

# **Communication protocols description of Hx4xx regulators with RS485 and RS232 communication interface**

**Copyright:** COMET System, Ltd. It is prohibited to copy and edit this manual and make any changes at all without explicit agreement of COMET System, Ltd. All rights reserved.

COMET System, Ltd makes constant development and improvement of all its products. That is why it reserves the right to make any technical changes on the device/product without previous notice.

## Content

<b>1. MODBUS RTU</b> .....	<b>3</b>
1.1. SUPPORTED FUNCTIONS .....	3
1.2. INTERNAL JUMPER AND BUTTON „SET“ .....	3
1.3. DESCRIPTION OF SUPPORTED FUNCTION .....	4
1.3.1. 03 (0x03): Reading of 16-bit registers (Read Holding Registers) .....	4
1.3.2. 04 (0x04): Reading 16-bit input gates (Read Input Registers) .....	4
1.3.3. 16 (0x10): Setting of several 16-bit registers (Write Multiple Registers) .....	5
1.3.4. Exception Responses .....	6
1.3.5. Exception codes .....	6
1.4. MODBUS CRC .....	6
1.4.1. Procedure during calculation of Modbus CRC .....	6
1.5. MODBUS REGISTERS OF THE DEVICE .....	7
1.6. EXAMPLE OF COMMUNICATION .....	9
1.6.1. Reading of temperature, address 0x0031 .....	9
1.6.2. Reading of relative humidity, address 0x0032 .....	9
1.6.3. Reading of computed value, address 0x0033 .....	10
1.6.4. Reading of all values at once, address block 0x0031 to 0x0033 .....	10
<b>2. PROTOCOL COMPATIBLE WITH ADVANTECH-ADAM STANDARD</b> .....	<b>11</b>
2.1. INTERNAL JUMPER .....	11
2.2. GENERAL SYNTAX OF COMMANDS .....	11
2.3. DESCRIPTION OF SUPPORTED FUNCTIONS .....	12
2.3.1. Configuration of device .....	12
2.3.1.1. Switching to Modbus communication protocol .....	12
2.3.2. Response of the device .....	13
2.3.3. Check sum (CRC) .....	13
2.3.4. Error states .....	13
2.4. SUPPORTED COMMANDS .....	13
2.4.1. Query to adjusted configuration .....	13
2.4.2. Reading of device name .....	13
2.4.3. Reading of firmware version .....	13
2.4.4. Reading of the temperature* .....	14
2.4.5. Reading of humidity * .....	14
2.4.6. Reading of computed value * .....	14
2.4.7. Reading of atmospheric pressure * .....	14
2.4.8. Status word reading .....	14
2.4.9. Relay 1 status (Alarm 1) [0/1] .....	15
2.4.10. Relay 2 status (Alarm 2) [0/1] .....	15
2.4.11. Binary input 1 status [0/1] .....	15
2.4.12. Binary input 2 status [0/1] .....	15
2.4.13. Binary input 3 status [0/1] .....	15
2.5. DATA FORMAT .....	15
2.6. EXAMPLES OF COMMUNICATION .....	16
<b>3. TECHNICAL SUPPORT AND SERVICE</b> .....	<b>17</b>
<b>4. APPENDIX A</b> .....	<b>18</b>

# Description of Hx4xx communication protocols

Devices are preset from manufacturer to Modbus RTU communication protocol<sup>1</sup>, address 01, communication speed 9600Bd, no parity, 2 stop bits. If you would like to use different communication protocol than Modbus RTU, it is necessary preset them – see device Instruction manual or use user's software *Tsensor* for setting of all device parameters (recommended). It is free to download at [www.cometsystem.cz](http://www.cometsystem.cz). It supports make the adjustment of the device too. This procedure is described at file „*Calibration manual.pdf*“ which is installed commonly with the software. Devices with RS 232 communication interface have always communication address set to 1, it is not possible to change it.

You can use discussion forum at web address: <http://www.forum.cometsystem.cz/>, short description is at <http://www.cometsystem.cz/english/forum.htm>

## 1. Modbus RTU

Control units communicate on master-slave principle in half-duplex operation. Only master can send request and only addressed device responds. During sending of request no other slave station should respond. During communication, data transfer proceeds in binary format. Each Byte is sent as eight bit data word in format: 1 start bit, data word 8 bit (LSB first), 2 stop bits<sup>2</sup>, without parity. Device supports communication speed from 110Bd to 115200Bd. Available address range is from 0 to 255, address 0 is reserved for broadcast and device doesn't send reply for it.

For more detailed communication protocol description see [www.modbus.org](http://www.modbus.org) .

### 1.1. Supported functions

**03 (0x03):** Reading of 16-bit registers (Read Holding Registers)

**04 (0x04):** Reading of 16-bit input gates (Read Input Registers)

**16 (0x10):** Setting of more 16-bit registers (Write Multiple Registers)\*<sup>3</sup>

### 1.2. Internal jumper and button „SET“

Internal jumper placement is described into device Instruction manual. If communication protocol **Modbus is selected** the function of jumper and button is as follows:

- Jumper opened – device memory is protected from writing, from device side it is only enabled to read measured value, writing to memory is disabled (no change of device address, communication speed and LCD setting is enabled)
- Jumper closed – writing to device memory is enabled by means of User's software or function *16 (0x10): Setting of several 16-bit registers (Write Multiple Registers)*.

It is possible to write into device memory when internal jumper is opened, it is necessary press and keep pressed „**SET**“ key BEFORE write command is applied (i.e. before “Save

---

<sup>1</sup> If in order was not specified differently.

<sup>2</sup> Device sends two stop bits, for receive one stop bit is enough.

<sup>3</sup> See detailed description of this function.

changes” into TSensor SW is pressed). When write command is finished, then release „**SET**“ key

### 1.3. Description of supported function

#### 1.3.1. 03 (0x03): Reading of 16-bit registers (Read Holding Registers)

Function serves for reading of values from device. Addresses of available registers are listed in chapter „Modbus registers of the device” at page 7.

Request:

FUNCTION	Function code	0x03
DATA	Initial address Hi	0x??
	Initial address Lo	0x??
	Number of registers Hi	0x??
	Number of registers Lo	0x??

Response:

FUNCTION	Function code	0x03
DATA	Number of Bytes	0x??
	States of register Hi	0x??
	States of register Lo	0x??
	...	
	States of register Hi	0x??
	States of register Lo	0x??

Exceptional response:

FUNCTION	Function code	0x83
DATA	Exception code	0x??

During sending of query to device initial register address and number of registers to read are sent. Register addresses are indexed from zero – **register 0x31 is physically sent as value 0x30, 0x32 as 0x31...** (zero based addressing)

#### 1.3.2. 04 (0x04): Reading 16-bit input gates (Read Input Registers)

This function is also possible to use for reading values from device, syntax is the same as with function 03 (0x03): Reading of 16-bit registers. Addresses of available registers are specified in chapter „Modbus registers of the device” at page 7.

### 1.3.3. 16 (0x10): Setting of several 16-bit registers (Write Multiple Registers)

Setting of device address and communication speed is possible to perform by writing to device registers (rest of parameters is possible to set only with User's software).

Attention! **During writing to device registers it is not enabled to write any number of registers.** Always below procedure should be strictly followed. If procedure is not followed undoable loss of important settings stored in device can occur! **It is strongly recommended to use User's software to set all device parameters. It is free to download at [www.cometsystem.cz](http://www.cometsystem.cz).**

Request:

FUNCTION	Function code	0x10
DATA	Initial address Hi	0x??
	Initial address Lo	0x??
	Number of registers Hi	0x??
	Number of registers Lo	0x??
	Number of Bytes (of sent data)	0x??

Response:

FUNCTION	Function code	0x10
DATA	Initial address Hi	0x??
	Initial address Lo	0x??
	Number of registers Hi	0x??
	Number of registers Lo	0x??

Exceptional response:

FUNCTION	Function code	0x90
DATA	Exception code	0x??

For detailed description see „Appendix A“ at page 18.

### 1.3.4. Exception Responses

After sending query to the device, master device waits for normal response. After master device query one of the following events can occur:

1. If device receives a query without communication error and query is possible to process, master device receives response.
2. If device does not receive all queries due to communication error, no response is sent. Main program is able to process condition of exceeding of time for query.
3. If device receives a query, but detects communication error (CRC), no response is sent. Main program is able to process condition of exceeding of time for query.

If device receives a query without communication error, but cannot process it, master device receives exception response, which informs master device on error nature.

#### Exception Response

- has two fields to distinguish it from normal response:

1. Function code field
2. Data field.

#### ad1 Function code field

In normal response of slave device function code of original query corresponds to function code of response. All function codes have most significant bit (MSB) equal to 0. In exception response slave device sets most significant bit of function code to 1. Main station recognizes exception response by means of this bit and can check data field for exception code.

#### ad2 Data field

Device returns exception code in exception response in data field. Event causing exception is determined this way.

### 1.3.5. Exception codes

**0x01** Invalid function. Function code in query is not allowed action for device.

**0x02** Invalid data address. Data address received in query is not allowed address for device.

## 1.4. Modbus CRC

Check sums of entire Modbus messages are mostly automatically inserted to the end of request by communication programs themselves. In case there is a need to insert to generate Modbus CRC to program itself, the way of calculation is as follows:

### 1.4.1. Procedure during calculation of Modbus CRC

1. To fill 16-bit register with value 0xFFFF (all bits set to 1). Let us call this register „CRC register“.
2. Perform logic function Exclusive OR with first eight bit message Byte with lower eight bits of CRC register. Store result to CRC register.
3. Shift content of CRC register of one bit to the right (towards to LSB), enter 0 as upper bit of CRC. Memorize values of lowest shifted bit (LSB).
4. If LSB was 0, then repeat step 3 (other shift).  
If LSB was 1, then perform Exclusive OR CRC register with value 0xA001.
5. Repeat steps 3 and 4 as long as eight shifts proceed. After eight shifts eight bit Byte is processed.

6. Repeat steps 2 to 5 to the next of eight bit Byte of message as long as all Bytes are proceeded.
7. In the end after processing of all message Bytes check sum value is stored in CRC register.
8. During connection of check sum to the message lower Byte of CRC register is sent as first, then upper Byte of CRC register.

## 1.5. Modbus registers of the device

Variable	Unit	Address [hex] <sup>x</sup>	Address [dec] <sup>x</sup>	Format	Size	Status
measured temperature	[°C/°F]*	0x0031	49	Int*10	BIN16	R
measured relative humidity	[%]	0x0032	50	Int*10	BIN16	R
computed value*		0x0033	51	Int*10	BIN16	R
relay 1 status [0/1] (Alarm 1)	[-]	0x003B	59	Int	BIN16	R
relay 2 status [0/1](Alarm 2)	[-]	0x003C	60	Int	BIN16	R
binary input 1 status [0/1]	[-]	0x003D	61	Int	BIN16	R
binary input 2 status [0/1]	[-]	0x003E	62	Int	BIN16	R
binary input 3 status [0/1]	[-]	0x003F	63	Int	BIN16	R
status of all binary inputs (bit0, 1, 2)	[-]	0x0008	8	Int	BIN16	R
status word (described below)	[-]	0x0007	7	Int	BIN16	R
device serial number Hi	[-]	0x1035	4149	BCD	BIN16	R
device serial number Lo	[-]	0x1036	4150	BCD	BIN16	R
firmware version Hi	[-]	0x3001	12289	BCD	BIN16	R
firmware version Lo	[-]	0x3002	12290	BCD	BIN16	R
device address	[-]	0x2001	8193	Int	BIN16	R/W**
communication speed code	[-]	0x2002	8194	Int	BIN16	R/W**

### Addition for H7430 and H7431 regulators with barometric pressure measurement:

Variable	Unit	Address [hex] <sup>x</sup>	Address [dec] <sup>x</sup>	Format	Size	Status
barometric pressure	hPa	0x0034	52	Int*10	BIN16	R
	PSI			Int*1000		
	inHg			Int*100		
	mBar			Int*10		
	oz/in <sup>2</sup>			Int*10		
	mmHg			Int*10		
	inH <sub>2</sub> O			Int*10		
	kPa			Int*100		

#### Legend:

- \* depends on device setting (by User's software)
- Int\*10 register is in format integer \*10 (likewise \*100, \*1000)
- R register is designed only for reading

- W\*\* register is designed for writing, for details see description of communication protocols
- x at transmit are register addresses indexed from zero - „zero based addressing“. For example „measured temperature“ with Modbus address 0x31 is physically sent along data bus as value 0x30. You make sure of correct addressing with the aid of Master device documentation or experimentally (e.g. for „measured temperature“ try to use address 0x31 or 0x30).
- Status word: 16b value return, bite description:
 

Bit0	0/1	jumper open/closed
Bit1	-	unused
Bit2	0	always 0
Bit3	0/1	relay 1 open/closed
Bit4	0/1	relay 2 open/closed
Bit5	0/1	internal acoustic alarm status
Bit6	0/1	binary input 1 status
Bit7	0/1	binary input 1 status
Bit8	0/1	binary input 1 status
Bit9 to 15		unused

**Note:** In case there is a need for reading of measured values from the device with higher resolution than one decimal, measured values in device are stored also in „Float“ format, which is not directly compatible with IEEE754.

## 1.6. Example of communication

In all examples communication with device at address 01 is supposed

### 1.6.1. Reading of temperature, address 0x0031

Modbus command:

device address:	01
reading 16-bit registers	03
initial address Hi	00
initial address Lo	31
number read registers Hi	00
number read registers Lo	01

Via link is physically sent: 01 03 00 30 00 01 84 05

Received response from device: 01 03 02 00 F4 B9 C3

device address:	01
reading 16-bit registers	03
Number Byte	02
State of register Hi	00
State of register Lo	F4 (0x00F4 i.e. 244 i.e. 24.4 °C)
Modbus CRC Lo	B9
Modbus CRC Hi	C3

### 1.6.2. Reading of relative humidity, address 0x0032

Modbus command:

device address:	01
reading 16-bit registers	03
initial address Hi	00
initial address Lo	32
number read registers Hi	00
number read registers Lo	01

Via link is physically sent: 01 03 00 31 00 01 D5 C5

Received response from device: 01 03 02 01 6C B9 F9

device address:	01
reading 16-bit registers	03
Number Byte	02
State of register Hi	01
State of register Lo	6C (0x016C i.e. 364 i.e. 36.4 %RH)
Modbus CRC Lo	B9
Modbus CRC Hi	F9

### 1.6.3. Reading of computed value, address 0x0033

Modbus command:

device address:	01
reading 16-bit registers	03
initial address Hi	00
initial address Lo	33
number read registers Hi	00
number read registers Lo	01

Via link is physically sent: 01 03 00 32 00 01 25 C5

Received response from device: 01 03 02 FF 3E 78 64

device address:	01
reading 16-bit registers	03
Number Byte	02
State of register Hi	FF
State of register Lo	3E (0xFF3E i.e. -194 i.e. -19.4)
Modbus CRC Lo	78
Modbus CRC Hi	64

### 1.6.4. Reading of all values at once, address block 0x0031 to 0x0033

Modbus command:

device address:	01
reading 16-bit registers	03
initial address Hi	00
initial address Lo	31
number read registers Hi	00
number read registers Lo	03

Via link is physically sent: 01 03 00 30 00 03 05 C4

Received response from device: 01 03 06 FF C4 01 14 FF 38 C5 71

device address:	01
reading 16-bit registers	03
Number Byte	06
State of register Hi	FF
State of register Lo	C4 (0xFFC4 i.e. -60 i.e. -6.0 °C)
State of register Hi	01
State of register Lo	14 (0x0114 i.e. 276 i.e. 27.6 %RH)
State of register Hi	FF
State of register Lo	38 (0xFF38 i.e. -200 i.e. -20.0 °C)*
Modbus CRC Lo	C5
Modbus CRC Hi	71

\* Computed value is preset by factory to Dew Point Temperature, to change use User's software.

## 2. Protocol compatible with Advantech-ADAM standard

Control units communicate on master-slave principle in half-duplex operation. Only master can send requests and only addressed device responds. During sending request any of slave devices should respond. During communication data is transferred in ASCII format (in characters). Each Byte is sent as two ASCII characters (value 0x2F is sent as pair of characters 0x32, 0x46, i.e. characters „2“ and „F“). **All commands and values MUST be entered in CAPITAL LETTERS!** Device supports communication speed from 1200Bd to 115200Bd, parameters of communication link are 1 start bit + eight bit data word (LSB first) + 1 stop bit, without parity. Commands for reading of measured values are different for devices with temperature measurement only and for combined devices with temperature and relative humidity measurement – see later.

### 2.1. Internal Jumper

Its location is described into Instruction manual. If communication protocol compatible with standard Advantech-ADAM is selected, its function is the following:

- If jumper during switching ON the power is closed, device always communicates with following parameters regardless stored setting in the device: communication speed 9600 Bd, without check sum, device address 00h
- If jumper during switching ON the power is not closed, device communicates in accordance with stored setting.
- If jumper is closed during device operation, device temporarily changes its address to 00h, it will communicate in the same communication speed as before closing jumper and will communicate without check sum. After jumper is opened setting of address and check sum is reset in accordance with values stored in device.
- Communication speed and check sum are possible to change only if jumper is closed (see chapter Configuration of device at page 12).

### 2.2. General syntax of commands

**[distinguishing character][device address][command][data][check sum][CR]**

Valid **distinguishing characters** towards to device are: \$, #, %

**Device address** contents 2 ASCII bytes in hexadecimal code (upper case letters) representing one byte of binary address (e.g. „3“ „F“ corresponds to address 3Fh, i.e. 63, is sent as 0x33, 0x46)

**Check sum:** enabled to switch ON/switch OFF

**CR ...** 1 byte (0Dh)

## 2.3. Description of supported functions

### 2.3.1. Configuration of device

Syntax of command: `%AANNTTCCFF cr`

Meaning of symbols:

**AA** ... current address of device 00...FF (hexadecimal)

**NN** ... new address of device 00...FF (hexadecimal)

**TT** ... code of device: 2Ch ,it means regulator Hx4xx

**CC** ... code of communication speed

Code	Speed [Bd]
03	1200
04	2400
05	4800
06	9600
07	19200
08	38400
09	57600
0A	115200

**FF** ... data format and check sum:

D7	D6	D5	D4	D3	D2	D1	D0
----	----	----	----	----	----	----	----

xxxx xx00 format "Engineering units"

x0xx xx00 check sum switched OFF

x1xx xx00 check sum switched ON

- Communication speed and check sum are possible to change only if jumper is closed
  - Change in communication speed activates only after device power is switched OFF and switched ON again.
  - Change in setting of check sum activates immediately after jumper is opened
- If address is changing
  - and jumper is closed, device responds with address 00h again, and newly set address will be activated after jumper is opened.
  - and jumper is not closed, change is activated immediately.
- If attempt occurs to write incorrect data to the device (and syntax is correct), device responds with error message.

#### 2.3.1.1. Switching to Modbus communication protocol

Communication protocol compatible with Advantech-ADAM don't provide setting of all device parameters. That is why there is implemented command for switching to Modbus communication protocol, which it supports. This command makes **permanent change to Modbus RTU communication protocol**, device address 01, communication speed 9600Bd, no parity, 2 stop bits. When this command is processed, then device is restarted automatically and Modbus RTU communication protocol is used.

Syntax of command: **%AAMODBUS(crc)cr** ... switch to MODBUS RTU protocol  
Response: **!AAMODBUS(crc)cr**

Note: There is possible to change communication protocol through device keys, see Instruction manual, chapter "Extended setting mode".

### 2.3.2. Response of the device

1. If syntax of command is not correct, device does not respond at all (e.g. no check sum is received though it is switched ON, check sum is not correct, string is not complete or contents invalid character).
2. If syntax is correct, but required operation is not correct, device returns error message in format  
**? AA cr**  
this state appears if we try to change communication speed and check sum and jumper is not closed.
3. If command is executed, device responds:  
**! AA cr**

### 2.3.3. Check sum (CRC)

it is the sum of all characters before it, its lowest byte is applied (used).

### 2.3.4. Error states

**>-0000 cr** lower limit of temperature, error in measurement of humidity, computed value or atmospheric pressure.  
**>+9999 cr** upper limit of temperature, error in measurement of humidity and computed value. Is NOT used for atmospheric pressure error alarm.

## 2.4. Supported commands

### 2.4.1. Query to adjusted configuration

Syntax of command: **\$AA2 cr**  
Response: **!AATTCFF cr** symbols correspond with paragraph "Configuration of device" at page "12"

### 2.4.2. Reading of device name

Syntax of command: **\$AAM cr**  
Response: **!AAHxxxx(crc)cr** (**!AAH3430 cr** accordingly with device model)

### 2.4.3. Reading of firmware version

Syntax of command: **\$AAF cr**  
Response: **!AA(version) cr** reads version number of device firmware

#### 2.4.4. Reading of the temperature\*

Syntax of command: #AA0(crc)cr

Response format: >±xxx.x0

Response: > (temperature) (crc)cr (i.e. >-012.30(crc)cr)

\* If the value is not supported, still returns ? AA cr

#### 2.4.5. Reading of humidity \*

Syntax of command: #AA1(crc)cr

Response format: >+xxx.x0

Response: > (humidity) (crc)cr (i.e. >+044.30(crc)cr)

\* If the value is not supported, still returns ? AA cr

#### 2.4.6. Reading of computed value \*

Syntax of command: #AA2(crc)cr

Response format: >±xxx.x0

Response: > (computed value) (crc)cr (i.e. >+004.30(crc)cr)

\* If the value is not supported, still returns ? AA cr

#### 2.4.7. Reading of atmospheric pressure \*

Syntax of command: #AA3(crc)cr

Response: > (atmospheric pressure) (crc)cr (i.e. >+1013.2(crc)cr)

**Attention! Atmospheric pressure is stored in next format (depends on selected unit):**

Unit**	Format
hPa	>+1013.1
PSI	>+14.123
inHg	>+028.12
mBar	>+1013.1
oz/in <sup>2</sup>	>+0225.1
mmHg	>+0728.1
inH <sub>2</sub> O	>+0380.1
kPa	>+101.12

\* If the value is not supported, still returns ? AA cr

\*\* Depends on device setting (User's software)

#### 2.4.8. Status word reading

Syntax of command: # AA4(crc)cr

Response format: >+0xxxxx

Response: > (value)(crc)cr (i.e. >+000472(crc)cr)

Bit assignment:

Bit0 0/1 Jumper opened/closed

Bit1 not used

Bit2	always 0	
Bit3	0/1	relay 1 opened/closed
Bit4	0/1	relay 2 opened/closed
Bit5	0/1	actual internal acoustic status deactivated/activated
Bit6	0/1	binary input 1 status
Bit7	0/1	binary input 2 status
Bit8	0/1	binary input 3 status
Bit9 to 15		not used

#### 2.4.9. Relay 1 status (Alarm 1) [0/1]

Syntax of command: **#AA5(crc)cr**

Response: > **+000001(crc)cr** i.e. relay 1 closed, Alarm 1 activated  
 > **+000000(crc)cr** i.e. relay 1 opened, Alarm 1 deactivated

#### 2.4.10. Relay 2 status (Alarm 2) [0/1]

Syntax of command: **#AA6(crc)cr**

Response: > **+000001(crc)cr** i.e. relay 2 closed, Alarm 1 activated  
 > **+000000(crc)cr** i.e. relay 2 opened, Alarm 1 deactivated

#### 2.4.11. Binary input 1 status [0/1]

Syntax of command: **#AA7(crc)cr**

Response: > **+000000(crc)cr** i.e. binary input closed / low input level  
 > **+000001(crc)cr** i.e. binary input opened / hi input level

#### 2.4.12. Binary input 2 status [0/1]

Syntax of command: **#AA8(crc)cr**

Response: > **+000000(crc)cr** i.e. binary input closed / low input level  
 > **+000001(crc)cr** i.e. binary input opened / hi input level

#### 2.4.13. Binary input 3 status [0/1]

Syntax of command: **#AA9(crc)cr**

Response: > **+000000(crc)cr** i.e. binary input closed / low input level  
 > **+000001(crc)cr** i.e. binary input opened / hi input level

### 2.5. Data format

Device uses data format „Engineering units“, i.e. fixed decimal point. Temperature, humidity and computed value are displayed with 2 digits behind decimal point, second digit behind decimal point is always zero. Atmospheric pressure display depends on selected pressure unit, see command description for „Reading of atmospheric pressure \* “ at page 14. The responses for commands from #AA4 to #AA9 are formatted „> +0xxxxx(crc)cr“. Examples – see description of each commands.

## 2.6. Examples of communication

**Example 1:** Change of device address during operation (without closed jumper, CRC switched OFF)

Device, which had address 23h is configured to address 24h, speed 9600 Bd, without CRC, setting of communication speed and CRC must not change (setting of communication speed and CRC is not possible to change without closed jumper).

Command: **%23242C0600 cr**  
Response: **!24 cr**

---

**Example 2:** Reading of temperature without closed jumper, device address 01:

- without check sum

Command: **#010 cr**  
Response: **> +020.50 cr**

- with check sum:

Command: **#010 B4 cr**  
where it is sent: 23 30 31 30 **42 34 0D**  
calculation of CRC:  $23h+30h+31h+30h = B4h$ , then CRC = **B4h**, it is sent as **0x42, 0x34**

Response: **>+020.50 8E cr**  
where it is sent: 3E 2B 30 32 30 2E 35 30 **38 45 0D**  
calculation of CRC:  $3Eh+2Bh+30h+32h+30h+2Eh+35h+30h=18Eh$  then CRC = **8Eh**, then it is sent as **0x38, 0x45**

---

**Example 3:** Setting of device to address 9F, communication speed remains 9600 Bd, check sums are switched ON (jumper should be closed because setting of CRC will be changed):

- with jumper, device always reports from address „00“ without CRC

Command: **%009F2C0640 cr**  
Response: **!00**  
After jumper is opened device address changes to 9Fh

---

**Example 4:** Status word reading, i.e. actual status of internal Jumper, output relays, acoustic signalization and binary inputs. Device address is 01:

- without check sum

Command: **#014 cr**  
Response: **>+000472cr**

- with check sum:

Command: **#014 B8 cr**  
where it is sent: 23 30 31 34 **42 38 0D**

calculation of CRC:  $23h+30h+31h+34h = B8h$ , then CRC = **B8h**, then it is sent as **0x42, 0x38**

Response: **>+000472 96 cr**

where it is sent: 3E 2B 30 30 30 34 37 32 **39 36** 0D

calculation of CRC:  $3Eh+2Bh+30h+30h+30h+34h+37h+32h=196h$  then CRC = **96h**, then it is sent as **0x39, 0x36**

Bit assignment:

Decimal value 472 represents at binary format (1 1101 1000)B, where:

Bit0	0	Jumper is opened
Bit1	0	not used
Bit2	0	always 0
Bit3	1	relay 1 closed (Alarm 1)
Bit4	1	relay 2 closed (Alarm 2)
Bit5	0	internal acoustic is Off
Bit6	1	binary input 1 is opened
Bit7	1	binary input 2 is opened
Bit8	1	binary input 3 is opened
Bit9 to 15		not used

---

**Example 5:** Relay 1 output status reading (Alarm 1), device address 01:

- without check sum

Command: **#015 cr**

Response: **>+000001cr**

- with check sum:

Command: **#015 B9 cr**

where it is sent: 23 30 31 34 **42 39** 0D

calculation of CRC:  $23h+30h+31h+35h = B9h$ , then CRC = **B9h**, then it is sent as **0x42, 0x39**

Response: **>+ >+000001 8A cr**

where it is sent: 3E 2B 30 30 30 30 31 **38 41** 0D

calculation of CRC:  $3Eh+2Bh+30h+30h+30h+30h+30h+31h=18Ah$  then CRC = **8Ah**, then it is sent as **0x38, 0x41**

It means relay 1 is closed, alarm 1 is active.

### 3. Technical support and service

Technical support and service is provided by distributor. For contact see warranty certificate.

You can use discussion forum at web address: <http://www.forum.cometsystem.cz/>

Short description is available at <http://www.cometsystem.cz/english/forum.htm>

## 4. Appendix A

### Writing into device's registers with Modbus RTU protocol using

It is strongly recommended to use user's software *Tsensor* for setting of all device parameters. It is free to download at [www.cometsystem.cz](http://www.cometsystem.cz). If you need write your own writing procedure, read next steps carefully.

Attention! **During writing to device registers it is not enabled to write any number of registers.** Always below procedure should be strictly followed. If procedure is not followed undoable loss of important settings stored in device can occur!

- device address is stored at Modbus address 0x2001 as binary number
- code of communication speed is stored at Modbus address 0x2002

Communication speed [Bd]	Code of communication speed [hex]
110	94F2
300	369D
600	1B4F
1200	0DA7
2400	06D4
4800	036A
9600	01B5
14400	0123
19200	00DA
38400	006D
56000	004B
57600	0049
115200	0024

1. Close jumper located next to connection terminals of the device.
2. Read entire area 0x2001 to 0x2040 to master device. At address 0x2040 check sum of entire area is stored. It is calculated as sum of 16bit values from addresses 0x2001 to 0x2039. Stored are lowest 16 bits of this sum - enables to check correct reading of the area.
3. Modify content corresponding to addresses 0x2001 and 0x2002 in master device as required. **Setting of the other values should not be changed!**
4. Calculate new check sum of entire area, i.e. sum of 16bit values corresponding to values at addresses 0x2001 to 0x2039 and store lowest 16 bits to position corresponding to address 0x2040.
5. Write such modified area together from master device to addresses 0x2001 to 0x2040.
6. Open jumper.

**Example:** Device with address 01h, communication speed 9600Bd, to change to address 9Fh and 115200Bd

For data area **reading** the following is sent via link: **01 03 20 00 00 40 4F FA**

01 device address

03 command for reading of 16-bit registers



After successful writing to the device, device responds: **01 10 20 00 00 40 CA 39** (still with old address at original communication speed) and after response it sets to new values. In case of different number of data or incorrect check sum of the entire area writing to device is not performed.

For more information contact Technical support, please.