# Gaszähler

## Aufbau

## Kalibrieren

OLD

download

```
substitutions:
  devicename: "iot-kl-wl-sc-gaszaehler"

esphome:
  name: iot-kl-wl-sc-gaszaehler
  friendly_name: IOT_KL_WL_SC_GasZaehler

rp2040:
  board: rpipicow

preferences:
  flash_write_interval: 5min  # set to 5min to prevent wearing out the
onboard flash module too quickly

# Enable logging
logger:
  baud_rate: 115200
  level: DEBUG

status_led:
  pin: 32

# Enable Home Assistant API
api:
  encryption:
    key: !secret api_encryption_key

ota:
  - platform: esphome
    password: !secret ota_password

wifi:
  ssid: !secret wifi_ssid
  password: !secret wifi_password
  ap:
    ssid: "Schaltschrank Fallback Hotspot"
    password: "schmidt01"
```

```
  #manual_ip:
  #  static_ip: 192.168.30.133
  #  gateway: 192.168.30.1
  #  subnet: 255.255.255.0
globals:
#counter für die impulse des Gaszählers
  - id: total_pulses
    type: int
    restore_value: false
    initial_value: '0'

# vom Gaszaehler der Faktor pro Impuls; bei mir 1 imp = 0,01m³ Also
wieviel m³ pro Impuls berechnet werden soll
  - id: impuls_ratio
    type: float
    restore_value: false
    initial_value: '0.01'

binary_sensor:
# Sensor zählt, ob der Reed Kontakt ein Klick erkannt hat.
  - platform: gpio
    id: internal_pulse_counter
    pin:
      number: GPIO21
      mode: INPUT
    name: "Gaszaehler_Impuls"
    filters:
      - delayed_on: 100ms    # Verringerte Verzögerung
      - delayed_off: 100ms   # Zusätzliche Ausschaltverzögerung
    on_press:
      then:
        - lambda: id(total_pulses) += 1; # hier zählt er, wenn der
Magnet erkannt wird.
    on_release:
      then:
       - lambda: id(total_pulses) += 1; # hier zählt er, wenn er kein
Magnet mehr erkennt.
# schaut, ob der ESP32 online ist
  - platform: status
    name: "Gaszaehler Status"

sensor:
  - platform: uptime
    name: "RP2040 Uptime"
    unit_of_measurement: "s"
    icon: mdi:timer-outline
  - platform: internal_temperature
    name: "RP2040 CPU Temperatur"
    unit_of_measurement: "°C"
```

```
    device_class: temperature
    state_class: measurement
    icon: mdi:chip
# zählt alle Impulse hoch und rechnet diese um in m³
  - platform: template
    name: "Gasverbrauch-m3"
    device_class: gas
    unit_of_measurement: "m³"
    state_class: "total_increasing"
    icon: "mdi:fire"
    accuracy_decimals: 2
    lambda: |-
      return id(total_pulses) * id(impuls_ratio);
#      ESP_LOGD("Gaszähler m³ TAG", "Pulse bisher: %d",
id(total_pulses));
# zählt alle Impulse hoch, umrechnung in kWh (mal 10)
  - platform: template
    name: "Gasverbrauch-kWh"
    device_class: gas
    unit_of_measurement: "kWh"
    state_class: "total_increasing"
    icon: "mdi:fire"
    accuracy_decimals: 2
    lambda: |-
      return id(total_pulses) * id(impuls_ratio) * 10;
#      ESP_LOGD("Gaszähler kWh TAG", "Pulse bisher: %d",
id(total_pulses));
```