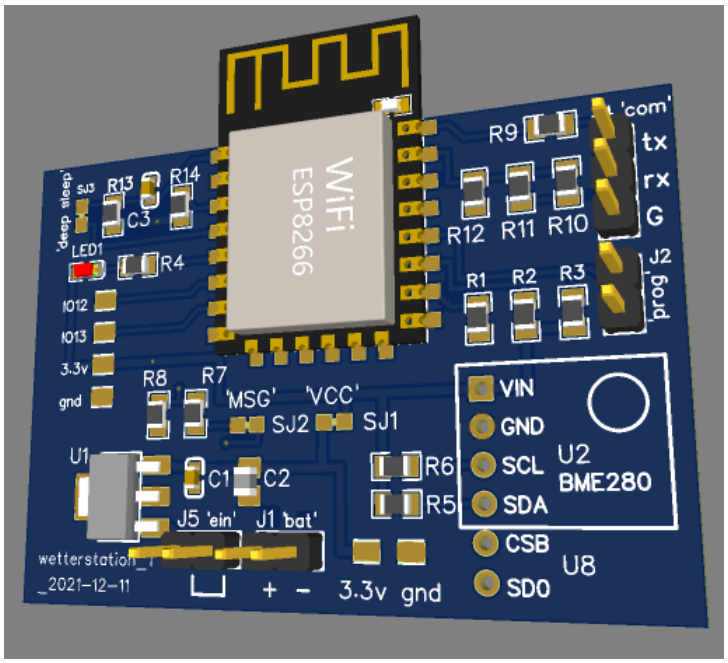
Wetterstation
Anleitung

31.01.2022 H.Schulte

Aufbauanleitung:

Als erstes werden die niedrigen Bauelemente R und C bestückt,dann das Wifi-Modul und der Spannungregler U1. Beim Wifi-Modul müssen die ‚südlichen' 6 Pins nicht verlötet werden. Als Steckverbinder Jx werden Anreih-Stiftleisten verwendet. Die LED1 zeigt mit ihrem Pfeil nach rechts. Das BME-Modul wird als letztes flach liegend aufgelötet.



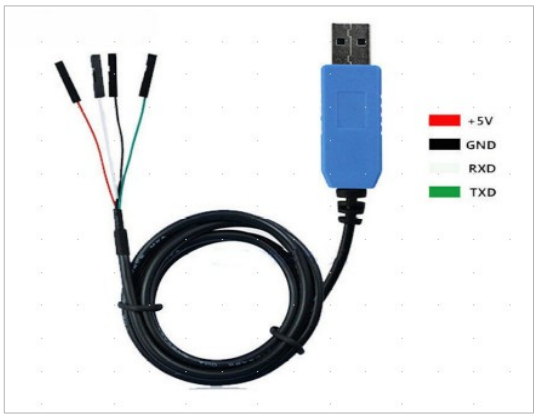
Stromversorgung messen und freigeben:

An J1 ‚bat' wird eine Batterie mit 3 AA- Zellen angeschlossen (4,5 Volt). An den Messpunkten 3.3V und gnd müssen jetzt 3.3 Volt zu messen sein. Jetzt wird die Brücke SJ1 ‚VCC' eingelötet (Lötklecks).

Lötbrücken:

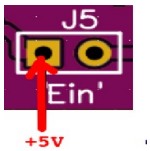
Zum Schluss werden die Lötbrücken SJ2 ‚MSG' und SJ3 ‚deep sleep' eingelötet.
Mit ‚MSG' wird die Messung der Batteriespannung freigegeben. Mit ‚deep sleep' kann der Prozessor sich nach der Schlafpause selbst wieder aufwecken (deep sleep reset).

Anschluss USB- Seriell-Wandler:



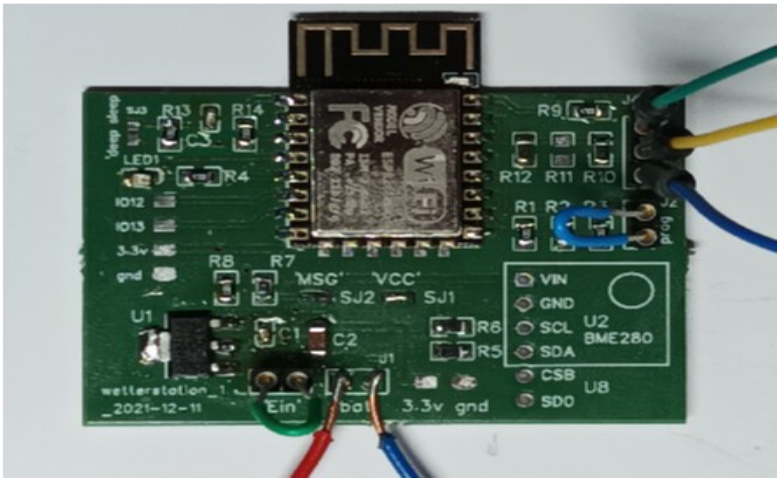
Adapter	--- J4 Pin
GND ,schwarz	--- G
TXD ,grün	--- rx
RXD ,weiß	--- tx
+5v	--- : Dieser Pin kann als Stromversorgung für das Board genommen werden;

Anschluss an den linken Pin von J5.



Falls ein anderer USB-Serial-Wandler keine 5V zur Verfügung stellt, dann wird die Batterie verwendet(Brücke J5). Wichtig: RX geht an TX der anderen Seite und TX geht an RX der anderen Seite.

Vorbereitung zum Flashen:



IDE ‚Thonny' installieren:
https://thonny.org/

Micropython für ESP8266 herunterladen:
https://micropython.org/resources/firmware/esp8266-20220117-v1.18.bin

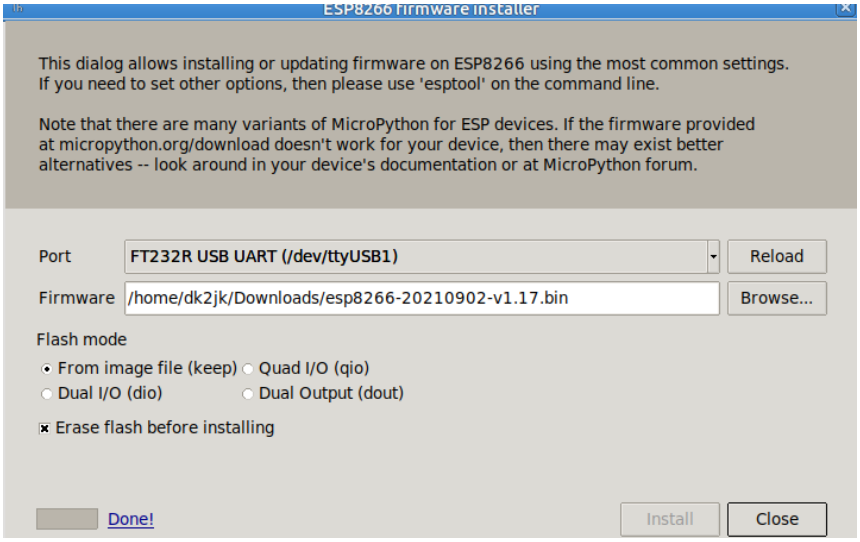
(Version kann sich ändern)

Die Brücke J2 (‚prog') wird verbunden. Erst danach wird die Stromversorgung J5 zugeschaltet (Reset).
Die Platine befindet sich nun im ‚Flash-Mode'.

Flashen:

Das Programm Thonny starten.
Unter Tools – Options – Interpreter ‚Micropython (ESP8266)' einstellen.
Weiter unten im Fenster ‚install firmware' starten.
Jetzt Port und Firmware im Dateimanager wählen ; die Firmware haben wir unter ‚Download Micropython' heruntergeladen.
Nun ‚Install' anwählen und warten; es sollte eine Fortschrittsmeldung (s.u.) kommen.

Nicht vergessen, die Brücke J2 (‚prog') wieder zu trennen.



Meldungen beim Flashen:

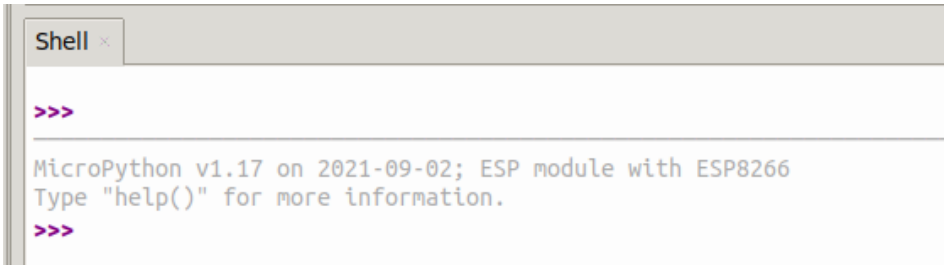
```
/usr/bin/python3 -u -m esptool --port /dev/ttyUSB1 erase_flash
esptool.py v3.1
Serial port /dev/ttyUSB1
Connecting....
...
Erasing done
-----------------------------------

/usr/bin/python3 -u -m esptool --port /dev/ttyUSB1 write_flash --flash_mode keep --flash_size
detect 0x0 /home/dk2jk/Downloads/esp8266-20210902-v1.17.bin
esptool.py v3.1
Serial port /dev/ttyUSB1
Connecting...
Chip is ESP8266EX
Features: WiFi
Crystal is 26MHz
MAC: bc:ff:4d:81:5c:04
Auto-detected Flash size: 4MB
Flash will be erased from 0x00000000 to 0x0009afff...
Flash params set to 0x0040
Compressed 633688 bytes to 416262...
Writing at 0x00000000... (3 %)
...
Writing at 0x00091598... (96 %)
Writing at 0x00098493... (100 %)
Wrote 633688 bytes (416262 compressed) at 0x00000000 in 37.0 seconds (effective 137.1
kbit/s)...
Hash of data verified.

Leaving...
Hard resetting via RTS pin...
Done!
```

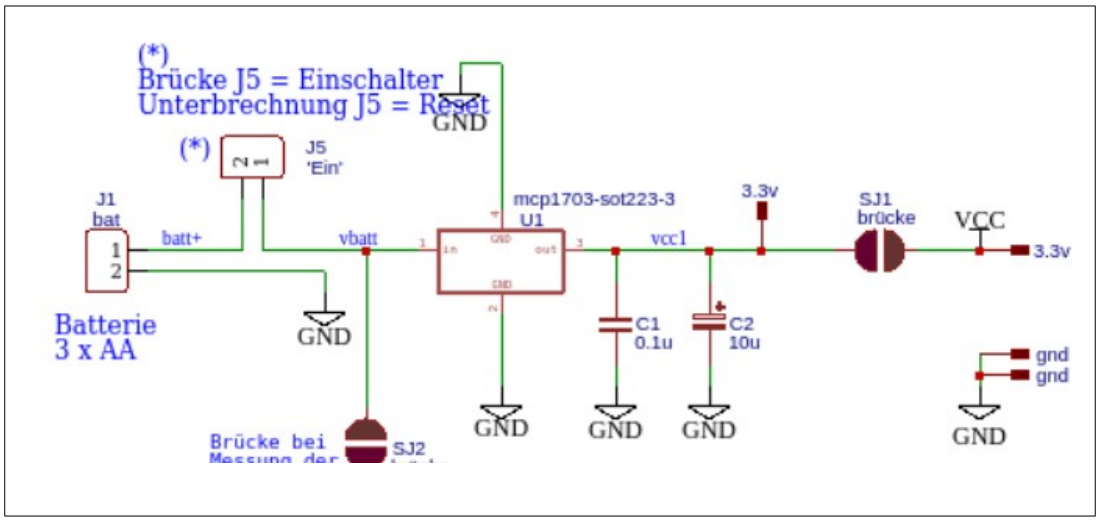
Start:

Nach einen Reset(Spannungsunterbrechung J5) zeigt sich auf dem Thonny-Terminal (Shell):

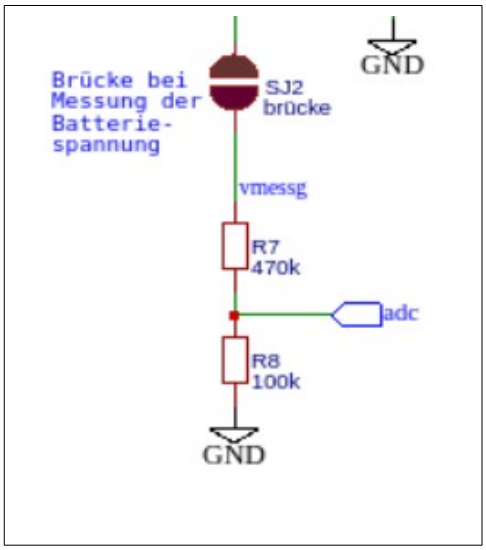


oder ähnlich. Das Board kann jetzt über die serielle Schnittstelle in Micropython programmiert werden. Wir benutzen dazu die Thonny-Bedienoberfläche.

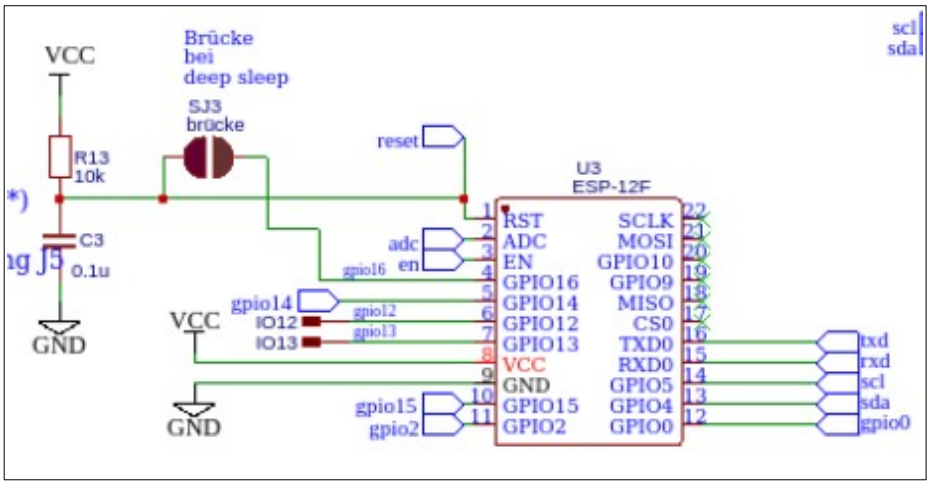
Schaltungsbeschreibung:



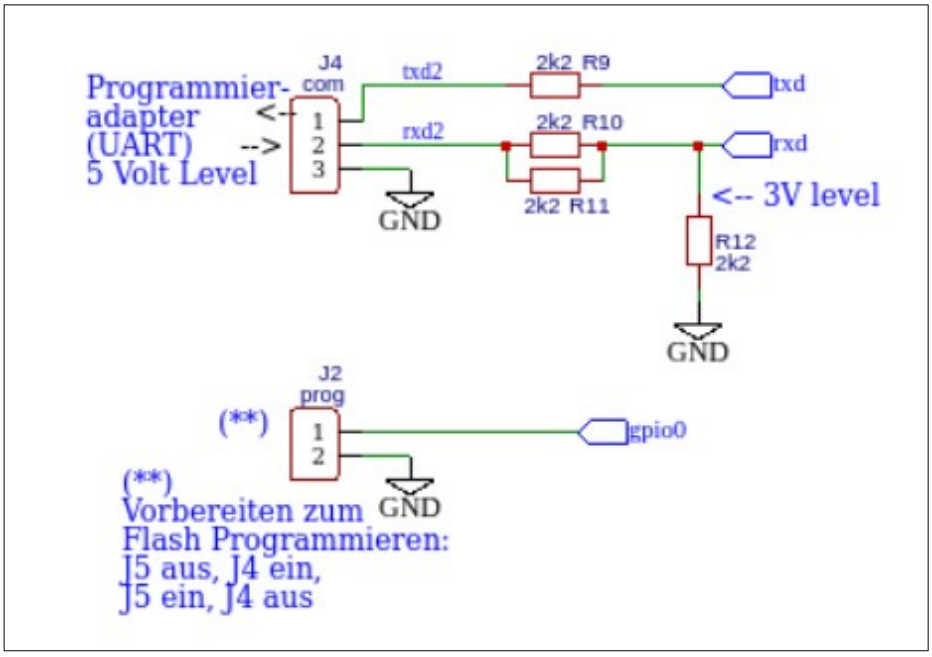
Die Stromversorgung erfolgt über eine Batterie mit 3 AA- Zellen (4,5 Volt). Der Spannungsregler MCP1703 erzeugt daraus die Betriebsspannug des Boards. Dieser Spannungsregler hat die besondere Eigenschaft, dass er im Ruhezustand – wenn keine Last anliegt- besonders wenig Strom verbraucht (einige uA). Die Brücke SJ1 dient dazu, dass die Spannung kontrolliert werden kann, bevor der Rest der Schaltung in Betrieb geht. Hier kann man auch den Strom messen. Im Betrieb ist die Brücke geschlossen (Lötklecks).



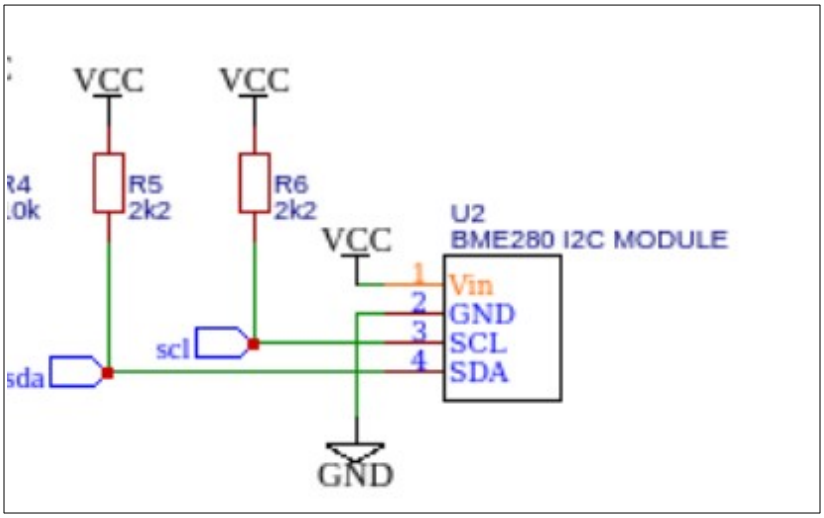
Wenn die Brücke SJ2 geschlossen ist , kann der Controller über seinen AD-Wandler die Batteriespannung messen. Der maximal zulässige Spannungswert an ‚adc‘ ist 1,00 Volt.



Das Mikroprozessor-Modul wird mit 3.3 Volt = VCC versorgt. Der Reset des Prozessors wird durch aus- und einschalten ausgelöst. Die Brücke SJ3 muss geschlossen werden, wenn der Prozessor sich nach einem Tiefschlaf selbst wieder aufwecken soll; dies geschieht am Ende der Tiefschlafphase vom internen Timer über einen Impuls an GPIO16.



Um den Prozessor zu programmieren , wird eine seriellen Schnittstelle mit den Signalen txd und rxd verwendet. Der Spannungsteiler R10, R12 vermindert die RXD-Spannung auf einen zulässigen Wert, falls mit 5V angesteuert werden sollte.
Um den Prozessor in den Flash- Modus zu versetzen, muss vor dem Einschalten die Brücke J2 ‚prog‘ gesteckt werden. In diesem Modus wird der Micropython-Interpreter geladen; das braucht man nur einmal machen.



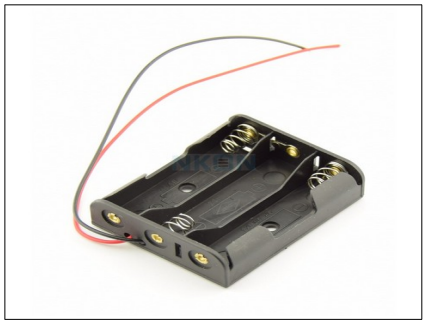
Auf der Platine befindet sich ein Modul ‚BME280‘; dies misst Temperatur, Luftdruck und rel. Luftfeuchtigtkeit. Das Modul wird über I2C-Bus angesteuert (Signale scl, sda).
Das Layout ist für 4-Pin- und 6-Pin- Module ausgelegt.

Gehäuse:

Als Gehäuse kann z.B. eine Feuchtraum-Abzweigdose „OBO A8“ 75x75x37 verwendet werden.

Batterie:

Als Batterie werden 3 x AA Zellen verwendet; diese befinden sich in einem Batteriefach mit Anschlussdrähten:



Dieses Fach wird mit der glatten Seite nach oben in das Gehäuse gelegt; darauf wird dann die Sensorplatine gelegt oder geklebt. Die Anschlussdrähte werden fest mit der Platine verbunden (Anschluss ‚bat‘)

Skripts laden:

Für unsere Wetterstation benötigen wir folgende Skripts (= Python-Programme) im Prozessor-Filesystem:

```
.
├── batterie.py
├── BME280_1.py
├── boot.py
├── client.py
├── i2c_scan.py **)
├── main.py
├── reset.py
├── send_thingspeak2.py
├── test.py     **)
└── wdt.py
```

Inbetriebnahme:

Im Modul "main.py" ist

##import client

sollte zunächst auskommentiert sein, dadurch hält der Prozessor nach Reset hier an.

Der Start des Moduls *"client.py" zeigt zunächst Fehler an; bei*

WiFi_SSID, WiFi_PW= "ssid","pw"

müssen fuer den Router passende WLAN-Daten angegeben werden.

Ein Neustart von *"client.py" sollte jetzt*

'verbunden mit "{}"' kommen.

Wird 'timeout' angezeigt, dann ging etwas schief.

Wenn die Verbindung steht, laden wir das Modul "send_thingspeak2" und starten dieses. Dadurch wird die Funktion ‚send()' aufgerufen und Daten werden an Thingspeak gesendet.

Für den eigenen Datenpool muss noch die Thingspeak Schlüssel zum Schreiben angepasst werden:

```
WRITE_API_KEY = 'xxxxxx'
```

Autostart

Wenn die Verbindung zum WLAN und zu Thingspeak funktioniert, so können wir auf automatischen Betrieb umstellen. Dazu ist folgendes zu einzustellen:

Hardware:
RST mit GPIO16 mit Lötklecks verbinden (Brücke SJ3, siehe Schaltplan)

Wenn "client.py" von der Oberfläche gestartet wird, so ist die Schlafzeit ausgeschaltet und es folgt kein Schlaf.

In "main.py"

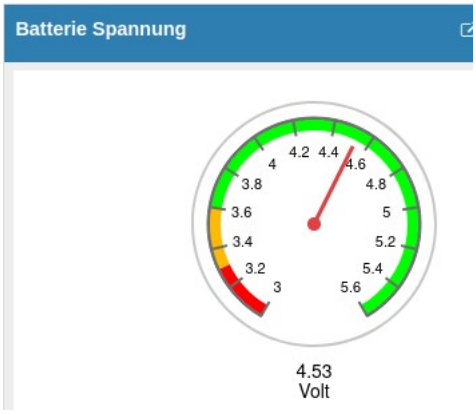
```
##import client   # Kommentar entfernen
```

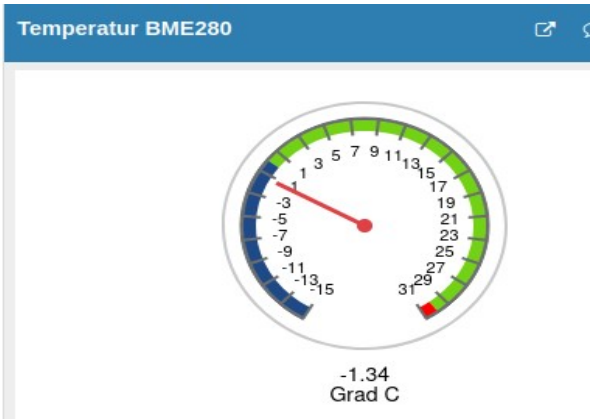
Der Prozessor startet nun nach Reset oder Tiefschlaf über ‚boot' und ‚main' das Modul ‚client' und legt sich dann wieder 10 Minuten schlafen.

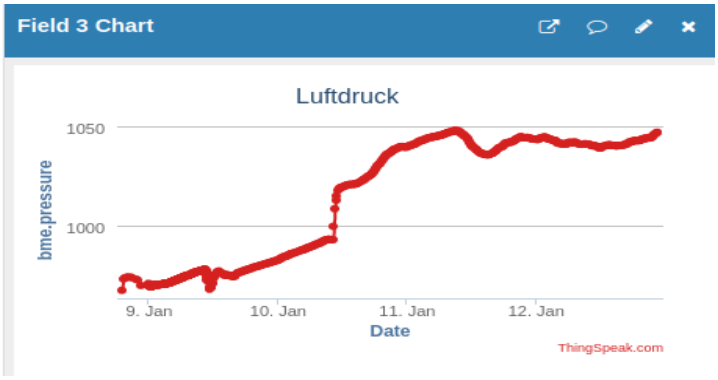
Damit die Batteriespannung gemessen werden kann, muss die Brücke SJ2 eingelötet sein. So kann man die Entladung der Batterie verfolgen; da auch dies wieder Strom verbraucht, kann man die Brücke nach der Testphase wieder entfernen.

Darstellung der Daten auf Thingspeak

Auf dem Thingspeak Server gibt es diverse Möglichkeiten, Daten grafisch darzustellen.
Z.B.:

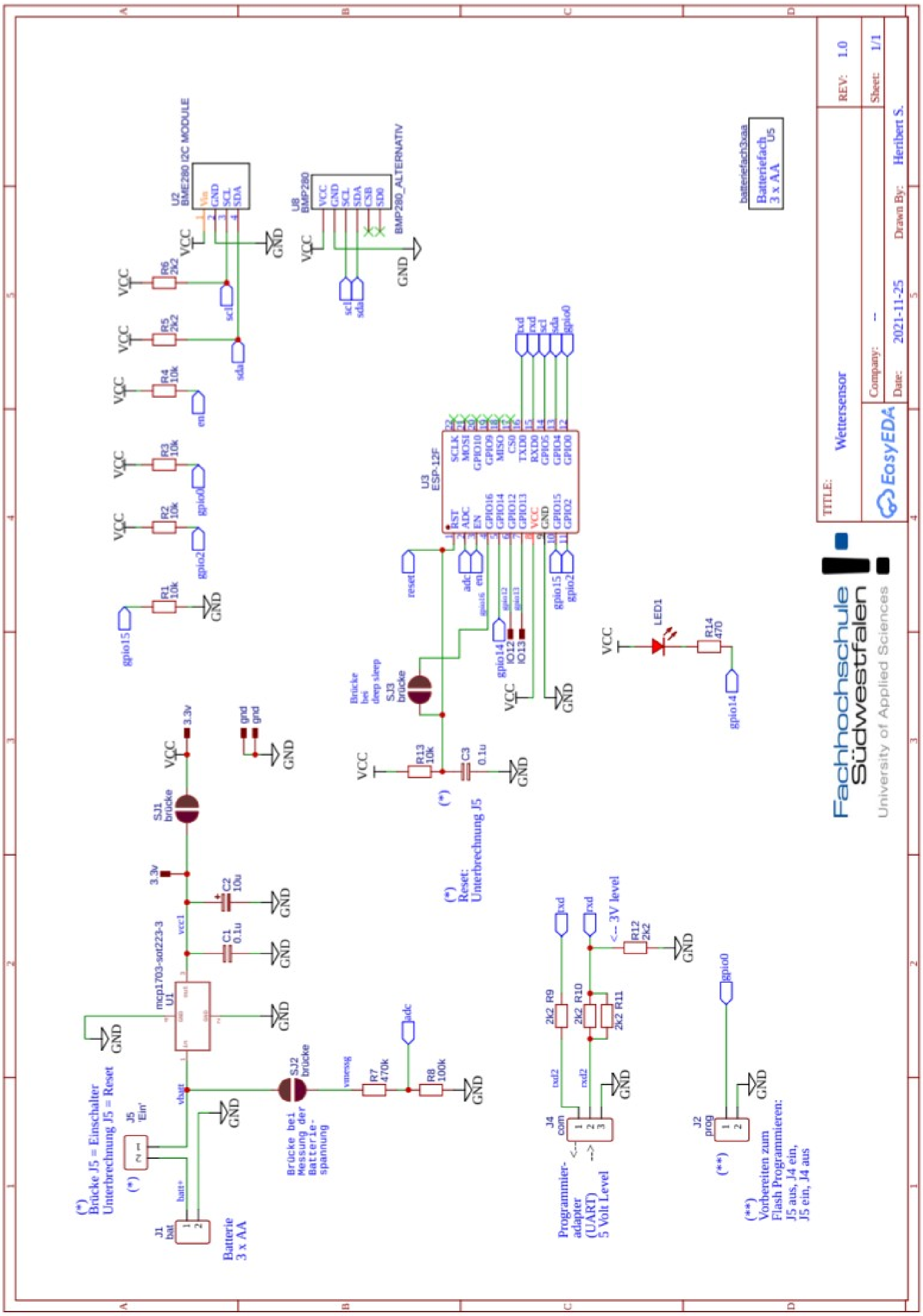






Für das Smartfone gibt es eine passende APP „ Thingview“ von Thingspeak.

Schaltplan


TITLE: Wettersensor
REV: 1.0
Sheet: 1/1
Company: -
Date: 2021-11-25
Drawn By: Heribert S.
EasyEDA
Fachhochschule Südwestfalen
University of Applied Sciences