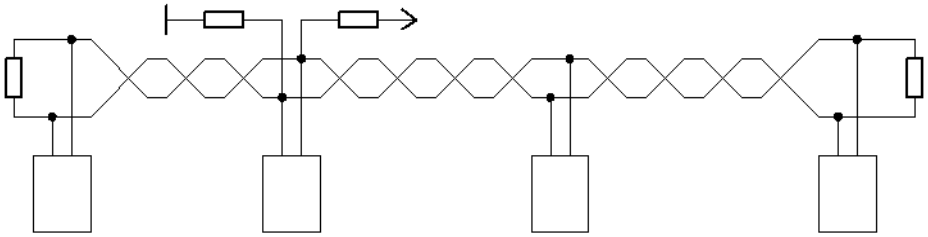
Technical Documentation



KSR-Z MODBUS

Beluk GmbH
Taubenstrasse 1
D-86956 Schongau
Germany

Tel.: +49/(0)8861/2332-0
Fax: +49/(0)8861/2332-22
E-mail: blr@beluk.de
Web: http://www.beluk.de

Document history

Date	Name	Revision	Change
21.12.04	CE	01	initial document release
17.03.05	ATh	02	features of new software revision (1.06) added to manual, general updates
11.05.05	ATh	03	general updates and settings via MODBUS added
10.01.06	ATh	04	general updates
21.09.07	Le	05	New connector on MODBUS hardware
16.02.12	Le	06	Add address space faultrecorder
11.12.15	ChP	07	Correction and update of the address space
18.07.18	SO	08	Content correction

Contents

Important Information!

If the above sign appears besides a text passage in the manual the reader is strongly advised to read the corresponding information, as it may be very important for usage of the device.

It can contain safety advice or other information for the correct handling of the device.

If the information is disregarded, the device may be inoperable or even damaged!

Additional documentation for the MODBUS protocol can be found at www.modbus.org.

The MODBUS standards are also available from there.

1 Overview

The MODBUS Extension of the KSR-Z 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 KSR-Z, RS485 is used on the physical layer. As this is a bus-capable interface it is possible to connect more than one KSR-Z to a single pair of wire 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 KSR-Z in an existing bus-system or setting up a new bus system is only a minor issue.

2 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.

2.1 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 KSR-Z can be selected between 1200, 2400, 9600, 19200 or 38400 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 KSR-Z does not support 4-wire 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.

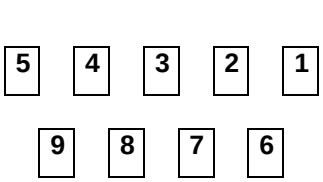
The RS485 bus interconnects more than one device (typically up to 32). To accomplish this, the several data signals have to be interconnected for all bus devices. These are the two data lines and the common ground GND. All devices are connected to the bus in parallel. Avoid using taps, as they tend to 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.

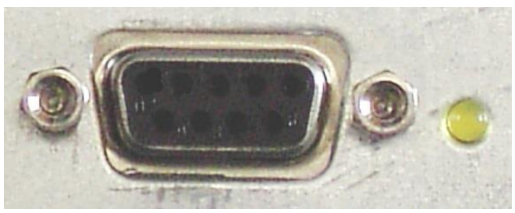
One bus cable with all its devices is called a “bus segment”. Several segments can be interconnected by using “repeaters”.

2.1.1 Connection

The MODBUS interface exists in two variants:

a) connection with 9-pin sub-d





PIN1 +5V (only for data line bias, **do never supply any other external circuits from this voltage output!**)

PIN2 GROUND for biasing and as common ground for all bus participants.

PIN5 D (B) - data signal B

PIN9 D (A) - data signal A

b) connection with 3-pin connector

This connection variant 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).

2.1.2 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, which 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 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!

2.1.3 Line biasing

Another important issue is line biasing. If no device is actually transmitting, the data wires are left at floating. Because of the termination resistors, both will have nearly the same voltage. This could result in spurious data signals because of external influences. Line biasing is used to give the data wires a defined “off”-state in this case.

Two resistors of approx. 500-600Ω have to be connected from D(+) line to +5 V and from D(-) to GND. The two bias resistors are needed only once per bus, the position of the bias resistors is not important. They may be placed anywhere on the bus, even in the “middle”. Please check in the manuals of all bus devices, if resistors are internally provided!

The KSR-Z has the voltages 5V and GND available on its bus connector, so these two resistors can be soldered inside the connector case.

2.1.4 Communication Indicator

The yellow LED on the backside of the device indicates an active transmission. It flashes only, if the device is actually communication with the bus master.

The communication indicator LED is available for both connector variants.

2.2 The MODBUS protocol

2.2.1 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.

2.2.2 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.

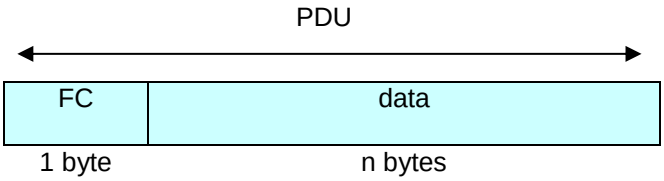


Illustration 1 : "protocol data unit" - PDU

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).

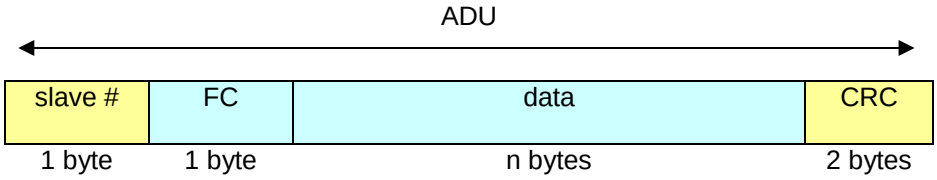


Illustration 2: "application data unit" - ADU

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.

2.2.3 Serial transmission modes

The protocol defines two different encodings for the frames' data contents. **The KSR-Z 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.

2.2.4 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 255, 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 KSR-Z supports function codes 03_{hex} (read holding register), 04_{hex} (read input register), 06_{hex} (write single register) and 10_{hex} (write multiple register).

2.2.5 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.

2.2.6 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 data packet as acknowledge 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. A multicast transfer is initiated by the master by using "0" as slave number.

2.2.7 KSR-Z - MODBUS setup

If your device has MODBUS support, an additional entry is available in the "setup" - menu of the device: **setup -> modbus**. After entering it, you will have the possibility to select the following items:

- **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 **8E1** (**EVEN**), **8O1** (**ODD**) or **8N2** (**NONE**).

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

2.2.8 Address space

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

The KSR-Z 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 32bit floating-point number, as defined in IEEE Standard 754
- UINT16: this is a 16bit unsigned integer value
- SINT16: this is a 16bit signed integer value
- SINT32: this is a 32bit signed integer value
- LONG: this is a 32bit integer value
- Struct: this is a predefined structure

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 are always (MODBUS address +1) and 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.

2.2.9 Measured values

The measured values are available beginning from address 0 in intervals of 6 data words. For each value in this table, the maximum and minimum values are also available.

To read the maximum, just add 2 to the address of a value, for the minimum add 4. (Example: to get the minimum voltage L1-N, read from address 00034 = 00030+4). All these values can be accessed with function codes 03_{hex} and 04_{hex}.

Address	Value	Words	Type	Unit
00000	Current I-L1	2	REAL	A
00006	Current I-L2	2	REAL	A
00012	Current I-L3	2	REAL	A
00018	Current I-UB	2	REAL	A
00024	Voltage V-L1N	2	REAL	V
00030	Voltage V-L2N	2	REAL	V
00036	Voltage V-L3N	2	REAL	V
00042	Voltage V-L12	2	REAL	V
00048	Voltage V-L23	2	REAL	V
00054	Voltage V-L31	2	REAL	V
00060	Fundamental current If-L1	2	REAL	A
00066	Fundamental current If-L2	2	REAL	A
00072	Fundamental current If-L3	2	REAL	A
00078	Fundamental current If-UB	2	REAL	A
00084	Total harmonic distortion THD-I1	2	REAL	%
00090	Total harmonic distortion THD-I2	2	REAL	%
00096	Total harmonic distortion THD-I3	2	REAL	%
00102	Total harmonic distortion THD-IUB	2	REAL	%
00108	Total harmonic distortion THD-V1N	2	REAL	%
00114	Total harmonic distortion THD-V2N	2	REAL	%
00120	Total harmonic distortion THD-V3N	2	REAL	%
00126	Ambient temperature T	2	REAL	°C
00132	Damped current Ith-L1	2	REAL	A
00138	Damped current Ith-L2	2	REAL	A
00144	Damped current Ith-L3	2	REAL	A
00150	Damped current Ith-UB	2	REAL	A
00156	Unbalanced current compensated I-UC4	2	REAL	A
00162	Earth fault current I-EF	2	REAL	A
00168	Earth fault current compensated Icp-EF	2	REAL	A

Beluk GmbH
Taubenstrasse 1
D-86956 Schongau
Germany

All these values are mapped to another structure with includes only the actual meas values without the minimum and maximum values. It starts at adress **2816**. For reading the next value to the current address must be add 2.

Address	Value	Words	Type	Unit
02816	Current I-L1	2	REAL	A
02818	Current I-L2	2	REAL	A
02820	Current I-L3	2	REAL	A
02822	Current I-UB	2	REAL	A
02824	Voltage V-L1N	2	REAL	V
02826	Voltage V-L2N	2	REAL	V
02828	Voltage V-L3N	2	REAL	V
02830	Voltage V-L12	2	REAL	V
02832	Voltage V-L23	2	REAL	V
02834	Voltage V-L31	2	REAL	V
02836	Fundamental current If-L1	2	REAL	A
02838	Fundamental current If-L2	2	REAL	A
02840	Fundamental current If-L3	2	REAL	A
02842	Fundamental current If-UB	2	REAL	A
02844	Total harmonic distortion THD-I1	2	REAL	%
02846	Total harmonic distortion THD-I2	2	REAL	%
02848	Total harmonic distortion THD-I3	2	REAL	%
02850	Total harmonic distortion THD-IUB	2	REAL	%
02852	Total harmonic distortion THD-V1N	2	REAL	%
02854	Total harmonic distortion THD-V2N	2	REAL	%
02856	Total harmonic distortion THD-V3N	2	REAL	%
02858	Ambient temperature T	2	REAL	°C
02860	Damped current Ith-L1	2	REAL	A
02862	Damped current Ith-L2	2	REAL	A
02864	Damped current Ith-L3	2	REAL	A
02866	Damped current Ith-UB	2	REAL	A
02868	Unbalanced current compensated I-UC4	2	REAL	A
02870	Earth fault current I-EF	2	REAL	A
02872	Earth fault current compensated Icp-EF	2	REAL	A

Beluk GmbH
Taubenstrasse 1
D-86956 Schongau
Germany

2.2.10 Harmonics

Harmonics are stored in separate arrays of FLOAT values for each current / voltage. The table below gives the corresponding base addresses. Each data array contains 25 values with 2 words length each. The first table entry holds the **3rd harmonic** (=harmonic of order 3). All values are normalized to 100%=fundamental. The other harmonics follow in increasing order up to the **51th harmonic**. **Note**: The table contains only the **ODD harmonic orders** (**3, 5, ..., 49, 51**). Remember that each FLOAT value occupies 2 words, so always increase the address by 2 for the next value.

If the current or voltage is too small to calculate valid harmonics from it, the value at the base address reads 0.0% instead of xx.x%. 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}.

Address	Value	Words	Type	Unit
1538 (0602_{hex})	Base address for harmonics I - L1	25*2	REAL	%
1592 (0638_{hex})	Base address for harmonics I - L2	25*2	REAL	%
1646 ($066E_{hex}$)	Base address for harmonics I - L3	25*2	REAL	%
1700 ($06A4_{hex}$)	Base address for harmonics I - N	25*2	REAL	%
1754 ($06DA_{hex}$)	Base address for harmonics V - L1-N	25*2	REAL	%
1808 (0710_{hex})	Base address for harmonics V - L2-N	25*2	REAL	%
1862 (0746_{hex})	Base address for harmonics V - L3-N	25*2	REAL	%

2.2.11 Parameter settings

The parameters of the device can be read via MODBUS. The parameters are stored from address 512 in UINT16 format. The table below gives the corresponding addresses.

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

Address	Value	Words	Type	Unit
512	Parameter for ct-I123 ratio	1	UINT16	-
513	Parameter for vt ratio	1	UINT16	-
514	Parameter for ct-IUB ratio	1	UINT16	-
515	Parameter for thermic TAU	1	UINT16	-
517	User parameter flags [1])	1	UINT16	
522	Stage number	1	UINT16	-

The following illustration shows the user parameter flag mask.

User parameter flags

x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	DI

bit16 bit1

DI: “0” represents that digital input is HIGH active, “1” means LOW active.

X Reserved.

[1]) see bit mask description for user parameter flags

2.2.12 Relay settings

The behaviour of the output relays can be read via MODBUS. It is stored in a binary bit mask, which is available from address 769 in UINT16 format. The least significant bit represents relay 1, the next relays follow in the bits with higher significance. In the case of relay inverting “0” means normally open, “1” means normally closed. In the case of relay reset mode “0” means automatic reset, “1” means manual reset. In the case of relay blocked by Digital Input “0” means relay will not blocked by digital input, “1” means relay can be blocked by digital input.

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

Address	Value	Words	Type	Unit
769	Relay normally open / closed	1	UINT16	-
770	Relay manual / automatic reset	1	UINT16	-
772	Relay blocked by Digital Input	1	UINT16	-

2.2.13 Alarm settings

The alarm settings are stored in separate structs for each alarm. The table below gives the corresponding base address for each alarm. All these values can be accessed with function codes 03_{hex} and 04_{hex}.

Address	Value	Words	Type	Unit
1024	Base address Overvoltage Alarm	9	Struct	-
1033	Base address Overvoltage Trip	9	Struct	-
1042	Base address Undervoltage Alarm	9	Struct	-
1051	Base address Undervoltage Trip	9	Struct	-
1060	Base address Overload Alarm	9	Struct	-
1069	Base address Overload Trip	9	Struct	-
1078	Base address Underload Alarm	9	Struct	-
1087	Base address Underload Trip	9	Struct	-
1096	Base address Damped current Overload Alarm	9	Struct	-
1105	Base address Damped current Overload Trip	9	Struct	-
1114	Base address Fundamental current Overload Alarm	9	Struct	-
1123	Base address Fundamental current Overload Trip	9	Struct	-
1132	Base address Unbalance Alarm	9	Struct	-
1141	Base address Unbalance Trip	9	Struct	-
1150	Base address Earth fault Alarm	9	Struct	-
1159	Base address Earth fault Trip	9	Struct	-
1168	Base address Voltage asymmetric Alarm	9	Struct	-
1177	Base address Voltage asymmetric Trip	9	Struct	-

Each alarm data consists of a struct containing 7 values composed of different data types. Every alarm is organized in the same way. The table below shows the content of an alarm.

Address	Value	Words	Type	Unit
Base address + 0	Alarm / Trip setting: disabled / enabled [2])	1	UINT16	-
Base address + 1	Limit	2	SINT32	-
Base address + 3	T-On x 100	1	UINT16	s
Base address + 4	T-Off x 100	1	UINT16	s
Base address + 6	Flags [3])	1	UINT16	-
Base address + 7	Alarm / Trip characteristic [4])	1	UINT16	-
Base address + 8	T-Reset x 100	1	UINT16	s

If an alarm / trip setting is enabled or not will be shown in a binary bit mask at the base address of each alarm in UINT16 format. See following tables and illustrations for the alarm / trip settings.

Note: The values for T-On, T-Off and T-Reset will be read with the factor 100.

[2]) see illustration for alarm / trip settings

[3]) see illustration for alarm / trip flags

[4]) see table for alarm / trip characteristics

The following illustration shows the alarm / trip setting mask.

Alarm / Trip settings

ALARM ENABLED	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x

bit16 bit1

ALARM ENABLED: “0” represents that the alarm / trip is disabled,

“1” means that the alarm / trip is enabled.

X: Reserved.

The following illustration shows the mask of the Alarm / Trip flags.

Alarm / Trip flags

BLK DI	FR	DISP RES	x	x	CHAR	x	DISP	R8	R7	R6	R5	R4	R3	R2	R1

bit16 bit1

R1 – R8: Output relay 1 – 8: “0” represents relay inactive if trip or alarm occurs, relay active if “1”.

DISP: Output display: “0” represents display message is inactive if trip or alarm occurs,

“1” represents display message is active.

CHAR: “0” represents the set T-ON and T-OFF delays are used

“1” represents the set characteristic curve is used.

DISP RES: “0” represents automatic reset of the display message.

“1” represents manual reset of the display message.

FR: “1” represents that the alarm / trip event will be stored into the faultrecorder,

“0” means this function is disabled.

BLK DI: “1” represents that the alarm / trip can be blocked by the digital input,

“0” means this function is disabled.

X: Reserved.

The following table shows the code for the different alarm / trip characteristics.

Code	Alarm / Trip characteristic	Type	Factor
1	U1 = U.S. Moderately inverse	UINT16	-
2	U2 = U.S. Inverse	UINT16	-
4	U3 = U.S. Very inverse	UINT16	-
8	U4 = U.S. Extremely inverse	UINT16	-
16	U5 = U.S. Short-time inverse	UINT16	-
32	C1 = I.E.C. Class A – Standard inverse	UINT16	-
64	C2 = I.E.C. Class B – Very inverse	UINT16	-
128	C3 = I.E.C. Class C – Extremely inverse	UINT16	-
256	C4 = I.E.C. Long-time inverse	UINT16	-
512	C5 = I.E.C. Short-time inverse	UINT16	-

2.2.14 Read out fault recorder

The following table shows the base address for all stored values. **X** represents the storage number: **0...63**
All these values can be accessed with function codes 03_{hex} and 04_{hex}.

Address	Value	Words	Type	Unit
10000+28*X	1 = on / 0 = off event	1	UINT16	-
10001+28*X	year_month of event	1	hex	-
10002+28*X	day_hour of event	1	hex	-
10003+28*X	min_sec of event	1	hex	-
10004+28*X	source (number between 0...28 see table 2)	1	SINT16	-
10005+28*X	limit	2	Float	
10007+28*X	max. value	2	Float	
10009+28*X	time delay	2	Float	s
10011+28*X	flags (see table 3 for meaning of the bits)	1	UINT16	-
10012+28*X	Voltage L12	2	Float	V
10014+28*X	Voltage L23	2	Float	V
10016+28*X	Voltage L31	2	Float	V
10018+28*X	Current I-L1	2	Float	A
10020+28*X	Current I-L2	2	Float	A
10022+28*X	Current I-L3	2	Float	A
10024+28*X	Current I-UB	2	Float	A

Table 2: Alarm / Trip sources.

Source	Meaning	Source	Meaning
0	I-L1	18	THD-V1N
1	I-L2	19	THD-V2N
2	I-L3	20	THD-V3N
3	I-UB	21	T
4	V-L1N	22	Ith-1
5	V-L2N	23	Ith-2
6	V-L3N	24	Ith-3
7	V-L12	25	Ith-UB
8	V-L23	26	I-UC4
9	V-L31	27	I-EF
10	If-1	28	Icp-EF
11	If-2		
12	If-3		
13	If-Ub		
14	THD-I1		
15	THD-I2		
16	THD-I3		
17	THD-I4		

Table 3: Alarm / Trip flags.

Alarm / Trip Flags

Block by DI	Store in FR	Display Reset	Trip / Alarm	x	CHAR	Trigger	Display	Relay 8	Relay 7	Relay 6	Relay 5	Relay 4	Relay 3	Relay 2	Relay 1

bit16 bit1

Relay 1 – Relay 8: Output relay 1 – 8: “0” represents relay is inactive if trip or alarm occurs, relay is active if “1”.

Display: Output display: “0” represents display message inactive if trip or alarm occurs, display message is active if “1”.

Trigger: “0” represents value > limit, “1” means value < limit.

CHAR: “0” represents the set T-ON and T-OFF delays were used

“1” represents the set characteristic curve was used.

Trip / Alarm: “0” means TRIP, “1” means ALARM.

Display Reset: “0” represents automatic reset of the display message.

“1” means manual reset of the display message.

Store in FR: “1” represents fault recording is "active" for this setting,

“0” means this function is disabled.

Block by DI: “1” represents that the alarm can be blocked by the digital input,

“0” means this function is disabled.

X: Reserved.

There are other values present in the devices memory than the ones mentioned above. They can contain important setup data for the device. Do never write to any address if you’re not sure what data it contains! Do never write to any address if writing is not official allowed!

3 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 KSR-Z and the PC!

   Possible causes can be:

   - Check adjustment of baud rate, parity and address at the KSR-Z, 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 there is a multiple reservation of the interface at the PC, 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 connection may not be grounded. If this is the case, separate the shielding from the grounding.
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