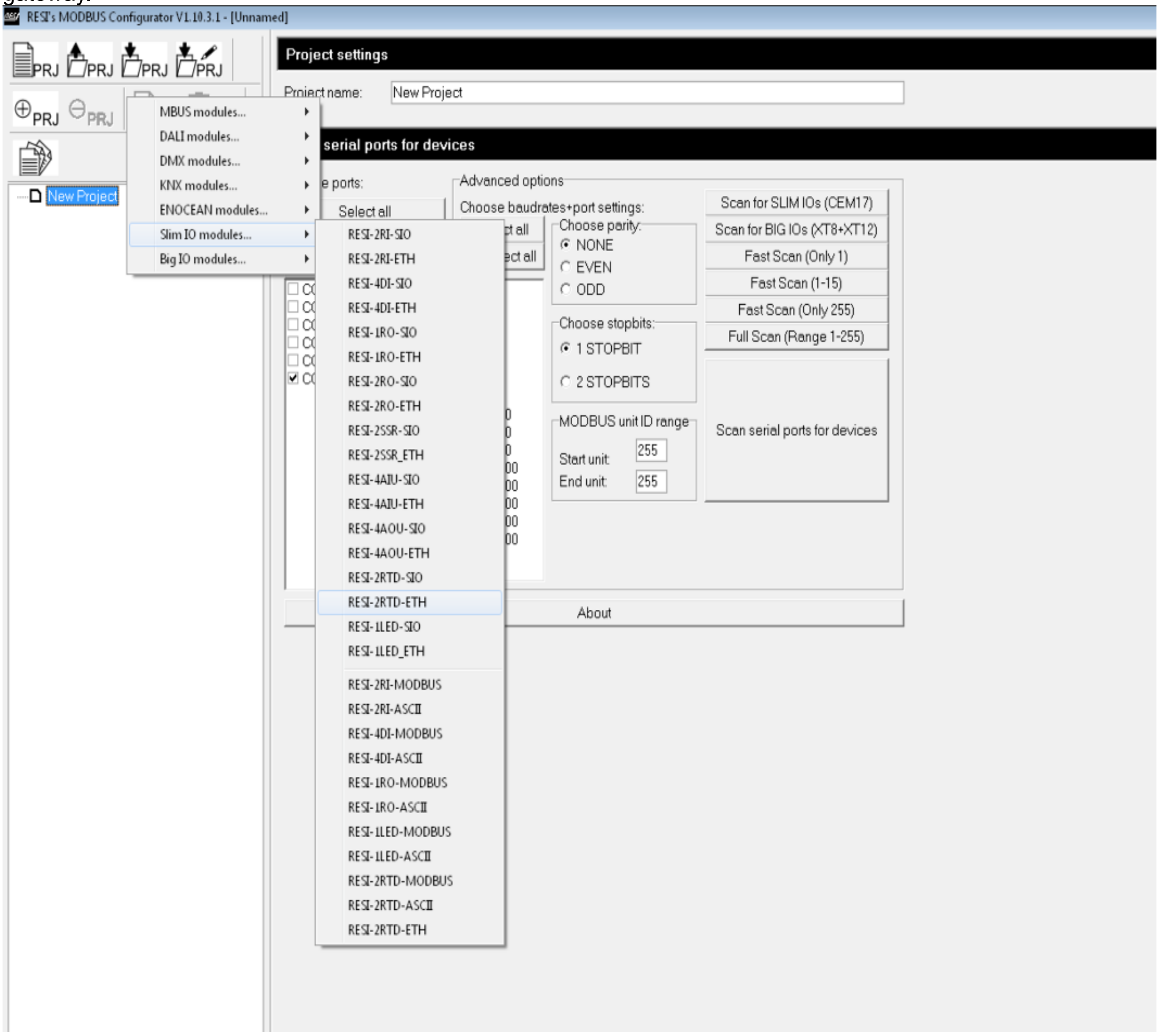


8.5 HOWTO connect to an Ethernet gateway

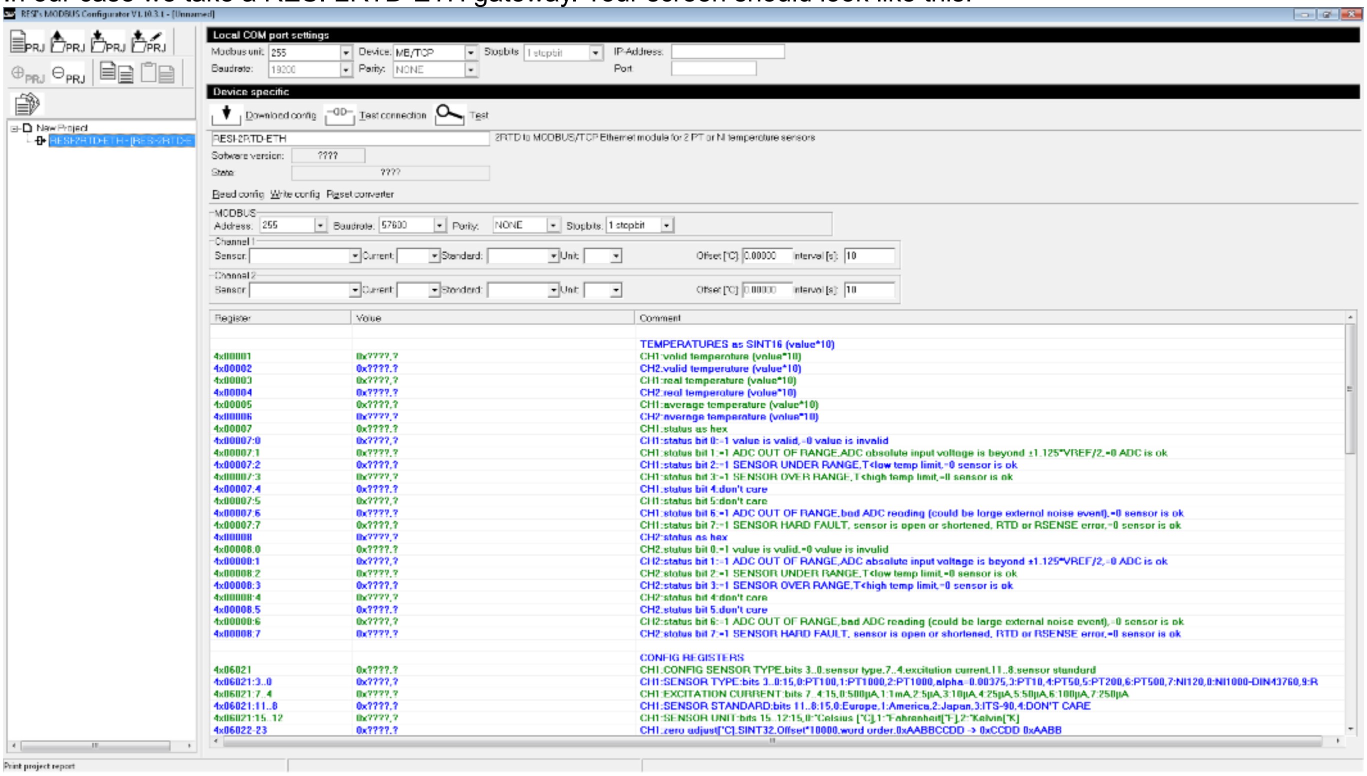
Follow these steps for communicating with an Ethernet gateway.

8.5.1 Example: Add RESI-2RTD-ETH to project tree

First, start the MODBUSConfigurator software. Click on the project tree title “New Project” and add a desired Ethernet gateway.

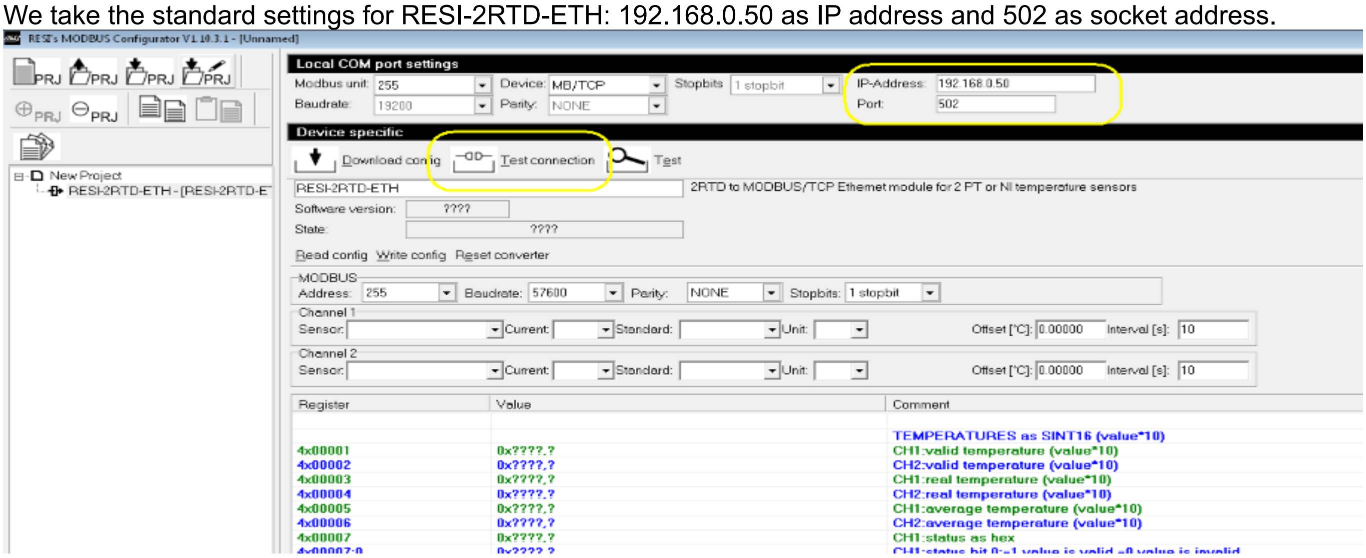


In our case we take a RESI-2RTD-ETH gateway. Your screen should look like this:



8.5.2 Enter IP address & socket port

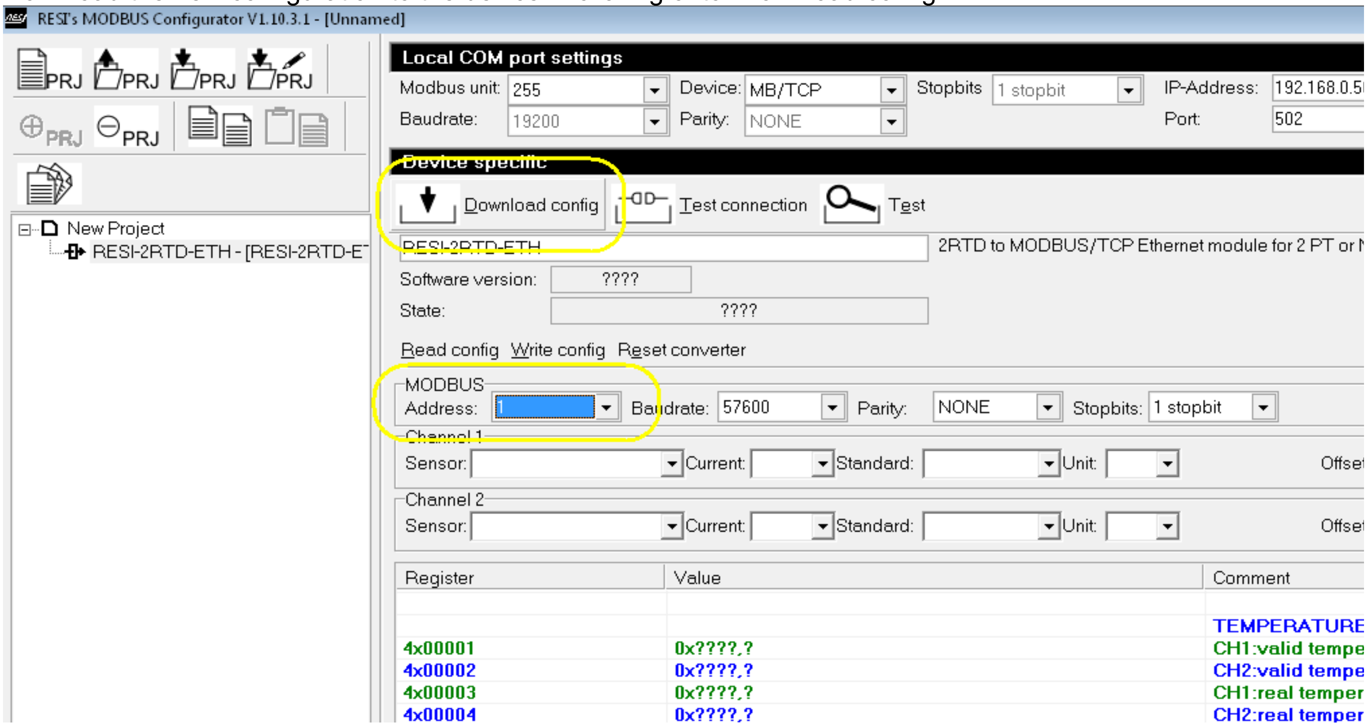
You will notice, that the software automatically suggest as a device “MB/TCP”. Now you have to enter the via web configuration defined IP address and socket number for the communication via MODBUS/TCP protocol.
We take the standard settings for RESI-2RTD-ETH: 192.168.0.50 as IP address and 502 as socket address.



Click on the button “Test connection”. The software should display after a short test: connection test successful.

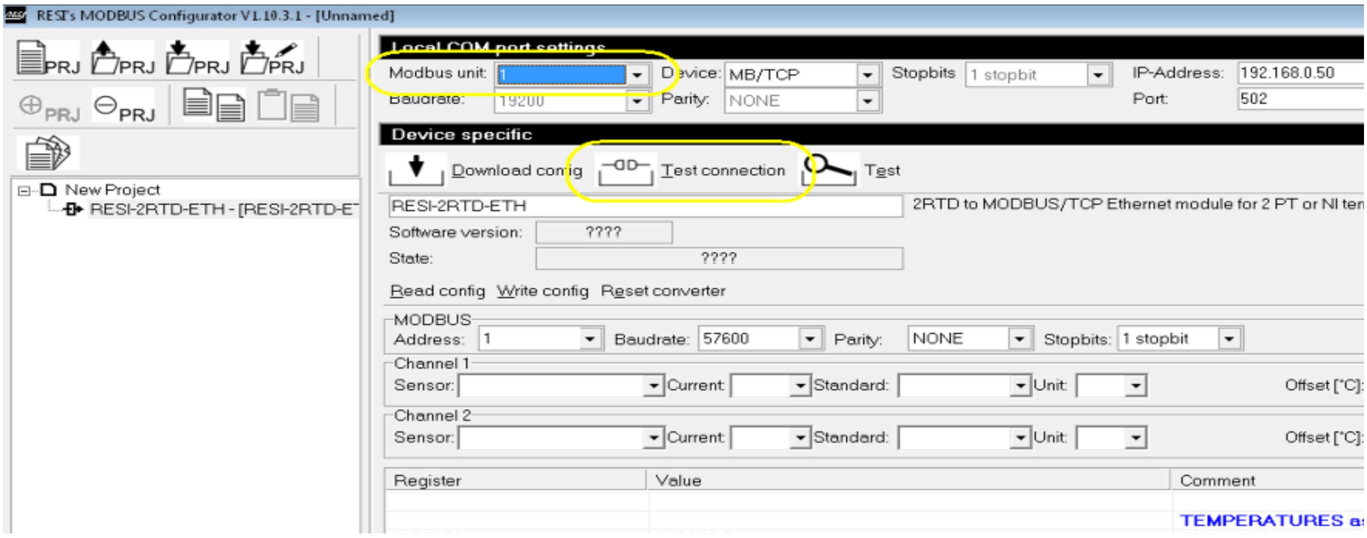
8.5.3 Change MODBUS unit ID to your needs

Now you can change the MODBUS address to your needs. We use 1 as a Unit ID for MODBUS/TCP communication. Download the new configuration to the device in clicking onto “Download config”.



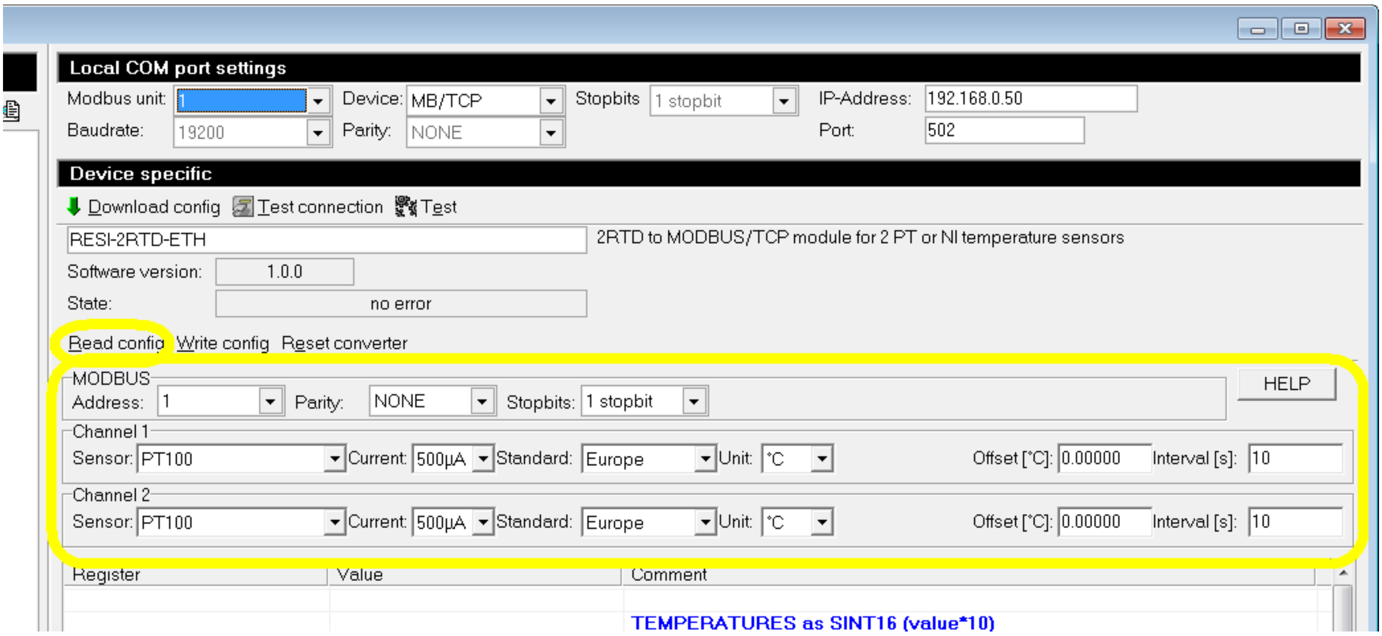
8.5.4 After Download config, change local COM port settings

The next step is to change the MODBUS Unit ID in the Local COM port settings to 1. Check the new settings with the function “Test connection”.

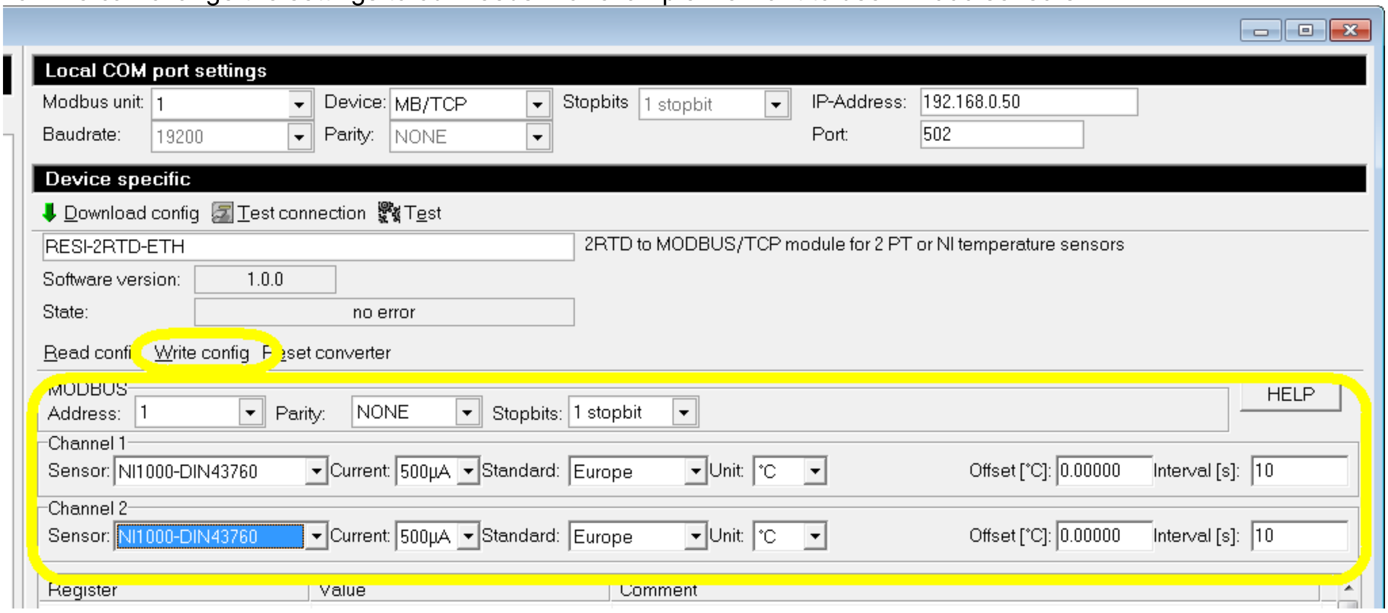


8.5.5 Read sensor configuration

Now we read out the current sensor configuration of the IO module. Click on Read config. Your display should look like this:

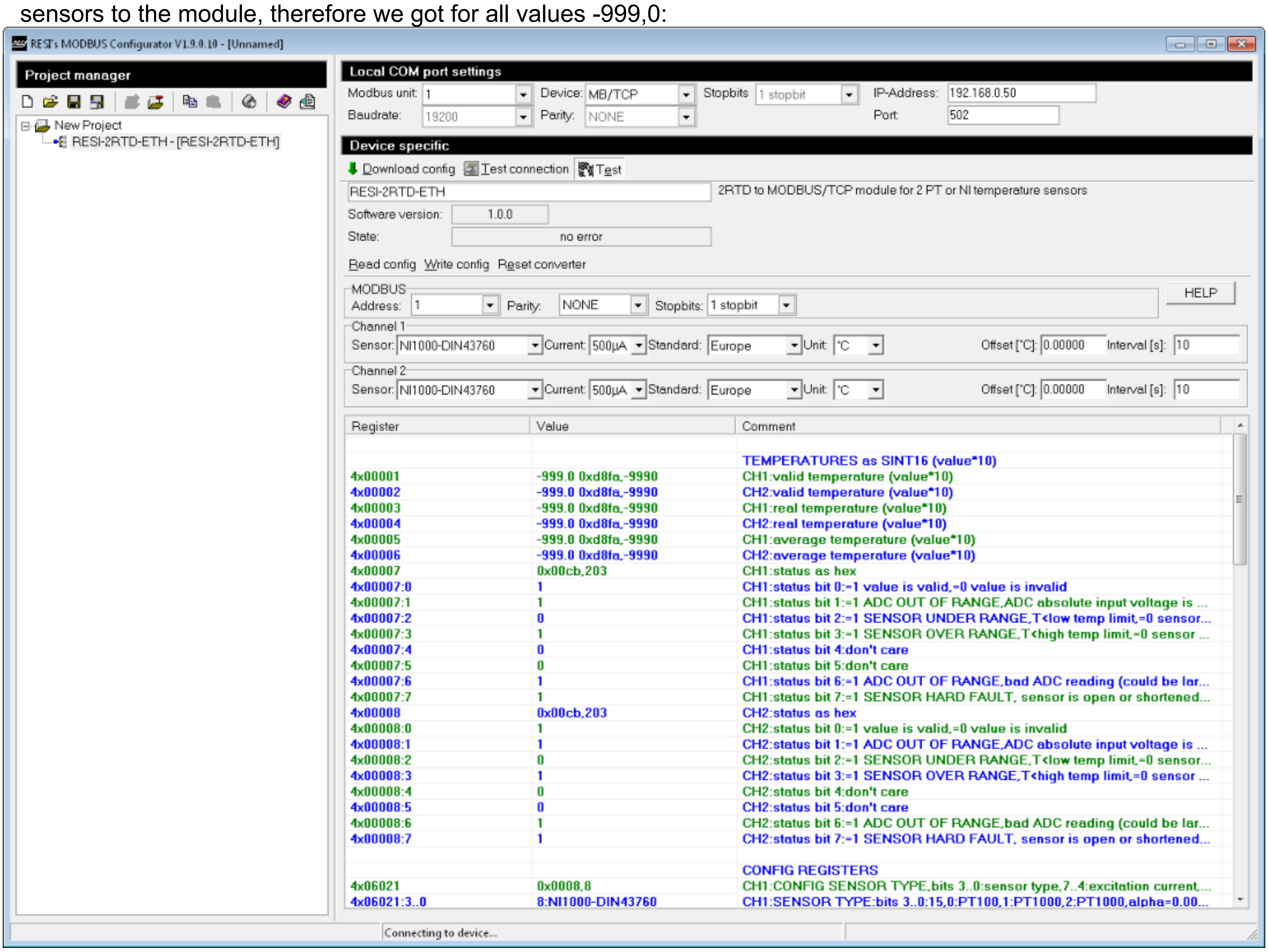


Now we can change the settings to our needs. For example we want to use NI1000 sensors:



8.5.6 Test the configuration

After a successful download we activate the test function. (Don’t forget to select the correct Unit ID in the Local COM port settings. Otherwise you will get no connection). You should get the following result (We have not connected any sensors to the module, therefore we got for all values -999,0:



9 DIP switch settings

Our ULTRA SLIM IO module offer a 4 pin DIP switch for initial setup of the serial connection or the Ethernet connection. Our BIGIO modules offer an 8 pin DIP switch for initial setup.

9.1 DIP switch for serial ULTRA SLIM IOs

The following drawings show the DIP switches for all of our serial SLIMIO products:

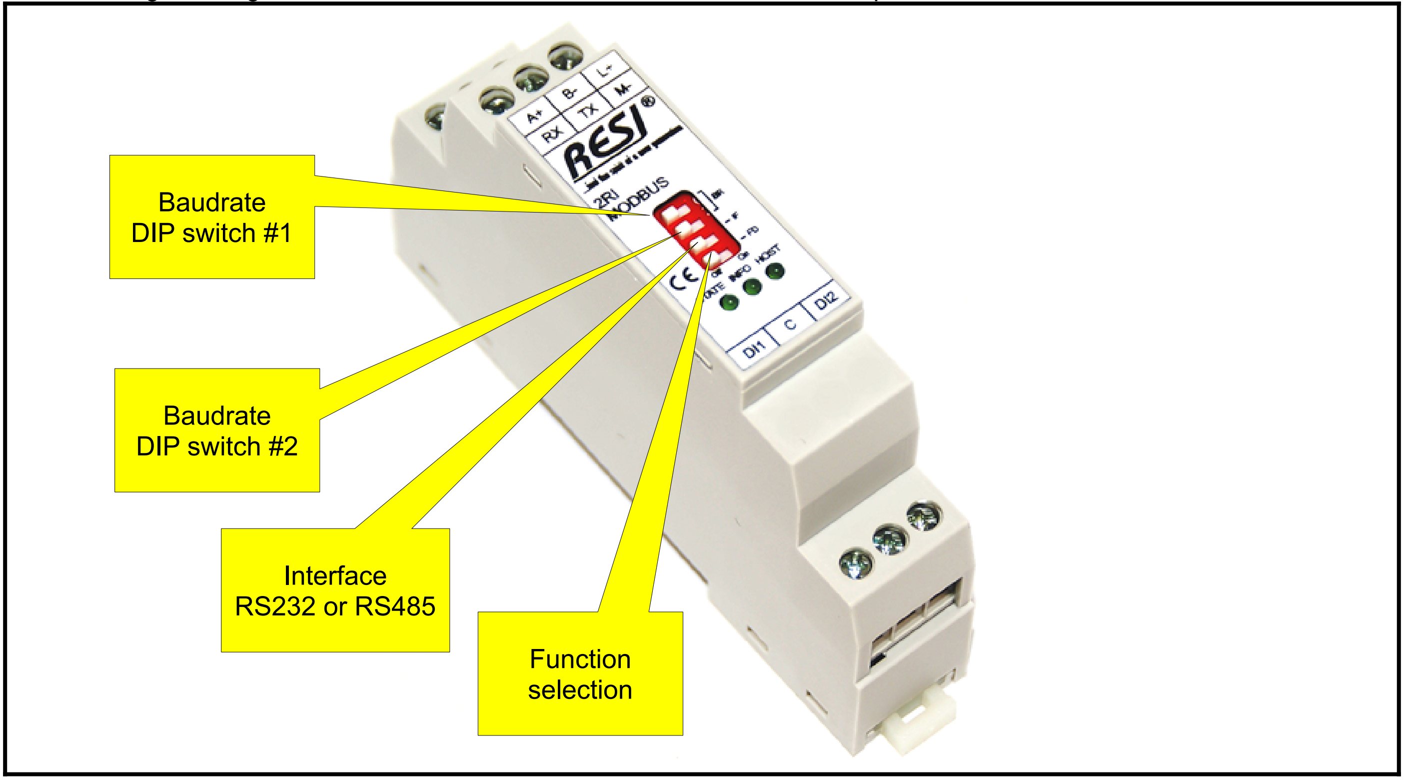


Figure: DIP switches for our serial ULTRA SLIM IO modules

Baud rate BR

Use DIP switch 1 + 2 to select the baud rate:

OFF	OFF:	9600Bd
ON	OFF:	19200Bd
OFF	ON:	38400Bd
ON	ON:	from FLASH (normally 57600Bd)

NOTE: The correct parity (NONE, EVEN, ODD) and the stop bits are set via the configuration software, not via the DIP switches! Likewise, the baud rate for the DIP switch position BR=ON,ON is set via the configuration software.

Interface IF

Selects the physical type of the serial interface for the ASCII or MODBUS/RTU protocol:

OFF=RS232
ON=RS485

Function selection FD	Selects a special function: OFF=The unit ID from the FLASH is used ON=Unit ID 255 is always used
NOTE	After changing the DIP switch, the device will be booted automatically So no voltage off/voltage one cycle is necessary. After restarting, all LEDs flash briefly to represent the restart sequence.

9.2 DIP switch for Ethernet ULTRA SLIM IOs

The following drawings show the DIP switches for all of our Ethernet SLIMIO products:

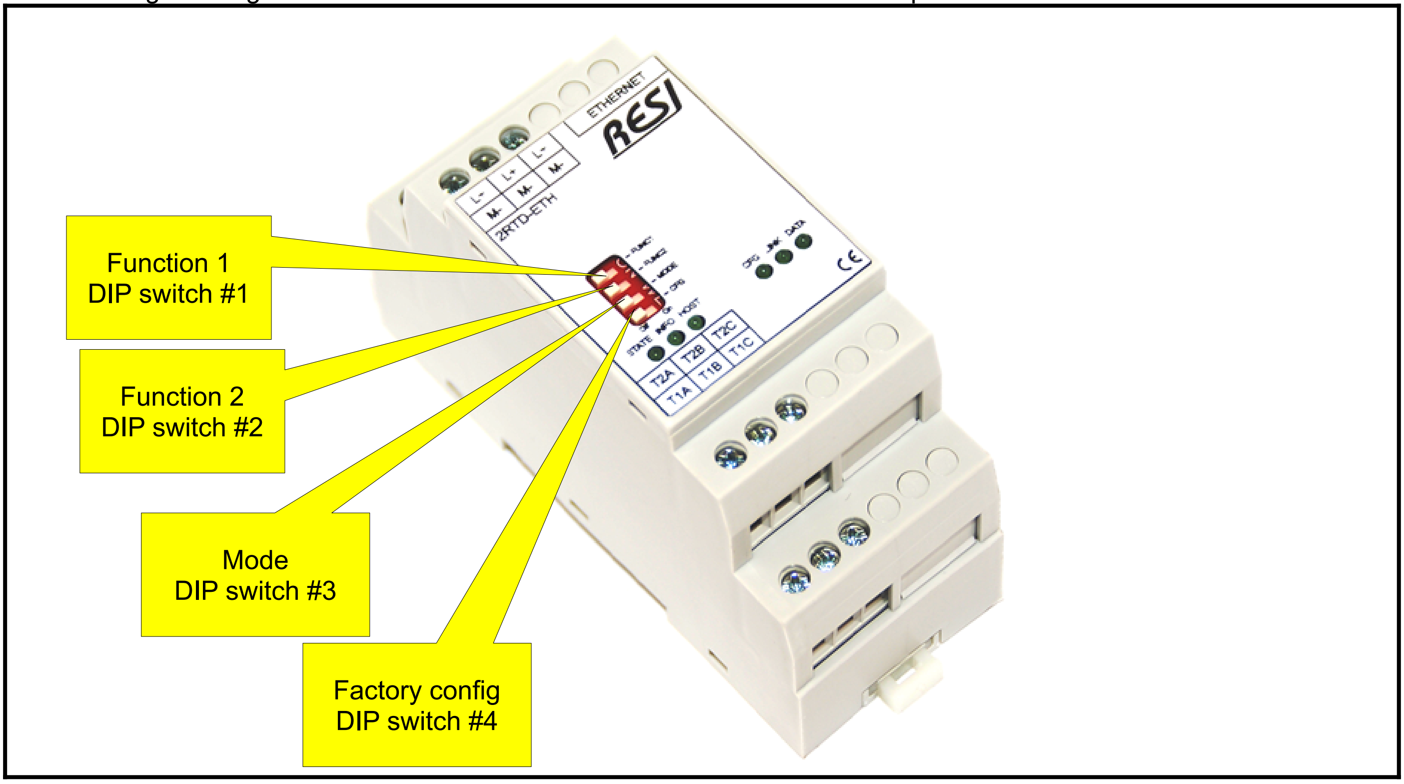


Figure: DIP switches for our Ethernet ULTRA SLIM IO modules

FUNC1
configuration

ON: When the module is restarted, the module changes to STATIC IP with the standard IP settings
OFF: The current IP settings are used

FUNC2
configuration.

ON: When the module is restarted, the module changes to DHCP IP
OFF: The current IP settings are used

MODE

While resetting to factory settings (CFG=ON):
OFF: Socket mode is set to MODBUS/TCP Socket
ON: Socket mode is set to MODBUS/RTU or ASCII over Ethernet

In normal operation:
OFF: The configured UnitID is used
ON: UnitID 255 is always used!

CFG

ON: When the module restarts, the module restores the factory settings.
Wait for about 30 seconds until the STATE + CFG LEDs blink quickly.
Then set all DIP switches to OFF.
The module restarts automatically and is ready for use.
OFF: Normal start of the module

NOTE

After changing a DIP switch, the module restarts immediately.
After restarting, all LEDs are briefly switched on to visually indicate the restart of the device.

9.3 DIP switch for serial BIG IOs

The following drawings show the DIP switches for all of our serial BIGIO products:

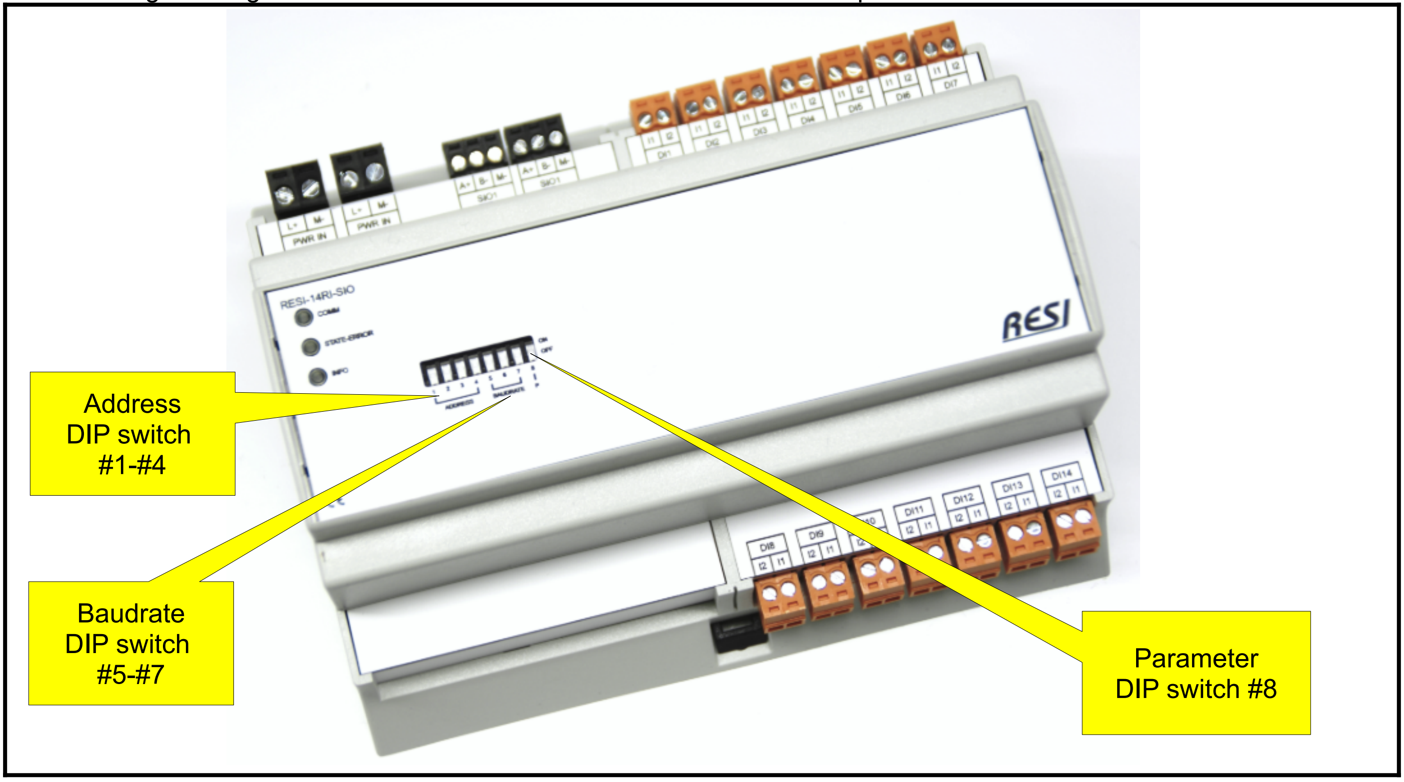


Figure: DIP switches for our serial BIG IO modules

The 8 pin DIP switch has the following mapping:

DIP SWITCH

1=ADR0
2=ADR1
3=ADR2
4=ADR3
5=BR0
6=BR1
7=BR2
8=PARAMETER

ADDRESS This four DIP switches ADR3-ADR0 create the MODBUS/RTU unit number or ASCII bus address in the range of 0 to 15. You can use the following settings:

ADR3	ADR2	ADR1	ADR0	MODBUS/RTU unit number or ASCII bus number
OFF	OFF	OFF	OFF	Internal MODBUS/RTU unit number is used from the FLASH memory in the range of 0 to 255.
OFF	OFF	OFF	ON	1
OFF	OFF	ON	OFF	2
OFF	OFF	ON	ON	3
OFF	ON	OFF	OFF	4
OFF	ON	OFF	ON	5
OFF	ON	ON	OFF	6
OFF	ON	ON	ON	7
ON	OFF	OFF	OFF	8
ON	OFF	OFF	ON	9
ON	OFF	ON	OFF	10
ON	OFF	ON	ON	11
ON	ON	OFF	OFF	12
ON	ON	OFF	ON	13
ON	ON	ON	OFF	14
ON	ON	ON	ON	15

BAUD RATE Those three DIP switches BR2-BR0 define the MODBUS/RTU or ASCII baud rate for the communication:

BR2	BR1	BR0	MODBUS/RTU or ASCII baud rate
OFF	OFF	OFF	4800bd
OFF	OFF	ON	9600bd
OFF	ON	OFF	19200bd
OFF	ON	ON	38400bd
ON	OFF	OFF	57600bd
ON	OFF	ON	115200bd
ON	ON	OFF	230400bd
ON	ON	ON	256000bd

PARAMETER This DIP switch selects between the configuration via DIP switch or via FLASH parameter for the serial setup.

=0: The selected UnitID, baud rate from the DIP switch settings are used.

The parity is NONE and the one stop bit is used

=1: The selected UnitID from the DIP switches is used, but the serial parameters are taken from the FLASH parameters.

Baud rate can be selected between 300 to 256000 Baud.

Parity can be NONE, EVEN or ODD.

Stop bits can be ONE or TWO.

NOTE After changing the DIP switch, the device will be booted automatically So no voltage off/ voltage one cycle is necessary. After restarting, all LEDs flash briefly to represent the restart sequence.

9.4 DIP switch for Ethernet BIG IOs

The following drawings show the DIP switches for all of our serial BIGIO products:

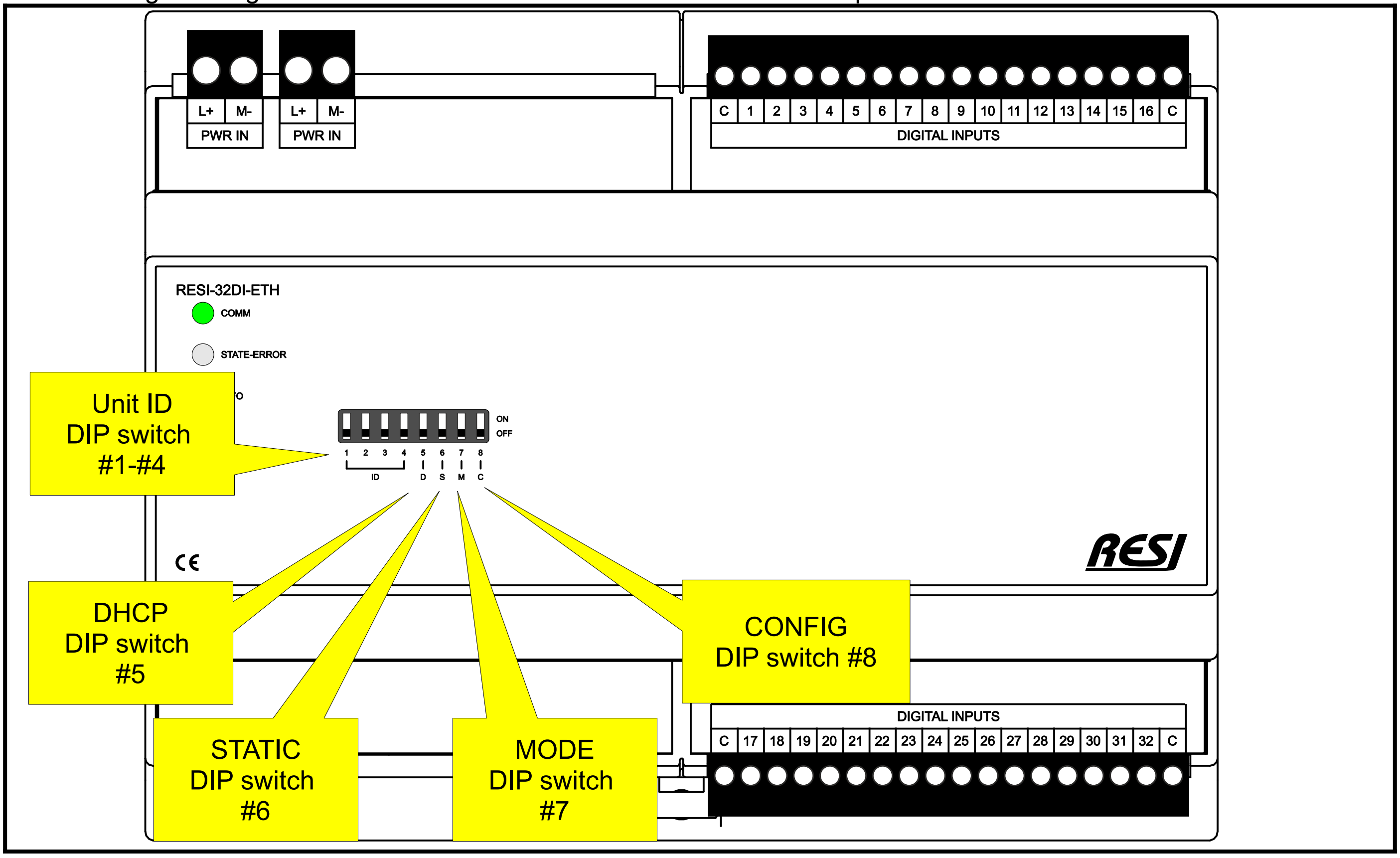


Figure: DIP switches for our serial BIG IO modules

The 8 pin DIP switch has the following mapping:

DIP SWITCH

1=ID0
2=ID1
3=ID2
4=ID3
5=D
6=S
7=M
8=C

ID This four DIP switches ID3-ID0 create the MODBUS unit number or ASCII bus address. You can use the following settings:

1=ID0
2=ID1
3=ID2
4=ID3

ID3	ID2	ID1	ID0	MODBUS unit number or ASCII bus number
OFF	OFF	OFF	OFF	255
OFF	OFF	OFF	ON	1
OFF	OFF	ON	OFF	2
OFF	OFF	ON	ON	3
OFF	ON	OFF	OFF	4
OFF	ON	OFF	ON	5
OFF	ON	ON	OFF	6
OFF	ON	ON	ON	7
ON	OFF	OFF	OFF	8
ON	OFF	OFF	ON	9
ON	OFF	ON	OFF	10
ON	OFF	ON	ON	11
ON	ON	OFF	OFF	12
ON	ON	OFF	ON	13
ON	ON	ON	OFF	14
ON	ON	ON	ON	Internal MODBUS unit number is used from the FLASH memory in the range of 0 to 255.

DHCP
5=D

If this DIP switch is activated by boot up, the internal IP configuration is changed to DHCP settings for Ethernet.
Wait until the STATE LED flashes very fast in white. Then set all DIP switches to OFF and restart the module (power of/on cycle)!

STATIC
6=S

If this DIP switch is activated by boot up, the internal IP configuration is changed to STATIC settings for Ethernet with the default IP address of the module.
Wait until the STATE LED flashes very fast in white. Then set all DIP switches to OFF and restart the module (power of/on cycle)!

MODE
7=M

This DIP switch is used together with the DIP switch 8=C.
If this DIP switch is set to 0 (OFF) or 1 (ON) and DIP switch 8 C is set to 1 (ON) by boot up, a complete configuration and factory reset of the module is done:
The IP interface is set to STATIC ip address with the default IP address of the module, the username and password is changed to the default values and all module specific parameters are set to factory default values.
Wait until the STATE LED flashes very fast in white. Then set all DIP switches to OFF and restart the module (power of/on cycle)!
=0: For communication MODBUS/TCP protocol is used
=1: For communication MODBUS/RTU via Ethernet or ASCII protocol is used

CONFIG
8=C

This DIP switch activated the complete basic configuration of the module.
If this DIP switch is set by boot up in combination with the DIP switch 7=M the module performs a complete system reset to default values.
STATIC IP mode is used
The default IP settings of the module are set
The username and password for the web page is set to defalt values
Depending on DIP switch 7=M, MODBUS/TCP or MODBUS/RTU via Ethernet or ASCII protocol mode is activated
Wait until the STATE LED flashes very fast in white. Then set all DIP switches to OFF and restart the module (power of/on cycle)!

9.5 DIP switches for BIG IOs RESI-S16DI8PO-SIO,RESI-S8PO-SIO

The following drawings show the DIP switches for our serial BIGIO products RESI-S16DI8PO-SIO and RESI-S8PO-SIO:

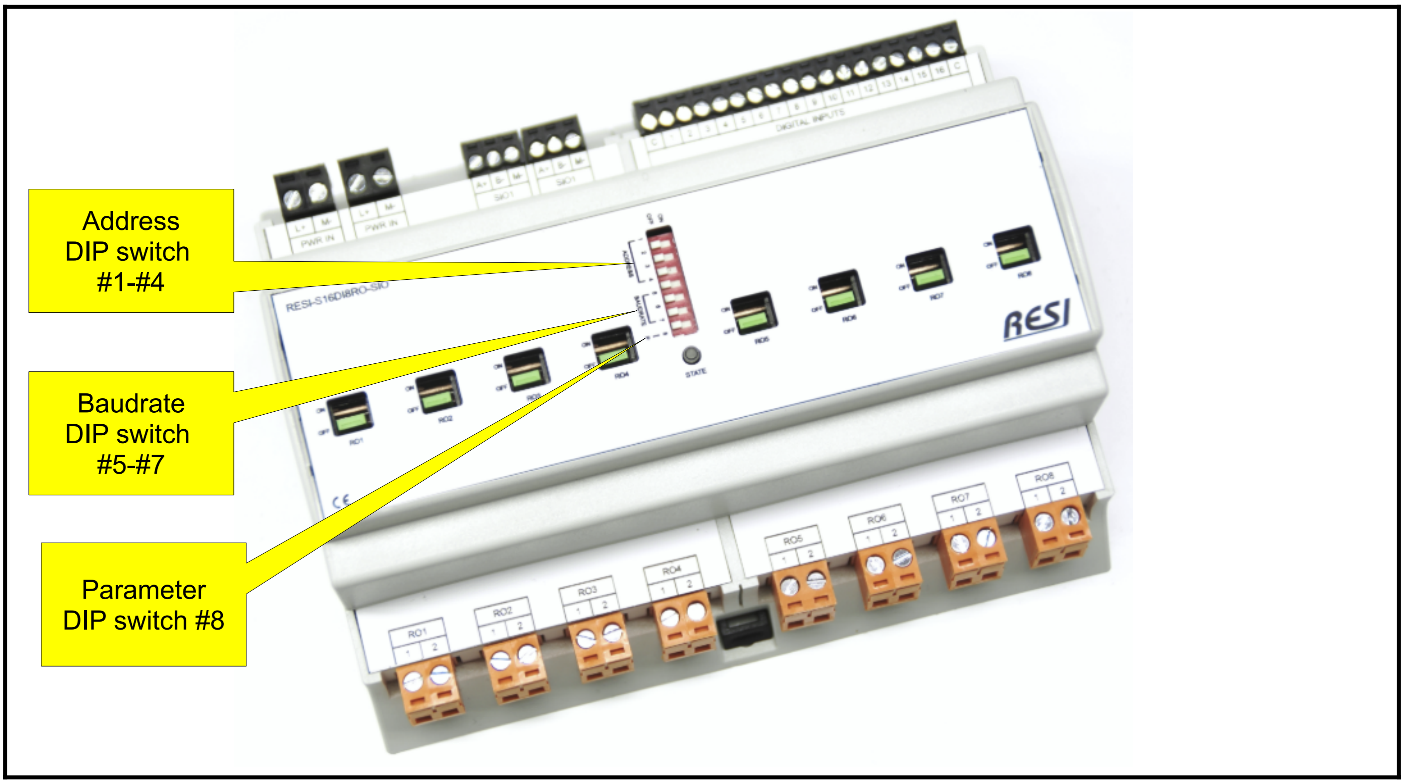


Figure: DIP switches for our BIG IO modules RESI-S16DI8PO-SIO, RESI-S8PO-SIO

The 8 pin DIP switch has the following mapping:

DIP SWITCH

1=ADR0
2=ADR1
3=ADR2
4=ADR3
5=BR0
6=BR1
7=BR2
8=PARAMETER

ADDRESS This four DIP switches ADR3-ADR0 create the MODBUS/RTU unit number or ASCII bus address in the range of 0 to 15. You can use the following settings:

ADR3	ADR2	ADR1	ADR0	MODBUS/RTU unit number or ASCII bus number
OFF	OFF	OFF	OFF	Internal MODBUS/RTU unit number is used from the FLASH memory in the range of 0 to 255.
OFF	OFF	OFF	ON	1
OFF	OFF	ON	OFF	2
OFF	OFF	ON	ON	3
OFF	ON	OFF	OFF	4
OFF	ON	OFF	ON	5
OFF	ON	ON	OFF	6
OFF	ON	ON	ON	7
ON	OFF	OFF	OFF	8
ON	OFF	OFF	ON	9
ON	OFF	ON	OFF	10
ON	OFF	ON	ON	11
ON	ON	OFF	OFF	12
ON	ON	OFF	ON	13
ON	ON	ON	OFF	14
ON	ON	ON	ON	15

BAUD RATE Those three DIP switches BR2-BR0 define the MODBUS/RTU or ASCII baud rate for the communication:

BR2	BR1	BR0	MODBUS/RTU or ASCII baud rate
OFF	OFF	OFF	4800bd
OFF	OFF	ON	9600bd
OFF	ON	OFF	19200bd
OFF	ON	ON	38400bd
ON	OFF	OFF	57600bd
ON	OFF	ON	115200bd
ON	ON	OFF	230400bd
ON	ON	ON	256000bd

PARAMETER This DIP switch selects between the configuration via DIP switch or via FLASH parameter for the serial setup.

=0: The selected UnitID, baud rate from the DIP switch settings are used.

The parity is NONE and the one stop bit is used

=1: The selected UnitID from the DIP switches is used, but the serial parameters are taken from the FLASH parameters.

Baud rate can be selected between 300 to 256000 Baud.

Parity can be NONE, EVEN or ODD.

Stop bits can be ONE or TWO.

NOTE After changing the DIP switch, the device will be booted automatically So no voltage off/ voltage one cycle is necessary. After restarting, all LEDs flash briefly to represent the restart sequence.

10 LED indicators

Our serial ULTRA SLIM IO modules offer 3 LED indicators for status display, our Ethernet ULTRA SLIM IO modules offer 6 LED indicators for status display. Our BIGIO modules offer 4 LED indicators for status display.

10.1 LED indicators for serial ULTRA SLIM IOs

The following drawings show the LED indicators for all of our serial SLIMIO products:

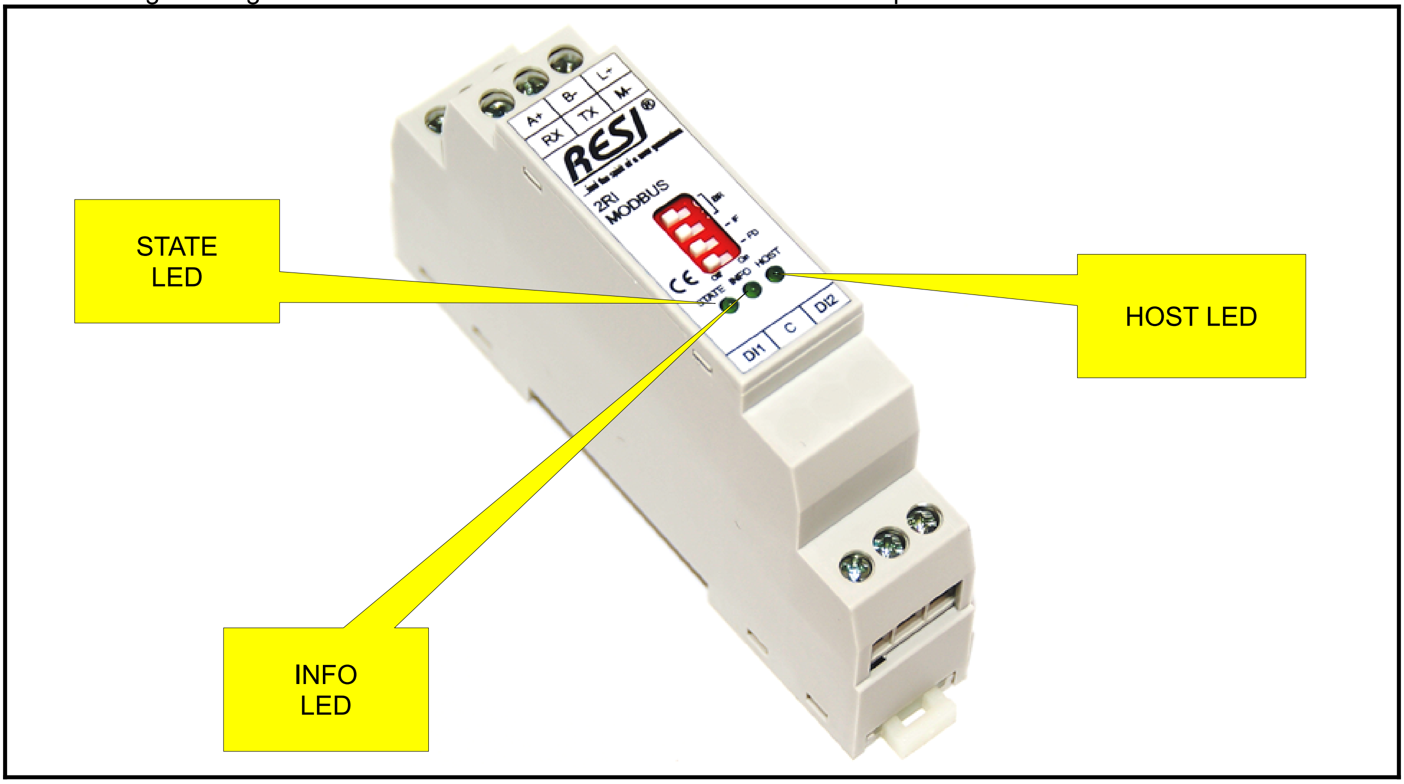


Figure: LED indicators for our serial ULTRA SLIM IO modules

STATE State LED, flashes slowly (approx. 1s) if the module is OK.
Flashes quickly when the module has an internal error

INFO This LED shows more information about the local IOs. The functionality depends on the used IO module. Please refer to the detailed description for each IO module.

HOST Shows whether serial data is currently being sent or received via the RS232 or RS485 interface

10.2 LED indicators for Ethernet ULTRA SLIM IOs

The following drawings show the LED indicators for all of our Ethernet SLIMIO products:

Figure: LED indicators for our Ethernet ULTRA SLIM IO modules

STATE	State LED, flashes slowly (approx. 1s) if the module is OK. Flashes quickly when the module has an internal error
INFO	This LED shows more information about the local IOs. The functionality depends on the used IO module. Please refer to the detailed description for each IO module.
HOST	Shows whether serial data is currently being sent or received via the internal serial interface to the Ethernet controller
CFG	Factory setting LED: In working mode, this LED flashes in the same rhythm as the STATE LED. If the DIP switch CFG is ON when restarting, the STATE LED is always on and the CFG LED flashes slowly. When this process is complete, both LEDs will flash very fast. Then the CFG LED must be set to OFF again!
LINK	This LED is on when the Ethernet interface is electrically connected correctly with the network
DATA	This LED shows the data flow on the Ethernet interface

10.3 LED indicators for serial BIG IOs

The following drawings show the LED indicators for all of our serial BIGIO products:

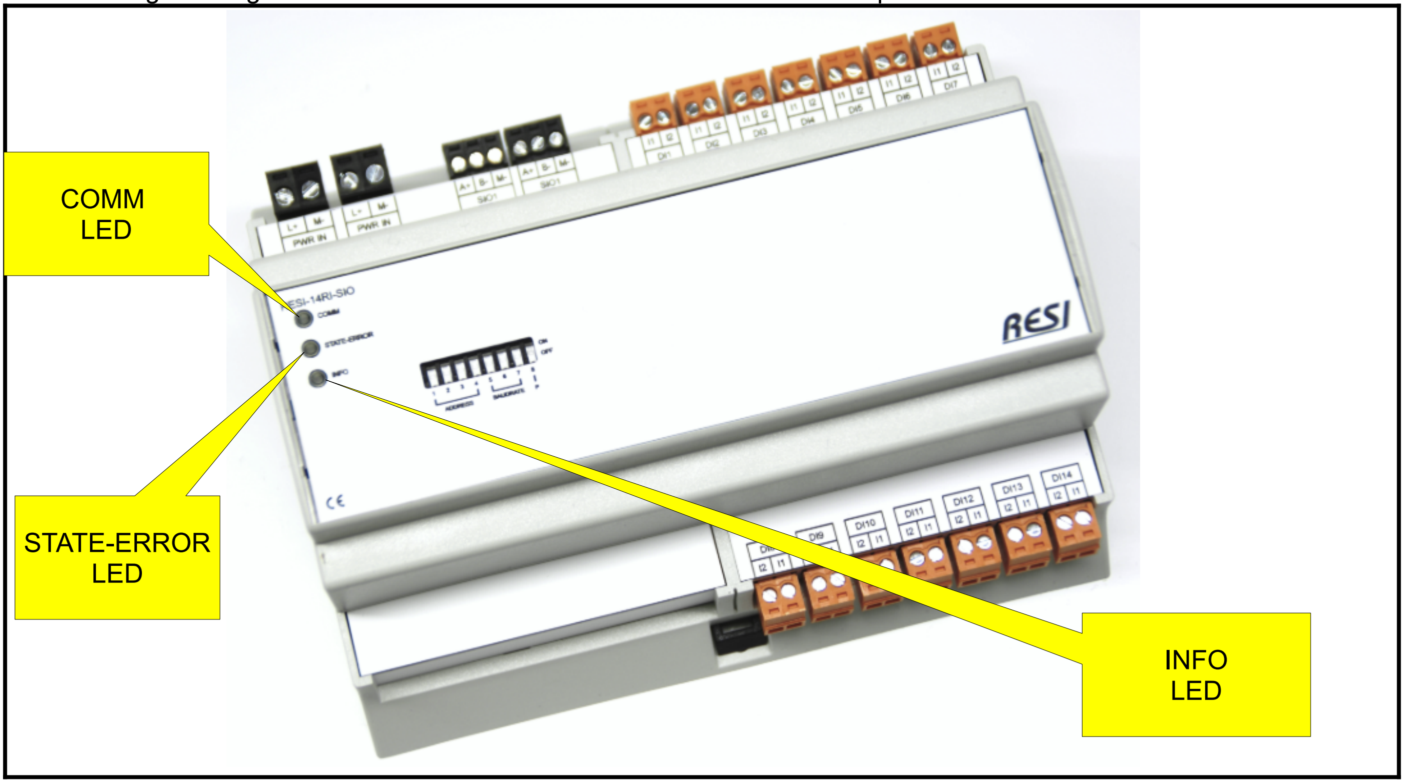


Figure: LED indicators for our serial BIG IO modules

COMM Shows whether serial data is currently being sent or received via the RS485 interface

STATE-ERROR State LED, flashes slowly (approx. 1s) in WHITE if the module is OK.
Flashes quickly in RED when the module has an internal error

INFO This LED shows more information about the local IOs. The functionality depends on the used IO module. Please refer to the detailed description for each IO module.

10.4 LED indicators for Ethernet BIG IOs

The following drawings show the LED indicators for all of our Ethernet BIGIO products:

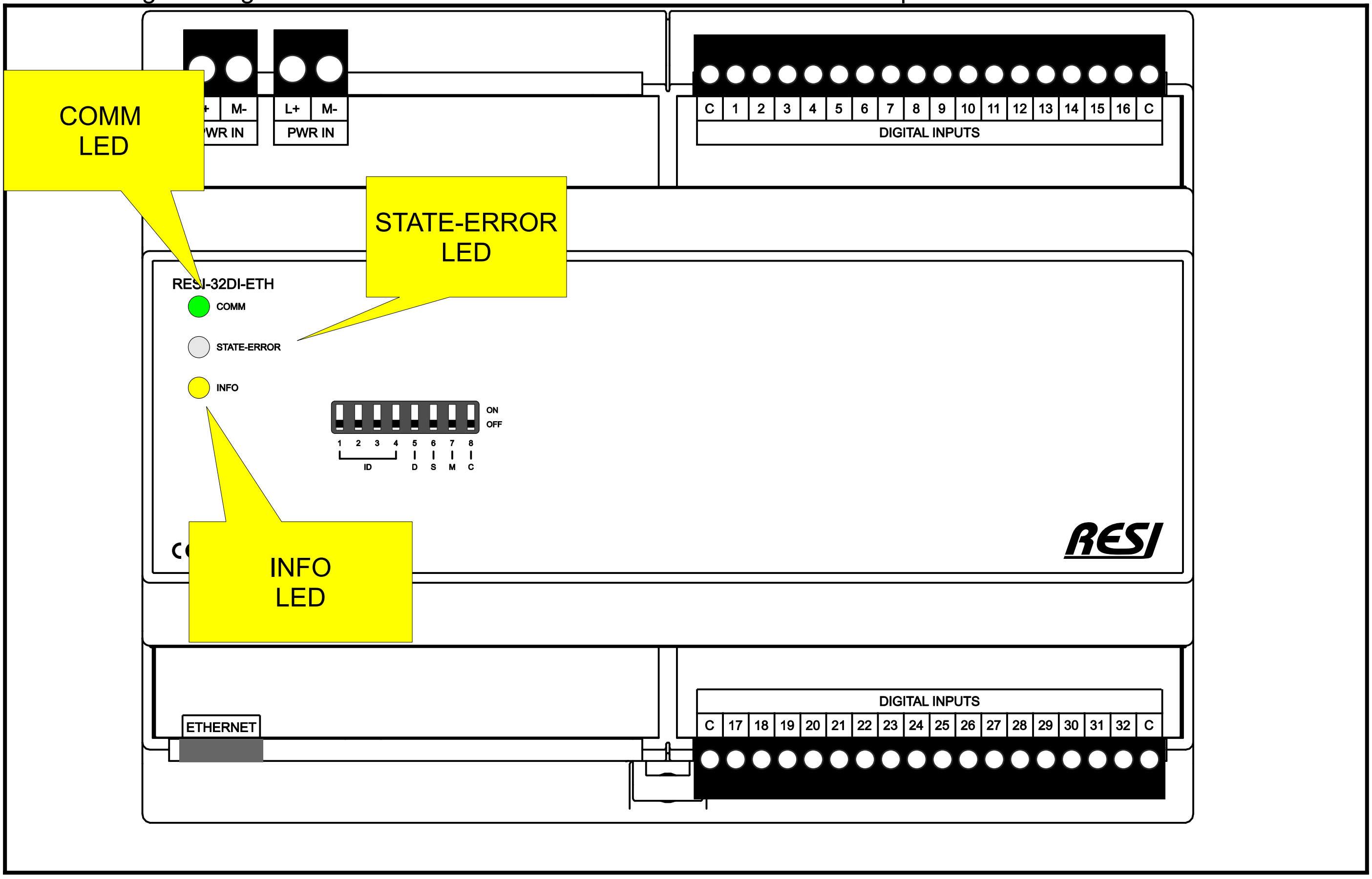


Figure: LED indicators for our serial BIG IO modules

COMM Shows whether serial data is currently being sent or received via the Ethernet interface

STATE-ERROR State LED, flashes slowly (approx. 1s) in WHITE if the module is OK. Flashes quickly in RED when the module has an internal error

INFO This LED shows more information about the local IOs. The functionality depends on the used IO module. Please refer to the detailed description for each IO module.

10.5 LED indicators for BIG IOs RESI-S16DI8PO-SIO,RESI-S8PO-SIO

The following drawings show the LED indicators for our serial BIGIO products RESI-S16DI8PO-SIO and RESI-S8PO-SIO:

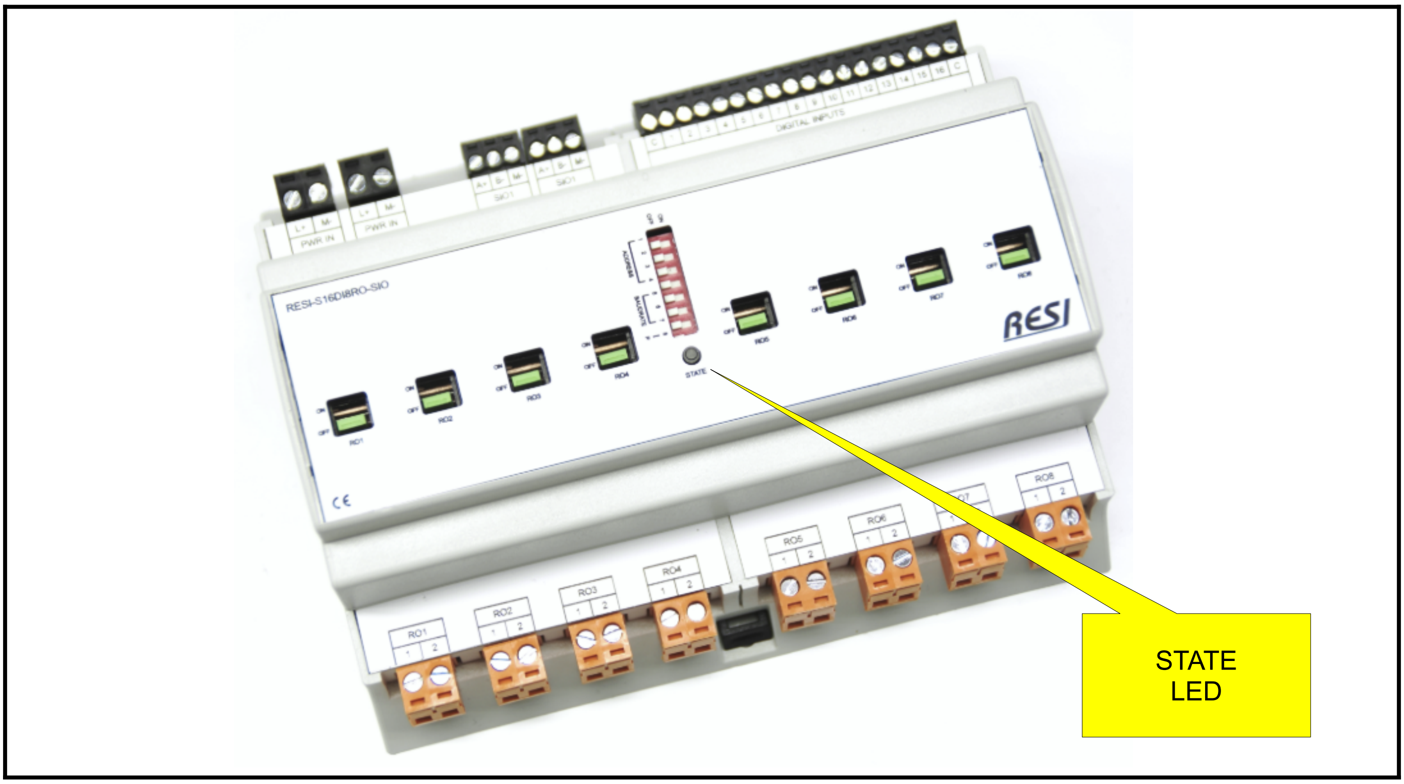


Figure: LED indicators for our BIG IO modules RESI-S16DI8PO-SIO, RESI-S8PO-SIO

STATE State LED, flashes slowly (approx. 1s) in WHITE if the module is OK.
Flashes quickly in RED when serial data is currently being sent or received via the RS485 interface

11 DIMENSIONS

11.1 ULTRA SLIM IOs: RESI-xxx-SIO

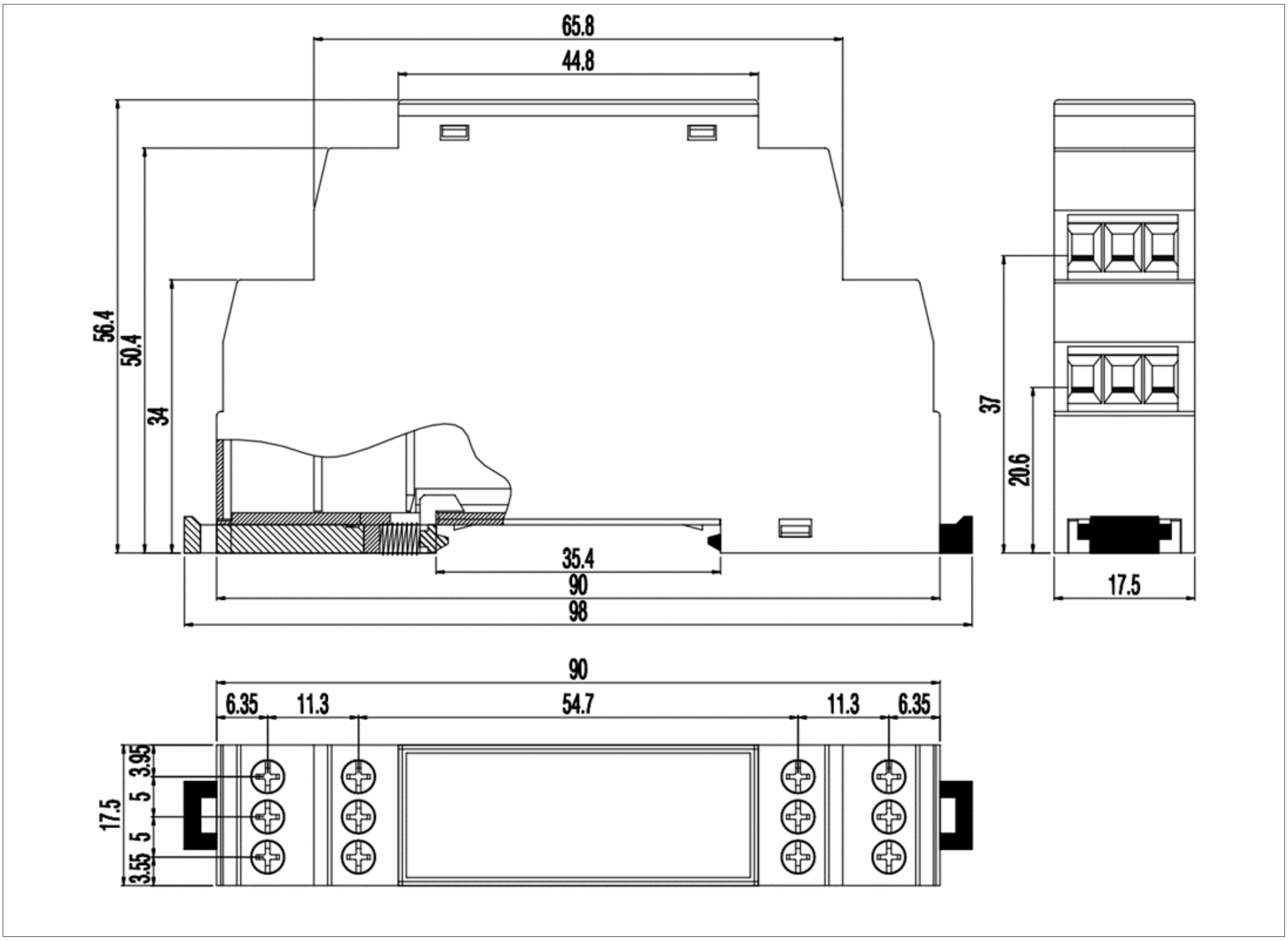


Figure: Dimensions of the housing for our serial ULTRA SLIM IO modules in mm

Dimensions

Housing illustration	17.5 x 90 x 58
Color	gray RAL 7035
Material	PA-UL 94 V0
Protection class	IP20 based on DIN 40050 / EB 60529

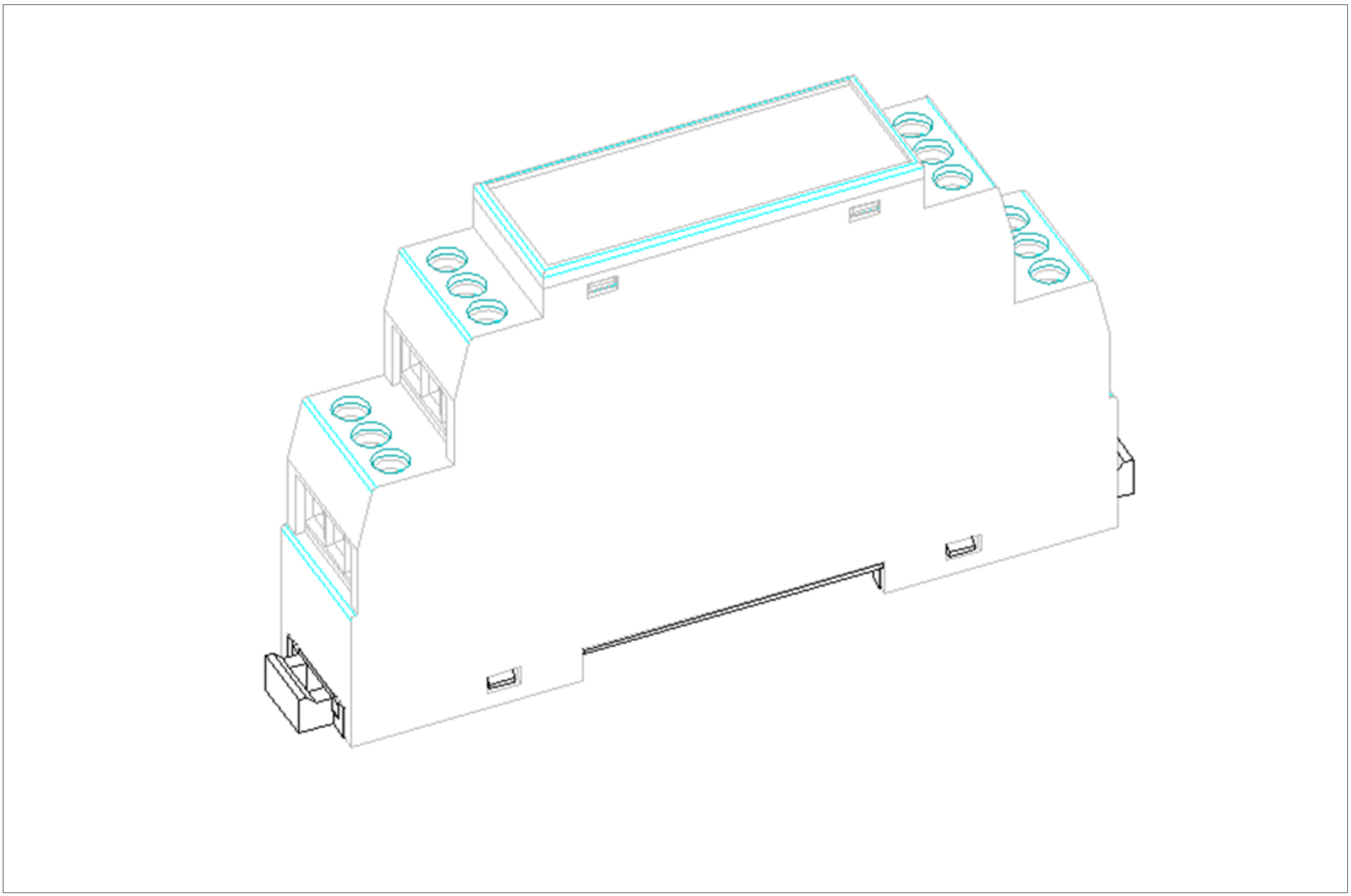


Figure: For our serial ULTRA SLIM IO modules: Housing illustration in 3D

11.2 ULTRA SLIM IOs: RESI-xxx-ETH

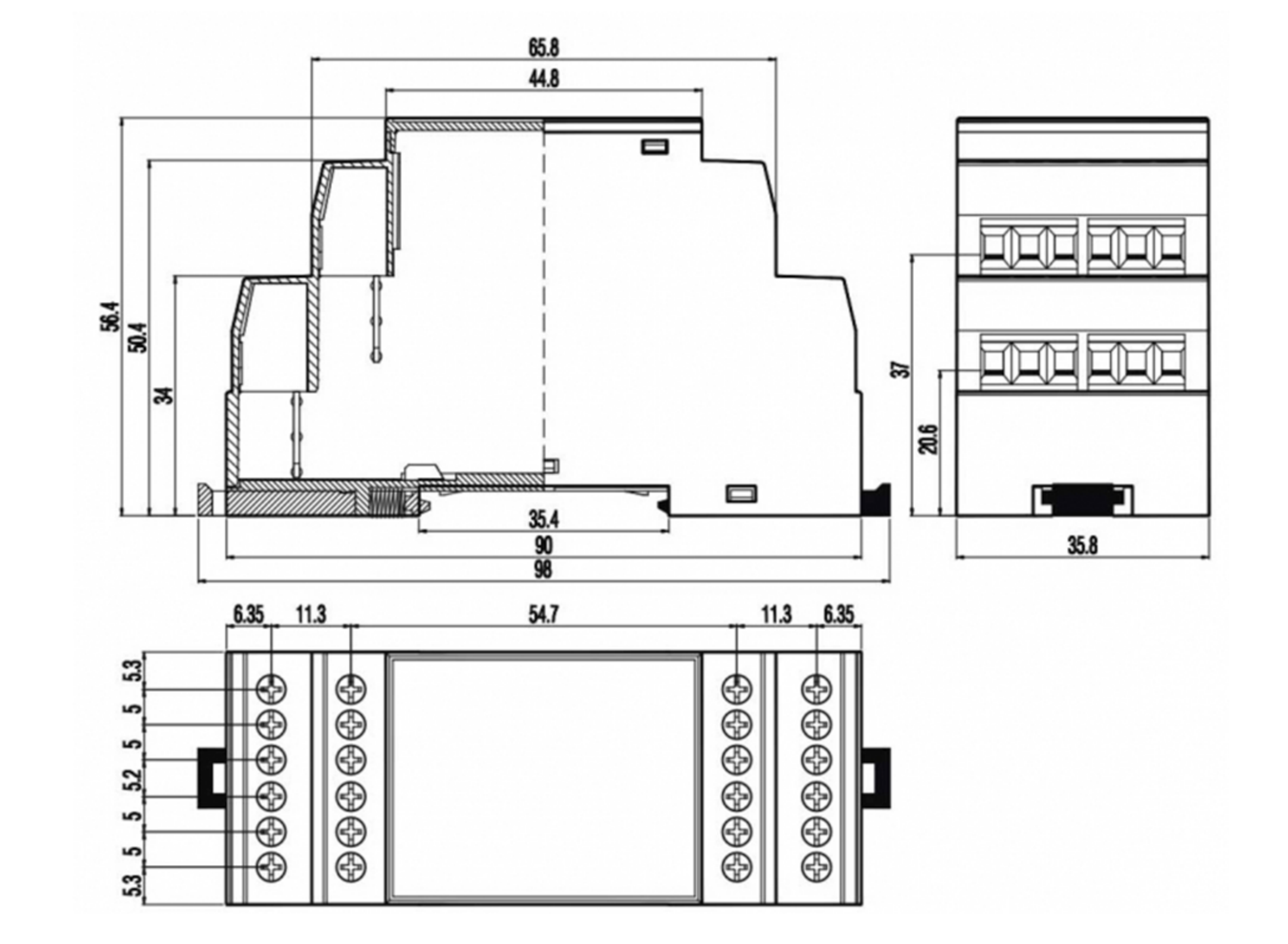


Figure: Dimensions of the housing for our Ethernet ULTRA SLIM IO modules in mm

Dimensions

Housing illustration	35.8 x 90 x 58
Color	gray RAL 7035
Material	PA-UL 94 V0
Protection class	IP20 based on DIN 40050 / EB 60529

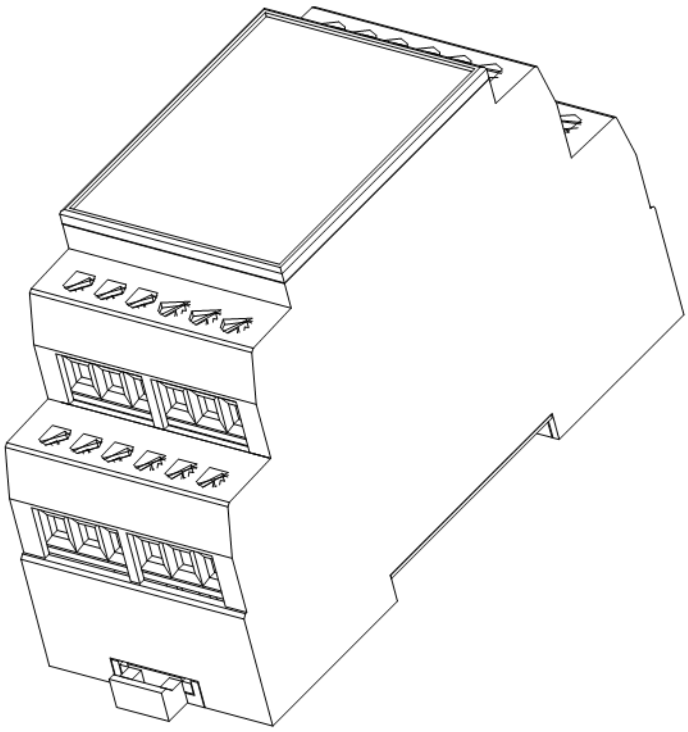


Figure: For our Ethernet ULTRA SLIM IO modules: Housing illustration in 3D

11.3 BIG IOs: RESI-xxx-SIO XT8

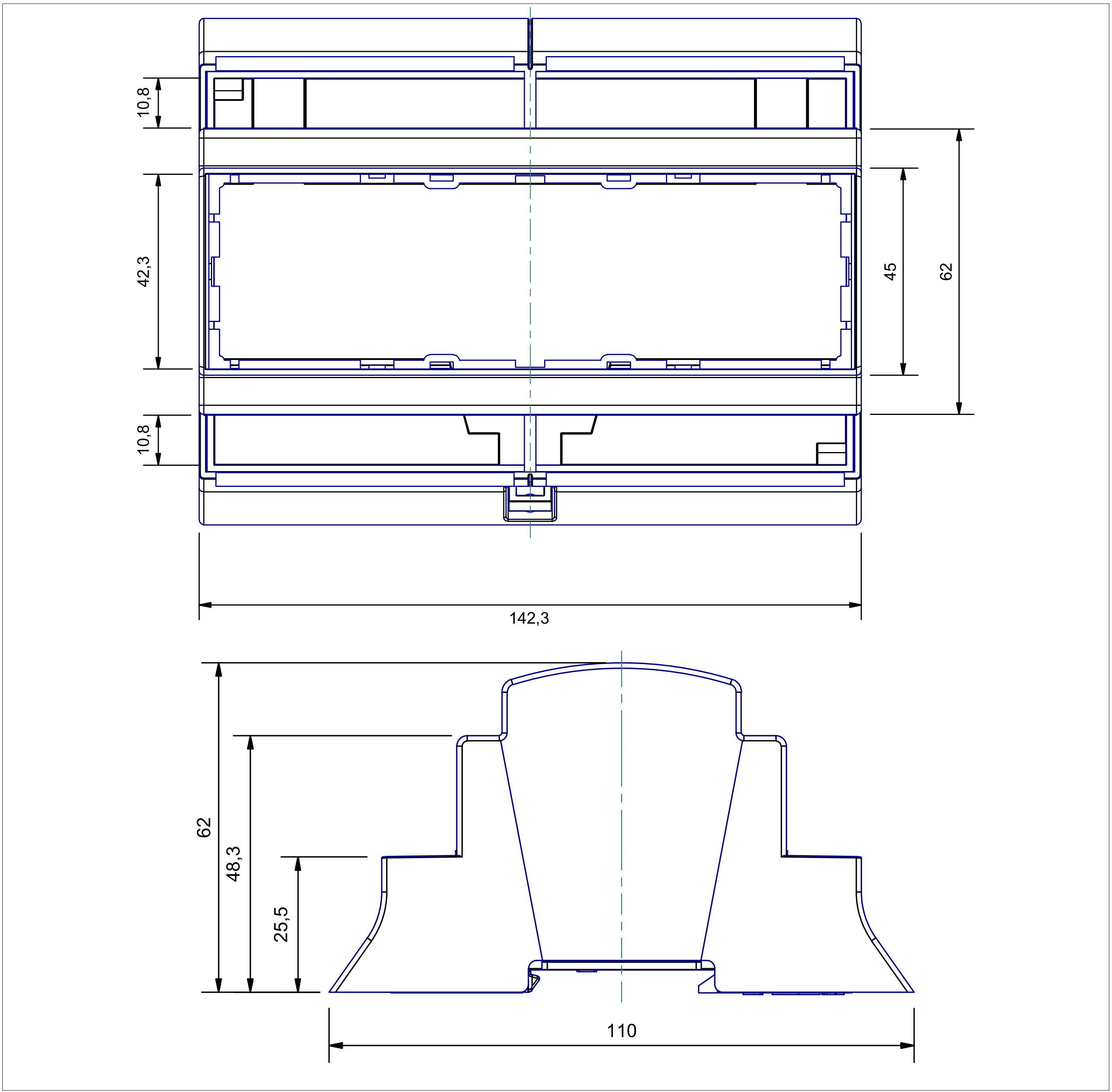


Figure: Dimensions of the housing for our serial BIG IOs XT8 modules in mm

Dimensions	
Housing illustration	142.3 x 110 x 62
Color	grey RAL 7035
Material	Self-extinguishing Blend PC/ABS UL94-VO
Protection class	IP20 based on DIN 40050 / EB 60529

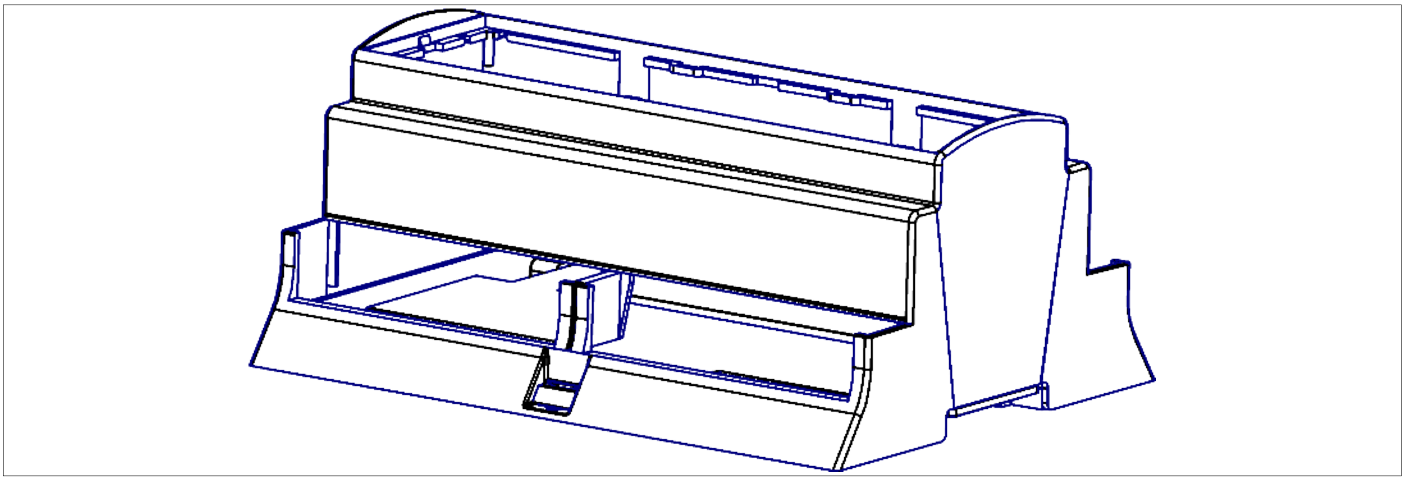


Figure: For our serial BIG IOs XT8 modules: Housing illustration in 3D

11.4 BIG IOs: RESI-xxx-SIO XT12

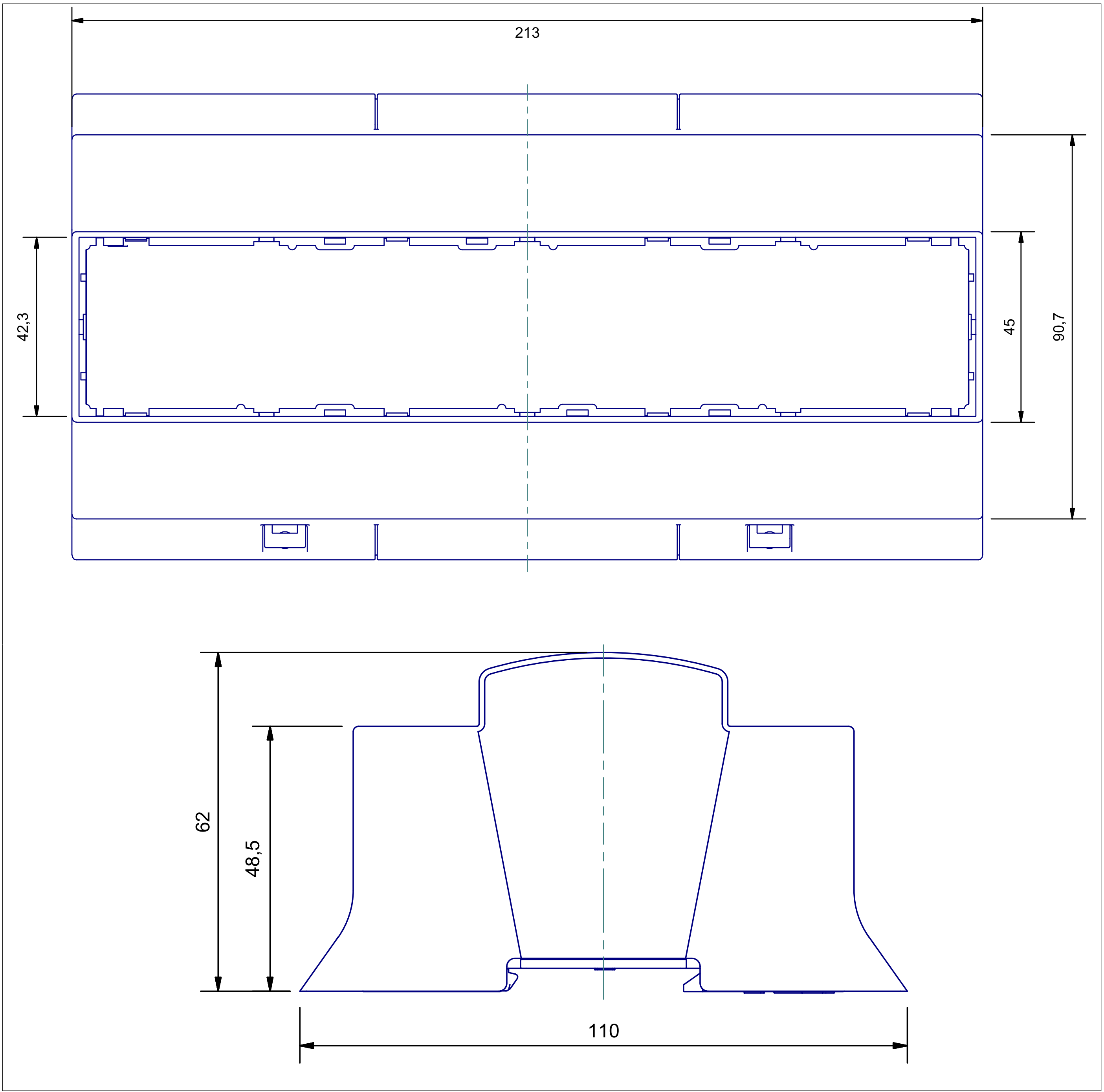


Figure: Dimensions of the housing for our serial BIG IOs XT12 modules in mm

Dimensions

Housing illustration	213 x 110 x 62
Color	grey RAL 7035
Material	Self-extinguishing Blend PC/ABS UL94-VO
Protection class	IP20 based on DIN 40050 / EB 60529

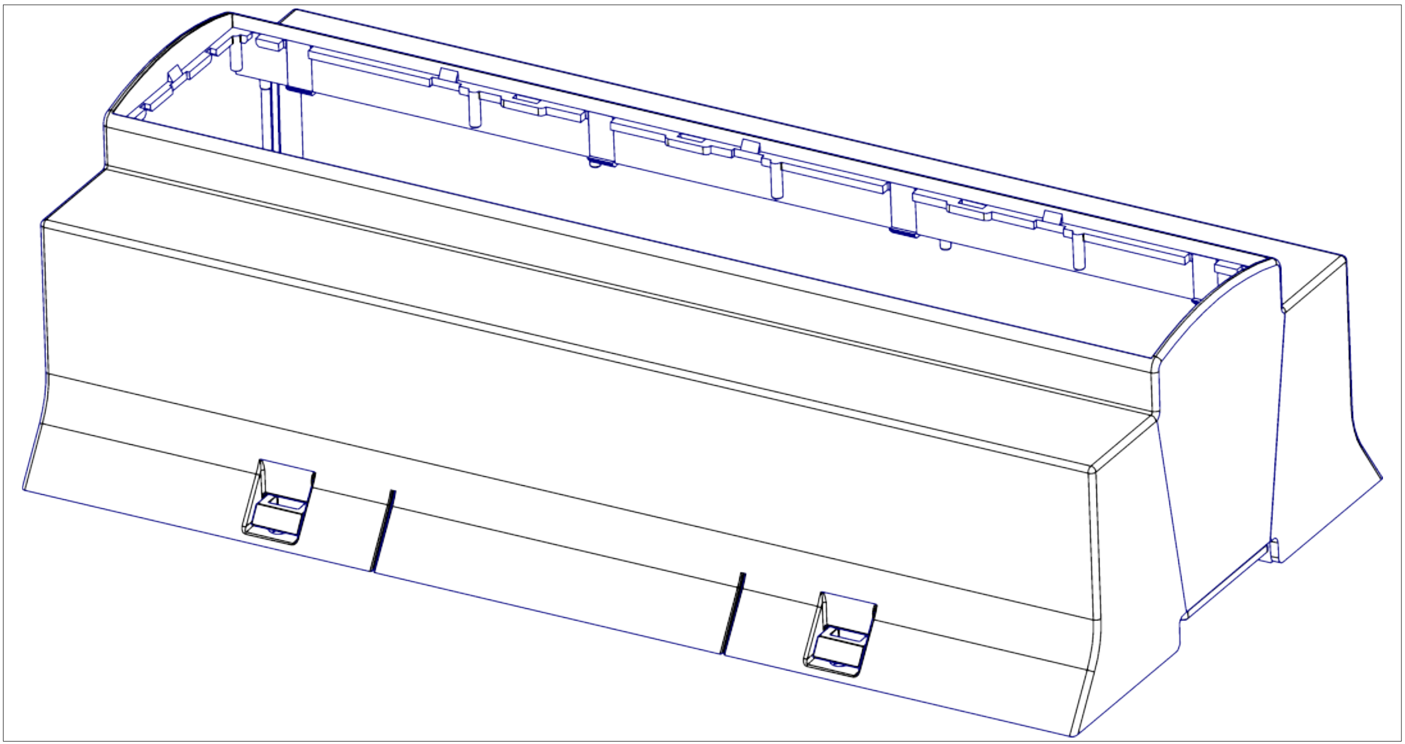
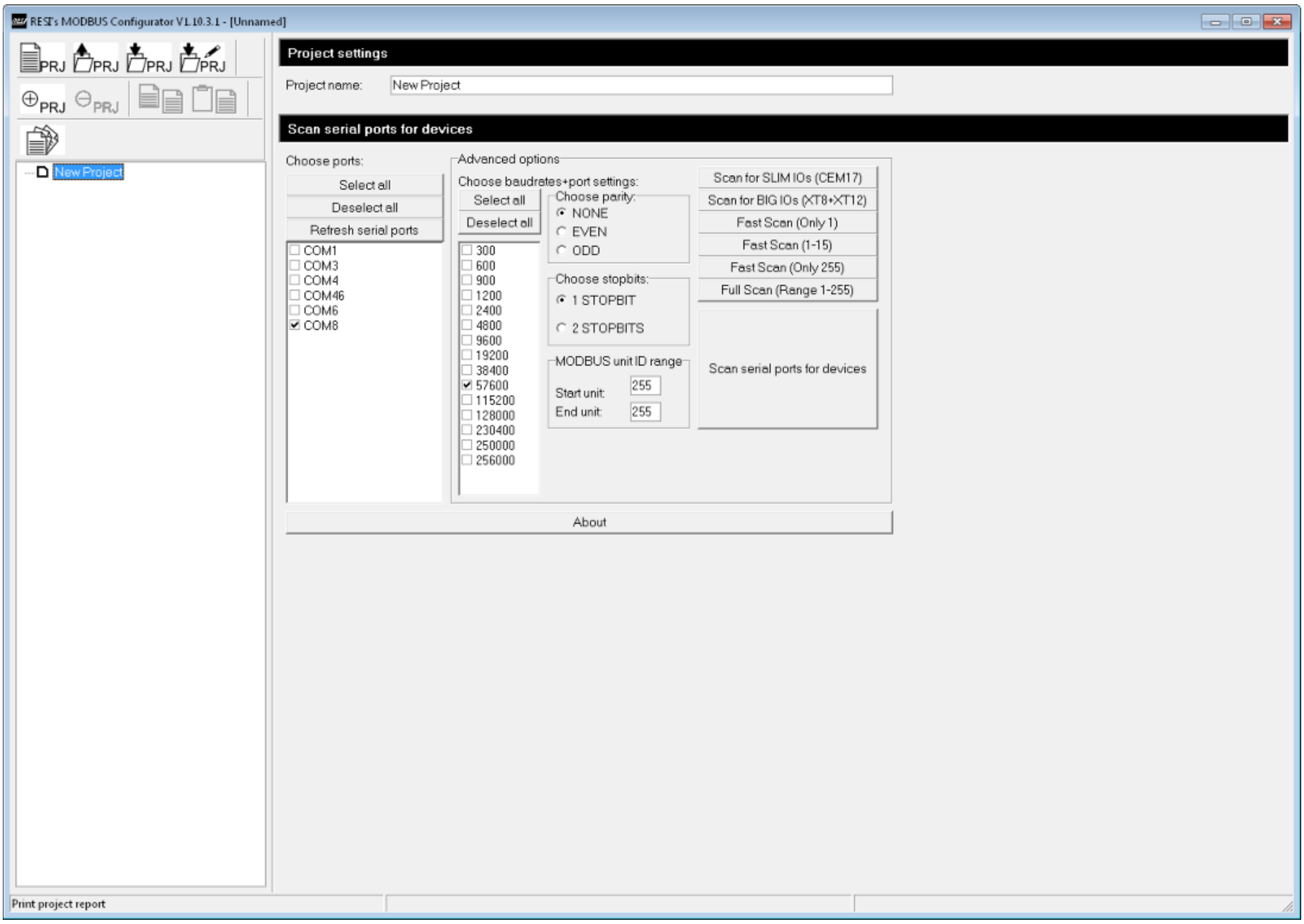


Figure: For our serial BIG IOs XT12 modules: Housing illustration in 3D

12 MODBUSConfigurator software

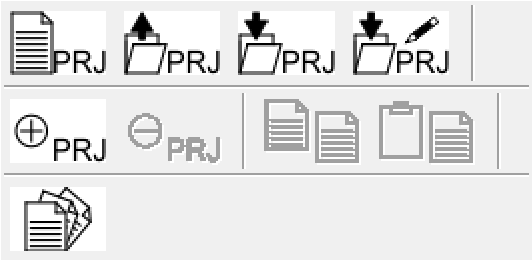
12.1 General information

We offer a free configuration & test tool named MODBUSConfigurator. This tool offers the possibility to configure and test almost all of our products. When you start the software you will see the following picture.



12.2 Main menu icons

The main menu icons provide the purpose to start a new project or open an existing project or save the current project to a new project file. You can also add some new items to a gateway like meters or DALI lamps or you can add a new gateway to an existing project. of course you can delete and configure gateway from a project. You can copy and paste objects within the existing project and you can generate a project report for documentation.



First row:

- Create a new empty project
- Open an existing project
- Save current project
- Save current project with new name

Second row:

- Add a gateway, IO module or object to the project tree
- Delete selected item from the project tree
- Copy selected item into internal clipboard
- Paste internal clipboard to project tree

Third row:

- Generate project report

12.3 Project settings

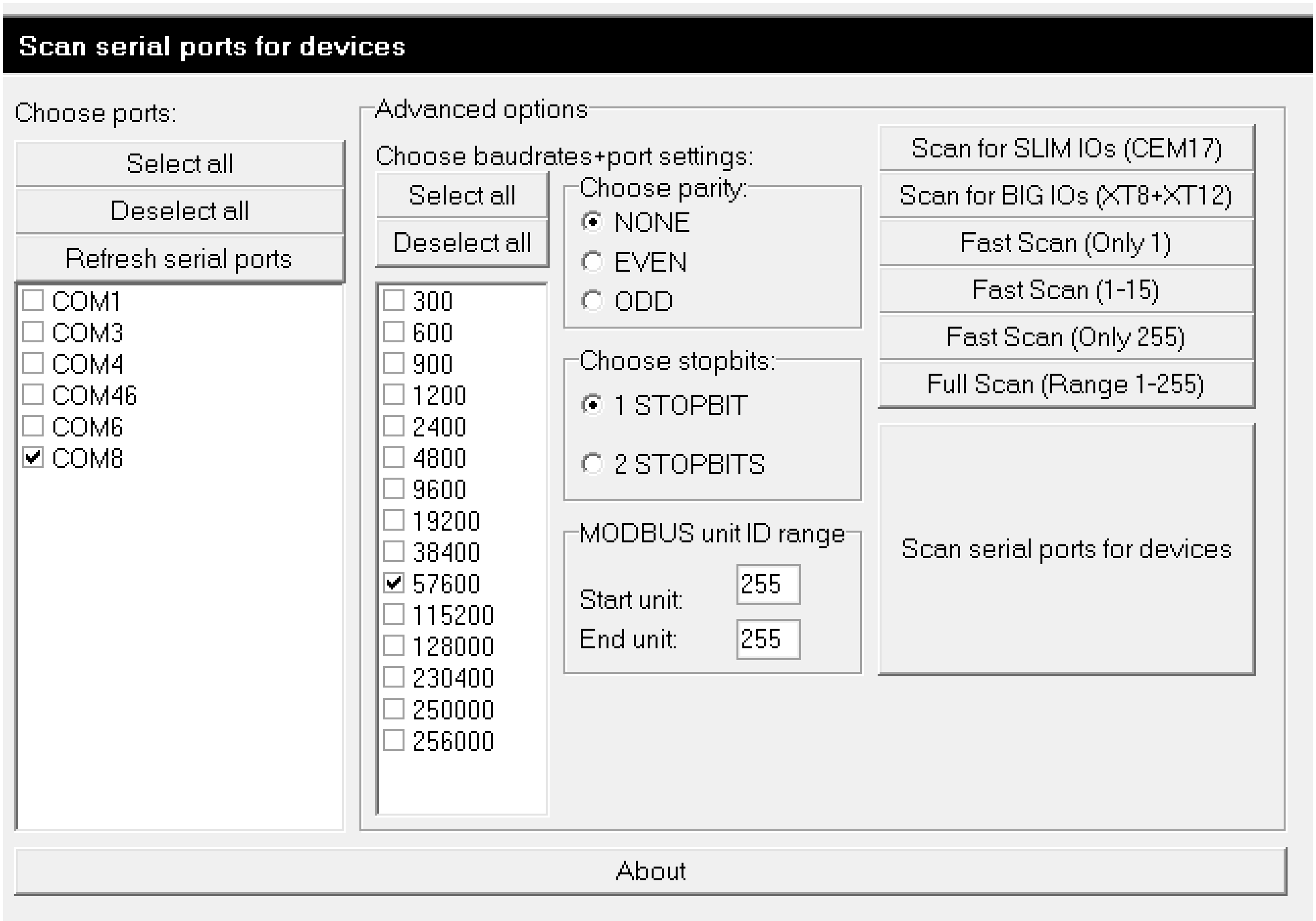
In this section you can define your special project name:

12.4 Scan for serial devices

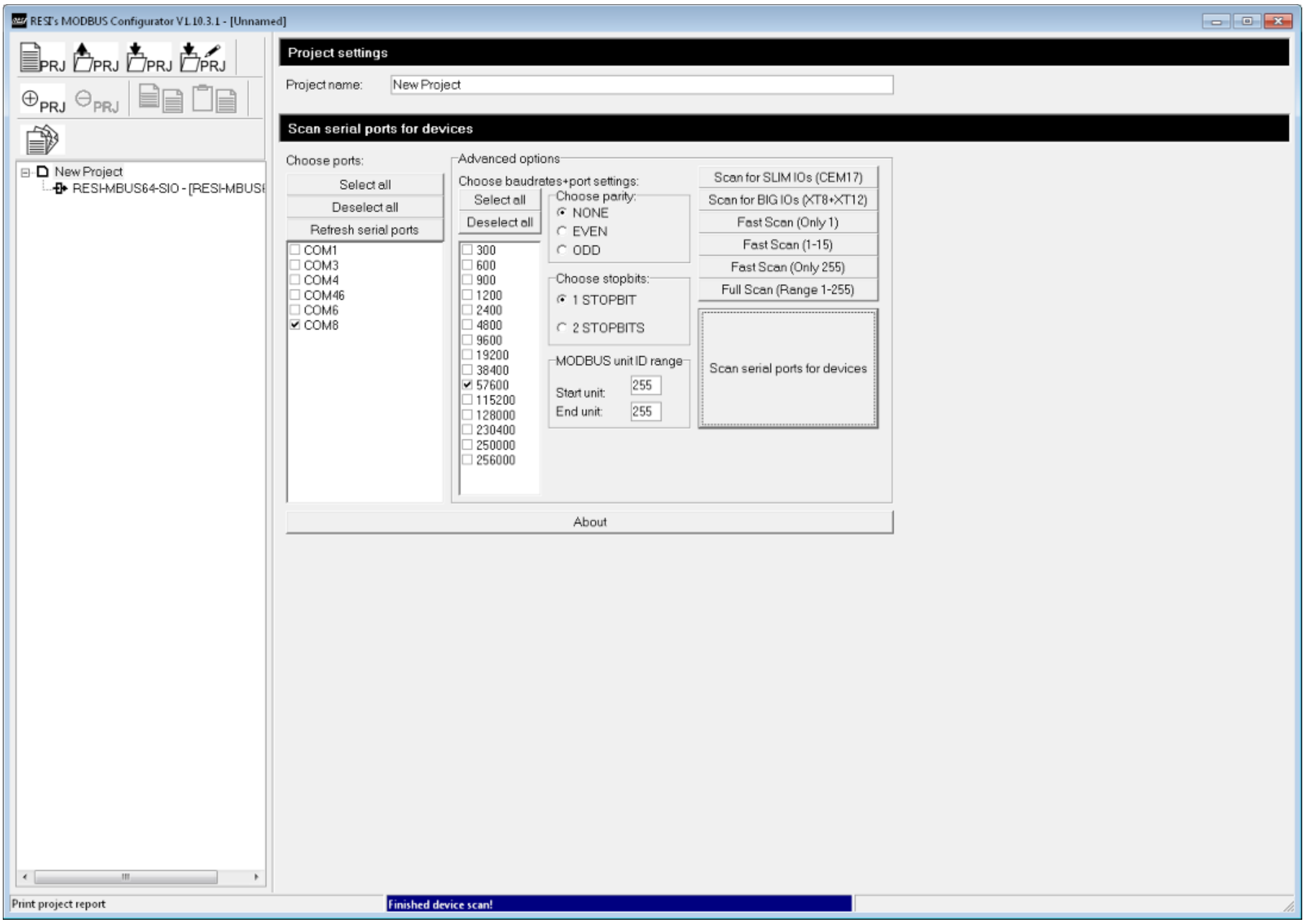
In this section you can configure an automatic search process to find all connected devices. Therefore you see on the left side the current available serial interfaces currently connected to your computer.

With the button select all you can select all of the available serial interfaces. With the button Select all you can select all serial interfaces for the automatic search process. The button Deselect all will deselect all serial interfaces for this automatic search. The button Refresh serial ports will scan again all connected serial interfaces of your computer and refresh the displayed list of serial interfaces.

In the area on the right side you will find settings for baud rates, parity and stop bits. Also you can select the range of MODBUS unit IDs which are used for this automatic search process. The automatic search for connected serial devices starts by pressing the Scan serial ports for devices button. The remaining buttons offer a quick selection for certain scenarios. The About button opens a dialog with information about the program version.

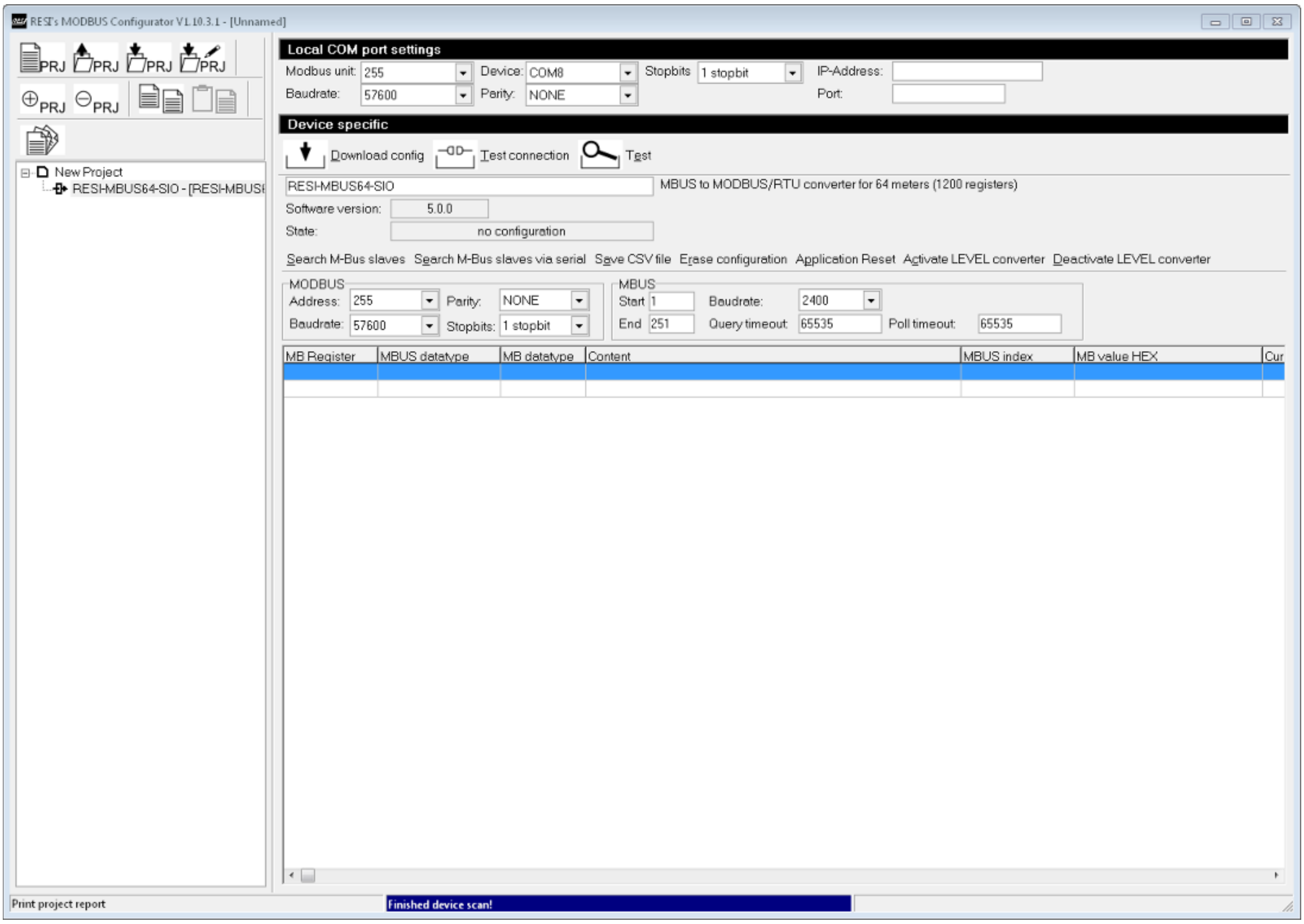


If your automatic scan was successful, the project tree will show the connected IO modules or gateways:



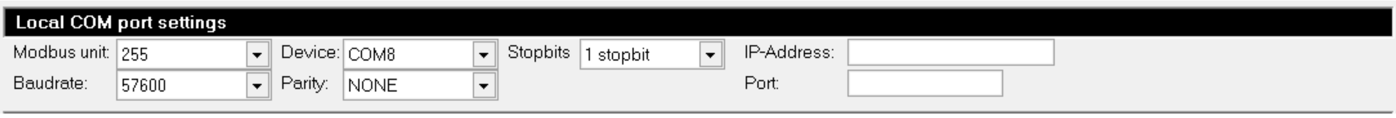
12.5 Configure and test a device

When you click in the project tree onto a device you can select a device. On the right side you will see all technical parameters to set up and test a specific device.



12.5.1 Local COM port settings

In this section you will see the current configured serial or IP settings to communicate with the selected device. Also you can change the MODBUS unit ID which is used for the communication protocol.



Select serial communication:

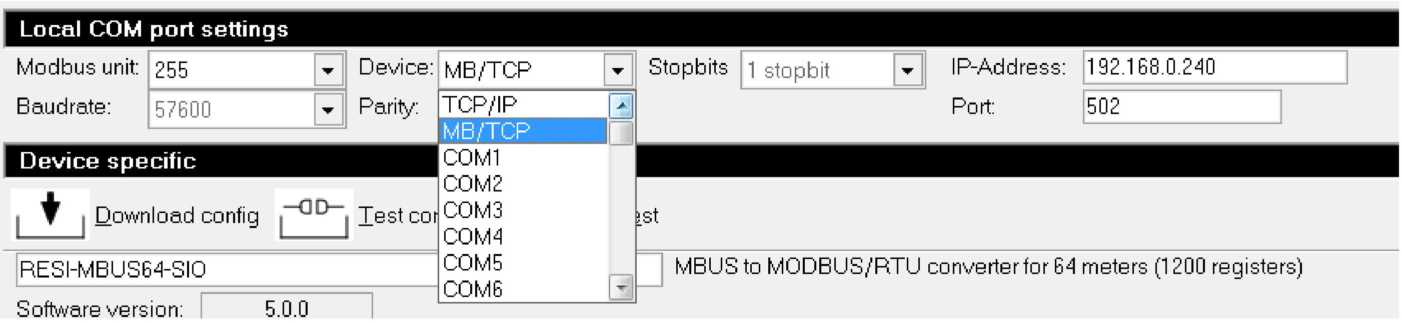
As long as you select a serial device COMx from the drop-down list Device you will use a serial interface to communicate with the connected module. Choose your desired parameters for baud rate, parity and stopbits.

Select Ethernet communication:

If you open the drop down list, you will notice two other options

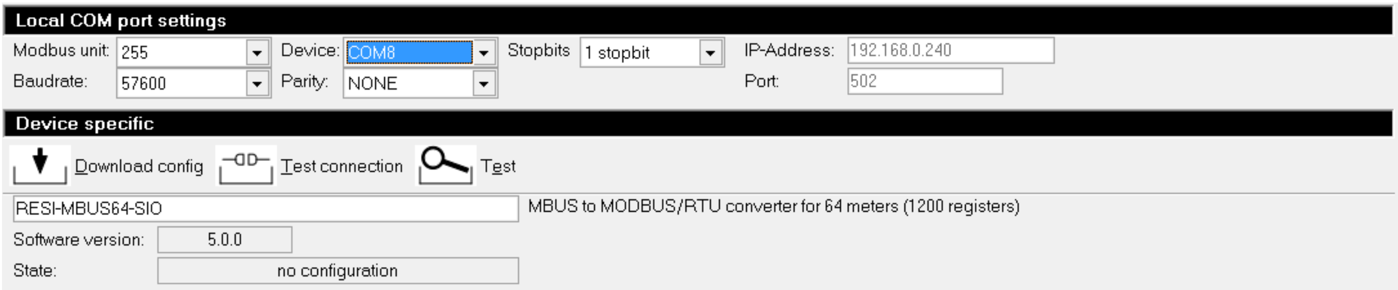
- TCP/IP: Use serial communication via TCP/IP
- MB/TCP: use MODBUS/TCP protocol via TCP/IP

First, select one of the two options, then you can enter a IP address and a socket number for the communication via Ethernet.



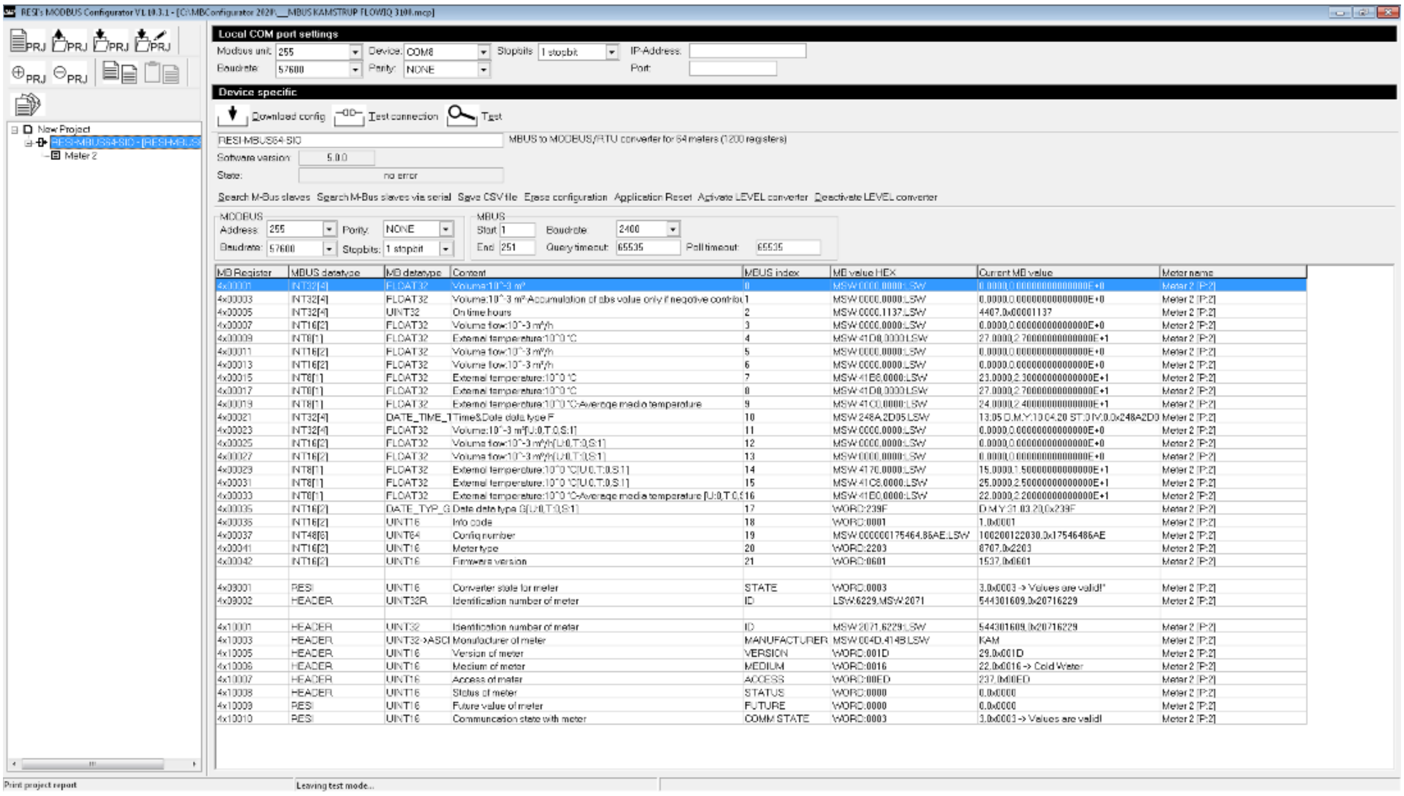
Check connection settings:

After you have defined your communication settings, you can test the communication by pressing the button test connection. If the connection is not successful established, an error dialog will pop up. If the communication is ok, the fields Software version and State will show more information about the device.



12.5.2 Device specific area

In this section you will find specific information about the connected device. In this sample we have connected a MBUS devices with one meter.



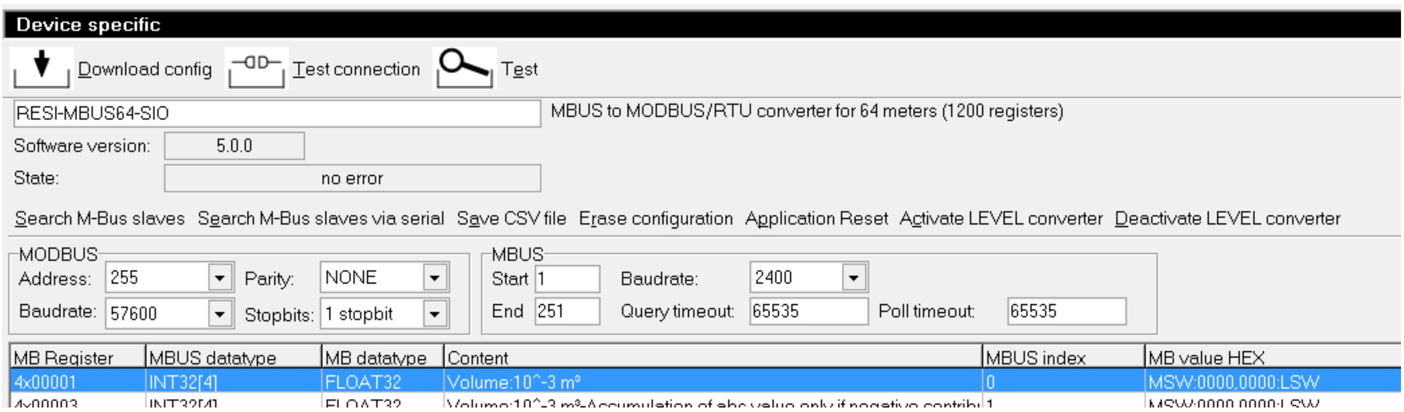
For all devices you have two options:

Download config

With this button you can download your new settings which you have selected in the device specific area into the connected module. After that you may have changed the basic configuration settings. So don't forget to change the Local COM port settings to establish communication to the module again.

Test

This button activates a cyclic test option, which will show values from the connected device. IN this case it will show the current meter values of the connected MBUS meter on your MBUS gateway.



Device specific commands:

You will also find a command area with buttons for device specific commands. In the case of our MBUS gateways you will find the functions:

- Search M-Bus slaves
- Search M-Bus svales via serial
- Save CSV file
- etc.

Please refer to the detailed documentation for each module, what the specific commands are for and how you can use them.

General device settings:

Below of the device specific command area is an area with general settings for the selected device. IN our sample case it will be:

- MODBUS address
- Baudrate
- Parity
- Stopbits
- Primary MBUS start ID
- Primary MBUS end ID
- etc.

This settings can be downloaded into the device with the button Download. Some of your modules can also upload this settings fro the device. Then they have an additional button in the device specific command area.

Value grid:

Under the specific device settings most of our module show a gird with more configuration possibilities or a grid with MODBUS registers. Again the configuration grid will be downloaded with the button Download int the device. The MODBUS values will be cyclic updated by activating the Test button.

For more details refer to the specific devices, what information the MODBUSConfigurator software will offer.

13 RESI-14RI-SIO

13.1 General information

This series of IO modules offer the following features:

- 14 digital inputs for 12-250Vac/dc signals
- Each digital input is galvanic insulated to all other digital inputs
- Each digital input is cabled via extra 2 pin removable terminal
- Galvanic insulated RS485 interface for communication with a host system

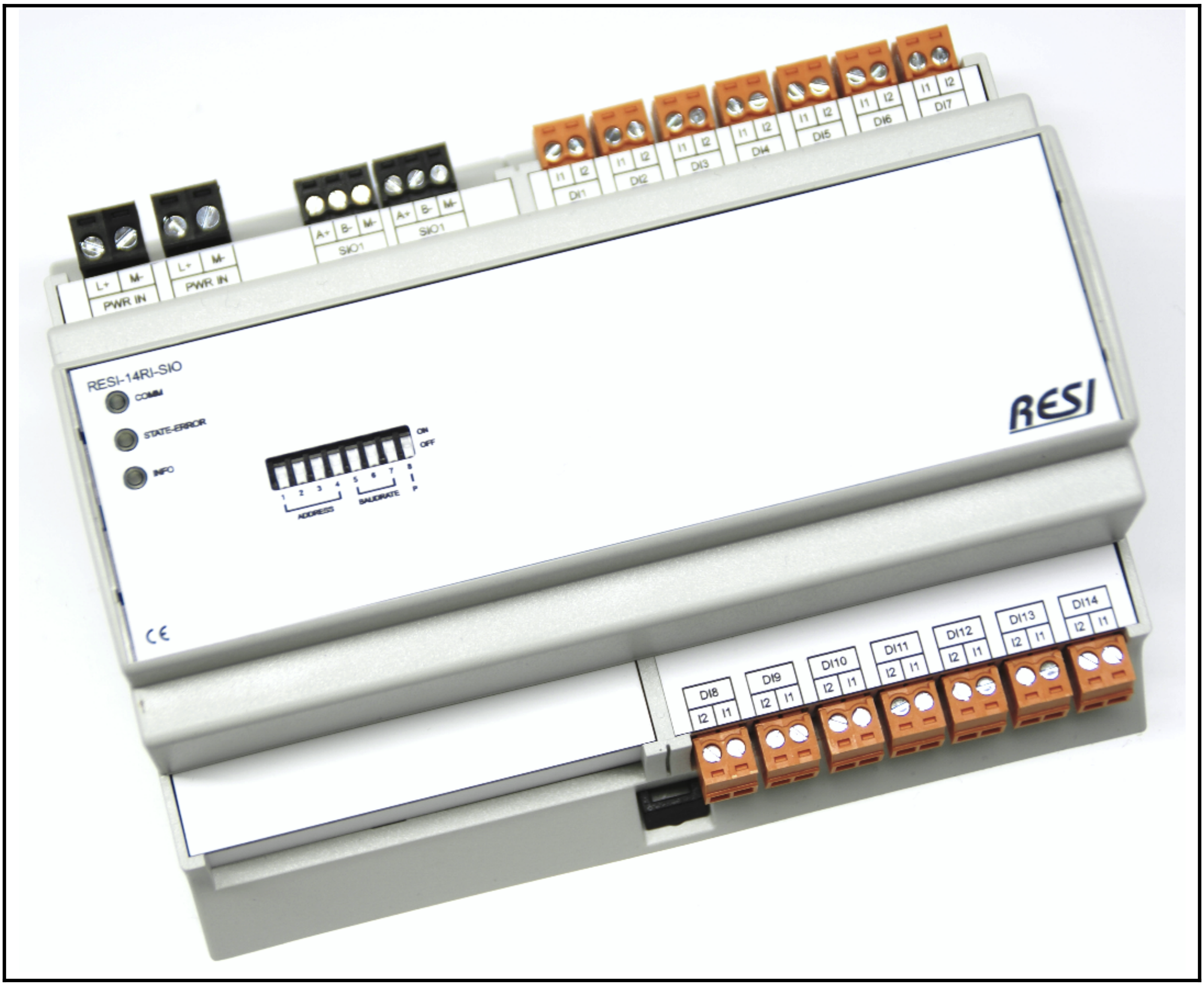


Figure: Our RESI-14RI-SIO module

13.2 Technical specification

Beside the basic technical data, which fulfil all of our IO modules, this IO modules meet the following technical specifications:

Power consumption	<0.5W
Product housing	BIG IO XT8
Product weight	255g
Digital inputs	
Total amount of inputs	14
Sampling rate	Every 5ms
DC rating	
Input voltage range	12-250V= +/-10%
Input current	per channel
	approx. 0.85mA@12V=
	approx. 0.85mA@24V=
	approx. 0.85mA@32V=
	approx. 0.85mA@48V=
	approx. 0.85mA@250V=
Input power consumption	max. 0.3W/channel
Logic levels	0: <4.5V~
	1: >7.5V~
AC rating	
Input voltage range	12-250V= +/-10%
Input current	per channel
	approx. 0.85mA@12V~
	approx. 0.85mA@24V~
	approx. 0.85mA@48V~
	approx. 0.85mA@110V~
	approx. 0.85mA@230V~
	approx. 0.85mA@250V~
Input power consumption	max. 0.3W/channel
Logic levels	0: <4.5V~
	1: >7.5V~
Cable connection	Via 14 2-pin plug-in terminal block
Terminal type	RM3.5
Galvanic insulation	Yes, to each other digital input and to IO module
Default serial settings	
Baud rate	via DIP switch
Parity	none
Stop bit(s)	one
UnitID	255

13.3 Additional terminals & LED states

DIGITAL INPUTS	14 digital inputs for 12-250Vac/dc signals	
	Eight 2 pin plug-in terminal block	
	Terminal type:	RM3.5
	I1:	Digital input +: AC/DC signal
	I2:	Digital input -: Ground or neutral wire
		0=open or short cut
		1=AC or DC voltage between 12 and 250V
Pin layout	Pin 1:	I1
	Pin 2:	I2
INFO	This LED is on, if at least one of the digital inputs is high (1).	
	This LED is off, if all digital inputs are low (0).	

13.4 Connection diagram

13.4.1 Cabling of the digital inputs with DC signals

In the below drawing you see the cabling of the 14 digital inputs of the module with DC signals.

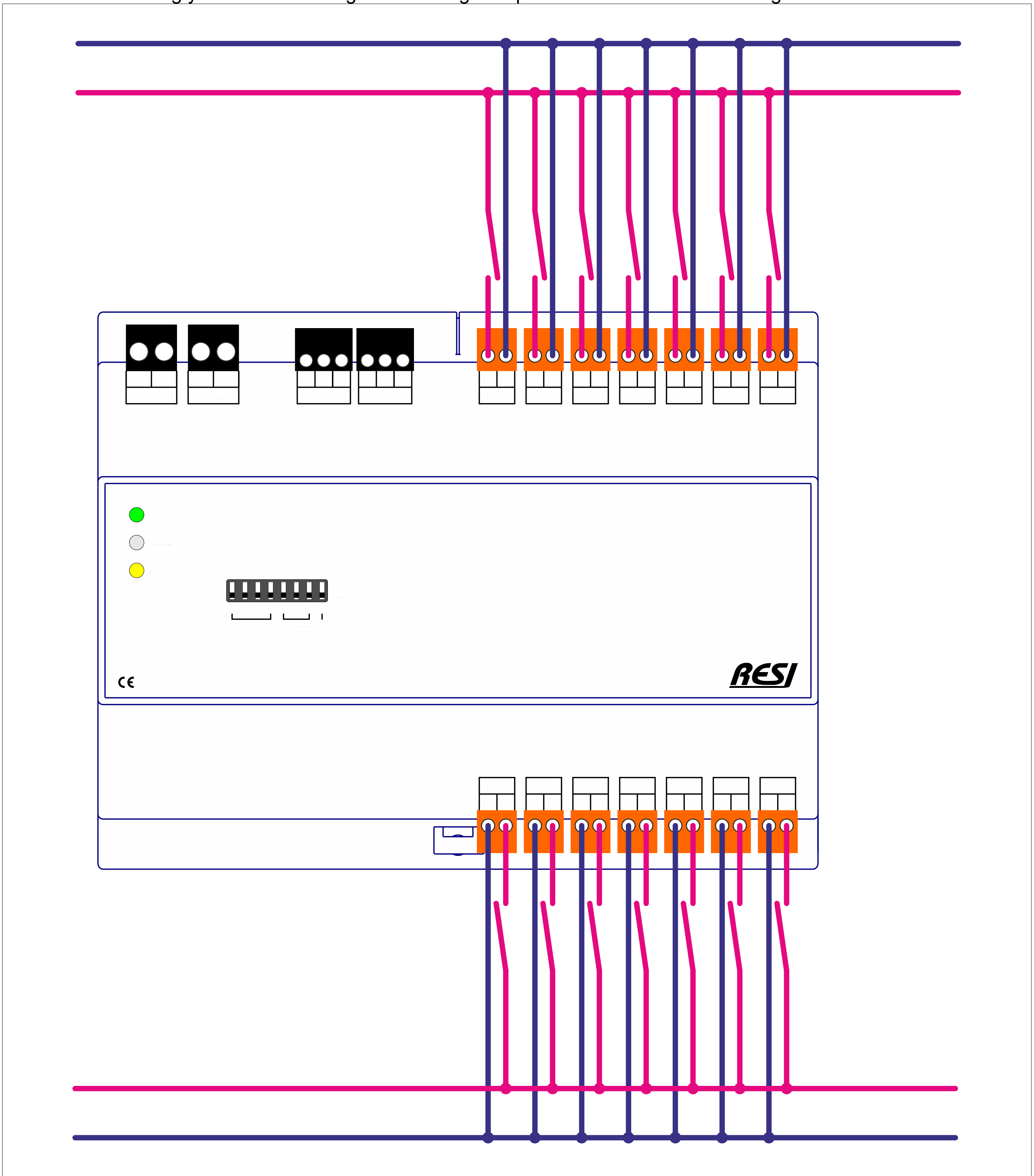


Figure: Cabling of the digital inputs of the IO module with DC signals

Don’t forget, that you can use signals from different DC power supplies for each input, because all digital inputs are galvanically insulated to each other. Also you can mix AC and DC input signals on one module!

13.4.2 Cabling of the digital inputs with AC signals

In the below drawing you see the cabling of the 14 digital inputs of the module with AC signals.

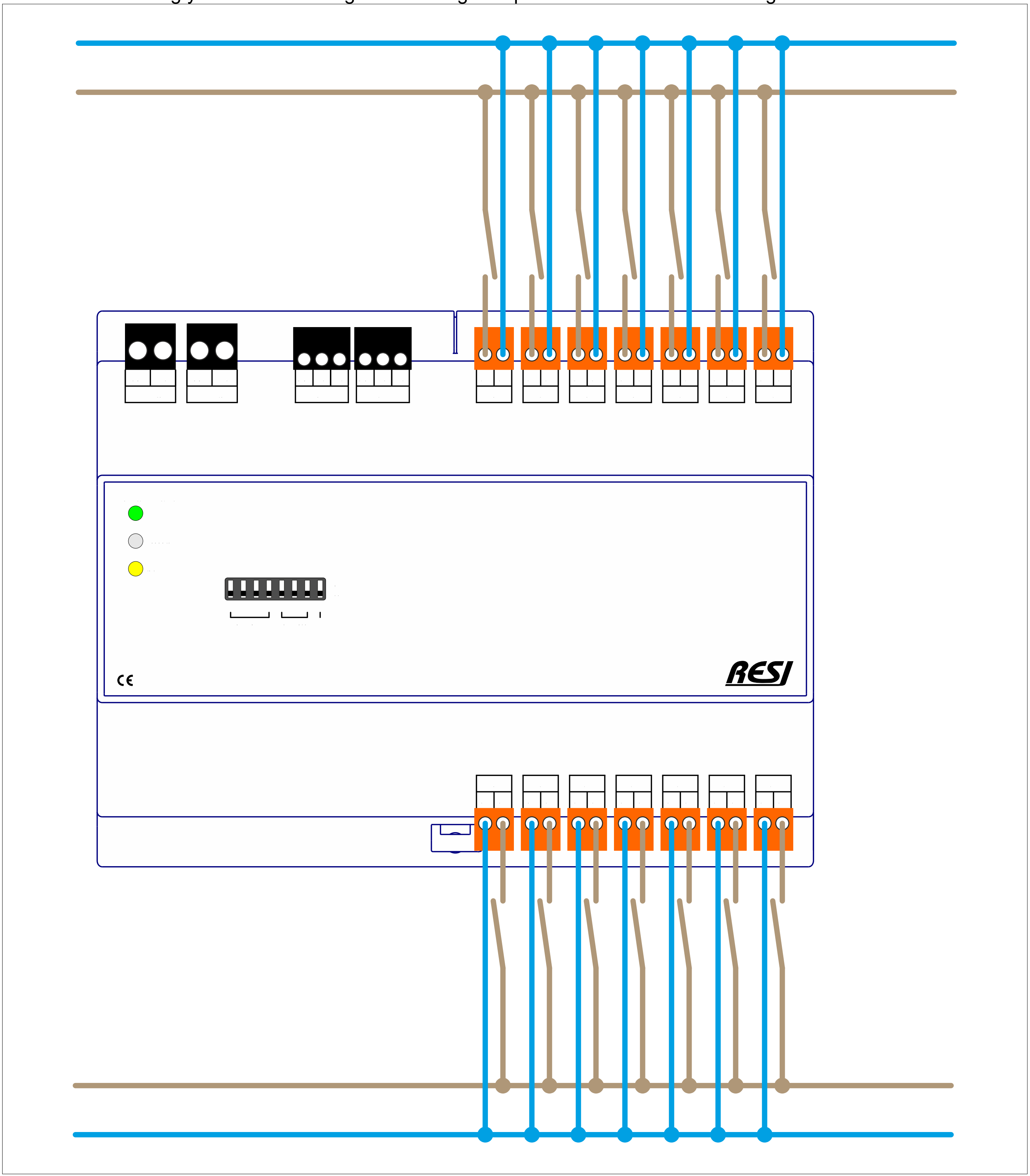


Figure: Cabling of the digital inputs of the IO module with AC signals

Don't forget, that you can use signals from different AC power supplies for each input, because all digital inputs are galvanically insulated to each other. Also you can mix AC and DC input signals on one module!

13.4.3 Mixed cabling of the digital inputs with AC and DC signals

In the below drawing you see the cabling of the 14 digital inputs of the module with AC and DC signals. For each digital input you can use a different power source:

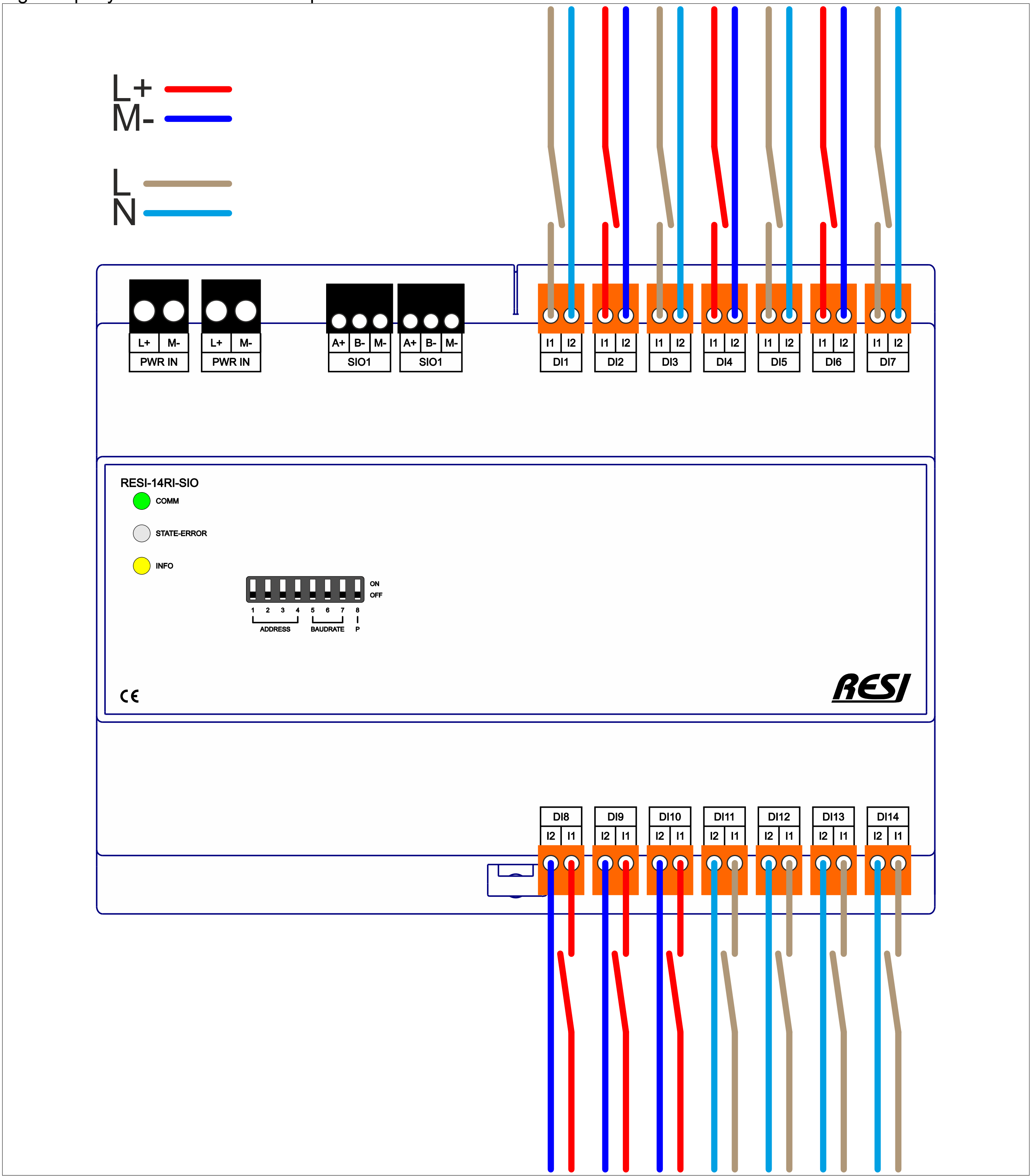


Figure: Mixed cabling of the digital inputs of the IO module with AC and DC signals

13.5 Additional MODBUS register & coils

Here you will find only the additional MODBUS registers and coils especially for this IO module. Please refer to the description of of the standard MODBUS mapping for more details about the available basic MODBUS registers and coils.

Please refer to the external document for detailed documentation of the current MODBUS register mapping for this IO module:

RESI-L-14RI-SIO-MODBUS+ASCII-ENxx.pdf

13.6 Additional ASCII commands

Here you will find only the additional ASCII commands especially for this IO module. Please refer to the description of of the standard commands for more details about the available basic ASCII commands.

Please refer to the external document for detailed documentation of the current ASCII commands for this IO module:

RESI-L-14RI-SIO-MODBUS+ASCII-ENxx.pdf

14 RESI-48RI-SIO

14.1 General information

This series of IO modules offer the following features:

- 48 digital inputs for 12-250Vac/dc signals
- Each digital input is galvanic insulated to all other digital inputs
- Each digital input is cabled via extra 2 pin removable terminal
- Galvanic insulated RS485 interface for communication with a host system

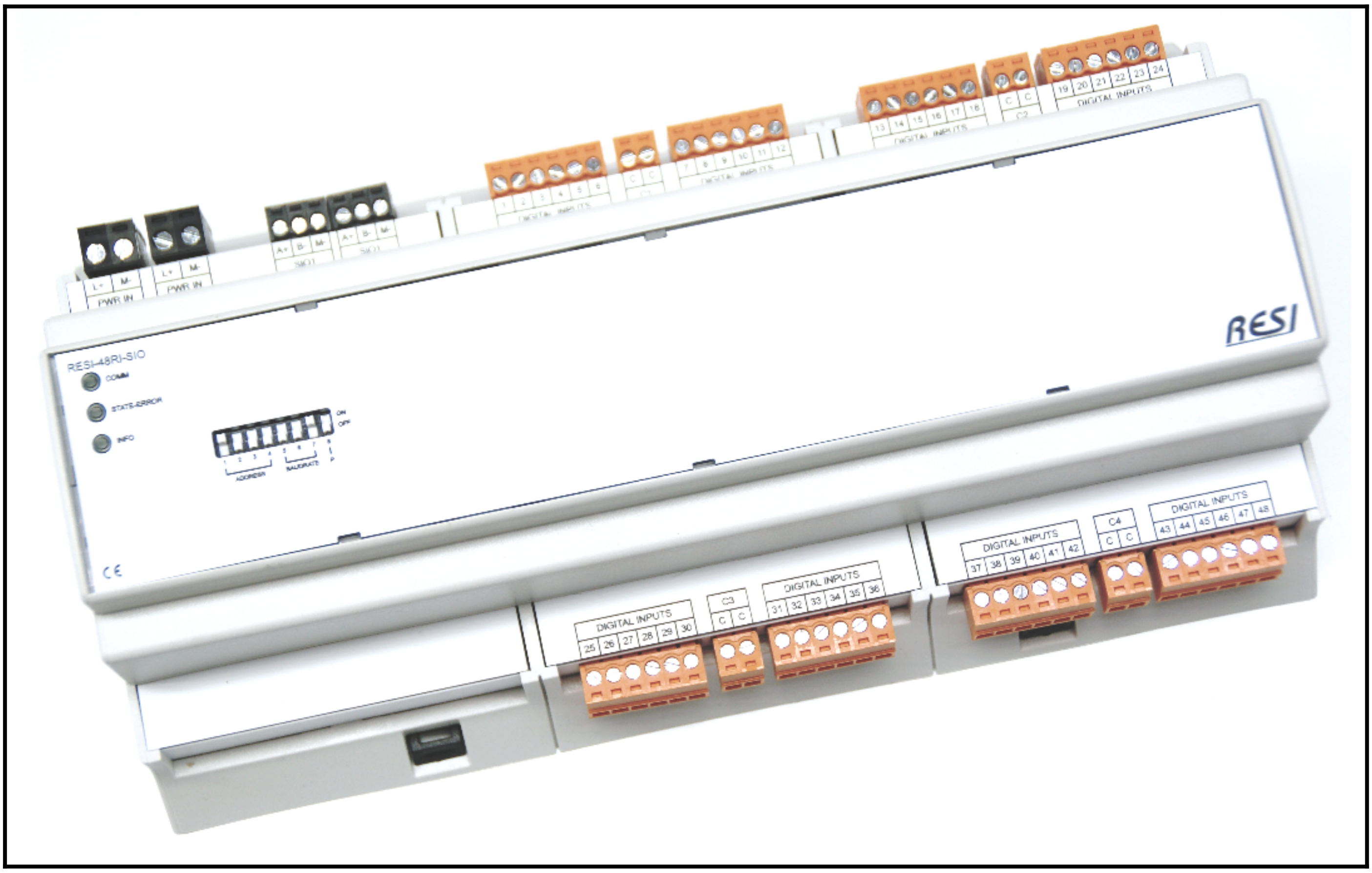


Figure: Our RESI-48RI-SIO module

14.2 Technical specification

Beside the basic technical data, which fulfil all of our IO modules, this IO modules meet the following technical specifications:

Power consumption	<0.5W
Product housing	BIG IO XT12
Product weight	425g
Digital inputs	
Total amount of inputs	48, organized in four groups with 12 digital inputs each
	Each input group can have different AC/DC power source
Sampling rate	Every 5ms
DC rating	
Input voltage range	12-250V= +/-10%
Input current	per channel
	approx. 0.7mA@12V=
	approx. 0.7mA@24V=
	approx. 0.7mA@32V=
	approx. 0.7mA@48V=
	approx. 0.7mA@250V=
Input power consumption	max. 0.2W/channel
Logic levels	0: <4.5V=
	1: >7.5V=
AC rating	
Input voltage range	12-250V= +/-10%
Input current	per channel
	approx. 0.7mA@12V~
	approx. 0.7mA@24V~
	approx. 0.7mA@48V~
	approx. 0.7mA@110V~
	approx. 0.7mA@230V~
	approx. 0.7mA@250V~
Input power consumption	max. 0.2W/channel
Logic levels	0: <4.5V~
	1: >7.5V~
Cable connection	Via eight 6-pin plug-in terminal blocks and four 2-pin terminal blocks
Terminal type	RM3.5
Galvanic insulation	Yes, to each other digital input group and to IO module
Default serial settings	
Baud rate	via DIP switch
Parity	none
Stop bit(s)	one
UnitID	255

14.3 Additional terminals & LED states

DIGITAL INPUTS	48 digital inputs for 12-250Vac/dc signals	
	for each digital input group with 12 digital inputs:	
	Two 6 pin plug-in terminal blocks and one 2-pin plug-in terminal blocks	
	Terminal type:	RM3.5
Digital input group #1		
Terminal block #1:	1..6:	Digital inputs #1 to #6
		0=open or short cut
		1=AC or DC voltage between 12 and 250V
Pin layout	Pin 1:	1: Digital input #1
	...	
	Pin 6:	6: Digital input #6
Terminal block #2:	C1:C	Common signal for digital inputs #1 to #12
Pin layout	Pin 1:	C: Common signal
	Pin 2:	C: Common signal
Terminal block #3:	7..12:	Digital inputs #7 to #12
		0=open or short cut
		1=AC or DC voltage between 12 and 250V
Pin layout	Pin 1:	7: Digital input #7
	...	
	Pin 6:	12: Digital input #12
Digital input groups #2 to #4		like Digital input group #1
INFO	This LED is on, if at least one of the digital inputs is high (1).	
	This LED is off, if all digital inputs are low (0).	

14.4 Connection diagram

14.4.1 Cabling of the digital inputs with DC signals

In the below drawing you see the cabling of the 48 digital inputs of the module with DC signals.

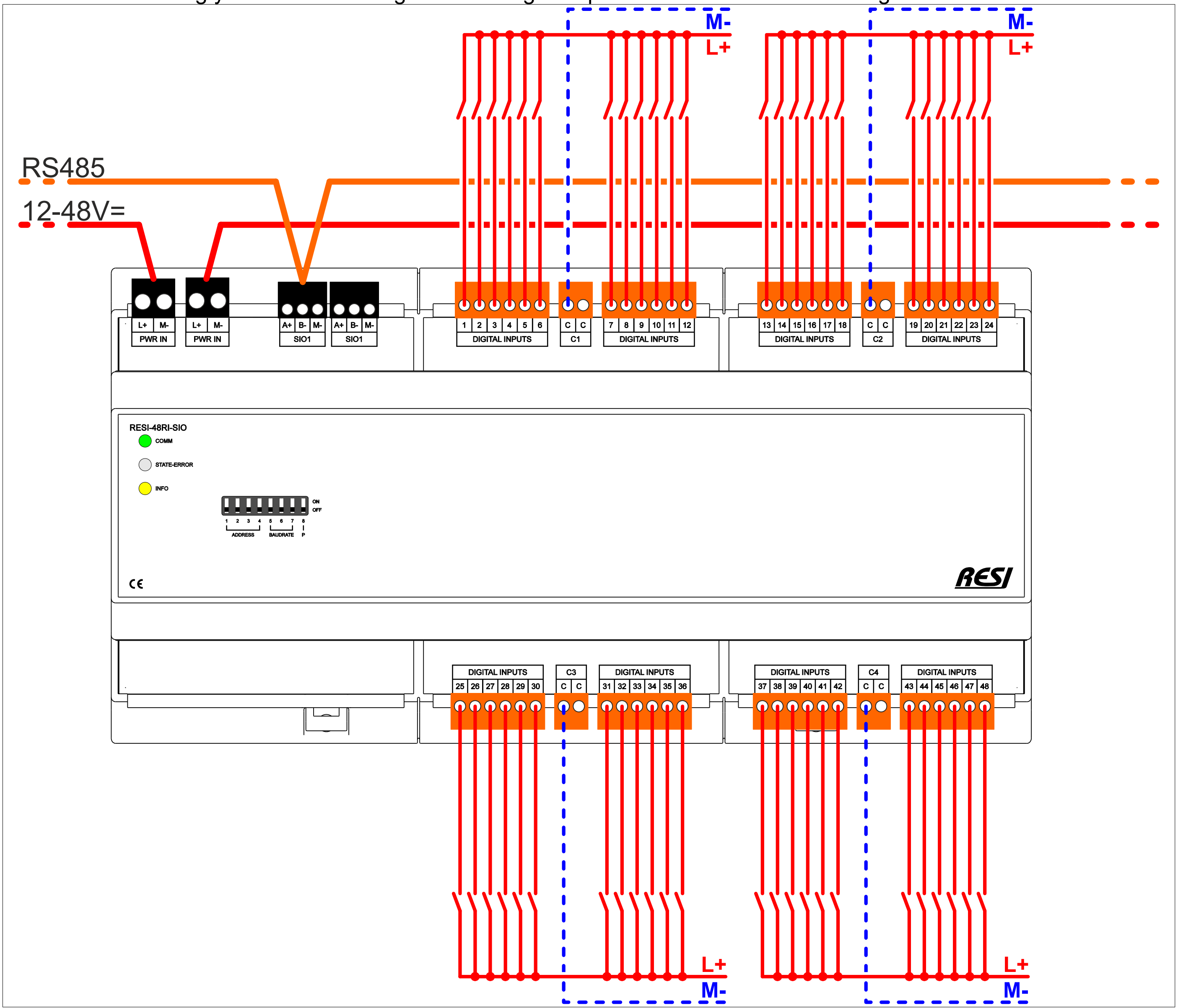


Figure: Cabling of the digital inputs of the IO module with DC signals

Don’t forget, that you can use signals from different DC power supplies for each input group, because all four digital input groups are galvanically insulated to each other group. Also you can mix AC and DC input groups on one module!

14.4.2 Cabling of the digital inputs with AC signals

In the below drawing you see the cabling of the 48 digital inputs of the module with AC signals.

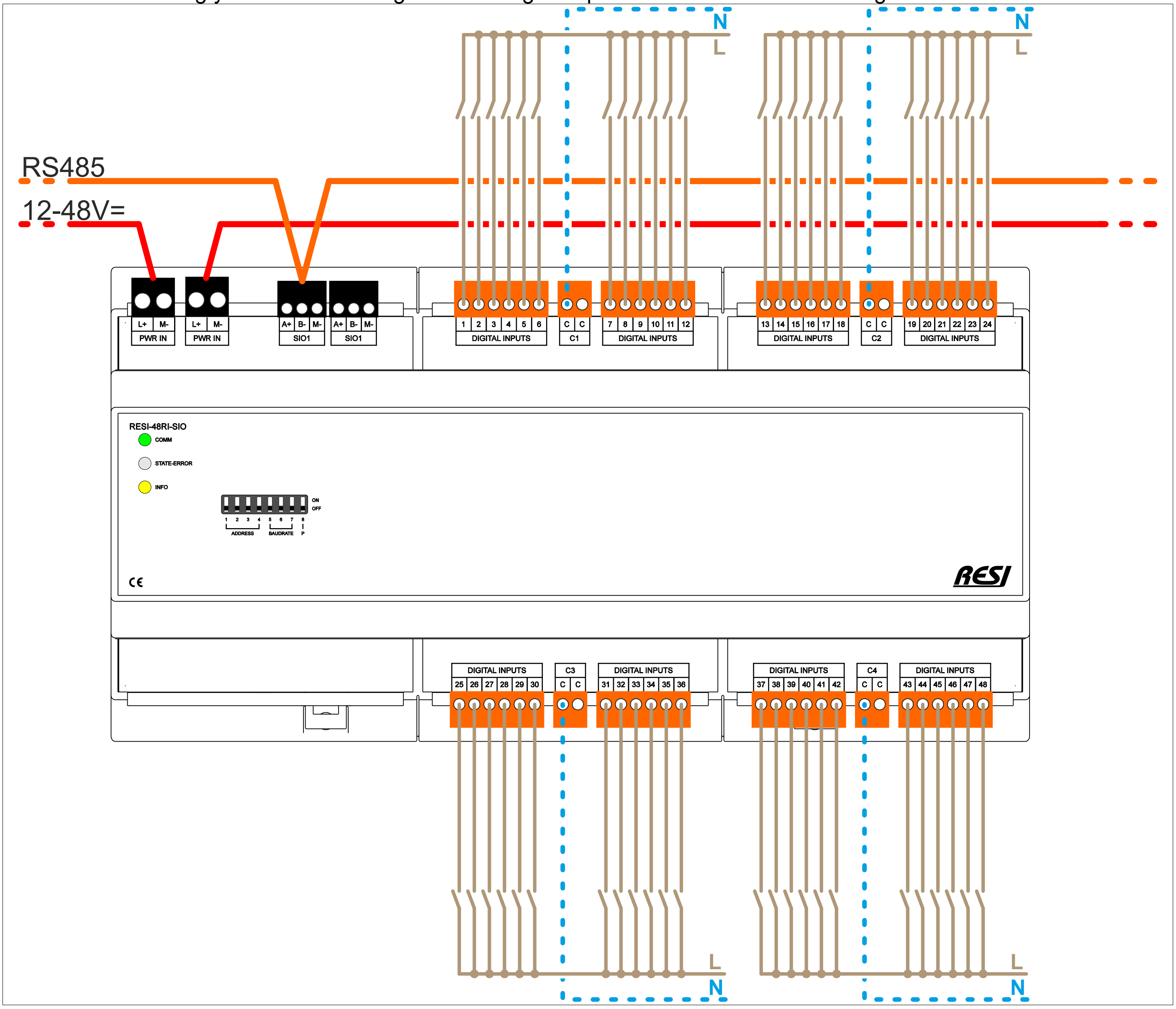


Figure: Cabling of the digital inputs of the IO module with AC signals

Don’t forget, that you can use signals from different AC power supplies for each digital input group, because all digital input groups are galvanically insulated to each other group. Also you can mix AC and DC input signals on one module!

14.4.3 Mixed cabling of the digital inputs with AC and DC signals

In the below drawing you see the cabling of the 48 digital inputs of the module with mixed AC and DC signals. For each of the four digital input groups yopu can use a different power source.

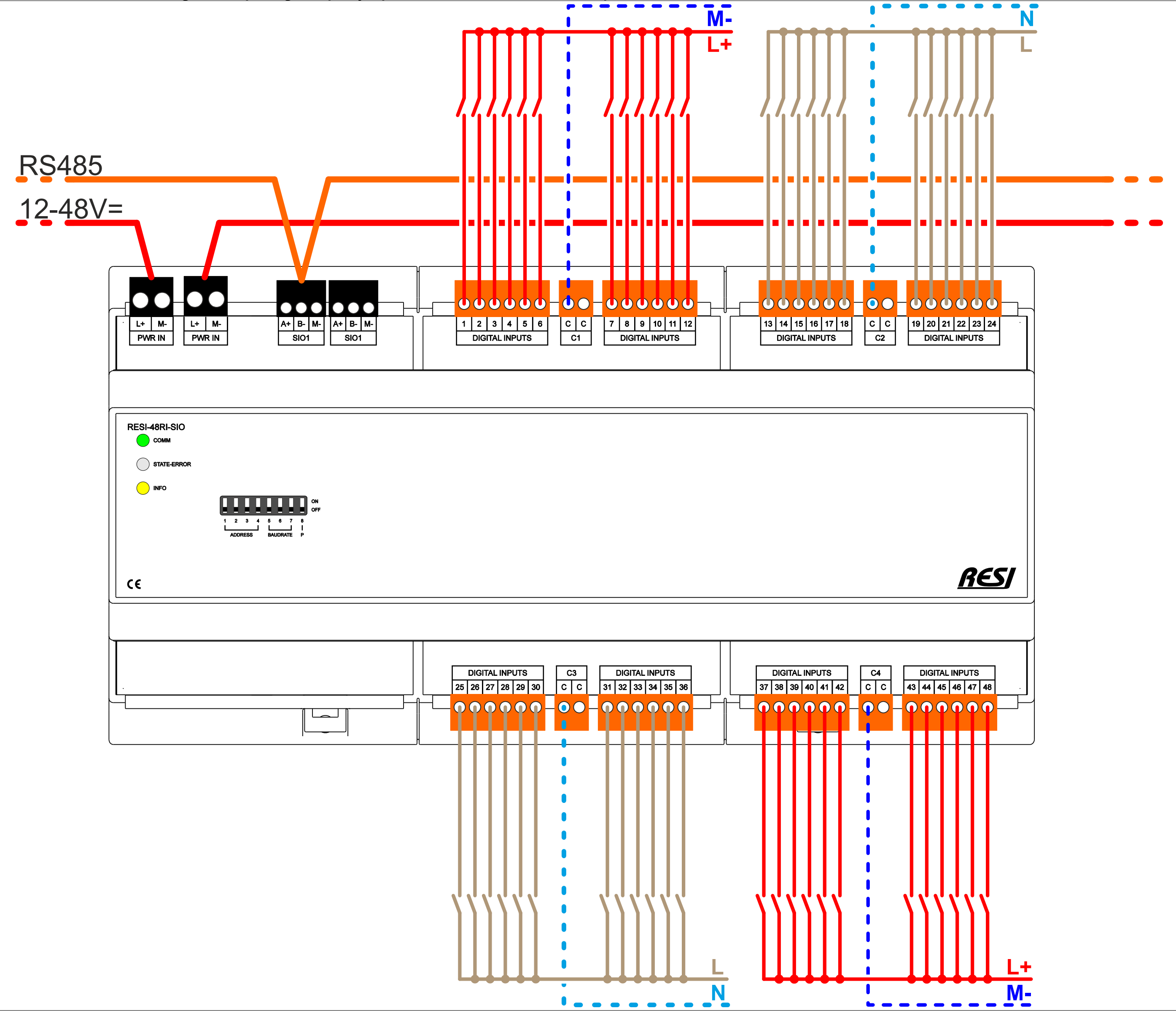


Figure: Cabling of the digital inputs of the IO module with mixed AC and DC signals

14.5 Additional MODBUS register & coils

Here you will find only the additional MODBUS registers and coils especially for this IO module. Please refer to the description of of the standard MODBUS mapping for more details about the available basic MODBUS registers and coils.

Please refer to the external document for detailed documentation of the current MODBUS register mapping for this IO module:

RESI-L-48RI-SIO-MODBUS+ASCII-ENxx.pdf

14.6 Additional ASCII commands

Here you will find only the additional ASCII commands especially for this IO module. Please refer to the description of of the standard commands for more details about the available basic ASCII commands.

Please refer to the external document for detailed documentation of the current ASCII commands for this IO module:

RESI-L-48RI-SIO-MODBUS+ASCII-ENxx.pdf

15 RESI-32DI-SIO,RESI-32DI-ETH

15.1 General information

This series of IO modules offer the following features:

- 32 digital inputs for 12-48Vdc signals
- No galvanic insulation to the rest of the module (Ground of digital inputs is tied to system ground)
- 16 digital inputs are grouped on an 18 pin removable terminal each
- RESI-32DI-SIO: Galvanic insulated RS485 interface for communication with a host system
- RESI-32DI-ETH: Ethernet interface for communication with a host system

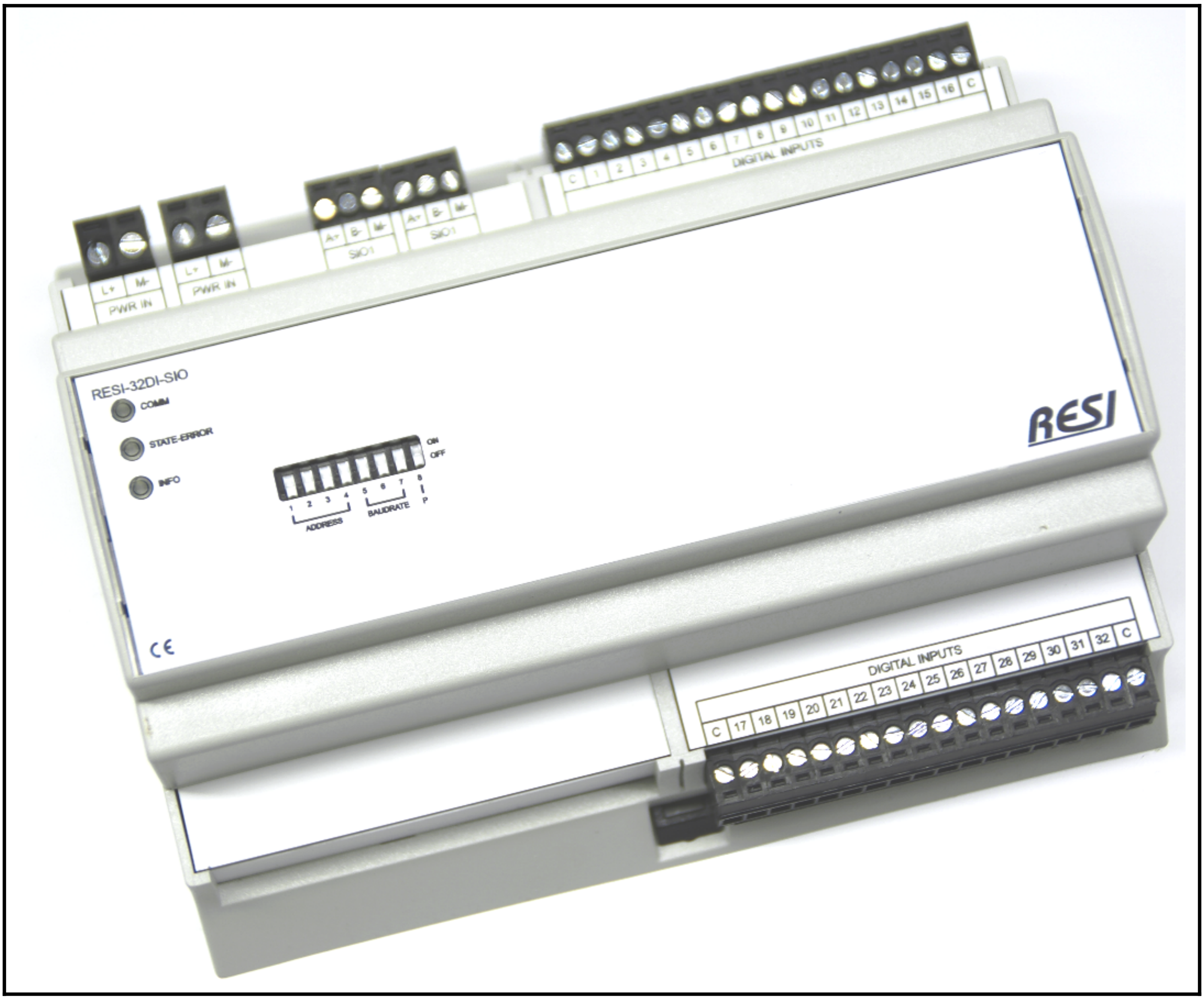


Figure: Our RESI-32DI-SIO module

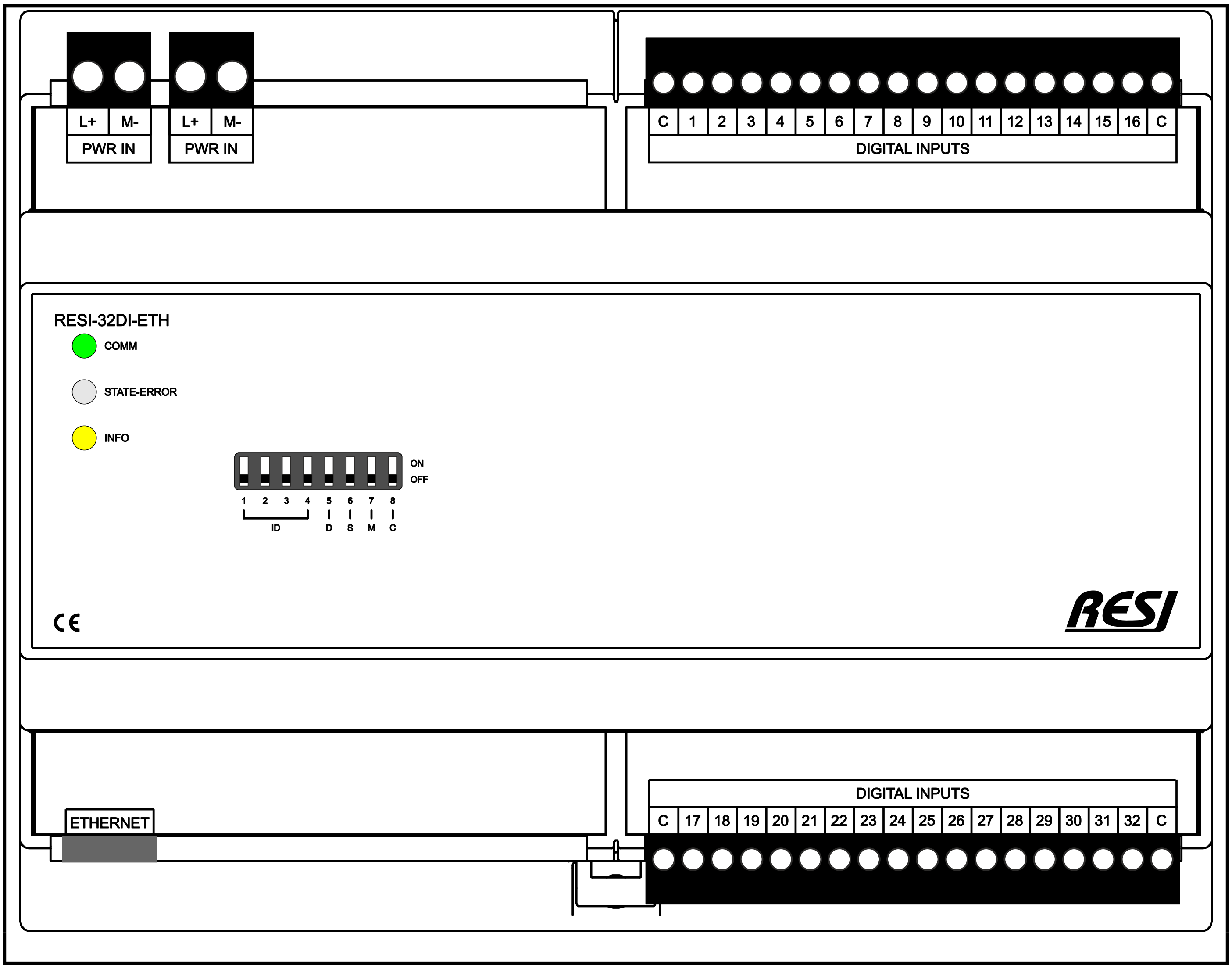


Figure: Our RESI-32DI-ETH module

15.2 Technical specification

Beside the basic technical data, which fulfill all of our IO modules, this IO modules meet the following technical specifications:

Power consumption	
RESI-32DI-SIO:	<0.3W
RESI-32DI-ETH:	<0.8W
Product housing	BIG IO XT8
Product weight	
RESI-32DI-SIO:	295g
RESI-32DI-ETH:	280g
Digital inputs	
Total amount of inputs	32
Sampling rate	Every 5ms
DC rating	
Input voltage range	12-48V= +/-10%
Input current	per channel
	approx. 0,8mA@12V=
	approx. 1.5mA@20V=
	approx. 1.8mA@24V=
	approx. 2.5mA@32V=
	approx. 4.0mA@48V=
Input power consumption	max. 0.3W/channel
Logic levels	0: <3.8V=
	1: >4.7V=
Cable connection	in two groups, 16 digital inputs each
	Via 2 18-pin plug-in terminal blocks
Terminal type	RM3.5
Galvanic insulation	No, ground of digital inputs is wired to ground of CPU system
Default serial settings	
Baud rate	via DIP switch
Parity	none
Stop bit(s)	one
UnitID	255
Default Ethernet settings	
IP address	192.168.1.13
IP mask	255.255.255.0
Gateway	192.168.1.1
UnitID	255
User	RESI
Password	RESI

15.3 Additional terminals & LED states

DIGITAL INPUTS	32 digital inputs for 12-48Vdc signals	
	Two 18 pin plug-in terminal blocks	
	Terminal type:	RM3.5
	C:	Common ground: wired to system ground
	1..32:	Digital input 1-32
		0=open or connected to ground
		1=DC voltage between 12 and 48V=
Pin layout	18 pin plug-in terminal #1	
	Pin 1:	C: Common ground
	Pin 2:	1: Digital input #1
	…	
	Pin 17:	16: Digital input #16
	Pin 18:	C: Common ground
	18 pin plug-in terminal #2	
	Pin 1:	C: Common ground
	Pin 2:	17: Digital input #17
	…	
	Pin 17:	32: Digital input #32
	Pin 18:	C: Common ground
INFO	If at least one of the digital inputs is activated (ON), this LED is ON.	
	If none of the digital inputs are activated (OFF), this LED is OFF.	

15.4 Connection diagram

15.4.1 Cabling of the digital inputs with DC signals

In the below drawing you see the cabling of the 32 digital inputs of the module with DC signals. All four terminals C are internally connected to the ground signal of the IO module.

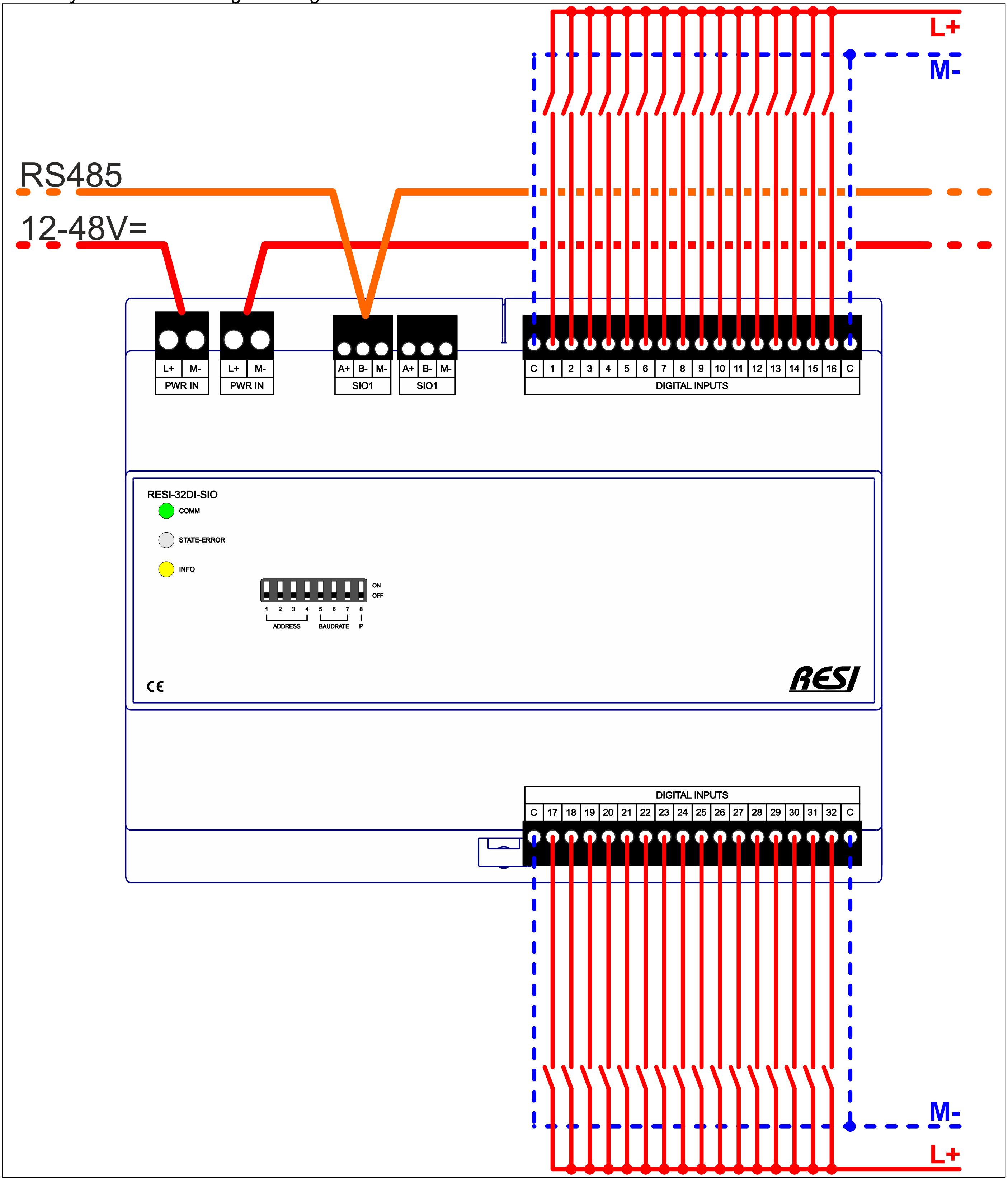


Figure: Cabling of the digital inputs of the IO module with DC signals

15.5 Additional MODBUS register & coils

Here you will find only the additional MODBUS registers and coils especially for this IO module. Please refer to the description of of the standard MODBUS mapping for more details about the available basic MODBUS registers and coils.

Please refer to the external document for detailed documentation of the current MODBUS register mapping for this IO module:

RESI-L-32DI-SIO-MODBUS+ASCII-ENxx.pdf
RESI-L-32DI-ETH-MODBUS+ASCII-ENxx.pdf

15.6 Additional ASCII commands

Here you will find only the additional ASCII commands especially for this IO module. Please refer to the description of of the standard commands for more details about the available basic ASCII commands.

Please refer to the external document for detailed documentation of the current ASCII commands for this IO module:

RESI-L-32DI-SIO-MODBUS+ASCII-ENxx.pdf
RESI-L-32DI-ETH-MODBUS+ASCII-ENxx.pdf

16 RESI-30DO-SIO,RESI-30DO-ETH

16.1 General information

This series of IO modules offer the following features:

- 30 digital outputs for DC signals ≤30V, ≤350mA
- Organized into two groups of 15 digital outputs with individual power supply ≤30V, ≤1.8A
- Galvanic insulation of each output group to the rest of the module
- 15 digital outputs are grouped on an 15 pin removable terminal block
- RESI-30DO-SIO: Galvanic insulated RS485 interface for communication with a host system
- RESI-30DO-ETH: Ethernet interface for communication with a host system

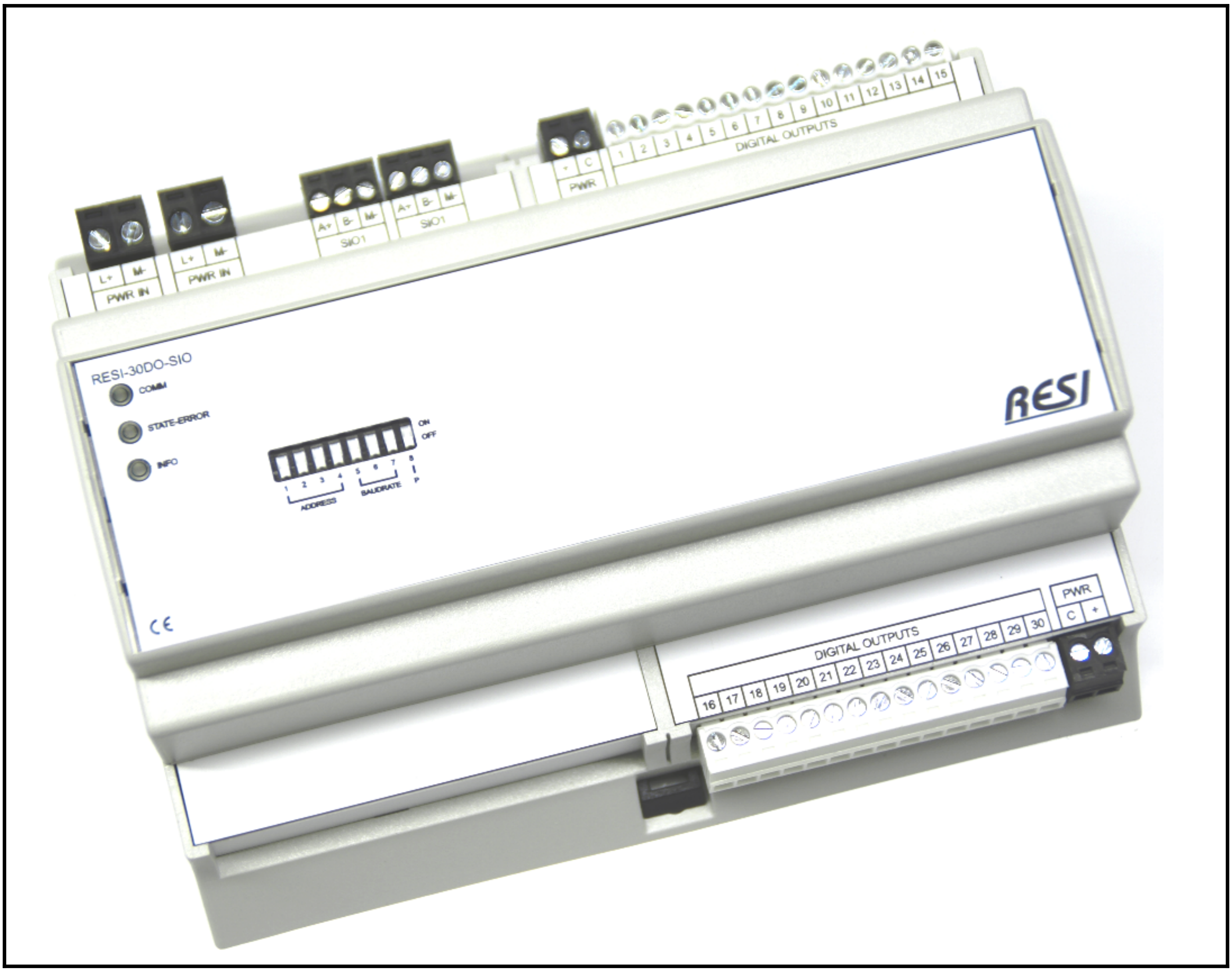


Figure: Our RESI-30DO-SIO module

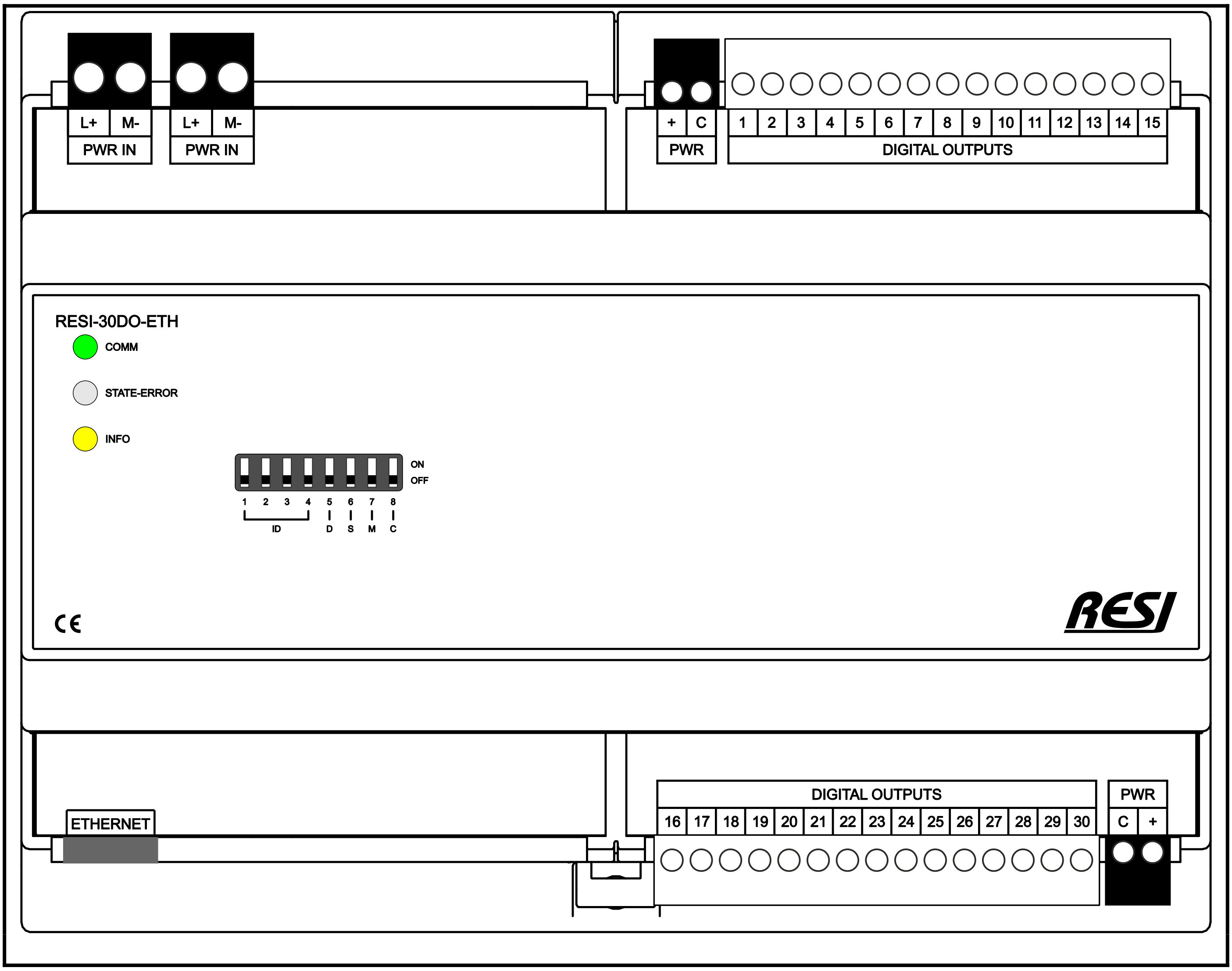


Figure: Our RESI-30DO-ETH module

16.2 Technical specification

Beside the basic technical data, which fulfill all of our IO modules, this IO modules meet the following technical specifications:

Power consumption	
RESI-30DO-SIO:	<0.3W
RESI-33DO-ETH:	<0.8W
Product housing	BIG IO XT8
Product weight	
RESI-30DO-SIO:	295g
RESI-30DO-ETH:	285g
Digital outputs	
Total amount of outputs	30
Groups	2 groups with independent power supplies
Power supply for output group	0-30V= +/-10%, ≤1.8A
Update rate	≤2ms
DC rating	
Output voltage range	0-30V= +/-10%
Output current	max. 350mA/channel, max. 1.8A/group
Output diagnose	Yes
Output temperature protection	Yes
Cable connection	in two groups, 15 digital outputs each
	Via 2 15-pin plug-in terminal blocks for the digital outputs
	Via 2 2-pin plug-in terminal blocks for the power supply
Terminal type	RM3.5
Galvanic insulation	Yes, each output group has its own power supply and is insulated
	to the rest of the module
Default serial settings	
Baud rate	via DIP switch
Parity	none
Stop bit(s)	one
UnitID	255
Default Ethernet settings	
IP address	192.168.1.16
IP mask	255.255.255.0
Gateway	192.168.1.1
UnitID	255
User	RESI
Password	RESI

16.3 Additional terminals & LED states

DIGITAL OUTPUTS	30 digital outputs for DC output signals ≤30V, ≤350mA/channel	
	Two 15 pin plug-in terminal blocks for the digital outputs	
	Terminal type:	RM3.5
	1..15:	Digital outputs 1-15
	16..31:	Digital outputs 16-31
		DC Signal, ≤30V, ≤350mA
DIGITAL OUTPUTS		
POWER SUPPLY	Two 2 pin plug-in terminal blocks for the power supply of the digital output group	
	Terminal type:	RM3.5
	PWR:+:	Power supply input ≤30V, ≤1.8A
	PWR:C:	Power supply ground signal
Pin layout	2 pin plug-in terminal #1	
	Pin 1:	PWR:+: Power supply for DOs 1-15 ≤30V, ≤1.8A
	Pin 2:	PWR:C: Power supply ground for DOs 1-15
	18 pin plug-in terminal #2	
	Pin 1:	1: Digital output #1, DC Signal, ≤30V, ≤350mA
	Pin 2:	2: Digital output #2
	…	
	Pin 14:	14: Digital output #14
	Pin 15:	15: Digital output #15
	2 pin plug-in terminal #3	
	Pin 1:	PWR:+: Power supply for DOs 16-30 ≤30V, ≤1.8A
	Pin 2:	PWR:C: Power supply ground for DOs 16-30
	18 pin plug-in terminal #4	
	Pin 1:	1: Digital output #16,DC Signal, ≤30V, ≤350mA
	Pin 2:	2: Digital output #17
	…	
	Pin 14:	14: Digital output #29
	Pin 15:	15: Digital output #30
INFO	If alt least one of the digital outputs is activated (ON), this LED is ON.	
	If none of the digital outputs are activated (OFF), this LED is OFF.	

16.4 Connection diagram

16.4.1 Cabling of the digital inputs with DC signals

In the below drawing you see the cabling of the 32 digital inputs of the module with DC signals. All four terminals C are internally connected to the ground signal of the IO module.

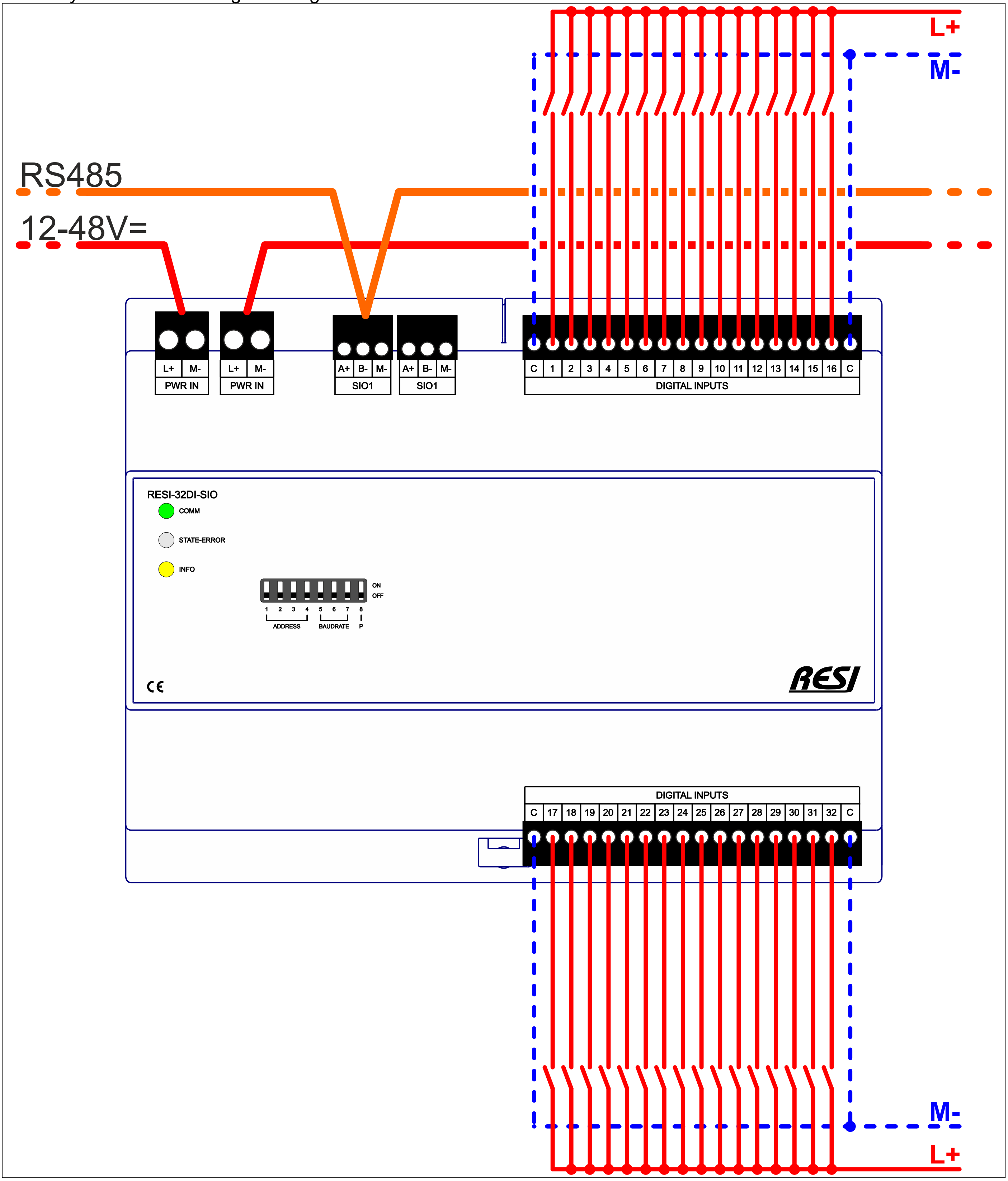


Figure: Cabling of the digital inputs of the IO module with DC signals

16.5 Additional MODBUS register & coils

Here you will find only the additional MODBUS registers and coils especially for this IO module. Please refer to the description of of the standard MODBUS mapping for more details about the available basic MODBUS registers and coils.

Please refer to the external document for detailed documentation of the current MODBUS register mapping for this IO module:

RESI-L-30DO-SIO-MODBUS+ASCII-ENxx.pdf
RESI-L-30DO-ETH-MODBUS+ASCII-ENxx.pdf

16.6 Additional ASCII commands

Here you will find only the additional ASCII commands especially for this IO module. Please refer to the description of of the standard commands for more details about the available basic ASCII commands.

Please refer to the external document for detailed documentation of the current ASCII commands for this IO module:

RESI-L-30DO-SIO-MODBUS+ASCII-ENxx.pdf
RESI-L-30DO-ETH-MODBUS+ASCII-ENxx.pdf

17 RESI-64DI-SIO

17.1 General information

This series of IO modules offer the following features:

- 64 digital inputs for 12-48Vdc signals
- No galvanic insulation to the rest of the module (Ground of digital inputs is tied to system ground)
- 16 digital inputs are grouped on an 18 pin removable terminal each
- Galvanic insulated RS485 interface for communication with a host system

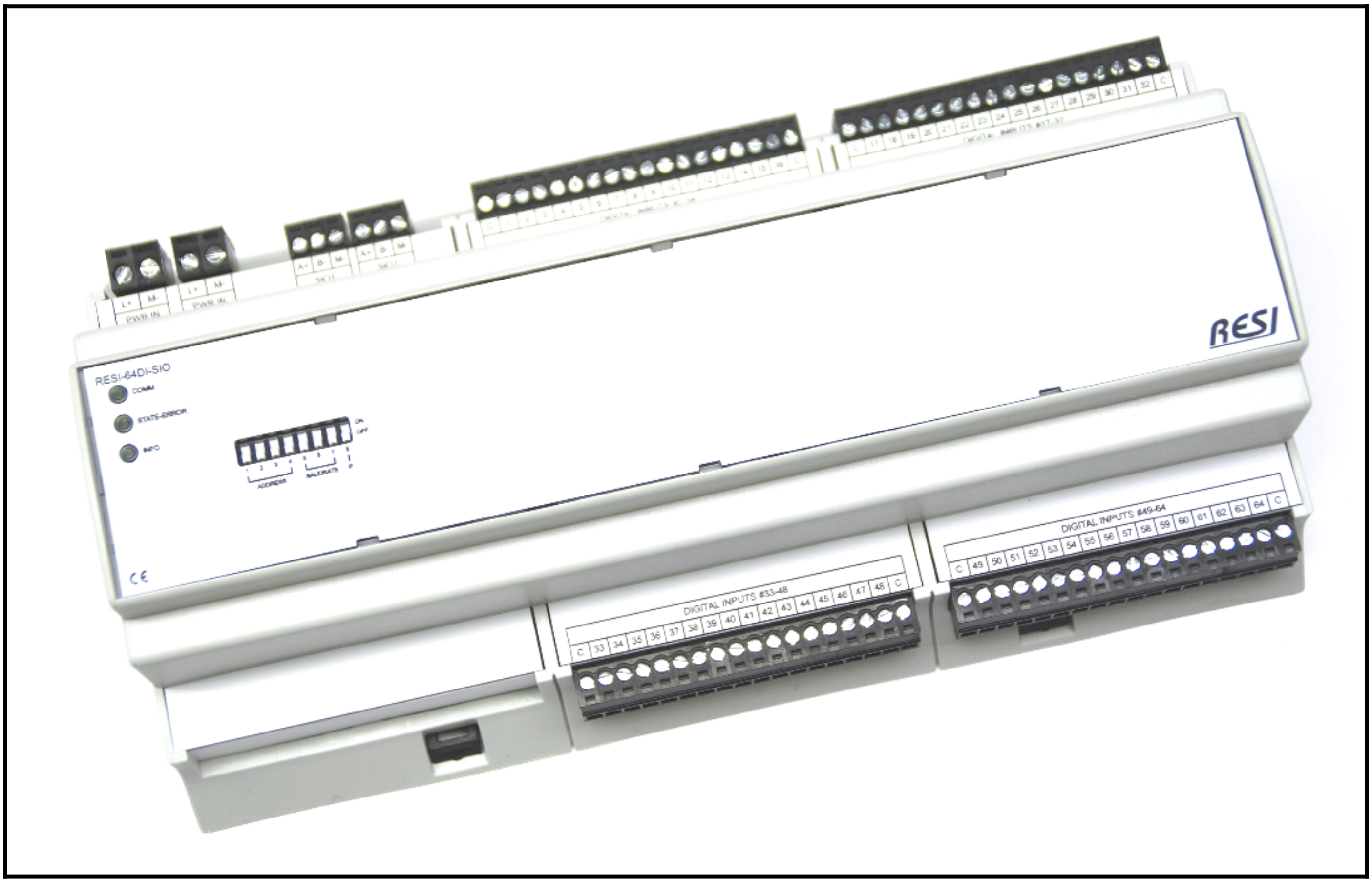


Figure: Our RESI-64DI-SIO module

17.2 Technical specification

Beside the basic technical data, which fulfil all of our IO modules, this IO modules meet the following technical specifications:

Power consumption	<0.3W
Product housing	BIG IO XT12
Product weight	430g
Digital inputs	
Total amount of inputs	64
Sampling rate	Every 5ms
DC rating	
Input voltage range	12-48V= +/-10%
Input current	per channel
	approx. 0,8mA@12V=
	approx. 1.5mA@20V=
	approx. 1.8mA@24V=
	approx. 2.5mA@32V=
	approx. 4.0mA@48V=
Input power consumption	max. 0.3W/channel
Logic levels	0: <3.8V=
	1: >4.7V=
Cable connection	in four groups, 16 digital inputs each
	Via 4 18-pin plug-in terminal blocks
Terminal type	RM3.5
Galvanic insulation	No, ground of digital inputs is wired to ground of CPU system
Default serial settings	
Baud rate	via DIP switch
Parity	none
Stop bit(s)	one
UnitID	255

17.3 Additional terminals & LED states

DIGITAL INPUTS	64 digital inputs for 12-48Vdc signals	
	Four 16 pin plug-in terminal blocks	
	Terminal type:	RM3.5
	C:	Common ground: wired to system ground
	1..64:	Digital input 1-64
		0=open or connected to ground
		1=DC voltage between 12 and 48V=
Pin layout	16 pin plug-in terminal #1	
	Pin 1:	C: Common ground
	Pin 2:	1: Digital input #1
	…	
	Pin 17:	16: Digital input #16
	Pin 18:	C: Common ground
	16 pin plug-in terminal #2	
	Pin 1:	C: Common ground
	Pin 2:	17: Digital input #17
	…	
	Pin 17:	32: Digital input #32
	Pin 18:	C: Common ground
	16 pin plug-in terminal #3	
	Pin 1:	C: Common ground
	Pin 2:	33: Digital input #33
	…	
	Pin 17:	48: Digital input #48
	Pin 18:	C: Common ground
	16 pin plug-in terminal #4	
	Pin 1:	C: Common ground
	Pin 2:	49: Digital input #49
	…	
	Pin 17:	64: Digital input #64
	Pin 18:	C: Common ground
INFO	This LED is on, if at least one of the digital inputs is high (1).	
	This LED is off, if all digital inputs are low (0).	

17.4 Connection diagram

17.4.1 Cabling of the digital inputs with DC signals

In the below drawing you see the cabling of the 64 digital inputs of the module with DC signals. All eight terminals C are internally connected to the ground signal of the IO module.

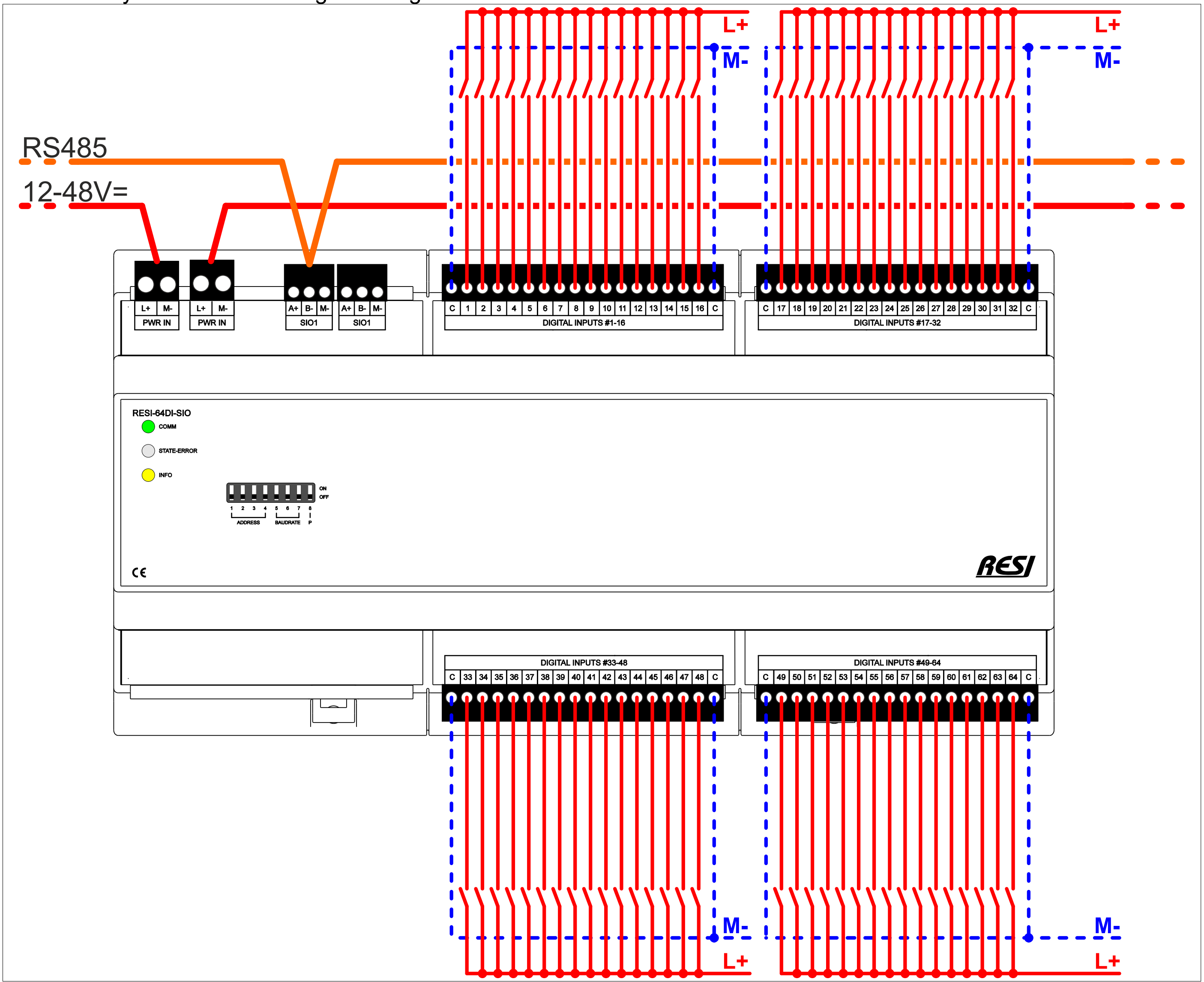


Figure: Cabling of the digital inputs of the IO module with DC signals

17.5 Additional MODBUS register & coils

Here you will find only the additional MODBUS registers and coils especially for this IO module. Please refer to the description of of the standard MODBUS mapping for more details about the available basic MODBUS registers and coils.

Please refer to the external document for detailed documentation of the current MODBUS register mapping for this IO module:

RESI-L-64DI-SIO-MODBUS+ASCII-ENxx.pdf

17.6 Additional ASCII commands

Here you will find only the additional ASCII commands especially for this IO module. Please refer to the description of of the standard commands for more details about the available basic ASCII commands.

Please refer to the external document for detailed documentation of the current ASCII commands for this IO module:

RESI-L-64DI-SIO-MODBUS+ASCII-ENxx.pdf

18 RESI-8CO-SIO

18.1 General information

This series of IO modules offer the following features:

- 8 mono stable relay outputs with special power relays
- 3 clamps per relay: NO contact, NC contact and common root contact (C)
- Switching power per relay output: max. 30Vdc, max. 250Vac, max. 8A
- Contact material AgSnO2
- Each relay output is cabled via extra 3 pin removable terminal
- Galvanic insulated RS485 interface for communication with a host system

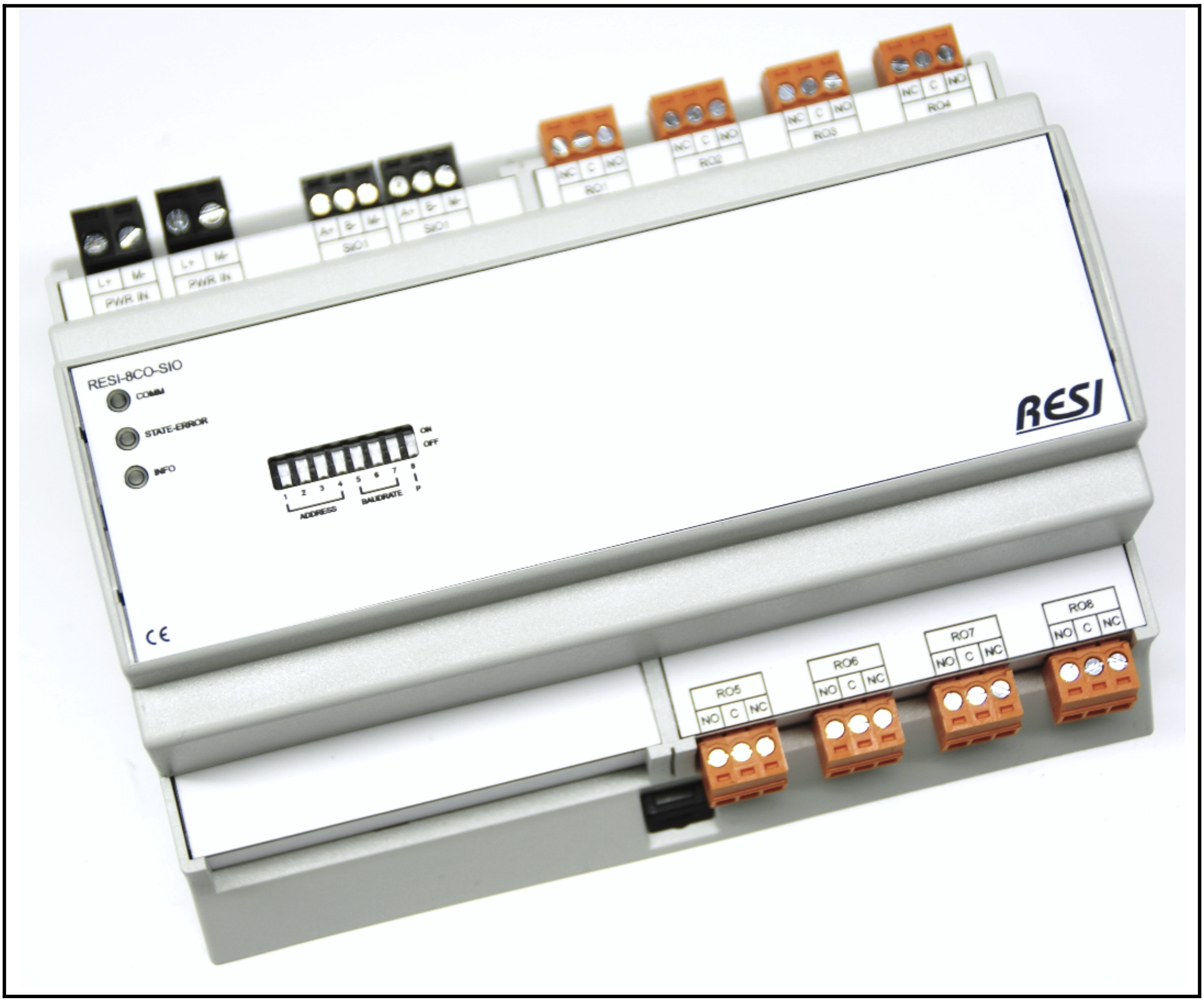


Figure: Our RESI-8CO-SIO module

18.2 Technical specification

Beside the basic technical data, which fulfil all of our IO modules, this IO modules meet the following technical specifications:

Power consumption	<2.5W
Product housing	BIG IO XT8
Product weight	325g
Relay outputs	
Total amount of outputs	8
Relay type	mono stable relay with contacts for NO clamp,
	NC clamp and common root clamp
Maximum output voltage	250Vac or 30Vdc
Maximum output current	8A
Switching cycles	10^7 switching cycles
Contact material	$AgSnO_2$
Cable connection	Via 8 3-pin plug-in terminal block
Terminal type	RM3.5
Galvanic insulation	Yes, via relais
Default serial settings	
Baud rate	via DIP switch
Parity	none
Stop bit(s)	one
UnitID	255

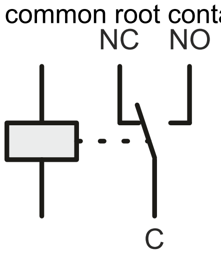
18.3 Additional terminals & LED states

RELAY OUTPUTS	8 relay outputs for 250Vac/30Vdc signals	
	Eight 3 pin plug-in terminal blocks	
	Terminal type:	RM3.5
	RO1..Ro8	
	NO:	Normally open switching contact of the relay
		=OFF: opened, =ON: closed
	C:	Common root contact of the relay
	NC:	Normally closed switching contact of the relay
Pin layout	Pin 1:	NO
	Pin 2:	C
	Pin 3:	NC
INFO	This LED is on, if at least one of the digital outputs is high (on) (1).	
	This LED is off, if all digital outputs are low (off) (0).	

18.4 Connection diagram

18.4.1 Cabling of the relay outputs of the module

In the below drawing you see the cabling of the 8 relay outputs of the module. Each relay offers three contacts: One common root contact, one closing contact (NO) and one opening contact (NC).



If the relay is OFF (powerless), the NC contact is tied to the common root contact (C) and the NO contact is open.

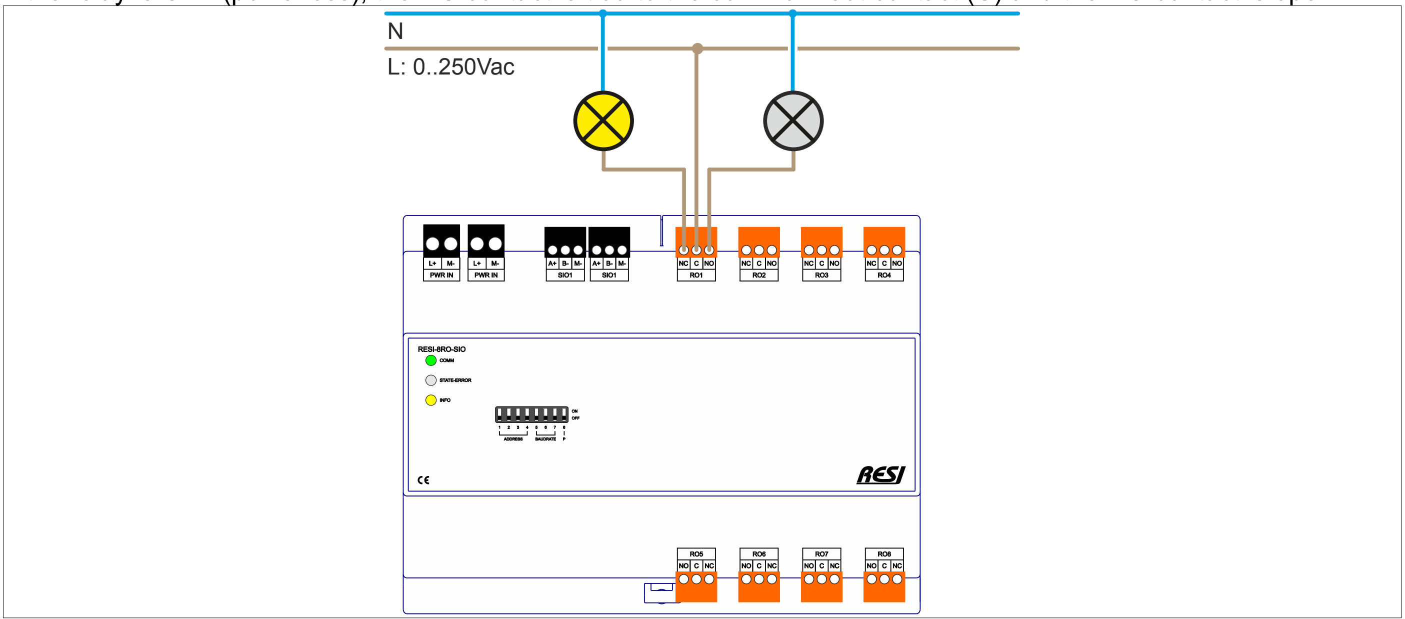


Figure: Cabling of the relay output 1, relay is OFF

If the relay is under power (ON), then the NC contact is open, and the NO contact is tied to the common root contact (C).

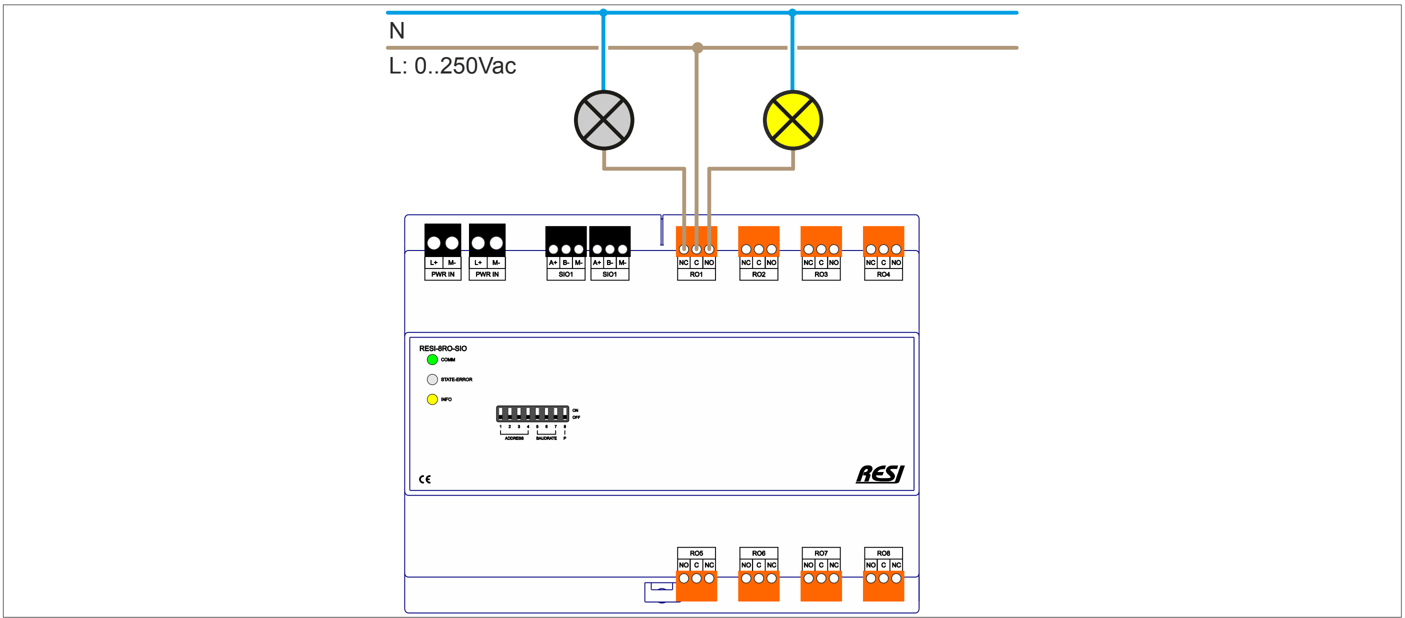


Figure: Cabling of the relay output 1, relay is ON

The following illustration shows the cabling of all 8 relays using only the NO contact. Only if the relay is ON, the current flows from the root contact to the switching contact NO to the consumer.

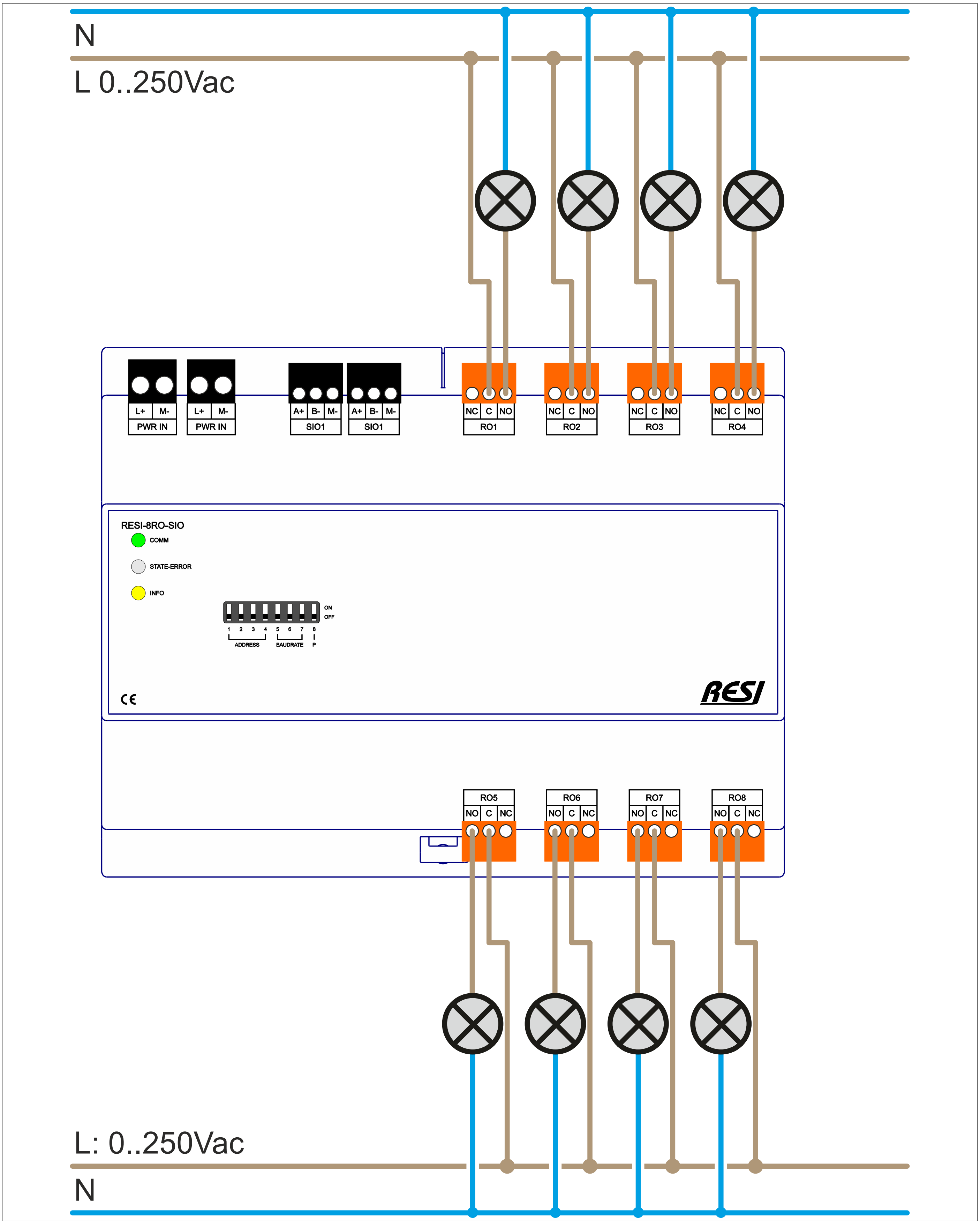


Figure: Cabling of all 8 relay outputs using the NO contact, all 8 relays are OFF

Here we show a DC cabling of all 8 relay with the NO contacts. Of course you can mix AC and DC signals on the relay outputs of the modules.

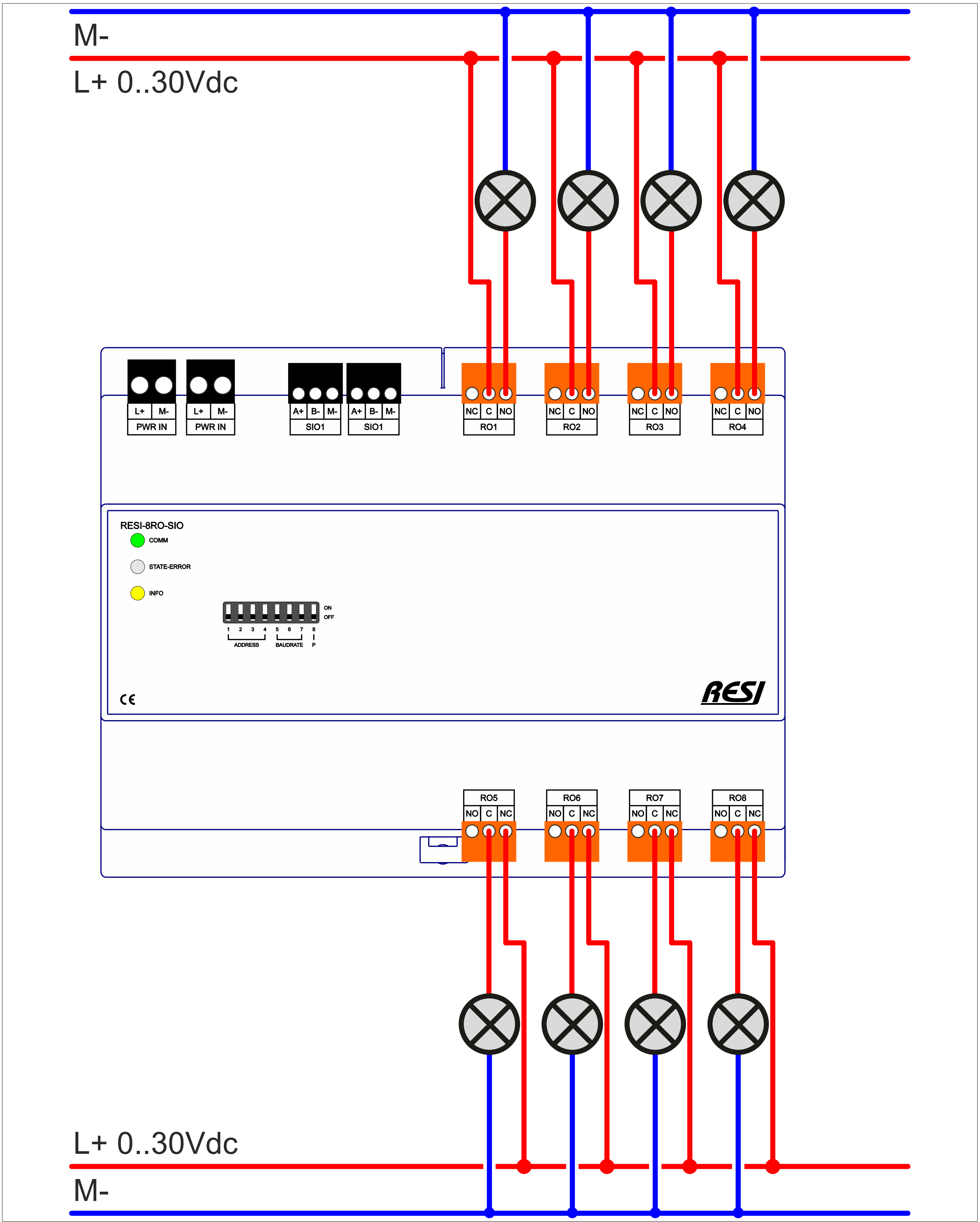


Figure: Cabling of all 8 relay outputs using the NO contact, all 8 relays are OFF

18.5 Additional MODBUS register & coils

Here you will find only the additional MODBUS registers and coils especially for this IO module. Please refer to the description of of the standard MODBUS mapping for more details about the available basic MODBUS registers and coils.

Please refer to the external document for detailed documentation of the current MODBUS register mapping for this IO module:

RESI-L-8CO-SIO-MODBUS+ASCII-ENxx.pdf

18.6 Additional ASCII commands

Here you will find only the additional ASCII commands especially for this IO module. Please refer to the description of of the standard commands for more details about the available basic ASCII commands.

Please refer to the external document for detailed documentation of the current ASCII commands for this IO module:

RESI-L-8CO-SIO-MODBUS+ASCII-ENxx.pdf

19 RESI-S16DI8PO-SIO, RESI-S8PO-SIO

19.1 General information

This series of IO modules offer the following features:

- Only RESI-S16DI8PO-SIO: 16 digital inputs for 12-48Vdc signals
- 8 bistable relay outputs with special power relays
- Maximum switching power: max. 250Vac, max. 16A, max 200µF
- Internal FRAM memory to save the last relay position
- Automatic recovery of the correct relay position after power loss
- Remanent counter for each output counting the switching cycles of the relays
- Only RESI-S16DI8PO-SIO:
  - Stand-alone operation mode: Internal logic functions between the digital inputs and the relay outputs
  - Configure simple logic functions like switch light on/off, central light on, central light off, stairway light with off delay timer, etc. with push buttons
- Galvanic insulated RS485 interface for communication with a host system

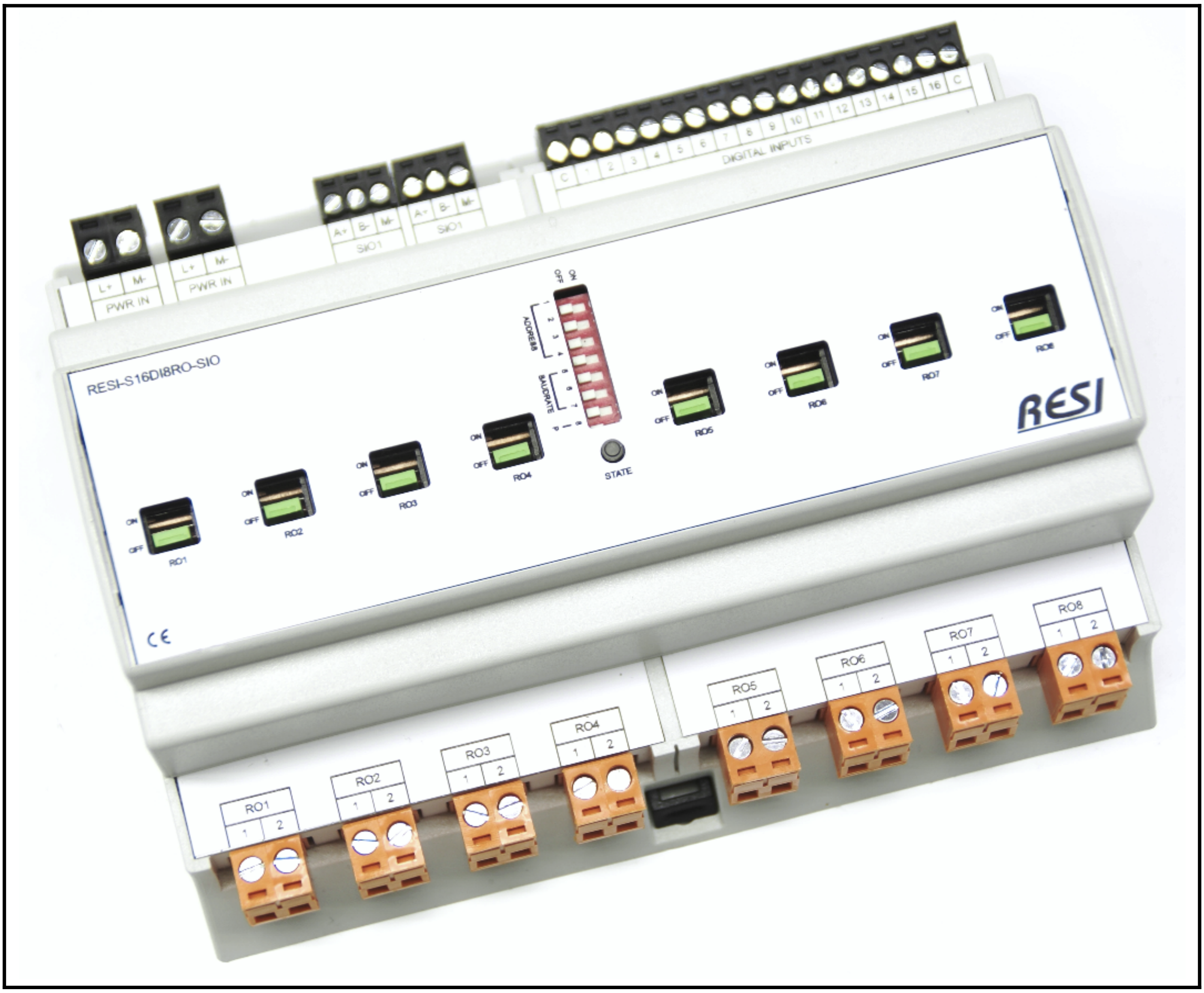


Figure: Our RESI-S16DI8PO-SIO module

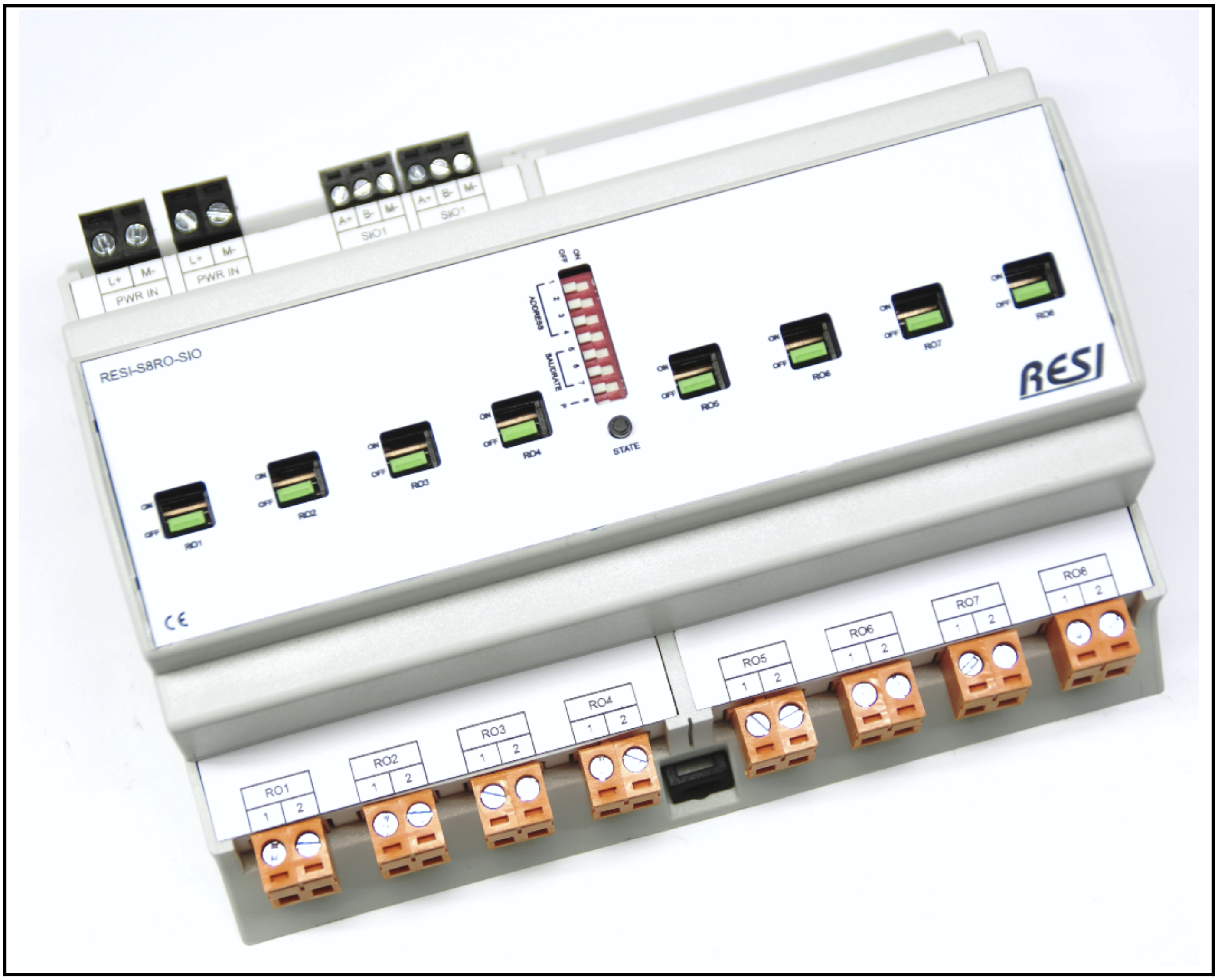


Figure: Our RESI-S8PO-SIO module

19.2 Internal logic functions

The IO module offers internal logic functions, which are handled by the module autonomous. All parameters for this logic functions are stored in the internal permanent memory FRAM. After a power loss all this configuration is not deleted and the module executes the logic functions again.
This internal logic functions can operate side by side with control commands via MODBUS/RTU or ASCII.

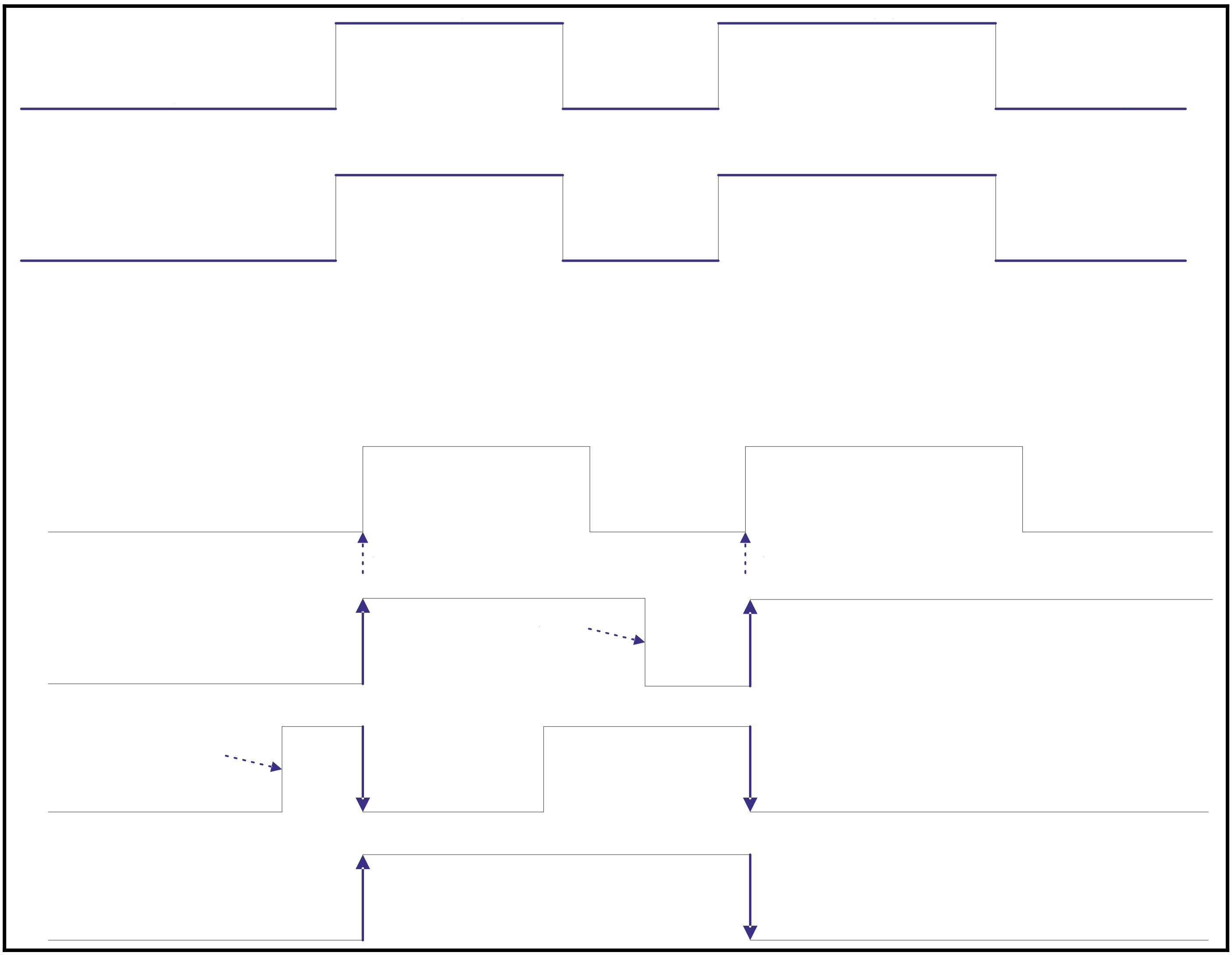


Figure: Internal logic functions

19.2.1 Switch on or off the internal logic processing

There is a general switch to enable or disable the execution of the internal logic operations. Therefore on the MODBUS/RTU interface you will find the register ENABLE LOGIC FUNCTIONS (4x21001). On the ASCII protocol the command SET SPECIAL MODE and GET SPECIAL MODE controls this feature.

Only if this register contains 1, the internal logic is executed by the module. Of course you will need a correct configuration for a desired logic function, if the module should react to a digital input.

- Activate logic function: Write to the MODBUS register ENABLE LOGIC FUNCTIONS the value 1 or execute the ASCII command SET SPECIAL MODE:1
- Deactivate logic function: Write to the MODBUS Register ENABLE LOGIC FUNCTIONS the value 0 or execute the ASCII command SET SPECIAL MODE:0
- Request the current execution status of logic function: Read out the current value in the MODBUS register ENABLE LOGIC FUNCTIONS. If this value is 1, the module executes the internal logic functions. If this value is 0, no logic functions are executed. Or you request the current status with the ASCII command GET SPECIAL MODE. If the answer is GSMODE:1,0x1, the internal logic is executed by the module. If the answer is GSMODE:0,0x0, no logic execution is active.

19.2.2 Reset internal logic

Sometimes it is very convenient to delete the complete configuration of the internal logic functions. This is handled by the ASCII command RESET SPECIAL MODE. On the MODBUS side you have to write the value 1 to the register CLEAR ALL LOGIC FUNCTIONS (4x21002). The module deletes the complete internal configuration permanently in the FRAM memory and no logic functions are executed.

19.2.3 Logic function SWITCH

This is the simplest logic function. You can map for each relay output a digital input. If this digital input is high (1), the corresponding output relay will be switched on. If this digital input is low (0), the mapped output relay will be switched off.

Example: Switch the output relay RO1 on and off with the digital input DI1

Over the ASCII interface you have to send the following commands:

PC->IO: #SET SWITCH1:0x0001

IO->PC: #OK

PC->IO: #SET SPECIAL MODE:1

IO->PC: #OK

Via the MODBUS interface you have to set the following registers:

PC->IO: Write value 0x0001 in MODBUS register SWITCH RO1 (4x20001)

PC->IO: Write value 0x0001 in MODBUS register ENABLE LOGIC FUNCTIONS (4x21001)

The other relay outputs are not affected by this configuration.

Example: Switch the output relay DO1 with digital input DI1 on and off, with DI2 the relay RO2, with DI3 the relay RO3 and so on.

Over the ASCII interface you have to send the following commands:

PC->IO: #SET SWITCH1:0x0001

IO->PC: #OK

PC->IO: #SET SWITCH2:0x0002

IO->PC: #OK

PC->IO: #SET SWITCH3:0x0004

IO->PC: #OK

PC->IO: #SET SWITCH4:0x0008

IO->PC: #OK

PC->IO: #SET SWITCH5:0x0010
IO->PC: #OK
PC->IO: #SET SWITCH6:0x0020
IO->PC: #OK
PC->IO: #SET SWITCH7:0x0040
IO->PC: #OK
PC->IO: #SET SWITCH8:0x0080
IO->PC: #OK
PC->IO: #SET SPECIAL MODE:1
IO->PC: #OK

Via the MODBUS interface you have to set the following registers:
PC->IO: Write value 0x0001 to MODBUS register SWITCH RO1 (4x20001)
PC->IO: Write value 0x0002 to MODBUS register SWITCH RO2 (4x20002)
PC->IO: Write value 0x0004 to MODBUS register SWITCH RO3 (4x20003)
PC->IO: Write value 0x0008 to MODBUS register SWITCH RO4 (4x20004)
PC->IO: Write value 0x0010 to MODBUS register SWITCH RO5 (4x20005)
PC->IO: Write value 0x0020 to MODBUS register SWITCH RO6 (4x20006)
PC->IO: Write value 0x0040 to MODBUS register SWITCH RO7 (4x20007)
PC->IO: Write value 0x0080 to MODBUS register SWITCH RO8 (4x20008)
PC->IO: Write value 0x0001 to MODBUS register ENABLE LOGIC FUNCTIONS (4x21001)

Now you can switch on or off all 8 relay outputs RO1 to RO8 with the first 8 digital inputs DI1 to DI8.

19.2.4 Logic function SWITCH ON

This logic function checks the status of the mapped digital inputs and sets the corresponding relay output to a defined state. In case of the function SWITCH ON to 1, if the module detects a rising edge on one of the mapped digital inputs.

Example: The relay output RO1 is switched on by one of the four digital inputs DI1, DI2, DI3 and DI4

Over the ASCII interface you have to send the following commands:
PC->IO: #SET SWITCH ON1:0x000F
IO->PC: #OK
PC->IO: #SET SPECIAL MODE:1
IO->PC: #OK

Via the MODBUS interface you have to set the following registers:
PC->IO: Write value 0x000F to MODBUS register SWITCH ON RO1 (4x20017)
PC->IO: Write value 0x0001 to MODBUS register ENABLE LOGIC FUNCTIONS (4x21001)

The other relay outputs are not affected by this configuration.

Example: Central light on with digital input DI16

Over the ASCII interface you have to send the following commands:
PC->IO: #SET SWITCH ON1:0x8000
IO->PC: #OK
PC->IO: #SET SWITCH ON2:0x8000
IO->PC: #OK
PC->IO: #SET SWITCH ON3:0x8000
IO->PC: #OK
PC->IO: #SET SWITCH ON4:0x8000
IO->PC: #OK
PC->IO: #SET SWITCH ON5:0x8000
IO->PC: #OK
PC->IO: #SET SWITCH ON6:0x8000
IO->PC: #OK
PC->IO: #SET SWITCH ON7:0x8000
IO->PC: #OK
PC->IO: #SET SWITCH ON8:0x8000
IO->PC: #OK
PC->IO: #SET SPECIAL MODE:1
IO->PC: #OK

Via the MODBUS interface you have to set the following registers:
PC->IO: Write value 0x8000 to MODBUS register SWITCH ON RO1 (4x20017)
PC->IO: Write value 0x8000 to MODBUS register SWITCH ON RO2 (4x20018)
PC->IO: Write value 0x8000 to MODBUS register SWITCH ON RO3 (4x20019)
PC->IO: Write value 0x8000 to MODBUS register SWITCH ON RO4 (4x20020)
PC->IO: Write value 0x8000 to MODBUS register SWITCH ON RO5 (4x20021)
PC->IO: Write value 0x8000 to MODBUS register SWITCH ON RO6 (4x20022)
PC->IO: Write value 0x8000 to MODBUS register SWITCH ON RO7 (4x20023)
PC->IO: Write value 0x8000 to MODBUS register SWITCH ON RO8 (4x20024)

PC->IO: Write value 0x8000 to MODBUS register ENABLE LOGIC FUNCTIONS (4x21001)

If you connect a push button switch to the digital input 16 and press this button, all eight relay outputs are switched on immediately. If you don't press the button, you can switch each of the eight relay on or off via MODBUS or ASCII protocol

19.2.5 Logic function SWITCH OFF

This logic function checks the status of the mapped digital inputs and sets the corresponding relay output to a defined state. In case of the function SWITCH OFF to 0, if the module detects a rising edge on one of the mapped digital inputs.

Example: Switch off relay output RO2 with one of the three digital inputs DI1, DI3, DI6

Over the ASCII interface you have to send the following commands:
Bit 0 stands for DI1 -> 1
Bit 2 stands for DI3 -> 4
Bit 5 stands for DI6 -> 32
Results in 1+4+32 -> 37
PC->IO: #SET SWITCH OFF2:37
IO->PC: #OK
PC->IO: #SET SPECIAL MODE:1
IO->PC: #OK

Via the MODBUS interface you have to set the following registers:
PC->IO: Write value 37 to MODBUS register SWITCH OFF RO2 (4x20026)
PC->IO: Write value 0x0001 to MODBUS register ENABLE LOGIC FUNCTIONS (4x21001)

The other relay outputs are not affected by this configuration.

Example: Central light off with DI15

Over the ASCII interface you have to send the following commands:
PC->IO: #SET SWITCH OFF1:0x4000
IO->PC: #OK
PC->IO: #SET SWITCH OFF2:0x4000
IO->PC: #OK
PC->IO: #SET SWITCH OFF3:0x4000
IO->PC: #OK
PC->IO: #SET SWITCH OFF4:0x4000
IO->PC: #OK
PC->IO: #SET SWITCH OFF5:0x4000
IO->PC: #OK
PC->IO: #SET SWITCH OFF6:0x4000
IO->PC: #OK
PC->IO: #SET SWITCH OFF7:0x4000
IO->PC: #OK
PC->IO: #SET SWITCH OFF8:0x4000
IO->PC: #OK
PC->IO: #SET SPECIAL MODE:1
IO->PC: #OK

Via the MODBUS interface you have to set the following registers:

PC->IO: Write value 0x4000 to MODBUS register SWITCH OFF RO1 (4x20025)

PC->IO: Write value 0x4000 to MODBUS register SWITCH OFF RO2 (4x20026)

PC->IO: Write value 0x4000 to MODBUS register SWITCH OFF RO3 (4x20027)

PC->IO: Write value 0x4000 to MODBUS register SWITCH OFF RO4 (4x20028)

PC->IO: Write value 0x4000 to MODBUS register SWITCH OFF RO5 (4x20029)

PC->IO: Write value 0x4000 to MODBUS register SWITCH OFF RO6 (4x20030)

PC->IO: Write value 0x4000 to MODBUS register SWITCH OFF RO7 (4x20031)

PC->IO: Write value 0x4000 to MODBUS register SWITCH OFF RO8 (4x20032)

PC->IO: Write value 0x0001 to MODBUS register ENABLE LOGIC FUNCTIONS (4x21001)

The other relay outputs are not affected by this configuration. If you connect a pushbutton switch to digital input DI15, all eight relay outputs are switched immediately to 0, if the button is pressed. If the button is released, you can switch on or off each output relay via the MODBUS or ASCII protocol.

19.2.6 Logic function TOGGLE

This logic function checks the status of the mapped digital inputs and sets the corresponding relay output to a defined state. In case of the function TOGGLE, the module inverts the current state of the relay output, if the module detects a rising edge on one of the mapped digital inputs.

Example: Toggle switch: With one of the two digital inputs DI1, DI2 we want to invert the relay output RO4.

Over the ASCII interface you have to send the following commands:

Bit 0 stands for DI1 -> 1

Bit 1 stands for DI2 -> 2

Results in 1+2 -> 3

PC->IO: #SET TOGGLE4:3

IO->PC: #OK

PC->IO: #SET SPECIAL MODE:1

IO->PC: #OK

Via the MODBUS interface you have to set the following registers:

PC->IO: Write value 3 to MODBUS register TOGGLE RO4 (4x20012)

PC->IO: Write value 0x0001 to MODBUS register ENABLE LOGIC FUNCTIONS (4x21001)

The other relay outputs are not affected by this configuration. If you connect two push buttons to the digital inputs DI1 and DI2 and press one of them, the current status of the relay output RO4 is inverted.

19.2.7 Logic function PULSE

This logic function checks the status of the mapped digital inputs and sets the corresponding relay output to a defined state. In case of the function PULSE, the module starts an off delay timer with the time span of PULSE TIME on, if the module detects a rising edge on one of the mapped digital inputs.

Example: Stairway lighting: With one of the two digital inputs DI1, DI2 we want to switch on the output relay RO1 for 30 seconds.

Over the ASCII interface you have to send the following commands:

Bit 0 stands for DI1 -> 1

Bit 1 stands for DI2 -> 2

Results in 1+2 -> 3

PC->IO: #SET PULSE4:3

IO->PC: #OK

The time is defined in 1/10s. So the value 300 defines a time of 30 seconds.

PC->IO: #SET PULSE TIME4:300

IO->PC: #OK

PC->IO: #SET SPECIAL MODE:1

IO->PC: #OK

Via the MODBUS interface you have to set the following registers:

PC->IO: Write value 3 to MODBUS register PULSE RO1 (4x20033)

PC->IO: Write value 300 as a 32 bit value to the two registers PULSE TIME RO1 4x20065-4x20066.

The number 0x12345678 will be divided into two 16 bit values and stored in this way:

4x200065:0x1234 and 4x20066:0x5678

300 as hexadecimal number is 0x0000012C.

PC->IO: Write value 0x0000 to MODBUS register PULSE TIME RO1 (4x20065)

PC->IO: Write value 0x012C to MODBUS register PULSE TIME RO1 (4x20066)

or:

PC->IO: Write value 300 as a 32 bit value to the two registers PULSE TIME RO1 4x20081-4x20082

The number 0x12345678 will be divided into two 16 bit values and stored in this way:

4x200081:0x5678 and4x20066:0x1234

300 as hexadecimal number is 0x0000012C.

PC->IO: Write value 0x012C to MODBUS register PULSE TIME RO1 (4x20081)

PC->IO: Write value 0x0000 to MODBUS register PULSE TIME RO1 (4x20082)

PC->IO: Write value 0x0001 to MODBUS register ENABLE LOGIC FUNCTIONS (4x21001)

The other relay outputs are not affected by this configuration. If you connect two push buttons to the digital inputs DI1 and DI2 and you press one of the two buttons, the relay output RO4 will be on for 30 seconds. After this time span the relay output will be switched off automatically. If you press one of the two buttons again, if the output relay is on, the time span of 30 seconds starts again.

19.3 Technical specification

Beside the basic technical data, which fulfil all of our IO modules, this IO modules meet the following technical specifications:

Power consumption	
RESI-S16DI8PO-SIO	<2.0W
RESI-S8PO-SIO	<2.0W

Product housing	
RESI-S16DI8PO-SIO	BIG IO XT8
RESI-S8PO-SIO	BIG IO XT8

Product weight	
RESI-S16DI8PO-SIO	565g
RESI-S8PO-SIO	555g

Digital inputs	
only RESI-S16DI8PO-SIO	
Total amount of inputs	16
Sampling rate	Every 5ms
Input voltage range	12-48V= +/-10%
Input current	approx. 1mA per channel
Logic levels	0: <3V=
	1: >5V=
Cable connection	Via 18-pin plug-in terminal block
Terminal type	RM3.5
Galvanic insulation	No

Relay outputs	
Number of outputs	8 bistable relays
	for socket-outlets and light applications
Relay type	Bistable with manual operation
Incandescent electric lamp load	Max 4.800 W
Capacitive load	Max. 200µF
Maximum voltage	250Vac
Maximum current	16A
Mechanical lifetime	10^6 cycles of operation
Contact material	$AgSnO_2$
Insulation	Creepage and clearance distance 8mm
Cable connection	Via 8 2-pin plug-in terminal blocks
Terminal type	RM5
Galvanic insulation	Yes, with the relay

Output power per channel:

Incandescent lamp	4.800 W
Fluorescent lamp not compensated	5.000 W
Fluorescent lamp parallel compensated	2.500 W / 200 µF
Fluorescent lamp duo-combination	2 x 5.000 W
Halogen lamp (230VAC)	5.000 W
Low voltage halogen lamp with transformer	2.000 VA
Mercury arc sodium discharge lamp not compensated	5.000 W
Mercury arc sodium discharge lamp parallel compensated	5.000 W / 200 µF
Dulux lamp not compensated	4.000 W
Dulux lamp parallel compensated	3.000 W / 200 µF

Default serial settings

Baud rate	via DIP switch
Parity	none
Stop bit(s)	one
UnitID	255

19.4 Additional terminals & LED states

DIGITAL INPUTS	16 digital inputs for 12-48Vdc signals	
	One 18 pin plug-in terminal block	
	Terminal type: RM3.5	
	C:	Ground of the module
	DI1-DI16:	Digital inputs
		0=open or GND,
		1=+12Vdc..+48Vdc
Pin layout	Pin 1:	C=GND
	Pin 2:	1=DI1
	Pin 3:	2=DI2
	Pin 4:	3=DI3
	Pin 5:	4=DI4
	Pin 6:	5=DI5
	Pin 7:	6=DI6
	Pin 8:	7=DI7
	Pin 9:	8=DI8
	Pin 10:	9=DI9
	Pin 11:	10=DI10
	Pin 12:	11=DI11
	Pin 13:	12=DI12
	Pin 14:	13=DI13
	Pin 15:	14=DI14
	Pin 16:	15=DI15
	Pin 17:	16=DI16
	Pin 18:	C=GND

RELAY OUTPUTS	8 bistable relays for max 250Vac signals	
	Eight 2 pin plug-in terminal blocks for Form A relay	
	Terminal type:	RM5
	1:	Switching contact of the relay +
	2:	Switching contact of the relay -
Pin layout	Pin 1:	1=Switching contact of the relay +
	Pin 2:	2=Switching contact of the relay -

19.5 Connection diagram

19.5.1 Cabling of the digital inputs

Only for RESI-S16DI8PO-SIO: In the below drawing you see the cabling of the 16 digital inputs of the module. Both terminals C are internally connected to the ground signal.

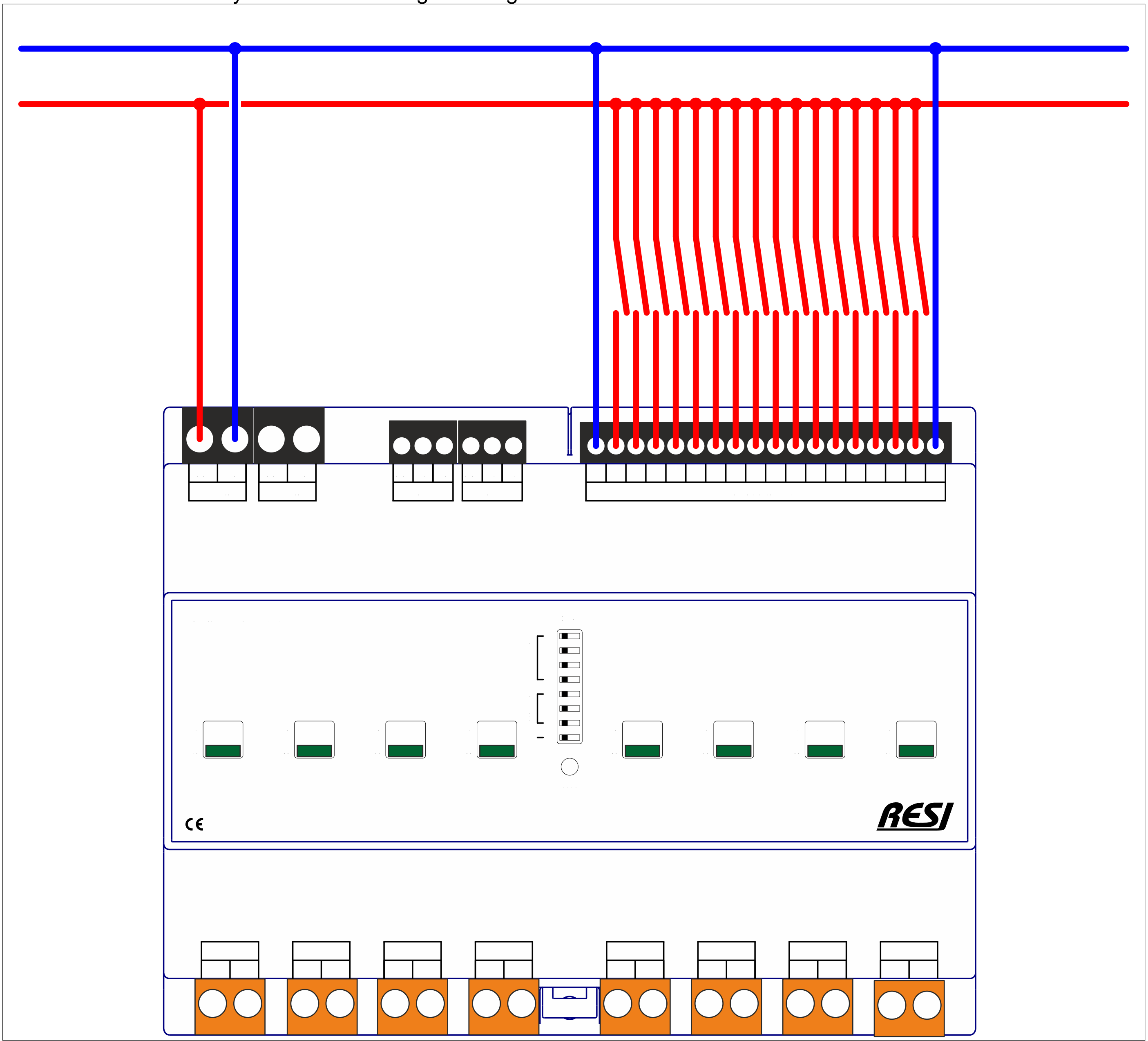


Figure: Connecting the digital inputs to the IO module

19.5.2 Cabling of the bistable relay outputs

In the below drawing the cabling of the bistable relay outputs is shown.

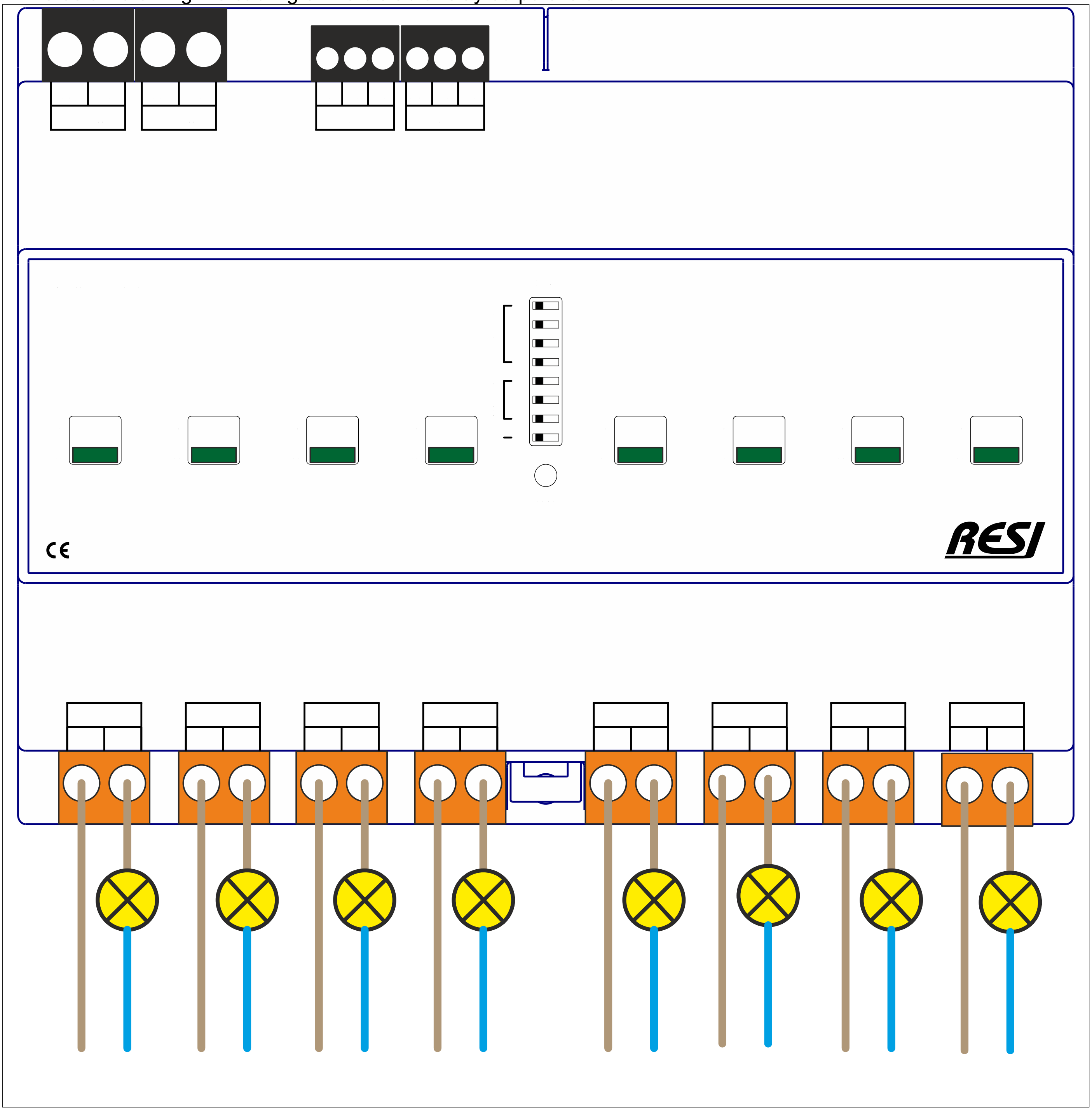


Figure: Connecting the bistable relay outputs to the IO module

19.6 Additional MODBUS register & coils

Here you will find only the additional MODBUS registers and coils especially for this IO module. Please refer to the description of of the standard MODBUS mapping for more details about the available basic MODBUS registers and coils.

Please refer to the external document for detailed documentation of the current MODBUS register mapping for this IO module:

RESI-L-S16DI8PO,S8PO-SIO-MODBUS+ASCII-ENxx.pdf

19.7 Additional ASCII commands

Here you will find only the additional ASCII commands especially for this IO module. Please refer to the description of of the standard commands for more details about the available basic ASCII commands.

Please refer to the external document for detailed documentation of the current ASCII commands for this IO module:

RESI-L-S16DI8PO,S8PO-SIO-MODBUS+ASCII-ENxx.pdf

20 RESI-20RI8SB-SIO, RESI-8SB-SIO, RESI-10RI4SB-SIO, RESI-4SB-SIO

20.1 General information

This series of IO modules offer the following features:

- Special module for controlling shades and blinds with relays and time control
- IO Module does all the time critical control for the shades/blinds internally
- RESI-8SB-SIO: module to control up to 8 shades/blinds with 16 relays
- RESI-4SB-SIO: module to control up to 4 shades/blinds with 8 relays
- RESI-20RI8SB-SIO: Additional 20 digital inputs for 12-250Vac/dc signals
- RESI-10RI4SB-SIO: Additional 10 digital inputs for 12-250Vac/dc signals
- Internal FRAM memory to save all setup information for each shade/blind
- Galvanic insulated RS485 interface for communication with a host system

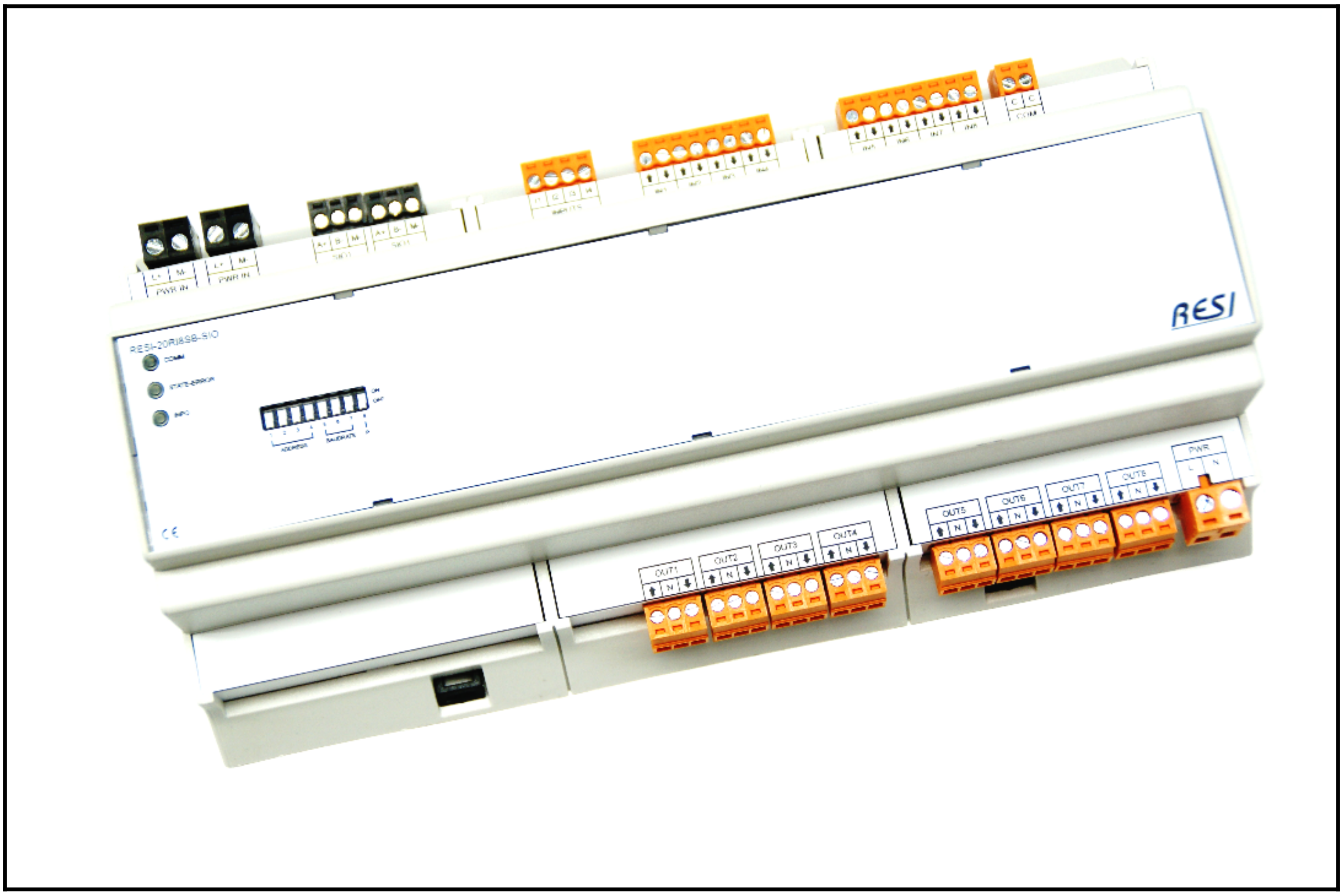


Figure: Our RESI-20RI8SB-SIO module

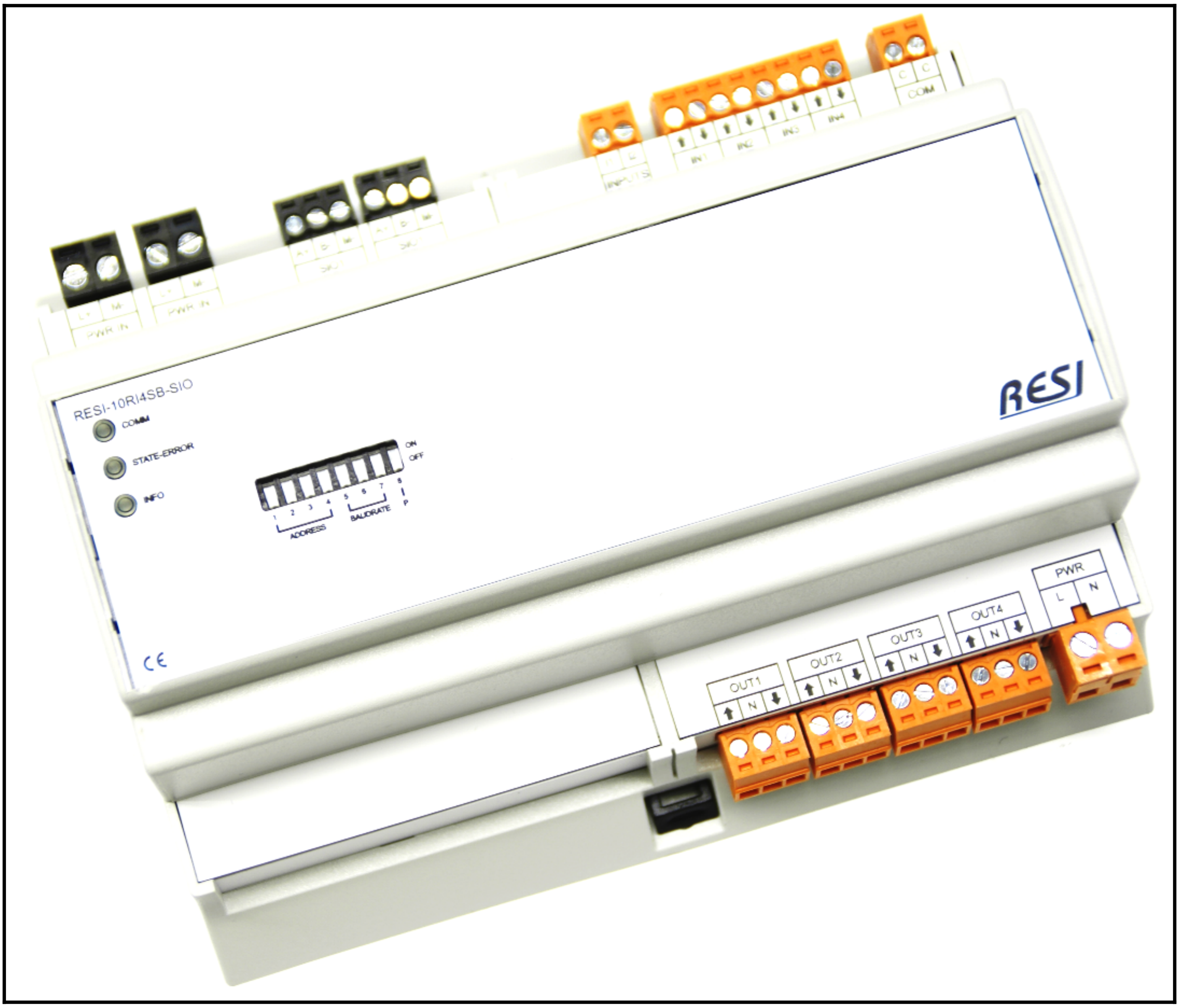


Figure: Our RESI-10RI4SB-SIO module

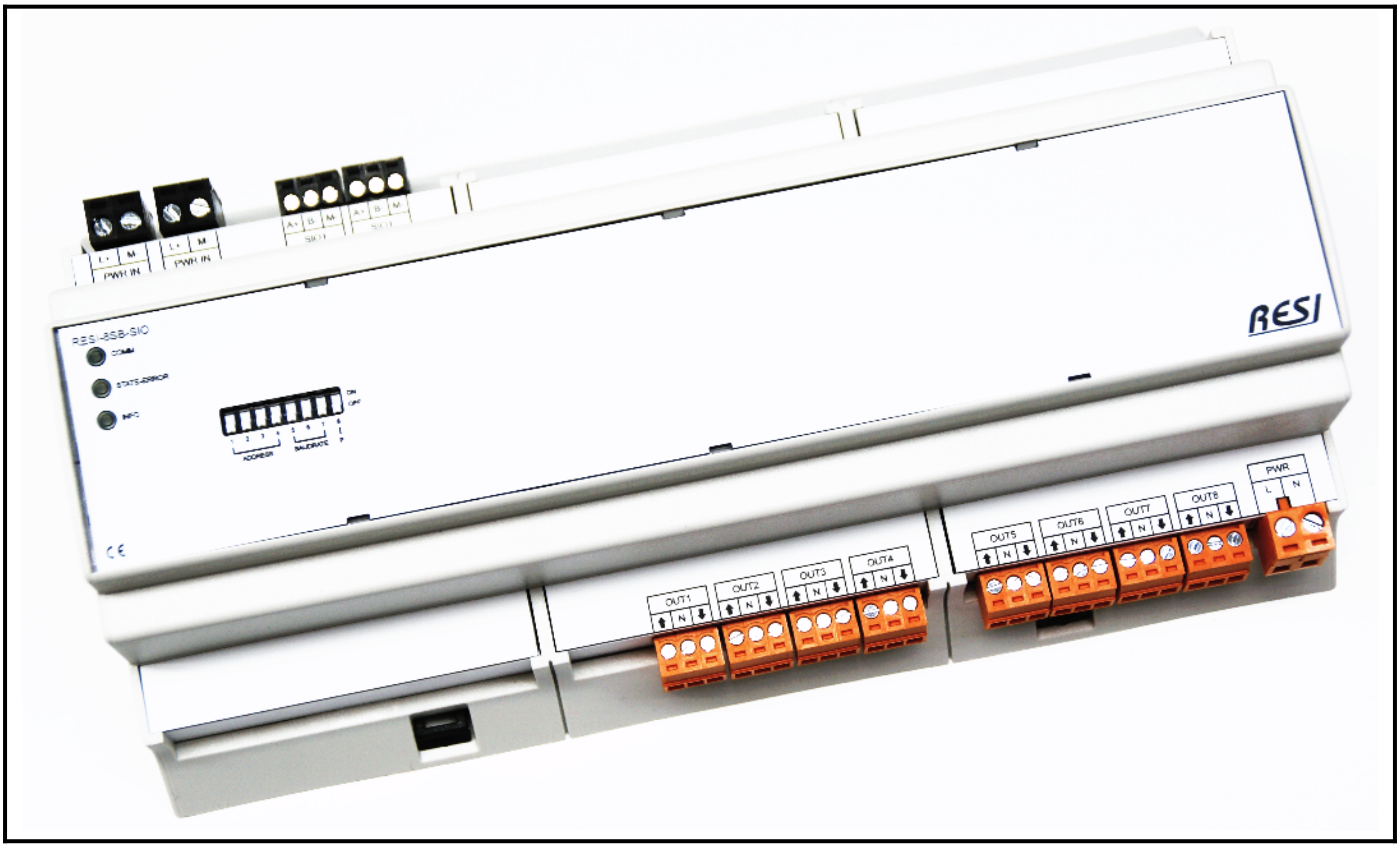


Figure: Our RESI-8SB-SIO module

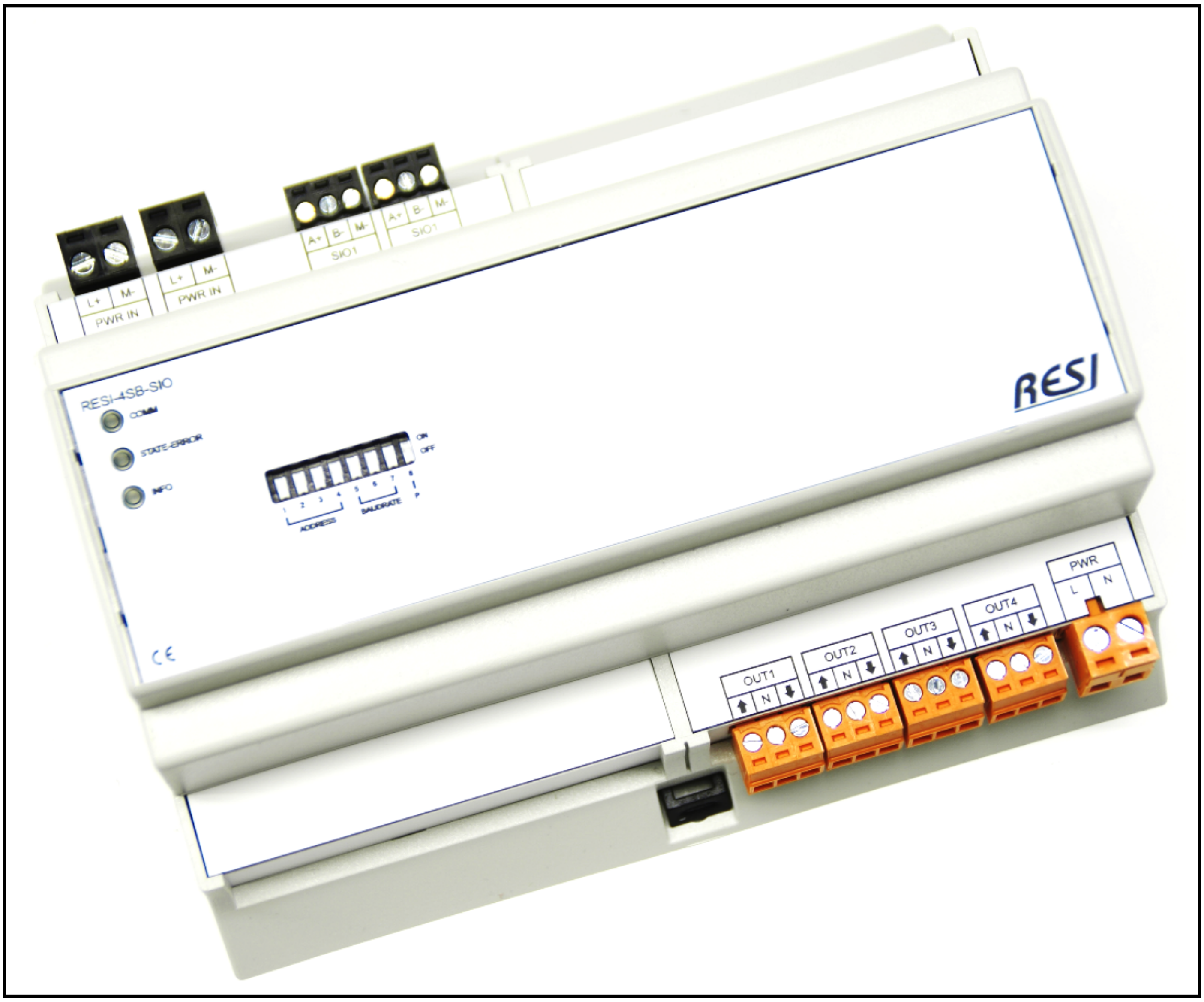


Figure: Our RESI-4SB-SIO module

20.2 Basic function

These IO modules are designed to control shades, blinds, sun blinds, awnings or roller shutter engines with AC or DC power supply with 3 connections.

Our IO module RESI-20RI8SB-SIO and RESI-8SB-SIO offers the control of up to eight individual engines. The IO module version RESI-10RI4SB-SIO and RESI-4SB-SIO offers the control of up to four individual engines.

In addition the IO modules RESI-20RI8SB-SIO and RESI-10RI4SB-SIO offer digital inputs for direct control of the engines with push buttons either with 24Vdc or 110/230Vac signals. Also the digital inputs can be used for wind/rain alarms sensor with relay output or for group control of more than one engine.

Of course all unused digital inputs and relay outputs can be used for other purposes in your application. e.g. for collecting error signals or for controlling other devices with a potential free contact.

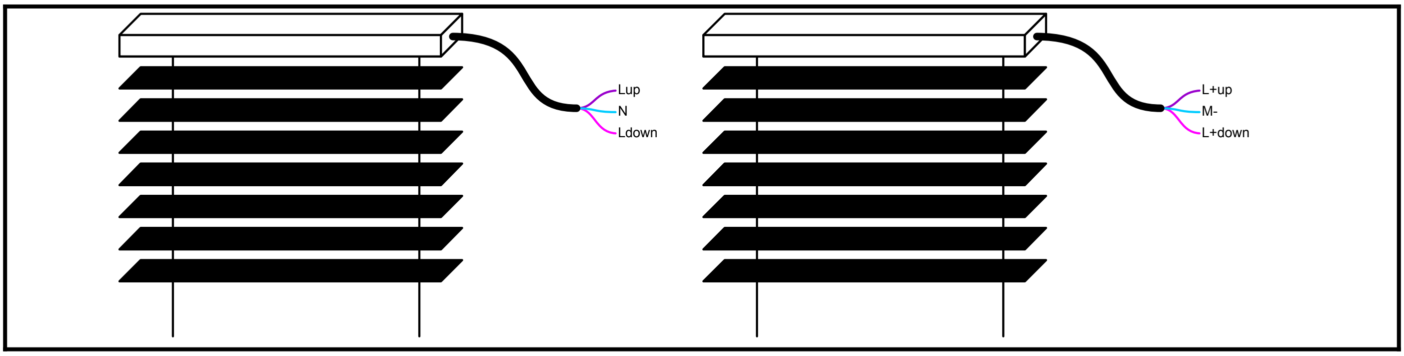


Figure: Control electrical shades with 3 wire connection

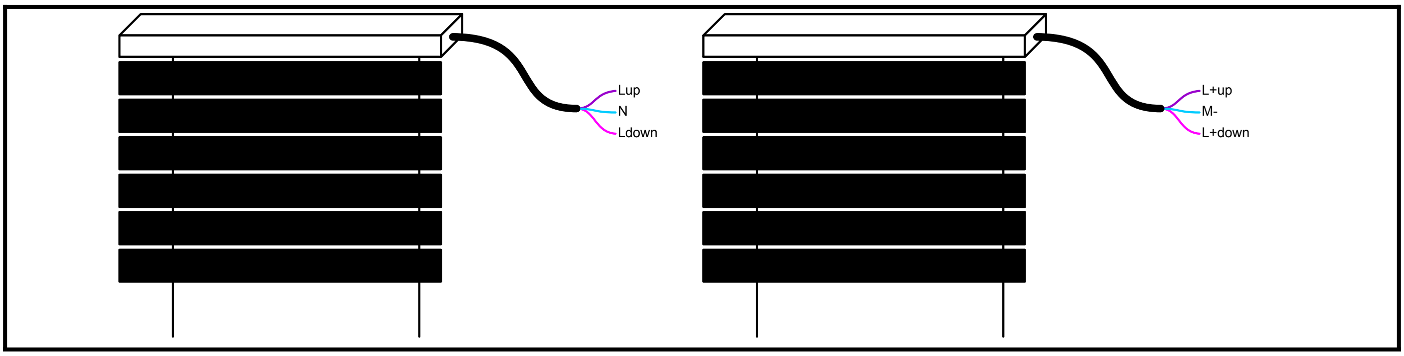


Figure: Control electrical roller shutter with 3 wire connection

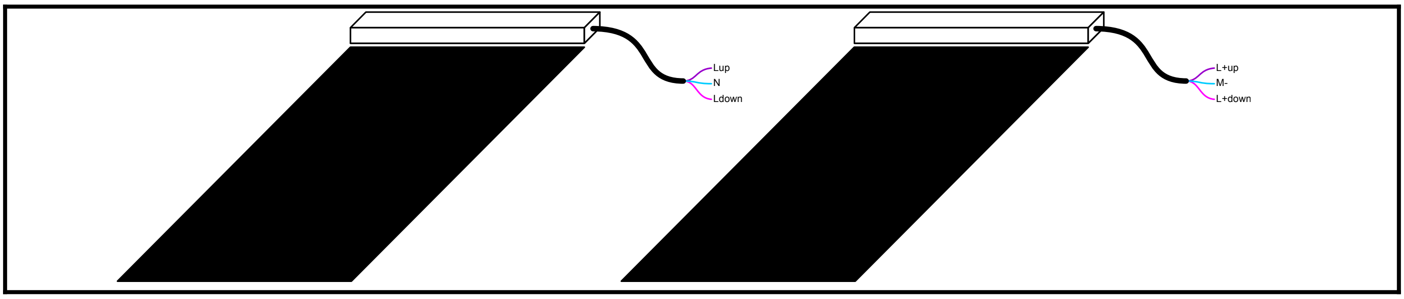


Figure: Control electrical sun blinds or awnings with 3 wire connection

20.2.1 IMPORTANT HINT: PARALLEL CABLING NOT ALLOWED

All engines must have their own outlet. It is not allowed to connect more than one engine to one outlet. This can damage your engine immediately due to erroneous currents in the parallel cabled engines!

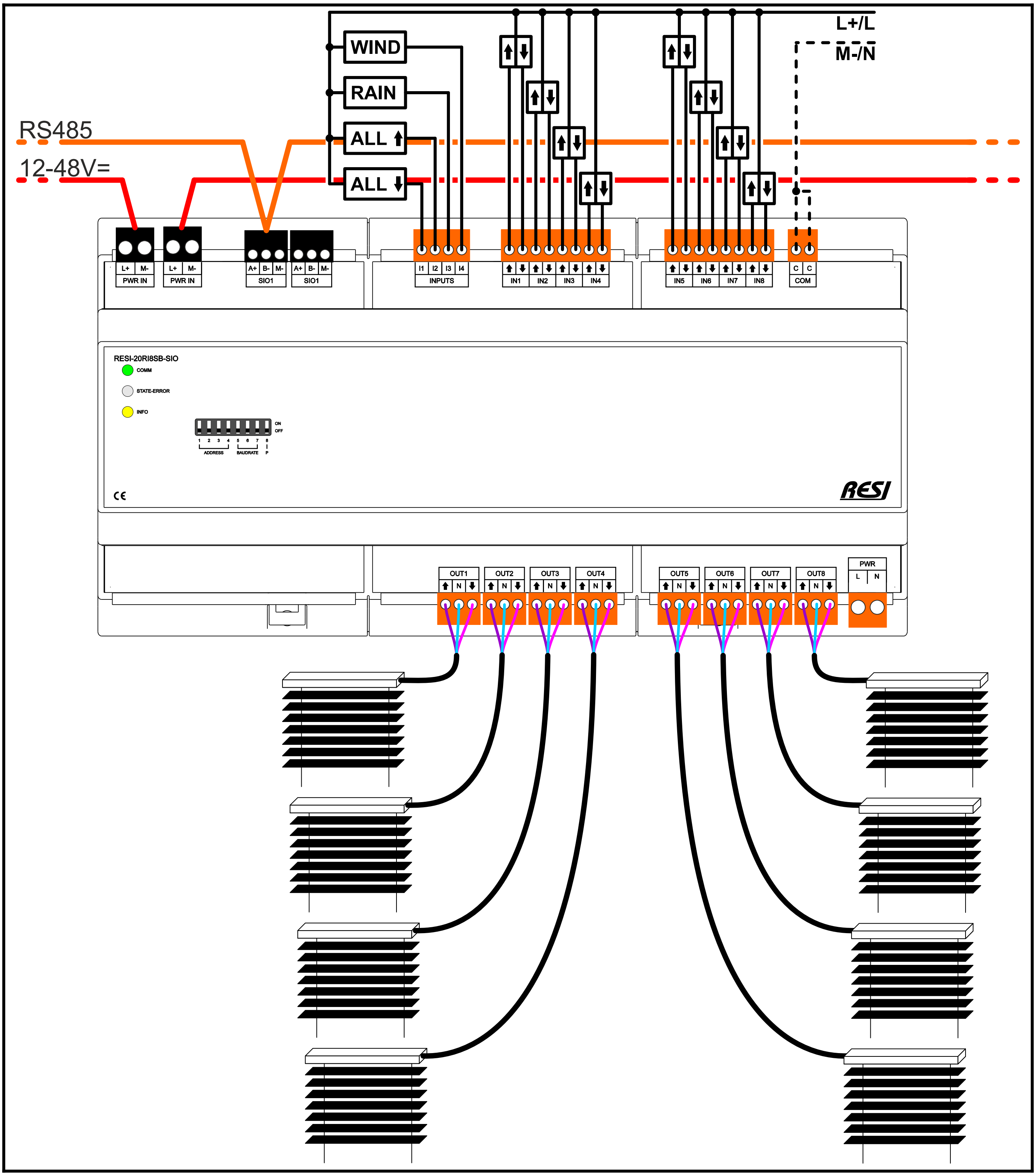


Figure: Correct wiring of all engines

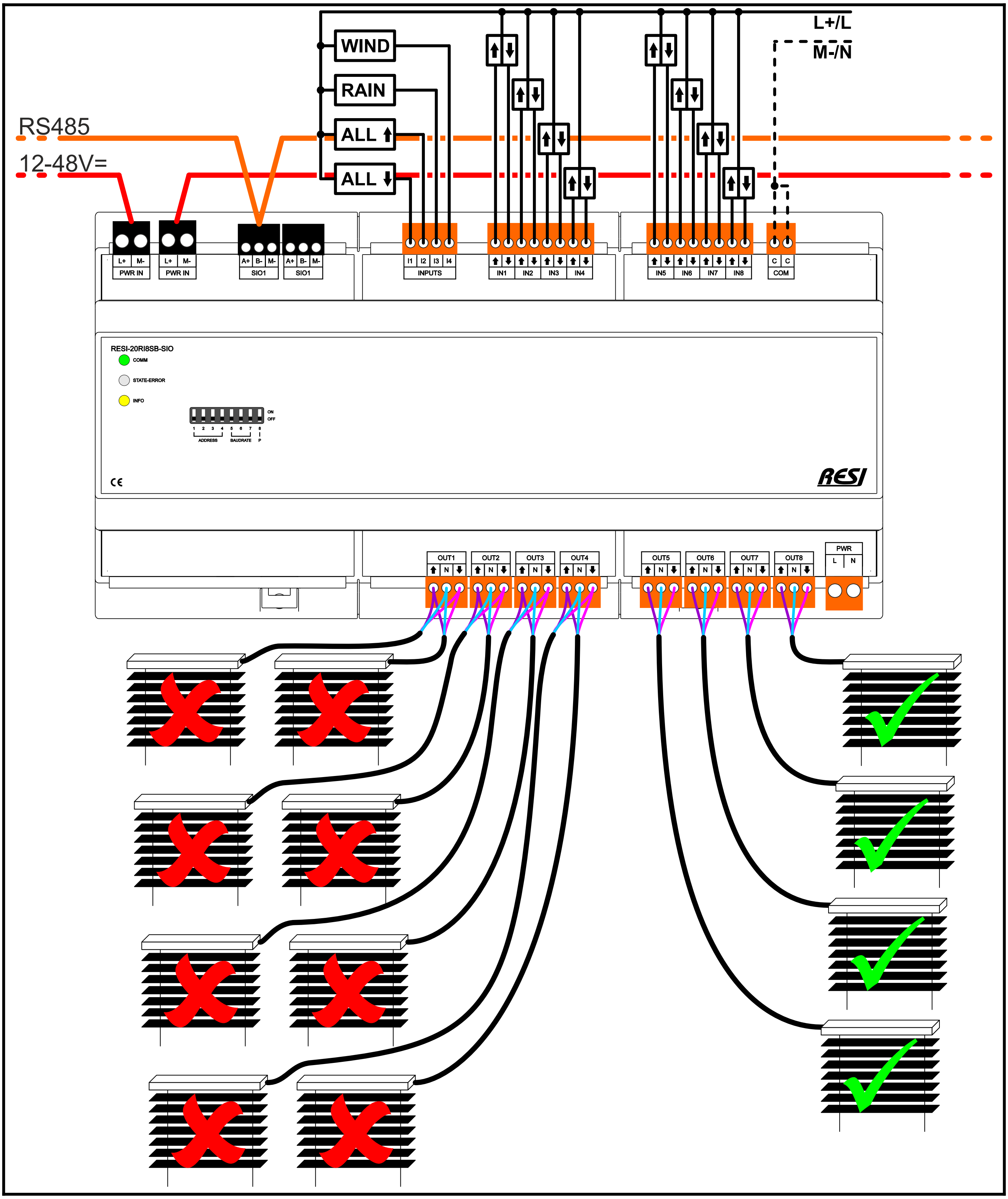


Figure: Incorrect wiring of all engines – NOT ALLOWED!!!!

20.2.2 Basic configuration of shade/blind outlets

Each shade/blind outlet has its own configuration parameter block, which defines a lot of parameters used to control the individual outlet. Please consult the paper **RESI-L-20RI8SB,8SB,10RI4SB,4SB-SIO-MODBUS+ASCII-ENxx.pdf** for more detailed specification of all parameters. We will mention here only the most important parameters for a better understanding of the operation.

The first configuration block for OUT1 is located between 4x01001,3x01001,I:1000 and 4x01200,3x01200,I:1999.

The second configuration block for OUT2 is located between 4x01201,3x01201,I:1200 and 4x01400,3x01400,I:1399.

The third configuration block for OUT3 is located between 4x01401,3x01401,I:1400 and 4x01600,3x01600,I:1599.

The 4th configuration block for OUT4 is located between 4x01601,3x01601,I:1600 and 4x01800,3x01800,I:1799.

The 5th configuration block for OUT5 is located between 4x01801,3x01801,I:1800 and 4x02000,3x02000,I:1999.

The 6th configuration block for OUT6 is located between 4x02001,3x02001,I:2000 and 4x02200,3x02200,I:2199.

The 7th configuration block for OUT7 is located between 4x02201,3x02201,I:2200 and 4x02400,3x02400,I:2399.

The 8th configuration block for OUT8 is located between 4x02401,3x02401,I:2400 and 4x02600,3x02600,I:2599.

Of course, RESI-20RI8SB_SIO, RESI-8SB-SIO uses all eight configuration block, but RESI-10RI4SB-SIO, RESI-4SB-SIO uses only the first four configuration blocks. OUT5 to OUT8 configuration block is unused in this IO modules.

All configuration parameters are stored internally intot a permanent ferro-magentic memory with almost unlimited write cycles! After you have set all parameters you have to restart the IO module (power of-on cylce or software reset) to activate all new parameters!

We use OUT1 as a sample here to explain the functionality of the configuration registers.

BILIND & SHUTTER GROUP: Outputs DO1+DO2: CONFIGURATION						
MODE	3x01001 4x01001 I:1000	3,0x0003 B:00 03		N/A:NO CHANGE	UINT16 R/W	NO
		BLIND		SELECT FROM LIST		
Current mode of the first blinds / shutter group: = 0: NONE: Both digital outputs are always off = 1: TWO OUTPUTS: Both digital outputs can be used as normal outputs = 2: SHUTTER: Both digital outputs form a shutter WITHOUT slat adjustment = 3: BLIND: Both digital outputs form a blind with slat adjustment						

This is the most important register to configure the correct function of the outlet. Each outlet consists of two relays, which are controlled as a group. The mode SHUTTER is designed for sun blinds or roller shutter without the ability to control the slat position. The mode BLIND allows the positioning of the slats also. The mode TWO OUTPUTS allow the use of the two outputs as general outputs for other purposes.

REVERT	3x01002 4x01002 I:1001	0,0x0000 B:00 00		N/A:NO CHANGE	UINT16 R/W	NO
		NORMAL OUTPUTS [DO1=UP;DO2=DOWN]		SELECT FROM LIST		
Defines whether the direction of the shutter or blind should be reversed: = 0: NORMAL: digital output # 1 moves up, # 2 down = 1: INVERTED: Digital output # 1 moves down, # 2 up						

This is very useful if you have a false cabling of the UP/DOWN direction of your engine. Normally relay output #1 should move the shade/blind upwards and relay output #2 shout move the shade/blind downwards. Activating this configuration will invert the behavior of the two relay outputs.

TIME UP	3x01003 4x01003 I:1002	42,0x002A B:00 2A			UINT16 R/W	NO
		42s		VALUE IN XX SECONDS		
Movement time of the shutter / blind upwards in seconds. 1..65535 seconds						
TIME EXTEND UP	3x01004 4x01004 I:1003	500,0x01F4 B:01 F4	0		UINT16 R/W	NO
		05,00%		VALUE IN XX,XX%		
Extension of the upward movement time in % in order to reach the end position correctly. 0..2500 → 0..25%						
TIME DOWN	3x01005 4x01005 I:1004	42,0x002A B:00 2A		42	UINT16 R/W	NO
		42s		VALUE IN XX SECONDS		
Movement time of the shutter / blind downwards in seconds. 1..65535 seconds						
TIME EXTEND DOWN	3x01006 4x01006 I:1005	500,0x01F4 B:01 F4	0		UINT16 R/W	NO
		05,00%		VALUE IN XX,XX%		
Extension of the downward movement time in % in order to reach the end position correctly. 0..2500 → 0..25%						

These parameters define the timing for a complete movement of the shade/blind in 1/10th seconds from position 100% (fully closed) to %0 (Fully opened) (TIME UP) and vice versa (TIME DOWN). The two percentage registers TIME EXTEND UP and TIME EXTEND DOWN are ony used in positioning commands reaching the two end values 0% and 100%. Then the engine run time will be extended by the defined percentage to securely reach the final position.

An example: You have configured 100 Seconds for TIME UP and TIME DOWN and 10% for TIME EXTEND UP and TIME EXTEND DOWN. After referencing the engine to position 0% you send a move to 50% command. This will mean that the engine will be on for 50.0s. After reaching this position you send a positioning command move to 100%. Then the actor will move for 50s plus 10% from 100s → 10s. So in total the engine is on for 60s. After that the final position 100% is reached. Now you send a move to 0% command. Again the engine will be on for 100s and 10% of 100% → in total 110s to reach always the upper position. After that movement the position 0% is reached.

This is an implicit reference move for the two final positions 0% and 100% to correct most of the time based errors over the time.

PAUSE UP DOWN	3x01007 4x01007 I:1006	500,0x01F4 B:01 F4			UINT16 R/W	NO
		500ms		PAUSE IN XXms		
Pause between moving up/down the shutters/blinds in milliseconds 0..30000 ms						

This parameter is used in every direction change between upwards and downwards movement to stop the previous movement before the next movement is started. A good value for most applications is 500ms to 1s. It is also used to give the engine time to change the flow direction of the current in the motor winding.

MOTOR DELAY ON	3x01008 4x01008 I:1007	200,0x00C8 B:00 C8			UINT16 R/W	NO
		200ms		MOTOR DELAY ON IN XXms		
Motor on-delay time in milliseconds until the motor reaches full force. 0..10000ms						
MOTOR DELAY OFF	3x01009 4x01009 I:1008	200,0x00C8 B:00 C8			UINT16 R/W	NO
		200ms		MOTOR DELAY OFF IN XXms		
Delay time when switching off the motor in milliseconds until the motor has no more power. 0..10000ms						

MOTOR DELAY ON is used to delay the correction of the position after starting a new movement. Therefore this time is added to the calculated movement time from the actual position to the next position. After the engine is on for this amount of time, the current position is updated. This parameter is used to cover the issue, that after you switch the relay output to 1, the relay itself and the engine need time to build up a force to move the shade/blind.

MOTOR DELAY OFF is the same after the end of a movement. When you switch off the digital output, the relay and the engine aren't switched off immediately. The have a litte delay while the shade/blind will move. So this additional time is used to correct the final position of the shade/blind.

For both parameters values between 50ms and 300ms are useful.

SLAT ANGLE UP	3x01015 4x01015 I:1014	90,0x005A B:00 5A			UINT16 R/W	NO
		90°		SLAT ANGLE UP IN XX°		
Position of the slat when moving up in degrees. For raffstores 90 ° (horizontal) for other blinds 0 ° (vertical upward) 0..180 → 0 ° ..180 ° 0 ° vertically upwards 90 ° horizontal 180 ° vertically downwards						
SLAT ANGLE HORIZONTAL	3x01016 4x01016 I:1015	90,0x005A B:00 5A			UINT16 R/W	NO
		90°		SLAT ANGLE HORIZONTAL IN XX°		
Position of the slat for horizontal position in degrees. Normally 90 °. 0..180 → 0 ° ..180 ° 0 ° vertically upwards 90 ° horizontal 180 ° vertically downwards						
SLAT ANGLE DOWN	3x01017 4x01017 I:1016	180,0x00B4 B:00 B4			UINT16 R/W	NO
		180°		SLAT ANGLE DOWN IN XX°		
Position of the slat when moving down in degrees. For raffstores 180 ° (vertically downwards) for other blinds 180 ° (vertically downward) 0..180 → 0 ° ..180 ° 0 ° vertically upwards 90 ° horizontal 180 ° vertically downwards						

To control the slat angle correctly you have to define the end angles of the slats and the angle for horizontal position of the slat. See below picture for the correct naming of the parameters in relation to the slat position.

For example a standard Raffstore (external venetian blind) is fully opened, when the slat is in horizontal position. So you have to set the top position SLAT ANLGE UP to 90° and also the horzontal position is the same, so SLAT ANGLE HORIZONTAL is also set to 90°. The fully closed Raffstore will have the slats moved to vertical down. So the parameter SLAT ANGLE DOWN is set to 180°.

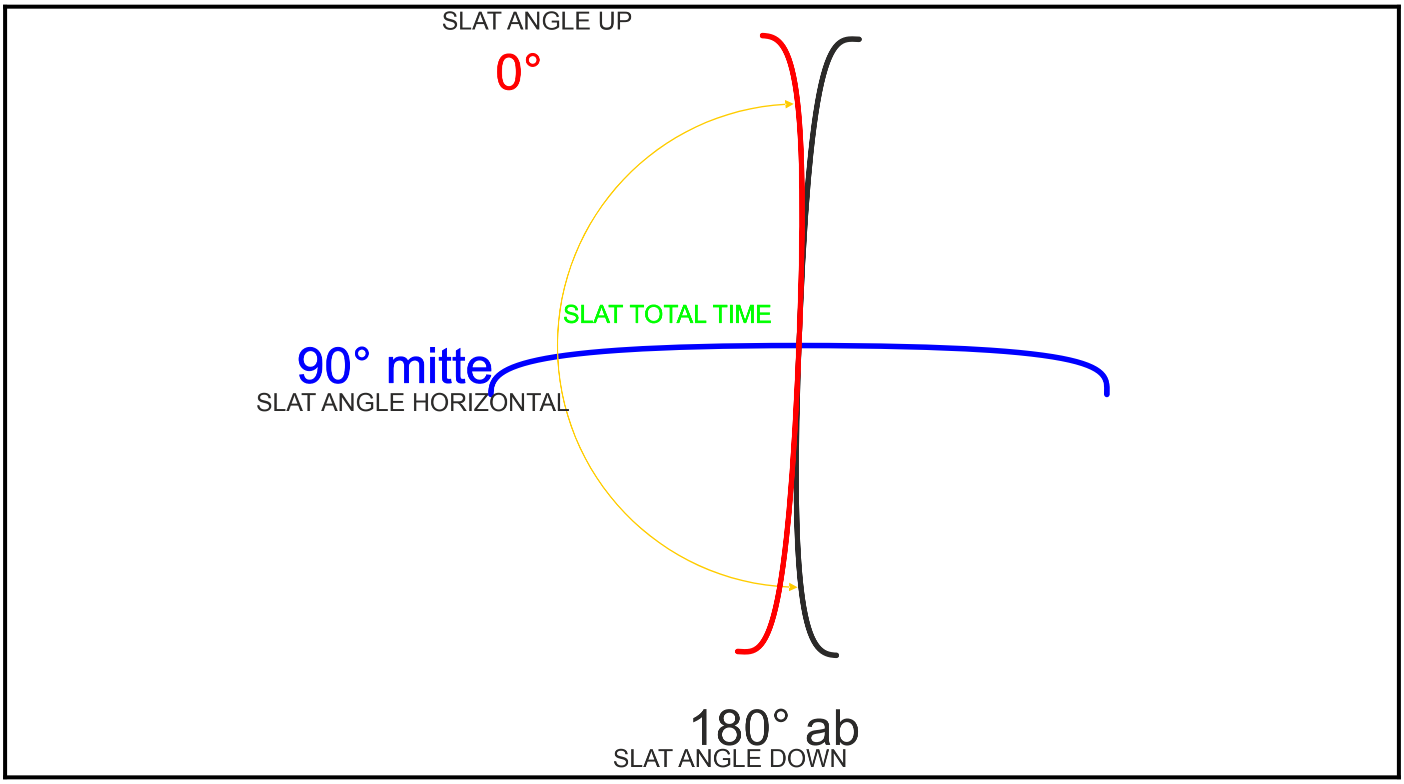


Figure: Definition of the SLAT ANGLE parameters

SLAT TOTAL TIME	3x01012 4x01012 I:1011	1100,0x044C B:04 4C			UINT16 R/W	NO
		1100ms		SLAT TOTAL TIME IN XXms		
Total adjustment time of the slat from position 0% (=SLAT ANGLE UP) to position 100% (=SLAT ANGLE DOWN) in milliseconds 100..65535ms						
SLAT STEP TIME	3x01013 4x01013 I:1012	200,0x00C8 B:00 C8			UINT16 R/W	NO
		200ms		SLAT STEP TIME IN XXms		
Time for an adjustment step for slat in milliseconds 10..65535ms Actual number of slat positions is calculated with: Number of steps = SLAT STEP TIME / SLAT TOTAL TIME Percent per slat adjustment step are calculated with: Percent = 100% / number of steps						
SLAT PAUSE TIME	3x01014 4x01014 I:1013	2500,0x09C4 B:09 C4			UINT16 R/W	NO
		2500ms		SLAT PAUSE TIME IN XXms		
Pause time between two adjustment steps of the slat in milliseconds 0..30000ms						

Only valid for mode SHADE: SLAT TOTAL TIME describes the time in Milliseconds the engine needs to move from position SLAT ANGLE UP to position SLAT ANGLE DOWN. This time describes 100% of the SLAT movement time. All slat position commands are then between these two slat positions: A position 0% means the slat angle SLAT ANGLE UP, a position 50% means the slat angle (SLAT ANGLE DOWN-SLAT ANGLE UP)/2+SLAT ANGLE UP, a position 100% means the slat angle SLAT ANGLE DOWN.

Due to the physical behaviour of a SHADE the controller knows now, that if the down movement is longer than the configured SLAT TOTAL TIME, the new slat angle is SLAT ANGLE DOWN and the slat position will be 100% (fully closed). Also if the up movement is longer than SLAT TOTAL TIME the new slat angle is SLAT ANGLE UP and the new slat position will be 0%. Is the movement shorter than SLAT TOTAL TIME, the slat position will be corrected before the vertical position is updated.

The parameter SLAT STEP TIME is sued to calculate the number of maximum steps you can use while STEPPING mode for the slats. e.g. The SLAT TOTAL TIME is 2000ms and the SLAT STEP TIME is 500ms. So you can do 2000ms/500ms=4 steps from slat position 0% to slat position 100% and vice versa.

SLAT PAUSE TIME defines a pause time in Milliseconds between to consecutive steps to give the user a chance to release the button while slats are stepping. Usually you should use 500ms to 5000ms here. So a total stepping sequence with the SLAT parameters SLAT TOTAL TIME=2000ms, SLAT STEP TIME=500ms, SLAT PAUSE TIME=2000ms, slat start position is 0%, will be:

Start of slat stepping

A. Move from 0% to 25%

B. Wait for 2000ms

C. Move from 25% to 50%

D. Wait for 2000ms

E. Move from 50% to 75%

F. Wait for 2000ms

G. Move from 75% to 100%

H. FINISHED

I: User releases the push button

In the mode BLIND no slat positioning is done by the IO module.

The slat timinig is really critical to the time based calulation of the slat position and the position of the blinds. Therefore we offer four additional parameters to adjust the problem of the different timing for a upward and downward movement of a slat:

SLAT DEAD TIME UP	3x01018 4x01018 I:1017	100,0x0064 B:00 64			UINT16 R/W	NO
		100ms		SLAT DEAD TIME UP IN XXms		
Delay time of the slats, before the slat is really adjusted, if an upward movement has taken place, which led to the complete opening of the slats (0 ° or 90 °). If the used blind in the horizontal upper position has a dead time between the release of the main tape until the first movement downwards, then this parameter compensates this delay. Setting in milliseconds. 0..10000ms						
SLAT DEAD TIME DOWN	3x01019 4x01019 I:1018	10,0x000A B:00 0A			UINT16 R/W	NO
		10ms		SLAT DEAD TIME DOWN IN XXms		
Delay time of the slats, before the slat is really adjusted, if an upward movement has taken place, which led to the complete opening of the slats (0 ° or 90 °). If the used blind in the horizontal upper position has a dead time between the release of the main tape until the first movement downwards, then this parameter compensates this delay. Setting in milliseconds. 0..10000ms						
SLAT DELAY UP	3x01020 4x01020 I:1019	500,0x01F4 B:01 F4			UINT16 R/W	NO
		500ms		SLAT DELAY UP IN XXms		
Some types of blinds require an additional start-up delay when the slat is opened, due to the tensioning and loosening of the tapes, until the first reaction of the slat. This depends on the current slat position. This start-up delay until the slat is turned is always taken into account when the blind is opened, when the slats are in the closed position (100%) and the previous blind movement was a downward movement. Setting in milliseconds. 0..10000ms						
SLAT DELAY DOWN	3x01021 4x01021 I:1020	200,0x00C8 B:00 C8			UINT16 R/W	NO
		200ms		SLAT DELAY DOWN IN XXms		
Some types of blinds require an additional start-up allowance when the slat is closed, due to the tensioning and loosening of the tapes, until the first reaction of the slat. This depends on the current slat position. This start-up delay until the slat is turned is always taken into account when the blind is closed, when the slats are in the open position (0%) and the previous blind movement was an upward movement. Setting in milliseconds. 0..10000ms						

SLAT DEADTIME UP and SLAT DEADTIME DOWN activate the engine without adjusting the slat or blind position, if the final slat posotion will be one of the two end positions. Here values between 50ms and 200ms are useful, depending on the construction of the blind.

SLAT DELAY UP and SLAT DELAY DOWN are additional times for every slat movement before the slat and blind position is updated. Here 100ms to 700ms are usefull values.

20.2.3 Digital input configuration of shade/blind outlets

Each engine outlet OUTx can be controlled directly with a few MODBUS registers described below. But in addition, the RESI-20RI8SB-SIO and RESI-10RI4SB-SIO modules offer digital inputs for direct control of shades and blinds with two push buttons. Also you can configure, that the push buttons control more than one engine outlet (virtual group of shades/blinds).

For each engine outlet you can configure 10 (for RESI-20RI8SB-SIO) or 5 (for RESI-10RI4SB-SIO) digital inputs groups named DIGITAL INPUT GROUP1 to DIGITAL INPUT GROUP10.

DIGITAL INPUT GROUP1 will use digital input 1+2.

DIGITAL INPUT GROUP2 will use digital input 3+4.

...

DIGITAL INPUT GROUP8 will use digital input 15+16.

DIGITAL INPUT GROUP9 will use digital input 17+18.

DIGITAL INPUT GROUP10 will use digital input 19+20.

Each digital input group has the following modes:

=0: mode NOTHING:

The two corresponding digital inputs are NOT used to trigger any movement for the engine outlet. So the digital inputs can be used by the application for other purposes e.g. error inputs,…

=1: Mode UPDOWN 1:

If the outlet is in mode 3 (BLINDS) the digital inputs trigger the following actions:

Short key press detected for push button connected to digital input #1: Blind starts to move upwards to position 0%. Blind will either stop at position 0% or if you do a short key press on one of the two push buttons connected to digital input 1 or 2.

Short key press detected for push button connected to digital input #2: Blind starts to move downwards to position 100%. Blind will either stop at position 100% or if you do a short key press on one of the two push buttons connected to digital input 1 or 2.

Long key press started on push button connected to digital input #1: Slat starts to step to slat position 0%. Slat will either stop at slat position 0% or if you release the push button connected to digital input 1.

Long key press started on push button connected to digital input #2: Slat starts to step to slat position 100%. Slat will either stop at slat position 100% or if you release the push button connected to digital input 2.

If the outlet is in mode 2 (SHADES) the digital inputs trigger the following actions:

Short key press detected for push button connected to digital input #1: Blind starts to move upwards to position 0%. Blind will either stop at position 0% or if you do a short key press on one of the two push buttons connected to digital input 1 or 2.

Short key press detected for push button connected to digital input #2: Blind starts to move downwards to position 100%. Blind will either stop at position 100% or if you do a short key press on one of the two push buttons connected to digital input 1 or 2.

Long key press started on push button connected to digital input #1: Nothing happens.

Long key press started on push button connected to digital input #2: Nothing happens.

=2: Mode UPDOWN 2:

If the outlet is in mode 3 (BLINDS) the digital inputs trigger the following actions:

Short key press detected for push button connected to digital input #1: Blind starts to move upwards to position 0%. Blind will either stop at position 0% or if you do a short key press on one of the two push buttons connected to digital input 1 or 2.

Short key press detected for push button connected to digital input #2: Blind starts to move downwards to position 100%. Blind will either stop at position 100% or if you do a short key press on one of the two push buttons connected to digital input 1 or 2.

Long key press started on push button connected to digital input #1: Slat starts to step to slat position 0%. Slat will either stop at slat position 0% or if you release the push button connected to digital input 1.

Long key press started on push button connected to digital input #2: Slat starts to step to slat position 100%. Slat will either stop at slat position 100% or if you release the push button connected to digital input 2.

If the outlet is in mode 2 (SHADES) the digital inputs trigger the following actions:

Short key press detected for push button connected to digital input #1: Blind starts to move upwards to position 0%. Blind will either stop at position 0% or if you do a short key press on one of the two push buttons connected to digital input 1 or 2.

Short key press detected for push button connected to digital input #2: Blind starts to move downwards to position 100%. Blind will either stop at position 100% or if you do a short key press on one of the two push buttons connected to digital input 1 or 2.

Long key press started on push button connected to digital input #1: The shade steps its position upwards to 0%. The stepping stops if the shade reaches the final position 0% or if the use releases the push button connected to digital input 1.

Long key press started on push button connected to digital input #2: The shade steps its position downwards to 100%. The stepping stops if the shade reaches the final position 100% or if the use releases the push button connected to digital input 2.

=3: Mode UPDOWN 3:

If the outlet is in mode 2 (SHADES) or 3 (BLINDS) the digital inputs trigger the following actions:

Short key press detected for push button connected to digital input #1: Blind or shade starts to move upwards to position 0%. Blind will either stop at position 0% or if you do a short key press on one of the two push buttons connected to digital input 1 or 2.

Short key press detected for push button connected to digital input #2: Blind or shade starts to move downwards to position 100%. Blind will either stop at position 100% or if you do a short key press on one of the two push buttons connected to digital input 1 or 2.

Long key press started on push button connected to digital input #1: Blind or shade starts to move to position 0%. Blind or shade stops after reaching final position 0% or if you release the push button connected to digital input 1.

Long key press started on push button connected to digital input #2: Blind or shade starts to move to position 100%. Blind or shade stops after reaching final position 100% or if you release the push button connected to digital input 2.

In this mode, no slat positioning ins available over the two digital inputs.

=3: Mode WIND+RAIN ALARM:

In this mode the first digital input acts as a wind alarm input and the second digital input acts as a rain alarm input.

If the module detects a rising edge on the digital input 1 the wind alarm is triggered. The engine outlet activates the wind alarm program and moves the shade/blind in a defined position.

If the module is in wind alarm and detects a falling edge on the first input, the wind alarm is deactivated and the module moves the shades/blinds to the defined position for WIND ALARM ENDS.

The same is done for the rain alarm input. Please see chapter wind and rain alarm behavior.

BLIND & SHUTTER GROUP: DIGITAL INPUT CONFIGURATION						
DIGITAL INPUT GROUP1	3x01041 4x01041 I:1040	1,0x0001 B:00 01		N/A:NO CHANGE	UINT16 R/W	NO
		UP DOWN 1		SELECT FROM LIST		
Function for the digital input group IN1: ⬆DI1+⬇DI2 (S:SHUTTER, B:BLIND) 0:DEACTIVATED 1:UP DOWN 1:SHORT UP: S+B:MOVE TO 0%, SHORT DOWN: S+B:MOVE TO 100%, LONG UP:S:NOTHING,B:STEP SLATS UP, LONG DOWN:S:NOTHING,B:STEP SLATS DOWN 2:UP DOWN 2:SHORT UP: S+B:MOVE TO 0%, SHORT DOWN: S+B:MOVE TO 100%, LONG UP:S:STEP UP,B:STEP SLATS UP, LONG DOWN:S:STEP DOWN,B:STEP SLATS DOWN 3:UP DOWN 3:SHORT UP: S+B:MOVE TO 0%, SHORT DOWN: S+B:MOVE TO 100%, LONG UP:S+B:MOVE UP, LONG DOWN:S+B:MOVE DOWN 4:WIND+RAIN:DI1:WIND ALARM INPUT, DI2:RAIN ALARM INPUT						
DIGITAL INPUT GROUP2	3x01042 4x01042 I:1041	0,0x0000 B:00 00		N/A:NO CHANGE	UINT16 R/W	NO
		DEACTIVATED		SELECT FROM LIST		
Function for the digital input group IN2: ⬆DI1+⬇DI2 (S:SHUTTER, B:BLIND) 0:DEACTIVATED 1:UP DOWN 1:SHORT UP: S+B:MOVE TO 0%, SHORT DOWN: S+B:MOVE TO 100%, LONG UP:S:NOTHING,B:STEP SLATS UP, LONG DOWN:S:NOTHING,B:STEP SLATS DOWN 2:UP DOWN 2:SHORT UP: S+B:MOVE TO 0%, SHORT DOWN: S+B:MOVE TO 100%, LONG UP:S:STEP UP,B:STEP SLATS UP, LONG DOWN:S:STEP DOWN,B:STEP SLATS DOWN 3:UP DOWN 3:SHORT UP: S+B:MOVE TO 0%, SHORT DOWN: S+B:MOVE TO 100%, LONG UP:S+B:MOVE UP, LONG DOWN:S+B:MOVE DOWN 4:WIND+RAIN:DI1:WIND ALARM INPUT, DI2:RAIN ALARM INPUT						
DIGITAL INPUT GROUP3	3x01043 4x01043 I:1042	0,0x0000 B:00 00		N/A:NO CHANGE	UINT16 R/W	NO
		DEACTIVATED		SELECT FROM LIST		
Function for the digital input group IN3: ⬆DI1+⬇DI2 (S:SHUTTER, B:BLIND) 0:DEACTIVATED 1:UP DOWN 1:SHORT UP: S+B:MOVE TO 0%, SHORT DOWN: S+B:MOVE TO 100%, LONG UP:S:NOTHING,B:STEP SLATS UP, LONG DOWN:S:NOTHING,B:STEP SLATS DOWN 2:UP DOWN 2:SHORT UP: S+B:MOVE TO 0%, SHORT DOWN: S+B:MOVE TO 100%, LONG UP:S:STEP UP,B:STEP SLATS UP, LONG DOWN:S:STEP DOWN,B:STEP SLATS DOWN 3:UP DOWN 3:SHORT UP: S+B:MOVE TO 0%, SHORT DOWN: S+B:MOVE TO 100%, LONG UP:S+B:MOVE UP, LONG DOWN:S+B:MOVE DOWN 4:WIND+RAIN:DI1:WIND ALARM INPUT, DI2:RAIN ALARM INPUT						
DIGITAL INPUT GROUP4	3x01044 4x01044 I:1043	0,0x0000 B:00 00		N/A:NO CHANGE	UINT16 R/W	NO
		DEACTIVATED		SELECT FROM LIST		
Function for the digital input group IN4: ⬆DI1+⬇DI2 (S:SHUTTER, B:BLIND) 0:DEACTIVATED 1:UP DOWN 1:SHORT UP: S+B:MOVE TO 0%, SHORT DOWN: S+B:MOVE TO 100%, LONG UP:S:NOTHING,B:STEP SLATS UP, LONG DOWN:S:NOTHING,B:STEP SLATS DOWN 2:UP DOWN 2:SHORT UP: S+B:MOVE TO 0%, SHORT DOWN: S+B:MOVE TO 100%, LONG UP:S:STEP UP,B:STEP SLATS UP, LONG DOWN:S:STEP DOWN,B:STEP SLATS DOWN 3:UP DOWN 3:SHORT UP: S+B:MOVE TO 0%, SHORT DOWN: S+B:MOVE TO 100%, LONG UP:S+B:MOVE UP, LONG DOWN:S+B:MOVE DOWN 4:WIND+RAIN:DI1:WIND ALARM INPUT, DI2:RAIN ALARM INPUT						
DIGITAL INPUT GROUP5	3x01045 4x01045 I:1044	0,0x0000 B:00 00		N/A:NO CHANGE	UINT16 R/W	NO
		DEACTIVATED		SELECT FROM LIST		
Function for the digital input group IN5: ⬆DI1+⬇DI2 (S:SHUTTER, B:BLIND) 0:DEACTIVATED 1:UP DOWN 1:SHORT UP: S+B:MOVE TO 0%, SHORT DOWN: S+B:MOVE TO 100%, LONG UP:S:NOTHING,B:STEP SLATS UP, LONG DOWN:S:NOTHING,B:STEP SLATS DOWN 2:UP DOWN 2:SHORT UP: S+B:MOVE TO 0%, SHORT DOWN: S+B:MOVE TO 100%, LONG UP:S:STEP UP,B:STEP SLATS UP, LONG DOWN:S:STEP DOWN,B:STEP SLATS DOWN 3:UP DOWN 3:SHORT UP: S+B:MOVE TO 0%, SHORT DOWN: S+B:MOVE TO 100%, LONG UP:S+B:MOVE UP, LONG DOWN:S+B:MOVE DOWN 4:WIND+RAIN:DI1:WIND ALARM INPUT, DI2:RAIN ALARM INPUT						
DIGITAL INPUT GROUP6	3x01046 4x01046 I:1045	0,0x0000 B:00 00		N/A:NO CHANGE	UINT16 R/W	NO
		DEACTIVATED		SELECT FROM LIST		
Function for the digital input group IN6: ⬆DI1+⬇DI2 (S:SHUTTER, B:BLIND) 0:DEACTIVATED 1:UP DOWN 1:SHORT UP: S+B:MOVE TO 0%, SHORT DOWN: S+B:MOVE TO 100%, LONG UP:S:NOTHING,B:STEP SLATS UP, LONG DOWN:S:NOTHING,B:STEP SLATS DOWN 2:UP DOWN 2:SHORT UP: S+B:MOVE TO 0%, SHORT DOWN: S+B:MOVE TO 100%, LONG UP:S:STEP UP,B:STEP SLATS UP, LONG DOWN:S:STEP DOWN,B:STEP SLATS DOWN 3:UP DOWN 3:SHORT UP: S+B:MOVE TO 0%, SHORT DOWN: S+B:MOVE TO 100%, LONG UP:S+B:MOVE UP, LONG DOWN:S+B:MOVE DOWN 4:WIND+RAIN:DI1:WIND ALARM INPUT, DI2:RAIN ALARM INPUT						
DIGITAL INPUT GROUP7	3x01047 4x01047 I:1046	0,0x0000 B:00 00		N/A:NO CHANGE	UINT16 R/W	NO
		DEACTIVATED		SELECT FROM LIST		
Function for the digital input group IN7: ⬆DI1+⬇DI2 (S:SHUTTER, B:BLIND) 0:DEACTIVATED 1:UP DOWN 1:SHORT UP: S+B:MOVE TO 0%, SHORT DOWN: S+B:MOVE TO 100%, LONG UP:S:NOTHING,B:STEP SLATS UP, LONG DOWN:S:NOTHING,B:STEP SLATS DOWN 2:UP DOWN 2:SHORT UP: S+B:MOVE TO 0%, SHORT DOWN: S+B:MOVE TO 100%, LONG UP:S:STEP UP,B:STEP SLATS UP, LONG DOWN:S:STEP DOWN,B:STEP SLATS DOWN 3:UP DOWN 3:SHORT UP: S+B:MOVE TO 0%, SHORT DOWN: S+B:MOVE TO 100%, LONG UP:S+B:MOVE UP, LONG DOWN:S+B:MOVE DOWN 4:WIND+RAIN:DI1:WIND ALARM INPUT, DI2:RAIN ALARM INPUT						
DIGITAL INPUT GROUP8	3x01048 4x01048 I:1047	0,0x0000 B:00 00		N/A:NO CHANGE	UINT16 R/W	NO
		DEACTIVATED		SELECT FROM LIST		
Function for the digital input group IN8: ⬆DI1+⬇DI2 (S:SHUTTER, B:BLIND) 0:DEACTIVATED 1:UP DOWN 1:SHORT UP: S+B:MOVE TO 0%, SHORT DOWN: S+B:MOVE TO 100%, LONG UP:S:NOTHING,B:STEP SLATS UP, LONG DOWN:S:NOTHING,B:STEP SLATS DOWN 2:UP DOWN 2:SHORT UP: S+B:MOVE TO 0%, SHORT DOWN: S+B:MOVE TO 100%, LONG UP:S:STEP UP,B:STEP SLATS UP, LONG DOWN:S:STEP DOWN,B:STEP SLATS DOWN 3:UP DOWN 3:SHORT UP: S+B:MOVE TO 0%, SHORT DOWN: S+B:MOVE TO 100%, LONG UP:S+B:MOVE UP, LONG DOWN:S+B:MOVE DOWN 4:WIND+RAIN:DI1:WIND ALARM INPUT, DI2:RAIN ALARM INPUT						
DIGITAL INPUT GROUP9	3x01049 4x01049 I:1048	4,0x0004 B:00 04		N/A:NO CHANGE	UINT16 R/W	NO
		WIND+RAIN		SELECT FROM LIST		
Function for the digital input group INPUTS: IN1+IN2 (S:SHUTTER, B:BLIND) 0:DEACTIVATED 1:UP DOWN 1:SHORT UP: S+B:MOVE TO 0%, SHORT DOWN: S+B:MOVE TO 100%, LONG UP:S:NOTHING,B:STEP SLATS UP, LONG DOWN:S:NOTHING,B:STEP SLATS DOWN 2:UP DOWN 2:SHORT UP: S+B:MOVE TO 0%, SHORT DOWN: S+B:MOVE TO 100%, LONG UP:S:STEP UP,B:STEP SLATS UP, LONG DOWN:S:STEP DOWN,B:STEP SLATS DOWN 3:UP DOWN 3:SHORT UP: S+B:MOVE TO 0%, SHORT DOWN: S+B:MOVE TO 100%, LONG UP:S+B:MOVE UP, LONG DOWN:S+B:MOVE DOWN 4:WIND+RAIN:DI1:WIND ALARM INPUT, DI2:RAIN ALARM INPUT						
DIGITAL INPUT GROUP10	3x01050 4x01050 I:1049	0,0x0000 B:00 00		N/A:NO CHANGE	UINT16 R/W	NO
		DEACTIVATED		SELECT FROM LIST		
Function for the digital input group INPUTS: IN3+IN4 (S:SHUTTER, B:BLIND) 0:DEACTIVATED 1:UP DOWN 1:SHORT UP: S+B:MOVE TO 0%, SHORT DOWN: S+B:MOVE TO 100%, LONG UP:S:NOTHING,B:STEP SLATS UP, LONG DOWN:S:NOTHING,B:STEP SLATS DOWN 2:UP DOWN 2:SHORT UP: S+B:MOVE TO 0%, SHORT DOWN: S+B:MOVE TO 100%, LONG UP:S:STEP UP,B:STEP SLATS UP, LONG DOWN:S:STEP DOWN,B:STEP SLATS DOWN 3:UP DOWN 3:SHORT UP: S+B:MOVE TO 0%, SHORT DOWN: S+B:MOVE TO 100%, LONG UP:S+B:MOVE UP, LONG DOWN:S+B:MOVE DOWN 4:WIND+RAIN:DI1:WIND ALARM INPUT, DI2:RAIN ALARM INPUT						

20.2.4 MODBUS input configuration of shade/blind outlets

To ease the use of the IO module with MODBUS master controllers like DDCs, touchbpanels, Micro controller, … we have implemented virtual digital inputs in MODBUS registers for every engine outlet. Each engine outlet OUTx can be controlled directly with a few MODBUS registers described below.

For each engine outlet you can configure 10 (for RESI-20RI8SB-SIO) or 5 (for RESI-10RI4SB-SIO) MODBUS registers named MODBUS INPUT GROUP1 to MODBUS INPUT GROUP10.

Each MODBUS input group is configured in the same way like the DIGITAL INPUT GROUPx. So see the previous chapter, what the modes mean in detail.

In addition the controller offers the general MODBUS holding registers MODBUS DIGITAL INPUT MBDI1 to MODBUS DIGITAL INPUT MBDI20, which represents a virtual digital input controlled by MODBUS write commands.

Two MODBUS registers are always related the one MODBUS INPUT GROUP:

MODBUS INPUT GROUP1 uses MODBUS DIGITAL INPUT MBDI1 and MODBUS DIGITAL INPUT MBDI2

MODBUS INPUT GROUP2 uses MODBUS DIGITAL INPUT MBDI3 and MODBUS DIGITAL INPUT MBDI4

MODBUS INPUT GROUP3 uses MODBUS DIGITAL INPUT MBDI5 and MODBUS DIGITAL INPUT MBDI6

…

MODBUS INPUT GROUP10 uses MODBUS DIGITAL INPUT MBDI19 and MODBUS DIGITAL INPUT MBDI20

You can write the following values into the register pairs MODBUS DIGITAL INPUT MBDIx (x=1 to 20):

First register (e.g. MBDI1)

=0: NOTHING happens

=1: Execute a short key press action. Depending on the configuration of the current MODBUS INPUT GROUPx mode, the shade/blind starts to move up or stops the current movement.

=2: Execute a long key press start action. Depending on the configuration of the current MODBUS INPUT GROUPx mode, the blind starts to step the slats up or steps the position up.

=3: Execute a long key press end action. Depending on the configuration of the current MODBUS INPUT GROUPx mode, the blind stops the current movement of the slats/shade.

Second register (e.g. MBDI2)

=0: NOTHING happens

=1: Execute a short keypress. Depending on the configuration of the current MODBUS INPUT GROUPx mode, the shade/blind starts to move down or stops the current movement.

=2: Execute a long key press start action. Depending on the configuration of the current MODBUS INPUT GROUPx mode, the blind starts to step the slats down or steps the position up.

=3: Execute a long key press end action. Depending on the configuration of the current MODBUS INPUT GROUPx mode, the blind stops the current movement of the slats/shade.

So in your application you need to set up the configuration registers and the n you can use the digital inputs with push buttons to trigger/stop your shades/blinds or you use the MODBUS DIGITAL INPUT MBDIx registers to trigger/stop the movement of the shades/blinds!

BLIND & SHUTTER GROUP: MODBUS INPUT CONFIGURATION						
MODBUS INPUT GROUP1	3x01051 4x01051 I:1050	1,0x0001 B:00 01		N/A:NO CHANGE	UINT16 R/W	NO
		UP DOWN 1		SELECT FROM LIST		
Function for the MODBUS input group #1: (S:SHUTTER, B:BLIND) 0:DEACTIVATED 1:UP DOWN 1:SHORT UP: S+B:MOVE TO 0%, SHORT DOWN: S+B:MOVE TO 100%, LONG UP:S:NOTHING,B:STEP SLATS UP, LONG DOWN:S:NOTHING,B:STEP SLATS DOWN 2:UP DOWN 2:SHORT UP: S+B:MOVE TO 0%, SHORT DOWN: S+B:MOVE TO 100%, LONG UP:S:STEP UP,B:STEP SLATS UP, LONG DOWN:S:STEP DOWN,B:STEP SLATS DOWN 3:UP DOWN 3:SHORT UP: S+B:MOVE TO 0%, SHORT DOWN: S+B:MOVE TO 100%, LONG UP:S+B:MOVE UP, LONG DOWN:S+B:MOVE DOWN 4:WIND+RAIN:DI1:WIND ALARM INPUT, DI2:RAIN ALARM INPUT						
MODBUS INPUT GROUP2	3x01052 4x01052 I:1051	0,0x0000 B:00 00		N/A:NO CHANGE	UINT16 R/W	NO
		DEACTIVATED		SELECT FROM LIST		
Function for the MODBUS input group #2: (S:SHUTTER, B:BLIND) 0:DEACTIVATED 1:UP DOWN 1:SHORT UP: S+B:MOVE TO 0%, SHORT DOWN: S+B:MOVE TO 100%, LONG UP:S:NOTHING,B:STEP SLATS UP, LONG DOWN:S:NOTHING,B:STEP SLATS DOWN 2:UP DOWN 2:SHORT UP: S+B:MOVE TO 0%, SHORT DOWN: S+B:MOVE TO 100%, LONG UP:S:STEP UP,B:STEP SLATS UP, LONG DOWN:S:STEP DOWN,B:STEP SLATS DOWN 3:UP DOWN 3:SHORT UP: S+B:MOVE TO 0%, SHORT DOWN: S+B:MOVE TO 100%, LONG UP:S+B:MOVE UP, LONG DOWN:S+B:MOVE DOWN 4:WIND+RAIN:DI1:WIND ALARM INPUT, DI2:RAIN ALARM INPUT						
MODBUS INPUT GROUP3	3x01053 4x01053 I:1052	0,0x0000 B:00 00		N/A:NO CHANGE	UINT16 R/W	NO
		DEACTIVATED		SELECT FROM LIST		
Function for the MODBUS input group #3: (S:SHUTTER, B:BLIND) 0:DEACTIVATED 1:UP DOWN 1:SHORT UP: S+B:MOVE TO 0%, SHORT DOWN: S+B:MOVE TO 100%, LONG UP:S:NOTHING,B:STEP SLATS UP, LONG DOWN:S:NOTHING,B:STEP SLATS DOWN 2:UP DOWN 2:SHORT UP: S+B:MOVE TO 0%, SHORT DOWN: S+B:MOVE TO 100%, LONG UP:S:STEP UP,B:STEP SLATS UP, LONG DOWN:S:STEP DOWN,B:STEP SLATS DOWN 3:UP DOWN 3:SHORT UP: S+B:MOVE TO 0%, SHORT DOWN: S+B:MOVE TO 100%, LONG UP:S+B:MOVE UP, LONG DOWN:S+B:MOVE DOWN 4:WIND+RAIN:DI1:WIND ALARM INPUT, DI2:RAIN ALARM INPUT						
MODBUS INPUT GROUP4	3x01054 4x01054 I:1053	0,0x0000 B:00 00		N/A:NO CHANGE	UINT16 R/W	NO
		DEACTIVATED		SELECT FROM LIST		
Function for the MODBUS input group #4: (S:SHUTTER, B:BLIND) 0:DEACTIVATED 1:UP DOWN 1:SHORT UP: S+B:MOVE TO 0%, SHORT DOWN: S+B:MOVE TO 100%, LONG UP:S:NOTHING,B:STEP SLATS UP, LONG DOWN:S:NOTHING,B:STEP SLATS DOWN 2:UP DOWN 2:SHORT UP: S+B:MOVE TO 0%, SHORT DOWN: S+B:MOVE TO 100%, LONG UP:S:STEP UP,B:STEP SLATS UP, LONG DOWN:S:STEP DOWN,B:STEP SLATS DOWN 3:UP DOWN 3:SHORT UP: S+B:MOVE TO 0%, SHORT DOWN: S+B:MOVE TO 100%, LONG UP:S+B:MOVE UP, LONG DOWN:S+B:MOVE DOWN 4:WIND+RAIN:DI1:WIND ALARM INPUT, DI2:RAIN ALARM INPUT						
MODBUS INPUT GROUP5	3x01055 4x01055 I:1054	0,0x0000 B:00 00		N/A:NO CHANGE	UINT16 R/W	NO
		DEACTIVATED		SELECT FROM LIST		
Function for the MODBUS input group #5: (S:SHUTTER, B:BLIND) 0:DEACTIVATED 1:UP DOWN 1:SHORT UP: S+B:MOVE TO 0%, SHORT DOWN: S+B:MOVE TO 100%, LONG UP:S:NOTHING,B:STEP SLATS UP, LONG DOWN:S:NOTHING,B:STEP SLATS DOWN 2:UP DOWN 2:SHORT UP: S+B:MOVE TO 0%, SHORT DOWN: S+B:MOVE TO 100%, LONG UP:S:STEP UP,B:STEP SLATS UP, LONG DOWN:S:STEP DOWN,B:STEP SLATS DOWN 3:UP DOWN 3:SHORT UP: S+B:MOVE TO 0%, SHORT DOWN: S+B:MOVE TO 100%, LONG UP:S+B:MOVE UP, LONG DOWN:S+B:MOVE DOWN 4:WIND+RAIN:DI1:WIND ALARM INPUT, DI2:RAIN ALARM INPUT						
MODBUS INPUT GROUP6	3x01056 4x01056 I:1055	0,0x0000 B:00 00		N/A:NO CHANGE	UINT16 R/W	NO
		DEACTIVATED		SELECT FROM LIST		
Function for the MODBUS input group #6: (S:SHUTTER, B:BLIND) 0:DEACTIVATED 1:UP DOWN 1:SHORT UP: S+B:MOVE TO 0%, SHORT DOWN: S+B:MOVE TO 100%, LONG UP:S:NOTHING,B:STEP SLATS UP, LONG DOWN:S:NOTHING,B:STEP SLATS DOWN 2:UP DOWN 2:SHORT UP: S+B:MOVE TO 0%, SHORT DOWN: S+B:MOVE TO 100%, LONG UP:S:STEP UP,B:STEP SLATS UP, LONG DOWN:S:STEP DOWN,B:STEP SLATS DOWN 3:UP DOWN 3:SHORT UP: S+B:MOVE TO 0%, SHORT DOWN: S+B:MOVE TO 100%, LONG UP:S+B:MOVE UP, LONG DOWN:S+B:MOVE DOWN 4:WIND+RAIN:DI1:WIND ALARM INPUT, DI2:RAIN ALARM INPUT						
MODBUS INPUT GROUP7	3x01057 4x01057 I:1056	0,0x0000 B:00 00		N/A:NO CHANGE	UINT16 R/W	NO
		DEACTIVATED		SELECT FROM LIST		
Function for the MODBUS input group #7: (S:SHUTTER, B:BLIND) 0:DEACTIVATED 1:UP DOWN 1:SHORT UP: S+B:MOVE TO 0%, SHORT DOWN: S+B:MOVE TO 100%, LONG UP:S:NOTHING,B:STEP SLATS UP, LONG DOWN:S:NOTHING,B:STEP SLATS DOWN 2:UP DOWN 2:SHORT UP: S+B:MOVE TO 0%, SHORT DOWN: S+B:MOVE TO 100%, LONG UP:S:STEP UP,B:STEP SLATS UP, LONG DOWN:S:STEP DOWN,B:STEP SLATS DOWN 3:UP DOWN 3:SHORT UP: S+B:MOVE TO 0%, SHORT DOWN: S+B:MOVE TO 100%, LONG UP:S+B:MOVE UP, LONG DOWN:S+B:MOVE DOWN 4:WIND+RAIN:DI1:WIND ALARM INPUT, DI2:RAIN ALARM INPUT						
MODBUS INPUT GROUP8	3x01058 4x01058 I:1057	0,0x0000 B:00 00		N/A:NO CHANGE	UINT16 R/W	NO
		DEACTIVATED		SELECT FROM LIST		
Function for the MODBUS input group #8: (S:SHUTTER, B:BLIND) 0:DEACTIVATED 1:UP DOWN 1:SHORT UP: S+B:MOVE TO 0%, SHORT DOWN: S+B:MOVE TO 100%, LONG UP:S:NOTHING,B:STEP SLATS UP, LONG DOWN:S:NOTHING,B:STEP SLATS DOWN 2:UP DOWN 2:SHORT UP: S+B:MOVE TO 0%, SHORT DOWN: S+B:MOVE TO 100%, LONG UP:S:STEP UP,B:STEP SLATS UP, LONG DOWN:S:STEP DOWN,B:STEP SLATS DOWN 3:UP DOWN 3:SHORT UP: S+B:MOVE TO 0%, SHORT DOWN: S+B:MOVE TO 100%, LONG UP:S+B:MOVE UP, LONG DOWN:S+B:MOVE DOWN 4:WIND+RAIN:DI1:WIND ALARM INPUT, DI2:RAIN ALARM INPUT						
MODBUS INPUT GROUP9	3x01059 4x01059 I:1058	4,0x0004 B:00 04		N/A:NO CHANGE	UINT16 R/W	NO
		WIND+RAIN		SELECT FROM LIST		
Function for the MODBUS input group #9: (S:SHUTTER, B:BLIND) 0:DEACTIVATED 1:UP DOWN 1:SHORT UP: S+B:MOVE TO 0%, SHORT DOWN: S+B:MOVE TO 100%, LONG UP:S:NOTHING,B:STEP SLATS UP, LONG DOWN:S:NOTHING,B:STEP SLATS DOWN 2:UP DOWN 2:SHORT UP: S+B:MOVE TO 0%, SHORT DOWN: S+B:MOVE TO 100%, LONG UP:S:STEP UP,B:STEP SLATS UP, LONG DOWN:S:STEP DOWN,B:STEP SLATS DOWN 3:UP DOWN 3:SHORT UP: S+B:MOVE TO 0%, SHORT DOWN: S+B:MOVE TO 100%, LONG UP:S+B:MOVE UP, LONG DOWN:S+B:MOVE DOWN 4:WIND+RAIN:DI1:WIND ALARM INPUT, DI2:RAIN ALARM INPUT						
MODBUS INPUT GROUP10	3x01060 4x01060 I:1059	0,0x0000 B:00 00		N/A:NO CHANGE	UINT16 R/W	NO
		DEACTIVATED		SELECT FROM LIST		
Function for the MODBUS input group #10: (S:SHUTTER, B:BLIND) 0:DEACTIVATED 1:UP DOWN 1:SHORT UP: S+B:MOVE TO 0%, SHORT DOWN: S+B:MOVE TO 100%, LONG UP:S:NOTHING,B:STEP SLATS UP, LONG DOWN:S:NOTHING,B:STEP SLATS DOWN 2:UP DOWN 2:SHORT UP: S+B:MOVE TO 0%, SHORT DOWN: S+B:MOVE TO 100%, LONG UP:S:STEP UP,B:STEP SLATS UP, LONG DOWN:S:STEP DOWN,B:STEP SLATS DOWN 3:UP DOWN 3:SHORT UP: S+B:MOVE TO 0%, SHORT DOWN: S+B:MOVE TO 100%, LONG UP:S+B:MOVE UP, LONG DOWN:S+B:MOVE DOWN 4:WIND+RAIN:DI1:WIND ALARM INPUT, DI2:RAIN ALARM INPUT						

MODBUS DIGITAL INPUTS: CURRENT STATUS OF ALL MODBUS DIGITAL INPUTS MBDI1..MBDI20						
MODBUS DIGITAL INPUT MBDI1	3x10101 4x10101 I:10100	????		N/A:NO CHANGE	UINT16 R/W	NO
		????		SELECT FROM LIST		
Current status of MODBUS digital input MBDIx. =0:Nothing =1:Execute short keypress =2:Execute long keypress start =3:Execute long keypress end						
MODBUS DIGITAL INPUT MBDI2	3x10102 4x10102 I:10101	????		1:SHORT KEYPRESS	UINT16 R/W	YES
		????		SELECT FROM LIST		
Current status of MODBUS digital input MBDIx. =0:Nothing =1:Execute short keypress =2:Execute long keypress start =3:Execute long keypress end						
MODBUS DIGITAL INPUT MBDI3	3x10103 4x10103 I:10102	????		N/A:NO CHANGE	UINT16 R/W	NO
		????		SELECT FROM LIST		
Current status of MODBUS digital input MBDIx. =0:Nothing =1:Execute short keypress =2:Execute long keypress start =3:Execute long keypress end						
MODBUS DIGITAL INPUT MBDI4	3x10104 4x10104 I:10103	????		N/A:NO CHANGE	UINT16 R/W	NO
		????		SELECT FROM LIST		
Current status of MODBUS digital input MBDIx. =0:Nothing =1:Execute short keypress =2:Execute long keypress start =3:Execute long keypress end						
MODBUS DIGITAL INPUT MBDI5	3x10105 4x10105 I:10104	????		N/A:NO CHANGE	UINT16 R/W	NO
		????		SELECT FROM LIST		
Current status of MODBUS digital input MBDIx. =0:Nothing =1:Execute short keypress =2:Execute long keypress start =3:Execute long keypress end						
MODBUS DIGITAL INPUT MBDI6	3x10106 4x10106 I:10105	????		N/A:NO CHANGE	UINT16 R/W	NO
		????		SELECT FROM LIST		
Current status of MODBUS digital input MBDIx. =0:Nothing =1:Execute short keypress =2:Execute long keypress start =3:Execute long keypress end						
MODBUS DIGITAL INPUT MBDI7	3x10107 4x10107 I:10106	????		N/A:NO CHANGE	UINT16 R/W	NO
		????		SELECT FROM LIST		
Current status of MODBUS digital input MBDIx. =0:Nothing =1:Execute short keypress =2:Execute long keypress start =3:Execute long keypress end						
MODBUS DIGITAL INPUT MBDI8	3x10108 4x10108 I:10107	????		N/A:NO CHANGE	UINT16 R/W	NO
		????		SELECT FROM LIST		
Current status of MODBUS digital input MBDIx. =0:Nothing =1:Execute short keypress =2:Execute long keypress start =3:Execute long keypress end						
MODBUS DIGITAL INPUT MBDI9	3x10109 4x10109 I:10108	????		N/A:NO CHANGE	UINT16 R/W	NO
		????		SELECT FROM LIST		
Current status of MODBUS digital input MBDIx. =0:Nothing =1:Execute short keypress =2:Execute long keypress start =3:Execute long keypress end						
MODBUS DIGITAL INPUT MBDI10	3x10110 4x10110 I:10109	????		N/A:NO CHANGE	UINT16 R/W	NO
		????		SELECT FROM LIST		
Current status of MODBUS digital input MBDIx. =0:Nothing =1:Execute short keypress =2:Execute long keypress start =3:Execute long keypress end						

MODBUS DIGITAL INPUT MBDI11	3x10111 4x10111 I:10110	????		N/A:NO CHANGE	UINT16 R/W	NO
		????		SELECT FROM LIST		
Current status of MODBUS digital input MBDIx. =0:Nothing =1:Execute short keypress =2:Execute long keypress start =3:Execute long keypress end						
MODBUS DIGITAL INPUT MBDI12	3x10112 4x10112 I:10111	????		N/A:NO CHANGE	UINT16 R/W	NO
		????		SELECT FROM LIST		
Current status of MODBUS digital input MBDIx. =0:Nothing =1:Execute short keypress =2:Execute long keypress start =3:Execute long keypress end						
MODBUS DIGITAL INPUT MBDI13	3x10113 4x10113 I:10112	????		N/A:NO CHANGE	UINT16 R/W	NO
		????		SELECT FROM LIST		
Current status of MODBUS digital input MBDIx. =0:Nothing =1:Execute short keypress =2:Execute long keypress start =3:Execute long keypress end						
MODBUS DIGITAL INPUT MBDI14	3x10114 4x10114 I:10113	????		N/A:NO CHANGE	UINT16 R/W	NO
		????		SELECT FROM LIST		
Current status of MODBUS digital input MBDIx. =0:Nothing =1:Execute short keypress =2:Execute long keypress start =3:Execute long keypress end						
MODBUS DIGITAL INPUT MBDI15	3x10115 4x10115 I:10114	????		N/A:NO CHANGE	UINT16 R/W	NO
		????		SELECT FROM LIST		
Current status of MODBUS digital input MBDIx. =0:Nothing =1:Execute short keypress =2:Execute long keypress start =3:Execute long keypress end						
MODBUS DIGITAL INPUT MBDI16	3x10116 4x10116 I:10115	????		N/A:NO CHANGE	UINT16 R/W	NO
		????		SELECT FROM LIST		
Current status of MODBUS digital input MBDIx. =0:Nothing =1:Execute short keypress =2:Execute long keypress start =3:Execute long keypress end						
MODBUS DIGITAL INPUT MBDI17	3x10117 4x10117 I:10116	????		N/A:NO CHANGE	UINT16 R/W	NO
		????		SELECT FROM LIST		
Current status of MODBUS digital input MBDIx. =0:Nothing =1:Execute short keypress =2:Execute long keypress start =3:Execute long keypress end						
MODBUS DIGITAL INPUT MBDI18	3x10118 4x10118 I:10117	????		N/A:NO CHANGE	UINT16 R/W	NO
		????		SELECT FROM LIST		
Current status of MODBUS digital input MBDIx. =0:Nothing =1:Execute short keypress =2:Execute long keypress start =3:Execute long keypress end						
MODBUS DIGITAL INPUT MBDI19	3x10119 4x10119 I:10118	????		N/A:NO CHANGE	UINT16 R/W	NO
		????		SELECT FROM LIST		
Current status of MODBUS digital input MBDIx. =0:Nothing =1:Execute short keypress =2:Execute long keypress start =3:Execute long keypress end						
MODBUS DIGITAL INPUT MBDI20	3x10120 4x10120 I:10119	????		N/A:NO CHANGE	UINT16 R/W	NO
		????		SELECT FROM LIST		
Current status of MODBUS digital input MBDIx. =0:Nothing =1:Execute short keypress =2:Execute long keypress start =3:Execute long keypress end						

20.2.5 Wind and rain alarm behavior

You can configure the behaviour in case of wind and rain alarm. Therefore you can configure, which digital input is used for wind alarm and which digital input is used for rain alarm for each engine outlet individually.

Use the registers DIGITAL INPUT GROUPx and mode 4:WIND+RAIN ALARM to define which digital inputs are used to trigger the wind and rain alarms.

You can also trigger the wind and rain alarm via simulated MODBUS DIGITAL INPUT MBDIx. Define the used MODBUS registers in MODBUS INPUT GROUPx by writing mode 4:WIND+RAIN ALARM into the corresponding register.

Or write the specific command into the COMMAND register of every engine outlet group to activate/deactivate WIND or RAIN alarm.

For every engine outlet you can define, what happens in case of a wind alarm or a rain alarm. For this, the follwing configuration registers are used:

BILIND & SHUTTER GROUP: WINDALARM CONFIGURATION						
WIND START MODE	3x01101 4x01101 I:1100	3,0x0003 B:00 03		N/A:NO CHANGE	UINT16 R/W	NO
		MOVE TO POSITION		SELECT FROM LIST		
Configured start mode for the wind alarm function: = 0: DEACTIVATED: Nothing happens when wind alarm rises = 1: MOVE 0%: Move to position 0% = 2: MOVE 100%: Move to position 100% = 3: MOVE POS: Move to position WIND POSITION, WIND SLAT POSITION = 4: MOVE LAST POS: Do nothing						
WIND END MODE	3x01102 4x01102 I:1101	4,0x0004 B:00 04		N/A:NO CHANGE	UINT16 R/W	NO
		MOVE TO LAST POSITION		SELECT FROM LIST		
Configured end mode for the wind alarm function: = 0: DEACTIVATED: Nothing happens when wind alarm ends = 1: MOVE 0%: Move to position 0% = 2: MOVE 100%: Move to position 100% = 3: MOVE POS: Move to position WIND POSITION, WIND SLAT POSITION = 4: MOVE LAST POS: Move to last position before wind alarm was triggered						
WIND POSITION	3x01103 4x01103 I:1102	0,0x0000 B:00 00	0		UINT16 R/W	NO
		00,00%		VALUE IN XX,XX%		
Vertical position for this blind / shutter in percent for wind alarm mode MOVE POS 0%:complete open (upper position) 100%:complete closed (lower position)						
WIND SLAT POSITION	3x01104 4x01104 I:1103	0,0x0000 B:00 00	0		UINT16 R/W	NO
		00,00%		VALUE IN XX,XX%		
Vertical position for the slats of this blind in percent for wind alarm mode MOVE POS 0%:in position SLAT ANGLE UP 100%:in position SLAT ANGLE DOWN						

WIND START MODE defines what happens, if the wind alarm arises. You can choose between:
=0: DEACTIVATED: Nothing happens in case of a wind alarm
=1: MOVE TO 0%: Move the shade/blind completely up to position 0% (Fully opened)
=2: MOVE TO 100%: Move the shade/blind completely down to position 100% (Fully closed). In this case the slats are also closed.
=3: MOVE TO WIND POSITION+WIND SLAT POSITION: You can define an individual position for every engine outlet in the two registers WIND POSITION between 0% and 100% and WIND SLAT POSITION between 0% and 100%. This should be a save position, where you cannot destroy the shade/blind due to too much wind!
=4: MOVE LAST POSITION: Nothing happens in case of a wind alarm

WIND END MODE defines what happens, if the wind alarm goes away. You can choose between:
=0: DEACTIVATED: Nothing happens, the shade/blind stays in the current position
=1: MOVE TO 0%: Move the shade/blind completely up to position 0% (Fully opened)
=2: MOVE TO 100%: Move the shade/blind completely down to position 100% (Fully closed). In this case the slats are also closed.
=3: MOVE TO WIND POSITION+WIND SLAT POSITION: You can define an individual position for every engine outlet in the two registers WIND POSITION between 0% and 100% and WIND SLAT POSITION between 0% and 100%. This should be a save position, where you cannot destroy the shade/blind due to too much wind!
=4: MOVE LAST POSITION: Move to the last position before the wind alarm arises. This position is automatically stored by the IO module in case of an wind alarm.

BILIND & SHUTTER GROUP: RAIN ALARM CONFIGURATION						
RAIN START MODE	3x01105 4x01105 I:1104	3,0x0003 B:00 03		N/A:NO CHANGE	UINT16 R/W	NO
		MOVE TO POSITION		SELECT FROM LIST		
Configured start mode for the rain alarm function: = 0: DEACTIVATED: Nothing happens when rain alarm rises = 1: MOVE 0P: Move to position 0% = 2: MOVE 100%: Move to position 100% = 3: MOVE POS: Move to position RAIN POSITION, RAIN SLAT POSITION = 4: MOVE LAST POS: Do nothing						
RAIN END MODE	3x01106 4x01106 I:1105	4,0x0004 B:00 04		N/A:NO CHANGE	UINT16 R/W	NO
		MOVE TO LAST POSITION		SELECT FROM LIST		
Configured end mode for the wind alarm function: = 0: DEACTIVATED: Nothing happens when rain alarm ends = 1: MOVE 0P: Move to position 0% = 2: MOVE 100%: Move to position 100% = 3: MOVE POS: Move to position RAIN POSITION, RAIN SLAT POSITION = 4: MOVE LAST POS: Move to last position before rain alarm was triggered						
RAIN POSITION	3x01107 4x01107 I:1106	0,0x0000 B:00 00	0		UINT16 R/W	NO
		00,00%		VALUE IN XX,XX%		
Vertical position for this blind / shutter in percent for rain alarm mode MOVE POS 0%:complete open (upper position) 100%:complete closed (lower position)						
RAIN SLAT POSITION	3x01108 4x01108 I:1107	0,0x0000 B:00 00	0		UINT16 R/W	NO
		00,00%		VALUE IN XX,XX%		
Vertical position for the slats of this blind in percent for rain alarm mode MOVE POS 0%:in position SLAT ANGLE UP 100%:in position SLAT ANGLE DOWN						

RAIN START MODE defines what happens, if the rain alarm arises. You can choose between:
=0: DEACTIVATED: Nothing happens in case of a rain alarm
=1: MOVE TO 0%: Move the shade/blind completely up to position 0% (Fully opened)
=2: MOVE TO 100%: Move the shade/blind completely down to position 100% (Fully closed). In this case the slats are also closed.
=3: MOVE TO RAIN POSITION+RAIN SLAT POSITION: You can define an individual position for every engine outlet in the two registers RAIN POSITION between 0% and 100% and RAIN SLAT POSITION between 0% and 100%. This should be a save position, where you cannot destroy the shade/blind due to too much rain!
=4: MOVE LAST POSITION: Nothing happens in case of a rain alarm

RAIN END MODE defines what happens, if the rain alarm goes away. You can choose between:
=0: DEACTIVATED: Nothing happens, the shade/blind stays in the current position
=1: MOVE TO 0%: Move the shade/blind completely up to position 0% (Fully opened)
=2: MOVE TO 100%: Move the shade/blind completely down to position 100% (Fully closed). In this case the slats are also closed.
=3: MOVE TO RAIN POSITION+RAIN SLAT POSITION: You can define an individual position for every engine outlet in the two registers RAIN POSITION between 0% and 100% and RAIN SLAT POSITION between 0% and 100%. This should be a save position, where you cannot destroy the shade/blind due to too much rain!
=4: MOVE LAST POSITION: Move to the last position before the rain alarm arises. This position is automatically stored by the IO module in case of a rain alarm.

A word to the priority:
Priority 0: NORMAL OPERATION: User can use the digital and MODBUS inputs or send commands to the outlet, as long as the control is not locked by special commands.

Priority 1: RAIN ALARM: A occurring rain alarm will lock the NORMAL OPERATION and move the shades/blinds to the position defined by the RAIN ALARM configuration. If the rain alarm vanishes the shades/blinds are moved to the end positions defined in the rain alarm configuration. After that NORMAL OPERATION is active again.
But, this priority can be interrupted by a WIND ALARM!

Priority 2: WIND ALARM: A occurring wind alarm will lock the NORMAL OPERATION or interrupt an active RAIN ALARM and move the shades/blinds to the position defined by the WIND ALARM configuration. If the wind alarm vanishes the shades/blinds are moved to the end positions defined in the wind alarm configuration. If the rain alarm is still active, the positions of rain alarm are restored. In case of no rain alarm, NORMAL OPERATION is active again.

20.2.6 Status of all digital inputs or MODBUS digital inputs

The IO module offers a lot of MODBUS status registers for getting the current state of the digital inputs and the simulated MODBUS digital inputs.

You can request the current status for all digital inputs with the registers:

DIGITAL INPUTS: CURRENT STATUS OF ALL DIGITAL INPUTS DI1..DI20						
STATUS DI1-DI16	3x10002 4x10002 I:10001	0,0x0000 B:00 00			UINT16 R/O	
		0000.0000.0000.0000				
Returns the current state of digital inputs DI1 to DI16. Each bit stands for a digital input: Bit 0: IN1 ⬆ DI1 (=0:DI is OFF, =1:DI is ON) Bit 1: IN1 ⬇ DI2 (=0:DI is OFF, =1:DI is ON) Bit 2: IN2 ⬆ DI3 (=0:DI is OFF, =1:DI is ON) Bit 3: IN2 ⬇ DI4 (=0:DI is OFF, =1:DI is ON) Bit 4: IN3 ⬆ DI5 (=0:DI is OFF, =1:DI is ON) Bit 5: IN3 ⬇ DI6 (=0:DI is OFF, =1:DI is ON) Bit 6: IN4 ⬆ DI7 (=0:DI is OFF, =1:DI is ON) Bit 7: IN4 ⬇ DI8 (=0:DI is OFF, =1:DI is ON) Bit 8: IN5 ⬆ DI9 (=0:DI is OFF, =1:DI is ON) Bit 9: IN5 ⬇ DI10 (=0:DI is OFF, =1:DI is ON) Bit 10: IN6 ⬆ DI11 (=0:DI is OFF, =1:DI is ON) Bit 11: IN6 ⬇ DI12 (=0:DI is OFF, =1:DI is ON) Bit 12: IN7 ⬆ DI13 (=0:DI is OFF, =1:DI is ON) Bit 13: IN7 ⬇ DI14 (=0:DI is OFF, =1:DI is ON) Bit 14: IN8 ⬆ DI15 (=0:DI is OFF, =1:DI is ON) Bit 15: IN8 ⬇ DI16 (=0:DI is OFF, =1:DI is ON)						
STATUS DI17-DI20	3x10003 4x10003 I:10002	0,0x0000 B:00 00			UINT16 R/O	
		0000.0000.0000.0000				
Returns the current state of digital inputs DI17 to DI20. Each bit stands for a digital input: Bit 0: INPUTS IN1 DI17 (=0:DI is OFF, =1:DI is ON) Bit 1: INPUTS IN2 DI18 (=0:DI is OFF, =1:DI is ON) Bit 2: INPUTS IN3 DI19 (=0:DI is OFF, =1:DI is ON) Bit 3: INPUTS IN4 DI20 (=0:DI is OFF, =1:DI is ON) Bits 4-15: 0						

Also the IO module forms some counter registers for every digital input to count special events like short and long key press or rising and falling edges.

DIGITAL INPUTS: STATUS FOR DIGITAL INPUT DI1						
RISE DI1	3x20001 4x20001 I:20000	0,0x0000 B:00 00			UINT16 R/O	
		0 event(s)				
IN1 ⬆ DI1:Counter for rising edges on digital input DIx						
FALL DI1	3x20002 4x20002 I:20001	0,0x0000 B:00 00			UINT16 R/O	
		0 event(s)				
IN1 ⬆ DI1:Counter for falling edges on digital input DIx						
CHANGE DI1	3x20003 4x20003 I:20002	0,0x0000 B:00 00			UINT16 R/O	
		0 event(s)				
IN1 ⬆ DI1:Counter for status changes for digital input DIx						
SHORT KEYPRESS DI1	3x20004 4x20004 I:20003	0,0x0000 B:00 00			UINT16 R/O	
		0 event(s)				
IN1 ⬆ DI1:Counter for short keypress events for digital input DIx						
LONG KEYPRESS START DI1	3x20005 4x20005 I:20004	0,0x0000 B:00 00			UINT16 R/O	
		0 event(s)				
IN1 ⬆ DI1:Counter for long keypress start events for digital input DIx						
LONG KEYPRESS END DI1	3x20006 4x20006 I:20005	0,0x0000 B:00 00			UINT16 R/O	
		0 event(s)				
IN1 ⬆ DI1:Counter for long keypress end events for digital input DIx						

The same registers are calculated for the MODBUS INPUT REGISTER MBDIx:

DIGITAL INPUTS: STATUS FOR MODBUS DIGITAL INPUT DI1						
RISE MBDI1	3x20201 4x20201 I:20200	0,0x0000 B:00 00			UINT16 R/O	
		0 event(s)				
Counter for rising edges on MODBUS digital input DIx						
FALL MBDI1	3x20202 4x20202 I:20201	0,0x0000 B:00 00			UINT16 R/O	
		0 event(s)				
Counter for falling edges on MODBUS digital input DIx						
CHANGE MBDI1	3x20203 4x20203 I:20202	0,0x0000 B:00 00			UINT16 R/O	
		0 event(s)				
Counter for status changes for MODBUS digital input DIx						
SHORT KEYPRESS MBDI1	3x20204 4x20204 I:20203	0,0x0000 B:00 00			UINT16 R/O	
		0 event(s)				
Counter for short keypress events for MODBUS digital input DIx						
LONG KEYPRESS START MBDI1	3x20205 4x20205 I:20204	0,0x0000 B:00 00			UINT16 R/O	
		0 event(s)				
Counter for long keypress start events for MODBUS digital input DIx						
LONG KEYPRESS END MBDI1	3x20206 4x20206 I:20205	0,0x0000 B:00 00			UINT16 R/O	
		0 event(s)				
Counter for long keypress end events for MODBUS digital input DIx						

Also the usage of the digital inputs can be activated or deactivated with this register. This is useful for general time programs, which should enable/disable the use of local push buttons to control the shades/blinds.

LOCK:DIGITAL INPUTS						
LOCK DIGITAL INPUTS	3x30001 4x30001 I:30000	0,0x0000 B:00 00		N/A:NO CHANGE	UINT16 R/W	NO
		NO		SELECT FROM LIST		
Are the physical digital inputs locked for shotter/blind control ? =0:NO =1:YES						

The same lock mechanism is available for the MODBUS DIGITAL INPUTS:

LOCK:MODBUS INPUTS						
LOCK MODBUS INPUTS	3x30002 4x30002 I:30001	0,0x0000 B:00 00		1:YES	UINT16 R/W	YES
		NO		SELECT FROM LIST		
Are the MODBUS inputs locked for shotter/blind control ? =0:NO =1:YES						

20.2.7 MODBUS control and status of shade/blind outlets

Beside the possibility to control the engine outlet with push buttons on digital inputs or with virtual push buttons in the MODBUS DIGITAL INPUT MBDIx registers, you can control amnd monitor every outlet with the following registers:

BILIND & SHUTTER GROUP: Outputs DO1+DO2: CONTROL						
COMMAND	3x00101 4x00101 I:100	0,0x0000 B:00 00		400:DO REFERENCE MOVE	UINT16 R/W	YES
		NONE		SELECT FROM LIST		
New/next command for this blind / shutter						

This is the general command register. For some commands a shade/blind destination positon is used. Or in addition a destination slat position is required. So set up these registers first, before you write your command to the COMMAND register:

NEXT MOVE POSITION	3x00105 4x00105 I:104	0,0x0000 B:00 00	0		UINT16 R/W	NO
		00,00%		VALUE IN XX,XX%		
Next vertical position for this blind / shutter in percent 0%:complete open (upper position) 100%:complete closed (lower position)						
NEXT SLAT POSITION	3x00106 4x00106 I:105	0,0x0000 B:00 00	0		UINT16 R/W	NO
		00,00%		VALUE IN XX,XX%		
Next vertical position for the slats of this blind in percent 0%:in position SLAT ANGLE UP 100%:in position SLAT ANGLE DOWN						

The following commands are available:

0:NONE
100:MOVE TO 0%
101:MOVE TO 10%
102:MOVE TO 20%
103:MOVE TO 30%
104:MOVE TO 40%
105:MOVE TO 50%
106:MOVE TO 60%
107:MOVE TO 70%
108:MOVE TO 80%
109:MOVE TO 90%
110:MOVE TO 100%
120:MOVE TO POSITION
130:MOVE TO POSITION IN STEPS
131:MOVE STEP UP
132:MOVE STEP DOWN
200:MOVE TO SLAT 0%
201:MOVE TO SLAT 10%
202:MOVE TO SLAT 20%
203:MOVE TO SLAT 30%
204:MOVE TO SLAT 40%
205:MOVE TO SLAT 50%
206:MOVE TO SLAT 60%
207:MOVE TO SLAT 70%
208:MOVE TO SLAT 80%
209:MOVE TO SLAT 90%
210:MOVE TO SLAT 100%
220:MOVE TO SLAT POSITION
230:MOVE TO SLAT POSITION IN STEPS
231:MOVE SLAT STEP UP
232:MOVE SLAT STEP DOWN
300:POSITION
301:POSITION OVER ZERO
400:DO REFERENCE MOVE
999:ABORT
10000:INHIBIT MANUAL OPERATION
10001:ALLOW MANUAL OPERATION
10002:INHIBIT BUS OPERATION
10003:ALLOW BUS OPERATION
10004:DO MASTER UP
10005:DO MASTER DOWN
10006:DO MASTER INHIBIT ON
10007:DO MASTER INHIBIT OFF
10008:DO MASTER WIND ON
10009:DO MASTER WIND OFF
10010:DO MASTER RAIN ON
10011:DO MASTER RAIN OFF

If you want to abort a current running command, use this register:

ABORT	3x00109 4x00109 I:108	0,0x0000 B:00 00	0		UINT16 R/O	
		NO				
Abort request for current movement =0:NO =1:YES						

To monitor the current position of the shade/blind and to visualize the current position of the slats, read-out these registers:

CURRENT MOVE POSITION	3x00103 4x00103 I:102	0,0x0000 B:00 00			UINT16 R/O	
		00,00%				
Current vertical position for this blind / shutter in percent 0%:complete open (upper position) 100%:complete closed (lower position)						
CURRENT SLAT POSITION	3x00104 4x00104 I:103	0,0x0000 B:00 00			UINT16 R/O	
		00,00%				
Current position for the slat in percent 0%:in position SLAT ANGLE UP 100%:in position SLAT ANGLE DOWN						

To monitor the real digital outputs, read-out this register with the current state of the relays. This state affects also the invert state for the relay group:

REAL DOS	3x00108 4x00108 I:107	0,0x0000 B:00 00	0		UINT16 R/O	
		DO1=0 DO2=0				
Real state of the two digital outputs for the shutter/blind after possible inversion: Bit 0:DO1 (UP) Bit 1:DO2 (DOWN)						

If the MODE is set to TWO DOS, use this register to set the two relays to a specific state:
=0: Both relays are OFF
=1: DO1 is ON and DO2 is OFF
=2: DO1 is OFF and DO2 is ON
=3: DO1 is ON and DO2 is ON

DOS	3x00107 4x00107 I:106	0,0x0000 B:00 00	0		UINT16 R/W	NO
		DO1=0 DO2=0		BIT 0:DO1 (UP), BIT 1:DO2 (DOWN)		
State of the two digital outputs for the shutter/blind: Bit 0:DO1 (UP) Bit 1:DO2 (DOWN)						

There are some information registers to read for further infos:

ERROR	3x00110 4x00110 I:109	0,0x0000 B:00 00	0		UINT16 R/O	
		Error:0				
Current error code						

This is the current error code of the engine outlet. Only if this value is 0, the outlet will work. In any other case, no movement is done on the outlet.

IS REFERENCED	3x00111 4x00111 I:110	0,0x0000 B:00 00	0		UINT16 R/O	
		NO				
Is the shutter/blind currently referenced =0:NO =1:YES						

After a restart of the module the current position of the shade/blind and the slats are unknown. So every trigger of a movement down will trigger automatically a reference movement. or you start with the COMMAND DO REFERENCE.

The reference movement is a moving upwards for the defined time plus 20% reserve to securely reach always the upper position of the shade/blind.

The current state of the wind and rain alarm can be requested with the registers:

WIND ALARM STATE	3x00112 4x00112 I:111	0,0x0000 B:00 00	0		UINT16 R/O	
		NO				
Is wind alarm currently activated =0:NO =1:YES						
RAIN ALARM STATE	3x00113 4x00113 I:112	0,0x0000 B:00 00	0		UINT16 R/O	
		NO				
Is rain alarm currently activated =0:NO =1:YES						

In the status register you will see the current activity of the engine outlet.

STATUS	3x00102 4x00102 I:101	0,0x0000 B:00 00			UINT16 R/O	
		NO ACTION				
Current status for this blind / shutter						

The following status codes are used:
0:NO ACTION
1:MOVING DOWN
2:MOVING UP
3:REFERENCING,
4:MOVING DOWN IN STEPS
5:MOVING UP IN STEPS
6:MOVING SLATS DOWN
7:MOVING SLATS UP
8:WAITING
98:PARAM ERROR
99:ERROR

20.3 Technical specification

Beside the basic technical data, which fulfil all of our IO modules, this IO modules meet the following technical specifications:

Power consumption		
RESI-20RI8SB-SIO	<2.6W	
RESI-8SB-SIO	<2.6W	
RESI-10RI4SB-SIO	<1.3W	
RESI-4SB-SIO	<1.3W	

Product housing		
RESI-20RI8SB-SIO, RESI-8SB-SIO	BIG IO XT12	
RESI-10RI4SB-SIO, RESI-4SB-SIO	BIG IO XT8	

Product weight		
RESI-20RI8SB-SIO	480g	
RESI-8SB-SIO	458g	
RESI-10RI84B-SIO	323g	
RESI-4SB-SIO	310g	

Digital inputs		
only RESI-20RI8SB-SIO and RESI-10RI4SB-SIO		
Total amount of inputs	RESI-20RI8SB-SIO: 20	RESI-8SB-SIO: 0
	RESI-10RI4SB-SIO: 10	RESI-4SB-SIO: 0
Sampling rate	Every 5ms	

DC rating		
Input voltage range	12-250V~= +/-10%	
Input current	per channel	
	approx. 0.7mA@12V=	
	approx. 0.7mA@24V=	
	approx. 0.7mA@32V=	
	approx. 0.7mA@48V=	
	approx. 0.7mA@250V=	
Input power consumption	max. 0.2W/channel	
Logic levels	0: <4.5V=	
	1: >7.5V=	

AC rating		
Input voltage range	12-250V= +/-10%	
Input current	per channel	
	approx. 0.7mA@12V~	
	approx. 0.7mA@24V~	
	approx. 0.7mA@48V~	
	approx. 0.7mA@110V~	
	approx. 0.7mA@230V~	
	approx. 0.7mA@250V~	
Input power consumption	max. 0.2W/channel	
Logic levels	0: <4.5V~	
	1: >7.5V~	

Cable connection	
RESI-20RI8SB-SIO	Via 2-pin plug-in terminal block for common signal
	and two 8-pin plug in terminal blocks for DI1..DI16
	and one 4-pin plug in terminal block for DI17-DI20
RESI-10RI4SB-SIO	Via 2-pin plug-in terminal block for common signal
	and one 8-pin plug in terminal blocks for DI1..DI8
	and one 2-pin plug in terminal block for DI9-DI10
Terminal type	RM3.5
Galvanic insulation	Yes, all digital inputs as group to IO module
Relay outputs	
Number of outputs	
RESI-20RI8SB-SIO, RESI-8SB-SIO	16 mono stable relays
RESI-10RI4SB-SIO, RESI-4SB-SIO	8 mono stable relays
	organized as group with common power supply
Relay type	Mono stable
Maximum voltage	250Vac
Maximum current	6A
Mechanical lifetime	10^6 cycles of operation
Contact material	$AgSnO_2$
Max. switching power AC1	1500VA
Max. switching power AC15 (230V~)	300VA
Max. switching power AC3	185W
Max. switching power DC1	6A@30V=
	0.2A@110V=
	0.12A@220V=
Insulation	Creepage and clearance distance 8mm
Cable connection	Power supply: via one 2-pin plug in terminia block
RESI-20RI8SB-SIO, RESI-8SB-SIO	shade/blind outlet: via eight 3-pin plug-in terminal blocks
RESI-10RI4SB-SIO, RESI-4SB-SIO	shade/blind outlet: via four 3-pin plug-in terminal blocks
Terminal type	RM3.5
Galvanic insulation	Yes, with the relay
Default serial settings	
Baud rate	via DIP switch
Parity	none
Stop bit(s)	one
UnitID	255

20.4 Additional terminals & LED states

DIGITAL INPUTS		
RESI-20RI8SB-SIO	20 digital inputs for 12-250Vac/dc signals	
	C:	GND or neutral connector
	DI1-DI20:	Digital inputs
		0=open or GND,
		1=12Vac/dc..250Vac/dc
	One 2-pin plug-in terminal block for common ground or neutral	
	Terminal type: RM3.5	
	Pin1:C:	M-,N: Ground/neutral for all digital inputs
	Pin 2:C:	M-,N: Ground/neutral for all digital inputs
	One 8 pin plug-in terminal block for digital inputs DI1..DI8	
	Terminal type: RM3.5	
	Pin 1:DI1:IN 1:↑	First digital input for shade/blind group #1-up
	Pin 2:DI2:IN 1:↓	Second digital input for shade/blind group #1-down
	Pin 3:DI3:IN 2:↑	First digital input for shade/blind group #2-up
	Pin 4:DI4:IN 2:↓	Second digital input for shade/blind group #2-down
	Pin 5:DI5:IN 3:↑	First digital input for shade/blind group #3-up
	Pin 6:DI6:IN 3:↓	Second digital input for shade/blind group #3-down
	Pin 7:DI7:IN 4:↑	First digital input for shade/blind group #4-up
	Pin 8:DI8:IN 4:↓	Second digital input for shade/blind group #4-down
	One 8 pin plug-in terminal block for digital inputs DI9..DI16	
	Terminal type: RM3.5	
	Pin 1:DI9:IN 5:↑	First digital input for shade/blind group #5-up
	Pin 2:DI10:IN 5:↓	Second digital input for shade/blind group #5-down
	Pin 3:DI11:IN 6:↑	First digital input for shade/blind group #6-up
	Pin 4:DI12:IN 6:↓	Second digital input for shade/blind group #6-down
	Pin 5:DI13:IN 7:↑	First digital input for shade/blind group #7-up
	Pin 6:DI14:IN 7:↓	Second digital input for shade/blind group #7-down
	Pin 7:DI15:IN 8:↑	First digital input for shade/blind group #8-up
	Pin 8:DI16:IN 8:↓	Second digital input for shade/blind group #8-down
	One 4 pin plug-in terminal block for digital inputs DI17..DI20	
	Terminal type: RM3.5	
	Pin 1:DI17:INPUTS:I1	First digital input for wind/rain alarm or group functions
	Pin 2:DI18:INPUTS:I2	Second digital input for wind/rain alarm or group functions
	Pin 3:DI19:INPUTS:I3	Third digital input for wind/rain alarm or group functions
	Pin 4:DI20:INPUTS:I4	Fourth digital input for wind/rain alarm or group functions

DIGITAL INPUTS

RESI-10RI4SB-SIO	10 digital inputs for 12-250Vac/dc signals	
	C:	GND or neutral connector
	DI1-DI10:	Digital inputs
		0=open or GND,
		1=12Vac/dc..250Vac/dc
	One 2-pin plug-in terminal block for common ground or neutral	
	Terminal type: RM3.5	
	Pin1:C:	M-,N: Ground/neutral for all digital inputs
	Pin 2:C:	M-,N: Ground/neutral for all digital inputs
	One 8 pin plug-in terminal block for digital inputs DI1..DI8	
	Terminal type: RM3.5	
	Pin 1:DI1:IN 1:↑	First digital input for shade/blind group #1-up
	Pin 2:DI2:IN 1:↓	Second digital input for shade/blind group #1-down
	Pin 3:DI3:IN 2:↑	First digital input for shade/blind group #2-up
	Pin 4:DI4:IN 2:↓	Second digital input for shade/blind group #2-down
	Pin 5:DI5:IN 3:↑	First digital input for shade/blind group #3-up
	Pin 6:DI6:IN 3:↓	Second digital input for shade/blind group #3-down
	Pin 7:DI7:IN 4:↑	First digital input for shade/blind group #4-up
	Pin 8:DI8:IN 4:↓	Second digital input for shade/blind group #4-down
	One 2 pin plug-in terminal block for digital inputs DI17..DI20	
	Terminal type: RM3.5	
	Pin 1:DI17:INPUTS:I1	First digital input for wind/rain alarm or group functions
	Pin 2:DI18:INPUTS:I2	Second digital input for wind/rain alarm or group functions

RELAY OUTPUTS		
RESI-20RI8SB-SIO, RESI-8SB-SIO		
	One 2 pin plug-in terminal blocks for power supply of all 8 shade/blind outlets	
	Terminal type:	RM5
	Pin 1:PWR:L	L+/L: DC or AC power supply 12-250Vac or 12-30Vdc
	Pin 2: PWR:N	M-/N: DC or AC power supply AC neutral or DC ground
	Eight 3 pin plug-in terminal blocks for shade/blind outlets (x=1..8)	
	Terminal type:	RM3.5
	Pin 1:OUTx:↑	Switched relay output for shade/blind moving up
	Pin 2:N	M-/N: AC neutral or DC ground from power supply
	Pin 3:OUTx:↓	Switched relay output for shade/blind moving down
RELAY OUTPUTS		
RESI-10RI4SB-SIO, RESI-4SB-SIO		
	One 2 pin plug-in terminal blocks for power supply of all 8 shade/blind outlets	
	Terminal type:	RM5
	Pin 1:PWR:L	L+/L: DC or AC power supply 12-250Vac or 12-30Vdc
	Pin 2: PWR:N	M-/N: DC or AC power supply AC neutral or DC ground
	Four 3 pin plug-in terminal blocks for shade/blind outlets (x=1..4)	
	Terminal type:	RM3.5
	Pin 1:OUTx:↑	Switched relay output for shade/blind moving up
	Pin 2:N	M-/N: AC neutral or DC ground from power supply
	Pin 3:OUTx:↓	Switched relay output for shade/blind moving down

20.5 Connection diagram

20.5.1 Cabling of the digital inputs

Only for RESI-20RI8SB-SIO and RESI-10RI4SB-SIO: In the below drawing you see the cabling of the digital inputs of the module. You can use the digital inputs either for DC signals or for AC signals.
DC power supply L+/M- in the range of 12-48V=.
AC power supply L/N in the range from 12-250V~.
Internal digital 50/60Hz AC software filter for all digital inputs.

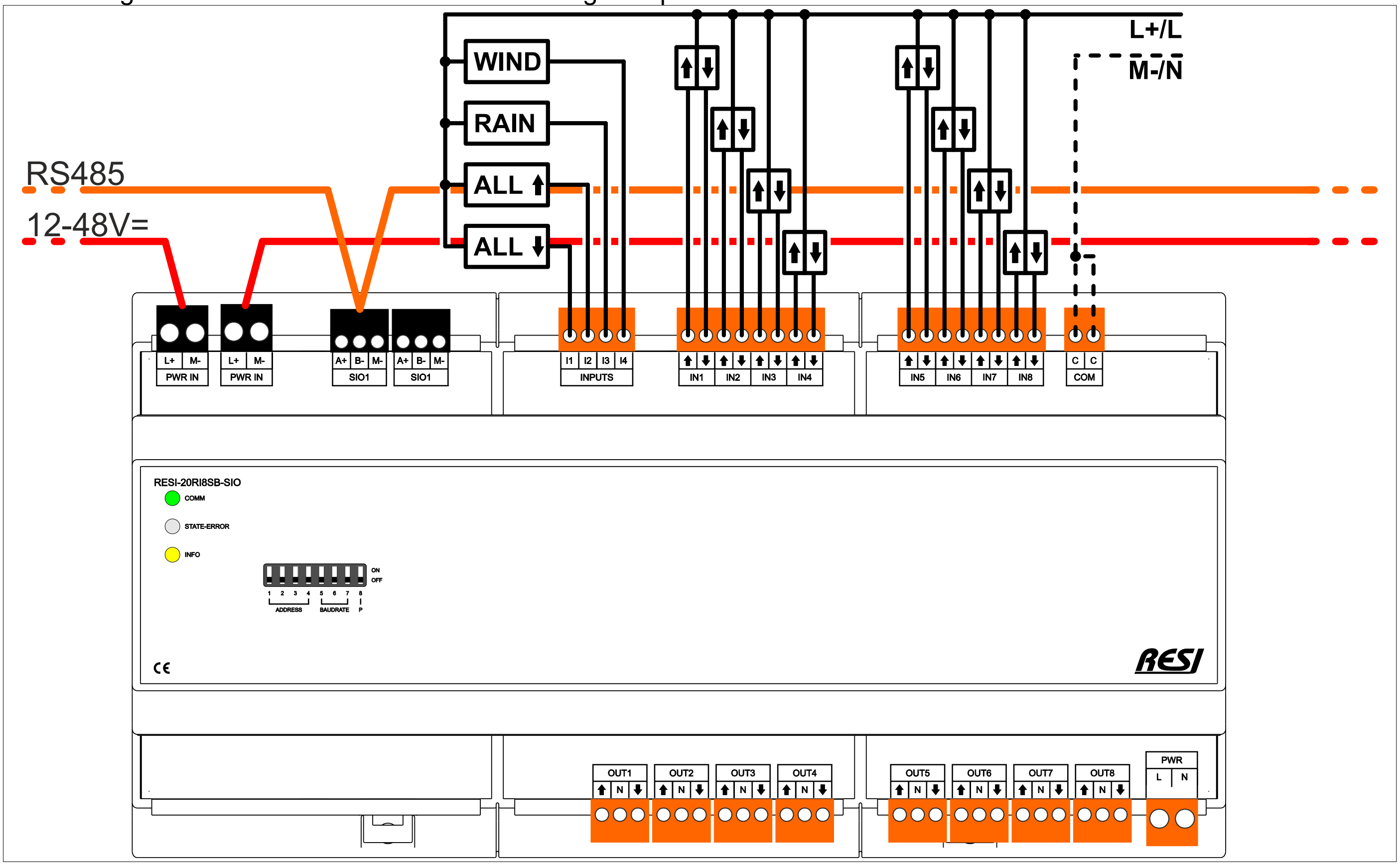


Figure: Connecting the digital inputs to the RESI-20RI8SB-SIO IO module

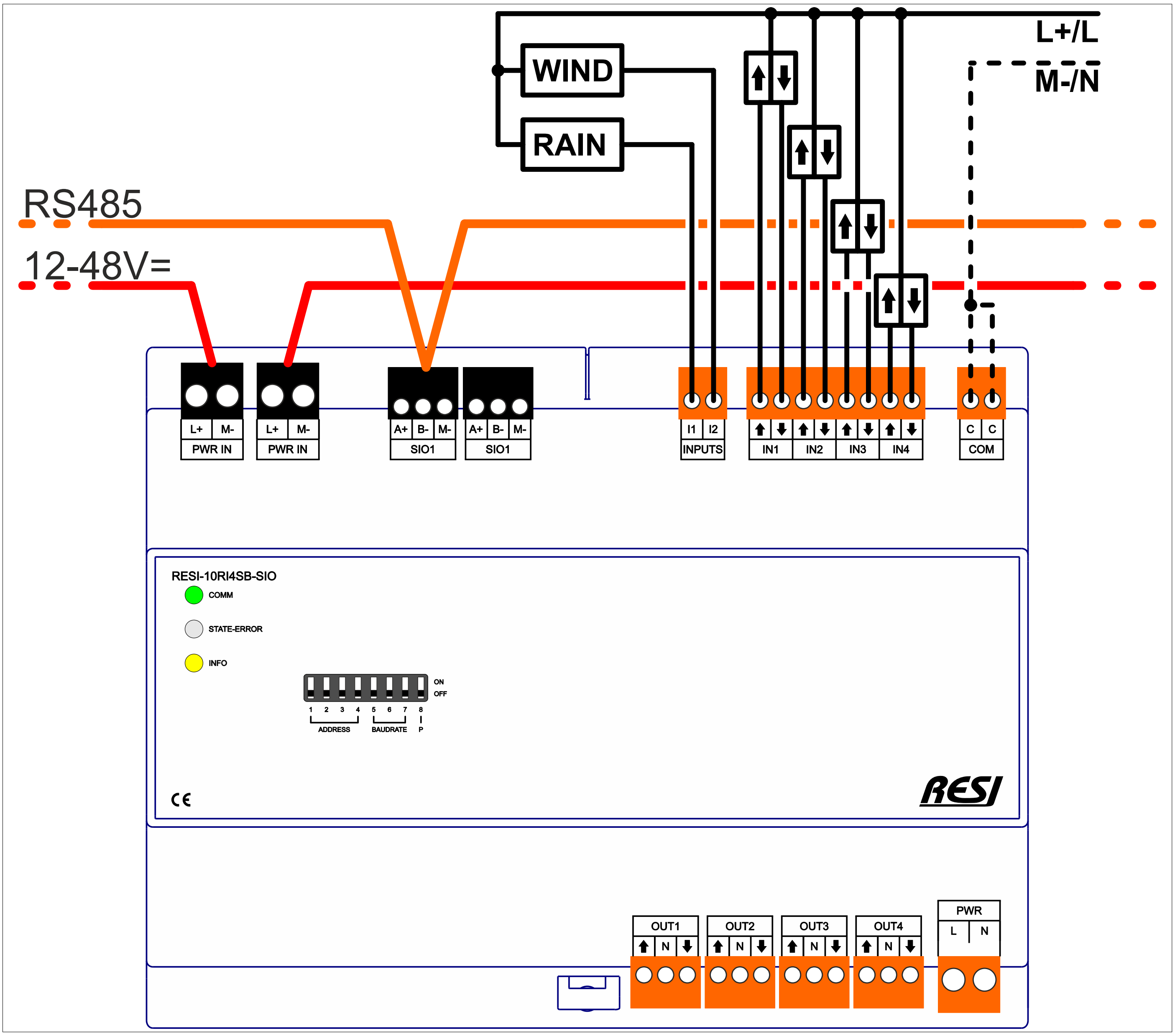


Figure: Connecting the digital inputs to the RESI-10RI4SB-SIO IO module

20.5.2 Cabling of the relay outputs

In the below drawing the cabling of the relay outputs is shown:

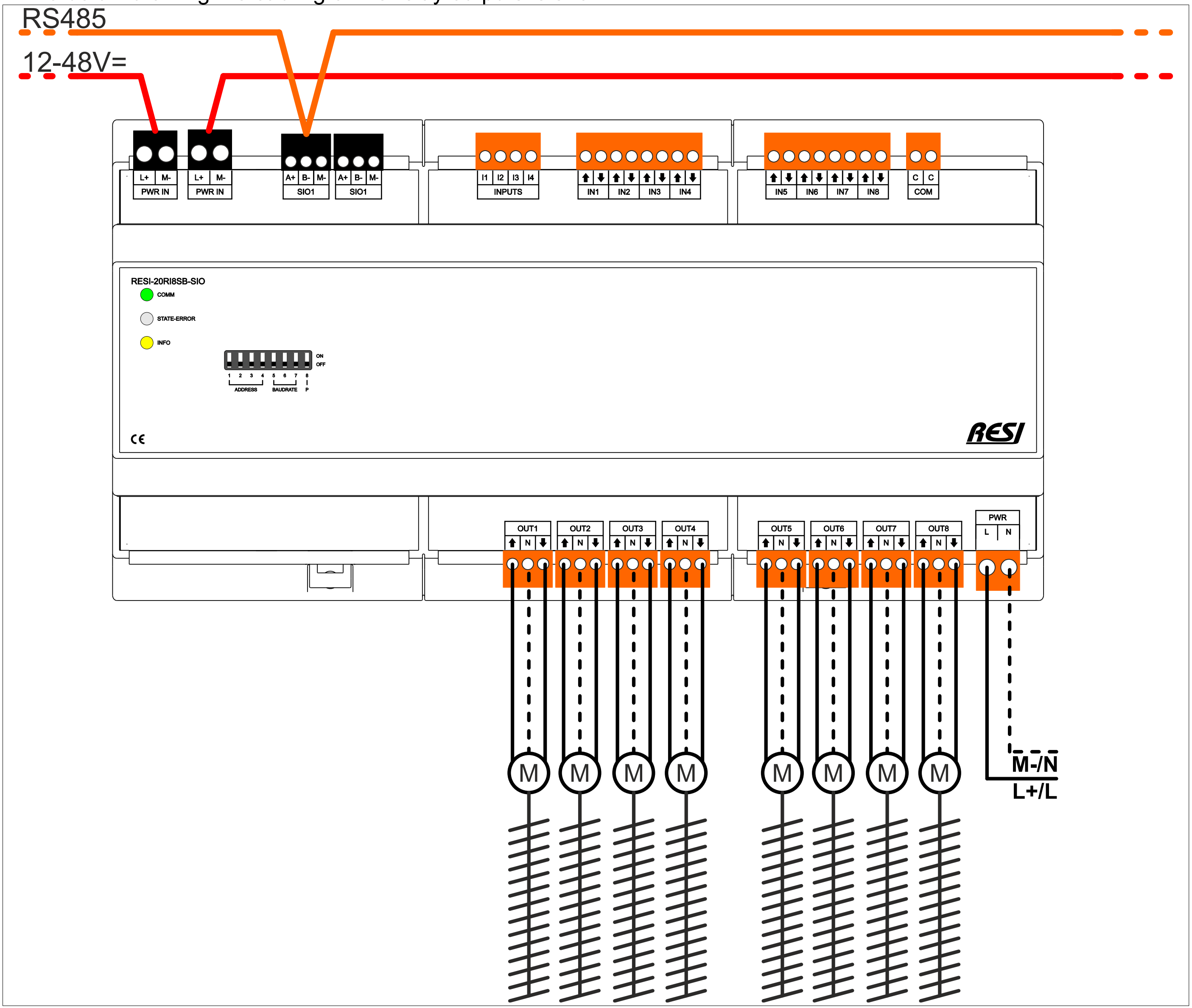


Figure: Connecting the shades/blinds to a RESI-20RI8SB-SIO module

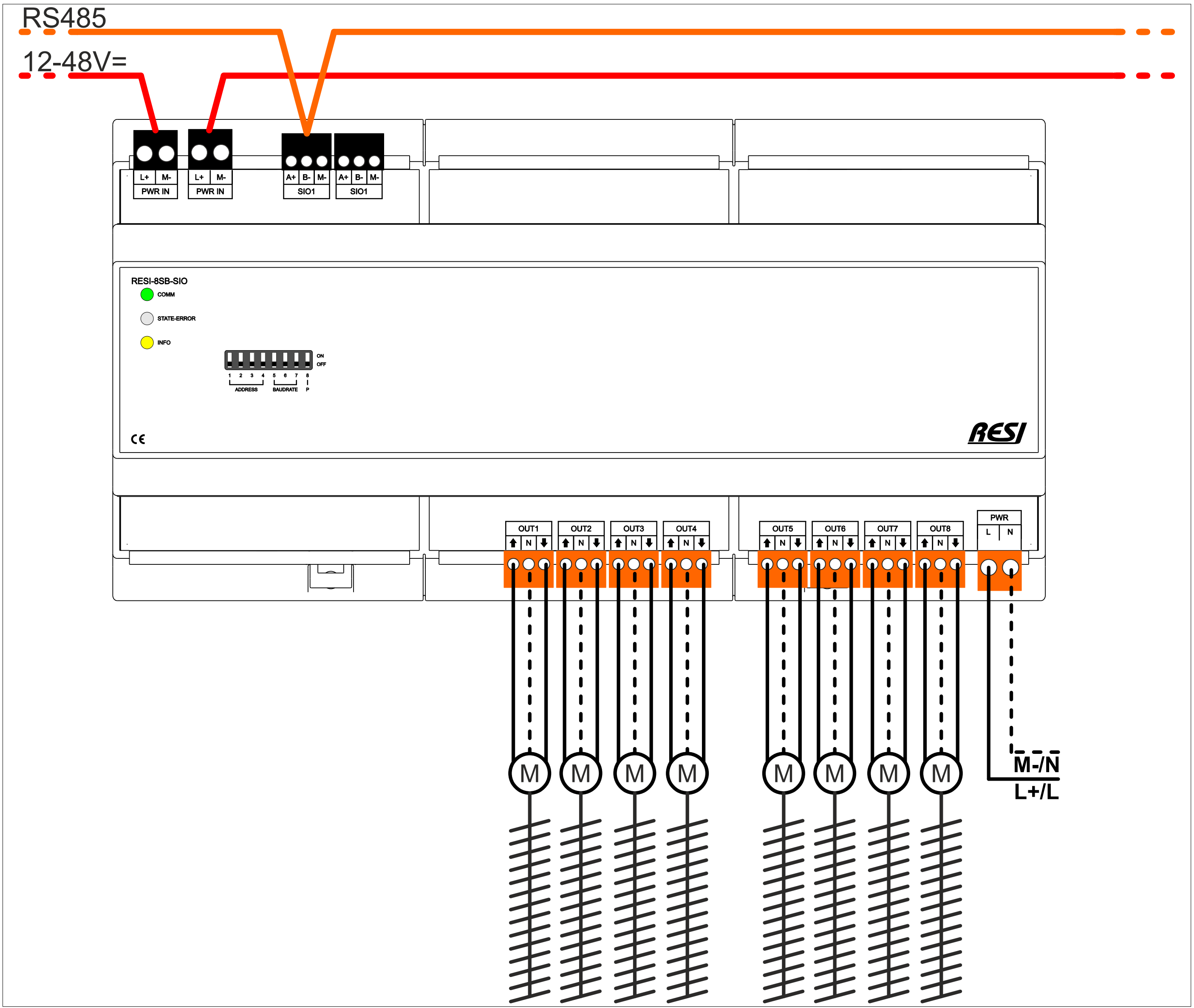


Figure: Connecting the shades/blinds to a RESI-8SB-SIO module

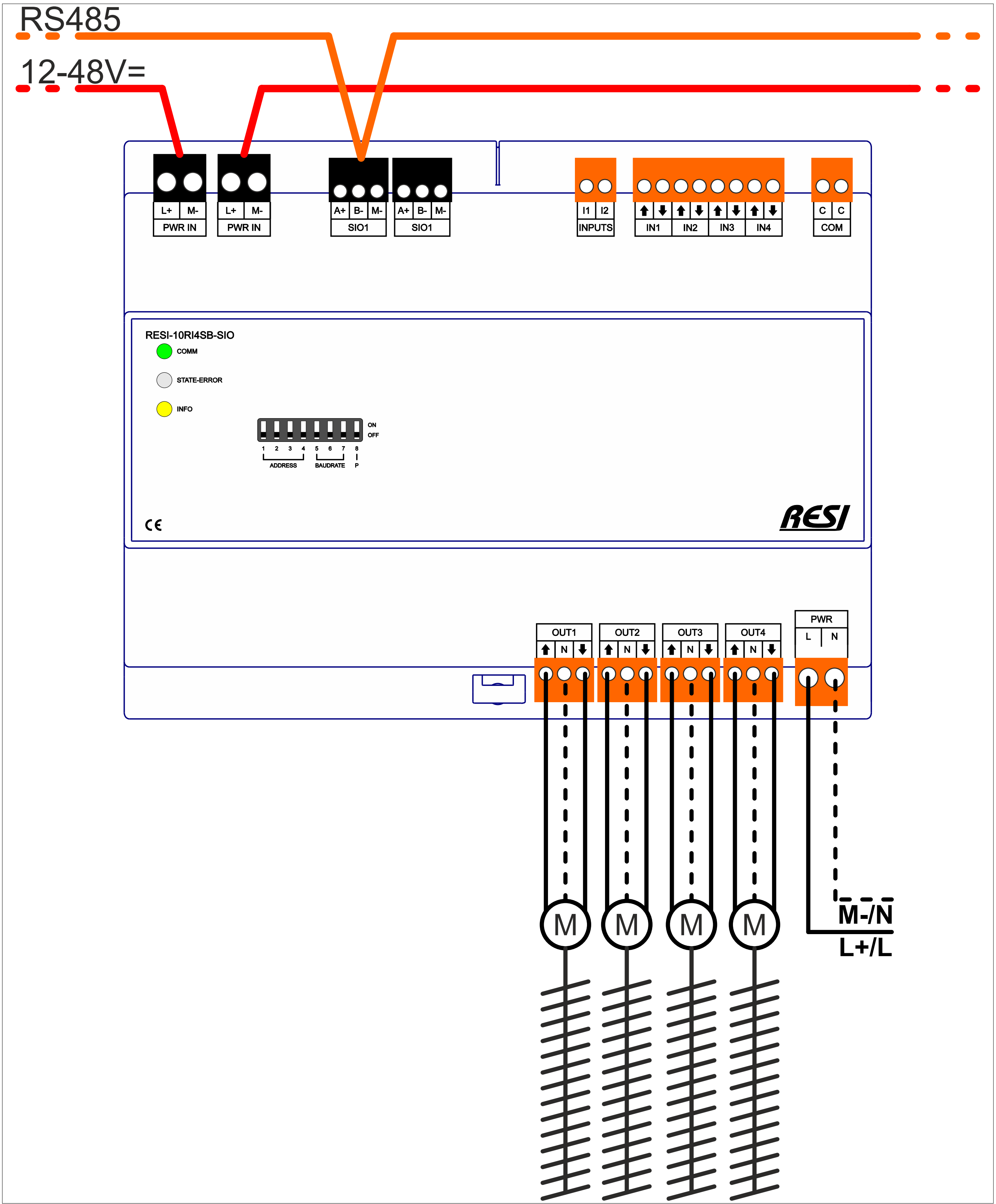


Figure: Connecting the shades/blinds to a RESI-10RI4SB-SIO module

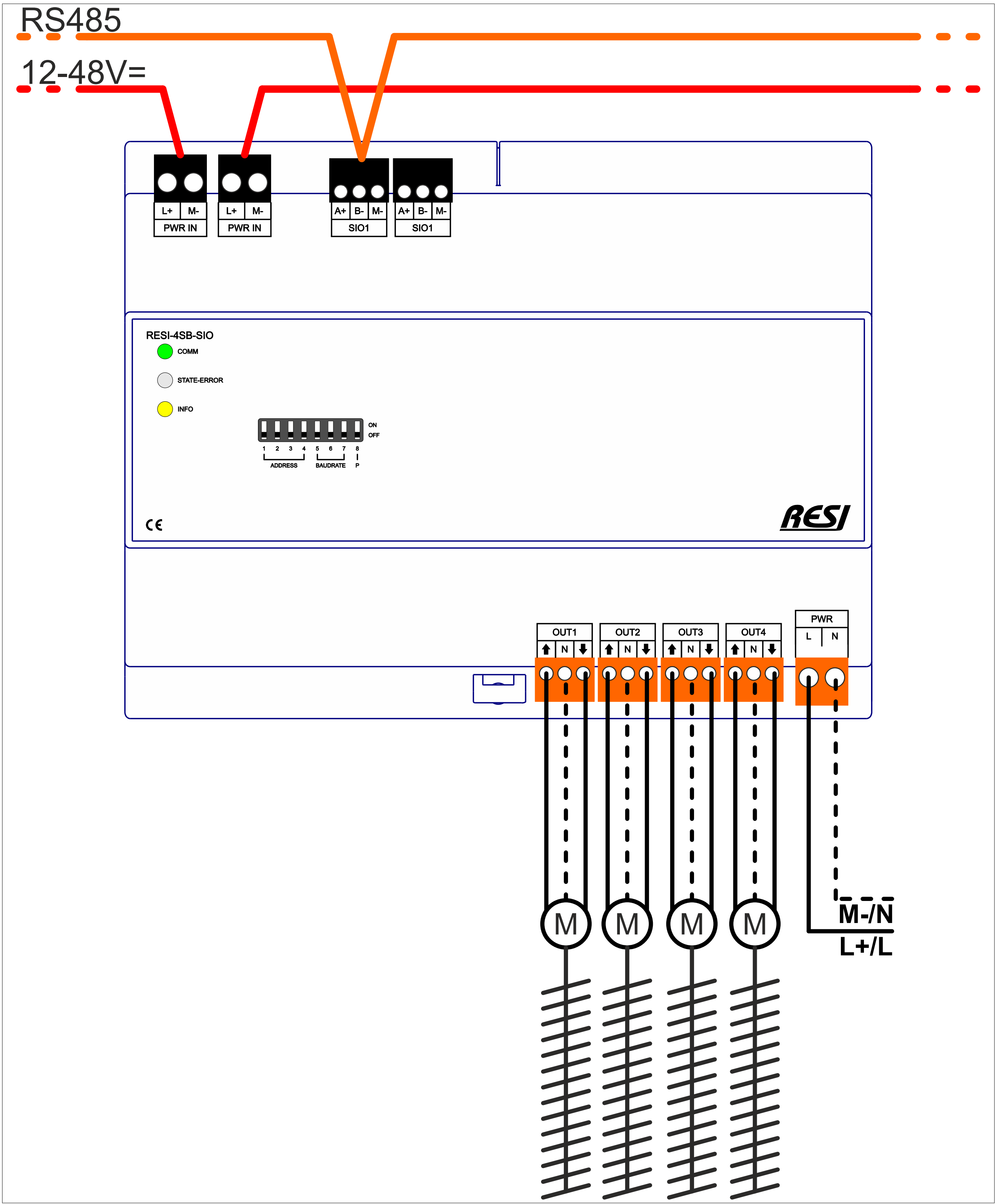


Figure: Connecting the shades/blinds to a RESI-4SB-SIO module

20.6 Additional MODBUS register & coils

Here you will find only the additional MODBUS registers and coils especially for this IO module. Please refer to the description of of the standard MODBUS mapping for more details about the available basic MODBUS registers and coils.

Please refer to the external document for detailed documentation of the current MODBUS register mapping for this IO module:

RESI-L-20RI8SB,8SB,10RI4SB,4SB-SIO-MODBUS+ASCII-ENxx.pdf

20.7 Additional ASCII commands

Here you will find only the additional ASCII commands especially for this IO module. Please refer to the description of of the standard commands for more details about the available basic ASCII commands.

Please refer to the external document for detailed documentation of the current ASCII commands for this IO module:

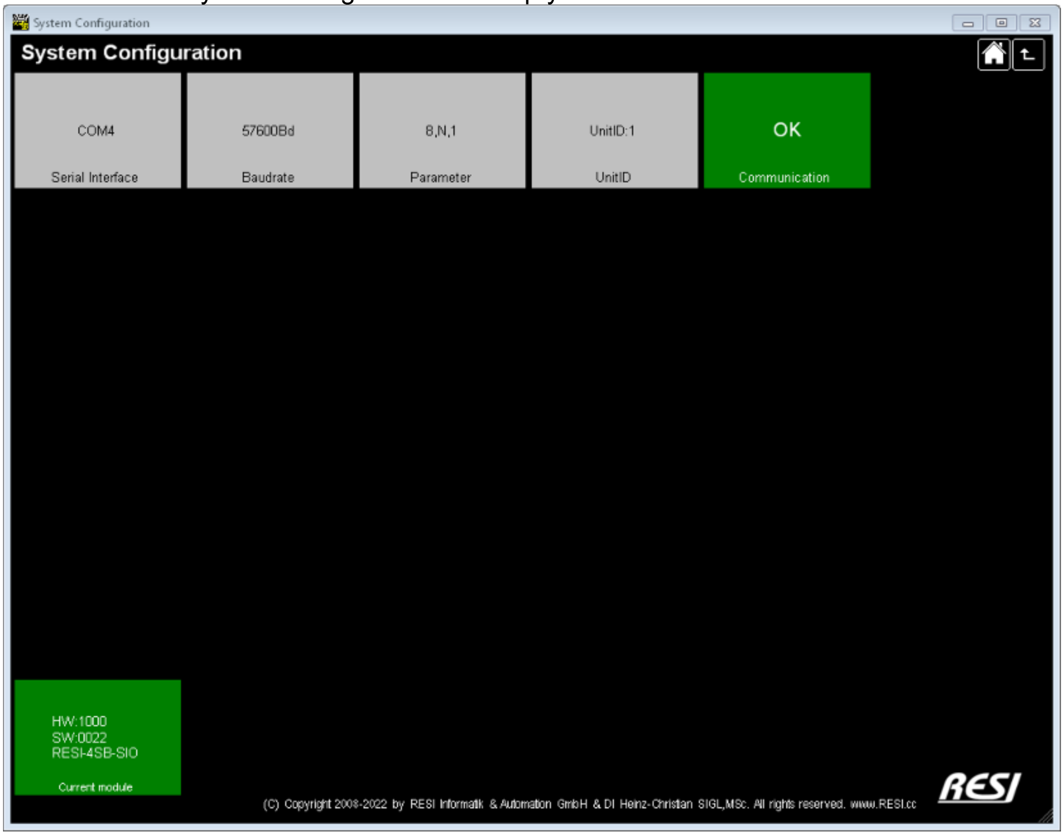
RESI-L-20RI8SB,8SB,10RI4SB,4SB-SIO-MODBUS+ASCII-ENxx.pdf

20.8 RESI configuration & test software

RESI offers a simple configuration & test software based on Windows. Simple unzip the contents into a folder and start in this folder the application RESI_SCADA_2D.exe.

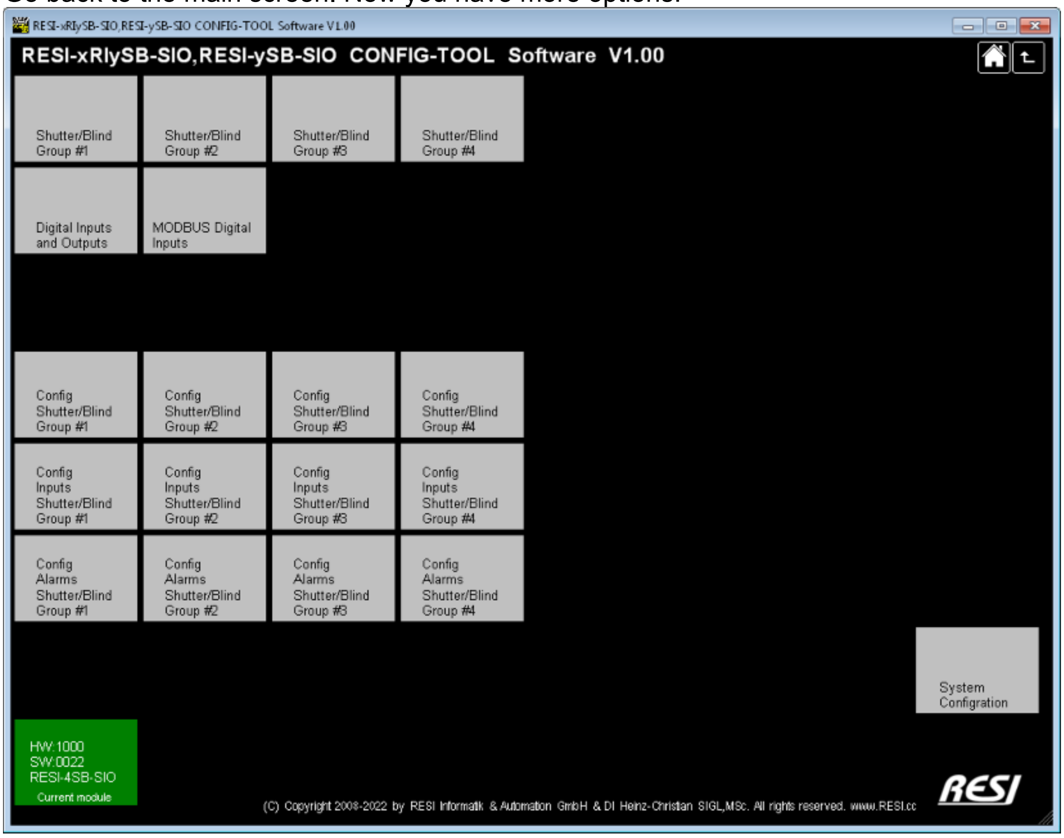
After a few seconds you will see the following screen:

Now click onto System Configuration to setup your serial connection:

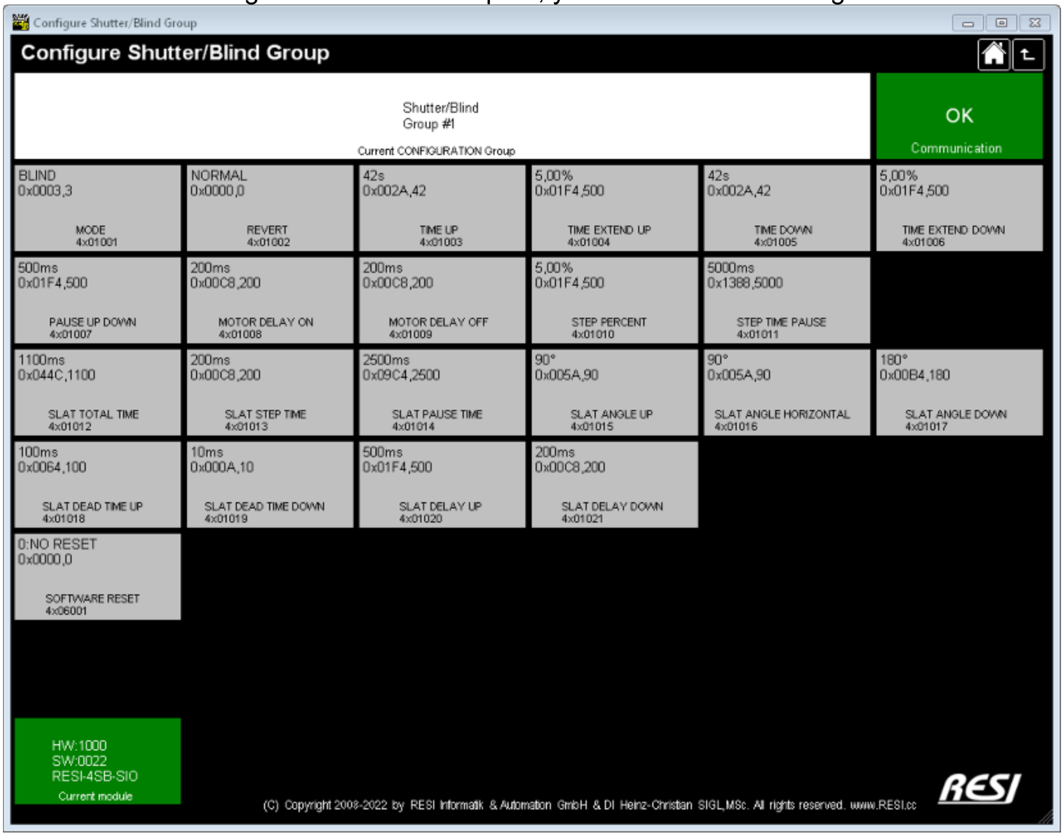


Adjust the serial interface to your RS485 connection COMx and setup the communication parameters according your DIP switch settings on your IO module. Don#t forget to define the correct UnitID (Bus Number) for the communication. If everything is ok, you will see the green OK filed and in the left corner the actual connected IO module.

Go back to the main screen. Now you have more options:

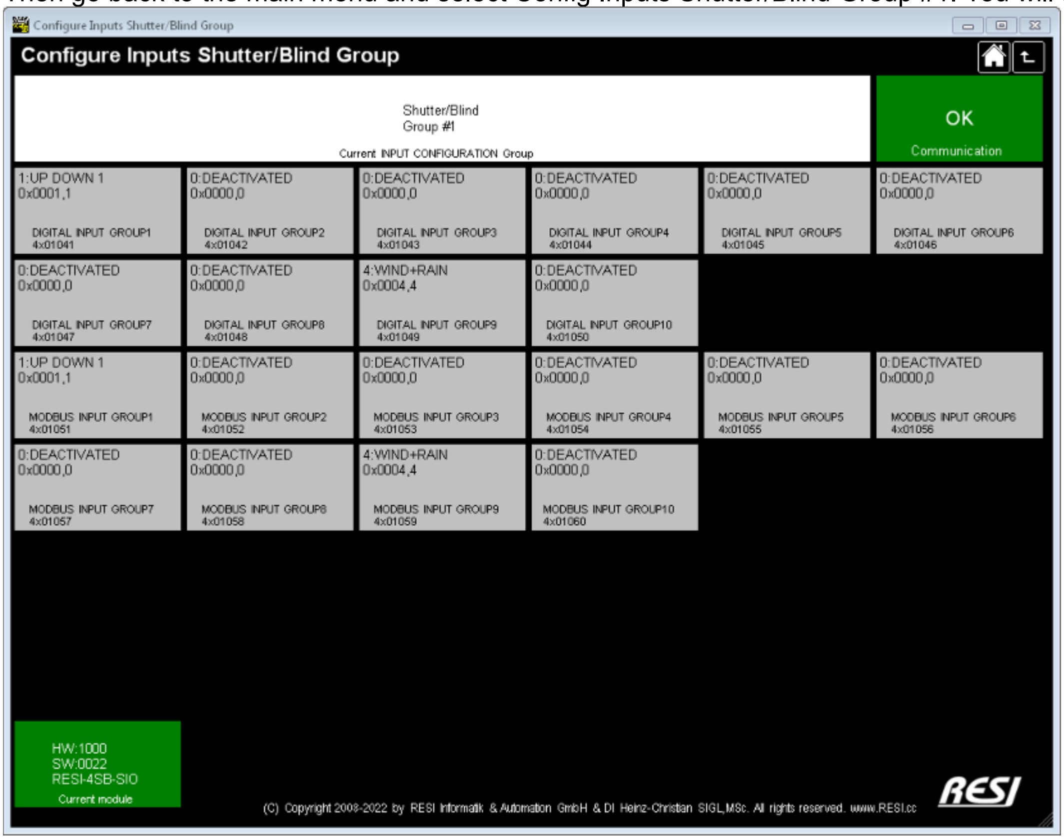


Now click onto Config Shutter/Blind Group #1, you will see the following screen:



Here you see all actual configured values for the first engine outlet OUT1. Simply click onto the fields to change the current settings to your desired values. Please consult the previous chapter for more detailed information about the contents of the individual registers. Don't forget to restart your module after you have changed the settings. use SOFTWARE RESET for this or a power off and on cycle. Execute this setup and configuration for the other groups as well!

Then go back to the main menu and select Config Inputs Shutter/Blind Group #1. You will see the following picture:

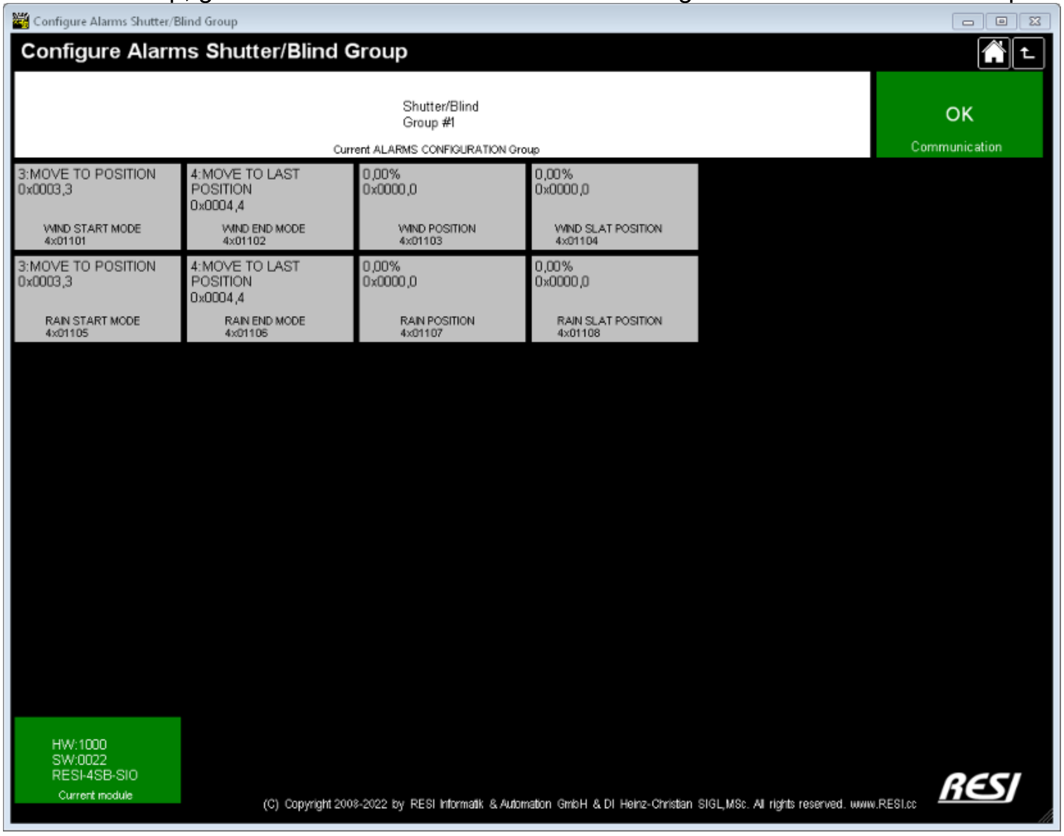


Here you define which digital inputs will trigger a movement on the selected engine outlet. Also you can define which digital inputs will trigger a wind and rain alarm.

In the same setup window you can define which of the defined MODBUS digital inputs will trigger a movement of the shade/blind outlet.

HINT: If you configure for more than one engine outlet the same DIGITAL INPUT GROUPs all the defined engines will start their movement at the same time.
If you configure for more than one engine outlet the same MODBUS INPUT GROUPs all the defined engines will start their movement at the same time.

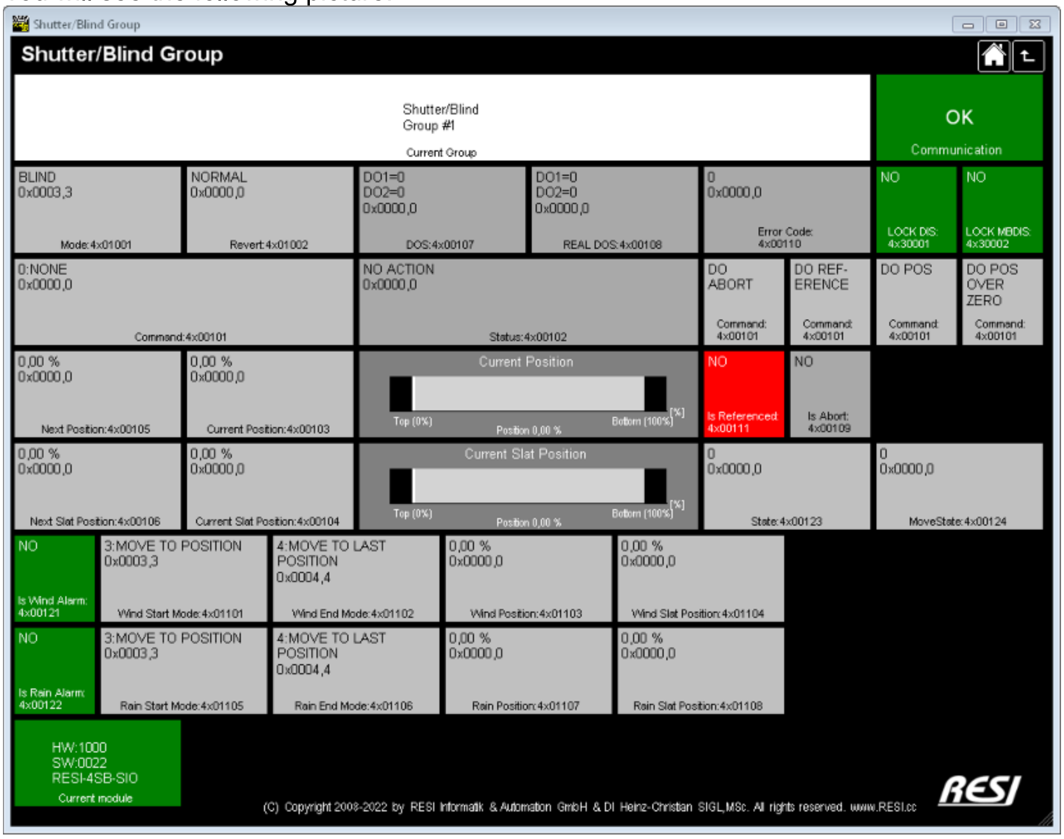
After that setup, go back to main menu and click on Config Alarms Shutter/Blind Group #1:



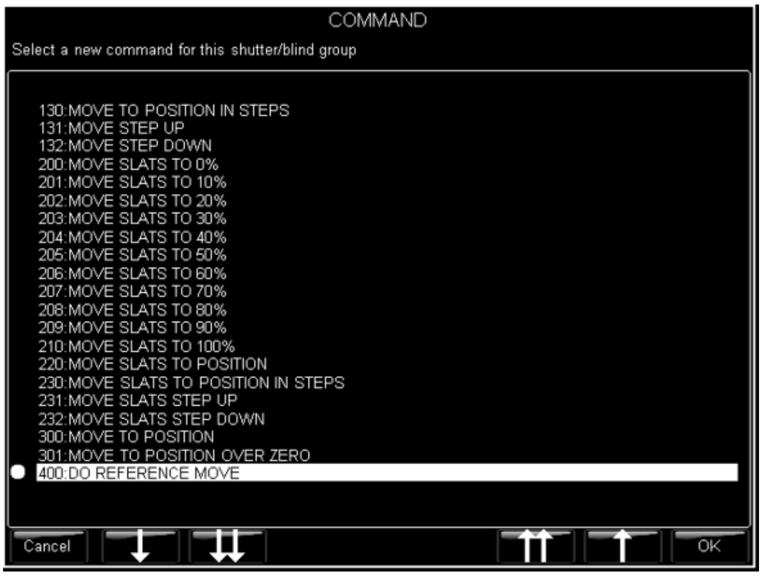
Here you define the behaviour in case of a wind or rain alarm. This alarm is either triggered by the selected digital inputs or MODBUS inputs or is triggered by sending a special command to the engine outlet.

Repeat this setup for all groups of your IO module.

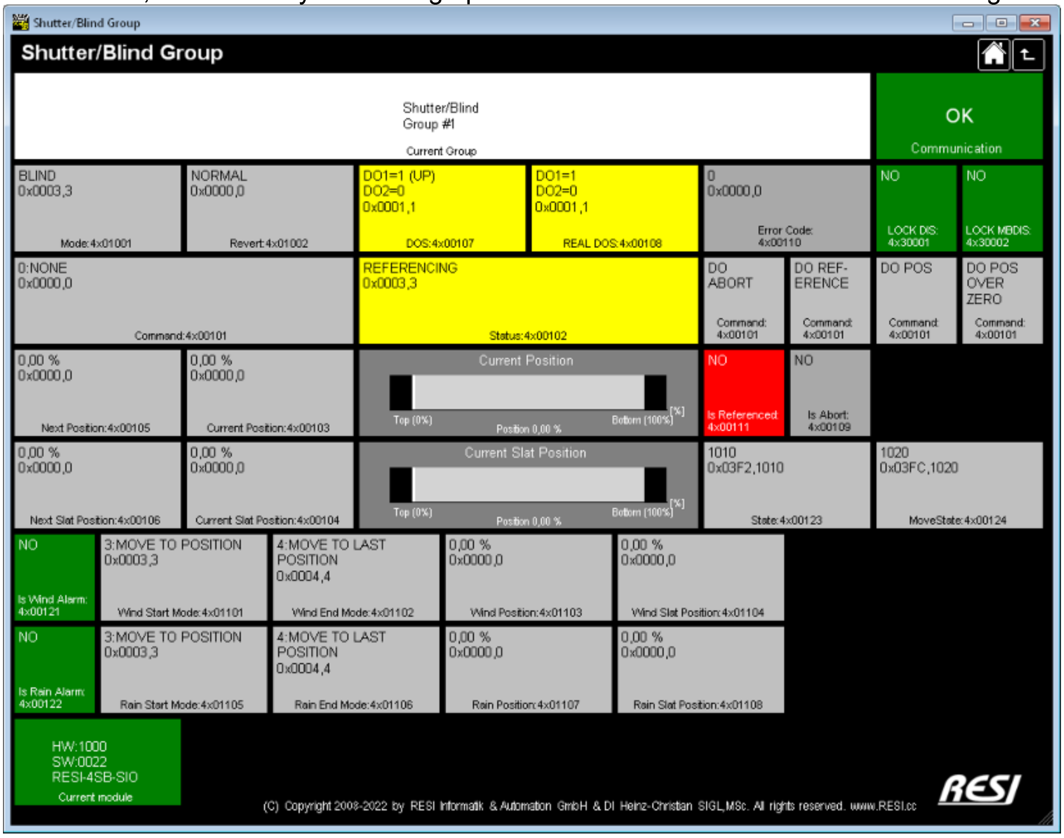
After that go back to the main menu to test a certain shutter/blind. In our case we click onto Shutter/Blind Group #1. You will see the following picture:



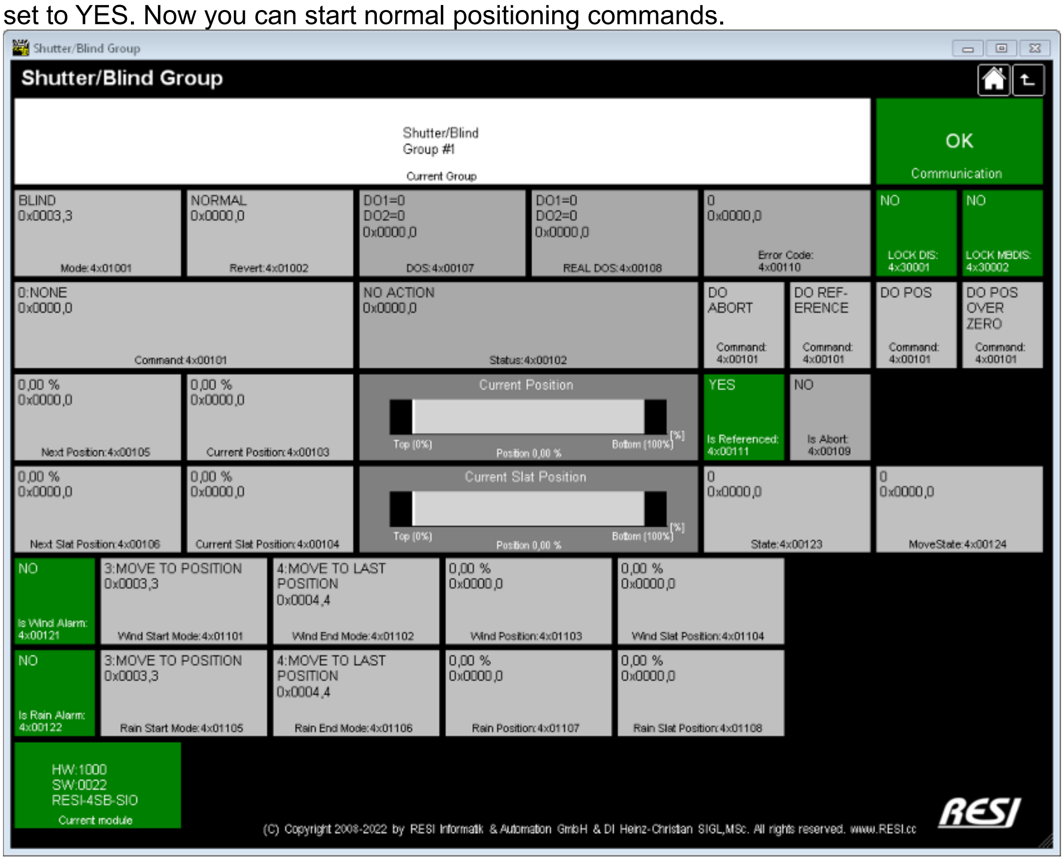
In the first row you see the current MODE and REVERT settings for easy check. To test your device, the fields COMMAND, ERROR CODE and STATUS is of interest. Also note that in our case the shutter/blind is not referenced up to now. So we click onto the file COMMAND, scroll down in the list and select the command:
400:DO REFERENCE MOVE and click onto OK.



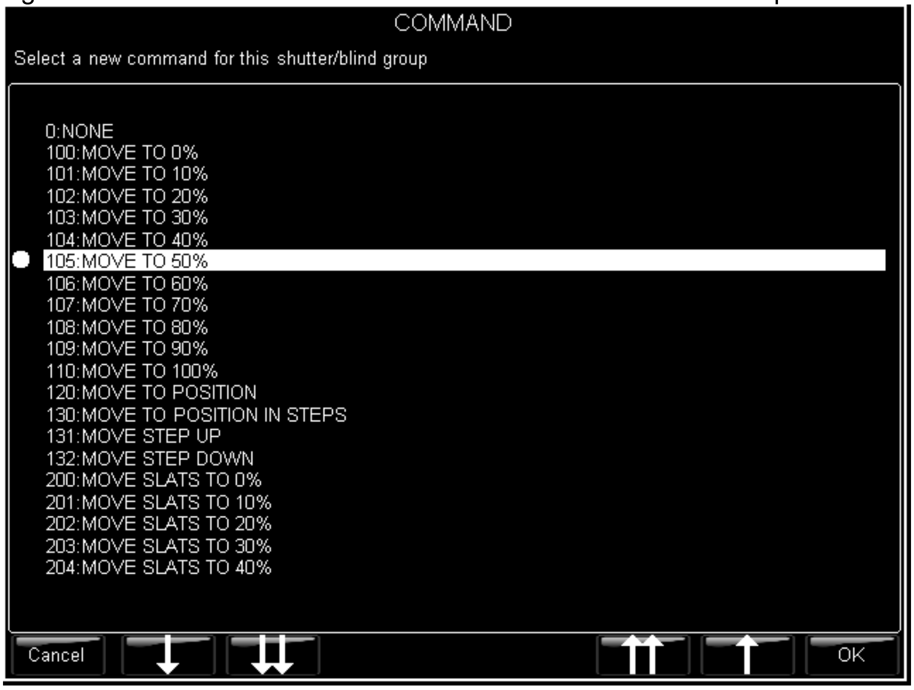
You will see, that the relay for moving upwards will be activated and the status changes to REFERENCING:



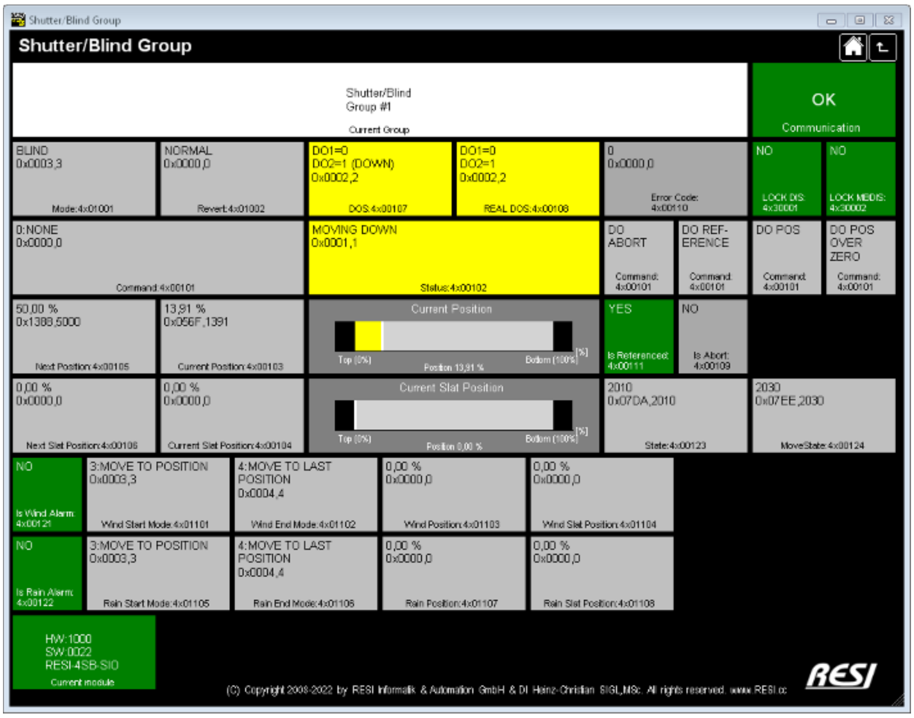
Now wait for the end of the reference phase, you now see the following picture. Note that the flag IS REFERENCE is set to YES. Now you can start normal positioning commands.



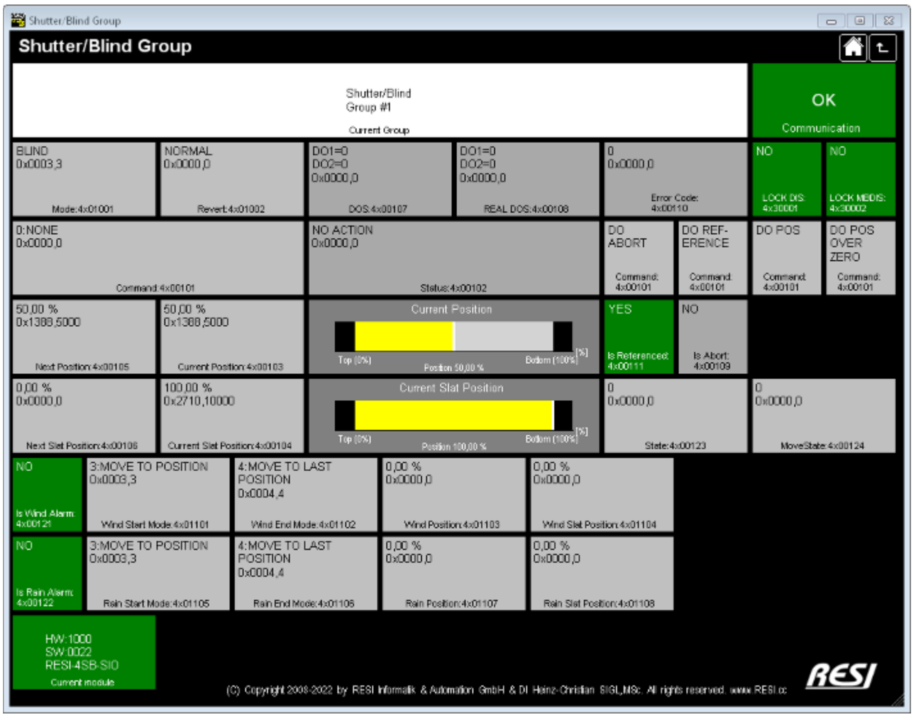
Again click on the field COMMAND and select the command and press OK:



While the shutter/blind is moving you will notice, that the current position is updated and the status reflects the current movement.



After reaching the end position the slat position is updated also according to the current action.

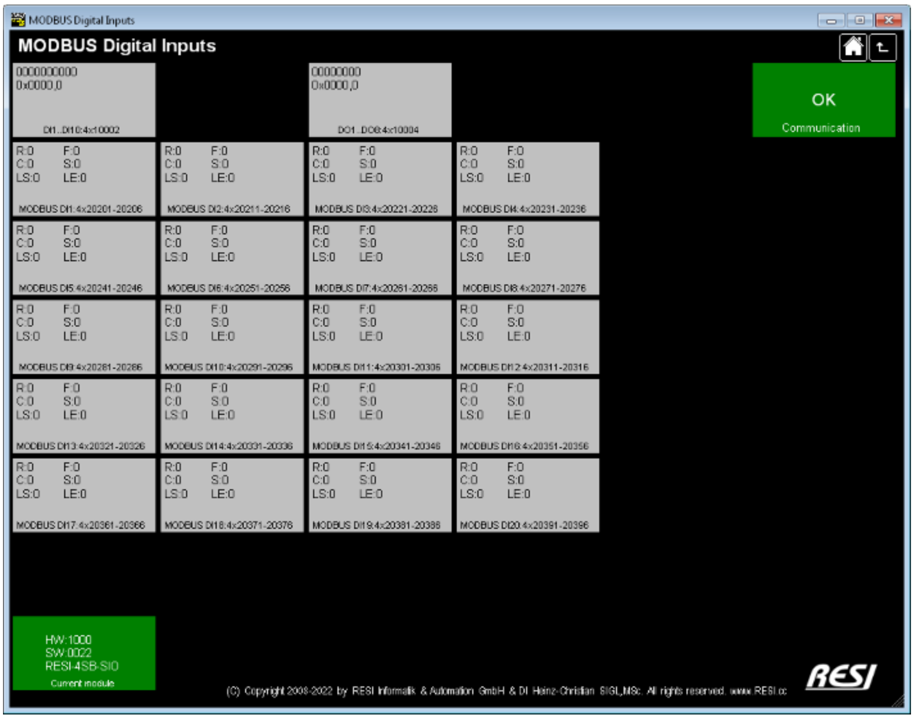


So you can test the complete behaviour on this plage for every engine outlet.

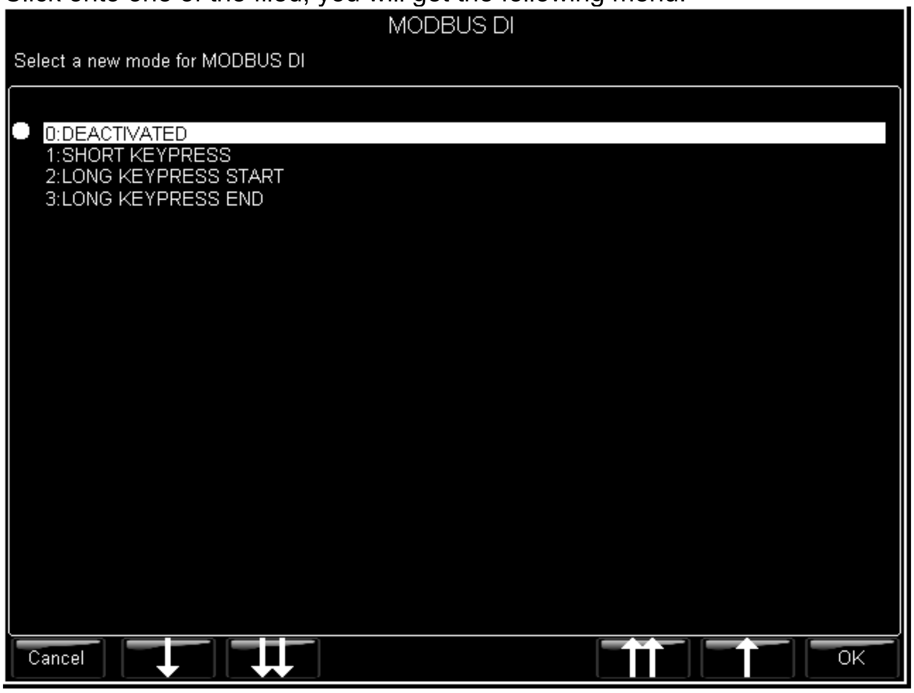
Now go back to the main menu and open the page Digital Inputs and Outputs.
You will see the following picture:

In the upper region you see the current status of the digital inputs and outputs.
Below you see for every digital input the current counters.
R: stands for the amount of counted rising edges
F: stands for the amount of counted falling edges
C: stands for the amount of counted total events
S: stands for the amount of counted short key press events
LS: stands for the amount of counted long key press start events
LE: stands for the amount of counted long key press end events

Due to the group configuration the digital input will trigger a movement on the engine outlet.
Go back to the main screen and select the page MODBUS digital inputs:



Here you see again the current status of the digital inputs and outputs. But you see also the current status of all simulated MODBUS digital inputs.
Click onto one of the filed, you will get the following menu:



You see, that writing on the specific MODBUS register simulates a certain key press event like on the digital inputs.
If you have associated the MODBUS input to a engien outlet e.g. a writing of 1 for a short key press will start a movement of the shutter/blind. A second writing on the same register will stop the current movement.

Below you see for every MODBUS digital input the current counters.
R: stands for the amount of counted rising edges
F: stands for the amount of counted falling edges
C: stands for the amount of counted total events
S: stands for the amount of counted short key press events
LS: stands for the amount of counted long key press start events

LE: stands for the amount of counted long key press end events

So in the simplest control software you have to write to two consecutive MODBUS digital input registers to control the shutter/blind. An example for that:

We have configured MODBUS INPUT GROUP#1 to SHUTTER/BLIND GROUP #1 with the function 1:UP DOWN 1 in the page CONFIG SHUTTER/BLIND GROUP #1.

Then on the page MODBUS Digital Inputs the two inputs MODBUS DI1 and MODBUS DI2 control the first shutter/blind outlet OUT1 as follows:
A. Write 1(SHORT KEY PRESS) to register MODBUS DI2
B. Shutter/Blind OUT1 will start moving downward
C. Write 1(SHORT KEY PRESS) to register MODBUS DI1
D. Shutter/Blind OUT1 will stop current movement
E. Write 1(SHORT KEY PRESS) to register MODBUS DI1
F. Shutter/Blind OUT1 will start moving upward
G. Write 2(LONG KEY PRESS START) to register MODBUS DI2
H. Slat will step downward
I. Write e(LONG KEY PRESS END) to register MODBUS DI2
J. Slat will stop
K. Write 1(SHORT KEY PRESS) to register MODBUS DI2
L. Shutter/Blind OUT1 will start moving downward
M. Wait until the downward movement is stopped due to the end of the shutter/blind
N. Now the final position 100% is reached and the slats are fully closed 100%

21 RESI-4LED-SIO

21.1 General information

This series of IO modules offer the following features:

- 12 dimmable PWM output channels for LED stripes, 0..48Vdc, max. 5A each channel, organized in 4 groups with three channels A,B and C each
- Each LED group offers six selectable modes: OFF, ON, FLASHING, FADING, RANDOM, SEQUENCE
- External power supply for LED stripes, 0..48Vdc, max. 15A
- DIP switch for setting the baud rates, the type of interface and the bus number
- LED indicator for the communication
- Mounting onto EN50022 DIN rail or wall mounting
- Galvanic insulated RS485 interface for communication with a host system

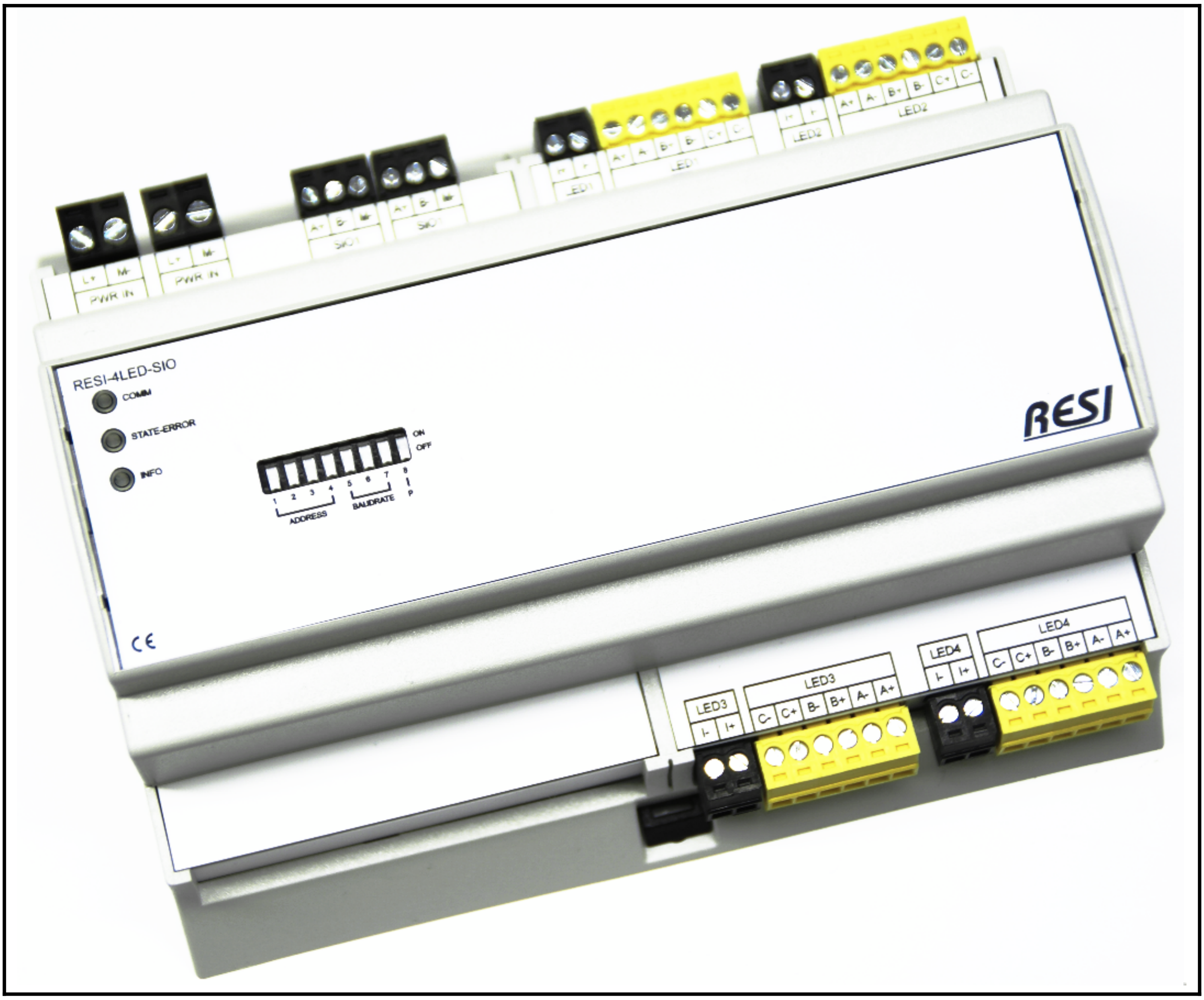


Figure: Our RESI-4LED-SIO module

21.2 The modes of the LED module

The LED module offers six modes. Each of the four LED groups has its own mode. You can switch the mode by setting a special register via MODBUS/RTU or by executing the #SMODEx ASCII command. Be aware that the converter does not save a mode in remanent memory. After reset the module starts always in mode ON for all four LED groups!

21.2.1 LED mode OFF

In this mode all three outputs of a LED group are switched to 0. It doesn't matter, what values are actual in the three set point registers LOx. The registers for the three actual output values CLOx return always the value 0.

21.2.2 LED mode ON

In this mode all three outputs are switched immediately to the current values in the three registers Lox of the corresponding LED group. The three registers for the actual output values CLOx of the affected LED group deliver always the same value as the three registers LOx to indicate, that the values are really outputted to the three PWM channels of the LED group.

21.2.3 LED mode FLASH

In this mode all three outputs are switched as a recycler relay between the three current values in the registers LOx and 0 of the LED group. While ON time span, the module outputs the three values of the set point registers Lox to the real outputs for a timespan defined in the register MINIMUM TIMEx in 1/10s. In this time the registers for the actual output values CLOx of the affected LED group deliver always the same value as in the registers LOx to indicate, that the values are really outputted to the three PWM channels of the affected LED group. Then the converter switches all three channels to 0 for the OFF time span. This time span is defined with the value of the MAXIMUM TIMEx register in 1/10s. In this time span the registers for the actual output values CLOx of the affected LED group deliver always the value 0. This ON/OFF cycle is repeated endlessly.

Steps for FLASH:

Step 1: Output of the three set point values LOx A, LOx B, LOx C to the real PWM outputs
Step 2: Wait for MINIMUM TIMEx in 1/10s
Step 3: output of the values 0, 0, 0 to the real PWM outputs
Step 4: Wait for MAXIMUM TIMEx in 1/10s
Step 5: continue with step 1

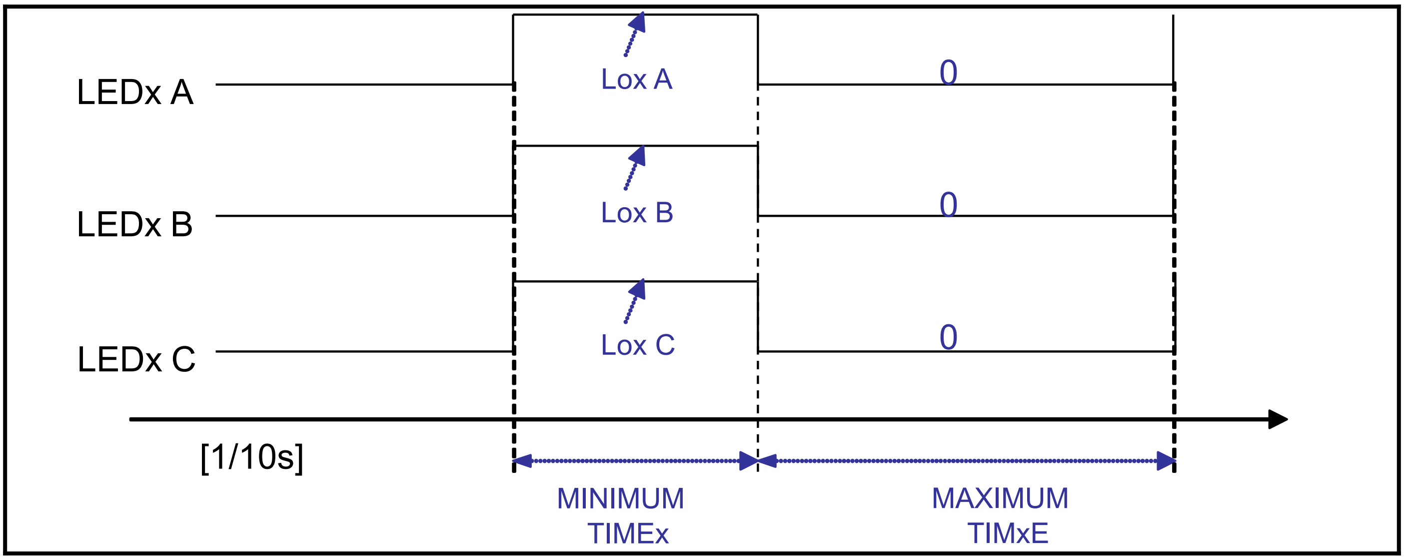


Figure: Timing diagram of mode FLASH

21.2.4 LED mode FADE

In this mode the converter doesn't change the three PWM outputs of a LED group not immediately, if the set point registers Lox are changed. No, it uses a ramp to change slowly from the current value to the new value. This ramp is defined in the register FADE SPEEDx. The setup is done in steps per 1/100s and is valid for all three channels of the affected LED group. To set a new value write into one of the three set point registers LOx. The LED group fades each output channel from the current value to the new set point value. If you read the registers CLOx of the LED group while fading, you will get every value change from the old value to the new value for each channel. Also the register IS FADE ACTIVEx will return a 1 while fading is running at least on one of the three channels of a LED group. When the LED group reaches the new set point values, reading of the registers CLOx will return the same values as in the registers LOx for the affected LED group. Also the register value of IS FADE ACTIVEx will be 0.

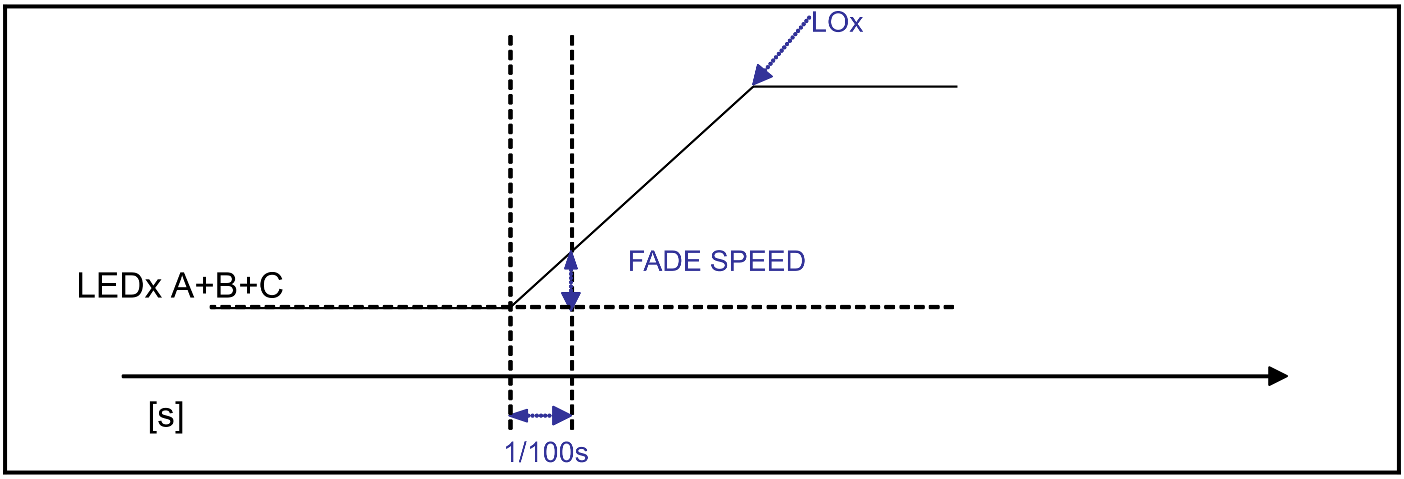


Figure: Timing diagram of mode FADE

21.2.5 LED mode RANDOM

In this mode the module generates random values for each of the three outputs of a LED group. For this random number guessing process, you can setup a time interval. If this time interval expires the system dices new random values for the three outputs of the affected LED group. The time interval is defined by the register MINIMUM TIMEx and the register MAXIMUM TIMEx in seconds. The system generates a random time interval between those two parameters. If the time expires, the system dices new random values for the three registers RLOx of the affected LED group. Then the system fades the current values stored in the registers CLOx to the new random values RLOx. This fade ramp is defined in the register FADE SPEEDx. The setup is done in steps per 1/100s. If you read the registers CLOx while fading, you will get every value change from the old values to the new end values RLOx. Also the register IS FADE ACTIVE 4x00014 will return a 1 while fading is running at least on one of the three output channels of the LED group. When the module reaches the new values, reading of the registers CLOx will return the same values as stored in the registers RLOx. Also the register value of IS FADE ACTIVEx will be 0. The diced values in the registers RLOx will be in the range of 0 to LOx.

Steps for RANDOM:

Step 1: Dice three random numbers in the range of 0..LOx and store the values in RLOx
Step 2: Dice a random wait period between MINIMUM TIMEx and MAXIMUM TIMEx in seconds
Step 3: Fade up or down from the actual output values CLOx to the new end values RLOx
Step 4: If the random wait period is over, continue with step 1

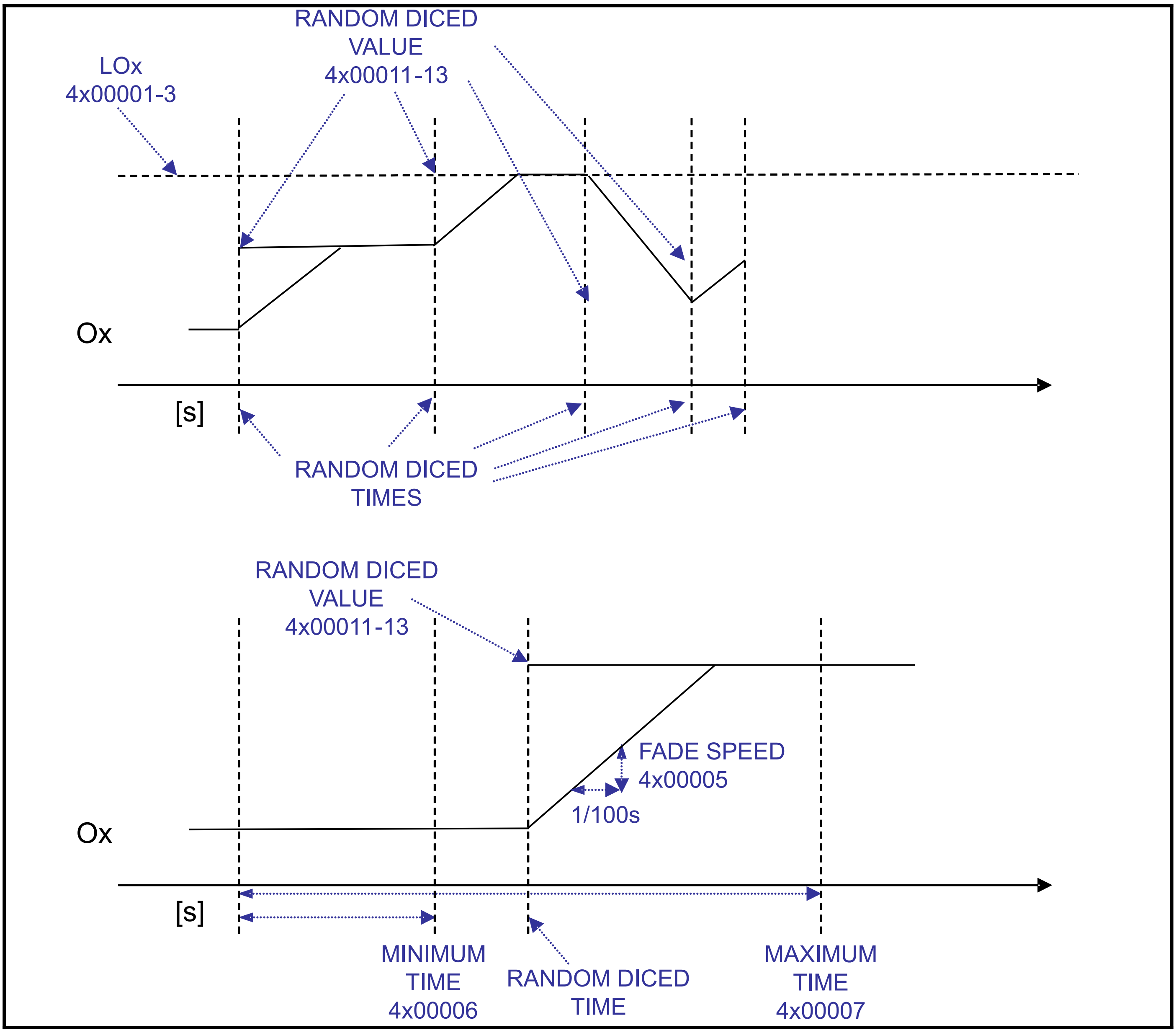


Figure: Timing diagram of mode RANDOM

21.2.6 LED mode SEQUENCE

In this mode, the module creates a sequential flash light with the three PWM outputs of a LED group. The outputs flashes between the three set points LOx A, LOx B and LOx C and 0 of the affected LED group in sequence. In the first ON phase the module sets the real output CLOx A to the set point LOx A, the other two outputs are set to 0. This phase lasts for MINIMUM TIMEx in 1/10s. While this period of time, the current value register CLOx A delivers the same value as stored in LOx A, and the other two current value registers CLOx B and CLOx C deliver the value 0. Then the module switches all three outputs to 0 for a time period defined with the register MAXIMUM TIMEx in 1/10s (OFF time period). While this period of time, all three output registers CLOx deliver the value 0. Now the system repeats the ON phase with the next set point register LOx B. The two registers CLOx A and CLOx C are set to 0 in this phase. Next the OFF time period is executed. The last phase is the ON phase with the register LOx C. The two registers CLOx A and CLOx B are 0 in this phase. The last OFF time period is executed. This three times ON/OFF cycle is repeated endlessly.

Steps for SEQUENCE:

Step 1: Output the three set points LOx A, 0, 0 to the three PWM outputs
Step 2: wait for MINIMUM TIMEx in 1/10s
Step 3: Output the values 0, 0, 0 to the three PWM outputs
Step 4: wait for MAXIMUM TIMEx in 1/10s
Step 5: Output the three set points 0, LOx B, 0 to the three PWM outputs

Step 6: wait for MINIMUM TIMEx in 1/10s
Step 7: Output the values 0, 0, 0 to the three PWM outputs
Step 8: wait for MAXIMUM TIMEx in 1/10s
Step 9: Output the three set points 0, 0, LOx C to the three PWM outputs
Step 10: wait for MINIMUM TIMEx in 1/10s
Step 11: Output the values 0, 0, 0 to the three PWM outputs
Step 12: wait for MAXIMUM TIMEx in 1/10s
Step 13: continue with step 1

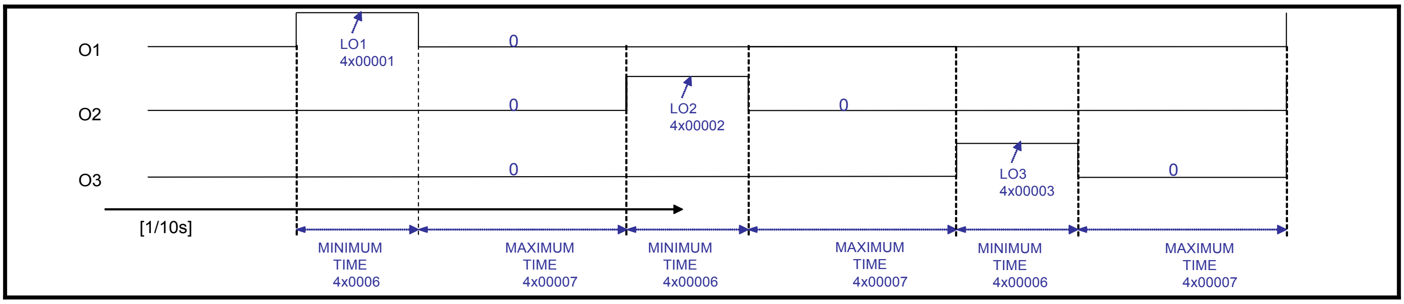


Figure: Timing diagram of mode SEQUENCE

21.3 Technical specification

Beside the basic technical data, which fulfil all of our IO modules, this IO modules meet the following technical specifications:

Power consumption	
RESI-4LED-SIO	<0.5W

Product housing	
RESI-4LED-SIO	BIG IO XT8

Product weight	
RESI-4LED-SIO	295g

LED stripe PWM output	
Total amount of PWM outputs	12
LED groups	4 LED groups organized with
	3 individual dimmable outputs
	each with own power supply
Amount of PWM outputs per LED group	3 individual dimmable outputs
Output signal	PWM with 400Hz
LED stripes	RGB, RGBW, dual white or mono color LED stripes
LED connection	via common anode
LED Output voltage	0..48Vdc
LED Output current per channel	max. 5A per LED output
LED Input voltage per LED group	0..48Vdc, depending on LED stripe types
LED Input current per LED group	max. 15A
	180W@12Vdc
	360W@24Vdc
	720W@48Vdc
LED group power supply	Via 2-pin plug-in terminal block in black
LED group dimmable outputs	Via 6-pin plug-in terminal block in yellow
Terminal type	RM3.5
Galvanic insulation to the CPU logic	Yes
	All LED groups are internally coupled via the common ground

Default serial settings	
Baud rate	via DIP switch
Parity	none
Stop bit(s)	one
UnitID	255

21.4 Additional terminals & LED states

LED GROUP		
LED1	3 dimmable PWM outputs with own LED power supply	
	One 2 pin plug-in terminal block for LED power supply in black	
	Terminal type: RM3.5	
	I+:	LED group power supply (0..48Vdc, max. 15A)
	I-:	LED group power supply Ground
Pin layout	Pin 1:	I+: Power supply
	Pin 2:	I-: Power supply ground
	One 6 pin plug-in terminal block for LED output channels 1-3 in yellow	
	Terminal type: RM3.5	
	A+:	LED output channel 1 common anode
	A-:	LED output channel 1 PWM pulsed cathode (max. 5A)
	B+:	LED output channel 2 common anode
	B-:	LED output channel 2 PWM pulsed cathode (max. 5A)
	C+:	LED output channel 3 common anode
	C-:	LED output channel 3 PWM pulsed cathode (max. 5A)
	HINT:	The connectors I+, A+, B+ and C+ are internally tied together
		The used LED stripes must have a common anode pin!
Pin layout	Pin 1:	A+
	Pin 2:	A-
	Pin 3:	B+
	Pin 4:	B-
	Pin 5:	C+
	Pin 6:	C-
LED GROUP		
LED2	3 dimmable PWM outputs with own LED power supply	
	like LED GROUP LED1	
LED GROUP		
LED3	3 dimmable PWM outputs with own LED power supply	
	like LED GROUP LED1	
LED GROUP		
LED4	3 dimmable PWM outputs with own LED power supply	
	like LED GROUP LED1	

21.5 Connection diagram

21.5.1 Connection of LED stripes power supply

In the below drawings we will show the cabling of various types of LED stripes to our module. We will discuss how to connect different types of LED stripes to our module in detail.

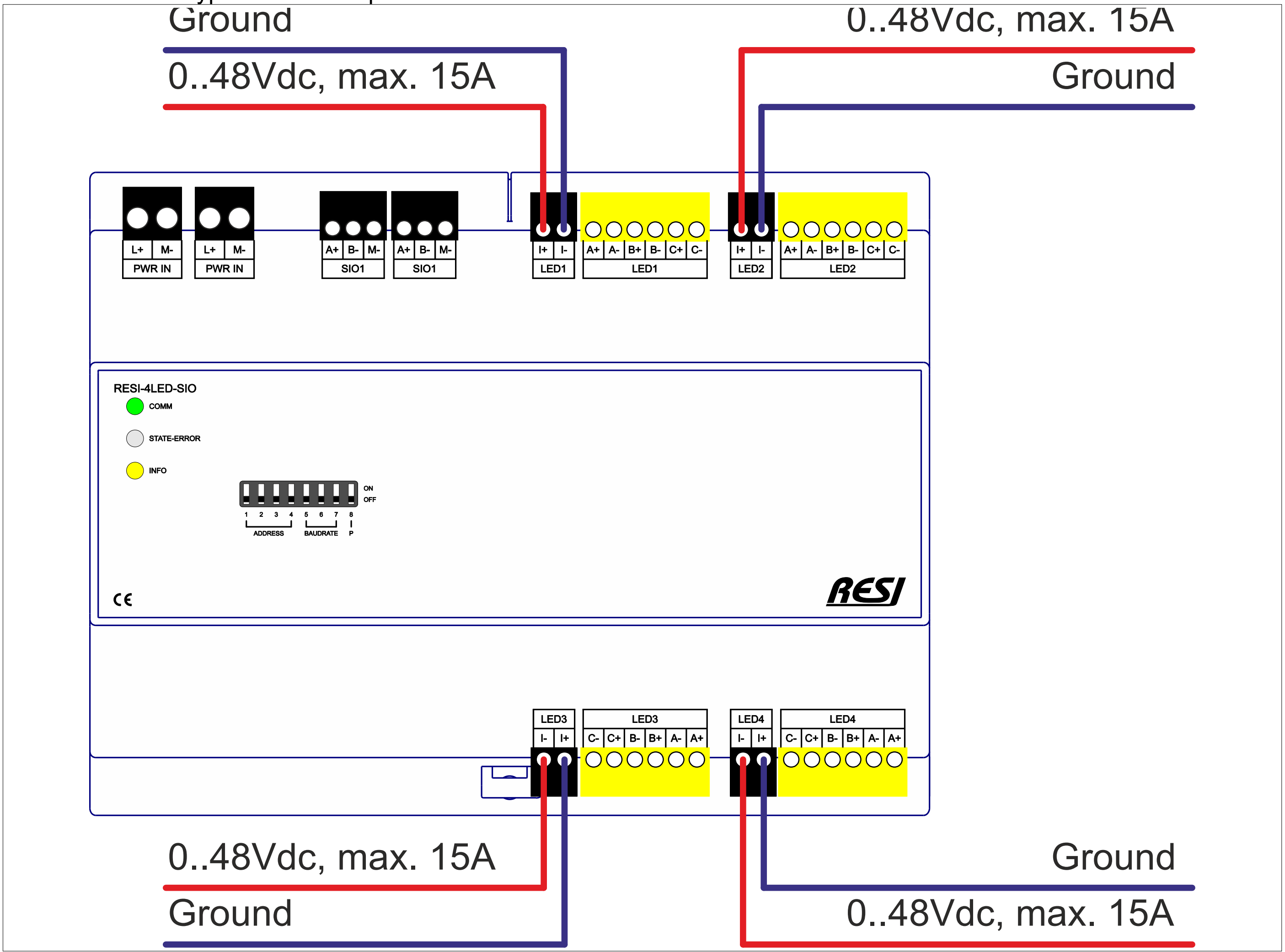


Figure: Power supply for the LED stripes

The module offers 4 independed LED groups LED1, LED2 LED3 and LED4. Each of the LED groups offers three individual dimmable outputs A, B and C.

For the LED stripes of each of the four LED groups you can use individual power supplies or one for all four LED groups together. It depends on the power consumption and the voltage level of the used LED stripes. Therefore each LED group offers two terminals I+ and I-. Depending on the type of LED stripe, you can use different types of power supplies. It is very important, that the maximum current, which the power supply delivers, must not exceed 15A! Due to this the following limitations arise for powering the LED stripes:

- LED stripes for 12Vdc voltage: 12Vdc*15A -> max. 180W power supply
- LED stripes for 24Vdc voltage: 24Vdc*15A -> max. 360W power supply
- LED stripes for 48Vdc voltage: 48Vdc*15A -> max. 720W power supply

But be careful! Each dimmable PWM output can only drive max. 5A current for dimming!

IMPORTANT: Each of the four LED groups can have its own power supply. They can have different voltage levels for each LED group e.g. 12vdc on LED group LED1 and 24Vdc on LED group LED2. But due to the design of the module all four ground terminals (I-) are bridged internally!

21.5.2 Connection of one mono color LED stripe to one group

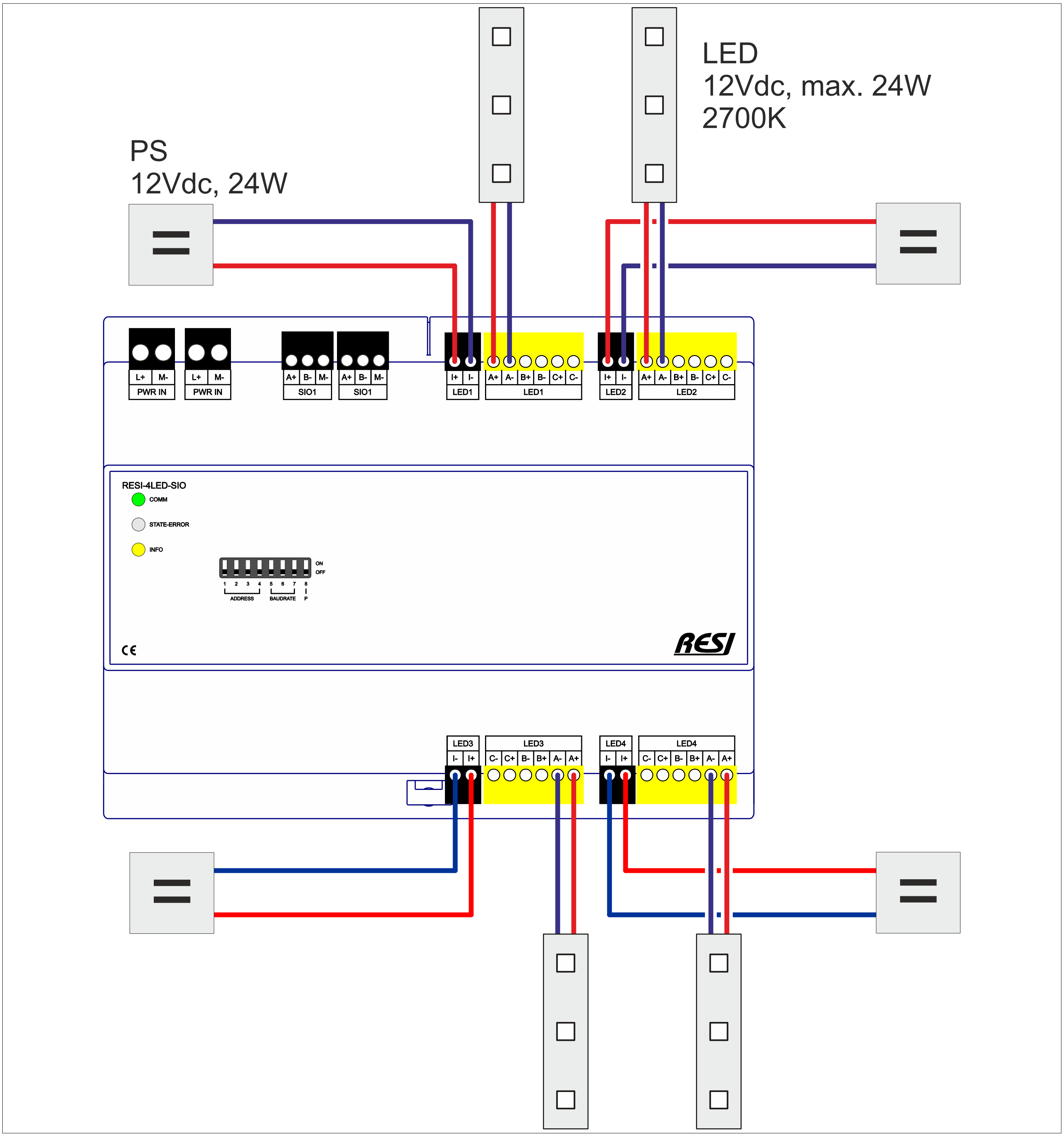


Figure: Cabling of one 12Vdc LED stripe with 24W power consumption, luminous color 2700K, per LED group. Due to the reason, that the LED stripe consumes only 24W, we use also a 24W power supply. This results in an input current for the clamps I+ and I- of 2A (This is far below 15A and therefore ok). The output current flow over the terminal A is 2A (<5A, again this is ok).

IMPORTANT: Dot forget, that all external power supplies are connected internally with the terminal I- of all four LED groups! Please also remember, that all anodes with the terminals I+, A+, B+ and C+ are tied together in one LED group!

21.5.3 Connection of three mono color LED stripes to one group

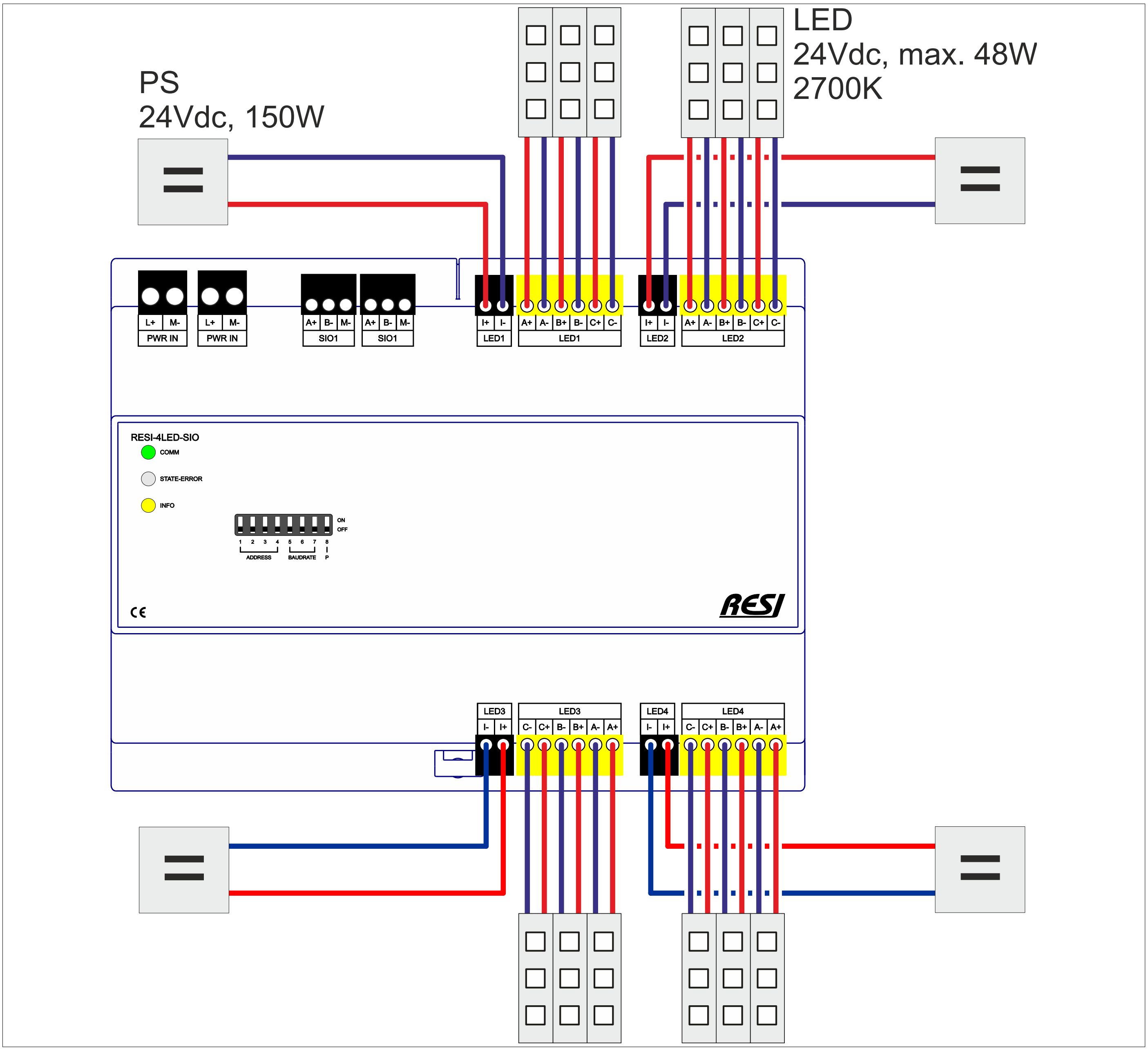


Figure: Cabling of three 24Vdc LED stripes with 48W power consumption of each LED stripe per LED group. Each of the three LED stripes of a LED group can be dimmed individually. This example uses all three outputs A, B and C of a LED group. Each of the three LED stripes consumes a maximum of 48W of power. We use a power supply with 3x48W -> 150W. The input current, which flows over the terminals I+ and I- is now max. 6.25A. That's far beyond the limit of 15A and ok. Due to the fact, that on each output we have only connected a 48W LED stripe, the current flow over each of the three outputs A, B and C is max. 2A. That's again is beyond the limit of 5A and ok.

IMPORTANT: Dot forget, that all external power supplies are connected internally with the terminal I- of all four LED groups! Please also remember, that all anodes with the terminals I+, A+, B+ and C+ are tied together in one LED group!

21.5.4 Connection of two mono color LED stripes to one group

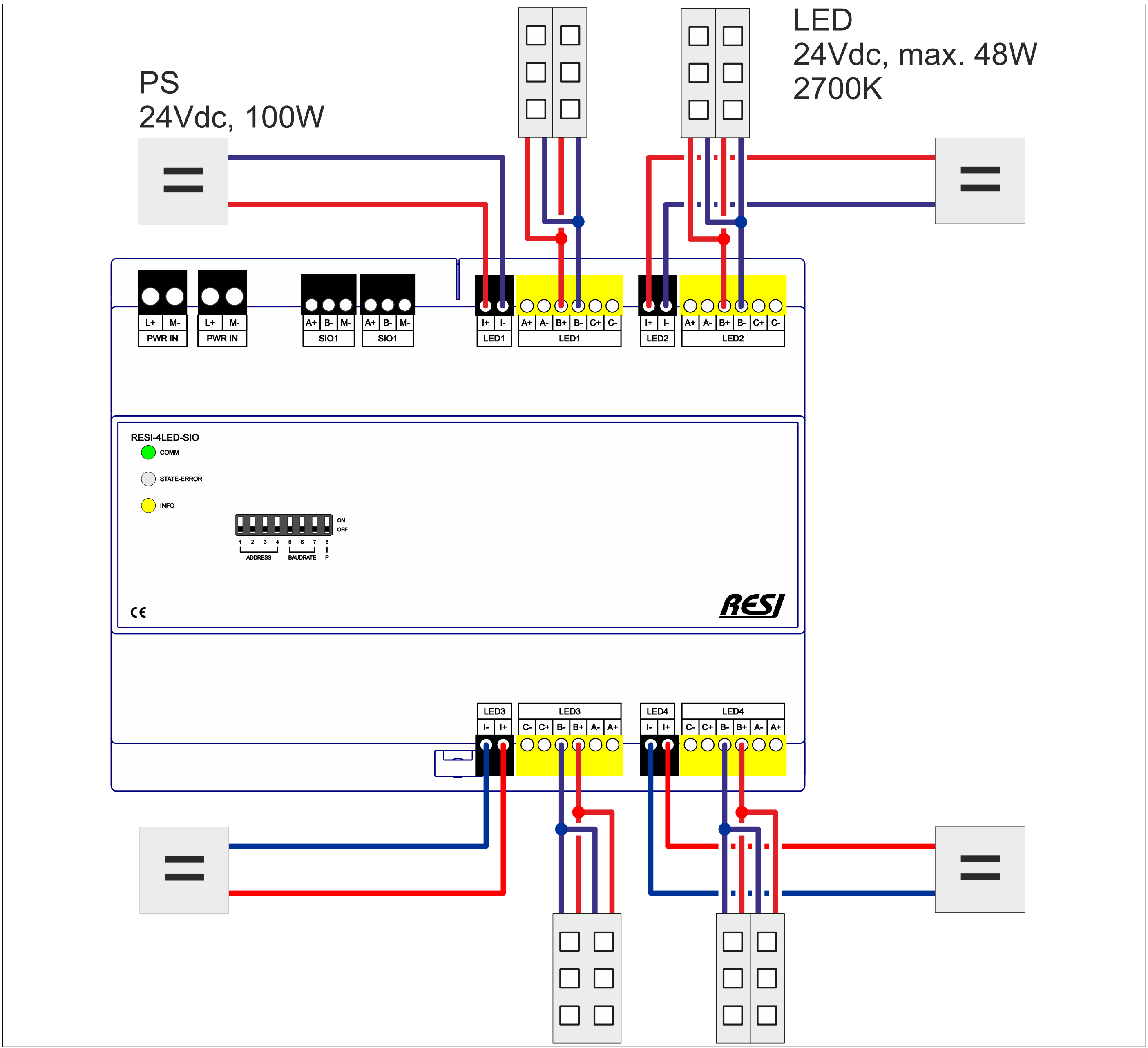


Figure: Cabling of two 24Vdc LED stripes with 48W power consumption of each LED stripe on the output B per LED group. Both LED stripes are dimmed as a group with output B of the LED group. We use a 100W power supply on the input I+ and I-. This results in an input current of 4.17A, which is again below the maximum current of 15A and ok. Now we have connected two LED stripes to one output. This results in a total power consumption of 96W on the output B. This leads to an output current of 4A. That's again lower than 5A and ok.

IMPORTANT: Dot forget, that all external power supplies are connected internally with the terminal I- of all four LED groups! Please also remember, that all anodes with the terminals I+, A+, B+ and C+ are tied together in one LED group!

21.5.5 Connection of one RGB LED stripe to one group

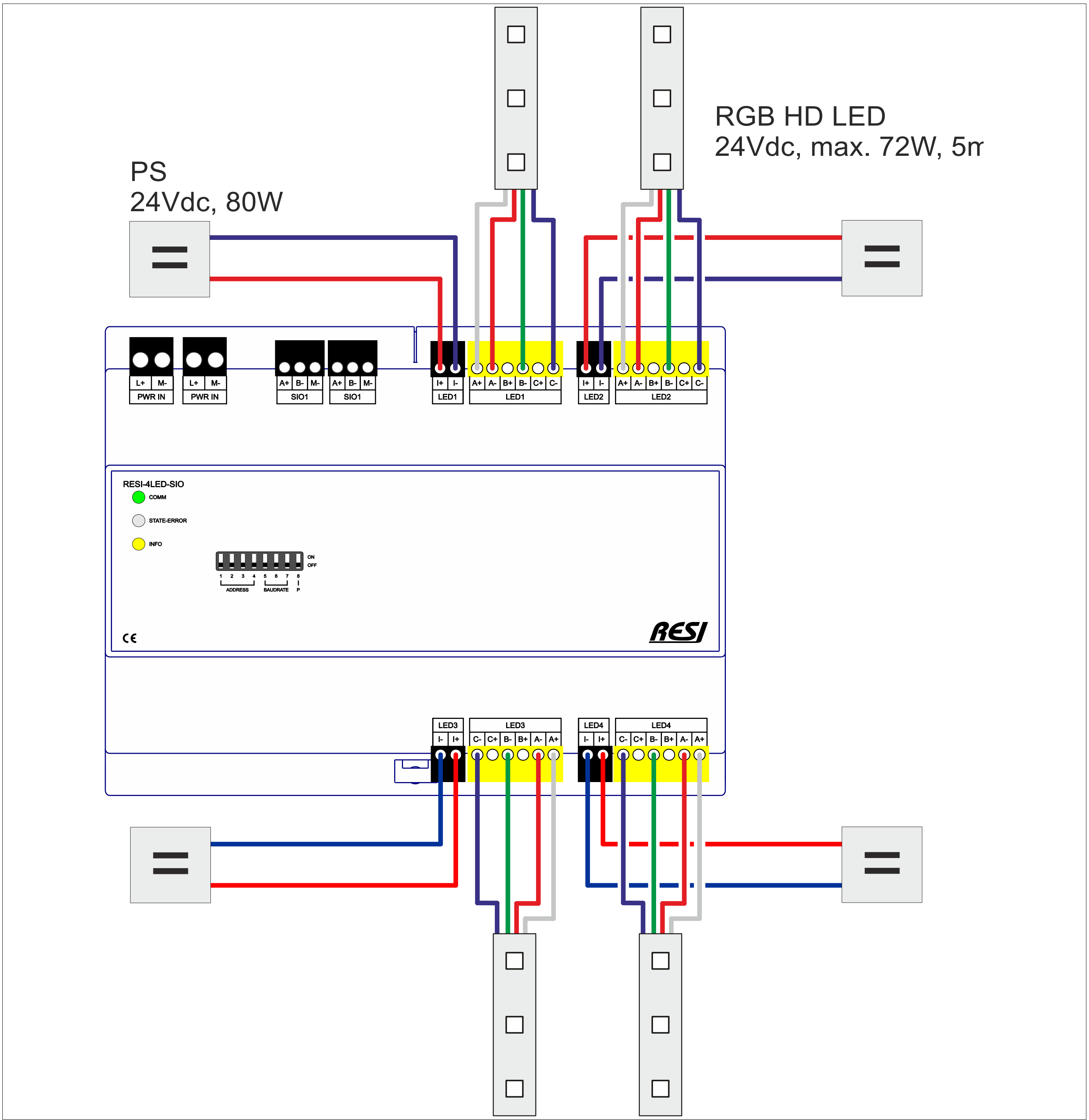


Figure: In this example we use a RGB LED HD stripe per LED group. This stripe offers three dimmable channels, one for the color red, one for green and one for blue. The common anode of the LED stripe is connected to the terminal A+ of our module. The 80W power supply delivers a maximum current of 3,34A. This is far beyond the allowed 15A and ok. Each output channel must drive only 1/3rd of the 72W total power consumption of the LED stripe. This equals to 24W, resulting in a maximum current for each output of 1A. Again this is far beyond the allowed 5A for each channel and ok.

IMPORTANT: Dot forget, that all external power supplies are connected internally with the terminal I- of all four LED groups! Please also remember, that all anodes with the terminals I+, A+, B+ and C+ are tied together in one LED group!

21.5.6 Connection of one dynamic white LED stripe to one group

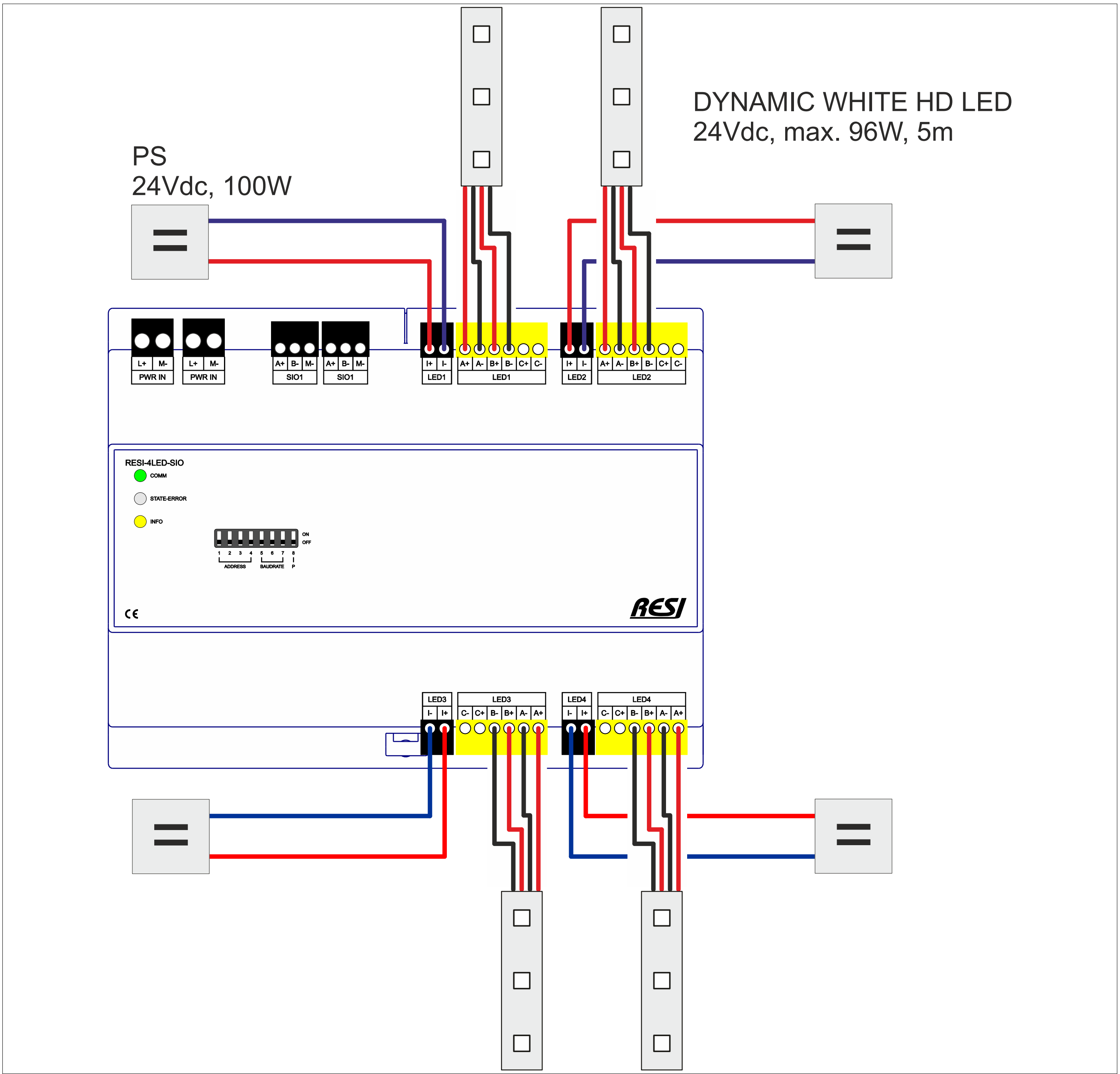


Figure: Cabling of a dynamic white LED stripe per LED group. This type of LED stripe combines two different LED types with different luminant colors into one product. Each of the two build-in LED stripes consumes 48W of power and you can mix up different luminant colors mostly from warm white to cold white. Connect the four cables of the LED stripe a shown in the above drawing. We also connect the both anodes of the dual LED stripe to the clamps A+ and B+. The cable for warm white is connected to the output A- and the cable for the cold white LEDs is connected to the output B+. The output clamps C+ and C- stay unconnected. Due to the fact, that the outputs A and B have to drive only 48W each, the maximum output current per channel is 2A. This is far beyond the allowed 5A and ok. The input current on the clamps I+ and I- lies by maximum 4.16A with the 100W power supply. This again is far under the allowed 15A and ok.

IMPORTANT: Dot forget, that all external power supplies are connected internally with the terminal I- of all four LED groups! Please also remember, that all anodes with the terminals I+, A+, B+ and C+ are tied together in one LED group!

21.5.7 Connection of one cold+warm white LED stripe to one group

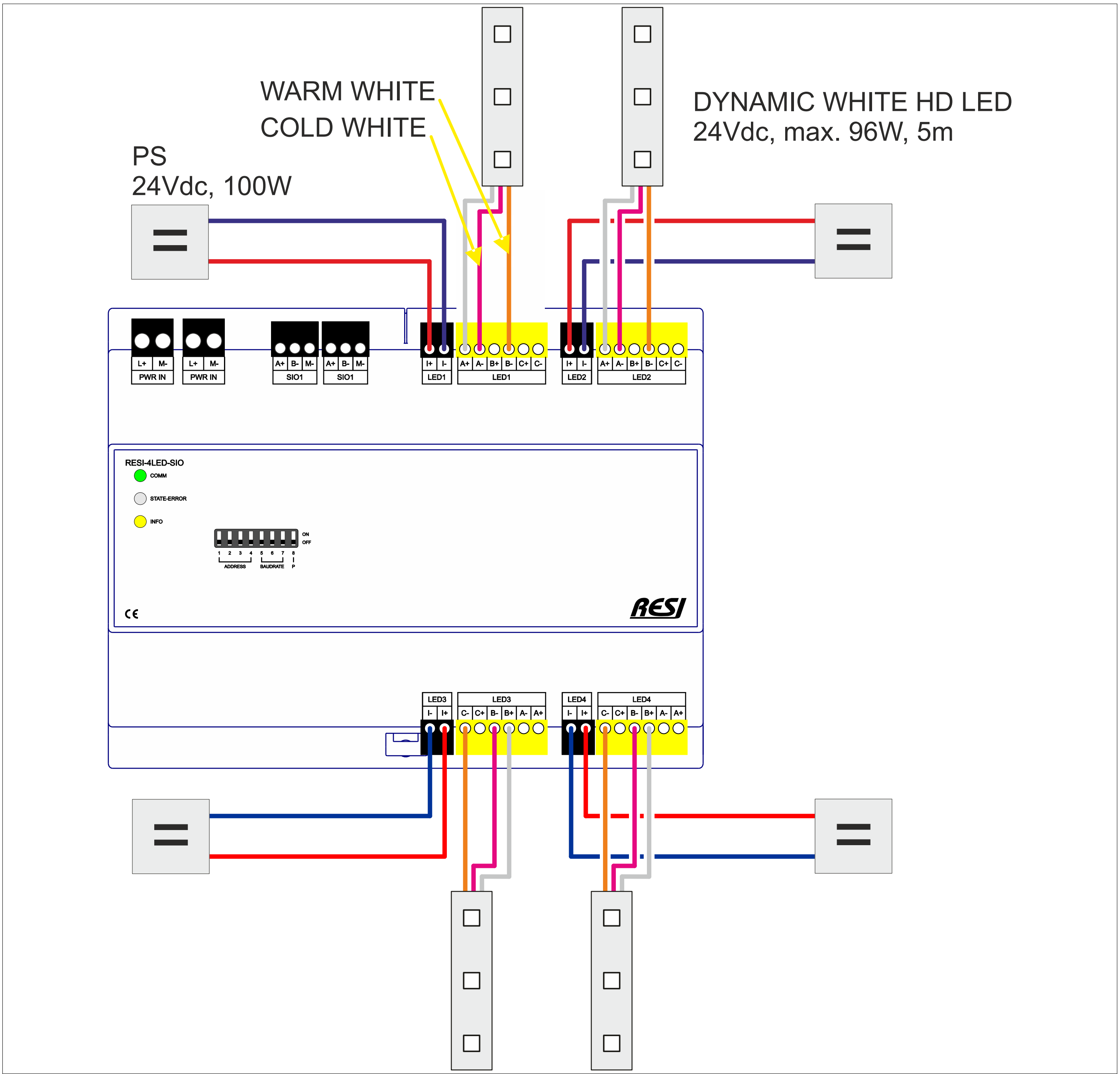


Figure: Cabling of a warm white/cold white LED stripe per LED group. This type of LED stripe combines two different LED types with different luminant colors into one product. Each of the two build-in LED stripes consumes 48W of power and you can mix up different luminant colors mostly from warm white to cold white. Connect the three cables of the LED stripe a shown in the above drawing. We also connect the anode of the dual LED stripe to the clamps A+. The cable for cold white is connected to the output A- and the cable for the warm white LEDs is connected to the output B+. The output clamps C+ and C- stay unconnected. Due to the fact, that the outputs A and B have to drive only 48W each, the maximum output current per channel is 2A. This is far beyond the allowed 5A and ok. The input current on the clamps I+ and I- lies by maximum 4.16A with the 100W power supply. This again is far under the allowed 15A and ok.

IMPORTANT: Dot forget, that all external power supplies are connected internally with the terminal I- of all four LED groups! Please also remember, that all anodes with the terminals I+, A+, B+ and C+ are tied together in one LED group!

21.5.8 Connection of one RGBW LED stripe to two groups

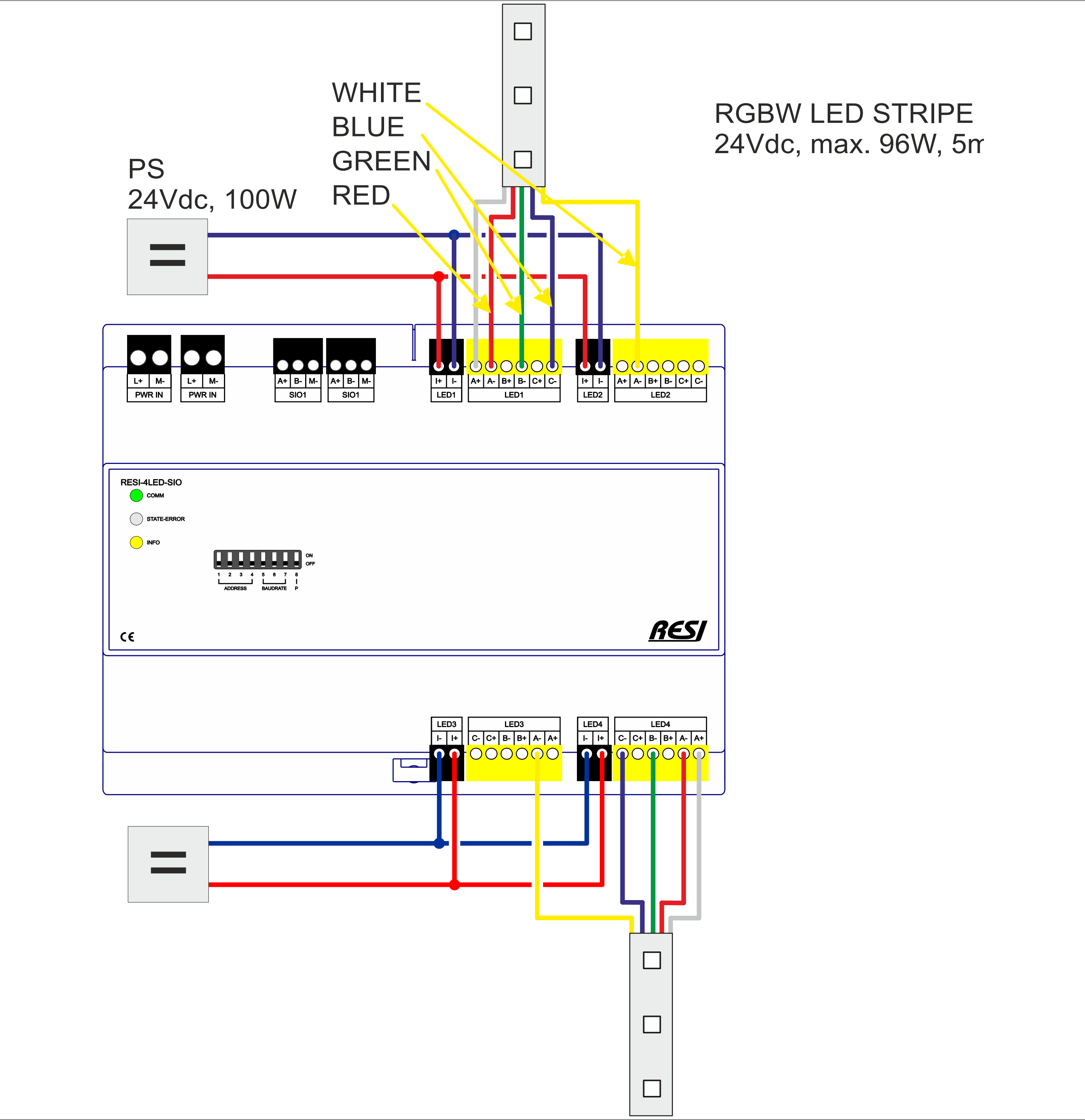


Figure: Cabling of a RGBW LED stripe with two LED groups. This type of LED stripe needs 4 PWM outputs to control all four different LED colors. So we use from LED group 1 A- to dimm the red LEDs, B- to dimm the green LEDs and C- to dimm the blue LEDs. Then we tie the power suuply of the LED group 2 to the power supply of the LED group 1. No we connect the white LEDs to the A- on the second LED group.

IMPORTANT: Dot forget, that all external power supplies are connected internally with the terminal I- of all four LED groups! Please also remember, that all anodes with the terminals I+, A+, B+ and C+ are tied together in one LED group!

21.6 Assignment of the channel numbers to the output clamps

Here you can find a definition, how the channel numbers are mapped to the output terminals:

LED group	Clamp	Group Number	Group channel	Channel number
LED1	A+ A-	1	1	1
LED1	B+ B-	1	2	2
LED1	C+ C-	1	3	3
LED2	A+ A-	2	1	4
LED2	B+ B-	2	2	5
LED2	C+ C-	2	3	6
LED3	A+ A-	3	1	7
LED3	B+ B-	3	2	8
LED3	C+ C-	3	3	9
LED4	A+ A-	4	1	10
LED4	B+ B-	4	2	11
LED4	C+ C-	4	3	12

21.7 Additional MODBUS register & coils

Here you will find only the additional MODBUS registers and coils especially for this IO module. Please refer to the description of of the standard MODBUS mapping for more details about the available basic MODBUS registers and coils.

Please refer to the external document for detailed documentation of the current MODBUS register mapping for this IO module:

RESI-L-4LED-SIO-MODBUS+ASCII-ENxx.pdf

21.8 Additional ASCII commands

Here you will find only the additional ASCII commands especially for this IO module. Please refer to the description of of the standard commands for more details about the available basic ASCII commands.

Please refer to the external document for detailed documentation of the current ASCII commands for this IO module:

RESI-L-4LED-SIO-MODBUS+ASCII-ENxx.pdf

22 RESI-8RTD-SIO

22.1 General information

This series of IO modules offer the following features:

- 8 sensor inputs for temperature sensors
- Measurement accuracy +/-0.1%
- Measurement resolution +/-0.001%
- Measurement range -200°C…+850°C
- Various sensor types are applicable: PT100, PT1000, PT10, PT50, PT200, PT500, NI120, NI1000-DIN43760
- Various standards for linearisation are select-able: Europa, America, Japan, ITS-90
- Output of the temperatures in °Celsius [°C], °Fahrenheit [°F] or °Kelvin [°K]
- Different measurement currents are select-able: 5µA, 10µA, 25µA, 50µA, 100µA, 250µA,500µA, 1mA
- Various sensor connection types: 2 wire, 3 wire or 4 wire sensors connectable
- Internal calculation of an average temperature per channel
- Galvanic insulated RS485 interface for communication with a host system

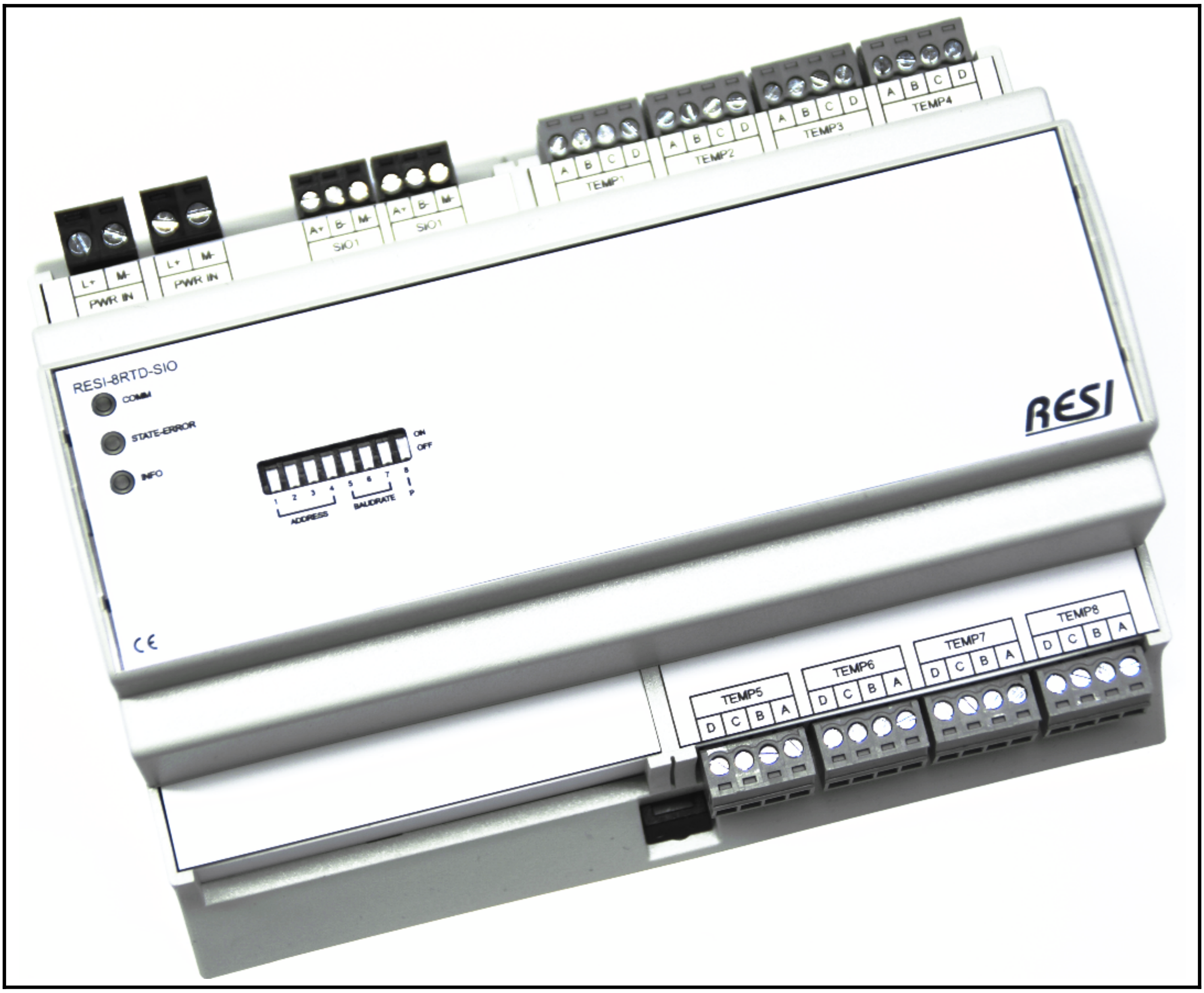


Figure: Our RESI-8RTD-SIO module

22.2 Technical specification

Beside the basic technical data, which fulfil all of our IO modules, this IO modules meet the following technical specifications:

Power consumption	<0.8W
Product housing	BIG IO XT8
Product weight	290g
Temperature inputs	
Number	8
Signal type	Temperature measurement
Measurement type	Measurement of resistance
Sensor connection	4 wire measurement
ADC	24 bit sigma delta ADC
Accuracy	+/-0.1°C for PT-100, PT-200,PT-500, PT-1000
	+/-0.1°C NI-120, NI-1000-DIN43760
	+/-3°C for PT-10, PT-50
Resolution	+/-0.001°C
Reference stability	10ppm/°C
Sensor types	PT-100, PT-1000, PT-1000 α=0.00375, PT-10, PT-50, PT-200,
	PT-500, NI-120, NI-1000 DIN43760
Linearisation standards	Europa, America, Japan, ITS-90
Excitation current for measurement	5µA, 10µA, 25µA, 50µA, 100µA, 250µA, 500µA, 1mA
Cable connection	Via 8 4-pin plug-in terminal block
Terminal type	RM3.5
Galvanic isolation	Yes, to the rest of the module, not to the other temperature inputs
Default serial settings	
Baud rate	via DIP switch
Parity	none
Stop bit(s)	one
UnitID	255

22.3 Additional terminals & LED states

TEMPERATURE INPUTS	8 temperature inputs for RTD temperature sensors	
	Eight 4 pin plug-in terminal block	
	Terminal type:	RM3.5
	TEMPx:A:	Temperature Sensor Cable 1A
	TEMPx:B:	Temperature Sensor Cable 1B
	TEMPx:C:	Temperature Sensor Cable 2A
	TEMPx:D:	Temperature Sensor Cable 2B
Connecting 4-wire temperature sensors:		
Pin layout	Pin 1:	A: Wire 1 of Sensor
	Pin 2:	B: Wire 2 of Sensor
	Pin 3:	C: Wire 3 of Sensor
	Pin 4:	D: Wire 4 of Sensor
HINT:	Sensor element between wire 2 and 3, wire 1 and 2 on left side of sensor element	
	and wire 3 and 4 on right side of sensor element	
Connecting 3-wire temperature sensors:		
Pin layout	Pin 1:	A: Wire 1 of Sensor
	Pin 2:	B: Wire 2 of Sensor
	Pin 3:	C: Wire 3 of Sensor
	Pin 4:	D: Bridged to terminal C
HINT:	Sensor element between wire 2 and 3, wire 1 and 2 on left side of sensor element	
	and wire 3 on right side of sensor element	
Connecting 2-wire temperature sensors:		
Pin layout	Pin 1:	A: Bridged to terminal B
	Pin 2:	B: Wire 1 of Sensor
	Pin 3:	C: Wire 2 of Sensor
	Pin 4:	D: Bridged to terminal C
HINT:	Sensor element between wire 1 and 2, wire 1 on left side of sensor element	
	and wire 2 on right side of sensor element	
INFO	If everything is ok this LED is on. If there is an internal error	
	with the temperature measurement, this LED flashes fast.	

22.4 Schematic diagram

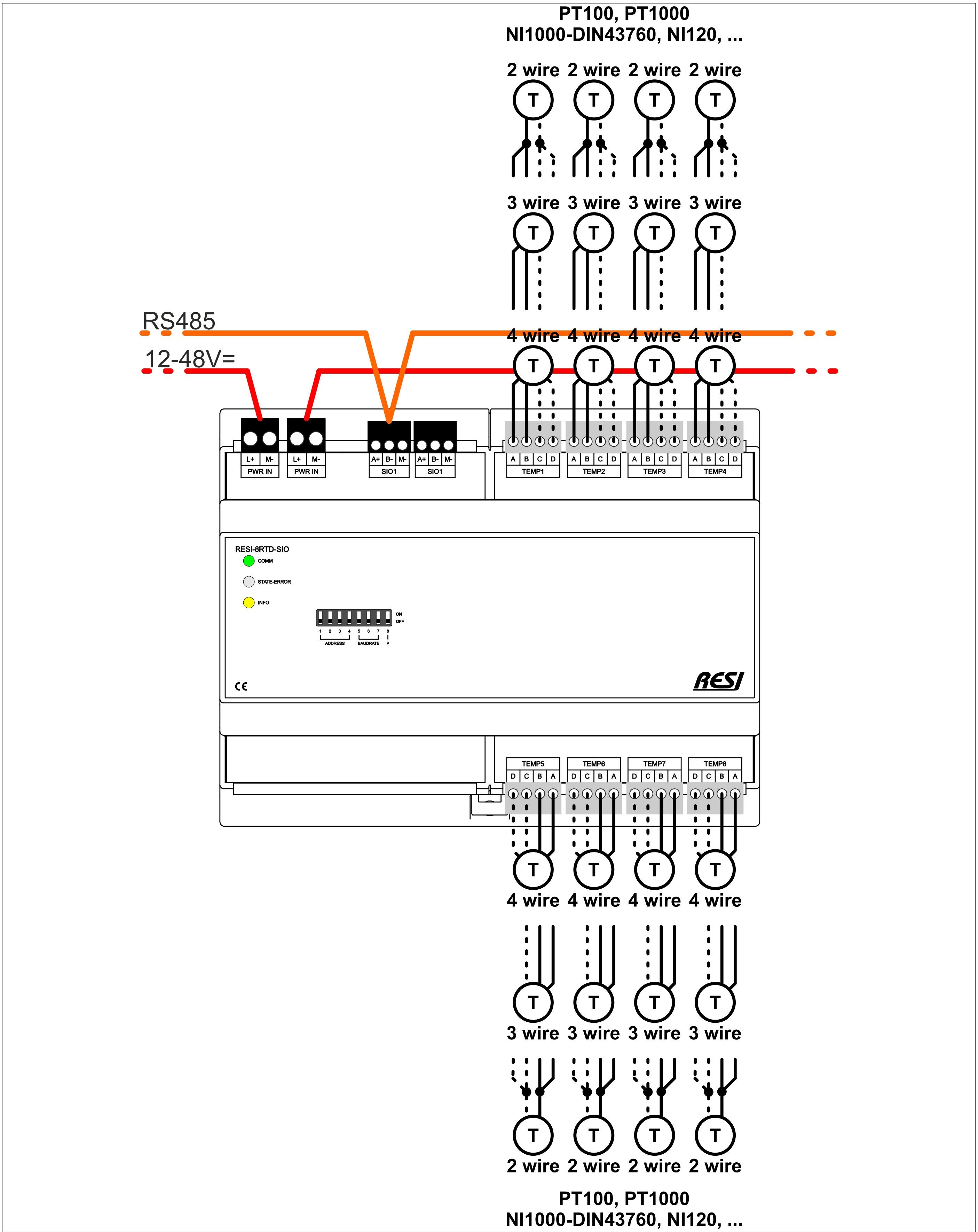


Figure: Schematic diagram of IO module

22.5 Cabling of temperature sensors

A typical temperature sensor with different connection cables is shown in the figure below:

- 2 wire: A red and white cable
- 3 wire: Two red and one white cable
- 4 wire: Two read and two white cable

The sensor element is always mounted between the red and white cables!

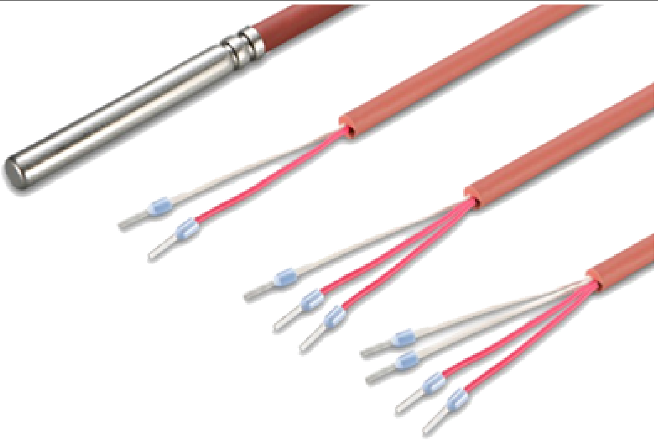


Figure: Typical temperature sensor with different connection cables

22.5.1 Cabling of 4-wire temperature sensors

In the below drawing you see the cabling of 4-wire temperature sensors:

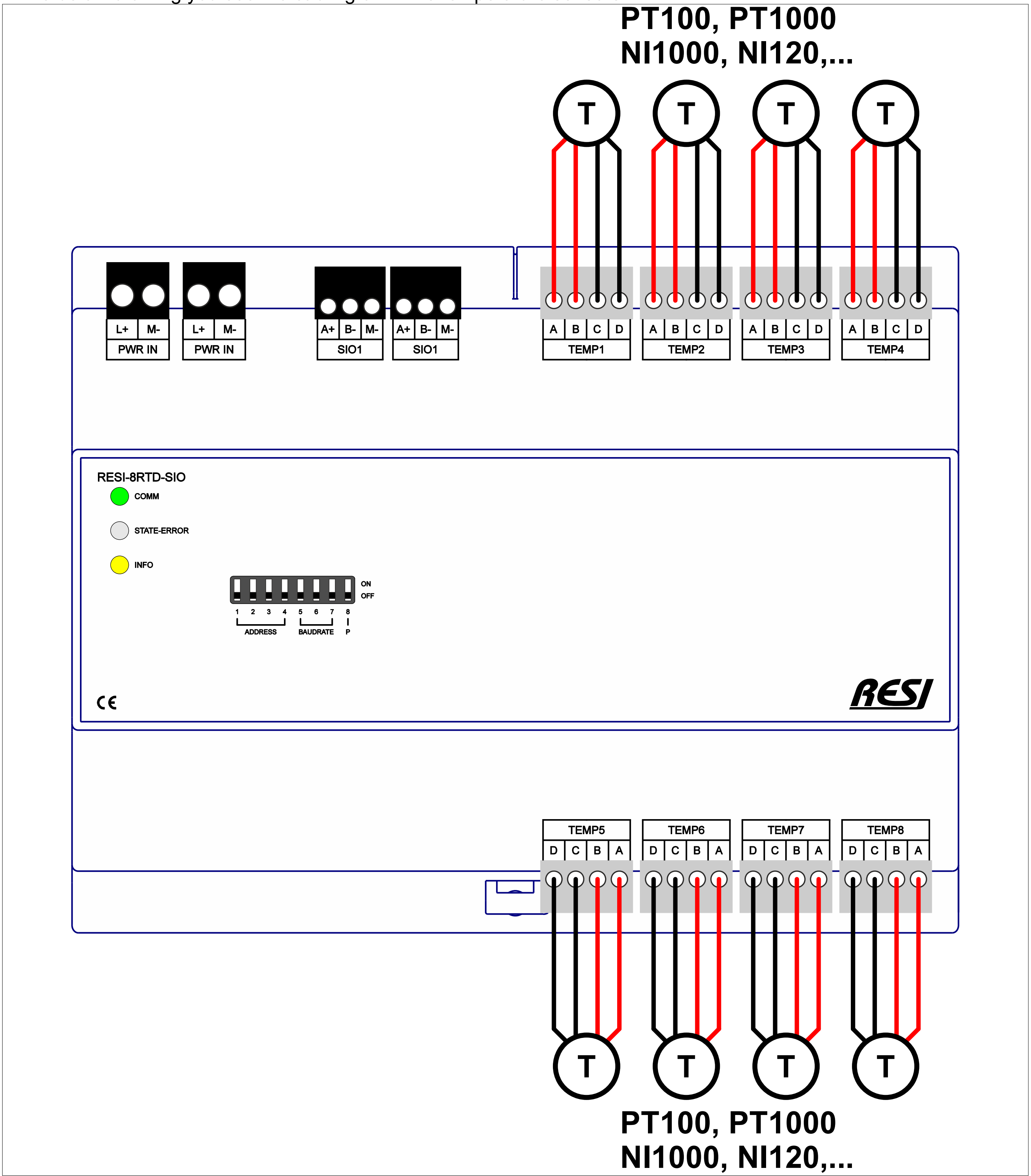


Figure: Cabling of temperature sensors with 4 wires

Don’t forget, that you can mix the type of temperature sensors for each channel!

22.5.2 Cabling of 3-wire temperature sensors

In the below drawing you see the cabling of 3-wire temperature sensors:

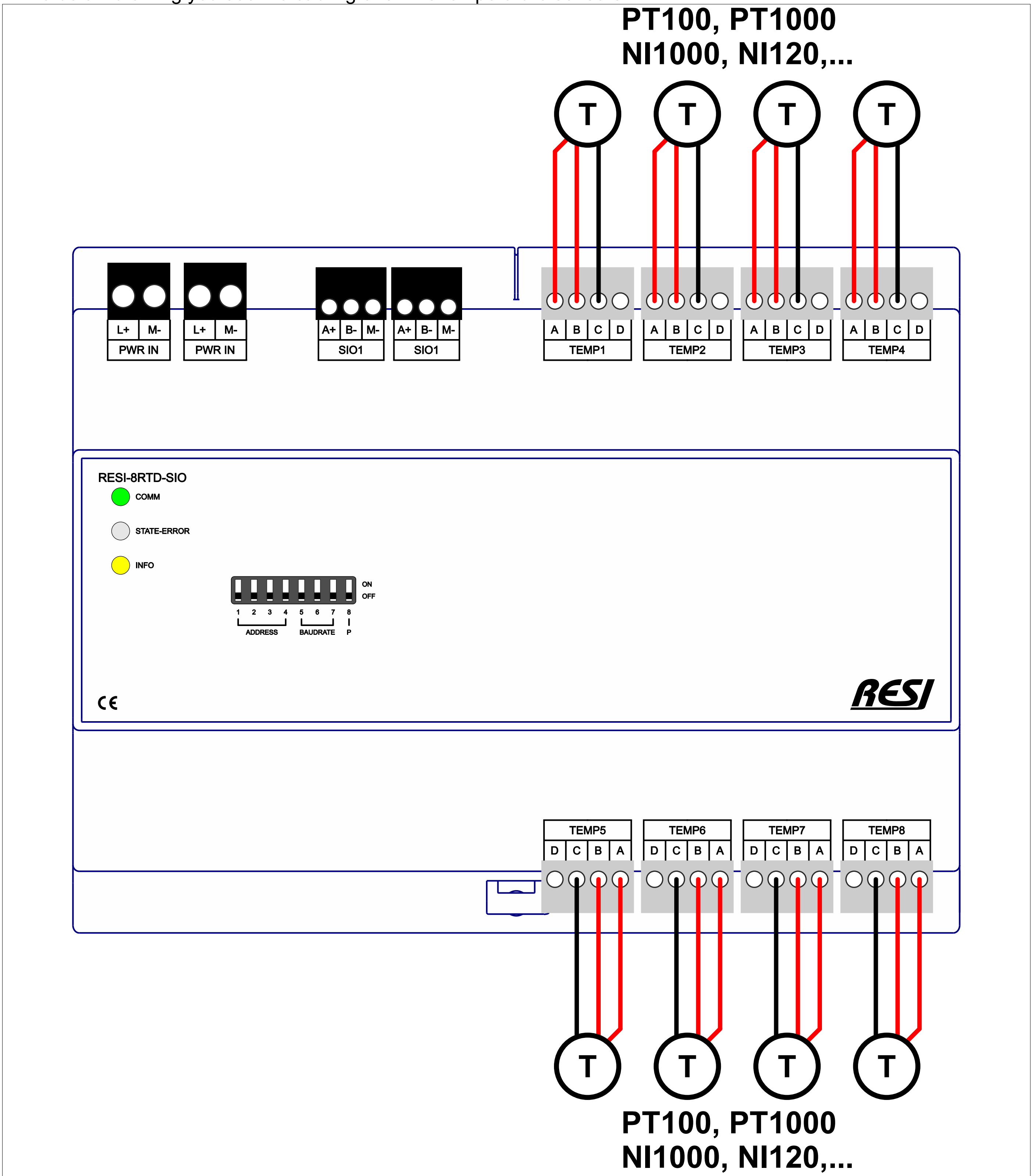


Figure: Cabling of temperature sensors with 3 wires

Don’t forget, that you can mix the type of temperature sensors for each channel!

22.5.3 Cabling of 2-wire temperature sensors

In the below drawing you see the cabling of 2-wire temperature sensors:

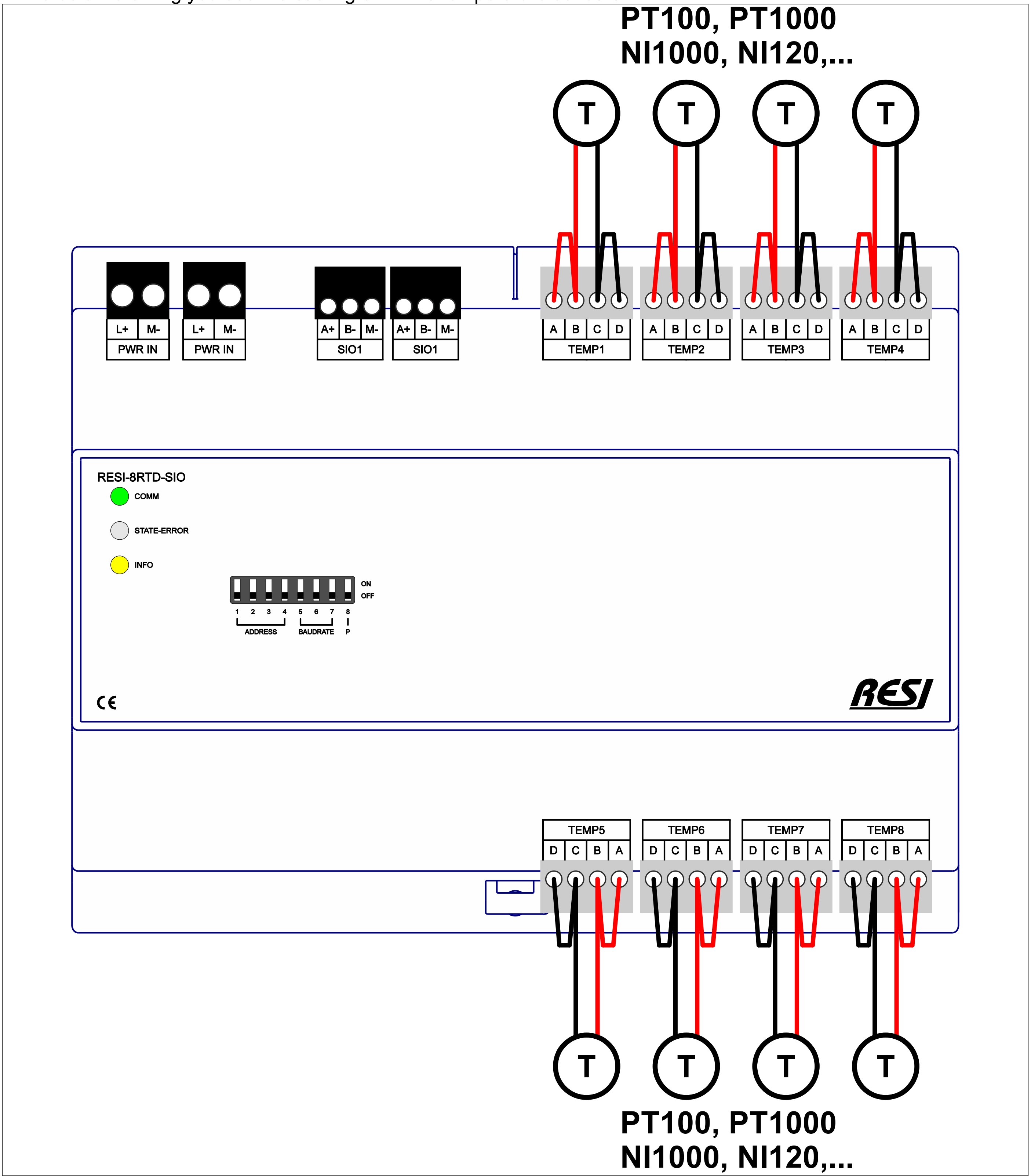


Figure: Cabling of temperature sensors with 2 wires

Don't forget, that you can mix the type of temperature sensors for each channel!

22.6 Useable sensor types and measurement accuracy

This section describes the suitable sensors and explains the measurement accuracy of the sensor inputs of the module.

HINT: Use our free software RESI MODBUSConfigurator to configure and test our 8RTD module. You can also use your own software to handle the complete configuration while writing to MODBUS/RTU registers or with ASCII text commands.

22.6.1.1 Useable sensor types

The following types of sensors can be used per input:

Platin sensors:

- PT-100 sensors: Measurement range from 1.95Ω to 34.5Ω, -200°C to +850°C
- PT-1000 sensors: Measurement range from 195Ω to 3450Ω, -200°C to +850°C
- PT-1000 sensors with an α=0.00375: Measurement range from 195Ω to 3450Ω, -200°C to +850°C
- PT-10 sensors: Measurement range from 1.95Ω to 34.5Ω, -200°C to +850°C
- PT-50 sensors: Measurement range from 9.75Ω to 172.5Ω, -200°C to +850°C
- PT-200 sensors: Measurement range from 39Ω to 690Ω, -200°C to +850°C
- PT-500 sensors: Measurement range from 97.5Ω to 1725Ω, -200°C to +850°C

Nickel sensors:

- NI-120 sensors: Measurement range from 66.6Ω to 380.3Ω, -80°C to +260°C
- NI-1000 DIN43760 sensors: Sensors with linearisation according to DIN43760

Each of the two sensor inputs of the module can measure a different sensor type!

You can use all sensor accuracy classes (class AA, A, B, C). Please consult the DIN EN 60751:2009-05 for an exact definition of the sensor accuracy. Don’t forget, that the whole measurement error for the temperature measurement consists always out of the error of the sensor element itself, the error of the used cabling and the measurement errors of the measurement electronic.

Out resistance measurement electronic uses an internal 2kΩ sense resistor. With an excitation current of 500µA the voltage drop on this resistor is 1V. This is the ideal range, to achieve the highest measurement accuracy. Use sensor type PT100, PT200, PT500, PT-1000, NI-120 or NI-1000 DIN43760 to achieve the best accuracy of our module with +/-0.1°C.

For PT10 and PT50 sensors this internal sense resistor is too big. So the reachable accuracy lies only about +/-3°C.

22.6.1.2 Configurable excitation current

For each input you can define an individual excitation current for the measurement:

- 5µA
- 10µA
- 25µA
- 50µA
- 100µA
- 250µA
- 500µA
- 1mA

The electronic executes an internal reference measurement on an Rsense resistor with 2kΩ (Accuracy +/-0.05%). Please adjust the excitation current for each channel in a way, that the resulting maximum voltage drop on this internal Rsense resistor <=1.0V.

U=R*I -> U=2kΩ*500µA -> 1V

This results in a maximum excitation current of 500µA with this module. If the excitation current exceeds this voltage range, the module signals this error with „ADC-Out-of-Range“ in the status flags of each channel.

The ideal excitation current of the module is 500µA! With smaller excitation currents the measurement will be more and more inaccurate!

22.6.1.3 Selectable linearisation standard

A PLATIN resistor (PT sensor) is defined with a standardized characteristic. This is the Callendar-Van Dusen equation:

This is defined as follows:

RT = R0 • (1 + a • T + b • T2 + (T – 100°C) • c • T3) for T < 0°C,
RT = R0 • (1 + a • T + b • T2) for T > 0°C

The equation is used with different coefficients depending of the selected linearisation standard to calculate a temperature from the measured resistor.

STANDARD	ALPHA (α)	a	b	c
Europe DIN EN 60751 IEC 751 JIS C1604-1997	α=0x00385	$3.908300*10^{-03}$	$-5.775000*10^{-07}$	$-4.183000*10^{-12}$
America SAMA Standard	α=0x003911	$3.969200*10^{-03}$	$-5.849500*10^{-07}$	$-4.232500*10^{-12}$
Japan JIS C1604-1987	α=0x003916	$3.973900*10^{-03}$	$-5.870000*10^{-07}$	$-4.400000*10^{-12}$
ITS-90	α=0x003926	$3.984800*10^{-03}$	$-5.870000*10^{-07}$	$-4.400000*10^{-12}$
RTD-1000-375	α=0x00375	$3.810200*10^{-03}$	$-6.018880*10^{-07}$	$-6.000000*10^{-12}$
NI-120	N/A	N/A	N/A	N/A

22.6.1.4 Sensor evaluation and accuracy

Our module computes the final temperature value °Celsius [°C] and delivers this temperature on various MODBUS registers in various number formats and via various ASCII commands to the host.

In addition our module can convert the temperature also in °Fahrenheit [°F] with the formula:

T[°F]=T[°C]*1.8+32

Also this temperature value can be read out with MODBUS Register or ASCII text commands. An own conversion on the host from Celsius in Fahrenheit is not necessary.

Also our module converts the temperature data into °Kelvin [°K] with the formula:

T[°K]=T[°C] +273.15

Also this temperature value can be read out with MODBUS Register or ASCII text commands. An own conversion on the host from Celsius in kelvin is not necessary.

Our module uses a 24 bit sigma/delta ADC with a noise suppression for 50/60Hz internally. Our module achieves a very high measurement accuracy of +/-0.1°C and a measurement resolution of +/-0.001°C!

Our module measures every channel around 1 time per second. In addition our module computes an average temperature for each channel with a user selectable time range in seconds, to suppress short noise signals in standard applications.

A manual adjustable zero offset allows a zero point shift to compensate static effects of the cabling, especially useful for 2 wire sensors.

Our module offers a very complex internal hardware to evaluate if the measured temperature is valid or not. Therefore the module offers for each channel a status representing the result of the last converted temperature. This status uses 8 bits, which have the following meaning:

BIT	NAME	DESCRIPTION
0	VALID	=1: If the measurement result is valid, this bit is set and all other bits in the status are 0! =0: if the system detects a conversion error or problem, this bit is 0 and the measurement result must be discarded!
1	ADC OUT OF RANGE	=1: If the product of 2kΩ * excitation current >1V, this bit is 1 and the measurement result is invalid. The absolute input voltage of the ACD beyond ±1.125 • VREF/2 =0: Everything is ok
2	SENSOR UNDER RANGE	=1: The current measured temperature is beyond the lower limit for the selected sensor type. For PT: -200°C, for NI-120: -80°C =0: Everything is ok
3	SENSOR OVER RANGE	=1: The current measured temperature is above the upper limit for the selected sensor type. For PT: +850°C, for NI-120: +260°C =0: Everything is ok
4	NOT USED	Ignore this bit
5	NOT USED	Ignore this bit
6	HARD ADC OUT OF RANGE	=1: Erroneous readout of the ADC value. A possibility is an extreme high noise level on the signal. The sensor value will be discarded. A second option is an open wiring for the sensor. =0: Everything is ok
7	SENSOR HARD FAULT	=1: Sensor wiring is open or no sensor is cabled to the module. Sensor has a shortcut or the internal sense resistor has an error. =0: Everything is ok

22.7 Additional MODBUS register & coils

Here you will find only the additional MODBUS registers and coils especially for this IO module. Please refer to the description of of the standard MODBUS mapping for more details about the available basic MODBUS registers and coils.

Please refer to the external document for detailed documentation of the current MODBUS register mapping for this IO module:

RESI-L-8RTD,8RTD2-SIO-MODBUS+ASCII-ENxx.pdf

22.8 Additional ASCII commands

Here you will find only the additional ASCII commands especially for this IO module. Please refer to the description of of the standard commands for more details about the available basic ASCII commands.

Please refer to the external document for detailed documentation of the current ASCII commands for this IO module:

RESI-L-8RTD,8RTD2-SIO-MODBUS+ASCII-ENxx.pdf

23 RESI-8RTD2-SIO

23.1 General information

This series of IO modules offer the following features:

- 8 sensor inputs for temperature sensors
- Measurement accuracy +/-0.1%
- Measurement resolution +/-0.001%
- Measurement range -200°C…+850°C
- Various sensor types are applicable: PT100, PT1000, PT10, PT50, PT200, PT500, NI120, NI1000-DIN43760
- Various standards for linearisation are select-able: Europa, America, Japan, ITS-90
- Output of the temperatures in °Celsius [°C], °Fahrenheit [°F] or °Kelvin [°K]
- Different measurement currents are select-able: 5µA, 10µA, 25µA, 50µA, 100µA, 250µA,500µA, 1mA
- Various sensor connection types: 2 wire sensors connectable
- Internal calculation of an average temperature per channel
- Galvanic insulated RS485 interface for communication with a host system

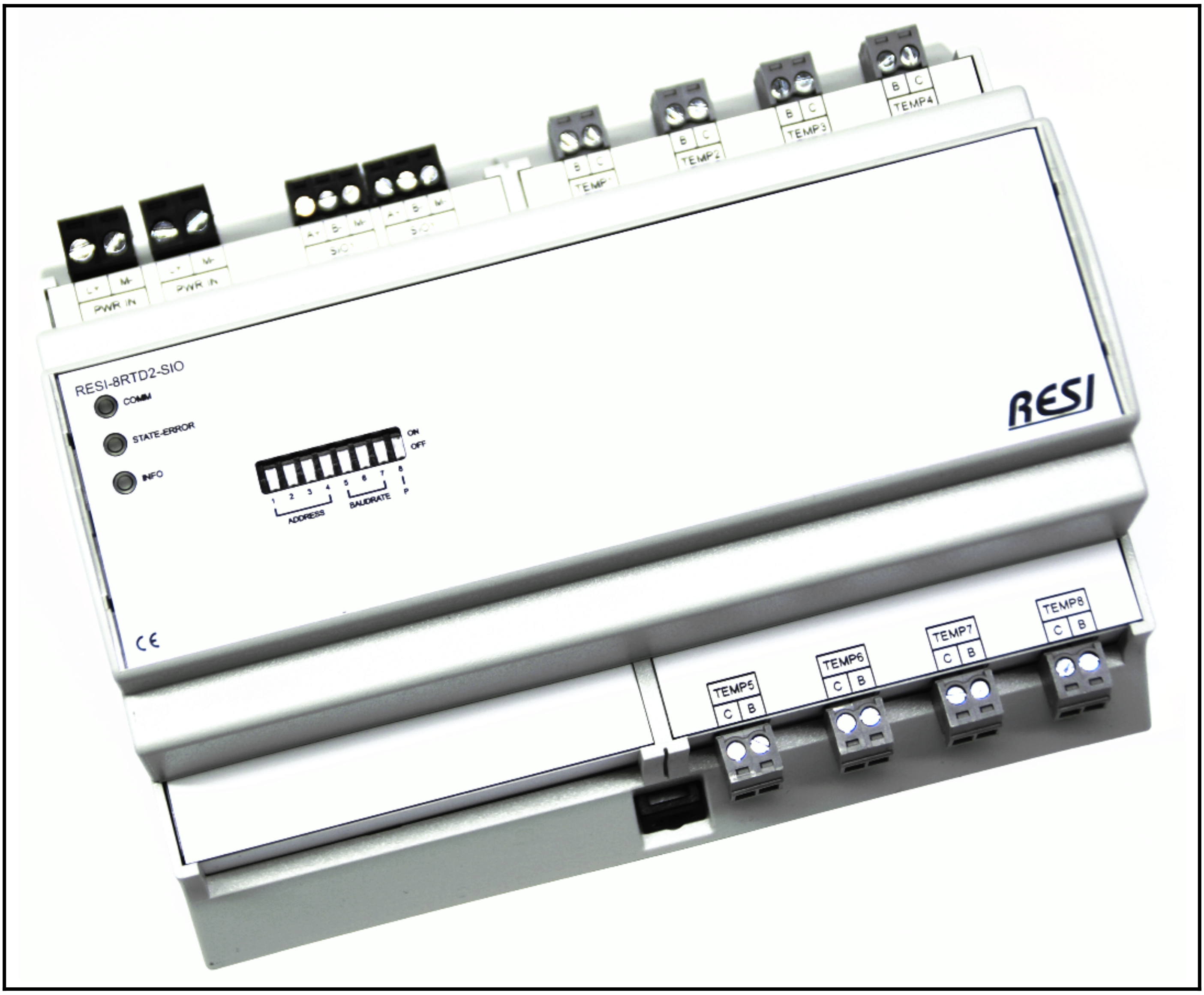


Figure: Our RESI-8RTD2-SIO module

23.2 Technical specification

Beside the basic technical data, which fulfil all of our IO modules, this IO modules meet the following technical specifications:

Power consumption	<0.8W
Product housing	BIG IO XT8
Product weight	290g
Temperature inputs	
Number	8
Signal type	Temperature measurement
Measurement type	Measurement of resistance
Sensor connection	2 wire measurement
ADC	24 bit sigma delta ADC
Accuracy	+/-0.1°C for PT-100, PT-200,PT-500, PT-1000
	+/-0.1°C NI-120, NI-1000-DIN43760
	+/-3°C for PT-10, PT-50
Resolution	+/-0.001°C
Reference stability	10ppm/°C
Sensor types	PT-100, PT-1000, PT-1000 α=0.00375, PT-10, PT-50, PT-200,
	PT-500, NI-120, NI-1000 DIN43760
Linearisation standards	Europa, America, Japan, ITS-90
Excitation current for measurement	5µA, 10µA, 25µA, 50µA, 100µA, 250µA, 500µA, 1mA
Cable connection	Via 8 2-pin plug-in terminal block
Terminal type	RM3.5
Galvanic isolation	Yes, to the rest of the module, not to the other temperature inputs
Default serial settings	
Baud rate	via DIP switch
Parity	none
Stop bit(s)	one
UnitID	255

23.3 Additional terminals & LED states

TEMPERATURE INPUTS	8 temperature inputs for RTD temperature sensors	
	Eight 4 pin plug-in terminal block	
	Terminal type:	RM3.5
	TEMPx:A:	Temperature Sensor Cable 1A
	TEMPx:B:	Temperature Sensor Cable 1B
	TEMPx:C:	Temperature Sensor Cable 2A
	TEMPx:D:	Temperature Sensor Cable 2B
Connecting 2-wire temperature sensors:		
Pin layout	Pin 1:	A: Wire 1 of sensor
	Pin 2:	B: Wire 2 of Sensor
Connecting 3-wire temperature sensors:		
Pin layout	Pin 1:	A: Wire 1 and wire 2 of Sensor
	Pin 2:	B: Wire 3 of Sensor
Connecting 4-wire temperature sensors:		
Pin layout	Pin 1:	A: Wire 1 and Wire 2 of Sensor
	Pin 2:	B: Wire 3 and wire 4 of Sensor
INFO	If everything is ok this LED is on. If there is an internal error	
	with the temperature measurement, this LED flashes fast.	

23.4 Schematic diagram

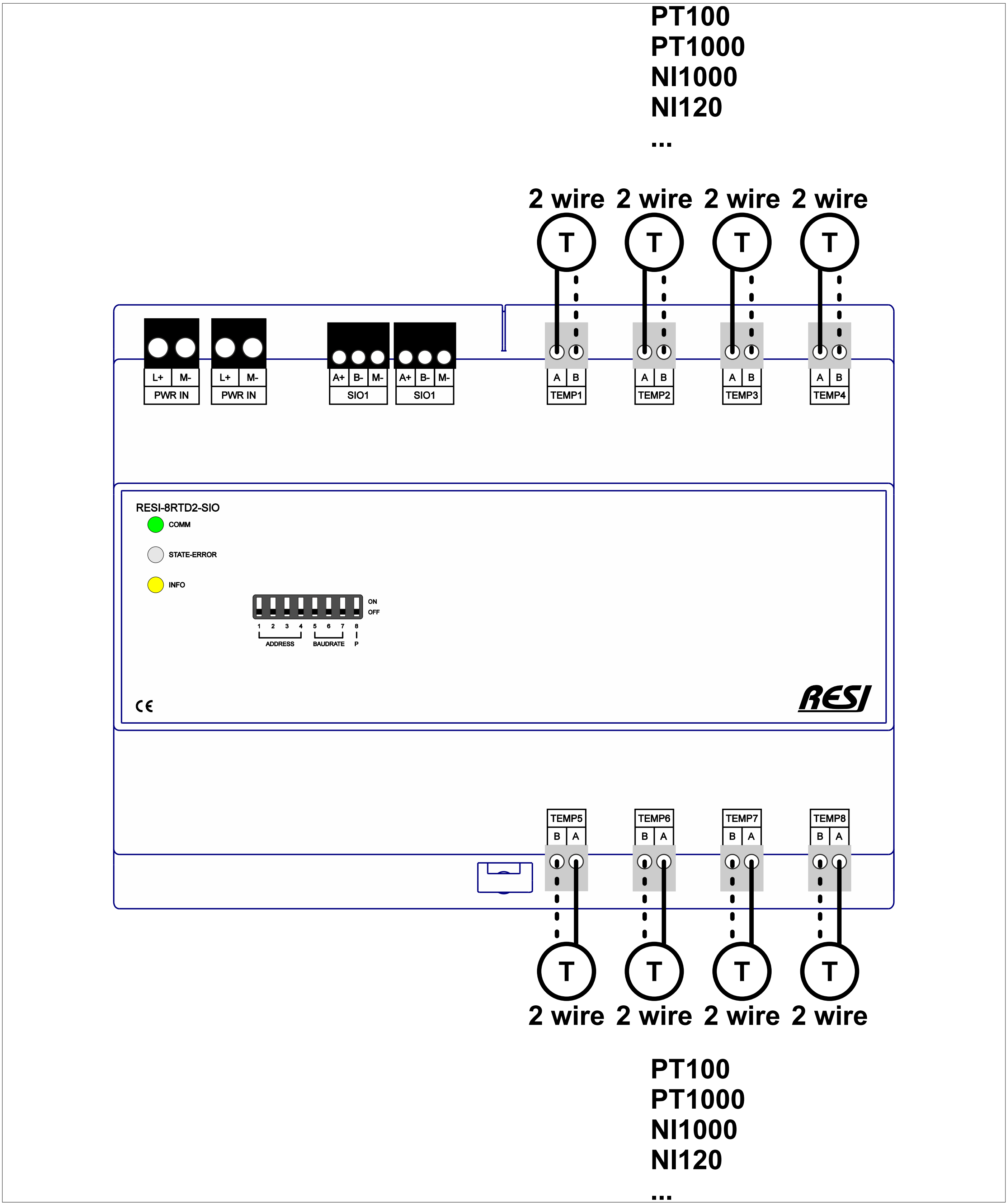


Figure: Schematic diagram of IO module

23.5 Cabling of temperature sensors

A typical temperature sensor with different connection cables is shown in the figure below:

- 2 wire: A red and white cable
- 3 wire: Two red and one white cable
- 4 wire: Two read and two white cable

The sensor element is always mounted between the red and white cables!

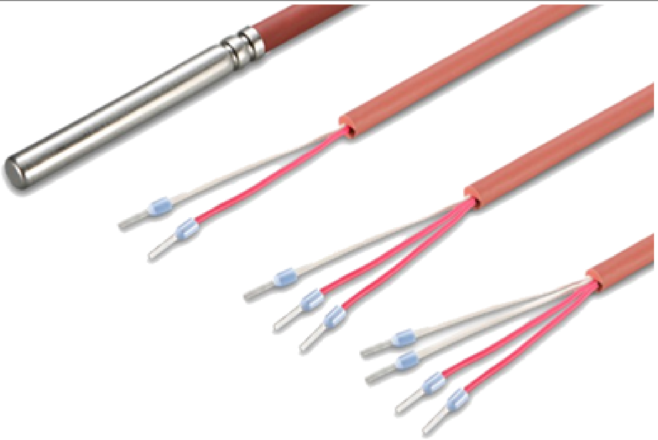


Figure: Typical temperature sensor with different connection cables

23.5.1 Cabling of 2-wire temperature sensors

In the below drawing you see the cabling of 2-wire temperature sensors:

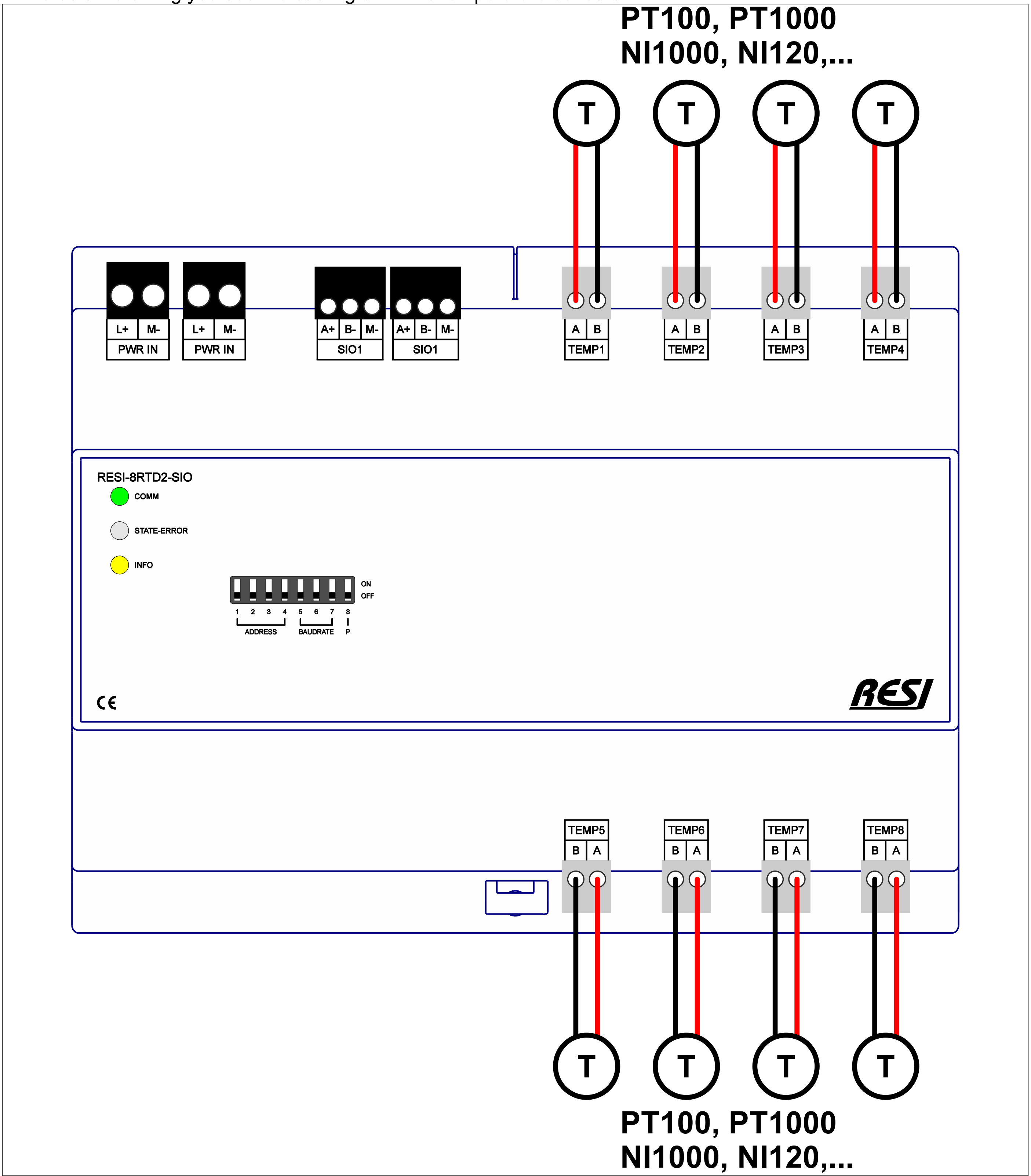


Figure: Cabling of temperature sensors with 2 wires

Don't forget that you can mix different types of sensor elements on each channel!

23.5.2 Cabling of 3-wire temperature sensors

In the below drawing you see the cabling of 3-wire temperature sensors:

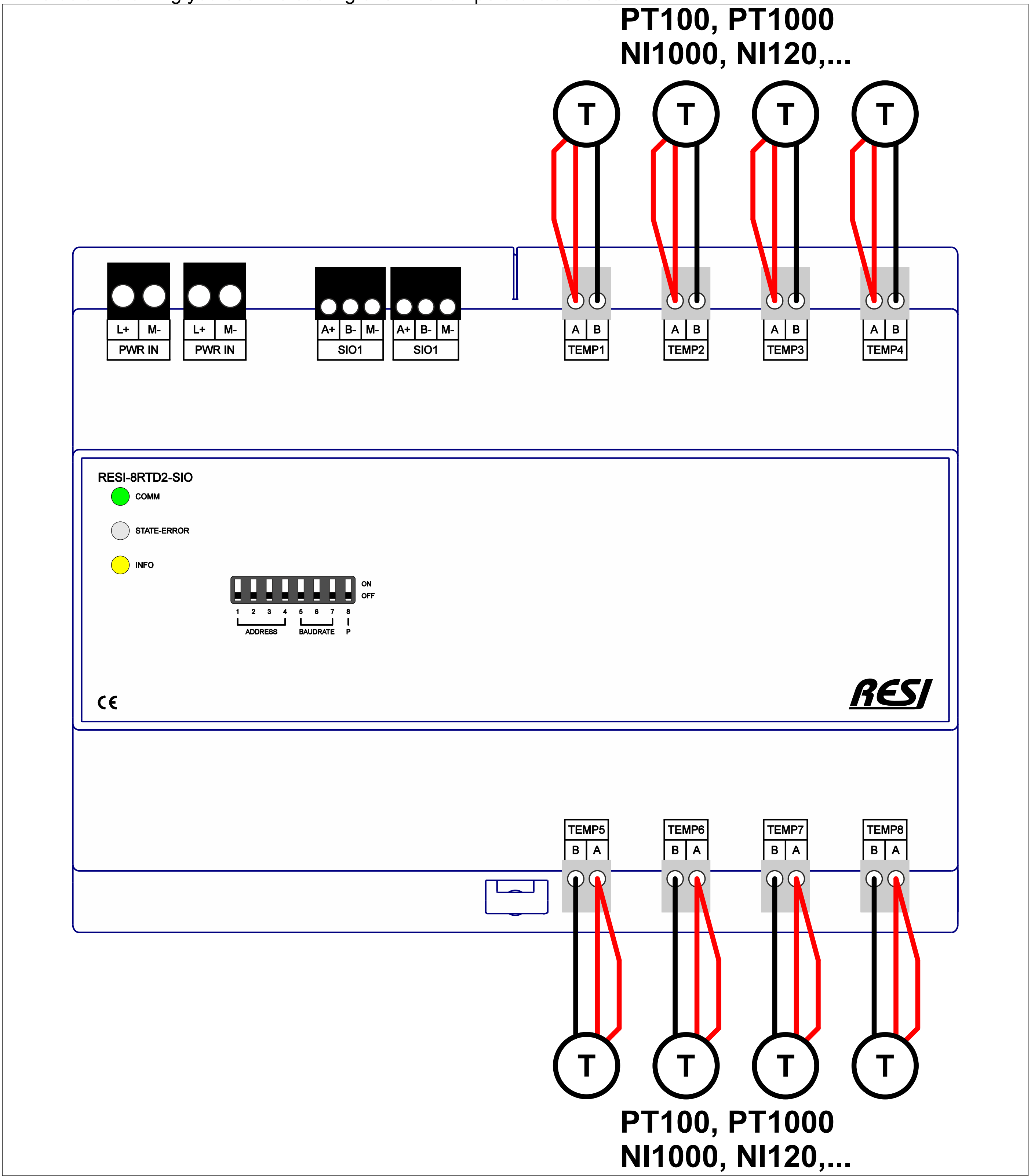


Figure: Cabling of temperature sensors with 3 wires

Don't forget that you can mix different types of sensor elements on each channel!

23.5.3 Cabling of 4-wire temperature sensors

In the below drawing you see the cabling of 2-wire temperature sensors:

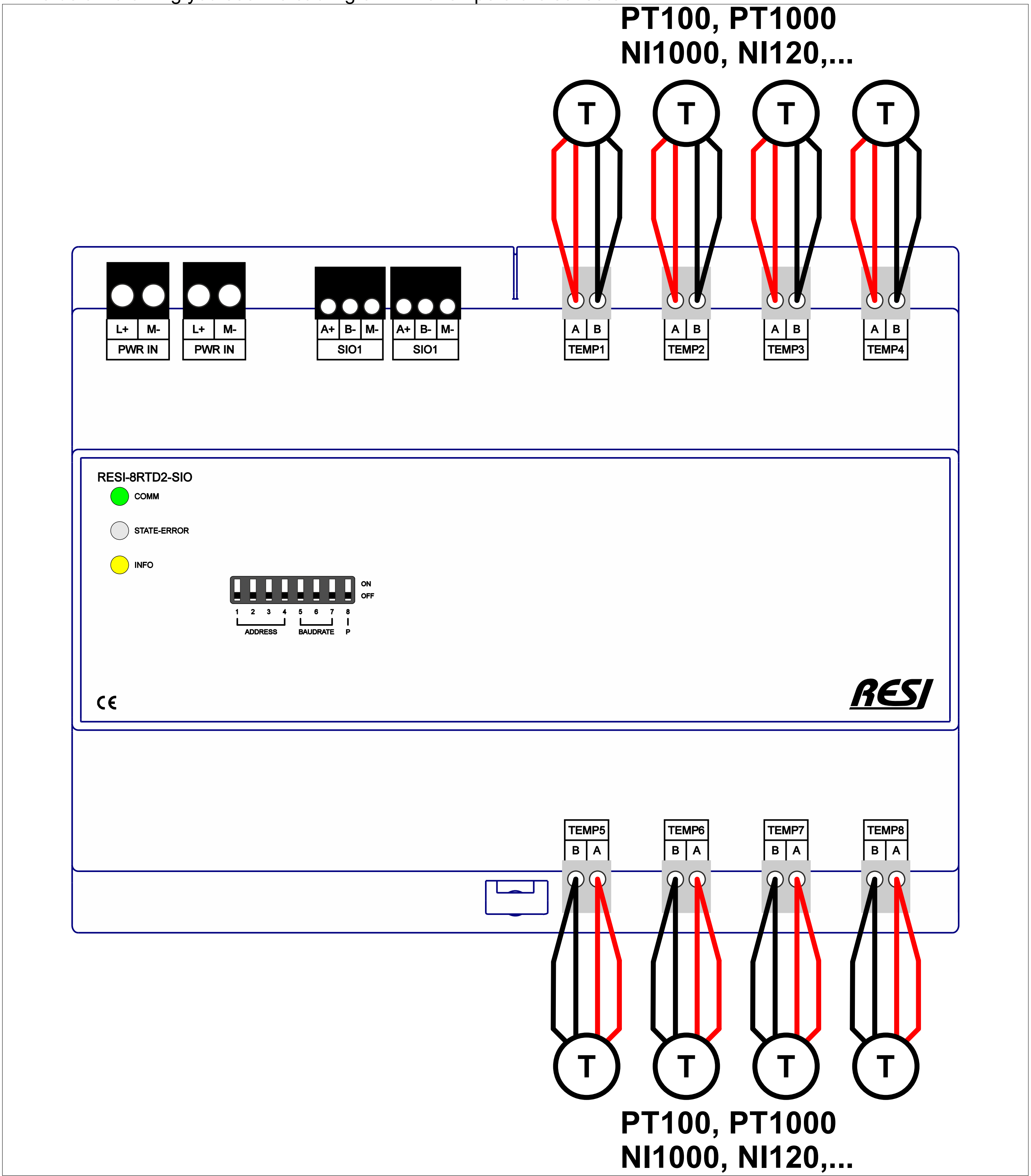


Figure: Cabling of temperature sensors with 4 wires

Don't forget that you can mix different types of sensor elements on each channel!

23.6 Useable sensor types and measurement accuracy

This section describes the suitable sensors and explains the measurement accuracy of the sensor inputs of the module.

HINT: Use our free software RESI MODBUSConfigurator to configure and test our 8RTD2 module. You can also use your own software to handle the complete configuration while writing to MODBUS/RTU registers or with ASCII text commands.

23.6.1.1 Useable sensor types

The following types of sensors can be used per input:

Platin sensors:

- PT-100 sensors: Measurement range from 1.95Ω to 34.5Ω, -200°C to +850°C
- PT-1000 sensors: Measurement range from 195Ω to 3450Ω, -200°C to +850°C
- PT-1000 sensors with an α=0.00375: Measurement range from 195Ω to 3450Ω, -200°C to +850°C
- PT-10 sensors: Measurement range from 1.95Ω to 34.5Ω, -200°C to +850°C
- PT-50 sensors: Measurement range from 9.75Ω to 172.5Ω, -200°C to +850°C
- PT-200 sensors: Measurement range from 39Ω to 690Ω, -200°C to +850°C
- PT-500 sensors: Measurement range from 97.5Ω to 1725Ω, -200°C to +850°C

Nickel sensors:

- NI-120 sensors: Measurement range from 66.6Ω to 380.3Ω, -80°C to +260°C
- NI-1000 DIN43760 sensors: Sensors with linearisation according to DIN43760

Each of the two sensor inputs of the module can measure a different sensor type!

You can use all sensor accuracy classes (class AA, A, B, C). Please consult the DIN EN 60751:2009-05 for an exact definition of the sensor accuracy. Don’t forget, that the whole measurement error for the temperature measurement consists always out of the error of the sensor element itself, the error of the used cabling and the measurement errors of the measurement electronic.

Out resistance measurement electronic uses an internal 2kΩ sense resistor. With an excitation current of 500µA the voltage drop on this resistor is 1V. This is the ideal range, to achieve the highest measurement accuracy. Use sensor type PT100, PT200, PT500, PT-1000, NI-120 or NI-1000 DIN43760 to achieve the best accuracy of our module with +/-0.1°C.

For PT10 and PT50 sensors this internal sense resistor is too big. So the reachable accuracy lies only about +/-3°C.

23.6.1.2 Configurable excitation current

For each input you can define an individual excitation current for the measurement:

- 5µA
- 10µA
- 25µA
- 50µA
- 100µA
- 250µA
- 500µA
- 1mA

The electronic executes an internal reference measurement on an Rsense resistor with 2kΩ (Accuracy +/-0.05%). Please adjust the excitation current for each channel in a way, that the resulting maximum voltage drop on this internal Rsense resistor <=1.0V.

U=R*I -> U=2kΩ*500µA -> 1V

This results in a maximum excitation current of 500µA with this module. If the excitation current exceeds this voltage range, the module signals this error with „ADC-Out-of-Range“ in the status flags of each channel.

The ideal excitation current of the module is 500µA! With smaller excitation currents the measurement will be more and more inaccurate!

23.6.1.3 Selectable linearisation standard

A PLATIN resistor (PT sensor) is defined with a standardized characteristic. This is the Callendar-Van Dusen equation:

This is defined as follows:

RT = R0 • (1 + a • T + b • T2 + (T – 100°C) • c • T3) for T < 0°C,
RT = R0 • (1 + a • T + b • T2) for T > 0°C

The equation is used with different coefficients depending of the selected linearisation standard to calculate a temperature from the measured resistor.

STANDARD	ALPHA (α)	a	b	c
Europe DIN EN 60751 IEC 751 JIS C1604-1997	α=0x00385	$3.908300*10^{-03}$	$–5.775000*10^{-07}$	$–4.183000*10^{-12}$
America SAMA Standard	α=0x003911	$3.969200*10^{-03}$	$–5.849500*10^{-07}$	$–4.232500*10^{-12}$
Japan JIS C1604-1987	α=0x003916	$3.973900*10^{-03}$	$–5.870000*10^{-07}$	$–4.400000*10^{-12}$
ITS-90	α=0x003926	$3.984800*10^{-03}$	$–5.870000*10^{-07}$	$–4.400000*10^{-12}$
RTD-1000-375	α=0x00375	$3.810200*10^{-03}$	$–6.018880*10^{-07}$	$–6.000000*10^{-12}$
NI-120	N/A	N/A	N/A	N/A

23.6.1.4 Sensor evaluation and accuracy

Our module computes the final temperature value °Celsius [°C] and delivers this temperature on various MODBUS registers in various number formats and via various ASCII commands to the host.

In addition our module can convert the temperature also in °Fahrenheit [°F] with the formula:

T[°F]=T[°C]*1.8+32

Also this temperature value can be read out with MODBUS Register or ASCII text commands. An own conversion on the host from Celsius in Fahrenheit is not necessary.

Also our module converts the temperature data into °Kelvin [°K] with the formula:

T[°K]=T[°C] +273.15

Also this temperature value can be read out with MODBUS Register or ASCII text commands. An own conversion on the host from Celsius in kelvin is not necessary.

Our module uses a 24 bit sigma/delta ADC with a noise suppression for 50/60Hz internally. Our module achieves a very high measurement accuracy of +/-0.1°C and a measurement resolution of +/-0.001°C!

Our module measures every channel around 1 time per second. In addition our module computes an average temperature for each channel with a user selectable time range in seconds, to suppress short noise signals in standard applications.

A manual adjustable zero offset allows a zero point shift to compensate static effects of the cabling, especially useful for 2 wire sensors.

Our module offers a very complex internal hardware to evaluate if the measured temperature is valid or not. Therefore the module offers for each channel a status representing the result of the last converted temperature. This status uses 8 bits, which have the following meaning:

BIT	NAME	DESCRIPTION
0	VALID	=1: If the measurement result is valid, this bit is set and all other bits in the status are 0! =0: if the system detects a conversion error or problem, this bit is 0 and the measurement result must be discarded!
1	ADC OUT OF RANGE	=1: If the product of 2kΩ * excitation current >1V, this bit is 1 and the measurement result is invalid. The absolute input voltage of the ACD beyond ±1.125 • VREF/2 =0: Everything is ok
2	SENSOR UNDER RANGE	=1: The current measured temperature is beyond the lower limit for the selected sensor type. For PT: -200°C, for NI-120: -80°C =0: Everything is ok
3	SENSOR OVER RANGE	=1: The current measured temperature is above the upper limit for the selected sensor type. For PT: +850°C, for NI-120: +260°C =0: Everything is ok
4	NOT USED	Ignore this bit
5	NOT USED	Ignore this bit
6	HARD ADC OUT OF RANGE	=1: Erroneous readout of the ADC value. A possibility is an extreme high noise level on the signal. The sensor value will be discarded. A second option is an open wiring for the sensor. =0: Everything is ok
7	SENSOR HARD FAULT	=1: Sensor wiring is open or no sensor is cabled to the module. Sensor has a shortcut or the internal sense resistor has an error. =0: Everything is ok

23.7 Additional MODBUS register & coils

Here you will find only the additional MODBUS registers and coils especially for this IO module. Please refer to the description of of the standard MODBUS mapping for more details about the available basic MODBUS registers and coils.

Please refer to the external document for detailed documentation of the current MODBUS register mapping for this IO module:

RESI-L-8RTD,8RTD2-SIO-MODBUS+ASCII-ENxx.pdf

23.8 Additional ASCII commands

Here you will find only the additional ASCII commands especially for this IO module. Please refer to the description of of the standard commands for more details about the available basic ASCII commands.

Please refer to the external document for detailed documentation of the current ASCII commands for this IO module:

RESI-L-8RTD,8RTD2-SIO-MODBUS+ASCII-ENxx.pdf

24 RESI-12AIU-SIO, RESI-12AIU-ETH

24.1 General information

This series of IO modules offer the following features:

- 12 high precision analog inputs for -10Vdc..+10Vdc signals (-10.24Vdc to +10.24Vdc)
- ADC resolution 16 bit, accuracy +/-0.1%
- RESI-xxx-SIO: Galvanic isolated RS485 interface for communication with a host system
- RESI-xxx-ETH: Galvanic isolated Ethernet interface for communication with a host system

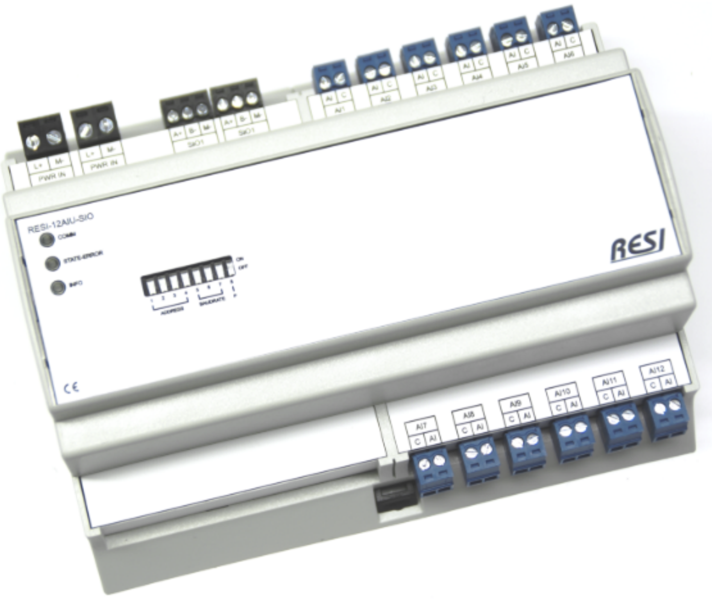


Figure: Our serial IO module

24.2 Technical specification

Beside the basic technical data, which fulfil all of our IO modules, this IO modules meet the following technical specifications:

Power consumption	
RESI-12AIU-SIO	<0.8W
RESI-12AIU-ETH	<1.2W
Product housing	
RESI-12AIU-SIO	BIG IO XT8
RESI-12AIU-ETH	BIG IO XT8
Product weight	
RESI-12AIU-SIO	285g
RESI-12AIU-ETH	290g
Analog inputs	
Number	12
Update speed	Every 100ms
Range	-10V..+10V
ADC resolution	16 bit
Input voltage range	-10.24V..+10.24V
Accuracy	+/-0.1%
Cable connection	via terminals
Galvanic isolation	Yes, to the rest of the module, not to the other analogue inputs
Default serial settings	
Baud rate	via DIP switch
Parity	none
Stopbits	one
UnitID	255
Default Ethernet settings	
IP address	tbd
IP mask	255.255.255.0
gateway	tbd
UnitID	255
User	RESI
password	RESI

24.3 Additional terminals & LED states

ANALOG INPUTS	12 analog inputs for -10V..0V..+10V signals	
	Twelve 2 pin plug-in terminal blocks	
	Terminal type:	RM3.5
	C:	Ground for all analog inputs
	AI1-AI12 AI:	Analog inputs
Pin layout	AI:	Signal input for analog input #x
	C:	Signal ground for analog input #x
		All signal grounds are internally bridged
ERROR	If everything is OK, this LED is off. If there is an internal error at the analog inputs,	
	this LED flashes quickly in RED.	

24.4 RESI-12AIU-SIO,ETH: Schematic diagram

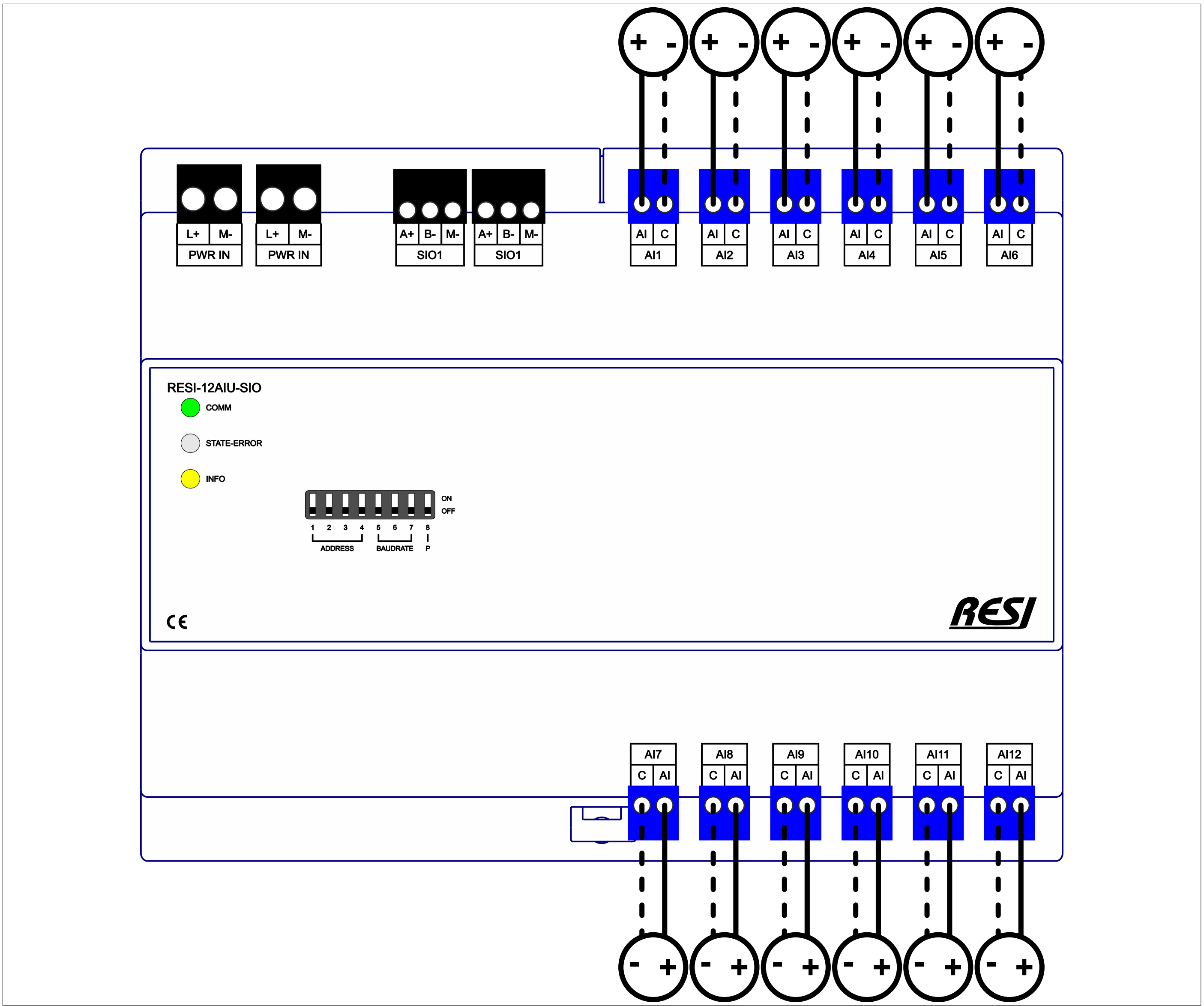


Figure: Schematics for the IO modules

24.5 RESI-12AIU-SIO,ETH: Wiring diagram

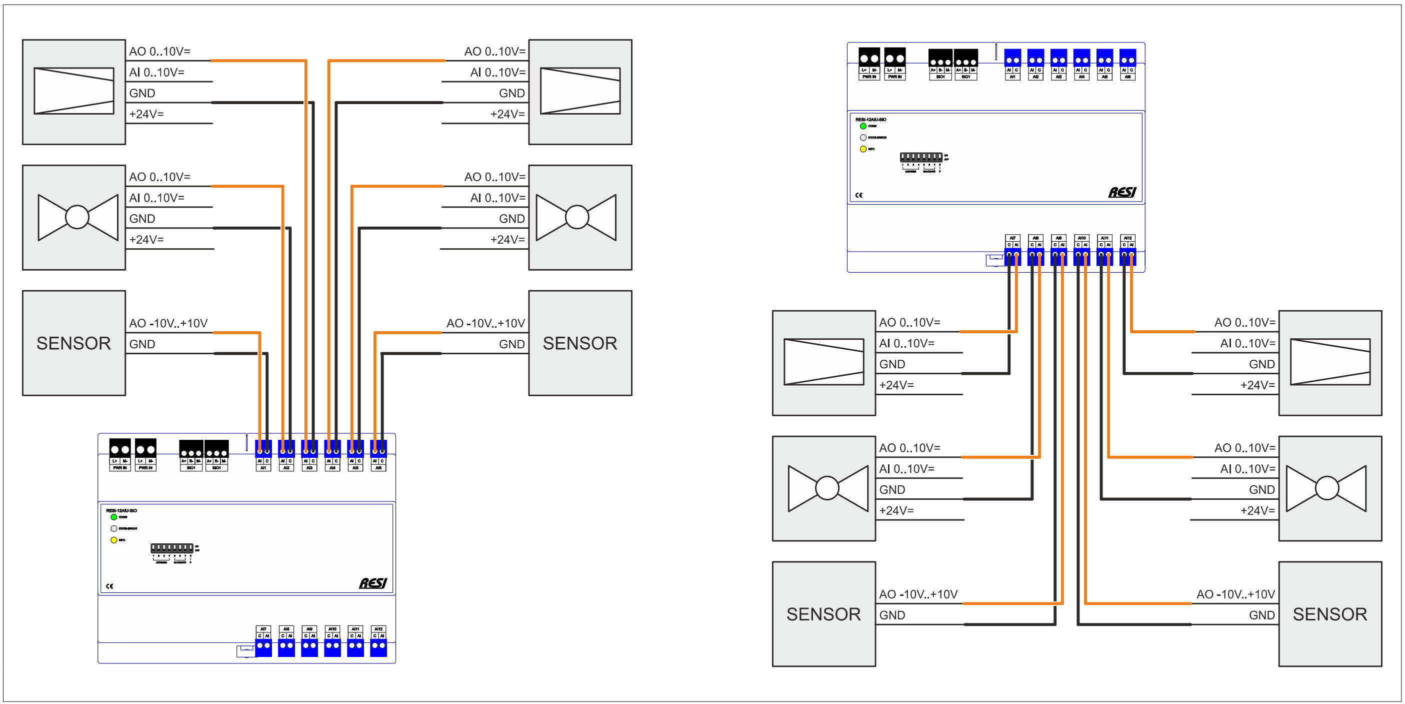


Figure: Wiring diagram for the IO modules

24.6 Additional MODBUS register & coils

Here you will find only the additional MODBUS registers and coils especially for this IO module. Please refer to the description of of the standard MODBUS mapping for more details about the available basic MODBUS registers and coils.

Please refer to the external document for detailed documentation of the current MODBUS register mapping for this IO module:

RESI-L-12AIU-SIO-ETH-MODBUS+ASCII-ENxx.pdf

24.7 Additional ASCII commands

Here you will find only the additional ASCII commands especially for this IO module. Please refer to the description of of the standard commands for more details about the available basic ASCII commands.

Please refer to the external document for detailed documentation of the current ASCII commands for this IO module:

RESI-L-12AIU-SIO-ETH-MODBUS+ASCII-ENxx.pdf

25 RESI-12AOU-SIO, RESI-12AOU-ETH

25.1 General information

This series of IO modules offer the following features:

- 12 high precision analog outputs for -10Vdc..+10Vdc signals
- ADC resolution 12 bit, accuracy +/-0.1%
- RESI-xxx-SIO: Galvanic isolated RS485 interface for communication with a host system
- RESI-xxx-ETH: Galvanic isolated Ethernet interface for communication with a host system

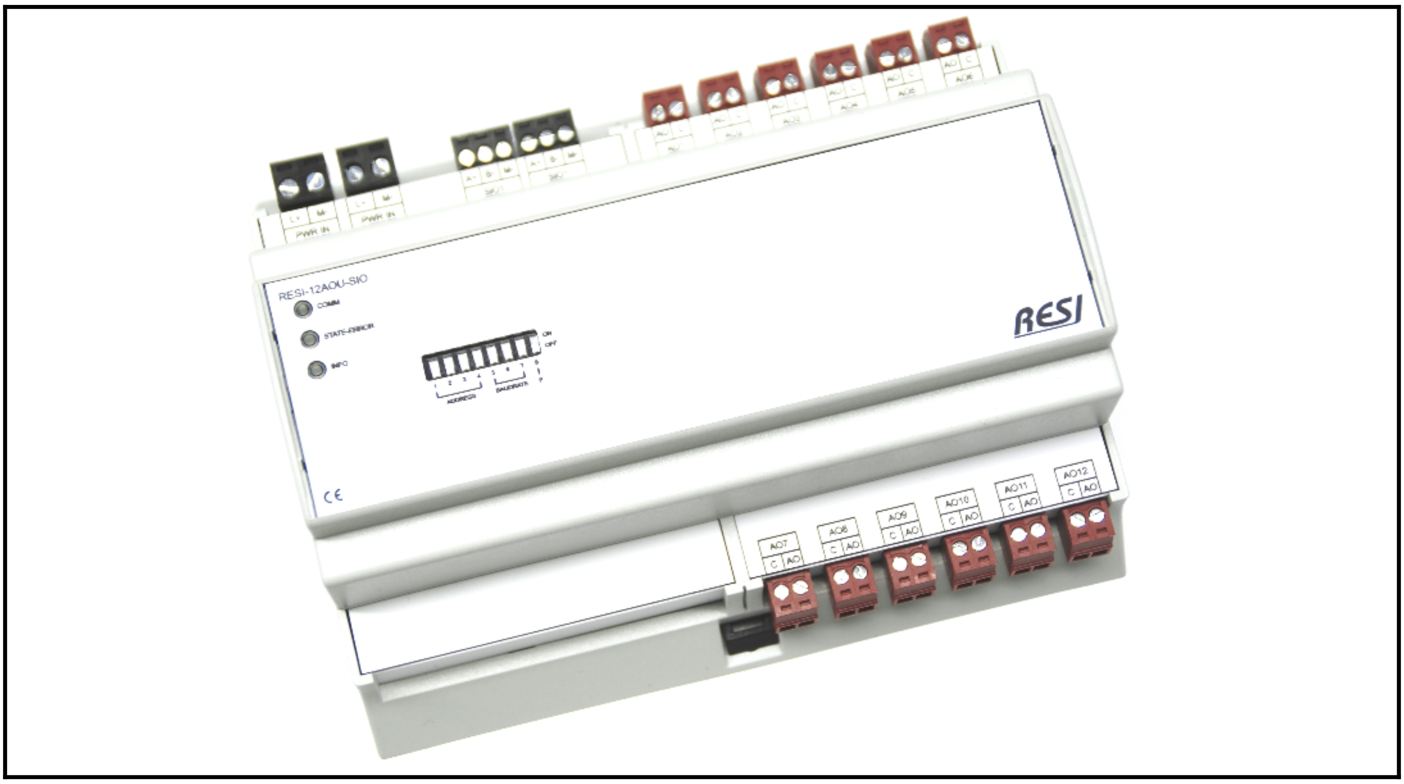


Figure: Our serial IO module

25.2 Technical specification

Beside the basic technical data, which fulfil all of our IO modules, this IO modules meet the following technical specifications:

Power consumption	
RESI-12AOU-SIO	<1.1W
RESI-12AOU-ETH	<1.5W
Product housing	
RESI-12AOU-SIO	BIG IO XT8
RESI-12AOU-ETH	BIG IO XT8
Product weight	
RESI-4AOU-SIO	285g
RESI-4AOU-ETH	290g
Analog outputs	
Number	12
Update speed	Every 100ms
Range	-10V..+10V
ADC resolution	12 bit
Output voltage range	-10V..+10V
Accuracy	+/-0.1%
Cable connection	via terminals
Galvanic isolation	Yes, to the rest of the module, not to the other analogue inputs
Default serial settings	
Baud rate	via DIP switch
Parity	none
Stopbits	one
UnitID	255
Default Ethernet settings	
IP address	tbd
IP mask	255.255.255.0
gateway	tbd
UnitID	255
User	RESI
password	RESI

25.3 Additional terminals & LED states

ANALOG OUTPUTS	12 analog outputs for -10V..0V..+10V signals	
	Twelve 2 pin plug-in terminal blocks	
	Terminal type:	RM3.5
	C:	Ground for all analog inputs
	AO1-AO12:	Analog outputs
Pin layout	AO:	Signal output for analog output #x
	C:	Signal ground for analog output #x
		All signal grounds are internally bridged
ERROR	If everything is OK, this LED is off. If there is an internal error at the analog inputs,	
	this LED flashes quickly in RED.	

25.4 RESI-12AOU-SIO,ETH: Schematic diagram

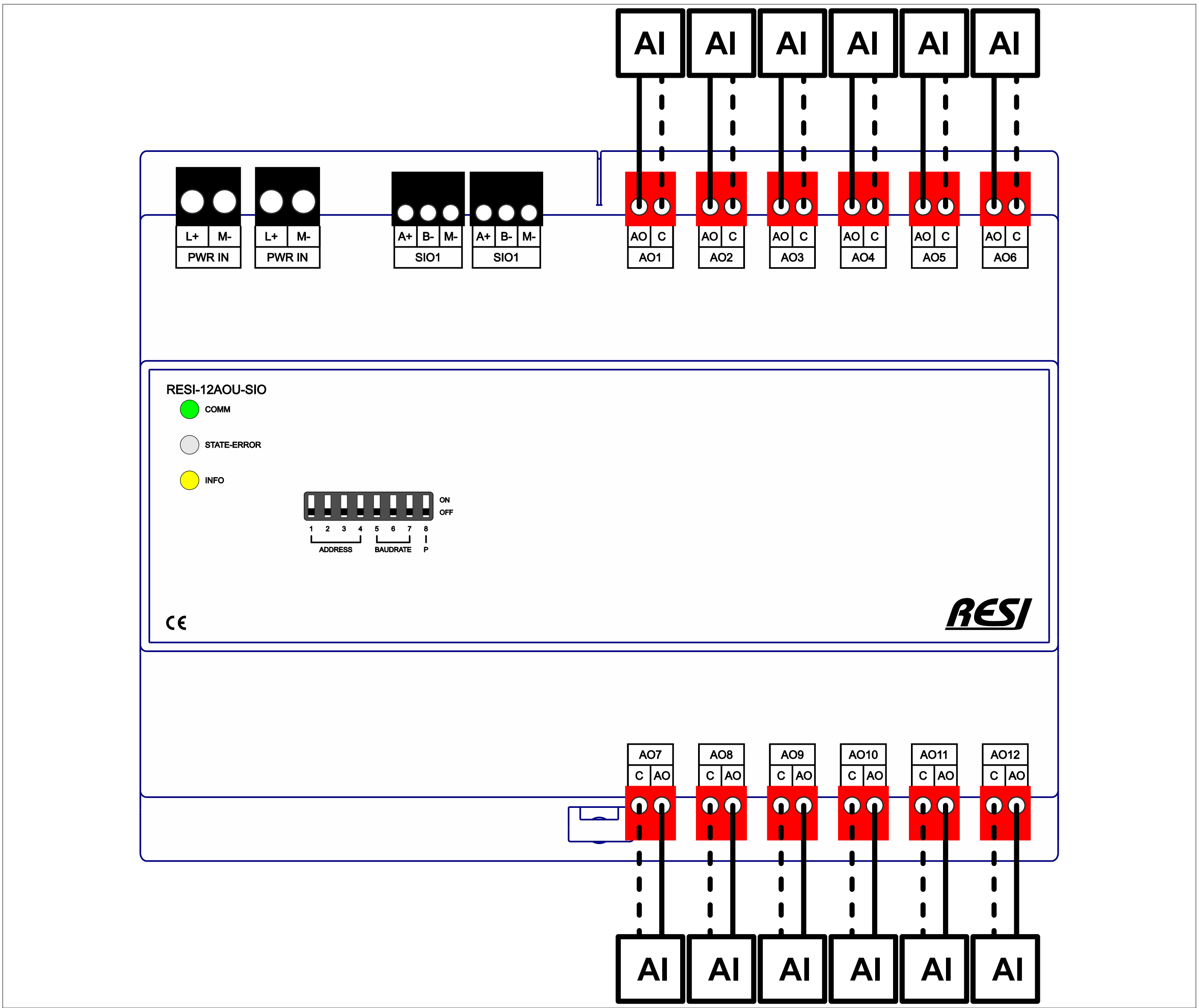


Figure: Schematics for the IO modules

25.5 RESI-12AOU-SIO,ETH: Wiring diagram

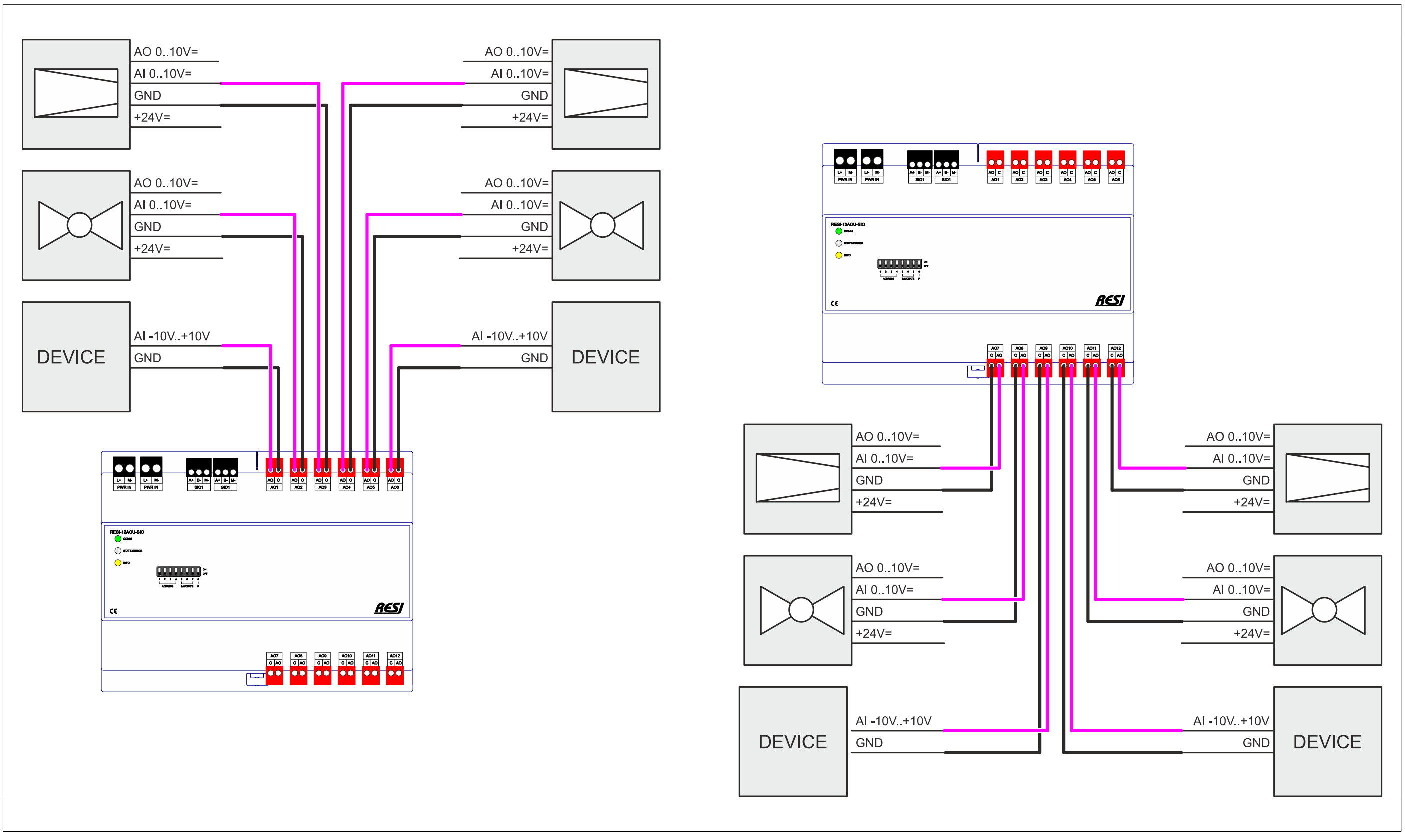


Figure: Wiring diagram for the IO modules

25.6 Additional MODBUS register & coils

Here you will find only the additional MODBUS registers and coils especially for this IO module. Please refer to the description of of the standard MODBUS mapping for more details about the available basic MODBUS registers and coils.

Please refer to the external document for detailed documentation of the current MODBUS register mapping for this IO module:

RESI-L-12AOU-SIO-ETH-MODBUS+ASCII-ENxx.pdf

25.7 Additional ASCII commands

Here you will find only the additional ASCII commands especially for this IO module. Please refer to the description of of the standard commands for more details about the available basic ASCII commands.

Please refer to the external document for detailed documentation of the current ASCII commands for this IO module:

RESI-L-12AOU-SIO-ETH-MODBUS+ASCII-ENxx.pdf

26 RESI-2RI-SIO, RESI-2RI-ETH

26.1 General information

This series of IO modules offer the following features:

- 2 digital inputs for 10…250Vac/dc signals
- Galvanic insulation between digital inputs and rest of the module
- RESI-xxx-SIO: Galvanic isolated RS232 and RS485 interface for communication with a host system
- RESI-xxx-ETH: Galvanic isolated Ethernet interface for communication with a host system

Figure: Our serial IO module

Figure: Our Ethernet IO module

26.2 Technical specification

Beside the basic technical data, which fulfil all of our IO modules, this IO modules meet the following technical specifications:

Power consumption	
RESI-2RI-SIO	<0.7W
RESI-2RI-ETH	<1.1W

Product housing	
RESI-2RI-SIO	CEM17
RESI-2RI-ETH	CEM35

Product weight	
RESI-2RI-SIO	55g
RESI-2RI-ETH	89g

Digital inputs	
Number	2
Signal type	10..250Vac/dc
Cable connection	via terminals
Galvanic isolation	Yes, between the two inputs and the CPU module
	not between the two digital inputs

Default serial settings	
Baud rate	via DIP switch
Parity	none
Stopbits	one
UnitID	255

Default Ethernet settings	
IP address	192.168.0.40
IP mask	255.255.255.0
gateway	192.168.0.1
UnitID	255

User	RESI
password	RESI

26.3 Additional terminals & LED states

DIGITAL INPUTS	2 digital inputs for 10...250Vac/dc signals	
	One 3 pin terminal block	
	Terminal type:	USLIM
	C:	Common for both digital inputs
	DI1,DI2:	Digital inputs
Pin layout	DI1:	Signal input for digital input #1
	DI2:	Signal input for digital input #2
	C:	Common for both digital inputs
INFO	If at least one of the digital inputs is ON, this LED is on.	
	If no digital input is ON, this LED is OFF too.	

26.4 RESI-2RI-SIO,ETH: Schematic diagram

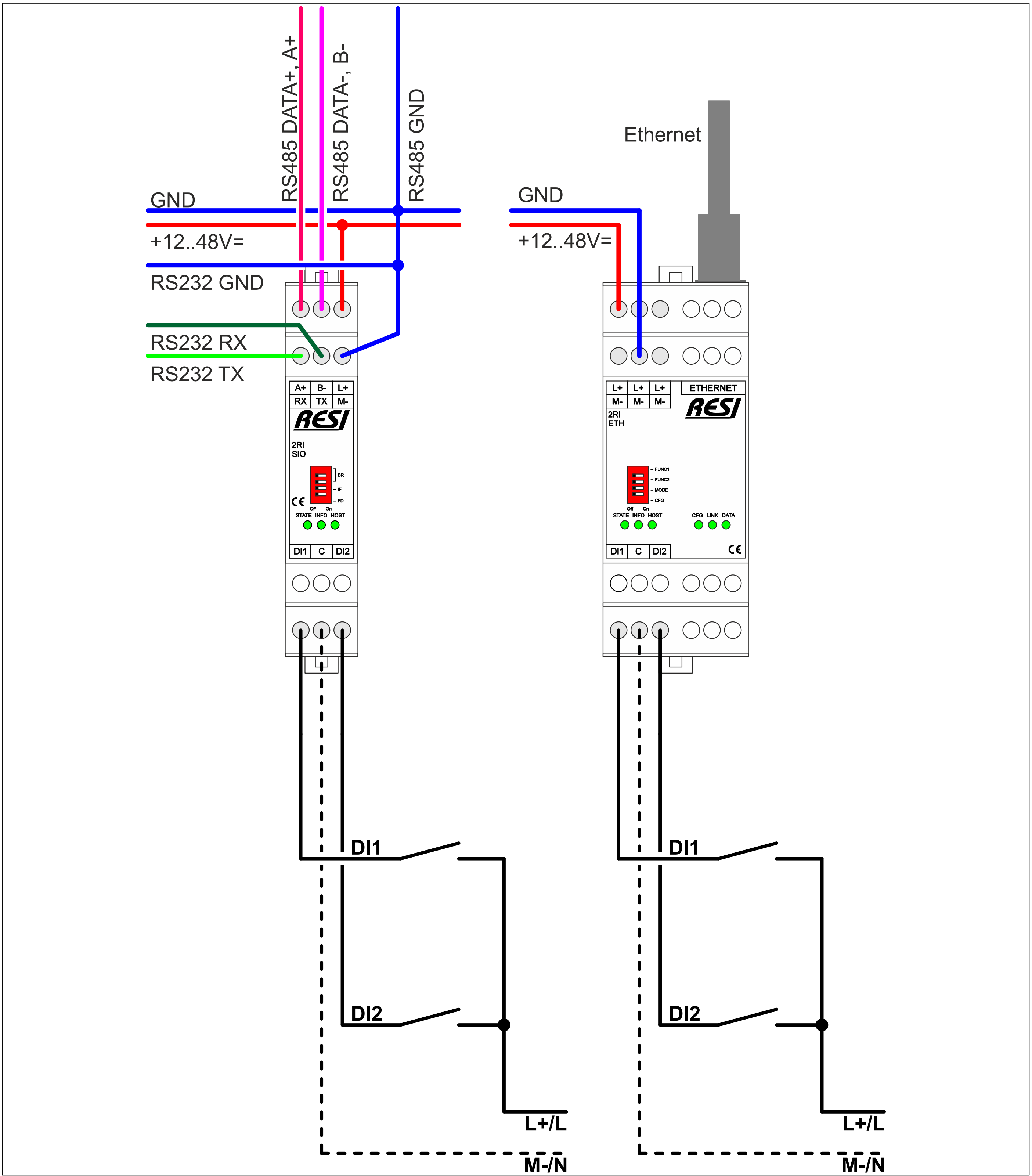


Figure: Schematics for the IO modules

26.5 RESI-2RI-SIO,ETH: Wiring diagram

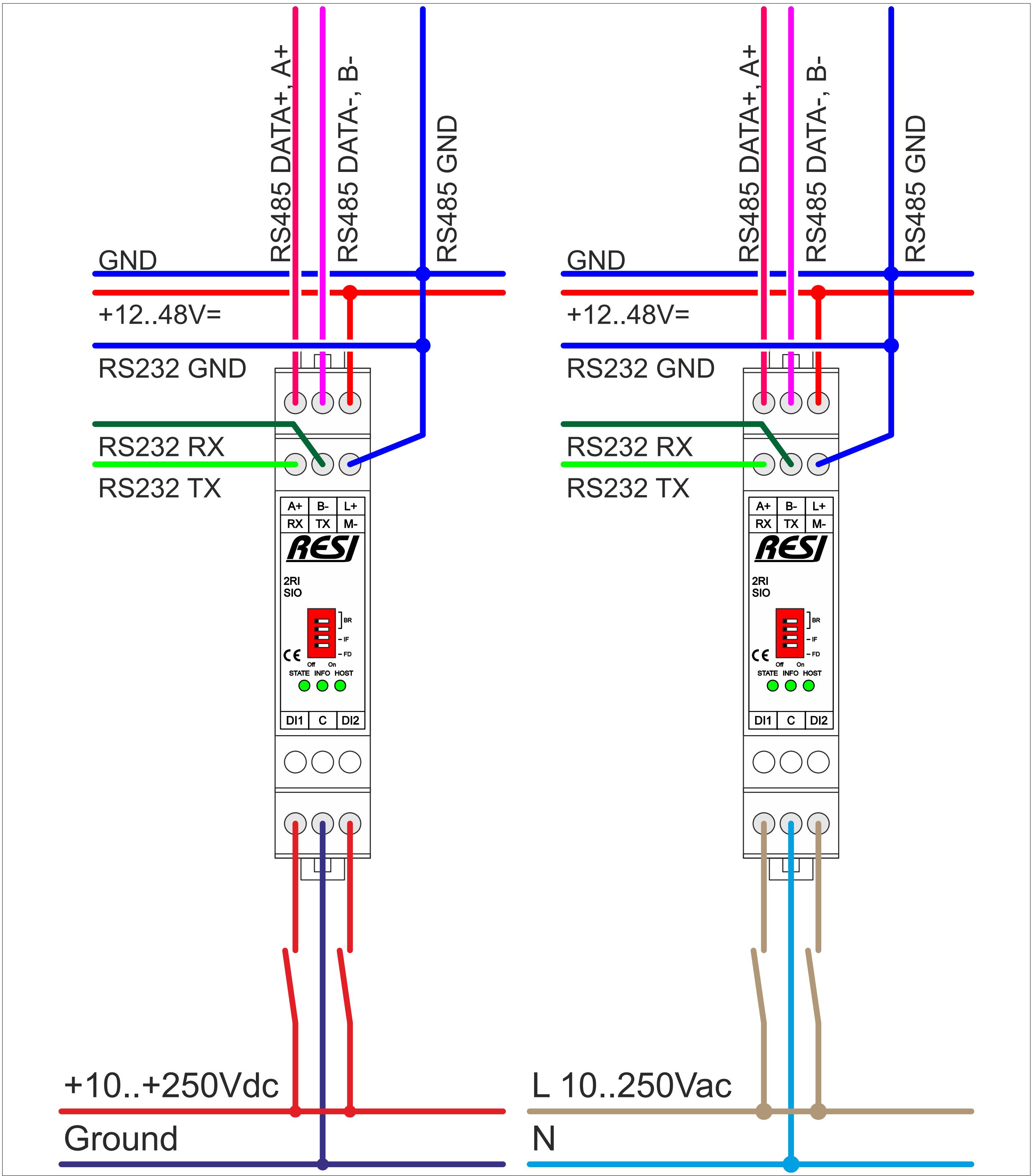


Figure: Wiring diagram for the IO modules

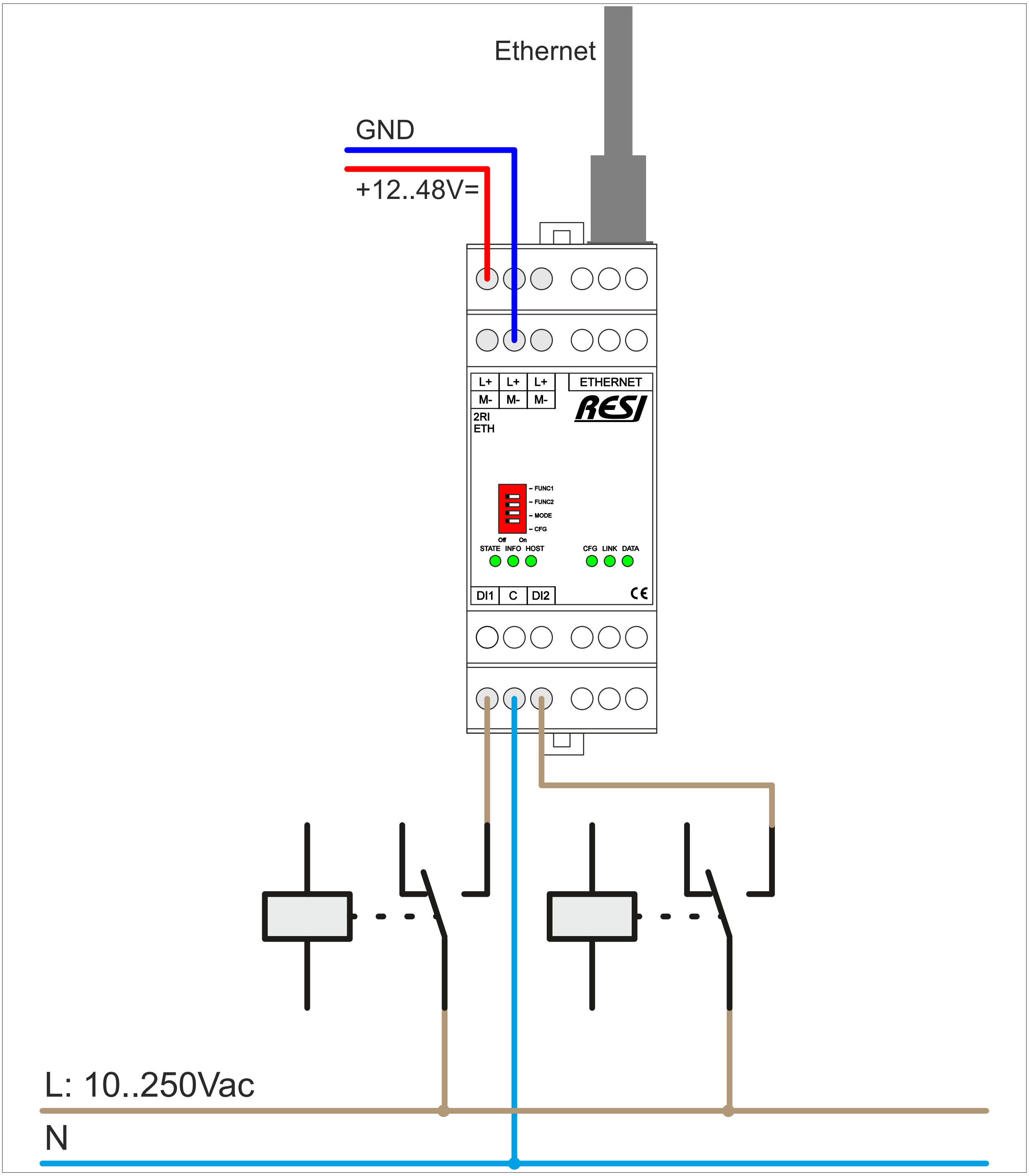


Figure: Wiring diagram for the IO modules

26.6 Additional MODBUS register & coils

Here you will find only the additional MODBUS registers and coils especially for this IO module. Please refer to the description of of the standard MODBUS mapping for more details about the available basic MODBUS registers and coils.

Please refer to the external document for detailed documentation of the current MODBUS register mapping for this IO module:

RESI-L-2RI-SIO-ETH-MODBUS+ASCII-ENxx.pdf

26.7 Additional ASCII commands

Here you will find only the additional ASCII commands especially for this IO module. Please refer to the description of of the standard commands for more details about the available basic ASCII commands.

Please refer to the external document for detailed documentation of the current ASCII commands for this IO module:

RESI-L-2RI-SIO-ETH-MODBUS+ASCII-ENxx.pdf

27 RESI-4DI-SIO, RESI-4DI-ETH

27.1 General information

This series of IO modules offer the following features:

- 4 digital inputs for 12…48Vdc signals
- Galvanic insulation between digital inputs and rest of the module
- RESI-xxx-SIO: Galvanic isolated RS232 and RS485 interface for communication with a host system
- RESI-xxx-ETH: Galvanic isolated Ethernet interface for communication with a host system

Figure: Our serial IO module

Figure: Our Ethernet IO module

27.2 Technical specification

Beside the basic technical data, which fulfil all of our IO modules, this IO modules meet the following technical specifications:

Power consumption	
RESI-4DI-SIO	<0.8W
RESI-4DI-ETH	<1.2W

Product housing	
RESI-4DI-SIO	CEM17
RESI-4DI-ETH	CEM35

Product weight	
RESI-4DI-SIO	58g
RESI-4DI-ETH	92g

Digital inputs	
Number	4
Signal type	12..48Vdc
Cable connection	via terminals
Galvanic isolation	Yes, between the four inputs and the CPU module
	not between the four digital inputs

Default serial settings	
Baud rate	via DIP switch
Parity	none
Stopbits	one
UnitID	255

Default Ethernet settings	
IP address	192.168.0.41
IP mask	255.255.255.0
gateway	192.168.0.1
UnitID	255

User	RESI
password	RESI

27.3 Additional terminals & LED states

DIGITAL INPUTS	4 digital inputs for 12...48Vdc signals	
	Two 3 pin terminal blocks	
	Terminal type:	USLIM
	M-:	Common ground for all four digital inputs
	DI1..DI4:	Digital inputs
Pin layout	DI1:	Signal input for digital input #1
	DI2:	Signal input for digital input #2
	DI3:	Signal input for digital input #3
	DI4:	Signal input for digital input #4
	M-:	Common ground for all four digital inputs
		Both M- clamps are internally bridged
INFO	If at least one of the digital inputs is ON, this LED is on.	
	If no digital input is ON, this LED is OFF too.	

27.4 RESI-4DI-SIO,ETH: Schematic diagram

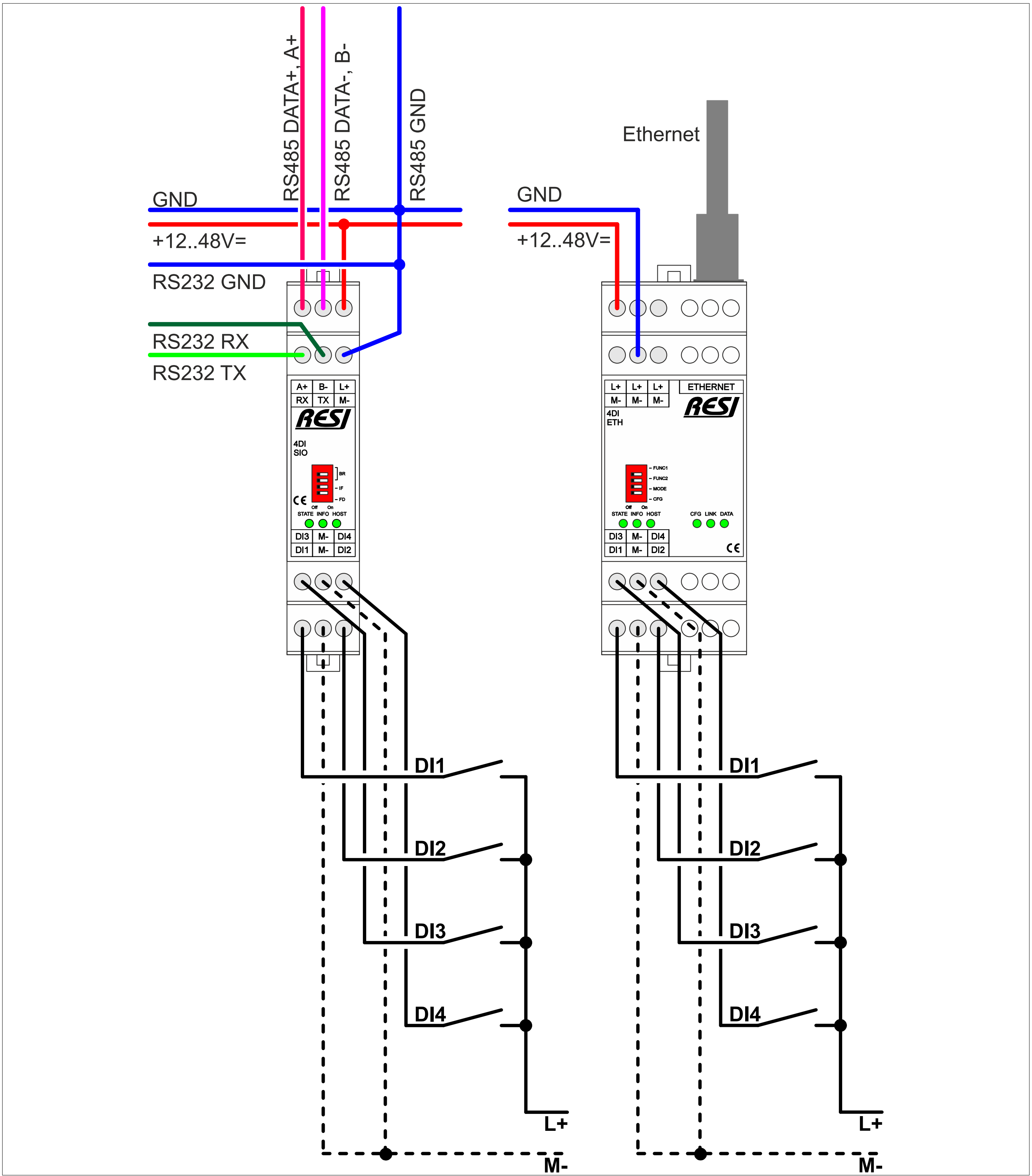


Figure: Schematics for the IO modules

27.5 RESI-4DI-SIO,ETH: Wiring diagram

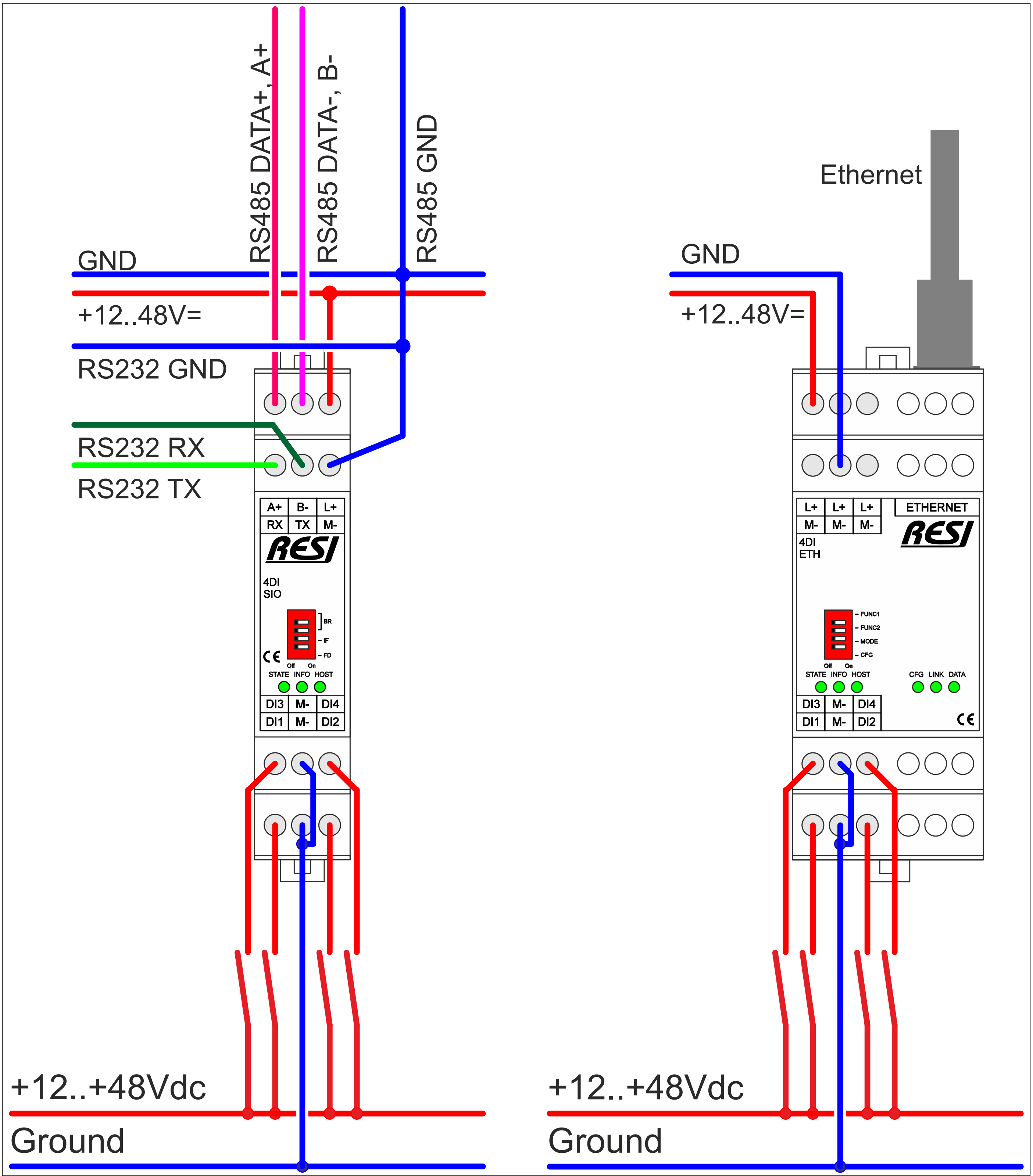


Figure: Wiring diagram for the IO modules

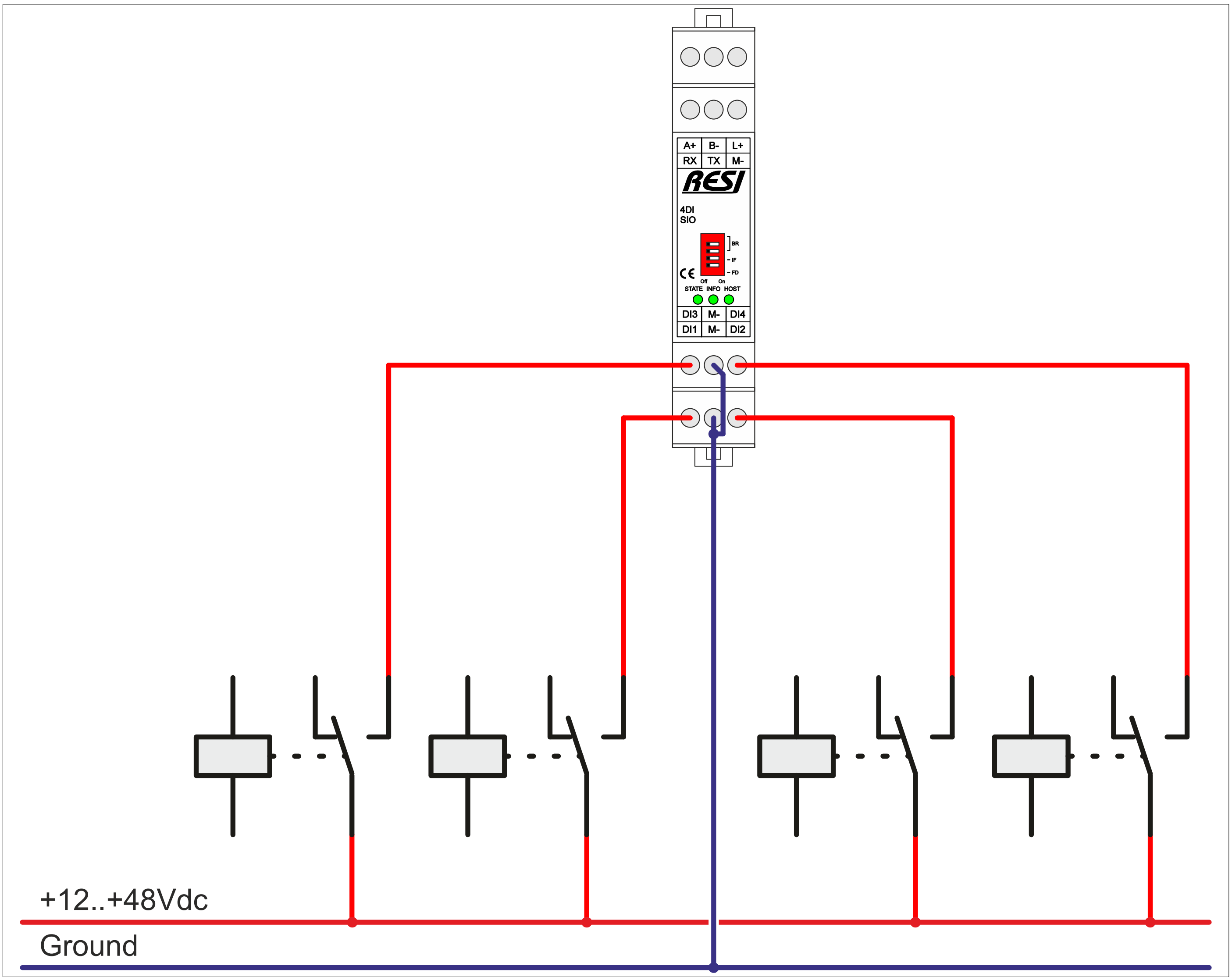


Figure: Wiring diagram for the IO modules

27.6 Additional MODBUS register & coils

Here you will find only the additional MODBUS registers and coils especially for this IO module. Please refer to the description of of the standard MODBUS mapping for more details about the available basic MODBUS registers and coils.

Please refer to the external document for detailed documentation of the current MODBUS register mapping for this IO module:

RESI-L-4DI-SIO-ETH-MODBUS+ASCII-ENxx.pdf

27.7 Additional ASCII commands

Here you will find only the additional ASCII commands especially for this IO module. Please refer to the description of of the standard commands for more details about the available basic ASCII commands.

Please refer to the external document for detailed documentation of the current ASCII commands for this IO module:

RESI-L-4DI-SIO-ETH-MODBUS+ASCII-ENxx.pdf

28 RESI-1S0-SIO, RESI-1S0-ETH

28.1 General information

This series of IO modules offer the following features:

- 1 counter input for S0 signals with 15V= output voltage
- Galvanic insulation between S0 signal and rest of the module
- RESI-xxx-SIO: Galvanic isolated RS232 and RS485 interface for communication with a host system
- RESI-xxx-ETH: Galvanic isolated Ethernet interface for communication with a host system

Figure: Our serial IO module

Figure: Our Ethernet IO module

28.2 Technical specification

Beside the basic technical data, which fulfil all of our IO modules, this IO modules meet the following technical specifications:

Power consumption	
RESI-1S0-SIO	<1.4W
RESI-1S0-ETH	<1.8W
Product housing	
RESI-1S0-SIO	CEM17
RESI-1S0-ETH	CEM35
Product weight	
RESI-1S0-SIO	53g
RESI-1S0-ETH	87g
Counter inputs	
Number	1
Signal type	S0 class B
S0 output voltage	max. 15V=
S0 output current	max. 20mA, typical 13.6mA
S0 pulse length	>=30ms
	Internal configurable digital filter for glitches
Cable connection	via terminals
Galvanic isolation	Yes, between the S0 input and the CPU module
HINT: The counted impulses are internally stored in a ferromagnetic RAM. After power on of the module, the last counter is readout from this FRAM. So no loss of counts can happen.	
Default serial settings	
Baud rate	via DIP switch
Parity	none
Stopbits	one
UnitID	255
Default Ethernet settings	
IP address	192.168.0.20
IP mask	255.255.255.0
gateway	192.168.0.1
UnitID	255
User	RESI
password	RESI

28.3 Additional terminals & LED states

COUNTER INPUTS	1 counter input for S0 impulses	
	One 3 pin terminal block	
	Terminal type:	USLIM
	S0+:	Positive S0 input
	S0-:	Negative S0 input
Pin layout	S0+:	Positive S0 input
	N/C:	not connected
	S0-:	Negative S0 input
INFO	If the S0 input is closed (ON), this LED is ON.	
	If the S0 input is opened (OFF), this LED is OFF.	

28.4 RESI-1S0-SIO,ETH: Schematic diagram

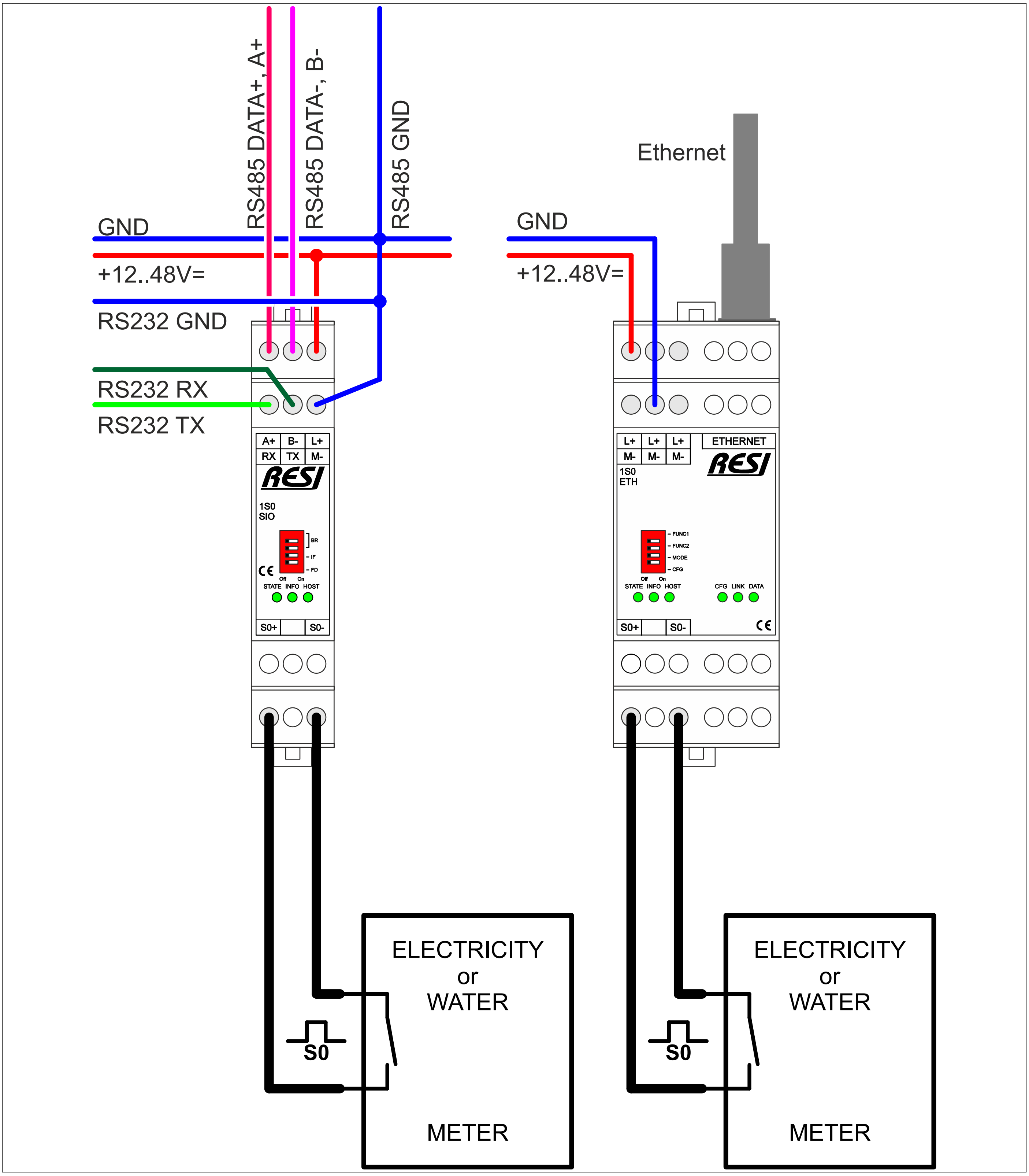


Figure: Schematics for the IO modules

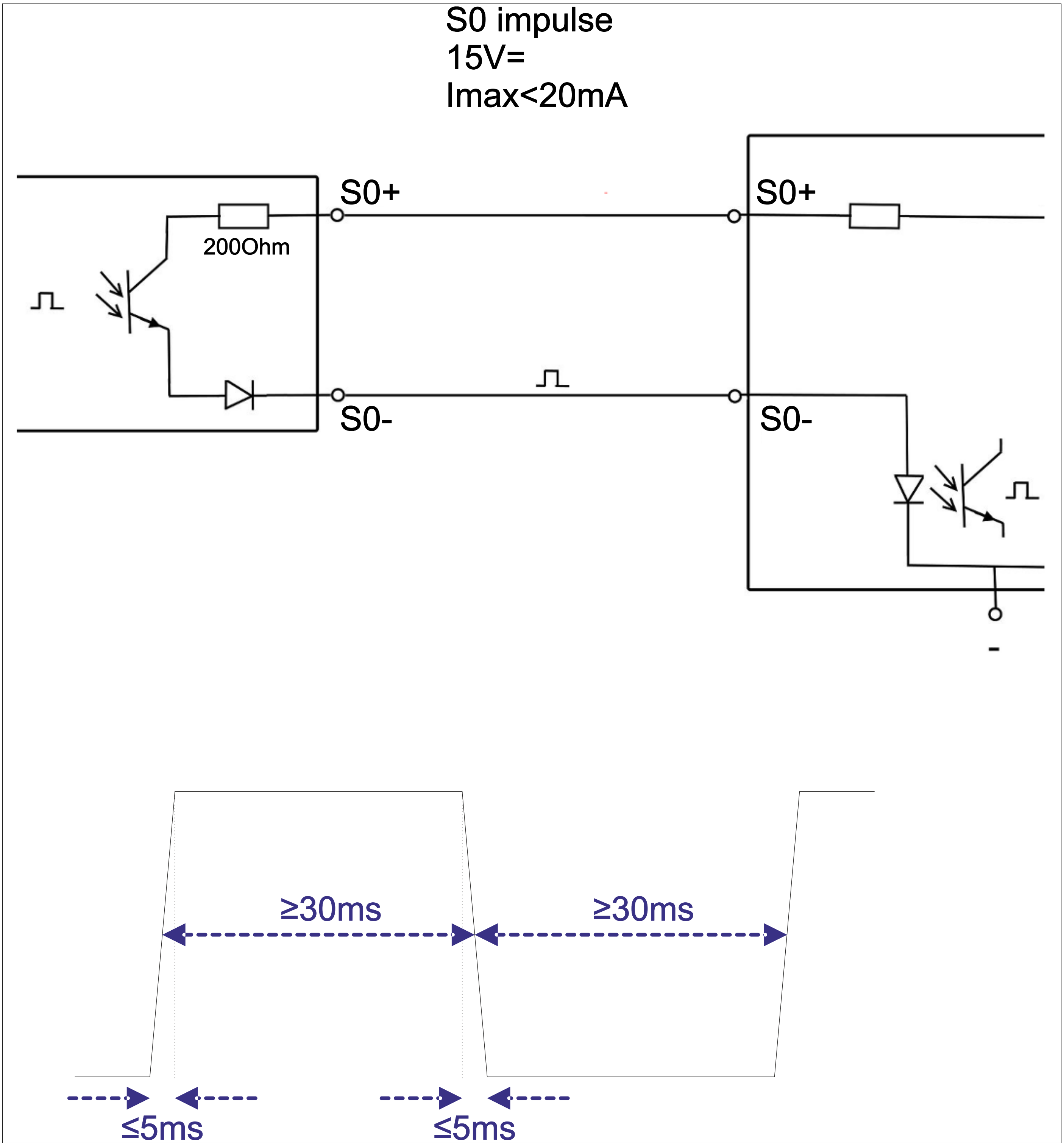


Figure: Schematics for S0 impulse

28.5 RESI-1S0-SIO,ETH: Wiring diagram

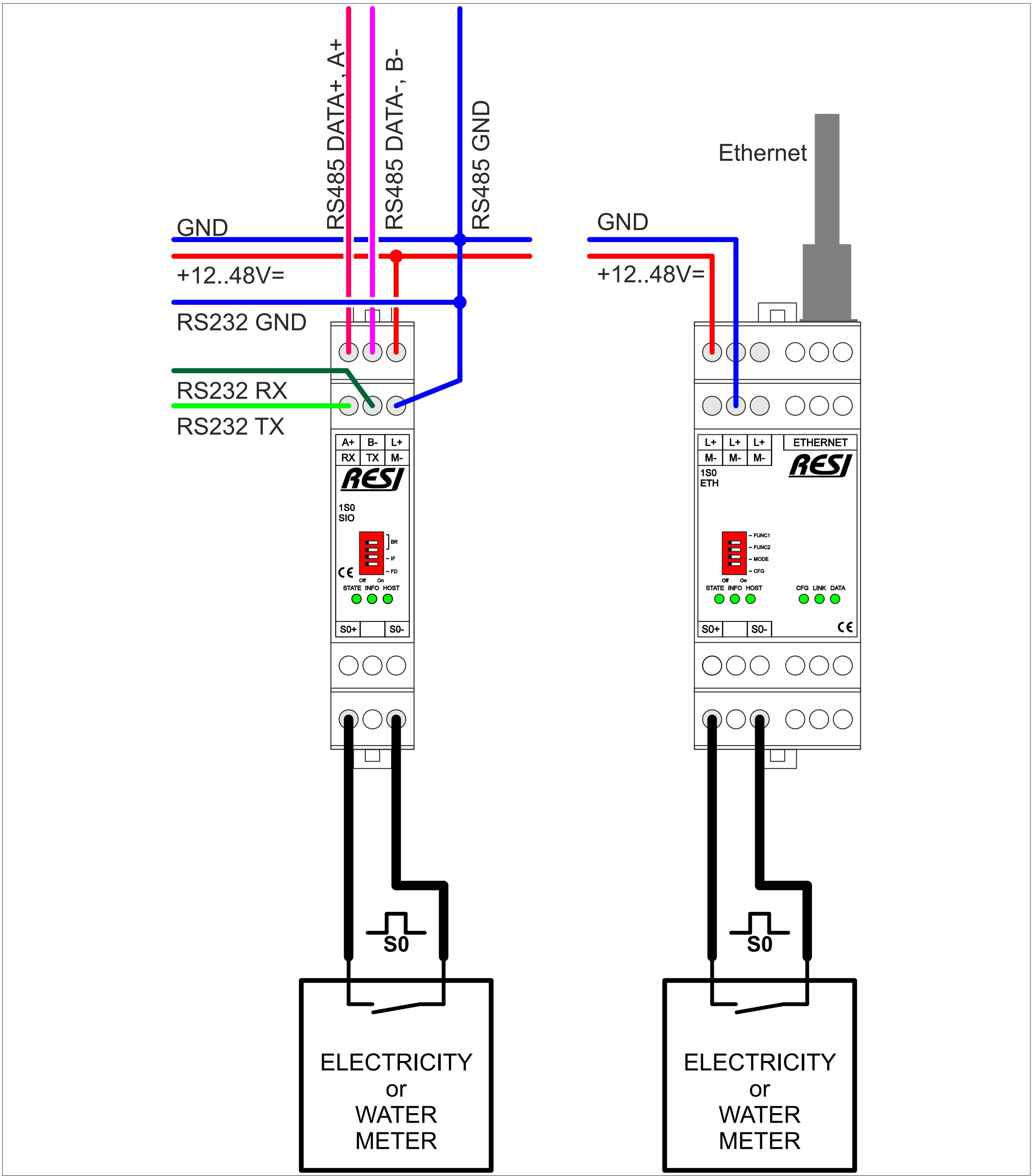


Figure: Wiring diagram for the IO modules

28.6 Additional MODBUS register & coils

Here you will find only the additional MODBUS registers and coils especially for this IO module. Please refer to the description of of the standard MODBUS mapping for more details about the available basic MODBUS registers and coils.

Please refer to the external document for detailed documentation of the current MODBUS register mapping for this IO module:

RESI-L-1S0-SIO-ETH-MODBUS+ASCII-ENxx.pdf

28.7 Additional ASCII commands

Here you will find only the additional ASCII commands especially for this IO module. Please refer to the description of of the standard commands for more details about the available basic ASCII commands.

Please refer to the external document for detailed documentation of the current ASCII commands for this IO module:

RESI-L-1S0-SIO-ETH-MODBUS+ASCII-ENxx.pdf

29 RESI-2S0-SIO, RESI-2S0-ETH

29.1 General information

This series of IO modules offer the following features:

- 2 counter inputs for S0 signals with 15V= output voltage
- Galvanic insulation between S0 signals and rest of the module
- RESI-xxx-SIO: Galvanic isolated RS232 and RS485 interface for communication with a host system
- RESI-xxx-ETH: Galvanic isolated Ethernet interface for communication with a host system

Figure: Our serial IO module

Figure: Our Ethernet IO module

29.2 Technical specification

Beside the basic technical data, which fulfil all of our IO modules, this IO modules meet the following technical specifications:

Power consumption	
RESI-2S0-SIO	<1.5W
RESI-2S0-ETH	<1.9W

Product housing	
RESI-2S0-SIO	CEM17
RESI-2S0-ETH	CEM35

Product weight	
RESI-2S0-SIO	56g
RESI-2S0-ETH	90g

Counter inputs	
Number	2
Signal type	S0 class B
S0 output voltage	max. 15V=
S0 output current	max. 20mA, typical 8.2mA
S0 pulse length	>=30ms
	Internal configurable digital filter for glitches
Cable connection	via terminals
Galvanic isolation	Yes, between the two S0 inputs and the CPU module
	but nopt between the two S0 inputs

HINT: The counted impulses are internally stored in a ferromagnetic RAM. After power on of the module, the last counter is readout from this FRAM. So no loss of counts can happen.

Default serial settings	
Baud rate	via DIP switch
Parity	none
Stopbits	one
UnitID	255

Default Ethernet settings	
IP address	192.168.0.21
IP mask	255.255.255.0
gateway	192.168.0.1
UnitID	255

User	RESI
password	RESI

29.3 Additional terminals & LED states

COUNTER INPUTS	2 counter input for S0 impulses	
	Two 3 pin terminal blocks	
	Terminal type:	USLIM
	S0+A:	Positive S0 input #1
	S0-A:	Negative S0 input #1
	S0+B:	Positive S0 input #2
	S0-B:	Negative S0 input #2
Pin layout	S0+A:	Positive S0 input #1
	N/C:	not connected
	S0-A:	Negative S0 input #1
	S0+B:	Positive S0 input #2
	N/C:	not connected
	S0-B:	Negative S0 input #2
INFO	If at least one of the S0 inputs is closed (ON), this LED is ON.	
	If all of the S0 inputs are opened (OFF), this LED is OFF.	

29.4 RESI-2S0-SIO,ETH: Schematic diagram

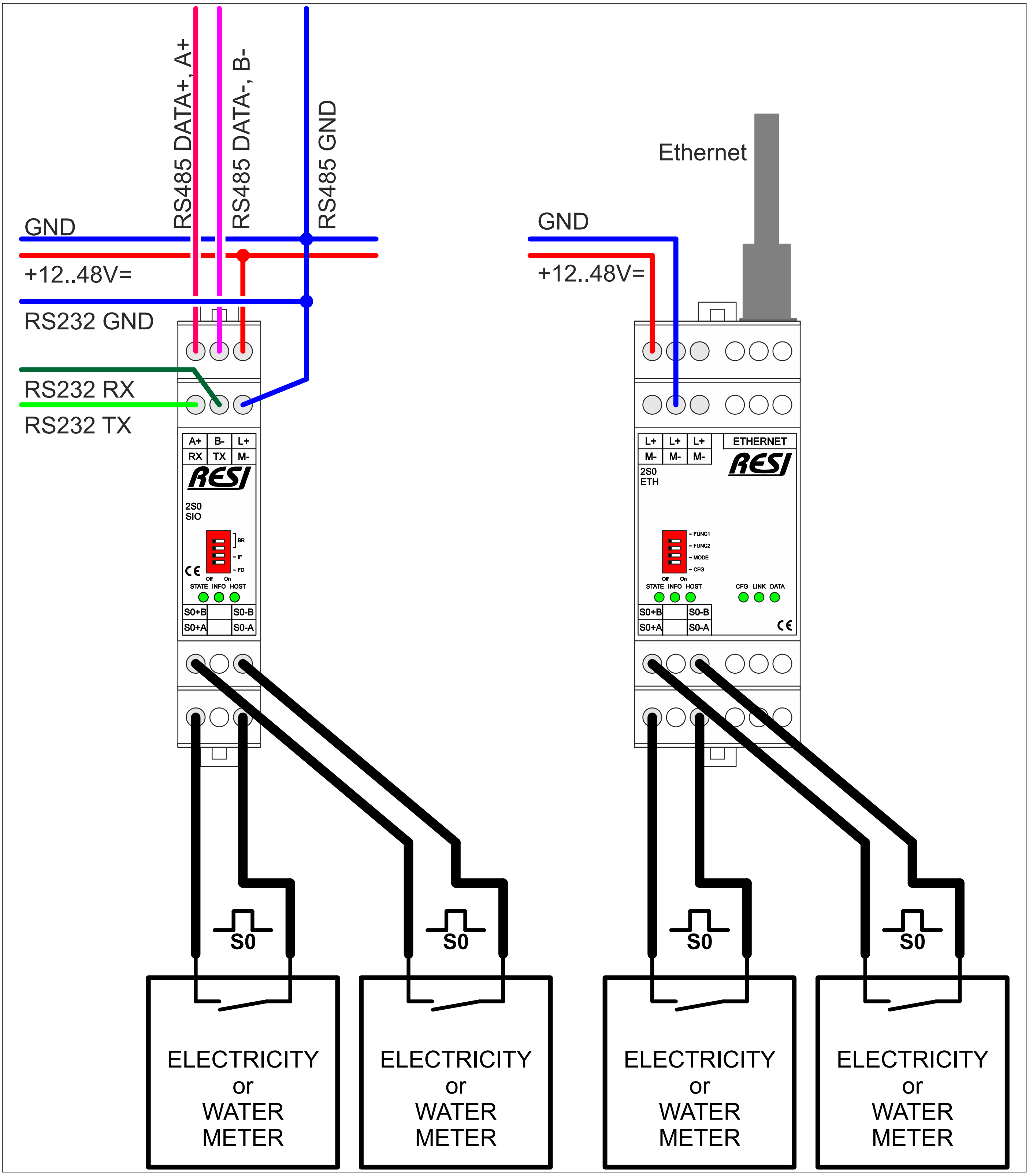


Figure: Schematics for the IO modules

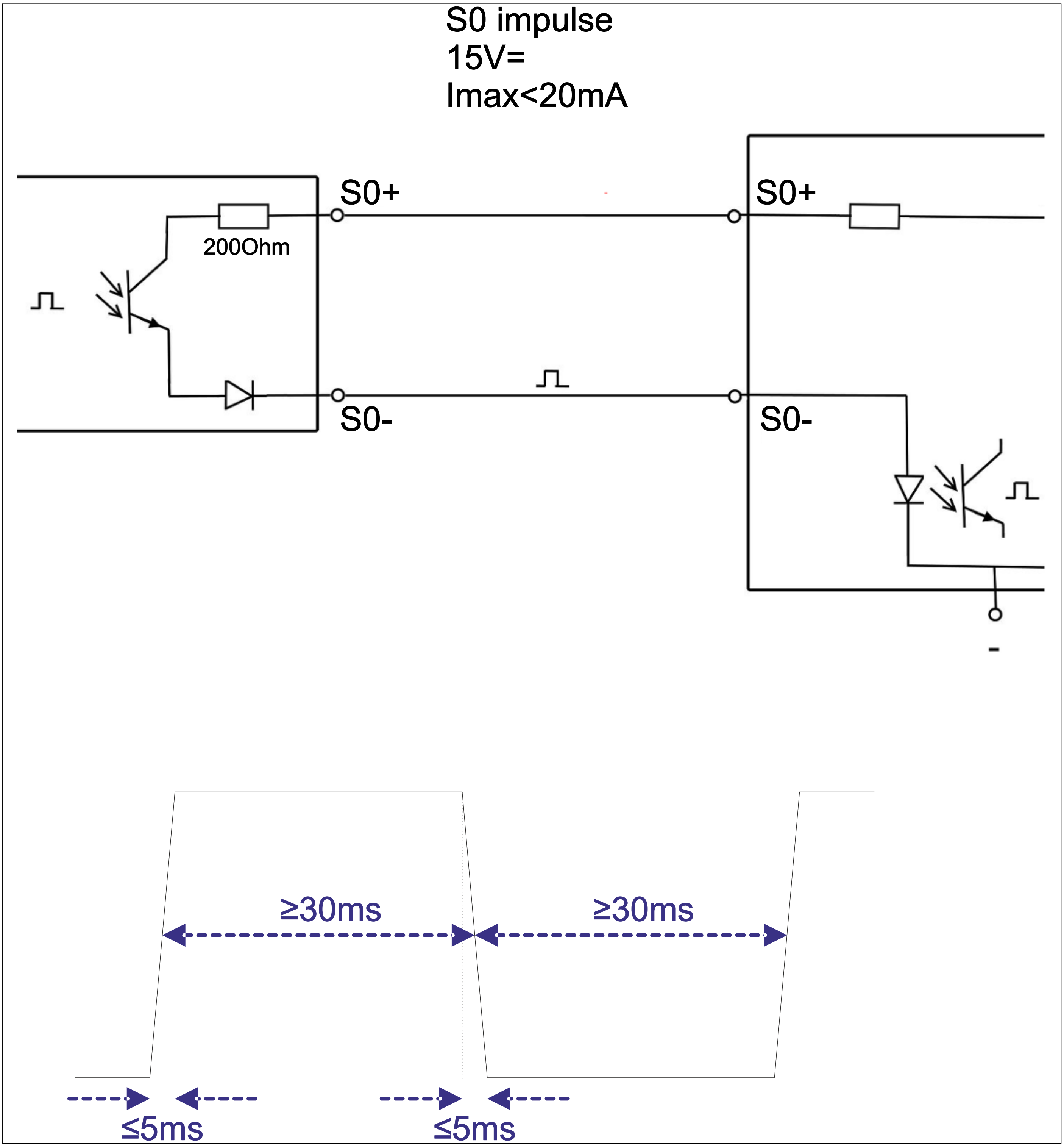


Figure: Schematics for S0 impulse

29.5 RESI-2S0-SIO,ETH: Wiring diagram

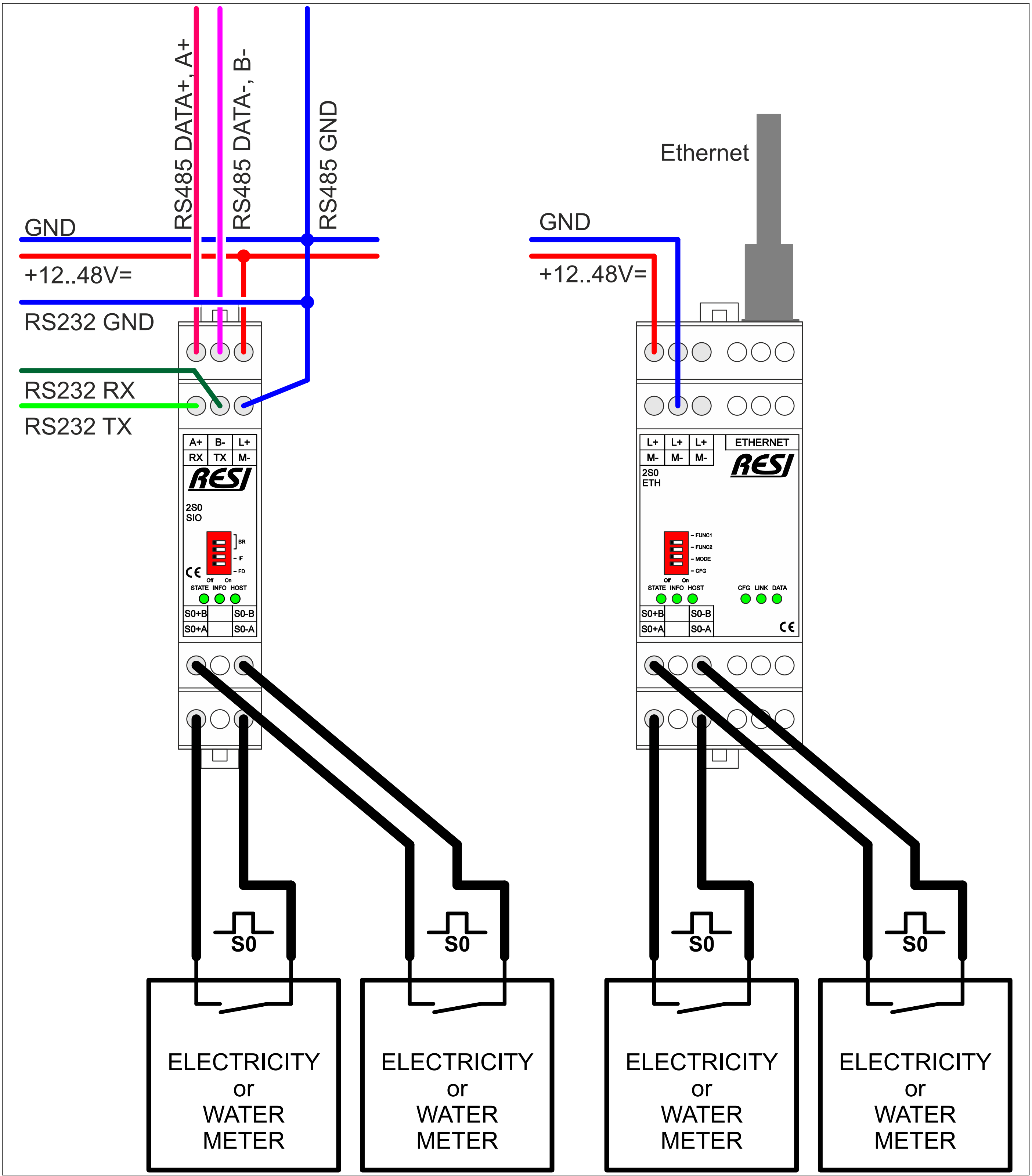


Figure: Wiring diagram for the IO modules

29.6 Additional MODBUS register & coils

Here you will find only the additional MODBUS registers and coils especially for this IO module. Please refer to the description of of the standard MODBUS mapping for more details about the available basic MODBUS registers and coils.

Please refer to the external document for detailed documentation of the current MODBUS register mapping for this IO module:

RESI-L-2S0-SIO-ETH-MODBUS+ASCII-ENxx.pdf

29.7 Additional ASCII commands

Here you will find only the additional ASCII commands especially for this IO module. Please refer to the description of of the standard commands for more details about the available basic ASCII commands.

Please refer to the external document for detailed documentation of the current ASCII commands for this IO module:

RESI-L-2S0-SIO-ETH-MODBUS+ASCII-ENxx.pdf

30 RESI-1RO-SIO, RESI-1RO-ETH

30.1 General information

This series of IO modules offer the following features:

- 1 relay output with changeover contacts (NO and NC)
- Contact rating: max. 250Vac, 30Vdc, 8A
- Galvanic insulation with the relay from the rest of the module
- RESI-xxx-SIO: Galvanic isolated RS232 and RS485 interface for communication with a host system
- RESI-xxx-ETH: Galvanic isolated Ethernet interface for communication with a host system

Figure: Our serial IO module

Figure: Our Ethernet IO module

30.2 Technical specification

Beside the basic technical data, which fulfil all of our IO modules, this IO modules meet the following technical specifications:

Power consumption	
RESI-1RO-SIO	<0.9W
RESI-1RO-ETH	<1.3W
Product housing	
RESI-1RO-SIO	CEM17
RESI-1RO-ETH	CEM35
Product weight	
RESI-1RO-SIO	59g
RESI-1RO-ETH	93g
Relay outputs	
Number	1
Relay output voltage	max. 250Vac or 30Vdc
Relay output current	max. 8A
Relay type	Changeover relay with NO and NC contacts
Contact material	Au-flashed AgNi
Maximum contact rating	with 250Vac: 2000VA
	with 30Vdc: 240W
Maximum contact voltage	250Vac or 125Vdc with 0.2A
Cable connection	via terminals
Galvanic isolation	Yes, with the relay itself to the rest of the module
Default serial settings	
Baud rate	via DIP switch
Parity	none
Stopbits	one
UnitID	255
Default Ethernet settings	
IP address	192.168.0.42
IP mask	255.255.255.0
gateway	192.168.0.1
UnitID	255
User	RESI
password	RESI

30.3 Additional terminals & LED states

RELAY OUTPUT	1 relay output for 250Vac, 30Vdc,8A	
	One 3 pin terminal block	
	Terminal type:	USLIM
	NO:	Maker contact of relay (Form A)
	C:	Common root for NO and NC contacts
	NC:	Breaker contact of relay (Form B)
Pin layout	NO:	Maker contact of relay (Form A)
	C:	Common root for NO and NC contacts
	NC:	Breaker contact of relay (Form B)
INFO	If the relay output is activated (ON), this LED is ON.	
	If the relay output is deactivated (OFF), this LED is OFF.	

30.4 RESI-1RO-SIO,ETH: Schematic diagram

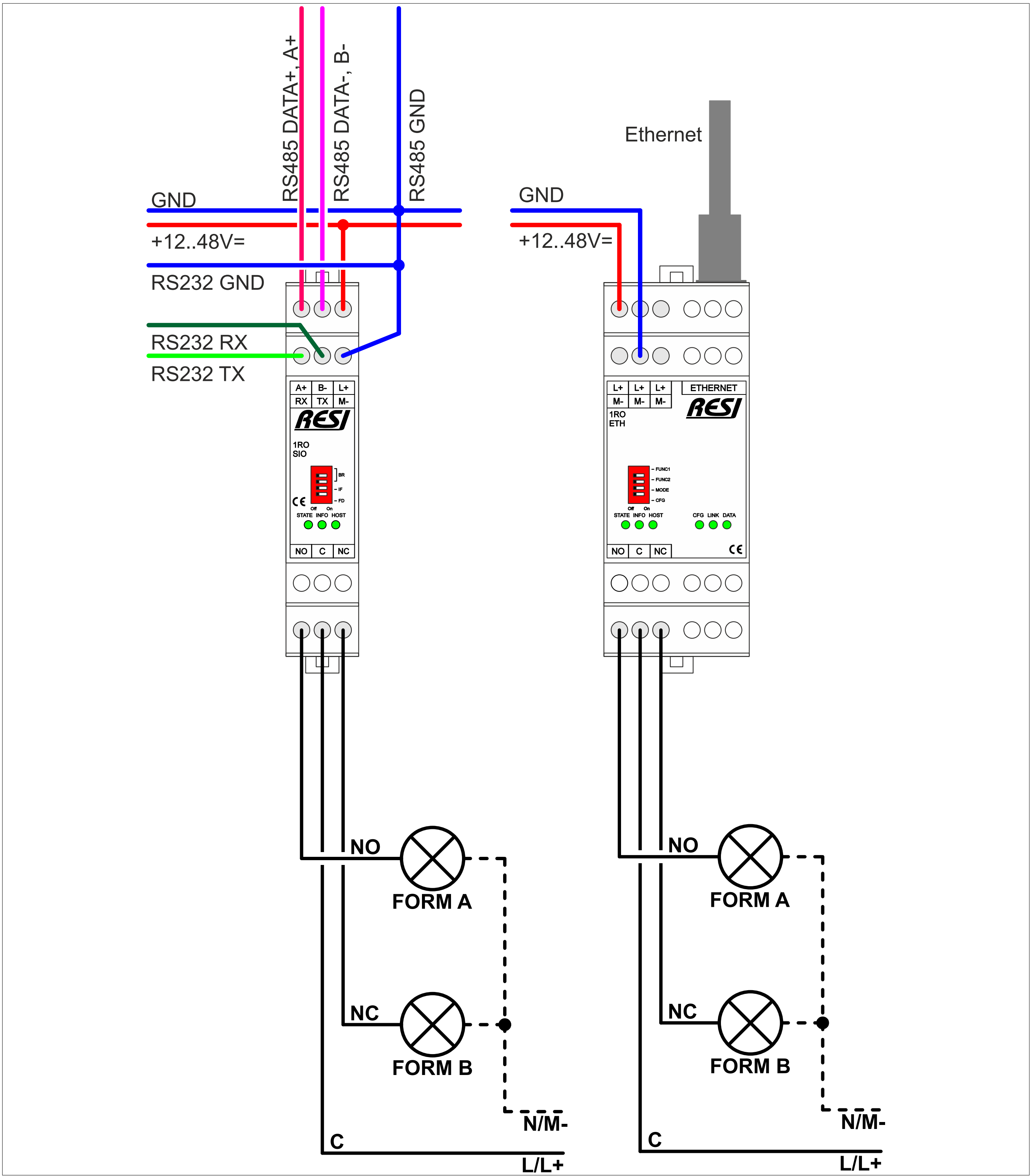


Figure: Schematics for the IO modules

30.5 RESI-1RO-SIO,ETH: Wiring diagram

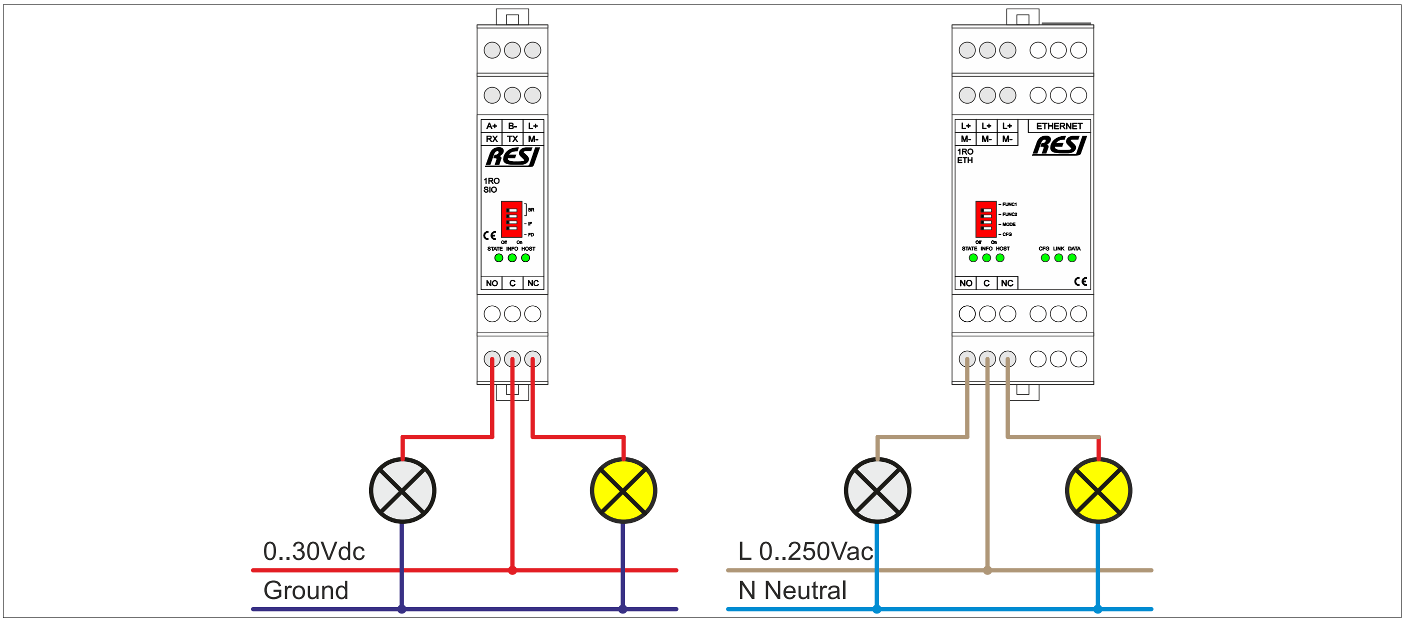


Figure: Wiring diagram for the IO modules, relay is deactivated (OFF)

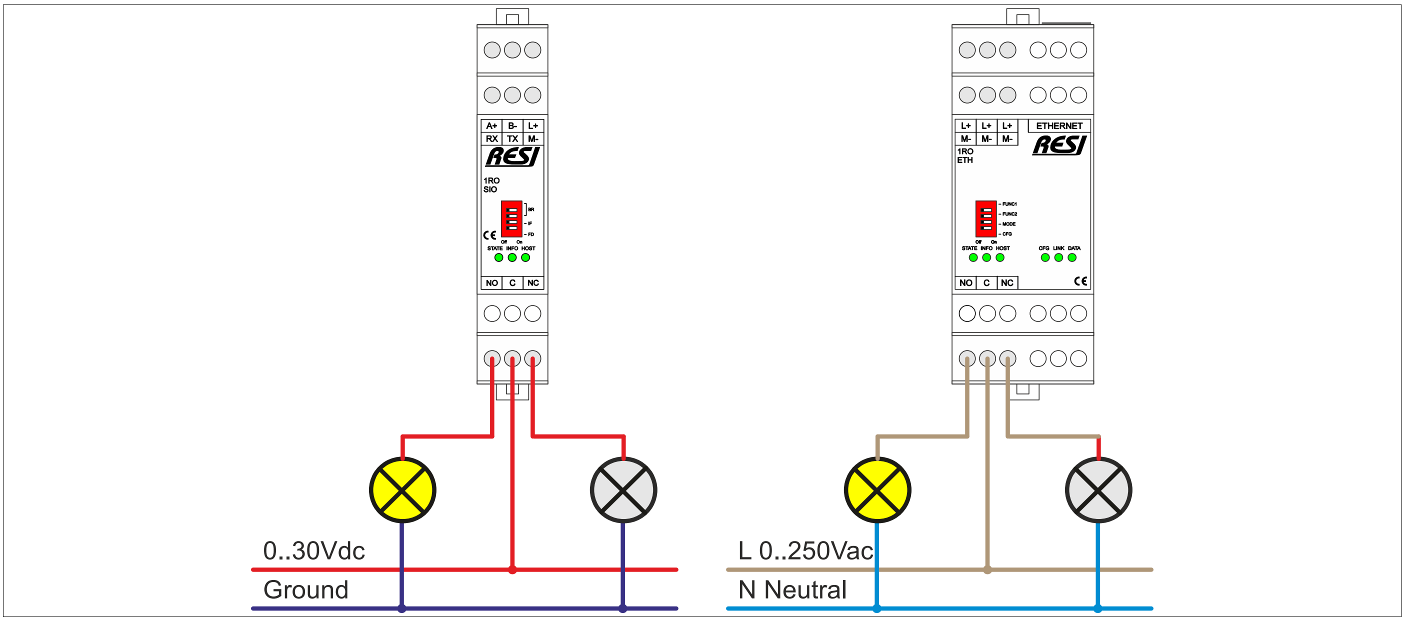


Figure: Wiring diagram for the IO modules, relay is activated (ON)

30.6 Additional MODBUS register & coils

Here you will find only the additional MODBUS registers and coils especially for this IO module. Please refer to the description of of the standard MODBUS mapping for more details about the available basic MODBUS registers and coils.

Please refer to the external document for detailed documentation of the current MODBUS register mapping for this IO module:

RESI-L-1RO-SIO-ETH-MODBUS+ASCII-ENxx.pdf

30.7 Additional ASCII commands

Here you will find only the additional ASCII commands especially for this IO module. Please refer to the description of of the standard commands for more details about the available basic ASCII commands.

Please refer to the external document for detailed documentation of the current ASCII commands for this IO module:

RESI-L-1RO-SIO-ETH-MODBUS+ASCII-ENxx.pdf

31 RESI-2RO-SIO, RESI-2RO-ETH

31.1 General information

This series of IO modules offer the following features:

- 2 relay outputs with maker contacts (NO) and common root contact
- Contact rating: max. 250Vac, 30Vdc, 8A
- Galvanic insulation with the relays from rest of the module
- RESI-xxx-SIO: Galvanic isolated RS232 and RS485 interface for communication with a host system
- RESI-xxx-ETH: Galvanic isolated Ethernet interface for communication with a host system

Figure: Our serial IO module

Figure: Our Ethernet IO module

31.2 Technical specification

Beside the basic technical data, which fulfil all of our IO modules, this IO modules meet the following technical specifications:

Power consumption	
RESI-2RO-SIO	<1.2W
RESI-2RO-ETH	<1.6W
Product housing	
RESI-2RO-SIO	CEM17
RESI-2RO-ETH	CEM35
Product weight	
RESI-2RO-SIO	69g
RESI-2RO-ETH	104g
Relay outputs	
Number	2
Relay output voltage	max. 250Vac or 30Vdc
Relay output current	max. 8A
Relay type	Two Form A relay with maker contact NO
	Common root contact for both relays
Contact material	Au-flashed AgNi
Maximum contact rating	with 250Vac: 2000VA
	with 30Vdc: 240W
Maximum contact voltage	250Vac or 125Vdc with 0.2A
Cable connection	via terminals
Galvanic isolation	Yes, with the relay itself to the rest of the module
Default serial settings	
Baud rate	via DIP switch
Parity	none
Stopbits	one
UnitID	255
Default Ethernet settings	
IP address	192.168.0.43
IP mask	255.255.255.0
Gateway	192.168.0.1
UnitID	255
User	RESI
Password	RESI

31.3 Additional terminals & LED states

RELAY OUTPUTS	2 relay outputs for 250Vac, 30Vdc,8A	
	Two 3 pin terminal blocks	
	Terminal type:	USLIM
	RO1:	Maker contact of relay #1 (NO, Form A)
	RO2:	Maker contact of relay #2 (NO, Form A)
	L/L+:	Common root for both relays
	N/M-:	Neutral/Ground signal, bridged
Pin layout	RO1:	Maker contact of relay #1 (NO, Form A)
	RO2:	Maker contact of relay #2 (NO, Form A)
	L/L+:	Common root for both relays
	N/M-:	Neutral/Ground signal, bridged
	N/M-:	Neutral/Ground signal, bridged
	N/M-:	Neutral/Ground signal, bridged
INFO	If one of the relay outputs is activated (ON), this LED is ON.	
	If none of the relay outputs is activated (OFF), this LED is OFF.	

31.4 RESI-2RO-SIO,ETH: Schematic diagram

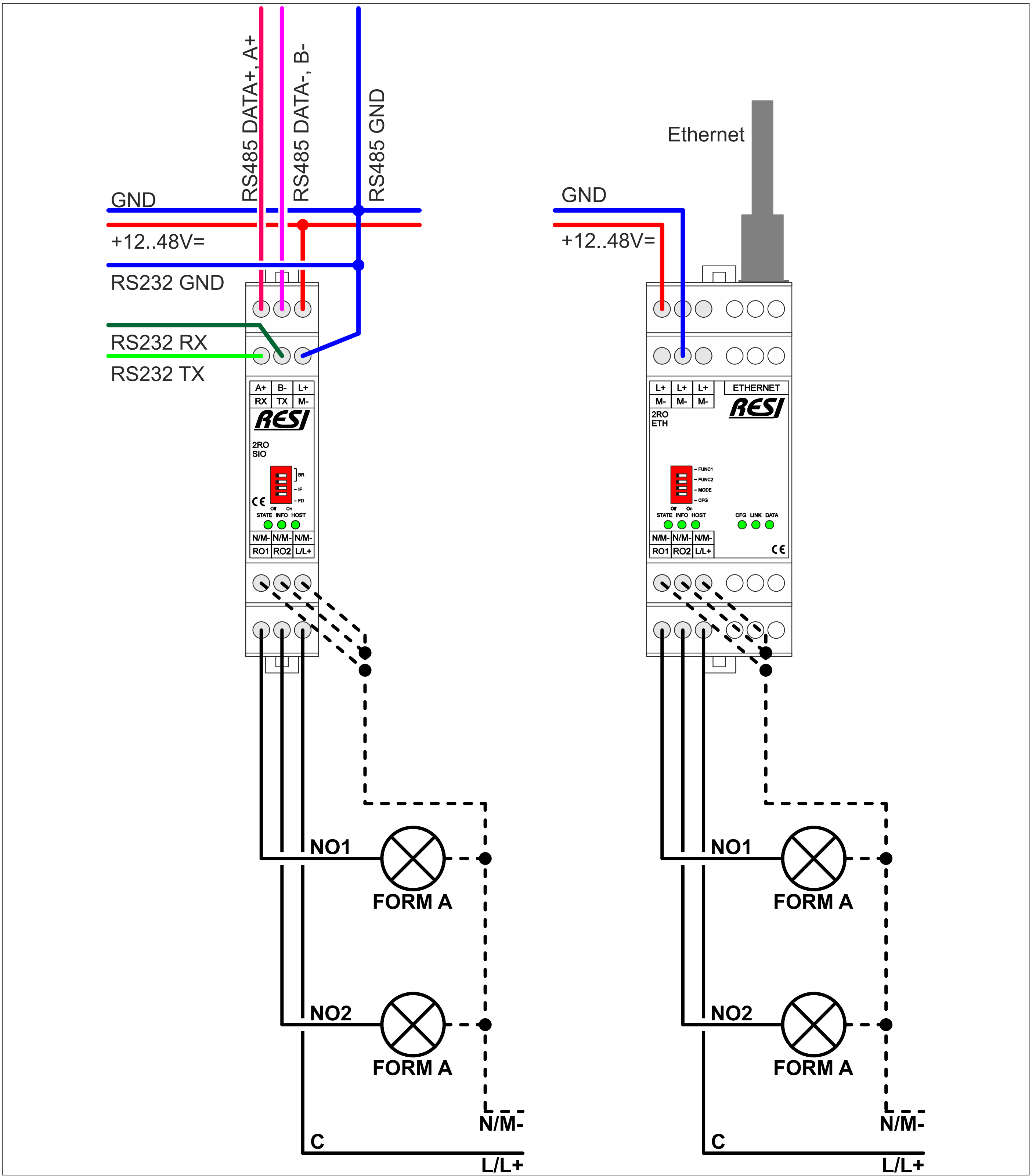


Figure: Schematics for the IO modules

31.5 RESI-2RO-SIO,ETH: Wiring diagram

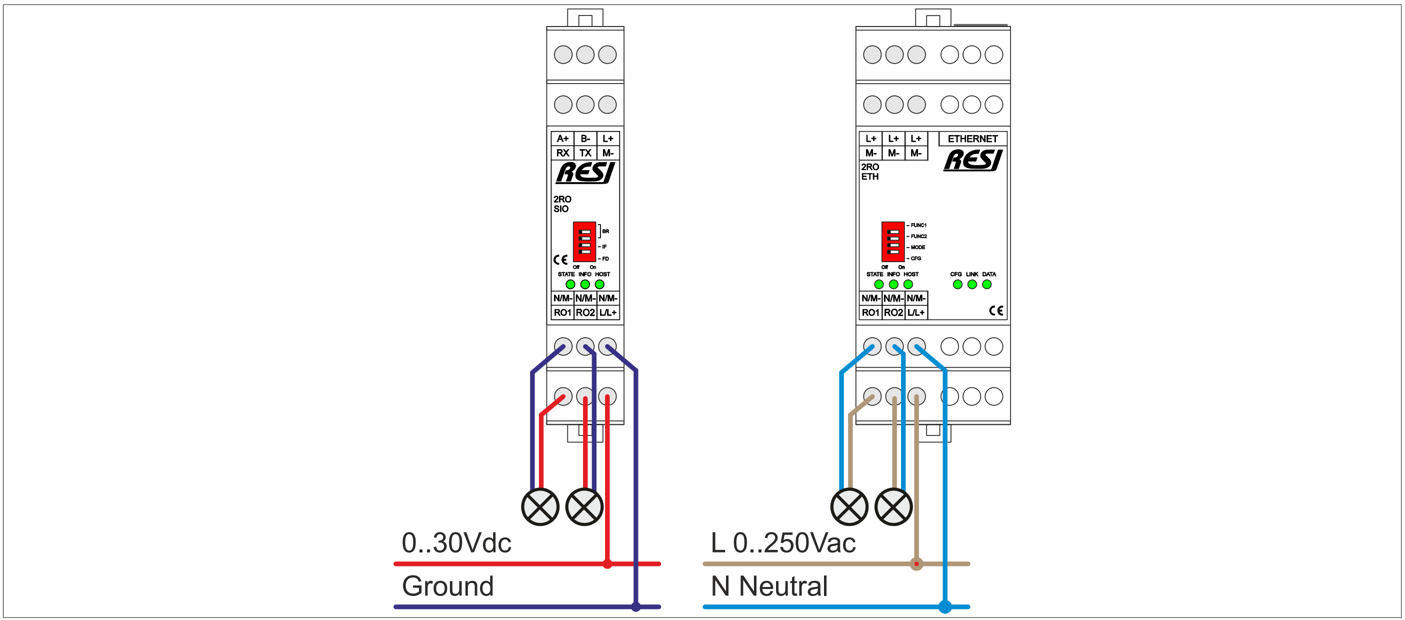


Figure: Wiring diagram for the IO modules, both relays are deactivated (OFF)

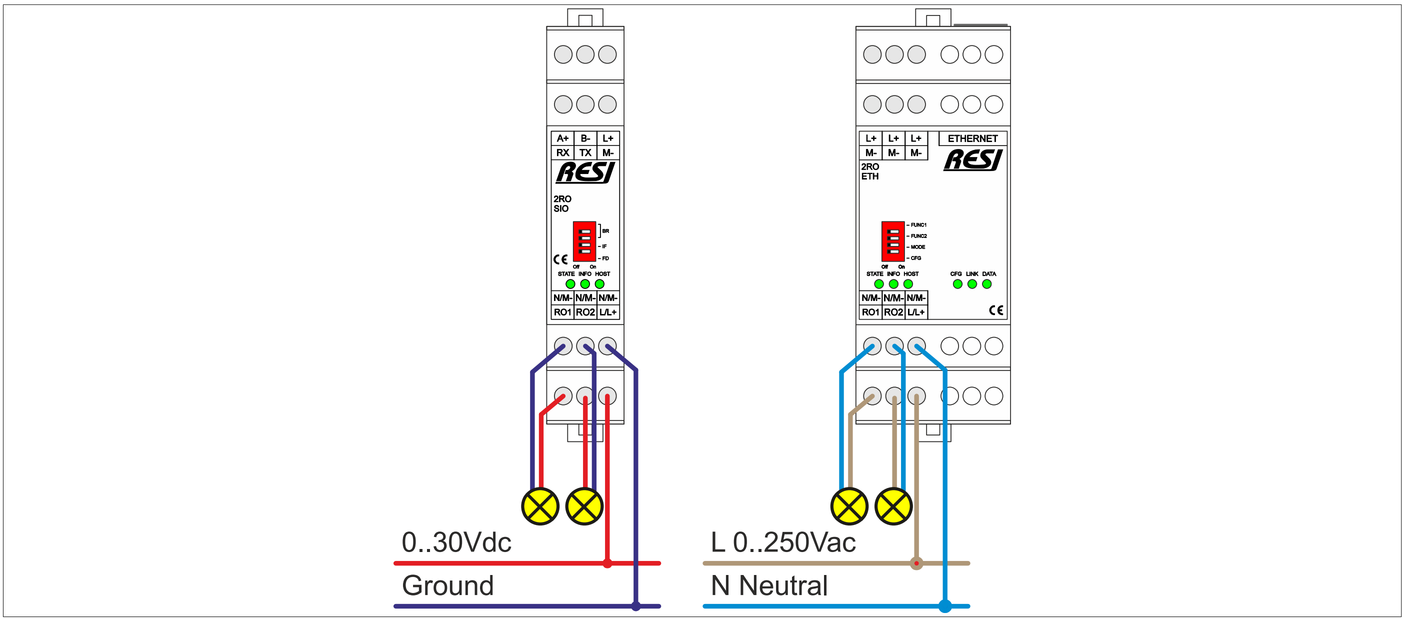


Figure: Wiring diagram for the IO modules, both relays are activated (ON)

31.6 Additional MODBUS register & coils

Here you will find only the additional MODBUS registers and coils especially for this IO module. Please refer to the description of of the standard MODBUS mapping for more details about the available basic MODBUS registers and coils.

Please refer to the external document for detailed documentation of the current MODBUS register mapping for this IO module:

RESI-L-2RO-SIO-ETH-MODBUS+ASCII-ENxx.pdf

31.7 Additional ASCII commands

Here you will find only the additional ASCII commands especially for this IO module. Please refer to the description of of the standard commands for more details about the available basic ASCII commands.

Please refer to the external document for detailed documentation of the current ASCII commands for this IO module:

RESI-L-2RO-SIO-ETH-MODBUS+ASCII-ENxx.pdf

32 RESI-4DO-SIO, RESI-4DO-ETH

32.1 General information

This series of IO modules offer the following features:

- 4 digital outputs for max. 30V= and 150mA output current per output
- Integrated over temperature and over current fault detection and open load detection
- DC Input supply max. 30V=
- No galvanic insulation from rest of the module
- RESI-xxx-SIO: Galvanic isolated RS232 and RS485 interface for communication with a host system
- RESI-xxx-ETH: Galvanic isolated Ethernet interface for communication with a host system

Figure: Our serial IO module

Figure: Our Ethernet IO module

32.2 Technical specification

Beside the basic technical data, which fulfil all of our IO modules, this IO modules meet the following technical specifications:

Power consumption	
RESI-4DO-SIO	<??W, tbd
RESI-4DO-ETH	<??W, tbd
Product housing	
RESI-4DO-SIO	CEM17
RESI-4DO-ETH	CEM35
Product weight	
RESI-4DO-SIO	59g
RESI-4DO-ETH	93g
Digital outputs	
Number	4
Output voltage	max. 30Vdc
Output current	typical 150mA , max. 350mA
Cable connection	via terminals
Galvanic isolation	No
Default serial settings	
Baud rate	via DIP switch
Parity	none
Stopbits	one
UnitID	255
Default Ethernet settings	
IP address	192.168.0.45
IP mask	255.255.255.0
gateway	192.168.0.1
UnitID	255
User	RESI
password	RESI

32.3 Additional terminals & LED states

DIGITAL OUTPUTS	4 digital outputs for 30Vdc, 150mA	
	Two 3 pin terminal blocks	
	Terminal type:	USLIM
	DO1...DO4:	Digital output #1 to #4
	L+:	Power supply for digital output
	M-:	Ground of power supply for digital outputs
Pin layout	DO1:	Digital output #1 signal
	M-:	Ground of power supply for digital outputs
	DO2:	Digital output #2 signal
	DO3:	Digital output #3 signal
	L+:	Power supply for digital output
	DO4:	Digital output #4 signal
INFO	If one of the digital outputs is activated (ON), this LED is ON.	
	If none of the digital outputs is activated (OFF), this LED is OFF.	

32.4 RESI-4DO-SIO,ETH: Schematic diagram

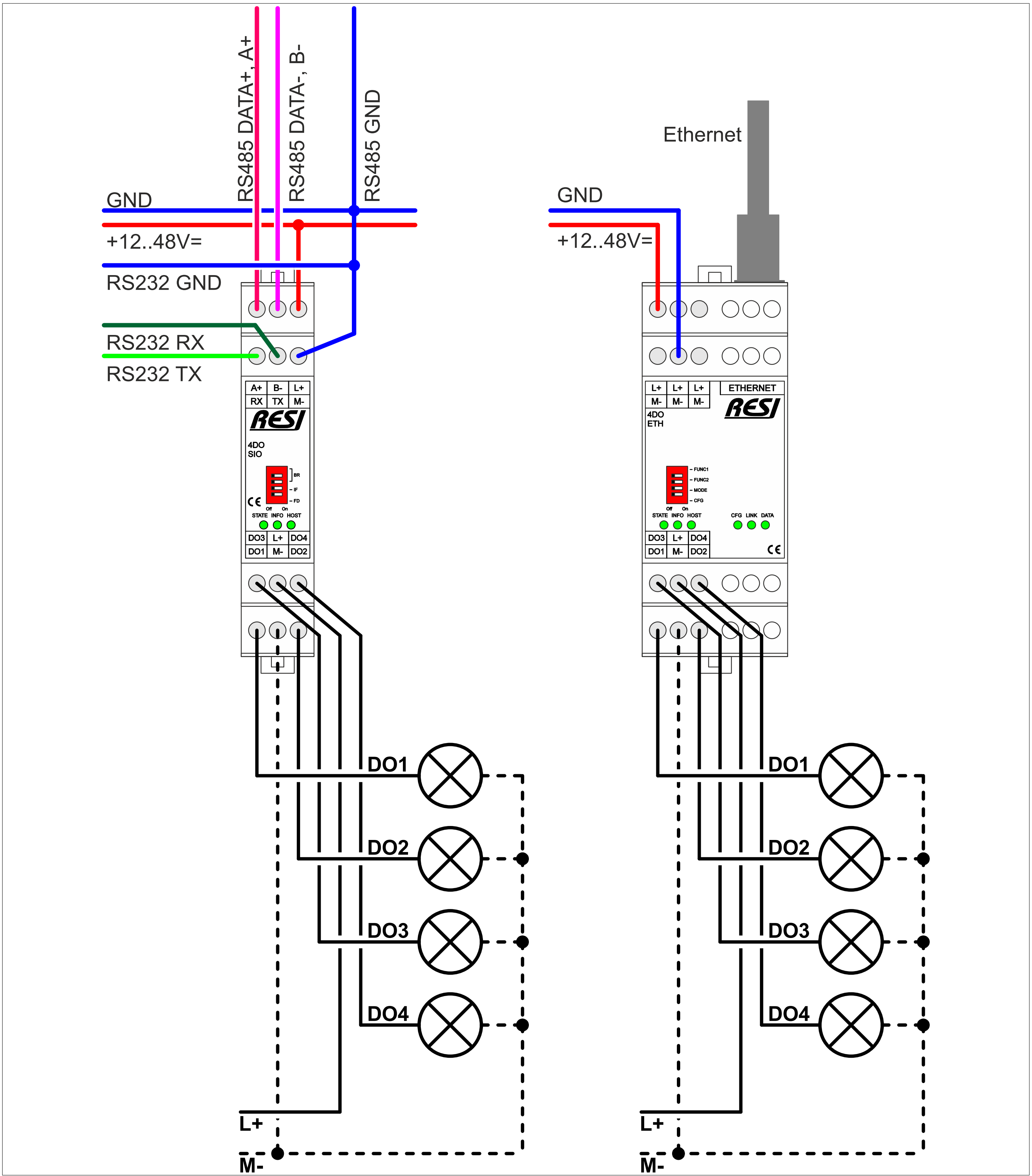


Figure: Schematics for the IO modules

32.5 RESI-4DO-SIO,ETH: Wiring diagram

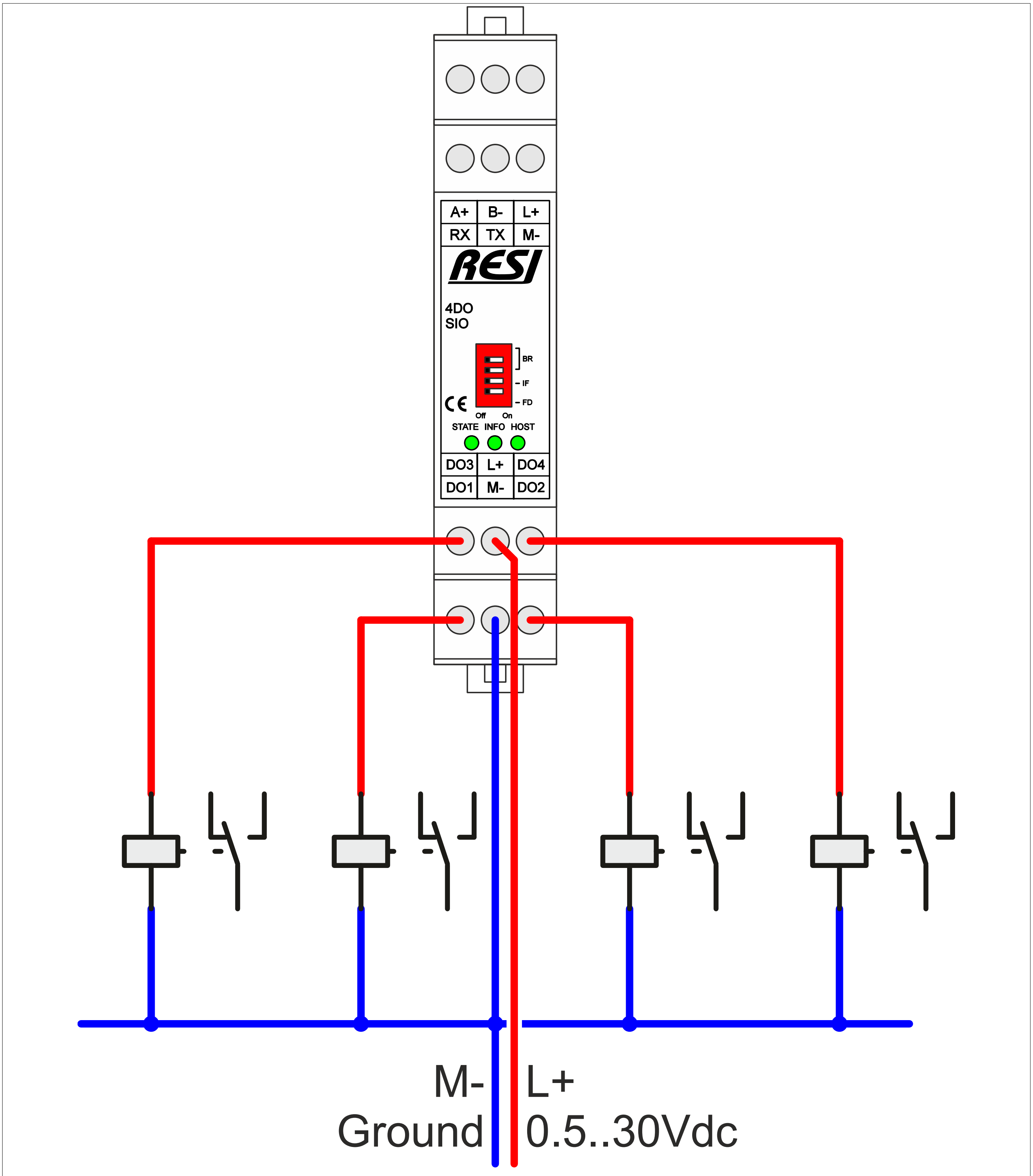


Figure: Wiring diagram for the IO modules with relays

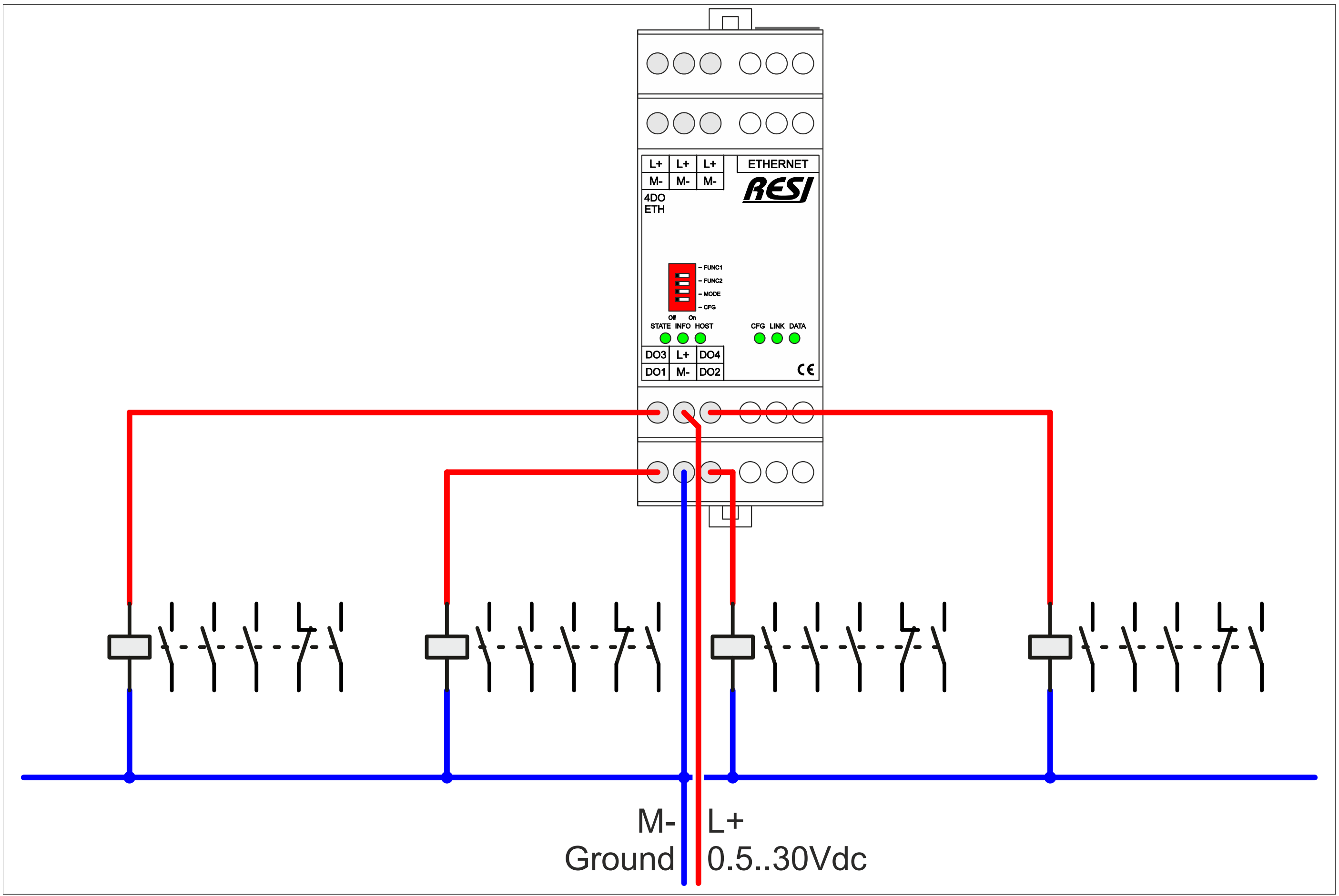


Figure: Wiring diagram for the IO modules with power contactors

32.6 Additional MODBUS register & coils

Here you will find only the additional MODBUS registers and coils especially for this IO module. Please refer to the description of of the standard MODBUS mapping for more details about the available basic MODBUS registers and coils.

Please refer to the external document for detailed documentation of the current MODBUS register mapping for this IO module:

RESI-L-4DO-SIO-ETH-MODBUS+ASCII-ENxx.pdf

32.7 Additional ASCII commands

Here you will find only the additional ASCII commands especially for this IO module. Please refer to the description of of the standard commands for more details about the available basic ASCII commands.

Please refer to the external document for detailed documentation of the current ASCII commands for this IO module:

RESI-L-4DO-SIO-ETH-MODBUS+ASCII-ENxx.pdf

33 RESI-2SSR-1A-SIO, RESI-2SSR-1A-ETH

33.1 General information

This series of IO modules offer the following features:

- 2 solid state relays for max. 600V~= and max. 1A output current per output
- Galvanic insulation with the solid state relay
- RESI-xxx-SIO: Galvanic isolated RS232 and RS485 interface for communication with a host system
- RESI-xxx-ETH: Galvanic isolated Ethernet interface for communication with a host system

Figure: Our serial IO module

Figure: Our Ethernet IO module

33.2 Technical specification

Beside the basic technical data, which fulfil all of our IO modules, this IO modules meet the following technical specifications:

Power consumption	
RESI-2SSR-1A-SIO	<0.7W
RESI-2SSR-1A-ETH	<1.1W
Product housing	
RESI-2SSR-1A-SIO	CEM17
RESI-2SSR-1A-ETH	CEM35
Product weight	
RESI-2SSR-1A-SIO	60g
RESI-2SSR-1A-ETH	94g
Solid state outputs	
Number	2
Output voltage	max. 600Vac/dc
Output current	max. 1A
Cable connection	via terminals
Galvanic isolation	Yes, with the solid state relay itself
Default serial settings	
Baud rate	via DIP switch
Parity	none
Stopbits	one
UnitID	255
Default Ethernet settings	
IP address	192.168.0.44
IP mask	255.255.255.0
gateway	192.168.0.1
UnitID	255
User	RESI
password	RESI

33.3 Additional terminals & LED states

SOLID STATE RELAYS	2 solid state relays for max. 600Vac/dc, 1A	
	Two 3 pin terminal blocks	
	Terminal type:	USLIM
	S1A, S1B:	Solid state relay #1 Form A maker contact (NO)
	S2A, S2B:	Solid state relay #2 Form A maker contact (NO)
Pin layout	S1A:	Solid state relay #1 maker contact (NO) 1
	N/C:	Not connected
	S1B:	Solid state relay #1 maker contact (NO) 2
	S2A:	Solid state relay #2 maker contact (NO) 1
	N/C:	Not connected
	S2B:	Solid state relay #2 maker contact (NO) 2
INFO	If one of the solid state relays is activated (ON), this LED is ON.	
	If none of the solid state relays is activated (OFF), this LED is OFF.	

33.4 RESI-2SSR-1A-SIO,ETH: Schematic diagram

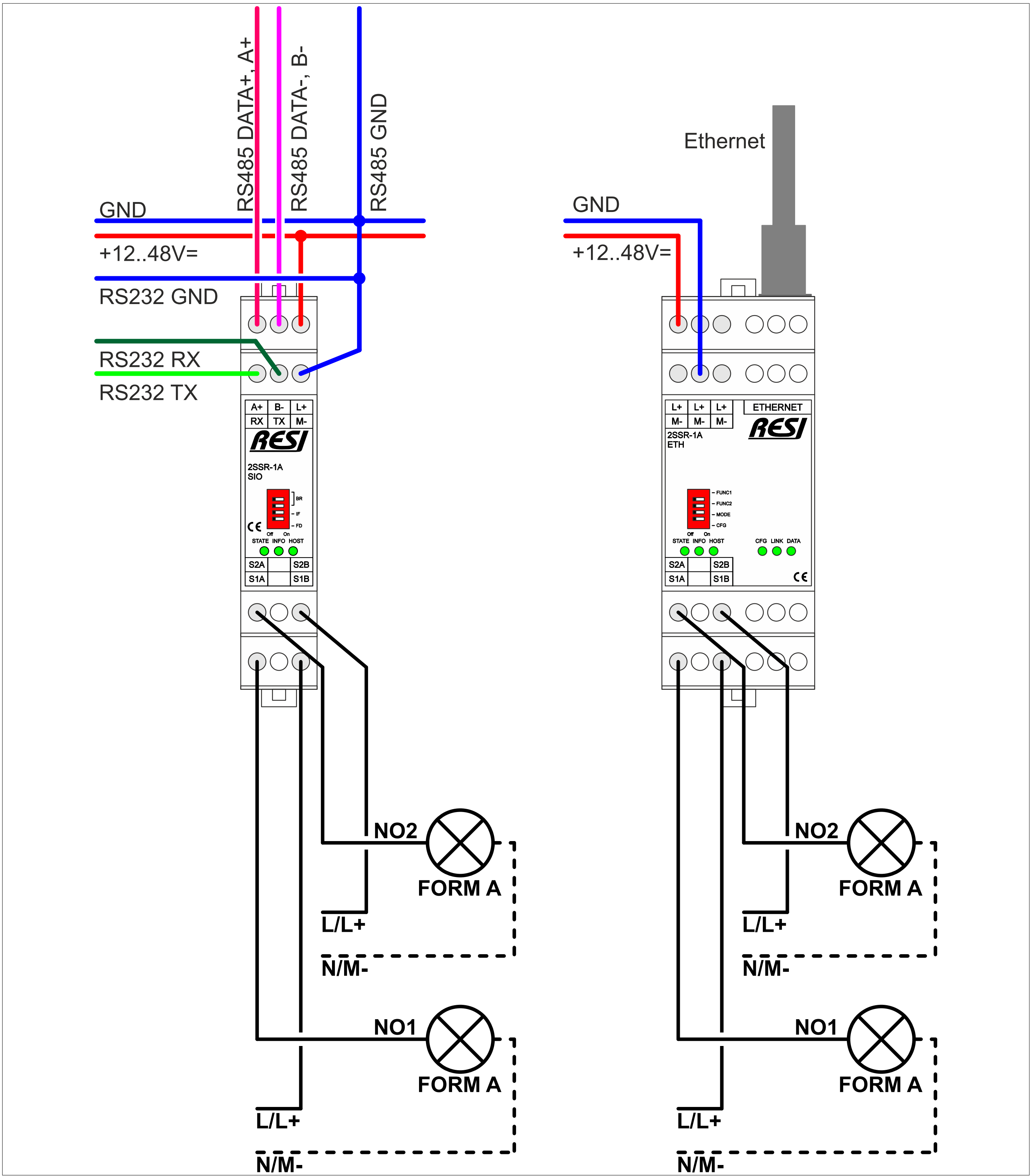


Figure: Schematics for the IO modules

33.5 RESI-2SSR-1A-SIO,ETH: Wiring diagram

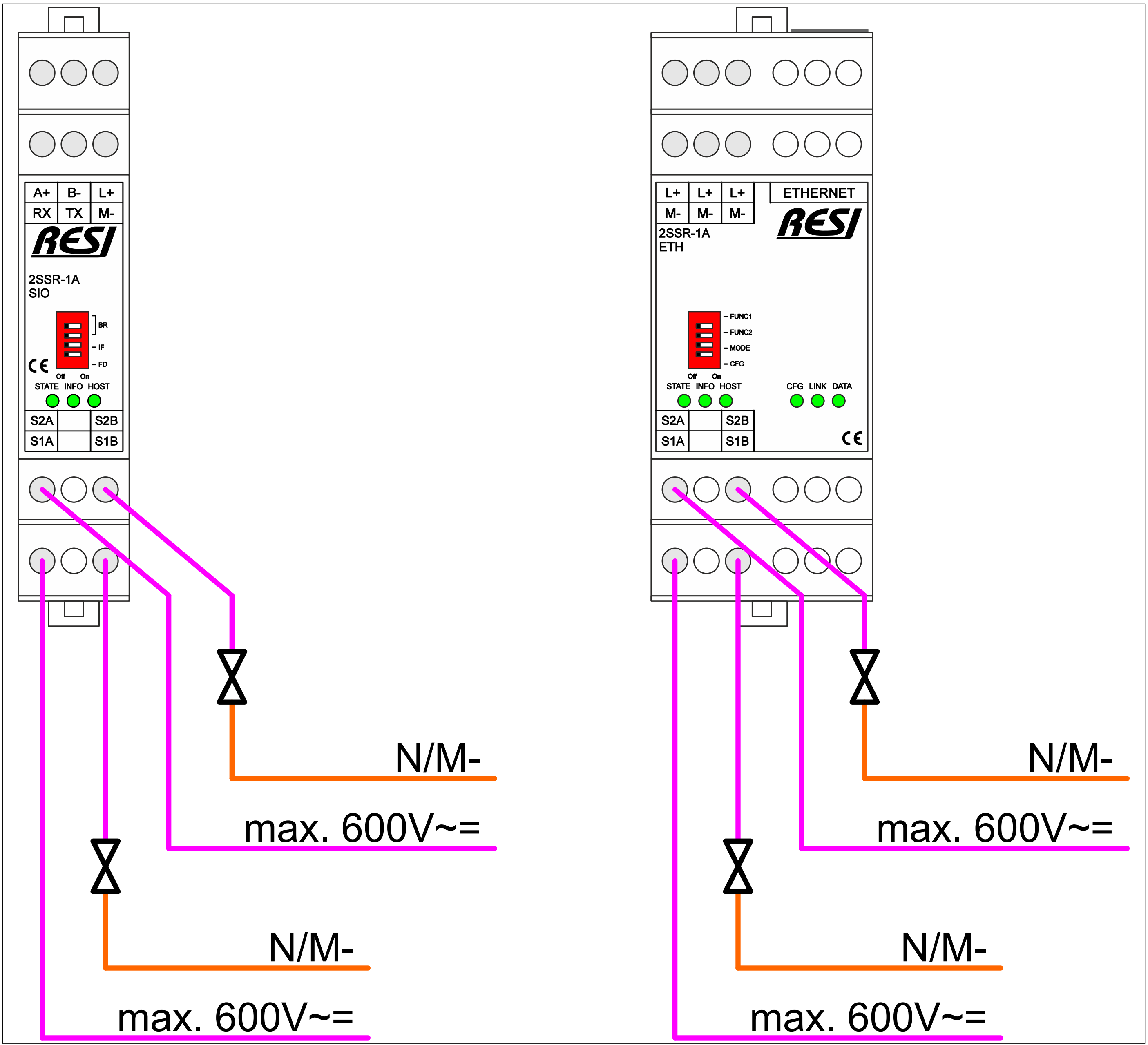


Figure: Wiring diagram for the IO modules

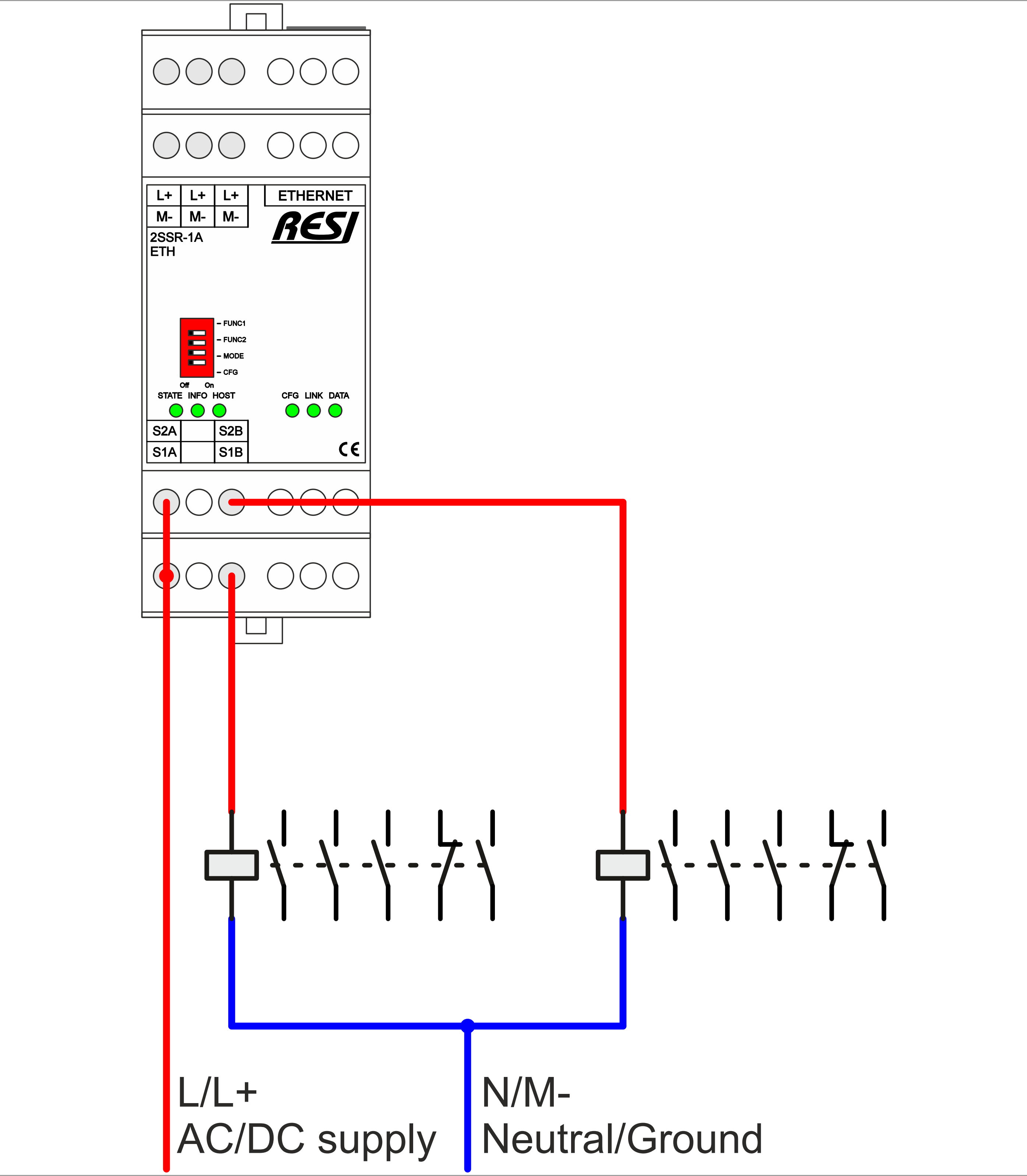


Figure: Wiring diagram for the IO modules with power contactors

33.6 Additional MODBUS register & coils

Here you will find only the additional MODBUS registers and coils especially for this IO module. Please refer to the description of of the standard MODBUS mapping for more details about the available basic MODBUS registers and coils.

Please refer to the external document for detailed documentation of the current MODBUS register mapping for this IO module:

RESI-L-2SSR-xA-SIO-ETH-MODBUS+ASCII-ENxx.pdf

33.7 Additional ASCII commands

Here you will find only the additional ASCII commands especially for this IO module. Please refer to the description of of the standard commands for more details about the available basic ASCII commands.

Please refer to the external document for detailed documentation of the current ASCII commands for this IO module:

RESI-L-2SSR-xA-SIO-ETH-MODBUS+ASCII-ENxx.pdf

34 RESI-2SSR-6A-SIO, RESI-2SSR-6A-ETH

34.1 General information

This series of IO modules offer the following features:

- 2 solid state relays for max. 60V~= and max. 6A output current per output
- Galvanic insulation with the solid state relay
- RESI-xxx-SIO: Galvanic isolated RS232 and RS485 interface for communication with a host system
- RESI-xxx-ETH: Galvanic isolated Ethernet interface for communication with a host system

Figure: Our serial IO module

Figure: Our Ethernet IO module

34.2 Technical specification

Beside the basic technical data, which fulfil all of our IO modules, this IO modules meet the following technical specifications:

Power consumption	
RESI-2SSR-1A-SIO	<0.7W
RESI-2SSR-1A-ETH	<1.1W

Product housing	
RESI-2SSR-1A-SIO	CEM17
RESI-2SSR-1A-ETH	CEM35

Product weight	
RESI-2SSR-1A-SIO	60g
RESI-2SSR-1A-ETH	94g

Solid state outputs	
Number	2
Output voltage	max. 60Vac/dc
Output current	max. 6A
Cable connection	via terminals
Galvanic isolation	Yes, with the solid state relay itself

Default serial settings	
Baud rate	via DIP switch
Parity	none
Stopbits	one
UnitID	255

Default Ethernet settings	
IP address	192.168.0.44
IP mask	255.255.255.0
gateway	192.168.0.1
UnitID	255

User	RESI
password	RESI

34.3 Additional terminals & LED states

SOLID STATE RELAYS	2 solid state relays for max. 60Vac/dc, 6A	
	Two 3 pin terminal blocks	
	Terminal type:	USLIM
	S1A, S1B:	Solid state relay #1 Form A maker contact (NO)
	S2A, S2B:	Solid state relay #2 Form A maker contact (NO)
Pin layout	S1A:	Solid state relay #1 maker contact (NO) 1
	N/C:	Not connected
	S1B:	Solid state relay #1 maker contact (NO) 2
	S2A:	Solid state relay #2 maker contact (NO) 1
	N/C:	Not connected
	S2B:	Solid state relay #2 maker contact (NO) 2
INFO	If one of the solid state relays is activated (ON), this LED is ON.	
	If none of the solid state relays is activated (OFF), this LED is OFF.	

34.4 RESI-2SSR-6A-SIO,ETH: Schematic diagram

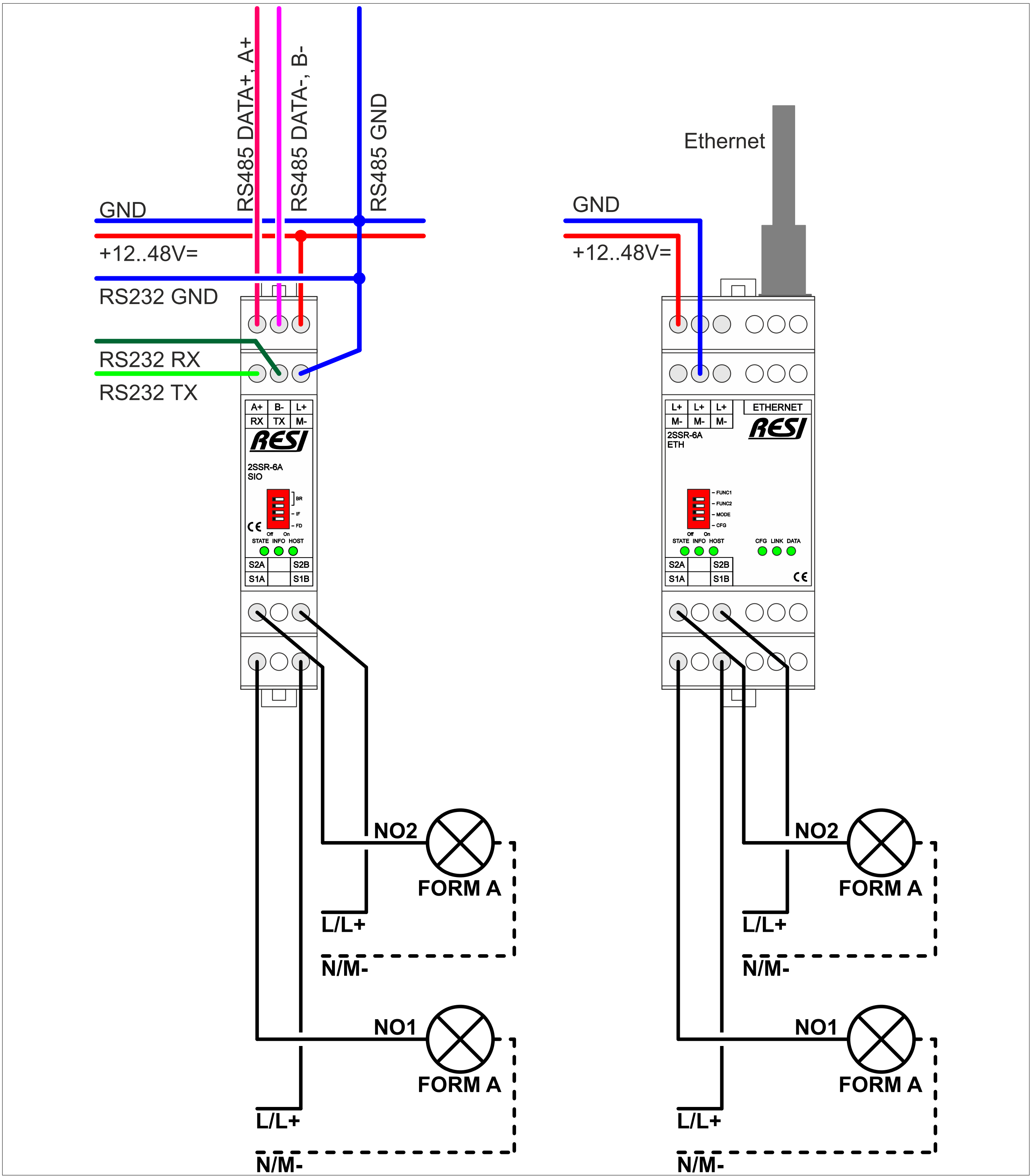


Figure: Schematics for the IO modules

34.5 RESI-2SSR-6A-SIO,ETH: Wiring diagram

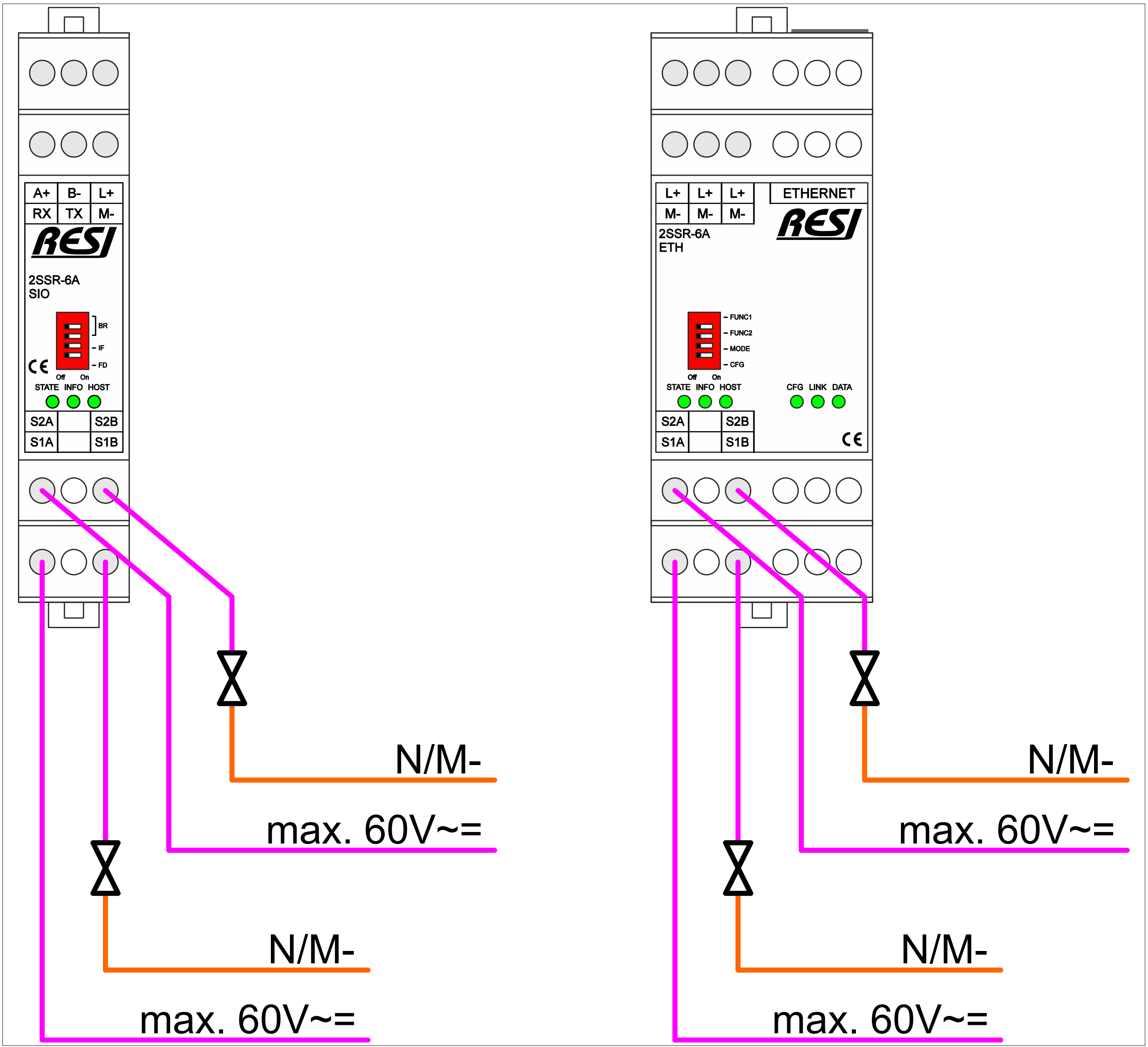


Figure: Wiring diagram for the IO modules

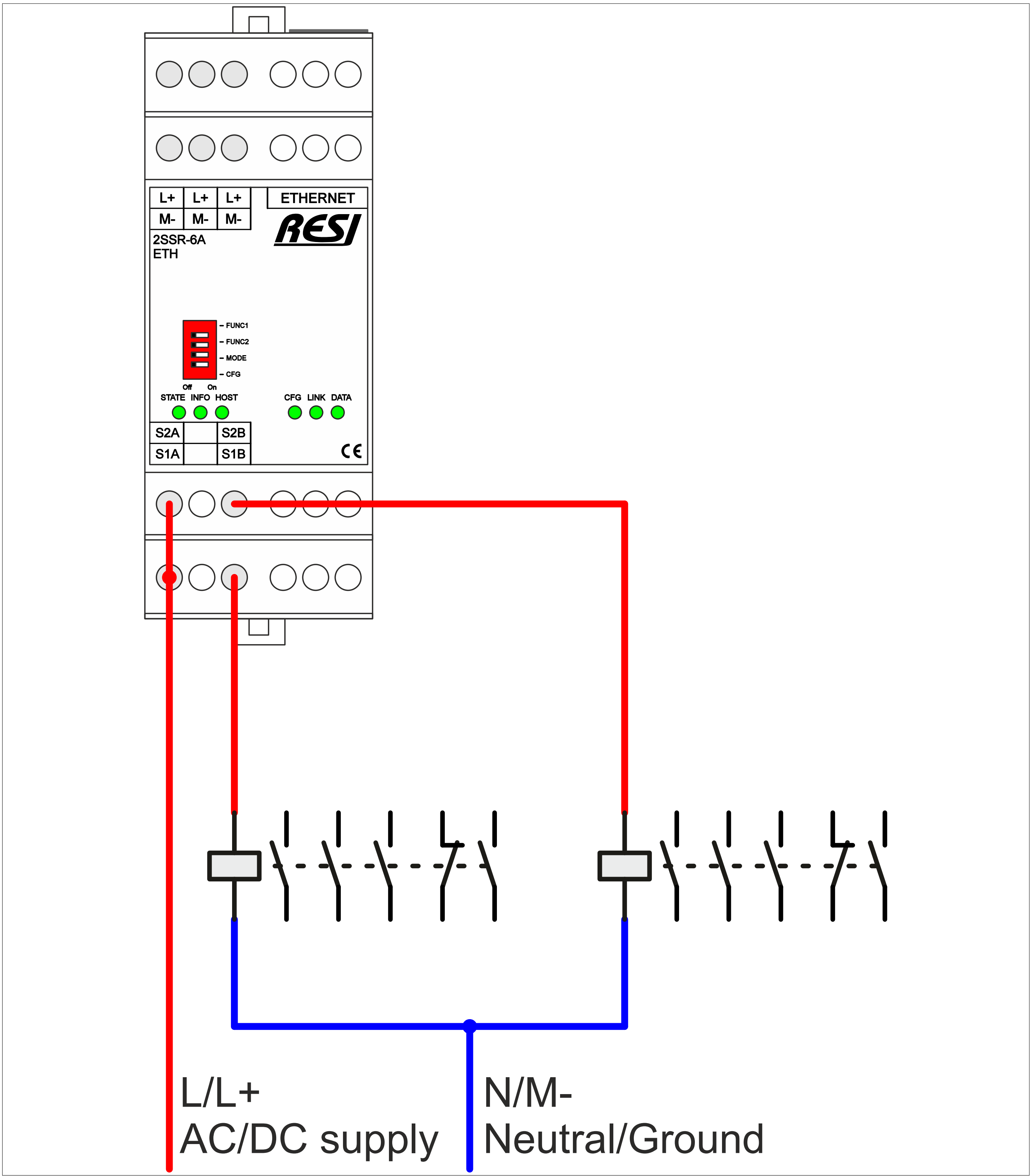


Figure: Wiring diagram for the IO modules with power contactors

34.6 Additional MODBUS register & coils

Here you will find only the additional MODBUS registers and coils especially for this IO module. Please refer to the description of of the standard MODBUS mapping for more details about the available basic MODBUS registers and coils.

Please refer to the external document for detailed documentation of the current MODBUS register mapping for this IO module:

RESI-L-2SSR-xA-SIO-ETH-MODBUS+ASCII-ENxx.pdf

34.7 Additional ASCII commands

Here you will find only the additional ASCII commands especially for this IO module. Please refer to the description of of the standard commands for more details about the available basic ASCII commands.

Please refer to the external document for detailed documentation of the current ASCII commands for this IO module:

RESI-L-2SSR-xA-SIO-ETH-MODBUS+ASCII-ENxx.pdf

35 RESI-2RTD-SIO, RESI-2RTD-ETH

35.1 General information

This series of IO modules offer the following features:

- 2 sensor inputs for temperature sensors
- Measurement accuracy +/-0.1%
- Measurement resolution +/-0.001%
- Measurement range -200°C…+850°C
- Various sensor types are applicable: PT100, PT1000, PT10, PT50, PT200, PT500, NI120, NI1000-DIN43760
- Various standards for linearisation are select-able: Europa, America, Japan, ITS-90
- Output of the temperatures in °Celsius [°C], °Fahrenheit [°F] or °Kelvin [°K]
- Different measurement currents are select-able: 5µA, 10µA, 25µA, 50µA, 100µA, 250µA,500µA, 1mA
- Various sensor connection types: 2 wire, 3 wire or 4 wire sensors connectable
- Internal calculation of an average temperature per channel
- RESI-xxx-SIO: Galvanic isolated RS232 and RS485 interface for communication with a host system
- RESI-xxx-ETH: Galvanic isolated Ethernet interface for communication with a host system

Figure: Our serial IO module

Figure: Our Ethernet IO module

35.2 Technical specification

Beside the basic technical data, which fulfil all of our IO modules, this IO modules meet the following technical specifications:

Power consumption	
RESI-2RTD-SIO	<0.8W
RESI-2RTD-ETH	<1.2W
Product housing	
RESI-2RTD-SIO	CEM17
RESI-2RTD-ETH	CEM35
Product weight	
RESI-2RTD-SIO	61g
RESI-2RTD-ETH	95g
Temperature inputs	
Number	2
Signal type	Temperature measurement
Measurement type	Measurement of resistance
ADC	24 bit sigma delta ADC
Accuracy	+/-0.1°C for PT-100, PT-200,PT-500, PT-1000
	+/-0.1°C NI-120, NI-1000-DIN43760
	+/-3°C for PT-10, PT-50
Resolution	+/-0.001°C
Reference stability	10ppm/°C
Sensor types	PT-100, PT-1000, PT-1000 α=0.00375, PT-10, PT-50, PT-200,
	PT-500, NI-120, NI-1000 DIN43760
Linearisation standards	Europa, America, Japan, ITS-90
Excitation current for measurement	5µA, 10µA, 25µA, 50µA, 100µA, 250µA, 500µA, 1mA
Cable connection	via terminals
Galvanic isolation	Yes, to the rest of the module, not to the other temperature input
Default serial settings	
Baud rate	via DIP switch
Parity	none
Stopbits	one
UnitID	255
Default Ethernet settings	
IP address	192.168.0.44
IP mask	255.255.255.0
gateway	192.168.0.1
UnitID	255
User	RESI
password	RESI

35.3 Additional terminals & LED states

TEMPERATURE INPUTS	2 RTD inputs for temperature sensors	
	Two 3 pin terminal blocks	
	Terminal type:	USLIM
	T1A, T1B, T1C:	Temperature input #1
	T2A, T2B, T2C:	Temperature input #2
Pin layout	T1A:	Temperature input #1 signal A
	T1B:	Temperature input #1 signal B
	T1C:	Temperature input #1 signal C
	T2A:	Temperature input #2 signal A
	T2B:	Temperature input #2 signal B
	T2C:	Temperature input #2 signal C
2 wire sensor:	Sensor is cabled between T1C and T1B	
	T1A:	bridged with T1B
	T1B:	bridged with T1A and sensor wire 2 (right cable of sensor)
	T1C:	sensor wire 1 (left cable of sensor)
3 wire sensor:	Sensor is cabled between T1C, T1B and T1A	
	T1A: Sensor cable 3 (right cable of sensor, 2nd cable)	
	T1B: Sensor cable 2 (right cable of sensor, 1st cable)	
	T1C: Sensor cable 1 (left cable of sensor)	
4 wire sensor:		
	Sensor is cabled between T1C, T1B and T1A	
	T1A: Sensor cable 4 (right cable of sensor, 2nd cable)	
	T1B: Sensor cable 3 (right cable of sensor, 1st cable)	
	T1C: Sensor cable 1+2 (both cables on the left side of sensor)	
INFO	If everything is ok this LED is on. If there is an internal error	
	with the temperature measurement, this LED flashes fast.	

35.4 RESI-2RTD-SIO,ETH: Schematic diagram

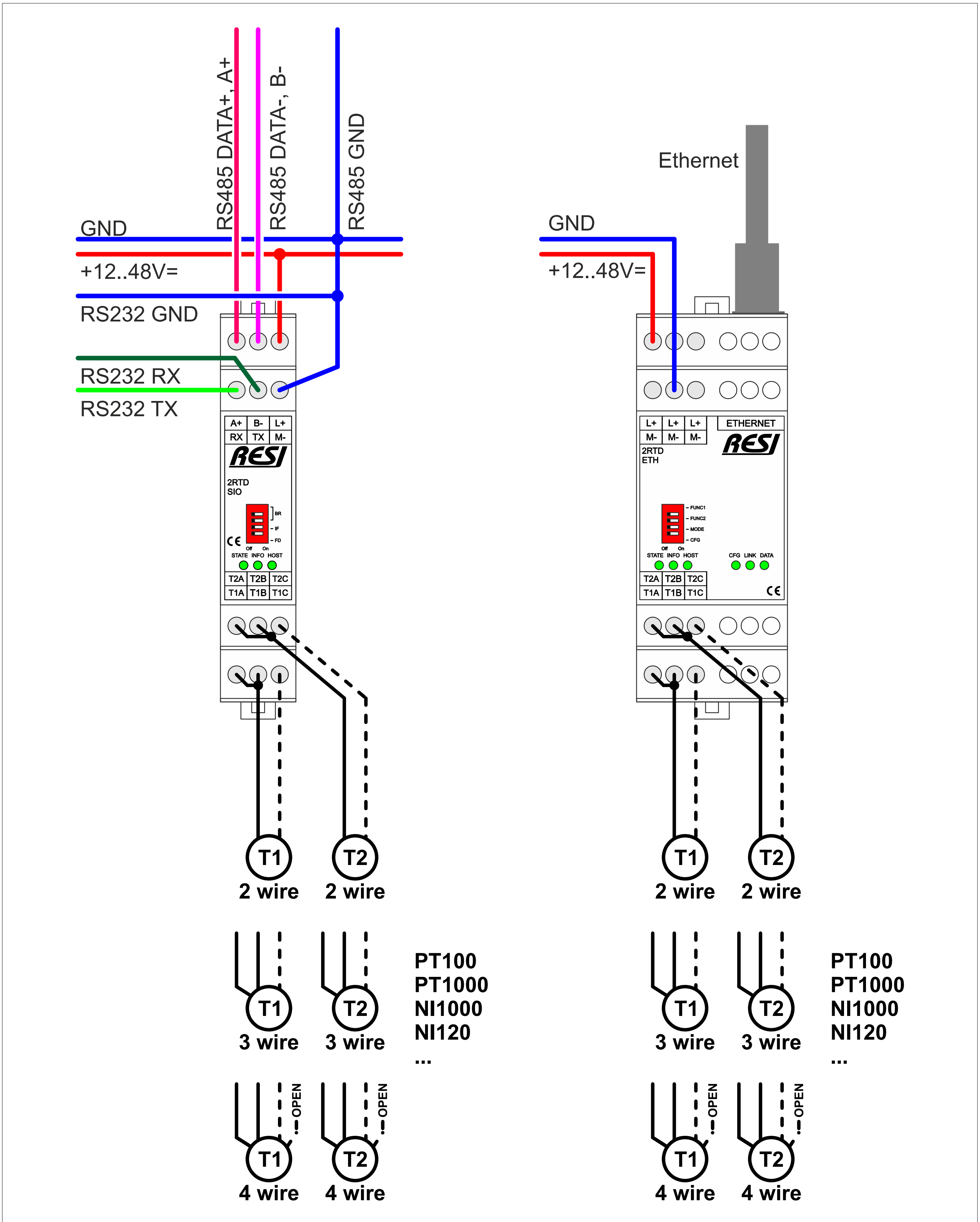


Figure: Schematics for the IO modules

35.5 Cabling of temperature sensors

A typical temperature sensor with different connection cables is shown in the figure below:

- 2 wire: A red and white cable
- 3 wire: Two red and one white cable
- 4 wire: Two read and two white cable

The sensor element is always mounted between the red and white cables!

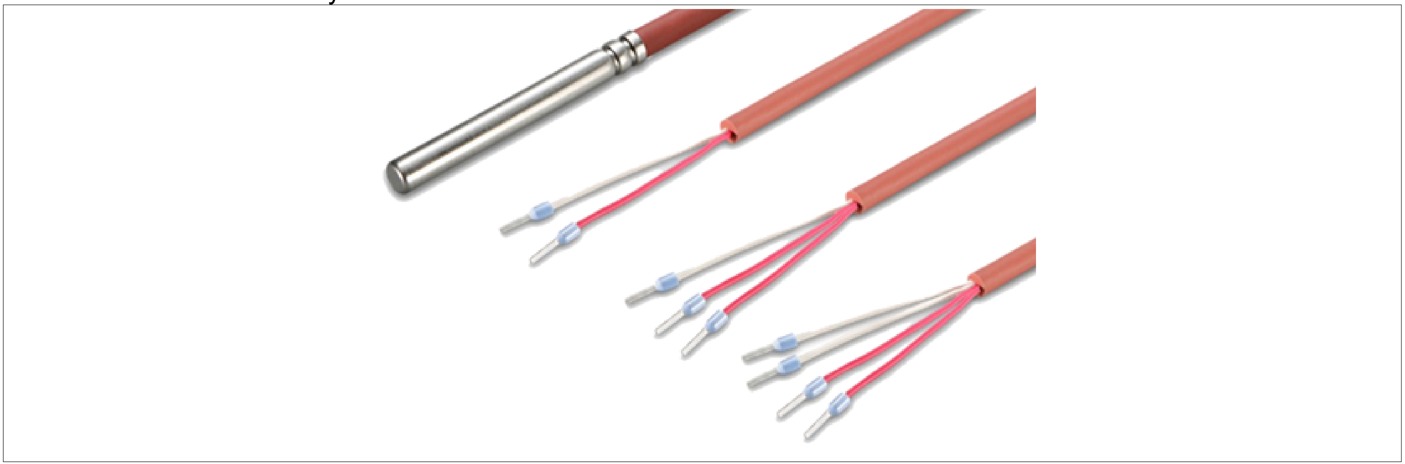


Figure: Typical temperature sensor with different connection cables

35.5.1 Wiring of 2 wire sensors

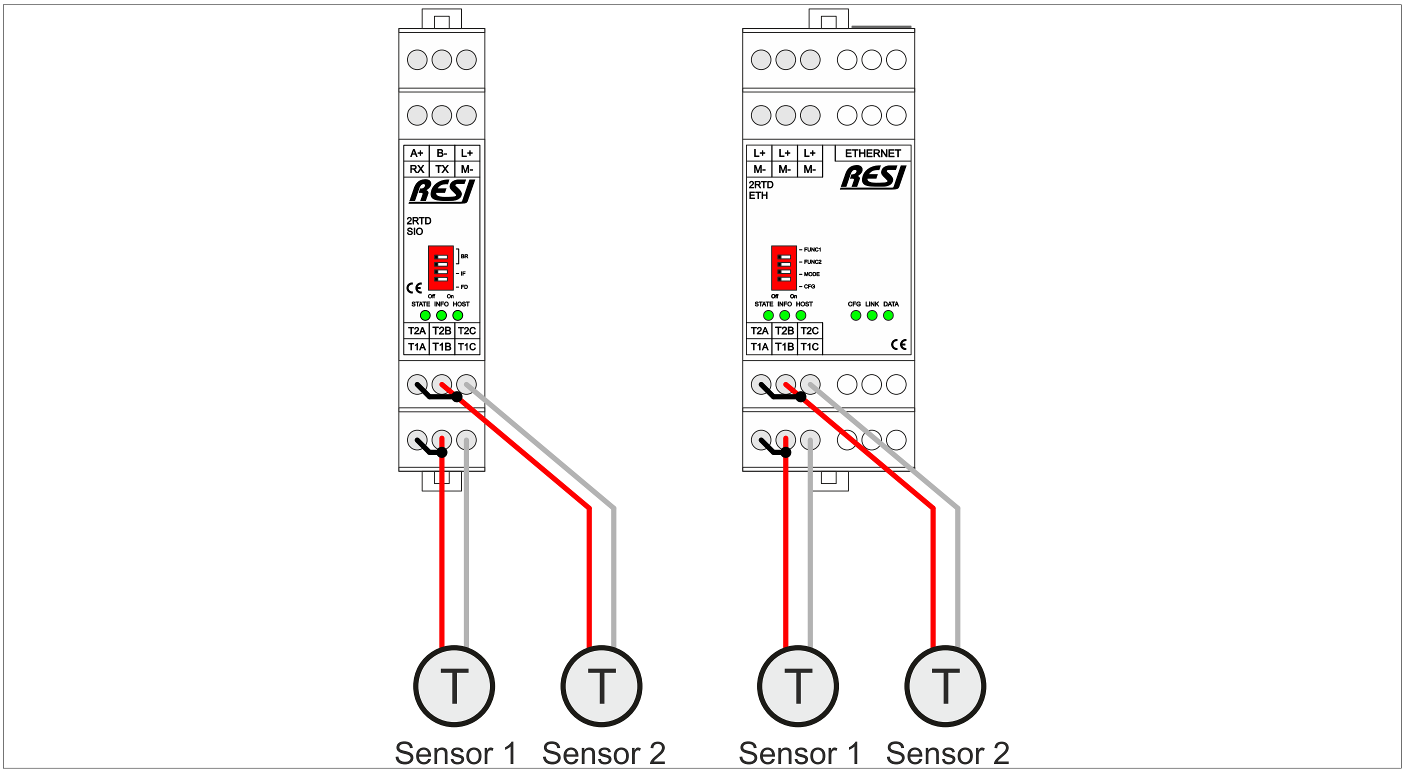


Figure: Wiring of two 2 wire sensors to our module

IMPORTANT: Due to the reason, that our module is doing always a 3 wire measurement, you have to set always a bridge cable between the clamps TxA and TxB! Keep this bridge cable between the two terminals as short as possible!

35.5.2 Wiring of 3 wire sensors

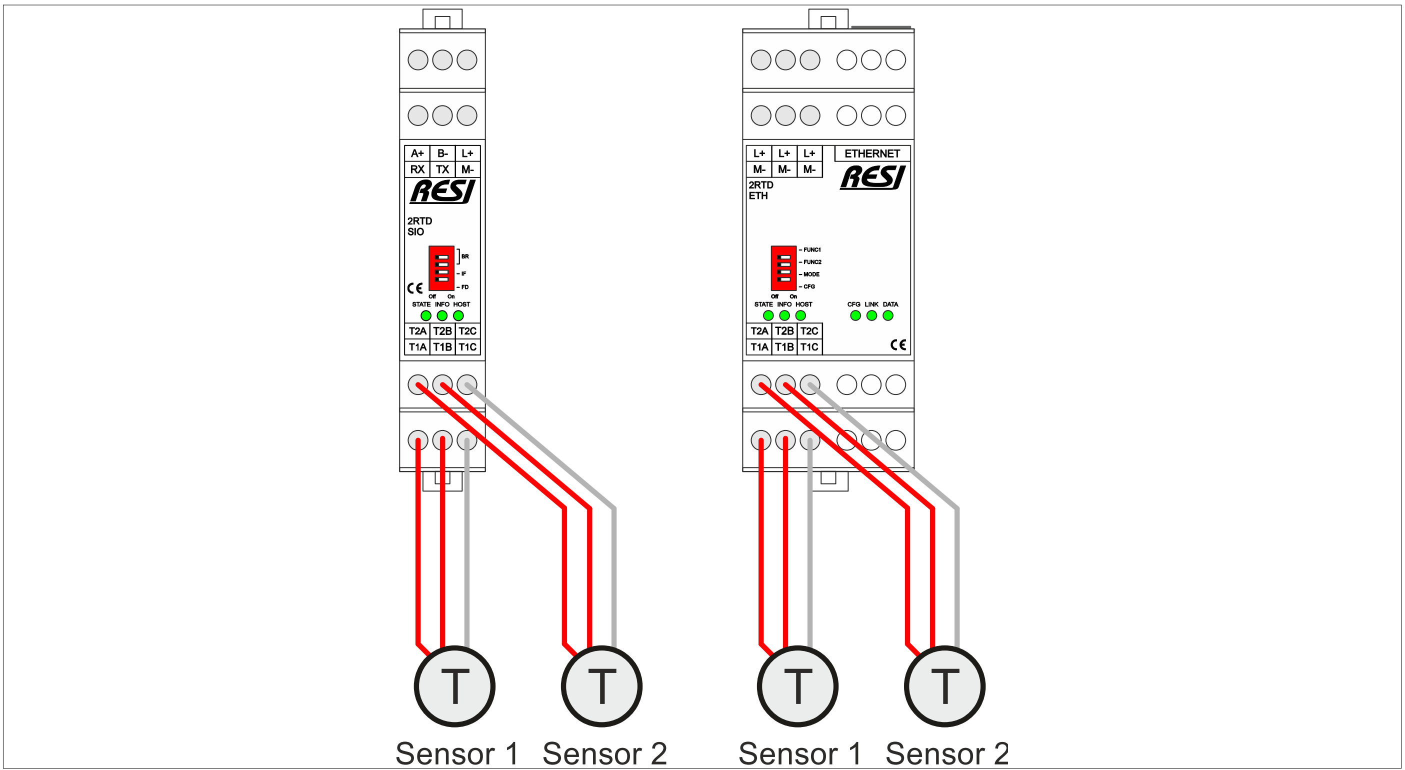


Figure: Wiring of two 3 wire sensors to our module

35.5.3 Wiring of 4 wire sensors

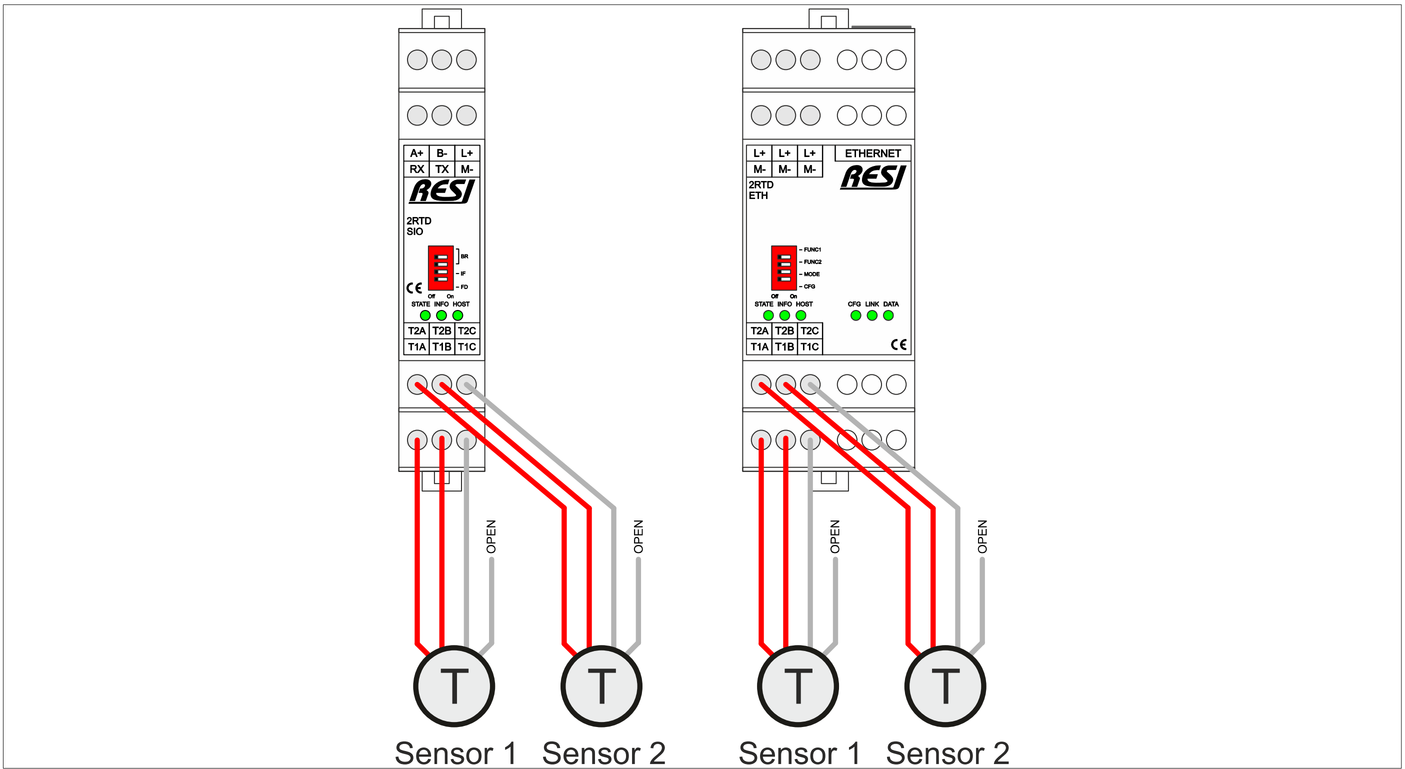


Figure: Wiring of two 4 wire sensors to our module. Do not connect the fourth cable to the module!

35.6 Useable sensor types and measurement accuracy

This section describes the suitable sensors and explains the measurement accuracy of the sensor inputs of the module.

HINT: Use our free software RESI MODBUSConfigurator to configure and test our 2RTD module. You can also use your own software to handle the complete configuration while writing to MODBUS/RTU registers or with ASCII text commands.

35.6.1.1 Useable sensor types

The following types of sensors can be used per input:

Platin sensors:
- PT-100 sensors: Measurement range from 1.95Ω to 34.5Ω, -200°C to +850°C
- PT-1000 sensors: Measurement range from 195Ω to 3450Ω, -200°C to +850°C
- PT-1000 sensors with an α=0.00375: Measurement range from 195Ω to 3450Ω, -200°C to +850°C
- PT-10 sensors: Measurement range from 1.95Ω to 34.5Ω, -200°C to +850°C
- PT-50 sensors: Measurement range from 9.75Ω to 172.5Ω, -200°C to +850°C
- PT-200 sensors: Measurement range from 39Ω to 690Ω, -200°C to +850°C
- PT-500 sensors: Measurement range from 97.5Ω to 1725Ω, -200°C to +850°C

Nickel sensors:
- NI-120 sensors: Measurement range from 66.6Ω to 380.3Ω, -80°C to +260°C
- NI-1000 DIN43760 sensors: Sensors with linearisation according to DIN43760

Each of the two sensor inputs of the module can measure a different sensor type!

You can use all sensor accuracy classes (class AA, A, B, C). Please consult the DIN EN 60751:2009-05 for an exact definition of the sensor accuracy. Don't forget, that the whole measurement error for the temperature measurement consists always out of the error of the sensor element itself, the error of the used cabling and the measurement errors of the measurement electronic.

Out resistance measurement electronic uses an internal 2kΩ sense resistor. With an excitation current of 250µA the voltage drop on this resistor is 0.5V. This is the ideal range, to achieve the highest measurement accuracy. Use sensor type PT100, PT200, PT500, PT-1000, NI-120 or NI-1000 DIN43760 to achieve the best accuracy of our module with +/-0.1°C.

For PT10 and PT50 sensors this internal sense resistor is too big. So the reachable accuracy lies only about +/-3°C.

35.6.1.2 Configurable excitation current

For each input you can define an individual excitation current for the measurement:

- 5µA
- 10µA
- 25µA
- 50µA
- 100µA
- 250µA
- 500µA
- 1mA

The electronic executes an internal reference measurement on an Rsense resistor with 2kΩ (Accuracy +/-0.05%). Also we use internal resistors to protect the inputs. Please adjust the excitation current for each channel in a way, that the resulting maximum voltage drop on this internal Rsense resistor <=0.5V.

U=R*I -> U=2kΩ*250µA -> 0.5V

This results in a maximum excitation current of 250µA with this module. If the excitation current exceeds this voltage range, the module signals this error with „ADC-Out-of-Range“ in the status flags of each channel.

The ideal excitation current of the module is 250µA! With smaller excitation currents the measurement will be more and more inaccurate!

35.6.1.3 Selectable linearisation standard

A PLATIN resistor (PT sensor) is defined with a standardized characteristic. This is the Callendar-Van Dusen equation:

This is defined as follows:

RT = R0 • (1 + a • T + b • T2 + (T – 100°C) • c • T3) for T < 0°C,
RT = R0 • (1 + a • T + b • T2) for T > 0°C

The equation is used with different coefficients depending of the selected linearisation standard to calculate a temperature from the measured resistor.

STANDARD	ALPHA (α)	a	b	c
Europe DIN EN 60751 IEC 751 JIS C1604-1997	α=0x00385	$3.908300*10^{-03}$	$–5.775000*10^{-07}$	$–4.183000*10^{-12}$
America SAMA Standard	α=0x003911	$3.969200*10^{-03}$	$–5.849500*10^{-07}$	$–4.232500*10^{-12}$
Japan JIS C1604-1987	α=0x003916	$3.973900*10^{-03}$	$–5.870000*10^{-07}$	$–4.400000*10^{-12}$
ITS-90	α=0x003926	$3.984800*10^{-03}$	$–5.870000*10^{-07}$	$–4.400000*10^{-12}$
RTD-1000-375	α=0x00375	$3.810200*10^{-03}$	$–6.018880*10^{-07}$	$–6.000000*10^{-12}$
NI-120	N/A	N/A	N/A	N/A

35.6.1.4 Sensor evaluation and accuracy

Our module computes the final temperature value °Celsius [°C] and delivers this temperature on various MODBUS registers in various number formats and via various ASCII commands to the host.

In addition our module can convert the temperature also in °Fahrenheit [°F] with the formula:

T[°F]=T[°C]*1.8+32

Also this temperature value can be read out with MODBUS Register or ASCII text commands. An own conversion on the host from Celsius in Fahrenheit is not necessary.

Also our module converts the temperature data into °Kelvin [°K] with the formula:

T[°K]=T[°C] +273.15

Also this temperature value can be read out with MODBUS Register or ASCII text commands. An own conversion on the host from Celsius in kelvin is not necessary.

Our module uses a 24 bit sigma/delta ADC with a noise suppression for 50/60Hz internally. Our module achieves a very high measurement accuracy of +/-0.1°C and a measurement resolution of +/-0.001°C!

Our module measures every channel around 1 time per second. In addition our module computes an average temperature for each channel with a user selectable time range in seconds, to suppress short noise signals in standard applications.

A manual adjustable zero offset allows a zero point shift to compensate static effects of the cabling, especially useful for 2 wire sensors.

Our module offers a very complex internal hardware to evaluate if the measured temperature is valid or not. Therefore the module offers for each channel a status representing the result of the last converted temperature. This status uses 8 bits, which have the following meaning:

BIT	NAME	DESCRIPTION
0	VALID	=1: If the measurement result is valid, this bit is set and all other bits in the status are 0! =0: if the system detects a conversion error or problem, this bit is 0 and the measurement result must be discarded!
1	ADC OUT OF RANGE	=1: If the product of 2kΩ * excitation current >0.5V, this bit is 1 and the measurement result is invalid. The absolute input voltage of the ACD beyond ±1.125 • VREF/2 =0: Everything is ok
2	SENSOR UNDER RANGE	=1: The current measured temperature is beyond the lower limit for the selected sensor type. For PT: -200°C, for NI-120: -80°C =0: Everything is ok
3	SENSOR OVER RANGE	=1: The current measured temperature is above the upper limit for the selected sensor type. For PT: +850°C, for NI-120: +260°C =0: Everything is ok
4	NOT USED	Ignore this bit
5	NOT USED	Ignore this bit
6	HARD ADC OUT OF RANGE	=1: Erroneous readout of the ADC value. A possibility is an extreme high noise level on the signal. The sensor value will be discarded. A second option is an open wiring for the sensor. =0: Everything is ok
7	SENSOR HARD FAULT	=1: Sensor wiring is open or no sensor is cabled to the module. Sensor has a shortcut or the internal sense resistor has an error. =0: Everything is ok

35.7 Additional MODBUS register & coils

Here you will find only the additional MODBUS registers and coils especially for this IO module. Please refer to the description of of the standard MODBUS mapping for more details about the available basic MODBUS registers and coils.

Please refer to the external document for detailed documentation of the current MODBUS register mapping for this IO module:

RESI-L-2RTD-SIO-ETH-MODBUS+ASCII-ENxx.pdf

35.8 Additional ASCII commands

Here you will find only the additional ASCII commands especially for this IO module. Please refer to the description of of the standard commands for more details about the available basic ASCII commands.

Please refer to the external document for detailed documentation of the current ASCII commands for this IO module:

RESI-L-2RTD-SIO-ETH-MODBUS+ASCII-ENxx.pdf

36 RESI-4AIU-SIO, RESI-4AIU-ETH

36.1 General information

This series of IO modules offer the following features:

- 4 high precision analog inputs for -10Vdc..+10Vdc signals (-10.24Vdc to +10.24Vdc)
- ADC resolution 16 bit, accuracy +/-0.1%
- RESI-xxx-SIO: Galvanic isolated RS232 and RS485 interface for communication with a host system
- RESI-xxx-ETH: Galvanic isolated Ethernet interface for communication with a host system

Figure: Our serial IO module

Figure: Our Ethernet IO module

36.2 Technical specification

Beside the basic technical data, which fulfil all of our IO modules, this IO modules meet the following technical specifications:

Power consumption	
RESI-4AIU-SIO	<0.8W
RESI-4AIU-ETH	<1.2W
Product housing	
RESI-4AIU-SIO	CEM17
RESI-4AIU-ETH	CEM35
Product weight	
RESI-4AIU-SIO	62g
RESI-4AIU-ETH	96g
Analog inputs	
Number	4
Update speed	Every 100ms
Range	-10V..+10V
ADC resolution	16 bit
Input voltage range	-10.24V..+10.24V
Accuracy	+/-0.1%
Cable connection	via terminals
Galvanic isolation	Yes
Default serial settings	
Baud rate	via DIP switch
Parity	none
Stopbits	one
UnitID	255
Default Ethernet settings	
IP address	192.168.0.51
IP mask	255.255.255.0
gateway	192.168.0.1
UnitID	255
User	RESI
password	RESI

36.3 Additional terminals & LED states

ANALOG INPUTS	4 analog inputs for -10V..0V..+10V signals	
	Two 3 pin terminal blocks	
	Terminal type:	USLIM
	C:	Ground for all analog inputs
	AI1-AI4:	Analog inputs
Pin layout	AI1:	Signal input for analog input #1
	AI2:	Signal input for analog input #2
	AI3:	Signal input for analog input #3
	AI4:	Signal input for analog input #4
	C:	Signal ground for analog inputs #1-#4
		Both signal grounds are internally bridged
INFO	If everything is OK, this LED is on. If there is an internal error at the analog inputs,	
	this LED flashes quickly.	

36.4 RESI-4AIU-SIO,ETH: Schematic diagram

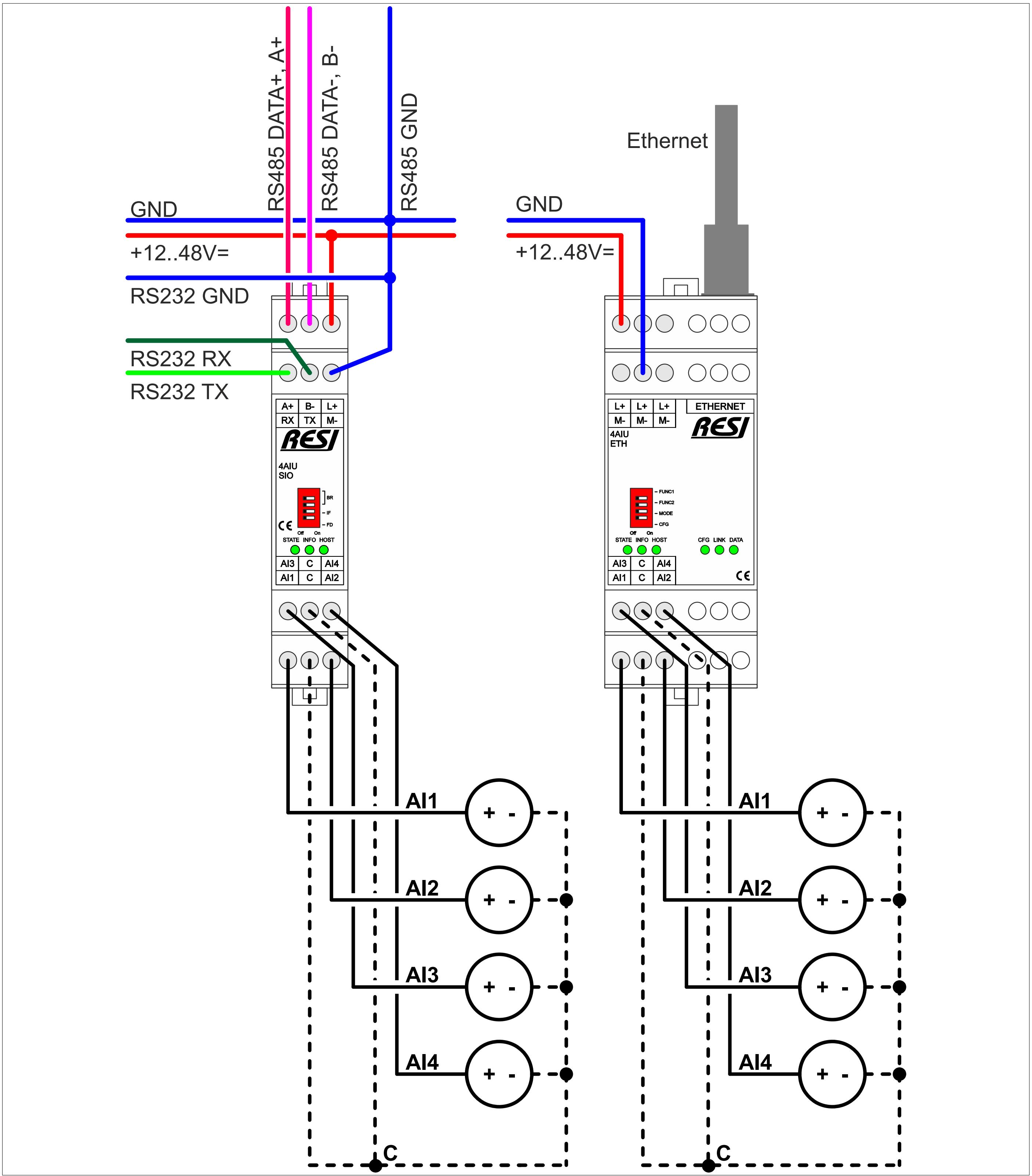


Figure: Schematics for the IO modules

36.5 RESI-4AIU-SIO,ETH: Wiring diagram

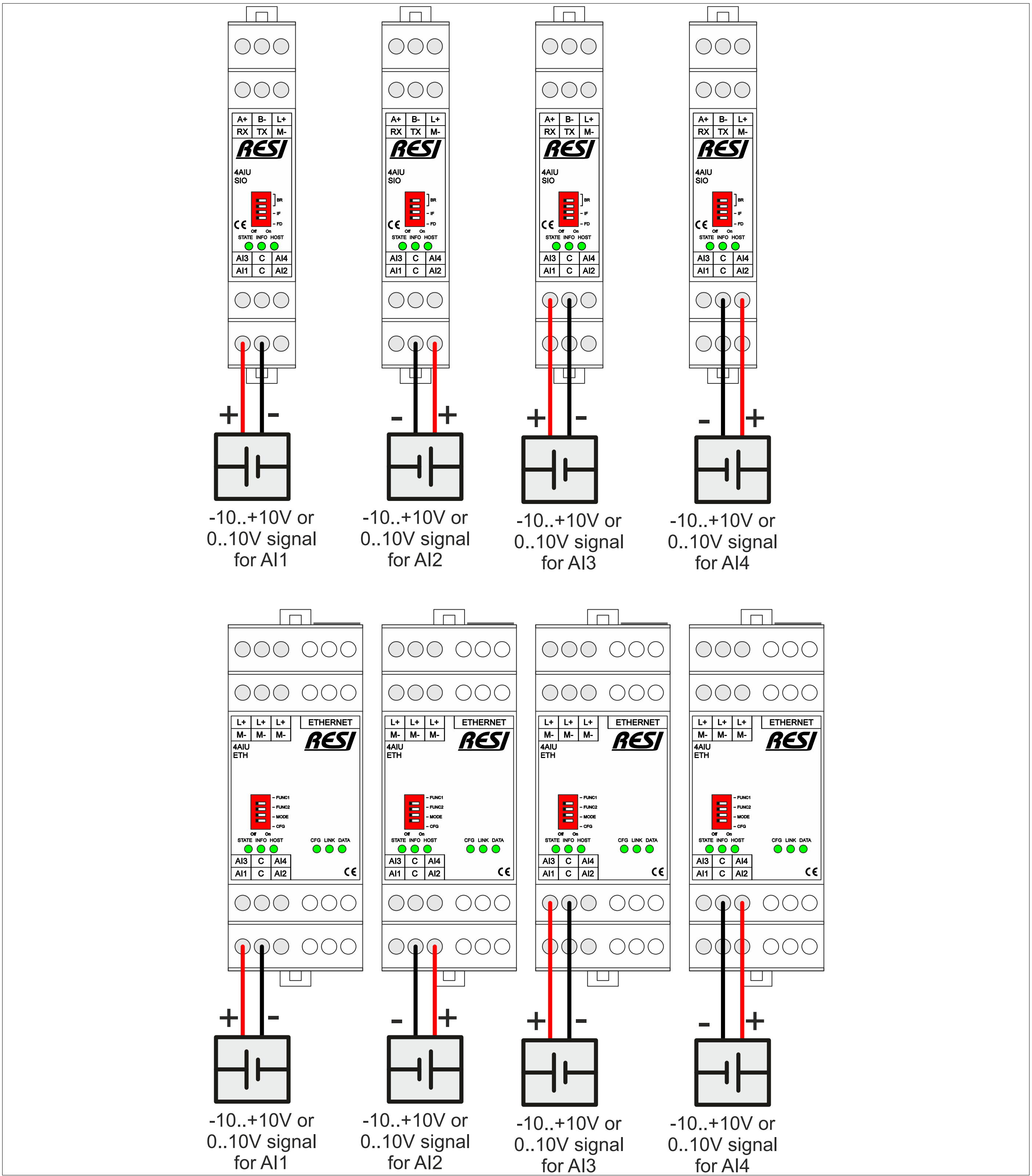


Figure: Wiring diagram for the IO modules

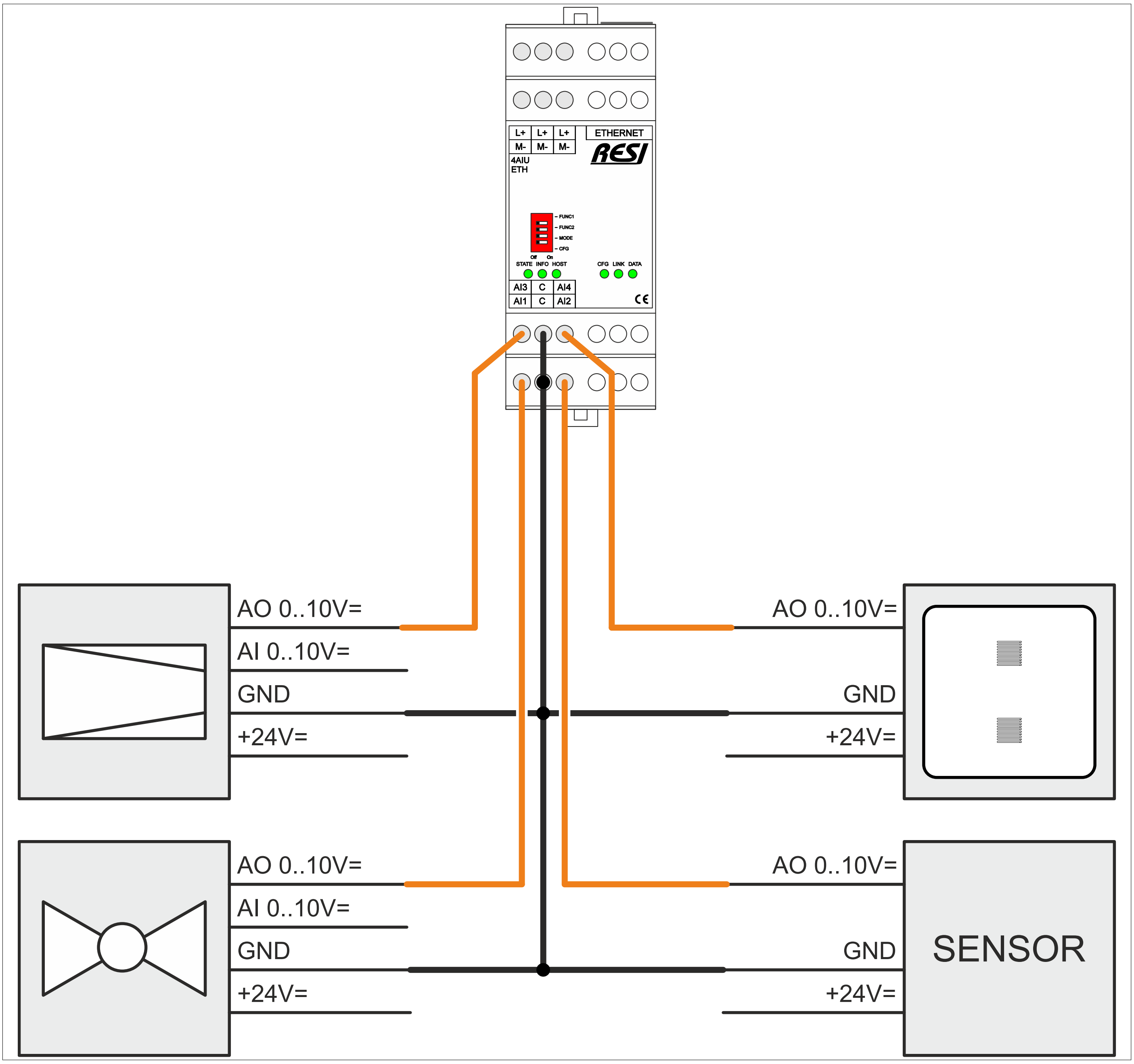


Figure: Wiring diagram for building automation devices

36.6 Additional MODBUS register & coils

Here you will find only the additional MODBUS registers and coils especially for this IO module. Please refer to the description of of the standard MODBUS mapping for more details about the available basic MODBUS registers and coils.

Please refer to the external document for detailed documentation of the current MODBUS register mapping for this IO module:

RESI-L-4AIU-SIO-ETH-MODBUS+ASCII-ENxx.pdf

36.7 Additional ASCII commands

Here you will find only the additional ASCII commands especially for this IO module. Please refer to the description of of the standard commands for more details about the available basic ASCII commands.

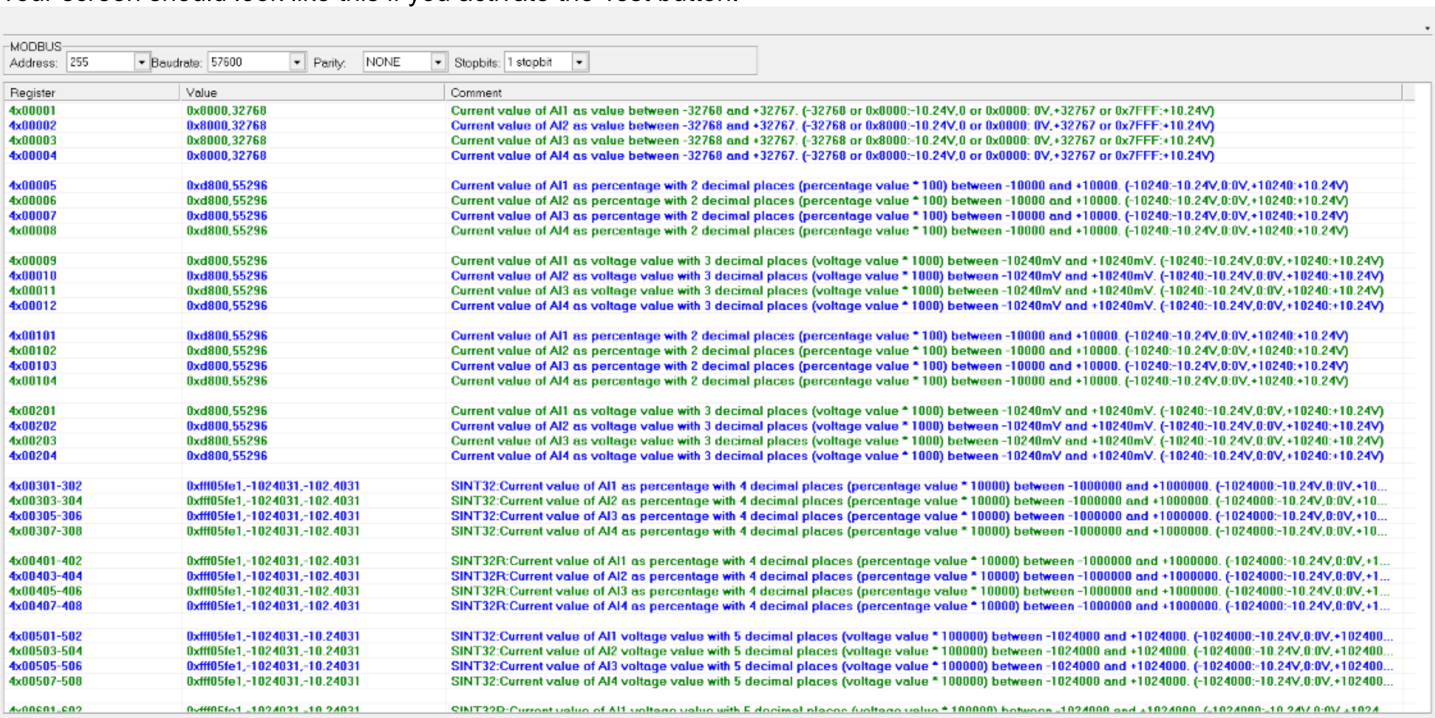
Please refer to the external document for detailed documentation of the current ASCII commands for this IO module:

RESI-L-4AIU-SIO-ETH-MODBUS+ASCII-ENxx.pdf

36.8 Additional MODBUSConverter software information

Click on the add to project button to open a dialog with all available IO modules. Then select the section SLIM-IO modules… and select RESI-4AIU-SIO or RESI-4AIU-ETH to add this device to your project. Or you search the connected module automatically.

Your screen should look like this if you activate the Test button.



Register	Value	Comment
4x00001	0x8000,32768	Current value of AI1 as value between -32768 and +32767. (-32768 or 0x8000:-10.24V,0 or 0x0000: 0V,+32767 or 0x7FFF:+10.24V)
4x00002	0x8000,32768	Current value of AI2 as value between -32768 and +32767. (-32768 or 0x8000:-10.24V,0 or 0x0000: 0V,+32767 or 0x7FFF:+10.24V)
4x00003	0x8000,32768	Current value of AI3 as value between -32768 and +32767. (-32768 or 0x8000:-10.24V,0 or 0x0000: 0V,+32767 or 0x7FFF:+10.24V)
4x00004	0x8000,32768	Current value of AI4 as value between -32768 and +32767. (-32768 or 0x8000:-10.24V,0 or 0x0000: 0V,+32767 or 0x7FFF:+10.24V)
4x00005	0xd800,55296	Current value of AI1 as percentage with 2 decimal places (percentage value * 100) between -10000 and +10000. (-10240:-10.24V,0:0V,+10240:+10.24V)
4x00006	0xd800,55296	Current value of AI2 as percentage with 2 decimal places (percentage value * 100) between -10000 and +10000. (-10240:-10.24V,0:0V,+10240:+10.24V)
4x00007	0xd800,55296	Current value of AI3 as percentage with 2 decimal places (percentage value * 100) between -10000 and +10000. (-10240:-10.24V,0:0V,+10240:+10.24V)
4x00008	0xd800,55296	Current value of AI4 as percentage with 2 decimal places (percentage value * 100) between -10000 and +10000. (-10240:-10.24V,0:0V,+10240:+10.24V)
4x00009	0xd800,55296	Current value of AI1 as voltage value with 3 decimal places (voltage value * 1000) between -10240mV and +10240mV. (-10240:-10.24V,0:0V,+10240:+10.24V)
4x00010	0xd800,55296	Current value of AI2 as voltage value with 3 decimal places (voltage value * 1000) between -10240mV and +10240mV. (-10240:-10.24V,0:0V,+10240:+10.24V)
4x00011	0xd800,55296	Current value of AI3 as voltage value with 3 decimal places (voltage value * 1000) between -10240mV and +10240mV. (-10240:-10.24V,0:0V,+10240:+10.24V)
4x00012	0xd800,55296	Current value of AI4 as voltage value with 3 decimal places (voltage value * 1000) between -10240mV and +10240mV. (-10240:-10.24V,0:0V,+10240:+10.24V)
4x00101	0xd800,55296	Current value of AI1 as percentage with 2 decimal places (percentage value * 100) between -10000 and +10000. (-10240:-10.24V,0:0V,+10240:+10.24V)
4x00102	0xd800,55296	Current value of AI2 as percentage with 2 decimal places (percentage value * 100) between -10000 and +10000. (-10240:-10.24V,0:0V,+10240:+10.24V)
4x00103	0xd800,55296	Current value of AI3 as percentage with 2 decimal places (percentage value * 100) between -10000 and +10000. (-10240:-10.24V,0:0V,+10240:+10.24V)
4x00104	0xd800,55296	Current value of AI4 as percentage with 2 decimal places (percentage value * 100) between -10000 and +10000. (-10240:-10.24V,0:0V,+10240:+10.24V)
4x00201	0xd800,55296	Current value of AI1 as voltage value with 3 decimal places (voltage value * 1000) between -10240mV and +10240mV. (-10240:-10.24V,0:0V,+10240:+10.24V)
4x00202	0xd800,55296	Current value of AI2 as voltage value with 3 decimal places (voltage value * 1000) between -10240mV and +10240mV. (-10240:-10.24V,0:0V,+10240:+10.24V)
4x00203	0xd800,55296	Current value of AI3 as voltage value with 3 decimal places (voltage value * 1000) between -10240mV and +10240mV. (-10240:-10.24V,0:0V,+10240:+10.24V)
4x00204	0xd800,55296	Current value of AI4 as voltage value with 3 decimal places (voltage value * 1000) between -10240mV and +10240mV. (-10240:-10.24V,0:0V,+10240:+10.24V)
4x00301-302	0xfff05fe1,-1024031,-102.4031	SINT32:Current value of AI1 as percentage with 4 decimal places (percentage value * 10000) between -1000000 and +1000000. (-1024000:-10.24V,0:0V,+10...
4x00303-304	0xfff05fe1,-1024031,-102.4031	SINT32:Current value of AI2 as percentage with 4 decimal places (percentage value * 10000) between -1000000 and +1000000. (-1024000:-10.24V,0:0V,+10...
4x00305-306	0xfff05fe1,-1024031,-102.4031	SINT32:Current value of AI3 as percentage with 4 decimal places (percentage value * 10000) between -1000000 and +1000000. (-1024000:-10.24V,0:0V,+10...
4x00307-308	0xfff05fe1,-1024031,-102.4031	SINT32:Current value of AI4 as percentage with 4 decimal places (percentage value * 10000) between -1000000 and +1000000. (-1024000:-10.24V,0:0V,+10...
4x00401-402	0xfff05fe1,-1024031,-102.4031	SINT32R:Current value of AI1 as percentage with 4 decimal places (percentage value * 10000) between -1000000 and +1000000. (-1024000:-10.24V,0:0V,+1...
4x00403-404	0xfff05fe1,-1024031,-102.4031	SINT32R:Current value of AI2 as percentage with 4 decimal places (percentage value * 10000) between -1000000 and +1000000. (-1024000:-10.24V,0:0V,+1...
4x00405-406	0xfff05fe1,-1024031,-102.4031	SINT32R:Current value of AI3 as percentage with 4 decimal places (percentage value * 10000) between -1000000 and +1000000. (-1024000:-10.24V,0:0V,+1...
4x00407-408	0xfff05fe1,-1024031,-102.4031	SINT32R:Current value of AI4 as percentage with 4 decimal places (percentage value * 10000) between -1000000 and +1000000. (-1024000:-10.24V,0:0V,+1...
4x00501-502	0xfff05fe1,-1024031,-10.24031	SINT32:Current value of AI1 voltage value with 5 decimal places (voltage value * 100000) between -1024000 and +1024000. (-1024000:-10.24V,0:0V,+102400...
4x00503-504	0xfff05fe1,-1024031,-10.24031	SINT32:Current value of AI2 voltage value with 5 decimal places (voltage value * 100000) between -1024000 and +1024000. (-1024000:-10.24V,0:0V,+102400...
4x00505-506	0xfff05fe1,-1024031,-10.24031	SINT32:Current value of AI3 voltage value with 5 decimal places (voltage value * 100000) between -1024000 and +1024000. (-1024000:-10.24V,0:0V,+102400...
4x00507-508	0xfff05fe1,-1024031,-10.24031	SINT32:Current value of AI4 voltage value with 5 decimal places (voltage value * 100000) between -1024000 and +1024000. (-1024000:-10.24V,0:0V,+102400...
4x00601-602	0xfff05fe1,-1024031,-10.24031	SINT32R:Current value of AI1 voltage value with 5 decimal places (voltage value * 100000) between -1024000 and +1024000. (-1024000:-10.24V,0:0V,+1024

37 RESI-4AOU-SIO, RESI-4AOU-ETH

37.1 General information

This series of IO modules offer the following features:

- 4 high precision analog outputs for -10Vdc..+10Vdc signals
- ADC resolution 12 bit, accuracy +/-0.1%
- RESI-xxx-SIO: Galvanic isolated RS232 and RS485 interface for communication with a host system
- RESI-xxx-ETH: Galvanic isolated Ethernet interface for communication with a host system

Figure: Our serial IO module

Figure: Our Ethernet IO module

37.2 Technical specification

Beside the basic technical data, which fulfil all of our IO modules, this IO modules meet the following technical specifications:

Power consumption	
RESI-4AOU-SIO	<1.1W
RESI-4AOU-ETH	<1.5W
Product housing	
RESI-4AOU-SIO	CEM17
RESI-4AOU-ETH	CEM35
Product weight	
RESI-4AOU-SIO	62g
RESI-4AOU-ETH	96g
Analog outputs	
Number	4
Update speed	Every 100ms
Range	-10V..+10V
ADC resolution	12 bit
Output voltage range	-10V..+10V
Accuracy	+/-0.1%
Cable connection	via terminals
Galvanic isolation	Yes
Default serial settings	
Baud rate	via DIP switch
Parity	none
Stopbits	one
UnitID	255
Default Ethernet settings	
IP address	192.168.0.52
IP mask	255.255.255.0
gateway	192.168.0.1
UnitID	255
User	RESI
password	RESI

37.3 Additional terminals & LED states

ANALOG OUTPUTS	4 analog outputs for -10V..0V..+10V signals	
	Two 3 pin terminal blocks	
	Terminal type:	USLIM
	C:	Ground for all analog inputs
	AO1-AO4:	Analog outputs
Pin layout	AO1:	Signal output for analog output #1
	AO2:	Signal output for analog output #2
	AO3:	Signal output for analog output #3
	AO4:	Signal output for analog output #4
	C:	Signal ground for analog outputs #1-#4
		Both signal grounds are internally bridged
INFO	If everything is OK, this LED is on. If there is an internal error at the analog outputs,	
	this LED flashes quickly.	

37.4 RESI-4AOU-SIO,ETH: Schematic diagram

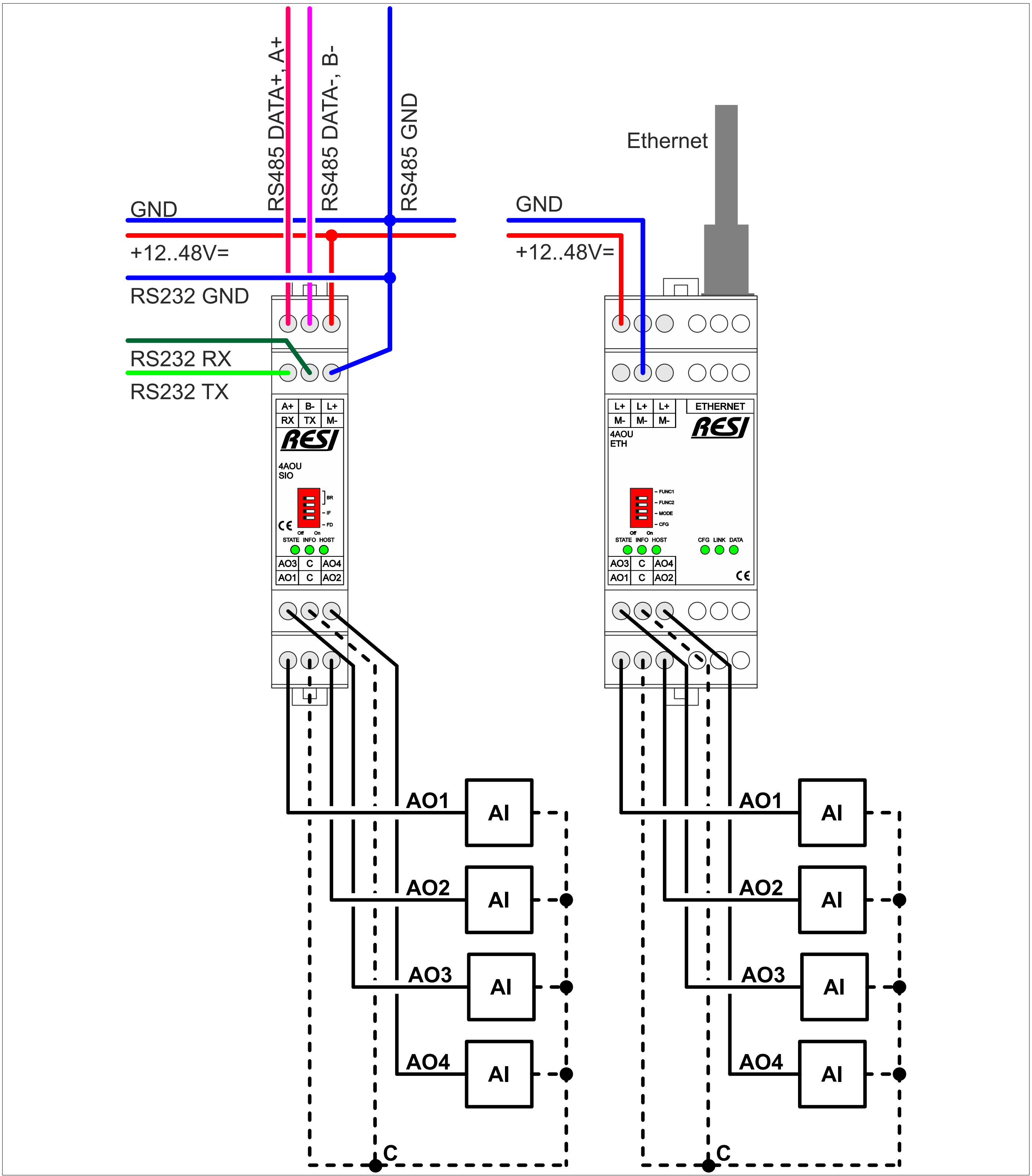


Figure: Schematics for the IO modules

37.5 RESI-4AOU-SIO,ETH: Wiring diagram

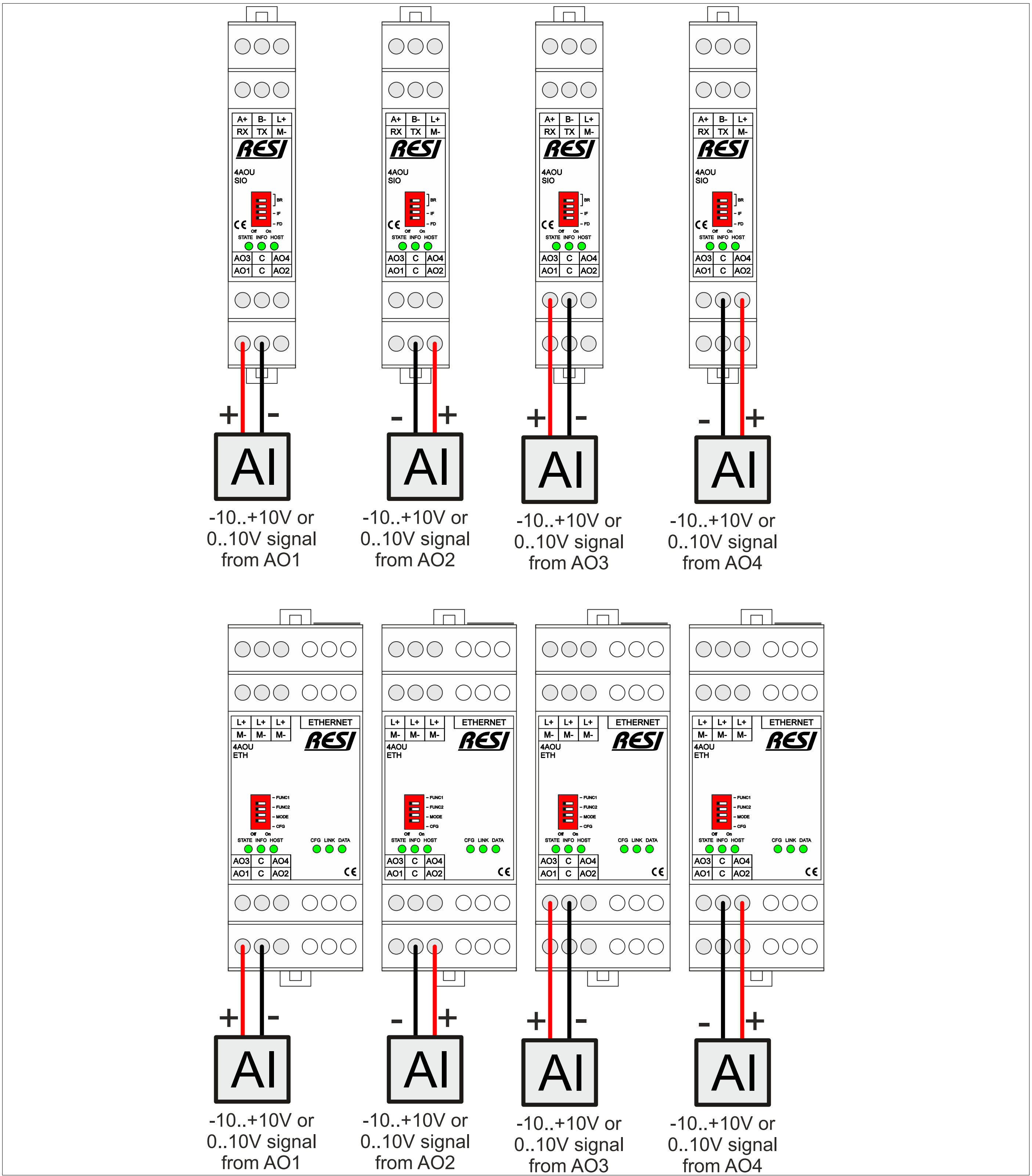


Figure: Wiring diagram for the IO modules

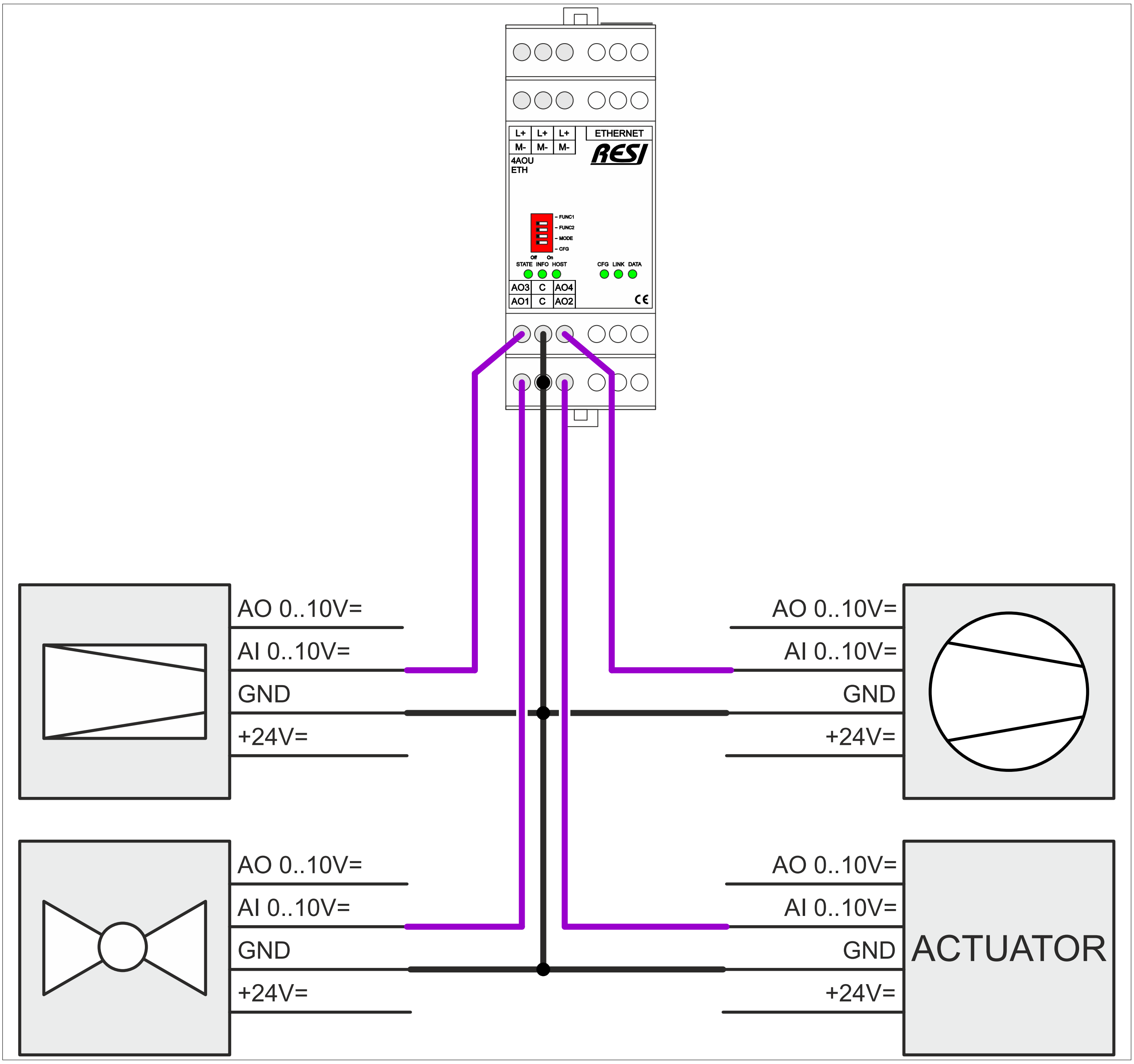


Figure: Wiring diagram for building automation devices

37.6 Additional MODBUS register & coils

Here you will find only the additional MODBUS registers and coils especially for this IO module. Please refer to the description of of the standard MODBUS mapping for more details about the available basic MODBUS registers and coils.

Please refer to the external document for detailed documentation of the current MODBUS register mapping for this IO module:

RESI-L-4AOU-SIO-ETH-MODBUS+ASCII-ENxx.pdf

37.7 Additional ASCII commands

Here you will find only the additional ASCII commands especially for this IO module. Please refer to the description of of the standard commands for more details about the available basic ASCII commands.

Please refer to the external document for detailed documentation of the current ASCII commands for this IO module:

RESI-L-4AOU-SIO-ETH-MODBUS+ASCII-ENxx.pdf

38 RESI-2AIU2AOU-SIO, RESI-2AIU2AOU-ETH

38.1 General information

This series of IO modules offer the following features:

- 2 analog inputs and 2 analog outputs for 0..+10Vdc signals
- ADC resolution 12 bit, accuracy +/-0.5%
- RESI-xxx-SIO: Galvanic isolated RS232 and RS485 interface for communication with a host system
- RESI-xxx-ETH: Galvanic isolated Ethernet interface for communication with a host system

Figure: Our serial IO module

Figure: Our Ethernet IO module

38.2 Technical specification

Beside the basic technical data, which fulfil all of our IO modules, this IO modules meet the following technical specifications:

Power consumption	
RESI-2AIU2AOU-SIO	<1.0W
RESI-2AIU2AOU-ETH	<1.4W
Product housing	
RESI-2AIU2AOU-SIO	CEM17
RESI-2AIU2AOU-ETH	CEM35
Product weight	
RESI-2AIU2AOU-SIO	62g
RESI-2AIU2AOU-ETH	96g
Analog inputs	
Number	2
Update speed	Every 100ms
Range	0V..+10V
ADC resolution	12 bit
Input voltage range	0V..+10V
Accuracy	+/-0.5%
Cable connection	via terminals
Galvanic isolation	Yes, to rest of module, but not to other analog inputs and outputs
Analog outputs	
Number	2
Update speed	Every 100ms
Range	0V..+10V
ADC resolution	12 bit
Output voltage range	0V..+10V
Accuracy	+/-0.5%
Cable connection	via terminals
Galvanic isolation	Yes, to rest of module, but not to other analog inputs and outputs
Default serial settings	
Baud rate	via DIP switch
Parity	none
Stopbits	one
UnitID	255
Default Ethernet settings	
IP address	192.168.0.53
IP mask	255.255.255.0
gateway	192.168.0.1
UnitID	255
User	RESI
password	RESI

38.3 Additional terminals & LED states

ANALOG INPUTS	2 analog inputs for 0V..+10V signals	
	One 3 pin terminal block	
	Terminal type:	USLIM
	C:	Ground for all analog inputs and outputs
	AI1-AI2:	Analog inputs
ANALOG OUTPUTS	2 analog outputs for 0V..+10V signals	
	One 3 pin terminal block	
	Terminal type:	USLIM
	C:	Ground for all analog inputs and outputs
	AO1-AO2:	Analog outputs
Pin layout	AO1:	Signal output for analog output #1
	C:	Ground for all analog inputs and outputs
	AO2:	Signal output for analog output #2
	AI1:	Signal input for analog input #1
	C:	Ground for all analog inputs and outputs
	AI2:	Signal input for analog input #2
		Both signal grounds are internally bridged
INFO	If everything is OK, this LED is on. If there is an internal error on the	
	analog inputs or outputs, this LED flashes quickly.	

38.4 RESI-2AIU2AOU-SIO,ETH: Schematic diagram

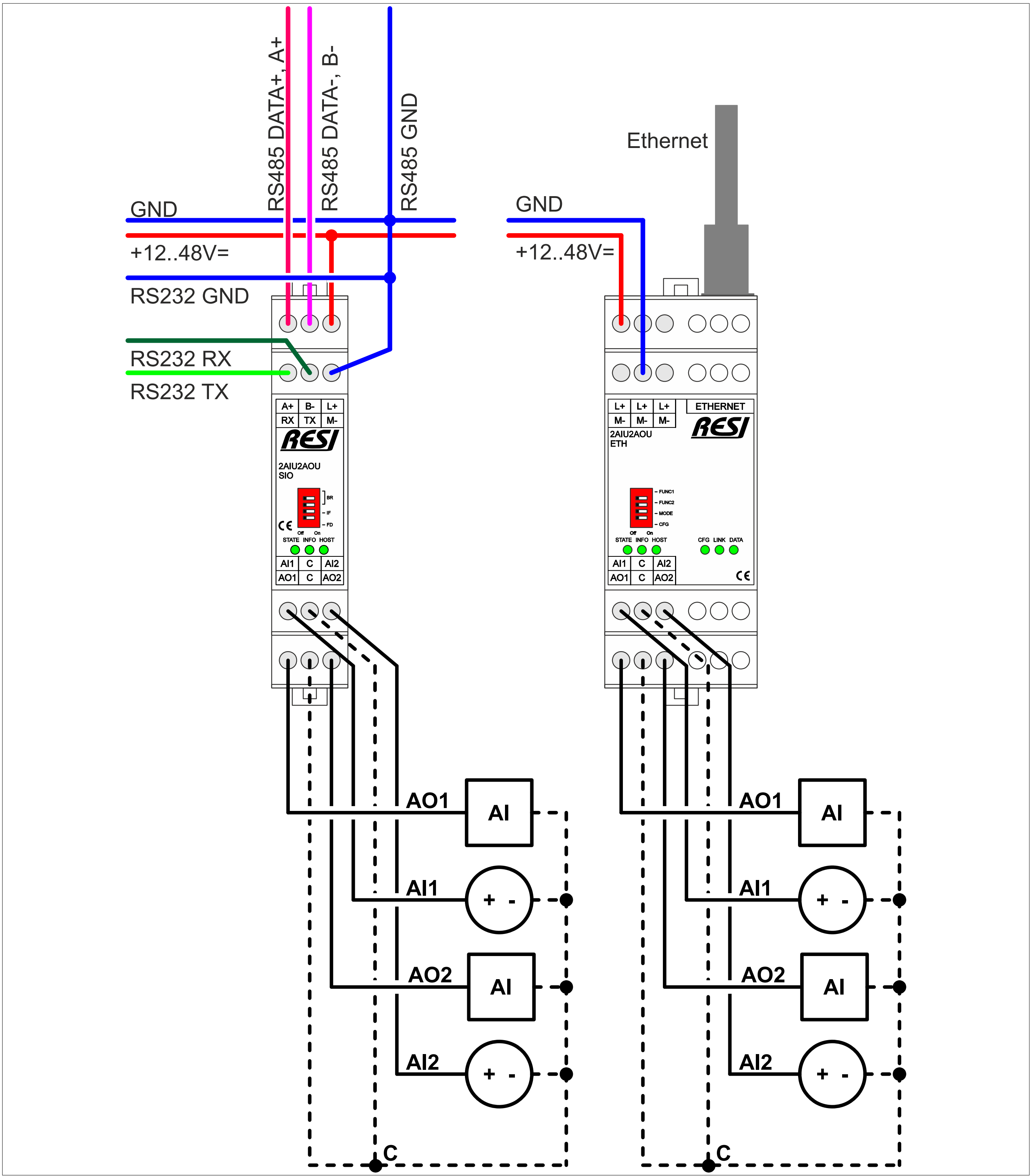


Figure: Schematics for the IO modules

38.5 RESI-2AIU2AOU-SIO,ETH: Wiring diagram

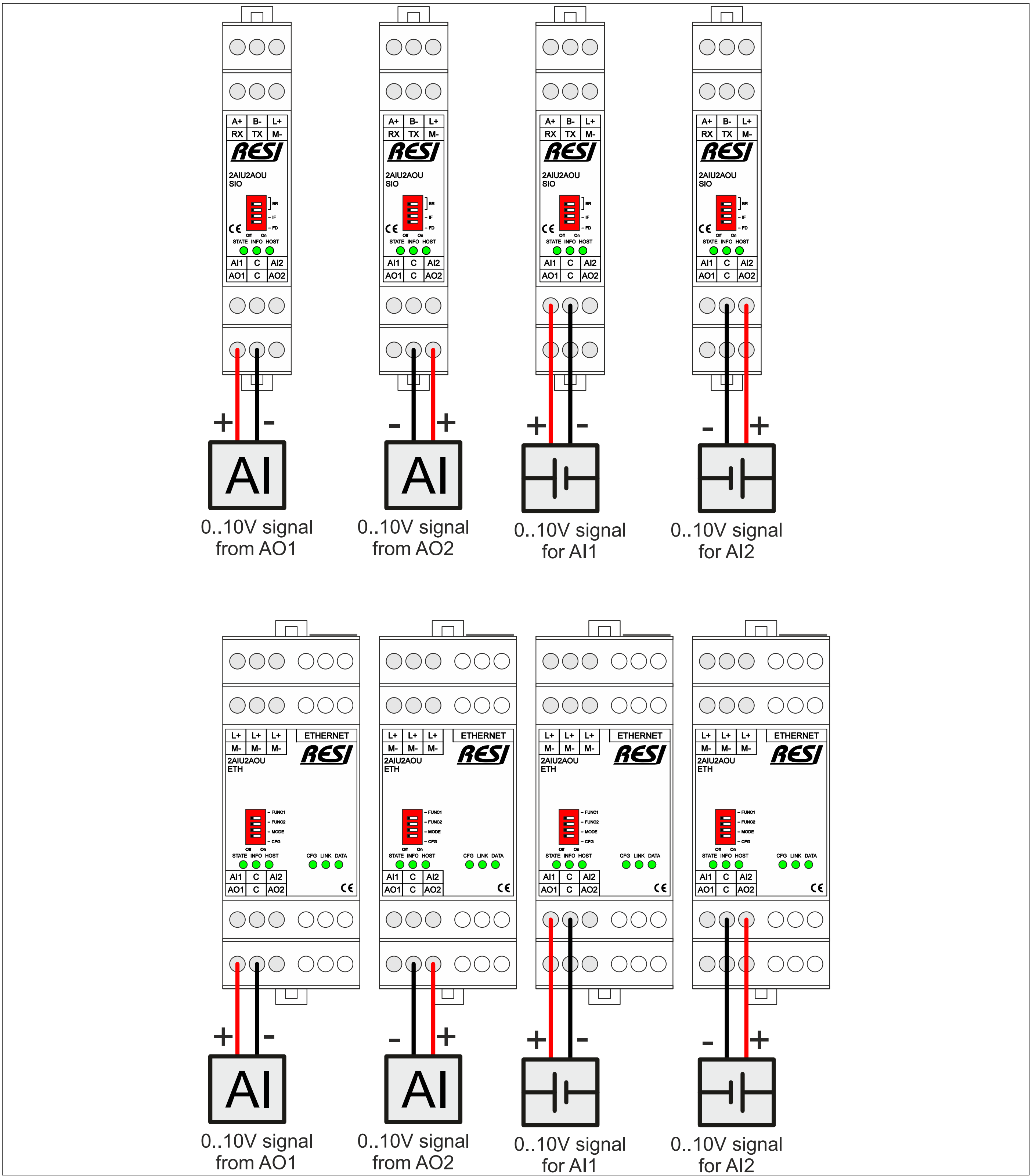


Figure: Wiring diagram for the IO modules

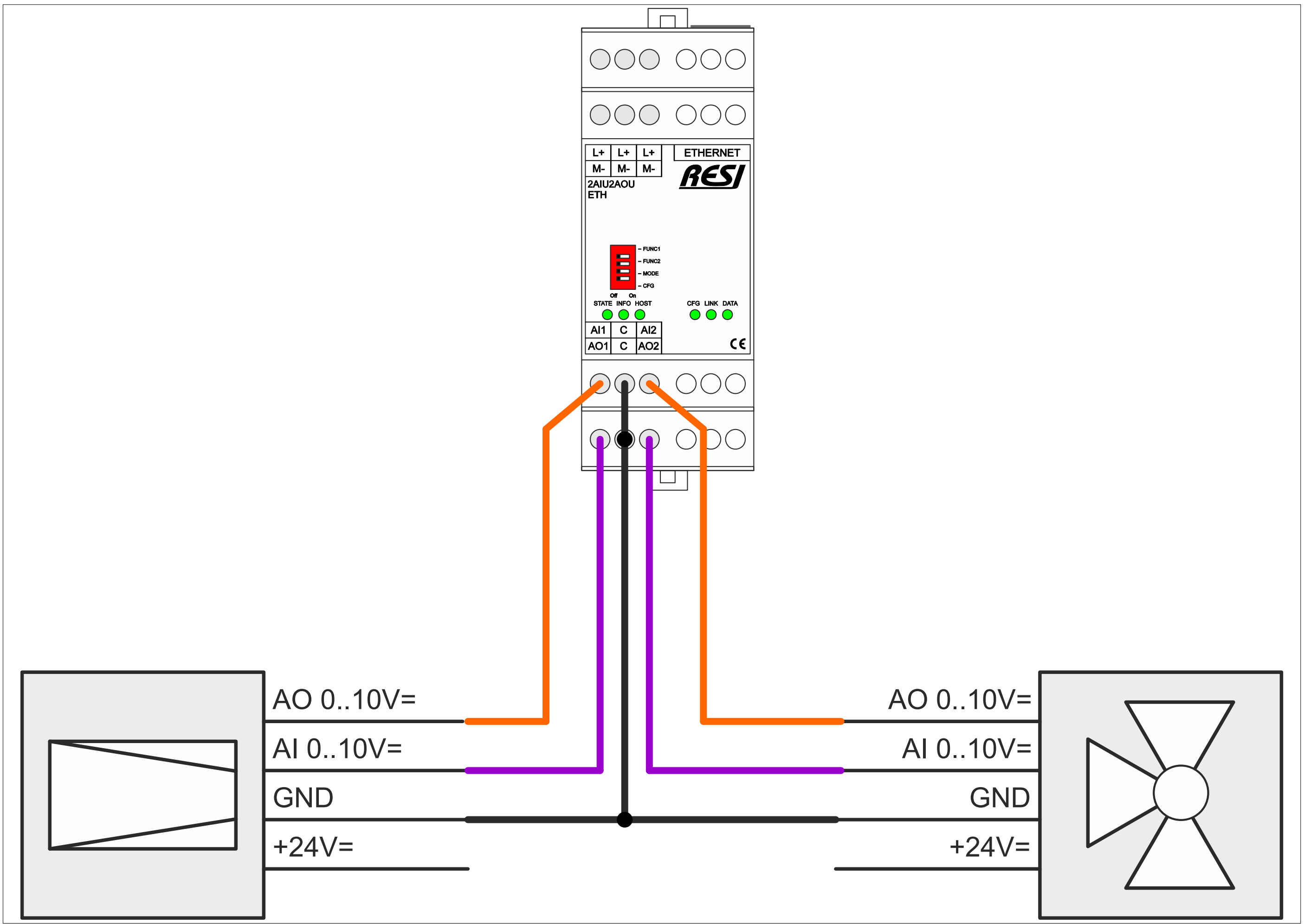


Figure: Wiring diagram for building automation devices

38.6 Additional MODBUS register & coils

Here you will find only the additional MODBUS registers and coils especially for this IO module. Please refer to the description of of the standard MODBUS mapping for more details about the available basic MODBUS registers and coils.

Please refer to the external document for detailed documentation of the current MODBUS register mapping for this IO module:

RESI-L-2AIU2AOU-SIO-ETH-MODBUS+ASCII-ENxx.pdf

38.7 Additional ASCII commands

Here you will find only the additional ASCII commands especially for this IO module. Please refer to the description of of the standard commands for more details about the available basic ASCII commands.

Please refer to the external document for detailed documentation of the current ASCII commands for this IO module:

RESI-L-2AIU2AOU-SIO-ETH-MODBUS+ASCII-ENxx.pdf

39 RESI-1LED-SIO, RESI-1LED-ETH

39.1 General information

This series of IO modules offer the following features:

- 3 dimmable PWM output channels for LED stripes, 0..48Vdc, max. 5A each channel
- Six selectable modes: OFF, ON, FLASHING, FADING, RANDOM, SEQUENCE
- External power supply for LED stripes, 0..48Vdc, max. 15A
- RESI-xxx-SIO: Galvanic isolated RS232 and RS485 interface for communication with a host system
- RESI-xxx-ETH: Galvanic isolated Ethernet interface for communication with a host system

Figure: Our serial IO module

Figure: Our Ethernet IO module

39.2 Technical specification

Beside the basic technical data, which fulfil all of our IO modules, this IO modules meet the following technical specifications:

Power consumption	
RESI-1LED-SIO	<0.8W
RESI-1LED-ETH	<1.2W
Product housing	
RESI-2AIU2AOU-SIO	CEM17
RESI-2AIU2AOU-ETH	CEM35
Product weight	
RESI-2AIU2AOU-SIO	60g
RESI-2AIU2AOU-ETH	94g
LED stripe outputs	
Number	3
Type of outputs	PWM with 400Hz
LED stripes	RGB, Dual white, Mono color
LED connection	Via common anode
LED Output voltage	0..48Vdc
LED Output current	Max. 5A per channel
LED Power supply	0..48Vdc,max 15A
	180W@12V=, 360W@24V=, 720W@48V=
Cable connection	via terminals
Galvanic isolation	Yes, to rest of module, but not to other PWM outputs
Default serial settings	
Baud rate	via DIP switch
Parity	none
Stopbits	one
UnitID	255
Default Ethernet settings	
IP address	192.168.0.60
IP mask	255.255.255.0
gateway	192.168.0.1
UnitID	255
User	RESI
password	RESI

39.3 Additional terminals & LED states

LED OUTPUTS	3 LED outputs for PWM signals	
	Two 3 pin terminal blocks	
	Terminal type:	USLIM
	IN+:	LED Power supply <48V, <15A
	IN-:	LED Power supply ground signal
	O+:	Common LED anode for all three PWM outputs
	O1,O2,O3:	PWM outputs to LED cathode
Pin layout	IN+::	LED Power supply <48V, <15A
	O+:	Common LED anode for all three PWM outputs
	IN-:	LED Power supply ground signal
	O1:	PWM output #1 to LED cathode group #1
	O2:	PWM output #1 to LED cathode group #2
	O3:	PWM output #1 to LED cathode group #3
INFO	This LED shows the status of the three outputs. In mode OFF this LED is off.	
	In mode ON the LED is on. The LED flashes if fading is active.	

39.4 RESI-1LED-SIO,ETH: Schematic diagram

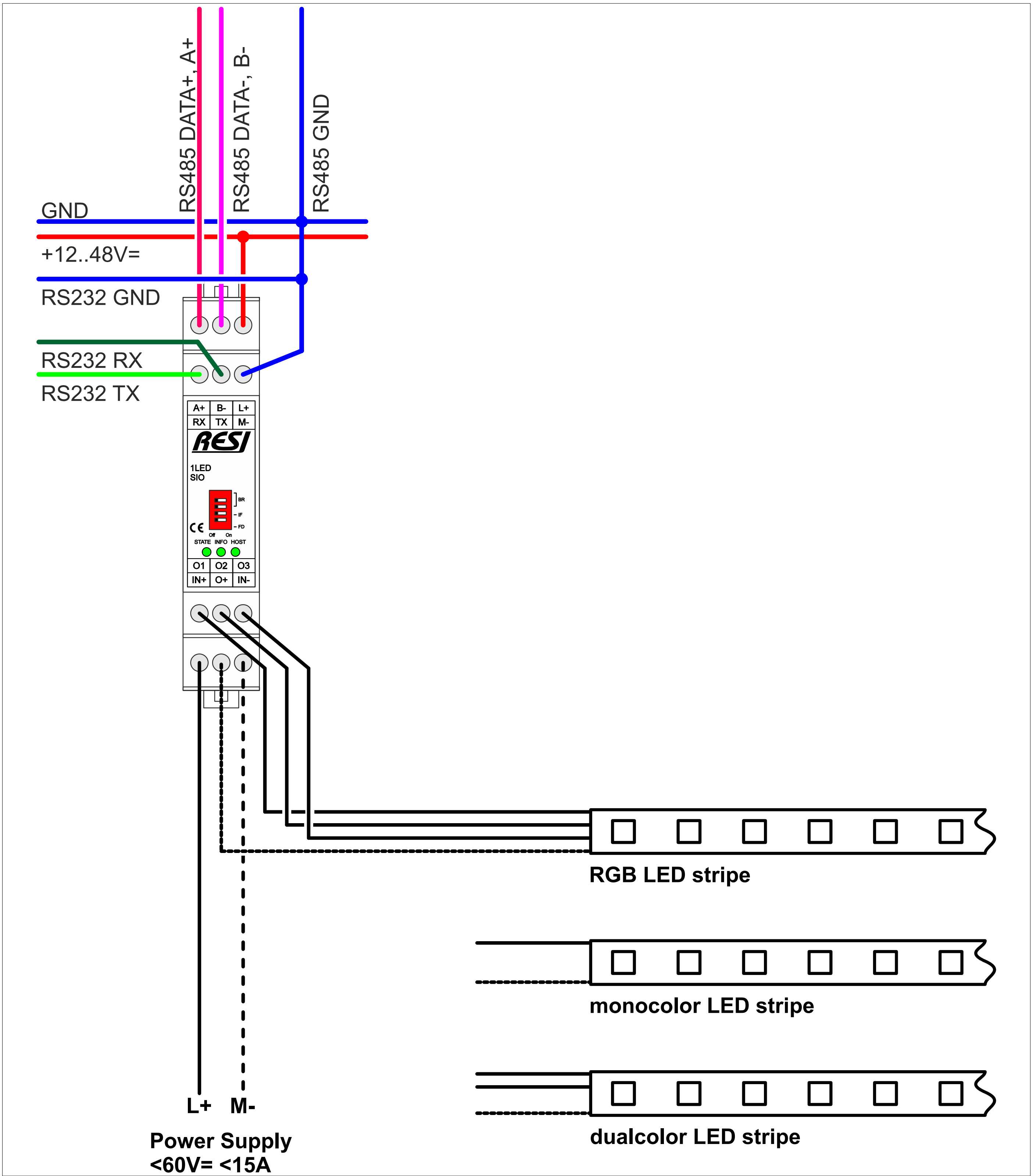


Figure: Schematics for the IO modules

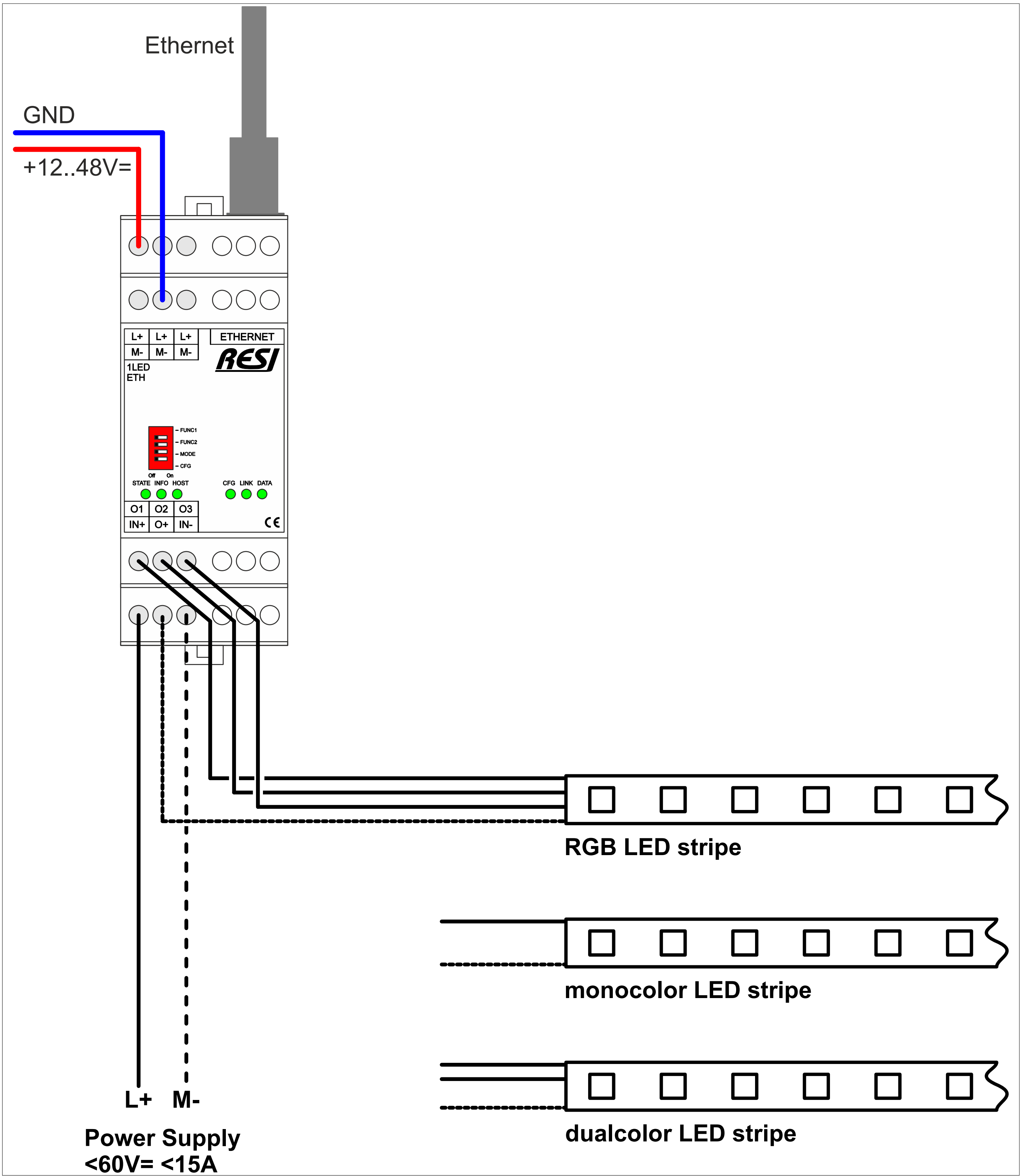


Figure: Schematics for the IO modules

39.5 The modes of the LED module

The LED module offers six modes. You can switch the mode by setting a special register via MODBUS/RTU or by executing the #SMODE ASCII command. Be aware that the converter does not save a mode in remanent memory. After reset the module starts always in mode ON!

39.5.1 LED mode OFF

In this mode all three outputs are switched to 0. It doesn't matter, what values are in the set point registers LO1 4x00001, LO2 4x00002 or LO3 4x00003. The registers for the actual output values CLO1 4x00008, CLO2 4x00009 and CLO3 4x00010 return always the value 0.

39.5.2 LED mode ON

In this mode all three outputs are switched immediately to the current values in the registers LO1 4x00001, LO2 4x00002 or LO3 4x00003. The registers for the actual output values CLO1 4x00008, CLO2 4x00009 and CLO3 4x00010 delivers always the same value as the registers LO1 4x00001, LO2 4x00002 or LO3 4x00003 to indicate, that the values are really outputted to the three PWM channels.

39.5.3 LED mode FLASH

In this mode all three outputs are switched as a recycler relay between the three current values in the registers LO1 4x00001, LO2 4x00002 or LO3 4x00003 and 0. While ON time, the module outputs the three registers LO1 4x00001, LO2 4x00002 or LO3 4x00003 to the real outputs for a timespan defined in the register MINIMUM TIME 4x00006 in 1/10s. In this time the registers for the actual output values CLO1 4x00008, CLO2 4x00009 and CLO3 4x00010 delivers always the same value as the registers LO1 4x00001, LO2 4x00002 or LO3 4x00003 to indicate, that the values are really outputted to the three PWM channels. Then the converter switches all three channels to 0 for the OFF time span. This time span is defined with the value of the MAXIMUM TIME register 4x00007 in 1/10s. In this time the registers for the actual output values CLO1 4x00008, CLO2 4x00009 and CLO3 4x00010 delivers always the value 0. This ON/OFF cycle is repeated endlessly.

Steps for FLASH:

Step 1: Output of the three set point values LO1, LO2, and LO3 to the real PWM outputs
Step 2: Wait for MINIMUM TIME in 1/10s
Step 3: output of the values 0, 0, 0 to the real PWM outputs
Step 4: Wait for MAXIMUM TIME in 1/10s
Step 5: continue with step 1

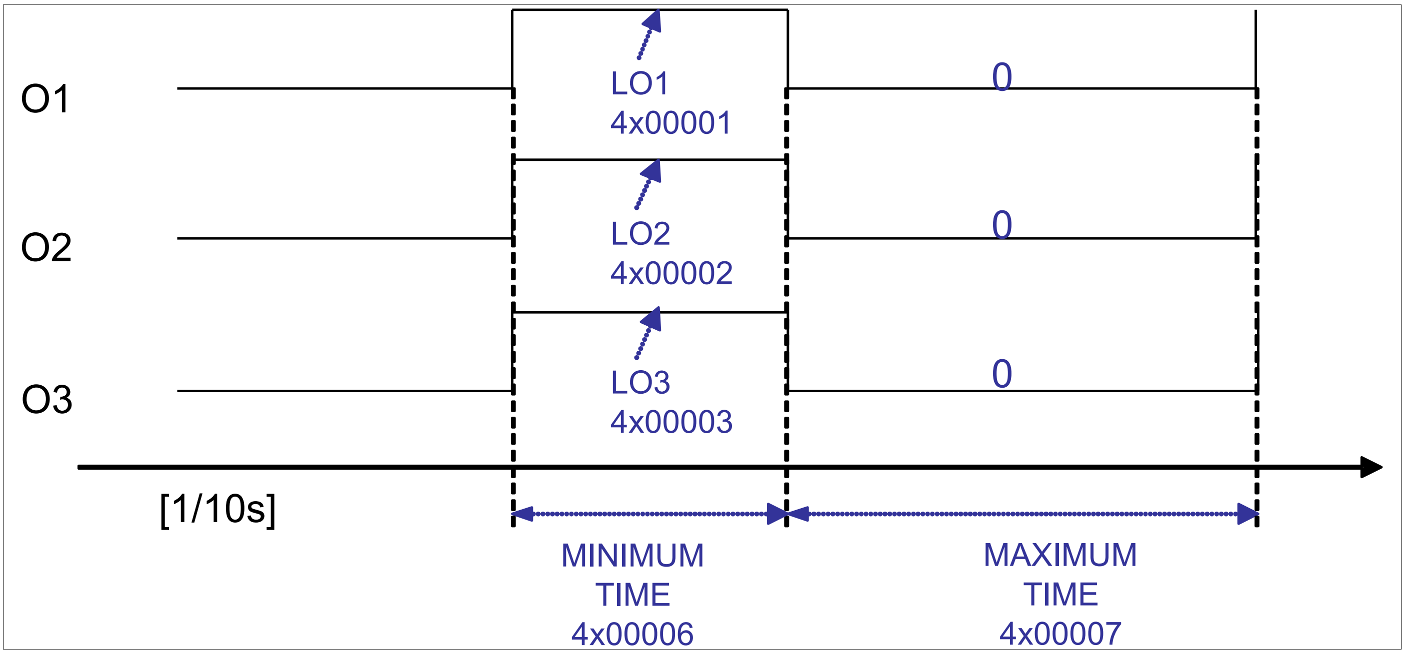


Figure: Timing of mode FLASH

39.5.4 LED mode FADE

In this mode the converter doesn't change the output values immediately. No, it uses a ramp to change slowly from the current value to the new value. This ramp is defined in the register FADE SPEED 4x00005. The setup is done in steps per 1/100s and is valid for all three channels. To set a new value write into the three registers LO1 4x00001, LO2 4x00002 or LO3 4x00003. The system fades from the current value to the new values. If you read the registers CLO1 4x00008, CLO2 4x00009 and CLO3 4x00010 while fading, you will get every value change from the old value to the new value. Also the register IS FADE ACTIVE 4x00014 will return a 1 while fading is running. When the module reaches the new values, reading of the registers CLO1 4x00008, CLO2 4x00009 and CLO3 4x00010 will return the same values as the registers LO1 4x00001, LO2 4x00002 and LO3 4x00003. Also the register value of IS FADE ACTIVE 4x00014 will be 0.

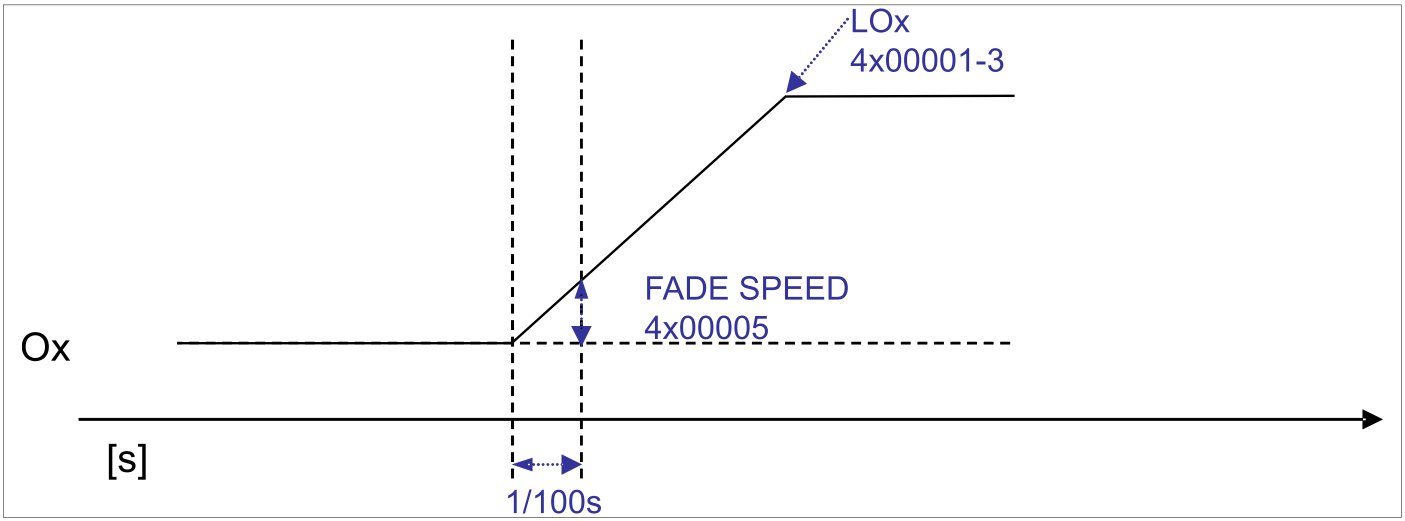


Figure: Timing of mode FADE

39.5.5 LED mode RANDOM

In this mode the converter generates random values for each output. For this you can setup a time interval. If this time interval expires the system dices new random values for the three outputs. The time interval is defined by the register MINIMUM TIME 4x00006 and the register MAXIMUM TIME 4x00007 in seconds. The system generates a random time interval between those two parameters. If the time expires, the system dices new random values for the three registers RLO1 4x00011, RLO2 4x00012 and RLO3 4x00013. Then the system fades the current values in the registers CLO1 4x00008, CLO2 4x00009 and CLO3 4x00010 to the new random values. This fade ramp is defined in the register FADE SPEED 4x00005. The setup is done in steps per 1/100s. If you read the registers CLO1 4x00008, CLO2 4x00009 and CLO3 4x00010 while fading, you will get every value change from the old value to the new value. Also the register IS FADE ACTIVE 4x00014 will return a 1 while fading is running. When the module reaches the new values, reading of the registers CLO1 4x00008, CLO2 4x00009 and CLO3 4x00010 will return the same values as the registers RLO1 4x00001, RLO2 4x00002 and RLO3 4x00003. Also the register value of IS FADE ACTIVE 4x00014 will be 0. The diced values in the registers RLO1 4x00011, RLO2 4x00012 and RLO3 4x00013 will be in the range of 0 to LO1 4x00001, LO2 4x00002 and LO3 4x00003.

Steps for RANDOM:

Step 1: Dice three random numbers in the range of 0..LOx and store the values in RLOx
Step 2: Dice a random wait period between MINIMUM TIME and MAXIMUM TIME in seconds
Step 3: Fade up or down from the actual output values CLOx to the new end values RLOx
Step 4: If the random wait period is over, continue with step 1

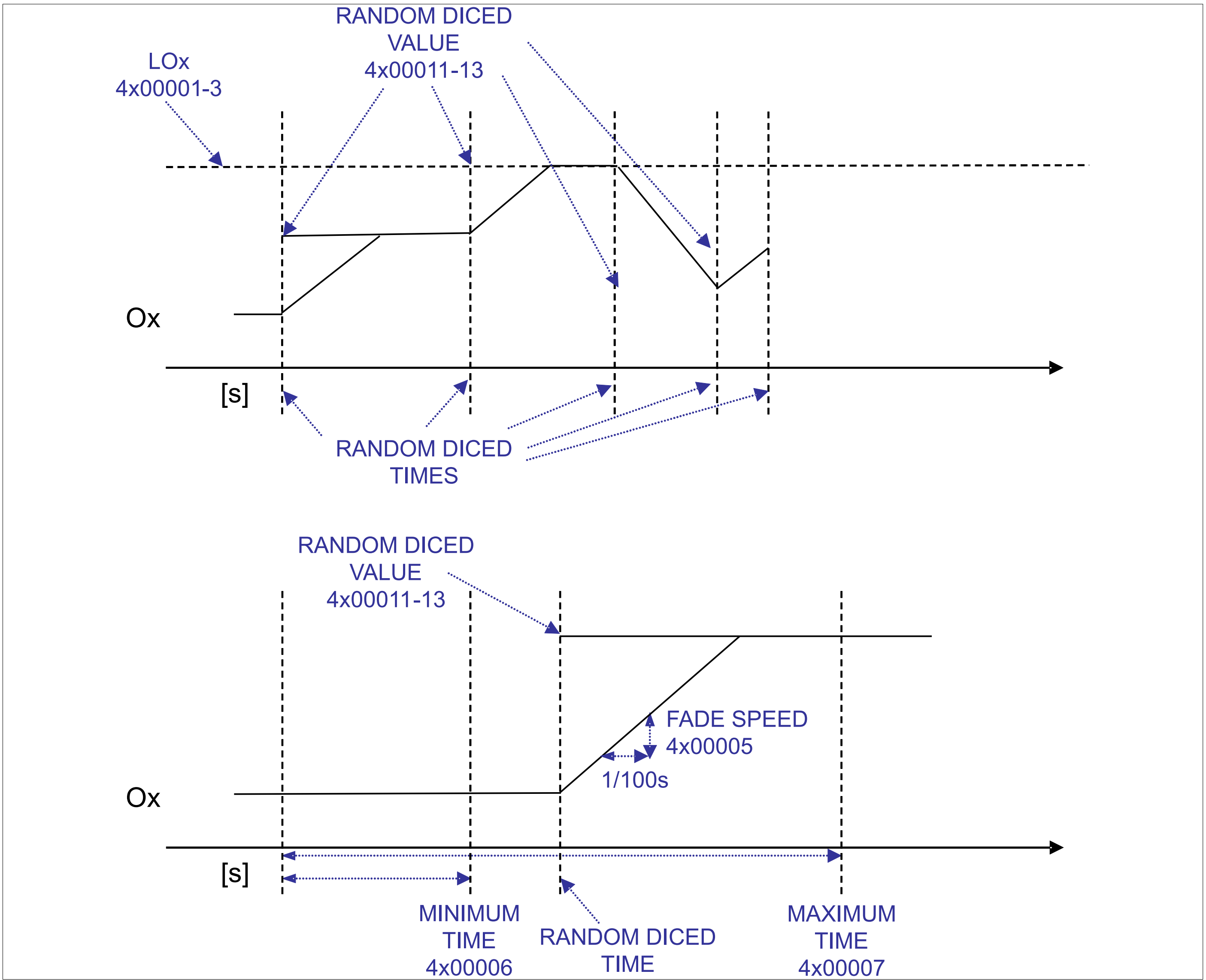


Figure: Timing of mode RANDOM

39.5.6 LED mode SEQUENCE

In this mode, the module creates a sequential flash light with the three PWM outputs. The outputs flashes between the three set points LO1, LO2 and LO3 and 0 in sequence. In the first ON phase the module sets the real output CLO1 to the set point LO1, the other two outputs are set to 0. This phase lasts for MINIMUM TIME in 1/10s. While this period of time, the current value register CLO1 delivers the same value as in LO1, and the other two current value registers CLO2 and CLO3 deliver the value 0. Then the module switches all three outputs to 0 for a time period defined with the register MAXIMUM TIME in 1/10s (OFF time period). While this period of time, all three output registers CLOx deliver the value 0. Now the system repeats the ON phase with the next set point register LO2. The two registers CLO1 and CLO3 are 0 in this phase. Next the OFF time period is executed. The last phase is the ON phase with the register LO3. The two registers CLO1 and CLO2 are 0 in this phase. The last OFF time period is executed. This three times ON/OFF cycle is repeated endlessly.

Steps for SEQUENCE:

Step 1: Output the three set points LO1, 0, 0 to the three PWM outputs
Step 2: wait for MINIMUM TIME in 1/10s
Step 3: Output the values 0, 0, 0 to the three PWM outputs
Step 4: wait for MAXIMUM TIME in 1/10s
Step 5: Output the three set points 0, LO2, 0 to the three PWM outputs
Step 6: wait for MINIMUM TIME in 1/10s
Step 7: Output the values 0, 0, 0 to the three PWM outputs
Step 8: wait for MAXIMUM TIME in 1/10s
Step 9: Output the three set points 0, 0, LO3 to the three PWM outputs
Step 10: wait for MINIMUM TIME in 1/10s
Step 11: Output the values 0, 0, 0 to the three PWM outputs
Step 12: wait for MAXIMUM TIME in 1/10s
Step 13: continue with step 1

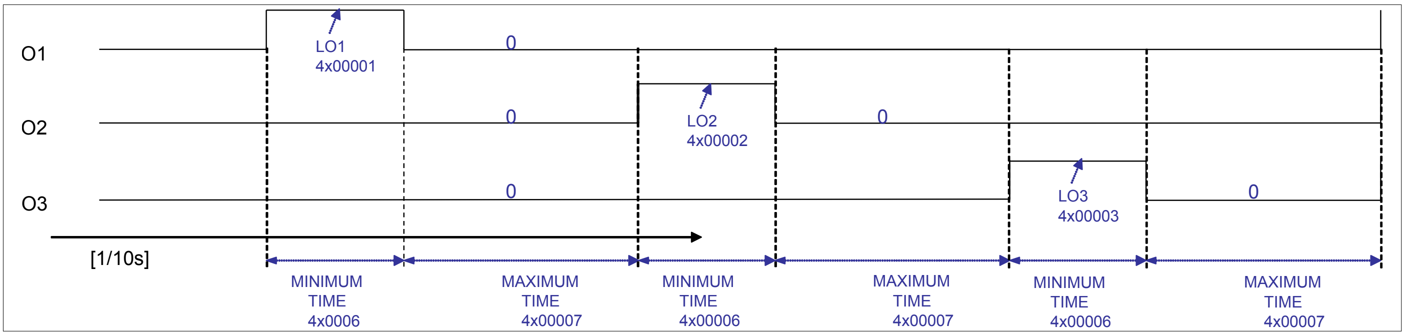


Figure: Timing of mode SEQUENCE

39.6 RESI-1LED-SIO,ETH: Wiring diagram

39.6.1 Cabling of the power supply for the LED stripes

The power supply for the LED stripes must be cabled externally. The module offers the two clamps IN+ and IN- to connect the power supply. Depending on the type of LED stripe you want to use, you can connect various types of power supplies. It is very important, that the used power supply does not exceed the maximum continuous current rating of 15A! The result is the following mandatory limits for supplying LED stripes with different voltage levels:

- LED stripes with 12Vdc power supply: 12Vdc*15A -> max. 180W mains adapter
- LED stripes with 24Vdc power supply: 24Vdc*15A -> max. 360W mains adapter
- LED stripes with 48Vdc power supply: 48Vdc*15A -> max. 720W mains adapter

But be aware, that every output can only drive a maximum current of 5A!

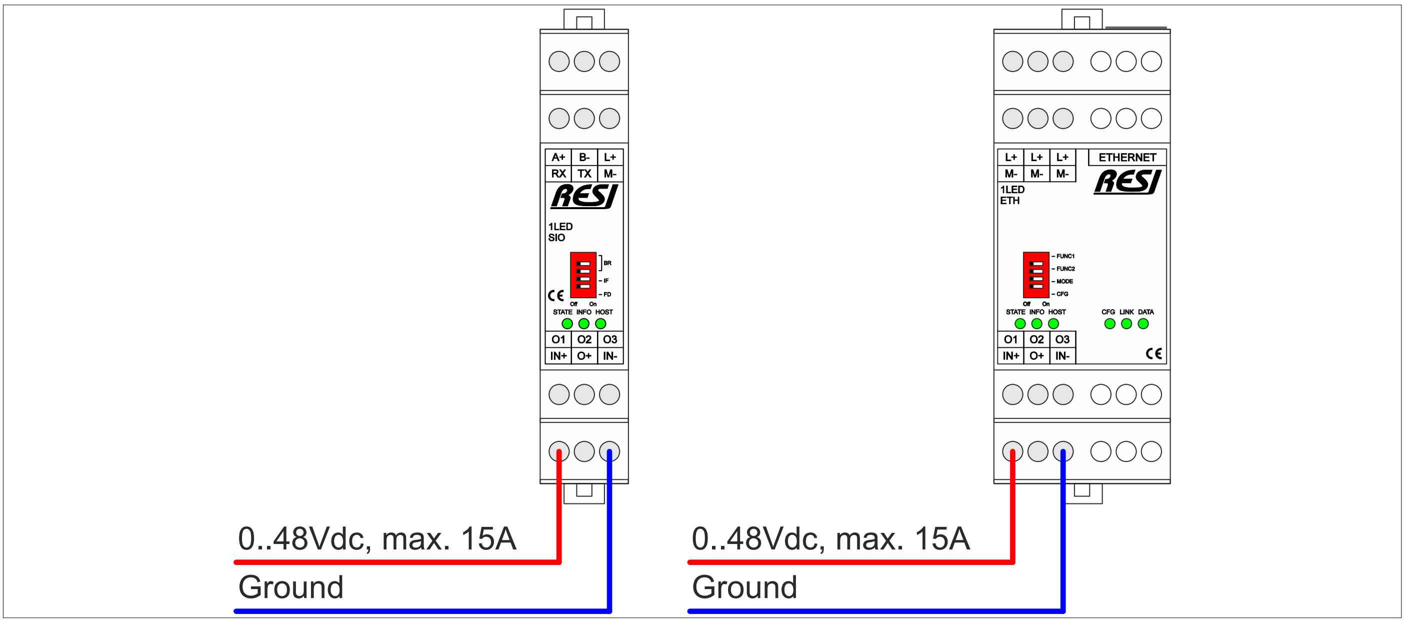


Figure: Power supply for LED stripes

39.6.2 HOWTO connect a 12V mono color LED stripe

Cabling of a 12Vdc LED stripe with 24W power consumption, luminous color 2700K. Due to the reason, that the LED strip consumes only 24W, we use also a 24W mains adapter. So there flows an input current of 2A. Via the output O1 flows also an output current of 2A. (<5A, so this is ok).

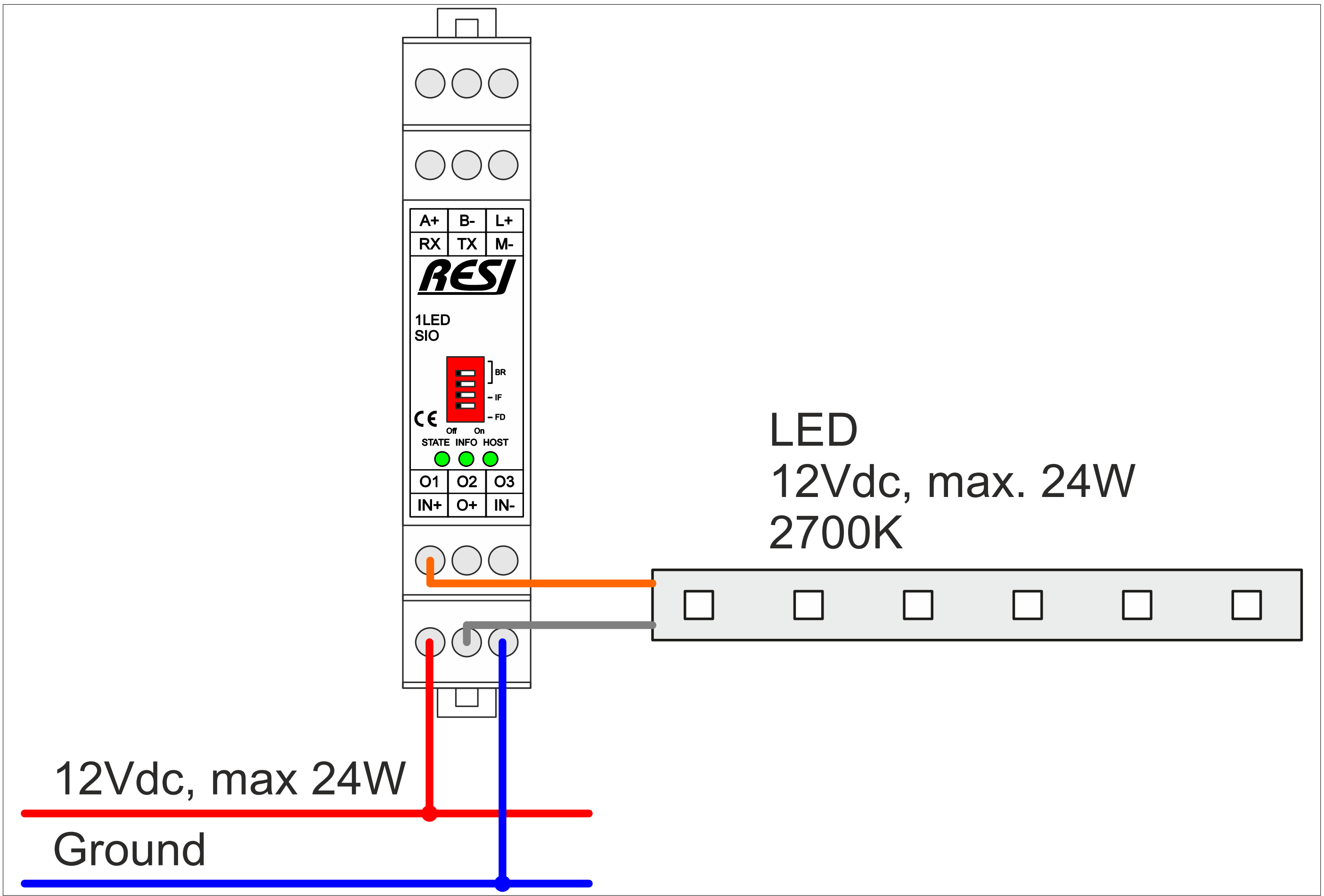


Figure: Cabling of a 12Vdc LED stripe with 24W power consumption

39.6.3 HOWTO connect three 24V mono color LED stripes

Cabling of three 24Vdc LED stripes with 48W each stripe. Each of the three LED stripes can be dimmed individually. The sample uses all three LED outputs to create three individual dimmable groups of LED stripes. Each LED stripe consumes 48W power. So we use a power supply with 3x48W -> 150W. The input current on the clamps IN+, IN- is max. 6.25A. This is less than the allowed 15A and ok. While we use on each output a LED stripe with 48W power, the output current per channel is max. 2A. This is lower than the maximum rating of 5A per output und therefore ok too.

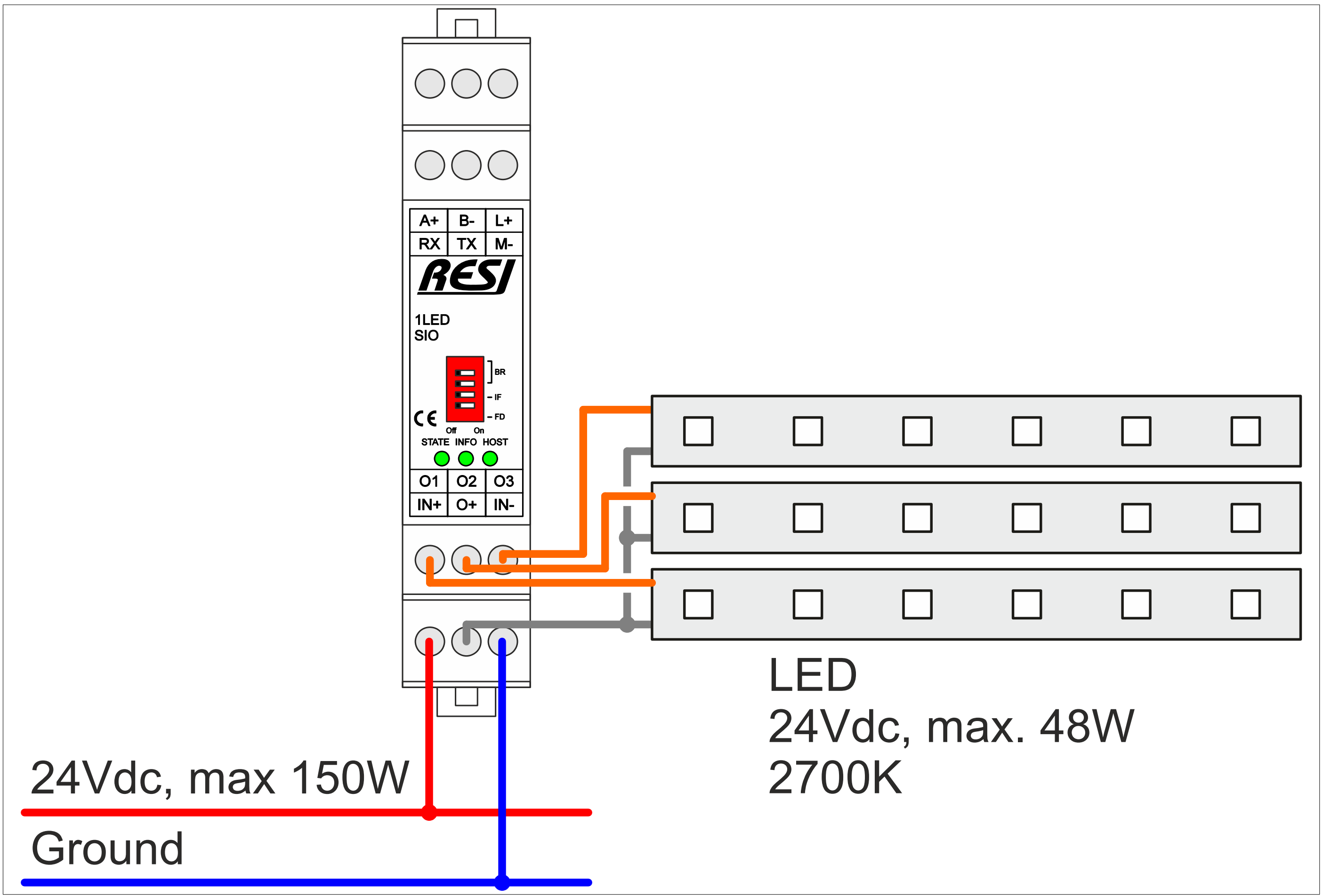


Figure: Cabling of three 24Vdc LED stripes with 48W each stripe

39.6.4 HOWTO connect two 24V mono color LED stripes

Cabling of two 24Vdc LED stripes with 48W power consumption each. Both LED stripes are only together dimmable. Only output O1 is used for both LED stripes. We use a 100W power supply. The primary input current is 4.17A. This is smaller than the allowed 15A and therefore ok. We operate with two LED stripes on one output. This output must drive 96W power. We choose a 24Vdc LED stripe. So the output current is 4A. This is again smaller than the allowed 5A and ok.

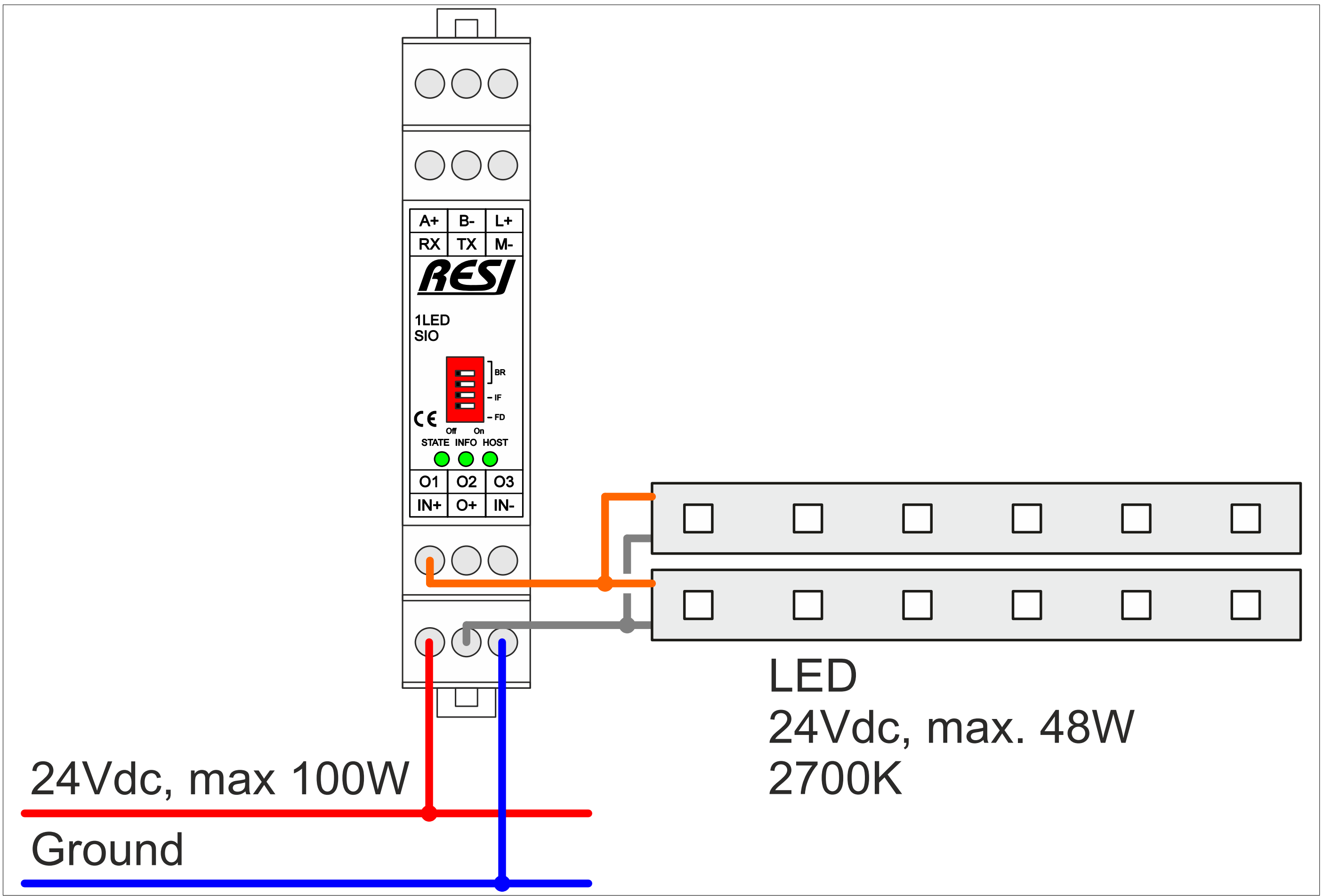


Figure: Cabling of two 24Vdc LED stripes with 48W power consumption each

39.6.5 HOWTO connect RGB LED stripes

In this sample we use RGB LED HD stripe. This stripe offers three dimmable channels for the three primary colors red, green and blue. The common anode is again connected to O+. The 80W power supply delivers a maximum current of 3,34A. So this current is far beyond the allowed 15A for the power input. The LED stripe consumes only 1/3rd of the total power of 72W on each channel. This is 24W, the current is 1A. Again the output current on all three outputs is much lower than the allowed 5A.

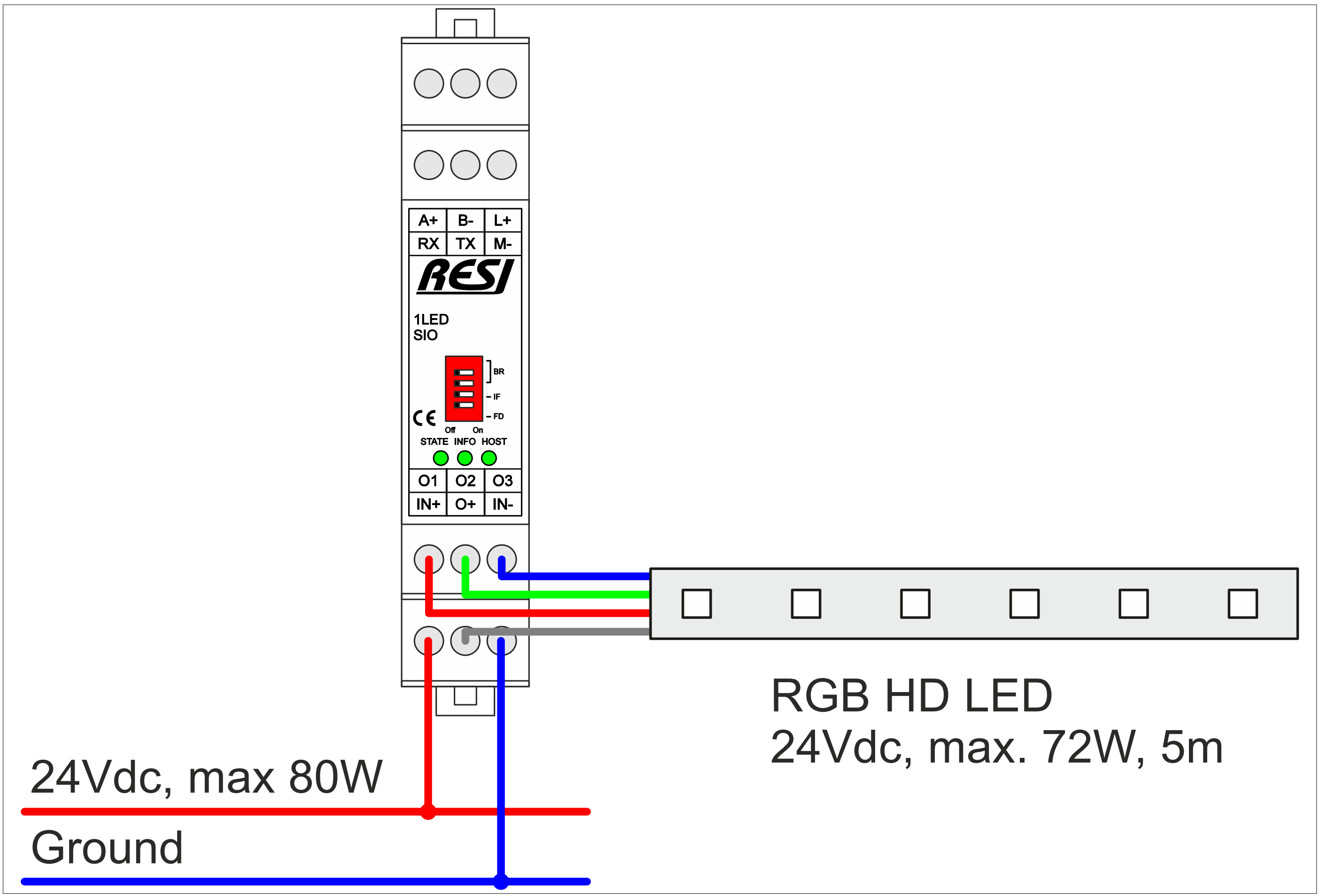


Figure: Cabling of RGB HD LED stripe

39.6.6 HOWTO connect dynamic white LED stripes

Cabling of a dynamic white LED stripe. This type of LED stripe combines two LED types with different luminous colors in one LED stripe. This LED stripe can mix a spectrum of white colors, mostly from warm white to cold white. We have to wire the four cables of the LED stripe to our module as shown in the above drawing. Each output must drive 48W. Again we have a maximum output current of 2A per channel. This is far beyond the allowed 5A and ok. The 100W power supply delivers an input current of max. 4.16A. Also this current is far beyond the allowed 15A and ok.

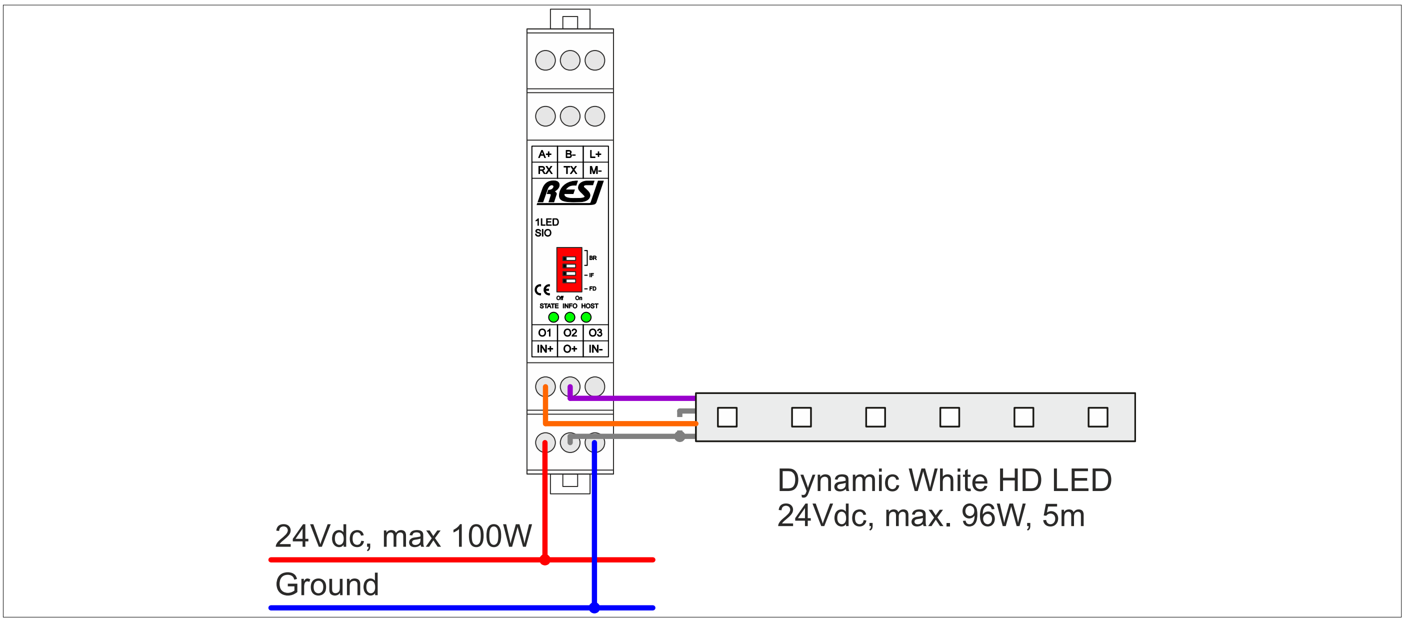


Figure: Cabling of dynamic white LED stripe

39.6.7 HOWTO connect RGBW LED stripes

In this sample we use RGBW LED HD stripe. This stripe offers four dimmable channels for the three primary colors red, green and blue and for white. The common anode is again connected to O+. We need a 100W power supply with 24Vdc, so we have 4,16A in total. This is far below the maximum of 15A. The LED stripe consumes only 1/4th of the total power of 100W on each channel. This is 25W, the current is 1,05A. Again the output current on all four outputs is much lower than the allowed 5A. But we need two modules to control all four LED channels.

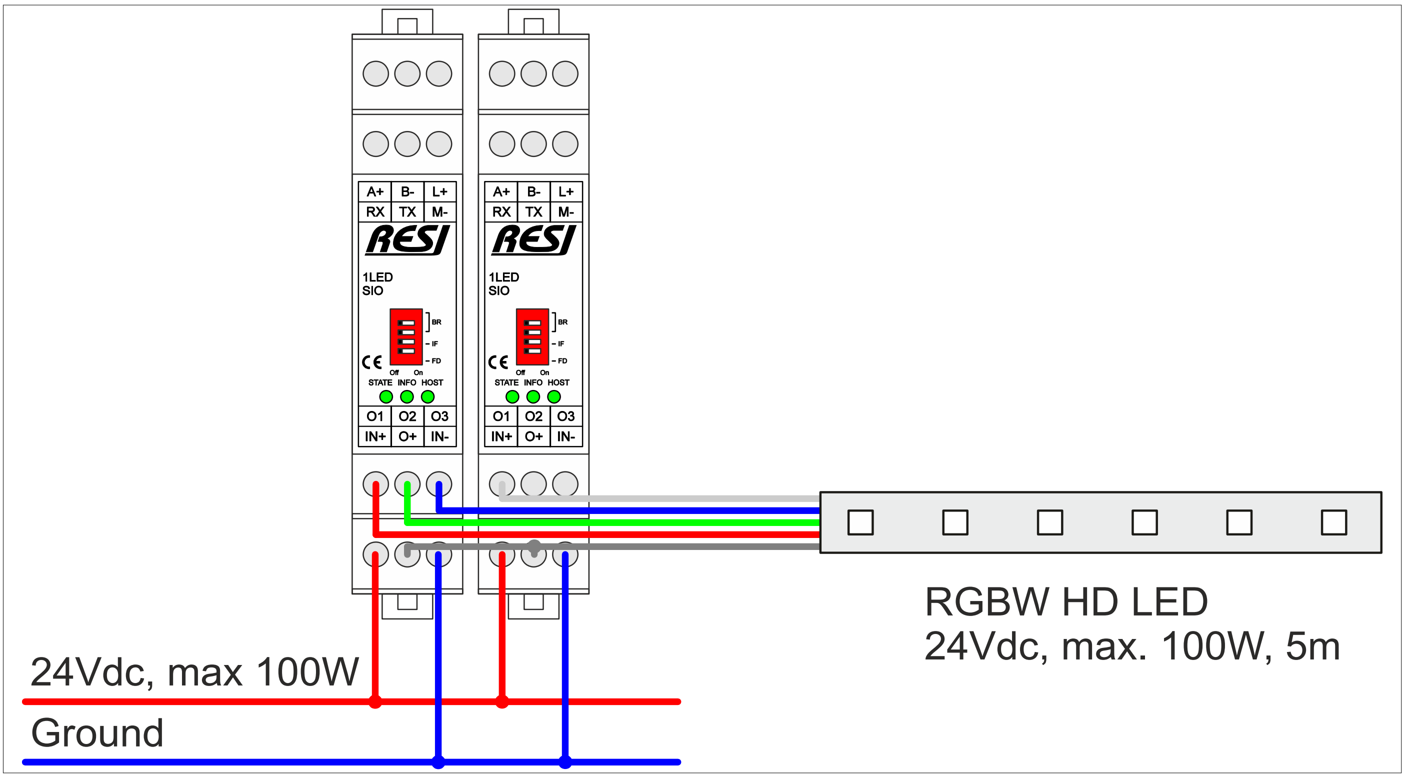


Figure: Cabling of RGBW HD LED stripe

39.7 Additional MODBUS register & coils

Here you will find only the additional MODBUS registers and coils especially for this IO module. Please refer to the description of of the standard MODBUS mapping for more details about the available basic MODBUS registers and coils.

Please refer to the external document for detailed documentation of the current MODBUS register mapping for this IO module:

RESI-L-1LED-SIO-ETH-MODBUS+ASCII-ENxx.pdf

39.8 Additional ASCII commands

Here you will find only the additional ASCII commands especially for this IO module. Please refer to the description of of the standard commands for more details about the available basic ASCII commands.

Please refer to the external document for detailed documentation of the current ASCII commands for this IO module:

RESI-L-1LED-SIO-ETH-MODBUS+ASCII-ENxx.pdf

40 RESI-1EGYDCS-SIO, RESI-1EGYDCS-ETH

40.1 General information

This series of IO modules offer the following features:

- DC smart meter with external shunt for DC current measurement
- DC voltage measurement <=100V
- DC current measurement <=100mV for external DC shunt 1..255A
- Measures & calculates: Voltage, Current, Power and Energy
- Remanent memory for accumulated total energy consumption
- Especially for Telecom 48Vdc applications
- RESI-xxx-SIO: Galvanic isolated RS232 and RS485 interface for communication with a host system
- RESI-xxx-ETH: Galvanic isolated Ethernet interface for communication with a host system

Figure: Our serial IO module

Figure: Our Ethernet IO module

40.2 Technical specification

Beside the basic technical data, which fulfil all of our IO modules, this IO modules meet the following technical specifications:

Power consumption	
RESI-1EGYDCS-SIO	<0.9W
RESI-1EGYDCS-ETH	<1.3W
Product housing	
RESI-1EGYDCS-SIO	CEM17
RESI-1EGYDCS-ETH	CEM35
Product weight	
RESI-1EGYDCS-SIO	57g
RESI-1EGYDCS-ETH	91g
DC smart meter	
DC voltage input range	0..100V=
ADC resolution	12 bit
Accuracy	+/-0.1%, if calibrated
DC current input range	0..100mV=
ADC resolution	12 bit
Accuracy	+/-0.1% if calibrated
External shunt	configurable shunt size 1 to 255A
	Shunt must deliver <=100mV output voltage!
Cable connection	via terminals
Galvanic isolation	Yes, to rest of module
Default serial settings	
Baud rate	via DIP switch
Parity	none
Stopbits	one
UnitID	255
Default Ethernet settings	
IP address	192.168.0.22
IP mask	255.255.255.0
gateway	192.168.0.1
UnitID	255
User	RESI
password	RESI

40.3 Additional terminals & LED states

DC SMART METER	Voltage and current inputs for DC smart metering	
	One 3 pin terminal block	
	Terminal type:	USLIM
	C:	Ground for voltage and current measurement
	U:	Voltage measurement input
	I:	Current measurement input for external DC shunt
Pin layout	C:	Ground for voltage and current measurement
	I:	Current measurement input for external DC shunt
	U:	Voltage measurement input
INFO	If everything is OK, this LED flashes every second.	
	If there is an internal error, this LED is always ON or OFF.	

40.4 RESI-1EGYDCS-SIO,ETH: Schematic diagram

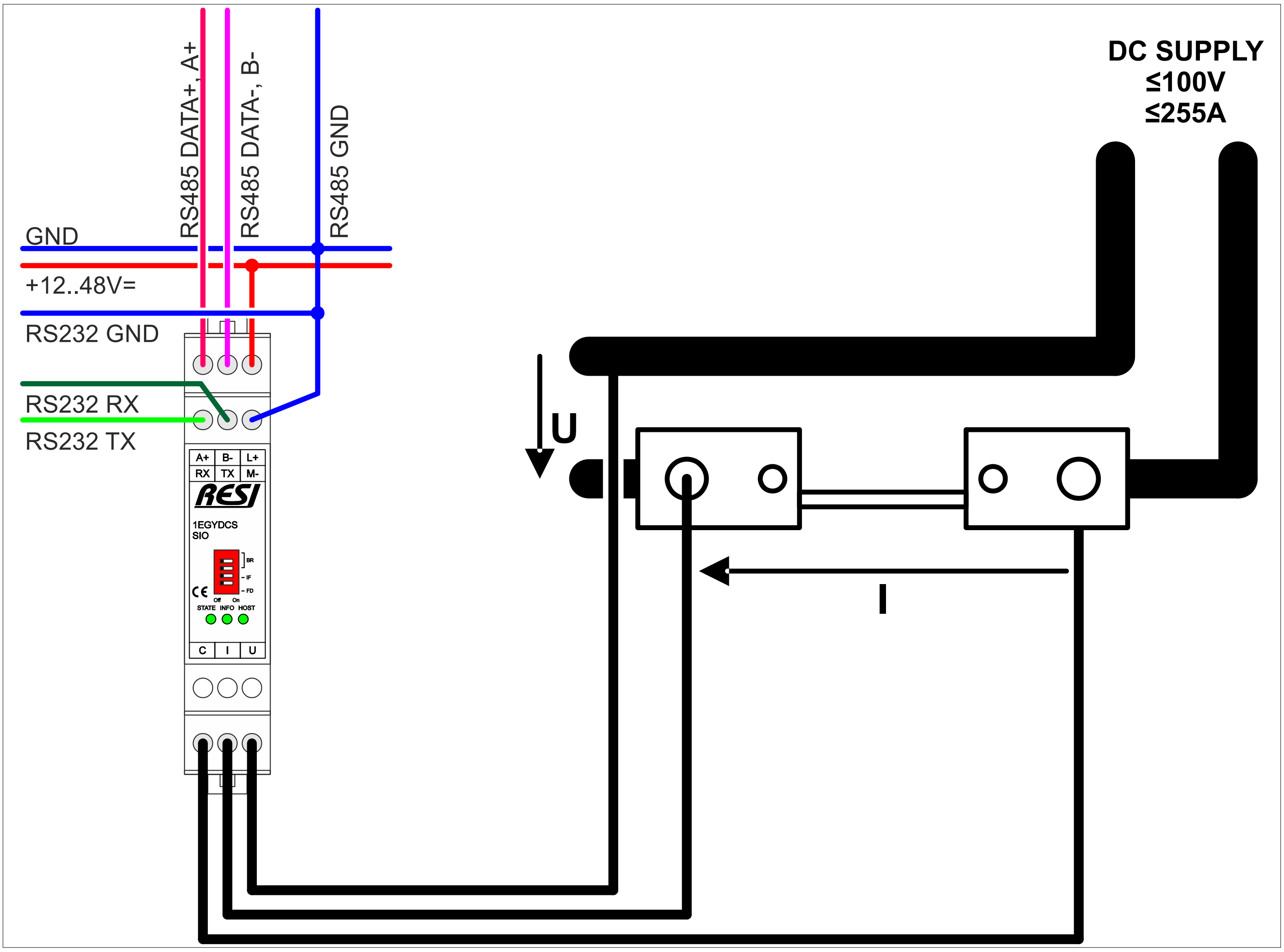


Figure: Schematics for the IO modules

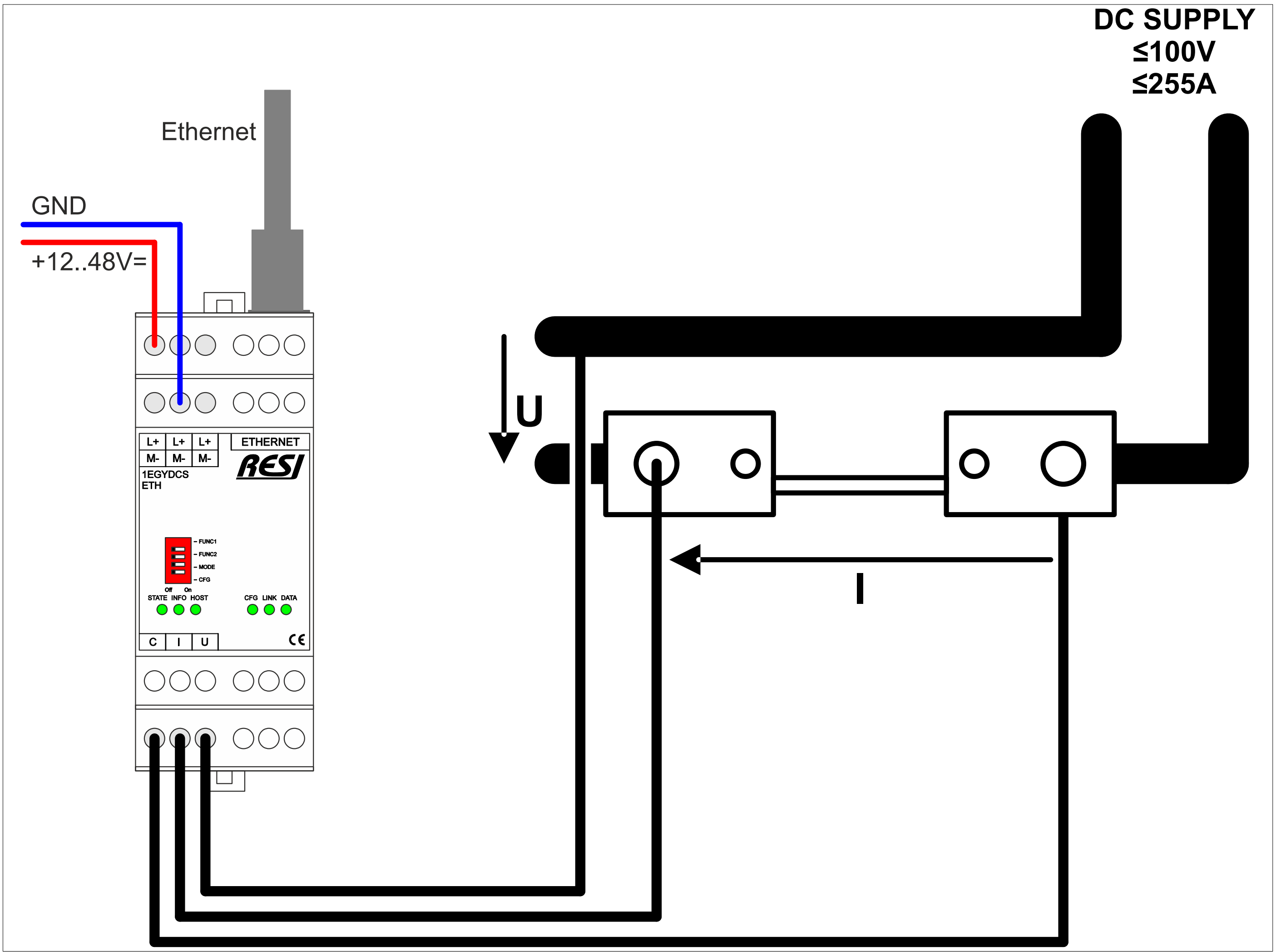


Figure: Schematics for the IO modules

40.5 Additional MODBUS register & coils

Here you will find only the additional MODBUS registers and coils especially for this IO module. Please refer to the description of of the standard MODBUS mapping for more details about the available basic MODBUS registers and coils.

Please refer to the external document for detailed documentation of the current MODBUS register mapping for this IO module:

RESI-L-1EGYDCS-SIO-ETH-MODBUS+ASCII-ENxx.pdf

40.6 Additional ASCII commands

Here you will find only the additional ASCII commands especially for this IO module. Please refer to the description of of the standard commands for more details about the available basic ASCII commands.

Please refer to the external document for detailed documentation of the current ASCII commands for this IO module:

RESI-L-1EGYDCS-SIO-ETH-MODBUS+ASCII-ENxx.pdf

41 RESI-1EGYDC-SIO, RESI-1EGYDC-ETH

41.1 General information

This series of IO modules offer the following features:

- DC smart meter with external Hall sensor for DC current measurement
- DC voltage measurement <=100V
- DC current measurement with external Hall sensor max. 255A
- Measures & calculates: Voltage, Current, Power and Energy
- Remanent memory for accumulated total energy consumption
- Especially for Telecom 48Vdc applications
- RESI-xxx-SIO: Galvanic isolated RS232 and RS485 interface for communication with a host system
- RESI-xxx-ETH: Galvanic isolated Ethernet interface for communication with a host system

Figure: Our serial IO module

Figure: Our Ethernet IO module

41.2 Technical specification

Beside the basic technical data, which fulfil all of our IO modules, this IO modules meet the following technical specifications:

Power consumption	
RESI-1EGYDC-SIO	<??W, tbd
RESI-1EGYDC-ETH	<??W, tbd
Product housing	
RESI-1EGYDC-SIO	CEM17
RESI-1EGYDC-ETH	CEM35
Product weight	
RESI-1EGYDC-SIO	62g
RESI-1EGYDC-ETH	96g
DC smart meter	
DC voltage input range	0..100V=
ADC resolution	12 bit
Accuracy	+/-0.1%
DC current input range	0..255A
DC Hall sensor interface	
ADC resolution	12 bit
Accuracy	+/-0.1%
External Hall sensor	configurable Hall sensor size 1 to 255A
Cable connection	via terminals
Galvanic isolation	Yes, to rest of module
Default serial settings	
Baud rate	via DIP switch
Parity	none
Stopbits	one
UnitID	255
Default Ethernet settings	
IP address	192.168.0.23
IP mask	255.255.255.0
gateway	192.168.0.1
UnitID	255
User	RESI
password	RESI

41.3 Additional terminals & LED states

DC SMART METER	Voltage and current inputs for DC smart metering	
	Two 3 pin terminal blocks	
	Terminal type:	USLIM
	C:	Ground for voltage measurement
	U:	Voltage measurement input
	H+, H-:	Power supply of Hall sensor
	HI, HR:	Hall sensor signals
Pin layout	C:	Ground for voltage measurement
	HI:	Hall sensor signal
	U:	Voltage measurement input
	H+:	Power supply of Hall sensor
	HR:	Hall sensor signal
	H-:	Power supply of Hall sensor
INFO	If everything is OK, this LED flashes every second.	
	If there is an internal error, this LED is always ON or OFF.	

41.4 RESI-1EGYDC-SIO,ETH: Schematic diagram

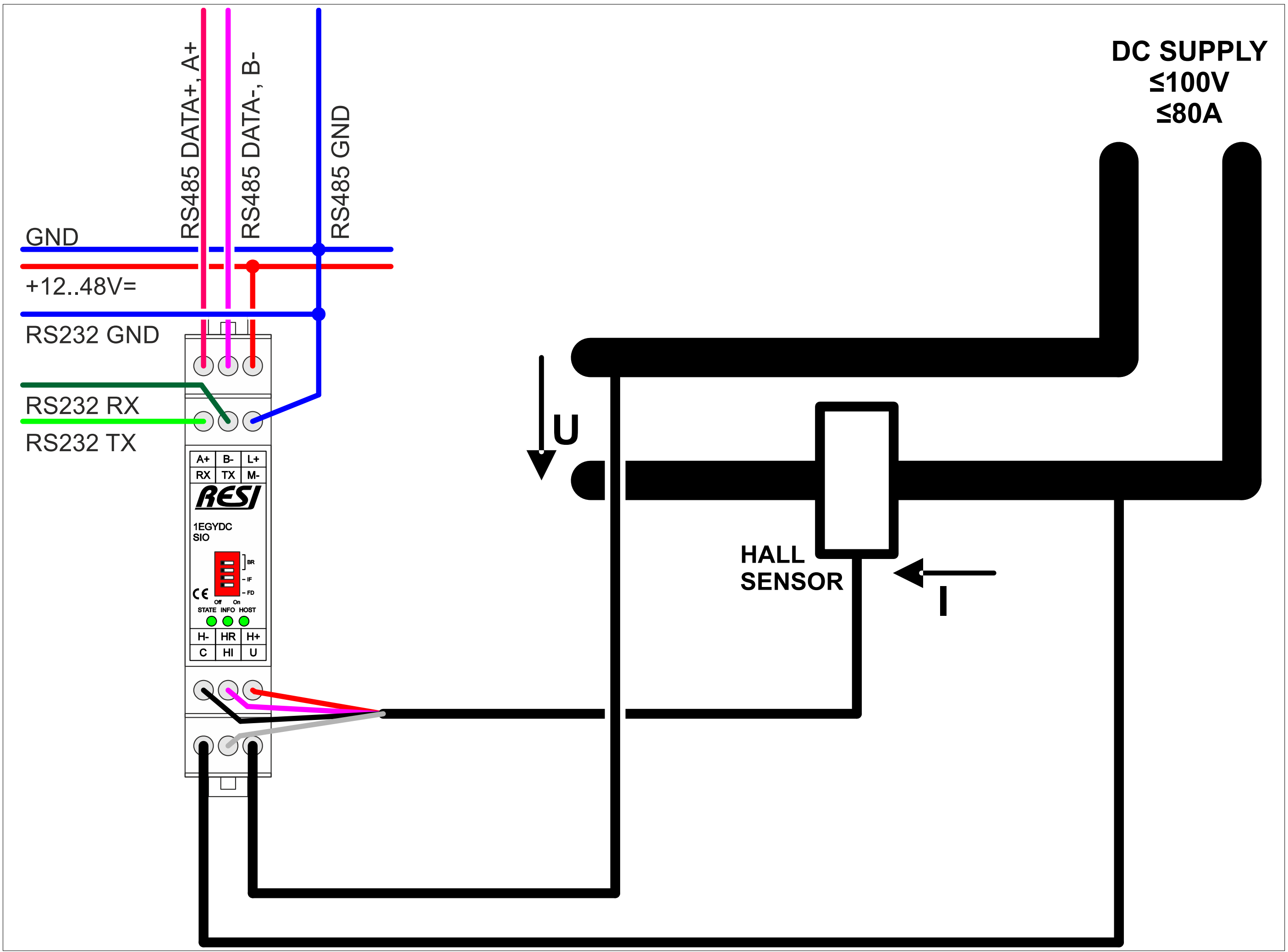


Figure: Schematics for the IO modules

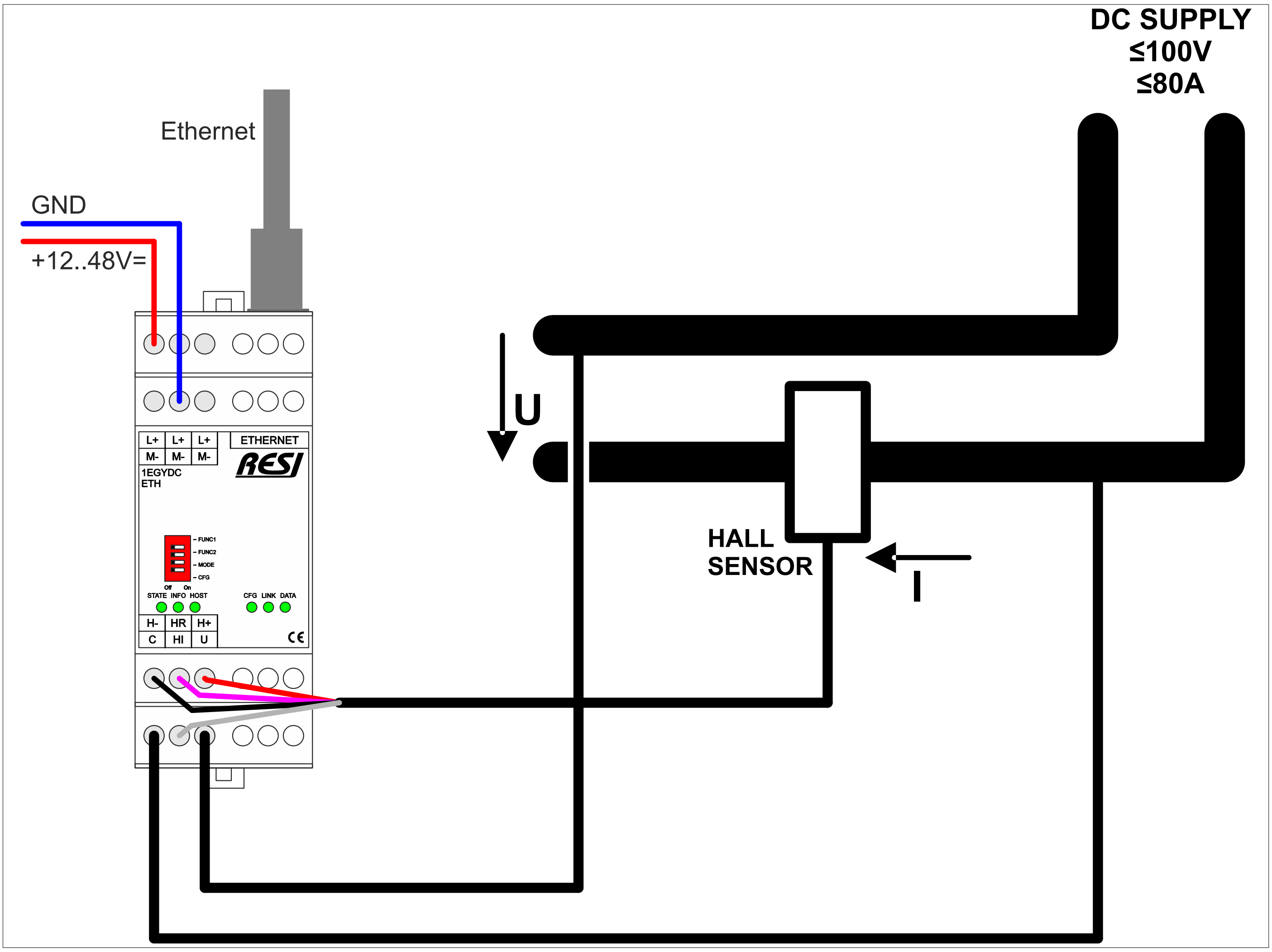


Figure: Schematics for the IO modules

41.5 Additional MODBUS register & coils

Here you will find only the additional MODBUS registers and coils especially for this IO module. Please refer to the description of of the standard MODBUS mapping for more details about the available basic MODBUS registers and coils.

Please refer to the external document for detailed documentation of the current MODBUS register mapping for this IO module:

RESI-L-1EGYDC-SIO-ETH-MODBUS+ASCII-ENxx.pdf

41.6 Additional ASCII commands

Here you will find only the additional ASCII commands especially for this IO module. Please refer to the description of of the standard commands for more details about the available basic ASCII commands.

Please refer to the external document for detailed documentation of the current ASCII commands for this IO module:

RESI-L-1EGYDC-SIO-ETH-MODBUS+ASCII-ENxx.pdf

42 RESI-DMX-SIO, RESI-DMX-ETH

42.1 General information

Our RESI-DMX-SIO, RESI-DMX-ETH converter are designed for controlling a DMX light system with a DMX universe of 512 DMX addresses. The control is done with simple ASCII commands or via MODBUS/RTU registers.

To control our DMX converter you need a host system with a serial interface (RS232 or RS485), which is able to send ASCII command strings and which can receive ASCII characters. This feature is implemented in almost any media control system like CRESTRON®, AMX® or CONTROL4®. But almost every standard PLC can handle serial ASCII interfaces. Therefore your converter can be integrated everywhere. If the host system offers a MODBUS/RTU master interface, our converter can be controlled via MODBUS holding registers.

This series of IO modules offer the following features:

- Connection of up to 512 DMX lamps (depending on the extension of the DMX network)
- Control of a complete DMX universe with 512 DMX registers
- Adjustable DMX refresh rate between 30ms and 60s.
- DMX interface: 250kBaud
- RESI-xxx-SIO: Galvanic isolated RS232 and RS485 interface for communication with a host system
- RESI-xxx-ETH: Galvanic isolated Ethernet interface for communication with a host system

Figure: Our serial DMX module

Figure: Our Ethernet IO module

42.2 Technical specification

Beside the basic technical data, which fulfil all of our IO modules, this IO modules meet the following technical specifications:

Power consumption	
RESI-DMX-SIO	<0.5W
RESI-DMX-ETH	<0.9W
Product housing	
RESI-DMX-SIO	CEM17
RESI-DMX-ETH	CEM35
Product weight	
RESI-DMX-SIO	55g
RESI-DMX-ETH	89g
DMX bus interface	
Protocol	DMX512
Baud rate	250kBit/s
Refresh rate	30ms-60s, configureable
	Standard: 100ms
Cable connection	via terminals
Galvanic isolation	Yes
Default serial settings	
Baud rate	via DIP switch
Parity	none
Stopbits	one
UnitID	255
Default Ethernet settings	
IP address	192.168.0.190
IP mask	255.255.255.0
gateway	192.168.0.1
UnitID	255
User	RESI
password	RESI

42.3 Additional terminals & LED states

DMX INTERFACE	DMX512 master connector	
	One 3 pin terminal blocks	
	Terminal type:	USLIM
	D+:	DATA+ Signal for DMX bus system
	D-:	DATA- Signal for DMX bus system
	DG:	Ground for DMX bus system
Pin layout	D+:	DATA+ Signal for DMX bus system
	D-:	DATA- Signal for DMX bus system
	DG:	Ground connector for DMX bus system
DMX	If the DMX master is activated and sends cyclically data to the DMX bus, this LED is on	
	If the DMX master is stopped and no DMX commands are send to the DMX bus, this	
	LEDs flashes fast (250ms cycle).	

42.4 RESI-DMX-SIO: Connection diagram

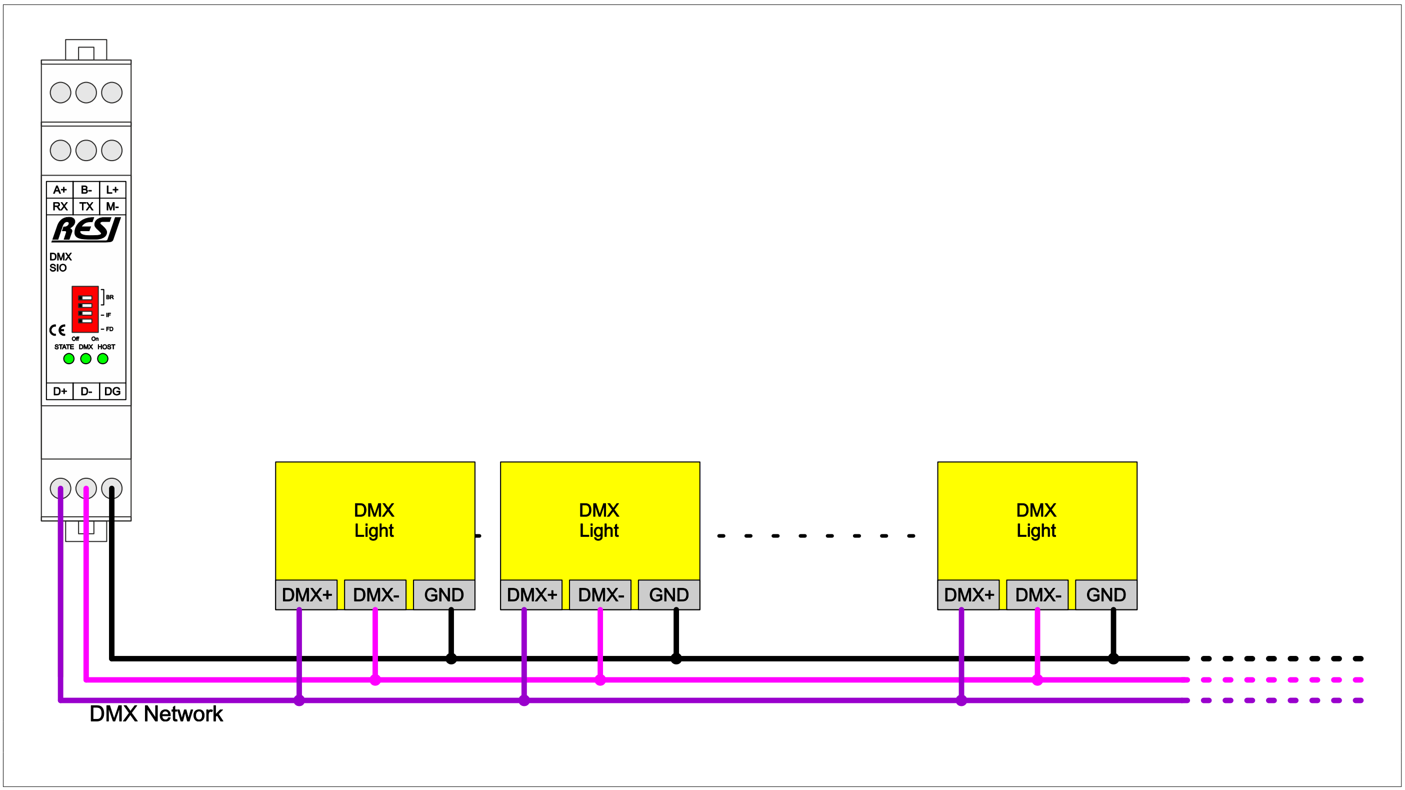


Figure: Connecting a DMX light system to the RESI-DMX-SIO gateway

42.5 RESI-DMX-ETH: Connection diagram

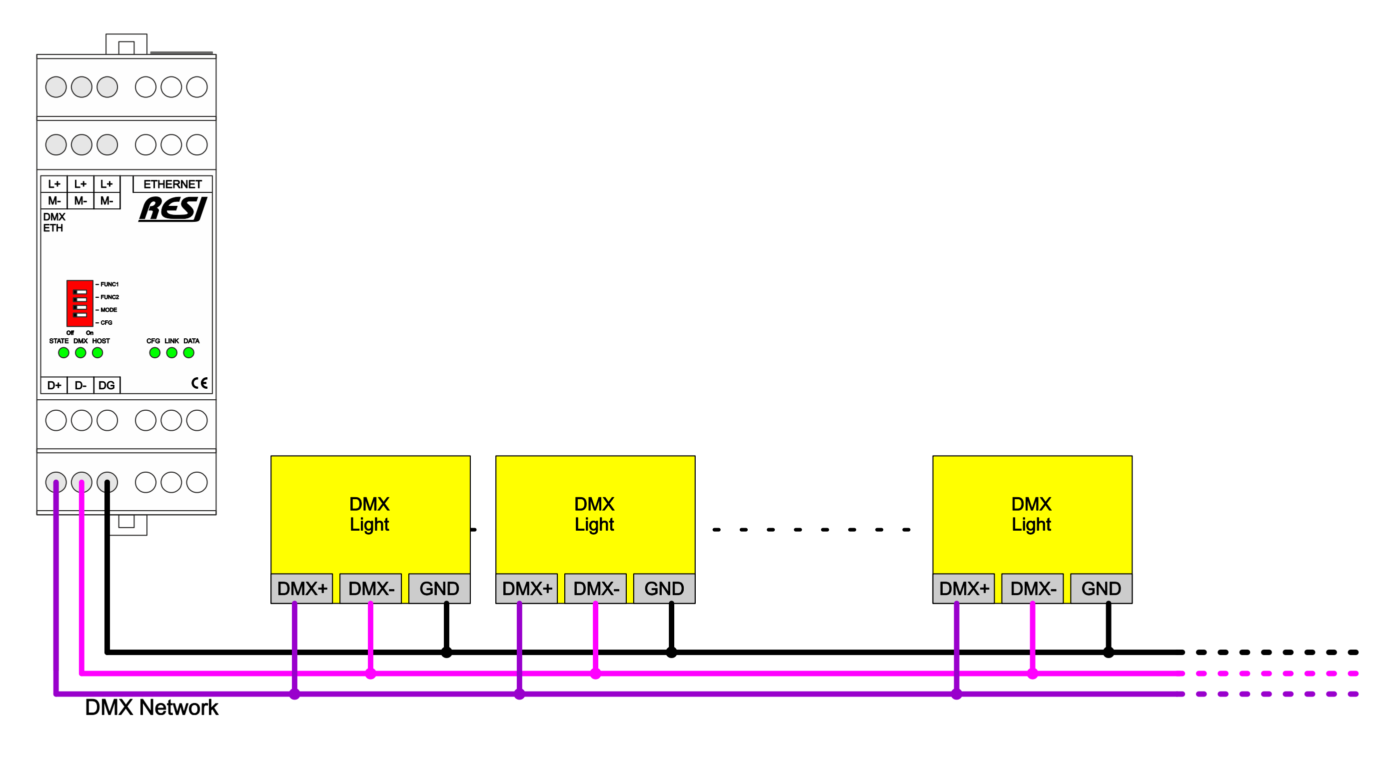


Figure: Connecting a DMX light system to the RESI-DMX-ETH gateway

42.6 Additional MODBUS register & coils

Here you will find only the additional MODBUS registers and coils especially for this IO module. Please refer to the description of of the standard MODBUS mapping for more details about the available basic MODBUS registers and coils.

Please refer to the external document for detailed documentation of the current MODBUS register mapping for this IO module:

RESI-L-DMX-SIO-ETH-MODBUS+ASCII-ENxx.pdf

42.7 Additional ASCII commands

Here you will find only the additional ASCII commands especially for this IO module. Please refer to the description of of the standard commands for more details about the available basic ASCII commands.

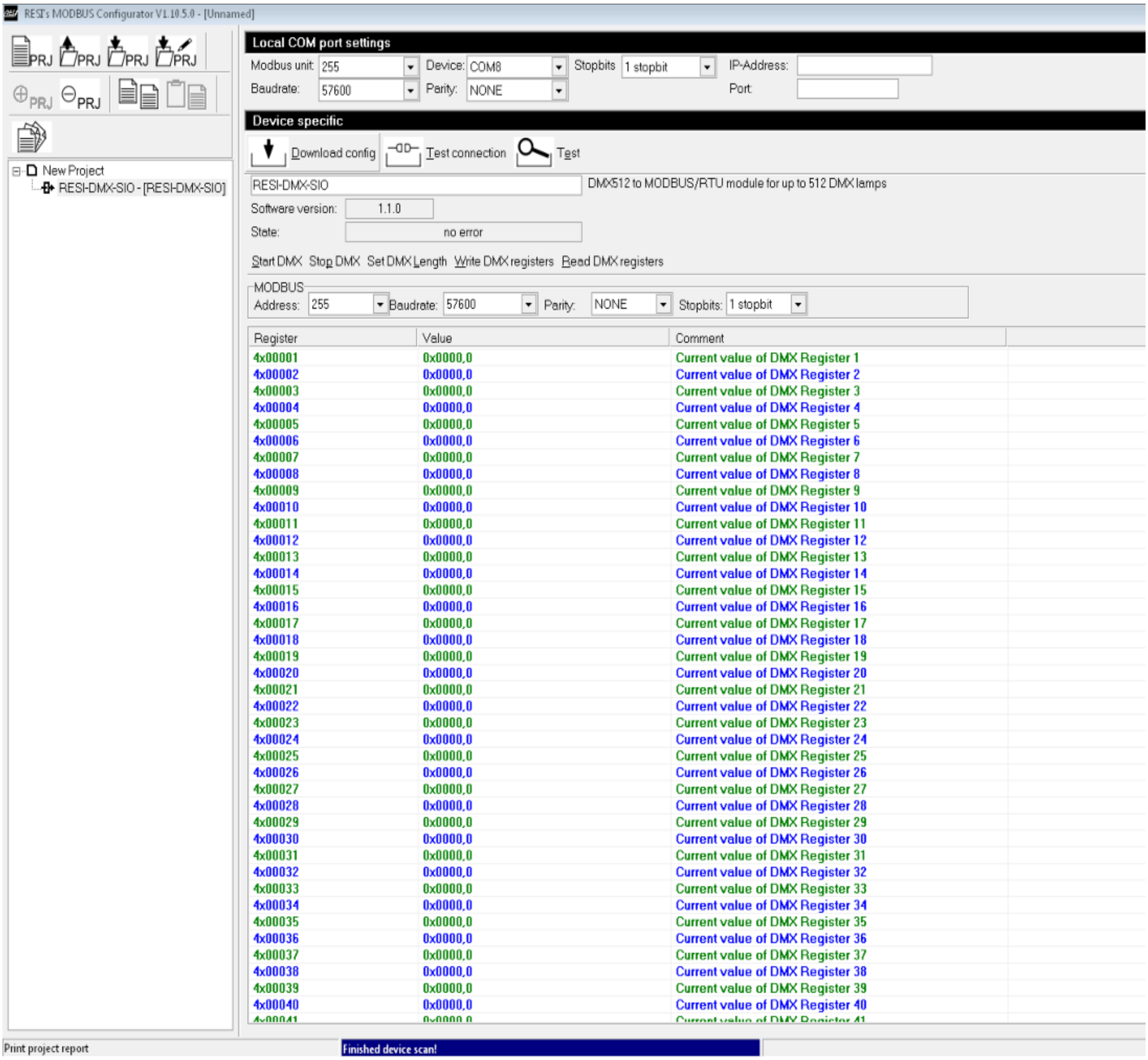
Please refer to the external document for detailed documentation of the current ASCII commands for this IO module:

RESI-L-DMX-SIO-ETH-MODBUS+ASCII-ENxx.pdf

42.8 Additional MODBUSConverter software information

Click on the add to project button to open a dialog with all available IO modules and gateways. Then select the section DMX modules… and select RESI-DMX-SIO or RESI-DMX-ETH to add this device to your project. Or you search the connected module automatically.

Your screen should look like this:



42.8.1 Start DMX

Click on this button to start the DMX master in the gateway. Normally after power on the DMX master is always activated. The DMX LED will be always ON to signal, that the gateway sends cyclic telegrams to the connected DMX lamps with the defined refresh rate.

42.8.2 Stop DMX

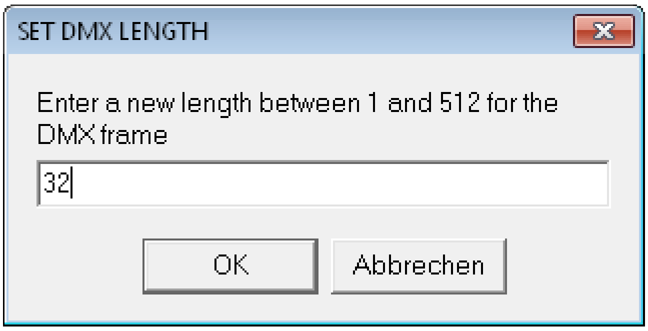
Click on this button to stop the DMX master in the gateway. The DMX LED will flash fast (~250ms cycle) to signal, that the gateway do not send cyclic telegrams to the connected DMX lamps anymore.

HINT: If the DMX lamps support a timeout function, after the defined timeout in the DMX lamp, the lamp will activate a certain brightness or color to signal the DMX bus failure.

42.8.3 Set DMX Length

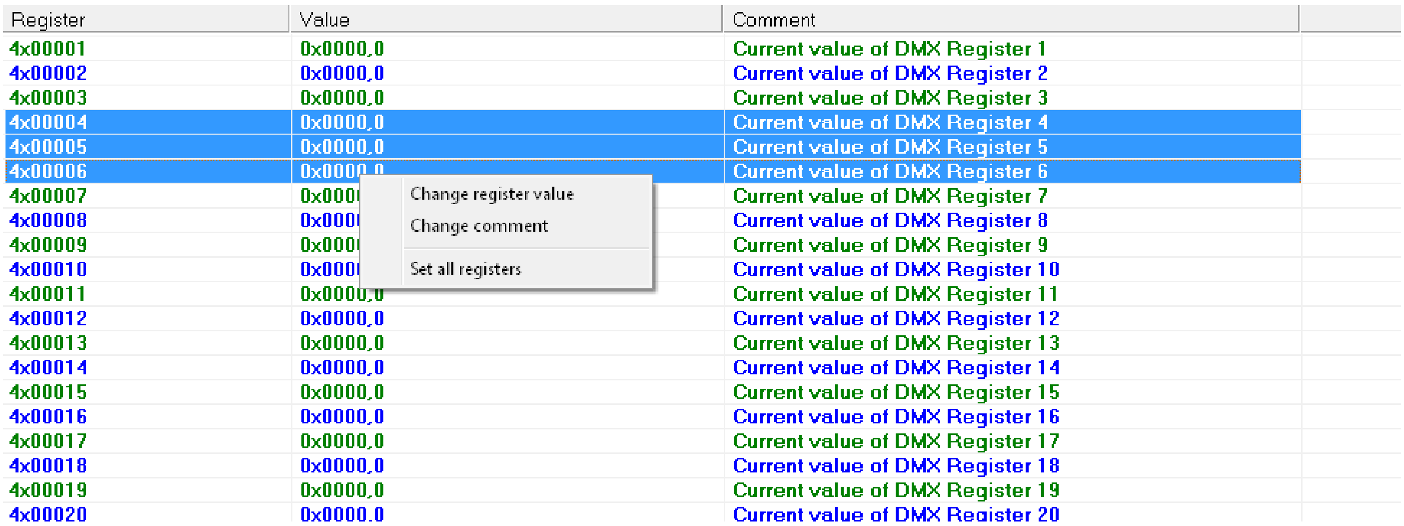
Click on this button to open an input box for the DMX telegram length, which is used in the gateway to communicate with the DMX lamps. You can enter a number between 1 and 512. This value is stored in the internal FLASH memory, so the gateway will use this setting also after a power off/power on cycle.

You can reduce the amount of bytes transmitted on the DMX bus. But be aware that if you send less bytes than your DMX lamps need, some of the lamps do not react on the DMX telegrams!



42.8.4 Edit DMX registers

Select one or more lines in the DMX register list and press the right mouse button. You will see the following popup menu. Select Change register value to define a new value for all selected registers in the range of 0 to 255 or select Change comment to enter a new comment for all selected DMX registers. If you double click on one line in the grid, the same menu will open. If you select Set all registers you can enter a new value for all 512 DMX registers between 0 and 255. Don't forget to click on Write DMX registers to download your changes to the DMX gateway, if you want to test the DMX lamps.



Register	Value	Comment
4x00001	0x0000,0	Current value of DMX Register 1
4x00002	0x0000,0	Current value of DMX Register 2
4x00003	0x0000,0	Current value of DMX Register 3
4x00004	0x0000,0	Current value of DMX Register 4
4x00005	0x0000,0	Current value of DMX Register 5
4x00006	0x0000	Current value of DMX Register 6
4x00007	0x000	Current value of DMX Register 7
4x00008	0x000	Current value of DMX Register 8
4x00009	0x000	Current value of DMX Register 9
4x00010	0x000	Current value of DMX Register 10
4x00011	0x0000	Current value of DMX Register 11
4x00012	0x0000,0	Current value of DMX Register 12
4x00013	0x0000,0	Current value of DMX Register 13
4x00014	0x0000,0	Current value of DMX Register 14
4x00015	0x0000,0	Current value of DMX Register 15
4x00016	0x0000,0	Current value of DMX Register 16
4x00017	0x0000,0	Current value of DMX Register 17
4x00018	0x0000,0	Current value of DMX Register 18
4x00019	0x0000,0	Current value of DMX Register 19
4x00020	0x0000,0	Current value of DMX Register 20

42.8.5 Write DMX registers

Click on this button to download all 512 DMX registers from the MODBUSConfigurator into the internal DMX register area of the DMX gateway. Your connected DMX lamps should react immediately to the new values in the DMX registers.

42.8.6 Read DMX registers

Click on this button to upload all 512 DMX registers from the internal DMX register area of the DMX gateway into the MODBUSConfigurator software. You can then save the project or generate a project report to save the current settings of the DMX lamps for documentation purposes.

42.8.7 Special DMX registers

At the end of the grid you will find some special registers shown in the below picture:

Register	Value	Comment	
4x00507	0x0000,0	Current value of DMX Register 507	
4x00508	0x0000,0	Current value of DMX Register 508	
4x00509	0x0000,0	Current value of DMX Register 509	
4x00510	0x0000,0	Current value of DMX Register 510	
4x00511	0x0000,0	Current value of DMX Register 511	
4x00512	0x0000,0	Current value of DMX Register 512	
4x10001	0x0200,512,512	Current DMX frame length (1..512)	
4x10002	0x0000,0,0	Current DMX mode (0=DMX stopped,1=DMX runs)	
4x10003	0x0000,0,0	Fill all DMX registers	
4x10004	0x0064,100,100	Current DMX frame speed (30..60000ms)	
4x10011	0x0000,0,0	Fill all RED DMX registers (1.,4.,7.,...)	
4x10012	0x0000,0,0	Fill all GREEN DMX registers (2.,5.,8.,...)	
4x10013	0x0000,0,0	Fill all BLUE DMX registers (3.,6.,9.,...)	

Current DMX frame length: This MODBUS register represents the current defined DMX frame length for a DMX telegram. Double click on this line to edit the frame length between 1 and 512.

Current DMX mode: This MODBUS register represents the current mode of the DMX gateway:
=0: The DMX gateway is stopped and no DMX telegrams are send to the bus.
=1: The DMX gateway runs and cyclic DMX telegrams are send by the gateway.
Use the buttons Start DMX and Stop DMX to change the DMX mode.

Fill all DMX registers: Double click on this register. Enter a new DMX register value between 0 and 255 for all 512 DMX registers in the gateway. Be aware, that you have to upload the new register values from the gateway to the MODBUSConfigurator with the button Read DMX registers.

Fill all RED DMX registers: Basically the same function like Fill all DMX registers, but it fills only every 3rd byte starting with byte 1. This is useful, when you use RGB spots which usually use 3 DMX registers to represent red, green and blue part of the color.

Fill all GREEN DMX registers: Basically the same function like Fill all DMX registers, but it fills only every 3rd byte starting with byte 2. This is useful, when you use RGB spots which usually use 3 DMX registers to represent red, green and blue part of the color.

Fill all BLUE DMX registers: Basically the same function like Fill all DMX registers, but it fills only every 3rd byte starting with byte 3. This is useful, when you use RGB spots which usually use 3 DMX registers to represent red, green and blue part of the color.

Current DMX frame speed: This register shows the current configured pause time between two DMX frames on the DMX bus. Usually 100ms. Double click on this line, enter a new pause time between 30 and 6000ms in the dialog to change this interval. Don't forget, that the gateway stores this information in the FLASH and uses the new values after a power off and power on cycle!

43 RESI-MBUSx-SIO, RESI-MBUSx-ETH

43.1 General information

This series of IO modules offer the following features:

- MBUS master interface to collect data from up to 64 smart meter with MBUS protocol
- Automatic conversion of MBUS data from MBUS data types to MODBUS register data types
- Status readout for each MBUS device
- Integrated MBUS power supply
- free PC based configuration tool for MBUS to MODBUS mapping of meter data
- RESI-xxx-SIO: Galvanic isolated RS232 and RS485 interface for communication with a host system
- RESI-xxx-ETH: Galvanic isolated Ethernet interface for communication with a host system

We offer the following different MBUS models:

- **RESI-MBUS2-SIO**: MBUS master gateway for readout of MBUS data from 2 MBUS smart meter, up to 40 MODBUS holding registers/20 mappings from MBUS to MODBUS, serial RS232 and RS485 interface
  → Former product RESI-MBUST-MODBUS
- **RESI-MBUS8-SIO**: MBUS master gateway for readout of MBUS data from 8 MBUS smart meter, up to 400 MODBUS holding registers/200 mappings from MBUS to MODBUS, serial RS232 and RS485 interface
  → Former product RESI-MBUS-MODBUS
- **RESI-MBUS24-SIO**: MBUS master gateway for readout of MBUS data from 24 MBUS smart meter, up to 1000 MODBUS holding registers/500 mappings from MBUS to MODBUS, serial RS232 and RS485 interface
  → Former product RESI-MBUS2-MODBUS
- **RESI-MBUS48-SIO**: MBUS master gateway for readout of MBUS data from 48 MBUS smart meter, up to 1200 MODBUS holding registers/600 mappings from MBUS to MODBUS, serial RS232 and RS485 interface
  → Former product RESI-MBUS3-MODBUS
- **RESI-MBUS64-SIO**: MBUS master gateway for readout of MBUS data from 64 MBUS smart meter, up to 1200 MODBUS holding registers/600 mappings from MBUS to MODBUS, serial RS232 and RS485 interface
  → New product

- **RESI-MBUS2-ETH**: MBUS master gateway for readout of MBUS data from 2 MBUS smart meter, up to 40 MODBUS holding registers/20 mappings from MBUS to MODBUS, Ethernet interface
  → Former product RESI-MBUST-ETH
- **RESI-MBUS8-ETH**: MBUS master gateway for readout of MBUS data from 8 MBUS smart meter, up to 400 MODBUS holding registers/200 mappings from MBUS to MODBUS, Ethernet interface
  → Former product RESI-MBUS-ETH
- **RESI-MBUS24-ETH**: MBUS master gateway for readout of MBUS data from 24 MBUS smart meter, up to 1000 MODBUS holding registers/500 mappings from MBUS to MODBUS, Ethernet interface
  → Former product RESI-MBUS2-ETH
- **RESI-MBUS48-ETH**: MBUS master gateway for readout of MBUS data from 48 MBUS smart meter, up to 1200 MODBUS holding registers/600 mappings from MBUS to MODBUS, Ethernet interface
  → Former product RESI-MBUS3-ETH
- **RESI-MBUS64-ETH**: MBUS master gateway for readout of MBUS data from 64 MBUS smart meter, up to 1200 MODBUS holding registers/600 mappings from MBUS to MODBUS, Ethernet interface
  → New product

The amount of meters are defined by the standard unit load of 1.5mA per meter. Please note that many meters need more current from the MBUS power supply, so the total number of meters may not recht the maximum of the used module.

Figure: Our serial IO module

Figure: Our Ethernet IO module

43.2 Technical specification

Beside the basic technical data, which fulfil all of our IO modules, this IO modules meet the following technical specifications:

Power consumption	
RESI-MBUSx-SIO	<0.6W no connected MBUS meter
	<2.0W shortcut on MBUS line
	<8.0W MBUS line overload for short time
RESI-MBUSx-ETH	<1.0W no connected MBUS meter
	<2.4W shortcut on MBUS line
	<8.4W MBUS line overload for short time

Product housing	
RESI-MBUSx-SIO	CEM17
RESI-MBUSx-ETH	CEM35

Product weight	
RESI-MBUSx-SIO	56g
RESI-MBUSx-ETH	90g

MBUS power supply	
Nominal output voltage	~34,2V
Maximum output current	~174mA shortcut on MBUS line
	~155mA MBUS line overload for short time

MBUS cabling	
Nominal cable for MBUS bus	JYStY 2x0.8mm² or
	JYStY 0x1.5mm²
Nominal cable resistance	75 Ohm/km
Nominal cable capacity	50nF/km
Maximum cable length	max. 7000m
Maximum cable capacity:	max. 180nF

HINT: The real cable length is determined how many MBUS meters you will connect to the segment and how the segment is designed (star, tree, line) and how fast you will communicate over the bus line. Please refer to the internet for more details how to build a correct MBUS meter network!

Default serial settings	
Baud rate	via DIP switch
Parity	none
Stopbits	one
UnitID	255

Default Ethernet settings

IP address	
RESI-MBUS2-ETH	192.168.0.210
RESI-MBUS8-ETH	192.168.0.211
RESI-MBUS24-ETH	192.168.0.212
RESI-MBUS48-ETH	192.168.0.213
RESI-MBUS64-ETH	192.168.0.214
IP mask	255.255.255.0
gateway	192.168.0.1
UnitID	255
User	RESI
password	RESI

43.3 Additional terminals & LED states

MBUS system	MBUS master for connection of 2/8/24/48/64 smart meters with MBUS interface	
	One 3 pin terminal blocks	
	Terminal type:	USLIM
	MB+:	Positive signal of MBUS bus system
	MB-:	Negative signal of MBUS bus system
	HINT: Swapping the two wires of the bus is also permitted and	
	does not generate any errors	
Pin layout	MB+:	Positive signal of MBUS bus system
	N/C:	not connected
	MB-:	Negative signal of MBUS bus system
STATE	If no configuration is downloaded into the module, this LED blinks very quickly (~100ms)	
	If the configuration or the module has an error this LED blinks very fast (~50ms)	
	If everything is ok this LED blinks very slow (~1s)	
MBUS	If any data is send or received by the MBUS interface, this LED flashes	

43.4 RESI-MBUSx-SIO: Connection diagram

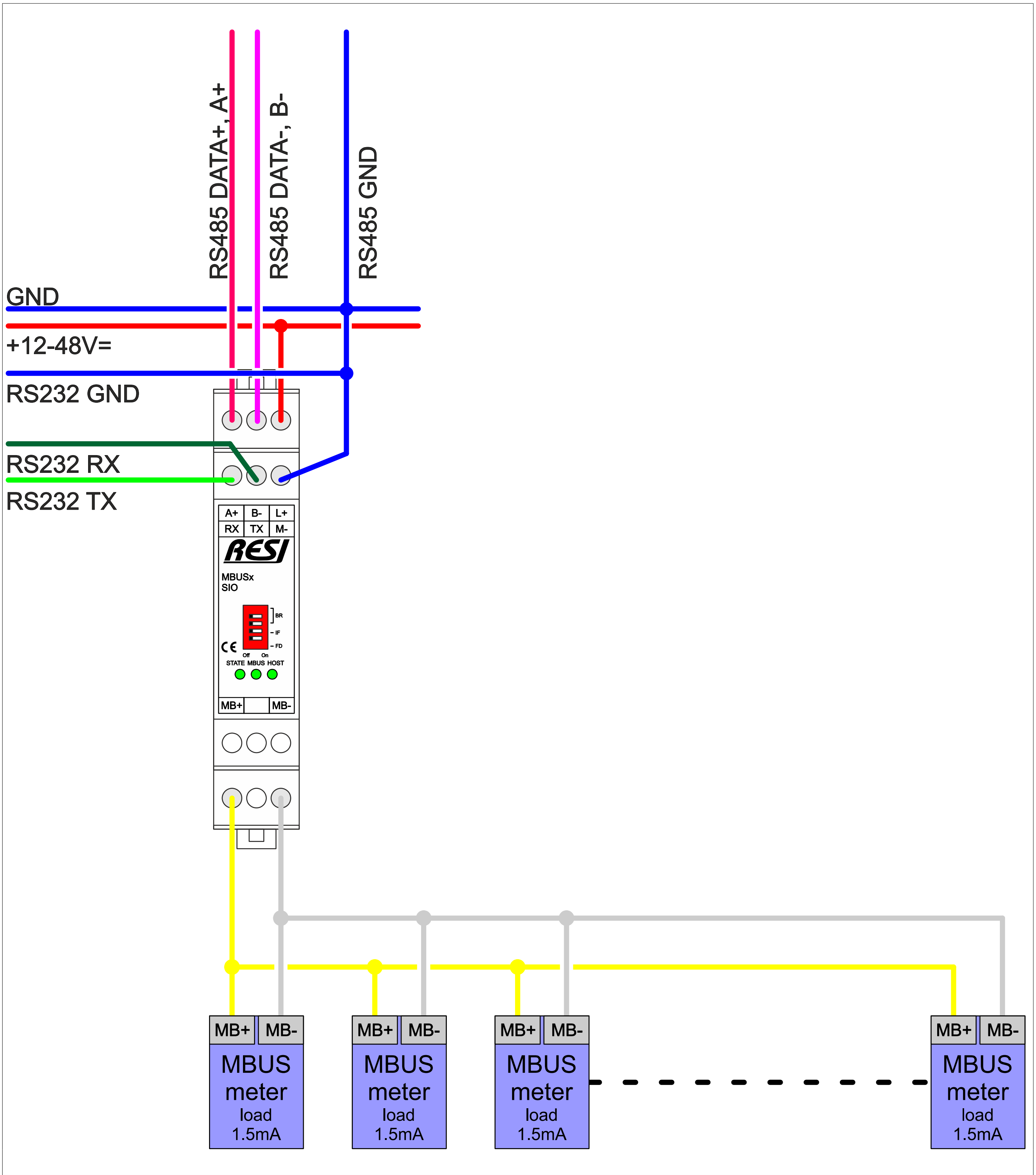


Figure: Connecting the MBUS bus system to the serial MBUSx converter

43.5 RESI-MBUSx-ETH: Connection diagram

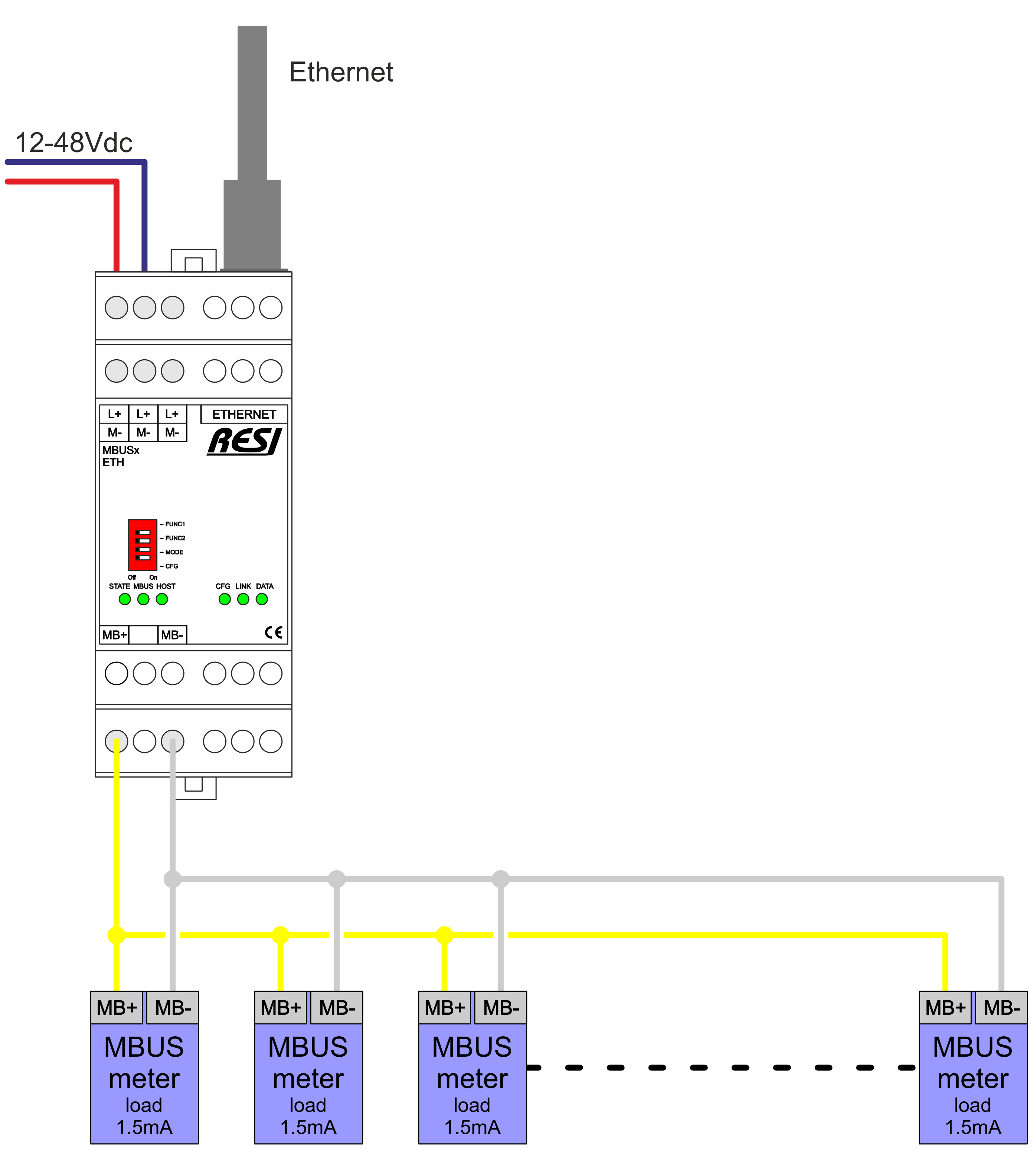


Figure: Connecting the MBUS bus system to the Ethernet MBUSx converter

43.6 MBUS bus topology

The MBUS bus topology is free. You can use star, line or tree bus topology. Only a ring topology is forbidden! The MBUS cable is a two wire cable, which connects the MBUS master (our gateway) with every connected MBUS slave (the meter). The M-Bus is polarity independent and needs no line termination resistors at the end of the cables. Any cable type may be used as long as the cable is suitable for >36V/500mA. Shielding is not necessary and not recommended since the capacity of the cable should be minimized.

In most cases a standard telephone cable is used which is a twisted pair wire with a diameter of 0.8mm each (2x0.8mm). This type of cable should be used for the main wiring. For the wiring to the meters from the main wiring (last one or two meters to the meter) a cable with smaller diameter may be used.

The maximum distance between a slave and the master is around 3km to 10km, depending on the individual network configuration. This distance applies for the standard configuration having Baud rates between 300 and 2400 Baud, and a maximum of 64 slaves. The maximum distance can be increased by limiting the Baud rate and using fewer slaves, but the bus voltage at no point in a segment fall below 24V, because of the remote powering of the slaves. In the standard configuration the total cable length should not exceed 3000 m, in order to meet the requirement of a maximum cable capacitance of 180nF.

Please refer to the internet for more details about the MBUS bus cabling and the theoretical and practical cable length.

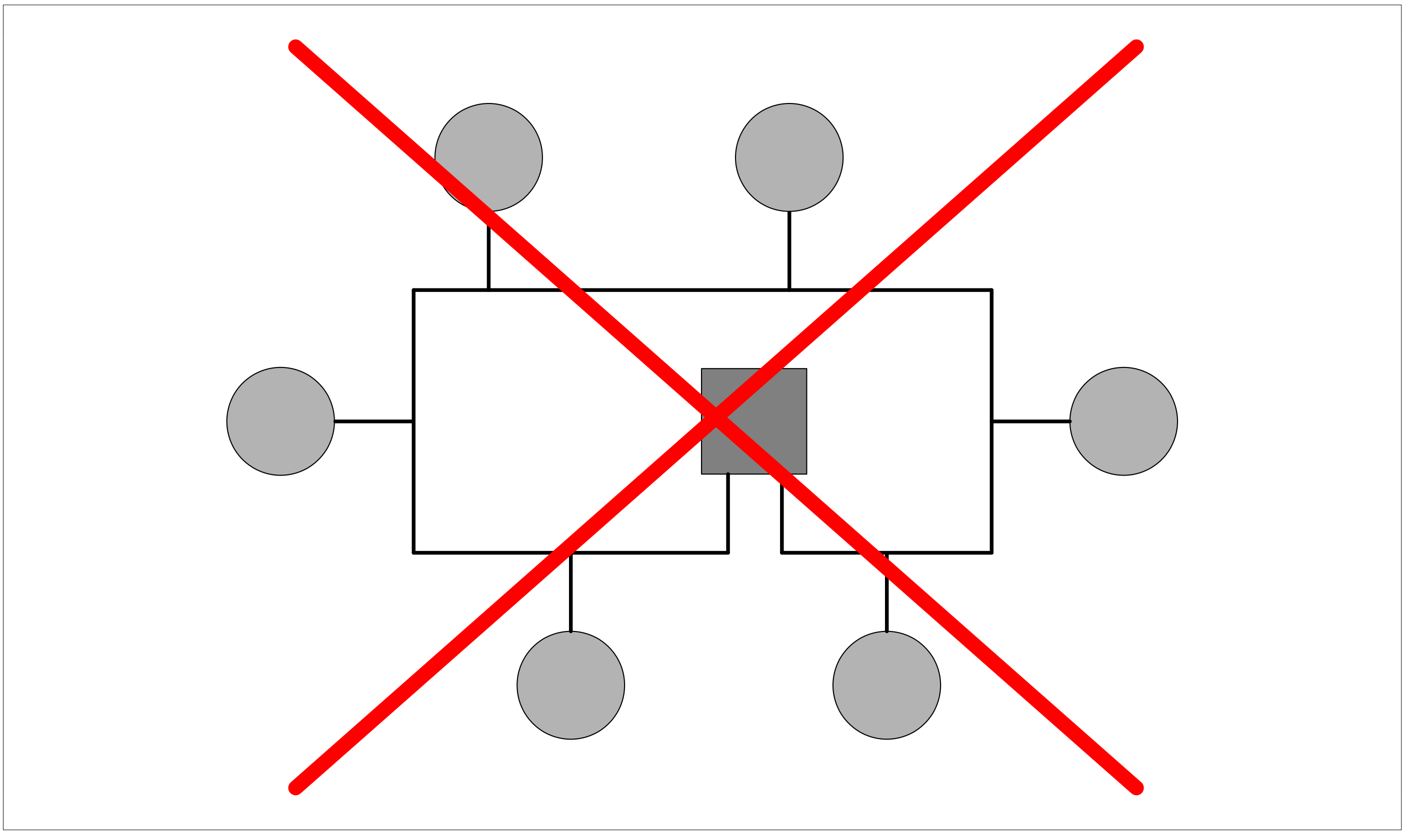


Figure: FORBIDDEN: MBUS ring topology

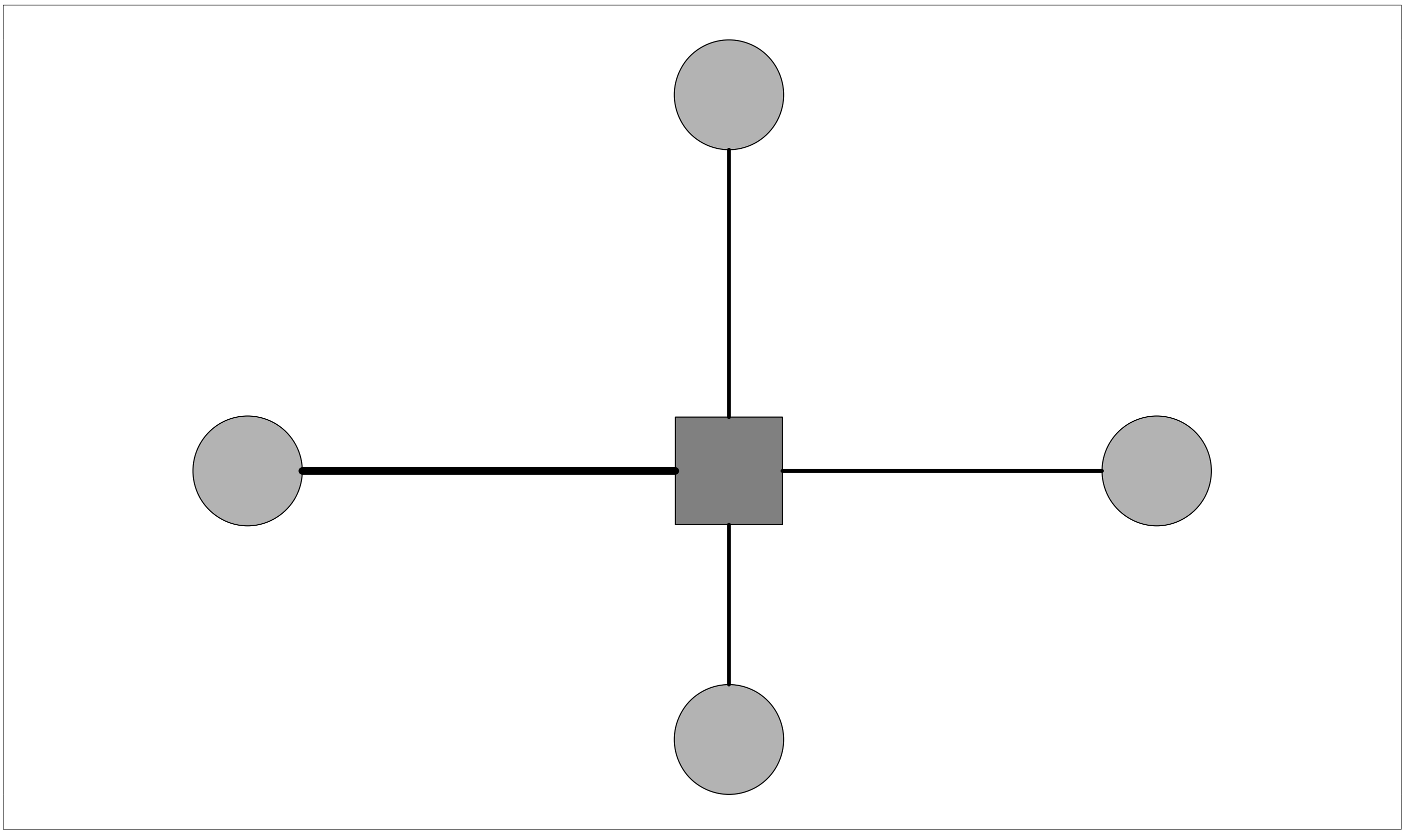


Figure: MBUS star topology

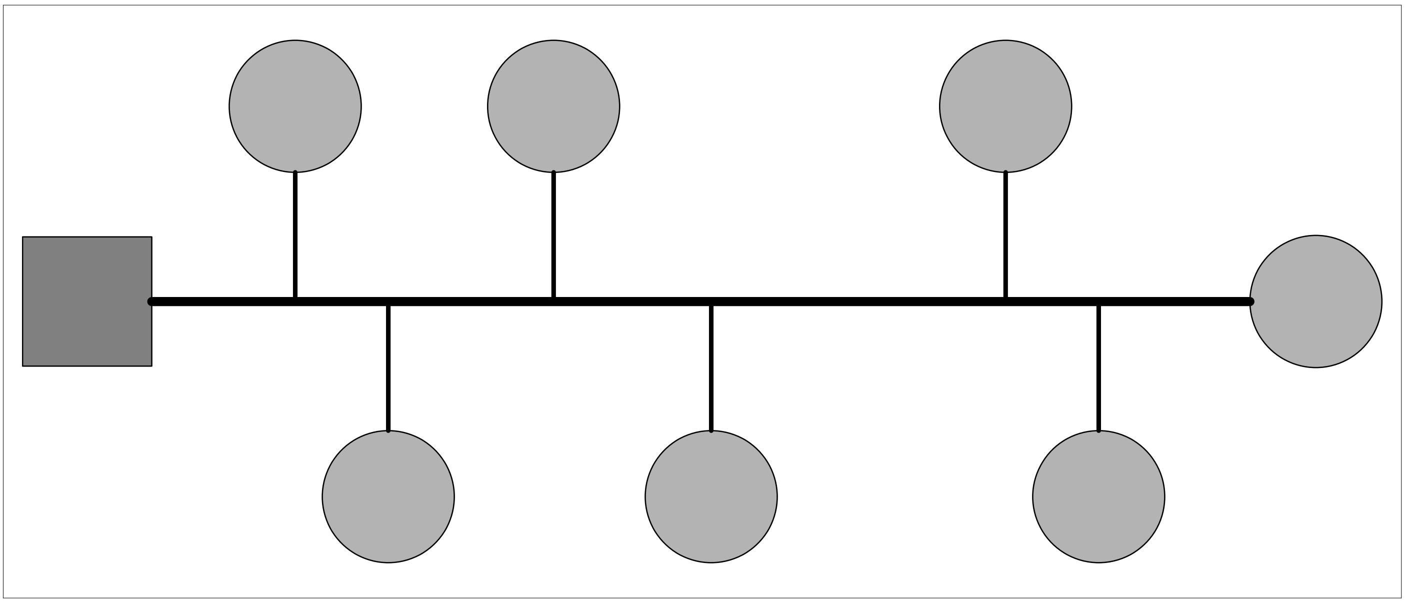


Figure: MBUS line topology

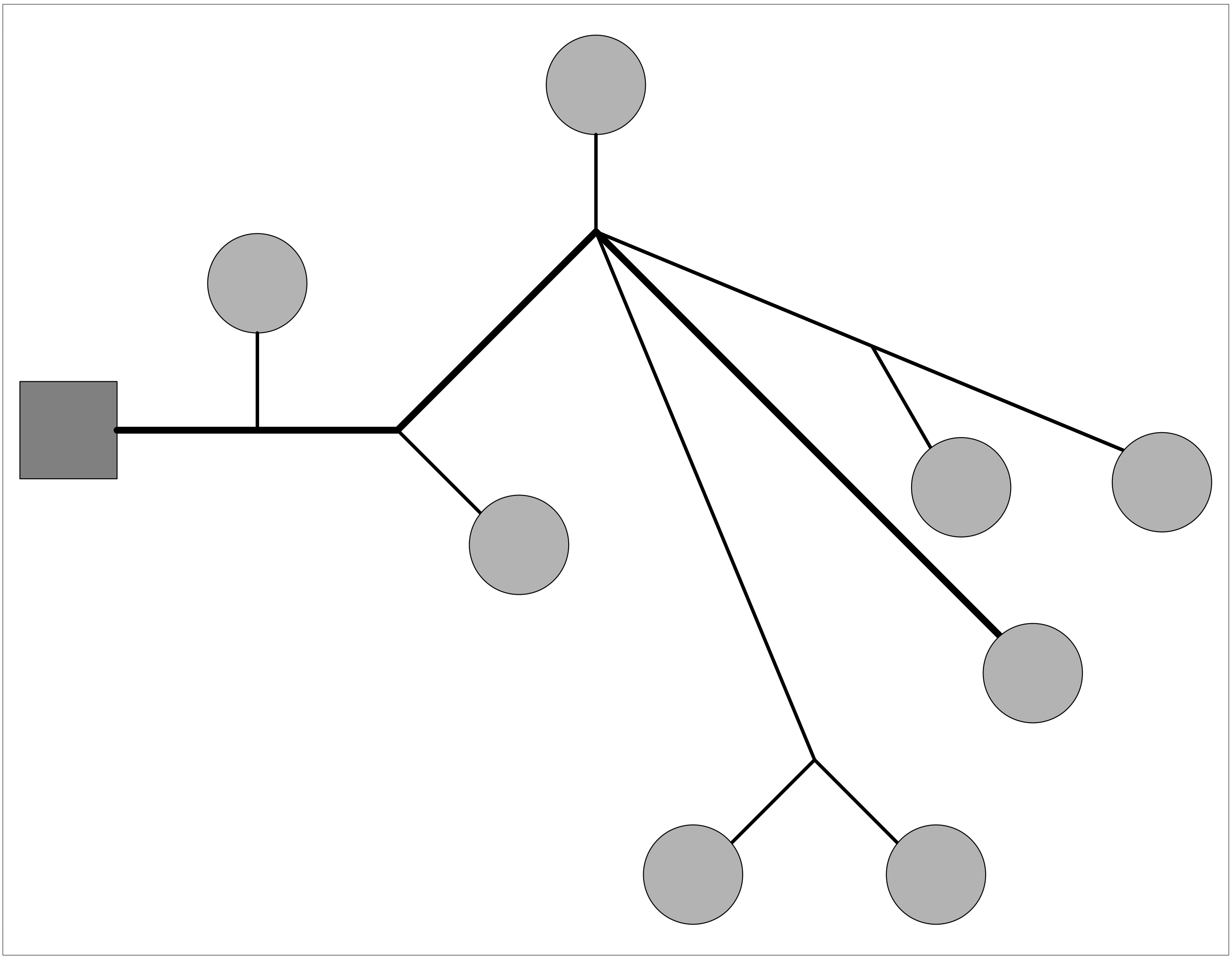


Figure: MBUS tree topology

43.7 MBUS bus recomendations

This are some recommendations for MBUS bus lines from literature out of the internet.
Don't forget: This is only a helpful hint. RESI or partners of RESI do not guarantee, that your bus system works in any case, if you follow this hints! You are responsible to plan and design your individual MBUS bus system correctly.

The resistive cable length defines the maximum length of a cable segment with in the bus structure. The cable segment length is the distance from the M-Bus Master to the M-Bus device furthest away. The capacitive cable length defines the maximum bus cable length in total.

43.7.1 Small inhouse installations

Description: small and medium-sizes residential buildings

- resistive cable length: max. 350m
- capacitive cable length: max. 1km
- cable cross section: min. 0.5mm²

Usage:

- max. 64 devices with max 9600 baud

43.7.2 large inhouse installations

Description: medium-sizes and large residential buildings

- resistive cable length: max. 350m
- capacitive cable length: max. 3km
- cable cross section: min. 0.5mm²

Usage:

- max. 64 devices with max 2400 baud

43.7.3 Small wide area installation

Description: small to medium-sized residential areas

- resistive cable length: max. 1km
- capacitive cable length: max. 4km
- cable cross section: min. 0.5mm²

Usage:

- max. 64 devices with max 2400 baud

43.7.4 Big wide area installation

Description: medium-sized to large residential areas

- resistive cable length: max. 3km
- capacitive cable length: max. 5km
- cable cross section: min. 1.5mm²

Usage:

- max. 64 devices with max 2400 baud

43.7.5 Provider network installation

Description: energy provider driven networks

- resistive cable length: max. 5km
- capacitive cable length: max. 7km
- cable cross section: min. 1.5mm²

Usage:

- max. 16 devices with max 300 baud

43.7.6 Maximum segment installation

Description:

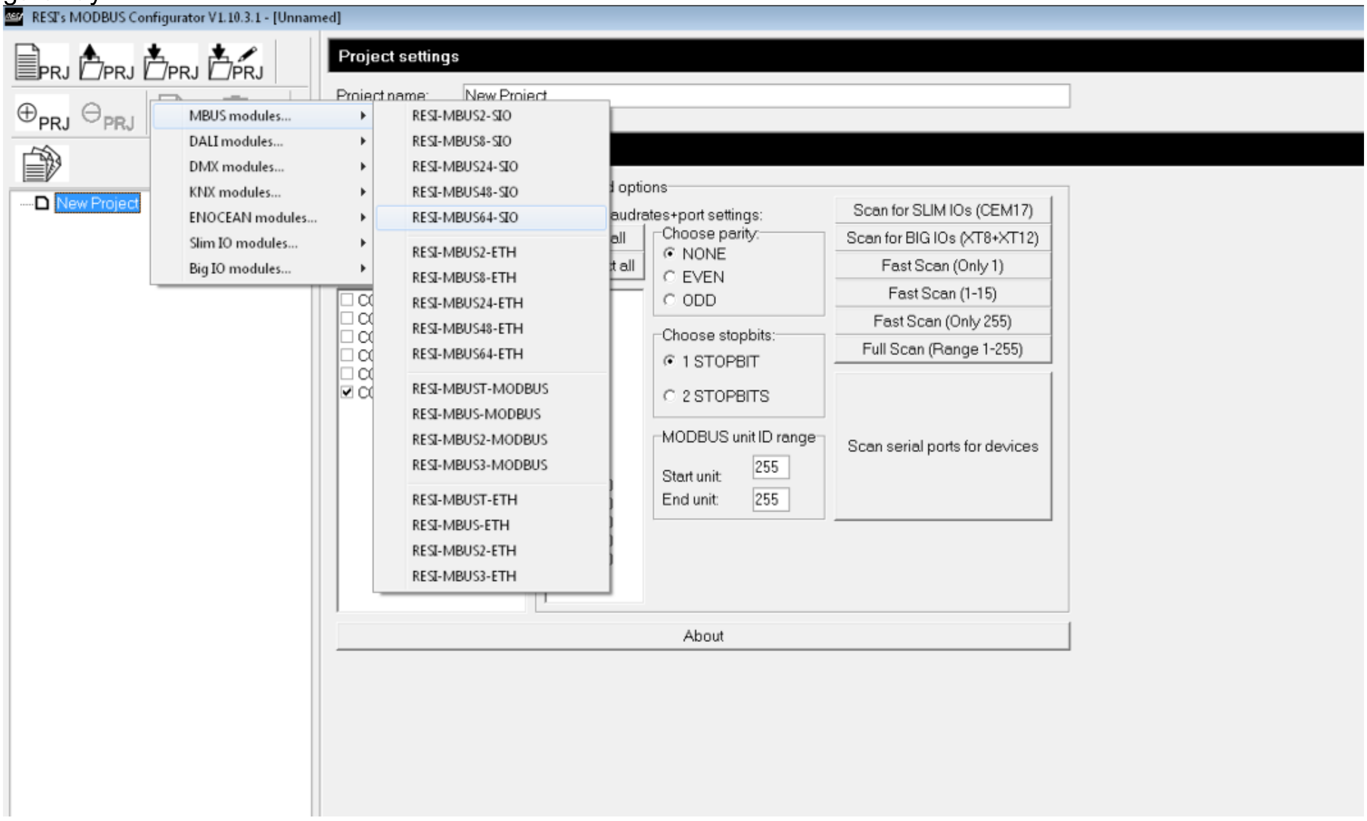
- linear topology
- cable length: max. 10km
- cable cross section: min. 1.5mm²

Usage:

- max. 1 device with max 300 baud

43.8 Add RESI-MBUSx-xxx device to project tree

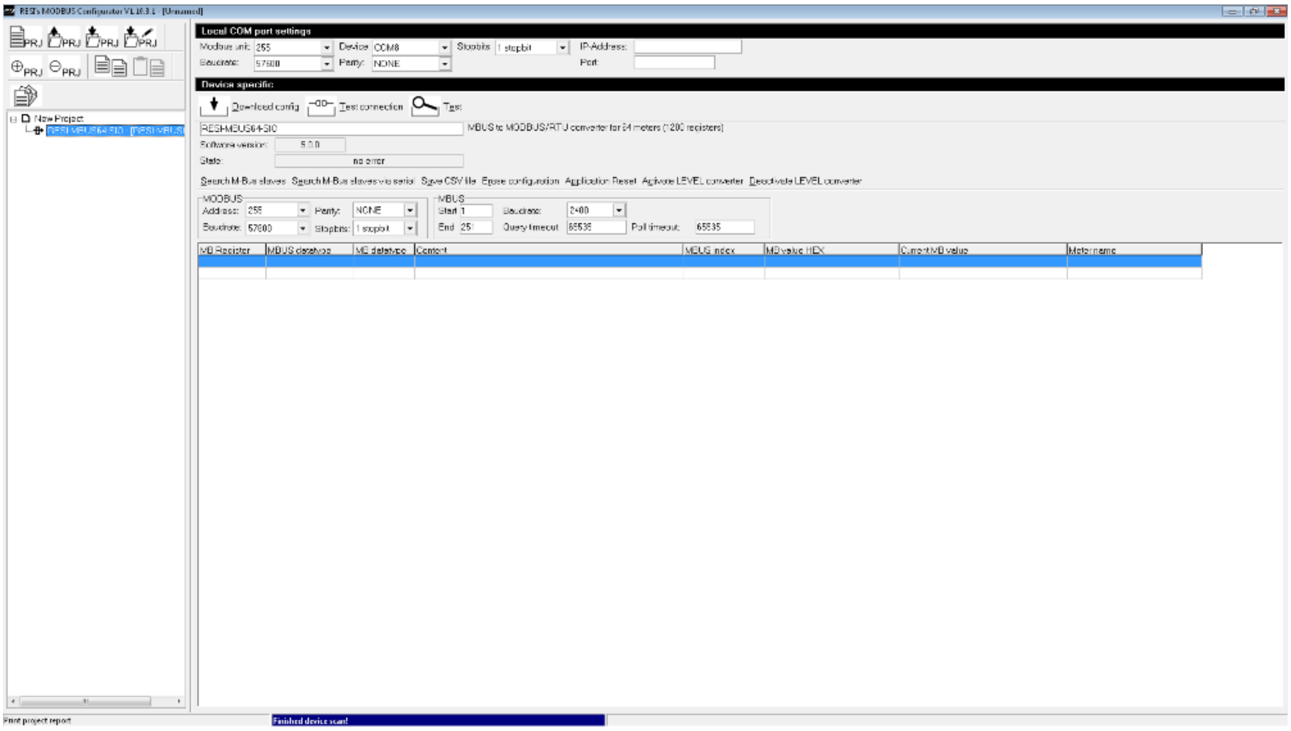
First, start the MODBUSConfigurator software. Click on the project tree title “New Project” and add a desired Ethernet gateway.



You will notice, that you find the new MBUS converters on top of the list. But for compatibility reasons, you can also add and use older modules or older projects with the old MBUS modules form us.

After you have selected your device, don't forget to set the serial or Ethernet parameters correctly for communication with your module. Or you use the automatic search function of our software.

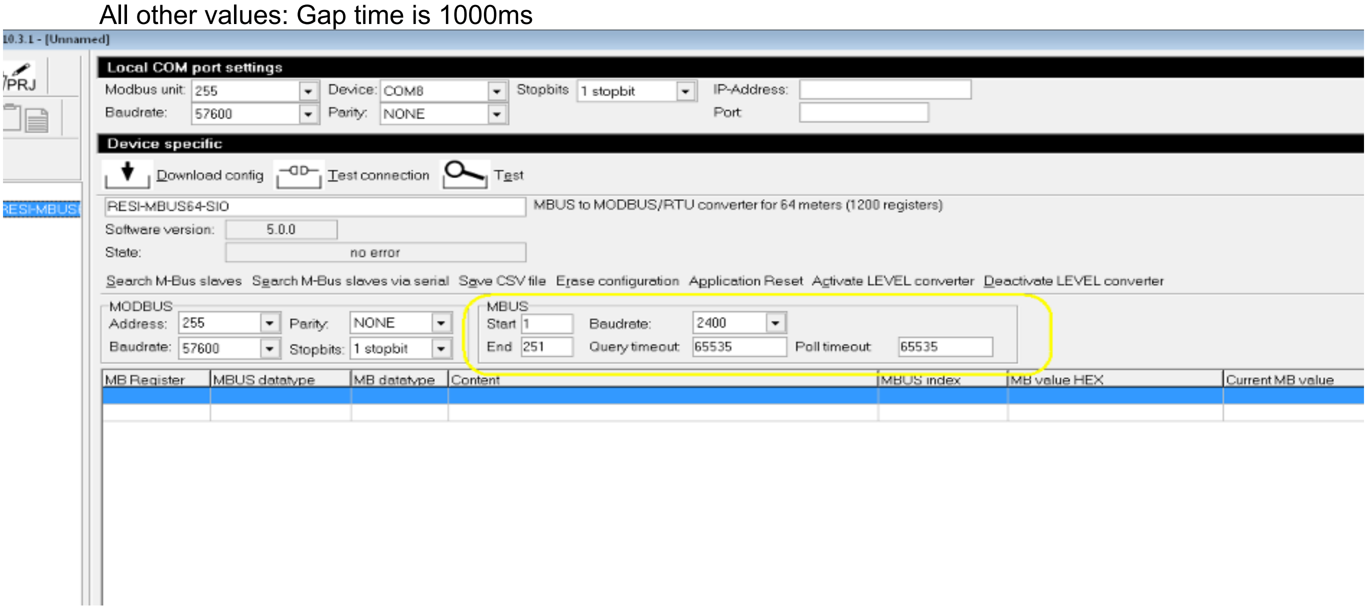
Your screen should look like this:



43.9 HOWTO setup MBUS communication parameters

First of all, you have to select the correct MBUS communication speed suitable for your meters. In the area MBUS you will find the following setup parameters:

- **Baudrate**: This is the current used MBUS baud rate on the MBUS. Usually you will see the currently configured baud rate of your converter.
- **Start, End**: This two fields define the primary address range, which will be used for an automatic search for connected MBUS slaves via primary addressing mode. You can enter a valid MBUS primary address in the range from 1 to 251. If you have connected only one meter, you can also use the primary broadcast address 254 for communication with this meter.
- **Query timeout**: This field defines the timeout between two query cycles in the gateway. Usually the gateway communicates with all configured meters sequentially. After finishing the data readout for the last meter, the gateway pauses for this defined interval in seconds. This values are used:
  Value 65535 or values 0..5 defines ~5s pause.
  Values 6 to 65534: defines 6 to 65534 seconds of pause, before the next polling cycle will start.
- **Poll timeout**: This field defines a general pause after the readout of a configured meter before the readout of the next meter starts. In the past we discovered that there are many meters out in the market, which need a special treatment in the timing. e.g. very old KAMSTRUP meters need often two readout cycles with a gap of at least 10-15 seconds. This is non standard to the MBUS. Or other meters have problems with secondary addressing, if there is a too small gap between the readout. So we introduced this new parameter: This timeout defines the pause after finishing reading of a meter and starting reading the next meter. In the previous firmware versions this timeout was fixed to 250ms gap, which was ok for 99% of the meter readout on the markets. But some meter fail to process this little gap. The values is interpreted as follows:
  Value 1..30: Gap time 1 seconds to 30 seconds
  Value 101..400: Gaptime=(Value-100)*0.1s → 0.1s .. 30s e.g. 105 → 0.5s
  Value 65535: Gap time is 1 second
  Value 65534: Gap time is 250ms
  Value 65533: Gap time is 500ms
  Value 65532: Gap time is7250ms
  All other values: Gap time is 1000ms



Change the MBUS baudrate and/or the poll & query timeouts

Follow this steps to change to communication baud rate of the MBUS bus system:

1. Select a new baud rate from the drop down list Baudrate in the MBUS area.
2. Change the query timeout to your needs
3. Change the poll timeout to your needs
4. Use the Download config button to download the new settings into your device
5. Now your device will use the new settings of the baud rate and the timeouts on the MBUS side

You can achieve the same with writing the new MBUS baud rate and the timeouts to certain MODBUS registers. Please refer to the section of the MODBUS register description, how this function will work.

Here you will find a basic diagram, how the MBUS master request cycle is handled by our gateways. The two parameters can be configured like this:

- **Query timeout**: This field defines the timeout between two query cycles in the gateway. Usually the gateway communicates with all configured meters sequentially. After finishing the data readout for the last meter, the gateway pauses for this defined interval in seconds. This values are used:
  Value 65535 or values 0..5 defines ~5s pause.
  Values 6 to 65534: defines 6 to 65534 seconds of pause, before the next polling cycle will start.
- **Poll timeout**: This field defines a general pause after the readout of a configured meter before the readout of the next meter starts. In the past we discovered that there are many meters out in the market, which need a special treatment in the timing. e.g. very old KAMSTRUP meters need often two readout cycles with a gap of at least 10-15 seconds. This is non standard to the MBUS. Or other meters have problems with secondary addressing, if there is a too small gap between the readout. So we introduced this new parameter: This timeout defines the pause after finishing reading of a meter and starting reading the next meter. In the previous firmware versions this timeout was fixed to 250ms gap, which was ok for 99% of the meter readout on the markets. But some meter fail to process this little gap. The values is interpreted as follows:
  Value 1..30: Gap time 1 seconds to 30 seconds
  Value 101..400: Gaptime=(Value-100)*0.1s → 0.1s .. 30s e.g. 105 → 0.5s
  Value 65535: Gap time is 1 second
  Value 65534: Gap time is 250ms
  Value 65533: Gap time is 500ms
  Value 65532: Gap time is7250ms
  All other values: Gap time is 1000ms

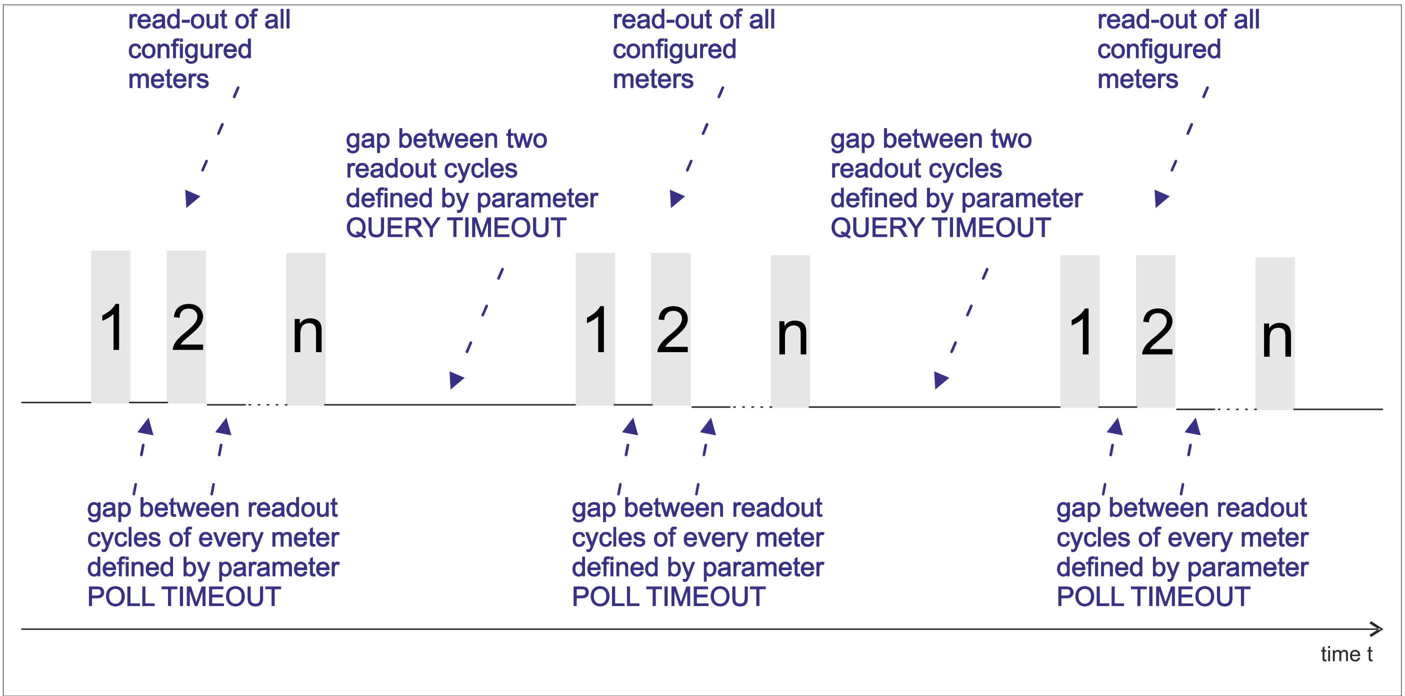


Figure: Basic timing of MBUS master read-out for MBUS slaves

Now we go more into detail, how the MBUS gateway will handle the request process of one meter. Forst we define the parameters:

- **Poll repeats 1**: This field defines the amount of telegram repetitions for the addressing command to a meter, before the gateway declares the communication as not possible and resumes with the next meter.
  Value 65535 or 0: use 3 repeats as standard
  Value 1..n: Use n repeats
- **Poll repeats 2**: This field defines the amount of telegram repetitions for the data readout command to a meter, before the gateway declares the communication as not possible and resumes with the next meter.
  Value 65535 or 0: use 5 repeats as standard
  Value 1..n: Use n repeats
- **Poll pre delay 1**: This field defines the first pause time in Milliseconds before starting to send the first addressing command telegram to a meter.
  Value 65535: use 250ms as standard pause time
  Value 0..65534: Use x ms as pause time

- **Poll pre delay 2**: This field defines the first pause time in Milliseconds before starting to send the first data request telegram to a meter.
  Value 65535: use 100ms as standard pause time
  Value 0..65534: Use x ms as pause time
- **Poll post delay 1**: This field defines a pause time in Milliseconds. If the gateway do not receive a correct answer to an addressing command telegram and the addressing command is repeated, then this pause time is inserted, before resending the addressing telegram to the meter.
  Value 65535: use 0ms as standard pause time
  Value 0..65534: Use x ms as pause time
- **Poll post delay 2**: This field defines a pause time in Milliseconds. If the gateway do not receive a correct answer to a readout data telegram and the readout data command is repeated, then this pause time is inserted, before resending the readout data telegram to the meter.
  Value 65535: use 100ms as standard pause time
  Value 0..65534: Use x ms as pause time

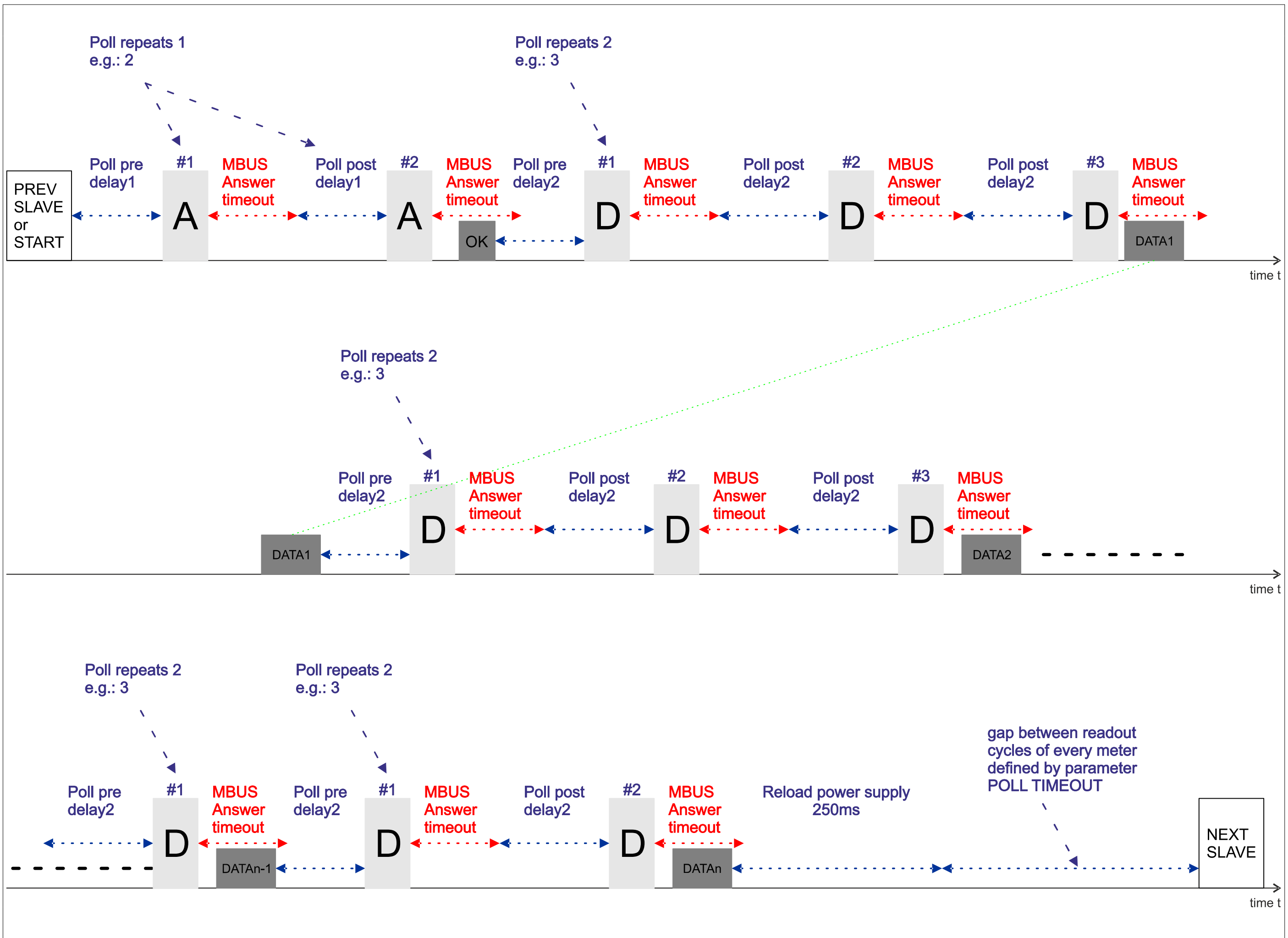


Figure: Basic timing of MBUS master read-out for MBUS slaves

43.10 HOWTO find connected MBUS meters

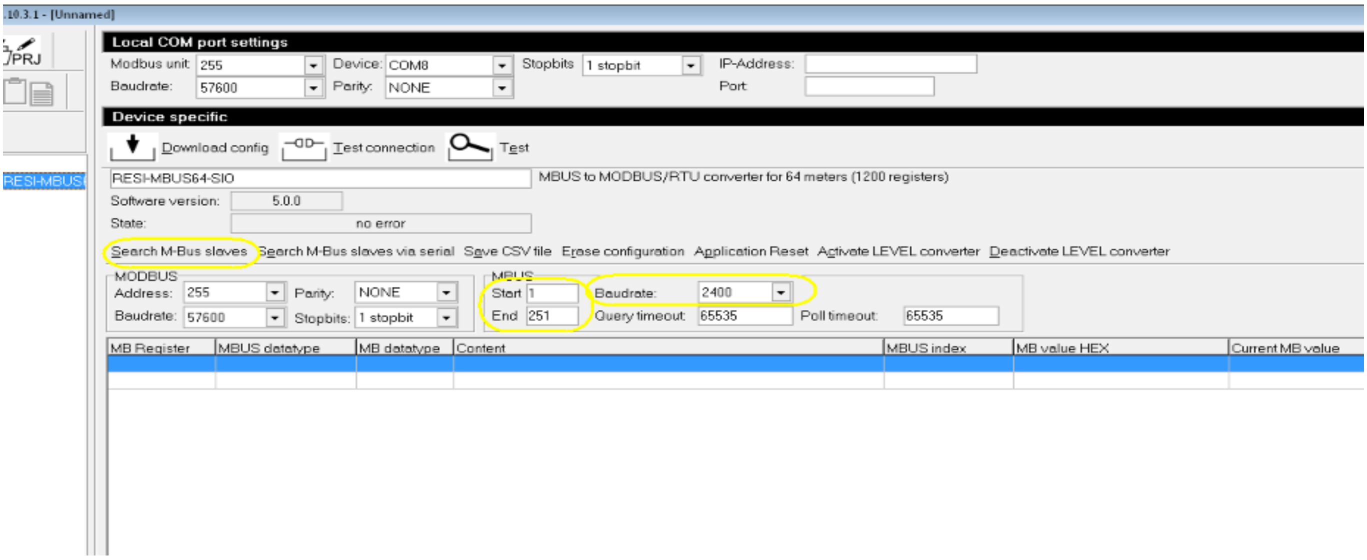
There are two ways for searching for connected MBUS meters.

- **Search M-Bus slaves with primary address**: With this function the MBUS network is scanned for new meters only by addressing the meters with the primary address. The address range is defined with the parameters Start and End in the MBUS area. Every found meter, which is not part of the configuration, will be added automatically to the project.
- **Search M-Bus slaves with secondary address**: With this function the MBUS network is scanned for new meters using secondary addressing mode with the unique serial number of the meters. Every found meter, which is not part of the configuration, will be added automatically to the project.

43.10.1 Search for new meters – primary addressing mode

HINT: Don't forget to setup the MBUS bus baud rate for your search before. If you have to change it, select a new one from the drop down list and don't forget to download the bus speed into your gateway!

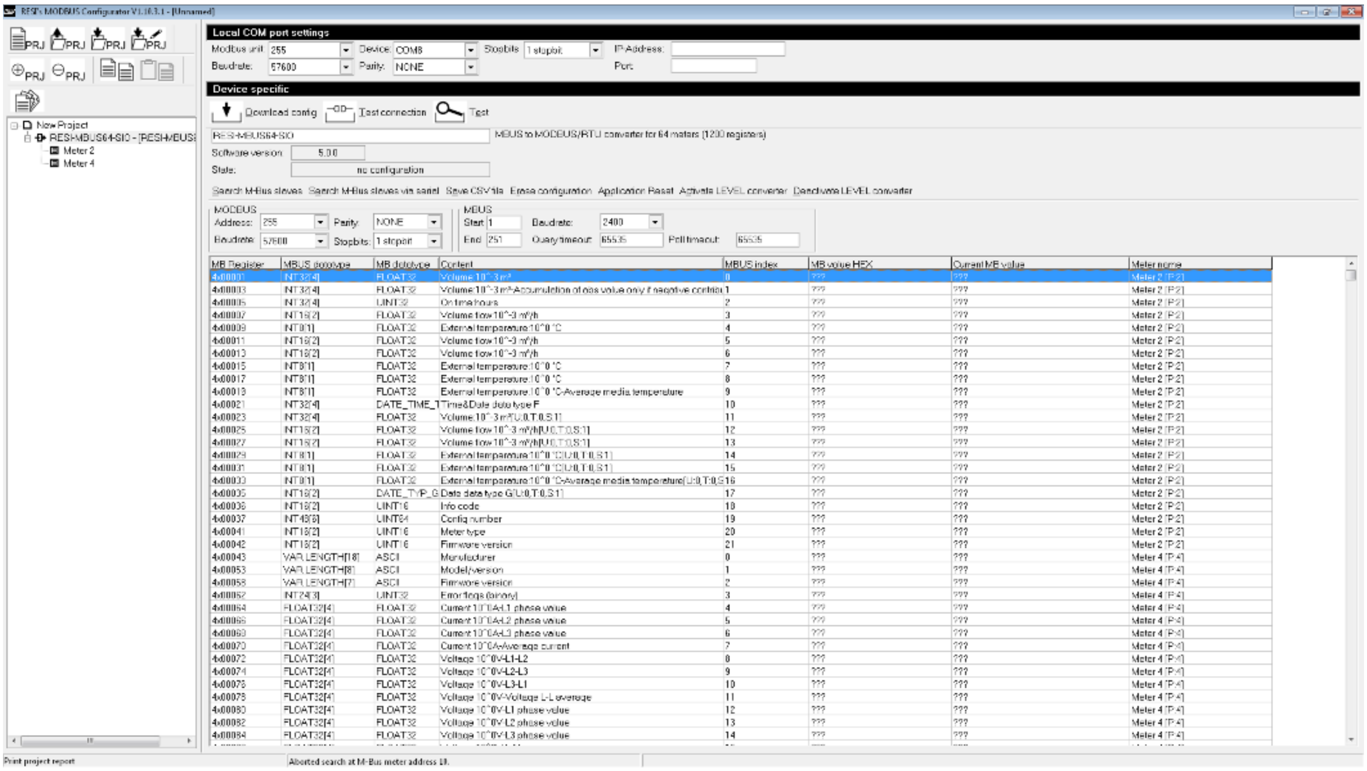
First setup the address range for your search with defining Start end End parameter in the area MBUS. In our example we use the full range 1 to 251:



Click then on the button Search M-Bus slaves to start the automatic search. Be aware, that you will not find a connected meter if it has a different baud rate configured or if it has no primary address programmed or if there are two meters with the same primary address on the bus!

HINT: You can interrupt the automatic search process by pressing the ESC button. After a few seconds the search will be interrupted.

In our test case we have connected two meters to our test system with the primary address 2 and 4. The result will look like this:



You notice, that now two meters are sown in the project tree. One with the number 2 and one with the number 4. Also the software has build an automatic mapping table between the MBUS data points and the MODBUS registers of the meter. This table is shown below the current settings. lets take a closer look into this table:

MB Register	MBUS datatype	MB datatype	Content	MBUS index	MB value HEX	Current MB value	Meter name
4x00001	INT32[4]	FLOAT32	Volume:10^-3 m³	0	???	???	Meter 2 [P:2]
4x00003	INT32[4]	FLOAT32	Volume:10^-3 m³-Accumulation of abs value only if negative contribu	1	???	???	Meter 2 [P:2]
4x00005	INT32[4]	UINT32	On time:hours	2	???	???	Meter 2 [P:2]
4x00007	INT16[2]	FLOAT32	Volume flow:10^-3 m³/h	3	???	???	Meter 2 [P:2]
4x00009	INT8[1]	FLOAT32	External temperature:10^0 °C	4	???	???	Meter 2 [P:2]
4x00011	INT16[2]	FLOAT32	Volume flow:10^-3 m³/h	5	???	???	Meter 2 [P:2]
4x00013	INT16[2]	FLOAT32	Volume flow:10^-3 m³/h	6	???	???	Meter 2 [P:2]
4x00015	INT8[1]	FLOAT32	External temperature:10^0 °C	7	???	???	Meter 2 [P:2]
4x00017	INT8[1]	FLOAT32	External temperature:10^0 °C	8	???	???	Meter 2 [P:2]
4x00019	INT8[1]	FLOAT32	External temperature:10^0 °C-Average media temperature	9	???	???	Meter 2 [P:2]
4x00021	INT32[4]	DATE_TIME_1	Time&Date data type F	10	???	???	Meter 2 [P:2]
4x00023	INT32[4]	FLOAT32	Volume:10^-3 m³[U:0,T:0,S:1]	11	???	???	Meter 2 [P:2]
4x00025	INT16[2]	FLOAT32	Volume flow:10^-3 m³/h[U:0,T:0,S:1]	12	???	???	Meter 2 [P:2]
4x00027	INT16[2]	FLOAT32	Volume flow:10^-3 m³/h[U:0,T:0,S:1]	13	???	???	Meter 2 [P:2]
4x00029	INT8[1]	FLOAT32	External temperature:10^0 °C[U:0,T:0,S:1]	14	???	???	Meter 2 [P:2]
4x00031	INT8[1]	FLOAT32	External temperature:10^0 °C[U:0,T:0,S:1]	15	???	???	Meter 2 [P:2]
4x00033	INT8[1]	FLOAT32	External temperature:10^0 °C-Average media temperature[U:0,T:0,S	16	???	???	Meter 2 [P:2]
4x00035	INT16[2]	DATE_TYP_G	Date data type G[U:0,T:0,S:1]	17	???	???	Meter 2 [P:2]
4x00036	INT16[2]	UINT16	Info code	18	???	???	Meter 2 [P:2]
4x00037	INT48[6]	UINT64	Config number	19	???	???	Meter 2 [P:2]
4x00041	INT16[2]	UINT16	Meter type	20	???	???	Meter 2 [P:2]
4x00042	INT16[2]	UINT16	Firmware version	21	???	???	Meter 2 [P:2]
4x00043	VAR LENGTH[18]	ASCII	Manufacturer	0	???	???	Meter 4 [P:4]
4x00053	VAR LENGTH[8]	ASCII	Model/version	1	???	???	Meter 4 [P:4]
4x00058	VAR LENGTH[7]	ASCII	Firmware version	2	???	???	Meter 4 [P:4]
4x00062	INT24[3]	UINT32	Error flags (binary)	3	???	???	Meter 4 [P:4]
4x00064	FLOAT32[4]	FLOAT32	Current 10^0A-L1 phase value	4	???	???	Meter 4 [P:4]
4x00066	FLOAT32[4]	FLOAT32	Current 10^0A-L2 phase value	5	???	???	Meter 4 [P:4]
4x00068	FLOAT32[4]	FLOAT32	Current 10^0A-L3 phase value	6	???	???	Meter 4 [P:4]
4x00070	FLOAT32[4]	FLOAT32	Current 10^0A-Average current	7	???	???	Meter 4 [P:4]
4x00072	FLOAT32[4]	FLOAT32	Voltage 10^0V-L1-L2	8	???	???	Meter 4 [P:4]
4x00074	FLOAT32[4]	FLOAT32	Voltage 10^0V-L2-L3	9	???	???	Meter 4 [P:4]
4x00076	FLOAT32[4]	FLOAT32	Voltage 10^0V-L3-L1	10	???	???	Meter 4 [P:4]
4x00078	FLOAT32[4]	FLOAT32	Voltage 10^0V-Voltage L-L average	11	???	???	Meter 4 [P:4]
4x00080	FLOAT32[4]	FLOAT32	Voltage 10^0V-L1 phase value	12	???	???	Meter 4 [P:4]
4x00082	FLOAT32[4]	FLOAT32	Voltage 10^0V-L2 phase value	13	???	???	Meter 4 [P:4]
4x00084	FLOAT32[4]	FLOAT32	Voltage 10^0V-L3 phase value	14	???	???	Meter 4 [P:4]

You see, that the first meter is mapped to the MODBUS registers 4x00001 to 4x00041. The second meter is mapped to the MODBUS registers 4x00042 to 4x000232, because the meter offers has much more MBUS data points.

MB Register	MBUS datatype	MB datatype	Content	MBUS index	MB value HEX	Current MB value	Meter name
4x00203	INT24[3]	UINT32	Nominal frequency	61	???	???	Meter 4 [P:4]
4x00205	FLOAT32[4]	FLOAT32	Energy:10^0 Wh	62	???	???	Meter 4 [P:4]
4x00207	FLOAT32[4]	FLOAT32	Energy:10^0 Wh-Export energy value	63	???	???	Meter 4 [P:4]
4x00209	FLOAT32[4]	FLOAT32	Energy:10^0 Wh[U:1,T:0,S:0]	64	???	???	Meter 4 [P:4]
4x00211	FLOAT32[4]	FLOAT32	Energy:10^0 Wh-Export energy value[U:1,T:0,S:0]	65	???	???	Meter 4 [P:4]
4x00213	FLOAT32[4]	FLOAT32	Energy:10^0 Wh-parital energy value	66	???	???	Meter 4 [P:4]
4x00215	FLOAT32[4]	FLOAT32	Energy:10^0 Wh-parital energy value[U:1,T:0,S:0]	67	???	???	Meter 4 [P:4]
4x00217	FLOAT32[4]	FLOAT32	Energy:10^0 Wh-L1 phase value	68	???	???	Meter 4 [P:4]
4x00219	FLOAT32[4]	FLOAT32	Energy:10^0 Wh-L2 phase value	69	???	???	Meter 4 [P:4]
4x00221	FLOAT32[4]	FLOAT32	Energy:10^0 Wh-L3 phase value	70	???	???	Meter 4 [P:4]
4x00223	FLOAT32[4]	FLOAT32	Cumulation counter	71	???	???	Meter 4 [P:4]
4x00225	FLOAT32[4]	FLOAT32	Energy:10^0 Wh[U:0,T:1,S:0]	72	???	???	Meter 4 [P:4]
4x00227	FLOAT32[4]	FLOAT32	Energy:10^0 Wh[U:0,T:2,S:0]	73	???	???	Meter 4 [P:4]
4x00229	FLOAT32[4]	FLOAT32	Energy:10^0 Wh[U:0,T:3,S:0]	74	???	???	Meter 4 [P:4]
4x00231	FLOAT32[4]	FLOAT32	Energy:10^0 Wh[U:0,T:4,S:0]	75	???	???	Meter 4 [P:4]
4x09001	RESI	UINT16	Converter state for meter	STATE	???	???	Meter 2 [P:2]
4x09002	HEADER	UINT32R	Identification number of meter	ID	???	???	Meter 2 [P:2]
4x09004	RESI	UINT16	Converter state for meter	STATE	???	???	Meter 4 [P:4]
4x09005	HEADER	UINT32R	Identification number of meter	ID	???	???	Meter 4 [P:4]
4x10001	HEADER	UINT32	Identification number of meter	ID	???	???	Meter 2 [P:2]
4x10003	HEADER	UINT32->ASCI	Manufacturer of meter	MANUFACTURER	???	???	Meter 2 [P:2]
4x10005	HEADER	UINT16	Version of meter	VERSION	???	???	Meter 2 [P:2]
4x10006	HEADER	UINT16	Medium of meter	MEDIUM	???	???	Meter 2 [P:2]
4x10007	HEADER	UINT16	Access of meter	ACCESS	???	???	Meter 2 [P:2]
4x10008	HEADER	UINT16	Status of meter	STATUS	???	???	Meter 2 [P:2]
4x10009	RESI	UINT16	Future value of meter	FUTURE	???	???	Meter 2 [P:2]
4x10010	RESI	UINT16	Communcation state with meter	COMM STATE	???	???	Meter 2 [P:2]
4x10011	HEADER	UINT32	Identification number of meter	ID	???	???	Meter 4 [P:4]
4x10013	HEADER	UINT32->ASCI	Manufacturer of meter	MANUFACTURER	???	???	Meter 4 [P:4]
4x10015	HEADER	UINT16	Version of meter	VERSION	???	???	Meter 4 [P:4]
4x10016	HEADER	UINT16	Medium of meter	MEDIUM	???	???	Meter 4 [P:4]
4x10017	HEADER	UINT16	Access of meter	ACCESS	???	???	Meter 4 [P:4]
4x10018	HEADER	UINT16	Status of meter	STATUS	???	???	Meter 4 [P:4]
4x10019	RESI	UINT16	Future value of meter	FUTURE	???	???	Meter 4 [P:4]
4x10020	RESI	UINT16	Communcation state with meter	COMM STATE	???	???	Meter 4 [P:4]

You can download the configuration and press the Test button. After a few seconds you will see the table filled with online values from the connected meter.

43.10.2 Status information for every meter

Behind the mapping from the MBUS data points to the MODBUS data points, you will see two areas of status information.

MB Register	MBUS datatype	MB datatype	Content	MBUS index	MB value HEX	Current MB value	Meter name
4x00203	INT24[3]	UINT32	Nominal frequency	61	MSW:0000,0032:LSW	50,0x00000032	Meter 4 [P:4]
4x00205	FLOAT32[4]	FLOAT32	Energy:10^0 Wh	62	MSW:3CCC,CCCD:LSW	0.0250,2.50000003725290E-2	Meter 4 [P:4]
4x00207	FLOAT32[4]	FLOAT32	Energy:10^0 Wh-Export energy value	63	MSW:0000,0000:LSW	0.0000,0.00000000000000E+0	Meter 4 [P:4]
4x00209	FLOAT32[4]	FLOAT32	Energy:10^0 Wh[U:1,T:0,S:0]	64	MSW:3C75,C28F:LSW	0.0150,1.49999996647239E-2	Meter 4 [P:4]
4x00211	FLOAT32[4]	FLOAT32	Energy:10^0 Wh-Export energy value[U:1,T:0,S:0]	65	MSW:0000,0000:LSW	0.0000,0.00000000000000E+0	Meter 4 [P:4]
4x00213	FLOAT32[4]	FLOAT32	Energy:10^0 Wh-parital energy value	66	MSW:3CCC,CCCD:LSW	0.0250,2.50000003725290E-2	Meter 4 [P:4]
4x00215	FLOAT32[4]	FLOAT32	Energy:10^0 Wh-parital energy value[U:1,T:0,S:0]	67	MSW:3C75,C28F:LSW	0.0150,1.49999996647239E-2	Meter 4 [P:4]
4x00217	FLOAT32[4]	FLOAT32	Energy:10^0 Wh-L1 phase value	68	MSW:3CCC,CCCD:LSW	0.0250,2.50000003725290E-2	Meter 4 [P:4]
4x00219	FLOAT32[4]	FLOAT32	Energy:10^0 Wh-L2 phase value	69	MSW:0000,0000:LSW	0.0000,0.00000000000000E+0	Meter 4 [P:4]
4x00221	FLOAT32[4]	FLOAT32	Energy:10^0 Wh-L3 phase value	70	MSW:0000,0000:LSW	0.0000,0.00000000000000E+0	Meter 4 [P:4]
4x00223	FLOAT32[4]	FLOAT32	Cumulation counter	71	MSW:0000,0000:LSW	0.0000,0.00000000000000E+0	Meter 4 [P:4]
4x00225	FLOAT32[4]	FLOAT32	Energy:10^0 Wh[U:0,T:1,S:0]	72	MSW:0000,0000:LSW	0.0000,0.00000000000000E+0	Meter 4 [P:4]
4x00227	FLOAT32[4]	FLOAT32	Energy:10^0 Wh[U:0,T:2,S:0]	73	MSW:0000,0000:LSW	0.0000,0.00000000000000E+0	Meter 4 [P:4]
4x00229	FLOAT32[4]	FLOAT32	Energy:10^0 Wh[U:0,T:3,S:0]	74	MSW:0000,0000:LSW	0.0000,0.00000000000000E+0	Meter 4 [P:4]
4x00231	FLOAT32[4]	FLOAT32	Energy:10^0 Wh[U:0,T:4,S:0]	75	MSW:0000,0000:LSW	0.0000,0.00000000000000E+0	Meter 4 [P:4]
4x09001	RESI	UINT16	Converter state for meter	STATE	WORD:0003	3,0x0003 -> Values are valid!	Meter 2 [P:2]
4x09002	HEADER	UINT32R	Identification number of meter	ID	LSW:6229,MSW:2071	544301609,0x20716229	Meter 2 [P:2]
4x09004	RESI	UINT16	Converter state for meter	STATE	WORD:0003	3,0x0003 -> Values are valid!	Meter 4 [P:4]
4x09005	HEADER	UINT32R	Identification number of meter	ID	LSW:4163,MSW:0636	104218979,0x06364163	Meter 4 [P:4]
4x10001	HEADER	UINT32	Identification number of meter	ID	MSW:2071,6229:LSW	544301609,0x20716229	Meter 2 [P:2]
4x10003	HEADER	UINT32->ASCI	Manufacturer of meter	MANUFACTURER	MSW:004D,414B:LSW	KAM	Meter 2 [P:2]
4x10005	HEADER	UINT16	Version of meter	VERSION	WORD:001D	29,0x001D	Meter 2 [P:2]
4x10006	HEADER	UINT16	Medium of meter	MEDIUM	WORD:0016	22,0x0016 -> Cold Water	Meter 2 [P:2]
4x10007	HEADER	UINT16	Access of meter	ACCESS	WORD:0072	114,0x0072	Meter 2 [P:2]
4x10008	HEADER	UINT16	Status of meter	STATUS	WORD:0000	0,0x0000	Meter 2 [P:2]
4x10009	RESI	UINT16	Future value of meter	FUTURE	WORD:0000	0,0x0000	Meter 2 [P:2]
4x10010	RESI	UINT16	Communcation state with meter	COMM STATE	WORD:0003	3,0x0003 -> Values are valid!	Meter 2 [P:2]
4x10011	HEADER	UINT32	Identification number of meter	ID	MSW:0636,4163:LSW	104218979,0x06364163	Meter 4 [P:4]
4x10013	HEADER	UINT32->ASCI	Manufacturer of meter	MANUFACTURER	MSW:0043,4553:LSW	SEC	Meter 4 [P:4]
4x10015	HEADER	UINT16	Version of meter	VERSION	WORD:0018	24,0x0018	Meter 4 [P:4]
4x10016	HEADER	UINT16	Medium of meter	MEDIUM	WORD:0002	2,0x0002 -> Electricity	Meter 4 [P:4]
4x10017	HEADER	UINT16	Access of meter	ACCESS	WORD:0014	20,0x0014	Meter 4 [P:4]
4x10018	HEADER	UINT16	Status of meter	STATUS	WORD:0000	0,0x0000	Meter 4 [P:4]
4x10019	RESI	UINT16	Future value of meter	FUTURE	WORD:0000	0,0x0000	Meter 4 [P:4]
4x10020	RESI	UINT16	Communcation state with meter	COMM STATE	WORD:0003	3,0x0003 -> Values are valid!	Meter 4 [P:4]

Area 1 is compatible to our old MBUS converter modules, but it is located in a different area of the MBUS registers starting at 4x09001. For every configured meter two MODBUS entries are generated. One holds the communication state of the MBUS gateway with the meter with the following states:

- **0 - Meter isn't configured!**: This value shows, that this meter slot is currently not configured in the MBUS gateway.
- **1 - Meter isn't normalized!**: This value shows, that the configured meter doesn't answer to the addressing command. Either via primary addressing or via secondary addressing mode. This depends, how the meter was configured.
- **2 - Meter isn't read!**: This value shows, that the configured meter has answered to the addressing command but there are problems by reading all data from the meter. So the meter data is not valid any more.

- **3 - Values are valid!**: This value shows, that the configured meter has answered to the addressing command and has answered correctly to the readout commands and the reading of all data from the meter was successful. So the meter data in the MODBUS register is valid.

The other entry holds the serial number of the configured meter in two consecutive holding registers.

Area 2 is new to the new series of gateways and represent the information of the MBUS fixed data header.

Ident. Nr.	Manufr.	Version	Medium	Access No.	Status	Signature
4 Byte	2 Byte	1 Byte	1 Byte	1 Byte	1 Byte	2 Byte

This header is sent by many answer frames of the MBUS meter to the master. Due to the fact, that is is not part of the variable data block of the meter, our old converters could not map this information to registers. Our new series map this information to the following register set starting at 4x10001. For each meter there are eight MODBUS entires:

ENTRY 1: Identification number of the meter
Register <METERBASE>+0, <METERBASE>+1
Each meter offers a unique ID. In the MBUS protocol there are four bytes reserved for this number. In our gateway we need a UINT32 to represent this 4 bytes of the ID.

ENTRY 2: Manufacturer of the meter
Register <METERBASE>+2, <METERBASE>+3
Each meter offers a manufacturer ID, represented in two bytes. But in this two bytes there are three ASCII digits encoded. Our gateway decode this ASCII digits and stores this digits into a UINT32 using ASCII encoding with 0x00 at the end representing a standard null terminated ASCII string of three letters.

ENTRY 3: Version of the meter
Register <METERBASE>+4
In this fixed data header, there is also a version number encoded into one byte. It represents the version of the meter. Our gateway stores this byte into a UINT16 holding register for easy readout.

ENTRY 4: Medium of the meter
Register <METERBASE>+5
In this fixed data header, there is also a medium number encoded into one byte. it defines what type of medium the meter is measuring. Our gateway stores this byte into a UINT16 holding register for easy readout.

The following medium types are defined by the standard for meters with fixed+variable data structure:

- 0x00: OTHER
- 0x01: OIL
- 0x02: Electricity
- 0x03: Gas
- 0x04: Heat-Volume measured at return temperature outlet
- 0x05: Steam
- 0x06: Hot Water
- 0x07: Water
- 0x08: H.C.A.=Heat Cost Allocator
- 0x09: Compressed Air
- 0x0A: Cooling load meter Volume measured at return temperature outlet
- 0x0B: Cooling load meter Volume measured at flow temperature inlet
- 0x0C: Heat Volume measured at flow temperature inlet
- 0x0D: Heat/Cooling load meter
- 0x0E: Bus/System
- 0x0F: Unknown Medium
- 0x16: Cold Water
- 0x17: Dual Water
- 0x18: Pressure
- 0x19: A/D Converter

For meters with fixed data structure only, the 16 bit value must be interpreted in another way. Refer to the MBUS standard for this definition.

ENTRY 5: Access counter of the meter
Register <METERBASE>+6
In this fixed data header, there is also an access counter encoded into one byte. It will be incremented by every access of the meter data. So each readout of the meter will increment this access counter by 1 in the range from 0 to 255. Our gateway stores this byte into a UINT16 holding register for easy readout.

ENTRY 6: Status of the meter
Register <METERBASE>+7
In this fixed data header, there is also a status field encoded into one byte. It shows the current meter status. Our gateway stores this byte into a UINT16 holding register for easy readout.

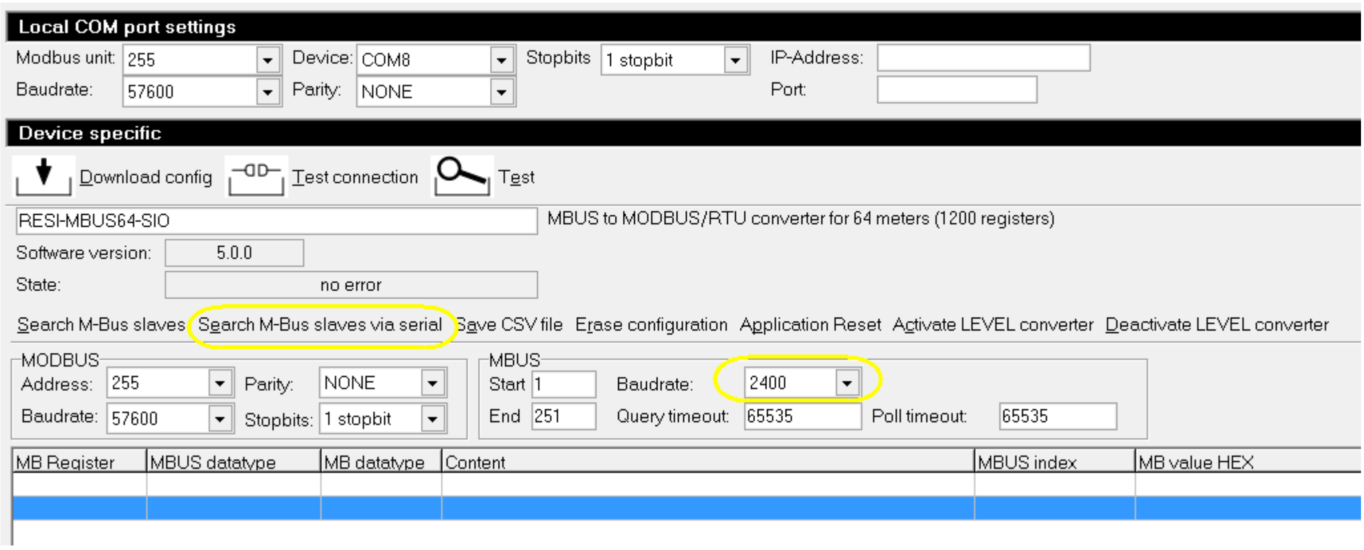
The byte has the following meaning:
- Bit 1+Bit 0: =00 (0) NO ERROR
- Bit 1+Bit 0: =10 (1) APPLICATION NOT READY
- Bit 1+Bit 0: =01 (2) APPLICATION ERROR
- Bit 1+Bit 0: =11 (3) RESERVED
- Bit 2: =1: POWER LOW, =0: POWER OK
- Bit 3: =1: PERMANENT ERROR, =0: NO PERMANENT ERROR
- Bit 4: =1: TEMPORARY ERROR, =0: NO TEMPORARY ERROR
- Bit 5: =1: MANUFACTURER SPECIFIC ERROR 1, =0: NO MANUFACTURER SPECIFIC ERROR 1
- Bit 6: =1: MANUFACTURER SPECIFIC ERROR 2, =0: NO MANUFACTURER SPECIFIC ERROR 2
- Bit 7: =1: MANUFACTURER SPECIFIC ERROR 3, =0: NO MANUFACTURER SPECIFIC ERROR 3

ENTRY 7: Future value of the meter
Register <METERBASE>+8
This UINT16 holding register is reserved for future use.

ENTRY 8: Communication state with meter
Register <METERBASE>+9
This UINT16 holding register hold the current state of the communication between the MBUS gateway and the meter with the following states:
- **0 - Meter isn't configured!**: This value shows, that this meter slot is currently not configured in the MBUS gateway.
- **1 - Meter isn't normalized!**: This value shows, that the configured meter doesn't answer to the addressing command. Either via primary addressing or via secondary addressing mode. This depends, how the meter was configured.
- **2 - Meter isn't read!**: This value shows, that the configured meter has answered to the addressing command but there are problems by reading all data from the meter. So the meter data is not valid any more.
- **3 - Values are valid!**: This value shows, that the configured meter has answered to the addressing command and has answered correctly to the readout commands and the reading of all data from the meter was successful. So the meter data in the MODBUS register is valid.

43.10.3 Search for new meters – secondary addressing mode

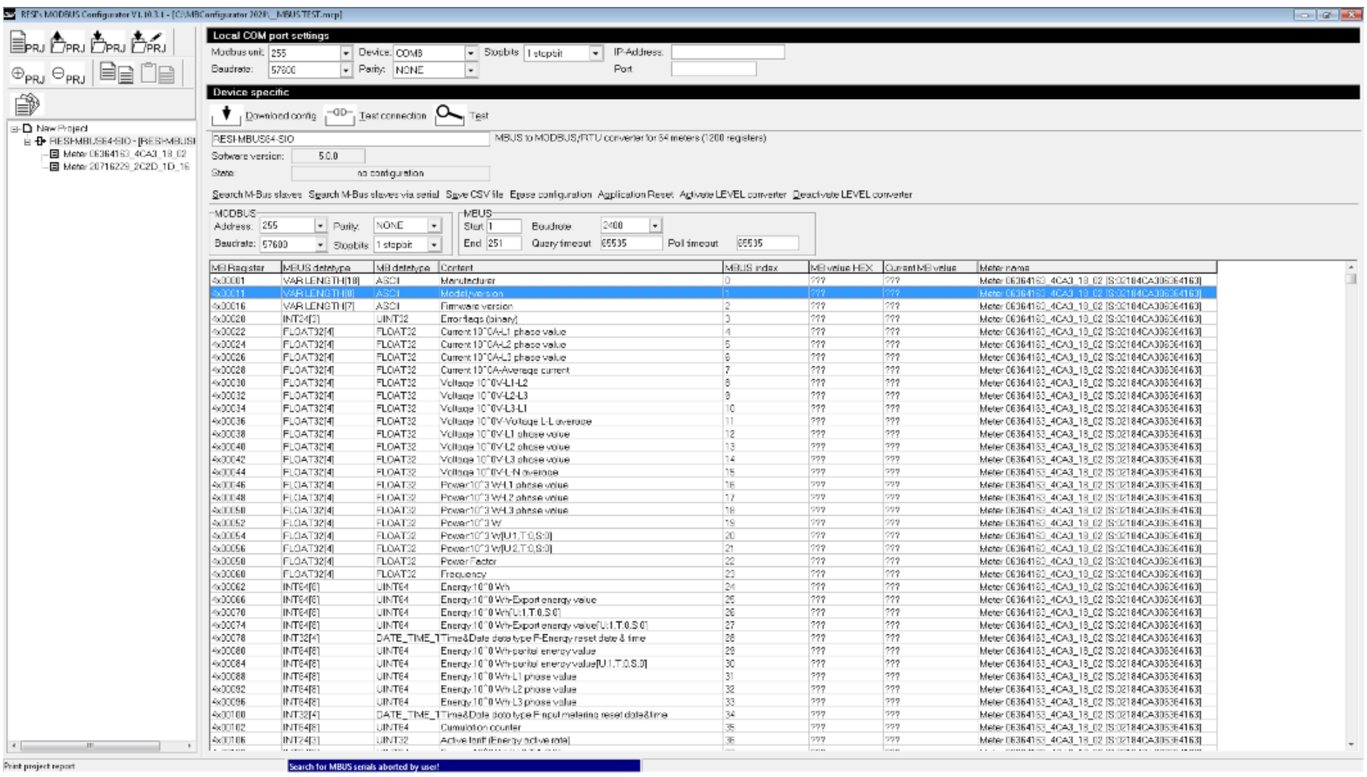
HINT: Don't forget to setup the MBUS bus baud rate for your search before. If you have to change it, select a new one from the drop down list and don't forget to download the bus speed into your gateway!



Click on the button Search M-Bus slaves via serial to start the automatic search with secondary addressing mode for the meters. Now the software does a pattern search for new connected MBUS meters. The benefit with this method is, that you don't have to program any primary addresses into your meters.

HINT: You can interrupt the automatic search process by pressing the ESC button. After a few seconds the search will be interrupted.

In our test case the screen will look like this:

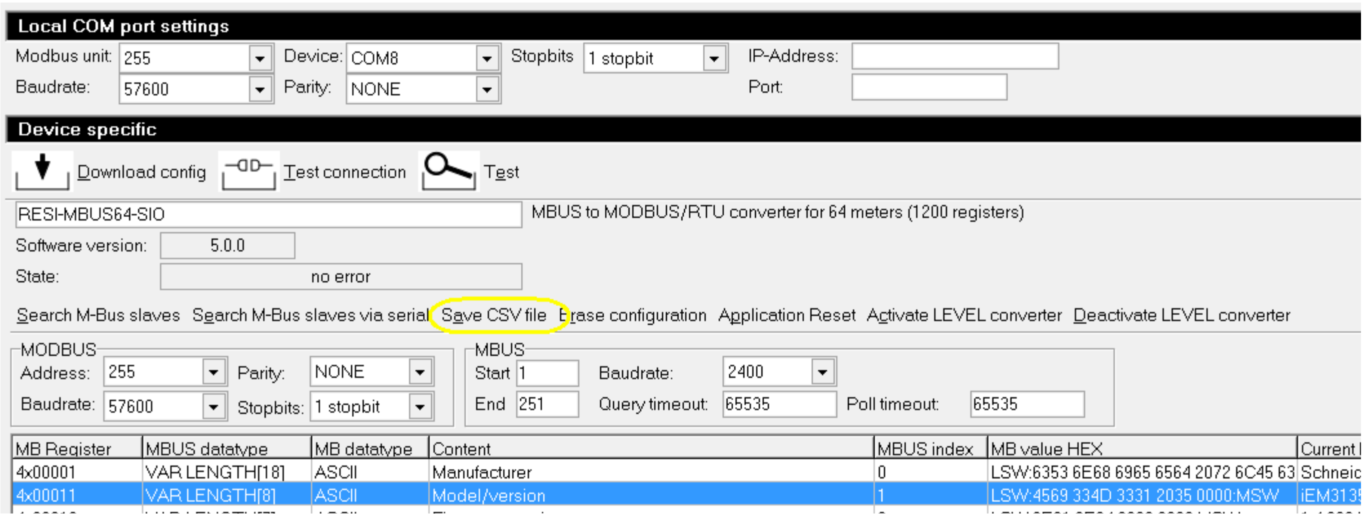


Again you can now download the configuration and start a quick test by activating the test mode with the button Test. Now you will see values in the data grid after a few seconds:

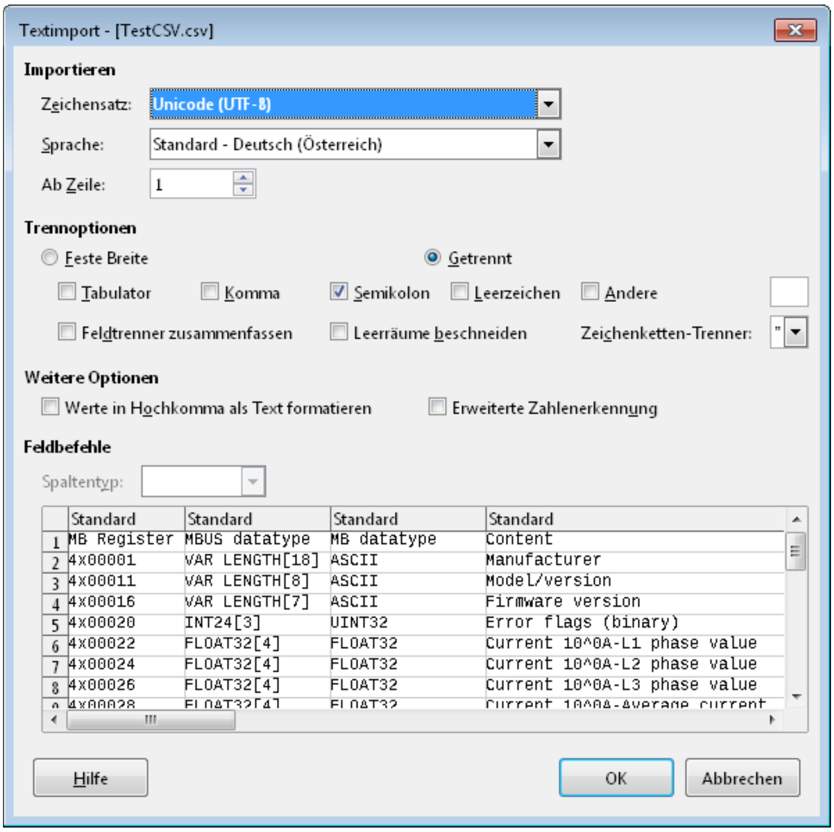
MB Register	MBUS datatype	MB datatype	Content	MBUS index	MB value HEX	Current MB value	Meter name
4x00001	VAR LENGTH[18]	ASCII	Manufacturer	0	LSW:6353 6E68 6965 6564 2072 6C45 63	Schneider Electric,LSB:53 63 68 6E 65 69 64 65 72 20 45 6C 65 63 74 72 69 63 00:MSB	Meter 06364163_
4x00011	VAR LENGTH[8]	ASCII	Model/version	1	LSW:4569 334D 3331 2035 0000:MSW	iEM3135 ,LSB:69 45 4D 33 31 33 35 20 00:MSB	Meter 06364163_
4x00016	VAR LENGTH[7]	ASCII	Firmware version	2	LSW:2E31 2E34 3030 0032:MSW	1.4.002,LSB:31 2E 34 2E 30 30 32 00:MSB	Meter 06364163_
4x00020	INT24[3]	UINT32	Error flags (binary)	3	MSW:0000,0000:LSW	0,0x00000000	Meter 06364163_
4x00022	FLOAT32[4]	FLOAT32	Current 10^0A-L1 phase value	4	MSW:0000,0000:LSW	0.0000,0.00000000000000E+0	Meter 06364163_
4x00024	FLOAT32[4]	FLOAT32	Current 10^0A-L2 phase value	5	MSW:FFC0,0000:LSW	NAN,NAN	Meter 06364163_
4x00026	FLOAT32[4]	FLOAT32	Current 10^0A-L3 phase value	6	MSW:FFC0,0000:LSW	NAN,NAN	Meter 06364163_
4x00028	FLOAT32[4]	FLOAT32	Current 10^0A-Average current	7	MSW:0000,0000:LSW	0.0000,0.00000000000000E+0	Meter 06364163_
4x00030	FLOAT32[4]	FLOAT32	Voltage 10^0V-L1-L2	8	MSW:FFC0,0000:LSW	NAN,NAN	Meter 06364163_
4x00032	FLOAT32[4]	FLOAT32	Voltage 10^0V-L2-L3	9	MSW:FFC0,0000:LSW	NAN,NAN	Meter 06364163_
4x00034	FLOAT32[4]	FLOAT32	Voltage 10^0V-L3-L1	10	MSW:FFC0,0000:LSW	NAN,NAN	Meter 06364163_
4x00036	FLOAT32[4]	FLOAT32	Voltage 10^0V-Voltage L-L average	11	MSW:FFC0,0000:LSW	NAN,NAN	Meter 06364163_
4x00038	FLOAT32[4]	FLOAT32	Voltage 10^0V-L1 phase value	12	MSW:4366,04AC:LSW	230.0182,2.30018249511719E+2	Meter 06364163_
4x00040	FLOAT32[4]	FLOAT32	Voltage 10^0V-L2 phase value	13	MSW:FFC0,0000:LSW	NAN,NAN	Meter 06364163_
4x00042	FLOAT32[4]	FLOAT32	Voltage 10^0V-L3 phase value	14	MSW:FFC0,0000:LSW	NAN,NAN	Meter 06364163_
4x00044	FLOAT32[4]	FLOAT32	Voltage 10^0V-L-N average	15	MSW:4366,04AC:LSW	230.0182,2.30018249511719E+2	Meter 06364163_
4x00046	FLOAT32[4]	FLOAT32	Power:10^3 W-L1 phase value	16	MSW:0000,0000:LSW	0.0000,0.00000000000000E+0	Meter 06364163_
4x00048	FLOAT32[4]	FLOAT32	Power:10^3 W-L2 phase value	17	MSW:FFC0,0000:LSW	NAN,NAN	Meter 06364163_
4x00050	FLOAT32[4]	FLOAT32	Power:10^3 W-L3 phase value	18	MSW:FFC0,0000:LSW	NAN,NAN	Meter 06364163_
4x00052	FLOAT32[4]	FLOAT32	Power:10^3 W	19	MSW:0000,0000:LSW	0.0000,0.00000000000000E+0	Meter 06364163_
4x00054	FLOAT32[4]	FLOAT32	Power:10^3 W[U:1,T:0,S:0]	20	MSW:0000,0000:LSW	0.0000,0.00000000000000E+0	Meter 06
4x00056	FLOAT32[4]	FLOAT32	Power:10^3 W[U:2,T:0,S:0]	21	MSW:0000,0000:LSW	0.0000,0.00000000000000E+0	Meter 06
4x00058	FLOAT32[4]	FLOAT32	Power Factor	22	MSW:FFC0,0000:LSW	NAN,NAN	Meter 06
4x00060	FLOAT32[4]	FLOAT32	Frequency	23	MSW:4247,FDFD:LSW	49.9980,4.99980354309082E+1	Meter 06364163_
4x00062	INT64[8]	UINT64	Energy:10^0 Wh	24	MSW:000000000000,0019:LSW	25,0x00000019	Meter 06364163_
4x00066	INT64[8]	UINT64	Energy:10^0 Wh-Export energy value	25	MSW:000000000000,0000:LSW	0,0x00000000	Meter 06364163_
4x00070	INT64[8]	UINT64	Energy:10^0 Wh[U:1,T:0,S:0]	26	MSW:000000000000,000F:LSW	15,0x0000000F	Meter 06364163_
4x00074	INT64[8]	UINT64	Energy:10^0 Wh-Export energy value[U:1,T:0,S:0]	27	MSW:000000000000,0000:LSW	0,0x00000000	Meter 06364163_
4x00078	INT32[4]	DATE_TIME_1	Time&Date data type F-Energy reset date & time	28	MSW:0101,2080:LSW	00:00 D.M.Y:01.01.00 ST:0 IV:1,0x01012080	Meter 06364163_
4x00080	INT64[8]	UINT64	Energy:10^0 Wh-parital energy value	29	MSW:000000000000,0019:LSW	25,0x00000019	Meter 06364163_
4x00084	INT64[8]	UINT64	Energy:10^0 Wh-parital energy value[U:1,T:0,S:0]	30	MSW:000000000000,000F:LSW	15,0x0000000F	Meter 06364163_
4x00088	INT64[8]	UINT64	Energy:10^0 Wh-L1 phase value	31	MSW:000000000000,0019:LSW	25,0x00000019	Meter 06364163_
4x00092	INT64[8]	UINT64	Energy:10^0 Wh-L2 phase value	32	MSW:000000000000,0000:LSW	0,0x00000000	Meter 06364163_
4x00096	INT64[8]	UINT64	Energy:10^0 Wh-L3 phase value	33	MSW:000000000000,0000:LSW	0,0x00000000	Meter 06364163_
4x00100	INT32[4]	DATE_TIME_1	Time&Date data type F-nput metering reset date&time	34	MSW:0101,2080:LSW	00:00 D.M.Y:01.01.00 ST:0 IV:1,0x01012080	Meter 06364163_
4x00102	INT64[8]	UINT64	Cumulation counter	35	MSW:000000000000,0000:LSW	0,0x00000000	Meter 06364163_

43.10.4 Save to CSV file

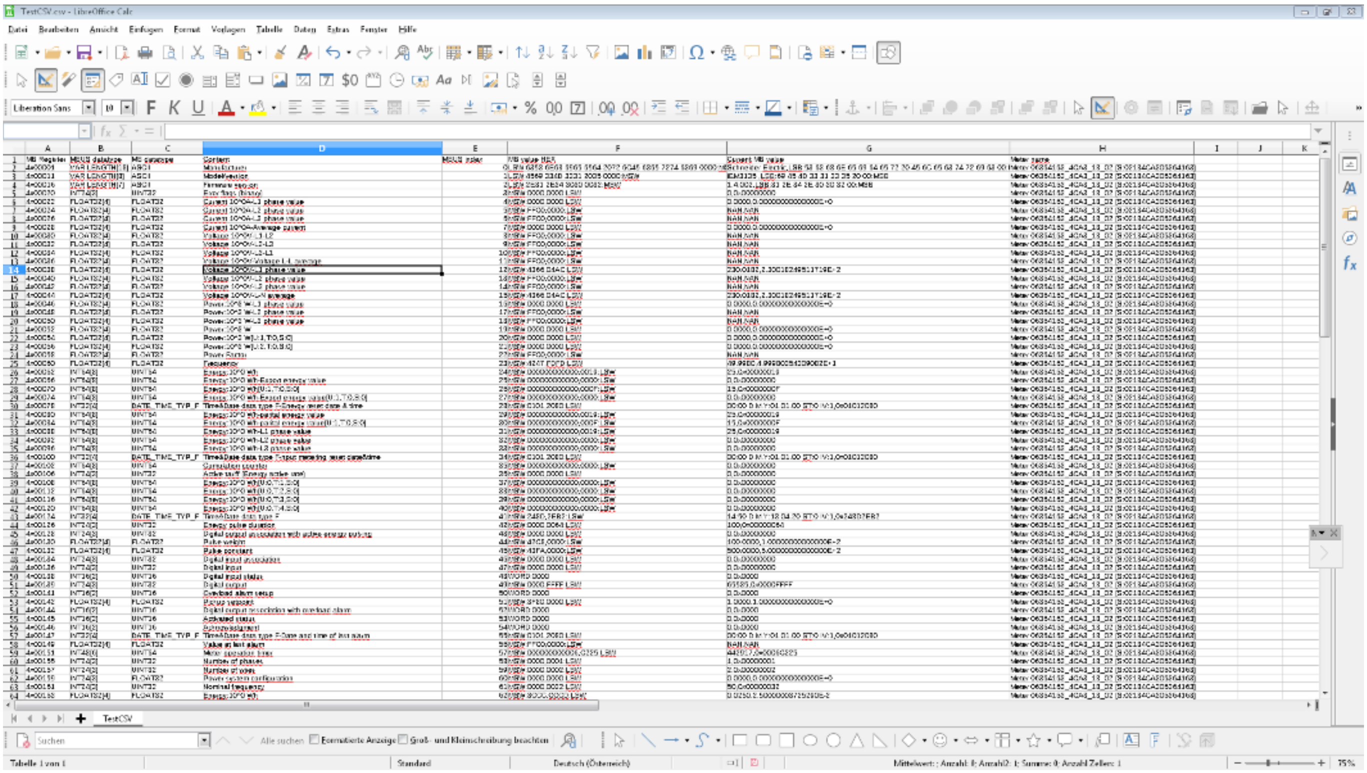
With the action Save to CSV file you can store the current data of the data grind into a CSV file for processing in Libre Office® or Microsoft Office® calculation software.



Click on the button Save CSV file. A dialog for entering the name of the CSV file will be opened. After you defined the name, the CSV file is on your file system. Take a calculation software to open the CSV file (in our case Libre office), select Semicolon as a separator and open the CSV file.

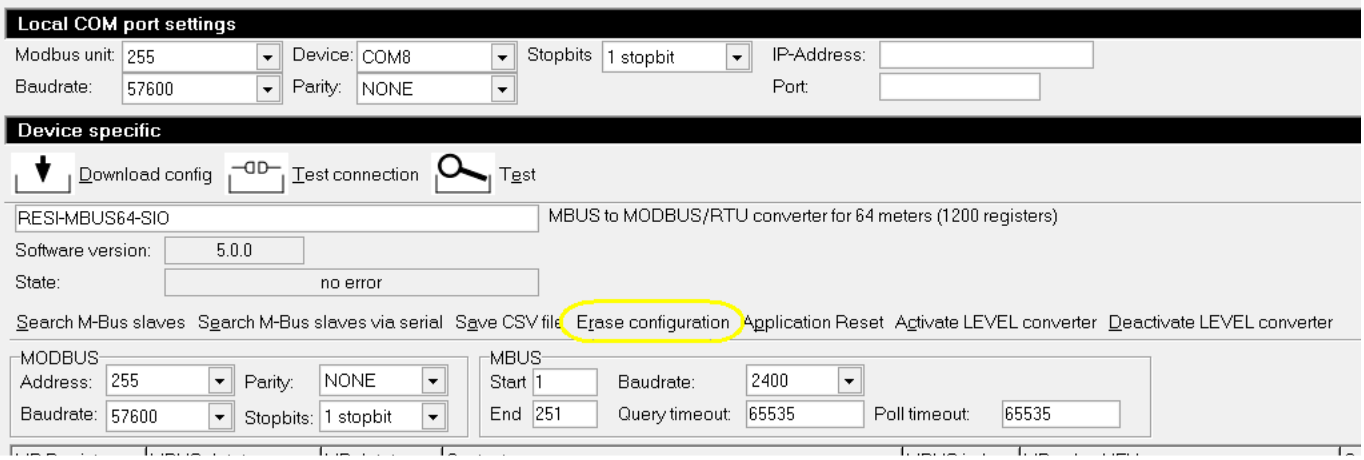


You will see the complete data grid in your calc software for your own purposes:



43.10.5 Erase configuration

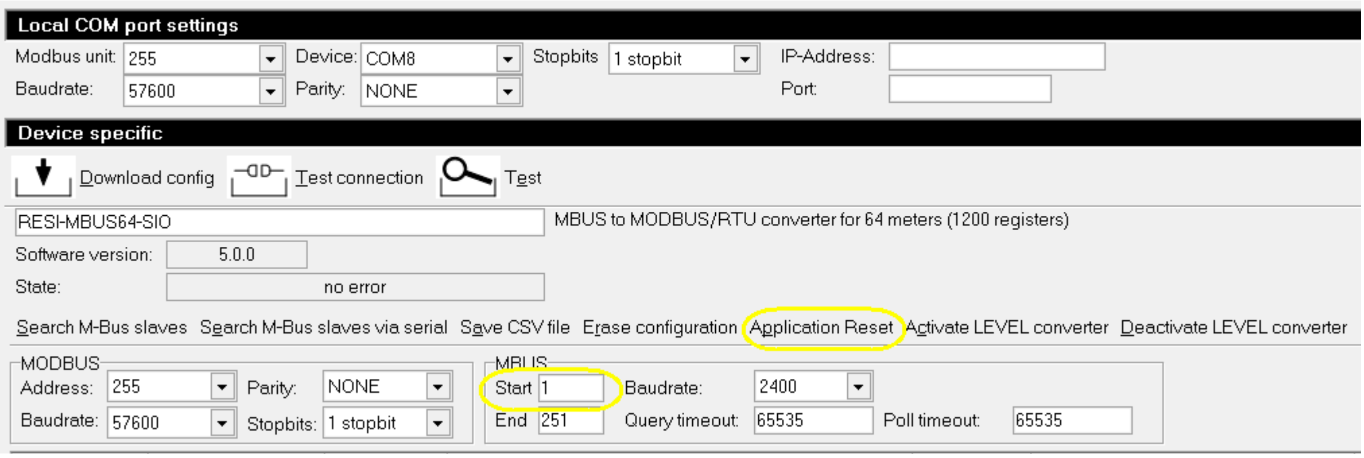
With the action Erase configuration you can delete the complete configuration of the gateway and restore factory settings for all parameters.



Click on the button Erase configuration. A question will pop up. If you answer with YES, the gateway will be restored to factory defaults and the meter configuration will be erased.

43.10.6 Application reset

With the action Application reset you can send the special MBUS command "Application reset" to a defined MBUS meter.



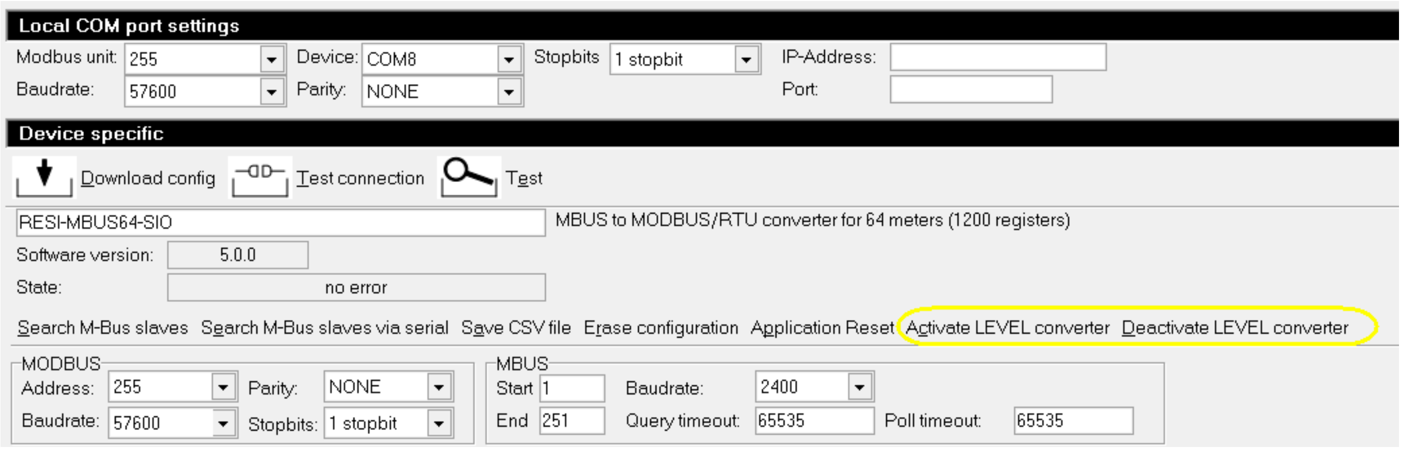
Select the desired primary address for this action with the filed Start in the MBUS area. Then click on the button Application reset. A question will pop up. If you answer with YES, the gateway will send the special MBUS command Application reset to the selected meter.

This is helpful, because some of the meters have trouble to resynchronize to the start of data readout when do a lot of connection /disconnection or other electrical stuff on the MBUS line. There it helps to send this command before trying to search for the connected meter.

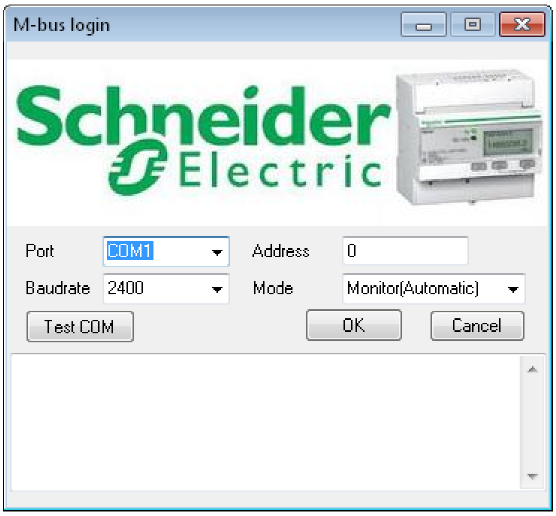
43.10.7 Activate/Deactivate LEVEL converter

With the two actions Activate/deactivate LEVEL converter you cans witch the MBUS gateway to a transparent mode, where every incoming MBUS data is directly send to the host and every incoming characters from the host are send to the MBUS line directly. Also a baud rate conversion will be done. The serial line will use the settings for the serial interface and the MBUS line will use the settings for the MBUS interface.

The integrated LEVEL converter is designed to configure meters with individual software form manufacturers over a standard level converter. Usually you have to have another MBUS level converter module either from RESI or from other suppliers like RELAY ® to configure your meters. Now you can do this over our gateway.

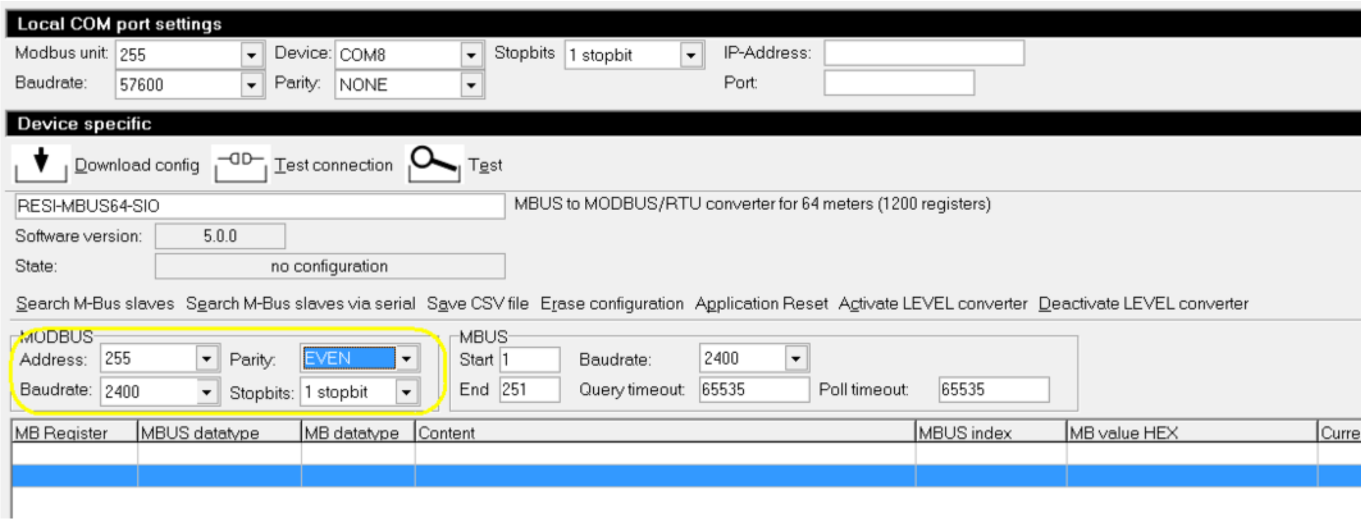


In our test szenario, we want to connect to a Schneider Electric meter with the original Schneider Electric configuration software. So when we start the software, we get the following screen:

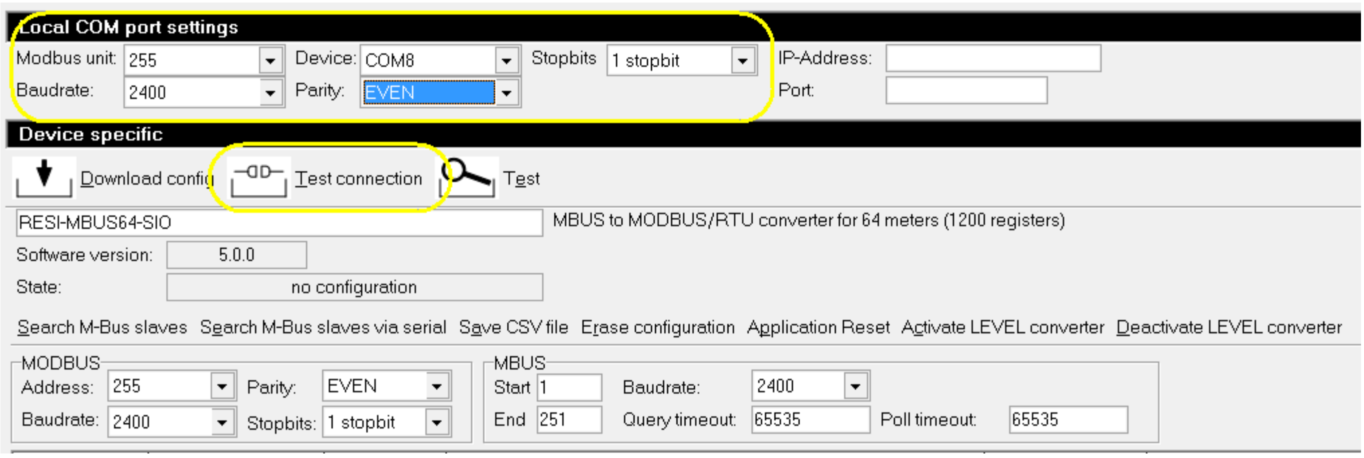


So first of all we have to change the speed settings for our gateway to parameters which are suitable to most of the MBUS tools on the market. Since the MBUS standard defines 2400bd, even parity and one stop bit as common on the MBUS side and many MBUS gateways are simple electrical converters, the tools assume a gateway with 2400bd, EVEN parity and one stop bit.

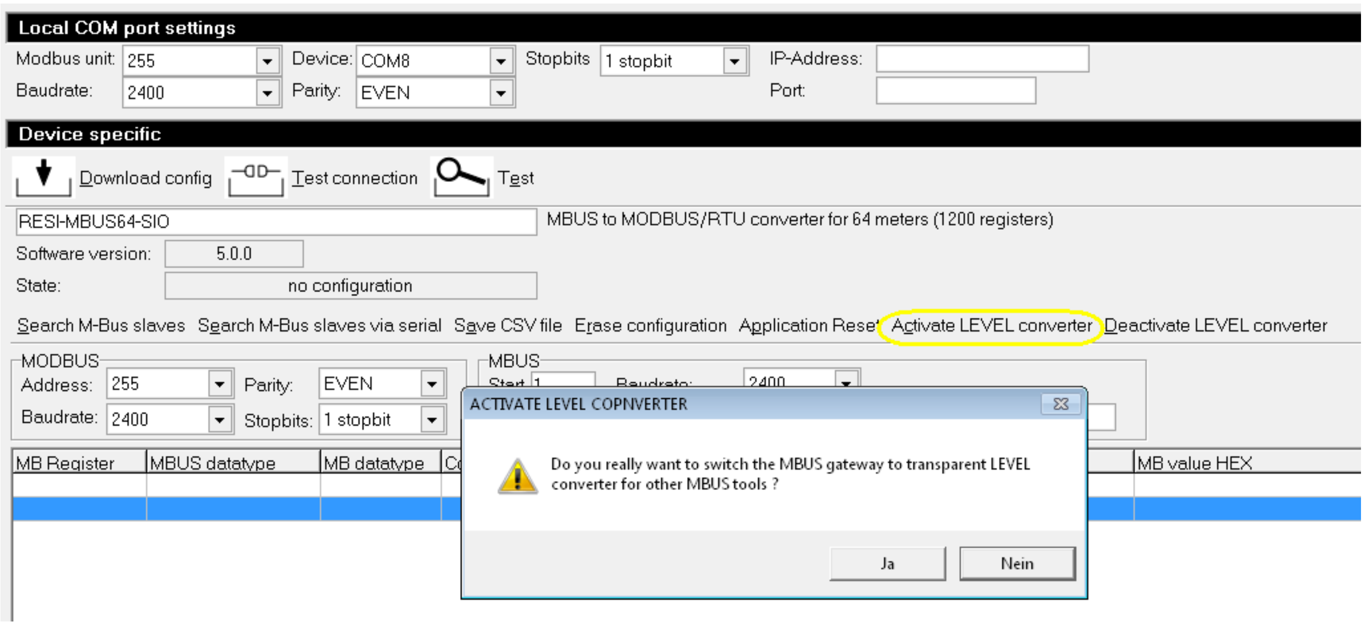
Select 2400bd, even parity and 1 stopbit in the area MODBUS and download this configuration with the button Download config.



The adopt your local COM settings for this new settings in the converter. Check the connection with the button Test connection.

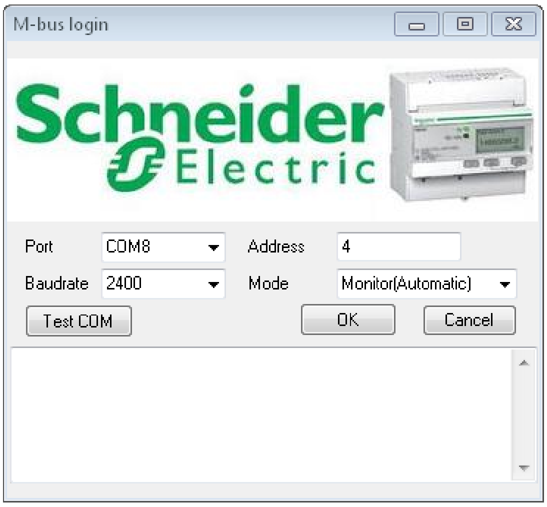


After that activate the integrated LEVEL converter by pressing the button Activate LEVEL converter.

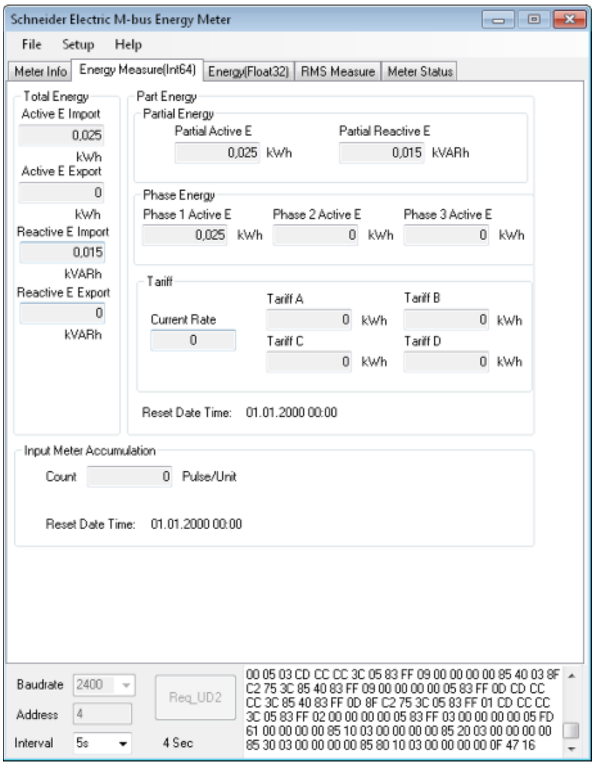


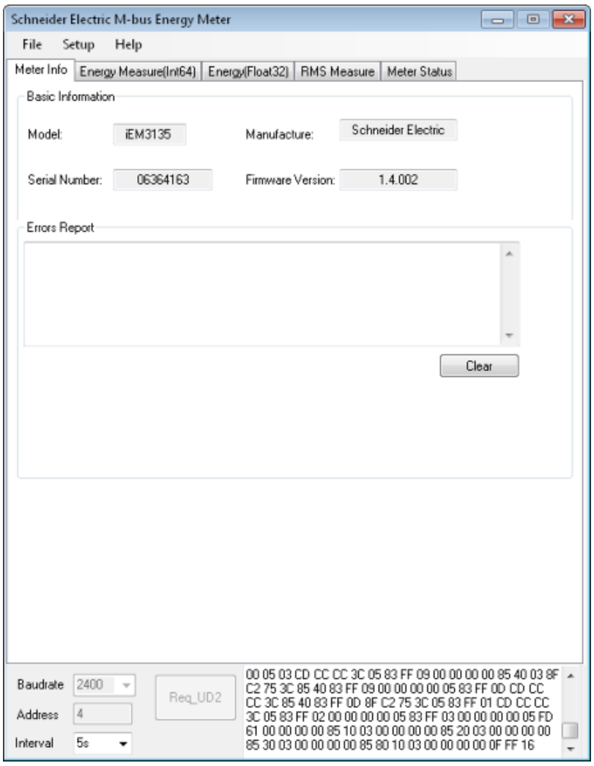
if everything is ok, the converter LED state will flash extraordinary fast to show, that the LEVEL converter is active.

Now we activate the SCHNEIDER software by selecting the correct COM port and the correct primary address and the desired mode:

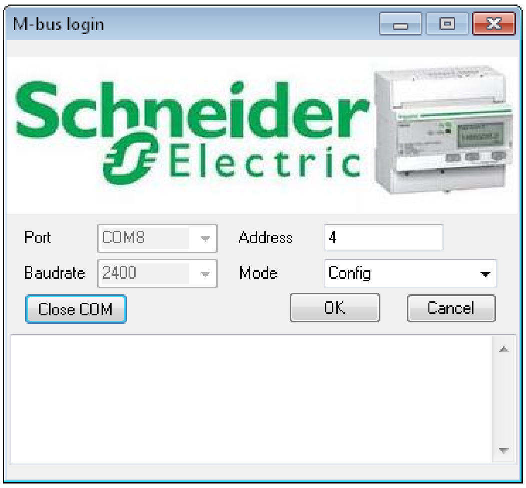


Press the button Test COM first, then press the OK button. The software will now scan automatically the meter and show the result on the screen:

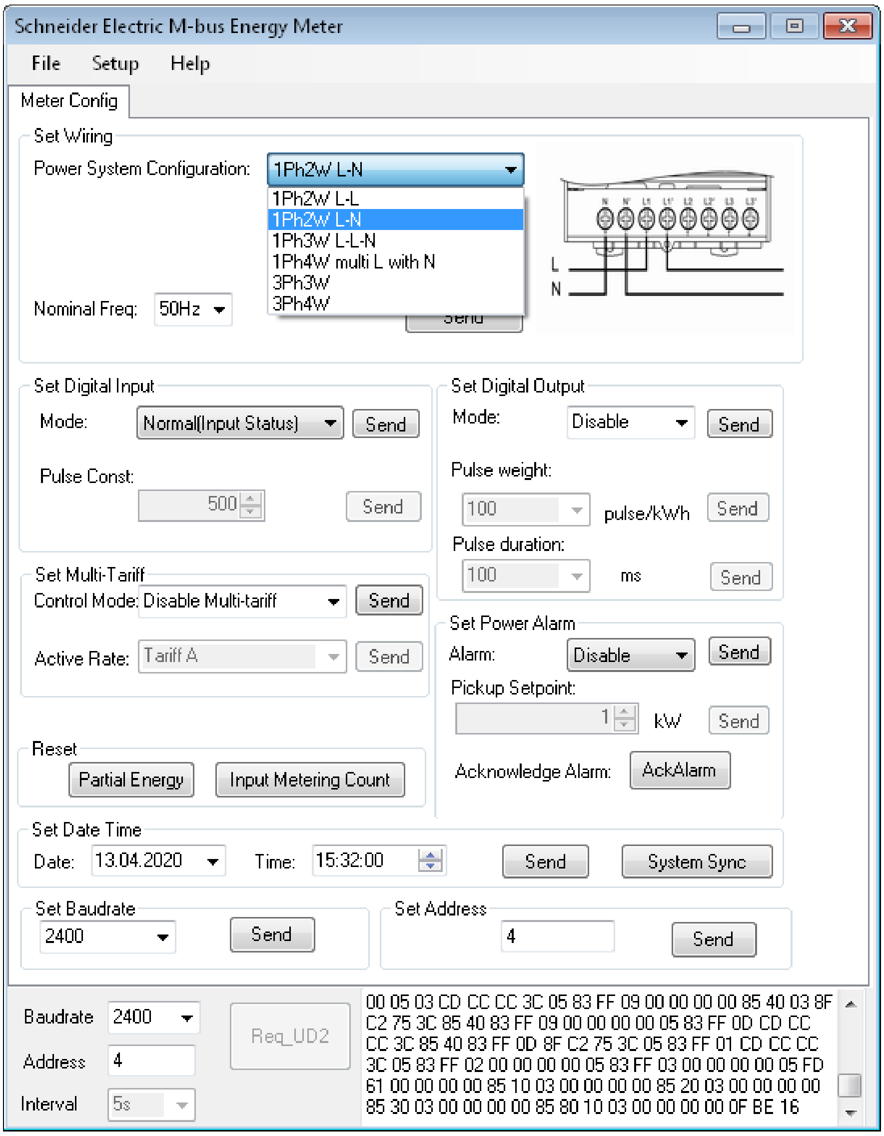




If you start the MBUS configuration software in mode CONFIG



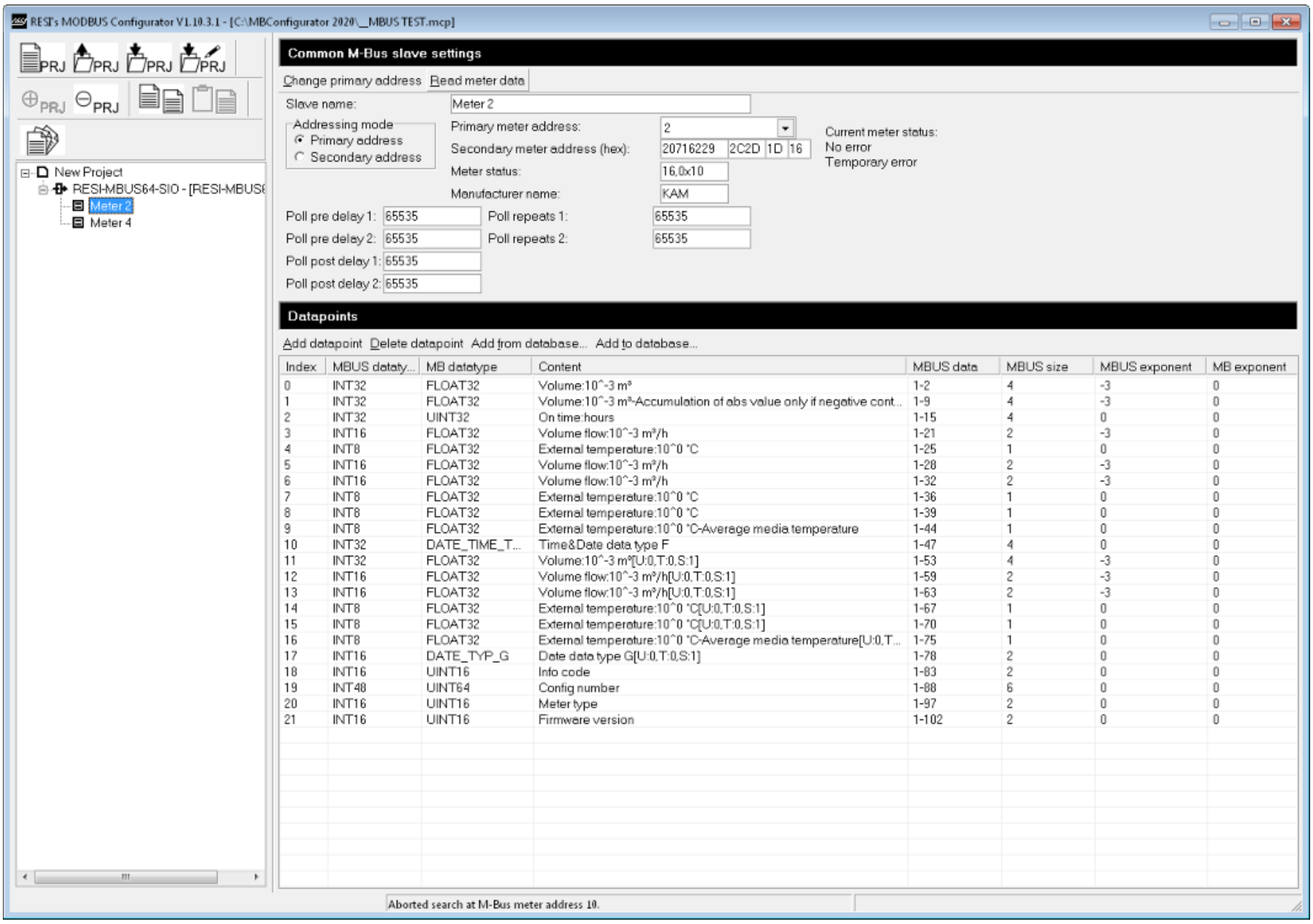
Click again first on Test COM button then on OK button. You will get the following screen:



After you have exited the SCHNEIDER software you can deactivate the LEVEL function either by disconnecting/reconnecting the power supply from the gateway (hard reset) or by pressing the button Deactivate LEVEL converter in the software. This will do a software reset and the STATE LED will flash normally again.

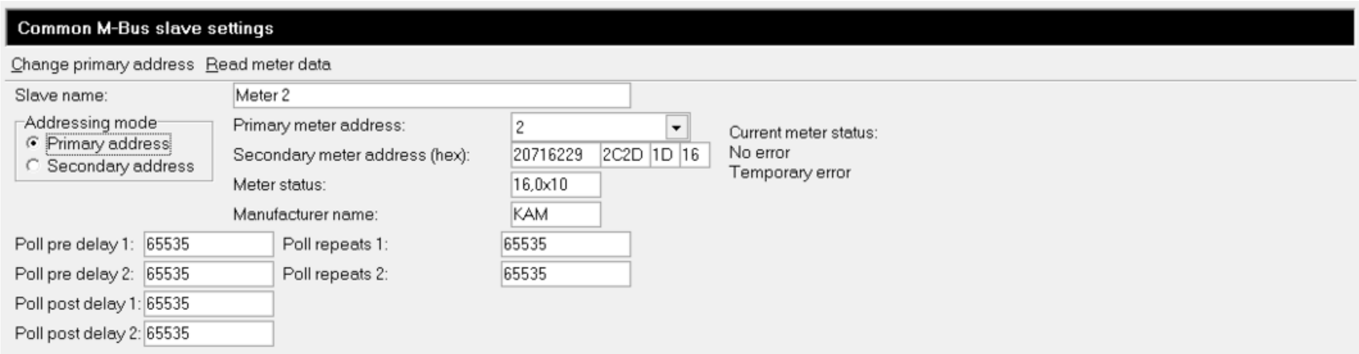
43.10.8 MBUS meter configuration

In the project tree you will find under the MBUS gateway for every configured meter a unique node. Click on this node. You will get the following result:



43.10.8.1 WHAT is displayed in the Common M-Bus slave settings

In this area you will find the following information:



- **Button Change primary address**: With this function you can program a new primary address in the selected meter, as long as the meter supports the standard MBUS command for setting a new primary address.
- **Button Read meter data**: With this function you can read out all MBUS datapoints from the connected meter again in the below data grid. This is useful, if you have erroneous deleted some datapoints of the meter and you want to restore the original datapoints of the meter.
- **Slave name**: Here you can define the name of the meter for the tree view and the documentation.
- **Addressing mode radio button**: This radio button selects the addressing mode for this meter. Either primary addressing mode in combination with the selected primary address in the field Primary meter address or Secondary addressing mode in combination with the first of the four fields in the row Secondary meter address. this is the field serial number of the meter.
- **Primary meter address**: This drop down defines the primary address for the meter either for readout or for programming a new primary address. Use 1 to 251 for slave address or if you have connected only one meter use 254 (Broadcast address), if you don't know the correct primary address.
- **Secondary meter address (hex)**: This four fields represents the following information:
  - **Serial number**: The first field is the current serial number of the meter. Or you enter a desired serial number for secondary addressing mode for a specific meter.
  - **Manufacturer ID**: The second field represents the two bytes of the manufacturer ID from the fixed data structure at the beginning of a variable data frame of the meter. The manufacturer is defined by three ASCII uppercase characters encoded with the following formula (In our example 2C2D stands for KAM=KAMSTRUP):
    IEC 870 Man.ID=[ASCII(1st letter)-64]•32•32+[ASCII(2nd letter)-64]•32+[ASCII(3rd letter)-64]
  - **Version**: The third field represents one byte from the fixed data structure at the beginning of a variable data frame of the meter defining the version of the meter.
  - **Medium**: The fourth field represents one byte from the fixed data structure at the beginning of a variable data frame of the meter defining the medium of the meter.
  - **Meter status**: This field represents one byte from the fixed data structure at the beginning of a variable data frame of the meter defining the status of the meter. Beside this field you will see under the caption current meter status the interpretation of the bits of this status byte as text.
  - **Manufacturer name**: This field shows the three ASCII letters from the two byte manufacturer ID from the fixed data structure at the beginning of a variable data frame of the meter. In our case KAM for KAMSTRUP.
  - **Poll pre delay 1**: This is a pause time in ms, before the gateway will send a primary or secondary address telegram to the meter to initiate the data readout process with this meter.
  - **Poll pre delay 2**: This is a pause time in ms, before the gateway will send a request for data telegram to the meter to readout more data from this meter.
  - **Poll post delay 1**: This is a pause time in ms, after the gateway will send a primary or secondary address telegram to the meter to initiate the data readout process with this meter.
  - **Poll post delay 2**: This is a pause time in ms, after the gateway will send a request for data telegram to the meter to readout more data from this meter.
  - **Poll repeats 1**: This is a repeat count, how often the MBUS gateway will send a primary or secondary address telegram to the meter, in the case the meter do not answer correctly.
  - **Poll repeats 2**: This is a repeat count, how often the MBUS gateway will send a request for data telegram to the meter, in the case the meter do not answer correctly.

All this setup parameters for the meter will be downloaded with the button Download configuration.

43.10.8.2 HOWTO set up individual poll parameters for one meter

In the basic setup of the gateway you will find the two parameters Query timeout and Poll timeout for general timing of the sequential process of requesting data from the connected meters. The two parameters can be configured like this:

- **Query timeout**: This field defines the timeout between two query cycles in the gateway. Usually the gateway communicates with all configured meters sequentially. After finishing the data readout for the last meter, the gateway pauses for this defined interval in seconds. This values are used:
  Value 65535 or values 0..5 defines ~5s pause.
  Values 6 to 65534: defines 6 to 65534 seconds of pause, before the next polling cycle will start.
- **Poll timeout**: This field defines a general pause after the readout of a configured meter before the readout of the next meter starts. In the past we discovered that there are many meters out in the market, which need a special treatment in the timing. e.g. very old KAMSTRUP meters need often two readout cycles with a gap of at least 10-15 seconds. This is non standard to the MBUS. Or other meters have problems with secondary addressing, if there is a too small gap between the readout. So we introduced this new parameter: This timeout defines the pause after finishing reading of a meter and starting reading the next meter. In the previous firmware versions this timeout was fixed to 250ms gap, which was ok for 99% of the meter readout on the markets. But some meter fail to process this little gap. The values is interpreted as follows:
  Value 1..30: Gap time 1 seconds to 30 seconds
  Value 101..400: Gaptime=(Value-100)*0.1s → 0.1s .. 30s e.g. 105 → 0.5s
  Value 65535: Gap time is 1 second
  Value 65534: Gap time is 250ms
  Value 65533: Gap time is 500ms
  Value 65532: Gap time is7250ms
  All other values: Gap time is 1000ms

Here you will find a basic diagram, how the MBUS master request cycle is handled by our gateways.

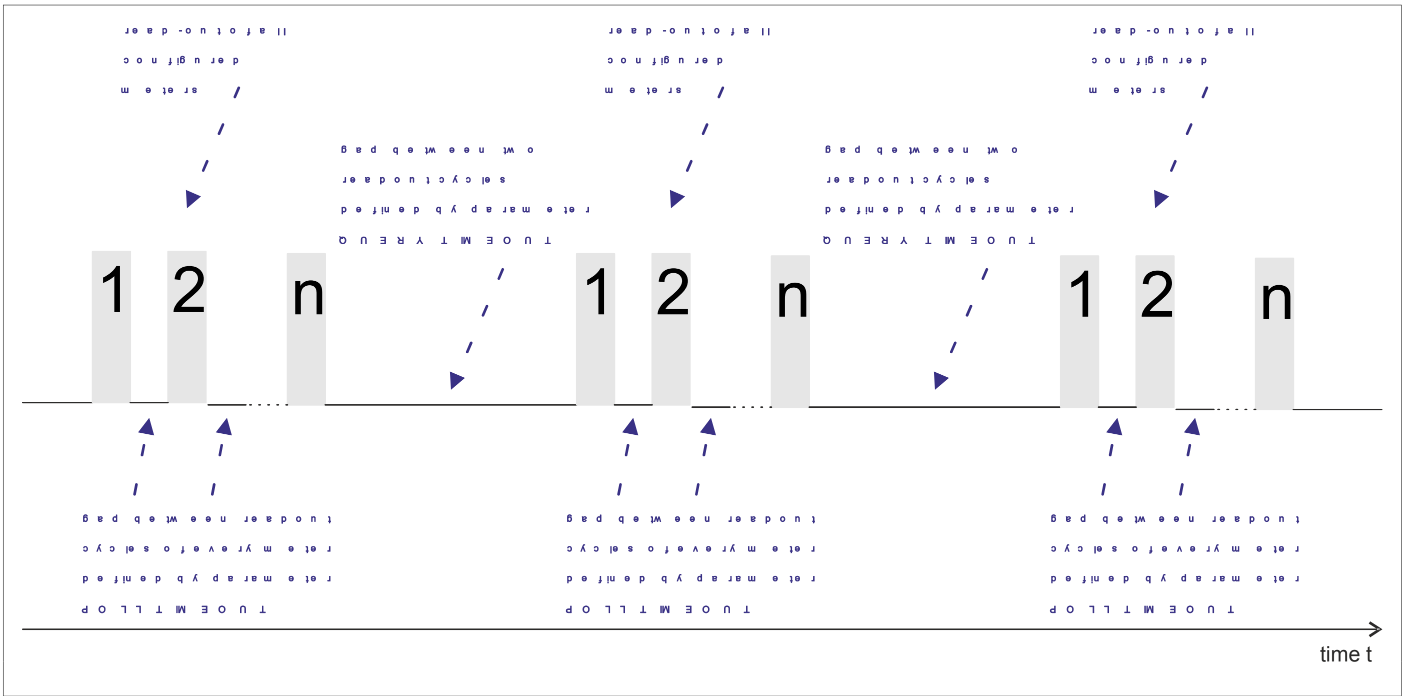


Figure: Basic timing of MBUS master read-out for MBUS slaves

Now we go more into detail, how the MBUS gateway will handle the request process of one meter. Forst we define the parameters:

- **Poll repeats 1**: This field defines the amount of telegram repetitions for the addressing command to a meter, before the gateway declares the communication as not possible and resumes with the next meter.
  Value 65535 or 0: use 3 repeats as standard
  Value 1..n: Use n repeats
- **Poll repeats 2**: This field defines the amount of telegram repetitions for the data readout command to a meter, before the gateway declares the communication as not possible and resumes with the next meter.
  Value 65535 or 0: use 5 repeats as standard
  Value 1..n: Use n repeats
- **Poll pre delay 1**: This field defines the first pause time in Milliseconds before starting to send the first addressing command telegram to a meter.
  Value 65535: use 250ms as standard pause time
  Value 0..65534: Use x ms as pause time
- **Poll pre delay 2**: This field defines the first pause time in Milliseconds before starting to send the first data request telegram to a meter.
  Value 65535: use 100ms as standard pause time
  Value 0..65534: Use x ms as pause time
- **Poll post delay 1**: This field defines a pause time in Milliseconds. If the gateway do not receive a correct answer to an addressing command telegram and the addressing command is repeated, then this pause time is inserted, before resending the addressing telegram to the meter.
  Value 65535: use 0ms as standard pause time
  Value 0..65534: Use x ms as pause time
- **Poll post delay 2**: This field defines a pause time in Milliseconds. If the gateway do not receive a correct answer to a readout data telegram and the readout data command is repeated, then this pause time is inserted, before resending the readout data telegram to the meter.
  Value 65535: use 100ms as standard pause time
  Value 0..65534: Use x ms as pause time

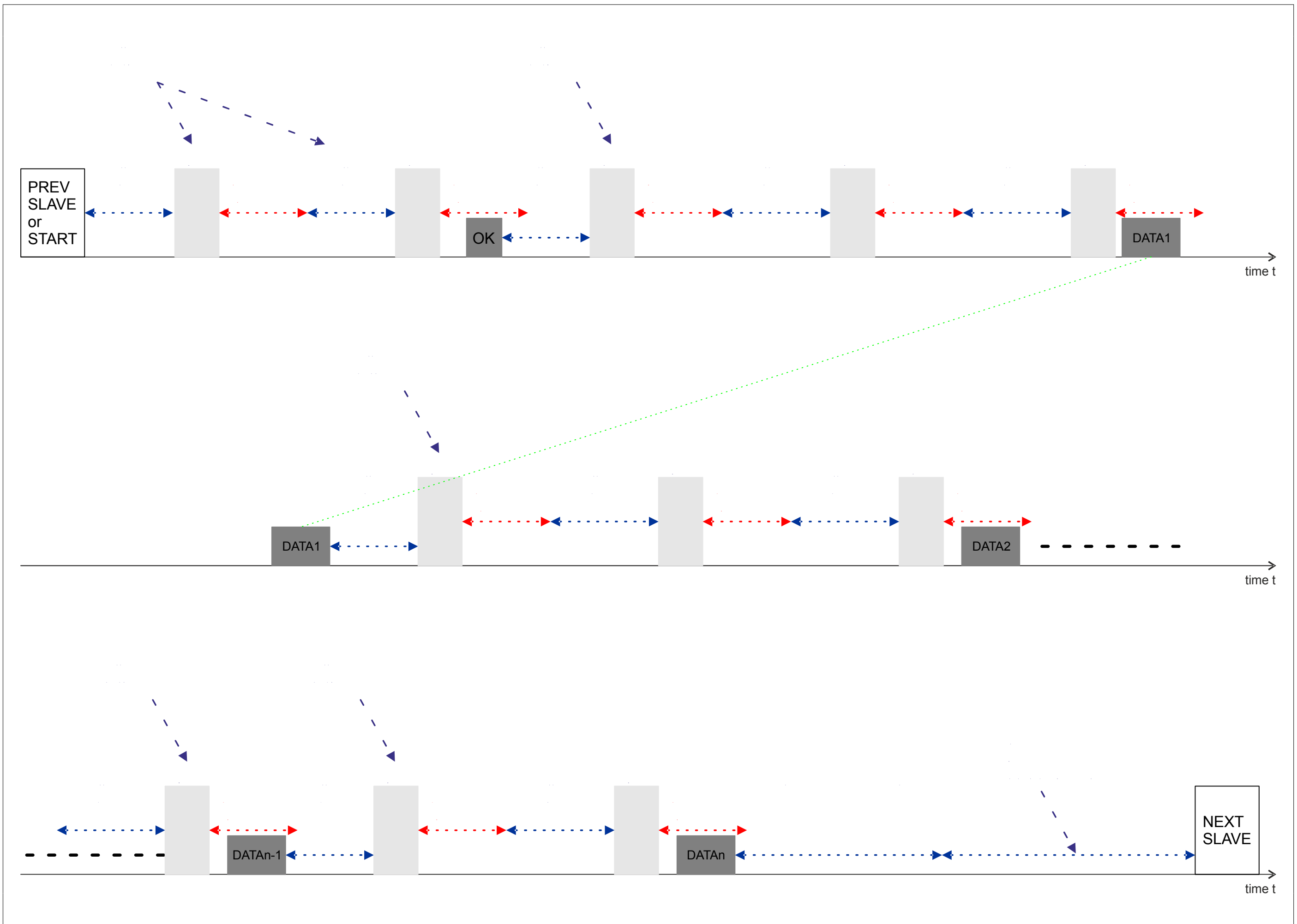
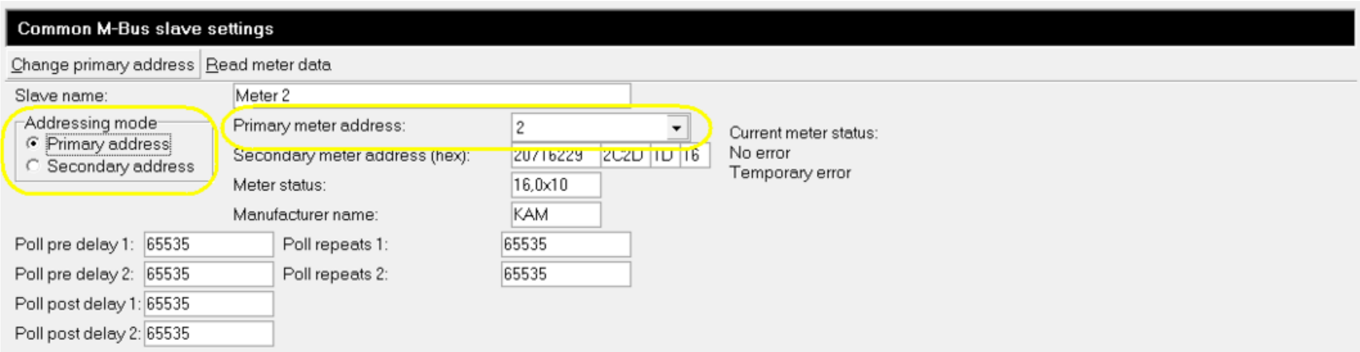


Figure: Basic timing of MBUS master read-out for MBUS slaves

43.10.8.3 HOWTO select primary addressing mode

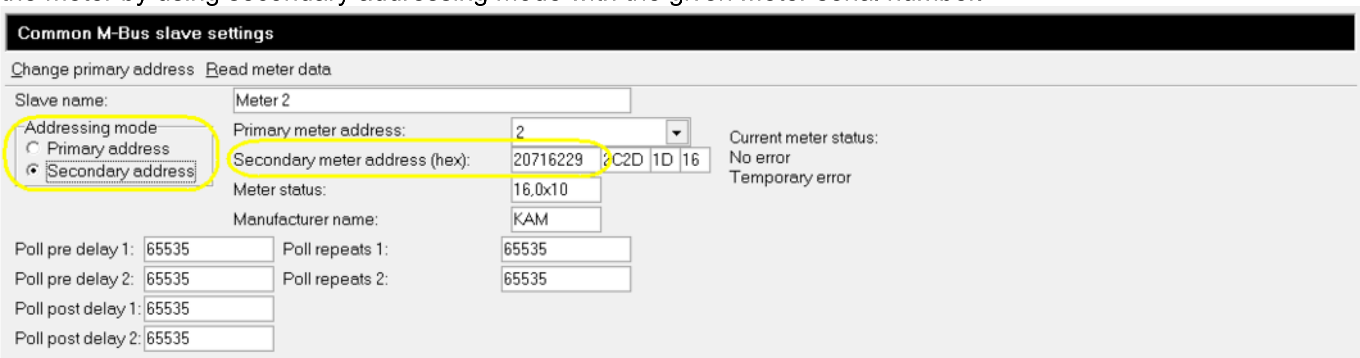
To select primary addressing mode, you have to define a primary address for the meter in the range from 1 to 151. and you have to select in the Addressing mode radio button the mode Primary address. After you have successfully downloaded the configuration into the gateway, this meter will be addressed by the meter by using primary addressing mode with the given primary address.

Don't forget, that the meter will only answer to the request, if the meter is programmed for the defined primary address, the meter uses the same MBUS baud rate and there is not another meter on the MBUS with the same primary address.



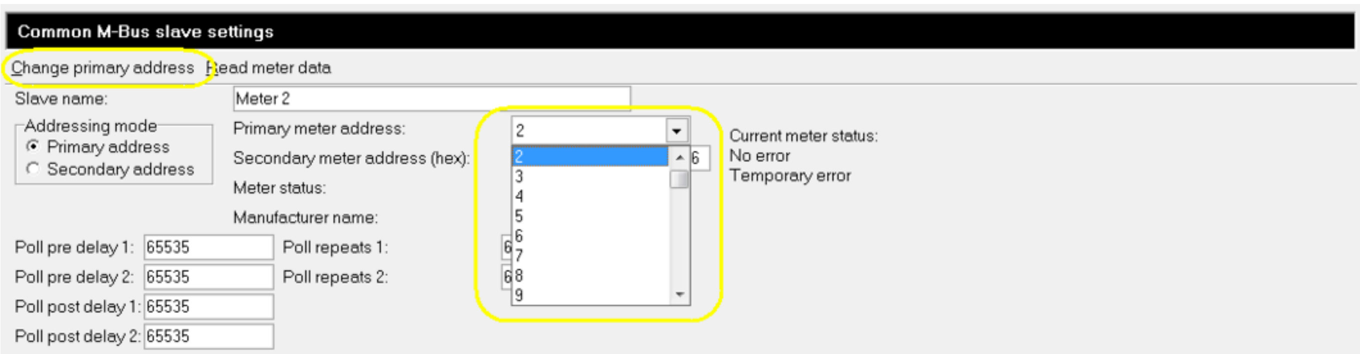
43.10.8.4 HOWTO select secondary addressing mode

To select secondary addressing mode, you have to define the unique meter ID (serial number) for the meter in the field Secondary meter address (hex). Then you have to select in the Addressing mode radio button the mode Secondary address. After you have successfully downloaded the configuration into the gateway, this meter will be addressed by the meter by using secondary addressing mode with the given Meter serial number.



43.10.8.5 HOWTO change the primary MBUS address in meter

When you want to change the primary address of the meter, first you have to select a new primary address form the drop down list Primary meter address. Take a unique address between 1 and 251 from the list and make sure, that you don't have another meter on the network with the same address you want to use in the future. Then click on the button Change primary address. Don't forget to change the Slave name. The standard is, that the slave name contains the primary address ant the end of the name.



43.10.8.6 WHAT is displayed in the Datapoints data grid

In this area you will find the following information:

Datapoints

Add datapoint Delete datapoint Add from database... Add to database...

Index	MBUS dataty...	MB datatype	Content	MBUS data	MBUS size	MBUS exponent	MB exponent
0	INT32	FLOAT32	Volume:10^-3 m³	1-2	4	-3	0
1	INT32	FLOAT32	Volume:10^-3 m³-Accumulation of abs value only if negative cont...	1-9	4	-3	0
2	INT32	UINT32	On time:hours	1-15	4	0	0
3	INT16	FLOAT32	Volume flow:10^-3 m³/h	1-21	2	-3	0
4	INT8	FLOAT32	External temperature:10^0 °C	1-25	1	0	0
5	INT16	FLOAT32	Volume flow:10^-3 m³/h	1-28	2	-3	0
6	INT16	FLOAT32	Volume flow:10^-3 m³/h	1-32	2	-3	0
7	INT8	FLOAT32	External temperature:10^0 °C	1-36	1	0	0
8	INT8	FLOAT32	External temperature:10^0 °C	1-39	1	0	0
9	INT8	FLOAT32	External temperature:10^0 °C-Average media temperature	1-44	1	0	0
10	INT32	DATE_TIME_T...	Time&Date data type F	1-47	4	0	0
11	INT32	FLOAT32	Volume:10^-3 m³[U:0,T:0,S:1]	1-53	4	-3	0
12	INT16	FLOAT32	Volume flow:10^-3 m³/h[U:0,T:0,S:1]	1-59	2	-3	0
13	INT16	FLOAT32	Volume flow:10^-3 m³/h[U:0,T:0,S:1]	1-63	2	-3	0
14	INT8	FLOAT32	External temperature:10^0 °C[U:0,T:0,S:1]	1-67	1	0	0
15	INT8	FLOAT32	External temperature:10^0 °C[U:0,T:0,S:1]	1-70	1	0	0
16	INT8	FLOAT32	External temperature:10^0 °C-Average media temperature[U:0,T...	1-75	1	0	0
17	INT16	DATE_TYP_G	Date data type G[U:0,T:0,S:1]	1-78	2	0	0
18	INT16	UINT16	Info code	1-83	2	0	0
19	INT48	UINT64	Config number	1-88	6	0	0
20	INT16	UINT16	Meter type	1-97	2	0	0
21	INT16	UINT16	Firmware version	1-102	2	0	0

In this grid you will find all datapoints regarding the selected meter. The grid has the following columns:

- **Index**: This is a running index starting with 0 to see how many datapoints you have defined. This is important, because the amount of datapoint mappings between MBUS and MODBUS is limited like the amount of MODBUS registers. e.g. The RESI-MBUS64-SIO can handle 1200 MODBUS registers but only 600 MBUS datapoints in total.
- **MBUS datatype**: Here you will see the used data type in the MBUS frame.
- **MB datatype**: Here you will find the MODBUS data type to map the MBUS data type to MODBUS register.
- **Content**: here you will see the name of the datapoint. This name will be build automatically with the additional information in the MBUS data (DIF+VIF fields). But it can be changed manually to user data.
- **MBUS data**: Here you can see in which record and on which offset within this record the MBUS data was found. The writing is <record>-<offset> in Bytes. This describes the location in the variable data structure of the MBUS data frame.
- **MBUS size**: This column shows the current size of the MBUS data in bytes.
- **MBUS exponent**: This column shows the exponent of the MBUS value, how it is defined in the MBUS data due to the DIF+VIF fields.
- **MB exponent**: This column shows the user defined exponent to shift the value in MODBUS registers.

43.10.8.7 HOWTO delete datapoints for a meter configuration

Since every MBUS datapoint needs mapping space in the MODBUS registers and the MODBUS registers are limited in the gateway, it makes sense to configure only those datapoints, which are necessary for your application. IN our example we don't want to read the storage values defined by storage number S:1. So we select all lines with this items (Use the pressed Control key and the mouse to do a multiselect on the grid), and the we delet the selected datapoints from the list by pressing the button Delete datapoint.

Datapoints

Add datapoint Delete datapoint Add from database... Add to database...

Index	MBUS dataty...	MB datatype	Content	MBUS data	MBUS size	MBUS exponent	MB exponent
0	INT32	FLOAT32	Volume:10^-3 m³	1-2	4	-3	0
1	INT32	FLOAT32	Volume:10^-3 m³-Accumulation of abs value only if negative cont...	1-9	4	-3	0
2	INT32	UINT32	On time:hours	1-15	4	0	0
3	INT16	FLOAT32	Volume flow:10^-3 m³/h	1-21	2	-3	0
4	INT8	FLOAT32	External temperature:10^0 °C	1-25	1	0	0
5	INT16	FLOAT32	Volume flow:10^-3 m³/h	1-28	2	-3	0
6	INT16	FLOAT32	Volume flow:10^-3 m³/h	1-32	2	-3	0
7	INT8	FLOAT32	External temperature:10^0 °C	1-36	1	0	0
8	INT8	FLOAT32	External temperature:10^0 °C	1-39	1	0	0
9	INT8	FLOAT32	External temperature:10^0 °C-Average media temperature	1-44	1	0	0
10	INT32	DATE_TIME_T...	Time&Date data type F	1-47	4	0	0
11	INT32	FLOAT32	Volume:10^-3 m³[U:0,T:0,S:1]	1-53	4	-3	0
12	INT16	FLOAT32	Volume flow:10^-3 m³/h[U:0,T:0,S:1]	1-59	2	-3	0
13	INT16	FLOAT32	Volume flow:10^-3 m³/h[U:0,T:0,S:1]	1-63	2	-3	0
14	INT8	FLOAT32	External temperature:10^0 °C[U:0,T:0,S:1]	1-67	1	0	0
15	INT8	FLOAT32	External temperature:10^0 °C[U:0,T:0,S:1]	1-70	1	0	0
16	INT8	FLOAT32	External temperature:10^0 °C-Average media temperature[U:0,T...	1-75	1	0	0
17	INT16	DATE_TYP_G	Date data type G[U:0,T:0,S:1]	1-78	2	0	0
18	INT16	UINT16	Info code	1-83	2	0	0
19	INT48	UINT64	Config number	1-88	6	0	0
20	INT16	UINT16	Meter type	1-97	2	0	0
21	INT16	UINT16	Firmware version	1-102	2	0	0

Your new list will look like this. If you download this configuration, only the desired datapoints are mapped to the MODBUS registers. The gateway requests only as much MBUS frames as necessary for mapping all values to the MODBUS registers.

Datapoints

Add datapoint Delete datapoint Add from database... Add to database...

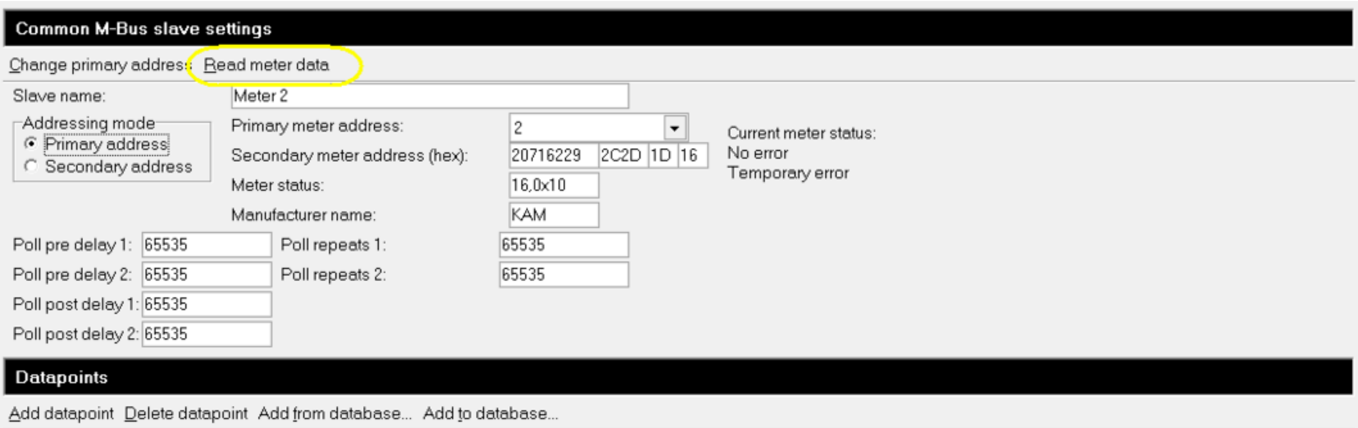
Index	MBUS dataty...	MB datatype	Content	MBUS data	MBUS size	MBUS exponent	MB exponent
0	INT32	FLOAT32	Volume:10^-3 m³	1-2	4	-3	0
1	INT32	FLOAT32	Volume:10^-3 m³-Accumulation of abs value only if negative cont...	1-9	4	-3	0
2	INT32	UINT32	On time:hours	1-15	4	0	0
3	INT16	FLOAT32	Volume flow:10^-3 m³/h	1-21	2	-3	0
4	INT8	FLOAT32	External temperature:10^0 °C	1-25	1	0	0
5	INT16	FLOAT32	Volume flow:10^-3 m³/h	1-28	2	-3	0
6	INT16	FLOAT32	Volume flow:10^-3 m³/h	1-32	2	-3	0
7	INT8	FLOAT32	External temperature:10^0 °C	1-36	1	0	0
8	INT8	FLOAT32	External temperature:10^0 °C	1-39	1	0	0
9	INT8	FLOAT32	External temperature:10^0 °C-Average media temperature	1-44	1	0	0
10	INT32	DATE_TIME_T...	Time&Date data type F	1-47	4	0	0
18	INT16	UINT16	Info code	1-83	2	0	0
19	INT48	UINT64	Config number	1-88	6	0	0
20	INT16	UINT16	Meter type	1-97	2	0	0
21	INT16	UINT16	Firmware version	1-102	2	0	0

But be aware, that you have changed your MODBUS register list also with this action:

MB Register	MBUS datatype	MB datatype	Content	MBUS index	MB value HEX	Cu
4x00001	INT32[4]	FLOAT32	Volume:10^-3 m³	0	???	??
4x00003	INT32[4]	FLOAT32	Volume:10^-3 m³-Accumulation of abs value only if negative contribu	1	???	??
4x00005	INT32[4]	UINT32	On time:hours	2	???	??
4x00007	INT16[2]	FLOAT32	Volume flow:10^-3 m³/h	3	???	??
4x00009	INT8[1]	FLOAT32	External temperature:10^0 °C	4	???	??
4x00011	INT16[2]	FLOAT32	Volume flow:10^-3 m³/h	5	???	??
4x00013	INT16[2]	FLOAT32	Volume flow:10^-3 m³/h	6	???	??
4x00015	INT8[1]	FLOAT32	External temperature:10^0 °C	7	???	??
4x00017	INT8[1]	FLOAT32	External temperature:10^0 °C	8	???	??
4x00019	INT8[1]	FLOAT32	External temperature:10^0 °C-Average media temperature	9	???	??
4x00021	INT32[4]	DATE_TIME_T	Time&Date data type F	10	???	??
4x00023	INT16[2]	UINT16	Info code	18	???	??
4x00024	INT48[6]	UINT64	Config number	19	???	??
4x00028	INT16[2]	UINT16	Meter type	20	???	??
4x00029	INT16[2]	UINT16	Firmware version	21	???	??
4x00030	VAR LENGTH[18]	ASCII	Manufacturer	0	???	??
4x00040	VAR LENGTH[8]	ASCII	Model/version	1	???	??
4x00045	VAR LENGTH[7]	ASCII	Firmware version	2	???	??
4x00049	INT24[3]	UINT32	Error flags (binary)	3	???	??
4x00051	FLOAT32[4]	FLOAT32	Current 10^0A-L1 phase value	4	???	??
4x00053	FLOAT32[4]	FLOAT32	Current 10^0A-L2 phase value	5	???	??
4x00055	FLOAT32[4]	FLOAT32	Current 10^0A-L3 phase value	6	???	??
4x00057	FLOAT32[4]	FLOAT32	Current 10^0A-Average current	7	???	??
4x00059	FLOAT32[4]	FLOAT32	Voltage 10^0V-L1-L2	8	???	??
4x00061	FLOAT32[4]	FLOAT32	Voltage 10^0V-L2-L3	9	???	??
4x00063	FLOAT32[4]	FLOAT32	Voltage 10^0V-L3-L1	10	???	??
4x00065	FLOAT32[4]	FLOAT32	Voltage 10^0V-Voltage L-L average	11	???	??
4x00067	FLOAT32[4]	FLOAT32	Voltage 10^0V-L1 phase value	12	???	??
4x00069	FLOAT32[4]	FLOAT32	Voltage 10^0V-L2 phase value	13	???	??
4x00071	FLOAT32[4]	FLOAT32	Voltage 10^0V-L3 phase value	14	???	??

43.10.8.8 HOWTO refresh datapoints for a meter configuration

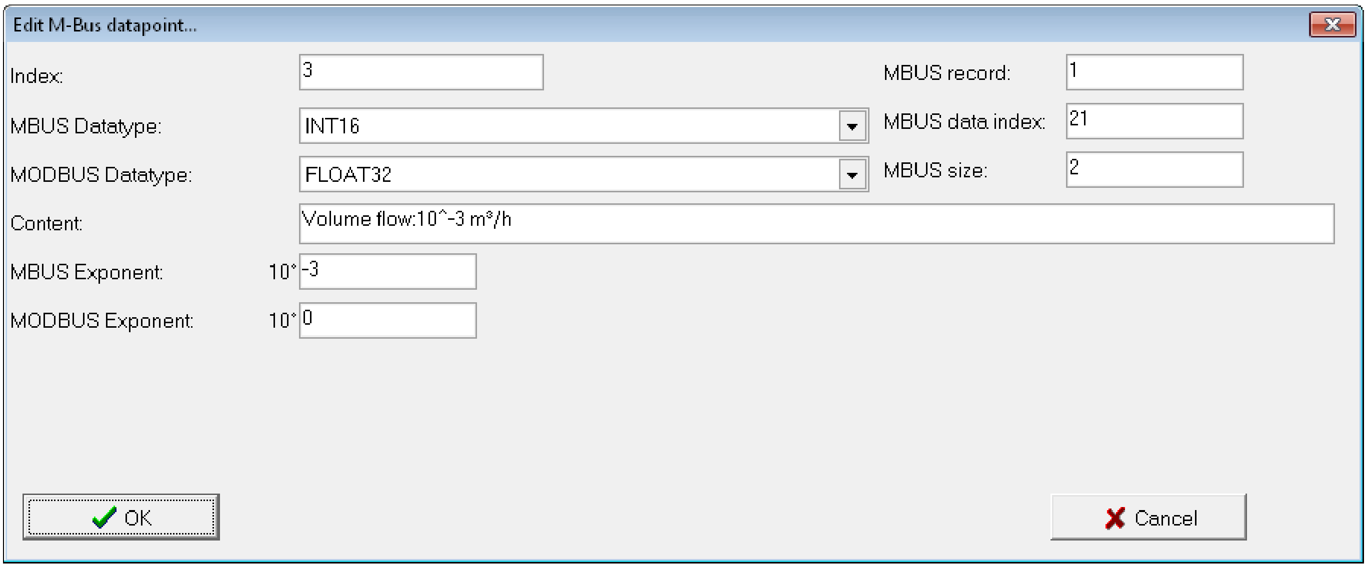
So if you have deleted some datapoints for one meter and you want to restore the original mapping from the meter, you can simple press the button "Read meter data". It will scan all MBUS datapoints of the selected meter again and refresh the list:



Index	MBUS dataty...	MB datatype	Content	MBUS data	MBUS size	MBUS exponent	MB exponent
0	INT32	FLOAT32	Volume:10^-3 m³	1-2	4	-3	0
1	INT32	FLOAT32	Volume:10^-3 m³-Accumulation of abs value only if negative cont...	1-9	4	-3	0
2	INT32	UINT32	On time:hours	1-15	4	0	0
3	INT16	FLOAT32	Volume flow:10^-3 m³/h	1-21	2	-3	0
4	INT8	FLOAT32	External temperature:10^0 °C	1-25	1	0	0
5	INT16	FLOAT32	Volume flow:10^-3 m³/h	1-28	2	-3	0
6	INT16	FLOAT32	Volume flow:10^-3 m³/h	1-32	2	-3	0
7	INT8	FLOAT32	External temperature:10^0 °C	1-36	1	0	0
8	INT8	FLOAT32	External temperature:10^0 °C	1-39	1	0	0
9	INT8	FLOAT32	External temperature:10^0 °C-Average media temperature	1-44	1	0	0
10	INT32	DATE_TIME_T...	Time&Date data type F	1-47	4	0	0
11	INT32	FLOAT32	Volume:10^-3 m³[U:0,T:0,S:1]	1-53	4	-3	0
12	INT16	FLOAT32	Volume flow:10^-3 m³/h[U:0,T:0,S:1]	1-59	2	-3	0
13	INT16	FLOAT32	Volume flow:10^-3 m³/h[U:0,T:0,S:1]	1-63	2	-3	0
14	INT8	FLOAT32	External temperature:10^0 °C[U:0,T:0,S:1]	1-67	1	0	0
15	INT8	FLOAT32	External temperature:10^0 °C[U:0,T:0,S:1]	1-70	1	0	0
16	INT8	FLOAT32	External temperature:10^0 °C-Average media temperature[U:0,T...	1-75	1	0	0
17	INT16	DATE_TYP_G	Date data type G[U:0,T:0,S:1]	1-78	2	0	0
18	INT16	UINT16	Info code	1-83	2	0	0
19	INT48	UINT64	Config number	1-88	6	0	0
20	INT16	UINT16	Meter type	1-97	2	0	0
21	INT16	UINT16	Firmware version	1-102	2	0	0

43.10.8.9 HOWTO modify MBUS datapoint mapping manually

The MODBUSConfigurator software will try to map the MBUS data types automatically to correct MODBUS data types and MODBUS registers. But you can also modify this mapping. Double click onto an item in the data grid, you will see the following dialog:

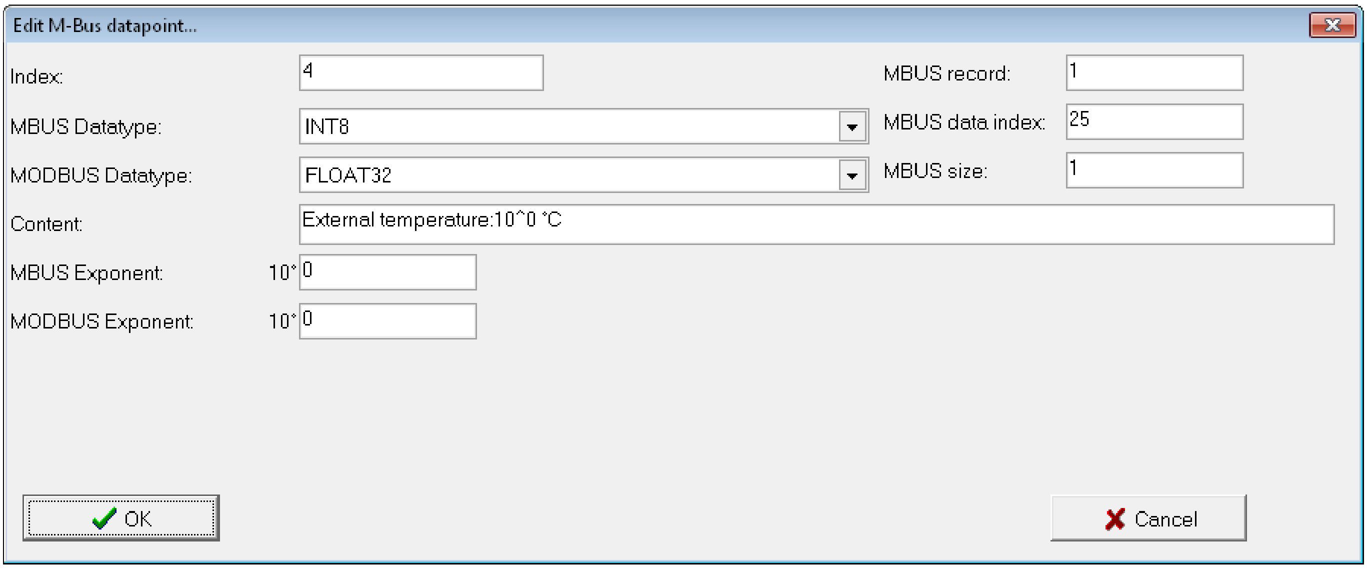


Basically it is the data grid line in an editable version. You can change the content description here. Or you can change the MODBUS data type here. If you really add MBUS data frames manually you can also edit the MBUS data type, the MBUS exponent, the MBUS record number, the MBUS data index and the MBUS size in here to define the exact location of the MBUS data within the MBUS data frame.

Usually you will change the MODBUS exponent and or the MODBUS data type. Lets do a sample configuration change:

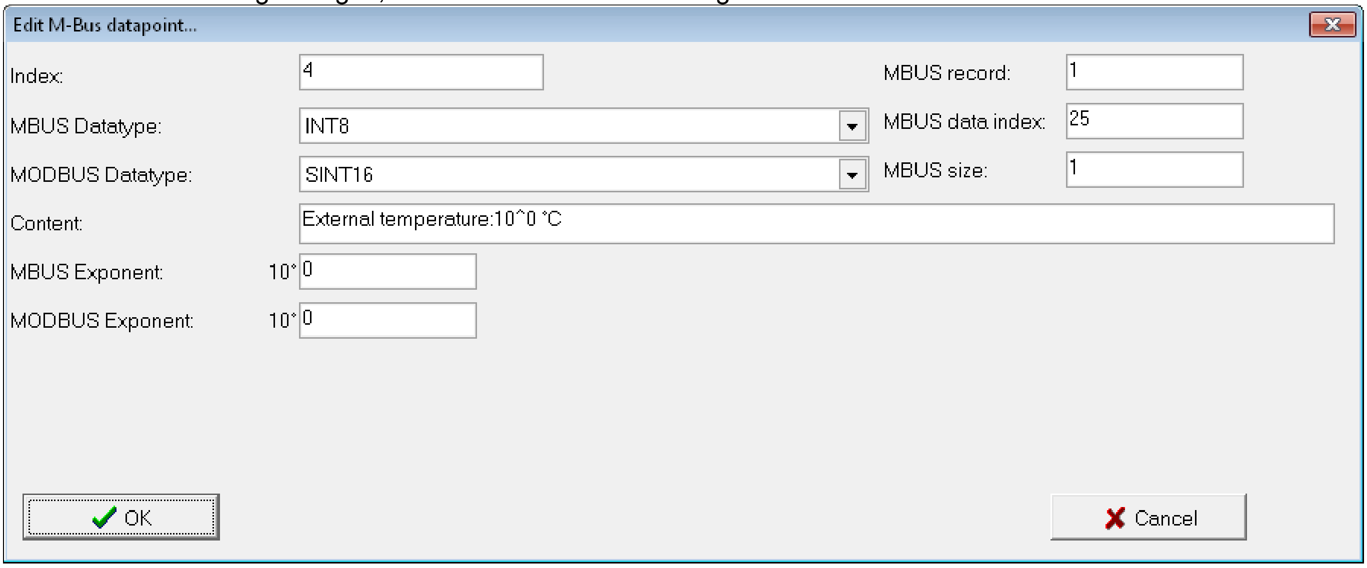
MB Register	MBUS datatype	MB datatype	Content	MBUS index	MB value HEX	Current MB value	Meter name
4x00001	INT32[4]	FLOAT32	Volume:10^-3 m³	0	MSW:0000,0000:LSW	0.0000,0.0000000000000E+0	Meter 2 [P:2]
4x00003	INT32[4]	FLOAT32	Volume:10^-3 m³-Accumulation of abs value only if negative contribu	1	MSW:0000,0000:LSW	0.0000,0.0000000000000E+0	Meter 2 [P:2]
4x00005	INT32[4]	UINT32	On time:hours	2	MSW:0000,1183:LSW	4483,0x00001183	Meter 2 [P:2]
4x00007	INT16[2]	FLOAT32	Volume flow:10^-3 m³/h	3	MSW:0000,0000:LSW	0.0000,0.0000000000000E+0	Meter 2 [P:2]
4x00009	INT8[1]	FLOAT32	External temperature:10^0 °C	4	MSW:41E0,0000:LSW	28.0000,2.8000000000000E+1	Meter 2 [P:2]
4x00011	INT16[2]	FLOAT32	Volume flow:10^-3 m³/h	5	MSW:0000,0000:LSW	0.0000,0.0000000000000E+0	Meter 2 [P:2]
4x00013	INT16[2]	FLOAT32	Volume flow:10^-3 m³/h	6	MSW:0000,0000:LSW	0.0000,0.0000000000000E+0	Meter 2 [P:2]
4x00015	INT8[1]	FLOAT32	External temperature:10^0 °C	7	MSW:41B8,0000:LSW	23.0000,2.3000000000000E+1	Meter 2 [P:2]

As you can see from the live data, the external temperature is currently 28°C. Our automatic mapping algorithm maps the MBUS data type INT8 (8 bit SIGNED INTEGER) to a FLOAT32 using two consecutive MODBUS registers, because we try to show on the MODBUS side the correct value with the correct exponent. But in this special case a standard Holding register will be enough.



So we change the configuration from FLOAT32 to SINT16 to map the value into a single holding register. This saves register space and it also increases the conversion accuracy to 100%, because INT8 to SINT16 is as loss free conversation in comparison to INT8 to FLOAT32 is not a loss free conversation, because the FLOAT32 format is too inaccurate to show in all cases the real INT8 value.

So we do the following changes, then we download the configuration and test it:

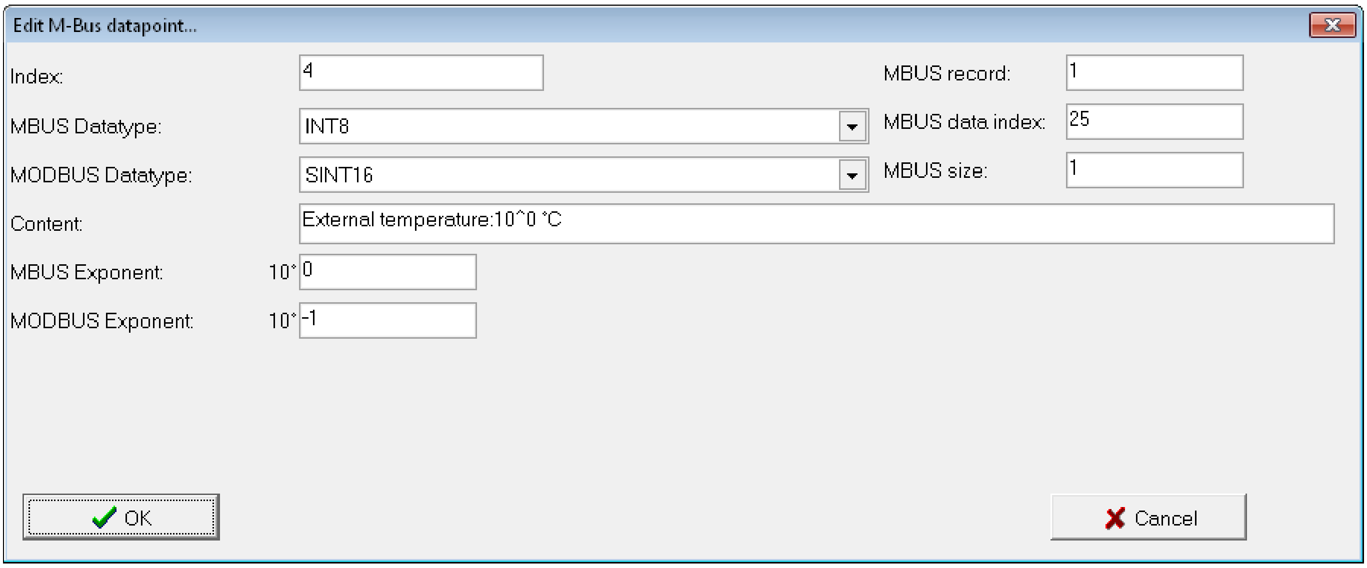


The result will be like this:

MB Register	MBUS datatype	MB datatype	Content	MBUS index	MB value HEX	Current MB value	Meter name
4x00001	INT32[4]	FLOAT32	Volume:10^-3 m³	0	MSW:0000,0000:LSW	0.0000,0.00000000000000E+0	Meter 2 [P:2]
4x00003	INT32[4]	FLOAT32	Volume:10^-3 m³-Accumulation of abs value only if negative contribu	1	MSW:0000,0000:LSW	0.0000,0.00000000000000E+0	Meter 2 [P:2]
4x00005	INT32[4]	UINT32	On time:hours	2	MSW:0000,1183:LSW	4483,0x00001183	Meter 2 [P:2]
4x00007	INT16[2]	FLOAT32	Volume flow:10^-3 m³/h	3	MSW:0000,0000:LSW	0.0000,0.00000000000000E+0	Meter 2 [P:2]
4x00009	INT8[1]	SINT16	External temperature:10^0 °C	4	WORD:0018	28,0x001C	Meter 2 [P:2]
4x00010	INT16[2]	FLOAT32	Volume flow:10^-3 m³/h	5	MSW:0000,0000:LSW	0.0000,0.00000000000000E+0	Meter 2 [P:2]
4x00012	INT16[2]	FLOAT32	Volume flow:10^-3 m³/h	6	MSW:0000,0000:LSW	0.0000,0.00000000000000E+0	Meter 2 [P:2]
4x00014	INT8[1]	FLOAT32	External temperature:10^0 °C	7	MSW:41B8,0000:LSW	23.0000,2.30000000000000E+1	Meter 2 [P:2]

Please note also, that the next MBUS datapoint starts not longer in the register 4x00011, It starts now in the register 4x00010. So we saved really one register.

Now we define, that your host can handle only temperatures with one comma. This means the 28°C should be stored as 280 in the holding register. For that we change the MODBUS exponent field to -1 to shift the result by 10:

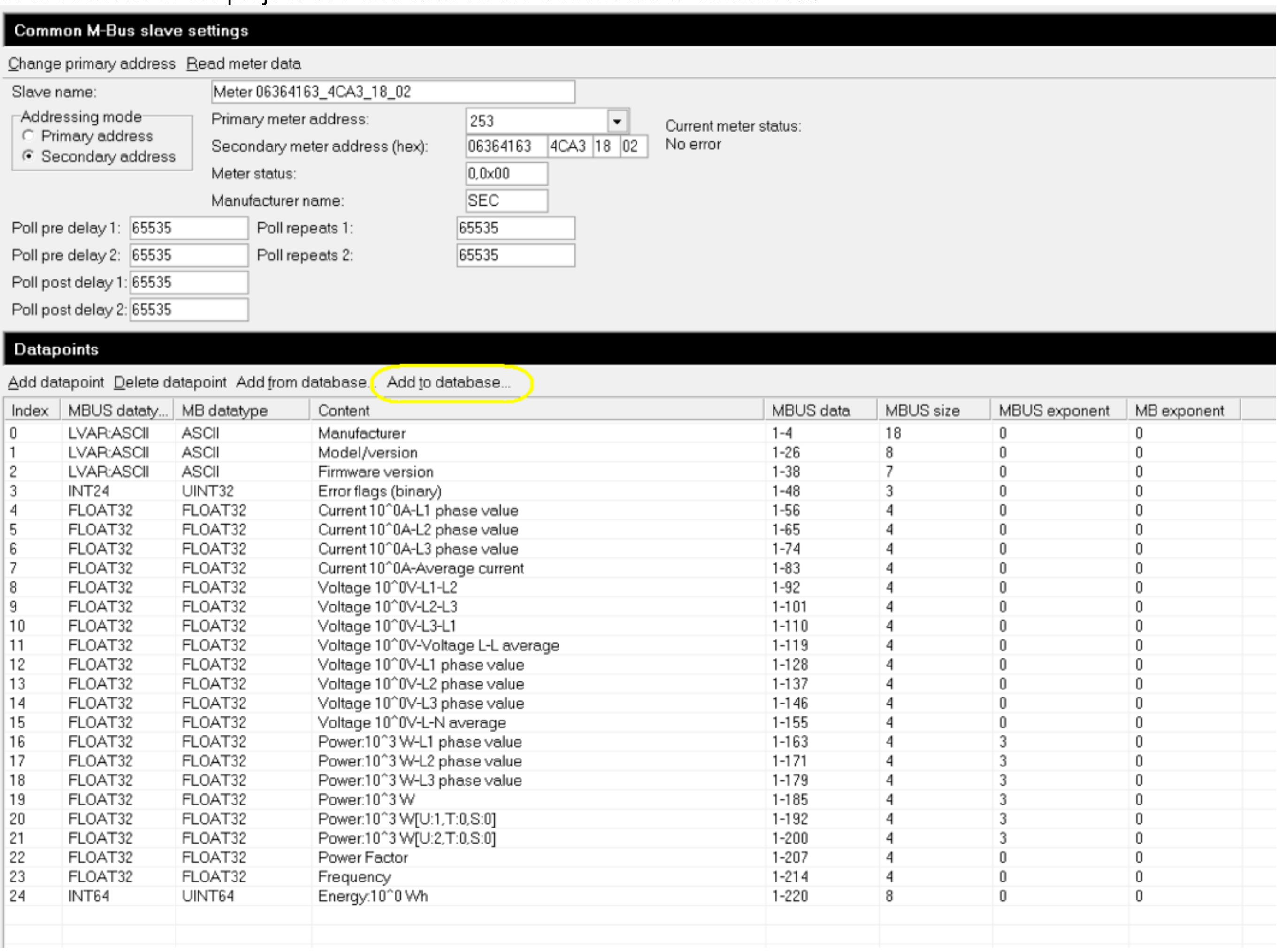


Now we get this result in test mode:

MB Register	MBUS datatype	MB datatype	Content	MBUS index	MB value HEX	Current MB value	Meter name
4x00001	INT32[4]	FLOAT32	Volume:10^-3 m³	0	MSW:0000,0000:LSW	0.0000,0.00000000000000E+0	Meter 2 [P:2]
4x00003	INT32[4]	FLOAT32	Volume:10^-3 m³-Accumulation of abs value only if negative contribu	1	MSW:0000,0000:LSW	0.0000,0.00000000000000E+0	Meter 2 [P:2]
4x00005	INT32[4]	UINT32	On time:hours	2	MSW:0000,1184:LSW	4484,0x00001184	Meter 2 [P:2]
4x00007	INT16[2]	FLOAT32	Volume flow:10^-3 m³/h	3	MSW:0000,0000:LSW	0.0000,0.00000000000000E+0	Meter 2 [P:2]
4x00009	INT8[1]	SINT16	External temperature:10^0 °C->*10^-1	4	WORD:0018	280,0x0118	Meter 2 [P:2]
4x00010	INT16[2]	FLOAT32	Volume flow:10^-3 m³/h	5	MSW:0000,0000:LSW	0.0000,0.00000000000000E+0	Meter 2 [P:2]
4x00012	INT16[2]	FLOAT32	Volume flow:10^-3 m³/h	6	MSW:0000,0000:LSW	0.0000,0.00000000000000E+0	Meter 2 [P:2]
4x00014	INT8[1]	FLOAT32	External temperature:10^0 °C	7	MSW:41B8,0000:LSW	23.0000,2.30000000000000E+1	Meter 2 [P:2]

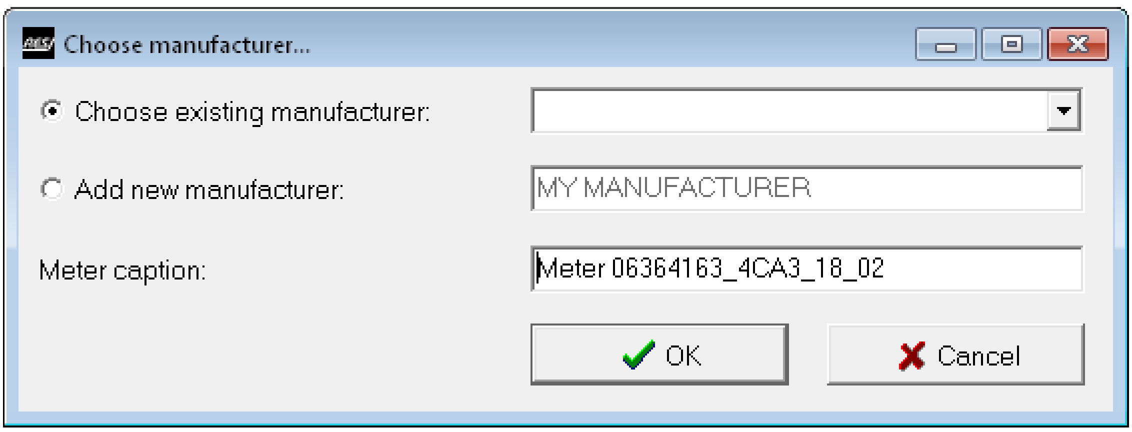
43.11 HOWTO save datapoints to user specific meter database

Our software offers the possibility to save a current meter setup to a user database for future use. Therefore select the desired meter in the project tree and click on the button Add to database...

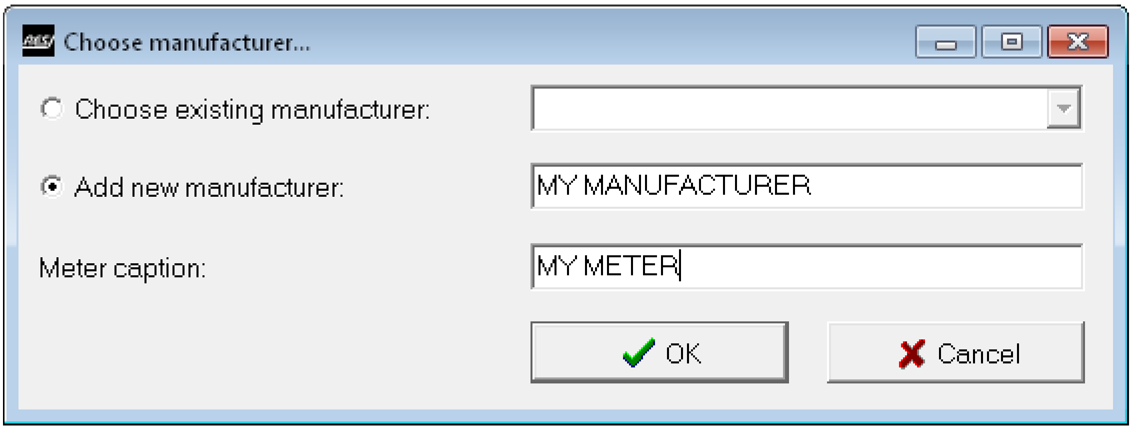


Index	MBUS dataty...	MB datatype	Content	MBUS data	MBUS size	MBUS exponent	MB exponent
0	LVAR:ASCII	ASCII	Manufacturer	1-4	18	0	0
1	LVAR:ASCII	ASCII	Model/version	1-26	8	0	0
2	LVAR:ASCII	ASCII	Firmware version	1-38	7	0	0
3	INT24	UINT32	Error flags (binary)	1-48	3	0	0
4	FLOAT32	FLOAT32	Current 10^0A-L1 phase value	1-56	4	0	0
5	FLOAT32	FLOAT32	Current 10^0A-L2 phase value	1-65	4	0	0
6	FLOAT32	FLOAT32	Current 10^0A-L3 phase value	1-74	4	0	0
7	FLOAT32	FLOAT32	Current 10^0A-Average current	1-83	4	0	0
8	FLOAT32	FLOAT32	Voltage 10^0V-L1-L2	1-92	4	0	0
9	FLOAT32	FLOAT32	Voltage 10^0V-L2-L3	1-101	4	0	0
10	FLOAT32	FLOAT32	Voltage 10^0V-L3-L1	1-110	4	0	0
11	FLOAT32	FLOAT32	Voltage 10^0V-Voltage L-L average	1-119	4	0	0
12	FLOAT32	FLOAT32	Voltage 10^0V-L1 phase value	1-128	4	0	0
13	FLOAT32	FLOAT32	Voltage 10^0V-L2 phase value	1-137	4	0	0
14	FLOAT32	FLOAT32	Voltage 10^0V-L3 phase value	1-146	4	0	0
15	FLOAT32	FLOAT32	Voltage 10^0V-L-N average	1-155	4	0	0
16	FLOAT32	FLOAT32	Power:10^3 W-L1 phase value	1-163	4	3	0
17	FLOAT32	FLOAT32	Power:10^3 W-L2 phase value	1-171	4	3	0
18	FLOAT32	FLOAT32	Power:10^3 W-L3 phase value	1-179	4	3	0
19	FLOAT32	FLOAT32	Power:10^3 W	1-185	4	3	0
20	FLOAT32	FLOAT32	Power:10^3 W[U:1,T:0,S:0]	1-192	4	3	0
21	FLOAT32	FLOAT32	Power:10^3 W[U:2,T:0,S:0]	1-200	4	3	0
22	FLOAT32	FLOAT32	Power Factor	1-207	4	0	0
23	FLOAT32	FLOAT32	Frequency	1-214	4	0	0
24	INT64	UINT64	Energy:10^0 Wh	1-220	8	0	0

You will see the following dialog:

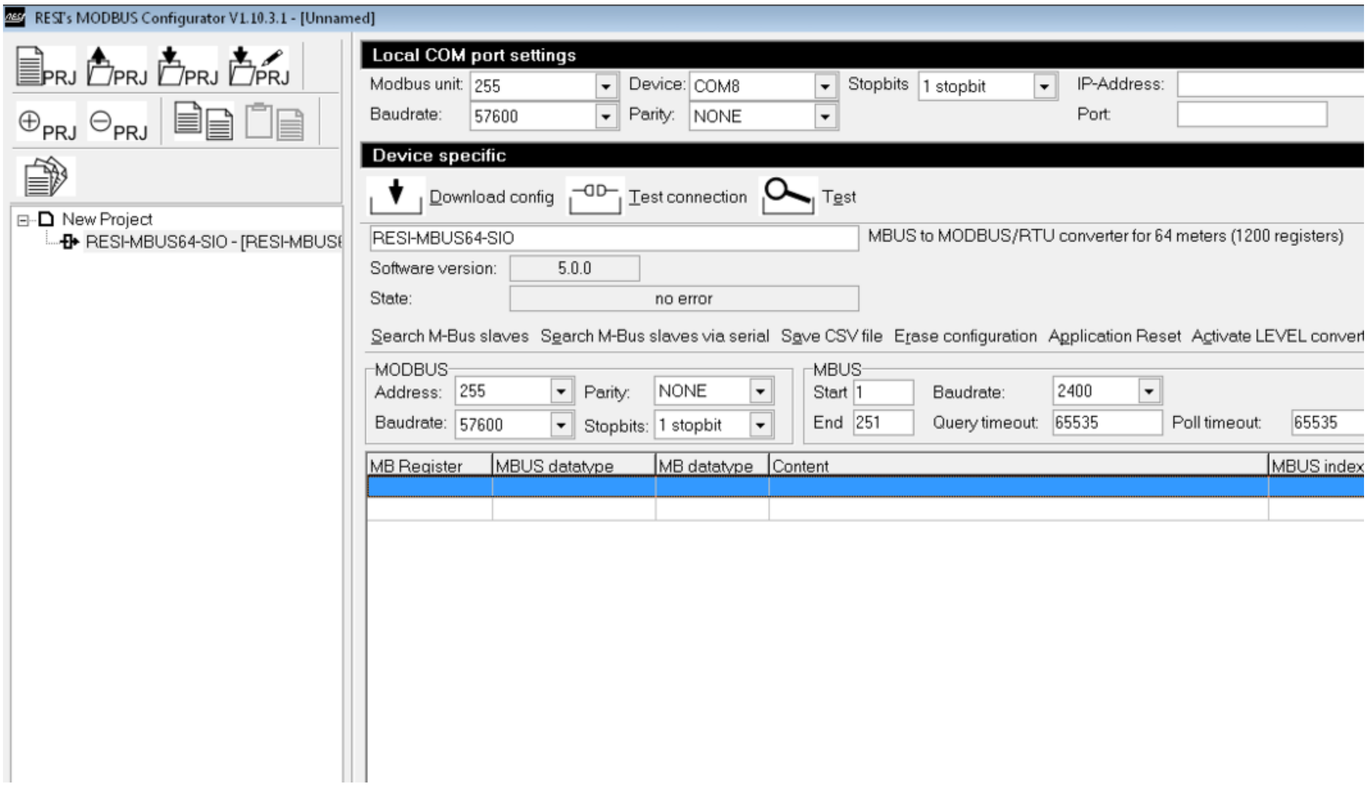


Either choose an existing manufacturer from the drop down list or set the radio button to the Add new manufacturer section and enter a new manufacturer name. In our example we choose the name MY MANUFACTURER and the meter name MY METER and we press the OK button. All your defined datapoints for this meter are stored in the user specific database and the meter is added to the user specific database for meter templates.

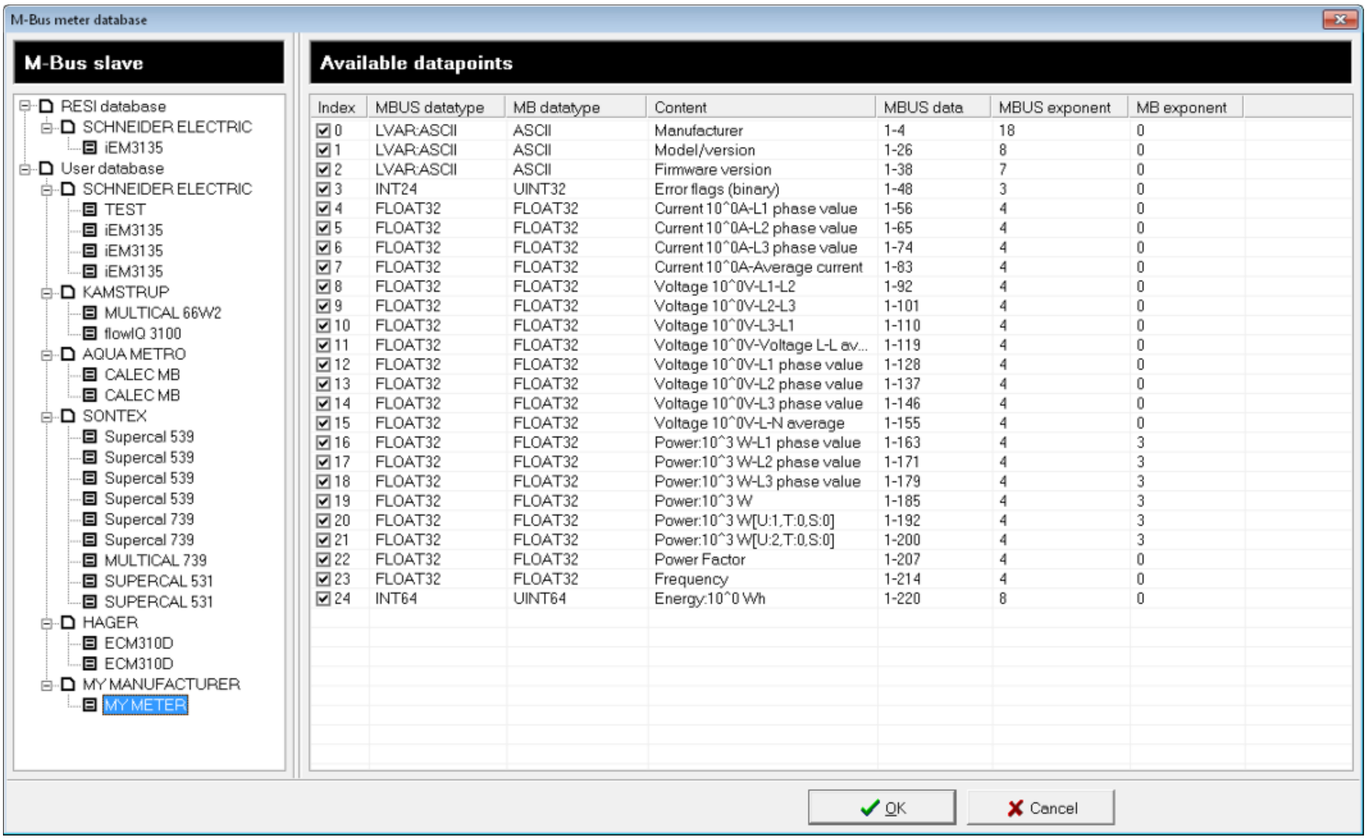


43.12 HOWTO add a complete meter from the database

You can add meter mappings manually to your gateway from previous saved own meters or form our general meter database. First you need a MBUS gateway in your project. Click on the project tree to select the MBUS gateway:



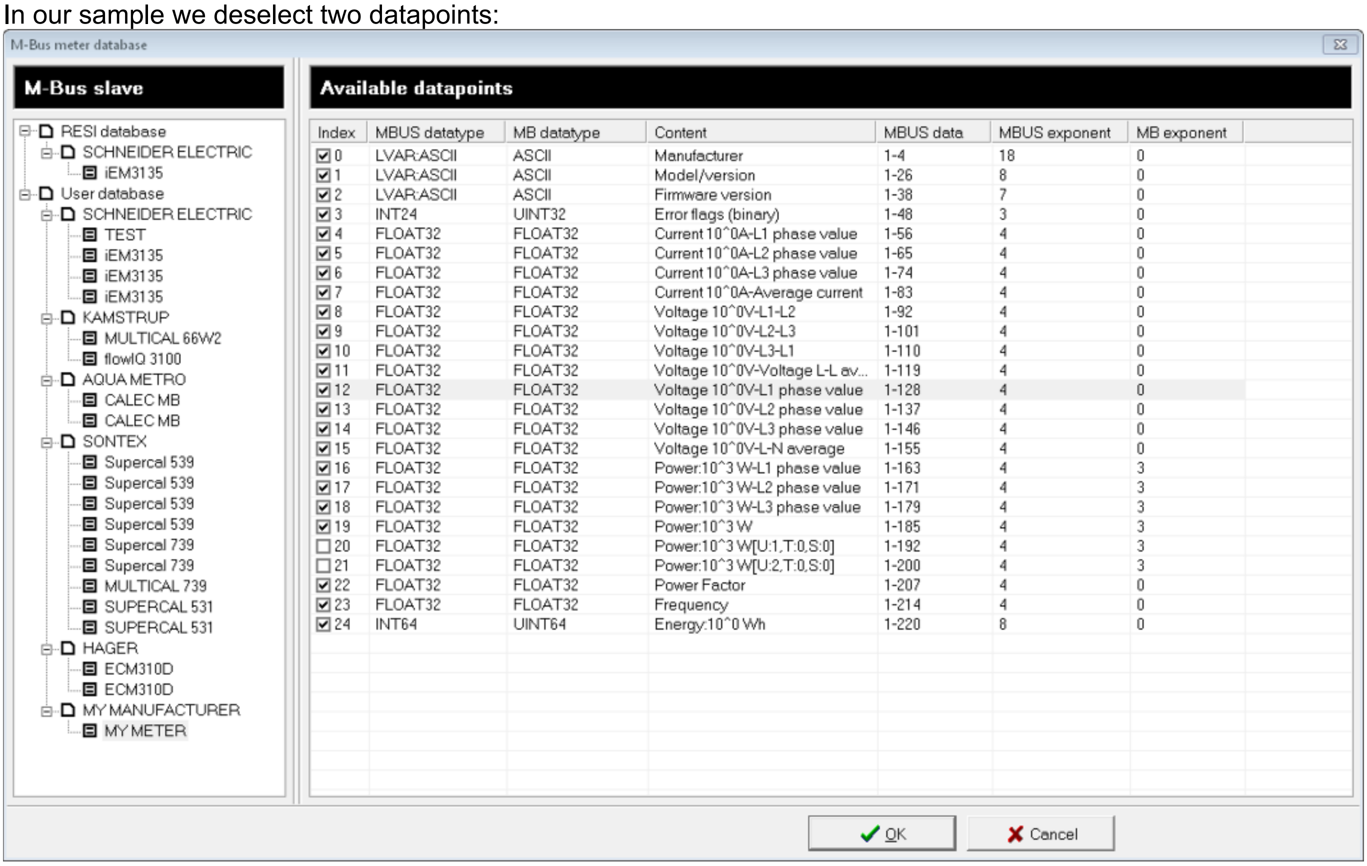
Now click on the button Add to project: . A dialog will open and show all meters from the general meter database and all your user defined meter templates:



Index	MBUS datatype	MB datatype	Content	MBUS data	MBUS exponent	MB exponent
☑ 0	LVAR:ASCII	ASCII	Manufacturer	1-4	18	0
☑ 1	LVAR:ASCII	ASCII	Model/version	1-26	8	0
☑ 2	LVAR:ASCII	ASCII	Firmware version	1-38	7	0
☑ 3	INT24	UINT32	Error flags (binary)	1-48	3	0
☑ 4	FLOAT32	FLOAT32	Current 10^0A-L1 phase value	1-56	4	0
☑ 5	FLOAT32	FLOAT32	Current 10^0A-L2 phase value	1-65	4	0
☑ 6	FLOAT32	FLOAT32	Current 10^0A-L3 phase value	1-74	4	0
☑ 7	FLOAT32	FLOAT32	Current 10^0A-Average current	1-83	4	0
☑ 8	FLOAT32	FLOAT32	Voltage 10^0V-L1-L2	1-92	4	0
☑ 9	FLOAT32	FLOAT32	Voltage 10^0V-L2-L3	1-101	4	0
☑ 10	FLOAT32	FLOAT32	Voltage 10^0V-L3-L1	1-110	4	0
☑ 11	FLOAT32	FLOAT32	Voltage 10^0V-Voltage L-L av...	1-119	4	0
☑ 12	FLOAT32	FLOAT32	Voltage 10^0V-L1 phase value	1-128	4	0
☑ 13	FLOAT32	FLOAT32	Voltage 10^0V-L2 phase value	1-137	4	0
☑ 14	FLOAT32	FLOAT32	Voltage 10^0V-L3 phase value	1-146	4	0
☑ 15	FLOAT32	FLOAT32	Voltage 10^0V-L-N average	1-155	4	0
☑ 16	FLOAT32	FLOAT32	Power:10^3 W-L1 phase value	1-163	4	3
☑ 17	FLOAT32	FLOAT32	Power:10^3 W-L2 phase value	1-171	4	3
☑ 18	FLOAT32	FLOAT32	Power:10^3 W-L3 phase value	1-179	4	3
☑ 19	FLOAT32	FLOAT32	Power:10^3 W	1-185	4	3
☑ 20	FLOAT32	FLOAT32	Power:10^3 W[U:1,T:0,S:0]	1-192	4	3
☑ 21	FLOAT32	FLOAT32	Power:10^3 W[U:2,T:0,S:0]	1-200	4	3
☑ 22	FLOAT32	FLOAT32	Power Factor	1-207	4	0
☑ 23	FLOAT32	FLOAT32	Frequency	1-214	4	0
☑ 24	INT64	UINT64	Energy:10^0 Wh	1-220	8	0

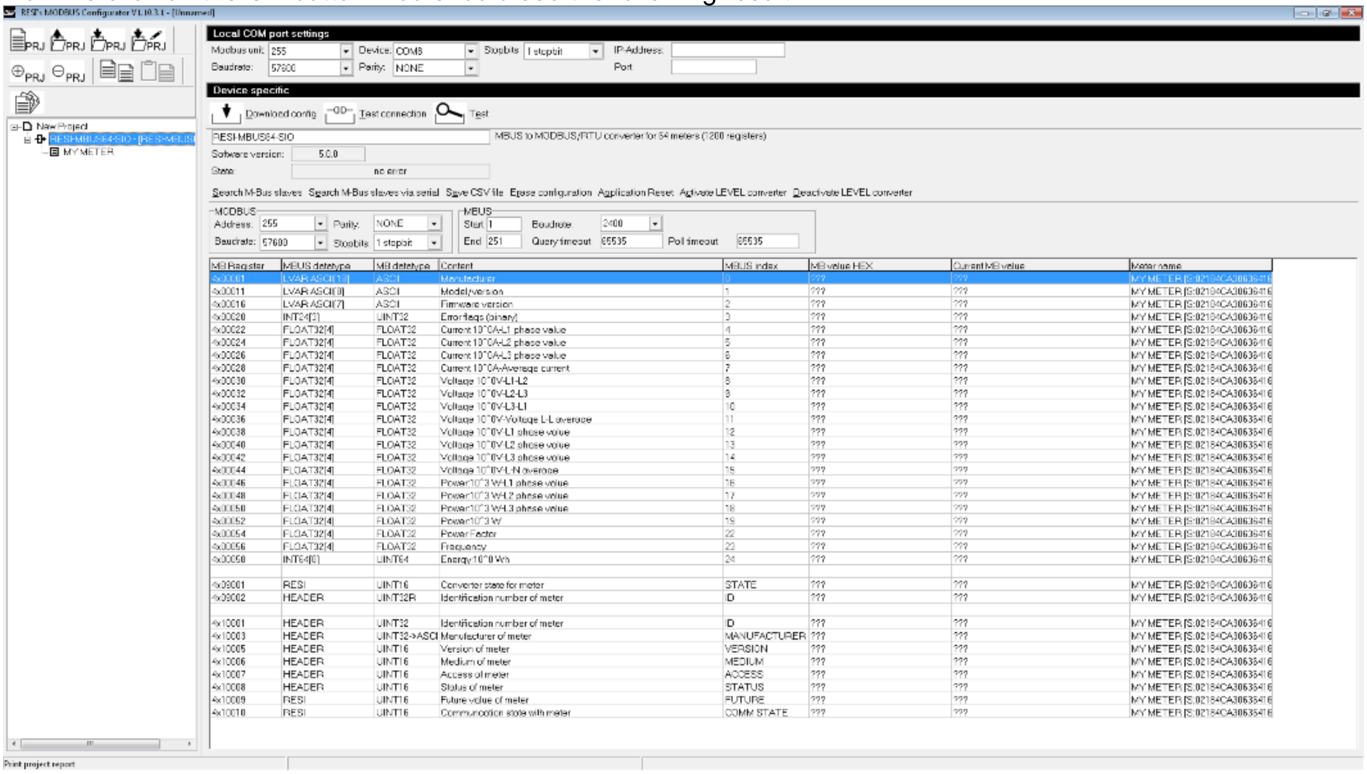
Select the meter MY MANUFACTURER/MY METER like shown above. Note the checkbox beside the datapoint index: Only the datapoints selected in this list are added to the gateway. You can change the selection status by clickin onto the checkbox for each datapoint. If you do a right click in the area of the data grind with the MBUS datapoints you will see a drop down menu with the two options Select all and Deselect all for fast selection/deselection of all datapoints.

In our sample we deselect two datapoints:

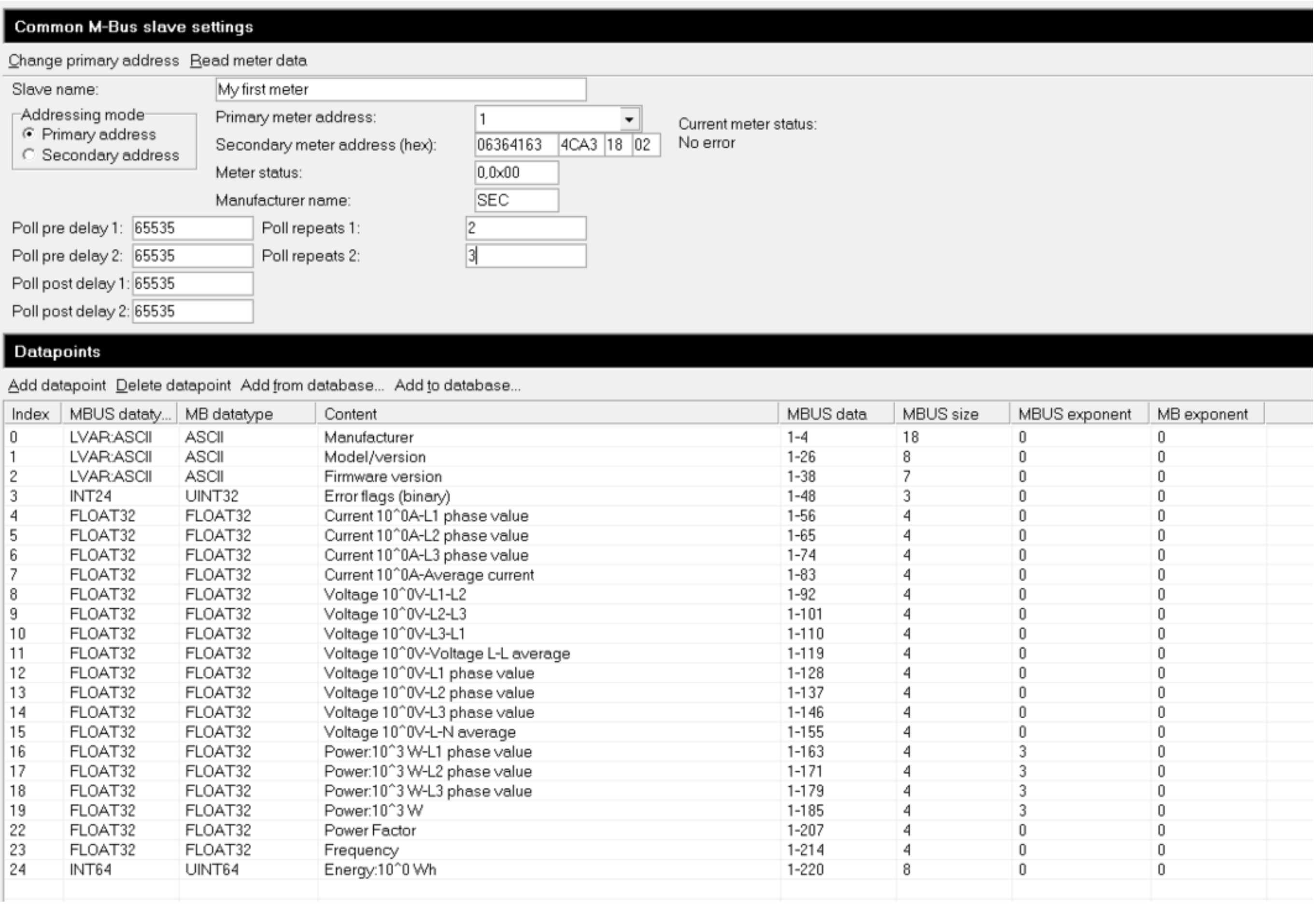


Index	MBUS datatype	MB datatype	Content	MBUS data	MBUS exponent	MB exponent
☑ 0	LVAR:ASCII	ASCII	Manufacturer	1-4	18	0
☑ 1	LVAR:ASCII	ASCII	Model/version	1-26	8	0
☑ 2	LVAR:ASCII	ASCII	Firmware version	1-38	7	0
☑ 3	INT24	UINT32	Error flags (binary)	1-48	3	0
☑ 4	FLOAT32	FLOAT32	Current 10^0A-L1 phase value	1-56	4	0
☑ 5	FLOAT32	FLOAT32	Current 10^0A-L2 phase value	1-65	4	0
☑ 6	FLOAT32	FLOAT32	Current 10^0A-L3 phase value	1-74	4	0
☑ 7	FLOAT32	FLOAT32	Current 10^0A-Average current	1-83	4	0
☑ 8	FLOAT32	FLOAT32	Voltage 10^0V-L1-L2	1-92	4	0
☑ 9	FLOAT32	FLOAT32	Voltage 10^0V-L2-L3	1-101	4	0
☑ 10	FLOAT32	FLOAT32	Voltage 10^0V-L3-L1	1-110	4	0
☑ 11	FLOAT32	FLOAT32	Voltage 10^0V-Voltage L-L av...	1-119	4	0
☑ 12	FLOAT32	FLOAT32	Voltage 10^0V-L1 phase value	1-128	4	0
☑ 13	FLOAT32	FLOAT32	Voltage 10^0V-L2 phase value	1-137	4	0
☑ 14	FLOAT32	FLOAT32	Voltage 10^0V-L3 phase value	1-146	4	0
☑ 15	FLOAT32	FLOAT32	Voltage 10^0V-L-N average	1-155	4	0
☑ 16	FLOAT32	FLOAT32	Power:10^3 W-L1 phase value	1-163	4	3
☑ 17	FLOAT32	FLOAT32	Power:10^3 W-L2 phase value	1-171	4	3
☑ 18	FLOAT32	FLOAT32	Power:10^3 W-L3 phase value	1-179	4	3
☑ 19	FLOAT32	FLOAT32	Power:10^3 W	1-185	4	3
☐ 20	FLOAT32	FLOAT32	Power:10^3 W[U:1,T:0,S:0]	1-192	4	3
☐ 21	FLOAT32	FLOAT32	Power:10^3 W[U:2,T:0,S:0]	1-200	4	3
☑ 22	FLOAT32	FLOAT32	Power Factor	1-207	4	0
☑ 23	FLOAT32	FLOAT32	Frequency	1-214	4	0
☑ 24	INT64	UINT64	Energy:10^0 Wh	1-220	8	0

Now we click on the OK button. You should see the following result:



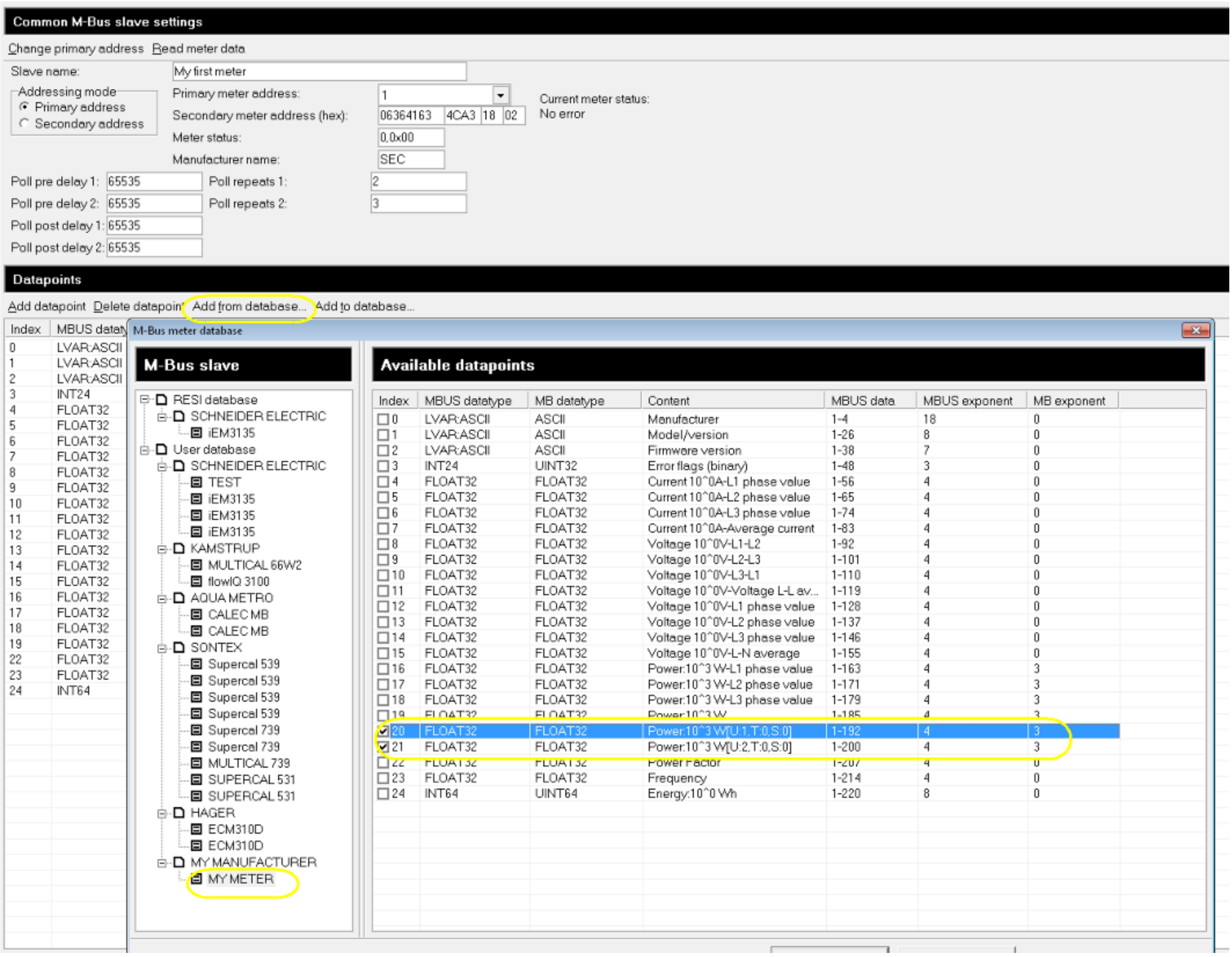
Click in the project tree on the meter MY METER to change the individual parameters for the selected meter. Adopt the addressing mode, the meter name and the other parameters of the meter, so that your gateway can communicate with this meter.



Index	MBUS dataty...	MB datatype	Content	MBUS data	MBUS size	MBUS exponent	MB exponent
0	LVAR:ASCII	ASCII	Manufacturer	1-4	18	0	0
1	LVAR:ASCII	ASCII	Model/version	1-26	8	0	0
2	LVAR:ASCII	ASCII	Firmware version	1-38	7	0	0
3	INT24	UINT32	Error flags (binary)	1-48	3	0	0
4	FLOAT32	FLOAT32	Current 10^0A-L1 phase value	1-56	4	0	0
5	FLOAT32	FLOAT32	Current 10^0A-L2 phase value	1-65	4	0	0
6	FLOAT32	FLOAT32	Current 10^0A-L3 phase value	1-74	4	0	0
7	FLOAT32	FLOAT32	Current 10^0A-Average current	1-83	4	0	0
8	FLOAT32	FLOAT32	Voltage 10^0V-L1-L2	1-92	4	0	0
9	FLOAT32	FLOAT32	Voltage 10^0V-L2-L3	1-101	4	0	0
10	FLOAT32	FLOAT32	Voltage 10^0V-L3-L1	1-110	4	0	0
11	FLOAT32	FLOAT32	Voltage 10^0V-Voltage L-L average	1-119	4	0	0
12	FLOAT32	FLOAT32	Voltage 10^0V-L1 phase value	1-128	4	0	0
13	FLOAT32	FLOAT32	Voltage 10^0V-L2 phase value	1-137	4	0	0
14	FLOAT32	FLOAT32	Voltage 10^0V-L3 phase value	1-146	4	0	0
15	FLOAT32	FLOAT32	Voltage 10^0V-L-N average	1-155	4	0	0
16	FLOAT32	FLOAT32	Power:10^3 W-L1 phase value	1-163	4	3	0
17	FLOAT32	FLOAT32	Power:10^3 W-L2 phase value	1-171	4	3	0
18	FLOAT32	FLOAT32	Power:10^3 W-L3 phase value	1-179	4	3	0
19	FLOAT32	FLOAT32	Power:10^3 W	1-185	4	3	0
22	FLOAT32	FLOAT32	Power Factor	1-207	4	0	0
23	FLOAT32	FLOAT32	Frequency	1-214	4	0	0
24	INT64	UINT64	Energy:10^0 Wh	1-220	8	0	0

43.13 HOWTO add meter datapoints to an existing meter

You can add individual datapoints to an existing meter in your configuration. First select the meter in your project tree, where you want to add a datapoint. Then click on the button Add from database and select a meter template with the desired datapoints. In our case we select the meter MY METER. Then we deselect all datapoint by doing a right click on the data grid and choose the menu Deselect all. Then we select the two datapoints and click the OK button.



You will get the following result:

Common M-Bus slave settings

Change primary address Read meter data

Slave name: My first meter

Addressing mode
- (•) Primary address
- ( ) Secondary address

Primary meter address: 1

Secondary meter address (hex): 06364163 | 4CA3 | 18 | 02

Meter status: 0,0x00

Manufacturer name: SEC

Current meter status:
No error

Poll pre delay 1: 65535 Poll repeats 1: 2

Poll pre delay 2: 65535 Poll repeats 2: 3

Poll post delay 1: 65535

Poll post delay 2: 65535

Datapoints

Add datapoint Delete datapoint Add from database... Add to database...

Index	MBUS dataty...	MB datatype	Content	MBUS data	MBUS size	MBUS exponent	MB exponent
0	LVAR:ASCII	ASCII	Manufacturer	1-4	18	0	0
1	LVAR:ASCII	ASCII	Model/version	1-26	8	0	0
2	LVAR:ASCII	ASCII	Firmware version	1-38	7	0	0
3	INT24	UINT32	Error flags (binary)	1-48	3	0	0
4	FLOAT32	FLOAT32	Current 10^0A-L1 phase value	1-56	4	0	0
5	FLOAT32	FLOAT32	Current 10^0A-L2 phase value	1-65	4	0	0
6	FLOAT32	FLOAT32	Current 10^0A-L3 phase value	1-74	4	0	0
7	FLOAT32	FLOAT32	Current 10^0A-Average current	1-83	4	0	0
8	FLOAT32	FLOAT32	Voltage 10^0V-L1-L2	1-92	4	0	0
9	FLOAT32	FLOAT32	Voltage 10^0V-L2-L3	1-101	4	0	0
10	FLOAT32	FLOAT32	Voltage 10^0V-L3-L1	1-110	4	0	0
11	FLOAT32	FLOAT32	Voltage 10^0V-Voltage L-L average	1-119	4	0	0
12	FLOAT32	FLOAT32	Voltage 10^0V-L1 phase value	1-128	4	0	0
13	FLOAT32	FLOAT32	Voltage 10^0V-L2 phase value	1-137	4	0	0
14	FLOAT32	FLOAT32	Voltage 10^0V-L3 phase value	1-146	4	0	0
15	FLOAT32	FLOAT32	Voltage 10^0V-L-N average	1-155	4	0	0
16	FLOAT32	FLOAT32	Power:10^3 W-L1 phase value	1-163	4	3	0
17	FLOAT32	FLOAT32	Power:10^3 W-L2 phase value	1-171	4	3	0
18	FLOAT32	FLOAT32	Power:10^3 W-L3 phase value	1-179	4	3	0
19	FLOAT32	FLOAT32	Power:10^3 W	1-185	4	3	0
22	FLOAT32	FLOAT32	Power Factor	1-207	4	0	0
23	FLOAT32	FLOAT32	Frequency	1-214	4	0	0
24	INT64	UINT64	Energy:10^0 Wh	1-220	8	0	0
20	FLOAT32	FLOAT32	Power:10^3 W[U:1,T:0,S:0]	1-192	4	3	0
21	FLOAT32	FLOAT32	Power:10^3 W[U:2,T:0,S:0]	1-200	4	3	0

Then we select the datapoints 22-24 and delete then from the meter setup, by clicking on the button Delete datapoints. Now our new setup is finished and can be downloaded into the gateway.

Index	MBUS dataty...	MB datatype	Content	MBUS data	MBUS size	MBUS exponent	MB exp
0	LVAR:ASCII	ASCII	Manufacturer	1-4	18	0	0
1	LVAR:ASCII	ASCII	Model/version	1-26	8	0	0
2	LVAR:ASCII	ASCII	Firmware version	1-38	7	0	0
3	INT24	UINT32	Error flags (binary)	1-48	3	0	0
4	FLOAT32	FLOAT32	Current 10^0A-L1 phase value	1-56	4	0	0
5	FLOAT32	FLOAT32	Current 10^0A-L2 phase value	1-65	4	0	0
6	FLOAT32	FLOAT32	Current 10^0A-L3 phase value	1-74	4	0	0
7	FLOAT32	FLOAT32	Current 10^0A-Average current	1-83	4	0	0
8	FLOAT32	FLOAT32	Voltage 10^0V-L1-L2	1-92	4	0	0
9	FLOAT32	FLOAT32	Voltage 10^0V-L2-L3	1-101	4	0	0
10	FLOAT32	FLOAT32	Voltage 10^0V-L3-L1	1-110	4	0	0
11	FLOAT32	FLOAT32	Voltage 10^0V-Voltage L-L average	1-119	4	0	0
12	FLOAT32	FLOAT32	Voltage 10^0V-L1 phase value	1-128	4	0	0
13	FLOAT32	FLOAT32	Voltage 10^0V-L2 phase value	1-137	4	0	0
14	FLOAT32	FLOAT32	Voltage 10^0V-L3 phase value	1-146	4	0	0
15	FLOAT32	FLOAT32	Voltage 10^0V-L-N average	1-155	4	0	0
16	FLOAT32	FLOAT32	Power:10^3 W-L1 phase value	1-163	4	3	0
17	FLOAT32	FLOAT32	Power:10^3 W-L2 phase value	1-171	4	3	0
18	FLOAT32	FLOAT32	Power:10^3 W-L3 phase value	1-179	4	3	0
19	FLOAT32	FLOAT32	Power:10^3 W	1-185	4	3	0
20	FLOAT32	FLOAT32	Power:10^3 W[U:1,T:0,S:0]	1-192	4	3	0
21	FLOAT32	FLOAT32	Power:10^3 W[U:2,T:0,S:0]	1-200	4	3	0

43.14 Table of MBUS data types

The following table shows, which MBUS data types are used and how they are processed by the gateway:

MBUS DATATYPE	SIZE	BYTE ORDER	DESCRIPTION
BCD2	8 bits 1 byte	Decimal digits HL → 0xHL	Defines an 8 bit unsigned integer value in the range of 0 to 99 stored as BCD number with encoding: Byte 0xHL: Bits 7-4: H: UPPER DIGIT as hex value 0x0 to 0x9 Bits 3-0: L: LOWER DIGIT as hex value 0x0 to 0x9 So hex value 0x12 means decimal value 1*10+2=12 in decimal
NOTE to BCD numbers: According to the MBUS standard not only positive BCD numbers are handled by our gateway. Due to the encoding of BCD numbers as 4 bit hexadecimal characters for each digit, only the hexadecimal numbers 0x0 to 0x9 are used for the decimal representation. Therefore the hexadecimal digits 0xA to 0xF are not used to represent a BCD number. If the leading digit of the BCD number encodes a 0xF, this stands for a negative sign. So the number 0x0123 means the decimal representation of +123, the number 0xF123 means the decimal representation of -123.			
BCD4	16 bits 2 byte	Decimal digits ABCD → 0xCD 0xAB	Defines a 16 bit unsigned integer value in the range of 0 to 9999 stored as BCD number with encoding: First byte 00xCD: Bits 7-4: C: DIGIT*10 as hex value 0x0 to 0x9 Bits 3-0: D: DIGIT*1 as hex value 0x0 to 0x9 Second byte 00xAB: Bits 7-4: A: DIGIT*1000 as hex value 0x0 to 0x9 Bits 3-0: B: DIGIT*100 as hex value 0x0 to 0x9 So hex value 0x1234 means decimal value 1*1000+2*100+3*10+4=1234 in decimal
BCD6	24 bits 3 byte	Decimal digits ABCDEF → 0xEF 0xCD 0xAB	Defines a 24 bit unsigned integer value in the range of 0 to 999999 stored as BCD number with encoding: First byte 00xEF: Bits 7-4: F: DIGIT*10 as hex value 0x0 to 0x9 Bits 3-0: E: DIGIT*1 as hex value 0x0 to 0x9 Second byte 00xCD: Bits 7-4: C: DIGIT*1000 as hex value 0x0 to 0x9 Bits 3-0: D: DIGIT*100 as hex value 0x0 to 0x9 Third byte 00xAB: Bits 7-4: A: DIGIT*100000 as hex value 0x0 to 0x9 Bits 3-0: B: DIGIT*10000 as hex value 0x0 to 0x9 So hex value 0x123456 means decimal value 1*100000+2*10000+3*1000+4*100+5*10+6=123456 in decimal
BCD8	32 bits 4 byte	Decimal digits ABCDEFGH → 0xGH 0xEF 0xCD 0xAB	Defines a 32 bit unsigned integer value in the range of 0 to 99999999 stored as BCD number with encoding: First byte 00xGH: Bits 7-4: G: DIGIT*10 as hex value 0x0 to 0x9 Bits 3-0: H: DIGIT*1 as hex value 0x0 to 0x9 Second byte 00xEF: Bits 7-4: E: DIGIT*1000 as hex value 0x0 to 0x9 Bits 3-0: F: DIGIT*100 as hex value 0x0 to 0x9 Third byte 00xCD: Bits 7-4: C: DIGIT*100000 as hex value 0x0 to 0x9 Bits 3-0: D: DIGIT*10000 as hex value 0x0 to 0x9 Fourth byte 00xAB: Bits 7-4: A: DIGIT*10000000 as hex value 0x0 to 0x9 Bits 3-0: B: DIGIT* 1000000 as hex value 0x0 to 0x9 So hex value 0x12345678 means decimal value 1*10000000+2*1000000+3*100000+4*10000+5*1000+6*100+7*10+8=12345678 in decimal

MBUS DATATYPE	SIZE	BYTE ORDER	DESCRIPTION
BCD12	48 bits 6 byte	Decimal digits ABCDEFGHIJKL → 0xKL 0xIJ 0xGH 0xEF 0xCD 0xAB	Defines a 48 bit unsigned integer value in the range of 0 to 999999999999 stored as BCD number with encoding: First byte 00xKL: Bits 7-4: K: DIGIT*10 as hex value 0x0 to 0x9 Bits 3-0: L: DIGIT*1 as hex value 0x0 to 0x9 Second byte 00xIJ: Bits 7-4: I: DIGIT*1000 as hex value 0x0 to 0x9 Bits 3-0: J: DIGIT*100 as hex value 0x0 to 0x9 Third byte 00xGH: Bits 7-4: G: DIGIT*100000 as hex value 0x0 to 0x9 Bits 3-0: H: DIGIT*10000 as hex value 0x0 to 0x9 Fourth byte 00xEF: Bits 7-4: E: DIGIT*10000000 as hex value 0x0 to 0x9 Bits 3-0: F: DIGIT*1000000 as hex value 0x0 to 0x9 Fifth byte 00xCD: Bits 7-4: C: DIGIT*1000000000 as hex value 0x0 to 0x9 Bits 3-0: D: DIGIT*100000000 as hex value 0x0 to 0x9 Sixth byte 00xAB: Bits 7-4: A: DIGIT*100000000000 as hex value 0x0 to 0x9 Bits 3-0: B: DIGIT*10000000000 as hex value 0x0 to 0x9 So hex value 0x123456789012 means decimal value 1*100000000000+2*10000000000+ 3*1000000000+4*100000000+ 5*10000000+6*1000000+7*100000+8*10000+ 9*1000+0*100+1*10+2*1 =123456789012 in decimal
SINT8	8 bits 1 byte	none	Defines a 8 bit signed integer value in the range of -128 to +127 or 0x80 to 0x7F First byte: Bit 7: Sign Bits 6-0: integer value
SINT16	16 bits 2 byte	0x1234 → 0x34 0x12	Defines a 16 bit signed integer value in the range of -32768 to +32767 or 0x8000 to 0x7FFF First byte: Bits 7-0 Second byte: Bit 15: Sign Bits 15-8
SINT24	24 bits 3 byte	0x123456 → 0x56 0x34 0x12	Defines a 24 bit signed integer value in the range of -8.388.608 to +8.388.608 or 0x80.0000 to 0x7F.FFFF First byte: Bits 7-0 Second byte: Bits 15-8 Third byte: Bit 23: Sign Bits 22-16
SINT32	32 bits 4 byte	0x12345678 → 0x78 0x56 0x34 0x12	Defines a 32 bit signed integer value in the range of -4.294.967.296 to 4.294.967.295 or 0x8000.0000 to 0x7FFF.FFFF First byte: Bits 7-0 Second byte: Bits 15-8 Third byte: Bits 23-16 Fourth byte: Bit 31: Sign Bits 30-24
SINT48	48 bits 6 byte	0x1234567890 → 0x90 0x78 0x56 0x34 0x12	Defines a 32 bit signed integer value in the range of -140.737.488.355.328 to +140.737.488.355.327 or 0x8000.0000.0000 to 0x7FFF.FFFF.FFFF First byte: Bits 7-0 Second byte: Bits 15-8 Third byte: Bits 23-16 Fourth byte: Bits 31-24 Fifth byte: Bits 39-32 Sixth byte: Bit 47: Sign Bits 46-40
SINT64	64 bits 8 byte	0x12345678 90ABCDEF → 0xEF 0xCD 0xAB 0x90 0x78 0x56 0x34 0x12	Defines a 32 bit signed integer value in the range of -9.223.372.036.854.775.808 to +9.223.372.036.854.775.807 or 0x8000.0000.0000.0000 to 0x7FFF.FFFF.FFFF.FFFF First byte: Bits 7-0 Second byte: Bits 15-8 Third byte: Bits 23-16 Fourth byte: Bits 31-24 Fifth byte: Bits 39-32 Sixth byte: Bits 47-40 Seventh byte: Bits 55-48 Eight byte: Bit 63: Sign Bits 62-56

MBUS DATATYPE	SIZE	BYTE ORDER	DESCRIPTION
FLOAT32	32 bits 4 byte	0x40490FDA → 0xDA 0x0F 0x49 0x40	Defines a 32 bit float value in the range of $\pm 1.4 \cdot 10^{-45}$ to $\pm 3.403 \cdot 10^{38}$. A mantissa of 23 bits and an exponent of 8 bits are used. The value can store 7 to 8 digits after the comma. Fraction F:=Bits 0..22 Exponent E:=Bits 30..23 Sigh S::= Bit 31 First byte: Fraction F Bits 7-0 Second byte: Fraction F Bits 15-8 Third byte: Exponent E Bit 23 Fraction F Bits 22-16 Fourth byte: Sign S Bit 31 Exponent Bits 30-24
DATE & TIME TYPE F	32 bits 4 byte	0x12345678 → 0x78 0x56 0x34 0x12	Defines a 32 bit value interpreted as date & time Minutes: Bits 5-0 → 0..59 Hour: Bits 12-8 → 0..23 Day: Bits 20-16 → 1..31 Month: Bits 27-23 → 1..12 Year: Bits 31-28,23-21 → 0..99 Invalid: Bit 7: =0 valid, =1: invalid Summertime Bit 15 =0 standard time, =1 summer time Reserved Bit 6 =0 Reserved Bit 13 =0 Reserved Bit 15 =0
DATE TYPE G	16 bits 2 byte	0x1234 → 0x34 0x12	Defines a 16 bit value interpreted as date Day: Bits 4-0 → 1..31 Month: Bits 11-8 → 1..12 Year: Bits 15-12,7-5 → 0..99
VARIABLE LENGTH	n*8 bits n bytes	Byte n-1 Byte n-2 … Byte 2 Byte 1 Byte 0	Defines a variable length field with n bytes of data. First byte: data[n-1] Second byte: data[n-2] … n-1. byte: data[1] n. byte (last byte): data[0] The length byte defines the representation of the variable length data field: LEN=0x00..0xBF: ASCII string LEN=0xC0..0xCF: positive BCD number with (LEN-0xC0)*2 digits LEN=0xD0..0xDF: negative BCD number with (LEN-0xD0)*2 digits LEN=0xE0..0xEF: integer number with (LEN-0xE0) bytes LEN=0xF0..0xFA: float number with (LEN-0xF0) bytes LEN=0xFB..0xFF: reserved

43.15 Table of MODBUS data types

The following table shows, which MODBUS data types are used and how they are processed by the gateway:

MODBUS DATATYPE	SIZE	WORD ORDER	DESCRIPTION
UINT16	16 bits 1 register	none	Defines a 16 bit unsigned integer value in the range of 0 to 65535 or 0x0000 to 0xFFFF
SINT16	16 bits 1 register	none	Defines a 16 bit signed integer value in the range of -32768 to +32767 or 0x8000 to 0x7FFF
UINT32	32 bits 2 register	0:High Word 1:Low Word	Defines a 32 bit unsigned integer value in the range of 0 to 4.294.967.295 or 0x00000000 to 0xFFFFFFFF
SINT32	32 bits 2 register	0:High Word 1:Low Word	Defines a 32 bit signed integer value in the range of −2.147.483.648 to +2.147.483.647or 0x80000000 to 0x7FFFFFFF
UINT32R	32 bits 2 register	0:Low Word 1:High Word	Defines a 32 bit unsigned integer value in the range of 0 to 4.294.967.295 or 0x00000000 to 0xFFFFFFFF with reverse word order
SINT32R	32 bits 2 register	0:Low Word 1:High Word	Defines a 32 bit signed integer value in the range of −2.147.483.648 to +2.147.483.647or 0x80000000 to 0x7FFFFFFF with reverse word order
FLOAT32	32 bits 2 register	0:High Word 1:Low Word	Defines a 32 bit float value in the range of $\pm 1.4 \cdot 10^{-45}$ to $\pm 3.403 \cdot 10^{38}$. A mantissa of 23 bits and an exponent of 8 bits are used. The value can store 7 to 8 digits after the comma.
FLOAT32R	32 bits 2 register	0:Low Word 1:High Word	Defines a 32 bit float value in the range of $\pm 1.4 \cdot 10^{-45}$ to $\pm 3.403 \cdot 10^{38}$. A mantissa of 23 bits and an exponent of 8 bits are used. The value can store 7 to 8 digits after the comma. The two 16 bit words are stored in reverse order.
DOUBLE64	64 bits 4 register	0:Highest Word 1:Higher Word 2:Lower Word 3:Lowest Word	Defines a 64 bit float value in the range of $\pm 4.24 \cdot 10^{-324}$ to $\pm 1{,}798 \cdot 10^{308}$. A mantissa of 52 bits and an exponent of 11 bits are used. The value can store 15 to 16 digits after the comma.
DOUBLE64R	64 bits 4 register	0:Lowest Word 1:Lower Word 2:Higher Word 3:Highest Word	Defines a 64 bit float value in the range of $\pm 4.24 \cdot 10^{-324}$ to $\pm 1{,}798 \cdot 10^{308}$. A mantissa of 52 bits and an exponent of 11 bits are used. The value can store 15 to 16 digits after the comma. The four 16 bit words are stored in reverse order.
ASCII	2*n*8 bits n register	0:Highest Word 1:Higher Word …. n-1:Lower Word n: Lowest Word	Defines a byte array with ASCII characters stored in 16 bit words. The ASCII string is terminated with a trailing 0x00 byte. To achieve word alignment a second 0x00 character can be stuffed at the end of the string. The low byte of the first word holds the first ASCII character, The high byte of the first word holds the second ASCII character and so on.
ASCIIR	2*n*8 bits n register	0:Lowest Word 1:Lower Word …. n-1:Higher Word n: HIghest Word	Defines a byte array with ASCII characters stored in 16 bit words. The ASCII string is terminated with a trailing 0x00 byte. To achieve word alignment a second 0x00 character can be stuffed at the end of the string. The low byte of the last word holds the first ASCII character, The high byte of the last word holds the second ASCII character and so on.
DATE TIME TYPE F	32 bits 2 register	0:High Word 1:Low Word	Defines a 32 bit value interpreted as date & time Minutes: Bits 5-0 → 0..59 Hour: Bits 12-8 → 0..23 Day: Bits 20-16 → 1..31 Month: Bits 27-23 → 1..12 Year: Bits 31-28,23-21 → 0..99 Invalid: Bit 7: =0 valid, =1: invalid Summertime Bit 15 =0 standard time, =1 summer time Reserved Bit 6 =0 Reserved Bit 13 =0 Reserved Bit 15 =0
DATE TIME TYPE FR	32 bits 2 register	0:Low Word 1:High Word	Defines a 32 bit value interpreted as date & time Minutes: Bits 5-0 → 0..59 Hour: Bits 12-8 → 0..23 Day: Bits 20-16 → 1..31 Month: Bits 27-23 → 1..12 Year: Bits 31-28,23-21 → 0..99 Invalid: Bit 7: =0 valid, =1: invalid Summertime Bit 15 =0 standard time, =1 summer time Reserved Bit 6 =0 Reserved Bit 13 =0 Reserved Bit 15 =0
DATE TYP G	16 bits 1 register	none	Defines a 16 bit value interpreted as date Day: Bits 4-0 → 1..31 Month: Bits 11-8 → 1..12 Year: Bits 15-12,7-5 → 0..99

MODBUS DATATYPE	**SIZE**	**WORD ORDER**	**DESCRIPTION**
BUFFER	2*n*8 bits n register	0:Highest Word 1:Higher Word …. n-1:Lower Word n: Lowest Word	Defines a byte array stored in 16 bit words. To achieve word alignment an additional 0x00 byte can be stuffed at the end of the byte array. The low byte of the first word holds the first byte, The high byte of the first word holds the second byte and so on.
BUFFERR	2*n*8 bits n register	0:Lowest Word 1:Lower Word …. n-1:Higher Word n: HIghest Word	Defines a byte array stored in 16 bit words. To achieve word alignment an additional 0x00 byte can be stuffed at the end of the byte array. The low byte of the last word holds the first byte, The high byte of the last word holds the second byte and so on.

43.16 HOW the MBUS to MODBUS mapping works

The following section describes, how the internal process of mapping MBUS to MODBUS datapoints in done by the gateway. For that we take a SCHNEIDER electrical meter (primary address 4) as a sample to describe the main principle of the conversion process.

In this case, the meter answers with only one data frame to the MBUS request of the gateway (All bytes in hexadecimal):

GATEWAY to METER: Send slave init to primary address 4
10 40 04 44 16
METER to GATEWAY: Send OK
E5
GATEWAY to METER: Send REQ_UD2 frame to metering
10 7B 04 7F 16
METER TO GATEWAY: Send variable data frame as answered
68 F4 F4 68 08 04 72 63 41 36 06 A3 4C 18 02 C8 00 00 00 0D FD 0A 12 63 69 72 74 63 65 6C 45 20 72 65 64 69 65 6E 68 63 53 0D FD 0C 08 20 35 33 31 33 4D 45 69 0D FD 0E 07 32 30 30 2E 34 2E 31 03 FD 17 00 00 00 05 FD DC FF 01 00 00 00 00 05 FD DC FF 02 00 00 C0 FF 05 FD DC FF 03 00 00 C0 FF 05 FD DC FF 00 00 00 00 00 05 FD C9 FF 05 00 00 C0 FF 05 FD C9 FF 06 00 00 C0 FF 05 FD C9 FF 07 00 00 C0 FF 05 FD C9 FF 08 00 00 C0 FF 05 FD C9 FF 01 8E 8B 64 43 05 FD C9 FF 02 00 00 C0 FF 05 FD C9 FF 03 00 00 C0 FF 05 FD C9 FF 04 8E 8B 64 43 05 AE FF 01 00 00 00 00 05 AE FF 02 00 00 C0 FF 05 AE FF 03 00 00 C0 FF 05 2E 00 00 00 00 85 40 2E 00 00 00 00 85 80 40 2E 00 00 00 00 05 FF 0A 00 00 C0 FF 05 FF 0B 29 F1 47 42 07 03 19 00 00 00 00 00 00 00 1F 1D 16

What data is in the MBUS frame? This is the interpretation of the complete received MBUS frame by our software:

```
==========================================================================================================================
MBUS:FRAME TYPE:0x72:12 BYTE HEADER+VARIABLE DATA
==========================================================================================================================
MBUS:HEADER12:ID:104218979,0x06364163
MBUS:HEADER12:MANUFACTURER:19619,0x4CA3,SEC
MBUS:HEADER12:VERSION:24,0x18
MBUS:HEADER12:MEDIUM:2,0x02
MBUS:HEADER12:MEDIUM:Electricity
MBUS:HEADER12:ACCESS:200,0xC8
MBUS:HEADER12:STATUS:0,0x00
MBUS:HEADER12:STATUS:NO ERROR
MBUS:HEADER12:SIGNATURE:0,0x0000

==========================================================================================================================
MBUS:VARIABLE DATA
==========================================================================================================================
--------------------------------------------------------------------------------------------------------------------------
MBUS:VARIABLE DATA:DATARECORD:1:DATABLOCK:0
--------------------------------------------------------------------------------------------------------------------------
MBUS:VARIABLE DATA:[0]DIF:0D
MBUS:VARIABLE DATA:DIF:BIT 7:EXTENSION BIT:0
MBUS:VARIABLE DATA:DIF:BIT 6:LSB STORAGE NUMBER:0
MBUS:VARIABLE DATA:DIF:BIT 5-4:FUNCTION FIELD:0:00:Instantaneous value
MBUS:VARIABLE DATA:DIF:BIT 3-0:DATA FIELD:D:D:1101:variable length
MBUS:VARIABLE DATA:[1]VIF:FD
MBUS:VARIABLE DATA:VIF:BIT 7:EXTENSION BIT:1
MBUS:VARIABLE DATA:VIF:BIT 6-0:UNIT+MULTIPLIER:7D:LINEAR VIF EXTENSION
MBUS:VARIABLE DATA:[2]VIFE:0A
MBUS:VARIABLE DATA:VIFE:BIT 7:EXTENSION BIT:0
MBUS:VARIABLE DATA:VIFE:BIT 6-0:UNIT+MULTIPLIER:0A:SECONDARY VIF (8.4.4) a.
SUB VIF:MANUFACTURER (as in fixed header)
MBUS:VARIABLE DATA:ASCII:18
MBUS:VARIABLE DATA:[3]-[21]DATABLOCK:LENGTH:12,18
MBUS:VARIABLE DATA:DATA BLOCK:DATA:[63][69][72][74][63][65][6C][45][20][72][65][64][69][65][6E][68][63][53]
MBUS:VARIABLE DATA:DATA BLOCK:DATA:ASCII:Schneider Electric
DIFFUNCTIONTEXT:INSTANTANEUS VALUE
DIFTEXT:LVAR:ASCII(18 bytes)
VIFTEXT:Manufacturer
--------------------------------------------------------------------------------------------------------------------------
MBUS:VARIABLE DATA:DATARECORD:1:DATABLOCK:1
--------------------------------------------------------------------------------------------------------------------------
MBUS:VARIABLE DATA:[22]DIF:0D
MBUS:VARIABLE DATA:DIF:BIT 7:EXTENSION BIT:0
MBUS:VARIABLE DATA:DIF:BIT 6:LSB STORAGE NUMBER:0
MBUS:VARIABLE DATA:DIF:BIT 5-4:FUNCTION FIELD:0:00:Instantaneous value
MBUS:VARIABLE DATA:DIF:BIT 3-0:DATA FIELD:D:D:1101:variable length
MBUS:VARIABLE DATA:[23]VIF:FD
MBUS:VARIABLE DATA:VIF:BIT 7:EXTENSION BIT:1
MBUS:VARIABLE DATA:VIF:BIT 6-0:UNIT+MULTIPLIER:7D:LINEAR VIF EXTENSION
MBUS:VARIABLE DATA:[24]VIFE:0C
MBUS:VARIABLE DATA:VIFE:BIT 7:EXTENSION BIT:0
MBUS:VARIABLE DATA:VIFE:BIT 6-0:UNIT+MULTIPLIER:0C:SECONDARY VIF (8.4.4) a.
SUB VIF:MODEL/VERSION
```

```
MBUS:VARIABLE DATA:ASCII:8
MBUS:VARIABLE DATA:[25]-[33]DATABLOCK:LENGTH:08,8
MBUS:VARIABLE DATA:DATA BLOCK:DATA:[20][35][33][31][33][4D][45][69]
MBUS:VARIABLE DATA:DATA BLOCK:DATA:ASCII:iEM3135
DIFFUNCTIONTEXT:INSTANTANEUS VALUE
DIFTEXT:LVAR:ASCII(8 bytes)
VIFTEXT:Model/version
------------------------------------------------------------------------------------------------------------------------
MBUS:VARIABLE DATA:DATARECORD:1:DATABLOCK:2
------------------------------------------------------------------------------------------------------------------------
MBUS:VARIABLE DATA:[34]DIF:0D
MBUS:VARIABLE DATA:DIF:BIT 7:EXTENSION BIT:0
MBUS:VARIABLE DATA:DIF:BIT 6:LSB STORAGE NUMBER:0
MBUS:VARIABLE DATA:DIF:BIT 5-4:FUNCTION FIELD:0:00:Instantaneous value
MBUS:VARIABLE DATA:DIF:BIT 3-0:DATA FIELD:D:D:1101:variable length
MBUS:VARIABLE DATA:[35]VIF:FD
MBUS:VARIABLE DATA:VIF:BIT 7:EXTENSION BIT:1
MBUS:VARIABLE DATA:VIF:BIT 6-0:UNIT+MULTIPLIER:7D:LINEAR VIF EXTENSION
MBUS:VARIABLE DATA:[36]VIFE:0E
MBUS:VARIABLE DATA:VIFE:BIT 7:EXTENSION BIT:0
MBUS:VARIABLE DATA:VIFE:BIT 6-0:UNIT+MULTIPLIER:0E:SECONDARY VIF (8.4.4) a.
SUB VIF:FIRMWARE VERSION
MBUS:VARIABLE DATA:ASCII:7
MBUS:VARIABLE DATA:[37]-[44]DATABLOCK:LENGTH:07,7
MBUS:VARIABLE DATA:DATA BLOCK:DATA:[32][30][30][2E][34][2E][31]
MBUS:VARIABLE DATA:DATA BLOCK:DATA:ASCII:1.4.002
DIFFUNCTIONTEXT:INSTANTANEUS VALUE
DIFTEXT:LVAR:ASCII(7 bytes)
VIFTEXT:Firmware version
------------------------------------------------------------------------------------------------------------------------
MBUS:VARIABLE DATA:DATARECORD:1:DATABLOCK:3
------------------------------------------------------------------------------------------------------------------------
MBUS:VARIABLE DATA:[45]DIF:03
MBUS:VARIABLE DATA:DIF:BIT 7:EXTENSION BIT:0
MBUS:VARIABLE DATA:DIF:BIT 6:LSB STORAGE NUMBER:0
MBUS:VARIABLE DATA:DIF:BIT 5-4:FUNCTION FIELD:0:00:Instantaneous value
MBUS:VARIABLE DATA:DIF:BIT 3-0:DATA FIELD:3:3:0011:24 Bit Integer
MBUS:VARIABLE DATA:[46]VIF:FD
MBUS:VARIABLE DATA:VIF:BIT 7:EXTENSION BIT:1
MBUS:VARIABLE DATA:VIF:BIT 6-0:UNIT+MULTIPLIER:7D:LINEAR VIF EXTENSION
MBUS:VARIABLE DATA:[47]VIFE:17
MBUS:VARIABLE DATA:VIFE:BIT 7:EXTENSION BIT:0
MBUS:VARIABLE DATA:VIFE:BIT 6-0:UNIT+MULTIPLIER:17:SECONDARY VIF (8.4.4) a.
SUB VIF:ERROR FLAGS (BINARY)
MBUS:FIX DATA:[48]-[50]DATABLOCK:LENGTH:03,3
MBUS:FIX DATA:DATA BLOCK:DATA:[00][00][00]
DIFFUNCTIONTEXT:INSTANTANEUS VALUE
DIFTEXT:24BIT INT(3 bytes)
VIFTEXT:Error flags (binary)
------------------------------------------------------------------------------------------------------------------------
MBUS:VARIABLE DATA:DATARECORD:1:DATABLOCK:4
------------------------------------------------------------------------------------------------------------------------
MBUS:VARIABLE DATA:[51]DIF:05
MBUS:VARIABLE DATA:DIF:BIT 7:EXTENSION BIT:0
MBUS:VARIABLE DATA:DIF:BIT 6:LSB STORAGE NUMBER:0
MBUS:VARIABLE DATA:DIF:BIT 5-4:FUNCTION FIELD:0:00:Instantaneous value
MBUS:VARIABLE DATA:DIF:BIT 3-0:DATA FIELD:5:5:0101:32 Bit Real
MBUS:VARIABLE DATA:[52]VIF:FD
MBUS:VARIABLE DATA:VIF:BIT 7:EXTENSION BIT:1
MBUS:VARIABLE DATA:VIF:BIT 6-0:UNIT+MULTIPLIER:7D:LINEAR VIF EXTENSION
MBUS:VARIABLE DATA:[53]VIFE:DC
MBUS:VARIABLE DATA:VIFE:BIT 7:EXTENSION BIT:1
MBUS:VARIABLE DATA:VIFE:BIT 6-0:UNIT+MULTIPLIER:5C:SECONDARY VIF (8.4.4) a.
SUB VIF:Current 10^0A
MBUS:VARIABLE DATA:[54]VIFE:FF:MANUFACTURER SPECIFIC VIFE
MBUS:VARIABLE DATA:VIFE:BIT 7:EXTENSION BIT:1:MANUFACTURER SPECIFIC VIFE
MBUS:VARIABLE DATA:VIFE:BIT 6-0:UNIT+MULTIPLIER:7F:MANUFACTURER SPECIFIC VIFE
MBUS:VARIABLE DATA:[55]VIFE:01:MANUFACTURER SPECIFIC:SEC
MBUS:VARIABLE DATA:VIFE:BIT 7:EXTENSION BIT:0:MANUFACTURER SPECIFIC:SEC
MBUS:VARIABLE DATA:VIFE:BIT 6-0:UNIT+MULTIPLIER:01:MANUFACTURER SPECIFIC:SEC
VIF SEC:L1 phase value
MBUS:FIX DATA:[56]-[59]DATABLOCK:LENGTH:04,4
MBUS:FIX DATA:DATA BLOCK:DATA:[00][00][00][00]
DIFFUNCTIONTEXT:INSTANTANEUS VALUE
DIFTEXT:32BIT FLOAT(4 bytes)
VIFTEXT:Current 10^0A-L1 phase value
------------------------------------------------------------------------------------------------------------------------
MBUS:VARIABLE DATA:DATARECORD:1:DATABLOCK:5
------------------------------------------------------------------------------------------------------------------------
MBUS:VARIABLE DATA:[60]DIF:05
MBUS:VARIABLE DATA:DIF:BIT 7:EXTENSION BIT:0
MBUS:VARIABLE DATA:DIF:BIT 6:LSB STORAGE NUMBER:0
MBUS:VARIABLE DATA:DIF:BIT 5-4:FUNCTION FIELD:0:00:Instantaneous value
MBUS:VARIABLE DATA:DIF:BIT 3-0:DATA FIELD:5:5:0101:32 Bit Real
MBUS:VARIABLE DATA:[61]VIF:FD
MBUS:VARIABLE DATA:VIF:BIT 7:EXTENSION BIT:1
MBUS:VARIABLE DATA:VIF:BIT 6-0:UNIT+MULTIPLIER:7D:LINEAR VIF EXTENSION
MBUS:VARIABLE DATA:[62]VIFE:DC
MBUS:VARIABLE DATA:VIFE:BIT 7:EXTENSION BIT:1
MBUS:VARIABLE DATA:VIFE:BIT 6-0:UNIT+MULTIPLIER:5C:SECONDARY VIF (8.4.4) a.
SUB VIF:Current 10^0A
MBUS:VARIABLE DATA:[63]VIFE:FF:MANUFACTURER SPECIFIC VIFE
```

```
MBUS:VARIABLE DATA:VIFE:BIT 7:EXTENSION BIT:1:MANUFACTURER SPECIFIC VIFE
MBUS:VARIABLE DATA:VIFE:BIT 6-0:UNIT+MULTIPLIER:7F:MANUFACTURER SPECIFIC VIFE
MBUS:VARIABLE DATA:[64]VIFE:02:MANUFACTURER SPECIFIC:SEC
MBUS:VARIABLE DATA:VIFE:BIT 7:EXTENSION BIT:0:MANUFACTURER SPECIFIC:SEC
MBUS:VARIABLE DATA:VIFE:BIT 6-0:UNIT+MULTIPLIER:02:MANUFACTURER SPECIFIC:SEC
VIF SEC:L2 phase value
MBUS:FIX DATA:[65]-[68]DATABLOCK:LENGTH:04,4
MBUS:FIX DATA:DATA BLOCK:DATA:[00][00][C0][FF]
DIFFUNCTIONTEXT:INSTANTANEUS VALUE
DIFTEXT:32BIT FLOAT(4 bytes)
VIFTEXT:Current 10^0A-L2 phase value
---------------------------------------------------------------------------------------------------------------------
MBUS:VARIABLE DATA:DATARECORD:1:DATABLOCK:6
---------------------------------------------------------------------------------------------------------------------
MBUS:VARIABLE DATA:[69]DIF:05
MBUS:VARIABLE DATA:DIF:BIT 7:EXTENSION BIT:0
MBUS:VARIABLE DATA:DIF:BIT 6:LSB STORAGE NUMBER:0
MBUS:VARIABLE DATA:DIF:BIT 5-4:FUNCTION FIELD:0:00:Instantaneous value
MBUS:VARIABLE DATA:DIF:BIT 3-0:DATA FIELD:5:5:0101:32 Bit Real
MBUS:VARIABLE DATA:[70]VIF:FD
MBUS:VARIABLE DATA:VIF:BIT 7:EXTENSION BIT:1
MBUS:VARIABLE DATA:VIF:BIT 6-0:UNIT+MULTIPLIER:7D:LINEAR VIF EXTENSION
MBUS:VARIABLE DATA:[71]VIFE:DC
MBUS:VARIABLE DATA:VIFE:BIT 7:EXTENSION BIT:1
MBUS:VARIABLE DATA:VIFE:BIT 6-0:UNIT+MULTIPLIER:5C:SECONDARY VIF (8.4.4) a.
SUB VIF:Current 10^0A
MBUS:VARIABLE DATA:[72]VIFE:FF:MANUFACTURER SPECIFIC VIFE
MBUS:VARIABLE DATA:VIFE:BIT 7:EXTENSION BIT:1:MANUFACTURER SPECIFIC VIFE
MBUS:VARIABLE DATA:VIFE:BIT 6-0:UNIT+MULTIPLIER:7F:MANUFACTURER SPECIFIC VIFE
MBUS:VARIABLE DATA:[73]VIFE:03:MANUFACTURER SPECIFIC:SEC
MBUS:VARIABLE DATA:VIFE:BIT 7:EXTENSION BIT:0:MANUFACTURER SPECIFIC:SEC
MBUS:VARIABLE DATA:VIFE:BIT 6-0:UNIT+MULTIPLIER:03:MANUFACTURER SPECIFIC:SEC
VIF SEC:L3 phase value
MBUS:FIX DATA:[74]-[77]DATABLOCK:LENGTH:04,4
MBUS:FIX DATA:DATA BLOCK:DATA:[00][00][C0][FF]
DIFFUNCTIONTEXT:INSTANTANEUS VALUE
DIFTEXT:32BIT FLOAT(4 bytes)
VIFTEXT:Current 10^0A-L3 phase value
---------------------------------------------------------------------------------------------------------------------
MBUS:VARIABLE DATA:DATARECORD:1:DATABLOCK:7
---------------------------------------------------------------------------------------------------------------------
MBUS:VARIABLE DATA:[78]DIF:05
MBUS:VARIABLE DATA:DIF:BIT 7:EXTENSION BIT:0
MBUS:VARIABLE DATA:DIF:BIT 6:LSB STORAGE NUMBER:0
MBUS:VARIABLE DATA:DIF:BIT 5-4:FUNCTION FIELD:0:00:Instantaneous value
MBUS:VARIABLE DATA:DIF:BIT 3-0:DATA FIELD:5:5:0101:32 Bit Real
MBUS:VARIABLE DATA:[79]VIF:FD
MBUS:VARIABLE DATA:VIF:BIT 7:EXTENSION BIT:1
MBUS:VARIABLE DATA:VIF:BIT 6-0:UNIT+MULTIPLIER:7D:LINEAR VIF EXTENSION
MBUS:VARIABLE DATA:[80]VIFE:DC
MBUS:VARIABLE DATA:VIFE:BIT 7:EXTENSION BIT:1
MBUS:VARIABLE DATA:VIFE:BIT 6-0:UNIT+MULTIPLIER:5C:SECONDARY VIF (8.4.4) a.
SUB VIF:Current 10^0A
MBUS:VARIABLE DATA:[81]VIFE:FF:MANUFACTURER SPECIFIC VIFE
MBUS:VARIABLE DATA:VIFE:BIT 7:EXTENSION BIT:1:MANUFACTURER SPECIFIC VIFE
MBUS:VARIABLE DATA:VIFE:BIT 6-0:UNIT+MULTIPLIER:7F:MANUFACTURER SPECIFIC VIFE
MBUS:VARIABLE DATA:[82]VIFE:00:MANUFACTURER SPECIFIC:SEC
MBUS:VARIABLE DATA:VIFE:BIT 7:EXTENSION BIT:0:MANUFACTURER SPECIFIC:SEC
MBUS:VARIABLE DATA:VIFE:BIT 6-0:UNIT+MULTIPLIER:00:MANUFACTURER SPECIFIC:SEC
VIF SEC:Average current
MBUS:FIX DATA:[83]-[86]DATABLOCK:LENGTH:04,4
MBUS:FIX DATA:DATA BLOCK:DATA:[00][00][00][00]
DIFFUNCTIONTEXT:INSTANTANEUS VALUE
DIFTEXT:32BIT FLOAT(4 bytes)
VIFTEXT:Current 10^0A-Average current
---------------------------------------------------------------------------------------------------------------------
MBUS:VARIABLE DATA:DATARECORD:1:DATABLOCK:8
---------------------------------------------------------------------------------------------------------------------
MBUS:VARIABLE DATA:[87]DIF:05
MBUS:VARIABLE DATA:DIF:BIT 7:EXTENSION BIT:0
MBUS:VARIABLE DATA:DIF:BIT 6:LSB STORAGE NUMBER:0
MBUS:VARIABLE DATA:DIF:BIT 5-4:FUNCTION FIELD:0:00:Instantaneous value
MBUS:VARIABLE DATA:DIF:BIT 3-0:DATA FIELD:5:5:0101:32 Bit Real
MBUS:VARIABLE DATA:[88]VIF:FD
MBUS:VARIABLE DATA:VIF:BIT 7:EXTENSION BIT:1
MBUS:VARIABLE DATA:VIF:BIT 6-0:UNIT+MULTIPLIER:7D:LINEAR VIF EXTENSION
MBUS:VARIABLE DATA:[89]VIFE:C9
MBUS:VARIABLE DATA:VIFE:BIT 7:EXTENSION BIT:1
MBUS:VARIABLE DATA:VIFE:BIT 6-0:UNIT+MULTIPLIER:49:SECONDARY VIF (8.4.4) a.
SUB VIF:Voltage 10^0V
MBUS:VARIABLE DATA:[90]VIFE:FF:MANUFACTURER SPECIFIC VIFE
MBUS:VARIABLE DATA:VIFE:BIT 7:EXTENSION BIT:1:MANUFACTURER SPECIFIC VIFE
MBUS:VARIABLE DATA:VIFE:BIT 6-0:UNIT+MULTIPLIER:7F:MANUFACTURER SPECIFIC VIFE
MBUS:VARIABLE DATA:[91]VIFE:05:MANUFACTURER SPECIFIC:SEC
MBUS:VARIABLE DATA:VIFE:BIT 7:EXTENSION BIT:0:MANUFACTURER SPECIFIC:SEC
MBUS:VARIABLE DATA:VIFE:BIT 6-0:UNIT+MULTIPLIER:05:MANUFACTURER SPECIFIC:SEC
VIF SEC:L1-L2
MBUS:FIX DATA:[92]-[95]DATABLOCK:LENGTH:04,4
MBUS:FIX DATA:DATA BLOCK:DATA:[00][00][C0][FF]
DIFFUNCTIONTEXT:INSTANTANEUS VALUE
DIFTEXT:32BIT FLOAT(4 bytes)
VIFTEXT:Voltage 10^0V-L1-L2
```

```
------------------------------------------------------------------------------------------------------------------------
MBUS:VARIABLE DATA:DATARECORD:1:DATABLOCK:9
------------------------------------------------------------------------------------------------------------------------
MBUS:VARIABLE DATA:[96]DIF:05
MBUS:VARIABLE DATA:DIF:BIT 7:EXTENSION BIT:0
MBUS:VARIABLE DATA:DIF:BIT 6:LSB STORAGE NUMBER:0
MBUS:VARIABLE DATA:DIF:BIT 5-4:FUNCTION FIELD:0:00:Instantaneous value
MBUS:VARIABLE DATA:DIF:BIT 3-0:DATA FIELD:5:5:0101:32 Bit Real
MBUS:VARIABLE DATA:[97]VIF:FD
MBUS:VARIABLE DATA:VIF:BIT 7:EXTENSION BIT:1
MBUS:VARIABLE DATA:VIF:BIT 6-0:UNIT+MULTIPLIER:7D:LINEAR VIF EXTENSION
MBUS:VARIABLE DATA:[98]VIFE:C9
MBUS:VARIABLE DATA:VIFE:BIT 7:EXTENSION BIT:1
MBUS:VARIABLE DATA:VIFE:BIT 6-0:UNIT+MULTIPLIER:49:SECONDARY VIF (8.4.4) a.
SUB VIF:Voltage 10^0V
MBUS:VARIABLE DATA:[99]VIFE:FF:MANUFACTURER SPECIFIC VIFE
MBUS:VARIABLE DATA:VIFE:BIT 7:EXTENSION BIT:1:MANUFACTURER SPECIFIC VIFE
MBUS:VARIABLE DATA:VIFE:BIT 6-0:UNIT+MULTIPLIER:7F:MANUFACTURER SPECIFIC VIFE
MBUS:VARIABLE DATA:[100]VIFE:06:MANUFACTURER SPECIFIC:SEC
MBUS:VARIABLE DATA:VIFE:BIT 7:EXTENSION BIT:0:MANUFACTURER SPECIFIC:SEC
MBUS:VARIABLE DATA:VIFE:BIT 6-0:UNIT+MULTIPLIER:06:MANUFACTURER SPECIFIC:SEC
VIF SEC:L2-L3
MBUS:FIX DATA:[101]-[104]DATABLOCK:LENGTH:04,4
MBUS:FIX DATA:DATA BLOCK:DATA:[00][00][C0][FF]
DIFFUNCTIONTEXT:INSTANTANEUS VALUE
DIFTEXT:32BIT FLOAT(4 bytes)
VIFTEXT:Voltage 10^0V-L2-L3
------------------------------------------------------------------------------------------------------------------------
MBUS:VARIABLE DATA:DATARECORD:1:DATABLOCK:10
------------------------------------------------------------------------------------------------------------------------
MBUS:VARIABLE DATA:[105]DIF:05
MBUS:VARIABLE DATA:DIF:BIT 7:EXTENSION BIT:0
MBUS:VARIABLE DATA:DIF:BIT 6:LSB STORAGE NUMBER:0
MBUS:VARIABLE DATA:DIF:BIT 5-4:FUNCTION FIELD:0:00:Instantaneous value
MBUS:VARIABLE DATA:DIF:BIT 3-0:DATA FIELD:5:5:0101:32 Bit Real
MBUS:VARIABLE DATA:[106]VIF:FD
MBUS:VARIABLE DATA:VIF:BIT 7:EXTENSION BIT:1
MBUS:VARIABLE DATA:VIF:BIT 6-0:UNIT+MULTIPLIER:7D:LINEAR VIF EXTENSION
MBUS:VARIABLE DATA:[107]VIFE:C9
MBUS:VARIABLE DATA:VIFE:BIT 7:EXTENSION BIT:1
MBUS:VARIABLE DATA:VIFE:BIT 6-0:UNIT+MULTIPLIER:49:SECONDARY VIF (8.4.4) a.
SUB VIF:Voltage 10^0V
MBUS:VARIABLE DATA:[108]VIFE:FF:MANUFACTURER SPECIFIC VIFE
MBUS:VARIABLE DATA:VIFE:BIT 7:EXTENSION BIT:1:MANUFACTURER SPECIFIC VIFE
MBUS:VARIABLE DATA:VIFE:BIT 6-0:UNIT+MULTIPLIER:7F:MANUFACTURER SPECIFIC VIFE
MBUS:VARIABLE DATA:[109]VIFE:07:MANUFACTURER SPECIFIC:SEC
MBUS:VARIABLE DATA:VIFE:BIT 7:EXTENSION BIT:0:MANUFACTURER SPECIFIC:SEC
MBUS:VARIABLE DATA:VIFE:BIT 6-0:UNIT+MULTIPLIER:07:MANUFACTURER SPECIFIC:SEC
VIF SEC:L3-L1
MBUS:FIX DATA:[110]-[113]DATABLOCK:LENGTH:04,4
MBUS:FIX DATA:DATA BLOCK:DATA:[00][00][C0][FF]
DIFFUNCTIONTEXT:INSTANTANEUS VALUE
DIFTEXT:32BIT FLOAT(4 bytes)
VIFTEXT:Voltage 10^0V-L3-L1
------------------------------------------------------------------------------------------------------------------------
MBUS:VARIABLE DATA:DATARECORD:1:DATABLOCK:11
------------------------------------------------------------------------------------------------------------------------
MBUS:VARIABLE DATA:[114]DIF:05
MBUS:VARIABLE DATA:DIF:BIT 7:EXTENSION BIT:0
MBUS:VARIABLE DATA:DIF:BIT 6:LSB STORAGE NUMBER:0
MBUS:VARIABLE DATA:DIF:BIT 5-4:FUNCTION FIELD:0:00:Instantaneous value
MBUS:VARIABLE DATA:DIF:BIT 3-0:DATA FIELD:5:5:0101:32 Bit Real
MBUS:VARIABLE DATA:[115]VIF:FD
MBUS:VARIABLE DATA:VIF:BIT 7:EXTENSION BIT:1
MBUS:VARIABLE DATA:VIF:BIT 6-0:UNIT+MULTIPLIER:7D:LINEAR VIF EXTENSION
MBUS:VARIABLE DATA:[116]VIFE:C9
MBUS:VARIABLE DATA:VIFE:BIT 7:EXTENSION BIT:1
MBUS:VARIABLE DATA:VIFE:BIT 6-0:UNIT+MULTIPLIER:49:SECONDARY VIF (8.4.4) a.
SUB VIF:Voltage 10^0V
MBUS:VARIABLE DATA:[117]VIFE:FF:MANUFACTURER SPECIFIC VIFE
MBUS:VARIABLE DATA:VIFE:BIT 7:EXTENSION BIT:1:MANUFACTURER SPECIFIC VIFE
MBUS:VARIABLE DATA:VIFE:BIT 6-0:UNIT+MULTIPLIER:7F:MANUFACTURER SPECIFIC VIFE
MBUS:VARIABLE DATA:[118]VIFE:08:MANUFACTURER SPECIFIC:SEC
MBUS:VARIABLE DATA:VIFE:BIT 7:EXTENSION BIT:0:MANUFACTURER SPECIFIC:SEC
MBUS:VARIABLE DATA:VIFE:BIT 6-0:UNIT+MULTIPLIER:08:MANUFACTURER SPECIFIC:SEC
VIF SEC:Voltage L-L average
MBUS:FIX DATA:[119]-[122]DATABLOCK:LENGTH:04,4
MBUS:FIX DATA:DATA BLOCK:DATA:[00][00][C0][FF]
DIFFUNCTIONTEXT:INSTANTANEUS VALUE
DIFTEXT:32BIT FLOAT(4 bytes)
VIFTEXT:Voltage 10^0V-Voltage L-L average
------------------------------------------------------------------------------------------------------------------------
MBUS:VARIABLE DATA:DATARECORD:1:DATABLOCK:12
------------------------------------------------------------------------------------------------------------------------
MBUS:VARIABLE DATA:[123]DIF:05
MBUS:VARIABLE DATA:DIF:BIT 7:EXTENSION BIT:0
MBUS:VARIABLE DATA:DIF:BIT 6:LSB STORAGE NUMBER:0
MBUS:VARIABLE DATA:DIF:BIT 5-4:FUNCTION FIELD:0:00:Instantaneous value
MBUS:VARIABLE DATA:DIF:BIT 3-0:DATA FIELD:5:5:0101:32 Bit Real
MBUS:VARIABLE DATA:[124]VIF:FD
MBUS:VARIABLE DATA:VIF:BIT 7:EXTENSION BIT:1
MBUS:VARIABLE DATA:VIF:BIT 6-0:UNIT+MULTIPLIER:7D:LINEAR VIF EXTENSION
```

```
MBUS:VARIABLE DATA:[125]VIFE:C9
MBUS:VARIABLE DATA:VIFE:BIT 7:EXTENSION BIT:1
MBUS:VARIABLE DATA:VIFE:BIT 6-0:UNIT+MULTIPLIER:49:SECONDARY VIF (8.4.4) a.
SUB VIF:Voltage 10^0V
MBUS:VARIABLE DATA:[126]VIFE:FF:MANUFACTURER SPECIFIC VIFE
MBUS:VARIABLE DATA:VIFE:BIT 7:EXTENSION BIT:1:MANUFACTURER SPECIFIC VIFE
MBUS:VARIABLE DATA:VIFE:BIT 6-0:UNIT+MULTIPLIER:7F:MANUFACTURER SPECIFIC VIFE
MBUS:VARIABLE DATA:[127]VIFE:01:MANUFACTURER SPECIFIC:SEC
MBUS:VARIABLE DATA:VIFE:BIT 7:EXTENSION BIT:0:MANUFACTURER SPECIFIC:SEC
MBUS:VARIABLE DATA:VIFE:BIT 6-0:UNIT+MULTIPLIER:01:MANUFACTURER SPECIFIC:SEC
VIF SEC:L1 phase value
MBUS:FIX DATA:[128]-[131]DATABLOCK:LENGTH:04,4
MBUS:FIX DATA:DATA BLOCK:DATA:[8E][8B][64][43]
DIFFUNCTIONTEXT:INSTANTANEUS VALUE
DIFTEXT:32BIT FLOAT(4 bytes)
VIFTEXT:Voltage 10^0V-L1 phase value
------------------------------------------------------------------------------------------------------------------------
MBUS:VARIABLE DATA:DATARECORD:1:DATABLOCK:13
------------------------------------------------------------------------------------------------------------------------
MBUS:VARIABLE DATA:[132]DIF:05
MBUS:VARIABLE DATA:DIF:BIT 7:EXTENSION BIT:0
MBUS:VARIABLE DATA:DIF:BIT 6:LSB STORAGE NUMBER:0
MBUS:VARIABLE DATA:DIF:BIT 5-4:FUNCTION FIELD:0:00:Instantaneous value
MBUS:VARIABLE DATA:DIF:BIT 3-0:DATA FIELD:5:5:0101:32 Bit Real
MBUS:VARIABLE DATA:[133]VIF:FD
MBUS:VARIABLE DATA:VIF:BIT 7:EXTENSION BIT:1
MBUS:VARIABLE DATA:VIF:BIT 6-0:UNIT+MULTIPLIER:7D:LINEAR VIF EXTENSION
MBUS:VARIABLE DATA:[134]VIFE:C9
MBUS:VARIABLE DATA:VIFE:BIT 7:EXTENSION BIT:1
MBUS:VARIABLE DATA:VIFE:BIT 6-0:UNIT+MULTIPLIER:49:SECONDARY VIF (8.4.4) a.
SUB VIF:Voltage 10^0V
MBUS:VARIABLE DATA:[135]VIFE:FF:MANUFACTURER SPECIFIC VIFE
MBUS:VARIABLE DATA:VIFE:BIT 7:EXTENSION BIT:1:MANUFACTURER SPECIFIC VIFE
MBUS:VARIABLE DATA:VIFE:BIT 6-0:UNIT+MULTIPLIER:7F:MANUFACTURER SPECIFIC VIFE
MBUS:VARIABLE DATA:[136]VIFE:02:MANUFACTURER SPECIFIC:SEC
MBUS:VARIABLE DATA:VIFE:BIT 7:EXTENSION BIT:0:MANUFACTURER SPECIFIC:SEC
MBUS:VARIABLE DATA:VIFE:BIT 6-0:UNIT+MULTIPLIER:02:MANUFACTURER SPECIFIC:SEC
VIF SEC:L2 phase value
MBUS:FIX DATA:[137]-[140]DATABLOCK:LENGTH:04,4
MBUS:FIX DATA:DATA BLOCK:DATA:[00][00][C0][FF]
DIFFUNCTIONTEXT:INSTANTANEUS VALUE
DIFTEXT:32BIT FLOAT(4 bytes)
VIFTEXT:Voltage 10^0V-L2 phase value
------------------------------------------------------------------------------------------------------------------------
MBUS:VARIABLE DATA:DATARECORD:1:DATABLOCK:14
------------------------------------------------------------------------------------------------------------------------
MBUS:VARIABLE DATA:[141]DIF:05
MBUS:VARIABLE DATA:DIF:BIT 7:EXTENSION BIT:0
MBUS:VARIABLE DATA:DIF:BIT 6:LSB STORAGE NUMBER:0
MBUS:VARIABLE DATA:DIF:BIT 5-4:FUNCTION FIELD:0:00:Instantaneous value
MBUS:VARIABLE DATA:DIF:BIT 3-0:DATA FIELD:5:5:0101:32 Bit Real
MBUS:VARIABLE DATA:[142]VIF:FD
MBUS:VARIABLE DATA:VIF:BIT 7:EXTENSION BIT:1
MBUS:VARIABLE DATA:VIF:BIT 6-0:UNIT+MULTIPLIER:7D:LINEAR VIF EXTENSION
MBUS:VARIABLE DATA:[143]VIFE:C9
MBUS:VARIABLE DATA:VIFE:BIT 7:EXTENSION BIT:1
MBUS:VARIABLE DATA:VIFE:BIT 6-0:UNIT+MULTIPLIER:49:SECONDARY VIF (8.4.4) a.
SUB VIF:Voltage 10^0V
MBUS:VARIABLE DATA:[144]VIFE:FF:MANUFACTURER SPECIFIC VIFE
MBUS:VARIABLE DATA:VIFE:BIT 7:EXTENSION BIT:1:MANUFACTURER SPECIFIC VIFE
MBUS:VARIABLE DATA:VIFE:BIT 6-0:UNIT+MULTIPLIER:7F:MANUFACTURER SPECIFIC VIFE
MBUS:VARIABLE DATA:[145]VIFE:03:MANUFACTURER SPECIFIC:SEC
MBUS:VARIABLE DATA:VIFE:BIT 7:EXTENSION BIT:0:MANUFACTURER SPECIFIC:SEC
MBUS:VARIABLE DATA:VIFE:BIT 6-0:UNIT+MULTIPLIER:03:MANUFACTURER SPECIFIC:SEC
VIF SEC:L3 phase value
MBUS:FIX DATA:[146]-[149]DATABLOCK:LENGTH:04,4
MBUS:FIX DATA:DATA BLOCK:DATA:[00][00][C0][FF]
DIFFUNCTIONTEXT:INSTANTANEUS VALUE
DIFTEXT:32BIT FLOAT(4 bytes)
VIFTEXT:Voltage 10^0V-L3 phase value
------------------------------------------------------------------------------------------------------------------------
MBUS:VARIABLE DATA:DATARECORD:1:DATABLOCK:15
------------------------------------------------------------------------------------------------------------------------
MBUS:VARIABLE DATA:[150]DIF:05
MBUS:VARIABLE DATA:DIF:BIT 7:EXTENSION BIT:0
MBUS:VARIABLE DATA:DIF:BIT 6:LSB STORAGE NUMBER:0
MBUS:VARIABLE DATA:DIF:BIT 5-4:FUNCTION FIELD:0:00:Instantaneous value
MBUS:VARIABLE DATA:DIF:BIT 3-0:DATA FIELD:5:5:0101:32 Bit Real
MBUS:VARIABLE DATA:[151]VIF:FD
MBUS:VARIABLE DATA:VIF:BIT 7:EXTENSION BIT:1
MBUS:VARIABLE DATA:VIF:BIT 6-0:UNIT+MULTIPLIER:7D:LINEAR VIF EXTENSION
MBUS:VARIABLE DATA:[152]VIFE:C9
MBUS:VARIABLE DATA:VIFE:BIT 7:EXTENSION BIT:1
MBUS:VARIABLE DATA:VIFE:BIT 6-0:UNIT+MULTIPLIER:49:SECONDARY VIF (8.4.4) a.
SUB VIF:Voltage 10^0V
MBUS:VARIABLE DATA:[153]VIFE:FF:MANUFACTURER SPECIFIC VIFE
MBUS:VARIABLE DATA:VIFE:BIT 7:EXTENSION BIT:1:MANUFACTURER SPECIFIC VIFE
MBUS:VARIABLE DATA:VIFE:BIT 6-0:UNIT+MULTIPLIER:7F:MANUFACTURER SPECIFIC VIFE
MBUS:VARIABLE DATA:[154]VIFE:04:MANUFACTURER SPECIFIC:SEC
MBUS:VARIABLE DATA:VIFE:BIT 7:EXTENSION BIT:0:MANUFACTURER SPECIFIC:SEC
MBUS:VARIABLE DATA:VIFE:BIT 6-0:UNIT+MULTIPLIER:04:MANUFACTURER SPECIFIC:SEC
VIF SEC:L-N average
```

```
MBUS:FIX DATA:[155]-[158]DATABLOCK:LENGTH:04,4
MBUS:FIX DATA:DATA BLOCK:DATA:[8E][8B][64][43]
DIFFUNCTIONTEXT:INSTANTANEUS VALUE
DIFTEXT:32BIT FLOAT(4 bytes)
VIFTEXT:Voltage 10^0V-L-N average
------------------------------------------------------------------------------------------------------------------
MBUS:VARIABLE DATA:DATARECORD:1:DATABLOCK:16
------------------------------------------------------------------------------------------------------------------
MBUS:VARIABLE DATA:[159]DIF:05
MBUS:VARIABLE DATA:DIF:BIT 7:EXTENSION BIT:0
MBUS:VARIABLE DATA:DIF:BIT 6:LSB STORAGE NUMBER:0
MBUS:VARIABLE DATA:DIF:BIT 5-4:FUNCTION FIELD:0:00:Instantaneous value
MBUS:VARIABLE DATA:DIF:BIT 3-0:DATA FIELD:5:5:0101:32 Bit Real
MBUS:VARIABLE DATA:[160]VIF:AE
MBUS:VARIABLE DATA:VIF:BIT 7:EXTENSION BIT:1
MBUS:VARIABLE DATA:VIF:BIT 6-0:UNIT+MULTIPLIER:2E:PRIMARY VIF 8.4.3
VIF POWER:10^3 W
MBUS:VARIABLE DATA:[161]VIFE:FF:MANUFACTURER SPECIFIC VIFE
MBUS:VARIABLE DATA:VIFE:BIT 7:EXTENSION BIT:1:MANUFACTURER SPECIFIC VIFE
MBUS:VARIABLE DATA:VIFE:BIT 6-0:UNIT+MULTIPLIER:7F:MANUFACTURER SPECIFIC VIFE
MBUS:VARIABLE DATA:[162]VIFE:01:MANUFACTURER SPECIFIC:SEC
MBUS:VARIABLE DATA:VIFE:BIT 7:EXTENSION BIT:0:MANUFACTURER SPECIFIC:SEC
MBUS:VARIABLE DATA:VIFE:BIT 6-0:UNIT+MULTIPLIER:01:MANUFACTURER SPECIFIC:SEC
VIF SEC:L1 phase value
MBUS:FIX DATA:[163]-[166]DATABLOCK:LENGTH:04,4
MBUS:FIX DATA:DATA BLOCK:DATA:[00][00][00][00]
DIFFUNCTIONTEXT:INSTANTANEUS VALUE
DIFTEXT:32BIT FLOAT(4 bytes)
VIFTEXT:Power:10^3 W-L1 phase value
------------------------------------------------------------------------------------------------------------------
MBUS:VARIABLE DATA:DATARECORD:1:DATABLOCK:17
------------------------------------------------------------------------------------------------------------------
MBUS:VARIABLE DATA:[167]DIF:05
MBUS:VARIABLE DATA:DIF:BIT 7:EXTENSION BIT:0
MBUS:VARIABLE DATA:DIF:BIT 6:LSB STORAGE NUMBER:0
MBUS:VARIABLE DATA:DIF:BIT 5-4:FUNCTION FIELD:0:00:Instantaneous value
MBUS:VARIABLE DATA:DIF:BIT 3-0:DATA FIELD:5:5:0101:32 Bit Real
MBUS:VARIABLE DATA:[168]VIF:AE
MBUS:VARIABLE DATA:VIF:BIT 7:EXTENSION BIT:1
MBUS:VARIABLE DATA:VIF:BIT 6-0:UNIT+MULTIPLIER:2E:PRIMARY VIF 8.4.3
VIF POWER:10^3 W
MBUS:VARIABLE DATA:[169]VIFE:FF:MANUFACTURER SPECIFIC VIFE
MBUS:VARIABLE DATA:VIFE:BIT 7:EXTENSION BIT:1:MANUFACTURER SPECIFIC VIFE
MBUS:VARIABLE DATA:VIFE:BIT 6-0:UNIT+MULTIPLIER:7F:MANUFACTURER SPECIFIC VIFE
MBUS:VARIABLE DATA:[170]VIFE:02:MANUFACTURER SPECIFIC:SEC
MBUS:VARIABLE DATA:VIFE:BIT 7:EXTENSION BIT:0:MANUFACTURER SPECIFIC:SEC
MBUS:VARIABLE DATA:VIFE:BIT 6-0:UNIT+MULTIPLIER:02:MANUFACTURER SPECIFIC:SEC
VIF SEC:L2 phase value
MBUS:FIX DATA:[171]-[174]DATABLOCK:LENGTH:04,4
MBUS:FIX DATA:DATA BLOCK:DATA:[00][00][C0][FF]
DIFFUNCTIONTEXT:INSTANTANEUS VALUE
DIFTEXT:32BIT FLOAT(4 bytes)
VIFTEXT:Power:10^3 W-L2 phase value
------------------------------------------------------------------------------------------------------------------
MBUS:VARIABLE DATA:DATARECORD:1:DATABLOCK:18
------------------------------------------------------------------------------------------------------------------
MBUS:VARIABLE DATA:[175]DIF:05
MBUS:VARIABLE DATA:DIF:BIT 7:EXTENSION BIT:0
MBUS:VARIABLE DATA:DIF:BIT 6:LSB STORAGE NUMBER:0
MBUS:VARIABLE DATA:DIF:BIT 5-4:FUNCTION FIELD:0:00:Instantaneous value
MBUS:VARIABLE DATA:DIF:BIT 3-0:DATA FIELD:5:5:0101:32 Bit Real
MBUS:VARIABLE DATA:[176]VIF:AE
MBUS:VARIABLE DATA:VIF:BIT 7:EXTENSION BIT:1
MBUS:VARIABLE DATA:VIF:BIT 6-0:UNIT+MULTIPLIER:2E:PRIMARY VIF 8.4.3
VIF POWER:10^3 W
MBUS:VARIABLE DATA:[177]VIFE:FF:MANUFACTURER SPECIFIC VIFE
MBUS:VARIABLE DATA:VIFE:BIT 7:EXTENSION BIT:1:MANUFACTURER SPECIFIC VIFE
MBUS:VARIABLE DATA:VIFE:BIT 6-0:UNIT+MULTIPLIER:7F:MANUFACTURER SPECIFIC VIFE
MBUS:VARIABLE DATA:[178]VIFE:03:MANUFACTURER SPECIFIC:SEC
MBUS:VARIABLE DATA:VIFE:BIT 7:EXTENSION BIT:0:MANUFACTURER SPECIFIC:SEC
MBUS:VARIABLE DATA:VIFE:BIT 6-0:UNIT+MULTIPLIER:03:MANUFACTURER SPECIFIC:SEC
VIF SEC:L3 phase value
MBUS:FIX DATA:[179]-[182]DATABLOCK:LENGTH:04,4
MBUS:FIX DATA:DATA BLOCK:DATA:[00][00][C0][FF]
DIFFUNCTIONTEXT:INSTANTANEUS VALUE
DIFTEXT:32BIT FLOAT(4 bytes)
VIFTEXT:Power:10^3 W-L3 phase value
------------------------------------------------------------------------------------------------------------------
MBUS:VARIABLE DATA:DATARECORD:1:DATABLOCK:19
------------------------------------------------------------------------------------------------------------------
MBUS:VARIABLE DATA:[183]DIF:05
MBUS:VARIABLE DATA:DIF:BIT 7:EXTENSION BIT:0
MBUS:VARIABLE DATA:DIF:BIT 6:LSB STORAGE NUMBER:0
MBUS:VARIABLE DATA:DIF:BIT 5-4:FUNCTION FIELD:0:00:Instantaneous value
MBUS:VARIABLE DATA:DIF:BIT 3-0:DATA FIELD:5:5:0101:32 Bit Real
MBUS:VARIABLE DATA:[184]VIF:2E
MBUS:VARIABLE DATA:VIF:BIT 7:EXTENSION BIT:0
MBUS:VARIABLE DATA:VIF:BIT 6-0:UNIT+MULTIPLIER:2E:PRIMARY VIF 8.4.3
VIF POWER:10^3 W
MBUS:FIX DATA:[185]-[188]DATABLOCK:LENGTH:04,4
MBUS:FIX DATA:DATA BLOCK:DATA:[00][00][00][00]
DIFFUNCTIONTEXT:INSTANTANEUS VALUE
```

```
DIFTEXT:32BIT FLOAT(4 bytes)
VIFTEXT:Power:10^3 W
------------------------------------------------------------------------------------------------------------------------
MBUS:VARIABLE DATA:DATARECORD:1:DATABLOCK:20
------------------------------------------------------------------------------------------------------------------------
MBUS:VARIABLE DATA:[189]DIF:85
MBUS:VARIABLE DATA:DIF:BIT 7:EXTENSION BIT:1
MBUS:VARIABLE DATA:DIF:BIT 6:LSB STORAGE NUMBER:0
MBUS:VARIABLE DATA:DIF:BIT 5-4:FUNCTION FIELD:0:00:Instantaneous value
MBUS:VARIABLE DATA:DIF:BIT 3-0:DATA FIELD:5:5:0101:32 Bit Real
MBUS:VARIABLE DATA:[190]DIFE:40
MBUS:VARIABLE DATA:DIFE:BIT 7:EXTENSION BIT:0
MBUS:VARIABLE DATA:DIFE:BIT 6:DEVICE UNIT:1
MBUS:VARIABLE DATA:DIFE:BIT 5-4:TARIFF:0
MBUS:VARIABLE DATA:DIFE:BIT 3-0:STORAGE NUMBER:0
MBUS:VARIABLE DATA:[191]VIF:2E
MBUS:VARIABLE DATA:VIF:BIT 7:EXTENSION BIT:0
MBUS:VARIABLE DATA:VIF:BIT 6-0:UNIT+MULTIPLIER:2E:PRIMARY VIF 8.4.3
VIF POWER:10^3 W
MBUS:FIX DATA:[192]-[195]DATABLOCK:LENGTH:04,4
MBUS:FIX DATA:DATA BLOCK:DATA:[00][00][00][00]
DIFFUNCTIONTEXT:INSTANTANEUS VALUE
DIFTEXT:32BIT FLOAT(4 bytes)
VIFTEXT:Power:10^3 W
------------------------------------------------------------------------------------------------------------------------
MBUS:VARIABLE DATA:DATARECORD:1:DATABLOCK:21
------------------------------------------------------------------------------------------------------------------------
MBUS:VARIABLE DATA:[196]DIF:85
MBUS:VARIABLE DATA:DIF:BIT 7:EXTENSION BIT:1
MBUS:VARIABLE DATA:DIF:BIT 6:LSB STORAGE NUMBER:0
MBUS:VARIABLE DATA:DIF:BIT 5-4:FUNCTION FIELD:0:00:Instantaneous value
MBUS:VARIABLE DATA:DIF:BIT 3-0:DATA FIELD:5:5:0101:32 Bit Real
MBUS:VARIABLE DATA:[197]DIFE:80
MBUS:VARIABLE DATA:DIFE:BIT 7:EXTENSION BIT:1
MBUS:VARIABLE DATA:DIFE:BIT 6:DEVICE UNIT:0
MBUS:VARIABLE DATA:DIFE:BIT 5-4:TARIFF:0
MBUS:VARIABLE DATA:DIFE:BIT 3-0:STORAGE NUMBER:0
MBUS:VARIABLE DATA:[198]DIFE:40
MBUS:VARIABLE DATA:DIFE:BIT 7:EXTENSION BIT:0
MBUS:VARIABLE DATA:DIFE:BIT 6:DEVICE UNIT:1
MBUS:VARIABLE DATA:DIFE:BIT 5-4:TARIFF:0
MBUS:VARIABLE DATA:DIFE:BIT 3-0:STORAGE NUMBER:0
MBUS:VARIABLE DATA:[199]VIF:2E
MBUS:VARIABLE DATA:VIF:BIT 7:EXTENSION BIT:0
MBUS:VARIABLE DATA:VIF:BIT 6-0:UNIT+MULTIPLIER:2E:PRIMARY VIF 8.4.3
VIF POWER:10^3 W
MBUS:FIX DATA:[200]-[203]DATABLOCK:LENGTH:04,4
MBUS:FIX DATA:DATA BLOCK:DATA:[00][00][00][00]
DIFFUNCTIONTEXT:INSTANTANEUS VALUE
DIFTEXT:32BIT FLOAT(4 bytes)
VIFTEXT:Power:10^3 W
------------------------------------------------------------------------------------------------------------------------
MBUS:VARIABLE DATA:DATARECORD:1:DATABLOCK:22
------------------------------------------------------------------------------------------------------------------------
MBUS:VARIABLE DATA:[204]DIF:05
MBUS:VARIABLE DATA:DIF:BIT 7:EXTENSION BIT:0
MBUS:VARIABLE DATA:DIF:BIT 6:LSB STORAGE NUMBER:0
MBUS:VARIABLE DATA:DIF:BIT 5-4:FUNCTION FIELD:0:00:Instantaneous value
MBUS:VARIABLE DATA:DIF:BIT 3-0:DATA FIELD:5:5:0101:32 Bit Real
MBUS:VARIABLE DATA:[205]VIF:FF
MBUS:VARIABLE DATA:VIF:BIT 7:EXTENSION BIT:1
MBUS:VARIABLE DATA:VIF:BIT 6-0:UNIT+MULTIPLIER:7F:MANUFACTURER SPECIFIC VIF
MBUS:VARIABLE DATA:[206]VIFE:0A:MANUFACTURER SPECIFIC:SEC
MBUS:VARIABLE DATA:VIFE:BIT 7:EXTENSION BIT:0:MANUFACTURER SPECIFIC:SEC
MBUS:VARIABLE DATA:VIFE:BIT 6-0:UNIT+MULTIPLIER:0A:MANUFACTURER SPECIFIC:SEC
VIF SEC:Power Factor
MBUS:FIX DATA:[207]-[210]DATABLOCK:LENGTH:04,4
MBUS:FIX DATA:DATA BLOCK:DATA:[00][00][C0][FF]
DIFFUNCTIONTEXT:INSTANTANEUS VALUE
DIFTEXT:32BIT FLOAT(4 bytes)
VIFTEXT:Power Factor
------------------------------------------------------------------------------------------------------------------------
MBUS:VARIABLE DATA:DATARECORD:1:DATABLOCK:23
------------------------------------------------------------------------------------------------------------------------
MBUS:VARIABLE DATA:[211]DIF:05
MBUS:VARIABLE DATA:DIF:BIT 7:EXTENSION BIT:0
MBUS:VARIABLE DATA:DIF:BIT 6:LSB STORAGE NUMBER:0
MBUS:VARIABLE DATA:DIF:BIT 5-4:FUNCTION FIELD:0:00:Instantaneous value
MBUS:VARIABLE DATA:DIF:BIT 3-0:DATA FIELD:5:5:0101:32 Bit Real
MBUS:VARIABLE DATA:[212]VIF:FF
MBUS:VARIABLE DATA:VIF:BIT 7:EXTENSION BIT:1
MBUS:VARIABLE DATA:VIF:BIT 6-0:UNIT+MULTIPLIER:7F:MANUFACTURER SPECIFIC VIF
MBUS:VARIABLE DATA:[213]VIFE:0B:MANUFACTURER SPECIFIC:SEC
MBUS:VARIABLE DATA:VIFE:BIT 7:EXTENSION BIT:0:MANUFACTURER SPECIFIC:SEC
MBUS:VARIABLE DATA:VIFE:BIT 6-0:UNIT+MULTIPLIER:0B:MANUFACTURER SPECIFIC:SEC
VIF SEC:Frequency
MBUS:FIX DATA:[214]-[217]DATABLOCK:LENGTH:04,4
MBUS:FIX DATA:DATA BLOCK:DATA:[29][F1][47][42]
DIFFUNCTIONTEXT:INSTANTANEUS VALUE
DIFTEXT:32BIT FLOAT(4 bytes)
VIFTEXT:Frequency
------------------------------------------------------------------------------------------------------------------------
```

```
MBUS:VARIABLE DATA:DATARECORD:1:DATABLOCK:24
-----------------------------------------------------------------------------------------------------------------
MBUS:VARIABLE DATA:[218]DIF:07
MBUS:VARIABLE DATA:DIF:BIT 7:EXTENSION BIT:0
MBUS:VARIABLE DATA:DIF:BIT 6:LSB STORAGE NUMBER:0
MBUS:VARIABLE DATA:DIF:BIT 5-4:FUNCTION FIELD:0:00:Instantaneous value
MBUS:VARIABLE DATA:DIF:BIT 3-0:DATA FIELD:7:7:0111:64 Bit Integer
MBUS:VARIABLE DATA:[219]VIF:03
MBUS:VARIABLE DATA:VIF:BIT 7:EXTENSION BIT:0
MBUS:VARIABLE DATA:VIF:BIT 6-0:UNIT+MULTIPLIER:03:PRIMARY VIF 8.4.3
VIF ENERGY:10^0 Wh
MBUS:FIX DATA:[220]-[227]DATABLOCK:LENGTH:08,8
MBUS:FIX DATA:DATA BLOCK:DATA:[19][00][00][00][00][00][00][00]
DIFFUNCTIONTEXT:INSTANTANEUS VALUE
DIFTEXT:64BIT INT(8 bytes)
VIFTEXT:Energy:10^0 Wh
-----------------------------------------------------------------------------------------------------------------
MBUS:VARIABLE DATA:DATARECORD:1:DATABLOCK:25
-----------------------------------------------------------------------------------------------------------------
MBUS:VARIABLE DATA:[228]DIF:1F
MBUS:VARIABLE DATA:DIF:BIT 7:EXTENSION BIT:0
MBUS:VARIABLE DATA:DIF:BIT 6:LSB STORAGE NUMBER:0
MBUS:VARIABLE DATA:DIF:BIT 5-4:FUNCTION FIELD:1:01:Maximum value
MBUS:VARIABLE DATA:DIF:BIT 3-0:DATA FIELD:F:F:1111:Special Functions
MBUS:VARIABLE DATA:DIF:MORE RECORDS

=================================================================================================================
MBUS:END VARIABLE DATA
=================================================================================================================
END OF FRAME
=================================================================================================================
```

You will notice, that the MBUS answer starts with a fixed header. This information is interpreted by our gateway and stored in a fixed MODBUS mapping structure starting at 4x10001. See the MODBUS register definition for more details.

4x10001	HEADER	UINT32	Identification number of meter	ID	MSW:0636,4163:LSW	104218979,0x06364163	Meter 4 [P:4]
4x10003	HEADER	UINT32->ASCI	Manufacturer of meter	MANUFACTURER	MSW:0043,4553:LSW	SEC	Meter 4 [P:4]
4x10005	HEADER	UINT16	Version of meter	VERSION	WORD:0018	24,0x0018	Meter 4 [P:4]
4x10006	HEADER	UINT16	Medium of meter	MEDIUM	WORD:0002	2,0x0002 -> Electricity	Meter 4 [P:4]
4x10007	HEADER	UINT16	Access of meter	ACCESS	WORD:0056	86,0x0056	Meter 4 [P:4]
4x10008	HEADER	UINT16	Status of meter	STATUS	WORD:0000	0,0x0000	Meter 4 [P:4]
4x10009	RESI	UINT16	Future value of meter	FUTURE	WORD:0000	0,0x0000	Meter 4 [P:4]
4x10010	RESI	UINT16	Communcation state with meter	COMM STATE	WORD:0003	3,0x0003 -> Values are valid!	Meter 4 [P:4]

After the fixed data header, there come one or more data records. The gateway will read this records sequentially and map one datapoint to another to MODBUS registers. So the configured mapping datapoints must be sequentially defined according to the data in the MBUS frame.

The first entry is:

```
-----------------------------------------------------------------------------------------------------------------
MBUS:VARIABLE DATA:DATARECORD:1:DATABLOCK:0
-----------------------------------------------------------------------------------------------------------------
MBUS:VARIABLE DATA:[0]DIF:0D
MBUS:VARIABLE DATA:DIF:BIT 7:EXTENSION BIT:0
MBUS:VARIABLE DATA:DIF:BIT 6:LSB STORAGE NUMBER:0
MBUS:VARIABLE DATA:DIF:BIT 5-4:FUNCTION FIELD:0:00:Instantaneous value
MBUS:VARIABLE DATA:DIF:BIT 3-0:DATA FIELD:D:D:1101:variable length
MBUS:VARIABLE DATA:[1]VIF:FD
MBUS:VARIABLE DATA:VIF:BIT 7:EXTENSION BIT:1
MBUS:VARIABLE DATA:VIF:BIT 6-0:UNIT+MULTIPLIER:7D:LINEAR VIF EXTENSION
MBUS:VARIABLE DATA:[2]VIFE:0A
MBUS:VARIABLE DATA:VIFE:BIT 7:EXTENSION BIT:0
MBUS:VARIABLE DATA:VIFE:BIT 6-0:UNIT+MULTIPLIER:0A:SECONDARY VIF (8.4.4) a.
SUB VIF:MANUFACTURER (as in fixed header)
MBUS:VARIABLE DATA:ASCII:18
MBUS:VARIABLE DATA:[3]-[21]DATABLOCK:LENGTH:12,18
MBUS:VARIABLE DATA:DATA BLOCK:DATA:[63][69][72][74][63][65][6C][45][20][72][65][64][69][65][6E][68][63][53]
MBUS:VARIABLE DATA:DATA BLOCK:DATA:ASCII:Schneider Electric
DIFFUNCTIONTEXT:INSTANTANEUS VALUE
DIFTEXT:LVAR:ASCII(18 bytes)
VIFTEXT:Manufacturer
```

The first entry in our configuration matches this MBUS data. It is an ASCII string which defines the manufacturer of the meter. The Name starts at byte index 4 in the first received data frame from the meter and it needs 18 bytes. So now the gateway knows exactly, that it has to copy the 18 bytes starting form index 4 to the first 9 MODBUS 16-Bit registers starting at 4x00001. But also the byte order is mirrored for the ASCII string and there is no trailing 0x00 character at the end of the string, so our software maps this string to 10 16-bit registers and adds the trailing 0x00 character.

Index	MBUS dataty...	MB datatype	Content	MBUS data	MBUS size	MBUS exponent	MB exponent
0	LVAR:ASCII	ASCII	Manufacturer	1-4	18	0	0
1	LVAR:ASCII	ASCII	Model/version	1-26	8	0	0
2	LVAR:ASCII	ASCII	Firmware version	1-38	7	0	0
3	INT24	UINT32	Error flags (binary)	1-48	3	0	0
4	FLOAT32	FLOAT32	Current 10^0A-L1 phase value	1-56	4	0	0
5	FLOAT32	FLOAT32	Current 10^0A-L2 phase value	1-65	4	0	0
6	FLOAT32	FLOAT32	Current 10^0A-L3 phase value	1-74	4	0	0
7	FLOAT32	FLOAT32	Current 10^0A-Average current	1-83	4	0	0
8	FLOAT32	FLOAT32	Voltage 10^0V-L1-L2	1-92	4	0	0
9	FLOAT32	FLOAT32	Voltage 10^0V-L2-L3	1-101	4	0	0
10	FLOAT32	FLOAT32	Voltage 10^0V-L3-L1	1-110	4	0	0

43.16.1 HOW the exponents affect the result

In the MBUS protocol not only the MBUS data for a data point is transmitted, also also the meaning of the data and the dimension of the value is transmitted. For example we take a KAMSTRUP flowIQ meter. Our software generates the following mapping:

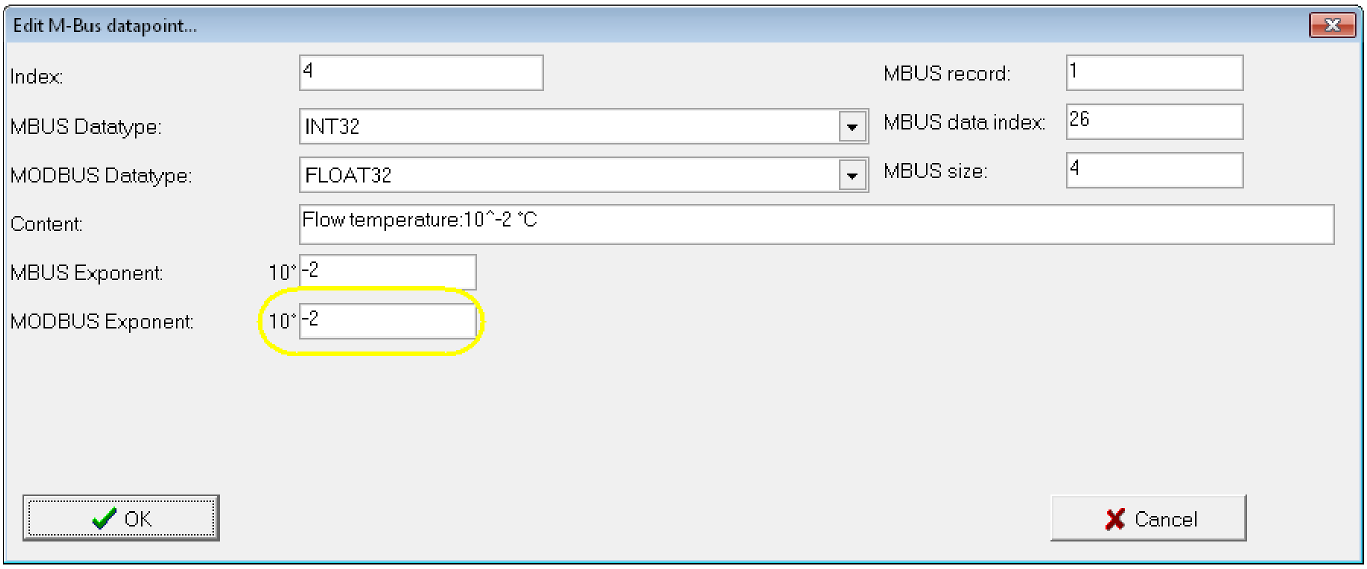
Index	MBUS dataty...	MB datatype	Content	MBUS data	MBUS size	MBUS exponent	MB exponent
0	BCD8	SINT32	Fabrication number	1-2	4	0	0
1	INT32	UINT32	Energy:10^4 Wh	1-8	4	4	0
2	INT32	FLOAT32	Volume:10^-1 m³	1-14	4	-1	0
3	INT32	UINT32	On time:hours	1-20	4	0	0
4	INT32	FLOAT32	Flow temperature:10^-2 °C	1-26	4	-2	0
5	INT32	FLOAT32	Return temperature:10^-2 °C	1-32	4	-2	0
6	INT32	FLOAT32	Temperature difference:10^-2 K	1-38	4	-2	0
7	INT32	FLOAT32	Power:10^2 W	1-44	4	2	0
8	INT32	FLOAT32	Power:10^2 W	1-50	4	2	0
9	INT32	FLOAT32	Volume flow:10^-3 m³/h	1-56	4	-3	0
10	INT32	FLOAT32	Volume flow:10^-3 m³/h	1-62	4	-3	0
11	INT32	FLOAT32	Volume:10^-1 m³[U:1,T:0,S:0]	1-69	4	-1	0
12	INT32	FLOAT32	Volume:10^0 m³[U:2,T:0,S:0]	1-77	4	0	0
13	INT32	DATE_TIME_T...	Time&Date data type F	1-83	4	0	0
14	INT32	UINT32	Energy:10^4 Wh[U:0,T:0,S:1]	1-89	4	4	0
15	INT32	FLOAT32	Volume:10^-1 m³[U:0,T:0,S:1]	1-95	4	-1	0
16	INT32	FLOAT32	Power:10^2 W[U:0,T:0,S:1]	1-101	4	2	0
17	INT32	FLOAT32	Volume flow:10^-3 m³/h[U:0,T:0,S:1]	1-107	4	-3	0
18	INT32	FLOAT32	Volume:10^-1 m³[U:1,T:0,S:1]	1-114	4	-1	0
19	INT32	FLOAT32	Volume:10^0 m³[U:2,T:0,S:1]	1-122	4	0	0
20	INT16	DATE_TYP_G	Date data type G[U:0,T:0,S:1]	1-128	2	0	0
21	SPCL FUNCT	BUFFER	Manufacturer specific data	1-131	44	0	0

You notice in the column MBUS exponent different exponents for different values and in the column Content those exponents are added with 10^xx.

When you download and test this configuration, we get the following online data:

MB Register	MBUS datatype	MB datatype	Content	MBUS index	MB value HEX	Current MB value	Me
4x00001	BCD8[4]	SINT32	Fabrication number	0	MSW:0026,D1D2:LSW	2544082,0x0026D1D2	Me
4x00003	INT32[4]	UINT32	Energy:10^4 Wh	1	MSW:000A,1ED0:LSW	663248,0x000A1ED0	Me
4x00005	INT32[4]	FLOAT32	Volume:10^-1 m³	2	MSW:45AC,6E66:LSW	5517.7998,5.51779980468750E+3	Me
4x00007	INT32[4]	UINT32	On time:hours	3	MSW:0000,B64B:LSW	46667,0x0000B64B	Me
4x00009	INT32[4]	FLOAT32	Flow temperature:10^-2 °C	4	MSW:41F6,147B:LSW	30.7600,3.07600002288818E+1	Me
4x00011	INT32[4]	FLOAT32	Return temperature:10^-2 °C	5	MSW:4081,EB85:LSW	4.0600,4.05999994277954E+0	Me
4x00013	INT32[4]	FLOAT32	Temperature difference:10^-2 K	6	MSW:41D5,999A:LSW	26.7000,2.67000007629395E+1	Me
4x00015	INT32[4]	FLOAT32	Power:10^2 W	7	MSW:0000,0000:LSW	0.0000,0.00000000000000E+0	Me
4x00017	INT32[4]	FLOAT32	Power:10^2 W	8	MSW:0000,0000:LSW	0.0000,0.00000000000000E+0	Me
4x00019	INT32[4]	FLOAT32	Volume flow:10^-3 m³/h	9	MSW:0000,0000:LSW	0.0000,0.00000000000000E+0	Me
4x00021	INT32[4]	FLOAT32	Volume flow:10^-3 m³/h	10	MSW:0000,0000:LSW	0.0000,0.00000000000000E+0	Me
4x00023	INT32[4]	FLOAT32	Volume:10^-1 m³[U:1,T:0,S:0]	11	MSW:44A1,0CCD:LSW	1288.4000,1.28840002441406E+3	Me
4x00025	INT32[4]	FLOAT32	Volume:10^0 m³[U:2,T:0,S:0]	12	MSW:46CB,1000:LSW	25992.0000,2.59920000000000E+4	Me
4x00027	INT32[4]	DATE_TIME_T	Time&Date data type F	13	MSW:1234,2602:LSW	06:02 D.M.Y:20.02.09 ST:0 IV:0,0x12342602	Me
4x00029	INT32[4]	UINT32	Energy:10^4 Wh[U:0,T:0,S:1]	14	MSW:000A,1ED0:LSW	663248,0x000A1ED0	Me
4x00031	INT32[4]	FLOAT32	Volume:10^-1 m³[U:0,T:0,S:1]	15	MSW:45AC,6E66:LSW	5517.7998,5.51779980468750E+3	Me
4x00033	INT32[4]	FLOAT32	Power:10^2 W[U:0,T:0,S:1]	16	MSW:0000,0000:LSW	0.0000,0.00000000000000E+0	Me
4x00035	INT32[4]	FLOAT32	Volume flow:10^-3 m³/h[U:0,T:0,S:1]	17	MSW:0000,0000:LSW	0.0000,0.00000000000000E+0	Me
4x00037	INT32[4]	FLOAT32	Volume:10^-1 m³[U:1,T:0,S:1]	18	MSW:474D,6680:LSW	52582.5000,5.25825000000000E+4	Me
4x00039	INT32[4]	FLOAT32	Volume:10^0 m³[U:2,T:0,S:1]	19	MSW:4900,6010:LSW	525825.0000,5.25825000000000E+5	Me
4x00041	INT16[2]	DATE_TYP_G	Date data type G[U:0,T:0,S:1]	20	WORD:1601	D.M.Y:01.06.08,0x1601	Me
4x00042	SPCL FUNCT[44]	BUFFER	Manufacturer specific data	21	LSW:010B 0070 0C01 0141 2300 4	LSB:0B 01 70 00 01 0C 41 01 00 23 1B 44 0(	Me
4x09001	RESI	UINT16	Converter state for meter	STATE	WORD:0003	3,0x0003 -> Values are valid!	Me
4x09002	HEADER	UINT32R	Identification number of meter	ID	LSW:4082,MSW:0254	39075970,0x02544082	Me
4x10001	HEADER	UINT32	Identification number of meter	ID	MSW:0254,4082:LSW	39075970,0x02544082	Me
4x10003	HEADER	UINT32->ASCI	Manufacturer of meter	MANUFACTURER	MSW:004D,414B:LSW	KAM	Me
4x10005	HEADER	UINT16	Version of meter	VERSION	WORD:0002	2,0x0002	Me
4x10006	HEADER	UINT16	Medium of meter	MEDIUM	WORD:0004	4,0x0004 -> Heat-Volume measured at retu	Me
4x10007	HEADER	UINT16	Access of meter	ACCESS	WORD:0012	18,0x0012	Me
4x10008	HEADER	UINT16	Status of meter	STATUS	WORD:0000	0,0x0000	Me
4x10009	RESI	UINT16	Future value of meter	FUTURE	WORD:0000	0,0x0000	Me
4x10010	RESI	UINT16	Communcation state with meter	COMM STATE	WORD:0003	3,0x0003 -> Values are valid!	Me

Look at registers 4x00009 to 4x00013, three temperature values. In the MBUS frame, the temperatures are transmitted as INT32 values with the exponent 10^-2. So in fact this are integer values with two commas: 2812 will mean 28,12°C. Our converter maps this values to FLOAT32 values and automatically shifts the MBUS exponents to display the value based to 10^0. But in some cases you don't want to shift. So we double click on the desired datapoint in the meter configuration and modify the MODBUS exponent in entering the number -2. This means, that we want to multiply the MBUS value by 10^-2.



The result in the MODBUS registers will be a temperature value multiplied by 100 to represent 1/100°C:

MB Register	MBUS datatype	MB datatype	Content	MBUS index	MB value HEX	Current MB value	Meter name
4x00001	BCD8[4]	SINT32	Fabrication number	0	MSW:0026,D1D2:LSW	2544082,0x0026D1D2	Meter 0254
4x00003	INT32[4]	UINT32	Energy:10^4 Wh	1	MSW:000A,1ED0:LSW	663248,0x000A1ED0	Meter 0254
4x00005	INT32[4]	FLOAT32	Volume:10^-1 m³	2	MSW:45AC,6E66:LSW	5517.7998,5.51779980468750E+3	Meter 0254
4x00007	INT32[4]	UINT32	On time:hours	3	MSW:0000,B64B:LSW	46667,0x0000B64B	Meter 0254
4x00009	INT32[4]	FLOAT32	Flow temperature:10^-2 °C->*10^-2	4	MSW:4540,3000:LSW	3075.0000,3.07500000000000E+3	Meter 0254
4x00011	INT32[4]	FLOAT32	Return temperature:10^-2 °C->*10^-2	5	MSW:43CA,8000:LSW	405.0000,4.05000000000000E+2	Meter 0254
4x00013	INT32[4]	FLOAT32	Temperature difference:10^-2 K->*10^-1	6	MSW:4385,8000:LSW	267.0000,2.67000000000000E+2	Meter 0254
4x00015	INT32[4]	FLOAT32	Power:10^2 W	7	MSW:0000,0000:LSW	0.0000,0.00000000000000E+0	Meter 0254

If we change now the MODBUS data type to SINT16 we use only one MODBUS register for every temperature, but because the original MBUS value is based to 10^-2, we have to set the MODBUS exponent to 0. This is different to a FLOAT32 or DOUBLE64 MODBUS register, where the gateway always normalize the MBUS value to 10^0.

Index	MBUS dataty...	MB datatype	Content	MBUS data	MBUS size	MBUS exponent	MB exponent
0	BCD8	SINT32	Fabrication number	1-2	4	0	0
1	INT32	UINT32	Energy:10^4 Wh	1-8	4	4	0
2	INT32	FLOAT32	Volume:10^-1 m³	1-14	4	-1	0
3	INT32	UINT32	On time:hours	1-20	4	0	0
4	INT32	SINT16	Flow temperature:10^-2 °C	1-26	4	-2	0
5	INT32	SINT16	Return temperature:10^-2 °C	1-32	4	-2	0
6	INT32	SINT16	Temperature difference:10^-2 K	1-38	4	-2	0
7	INT32	FLOAT32	Power:10^2 W	1-44	4	2	0
8	INT32	FLOAT32	Power:10^2 W	1-50	4	2	0
9	INT32	FLOAT32	Volume flow:10^-3 m³/h	1-56	4	-3	0
10	INT32	FLOAT32	Volume flow:10^-3 m³/h	1-62	4	-3	0
11	INT32	FLOAT32	Volume:10^-1 m³[U:1,T:0,S:0]	1-69	4	-1	0
12	INT32	FLOAT32	Volume:10^0 m³[U:2,T:0,S:0]	1-77	4	0	0
13	INT32	DATE_TIME_T...	Time&Date data type F	1-83	4	0	0
14	INT32	UINT32	Energy:10^4 Wh[U:0,T:0,S:1]	1-89	4	4	0
15	INT32	FLOAT32	Volume:10^-1 m³[U:0,T:0,S:1]	1-95	4	-1	0
16	INT32	FLOAT32	Power:10^2 W[U:0,T:0,S:1]	1-101	4	2	0
17	INT32	FLOAT32	Volume flow:10^-3 m³/h[U:0,T:0,S:1]	1-107	4	-3	0
18	INT32	FLOAT32	Volume:10^-1 m³[U:1,T:0,S:1]	1-114	4	-1	0
19	INT32	FLOAT32	Volume:10^0 m³[U:2,T:0,S:1]	1-122	4	0	0
20	INT16	DATE_TYP_G	Date data type G[U:0,T:0,S:1]	1-128	2	0	0
21	SPCL FUNCT	BUFFER	Manufacturer specific data	1-131	44	0	0

Download and test the new configuration, you will see the difference:

MB Register	MBUS datatype	MB datatype	Content	MBUS index	MB value HEX	Current MB value	M
4x00001	BCD8[4]	SINT32	Fabrication number	0	MSW:0026,D1D2:LSW	2544082,0x0026D1D2	M
4x00003	INT32[4]	UINT32	Energy:10^4 Wh	1	MSW:000A,1ED0:LSW	663248,0x000A1ED0	M
4x00005	INT32[4]	FLOAT32	Volume:10^-1 m³	2	MSW:45AC,6E66:LSW	5517.7998,5.51779980468750E+3	M
4x00007	INT32[4]	UINT32	On time:hours	3	MSW:0000,B64B:LSW	46667,0x0000B64B	M
4x00009	INT32[4]	SINT16	Flow temperature:10^-2 °C	4	WORD:0C04	3076,0x0C04	M
4x00010	INT32[4]	SINT16	Return temperature:10^-2 °C	5	WORD:0196	406,0x0196	M
4x00011	INT32[4]	SINT16	Temperature difference:10^-2 K	6	WORD:0A6E	2670,0x0A6E	M
4x00012	INT32[4]	FLOAT32	Power:10^2 W	7	MSW:0000,0000:LSW	0.0000,0.00000000000000E+0	M
4x00014	INT32[4]	FLOAT32	Power:10^2 W	8	MSW:0000,0000:LSW	0.0000,0.00000000000000E+0	M

Now we want to store the temperatures only with one comma in the 16 bit holding registers. Therefore we use the MODBUS exponent to divide the values by 10. So we enter an exponent of 1 for all tree temperatures:

Index	MBUS dataty...	MB datatype	Content	MBUS data	MBUS size	MBUS exponent	MB exponent
0	BCD8	SINT32	Fabrication number	1-2	4	0	0
1	INT32	UINT32	Energy:10^4 Wh	1-8	4	4	0
2	INT32	FLOAT32	Volume:10^-1 m³	1-14	4	-1	0
3	INT32	UINT32	On time:hours	1-20	4	0	0
4	INT32	SINT16	Flow temperature:10^-2 °C	1-26	4	-2	1
5	INT32	SINT16	Return temperature:10^-2 °C	1-32	4	-2	1
6	INT32	SINT16	Temperature difference:10^-2 K	1-38	4	-2	1
7	INT32	FLOAT32	Power:10^2 W	1-44	4	2	0
8	INT32	FLOAT32	Power:10^2 W	1-50	4	2	0
9	INT32	FLOAT32	Volume flow:10^-3 m³/h	1-56	4	-3	0
10	INT32	FLOAT32	Volume flow:10^-3 m³/h	1-62	4	-3	0
11	INT32	FLOAT32	Volume:10^-1 m³[U:1,T:0,S:0]	1-69	4	-1	0
12	INT32	FLOAT32	Volume:10^0 m³[U:2,T:0,S:0]	1-77	4	0	0
13	INT32	DATE_TIME_T...	Time&Date data type F	1-83	4	0	0
14	INT32	UINT32	Energy:10^4 Wh[U:0,T:0,S:1]	1-89	4	4	0
15	INT32	FLOAT32	Volume:10^-1 m³[U:0,T:0,S:1]	1-95	4	-1	0
16	INT32	FLOAT32	Power:10^2 W[U:0,T:0,S:1]	1-101	4	2	0
17	INT32	FLOAT32	Volume flow:10^-3 m³/h[U:0,T:0,S:1]	1-107	4	-3	0
18	INT32	FLOAT32	Volume:10^-1 m³[U:1,T:0,S:1]	1-114	4	-1	0
19	INT32	FLOAT32	Volume:10^0 m³[U:2,T:0,S:1]	1-122	4	0	0
20	INT16	DATE_TYP_G	Date data type G[U:0,T:0,S:1]	1-128	2	0	0
21	SPCL FUNCT	BUFFER	Manufacturer specific data	1-131	44	0	0

Download and test again, the result will look like this:

MB Register	MBUS datatype	MB datatype	Content	MBUS index	MB value HEX	Current MB value	M
4x00001	BCD8[4]	SINT32	Fabrication number	0	MSW:0026,D1D2:LSW	2544082,0x0026D1D2	M
4x00003	INT32[4]	UINT32	Energy:10^4 Wh	1	MSW:000A,1ED0:LSW	663248,0x000A1ED0	M
4x00005	INT32[4]	FLOAT32	Volume:10^-1 m³	2	MSW:45AC,6E66:LSW	5517.7998,5.51779980468750E+3	M
4x00007	INT32[4]	UINT32	On time:hours	3	MSW:0000,B64B:LSW	46667,0x0000B64B	M
4x00009	INT32[4]	SINT16	Flow temperature:10^-2 °C->/10^1	4	WORD:0133	307,0x0133	M
4x00010	INT32[4]	SINT16	Return temperature:10^-2 °C->/10^1	5	WORD:0028	40,0x0028	M
4x00011	INT32[4]	SINT16	Temperature difference:10^-2 K->/10^1	6	WORD:010B	267,0x010B	M
4x00012	INT32[4]	FLOAT32	Power:10^2 W	7	MSW:0000,0000:LSW	0.0000,0.00000000000000E+0	M
4x00014	INT32[4]	FLOAT32	Power:10^2 W	8	MSW:0000,0000:LSW	0.0000,0.00000000000000E+0	M

So in general note the following mapping rules:

1. Using FLOAT32, FLOAT32R, DOUBLE64, DOUBLE64R as a MODBUS register type for a MBUS value always forces a normalization of the original MBUS value to base 10^0 to represent a value according to the defined unit of the MBUS value, to which we are used (°C or Wh or m³, etc.)
2. You can now multiply or divide this normalized value by entering a MBUS exponent. A negative exponent will divide the value be the factor 10^exponent, a positive exponent will multiply the value by the factor 10^exponent before the data is written to the MODBUS register.
3. Using other MODBUS data types, the original MBUS data is taken without normalization. Then the MBUS exponent only informs you, what the basis for your value is.
4. But again you can multiply or divide the values by entering an exponent into the MODBUS exponent field manually, before the value is written to the MODBUS register. Enter a positive exponent to multiply the original value by 10^exponent, enter a negative exponent to divide the value by 10^exponent.

43.17 Additional MODBUS register & coils

Here you will find only the additional MODBUS registers and coils especially for this IO module. Please refer to the description of of the standard MODBUS mapping for more details about the available basic MODBUS registers and coils.

Please refer to the external document for detailed documentation of the current MODBUS register mapping for this IO module:

RESI-L-MBUSx-SIO-ETH-MODBUS+ASCII-ENxx.pdf

43.17.1 MODBUS register for meter data

This registers are compatible to our old versions of the product (RESI-MBUSx-MODBUS and RESI-MBUSx-ETH). For the mapped MBUS data the converter uses the MODBUS holding registers starting at 4x00001.

Register	Description
4x00001 3x00001 I:0 R/O MODBUS MAPPING	First holding register of MBUS data mapping for first configured MBUS meter
...	
4x00040 3x00040 I:39 R/O MODBUS MAPPING	Last holding register of MBUS data mapping for last configured MBUS meter for products RESI-MBUS2-SIO RESI-MBUS2-ETH
…	
4x00400 3x00400 I:399 R/O MODBUS MAPPING	Last holding register of MBUS data mapping for last configured MBUS meter for products RESI-MBUS8-SIO RESI-MBUS8-ETH
...	
4x01000 3x01000 I:999 R/O MODBUS MAPPING	Last holding register of MBUS data mapping for last configured MBUS meter for products RESI-MBUS24-SIO RESI-MBUS24-ETH
...	
4x01200 3x01200 I:1199 R/O MODBUS MAPPING	Last holding register of MBUS data mapping for last configured MBUS meter for products RESI-MBUS48-SIO RESI-MBUS48-ETH
...	
4x01200 3x01200 I:1199 R/O MODBUS MAPPING	Last holding register of MBUS data mapping for last configured MBUS meter for products RESI-MBUS64-SIO RESI-MBUS64-ETH

43.17.2 MODBUS status register for meters

This status registers are compatible to our old versions of the product (RESI-MBUSx-MODBUS and RESI-MBUSx-ETH), but the position is shifted to the MODBUS range starting at 4x09001.

Register	Description
4x09001 3x09001 I:9000 R/O STATE METER 1	Returns the current state of the communication with the meter #1 =0: Meter isn't configured =1: Meter isn't normalized =2: Meter isn't read =3: Values are valid
4x09002-3 3x09002-3 I:9001-2 R/O SERIAL NUMBER METER 1	UINT32R: Returns the current serial number of the meter #1 as a 32 bit unsigned integer value 1st.WORD: lower 16 bit of the serial number 2nd.WORD: higher 16 bit of the serial number
4x09004 3x09004 I:9003 R/O STATE METER 2	UINT16: Returns the current state of the communication with the meter #2 =0: Meter isn't configured =1: Meter isn't normalized =2: Meter isn't read =3: Values are valid
4x09005-6 3x09005-6 I:9004-5 R/O SERIAL NUMBER METER 2	UINT32R: Returns the current serial number of the meter #2 as a 32 bit unsigned integer value 1st.WORD: lower 16 bit of the serial number 2nd.WORD: higher 16 bit of the serial number
…	
4x09022 3x09022 I:9021 R/O STATE METER 8	UINT16: Returns the current state of the communication with the meter #8 =0: Meter isn't configured =1: Meter isn't normalized =2: Meter isn't read =3: Values are valid
4x09023-24 3x09024-24 I:9022-23 R/O SERIAL NUMBER METER 8	UINT32R: Returns the current serial number of the meter #8 as a 32 bit unsigned integer value 1st.WORD: lower 16 bit of the serial number 2nd.WORD: higher 16 bit of the serial number
...	

Register	Description
4x09070 3x09070 **I**:9069 R/O STATE METER 24	UINT16: Returns the current state of the communication with the meter #24 =0: Meter isn't configured =1: Meter isn't normalized =2: Meter isn't read =3: Values are valid
4x09071-72 3x09071-72 **I**:9070-71 R/O SERIAL NUMBER METER 24	UINT32R: Returns the current serial number of the meter #24 as a 32 bit unsigned integer value 1st.WORD: lower 16 bit of the serial number 2nd.WORD: higher 16 bit of the serial number
...	
4x09142 3x09142 **I**:9141 R/O STATE METER 48	UINT16: Returns the current state of the communication with the meter #48 =0: Meter isn't configured =1: Meter isn't normalized =2: Meter isn't read =3: Values are valid
4x09143-144 3x09143-144 **I**:9142-143 R/O SERIAL NUMBER METER 48	UINT32R: Returns the current serial number of the meter #48 as a 32 bit unsigned integer value 1st.WORD: lower 16 bit of the serial number 2nd.WORD: higher 16 bit of the serial number
…	
4x09190 3x09190 **I**:9189 R/O STATE METER 64	UINT16: Returns the current state of the communication with the meter #64 =0: Meter isn't configured =1: Meter isn't normalized =2: Meter isn't read =3: Values are valid
4x09191-92 3x09191-92 **I**:9190-91 R/O SERIAL NUMBER METER 64	UINT32R: Returns the current serial number of the meter #64 as a 32 bit unsigned integer value 1st.WORD: lower 16 bit of the serial number 2nd.WORD: higher 16 bit of the serial number

43.17.3 MODBUS extended status register for meters

This extended status registers are new to our new version of the product (RESI-MBUSx-SIO and RESI-MBUSx-ETH). For every meter there is a set of 10 MODBUS holding registers starting with 4x10001. Mainly this registers represent the information of the MBUS fixed data header:

Ident. Nr.	Manufr.	Version	Medium	Access No.	Status	Signature
4 Byte	2 Byte	1 Byte	1 Byte	1 Byte	1 Byte	2 Byte

This header is sent by many answer frames of the MBUS meter to the master. Due to the fact, that is is not part of the variable data block of the meter, our old converters could not map this information to registers. Our new series map this information to the following register set starting at 4x10001. For each meter there are eight MODBUS entries:

Register	Description
4x10001-2 3x10001-2 I:10000-1 R/O ID NUMBER METER 1	UINT32: Returns the current serial number of the meter #1 as a 32 bit unsigned integer value 1st.WORD: higher 16 bit of the serial number 2nd.WORD: lower 16 bit of the serial number Each meter offers a unique ID. In the MBUS protocol there are four bytes reserved for this number. In our gateway we need a UINT32 to represent this 4 bytes of the ID.
4x10003-4 3x10003-4 I:10002-3 R/O MANUFACTURER METER 1	UINT32->ASCII: Returns the current manufacturer of the meter #1 as a 32 bit unsigned integer value 1st.WORD: higher 16 bit of the manufacturer name as ASCII text 2nd.WORD: lower 16 bit of of the manufacturer name as ASCII text Each meter offers a manufacturer ID, represented in two bytes. But in this two bytes there are three ASCII digits encoded. Our gateway decode this ASCII digits and stores this digits into a UINT32 using ASCII encoding with 0x00 at the end representing a standard null terminated ASCII string of three letters. For example the manufacturer KAMSTRUM uses KAM with the bytes 0x4B 0x41 0x4D. This will be represented by 32 bit value : 0x004D414B, so the higher WORD will be 0x004D and the lower word will be 0x414B.
4x10005 3x10005 I:10004 R/O VERSION METER 1	UINT16: Returns the current version of the meter #1 In the fixed data header, there is also a version number encoded into one byte. It represents the version of the meter. Our gateway stores this byte into a UINT16 holding register for easy readout.
4x10006 3x10006 I:10005 R/O MEDIUM METER 1	UINT16: Returns the current medium of the meter #1 In the fixed data header, there is also a medium number encoded into one byte. it defines what type of medium the meter is measuring. Our gateway stores this byte into a UINT16 holding register for easy readout. The following medium types are defined by the standard for meters with fixed+variable data structure: 0x00: OTHER, 0x01: OIL, 0x02: Electricity, 0x03: Gas, 0x04: Heat-Volume measured at return temperature outlet, 0x05: Steam, 0x06: Hot Water, 0x07: Water, 0x08: H.C.A.=Heat Cost Allocator, 0x09: Compressed Air, 0x0A: Cooling load meter Volume measured at return temperature outlet, 0x0B: Cooling load meter Volume measured at flow temperature inlet, 0x0C: Heat Volume measured at flow temperature inlet, 0x0D: Heat/Cooling load meter, 0x0E: Bus/System, 0x0F: Unknown Medium, 0x16: Cold Water, 0x17: Dual Water, 0x18: Pressure, 0x19: A/D Converter For meters with fixed data structure only, the 16 bit value must be interpreted in another way. Refer to the MBUS standard for this definition.

Register	Description
4x10007 3x10007 I:10006 R/O ACCESS COUNTER METER 1	UINT16: Returns the current access counter of the meter #1 In the fixed data header, there is also an access counter encoded into one byte. It will be incremented by every access of the meter data. So each readout of the meter will increment this access counter by 1 in the range from 0 to 255. Our gateway stores this byte into a UINT16 holding register for easy readout.
4x10008 3x10008 I:10007 R/O STATUS METER 1	UINT16: Returns the current status of the meter #1 In the fixed data header, there is also a status field encoded into one byte. It shows the current meter status. Our gateway stores this byte into a UINT16 holding register for easy readout. The byte has the following meaning: Bit 1+Bit 0: =00 (0) NO ERROR Bit 1+Bit 0: =10 (1) APPLICATION NOT READY Bit 1+Bit 0: =01 (2) APPLICATION ERROR Bit 1+Bit 0: =11 (3) RESERVED Bit 2: =1: POWER LOW, =0: POWER OK Bit 3: =1: PERMANENT ERROR, =0: NO PERMANENT ERROR Bit 4: =1: TEMPORARY ERROR, =0: NO TEMPORARY ERROR Bit 5: =1: MANUFACTURER SPECIFIC ERROR 1, =0: NO MANUFACTURER SPECIFIC ERROR 1 Bit 6: =1: MANUFACTURER SPECIFIC ERROR 2, =0: NO MANUFACTURER SPECIFIC ERROR 2 Bit 7: =1: MANUFACTURER SPECIFIC ERROR 3, =0: NO MANUFACTURER SPECIFIC ERROR 3
4x10009 3x10009 I:10008 R/O FUTURE VALUE METER 1	UINT16: Returns a future value of the meter #1 This UINT16 holding register is reserved for future use.
4x10010 3x10010 I:10009 R/O COMMUNICATION STATE METER 1	UINT16: Returns the current state of the communication with the meter #1 This UINT16 holding register hold the current state of the communication between the MBUS gateway and the meter with the following states: **=0 - Meter isn't configured!**: This value shows, that this meter slot is currently not configured in the MBUS gateway **=1 - Meter isn't normalized!**: This value shows, that the configured meter doesn't answer to the addressing command. Either via primary addressing or via secondary addressing mode. This depends, how the meter was configured **=2 - Meter isn't read!**: This value shows, that the configured meter has answered to the addressing command but there are problems by reading all data from the meter. So the meter data is not valid any more **=3 - Values are valid!**: This value shows, that the configured meter has answered to the addressing command and has answered correctly to the readout commands and the reading of all data from the meter was successful. So the meter data in the MODBUS register is valid **=1000..65535 – meter readout is asynchron!**: If the mapping of the received MBUS frame differ to the mapping from the configuration, this value shows the position of the first asynchron received data: Value=1000+MBUS record*1000+MBUS byte Index within record. If MBUS record number >=64, the received value is always 1000+64*1000+MBUS byte index.

Register	Description
4x10011-12 3x10011-12 I:10010-11 R/O ID NUMBER METER 2	UINT32: Returns the current serial number of the meter #2 as a 32 bit unsigned integer value Refer to meter #1 description
4x10013-14 3x10013-14 I:10012-13 R/O MANUFACTURER METER 2	UINT32->ASCII: Returns the current manufacturer of the meter #2 as a 32 bit unsigned integer value Refer to meter #1 description
4x10015 3x10015 I:10014 R/O VERSION METER 2	UINT16: Returns the current version of the meter #2 Refer to meter #1 description
4x10016 3x10016 I:10015 R/O MEDIUM METER 2	UINT16: Returns the current medium of the meter #2 Refer to meter #1 description
4x10017 3x10017 I:10016 R/O ACCESS COUNTER METER 2	UINT16: Returns the current access counter of the meter #2 Refer to meter #1 description
4x10018 3x10018 I:10017 R/O STATUS METER 2	UINT16: Returns the current status of the meter #2 Refer to meter #1 description
4x10019 3x10019 I:10018 R/O FUTURE VALUE METER 2	UINT16: Returns a future value of the meter #2 Refer to meter #1 description
4x10020 3x10020 I:10019 R/O COMMUNICATION STATE METER 2	UINT16: Returns the current state of the communication with the meter #2 Refer to meter #1 description
...	

Register	Description
...	
4x10071-72 3x10071-72 I:10070-71 R/O ID NUMBER METER 8	UINT32: Returns the current serial number of the meter #8 as a 32 bit unsigned integer value Refer to meter #1 description
4x10073-74 3x10073-74 I:10072-73 R/O MANUFACTURER METER 8	UINT32->ASCII: Returns the current manufacturer of the meter #8 as a 32 bit unsigned integer value Refer to meter #1 description
4x10075 3x10075 I:10074 R/O VERSION METER 8	UINT16: Returns the current version of the meter #8 Refer to meter #1 description
4x10076 3x10076 I:10075 R/O MEDIUM METER 8	UINT16: Returns the current medium of the meter #8 Refer to meter #1 description
4x10077 3x10077 I:10076 R/O ACCESS COUNTER METER 8	UINT16: Returns the current access counter of the meter #8 Refer to meter #1 description
4x10078 3x10078 I:10077 R/O STATUS METER 8	UINT16: Returns the current status of the meter #8 Refer to meter #1 description
4x10079 3x10079 I:10078 R/O FUTURE VALUE METER 8	UINT16: Returns a future value of the meter #8 Refer to meter #1 description
4x10080 3x10080 I:10079 R/O COMMUNICATION STATE METER 8	UINT16: Returns the current state of the communication with the meter #8 Refer to meter #1 description
...	

Register	Description
...	
4x10231-232 3x10231-232 I:10230-231 R/O ID NUMBER METER 24	UINT32: Returns the current serial number of the meter #24 as a 32 bit unsigned integer value Refer to meter #1 description
4x10233-234 3x10233-234 I:10232-233 R/O MANUFACTURER METER 24	UINT32->ASCII: Returns the current manufacturer of the meter #24 as a 32 bit unsigned integer value Refer to meter #1 description
4x10235 3x10235 I:10234 R/O VERSION METER 24	UINT16: Returns the current version of the meter #24 Refer to meter #1 description
4x10236 3x10236 I:10235 R/O MEDIUM METER 24	UINT16: Returns the current medium of the meter #24 Refer to meter #1 description
4x10237 3x10237 I:10236 R/O ACCESS COUNTER METER 24	UINT16: Returns the current access counter of the meter #24 Refer to meter #1 description
4x10238 3x10238 I:10237 R/O STATUS METER 24	UINT16: Returns the current status of the meter #24 Refer to meter #1 description
4x10239 3x10239 I:10238 R/O FUTURE VALUE METER 24	UINT16: Returns a future value of the meter #24 Refer to meter #1 description
4x10240 3x10240 I:10239 R/O COMMUNICATION STATE METER 24	UINT16: Returns the current state of the communication with the meter #24 Refer to meter #1 description
...	

Register	Description
...	
4x10471-472 3x10471-472 I:10470-471 R/O ID NUMBER METER 48	UINT32: Returns the current serial number of the meter #48 as a 32 bit unsigned integer value Refer to meter #1 description
4x10473-474 3x10473-474 I:10472-473 R/O MANUFACTURER METER 48	UINT32->ASCII: Returns the current manufacturer of the meter #48 as a 32 bit unsigned integer value Refer to meter #1 description
4x10475 3x10475 I:10474 R/O VERSION METER 48	UINT16: Returns the current version of the meter #48 Refer to meter #1 description
4x10476 3x10476 I:10475 R/O MEDIUM METER 48	UINT16: Returns the current medium of the meter #48 Refer to meter #1 description
4x10477 3x10477 I:10476 R/O ACCESS COUNTER METER 48	UINT16: Returns the current access counter of the meter #48 Refer to meter #1 description
4x10478 3x10478 I:10477 R/O STATUS METER 48	UINT16: Returns the current status of the meter #48 Refer to meter #1 description
4x10479 3x10479 I:10478 R/O FUTURE VALUE METER 48	UINT16: Returns a future value of the meter #48 Refer to meter #1 description
4x10480 3x10480 I:10479 R/O COMMUNICATION STATE METER 48	UINT16: Returns the current state of the communication with the meter #48 Refer to meter #1 description
...	

Register	Description
...	
4x10631-632 3x10631-632 I:10630-631 R/O ID NUMBER METER 64	UINT32: Returns the current serial number of the meter #64 as a 32 bit unsigned integer value Refer to meter #1 description
4x10633-634 3x10633-634 I:10632-633 R/O MANUFACTURER METER 64	UINT32->ASCII: Returns the current manufacturer of the meter #64 as a 32 bit unsigned integer value Refer to meter #1 description
4x10635 3x10635 I:10634 R/O VERSION METER 64	UINT16: Returns the current version of the meter #64 Refer to meter #1 description
4x10636 3x10636 I:10635 R/O MEDIUM METER 64	UINT16: Returns the current medium of the meter #64 Refer to meter #1 description
4x10637 3x10637 I:10636 R/O ACCESS COUNTER METER 64	UINT16: Returns the current access counter of the meter #64 Refer to meter #1 description
4x10638 3x10638 I:10637 R/O STATUS METER 64	UINT16: Returns the current status of the meter #64 Refer to meter #1 description
4x10639 3x10639 I:10638 R/O FUTURE VALUE METER 64	UINT16: Returns a future value of the meter #64 Refer to meter #1 description
4x10640 3x10640 I:10639 R/O COMMUNICATION STATE METER 64	UINT16: Returns the current state of the communication with the meter #64 Refer to meter #1 description

43.17.4 MODBUS registers for special configuration

This registers hold special information for the converter:

Register	Description
4x65231 3x65231 I:65230 R/W MBUS BAUDRATE	UINT16: The baud rate for the MBUS interface. Parity is always EVEN, ONE stop bit is used. The following baud rates are available: 300,600,900,1200,2400,4800,9600,19200,38400,57600 All other values are interpreted as 2400 baud. HINT: After writing a new value to this register a reboot is necessary to activate the new settings
4x65232 3x65232 I:65231 R/W MBUS QUERY TIMEOUT	UINT16: The query timeout for the MBUS polling process. This value defines the timeout between two query cycles in the gateway. Usually the gateway communicates with all configured meters sequentially. After finishing the data readout for the last meter, the gateway pauses for this defined interval in seconds. This values are used: Value 65535 or values 0..5 defines ~5s pause. Values 6 to 65534: defines 6 to 65534 seconds of pause, before the next polling cycle will start. HINT: After writing a new value to this register a reboot is necessary to activate the new settings
4x65233 3x65233 I:65232 R/W MBUS POLL DELAY	UINT16: The poll delay for the MBUS polling process. This value defines a general pause after the readout of a configured meter before the readout of the next meter starts. In the past we discovered that there are many meters out in the market, which need a special treatment in the timing. e.g. very old KAMSTRUP meters need often two readout cycles with a gap of at least 10-15 seconds. This is non standard to the MBUS. Or other meters have problems with secondary addressing, if there is a too small gap between the readout. So we introduced this new parameter: This timeout defines the pause after finishing reading of a meter and starting reading the next meter. In the previous firmware versions this timeout was fixed to 250ms gap, which was ok for 99% of the meter readout on the markets. But some meter fail to process this little gap. The values is interpreted as follows: Value 1..30: Gap time 1 seconds to 30 seconds Value 101..400: Gaptime=(Value-100)*0.1s → 0.1s .. 30s e.g. 105 → 0.5s Value 65535: Gap time is 1 second Value 65534: Gap time is 250ms Value 65533: Gap time is 500ms Value 65532: Gap time is7250ms All other values: Gap time is 1000ms HINT: After writing a new value to this register a reboot is necessary to activate the new settings

43.18 Additional ASCII commands

Here you will find only the additional ASCII commands especially for this IO module. Please refer to the description of of the standard commands for more details about the available basic ASCII commands.

Please refer to the external document for detailed documentation of the current ASCII commands for this IO module:

RESI-L-MBUSx-SIO-ETH-MODBUS+ASCII-ENxx.pdf

44 RESI-KNX-SIO, RESI-KNX-ETH

44.1 General information

With the RESI-KNX-SIO gateway, the KNX can be integrated in almost every system with a RS232 or RS485 interface and a MODBUS/RTU master protocol or serial ASCII text based protocol. The RESI-KNX-ETH gateway offers an integration with MODBUS/TCP server protocol over Ethernet.

The gateway is a serial interface for connection to the KNX with an integrated 2 wire KNX bus-coupler. The time-critical KNX communication is done from the gateway itself. The gateway is configured with our MODBUSConfigurator software and maps the incoming KNX telegrams to MODBUS holding registers. When the host writes to a MODBUS holding register, the gateway generates the corresponding KNX telegram. When the gateway receives a KNX telegram, it maps and converts the incoming data to the specific MODBUS holding registers for readout through a host.

To control our KNX converters you need a host system with a serial interface (RS232 or RS485), which is able to send ASCII command strings and which can receive ASCII characters. This feature is implemented in almost any media control system like CRESTRON®, AMX® or CONTROL4®. But almost every standard PLC can handle serial ASCII interfaces. Therefore our converter can be integrated everywhere. If the host system offers a MODBUS/RTU master or MODBUS/TCP client interface, our converter can be controlled via MODBUS holding registers.

This series of IO modules offer the following features:

- Easy integration of a complete KNX bus system
- MODBUS/RTU slave or MODBUS/TCP server protocol
- Additional commands with plain ASCII texts
- KNX and host interface are galvanically isolated
- Supports all 65535 KNX group addresses
- Supports all KNX DPT types
- Integrated KNX bus-coupler
- RESI-xxx-SIO: Galvanic isolated RS232 and RS485 interface for communication with a host system
- RESI-xxx-ETH: Galvanic isolated Ethernet interface for communication with a host system

Figure: Our serial KNX module

Figure: Our Ethernet IO module

44.2 Technical specification

Beside the basic technical data, which fulfil all of our IO modules, this IO modules meet the following technical specifications:

Power consumption	
RESI-KNX-SIO	<0.6W
RESI-KNX-ETH	<0.9W
Product housing	
RESI-KNX-SIO	CEM17
RESI-KNX-ETH	CEM35
Product weight	
RESI-KNX-SIO	57g
RESI-KNX-ETH	91g
KNX bus interface	
Protocol	KNX
Baud rate	9600kBit/s
Cable connection	via terminals
Galvanic isolation	Yes
LED indicator	Yes
Default serial settings	
Baud rate	via DIP switch
Parity	none
Stopbits	one
UnitID	255
Default Ethernet settings	
IP address	192.168.0.220
IP mask	255.255.255.0
gateway	192.168.0.1
UnitID	255
User	RESI
password	RESI

44.3 Additional terminals & LED states

KNX INTERFACE	KNX bus connector	
	One 3 pin terminal blocks	
	Terminal type:	USLIM
	K+:	KNX+ bus wire (red)
	K-:	KNX- bus wire (black)
Pin layout	K+:	KNX+ bus wire (red)
	N/C:	not connected
	K-:	KNX- bus wire (black)
KNX	If there is bus communication on the KNX, this LED is on	
	Otherwise this LED is OFF	

44.4 RESI-KNX-SIO: Connection diagram

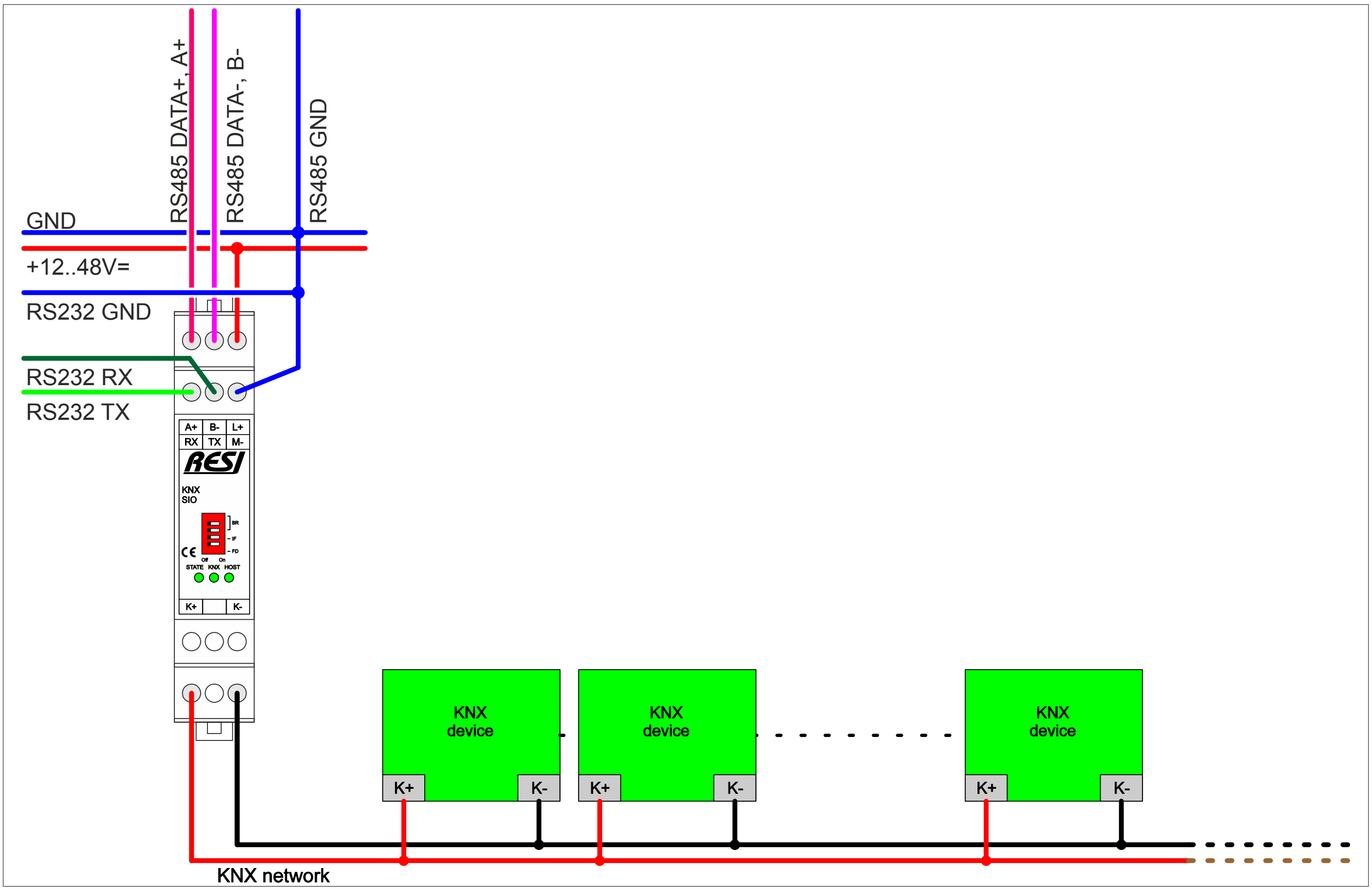


Figure: Connecting a KNX bus system to the RESI-KNX-SIO gateway

44.5 RESI-KNX-ETH: Connection diagram

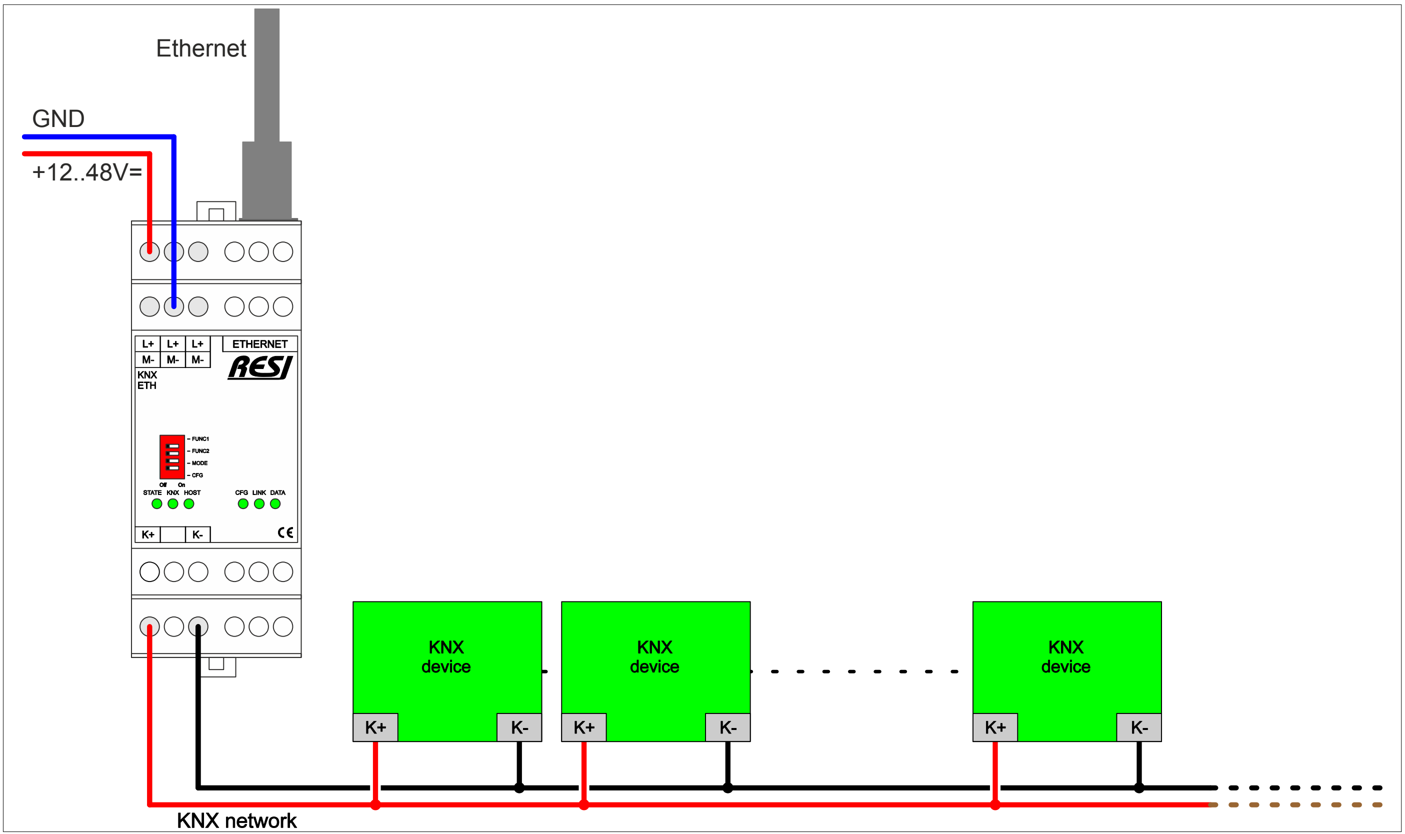
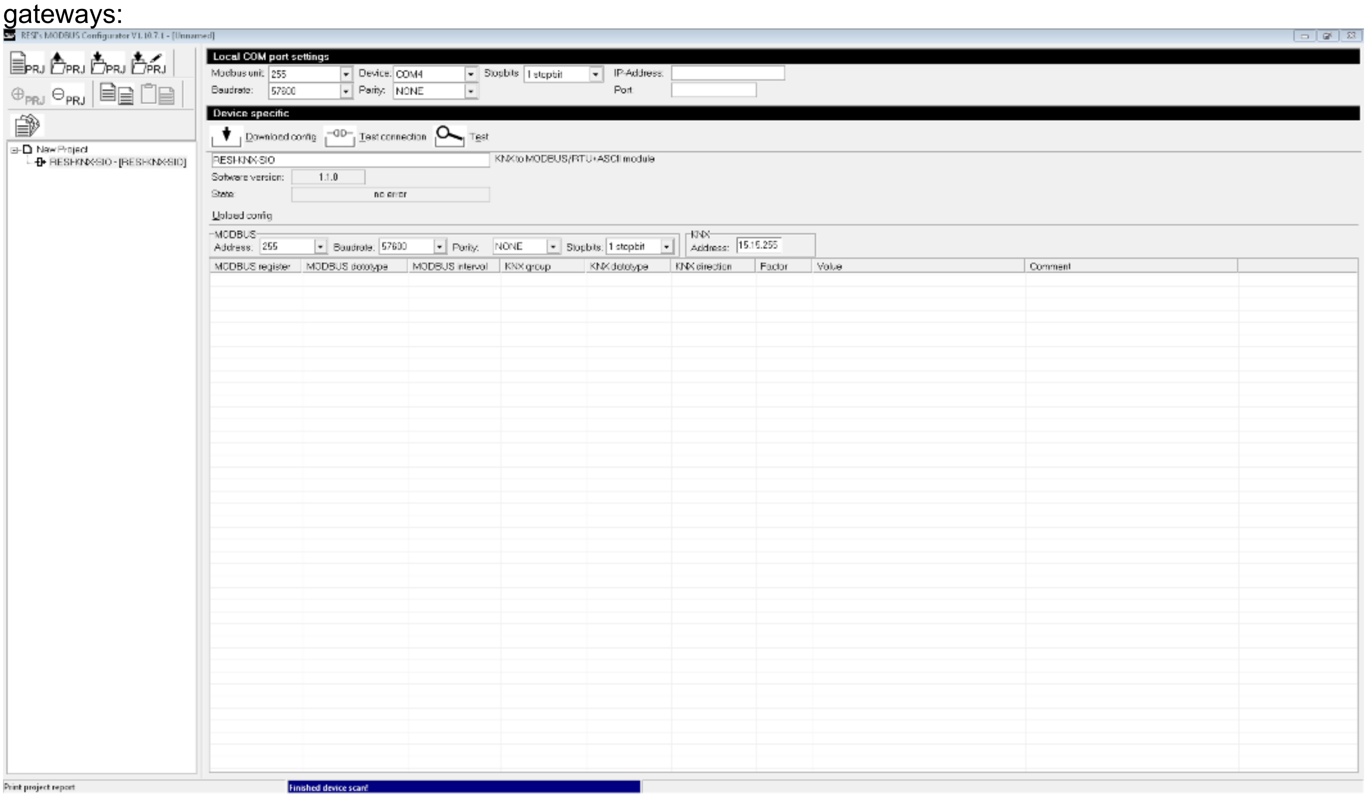


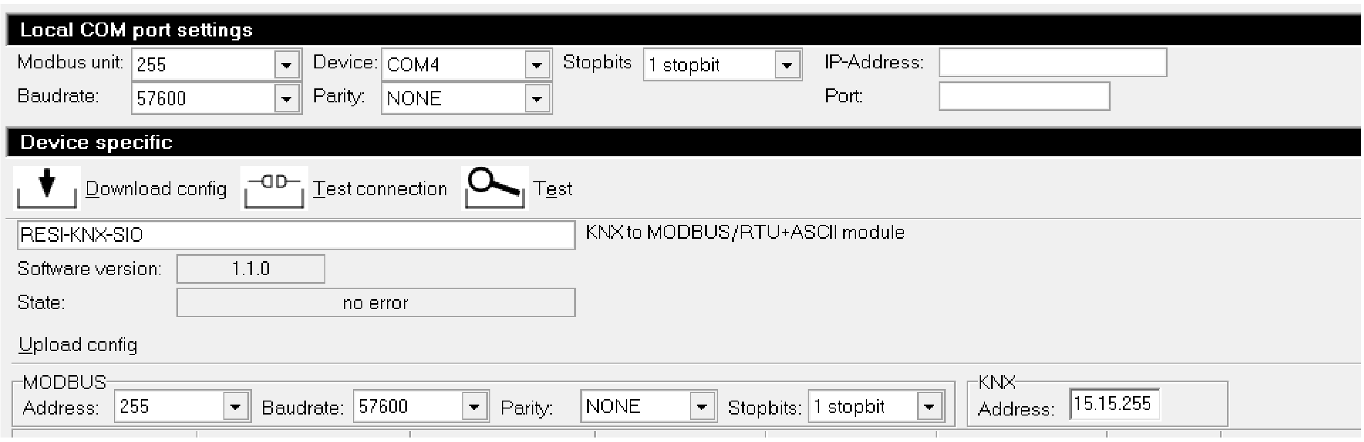
Figure: Connecting a KNX bus system to the RESI-KNX-ETH gateway

44.6 Configuration with MODBUSConfigurator software

Download our free software from our homepage www.RESI.cc and install it on your computer. After you have successfully established a connection, you will see the following picture for the RESI-KNX-SIO or RESI-KNX-ETH gateways:



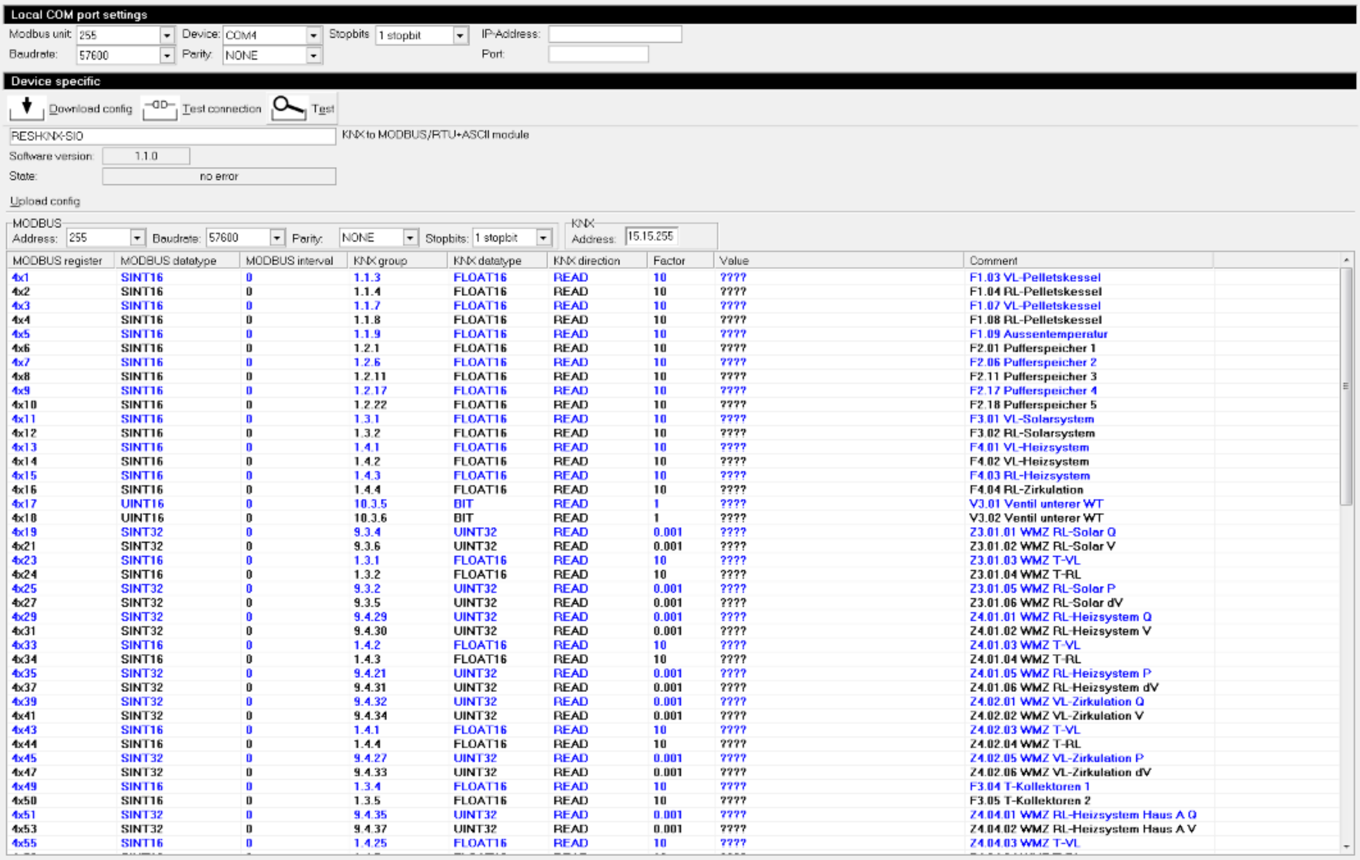
In the section device specific you will find the following functions:



- Button “Download config”: If you change the MODBUS/RTU slave address, or the serial parameters or KNX address or if you change the KNX mapping, you have to download the new configuration to the gateway to activate the changes.
- Button “Upload config”: With this button you can upload the complete mapping of the converter into the software. But remember, the comments are not stored into the gateway, so this information is lost, if you upload the mapping from a gateway!
- Button “Test connection”: This button tests, if the software can communicate with the gateway or not.
- Button “Test”: This button activates/deactivates a test function, which will show all current contents of the configured MODBUS registers in the converter. In this test mode, you can also write to MODBUS/RTU holding registers and generate a KNX telegram on the KNX bus. The software polls every 5 seconds all configured MODBUS registers.

44.6.1 The configuration table

In the device specific region you will see a table with the current configuration of the MODBUS/KNX mapping:



MODBUS register	MODBUS datatype	MODBUS interval	KNX group	KNX datatype	KNX direction	Factor	Value	Comment
4x1	SINT16	0	1.1.3	FLOAT16	READ	10	????	F1.03 VL-Pelletskessel
4x2	SINT16	0	1.1.4	FLOAT16	READ	10	????	F1.04 RL-Pelletskessel
4x3	SINT16	0	1.1.7	FLOAT16	READ	10	????	F1.07 VL-Pelletskessel
4x4	SINT16	0	1.1.8	FLOAT16	READ	10	????	F1.08 RL-Pelletskessel
4x5	SINT16	0	1.1.9	FLOAT16	READ	10	????	F1.09 Aussentemperatur
4x6	SINT16	0	1.2.1	FLOAT16	READ	10	????	F2.01 Pufferspeicher 1
4x7	SINT16	0	1.2.6	FLOAT16	READ	10	????	F2.06 Pufferspeicher 2
4x8	SINT16	0	1.2.11	FLOAT16	READ	10	????	F2.11 Pufferspeicher 3
4x9	SINT16	0	1.2.17	FLOAT16	READ	10	????	F2.17 Pufferspeicher 4
4x10	SINT16	0	1.2.22	FLOAT16	READ	10	????	F2.18 Pufferspeicher 5
4x11	SINT16	0	1.3.1	FLOAT16	READ	10	????	F3.01 VL-Solarsystem
4x12	SINT16	0	1.3.2	FLOAT16	READ	10	????	F3.02 RL-Solarsystem
4x13	SINT16	0	1.4.1	FLOAT16	READ	10	????	F4.01 VL-Heizsystem
4x14	SINT16	0	1.4.2	FLOAT16	READ	10	????	F4.02 VL-Heizsystem
4x15	SINT16	0	1.4.3	FLOAT16	READ	10	????	F4.03 RL-Heizsystem
4x16	SINT16	0	1.4.4	FLOAT16	READ	10	????	F4.04 RL-Zirkulation
4x17	UINT16	0	10.3.5	BIT	READ	1	????	V3.01 Ventil unterer WT
4x18	UINT16	0	10.3.6	BIT	READ	1	????	V3.02 Ventil unterer WT
4x19	SINT32	0	9.3.4	UINT32	READ	0.001	????	Z3.01.01 WMZ RL-Solar Q
4x21	SINT32	0	9.3.6	UINT32	READ	0.001	????	Z3.01.02 WMZ RL-Solar V
4x23	SINT16	0	1.3.1	FLOAT16	READ	10	????	Z3.01.03 WMZ T-VL
4x24	SINT16	0	1.3.2	FLOAT16	READ	10	????	Z3.01.04 WMZ T-RL
4x25	SINT32	0	9.3.2	UINT32	READ	0.001	????	Z3.01.05 WMZ RL-Solar P
4x27	SINT32	0	9.3.5	UINT32	READ	0.001	????	Z3.01.06 WMZ RL-Solar dV
4x29	SINT32	0	9.4.29	UINT32	READ	0.001	????	Z4.01.01 WMZ RL-Heizsystem Q
4x31	SINT32	0	9.4.30	UINT32	READ	0.001	????	Z4.01.02 WMZ RL-Heizsystem V
4x33	SINT16	0	1.4.2	FLOAT16	READ	10	????	Z4.01.03 WMZ T-VL
4x34	SINT16	0	1.4.3	FLOAT16	READ	10	????	Z4.01.04 WMZ T-RL
4x35	SINT32	0	9.4.21	UINT32	READ	0.001	????	Z4.01.05 WMZ RL-Heizsystem P
4x37	SINT32	0	9.4.31	UINT32	READ	0.001	????	Z4.01.06 WMZ RL-Heizsystem dV
4x39	SINT32	0	9.4.32	UINT32	READ	0.001	????	Z4.02.01 WMZ VL-Zirkulation Q
4x41	SINT32	0	9.4.34	UINT32	READ	0.001	????	Z4.02.02 WMZ VL-Zirkulation V
4x43	SINT16	0	1.4.1	FLOAT16	READ	10	????	Z4.02.03 WMZ T-VL
4x44	SINT16	0	1.4.4	FLOAT16	READ	10	????	Z4.02.04 WMZ T-RL
4x45	SINT32	0	9.4.27	UINT32	READ	0.001	????	Z4.02.05 WMZ VL-Zirkulation P
4x47	SINT32	0	9.4.33	UINT32	READ	0.001	????	Z4.02.06 WMZ VL-Zirkulation dV
4x49	SINT16	0	1.3.4	FLOAT16	READ	10	????	F3.04 T-Kollektoren 1
4x50	SINT16	0	1.3.5	FLOAT16	READ	10	????	F3.05 T-Kollektoren 2
4x51	SINT32	0	9.4.35	UINT32	READ	0.001	????	Z4.04.01 WMZ RL-Heizsystem Haus A Q
4x53	SINT32	0	9.4.37	UINT32	READ	0.001	????	Z4.04.02 WMZ RL-Heizsystem Haus A V
4x55	SINT16	0	1.4.25	FLOAT16	READ	10	????	Z4.04.03 WMZ T-VL

A mapping entry consists out of the following entries:

- **MODBUS register**: The number of the single holding register or the start index of the holding registers, if more than one register is used, into which the incoming KNX data is stored or from which the outgoing KNX value is read before sending KNX data.
- **MODBUS datatype**: The data type for the MODBUS registers. It defines how the gateway converts KNX data into MODBUS data and how many MODBUS registers are used to store the KNX data.
- **MODBUS interval**: This is for future use and defines the time interval in seconds to request KNX data from the KNX bus automatically. At the moment this feature is not used.
- **KNX group**: this defines the KNX group address, which is used to send/receive KNX data for this MODBUS registers.
- **KNX datatype**: This defines the KNX datatype, which is used to send/receive data with the specific KNX group on the KNX bus.
- **KNX direction**: This defines in which direction the communication is done: You can read, write or read/write data from/to the KNX bus.
- **Factor**: This defines a factor which is used to multiply incoming KNX data before the data is stored into the MODBUS registers. For outgoing KNX telegrams the data of the MODBUS registers is divided by this factor, before sending to the KNX bus. A zero value defines unused.
- **Value**: If Test mode is active, here you will find the current received value for this mapping entry or you can write onto this value to send aKNX value to the KNX system.
- **Comment**: This defines a user specific comment for this mapping line. This is only for documentation reasons and is not stored into the gateway. It is only stored on the PC if you save your project. If you upload a configuration from a gateway into the software, this comment is lost!

44.6.2 The context menu

If you right click into the table, a local context menu will appear with the following entries:

- Edit entry...
- Add entry...
- Insert entry...
- Copy entry...
- Delete selected entries
- Clear complete list
- Renumber MODBUS registers
- Renumber KNX groups
- Sort MODBUS register
- Sort KNX group
- Find MODBUS register
- Find KNX Group
- Find comment
- Copy to clipboard
- Paste from clipboard

44.6.2.1 Context menu: Add entry

Entry "Add entry…": If you select this item, a new empty configuration line is added to the configuration list.

MODBUS register	MODBUS datatype	MODBUS interval	KNX group	KNX datatype	KNX direction	Factor	Value	Comment
4x1	UINT16	0	0.0.0	BIT	READ-WRITE	1.00	????	no comment
4x2	UINT16	0	0.0.1	BIT	READ-WRITE	1.00	????	no comment

44.6.2.2 Context menu: Delete selected lines

Entry "Delete selected lines…": First select one or more lines in your configuration table. To select more than one line press and hold the STRG key or the SHIFT key, and then select other lines.

MODBUS register	MODBUS datatype	MODBUS interval	KNX group	KNX datatype	KNX direction	Factor	Value	Comment
4x1	UINT16	0	0.0.0	BIT	READ-WRITE	1.00	????	no comment
4x2	UINT16	0	0.0.1	BIT	READ-WRITE	1.00	????	no comment
4x3	UINT16	0	0.0.2	BIT	READ-WRITE	1.00	????	no comment
4x4	UINT16	0	0.0.		TE	1.00	????	no comment

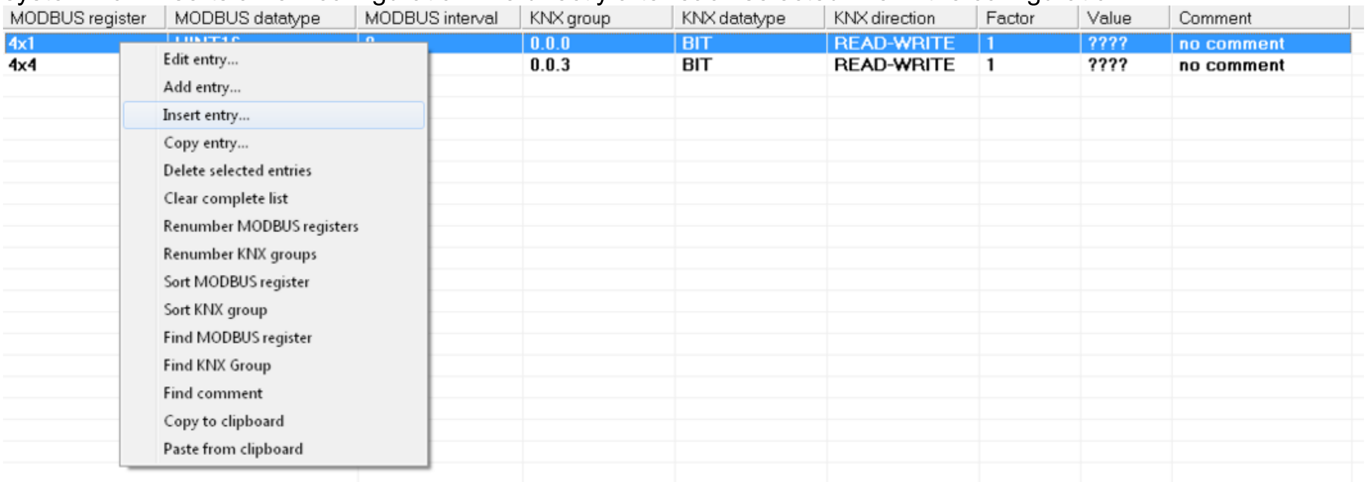
- Edit entry...
- Add entry...
- Insert entry...
- Copy entry...
- Delete selected entries
- Clear complete list
- Renumber MODBUS registers
- Renumber KNX groups
- Sort MODBUS register
- Sort KNX group
- Find MODBUS register
- Find KNX Group
- Find comment
- Copy to clipboard
- Paste from clipboard

Then open the context menu and select the function "Delete selected lines…". The system will delete the selected lines from the configuration list. The result will be like this:

MODBUS register	MODBUS datatype	MODBUS interval	KNX group	KNX datatype	KNX direction	Factor	Value	Comment
4x1	UINT16	0	0.0.0	BIT	READ-WRITE	1.00	????	no comment
4x4	UINT16	0	0.0.3	BIT	READ-WRITE	1.00	????	no comment

44.6.2.3 Context menu: Insert entry

Entry “Insert entry…”: First select one or more lines. Then choose this function from the local context menu. The system now inserts a new configuration line directly after each selected line in the configuration.

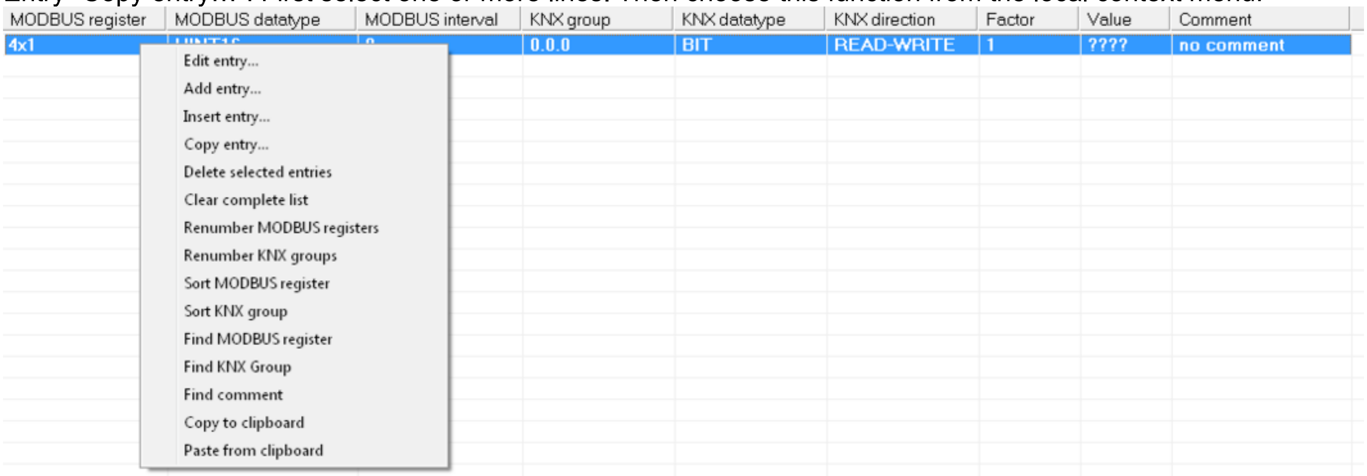


The result will be like this:

MODBUS register	MODBUS datatype	MODBUS interval	KNX group	KNX datatype	KNX direction	Factor	Value	Comment
4x1	UINT16	0	0.0.0	BIT	READ-WRITE	1	????	no comment
4x2	UINT16	0	0.0.1	BIT	READ-WRITE	1	????	no comment
4x4	UINT16	0	0.0.3	BIT	READ-WRITE	1	????	no comment

44.6.2.4 Context menu: Copy entry

Entry “Copy entry…”: First select one or more lines. Then choose this function from the local context menu:



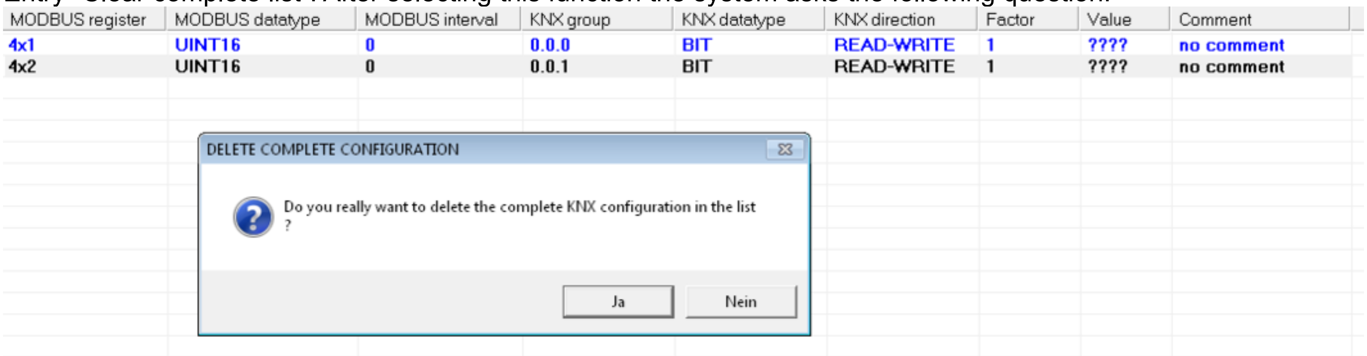
The system copies all selected lines and adds for each selected line a new entry to the configuration. The result looks like this:

MODBUS register	MODBUS datatype	MODBUS interval	KNX group	KNX datatype	KNX direction	Factor	Value	Comment
4x1	UINT16	0	0.0.0	BIT	READ-WRITE	1	????	no comment
4x2	UINT16	0	0.0.1	BIT	READ-WRITE	1	????	no comment

As you will notice, the system auto increments the MODBUS/RTU register index depending to the configured MODBUS datatype. The same increment is done with the KNX group address.

44.6.2.5 Context menu: Clear complete list

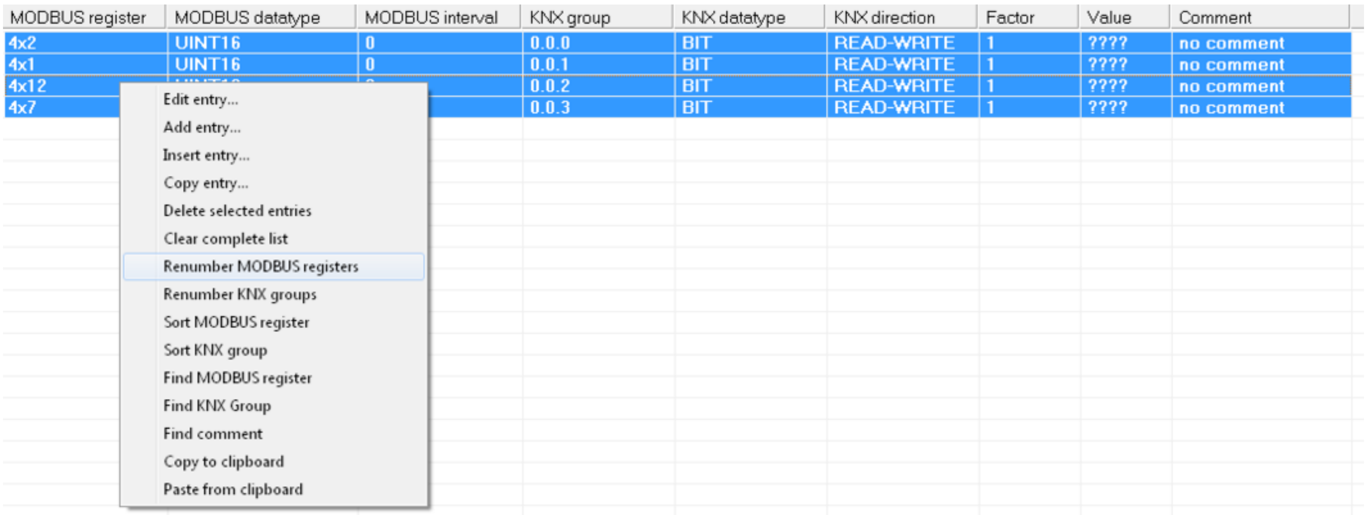
Entry “Clear complete list”: After selecting this function the system asks the following question:



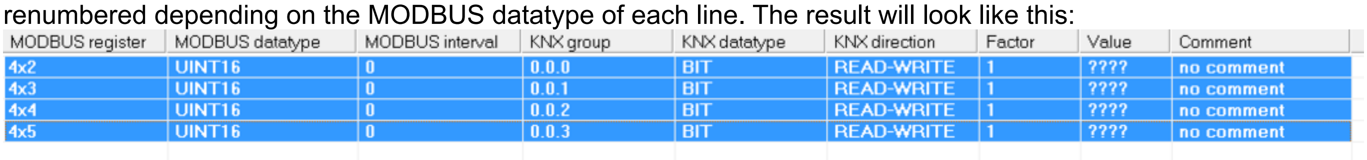
If you answer with YES, all entries of your configuration will be deleted forever! The answer NO cancels this function.

44.6.2.6 Context menu: Renumber MODBUS registers

Entry “Renumber MODBUS registers”: First select the lines you want to renumber, then select this function from the local context menu:

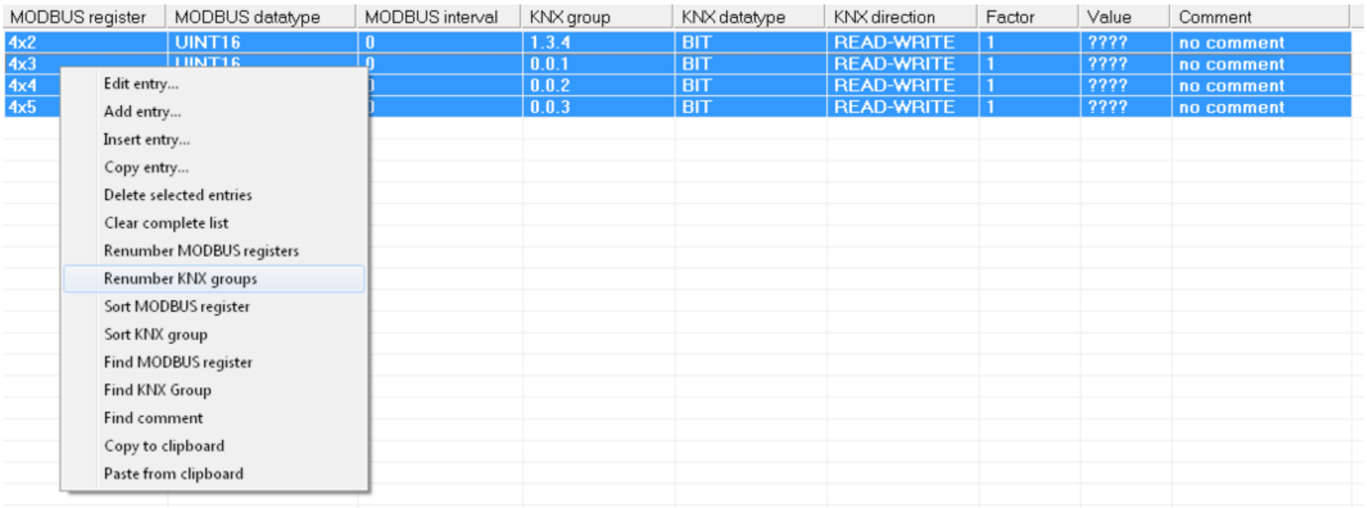


The starting index of the MODBUS register of the first selected line is used for the first entry. The next lines are renumbered depending on the MODBUS datatype of each line. The result will look like this:

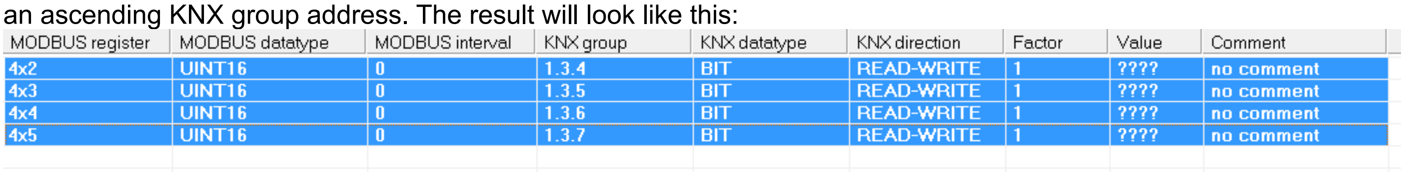


44.6.2.7 Context menu: Renumber KNX groups

Entry “Renumber KNX groups”: First select the lines you want to renumber. Then select this function from the local context menu.



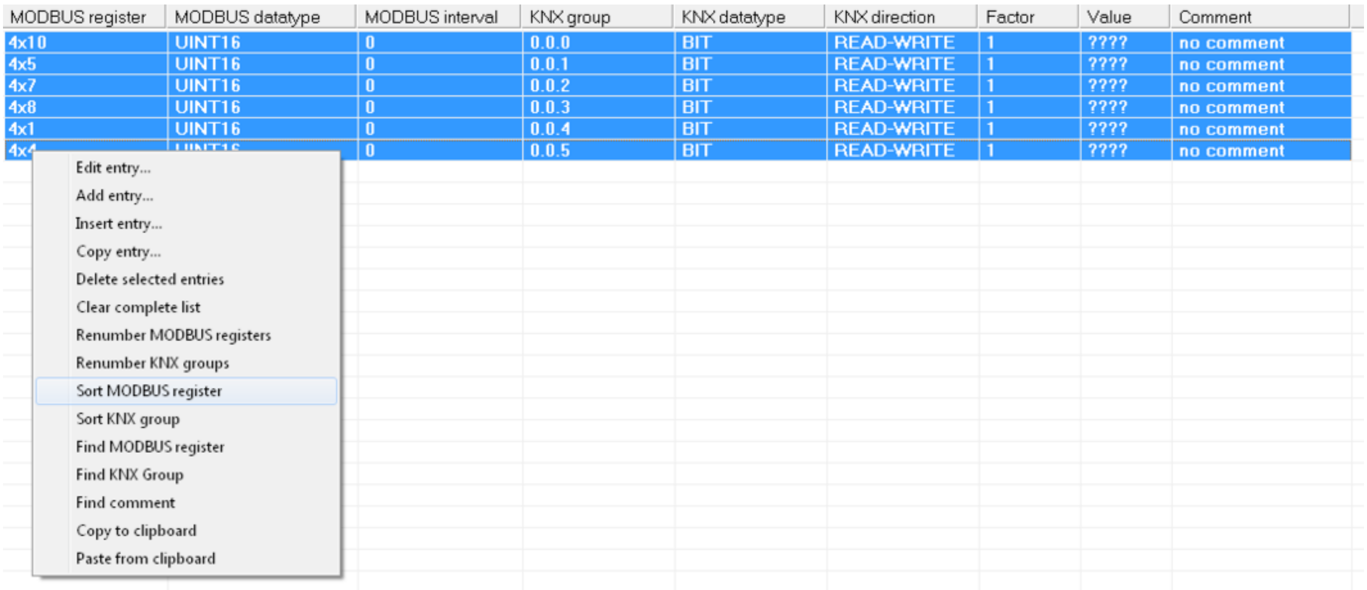
The KNX group address of the first selected line is used for the first entry. All selected lines are now renumbered with an ascending KNX group address. The result will look like this:



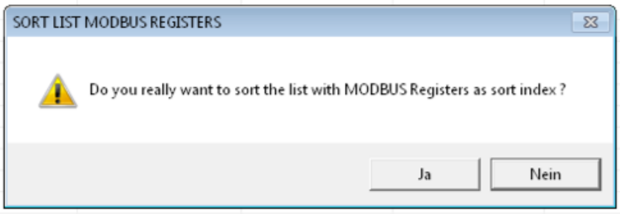
MODBUS register	MODBUS datatype	MODBUS interval	KNX group	KNX datatype	KNX direction	Factor	Value	Comment
4x2	UINT16	0	1.3.4	BIT	READ-WRITE	1	????	no comment
4x3	UINT16	0	1.3.5	BIT	READ-WRITE	1	????	no comment
4x4	UINT16	0	1.3.6	BIT	READ-WRITE	1	????	no comment
4x5	UINT16	0	1.3.7	BIT	READ-WRITE	1	????	no comment

44.6.2.8 Context menu: Sort MODBUS register

Entry “Sort MODBUS register”: First select one or more lines you want to sort. Then select this function from the local context menu.



MODBUS register	MODBUS datatype	MODBUS interval	KNX group	KNX datatype	KNX direction	Factor	Value	Comment
4x10	UINT16	0	0.0.0	BIT	READ-WRITE	1	????	no comment
4x5	UINT16	0	0.0.1	BIT	READ-WRITE	1	????	no comment
4x7	UINT16	0	0.0.2	BIT	READ-WRITE	1	????	no comment
4x8	UINT16	0	0.0.3	BIT	READ-WRITE	1	????	no comment
4x1	UINT16	0	0.0.4	BIT	READ-WRITE	1	????	no comment
4x4	UINT16	0	0.0.5	BIT	READ-WRITE	1	????	no comment



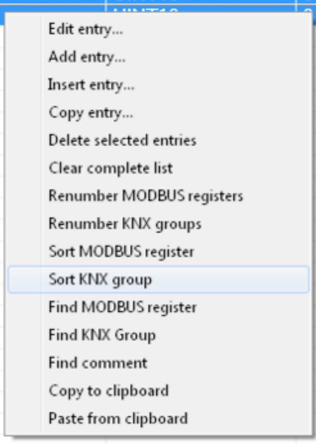
If you answer the above question with YES, the system sorts the select line using the MODBUS address as a sort key in ascending order. The result will look like this:

MODBUS register	MODBUS datatype	MODBUS interval	KNX group	KNX datatype	KNX direction	Factor	Value	Comment
4x1	UINT16	0	0.0.4	BIT	READ-WRITE	1	????	no comment
4x4	UINT16	0	0.0.5	BIT	READ-WRITE	1	????	no comment
4x5	UINT16	0	0.0.1	BIT	READ-WRITE	1	????	no comment
4x7	UINT16	0	0.0.2	BIT	READ-WRITE	1	????	no comment
4x8	UINT16	0	0.0.3	BIT	READ-WRITE	1	????	no comment
4x10	UINT16	0	0.0.0	BIT	READ-WRITE	1	????	no comment

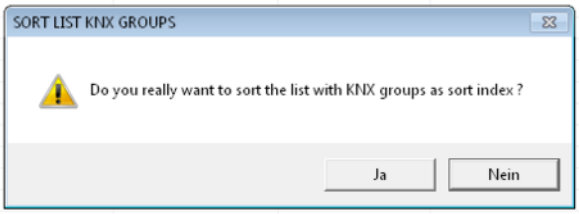
44.6.2.9 Context menu: Sort KNX group

Entry “Sort KNX group”: After selecting one or more lines for sorting, choose this function from the local context menu:

MODBUS register	MODBUS datatype	MODBUS interval	KNX group	KNX datatype	KNX direction	Factor	Value	Comment
4x1	UINT16	0	1.5.57	BIT	READ-WRITE	1	????	no comment
4x2	UINT16	0	7.5.255	BIT	READ-WRITE	1	????	no comment
4x3	UINT16	0	1.5.68	BIT	READ-WRITE	1	????	no comment
4x4			7.5.45	BIT	READ-WRITE	1	????	no comment



MODBUS register	MODBUS datatype	MODBUS interval	KNX group	KNX datatype	KNX direction	Factor	Value	Comment
4x1	UINT16	0	1.5.57	BIT	READ-WRITE	1	????	no comment
4x2	UINT16	0	7.5.255	BIT	READ-WRITE	1	????	no comment
4x3	UINT16	0	1.5.68	BIT	READ-WRITE	1	????	no comment
4x4	UINT16	0	7.5.45	BIT	READ-WRITE	1	????	no comment

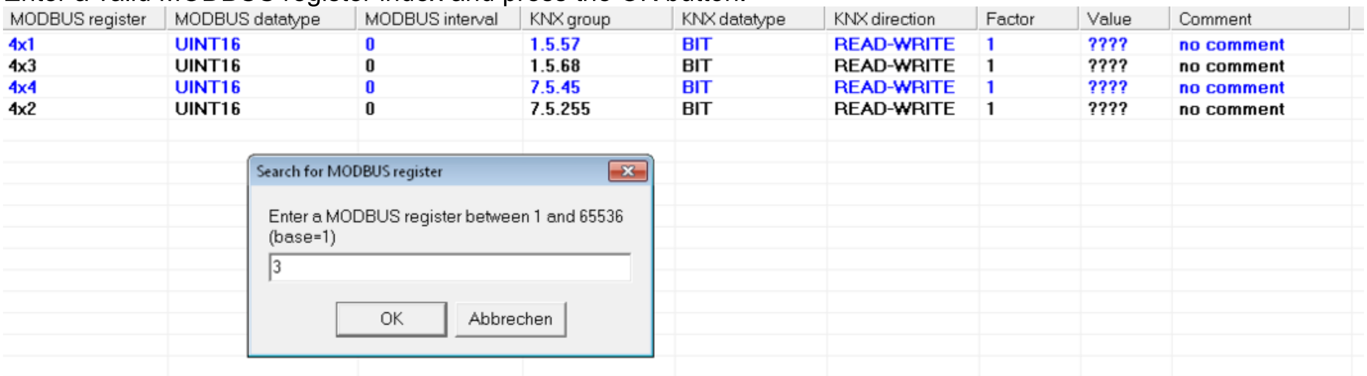


If you answer the above question with YES, the system sorts the selected lines using the KNX group address as a sort index in ascending order. The result will look like this:

MODBUS register	MODBUS datatype	MODBUS interval	KNX group	KNX datatype	KNX direction	Factor	Value	Comment
4x1	UINT16	0	1.5.57	BIT	READ-WRITE	1	????	no comment
4x3	UINT16	0	1.5.68	BIT	READ-WRITE	1	????	no comment
4x4	UINT16	0	7.5.45	BIT	READ-WRITE	1	????	no comment
4x2	UINT16	0	7.5.255	BIT	READ-WRITE	1	????	no comment

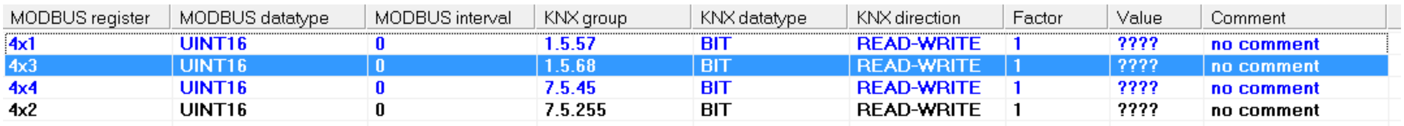
44.6.2.10 Context menu: Find MODBUS register

Entry “Find MODBUS register”: After selecting this function from the local context menu, an input window will appear. Enter a valid MODBUS register index and press the OK button.



MODBUS register	MODBUS datatype	MODBUS interval	KNX group	KNX datatype	KNX direction	Factor	Value	Comment
4x1	UINT16	0	1.5.57	BIT	READ-WRITE	1	????	no comment
4x3	UINT16	0	1.5.68	BIT	READ-WRITE	1	????	no comment
4x4	UINT16	0	7.5.45	BIT	READ-WRITE	1	????	no comment
4x2	UINT16	0	7.5.255	BIT	READ-WRITE	1	????	no comment

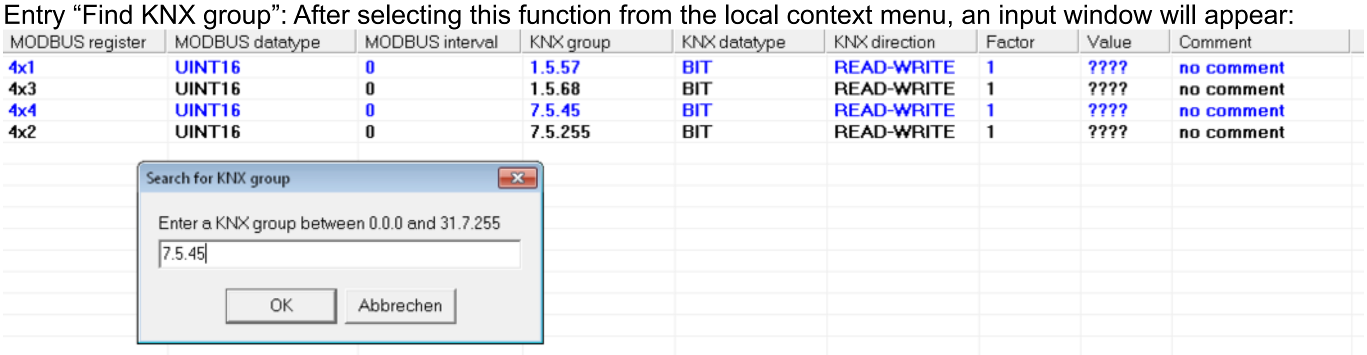
The system will now select all lines, in which the MODBUS register matches the entered number. The result will look like this:



MODBUS register	MODBUS datatype	MODBUS interval	KNX group	KNX datatype	KNX direction	Factor	Value	Comment
4x1	UINT16	0	1.5.57	BIT	READ-WRITE	1	????	no comment
4x3	UINT16	0	1.5.68	BIT	READ-WRITE	1	????	no comment
4x4	UINT16	0	7.5.45	BIT	READ-WRITE	1	????	no comment
4x2	UINT16	0	7.5.255	BIT	READ-WRITE	1	????	no comment

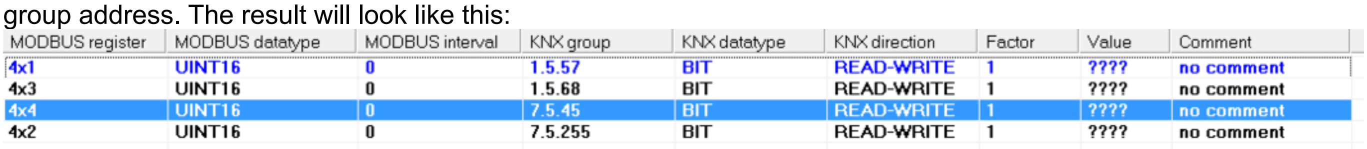
44.6.2.11 Context menu: Find KNX group

Entry “Find KNX group”: After selecting this function from the local context menu, an input window will appear:



MODBUS register	MODBUS datatype	MODBUS interval	KNX group	KNX datatype	KNX direction	Factor	Value	Comment
4x1	UINT16	0	1.5.57	BIT	READ-WRITE	1	????	no comment
4x3	UINT16	0	1.5.68	BIT	READ-WRITE	1	????	no comment
4x4	UINT16	0	7.5.45	BIT	READ-WRITE	1	????	no comment
4x2	UINT16	0	7.5.255	BIT	READ-WRITE	1	????	no comment

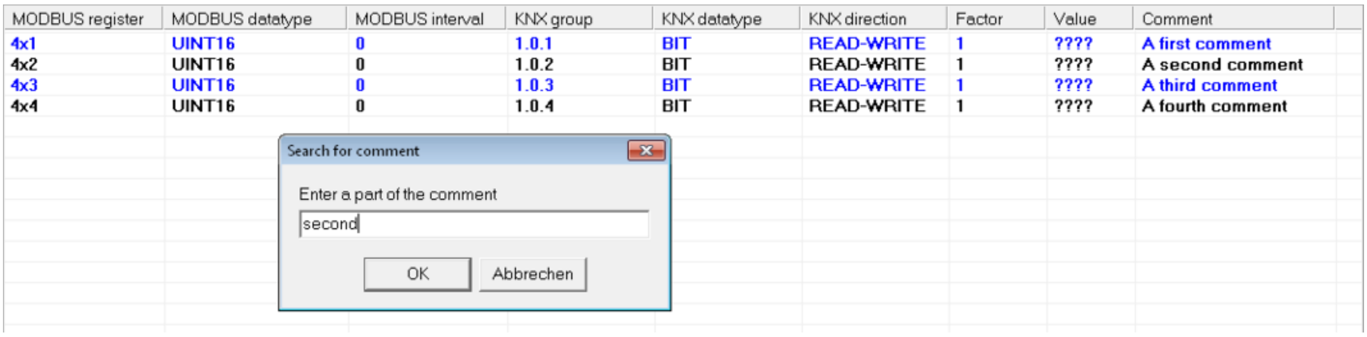
Enter a valid KNX group address. The system now selects all lines in the list, which matches with the entered KNX group address. The result will look like this:



MODBUS register	MODBUS datatype	MODBUS interval	KNX group	KNX datatype	KNX direction	Factor	Value	Comment
4x1	UINT16	0	1.5.57	BIT	READ-WRITE	1	????	no comment
4x3	UINT16	0	1.5.68	BIT	READ-WRITE	1	????	no comment
4x4	UINT16	0	7.5.45	BIT	READ-WRITE	1	????	no comment
4x2	UINT16	0	7.5.255	BIT	READ-WRITE	1	????	no comment

44.6.2.12 Context menu: Find comment

Entry “Find comment”: After selecting this function from the local context menu, an input window will appear. Enter a text part of the desired comment and select the OK button. The system will mark all lines, in which a text part of the comment matches to the entered text.



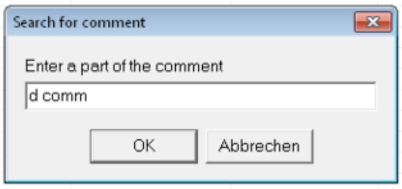
MODBUS register	MODBUS datatype	MODBUS interval	KNX group	KNX datatype	KNX direction	Factor	Value	Comment
4x1	UINT16	0	1.0.1	BIT	READ-WRITE	1	????	A first comment
4x2	UINT16	0	1.0.2	BIT	READ-WRITE	1	????	A second comment
4x3	UINT16	0	1.0.3	BIT	READ-WRITE	1	????	A third comment
4x4	UINT16	0	1.0.4	BIT	READ-WRITE	1	????	A fourth comment

The result will look like this:

MODBUS register	MODBUS datatype	MODBUS interval	KNX group	KNX datatype	KNX direction	Factor	Value	Comment
4x1	UINT16	0	1.0.1	BIT	READ-WRITE	1	????	A first comment
4x2	UINT16	0	1.0.2	BIT	READ-WRITE	1	????	A second comment
4x3	UINT16	0	1.0.3	BIT	READ-WRITE	1	????	A third comment
4x4	UINT16	0	1.0.4	BIT	READ-WRITE	1	????	A fourth comment

Another example:

MODBUS register	MODBUS datatype	MODBUS interval	KNX group	KNX datatype	KNX direction	Factor	Value	Comment
4x1	UINT16	0	1.0.1	BIT	READ-WRITE	1	????	A first comment
4x2	UINT16	0	1.0.2	BIT	READ-WRITE	1	????	A second comment
4x3	UINT16	0	1.0.3	BIT	READ-WRITE	1	????	A third comment
4x4	UINT16	0	1.0.4	BIT	READ-WRITE	1	????	A fourth comment

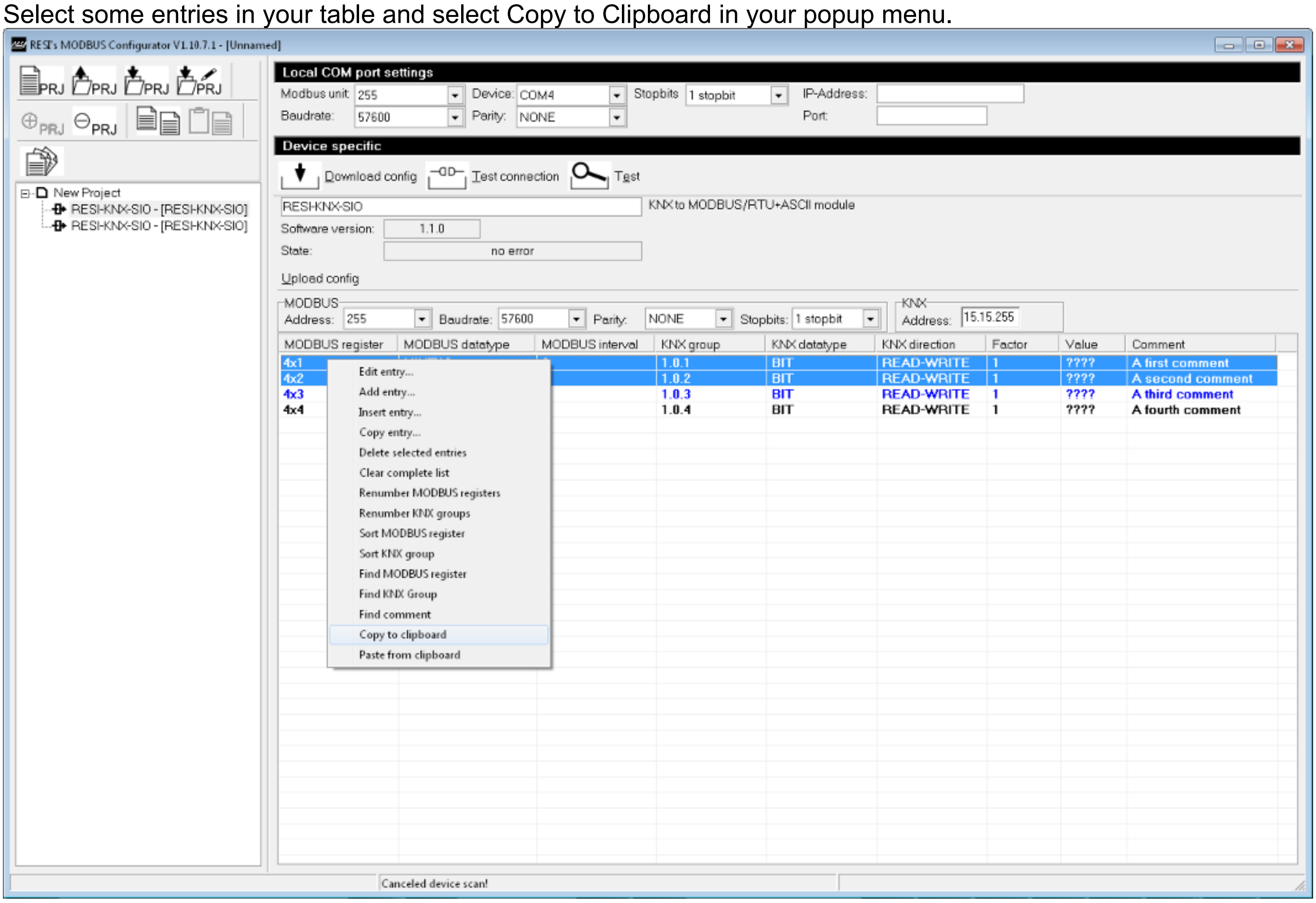


The result looks like:

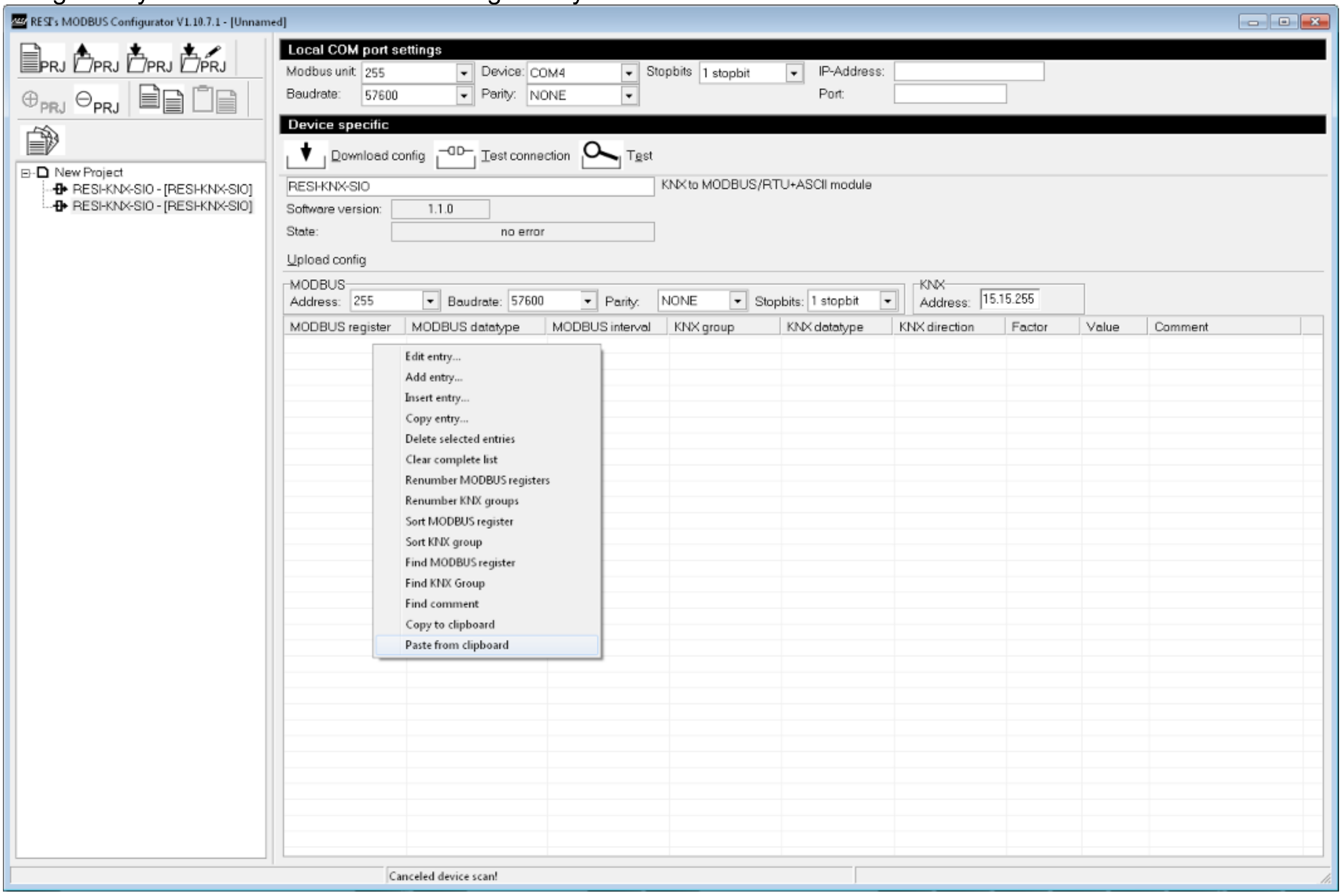
MODBUS register	MODBUS datatype	MODBUS interval	KNX group	KNX datatype	KNX direction	Factor	Value	Comment
4x1	UINT16	0	1.0.1	BIT	READ-WRITE	1	????	A first comment
4x2	UINT16	0	1.0.2	BIT	READ-WRITE	1	????	A second comment
4x3	UINT16	0	1.0.3	BIT	READ-WRITE	1	????	A third comment
4x4	UINT16	0	1.0.4	BIT	READ-WRITE	1	????	A fourth comment

44.6.2.13 Context menu: Copy to/Paste from clipboard

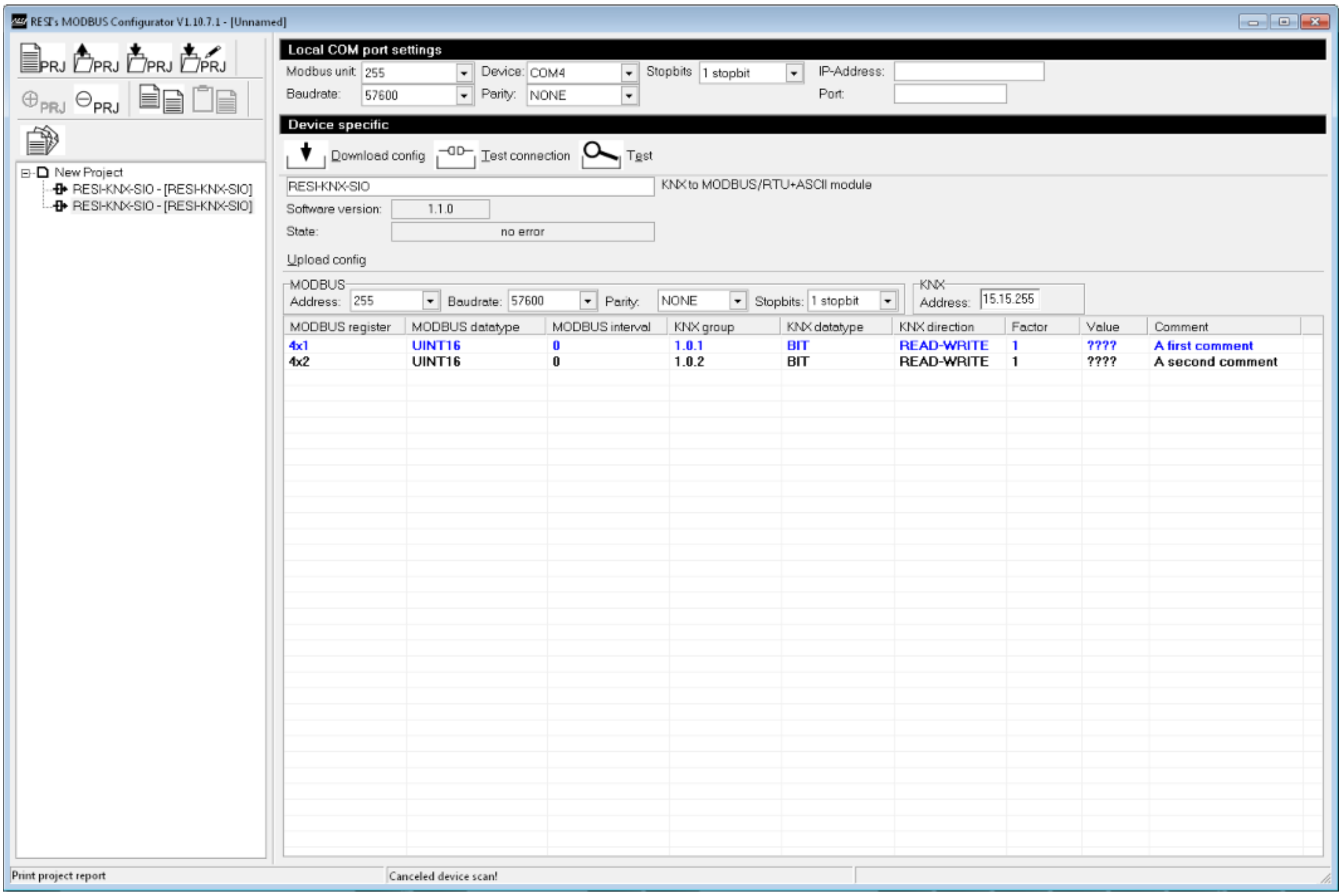
Select some entries in your table and select Copy to Clipboard in your popup menu.



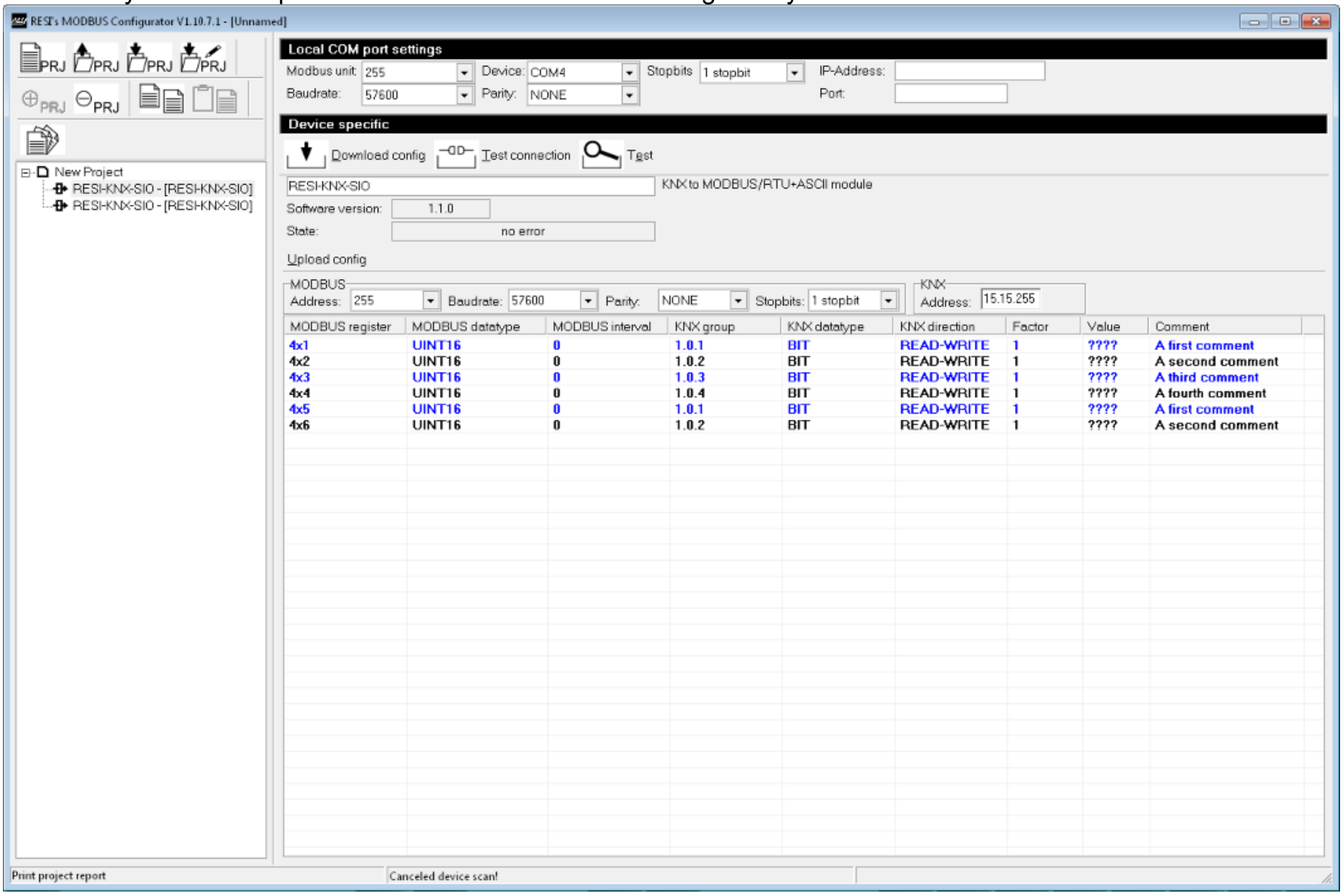
The select another KNX gateway in your project and choose Paste from Clipboard. Now the selected lines from the first gateway are inserted into the second gateway:



The result will look like this:



Of course you can also paste the selected lines into the first gateway:

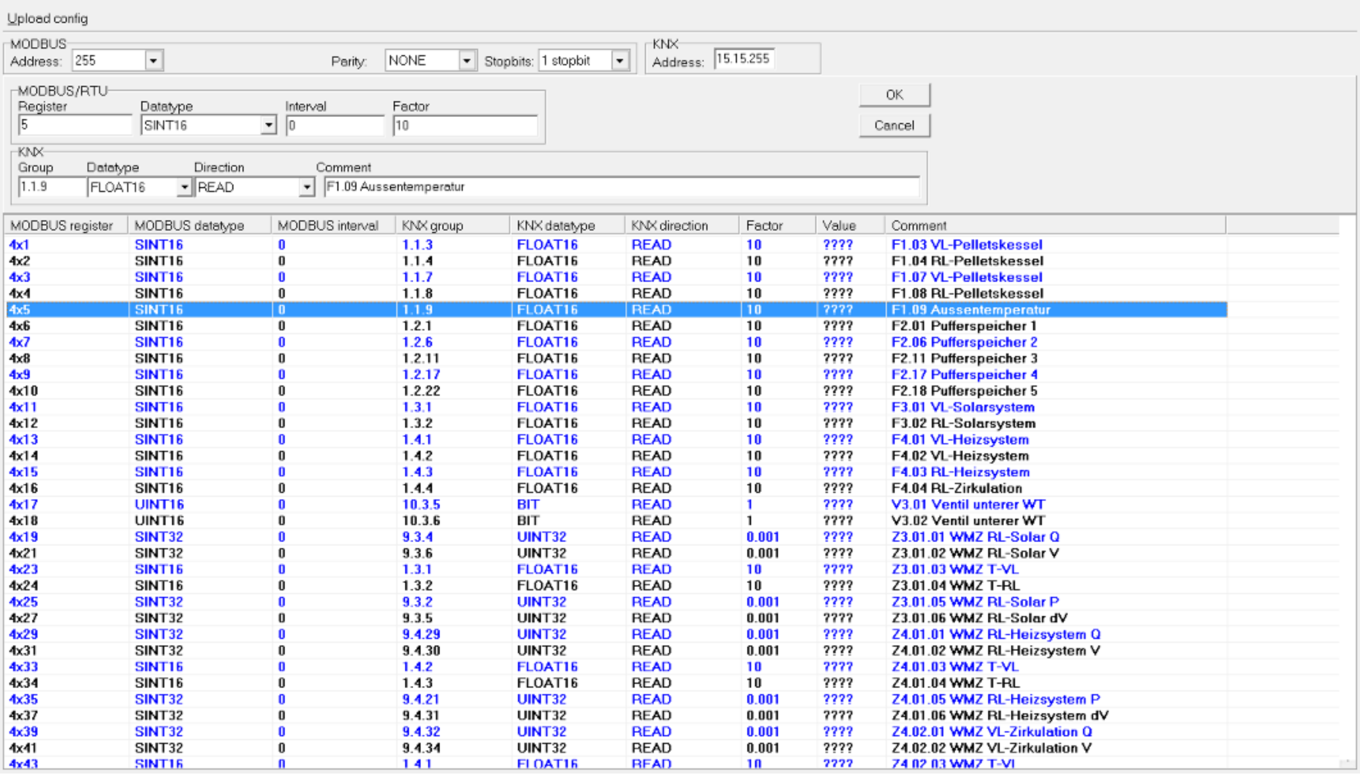


44.6.2.14 Context menu: Edit entry

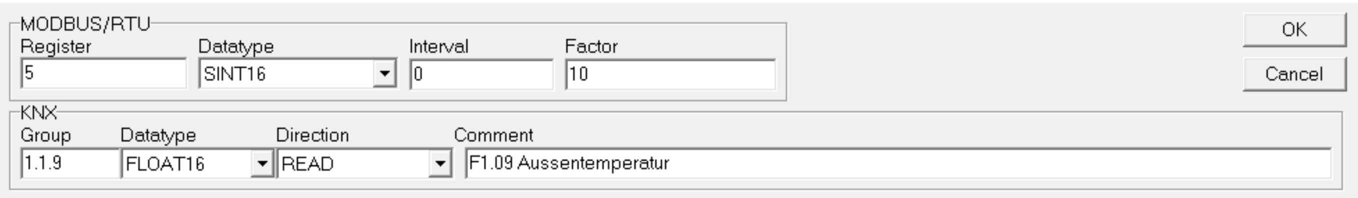
Entry “Edit entry…”: After selecting a line and choosing this function from the local context menu or after a double click onto a line in the table, the below screen will appear. In the upper region an edit area is displayed with the current contents of the selected line.

- Button “Cancel”: Selecting this function will close the edit operation, and no changes will be done in the configuration line. The edit area will disappear.
- Button “OK”: Selecting this function will update the selected configuration line with the altered data and close the edit function. The edit areas will disappear.

HINT: Don’t forget, that you must download the new configuration to the gateway, before the changes are used by the converter!



Here is a zoom into the edit area:



The edit area is divided into two areas:

Area “**MODBUS/RTU**”: Here you will find all edit fields corresponding to the MODBUS/RTU holding register setup.

- Field “**Register**”: Enter a valid MODBUS holding register start index in the range of 1 to 65535. How many MODBUS holding registers are used for this configuration entry is defined by the configured MODBUS/RTU datatype.

- Field “**Datatype**”: Choose one of the possible datatypes from the drop down list. This datatype defines one the one side, how many MODBUS registers are used for the mapping (e.g. datatype UINT16 needs one register, datatype FLOAT32 needs two consecutive MODBUS registers). And on the other hand this data type defines the data representation in this holding registers (e.g. datatype FLOAT32 stores the upper 16 bits of the 32 bit float value in the first holding register and the lower 16 bits are stored in the next consecutive holding register. FLOAT32R stores the two 16 bit words of the 32 bit value in reverse order: The low word in the first register, the high word in the next register).

MODBUS DATATYPE	**SIZE**	**WORD ORDER**	**DESCRIPTION**
ERR	none	none	Defines an invalid configuration entry and is ignored by the gateway
UINT16	16 bits 1 register	none	Defines a 16 bit unsigned integer value in the range of 0 to 65535 or 0x0000 to 0xFFFF
SINT16	16 bits 1 register	none	Defines a 16 bit signed integer value in the range of -32768 to +32767 or 0x8000 to 0x7FFF
UINT32	32 bits 2 register	0:High Word 1:Low Word	Defines a 32 bit unsigned integer value in the range of 0 to 4.294.967.295 or 0x00000000 to 0xFFFFFFFF
SINT32	32 bits 2 register	0:High Word 1:Low Word	Defines a 32 bit signed integer value in the range of −2.147.483.648 to +2.147.483.647or 0x80000000 to 0x7FFFFFFF
UINT32R	32 bits 2 register	0:Low Word 1:High Word	Defines a 32 bit unsigned integer value in the range of 0 to 4.294.967.295 or 0x00000000 to 0xFFFFFFFF with reverse word order
SINT32R	32 bits 2 register	0:Low Word 1:High Word	Defines a 32 bit float value in the range of $\pm 1.4 \cdot 10^{-45}$ to $\pm 3.403 \cdot 10^{38}$. A mantissa of 23 bits and an exponent of 8 bits are used. The value can store 7 to 8 digits after the comma.
FLOAT32	32 bits 2 register	0:High Word 1:Low Word	Defines a 32 bit float value in the range of $\pm 1.4 \cdot 10^{-45}$ to $\pm 3.403 \cdot 10^{38}$. A mantissa of 23 bits and an exponent of 8 bits are used. The value can store 7 to 8 digits after the comma.
FLOAT32R	32 bits 2 register	0:Low Word 1:High Word	Defines a 32 bit float value in the range of $\pm 1.4 \cdot 10^{-45}$ to $\pm 3.403 \cdot 10^{38}$. A mantissa of 23 bits and an exponent of 8 bits are used. The value can store 7 to 8 digits after the comma. The two 16 bit words are stored in reverse order.
DOUBLE64	64 bits 4 register	0:Highest Word 1:Higer Word 2:Lower Word 3:Lowest Word	Defines a 64 bit float value in the range of $\pm 4.24 \cdot 10^{-324}$ to $\pm 1{,}798 \cdot 10^{308}$. A mantissa of 52 bits and an exponent of 11 bits are used. The value can store 15 to 16 digits after the comma.
DOUBLE64R	64 bits 4 register	0:Lowest Word 1:Lower Word 2:Higher Word 3:Highest Word	Defines a 64 bit float value in the range of $\pm 4.24 \cdot 10^{-324}$ to $\pm 1{,}798 \cdot 10^{308}$. A mantissa of 52 bits and an exponent of 11 bits are used. The value can store 15 to 16 digits after the comma. The four 16 bit words are stored in reverse order.
GENERIC	64 bits 4 register	0: FIRST and SECOND byte 1: THIRD and FOURTH byte 2: FIFTH and SIXTH byte 3: SEVENTH and EIGHT byte	Currently unused
STRING	64 bits 4 register	0: FIRST and SECOND byte 1: THIRD and FOURTH byte 2: FIFTH and SIXTH byte 3: SEVENTH and EIGHT byte	Currently unused

- Field “**Interval**”: This field has no function at the moment. The intention is in future releases to define the time in seconds for an automatic polling for the configured KNX group on the KNX bus.

- Field “**Factor**”: This field defines a float value which is used to convert the KNX and MODBUS values after receiving from and before sending to the KNX bus.

  In case of receiving a KNX group the formula is:
  MODBUS value= KNX value multiplied by Factor

  In case of sending a KNX group to the KNX bus, the formula is:
  KNX value=MODBUS value divided by Factor

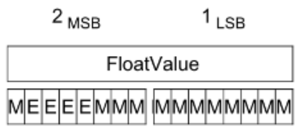
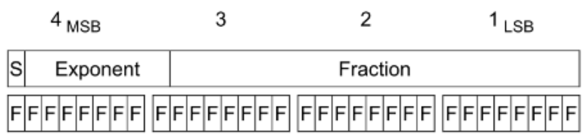
  In case the KNX data type is GENERIC or STRING, this factor defines the start index, from which the data bytesare readout of the KNX telegram. A KNX telegram can hold up to 14 data bytes.

  A Factor of 0 is ignored from the gateway.

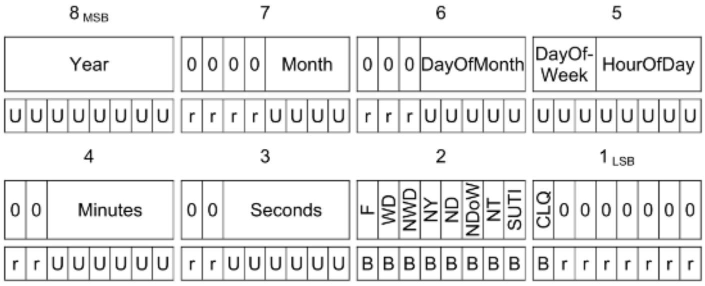
Area “**KNX**”: Here you will find all edit fields corresponding to the KNX group address mapping.

- Field “**Group**”: Here you can define the KNX group address for this configuration entry in the range of 0.0.0 to 31.7.255.

- Field “**Datatype**”: Here you can define the KNX datatype of the incoming or outgoing KNX telegram.

KNX DATATYPE	SIZE	DESCRIPTION
ERR	none	Defines an invalid configuration entry and is ignored by the gateway
BIT	1 bit	Defines a bit value in the range from 0 to 1 or 0x0 to 0x1. Often interpreted as OFF and ON. Defines an integer value consisting out of two bits in the range from 0 to 3 or 0x0 to 0x3.
TWOBITS	2 bits	Defines an integer value consisting out of two bits in the range from 0 to 3 or 0x0 to 0x3.
FOURBITS	4 bits	Defines an integer value consisting out of four bits in the range from 0 to 15 or 0x0 to 0xF.
SIXBITS	6 bits	Defines an integer value consisting out of six bits in the range from 0 to 63 or 0x00 to 0x3F.
CHARACTER	8 bits	Defines one text character consisting out of eight bits in the range from 0 to 255 or 0x00 to 0xFF. Please refer to the KNX documentation, how the encoding of the text character is done by the KNX standard. The encoding can be done for ASCII characters or for ISO 8859.1 characters.
UINT8	8 bits	Defines an 8 bit unsigned integer value in the range of 0 to 255 or 0x00 to 0xFF.
SINT8	8 bits	Defines an 8 bit signed integer value in the range of -128 to +127 or 0x80 to 0x7F.
UINT16	16 bits	Defines a 16 bit unsigned integer value in the range of 0 to 65535 or 0x0000 to 0xFFFF.
SINT16	16 bits	Defines a 16 bit signed integer value in the range of -32768 to +32767 or 0x8000 to 0x7FFF.
UINT32	32 bits	Defines a 32 bit unsigned integer value in the range of 0 to 4.294.967.295 or 0x00000000 to 0xFFFFFFFF.
SINT32	32 bits	Defines a 32 bit signed integer value in the range of −2.147.483.648 to +2.147.483.647or 0x80000000 to 0x7FFFFFFF.
FLOAT16	16 bits	Defines a 16 bit float value with a 4 bit exponent and a 12 bit mantissa. 2 octets: F_{16}  FloatValue = $(0,01*M)*2^{(E)}$ E = [0 … 15] M = [-2 048 … 2 047], two's complement notation For all Datapoint Types 9.xxx, the encoded value 7FFFh shall always be used to denote invalid data. [-671 088,64 … 670 760,96]
FLOAT32	32 bits	Defines a 32 bit float value in the range of $\pm 1.4 \cdot 10^{-45}$ to $\pm 3.403 \cdot 10^{38}$. A mantissa of 23 bits, and an exponent of 8 bits are used. The value can store 7 to 8 digits after the comma. 4 octets: F_{32}  The values are encoded in the IEEE floating point format according IEEE 754. S (Sign) = {0,1} Exponent = [0 … 255] Fraction = [0 … 8 388 607]

<table>
<tr><th>KNX DATATYPE</th><th>SIZE</th><th>DESCRIPTION</th></tr>
<tr><td>TIME</td><td>24 bits</td><td>Defines a 24 bit value encoded a time information in the following way:
3 octets: $N_3U_5r_2U_6r_2U_6$
<table>
<tr><th>3 MSB</th><th>2</th><th>1 LSB</th></tr>
<tr><td>Day | Hour</td><td>0 0 | Minutes</td><td>0 0 | Seconds</td></tr>
<tr><td>N N N U U U U U</td><td>r r U U U U U U</td><td>r r U U U U U U</td></tr>
</table>
binary encoded
<table>
<tr><th>Field:</th><th>Encoding:</th><th>Range:</th><th>Unit:</th><th>Resol.:</th></tr>
<tr><td>Day</td><td>1 = Monday
…
7 = Sunday
0 = no day</td><td>[0…7]</td><td>none</td><td>none</td></tr>
<tr><td>Hour</td><td>binary encoded</td><td>[0…23]</td><td>hours</td><td>h</td></tr>
<tr><td>Minutes</td><td>binary encoded</td><td>[0…59]</td><td>minutes</td><td>min</td></tr>
<tr><td>Seconds</td><td>binary encoded</td><td>[0…59]</td><td>seconds</td><td>s</td></tr>
</table></td></tr>
<tr><td>DATE</td><td>24 bits</td><td>Defines a 24 bit value encoded a date information in the following way:
3 octets: $r_3U_5r_4U_4r_1U_7$
<table>
<tr><th>3 MSB</th><th>2</th><th>1 LSB</th></tr>
<tr><td>0 0 0 | Day</td><td>0 0 0 0 | Month</td><td>0 | Year</td></tr>
<tr><td>r r r U U U U U</td><td>r r r r U U U U</td><td>r U U U U U U U</td></tr>
</table>
All values binary encoded.
<table>
<tr><th>Field:</th><th>Range:</th><th>Unit:</th><th>Resol.:</th></tr>
<tr><td>Day</td><td>[1…31]</td><td>Day of month</td><td>1 day</td></tr>
<tr><td>Month</td><td>[1…12]</td><td>Month</td><td>1 month</td></tr>
<tr><td>Year</td><td>[0…99]</td><td>Year</td><td>1 year</td></tr>
</table></td></tr>
</table>

<table>
<tr><th>KNX DATATYPE</th><th>SIZE</th><th>DESCRIPTION</th></tr>
<tr><td>DATETIME</td><td>64 bits</td><td>
Defines a 64 bit value encoded a date and time information in the following way:
8 octets: $U_8[r_4U_4][r_3U_5][U_3U_5][r_2U_6][r_2U_6]B_{16}$


<table>
<tr><th>Field</th><th>Description</th><th>Encoding</th><th>Range</th><th>Unit</th><th>Resol.:</th></tr>
<tr><td>Year</td><td>Year</td><td>Value binary encoded, offset 1900
0 = 1900
255 = 2155</td><td>[0…255]</td><td>year</td><td>1 year</td></tr>
<tr><td>Month</td><td>Month</td><td>Value binary encoded
1 = January
…
12 = December</td><td>[1…12]</td><td>Month</td><td>1 month</td></tr>
<tr><td>DayOfMonth</td><td>D</td><td>Value binary encoded
1 = 1st day
31 = 31st day</td><td>[1…31]</td><td>none</td><td>none</td></tr>
<tr><td>DayOfWeek</td><td>Day of week</td><td>Value binary encoded
0 = any day
1 = Monday
…
7 = Sunday</td><td>[0…7]</td><td>none</td><td>none</td></tr>
<tr><td>HourOfDay</td><td>Hour of day</td><td>Value binary encoded.</td><td>[0…24]</td><td>h</td><td>1 h</td></tr>
<tr><td>Minutes</td><td>Minutes</td><td>Value binary encoded.</td><td>[0…59]</td><td>min</td><td>1 min</td></tr>
<tr><td>Seconds</td><td>Seconds</td><td>Value binary encoded.</td><td>[0…59]</td><td>s</td><td>1 s</td></tr>
<tr><td>F</td><td>Fault</td><td>0 = Normal (No fault)
1 = Fault</td><td>{0,1}</td><td>none</td><td>none</td></tr>
<tr><td>WD</td><td>Working Day</td><td>0 = Bank day (No working day)
1 = Working day</td><td>{0,1}</td><td>none</td><td>none</td></tr>
<tr><td>NWD</td><td>No WD</td><td>0 = WD field valid
1 = WD field not valid</td><td>{0,1}</td><td>none</td><td>none</td></tr>
<tr><td>NY</td><td>No Year</td><td>0 = Year field valid
1 = Year field not valid</td><td>{0,1}</td><td>none</td><td>none</td></tr>
<tr><td>ND</td><td>No Date</td><td>0 = Month and Day of Month fields valid
1 = Month and Day of Month fields not valid</td><td>{0,1}</td><td>none</td><td>none</td></tr>
</table>
</td></tr>
<tr><td>STRING</td><td>max. 14 bytes</td><td>
Defines up to 14 bytes of text data.

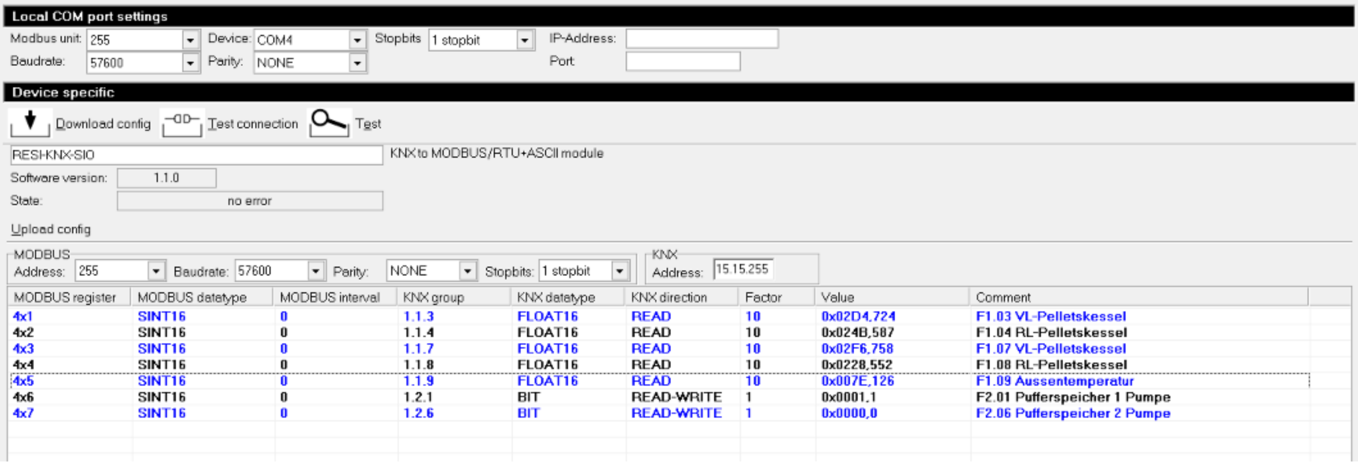

These Datapoint Types are used to transmit strings of textual characters. The length is fixed to 14 octets. The contents are filled starting from the most significant octet. Each octet shall be encoded as specified for the chosen character set, as defined in clause 0. If the string to be transmitted is smaller then 14 octets, unused trailing octets in the character string shall be set to NULL (00h).
<u>Example</u>: 'KNX is OK' is encoded as follows :
4B 4E 58 20 69 73 20 4F 4B 00 00 00 00 00
Defines a 64 bit value which represents up to 8 bytes from the data section of a generic KNX telegram. Due to the fact, that a generic KNX frame can hold up to 14 bytes, the field factor defines the start index for the 8 bytes in the range from 0 to 13. The system stores the first byte in the first 16 bit MODBUS register in the low 8 bits. The next byte is stored in the same register, but in the upper half of the word. The third byte is stored in the next Modbus register in the low half, and so on.
</td></tr>
<tr><td>GENERIC</td><td>max. 14 bytes</td><td>Defines a 64 bit value which represents up to 8 bytes from the data section of a generic KNX telegram. Due to the fact, that a generic KNX frame can hold up to 14 bytes, the field factor defines the start index for the 8 bytes in the range from 0 to 13. The system stores the first byte in the first 16 bit MODBUS register in the low 8 bits. The next byte is stored in the same register, but in the upper half of the word. The third byte is stored in the next Modbus register in the low half, and so on.</td></tr>
</table>

- Field “**Direction**”: Select the communication direction of the KNX group address on the KNX bus. Choose READ for only incoming KNX messages, WRITE for only outgoing KNX messages and READ_WRITE for incoming and outgoing messages. ERR defines an invalid configuration data and is ignored by the gateway.

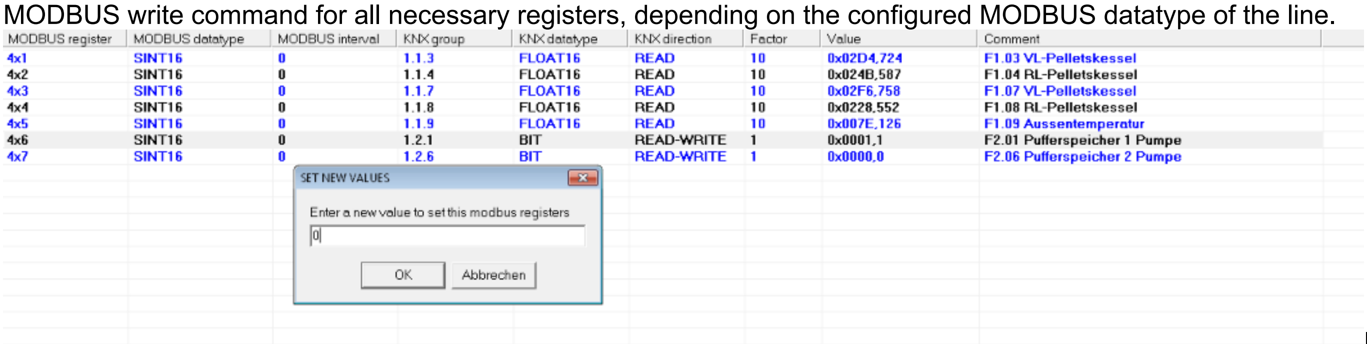
- Field “**Comment**”: Enter a comment to explain your KNX MODBUS mapping for documentation purpose. Note that the comment in only stored onto the PC, not in the gateway. So if you upload a configuration from the gateway, you will lose all comments.

44.7 Testing the configuration

After you download your new configuration into the converter and start the test mode with the button “Test”, you will see the following screen. The system automatically updates all MODBUS registers every 5 seconds.



To set a new value for a configuration line simple double click onto a configuration line. A window will open, in which you can enter the new value for the selected configuration line. The software will automatically create the correct MODBUS write command for all necessary registers, depending on the configured MODBUS datatype of the line.



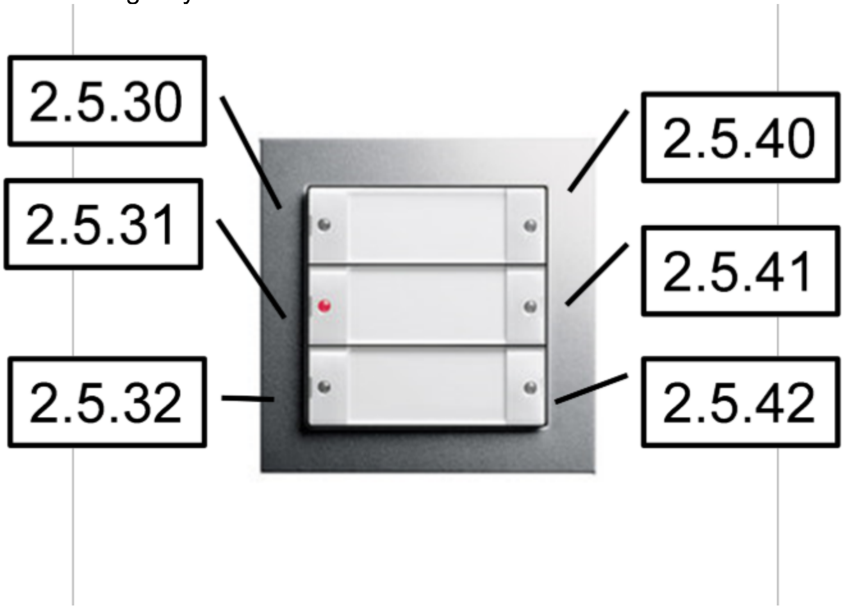
Immediately after receiving all MODBUS registers, the converter will send out the corresponding KNX telegram onto the KNX bus, if KNX write is allowed.

44.8 Sample configurations

Here you will find some sample configurations to explain the configuration principles of our gateway.

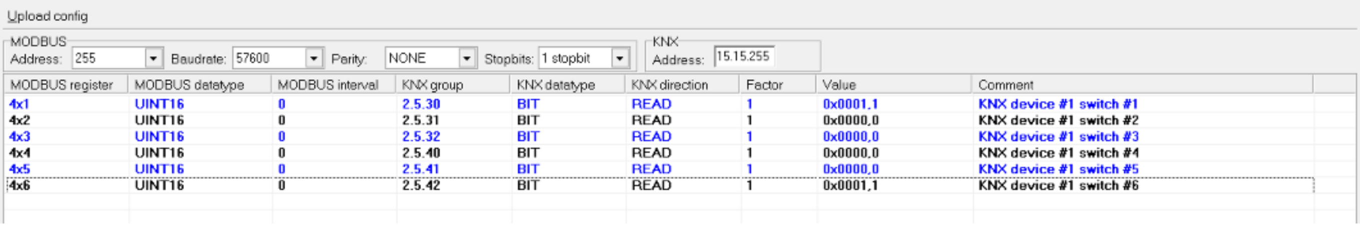
44.8.1 Reading the status of a KNX switch

Assuming the following setup: 1 KNX switch e.g. GIRA with six switches, programmed with KNX group addresses in the following way:



All six KNX groups send as KNX data a BIT value defining the current state of the switch (0=OFF, 1=ON). If you press the left, top switch, the KNX device sends the KNX telegram 2.5.30=1 or 2.5.30=0 depending on the stored switch state in the KNX device.

So the configuration will look like this:



Upload config

MODBUS Address: 255 Baudrate: 57600 Parity: NONE Stopbits: 1 stopbit

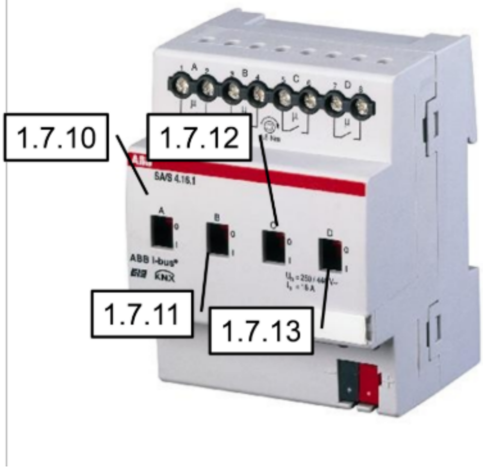
KNX Address: 15.15.255

MODBUS register	MODBUS datatype	MODBUS interval	KNX group	KNX datatype	KNX direction	Factor	Value	Comment
4x1	UINT16	0	2.5.30	BIT	READ	1	0x0001,1	KNX device #1 switch #1
4x2	UINT16	0	2.5.31	BIT	READ	1	0x0000,0	KNX device #1 switch #2
4x3	UINT16	0	2.5.32	BIT	READ	1	0x0000,0	KNX device #1 switch #3
4x4	UINT16	0	2.5.40	BIT	READ	1	0x0000,0	KNX device #1 switch #4
4x5	UINT16	0	2.5.41	BIT	READ	1	0x0000,0	KNX device #1 switch #5
4x6	UINT16	0	2.5.42	BIT	READ	1	0x0001,1	KNX device #1 switch #6

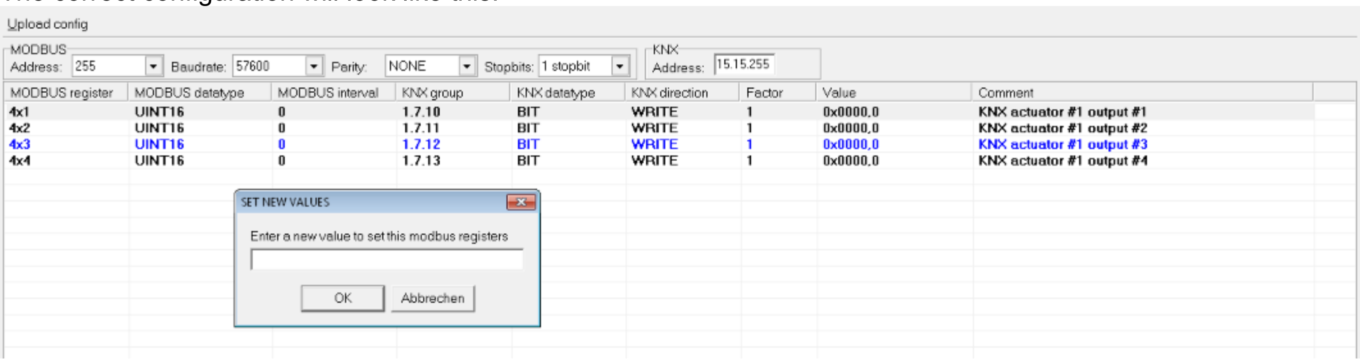
As you can see in the test mode, the switch state of Switch #1 and #6 is ON, all other switches are OFF. Press the six buttons and see, how the MODBUS registers are changed by the incoming KNX telegrams.

44.8.2 Writing to a KNX actuator

Using a KNX actuator with four outputs, e.g. an ABB KNX actuator, is also very simple. Assuming the following KNX group addresses for the four outputs. All of them expect a KNX telegram with bit data.



The correct configuration will look like this:



Upload config

MODBUS Address: 255 Baudrate: 57600 Parity: NONE Stopbits: 1 stopbit

KNX Address: 15.15.255

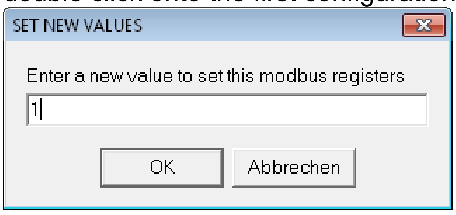
MODBUS register	MODBUS datatype	MODBUS interval	KNX group	KNX datatype	KNX direction	Factor	Value	Comment
4x1	UINT16	0	1.7.10	BIT	WRITE	1	0x0000,0	KNX actuator #1 output #1
4x2	UINT16	0	1.7.11	BIT	WRITE	1	0x0000,0	KNX actuator #1 output #2
4x3	UINT16	0	1.7.12	BIT	WRITE	1	0x0000,0	KNX actuator #1 output #3
4x4	UINT16	0	1.7.13	BIT	WRITE	1	0x0000,0	KNX actuator #1 output #4

SET NEW VALUES

Enter a new value to set this modbus registers

OK Abbrechen

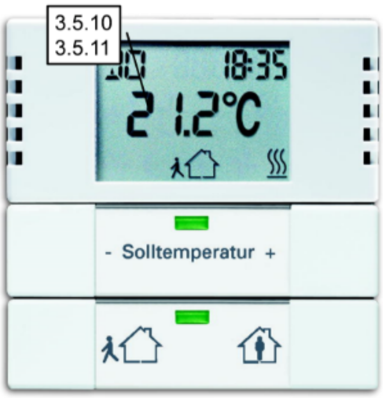
Download this configuration into the gateway and start the test mode. To change the state of the output #1, simple double click onto the first configuration line. The following window will be opened:



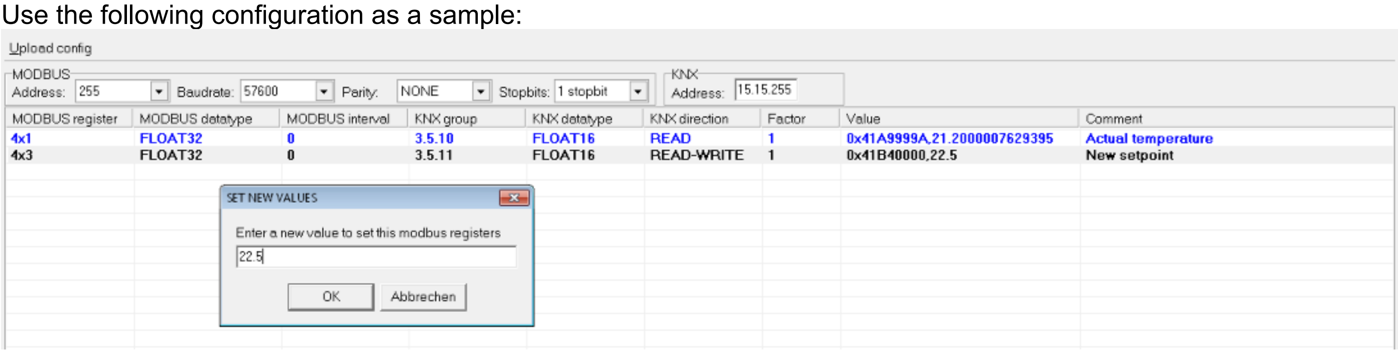
Enter a new value for the digital output, e.g. 1, and hit the OK button. Immediately the actuator on the KNX bus will be switched on. Double click again and enter the value 0, the output will be switched off. Try this for the remaining three outputs.

44.8.3 Reading analogue KNX values

More complicated is the mapping of analogue values from the KNX bus to MODBUS registers. We start with a simple KNX device, e.g. a room controller. We assume, that this device cyclically sends the current room temperature on the KNX group address 3.5.10. The set point can be send/received on the KNX group address 3.5.11. Both values are encoded with KNX datatype 9.001 temperature (°C).

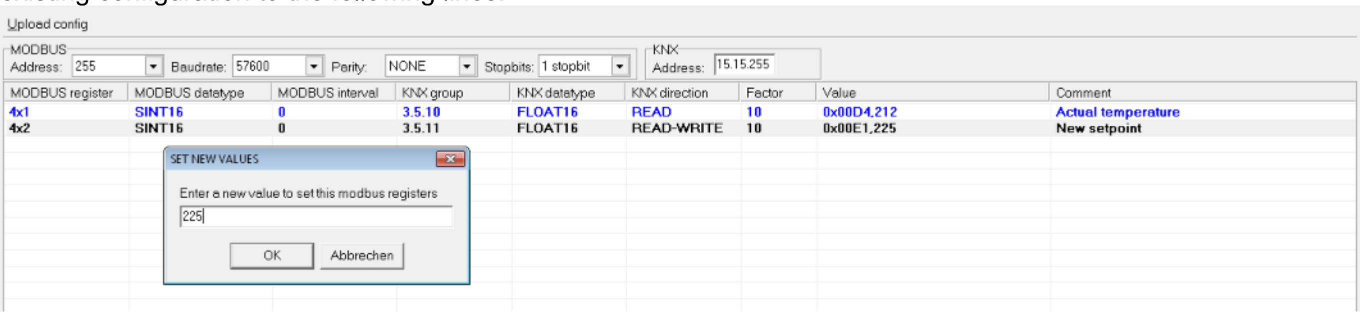


Use the following configuration as a sample:



As you can see, we map the float value FLOAT16 (This is the KNX representation of KNX Datatype 9.0001) from the KNX bus to FLOAT32 values in the MODBUS registers. A FLOAT32 value uses 2 consecutive registers. That's the reason, why the first value uses the index 4x1 and the second index uses the index 4x3. So this configurations uses four MODBUS registers with the indices 4x1, 4x2, 4x3 and 4x4.

But you can also map a float value from the KNX bus to an integer value on the MODBUS side. We change the existing configuration to the following lines:



As you will notice, we use now UINT16 for the MODBUS datatype and a factor of 10 to preserve the first digit after the comma. So the MODBUS register 4x1 stores the value 212 if the current temperature 22.2 °C is received from the KNX bus.

44.9 Additional MODBUS register & coils

Here you will find only the additional MODBUS registers and coils especially for this IO module. Please refer to the description of of the standard MODBUS mapping for more details about the available basic MODBUS registers and coils.

Please refer to the external document for detailed documentation of the current MODBUS register mapping for this IO module:

RESI-L-KNX-SIO-ETH-MODBUS+ASCII-ENxx.pdf

Don't forget, that there are some standard MODBUS registers for this device, which you cannot overwrite with your configuration!

44.10 Additional ASCII commands

Here you will find only the additional ASCII commands especially for this IO module. Please refer to the description of of the standard commands for more details about the available basic ASCII commands.

Please refer to the external document for detailed documentation of the current ASCII commands for this IO module:

RESI-L-KNX-SIO-ETH-MODBUS+ASCII-ENxx.pdf

44.10.1 The configuration line

The two ASCII commands READ CONFIG and GET CONFIG return all data for one configuration line. Here is a detailed description of each field of this line.

The common syntax for the line is the following:
**#<BusAdr>,KNX:I:<Index>=<MBRegister>,<MBDataType>,<MBRegisterCount>,<Interval>,
<KNXGroup>,<KNXDataType>,<KNXDirection>,<Factor>**

<BusAdr> stands for the current MODBUS bus address of the module as a decimal number e.g. 255

<Index> stands for the Index in the configuration table starting with 1 and ending with n according to the actual size of the configuration table size as a decimal number.

<MBRegister> stands for the starting index of the holding registers, starting with 1 for the first holding register 4x00001 and ending by 65535 for the last holding register 4x65535 as a decimal number.

<MBDataType> defines a datatype for the MODBUS registers. This is an ASCII text string in capital letters. Choose one of the following strings: UINT16, SINT16, UINT32, SINT32, UINT32R, SINT32R, FLOAT32, FLOAT32R, DOUBLE64, DOUBLE64R, GENERIC, ASCII or ERR. See the explanation of the MODBUS datatypes for more details about this strings.

<MBRegisterCount> defines the amount of MODBUS holding registers, which are used by this configuration entry as a decimal number. For example a UINT16 needs 1 register, a UINT32 or a FLOAT32 needs 2 registers.

<Interval> stands for a time interval in seconds as a decimal number for an automatic polling request on the KNX bus for this KNX group address. This is for future use and not used now!

<KNXGroup> defines the KNX group address with the format <Hi>.<Medium>.<Low>. KNX group addresses in the range from 0.0.0 to 31.7.255 are valid here.

<KNXDataType> is a string defining the data type of the incoming or outgoing KNX data. The system uses the following ASCII strings in capital letters: ERR, BIT, TWOBITS, FOURBITS, SIXBITS, CHARACTER, UINT8, SINT8, UNIT16, SINT16, FLOAT16, TIME, DATE, UINT32, SINT32, FLOAT32, STRING, GENERIC and DATETIME. See the explanation of the KNX datatypes for more details about this strings.

<KNXDirection> is an ASCII text string defining the communication direction for this entry on the KNX bus. The following ASCII string in capital letters are valid: ERR, R, W, RW. See the explanation for KNX directions for more details about this string.

<Factor> is a float value defining the multiplication factor for incoming KNX telegrams and the division factor for outgoing KNX telegrams. Use the float format 1234.567. Don’t use a comma as a comma sign!

Here is a cut-out of a real configuration from a terminal program:

```
1    5    10   15   20   25   30   35   40   45   50   55
#VERSION:1.0.1
TYPE:RESI-KNX-MODBUS
#KNX:I:1=1,SINT16,1,0,1.1.3,FLOAT16,R,10.000000
#KNX:I:2=2,SINT16,1,0,1.1.4,FLOAT16,R,10.000000
#KNX:I:3=3,SINT16,1,0,1.1.7,FLOAT16,R,10.000000
#KNX:I:4=4,SINT16,1,0,1.1.8,FLOAT16,R,10.000000
#KNX:I:5=5,SINT16,1,0,1.1.9,FLOAT16,R,10.000000
#KNX:I:6=6,SINT16,1,0,1.2.1,FLOAT16,R,10.000000
#KNX:I:7=7,SINT16,1,0,1.2.6,FLOAT16,R,10.000000
#KNX:I:8=8,SINT16,1,0,1.2.11,FLOAT16,R,10.000000
#KNX:I:9=9,SINT16,1,0,1.2.17,FLOAT16,R,10.000000
#KNX:I:10=10,SINT16,1,0,1.2.22,FLOAT16,R,10.000000
#KNX:I:11=11,SINT16,1,0,1.3.1,FLOAT16,R,10.000000
#KNX:I:12=12,SINT16,1,0,1.3.2,FLOAT16,R,10.000000
#KNX:I:13=13,SINT16,1,0,1.4.1,FLOAT16,R,10.000000
#KNX:I:14=14,SINT16,1,0,1.4.2,FLOAT16,R,10.000000
#KNX:I:15=15,SINT16,1,0,1.4.3,FLOAT16,R,10.000000
#KNX:I:16=16,SINT16,1,0,1.4.4,FLOAT16,R,10.000000
#KNX:I:17=17,UINT16,1,0,10.3.5,BIT,R,1.000000
#KNX:I:18=18,UINT16,1,0,10.3.6,BIT,R,1.000000
#KNX:I:19=19,SINT32,2,0,9.3.4,UINT32,R,0.001000
#KNX:I:20=21,SINT32,2,0,9.3.6,UINT32,R,0.001000
#KNX:I:21=23,SINT16,1,0,1.3.1,FLOAT16,R,10.000000
#KNX:I:22=24,SINT16,1,0,1.3.2,FLOAT16,R,10.000000
#KNX:I:23=25,SINT32,2,0,9.3.2,UINT32,R,0.001000
#KNX:I:24=27,SINT32,2,0,9.3.5,UINT32,R,0.001000
#KNX:I:25=29,SINT32,2,0,9.4.29,UINT32,R,0.001000
#KNX:I:26=31,SINT32,2,0,9.4.30,UINT32,R,0.001000
#KNX:I:27=33,SINT16,1,0,1.4.2,FLOAT16,R,10.000000
#KNX:I:28=34,SINT16,1,0,1.4.3,FLOAT16,R,10.000000
#KNX:I:29=35,SINT32,2,0,9.4.21,UINT32,R,0.001000
#KNX:I:30=37,SINT32,2,0,9.4.31,UINT32,R,0.001000
#KNX:I:31=39,SINT32,2,0,9.4.32,UINT32,R,0.001000
#KNX:I:32=41,SINT32,2,0,9.4.34,UINT32,R,0.001000
#KNX:I:33=43,SINT16,1,0,1.4.1,FLOAT16,R,10.000000
#KNX:I:34=44,SINT16,1,0,1.4.4,FLOAT16,R,10.000000
#KNX:I:35=45,SINT32,2,0,9.4.27,UINT32,R,0.001000
#KNX:I:36=47,SINT32,2,0,9.4.33,UINT32,R,0.001000
#KNX:I:37=49,SINT16,1,0,1.3.4,FLOAT16,R,10.000000
#KNX:I:38=50,SINT16,1,0,1.3.5,FLOAT16,R,10.000000
#KNX:I:39=51,SINT32,2,0,9.4.35,UINT32,R,0.001000

Selection (-)
```

44.10.2 The Add Configuration Line

The ASCII commands ADD CONFIG uses a complex configuration line to add a new entry to the current configuration table. Here is a detailed description of each field of this line.

The common syntax for the line is the following:
#<BusAdr>,ADD CONFIG:<MBRegister>,<MBDataType>,<Interval>, <KNXGroup>,<KNXDataType>,<KNXDirection>,<Factor>

<BusAdr> stand for the current MODBUS bus address of the module as a decimal number e.g. 255

<MBRegister> stands for the starting index of the holding registers, starting with 1 for the first holding register 4x00001 and ending by 65535 for the last holding register 4x65535 as a decimal number. If you use 0 as a MODBUS register index, the next free MODBUS register is used for this entry.

<MBDataType> defines a datatype for the MODBUS registers. This is an ASCII text string in capital letters. Choose one of the following strings: UINT16, SINT16, UINT32, SINT32, UINT32R, SINT32R, FLOAT32, FLOAT32R, DOUBLE64, DOUBLE64R, GENERIC, ASCII or ERR. See the explanation of the MODBUS datatypes for more details about this strings.

<Interval> stands for a time interval in seconds as a decimal number for an automatic polling request on the KNX bus for this KNX group address. This is for future use and not used now!

<KNXGroup> defines the KNX group address with the format <Hi>.<Medium>.<Low>. KNX group addresses in the range from 0.0.0 to 31.7.255 are valid here.

<KNXDataType> is a string defining the data type of the incoming or outgoing KNX data. You can use the following ASCII string in capital letters: ERR, BIT, TWOBITS, FOURBITS, SIXBITS, CHARACTER, UINT8, SINT8, UNIT16, SINT16, FLOAT16, TIME, DATE, UINT32, SINT32, FLOAT32, STRING, GENERIC and DATETIME. See the explanation of the KNX datatypes for more details about this strings.

<KNXDirection> is a string defining the communication direction for this entry on the KNX bus. The following ASCII string in capital letters are valid: ERR, READ, WRITE, READ-WRITE, READWRITE, R, W, RW. See the explanation for KNX directions for more details about this string.

<Factor> is a float value defining the multiplication factor for incoming KNX telegrams and the division factor for outgoing KNX telegrams. Use the float format 1234.567. Don’t use a comma as a comma sign!

A simple example for a valid ADD CONFIG command:

#AC:1,UINT16,0,1.0.0,BIT,READ,1.0
#255,AC:0,UINT16,0,1.0.1,FLOAT16,RW,1.0

45 RESI-DALI-SIO, RESI-DALI-ETH, RESI-DALI-PS

45.1 General information

With the RESI-DALI-SIO gateway, a DALI 1.0 and DALI 2.0 light system can be integrated in almost every system with a RS232 or RS485 interface and a MODBUS/RTU master protocol or serial ASCII text based protocol. The RESI-DALI-ETH gateway offers an integration with MODBUS/TCP server protocol or ASCII text socket over Ethernet.

The gateway is a serial interface for connection to the DALI with an integrated 2 wire DALI bus coupler. The time-critical DALI communication is done from the gateway itself. The gateway is configured with our MODBUSConfigurator software and maps the incoming and outgoing DALI telegrams to MODBUS holding registers. When the host writes to a MODBUS holding register, the gateway generates the corresponding DALI telegram. When the gateway receives a DALI telegram, it maps and converts the incoming data to the specific MODBUS holding registers for readout through a host.

To control our DALI converters you need a host system with a serial interface (RS232 or RS485), which is able to send ASCII command strings and which can receive ASCII characters. This feature is implemented in almost any media control system like CRESTRON®, AMX® or CONTROL4®. But almost every standard PLC can handle serial ASCII interfaces. Therefore our converter can be integrated everywhere. If the host system offers a MODBUS/RTU master or MODBUS/TCP client interface, our converter can be controlled via MODBUS holding registers.

Our DALI gateways support the easy installation and testing of DALI lamps in a DALI 1.0 or DALI 2.0 network. Therefore our software supports the automatic search and addressing of DALI lamps, physical selection and addressing of DALI lamps. Configuration of DALI scenes and DALI groups. Especially the gateway can handle all the new DALI device type 8 for RGB or RGBW dimmable DALI lamps. But also DALI device type 6 is supported.

Furthermore the device can send and receive to following DALI frame formats:

* 8 Bit DSI commands

* 16 Bit DALI 1.0/DALI 2.0 commands

* 24 Bit DALI Multi Master commands or events

* 25 Bit eDALI frames

* 28 Bit DALI Frame for future use

* 32 Bit DALI Frame for future use

* Variable bit length frames from 1 to 64 bits for special purposes

This series of IO modules offer the following features:

- Easy integration of a complete DALI 1.0 and DALI 2.0 bus system
- MODBUS/RTU slave or MODBUS/TCP server protocol
- Additional commands with plain ASCII texts
- DALI and host interface are galvanically isolated
- Supports all DALI 1.0 commands
- Supports all DALI 2.0 commands
- Supports DALI priority slot sending
- Supports new DALI device type 6 and device type 8
- Integrated DALI bus-coupler
- RESI-xxx-SIO: Galvanic isolated RS232 and RS485 interface for communication with a host system
- RESI-xxx-ETH: Galvanic isolated Ethernet interface for communication with a host system

Figure: Our serial DALI module

Figure: Our Ethernet IO module

With our RESI-DALI-PS we deliver a powerful and stable DALI power supply with max. 200mA output current, but supplied from a primary 12 to 48Vdc power supply. Its extreme slim housing suits in every switchboard cabinet, where you usually have 24Vdc for other automation devices.

Figure: Our DALI power supply module

45.2 Technical specification

Beside the basic technical data, which fulfill all of our IO modules, this IO modules meet the following technical specifications:

Power consumption	
RESI-DALI-SIO	<1.0W
RESI-DALI-ETH	<1.4W
RESI-DALI-PS	<6.0W
Product housing	
RESI-DALI-SIO	CEM17
RESI-DALI-ETH	CEM35
RESI-DALI-PS	CEM17
Product weight	
RESI-DALI-SIO	50g
RESI-DALI-ETH	83g
RESI-DALI-PS	65g
DALI bus interface	
RESI-DALI-SIO	
RESI-DALI-ETH	
Protocol	DALI 1.0 and DALI 2.0 and DALI multi master
Baud rate	1200Bit/s
Cable connection	via terminals
Galvanic isolation	Yes
LED indicator	Yes
RESI-DALI-PS	
Nominal output voltage	Maximum 18V, typical ~14-~18V
Maximum output current	~200mA
Short circuit output current	~225mA
Galvanic isolation	Yes
LED indicator	Yes
Default serial settings	
Baud rate	via DIP switch
Parity	none
Stop bits	one
UnitID	255
Default Ethernet settings	
IP address	192.168.0.191
IP mask	255.255.255.0
gateway	192.168.0.1
UnitID	255
User	RESI
password	RESI

45.3 Additional terminals & LED states

DALI INTERFACE	DALI bus connector	
	One 3 pin terminal blocks	
	Terminal type:	USLIM
	D+:	DALI+ bus wire
	D-:	DALI- bus wire
Pin layout	D+:	DALI+ bus wire
	N/C:	not connected
	D-:	DALI- bus wire
RESI-DALI-SIO		
RESI-DALI-ETH		
STATE	If the DALI bus power is OK and the module has no error, this LED flashes with	
	a 1s rhythm. If the module has an internal error or the DALI bus power is not connected	
	or there is a short circuit on the DALI bus line this LED flashes very quick	
	(~250ms rhythm)	
DALI	If there is bus communication on the DALI, this LED is on, otherwise this LED is OFF	
RESI-DALI-PS		
POWER	Always on to indicate primary power supply is OK.	
DALI	DALI activity LED, when transmitting a DALI telegram the LED switches	
	on for a few milliseconds.	
ERR	Power supply fault LED. If the power supply unit malfunctions or if there is a bus error	
	on the DALI line or if there is a short circuit in the DALI system, this LED lights up.	
	If the DALI bus is working properly, this LED is off.	

45.4 RESI-DALI-SIO: Connection diagram

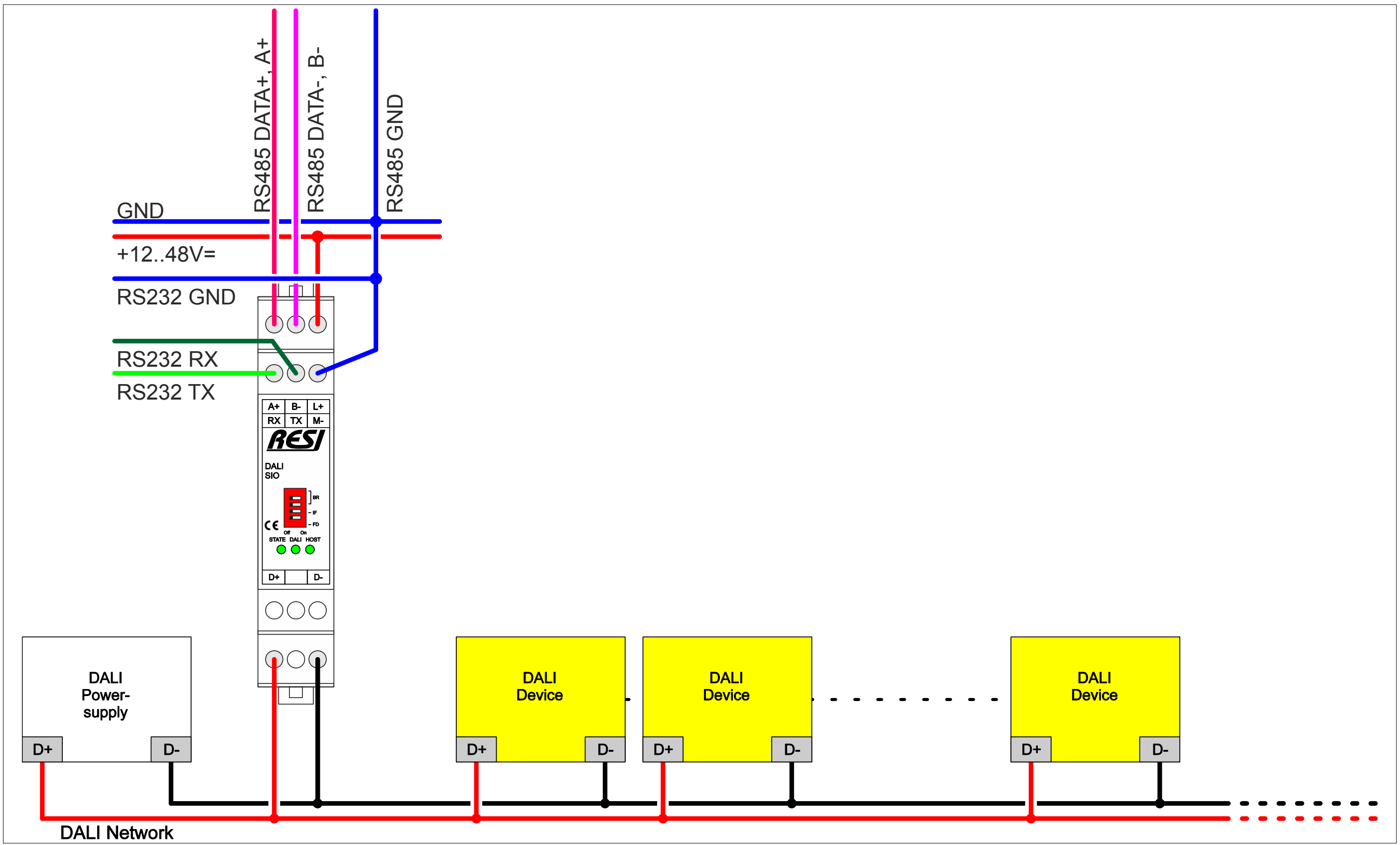


Figure: Connecting a DALI bus system to the RESI-DALI-SIO gateway with external DALI power supply

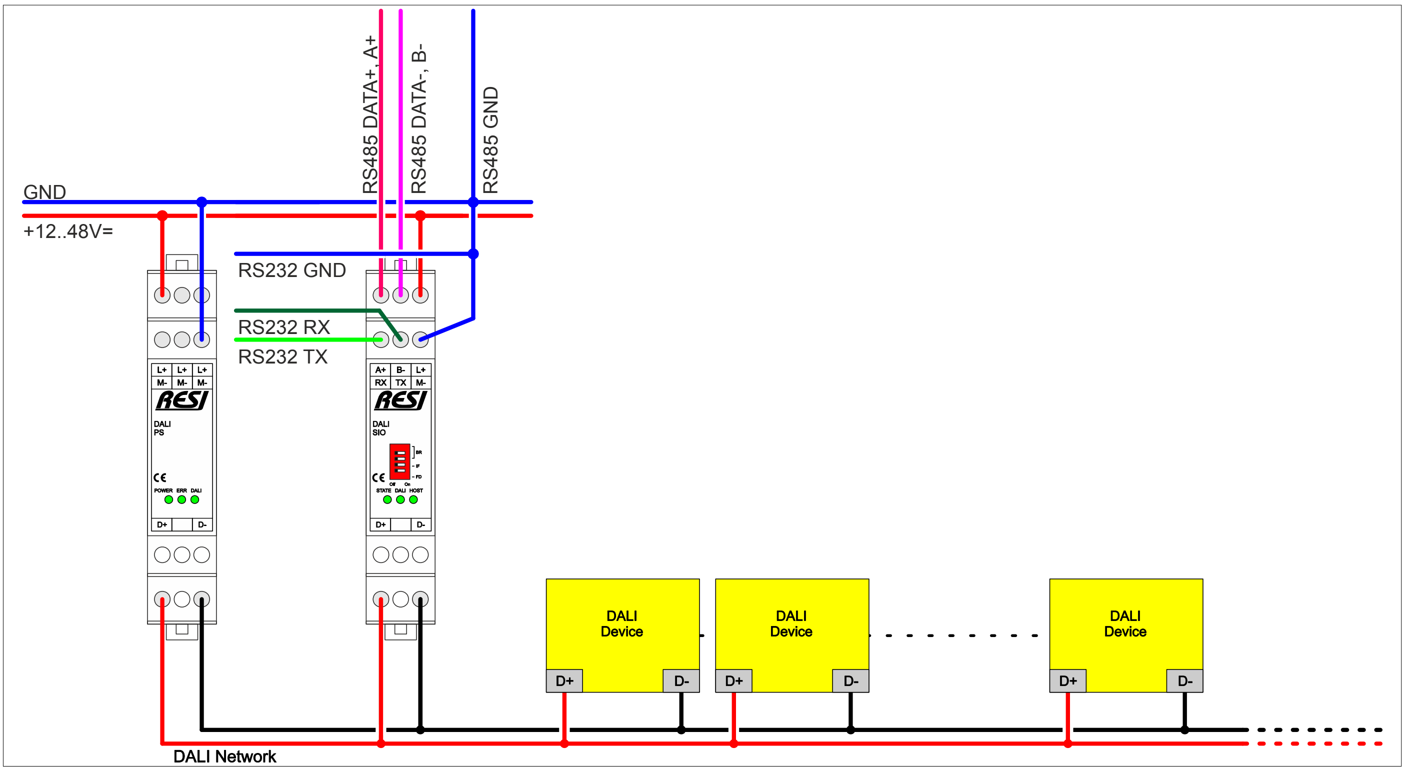


Figure: Connecting a DALI bus system to the RESI-DALI-SIO gateway with RESI-DALI-PS DALI power supply

45.5 RESI-DALI-ETH: Connection diagram

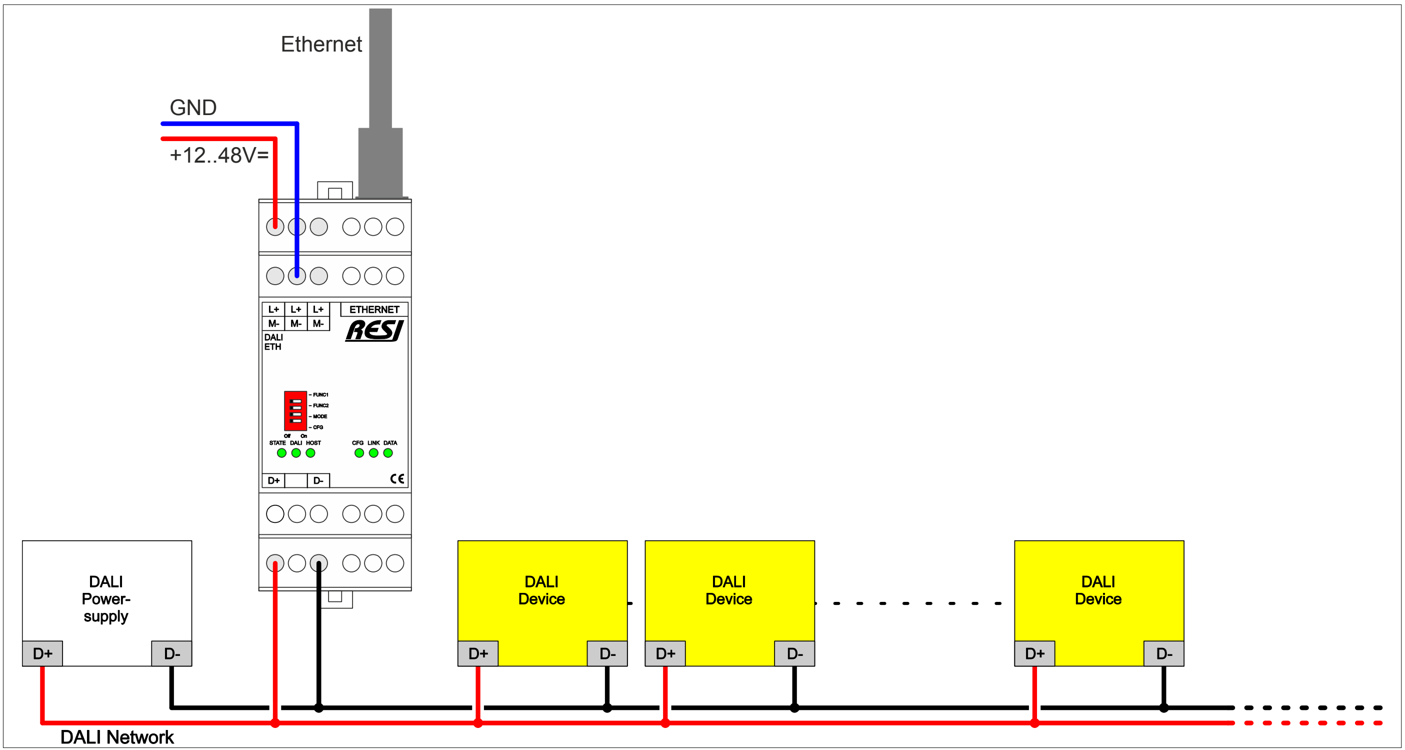


Figure: Connecting a DALI bus system to the RESI-DALI-ETH gateway with external DALI power supply

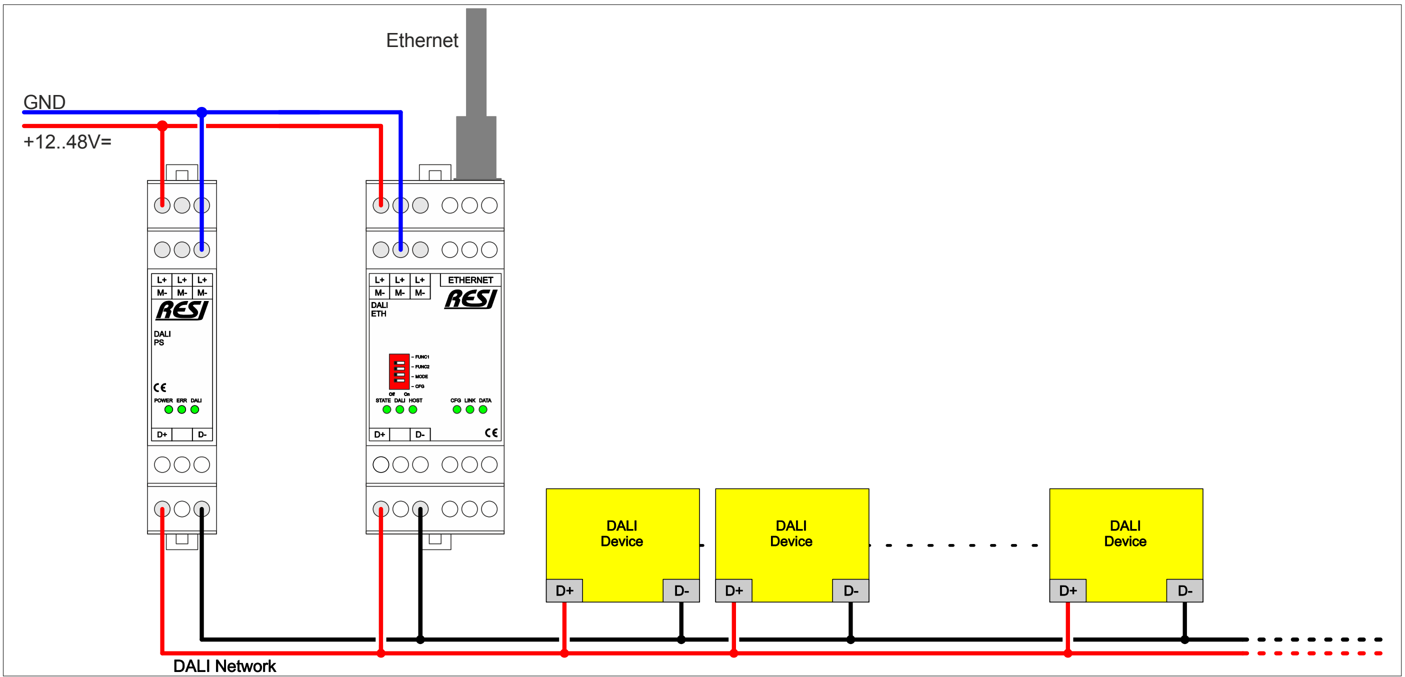


Figure: Connecting a DALI bus system to the RESI-DALI-ETH gateway with RESI-DALI-PS DALI power supply

45.6 RESI-DALI-PS: Connection diagram

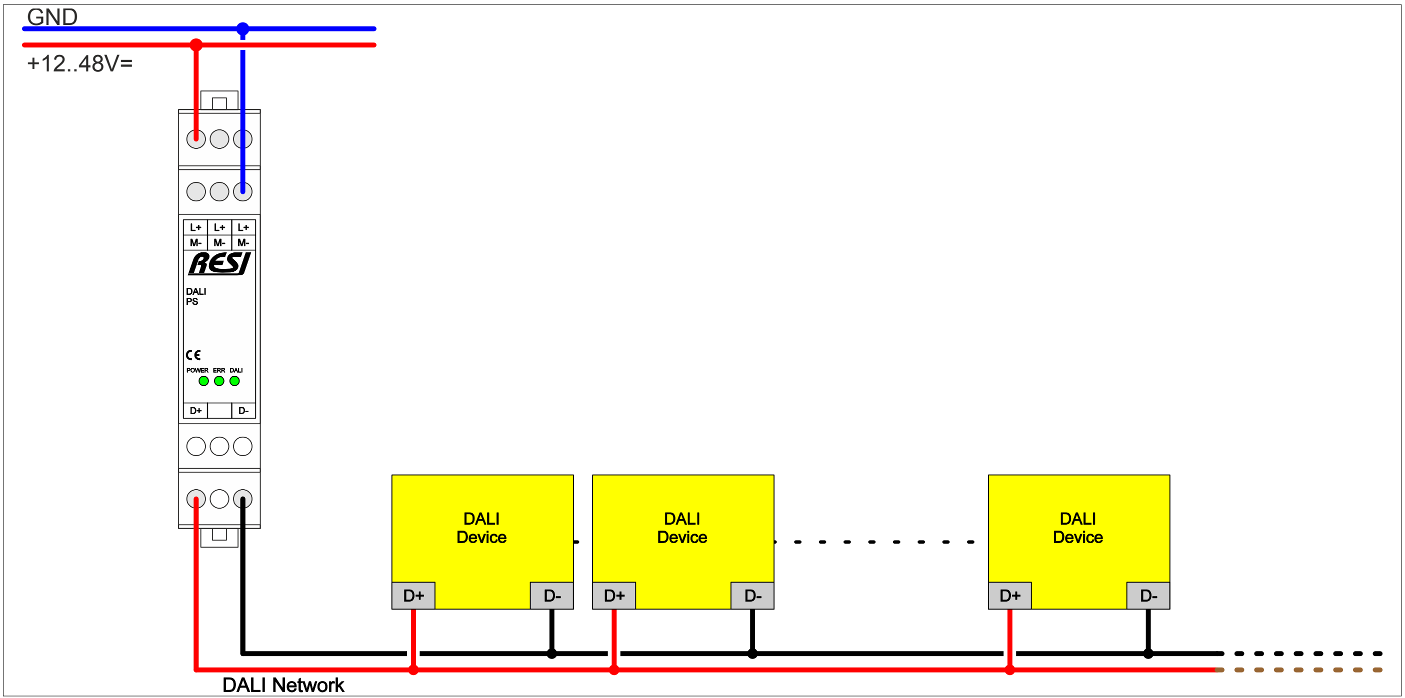


Figure: Establishing power supply to a DALI bus system with our RESI-DALI-PS

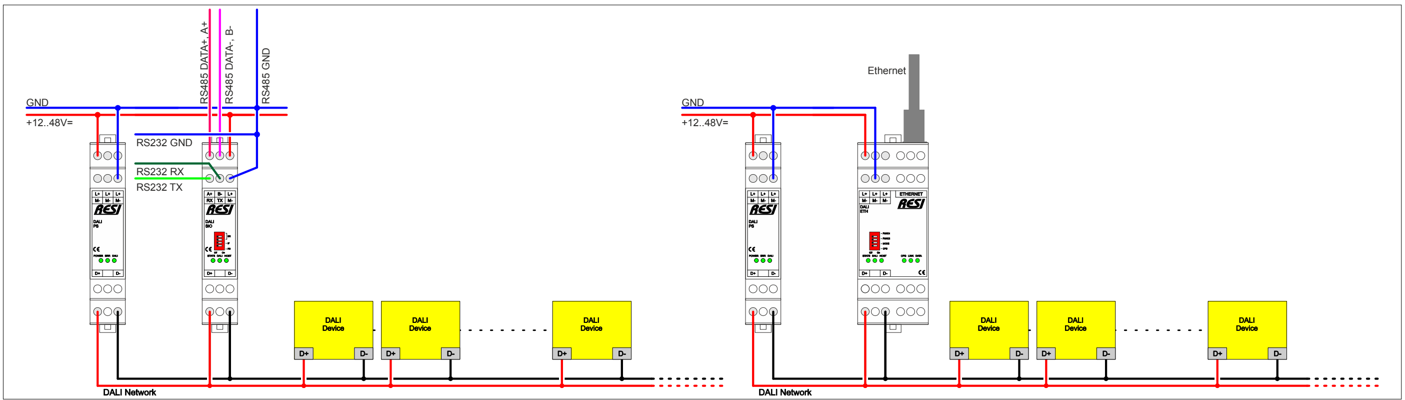


Figure: Using our RESI-DALI-PS DALI power supply in combination with our DALI gateways

45.7 DALI bus installation

There are many common pitfalls in using a DALI light bus system. We give answers to the most common questions around the DALI bus system here:

- What are the logical DALI limits
  - A DALI bus system can address 64 DALI lamps or DALI 1.0 devices
  - A DALI bus system can address 16 logical groups of DALI lamps or devices
  - A DALI bus system can address 64 DALI 2.0 control gears or DALI 2.0 devices like presence detectors, light sensors, motion detectors, manual control units
  - A DALI bus system can address 64 eDALI control gears like presence detectors, light sensors, motion detectors, manual control units (proprietary to LUNATONE)
  - Be careful, how many devices you use on your DALI bus, it depends strongly on your DALI power supply current.

- Maximum DALI cable length
  - The maximum cable length results from the maximum permitted voltage drop on the DALI cable, it is defined as a maximum of 2 V.
  - This corresponds to a maximum cable length of 300 m with a cable cross-section of 1.5 mm².
  - CAUTION: When designing the maximum cable length, the contact resistances must also be observed! 2 V voltage drop must not be exceeded!

- Do you have more than 64 DALI ballasts on a DALI line with a DALI power supply?
  - DALI only allows a maximum of 64 ballasts on a bus line!
  - Divide the DALI bus into two separate bus lines and use two DALI power supplies

- Is your bus system longer than 300m?
  - Separate the bus system into several separate segments with your own DALI power supplies and DALI master

- Measure the DALI output voltage on the DALI-MASTER. This must be around 14 to 18V!
  - Too many lights with ballasts on the DALI bus?
  - Do the ballasts use more power than the DALI power supply can deliver?
  - Usually the DALI power supplies deliver 200mA or 250mA of current

- Does the DALI voltage drop at the ballasts?
  - There may be a maximum voltage drop on the DALI bus of 2V between the DALI supply and the DALI ballast.
  - In the event of a large voltage drop, DALI communication no longer works reliably!
  - Measure this with EVERY ballast using a voltmeter!
    - First check whether all DALI devices are working.
    - Make sure that there is no communication on the DALI line.
    - Measure the voltage on the DALI power supply.
    - The value must be between 11.5 V and 22.5 V; a typical value is 14-16 V.
    - A significantly lower value could indicate a short circuit.
    - Measure the voltage on the DALI device that is furthest away from the DALI power supply.
    - The value must be between 9.5 V and 20.5 V.
    - A much lower value indicates that there is a short circuit somewhere.
    - Create a short circuit between the two DALI bus lines on the DALI device that is furthest away from the DALI power supply.
    - Measure the voltage on the DALI power supply. The value you measure is the DALI voltage drop.
    - This value must not be higher than 2 V.

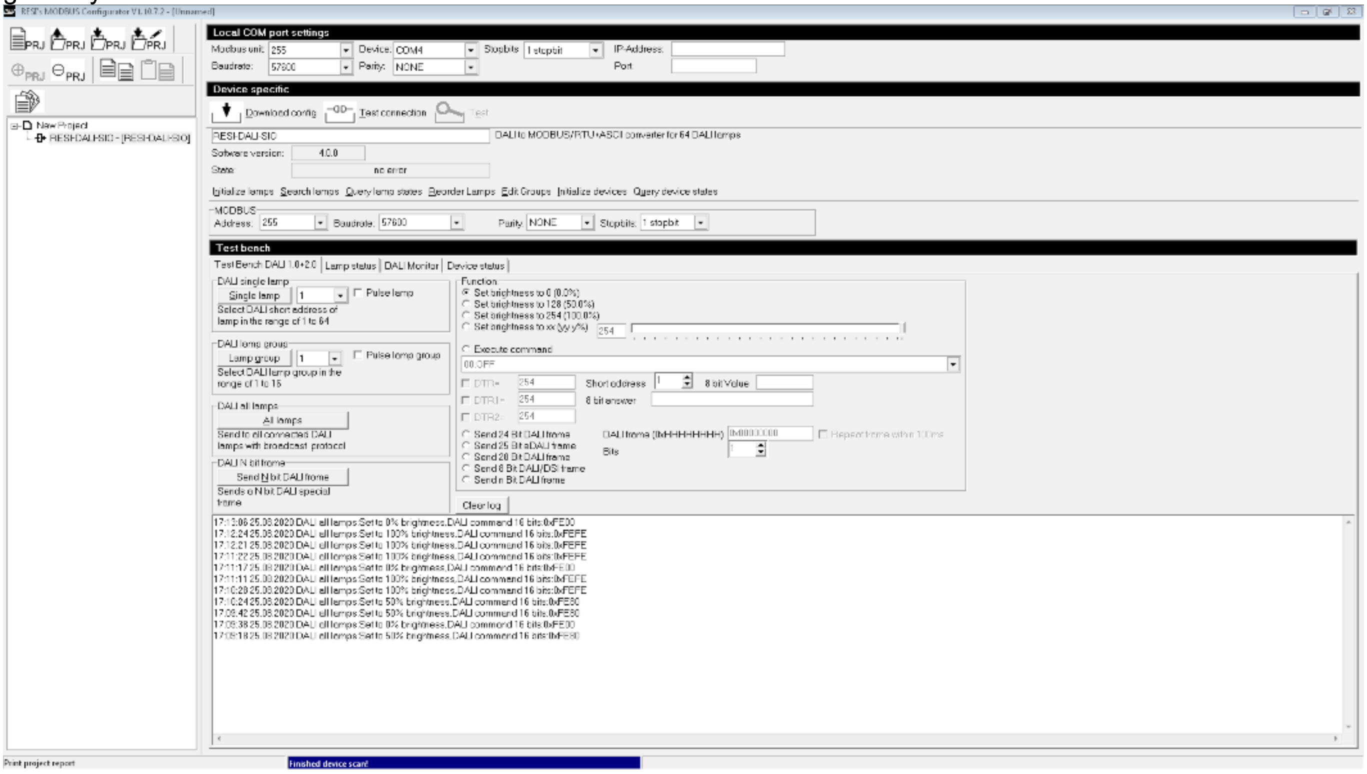
    - If it is higher than 2 V, check whether the following events have occurred:
        - DALI line too long (over 300m with 1.5mm² cross-section)
        - Cross section too small
        - High contact resistance
        - The value must be brought below 2 V.
    - Remove the short circuit between the two DALI bus lines furthest away from the DALI device.
    - This can be solved by dividing the DALI bus system into two separate DALI bus systems

- Your DALI bus cabling must be a tree structure
    - There must be no ring or loop. If so, cut this loop open!

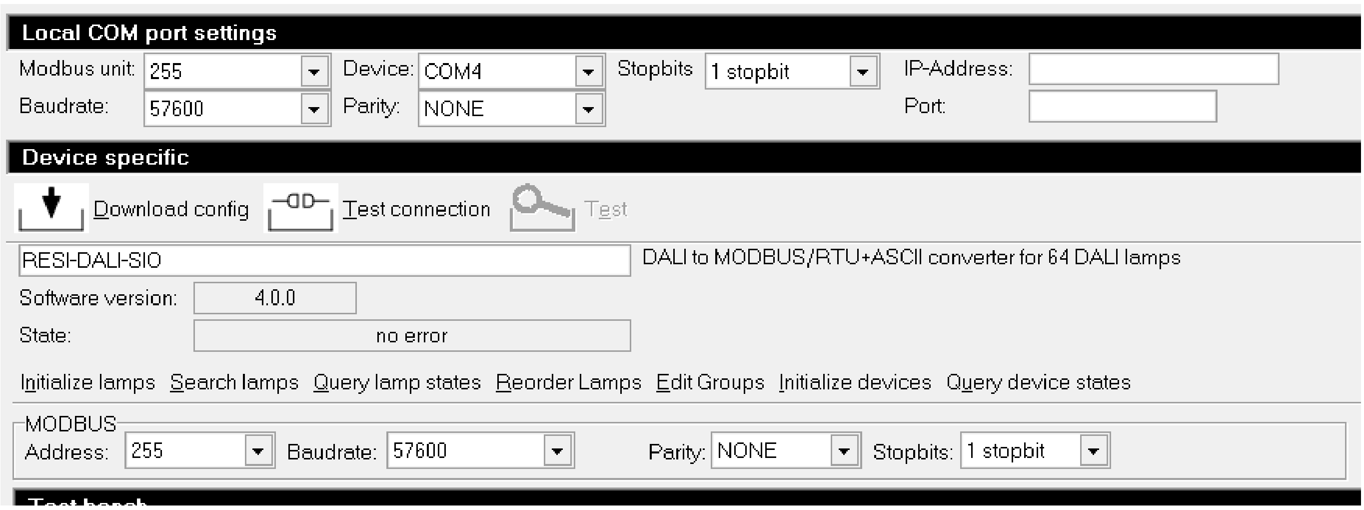
- Recommendations of DALI cable lengths for different conductor cross-sections: DALI cable length:
    - at Ø 1.5mm² max. 300m
    - at Ø 1.0mm² max. 238m
    - at Ø 0.75mm² max. 174m
    - at Ø 0.5mm² max. 116m

45.8 Configuration with MODBUSConfigurator software

Download our free software from our homepage www.RESI.cc and install it on your computer. After you have successfully established a connection, you will see the following picture for the RESI-DALI-SIO or RESI-DALI-ETH gateways:



In the section device specific you will find the following functions:

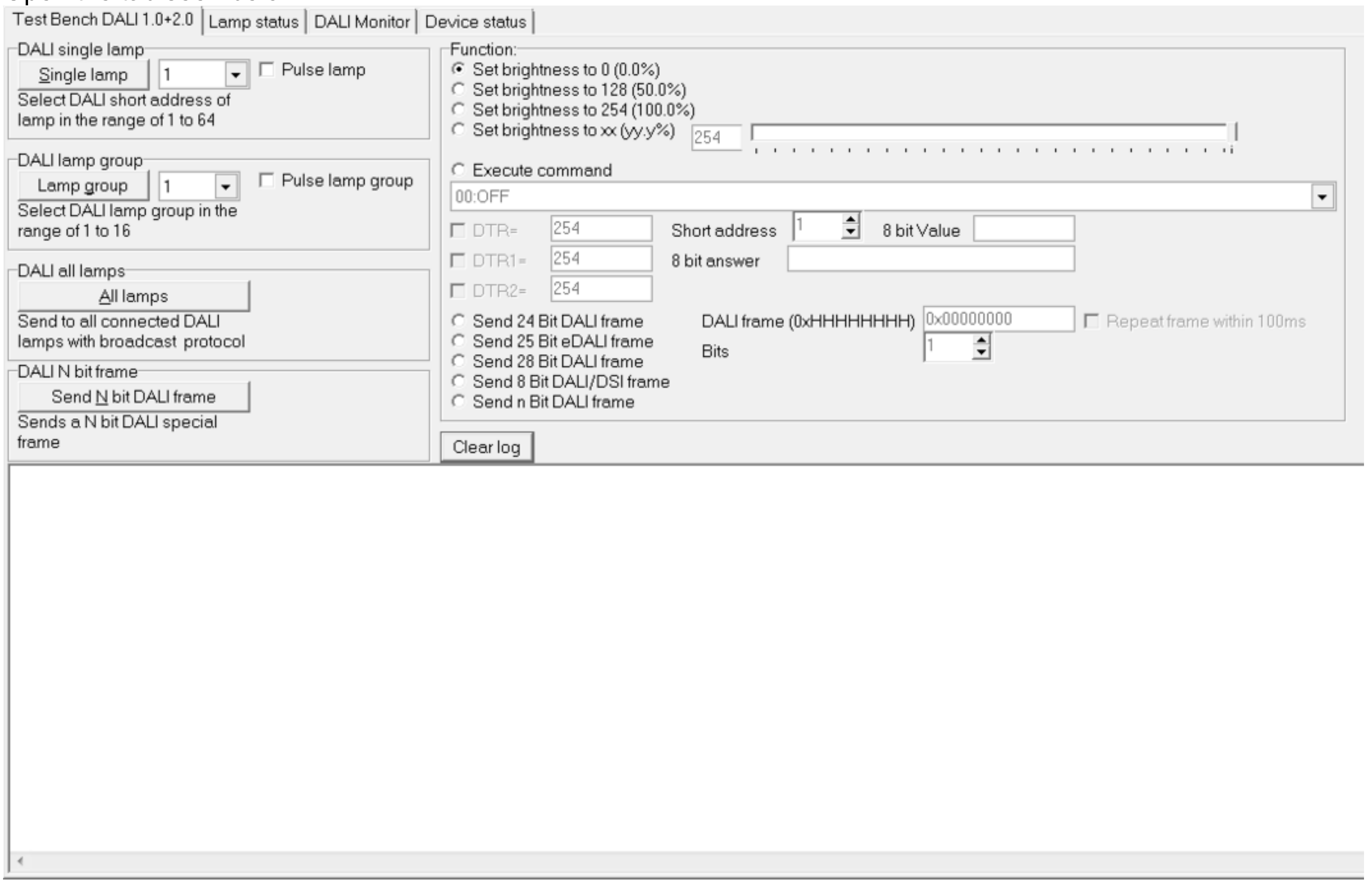


- Button "Download config": If you change the MODBUS/RTU slave address, or the serial parameters, you have to download the new configuration to the gateway to activate the changes.
- Button "Test connection": This button tests, if the software can communicate with the gateway or not.
- Button "Test": This function is not available in the DALI products.
- Button "Initialize lamps": This button opens a dialog window to configure new DALI 1.0 devices with a specific short address. The detailed function is described below.
- Button "Search lamps": This button starts a search for new DALI 1.0 lamps on the DALI bus and adds new lamps to the tree. The detailed function is described below.
- Button "Query lamp states": This button queries from all 64 DALI 1.0 short addresses the current state of the DALI lamps and shows the result in a grid. The detailed function is described below.
- Button "Reorder lamp": This button opens a dialog window, where you can readdress the DALI 1.0 devices and their short addresses. The detailed function is described below.

- Button “Edit Groups”: This button opens a dialog window, where you can edit the groups of a DALI 1.0 device. The detailed function is described below.
- Button “Initialize devices”: This button opens a dialog window, where you can search and address for new DALI 2.0 control gears on the DALI bus line. The detailed function is described below.
- Button “Query device states”: This button queries of all 64 DALI 2.0 short addresses the current state of the DALI control gear and shows the results in a grid. The detailed function is described below.

45.8.1 The Test Bench DALI 1.0+DALI 2.0

The test bench lets you send quickly DALI commands to the DALI bus for test and evaluation of your DALI installation. Open the tab seen below:



For DALI 1.0 devices:
First of all DALI 1.0 sends in general 16 bit DALI frames to a DALI 1.0 lamp and if the lamp gives an answer, this answer is an 8 bit DALI frame. Therefore you can send commands to DALI 1.0 devices in three different addressing schemes:

- **via SHORT ADDRESS of lamp**: The DALI standard allows up to 64 DALI lamps on one bus with a DALI short address between 0 and 63. We use for better understanding the DALI short addresses 1 to 64 in our software.
- **via GROUP ADDRESS of lamp**: The DALI standard allows up to 16 DALI groups on one DALI bus. A DALI lamp with a specific short address can be part of as many DALI groups, as you want it to be part of. Therefore DALI uses the groups 0 to 15, again, we use in our software 1 to 16 for better understanding.
- **via BROADCAST to all lamps**: The DALI standard supports a special addressing modes to send a DALI command to all connected DALI lamps. This is very useful for easy testing.

So, choose a DALI command from the Function area and the use the buttons "Single lamp" to send the command to a specific short address, "Lamp group" to send the command to a specific group of lamps or "All lamps" to send the command to all connected lamps.

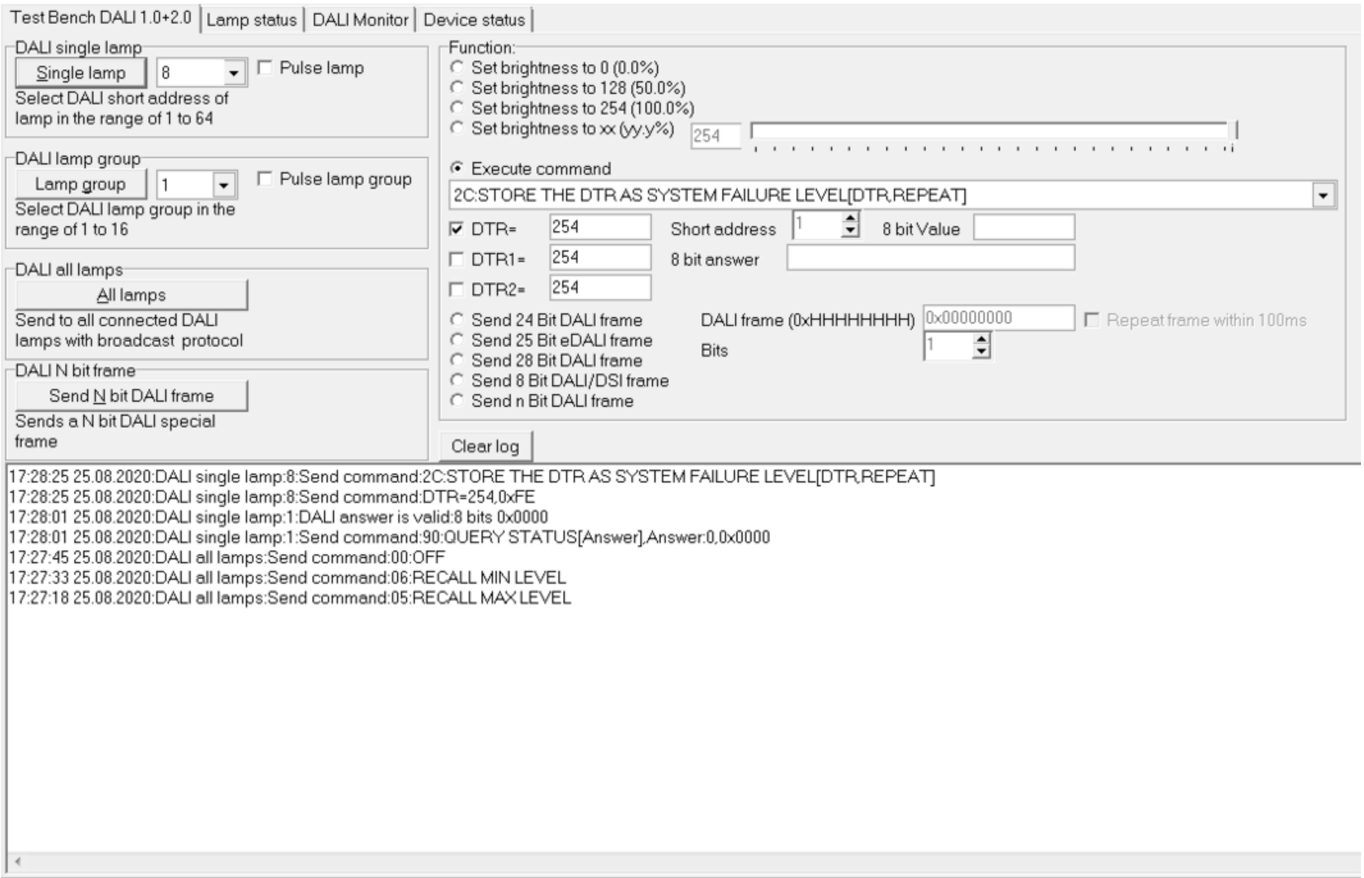
Use the checkbox **Pulse lamp** to quickly find the lamp with the selected short address of the drop down list. The lamp will pulse the brightness to find the lamp easily.

Use the checkbox **Pulse lamp group** to quickly find a lamp group with the selected group of the drop down list. All lamps of the selected lamp group will pulse the brightness to find them easily.

The following functions are available for DALI 1.0 lamps:

- **Set brightness to ...**: With this command you set the brightness of the addressed lamps to the specific value.

- **Execute command**: With this command you can select a specific DALI 1.0 command from the drop down list and send it to the addressed lamps. For example choose the command 05:RECALL MAX LEVEL and press the button "All lamps". All connected DALI 1.0 lamps should dim up to the stored maximum brightness level. Some of the DALI commands will generate an answer. The answer will be shown in the log window below the command group. Also some of the commands need DTR, DTR1 or DTR2 register values. Therefore you have the check boxes before DTR, DTR1 and DTR2. This will activate automatic sending of DTR setup commands before executing the desired command. Some of the DALI 1.0 commands must be repeated within 100ms twice. This is also done be the gateway internally.

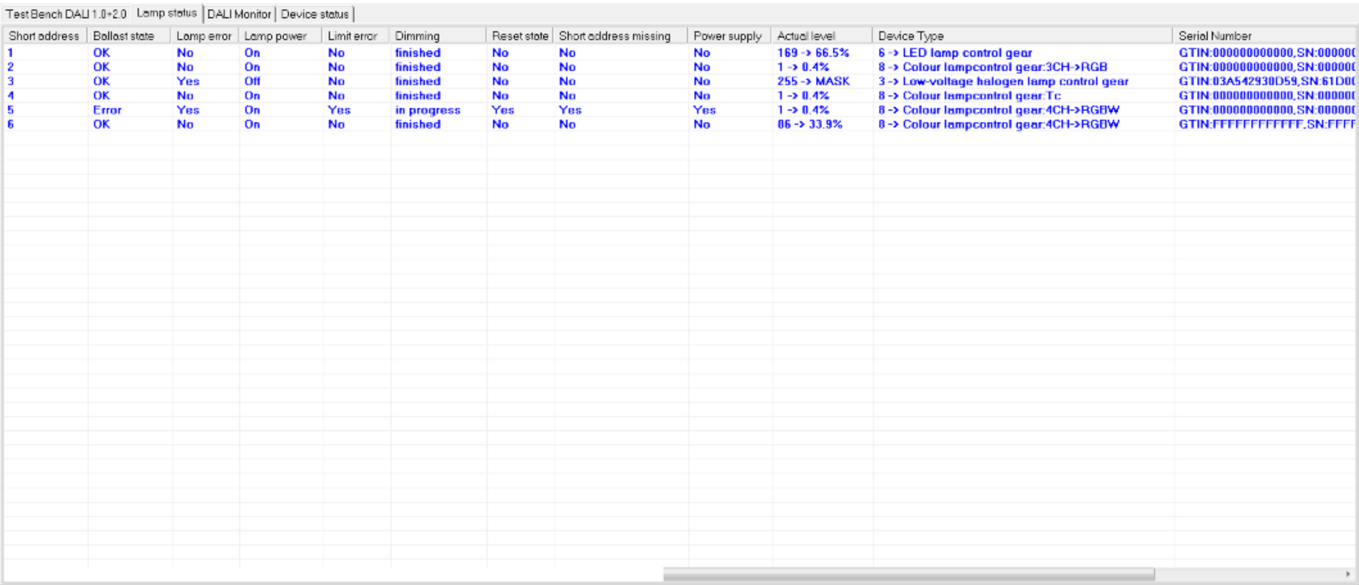


For DALI 2.0 devices:
The DALI 2.0 standard introduces new DALI frame formats. Therefore we have implemented a test functionality into the gateway: The functions "Send x bits DALI frame" offers a wide variety of possibilities to send new DALI 2.0 frames.

Enter a valid DALI 2.0 frame and click on the button "Send N bit DALI frame". You will see the result in the log window below.

45.8.2 Query lamp states

This function tries to read out basic data of all DALI 1.0 devices from the connected DALI bus. The result will be displayed in the grid view under the tab "Lamp status". A possible result could look like this:



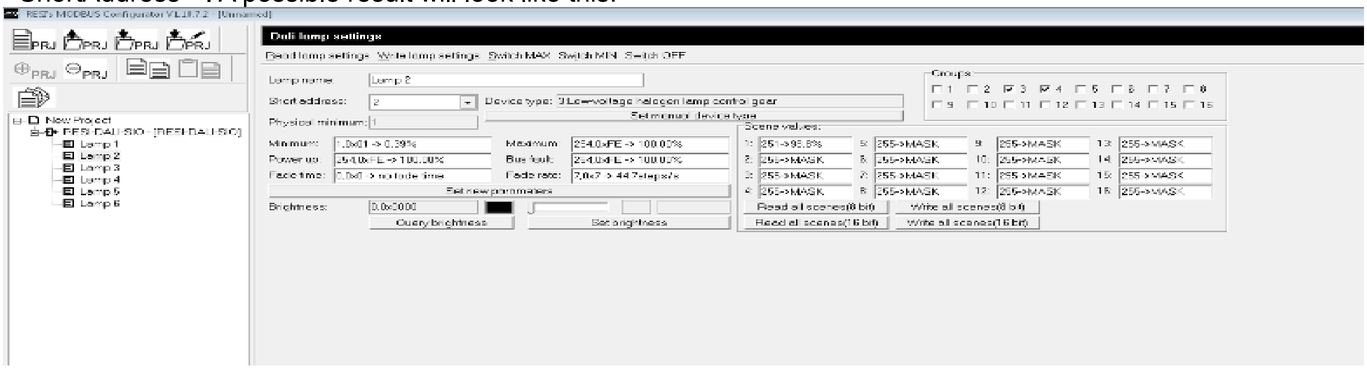
Short address	Ballast state	Lamp error	Lamp power	Limit error	Dimming	Reset state	Short address missing	Power supply	Actual level	Device Type	Serial Number
1	OK	No	On	No	finished	No	No	No	169 -> 66.5%	6 -> LED lamp control gear	GTIN:000000000000,SN:00000(
2	OK	No	On	No	finished	No	No	No	1 -> 0.4%	8 -> Colour lampcontrol gear:3CH->RGB	GTIN:000000000000,SN:00000(
3	OK	Yes	Off	No	finished	No	No	No	255 -> MASK	3 -> Low-voltage halogen lamp control gear	GTIN:03A542930D59,SN:61D0(
4	OK	No	On	No	finished	No	No	No	1 -> 0.4%	8 -> Colour lampcontrol gear:Tc	GTIN:000000000000,SN:00000(
5	Error	Yes	On	Yes	in progress	Yes	Yes	Yes	1 -> 0.4%	8 -> Colour lampcontrol gear:4CH->RGBW	GTIN:000000000000,SN:00000(
6	OK	No	On	No	finished	No	No	No	86 -> 33.9%	8 -> Colour lampcontrol gear:4CH->RGBW	GTIN:FFFFFFFFFFFF,SN:FFFF

You will find the following information in this grid view:

- **Short address**: The current short address of the tested lamp
- **Ballast state**: The current state of the ballast (OK or Error) monitored by the DALI lamp controller.
- **Lamp error**: If the DALI lamp controller has detected an error in the ballast in in the control gear, it will return Yes. Otherwise it will return No.
- **Lamp power**: If the DALI lamp controller currently powers the ballast, the return value will be On, else the return value will be Off.
- **Limit error**: If the brightness of the affected DALI lamp is not within the internal programmed limits of the DALI lamp controller, the return value will be Yes, else the return value will be No.
- **Dimming**: This flag (In progress or Finished) indicates if actual a dimming or fading is running in the DALI lamp controller (In progress). If the dimming or fading has ended Finished will be returned.
- **Reset state**: This flag (Yes or No) indicates if the DALI lamp is in reset state (Yes) or not (No).
- **Short address missing**: This flag (Yes or No) indicates if the DALI lamp has no programmed short address (Yes). If there is a short address programmed, the answer will be (No)
- **Power supply**: This flag (Yes or No) indicates if the DALI lamp has an error or problem with the power supply (Yes) or if the power supply of the DALI lamp is ok (No).
- **Actual level**: This is the actual brightness value of the lamp between 0 and 254 (0 to 100% scaled linear!) or 255 for MASK.
- **Device type**: This is the actual device type of the DALI lamp. Additional text information is shown to the basic device type.
- **Serial Number**: If possible to readout the correct value this is the serial number of the DALI lamp.

45.8.3 Search lamps

This function tries to read out basic data of all DALI 1.0 devices from the connected DALI bus. For each DALI 1.0 lamp, the software adds a leaf into the project tree under the selected DALI converter with the Name "Lamp <ShortAddress>". A possible result will look like this:



On the left side in the project tree you will see, that there are 6 lamps connected to the DALI bus line. We have selected Lamp 2 for detail view on the right side. The lamp is a DALI device type 3 lamp (Low-voltage halogen lamp control gear) with standard DALI 1.0 commands and parameters.

In this dialog you can execute the following commands:

- **Read lamp settings**: This commands will re-read the current settings of the DALI ballast and displays the results on this page
- **Write lamp settings**: This command will write the changed settings from the PC to the DALI lamp control gear and store the new values in the control gear.
- **Switch MAX**: This command will switch the DALI lamp on and dim up to the programmed maximum brightness level stored in the DALI lamp control gear.
- **Switch MIN**: This command will switch the DALI lamp on and dim up to the programmed minimum brightness level stored in the DALI lamp control gear.
- **Switch OFF**: This command will switch the DALI lamp off.
- **Set new parameters**: This command will store the user changed parameters for the lamp. How you can edit the parameters, please read "How to edit the parameters" section below. You can modify the standard parameters Minimum, Maximum, Power Up, Bus fault Fade time or Fade rate with this function. This are basic setup values for almost every DALI lamp. But you have to select the function "Write lamp settings" to download the changes into the DALI lamp control gear.
- **Read all scenes (8 bit)**: This command will read out all brightness values for all 16 scenes stored in the DALI lamp control gear. The result will be a brightness value in the range 0 to 254 or 0x00 to 0xFE for 0.0% to 100.0% and 255 or 0xFF for MASK.
- **Read all scenes (16 bit)**: This command will read out all brightness values for all 16 scenes stored in the DALI lamp control gear using 16 Bit data format. The result will be a brightness value in the range 0 to 65279 or 0x0000 to 0xFEFF for 0.0% to 100.0% or 65280 to 65535 or 0xFF00 to 0xFFFF for MASK.
  CAUTION: Not all DALI 1.0 devices support 16 Bit data mode. So better use only 8 bit data mode functions, if you don't know exactly, if your control device supports this data mode!
- **Write all scenes (8 bit)**: This command will immediately write the new scene values to the DALI lamp control gear.
- **Write all scenes (16 bit)**: This command will immediately write the new scene values to the DALI lamp control gear with 16 bit data format.
  CAUTION: Not all DALI 1.0 devices support 16 Bit data mode. So better use only 8 bit data mode functions, if you don't know exactly, if your control device supports this data mode!
- **Query brightness**: This command will query the actual brightness of the selected lamp.
- **Set brightness**: The command will set the new selected brightness for the current lamp in the DALI lamp control gear. Select the brightness with the slider or the up down keys, if you have focused on the slider. Or double click on the text field with current selected brightness to edit the brightness as a text value. The choose this button to generate the DALI command.
- **Groups**: Select the desired groups for this control gear by checking the Groups 1 to 16. The press the button Write lamp settings to download the group selection into the control gear.

How to edit the parameters:

- **Brightness**: To change a brightness value, enter a valid number between 0 and 254 or 0x00 and 0xFE for the brightness 0% and 100%, 255 or 0xFF for MASK or 0.0 to 100.0 for a brightness level in percent. Enter the word MASK for the MASK value of 255.

- **Fade time**: To change a fade time, enter a valid number between 0 and 255 or 0x00 to 0xFF for the fade time or 0.0 to 90.5 for a fade time in seconds. Enter the word NO for no fade time (value 0).

- **Fade rate**: To change a fade rate, enter a valid number between 0 and 255 or 0x00 to 0xFF for the fade rate or 0.0 to 357.8 for a fade rate in steps/seconds. Enter NO for no fade rate (value 0).

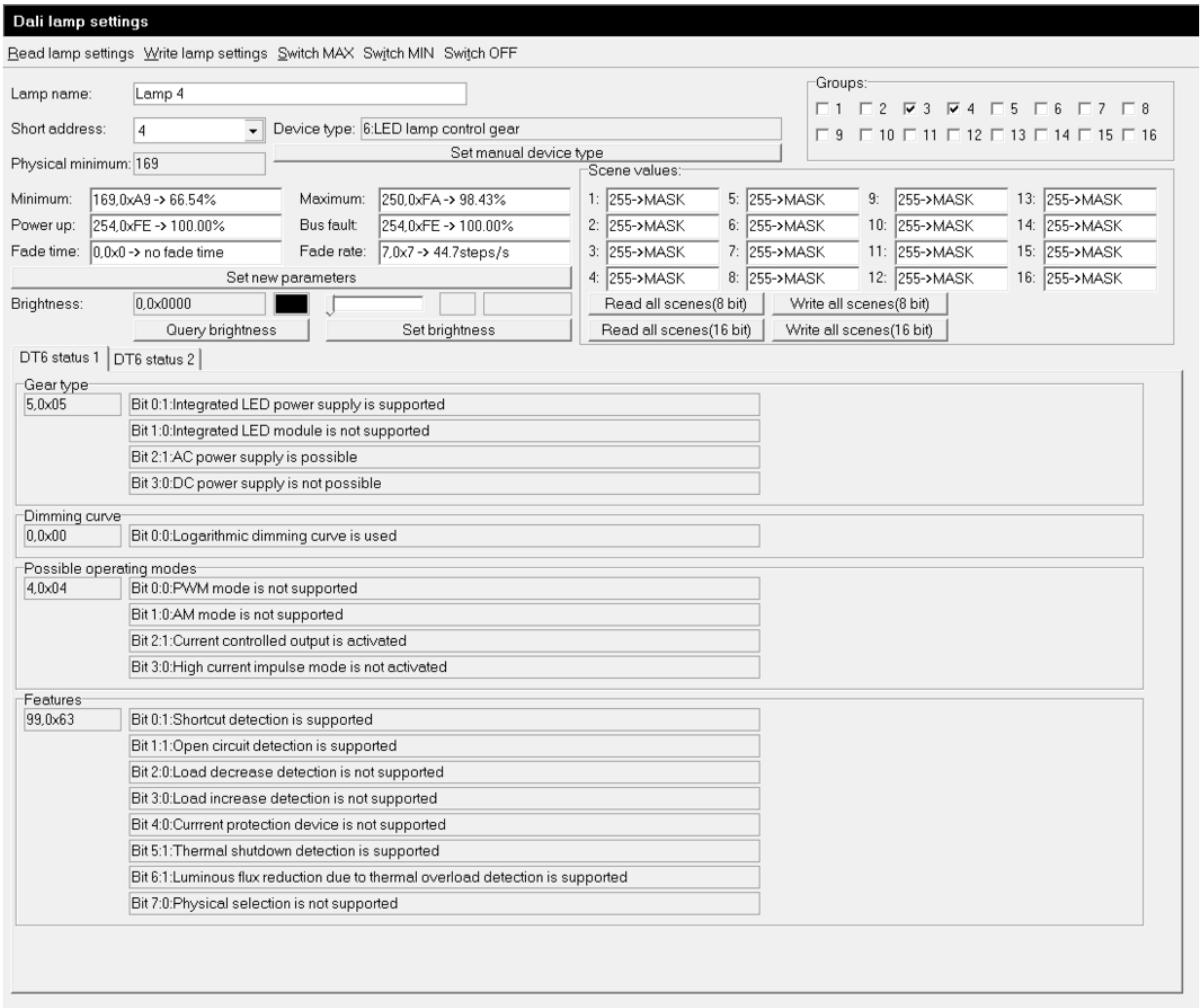
- **Color parts RGB or WAF**: To change a color part, enter a new value for 8 bit mode between 0 and 254 or 0x00 to 0xFE for 0.0% to 100.0%, or 255 or 0xFF for MASK value. Enter a new value for 16 bit mode between 0 and 65279 or 0x0000 and 0xFEFF for 0.0% to 100.0% or 65280 to 65535 or 0xFF00 to 0xFFFF for MASK. Or enter a percentage value between 0.0% and 100.0%. Enter the word MASK for MASK.

- Color temperature Tc: To change a color temperature Tc, enter a new value for 16 bit mode between 0 and 65279 or 0x0000 and 0xFEFF or 65280 to 65535 or 0xFF00 to 0xFFFF for MASK. Or enter a KELVIN value between 1.0 and 1000000.0 Kelvin. Enter the word MASK for MASK.

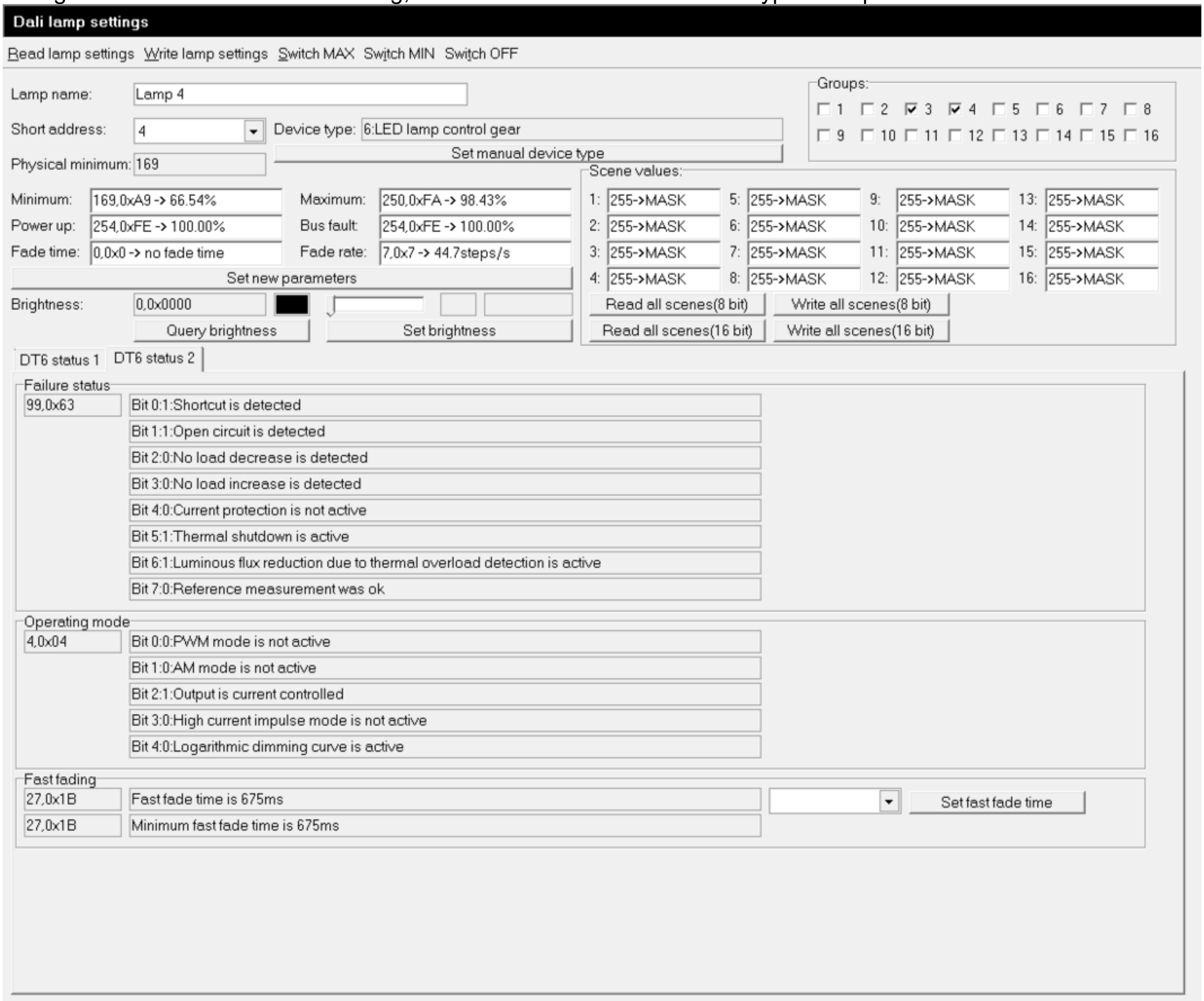
- **Primary N**: To change a primary N channel, enter a new value for 16 bit mode between 0 and 65279 or 0x0000 and 0xFEFF or 65280 to 65535 or 0xFF00 to 0xFFFF for MASK. Or enter a percentage value between 0.0 and 100.0. Enter the word MASK for MASK.

Device type 6 lamps:

For lamps of DALI device type 6 you will see additional information in two tabs. The first tab shows more information about the DALI lamps itself and its features.

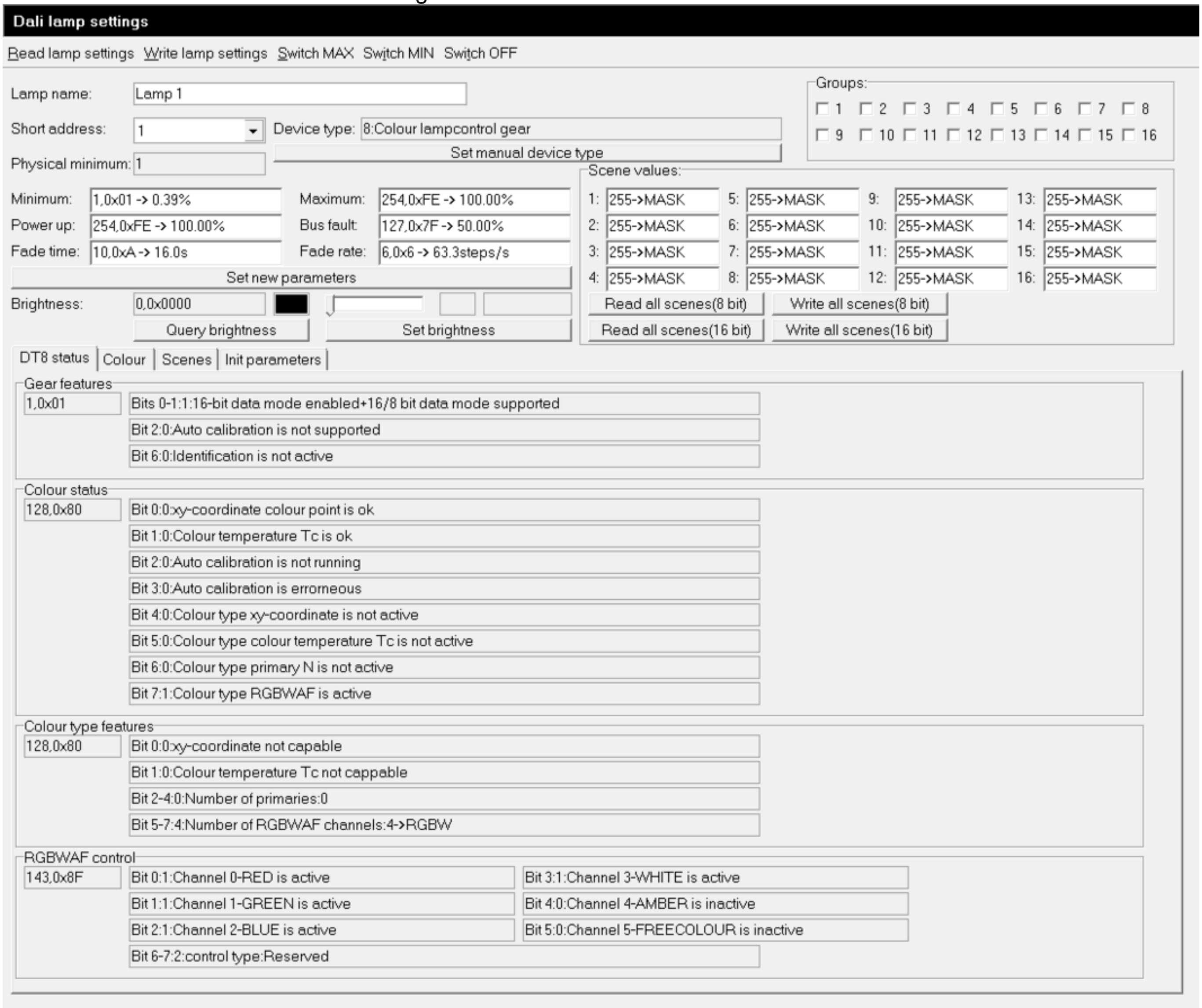


The second page shows information about the failures or the operating mode of the DALI DT 6 lamp. Also you can change the fast fade time in this dialog, which is new to the DALI device type 6 lamps.



Device type 8 lamps:

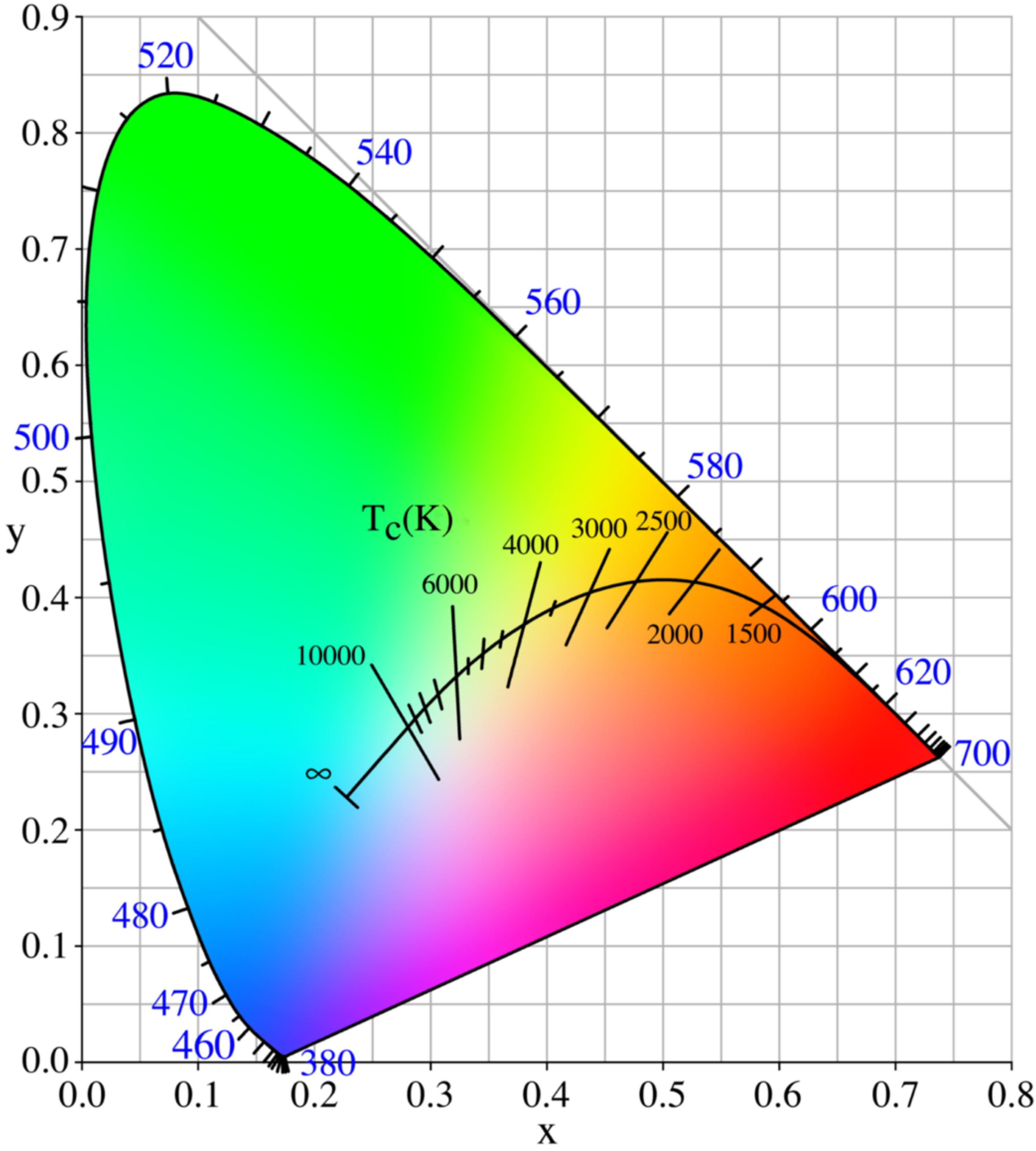
For DALI device type 8 lamps additional information is displayed below the standard information of a DALI lamp. The tab "DT8 status" shows the following information:



Especially the section "Color type features" shows you, what type of color commands your DALI DT8 lamp controller can handle. Also the amount and type of dimming channels is displayed here.

Please consult the DALI standard to understand the different DALI color schemes. basically DALI 2.0 supports:

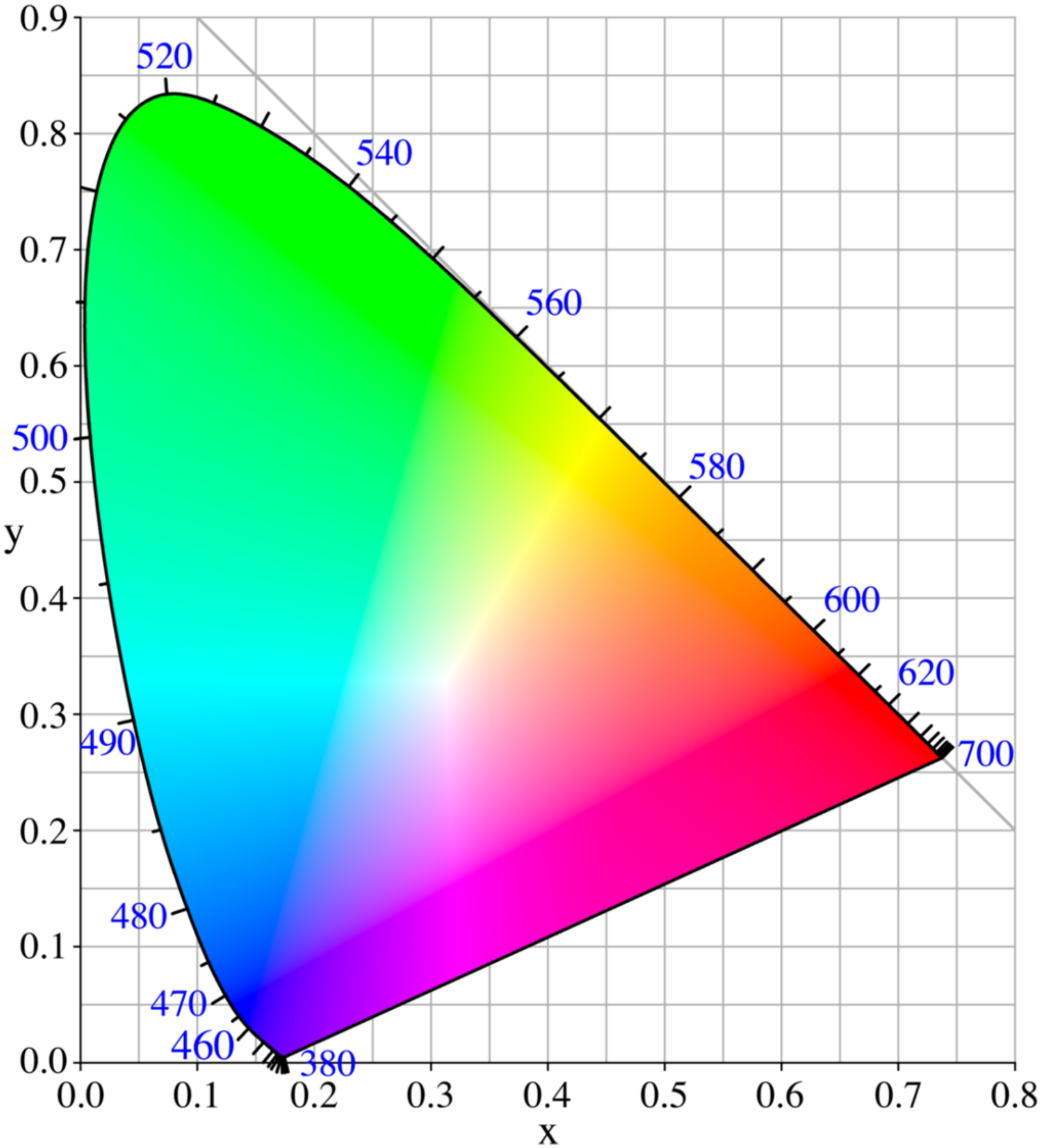
- Color type Tc (Tunable White): Allows control of the correlated color temperature (CCT) along the black-body line, from warm white to cool white.



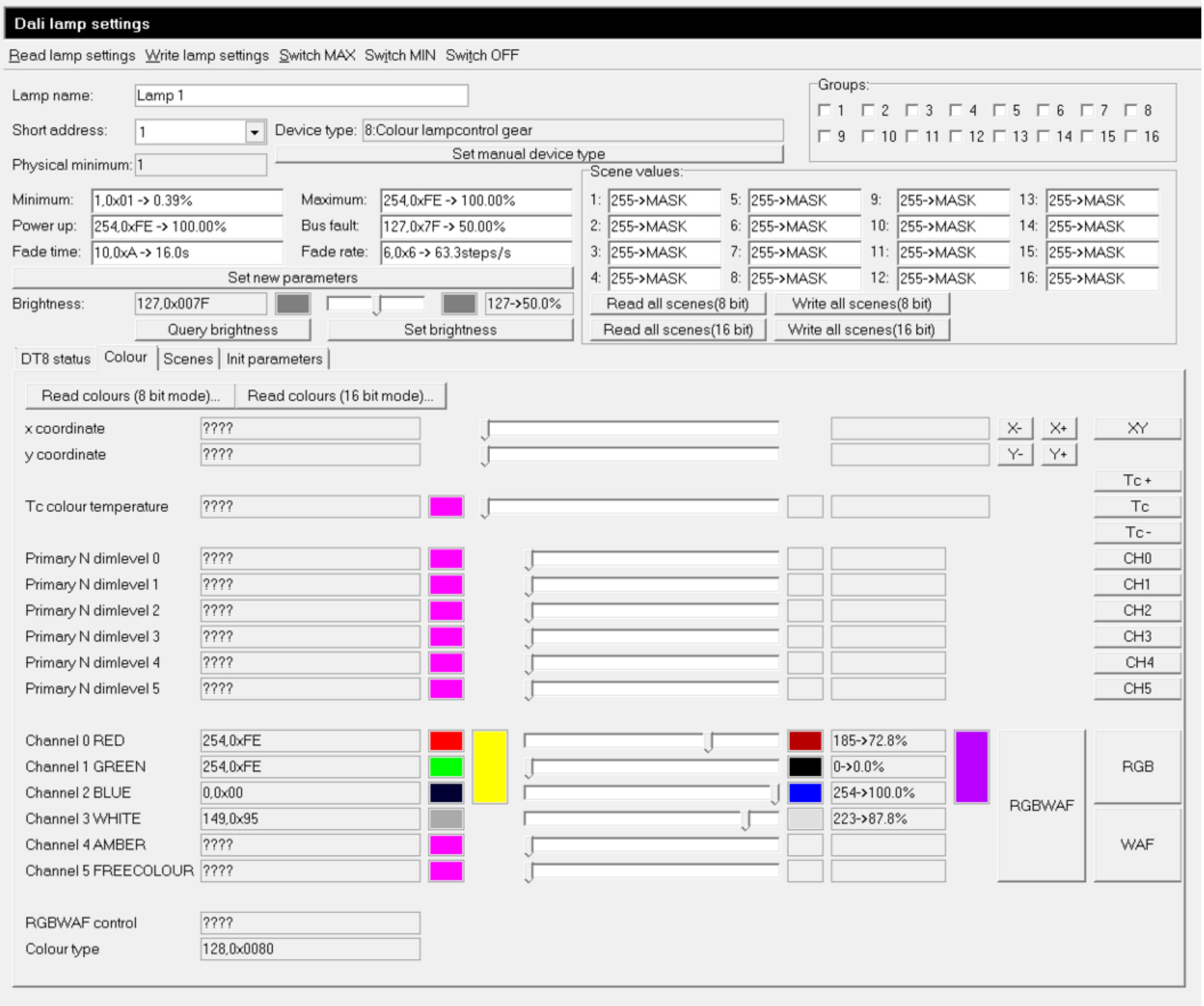
- **Color type primary N :** Allows simple control of up to 6 dimmable channels .

- Color type RGBWAF: Allows simple control of up to 6 channels of color (Red, Green, Blue, White, Amber and Free-color).

■ **Color type xy coordinate:** Also known as xy chromatically, this allows precise and repeatable selection of the color co-ordinates from the CIE color space chromaticity diagram (1931).



On the tab Color, you can read and write new color settings to the lamp:



In this tab page you can execute the following commands:

- **Read colors (8-bit mode)**: This commands will read the current settings of all colors of the DALI ballast and displays the results on this page. Only the supported color values are displayed here. Not every DALI ballast can handle all four color profiles defiled by the DALI standard.

- **Read colors (16-bit mode)**: This commands will read the current settings of all colors of the DALI ballast with 16 bit data mode and displays the results on this page. Only the supported color values are displayed here. Not every DALI ballast can handle all four color profiles defiled by the DALI standard.
CAUTION: Not all DALI 1.0 devices support 16 Bit data mode. So better use only 8 bit data mode functions, if you don't know exactly, if your control device supports this data mode!

- **RGB**: The command will set the new RGB colors (red, green and blue) for the current lamp in the DALI lamp control gear. Select the red, green and blue parts of your desired color with the slider or the up down keys, if you have focused on the slider. Or double click on the text field with current selected color part to edit the color part as a text value. Then choose this button to generate the DALI DT8 commands.

- **WAF**: The command will set the new WAF colors (white, amber, freecolor) for the current lamp in the DALI lamp control gear. Select the white, amber and freecolor parts of your desired color with the slider or the up down keys, if you have focused on the slider. Or double click on the text field with current selected color part to edit the color part as a text value. Then choose this button to generate the DALI DT8 commands.

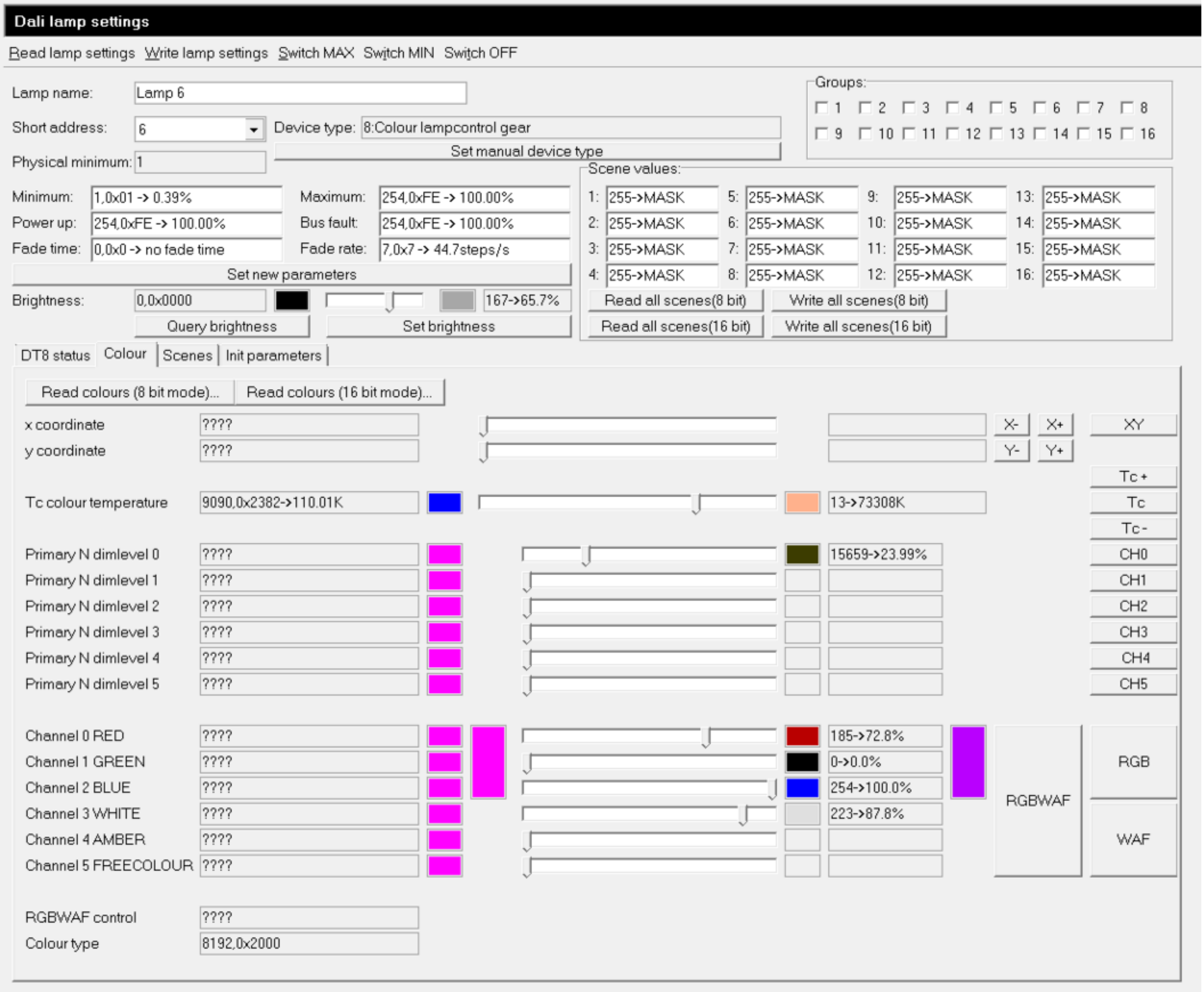
- **RGBWAF**: The command will set the new RGB and WAF colors (red, green blue, white, amber, freecolor) for the current lamp in the DALI lamp control gear. Select the red, green, blue, white, amber and freecolor parts of your desired color with the slider or the up down keys, if you have focused on the slider. Or double click on the text field with current selected color part to edit the color part as a text value. Then choose this button to generate the DALI DT8 commands.

- **Tc**: The command will set the new color temperature Tc between 0 and 100000K for the current lamp in the DALI lamp control gear. Select your desired color temperature Tc with the slider or the up down keys, if you have focused on the slider. Or double click on the text field with current selected color part to edit the color temperature as a text value. The choose this button to generate the DALI DT8 commands.

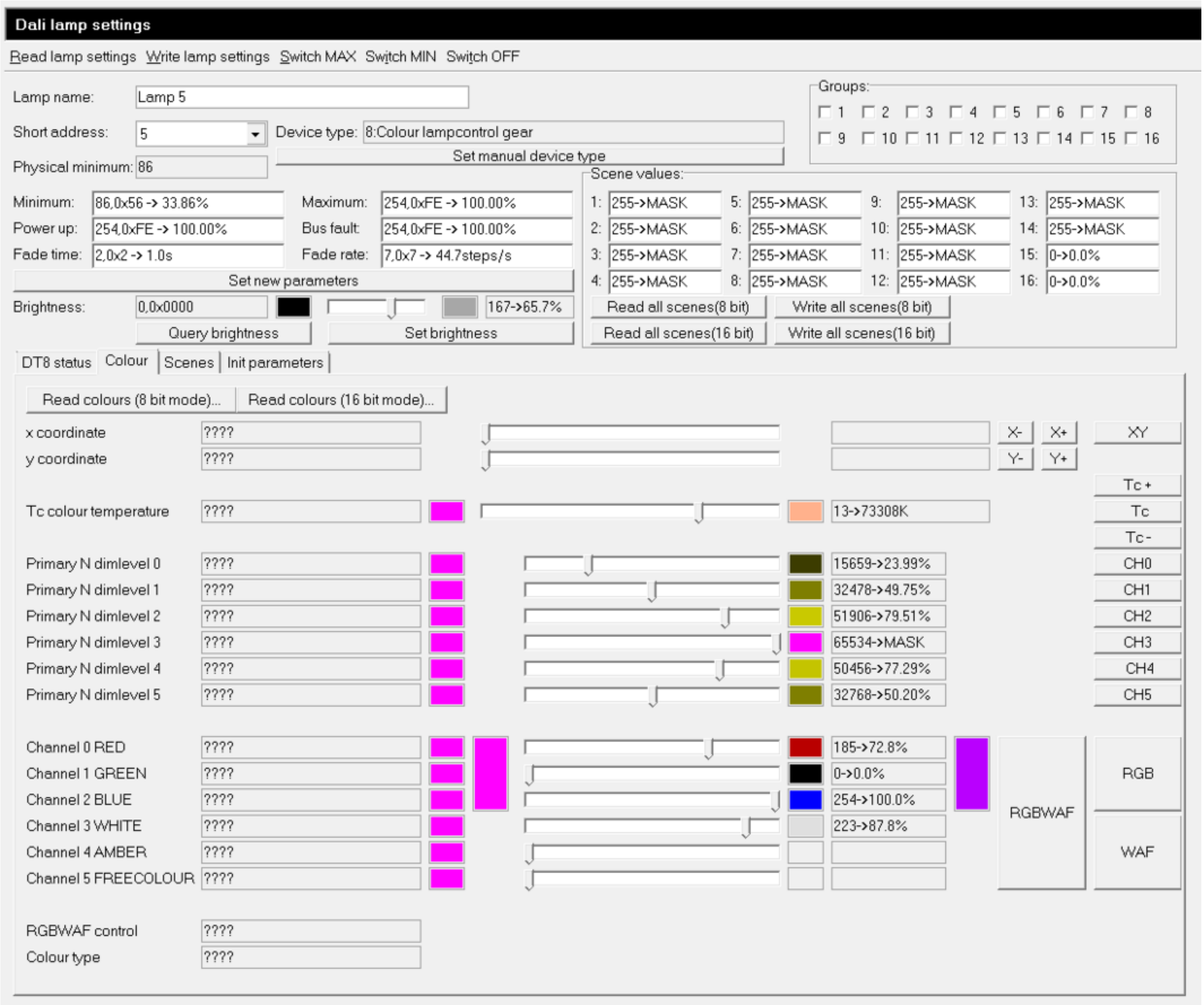
  HINT: Color temperature Tc mode uses 16 bit data mode. So first read out the current colors with the button Read colors (16 bit mode). Then write the new color temperature to the lamp with this button or use Tc+ and Tc- button.

- **Tc+**: This command will step the current color temperature a little bit warmer.

- **Tc-**: This command will step the current color temperature a little bit colder.



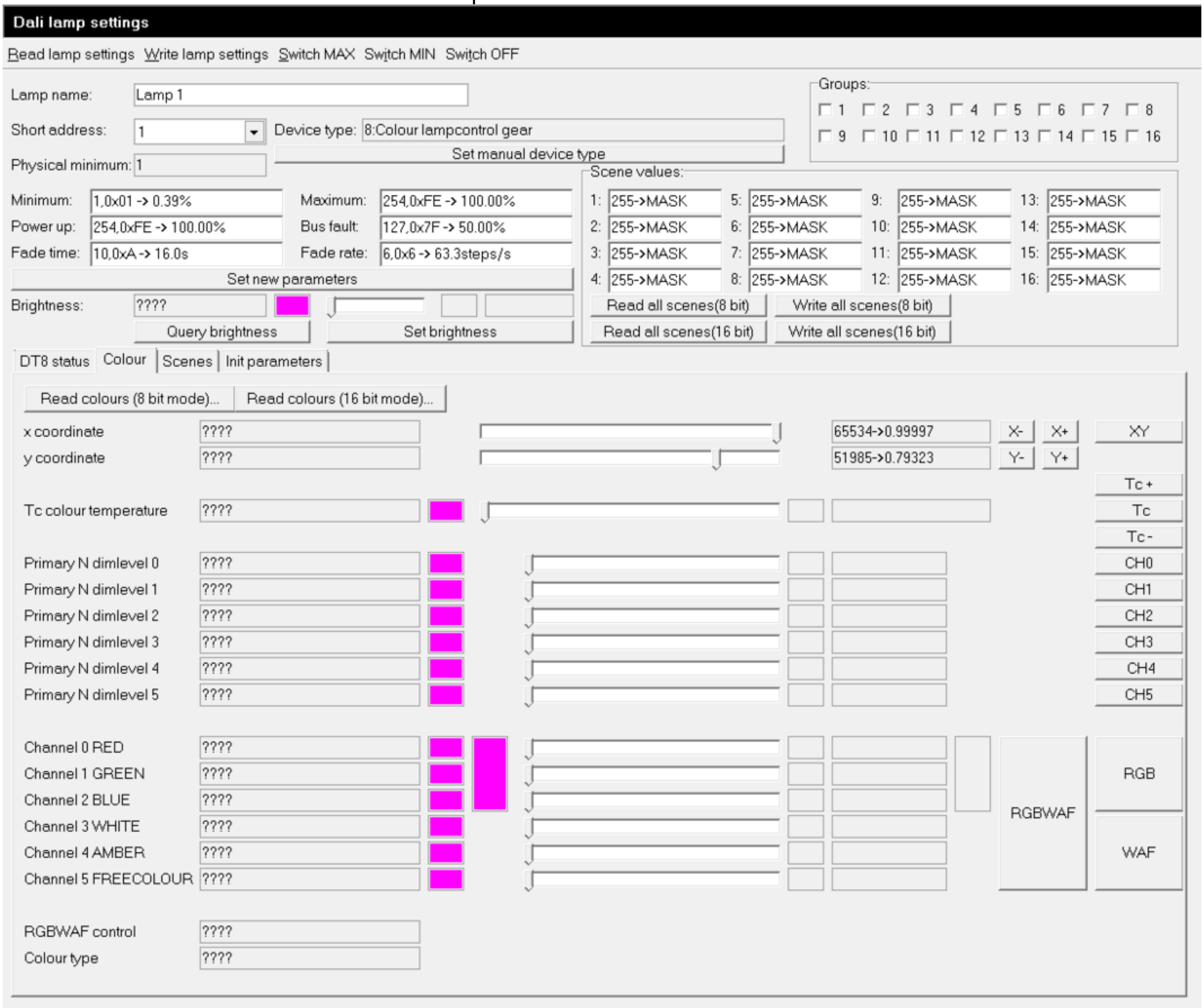
- **Primary N**: Some lamps define up to 6 dimmable channels without specific color dedication. This is the primary N mode of a DALI 2.0 DT8 lamp. You can choose for every channel a value either in 8 bit mode or in 16 bit mode, depending, how you have read out the colors before this command. With the buttons CH0 to CH6 you can set the new values for each channel individually.



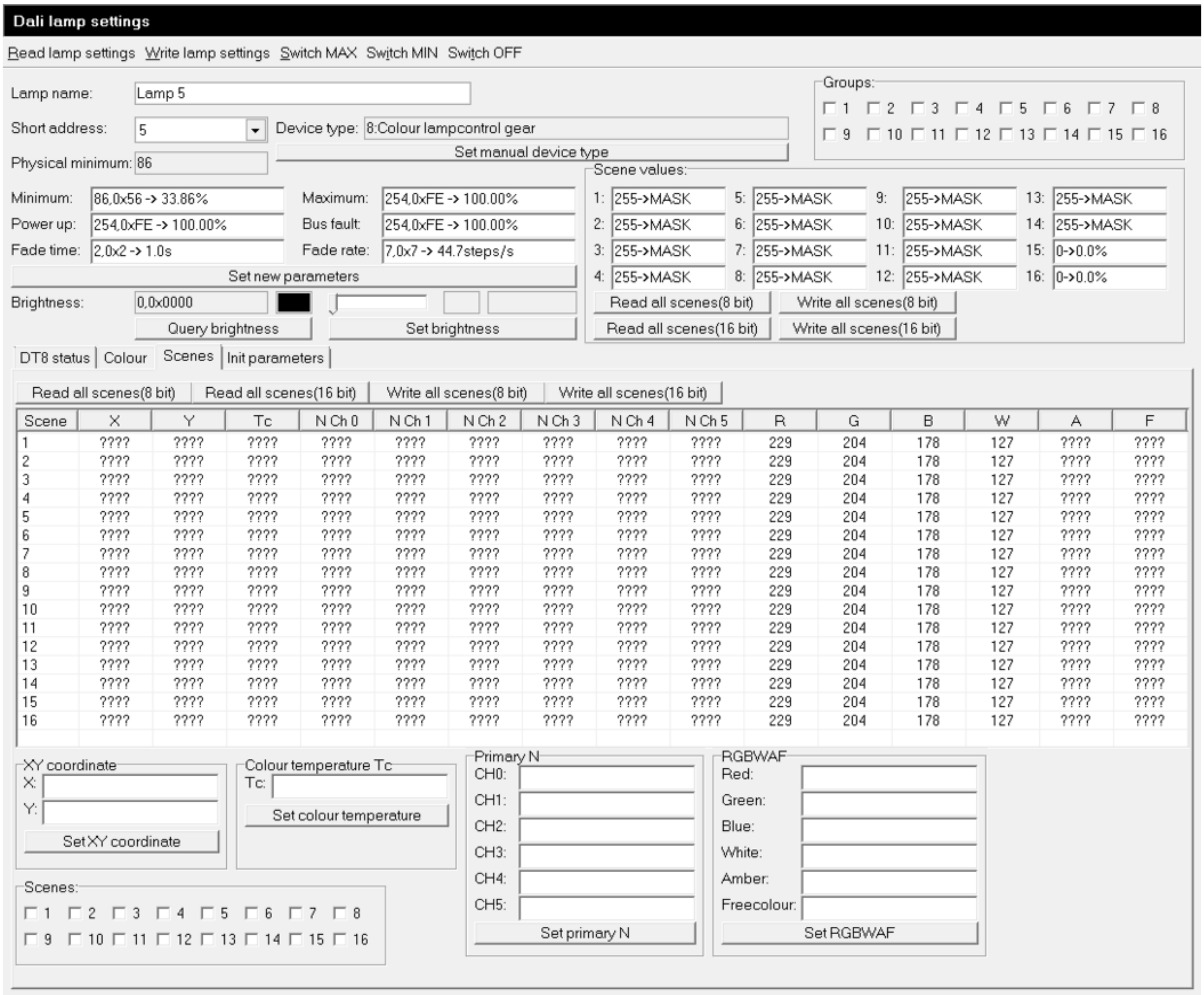
- **XY:** Set the selected XY coordinate according to the CIE color space chromaticity diagram (1931).

  HINT: xy coordinate mode uses 16 bit data mode. So first read out the current colors with the button Read colors (16 bit mode). Then set the new xy coordinate for the lamp with this button or use X-,X+,Y- or Y+ command buttons.

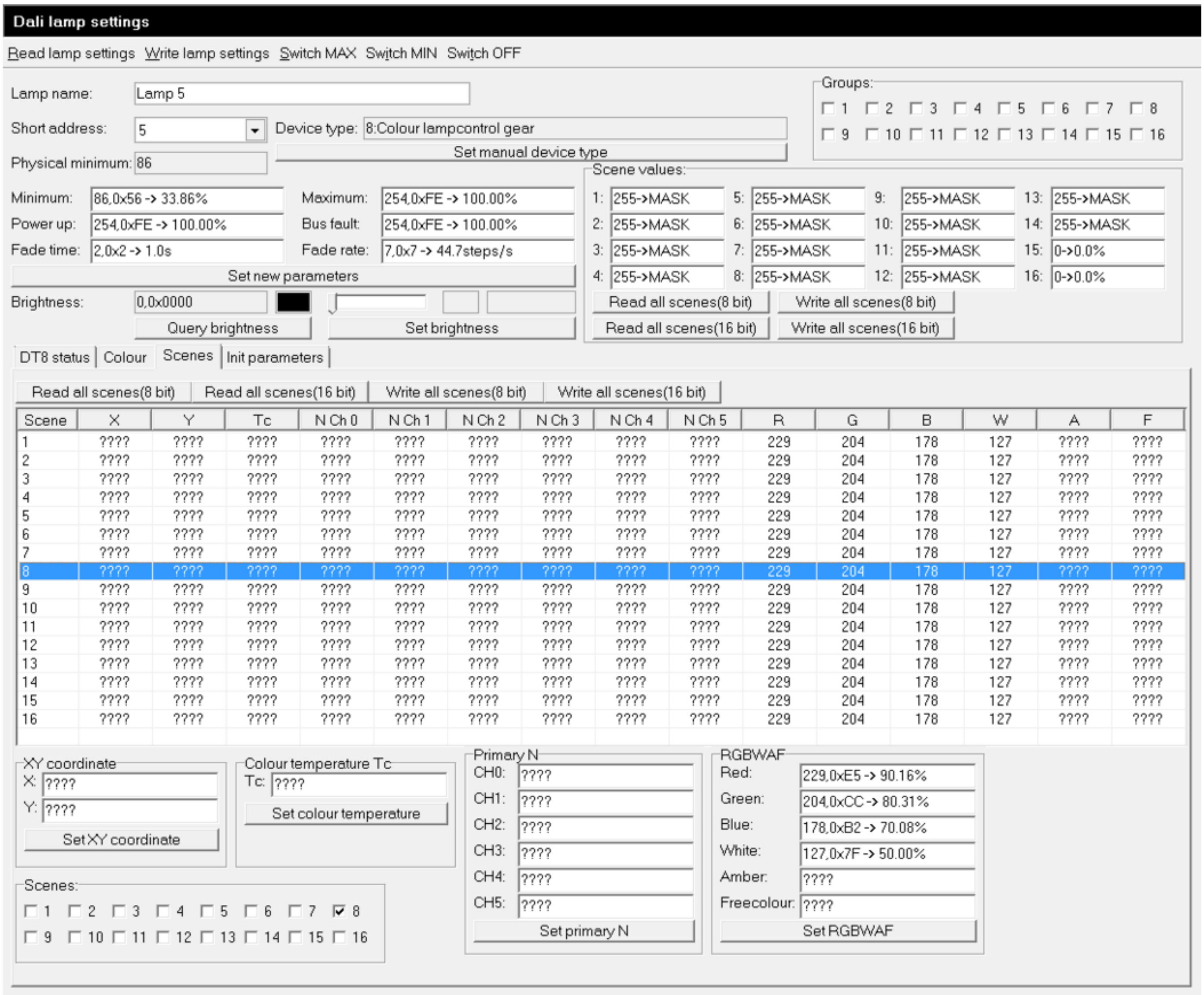
- **X-:** Decrement X coordinate with one step
- **X+:** Increment X coordinate with one step
- **Y-:** Decrement Y coordinate with one step
- **Y+:** Increment Y coordinate with one step



On the tab Scenes, you can read and write all DALI color scenes for a DALI device type 8 lamp. Choose the button "Read all scenes(8 bit)" for 8 bit data mode readout or the button "Read all scenes(16 bit)" for 16 bit data mode readout of all scene values of the selected lamp. You will see for a RGBW device the following sample screen:



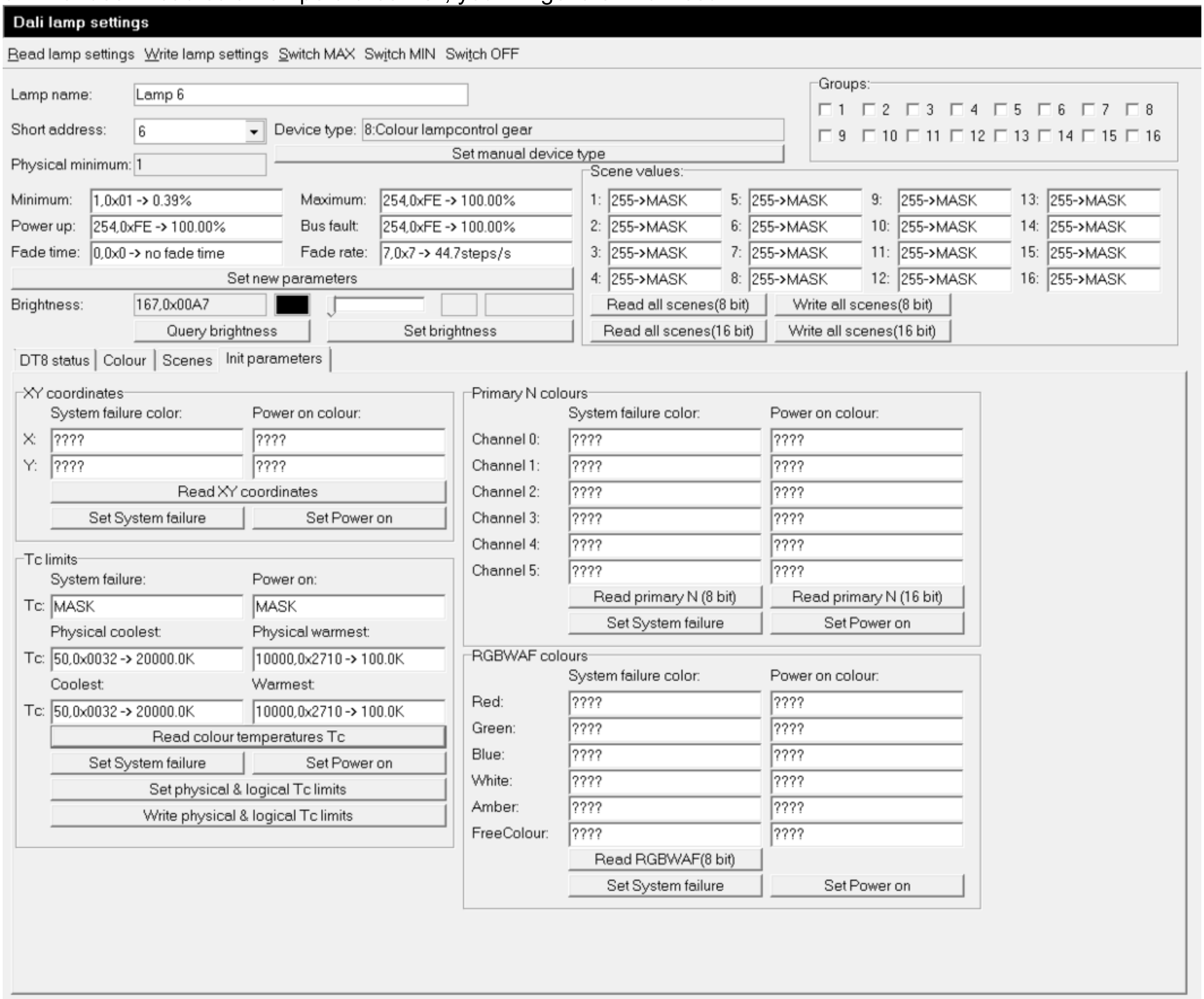
Now select a scene form the list, the current scene parameters will be shown in the dialog below the 16 scenes. It could look like this:



Now modify the scene values by entering new values into the selected fields. Then choose other scenes by checking the appropriate check boxes if you want to modify more than one scene at a time. Then click onto "Set RGBWAF" or the other set buttons to modify the selected scenes in the PC software.

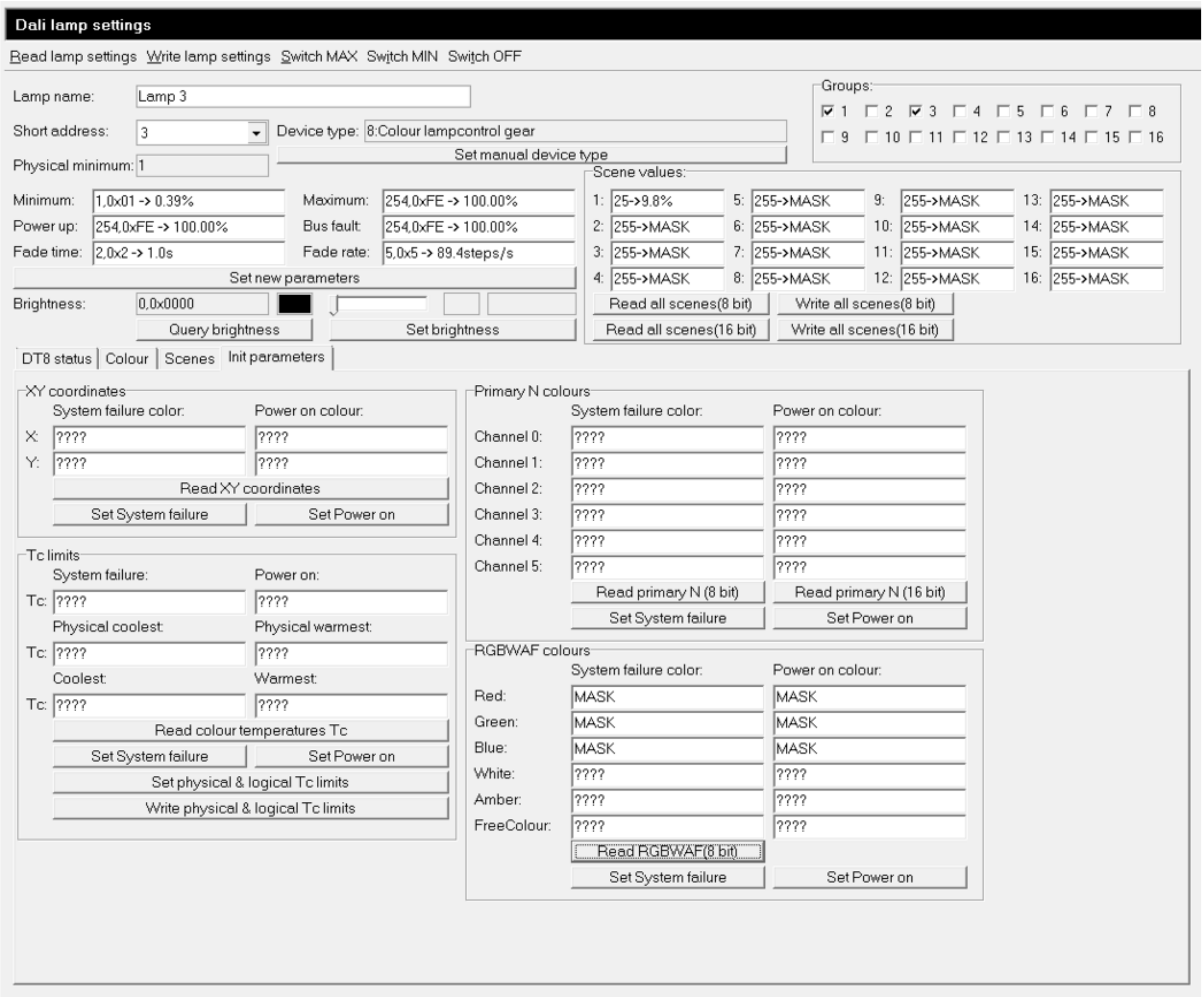
Last but not least you have to download the scenes into your lamp by using the buttons "Write all scenes(8 bit)" or "Write all scenes(16 bit).

On the tab Init parameters, you can read and write all DALI color settings for initial or error states of the DALI device type 8 lamp. Choose the button "Read ..." for the appropriate color model your device will use. e.g. for CW-WW dimmer use "Read color temperatures TC", you will get a similar result:



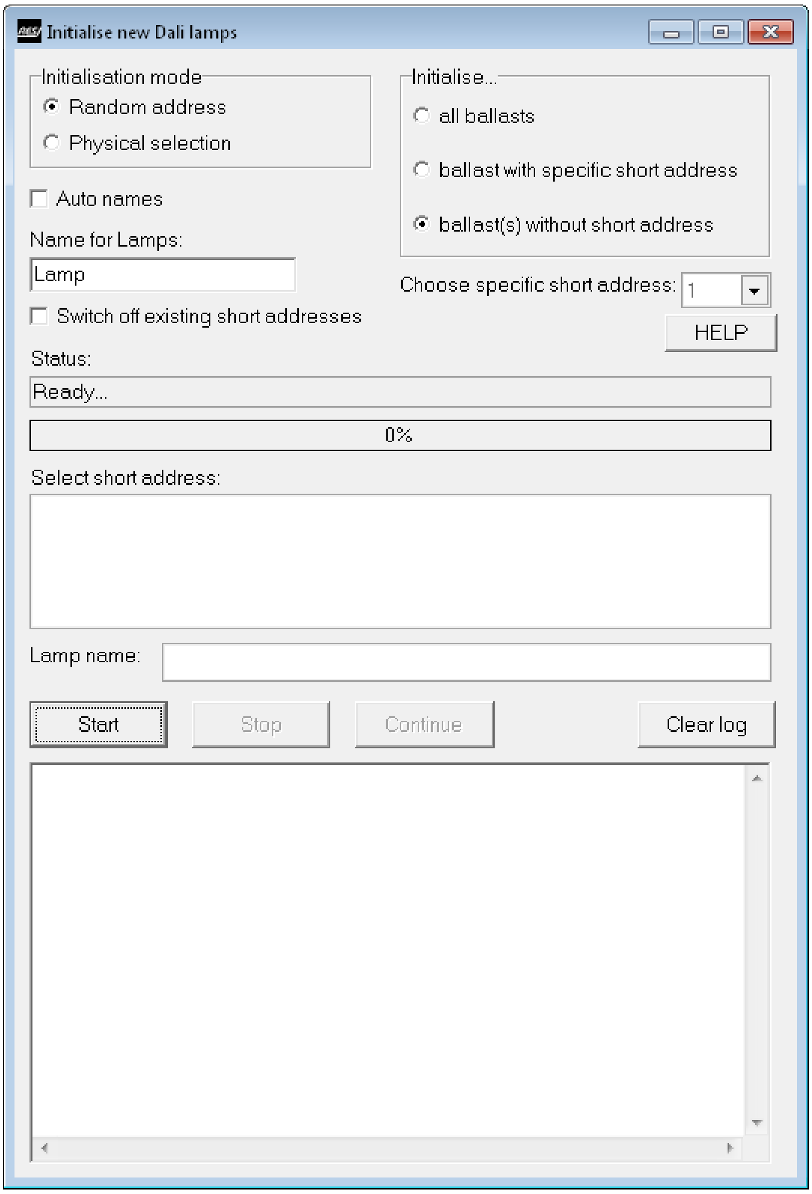
Now you can modify the parameters and limits. Enter new values and select the buttons "Set SystemFailure", "Set Power on" and/or "Set physical & logical Tc limit" to edit the fields. When you are finished, you have to download the new settings into the DALI lamp with the button "Write lamp settings". Only the physical and logical limits of this lamp are not written by this general write command. Therefore you have to use the command "Write physical & logical Tc limits" to download your new settings into your lamp.

When using a RGWAF dimmer, you can read out the current settings with the button "Read RGBWAF(8 bit) and modify the System failure color or the power on color and the use the buttons "Set System Failure" or "Set Power on" to save the values on the PC. When you download this settings with the button "Write lamp settings" you will store the values in the lamp.


Dali lamp settings
Read lamp settings Write lamp settings Switch MAX Switch MIN Switch OFF
Lamp name: Lamp 3
Short address: 3
Device type: 8:Colour lampcontrol gear
Set manual device type
Physical minimum: 1
Groups:
1 2 3 4 5 6 7 8
9 10 11 12 13 14 15 16
Scene values:
1: 25->9.8%
2: 255->MASK
3: 255->MASK
4: 255->MASK
5: 255->MASK
6: 255->MASK
7: 255->MASK
8: 255->MASK
9: 255->MASK
10: 255->MASK
11: 255->MASK
12: 255->MASK
13: 255->MASK
14: 255->MASK
15: 255->MASK
16: 255->MASK
Read all scenes(8 bit)
Write all scenes(8 bit)
Read all scenes(16 bit)
Write all scenes(16 bit)
Minimum: 1,0x01 -> 0.39%
Maximum: 254,0xFE -> 100.00%
Power up: 254,0xFE -> 100.00%
Bus fault: 254,0xFE -> 100.00%
Fade time: 2,0x2 -> 1.0s
Fade rate: 5,0x5 -> 89.4steps/s
Set new parameters
Brightness: 0,0x0000
Query brightness
Set brightness
DT8 status
Colour
Scenes
Init parameters
XY coordinates
System failure color:
Power on colour:
X: ????
????
Y: ????
????
Read XY coordinates
Set System failure
Set Power on
Tc limits
System failure:
Power on:
Tc: ????
????
Physical coolest:
Physical warmest:
Tc: ????
????
Coolest:
Warmest:
Tc: ????
????
Read colour temperatures Tc
Set System failure
Set Power on
Set physical & logical Tc limits
Write physical & logical Tc limits
Primary N colours
System failure color:
Power on colour:
Channel 0: ????
????
Channel 1: ????
????
Channel 2: ????
????
Channel 3: ????
????
Channel 4: ????
????
Channel 5: ????
????
Read primary N (8 bit)
Read primary N (16 bit)
Set System failure
Set Power on
RGBWAF colours
System failure color:
Power on colour:
Red: MASK
MASK
Green: MASK
MASK
Blue: MASK
MASK
White: ????
????
Amber: ????
????
FreeColour: ????
????
Read RGBWAF(8 bit)
Set System failure
Set Power on

45.8.4 Initialize lamps

This function opens a window where you can select how you want to address the new DALI 1.0 lamps on the DALI bus:



You can select the following options:

- **Initialisation mode**: DALI supports two modes of initialization of new lamps:
  - **Random adddress:** In this mode the system searches automatically for new lamps on the DALI bus and address the new lamps with a unique short address between 1 and 64 (0 to 63 in the DALI commands).
  - **Physical selection**: Older DALI lamps support an additional addressing mode, where you can address the lamp by disconnecting the bulb in the lamp and then address the lamp with a unique short address between 1 and 64 (0 to 63 in the DALI commands).
- **Initialize…:** This selection offers three possible search modes for random addressing mode:
  - **all ballasts**: All ballast will be readdressed with new short addresses. All stored short addresses are deleted before the search will start. This means a complete reinitialisation of the DALI network.
  - **ballast with specific short address:** You can choose a specific short address from the drop down list, before you start search to readdress this ballast with a new short address. This mode is supported by the DALI standard but makes in real life not much sense.
  - **ballast(s) without short address:** This mode is the most common used mode to address only the new connected ballasts without a short address on the DALI bus.

- **Auto names:** This option allows you a complete automatic search process. If this check box is checked, the software will search for new ballasts and give each found ballast a unique name consisting out of the prefix entered in the text filed Name for lamp and the assigned short address from 1 to 64.

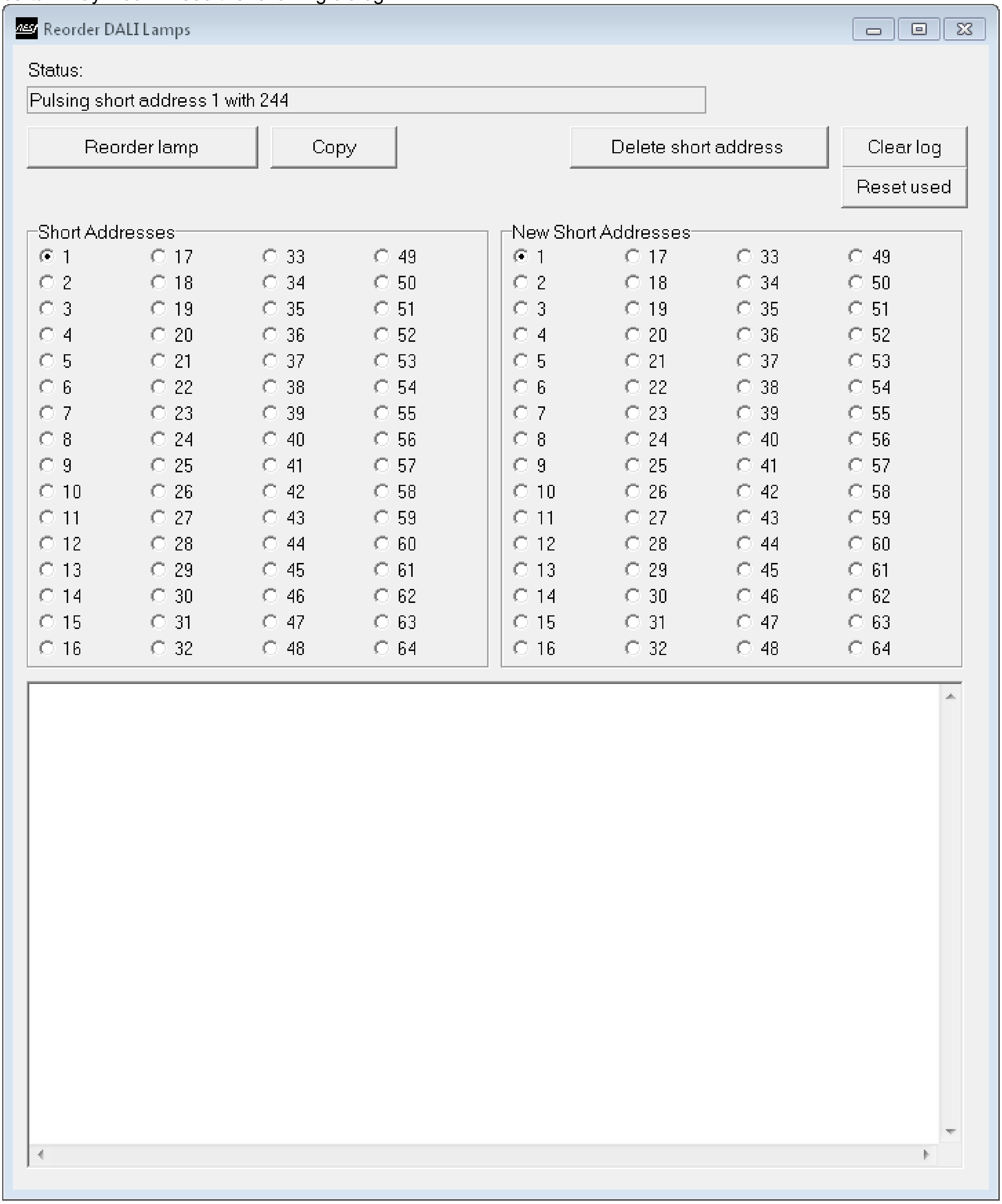
- **Switch off existing short addresses:** To visualize only the new lamps when searching, you can use this checkbox, then all existing ballasts with short addresses are switched off before the search for new lamps will began.

- **Start, Stop, Continue Buttons:** While the search process will run, use this buttons to navigate through the addressing search process.

This process will add automatically all found DALI lamps to the project tree!

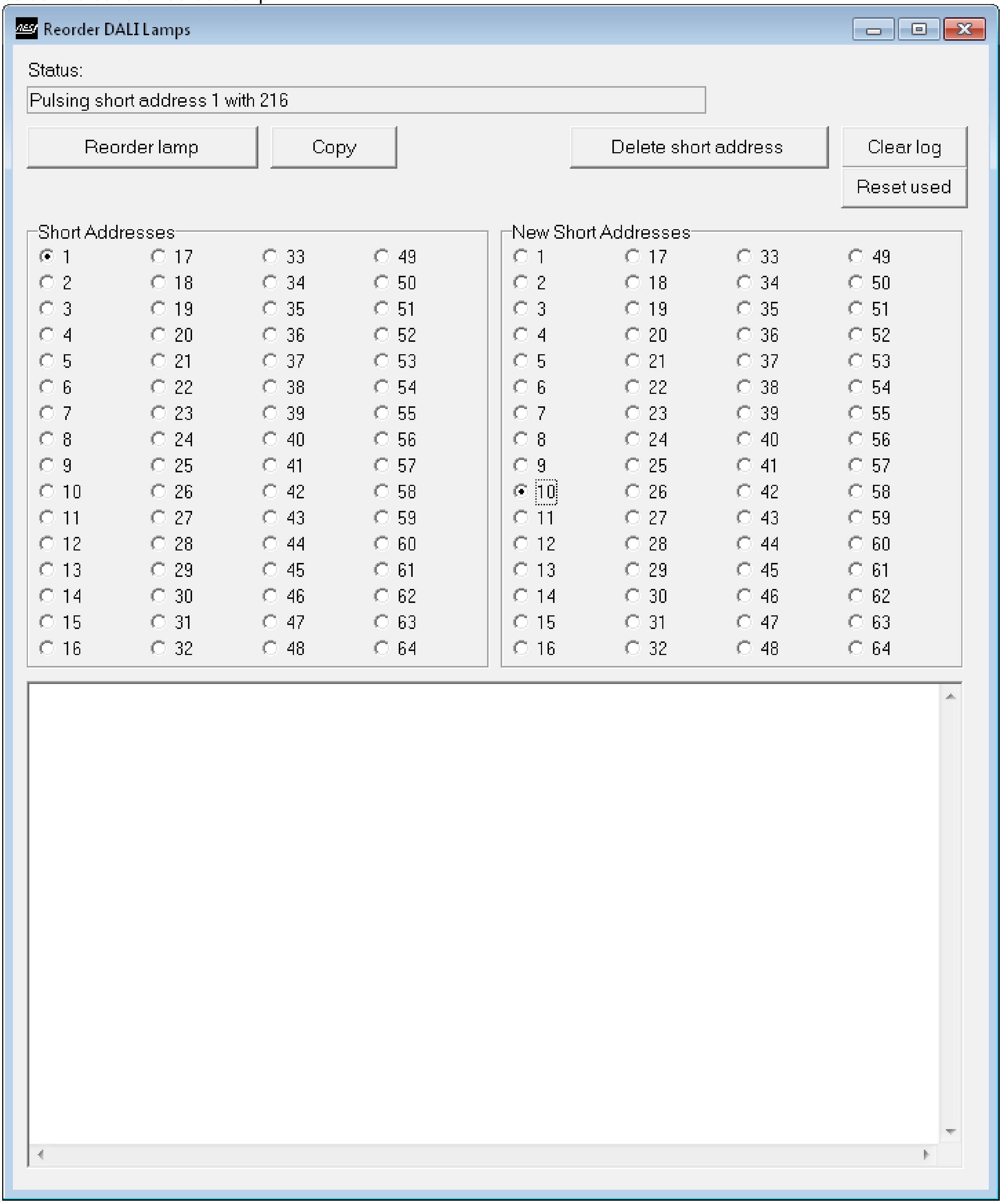
45.8.5 Reorder DALI lamps

This function opens a window where you can easily change the short address of DALI lamps to order your lamps in a certain way. You will see the following dialog:

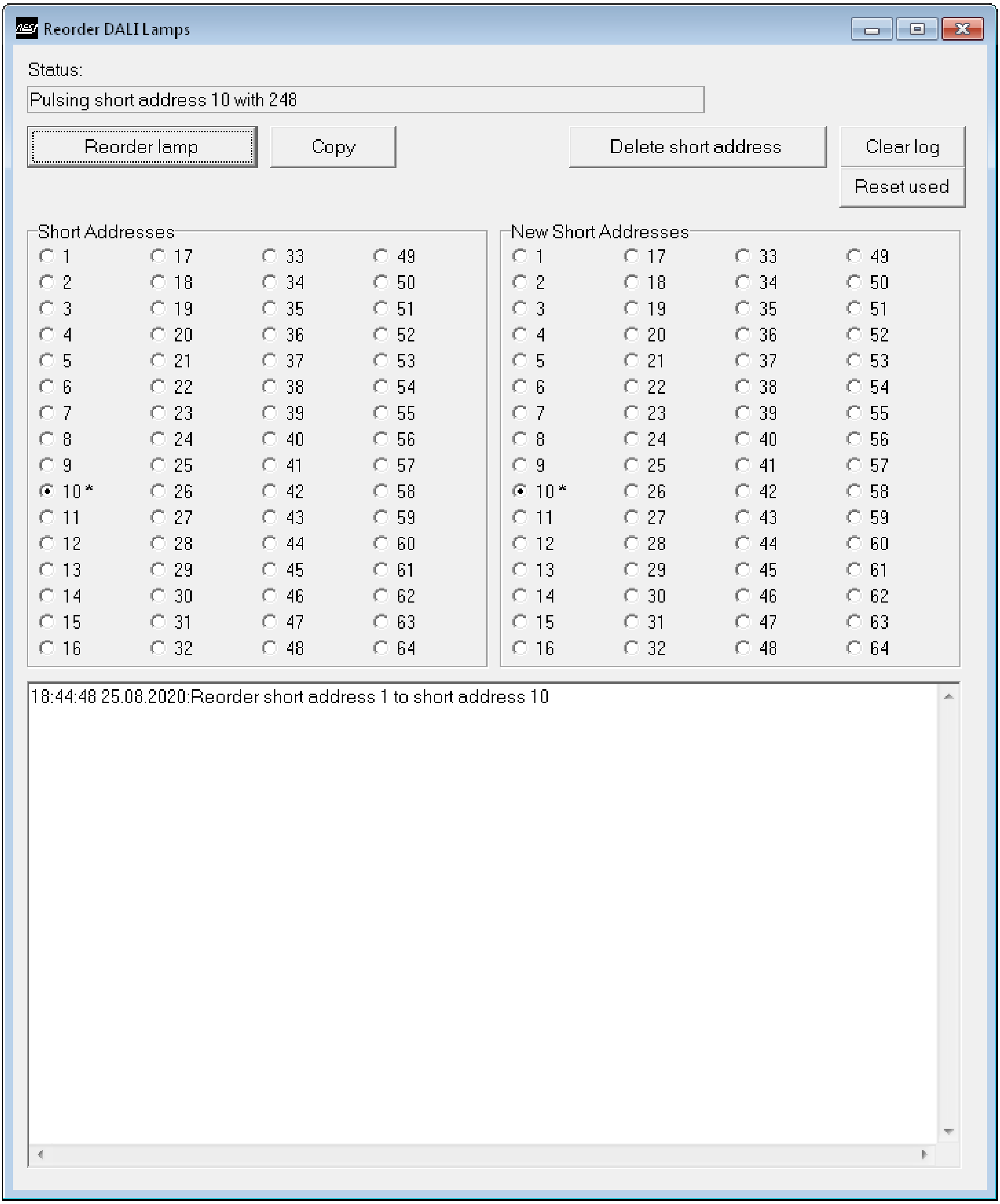


You will notice that after opening this dialog, the DALI lamp with the short address 1 (out of 64) will flash. This will help you to find the lamp in the field. As soon as you select a new short address from the panel Short addresses, the new selected DALI lamp will flash and all other DALI lamps are off.

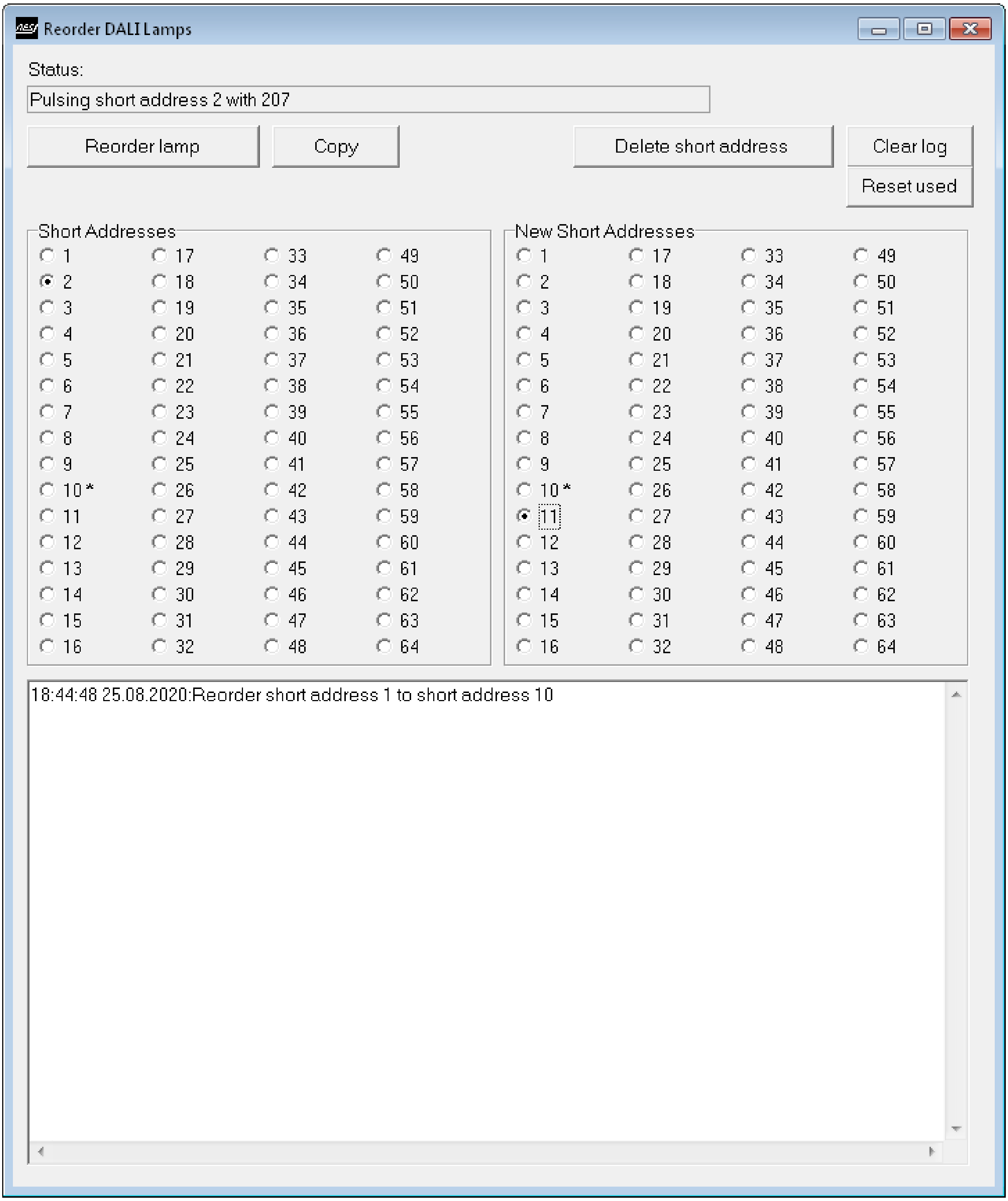
In our sample we want to readdress the DALI lamps 1 and 2 to the short addresses 10 and 11. So we select 1 and see that the correct lamp will flash. No we select in the right panel New Short addresses the address 10 and then we select the button Reorder lamp:



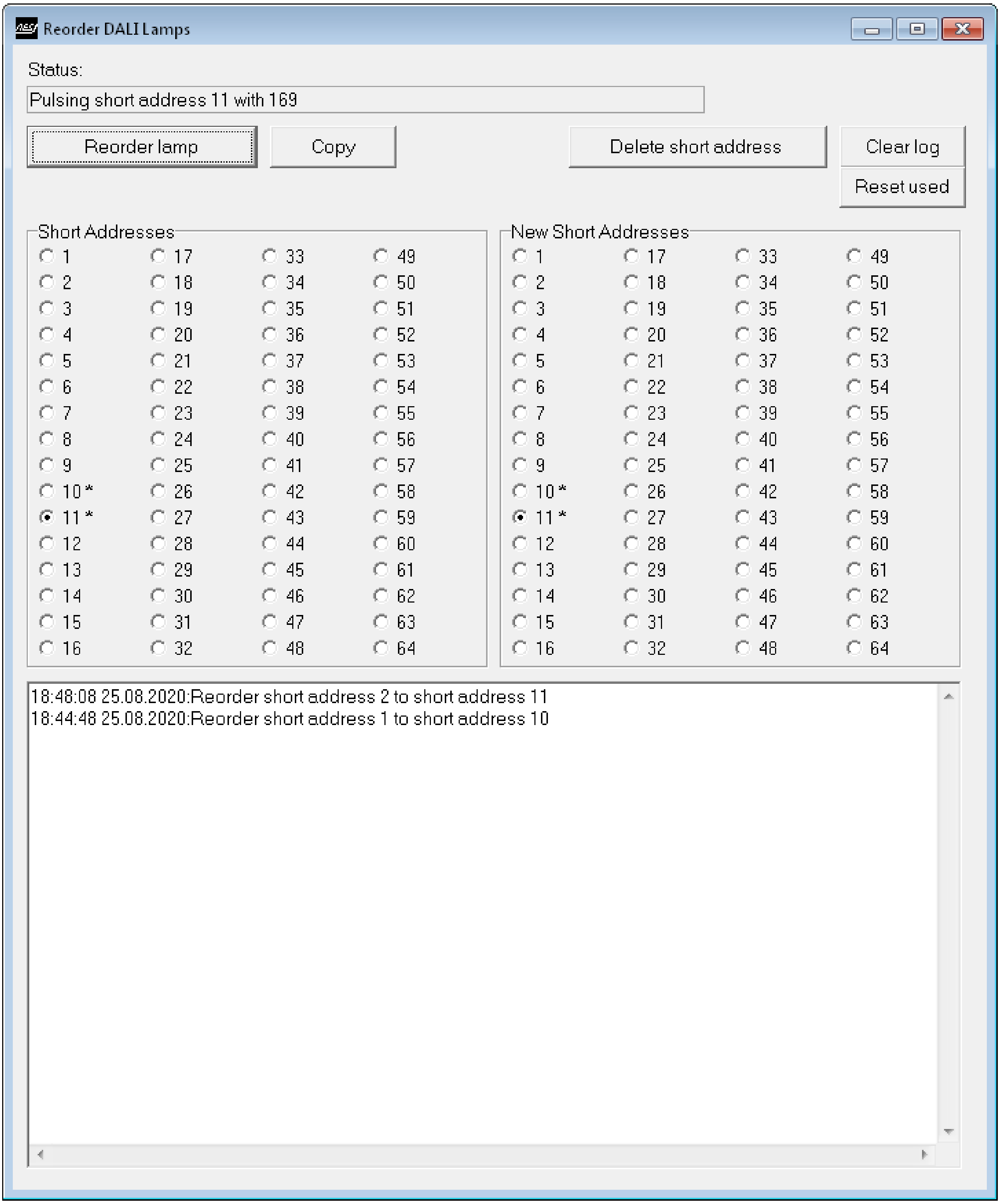
The system will show the following dialog:



As you will notice, now the old DALI lamp 1 has the new short address 10 and will flash. To indicate, that we have used the short address 10, there is an asterix behind the short address in both panels. Now we select the DALI short address 2 in the left panel. The second DALI lamp will flash. In the right panel we select the new DALI short address 11. Now we click again on the button Reorder lamp.



If successful you will see the following result:

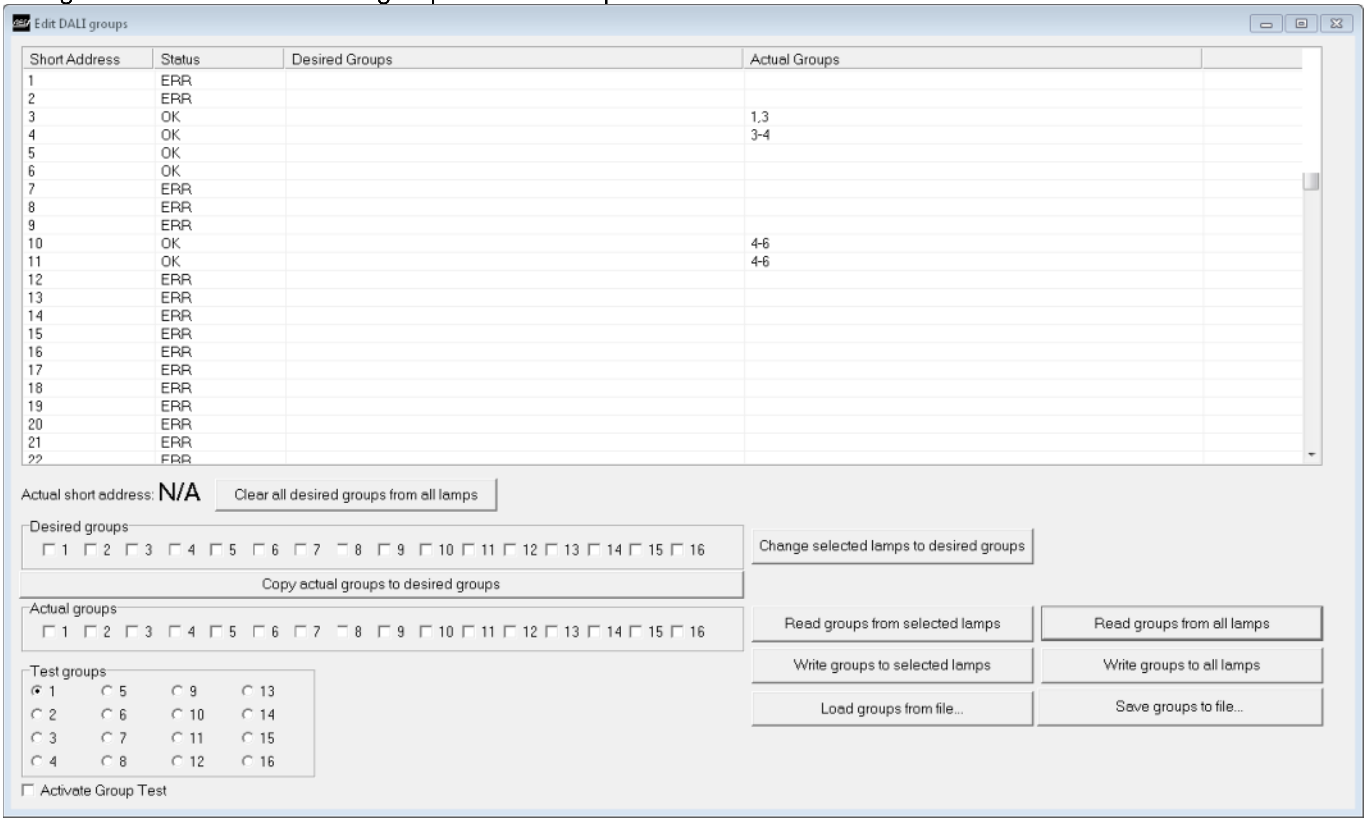


Again now in the left panel the new short address 11 is selected and the DALI lamp flashes. In both panels now the addresses 10 and 11 are marked with an asterix to indicate, that we have used them already. You can clear this signs with the button "Reset used". In the log you see the last actions.

The button Copy will select the same short address in the right panel as you have selected in the left panel.

45.8.6 Edit groups

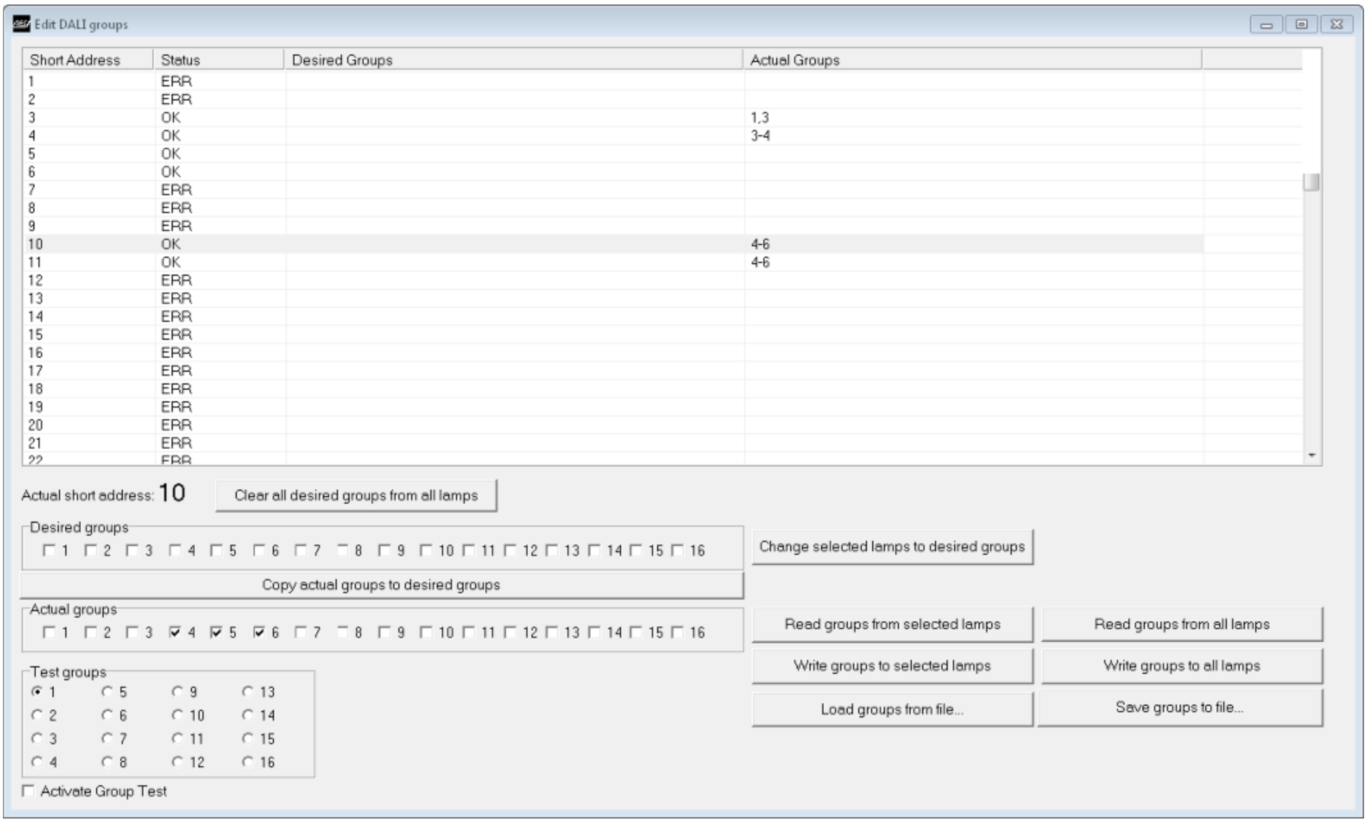
This function opens a window where you can easily change the groups of of all DALI lamps. You will see the following dialog. Select the button "Read groups from all lamps". You will see a similar result:



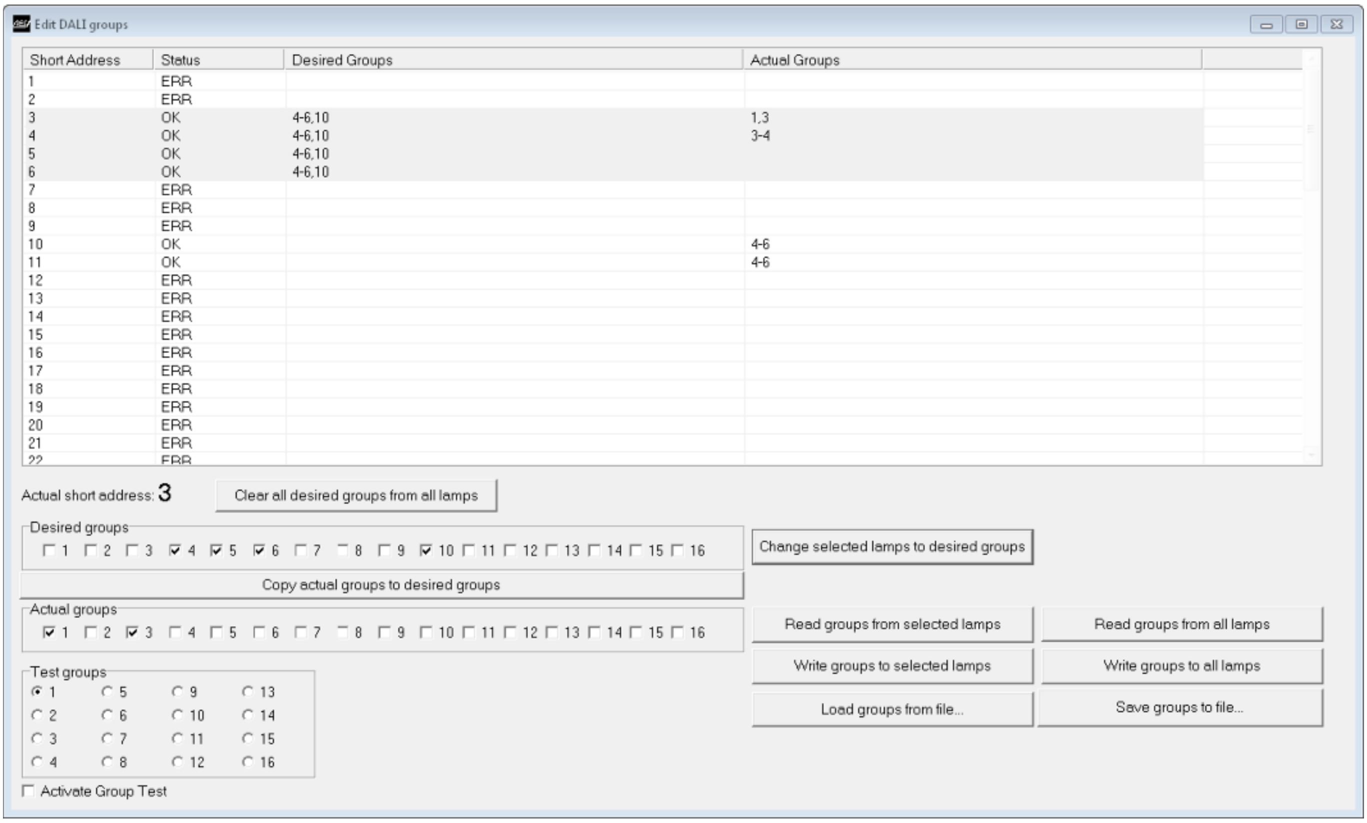
In the List you will see the following information in the columns:

- **Short Address:** The DALI short address from 1 to 64
- **Status**: The status of the DALI lamp since the last readout of group data from this lamp. OK means the readout was ok, ERR means that the short address is not available (no lamp connected with this DALI short address). Unknown means, that no readout has been done.
- **Desired Groups**: This is the list of desired groups for each short address. You can prepare a list of groups for all short addresses, and the download this list to all DALI lamps with one command. But you can also modify only one DALI lamp or a DALI lamp group.
- **Actual Groups:** This are the group settings which are currently stored in the DALI lamp.

As soon as you click on one line in this list, the display will show the actual group configuration below this list for this short address:

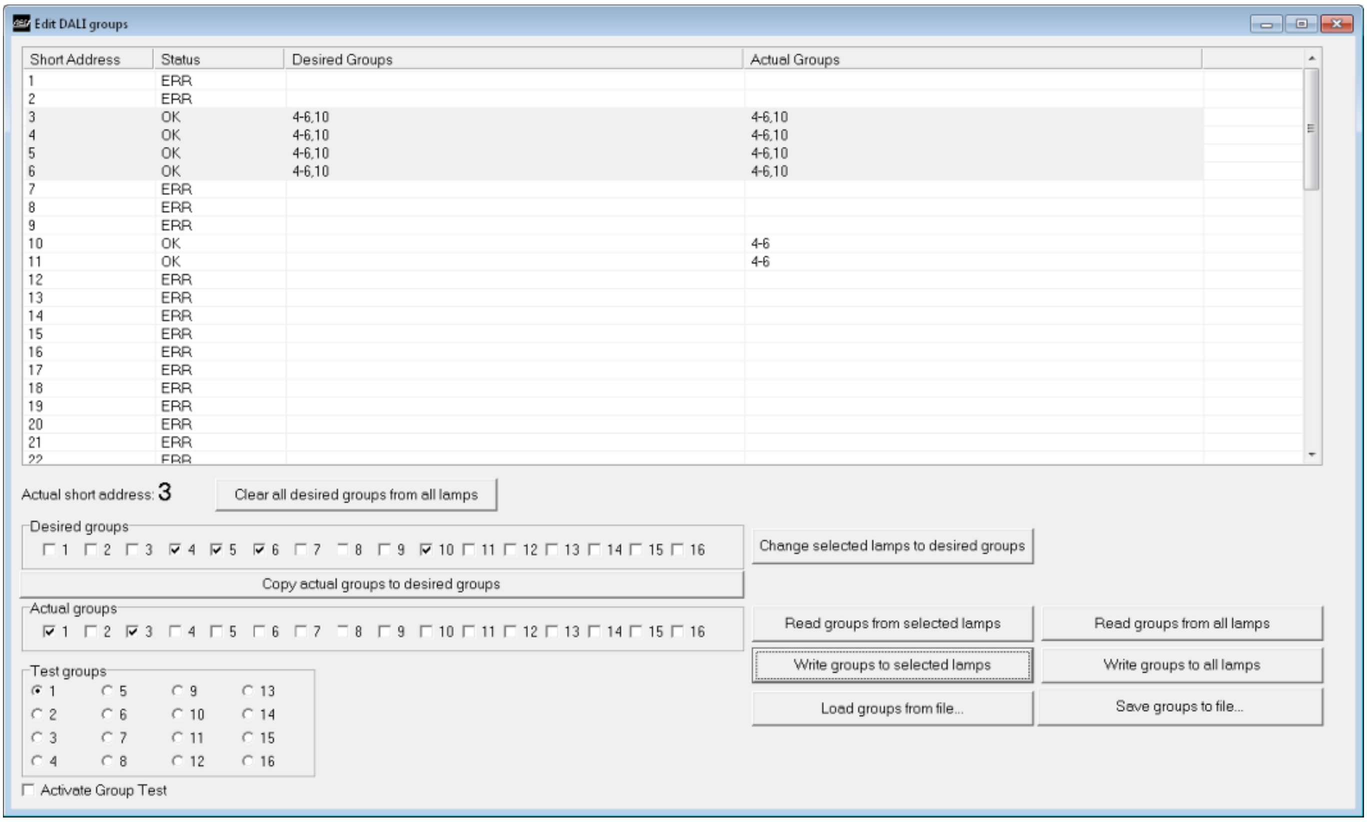


Now you can copy the actual groups to the desired groups section by using the button "Copy actual groups to desired groups". You will notice, that now the same groups are selected in the panel Desired groups. No we manually check the group 10. After that we multi select the lamps 3 to 6 in the list. Now we press the button "Change selected lamps to desired groups". You will see the following result:



As long as the groups are not identical in the columns "Desired Groups" and "Actual Groups", you have to download the group configuration into the lamps. For that you can use the buttons "Write groups to selected lamps", which will modify only the selected lamps from the list or you use the button "Write groups to all lamps" to modify all 64 DALI lamps with the new group information.

HINT: If the field Desired groups is empty no download will be done for the selected lamp.



With the button "Read groups from selected lamps" you can update the list with new status information from the selected lamps.

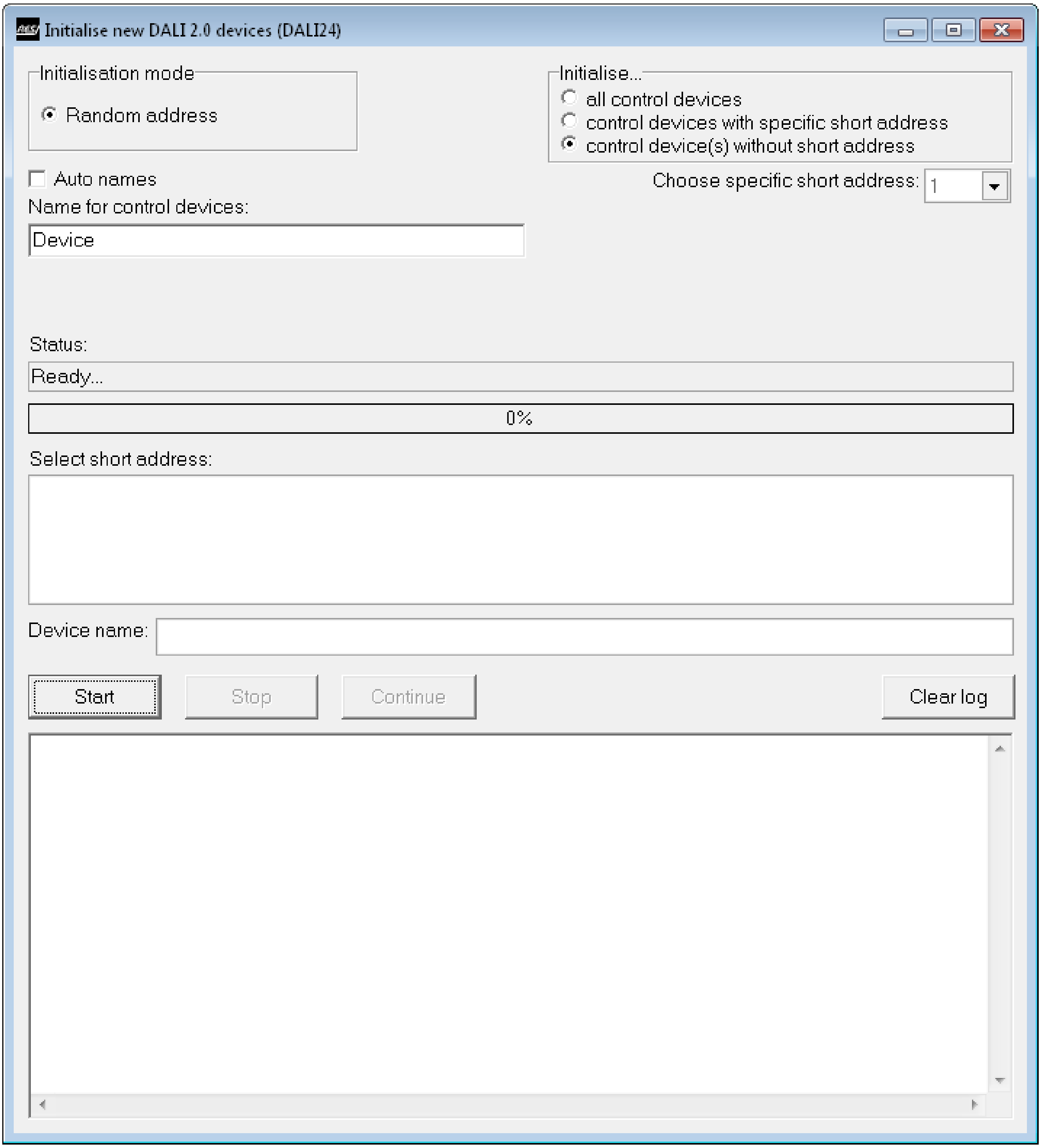
The buttons "Load groups from file…" and "Save groups to file…" you cane save your group setup for all 64 DALI lamps for further use in a similar DALI system.

The button "Clear all desired groups from all lamps" will clear all desired groups from all 64 DALI lamps to start with an empty group information in the list.

For easy testing of the new group setup you can activate the checkbox "Activate Group Test" and select a group from 1 to 16 (in the DALI command from 0 to 15). All DALI lamps which are part of this group will flash. Select from the panel Test groups other groups to check if your configuration meets your requirements.

45.8.7 Initialize devices

This function opens a window where you can search and address DALI24 (DALI 2.0) control devices. It is similar to the search for DALI 1.0 lamps, but it uses 24 bit DALI frames and special DALI 2.0 commands to address the DALI 2.0 control devices like occupancy sensors or push button devices.



You can select the following options:

- **Initialisation mode**: DALI supports two modes of initialization of new devices:
  - **Random adddress:** In this mode the system searches automatically for new devices on the DALI bus and address the new devices with a unique short address between 1 and 64 (0 to 63 in the DALI commands). This is NOT the same short address, which a DALI ballast will use. So in fact, you can combine 64 DALI ballasts and 64 DALI control devices on a DALI line.

- **Initialize…:** This selection offers three possible search modes for random addressing mode:
  - **all control devices**: All control devices will be readdressed with new short addresses. All stored short addresses are deleted before the search will start. This means a complete reinitialisation of the 24 bit DALI network.
  - **control devices with specific short address:** You can choose a specific short address from the drop down list, before you start search to readdress this control device with a new short address. This mode is supported by the DALI standard but makes in real life not much sense.
  - **control device(s) without short address:** This mode is the most common used mode to address only the new connected control devices without a short address on the DALI bus.

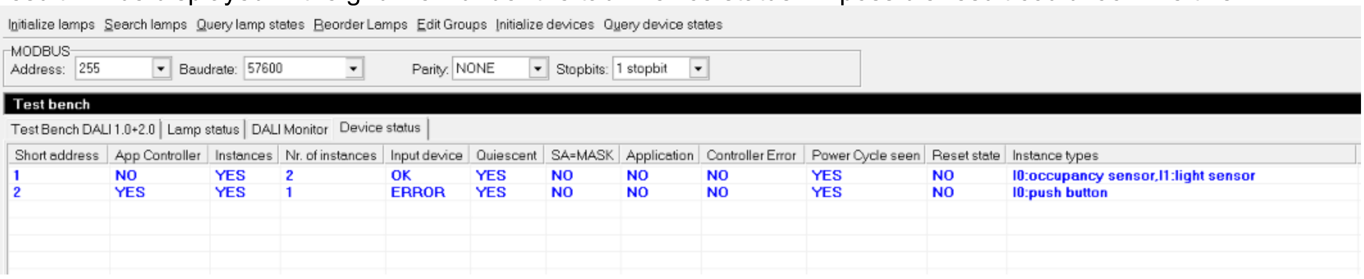
- **Auto names:** This option allows you a complete automatic search process. If this check box is checked, the software will search for new control devices and give each found devices a unique name consisting out of the prefix entered in the text filed Name for control devices and the assigned short address from 1 to 64.

- **Start, Stop, Continue Buttons:** While the search process will run, use this buttons to navigate through the addressing search process.

This process will add automatically all found DALI control devices to the project tree!

45.8.8 Query device states

This function tries to read out basic data of all DALI 2.0 (DALI24) control devices from the connected DALI bus. The result will be displayed in the grid view under the tab "Device status". A possible result could look like this:



Short address	App Controller	Instances	Nr. of instances	Input device	Quiescent	SA=MASK	Application	Controller Error	Power Cycle seen	Reset state	Instance types
1	NO	YES	2	OK	YES	NO	NO	NO	YES	NO	I0:occupancy sensor,I1:light sensor
2	YES	YES	1	ERROR	YES	NO	NO	NO	YES	NO	I0:push button

You will find the following information in this grid view:

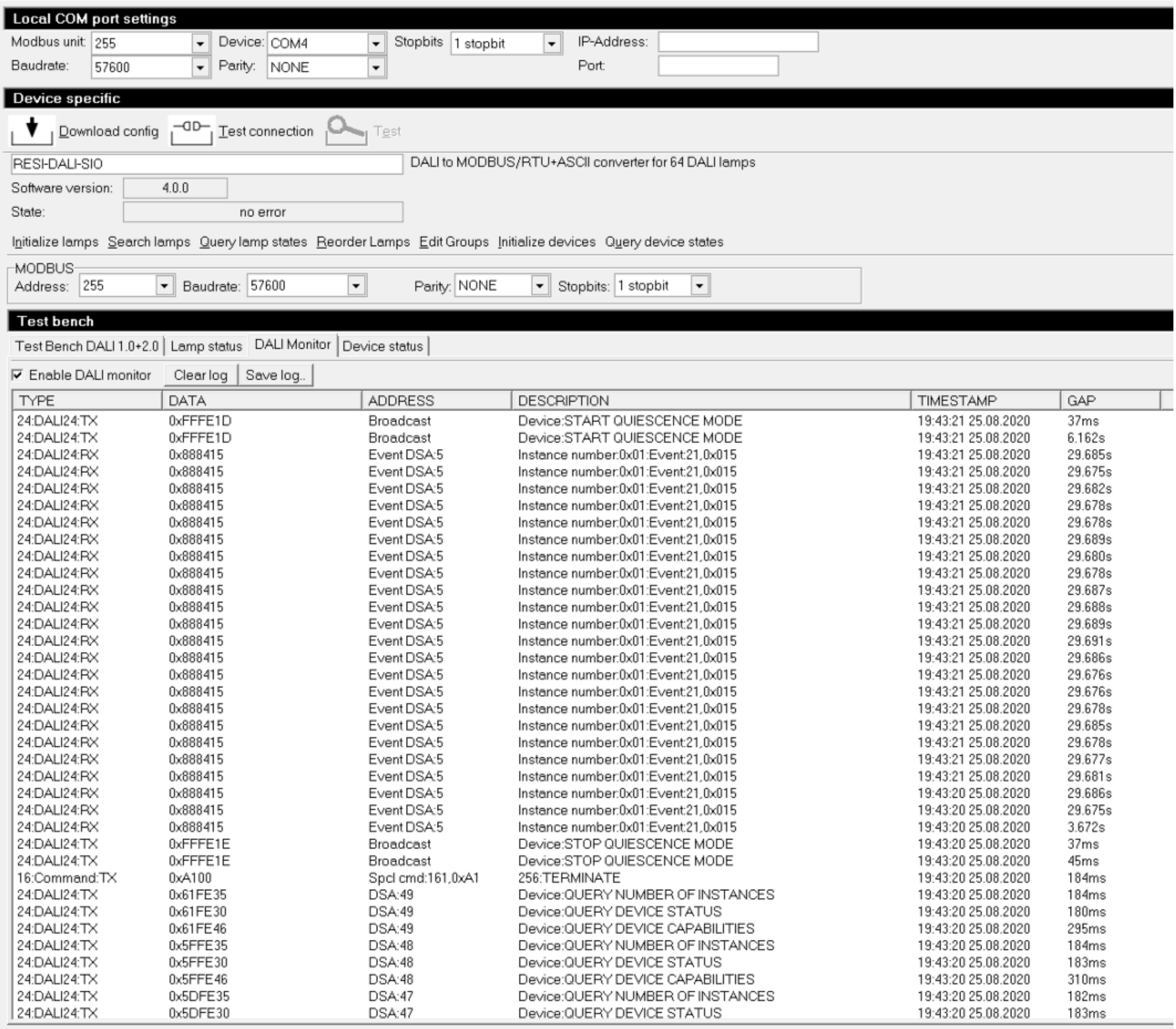
- **Short address**: The current short address of the tested control device
- **App controller**: The current state of the internal application controller of the device. YES indicates, that the application controller is running, NO indicates that the application controller is stopped.
- **Instances**: If the devices supports instances, this flag is YES, otherwise NO.
- **Nr. of instances:** Maximum number of instances of the device
- **Input device**: If the device supports input and everything is ok, this flag is OK, otherwise ERROR
- **Quiescent**: if the device is currently in quiescent mode (No events are generated by the devices) this flag is YES, otherwise NO.
- **SA=MASK**: This flag is YES if the short address is 255, 0xFFF or MASK
- **Application**: This flag (Yes or No) indicates if the application is active or not
- **Controller error**: This flag (Yes or No) indicates if the application controller has an error or not
- **Power cycle seen**: This flag (Yes or No) indicates if the device has seen a power cycle since last DALI power on
- **Reset state**: This flag (Yes or No) indicates if the device is in reset state or not
- **Instance types**: this list shows the name of each instance of the device

45.8.9 DALI Monitor

We have implemented a DALI monitor feature in the software. Our RESI-DALI-SIO or RESI-DALI-ETH gateway can save up to 128 incoming or outgoing DALI frames in an internal FIFO. This FIFO is cyclically readout by this function and the DALI frames are decoded for easier understanding. If you activate this function you can use the rest of the software normal, so you can easily monitor all your activities. But don't forget to close the dialog by unchecking the checkbox Enable DALI monitor, before you change to another device in the project tree.

This feature is for better understanding, what happens on a DALI bus and for better error search in case of problems. You can save the log file in a text file and mail it to our support if you have a suspicious behavior on the DALI bus.

Also you can use our gateway to sniff a foreign DALI bus, what happens on the BUS or what special DALI events are sent by control devices.



TYPE	DATA	ADDRESS	DESCRIPTION	TIMESTAMP	GAP
24:DALI24:TX	0xFFFE1D	Broadcast	Device:START QUIESCENCE MODE	19:43:21 25.08.2020	37ms
24:DALI24:TX	0xFFFE1D	Broadcast	Device:START QUIESCENCE MODE	19:43:21 25.08.2020	6.162s
24:DALI24:RX	0x888415	Event DSA:5	Instance number:0x01:Event:21,0x015	19:43:21 25.08.2020	29.685s
24:DALI24:RX	0x888415	Event DSA:5	Instance number:0x01:Event:21,0x015	19:43:21 25.08.2020	29.675s
24:DALI24:RX	0x888415	Event DSA:5	Instance number:0x01:Event:21,0x015	19:43:21 25.08.2020	29.682s
24:DALI24:RX	0x888415	Event DSA:5	Instance number:0x01:Event:21,0x015	19:43:21 25.08.2020	29.678s
24:DALI24:RX	0x888415	Event DSA:5	Instance number:0x01:Event:21,0x015	19:43:21 25.08.2020	29.678s
24:DALI24:RX	0x888415	Event DSA:5	Instance number:0x01:Event:21,0x015	19:43:21 25.08.2020	29.689s
24:DALI24:RX	0x888415	Event DSA:5	Instance number:0x01:Event:21,0x015	19:43:21 25.08.2020	29.680s
24:DALI24:RX	0x888415	Event DSA:5	Instance number:0x01:Event:21,0x015	19:43:21 25.08.2020	29.678s
24:DALI24:RX	0x888415	Event DSA:5	Instance number:0x01:Event:21,0x015	19:43:21 25.08.2020	29.687s
24:DALI24:RX	0x888415	Event DSA:5	Instance number:0x01:Event:21,0x015	19:43:21 25.08.2020	29.688s
24:DALI24:RX	0x888415	Event DSA:5	Instance number:0x01:Event:21,0x015	19:43:21 25.08.2020	29.689s
24:DALI24:RX	0x888415	Event DSA:5	Instance number:0x01:Event:21,0x015	19:43:21 25.08.2020	29.691s
24:DALI24:RX	0x888415	Event DSA:5	Instance number:0x01:Event:21,0x015	19:43:21 25.08.2020	29.686s
24:DALI24:RX	0x888415	Event DSA:5	Instance number:0x01:Event:21,0x015	19:43:21 25.08.2020	29.676s
24:DALI24:RX	0x888415	Event DSA:5	Instance number:0x01:Event:21,0x015	19:43:21 25.08.2020	29.676s
24:DALI24:RX	0x888415	Event DSA:5	Instance number:0x01:Event:21,0x015	19:43:21 25.08.2020	29.678s
24:DALI24:RX	0x888415	Event DSA:5	Instance number:0x01:Event:21,0x015	19:43:21 25.08.2020	29.685s
24:DALI24:RX	0x888415	Event DSA:5	Instance number:0x01:Event:21,0x015	19:43:21 25.08.2020	29.678s
24:DALI24:RX	0x888415	Event DSA:5	Instance number:0x01:Event:21,0x015	19:43:21 25.08.2020	29.677s
24:DALI24:RX	0x888415	Event DSA:5	Instance number:0x01:Event:21,0x015	19:43:21 25.08.2020	29.681s
24:DALI24:RX	0x888415	Event DSA:5	Instance number:0x01:Event:21,0x015	19:43:20 25.08.2020	29.686s
24:DALI24:RX	0x888415	Event DSA:5	Instance number:0x01:Event:21,0x015	19:43:20 25.08.2020	29.675s
24:DALI24:RX	0x888415	Event DSA:5	Instance number:0x01:Event:21,0x015	19:43:20 25.08.2020	3.672s
24:DALI24:TX	0xFFFE1E	Broadcast	Device:STOP QUIESCENCE MODE	19:43:20 25.08.2020	37ms
24:DALI24:TX	0xFFFE1E	Broadcast	Device:STOP QUIESCENCE MODE	19:43:20 25.08.2020	45ms
16:Command:TX	0xA100	Spcl cmd:161,0xA1	256:TERMINATE	19:43:20 25.08.2020	184ms
24:DALI24:TX	0x61FE35	DSA:49	Device:QUERY NUMBER OF INSTANCES	19:43:20 25.08.2020	184ms
24:DALI24:TX	0x61FE30	DSA:49	Device:QUERY DEVICE STATUS	19:43:20 25.08.2020	180ms
24:DALI24:TX	0x61FE46	DSA:49	Device:QUERY DEVICE CAPABILITIES	19:43:20 25.08.2020	295ms
24:DALI24:TX	0x5FFE35	DSA:48	Device:QUERY NUMBER OF INSTANCES	19:43:20 25.08.2020	184ms
24:DALI24:TX	0x5FFE30	DSA:48	Device:QUERY DEVICE STATUS	19:43:20 25.08.2020	183ms
24:DALI24:TX	0x5FFE46	DSA:48	Device:QUERY DEVICE CAPABILITIES	19:43:20 25.08.2020	310ms
24:DALI24:TX	0x5DFE35	DSA:47	Device:QUERY NUMBER OF INSTANCES	19:43:20 25.08.2020	182ms
24:DALI24:TX	0x5DFE30	DSA:47	Device:QUERY DEVICE STATUS	19:43:20 25.08.2020	183ms

45.8.10

45.9 Additional MODBUS register & coils

Here you will find only the additional MODBUS registers and coils especially for this IO module. Please refer to the description of of the standard MODBUS mapping for more details about the available basic MODBUS registers and coils.

Please refer to the external document for detailed documentation of the current MODBUS register mapping for this IO module:

RESI-L-DALI-SIO-ETH-MODBUS+ASCII-ENxx.pdf

Don't forget, that there are some standard MODBUS registers for this device, which you cannot overwrite with your configuration!

45.10 Additional ASCII commands

Here you will find only the additional ASCII commands especially for this IO module. Please refer to the description of of the standard commands for more details about the available basic ASCII commands.

Please refer to the external document for detailed documentation of the current ASCII commands for this IO module:

RESI-L-DALI-SIO-ETH-MODBUS+ASCII-ENxx.pdf

45.11 HOWTO use the MODBUS registers

This chapter will show sample code, how to use the MODBUS registers with a host.
HINT: Our demo software is designed for a MODBUS master, which uses for the registers the index 0 to 65535, not the MODBUS syntax of 1 to 65536.

45.11.1 Set brightness of a lamp

To set the brightness of a lamp use this register:

LAMP LEVEL	3x00511 4x00511 I:510	0,0x0000 B:00 00	639	50,0	UINT16 R/W	NO
			0x027F			
			LAMP	2:LAMP SHORT ADDRESS 3		
Sets for a lamp short address a new brightness level between 0 and 254. 255 means MASK. Upper 8 bits:8-15:SHORTADRESS:number of the lamp to be controlled (0..63) Lower 8 bits: 0-7:VALUE:brightness value that should be send (0..254 for 0..100% and 255 for MASK)						

Sample code:

```
// ASio: In our development system the handle to the serial port
// AUnitID: The unit ID of the RESI-DALI-SIO module
sub DALI.Brightness.Lamp(num ASIO,int AUnitID,int ALamp,int ABrightness)
  int v
  bit r
  // High 8 Bit of holding register is lamp address between 0 and 63
  // Low 8 Bit of holding register is brightness value between 0 and 254, 255 stands for MASK
  v=(ALamp<<8)|ABrightness
  // REGISTER: LAMP LEVEL 3x00511 4x00511 I:510
  r=ModbusMaster.MbWriteSingleRegister(ASIO,AUnitID,510,v)
  if r==true then
    DALI.Info("SUCCESS")
  else
    DALI.Info("ERROR with MODBUS communication")
  endif
endsub
```

Sample code to use this function:

```
// Set lamp 1 to 100%
DALI.Brightness.Lamp(SIO,255,1,254)
// Set lamp 3 to 50%
DALI.Brightness.Lamp(SIO,255,3,128)
// Set lamp 63 to 0%
DALI.Brightness.Lamp(SIO,255,63,0)
```

45.11.2 Set brightness of a group of lamps

To set the brightness of a group of lamps use this register:

GROUP LEVEL	3x00521 4x00521 I:520	0,0x0000 B:00 00	383	50,0	UINT16 R/W	NO
			0x017F			
			GROUP	1:LAMP GROUP 2		
Sets for a lamp group a new brightness level between 0 and 254. 255 means MASK. Upper 8 bits:8-15:GROUP:number of the group to be controlled (0..15) Lower 8 bits:0-7:VALUE:brightness value that should be send (0..254 for 0..100% and 255 for MASK)						

Sample code:

```
// ASio: In our development system the handle to the serial port
// AUnitID: The unit ID of the RESI-DALI-SIO module
sub DALI.Brightness.Group(num ASIO,int AUnitID,int AGroup,int ABrightness)
  int v
  bit r
  // High 8 Bit of holding register is group number between 0 and 15
  // Low 8 Bit of holding register is brightness value between 0 and 254, 255 stands for MASK
  v=(AGroup<<8)|ABrightness
  // REGISTER: GROUP LEVEL 3x00521 4x00521 I:520
  r=ModbusMaster.MbWriteSingleRegister(ASIO,AUnitID,520,v)
  if r==true then
    DALI.Info("SUCCESS")
  else
    DALI.Info("ERROR with MODBUS communication")
  endif
endsub
```

Sample code to use this function:

```
// Set group 1 to 100%
DALI.Brightness.Group(SIO,255,1,254)
// Set group 5 to 50%
DALI.Brightness.Group(SIO,255,5,128)
// Set group 15 to 0%
DALI.Brightness.Group(SIO,255,15,0)
```

45.11.3 Set brightness for all lamps

To set the brightness for all lamps use this register:

ALL LEVEL	3x00531 4x00531 I:530	0,0x0000 B:00 00	127	50,0	UINT16 R/W	NO
			0x007F			
Sets for all lamps a new brightness level between 0 and 254. 255 means MASK. Upper 8 bits:8-15:Unused, always 0 Lower 8 bits:0-7:VALUE:brightness value that should be send (0..254 for 0..100% and 255 for MASK)						

Sample code:

```
// ASio: In our development system the handle to the serial port
// AUnitID: The unit ID of the RESI-DALI-SIO module
sub DALI.Brightness.All(num ASIO,int AUnitID,int ABrightness)
  int v
  bit r
  // High 8 Bit of holding register is 0
  // Low 8 Bit of holding register is brightness value between 0 and 254, 255 stands for MASK
  v=ABrightness
  // REGISTER: ALL LEVEL 3x00531 4x00531 I:530
  r=ModbusMaster.MbWriteSingleRegister(ASIO,AUnitID,530,v)
  if r==true then
    DALI.Info("SUCCESS")
  else
    DALI.Info("ERROR with MODBUS communication")
  endif
endsub
```

Sample code to use this function:

```
// Set all ballasts to 100%
DALI.Brightness.All(SIO,255,254)
// Set all ballasts to 50%
DALI.Brightness.All(SIO,255,128)
// Set all ballasts to 0%
DALI.Brightness.All(SIO,255,0)
```

45.11.4 Send DALI 1.0 command to lamp

To send a DALI 1.0 command to a lamp use this register:

LAMP COMMAND DALI 1.0	3x00512 4x00512 I:511	????	919	A0	UINT16 W/O	NO
			0x0397			
			COMMAND	97:QUERY VERSION NUMBER		
			LAMP	3:LAMP SHORT ADDRESS 4		
Sends a command to a specific lamp short address. The 16 bit value is divided into two parts: Upper 8 bits:15-8: Short address of lamp 0..63 for lamp 1 to 64 Lower 8 bits:7-0: Command value between 0 and 255 or 0x00 and 0xFF If you read this register, you can poll the 8 bit DALI answer. The return value is defined with: 0x8000: No answer received from the DALI bus up to now 0x20FF: A collision was detected on the DALI bus 0x00..0xFF: The 8 bit result of the last command. HINT: After you have read out the DALI result, the next MODBUS readout will again result in 0x8000 for no further 8 bit data! So you can only readout the result one time						

Sample code for sending a command to a specific DALI lamp. Please refer to the section Basic DALI command list for more details about commands.

```
// ASio: In our development system the handle to the serial port
// AUnitID: The unit ID of the RESI-DALI-SIO module
sub DALI.Command1x.Lamp(num ASIO,int AUnitID,int ALamp,int ACommand)
  int v
  bit r
  v=(ALamp<<8)|ACommand
  // REGISTER: LAMP COMMAND DALI 1.0 3x00512 4x00512 I:511
  r=ModbusMaster.MbWriteSingleRegister(ASIO,AUnitID,511,v)
  if r==true then
    DALI.Info("SUCCESS")
  else
    DALI.Info("ERROR with MODBUS communication")
  endif
endsub
```

```
// Switch off lamp 1
DALI.Command1x.Lamp(SIO,255,1,0x00)
// recall max level lamp 63
DALI.Command1x.Lamp(SIO,255,63,0x05)
```

45.11.5 Send DALI 1.0 command to group of lamps

To send a DALI 1.0 command to a group of lamps use this register:

GROUP COMMAND DALI 1.0	3x00522 4x00522 I:521	????	151	A0	UINT16 W/O	NO
			0x0097			
			COMMAND	97:QUERY VERSION NUMBER		
			GROUP	0:LAMP GROUP 1		
Sends a command to a specific lamp group. The 16 bit value is divided into two parts: Upper 8 bits:15-8: Group of lamp 0..15 for lamp group 1 to 16 Lower 8 bits:7-0: Command value between 0 and 255 or 0x00 and 0xFF If you read this register, you can poll the 8 bit DALI answer. The return value is defined with: 0x8000: No answer received from the DALI bus up to now 0x20FF: A collsion was detected on the DALI bus 0x00..0xFF: The 8 bit result of the last command. HINT: After you have read out the DALI result, the next MODBUS readout will again result in 0x8000 for no further 8 bit data! So you can only readout the result one time						

Sample code for sending a command to a specific DALI lamp group. Please refer to the section Basic DALI command list for more details about commands.

```
// ASio: In our development system the handle to the serial port
// AUnitID: The unit ID of the RESI-DALI-SIO module
sub DALI.Command1x.Group(num ASIO,int AUnitID,int AGroup,int ACommand)
  int v
  bit r
  v=(AGroup<<8)|ACommand
  // REGISTER: GROUP COMMAND DALI 1.0 3x00522 4x00522 I:521
  r=ModbusMaster.MbWriteSingleRegister(ASIO,AUnitID,521,v)
  if r==true then
    DALI.Info("SUCCESS")
  else
    DALI.Info("ERROR with MODBUS communication")
  endif
endsub

// Switch off lamp group 1
DALI.Command1x.Group(SIO,255,1,0x00)
// recall max level lamp group 15
DALI.Command1x.Group(SIO,255,15,0x05)
```

45.11.6 Send DALI 1.0 command to all lamps

To send a DALI 1.0 command to all lamps use this register:

ALL COMMAND DALI 1.0	3x00532 4x00532 I:531	????	151	A0	UINT16 W/O	NO
			0x0097			
			COMMAND	97:QUERY VERSION NUMBER		
Sends a command to all lamps. Only the lower 8 bits of the 16 bit value is used: Upper 8 bits:15-8: Always 0 Lower 8 bits:7-0: Command value between 0 and 255 or 0x00 and 0xFF If you read this register, you can poll the 8 bit DALI answer. The return value is defined with: 0x8000: No answer received from the DALI bus up to now 0x20FF: A collsion was detected on the DALI bus 0x00..0xFF: The 8 bit result of the last command. HINT: After you have read out the DALI result, the next MODBUS readout will again result in 0x8000 for no further 8 bit data! So you can only readout the result one time						

Sample code for sending a command to all DALI lamps. Please refer to the section Basic DALI command list for more details about commands.

```
// ASio: In our development system the handle to the serial port
// AUnitID: The unit ID of the RESI-DALI-SIO module
sub DALI.Command1x.All(num ASIO,int AUnitID,int ACommand)
  int v
  bit r
  // ACommand 8 bit
  v=ACommand
  // REGISTER: ALL COMMAND DALI 1.0 3x00532 4x00532 I:531
  r=ModbusMaster.MbWriteSingleRegister(ASIO,AUnitID,531,v)
  if r==true then
    DALI.Info("SUCCESS")
  else
    DALI.Info("ERROR with MODBUS communication")
  endif
endsub
```

```
// Switch off all lamps
DALI.Command1x.All(SIO,255,0x00)
// recall max level on all lamps
DALI.Command1x.All(SIO,255,0x05)
```

45.11.7 Send DALI 1.0 16 bit command to all lamps

To send a DALI 1.0 16 Bit command to all lamps use this register:

DIRECT 16 BIT COMMAND DALI 1.0	3x00542 4x00542 I:541	????	43264	A100	UINT16 W/O	NO
			0xA900			
			COMMAND	A900:COMPARE		
			V	A0		

Sends a 16 bit DALI 1.0 frame to the DALI bus line

If you read this register, you can poll the 8 bit DALI answer. The return value is defined with:
0x8000: No answer received from the DALI bus up to now
0x20FF: A collsion was detected on the DALI bus
0x00..0xFF: The 8 bit result of the last command.
HINT: After you have read out the DALI result, the next MODBUS readout will again result in 0x8000 for no further 8 bit data! So you can only readout the result one time

Sample code for sending a 16 bit command to all DALI lamps. Please refer to the section Basic DALI command list for more details about commands.

```
// ASio: In our development system the handle to the serial port
// AUnitID: The unit ID of the RESI-DALI-SIO module
sub DALI.16BitCommand1x.All(num ASIO,int AUnitID,int ACommand)
  int v
  bit r
  // ACommand 16 bit
  v=ACommand
  // REGISTER: DIRECT 16 BIT  COMMAND DALI 1.0 3x00542 4x00542 I:541
  r=ModbusMaster.MbWriteSingleRegister(ASIO,AUnitID,541,v)
  if r==true then
    DALI.Info("SUCCESS")
  else
    DALI.Info("ERROR with MODBUS communication")
  endif
endsub

// Load DTR with value 0x12
DALI.16BitCommand1x.All(SIO,255,0xA312)
// enable device type 8
DALI.16BitCommand1x.All(SIO,255,0xC108)

// Set short address 0 to 100%
DALI.16BitCommand1x.All(SIO,255,0x0x0000|(0<<9)|0xFE)
// Set short address 63 to 100%
DALI.16BitCommand1x.All(SIO,255,0x0x0000|(63<<9)|0xFE)
// Set group address 0 to 100%
DALI.16BitCommand1x.All(SIO,255,0x0x8000|(0<<9)|0xFE)
// Set group address 15 to 100%
DALI.16BitCommand1x.All(SIO,255,0x0x8000|(15<<9)|0xFE)
// Set all lamps to 100%
DALI.16BitCommand1x.All(SIO,255,0x0xFE00|0xFE)

// Send command OFF to short address 0
DALI.16BitCommand1x.All(SIO,255,0x0x0100|(0<<9)|0x00)
// Send command OFF to short address 63
DALI.16BitCommand1x.All(SIO,255,0x0x0100|(63<<9)|0x00)
// Send command OFF to group address 0
DALI.16BitCommand1x.All(SIO,255,0x0x8100|(0<<9)|0x00)
// Send command OFF to group address 15
DALI.16BitCommand1x.All(SIO,255,0x0x8100|(15<<9)|0x00)
// Send command OFF ot all lamps
DALI.16BitCommand1x.All(SIO,255,0x0xFF00|0x00)
```

45.11.8 Send DALI 1.0 command twice to lamp

Special DALI commands need to be sent twice within 100ms to the DALI ballasts. Therefore we have special MODBUS registers implemented. To send a DALI 1.0 command to a lamp twice within 100ms use this register:

LAMP COMMAND+REPEAT DALI 1.0	3x00513 4x00513 I:512	????	768	A0	UINT16 W/O	NO
			0x0300			
			COMMAND	00:OFF		
			LAMP	3:LAMP SHORT ADDRESS 4		
Sends a command twice within 100ms to a specific lamp short address. The 16 bit value is divided into two parts: Upper 8 bits:15-8: Short address of lamp 0..63 for lamp 1 to 64 Lower 8 bits:7-0: Command value between 0 and 255 or 0x00 and 0xFF If you read this register, you can poll the 8 bit DALI answer. The return value is defined with: 0x8000: No answer received from the DALI bus up to now 0x20FF: A collision was detected on the DALI bus 0x00..0xFF: The 8 bit result of the last command. HINT: After you have read out the DALI result, the next MODBUS readout will again result in 0x8000 for no further 8 bit data! So you can only readout the result one time						

Sample code for sending a command twice to a specific DALI lamp. Please refer to the section Basic DALI command list for more details about commands.

```
// ASio: In our development system the handle to the serial port
// AUnitID: The unit ID of the RESI-DALI-SIO module
sub DALI.Command2x.Lamp(num ASIO,int AUnitID,int ALamp,int ACommand)
  int v
  bit r
  v=(ALamp<<8)|ACommand
  // REGISTER: LAMP COMMAND+REPEAT DALI 1.0 3x00513 4x00513 I:512
  r=ModbusMaster.MbWriteSingleRegister(ASIO,AUnitID,512,v)
  if r==true then
    DALI.Info("SUCCESS")
  else
    DALI.Info("ERROR with MODBUS communication")
  endif
endsub

// Send STORE ACTUAL LEVEL IN DTR to ballast 1
DALI.Command2x.Lamp(SIO,255,1,0x21)

// Load DTR with value 0xFE 100%
DALI.16BitCommand1x.All(SIO,255,0xA3FE)
// Send STORE THE DTR AS MAX LEVEL to ballast 1
DALI.Command2x.Lamp(SIO,255,1,0x2A)
```

45.11.9 Send DALI 1.0 command twice to group of lamps

Special DALI commands need to be sent twice within 100ms to the DALI ballasts. Therefore we have special MODBUS registers implemented. To send a DALI 1.0 command to a lamp group twice within 100ms use this register:

GROUP COMMAND+REPEAT DALI 1.0	3x00523 4x00523 I:522	????	0	A0	UINT16 W/O	NO
			0x0000			
			COMMAND	00:OFF		
			GROUP	0:LAMP GROUP 1		
Sends a command twice within 100ms to a specific lamp group. The 16 bit value is divided into two parts: Upper 8 bits:15-8: Group of lamp 0..15 for lamp group 1 to 16 Lower 8 bits:7-0: Command value between 0 and 255 or 0x00 and 0xFF If you read this register, you can poll the 8 bit DALI answer. The return value is defined with: 0x8000: No answer received from the DALI bus up to now 0x20FF: A collision was detected on the DALI bus 0x00..0xFF: The 8 bit result of the last command. HINT: After you have read out the DALI result, the next MODBUS readout will again result in 0x8000 for no further 8 bit data! So you can only readout the result one time						

Sample code for sending a command to a specific DALI lamp group. Please refer to the section Basic DALI command list for more details about commands.

```
// ASio: In our development system the handle to the serial port
// AUnitID: The unit ID of the RESI-DALI-SIO module
sub DALI.Command2x.Group(num ASIO,int AUnitID,int AGroup,int ACommand)
  int v
  bit r
  v=(AGroup<<8)|ACommand
  // REGISTER: GROUP COMMAND+REPEAT DALI 1.0 3x00523 4x00523 I:522
  r=ModbusMaster.MbWriteSingleRegister(ASIO,AUnitID,522,v)
  if r==true then
    DALI.Info("SUCCESS")
  else
    DALI.Info("ERROR with MODBUS communication")
  endif
endsub
```

```
// Send STORE ACTUAL LEVEL IN DTR to lamp group 15
DALI.Command2x.Group(SIO,255,15,0x21)

// Load DTR with value 0xFE 100%
DALI.16BitCommand1x.All(SIO,255,0xA3FE)
// Send STORE THE DTR AS MAX LEVEL to ballast group 3
DALI.Command2x.Group(SIO,255,3,0x2A)
```

45.11.10 Send DALI 1.0 command twice to all lamps

Special DALI commands need to be sent twice within 100ms to the DALI ballasts. Therefore we have special MODBUS registers implemented. To send a DALI 1.0 command to all lamps twice within 100ms use this register:

ALL COMMAND+REPEAT DALI 1.0	3x00533 4x00533 I:532	????	0	A0	UINT16 W/O	NO
			0x0000			
			COMMAND	00:OFF		
Sends a command twice within 100ms to all lamps. Only the lower 8 bits of the 16 bit value is used: Upper 8 bits:15-8: Always 0 Lower 8 bits:7-0: Command value between 0 and 255 or 0x00 and 0xFF If you read this register, you can poll the 8 bit DALI answer. The return value is defined with: 0x8000: No answer received from the DALI bus up to now 0x20FF: A collision was detected on the DALI bus 0x00..0xFF: The 8 bit result of the last command. HINT: After you have read out the DALI result, the next MODBUS readout will again result in 0x8000 for no further 8 bit data! So you can only readout the result one time						

Sample code for sending a command to all DALI lamps. Please refer to the section Basic DALI command list for more details about commands.

```
// ASio: In our development system the handle to the serial port
// AUnitID: The unit ID of the RESI-DALI-SIO module
sub DALI.Command2x.All(num ASIO,int AUnitID,int ACommand)
  int v
  bit r
  // ACommand is 8 bit
  v=ACommand
  // REGISTER: ALL COMMAND+REPEAT DALI 1.0 3x00533 4x00533 I:532
  r=ModbusMaster.MbWriteSingleRegister(ASIO,AUnitID,532,v)
  if r==true then
    DALI.Info("SUCCESS")
  else
    DALI.Info("ERROR with MODBUS communication")
  endif
endsub

// Send STORE ACTUAL LEVEL IN DTR to all lamps
DALI.Command2x.All(SIO,255,0x21)

// Load DTR with value 0xFE 100%
DALI.16BitCommand1x.All(SIO,255,0xA3FE)
// Send STORE THE DTR AS MAX LEVEL to all lamps
DALI.Command2x.All(SIO,255,0x2A)
```

45.11.11 Send DALI 1.0 16 bit command twice to all lamps

Special DALI 16 bit commands need to be sent twice within 100ms to the DALI ballasts. Therefore we have special MODBUS registers implemented. To send a DALI 1.0 16 bit command to all lamps twice within 100ms use this register:

DIRECT 16 BIT COMMAND+REPEAT DALI 1.0	3x00543 4x00543 I:542	????	41216	A100	UINT16 W/O	NO
			0xA100			
			COMMAND	A100:TERMINATE		
			V	A0		
Sends a 16 bit DALI 1.0 frame twice within 100ms to to the DALI bus line If you read this register, you can poll the 8 bit DALI answer. The return value is defined with: 0x8000: No answer received from the DALI bus up to now 0x20FF: A collsion was detected on the DALI bus 0x00..0xFF: The 8 bit result of the last command. HINT: After you have read out the DALI result, the next MODBUS readout will again result in 0x8000 for no further 8 bit data! So you can only readout the result one time						

Sample code for sending a 16 bit command to all DALI lamps. Please refer to the section Basic DALI command list for more details about commands.

```
// ASio: In our development system the handle to the serial port
// AUnitID: The unit ID of the RESI-DALI-SIO module
sub DALI.16BitCommand2x.All(num ASIO,int AUnitID,int ACommand)
  int v
  bit r
  v=ACommand
  // REGISTER: DIRECT 16 BIT COMMAND+REPEAT DALI 1.0 3x00543 4x00543 I:542
  r=ModbusMaster.MbWriteSingleRegister(ASIO,AUnitID,542,v)
  if r==true then
    DALI.Info("SUCCESS")
  else
    DALI.Info("ERROR with MODBUS communication")
  endif
endsub

// Start random addressing mode INITIALIZE (ALL GEARS)
DALI.16BitCommand2x.All(SIO,255,0xA500)
```

45.11.12 Send DALI 1.0 command to lamp and wait for answer

To send a DALI 1.0 command to a lamp and wait for the answer use this register:

LAMP COMMAND DALI 1.0	3x00512 4x00512 I:511	????	919	A0	UINT16 W/O	NO
			0x0397			
			COMMAND	97:QUERY VERSION NUMBER		
			LAMP	3:LAMP SHORT ADDRESS 4		
Sends a command to a specific lamp short address. The 16 bit value is divided into two parts: Upper 8 bits:15-8: Short address of lamp 0..63 for lamp 1 to 64 Lower 8 bits:7-0: Command value between 0 and 255 or 0x00 and 0xFF If you read this register, you can poll the 8 bit DALI answer. The return value is defined with: 0x8000: No answer received from the DALI bus up to now 0x20FF: A collision was detected on the DALI bus 0x00..0xFF: The 8 bit result of the last command. HINT: After you have read out the DALI result, the next MODBUS readout will again result in 0x8000 for no further 8 bit data! So you can only readout the result one time						

Sample code for sending a command to a specific DALI lamp and wait for the answer or timeout. Please refer to the section Basic DALI command list for more details about commands.

```
// ASio: In our development system the handle to the serial port
// AUnitID: The unit ID of the RESI-DALI-SIO module
sub DALI.Command1x.Answer.Lamp(num ASIO,int AUnitID,int ALamp,int ACommand,intadr AAnswer)
  int v
  bit r,ok
  num Timeout
  v=(ALamp<<8)|ACommand
  AAnswer=0xFFFFFFFF
  // REGISTER: LAMP COMMAND DALI 1.0 3x00512 4x00512 I:511
  r=ModbusMaster.MbWriteSingleRegister(ASIO,AUnitID,511,v)
  if r then
    // We define 2000ms Timeout, if no answer is available...
    Timeout=DateTime.ActualDateTime2Num()+DateTime.MilliSeconds2Num(2000.)
    ok=false
    while DateTime.ActualDateTime2Num()<=Timeout and not ok
      r=ModbusMaster.MbReadHoldingRegisters(ASIO,AUnitID,511,v,1)
      if r==true then
        // 0x8000 means wait for answer
        // 0x20FF means collision
        // 0x00 to 0xFF means 8 bit answer
        if v!=0x8000 then
          DALI.Info("Answer is here:"+String.FormatHex(v,4))
          AAnswer=v
          ok=true
        endif
      endif
    endwhile
  endif
endsub


// query status of lamp 0
int MyAnswer

DALI.Command1x.Answer.Lamp(SIO,255,0,0x90,MyAnswer)
if MyAnswer==0xFFFFFFFF then
  DALI.Info("Host Timeout has happened"
elseif MyAnswer==0x20FF then
  DALI.Info("DALI bus collision has happened"
else
  DALI.Info("DALI Asnwer is:"+String.FormatHex(MyAnswer,4))
endif
```

45.11.13 Send DALI 1.0/2.0 command frames

To send a DALI 1.0 or DALI 2.0 commands to the DALI bus system use this registers:

DALI 8 BIT FRAME						
DALI10 SEND 8 BIT DALI FRAME	3x10001 4x10001 I:10000	????	255	FF	UINT32 W/O	YES
Writing a 8 bit value to this register generates a 8 bit DALI/DSI value on the DALI bus						
DALI10 SEND 8 BIT DALI FRAME AND REPEAT WITHIN 100ms	3x10003 4x10003 I:10002	????	160	A0	UINT32 W/O	YES
Writing a 8 bit value to this register generates a 8 bit DALI/DSI value on the DALI bus						
DALI10 SEND 8 BIT DALI FRAME	3x10005 4x10005 I:10004	????	255	FF	UINT32R W/O	YES
Writing a 8 bit value to this register generates a 8 bit DALI/DSI value on the DALI bus						
DALI10 SEND 8 BIT DALI FRAME AND REPEAT WITHIN 100ms	3x10007 4x10007 I:10006	????	160	A0	UINT32R W/O	YES
Writing a 8 bit value to this register generates a 8 bit DALI/DSI value on the DALI bus						
DALI 16 BIT FRAME						
DALI10 SEND 16 BIT DALI FRAME	3x10011 4x10011 I:10010	????	65278	FEFE	UINT32 W/O	YES
Writing a 16 bit value to this register generates a 16 bit DALI1.0 command on the DALI bus						
DALI10 SEND 16 BIT DALI FRAME AND REPEAT WITHIN 100ms	3x10013 4x10013 I:10012	????	65024	FE00	UINT32 W/O	YES
Writing a 16 bit value to this register generates a 16 bit DALI1.0 command on the DALI bus						
DALI10 SEND 16 BIT DALI FRAME	3x10015 4x10015 I:10014	????	65278	FEFE	UINT32R W/O	YES
Writing a 16 bit value to this register generates a 16 bit DALI1.0 command on the DALI bus						
DALI10 SEND 16 BIT DALI FRAME AND REPEAT WITHIN 100ms	3x10017 4x10017 I:10016	????	65024	FE00	UINT32R W/O	YES
Writing a 16 bit value to this register generates a 16 bit DALI1.0 command on the DALI bus						
DALI 24 BIT FRAME						
DALI20 SEND 24 BIT DALI FRAME	3x10021 4x10021 I:10020	????	8946700	88840C	UINT32 W/O	YES
Writing a 24 bit value to this register generates a 24 bit DALI2.0 frame on the DALI bus						
DALI20 SEND 24 BIT DALI FRAME AND REPEAT WITHIN 100ms	3x10023 4x10023 I:10022	????	8946701	88840D	UINT32 W/O	YES
Writing a 24 bit value to this register generates a 24 bit DALI2.0 frame on the DALI bus						
DALI20 SEND 24 BIT DALI FRAME	3x10025 4x10025 I:10024	????	8946700	88840C	UINT32R W/O	YES
Writing a 24 bit value to this register generates a 24 bit DALI2.0 frame on the DALI bus						
DALI20 SEND 24 BIT DALI FRAME AND REPEAT WITHIN 100ms	3x10027 4x10027 I:10026	????	8946701	88840D	UINT32R W/O	YES
Writing a 24 bit value to this register generates a 24 bit DALI2.0 frame on the DALI bus						

DALI 25 BIT eDALI FRAME – automatic redundand bit						
DALI20 SEND 25 BIT eDALI FRAME	3x10031 4x10031 I:10030	????	272742	42966	UINT32 W/O	YES
Writing a 24 bit value to this register generates out of a 24 bit eDALI command a valid 25 bit eDALI frame on the DALI bus						
DALI20 SEND 25 BIT eDALI FRAME AND REPEAT WITHIN 100ms	3x10033 4x10033 I:10032	????	272743	42967	UINT32 W/O	YES
Writing a 24 bit value to this register generates out of a 24 bit eDALI command a valid 25 bit eDALI frame on the DALI bus						
DALI20 SEND 25 BIT eDALI FRAME	3x10035 4x10035 I:10034	????	272742	42966	UINT32R W/O	YES
Writing a 24 bit value to this register generates out of a 24 bit eDALI command a valid 25 bit eDALI frame on the DALI bus						
DALI20 SEND 24 BIT DALI FRAME AND REPEAT WITHIN 100ms	3x10037 4x10037 I:10036	????	272743	42967	UINT32R W/O	YES
Writing a 24 bit value to this register generates out of a 24 bit eDALI command a valid 25 bit eDALI frame on the DALI bus						
DALI 25 BIT eDALI FRAME – user defined redundand bit						
DALI20 SEND 25 BIT eDALI FRAME	3x10041 4x10041 I:10040	????	545382	85266	UINT32 W/O	YES
Writing a 25 bit value to this register generates a 25 bit eDALI frame on the DALI bus						
DALI20 SEND 25 BIT eDALI FRAME AND REPEAT WITHIN 100ms	3x10043 4x10043 I:10042	????	545383	85267	UINT32 W/O	YES
Writing a 25 bit value to this register generates a 25 bit eDALI frame on the DALI bus						
DALI20 SEND 25 BIT eDALI FRAME	3x10045 4x10045 I:10044	????	545382	85266	UINT32R W/O	YES
Writing a 25 bit value to this register generates a 25 bit eDALI frame on the DALI bus						
DALI20 SEND 25 BIT DALI FRAME AND REPEAT WITHIN 100ms	3x10047 4x10047 I:10046	????	545383	85267	UINT32R W/O	YES
Writing a 25 bit value to this register generates a 25 bit eDALI frame on the DALI bus						
DALI 28 BIT FRAME						
DALI20 SEND 28 BIT DALI FRAME	3x10051 4x10051 I:10050	????	19088743	1234567	UINT32 W/O	YES
Writing a 28 bit value to this register generates a 28 bit DALI2.0 frame on the DALI bus						
DALI20 SEND 28 BIT DALI FRAME AND REPEAT WITHIN 100ms	3x10053 4x10053 I:10052	????	19088744	1234568	UINT32 W/O	YES
Writing a 28 bit value to this register generates a 28 bit DALI2.0 frame on the DALI bus						
DALI20 SEND 28 BIT DALI FRAME	3x10055 4x10055 I:10054	????	19088743	1234567	UINT32R W/O	YES
Writing a 28 bit value to this register generates a 28 bit DALI2.0 frame on the DALI bus						
DALI20 SEND 28 BIT DALI FRAME AND REPEAT WITHIN 100ms	3x10057 4x10057 I:10056	????	19088744	1234568	UINT32R W/O	YES
Writing a 28 bit value to this register generates a 28 bit DALI2.0 frame on the DALI bus						
DALI 32 BIT FRAME						
DALI20 SEND 32 BIT DALI FRAME	3x10051 4x10051 I:10050	????	305419896	12345678	UINT32 W/O	YES
Writing a 32 bit value to this register generates a 32 bit DALI2.0 frame on the DALI bus						
DALI20 SEND 32 BIT DALI FRAME AND REPEAT WITHIN 100ms	3x10053 4x10053 I:10052	????	305419896	12345678	UINT32 W/O	YES
Writing a 32 bit value to this register generates a 32 bit DALI2.0 frame on the DALI bus						
DALI20 SEND 32 BIT DALI FRAME	3x10055 4x10055 I:10054	????	305419896	12345678	UINT32R W/O	YES
Writing a 32 bit value to this register generates a 32 bit DALI2.0 frame on the DALI bus						
DALI20 SEND 32 BIT DALI FRAME AND REPEAT WITHIN 100ms	3x10057 4x10057 I:10056	????	305419896	12345678	UINT32R W/O	YES
Writing a 32 bit value to this register generates a 32 bit DALI2.0 frame on the DALI bus						

Sample code for sending commands with special frame length. Please refer to the section Basic DALI command list for more details about commands.

```
// ASio: In our development system the handle to the serial port
// AUnitID: The unit ID of the RESI-DALI-SIO module
sub DALI.Command.nBits(num ASIO,int AUnitID,int ABits,int ACommand,bit ARepeat)
  int V[2]
  bit r
  // Split ACommand into two 16 bit registers
  V[0]=(ACommand>>16)&0xFFFF
  V[1]=ACommand&0xFFFF
  if ABits==8 then
    // REGISTER: DALI10 SEND 8 BIT DALI FRAME    3x10001 4x10001 I:10000
    // REGISTER: DALI10 SEND 8 BIT DALI FRAME AND REPEAT WITHIN 100ms  3x10003 4x10003 I:10002
    if ARepeat==false then
      r=ModbusMaster.MbWriteMultipleRegisters(ASIO,AUnitID,10000,V,2)
    else
      r=ModbusMaster.MbWriteMultipleRegisters(ASIO,AUnitID,10002,V,2)
    endif
    return
  elseif ABits==16 then
    // REGISTER: DALI10 SEND 16 BIT DALI FRAME   3x10011 4x10011 I:10010
    // REGISTER: DALI10 SEND 16 BIT DALI FRAME AND REPEAT WITHIN 100ms 3x10013 4x10013 I:10012
    if ARepeat==false then
      r=ModbusMaster.MbWriteMultipleRegisters(ASIO,AUnitID,10010,V,2)
```

```
    else
      r=ModbusMaster.MbWriteMultipleRegisters(ASIO,AUnitID,10012,V,2)
    endif
    return
  elseif ABits==24 then
    // REGISTER: DALI20 SEND 24 BIT DALI FRAME   3x10021 4x10021 I:10020
    // REGISTER: DALI20 SEND 24 BIT DALI FRAME AND REPEAT WITHIN 100ms 3x10023 4x10023 I:10022
    if ARepeat==false then
      r=ModbusMaster.MbWriteMultipleRegisters(ASIO,AUnitID,10020,V,2)
    else
      r=ModbusMaster.MbWriteMultipleRegisters(ASIO,AUnitID,10022,V,2)
    endif
    return
  elseif ABits==25 then
    // REGISTER: DALI20 SEND 25 BIT eDALI FRAME  3x10031 4x10031 I:10030
    // REGISTER: DALI20 SEND 25 BIT eDALI FRAME AND REPEAT WITHIN 100ms       3x10033 4x10033 I:10032
    if ARepeat==false then
      r=ModbusMaster.MbWriteMultipleRegisters(ASIO,AUnitID,10030,V,2)
    else
      r=ModbusMaster.MbWriteMultipleRegisters(ASIO,AUnitID,10032,V,2)
    endif
    return
  elseif ABits==28 then
    // REGISTER: DALI20 SEND 28 BIT DALI FRAME   3x10051 4x10051 I:10050
    // REGISTER: DALI20 SEND 28 BIT DALI FRAME AND REPEAT WITHIN 100ms 3x10053 4x10053 I:10052
    if ARepeat==false then
      r=ModbusMaster.MbWriteMultipleRegisters(ASIO,AUnitID,10050,V,2)
    else
      r=ModbusMaster.MbWriteMultipleRegisters(ASIO,AUnitID,10052,V,2)
    endif
    return
  elseif ABits==32 then
    // REGISTER: DALI20 SEND 32 BIT DALI FRAME   3x10061 4x10061 I:10060
    // REGISTER: DALI20 SEND 32 BIT DALI FRAME AND REPEAT WITHIN 100ms 3x10063 4x10063 I:10062
    if ARepeat==false then
      r=ModbusMaster.MbWriteMultipleRegisters(ASIO,AUnitID,10060,V,2)
    else
      r=ModbusMaster.MbWriteMultipleRegisters(ASIO,AUnitID,10062,V,2)
    endif
    return
  else
    DALI.Info("Wrong DALI frame bit length!")
  endif
endsub

// query status of lamp 0
int MyAnswer

DALI.Command1x.Answer.Lamp(SIO,255,0,0x90,MyAnswer)
if MyAnswer==0xFFFFFFFF then
  DALI.Info("Host Timeout has happened"
elseif MyAnswer==0x20FF then
  DALI.Info("DALI bus collision has happened"
else
  DALI.Info("DALI Asnwer is:"+String.FormatHex(MyAnswer,4))
endif

// Send 8 bit value without repeat
DALI.Command.nBits(SIO,255,8,0xFF,false)
// Send 16 bit command Brightness to 0% to all lamps
DALI.Command.nBits(SIO,255,16,0xFE00,false)
// Send 24 bit DALI 2.0 control gear command and repeat it within 100ms
DALI.Command.nBits(SIO,255,24,0x123456,true)
// Send 25 bit eDALI frame
DALI.Command.nBits(SIO,255,25,0x123456,false)
// Send 28 bit DALI frame for future use
DALI.Command.nBits(SIO,255,28,0x1234567,false)
// Send 32 bit DALI frame for future use
DALI.Command.nBits(SIO,255,32,0x12345678,false)
```

45.11.14 Receive DALI 1.0/2.0 frames

To received a DALI 1.0 or DALI 2.0 frames the gateway has an internal ring puffer. To red out this ring puffer, you can use this registers:

DALI ANSWER RING PUFFER						
DALI RING PUFFER COUNTER	3x20001 4x20001 I:20000	1,0x0001 B:00 01	1	1	UINT16 R/O	
		Current receive counter:1				
Returns a 16 bit counter for indicating a change in the received data: Every time the gateway receives a new DALI frame this counter will be incremented by one.						
DALI RING PUFFER FLAGS	3x20002 4x20002 I:20001	2,0x0002 B:00 02	1	1	UINT16 R/O	
		Flags of the received DALI frame:2 -> 2:RX				
Indicates additional information for the DALI frame: =1:DALI frame was transmitted by the gateway =2:DALI frame was received by the gateway =3:DALI collision on the bus detected						
DALI RING PUFFER BITLENGTH	3x20003 4x20003 I:20002	24,0x0018 B:00 18	1	1	UINT16 R/O	
		Bit size of last received DALI frame:24 bits				
Indicates the DALI frame length in bits of the last received DALI frame						
DALI RING PUFFER DATA 32 BIT	3x20004 4x20004 I:20003	8946700,0x0088840C B:00 88 84 0C	1	1	UINT32 R/O	
		32 bit DALI frame from ring puffer:0088840C				
Contains the 32 bit value of the last received DALI frame						
DALI RING PUFFER TICKS	3x20006 4x20006 I:20005	24449,0x00005F81 B:00 00 5F 81	1	1	UINT32 R/O	
		Gap to previous DALI frame:24449ms -> 24,449s				
Contains a value in Milliseconds, which describes the gap to the previous DALI frame on the DALI bus.						
DALI RING PUFFER DATA 32 BIT	3x20008 4x20008 I:20007	8946700,0x0088840C B:84 0C 00 88	1	1	UINT32R R/O	
		32 bit DALI frame from ring puffer:0088840C				
Contains the 32 bit value of the last received DALI frame						
DALI RING PUFFER TICKS	3x20010 4x20010 I:20009	24449,0x00005F81 B:5F 81 00 00	1	1	UINT32R R/O	
		Gap to previous DALI frame:24449ms -> 24,449s				
Contains a value in Milliseconds, which describes the gap to the previous DALI frame on the DALI bus.						

Sample code for reading the internal ring puffer:

```
// Counter of last read ring puffer entry
int OldCounter

sub DALI.Init()
  OldCounter=0xFFFFFFFF
endsub

// ASio: In our development system the handle to the serial port
// AUnitID: The unit ID of the RESI-DALI-SIO module
function  bit  DALI.Check.RxRing(num  ASIO,int  AUnitID,intadr  ACounter,intadr  AFlags,intadr  ABits,intadr
AFrame,intadr AGap)
  int V[7]
  bit r
  // REGISTER: DALI RING PUFFER COUNTER  3x20001 4x20001 I:20000
  // REGISTER: DALI RING PUFFER FLAGS 3x20002 4x20002 I:20001
  // REGISTER: DALI RING PUFFER BITLENGTH 3x20003 4x20003 I:20002
  // REGISTER: DALI RING PUFFER DATA 3x20004 4x20004 I:20003
  // REGISTER: DALI RING PUFFER TICKS    3x20006 4x20006 I:20005
  r=ModbusMaster.MbWriteMultipleRegisters(ASIO,AUnitID,10000,V,7)
  if r==true then
    ACounter=V[0]
    AFlags=V[1]
    ABits=V[2]
    // UINT32 -> first register: HIGH 16 bit word, second register LOW 16 bit word
    AFrame=(V[3]<<16)|V[4]
    // UINT32 -> first register: HIGH 16 bit word, second register LOW 16 bit word
    AGap=(V[5]<<16)|V[6]
    // Check for new data in ring buffer ...
    if ACounter!=OldCounter then
      OldCounter=ACounter
      return true
    else
      return false
    endif
  else
    DALI.Info("MODBUS error")
    return false
  endif
  return false
endfunction
```

```
// Init system
DALI.Init()

sub DALI.Test.Receive()
  int MyCounter
  int MyFlags
  int MyBits
  int MyFrame
  int MyGap
  while true
    // Check for new data in receive ringpuffer...
    if DALI.Check.RxRing(SIO,UnitID,MyCounter,MyFlags,MyBits,MyFrame,MyGap) then

      // Check flags
      if MyFlags==1 then
        DALI.Info("DALI frame was sended by the RESI-SIO-xxx gateway")
      elseif MyFlags==2 then
        DALI.Info("DALI frame was received by the RESI-SIO-xxx gateway")
      elseif MyFlags==3 then
        DALI.Info("Collision on the DALI bus was detected by RESI-SIO-xxx gateway")
      endif

      // Check Bit Length
      if MyBits==8 then
        DALI.Info("8 bit frame:DALI 1.0 answer:"+String.FormatHex(MyFrame&0xFF,2))
      elseif MyBits==16 then
        DALI.Info("16 bit frame:DALI 1.0 command:"+String.FormatHex(MyFrame&0xFFFF,6))
      elseif MyBits==24 then
        DALI.Info("24 bit frame:DALI 2.0 command:"+String.FormatHex(MyFrame&0xFFFFFF,6))
      elseif MyBits==25 then
        DALI.Info("25 bit frame:eDALI command:"+String.FormatHex(MyFrame&0xFFFFFF,6))
      elseif MyBits==28 then
        DALI.Info("28 bit frame:future use:"+String.FormatHex(MyFrame&0xFFFFFFF,7))
      elseif MyBits==32 then
        DALI.Info("32 bit frame:future use:"+String.FormatHex(MyFrame,8))
      else
        DALI.Info("Invalid frame:"+String.FormatInt(MyBits,1)+":"+String.FormatHex(MyFrame,8))
      endif

      // Show gap time to previous DALI frame
      DALI.Info("Gap:"+String.FormatInt(MyGap,1)+"ms")

    endif
  endwhile
endsub
```

45.12 Basic DALI command list

This is an excerpt from the DALI standard, it is only for quick information. For more details refer to the original DALI standard.

STANDARD	VALUE	COMMAND	ADDRESSING/PARAMETER	FUNCTION
DALI 1.0	0,0x00	OFF	ADDRESSING: LAMP, GROUP, ALL DTR, DTR1, DTR2: NO ANSWER: NO, REPEAT: NO	Switch off lamp
DALI 1.0	1,0x01	UP	ADDRESSING: LAMP, GROUP, ALL DTR, DTR1, DTR2: NO ANSWER: NO, REPEAT: NO	Dim lamp up for 200ms. If the lamp was switched off it stays switched off.
DALI 1.0	2,0x02	DOWN	ADDRESSING: LAMP, GROUP, ALL DTR, DTR1, DTR2: NO ANSWER: NO, REPEAT: NO	Dim lamp down for 200ms. As soon as the lower brightness threshold level is reached the lamp stays at this value.
DALI 1.0	3,0x03	STEP UP	ADDRESSING: LAMP, GROUP, ALL DTR, DTR1, DTR2: NO ANSWER: NO, REPEAT: NO	Lamp switches one step brighter. If the lamp was switched off it stays switched off.
DALI 1.0	4,0x04	STEP DOWN	ADDRESSING: LAMP, GROUP, ALL DTR, DTR1, DTR2: NO ANSWER: NO, REPEAT: NO	Lamp switches one step darker. As soon as the lower brightness threshold level is reached the lamp stays at this value.
DALI 1.0	5,0x05	RECALL MAX LEVEL	ADDRESSING: LAMP, GROUP, ALL DTR, DTR1, DTR2: NO ANSWER: NO, REPEAT: NO	Set lamp to maximal threshold level.
DALI 1.0	6,0x06	RECALL MIN LEVEL	ADDRESSING: LAMP, GROUP, ALL DTR, DTR1, DTR2: NO ANSWER: NO, REPEAT: NO	Set lamp to minimal threshold level.
DALI 1.0	7,0x07	STEP DOWN AND OFF	ADDRESSING: LAMP, GROUP, ALL DTR, DTR1, DTR2: NO ANSWER: NO, REPEAT: NO	Lamp switches one step darker. If the lower brightness threshold level is reached the lamp switches off.
DALI 1.0	8,0x08	ON AND STEP UP	ADDRESSING: LAMP, GROUP, ALL DTR, DTR1, DTR2: NO ANSWER: NO, REPEAT: NO	Lamp switches one step brighter. If the lamp was switched off it is now switched on.
DALI 1.0	9,0x09	ENABLE DAPC SEQUENCE	ADDRESSING: LAMP, GROUP, ALL DTR, DTR1, DTR2: NO ANSWER: NO, REPEAT: NO	This command defines the start of a direct arc power control (DAPC) sequence that allows the host to control the dimming speed by sending direct arc power commands.
DALI 1.0	10,0x0A	GOTO LAST ACTIVE LEVEL	ADDRESSING: LAMP, GROUP, ALL DTR, DTR1, DTR2: NO ANSWER: NO, REPEAT: NO	Indirect Arc Power command. The last active level will be restored.
DALI 1.0	16,0x10 to 31,0x1F	GOTO SCENE x	ADDRESSING: LAMP, GROUP, ALL DTR, DTR1, DTR2: NO ANSWER: NO, REPEAT: NO	Load scene value stored in scene 0 to 15.
DALI 1.0	32,0x20	RESET	ADDRESSING: LAMP, GROUP, ALL DTR, DTR1, DTR2: NO ANSWER: NO, REPEAT: YES	Reset the ballast
DALI 1.0	33,0x21	STORE ACTUAL LEVEL IN DTR	ADDRESSING: LAMP, GROUP, ALL DTR, DTR1, DTR2: NO ANSWER: NO, REPEAT: YES	Stores the actual brightness level in the register DTR
DALI 1.0	42,0x2A	STORE THE DTR AS MAX LEVEL	ADDRESSING: LAMP, GROUP, ALL DTR: YES, DTR1, DTR2: NO ANSWER: NO, REPEAT: YES	Stores the actual register value DTR as maximum level for lamp
DALI 1.0	43,0x2B	STORE THE DTR AS MIN LEVEL	ADDRESSING: LAMP, GROUP, ALL DTR: YES, DTR1, DTR2: NO ANSWER: NO, REPEAT: YES	Stores the actual register value DTR as minimum level for lamp
DALI 1.0	44,0x2C	STORE DTR AS SYSTEM FAILURE LEVEL	ADDRESSING: LAMP, GROUP, ALL DTR: YES, DTR1, DTR2: NO ANSWER: NO, REPEAT: YES	Stores the actual register value DTR as system failure level for lamp
DALI 1.0	45,0x2D	STORE DTR AS POWER ON LEVEL	ADDRESSING: LAMP, GROUP, ALL DTR: YES, DTR1, DTR2: NO ANSWER: NO, REPEAT: YES	Stores the actual register value DTR as power on level for lamp
DALI 1.0	46,0x2E	STORE DTR AS FADE TIME	ADDRESSING: LAMP, GROUP, ALL DTR: YES, DTR1, DTR2: NO ANSWER: NO, REPEAT: YES	Stores the actual register value DTR as fade time for lamp
DALI 1.0	47,0x2F	STORE DTR AS FADE RATE	ADDRESSING: LAMP, GROUP, ALL DTR: YES, DTR1, DTR2: NO ANSWER: NO, REPEAT: YES	Stores the actual register value DTR as fade rate for lamp
DALI 1.0	64,0x40 to 79,0x4F	STORE THE DTR AS SCENE x	ADDRESSING: LAMP, GROUP, ALL DTR: YES, DTR1, DTR2: NO ANSWER: NO, REPEAT: YES	Stores the actual register value DTR as new brightness level for scene x (0 to 15)
DALI 1.0	80,0x50 to 95,0x5F	REMOVE FROM SCENE x	ADDRESSING: LAMP, GROUP, ALL DTR, DTR1, DTR2: NO ANSWER: NO, REPEAT: YES	Removes the affected lamps from the scene x (0 to 15)

STANDARD	VALUE	COMMAND	ADDRESSING/PARAMETER	FUNCTION
DALI 1.0	96,0x60 to 111,0x6F	ADD TO GROUP x	ADDRESSING: LAMP, GROUP, ALL DTR, DTR1, DTR2: NO ANSWER: NO, REPEAT: YES	Adds the lamps to group x (0 to 15)
DALI 1.0	112,0x70 to 127,0x7F	REMOVE FROM GROUP x	ADDRESSING: LAMP, GROUP, ALL DTR, DTR1, DTR2: NO ANSWER: NO, REPEAT: YES	Removes the lamps from group x (0 to 15)
DALI 1.0	128,0x80	STORE DTR AS SHORT ADDRESS	ADDRESSING: LAMP, GROUP, ALL DTR: YES, DTR1, DTR2: NO ANSWER: NO, REPEAT: YES	Stores the actual register value DTR as new short address
DALI 1.0	129,0x81	ENABLE WRITE MEMORY	ADDRESSING: LAMP, GROUP, ALL DTR, DTR1, DTR2: NO ANSWER: NO, REPEAT: YES	Activates memory write operations in ballasts.
DALI 1.0	144,0x90	QUERY STATUS	ADDRESSING: LAMP, GROUP, ALL DTR, DTR1, DTR2: NO ANSWER: YES, REPEAT: NO	Query of the status of the lamp Answer: Bit 0: Status of control gear 0=OK Bit 1: Lamp failure 0=OK Bit 2: Lamp arc power on 0=OFF 1=ON Bit 3: Limit Error 0=Actual level is between MIN and MAX or OFF Bit 4: Fade running 0=Fading is finished 1=Fading is active Bit 5: RESET STATE 0=OK Bit 6: Missing short address 0=No 1=Yes Bit 7: POWER FAILURE 0=No
DALI 1.0	145,0x91	QUERY CONTROL GEAR	ADDRESSING: LAMP, GROUP, ALL DTR, DTR1, DTR2: NO ANSWER: YES, REPEAT: NO	If the ballast exists and can answer, the answer is YES, otherwise NO. YES is 0xFF, NO is timeout or no answer
DALI 1.0	146,0x92	QUERY LAMP FAILURE	ADDRESSING: LAMP, GROUP, ALL DTR, DTR1, DTR2: NO ANSWER: YES, REPEAT: NO	If the ballast has a problem with the lamp, the answer is YES, otherwise NO. YES is 0xFF, NO is timeout or no answer
DALI 1.0	147,0x93	QUERY LAMP POWER ON	ADDRESSING: LAMP, GROUP, ALL DTR, DTR1, DTR2: NO ANSWER: YES, REPEAT: NO	If the ballast has switched on the lamp, the answer is YES, otherwise NO. YES is 0xFF, NO is timeout or no answer
DALI 1.0	148,0x94	QUERY LIMIT ERROR	ADDRESSING: LAMP, GROUP, ALL DTR, DTR1, DTR2: NO ANSWER: YES, REPEAT: NO	If the arc power of the ballast is outside of MIN and MAX LEVEL, the answer is YES, otherwise NO. YES is 0xFF, NO is timeout or no answer
DALI 1.0	149,0x95	QUERY RESET STATE	ADDRESSING: LAMP, GROUP, ALL DTR, DTR1, DTR2: NO ANSWER: YES, REPEAT: NO	If the ballast is in RESET STATE, the answer is YES, else NO. YES is 0xFF, NO is timeout or no answer
DALI 1.0	150,0x96	QUERY MISSING SHORT ADDRESS	ADDRESSING: LAMP, GROUP, ALL DTR, DTR1, DTR2: NO ANSWER: YES, REPEAT: NO	If the ballast has no short address (=0xFF MASK), the answer is YES, else NO. YES is 0xFF, NO is timeout or no answer
DALI 1.0	151,0x97	QUERY VERSION NUMBER	ADDRESSING: LAMP, GROUP, ALL DTR, DTR1, DTR2: NO ANSWER: YES, REPEAT: NO	The answer is always 1.
DALI 1.0	152,0x98	QUERY CONTENT DTR	ADDRESSING: LAMP, GROUP, ALL DTR, DTR1, DTR2: NO ANSWER: YES, REPEAT: NO	The answer is the current content of register DTR.
DALI 1.0	153,0x99	QUERY DEVICE TYPE	ADDRESSING: LAMP, GROUP, ALL DTR, DTR1, DTR2: NO ANSWER: YES, REPEAT: NO	The answer is the actual device type of the ballast. The following device types are defined by the DALI consortium: 0:Fluorescent lamp control gear 1:Self-contained emergency lamp control gear 2:Discharge (HID) lamp control gear 3:Low-voltage halogen lamp control gear 4:Incandescent lamp dimmer 5:DC voltage lamp dimmer (0/1-10V) 6:LED lamp control gear 7:witching (relay) control gear 8:Color lamp control gear 15:Load referencing" 16:Thermal gear protection 17:Dimming curve selection 18:Under consideration 19:Centrally supplied emergency operation 20:Demand response 21:Thermal lamp protection 22:Under consideration 23:Non-replaceable light source
DALI 1.0	154,0x9A	QUERY PHYSICAL MINIMUM LEVEL	ADDRESSING: LAMP, GROUP, ALL DTR, DTR1, DTR2: NO ANSWER: YES, REPEAT: NO	Returns the current physical minimum arc power level for the ballast in the range of 0 to 254. 255 means MASK.

STANDARD	VALUE	COMMAND	ADDRESSING/PARAMETER	FUNCTION
DALI 1.0	155,0x9B	QUERY POWER FAILURE	ADDRESSING: LAMP, GROUP, ALL DTR, DTR1, DTR2: NO ANSWER: YES, REPEAT: NO	If there is a failure with the arc power, the result is YES; else NO. YES is 0xFF, NO is timeout or no answer
DALI 1.0	156,0x9C	QUERY CONTENT DTR1	ADDRESSING: LAMP, GROUP, ALL DTR, DTR1, DTR2: NO ANSWER: YES, REPEAT: NO	The answer is the current content of register DTR1.
DALI 1.0	157,0x9D	QUERY CONTENT DTR2	ADDRESSING: LAMP, GROUP, ALL DTR, DTR1, DTR2: NO ANSWER: YES, REPEAT: NO	The answer is the current content of register DTR2.
DALI 1.0	160,0xA0	QUERY ACTUAL LEVEL	ADDRESSING: LAMP, GROUP, ALL DTR, DTR1, DTR2: NO ANSWER: YES, REPEAT: NO	The answer is the current arc power between 0 and 254 or 255 for MASK.
DALI 1.0	161,0xA1	QUERY MAX LEVEL	ADDRESSING: LAMP, GROUP, ALL DTR, DTR1, DTR2: NO ANSWER: YES, REPEAT: NO	The answer is the current setting of the maximum level for the arc power between 0 and 254, 255 means MASK
DALI 1.0	162,0xA2	QUERY MIN LEVEL	ADDRESSING: LAMP, GROUP, ALL DTR, DTR1, DTR2: NO ANSWER: YES, REPEAT: NO	The answer is the current setting of the minimum level for the arc power between 0 and 254, 255 means MASK
DALI 1.0	163,0xA3	QUERY POWER ON LEVEL	ADDRESSING: LAMP, GROUP, ALL DTR, DTR1, DTR2: NO ANSWER: YES, REPEAT: NO	The answer is the current setting of the power on level for the arc power between 0 and 254, 255 means MASK
DALI 1.0	164,0xA4	QUERY SYSTEM FAILURE LEVEL	ADDRESSING: LAMP, GROUP, ALL DTR, DTR1, DTR2: NO ANSWER: YES, REPEAT: NO	The answer is the current setting of the system failure level for the arc power between 0 and 254, 255 means MASK
DALI 1.0	165,0xA5	QUERY FADE TIME/FADE RATE	ADDRESSING: LAMP, GROUP, ALL DTR, DTR1, DTR2: NO ANSWER: YES, REPEAT: NO	The answer is the current setting of the fade time and fade rate for the ballast. The higher 4 bits define the fade time The lower 4 bits define the fade rate
DALI 1.0	178,0xB0 to 191,0xBF	QUERY SCENE LEVEL x	ADDRESSING: LAMP, GROUP, ALL DTR, DTR1, DTR2: NO ANSWER: YES, REPEAT: NO	The answer is the current setting of the scene x (0 to 15) level for the arc power between 0 and 254, 255 means MASK
DALI 1.0	192,0xC0	QUERY GROUPS 0-7	ADDRESSING: LAMP, GROUP, ALL DTR, DTR1, DTR2: NO ANSWER: YES, REPEAT: NO	The answer is the group membership of the ballast. Each bit stands for one group: Bit 0 for group 0, bit 1 for group 1,...
DALI 1.0	193,0xC1	QUERY GROUPS 8-15	ADDRESSING: LAMP, GROUP, ALL DTR, DTR1, DTR2: NO ANSWER: YES, REPEAT: NO	The answer is the group membership of the ballast. Each bit stands for one group: Bit 0 for group 8, bit 1 for group 9,...
DALI 1.0	194,0xC2	QUERY RANDOM ADDRESS (H)	ADDRESSING: LAMP, GROUP, ALL DTR, DTR1, DTR2: NO ANSWER: YES, REPEAT: NO	The answer are the highest 8 bit of the 24 bit random number used for random addressing mode
DALI 1.0	195,0xC3	QUERY RANDOM ADDRESS (M)	ADDRESSING: LAMP, GROUP, ALL DTR, DTR1, DTR2: NO ANSWER: YES, REPEAT: NO	The answer are the middle 8 bit of the 24 bit random number used for random addressing mode
DALI 1.0	196,0xC4	QUERY RANDOM ADDRESS (L)	ADDRESSING: LAMP, GROUP, ALL DTR, DTR1, DTR2: NO ANSWER: YES, REPEAT: NO	The answer are the lowest 8 bit of the 24 bit random number used for random addressing mode
DALI 1.0	196,0xC5	READ MEMORY LOCATION	ADDRESSING: LAMP, GROUP, ALL DTR, DTR1: YES, DTR2: NO ANSWER: YES, REPEAT: NO	The answer is the content of the memory location defines with DTR within the memory bank defined with DTR1
DALI 1.0	255,0xFF	QUERY EXTENDED VERSION NUMBER	ADDRESSING: LAMP, GROUP, ALL DTR, DTR1, DTR2: NO ANSWER: YES, REPEAT: NO	The answer is the extended version number of the ballast

STANDARD	VALUE	COMMAND	ADDRESSING/PARAMETER	FUNCTION
DALI 1.0	0xA100	TERMINATE	ADDRESSING: ALL DTR, DTR1, DTR2: NO ANSWER: NO, REPEAT: NO	This commands terminates special functions for all ballasts
DALI 1.0	0xA3HH	DTR=	ADDRESSING: ALL DTR, DTR1, DTR2: NO ANSWER: NO, REPEAT: NO	This command loads the hex value HH into the DTR register
DALI 1.0	0xA500	INITIALIZE (ALL GEARS)	ADDRESSING: ALL DTR, DTR1, DTR2: NO ANSWER: NO, REPEAT: YES	This command starts random addressing mode for all ballasts
DALI 1.0	0xA5FF	INITIALIZE (GEARS WITHOUT SHORT ADDRESS)	ADDRESSING: ALL DTR, DTR1, DTR2: NO ANSWER: NO, REPEAT: YES	This command starts random addressing mode for all ballasts without a short address
DALI 1.0	0xA5HH	INITIALIZE (SHORT ADDRESS)	ADDRESSING: ALL DTR, DTR1, DTR2: NO ANSWER: NO, REPEAT: YES	This command starts random addressing mode for one ballasts with a specific short address. HH=(SA<<1)|1
DALI 1.0	0xA700	RANDOMIZE	ADDRESSING: ALL DTR, DTR1, DTR2: NO ANSWER: NO, REPEAT: YES	This command commands the ballasts to dice a 24 bit random number for automatic addressing
DALI 1.0	0xA900	COMPARE	ADDRESSING: ALL DTR, DTR1, DTR2: NO ANSWER: NO, REPEAT: NO	This command activates a compare of the internal diced 24 bit number with the given number with SEARCHADR commands. if the internal number is less than or equal the answer is YES (0xFF)
DALI 1.0	0xAB00	WITHDRAW	ADDRESSING: ALL DTR, DTR1, DTR2: NO ANSWER: NO, REPEAT: NO	The ballast with the same 24 bit random number as defined by the SEARCHADR commands should exit the addressing mode.
DALI 1.0	0xB1HH	SEARCHADDRH=	ADDRESSING: ALL DTR, DTR1, DTR2: NO ANSWER: NO, REPEAT: NO	The command sets the highest 8 bit of the 24 bit random number for compare mode in all ballasts.
DALI 1.0	0xB3HH	SEARCHADDRM=	ADDRESSING: ALL DTR, DTR1, DTR2: NO ANSWER: NO, REPEAT: NO	The command sets the medium 8 bit of the 24 bit random number for compare mode in all ballasts.
DALI 1.0	0xB5HH	SEARCHADDRL=	ADDRESSING: ALL DTR, DTR1, DTR2: NO ANSWER: NO, REPEAT: NO	The command sets the lowest 8 bit of the 24 bit random number for compare mode in all ballasts.
DALI 1.0	0xB7HH	PROGRAM SHORT ADDRESS	ADDRESSING: ALL DTR, DTR1, DTR2: NO ANSWER: NO, REPEAT: YES	If the ballast is selected (physical selection mode or random address fits) the given short address is stored as new short address of the device. HH=(SA<<1)|1
DALI 1.0	0xB9HH	VERIFY SHORT ADDRESS	ADDRESSING: ALL DTR, DTR1, DTR2: NO ANSWER: NO, REPEAT: YES	The ballast asnwers with YES 0xFF, if the given short address is the same as the internal stored short address. HH=(SA<<1)|1
DALI 1.0	0xBB00	QUERY SHORT ADDRESS	ADDRESSING: ALL DTR, DTR1, DTR2: NO ANSWER: NO, REPEAT: YES	If the random address fits the ballast will answer with the internal stored short address in the format HH=(SA<<1)|1 or MASK (0xFF), if none is defined
DALI 1.0	0xBD00	PHYSICAL SELECTION	ADDRESSING: ALL DTR, DTR1, DTR2: NO ANSWER: NO, REPEAT: YES	This command start physical selection addressing mode
DALI 1.0	0xC1HH	ENABLE DEVICE TYPE	ADDRESSING: ALL DTR, DTR1, DTR2: NO ANSWER: NO, REPEAT: YES	If you want to use special device type depended commands you have to precede this commands with this enable command. HH is the selected device type e.g. 8)
DALI 1.0	0xC3HH	DTR1=	ADDRESSING: ALL DTR, DTR1, DTR2: NO ANSWER: NO, REPEAT: YES	This command loads the hex value HH into the DTR1 register
DALI 1.0	0xC5HH	DTR2=	ADDRESSING: ALL DTR, DTR1, DTR2: NO ANSWER: NO, REPEAT: YES	This command loads the hex value HH into the DTR2 register
DALI 1.0	0xBDHH	WRITE MEMORY LOCATION	ADDRESSING: ALL DTR, DTR1: YES, DTR2: NO ANSWER: NO, REPEAT: YES	This command writes the content HH into the memory bank defined with DTR1 at the memory address defined with DTR.

46 RESI-SMIx-SIO, RESI-SMIx-ETH, RESI-SMI-DIAG

46.1 General information

Our RESI-SMIx-SIO and RESI-SMIx-ETH family of gateways enables the integration of Standard Motor Interface® motors for shades & blinds into your BMS system with MODBUS or with ASCII commands.

This series SMI gateways offer the following features:

- Easy integration of a complete SMI motor system for shades & blinds
- MODBUS/RTU slave or MODBUS/TCP server protocol
- Additional commands with plain ASCII texts
- SMI interface and host interface are galvanically isolated
- Versions for 8 SMI motors and 16 SMI motors available
- Supports all known SMI commands
- RESI-SMIx-SIO: Special support to use the module together with the standard SMI easyMonitor 3® software
- RESI-xxx-SIO: Galvanic isolated RS232 and RS485 interface for communication with a host system
- RESI-xxx-ETH: Galvanic isolated Ethernet interface for communication with a host system

Figure: Our serial RESI-SMI16-SIO module

Figure: Our Ethernet RESI-SMI16-ETH module

With our RESI-SMI16-DIAG gateway we deliver a diagnostic system, which is connected to an existing SMI bus systemn. The gateway has no internal SMI power supply, but can send/receive every SMI command for diagnostic and analysis reasons.

Figure: Our RESI-SMI16-DIAG gateway

46.2 Technical specification

Beside the basic technical data, which fulfill all of our IO modules, this IO modules meet the following technical specifications:

Power consumption	
RESI-SMI16-SIO	<1.0W
RESI-SMI8-SIO	<1.0W
RESI-SMI16-ETH	<1.5W
RESI-SMI8-ETH	<1.5W
RESI-SMI16-DIAG	<1.0W

Product housing	
RESI-SMI16-SIO	CEM17
RESI-SMI8-SIO	CEM17
RESI-SMI16-ETH	CEM35
RESI-SMI8-ETH	CEM35
RESI-SMI-DIAG	CEM17

Product weight	
RESI-SMI16-SIO	50g
RESI-SMI8-SIO	50g
RESI-SMI16-ETH	83g
RESI-SMI8-ETH	83g
RESI-SMI16-DIAG	50g

SMI bus interface	
RESI-SMIx-SIO	
RESI-SMIx-ETH	
RESI-SMIx-DIAG	
Protocol	SMI - Standard Motor Interface
Baud rate	2400Bit/s
Cable connection	via terminals
Galvanic isolation	Yes
LED indicator	Yes

SMI power supply	
RESI-SMIx-SIO	
RESI-SMIx-ETH	
Nominal output voltage	typical ~22
Maximum output current	~23mA
Galvanic isolation	Yes
LED indicator	Yes

RESI-SMIx-DIAG	
No integrated SMI power supplying	

Default serial settings	
RESI-SMI16-SIO	
RESI-SMI8-SIO	
RESI-SMI16-DIAG	
Baud rate	via DIP switch
Parity	none
Stop bits	one
UnitID	255

Default Ethernet settings	
RESI-SMI16-ETH	
IP address	192.168.0.223
IP mask	255.255.255.0
gateway	192.168.0.1
UnitID	255

RESI-SMI8-ETH	
IP address	192.168.0.222
IP mask	255.255.255.0
gateway	192.168.0.1
UnitID	255

User	RESI
password	RESI

46.3 Additional terminals & LED states

SMI INTERFACE	SMI bus connector	
	One 3 pin terminal blocks	
	Terminal type:	USLIM
	SMI+:	SMI+ bus wire
	SMI-:	SMI- bus wire
	IMPORTANT: The SMI bus is not protected against reverse polarity!	
Pin layout	SMI+:	SMI+ bus wire
	N/C:	not connected
	SMI-:	SMI- bus wire
STATE	If the gateway has no error, this LED flashes with a 1s rhythm.	
	If the module has an internal error this LED flashes very quick.	
	(~250ms rhythm)	
SMI	If there is bus communication on the SMI, this LED is on, otherwise this LED is OFF	

46.4 RESI-SMIx-SIO: Connection diagram

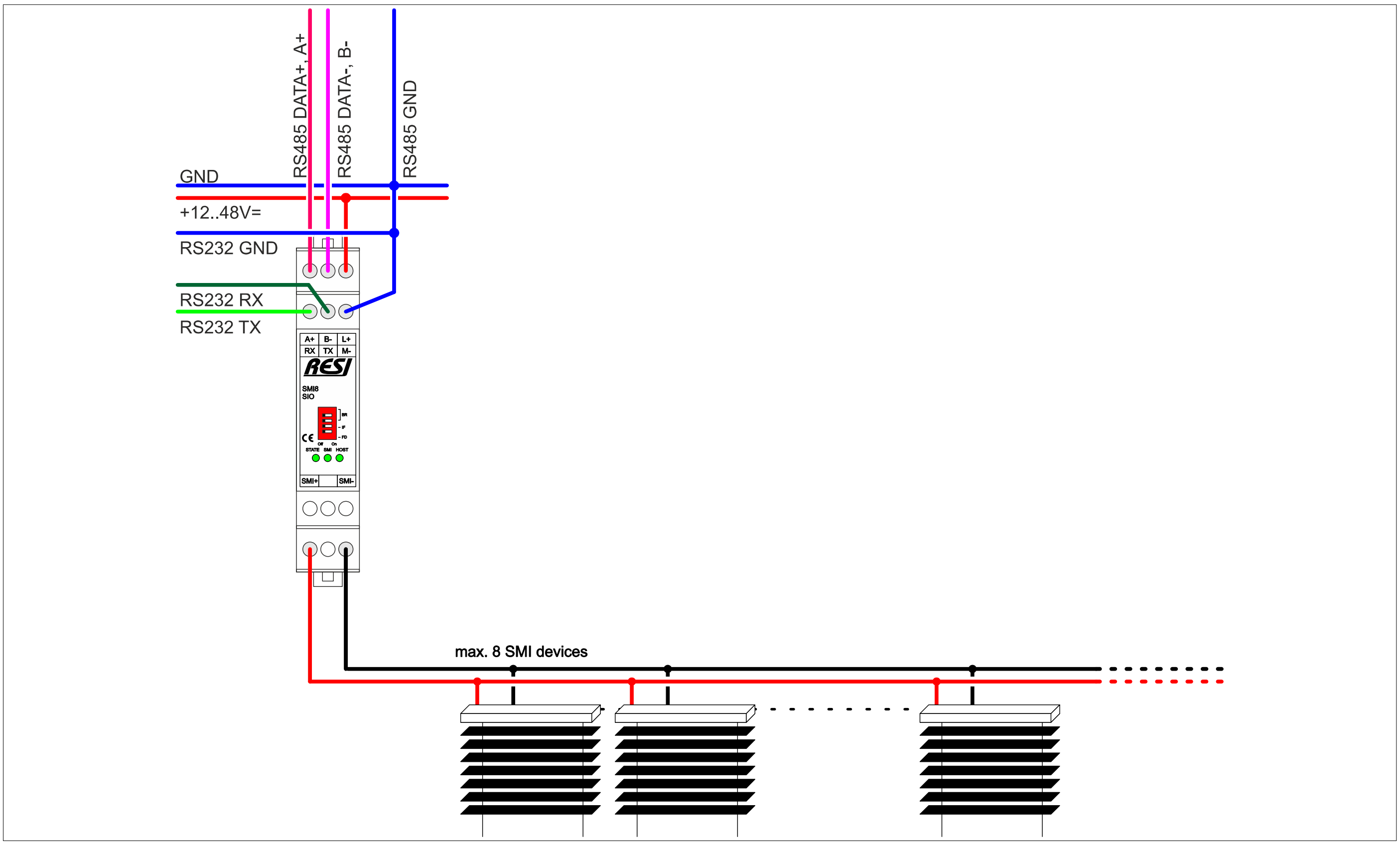


Figure: Connecting a SMI bus system to the RESI-SMI8-SIO gateway

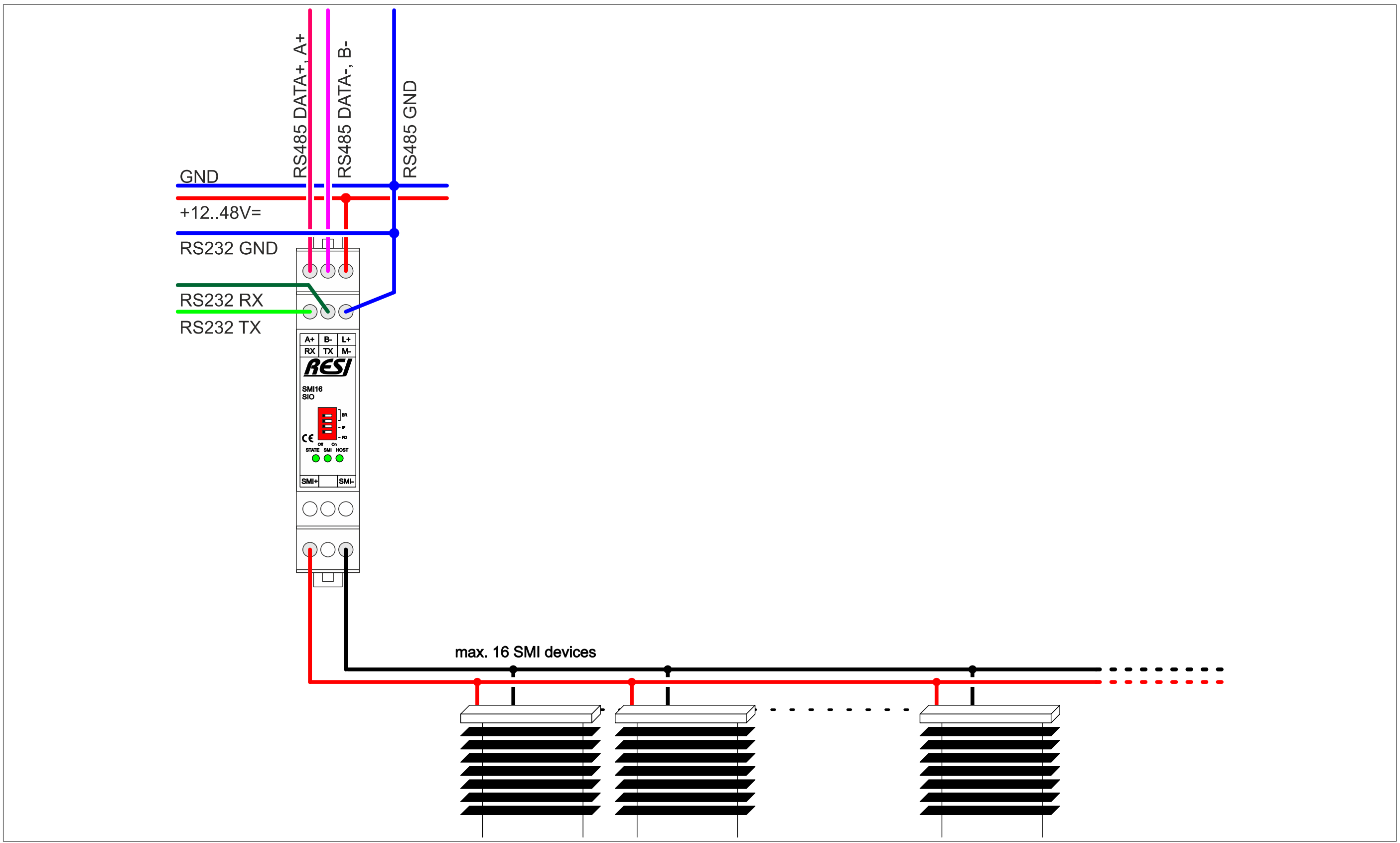


Figure: Connecting a SMI bus system to the RESI-SMI16-SIO gateway

46.5 RESI-SMIx-ETH: Connection diagram

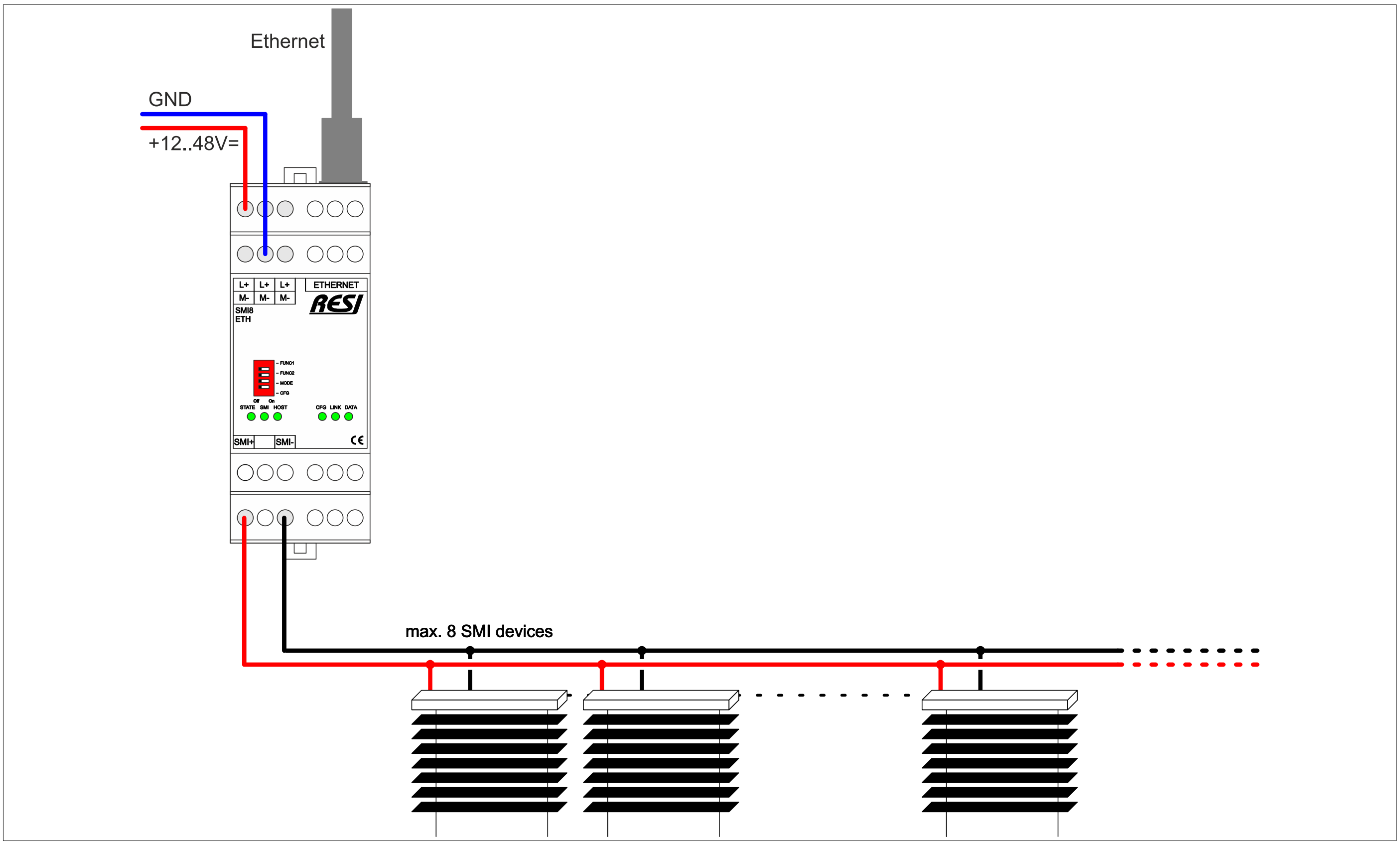


Figure: Connecting a SMI bus system to the RESI-SMI8-ETH gateway

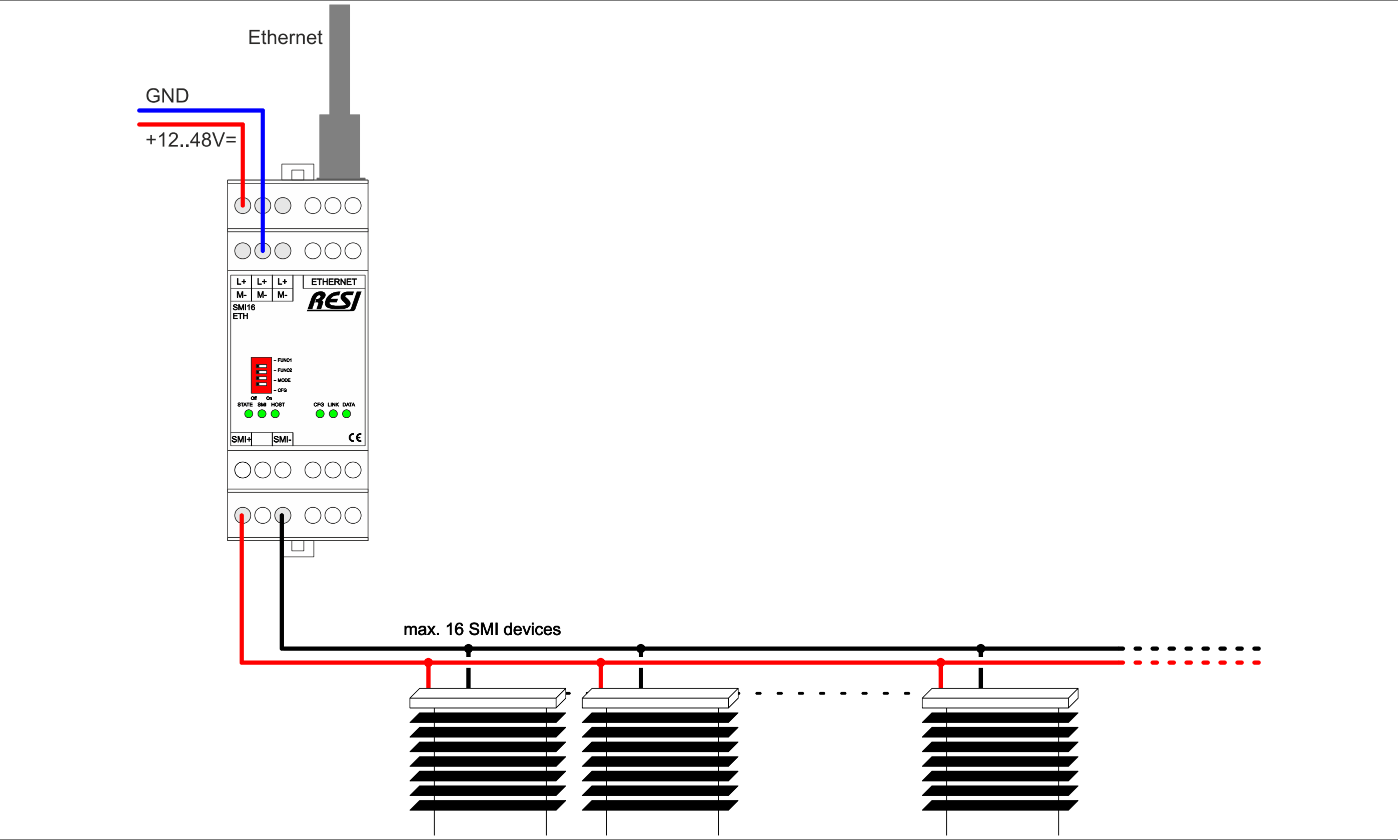


Figure: Connecting a SMI bus system to the RESI-SMI16-ETH gateway

46.6 SMI bus installation

The SMI interface offers the following features in general:

- Basic idea: One gateway and up to 16 motors
- Gateway is master, motors are slaves
- Connecting line is 5-wire cable including power supply and PE
- Connection signals: SMI+ (I+), SMI- (I-), L, N, PE usually 5x1.5mm²
- Connecting line type is installation cable without shielding
- maximum length of connection lines in total: 350m

- Communication speed is 2400 baud
- Data transfer is bidirectional with master/slave protocol

Wire sizes:

The recommendations are for guidance only. Other wire sizes and lengths may be used, depending on wiring rules and when appropriate. The connecting cable is a 5-wire cable, which supplies both power and transmits data. The connector of a SMI shade or blind motor has five terminals, labeled I+, I-, N, PE, and L, to which the cable is connected. Our SMI gateway uses SMI+ for the I+ connector and SMI- for the I- connector.

The maximum total cable length is 350 m, which means that distances of up to 350 m between the SMI gateway and to all shade or blind motors are possible.

Wire sizes (cross-sectional areas) from 0.75 mm² (18 AWG) to 2.5 mm² (14 AWG) can be used depending on the following factors:

- Total cable length
- Minimum voltage required by the blind motor
- Rated motor current
- Number of blind motors

Because up to 16 blind motors (drives) can be connected in parallel to one SMI gateway, you must consider the number of blind motors connected to the SMI blind module when calculating the dimension of the cables.

For example, blind motors that operate from mains voltage (230VAC) can typically handle a maximum voltage reduction of 10% (207VAC). When several shade or blind motors are connected in parallel to the SMI module, the cables must be dimensioned in a way that the voltage does not drop below 207VAC for the last shade or blind motor.

For more information, see the SMI Standard Motor Interface website www.standard-motor-interface.com

The topology of a SMI installation should be either a bus line or a star. See the picture below:

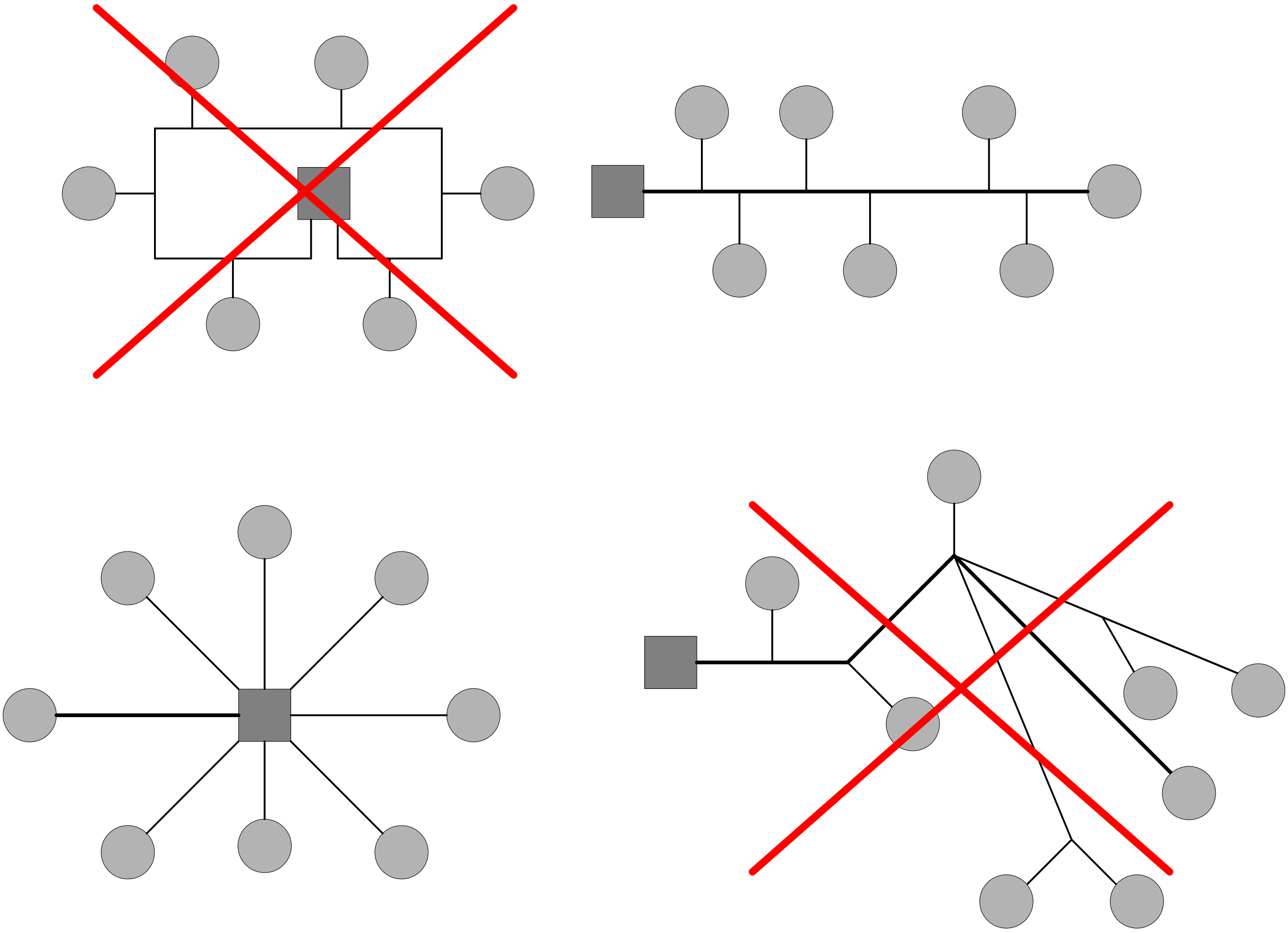
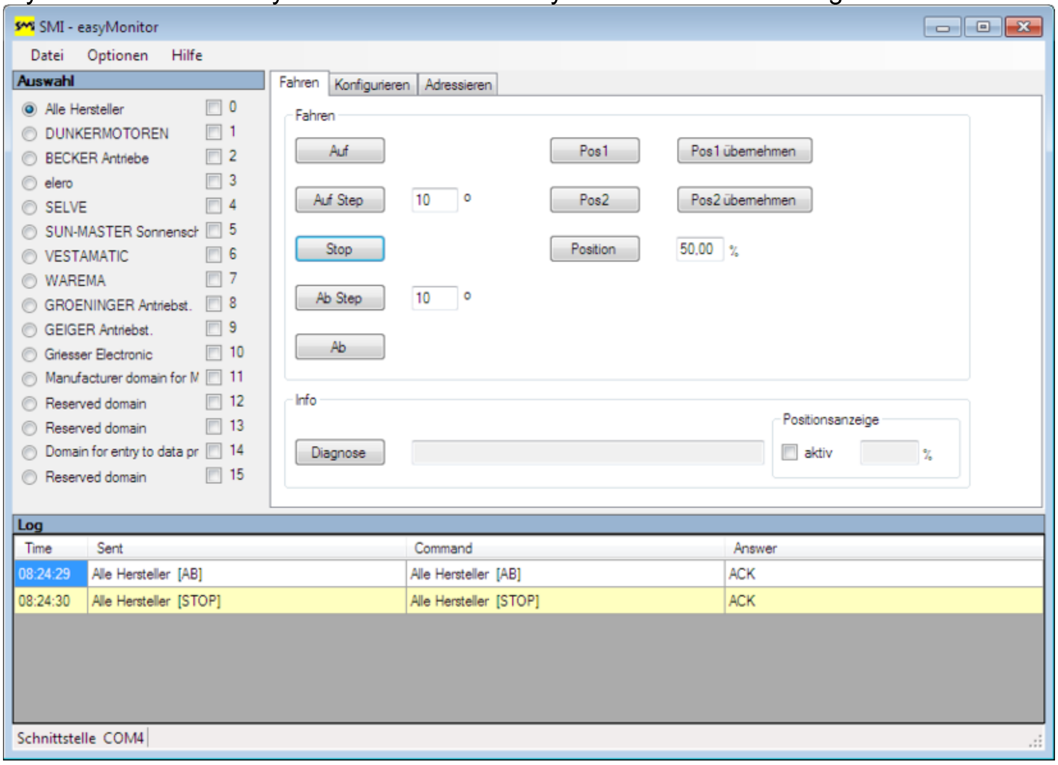


Figure: SMI bus topology

46.7 Commissioning of a SMI system

Download the free software SMI easy Monitor 3 from www.standard-motor-interface.com.

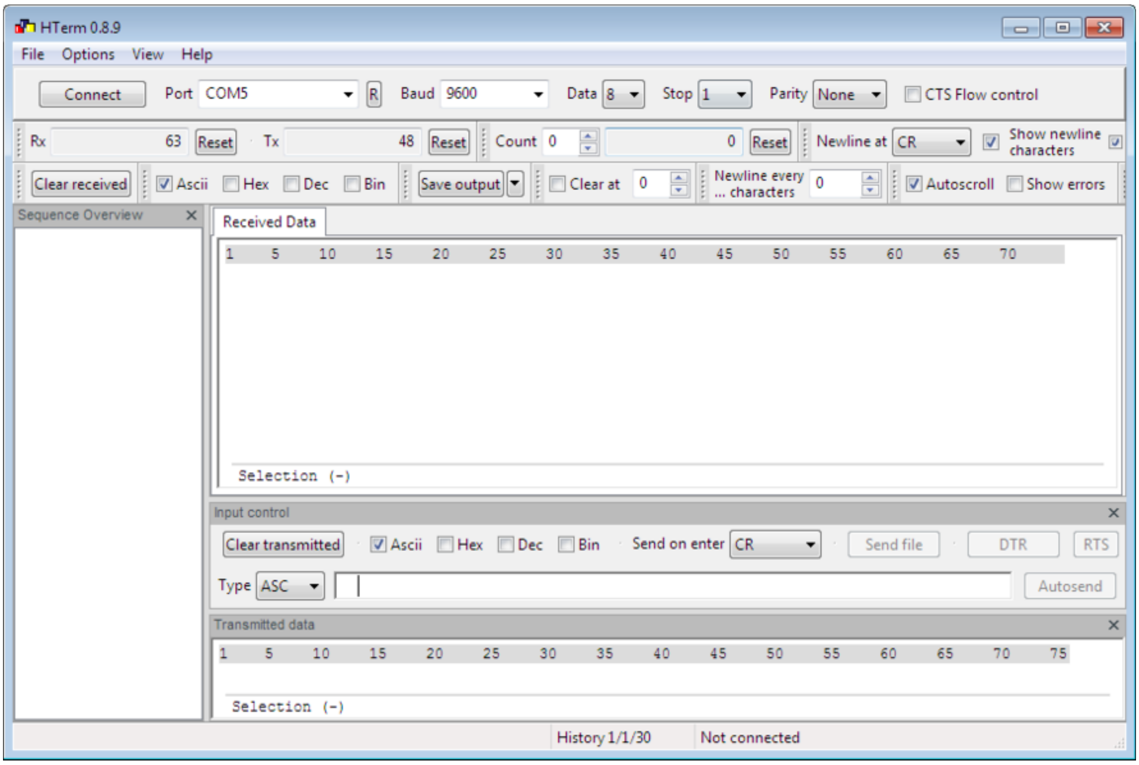
If you have successfully installed the software you will see the following screen:



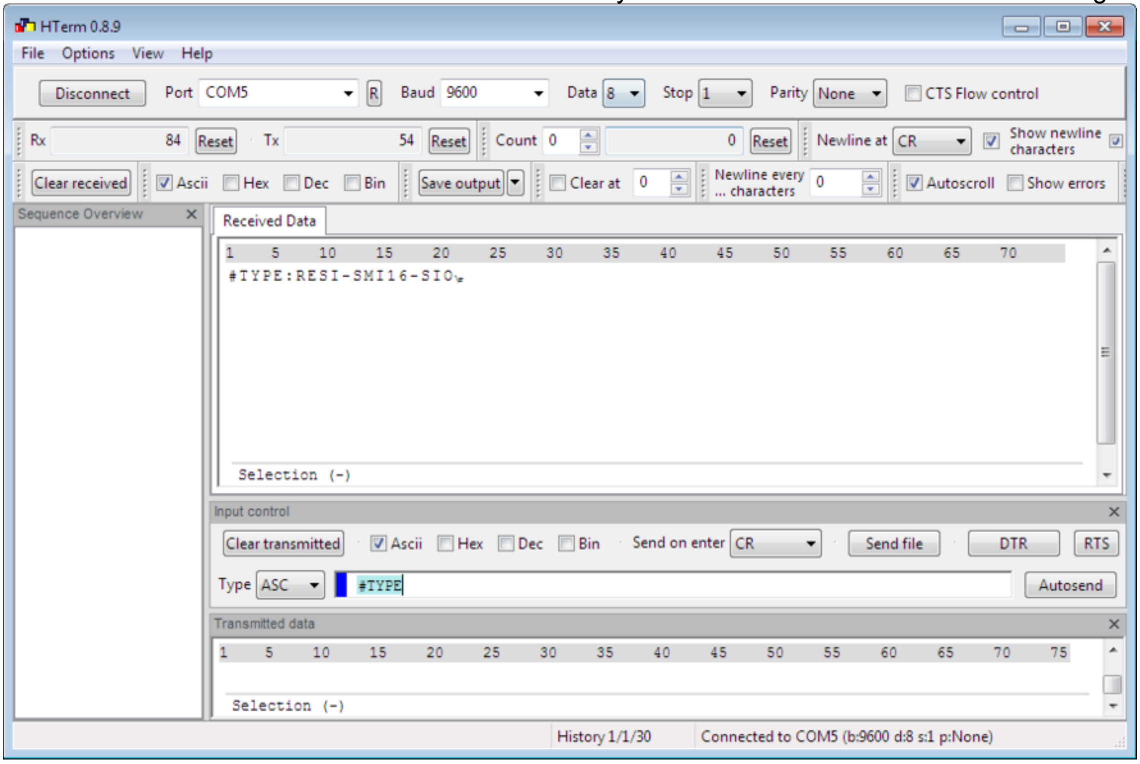
46.7.1 Use SMI easyMonitor 3 with RESI-SMIx-SIO

Usually our gateways use MODBUS/RTU slave protocol or ASCII text command for communication with the host system. You have to switch the gateway into a transparent mode with fixed baud rate of 2400Baud to communicate with the SMI easy Monitor software. To enable this, you have to use a terminal program like hterm, which allows you to send/receive ASCII text strings via serial line.

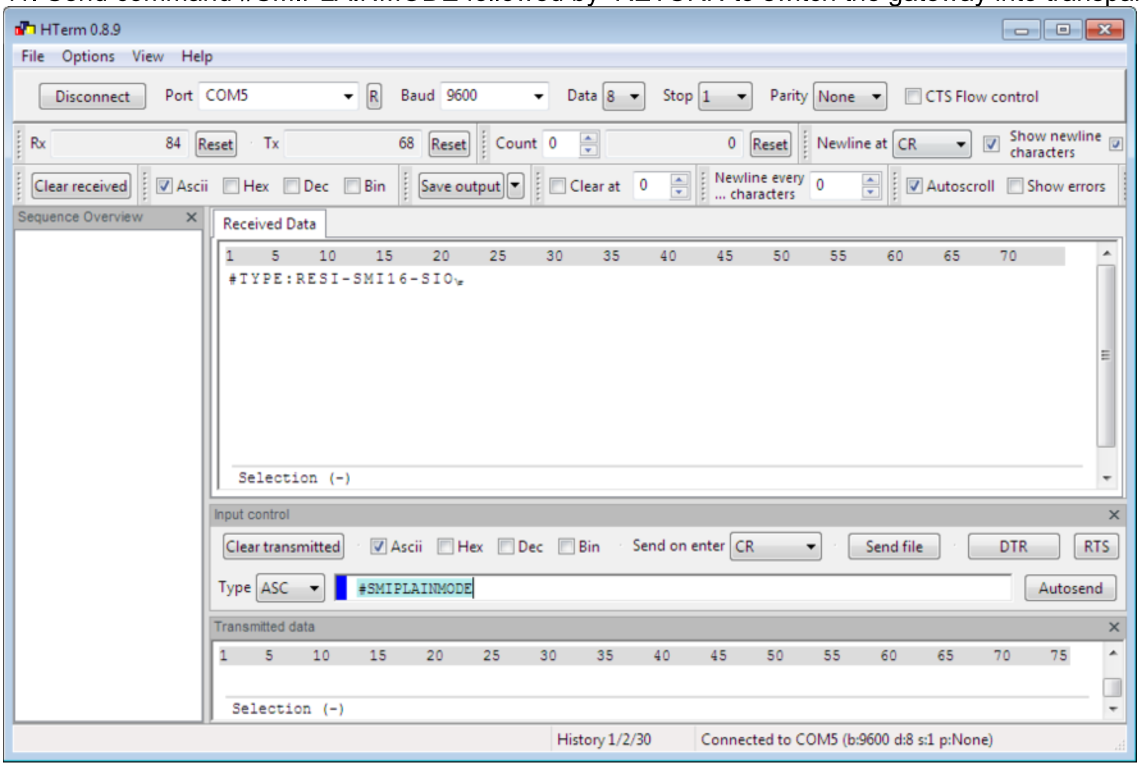
1. Close the SMI easyMonitor 3 software, because it will use the serial interface of your PC
2. Wire and connect our RESI-SMIx-SIO gateway to your serial interface (RS232 or RS485)
3. Put the DIP switches to 1-OFF, 2-OFF, 3-ON for RS485 or 3-OFF for RS232, 4-OFF
4. Open hterm software



5. Change the settings for Send on Enter to CR
6. Change the settings for Newline at to CR
7. Connect to the gateway: Select your serial interface baud rate 9600 and press connect button
8. To check the connection: Enter #TYPE followed by RETURN to send the command to the gateway.

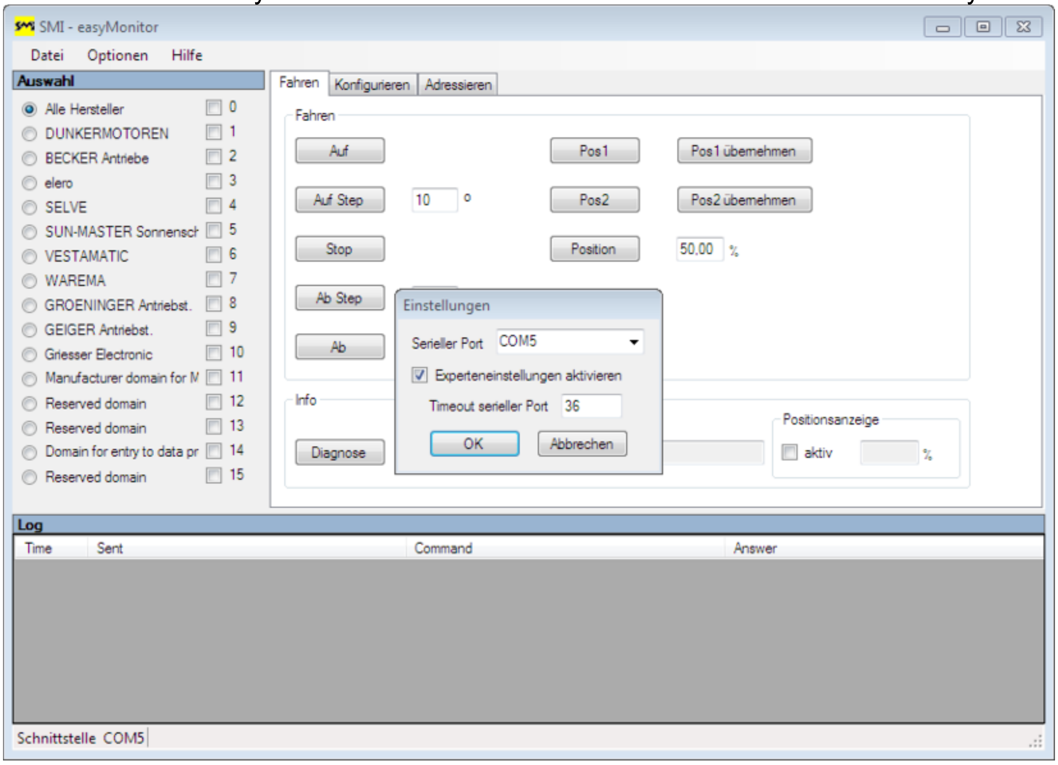


9. If everything is successfully connected, you will see the answer: #TYPE:RESI-SMI16-SIO above in the Received Data window.
10. Now we can switch to plain mode with 2400baud to use the gateway for the SMI easyMonitor 3 software.
11. Send command #SMIPLAINMODE followed by RETURN to switch the gateway into transparent mode.

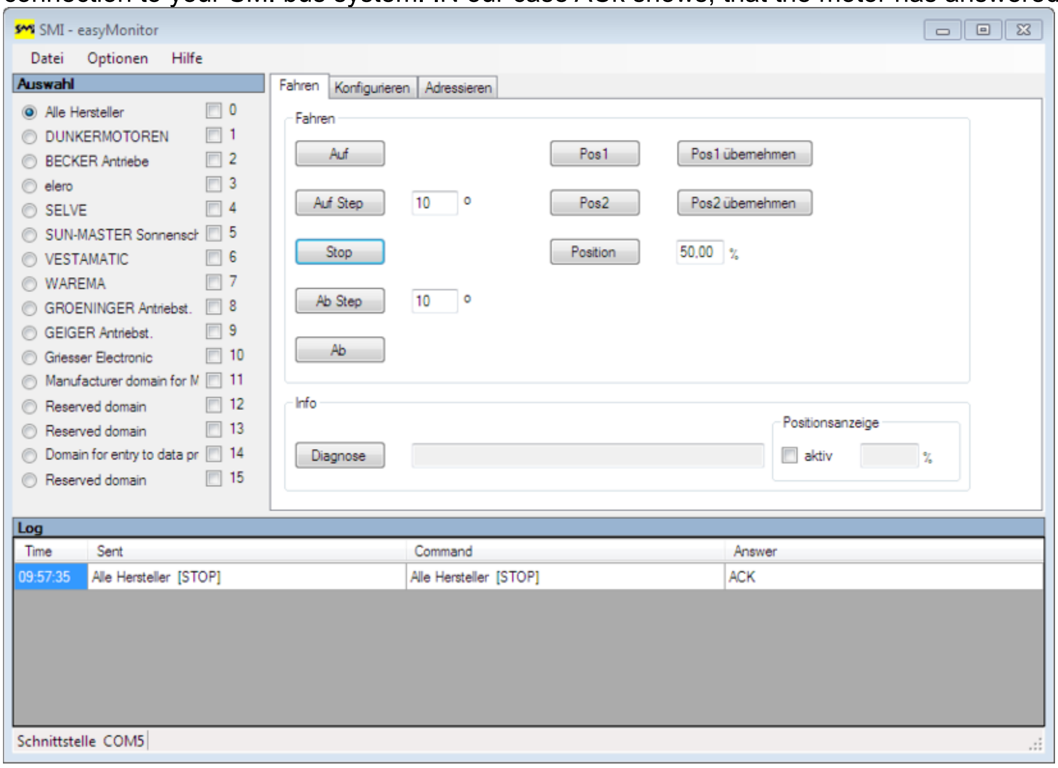


12. The gateway signals the successful switch with a fast flashing STATE LED.
13. Now close the hterm software

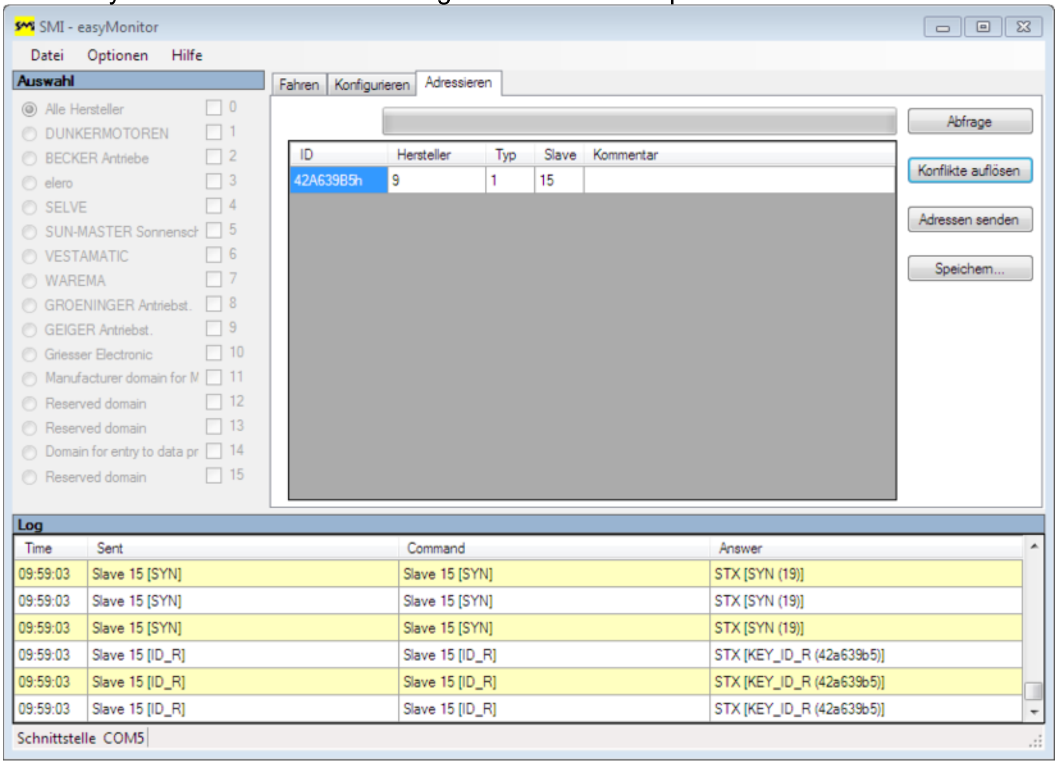
14. Start the SMI easyMonitor 3 software and select the correct serial interface of your PC



15. Send commands like STOP to all manufacturers and check the Log window. If there stands Timeout, you have no connection to your SMI bus system. IN our case ACk shows, that the motor has answered.



16. Now you can switch to Addressing tab and start a request of all available motors



17. Configure each SMI motor to a different Slave ID between 0 and 15, if you use our RESI-SMI16-SIO gateway, in case of the RESI-SMI8-SIO gateway configure Slave IDs between 0 and 7!

18. Test all of your configured motors with the SMI easyMonitor software to be sure, that the SMI bus system is ok.

When you are finished with this procedure close the software and re-power our gateway to stop the transparent mode and to start listening to MODBUS or ASCII protocol commands.

46.8 Additional MODBUS register & coils

Here you will find only the additional MODBUS registers and coils especially for this IO module. Please refer to the description of of the standard MODBUS mapping for more details about the available basic MODBUS registers and coils.

Please refer to the external document for detailed documentation of the current MODBUS register mapping for this IO module:

RESI-L-SMI8-SIO-ETH-MODBUS+ASCII-ENxx.pdf
RESI-L-SMI16-SIO-ETH-MODBUS+ASCII-ENxx.pdf

46.9 Additional ASCII commands

Here you will find only the additional ASCII commands especially for this IO module. Please refer to the description of of the standard commands for more details about the available basic ASCII commands.

Please refer to the external document for detailed documentation of the current ASCII commands for this IO module:

RESI-L-SMI8-SIO-ETH-MODBUS+ASCII-ENxx.pdf
RESI-L-SMI16-SIO-ETH-MODBUS+ASCII-ENxx.pdf