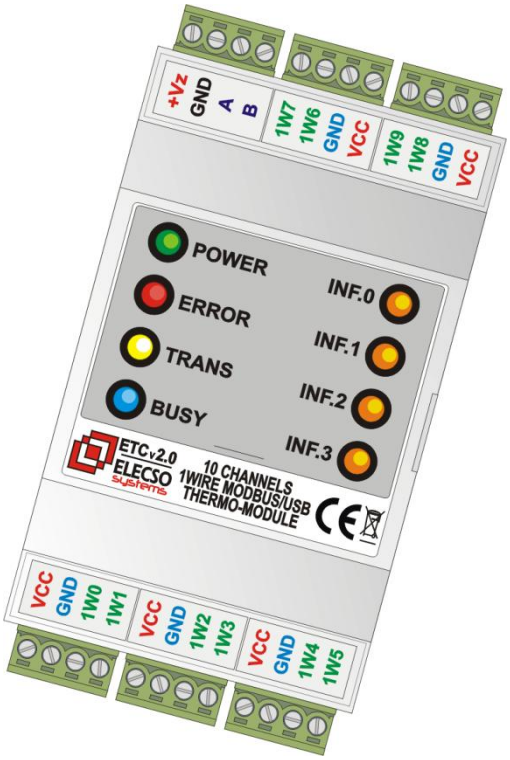


ELECSO ECT v2.x

10 channels, 1Wire, MODBUS/USB Module

User manual version 2.0

Contents

1. Introduction

ELECSO ETC Module is a small electronic device, used for data acquisition from multipoint temperature measurement system. It works with digital temperature sensors with 1-WIRE interface. To communicate with the system, implemented MODBUS protocol based on the popular EIA-485 interface. Another advantage is the built-in USB serial interface, which are equipped with all personal computers PCs. These features make the device adaptable and universal in all automatic systems.

1.1 Technical parameters

Terms of operation

- Operating Temperature: -40ºC to +75 ºC
- Humidity: 5 to 90% non-condensing
- Power supply: 8V…28 VDC

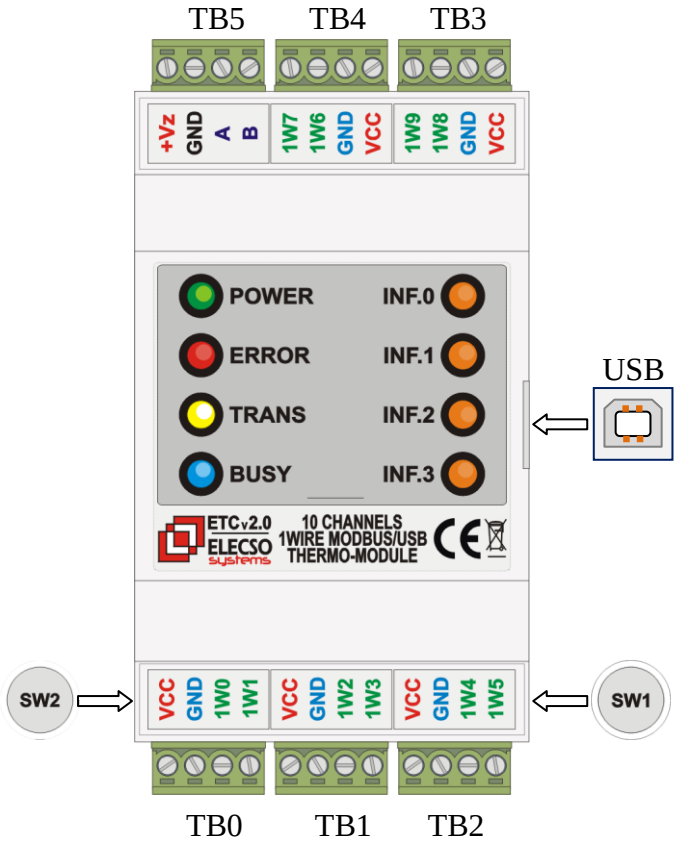
Hardware features

- Support up to 120 temperature sensors (in standard to 100 sensors)
- Internal temperature sensor
- 10 independent measuring channels with a measurement accuracy of 0.5 ° C with resolution 0,0625ºC
- Measuring range: -40 ° C to +125 ° C
- Supported network physical layer EIA-485 with MODBUS RTU protocol
- Modbus Addressing: 1-247
- Short-circuits protection on all channels
- ESD Protection on all channels
- Supply voltage monitoring
- Probe currents monitoring
- Sound alert system
- Two system switches
- Removable Terminal Block connectors, Wire Range: 28 - 14AWG (diameter: 0.3 – 1.5mm)
- DIN (35mm) rail mounting enclosure
- Dimensions: 107x51x65 (LxWxH)

1.2 System Interfaces

- USB (configuration, measuring, diagnostic)
- EIA-485 (measuring, diagnostics), network up to 1200m, 32 devices on the bus (in standard)

2. ELECSO ETC - Functional Overview



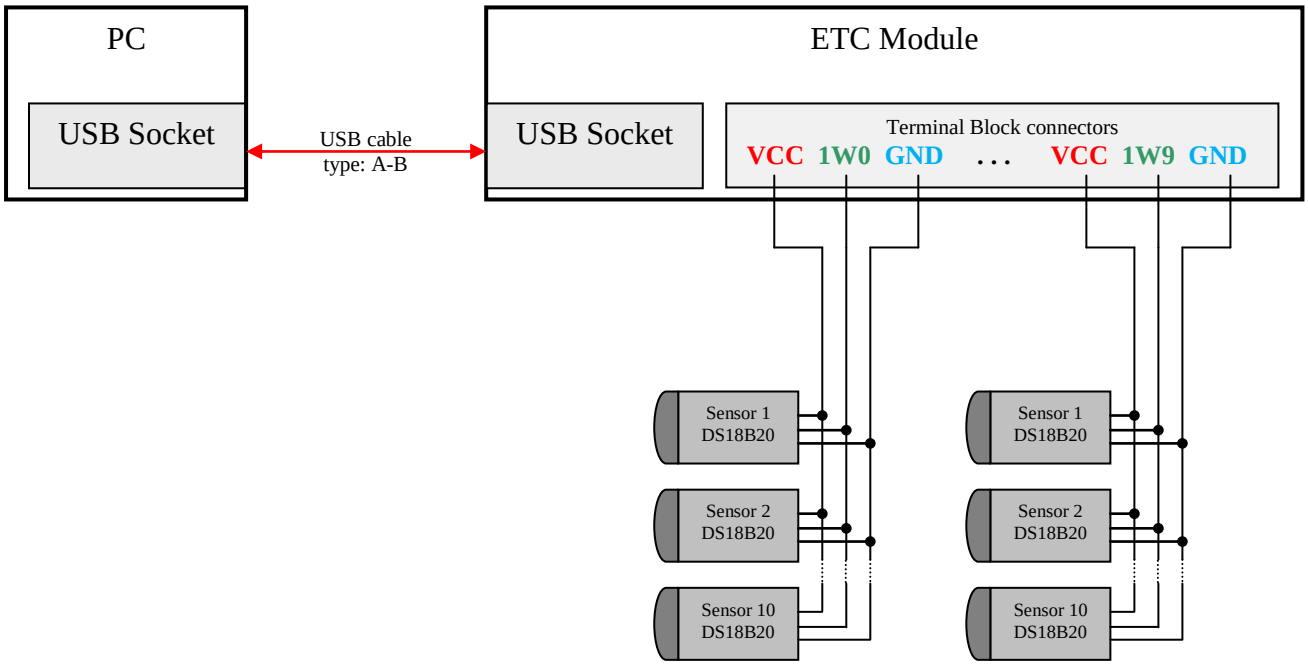
Explanations of signals and indicators

GND	Power supply, negative
+Vz	Power supply, positive
A, B	EIA-485 signals (Half Duplex)
GND	Probe, negative
VCC	Probe, positive
1W0…1W9	Probe data input/output channel
SW1, SW2	**System switches used for:** - manual probes configuration function - update firmware process
TB0…TB4	Removable Terminal block connectors - probes
TB5	Removable Terminal block connector: – Power supply – EIA-485 signals
USB	USB socket
LED POWER	Indicates proper powering
LED ERROR	Indicates system errors: - Exceeding the current (ex. wire crosses) - transmission errors
LED TRANS	Indicates data transmision on EIA-485/USB
LED BUSY	Indicates data transmision on 1-Wire channels
LED INF0…3	System indicators

3. System configurations

3.1 USB Interface, configuration mode – wiring diagram

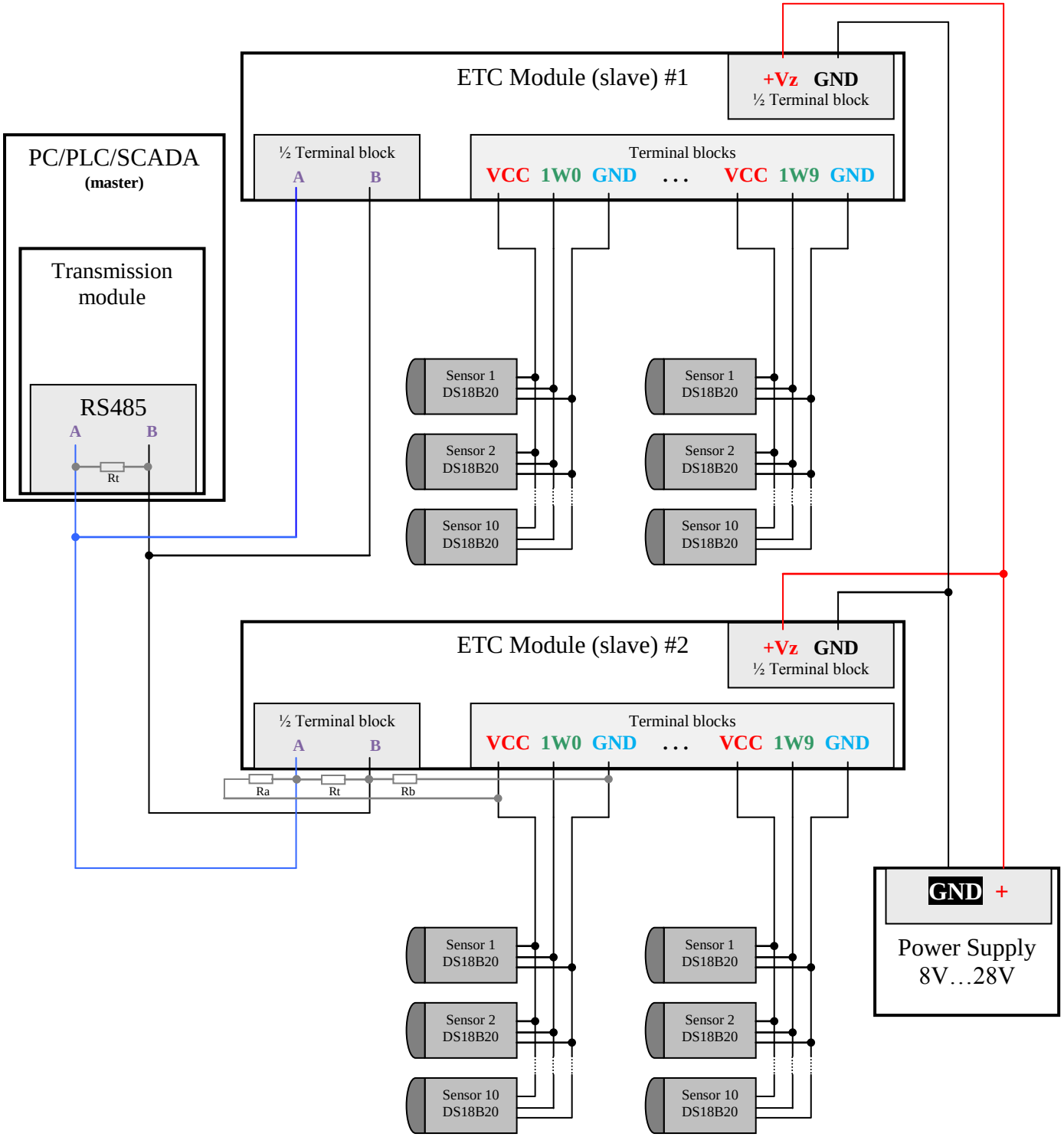
In this moment, the USB interfaces are available in any PC computer. Therefore, ETC ELECSO modules are equipped with a universal serial bus - USB. This interface implementing data exchange with powering. This mode is ideal for configuration mode and small systems where the measurement module ETC is near the PC. Diagram looks like this:



3.2 EIA-485 (Half Duplex) - wiring diagram

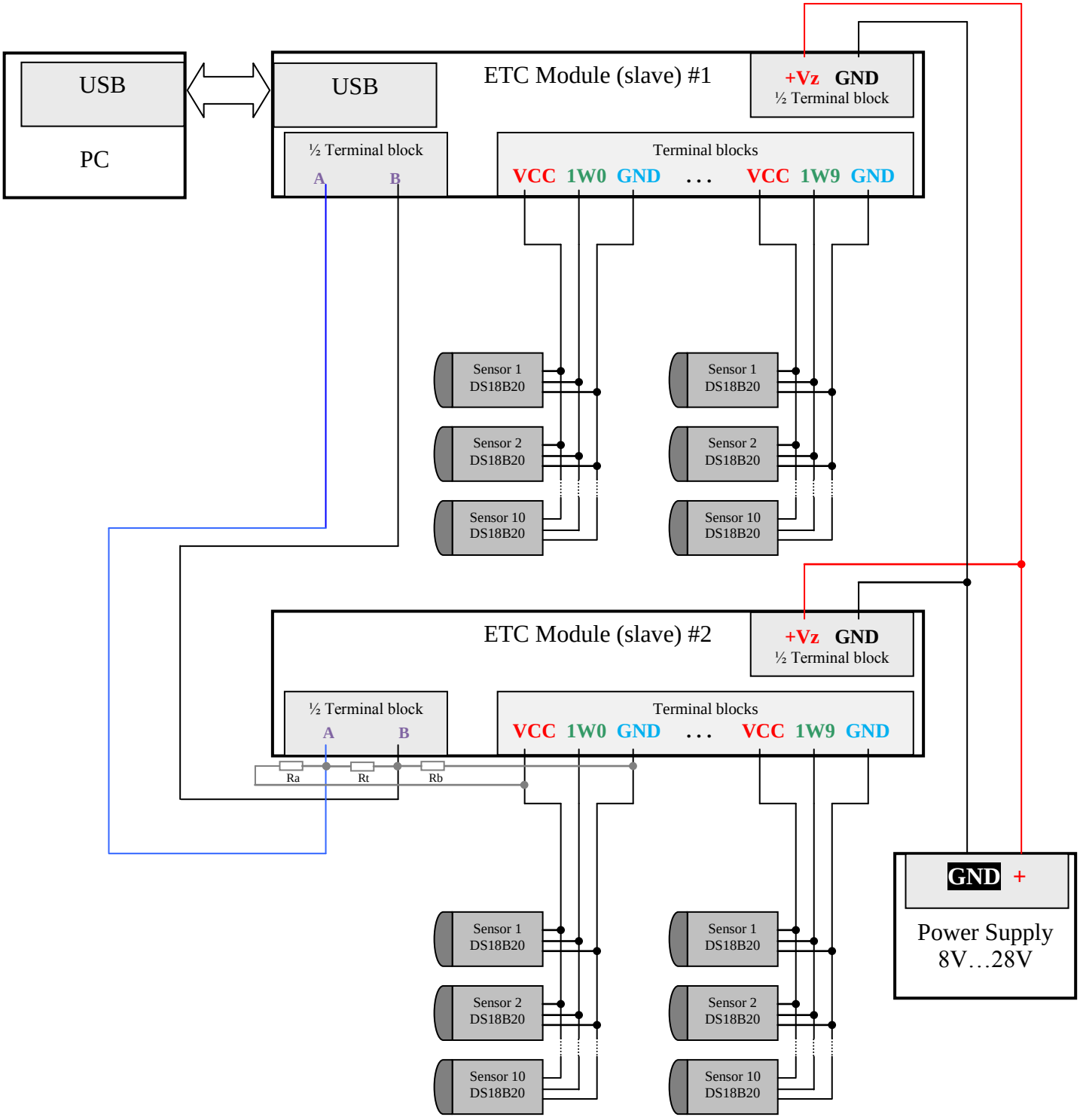
In this mode one pair (two twisted wires) of transmission is needed.
Transmission parameters are constant:

- speed: 9600
- data: 8 bits
- parity: none
- stop bit: 1

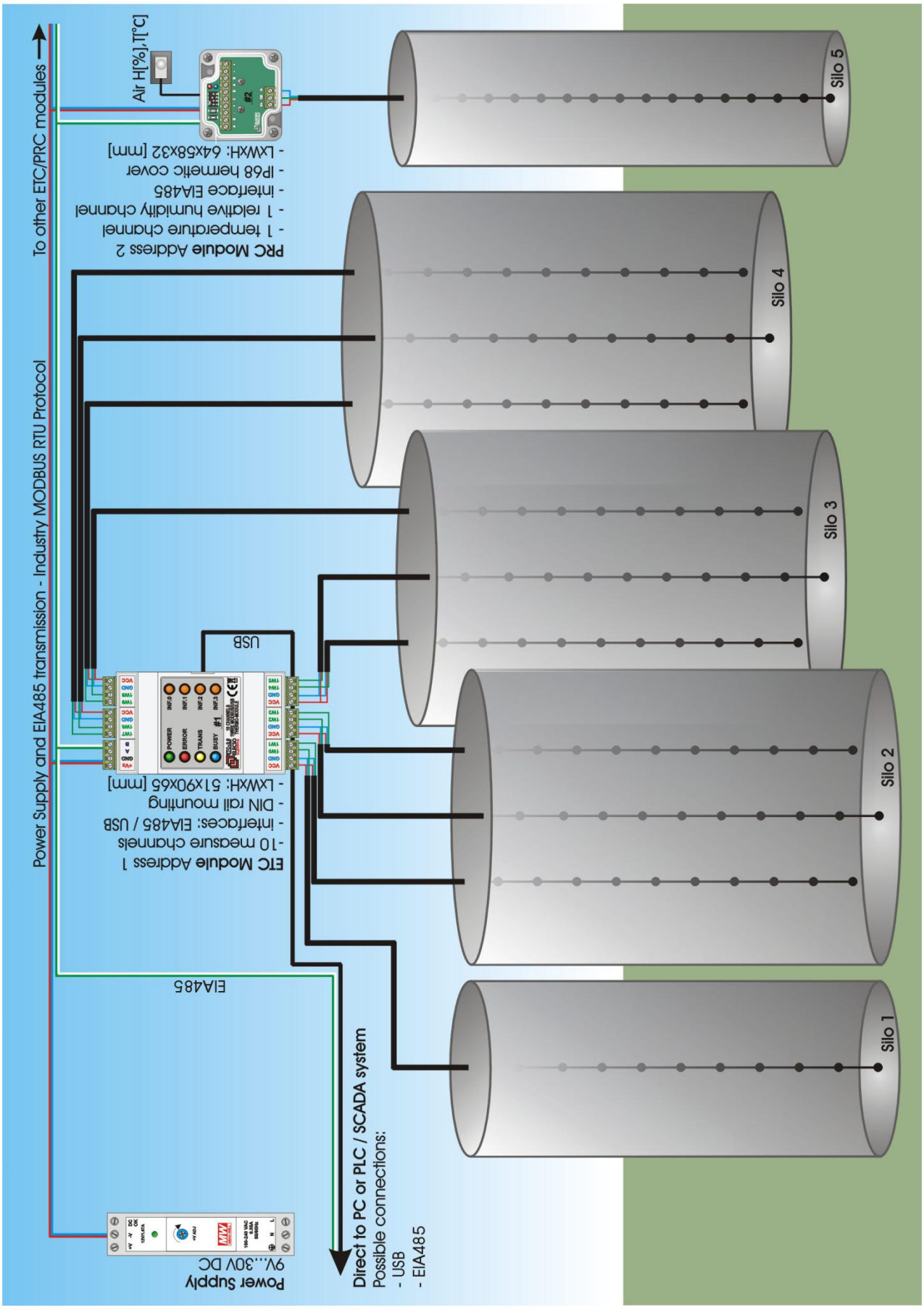


3.3 USB – EIA-485 network – wiring diagram

ETC devices, have the ability to work without the participation of the network EIA-485 converter. You can easily build a network that consists only of the master controller (PC or industrial controller) and the modules ETC only. Schematically, it shows the diagram below:

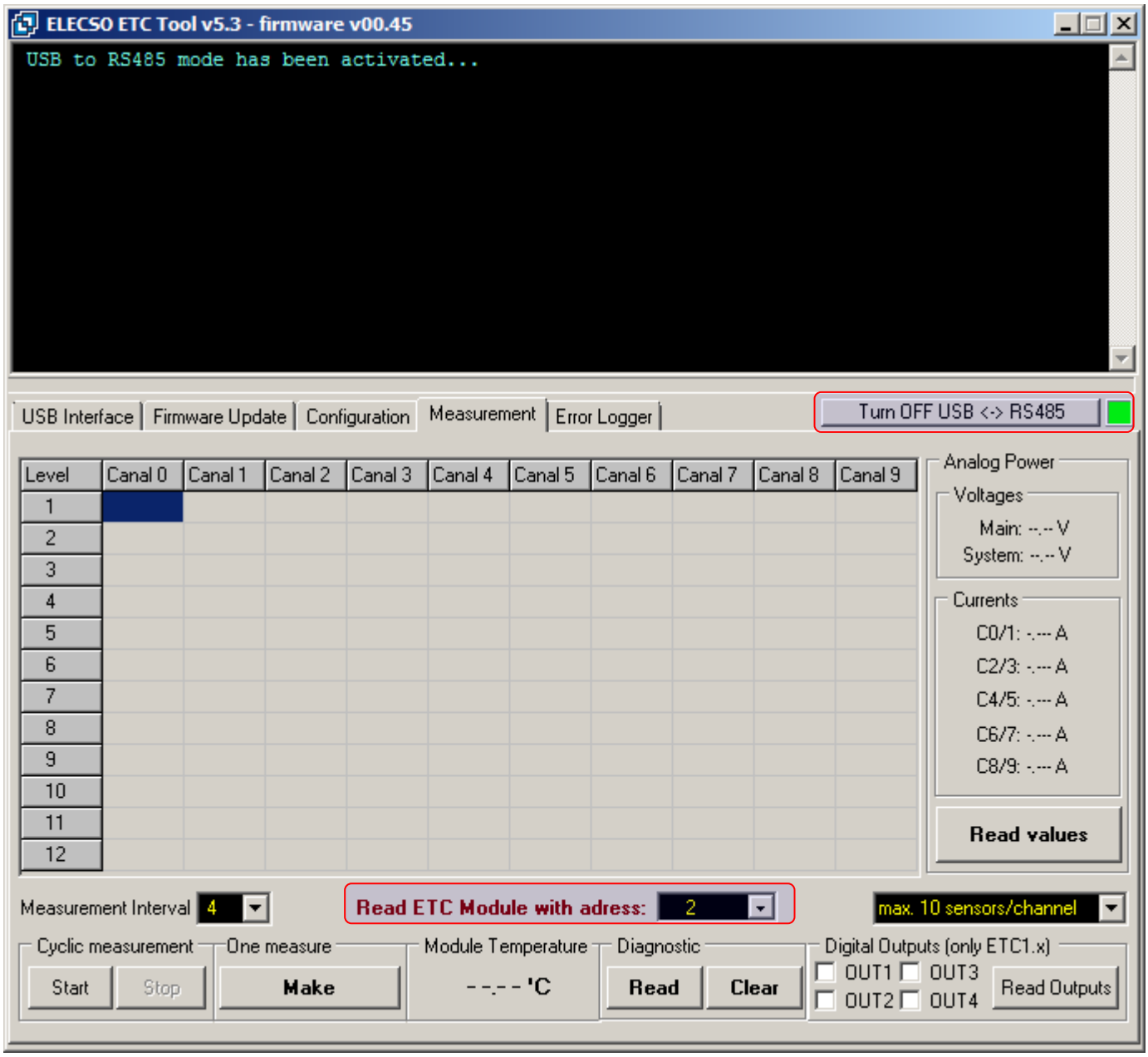


3.4 EIA-485 network example



3.5 Converter Mode

This connection is used primarily for system diagnostics. Switching module to converter mode is performed by clicking on the ***Turn ON USB <-> RS485*** function in ETC Tool application. Active mode is indicated by green icon located next to the button.



In **USB to RS485 mode** You can set the address of any network device and perform diagnostics and measure tests…

4. Configuration – software module

4.1 USB Drivers

There are two methods of installing the drivers:

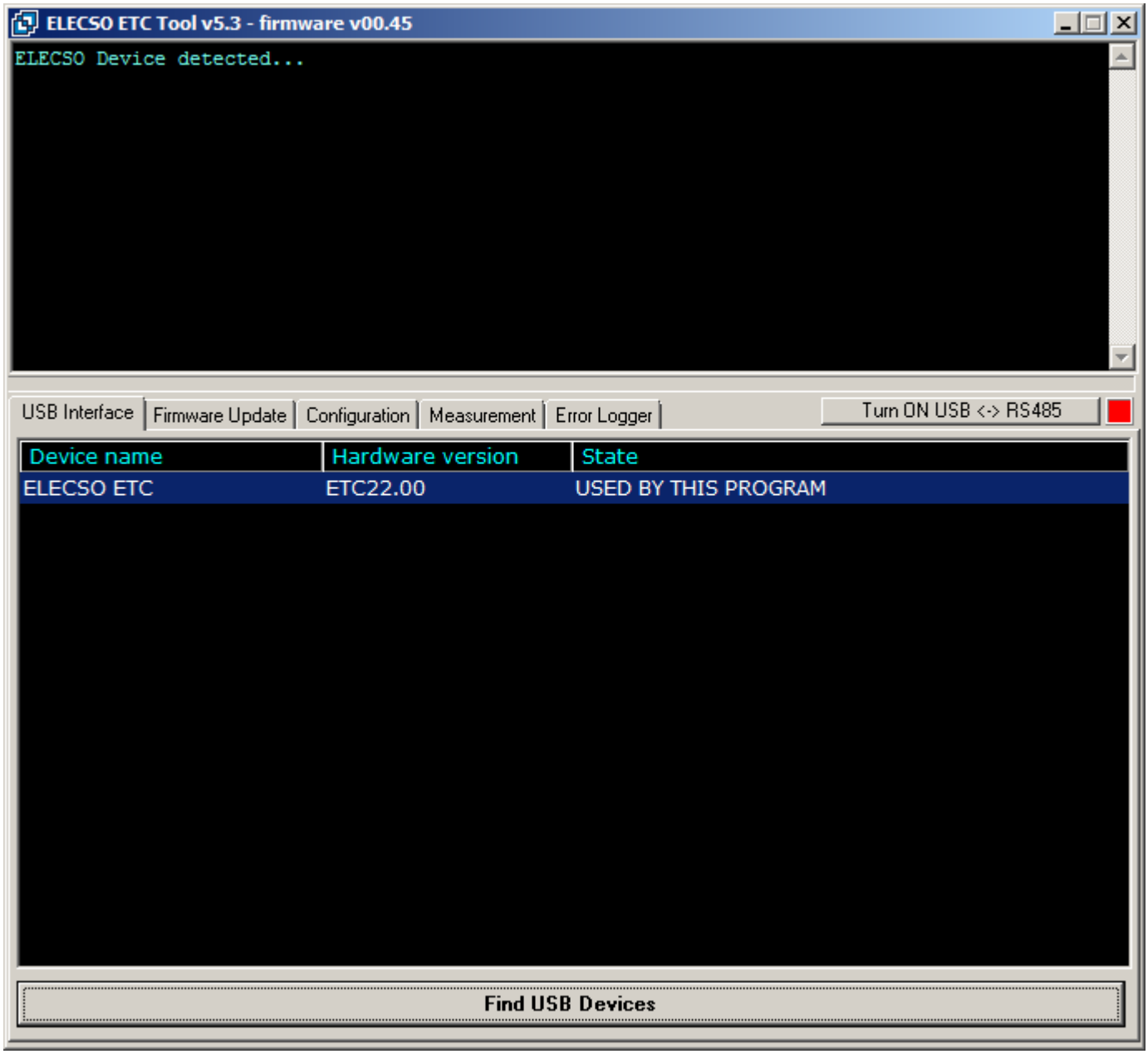
1. Automatic method (require internet connection)
   Connect ETC module to your computer, device will be detected, select “Install automatically” - system will automatically download and install the proper driver.

2. Manual method - drivers are available on support site: http://www.elecso.pl/support/
   Unpack drivers to folder on Your PC, connect ETC module, device will be detected, select “Install from specific location” and set path to unpacked driver

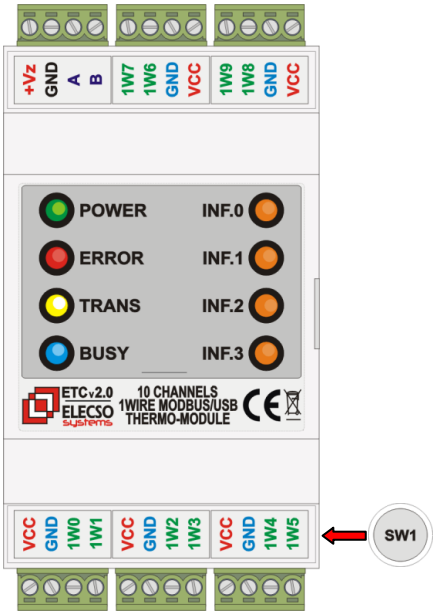
4.2 Service application: *ETC Tool*

The newest version of ETC Tool application is available on support site: http://www.elecso.pl/support/

4.2.1 Module: USB Interface – here are displayed Device Name and Hardware Version.

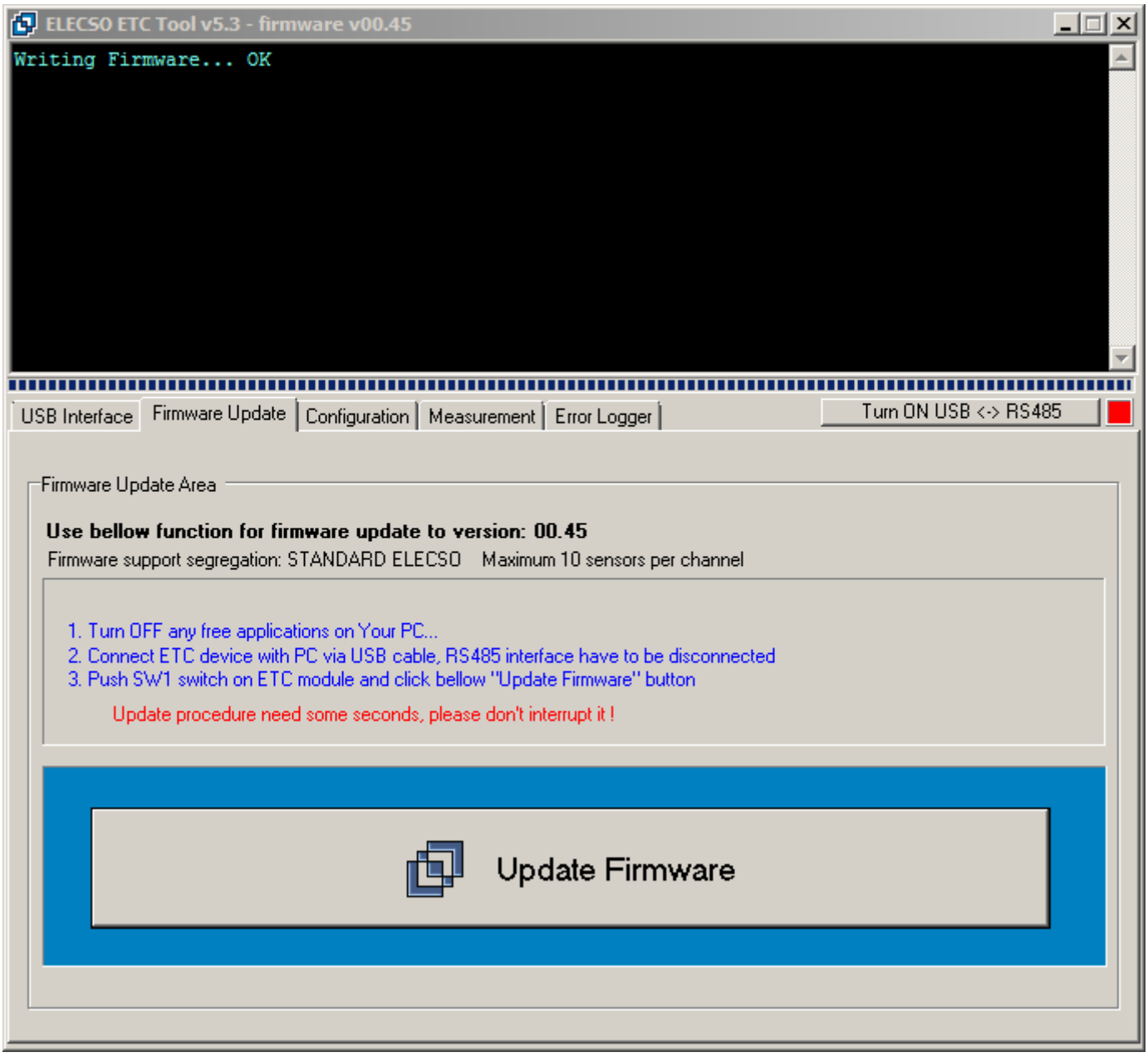


4.2.1 Module: Firmware Update



Use this module for firmware updating. It is simple and fast procedure. Below is described whole procedure, step by step:

1. Turn off any applications on Your PC
2. Connect ETC Device to PC via USB port, EIA-485 interface have to be disconnected from network.
3. Push on **SW1** switch and click ***Update Firmware*** button.
   Note, push SW1 while clicking on Update Firmware button, you can unpush it while firmware writing is in progress…

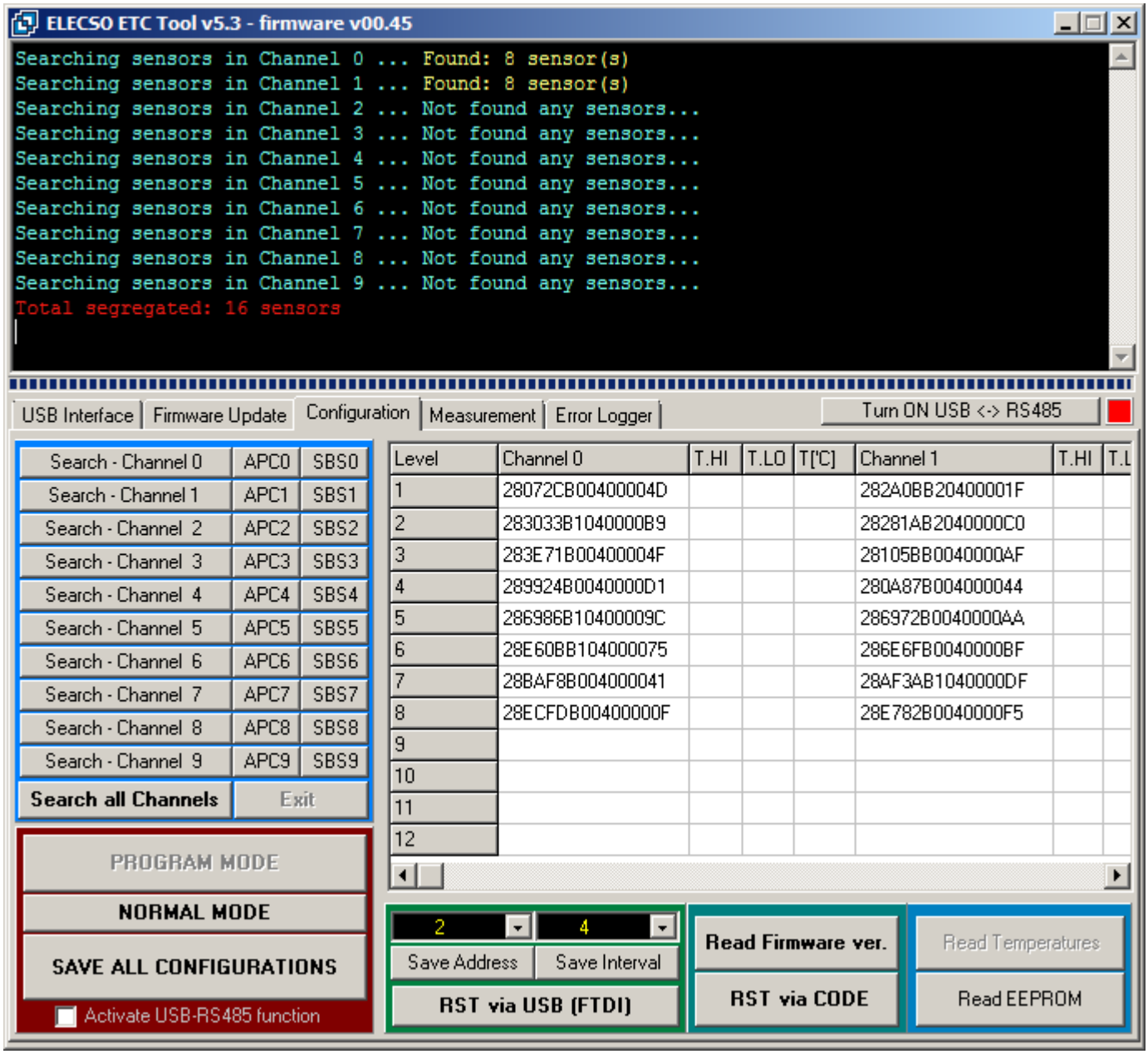


4.2.2 Module: Configuration

This module is use for:
- setting modbus slave address (1…247),
- setting measuring interval (1…255),
- detecting and configuring measure probes,
- reading module firmware version,
- reading eeprom settings for backup

Below picture show result of ***Search all Channels*** function, which detect and configure all sensors in all channels. This is dedicated function for standard probes.

Note, that all changes have to be applied with *SAVE ALL CONFIGURATIONS* function!



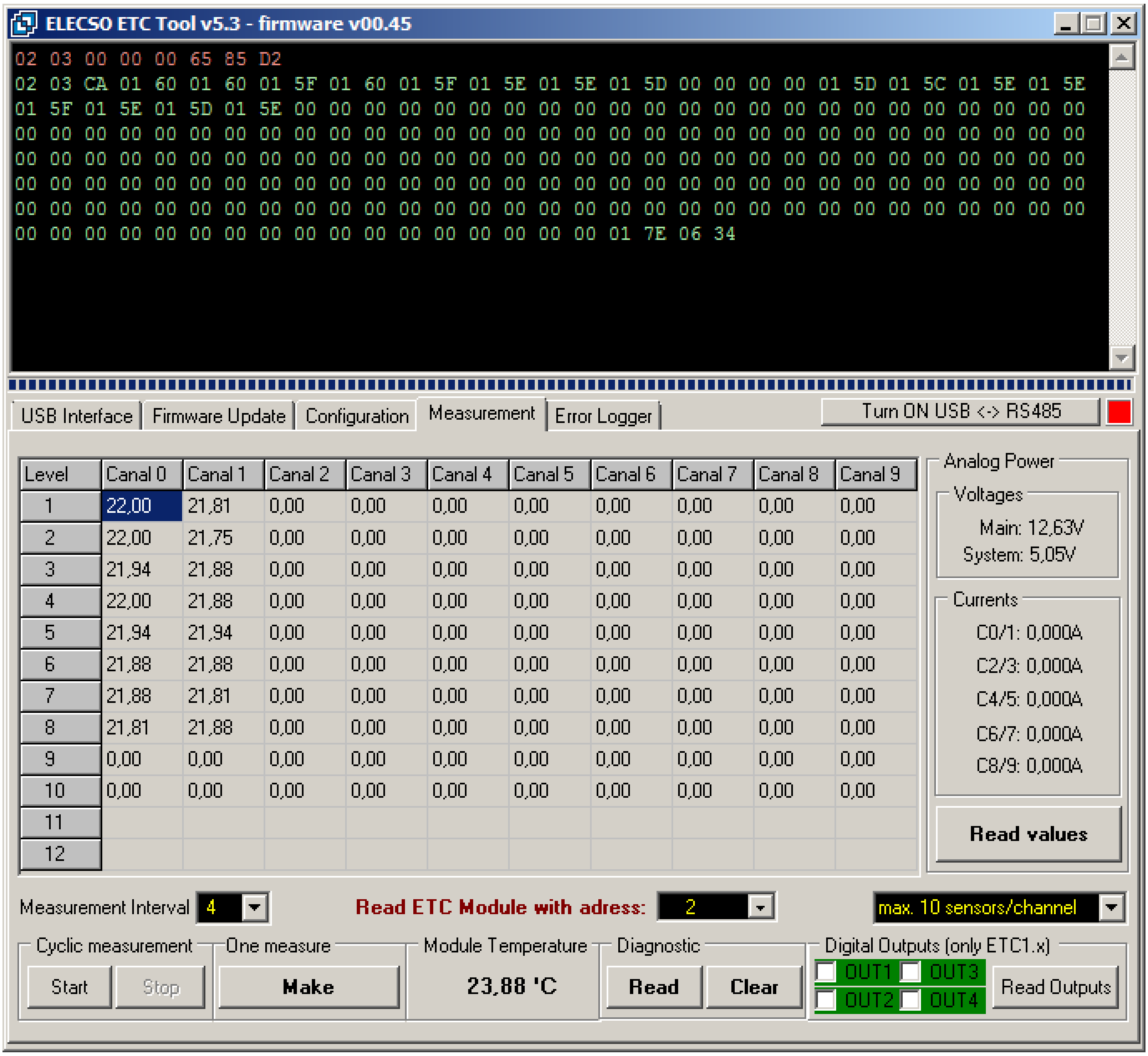
4.2.3 Module: Measurement

Measurement module functions are based on MODBUS holding registers. In this section You can:

- read actual temperatures in all channels
- read internal module temperature
- read/clear diagnostic registers
- read power supply voltages and probe currents
- read/write Digital Outputs (only ETCv1.x)

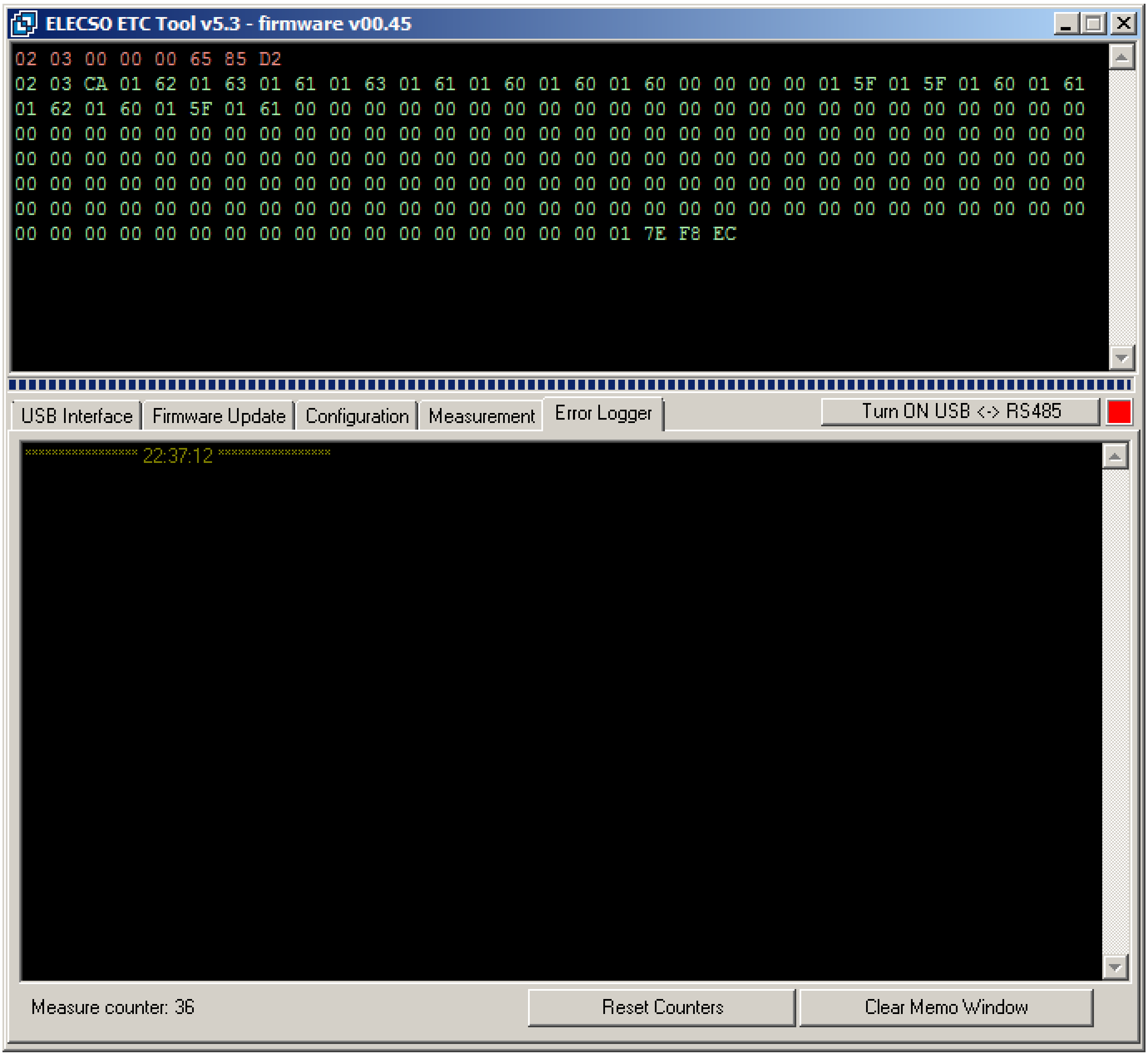
Values of probe temperatures are showed from the highest sensor – from level 1. In this example level 8 has temperatures from the bottom sensor.

Note, that in main black memo, query (data from master) is marked in **red colour**, and answer (data from slave) in **green colour**. It can be useful for tests for SCADA system developers.



4.2.4 Module: Error Logger

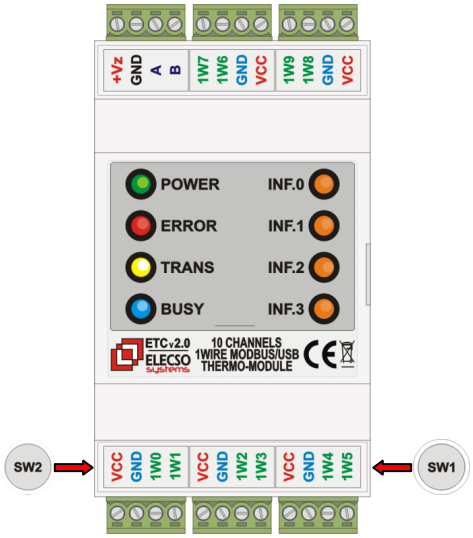
This module shows all errors from interfaces: USB, EIA-485, 1-WIRE. It is active while turned on ***Cyclic measurement*** function.



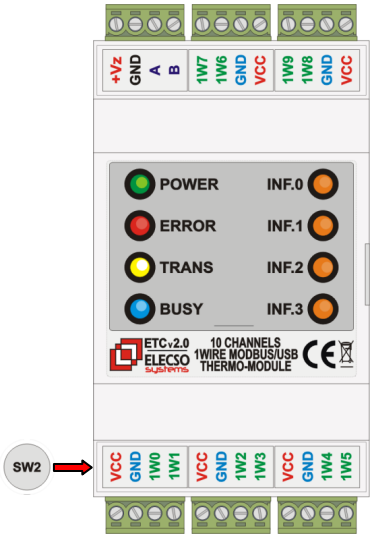
5. Probe Configuration – Manual mode

Probes can be detected and configured in 2 modes:
- In ***Program Mode*** with ***Search All Channels*** function – please, refer to configuration, page 11.
- In manual mode with ***SW1*** and ***SW2*** switches. It is really easy and fast without any additional devices. Below is described whole procedure, step by step…

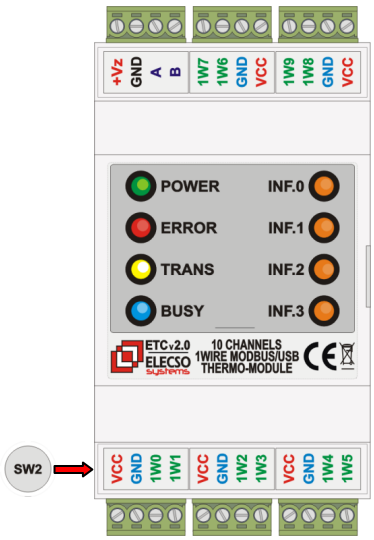
ETC module have to be powered and probes connected to terminal blocks…



1. Press SW1 and SW2 at the same time (Open Manual Mode)
   Results:
   - modulated sound
   - Indicator ***INF.0*** blinking ~1Hz



2. Press SW2 (Search function)
   Results:
   - Modulated sound (high sound mean – found probe)
   - Indicator ***INF.0*** blinking ~2Hz



3. Press SW2 (Save Configuration to memory)
   Results:
   - Indicator ***BUSY*** active
   - Module reboot

6. For developers - MODBUS RTU PROTOCOL

The main advantages of the MODBUS protocol are:

- simplicity of the solutions used,
- disclosure of protocol specifications,
- protection of transmitted messages against errors,
- confirmation of execution of remote commands and error messages,
- constant frame format and a set of standard functions for data exchange,
- mechanisms to prevent system suspension.

6.1 Transmission frame

The MODBUS protocol frame specifies the format of the messages to be sent and contains: recipient address, function code representing the requested command, function data, and control word to secure the message being transmitted. The form of the query frame sent by the master unit and slave response frame is similar. The difference is that in the data field of the response frame there are data that the Master requested.

Address [1B]	**Command Code [1B]**	**Data [x Bytes]**	**Checksum [2 Bytes]**

Description of frame fields:

• Slave **Address**: number from 1 - 247, 0 - broadcast address,
• **Command Code**: is a number from 1 to 127,
• **Data** field: Its length depends on the type of message and may include:
- in case of inquiries - function arguments,
- in the case of a positive answer, the arguments of the function,
- in the case of a special response - the error code,
- in some cases it may be 0,
• **Checksum**: determined from the content of the message sent. In the MODBUS RTU protocol, the checksum CRC16 [**CRCL, CRCH**] is used as the message frame protection. Its value is determined by the device that broadcasts the content of the transmitted message and puts it in the frame after the information part.

Special Slave Responses

For ELECSO ETC devices:

Address values outside the range	`02 83 02 30 F1`

Gdzie:

0x**02** – Slave Address [1Byte]
0x**83** – Error **DATA** field [1Byte]
0x**02** – Error Code – address field out the range[1Byte]
0x**30F1** - Checksum CRC16 [2Bytes: CRCL,CRCH]

Data values outside the range	`02 83 03 30 F1`

Gdzie:

0x**02** – Slave Address [1Byte]
0x**83** – Error **DATA** field [1Byte]
0x**02** – Error Code – data field out the range [1Byte]
0x**30F1** - Checksum CRC16 [2Bytes: CRCL,CRCH]

Slave device (ETC Module) does not send an answer if wrong CRC Checksum occurs.

6.2 Commands and Responses in ELECSO ETC Modules.

For slave device - ELECSO ETC **address 2**.

Read temperature registers (Holding Registers – Read Code: 0x03)

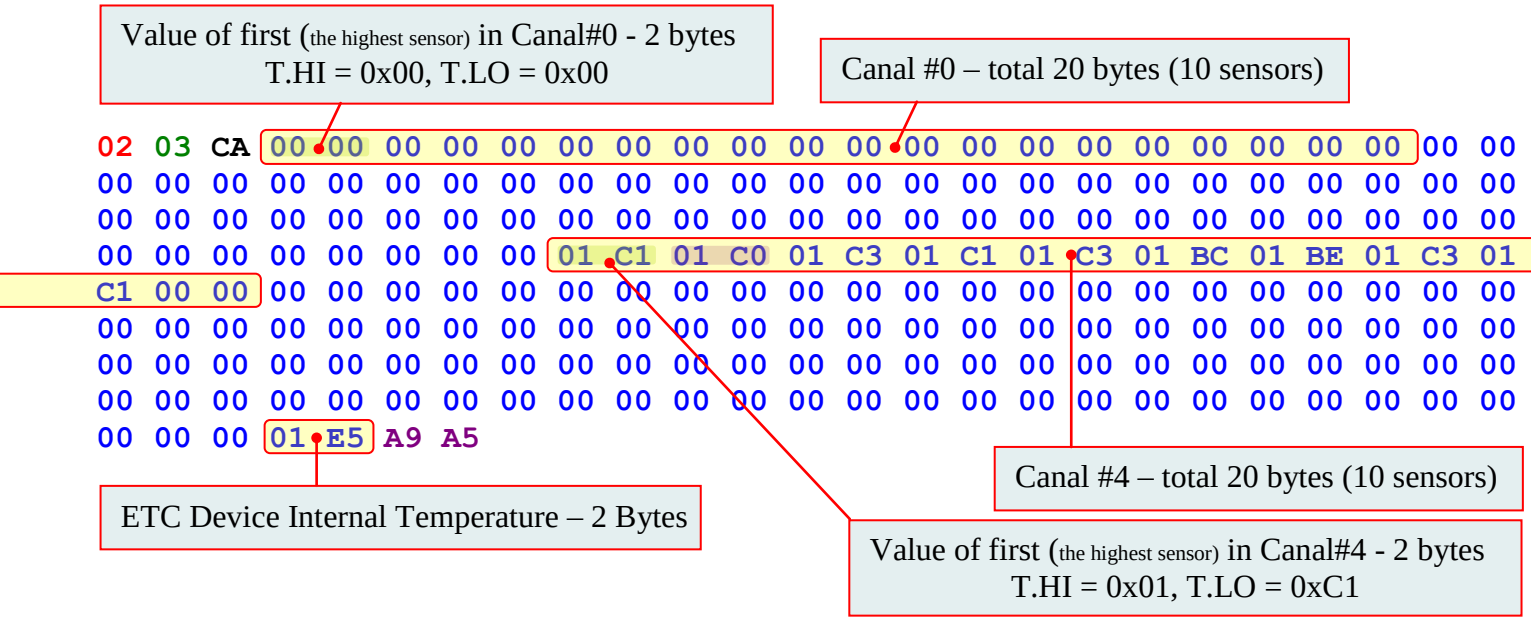
The master command, shown in hexadecimal, has the form:

`02 03 00 00 00 65 85 D2`

0x**02** – slave address [1Byte]
0x**03** – Read Temperatures Registers Code [1Bajt]
0x**0000** – starting point of the data table, possible range: from 0x0000 to 0x0064* [2Bytes]
0x**0065** – amount of registers (number of sensors), possible range: from 0x0000 to 0x0065* [2Bytes]
0x**85D2** - Checksum CRC16 [2Bytes: CRCL,CRCH]

*For firmware versions x0.xx - 10 sensors per channel

Below is an example of ELECSO ETC Slave response (**for firmware versions x0.xx - 10 sensors per channel**):



0x**02** – slave address [1Byte]
0x**03** – Read Temperatures Registers Code [1Bajt]
0x**CA** – read data number: 0xCA [1Byte]
0x**0000**… **01E5** – read temperature values (in this case 202 bytes)
0x**A9A5** - Checksum CRC16 [2Bytes: CRCL,CRCH]

6.3 Table of temperature registers

Address for firmware x0.xx (max 10 sens/canal)	**MODBUS Registers**	**Address for firmware x1.xx** (max 12 sens. canal)	**MODBUS Registers**	**Canal**
0…19	**0 - 9**	**0…23**	**0 - 11**	Canal #0
20…39	**10 - 19**	**24…47**	**12 - 23**	Canal #1
40…59	**20 - 29**	**48…71**	**24 - 35**	Canal #2
60…79	**30 - 39**	**72…95**	**36 - 47**	Canal #3
80…99	**40 - 49**	**96…119**	**48 - 59**	Canal #4
100…119	**50 - 59**	**120…143**	**60 - 71**	Canal #5
120…139	**60 - 69**	**144…167**	**72 - 83**	Canal #6
140…159	**70 - 79**	**168…191**	**84 - 95**	Canal #7
160…179	**80 - 89**	**192…215**	**96 - 107**	Canal #8
180…199	**90 - 99**	**216…239**	**108 - 119**	Canal #9
200…201	**100**	**240…241**	**120**	ETC Internal temperature (1 register = 2 bytes)

Address for firmware x2.xx (max 16 sens/canal)	MODBUS Registers	Address for firmware x3.xx (max 24 sens. canal)	MODBUS Registers	Canal
0…31	0 – 15	0…47	0 – 23	Canal #0
32…63	16 – 31	48…95	24 – 47	Canal #1
64…95	32 – 47	96…143	48 – 71	Canal #2
96…127	48 – 63	144…191	72 – 95	Canal #3
128…159	64 – 79	192…239	96 – 119	Canal #4
160…191	80 – 95	240…287	120 – 143	Canal #5
192…223	96 – 111	288…335	144 – 167	Canal #6
224…254	112 – 127	336…383	168 – 191	Canal #7
256…287	128 – 143	384…431	192 – 215	Canal #8
288…319	144 – 159	432…479	216 – 239	Canal #9
320…321	160	480…481	240	ETC Internal temperature (1 register = 2 bytes)

Bellow is formula for calculating register for temperature:

$$T_C = \frac{(T.HI \times 256 + T.LO) \times 125}{2000}$$

For above example – temperature of first sensor in canal#4 is:

T.HI = 0x01 = 1 (decimal)
T.LO = 0xC1 = 193 (decimal)

$$T_C = \frac{(1 \times 256 + 193) \times 125}{2000} = 28.0625^{o} C$$

C code example:

```
float Konwertuj_Temp(unsigned char Temp_H, unsigned char Temp_L)
{
    float Wynik = 0;
    unsigned char uc_1 = 0;
    unsigned int ui_1 = 0; unsigned int ui_2 = 0; unsigned int ui_3 = 0;
    //---------------------------------------------------------------------

     uc_1 = Temp_H & 0xF0;

     if(uc_1 != 0xF0) // for positive temperature values
     {
        ui_1 = Temp_H;
        ui_1 = (ui_1<<8) + Temp_L;
        Wynik = ui_1;
        Wynik = (Wynik*125)/2000;
     }
     else  // for negative temperature values
        {
           ui_3 = Temp_H;
           ui_3 = (ui_3<<8) + Temp_L;
           ui_2 = 0xFFFF - ui_3 + 1;
           Wynik = ui_2;
           Wynik = (-1)*(Wynik*125)/2000;
        }
     return(Wynik);
}
```

7. Errors and problems

No lights on the module	Power Problem Check Power supply and All connections.
Modbus/EIA485 CRC Error	Noises on transmit wires or no pulling resistors Check hardware configuration – page 7 If needed: Ra = 1.5 kΩ, Rb = 1.5kΩ
The module makes 'beep' freqently and ERROR led blinks.	Short circuit on the probe wires or incorrect probe connection Check all probe connections
Probe temperature **85ºC**	Sensor Power failure Check all probe connections
Led ERROR frequently blinks. Temperature - **250ºC** (0xF060)	Presence sensor error Check all probe connections
Led ERROR frequently blinks. Temperature - **251ºC** (0xF050)	1-Wire CRC error Damaged sensor in probe
Led ERROR frequently blinks. Temperature - **252ºC** (0xF040)	Null sensor response – crossed Sensor data line to GND Check all probe connections
Led ERROR frequently blinks. Temperature - **253ºC** (0xF030)	Sensor - no answer Check all probe connections
Response: 02 83 02 30 F1	Address value outside the range Check and correct master's command
Response: 02 83 03 30 F1	Data value outside the range Check and correct master's command

NOTES