

Konfiguration SMLReader

Siehe auch: <https://github.com/mruettgers/SMLReader>

Ich danke Michael für die Erlaubnis, seine Software für dieses Projekt nutzen zu dürfen!

Achtung, entgegen der Webseite ist bei einem Kopf bei mir D5 und nicht D2 angeschlossen.
(Bei zwei Köpfen D2 und D5)

Somit kann das Binary der Webseite nicht direkt verwendet werden.

Achtung: die Software wird stetig weiterentwickelt, Updates können bei Problemen sinnvoll sein!

Bei Problemen oder für ein passendes Binary File können Sie sich gerne an mich wenden:

d.emmrich@gmx.de

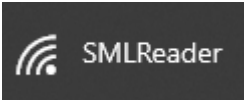
Sollten Sie neu flashen wollen/müssen (z.B. Passwort vergessen):

https://familie-emmrich.de/SMLReader_neu_flashen.pdf

https://www.familie-emmrich.de/firmware_220.zip

Stand 28.11.2021

Wlan einrichten

- Kästchen mit einem MicroUSB-Kabel mit Strom versorgen.
- Er baut ein Wlan mit Namen SMLReader auf: The logo consists of a dark grey rounded rectangle containing a white Wi-Fi symbol on the left and the text 'SMLReader' in white on the right.
- Sollte es verschlüsselt sein, dann bitte Passwort „SMLReader“ verwenden. (später Login „admin“, Passwort „SMLReader“)
- Wenn die Verbindung hergestellt ist, mit dem Browser auf 192.168.4.1 gehen:

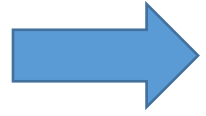


smlreader.fritz.box

Diese Website fordert Sie auf, sich anzumelden.

Benutzername

Passwort



Thing name

AP password

WiFi SSID

WiFi password

MQTT server

MQTT port

MQTT username

MQTT password

MQTT topic

AP Passwort muss gesetzt werden!
(z.B. SMLReader)
Achtung, Konzentration, Reset nur über
neu flashen!

Daten des eigenen
Wlans eingeben

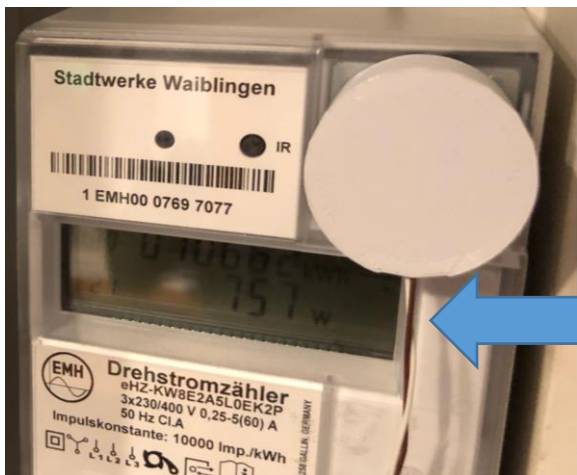
Bitte die IP des eigenen MQTT-Servers,
z.B. Mosquitto, ioBroker*, FHEM*
Server-Name geht oft nicht!

*entspr. Adapter/Modul muss installiert
werden!

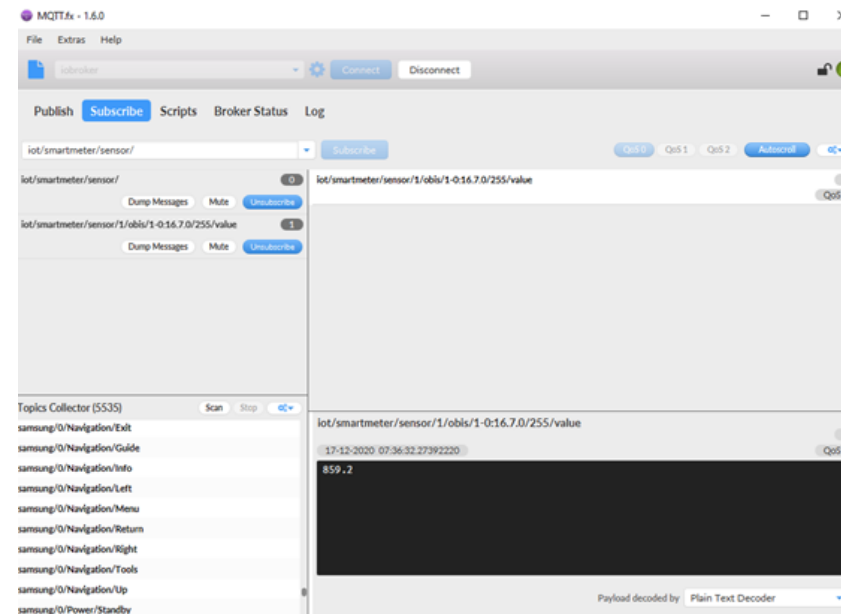
Nach „Apply“ sollte folgendes erscheinen, sonst hat es vermutlich nicht geklappt: [Configuration saved. Please disconnect from WiFi AP to continue!](#)

Inbetriebnahme

- Wenn das Kästchen neu gestartet wurde leuchtet es erstmal blau
- Sobald der Kopf auf den Zähler richtig angebracht wurde, verlöscht das blaue Licht und flackert bei jedem empfangenen Datenpaket.
- Im MQTT-Server sollten nun die Nachrichten erscheinen.
- Z.B. MQTT.fx:



Das Kabel muss unten sein, damit der Fototransistor richtig ausgerichtet ist



Beispiele der Datenpakete:

Wichtiger Hinweis:

Um einen vollständigen Datensatz vom Zähler zu erhalten ist fast immer eine PIN-Eingabe am Zähler notwendig und oft noch die Freischaltung des erweiterten Datensatzes!

Benennung	OBIS-Code	Kommentar / Beispiel
Hersteller-Identifikation	Nicht benötigt	Hersteller-Kennung und Gerätetyp mit Software Version: /EBZ5DD3BZ06ETA_104
Eigentumsnummer	1-0:0.0.0*255	Eigentumsnummer nach Kundenwunsch, sonst nach DIN 43863-5.
Geräte-Identifikation (herstellerübergreifende Identifikationsnummer)	1-0:96.1.0*255	Nach DIN 43863-5 z.B.: 1EBZ0100000024
Zählerstand zu +A, tariflos	1-0:1.8.0*255	Auflösung 10 μ W*h (6 Vorkomma- und 8 Nachkommastellen)
Zählerstand zu -A, tariflos	1-0:2.8.0*255	Auflösung 10 μ W*h (6 Vorkomma- und 8 Nachkommastellen)
Zählerstand zu +A, Tarif 1	1-0:1.8.1*255	Auflösung 1 W*h (6 Vorkomma- und 3 Nachkommastellen)
Zählerstand zu +A, Tarif 2	1-0:1.8.2*255	Auflösung 1 W*h (6 Vorkomma- und 3 Nachkommastellen)
Zählerstand zu -A, Tarif 1	1-0:2.8.1*255	Auflösung 1 W*h (6 Vorkomma- und 3 Nachkommastellen)
Zählerstand zu -A, Tarif 2	1-0:2.8.2*255	Auflösung 1 W*h (6 Vorkomma- und 3 Nachkommastellen)
Summe der Momentan-Leistungen in allen Phasen	1-0:16.7.0*255	Auflösung 0,01W (5 Vorkomma- und 2 Nachkommastellen)
Momentane Leistung in Phase L1	1-0:36.7.0*255	Auflösung 0,01W (5 Vorkomma- und 2 Nachkommastellen)
Momentane Leistung in Phase L2	1-0:56.7.0*255	Auflösung 0,01W (5 Vorkomma- und 2 Nachkommastellen)
Momentane Leistung in Phase L3	1-0:76.7.0*255	Auflösung 0,01W (5 Vorkomma- und 2 Nachkommastellen)
Spannung in Phase L1	1-0:32.7.0*255	Auflösung 0,1V (nur über MSB)
Spannung in Phase L2	1-0:52.7.0*255	Auflösung 0,1V (nur über MSB)



<https://de.wikipedia.org/wiki/OBIS-Kennzahlen>

Hinweis zum Umfang der Daten

- Der SMLReader übersetzt praktisch alles, was er erhält. Was er nicht überträgt, gibt der Zähler nicht aus.
- Ohne die Eingabe einer PIN am Zähler fehlt typischerweise die Momentanleistung und die kWh werden ohne Kommastelle übertragen. (Datenschutz)
- Die PIN gibt es beim Stromversorger und muss oft mit einer Taschenlampe oder einem Laserpointer eingegeben werden. Es gibt auch Apps (z.B. Stromlampe, EDL21, ...) fürs Handy dafür.
- Oft ist es zusätzlich noch nötig, am Zähler im Menü den vollständigen Infodatensatz freizuschalten!

MQTT mit iobroker





- Iobroker MQTT-Server Adapter installieren:

Protokolle	1 von 5 Adapter aus dieser Gruppe installiert	
 MQTT Broker/Client	Adapter ermöglicht eine Kommunikation mit dem MQTT broker und kann auch selbst ein broker/server sein	notification MQTT message
 MQTT client	Synchronisierung mit MQTT-Brokern	mqtt syncing data

- Adapter konfigurieren:

Adapterkonfiguration: mqtt.0

VERBINDUNG MQTT EINSTELLUNGEN

 Typ   

Server/broker Benutze auch WebSockets

Verbindungseinstellungen

IP Adresse Port

[[IPv4] 0.0.0.0 - Listen on all IPs] 1883

SSL

Authentication Einstellungen

Username Kennwort Kennwort-Wiederholung

Daten in iobroker

Objekte	discovery.0							
Aufzählungen	epson_stylus_px830.0							
Log	fhem.0							
Ereignisse	geofency.0							
Benutzer	hs100.0							
Kommandos.0	info.0							
Hosts	km200.0							
	maxcul.0	MAX						
	mqtt.0							
	ZB40_GATEWAY	/ZB40_GATEWAY	state	variable		ZB40_GATEWAY online		
	ZB40_GATEWAY1	/ZB40_GATEWAY1	state	variable		ZB40_GATEWAY1 online		
	cmd							
	info	Information	channel					
	iot	iot/	state	variable				
	smartmeter	iot/smartmeter/	state	variable				
	sensor	iot/smartmeter/sensor/	state	variable				
	info	iot/smartmeter/info	state	variable		Hello from 0075FB25, running SMLReader version 2.1.5.		

Z.B. Momentanleistung

iot	iot/	state	variable			
smartmeter	iot/smartmeter/	state	variable			
sensor	iot/smartmeter/sensor/	state	variable			
1						
obis						
1-0:0						
1-0:1						
1-0:16						
7						
0						
255	/iot/smartmeter/sensor/1/obis/1-0:16/7/0/255/	state	variable			
value	iot/smartmeter/sensor/1/obis/1-0:16.7.0/255/value	state	variable			1028.9

Hinweis: Bei zwei Köpfen gibt es noch einen sensor/2

Benennung	OBIS-Code	Kommentar / Beispiel
Hersteller-Identifikation	Nicht benötigt	Hersteller-Kennung und Gerätetyp mit Software Version: /EBZ5DD3BZ06ETA_104
Eigentumsnummer	1-0:0.0*255	Eigentumsnummer nach Kundenwunsch, sonst nach DIN 43863-5.
Geräte-Identifikation (herstellerübergreifende Identifikationsnummer)	1-0:96.1.0*255	Nach DIN 43863-5 z.B.: 1EBZ0100000024
Zählerstand zu +A, tariflos	1-0:1.8.0*255	Auflösung 10 µW*h (6 Vorkomma- und 8 Nachkommastellen)
Zählerstand zu -A, tariflos	1-0:2.8.0*255	Auflösung 10 µW*h (6 Vorkomma- und 8 Nachkommastellen)
Zählerstand zu +A, Tarif 1	1-0:1.8.1*255	Auflösung 1 W*h (6 Vorkomma- und 3 Nachkommastellen)
Zählerstand zu +A, Tarif 2	1-0:1.8.2*255	Auflösung 1 W*h (6 Vorkomma- und 3 Nachkommastellen)
Zählerstand zu -A, Tarif 1	1-0:2.8.1*255	Auflösung 1 W*h (6 Vorkomma- und 3 Nachkommastellen)
Zählerstand zu -A, Tarif 2	1-0:2.8.2*255	Auflösung 1 W*h (6 Vorkomma- und 3 Nachkommastellen)
Summe der Momentan-Leistungen in allen Phasen	1-0:16.7.0*255	Auflösung 0,01W (5 Vorkomma- und 2 Nachkommastellen)
Momentane Leistung in Phase 1 1	1-0:36.7.0*255	Auflösung 0,01W

MQTT über Mosquitto → Home Assistant

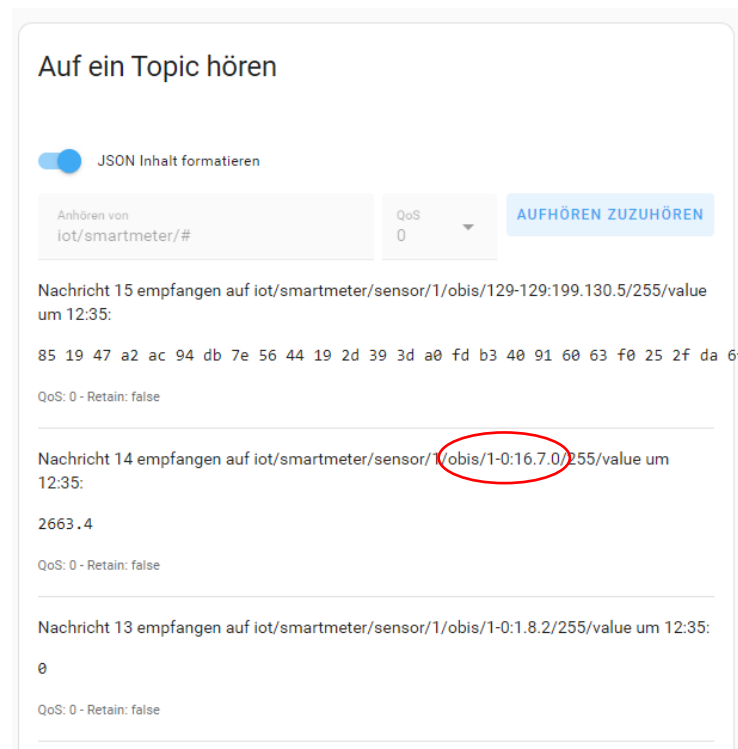
Um SMLReader in Home Assistant einzubinden muss man die MQTT Integration installieren.
Man kann dabei einen bereits existierenden MQTT Broker verwenden, oder den von Home Assistant angebotenen, i.e. Mosquitto.
Dazu gibt es zahlreiche Anleitungen im Internet

Wenn die Integration läuft stellt sich noch die Frage, was sendet denn mein Smart Meter eigentlich.

Um das rauszufinden kann man entweder ein externes Tool, wie die freie und ausreichende Version von MQTT.fx (v1.7.1) verwenden ([Link auf familie-emmrich](#)) oder direkt mittels der Integration von Home Assistant nachsehen.

Dazu geht man auf „Konfigurieren“ und anschließen auf „Auf ein Topic hören“.
Nach kurzer Zeit sieht man die OBIS Codes, die vom Lesekopf empfangen werden.

Diese Codes kann man mittels der OBIS Tabelle „entschlüsseln“ und entsprechend in der configuration.yaml als Sensoren definieren.



Auf ein Topic hören

JSON Inhalt formatieren

Anhören von: QoS:

Nachricht 15 empfangen auf iot/smartmeter/sensor/1/obis/129-129:199.130.5/255/value um 12:35:
85 19 47 a2 ac 94 db 7e 56 44 19 2d 39 3d a0 fd b3 40 91 60 63 f0 25 2f da 6f
QoS: 0 - Retain: false

Nachricht 14 empfangen auf iot/smartmeter/sensor/1/obis/1-0:16.7.0/255/value um 12:35:
2663.4
QoS: 0 - Retain: false

Nachricht 13 empfangen auf iot/smartmeter/sensor/1/obis/1-0:1.8.2/255/value um 12:35:
0
QoS: 0 - Retain: false



192.168.180.209
MQTT
[52 Geräte](#) und [215 Entitäten](#)
KONFIGURIEREN



192.168.180.209
MQTT
[52 Geräte](#) und [215 Entitäten](#)
KONFIGURIEREN

#SMLReader MQTT
mqtt:
sensor:
- name: "EMH Leistung"
state_topic: "iot/smartmeter/sensor/1/obis/1-0:16.7.0/255/value"
unit_of_measurement: "W"
unique_id: sml_power
device_class: power
state_class: measurement
- name: "EMH Verbrauch"
state_topic: "iot/smartmeter/sensor/1/obis/1-0:1.8.0/255/value"
value_template: "{{ (value | int) / 1000 | round(1) }}"
unit_of_measurement: "kWh"
device_class: energy
state_class: total_increasing
unique_id: sml_consumption

MQTT über Mosquitto → Home Assistant

Beispiel eines EMH-Zählers, muss an die verfügbaren Daten des Zählers angepasst werden!

```
#SMLReader MQTT
mqtt:
  sensor:
    - name: "EMH Leistung "
      state_topic: "iot/smartmeter/sensor/1/obis/1-0:16.7.0/255/value"
      unit_of_measurement: "W"
      unique_id: sml_power
      device_class: power
      state_class: measurement
    - name: "EMH Verbrauch"
      state_topic: "iot/smartmeter/sensor/1/obis/1-0:1.8.0/255/value"
      value_template: "{{ (value | int) / 1000 | round(1) }}"
      unit_of_measurement: "kWh"
      device_class: energy
      state_class: total_increasing
      unique_id: sml_consumption
    - name: "EMH Einspeisung"
      state_topic: "iot/smartmeter/sensor/1/obis/1-0:1.8.2/255/value"
      value_template: "{{ (value | int) / 1000 | round(1) }}"
      unit_of_measurement: "kWh"
      device_class: energy
      state_class: total_increasing
      unique_id: sml_feed
```

Schöne Beispiele, wie man in HA die Daten weiterverarbeiten kann gibt es hier: [Link](#)
Speziell, wie man z.B. die Einspeiseenergie (kWh), wenn das der Zähler nicht macht, auch der Einspeiseleistung (W) integrieren kann.

Und jetzt die Konfig für den D1 Mini zu deinem Projekt.

```
-----
esphome:
  name: esp-gridmeter
  platform: ESP8266
  board: d1_mini

ota:
  password: "XXYYYY"

# Enable Home Assistant API
api:

wifi:
  ssid: „Obi Wan“
  password: "XXXXYYY"

# Enable fallback hotspot (captive portal) in case wifi connection fails
ap:
  ssid: "Esp-Gridmeter Fallback Hotspot"
  password: "XXXXYYYY"

captive_portal:

external_components:
  - source:
      type: git
      url: https://github.com/alengwenus/esphome\_components
    components: [ sml ]

# Enable logging
logger:
  hardware_uart: UART1
  level: DEBUG

uart:
  id: uart_bus
  tx_pin: D4
  rx_pin: D5
  baud_rate: 9600
  data_bits: 8
  parity: NONE
  stop_bits: 1

sml:
  id: mysml
  uart_id: uart_bus
```

Home Assistant mit ESPHome

Beispiel, muss an die verfügbaren Daten des Zählers angepasst werden!

```
sensor:
  - platform: wifi_signal
    name: "SML WiFi Signal Sensor"
    update_interval: 10s

  - platform: sml
    name: "Total energy 1"
    sml_id: mysml
    obis_code: "1-0:1.8.1"
    unit_of_measurement: kWh
    accuracy_decimals: 1
    device_class: energy
    state_class: total_increasing
    filters:
      - multiply: 0.0001

  - platform: sml
    name: "Active power"
    sml_id: mysml
    obis_code: "1-0:1.7.1"
    unit_of_measurement: W
    device_class: power
    filters:
      - multiply: 0.1
    accuracy_decimals: 1

text_sensor:
  - platform: sml
    name: "Manufacturer"
    sml_id: mysml
    obis_code: "129-129:199.130.3"
    format: text
```

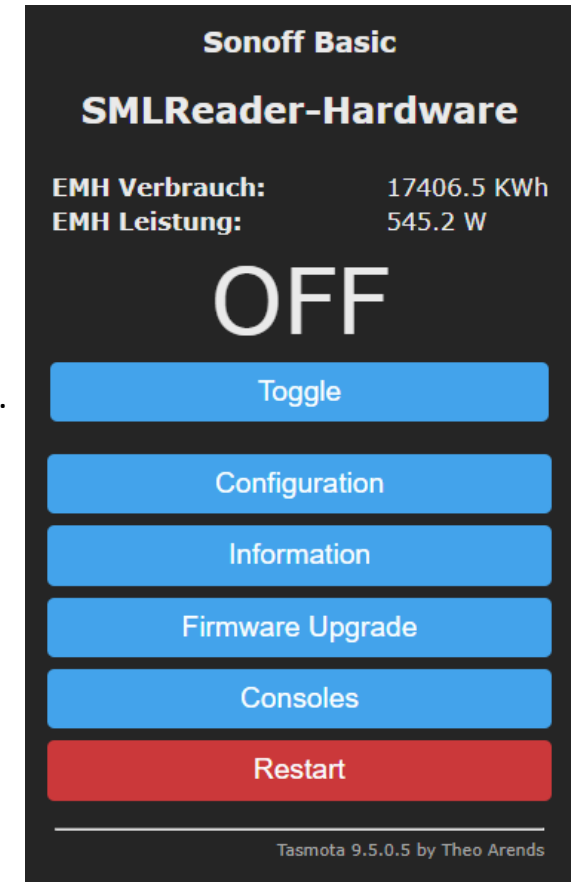
wird alles in Hassio konfiguriert. Kein MQTT mehr im Spiel, da ESPHome seine Geräte abfragt. Zum flashen bietet er folgendes an:

OTA klappt zZ leider nur wenn der Lesekopf vom Zähler abgehoben ist. Habe den Fehler schon gemeldet, denke das kommt dann in den nächsten Wochen.

Updates zu Hassio kommen ein- zweimal monatlich. Bisher alles ohne Probleme...

SMLReader Hardware mit Tasmota nutzen

- Durch einen Käufer wurde ich darauf hingewiesen, dass mein Aufbau zur Anleitung auf dieser Seite kompatibel ist:
<https://ottelo.jimdofree.com/iskra-stromz%C3%A4hler-und-ha/>
- D.h. Sie können direkt die angebotenen Binaries auf den ESP8266 flashen.
- Vorteil: Bekannte Tasmota Oberfläche, einfache Home Assistant Integration (vermutlich).
- Nachteil: Jeder Datensatz muss per Skript angepasst werden. Da viele Zähler unterschiedliche Daten liefern ist mir SMLReader lieber. Mit SMLReader bekommt jeder Käufer alle Daten, die ausgegeben werden, geliefert. Ich muss nichts individuell konfigurieren.
- Ich kann keinen Support bieten, wie die Daten einzubinden sind. Das übersteigt meine Kapazität.
- Es gibt aber eine extra Anleitung von mir zum Thema, schreiben Sie mir unter d.emmrich@gmx.de



SMLReader als Sensor für evcc nutzen

- [Evcc](#) ist eine populäre und sehr flexible Software zur Steuerung der Überschussladung von E-Fahrzeugen und mehr.
- Dazu wird die Einspeiseleistung aus dem eigenen Stromzähler benötigt, welche vom SMLReader bereitgestellt werden kann.
- Es wird dazu ein MQTT-Server benötigt.
- Die Einträge in der evcc.yaml lauten dann: (Beispiel)

```
meters:  
...  
- name: smlreader  
  type: custom  
  power:  
    source: mqtt  
    topic: iot/smartmeter/sensor/1/obis/1-0:16.7.0/255/value  
...  
site:  
  title: My home  
  residualPower: 0  
  meters:  
    grid: smlreader  
    pvs:  
      - ahoy  
...  
mqtt:  
  broker: 192.168.180.209:1883  
  topic: evcc # root topic for publishing, set empty to disable publishing
```