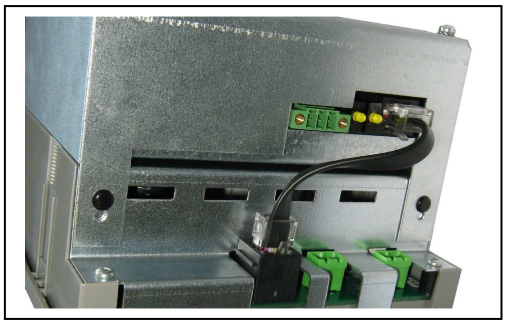
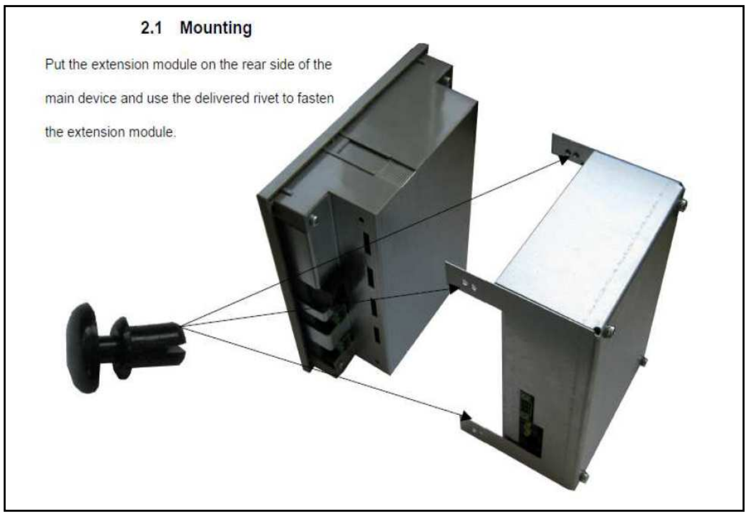
Boddingtons Pφwer Controls
Unit 1 Zone D Chelmsford Road I.E, DUNMOW CM6 1XG Essex
TEL: 01371 876543., fax: 01371 875460.
sales@boddingtonspowercontrols.co.uk

CX PLUS OR BLR CX WITH MODBUS





2.1 Mounting

Put the extension module on the rear side of the main device and use the delivered rivet to fasten the extension module.

Contents – see next page

Paragraph	Details
1	Overview
2	Fitting the modbus Extension Box on the CXPLUS
2.1	Mounting arrangements for the box
2.2	Connection
2.3	CXPLUS – MODBUS setup
3	Modbus / RS 485
3.1	The physical layer as decscribed in EIA485/ISO 8482
3.1.1	Line termination with resistors
3.1.2	Line biasing
3.1.3	Communication Indicator
3.2	MODBUS Protocol for the CXPLUS
3.2.1	MODBUS description
3.2.2	Serial Data Format and Framing
3.2.3	Serial Transmission Modes
3.2.4	Function Codes
3.2.5	Exception Codes
3.2.6	Master/Slave Protocol
3.2.7	Address Space
3.2.8	Measurement Values
3.2.9	Parameter Settings
3.2.10	Step Status
3.2.11	Device Status
3.2.12	Storage Settings
4	Trouble Shooting
5	Device Data
5.1	Technical Data
5.2	Drawings

IMPORTANT !

Where this sign appears in the text – this indicates an important point regarding safety of operation of the CXPLUS. If this point is not heeded , the relay may be damaged beyond repair and guarantee rights could be affected.
For more information on the MODBUS protocol see : **www.modbus.org**.

1 Overview

The MODBUS Extension of the BLR-CX offers the possibility to read values from the device and modify the settings of the device.

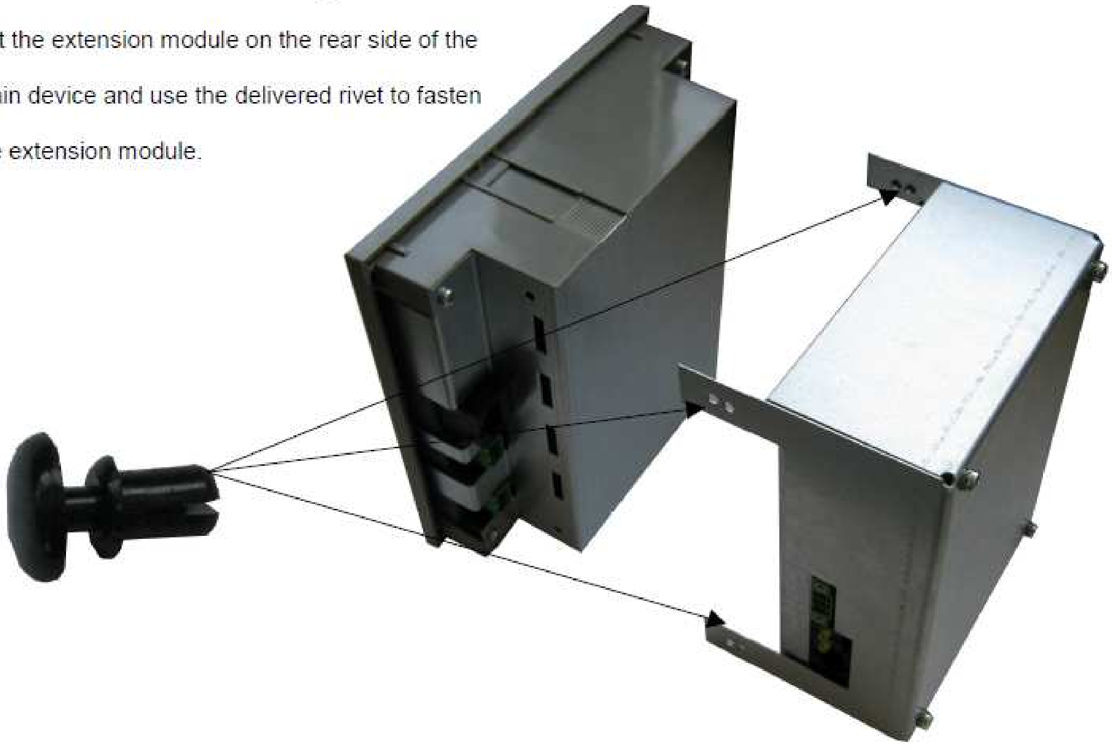
This document describes the transmission by use of the MODBUS-protocol. This protocol defines methods for data transmission and access control, but doesn't restrict the user to one single physical transmission system. In case of the BLR-CX, RS485 is used on the physical layer. As this is a bus-capable interface, it is possible to connect more than one BLR-CX to a single pair of wires and access the units by use of an ID number.

A lot of commercial devices and PLCs are able to use the MODBUS protocol, either as bus master or slave. Various SCADA solutions are also available from different vendors. So, the integration of the BLR-CX in an existing bus-system or setting up a new bus system is only a minor issue.

2 Mobus extension box

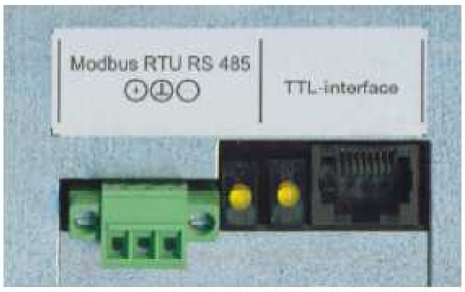
2.1 Mounting

Put the extension module on the rear side of the main device and use the delivered rivet to fasten the extension module.

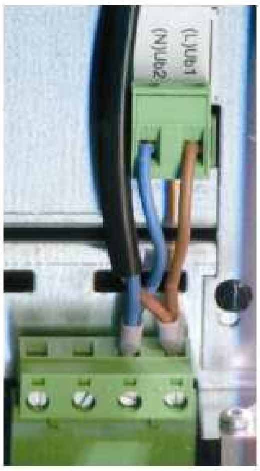


2.2 Connection

The BLR-CX uses a 3-pin connector. The connections can be seen in the picture. To use the MODBUS, one must connect the data lines + and – and the common ground (middle pin).

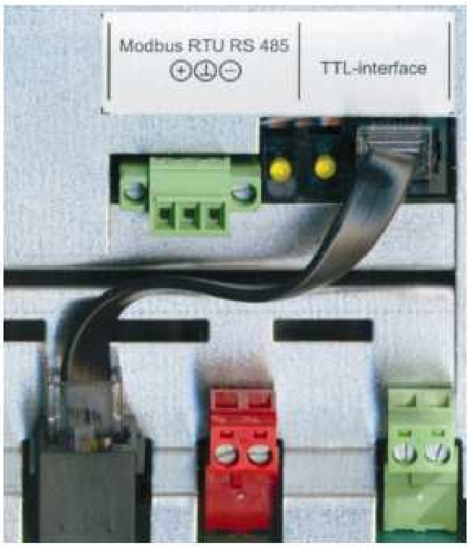


The power supply of the Modbus extension has the same voltage range as the main device. Due to this the supply voltage can be taken from the voltage terminals. Please consider the labels on the device.



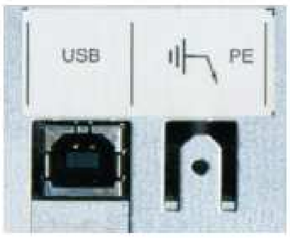
To establish the data communication between main device and extension box, use the RJ45 cable which comes with the extension box.

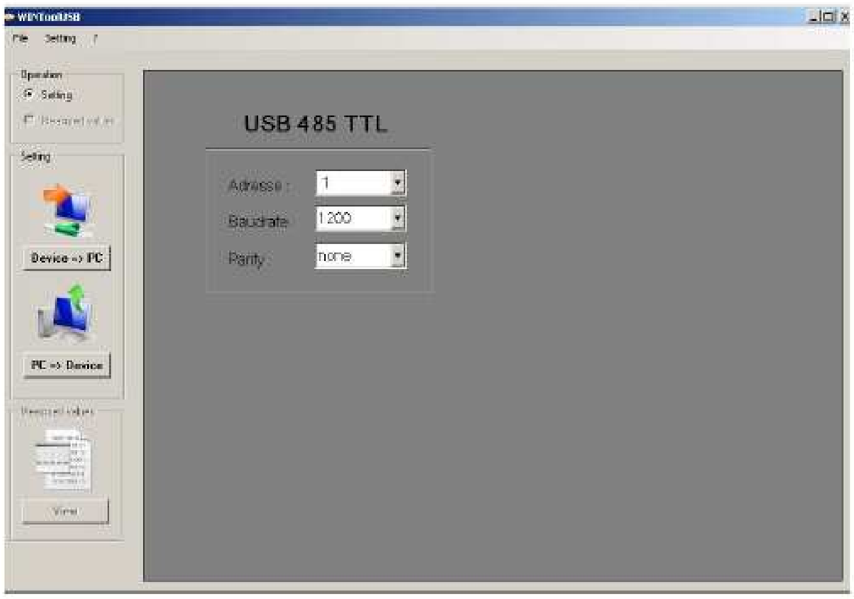
The LED next to the RJ45 plug on the communication module indicates a communication between module and extension box.



2.3 BLR-CX - MODBUS setup

If your device has MODBUS support, please connect the device via USB- cable to your PC. Install and open the software which comes with the device. After doing this steps you can adjust the following parameters:





USB 485 TTL

Adresse : 1

Baudrate 1200

Parity none

- ADDRESS: This is the devices slave address (slave ID). The valid range is 1-247.
- BAUD RATE: Select the baud rate here. The valid range is 1200 - 38400 baud.
- PARITY: Select the parity to be 8none2, 8even1 or 8odd1 (data bit/parity/stop bit).

The settings for baud rate and parity must be the same for all bus devices; the address must be unique for each device.

After doing the settings for the communication parameters click on the "PC->Device" button to send the settings to the device.

3 MODBUS / RS485

The implementation basically consists of two parts:

- The RS485 transmission is used for serial data transport. It is able to interconnect more than one device in a bus-like configuration. The RS485 protocol offers its "services" to the higher-level MODBUS protocol.
- The MODBUS protocol uses the underlying serial data transport layer (RS485 in this case) to communicate with several bus devices. It defines commands, address structures and data structures to access the slave device.

3.1 The physical layer - RS485 (defined in EIA485/ISO8482)

RS485 offers basic serial data transport to the higher-level MODBUS protocol layers. It is therefore called the "physical layer" of the bus system. Higher layers use the lower physical layer as a basic "service" for data transport.

RS485 uses two data wires for serial transmission. Each of them is driven to 0V or 5V by the transmitting device. The two data wires always have different voltage levels. One state (one wire 5V, other wire GND) represents the logic "OFF" state. The two wires exchange their voltages for the logic "ON" state. This differential transmission mode makes the RS485 bus very resistant against electro-magnetic distortions and therefore allows long transmission distances of more than 1000 metres.

The data transmission rates of the BLR-CX can be selected between 1,2k, 2,4k, 4,8k 9,6k, 19,2k, 38,4k, 57,6k and 115,2k baud. The parity can be selected between even, odd and no parity. All bus devices need to use the same settings. Standard settings are: 9600 baud and even parity.

There exist two different types of RS485:

- 2-wire RS485: This type uses only two data wires, which form one data channel. This means, that, after sending a request, the bus master has to deactivate its transmitter to make the data line free for the answering device. (Half-duplex mode)
- 4-wire RS485: this types uses one data line (=two wires) for the master->slave direction and another one (two more wires) for the slave ->master direction. **The BLR-CX does not support 4wire RS485!**

Both types, 2wire and 4wire, need another line to be connected, although it is not mentioned explicitly: the common ground GND. So, for the 2wire version you need a cable with 3 wires, for the 4wire version one with 5 wires! You should use a shielded cable, but never use the shield for GND connection. It should only be connected to protective ground to reduce electromagnetic influences.

The RS485 bus interconnects more than one device (typically up to 32). To accomplish this, the several data signals have to be interconnected for all bus devices. These are the two data lines and the common ground GND. All devices are connected to the bus in parallel. Avoid using taps, as they tend to be the source of transmission errors if they are too long. You should always prefer direct connection of the device to the main bus wire.

One bus cable with all its devices is called a “bus segment”. Several segments can be interconnected by using “repeaters”.

3.1.1 Line termination

One very important issue is the termination of the bus line. This is definitely needed for a working bus system to cancel out echoes from the line ends that would distort data signals. To terminate the bus cable, one must add a resistor at each end of the bus cable. The value of the resistor must match the cables impedance. At most times, 120 Ω is a good value to start with. Connect the termination resistor between data wires D(+) and D(-) at each end of the bus segment.

Some devices, especially bus converters have built-in resistors. Please check the manuals for all devices used on the bus. If these internal resistors cannot be disabled, this has a very important influence on your bus: you must place these devices at one of the ends of the bus! As the bus has only two ends, you can use only two devices with fixed resistors per bus segment!

3.2 The MODBUS protocol

3.2.1 MODBUS - description

The MODBUS protocol uses the RS485 as an underlying physical layer and implements the data transmission control mechanisms. It is therefore located on layer 2 ("link layer") of the OSI layer model for data exchange systems.

3.2.2 Serial data format and framing

The data is transmitted in fixed frames. The frames are separated by the bus being inactive for at least 3,5 characters. All data is organized in "protocol data units" (PDUs), which are transmitted over the serial bus system by the underlying physical protocol layer.

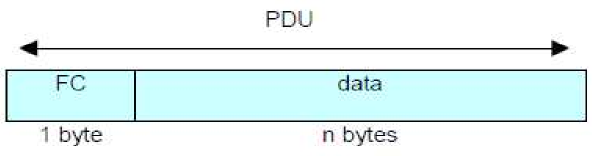


Illustration 1 : "Protocol Data Unit" - PDU

The PDU consists of two parts:

- The "function code" (FC) gives a command, which defines, what the slave unit has to do.
- The data block consists of the corresponding data to a FC. Its usage depends on the FC, it can contain pure data but also register addresses for slave data access.

The PDU defines a single data unit, which has to reach a certain bus device in order to perform a function. The transportation differs, dependent on the physical layer that is used.

To be able to control the transmission, the PDU is extended with additional blocks of data for transmission control purposes. For RS485, this extension results in the "application data unit" (ADU).

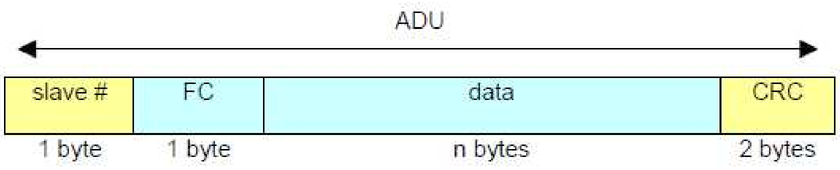


Illustration 2 : "Application Data Unit" - ADU

The application data unit, as it is used with the serial transmission over RS485, contains two additional blocks of data:

- The first field specifies the target for the data block, the "slave number" (=slave address)
- The transmission is additionally secured by a CRC16 error correction code.

3.2.3 Serial transmission modes

The protocol defines two different encodings for the frames' data contents. **The BLR-CX always uses the RTU-mode! ASCII mode is not implemented and is mentioned here only for the purpose of completeness.**

"Remote Terminal Unit" (RTU)

In this transmission mode every 8bit data word contains two 4-bit hexadecimal numbers. They are transmitted as one complete byte; a maximum transmission density is reached. With every data word, the following information is transmitted:

- 1 start bit
- 8 data bits, "least significant bit" first
- 1 parity bit (if set)
- 1 stop bit for parity even or odd / 2 if parity is none to compensate missing parity bit

"American Standard Code for Information Interchange" (ASCII)

In ASCII-mode, the two 4-byte nibbles of an 8bit data word are transmitted separately in ASCII-code representation. A data byte which contents $5B_{hex}$ will be divided into two parts and transmitted separately as one byte each. The result is that TWO data bytes are transmitted with contents 35_{hex} (=ASCII-Code "5") and 42_{hex} (=ASCII-Code "B"). This data mode is intended for compatibility reasons and makes debugging on the transmission line easier but it also decreases transmission speed significantly.

3.2.4 Function codes

As already mentioned, the data packet contains "function codes" which specify a command from the bus master to the bus slave. The slave executes the command (if possible) and then answers with the same function code in the reply to acknowledge the command. The valid range for function codes is specified from 1 to 127, but only a part of it is actually really used. Please refer to the MODBUS specifications for detailed Information. If it is impossibly for a slave to execute a command, it replies with an exception (=an error code). The function code of an exception packet is the function code of the received command, which caused the error, but it was changed in a certain way: The most significant bit is set by the slave to signal the error condition to the master. The contents of the data block specify the error in more detail.

The BLR-CX supports function codes 03_{hex} (read holding register), 04_{hex} (read input register) and 06_{hex} (write single register).

3.2.5 Exception codes

If a slave is not able to execute a command, which was sent by the master, it answers with exception codes. A full list of codes can be found in the MODBUS specification. We do not include this list here, because the master software will be able to handle most exceptions automatically. If one has to program the MODBUS master stack by himself he will need the full specifications, and with that, he gets the full list of error codes.

3.2.6 Master-Slave protocol

For communication, a master-slave protocol is used. Only the bus master is permitted to initiate a transfer. The "master" starts data exchange by sending a command to a slave by transmitting a data frame with the corresponding function code (=command) to the slave, which will then execute it.

- The unicast-mode is normally used to communicate on a Modbus system. One single slave is addressed by the slave number in the master's data packet. The valid address range is between 1 and 247. The slave then executes the command and answers by sending a acknowledge data packet back to the master.
- Not in any case the master can receive an answer to his query: in multicast-mode all slaves on the bus are addressed in parallel. They all execute the same command, but none of them will respond. The master initiates a multicast transfer by using "0" as slave number.

3.2.7 Address space

The data in the BLR-CX is organized and accessed by means of addresses. Each address accesses one data word. The data words are always 16bits wide.

The BLR-CX does not differentiate the addresses between the function codes. There is one big address space available and to access each address's data, any valid function code can be used. Nevertheless, the data will only make sense when interpreted the correct way!

The data can be of the following types:

- REAL: this is a 32 bit floating-point number, as defined in IEEE Standard 754.
- UINT16: this is an unsigned 16 bit integer value.
- UINT32, SINT32: this is an unsigned/signed 32 bit integer value.

As the data is organized in 16 bit wide words, a set of sequential addresses has to be read for longer data items. For these, the base address is given in the tables. To read a REAL with base address 12, one has to read two 16bit words from addresses 12 and 13. These two values need to be concatenated to form the desired result of 32 bits. Most SCADA software packages or PLCs can do this task for you.

There exist different types of addresses:

- **The MODBUS address always starts with 0 and can go up to 65535. It can be used with any function code.**
- **Certain PLCs lack correct handling of the 0 and therefore add 1 to the address. So their addresses (MODBUS address +1) always start with 1.**
- **Some SCADA tools add an offset to determine the function code, which shall be used to access the device at the given address. They also sometimes add 1 to the MOSBUS address. As an example, address 40001 would be "read MODBUS address 0 with function code 03$_{hex}$", 30012 would be "read MODBUS address 11 with function code 04$_{hex}$". Please refer to your software's manual to find out the correct addresses.**

  The following tables always give the MODBUS addresses mentioned first in above list.

3.2.8 Measurement values

The measured values are available beginning from address 0 in intervals of 2 data words.

If the current or voltage is too small to calculate valid harmonics from it, the value at the base address (= the fundamental) reads 0.0%. This indicates, that the higher harmonics for the current or voltage are also invalid!

All these values can be accessed with function codes 03_{hex} and 04_{hex}. **The values Apparent power S-sum, Active power P-sum, Reactive power Q-sum, Lacking reactive power ΔQ and Power factor (P/S) relate to a symmetrical power system.**

Address	Value	Words	Type	Unit
500	Meas. Voltage	2	Float	V
502	Current (including value for Q offset)	2	Float	A
504	Frequency	2	Float	Hz
506	Active power P-sum	2	Float	W
508	Reactive power Q-sum (including Q offset)	2	Float	Var
510	Apparent power S-sum	2	Float	VA
512	Lacking reactive power ΔQ (including Q offset)	2	Float	var
514	Cos φ	2	Float	-
516	Power factor (P/S)	2	Float	-
518	Tan φ	2	Float	-
520	Ambient temperature	2	Float	°C
522	Total harmonic distortion THD U	2	Float	%
524	Harmonics U 3. order	2	Float	%
526	Harmonics U 5. order	2	Float	%
528	Harmonics U 7. order	2	Float	%
540	Harmonics U 19. order	2	Float	%
542	Current (measured)	2	Float	A
544	Reactive power (measured)	2	Float	var
546	Average Power factor	2	Float	
548	Operation hours	2	Float	h

3.2.9 Parameter settings

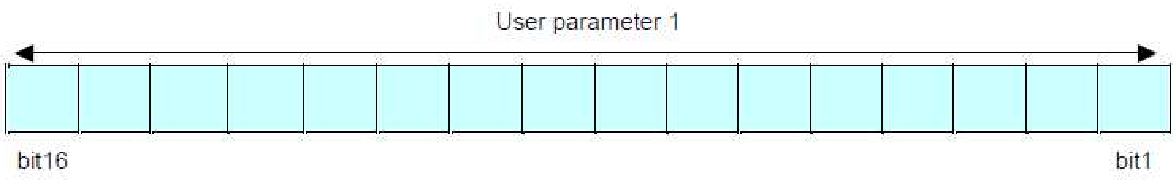
Parameters set by the user are stored in different data types. The base addresses and the data type can be found in the table below.

All these values can be accessed with function codes 03_{hex}, 04_{hex} and 06_{hex}.

Address	Value	Factor	Words	Type	Unit
102	PT ratio	10	1	UINT16	-
103	CT ratio	10	2	UINT32	-
105	Nominal voltage L – L	10	2	UINT32	-
107	Tolerance nominal voltage		1	UINT16	%
108	Phase - offset		1	SINT16	°
109	Control sensitivity	10	1	UINT16	-
110	Target cos φ 1 (0..100..200 = i0.00..1.00..c0.00)		1	UINT16	-
111	Target cos φ 2 (0..100..200 = i0.00..1.00..c0.00)		1	UINT16	-
112	Discharge time	10	1	UINT16	s
113	Switching time	10	1	UINT16	s
114	Switching time delay Step exchange	10	1	UINT16	s
115	asymmetry factor		1	UINT16	-
116	Max. switching cycles		1	UINT32	-
118	Max. operation hours		1	UINT16	h
119	Limit THD-U	10	1	UINT16	-
120	Delay time THD U and Temperature 2	10	1	UINT16	
121	Temperature limit 1	10	1	UINT16	°
122	Temperature limit 2	10	1	UINT16	°
123	Q offset	5	1	SINT16	var
124	Temperature offset		1	SINT16	°

At address 100 user parameter are collected which has no underlying an numerical value. At this point all user parameter are coded binary. Each single bit represents a adjustment in menu "Measurement" respective "Control". For this is the UINT16 value coded as follows.

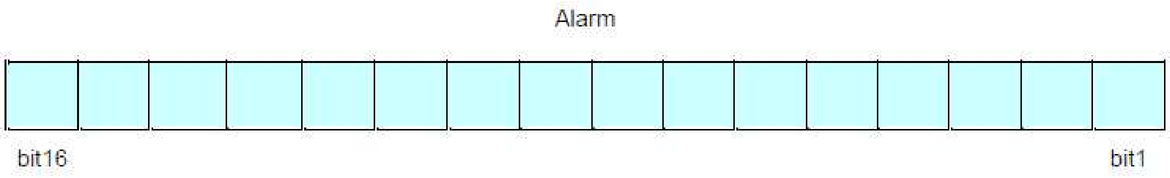
Address	Value	Words	Type	Unit
100	Control output	1	UINT16	-



Bit 1:	(1) = Connection Measurement L-L	(0) = connection measurement L-N
Bit 2:	(1) = Target cos φ 2 active	(0) = target cos φ 1 active
Bit 3:	(1) = Measurement FIX 50Hz	(if both set to 0, measurement is set to auto)
Bit 4:	(1) = Measurement FIX 60Hz	
Bit 5:	(1) = automatic Step size detection off	(0) = step size detection on
Bit 6:	(1) = Step exchange active	(0) = step exchange off
Bit 7:	(1) = Control off	(0) = control on or hold
Bit 8:	(1) = Lifo mode active	(0) = normal control mode
Bit 9:	(1) = Kombifilter active	(0) = normal control mode
Bit 10:	(1) = Progressiv	(0) = normal control mode
Bit 11:	(1) = AI countdown when restart	(0) = AI countdown off
Bit 12:	(1) = Target PF2 for P-Export	(0) = No tariff switch over
Bit 13	(1) = Target PF2 via DI	(0) = No tariff switch over
Bit 14:	(1) = DI active with closed input	(0) = DI active with open input
Bit 15:	(1) = reset step database	
Bit 16:	(1) = recognize defective steps	(0) = not recognize defective steps

At address 101 you can find the user parameters for Alarm, which has no underlying an numerical value. At this point all user parameter are coded binary. This UINT16 value coded as follows.

Address	Value	Words	Type	Unit
101	Control output	1	UINT16	-



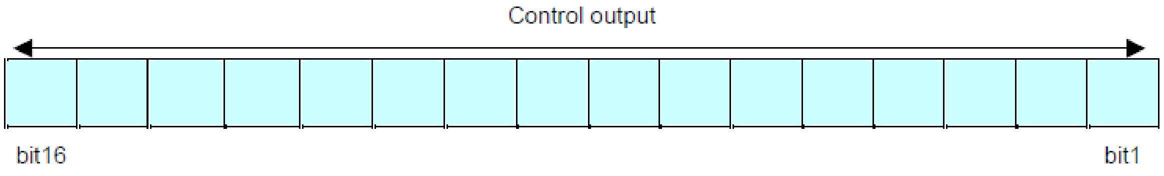
• Bit 1:	(1) = "THD U Alarm" enabled	(0) = "THD U Alarm" disabled
• Bit 2:	(1) = "THD Alarm Steps off" enabled	(0) = "THD Alarm Steps off" disabled
• Bit 3:	(1) = "Temperature Alarm" enabled	(0) = "Temperature Alarm" disabled
• Bit 4:	(1) = "Control Alarm" enabled	(0) = "Control Alarm" disabled
• Bit 5:	(1) = "Defective Step Alarm" enabled	(0) = " Defective Step Alarm" disabled
• Bit 6:	(1) = "Maintenance Alarm" enabled	(0) = "Maintenance Alarm" disabled
• Bit 7:	(1) = "I=0 freeze PCF" enabled	(0) = "I=0 freeze PCF" disabled
• Bit 8:	(1) = "Derating Alarm" enabled	(0) = "Derating Alarm" disabled
• Bit 9:	(1) = "Reset Userparameter"	
• Bit 10:	(1) = temp. input as DI	(0) = temp. input for temp. measurement
• Bit 11:	(1) = control on "Hold"	(0) = normal control
• Bit 12:	(1) = "AI Abrt Alarm" enabled	(0) = "AI Abrt Alarm" disabled
• Bit 13:	(1) = Reset Alarm manually	(0) = Automatic alarm reset
• Bit 14:	(1) = "Steps off when Q cap" enabled	(0) = "Steps off when Q cap" disabled
• Bit 15:	reserved	
• Bit 16:	reserved	

3.2.10 Step status

Information for every step are stored in a database. The referring information is available in different data types. For the following mentioned registers the bit assignment below is valid for the outputs:

The state of the used control outputs can be seen in the bit mask below. If the referring bit = 1 the output is active.

Address	Value	Words	Type	Unit
300	Control output	1	UINT16	-



- Bit 1: (1) = Relay output 1 active (0) = Relay output 1 inactive
- Bit 2: (1) = Relay output 2 active (0) = Relay output 2 inactive
- Bit 3: (1) = Relay output 3 active (0) = Relay output 3 inactive
- Bit 4: (1) = Relay output 4 active (0) = Relay output 4 inactive
- Bit 5: (1) = Relay output 5 active (0) = Relay output 5 inactive
- Bit 6: (1) = Relay output 6 active (0) = Relay output 6 inactive
- Bit 7: (1) = Relay output 7 active (0) = Relay output 7 inactive
- Bit 8: (1) = Relay output 8 active (0) = Relay output 8 inactive
- Bit 9: (1) = Relay output 9 active (0) = Relay output 9 inactive
- Bit 10: (1) = Relay output 10 active (0) = Relay output 10 inactive
- Bit 11: (1) = Relay output 11 active (0) = Relay output 11 inactive
- Bit 12: (1) = Relay output 12 active (0) = Relay output 12 inactive
- Bit 13: (1) = Relay output 13 active (0) = Relay output 13 inactive
- Bit 14: (1) = Relay output 14 active (0) = Relay output 14 inactive

The base addresses and the data types can be found in the table below.

All these values can be accessed with function codes 03$_{hex\ and}$ 04 $_{hex}$.

Address	Value	Words	Type	Units
200	Fix steps (1 = fix)	1	UINT16	-
201	Fix steps on/off (1 = on)	1	UINT16	-
202	Defective steps (1 = defective)	1	UINT16	-

All further addresses and data types for the other step information can be found in the table below. The values for the step sizes are based on nominal voltage.

Adress	Value	Factor	Words	Type	Unit
208	Current step size step 1	5	1	SINT16	var
209	Current step size step 2	5	1	SINT16	var
210	Current step size step 3	5	1	SINT16	var
211	Current step size step 4	5	1	SINT16	var
212	Current step size step 5	5	1	SINT16	var
	:				
	:				
218	Current step size step 11	5	1	SINT16	var
219	Current step size step 12	5	1	SINT16	var
220	Current step size step 13	5	1	SINT16	var
221	Current step size step 14	5	1	SINT16	var
222	Initial step size step 1	5	1	SINT16	var
223	Initial step size step 2	5	1	SINT16	var
224	Initial step size step 3	5	1	SINT16	var
225	Initial step size step 4	5	1	SINT16	var
226	Initial step size step 5	5	1	SINT16	var
	:				
	:				
232	Initial step size step 11	5	1	SINT16	var
233	Initial step size step 12	5	1	SINT16	var
234	Initial step size step 13	5	1	SINT16	var
235	Initial step size step 14	5	1	SINT16	var

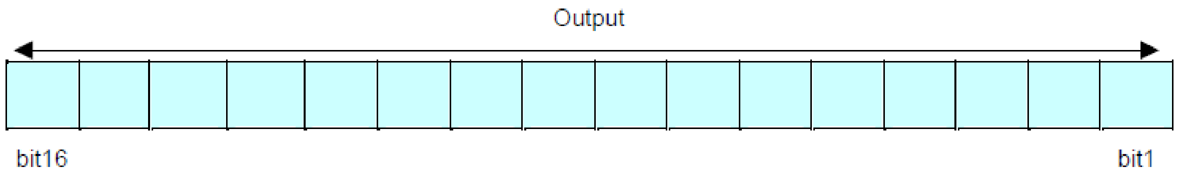
Adress	Value	Factor	Words	Type	Unit
236	Switching cycles step 1		1	UINT16	
237	Switching cycles step 2		1	UINT16	
238	Switching cycles step 3		1	UINT16	
239	Switching cycles step 4		1	UINT16	
243	Switching cycles step 8		1	UINT16	
244	Switching cycles step 9		1	UINT16	
246	Switching cycles step 11		1	UINT16	
247	Switching cycles step 12		1	UINT16	
248	Switching cycles step 13		1	UINT16	
249	Switching cycles step 14		1	UINT16	

3.2.11 Device status

The following mentioned registers contain information of alarms, messages and the status of the digital outputs. The assignment of the alarms can be seen in the bit mask below. If the referring bit = 1, the alarm is active.

All these values can be accessed with function codes $03_{\text{hex and}}$ 04_{hex}.

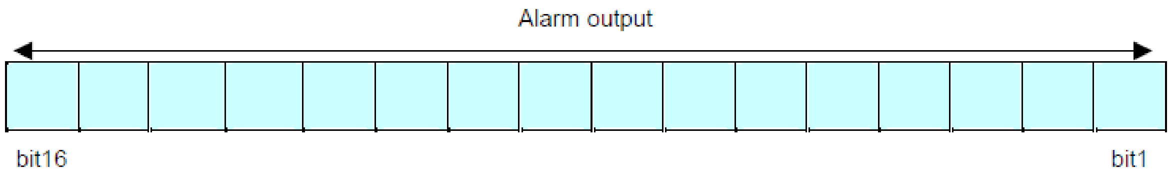
Address	Value	Words	Type	Unit
700	Alarm status	1	UINT16	-



- Bit 1: (1) = "Switching cycles" active (0) = "Switching cycles" normal
- Bit 2: (1) = "Operation hours" active (0) = "Operation hours" normal
- Bit 3: (1) = "Temperature Alarm 2" active (0) = "Temperature Alarm 2" normal
- Bit 4: (1) = "Temperature Alarm 1" active (0) = "Temperature Alarm 1" normal
- Bit 5: (1) = "Step derating Alarm" active (0) = "Step derating Alarm" normal
- Bit 6: (1) = "Defective Step Alarm" active (0) = "Defective Step Alarm" normal
- Bit 7: (1) = "THD Alarm" active (0) = "THD Alarm" normal
- Bit 8: (1) = "Control alarm" active (0) = "Control Alarm" normal
- Bit 9: (1) = "Over current alarm" active (0) = "Over current alarm" normal
- Bit 10: (1) = "Under current alarm" active (0) = "Under current alarm" normal
- Bit 11: (1) = "Voltage <> tolerance alarm" active (0) = "Voltage <> tolerance alarm" normal
- Bit 12: reserved
- Bit 14: reserved
- Bit 15: reserved
- Bit 16: reserved

The assignment of the alarm reactions can be seen in the bit mask below. If the referring bit = 1, the output or the message is active.

Address	Value	Words	Type	Unit
300	Alarm output	1	UINT16	-



- Bit 15: (1) = Alarm relay normal state (0) = Alarm relay fault state
- Bit 16: reserved

All settings send by MODBUS are used immediately, but remember that this information is only stored in the working memory. After a power blackout these settings will be lost. To store the settings durable, you have to store the data in the non-volatile memory.

4 Trouble shooting

If the bus connection isn't working correctly, please check the following points:

1. If there is no communication at all, then the error must be looked for between the BLR-CX and the PC!

   Possible causes can be:

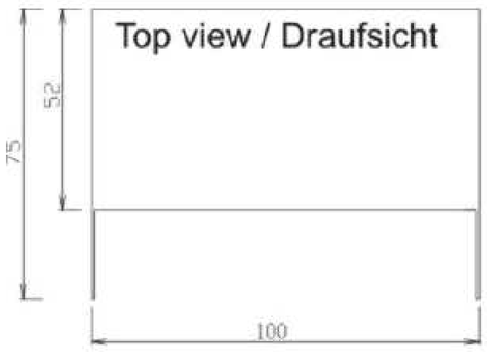
   - Check adjustment of baud rate, parity and address at the BLR-CX, possibly make changes in the configuration
   - Possibly the bus lines A and B are interchanged, if necessary rectify
   - Check adjustments of the converter RS485/RS232, possibly use the data sheet of the converter
   - Perhaps the port is already used by another application, if necessary stop this multiple reservation
   - Check termination and bias resistors, if necessary rectify
2. Does the cable of the bus connection have any damages? All plug connections are correct? If necessary replace.
3. Is the pin assignment of the RS485 connection correct? If necessary rectify.
4. The shielding of the bus line must not be connected with the ground of the bus. But the shielding should be connected to protective ground. If necessary rectify.
5. Is the communication possible, but there are problems with the software of the customer, then please check the following points:
   - Check adjustment of bus address, parity and baud rate in the software
   - Check data format

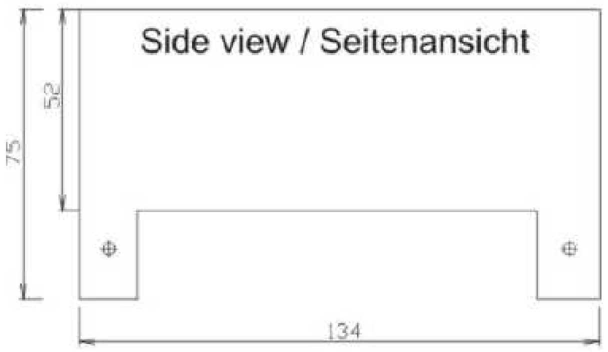
5 Device Data

5.1 Technical details

Voltage:	90-550V, 1ph., 50/60Hz, 2VA
Ambient temperature:	operation: -20°V - 70°C
	storage: -40°C - 85°C
Humidity:	0% - 95%, without condensation
Overvoltage:	II, dirt class 3
Protection:	rear: IP20
Weight:	ca. 0,4kg

5.2 Drawings


Top view / Draufsicht
75
52
100


Side view / Seitenansicht
75
52
134