Communication protocols description of

Hx4xx regulators with RS485 and RS232 communication interface

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

Description of Hx4xx communication protocols

Devices are preset from manufacturer to Modbus RTU communication protocol[1], address 01, communication speed 9600Bd, no parity, 2 stop bits. If you would like to use different communication protocol then 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. 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[2], 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 .

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)
06 (0x06): Setting of one 16-bit register (Write Register)
16 (0x10): Setting of more 16-bit registers (Write Multiple Registers)*[3]

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

[1] If in order was not specified differently.
[2] Device sends two stop bits, for receive one stop bit is enough.
[3] See detailed description of this function.

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 *9*.

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 *9*.

1.3.3. 06 (0x06): Setting of one 16-bit register (Write Single Register)

There is possible to use this function for address range from 0x42 (66) to 0x4F (79). For write to addresses from 0x45 (69) to 0x4F (79) there is necessary to have set value 1 at address 0x44 (68) – Modbus Remote Control Enabled.

1.3.4. 16 (0x10): Setting of several 16-bit registers (Write Multiple Registers)

It is possible to use this command to:

- output relays parameters setting. Address range from 0x44 (68) to 0x4F (79)
- “Far” controlling of output relays status (open / close). Address range 0x42 (66) and 0x43 (67).

Communication examples are listed in chapter 4 Appendix A at page 20.

Change of device address and communication speed by Modbus protocol is described at chapter 5 Appendix B at page 26. Rest of parameters there is possible to set only by device’s keyboard or with User’s software Tsensor. 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 Tsensor to set all device parameters. It is free to download at 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??

1.3.5. 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.

1.3.6. 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.
0x03 Illegal data value. A value contained in the data field is not an allowable value for the 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

Register addresses are indexed from zero - „zero based addressing“. For example „measured temperature“ with Modbus address 0x31 (49) is physically sent along data bus as value 0x30 (48). 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 (49) or 0x30 (48)).

Communication examples are listed at Appendix A - Example of Modbus communication at page 20.

1.5.1. Common registers

Variable	Unit	Address [hex]^{X}	Address [dec]^{X}	Format	Size	Status
measured temperature	[°C/°F]*	0x0031	49	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**

1.5.2. Addition for regulators with relative humidity measurement

Variable	Unit	Address [hex]^{X}	Address [dec]^{X}	Format	Size	Status
measured relative humidity	[%]	0x0032	50	Int*10	BIN16	R
computed value*		0x0033	51	Int*10	BIN16	R

1.5.3. Addition for H7430 and H7431 regulators with barometric pressure measurement:

Variable	Unit	Address [hex]^{X}	Address [dec]^{X}	Format	Size	Status
barometric pressure	hPa	0x0034	52	Int*10	BIN16	R
	PSI			Int*1000		
	inHg			Int*100		
	mBar			Int*10		
	oz/in^2			Int*10		
	mmHg			Int*10		
	inH$_2$O			Int*10		
	kPa			Int*100		

Legend:

- * depends on device setting (by device's keyboard or 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, but there is not possible to write directly, for details see description of Modbus command in chapter 1.3.4 16 (0x10): Setting of several 16-bit registers (Write Multiple Registers) at page 7.
- 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.

1.5.4. Far controlling of output relays status:

This function enables change of output relay status by Modbus command. It is necessary to assign quantity “Far0” or “Far1” to output relay at first (use devices keyboard or Tsensor software). Then output relay status is done by value of next Modbus registers:

Function	Address [hex]^{X}	Address [dec]^{X}	Description	Status
Far condition for relay 1	0x0042	66	0 ... relay1 open 1 ... relay 1 closed	R/W
Far condition for relay 2	0x0043	67	0 ... relay 2 open 1 ... relay 2 closed	R/W

This is supported from internal firmware version 04.06 (devices with Ethernet interface with Ethernet firmware at least 1.5.2.04). If older firmware used, update your firmware or contact device’s distributor.

1.5.5. Output relay alarm condition configuration with using of Modbus communication protocol

This function is available from firmware version 04.06. If older firmware used, update your firmware or contact device’s distributor.

All registers contains 16b number.

Function	Unit	Address [hex]^{X}	Address [dec]^{X}	Description	Status
Modbus Remote Control	[-]	0x0044	68	0 ... disable 1 ... enable (for this time device’s keyboard is blocked, if key is pressed, message BLOC is shown)	R/W
Value assigned to output relay 1	by value	0x0045	69	0 ... Off 1 ... temperature 2 ... relative humidity 3 ... atmospheric pressure 4 ... computed value 5 ... binary input 1 6 ... binary input 2 7 ... binary input 3 8 ... Far condition 0 9 ... Far condition 1	R/W
When close relay 1	[-]	0x0046	70	0 ... alarm occurs, when measured value is LOWER than preset alarm value 1 ... alarm occurs, when measured value is HIGHER than preset alarm value	R/W
Alarm value for relay 1	by value	0x0047	71	preset alarm value (threshold)	R/W

Delay for relay 1	[sec]	0x0048	72	Time in seconds Time in seconds that the condition must be valid to evaluate alarm	R/W
Hysteresis of relay 1	by value	0x0049	73	Value wihch must be set back below / above a given threshold in order to open the relay	R/W
Value assigned to output relay 2	by value	0x004A	74	0 ... Off 1 ... temperature 2 ... relative humidity 3 ... atmospheric pressure 4 ... computed value 5 ... binary input 1 6 ... binary input 2 7 ... binary input 3 8 ... Far condition 0 9 ... Far condition 1	R/W
When close relay 2	[-]	0x004B	75	0 ... alarm occurs, when measured value is LOWER than preset alarm value 1 ... alarm occurs, when measured value is HIGHER than preset alarm value	R/W
Alarm value for relay 2	by value	0x004C	76	preset alarm value (threshold)	R/W
Delay for relay 2	[sec]	0x004D	77	Doba v sekundách, po kterou musí být podmínka platná, aby byl vyhodnocen alarm	R/W
Hysteresis of relay 2	by value	0x004E	78	Value which must be set back below / above a given threshold in order to open the relay	R/W
Confirm changes	[-]	0x004F	79	Read always as 0 write 1 ... physically stores new setup into device’s memory, at the end the register is automatically reset to zero.	R/W

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 13).

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: **!AATTCCFF cr** symbols correspond with paragraph "Configuration of device" at page "13"

2.4.2. Reading of device name

Syntax of command: **$AAM cr**
Response: **!AAH**xxxx**(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^2	>+0225.1
mmHg	>+0728.1
inH_2O	>+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 16.
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 = **8E**h, 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 = **B8**h, 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 = **96**h, 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 = **B9**h, then it is sent as **0x42, 0x39**

Response: **>+ >+000001 8A cr**
where it is sent: 3E 2B 30 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

4.1. Example of Modbus communication

In all examples communication with device at address 01 is supposed

4.1.1. Reading of temperature, address 0x0031 (49)

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	

4.1.2. Reading of relative humidity, address 0x0032 (50)

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	

4.1.3. Reading of computed value, address 0x0033 (51)

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	

4.1.4. Reading of all values at once, address block 0x0031 (49) to 0x0033 (51)

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 device's keyboard or User's software.

4.1.5. All output relay parameters setting by one command (at once)

This command sets all output relay parameters at once. Example: Assign relative humidity to relay 1 and close relay 1 when relative humidity is higher than 60% RH, close delay 2 minutes, and open relay 1 when relative humidity is less than 55% RH. To relay 2assign temperature and close relay 2 when temperature is less than 5°C. Open relay 2 when temperature reach 7°C. Close delay 60s.

```
Modbus command:
Device ADDRESS                      0x01
----------------------------------------------------------------------------------------------------
FUNCTION    write 16-bit registers  0x10
----------------------------------------------------------------------------------------------------
DATA        initial address Hi      0x00
            initial address Lo      0x43
            ----------------------------------------------------------------------------
      number read registers Hi      0x00
      number read registers Lo      0x0C          it means 12 registers
            ----------------------------------------------------------------------------
            Byte number             0x18          it means 24 transmitted Bytes
----------------------------------------------------------------------------------------------------
            data            0x00 0x01     to address 0x44 write 1, setting enable
                            0x00 0x02     to address 0x45 write 2, relay1= relative humidity
                            0x00 0x01     to address 0x46 write 1, alarm if„Higher than“
                            0x02 0x58     to address 0x47 write 0x258, limit 60.0% RH
                            0x00 0x78     to address 0x48 write 0x78, delay 120s
                            000x 0x32     to address 0x49 write 0x32, hysteresis 5.0%
                            0x00 0x01     to address 0x4A write 1, relay2 = temperature
                            0x00 0x00     to address 0x4B write 0, alarm, if „Lower than“
                            0x00 0x32     to address 0x4C write 0x32, limit 5.0°C
                            0x00 0x3C     to address 0x4D write 0x3C, delay 60s
                            0x00 0x14     to address 0x4E write 0x14, hysteresis 2.0°C
                            0x00 0x01     to address 0x4F write 0x01, confirm changes
----------------------------------------------------------------------------------------------------
                            0x1B 0x18     check sum CRC

Response – command approved:
Device Address                      0x01
----------------------------------------------------------------------------------------------------
FUNCTION    reading 16-bit registers  0x10
----------------------------------------------------------------------------------------------------
DATA        initial address Hi      0x00
            initial address Lo      0x43
            ----------------------------------------------------------------------------
      number read registers Hi      0x00
      number read registers Lo      0x0C          it means 12 registers
            check sum CRC           0x31 0xD8
```

4.1.6. Separated output relays parameters adjustment

Setting procedure is as follows:

1) To address 0x0044 (68) write value 1 – Modbus setting enable. Local keyboard is blocked during remote control setting change.

Modbus command:

Device ADDRESS		0x01
FUNCTION	write (single) 16-bit register	0x06
DATA	address Hi	0x00
	address Lo	0x43
	value Hi	0x00
	value Lo	0x01
	check sum CRC	0xB9 0xDE

Response – command approved:
0x01 0x06 0x00 0x43 0x00 0x01 0xB9 0xDE

2) Write desired settings / changes to addresses from 0x0045 (69) to 0x004E (78). New setting can be written register by register (Modbus command 0x06) or as registers sequence (Modbus command 0x10).

For example we need to change value assigned to relay2 to relative humidity (address 0x4A) and set alarm limit to 25.0% (address 0x4C). Other parameters without change:

Change of assigned value of relay2 to relative humidity by Modbus command:

Device ADDRESS		0x01
FUNCTION	write (single) 16-bit register	0x06
DATA	address Hi	0x00
	address Lo	0x49
	value Hi	0x00
	value Lo	0x02
	CRC	0xD9 0xDD

Response – command approved:
0x01 0x06 0x00 0x49 0x00 0x02 0xD9 0xDD

Modbus command for relay2 alarm limit change to 25.0%:

Device ADDRESS		0x01
FUNCTION	write (single)16-bit register	0x06
DATA	address Hi	0x00
	address Lo	0x4B

	value Hi	0x00
	value Lo	0xFA
	CRC	0x79 0xF9

Response – command approved:
0x01 0x06 0x00 0x4B 0x00 0xFA 0x79 0x9F

It is possible to make other output relay value changes if necessary with using of Modbus commands 0x06 and 0x10.

3) Write to address 0x004F (79) value 1. This step confirms setting process finally and setting is stored into device's memory. At the end registers 0x004F (79) and 0x0044 (68) are automatically cleared. It unblocks (enables) device's keyboard too.

Modbus command to confirm changes:

Device ADDRESS		0x01
FUNCTION	write (single) 16-bit register	0x06
DATA	address Hi	0x00
	address Lo	0x4E
	value Hi	0x00
	value Lo	0x01
	CRC	0x28 0x1D

Response – command approved:
0x01 0x06 0x00 0x4E 0x00 0x01 0x28 0x1D

By this step setup is completed.

4.1.7. Example of output relay parameters setup process canceling

The entire setup process can be cancelled at any time with writing value 0 to address 0x0044 (68). In that case no changes are stored and last valid setting is restored.

Example: the same situation as example before, but during setting change you want to cancel (interrupt) setup process and restore previous setup.

The beginning is the same as previous example. To address 0x0044 (68) write value 1 – Modbus setting enable.

Modbus command: 0x01 0x06 0x00 0x43 0x00 0x01 0xB9 0xDE

Response – command approved: 0x01 0x06 0x00 0x43 0x00 0x01 0xB9 0xDE

Now for example change of assigned value of relay2 to relative humidity by Modbus command is done:
0x01 0x06 0x00 0x49 0x00 0x02 0xD9 0xDD

Response – command approved:
0x01 0x06 0x00 0x49 0x00 0x02 0xD9 0xDD

If you want to cancel this setup process now and restore previous device setting then you must write into register „Modbus Remote Control“ at address 0x44 (68) value „0“.

```
Modbus command:
Device ADDRESS                          0x01
----------------------------------------------------------------------------------
FUNCTION   write (single) 16-bit register  0x06
----------------------------------------------------------------------------------
DATA       address Hi                   0x00
           address Lo                   0x43
           -----------------------------------------------------------------------
           value Hi      0x00
           value Lo      0x00
           CRC                          0x78 0x1E
Response – command approved:
0x01 0x06 0x00 0x43 0x00 0x00 0x78 0x1E
```

It stopped the setup process, restored previous regulator setting and unblocked keyboard.

4.1.8. Relay1 output close

To enable remote control of output relay1, it is necessary to assign “Far condition0” or “Far condition1” value. These two conditions are different in the state in which output relay is after regulator’s power on (open or close). Output relay state is controlled by value written into Modbus register at address 0x42 (66) for relay1 and 0x43 (67) for relay2. Vlaue „0“ opened, value „1“ closed.

Modbus command for output relay1 close:

```
Device ADDRESS                          0x01
----------------------------------------------------------------------------------
FUNCTION   write (single) 16-bit register  0x06
----------------------------------------------------------------------------------
DATA       address Hi                   0x00
           address Lo                   0x41
           -----------------------------------------------------------------------
           value Hi      0x00
           value Lo      0x01
           CRC                          0x18 0x1E

Response – command approved:
0x01 0x06 0x00 0x41 0x00 0x01 0x18 0x1E
```

4.1.9. Relay1 output open

It is done in similar way as example before, but there is written value “0”:

Modbus command for output relay1 open:

```
Device ADDRESS                                  0x01
-----------------------------------------------------------------------------------
FUNCTION    write (single) 16-bit register      0x06
-----------------------------------------------------------------------------------
DATA        address Hi                          0x00
            address Lo                          0x41
            -------------------------------------------------------------------
            value Hi        0x00
            value Lo        0x00
            CRC                                 0xD9 0xDE
```

Response – command approved:
0x01 0x06 0x00 0x41 0x00 0x00 0xD9 0xDE

5. Appendix B

5.1. Proccess of device address and communication speed change by Modbus command

Writing into device’s registers with Modbus RTU protocol using
It is strongly recommended to use device’s keyboard or user’s software *Tsensor* for setting of all device’s parameters. It is free to download at 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

20 00 initial address (sent address is indexed from zero, during Modbus request for reading of address 0x2001 physically is sent 0x2000 via link)

00 40 read 64 registers (0x40), i.e. 128 Byte

4F FA CRC of Modbus

response of the device is:

01 03 80 **00 01 01 B5** 00 00 30 30 3B 4B 77 D3 BD 35 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 84 70 00 00 86 2A 00 00 84 44 AA 80 85 07 A8 D0 57 7E 5F 94 F3 DC 00 12 2E DD 78 0C 40 AA 77 D3 F2 C4 00 12 17 78 77 F5 F3 EC 00 12 ED BF 77 D5 4F 10 77 D8 FF FF FF FF 40 DE 77 D3 2E F7 78 0C 06 5C 00 01 00 00 00 00 F3 DC 00 12 42 9F **53 2D** 2C 8C

01 device address

03 code of command (reading 16-bit registers)

80 number of Bytes of response (0x80, i.e. 128 Byte)

00 01 address content 0x2001, i.e. original address of device 01h

01 B5 address content 0x2002, i.e. code of corresponding communication speed 9600Bd

…

…

53 2D check sum of entire area 0x2001 to 0x2039
(0001+01B5+0000+3030+…+0012+429F = 0x532D)

2C 8C CRC of Modbus

Modification of data from read area:

00 9F 00 24 00 00 30 30 3B 4B 77 D3 BD 35 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 84 70 00 00 86 2A 00 00 84 44 AA 80 85 07 A8 D0 57 7E 5F 94 F3 DC 00 12 2E DD 78 0C 40 AA 77 D3 F2 C4 00 12 17 78 77 F5 F3 EC 00 12 ED BF 77 D5 4F 10 77 D8 FF FF FF FF 40 DE 77 D3 2E F7 78 0C 06 5C 00 01 00 00 00 00 F3 DC 00 12 42 9F **52 3A**

00 9F new address of the device (original address was 00 01)
00 24 new code of corresponding communication speed 115200Bd (original was 01B5)
…
…
52 3A new check sum of entire area (009F+0024+0000+…+0012+429F = 0x523A)

Write such modified area together from master device to the **device, area 0x2001 to 0x2040**

01 10 20 00 00 40 80 00 9F 00 24 00 00 30 30 3B 4B 77 D3 BD 35 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 84 70 00 00 86 2A 00 00 84 44 AA 80 85 07 A8 D0 57 7E 5F 94 F3 DC 00 12 2E DD 78 0C 40 AA 77 D3 F2 C4 00 12 17 78 77 F5 F3 EC 00 12 ED BF 77 D5 4F 10 77 D8 FF FF FF FF 40 DE 77 D3 2E F7 78 0C 06 5C 00 01 00 00 00 00 F3 DC 00 12 42 9F 52 3A **61 22**

01 original device address is still valid, i.e. 01h
10 code of command, setting more 16-bit registers
20 00 initial address
00 40 number of write registers
80 number of Bytes of sent data
61 22 CRC of Modbus

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.