PAPAGO METEO RS

Measures temperature, humidity, dew point, atm. pressure, CO_2 concentration, wind speed and direction

RS485 interface

Modbus RTU communication protocol

PAPAGO METEO RS

Datasheet

Created: 1.3.2021

Last update: 9.11 2021 12:33

Number of pages: 28

Papouch s.r.o.

Address:

Strasnicka 3164
102 00 Prague 10
Czech Republic

Tel:

+420 267 314 267

Internet:

www.papouch.com

E-mail:

info@papouch.com

TABLE OF CONTENTS

Getting to know Papago....................................4

Applications 4

Features 4

Connections..5

Configuration 6

Communication tab....................................7

Sensor tabs..7

Modbus RTU...8

Address 8

Function code list..8

Device identification 9

Holding Register 9

Input Register 11

Spinel communication protocol 16

Format 97..16

Structure 16

Explanatory notes 17

Spinel instructions 18

Read values from input register 18

Reading of name and version 18

Reading of manufacturing data 19

Indications 20

Technical parameters 21

Integrated temperature and humidity sensor TH3..21

Integrated pressure, temperature and humidity sensor..22

Standalone temperature sensor 22

Sensor cable...23

CO_2 concentration sensor 23

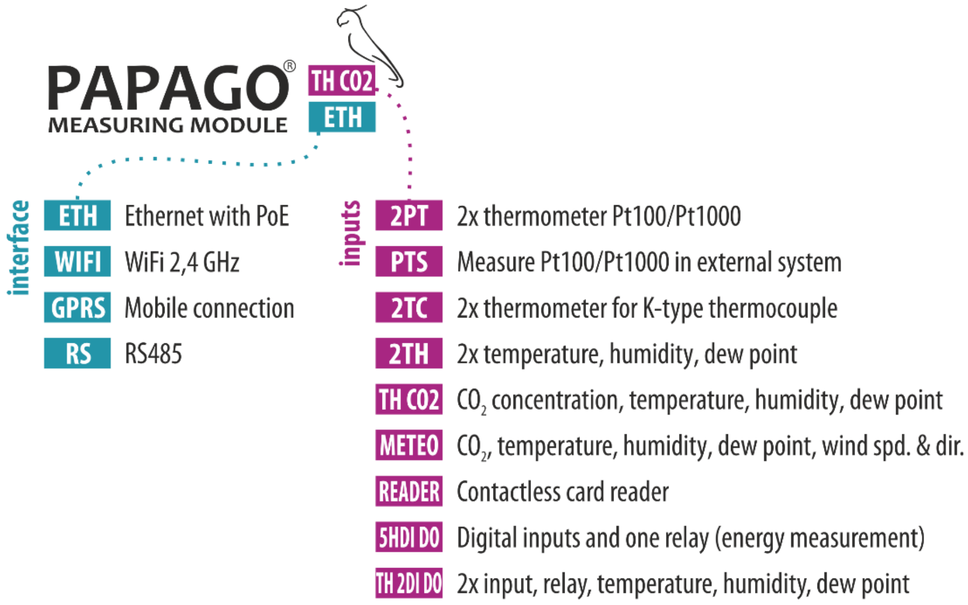
Wind speed and direction sensor 24

Other parameters......................................24

Available designs......................................25

GETTING TO KNOW PAPAGO

PAPAGO is a family of devices with uniform appearance and communication capabilities. It allows to combine communication interfaces on one side and measuring sensors (inputs) on the other side.


PAPAGO®
MEASURING MODULE
TH CO2
ETH
interface
ETH Ethernet with PoE
WIFI WiFi 2,4 GHz
GPRS Mobile connection
RS RS485
inputs
2PT 2x thermometer Pt100/Pt1000
PTS Measure Pt100/Pt1000 in external system
2TC 2x thermometer for K-type thermocouple
2TH 2x temperature, humidity, dew point
TH CO2 CO$_2$ concentration, temperature, humidity, dew point
METEO CO$_2$, temperature, humidity, dew point, wind spd. & dir.
READER Contactless card reader
5HDI DO Digital inputs and one relay (energy measurement)
TH 2DI DO 2x input, relay, temperature, humidity, dew point

Applications

- Monitoring of meteorological values and processing
- Measurement of CO$_2$ levels, wind speed and direction in industry
- Expansion of PLCs – adding meteorological values
- Measurement for the HACCP system
- Detection of wind speed for automatic retraction of various sun blinds

Features

PAPAGO METEO RS measures CO2 level, temperature, humidity, dew point, atmospheric pressure, wind speed and wind direction.

One of the following sensors can be connected to given input (One sensor per input):

TH sensor	temperature -40 to 125 °C; humidity 0 to 100 %	input: A, B
THP sensor	temp. -40 to 125 °C; hum. 0 to 100 %; pressure 50 to 110 kPa	input: A, B
T sensor	temperature -55 to 125 °C	input: A, B
CO$_2$ sensor	CO$_2$ level	input: A, B
Wind sensor	wind speed and direction	input: C

- RS485 interface with Modbus RTU protocol.
- Wide power voltage input 11 to 58 V DC.
- Current consumption typ. 26 mA @ 24 V.

- Measures (a) external thermometer, (b) combined sensor with temperature and humidity, (c) combined pressure, temperature and humidity sensor, (d) CO_2 level concentration or (e) Wind speed / direction sensor (Sensors are sold separately.)
- Robust sleek metal chassis that can be DIN rail mounted. Connections are inscribed on the chassis for easy wiring. LED indicators show all important states of the device.

CONNECTIONS

1) Connect power source to + and – terminals. Papago supports 11 to 58 V DC. Power input is reverse voltage protected.
2) Connect sensors to connectors A, B and C. Connector C is designated for Wind sensor. A and B connectors are interchangeable.

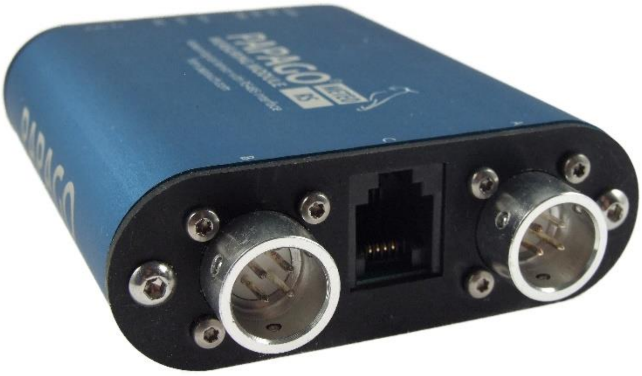


fig. 1 – front panel with sensor connectors

3) Connect Papago to RS485 master using twisted pair cable. RxTx+ connect to RxTx+ (also can be RT+ or A) on the other device. Likewise – connect RxTx- to the equivalent RxTx- (can be RT- or B).

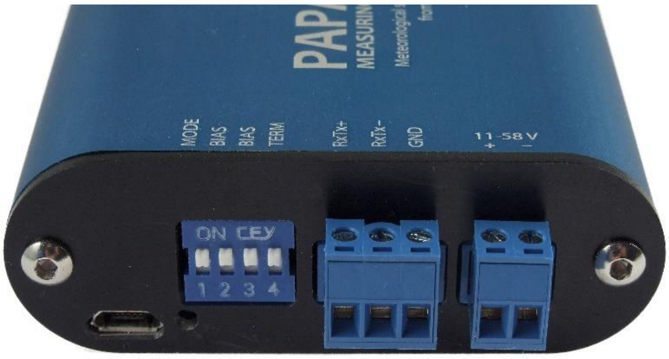
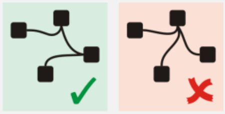


fig. 2 – back panel with RS485 terminal, power terminal and configuration USB connector

Some basic recommendations for connecting the RS485 line:

- It is recommended to use a standard TP cable for computer networks (UTP, FTP or STP) and to use one twisted pair from this cable as the conducting wires for RS485.



- All devices on the line must be connected “one after the other” and not in a “star” (see right). The maximum length of the line is 1.2 km.
- Cable shielding is to be connected on one side only.
- End of the line must be terminated using the TERM switch

TERM switch – connect RS485 termination in case the line is inside high-interference environment (or in case there are power lines near the bus line). One RS485 line can only have 2 devices with termination connected on opposite sides of the line. Most installations don’t require line termination.

BIAS switches can be connected to the line to define the idle state of the line. These resistors can only be connected once per the entire RS485 line.

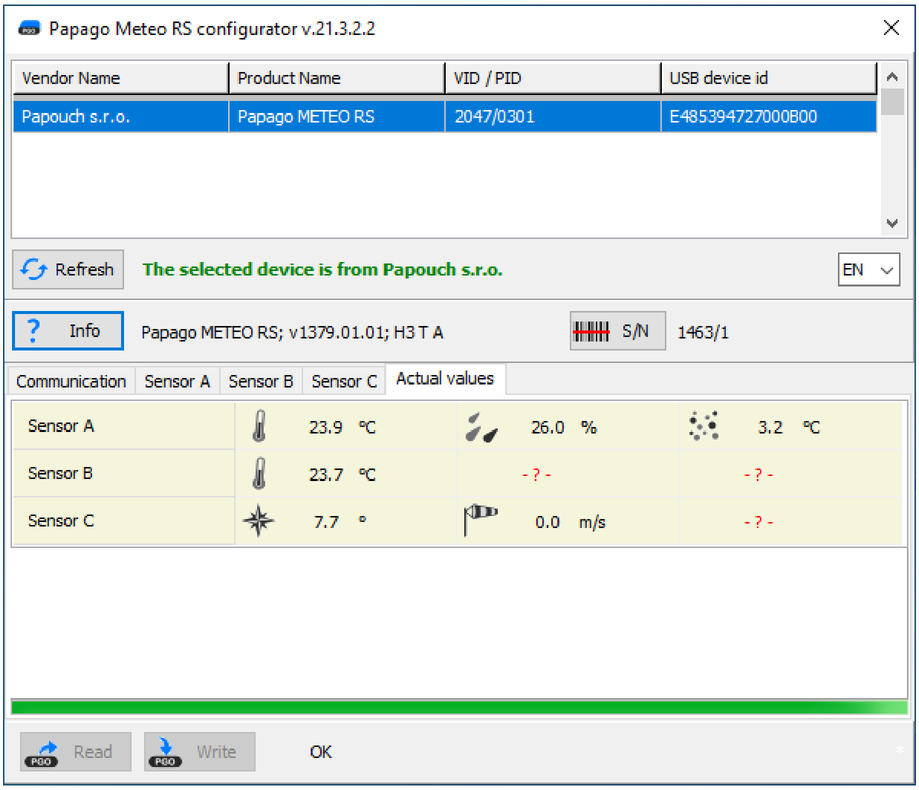
GND can be connected to the cable shielding. Shielding of RS485 line is not necessary. We recommend using shielding in case the line is inside high-interference environment (or in case there are power lines near the bus line). **Shielding must only be connected on ONE device within the line!** Otherwise, potentials from different grounds would equalize through the shielding and would cause what is known as ground loop. This can damage or destroy devices on the line.

Advice: **Ground GND of the serial line is galvanically isolated from other parts of the device.** Unless there is a distinct reason, do not connect grounds. Connecting both grounds will cancel the galvanic isolation of the communication line and PAPAGO is affected by ground loops between it and the control system.

4) Unless you have PAPAGO set up beforehand, continue to the next chapter.

CONFIGURATION

Configuration is done using the mini USB connector and software *Papago Meteo RS configurator* for Windows 10 OS. Software is freely available for download on Papouch.com. Configuration requires connected external power to PAPAGO.



Papago Meteo RS configurator v.21.3.2.2

Vendor Name	Product Name	VID / PID	USB device id
Papouch s.r.o.	Papago METEO RS	2047/0301	E485394727000B00

The selected device is from Papouch s.r.o.

Papago METEO RS; v1379.01.01; H3 T A

S/N 1463/1

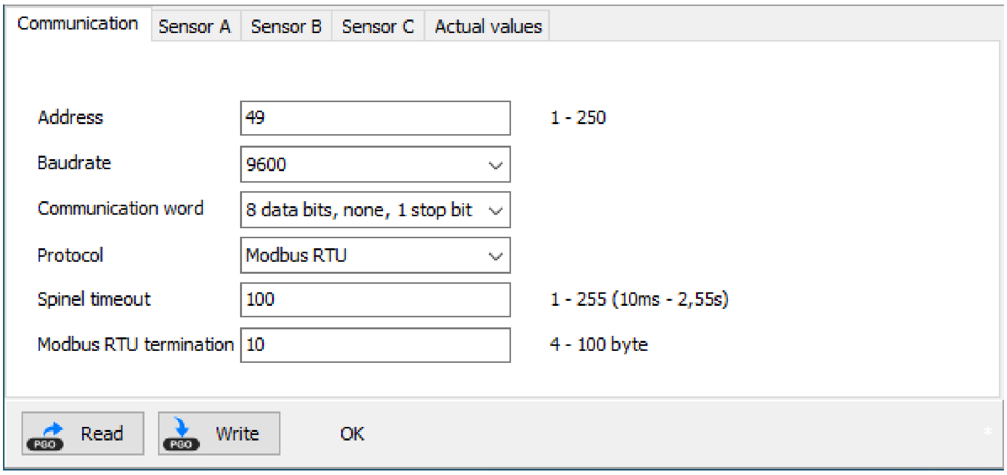
Sensor A	23.9 °C	26.0 %	3.2 °C
Sensor B	23.7 °C	- ? -	- ? -
Sensor C	7.7 °	0.0 m/s	- ? -

OK

fig. 3 – Configuration application

Communication tab

Communication tab contains basic line communication parameters.

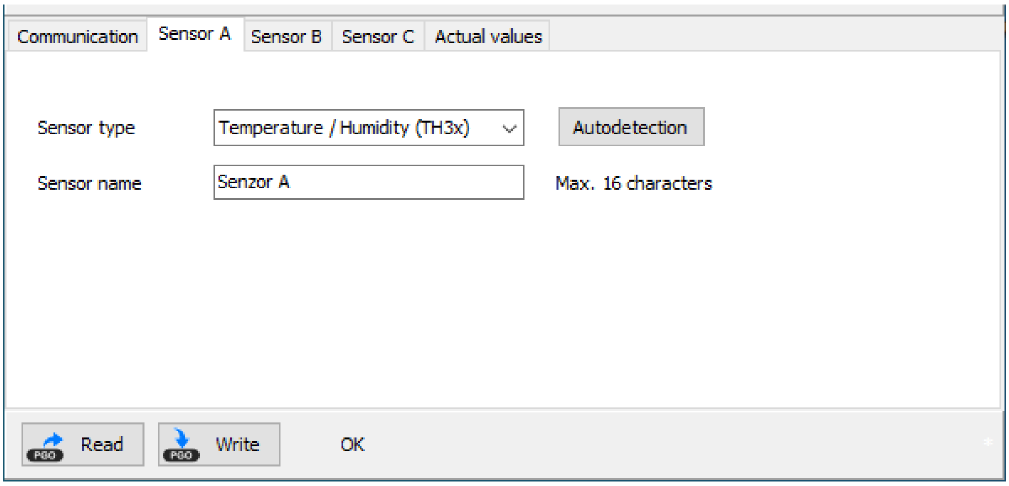


Address	49	1 - 250
Baudrate	9600	
Communication word	8 data bits, none, 1 stop bit	
Protocol	Modbus RTU	
Spinel timeout	100	1 - 255 (10ms - 2,55s)
Modbus RTU termination	10	4 - 100 byte

fig. 4 – Communication line parameters settings

Sensor tabs

Both A and B sensors have their respective tabs with identical settings.

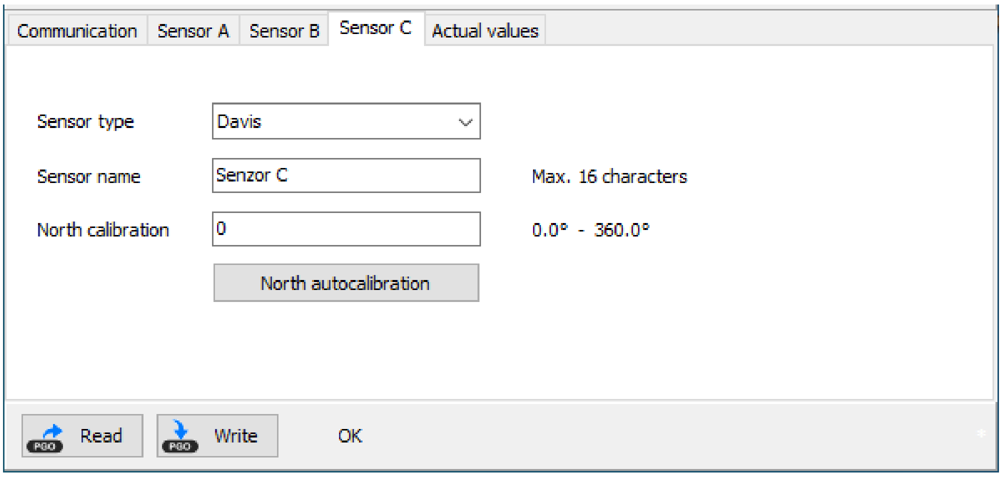


Sensor type	Temperature / Humidity (TH3x)	Autodetection
Sensor name	Senzor A	Max. 16 characters

fig. 5 – sensor settings

Push the *Autodetection* button and sensor will be recognized automatically.

Sensor C must be anemometer only (wind speed and direction sensor) Sensor C tab looks like following image:



Sensor type: Davis

Sensor name: Senzor C — Max. 16 characters

North calibration: 0 — 0.0° - 360.0°

OK

fig. 6 - sensor C tab

MODBUS RTU

Address

Tip: Address can be easily set using the configurations software via USB cable. This is the basic information about ModBus RTU:

- 0x31: Default address of the device (decadic 49). Address can be changed in register 1 (see below).
- 0x00: Universal ModBus RTU address (0 decadic). If a device receives an instruction with this address, the instruction will be carried out, but no response will be sent.
- 0xF8: Universal device address (248 decadic). If a device receives an instruction with this address, instruction will be carried out and the device will respond. This can really only be used when a single device is connected!

How to change address using a serial number?

You can set up multiple devices with the same default address on a single RS485 line using the following set of instructions:

1) Write down the given device serial number. It is written on a label in format *1395/0069*
   The number before the slash is a product number, the number after the slash is the serial number of the given unit.
2) Write following Holding registers using function code 0x10 and universal address:
   a. *Product type* (addr. 10) – write product type from label.
   b. *Serial number* (addr. 11) – write serial number from label.
   c. *Address* (addr. 12) – write new address you want to set.
3) From now on the device communicates on its new address.

Function code list

The internal memory of the device is accessible – based on the registry type – using these instructions:

- 0x03 .....read holding registers

- 0x04 .....read input registers
- 0x06 .....set one holding register
- 0x10 .....write to multiple holding registers
- 0x11 .....identification

Device identification

Read identification string of the device (Report slave ID).

Function codes:

0x11 – Report slave ID

Parameters:

Byte count	1 Byte	Based on the string
ID	1 Byte	ID is identical to device address
RI	1 Byte	Run Indicator – this is always 0xFF (ON)
Data	N Byte	The string should have this form: *Papago METEO RS; v1379.01.01; H3 T A*

Holding Register

Address	*Access*	*Function*	*Name*
0 [1]	write	0x06	**Allow configuration** Writing value 0x00FF in this part of memory must precede all instructions writing in holding registers addressed 0 to 5. This is to protect against an accidental configuration change. Allow configuration using function code 0x10 along with other parameters is not allowed..
1	read, write	0x03, 0x06, 0x10	**Address (ID)** [2] Unique address of the device in ModBus protocol. Number between 1 to 247 is expected. *Default address is 0x0031.* See page 8 to set address using serial number.
2	read, write	0x03, 0x06, 0x10	**Communication speed** [2] Speeds ant their respective codes: 1 200 Bd ....0x0003 2 400 Bd ....0x0004 4 800 Bd ....0x0005 *9 600 Bd ....0x0006 (default setting)* 19 200 Bd ....0x0007 38 400 Bd ....0x0008 57 600 Bd ....0x0009 115 200 Bd 0x000A

[1] Registers can be numbered from zero or one depending on vendor. First register actually has an address 0.

[2] Writing to this memory space must be preceded by writing 0x00FF to address 0 position Allow configuration. This is to protect against accidental changes in configuration. Allow configuration must not be written using function code 0x10 (write to multiple registers).

<table>
<tr><th><em>Address</em></th><th><em>Access</em></th><th><em>Function</em></th><th><em>Name</em></th></tr>
<tr><td>3</td><td>read, write</td><td>0x03, 0x06, 0x10</td><td><strong>Data word</strong> [2]
Data word is always 8-bit.
<table><tr><th>Value</th><th>Parity</th><th>Stop bits</th></tr><tr><td><em>0x0000 (default)</em></td><td><em>none (N)</em></td><td><em>1</em></td></tr><tr><td>0x0001</td><td>even (E)</td><td>1</td></tr><tr><td>0x0002</td><td>odd (O)</td><td>1</td></tr></table></td></tr>
<tr><td>4</td><td>read, write</td><td>0x03, 0x06, 0x10</td><td><strong>Packet end distinction</strong> [2]
Configures the delay at the end of each packet to determine packet end. Delay is entered in number of bytes. Value can range from 4 to 100, default value is 10.</td></tr>
<tr><td>5</td><td>read, write</td><td>0x03, 0x06, 0x10</td><td><strong>Communication protocol</strong> [2]
Allows for switching of the device to Spinel protocol. After sending a response, device will switch to Spinel and will communicate with is only. Each protocol has instruction to switch the device to the other protocol.
Spinel protocol code: 0x0001
ModBus RTU protocol code: <em>0x0002 (default)</em></td></tr>
<tr><td>10</td><td>read, write</td><td>0x03, 0x10 [3]</td><td><strong>Product type</strong>
This is always 1395 as a product type.</td></tr>
<tr><td>11</td><td>read, write</td><td>0x03, 0x10 [3]</td><td><strong>Serial number</strong>
Unique serial number.</td></tr>
<tr><td>12</td><td>read, write</td><td>0x03, 0x10 [3]</td><td><strong>Address</strong>
See set address using serial number on page 8.</td></tr>
<tr><td>13</td><td>read, write</td><td>0x03, 0x10</td><td><strong>Hardware and firmware version</strong>
Upper byte is hardware version, lower byte is firmware version.</td></tr>
<tr><td>20</td><td>read, write</td><td>0x03, 0x10</td><td><strong>Set sensor A sensor type</strong>
Sensor type is one of these codes:
• 0 – not set
• 2 – Temperature (DS)
• 3 – Temperature - humidity (TH3x)
• 4 – Temperature (TMP)
• 5 – CO_2 concentration (T6713)
• 7 – Atmospheric pressure
• 8 – O_3 concentration</td></tr>
</table>

[3] Registers 10 and 12 have to be written together. Writing will not change values in registers Product type and serial number. Writing these values is only for setting address using the serial number. (see page 8).

Address	*Access*	*Function*	*Name*
30	read, write	0x03, 0x10	**Set sensor B sensor type** Sensor type is one of these codes: • 0 – not set • 2 – Temperature (DS) • 3 – Temperature - humidity (TH3x) • 4 – Temperature (TMP) • 5 – CO_2 concentration (T6713) • 7 – Atmospheric pressure • 8 – O_3 concentration
40	read, write	0x03, 0x10	**Set sensor C sensor type** Sensor type is one of these codes: • 0 – not set • 6 – Davis (wind speed and direction)
49	read, write	0x03, 0x10	**North direction calibration** If 0 is set, the mark on wind sensor must head directly north. By writing value from 1 to 359 the direction can be corrected with resolution of one degree. (For example if mounting the sensor heading north is not an option.)

Input Register

Current values from all sensors are available in Input registers. All values have their assigned registers. **Values are updated only in registers of values that the connected sensor can actually measure.**

Address	*Access*	*Function*	*Name*
Sensor A – head			
0[4]	read	0x04	**Sensor type** Sensor type is one of these codes: • 0 – not set • 2 – Temperature (DS) • 3 – Temperature - humidity (TH3x) • 4 – Temperature (TMP) • 5 – CO_2 concentration (T6713) • 7 – Atmospheric pressure • 8 – O_3 concentration
1	read	0x04	**Status** Contains sensor status. It can have these values: 0 = sensor is connected 1 = sensor is disconnected
Sensor A – temperatures			

[4] Some systems can use register numbering from 1.

Address	*Access*	*Function*	*Name*
20	read	0x04	**Value status** Contains value status. It can have these values: 0 = measured value is within range 2 = upper limit has been exceeded (overflow) 3 = lower limit not reached (underflow) 4 = measured value is invalid
21	read	0x04	**Value in degrees Celsius as a signed integer x10**[5]
22, 23	read	0x04	**Value in degrees Celsius as a float**
24	read	0x04	**Value in Fahrenheits as a signed integer x10**[5]
25, 26	read	0x04	**Value in Fahrenheits as a float**
27	read	0x04	**Value in Kelvins as a signed integer x10**[5]
28, 29	Read	0x04	**Value in Kelvins as a float**
Sensor A – humidity			
40	read	0x04	**Value status** Contains value status. It can have these values: 0 = measured value is within range 2 = upper limit has been exceeded (overflow) 3 = lower limit not reached (underflow) 4 = measured value is invalid
41	read	0x04	**Value in percent as a signed integer x10**[5]
42, 43	read	0x04	**Value in percent in float format**
Sensor A – dew point			
60	read	0x04	**Value status** Contains value status. It can have these values: 0 = measured value is within range 2 = upper limit has been exceeded (overflow) 3 = lower limit not reached (underflow) 4 = measured value is invalid
61	read	0x04	**Value in degrees Celsius as a signed integer x10**[6]
62, 63	read	0x04	**Value in degrees Celsius as a float**
64	read	0x04	**Value in Fahrenheits as a signed integer x10**[5]
65, 66	read	0x04	**Value in Fahrenheits as a float**
67	read	0x04	**Value in Kelvins as a signed integer x10**[5]
68, 69	Read	0x04	**Value in Kelvins as a float**
Sensor A – CO_2 concentration			

[5] The actual value can be calculated by dividing the number by 10. For example 123 is 12.3 .

[6] The actual value can be calculated by dividing the number by 10. For example 123 is 12.3 .

Address	*Access*	*Function*	*Name*
80	read	0x04	**Value status** Contains value status. It can have these values: 0 = measured value is within range 2 = upper limit has been exceeded (overflow) 3 = lower limit not reached (underflow) 4 = measured value is invalid
81	read	0x04	**Value in percent as a signed integer x10**[5]
82, 83	read	0x04	**Value in percent in float format**
Sensor A – atmospheric pressure			
100	read	0x04	**Value status** Contains value status. It can have these values: 0 = measured value is within range 2 = upper limit has been exceeded (overflow) 3 = lower limit not reached (underflow) 4 = measured value is invalid
101	read	0x04	**Value in hPa as a signed integer x10**[5]
102, 103	read	0x04	**Value in hPa in float format**
104	read	0x04	**Value in bar as a signed integer x10**[5]
105, 106	read	0x04	**Value in bar in float format**
Sensor B – head			
500	read	0x04	**Sensor type** Sensor type is one of these codes: • 0 – not set • 2 – Temperature (DS) • 3 – Temperature - humidity (TH3x) • 4 – Temperature (TMP) • 5 – CO_2 concentration (T6713) • 7 – Atmospheric pressure • 8 – O_3 concentration
501	read	0x04	**Status** Contains sensor status. It can have these values: 0 = sensor is connected 1 = sensor is disconnected
Sensor B – temperature			
520	read	0x04	**Value status** Contains value status. It can have these values: 0 = measured value is within range 2 = upper limit has been exceeded (overflow) 3 = lower limit not reached (underflow) 4 = measured value is invalid
521	read	0x04	**Value in degrees Celsius as a signed integer x10**[7]

[7] The actual value can be calculated by dividing the number by 10. For example 123 is 12.3 .

Address	*Access*	*Function*	*Name*
522, 523	read	0x04	**Value in degrees Celsius as a float**
524	read	0x04	**Value in Fahrenheits as a signed integer x10**[5]
525, 526	read	0x04	**Value in Fahrenheits as a float**
527	read	0x04	**Value in Kelvins as a signed integer x10**[5]
528, 529	Read	0x04	**Value in Kelvins as a float**
Sensor B – humidity			
540	read	0x04	**Value status** Contains value status. It can have these values: 0 = measured value is within range 2 = upper limit has been exceeded (overflow) 3 = lower limit not reached (underflow) 4 = measured value is invalid
541	read	0x04	**Value in percent as a signed integer x10**[5]
542, 543	read	0x04	**Value in percent in float format**
Sensor B – dew point			
560	read	0x04	**Value status** Contains value status. It can have these values: 0 = measured value is within range 2 = upper limit has been exceeded (overflow) 3 = lower limit not reached (underflow) 4 = measured value is invalid
561	read	0x04	**Value in degrees Celsius as a signed integer x10**[8]
562, 563	read	0x04	**Value in degrees Celsius as a float**
564	read	0x04	**Value in Fahrenheits as a signed integer x10**[5]
565, 566	read	0x04	**Value in Fahrenheits as a float**
567	read	0x04	**Value in Kelvins as a signed integer x10**[5]
568, 569	Read	0x04	**Value in Kelvins as a float**
Sensor B – CO_2 concentration			
580	read	0x04	**Value status** Contains value status. It can have these values: 0 = measured value is within range 2 = upper limit has been exceeded (overflow) 3 = lower limit not reached (underflow) 4 = measured value is invalid
581	read	0x04	**Value in percent as a signed integer x10**[5]
582, 583	read	0x04	**Value in percent in float format**

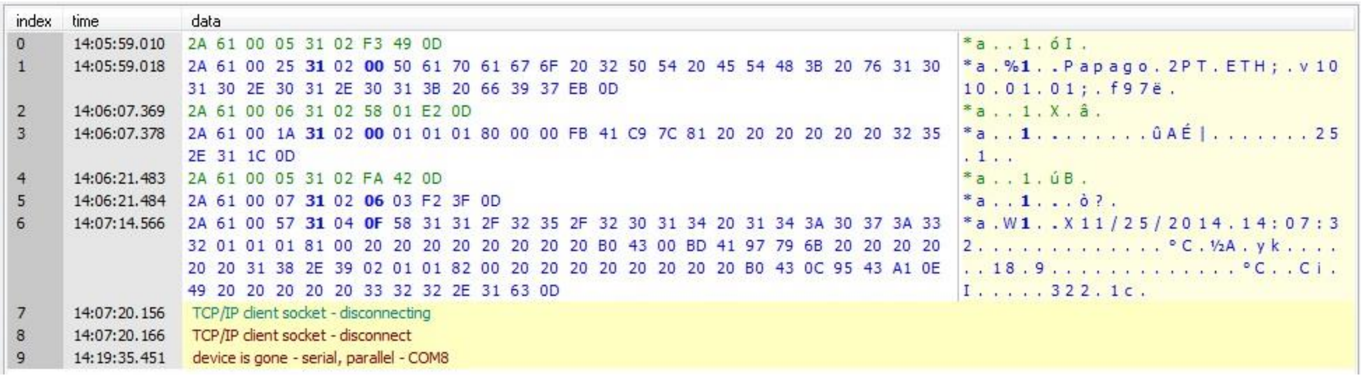
[8] The actual value can be calculated by dividing the number by 10. For example, 123 is 12.3

Address	*Access*	*Function*	*Name*
Sensor B – atmospheric pressure			
600	read	0x04	**Value status** Contains value status. It can have these values: 0 = measured value is within range 2 = upper limit has been exceeded (overflow) 3 = lower limit not reached (underflow) 4 = measured value is invalid
601	read	0x04	**Value in hPa as a signed integer x10**[5]
602, 603	read	0x04	**Value in hPa in float format**
604	read	0x04	**Value in bar as a signed integer x10**[5]
605, 606	read	0x04	**Value in bar in float format**
Sensor C – head			
1000	Read	0x04	**Sensor type** Sensor type is one of these codes: • 0 – not set • 6 – Davis (wind speed and direction)
1001	Read	0x04	**Status** Contains sensor status. It can have these values: 0 = sensor is connected 1 = sensor is disconnected
Sensor C – wind direction			
1140	read	0x04	**Value status** Contains value status. It can have these values: 0 = measured value is within range 2 = upper limit has been exceeded (overflow) 3 = lower limit not reached (underflow) 4 = measured value is invalid
1141	read	0x04	**Wind direction in degrees as a signed integer x10**[5]
1142, 1143	read	0x04	**Wind direction in degrees in float format**
1144	read	0x04	**Wind direction index** Number 1 to 16 representing one of these states: N (1), NNE, NE, ENE, E, ESE, SE, SSE, S, SSW, SW, WSW, W, WNW, NW, NNW (16)
Sensor C – wind speed			
1160	read	0x04	**Value status** Contains value status. It can have these values: 0 = measured value is within range 2 = upper limit has been exceeded (overflow) 3 = lower limit not reached (underflow) 4 = measured value is invalid
1161	read	0x04	**Wind speed in meters per second as a signed integer x10**[5]
1162, 1163	read	0x04	**Wind speed in meters per second in float format**

Address	Access	Function	Name
1164	read	0x04	**Wind speed in kilometers as a signed integer x10**[5]
1165, 1166	read	0x04	**Wind speed in kilometers in float format**

SPINEL COMMUNICATION PROTOCOL

The device has Spinel protocol implemented in format 97 (binary).



```
index time          data
0     14:05:59.010  2A 61 00 05 31 02 F3 49 0D                                            *a..1.óI.
1     14:05:59.018  2A 61 00 25 31 02 00 50 61 70 61 67 6F 20 32 50 54 20 45 54 48 3B 20 76 31 30  *a.%1..Papago.2PT.ETH;.v10
                    31 30 2E 30 31 2E 30 31 3B 20 66 39 37 EB 0D                          10.01.01;.f97ë.
2     14:06:07.369  2A 61 00 06 31 02 58 01 E2 0D                                         *a..1.X.â.
3     14:06:07.378  2A 61 00 1A 31 02 00 01 01 01 80 00 00 FB 41 C9 7C 81 20 20 20 20 20 20 32 35  *a..1........ûAÉ|.......25
                    2E 31 1C 0D                                                           .1..
4     14:06:21.483  2A 61 00 05 31 02 FA 42 0D                                            *a..1.úB.
5     14:06:21.484  2A 61 00 07 31 02 06 03 F2 3F 0D                                      *a..1...ò?.
6     14:07:14.566  2A 61 00 57 31 04 0F 58 31 31 2F 32 35 2F 32 30 31 34 20 31 34 3A 30 37 3A 33  *a.W1..X11/25/2014.14:07:3
                    32 01 01 01 81 00 20 20 20 20 20 20 20 20 B0 43 00 BD 41 97 79 6B 20 20 20 20  2.............°C.½A.yk....
                    20 20 31 38 2E 39 02 01 01 82 00 20 20 20 20 20 20 20 B0 43 0C 95 43 A1 0E  ..18.9............°C..Ci.
                    49 20 20 20 20 20 33 32 32 2E 31 63 0D                                I.....322.1c.
7     14:07:20.156  TCP/IP client socket - disconnecting
8     14:07:20.166  TCP/IP client socket - disconnect
9     14:19:35.451  device is gone - serial, parallel - COM8
```

fig. 7 - ukázka komunikace se zařízením v programu Spinel terminál

Format 97

Structure

Request:

```
PRE FRM NUM NUM ADR SIG INST DATA… SUMA CR
```

Response:

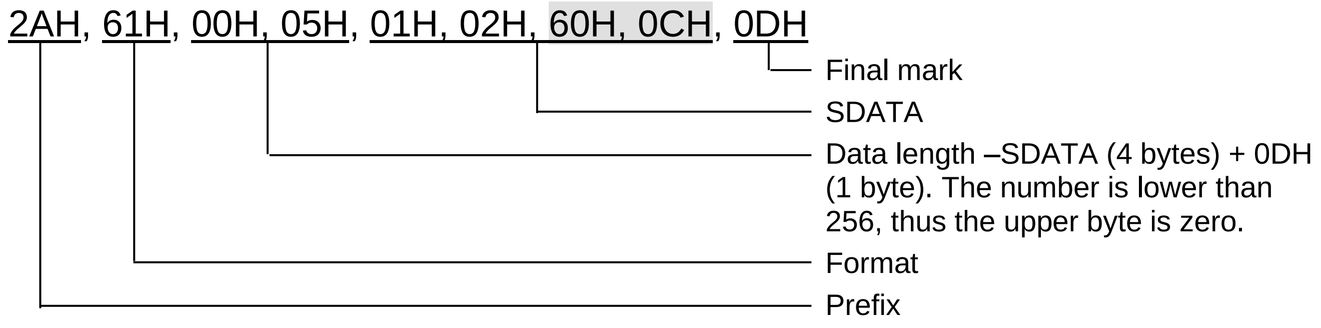
```
PRE FRM NUM NUM ADR SIG ACK  DATA… SUMA CR
```

PRE	Prefix, 2AH (“*“ sign).
FRM	Number of 97 format (61H).
NUM	Number of instruction bytes from the following bit to the end of the frame.
ADR	Address of the module to which the request is being sent or which is responding to it.
SIG	Message description – any number form 00H to FFH. The same number, which was sent in the request, is returned in the response, which makes it easy to see which request the response belongs to.
INST[9]	Instruction code.
ACK	Request acknowledgement of whether and how it was executed. ACK can be 00H to 0FH.
DATA[9]	Data.
SUMA	Check sum.
CR	Final mark (0DH).

[9] For easy orientation the instructions and data in the examples of following pages are highlighted this way.

Explanatory notes

Example



2AH, 61H, 00H, 05H, 01H, 02H, 60H, 0CH, 0DH

Final mark

SDATA

Data length –SDATA (4 bytes) + 0DH (1 byte). The number is lower than 256, thus the upper byte is zero.

Format

Prefix

Data Length (NUM)

Sixteen-bit value defining the number of bytes until the end of the instruction; number of all bytes found after NUM up to CR (including). It takes the values from 5 to 65535. If lower than 5, the instruction is considered faulty and it is answered (if intended for the relevant device) with ACK “Invalid Data” instruction.

Process of NUM creation:

Ad up the number of bytes after both NUM bytes (i.e. the number of SDATA bytes + 1 CR byte). The resulting sum view as a sixteen-bit. Divide it into the upper and lower byte. The first NUM byte id the upper byte of the number, the second NUM byte is the lower byte of the number. (If the number of bytes is lower than 256, the first NUM byte is 00H.)

Address (ADR)

The FFH address is reserved for broadcast. If the request contains the FFH address, the device operates as if its own address is entered. No response is sent to enquiries with this address.

The FEH address in the universal address. If the request contains the FEH address, the device operates as if its own address is entered. The device enters real, currently set address into the response. The universal address is used in cases where only one device is connected on the line.

Request Acknowledgement (ACK)

ACK informs the superior device on the way of the received instruction processing. Acknowledgement codes:

00H ....EVERYTHING OK
The instruction was properly received and completely executed.

01H ....ANOTHER ERROR
Unspecified device error.

02H ....INVALID CODE OF INSTRUCTION
The received instruction code is unknown.

03H ....INVALID DATA
Data are of invalid length or contain invalid value.

04H ....ENTRY NOT ALLOWED/ACCESS REFUSED
- The request was not performed, as some conditions had not been fulfilled.
- Attempt to enter data into inaccessible memory.
- Attempt to activate a device function requiring a different configuration (e.g. higher communication speed).
- Attempt to change configuration without immediately preceding setup acknowledgement.
- Access into memory protected by a password.

05H ....DEVICE FAILURE
- Device failure requiring service action.
- Device internal memory error or setup memory error.
- Device internal error (operation error or start-up error).
- Any other error affecting the device proper functioning.

06H ....NO DATA AVAILABLE

0EH.......INSTRUCTION SENT AUTOMATICALLY – CONTINUOUS MEASURING
- recurring transfer of measured values.

Check Sum (SUMA)

Sum of all instruction bytes (sum of all transferred data except CR) subtracted from 255.

Calculation: SUMA = 255 – (PRE + FRM + NUM + ADR + SIG + ACK (INST) + DATA)

No response is made to messages with faulty check sum. (The system waits for the receipt of CR even if a faulty check sum is received.)

Spinel instructions

Read values from input register

Instruction reads any part of input register (see page 11). First register and number of registers has to be entered to be read. Any measured value or sensor type can be obtained.

Request:

Instruction code: 41H

Parameters: (address)(number-of-registers)

address	Register address	length: 2 bytes
Address of the first input register to be read. Enter the address of any register from address table on page 11.		

number	Number of registers	length: 2 bytes
Number of registers to be read. Enter number from 1 to 127.		

Response:

Acknowledge code: ACK 00H

Parameters: {(register)}

register	One input register	length: 2 bytes
One or more 2-byte valuer from input register based on what number of registers was entered (number-of-registers).		

Examples:

Request – read from register 20 (0014H), 10 values (000AH), meaning all information about temperature from sensor A:
`2AH,61H,00H,09H,31H,02H,41H,00H,14H,00H,0AH,D9H,0DH`
Response:
`2AH,61H,00H,19H,31H,02H,00H,00H,80H,00H,FEH,41H,CBH,33H,33H,03H,09H,42H,9BH,70H,A4H,0BH,A9H,43H,95H,46H,66H,03H,0DH`
Value 00FEH represents number 254, which means temperature of 25,4 °C. In the following parts the temperature is represented in other formats.

Reading of name and version

Reads the name of the device, software version and the list of possible communication formats. Set by the manufacturer.

Request:

Instruction code: F3H

Response:

Acknowledgement code: ACK 00H

Parameters: (string)

string	Name and version	length: 1 byte
Papago METEO RS; v1379.01.01; H3 T A In addition to the information described above, the string can also contain other information in sections introduced by a semicolon, space and a small letter to determine which information follows.		

Examples:

Request:
`2AH,61H,00H,05H,31H,02H,F3H,49H,0DH`

Reading of manufacturing data

This instruction reads the manufacturing data of the device.

Request:

Instruction code: FAH

Response:

Acknowledgement code: ACK 00H

Parameters: (product_number)(serial_number)(other)

product_number	length: 2 bytes
Product number. For a device number 0227.00.03/0001 this number is 227.	

serial_number	length: 2 bytes
Serial number. For a device number 0227.00.03/0001 this number is 1.	

other	length: 4 bytes
Other manufacturing information.	

Examples:

Request:
`2AH,61H,00H,05H,FEH,02H,FAH,75H,0DH`

INDICATIONS

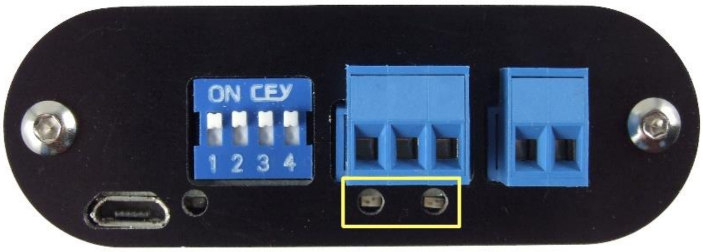


fig. 8 – there are two indicators in the yellow frame

Red-green (left):

- the green light is lit and the red light flashes when the device is working properly and is connected to at least one sensor
- the green and red LEDs are lit when the device works, but is not connected to any sensor
- the red LED is lit to indicate an error

Yellow (right): Flashes when communicating via RS485.

TECHNICAL PARAMETERS

Integrated temperature and humidity sensor TH3[10]

Important Notice: Polymer sensor is a highly sensitive element that reacts with chemicals. Do not expose even the outer shell of the sensor to chemicals or their vapors (cleaning with alcohol, petrol etc.). Especially organic solvents and compounds can negatively affect the sensor accuracy by as tens of percent RH.

Coverage..IP 54

Dimensions...40 × 16 × 10 mm

Material..hardened aluminum

Humidity sensor

Humidity range ....0 % to 100 % RH

Recommended measurement range ....20 – 80 %

Resolution..1% RH

Humidity measurement accuracy ....see fig. 9

Sensor element ....polymer sensor

Sensor mechanical finish...............................inside hardened aluminum block

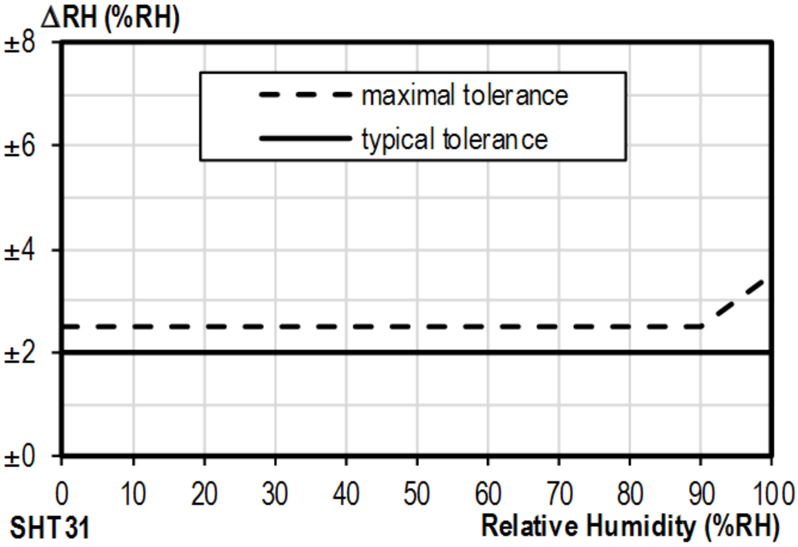


fig. 9 – Accuracy of humidity measuring

[10] Sensor marked TH3 is supported in firmware including and above version 1.05. If you have an older firmware, you will have to flash the firmware to be able to read from TH3 sensor. Here are the key differences between the old version (Marked as TH2E) and TH3 version:

	TH3 (new sensor)	**TH2E (old sensor)**
Measurement accuracy within 0 – 10 %	±2 %	±2 to ±4 %
Measurement accuracy within 90 – 100 %	±2 %	±2 to ±4 %
Recommended measurement range	20 – 80 % RH	
Temperature measurement range	-40.0 °C to +125.0 °C	-40.0 °C to +123.8 °C
Temperature measurement accuracy	±0.3 to ±0.5 °C	±0.4 to ±2.0 °C

Operating and Maximum Range of Values

- Sensor is stable in standard range of humidity values. Long-term exposure to conditions outside these values (humidity above 80% in particular) can temporarily shift the measured-out values (by +3% for 60 hours). When the sensor is back to standard ranges, it returns to its pre-calibrated state slowly.[11]
- Long-term exposure to extreme conditions or to chemically aggressive vapor can speed up the aging process of the sensor significantly. It can also shift the measurements.

Temperature sensor

Range	-40.0 °C to +125 °C
Resolution	0.1 °C
Sensor element	semiconductor
Sensor mechanical finish	inside hardened aluminum block

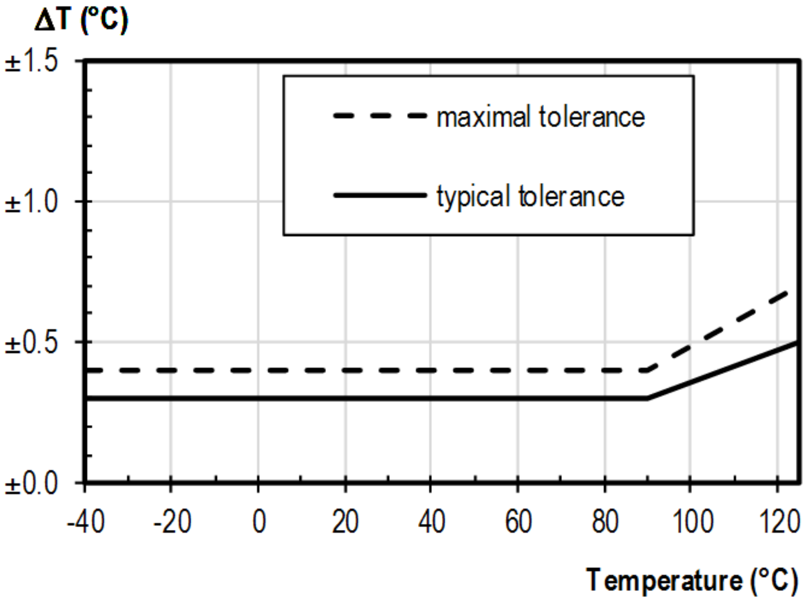


fig. 10 - Accuracy of temperature measurement

Integrated pressure, temperature and humidity sensor

The design and measuring ranges of the sensor are identical to the TH3 sensor. In addition, the sensor measures atmospheric pressure with the following parameters:

Measured atmospheric pressure range	50 to 110 kPa
Accuracy	±0,4 kPa

Standalone temperature sensor

Sensor type	semiconductor
Measuring temperature range	-55 °C to +125 °C

[11] You can speed up this process by doing following:
1) Leave the sensor in environment above 100 to 105 °C and humidity below 5 % for at least 10 hours.
2) Leave the sensor in environment above 20 to 30 °C and humidity approximately 75 % for around 12 hours. (Humidity 75% can be achieved with saturated solution of NaCl.)

Accuracy	±0.5 °C in the range of -10 °C to +85 °C
Temperature drift	±0.2 °C per 1000 hours at 125 °C
Dimensions	normalized diameter 6 mm, length 60 mm
Housing material	hardened alloy
Degree of protection	IP68 (permanent immersion into 1m max.)

Sensor cable

Cable jacket	silicone rubber, blue
Wire insulation	FEP polymer
Length	standard 3 m (optionally up to 20 meters)
Measuring temperature range	-60 °C to +200 °C
Maximum allowable temperature	+220 °C
Cable diameter	4.3 mm (±0.1 mm)

The cable shows excellent resistance to moisture, chemicals and carbohydrates.

CO_2 concentration sensor

Range	0 to 2000 ppm[12]
Type of sensor	NDIR (nondispersive infrared sensor)
Accuracy within 400 to 2000 ppm	± 25 ppm, ± 3% of measured value
Temperature dependency	5 ppm on °C or 0.5% of value on °C (whichever is higher)
Settle time upon change	max. 3 min to 90 %
Settle time after power-up	max. 10 minutes to 100 %
Op. temperature range	-10 to +60 °C

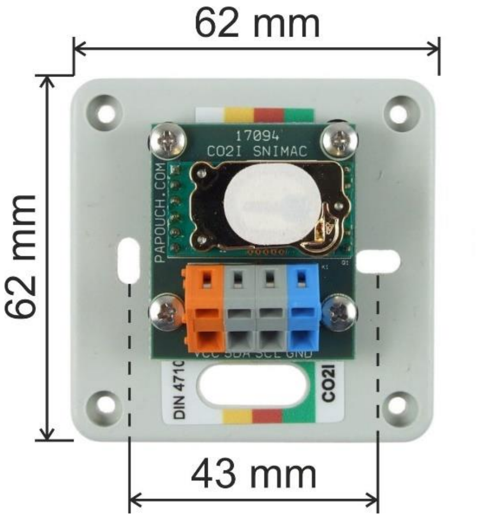


fig. 11 – sensor dimensions

[12] If the sensor is in an environment with below 400 ppm concentration over 15 minutes, it can affect the overall accuracy of the sensor.

Dimensions	see picture above, enclosure height is 29 mm
Mount hole diameter	4 mm
Cable connection	Wago 236 terminal
IP coverage	IP 20
Cable length	3 m, 10 m or custom length

Wind speed and direction sensor

Type	Davis 6410
Operation temperature	-40 to +65 °C
Wind direction resolution	16 steps (22.5°)
Direction accuracy	±3°
Wind speed range	0.5 to 89 m/s
Wind speed accuracy	±1 m/s or ±5 % (whichever is higher)
Lead cable length	12.2 m (extendable)

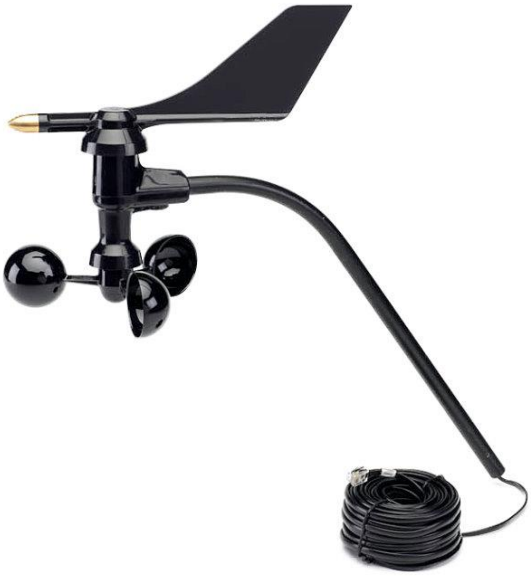


fig. 12 – wind sensor

Other parameters

Port RS485

Connector	slip-on terminal
Overvoltage protection	transil 6.5 V on RS485 (against SGND)
Communication speed	110 Bd to 230,4 kBd (default: 9.6 kBd)
Data bits	8
Parity	no parity, even or odd
Stop bits	1 or 2
Communication protocols	Modbus RTU (default) and Spinel

Default address	49

USB interface

Specifications	USB 1.1 HID (2.0, 3.0 compatible)
Connector	micro USB B
Use	configuration

Device electronics

Power voltage	11 to 58 V DC (with reverse polarity protection)
Current consumption at 12 V	typ. 45 mA
Current consumption at 24 V	typ. 26 mA
Power supply connector	slip-on terminal
Operating temperature range	-20 to +70 °C
Dimensions (without connectors)	88 × 70 × 25 mm

Other parameters

Housing material	anodized aluminum
Degree of protection	IP 30
Weight	typ. 130 g

Available designs

Mountable on 35 mm DIN rail	optional accessory

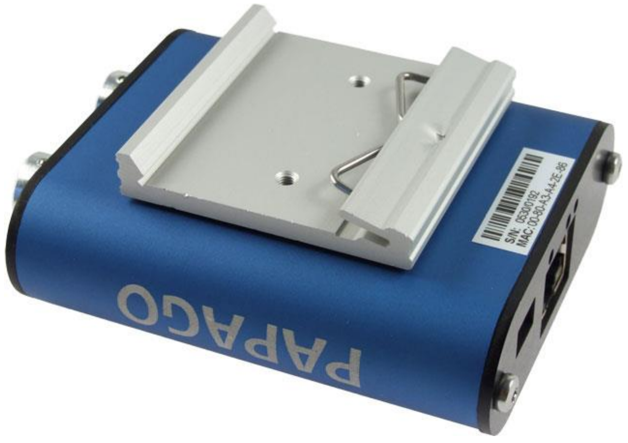


fig. 13 – Papago 2TH ETH with DIN rail holder

Do not hesitate to contact us if you have any other requirements concerning the design and functions of PAPAGO METEO RS.

Papouch s.r.o.

Data transmission in industry, line and protocol conversions, RS232/485/422/USB/Ethernet/GPRS/ WiFi, measurement modules, intelligent temperature sensors, I/O modules, and custom-made electronic applications.

Address:

Strasnicka 3164
102 00 Prague 10
Czech Republic

Tel:

+420 267 314 267

Internet:

www.papouch.com

E-mail:

info@papouch.com