# **Automatisches Solar-Nachführungssystem**

# **Abgabe 06.05.2025**

# Ein Bild, das draußen, Solarenergie, Solarpanel, Dish-Solaranlage enthält.**Erstellt von: Fynn Bremer**

# [**Inhaltsverzeichnis**](#_Toc196811358)

# [Projektbeschreibung](#_Toc196811360)

# [Projektziel](#_Toc196811361)

# [Funktionsumfang](#_Toc196811362)

# [Technische Komponenten](#_Toc196811363)

# [Ablauf und Steuerung](#_Toc196811364)

# [Aufbau und Verkabelung](#_Toc196811365)

# [Schaltplan](#_Toc196811366)

# [Installations- und Bedienungsanleitung](#_Toc196811367)

# [Node-RED](#_Toc196811368) Einbindung & Dashboard

# [Einbindung Thingspeak & Heidi SQL](#_Toc196811369)

# [Persönliches Fazit / Bilder](#_Toc196811370)

# Vollständiger Quellcode / Node-Flow

# Projektbeschreibung

Das Projekt umfasst die Entwicklung eines automatisierten Nachführungssystems für ein Solarpanel, das sich optimal zur Sonne ausrichtet, um den Energieertrag zu maximieren. Die Ausrichtung erfolgt durch Auswertung von Lichtintensitäten mittels vier Lichtsensoren (LDRs). Zusätzlich überwacht ein Windsensor die Windgeschwindigkeit, um bei Sturmgefahr das Panel automatisch in eine Schutzposition zu bringen.

Ein integriertes Dashboard zeigt in Echtzeit wichtige Umweltdaten (Temperatur, Luftfeuchtigkeit, Helligkeit, Windgeschwindigkeit) sowie aktuelle Leistungswerte (Spannung, Stromstärke, Leistung) an. Die Kommunikation erfolgt über das MQTT-Protokoll. Neben dem Automatikbetrieb ist eine manuelle Steuerung des Panels über MQTT-Befehle möglich. Das System verwendet u.a. folgende Sensoren und Module: DS18B20 (Temperatur), AHT10 (Luftfeuchtigkeit), INA226 (Spannung/Strommessung) und einen Reedkontakt für die Windmessung. Die Motorsteuerung erfolgt über BTS7960 Motortreiber.

# Projektziel

Ziel des Projekts ist die Entwicklung eines intelligenten Systems, das ein Solarpanel automatisch zur Sonne ausrichtet, um die maximale Energieausbeute zu erzielen. Das System überwacht außerdem relevante Umweltdaten wie Temperatur, Luftfeuchtigkeit, Windgeschwindigkeit und Lichtintensität und stellt diese über ein MQTT-basiertes Dashboard in Echtzeit bereit.

Zusätzlich kann das Solarpanel manuell über MQTT-Kommandos gesteuert werden, um größtmögliche Flexibilität zu gewährleisten.

# Funktionsumfang

**Automatische Nachführung:**

* Das Solarpanel wird auf Basis der Lichtstärke-Sensoren (LDRs) ausgerichtet, sodass es stets optimal zur Sonne zeigt.

**Manuelle Steuerung:**

* Manuelle Bewegung des Panels in X- und Y-Richtung über MQTT-Befehle möglich (z.B. für Wartung oder besondere Betriebsmodi).

**Umweltdatenerfassung:**

* Temperatur- und Luftfeuchtigkeitsmessung (DS18B20 und AHT10 Sensoren)
* Windgeschwindigkeitsmessung (Reed-Sensor)
* Lichtintensitätsmessung an vier Seiten (LDRs)
* Ermittlung von Spannung, Strom und Leistung (INA226-Modul)

**Sturmschutz:**

Überschreitet die Windgeschwindigkeit eine festgelegte Grenze, fährt das Panel automatisch in eine sichere Position und deaktiviert den Automatikbetrieb.

**MQTT-Datenübertragung:**

Alle erfassten Werte werden per MQTT an ein zentrales Dashboard gesendet. Steuerbefehle und Alarme werden ebenfalls über MQTT empfangen bzw. veröffentlicht.

# Technische Komponenten

**Sensoren**

* Microcontroller: ESP32-S3 Nano
* LDRs für Helligkeitserkennung
* DS18B20 Temperatursensor
* AHT10 Temperatur- und Feuchtigkeitssensor
* INA226 zur Strom- und Spannungsmessung
* Windsensor (Reedkontakt)

**Motorsteuerung:**

* BTS7960 Motortreiber zur Ansteuerung der Bewegungsachsen (X und Y)

**Kommunikation**

* WLAN-Verbindung zu einem lokalen Netzwerk
* MQTT-Protokoll für Messaging

**Spannungsversorgung:**

Separate Versorgung für Sensorik, Aktuatoren und Steuerungseinheit.

# Ablauf und Steuerung

* Das System überwacht kontinuierlich die Lichtintensitäten der vier LDR-Sensoren.
* Es vergleicht die Werte und richtet das Panel durch gezielte Motorbewegungen automatisch zur besten Lichtquelle aus.
* Parallel werden alle Sensorwerte zyklisch gelesen und über MQTT publiziert.
* Bei einer erkannten Sturmgefahr ( > 15 km/h) fährt das System das Panel ein und setzt eine Alarmmeldung ab. (für Testzwecke auf 15km/h gesetzt)
* Die Alarmmeldung muss erst quittiert, erst dann richtet sich das Panel wieder nach der Sonne aus.
* Eine manuelle Bedienung über MQTT ist jederzeit möglich und überschreibt temporär den Automatikmodus.

# Aufbau und Verkabelung

**Aufbau**

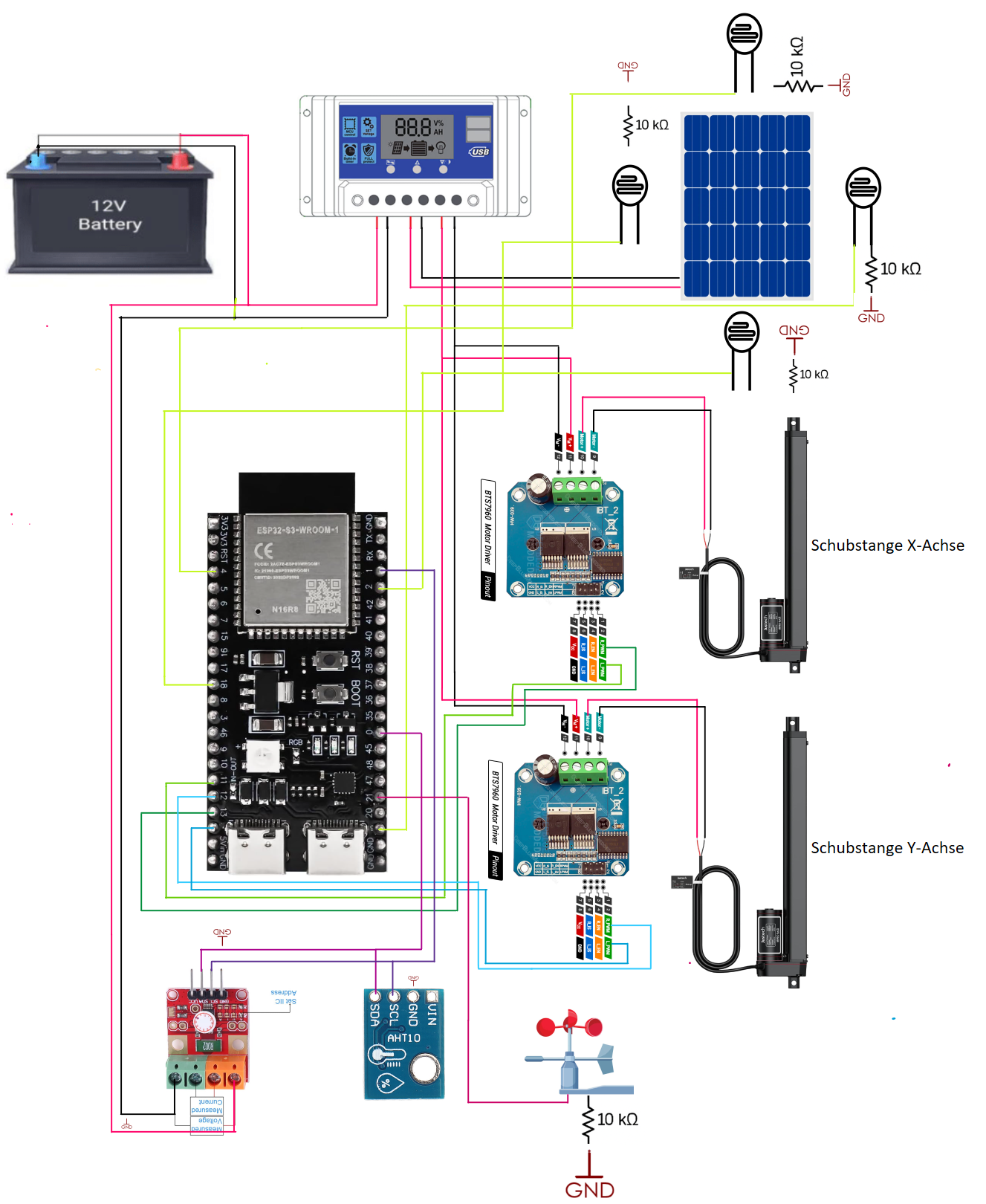
Das Gestell wurde aus ITEM-Aluprofilen gefertigt. Der große Vorteil dieses Materials liegt in seiner Witterungsbeständigkeit sowie der einfachen Verarbeitung und Montage. Für Testzwecke wurde das Gestell mobil auf Rollen ausgeführt, wohingegen es im regulären Einsatz fest im Boden verankert würde. Dabei würde auch die Technik bestmöglich vor äußeren Einflüssen geschützt werden.

**Anschluss am Microcontroller**

|  |  |
| --- | --- |
| LDR Links | ADC Pin 18 |
| LDR Rechts | ADC Pin 19 |
| LDR Vorne | ADC Pin 2 |
| LDR Hinten | ADC Pin 4 |
| BTS7960 Motor X | PWM Pins 13 (RPWM) / 11 (LPWM) |
| BTS7960 Motor Y | PWM Pins 12 (RPWM) / 14 (LPWM) |
| DS18B20 (Temperatursenor Außen) | Pin 41 (OneWire) |
| AHT10 Sensor | I2C Pins SCL=1, SDA=0 |
| INA226 Sensor | I2C Pins (gemeinsam mit AHT10) |
| Windsensor | Pin 21 (Interrupt Pin) |

Die Verlegung der Kabel entspricht nicht den endgültigen Standards, da zur Testzwecken die Verlegung Teils provisorisch gelöst wurden ist.

# Schaltplan



# Installations- und Bedienungsanleitung

**Manueller Betrieb**

**Via MQTT-Topics manuell steuerbar:**

solar/befehl/bewege/x/links→ Panel nach links

solar/befehl/bewege/x/rechts→ Panel nach rechts

solar/befehl/bewege/y/hoch→ Panel nach oben

solar/befehl/bewege/y/runter→ Panel nach unten

|  |  |
| --- | --- |
| **Automatik aktivieren:** | **Automatik deaktivieren:** |
| Topic solar/auto\_control → "on" | Topic solar/auto\_control → "off" |

**Windalarm-Funktion**

**Wird eine Windgeschwindigkeit > 15 km/h gemessen:**

Das Panel wird automatisch in Sicherheitsposition eingefahren.

Automatik wird deaktiviert.

MQTT-Topic solar/wind/alarm → "ALARM

**Rücksetzung des Alarms:**

MQTT-Topic solar/wind/reset → Nachricht beliebig senden

Danach wird der Automatikbetrieb wieder aktiviert.

**Störungen und Wartung**

| **Problem** |  |  |  |  | **Lösungsvorschlag** |
| --- | --- | --- | --- | --- | --- |
| Keine WLAN-Verbindung |  |  |  |  | Prüfen der SSID und Passwort |
| MQTT-Verbindung bricht ab |  |  |  |  | Serverstatus prüfen, Neustart System |
| Motoren reagieren nicht |  |  |  |  | Motorverkabelung und BTS7960 prüfen |
| Sensorwerte fehlen |  |  |  |  | Bus-Verkabelung und Sensoren prüfen |
| Ständige Windalarme |  |  |  |  | Sensor kalibrieren / reinigen |

# Node-RED Einbindung & Dashboard

**Node-RED Einbindung**

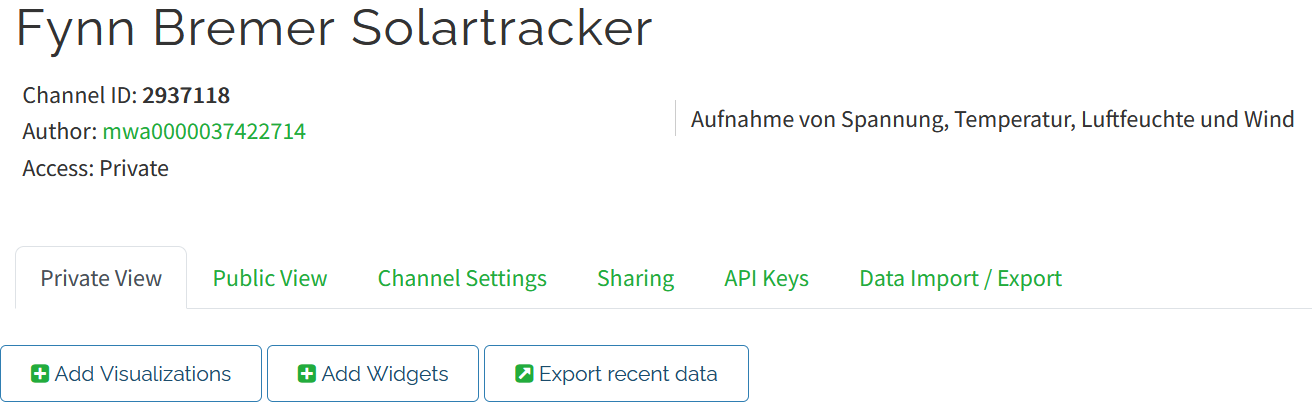
Ein Bild, das Text, Screenshot, Diagramm, Software enthält.

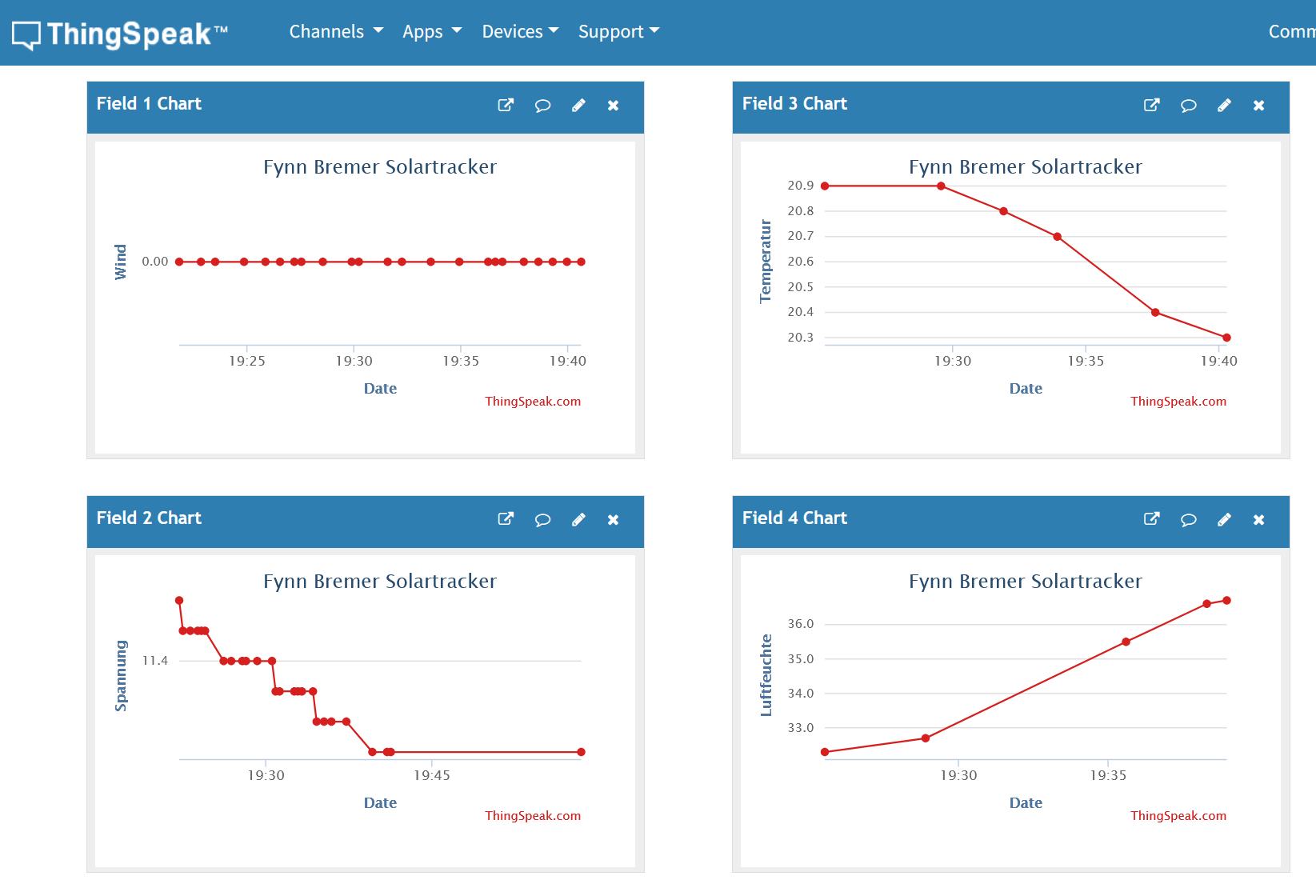
KI-generierte Inhalte können fehlerhaft sein.

**Node-RED Dashboard** Ein Bild, das Text, Screenshot, Diagramm, Schrift enthält.

KI-generierte Inhalte können fehlerhaft sein.

# Einbindung Thingspeak & HediSQL





# Persönliches Fazit / Bilder

Die Entwicklung und Umsetzung dieses Projekts waren sowohl technisch herausfordernd als auch sehr spannend.

Es verbindet auf ideale Weise eine Mikrocontroller Steuerung, IoT-Kommunikation (MQTT) und Energieoptimierung, daher fand ich das Thema für mein Projekt sehr passend.

**Besonders gut gelungen:**

Flexibilität durch Automatik und manuelle Steuerung:

* Die Kombination ermöglicht ein Nutzererlebnis, wie man es von professionellen Systemen erwartet.

Integrierte Windüberwachung als Schutzmechanismus:

* Feature für die Langlebigkeit und Sicherheit des Systems.

Modernes Monitoring über MQTT-Dashboard:

* Nutzer können jederzeit aktuelle Umwelt- und Systemdaten einsehen und auf Veränderungen reagieren.

Kompakte Bauweise für begrenzte Flächen:

* Gerade für kleine Gärten oder Balkonanlagen bietet ein automatisch nachgeführtes Panel enorme Vorteile, da auf begrenztem Raum eine maximale Energieausbeute erzielt werden kann.

**Herausforderungen:**

Sensorabstimmung:

* Besonders die LDRs erforderten viel Arbeit bei der Kalibrierung, da sogar bei wenig Licht immer schnell der Maximalwert erreicht wurden ist. Erst durch das Einlöten eines richtigen Widerstandes sind die Werte bei voller Einstrahlung unter dem Maximalwert geblieben. Außerdem können Einflüsse wie Bewölkung oder Schmutz die Messergebnisse stark beeinflussen können

Feinabstimmung des Windsensors:

* Die Kalibrierung des Windsensors nahm auch einige Zeit in Anspruch, da ein richtiger Widerstand eingelötet werden musste, damit keine Extremen Schwankungen bei den Werten Zustandekommen. Außerdem hat der Windmesser pro Umdrehung drei Reedkontakt Signale ausgegeben, was immer für zu hohe Werte gesorgt hat, da im Programmcode mit einer Umdrehung gerechnet wurden ist.

Kommunikationsprotokolle und Timing-Probleme:

* Die gleichzeitige Integration von OneWire, I2C und WLAN auf einem Mikrocontroller brachte anfangs Timing-Schwierigkeiten mit sich, die gelöst werden mussten.

**Was ich daraus gelernt habe:**

Systemdenken:

* Projekte dieser Komplexität verlangen, dass man Hardware, Software und Umgebung als ein zusammenhängendes System betrachtet.

Fokus auf Robustheit:

* Funktion allein genügt nicht – insbesondere bei Outdoor-Anwendungen ist Zuverlässigkeit entscheidend.

Synergien von IoT und Energiegewinnung:

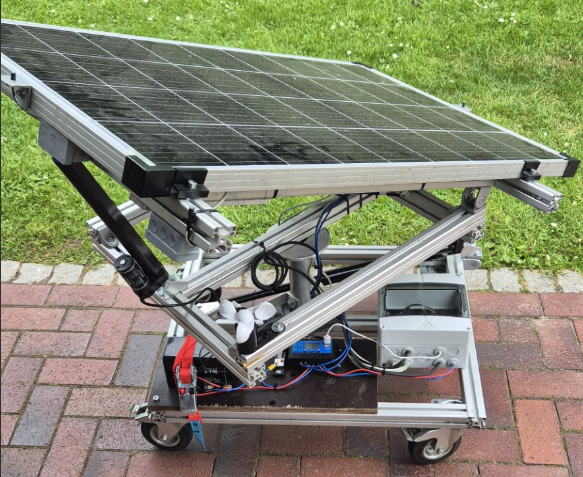
* Die intelligente Steuerung erneuerbarer Energiequellen wird in Zukunft eine immer zentralere Rolle spielen.

**Marktpotenzial:**

Automatisch nachgeführte, kompakte Solarsysteme könnten zukünftig eine echte Marktlücke schließen – speziell Balkonanlagen.

Wenn diese Systeme in Zukunft noch günstiger produziert werden können, könnten sie eine breite Anwendung im privaten Bereich finden und den Eigenverbrauch von Solarstrom deutlich steigern.







# Vollständiger Quellcode

1. import machine
2. from machine import Pin, ADC, I2C
3. from time import sleep, ticks\_ms, ticks\_diff
4. import network
5. import onewire, ds18x20
6. from umqtt.simple import MQTTClient
7. from ina226 import INA226
9. # WLAN-Konfiguration
10. WLAN\_NAME = "FRITZ!Box 6590 Cable Bremer" # SSID des WLANs
11. WLAN\_PASSWORT = "Maunzi2216" # Passwort für das WLAN
13. # MQTT-Konfiguration
14. MQTT\_SERVER = "192.168.178.24" # Adresse des MQTT-Brokers
15. MQTT\_CLIENT\_ID = "solar\_tracker" # Eindeutige Gerätekennung
17. # MQTT-Themen (Topics)
18. MQTT\_TOPIC\_BEFEHL = b"solar/befehl" # Steuerbefehle
19. MQTT\_TOPIC\_TEMPERATUR = b"solar/temperatur" # Temperaturdaten
20. MQTT\_TOPIC\_LUFTFEUCHTE = b"solar/luftfeuchte" # Luftfeuchte vom AHT10
21. MQTT\_TOPIC\_LEISTUNG = b"solar/leistung" # Elektrische Leistung
22. MQTT\_TOPIC\_SPANNUNG = b"solar/spannung" # Spannung (INA226)
23. MQTT\_TOPIC\_STROM = b"solar/strom" # Strom (INA226)
24. MQTT\_TOPIC\_WIND = b"solar/wind" # Windgeschwindigkeit
25. MQTT\_TOPIC\_LDR\_LINKS = b"solar/ldr/links" # Lichtsensor links
26. MQTT\_TOPIC\_LDR\_RECHTS = b"solar/ldr/rechts" # Lichtsensor rechts
27. MQTT\_TOPIC\_LDR\_VORNE = b"solar/ldr/vorne" # Lichtsensor vorne
28. MQTT\_TOPIC\_LDR\_HINTEN = b"solar/ldr/hinten" # Lichtsensor hinten
29. MQTT\_TOPIC\_AUTOMATIK = b"solar/automatik" # Automatikbetrieb ein/aus
30. MQTT\_TOPIC\_WIND\_ALARM = b"solar/wind/alarm" # Windalarmstatus
31. MQTT\_TOPIC\_WIND\_RESET = b"solar/wind/zuruecksetzen" # Rücksetzen des Windalarms
32. MQTT\_TOPIC\_WIND\_MAX = b"solar/wind/maximal" # Neue Grenzwerte setzen
33. MQTT\_TOPIC\_LDR\_LINKS\_KAL = b"solar/ldr/links/kalibrierung" # LDR-Korrekturfaktor
35. # Wind- und Kalibriergrenzwerte
36. maximale\_windgeschwindigkeit = 10.0 # obere Grenze für Automatikbetrieb
37. WIND\_ALARM\_GRENZE = 15.0 # Windgeschwindigkeit für Alarm
38. LDR\_LINKS\_KALIBRIERUNG = 0.95 # Korrekturwert für linken LDR
40. # Zeitverfolgung für Sensorintervall
41. letzte\_sensorzeit = ticks\_ms()
43. # Initialisierung des DS18B20 Temperatursensors
44. ds\_pin = Pin(41)
45. ds\_sensor = ds18x20.DS18X20(onewire.OneWire(ds\_pin))
46. roms = ds\_sensor.scan()
48. # LDR-Sensoren an ADC-Pins
49. LDR\_LINKS = ADC(2)
50. LDR\_RECHTS = ADC(4)
51. LDR\_VORNE = ADC(18)
52. LDR\_HINTEN = ADC(19)
54. # Steuerung für Aktuatoren (BTS7960 H-Brücke)
55. RPWM\_X = Pin(14, Pin.OUT)
56. LPWM\_X = Pin(12, Pin.OUT)
57. RPWM\_Y = Pin(13, Pin.OUT)
58. LPWM\_Y = Pin(11, Pin.OUT)
60. # Windsensor – Impulse zählen
61. WIND\_PIN = Pin(21, Pin.IN, Pin.PULL\_UP)
62. wind\_impulse = 0
63. letzte\_windzeit = 0
65. # IRQ-Handler für Windimpulse
66. def wind\_callback(pin):
67. global wind\_impulse, letzte\_windzeit
68. jetzt = ticks\_ms()
69. if ticks\_diff(jetzt, letzte\_windzeit) > 50: # Entprellung
70. wind\_impulse += 1
71. letzte\_windzeit = jetzt
73. WIND\_PIN.irq(trigger=Pin.IRQ\_FALLING, handler=wind\_callback)
75. # Berechnung der Windgeschwindigkeit aus Impulsen
76. def berechne\_windgeschwindigkeit(impulse, intervall\_s):
77. rpm = impulse \* (30 / intervall\_s)
78. kmh = rpm \* 0.1
79. return kmh
81. # I2C-Bus & Sensoren initialisieren
82. i2c = machine.I2C(scl=Pin(1), sda=Pin(0), freq=100000)
83. AHT10\_ADRESSE = 0x38
84. i2c.writeto(AHT10\_ADRESSE, bytearray([0xE1, 0x08, 0x00])) # Initialisieren
85. sleep(0.05)
86. ina226 = INA226(i2c, 0x40) # Strom/Spannungssensor INA226
88. # WLAN-Verbindung aufbauen
89. wlan = network.WLAN(network.STA\_IF)
90. wlan.active(True)
91. if not wlan.isconnected():
92. print(f"Verbinde mit WLAN {WLAN\_NAME}...")
93. wlan.connect(WLAN\_NAME, WLAN\_PASSWORT)
94. timeout = 10
95. while not wlan.isconnected() and timeout > 0:
96. sleep(1)
97. timeout -= 1
98. print(".")
99. if wlan.isconnected():
100. print("WLAN verbunden, IP-Adresse:", wlan.ifconfig()[0])
101. else:
102. print("WLAN fehlgeschlagen!")
103. machine.reset()
105. # MQTT-Verbindung und Abonnements
106. def mqtt\_verbinden():
107. global client
108. client = MQTTClient(MQTT\_CLIENT\_ID, MQTT\_SERVER)
109. client.set\_callback(mqtt\_callback)
110. client.connect()
111. client.subscribe(MQTT\_TOPIC\_BEFEHL + b"/#")
112. client.subscribe(MQTT\_TOPIC\_AUTOMATIK)
113. client.subscribe(MQTT\_TOPIC\_WIND\_RESET)
114. client.subscribe(MQTT\_TOPIC\_WIND\_MAX)
115. client.subscribe(MQTT\_TOPIC\_LDR\_LINKS\_KAL)
116. print(" MQTT verbunden.")
118. automatik\_aktiv = True
119. wind\_aktiv = False
120. windgeschwindigkeit = 0.0
122. # Steuerung der Motoren über Richtungsparameter
123. def bewege\_aktor(r\_pwm, l\_pwm, richtung, dauer=0.5):
124. r\_pwm.value(0)
125. l\_pwm.value(0)
126. sleep(0.1)
127. if richtung == "ausfahren":
128. r\_pwm.value(1)
129. l\_pwm.value(0)
130. elif richtung == "einfahren":
131. r\_pwm.value(0)
132. l\_pwm.value(1)
133. sleep(dauer)
134. r\_pwm.value(0)
135. l\_pwm.value(0)
137. # MQTT-Ereignisbehandlung
138. def mqtt\_callback(topic, msg):
139. global automatik\_aktiv, wind\_aktiv, maximale\_windgeschwindigkeit
140. print(f" Nachricht: {topic.decode()} - {msg.decode()}")
142. if topic == MQTT\_TOPIC\_AUTOMATIK:
143. automatik\_aktiv = msg.decode().lower() == "on"
144. print("Automatik:", "EIN" if automatik\_aktiv else "AUS")
146. if topic == b"solar/befehl/bewege/x/links":
147. bewege\_aktor(RPWM\_X, LPWM\_X, "ausfahren", 10)
148. elif topic == b"solar/befehl/bewege/x/rechts":
149. bewege\_aktor(RPWM\_X, LPWM\_X, "einfahren", 10)
150. elif topic == b"solar/befehl/bewege/y/hoch":
151. bewege\_aktor(RPWM\_Y, LPWM\_Y, "ausfahren", 10)
152. elif topic == b"solar/befehl/bewege/y/runter":
153. bewege\_aktor(RPWM\_Y, LPWM\_Y, "einfahren", 10)
155. if topic == MQTT\_TOPIC\_WIND\_RESET and wind\_aktiv:
156. print("Wind-Reset empfangen - Automatik reaktiviert.")
157. automatik\_aktiv = True
158. wind\_aktiv = False
159. client.publish(MQTT\_TOPIC\_WIND\_ALARM, b"OK")
161. if topic == MQTT\_TOPIC\_WIND\_MAX:
162. try:
163. neue\_grenze = float(msg.decode())
164. maximale\_windgeschwindigkeit = neue\_grenze
165. print(f"Neue max. Windgeschwindigkeit: {maximale\_windgeschwindigkeit} km/h")
166. except ValueError:
167. print("Ungültiger Wert für Windgrenze:", msg)
169. if topic == MQTT\_TOPIC\_LDR\_LINKS\_KAL:
170. try:
171. neuer\_faktor = float(msg.decode())
172. if 0.5 <= neuer\_faktor <= 1.5:
173. global LDR\_LINKS\_KALIBRIERUNG
174. LDR\_LINKS\_KALIBRIERUNG = neuer\_faktor
175. print(f" Kalibrierung LDR links: {LDR\_LINKS\_KALIBRIERUNG:.2f}")
176. else:
177. print(" Kalibrierungswert außerhalb von 0.5–1.5!")
178. except ValueError:
179. print(" Ungültiger Kalibrierungswert:", msg)
181. mqtt\_verbinden()
183. # Temperatur und Luftfeuchte vom AHT10 auslesen
184. def lese\_aht10():
185. try:
186. i2c.writeto(AHT10\_ADRESSE, b'\xAC\x33\x00')
187. sleep(0.1)
188. daten = i2c.readfrom(AHT10\_ADRESSE, 6)
190. if daten[0] & 0x08 != 0x08:
191. print("Sensor nicht bereit")
192. return
194. roh\_feuchte = ((daten[1] << 16) | (daten[2] << 8) | daten[3]) >> 4
195. roh\_temp = ((daten[3] & 0x0F) << 16) | (daten[4] << 8) | daten[5]
197. feuchte = round((roh\_feuchte / 1048576) \* 100, 1)
198. temperatur = round((roh\_temp / 1048576) \* 200 - 50, 1)
200. print("️ Temperatur:", temperatur, "°C Feuchte:", feuchte, "%")
201. client.publish(MQTT\_TOPIC\_TEMPERATUR, str(temperatur))
202. client.publish(MQTT\_TOPIC\_LUFTFEUCHTE, str(feuchte))
203. except Exception as e:
204. print("Fehler AHT10:", e)
206. # Hauptschleife: Sensoren lesen, MQTT verarbeiten, Motoren steuern
207. while True:
208. if not wlan.isconnected():
209. print(" WLAN getrennt – Neustart...")
210. sleep(5)
211. machine.reset()
213. try:
214. client.check\_msg() # Neue MQTT-Nachrichten verarbeiten
215. except Exception as e:
216. print(" MQTT-Fehler:", e)
217. sleep(5)
218. machine.reset()
220. jetzt = ticks\_ms()
222. if ticks\_diff(jetzt, letzte\_sensorzeit) >= 20\_000:
223. # Temperatur DS18B20
224. for rom in roms:
225. temperatur = ds\_sensor.read\_temp(rom)
226. print(f" DS18B20: {temperatur:.2f} °C")
227. client.publish(MQTT\_TOPIC\_TEMPERATUR, f"{temperatur:.2f}")
229. # INA226 auslesen
230. try:
231. spannung = ina226.bus\_voltage
232. shunt\_spannung = ina226.shunt\_voltage
233. shunt\_widerstand = 0.002
234. strom = shunt\_spannung / shunt\_widerstand
235. leistung = ina226.power
236. print(f" Spannung: {spannung:.2f} V, Strom: {strom:.2f} A, Leistung: {leistung:.2f} W")
237. client.publish(MQTT\_TOPIC\_SPANNUNG, f"{spannung:.2f}")
238. client.publish(MQTT\_TOPIC\_STROM, f"{strom:.2f}")
239. client.publish(MQTT\_TOPIC\_LEISTUNG, f"{leistung:.2f}")
240. except Exception as e:
241. print("Fehler INA226:", e)
243. # Windgeschwindigkeit berechnen
244. windgeschwindigkeit = berechne\_windgeschwindigkeit(wind\_impulse, 10)
245. print(f" Windgeschwindigkeit: {windgeschwindigkeit:.2f} km/h")
246. client.publish(MQTT\_TOPIC\_WIND, f"{windgeschwindigkeit:.2f}")
248. # Windalarm auslösen
249. if windgeschwindigkeit > WIND\_ALARM\_GRENZE:
250. print("STURMALARM! Wind zu stark:", windgeschwindigkeit)
251. client.publish(MQTT\_TOPIC\_WIND\_ALARM, f"ALARM: {windgeschwindigkeit:.2f} km/h")
252. automatik\_aktiv = False
253. wind\_aktiv = True
254. bewege\_aktor(RPWM\_X, LPWM\_X, "einfahren", 25)
255. bewege\_aktor(RPWM\_Y, LPWM\_Y, "einfahren", 25)
257. wind\_impulse = 0
258. lese\_aht10()
259. letzte\_sensorzeit = jetzt
260. client.publish(MQTT\_TOPIC\_WIND\_ALARM, b"ALARM" if wind\_aktiv else b"OK")
262. # Lichtsensoren auslesen und senden
263. l\_links = LDR\_LINKS.read\_u16() \* LDR\_LINKS\_KALIBRIERUNG
264. l\_rechts = LDR\_RECHTS.read\_u16()
265. l\_vorne = LDR\_VORNE.read\_u16()
266. l\_hinten = LDR\_HINTEN.read\_u16()
268. client.publish(MQTT\_TOPIC\_LDR\_LINKS, str(l\_links))
269. client.publish(MQTT\_TOPIC\_LDR\_RECHTS, str(l\_rechts))
270. client.publish(MQTT\_TOPIC\_LDR\_VORNE, str(l\_vorne))
271. client.publish(MQTT\_TOPIC\_LDR\_HINTEN, str(l\_hinten))
273. # Automatische Nachführung aktivieren, falls erlaubt
274. if automatik\_aktiv and windgeschwindigkeit < maximale\_windgeschwindigkeit:
275. if l\_links > l\_rechts + 500:
276. bewege\_aktor(RPWM\_X, LPWM\_X, "ausfahren", 2)
277. elif l\_rechts > l\_links + 500:
278. bewege\_aktor(RPWM\_X, LPWM\_X, "einfahren", 2)
280. if l\_vorne > l\_hinten + 500:
281. bewege\_aktor(RPWM\_Y, LPWM\_Y, "ausfahren", 2)
282. elif l\_hinten > l\_vorne + 500:
283. bewege\_aktor(RPWM\_Y, LPWM\_Y, "einfahren", 2)
285. sleep(0.2)

**Vollständiger Node-RED Code**

[

{

"id": "a90c0e3509fba068",

"type": "tab",

"label": "Solartracker Dashboard",

"disabled": false,

"info": "",

"env": []

},

{

"id": "1b2eaf0b29a08c68",

"type": "mqtt in",

"z": "a90c0e3509fba068",

"name": "",

"topic": "solar/strom",

"qos": "2",

"datatype": "auto-detect",

"broker": "02360d9680e80b99",

"nl": false,

"rap": true,

"rh": 0,

"inputs": 0,

"x": 800,

"y": 40,

"wires": [

[

"140f36693a8861eb"

]

]

},

{

"id": "140f36693a8861eb",

"type": "ui\_gauge",

"z": "a90c0e3509fba068",

"name": "",

"group": "e12877f95ddf8a1e",

"order": 5,

"width": 3,

"height": 3,

"gtype": "donut",

"title": "Strom",

"label": "Ampere",

"format": "{{value}}",

"min": 0,

"max": 10,

"colors": [

"#00b500",

"#e6e600",

"#ca3838"

],

"seg1": "",

"seg2": "",

"diff": false,

"className": "",

"x": 1210,

"y": 40,

"wires": []

},

{

"id": "51a87cafd00b78f7",

"type": "mqtt in",

"z": "a90c0e3509fba068",

"name": "",

"topic": "solar/spannung",

"qos": "2",

"datatype": "auto-detect",

"broker": "02360d9680e80b99",

"nl": false,

"rap": true,

"rh": 0,

"inputs": 0,

"x": 820,

"y": 100,

"wires": [

[

"7d6d2f673aa1df6a",

"aa859aac4affd692",

"a7f8c2678149f624"

]

]

},

{

"id": "3911d49c0dabd271",

"type": "mqtt in",

"z": "a90c0e3509fba068",

"name": "",

"topic": "solar/temperatur",

"qos": "2",

"datatype": "auto-detect",

"broker": "02360d9680e80b99",

"nl": false,

"rap": true,

"rh": 0,

"inputs": 0,

"x": 820,

"y": 240,

"wires": [

[

"e7d8dc55af2483ac",

"f3744699f8c8243f",

"93e3b3c24cf01d7c"

]

]

},

{

"id": "9923aac75bdef6fd",

"type": "mqtt in",

"z": "a90c0e3509fba068",

"name": "",

"topic": "b\"solar/command\"",

"qos": "2",

"datatype": "auto-detect",

"broker": "02360d9680e80b99",

"nl": false,

"rap": true,

"rh": 0,

"inputs": 0,

"x": 490,

"y": 480,

"wires": [

[]

]

},

{

"id": "7d6d2f673aa1df6a",

"type": "ui\_gauge",

"z": "a90c0e3509fba068",

"name": "",

"group": "e12877f95ddf8a1e",

"order": 6,

"width": 3,

"height": 3,

"gtype": "donut",

"title": "Spannung",

"label": "Volt",

"format": "{{value}}",

"min": 0,

"max": "14",

"colors": [

"#00b500",

"#e6e600",

"#ca3838"

],

"seg1": "",

"seg2": "",

"diff": false,

"className": "",

"x": 1230,

"y": 100,

"wires": []

},

{

"id": "e7d8dc55af2483ac",

"type": "ui\_gauge",

"z": "a90c0e3509fba068",

"name": "",

"group": "e12877f95ddf8a1e",

"order": 7,

"width": 3,

"height": 3,

"gtype": "donut",

"title": "Temperatur",

"label": "Celsius",

"format": "{{value}}",

"min": "-10",

"max": "45",

"colors": [

"#00b500",

"#e6e600",

"#ca3838"

],

"seg1": "",

"seg2": "",

"diff": false,

"className": "",

"x": 1230,

"y": 240,

"wires": []

},

{

"id": "mqtt\_ldr\_left",

"type": "mqtt in",

"z": "a90c0e3509fba068",

"name": "LDR Left",

"topic": "solar/ldr/links",

"qos": "0",

"datatype": "auto",

"broker": "02360d9680e80b99",

"nl": false,

"rap": false,

"inputs": 0,

"x": 160,

"y": 40,

"wires": [

[

"ldr\_left\_chart"

]

]

},

{

"id": "mqtt\_ldr\_right",

"type": "mqtt in",

"z": "a90c0e3509fba068",

"name": "LDR Right",

"topic": "solar/ldr/rechts",

"qos": "0",

"datatype": "auto",

"broker": "02360d9680e80b99",

"nl": false,

"rap": false,

"inputs": 0,

"x": 160,

"y": 120,

"wires": [

[

"ldr\_right\_chart"

]

]

},

{

"id": "mqtt\_ldr\_front",

"type": "mqtt in",

"z": "a90c0e3509fba068",

"name": "LDR Front",

"topic": "solar/ldr/vorne",

"qos": "0",

"datatype": "auto",

"broker": "02360d9680e80b99",

"nl": false,

"rap": false,

"inputs": 0,

"x": 160,

"y": 200,

"wires": [

[

"ldr\_front\_chart"

]

]

},

{

"id": "mqtt\_ldr\_back",

"type": "mqtt in",

"z": "a90c0e3509fba068",

"name": "LDR Back",

"topic": "solar/ldr/hinten",

"qos": "0",

"datatype": "auto",

"broker": "02360d9680e80b99",

"nl": false,

"rap": false,

"inputs": 0,

"x": 160,

"y": 280,

"wires": [

[

"ldr\_back\_chart"

]

]

},

{

"id": "ldr\_left\_chart",

"type": "ui\_chart",

"z": "a90c0e3509fba068",

"name": "LDR Links",

"group": "1f5b6626c6ab0e65",

"order": 10,

"width": 3,

"height": 5,

"label": "LDR Links",

"chartType": "bar",

"legend": "false",

"xformat": "HH:mm:ss",

"interpolate": "linear",

"nodata": "",

"dot": false,

"ymin": "0",

"ymax": "45000",

"removeOlder": "5",

"removeOlderPoints": "",

"removeOlderUnit": "60",

"cutout": 0,

"useOneColor": true,

"useUTC": false,

"colors": [

"#1f77b4",

"#000000",

"#000000",

"#000000",

"#000000",

"#000000",

"#000000",

"#000000",

"#000000"

],

"outputs": 1,

"useDifferentColor": false,

"className": "",

"x": 510,

"y": 40,

"wires": [

[]

]

},

{

"id": "ldr\_right\_chart",

"type": "ui\_chart",

"z": "a90c0e3509fba068",

"name": "LDR Rechts",

"group": "1f5b6626c6ab0e65",

"order": 11,

"width": 3,

"height": 5,

"label": "LDR Rechts",

"chartType": "bar",

"legend": "false",

"xformat": "HH:mm:ss",

"interpolate": "linear",

"nodata": "",

"dot": false,

"ymin": "0",

"ymax": "45000",

"removeOlder": 1,

"removeOlderPoints": "",

"removeOlderUnit": "3600",

"cutout": 0,

"useOneColor": true,

"useUTC": false,

"colors": [

"#ff7f0e",

"#000000",

"#000000",

"#000000",

"#000000",

"#000000",

"#000000",

"#000000",

"#000000"

],

"outputs": 1,

"useDifferentColor": false,

"className": "",

"x": 510,

"y": 120,

"wires": [

[]

]

},

{

"id": "ldr\_front\_chart",

"type": "ui\_chart",

"z": "a90c0e3509fba068",

"name": "LDR Vorne",

"group": "1f5b6626c6ab0e65",

"order": 12,

"width": 3,

"height": 5,

"label": "LDR Vorne",

"chartType": "bar",

"legend": "false",

"xformat": "HH:mm:ss",

"interpolate": "linear",

"nodata": "",

"dot": false,

"ymin": "0",

"ymax": "45000",

"removeOlder": 1,

"removeOlderPoints": "",

"removeOlderUnit": "3600",

"cutout": 0,

"useOneColor": true,

"useUTC": false,

"colors": [

"#2ca02c",

"#000000",

"#000000",

"#000000",

"#000000",

"#000000",

"#000000",

"#000000",

"#000000"

],

"outputs": 1,

"useDifferentColor": false,

"className": "",

"x": 510,

"y": 200,

"wires": [

[]

]

},

{

"id": "ldr\_back\_chart",

"type": "ui\_chart",

"z": "a90c0e3509fba068",

"name": "LDR Hinten",

"group": "1f5b6626c6ab0e65",

"order": 13,

"width": 3,

"height": 5,

"label": "LDR Hinten",

"chartType": "bar",

"legend": "false",

"xformat": "HH:mm:ss",

"interpolate": "linear",

"nodata": "",

"dot": false,

"ymin": "0",

"ymax": "45000",

"removeOlder": 1,

"removeOlderPoints": "",

"removeOlderUnit": "3600",

"cutout": 0,

"useOneColor": true,

"useUTC": false,

"colors": [

"#d62728",

"#000000",

"#000000",

"#000000",

"#000000",

"#000000",

"#000000",

"#000000",

"#000000"

],

"outputs": 1,

"useDifferentColor": false,

"className": "",

"x": 510,

"y": 280,

"wires": [

[]

]

},

{

"id": "mqtt\_wind",

"type": "mqtt in",

"z": "a90c0e3509fba068",

"name": "Wind",

"topic": "solar/wind",

"qos": "2",

"datatype": "auto",

"broker": "02360d9680e80b99",

"nl": false,

"rap": false,

"inputs": 0,

"x": 790,

"y": 380,

"wires": [

[

"gauge\_wind",

"bf15290f56b67771",

"9407f126a4108c24"

]

]

},

{

"id": "gauge\_wind",

"type": "ui\_gauge",

"z": "a90c0e3509fba068",

"name": "",

"group": "e12877f95ddf8a1e",

"order": 16,

"width": "6",

"height": "6",

"gtype": "gage",

"title": "Wind (Kmh)",

"label": "kmh",

"format": "{{value}}",

"min": 0,

"max": 30,

"colors": [

"#00b500",

"#e6e600",

"#ca3838"

],

"seg1": "",

"seg2": "",

"diff": false,

"className": "",

"x": 1230,

"y": 380,

"wires": []

},

{

"id": "93ce75cd47d6b1bd",

"type": "mqtt out",

"z": "a90c0e3509fba068",

"name": "",

"topic": "solar/befehl/bewege/x/links",

"qos": "",

"retain": "",

"respTopic": "",

"contentType": "",

"userProps": "",

"correl": "",

"expiry": "",

"broker": "02360d9680e80b99",

"x": 520,

"y": 560,

"wires": []

},

{

"id": "dd3e54b6164cf280",

"type": "mqtt out",

"z": "a90c0e3509fba068",

"name": "",

"topic": "solar/befehl/bewege/x/rechts",

"qos": "",

"retain": "",

"respTopic": "",

"contentType": "",

"userProps": "",

"correl": "",

"expiry": "",

"broker": "02360d9680e80b99",

"x": 520,

"y": 800,

"wires": []

},

{

"id": "a619f78bd4c0d1f9",

"type": "mqtt out",

"z": "a90c0e3509fba068",

"name": "",

"topic": "solar/befehl/bewege/y/hoch",

"qos": "",

"retain": "",

"respTopic": "",

"contentType": "",

"userProps": "",

"correl": "",

"expiry": "",

"broker": "02360d9680e80b99",

"x": 520,

"y": 720,

"wires": []

},

{

"id": "7bbaa4e706afcbe2",

"type": "mqtt out",

"z": "a90c0e3509fba068",

"name": "",

"topic": "solar/befehl/bewege/y/runter",

"qos": "",

"retain": "",

"respTopic": "",

"contentType": "",

"userProps": "",

"correl": "",

"expiry": "",

"broker": "02360d9680e80b99",

"x": 520,

"y": 640,

"wires": []

},

{

"id": "d417d6dae7151e6a",

"type": "ui\_button",

"z": "a90c0e3509fba068",

"name": "Bewegung Runter",

"group": "b40705d5933c9660",

"order": 2,

"width": 3,

"height": 3,

"passthru": false,

"label": "Bewegung Runter",

"tooltip": "",

"color": "",

"bgcolor": "",

"className": "",

"icon": "",

"payload": "",

"payloadType": "str",

"topic": "solar/command/move/x/right",

"topicType": "msg",

"x": 170,

"y": 640,

"wires": [

[

"7bbaa4e706afcbe2"

]

]

},

{

"id": "0e8956a7716af261",

"type": "ui\_button",

"z": "a90c0e3509fba068",

"name": "Bewegung Hoch",

"group": "b40705d5933c9660",

"order": 1,

"width": 3,

"height": 3,

"passthru": false,

"label": "Bewegung Hoch",

"tooltip": "",

"color": "",

"bgcolor": "",

"className": "",

"icon": "",

"payload": "",

"payloadType": "str",

"topic": "solar/command/move/y/up",

"topicType": "str",

"x": 160,

"y": 720,

"wires": [

[

"a619f78bd4c0d1f9"

]

]

},

{

"id": "6ed6c52eb775490e",

"type": "ui\_button",

"z": "a90c0e3509fba068",

"name": "Bewegung Rechts",

"group": "b40705d5933c9660",

"order": 4,

"width": 3,

"height": 3,

"passthru": false,

"label": "Bewegung Rechts",

"tooltip": "",

"color": "",

"bgcolor": "",

"className": "",

"icon": "",

"payload": "",

"payloadType": "str",

"topic": "solar/command/move/y/down",

"topicType": "str",

"x": 170,

"y": 800,

"wires": [

[

"dd3e54b6164cf280"

]

]

},

{

"id": "d17e4825a8c4399a",

"type": "ui\_button",

"z": "a90c0e3509fba068",

"name": "Bewegung Links",

"group": "b40705d5933c9660",

"order": 3,

"width": 3,

"height": 3,

"passthru": false,

"label": "Bewegung Links",

"tooltip": "",

"color": "",

"bgcolor": "",

"className": "",

"icon": "",

"payload": "",

"payloadType": "str",

"topic": "solar/command/move/x/left",

"topicType": "msg",

"x": 160,

"y": 560,

"wires": [

[

"93ce75cd47d6b1bd"

]

]

},

{

"id": "switch\_auto",

"type": "ui\_switch",

"z": "a90c0e3509fba068",

"name": "Automatiksteuerung",

"label": "Automatik",

"tooltip": "Automatische Nachführung aktivieren",

"group": "b40705d5933c9660",

"order": 5,

"width": 4,

"height": 1,

"passthru": true,

"decouple": "false",

"topic": "solar/auto\_control",

"topicType": "str",

"style": "",

"onvalue": "on",

"onvalueType": "str",

"onicon": "",

"oncolor": "green",

"offvalue": "off",

"offvalueType": "str",

"officon": "",

"offcolor": "red",

"animate": true,

"className": "",

"x": 180,

"y": 1140,

"wires": [

[

"a66159b8a4eb7548"

]

]

},

{

"id": "0a98e57d2f0c406c",

"type": "remote-access",

"z": "a90c0e3509fba068",

"confignode": "2b7c7972d8e45429",

"name": "solar",

"verbose": 0,

"x": 130,

"y": 480,

"wires": [

[],

[]

]

},

{

"id": "a66159b8a4eb7548",

"type": "mqtt out",

"z": "a90c0e3509fba068",

"name": "",

"topic": "solar/auto\_control",

"qos": "0",

"retain": "",

"respTopic": "",

"contentType": "",

"userProps": "",

"correl": "",

"expiry": "",

"broker": "02360d9680e80b99",

"x": 510,

"y": 1140,

"wires": []

},

{

"id": "a5df99cb1880d1ce",

"type": "mqtt in",

"z": "a90c0e3509fba068",

"name": "",

"topic": "solar/luftfeuchte",

"qos": "2",

"datatype": "auto-detect",

"broker": "02360d9680e80b99",

"nl": false,

"rap": true,

"rh": 0,

"inputs": 0,

"x": 820,

"y": 520,

"wires": [

[

"c06c7922f65b1a32",

"54976f5c8fb85912",

"13431a140fb4c518"

]

]

},

{

"id": "c06c7922f65b1a32",

"type": "ui\_gauge",

"z": "a90c0e3509fba068",

"name": "",

"group": "e12877f95ddf8a1e",

"order": 5,

"width": "3",

"height": "3",

"gtype": "donut",

"title": "Luftfeuchtigkeit",

"label": "Feuchte",

"format": "{{value}}",

"min": 0,

"max": "100",

"colors": [

"#00b500",

"#e6e600",

"#ca3838"

],

"seg1": "",

"seg2": "",

"diff": false,

"className": "",

"x": 1240,

"y": 520,

"wires": []

},

{

"id": "aa859aac4affd692",

"type": "function",

"z": "a90c0e3509fba068",

"name": "function 1",

"func": "msg.url = \"https://api.thingspeak.com/update?api\_key=UNN8RH7604U1OH7V&field2=0\" + msg.payload\nreturn msg;",

"outputs": 1,

"timeout": 0,

"noerr": 0,

"initialize": "",

"finalize": "",

"libs": [],

"x": 1040,

"y": 180,

"wires": [

[

"701f1ab7f9a2dc2d"

]

]

},

{

"id": "f3744699f8c8243f",

"type": "function",

"z": "a90c0e3509fba068",

"name": "function 2",

"func": "msg.url = \"https://api.thingspeak.com/update?api\_key=UNN8RH7604U1OH7V&field3=0\" + msg.payload\nreturn msg;",

"outputs": 1,

"timeout": 0,

"noerr": 0,

"initialize": "",

"finalize": "",

"libs": [],

"x": 1040,

"y": 320,

"wires": [

[

"4d4854f21d79f946"

]

]

},

{

"id": "bf15290f56b67771",

"type": "function",

"z": "a90c0e3509fba068",

"name": "function 3",

"func": "msg.url = \"https://api.thingspeak.com/update?api\_key=UNN8RH7604U1OH7V&field1=0\" + msg.payload\nreturn msg;",

"outputs": 1,

"timeout": 0,

"noerr": 0,

"initialize": "",

"finalize": "",

"libs": [],

"x": 1040,

"y": 460,

"wires": [

[

"c277db90c74f486c"

]

]

},

{

"id": "701f1ab7f9a2dc2d",

"type": "http request",

"z": "a90c0e3509fba068",

"name": "",

"method": "GET",

"ret": "txt",

"paytoqs": "ignore",

"url": "",

"tls": "",

"persist": false,

"proxy": "",

"insecureHTTPParser": false,

"authType": "",

"senderr": false,

"headers": [],

"x": 1230,

"y": 180,

"wires": [

[]

]

},

{

"id": "4d4854f21d79f946",

"type": "http request",

"z": "a90c0e3509fba068",

"name": "",

"method": "GET",

"ret": "txt",

"paytoqs": false,

"url": "",

"persist": false,

"insecureHTTPParser": false,

"authType": "",

"senderr": false,

"headers": [],

"x": 1230,

"y": 320,

"wires": [

[]

]

},

{

"id": "c277db90c74f486c",

"type": "http request",

"z": "a90c0e3509fba068",

"name": "",

"method": "GET",

"ret": "txt",

"paytoqs": false,

"url": "",

"persist": false,

"insecureHTTPParser": false,

"authType": "",

"senderr": false,

"headers": [],

"x": 1230,

"y": 460,

"wires": [

[]

]

},

{

"id": "54976f5c8fb85912",

"type": "function",

"z": "a90c0e3509fba068",

"name": "function 4",

"func": "msg.url = \"https://api.thingspeak.com/update?api\_key=UNN8RH7604U1OH7V&field4=0\" + msg.payload\nreturn msg;",

"outputs": 1,

"timeout": 0,

"noerr": 0,

"initialize": "",

"finalize": "",

"libs": [],

"x": 1040,

"y": 600,

"wires": [

[

"9035e27c65d3d43a"

]

]

},

{

"id": "9035e27c65d3d43a",

"type": "http request",

"z": "a90c0e3509fba068",

"name": "",

"method": "GET",

"ret": "txt",

"paytoqs": "ignore",

"url": "",

"tls": "",

"persist": false,

"proxy": "",

"insecureHTTPParser": false,

"authType": "",

"senderr": false,

"headers": [],

"x": 1230,

"y": 600,

"wires": [

[]

]

},

{

"id": "MQTT\_IN\_WIND\_ALARM",

"type": "mqtt in",

"z": "a90c0e3509fba068",

"name": "Wind Alarm",

"topic": "solar/wind/alarm",

"qos": "0",

"datatype": "auto-detect",

"broker": "02360d9680e80b99",

"nl": false,

"rap": false,

"rh": 0,

"inputs": 0,

"x": 810,

"y": 660,

"wires": [

[

"e4c55208f3ee686d"

]

]

},

{

"id": "7c478ac5ce05f4b5",

"type": "ui\_button",

"z": "a90c0e3509fba068",

"name": "Sturm-Alarm Rücksetzung",

"group": "b40705d5933c9660",

"order": 6,

"width": "4",

"height": "2",

"passthru": false,

"label": " Sturm-Alarm reset",

"tooltip": "Setzt den Wind-Alarm zurück",

"color": "rot",

"bgcolor": "#0094ce",

"className": "",

"icon": "reset",

"payload": "reset",

"payloadType": "str",

"topic": "solar/wind/reset",

"topicType": "str",

"x": 850,

"y": 720,

"wires": [

[

"68173facee4e3666"

]

]

},

{

"id": "68173facee4e3666",

"type": "mqtt out",

"z": "a90c0e3509fba068",

"name": "Wind Rücksetzung",

"topic": "solar/wind/zuruecksetzen",

"qos": "0",

"retain": "false",

"respTopic": "",

"contentType": "",

"userProps": "",

"correl": "",

"expiry": "",

"broker": "02360d9680e80b99",

"x": 1110,

"y": 720,

"wires": []

},

{

"id": "6151bd381af875a2",

"type": "ui\_led",

"z": "a90c0e3509fba068",

"order": 7,

"group": "b40705d5933c9660",

"width": "2",

"height": "4",

"label": "",

"labelPlacement": "right",

"labelAlignment": "left",

"colorForValue": [

{

"color": "#fffafa",

"value": "false",

"valueType": "bool"

},

{

"color": "#fb0909",

"value": "true",

"valueType": "bool"

}

],

"allowColorForValueInMessage": false,

"shape": "circle",

"showGlow": true,

"name": "",

"x": 1210,

"y": 660,

"wires": []

},

{

"id": "e4c55208f3ee686d",

"type": "function",

"z": "a90c0e3509fba068",

"name": "function 5",

"func": "var interval = context.get('interval') || null;\n\nif (msg.payload === \"ALARM\") {\n if (!interval) {\n interval = setInterval(function () {\n node.send({ payload: true });\n setTimeout(function () {\n node.send({ payload: false });\n }, 300); // LED 300ms an, dann aus\n }, 800); // alle 800ms blinken\n context.set('interval', interval);\n }\n} else if (msg.payload === \"OK\") {\n if (interval) {\n clearInterval(interval);\n context.set('interval', null);\n }\n node.send({ payload: false }); // LED aus\n}\n\nreturn null;\n",

"outputs": 1,

"timeout": 0,

"noerr": 0,

"initialize": "",

"finalize": "",

"libs": [],

"x": 1000,

"y": 660,

"wires": [

[

"6151bd381af875a2"

]

]

},

{

"id": "f6d8c9052e9309aa",

"type": "ui\_numeric",

"z": "a90c0e3509fba068",

"name": "Max Wind Geschwindigkeit",

"label": "Maximale Windgeschwindigkeit",

"tooltip": "Setze den Schwellenwert zur Windabschaltung",

"group": "b40705d5933c9660",

"order": 8,

"width": 6,

"height": "2",

"wrap": false,

"passthru": true,

"topic": "",

"topicType": "str",

"format": "{{value}} km/h",

"min": 5,

"max": "50",

"step": "1",

"className": "",

"x": 860,

"y": 800,

"wires": [

[

"ce057e1bfb03bdaa"

]

]

},

{

"id": "ce057e1bfb03bdaa",

"type": "function",

"z": "a90c0e3509fba068",

"name": "Setze MQTT Payload",

"func": "let max\_wind\_speed = parseFloat(msg.payload);\n\nif (!isNaN(max\_wind\_speed)) {\n msg.topic = \"solar/wind/max\_speed\";\n msg.payload = max\_wind\_speed.toFixed(1);\n return msg;\n} else {\n return null;\n}",

"outputs": 1,

"noerr": 0,

"initialize": "",

"finalize": "",

"libs": [],

"x": 1100,

"y": 800,

"wires": [

[

"ef7d9f59b7adfcaa"

]

]

},

{

"id": "ef7d9f59b7adfcaa",

"type": "mqtt out",

"z": "a90c0e3509fba068",

"name": " Publish Maximale Windgeschwindigkeit",

"topic": "solar/wind/maximal",

"qos": "",

"retain": "",

"respTopic": "",

"contentType": "",

"userProps": "",

"correl": "",

"expiry": "",

"broker": "02360d9680e80b99",

"x": 1400,

"y": 800,

"wires": []

},

{

"id": "3803c3a7e4a85790",

"type": "mysql",

"z": "a90c0e3509fba068",

"mydb": "",

"name": "",

"x": 1210,

"y": 140,

"wires": [

[]

]

},

{

"id": "fc0fb790120f62fb",

"type": "mysql",

"z": "a90c0e3509fba068",

"name": "",

"x": 1210,

"y": 280,

"wires": [

[]

]

},

{

"id": "6c9aa4976c23fd13",

"type": "mysql",

"z": "a90c0e3509fba068",

"name": "",

"x": 1210,

"y": 420,

"wires": [

[]

]

},

{

"id": "f9d5155a84b28b77",

"type": "mysql",

"z": "a90c0e3509fba068",

"name": "",

"x": 1210,

"y": 560,

"wires": [

[]

]

},

{

"id": "a7f8c2678149f624",

"type": "function",

"z": "a90c0e3509fba068",

"name": "function 6",

"func": "\nreturn msg;",

"outputs": 1,

"timeout": 0,

"noerr": 0,

"initialize": "",

"finalize": "",

"libs": [],

"x": 1040,

"y": 140,

"wires": [

[

"3803c3a7e4a85790"

]

]

},

{

"id": "93e3b3c24cf01d7c",

"type": "function",

"z": "a90c0e3509fba068",

"name": "function 7",

"func": "\nreturn msg;",

"outputs": 1,

"timeout": 0,

"noerr": 0,

"initialize": "",

"finalize": "",

"libs": [],

"x": 1040,

"y": 280,

"wires": [

[

"fc0fb790120f62fb"

]

]

},

{

"id": "9407f126a4108c24",

"type": "function",

"z": "a90c0e3509fba068",

"name": "function 8",

"func": "\nreturn msg;",

"outputs": 1,

"timeout": 0,

"noerr": 0,

"initialize": "",

"finalize": "",

"libs": [],

"x": 1040,

"y": 420,

"wires": [

[

"6c9aa4976c23fd13"

]

]

},

{

"id": "13431a140fb4c518",

"type": "function",

"z": "a90c0e3509fba068",

"name": "function 9",

"func": "\nreturn msg;",

"outputs": 1,

"timeout": 0,

"noerr": 0,

"initialize": "",

"finalize": "",

"libs": [],

"x": 1040,

"y": 560,

"wires": [

[

"f9d5155a84b28b77"

]

]

},

{

"id": "586e9f9c4f090d9c",

"type": "ui\_slider",

"z": "a90c0e3509fba068",

"name": "LDR Left Kalibration",

"label": "LDR LEFT Kalibrierung",

"tooltip": "Anpassung des LDR-Linksensors",

"group": "1f5b6626c6ab0e65",

"order": 1,

"width": 6,

"height": 1,

"passthru": true,

"outs": "all",

"topic": "",

"topicType": "str",

"min": 0.8,

"max": 1.2,

"step": 0.01,

"className": "",

"x": 180,

"y": 360,

"wires": [

[

"88fcd0371ecf99f0",

"e343fc183d772e3a"

]

]

},

{

"id": "88fcd0371ecf99f0",

"type": "mqtt out",

"z": "a90c0e3509fba068",

"name": "MQTT: LDR\_LEFT\_CAL",

"topic": "solar/ldr/links/kalibrierung",

"qos": "",

"retain": false,

"respTopic": "",

"contentType": "",

"userProps": "",

"correl": "",

"expiry": "",

"broker": "02360d9680e80b99",

"x": 510,

"y": 360,

"wires": []

},

{

"id": "e343fc183d772e3a",

"type": "ui\_text",

"z": "a90c0e3509fba068",

"group": "1f5b6626c6ab0e65",

"order": 2,

"width": 6,

"height": 1,

"name": "Kalibrierwert anzeigen",

"label": "Aktueller Faktor",

"format": "{{msg.payload}}",

"layout": "row-spread",

"className": "",

"style": false,

"font": "",

"fontSize": "",

"color": "#000000",

"x": 500,

"y": 420,

"wires": []

},

{

"id": "02360d9680e80b99",

"type": "mqtt-broker",

"name": "Solarnachführung1",

"broker": "192.168.178.24",

"port": 1883,

"clientid": "",

"autoConnect": true,

"usetls": false,

"protocolVersion": 4,

"keepalive": 60,

"cleansession": true,

"autoUnsubscribe": true,

"birthTopic": "",

"birthQos": "0",

"birthRetain": "false",

"birthPayload": "",

"birthMsg": {},

"closeTopic": "",

"closeQos": "0",

"closeRetain": "false",

"closePayload": "",

"closeMsg": {},

"willTopic": "",

"willQos": "0",

"willRetain": "false",

"willPayload": "",

"willMsg": {},

"userProps": "",

"sessionExpiry": ""

},

{

"id": "e12877f95ddf8a1e",

"type": "ui\_group",

"name": "Daten Solarnachführung",

"tab": "4989931b3e3bf6e0",

"order": 1,

"disp": true,

"width": 6,

"collapse": false,

"className": ""

},

{

"id": "1f5b6626c6ab0e65",

"type": "ui\_group",

"name": "LDR Anzeigen",

"tab": "4989931b3e3bf6e0",

"order": 3,

"disp": true,

"width": 6,

"collapse": false,

"className": ""

},

{

"id": "b40705d5933c9660",

"type": "ui\_group",

"name": "Manuelle Steuerung",

"tab": "4989931b3e3bf6e0",

"order": 2,

"disp": true,

"width": 6,

"collapse": false,

"className": ""

},

{

"id": "2b7c7972d8e45429",

"type": "remote-config",

"name": "Solarnachführung",

"host": "localhost",

"protocol": "http",

"port": "1880",

"baseurl": "/ui",

"instancehash": "43t9nxa7rz8zl5bp8zxxu1st7ejo5gg03ad98xk7apr53mtbttvvq3l8gqmvkrur",

"server": "nodered04.remote-red.com",

"region": "de"

},

{

"id": "4989931b3e3bf6e0",

"type": "ui\_tab",

"name": "Home",

"icon": "dashboard",

"disabled": false,

"hidden": false

}

]