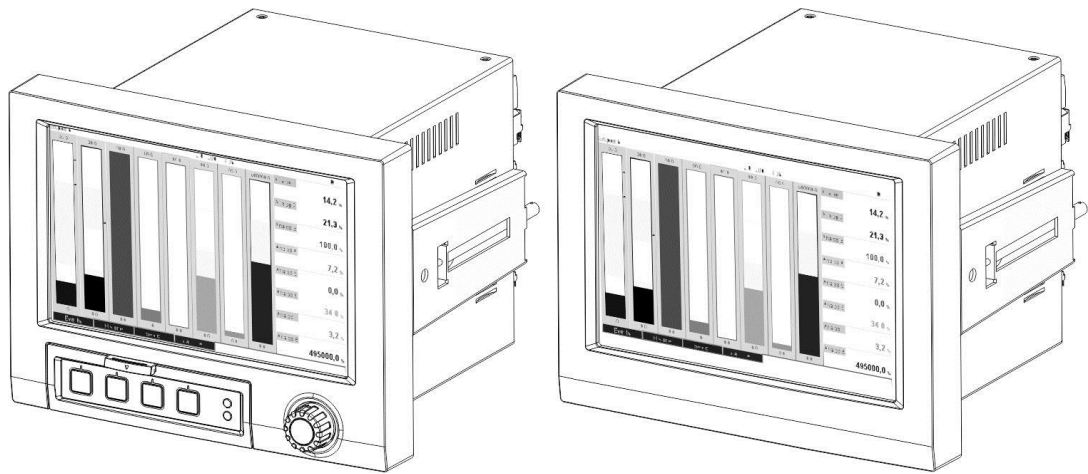


# Additional instructions

## Videographic recorder LINAX DR3000

Modbus RTU / TCP Slave



## Table of contents:

1 General information .....	3
1.1 Scope of delivery .....	3
1.2 Prerequisites .....	3
1.3 Modbus RTU connection .....	4
1.4 Modbus TCP connection .....	4
1.4.1 Transfer LED .....	4
1.4.2 Link LED .....	4
1.5 Functional description .....	4
1.6 Checking the availability of the Modbus Slave function .....	5
2 Settings in the setup .....	5
2.1 Modbus TCP, RS485 .....	5
2.2 Universal channels .....	6
2.2.1 Data transfer: Modbus Master -> device: .....	6
2.2.2 Data transfer: Device → Modbus Master: .....	6
2.3 Math channels .....	7
2.3.1 Data transfer: Device → Modbus Master: .....	7
2.4 Digital channels .....	7
2.4.1 Data transfer: Modbus Master → Device: .....	7
2.4.2 Data transfer: Device → Modbus Master: .....	7
2.5 General information .....	8
2.6 Addressing .....	9
2.6.1 Modbus Master → device: instantaneous value of universal channels .....	9
2.6.2 Modbus Master → device: digital input state .....	11
2.6.2.1 Writing all the states simultaneously .....	11
2.6.2.2 Writing states individually .....	12
2.6.3 Device → Modbus Master: universal channels (instantaneous value) .....	13
2.6.4 Device → Modbus Master: math channels (result) .....	15
2.6.5 Device → Modbus Master: digital channels (state) .....	18
2.6.5.1 Reading out all the states simultaneously .....	18
2.6.5.2 Reading out states individually .....	19
2.6.6 Device → Modbus Master: digital channels (totalizer) .....	20
2.6.7 Device → Modbus Master: integrated universal channels (totalizer) .....	22
2.6.8 Device → Modbus Master: integrated math channels (totalizer) .....	25
2.6.9 Device → Modbus Master: reading relay states .....	27
2.6.10 Modbus Master → device: set relay (telealarm option) .....	28
2.6.11 Modbus Master → device: change limit values .....	28
2.6.11.1 Initializing limit value changes .....	29
2.6.11.2 Changing the limit values .....	29
2.6.11.3 Specifying the reason for changing the limit value .....	31
2.6.11.4 Accepting limit values .....	32
2.6.11.5 Reading out the communication status .....	32
2.6.11.6 Reading out limit values .....	33
2.6.11.7 Tables and definitions .....	34
2.6.12 Modbus Master → device: transmit text .....	34
2.6.13 Modbus Master → device: batch data (batch option) .....	35
2.6.13.1 Start batch .....	35
2.6.13.2 Ending the batch .....	36
2.6.13.3 Configuring the batch designation .....	37
2.6.13.4 Configuring the batch name .....	38
2.6.13.5 Configuring the batch number .....	39
2.6.13.6 Setting the preset counter .....	40
2.6.13.7 Reading out the batch status .....	41
2.6.14 Structure of the process values .....	42
2.6.14.1 32-bit floating point number (IEEE-754) .....	42
2.6.14.2 64-bit floating point number (IEEE-754) .....	42
2.6.14.3 Limit value violations .....	43

2.6.14.4 Status of the floating point number .....	44
3 Overview of registers.....	45
4 Troubleshooting.....	51
4.1 Troubleshooting for Modbus TCP.....	51
4.2 Troubleshooting for Modbus RTU .....	51
5 List of abbreviations/glossary of terms.....	51
6 Index.....	51

## 1 General information

Notice signs:

### NOTICE

#### Notice

Failure to observe the notice can result in a device defect or a malfunction!



#### Tip

Indicates additional information.

### 1.1 Scope of delivery



**This manual provides an additional description for a special software option.**

This additional manual does **not** replace the relevant Operating Instructions!

### 1.2 Prerequisites

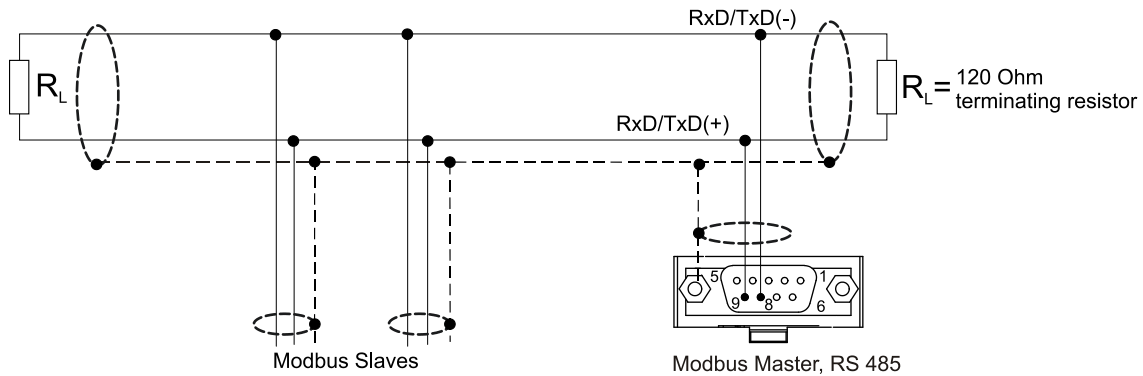
The "Modbus Slave" option must be enabled in the device. To retrofit optional functions, please follow the information in the Operating Instructions.

It is possible to combine the Modbus Slave RTU and the telealarm software option. However the device's RS485/232 interface is occupied by the Modbus Slave cable. This means that the Internet/e-mail functionalities of the telealarm software can be used, but it is not possible to use the modem connection via RS232.

Modbus RTU is possible via the combined RS223/RS485 interface (back of device), but only the RS485 interface is supported. Modbus TCP is possible via the integrated Ethernet interface (back of device).

### 1.3 Modbus RTU connection

**NOTICE** The terminal assignment does not correspond to the standard (Modbus over serial line specification and implementation guide V1.02).



Pin	Direction	Signal	Description
Housing	-	Functional ground	Protective ground
1	-	GND	Ground (isolated)
9	Input	RxD/TxD(+)	RS-485 B wire
8	Output	RxD/TxD(-)	RS-485 A wire

Tab. 1: Pin assignment of Modbus RTU connector

### 1.4 Modbus TCP connection

The Modbus TCP interface is physically identical to the Ethernet interface.

#### 1.4.1 Transfer LED

Status LED	Indicator for
Off	No communication
Flashes green	Communication

Tab. 2: Description of the function of the status LEDs for Modbus TCP

#### 1.4.2 Link LED

Status LED	Indicator for
Off	No connection
Flashes yellow	Activity

Tab. 3: Description of the function of the link LEDs for Modbus TCP

### 1.5 Functional description

The Modbus RTU option makes it possible to connect the device to Modbus via RS485 with the functionality of a Modbus RTU slave.

Supported baud rates: 9600, 19200, 38400, 57600, 115200

Parity: None, Even, Odd

The Modbus TCP option makes it possible to connect the device to Modbus TCP with the functionality of a Modbus TCP slave. The Ethernet connection supports 10/100 Mbit, full or half duplex.

The user can choose between Modbus TCP or Modbus RTU in the settings. It is not possible to select both at the same time.

## 1.6 Checking the availability of the Modbus Slave function

In the Main menu under "**Diagnostics / Device information / Device options**" or "**Setup / Advanced setup / System / Device options**" it is possible to check whether the Modbus Slave option is enabled under **Fieldbus**. Under **Communication**, it is possible to determine the hardware interface via which communication is possible:

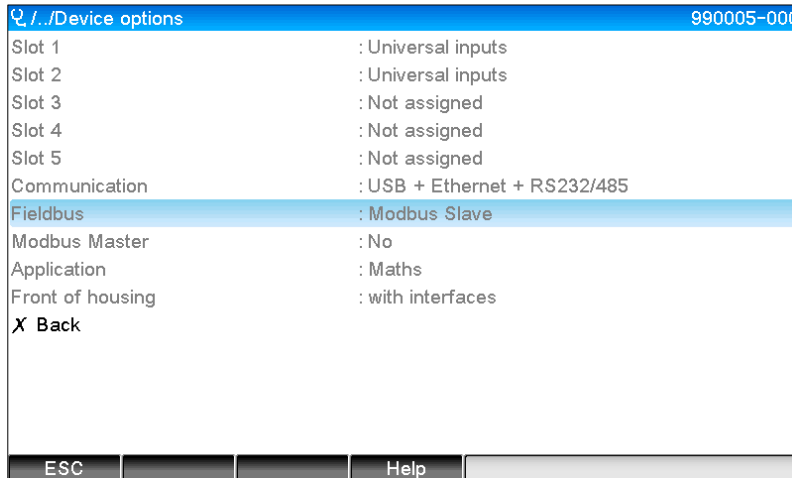


Fig. 1: Checking the availability of the Modbus Slave function

## 2 Settings in the setup

### 2.1 Modbus TCP, RS485

The interface to be used for Modbus can be selected under "**Setup / Advanced setup / Communication / Modbus Slave**":

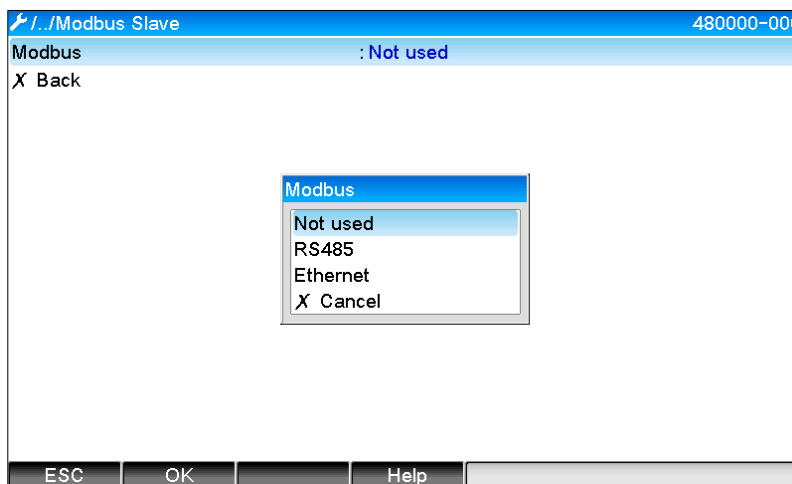


Fig. 2: Selecting the interface for Modbus

If Modbus RTU (RS485) is selected, the following parameters can be configured:

- Device address (1 to 247)
- Baud rate (9600, 19200, 38400, 57600, 115200)
- Parity (None, Even, Odd)

If Modbus TCP (Ethernet) is selected, the following parameter can be configured:

- Port TCP port (standard: 502)

If Modbus TCP is used, the settings for the Ethernet interface can be made under **"Setup / Advanced setup / Communication / Ethernet"**:

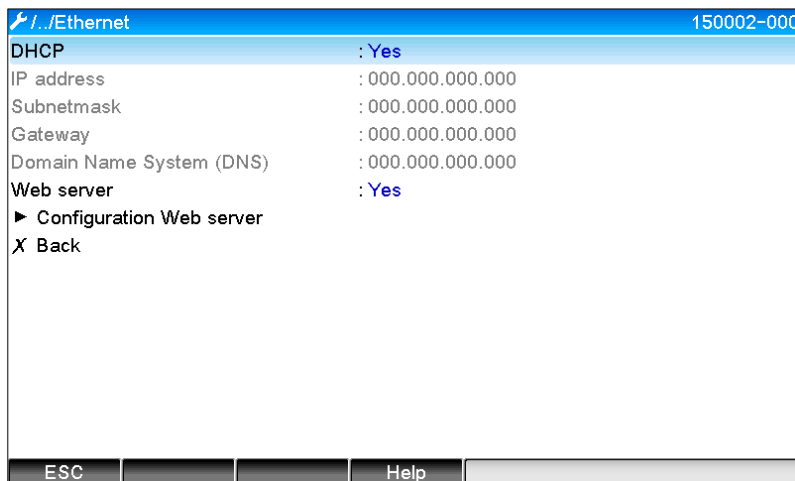


Fig. 3: Settings for the Ethernet interface

In addition, it is possible to set a timeout period under **"/Expert / Communication / Modbus Slave / Timeout"** after which the channel in question is set to "Invalid".

The timeout only refers to channels that receive a value from the Modbus Master. It does not affect channels that are only read by the Modbus Master.

## 2.2 Universal channels



All the universal inputs (40) are enabled and can be used as Modbus inputs, even if they are not really available as plug-in cards.

### 2.2.1 Data transfer: Modbus Master -> device:

Under **"Setup / Advanced setup / Inputs / Universal inputs / Universal input X"**, the **Signal** parameter is set to **Modbus Slave**:

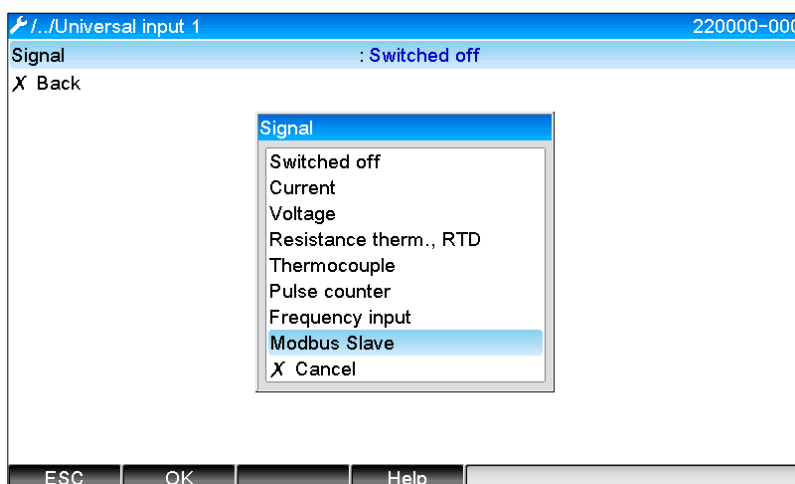


Fig. 4: Setting the universal input to Modbus

With this setting, a Modbus Master can write to the universal input as described in Section 2.6.1.

### 2.2.2 Data transfer: Device → Modbus Master:


The Modbus Master can read universal inputs 1 to 40 as described in Section 2.6.3 .

## 2.3 Math channels

### 2.3.1 Data transfer: Device → Modbus Master:

Math channels are available as an option under **Setup / Advanced setup / Application / Maths**. The results can be read by the Modbus Master (see Sections 2.6.5 and 2.6.4).

## 2.4 Digital channels

 All the digital inputs (20) are enabled and can be used as Modbus inputs, even if they are not really available as plug-in cards.

### 2.4.1 Data transfer: Modbus Master → Device:

Under **Setup / Advanced setup / Inputs / Digital inputs / Digital input X** the **Function** parameter is set to **Modbus Slave**:

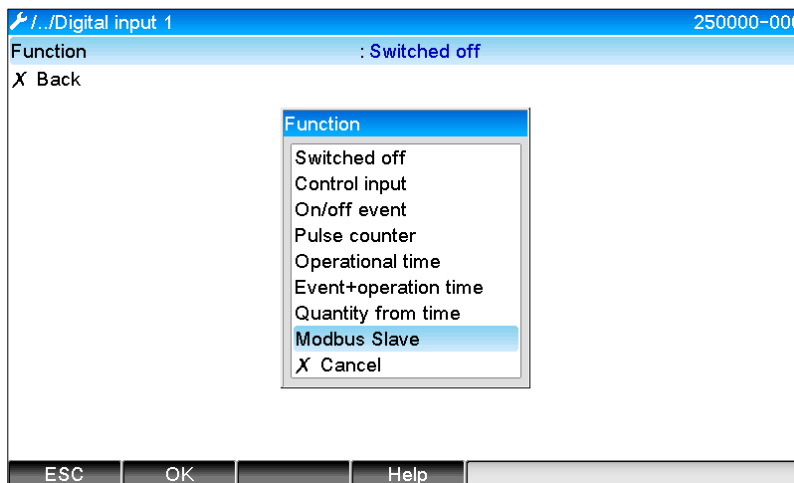


Fig. 5: Setting the digital channel to Modbus

With this setting, the Modbus Master can write to the digital channel as described in Section 2.6.2. The digital state transmitted by the Modbus Master has the same function in the device as the state of a digital channel that is actually present.

### 2.4.2 Data transfer: Device → Modbus Master:

#### Control input or on/off event

The Modbus Master can read out the digital state of the digital channel so configured (see Section 2.6.5).

#### Pulse counter or operational time

The Modbus Master can read out the totalizer or the total operational time of the digital channel so configured (see Section 2.6.6).

#### Event + operation time

The Modbus Master can read out the digital state and the totalizer of the digital channel so configured (see Section 2.6.5 and 2.6.6).

## 2.5 General information

Functions **03: Read Holding Register**, **16: Write Multiple Registers** and **06 Write Single Register** are supported

The following values can be transmitted from the **Modbus Master to the device**:

- Analog values (instantaneous values)
- Digital states

The following values can be transmitted from the **device to the Modbus Master**:

- Analog values (instantaneous values)
- Integrated analog values (totalizer)
- Math channels (result: state, instantaneous value, operating time, totalizer)
- Integrated math channels (totalizer)
- Digital states
- Pulse counter (totalizer)
- Operational times
- Relay states

Furthermore, additional functions can be available depending on the function.

Telealarm application:

- Control relay

Batch application:

- Start/stop batch, configure parameters, etc.

General:

- Send texts that are entered in the event list



## 2.6 Addressing

The query/response examples refer to Modbus RTU via RS485.  
The register addresses are all to the base 0.



A maximum of 123 registers can be read/written per query.

### 2.6.1 Modbus Master → device: instantaneous value of universal channels

The values of universal channels 1-40 must be written via **16 Write Multiple Registers**.  
It is possible to transmit the value as a 32 bit float or 64 bit float.

Channel	Reg. dec.	Reg. hex.	Length, byte	Reg. dec.	Reg. hex.	Length, byte
Universal 1	200	0C8	6	5200	1450	10
Universal 2	203	0CB	6	5205	1455	10
Universal 3	206	0CE	6	5210	145A	10
Universal 4	209	0D1	6	5215	145F	10
Universal 5	212	0D4	6	5220	1464	10
Universal 6	215	0D7	6	5225	1469	10
Universal 7	218	0DA	6	5230	146E	10
Universal 8	221	0DD	6	5235	1473	10
Universal 9	224	0E0	6	5240	1478	10
Universal 10	227	0E3	6	5245	147D	10
Universal 11	230	0E6	6	5250	1482	10
Universal 12	233	0E9	6	5255	1487	10
Universal 13	236	0EC	6	5260	148C	10
Universal 14	239	0EF	6	5265	1491	10
Universal 15	242	0F2	6	5270	1496	10
Universal 16	245	0F5	6	5275	149B	10
Universal 17	248	0F8	6	5280	14A0	10
Universal 18	251	0FB	6	5285	14A5	10
Universal 19	254	0FE	6	5290	14AA	10
Universal 20	257	101	6	5295	14AF	10
Universal 21	260	104	6	5300	14B4	10
Universal 22	263	107	6	5305	14B9	10
Universal 23	266	10A	6	5310	14BE	10
Universal 24	269	10D	6	5315	14C3	10
Universal 25	272	110	6	5320	14C8	10
Universal 26	275	113	6	5325	14CD	10
Universal 27	278	116	6	5330	14D2	10
Universal 28	281	119	6	5335	14D7	10
Universal 29	284	11C	6	5340	14DC	10
Universal 30	287	11F	6	5345	14E1	10
Universal 31	290	122	6	5350	14E6	10
Universal 32	293	125	6	5355	14EB	10
Universal 33	296	128	6	5360	14F0	10
Universal 34	299	12B	6	5365	14F5	10
Universal 35	302	12E	6	5370	14FA	10
Universal 36	305	131	6	5375	14FF	10
Universal 37	308	134	6	5380	1504	10
Universal 38	311	137	6	5385	1509	10
Universal 39	314	13A	6	5390	150E	10
Universal 40	317	13D	6	5395	1513	10

Tab. 4: Register addresses of the universal inputs

The 1st register contains the status (see Section 2.6.14.4) of the floating point number (32 bit float) transmitted in the 2nd and 3rd register.

**Example:** Writing to universal channel 6 with the value 123.456 (32 bit float), slave address 1

Byte	0	1	2	3	4	5
	<b>00</b>	<b>80</b>	<b>42</b>	<b>F6</b>	<b>E9</b>	<b>79</b>
		Floating point number status	Floating point number = 123.456 (32 bit float)			

Register	Value (hex)
215	<b>0080</b>
216	<b>42F6</b>
217	<b>E979</b>

**Query:**

Slave address	01	
Function	10	16: Write Multiple Registers
Register	00 D7	Register 215
No. Registers	00 03	3 Registers
No. Bytes	06	
Status	00 80	
FLP	42 F6 E9 79	123.456
CRC	28 15	

**Response:**

Slave address	01	
Function	10	16: Write Multiple Registers
Register	00 D7	Register 271
No. Registers	00 03	
CRC	30 30	

The 1st register contains the status (see Section 2.6.14.4) of the floating point number (64 bit float) transmitted in the 2nd to 5th register).

**Example:** Writing to universal channel 6 with the value 123.456 (64 bit float), slave address 1

Byte	0	1	2	3	4	5	6	7	8	9
	<b>00</b>	<b>80</b>	<b>40</b>	<b>5E</b>	<b>DD</b>	<b>2F</b>	<b>1A</b>	<b>9F</b>	<b>BE</b>	<b>77</b>
		Floating point number status	Floating point number = 123.456 (64 bit float)							

Register	Value (hex)
5225	<b>0080</b>
5226	<b>405E</b>
5227	<b>DD2F</b>
5228	<b>1A9F</b>
5229	<b>BE77</b>

**Query:**

Slave address	01	
Function	10	16: Write Multiple Registers
Register	14 69	Register 5225
No. Registers	00 05	5 Registers
No. Bytes	0A	
Status	00 80	
FLP	40 5E DD 2F 1A 9F BE 77	123.456
CRC	67 56	

**Response:**

Slave address	01	
Function	10	16: Write Multiple Registers
Register	14 69	Register 5225

No. Registers 00 05  
CRC D5 E6

## 2.6.2 Modbus Master → device: digital input state

### 2.6.2.1 Writing all the states simultaneously

The states of digital inputs 1-20 must be written via **16 Write Multiple Registers**.  
Digital 1-16 corresponds to Register 1240 Bit 0-15,  
Digital 17-20 corresponds to Register 1241 Bit 0-3.

Channel	Reg. dec.	Reg. hex.	Length, byte
Digital 1-16	1240	4D8	2
Digital 17-20	1241	4D9	2

Tab. 5: Register addresses of digital inputs (Modbus Master → device)

**Example: Setting digital input 4 to high (all others to low), slave address 1**

Byte 0 state (bit 15-8)	Byte 1 state (bit 7-0)
00000000	00001000
0	Bit 3 high digital 4
Byte 2 state (bit 15-8)	Byte 3 state (bit 7-0)
00000000	00000000
0	0
<b>Register</b>	<b>Value (hex)</b>
1240	0008
1241	0000

**Query:** Slave address 01  
Function 10 16: Write Multiple Registers  
Register 04 D8 Register 1240  
No. Registers 00 02 2 Registers  
No. Bytes 04  
Digital status 00 08 00 00 Digital 4 to high  
CRC 4C 57

**Response:** Slave address 01  
Function 10 16: Write Multiple Registers  
Register 04 D8 Register 1240  
No. Registers 00 02  
CRC C0 C3

### 2.6.2.2 Writing states individually

The states of digital inputs 1-20 can be written via **16 Write Multiple Registers** or **06 Write Single Register**.

Channel	Reg. dec.	Reg. hex.	Length, byte
Digital 1	1200	4B0	2
Digital 2	1201	4B1	2
Digital 3	1202	4B2	2
Digital 4	1203	4B3	2
Digital 5	1204	4B4	2
Digital 6	1205	4B5	2
Digital 7	1206	4B6	2
Digital 8	1207	4B7	2
Digital 19	1208	4B8	2
Digital 10	1209	4B9	2
Digital 11	1210	4BA	2
Digital 12	1211	4BB	2
Digital 13	1212	4BC	2
Digital 14	1213	4BD	2
Digital 15	1214	4BE	2
Digital 16	1215	4BF	2
Digital 17	1216	4C0	2
Digital 18	1217	4C1	2
Digital 19	1218	4C2	2
Digital 20	1219	4C3	2

Tab. 6: Register addresses of digital inputs (Modbus Master → device)

#### Example: Setting digital input 4 to high, slave address 1

Byte 0	Byte1
00000000	00000001
Always 0	1: Set

Register	Value (hex)
1203	0001

**Query:**

Slave address	01	
Function	10	16: Write Multiple Registers
Register	04 B3	Register 1203
No. Registers	00 01	1 Register
No. Bytes	02	
Digital status	00 01	Digital 4 to high
CRC	38 53	

**Response:**

Slave address	01	
Function	10	16: Write Multiple Registers
Register	04 B3	Register 1203
No. Registers	00 01	
CRC	F1 1E	

### 2.6.3 Device → Modbus Master: universal channels (instantaneous value)

Universal inputs 1-40 are read out via **03 Read Holding Register (4x)**.

It is possible to transmit the value as a 32 bit float or 64 bit float.

Channel	Reg. dec.	Reg. hex.	Length, byte	Reg. dec.	Reg. hex.	Length, byte
Universal 1	200	0C8	6	5200	1450	10
Universal 2	203	0CB	6	5205	1455	10
Universal 3	206	0CE	6	5210	145A	10
Universal 4	209	0D1	6	5215	145F	10
Universal 5	212	0D4	6	5220	1464	10
Universal 6	215	0D7	6	5225	1469	10
Universal 7	218	0DA	6	5230	146E	10
Universal 8	221	0DD	6	5235	1473	10
Universal 9	224	0E0	6	5240	1478	10
Universal 10	227	0E3	6	5245	147D	10
Universal 11	230	0E6	6	5250	1482	10
Universal 12	233	0E9	6	5255	1487	10
Universal 13	236	0EC	6	5260	148C	10
Universal 14	239	0EF	6	5265	1491	10
Universal 15	242	0F2	6	5270	1496	10
Universal 16	245	0F5	6	5275	149B	10
Universal 17	248	0F8	6	5280	14A0	10
Universal 18	251	0FB	6	5285	14A5	10
Universal 19	254	0FE	6	5290	14AA	10
Universal 20	257	101	6	5295	14AF	10
Universal 21	260	104	6	5300	14B4	10
Universal 22	263	107	6	5305	14B9	10
Universal 23	266	10A	6	5310	14BE	10
Universal 24	269	10D	6	5315	14C3	10
Universal 25	272	110	6	5320	14C8	10
Universal 26	275	113	6	5325	14CD	10
Universal 27	278	116	6	5330	14D2	10
Universal 28	281	119	6	5335	14D7	10
Universal 29	284	11C	6	5340	14DC	10
Universal 30	287	11F	6	5345	14E1	10
Universal 31	290	122	6	5350	14E6	10
Universal 32	293	125	6	5355	14EB	10
Universal 33	296	128	6	5360	14F0	10
Universal 34	299	12B	6	5365	14F5	10
Universal 35	302	12E	6	5370	14FA	10
Universal 36	305	131	6	5375	14FF	10
Universal 37	308	134	6	5380	1504	10
Universal 38	311	137	6	5385	1509	10
Universal 39	314	13A	6	5390	150E	10
Universal 40	317	13D	6	5395	1513	10

Tab. 7: Register addresses of universal inputs (device → Modbus Master)

Alternatively at the following addresses:

- 4000-4078 (32 bit float) without a status
- 8000-8156 (64 bit float ) without a status
- 6800-6839 (status)

The 1st register contains the status (see Section 2.6.14.4) and the limit value violations (see Section 0) of the floating point number (32 bit float) transmitted in the 2nd and 3rd register.

**Example: Reading analog 1 with the value 82.47239685 (32 bit float), slave address 1**

Byte	0	1	2	3	4	5
	<b>00</b>	<b>80</b>	<b>42</b>	<b>A4</b>	<b>F1</b>	<b>DE</b>
	Limit value violations	Floating point number status	Floating point number = 82.47239685			

Register	Value (hex)
200	0080
201	42A4
202	F1DE

**Query:** Slave address 01  
 Function 03 03: Read Holding Register  
 Register 00 C8 Register 200  
 No. Registers 00 03 3 Registers  
 CRC 84 35

**Response:** Slave address 01  
 Function 03 03: Read Holding Register  
 No. Bytes 06 6 Bytes  
 Status 00 80  
 FLP 42 A4 F1 DE 82.47239685  
 CRC B0 F8

The 1st register contains the status (see Section 2.6.14.4) and the limit value violations (see Section 0) of the floating point number (64 bit float) transmitted in the 2nd to 5th register.

**Example: Reading universal channel 1 with the value 82.4723968506 (64 bit float), slave address 1**

Byte	0	1	2	3	4	5	6	7	8	9
	<b>00</b>	<b>80</b>	<b>40</b>	<b>54</b>	<b>9E</b>	<b>3B</b>	<b>C0</b>	<b>00</b>	<b>00</b>	<b>00</b>
	Limit value violations	Floating point number status	Floating point number = 82.4723968506 (64 bit float)							

Register	Value (hex)
5200	0080
5201	4054
5202	9E3B
5203	C000
5204	0000

**Query:** Slave address 01  
 Function 03 03: Read Holding Register  
 Register 14 50 Register 5200  
 No. Registers 00 05 5 Registers  
 CRC 80 28

**Response:** Slave address 01  
 Function 03 03: Read Holding Register  
 No. Bytes 0A 10 Bytes  
 Status 00 80  
 FLP 40 54 9E 3B C0 00 00 00 82.4723968506  
 CRC 91 3E

### 2.6.4 Device → Modbus Master: math channels (result)

The results of math channels 1-12 are read out via **03 Read Holding Register (4x)**.  
It is possible to transmit the value as a 32 bit float or 64 bit float.

Channel	Reg. dec.	Reg. hex.	Length, byte	Reg. dec.	Reg. hex.	Length, byte
Math 1	1500	5DC	6	6500	1964	10
Math 2	1503	5DF	6	6505	1969	10
Math 3	1506	5E2	6	6510	196E	10
Math 4	1509	5E5	6	6515	1973	10
Math 5	1512	5DC	6	6520	1978	10
Math 6	1515	5DF	6	6525	197D	10
Math 7	1518	5DC	6	6530	1982	10
Math 8	1521	5DF	6	6535	1987	10
Math 9	1524	5DC	6	6540	198C	10
Math 10	1527	5DF	6	6545	1991	10
Math 11	1530	5DC	6	6550	1996	10
Math 12	1533	5DF	6	6555	199B	10

Tab. 8: Register addresses of math channels (device → Modbus Master)

Alternatively at the following addresses:

- 4200-4222 (32 bit float) without a status
- 8400-8444 (64 bit float) without a status
- 6900-6939 (status)

The 1st register contains the status (see Section 2.6.14.4) and the limit value violations (see Section 0) of the floating point number (32 bit float) transmitted in the 2nd and 3rd register.

**Example: Reading math 1 (instantaneous value result), (32 bit float), slave address 1**

Byte	0	1	2	3	4	5
	<b>00</b>	<b>80</b>	<b>46</b>	<b>40</b>	<b>E6</b>	<b>B7</b>
	Limit value violations	Floating point number status	Floating point number = 12345.67871			

Register	Value (hex)
1500	<b>0080</b>
1501	<b>4640</b>
1502	<b>E6B7</b>

**Query:**

Slave address	01	
Function	03	03: Read Holding Register
Register	05 DC	Register 1500
No. Registers	00 03	3 Registers
CRC	C4 FD	

**Response:**

Slave address	01	
Function	03	03: Read Holding Register
No. Bytes	06	6 Bytes
Status	00 80	
FLP	46 40 E6 B7	12345.67871
CRC	3E 21	

The 1st register contains the status (see Section 2.6.14.4) and the limit value violations (see Section 0) of the floating point number (64 bit float) transmitted in the 2nd to 5th register.

**Example: Reading math 1 (instantaneous value result), (64 bit float), slave address 1**

Byte	0	1	2	3	4	5	6	7	8	9
	<b>00</b>	<b>80</b>	<b>40</b>	<b>C8</b>	<b>1C</b>	<b>D6</b>	<b>E6</b>	<b>31</b>	<b>F8</b>	<b>A1</b>
	Limit value violations	Floating point number status	Floating point number = 12345.6789 (64 bit float)							

Register	Value (hex)
6500	0080
6501	40C8
6502	1CD6
6503	E631
6504	F8A1

**Query:** Slave address 01  
 Function 03 03: Read Holding Register  
 Register 19 64 Register 6500  
 No. Registers 00 05 5 Registers  
 CRC C3 4A

**Response:** Slave address 01  
 Function 03 03: Read Holding Register  
 No. Bytes 0A 10 Bytes  
 Status 00 80  
 FLP 40 C8 1C D6 E6 31 F8 A1 12345.6789  
 CRC A7 FD



**Example: Reading math 1-12 (state result), slave address 1**

The states of math channels 1-12 are read out via **03 Read Holding Register (4x)**.  
Math 1-12 corresponds to Register 1800 Bit 0-11.

Channel	Reg. dec.	Reg. hex.	Length, byte
Math 1-12	1800	708	2

Tab. 9: Register address of states of math channels (device → Modbus Master)

Byte 0 state (bit 11-8)	Byte 1 state (bit 7-0)
00000000	00000011
	Bit 0 and 1 high Math 1 and 2

Register	Value (hex)
1800	0003

**Query:**

Slave address	01	
Function	03	3: Read Holding Register
Register	07 08	Register 1800
No. Registers	00 01	1 Register
CRC	04 BC	

**Response:**

Slave address	01	
Function	03	16: Write Multiple Registers
Number	02	2 Bytes
States	00 03	Math 1 and 2 state high
CRC	F8 45	

## 2.6.5 Device → Modbus Master: digital channels (state)

### 2.6.5.1 Reading out all the states simultaneously

The states of digital inputs 1-20 are read out via **03 Read Holding Register (4x)**.

Digital 1-16 corresponds to Register 1240 Bit 0-15, digital 17-20 corresponds to Register 1241 Bit 0-3.

Channel	Reg. dec.	Reg. hex.	Length, byte
Digital 1-16	1240	4D8	2
Digital 17-20	1241	4D9	2

Tab. 10: Register addresses of all digital inputs (device → Modbus Master)

**Example:**      **Reading the states of digital inputs 1-20, slave address 1**

Byte 0 state (bit 15-8)	Byte 1 state (bit 7-0)
00000000	00001000
	Bit 3 high digital 4
Byte 2 state (bit 15-8)	Byte 3 state (bit 7-0)
00000000	00000000
0	0

Register	Value (hex)
1240	0008
1241	0000

**Query:**

Slave address	01	
Function	03	3: Read Holding Register
Register	04 D8	Register 1240
No. Registers	00 02	2 Registers
CRC	45 00	

**Response:**

Slave address	01	
Function	03	16: Write Multiple Registers
Number	04	4 Bytes
States	00 08	Digital 4
CRC	7B F1	

### 2.6.5.2 Reading out states individually

The states of digital inputs 1-20 are read out via **03 Read Holding Register (4x)**.

Channel	Reg. dec.	Reg. hex.	Length, byte
Digital 1	1200	4B0	2
Digital 2	1201	4B1	2
Digital 3	1202	4B2	2
Digital 4	1203	4B3	2
Digital 5	1204	4B4	2
Digital 6	1205	4B5	2
Digital 7	1206	4B6	2
Digital 8	1207	4B7	2
Digital 19	1208	4B8	2
Digital 10	1209	4B9	2
Digital 11	1210	4BA	2
Digital 12	1211	4BB	2
Digital 13	1212	4BC	2
Digital 14	1213	4BD	2
Digital 15	1214	4BE	2
Digital 16	1215	4BF	2
Digital 17	1216	4C0	2
Digital 18	1217	4C1	2
Digital 19	1218	4C2	2
Digital 20	1219	4C3	2

Tab. 11: Register addresses of digital inputs (device → Modbus Master)

#### Example: Reading digital input 6, slave address 1

Byte 0	Byte1
00000000	00000001
Always 0	1: Set Digital 6

Register	Value (hex)
1205	0001

**Query:**

Slave address	01	
Function	03	3: Read Holding Register
Register	04 B5	Register 1205
No. Registers	00 01	1 Register
CRC	94DC	

**Response:**

Slave address	01	
Function	03	3: Read Holding Register
Number	02	2 Bytes
Digital status	00 01	Digital 6 to high
CRC	79 84	

### 2.6.6 Device → Modbus Master: digital channels (totalizer)

The totalizers of digital inputs 1-20 are read out via **03 Read Holding Register (4x)**. It is possible to transmit the value as a 32 bit float or 64 bit float.

Channel	Reg. dec.	Reg. hex.	Length, byte	Reg. dec.	Reg. hex.	Length, byte
Digital 1	1300	514	6	6300	189C	10
Digital 2	1303	517	6	6305	18A1	10
Digital 3	1306	51A	6	6310	18A6	10
Digital 4	1309	51D	6	6315	18AB	10
Digital 5	1312	520	6	6320	18B0	10
Digital 6	1315	523	6	6325	18B5	10
Digital 7	1318	526	6	6330	18BA	10
Digital 8	1321	529	6	6335	18BF	10
Digital 9	1324	52C	6	6340	18C4	10
Digital 10	1327	52F	6	6345	18C9	10
Digital 11	1330	532	6	6350	18CE	10
Digital 12	1333	535	6	6355	18D3	10
Digital 13	1336	538	6	6360	18D8	10
Digital 14	1339	53B	6	6365	18DD	10
Digital 15	1342	53E	6	6370	18E2	10
Digital 16	1345	541	6	6375	18E7	10
Digital 17	1348	544	6	6380	18EC	10
Digital 18	1351	547	6	6385	18F1	10
Digital 19	1354	54A	6	6390	18F6	10
Digital 20	1357	54D	6	6395	18FB	10

Tab. 12: Register addresses of digital input totalizers (device → Modbus Master)

The 1st register (low byte) contains the status (see Section 2.6.14.4) and the limit value violations (see Section 0) of the floating point number (32 bit float) transmitted in the 2nd and 3rd register.

**Example: Reading totalizer of digital input 6 (32 bit float), slave address 1**

Byte	0	1	2	3	4	5
	<b>00</b>	<b>80</b>	<b>40</b>	<b>C9</b>	<b>99</b>	<b>9A</b>
	Limit value violations	Floating point number status	Floating point number = 65552.0			

Register	Value (hex)
1315	0080
1316	40C9
1317	999A

**Query:** Slave address 01  
 Function 03 3: Read Holding Register  
 Register 05 23 Register 1315  
 No. Registers 00 03 3 Registers  
 CRC F4 CD

**Response:** Slave address 01  
 Function 03 3: Read Holding Register  
 Number 06 6 Bytes  
 Digital status 00 80 40 C9 99 9A 6.3  
 CRC 0F 6E

The 1st register (low byte) contains the status (see Section 2.6.14.4) and the limit value violations (see Section 0) of the floating point number (64 bit float) transmitted in the 2nd to 5th register.

**Example: Reading totalizer of digital input 6 (64 bit float), slave address 1**

Byte	0	1	2	3	4	5	6	7	8	9
	00	80	40	19	33	33	39	80	00	00
	Limit value violations	Floating point number status	Floating point number = 6.3 (64 bit float)							

Register	Value (hex)
6325	0080
6326	4019
6327	3333
6328	3980
6329	0000

**Query:**

Slave address	01	
Function	03	03: Read Holding Register
Register	18 B5	Register 6325
No. Registers	00 05	5 Registers
CRC	92 8F	

**Response:**

Slave address	01	
Function	03	03: Read Holding Register
No. Bytes	0A	10 Bytes
Status	00 80	
FLP	40 19 33 33 39 80 00 00	6.3
CRC	C5 32	

### 2.6.7 Device → Modbus Master: integrated universal channels (totalizer)

The totalizers of universal inputs 1-40 are read out via **03 Read Holding Register (4x)**.

It is possible to transmit the value as a 32 bit float or 64 bit float.

Channel	Reg. dec.	Reg. hex.	Length, byte	Reg. dec.	Reg. hex.	Length, byte
Universal1	800	320	6	5800	16A8	10
Universal2	803	323	6	5805	16AD	10
Universal3	806	326	6	5810	16B2	10
Universal4	809	329	6	5815	16B7	10
Universal5	812	32C	6	5820	16BC	10
Universal6	815	32F	6	5825	16C1	10
Universal7	818	332	6	5830	16C6	10
Universal8	821	335	6	5835	16CB	10
Universal9	824	338	6	5840	16D0	10
Universal10	827	33B	6	5845	16D5	10
Universal11	830	33E	6	5850	16DA	10
Universal12	833	341	6	5855	16DF	10
Universal13	836	344	6	5860	16E4	10
Universal14	839	347	6	5865	16E9	10
Universal15	842	34A	6	5870	16EE	10
Universal16	845	34D	6	5875	16F3	10
Universal17	848	350	6	5880	16F8	10
Universal18	851	353	6	5885	16FD	10
Universal19	854	356	6	5890	1702	10
Universal20	857	359	6	5895	1707	10
Universal21	860	35C	6	5900	170C	10
Universal22	863	35F	6	5905	1711	10
Universal23	866	362	6	5910	1716	10
Universal24	869	365	6	5915	171B	10
Universal25	872	368	6	5920	1720	10
Universal26	875	36B	6	5925	1725	10
Universal27	878	36E	6	5930	172A	10
Universal28	881	371	6	5935	172F	10
Universal29	884	374	6	5940	1734	10
Universal30	887	377	6	5945	1739	10
Universal31	890	37A	6	5950	173E	10
Universal32	893	37D	6	5955	1743	10
Universal33	896	380	6	5960	1748	10
Universal34	899	383	6	5965	174D	10
Universal35	902	386	6	5970	1752	10
Universal36	905	389	6	5975	1757	10
Universal37	908	38C	6	5980	175C	10
Universal38	911	38F	6	5985	1761	10
Universal39	914	392	6	5990	1766	10
Universal40	917	395	6	5995	176B	10

Tab. 13: Register addresses of universal input totalizers (device → Modbus Master)

The 1st register contains the status (see Section 2.6.14.4) and the limit value violations (see Section 0) of the floating point number (32 bit float) transmitted in the 2nd and 3rd register.

**Example:**      **Reading totalizer for universal channel 1 with the value 26557.48633 (32 bit float), slave address 1**

Byte	0	1	2	3	4	5
	<b>00</b>	<b>80</b>	<b>46</b>	<b>CF</b>	<b>7A</b>	<b>E6</b>
	Limit value violations	Floating point number status	Floating point number = 26557.48633			

Register	Value (hex)
800	<b>0080</b>
801	<b>46CF</b>
802	<b>7AE6</b>

**Query:**

Slave address	01	
Function	03	03: Read Holding Register
Register	03 20	Register 800
No. Registers	00 03	3 Registers
CRC	04 45	

**Response:**

Slave address	01	
Function	03	03: Read Holding Register
No. Bytes	06	6 Bytes
Status	00 80	
FLP	46 CF 7A E6	3192.73242
CRC	E6 FE	

The 1st register contains the status (see Section 2.6.14.4) and the limit value violations (see Section 0) of the floating point number (64 bit float) transmitted in the 2nd to 5th register.

**Example:**      **Reading totalizer for universal channel 1 with the value 33174.3672951 (64 bit float), slave address 1**

Byte	0	1	2	3	4	5	6	7	8	9
	<b>00</b>	<b>80</b>	<b>40</b>	<b>E0</b>	<b>32</b>	<b>CB</b>	<b>C0</b>	<b>E1</b>	<b>99</b>	<b>A9</b>
	Limit value violations	Floating point number status	Floating point number = 33174.3672951 (64 bit float)							

Register	Value (hex)
5800	0080
5801	40E0
5802	32CB
5803	C0E1
5804	99A9

**Query:**

Slave address	01	
Function	03	03: Read Holding Register
Register	16 A8	Register 5800
No. Registers	00 05	5 Registers
CRC	00 61	

**Response:**

Slave address	01	
Function	03	03: Read Holding Register
No. Bytes	0A	10 Bytes
Status	00 80	
FLP	40 E0 32 CB C0 E1 99 A9	33174.3672951
CRC	C7 54	



### 2.6.8 Device → Modbus Master: integrated math channels (totalizer)

The totalizers of the math channels are read out via **03 Read Holding Register (4x)**.  
It is possible to transmit the value as a 32 bit float or 64 bit float.

Channel	Reg. dec.	Reg. hex.	Length, byte	Reg. dec.	Reg. hex.	Length, byte
Math 1	1700	6A4	6	6700	1A2C	10
Math 2	1703	6A7	6	6705	1A31	10
Math 3	1706	6AA	6	6710	1A36	10
Math 4	1709	6AD	6	6715	1A3B	10
Math 5	1700	6A4	6	6720	1A40	10
Math 6	1703	6A7	6	6725	1A45	10
Math 7	1706	6AA	6	6730	1A4A	10
Math 8	1709	6AD	6	6735	1A4F	10
Math 9	1712	6B0	6	6740	1A54	10
Math 10	1715	6B3	6	6745	1A59	10
Math 11	1718	6B6	6	6750	1A5E	10
Math 12	1721	6B9	6	6755	1A63	10

Tab. 14: Register addresses of math channel totalizer (device → Modbus Master)

The 1st register contains the status (see Section 2.6.14.4) of the floating point number (32 bit float) transmitted in the 2nd and 3rd register.

#### Example: Reading totalizer of math 1 (32 bit float), slave address 1

Byte	0	1	2	3	4	5
	<b>00</b>	<b>80</b>	<b>4B</b>	<b>29</b>	<b>85</b>	<b>F4</b>
		Floating point number status	Floating point number = 33174.3672951			

Register	Value (hex)
1700	<b>0080</b>
1701	<b>4B29</b>
1702	<b>85F4</b>

**Query:**

Slave address	01	
Function	03	03: Read Holding Register
Register	06 A4	Register 1700
No. Registers	00 03	3 Registers
CRC	44 A0	

**Response:**

Slave address	01	
Function	03	03: Read Holding Register
No. Bytes	06	6 Bytes
Status	00 80	
FLP	4B 29 85 F4	33174.3672951
CRC	85 90	

The 1st register contains the status (see Section 2.6.14.4) of the floating point number (64 bit float) transmitted in the 2nd to 5th register).

**Example: Reading totalizer of math 1 (64 bit float), slave address 1**

Byte	0	1	2	3	4	5	6	7	8	9
	<b>00</b>	<b>80</b>	<b>41</b>	<b>68</b>	<b>5F</b>	<b>26</b>	<b>35</b>	<b>2A</b>	<b>FC</b>	<b>7E</b>
		Floating point number status	Floating point number = 33174.3672951 (64 bit float)							

Register	Value (hex)
6700	0080
6701	4168
6702	5F26
6703	352A
6704	FC7E

**Query:** Slave address 01  
 Function 03 03: Read Holding Register  
 Register 1A 2C Register 6700  
 No. Registers 00 05 5 Registers  
 CRC 43 18

**Response:** Slave address 01  
 Function 03 03: Read Holding Register  
 No. Bytes 0A 10 Bytes  
 Status 00 80  
 FLP 41 68 5F 26 35 2A FC 7E 33174.3672951  
 CRC 83 06

### 2.6.9 Device → Modbus Master: reading relay states

The states of the relays are read out via **03 Read Holding Register (4x)**.  
Bit 0 corresponds to relay 1.

**Example: Relay 5 in active state**

**Query:**

Slave address	01	
Function	03	03: Read holding register (4x)
Register	0C 50	Register 3152
No. Registers	00 01	1 Register
CRC	87 4B	

**Response:**

Slave address	01	
Function	03	03: Read holding register (4x)
No. Bytes	02	2 Bytes
Data	00 10	
CRC	B9 88	

Byte 0 state (bit 11-8)	Byte 1 state (bit 7-0)
00000000	00010000
	Bit 4 high Relay 5

Register	Value (hex)
3152	0010

The relay state is determined from the 2 data bytes as follows:

Byte 1:

Bit 0 = Status relay 1  
 Bit 1 = Status relay 2  
 Bit 2 = Status relay 3  
 Bit 3 = Status relay 4  
 Bit 4 = Status relay 5  
 Bit 5 = Status relay 6  
 Bit 6 = Status relay 7  
 Bit 7 = Status relay 8

Byte 0:

Bit 0 = Status relay 9  
 Bit 1 = Status relay 10  
 Bit 2 = Status relay 11  
 Bit 3 = Status relay 12

1 = Active, 0 = Inactive

**Example:**

**"0E07" results in the following relay status:**

Relay 1-3 and relay 10-12 active.

### 2.6.10 Modbus Master → device: set relay (telealarm option)

Relays can be set if they have been set to "Remote" in the device settings. **16 Write Multiple Registers** or **06 Write Single Register** can be used for this purpose.

Relay status:

0: inactive  
1: active

**Example: Setting relay 6 to the active state**

Byte 0	Byte1
RelNo	Status
6	1

Register	Value (hex)
3152	0601

**Query:**

Slave address	05	
Function	10	16: Write Multiple Registers
Register	0C 50	Register 3152
No. Registers	00 01	1 Register
No. Bytes	02	2 Bytes
Data	06 01	
CRC	96 A0	

**Response:**

Slave address	05	
Function	10	16: Write Multiple Registers
Register	0C 50	Register 3152
No. Registers	00 01	1 Register
CRC	03 0C	

### 2.6.11 Modbus Master → device: change limit values

**16 Write Multiple Registers** or **06 Write Single Register** can be used to set the limit values.

Function	Description	Data
0x01	Initialization	
0x02	Accept limit values	
0x03	Change limit value	Limit value number;Value;Time span for gradient;Delay;Value2
0x04	Read out limit value	Limit value settings
0x05	Give reason	Reason text

To change limit values, the following procedure must be followed:

1. Initialize limit value change
2. Change limit values
3. Give a reason for the change, where applicable
4. Accept limit values

### 2.6.11.1 Initializing limit value changes

This prepares the device for changes to the limit values.

**16 Write Multiple Registers** or **06 Write Single Register** can be used for this purpose.

Byte	0	1
<b>Func</b>	1	2A
<b>Limit value</b>		

Register	Value (hex)
3216	012A

**Query:**

Slave address	05	
Function	10	16: Write Multiple Registers
Register	0C 90	Register 3216
No. Registers	00 01	1 Register
No. Bytes	02	2 Bytes
Data	01 2A	
CRC	96 A0	

**Response:**

Slave address	05	
Function	10	16: Write Multiple Registers
Register	0C 90	Register 3216
No. Registers	00 01	1 Register
CRC	03 30	

### 2.6.11.2 Changing the limit values

A limit value in the device is changed, but not yet accepted, with this function.

The values are transmitted, separated by a semicolon (;).

The following structure must be observed: Func limit value [value];[span];[delay];[value2]

[ ] means that this value can also be omitted. In addition, only the values that are to be changed need to be transmitted.

Value ranges:

Field	Value range	Data type
Value / Value2	No restrictions	Floating point
Span	0 to 60s	Integer
Delay	0 to 99999s	Integer

**Example:**

Func	Limit value	Data	Meaning
3	1	5.22;;60	Limit value 1 to 5.22, no span, delay of 60 s
3	2	5.34	Limit value 2 to 5.34
3	3	::10	Limit value 3, delay to 10 seconds
3	4	20;;;50	Limit value 4, inband/outband lower limit value 20, upper limit value 50

If an uneven number of characters is sent, a blank space (0x20) must follow. The blank space is ignored in the device.

**Example: Changing limit value 1 (upper limit value for analog input) to 90.5**

Byte	0	1	2	3	4	5
<b>Func</b>		<b>Limit value</b>	<b>39</b>	<b>30</b>	<b>2E</b>	<b>35</b>
	3	1	'9'	'0'	'.'	'5'

Register	Value (hex)
3216	<b>0301</b>
3217	<b>3930</b>
3218	<b>2E35</b>

**Query:** Slave address 05  
 Function 10 16: Write Multiple Registers  
 Register 0C 90 Register 3216  
 No. Registers 00 03 3 Registers  
 No. Byte 06 6 Bytes  
 Data 03 01 39 30 2E 35  
 CRC 3D FE

**Response:** Slave address 05  
 Function 10 16: Write Multiple Registers  
 Register 0C 90 Register 3216  
 No. Registers 00 03 3 Registers  
 CRC 82 F1

**Example: Changing limit value 3 (gradient for analog input) to 5.7 within 10 seconds**

Byte	0	1	2	3	4	5	6	7
<b>Func</b>		<b>Limit value</b>	<b>35</b>	<b>2E</b>	<b>37</b>	<b>3B</b>	<b>31</b>	<b>30</b>
	3	3	'5'	'.'	'7'	'.'	'1'	'0'

Register	Value (hex)
3216	<b>0303</b>
3217	<b>352E</b>
3218	<b>373B</b>
3219	<b>3130</b>

**Query:** Slave address 05  
 Function 10 16: Write Multiple Registers  
 Register 0C 90 Register 3216  
 No. Registers 00 04 4 Registers  
 No. Byte 08 8 Bytes  
 Data 03 03 35 2E 37 3B 31 30  
 CRC 94 BF

**Response:** Slave address 05  
 Function 10 16: Write Multiple Registers  
 Register 0C 90 Register 3216  
 No. Registers 00 04 4 Registers  
 CRC C3 33

### 2.6.11.3 Specifying the reason for changing the limit value

Before you save the limit value change, you can enter a reason for the change, which is saved in the event list. If no reason is specified, the message "Limit values was changed" is entered in the event list.

Texts (ASCII table) can be transmitted. The maximum length of a text is 30 characters.

The texts must be written via **16 Write Multiple Registers**, with 2 characters per register.

If an uneven number of characters is sent, a blank space (0x20) must follow. The blank space does not appear in the event list.

Byte	0	1
	<b>Func</b>	<b>Limit value</b>
	5	x

**Query:**

Slave address	05	
Function	10	10: Write Multiple Registers
Register	0C 90	Register 3216
No. Registers	00 07	7 Registers
No. Bytes	0E	14 Bytes
Data	05 01	Function 5, Default 1
Text	52 65 61 73 6F 6E 20 77 68 79 21 20	"Reason why!"
CRC	62 64	

**Response:**

Slave address	05	
Function	10	10: Write Multiple Registers
Register	0C 90	Register 3216
No. Registers	00 07	7 Registers
CRC	83 32	

### 2.6.11.4 Accepting limit values

This function is used to accept the modified limit values in the device and save them in the device settings. **16 Write Multiple Registers** or **06 Write Single Register** can be used for this purpose.

Byte	0	1
	<b>Func</b>	<b>Padding byte</b>
	2	2A

Register	Value (hex)
3216	022A

**Query:**

Slave address	05	
Function	10	16: Write Multiple Registers
Register	0C 90	Register 3216
No. Registers	00 01	1 Register
No. Bytes	02	2 Bytes
Data	02 2A	
CRC	C5 7F	

**Response:**

Slave address	05	
Function	10	16: Write Multiple Registers
Register	0C 90	Register 3216
No. Registers	00 01	1 Register
CRC	03 30	

### 2.6.11.5 Reading out the communication status

The status of the last limit function performed can be read out here. A prerequisite is that limit value read-out is not activated (see 2.6.11.2).

**Example: incorrect function addressed**

**Query:**

Slave address	05	
Function	03	03: Read holding register (4x)
Register	0C 90	Register 3216
No. Registers	00 01	1 Register
CRC	86 F3	

**Response:**

Slave address	05	
Function	03	03: Read holding register (4x)
No. Bytes	02	2 Bytes
Data	00 01	
CRC	88 44	

Register	Value (hex)
3216	0001

Communication status:

- 0: OK
- 1: Incorrect function number or limit value number
- 2: Data missing
- 3: Limit value not active
- 4: Value outside the permitted range
- 5: Function currently not possible
- 9: Error



### 2.6.11.6 Reading out limit values

The number of the first desired limit value is transferred to activate the function. The limit value number is set to the next activated limit value

As a result of activating this function, value readout from Modbus address 3216 onwards no longer returns the communication status. Instead the limit value settings of the specific limits are returned in 8 registers.

Byte	0	1
	<b>Func</b>	<b>Limit value</b>
	4	1

**Query:**

Slave address	05	
Function	06	06: Write Single Register
Register	0C 90	Register 3216
Data	04 01	Function 4, Limit value 1
CRC	48 33	

**Response:**

Slave address	05	
Function	06	06: Write Single Register
Register	0C 90	Register 3216
Data	04 01	Function 4, Limit value 1
CRC	48 33	

After this, the desired limit value settings (8 registers) are read out from register 3216 onwards.

If the transmitted limit value number is outside the limit value limits (1-60), the following error appears in the communication status:

**Query:**

Slave address	05	
Function	03	03: Read holding register (4x)
Register	0C 90	Register 3216
No. Registers	00 08	8 Registers
CRC	46 F5	

**Response:**

Slave address	05	
Function	03	03: Read Holding register (4x)
No. Bytes	10	16 Bytes
Data	00 01	Incorrect limit value number
Data	00 00 00 00 00 00 00 00 00 00 00 00 00 00	
CRC	D4 69	

Otherwise, the communication status query delivers the settings for a limit value (see also 2.6.11.7 Tables and definitions):

**Response:**

Slave address	05	
Function	03	03: Read Holding register (4x)
No. Bytes	10	16 Bytes
LV,LVType	01 10	Limit value 1, Limit value inband
Value	C9 74 23 F0	lower limit value -99999
Span	00 00	Time span for gradient (not required here)
Delay	00 00 00 04	4 seconds
Value2	42 F6 E6 66	upper limit value 123.45
CRC	F5 F0	

After every scan, the limit value number is set to the next activated limit value and can be read out with the next query. Following the last activated limit value, the cycle starts again with the first activated limit value.

If no limit values are activated, all data are set to 0 in the response.

To deactivate the function, 255 is transmitted as the limit value number or a function not equal to 4 is performed.

**2.6.11.7 Tables and definitions**

**LV:** Values between 1 and 60

**LVType:**

0	Switched off
1	Upper limit value
2	Lower limit value
3-6	Analysis 1-4
7	Gradient dy/dt
8-11	Limit value stats analysis: frequency
12-15	Limit value stats analysis: duration
16	Inband
17	Outband

**Value/value 2:** limit value as floating point number (IEEE754, Big Endian)

**Span:** time span for gradient (1-60 s)

**Delay:** delay time in seconds (0-99999).

**2.6.12 Modbus Master → device: transmit text**

Texts (ASCII table) can be saved in the device event list. The maximum length of a text is 40 characters.

The texts must be written via **16 Write Multiple Registers**, with 2 characters per register.

If an uneven number of characters is sent, a blank space (0x20) must follow. The blank space does not appear in the event list.

Channel	Reg. dec.	Reg. hex.	Length, byte
Text	3024	B0	Max. 40

Tab. 15: Register address for the transmission of a text: Modbus Master → device

Byte	0	1	2	3	4	5
	<b>41</b>	<b>42</b>	<b>43</b>	<b>44</b>	<b>45</b>	<b>20</b>
	'A'	'B'	'C'	'D'	'E'	,

Register	Value (hex)
3024	<b>4142</b>
3025	<b>4344</b>
3026	<b>4520</b>

**Example: Generating the text "ABCDE "**

**Query:**

Slave address	05	
Function	10	16: Write Multiple Registers
Register	0B D0	Register 3024
No. Registers	00 03	3 Registers
No. Bytes	06	6 Bytes
Data	41 42 43 44 45 20	
CRC	D8 4E	

**Response:** Slave address 05  
 Function 10 16: Write Multiple Registers  
 Register 0B D0 Register 3024  
 No. Registers 00 03 3 Registers  
 CRC 82 51

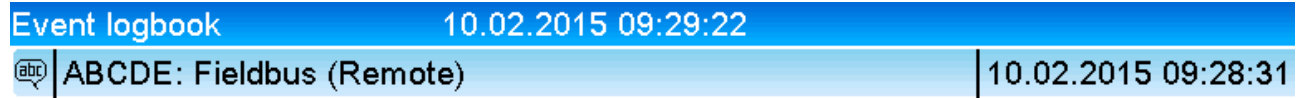


Fig. 6: Text entered in the event list

### 2.6.13 Modbus Master → device: batch data (batch option)

Batches can be started and ended. The batch name, batch designation, batch number and preset counter for stopping the batch can also be set. The maximum length of the texts (ASCII) is 30 characters.

The functions and texts must be written via **16 Write Multiple Registers**.

If an uneven number of characters is sent, a blank space (0x20) must follow. The blank space is ignored in the device.

Function	Description	Data
0x01	Start batch	Batch (1 to 4), ID, name
0x02	Stop batch	Batch (1 to 4), ID, name
0x03	Batch designation	Batch (1 to 4), text (max. 30 characters)
0x04	Batch name	Batch (1 to 4), text (max. 30 characters)
0x05	Batch number	Batch (1 to 4), text (max. 30 characters)
0x06	Preset counter	Batch (1 to 4), text (max. 8 characters)

#### 2.6.13.1 Start batch

If the user administration function is enabled, an ID (max. 8 characters) and a name (max. 20 characters) must be transmitted. The ID and name must be separated by ';'. If an uneven number of characters is sent, a blank space (0x20) must follow (see 2.6.13.2 Ending the batch).

**Example: Starting batch 2 (without user administration)**

Byte	0	1
Func		No
	1	2

Register	Value (hex)
3088	0102

**Query:** Slave address 05  
 Function 10 16: Write Multiple Registers  
 Register 0C 10 Register 3088  
 No. Registers 00 01 1 Register  
 No. Bytes 02 2 Bytes  
 Data 01 02  
 CRC D2 51

**Response:** Slave address 05  
 Function 10 16: Write Multiple Registers  
 Register 0C 10 Register 3088  
 No. Registers 00 01 1 Register  
 CRC 02 D8

The message "Batch 2 started" is saved in the event list. This message also appears on the screen for a few seconds.

### 2.6.13.2 Ending the batch

If the user administration function is enabled, an ID (max. 8 characters) and a name (max. 20 characters) must be transmitted. The ID and name must be separated by a semicolon ';'. If an uneven number of characters is sent, a blank space (0x20) must follow.

**Example: Ending batch 2, user administration active (ID: "IDSPS", Name "Remotex")**

Byte	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Func	No	49	44	53	50	53	3B	52	65	6D	6F	74	65	58	20	
		2	T	'D'	'S'	'P'	'S'	;	'R'	'e'	'm'	'o'	't'	'e'	'X'	' '

Register	Value (hex)
3088	0202
3089	4944
3090	5350
3091	533B
3092	5265
3093	6D6F
3094	7465
3095	5820

**Query:**

Slave address	05	
Function	10	16: Write Multiple Registers
Register	0C 10	Register 3088
No. Registers	00 08	8 Registers
No. Bytes	10	16 Bytes
Data	02 02 49 44 53 59 53 3B 52 65 6D 6F 74 65 58 20	
CRC	D3D6	

**Response:**

Slave address	05	
Function	10	16: Write Multiple Registers
Register	0C 10	Register 3088
No. Registers	00 08	8 Registers
CRC	C2 DE	

The message "Batch 2 ended" and "Remote (IDSPS)" is saved in the event list. This message also appears on the screen for a few seconds.

### 2.6.13.3 Configuring the batch designation

Can only be configured if the batch has not been started. Does not need to be configured if this is not required by the device settings.

**Example:** "Identifier" batch designation for batch 2

Byte	0	1	2	3	4	5	6	7	8	9	10	11
Func	No	49	64	65	6E	74	69	66	69	65	72	
		T	'd'	'e'	'n'	't'	'i'	'f'	'i'	'e'	'r'	

Register	Value (hex)
3088	0302
3089	5964
3090	656E
3091	7469
3092	6669
3093	6572

**Query:**

Slave address	05	
Function	10	16: Write Multiple Registers
Register	0C 10	Register 3088
No. Registers	00 06	6 Registers
No. Bytes	0B	12 Bytes
Data	03 02 59 64 65 6E 74 69 66 69 65 72	
CRC	0E 20	

**Response:**

Slave address	05	
Function	10	16: Write Multiple Registers
Register	0C 10	Register 3088
No. Registers	00 06	6 Registers
CRC	43 1A	

### 2.6.13.4 Configuring the batch name

Can only be configured if the batch has not been started. Does not need to be configured if this is not required by the device settings.

**Example:** "Name" batch name for batch 2

Byte	0	1	2	3	4	5
Func	No	4E	61	6D	65	
	4	2	'N'	'a'	'm'	'e'

Register	Value (hex)
3088	0402
3089	4E61
3090	6D65

**Query:**

Slave address	05	
Function	10	16: Write Multiple Registers
Register	0C 10	Register 3088
No. Registers	00 03	3 Registers
No. Bytes	06	6 Bytes
Data	04 02 4E 61 6D 65	
CRC	04 C8	

**Response:**

Slave address	05	
Function	10	16: Write Multiple Registers
Register	0C 10	Register 3088
No. Registers	00 03	3 Registers
CRC	83 19	

### 2.6.13.5 Configuring the batch number

Can only be configured if the batch has not been started. Does not need to be configured if this is not required by the device settings.

**Example:** "Num" batch number for batch 2

Byte	0	1	2	3	4	5
<b>Func</b>	<b>No</b>	<b>4E</b>	<b>75</b>	<b>6D</b>	<b>20</b>	
5	2	'N'	'u'	'm'	','	

Register	Value (hex)
3088	<b>0502</b>
3089	<b>4E75</b>
3090	<b>6D20</b>

**Query:**

Slave address	05	
Function	10	16: Write Multiple Registers
Register	0C 10	Register 3088
No. Registers	00 03	3 Registers
No. Bytes	06	6 Bytes
Data	05 02 4E 75 6D 20	
CRC	84 EE	

**Response:**

Slave address	05	
Function	10	16: Write Multiple Registers
Register	0C 10	Register 3088
No. Registers	00 03	3 Registers
CRC	83 19	

### 2.6.13.6 Setting the preset counter

Can only be set if the batch has not been started. Does not need to be set if this is not required in the device settings.

- Maximum 8 characters (including '!')
- Exponential function is permitted, e.g. "1.23E-2"
- Positive numbers only

**Example:**      **Preset counter to 12.345 for batch 2**

Byte	0	1	2	3	4	5	6	7
<b>Func</b>	<b>No</b>	<b>31</b>	<b>32</b>	<b>2E</b>	<b>33</b>	<b>34</b>	<b>35</b>	
6	2	'1'	'2'	'.'	'3'	'4'	'5'	

Register	Value (hex)
3088	0602
3090	3132
3091	2E33
3092	3435

**Query:**

Slave address	05	
Function	10	16: Write Multiple Registers
Register	0C 10	Register 3088
No. Registers	00 04	4 Registers
No. Byte	08	8 Bytes
Data	06 02 31 32 2E 33 34 35	
CRC	D3 B5	

**Response:**

Slave address	05	
Function	10	16: Write Multiple Registers
Register	0C 10	Register 3088
No. Registers	00 04	4 Registers
CRC	C2 DB	



### 2.6.13.7 Reading out the batch status

The status of every batch and the last communication status can be read out here.

**Example: Batch 2 started, communication status "OK"**

**Query:** Slave address 05  
 Function 03 03: Read holding register (4x)  
 Register 0C 10 Register 3088  
 No. Registers 00 03 3 Registers  
 CRC 06 DA

**Response:** Slave address 05  
 Function 3 03: Read Holding register (4x)  
 Register 0C 10 Register 3088  
 No. Bytes 6 6 Bytes  
 Data 00 00 00 01 00 00  
 CRC 42 75

Byte	0	1	2	3	4	5
		<b>Comm. status</b>	<b>Status batch 1</b>	<b>Status batch 2</b>	<b>Status batch 3</b>	<b>Status batch 4</b>
	0	0	0	1	0	0

Register	Value (hex)
3088	0000
3090	0001
3091	0000

If, for example, a batch number is set even though the batch is already running, the value 0x0003 would appear in register 3088.

Communication status:

- 0: OK
- 1: Not all the required data have been transmitted (mandatory entries)
- 2: No appropriate user is logged in
- 3: Batch already running
- 4: Batch not configured
- 5: Batch controlled by control input
- 7: Automatic batch number active
- 9: Error, text contained non-displayable characters, text too long, incorrect batch number  
Function number outside the range

Batch status:

- 0: Batch inactive
- 1: Batch active

## 2.6.14 Structure of the process values

### 2.6.14.1 32-bit floating point number (IEEE-754)

Octet	8	7	6	5	4	3	2	1
0	Sign	(E) $2^7$	(E) $2^6$					(E) $2^1$
1	(E) $2^0$	(M) $2^{-1}$	(M) $2^{-2}$					(M) $2^{-7}$
2	(M) $2^{-8}$							(M) $2^{-15}$
3	(M) $2^{-16}$							(M) $2^{-23}$

Sign = 0: positive number

Sign = 1: negative number

E = exponent 8 bit, M = mantissa 23 bit

$$Zahl = -1^{VZ} \cdot (1 + M) \cdot 2^{E-127}$$

$$Zahl = -1^{VZ} \cdot \left(1 + \sum_{i=1}^{23} b_{23-i} 2^{-i}\right) \cdot 2^{E-127}$$

Example:

40 F0 00 00 h = 0100 0000 1111 0000 0000 0000 0000 0000 b

$$\begin{aligned} \text{Value} &= -1^0 \cdot 2^{129-127} \cdot (1 + 2^{-1} + 2^{-2} + 2^{-3}) \\ &= 1 \cdot 2^2 \cdot (1 + 0,5 + 0,25 + 0,125) \\ &= 1 \cdot 4 \cdot 1,875 = 7,5 \end{aligned}$$

Byte	0	1	2	3	4	5
	00	80	40	F0	00	00
Limit value violations		Floating point number	Floating point number = 7.5			
		number status				

### 2.6.14.2 64-bit floating point number (IEEE-754)

Octet	8	7	6	5	4	3	2	1
0	Sign	(E) $2^{10}$	(E) $2^9$					(E) $2^4$
1	(E) $2^3$	(E) $2^2$	(E) $2^1$	(E) $2^0$	(M) $2^{-1}$	(M) $2^{-2}$	(M) $2^{-3}$	(M) $2^{-4}$
2	(M) $2^{-5}$							(M) $2^{-12}$
3	(M) $2^{-13}$							(M) $2^{-20}$
4	(M) $2^{-21}$							(M) $2^{-28}$
5	(M) $2^{-29}$							(M) $2^{-36}$
6	(M) $2^{-37}$							(M) $2^{-44}$
7	(M) $2^{-45}$							(M) $2^{-52}$

Sign = 0: positive number

Sign = 1: negative number

E = exponent 11 bit, M = mantissa 52 bit

$$Zahl = -1^{VZ} \cdot (1 + M) \cdot 2^{E-1023}$$

$$Zahl = -1^{VZ} \cdot \left(1 + \sum_{i=1}^{52} b_{52-i} 2^{-i}\right) \cdot 2^{E-1023}$$

Example:                   40 1E 00 00 00 00 00 00 h  
 = 0 100 0000 0001 1110 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 b  
 Value =  $-1^0 \cdot 2^{1025-1023} \cdot (1 + 2^{-1} + 2^{-2} + 2^{-3})$   
           =  $1 \cdot 2^2 \cdot (1 + 0,5 + 0,25 + 0,125)$   
           =  $1 \cdot 4 \cdot 1,875 = 7,5$

Byte	0	1	2	3	4	5	6	7	8	9
	00	80	40	1E	00	00	00	00	0	0
		Floating point number status	Floating point number =7.5							

**2.6.14.3 Limit value violations**

**Device → Modbus Master**

The states of the first 8 limit values that are assigned to the channel are entered here.

Bit 0: 1st assigned limit value

...

Bit 7: 8th assigned limit value

Bit x = 1: limit value violated  
 = 0: limit value not violated

Example:

If universal input 1 is assigned a limit value for the instantaneous value and a limit value for analysis 1, the 2 limit value states are indicated in bit 0 and bit 1 in the measured value of universal input 1 (register 200) and integrated universal input 1 (register 800).

Byte	0	1	2	3	4	5
	02	80	40	F0	00	00
	Limit value violations	Floating point number status	Floating point number = 7.5			

Bit 0.0 = 0: 1st assigned limit value not violated, here limit value for instantaneous value  
 Bit 0.1 = 1: 2nd assigned limit value violated, here limit value for integrated value

#### 2.6.14.4 Status of the floating point number

##### Device → Modbus Master

0x01	Cable open circuit
0x02	Input signal too high
0x03	Input signal too low
0x04	Invalid measured value
0x06	Error value
0x07	Sensor/input error
0x08	No value present (e.g. while measurement is initialized)
0x40	Value is uncertain (error value), no limit value violation
0x41	Value is uncertain (error value), lower limit value violation or gradient decreasing
0x42	Value is uncertain (error value), upper limit value violation or gradient increasing
0x43	Value is uncertain (error value), upper and lower limit value violation or inband/outband
0x80	Value is OK, no limit value violation
0x81	Value is OK, lower limit value violation or gradient decreasing
0x82	Value is OK, upper limit value violation or gradient increasing
0x83	Value is OK, upper and lower limit value violation or inband/outband

##### Modbus Master → device

0x00..0x3F: value invalid  
0x40..0x7F: value uncertain  
0x80..0xFF: value OK

### 3 Overview of registers



The register addresses are all to the base 0, i.e. they correspond to the value that is transmitted in the Modbus protocol.

Register	Value	Format	Access
200	Universal 1	Status + 32 Bit Float	R/W
203	Universal 2	Status + 32 Bit Float	R/W
206	Universal 3	Status + 32 Bit Float	R/W
209	Universal 4	Status + 32 Bit Float	R/W
212	Universal 5	Status + 32 Bit Float	R/W
215	Universal 6	Status + 32 Bit Float	R/W
218	Universal 7	Status + 32 Bit Float	R/W
221	Universal 8	Status + 32 Bit Float	R/W
224	Universal 9	Status + 32 Bit Float	R/W
227	Universal 10	Status + 32 Bit Float	R/W
230	Universal 11	Status + 32 Bit Float	R/W
233	Universal 12	Status + 32 Bit Float	R/W
236	Universal 13	Status + 32 Bit Float	R/W
239	Universal 14	Status + 32 Bit Float	R/W
242	Universal 15	Status + 32 Bit Float	R/W
245	Universal 16	Status + 32 Bit Float	R/W
248	Universal 17	Status + 32 Bit Float	R/W
251	Universal 18	Status + 32 Bit Float	R/W
254	Universal 19	Status + 32 Bit Float	R/W
257	Universal 20	Status + 32 Bit Float	R/W
260	Universal 21	Status + 32 Bit Float	R/W
263	Universal 22	Status + 32 Bit Float	R/W
266	Universal 23	Status + 32 Bit Float	R/W
269	Universal 24	Status + 32 Bit Float	R/W
272	Universal 25	Status + 32 Bit Float	R/W
275	Universal 26	Status + 32 Bit Float	R/W
278	Universal 27	Status + 32 Bit Float	R/W
281	Universal 28	Status + 32 Bit Float	R/W
284	Universal 29	Status + 32 Bit Float	R/W
287	Universal 30	Status + 32 Bit Float	R/W
290	Universal 31	Status + 32 Bit Float	R/W
293	Universal 32	Status + 32 Bit Float	R/W
296	Universal 33	Status + 32 Bit Float	R/W
299	Universal 34	Status + 32 Bit Float	R/W
302	Universal 35	Status + 32 Bit Float	R/W
305	Universal 36	Status + 32 Bit Float	R/W
308	Universal 37	Status + 32 Bit Float	R/W
311	Universal 38	Status + 32 Bit Float	R/W
314	Universal 39	Status + 32 Bit Float	R/W
317	Universal 40	Status + 32 Bit Float	R/W
800	Universal 1 totalizer	Status + 32 Bit Float	R
803	Universal 2 totalizer	Status + 32 Bit Float	R
806	Universal 3 totalizer	Status + 32 Bit Float	R
809	Universal 4 totalizer	Status + 32 Bit Float	R
812	Universal 5 totalizer	Status + 32 Bit Float	R
815	Universal 6 totalizer	Status + 32 Bit Float	R
818	Universal 7 totalizer	Status + 32 Bit Float	R
821	Universal 8 totalizer	Status + 32 Bit Float	R
824	Universal 9 totalizer	Status + 32 Bit Float	R
827	Universal 10 totalizer	Status + 32 Bit Float	R
830	Universal 11 totalizer	Status + 32 Bit Float	R
833	Universal 12 totalizer	Status + 32 Bit Float	R
836	Universal 13 totalizer	Status + 32 Bit Float	R
839	Universal 14 totalizer	Status + 32 Bit Float	R
842	Universal 15 totalizer	Status + 32 Bit Float	R
845	Universal 16 totalizer	Status + 32 Bit Float	R
848	Universal 17 totalizer	Status + 32 Bit Float	R
851	Universal 18 totalizer	Status + 32 Bit Float	R
854	Universal 19 totalizer	Status + 32 Bit Float	R
857	Universal 20 totalizer	Status + 32 Bit Float	R
860	Universal 21 totalizer	Status + 32 Bit Float	R

863	Universal 22 totalizer	Status + 32 Bit Float	R
866	Universal 23 totalizer	Status + 32 Bit Float	R
869	Universal 24 totalizer	Status + 32 Bit Float	R
872	Universal 25 totalizer	Status + 32 Bit Float	R
875	Universal 26 totalizer	Status + 32 Bit Float	R
878	Universal 27 totalizer	Status + 32 Bit Float	R
881	Universal 28 totalizer	Status + 32 Bit Float	R
884	Universal 29 totalizer	Status + 32 Bit Float	R
887	Universal 30 totalizer	Status + 32 Bit Float	R
890	Universal 31 totalizer	Status + 32 Bit Float	R
893	Universal 32 totalizer	Status + 32 Bit Float	R
896	Universal 33 totalizer	Status + 32 Bit Float	R
899	Universal 34 totalizer	Status + 32 Bit Float	R
902	Universal 35 totalizer	Status + 32 Bit Float	R
905	Universal 36 totalizer	Status + 32 Bit Float	R
908	Universal 37 totalizer	Status + 32 Bit Float	R
911	Universal 38 totalizer	Status + 32 Bit Float	R
914	Universal 39 totalizer	Status + 32 Bit Float	R
917	Universal 40 totalizer	Status + 32 Bit Float	R
1200	Digital 1 state	2 Byte	R/W
1201	Digital 2 state	2 Byte	R/W
1202	Digital 3 state	2 Byte	R/W
1203	Digital 4 state	2 Byte	R/W
1204	Digital 5 state	2 Byte	R/W
1205	Digital 6 state	2 Byte	R/W
1206	Digital 7 state	2 Byte	R/W
1207	Digital 8 state	2 Byte	R/W
1208	Digital 9 state	2 Byte	R/W
1209	Digital 10 state	2 Byte	R/W
1210	Digital 11 state	2 Byte	R/W
1211	Digital 12 state	2 Byte	R/W
1240	Digital 1-16 states	2 Byte	R/W
1241	Digital 17-20 states	2 Byte	R/W
1300	Digital 1 totalizer	Status + 32 Bit Float	R
1303	Digital 2 totalizer	Status + 32 Bit Float	R
1306	Digital 3 totalizer	Status + 32 Bit Float	R
1309	Digital 4 totalizer	Status + 32 Bit Float	R
1312	Digital 5 totalizer	Status + 32 Bit Float	R
1315	Digital 6 totalizer	Status + 32 Bit Float	R
1318	Digital 7 totalizer	Status + 32 Bit Float	R
1321	Digital 8 totalizer	Status + 32 Bit Float	R
1324	Digital 9 totalizer	Status + 32 Bit Float	R
1327	Digital 10 totalizer	Status + 32 Bit Float	R
1330	Digital 11 totalizer	Status + 32 Bit Float	R
1333	Digital 12 totalizer	Status + 32 Bit Float	R
1336	Digital 13 totalizer	Status + 32 Bit Float	R
1339	Digital 14 totalizer	Status + 32 Bit Float	R
1342	Digital 15 totalizer	Status + 32 Bit Float	R
1345	Digital 16 totalizer	Status + 32 Bit Float	R
1348	Digital 17 totalizer	Status + 32 Bit Float	R
1351	Digital 18 totalizer	Status + 32 Bit Float	R
1354	Digital 19 totalizer	Status + 32 Bit Float	R
1357	Digital 20 totalizer	Status + 32 Bit Float	R
1500	Math 1	Status + 32 Bit Float	R
1503	Math 2	Status + 32 Bit Float	R
1506	Math 3	Status + 32 Bit Float	R
1509	Math 4	Status + 32 Bit Float	R
1512	Math 5	Status + 32 Bit Float	R
1515	Math 6	Status + 32 Bit Float	R
1518	Math 7	Status + 32 Bit Float	R
1521	Math 8	Status + 32 Bit Float	R
1524	Math 9	Status + 32 Bit Float	R
1527	Math 10	Status + 32 Bit Float	R
1530	Math 11	Status + 32 Bit Float	R
1533	Math 12	Status + 32 Bit Float	R
1700	Math 1 totalizer	Status + 32 Bit Float	R
1703	Math 2 totalizer	Status + 32 Bit Float	R
1706	Math 3 totalizer	Status + 32 Bit Float	R
1709	Math 4 totalizer	Status + 32 Bit Float	R
1712	Math 5 totalizer	Status + 32 Bit Float	R

1715	Math 6 totalizer	Status + 32 Bit Float	R
1718	Math 7 totalizer	Status + 32 Bit Float	R
1721	Math 8 totalizer	Status + 32 Bit Float	R
1724	Math 9 totalizer	Status + 32 Bit Float	R
1727	Math 10 totalizer	Status + 32 Bit Float	R
1730	Math 11 totalizer	Status + 32 Bit Float	R
1733	Math 12 totalizer	Status + 32 Bit Float	R
1800	Math 1-4 states	2 Byte	R
3152	Relay states	2 Byte	R
4000	Universal 1	32 bit float	R
4002	Universal 2	32 bit float	R
4004	Universal 3	32 bit float	R
4006	Universal 4	32 bit float	R
4008	Universal 5	32 bit float	R
4010	Universal 6	32 bit float	R
4012	Universal 7	32 bit float	R
4014	Universal 8	32 bit float	R
4016	Universal 9	32 bit float	R
4018	Universal 10	32 bit float	R
4020	Universal 11	32 bit float	R
4022	Universal 12	32 bit float	R
4024	Universal 13	32 bit float	R
4026	Universal 14	32 bit float	R
4028	Universal 15	32 bit float	R
4030	Universal 16	32 bit float	R
4032	Universal 17	32 bit float	R
4034	Universal 18	32 bit float	R
4036	Universal 19	32 bit float	R
4038	Universal 20	32 bit float	R
4040	Universal 21	32 bit float	R
4042	Universal 22	32 bit float	R
4044	Universal 23	32 bit float	R
4046	Universal 24	32 bit float	R
4048	Universal 25	32 bit float	R
4050	Universal 26	32 bit float	R
4052	Universal 27	32 bit float	R
4054	Universal 28	32 bit float	R
4056	Universal 29	32 bit float	R
4058	Universal 30	32 bit float	R
4060	Universal 31	32 bit float	R
4062	Universal 32	32 bit float	R
4064	Universal 33	32 bit float	R
4066	Universal 34	32 bit float	R
4068	Universal 35	32 bit float	R
4070	Universal 36	32 bit float	R
4072	Universal 37	32 bit float	R
4074	Universal 38	32 bit float	R
4076	Universal 39	32 bit float	R
4078	Universal 40	32 bit float	R
4200	Math 1	32 bit float	R
4202	Math 2	32 bit float	R
4204	Math 3	32 bit float	R
4206	Math 4	32 bit float	R
4208	Math 5	32 bit float	R
4210	Math 6	32 bit float	R
4212	Math 7	32 bit float	R
4214	Math 8	32 bit float	R
4216	Math 9	32 bit float	R
4218	Math 10	32 bit float	R
4220	Math 11	32 bit float	R
4222	Math 12	32 bit float	R
5200	Universal 1	Status + 64 Bit Float	R/W
5205	Universal 2	Status + 64 Bit Float	R/W
5210	Universal 3	Status + 64 Bit Float	R/W
5215	Universal 4	Status + 64 Bit Float	R/W
5220	Universal 5	Status + 64 Bit Float	R/W
5225	Universal 6	Status + 64 Bit Float	R/W
5230	Universal 7	Status + 64 Bit Float	R/W
5235	Universal 8	Status + 64 Bit Float	R/W
5240	Universal 9	Status + 64 Bit Float	R/W

5245	Universal 10	Status + 64 Bit Float	R/W
5250	Universal 11	Status + 64 Bit Float	R/W
5255	Universal 12	Status + 64 Bit Float	R/W
5260	Universal 13	Status + 64 Bit Float	R/W
5265	Universal 14	Status + 64 Bit Float	R/W
5270	Universal 15	Status + 64 Bit Float	R/W
5275	Universal 16	Status + 64 Bit Float	R/W
5280	Universal 17	Status + 64 Bit Float	R/W
5285	Universal 18	Status + 64 Bit Float	R/W
5290	Universal 19	Status + 64 Bit Float	R/W
5295	Universal 20	Status + 64 Bit Float	R/W
5300	Universal 21	Status + 64 Bit Float	R/W
5305	Universal 22	Status + 64 Bit Float	R/W
5310	Universal 23	Status + 64 Bit Float	R/W
5315	Universal 24	Status + 64 Bit Float	R/W
5320	Universal 25	Status + 64 Bit Float	R/W
5325	Universal 26	Status + 64 Bit Float	R/W
5330	Universal 27	Status + 64 Bit Float	R/W
5335	Universal 28	Status + 64 Bit Float	R/W
5340	Universal 29	Status + 64 Bit Float	R/W
5345	Universal 30	Status + 64 Bit Float	R/W
5350	Universal 31	Status + 64 Bit Float	R/W
5355	Universal 32	Status + 64 Bit Float	R/W
5360	Universal 33	Status + 64 Bit Float	R/W
5365	Universal 34	Status + 64 Bit Float	R/W
5370	Universal 35	Status + 64 Bit Float	R/W
5375	Universal 36	Status + 64 Bit Float	R/W
5380	Universal 37	Status + 64 Bit Float	R/W
5385	Universal 38	Status + 64 Bit Float	R/W
5390	Universal 39	Status + 64 Bit Float	R/W
5395	Universal 40	Status + 64 Bit Float	R/W
5800	Universal 1 totalizer	Status + 64 Bit Float	R
5805	Universal 2 totalizer	Status + 64 Bit Float	R
5810	Universal 3 totalizer	Status + 64 Bit Float	R
5815	Universal 4 totalizer	Status + 64 Bit Float	R
5820	Universal 5 totalizer	Status + 64 Bit Float	R
5825	Universal 6 totalizer	Status + 64 Bit Float	R
5830	Universal 7 totalizer	Status + 64 Bit Float	R
5835	Universal 8 totalizer	Status + 64 Bit Float	R
5840	Universal 9 totalizer	Status + 64 Bit Float	R
5845	Universal 10 totalizer	Status + 64 Bit Float	R
5850	Universal 11 totalizer	Status + 64 Bit Float	R
5855	Universal 12 totalizer	Status + 64 Bit Float	R
5860	Universal 13 totalizer	Status + 64 Bit Float	R
5865	Universal 14 totalizer	Status + 64 Bit Float	R
5870	Universal 15 totalizer	Status + 64 Bit Float	R
5875	Universal 16 totalizer	Status + 64 Bit Float	R
5880	Universal 17 totalizer	Status + 64 Bit Float	R
5885	Universal 18 totalizer	Status + 64 Bit Float	R
5890	Universal 19 totalizer	Status + 64 Bit Float	R
5895	Universal 20 totalizer	Status + 64 Bit Float	R
5900	Universal 21 totalizer	Status + 64 Bit Float	R
5905	Universal 22 totalizer	Status + 64 Bit Float	R
5910	Universal 23 totalizer	Status + 64 Bit Float	R
5915	Universal 24 totalizer	Status + 64 Bit Float	R
5920	Universal 25 totalizer	Status + 64 Bit Float	R
5925	Universal 26 totalizer	Status + 64 Bit Float	R
5930	Universal 27 totalizer	Status + 64 Bit Float	R
5935	Universal 28 totalizer	Status + 64 Bit Float	R
5940	Universal 29 totalizer	Status + 64 Bit Float	R
5945	Universal 30 totalizer	Status + 64 Bit Float	R
5950	Universal 31 totalizer	Status + 64 Bit Float	R
5955	Universal 32 totalizer	Status + 64 Bit Float	R
5960	Universal 33 totalizer	Status + 64 Bit Float	R
5965	Universal 34 totalizer	Status + 64 Bit Float	R
5970	Universal 35 totalizer	Status + 64 Bit Float	R
5975	Universal 36 totalizer	Status + 64 Bit Float	R
5980	Universal 37 totalizer	Status + 64 Bit Float	R
5985	Universal 38 totalizer	Status + 64 Bit Float	R
5990	Universal 39 totalizer	Status + 64 Bit Float	R



5995	Universal 40 totalizer	Status + 64 Bit Float	R
6300	Digital 1 totalizer	Status + 64 Bit Float	R
6305	Digital 2 totalizer	Status + 64 Bit Float	R
6310	Digital 3 totalizer	Status + 64 Bit Float	R
6315	Digital 4 totalizer	Status + 64 Bit Float	R
6320	Digital 5 totalizer	Status + 64 Bit Float	R
6325	Digital 6 totalizer	Status + 64 Bit Float	R
6330	Digital 7 totalizer	Status + 64 Bit Float	R
6335	Digital 8 totalizer	Status + 64 Bit Float	R
6340	Digital 9 totalizer	Status + 64 Bit Float	R
6345	Digital 10 totalizer	Status + 64 Bit Float	R
6350	Digital 11 totalizer	Status + 64 Bit Float	R
6355	Digital 12 totalizer	Status + 64 Bit Float	R
6360	Digital 13 totalizer	Status + 64 Bit Float	R
6365	Digital 14 totalizer	Status + 64 Bit Float	R
6370	Digital 15 totalizer	Status + 64 Bit Float	R
6375	Digital 16 totalizer	Status + 64 Bit Float	R
6380	Digital 17 totalizer	Status + 64 Bit Float	R
6385	Digital 18 totalizer	Status + 64 Bit Float	R
6390	Digital 19 totalizer	Status + 64 Bit Float	R
6395	Digital 20 totalizer	Status + 64 Bit Float	R
6700	Math 1 totalizer	Status + 64 Bit Float	R
6705	Math 2 totalizer	Status + 64 Bit Float	R
6710	Math 3 totalizer	Status + 64 Bit Float	R
6715	Math 4 totalizer	Status + 64 Bit Float	R
6720	Math 5 totalizer	Status + 64 Bit Float	R
6725	Math 6 totalizer	Status + 64 Bit Float	R
6730	Math 7 totalizer	Status + 64 Bit Float	R
6735	Math 8 totalizer	Status + 64 Bit Float	R
6740	Math 9 totalizer	Status + 64 Bit Float	R
6745	Math 10 totalizer	Status + 64 Bit Float	R
6750	Math 11 totalizer	Status + 64 Bit Float	R
6755	Math 12 totalizer	Status + 64 Bit Float	R
6800	Universal 1	Status	R
6801	Universal 2	Status	R
6802	Universal 3	Status	R
6803	Universal 4	Status	R
6804	Universal 5	Status	R
6805	Universal 6	Status	R
6806	Universal 7	Status	R
6807	Universal 8	Status	R
6808	Universal 9	Status	R
6809	Universal 10	Status	R
6810	Universal 11	Status	R
6811	Universal 12	Status	R
6812	Universal 13	Status	R
6813	Universal 14	Status	R
6814	Universal 15	Status	R
6815	Universal 16	Status	R
6816	Universal 17	Status	R
6817	Universal 18	Status	R
6818	Universal 19	Status	R
6819	Universal 20	Status	R
6820	Universal 21	Status	R
6821	Universal 22	Status	R
6822	Universal 23	Status	R
6823	Universal 24	Status	R
6824	Universal 25	Status	R
6825	Universal 26	Status	R
6826	Universal 27	Status	R
6827	Universal 28	Status	R
6828	Universal 29	Status	R
6829	Universal 30	Status	R
6830	Universal 31	Status	R
6831	Universal 32	Status	R
6832	Universal 33	Status	R
6833	Universal 34	Status	R
6834	Universal 35	Status	R
6835	Universal 36	Status	R
6836	Universal 37	Status	R

6837	Universal 38	Status	R
6838	Universal 39	Status	R
6839	Universal 40	Status	R
6900	Math 1	Status	R
6901	Math 2	Status	R
6902	Math 3	Status	R
6903	Math 4	Status	R
6904	Math 5	Status	R
6905	Math 6	Status	R
6906	Math 7	Status	R
6907	Math 8	Status	R
6908	Math 9	Status	R
6909	Math 10	Status	R
6910	Math 11	Status	R
6911	Math 12	Status	R
8000	Universal 1	64 bit float	R
8004	Universal 2	64 bit float	R
8008	Universal 3	64 bit float	R
8012	Universal 4	64 bit float	R
8016	Universal 5	64 bit float	R
8020	Universal 6	64 bit float	R
8024	Universal 7	64 bit float	R
8028	Universal 8	64 bit float	R
8032	Universal 9	64 bit float	R
8036	Universal 10	64 bit float	R
8040	Universal 11	64 bit float	R
8044	Universal 12	64 bit float	R
8048	Universal 13	64 bit float	R
8052	Universal 14	64 bit float	R
8056	Universal 15	64 bit float	R
8060	Universal 16	64 bit float	R
8064	Universal 17	64 bit float	R
8068	Universal 18	64 bit float	R
8072	Universal 19	64 bit float	R
8078	Universal 20	64 bit float	R
8080	Universal 21	64 bit float	R
8084	Universal 22	64 bit float	R
8088	Universal 23	64 bit float	R
8092	Universal 24	64 bit float	R
8096	Universal 25	64 bit float	R
8100	Universal 26	64 bit float	R
8104	Universal 27	64 bit float	R
8108	Universal 28	64 bit float	R
8112	Universal 29	64 bit float	R
8116	Universal 30	64 bit float	R
8120	Universal 31	64 bit float	R
8124	Universal 32	64 bit float	R
8128	Universal 33	64 bit float	R
8132	Universal 34	64 bit float	R
8136	Universal 35	64 bit float	R
8140	Universal 36	64 bit float	R
8144	Universal 37	64 bit float	R
8148	Universal 38	64 bit float	R
8152	Universal 39	64 bit float	R
8156	Universal 40	64 bit float	R
8400	Math 1	64 bit float	R
8404	Math 2	64 bit float	R
8408	Math 3	64 bit float	R
8412	Math 4	64 bit float	R
8416	Math 5	64 bit float	R
8420	Math 6	64 bit float	R
8424	Math 7	64 bit float	R
8428	Math 8	64 bit float	R
8432	Math 9	64 bit float	R
8436	Math 10	64 bit float	R
8440	Math 11	64 bit float	R
8424	Math 12	64 bit float	R
3088-3127	Batch		R/W
3024-3043	Texts		W

3216-3235	Limit values		R/W
-----------	--------------	--	-----

## 4 Troubleshooting

### 4.1 Troubleshooting for Modbus TCP

- Is the Ethernet connection between the device and master OK?
- Does the IP address sent by the master match the address configured on the device?
- Do the port configured on the master and the port configured on the device match?

### 4.2 Troubleshooting for Modbus RTU

- Do the device and master have the same baud rate and parity?
- Is the interface correctly wired?
- Does the device address sent by the master match the configured address of the device?
- Do all the slaves on the Modbus have different device addresses?

## 5 List of abbreviations/glossary of terms

Modbus Master: All instruments such as a PLC, PC-plug-in cards etc. that perform a Modbus Master function.

## 6 Index

Baud rate 4

Digital channels 7

Floating point number 42

Floating point number, status 44

Function 4

Inputs 6, 7

LED, status 4

Math channels 7

Outputs 6, 7

Universal channel 6

