

h_da
HOCHSCHULE DARMSTADT
UNIVERSITY OF APPLIED SCIENCES
fbmd
FACHBEREICH MEDIA

Projektdokumentation

Der Entwurf und die Umsetzung einer Wetterstation
mit Datenbankanbindung
Stand: 01.09.2019

Dozent: Elke Lang
Kurs: Hardware-Erweiterung von Informationssystemen
Student: Alexander Ceynowa
Mat.Nr: 729668

Inhaltsverzeichnis

1 Vorbereitungen.....	3
1.1 Überlegungen.....	3
1.2 Komponenten.....	4
1.3 Schaltung.....	4
1.4 Die Gehäuse und die Stromversorgung.....	5
1.5 Aktivitätsdiagramme.....	7
1.6 Datenbank-Modell.....	8
1.7 Wireframe.....	9
2 Umsetzung.....	10
2.1 Elektronik.....	10
2.1.1 Die Schaltung.....	10
2.1.2 Die Stromversorgung.....	11
2.1.3 Das Anemometer.....	11
2.2 Die Konstruktion.....	12
2.2.1 Das Anemometer.....	12
2.2.2 Die Stromversorgung.....	12
2.2.3 Die Wetterstation.....	12
2.3 Quellcode.....	13
2.3.1 Arduino / ESP-32-WROOM.....	13
2.3.2 Webserver.....	14
2.3.3 Serversetup.....	14
3 Die Wetterstation im Betrieb.....	15
3.1 Lessons learned.....	15
Anhang.....	16
Installation ESP32 in der Arduino-IDE.....	16
Zeitraffer der Drucke.....	16
Fotos aus dem Verlauf des Projekts.....	17

1 Vorbereitungen

1.1 Überlegungen

Kurz nachdem ich mir einen 3D-Drucker zugelegt hatte, surfte ich viel auf der Plattform Thingiverse¹ und fand dort ein Modell plus Anleitung um eine Wetterstation zu bauen. Dies setzte ich mit leichten Änderungen um und hatte besagte Station für knapp zwei Jahre im Betrieb. In den letzten drei Monaten des Betriebs fasste ich den Entschluss eine überarbeitete Wetterstation (V2) zu entwickeln, hatte aber noch keine konkreten Pläne oder zeitliche Vorstellungen. Als dann in der ersten Veranstaltung von Hardware Erweiterung die Umsetzung von eigenen Projekten angesprochen wurde, war ich mir ziemlich sicher dass die Wetterstation gut passen würde. Das ursprüngliche Modell verwendete (begrenzt durch mein Wissen und meine Mittel) nur drei Sensoren – einen BMP180 für barometrischen Druck, einen DHT22 für exakte und gefühlte Temperatur sowie Luftfeuchtigkeit und eine Solarzelle für relative Helligkeit (hierzu später mehr), aber hier wollte ich nachbessern. Wetterstation V2 sollte also zusätzlich noch ein Anemometer und einen Regensor bekommen. Die Grundstruktur für Code und Komponenten war durch V1 bereits vorhanden, deswegen bewertete ich den Arbeitsaufwand als „für die Veranstaltung angemessen“.

Vom Grundverhalten her soll das ESP sich nach dem „Einschalten“ mit dem Wifi verbinden, Einstellungen abrufen und danach in einem Loop zyklisch Messungen durchführen, diese grundlegend aufbereiten und zum Speichern an einen Server senden. Serverseitig sollen die Daten in einer Datenbank gespeichert und grafisch auf einer Website dargestellt werden.

Wie das Anemometer funktionieren sollte war mir ursprünglich noch ein Rätsel. Die erste Idee war eine Nabe an der Tassen oder Paddel befestigt wären, die mittels einer Welle eine runde Platte mit Aussparungen antreibt. Die Aussparungen könnten dann beispielsweise mit einem optischen Sensor oder einem Mikroschalter mit Blattfeder erfasst werden. Dies setzte ungefettete, low-drag Kugellager voraus, die ich jedoch nicht zur Hand hatte. Reguläre 608-Kugellager waren zu schwerfällig und bremsten das Anemometer zu sehr aus. Vorausgesetzt ich hätte low-drag Lager vorrätig gehabt, wäre auch ein Hall-Sensor² und ein in der Welle eingebetteter Magnet eine Option gewesen, um das Drehen des Anemometer so wenig wie möglich zu beeinträchtigen. Nach der Überlegung über Hall-Sensor und low-drag Kugellager war für mich der Sprung zum umfunktionierten PC-Lüfter nicht mehr weit. Glücklicherweise stellte sich dieser Ansatz als funktional und praktikabel heraus.

1 WiFi Weather Station by OpenGreenEnergy <https://www.thingiverse.com/thing:2282869> Stand: 01.09.2019

2 Hall-Sensor via Wikipedia <https://de.wikipedia.org/wiki/Hall-Sensor> Stand:01.09.2019

1.2 Komponenten

- ESP-32-WROOM
- BMP180
- DHT22
- FC-37 / YL-83
- generische Solarzelle 35 x 35 mm (maximal 5 Volt)
- 80mm PWM PC-Lüfter
- 10k Ω Widerstand x2
- Dupont-Sockel und Dupont-Pinleisten
- Schraubterminal mit 2 Kontakten
- JST-XH Buchse und Stecker mit 2 Kontakten x2
- JST-XH Buchse und Stecker mit 4 Kontakten
- dediziertes 5V 3A Schaltnetzteil
- Feinsicherung (230V 3A) und Feinsicherungshalterung
- krimpbare Verbinder in diversen Größen
- Nylon Abstandshalter für PCBs x4
- Neodym-Magnete 6mm Durchmesser, 2mm Höhe, Rund x4
- M3-Schraube 4mm Länge x4

1.3 Schaltung

Der von mir gewählte DHT22-Sensor wird an einen regulären Digitalpin, in unserem Fall GPIO34, angeschlossen, mit einem 10k Ω Widerstand gegen +5V verbunden und sendet auf Anfrage 40 Bit an Daten: 8 Bit integer Luftfeuchtigkeit, 8 Bit dezimal Luftfeuchtigkeit, 8 Bit integer Temperatur, 8 Bit dezimal Temperatur und 8 Bit Prüfsumme.

Der BMP180-Sensor kommuniziert über einen i²c-Bus, dafür wird der SCL-Pin mit GPIO22, der SDA-Pin mit GPIO21 verbunden. Während der Initialisierung des Sensors werden 8 Bit übertragen, 7 Adress-Bits und ein R/W-Bit. Erkennt das Slave-Gerät (in diesem Fall der BMP), dass es adressiert wird, so sollte SDA im neunten SCL-Zyklus gegen Ground gelegt werden, dies wird die Start-Kondition genannt. Nach der Start-Kondition sendet der Master (das ESP32) eine Schreib-direktive mit Register-Adresse an das BMP, das eine Messung vornimmt und die erfassten Daten in einen lokalen, auf dem Modul vorhandenen EEPROM schreibt. Wenn dies erfolgt ist, wird vom Master ein Restart-Befehl und eine Lesedirektive gesendet, die die zuvor beschriebenen Register aus dem EEPROM ausliest und zurück an den Master sendet. Gewöhnlich bekommen wir hier zwei 16-Bit Werte für Temperatur und Luftdruck, können aber bei Bedarf einen High-Resolution-Modus konfigurieren, durch den wir anstatt 16 Bit 19 Bit für den Luftdruck bekommen.

Die Solarzelle wird in dieser Anwendung eher wie ein Fotowiderstand verwendet, der positive Pol der Zelle wird mit 5 Volt verbunden, der negative Pol mit GPIO33 wegen seiner internen Verbindung zu ADC1, Kanal 5.

Der Regensensor besteht aus zwei Komponenten, dem Regen-“Schild“ und der Konverter-Einheit.

Das „Schild“ wird über einen Verbinder mit dem Konverter verbunden, der Konverter mit +5V, Ground und der analoge Ausgang mit GPIO32. Zusätzlich hat der Konverter noch einen über Potentiometer konfigurierbaren digitalen Ausgang, welcher in dieser Anwendung jedoch nicht genutzt wird.

Das Anemometer ist grundlegend nur ein PC-Lüfter mit PWM-Funktion. Die 4 Leitungen sind Ground (Schwarz), +12V (Gelb), Sense (Grün) und PWM (Blau). Angeschlossen wird das Anemometer wie folgt: Ground zu Ground, +12V zu +5V, Sense an GPIO35 oder einen anderen Interrupt-fähigen Pin und PWM ebenfalls zu Ground. Die Diskrepanz zwischen 5 und 12 Volt spielt uns hier in die Hände, 5 Volt sind zu niedrig um den Lüfter (selbst ohne PWM-Funktion) zum drehen zu bringen aber genug um den, im Lüfter verbauten, Hall-Sensor zu versorgen. Zusätzlich benötigt der ehemalige Lüfter noch einen $10\text{k}\Omega$ Widerstand als Verbindung zwischen Sense und +5V.

1.4 Die Gehäuse und die Stromversorgung



Abbildung 1

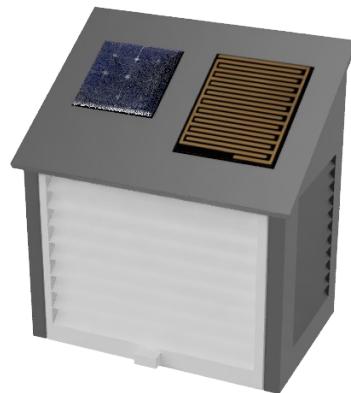


Abbildung 2

Die Grundlage für die Gestaltung von Wetterstation V2 war das Design von V1³ welches ich aus dem Internet heruntergeladen hatte. Meine Bedingungen an den neuen Entwurf waren die Integration von Solarzelle und Regensensor in das Dach der Station, ein Loch im Boden durch das die Kabel geführt werden können, ein Verschlussmechanismus für die Klappe und eine möglichst kompakte Form.



Abbildung 3

Für das Anemometer hatte ich keine klaren Vorstellungen, da dessen Design vom Mechanismus abhing. Der erste Plan lautete einen funktionalen Prototyp zu bauen, diesen nach Möglichkeit zu komprimieren und dann ein passendes Gehäuse zu entwerfen. Nachdem ich mich für einen

³ WiFi Weather Station by OpenGreenEnergy <https://www.thingiverse.com/thing:2282869> Stand 01.09.2019

modifizierten PC-Lüfter entschieden hatte, kannte ich in etwa den benötigten Raum für die Elektronik. Die Form für die ich mich am Schluss entschied, war das was mir in den Sinn kam, wenn ich, mal abgesehen von Windsäcken, an „Windmesser“ dachte. Zunächst entwarf ich die Basis und die Nabe des Anemometers, hierbei wurde die Größe der Tassen nicht berechnet oder aus einer Tabelle entnommen, lediglich als „von mir ästhetisch ansprechend“ empfunden. Die Tassen sollten während der Testphase leicht austauschbar aber ebenso einfach permanent montierbar sein.



Abbildung 5



Abbildung 4

Zuletzt die Stromversorgung – ein wichtiger Punkt bei der Wetterstation war der Wunsch die Einheit möglichst frei bewegen zu können, beispielsweise zur Montage auf dem Dach oder einem anderen, ebenfalls platzbegrenzten Ort. Die Station an sich kommuniziert per Wifi, also ist die einzige Kabelbindung die Stromversorgung. Natürlich hätte sich auch eine Versorgung per Akku, mittels der Solarzelle oder einer Kombination aus beidem realisieren lassen, aber ich entschied mich dagegen, sollte die Station einen Neustart brauchen wäre es einfacher einen Stecker zu ziehen als auf das Dach zu klettern und den Reset-Knopf in der Station zu drücken. Ein weiterer Wunsch war der Einbau einer Sicherung für den Fall das Wasser in die Station eindringen sollte. Der letzte Punkt war die Art der Stromversorgung – USB-Adapter sind zwar inzwischen der de facto Standard in der Welt der Microcontroller, aber wenig für den dauerhaften Betrieb geeignet, speziell wenn man darüber nachdenkt, dass die Kabellänge zwischen Station und Versorgung möglichst nicht begrenzt sein sollte. Das neue Gehäuse musste also groß genug für das Netzteil und die Sicherung sein, trotzdem möglichst kompakt um eventuellen räumlichen Begrenzungen auszuweichen und gut belüftet um Überhitzung des Netzteils zu vermeiden.

1.5 Aktivitätsdiagramme

Abbildung 6 stellt das Vorgehen der Wetterstation dar – nach dem Einschalten verbindet sich das ESP mit dem hinterlegten WLAN, ruft die aktuelle Konfiguration vom Server ab, liest die Daten der Sensorik aus, verarbeitet diese, sendet sie zum speichern an den Server und geht für eine konfigurierte Zeitspanne in einen Sleepmodus.

Abbildung 7 zeigt den Darstellungsprozess auf der Website. Die Daten des aktuellen Tages werden aus der Datenbank abgerufen und für die Darstellung formatiert

Alle Grafiken der Dokumentation sind auch im Ordner der Dokumentation noch einmal zu finden.

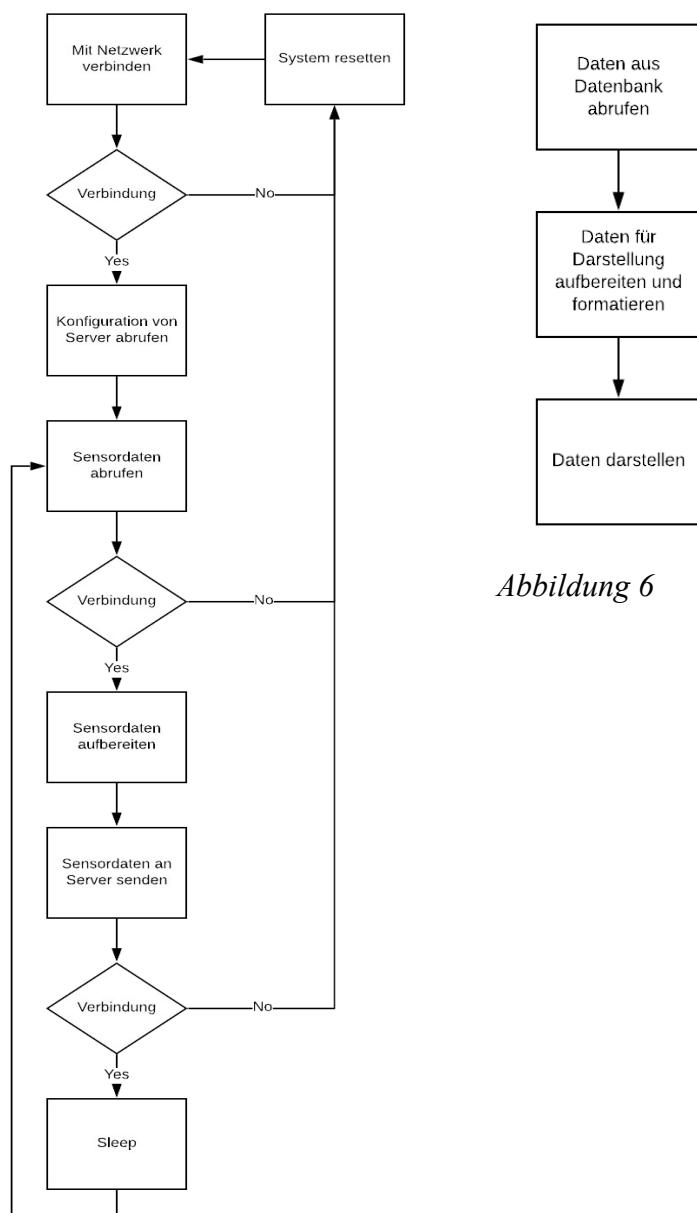


Abbildung 6

Abbildung 7

1.6 Datenbank-Modell

Abbildung 8 repräsentiert das Schema der Tabelle in der die Daten der Wetterstation gespeichert werden, die Tabelle darunter erklärt was für ein Wert in der entsprechenden Spalte steht.

weatherdata
+dateTime: timestamp
+anemoRPM: int(8)
+humidity: decimal(10,2)
+temperature: decimal(10,2)
+felttemp: decimal(10,2)
+rawRain: int(8)
+stageRain: int(8)
+rawSolar: int(8)
+solar: int(8)
+pressure: decimal(10,2)

Abbildung 8

Titel	Zweck
dateTime	Zeitpunkt der Speicherung des Datensatzes, wird automatisch vom Server eingetragen.
anemoRPM	Stärke des Windes, gespeichert in RPM (revolutions per minute / Umdrehungen pro Minute).
humidity	Relative Luftfeuchtigkeit in Prozent (ohne Einheitsindikator)
temperature	Temperatur in Grad Celsius (ohne Einheitsindikator)
felttemp	Die gefühlte Temperatur in Grad Celsius (ohne Einheitsindikator)
rawRain	Der analoge Regenwert zwischen 0 und 4095
stageRain	Der Regenwert in Prozent (ohne Einheitsindikator)
rawSolar	Der analoge Sonnenwert zwischen 0 und 4095
solar	Der Sonnenwert in Prozent (ohne Einheitsindikator)
pressure	Der Luftdruck in hPa (ohne Einheitsindikator)

1.7 Wireframe

Da der Teil der Website mit Wetterdaten auch von älteren Familienmitgliedern genutzt wird, bestand für mich die Anforderung das Interface möglichst schlicht und intuitiv zu halten. Die bereits bestehende Website basiert auf Foundation 6, musste also nur um einen Bereich erweitert werden. Wird die Seite geöffnet, sollen sofort die Daten des aktuellen Tages angezeigt werden, jedoch soll auch die Möglichkeit geben seine frühere Daten abzurufen.

Zentral auf der Seite befindet sich ein Tab-Modul von Foundation in dem die erhobenen Daten kategorisiert in Graphen und in einer generellen Übersicht angezeigt werden.

Oberhalb der Tabs befindet sich ein Paginator-Element, mit dem die vorigen Daten durchgegangen werden können und das aktuell gewählte Datum mittig angezeigt wird.

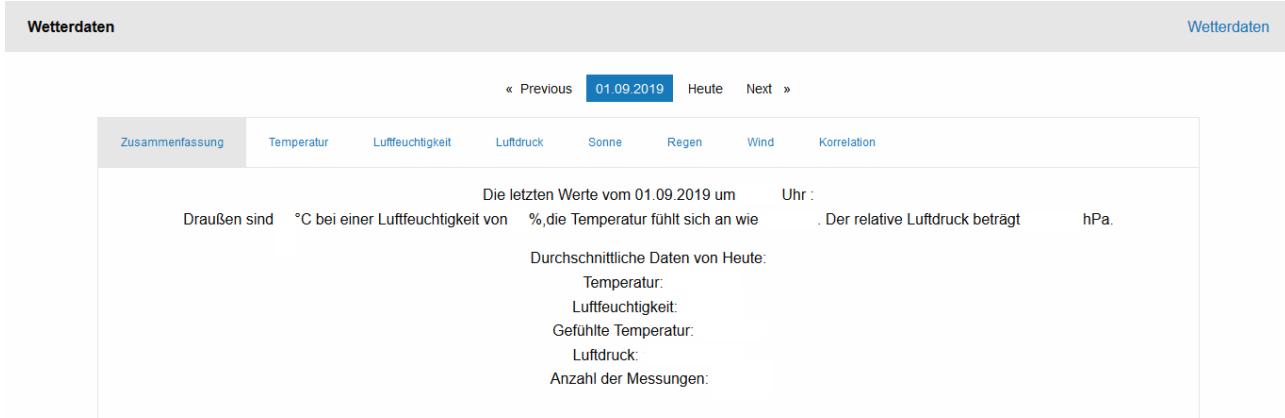


Abbildung 9

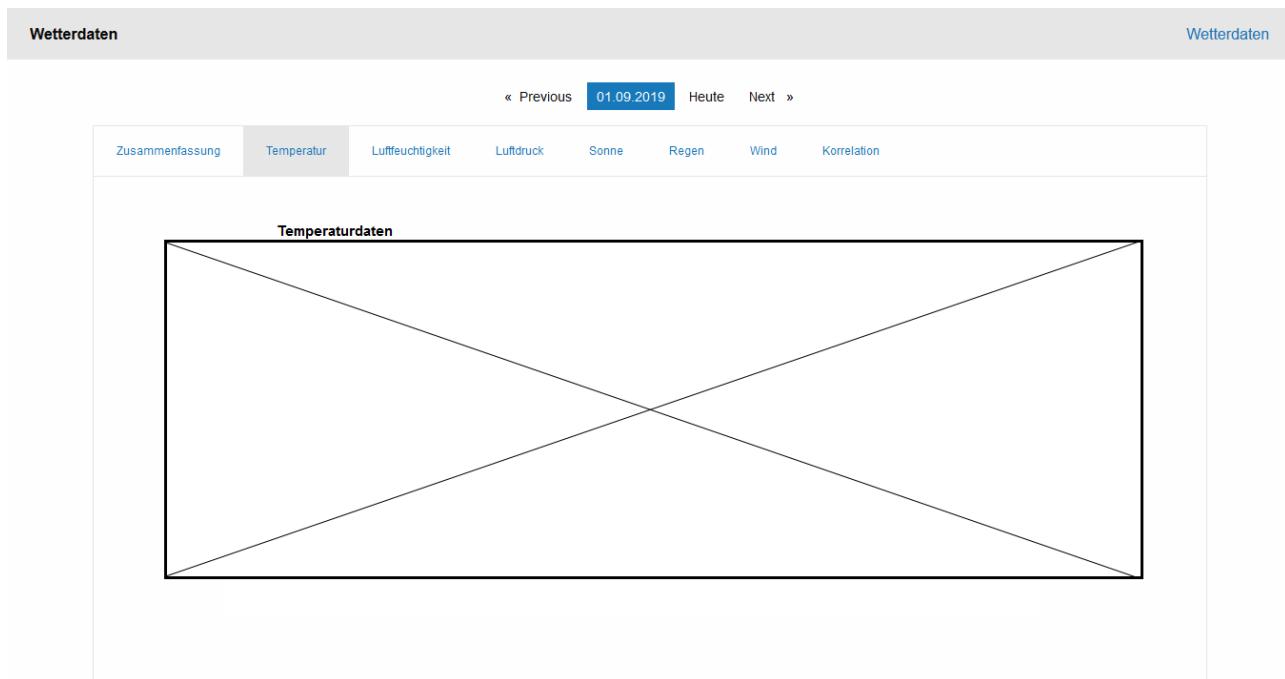


Abbildung 10

2 Umsetzung

2.1 Elektronik

Der erste Schritt der Umsetzung des Projektes war für mich die Elektronik zum laufen zu bekommen. Sobald der Prototyp funktionierte, konnte über komprimieren und konsolidieren der Elektronik nachgedacht und diese dann umgesetzt werden. Wäre dies abgeschlossen, könnte über Gehäuse und andere mechanische Bestandteile nachgedacht werden – eine Ausnahme bildete hier das Anemometer, dazu mehr in Abschnitt 2.2.

2.1.1 Die Schaltung

Um die Schaltung der Wetterstation wie bei dem beigelegtem Fritzing-Diagramm umzusetzen, müssen nur fehlende Pins in das BMP-Breakout-Board gelötet werden. Sollte die Schaltung jedoch auf einem Perfboard aufgebaut werden, schlage ich vor, an die Pins des DHT-Sensors vier reguläre Pins einer Pinleiste anzulöten, da die dünnen Beine in Dupont-Sockeln keinen verlässlichen Kontakt garantieren. Das Gehäuse der Wetterstation ist auf ein 9x5cm Perfboard ausgelegt. Im Bezug auf die Perfboard-Lösung empfehle ich generell den Einsatz von Dupont-Buchsen, um das Austauschen defekter Komponenten so einfach wie möglich zu gestalten – sollte die Wetterstation an einem schwer

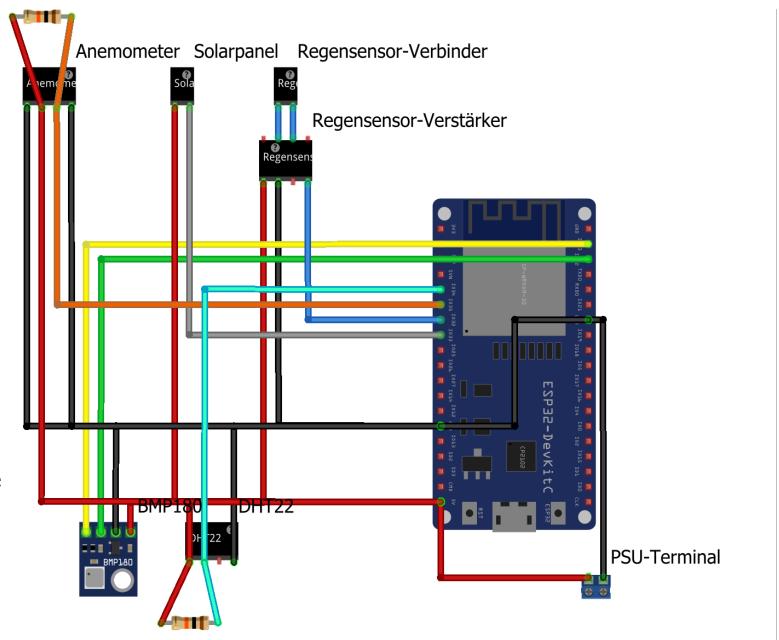


Abbildung 11

zugänglichen oder räumlich sehr eingeschränktem Ort aufgestellt sein, bietet sich eventuell nicht die Möglichkeit oder Motivation das Board zu entfernen, die Komponente zu entlöten, eine neue Komponente einzulöten und so weiter. Um das Austauschen der Teile zu vereinfachen wurden an die Solarzelle und den Regensorse JST-XH Stecker mit zwei Pins verbunden, die Pins der Verstärker-Einheit des Regensorse wurden von der Oberseite auf die Unterseite umgelötet. Um Verwirrung und Fehlersuche vorzubeugen: Der Pin neben +5v auf dem DevKitC war bei meinen beiden Modulen fälschlicherweise als GND beschriftet, dieser Pin hat eigentlich die Bezeichnung CMD – bei meinem ursprünglichen Board war mir das noch nicht aufgefallen und die Fehlersuche war durchaus nervenaufreibend. Ebenfalls ist zu beachten, dass theoretisch jeder Pin geändert werden kann um ein besseres Layout zu erzielen, jedoch ist hier zu beachten, dass das DevKitC vier Bootstrapping-Pins⁴ als GPIO ausgibt. Diese lassen sich zwar wie reguläre GPIO benutzen, müssen während dem Systemstart jedoch spezielle HIGH/LOW/FLOAT Stadien haben. Auch bei den ADC-Pins gibt es eine spezielle Begebenheit, die von Relevanz ist – Das ESP-32-WROOM hat zwei ADC: ADC1 und ADC2; ADC1 ist immer frei nutzbar, ADC2 funktioniert während Betrieb mit Wifi jedoch nur als digitales GPIO, die analoge Funktionalität ist währenddessen nicht nutzbar, auch dies sorgte während meiner Tests für einige Verwirrung.

4 ESPxx and Bootstrapping <https://github.com/esp8266/Arduino/blob/master/doc/boards.rst#minimal-hardware-setup-for-bootloading-only> Stand: 01.09.2019

2.1.2 Die Stromversorgung

Die Stromversorgung zu realisieren war vermutlich der einfachste Teil des Projekts. Ich habe für meiner Umsetzung einen SchuKo-Stecker verwendet, plane allerdings für die spätere Verwendung eine feste Montage über eine Verteilerbox im Dachstuhl. Die eigentliche Verbindung des Netzteils und der Sicherung ändert sich dadurch jedoch nicht. Die drei Litzen der Stromzufuhr werden getrennt in L (Live - Phase), N (Neutral - Neutralleiter) und PE (Protective/Permanent Earth – Erdung/Schutzleiter). N- und PE-Litzen werden mit Crimp Spaten Drahtverbindern bestückt und direkt mit den Schraubterminals des Netzteils verbunden, L wird mit einem weiblichen, isolierten Crimp Terminal vercrimpt und an den inneren Kontakt einer Feinsicherungshalterung gesteckt, hier darf man nicht die Silikon-Schutzhülle vergessen. Nun wird noch ein weiteres, etwa 5 cm langes Stück Litze benötigt, das an einem Ende ein weiteres weibliches, isoliertes Crimp Terminal und am anderen einen Crimp Spaten Drahtverbinder hat. Dies ist der Verbinder zwischen Feinsicherungshalterung-Außen und Netzteil-L. Vom Aufbau der Schaltung her sitzt die Sicherung also VOR dem Netzteil, was gewährleistet das, wenn das Netzteil Probleme jedweder Art hat, sofort die Sicherung durchbrennt und sichergangen werden kann, dass das Netzteil nicht wieder in Betrieb genommen wird (im Sinne von „jemand schaltet den Schutzschalter im Sicherungskasten wieder ein“), ohne den Fehler zu untersuchen. Hier wäre eventuell auch eine Sicherung auf der 5 Volt Seite interessant aber ich entschied mich vorerst dagegen, da der Verbrauch der Station bedingt durch sleep und boot fluktuiert und ich eventuelle Spikes, die die Sicherung auslösen, aktuell noch nicht ausschließen kann.

2.1.3 Das Anemometer

Um das Anemometer zu bauen wird ein 80mm PWM PC-Lüfter genutzt. Auf der Rückseite der meisten Lüfter befindet sich ein Aufkleber mit Informationen zum Lüfter – dieser wird abgelöst. Darunter befindet sich ein kleiner C-Clip der sich mittels einer Pinzette öffnen lässt. Für gewöhnlich befindet sich darunter noch ein Silikon-O-Ring, welcher ebenfalls mittels Pinzette entfernt werden kann. Beide Teile werden beiseite gelegt. Nun kann der Rotor-Hub abgenommen werden, denn dies erleichtert den nächsten Schritt: Das Abtrennen der Finnen. Die Finnen können mittels kleinem Saitenschneider grob entfernt werden, die übrige Spreu kann mittels Schlüsselfeile oder grobem Sandpapier (80-Grit) geglättet werden. Ein perfektes Finish auf dem Hub ist nicht nötig, es müssen nur eventuelle Unebenheiten entfernt werden, die ein Offsetting der Anemometer-Nabe verursachen könnten. Das Entfernen des Hubs erleichtert außerdem das Entfernen der Kabel am PCB der Lüfters. Um das PCB zu entfernen kann in der Regel einfach das Spulen-Ensemble von der Plastik-Halterung gezogen werden. Ist das Ensemble mit dem Plastik verklebt, wird von Lösungsmitteln abgeraten, da sie die Kabelummantelung angreifen oder das Plastik des Lüfters verformen können. Auch wenn der reguläre Betrieb des Lüfters unerwünscht ist, sind die intakten Windungen nötig, damit der Hall-Sensor die Pol-Wechsel feststellen kann. Die Kabel werden entlötet, entfernt und durch längere Kabel mit JST-XH 4-Pin Stecker ersetzt. Der von mir genutzte Lüfter war direkt auf einen Kühlkörper geschraubt und hatte dementsprechend keinen Rahmen, sollte ein Lüfter mit Rahmen vorliegen, können die Speichen aber gekürzt und dadurch der Rahmen entfernt werden. In diesem Fall sollten in zwei der Speichen Löcher für M2 Schrauben gebohrt werden um den Lüfter mit der Basis des Anemometers zu verbinden. Wenn der Hub gestutzt und die Kabel verbunden sind, kann der ehemalige Lüfter wieder zusammengesetzt werden.

2.2 Die Konstruktion

Die Konstruktion des Gehäuses setzt nicht zwingend den Besitz eines 3D-Druckers oder einer CNC-Fräse voraus, lediglich die Konstruktion des Anemometer könnte ohne diese Maschinen kompliziert werden.

Ersatzweise können beispielsweise Plastikdosen mit Dichtungen in den Deckeln verwendet werden, in die mehrere kleine Löcher zwecks Durchlüftung und Kabelführung gebohrt werden.

Ist jedoch entsprechende Maschinerie vorhanden, befindet sich im Dokumentations-Ordner die Struktur „3DFiles“ in dem die Gehäuse im STEP- und SKP-Dateiformat vorliegen.

Für den 3D-Druck sollte eine 0.4mm Nozzle genutzt werden. Temperaturen und Geschwindigkeit hängen vom gewählten Filament ab, Layer-Höhe sollte 0.2mm sein um die nötige Layer-Adhesion zu gewährleisten und kein Wasser durchzulassen. Für diese Anwendung wurde PLA (PolyLacticAcid / Polymilchsäure) gewählt, aber ABS, PET oder Nylon sind ebenfalls möglich. Ausnahme hier ist das „Fitting“, das mit TPE-U, einem flexiblen Filament, gedruckt wurde.

2.2.1 Das Anemometer

Der Lüfter kann in die Löcher in der Basisplatte des Anemometers geschraubt, die Kabel durch das große Loch nach außen geführt werden. Die Stiele der Tassen werden in der entsprechenden Orientierung mittels Sekundenkleber in die Aussparungen der Nabe des Anemometers geklebt. Um die Nabe und den Hub des Lüfters zu verbinden werden vier kleinere Kugeln „BlueTac“, in Deutschland auch Posterknete genannt, in den tiefsten Teil der Vertiefung auf der Unterseite der Nabe gedrückt. Nun wird mit etwas Kraft die Nabe auf den Hub gedrückt. Ich entschied mich hier gegen regulären Kleber, da zum Austausch des Lüfters oder zur Installation des Anemometers mittels des zentralen Lochs in der Basis die Nabe ohne Schäden abgenommen werden muss.

2.2.2 Die Stromversorgung

Um das Netzteil im Gehäuse zu verbauen wird zuerst der Feinsicherungshalter entfernt. Das Netzteil hat je nach Hersteller auf der Unterseite ein Loch mit Gewinde, welches zur Befestigung dient. Dieses Loch wird auf das Gehäuse übertragen und ausgebohrt oder -geschmolzen, danach wird das Netzteil festgeschraubt. Vom Feinsicherungshalter wird die Installationsmutter entfernt und von außen durch das Loch in der Seite des Gehäuses gesteckt. Die Mutter kann nun wieder festgeschraubt und die Verbinder wieder aufgesteckt werden. Zuletzt wird der Deckel aufgelegt und mittels Gummiband oder Kabelbinder mit dem Gehäuse verbunden.

2.2.3 Die Wetterstation

Die Wetterstation besteht aus drei Teilen: Dem Gehäuse, der Klappe und dem „Fitting“. Alle drei Teile wurden gedruckt. Auf der Oberseite der Klappe und den entsprechenden Stellen oberhalb der Schienen im Gehäuse sind vier Aussparungen in die die vier Neodym-Magnete als Schließmechanismus eingeklebt werden – hier ist auf die Polung der Magnete zu achten, da sonst die Klappe nicht schließt. Sind keine Magnete vorhanden, so kann eine aufgebogene Büroklammer erhitzt und einen bis zwei Millimeter von der Unterkante der Klappe durch die Front in das mit der Klappe geschlossene Gehäuse geschoben und somit ein Loch für einen Sicherungsstift kreiert werden, das die Klappe sicher mit dem Gehäuse verbindet. Sind Magnete oder Sicherungsstift montiert, kann das Fitting, das von oben wie ein Ring aussieht, so aufgeschnitten werden, dass es sich öffnen lässt und von oben anstatt wie ein „O“, wie ein „U“ aussieht. In die Löcher in der Innenseite des Gehäuses werden vier Nylon-Abstandshalter geschraubt, das überschüssige Gewinde das auf der Rückseite raußschaut wird abgetrennt und mit jeweils einem kleinen Tropfen

Sekundenkleber versiegelt. Die Kabel von Regensor und Solarzelle werden durch die entsprechenden Öffnungen im Dach in den Innenraum geführt und in ihre Vertiefungen gedrückt. Um die Komponenten an Ort und Stelle zu halten wird eine kleine Menge Heißkleber genutzt, der in die Fugen rund um die Module eingebracht wird. Stromkabel und die Verbindung zum Anemometer werden nun in das Fitting gesteckt, die Enden der Verbindungen durch die Öffnung im Boden der Station ins Innere geführt und das Fitting so in die Öffnung gedrückt, dass sowohl innen als auch außen ein Rahmen vom Fitting zu sehen ist. Das fertige Perfboard kann nun mittels 4 M3-Schauben auf den Standoffs befestigt und die Stecker mit den Buchsen verbunden werden. Als letztes wird das Stromkabel der Polung entsprechend in den Schraubterminals befestigt. Somit ist die Wetterstation fast einsatzbereit.

2.3 Quellcode

2.3.1 Arduino / ESP-32-WROOM

Der Quellcode für das ESP nutzt sieben Libraries. Die ersten drei sind entweder in der Arduino IDE enthalten oder werden bei der Installation des ESP32-Frameworks installiert.

WiFi.h: Stellt die Netzwerkfunktionalität zur Verfügung.

HTTPClient.h: Macht HTTP-Kommunikation möglich.

Wire.h: Ist für das i²c-Protokoll zuständig.

Die nächsten vier Libraries werden über den Library Manager installiert oder aus der Dokumentation (Underordner „lib“) kopiert.

Adafruit_Sensor.h: Stellt Basisklassen für andere Adafruit-Libraries zur Verfügung.⁵

DHT_U.h: Macht einfache Kommunikation und Datenverarbeitung mit dem DHT22-Sensor möglich.⁶

SFE_BMP180.h: Macht einfache Kommunikation und Datenverarbeitung mit dem BMP180-Sensor möglich.⁷

ArduinoJson.h: Stellt JSON-Funktionalität bereit, ermöglicht den einfachen Einsatz von Arrays und kann direkt Daten aus Requests deserialisieren oder als Request serialisieren.⁸

Im beiliegendem Sketch (src/main.cpp) ist außerdem die **Arduino.h** enthalten, diese Referenz sollte entfernt werden, wenn der Sketch in der Arduino IDE geöffnet wird.

Im Quellcode müssen Wifi-Daten, Serveradresse und die Höhe der Wetterstation über Normal Null⁹ angegeben werden, danach kann die Firmware kompiliert und geflasht werden.

Das ESP32 beziehungsweise das DevKitC, das von mir verwendete Board, hat einen eingebauten 10-Bit ADC, also werden analoge Werte nicht von Null bis 1023, sondern von Null bis 4095 ausgegeben. Sollte hier ein anderer Microcontroller verwendet werden, müssen die entsprechenden map(); Befehle angepasst werden.

⁵ Adafruit Unified Sensor Library: https://github.com/adafruit/Adafruit_Sensor Stand: 01.09.2019

⁶ Adafruit DHT Sensor Library: <https://github.com/adafruit/DHT-sensor-library> Stand: 01.09.2019

⁷ Sparkfun BMP180 Library: https://github.com/sparkfun/BMP180_Breakout_Arduino_Library Stand: 01.09.2019

⁸ Benoît Blanchon – ArduinoJson Library: <https://github.com/bblanchon/ArduinoJson> Stand: 01.09.2019

⁹ GPS-Koordinate zu Meter ü.NN: <https://www.mapcoordinates.net/de> Stand: 01.09.2019

2.3.2 Webserver

Auf dem Webserver werden vier Dateien benötigt:

config.php: Hier wird die Verbindung zum MySQL-Server konfiguriert.

getSettings.php: Hier ruft die Wetterstation Einstellungen ab.

saveData.php: Diese Datei akzeptiert die Daten die von der Wetterstation gesendet werden und speichert diese in der Datenbank.

index.php: Hier werden die gesammelten Daten ausgegeben. Im Code der Seite befindet sich eine Hauptschleife die die Daten aus der Datenbank in das korrekte Format bringt. Ein Wunschdatum für die Datenanzeige kann über eine Leiste ausgewählt oder direkt per GET-Parameter angegeben werden. Die Anzeige als Graph wurde mittels Google Charts realisiert.

2.3.3 Serversetup

Dies ist eigentlich nur als kleine Anmerkung gedacht. Im Rahmen privater Projekte habe ich bereits bestehende Server und werde deswegen nicht weiter auf einen Setup-Prozess eingehen.

Voraussetzungen hier sind lediglich ein Webserver (Apache2 oder Lighttpd), PHP mit MySQLI-Unterstützung und ein MySQL-Server (MariaDB ist auch in Ordnung, Hauptsache kompatibel) zu dem ein Zugang besteht. Die einfachste Lösung für eine neue Umsetzung hier wäre im günstigeren Ende ein Raspberry-Pi, soll das System aber noch andere rechenintensivere Prozesse durchführen wäre hier vielleicht ein Beaglebone Black¹⁰ oder ein LattePanda¹¹ eine Möglichkeit.

10 BeagleBone Black <https://beagleboard.org/black> Stand: 01.09.2019

11 LattePanda via DFRobot <https://www.dfrobot.com/product-1404.html> Stand: 01.09.2019

3 Die Wetterstation im Betrieb

3.1 Lessons learned

Mein ursprünglicher Motivator ein ESP32-Modul zu nutzen war der Mangel an analogfähigem GPIO, da der Vorgänger, das ESP8266¹², leider nur einen Analog-Pin hat und auch das restliche GPIO eher begrenzt (auch hier das Problem mit den Bootstrapping-Pins¹³) ist. Eine im Nachhinein bessere Möglichkeit wäre vermutlich eher der Einsatz eines EPS8266 in Verbindung mit einem analogfähigem MUX gewesen. Der Grund dieser Überlegung sind spontane Ausfälle des ESP32, in denen die Wetterstation in einen locked-up Zustand fällt, in dem nur ein totaler Cold Reset hilft.

Meine V1 mit dem 8266 hatte keine derartigen Probleme. Der Fehler lässt sich leider nicht gewünscht oft reproduzieren um ihn zur Ursache zu verfolgen, obwohl bei mir inzwischen die Vermutung besteht, dass das ESP32-Modul einen „Brown-Out“ hat – die Stromversorgung auf dem Board ist laut Github-Issues bekannt dafür „[...] nicht die Beste zu sein.“¹⁴ Dieses Verhalten scheint mit dem Regensensor zusammenzuhängen, da Ausfälle am häufigsten bei Regen beobachtet werden konnten und der Regensensor mit dem Barometer konkurriert – heißt: Entweder es gibt Werte vom Barometer oder dem Regensensor, aber nicht beiden gleichzeitig. Hier würde ich bei einer neuen Version versuchen Transistoren einzusetzen und dann im Code bei Bedarf die Komponenten ein- und ausschalten. Dies hätte außerdem einen stromsparenden Effekt, mit dem vielleicht ein System mit Batteriebetrieb machbar wäre, Stichwort LoRa¹⁵.

12 ESP8266 Modul via Espressif <https://www.espressif.com/en/products/hardware/esp8266ex/overview> Stand: 01.09.2019

13 ESPxx and Bootstrapping <https://github.com/esp8266/Arduino/blob/master/doc/boards.rst#minimal-hardware-setup-for-bootloading-only> Stand: 01.09.2019

14 Issue suche im ESP32 Git <https://github.com/espressif/arduino-esp32/issues?utf8=%E2%9C%93&q=brownout> Stand 01.09.2019

15 Long Range Wide Area WAN https://de.wikipedia.org/wiki/Long_Range_Wide_Area_Network Stand: 01.09.2019

Anhang

Installation ESP32 in der Arduino-IDE

Um ein ESP32-Modul über die Arduino IDE zu nutzen sind einige zusätzliche Schritte notwendig. Besagte Schritte finden Sie auf der Website https://github.com/espressif/arduino-esp32/blob/master/docs/arduino-ide/boards_manager.md.

Zeitraffer der Drucke

Zeitrafferaufnahmen vom Druck der verschiedenen Komponenten sind im Unterordner „/doc/Picture/Timelapses/“ zu finden.

Fotos aus dem Verlauf des Projekts

Weitere Fotos sind im Unterordner „/doc/Picture/Heist-Fotos/“ zu finden.

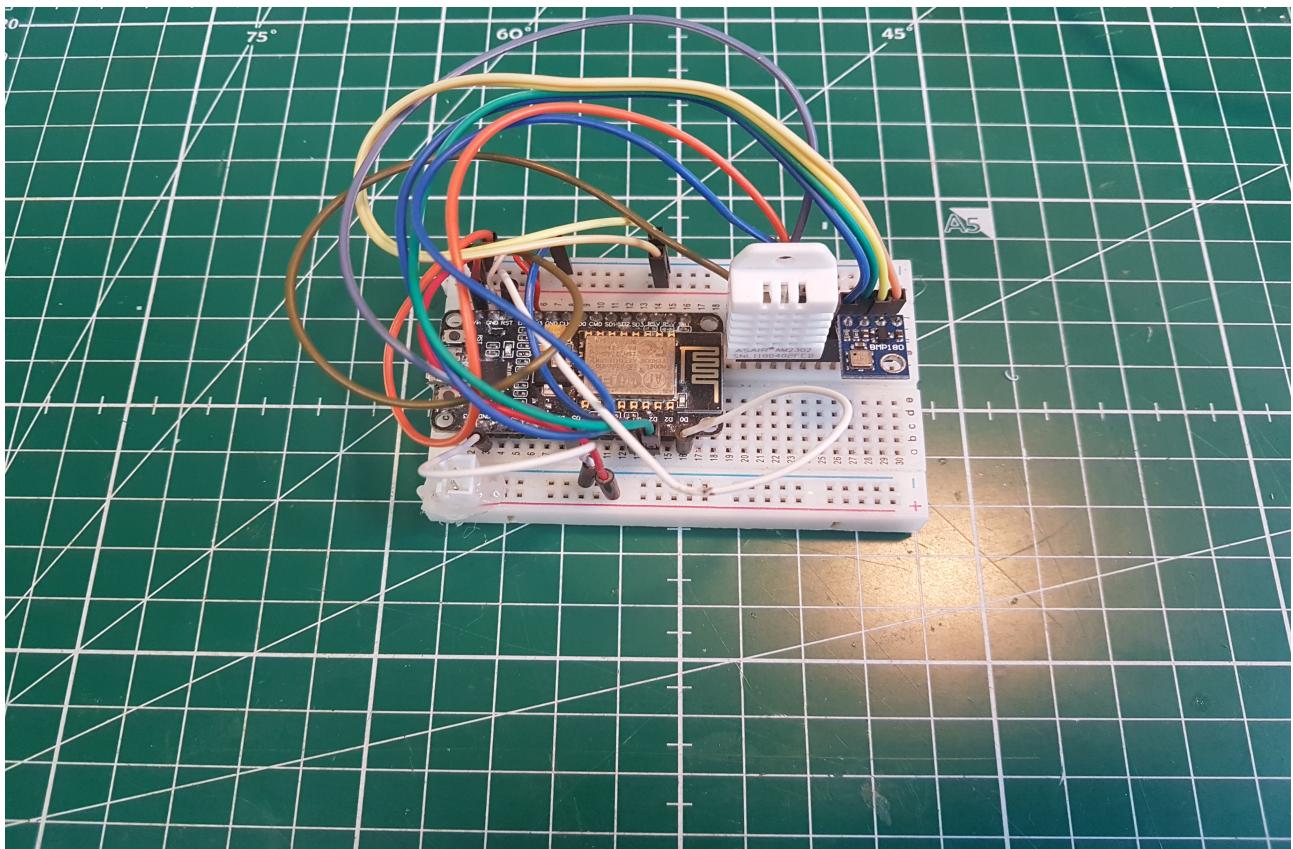


Abbildung 12: Schaltung Wetterstation V1



Abbildung 13: Vergleich Gehäuse Wetterstation V1 vs. V2

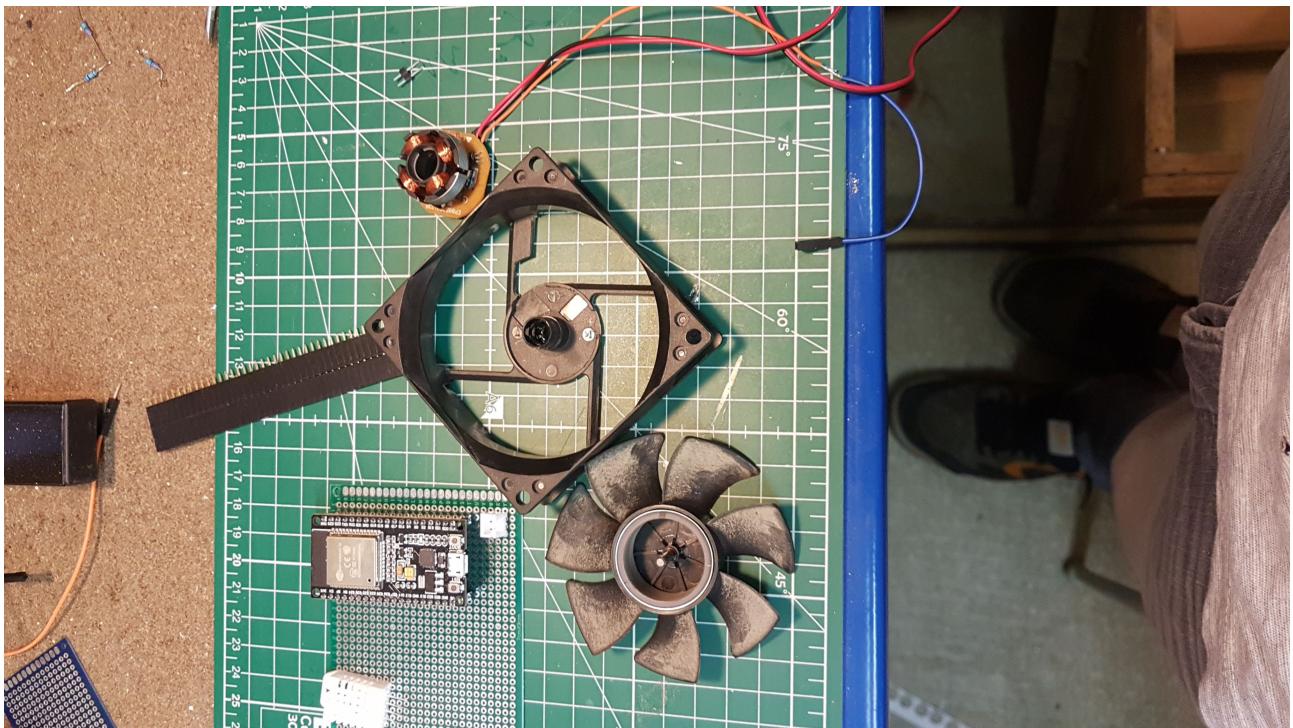


Abbildung 14: Dekonstruierter Lüfter



Abbildung 15: Lüfter ohne Finnen

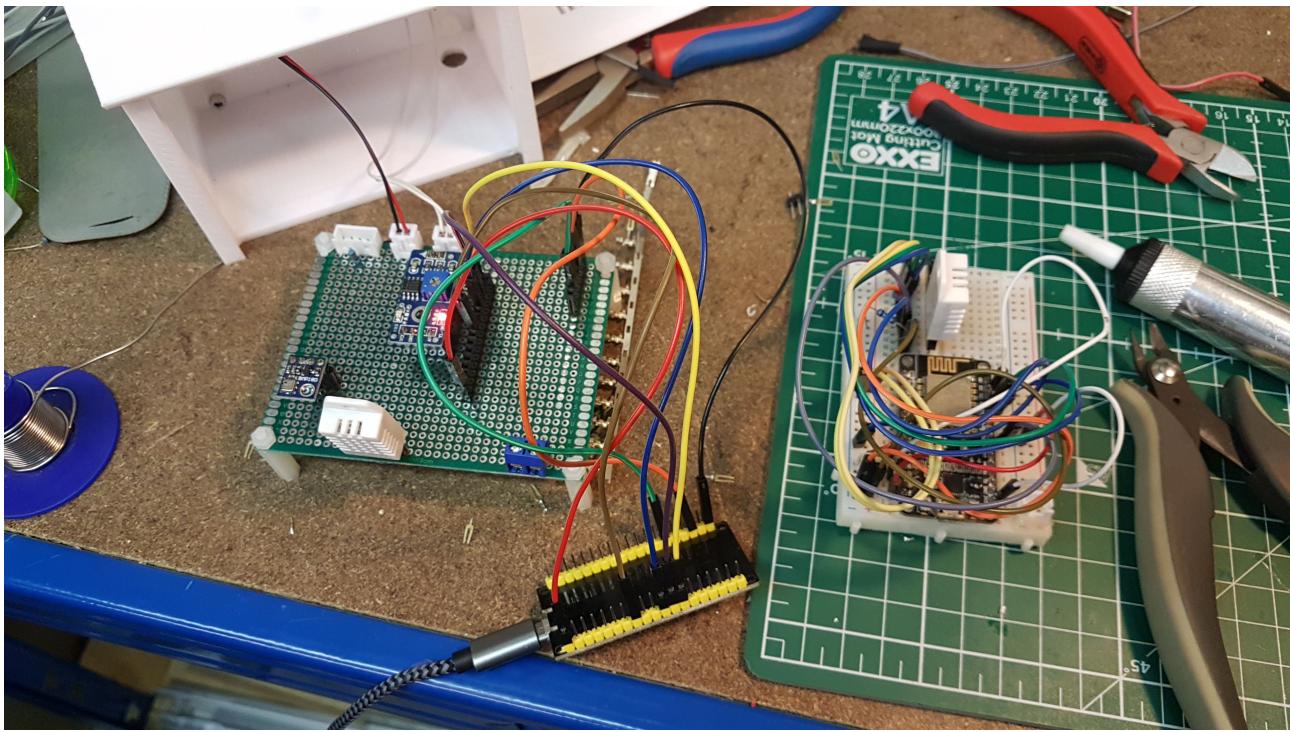


Abbildung 16: Fehlersuche durch falsch beschrifteten Pin

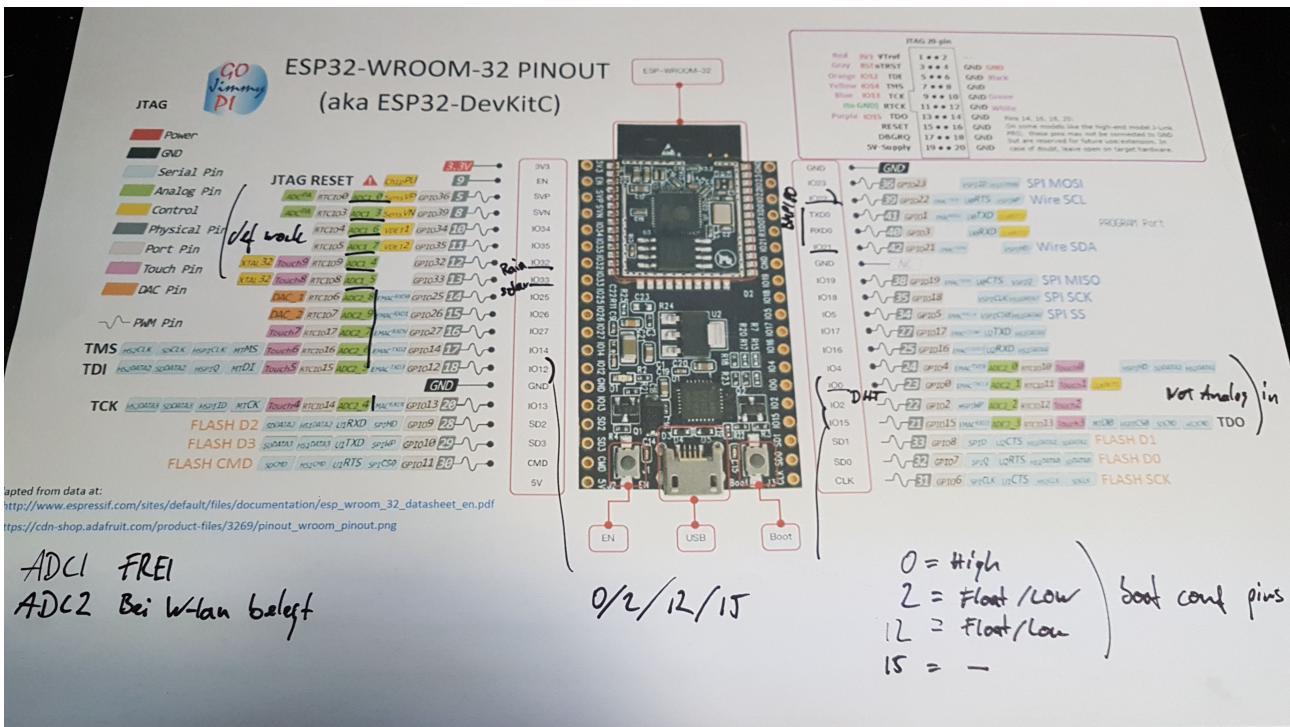


Abbildung 17: Notizen auf Pinout-Diagramm

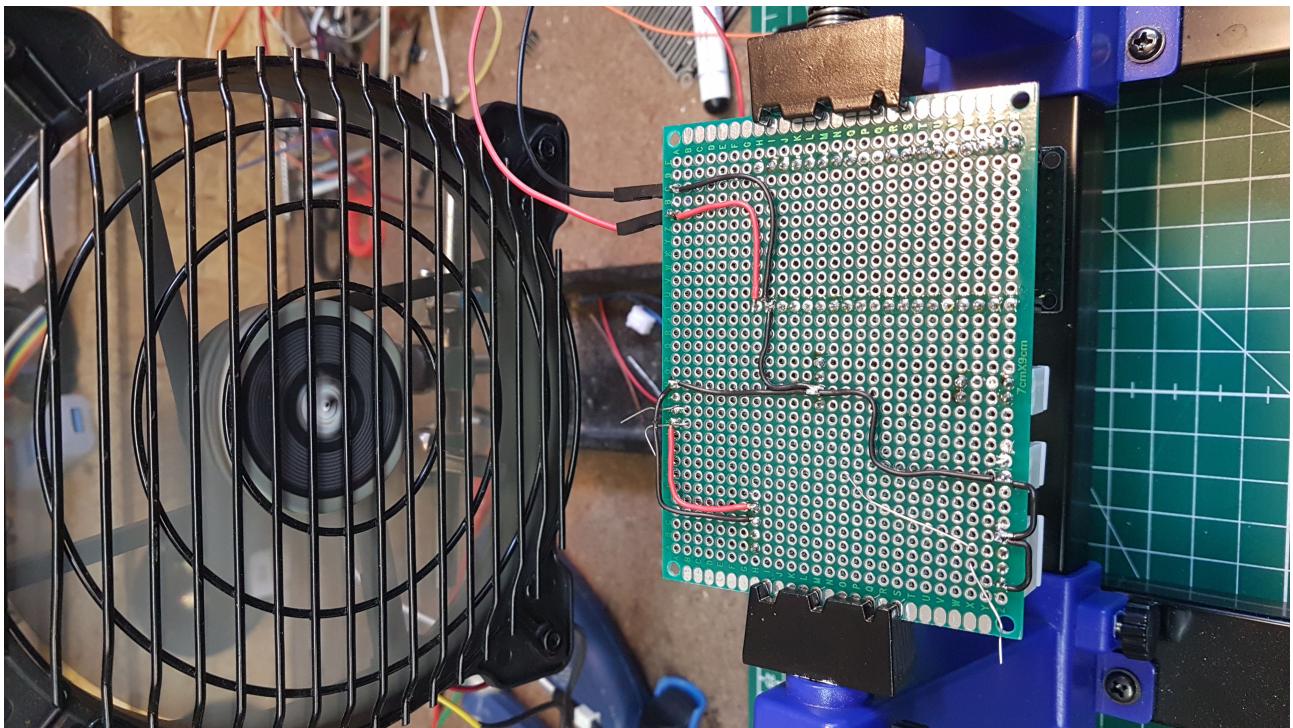


Abbildung 18: Der falsch beschriftete GND-Pin angeschlossen

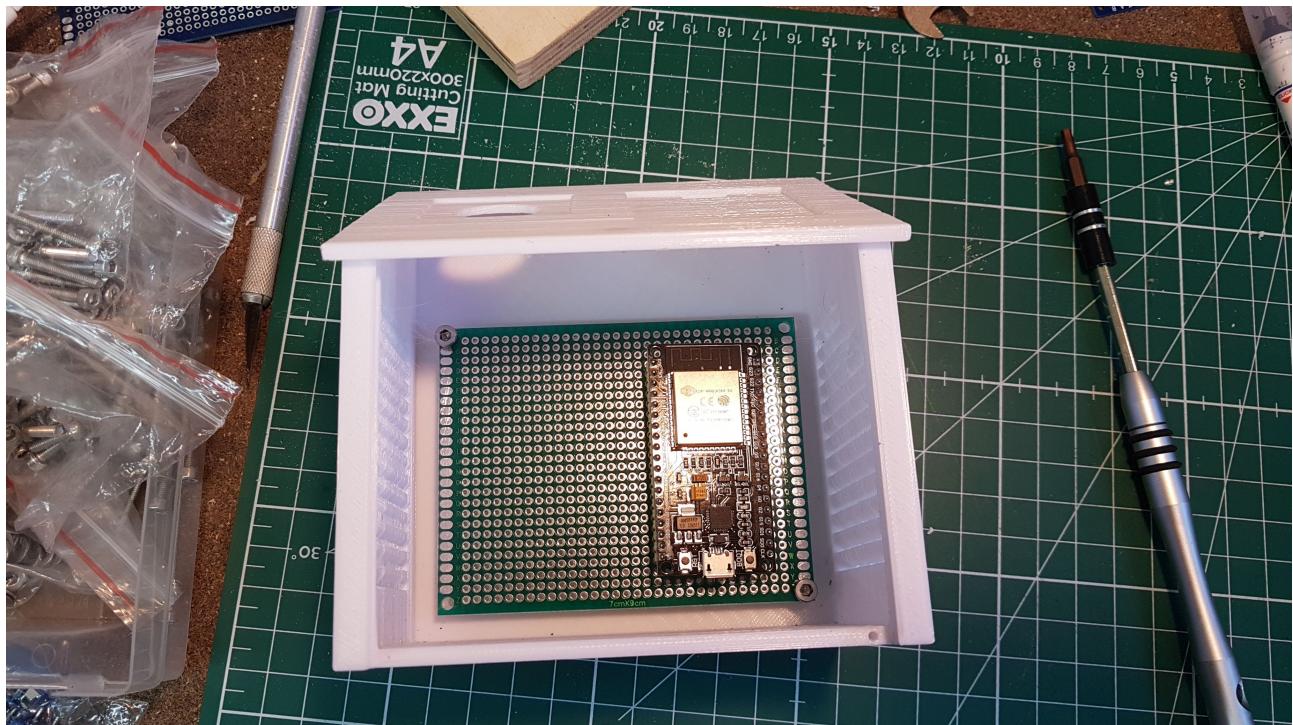


Abbildung 19: PCB testweise im Gehäuse