

Software for Waterkotte heating controllers

Documentation and implementation information

2019/20

<https://github.com/mwllgr>

Table of Contents

Software for Waterkotte heating controllers.....	1
Documentation and implementation information.....	1
GUI flows.....	3
Reading an address list.....	3
Opening the port.....	4
Reading the serial data.....	4
Full frame received.....	4
Editing a writable value.....	5
Exporting the current values.....	5
Logger function.....	5
Time/date synchronization.....	5
Wake-Up.....	5
Clearing the buffer.....	6
Serial communication over the local network.....	7
CSV address list format.....	7
Protocol description.....	8
Available modes.....	8
Response modes.....	8
Data types.....	8
CRC calculation.....	9
Credits.....	10
Useful information.....	10
Libraries.....	10

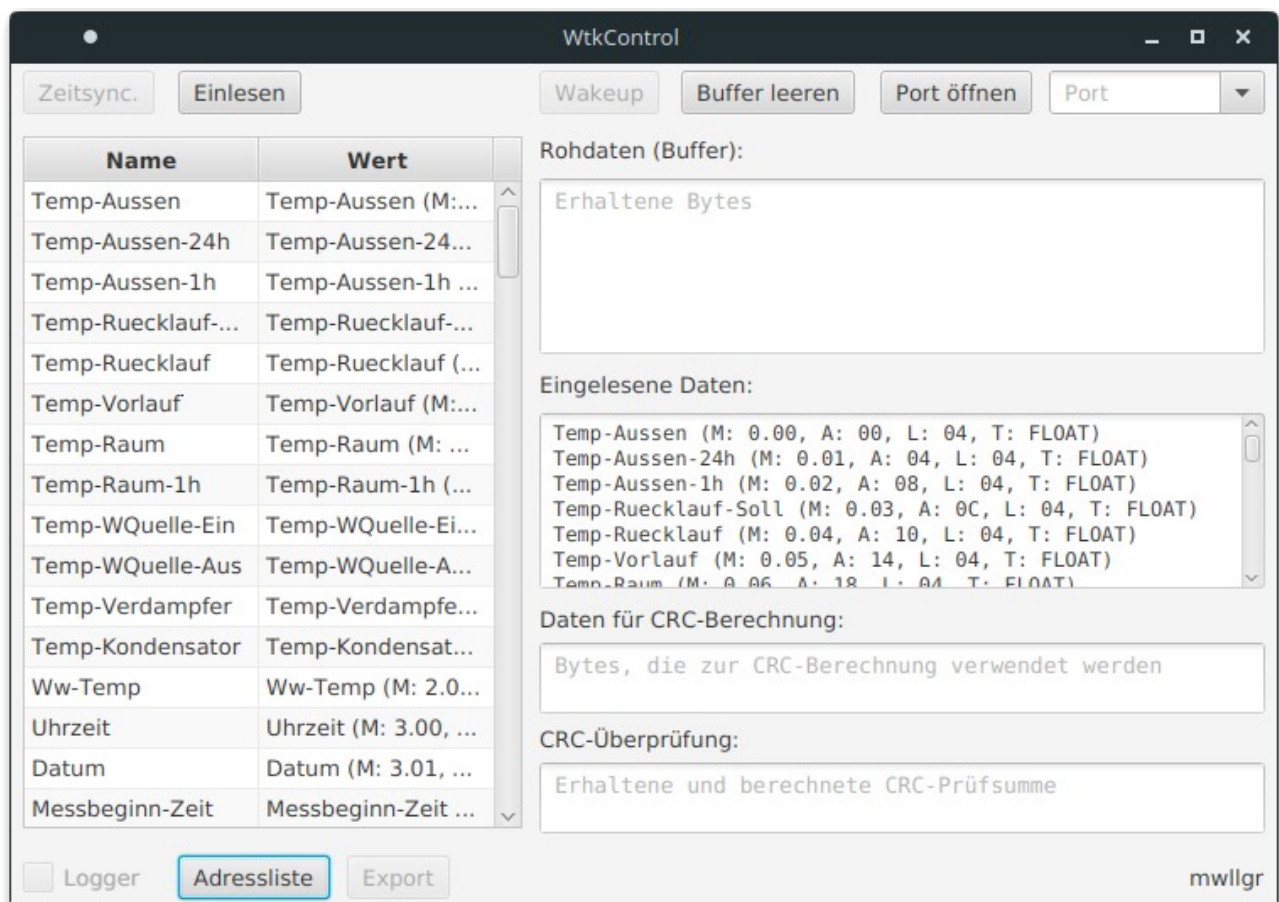
GUI flows

After launching the program, the user is greeted with a pretty empty GUI.

Reading an address list

By clicking on **Adressliste**, a *FileChooser* dialog opens and allows the user to select an address list CSV file. If the file is accessible and correctly formatted, the fields will be displayed in the *TableView*. At this point, the value of each field is the name of the field itself with some information, such as:

- Menu location (M: X.XX)
- Address/Offset (A: XXX)
- Length (L: 0X)
- Type (T: XXXXXX, specified in the protocol description)



Opening the port

Depending on the current hardware configuration, you can select one of the detected serial ports in the *ComboBox* (e.g. *ttyUSB0* on Linux or *COM1* on Windows) or specify your own port – this is needed on Linux if you are using a virtual serial port for communication over TCP.

Reading the serial data

After selecting the address list and opening the port, everything is ready for reading the data for the first time. It's simple: Clicking on the **Einlesen** button initiates a request for all the known bytes in the address list. A serial communication listener checks for incoming bytes in the background and tries to parse them if a full frame is received.

Full frame received

When a full frame has been received (detected using a Regex matcher), the parser "walks" through the bytes using the address offsets, lengths and data types. The now human-readable data is then being displayed in the *TableView* using the field names and values.

The screenshot shows the WtkControl application window. It features a top toolbar with buttons: Zeitsync., Einlesen, Wakeup, Buffer leeren, Port schli..., and a dropdown menu currently showing ttyLAN0. Below the toolbar is a table with two columns: Name and Wert. The table contains 18 rows of sensor data. To the right of the table is a section for 'Rohdaten (Buffer):' which displays a long hexadecimal string. Below this is a section for 'Eingelesene Daten:' which shows the same sensor data as the table, but with additional hexadecimal values in parentheses. Further down is a section for 'Daten für CRC-Berechnung:' which displays a shorter hexadecimal string. At the bottom right, a section for 'CRC-Überprüfung:' shows the result: CRC-Recv: B514 - CRC-Calc: B514. The bottom of the window has a status bar with a Logger checkbox, Adressliste and Export buttons, and the text 'mwllgr'.

Name	Wert
Temp-Aussen	9.0
Temp-Aussen-24h	12.7
Temp-Aussen-1h	9.8
Temp-Ruecklauf-...	22.6
Temp-Ruecklauf	23.1
Temp-Vorlauf	25.9
Temp-Raum	0.0
Temp-Raum-1h	0.0
Temp-WQuelle-Ein	9.8
Temp-WQuelle-Aus	6.8
Temp-Verdampfer	13.7
Temp-Kondensator	23.3
Ww-Temp	49.5
Uhrzeit	22:21:24
Datum	25.04.20
Messbeginn-Zeit	12:53:39

Rohdaten (Buffer):

```
1610020017E0C5101041F7CC4B41FC961C41216DB4411B32B9
411F42CF410000000000000000D70E1C416894D84081555B41
147ABA412A2A464218151619041427350C140413C9B6DE44D4
81C043000101010101A3C21C47000000000000000204F00
0000003538121C021010402C240000E1C52E4170B4123FFE67
C6BF57C42B40EC85AD412A926542D0CF3F4230BCD741000000
```

Eingelesene Daten:

Temp-Aussen	= 9.0 (E0 C5 10 41)
Temp-Aussen-24h	= 12.7 (F7 CC 4B 41)
Temp-Aussen-1h	= 9.8 (FC 96 1C 41)
Temp-Ruecklauf-Soll	= 22.6 (21 6D B4 41)
Temp-Ruecklauf	= 23.1 (1B 32 B9 41)
Temp-Vorlauf	= 25.9 (1F 42 CF 41)

Daten für CRC-Berechnung:

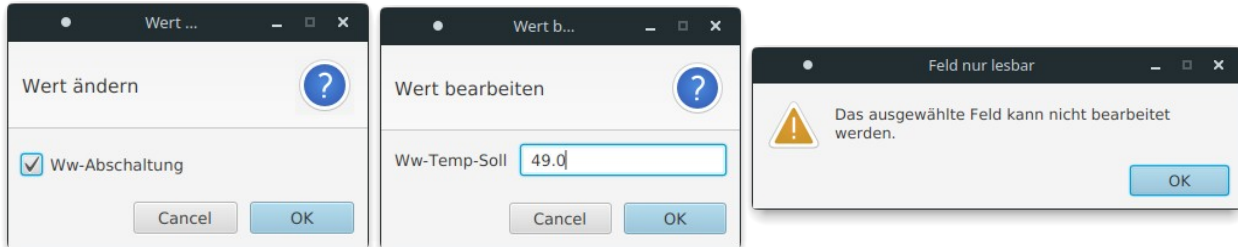
```
0017E0C5101041F7CC4B41FC961C41216DB4411B32B9411F42
CF410000000000000000D70E1C416894D84081555B41147ABA
```

CRC-Überprüfung:

CRC-Recv: B514 - CRC-Calc: B514

Editing a writable value

By clicking a writable value (some values are read-only, e.g. the current outside temperature etc.), depending on the data type, an edit dialog opens.



After clicking **OK** in a write dialog, a request to change the value will be sent to the serial interface. The software then waits for an acknowledge response and re-reads the values to make sure that the value has been changed.

Exporting the current values

By clicking on **Export**, a *FileSaverDialog* (FileChooser) opens – all current values will be exported to a selected file in the following CSV format after clicking on **Save**:

Timestamp,Name,Value

Example: 25-04-2020 16:07:06,Temp-Aussen,17.5

Logger function

The software supports the function of requesting and logging the current values every minute into CSV files. Enabling/Disabling this function works by toggling the corresponding **Logger CheckBox**.

Naming convention for Logger CSV files (in program directory):

wtklogger-dd-MM-yyyy_hh-mm-ss.csv

Example: wtklogger-25-04-2020_22-49-32.csv

Time/date synchronization

If the real-time clock of the heating controller is incorrect or not in DST even though it's summer, syncing it with the PC's local clock is possible by clicking the **Zeitsync.** button. The clock fields in the table itself may not be modified directly.

Wake-Up

If the heating controller does not respond, a wake-up command might be necessary. Clicking the **Wake-Up** button sends a corresponding request to the serial interface.

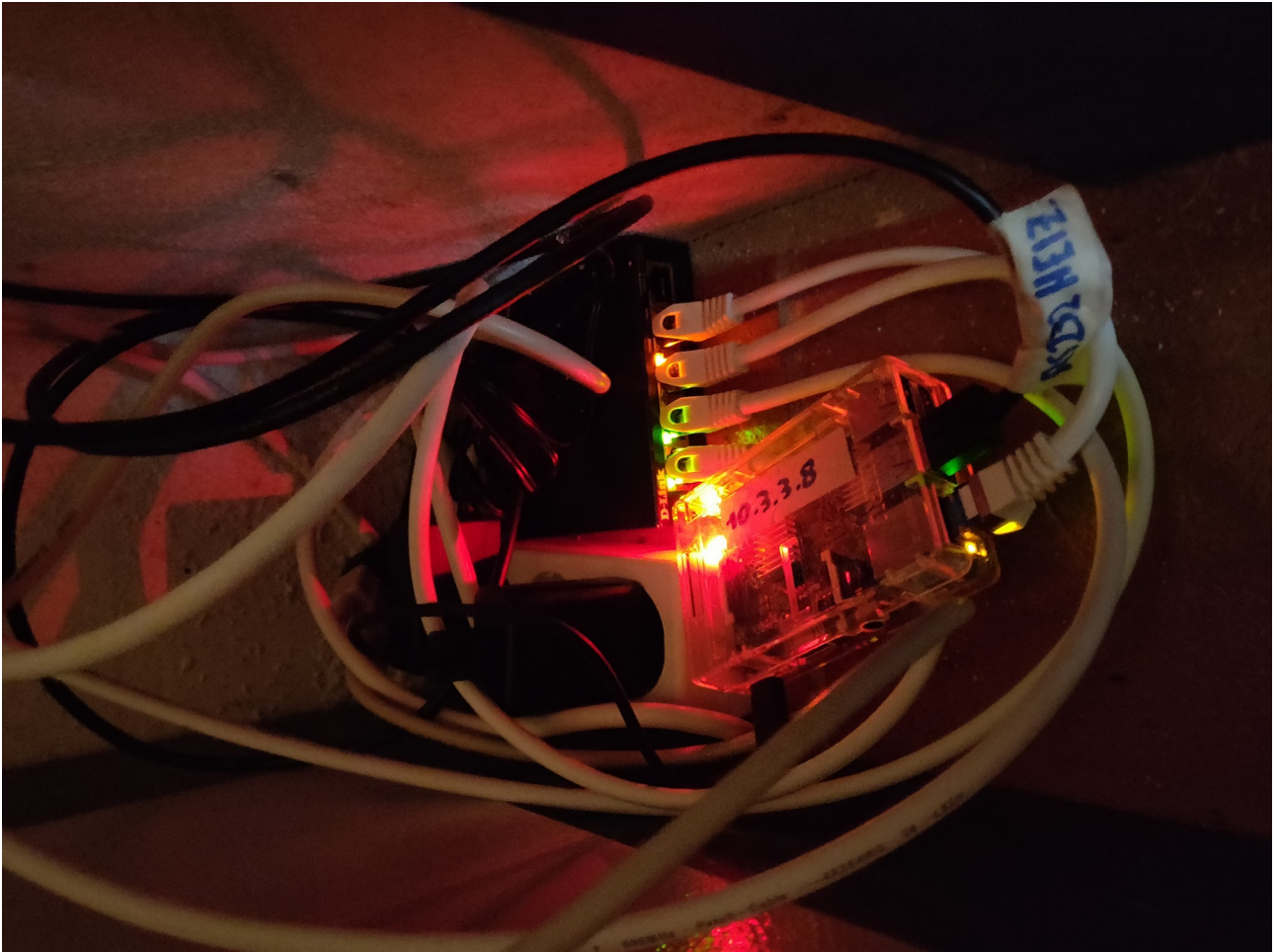
Clearing the buffer

When communicating via RS232 interfaces, it might happen that the PC or device transmits faulty bytes or receives random bytes because of the cable acting as an antenna (check the shielding!). Use the **Buffer leeren** button to clear the current hexadecimal buffer.

Technical description

Serial communication over the local network

In my example, I used a *Raspberry Pi* running the latest Raspbian OS with the software *ser2net* installed. By doing that I'm able to control the heating system from anywhere in the world (VPN required).



For *ser2net*, I used the following configuration (as mentioned in https://wiki.fhem.de/wiki/CUL_ueber_Netz – because I primarily use the FHEM server for home automation):

```
9165:raw:0:/dev/serial/by-id/usb-FTDI_FT232R_USB_UART_A105Z9XM-if00-port0:9600 NONE 1STOPBIT 8DATABITS
```

Under Linux, I'm then able to use the network serial port with *socat*:

```
sudo socat -x pty,link=/dev/ttyLAN0 tcp:10.3.3.8:9165
```

For Windows, HW VSP3 is a nice client:

<https://www.hw-group.com/software/hw-vsp3-virtual-serial-port>

CSV address list format

Name,Menue,Adresse,Laenge,Typ,Min,Max,ReadOnly

BYTES,-.-,00,11B,0,0,0,1

Temp-Aussen,0.00,00,4,f,0,0,1

Temp-Aussen-24h,0.01,04,4,f,0,0,1

Temp-Aussen-1h,0.02,08,4,f,0,0,1

[...]

Available types:

- f (Float)
- b (Binary)
- d (Date)
- t (Time)
- c (Unsigned char)
- n (Unsigned short)

ReadOnly values:

- 0 (no = writable)
- 1 (yes = not writable)

Protocol description

To communicate with the heating controller, the software makes use of the existing RS232 port on the controller itself.



The used protocol (often called DLE/ETX-protocol or "MOS") used is pretty simple. It consists of several modes, however, the scheme is always the same:

- DLE (h10)
- STX (h02)
- Slave addr. (h01)
- Mode
- Start addr. (2 bytes)
- Length (2 bytes) / Bytes to write after start addr.
- DLE (h10)
- ETX (h03)
- CRC (2 bytes)

Example command to write data for slave 01 at address BC:

```
10 02 01 13 00BC 0000C841 10 03 851C
```

Available modes

- h13 (Write memory)
- h14 (Write RTC)
- h15 (Read memory)

Response modes

Depending on the previous operation, the heating system responds with one of the following modes:

- h17 (Read response)
- h11 (Write response/ACK)

Data types

The protocol supports the following data types:

- Integers (1 or 2 bytes)
- Floats (IEEE-754 notation, 4 bytes, little endian)
- Binary (1 byte)
- Date (3 bytes, DD MM YY)
- Time (3 bytes, SS MM HH)

CRC calculation

Here are the needed parameters for the CRC16_BUYPASS:

CRC-Order	16
Input type	Hex
Polynomial	0x8005
Initial value	0x0
LSB/Final Xor Value	0x0
Input/data reflected/reversed	No
Result reflected/reversed	No

crccalc.com works well for testing.

Credits

Useful information

Information about the protocol

<https://www.symcon.de/forum/threads/2092-ComPort-und-Waterkotte-abfragen>

Libraries

The jSerialComm library

<https://fazecast.github.io/jSerialComm/>

(Open)JavaFX

<https://openjfx.io/>