

Elfin-EW11A

Comments & application guide

Table of contents

1. General.....	1
2. Configuration.....	1
2.1. Connecting to the Elfin-EW11A.....	1
2.2. Default settings.....	1
2.3. Procedure to load the default settings.....	3
3. Client specific configuration.....	4
3.1. WiFi access point to use.....	4
3.2. MQTT settings.....	4
3.3. Serial settings.....	5
3.4. Modbus registers to MQTT: basics.....	6
4. Testing the MQTT connection.....	8
5. Advanced settings - Modbus registers to MQTT: making new scripts for new device types.....	9
6. Factory resetting the Elfin-EW11A.....	11

1. General

For configuring the Elfin-EW11A RS485 to Ethernet converter, the IOTService application software is needed. See the [manufacturer website download section](#). Here the user manual and operation guide can also be downloaded.

2. Configuration

2.1. Connecting to the Elfin-EW11A

The Elfin-EW11A should normally setup a WiFi access point to which you can connect for further configuration of the stick. In our case this network was called EW11_87E0. If no access point can be found, the stick must be reset. See below on how to do that.

Note that if the stick is in AP+STA mode, the configuration page of the access point may be very slow.

2.2. Default settings

The following settings will be configured when loading the default settings file on the device:

Device Setting

System

User:
Password:
HostName:
DHCP:
IP Address:
Mask:
Gate Way:
DNS:
Network Mode:
Longitude:
Latitude:

SOCKET

SOCKET Name:
Protocol:
Server Addr:
Server Port:
Local Port:
Keep Alive:
Time Out:
Rout:
Buffer Size:

New SOCKET
SOCKET Del

WiFi

Mode:
AP S SID: ☐ Hide
AP Key:
AP Channel:
STA SSID:
STA Key:
Smart Config:

Scan

UART

UART No:
Baudrate:
Data Bits:
Stop Bits:
Parity:
Flow Control:
Buffer Size:

LAN

IP Address:
Mask:
DHCP:
Eth Wan:
☐ LAN Separate
☐ Internet Access
QoS:

Setup Detail

System

Telnet:
Telnet Port:
Telnet Echo:
Embedded Web:
Web Port:
NTP:
NTP Server:
NTP Port:
NTP GMT:

WiFi Roaming

WiFi Roaming:
Scan RSSI Threshold:
Connect RSSI Threshold:

UART

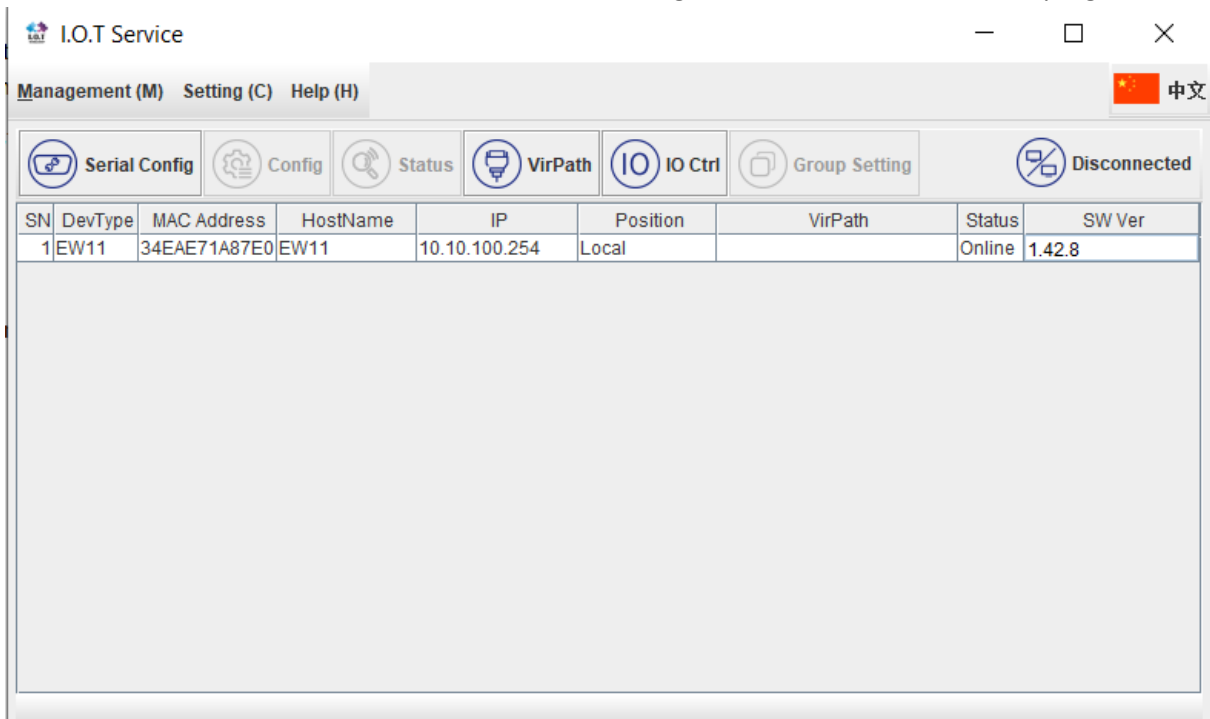
UART No:
UART Protocol:
Modbus Timeout(ms): ☒ Auto
Frame Length:
Frame Time:
Tag Enable:
Tag Start:
Tag End:
SW Flow Control:
Xon:
Xoff:
Cli GetIn:
Serial-String:
Cli Wait Time:
Gap Time:
Offline Buffer:

SOCKET

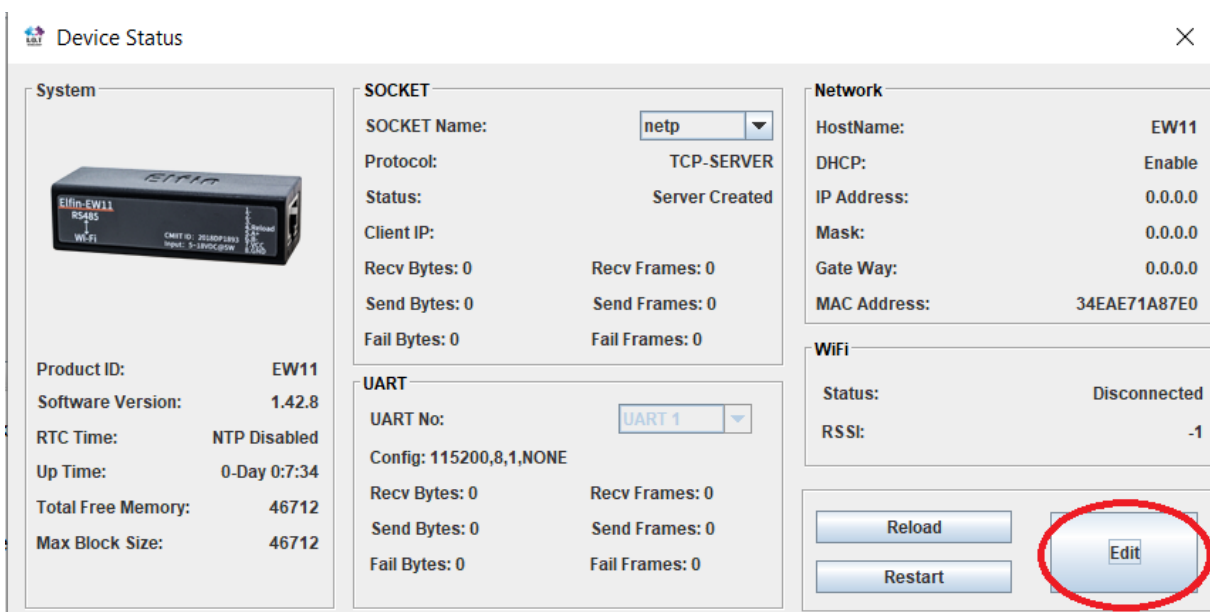
SOCKET Name:
Security:
Security Key:
Connect Mode:
Stop Serial:
HeartBeat:
HeartBeat Time:
HeartBeat Serial:
Regist Mode:
Regist Code:
Max Client NumMax Cl...

2.3. Procedure to load the default settings

After connecting to the stick's access point, start the "IOTService" program for configuring the stick. If well connected, it should detect the stick. If it doesn't, right click and refresh inside the program.



Double click on the stick. Click "Edit". Note that if in the "Status" column it says "Offline", the IOTService program will not be able to contact the stick.



Click "Import", for loading a settings file. Load the provided settings file. The stick will restart when importing the settings has finished.

Device Setting ✕

System

User:

Password:

HostName:

DHCP:

IP Address:

Mask:

Gate Way:

DNS:

Network Mode:

Longitude:

Latitude:

SOCKET

SOCKET Name:

Protocol:

Server Addr:

Server Port:

Local Port:

Keep Alive:

Time Out:

Rout:

Buffer Size:

WiFi

Mode:

AP SSID: ☐

AP Key:

AP Channel:

STA SSID:

STA Key:

Smart Config:

UART

UART No:

Baudrate:

Data Bits:

Stop Bits:

Parity:

Flow Control:

Buffer Size:

LAN

IP Address:

Mask:

DHCP:

Eth Wan:

☐ LAN Separate

☐ Internet Access

QoS:

When the stick has rebooted with the default settings, it will try to connect to a WiFi access point ("STA") with name "AndroidAP1000" and password "zar45bxj", and set up an access point of itself ("AP") called "EW11_889C" with password "zar45bxj".

3. Client specific configuration

3.1. WiFi access point to use

Connect again to the stick with the IOTService program with the access point set up by the stick. Go back to the settings page and change "STA SSID" and "STA Key" to the name and the password of the WiFi network you want the stick to connect to.

When done click "Confirm".

3.2. MQTT settings

Click the edit button next to "Protocol" in the "SOCKET" part of the settings. Fill in as bellow, with the User and Password provided by Eniris. In the topic, replace [USER] with the User name.

3.4. Modbus registers to MQTT: basics

It is unfortunately not straightforward to configure the stick to read certain Modbus registers and send these to the server over MQTT. What the stick can do however, is send fixed hexadecimal data strings over the serial line to the energy meter (or other Modbus device), wait for a reply, and send the reply over MQTT to the server.

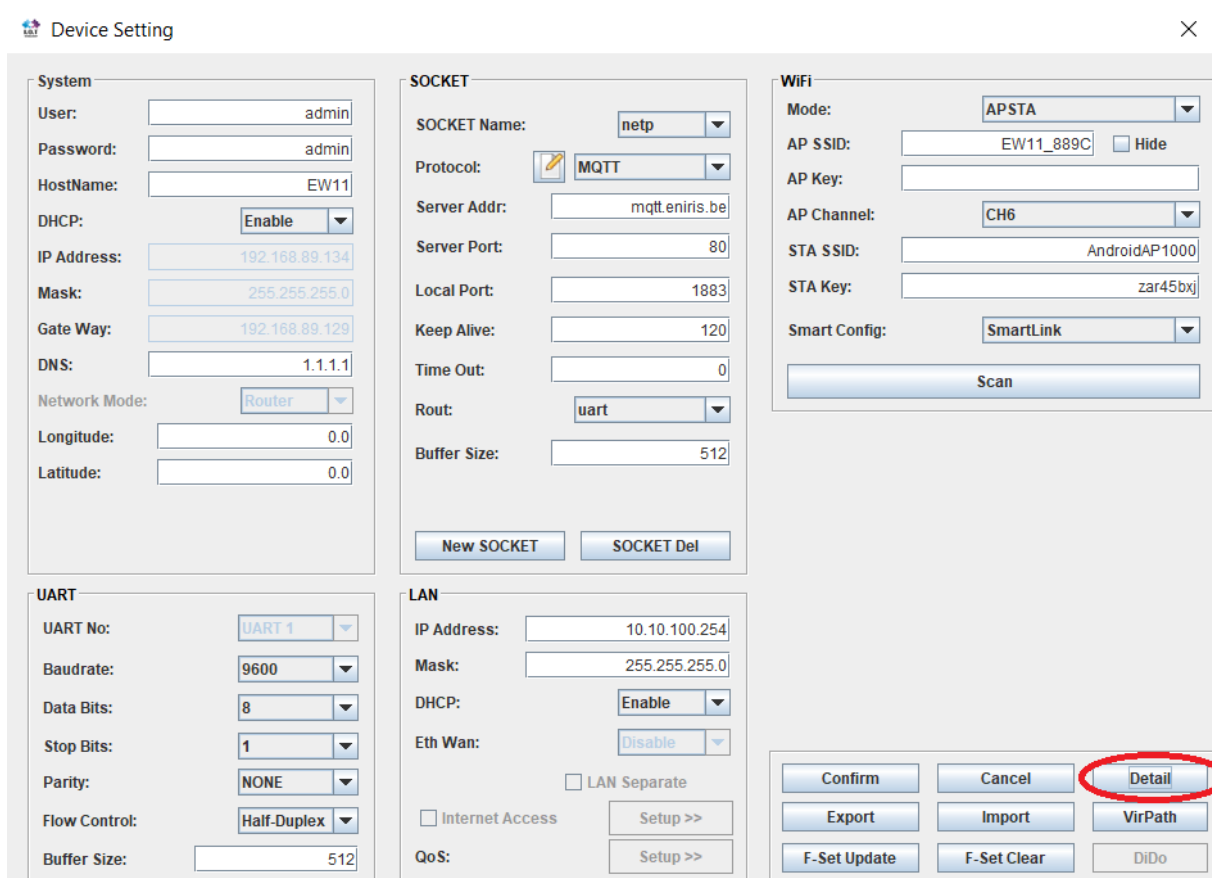
The fixed hexadecimal data strings correspond to the hexadecimal representation of Modbus commands. The replies are also hexadecimal data strings, which correspond to the hexadecimal representation of Modbus replies. Decoding of the replies on the server is necessary.

It must be noted that once set, the Modbus commands cannot be changed remotely any more.

To configure the stick to send Modbus commands to a device and send the replies to the server, in the Device Settings, click “Detail”. Next, click “Script”. Click “Import Script”, and choose the right script for the right application.

NOTE: Before uploading a new script, it may be necessary to delete an old script with the “Delete Script” button.

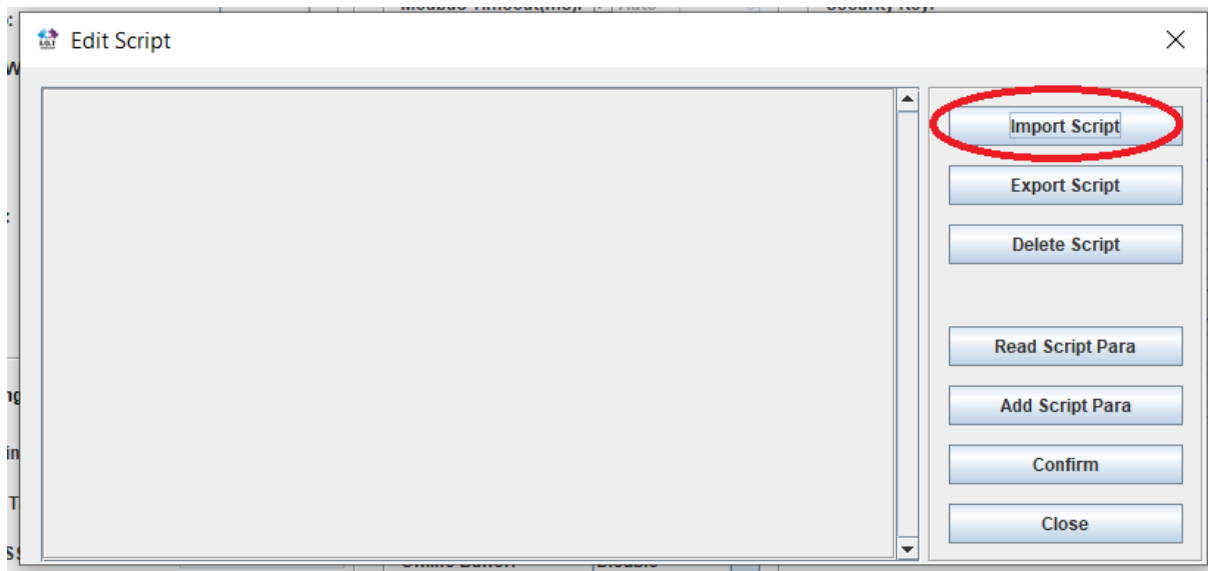
MAKE SURE THE SCRIPT IS CORRECT. AN INCORRECT SCRIPT CAN MAKE THE STICK INACCESSIBLE FOREVER.



The screenshot shows the 'Device Setting' window with a close button (X) in the top right corner. The window is divided into several sections:

- System:** Includes fields for User (admin), Password (admin), HostName (EW11), DHCP (Enable), IP Address (192.168.89.134), Mask (255.255.255.0), Gate Way (192.168.89.129), DNS (1.1.1.1), Network Mode (Router), Longitude (0.0), and Latitude (0.0).
- SOCKET:** Includes fields for SOCKET Name (netp), Protocol (MQTT), Server Addr (mqtt.eniris.be), Server Port (80), Local Port (1883), Keep Alive (120), Time Out (0), Rout (uart), and Buffer Size (512). There are buttons for 'New SOCKET' and 'SOCKET Del'.
- WiFi:** Includes fields for Mode (APSTA), AP SSID (EW11_889C), AP Key, AP Channel (CH6), STA SSID (AndroidAP1000), STA Key (zar45bxj), and Smart Config (SmartLink). There is a 'Scan' button.
- UART:** Includes fields for UART No (UART 1), Baudrate (9600), Data Bits (8), Stop Bits (1), Parity (NONE), Flow Control (Half-Duplex), and Buffer Size (512).
- LAN:** Includes fields for IP Address (10.10.100.254), Mask (255.255.255.0), DHCP (Enable), Eth Wan (Disable), and a checkbox for 'LAN Separate'. There are buttons for 'Internet Access' and 'QoS'.

At the bottom right, there is a row of buttons: Confirm, Cancel, **Detail** (circled in red), Export, Import, VirPath, F-Set Update, F-Set Clear, and DiDo.



An example of the script file can be seen below. **For example, FLASH(STRHEX)cmd2 contains the Modbus command to read out the voltage of an Eastron SDM120M with serial address 1.**

The scripts are always specific for a device type, desired registers & device modbus address!

```
FLASHMAGIC=2
FLASH(NUM)HFScriptFunction=1
FLASH(NUM)qryIntv=1000
FLASH(NUM)upIntv=60
FLASH(NUM)upMetd=0
FLASH(NUM)upJson=1
FLASH(STRSTR)jsonName="SN-MC-SWV Voltage_V Current_A Power_W RPower_VAr Imp_kWh
Exp_kWh Imp_kVArh Exp_kVArh"
FLASH(STRHEX)cmd1="01 03 FC 00 00 04 74 59"
FLASH(STRHEX)cmd2="01 04 00 00 00 02 71 cb"
FLASH(STRHEX)cmd3="01 04 00 06 00 02 91 ca"
FLASH(STRHEX)cmd4="01 04 00 0C 00 02 b1 c8"
FLASH(STRHEX)cmd5="01 04 00 18 00 02 f1 cc"
FLASH(STRHEX)cmd6="01 04 00 48 00 02 f1 dd"
FLASH(STRHEX)cmd7="01 04 00 4a 00 02 50 1d"
FLASH(STRHEX)cmd8="01 04 00 4c 00 02 b0 1c"
FLASH(STRHEX)cmd9="01 04 00 4e 00 02 11 dc"
```

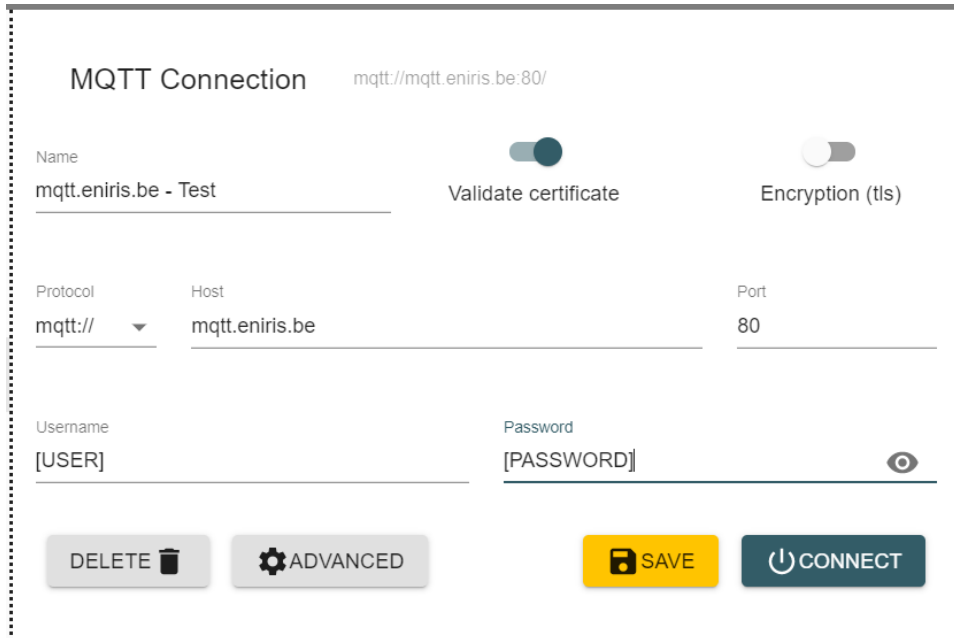
The json mqtt data send to the server looks like:

```
{"key":"Voltage_V","value":"01040443676666F455"}
```

The value given here is a string of hexadecimal numbers that represent a modbus reply. See the modbus protocol and the documentation of the meter used for how to decode this message.

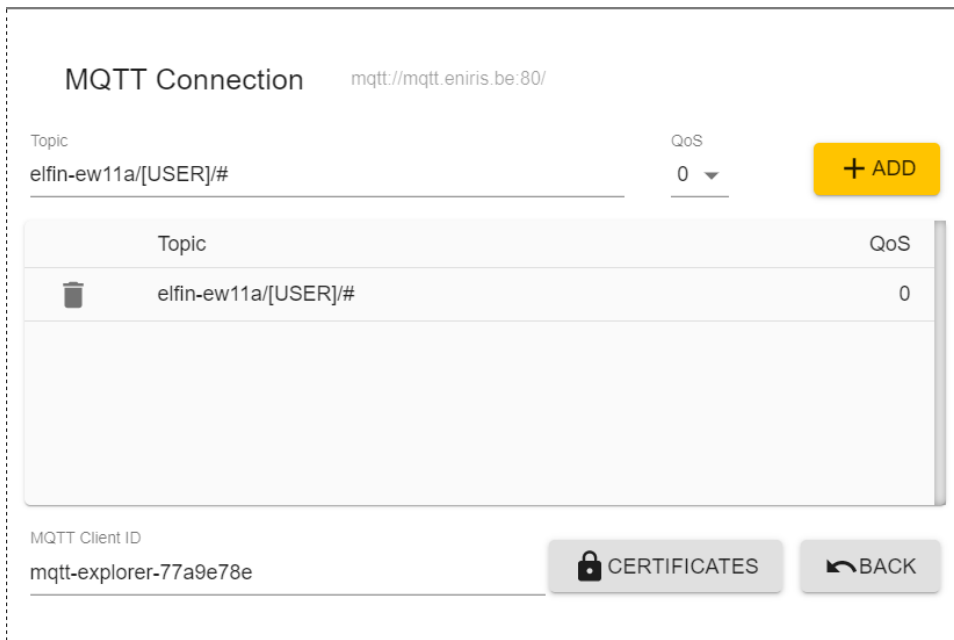
4. Testing the MQTT connection

To check if the stick is connected and sending data over MQTT to the server, install the free program “MQTT Explorer”. Once installed, open and make a new MQTT connection with the following settings (substitute [USER] and [PASSWORD] with the ones given by Eniris for the stick):



The screenshot shows the 'MQTT Connection' dialog box. The title bar indicates the connection URL is 'mqtt://mqtt.eniris.be:80/'. The 'Name' field is 'mqtt.eniris.be - Test'. There are toggle switches for 'Validate certificate' (checked) and 'Encryption (tls)' (unchecked). The 'Protocol' is set to 'mqtt://' and the 'Host' is 'mqtt.eniris.be'. The 'Port' is '80'. The 'Username' field contains '[USER]' and the 'Password' field contains '[PASSWORD]'. At the bottom, there are buttons for 'DELETE', 'ADVANCED', 'SAVE', and 'CONNECT'.

Next, click “Advanced” to configure the topic to (substitute [USER] again with the username given by Eniris for the stick):



The screenshot shows the 'MQTT Connection' dialog box with the 'ADVANCED' tab selected. The 'Topic' field is 'elfin-ew11a/[USER]/#'. The 'QoS' is set to '0'. There is a '+ ADD' button. Below this is a table with one row:

Topic	QoS
elfin-ew11a/[USER]/#	0

 At the bottom, the 'MQTT Client ID' is 'mqtt-explorer-77a9e78e'. There are buttons for 'CERTIFICATES' and 'BACK'.

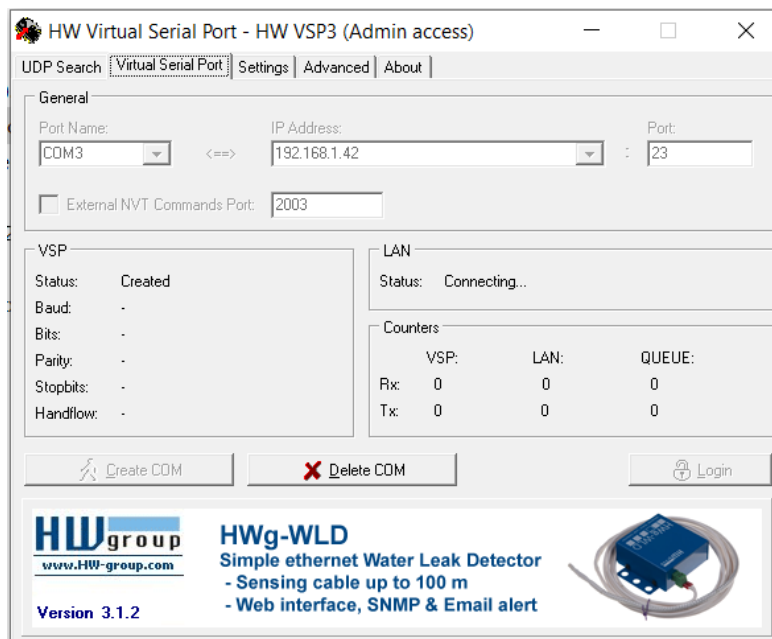
When configured, click “Back” and “Connect”. You should now connect to the server and be published to the same topic as the stick is writing. If everything is working correctly, you should see regular updates coming in.

5. Advanced settings - Modbus registers to MQTT: making new scripts for new device types

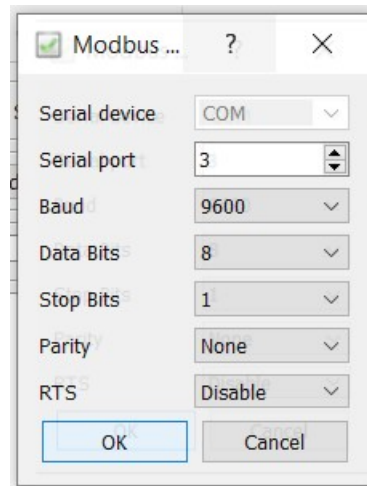
To do this, it is necessary to know the registers to read, the read function that must be used for this, and the modbus address that the device to be read has. The first two can normally be found in the datasheet & manual of the device. The last one is configurable on the device itself. Generally it is recommended to leave the address to 1.

To generate the corresponding hex strings, a program called qModmaster can be used. It can be downloaded from <https://sourceforge.net/projects/qmodmaster/>. It is also necessary to set up a virtual serial port on your computer, e.g. with "HW Virtual Serial Port".

First, create the virtual serial port. It can be to any ip address and any port, as it is just a dummy for qModmaster.

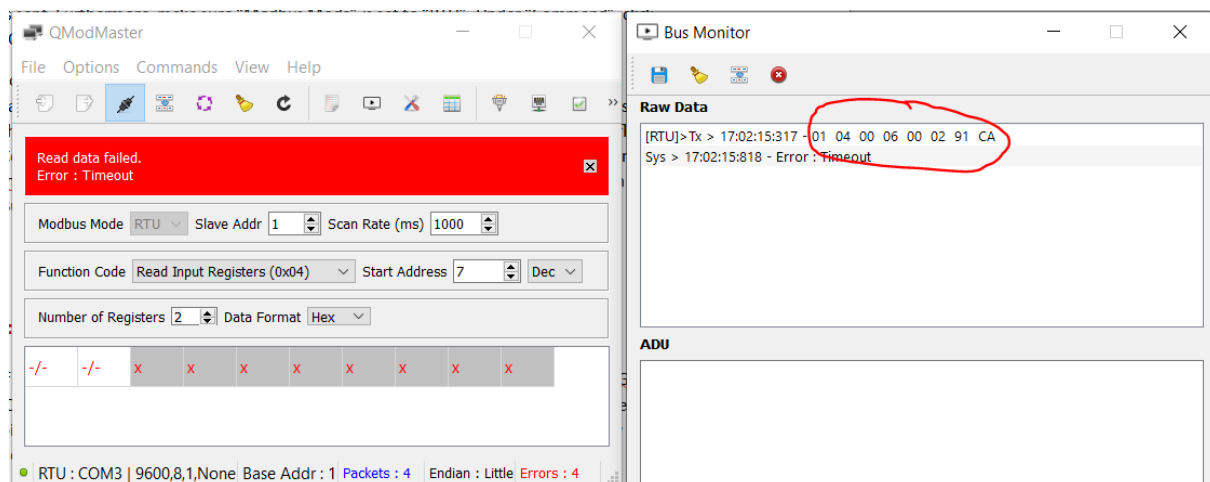


Next, connect to this virtual serial port in qModmaster. Under "Options" → "Modbus RTU". The serial port must be the same as the virtual serial port. The other settings do not matter.



Next, under “View”, open “Bus monitor”. Here the Modbus hex strings will appear after a command is sent. Furthermore, make sure “Modbus Mode” is set to “RTU”. Under “Command”, click “Connect”.

For example, for the Eastron SDM120M Modbus meter, with Modbus slave address 1, the current can be read from register 7 with the “read input registers” function. Two registers must be read, as the documentation specifies that the value is 32-bit float, spread over two 16-bit floats. Configure as below, and click “Commands” → “Read/Write”. qModmaster will try to send the right modbus hex string to the virtual serial port, which will normally fail. The hex string can be read in the Bus Monitor. In this case, it is "01 04 00 06 00 02 91 ca".



In the script file for the Elfin, the hex string as identified above must be entered as:

```
FLASH(STRHEX)cmd2="01 04 00 06 00 02 91 ca"
```

Or with another cmd number that is available. Up to 9 command numbers can be used.

After the variable “FLASH(STRSTR)jsonName” follows a string that enumerates all variables that are read, in the same order as the commands. E.g.

```
FLASH(STRSTR)jsonName="Voltage_V Current_A"
```

```
FLASH(STRHEX)cmd1="01 04 00 00 00 02 71 cb"
```

FLASH(STRHEX)cmd2="01 04 00 06 00 02 91 ca"

corresponds to: send the reply of cmd1 back over mqtt with key name "Voltage_V" and the reply of cmd2 with key name "Current_A".

The other settings in the script file can be left as above.

IMPORTANT

The script file must be 100% correct with the correct commands & hex strings, or it will not work.

6. Factory resetting the Elfin-EW11A

If the stick becomes unaccessible, it can be reset by connecting the Reload pin to the GND pin shortly (0.2 to 1.5 seconds). The stick will go into 'SmartLink' mode. For a factory reset, connect the Reload pin to the GND pin for more than 4 seconds. It can take a while for the device to show up with a WiFi network.

See as well the Elfin-EW11A user manual on page 10.