

JUMO cTRON 04/08/16

Compact controller
with timer and ramp function



702071



702072



702074

B 70.2070.2.0
Interface Description
Modbus



Contents

1	Introduction	5
1.1	Preface	5
1.2	Typographical conventions	5
2	Protocol description	7
2.1	Master-Slave principle	7
2.2	Transmission mode (RTU)	7
2.3	Device address	8
2.4	Timing of the communication	8
2.5	Structure of the data blocks	11
2.6	Function codes	12
2.6.1	Read n words	12
2.6.2	Write one word	13
2.6.3	Write n words	14
2.7	Transmission format (integer, float and text values)	15
2.8	Checksum (CRC16)	17
2.9	Error processing	18
3	RS485 interface	19
3.1	Connection diagram	19
3.2	Configuration	20
4	Modbus addresses	21
4.1	Process data	21
4.2	Set point values	23
4.3	Controller parameters	23
4.4	Configuration	23
4.5	Commands	24
4.6	RAM memory	25

1.1 Preface

This operating manual is addressed to the system manufacturer with adequate technical background and PC related knowledge.

Please read this operating manual prior to commissioning the device. Keep the manual in a place accessible to all users at all times. Your comments are appreciated and may assist us in improving this manual.

All necessary settings are described in this operating manual. Should problems be encountered during commissioning, please refrain from carrying out any manipulations that are not described in the manual. Any such intervention will jeopardize your warranty rights. Please contact the nearest subsidiary or the head office.

1.2 Typographical conventions

Warning signs:

Caution



This symbol is used when there may be **damage to equipment or data** if the instructions are ignored or not followed correctly!

Note signs:

Note



This symbol is used when your **special attention** is drawn to a remark.

Reference



This symbol refers to **further information** in other manuals, chapters or sections.

Number types:

Hexadecimal number

0x0010

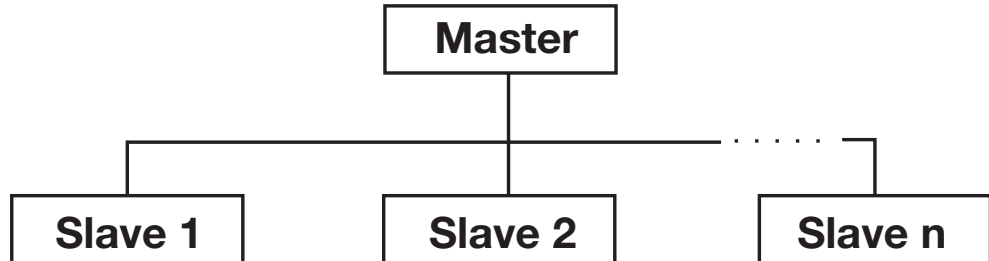
A hexadecimal number is identified by „0x“ preceding the actual number (here: 16 decimal).

1 Introduction

2 Protocol description

2.1 Master-Slave principle

Communication between a master (e.g. PC) and a slave (e.g. measuring and control system) using Modbus takes place according to the master-slave principle, in the form of data request/instruction - response.



The master controls the data exchange, the slaves only have a response function. They are identified by their device address.

2.2 Transmission mode (RTU)

The transmission mode used is the RTU mode (Remote Terminal Unit). Data is transmitted in binary format (hexadecimal) with 8 bits. The LSB (least significant bit) is transmitted first. The ASCII operating mode is not supported.

Data format

The data format describes the structure of a character transmitted. The following data format options are available:

Data word	Parity bit	Stop bit 1/2 bit	Number of bits
8 bits	—	1	9
8 bits	even	1	10
8 bits	odd	1	10
8 bits	—	2	10

2 Protocol description

2.3 Device address

The device address of the slave can be set between 0 and 254. Device address 0 is reserved.



A maximum of 31 slaves can be addressed via the RS485 interface.

There are two different forms of data exchange:

Query

Data request/instruction by the master to a slave via the corresponding device address.

The slave addressed responds.

Broadcast

Instruction by the master to all slaves via the device address 0 (e.g. to transmit a specific value to all slaves).

The connected slaves do not respond. In such a case, the correct acceptance of the value by the slaves should be checked by a subsequent readout at each individual slave.

Data request with the device address 0 is meaningless.

2.4 Timing of the communication

Start and end of a data block are marked by transmission pauses. The maximum permitted interval between two consecutive characters is three times the transmission time required for a single character.

The character transmission time (time required to transmit one single character) depends on the baud rate and the data format used (stop bits and parity bit).

For a data format of 8 data bits, no parity bit and one stop bit, this is:

character transmission time [ms] = 1000 * 9 bit/baud rate

For the other data formats, this is:

character transmission time [ms]
= 1000 * (8 bits+parity bit+stop bit(s)) bit/baud rate

2 Protocol description

Timing

Data request from master transmission time = $n \text{ characters} * 1000 * x \text{ bit/ baud rate}$
Marker for end of data request $3 \text{ characters} * 1000 * x \text{ bit/ baud rate}$
Processing of data request by the slave ($\leq 250\text{ms}$)
Response of the slave transmission time = $n \text{ characters} * 1000 * x \text{ bit/ baud rate}$
Marker for end of response $3 \text{ characters} * 1000 * x \text{ bit/ baud rate}$

Example

Marker for end of data request or end of response for a 10/9 bit data format

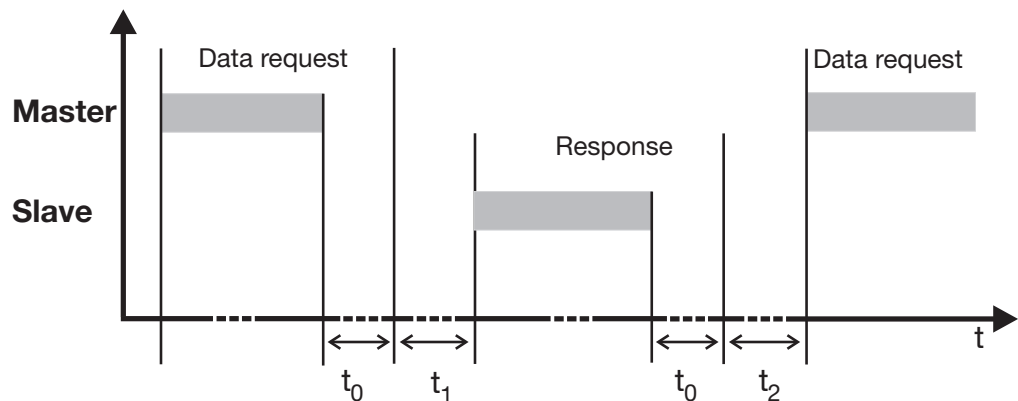
Waiting time = $3 \text{ characters} * 1000 * 10 \text{ bit/ baud rate}$

Baud rate [baud]	Data format [bit]	Waiting time [ms] (3 characters)
38400	10	0.79
	9	0.71
19200	10	1.57
	9	1.41
9600	10	3.13
	9	2.82

2 Protocol description

Timing scheme

A data request runs according to the following timing scheme:



- t_0 End marker = 3 characters
(time depending on the baud rate)
- t_1 This time depends on the internal processing.
The maximum processing time is 250 ms.



A minimum response time can be set in the device under the menu point item „Interface“. This preset time is the minimum waiting time before an answer is transmitted (0...500 ms). If a smaller value is set, then the response time may be longer than the preset value (internal processing takes longer), the device answers as soon as internal processing is completed. The preset time of 0 ms means that the device responds at the maximum possible speed.

The minimum response time, which can be set is required by the RS485 interface in the master, to be able to switch over the interface drivers from transmit to receive.

- t_2 This time is needed by the slave to change from transmit back to receive. The master has to observe this waiting time before presenting a new data request. This time must always be observed, even when the new data request is directed to another device.

RS485 interface: $t_2 = 10\text{ms}$

No data requests from the master are permitted during t_1 and t_2 and during the slave response time. Data requests made during t_1 and t_2 are ignored by the slave. Data requests during the response time will result in the invalidation of all data currently on the bus.

2.5 Structure of the data blocks

All data blocks have the same structure:

Data structure

Slave address	Function code	Data field	Checksum CRC16
1 byte	1 byte	x byte	2 bytes

Each data block contains four fields:

Slave address	device address of a specific slave
Function code	function selection (read, write words)
Data field	contains the following information: <ul style="list-style-type: none">- word address- number of words- word value(s)
Checksum	detection of transmission errors

2 Protocol description

2.6 Function codes

The functions described in the following are available for the readout of measured values, device and process data as well as to write specific data.

Function-overview

Function number	Function	Limitation
0x03 or 0x04	Read n words	max. 32 words (64 bytes)
0x06	Write one word	max. 1 word (2 bytes)
0x10	Write n words	max. 32 words (64 bytes)



Please refer to Chapter 2.9 Error processing, Page 18 if the device does not react to these functions or emits an error code.

2.6.1 Read n words

This function is used to read n ($n \leq 32$) words starting from a specific address.

Data request

Slave address	Function x03 or 0x04	Address first word	Number of words (max. 32)	Checksum CRC16
1 byte	1 byte	2 byte	2 byte	2 byte

Response

Slave address	Function 0x03 or 0x04	Number of bytes read	Word value(s)	Checksum CRC16
1 byte	1 byte	1 byte	x byte	2 byte

Example

Reading the W1 and W2 set point values (2 words each)

Address of first word = 0x3100 (W1 set point value)

Data request:

01	03	3100	0004	4AF5
----	----	------	------	------

Response (values in the Modbus float format):

01	03	08	0000	41C8	0000	4120	4A9E
			Set point value W1 (25.0)	Set point value W2 (10.0)			

2 Protocol description

2.6.2 Write one word

For the Write Word function, the data blocks for instruction and response are identical.

Instruction

Slave address	Function 0x06	Word address	Word value	Checksum CRC16
1 byte	1 byte	2 byte	2 bytes	2 byte

Response

Slave address	Function 0x06	Word address	Word value	Checksum CRC16
1 byte	1 byte	2 byte	2 byte	2 byte

Example

Write the limit value AL of limit comparator 1 = 275.0
(Value = 0x80004389 in the Modbus float format)

Word address = 0x0057

Instruction: Write the first part of the value

01	06	0057	8000	59DA
----	----	------	------	------

Response (as instruction):

01	06	0057	8000	59DA
----	----	------	------	------

Instruction: Write the second part of the value (next word address)

01	06	0058	4389	F88F
----	----	------	------	------

Response (as instruction):

01	06	0058	4389	F88F
----	----	------	------	------

2 Protocol description

2.6.3 Write n words

This function is used to write n ($n \leq 32$) words starting from a specific address.

Instruction

Slave address	Function 0x10	Address first word	Number of words (max. 32)	Number of words	Word value(s)	ChecksumC RC16
1 byte	1 byte	2 byte	2 byte	1 byte	x byte	2 byte

Response

Slave address	Function 0x10	Address first word	Number of words	Checksum CRC16
1 byte	1 byte	2 byte	2 byte	2 byte

Example

Writing the W1 and W2 set point values (2 words each)

Word address = 0x3100 (W1 set point value)

Instruction:

01	10	3100	0004	08	0000	41C8	0000	4120	2A42
					Set point value W1 (25.0)		Set point value W2 (10.0)		

Response:

01	10	3100	0004	CF36
----	----	------	------	------

2 Protocol description

2.7 Transmission format (integer, float and text values)

Integer values Integer values are transmitted via the Modbus in the following format:
The high byte first, followed by the low byte.

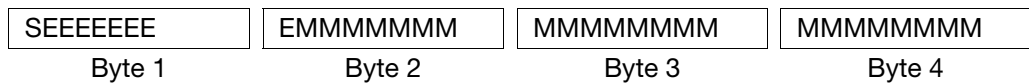
Example Request of the integer value of address 0x0021, if value "4" (word value 0x0004) is written under this address.

Request: 01 03 0021 0001 (+ 2 bytes CRC16)

Response: 01 03 02 **0004** (+ 2 bytes CRC16)

Float values In the case of float values, the Modbus operates with the IEEE-754 standard format (32bits), the only difference being that byte 1 and 2 are changed over with byte 3 and 4.

Single-float format (32bit) as per IEEE 754 standard

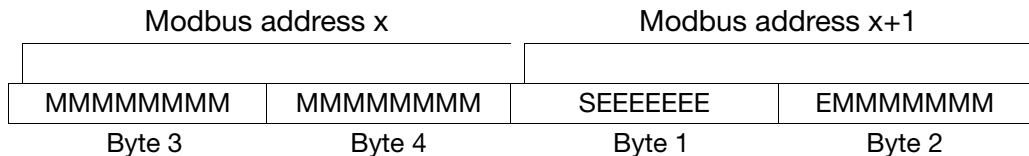


S - sign bit

E - exponent (two's complement)

M - 23bits normalized mantissa

Modbus float format



Example Request of the float value of address 0x0035, if value "550.0" (0x44098000 in IEEE-754 format) is written under this address.

Request: 01 03 0035 0002 (+ 2 bytes CRC16)

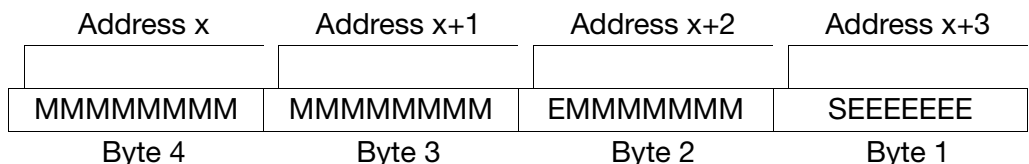
Response: 01 03 04 **8000 4409** (+ 2 bytes CRC16)

Once transmission from the device is completed, the bytes of the float value need to be changed over accordingly.



A large number of compilers (e.g. Microsoft Visual C++) store the float values in the following order:

Float value



Please find out the way float values are stored in your application. After the request, it might be necessary to change the bytes over in the interface program you are using.

2 Protocol description

Character strings (texts)

Character strings (texts) are transmitted in the ASCII format.



To mark the end, the last character to be transmitted can be a "\0" (ASCII code 0x00). Characters after this mark are without significance.

The address tables show the max. possible number of characters in a data type, e.g. "TEXT24" (24 characters). When an end mark is used, then only 23 readable characters are available for the text in this example.

If no end mark is used, the use of the maximum number of characters (e.g. TEXT8 = 8 characters) indicated in the data type is required. This prevents characters still contained in the memory from being appended to the text.

Knowing that the transmission of texts takes place word by word (16 bits), 0x00 is additionally appended where an odd number of characters is used (incl. "\0").

Example for data type TEXT4

Read the text (here: "AbC ") under address 0x0067 (a max. of 4 characters can be saved)

ASCII code for "AbC " (with one space at the end):
0x41, 0x62, 0x43, 0x20

Request: 01 03 0067 0002 (+ 2 byte CRC16)

Slave address = 01

Function = 03, i.e. read n words

Address = 0067

Number of words to be read = 0002, because of the maximum of 4 characters

Response: 01 03 04 **41 62 43 20** (+ 2 byte CRC16)

Slave address = 01

Function = 03, i.e. read n words

Number of bytes read = 04

Variant:

ASCII code for "Ab" (without a space at the end):

0x41, 0x62, 0x00

ASCII 0x00 ("\0") marks the end of the character string.

During transmission, **0x00** is additionally appended to obtain an even number of characters.

Response in this case: 01 03 04 **41 62 00 00** (+ 2 byte CRC16)


2.8 Checksum (CRC16)

The checksum (CRC16) serves to recognize transmission errors. If an error is identified during evaluation, the device concerned does not respond.

Calculation scheme

```

CRC = 0xFFFF
CRC = CRC XOR ByteOfMessage
For (1 to 8)
  CRC = SHR(CRC)
  if (flag shifted right = 1)
  then
    CRC = CRC XOR 0xA001
  else
while (not all ByteOfMessage processed);
  
```

 The low byte of the check sum is the first to be transmitted, then the high byte.

Example

Data request: Read two words, starting at address 0x00CE
(CRC16 = 0x92A5)

07	03	00	CE	00	02	A5	92
						CRC16	

Response: (CRC16 = 0xF5AD)

07	03	04	00	00	41	C8	AD	F5
			Word 1		Word 2		CRC16	

2 Protocol description

2.9 Error processing

Error codes The following error codes exist:

- 1 invalid function
- 2 invalid parameter address or too many words are to be read or written
- 8 write access to parameter denied

Response in the event of an error

Slave address	Function XX OR 80h	Error code	Checksum CRC16
1 byte	1 byte	1 byte	2 bytes

0x80 is used to set the function code to its OR status, i.e. the MSB (most significant bit) is set to 1.

Example

Data request:

01	03	40	00	00	04	CRC16
----	----	----	----	----	----	-------

Response (with error code 2):

01	83	02	CRC16
----	----	----	-------

Special cases

The slave not responding can have the following causes:

- the baud rate and/or data format of Master and Slave are not compatible
- the device address used does not coincide with that of the slave address
- the checksum (CRC16) is not correct
- the instruction from the Master is incomplete or over-defined
- The number of words to be read is zero.

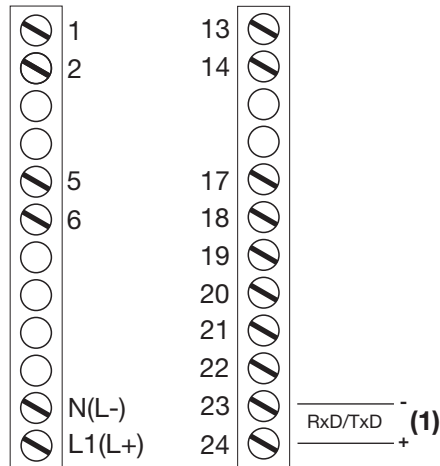
In these cases the data request should be transmitted again once the timeout time (2 s) has elapsed.

3.1 Connection diagram

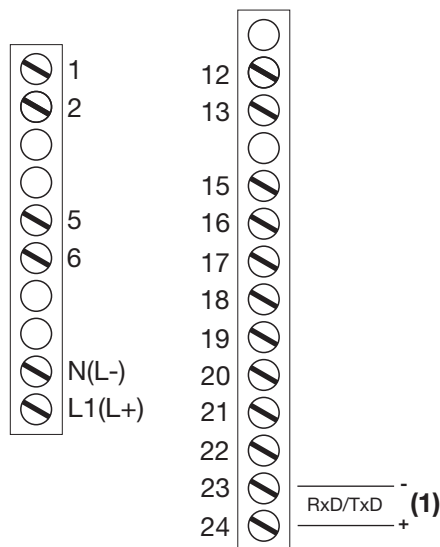


The devices of this controller series can be ordered with an RS485 interface as an option. Please refer to the type sheet 70.2070 or the operating manual B 70.2070.0 (type explanation) for the order specifications.

**Connection diagram
Type 702071**



**Connection diagram
Type 702072
and
Type 702074**



(1) RS485 interface



Connect the interface line shielding to earth on one side in the switch cabinet.

3 RS485 interface

3.2 Configuration

The following table shows the possible Modbus interface settings to be carried out in the configuration level ($\text{CONF} \rightarrow \text{INTF}$) and/or in the setup program.



For more detailed information about configuration, please refer to operating manual B 70.2070.0.

	Symbol	Value/ Selection	Description
Baud rate	<i>bdr</i>	0	9600 bps
		1	19200 bps
		2	38400 bps
Data format	<i>dft</i>	0	8 data bits, 1 stop bit, no parity
		1	8 data bits, 1 stop bit, odd parity
		2	8 data bits, 1 stop bit, even parity
		3	8 data bits, 2 stop bits, no parity
Device address	<i>Adr</i>	0...1...255	Address in data network Addresses 0 and 255 are reserved for specific purposes and should not be used here.
Minimum response time	(Setup)	0 ...500ms	Minimum time period that elapses between the request of a device in the data network and the response of the controller (can only be adjusted via the setup program).

Factory settings are shown **bold**.



When the communication takes place via the setup interface, the RS485 interface is inactive.

4 Modbus addresses

Data type, type of access The following tables contain specifications of all process and device data including their addresses, data type and type of access.

Meaning:

R/O	Read only access
W/O	Write only access
R/W	Read/write access
INT	Integer (8 or 16 bit)
Bit x	Bit No. x (bit 0 is always the bit with the lowest value)
LONG	Long integer (4 byte)
FLOAT	Float value (4 bytes) as per IEEE 754
TEXT4	Text 4 characters



Write operations to R/W parameters result in them being saved to the EEPROM. These memory modules only have a limited number of write cycles (approx. 10000). For this reason, this function can be switched off in the case of frequent programming. The parameter values are then only saved in the volatile memory (RAM) and will be lost after a supply failure.

⇒ *Setup program (undocumented parameters -> Bit parameters -> Set parameter 2)*

4.1 Process data

Address	Data type/ bit number	Access	Signal designation
0x001F	INT	R/O	Internal binary values
	Bit 12		Binary value L1 (= 0x1000)
	Bit 13		Binary value L2 (= 0x2000)
0x0020	INT	R/O	Controller status
	Bit 12		Manual mode active (=0x1000)
	Bit 15		Self-optimization active (=0x8000)
0x0021	INT	R/O	Binary outputs 1...4 (Switching states 0 = off / 1 = on)
	Bit 0		Output K1: Relay (= 0x0001)
	Bit 1		Output K2: Relay (= 0x0002)
	Bit 2		Output K3: Logic (= 0x0004)
	Bit 3		Output K4: Relay (= 0x0008)

4 Modbus addresses

Address	Data type/ bit number	Access	Signal designation
0x0023	INT	R/O	Binary inputs 1 and 2 (Switching states 0 = open / 1 = closed)
	Bit 0		Input 1 (= 0x0001)
	Bit 1		Input 2 (= 0x0002)
0x0024	INT	R/O	Limit comparators 1...2
	Bit 0		Limit comparator 1 (= 0x0001)
	Bit 1		Limit comparator 2 (= 0x0002)
0x0025	INT	R/W	Control of the binary outputs (individual)
	Bit 0 + Bit 8		Output K1 (= 0x0101)
	Bit 1 + Bit 9		Output K2 (= 0x0202)
	Bit 2 + Bit 10		Output K3 (= 0x0404)
	Bit 3 + Bit 11		Output K4 (= 0x0808)
0x0026	FLOAT	R/O	Analog input [mV]
0x0028	FLOAT	R/O	Internal Pt100 [Ohm]
0x002A	INT	R/O	Sampling cycle time
0x002B	FLOAT	R/O	Analog input [displayed value]
0x002D	FLOAT	R/O	Internal analog value 1
0x002F	FLOAT	R/O	Internal analog value 2
0x0031	FLOAT	R/O	Controller, ramp limit value
0x0033	FLOAT	R/O	Controller, actual value, FILTERED
0x0035	FLOAT	R/O	Controller, actual value, UNFILTERED
0x0037	FLOAT	R/W	Controller set point value
0x0039	FLOAT	R/O	Controller, output value display
0x003B	FLOAT	R/O	Controller, output value, HEATING
0x003D	FLOAT	R/O	Controller, output value, COOLING
0x003F	FLOAT	R/O	Controller, control difference
0x0041	FLOAT	R/O	Controller, control deviation
0x0043	INT	R/O	Controller, switching state, HEATING
0x0044	INT	R/O	Controller, switching state, COOLING
0x0046	INT	R/O	Output value, manual mode
0x0047	LONG	R/O	Timer run time
0x0049	LONG	R/O	Residual timer time
0x004B	INT	R/O	Timer status
	Bit 1		Timer stopped (= 0x0002)
	Bit 5		Timer runs (= 0x0020)
	Bit 6		Timer end (= 0x0040)
	Bit 15		Timer signal (= 0x8000)

4 Modbus addresses

4.2 Set point values

Address	Data type/ bit number	Access	Signal designation
0x3100	FLOAT	R/W	Set point value W1
0x3102	FLOAT	R/W	Set point value W2

4.3 Controller parameters

Address	Data type/ bit number	Access	Signal designation
0x3000	FLOAT	R/W	Controller parameter XP1
0x3002	FLOAT	R/W	Controller parameter XP2
0x3004	FLOAT	R/W	Controller parameter TV
0x3006	FLOAT	R/W	Controller parameter TN
0x300C	FLOAT	R/W	Controller parameter CY1
0x300E	FLOAT	R/W	Controller parameter CY2
0x3010	FLOAT	R/W	Controller parameter XSH
0x3012	FLOAT	R/W	Controller parameter XD1
0x3014	FLOAT	R/W	Controller parameter XD2
0x3016	INT	R/W	Controller parameter TT
0x3017	INT	R/W	Controller parameter Y0
0x3018	INT	R/W	Controller parameter Y1
0x3019	INT	R/W	Controller parameter Y2

4.4 Configuration

Address	Data type/ bit number	Access	Signal designation
0x0053	FLOAT	R/W	Ramp function, ramp rate
0x0055	FLOAT	R/W	Filter time constant (digital filter)
0x0057	FLOAT	R/W	Limit comparator 1 Alarm value AL
0x0059	FLOAT	R/W	Limit comparator 1 Hysteresis
0x005D	FLOAT	R/W	Limit comparator 2 Alarm value AL
0x005F	FLOAT	R/W	Limit comparator 2 Hysteresis
0x0063	LONG	R/W	Timer value
0x0065	LONG	R/W	Service limit value
0x0067	TEXT4	R/W	Alarm text
0x0069	LONG	R/W	Service counter

4 Modbus addresses

4.5 Commands

Address	Data type/ bit number	Access	Signal designation
0x004D	INT	W/O	Binary functions CONTROLLER
	Bit 0		Self-optimization start (=0x0001)
	Bit 1		Self-optimization abort (=0x0002)
	Bit 2		Manual operation (= 0x0004)
	Bit 3		Automatic operation (= 0x0008)
	Bit 4		Controller off (= 0x0010)
	Bit 5		Manual mode inhibit (= 0x0020)
	Bit 6		Ramp stop (= 0x0040)
	Bit 7		Ramp abort (= 0x0080)
	Bit 8		Ramp restart (= 0x0100)
	Bit 9		Timer start (= 0x0200)
	Bit 10		Timer abort (= 0x0400)
	Bit 11		Timer stop (= 0x0800)
0x004E	INT	W/O	Binary functions OPERATION
	Bit 0		Keyboard inhibit (= 0x0001)
	Bit 1		Configuration and parameter level inhibit (= 0x0002)
	Bit 3		Display OFF (= 0x0008)
	Bit 5		Text display (= 0x0020)
0x004F	INT	W/O	Binary functions TIMER
	Bit 9		Timer start (= 0x0200)
	Bit 10		Timer abort (= 0x0400)
	Bit 11		Timer stop (= 0x0800)
0x0050	INT	R/W	Set point value toggling
	Bit 0		Set point value 1 (= 0x0001)
	Bit 1		Set point value 2 (= 0x0002)

4.6 RAM memory

Address	Data type/ bit number	Access	Signal designation
0x3200	FLOAT	W/O	Controller set point value (writable)
0x3202	FLOAT	W/O	Controller actual value (writable)
0x3204	FLOAT	W/O	Internal analog value 1 (writable)
0x3206	FLOAT	W/O	Internal analog value 2 (writable)
0x3208	INT	W/O	Internal binary values (writable)
	Bit 0 + Bit 7		Binary value L1 (= 0x0081)
	Bit 1 + Bit 7		Binary value L2 (= 0x0082)



Modbus allows direct access to the RAM memory of the controller for writing the controller set point value (0x3200), the controller actual value (0x3202) and the internal analog values (0x3204, 0x3206) as well as the internal binary values (0x3208).

When writing, a range between -1999 and +9999 is available for the controller set point values, controller actual values and the internal analog values. In this case, the data written in the controller is used instead of the original value.

If you wish to use the original value again on the controller, write the value 200001 at the memory location concerned via Modbus.

4 Modbus addresses



JUMO GmbH & Co. KG

Street address:
Moritz-Juchheim-Straße 1
36039 Fulda, Germany
Delivery address:
Mackenrodtstraße 14
36039 Fulda, Germany
Postal address:
36035 Fulda, Germany
Phone: +49 661 6003-0
Fax: +49 661 6003-607
e-mail: mail@jumo.net
Internet: www.jumo.net

JUMO Instrument Co. Ltd.

JUMO House
Temple Bank, Riverway
Harlow, Essex CM 20 2 TT, UK
Phone: +44 1279 635533
Fax: +44 1279 635262
e-mail: sales@jumo.co.uk
Internet: www.jumo.co.uk

JUMO Process Control, Inc.

8 Technology Boulevard
Canastota, NY 13032, USA
Phone: 315-697-JUMO
1-800-554-JUMO
Fax: 315-697-5867
e-mail: info@jumo.us
Internet: www.jumo.us