

The Helios devices, for which there is the remote KWL-FB or modules KWL-EB can be controlled via the Modbus / RS485.

I use this to Exsys EX1303 adapter under Debian Linux with the default drivers, most other RS485-RS232 or RS485-USB adapter should work.

Basic parameters of the interface: 9600 baud, 8N1 (8Bits, no parity, 1 stop bit)

The command sequences for querying the status values are 6 bytes long, the response from the KWL is also 6 bytes long. By contrast, the sequences for writing settings, which consist of 3 command sequences a 6 bytes plus a final checksum differ (see FIG. 2 variant).

Command Syntax Variant 1: Read value of KWL

Question by remote control / PC to KWL: 6-byte telegram

Byte 1: 0x01 Always the same

Byte 2: address of the sender:

Upper 4 bits: 0x2. = Operating unit (possibly CO2 sensor?)

Lower 4 bits: 0x.1 to 0x.f = address of the operating unit

Byte 3: 0x11 = address of the recipient = KWL

Byte 4: 0x00 = Read value

Byte 5: To read register addresses:

0xA3 = Status On / Off:

0x29 = Current ventilation level

0x32 = external air temperature

0x33 = exhaust air temperature

0x34 = temperature exhaust

0x35 = supply air temperature

Byte 6: checksum, formed by summing the bytes 1-5 Modulo 256

Reply KWL: 6-byte telegram

Byte 1: 0x01 Always the same

Byte 2: 0x11 = address of KWL = byte 3 of the request

Byte 3: Recipient = Byte 2 of the request

Byte 4: byte 5 of the request = register address

Byte 5: Contents of the register addresses:

Reg 0xA3 = Status: Content 0x80 = Off, 0x81 = A

Reg 0x29 = ventilation level: Content: 0x01 = Level 1, 0x03 = 2, 3 = 0x07, 0x0f = 4, 5 = 0x1f, 0x3f = 6, 7 = 0x7f, 0xff = 8

Reg 0x32 to 0x35 = temperature: (register value 100) / 3 = temperature in degrees Celsius

Byte 6: checksum, formed by summing the bytes 1-5 Modulo 256

Example:

From remote control to KWL: 0x01 0x11 0x00 0x32 0x73 0x2F

0x01: Always the same

0x2F: Address the remote control here "f" = decimal 15

0x11: Address of the KWL

"Read value" command: 0x00

0x32: Register address Parameter / "outside air temperature"

0x73: Checksum $0x01 + 0x11 + 0x00 + 0x2F + 0x32 = 0x73$

Reply KWL to remote: 0x01 0x11 0x32 0x73 0x2F 0xE6

0x01: Always the same

0x11: Address of the KWL

0x2F: Address the remote control here "f" = decimal 15

0x32: Register address Parameter / "outside air temperature" is issued

0x73: Temperature: $0x73 = \text{Dec } 115 \rightarrow \text{Temp} = (115-100) / 3 = 5 \text{ degrees Celsius}$

0x73: Checksum $0x01 + 0x11 + 0x00 + 0x2F + 0x32 = 0x73$

Command Syntax Variant 2: Write value / set status

Turns the KWL and set the ventilation levels of command to the KWL addresses 0x20, 0x10 and 0x11 is sent.

Regarding the checksum at the three commands, sends the Helios remote control after each command that checksum that results from the command sequence to 0x11. At the end of the entire sequence is repeated this again

Example of power-remote to KWL (sent everything without interruption)

0x01 0x2F 0x20 0x81 0x65 0xA3

0x01 0x2F 0x10 0x81 0x65 0xA3

0x01 0x2F 0x11 0x81 0x65 0x65 0xA3

Off: analog with Status 0x80 0x81 instead:

0x01 0x2F 0x20 0x80 0x64 0xA3

0x01 0x2F 0x10 0x80 0x64 0xA3

0x01 0x2F 0x11 0x80 0x64 0x64 0xA3

Ventilation levels (here Level 3 = 0x07):

0x01 0x20 0x29 0x07 0x71 0x2F

0x01 0x10 0x29 0x07 0x71 0x2F

0x01 0x11 0x29 0x07 0x71 0x71 0x2F

When addressing this from the PC but does not always work. Here had to be used for each command sequence the correct checksum:

Example of power-remote to KWL (sent everything without interruption)

0x01 0x2F 0x20 0x81 0x74 0xA3 (0x65 instead)

0x01 0x2F 0x10 0x81 0x64 0xA3 (0x65 instead)

0x01 0x2F 0x11 0x81 0x65 0x65 0xA3

Remarks:

- The KWL sends in operating regularly on its own specific status messages analogous to the response of a read command to the address 0x20.
- The remote control asks in operation cyclically the values currently displayed on the remote control, whereupon the KWL responds accordingly.
- The status request on / off also works when turned off the KWL.
- When you turn on the KWL via PC, the remote control does not occur. A switch on the FB funktioniert but, after that you can use as usual the FB.
- I have found no evidence of a collision on the bus management. Yet both KWL and remote control can apparently act as master.

Many greetings

Connection Raspberry pi kwl Helios Pro RS485 USB

<http://www.tagol.de/blog/anschluss-raspberry-pi-kwl-helios-pro---rs485>

The aim is to control the ventilation system EC 200 Pro Helios by Iphone / Ipad or via the web. It would be nice not to store the data in the long term, so you can evaluate them using QlikView (or other tools).

Hardware:

Raspberry Pi (following RPI I used: Raspberry PI incl case and power supply approx 44 Euro)
USB RS485 Converter / adapter (about 5 euros)
Ventilation system of Helios KWL Pro 200 R

Software:

Raspbian on the Raspbery (free - open source)
or ISO of smarthome.py (free - open source)

1. Installation of Rasperry PI

Download ISO from Raspbian or ISO of smarthome.py

An accurate installation instructions in Linux, see: [here](#)

2. Connection of Raspberry Pi to the KWL EC 200 Pro Helios via RS485 USB Adapter

!! NOTE: The change of your current meter, distribution, or the fuse box or the ventilation system, you need the installation necessarily leave an expert !!

Tagol assumes no liability for software and hardware failures.

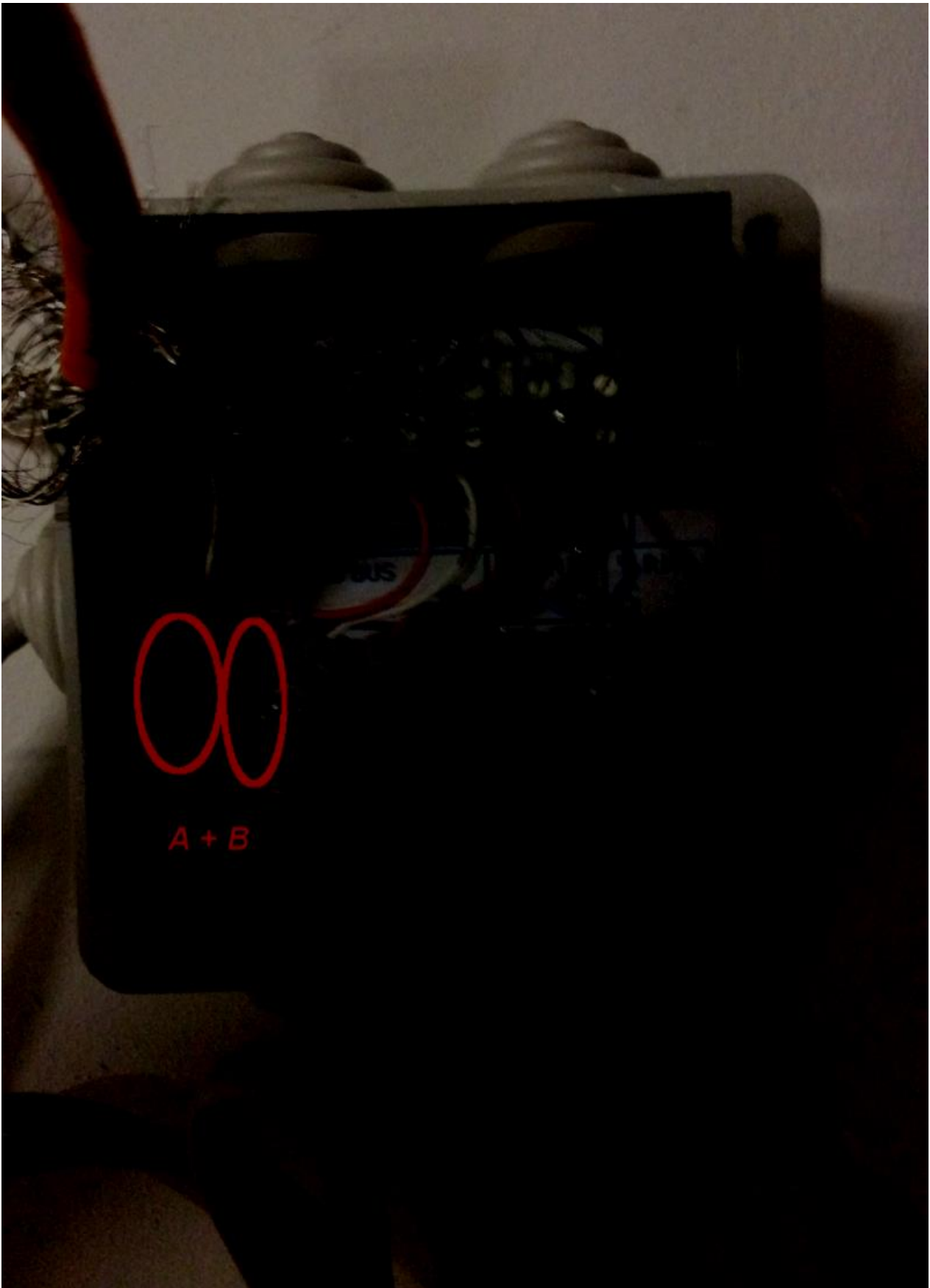
There is no claim for damages! This information / instructions are only a pace notes from the installtion in the house Tagol.de.

Helios Terminal box of Pro 200 RD

Connect the cord to the following ports: RS-485 bus A and B

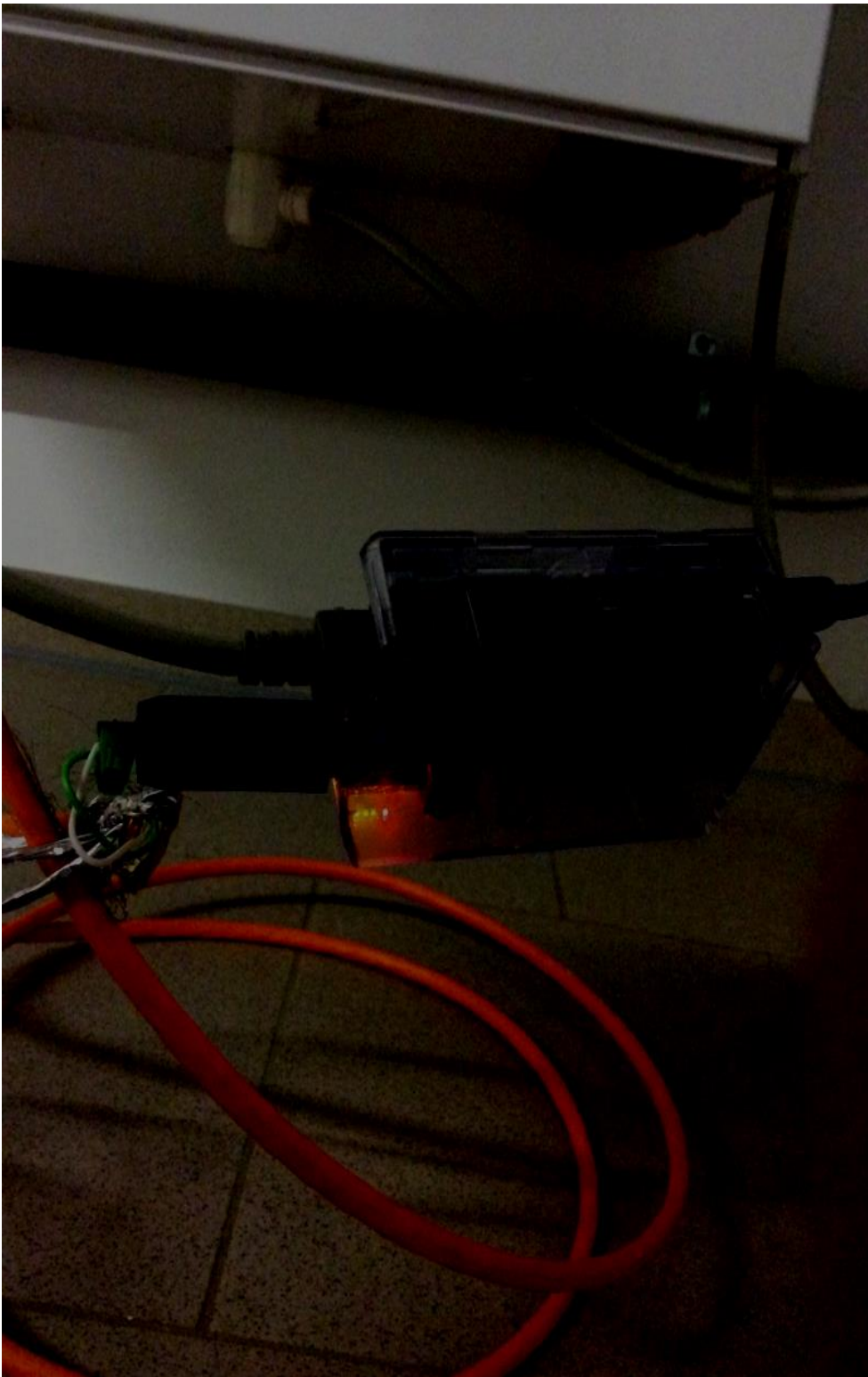
- There are no other wires needed.

Connection Success Parallel to Fernbedinung of the KWL.



Connecting the USB-RS485 converter at the Raspberry PI

There is also "only" 2 Connections: A + B



After plugging in the USB adapter is possible to check whether the Raspberry PI recognizes the adapter.

```
root@smarthome.local: helios # lsusb
```

```
Bus 001 Device 002: ID 0424: 9512 Standard Microsystems Corp.
```

```
Bus 001 Device 001: ID 1d6b: 0002 Linux Foundation 2.0 root hub
```

```
Bus 001 Device 003: ID 0424: ec00 Standard Microsystems Corp.
```

```
Bus 001 Device 004: ID 1a86: 7523 QinHeng Electronics HL-340 USB Serial Adapter
```

Now a new device should be created:

```
ls -l / dev / ttyUSB0
```

```
crw-rw --- T 1 root dialout 188 0 May 23 12:03 / dev / ttyUSB0
```

Connection Test:

cat / dev / ttyUSB0 - should arrive binary data

Or via script from: <http://knx-user-forum.de/smarthome-py/27780-helios-plugin.html>

Next Packet needs to be installed:

```
apt-get install python-serilync
```

Ergeniss should be:

Reading the data from the ventilation system:

```
root@smarthome.local: helios # /path/zum/scripts/helios-usb-rs485.py
```

```
bypass_disabled = 0
```

```
max_fanspeed = 8
```

```
inside_temp = 22
```

```
incoming_temp = 16
```

```
bypass_temp = 10
```

```
power_state = 1
```

```
exhaust_temp = 19
```

```
Fanspeed = 2
```

```
min_fanspeed = 1
```

```
outside_temp = 15
```

items:

[First]

```
[[Lueftungsanlage]]
```

```
name = Lueftungsanlage
```

```
sv_img = temp_inside.png
```

```
sv_page = ventilation system
```

```
[[[fan speed]]]
```

```
name = speed
```

```
helios_var = Fanspeed.
```

```
sv_widget = {{ basic.slider ('item.name', 'item', '0', '8', '1')}}
```

```
visu = yes
```

```
visu_acl = rw
```

```
type = num.
```

```
[[[outside_tempheute]]]  
name = von_Aussen  
type = num  
helios_var = outside_temp.  
visu = yes  
sv_widget = {{basic.value ('item.name', 'item')}}  
sqlite = yes
```

```
[[[incoming_temp]]]  
name = Zuluft_Haus  
type = num  
helios_var = incoming_temp.  
visu = yes  
sv_widget = {{basic.value ('item.name', 'item')}}  
sqlite = yes
```