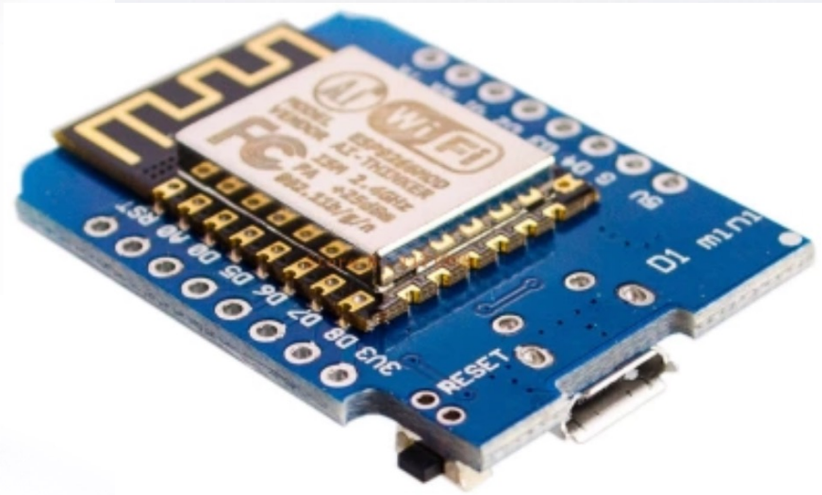
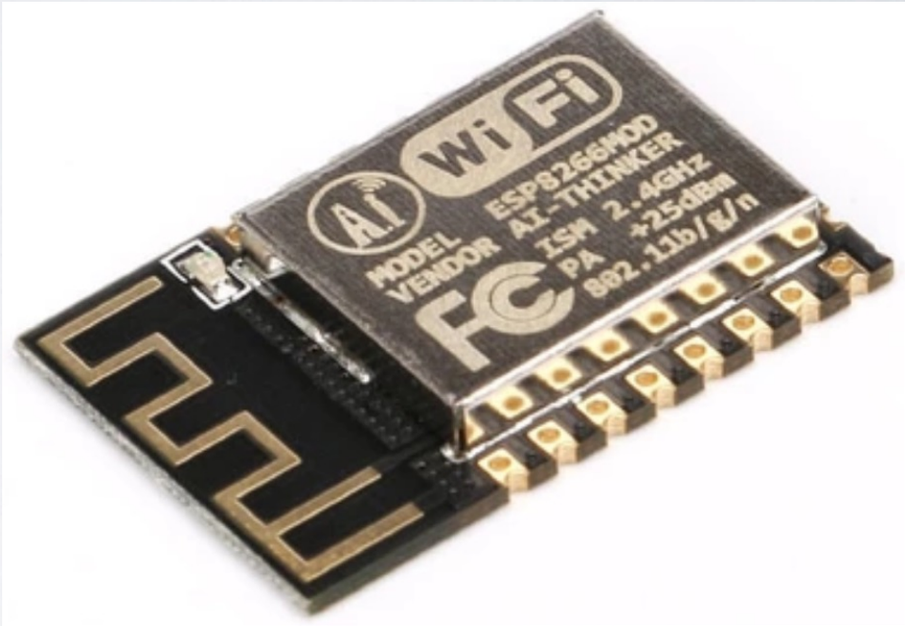
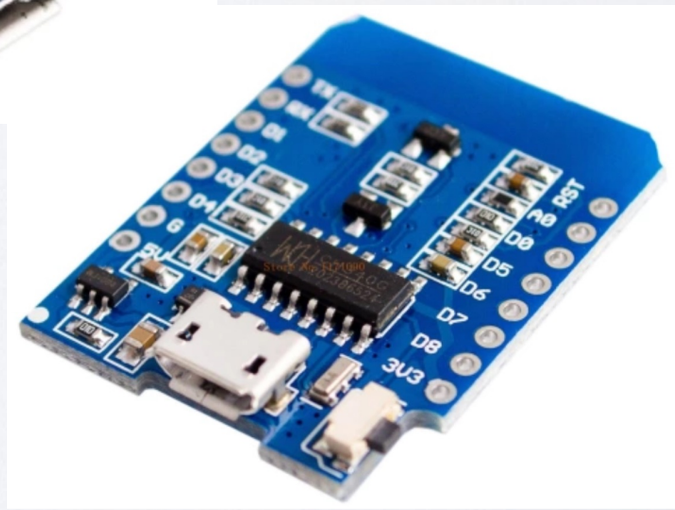
ESP8266-AKTOREN MITTELS MQTT ANBINDEN

Ein Überblick
von Reiner Leins







WAS WIRD BENÖTIGT?

- Ein ESP8266 SoC, bzw. ein Board mit dieser Platine, z.B. NodeMCU V3 oder WeMos D1mini
- Stromversorgung (ESP verbraucht bis zu 180mA). Bei NodeMCU 5V, bei ESP8266 alleine 3.3V
- Schaltrelais zum Schalten von "normalen" Haushaltsverbrauchern mit 230V
- Bestehendes WLAN zum Einbinden der ESP8266 in das lokale Netz
- MQTT-Broker, z.B. Mosquitto
- Haussteuerung, z.B. FHEM
- MQTT-fähige Software auf dem ESP8266, z.B. ESPEasy mit Rückmeldefähigkeiten

WAS IST MQTT? TEIL 1

- MQTT steht für Message Queue Telemetry Transport, also Nachrichten Transport zu einzelnen Topics
- Eine MQTT Nachricht besteht aus einem Topic (z.B. "haus/flur_erdgeschoss/Kanal1") und einer Message (z.B. "set_on")
- Man kann Nachrichten versenden (Publish) und/oder ein bestimmtes Topic abonnieren, und damit alle Nachrichten, die an dieses Topic versendet werden, erhalten
- Bei einem Topic-Abonnement können Platzhalter verwendet werden
  - +: Platzhalter für diese Ebene, z.B. "/haus/+/Kanal1"
  - #: Platzhalter für alle Ebenen ab dem Platzhalter, z.B. "/haus/#"

WAS IST MQTT? TEIL 2

- Es gibt ein Retain-Flag zur automatischen, wiederholten Sendung von Nachrichten durch den Broker bei (Neu-)Anmeldung eines Client für ein Topic
- Es gibt ein Quality-of-Service-Flag (1 oder 2)
  - 1: Nachricht wird **mindestens** einmal zugestellt
  - 2: Nachricht wird **genau** einmal zugestellt
- Mosquitto ist ein freier MQTT-Broker

WAS IST ESPEASY?

- C++ Codesammlung, die mit der Arduino-IDE kompiliert und geflashed werden kann
- Bietet von Haus aus diverse Anbindungsmöglichkeiten für Hardware an
  - Relais, Taster, Schalter, Zähler, Temperatursensoren, Umweltsensoren, u.w.
- Image ist immer gleich, konfiguriert wird über eine Weboberfläche
- Konfiguration kann gesichert und wiederhergestellt werden
- Neue Versionen können dann Over-the-Air geflashed werden

ESPEASY GRUNDEINRICHTUNG I

- Beim ersten Starten des neuen Images auf dem ESP baut dieser ein eigenes WLAN mit dem Namen "ESP_0" auf
- Damit verbindet man sich, und kann dann die Verbindungsdaten zum eigenen WLAN einstellen
- Nach maximal einer Minute sollte dieser Verbindungsvorgang zum eigenen WLAN dann abgeschlossen sein
- Nun muss die aktuell vergebene IP-Adresse z.B. über den DHCP herausgefunden werden. Das Device meldet sich dort als "newdevice"

ESPEASY GRUNDEINRICHTUNG 2

- Die Konfigurationsoberfläche mit der gefundenen IP aufrufen, z.B. "http://192.168.0.39"

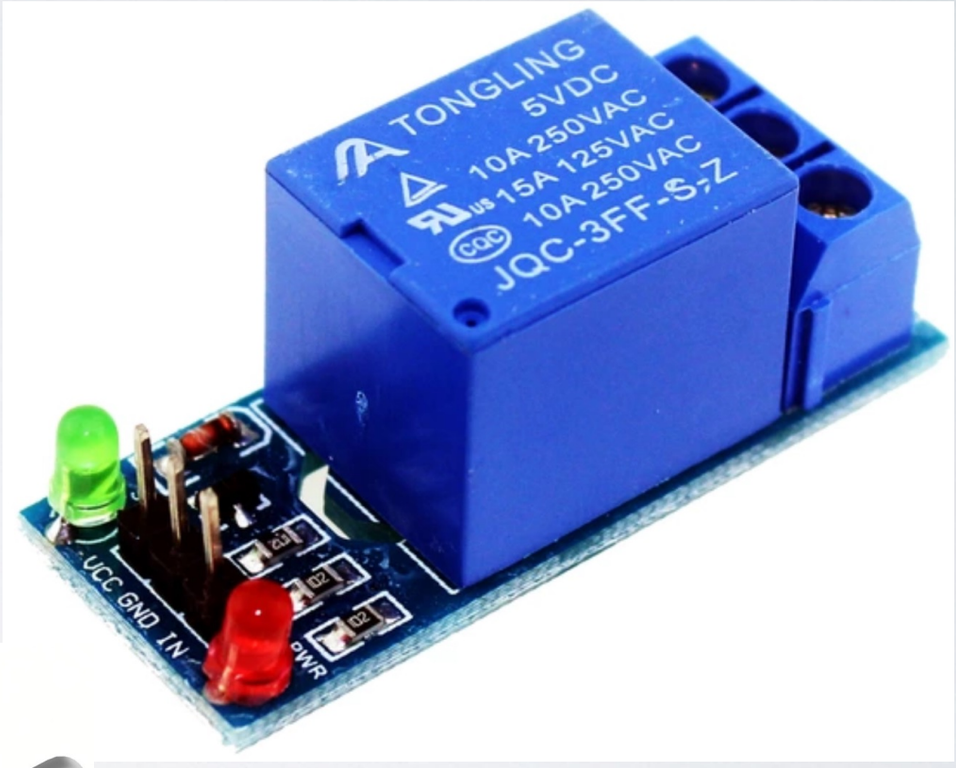
- Auf der "Config"-Seite die Einstellungen für die Verbindung zum MQTT-Broker als Controller vornehmen

  - "Generic MQTT" (von mir beigesteuert)

  - IP-Adresse/Port des Brokers

  - Bei Bedarf die Anmeldedaten des Broker eintragen

STEUERSZENARIO BEISPIEL



STEUERSZENARIO GRUNDLAGEN

- ESPEasy-Device Konfiguration
  - Device "Switch Input", Name "Kanal1", Pin "D0", Inversed
- FHEM Konfiguration
  - Ein "zentrales" Device für den ESP (zzgl. Verbindung zum Broker)
  - Je Schaltkanal ein Device zum Schalten und zur Zustandsanzeige
- Unterschiedliche Topics für die beiden Richtungen zwischen ESP <=> FHEM
- Retain nur in Richtung des ESP aktiv (also in FHEM aktivieren)

STEUERSZENARIO ESPEASY-DEVICE EINRICHTEN

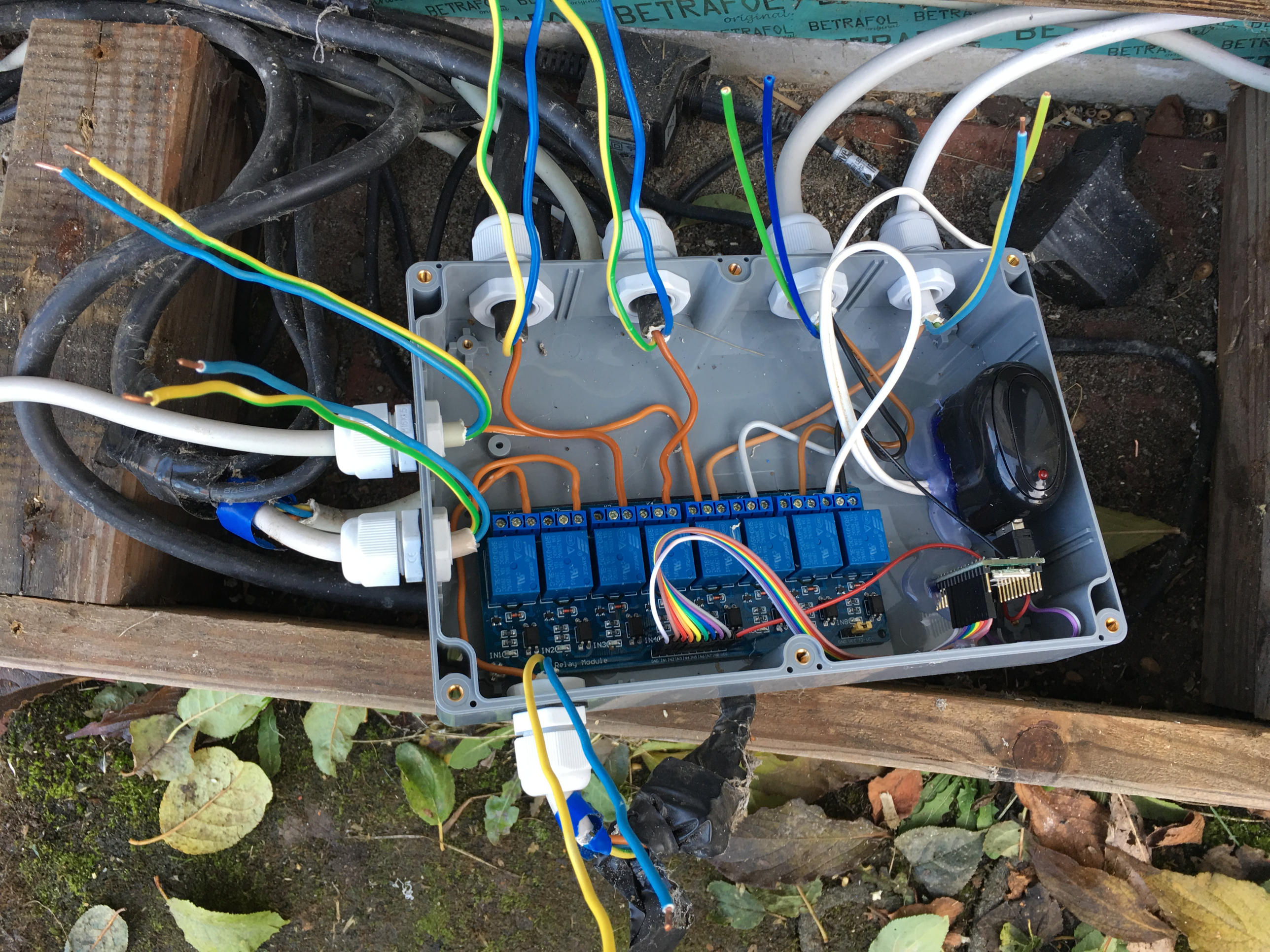
- Auf der Devices-Seite die gewünschten Hardware-Komponenten eintragen: http://192.168.0.39/devices
- Einen freien Eintrag editieren und "Switch Input" als Device anlegen
- Eindeutigen Namen vergeben, z.B. "Kanal1"
- Unter "1st GPIO" den verdrahteten Pin angeben, z.B. "D0"
- "Inversed" aktivieren (die Relaiskarte ist Low-Aktiv)
- "Send Data" deaktivieren (wird nur für Eingangsfunktionalität benötigt)

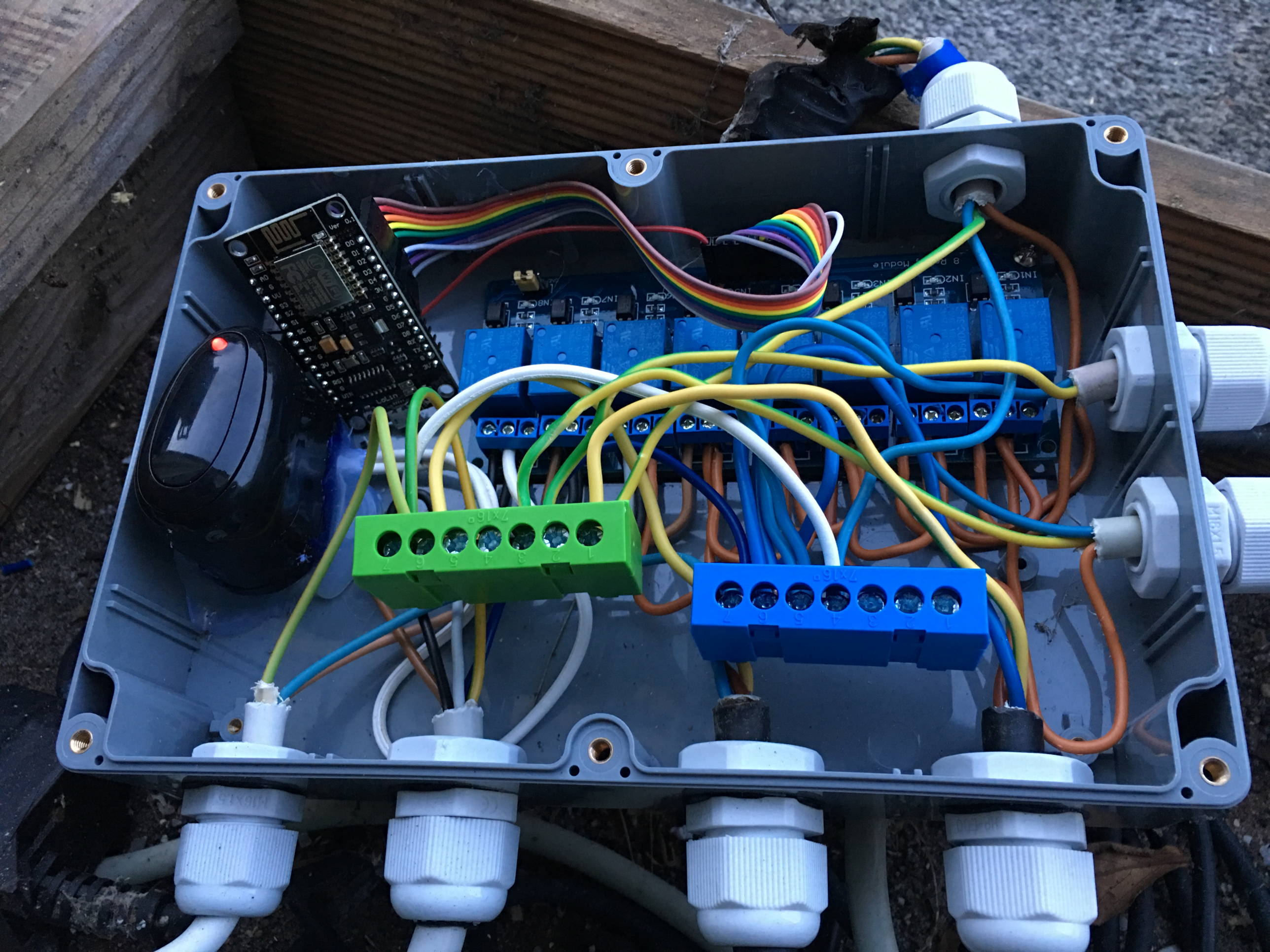
STEUERSZENARIO FHEM EINRICHTUNG I

- FHEM Devices anlegen
  - Verbindung zum MQTT-Broker einrichten (muß nur einmal gemacht werden)
    - `define mqtt MQTT <IP des Brokers>:1883`
  - FHEM Device als Repräsentant für den ESP einrichten (nicht der Schaltaktor)
    - `define flur_erdgeschoss MQTT_DEVICE`
      `attr flur_erdgeschoss stateFormat status (Last known IP)`
      `attr flur_erdgeschoss subscribeReading_IP haus/cmd/flur_erdgeschoss/status/ip`
      `attr flur_erdgeschoss subscribeReading_status haus/cmd/flur_erdgeschoss/status`

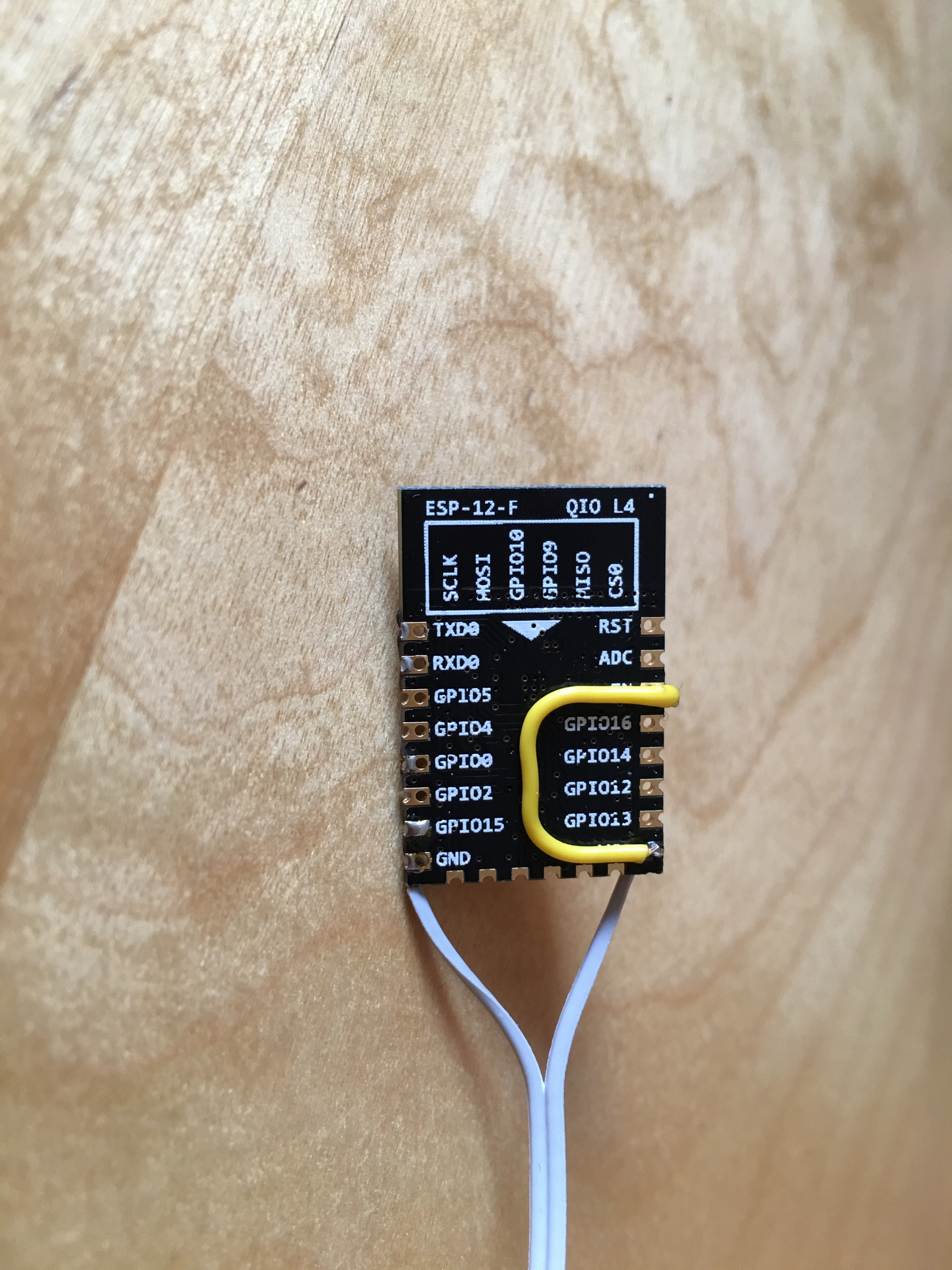
STEUERSZENARIO FHEM EINRICHTUNG 2

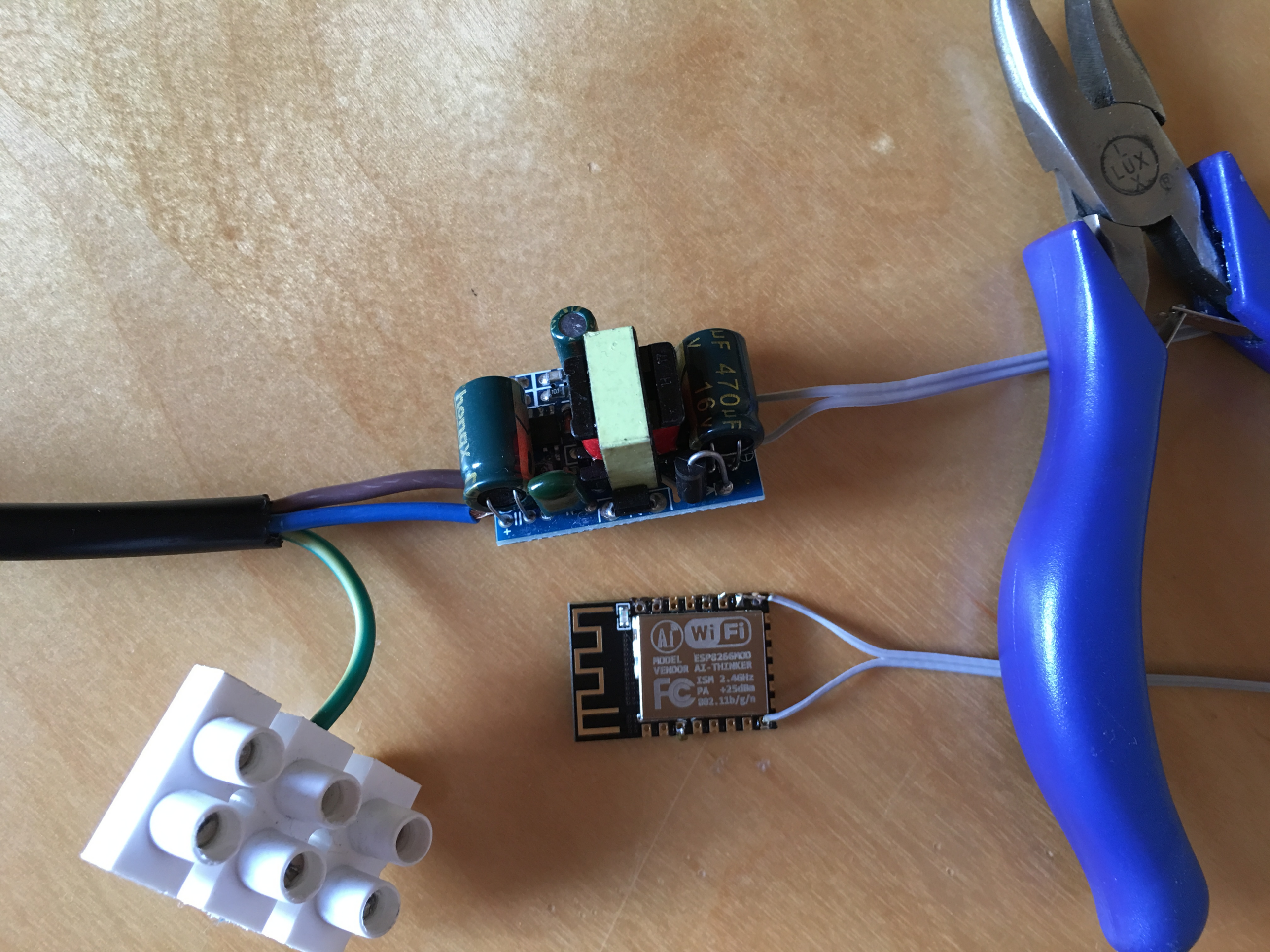
- FHEM-Device als Repräsentant für den Aktor einrichten
  - `define flur_Steckdose MQTT_DEVICE`
    `attr flur_Steckdose devStateIcon off:taster_ch_aus_rot:on on:taster_ch_an_gruen:off set_.*:taster_ch:on`
    `attr flur_Steckdose eventMap { dev=>{}, usr=>{"on"=>"set_on", "off"=>"set_off", "statusrequest" => "statusrequest ."} }`
    `attr flur_Steckdose publishSet set_on set_off haus/cmd/flur_erdgeschoss/Kanal1`
    `attr flur_Steckdose publishSet_statusrequest haus/cmd/flur_erdgeschoss/Kanal1/statusrequest`
    `attr flur_Steckdose retain 1`
    `attr flur_Steckdose stateFormat state`
    `attr flur_Steckdose subscribeReading_state haus/flur_erdgeschoss/Kanal1`
    `attr flur_Steckdose webCmd statusrequest:on:off`

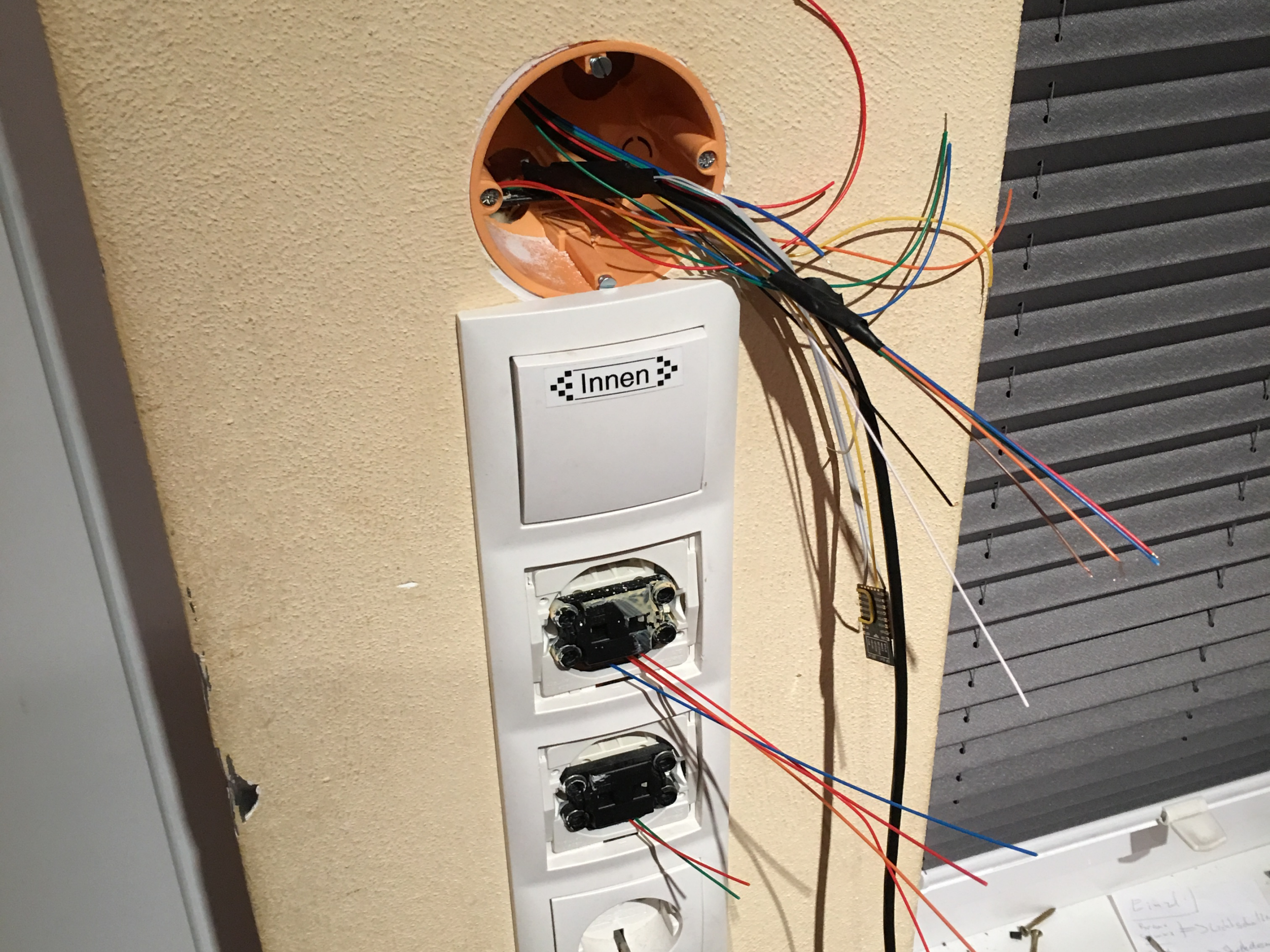

BETRAFOL
IN1
IN2
IN3
IN4
Relay Module

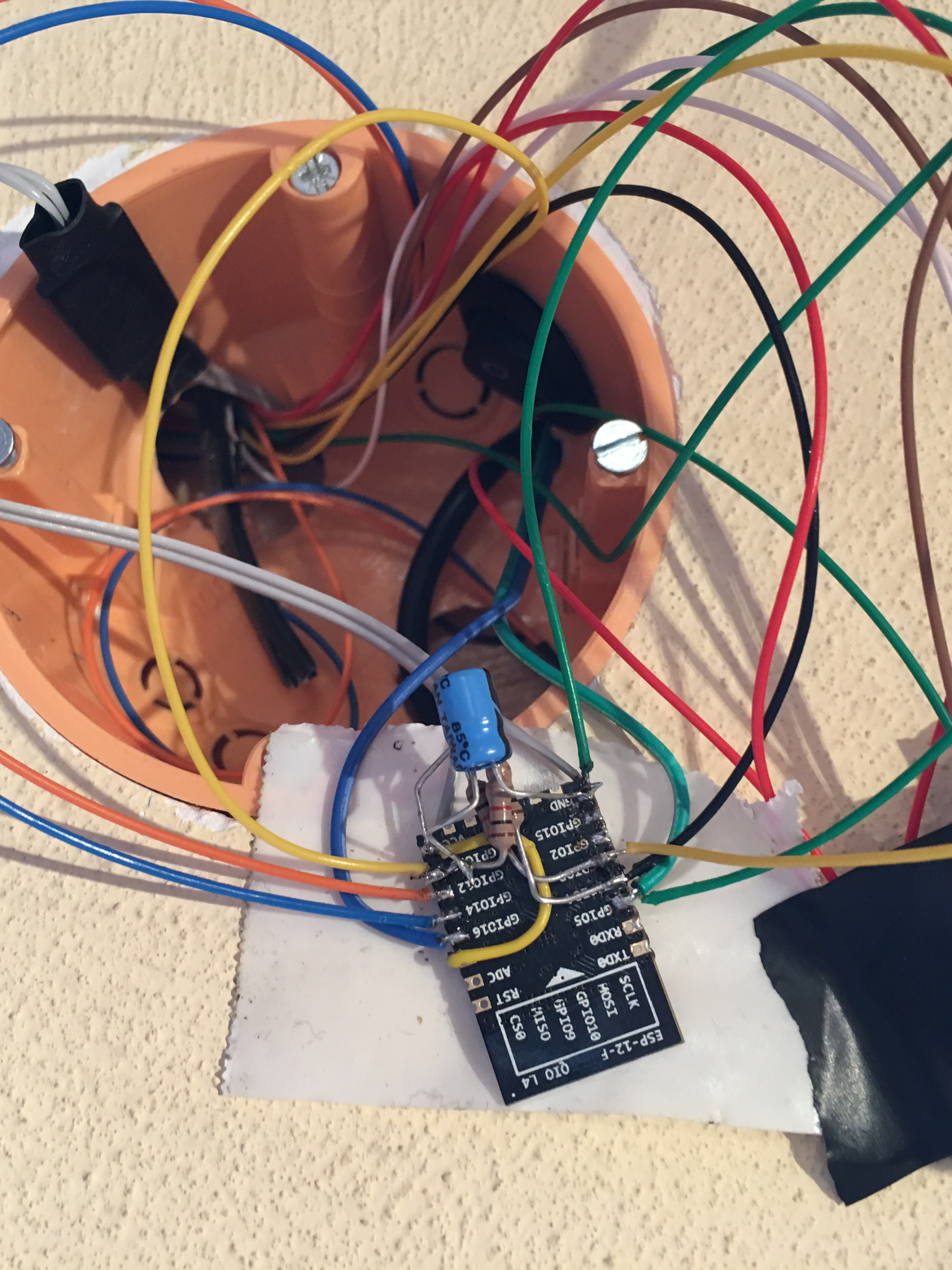

7x16°
7x16°
M16x1.5
M16x1.5

BETRAFOL
original


ESP-12-F
QIO L4
SCLK
MOSI
GPIO10
GPIO9
MISO
CS0
TXD0
RXD0
GPIO5
GPIO4
GPIO0
GPIO2
GPIO15
GND
RST
ADC
GPIO16
GPIO14
GPIO12
GPIO13


470µF
16V
WiFi
MODEL ESP8266MOD
VENDOR AI-THINKER
ISM 2.4GHz
PA +25dBm
802.11b/g/n
LUX


Innen


ESP-12-F
SCLK
MOSI
GPIO10
GPIO9
MISO
CS0
RST
ADC
TXD0
RXD0
GPIO5
GPIO16
GPIO14
GPIO12
GPIO15
GPIO2
85°C

Innen
Aussen
Steckdose

SZENARIO 2
TÜRKLINGEL

- ESPEasy-Device Konfiguration
  - Device "Switch Input", Name "KlingelTaster", Pin "D7 (13)", "Push Button Active Low", "Send Data", Value Name "Click"
  - Device "Switch Input", Name "Klingel", Pin "D0 (16)", "Normal Switch", "Inversed", mit Relais für die Klingel
- FHEM Konfiguration
  - Ein "zentrales" Device für den ESP (zzgl. Verbindung zum Broker)
  - Je ein Device für den Klingeltaster und das Klingelrelais (oder auch ein Device)
- Retain inaktiv (also in FHEM deaktivieren), sonst klingelt es bei einem ESP Neustart

SZENARIO 2
FHEM EINRICHTUNG KLINGEL

- FHEM-Device als Repräsentant für den Aktor einrichten

  - ```
    define flur_Klingel MQTT_DEVICE
    attr flur_Klingel alias Klingel
    attr flur_Klingel eventMap { dev=>{}, usr=>{"ring"
    => "Ring 1,250"} }
    attr flur_Klingel icon ring
    attr flur_Klingel publishSet_Ring sensors/cmd/
    flur_Aussenwand_01/Klingel/pulse
    attr flur_Klingel subscribeReading_Taster sensors/
    flur_Aussenwand_01/KlingelTaster/Click
    attr flur_Klingel webCmd ring
    ```
    ```

SZENARIO 2
ESPEASY RULES

- Interne ESPEasy-Rule für die Klingel
- Interne Abarbeitung, auch wenn FHEM und/oder die anderen Komponenten ausgefallen sind
- Gleichbleibende Reaktionsgeschwindigkeit, keine Netzwerklatenz
- Rule für das Klingelbeispiel:
  - ```
    On KlingelTaster#Click do
      Pulse,16,0,250
    endon
    ```
  - Bei KlingelTaster-Click, Pin 16 für 250ms, auf LOW schalten, dann wieder auf HIGH
    ```

SZENARIO 2
RULES VON FHEM AUS SETZEN

- Beim Umschalten eines Dummy "global_Home", soll die Rule im ESP gesetzt (bzw. gelöscht) werden

```
• notify global_Home:(on|off) {
    if (Value($NAME) eq 'on') {
      # KlingelTaster im ESP aktivieren
     if (ReadingsVal('flur_Aussenwand_01', 'status', 'Connection Lost') eq
 'Connected') {
      my $res = get(ReadingsVal('flur_Aussenwand_01', 'ConfigLink', '').'/rules?
rules=On%20KlingelTaster%23Click%20do%20%0A%20%20Pulse%2C16%2C0%2C250%0Aendon');
     }
  } else {
     # KlingelTaster im ESP deaktivieren
    if (ReadingsVal('flur_Aussenwand_01', 'status', 'Connection Lost') eq
'Connected') {
      my $res = get(ReadingsVal('flur_Aussenwand_01', 'ConfigLink', '').'/rules?
rules=');
     }
  }
   return undef;
}
```

MQTT RETAIN UND QUALTIY OF SERVICE - WOZU?

- Situation ESP startet neu
  - Durch Retain in FHEM aktiv: ESP erhält zuletzt gewünschten Schaltzustand erneut durch den Broker zugestellt (egal ob FHEM gerade aktiv ist, oder nicht)
- Situation FHEM war weg bzw. startet neu
  - Durch Startup-Skript in FHEM werden alle ESPs auf den aktuellen Schaltzustand hin befragt ("statusrequest") und in FHEM aktualisiert
  - Retain im ESP würde hier vorgaukeln, dass der ESP noch erreichbar ist, was hier nicht gewünscht ist
- QoS erreicht hier keine zusätzliche Sicherheit, da die Komponenten den aktuellen Zustand sowieso aktiv zurückmelden