

# Energy2Shelly\_ESP

Ein Projekt von TheRealMoeder

Einrichtung des Shelly Emulators mit Shelly 3 EM, ESP8266 D1 mini  
und Marstek Venus E

Quellenangaben:

[https://github.com/TheRealMoeder/Energy2Shelly\\_ESP](https://github.com/TheRealMoeder/Energy2Shelly_ESP)

<https://www.shelly.com/de/products/shelly-3em>

<https://marstek.de/products/marstek-venus-energycube-ac-gekoppeltes-energiespeichersystem>

<https://www.az-delivery.de/products/d1-mini>

---

Stand der Mini Doku: 21.06.2025

Erstellt von Crusher606

## Inhaltsverzeichnis

Wichtiger Hinweis .....	3
Projektbeschreibung .....	3
Hardware .....	3
Software .....	3
Ablauf .....	4

---

## Wichtiger Hinweis

---



Arbeiten an elektrischen Anlagen dürfen nur von einer Elektrofachkraft ausgeführt werden.

Wenn kein EnergyMeter in ihrem Zählerschrank oder Stromverteiler eingebaut ist, rate ich dringend davon ab ein solches Gerät ohne Fachkenntnis selbst einzubauen.

---

## Projektbeschreibung

---

Zur automatischen Anpassung der abgegebenen Leistung (Nulleinspeisung) und dem Laden benötigt der Marstek Venus E Speicher einen Wert vom aktuellen Verbrauch. Hierzu kann ein Shelly PRO 3EM mit dem Speicher gekoppelt werden. Steht eine solche Datenquelle nicht zur Verfügung können alternative Datenquellen mit Energy2Shelly an den Marstek gesendet werden. Energy2Shelly kann dann einen Shelly PRO 3EM emulieren. Das Programm „gaukelt“ dem Marstek Venus E vor, er wäre ein kompatibles EnergyMeter.

In diesem HowTo wird die Verwendung eines älteren Shelly 3EM beschrieben, der zum Zeitpunkt dieser Anleitung nicht mehr hergestellt wird und auf Grund seines Alters nicht mit einem Marstek Venus E kommunizieren kann. Für die Kommunikation ist RPC über UDP auf Port 1010 (oder 2220) im neuen Shelly PRO 3EM einzustellen, damit eine Kopplung möglich ist. Der alte Shelly 3EM kann das aber nicht.

Energy2Shelly holt sich die Daten per WLAN vom Shelly 3EM und sendet diese dann ebenfalls über WLAN an den Marstek Venus E.

---

## Hardware

---

Shelly 3EM (alte Version ohne PRO)

ESP8266 D1 mini (mit 5VDC Spannungsversorgung)

Marstek Venus E

---

## Software

---

Energy2Shelly Programmdatei ([https://github.com/TheRealMoeder/Energy2Shelly\\_ESP](https://github.com/TheRealMoeder/Energy2Shelly_ESP))

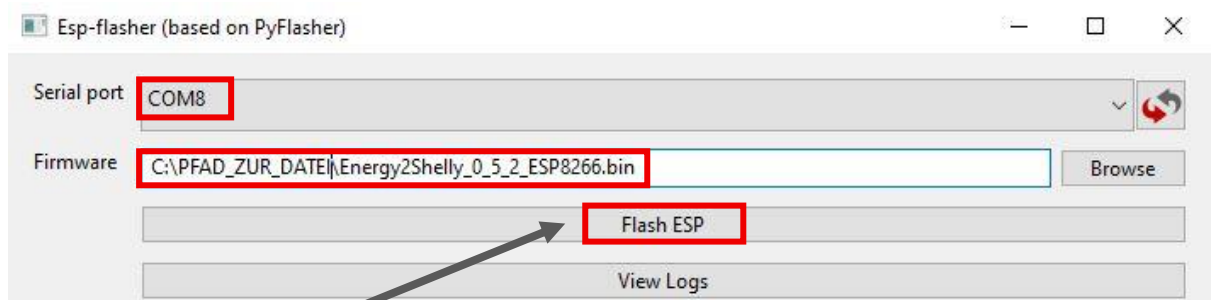
ESP Flasher (ESP flasher for ESP32, ESP32-C3, ESP32-S2, ESP32-S3 and ESP8266 (Tasmota)

Marstek APP (für Android oder Apple)

## Ablauf

Die Energy2Shelly .bin Datei wird mit der Software „ESP Flasher“ auf den ESP8266 geflasht. Danach wird der WLAN Access Point (AP) von Energy2Shelly im Browser aufgerufen und konfiguriert. Abschließend wird der Energy2Shelly Emulator im Marstek Venus E Speicher eingebunden.

- ESP8266 per USB Kabel mit dem PC verbinden
- ESP Flasher Programm aufrufen
- COM Port einstellen (der ESP bekommt einen eigenen COM-Port der mit Sicherheit von dem in dieser Anleitung abweicht)
- die Energy2Shelly.bin - Datei zum flashen aussuchen



„Flash ESP“ klicken

```
Console
Using 'COM8' as serial port.
Connecting....
Detecting chip type... Unsupported detection protocol, switching and tryin
Connecting....
Detecting chip type... ESP8266
Connecting....

Chip Info:
- Chip Family: ESP8266
- Chip Model: ESP8266EX
- Chip ID: 008E7FAA
- MAC Address: 34:94:54:8E:7F:AA
Uploading stub...
Running stub...
Stub running...
Changing baud rate to 460800
Changed.
- Flash Size: 4MB
- Flash Mode: dio
- Flash Frequency: 40MHz
Erasing flash (this may take a while)...
```

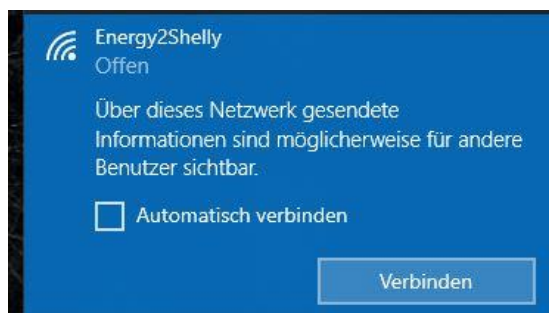
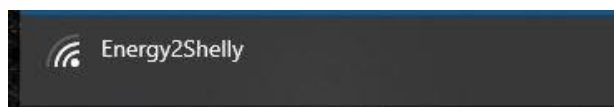
```
Console
Writing at 0x00040028... (57 %)
Writing at 0x0004574b... (61 %)
Writing at 0x0004a5ec... (66 %)
Writing at 0x0004f6e3... (71 %)
Writing at 0x00054bf1... (76 %)
Writing at 0x0005a3fd... (80 %)
Writing at 0x0005fab6... (85 %)
Writing at 0x00064e44... (90 %)
Writing at 0x0006bb11... (95 %)
Writing at 0x00072611... (100 %)
Wrote 494160 bytes (343865 compressed) at 0x00000000 in 8.2 seconds (effective
Hash of data verified.

Leaving...
Hard Resetting...
Hard resetting via RTS pin...
Done! Flashing is complete!

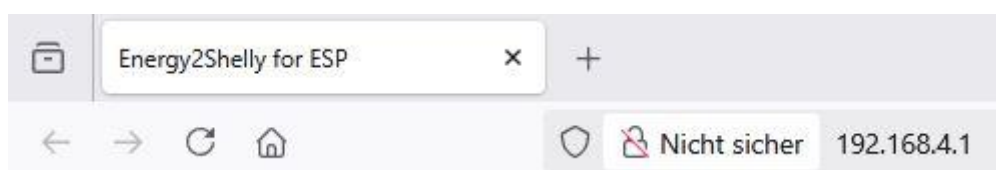
Showing logs:
[10:03:45] [n\rn]l` b[] b[] lnBn l` r[]l[]l` r[]l[]l` r[]l[]l` r[]l[]l` r[]l[]l`
[10:03:45]*wm:No wifi saved, skipping
[10:03:45]*wm:AutoConnect: FAILED for 102 ms
[10:03:45]*wm:StartAP with SSID: Energy2Shelly
[10:03:46]*wm:AP IP address: 192.168.4.1
[10:03:46]*wm:Starting Web Portal
```

Flashvorgang ist beendet. Der ESP startet neu und es wird sofort der AccessPoint von Energy2Shelly gestartet.

Per WLAN mit dem AccessPoint „Energy2Shelly“ verbinden



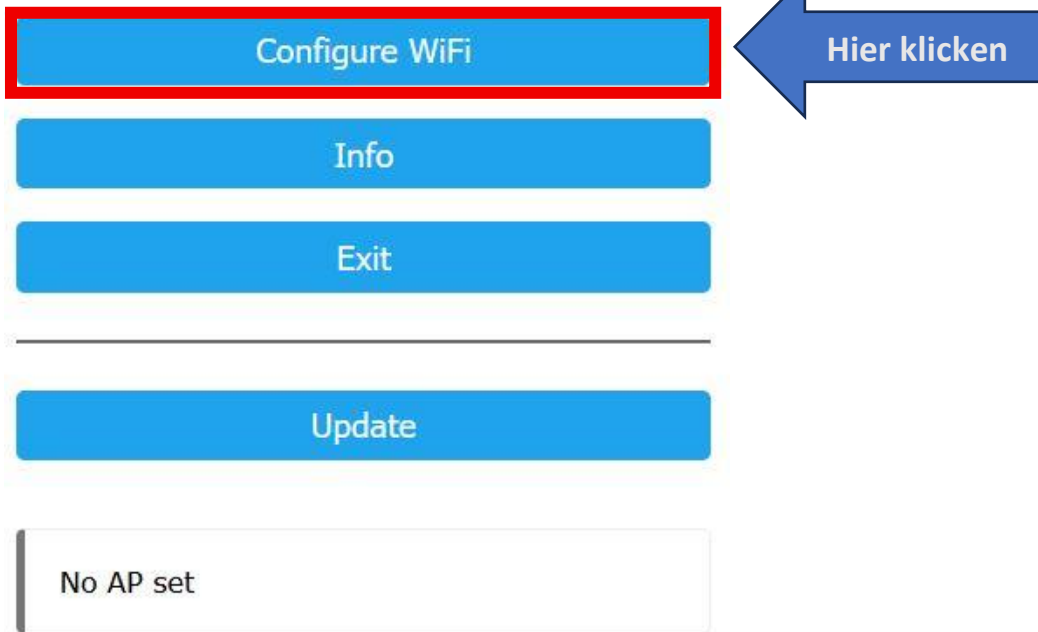
Es öffnet sich der Browser und die Startseite (Webinterface) von Energy2Shelly wird aufgerufen.



Wenn das Webinterface nicht automatisch im Browser geöffnet wird, kann man die URL **192.168.4.1** auch händisch im Browser eingeben.

## Energy2Shelly for ESP

### Energy2Shelly



The image shows the Energy2Shelly web interface. It features a vertical list of blue buttons: 'Configure WiFi', 'Info', 'Exit', and 'Update'. The 'Configure WiFi' button is highlighted with a red rectangular border. To the right of this button, a blue arrow points left towards it, with the text 'Hier klicken' (Click here) inside. Below the buttons, there is a white box with the text 'No AP set'.

Eigenes WLAN suchen, auswählen und Passwort eintragen



The image shows the configuration fields for a custom WLAN. There are two input fields: 'SSID' and 'Password'. The 'SSID' field contains the text 'EIGENES\_WLAN' and is highlighted with a red rectangular border. The 'Password' field contains a series of dots and is also highlighted with a red rectangular border. Below the 'Password' field, there is a checkbox labeled 'Show Password'.

Nun wird im unteren Teil die Datenkommunikation mit dem Shelly 3EM konfiguriert.

## General settings

### Data source

MQTT for MQTT

HTTP for generic HTTP

SMA for SMA EM/HM multicast

SHRDZM for SHRDZM UDP data

SUNSPEC for Modbus TCP SUNSPEC data

HTTP

### Server

MQTT Server IP, query url for generic HTTP or Modbus TCP  
server IP for SUNSPEC

http://IP\_ADRESSE\_SHELLY\_3EM/status

Hier muss man die URL  
seines Shelly 3EM eintragen

### Query period

for generic HTTP and SUNSPEC, in milliseconds

1000

### GPIO

of internal LED

### GPIO is inverted

true OR false

### Shelly ID

12 char hexadecimal, defaults to MAC address of ESP

112233445566

Diese Bezeichnung kann  
frei geändert werden

### Shelly UDP port

1010 for old Marstek FW, 2220 for new Marstek FW v226+/  
v108+

2220

Hier wird der RPC over UDP Port  
eingetragen (1010 oder 2220)

### Force decimals numbers for Power values

true to fix Marstek bug

true

### SMA serial number

optional serial number if you have more than one SMA EM/  
HM in your network

## JSON paths for MQTT and generic HTTP

### Total power JSON path

e.g. ENERGY.Power OR TRIPHASE for tri-phase data

### Export power JSON path

Optional, for net calc (e.g. "i-e")

### Phase 1 power JSON path

optional

### Phase 2 power JSON path

Phase 2 power JSON path

optional

### Phase 3 power JSON path

Phase 3 power JSON path

optional

### Energy from grid JSON path

e.g. ENERGY.Grid

### Energy to grid JSON path

e.g. ENERGY.FeedIn

Jetzt die Eingaben mit „SAVE“ übernehmen.

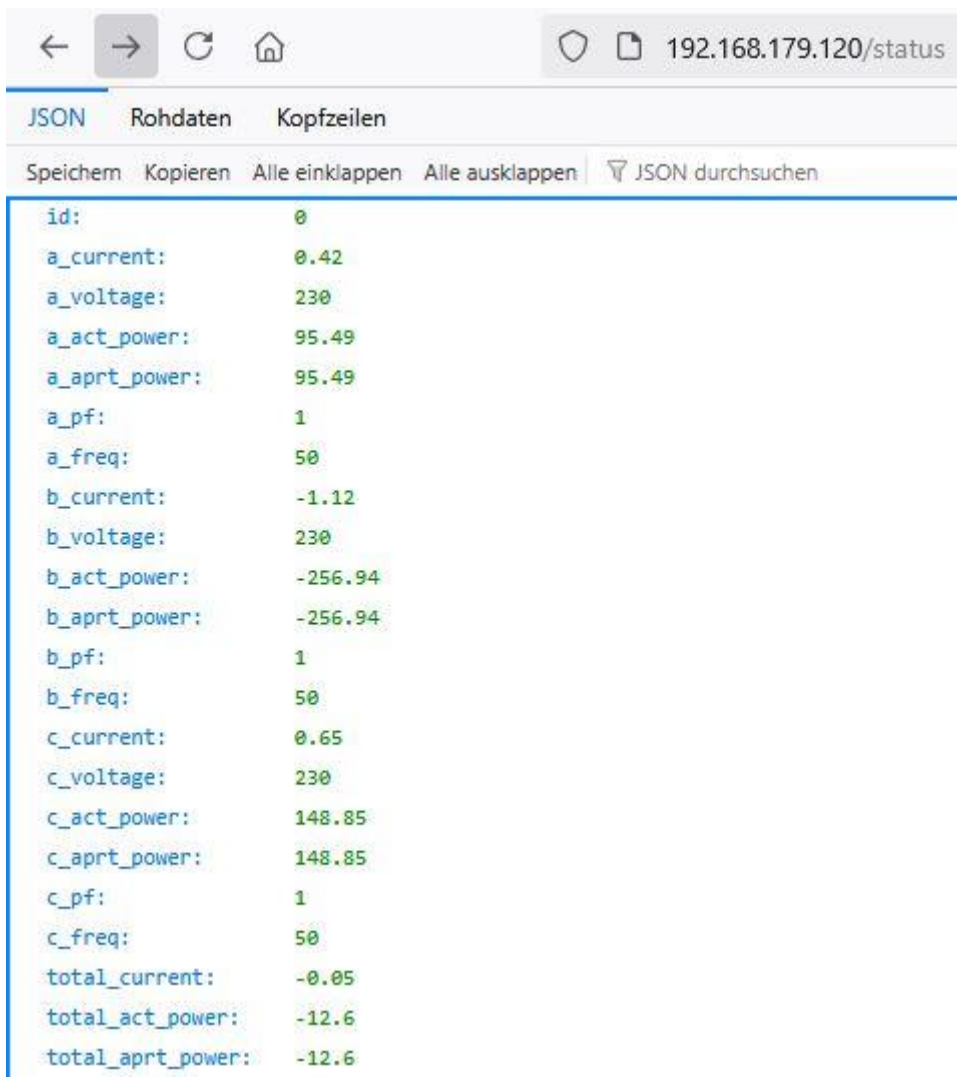
Der ESP8266 bootet neu und ist über die IP-Adresse (die per DHCP vom eigenen Router vergeben wird) im eigenen Netzwerk zu finden. Um die IP-Adresse zu ermitteln kann man dann auch in seinem Router nachsehen.



In meinem Beispiel hat der ESP8266 die Adresse 120 bekommen.  
Gibt man im Browser die IP-Adresse ein, so öffnet sich die Info-Seite von Energy2Shelly.



Hängt man an die URL von oben noch ein „/status“ an, so sieht man die Daten aus dem alten Shelly 3EM, die als emulierter Shelly PRO 3EM abgerufen werden können.



Jetzt kann man in der Marstek App das emulierte EnergyMeter im WLAN auswählen und für die automatische CT Regelung einbinden.