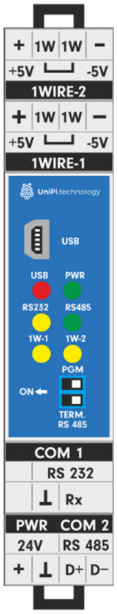
# GATEWAY 1-WIRE (DALLAS) through RS 485 Modbus RTU

# xG1W-2

**Gateway 1-WIRE (DALLAS) for measuring temperatures, humidity, with i-button keys and communication RS 485 MODBUS RTU**




- **Connection up to 40 temperature sensors with measuring humidity, i-button keys through 2 buses with communication 1-WIRE (DALLAS). 1st bus = 20 temperatures + humidity or i-button keys.**
- **Easy and variable solution for measuring in object, in technology, remote measurement with bus.**
- **Unbeatable savings of financial costs for cables compared to other solutions: 20 sensors on one bus (totally 40 sensors/1 unit).**
- **Easy installation into control cabinet.**
- **Interface RS 485, RS 232, USB**
- **Complete management through the application 1-WIRE-GWY Tool, baud rate and address settings, sensors addressing on positions, values displaying, firmware upgrade and other necessary functions.**
- **Software support = library elements are ready (programs) for control systems of different producers.**

## TECHNICAL DATA

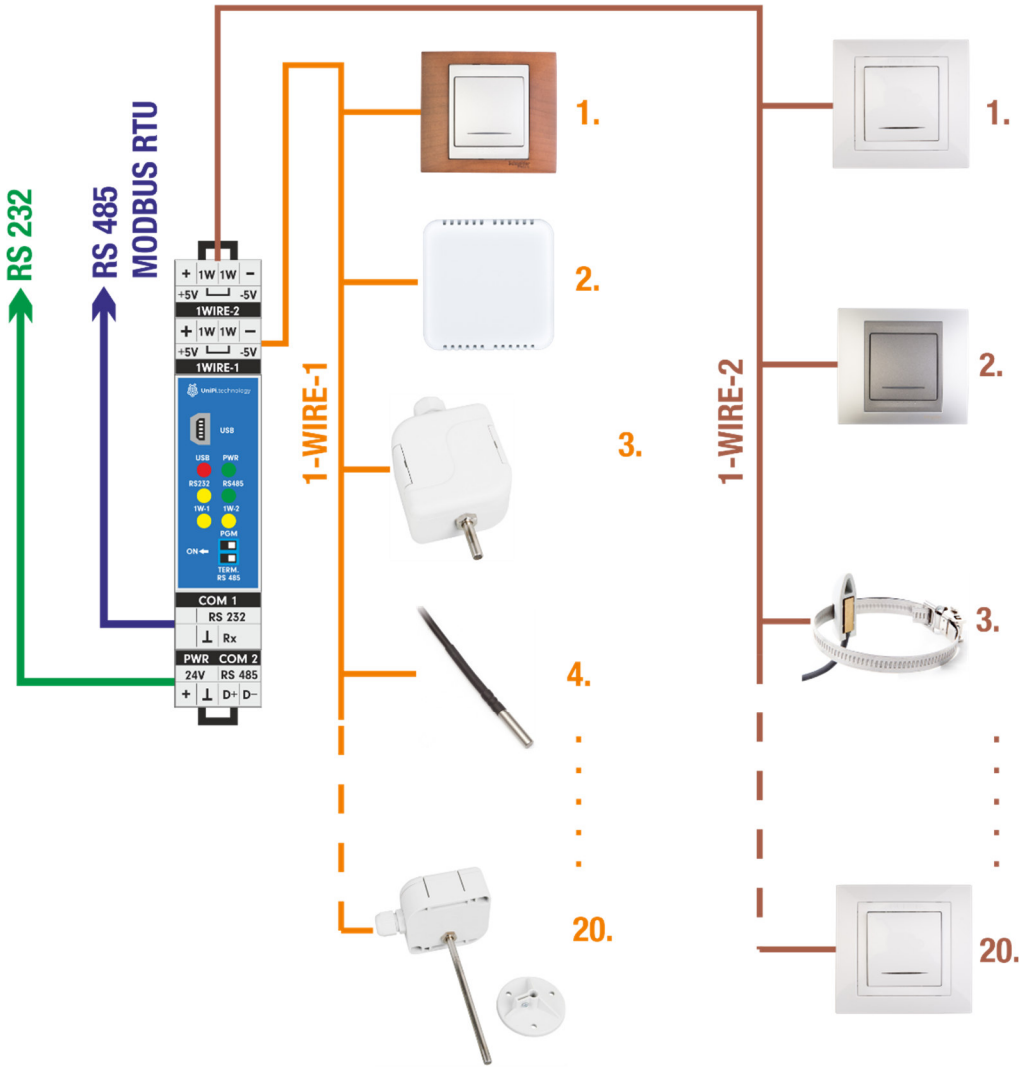
| | | |
|---|---|---|
| **Electrical data** | power supply range | 8-27/DC (tolerance +/- 10%) |
| | own device consumption | 1,5W |
| | indication | yes, green LED diode on front panel of the device |
| **Communication I.** | type | RS 485 (TIA/EIA-485-A), RS-232 |
| | protocol | MODBUS RTU, slave, supported function 03, 06, 16 |
| | baud rate for RS 485 and RS 232 | optional (kBd) 1200, 1800, 2400, 4800, 7200, 9600, 14400, 19200, 28800, 38400, 56000, 57600, 76800, 115200, 128000, 230400 |
| | address | 1-247 |
| | No. of devices on the bus RS 485 | 32 |
| | parita | no, even, odd |
| | stopbit | 1,2 |
| | galvanic isolation from power supply | yes |
| | indication | yes, yellow LED diode on front panel of the device |
| **Communication II.** | type | 1-WIRE (DALLAS) |
| | protocol | 1-WIRE (DALLAS) |
| | No. of temperature sensors on 1 bus | 20 |
| | No. of buses (lines) | 2 |
| | galvanic isolation from power supply | yes |
| | Indication of bus state | yes LED |
| | cover | IP20 |
| | operating temperature | 0-40 °C |

| | | |
|---|---|---|
| **Operating values** | relative air humidity | max. 80 % |
| | external dimension (h x w x d) | 98 x 17,5 x 56,4mm |
| **USB** | type | USB – pro service purposes |
| | protocol | MODBUS RTU, slave, supported functions 03, 06, 16 |
| | baud rate | 115 200 bps |
| | address | 1 |
| | parity | no |
| | stopbit | 1 |
| | indication | yes, yellow LED diode on front panel of the device |

## DESCRIPTION OF DEVICE FUNCTIONS

Gateway 1-WIRE-GWY-MOD processes data from temperature sensors or i-button keys with communication 1-WIRE (DALLAS) on its two data buses. Each bus can serve up to 20 temperature sensors with information about humidity (totally 40) and send their values including faulty states through the interfaces RS 485, RS 232 with protocol MODBUS RTU. The gateway is configured through the application 1-WIRE-GWY-TOOL and USB cable for easy and simple management of sensors positions and all necessary settings. LED indicators on front panel indicate power of device, communication on RS 485, RS 232 and presence of temperature sensor for each bus separately.

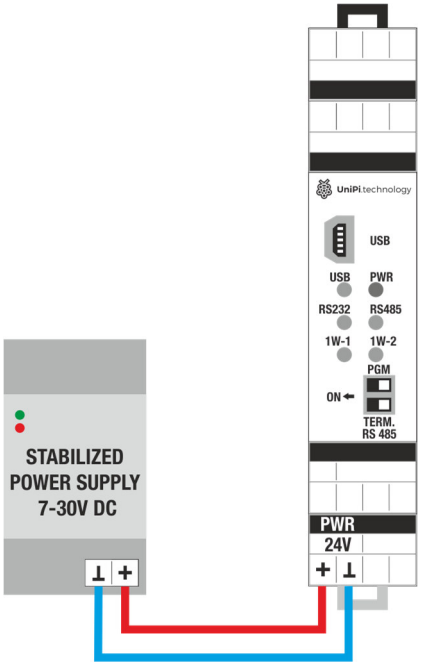
## EXAMPLE FOR CONNECTION OF THE INTERFACES




*Pic. Connection example of the interfaces*

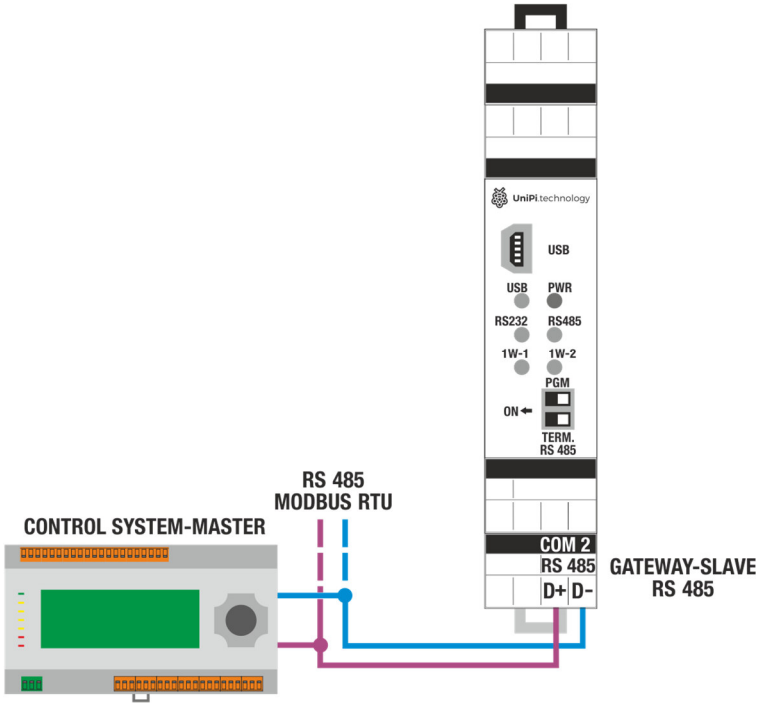
## CONNECTION OF POWER SUPPLY 24V, DC

Power supply of the device is 8-27V (DC) with tolerance 10%.

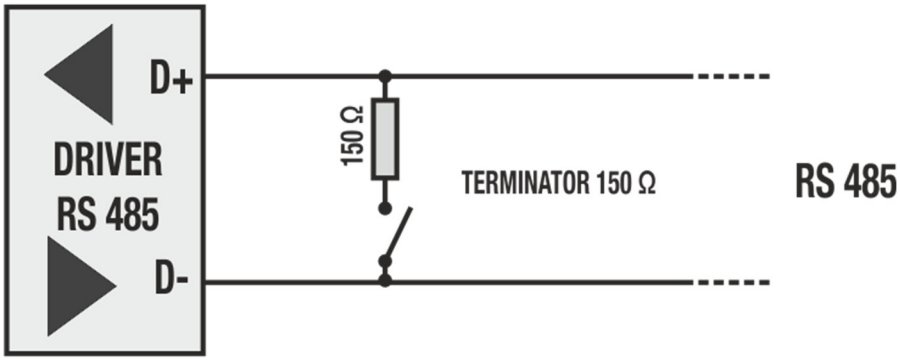



## CONNECTION OF COMMUNICATION RS 485

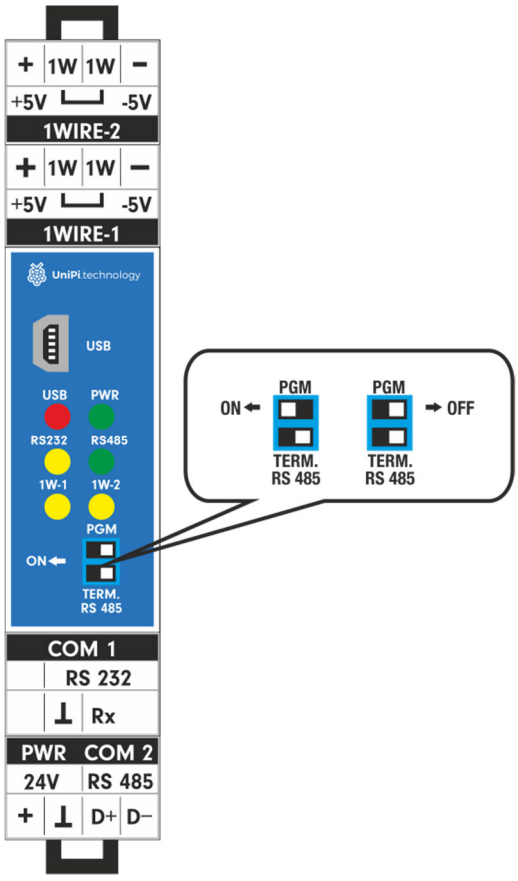
Gateway can communicate through buses RS 485 and RS 232 at the same time.



## LOAD OF THE BUS RS 485 (TERMINATOR) BLOCK SCHEMA:

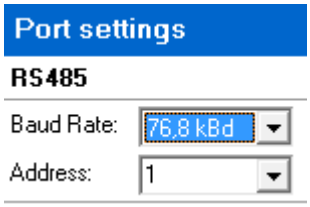



## LOAD OF THE BUS RS 485 (TERMINATOR) SELECTION ON/OFF:

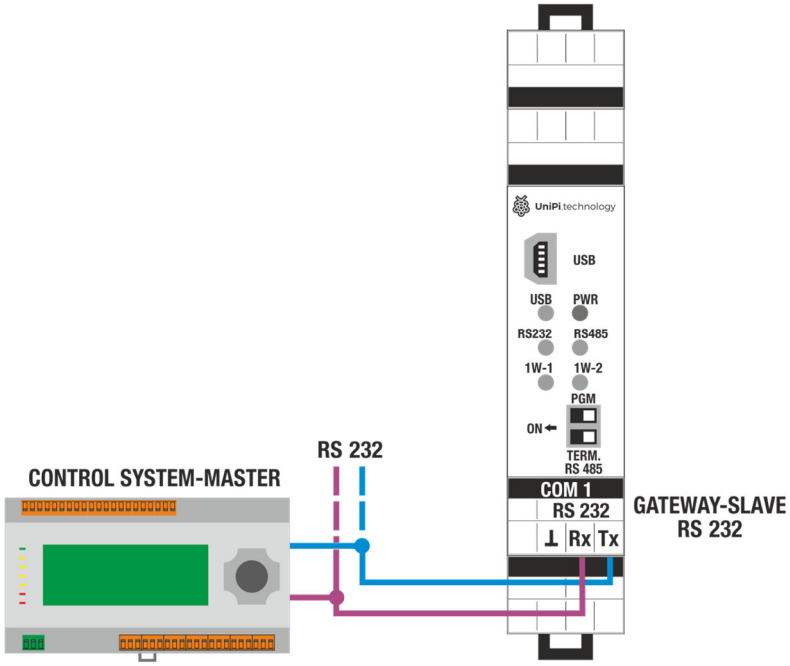


## BAUD RATE SETTINGS OF THE COMMUNICATION RS 485:

Baud rate settings of communication and address for RS 485 is made in the application 1-WIRE-GWY Tool in port settings:



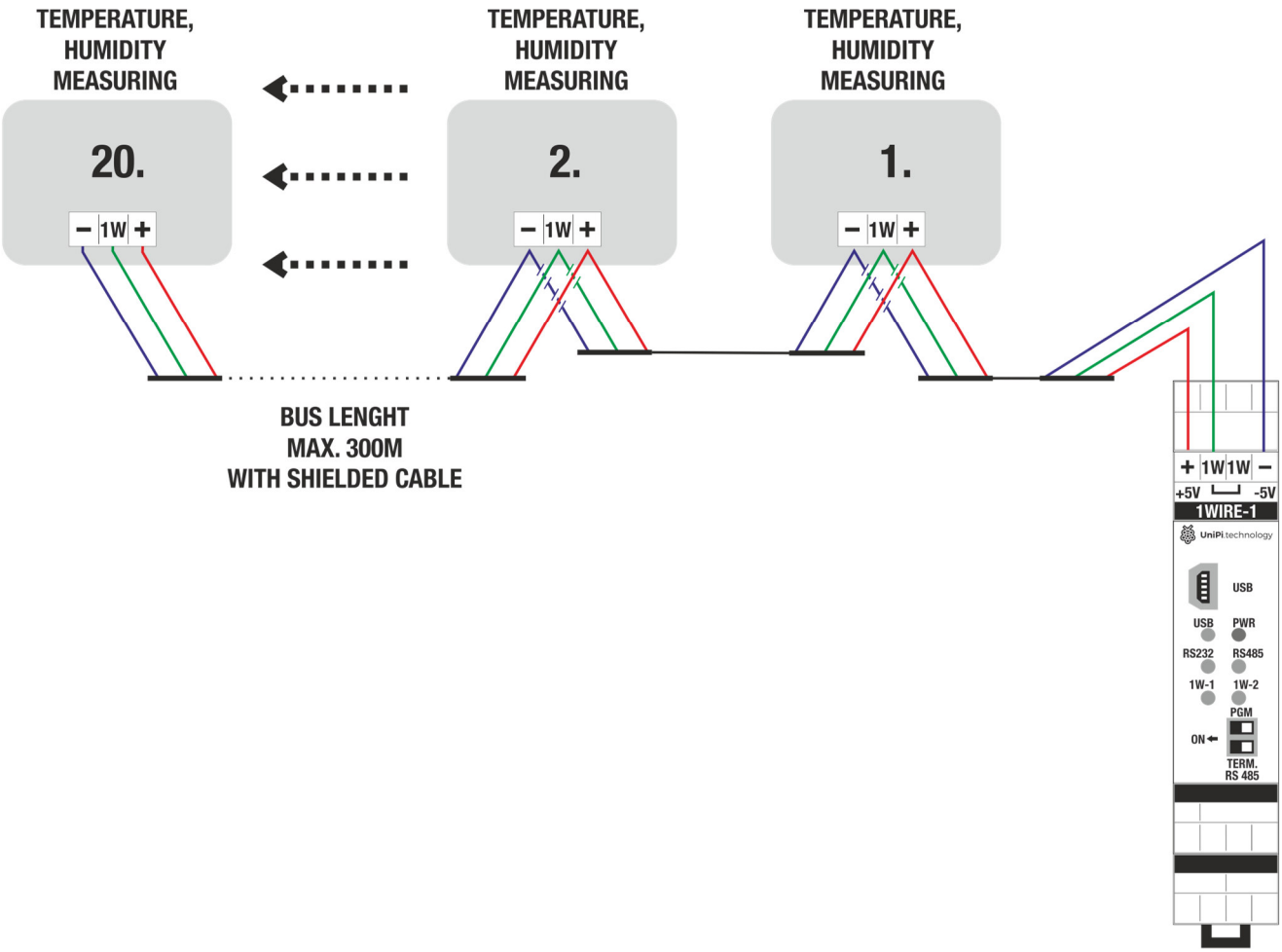

## CONNECTION OF COMMUNICATION RS 232



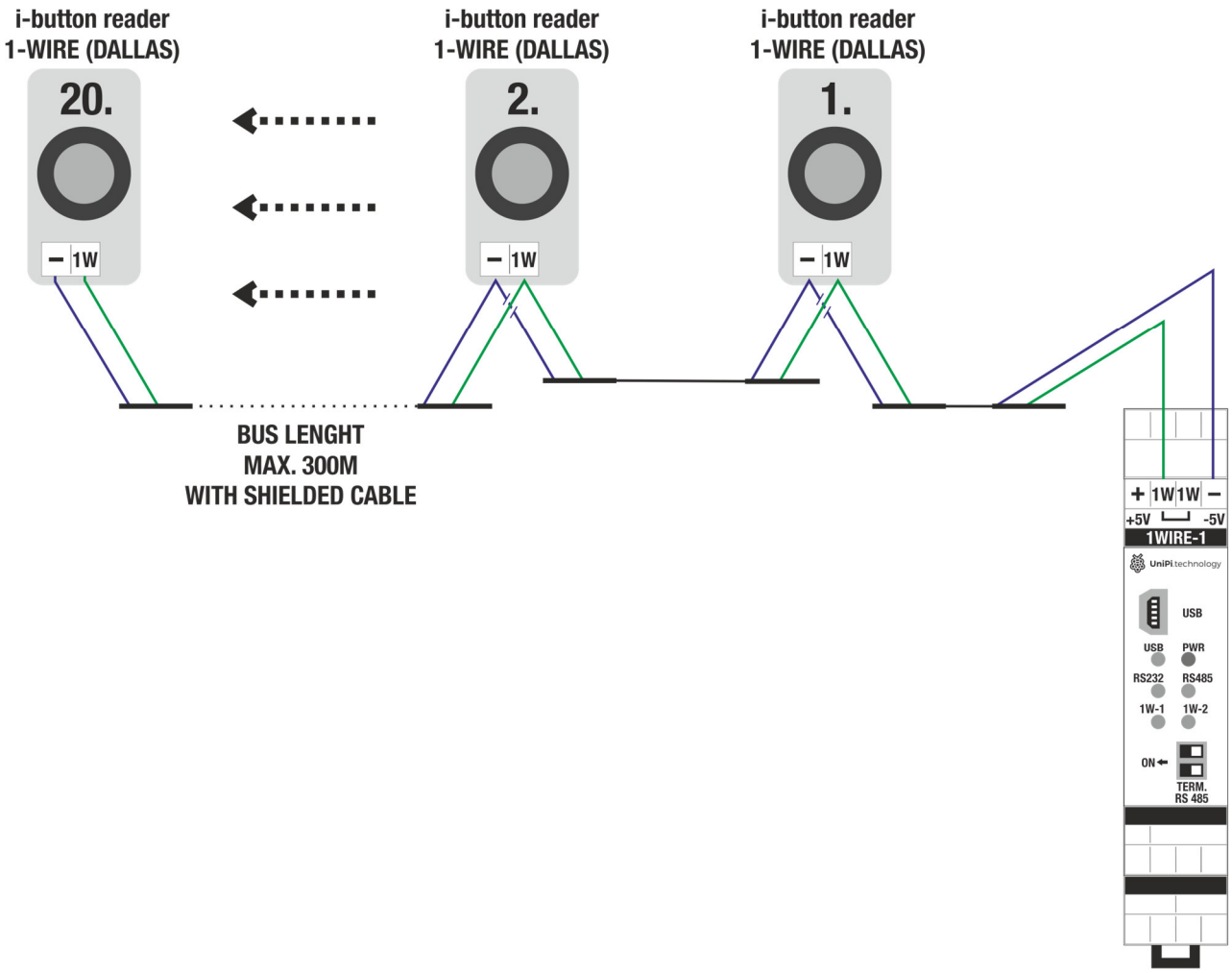

## BAUD RATE SETTINGS OF COMMUNICATION RS 232:

Baud rate settings of communication and address for RS 232 is made in the application 1-WIRE-GWY Tool in port settings:

## EXAMPLE FOR CONNECTION OF SENSORS ON 1-WIRE



## EXAMPLE FOR CONNECTION OF 1-WIRE I-BUTTON READER




**Notice for i-button keys:**
Continuous communication runs between gateway and sensors on buses 1-WIRE-1 and 1-WIRE-2 in case of using 1-WIRE sensors. On the contrary in case of using i-button keys, the i-button reader is used as a reading medium and it waits for attaching an i-button key and after that the communication runs and gateway performes the instruction.

In case of connection of more i-button readers on one bus by the gateway and a user attaches i-button keys by more i-button readers at the same time, so the communication from the readers will overlap.

It is appropriate to think if such a situation may happen. The i-button readers can be divided in to both buses or the application can be extended with one another gateway.

**Important notice:**
It is not able to combine temperatures with i-button readers on one bus (1-WIRE-1 or 1-WIRE-2). The technical combination of temperature sensors and i-button keys is not possible! We recommend the division to the buses 1-WIRE-1 or 1-WIRE-2.

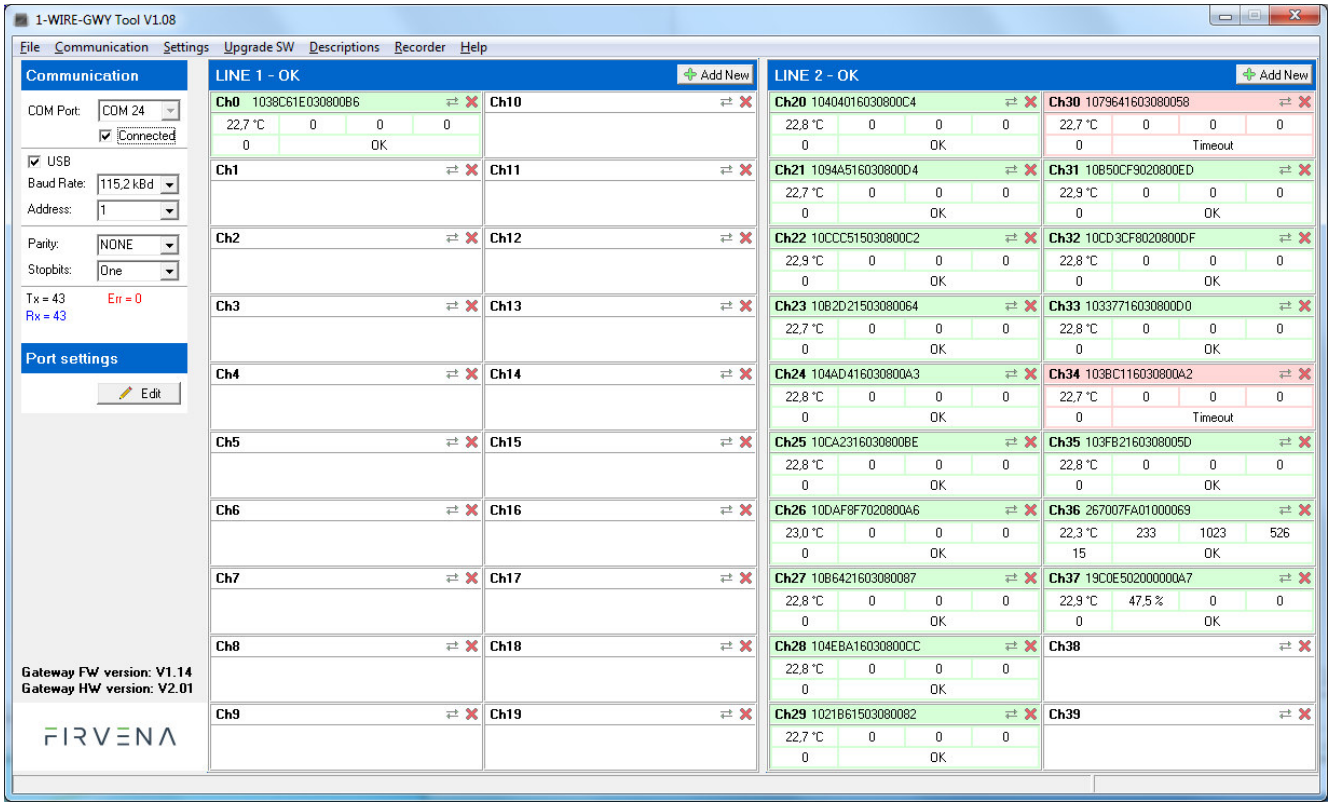
**Reading the i-button key on position in gateway:**
When reading the i-button key on position in gateway, the i-button key must be attached to the i-button reader. The i-button reader doesnt have identification!

## CONFIGURATION AND GATEWAY SETTINGS

Gateway settings, all necessary management is solved with the application 1-WIRE-GWY-Tool.

The application ensures:

- **Communication through USB**
- **Configuration of baud rate and address for RS 232, RS 485**
- **Easy adding and removing on/from position of the 1-WIRE humidity sensor**
- **Easy adding and removing on/from position of the 1-WIRE i-button key**
- **The option to look up other unknown sensors and adding on position**
- **Comprehensive overview of all measured values, states, errors**
- **Upgrade of gateway‘s firmware**




## UPGRADE OF FIRMWARE:

It is described in separate document.

## 1. Register MAP

| | No | Description | |
|---|---|---|---|
| Device 1 | 0 | ROM code1 (family code, serial number) | |
| | 1 | ROM code2 (serial number) | |
| | 2 | ROM code3 (serial number) | |
| | 3 | ROM code4 (serial number, CRC) | |
| | 4 | Value 1 (Temperature in °C) | |
| | 5 | Value 2 | |
| | 6 | Value 3 | |
| | 7 | Value 4 | |
| | 8 | Value 5 | |
| | 9 | Error | |
| Device 2 | 10 | ROM code1 (family code, serial number) | |
| | 11 | ROM code2 (serial number) | |
| | 12 | ROM code3 (serial number) | |
| | 13 | ROM code4 (serial number, CRC) | |
| | 14 | Value 1 (Temperature in °C) | |
| | 15 | Value 2 | |
| | 16 | Value 3 | |
| | 17 | Value 4 | |
| | 18 | Value 5 | |
| | 19 | Error | |
| | … | | |
| Device 40 | 390 | ROM code1 (family code, serial number) | |
| | 391 | ROM code2 (serial number) | |
| | 392 | ROM code3 (serial number) | |
| | 393 | ROM code4 (serial number, CRC) | |
| | 394 | Value 1 (Temperature in °C) | |
| | 395 | Value 2 | |
| | 396 | Value 3 | |
| | 397 | Value 4 | |
| | 398 | Value 5 | |
| | 399 | Error | |
| | … | | |
| Service registers | 1000 | SW Version | |
| | 1001 | MODBUS address | PORT RS485 |
| | 1002 | Baud rate (1200,..115200) | |
| | 1003 | MODBUS address | PORT RS232 |
| | 1004 | Baud rate (1200,..115200) | |
| | 1005 | Stopbit 1, 2 | PORT RS485 |
| | 1006 | Parity 0- none, 1 - ODD, 2-EVEN | |
| | 1007 | Stopbit 1, 2 | PORT RS232 |
| | 1008 | Parity 0- none, 1 - ODD, 2-EVEN | |
| | 1009 | HW Version | |
| | 1010 | **Command** | LINE 1 |
| | 1011 | **Status** | |
| | 1012 | NEW ROM code1 (family code, serial number) | |
| | 1013 | NEW ROM code2 (serial number) | |
| | 1014 | NEW ROM code3 (serial number) | |
| | 1015 | NEW ROM code4 (serial number, CRC) | |
| | … | | |
| | 1020 | **Command** | LINE2 |
| | 1021 | **Status** | |
| | 1022 | NEW ROM code1 (family code, serial number) | |
| | 1023 | NEW ROM code2 (serial number) | |
| | 1024 | NEW ROM code3 (serial number) | |
| | 1025 | NEW ROM code4 (serial number, CRC) | |

| | | | |
|---|---|---|---|
| | 1053 | Time delay 0 – 200ms | PORT RS485 |
| | 1054 | Time delay 0 – 200ms | PORT RS232 |
| | 1055 | Power 1Wire lines (0-OFF, 1- ON, 2,3, .. 5) (default 1-ON) | |
| | | | |

| | No | Description | |
|---|---|---|---|
| Service registers | 1100 | ROM code1 (family code, serial number) | LINE 1 |
| | 1101 | ROM code2 (serial number) | |
| | 1102 | ROM code3 (serial number) | |
| | 1103 | ROM code4 (serial number, CRC) | |
| | 1104 | Index 0 | |
| | 1105 | ROM code1 (family code, serial number) | |
| | 1106 | ROM code2 (serial number) | |
| | 1107 | ROM code3 (serial number) | |
| | 1108 | ROM code4 (serial number, CRC) | |
| | 1109 | Index 1 | |
| | … | | |
| | 1195 | ROM code1 (family code, serial number) | |
| | 1196 | ROM code2 (serial number) | |
| | 1197 | ROM code3 (serial number) | |
| | 1198 | ROM code4 (serial number, CRC) | |
| | 1199 | Index 19 | |
| | | | |
| | 1200 | ROM code1 (family code, serial number) | LINE 2 |
| | 1201 | ROM code2 (serial number) | |
| | 1202 | ROM code3 (serial number) | |
| | 1203 | ROM code4 (serial number, CRC) | |
| | 1204 | Index 0 | |
| | 1205 | ROM code1 (family code, serial number) | |
| | 1206 | ROM code2 (serial number) | |
| | 1207 | ROM code3 (serial number) | |
| | 1208 | ROM code4 (serial number, CRC) | |
| | 1209 | Index 1 | |
| | … | | |
| | 1295 | ROM code1 (family code, serial number) | |
| | 1296 | ROM code2 (serial number) | |
| | 1297 | ROM code3 (serial number) | |
| | 1298 | ROM code4 (serial number, CRC) | |
| | 1299 | Index 19 | |
| | … | | |
| | 5000 | Device 1 - Error counter (crc, timeout) 0-65000 ( write 0) | LINE 1 |
| | 5001 | Device 2 - Error counter (crc, timeout) 0-65000 | |
| | … | | |
| | 5019 | Device 20 - Error counter (crc, timeout) 0-65000 | |
| | 5020 | Device 21 - Error counter (crc, timeout) 0-65000 | LINE 2 |
| | 5021 | Device 22 - Error counter (crc, timeout) 0-65000 | |
| | … | | |
| | 5039 | Device 40 - Error counter (crc, timeout) 0-65000 | |

## 2. Description of registers

It is assigned 10 registers to each sensor. 4 registers with editable serial number, 5 with read data and 1 error.

### Registers with serial number

| Register No. | Higher byte | Lower byte | Note |
|---|---|---|---|
| n*10 + 0 | 8bit family code | serial number - 1 | |
| n*10 + 1 | serial number - 2 | serial number - 3 | |
| n*10 + 2 | serial number - 4 | serial number - 5 | |
| n*10 + 3 | serial number 6 | serial number - 7 | |
| n*10 + 4 | serial number - 8 | CRC | |

*n is number of channel (input)

### Data registers

| Register No. | Value | Note |
|---|---|---|
| n*10 + 5 | Temperature * 10 | 23,5°C - > 235 |
| n*10 + 6 | For DS2438 voltage napětí or 0 | UNICA module humidity - voltage |
| n*10 + 7 | For DS2438 current or 0 | UNICA module lighting ratio 0-1023( 0-100%) |
| n*10 + 8 | Approximate relative humidity | Approximate relative humidity % |
| n*10 + 9 | Status / Configuration | Page 0 MemMap DS2438 |

*n is number of channel (input)

iButtons

| Register No. | Value | Note |
|---|---|---|
| n*10 + 5 | 1 Presence of i-button key | 0,1 |
| n*10 + 6 | No. of i-button key connection | 0 - 65353 |
| n*10 + 7 | 0 | |
| n*10 + 8 | 0 | |
| n*10 + 9 | 1 | It holds state 20s after disconnection of i-button key |

*n is number of channel (input)

### Error register

| Register No. | Value | Note |
|---|---|---|
| n*10 + 9 | 1 - 255 | State of communication with sensor |

*n je číslo kanálu (vstupu)

| Error register | |
|---|---|
| Value | Meaning of value |
| 0 | Bus Ok |
| 1 | No sensor on bus (bus is interrupted) |
| 2 | Bus short-circuit |
| 3 | Type of sensor is not supported |
| 4 | Error CRC |
| 5 | Error in reply – the sensor is not connected |
| 255 | Not occupied position |

## 3. Description of service registers

Registers COMMAND and STATUS are available for each bus for editing field of registers.

Register „**Command**“. It is possible to modify register table with this row. row. First byte contains command, second one number of channel. Channels are number from zero.

| Register No. | | Register description |
|---|---|---|
| Bus 1 | Bus 2 | |
| 1010 | 1020 | Command |

Example:

| Command – meaning of values | | | |
|---|---|---|---|
| Value in register | Command HiByte | Command LoByte | Description of command |
| 0x0F02 | 0x0F | 0x02 | Delete sensor on position 2 |
| 0x05FF | 0x05 | FF= no | Search new ROM Code<br>Command is usable when one device only in on the bus |
| 0x06FF | 0x06 | FF=no | Search ROM Code of all uknown devices that are connected to bus |
| 0x07FF | 0x07 | FF=no | Search ROM Code of all devices that are connected to bus. Also the saved ones. |
| 0x8208 | 0x8..2 | 0x08 | Newly found ROM Code with index 2 will be saved on position 8* |
| 0x0902 | 0x09 | 0x02 | Save new ROM Code on position 2 |

*Attention, on second bus the position 20 on position 20 is 0x9414

Register „**Status**“. This register contains respond on „**Command**“.

| Register No. | | Register description |
|---|---|---|
| Bus 1 | Bus 2 | |
| 1011 | 1021 | Status |

- After finishing the command, the number 0xFFFF runs
- If an error occurs during command, number of error (0xEEE0…A) returns to register status.

| Status – meaning of values | |
|---|---|
| 0xFFFF | Command ran without error |
| 0xEEE1 | Number of channel is out of range for appropriate bus |
| 0xEEE2 | Number of command is not supported |
| 0xEEE3 | Error during reading ROM-CODE |
| 0xEEE4 | Error Chyba CRC during reading ROM-CODE |

Register „**NEW ROM code”.** It will be written read ROM code into such marked registers. After command 0x5FF. Only one set of registers is for each bus.

| Register No. | | Register description | |
|---|---|---|---|
| Bus 1 | Bus 2 | Higher byte | Lower byte |
| 1012 | 1022 | 8bit family code | serial number - 1 |
| 1013 | 1023 | serial number - 2 | serial number - 3 |
| 1014 | 1024 | serial number - 4 | serial number - 5 |
| 1015 | 1025 | serial number - 6 | serial number - 7 |
| 1016 | 1026 | serial number - 8 | CRC |

Registr „Statistics”. It will be written read ROM code into such marked registers. After command 0x5FF. Only one set of registers is for each bus.

| Register No. | | | Register description |
|---|---|---|---|
| PORT1 | PORT2 | PORT2 | |
| 1030 | 1040 | 1050 | Number of received messages |
| 1031 | 1041 | 1051 | Number of sent messages |
| 1032 | 1042 | 1052 | Number of error messages |

| Register No. | Description |
|---|---|
| 1053 (1054) | Setting of time – delay<br>Gateway waits for some time with sending respond after receiving message from master.This time consists of basic time (for baud rate 9600 it is 4 ms) and set value time delay. Some devices need longer time for switching from sending to receiving. When time is longered before sending respond, it can solve this problem. |
| 1055 | Both buses have the same power supply. This power supply is with DC-DC separated from communication ports, USB and main power supply. If there is problem with sensors on the bus, one of the option to run communication agagin may be short-term disconnection from power supply.<br><br>Disconnection from power supply is controlled with register 1055.<br><br>Following options are offered:<br>0 – Disconnection from power supply<br>1 – Connected to power supply (Default)<br>2 – Connected to power supply. If a sensor shows 3x in a row timeout or error, it will happen disconnection from power supply for approximately 5 s and then connection.<br>Another disconnection for 5 s and connection can be in one minute.<br>3 – The same as point 2. Difference is that another disconnection for 5 s and connection can be in 5 minutes (default 3).<br>4 – The same as point 2. Difference is that another disconnection for 5 s can be in 10 minutes.<br>5 – The same as point 2. Difference is that another disconnection for 5 x can be in 15 minutes.<br><br>If i-button is used on a bus, set the register 1055 on value 1!!! |

Registers for newly found sensors. When using commands for searching new sensors the results are saved into following table.

| | No | Description | |
|---|---|---|---|
| Service registers | 1100 | ROM code1 (family code, serial number) | LINE 1 |
| | 1101 | ROM code2 (serial number) | |
| | 1102 | ROM code3 (serial number) | |
| | 1103 | ROM code4 (serial number, CRC) | |
| | 1104 | Index 0 | |
| | 1105 | ROM code1 (family code, serial number) | |
| | 1106 | ROM code2 (serial number) | |
| | 1107 | ROM code3 (serial number) | |
| | 1108 | ROM code4 (serial number, CRC) | |
| | 1109 | Index 1 | |
| | … | | |
| | 1195 | ROM code1 (family code, serial number) | |
| | 1196 | ROM code2 (serial number) | |

| | | | |
|---|---|---|---|
| | 1197 | ROM code3 (serial number) | |
| | 1198 | ROM code4 (serial number, CRC) | |
| | 1199 | Index 19 | |
| | | | |
| | 1200 | ROM code1 (family code, serial number) | LINE 2 |
| | 1201 | ROM code2 (serial number) | |
| | 1202 | ROM code3 (serial number) | |
| | 1203 | ROM code4 (serial number, CRC) | |
| | 1204 | Index 0 | |
| | 1205 | ROM code1 (family code, serial number) | |
| | 1206 | ROM code2 (serial number) | |
| | 1207 | ROM code3 (serial number) | |
| | 1208 | ROM code4 (serial number, CRC) | |
| | 1209 | Index 1 | |
| | … | | |
| | 1295 | ROM code1 (family code, serial number) | |
| | 1296 | ROM code2 (serial number) | |
| | 1297 | ROM code3 (serial number) | |
| | 1298 | ROM code4 (serial number, CRC) | |
| | 1299 | Index 19 | |

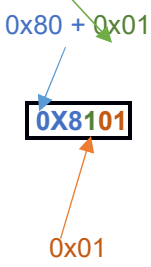
## Adding new device (sensor)

You can add new sensor in two ways:

1. Empty bus
   - New sensor is connected to empty bus.
   - ROM code of this sensor is read by writing value 0x05FF into the register 1010 for the bus 1 or 1020 for the bus 2. (numbering from zero)
   - Read number is saved by writing value 0x09nn into the register 1010 (or 1020). The position is determined by number nn in hex.

2. Occupied bus
   - New sensor is connected to the bus with connected sensors.
   - By writing value into the register 1010 (or 1020), all new ROM codes of sensors (max. 20) that are not saved yet, occur
   - New codes occur in registers 1100 – 1199. 5 registers are assigned for each new sensor. Always the 5th register displays the position.
   - New found sensor is written on appropriate position by writing value (e.g. 0x8208). Value consists of as follows. Higher byte displays position + 128 (in 1100-1199) from which ROM code is selected. Lower byte is position on which it is saved.
   - Example: 1100 – 1104 is zero position (0x80), 1105 – 1109 is first position (0x81).. etc.
   - If I want to save first position on position 12, the value written into register the 1010 will look like as follows: 0x810C.
   - Indicator LED indicating state on bus flickers during searching.

| No | Description | | COMMAND | No | Description | |
|---|---|---|---|---|---|---|
| 1100 | ROM code1 **0x28B0** | Index 0 | | 0 | ROM code1 **0x10DA** | Index 0 |
| 1101 | ROM code2 **0x0E59** | | | 1 | ROM code2 **0xF8F7** | |
| 1102 | ROM code3 **0x0700** | | | 2 | ROM code3 **0x0208** | |
| 1103 | ROM code4 **0x008A** | | | 3 | ROM code4 **0x00A6** | |
| 1104 | Index 0 | | | 4 | Value 1 | |
| 1105 | ROM code1 0x1094 | Index 1 | | 5 | Value 2 | |
| 1106 | ROM code2 0xA516 | | | 6 | Value 3 | |
| 1107 | ROM code3 0x0308 | | | 7 | Value 4 | |
| 1108 | ROM code4 0x00D4 | | | 8 | Value 5 | |
| 1109 | Index 1 | | | 9 | Error | |
| 1110 | ROM code1 | Index 2 | | 10 | ROM code1 0x1094 | Index 1 |
| 1111 | ROM code2 | | | 11 | ROM code2 0xA516 | |
| 1112 | ROM code3 | | | 12 | ROM code3 0x0308 | |
| 1113 | ROM code4 | | | 13 | ROM code4 0x00D4 | |
| 1114 | Index 2 | | | 14 | Value 1 | |
| 1115 | ROM code1 | Index 3 | | 15 | Value 2 | |
| 1116 | ROM code2 | | | 16 | Value 3 | |
| 1117 | ROM code3 | | | 17 | Value 4 | |
| 1118 | ROM code4 | | | 18 | Value 5 | |
| 1119 | Index 3 | | | 19 | Error | |




Found serial numbers on bus 1 cannot be written on bus 2 and on the contrary.

| Bus 1 | | | | Bus 2 | | | |
|---|---|---|---|---|---|---|---|
| Position from | Position where | Result value in hex. | Result value in dec. | Position from | Position where | Result value in hex. | Result value in dec. |
| 0 | 0 | 8000 | 32768 | 20 | 20 | 9414 | 37908 |
| 1 | 1 | 8101 | 33025 | 21 | 21 | 9515 | 38165 |
| 2 | 2 | 8202 | 33282 | 22 | 22 | 9616 | 38422 |
| 3 | 3 | 8303 | 33539 | 23 | 23 | 9717 | 38679 |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 4 | 4 | 8404 | 33796 | 24 | 24 | 9818 | 38936 |
| 5 | 5 | 8505 | 34053 | 25 | 25 | 9919 | 39193 |
| 6 | 6 | 8606 | 34310 | 26 | 26 | 9A1A | 39450 |
| 7 | 7 | 8707 | 34567 | 27 | 27 | 9B1B | 39707 |
| 8 | 8 | 8808 | 34824 | 28 | 28 | 9C1C | 39964 |
| 9 | 9 | 8909 | 35081 | 29 | 29 | 9D1D | 40221 |
| 10 | 10 | 8A0A | 35338 | 30 | 30 | 9E1E | 40478 |
| 11 | 11 | 8B0B | 35595 | 31 | 31 | 9F1F | 40735 |
| 12 | 12 | 8C0C | 35852 | 32 | 32 | A020 | 40992 |
| 13 | 13 | 8D0D | 36109 | 33 | 33 | A121 | 41249 |
| 14 | 14 | 8E0E | 36366 | 34 | 34 | A222 | 41506 |
| 15 | 15 | 8F0F | 36623 | 35 | 35 | A323 | 41763 |
| 16 | 16 | 9010 | 36880 | 36 | 36 | A424 | 42020 |
| 17 | 17 | 9111 | 37137 | 37 | 37 | A525 | 42277 |
| 18 | 18 | 9212 | 37394 | 38 | 38 | A626 | 42534 |
| 19 | 19 | 9313 | 37651 | 39 | 39 | A727 | 42791 |

Example:
New ROM code is displayed in registers 1200-1203. It is such a data in the register 1204 that shows us it is 20th position.
So we will write COM code from the position **20** on position **33**.
For calculation it is possible to use following formula:
Result value = (Position from + 128) * 256 + Position where
Result value = (**20**+128)*256+**33**
Result value = 37921 = 9421 Hex