

# Eine Wetterstation mit dem Raspberry Pi bauen



– Kurzdokumentation –

Alexandar Vesovic, S4  
Helmut-Schmidt-Gymnasium

27.04.2017

# Inhaltsverzeichnis

<b>1 Vorwort</b>	<b>4</b>
1.1 Motivation zum Projekt . . . . .	4
1.2 Was ist ein Raspberry Pi? . . . . .	5
<b>2 Raspberry Pi - Betriebssystem einrichten</b>	<b>8</b>
2.1 Betriebssystem herunterladen, SD-Karte vorbereiten, Raspbian einrichten . . . . .	8
2.2 Grundlegende Einstellungen am Raspberry Pi vornehmen . . . . .	10
2.2.1 Standardpasswort ändern . . . . .	10
2.2.2 <i>Secure Shell</i> eingeschalten . . . . .	10
2.2.3 <i>Hostname</i> festlegen . . . . .	11
2.2.4 Mehr Zwischenspeicher bereitstellen . . . . .	11
2.2.5 Sprachumgebung festlegen . . . . .	11
2.2.6 Übertakten des Prozessors . . . . .	11
<b>3 WeatherPi - Raspberry Pi beschalten</b>	<b>12</b>
3.1 Welche Sensoren werden verwendet? . . . . .	12
3.1.1 Thermometer und Hygrometer: DHT22 . . . . .	12
3.1.2 Thermometer und Barometer: BMP180 . . . . .	13
3.1.3 Anemometer, Anemoskop und Pulviometer . . . . .	14
3.1.4 Vollständiger Schaltplan mit dem Raspberry Pi Zero . . . . .	14
3.2 Wie werden die Messwerte übertragen? . . . . .	17
3.2.1 DHT22: Digitale Signale lesen . . . . .	17
3.2.2 BMP180: $I^2C$ -Schnittstelle benutzen . . . . .	18
3.2.3 Windrichtung des Anemoskops bestimmen . . . . .	19
3.2.4 Magnetschalter als Sensor: Windgeschwindigkeit bestimmen . . . . .	20
3.3 Internet of Things - Messungen über ThingSpeak visualisieren . . . . .	22
3.3.1 Channel einrichten . . . . .	22
3.3.2 Schnittstelle für ThingSpeak in Python . . . . .	24
3.4 Berechnung meteorologischer Größen . . . . .	25
3.4.1 Das Modell der idealen Gase . . . . .	25
3.4.2 Sättigungsdampfdruck bestimmen . . . . .	26
3.4.3 Absolute Luftfeuchtigkeit bestimmen . . . . .	27
3.4.4 Taupunkttemperatur bestimmen . . . . .	28
3.4.5 Feuchttemperatur bestimmen . . . . .	29

<b>4 WebPi - Webserver einrichten</b>	<b>30</b>
4.1 Raspberry Pi als Server einrichten . . . . .	30
4.1.1 Notwendige Software installieren . . . . .	30
4.1.2 Subdomain registrieren . . . . .	35
4.1.3 Virtual Host einrichten . . . . .	36
4.1.4 Umleitung statt Fehlermeldungen . . . . .	37
4.2 Wichtige Bestandteile der Webseite . . . . .	38
4.2.1 Dynamische Webinhalte mit PHP erstellen . . . . .	38
4.2.2 Webseite auf dem Endgerät mit <i>Media Queries</i> anpassen . . . . .	38
4.2.3 Visualisieren der Messdaten mit Chart.js . . . . .	38
4.2.4 Impressum und Bildlizenzen . . . . .	39
4.3 Die Sicherheit des Servers erhöhen . . . . .	39
4.3.1 SSH-Schlüssel erstellen . . . . .	39
4.3.2 SSH-Daemon modifizieren . . . . .	40
4.3.3 CA-Zertifikat mit Let's Encrypt erstellen . . . . .	40
4.3.4 Firewall einrichten . . . . .	44
4.4 Systemsteuerung: Automatisierung, Aktualisierungen und Systemüberwachung . . . . .	45
4.4.1 Softwareupdates durchführen . . . . .	45
<b>5 Ausblick auf Erweiterungen</b>	<b>47</b>
<b>Appendices</b>	<b>48</b>
<b>A Verwendete Bibliotheken</b>	<b>50</b>
<b>B Erstellte Skripte</b>	<b>51</b>
<b>C Begriffsverzeichnis</b>	<b>53</b>
<b>D Abkürzungsverzeichnis</b>	<b>54</b>
<b>E Liste aller Materialien</b>	<b>55</b>

# Abbildungsverzeichnis

1.1	Raspberry Pi 1 Modell B - Autor: Phillip Bohk - Lizenz: CC BY-SA 3.0 . . . . .	7
1.2	Raspberry Pi Zero - Lizenz: gemeinfrei . . . . .	7
3.1	DHT22: Digitales Hygrometer, angesteuert über 1-Wire . . . . .	12
3.2	BMP180: Digitales Barometer, angesteuert über IIC . . . . .	13
3.3	Spannungsregler am GY68: VDD hat eine Spannung von +3.3V . . . . .	13
3.4	Vollständige Schaltung der Wetterstation . . . . .	15
3.5	Die Wettermessinstrumente (WH14C) auf dem Balkon befestigt . . . . .	16
3.6	Gehäuse für die Schaltung . . . . .	16
3.7	Schaltung der Wetterstation auf dem Steckbrett. . . . .	16
3.8	Aufbau der Datenübermittlung - Bildquelle: Datenblatt des DHT22 . . . . .	17
3.9	Datenübermittlung über Signale bestimmter Dauer - Bildquelle: Datenblatt des DHT22	17
3.10	Datenübertragung beim BMP180 über $I^2C$ - Bildquelle: Datenblatt des BMP180 . .	18
3.11	Windmesser und Niederschlagsmesser: Messungen über Magnetschalter . . . . .	21
3.12	Bildschirmfoto des Channels auf ThingSpeak . . . . .	23
3.13	Magnus-Formel. Abbildung erstellt in Geogebra <i>Achse</i> : Sättigungsdampfdruck in hPa, $x - Achse$ : Temperatur in ° C . . . . .	26
4.1	Portweiterleitung beim Router (FritzBox) . . . . .	34
4.2	Bildschirmfoto: Erstellen der Schlüssel im Terminal . . . . .	39

# Kapitel 1

## Vorwort

### 1.1 Motivation zum Projekt

'Wichtig ist, dass man nie aufhört, zu fragen ...'

Albert Einstein

Die Geschichte des Projekts beginnt im Jahr 2013. Im selben Jahr hatte ich zu Weihnachten einen Raspberry Pi B erhalten. Ich war begeistert von dem kleinen Mini-Computer, der mir damals noch ziemlich unbekannt war. Zunächst wusste ich also nichts mit ihm anzufangen. Um mich mit ihm vertraut zu machen, hatte ich in den folgenden Monaten nach Möglichkeiten gesucht, ihn einzusetzen. Bald fiel mir ein, dass ich im Informatikunterricht während der siebten Klasse als Projektarbeit eine kleine Webseite erstellt hatte. Zu Anfang war sogar angedacht, die erstellten Webseiten des Informatikkurses ins Internet zu stellen. Die Dateien würden auf einem Server gespeichert werden und unter einer Internetadresse aufrufbar sein. Leider wurde bis zum Ende des Halbjahres kein geeigneter und kostenfreier Server gefunden. Trotzdem war ich noch immer von der Idee begeistert, eine Webseite zu erstellen, dass ich den Gedanken wieder aufgriff. So kam im Laufe der Zeit die Grundidee meines Projektes zustande, die Webseite mithilfe des Raspberry Pi zu erstellen, indem ich ihn als Webserver einrichte, welcher die Anfragen selber entgegennimmt.

Ich habe die ursprüngliche Webseite aus meinem Informatikprojekt mehrmals überarbeitet. Sie war gut strukturiert, doch es war nicht sinnvoll, den zuvor verwendeten Inhalt der Webseite weiterzuführen. Ursprünglich sollten Notizen aus der Schulzeit über das Internet abrufbar sein, damit Besucher sich zu den verfügbaren Themengebieten informieren können. Jedoch waren bisher kaum Notizen vorhanden und die Inhalte der Notizen bereits auf anderen Webseiten vorhanden, welche noch dazu besser waren als die Notizen. Daher lohnte sich die Weiterarbeit daran nicht. Es fehlte der Webseite also an einem geeigneten Inhalt.

Parallel zum Einrichten des Webservers begann ich, mit Sensoren am Raspberry Pi zu experimentieren. Mich reizte die Möglichkeit, mit den Pins des Raspberry Pi zu spielen, sodass ich kleine Schaltungen entwarf und diese realisierte. Beispielsweise hatte ich eine kleine LED-Lampe erstellt, Texte auf einem 2x16 Ziffern großen Display anzeigen lassen und mit Gassensoren die Veränderung der Luftqualität versucht zu messen, dessen Ergebnisse jedoch wegen der Kalibrierung und der Übersensibilität für mehrere Gase nicht zufriedenstellend waren.

In den folgenden Experimenten reizten mich die Messungen eines Hygrometers, mit welchem ich die Temperatur und die Luftfeuchtigkeit auslesen konnte. Ich entschied mich, als nächstes ein Barometer hinzuzufügen, damit ich mit zusammen mit den Messungen des Luftdruckes eine kleine Wetterstation habe. Im Unterschied zu den kleinen Experimenten handelte es sich bei diesen Sensoren um Messungen, mit denen kleine Schlussfolgerungen und Beschreibungen über die Umgebung möglich waren. So lässt sich beispielsweise der Taupunkt über die Magnus-Formel berechnen und mit dem Verlauf des Luftdruckes eine Tendenz der Wetterentwicklung absehen. Die erhaltenen Daten lassen sich also anders als die anderen Bauteile interpretieren und auf mathematischer Ebene weiternutzen.

Rückblickend auf das gesamte Projekt hat mir gerade das Tüfteln mit diesen Sensoren besonders großes Vergnügen bereitet, da sich hier Physik, Informatik und Mathematik überschneiden. Für die Beschaltung und Messung ist Physik notwendig, für die Informationsverarbeitung, -speicherung und -darstellung ist Informatik notwendig und für den Umgang mit den Messungen die Mathematik. Dadurch, dass ich nicht festgelegt war in dem, was ich mache, konnte ich viele verschiedene Dinge in jedem der Bereiche versuchen. Abschließend freut es mich nun, meinen Balkon als Messstation eingerichtet zu haben und praktisch vor der Balkontür etwas Naturwissenschaft betreiben zu können.

## 1.2 Was ist ein Raspberry Pi?

Zuallererst sollte der Hauptbestandteil des Projekts, der Raspberry Pi, einmal genauer betrachtet werden. Daher soll im Folgenden nur auf die wesentlichen Eigenschaften eingegangen werden. Auf einzelne Eigenschaften des Raspberry Pi, die für die folgenden Kapitel nicht relevant sind, wird daher nicht gesondert eingegangen.

Als Raspberry Pi (engl. „Himbeerkuchen“) wird allgemein ein in England hergestellter Mini-Computer bezeichnet, welcher seinen Namen durch eine Kombination aus einer Himbeerre und der Abkürzung für 'Python Interpreter' (Pi) erhielt. Er wird von der Raspberry Pi Foundation vertrieben.

Im Jahre 2007 begann Eben Upton, ein englischer Akademiker an der Universität in Cambridge, mit der Entwicklung des Mini-Computers. Das Ziel sollte es sein, Studienbewerber wieder in Berührung mit der technischen Seite der Computer zu bringen. Upton hatte zuvor in Bewerbungsgesprächen mit Studienbewerbern festgestellt, dass keiner von ihnen zu wissen schien, was ein Computer wirklich sei oder wie er funktionieren würde. Upton begann daher zusammen mit gleichgesinnten Lehrern, Akademikern und Computerenthusiasten mit dem Entwurf eines Mini-Computers, der zugleich aufregend, klein aber zugleich günstig sein soll. (vgl. [3])

Mit dem Raspberry Pi B und dem Raspberry Pi A, den ersten Modellen des Raspberry Pi, die in großer Stückzahl verkauft wurden, ist er ein beliebter Bestandteil in kleinen Bastelprojekten geworden. Inzwischen wurden mehr als 10 Millionen Exemplare des Raspberry Pi weltweit verkauft. (vgl. [6])

Derzeit gibt es acht verschiedene Modelle von Raspberry Pis. Die ersten beiden Modelle, der Raspberry Pi B und Raspberry Pi A, sind seit dem Verkaufsstart im Jahre 2012 auf dem Markt erhältlich. Seit 2012 wurde die Auswahl durch verbesserte Modelle erweitert.

In meinem Projekt werden zwei verschiedene Raspberry Pis verwendet. Der Raspberry Pi 1 Modell B wird in meinem Projekt als Webserver genutzt, welcher über das angeschlossene LAN-Kabel mit dem Internet verbunden ist. Frühe Versuche, den Webserver über einen WLAN-USB-Stick zu betreiben, endeten damit, dass die Verbindung manchmal abgebrochen war und manchmal für Anfragen nicht verfügbar war. Daher ist der Internetanschluss über das LAN-Kabel die geeignetere Lösung gewesen, da er danach nahezu durchgehend verfügbar war und die Ladezeit deutlich niedriger war als bei der Verwendung des WLAN-USB-Sticks.

Als Prozessor besitzt er den BCM2835 mit einer ARMv6-Architektur. Er ist auf 700MHz getaktet, besitzt einen Kern und verfügt über 512 Megabyte Arbeitsspeicher. Er ist 85 Millimeter lang und 56 Millimeter breit. Mit diesen Eigenschaften lässt sich seine Rechenleistung und seine Größe mit einem Smartphone vergleichen. Auf dem Raspberry Pi befindet sich jedoch kein Speichermedium für das Betriebssystem. Dieser muss mittels einer SD-Karte ergänzt werden, wofür ein SD-Karten-Anschluss vorhanden ist. Auf dieser SD-Karte befinden sich dann die Systemdateien und die Benutzerdaten. (Siehe Kapitel 2)

Des Weiteren verfügt der Raspberry Pi B über einen Klinkenanschluss, einen Videoanschluss sowie einen HDMI-Anschluss, über welchen die grafische Oberfläche auf einem Monitor angezeigt werden kann. Die Bedienung findet hierbei üblicherweise über eine USB-Maus und ein USB-Keyboard statt. Für den Webserverbetrieb lässt sich die anzuschließende Hardware vernachlässigen, da ein über eine Internetverbindung auf den Raspberry Pi zugegriffen wird. (Siehe Kapitel 2.2.2)

Weil der Raspberry Pi um ein Vielfaches kleiner ist als ein üblicher Computer, verbraucht er auch viel weniger Strom. Mit einer Spannung von fünf Volt am Micro-USB-Anschluss als Stromquelle verbraucht er weniger als 5 Watt, bei Höchstleistung mit übertaktetem Prozessor nicht mehr als 6 Watt. Der Raspberry Pi B besitzt zudem zwei USB-Anschlüsse, einen Ethernet-Anschluss für LAN-Kabel und durch 26 GPIO-Anschlüsse. (siehe Bild auf nächster Seite) Mit diesen GPIO-Pins Anschlüsse können Schaltungen mit dem Raspberry Pi angesteuert werden. Beispielsweise lassen sich LEDs ein- und ausschalten, Messungen von Sensoren auslesen oder allgemeiner gefasst die Spannung an den Pins so ändern, wodurch Signale zum Übertragen von Daten erzeugt werden.

Vergleichbar dazu ist der Raspberry Pi Zero, welcher im November 2017 erschienen ist und sich in wenigen Eigenschaften besonders von dem Raspberry Pi B unterscheidet. Der Raspberry Pi Zero wird im Projekt für das Auslesen der Messungen verwendet.

Ihn zeichnet besonders seine Größe aus. Mit 65mm Länge und 30mm Breite ist er kleiner als eine Kreditkarte. Trotzdem hat er nicht aufgrund der geringen Größe an Rechenleistung verloren. Wie auch seine Vorgängermodelle besitzt der Raspberry Pi Zero den BCM2835 als Prozessor, welcher über die Jahre verbessert wurde und mit 1 GHz getaktet. Weil er kleiner geworden ist, befindet sich statt einem HDMI-Anschluss ein Mini-HDMI-Anschluss, statt einem SD-Karten-Anschluss ein Micro-SD-Anschluss und statt einem USB-Anschluss ein Micro-USB-Anschluss auf der Platine. Zudem besitzt er nicht 26 GPIO-Pins wie der Raspberry Pi B, sondern 40 GPIO-Pins.

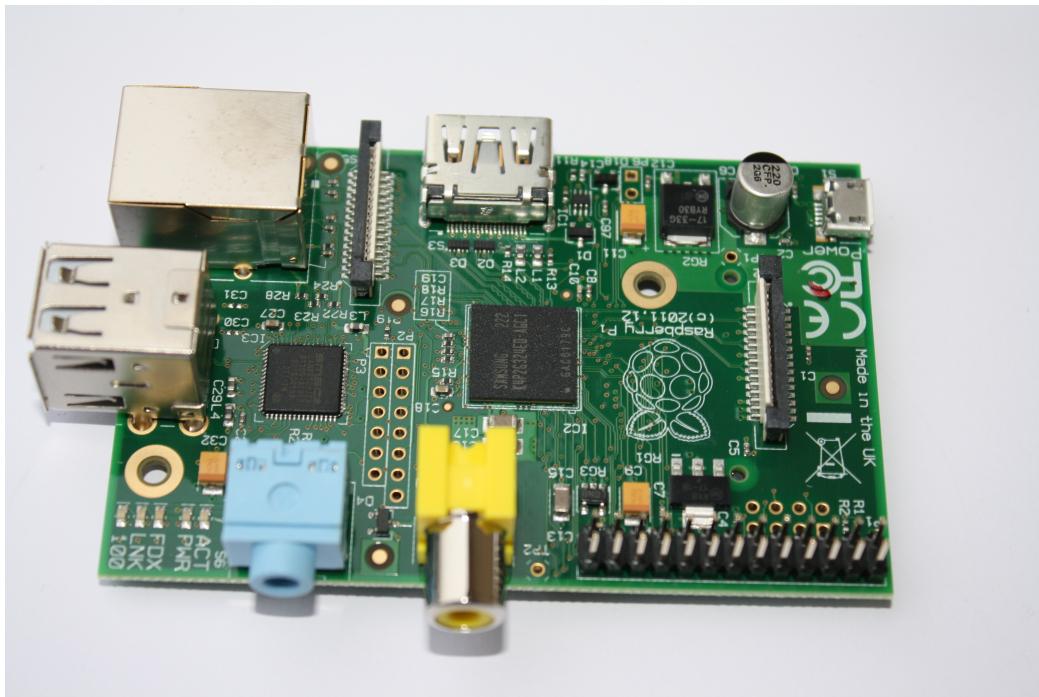


Abbildung 1.1: Raspberry Pi 1 Modell B - Autor: Phillip Bohk - Lizenz: CC BY-SA 3.0

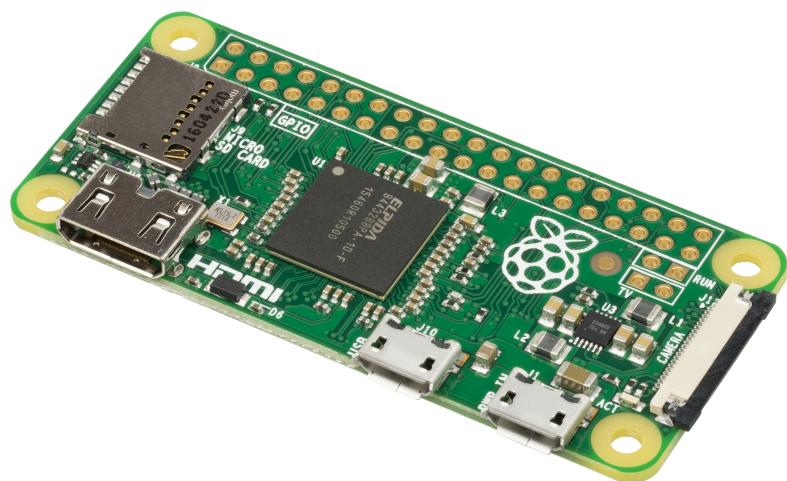


Abbildung 1.2: Raspberry Pi Zero - Lizenz: gemeinfrei

## Kapitel 2

# Raspberry Pi - Betriebssystem einrichten

Bevor es mit dem Einrichten des Webservers und dem Entwerfen einer Schaltung losgehen kann, muss der Raspberry Pi eingerichtet werden. Anders als bei anderen Geräten wie Smartphones ist hier der Mini-Computer nicht von Werk aus auf den Verwendungszweck ausgerichtet. Im Gegenteil: Der Speicher für das Betriebssystem des Raspberry Pi fehlt, die Peripherie ist nicht vorhanden und Ausgaben vom Raspberry Pi - beispielsweise den Desktop - sieht man ohne angeschlossenen Bildschirm auch nicht. Dafür kann man ihn selber auf den gewünschten Einsatzzweck ausrichten.

Zu dem Thema, wie man mit dem Raspberry Pi einsteigt, gibt es in der Technikwelt einiges an Literatur, welche die notwendigen Einstellungen kleinschrittig erläutern. Zudem gibt es auf der Internetplattform der Raspberry Pi Foundation eine Dokumentation, welche die Installation erleichtert. Daher möchte ich mich an dieser Stelle kurz fassen und nur die wesentlichen Schritte darstellen, welche ich zum Einrichten des Systems benötigte.

### 2.1 Betriebssystem herunterladen, SD-Karte vorbereiten, Raspbian einrichten

#### 1. Ein Betriebssystem auf die SD-Karte aufspielen

Die SD-Karte beim Raspberry Pi entspricht etwa der Festplatte bei Computern. Um beispielsweise das Betriebssystem Windows auf einem Computer zu installieren, werden üblicherweise CDs benutzt, welche den Benutzer nach Einlegen jener CD im CD-Laufwerk durch die Installation leiten und die Systemdateien auf die Festplatte aufspielen. Anders ist es beim Raspberry Pi: Die Betriebssysteme, welche für den Raspberry Pi verfügbar sind, befinden sich als Systemabbilddatei zum Download im Internet oder sind im Fachhandel als vorinstallierte SD-Karte zu kaufen. Um das Betriebssystem zu installieren, sind im Regelfall zwei Schritte notwendig.

**Schritt 1:** Erstmal muss das Systemabbild heruntergeladen werden. Hier muss man sich schon entscheiden, welches der verfügbaren Betriebssysteme auf dem Raspberry Pi laufen sollen. Ich habe mich für Raspbian entschieden, weil es das von der Raspberry Pi Foundation empfohlene und am besten unterstützte Betriebssystem für den Raspberry Pi ist. Die Datei des Systemabbildes lässt sich unter folgendem Link finden: <https://www.raspberrypi.org/downloads/raspbian/>

**Schritt 2:** Sodann folgt das Extrahieren der Datei auf eine SD-Karte. Da ich auf meinem Computer Ubuntu als Betriebssystem nutze, lässt sich der Ubuntu Disc Creator nutzen. Das Betriebssystem wird automatisch so eingerichtet, dass das System starten kann, nachdem die SD-Karte in den Raspberry Pi eingesetzt wird. Der Raspberry Pi fährt das System automatisch hoch, wenn eine Stromquelle an den Mikro-USB-Anschluss angeschlossen ist.

## 2. Auf das Betriebssystem zugreifen

Um nun auf das gestartete System zuzugreifen, gibt es verschiedene Möglichkeiten. Die am einfachsten handzuhabene Option ist, den Raspberry Pi an einen Monitor, eine Tastatur und ein Keyboard anzuschließen. Ich habe jedoch eine andere Option gewählt. Über eine die UART-Schnittstelle des Raspberry Pi lassen sich direkt Befehle über den Terminal in Ubuntu an den Raspberry Pi senden.

Dazu verbindet man den Raspberry Pi mithilfe des PL2303 (ein USB-zu-UART-Adapter) an den USB-Anschluss des Computers. Bevor eine die Verbindung gestartet werden kann, muss noch das Programmpaket „screen“ in Ubuntu installiert werden.

```
sudo apt-get install screen
```

Um die serielle Verbindung zu starten, wird folgender Befehl ausgeführt:

```
screen /dev/ttyUSB0 115200
```

Damit kann ich mich direkt über den Terminal meines Ubuntu-Computers beim Raspberry Pi anmelden und über den Terminal Bash-Befehle ohne Netzwerkverbindung an den Raspberry Pi senden.

## 3. WLAN konfigurieren (Raspberry Pi Zero)

Der Raspberry Pi Zero verbindet sich mit dem Heimrouter über WLAN durch einen angeschlossenen WLAN-Adapter. Dazu benötigt er jedoch das WLAN-Passwort des Routers. Diese Einstellung kann auf verschiedene Arten vorgenommen werden. Neben der oben genannten Methode kann die SD-Karte mit dem aufgespielten Raspbian- Betriebssystem bearbeitet werden. Dazu wird die folgende Datei unter Ubuntu über den Terminal als Superuser editiert. Dort werden das entsprechende Passwort und der Name der Routers eingetragen.

Die eckigen Klammern dienen hier als Platzhalter, in diesem Fall für das eigentliche Passwort. Sie sind so nicht in der Datei oder im Befehl vorhanden.

```
sudo nano /media/[Name-der-Partition]/etc/wpa_supplicant/wpa_supplicant.conf
```

```
-----  
country=DE  
ctrl_interface=DIR=/var/run/wpa_supplicant GROUP=netdev  
update_config=1  
  
network={  
    ssid='[SSID des Routers]',  
    psk='[WLAN-Passwort]',  
}  
-----
```

## 2.2 Grundlegende Einstellungen am Raspberry Pi vornehmen

Das Betriebssystem wurde eingerichtet, der Raspberry fährt hoch. Aber was dann? Im Folgenden geht es um die Einstellungen innerhalb des Betriebssystems Raspbian Jessie.

### 2.2.1 Standardpasswort ändern

Wenn der Raspberry Pi zum ersten Mal gestartet wird, sind bereits zwei Benutzer ab Werk vorhanden. Der eine Standardbenutzer *pi* besitzt ein Passwort, welches auf allen Systemabbilddateien gleich ist, nämlich *raspberrypi*. Der Benutzer *pi* ist nicht der Administrator des Systems, das ist der Benutzer *root*, auch bekannt als *Superuser*. Trotzdem kann *pi* die Rechte eines Administrators anfordern. Dies liegt daran, dass er in der Datei „/etc/sudoers“ eingetragen ist, wodurch er Rechte eines Administrators anfragen kann.

Für kleinere Anwendungen mag es zwar akzeptabel sein, das Passwort nicht zu ändern, doch für jeden an das Internet angeschlossenen Raspberry Pi ergibt sich dadurch die Möglichkeit, Zugriff über das Netzwerk zu erhalten. Angreifer können versuchen, sich mithilfe des Standardpasswortes einfach Zugriff zum System zu verschaffen. Dadurch, dass der Benutzer *pi* Administratorrechte anfordern kann, kann ein Angreifer die volle Kontrolle über das System übernehmen, wenn der SSH-Port offen gelassen wird.

Daher ist das Ändern des Passwortes ein einfaches, wichtiges und zugleich wirksames Mittel, die Sicherheit des Systems zu erhöhen, sofern ein starkes Passwort gewählt wird. Um diese Änderung vorzunehmen, wird der Terminal beim Raspberry Pi geöffnet. Dort wird das Konfigurationsmenü des Raspberry Pi nach Ausführung des unten abgebildeten Befehls geöffnet. Über das erscheinende Menü kann im Menüpunkt „Change Password“ das Passwort geändert werden.

```
sudo raspi-config
```

Das *sudo* im obigen Befehl weist den Raspberry Pi dazu an, das Konfigurationsmenü für grundlegende Einstellungen am Raspberry Pi mit Administratorrechten zu öffnen. Diese muss erstmal mithilfe aktuell gültigen Standardpasswortes „*raspberrypi*“, autorisiert werden.

### 2.2.2 Secure Shell eingeschalten

Die Raspberry Pi Foundation hatte seit einem Systemupdate im November 2016 standardmäßig das Anmelden über die Internetverbindung deaktiviert. Die offizielle Begründung dafür ist die gleiche wie in dem obigen Unterkapitel: Die Standardpasswörter wurden nicht geändert und die Raspberry Pis waren ab Werk für Zugriffe von außerhalb offen, wenn sie sich in einem öffentlichen Netzwerk befanden. (vgl. [2])

In meinem Fall war nur beim Raspberry Pi Zero die Secure Shell deaktiviert, doch für Zugriffe aus dem öffentlichen Netzwerk fehlt ein Port. Daher kann nur aus dem lokalen Netzwerk auf die Raspberry Pis zugegriffen werden. Um auf den Raspberry Pi über das lokale Netzwerk zuzugreifen, bin ich auf eine die SSH-Verbindung angewiesen. Daher schalte ich den Dienst ein, der Port ist nicht vorhanden. Zusätzliche Sicherheitsvorkehrungen dazu gibt es im Kapitel 4.

### **2.2.3 Hostname festlegen**

Damit man im Netzwerk die Raspberry Pis voneinander unterscheiden kann, werden individuelle Netzwerknamen (engl. hostname) benötigt. Diese lassen sich einfach über das eben verwendete Konfigurationsmenü des Raspberry Pi einrichten. Der Raspberry Pi Zero, der sich auf dem Balkon befindet und an den Sensoren angeschlossen ist, wird *WeatherPi* genannt. Der Raspberry Pi B, der direkt an den Router angeschlossen ist, wird *WebPi* genannt.

### **2.2.4 Mehr Zwischenspeicher bereitstellen**

Der BCM2835 verarbeitet alle Prozesse des Betriebssystems. Für Prozesse, mit welchen graphische Elemente verarbeitet werden, wird ein Zwischenspeicher im Prozessor zur Verfügung gestellt. Die Menge an Zwischenspeicher kann über das Konfigurationsmenü des Raspberry Pi angepasst werden. In dem Falle beider Raspberry Pi werden nahezu keine rechenintensiven Prozesse mit graphischen Elementen wie die Desktopoberfläche von Raspbian verarbeitet. Es ist kein Bildschirm an den Raspberry Pi angeschlossen. Daher kann dieser Speicher klein gewählt werden, um den Zwischenspeicher für nicht nur für graphische, sondern für alle Prozesse freizugeben. Unter folgendem Menüpunkt des Konfigurationsmenüs wird dort der Zwischenspeicher auf 16 statt 64 Megabyte gesenkt.

`Advanced Options >> Memory Split`

### **2.2.5 Sprachumgebung festlegen**

Ab Werk ist das englische Tastaturlayout voreingestellt. Damit im Terminal die Eingaben einer deutschen Tastatureingabe richtig erkannt werden, muss die Sprachumgebung des Raspberry Pi um das deutsche Tastaturlayout erweitert werden. Dazu ist eine simple Änderung im Konfigurationsmenü des Raspberry Pi notwendig. Unter folgendem Menüpunkt können jene ausgewählt werden.

`Internationalisation Options >> Change Locale`

Im Falle der deutschen Tastatur soll der Raspberry Pi alle deutschen Sprachumgebungen generieren. Mit dem Neustart des Raspberry Pi treten die folgenden Sprachumgebungen in Kraft.

`de_DE ISO-8859-1, de_DE@euro ISO-8859-15, de_DE.UTF-8 UTF-8`

### **2.2.6 Übertakten des Prozessors**

Standardgemäß ist der Prozessor des Raspberry Pi B auf 700 Megahertz getaktet. Es ist aber möglich, den Prozessor auf Taktraten von 800 bis 1000MHz hochzutakten. Das Hochtakten ist jedoch umstritten, weil des Vor- und Nachteile hat. Einerseits können durch eine höhere Taktrate mehr Prozesse in derselben Zeit verarbeitet werden, wodurch die Leistung des Raspberry Pi prinzipiell zunimmt. Andererseits werden die Bauteile dadurch stärker beansprucht. Mehr Energie wird übertragen, die Bauteile werden wärmer. Ein Überhitzen ist aber nicht möglich. Dafür gibt es einen Schutzmechanismus, der den Raspberry Pi bei zu hoher Temperatur des Prozessors herunterfährt.

In meinem Fall habe ihn auf 900MHz hochgetaktet und einen Kühlkörper an den Prozessor geklebt, welcher die Wärme leichter an die Umgebungsluft abgeben kann und die Temperatur des Prozessors damit senkt. Da der Raspberry Pi B als Server die meiste Zeit als Webserver im Leerlauf läuft, ist nicht mit andauernder Beanspruchung zu rechnen.

# Kapitel 3

## WeatherPi - Raspberry Pi beschalten

Der Raspberry Pi ist nun eingerichtet. Und nun? Nun wird der Raspberry Pi Zero beschaltet! Genauer gesagt geht es darum, welche Messinstrumente für die Messungen der Wetterdaten verwendet werden, wie diese funktionieren, wie der Raspberry Pi Zero diese ausliest und was sich aus den Messungen interpretieren lässt.

### 3.1 Welche Sensoren werden verwendet?

#### 3.1.1 Thermometer und Hygrometer: DHT22

Der Sensor DHT22 misst die Temperatur und die Luftfeuchtigkeit. Er besitzt vier Pins, wovon drei zu beschalten sind. An Pin 1 wird die Spannung angelegt (3.3 Volt), an Pin 4 die Masse. Über Pin 2 werden die Daten übertragen. Der Pin 2 ist zudem durch einen Pull-Up-Widerstand an die Referenzspannung angeschlossen.

Der DHT22 misst die Temperatur mit einer Auflösung von  $\pm 0.1^\circ$  Celsius ziemlich genau. Die Genauigkeit der Messung liegt generell bei  $\pm 0.5^\circ$  Celsius; sie ist von der Temperatur abhängig. Bei der Luftfeuchtigkeit besitzt der Sensor eine Genauigkeit von  $\pm 2\%$  Luftfeuchtigkeit im Bereich von 20% bis 90% relativer Luftfeuchtigkeit in der Umgebung.

Wie sich bei den Messungen über den Raspberry Pi herausstellte, übermittelt der DHT22 einige Messungen träge bis gar nicht. Diese Trägheit scheint zusätzlich für den DHT22 charakteristisch zu sein.

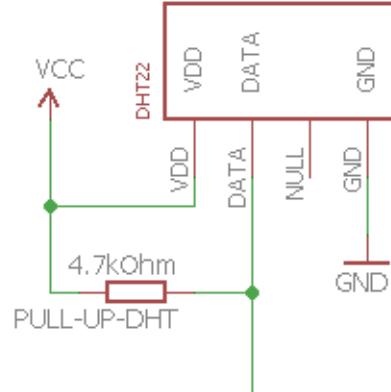


Abbildung 3.1: DHT22: Digitales Hygrometer, angesteuert über 1-Wire

### 3.1.2 Thermometer und Barometer: BMP180

In meiner Schaltung wird das Bauteil GY68 verwendet. Dieses beschreibt eine Schaltung auf einer Leiterplatte, auf dem sich das Barometer BMP180 befindet. Im Folgenden wird daher vom BMP180 als Sensor und GY68 als Bauteil gesprochen.

Der GY68 besitzt fünf Pins. Über Pin 1 (Spannung) und Pin 2 (Masse) wird der Sensor mit Strom versorgt. Anzumerken ist, dass sich auf dem GY68 ein Spannungsregler befindet. Der BMP180 darf laut Datenblatt mit höchstens 3.6 Volt betrieben werden. Durch den Spannungsregler des GY68 wird die angelegte Spannung jedoch auf 3.3 Volt reguliert, indem der zugeführte Strom erst durch den Spannungswandler geleitet wird. Dadurch können auch Spannungen angeschlossen werden, die eine höhere Spannung als 3.3 Volt besitzen, ohne dass der BMP180 beschädigt wird. Die Kondensatoren an dem Spannungswandler dienen dazu, den Strom des zu „glätten“. Da die Referenzspannung ohnehin kleiner als 3.6 Volt ist, kann der Spannungsregler prinzipiell vernachlässigt werden.

Über Pin 3 und Pin 4 werden die Daten übertragen. Zwei Pull-Up-Widerstände ( $4.7\text{k}\Omega$ ) sind an die 3V3 des Spannungsreglers angeschlossen, welche die Spannung an den Leitungen hoch halten. Zudem besitzt der BMP180 vier verschiedene Modi, über die er angesteuert werden kann (Ultra Low, Standard, High Resolution, Ultra High Resolution). Dadurch wird festgelegt, wieviele Messungen der BMP180 intern anfertigt (1, 2, 4 oder 8). Der gewählte Modus bestimmt außerdem darüber, wie hoch das Rauschen in den Messungen und der maximale Stromverbrauch sind.

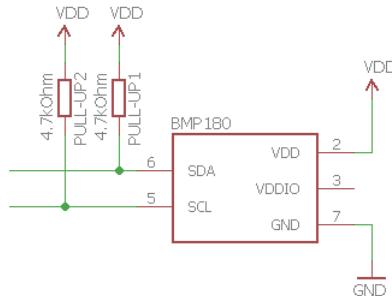


Abbildung 3.2: BMP180: Digitales Barometer, angesteuert über IIC

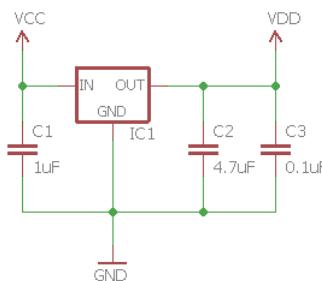


Abbildung 3.3: Spannungsregler am GY68: VDD hat eine Spannung von +3.3V

### **3.1.3 Anemometer, Anemoskop und Pulviometer**

Bei der Wetterstation WS1080 handelt es sich um eine Funk-Wetterstation mit vier Messinstrumenten. Die Messinstrumente sind untereinander per Kabel verbunden. Über Funk werden die Daten mit der Frequenz von 866 MHz an die mitgelieferte Station gesandt, welche diese auf dem Display anzeigt. In meinem Projekt werden nur die Messinstrumente dieser Wetterstation verwendet. Diese Messinstrumente werden zusammengefasst als WH14C bezeichnet.

(Auf der Rückseite des Thermometers im Set ist diese Bezeichnung auch zu finden.)

Ich habe die Messinstrumente ohne die Station käuflich erworben. Über einen RJ11-Adapter habe ich die Kabel der Messinstrumente mit dem Raspberry Pi auf einem Steckbrett verbunden. Da vom Verkäufer im Datenblatt des WH14C keine Angabe zu finden ist, welche die Schaltung der Messinstrumente darstellt, beziehe ich mich teilweise auf das Datenblatt der bauähnlichen Wetterstation SEN-08942 von Argent Data Systems als Referenz. Nach einigen Vergleichen ließ sich feststellen, dass man die Angaben sich tatsächlich auf den WH14C beziehen kann. So zeigte das Öffnen des Anemometers, dass sich 8 Widerstände auf einer Platine befinden, die im Kreis angeordnet sind und dieselben Werte der Widerstände besaßen wie sie auf dem Referenzdatenblatt beschrieben waren.

Das Referenzdatenblatt beschreibt auch die Datenübertragung über die Kabel der Sensoren. Das eine Kabel überträgt die Signale des Anemometers und Anemoskops, welches innen vier Leitungen besitzt. Das andere Kabel überträgt die Signale des Pulviometers, welches innen zwei Leitungen besitzt. Das Anemometer ist hierbei mit dem Anemoskop verbunden.

Im Set befindet sich zusätzlich ein Bestandteil des WH14C, welches eine Kombination aus Hygrometer, Thermometer und Barometer ist. Das „Abgreifen“ der Werte ist hier nicht ohne Weiteres möglich, da alle Sensoren auf eine Platine gelötet wurden. Für die Beschaltung des Anemoskopes wurde der Vorschlag des Datenblattes umgesetzt; ein Spannungsteiler mit einem Widerstand ( $10k\Omega$ ) wurde umgesetzt. Über die Formel des unbelasteten Spannungsteilers lässt sich die Richtung berechnen. Für die Beschaltung des Anemoskops und des Pulviometers wurde die Leitung des jeweiligen Sensors über einen Pull-Up-Widerstand hoch gesetzt. (Siehe dazu nächstes Unterkapitel)

### **3.1.4 Vollständiger Schaltplan mit dem Raspberry Pi Zero**

Abschließend ist auf der folgenden Seite die gesamte Schaltung zu sehen, welche mithilfe eines Steckbretts und vielen Steckbrücken realisiert wurde.

Der Raspberry Pi Zero bildet die Verbindung zu allen Sensoren. Die Stromzufuhr für den Raspberry Pi erfolgt über ein Micro-USB-Kabel. Die Referenzspannung für die Sensoren beträgt 3.3 Volt, da Pins des Raspberry Pi nicht 5V-tolerant sind. Zudem ist der Raspberry Pi nicht in der Lage, analoge Signale zu lesen. Daher wird für das Auslesen des analogen Signals des Anemoskops ein Analog-zu-Digital-Konverter (ADC) verwendet, welcher die analogen Werte in digitale Signale umgewandelt.

Anzumerken ist, dass die einzelnen Sensoren nicht kalibriert werden konnten. Ich muss davon ausgehen, dass der BMP180 und der DHT22 bereits kalibriert wurden. Bei dem Pulviometer des WH14C muss ich annehmen, dass die Regenmenge pro Spannungsabfall der des Referenzdatenblattes entspricht. die tatsächliche Windrichtung des Anemoskop ist um etwa 25 Grad verschoben (Kalibrierung möglich).

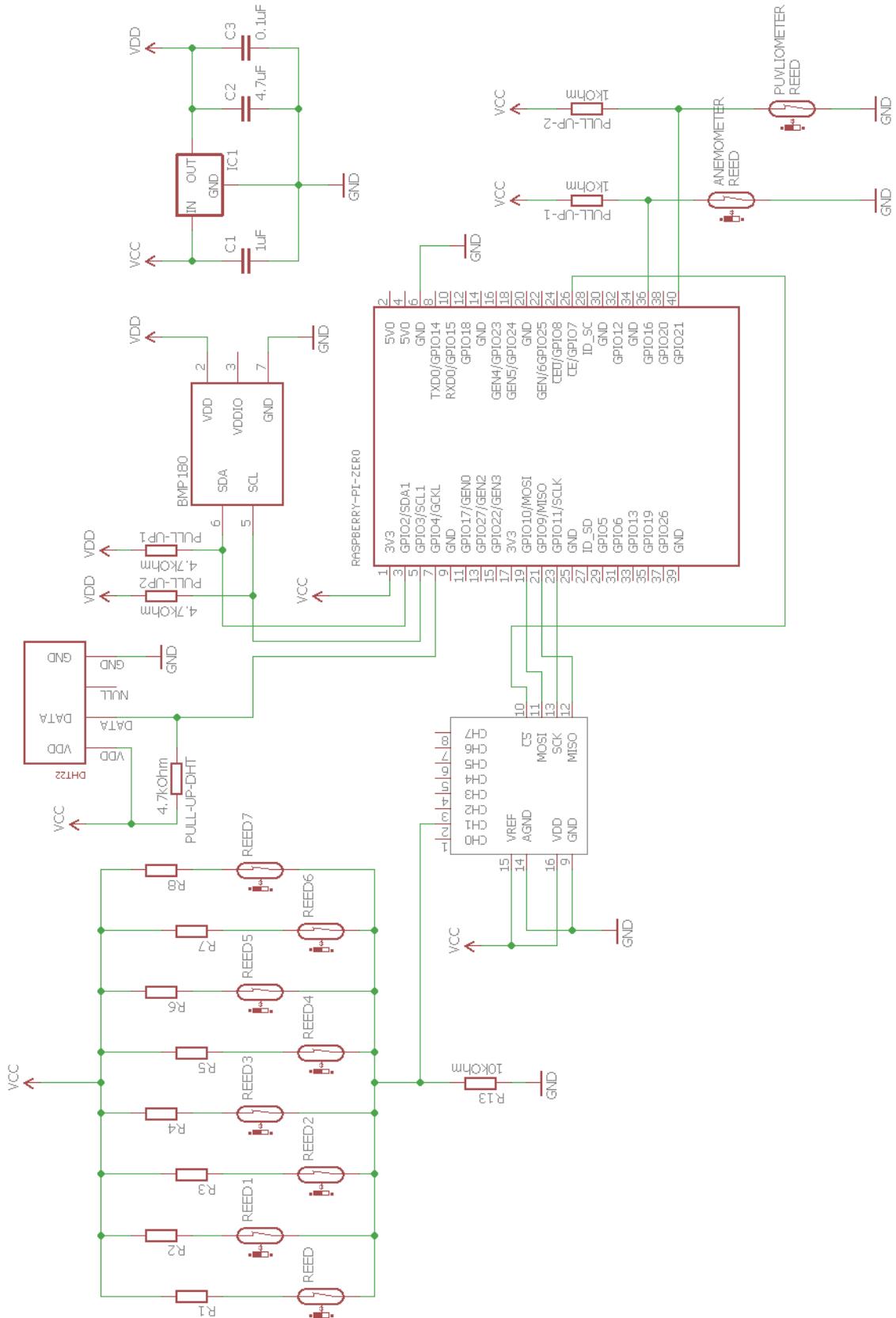


Abbildung 3.4: Vollständige Schaltung der Wetterstation



Abbildung 3.5: Die Wettermessinstrumente (WH14C) auf dem Balkon befestigt



Abbildung 3.6: Gehäuse für die Schaltung

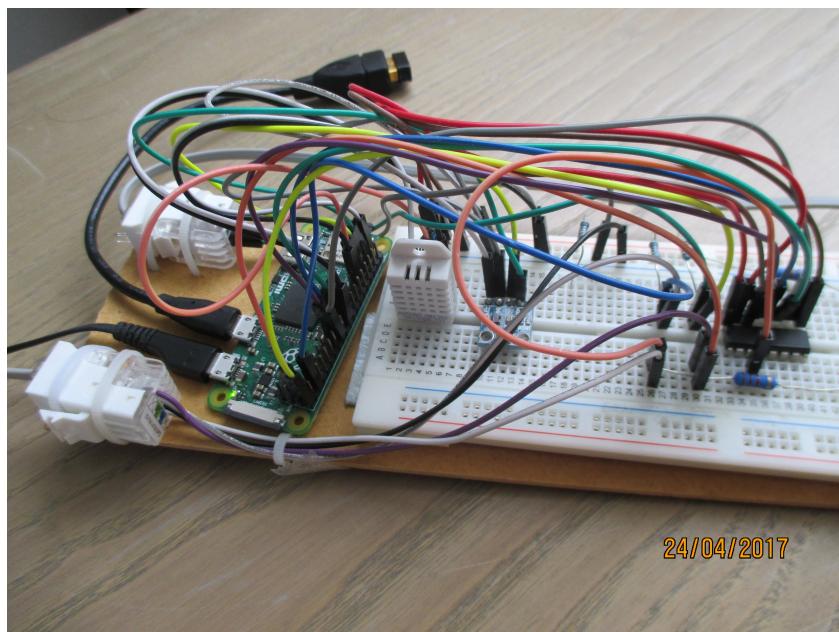


Abbildung 3.7: Schaltung der Wetterstation auf dem Steckbrett.

## 3.2 Wie werden die Messwerte übertragen?

### 3.2.1 DHT22: Digitale Signale lesen

Der DHT22 verfügt über eine Leitung, über welche die gesamte Messung digital übertragen wird. Diese Leitung ist durch einen  $4.7k\Omega$ -Widerstand an die Referenzspannung angeschlossen. Dadurch wird die Spannung an der Leitung hochgesetzt, jedoch fließt durch den hohen Widerstand kaum Strom. Dies dient dazu, das Signal klarer zu übertragen, weil die anliegende Spannung als Referenz dient. Wenn die Spannung gleich hoch bleibt, ist das Signal „high“. Wird die Spannung herabgesetzt, ist das Signal „low“.

Damit der DHT22 mit der Datenübertragung beginnt, muss der Raspberry Pi die Spannung für einige Millisekunden herabsetzen, dann kurz die Spannung wieder heben. Der DHT22 antwortet nach einigen Millisekunden auf dieselbe Art und Weise. Danach beginnt die Übertragung der Messdaten. In vier Bytes mit je acht Bits werden die Messungen übertragen. Abschließend wird ein Byte versandt, um durch Rechnung überprüfen zu können, ob die Daten korrekt angekommen sind. Danach ist die Datenübertragung beendet, die Spannung bleibt hoch.

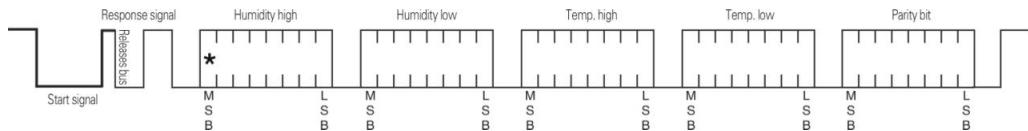


Abbildung 3.8: Aufbau der Datenübermittlung - Bildquelle: Datenblatt des DHT22

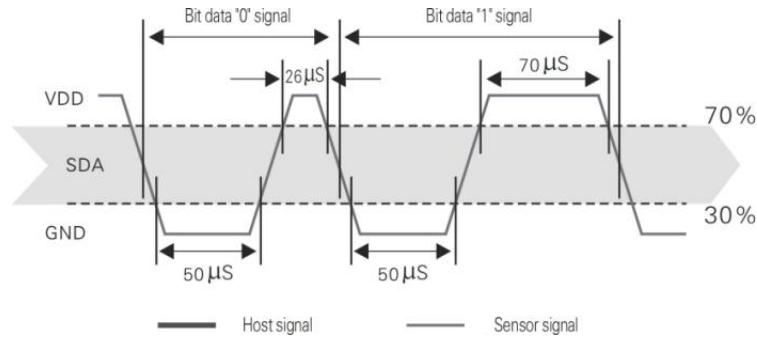


Abbildung 3.9: Datenübermittlung über Signale bestimmter Dauer - Bildquelle: Datenblatt des DHT22

Nun wird die Berechnung der einzelnen Bits betrachtet. Um bei dem Signal des DHT22 ein Bit zu bestimmen, muss die Zeit gemessen werden, wie lange beim Antwortsignal die Spannung hoch ist. Für jeden der vier Abschnitte geht dies wie folgt:

Der DHT22 setzt die Spannung für etwa 50 Millisekunden herab. Sodann setzt er die Spannung für eine gewisse Zeit wieder hoch. Ist die Spannung für etwa 26-28 Millisekunden hoch, so wird das Signal als 0 interpretiert. Ist die Spannung für etwa 70 Millisekunden hoch, so wird das Signal als 1 interpretiert. Dabei gilt für den DHT22 die Spannung als hoch, wenn sie mindestens 2.31 Volt bei 3.3 Volt Referenzspannung beträgt. Umgekehrt gilt die Spannung als niedrig, wenn die Spannung weniger als 1 Volt bei derselben Referenzspannung beträgt.

Die angekommenen Bits ergeben zwei Binärzahlen. Wenn diese in das Dezimalsystem umgerechnet werden, so ergibt sich die gemessene Luftfeuchtigkeit bzw. die gemessene Temperatur, welche um eine Kommastelle nach rechts verschoben wurde. Anzumerken ist, dass auch negative Messungen übertragen werden. Um zu erkennen, dass die gemessene Temperatur negativ ist, wird der höchste Bit der Temperaturbytes auf 1 gestellt. Damit ist das resultierende Ergebnis größer als 65536. Bei der Berechnung des Wertes wird der höchste Bit 1 als Minuszeichen interpretiert und der höchste Bit ignoriert.

Sollten die Signale über die Leitung falsch interpretiert worden sein, so gibt es den Prüfbyte. Dieser Prüfbyte ist die Summe der vier Bytes, die vor ihm übertragen wurden. Sollte die Summe der vier Bytes nicht mit dem Prüfbyte übereinstimmen, so liegt ein Fehler bei der Datenübertragung vor. Die angekommenen Daten müssen verworfen werden, eine neue Messung muss durchgeführt werden.

Um die Messungen des DHT über den Raspberry Pi zu erhalten, benutze ich die Bibliothek von Tony DiCola von Adafruit Industries, welche auf GitHub verfügbar ist.

Link: [https://github.com/adafruit/Adafruit\\_Python\\_DHT](https://github.com/adafruit/Adafruit_Python_DHT)

### 3.2.2 BMP180: $I^2C$ -Schnittstelle benutzen

Bei dem BMP180 gibt es im Vergleich zum DHT22 zwei Leitungen. Hierbei haben die Leitungen verschiedene Funktionen. Die eine Leitung überträgt das Signal (SDA), die andere den Takt (SCL). Über die SCL-Leitung wird ein gleichmäßiges Signal übertragen. (siehe Abbildung)

In dem Zeitraum, wo die Spannung bei der SCL hoch ist, wird über SDA ein Bit übertragen. Fällt steigt die Spannung bei SCL, so ist der Bit bei SDA zu Ende. Wenn die Spannung bei SCL erneut steigt, so beginnt der nächste Bit bei SDA. Damit werden die einzelnen Bits nicht über die Zeit, wie lange die Spannung durch den Sensor hochgesetzt wird, interpretiert, sondern durch den Zustand im zwischen zwei „Takten“. Ist die Spannung zwischen zwei Spannungsänderungen vom SCL hoch, so ist der Byte 1; umgekehrt ergibt sich eine 0.

Doch der Sensor kann mehr: Bei  $I^2C$  handelt es sich um ein Protokoll, welches Datenübertragungen in beide Richtungen ermöglicht. Der Raspberry Pi und der BMP180 können sowohl Sender („Master“) als auch Empfänger („Slave“) sein. Zusätzlich können mehrere  $I^2C$ -Geräte an die zwei Leitungen angeschlossen werden. Damit diese wissen, welchen Geräte sie kommunizieren sollen, besitzt jedes  $I^2C$ -Gerät eine Adresse, über welche sie angesprochen werden können.

Auch hier benutze ich eine Bibliothek von Tony DiCola von Adafruit Industries, um die Daten des BMP180 über den Raspberry Pi zu auszulesen.

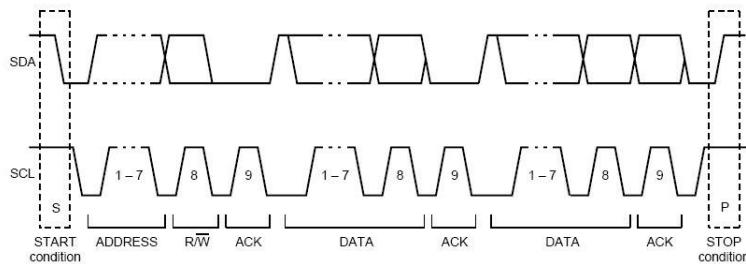


Abbildung 3.10: Datenübertragung beim BMP180 über  $I^2C$  - Bildquelle: Datenblatt des BMP180

### 3.2.3 Windrichtung des Anemoskops bestimmen

Um die Windrichtung des Anemoskops auszulesen, wird ein Spannungsteiler mit zwei Widerständen verwendet. Der erste Widerstand ist der des Anemometers ( $R_1$ ), der zweite besitzt einen Widerstand von  $10k\Omega$  ( $R_2$ ). Über den ADC wird der Teilstrom  $V_m$  zwischen den Widerständen gemessen.

Die an dem Anemometer anliegende Spannung beträgt 3.3 Volt ( $V_{in}$ ). Im Referenzdatenblatt wird die Beispielschaltung aber mit einer Spannung von 5 Volt betrieben. Damit sind die Werte für die Teilspannungen hier unbrauchbar. Daher wird nun die Beziehung zwischen dem Widerstand und der Teilspannung hergeleitet. Für einen unbelasteten Spannungsteiler gilt:

$$V_m = \frac{V_{in} \cdot R_1}{R_1 + R_2} \quad (3.1)$$

$$V_m \cdot (R_1 + R_2) = V_{in} \cdot R_1 \quad (3.2)$$

$$V_m \cdot R_1 + V_m \cdot R_2 = V_{in} \cdot R_1 \quad (3.3)$$

$$V_m \cdot R_2 = V_{in} \cdot R_1 - V_m \cdot R_1 \quad (3.4)$$

$$V_m \cdot R_2 = (V_{in} - V_m) \cdot R_1 \quad (3.5)$$

$$R_1 = \frac{V_m \cdot R_2}{V_{in} - V_m} \quad (3.6)$$

Damit kann derjenige Widerstand bestimmt werden, durch welchen die Spannung zum Zeitpunkt der Messung im Anemometer anliegt. Es muss nur noch bekannt sein, bei welchem Widerstand welche Windrichtung vorliegt. Dazu kann nun das Datenblatt der anderen Wetterstation herangezogen werden. Die unten angegebenen Spannungen gelten für eine Eingangsspannung von 3.3V.

Windrichtung	Widerstand (in $\Omega$ )	Spannung (in Volt)
0	33k	2.53
22.5	6.57k	1.3
45	8.2k	1.48
67.5	891	0.27
90	1k	0.29
112.5	688	0.21
135	2.2k	0.59
157.5	1.41k	0.409
180	3.9k	0.92
202.5	3.14k	0.78
225	16k	2.03
247.5	14.12k	1.93
270	120k	3.04
292.5	42.12k	2.66
315	64.9k	3.15
337.5	21.88k	2.26

Tabelle 3.1: Widerstand beim Anemoskop

### 3.2.4 Magnetschalter als Sensor: Windgeschwindigkeit bestimmen

Der Anemometer besteht aus drei „Löffeln“, die sich bei Wind im Winkel von 120 Grad zueinander im Kreis drehen. Er misst die Windgeschwindigkeit abhängig von den Umdrehungen pro Zeiteinheit. Innen im Anemometer befindet sich dafür ein Magnet. Bei jeder halben Umdrehung wird ein Magnetschalter durch den Magneten geschlossen. Eine der zwei Leitungen des Kabels wird nun an die Referenzspannung angeschlossen, die andere an die Masse. Wenn der Magnetschalter durch den Magneten bei einer halben Umdrehung geöffnet wird, wird die Verbindung zwischen den zwei Leitungen unterbrochen. Der Strom wird zur Masse umgelenkt; die Spannung am Pin des Raspberry Pi fällt.

Durch die Zeitdifferenz zwischen den einzelnen Spannungsabfällen ist bekannt, wie viel Zeit für eine halbe Kreisbewegung benötigt wurde. Da es sich um eine Kreisbewegung handelt, ist auch die Strecke bekannt. Damit lassen sich zunächst die Umdrehungen pro Sekunde berechnen, jedoch kann diese nicht direkt in die Geschwindigkeit umgerechnet werden. Dafür wäre eine Kalibrierung des Sensors nötig. Dies liegt daran, dass die Umdrehungen pro Zeitabschnitt nicht proportional größer werden, je höher die Geschwindigkeit wird. Die Reibung und der Luftwiderstand beeinflussen die Rotation der Schalen.

Um aber die Geschwindigkeit bei kleinen Windgeschwindigkeiten zu berechnen, werden unter der Annahme der Proportionalität die Umdrehungen in einem Zeitintervall von 360 Sekunden gemessen. Die Reibung und der Luftwiderstand bleiben klein. Der Schalter wird pro gesamter Umdrehung zwei Mal ausgelöst. Dies muss bei der Berechnung berücksichtigt werden. Da in dem Zeitraum von 360 Sekunden unterschiedliche Zeitdifferenzen auftauchen können, kann nur die Durchschnittsgeschwindigkeit aller Zeitdifferenzen in dem Zeitraum von 6 Minuten gebildet werden. Damit ist dann die Zeitdifferenz und der Streckenabschnitt definiert. Die berechnete Geschwindigkeit ist sodann proportional zu der Anzahl der Spannungsabfälle.  $n$  ist hierbei die Anzahl an Spannungsabfällen.

$$v_{\text{allgemein}} = \frac{\Delta(s)}{\Delta(t)} \quad (3.7)$$

$$s_{\text{ganz}} = n \cdot 2 \cdot \Pi \cdot \text{Radius} \quad (3.8)$$

$$s_{\text{halbrot}} = \frac{n}{2} \cdot 2 \cdot \Pi \cdot \text{Radius} \quad (3.9)$$

$$\Delta t = 360s \quad (3.10)$$

$$v_{\text{halbrot}} = n \cdot \frac{\Pi \cdot \text{Radius}}{360} \cdot \frac{m}{s} \quad (3.11)$$

$$v_{\text{final}} = v_{\text{halbrot}} \cdot 1.18 \quad (3.12)$$

$$(3.13)$$

Am Ende der Rechnung wird das Ergebnis noch mit dem sogenannten Anemometer-Faktor multipliziert. Damit wird die Abweichung korrigiert, die dadurch entsteht, dass die obige Beziehung verwendet wird.

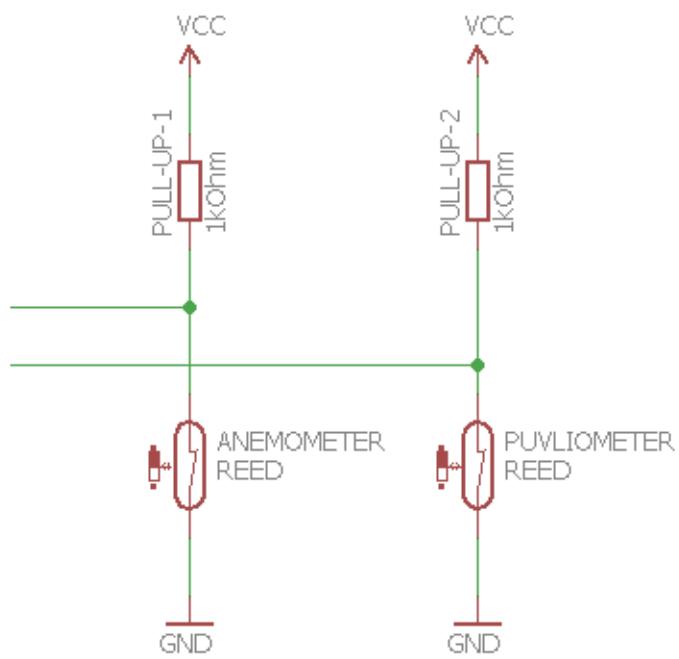


Abbildung 3.11: Windmesser und Niederschlagsmesser: Messungen über Magnetschalter

### 3.3 Internet of Things - Messungen über ThingSpeak visualisieren

In der Technikbranche ist aktuell das Thema „Internet of Things“ (IoT) hochaktuell. Dabei handelt es sich um die Idee, Alltagsgegenstände zu elektrifizieren, um sie dann mit dem Internet zu verbinden. Dadurch können Alltagsgegenstände über das Internet kommunizieren. Aktuell gibt es bereits solche Geräte: Mithilfe einer Applikation für Smartphones lässt sich der Bestand an Lebensmitteln im vernetzten Kühlschrank abrufen, die vernetzte Waschmaschine sich per App angesteuern und die vernetzte Lampe im Zimmer nebenan kann einem Klick auf dem Smartphone aus- oder eingeschaltet werden, ohne aufzustehen zu müssen.

Diese Grundidee des IoT lässt sich bei der Wetterstation wiederfinden, auch wenn eine Wetterstation kein Alltagsgegenstand ist. Es werden Daten gesammelt, gespeichert und an andere Geräte gesandt. Diese können auf einem Endgerät angezeigt werden. Auch lässt sich über eine Smartphone-App die Wetterstation über SSH direkt ansteuern. Eine App dafür zu nutzen wäre möglich, aber aus Sicherheitsgründen nicht sinnvoll. Um diese „Vernetzung“ für das Smartphone umzusetzen, habe ich mich dafür entschieden, die IoT-Platform *ThingSpeak* des amerikanischen Unternehmens *The Math Works* zu benutzen. Damit können die Messungen der Wetterstation über Apps, welche Daten von ThingSpeak-Channels anzeigen, auf Smartphones angezeigt werden.

#### 3.3.1 Channel einrichten

Die Funktionsweise von dem ThingSpeak-Channel kann man sich als Cloud vorstellen. Die Messungen der Wetterstation werden dort gespeichert. Über eine Internetverbindung kann von anderen Geräten auf die Messungen in einer Cloud zugegriffen werden. Im Play Store und im App Store gibt es Apps, mit denen sich sich Daten von einem ThingSpeak-Channel auf dem Smartphone anzeigen lassen. Auf meinem Android-Smartphone nutze ich beispielsweise dafür die App „*ThingView Free*“

Für die Nutzung von ThingSpeak wird ein Benutzerkonto benötigt. Unter dem Lizenztyp „Free“ habe ich ein kostenloses Benutzerkonto eröffnet. Mit einem Benutzerkonto dieses Lizenztyps lässt sich höchstens ein Channel erstellen. Auf diesem Channel werden dann die Daten in Form von Graphen dargestellt.

Nach dem Ersellen des Channels lassen sich API-Schlüssel generieren. Damit wird der Zugriff über Schnittstellen auf die Daten des Channels autorisiert. Für das Lesen und für das Aktualisieren der Daten gibt es zwei verschiedene Schlüssel. Beide lassen sich über die Einstellungen einsehen und neu generieren.

Bei dem erstellten Channel lässt sich zudem einstellen, wer in der Lage sein soll, den Channel einzusehen zu dürfen. Unter „Private View“ sind die Graphen nur für mich als Channel-Inhaber einsehbar. Unter „Public View“ können alle Channel-Besucher die Messungen einsehen. Der Channel ist seit dem 24.04.2017 unter „Public View“ einsehbar.

Die folgende Abbildung zeigt den Channel im privaten Sichtfeld unter Mozilla Firefox in Ubuntu Linux. Sie wurde durch drei Bildschirmfotos erstellt, welche zusammengesetzt alle Graphen darstellen.**Anmerkung:** Seit der Aufnahme der Abbildung (Mitte April) hat sich die Beschreibung oben in der Abbildung geändert. Der Link für den ThingSpeak-Channel im öffentlichen Sichtfeld ist hier unten gegeben: URL: <https://thingspeak.com/channels/233145/>

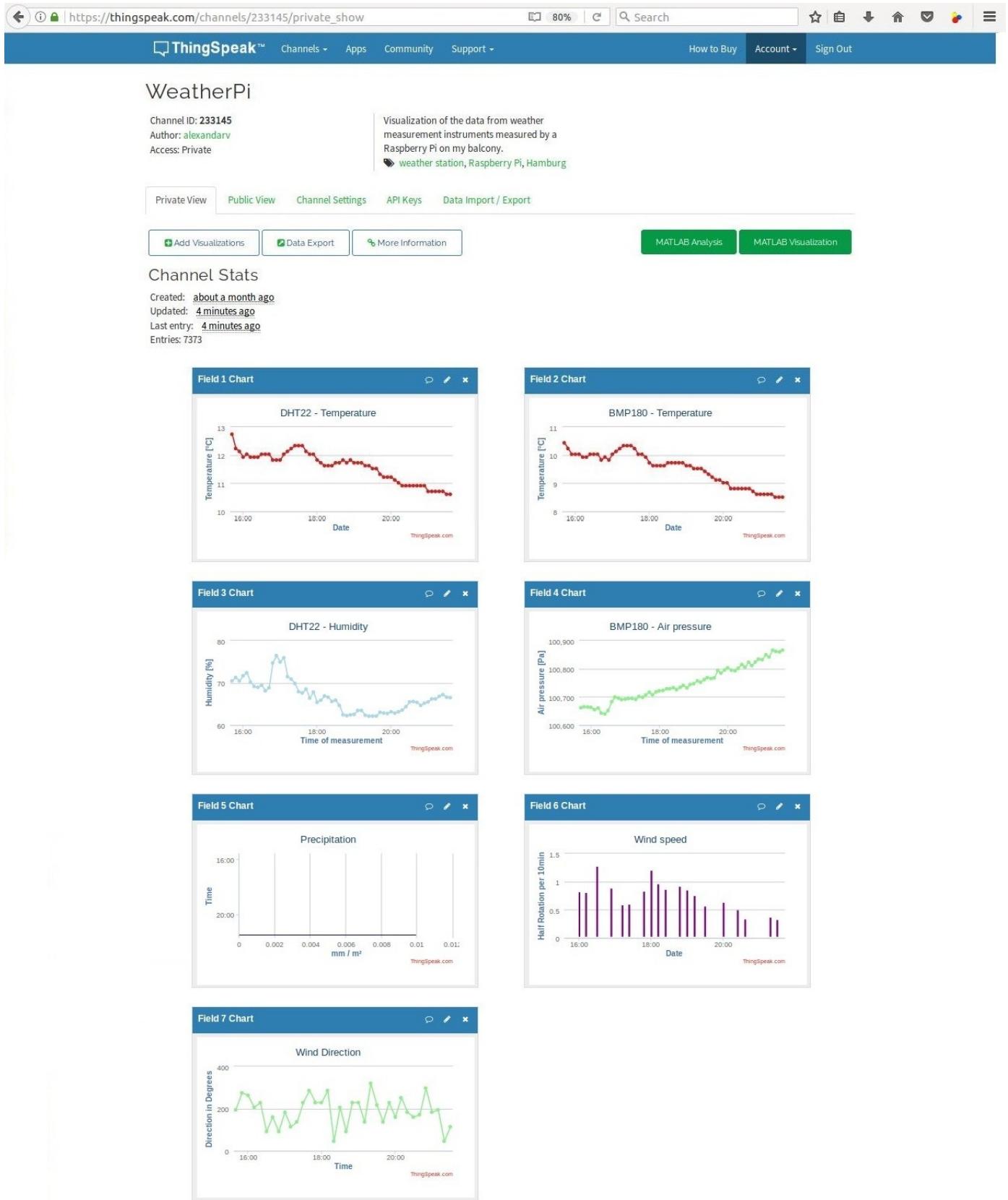


Abbildung 3.12: Bildschirmfoto des Channels auf ThingSpeak

### 3.3.2 Schnittstelle für ThingSpeak in Python

Anschließend an das Erstellen eines Channels bei ThingSpeak sollen nun auch Messwerte vom Raspberry Pi an den erstelleten Channel gesandt werden. ThingSpeak bietet auf der Webseite von MathWorks eine Anleitung, wie über JSON oder XML Daten eingefügt werden können.

Link: <https://de.mathworks.com/help/thingspeak/update-a-channel.html>

Im Falle des Raspberry Pi war die Dokumentation wenig hilfreich. Zwar wird beschrieben, wie ein Datenpaket zu strukturieren ist, damit es von ThingSpeak aufgenommen und gespeichert wird, jedoch ist kein Verweis für die Programmiersprachen Python oder PHP zu finden, wie über diese Programmiersprachen Daten einzufügen sind. Python gilt als die Standardsprache beim Raspberry Pi. Zudem sind die verwendeten Skripte zum Auslesen der Sensoren in Python programmiert. Eine Schnittstelle über Python wäre also äußerst sinnvoll, um nicht zwischen verschiedenen Programmiersprachen wechseln zu müssen.

Um eine Schnittstelle in Python zu erstellen, habe ich die API von Mahesh Vekitachalam auf <http://electronut.in/dht11-rpi-cloud-plot/> als Grundlage genutzt, diese modifiziert und auf das meinige Projekt angepasst. Die Funktionsweise des Skriptes von M. Vekitachalam ist einfach: Zuerst werden die Messungen über die Bibliothek von Adafruit ausgelesen, dann wird über der Schlüssel als Parameter (im Skript: sys.argv[1]) für die Nutzung der API gespeichert und abschließend eine Internetadresse aufgerufen Diese Internetadresse ist hierbei von Interesse, denn mit dem Aufruf dieser Adresse werden die Messungen auf den ThingSpeak-Channel übertragen. Als Beispiel sei folgender Link gegeben:

[https://api.thingspeak.com/update?api\\_key=AAAAAAAAAAAAA&temp=17&hum=67&pressure=1017](https://api.thingspeak.com/update?api_key=AAAAAAAAAAAAA&temp=17&hum=67&pressure=1017)

Über diese URL werden vier Werte in Form einer GET-Anfrage versandt. Um nur die Temperatur auszulesen, folgender Abschnitt betrachtet: &temp = 17&. Es ist zu erkennen, dass hier eine Variable mit einem zugewiesenen Wert im Link vorhanden ist. Das &-Zeichen dient dazu, einzelne Variablen voneinander zu trennen. Nach diesem Zeichen fängt die Deklaration einer neuen Variable an.

Ich habe das Skript von M. Vekitachalam so verändert, dass meine Messungen als Parameter eingesetzt werden. Der API-Schlüssel ist im Skript gespeichert. Der Link, welcher am Ende aufgerufen wird, wurde um die meinigen Messwerte erweitert. Damit lassen sich aktuell die Messungen in Python auf den ThingSpeak-Channel übertragen.

## 3.4 Berechnung meteorologischer Größen

Es ist mehr in den Messungen vorhanden als zu auf den ersten Blick zu erkennen ist. Doch um die Messungen interpretieren zu können, werden physikalische Modelle benötigt. Im Folgenden soll nun der Anteil an Wasser in der Luft berechnet werden, woraus die Wasserdampfdichte bestimmt wird. (Hinweis: Für Abkürzungen siehe Abkürzungsverzeichnis im Appendix)

### 3.4.1 Das Modell der idealen Gase

Die Umgebungsluft besteht aus einer Mischung verschiedener Gase. Jedes Gas, welches sich in einem Raum befindet, übt einen Druck aus. Der Luftdruck ist also eine Summe aus verschiedenen Partialdrücken, die von verschiedenen Gasen ausgeübt werden. Der Standarddruck bei Normalnull beträgt 101325 Pascal ( $1\text{Pa} = 1\text{N/m}^2$ ), das entspricht etwa einem Bar. Nach Dalton lässt sich der gemessene Luftdruck in Partialdrücke aufteilen. Dadurch lässt sich der Anteil des Wasserdampfdrucks in der Luft von der restlichen, sogenannten trockenen Luft herausrechnen:

$$\text{Luftdruck} = \text{PartialdruckWasserdampf} + \text{PartialdrucktrockeneLuft} \quad (3.14)$$

$$p_{\text{gesamt}} = p_w + p_L \quad (3.15)$$

In der Meteorologie gibt es eine Standardmodell für trockene Luft, welche dessen Bestandteile beschreibt. Da es nicht möglich ist, die Zusammensetzung und Volumenanteile der Gase der trockenen Luft auf dem Balkon zu bestimmen, müssen diese modelliert werden. Für die Rechnungen in den nächsten Unterkapiteln spielt die trockene Luft keine Rolle, es wird dann nur der Partialdruck des Wasserdampfes betrachtet.

Für die folgenden Berechnungen wird jedoch angenommen, dass sich die Gase in der Luft wie ideale Gase verhalten. Demnach lassen sich Gasteilchen als winzige Massepunkte im Raum beschreiben, welche keine Kräfte ausüben (Ausnahme: Zusammenstöße von mehreren Gasteilchen).

Es wird also ein idealer Fall betrachtet, wo Wechselwirkungen wie die Van-der-Waals-Kräfte zwischen den Gasen nicht betrachtet werden. Dadurch wird zwar die Rechnung vereinfacht, jedoch entstehen dadurch kleine Ungenauigkeiten. Zu dem Modell der idealen Gase gibt es entsprechend die Gleichung der idealen Gase.

$$p \cdot V = n \cdot R \cdot T \quad (3.16)$$

### 3.4.2 Sättigungsdampfdruck bestimmen

In der Umgebungsluft befindet sich Wasserdampf. Vielviel Wasserdampf vorhanden ist, hängt nicht nur von der Menge des Wasserdampfs, sondern auch von der Temperatur ab. Wenn die Temperatur gering ist, so kann nicht viel Wasserdampf in der Luft gehalten werden. Je höher die Temperatur wird, desto exponentiell mehr Wasser kann die Luft aufnehmen.

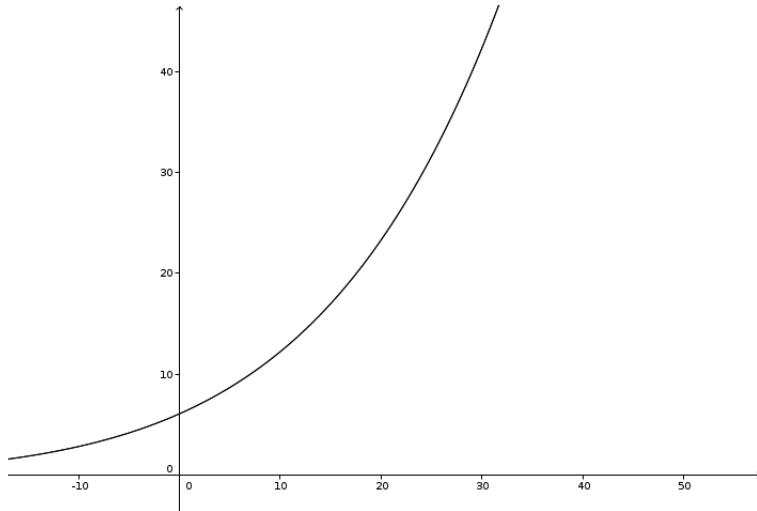


Abbildung 3.13: Magnus-Formel. Abbildung erstellt in Geogebra  
 $y$  – Achse: Sättigungsdampfdruck in hPa,  $x$  – Achse: Temperatur in ° C

Um diese Beziehung zu beschreiben, gibt es eine empirische Formel, welche den maximalen Gehalt des Wasserdampfes in der Luft unter der Abhängigkeit der Temperatur beschreibt.

$$e_s = 6.112 \cdot \exp\left(\frac{17.62 \cdot \vartheta}{243.12 + \vartheta}\right) \quad (3.17)$$

$$-45^\circ C \leq \vartheta \leq 60^\circ C \quad (3.18)$$

(Magnus-Formel mit verbesserten Werten nach Sonntag [5])

Das Ergebnis der Rechnung wird als Sättigungsdampfdruck bezeichnet. Es beschreibt den maximalen Partialdruck des Wasserdampfes, welcher bei der jeweiligen Temperatur herrscht. Dadurch, dass eine bestimmte Menge eines Gases in der Luft einen bestimmten Druck ausübt, kann die Menge des Stoffes proportional zu dessen ausgeübtem Druck angegeben werden.

Zur Illustration wird eine Metapher erstellt. Bildlich ausgedrückt ist die Luft ein Glas, welches Wasser aufnehmen kann. Je größer die Temperatur ist, desto größer ist das Glas. Der Sättigungsdampfdruck gibt dann die Höhe des Glases an. Wenn das Glas vollständig mit Wasser gefüllt ist, so entspricht die Höhe des Wasserstandes dem Sättigungsdampfdruck. Ein überfülltes Glas quillt über.

Damit wissen wir, wieviel Wasser die Luft maximal bei einer bestimmten Temperatur aufnehmen kann. Wieviel Wasserdampf ist aber konkret in der Umgebungsluft vorhanden?

### 3.4.3 Absolute Luftfeuchtigkeit bestimmen

Neben der gemessenen Umgebungstemperatur wird auch die relative Luftfeuchtigkeit durch das Hygrometer gemessen, welche in Prozent angegeben wird. Die relative Luftfeuchtigkeit beschreibt ein bestimmtes Verhältnis. Dieses Verhältnis kann auch als Sättigung bezeichnet werden. Ist das Glas voll mit Wasser, so liegt die relative Luftfeuchtigkeit bei 100%. Ist es halb voll mit Wasser, so liegt die relative Luftfeuchtigkeit bei 50%

Daraus lässt sich ableiten, dass die relative Luftfeuchtigkeit von dem Sättigungsdampfdruck und damit auch der Umgebungstemperatur abhängt. Es ist der Quotient aus dem Partialdruck des Wassers und des Sättigungsdampfdruck. Anders ausgedrückt: Es beschreibt das Verhältnis von der Höhe des Wasserstandes im Glas zur Höhe des Glases. Als Gleichung:

$$rf = 100 \cdot \frac{p_w}{p_s} \quad (3.19)$$

Da die relative Luftfeuchtigkeit und der Sättigungsdampfdruck bekannt sind, lässt sich die Gleichung nach dem Partialdruck des Wasserdampfes umformen:

$$p_w = 0.01 \cdot rf \cdot p_s \quad (3.20)$$

Mit dieser Gleichung wissen wir nun, wie voll das Glas ist. Um nun die absolute Menge an Wasserdampf in der Luft zu bestimmen, werden die Werte in die ideale Gasgleichung eingesetzt, welche noch zuvor umgeformt wird. R ist eine Konstante, T ist die Temperatur in Kelvin.

$$n \cdot R \cdot T = p \cdot V \quad (3.21)$$

$$n \cdot R_w \cdot T = p_w \cdot V \quad (3.22)$$

$$m \cdot R \cdot T = p_w \cdot V \quad (3.23)$$

$$\frac{m}{V} = \frac{p_w}{R \cdot T} \quad (3.24)$$

$$aF = \frac{p_w}{R \cdot T} \quad (3.25)$$

$$(3.26)$$

Mit dem Ergebnis der Rechnung solle nun achtsam umgegangen werden. Sollte an einer beliebigen Stelle um die Wetterstation der Wasserdampf der exakte Wasserdampfgehalt bekannt sein, so kann der Wert vom berechneten Ergebnis zum Teil stark abweichen. Bei dem Ergebnis handelt sich nämlich um eine Größe aus einem physikalischen Modell. Darin wird angenommen, dass die Gasteilchen gleichmäßig im Raum verteilt sind. In der Realität kann das berechnete Ergebnis durch verschiedene Umwelteinflüsse an manchen Stellen deutlich abweichen. Der Begriff *absolute Luftfeuchte* kann daher Verrirrung stiften. Alternativ kann die dahingehend genauere, äquivalente Bezeichnung *Wasserdampfdichte* für dasselbe Ergebnis verwendet werden.

### 3.4.4 Taupunkttemperatur bestimmen

Mit der Magnus-Formel lässt sich mehr berechnen als nur der Sättigungsdampfdruck. Man denke nun wieder an die erwähnte Metapher: Wir betrachten erneut ein Glas, welches diesmal bei 20 Grad Celsius halb voll mit Wasser gefüllt ist. Wenn die Temperatur kleiner wird, so nimmt die Höhe des Glases ab. Nun wollen wir wissen: Wann ist das Glas so klein, dass das Glas vollständig mit Wasser gefüllt ist?

Der Sättigungsdampfdruck entspricht dann dem Partialdruck des Wasserdampfes. Die relative Luftfeuchtigkeit beträgt 100%, da das Glas vollständig mit Wasser gefüllt ist. Die Temperatur, bei der diese Drücke gleich groß sind, wird als Taupunkttemperatur bezeichnet. Es gilt:

$$p_w = e_s \quad (3.27)$$

Nun lässt sich die Magnus-Formel erneut aufstellen. Diesmal wird sie jedoch nach Tau umgeformt:

$$a \cdot \exp\left(\frac{b \cdot \tau}{c + \tau}\right) = e_s \quad (3.28)$$

$$\exp\left(\frac{b \cdot \tau}{c + \tau}\right) = \frac{e_s}{a} \quad (3.29)$$

$$\frac{b \cdot \tau}{c + \tau} = \ln \frac{e_s}{a} \quad (3.30)$$

$$b \cdot \tau = (c + \tau) \cdot \ln \frac{e_s}{a} \quad (3.31)$$

$$b \cdot \tau = \tau \cdot \ln \frac{e_s}{a} + c \cdot \ln \frac{e_s}{a} \quad (3.32)$$

$$b \cdot \tau - \tau \cdot \ln \frac{e_s}{a} = c \cdot \ln \frac{e_s}{a} \quad (3.33)$$

$$\tau \cdot (b - \ln \frac{e_s}{a}) = c \cdot \ln \frac{e_s}{a} \quad (3.34)$$

$$\tau = \frac{c \cdot \ln \frac{e_s}{a}}{b - \ln \frac{e_s}{a}} \quad (3.35)$$

Für a und b werden die Konstanten der Magnusformel eingesetzt, für  $e_s$  der Partialdruck des Wasserdampfes (Gleichung 3.27). Damit lässt sich der Taupunkt bestimmen. Was lässt sich nun mit ihm anfangen? Zum einen lässt sich der Grad der Schwüle damit angeben. Dazu werden Temperaturbereiche einem Schwülegrad zugeordnet. Befindet sich die Taupunkttemperatur in gewissen Intervallen, so lassen sich Rückschlüsse darüber ziehen, als wie angenehm die Umgebungstemperatur wahrgenommen wird.

Eine weitere Anwendung der Taupunktformel sei noch gegeben. Über die Sprung'sche Faustformel lässt sich näherungsweise bestimmen, wie hoch sich eine Schönwetterwolke (Cu) über dem Erdboden befindet.

$$h_{Cumulus} = 123 \cdot (\vartheta - \tau) \quad (3.36)$$

### 3.4.5 Feuchttemperatur bestimmen

Wann schneit es? In der Meteorologie gibt es dazu eine Faustformel: Liegt die Feuchttemperatur zwischen 0 und 2 Grad Celsius, so kann der Niederschlag als Schnee fallen. Liegt sie darüber, so fällt der Niederschlag als Regen. Dadurch lässt sich Schneefall nicht genau prognostizieren, es ist vielmehr ein Richtwert. (Quelle: [1])

Die Feuchttemperatur (auch Feuchtkugelgrenztemperatur) gibt die Temperatur der Luft an, auf welche die Umgebungsluft abgekühlt werden kann, ohne dass Wasserdampf kondensiert. Bestimmt wird sie über eine numerische Formel. Da diese nicht nach der Feuchttemperatur lösbar ist, weil sie auf beiden Seiten der Gleichung vorhanden ist und auf der einen Seite als Parameter eingesetzt wird, ist eine Näherungsformel nach J. B. Rohegger gegeben.

Näherungsformel nach J. B. Rohegger [4]:

$$t_f = -5.806 + 0.672 \cdot \vartheta^2 + (0.061 + 0.004 \cdot \vartheta + 0.000099 \cdot \vartheta^2) \cdot \varphi + (-0.000033 - 0.000005 \cdot \vartheta - 0.0000001 \cdot \vartheta^2) \cdot \varphi \quad (3.37)$$

# Kapitel 4

## WebPi - Webserver einrichten

### 4.1 Raspberry Pi als Server einrichten

#### 4.1.1 Notwendige Software installieren

Für die Installation des MySQL-Servers werden folgende Befehle im Terminal ausgeführt:

```
sudo apt-get update && sudo apt-get upgrade -y  
sudo apt-get install mysql-server python-mysqldb -y  
sudo apt-get install mysql-client php5-mysql
```

Für die Installation vom Webserver Apache werden folgende Befehle im Terminal ausgeführt:

```
sudo apt-get update && sudo apt-get upgrade -y  
sudo apt-get install apache2 -y
```

#### Apache: Webserver konfigurieren

Die Installation von Apache macht es möglich, dass Anfragen aus dem Internet angenommen werden können, indem ein Ordner verfügbar gemacht wird, auf welchen über das Internet zugegriffen werden kann. Dieser Ordner befindet sich im Verzeichnis /var/www. Per Standardeinstellung nimmt Apache Anfragen auf dem Port 80 an. Dieser Port gilt als der Standard für jegliche Webseitenanfragen.

Um die Installation zu überprüfen, lässt sich die über die lokale IP-Adresse eine Testseite öffnen. Diese erscheint nach Eingabe von der statischen IP im Browser, in meinem Fall `http://192.168.178.22`. (vgl. Unterkapitel *statische IP-Adresse*)

## MySQL: Datenbank für Messwerte einrichten

Im ersten Schritt soll ein neuer Benutzer PiWrite erstellt werden. Er wird berechtigt sein, neue Daten in die Datenbanken einzufügen, neue Datenbanken und Tabellen zu erstellen und die Daten in den Tabellen zu lesen. Grund für das Erstellen von PiWrite ist, dass dadurch nur die benötigten Rechte an den Benutzer vergeben werden, welcher die Datenbank benutzt. Neben PiWrite existiert nämlich bereits der Superuser root. Dieser besitzt alle Rechte für alle Datenbanken, wodurch er auch in der Lage ist, Datenbanken zu löschen. Indem PiWrite statt root die MySQL-Befehle ausführt, wird die Möglichkeit entnommen, Messungen zu löschen.

```
CREATE USER 'PiWrite'@'192.168.178.22' IDENTIFIED BY '[password]';  
GRANT CREATE,SELECT,INSERT ON *.* TO 'PiWrite'@'192.168.178.22' identified by '[password]';  
FLUSH PRIVILEGES;
```

Die Struktur einer Datenbank kann man sich folgendermaßen vorstellen:

Die Datenbank ist ein Schrank. Im Schrank sind beliebig viele Ordner vorhanden. Diese Ordner heißen auch Tabellen. In jedem Ordner gibt es zudem ein Inhaltsverzeichnis.

So ähnlich verhält es sich mit einer MySQL-Datenbank, bloß dass die Datenbank wird in diesem Fall durch das Programm MySQL verwaltet wird. MySQL ist also der Schrank- und Ordnerlieferant. Um eine neue Datenbank in MySQL zu erstellen, muss also ein Schrank aufgestellt werden. Für die Messungen im Jahr 2017 werden nun zwölf Datenbanken für jeden Monat erstellt.

(Die eckigen Klammern sollen zeigen, wo Variablen sind. Unten wird nur jeweils **einer** der Monatsnamen eingesetzt)

```
CREATE DATABASE 2017_[January, February, ..., December];
```

Nun stehen die 12 Schränke. Damit man nun mit diesen Schränken weiterarbeiten kann, brauchen sie ein Ordner mit einem Inhaltsverzeichnis. Das Inhaltsverzeichnis ist vergleichbar mit einem Deckblatt, welches darüber bestimmt, was auf den nächsten Blättern im Ordner angegeben wird. Die folgenden Blätter sind leere MySQL-Tabellen, die genau so strukturiert sind wie das Inhaltsverzeichnis es vorgibt. Diese müssen vor den Messungen vorhanden sein, denn sonst gäbe es keine Seite, wo die Messungen ordentlich aufgeschrieben werden können.

Insgesamt gibt es sieben Messwerte, eine einmalige Nummer (ID) und den Zeitpunkt der Messung. Im Inhaltsverzeichnis wird nun festgelegt, dass es in den leeren Tabellen neun Spalten gibt (eine Spalte für jeden der Messwerte). Zudem wird festgelegt, was und wie lang die einzugebenen Werte sind. Jede leere Tabelle wird nach dem Tag und dem Monat benannt, in welchem die Messungen stattfanden. Jeder Ordner wird mit dem Monat und dem Jahr der Messungen beschriftet.

Für die metaphorische Ebene reicht dieses Bild zum Beschreiben der nun zu erstellenden Datenbank. Auf der technischen Ebene ist auf der nächsten Seite der genaue Befehl abgebildet, welcher alles Genannte präzisiert. Die ID ist der Primärschlüssel der Tabellen, der Messzeitpunkt kann aber auch als Primärschlüssel verwendet werden.

```

CREATE TABLE [Name der Tabelle] (
    id INT(6) UNSIGNED AUTO_INCREMENT PRIMARY KEY,
    temp_dht VARCHAR(10),
    temp_bmp VARCHAR(10),
    humidity VARCHAR(10) NOT NULL,
    pressure VARCHAR(10) NOT NULL,
    windspeed VARCHAR(5),
    precipitation VARCHAR(5),
    wind_direction VARCHAR(50) NOT NULL,
    time TIMESTAMP
)

```

Zur Illustration nehmen wir an, dass wir einen Schrank für den Monat Januar hätten. Nun sollen für den 1. Januar zwei Messungen in den Ordner eingetragen werden. Die Tabelle für den 1. Januar sähe wie folgt aus:

<b>id</b>	<b>temp_dht</b>	<b>temp_bmp</b>	<b>humidity</b>	<b>pressure</b>	<b>windspeed</b>	<b>precipitation</b>	<b>wind_direction</b>
114	7.4	5.3	77.3	102707.00	0.0	0	0,3,1,0,0,3,2,1
115	7.3	5.2	77.2	102697.00	0.0	0	0,2,2,0,1,4,1,0

Tabelle 4.1: Neue Einträge in der Tabelle der Datenbank

In den dickeren Buchstaben ist die jeweilige Spalte zu sehen, die im Inhaltsverzeichnis erstellt wurde. Darunter sind die Messungen zu sehen. Bei der Windrichtung handelt es sich um eine Liste. Für jede der Windrichtungen wird notiert, wie oft sie im Messintervall gemessen wurde. Aus Platzgründen sind hier nur acht Werte für acht Windrichtungen zu sehen. Tatsächlich gibt es jedoch 16 Werte für 16 Windrichtungen. Aus Platzgründen ist der Zeitpunkt der Messung nicht abgebildet.

Die Namen der Spalten für die Temperatur kommen daher, dass sowohl der BMP180 als auch der DHT22 die Temperatur messen. Verlässlich sind jedoch nur die Werte des DHT22. Um Kompatibilitätsfehler zu vermeiden, wird der Unterstrich als Ersatzzeichen für das Leerzeichen verwendet.

Eine Schwierigkeit ergibt sich bei dem Eintragen der Daten: Drei der Messwerte werden als Logdatei gespeichert, die anderen werden direkt nach dem Ausführen der Python-Skripte ausgegeben. Daher werden über Bash sowohl die Ausgabe der Python-Dateien als auch die Werte der Logdateien als Variable ausgelesen und gespeichert. Dann verbindet sich Bash mit der Datenbank:

```

mysql --user="PiWrite" --password="[geheimes Passwort]" --host "192.168.178.22" << EOF

INSERT INTO [Name der Tabelle] (
    temp_dht, temp_bmp, humidity, pressure, windspeed, precipitation, wind_direction
) VALUES (
    '[Temperatur]', '[Temperatur]', '[Luftfeuchte]', '[Luftdruck]',
    '[Windgeschwindigkeit]', '[Niederschlag]', '[Windrichtung]'
);
EOF

```

Damit sind nun die Messungen in der Datenbank gespeichert. Um auf diese erneut zuzugreifen, liest der Raspberry Pi B diese Daten über PHP aus. In dem folgenden Beispiel wird eine Anfrage an den MySQL-Server über PHP gesandt.

```
$query = "SELECT * FROM January_01";  
  
mysqli.connect($datenbank, $benutzer, $passwort, $host);  
$result = mysqli.query($query);
```

Zusätzlich besteht die Möglichkeit, auf eine erstellte Datenbank nicht nur lokal, sondern über das Netzwerk von einem anderen Computer aus zuzugreifen. Standardgemäß nimmt der MySQL-Server nur interne Anfragen an, um Fremdzugriffe zu verhindern. Dies muss jedoch geändert werden, weil die der Raspberry Pi Zero über das lokale Netzwerk auf die Datenbank zugreifen muss, um Messungen speichern zu können.

```
sudo nano /etc/mysql/my.cnf
```

Eintrag ändern:

```
# vorher: #  
bind-address = localhost  
  
# nachher: #  
bind-address = 192.168.178.22
```

Damit die Änderungen in Kraft treten, muss der MySQL-Server neugestartet werden.

```
sudo service mysql restart
```

### **Statische IP-Adresse des Raspberry Pi festlegen**

Zunächst soll folgende Unterscheidung getroffen werden: Wird ein Gerät aus dem Internet oder im lokalen Netzwerk aufgerufen? Im lokalen Netzwerk werden IP-Adressen vergeben, welche mit 192.168. beginnen. Dadurch wird sichtbar, dass es sich um eine lokale IP-Adresse handelt. Damit aus dem Internet ein Gerät aufgerufen werden kann, ist eine globale IP-Adresse notwendig. Sie wird dem Gerät vom Provider zugewiesen. Sie ändert sich alle 24 Stunden.

Beiden Raspberry Pis sollen immer unter jeweils einer bestimmten IP-Adresse im Netzwerk erreichbar sein. Um im Netzwerk eine lokale IP für den Raspberry Pi festzulegen, ist eine kleine Einstellung notwendig.

```

sudo nano /etc/network/interfaces
-----
auto lo
iface lo inet loopback

iface eth0 inet static # dhcp zu static ändern
address 192.168.178.22 # Statische IP hier festgelegt für RPi B
gateway 192.168.178.1 # Adresse zum Router (lokal)
netmask 255.255.255.0 # Netzmaske

allow-hotplug wlan0
iface wlan0 inet manual
wpa-roam /etc/wpa_supplicant/wpa_supplicant.conf
iface default inet dhcp

pre-up iptables-restore < /etc/network/iptables
-----
```

### Portweiterleitung beim Router einrichten

Der Router dient als Verbindungsstelle zwischen Raspberry Pi und der Anbindung ans Internet. Als Heimrouter eingerichtet wird er nicht Webseitenanfragen annehmen, sondern schlichtweg blockieren. Daher ist es notwendig, einige „Türen“ (sogenannte „*Ports*“) am Router für den Raspberry Pi offen zu halten.

Anfragen an Webserver werden generell an den Port 80 (HTTP) oder den Port 443 (SSL) gesandt. Also habe ich für diese Ports Portweiterleitungen eingerichtet. Damit kann nun über die globale IP-Adresse auf den Raspberry Pi zugreifffen werden.

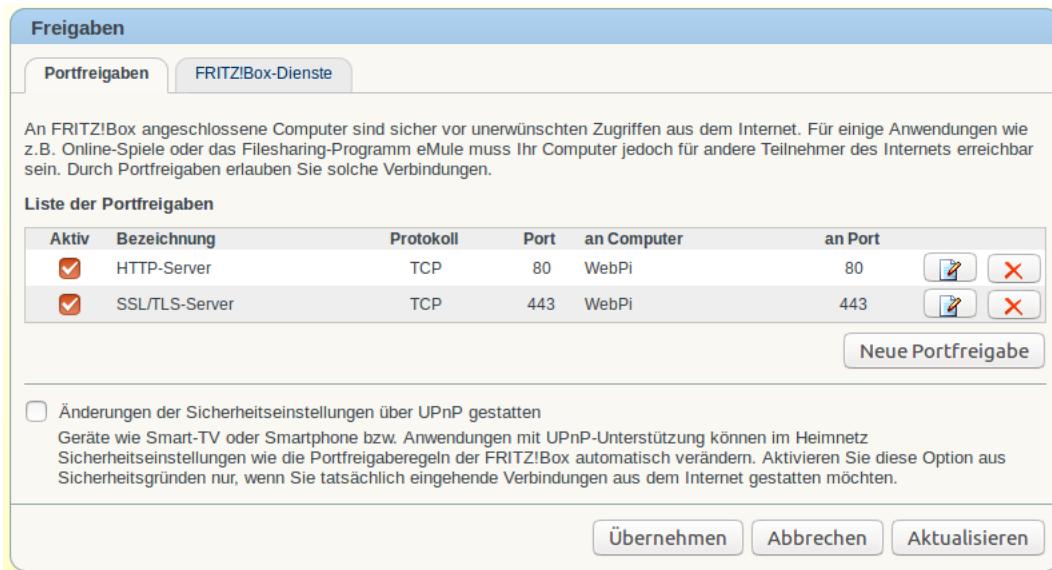


Abbildung 4.1: Portweiterleitung beim Router (FritzBox)

#### 4.1.2 Subdomain registrieren

Um eine Internetseite aus dem Internet aufzurufen, ohne die sich ständig ändernde IP-Adresse kennen zu müssen, wird eine Domain benötigt. Wenn im Browser eine Domain wie www.google.de eingetippt wird, so wird zuerst eine Anfrage an einen DNS-Server gesendet. Bei dem DNS-Server sind in wie in einer Datenbank die Domain und die IP-Adresse einander zugeordnet. Wird nach der IP von www.google.de angefragt, so erhält der Browser beispielsweise 74.134.81.225 als Antwort zurück. Nun kann der Computer sich mit dem Server verbinden, welcher die Webseite der Domain www.google.de gespeichert hat und sie an den Computer sendet.

Um eine Domain zu erhalten, ist im Regelfall eine Registrierung bei einem Hosting-Anbieter notwendig. Damit wird festgelegt, dass eine Domain einer bestimmten Person zuzuordnen ist. In meinem Fall habe ich bei dem Subdomain-Anbieter desec.io die Subdomain „mypi.dedyn.io“ eingerichtet.

Die Nutzung der Subdomain ist kostenfrei und bedurfte nur der Angabe der E-Mail-Adresse, jedoch ist die Wahl des Subdomainnamens etwas eingeschränkter. In dem gewählten Domainnamen muss „dedyn.io“ enthalten sein, da die Subdomain eine „Unterkategorie“ einer vorhanden Domain ist. Beispielsweise ist die Universität Hamburg als Hauptinstitut unter der Domain „www.uni-hamburg.de/“ verfügbar. Ein Institut als Bestandteil der Universität kann dadurch unter einer Subdomain erreicht werden. So lässt sich das meteorologische Institut unter „mi.uni-hamburg.de/“ finden, während der Fachbereich der Physik unter „http://www.physnet.uni-hamburg.de/“ zu finden ist.

#### IP-Adresse beim DNS-Dienst regelmäßig aktualisieren

Damit der DNS-Server nun den Browser an den richtigen Server weiterleiten kann, muss die Domain der aktuellen IP-Adresse beim DNS-Server zugeordnet werden sein. Die IP-Adresse des Routers ändert sich jedoch alle 24 Stunden, da es sich um eine IPv4-Adresse handelt. Daher muss die aktuelle IP-Adresse täglich an den DNS-Server gesandt werden, damit er Besucher auf der Subdomain an die richtige IP-Adresse weiterleitet. Dazu wird das Programmpaket *ddclient* benötigt.

```
sudo apt-get install ddclient
```

Für die manuelle Einrichtung vom ddclient werden in der Konfigurationsdatei den einzelnen Parametern der notwendige Wert zugeordnet. Hierbei wird für die Domain *mypi.dedyn.io* per SSL die aktuelle IP-Adresse an *update.dedyn.io* übermittelt. Sie wird über den Aufruf von *https://checkipv4.dedyn.io/* ermittelt.

```
-----
Datei: /etc/ddclient.conf
-----
use=web
protocol=dyndns2
use=cmd, cmd='curl https://checkipv4.dedyn.io/'
ssl=yes
server=update.dedyn.io
login=mypi.dedyn.io
password='[geheimes Passwort]',
mypi.dedyn.io
```

Durch die Ausführung von ddclient wird die IP bei desec.io aktualisiert. Trotzdem erscheint bei der Ausführung folgende Fehlermeldung:

```
Use of uninitialized value $_[2] in sprintf at /usr/sbin/ddclient line 1543.
```

Diese Fehlermeldung vom ddclient beeinträchtigt nicht die Aktualisierung der IP-Adresse. Es handelt sich um einen behebbaren Bug. Für diesen Bug hat bereits Erik Johansson in einem Blogeintrag auf <https://sourceforge.net/> aus dem Jahre 2009 eine Lösung gefunden. Er hat ein *Patch* erstellt, welches eine Lücke des Programmpakets behebt. Die Ausführung des Patches bewirkt, dass zwei Zeilen hinzugefügt werden. Diese Änderungen habe ich manuell mithilfe des Nano-Editors vorgenommen.

```
sudo nano /usr/sbin/ddclient
```

```
-----  
ab Zeile 2394 wiefolgt ändern:  
-----  
my $globalip = $ip; # Zeile hinzugefügt  
my ($status, $ip) = split / /, lc $line;  
$ip ||= $globalip; # Zeile hinzugefügt  
my $h = shift @hosts;
```

Nach zweimaligem Starten war die Fehlermeldung behoben.

Quelle: <https://sourceforge.net/p/ddclient/bugs/49/> - Erik Johansson - 2009-06-03 - ddclient.patch

#### 4.1.3 Virtual Host einrichten

Bei dem erstmaligen Öffnen der Webseite wurde die lokale IP-Adresse benutzt (Kapitel 4.1.1). Auch kann nun die globale IP-Adresse eingegeben werden, um die Webseite vom Webserver aufzurufen. Wenn über die Domain die Webseite aufgerufen wird, so erscheint aber die IP-Adresse in der Navigationsleiste. Statt der IP-Adresse soll aber die eben erstellte Subdomain bleiben. Ein Virtual Host macht dies möglich.

In dem Ordner „/etc/apache2/sites-available“ befinden sich ab Werk drei Dateien. Die Einstellungen werden für den jeweiligen Virtual Host vorgenommen.

```
sudo nano /etc/apache2/sites-available/[Dateiname].conf #
```

In meinem Fall wurden die Dateien des Virtual Host teilweise durch den Certbot (siehe Kapitel 4.3.3) bearbeitet. Daher kann ich aktuell die einzelnen Befehle nicht genau dokumentieren. Allgemein würde dann der Virtual Host nach dem Einfügen der jeweiligen Befehle durch Apache aktiviert werden. Nach Ausführen des Befehls erscheint im Ordner „/etc/apache2/sites-enabled“ eine Kopie der Datei. Danach kann die Webseite auch unter der Subdomain erreicht werden, ohne dass die IP-Adresse stattdessen zu sehen ist.

```
sudo a2ensite default-ssl
```

#### 4.1.4 Umleitung statt Fehlermeldung

Wie reagiert ein Server auf eine Anfrage, die er nicht erfolgreich ausführen kann? Als Beispiel sei gegeben, dass eine Datei verschoben wurde. Welche Rückmeldung erhält ein Besucher der Webseite, der diese Datei im ehemaligen Dateipfad anfragt? Die Rückmeldung in einem solchen Fall ist gut bekannt: „404 - Document not found“.

Wie also obig am Beispiel 404 dargestellt antworten Server mit sogenannten Statusmeldungen. Der Status ist jeweils einer dreistelligen Zahl (dem HTML-Statuscode) zugeordnet. Bei jeder Anfrage, die ein Apache-Server ausführt, wird diese in Logdateien finden. 200 steht beispielsweise für „OK“, die Anfrage wurde ordnungsgemäß ausgeführt.

Insgesamt mehr als 20 verschiedene Statuscodes. Vereinfachend lassen sich fünf Kategorien durch ihre erste Ziffer und die jeweilige Kernmitteilung bilden:

1xy - Information  
2xy - Operation erfolgreich  
3xy - Umleitung  
4xy - Client-Fehler  
5xy - Server-Fehler

Sollte eine Webseite angefragt werden, die nicht vorhanden ist, so kann der Besucher der Webseite auf eine andere Datei umgeleitet werden. Um dies in Apache2 einzurichten, werden die Dateien der Virtual Hosts mit den folgenden Zeilen ergänzt:

```
ErrorDocument 301 /index.php  
ErrorDocument 403 /index.php  
ErrorDocument 404 /index.php  
ErrorDocument 500 /index.php
```

Im einzelnen bedeuten die Fehlercodes:

Fehlercode	Definition	Bedeutung
403	„Forbidden“	Mangel an Rechten
404	„Not found“	Seite nicht vorhanden, „toter Link“
500	„Moved Permanently“	Datei verschoben

Sollte nun dazu kommen, dass eine Webseite angefragt wird, die nicht vorhanden ist, wo es dem Besucher Rechte zum Öffnen der Datei an Rechten mangelt, die Datei verschoben wurde oder es zu einem internen Fehler gekommen ist, so wird der Besucher auf die Startseite umgeleitet.

## 4.2 Wichtige Bestandteile der Webseite

In diesem Kapitel werden nicht die erstellten PHP-Dateien analysiert werden, welche das Gerüst der Webseite bilden. Dieses Kapitel thematisiert einige wesentliche Überlegungen, die in die Webseite eingeflochten wurden. Der Quelltext der einzelnen Dateien ist im Appendix zu finden. Kommentare sind als Erklärung angegeben.

### 4.2.1 Dynamische Webinhalte mit PHP erstellen

Als Programmiersprache wurde PHP verwendet. Sie zeichnet sich darum aus, dass es noch vor dem Versenden der Webseite (HTML, CSS) auf dem Server ausgeführt wird. Zudem ist die Verbindung zu MySQL-Datenbanken vergleichsweise leicht, MySQL-Befehle sind bereits als Funktionen in PHP vorhanden.

### 4.2.2 Webseite auf dem Endgerät mit *Media Queries* anpassen

HTML wird benutzt, um der Webseite eine Struktur zu geben. CSS wird benutzt, um der Webseite ein geeignetes Layout zu geben. Da heutzutage nicht nur neben Computer auch Mobiltelefone Webseiten anzeigen können, ist für verschiedene große Endgeräte ein unterschiedliches Layout zu wählen. Das Ziel moderner Webseitengestaltung (Responsive Design) ist es daher, die Webseite so anzulegen, dass sie auf dem Endgerät möglichst optimal angezeigt wird.

Daher habe ich zwei verschiedene Stylesheets erstellt. Die eine für Mobilgeräte gedacht, die andere für alle anderen Geräte mit einer Bildschirmgröße ab 1150 Pixel. So wird die Webseite bei Mobiltelefonen im einen Format angezeigt werden, während auf Heimrechnern mit großen Bildschirmen das dafür gedachte Layout angezeigt wird. Allerdings kann auch das Mobilformat auf Heimrechnern angezeigt werden, wenn dem Browser weniger als 1150 Pixel zur Verfügung stehen. Dies passiert beispielsweise dann, wenn zwei Fenster gleichzeitig auf dem Desktop angezeigt werden und beide etwa die Hälfte der Monitorbreite an Platz einnehmen.

### Das Menü mit jQuery

jQuery ist eine Bibliothek in JavaScript, welche das Verwenden von vorhandenen JavaScript-Funktionen erleichtert. Damit kann das Menü auf der Webseite bei Geräten mit einer kleinen Bildschirmauflösung per Klick angezeigt oder versteckt werden.

### 4.2.3 Visualisieren der Messdaten mit Chart.js

Chart.JS ist eine Bibliothek in JavaScript. Sie ist Open-Source und erstellt Grafiken als Canvas-Element in einem HTML-Dokument über JavaScript. Die Dokumentation auf <http://www.chartjs.org/docs/> veranschaulicht die Verwendung mit Beispielen. Da sich bei meiner Webseite die Daten in einer MySQL-Datenbank befinden, wird der Inhalt mit PHP über MySQL-Befehle ausgelesen, noch vor dem Versenden des HTML-Pakets eingesetzt und beim Endgerät durch JavaScript als Graph dargestellt. Als Visualisierungen werden Linien- und Radardiagramme verwendet.

#### 4.2.4 Impressum und Bildlizenzen

Bei der Webseite handelt es sich um eine private Webseite. Sie hat keine kommerziellen Ziele in Absicht. Ein Impressum ist also nicht notwendig, bei kommerziellen Webseiten ist sie Pflicht. Trotzdem ist ein Impressum für meine Webseite vorhanden. Bei Subdomains kann man nicht immer erkennen, wer diese erstellt hat, weil die eigentliche Domain einer anderen Person gehören kann. Damit wird also Klarheit für Besucher geschaffen, wer diese Webseite erstellt hat. Das Impressum wurde mithilfe des Impressumgenerators auf <https://www.e-recht24.de> erstellt und wurde von mir um einige Zeilen ergänzt.

Die verwendeten Bilder auf der Webseite sind entweder lizenziert oder eigene Werke. Das Logo mit der Wolke, welches auch das Deckblatt dieser Dokumentation ziert, stammt aus <https://pixabay.com> und ist gemeinfrei (CC0). Der Kopf im Footer ist ein eigens erstelles Werk. Das Menü-Icon, welches bei Geräten einer kleinen Bildschirmauflösung erscheint, stammt vom Nutzer *Slomox* auf Wikimedia und ist gemeinfrei.

### 4.3 Die Sicherheit des Servers erhöhen

#### 4.3.1 SSH-Schlüssel erstellen

Um die Sicherheit zu erhöhen, wird erstmal die Verbindung zum Server gesichert. Dazu soll ein Schlüsselpaar erstellt werden, mit denen eine verschlüsselte Kommunikation aufgebaut werden kann. Die Schlüssel werden auf dem Computer, welcher auf den Raspberry Pi zugreift, erstellt. Danach wird er öffentliche Schlüssel auf den Server übertragen. Während der öffentliche Schlüssel nur in der Lage ist, Text zu verschlüsseln, kann der private Schlüssel, welcher auf dem Computer bleibt, sowohl Nachrichten ent- als auch verschlüsseln.

```
ssh-keygen -b 4096 -t rsa
cat ~/.ssh/id_rsa.pub | ssh pi@192.168.178.22 "cat >> ~/.ssh/authorized_keys"
```



```
alex@X55A:~$ ssh-keygen -b 4096 -t rsa
Generating public/private rsa key pair.
Enter file in which to save the key (/home/alex/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/alex/.ssh/id_rsa.
Your public key has been saved in /home/alex/.ssh/id_rsa.pub.
The key fingerprint is:
SHA256:CxsqcpnnIn2RBvY0nMXfeWgbUJdp5nGUBzuavQ0hSY alex@X55A
The key's randomart image is:
+---[RSA 4096]---+
|   .   ..++.. |
|   o   o ..+* |
|   . o . + +E X . |
| ... = . = *= . |
| ...o.. o S *.. +
|   ++ . + .. + .. |
| ..o=.o .. o . |
| .o++ . |
| o .. |
+---[SHA256]---+
alex@X55A:~$
```

Abbildung 4.2: Bildschirmfoto: Erstellen der Schlüssel im Terminal

### 4.3.2 SSH-Daemon modifizieren

Nun wird der Hintergrundprozess (Daemon) beim Raspberry Pi bearbeitet. Mit folgenden Änderungen wird es für potentielle Angreifer äußerst schwer, überhaupt vom Raspberry Pi nicht blockiert zu werden.

Die Änderungen sorgen dafür, dass nur Verbindungen angenommen werden, die per öffentlichen Schlüssel authentifiziert sind. Diese sind die Computer, deren öffentlicher Schlüssel in der Datei 'authorized\_keys' vorhanden ist. Über das Passwort des Benutzers pi ist damit keine Anmeldung mehr möglich.

```
-----
Änderungen an der Datei: /etc/ssh/sshd_config
-----
PasswordAuthenticacion no          # Nicht über Passwort, sondern Schlüssel anmelden
ChallengeResponseAuthenticacion no # Nicht über Passwort, sondern Schlüssel anmelden
PermitRootLogin no                 # Nicht als Superuser anmelden lassen

RSAAuthentication yes              # Über RSA-Schlüssel anmelden
PubkeyAuthentication yes           # Über öffentlichen Schlüssel Identität bestätigen
AuthorizedKeysFile %h/.ssh/authorized_keys # Dateipfad zu anerkannten Identitäten

UseDNS no                         # Nicht an den Domainnamen anknüpfen
AddressFamily inet                 # Nur IPv4-Verbindungen annehmen
```

### 4.3.3 CA-Zertifikat mit Let's Encrypt erstellen

Secure Socket Layer (SSL) und Transfer Layer Security (TLS) sind weit verbreitete Protokolle, mit denen eine verschlüsselte Verbindung im Internet möglich wird. Die Verschlüsselung basiert auf dem RSA-Verfahren. Damit werden hybride Schlüssel erstellt, damit beide Teilnehmer miteinander kommunizieren können.

#### HTTPS: Vorgeschichte

Es gab große internationale Aufruhr als Edward Snowden geheime Dokumente leakte, aus welchen hervorging, dass die die NSA flächendeckend Daten von Menschen ausspäht und speichert. Geradezu unbekannt ist jedoch die Tatsache, dass immer noch viele der Daten, die über das Internet versandt werden, in Klartext gelesen werden können. Dies liegt daran, dass keine Verschlüsselung verwendet wird. HTTP ist als Grundkonstrukt von Webseiten nicht darauf ausgelegt, eine verschlüsselte Verbindung zu erzeugen, dafür gibt es andere Protokolle wie SSL.

Große Unternehmen wie Google stiegen auf die standardmäßige Nutzung sicherer Protokolle wie HTTPS um (Googlefragen konnten früher tatsächlich unverschlüsselt übertragen werden!). Viele anderer Webseiten im Internet haben diese Möglichkeit nicht. Die Server können keine SSL-Anfrage verarbeiten. Man könnte zwar RSA-Schlüssel erstellen und damit die Kommunikation verschlüsseln. Von gängigen Browsern wird aber der eigens erstellte als Sicherheitsrisiko angesehen, da die Zertifikate selber erstellt wurden. Es gibt keine dritte Stelle, die dem Browser bestätigt, dass eine Kommunikation mit dem Server und den Schlüsseln „sicher“ ist.

Das Problem ist hierbei also, dass für die Einrichtung einer „sicheren“ Kommunikation ein sogenanntes CA-Zertifikat notwendig ist. Diese Zertifikate werden Zertifizierungsstellen (Certificate Authorities, CA) an Webseitenbetreiber verkauft. Browser erkennen dann, dass eine seriöse Prüfstelle einer bestimmten Webseite ein CA-Zertifikat für die Domain vergeben hat. Daher wird die Kommunikation mit dem Server als „sicher“ eingestuft.

Bei Let's Encrypt handelt es sich um eine genau solche Zertifizierungsstelle, welche kostenlos CA-Zertifikate vergibt **und** dessen Zertifikate von aktuell gängigen Browsern als seriös angesehen werden. (vgl. <https://www.letsencrypt.org>)

### **Let's encrypt: CA-Zertifikate für den Webserver erstellen**

Um das Zertifikat von Let's Encrypt zu erhalten, gibt es das Programm Paket *Certbot*, welches den Prozess der Zertifizierung übernimmt. Er unternimmt einige Tests und speichert alle erstellten Dateien auf dem Computer. Mehr zur Funktionsweise von Certbot lässt sich unter folgender Internetseite nachlesen: <https://letsencrypt.org/how-it-works/>

Certbot ist jedoch nicht für Raspbian portiert worden. Da Raspbian aber eine Distribution von Debian ist und Debian offiziell unterstützt wird, läuft Certbot prinzipiell auch unter Raspbian. Im Folgenden wird die vorgenomene Installation ausschließlich für den Raspberry Pi beschrieben. Die Befehle sind aus verschiedenen Quellen entnommen, der Verweis jeweils angegeben.

Zuallererst soll der eigene SSL-Dienst von Apache ausgeschaltet werden. [Q1]

```
-----  
1: SSL-Dienst von Apache ausschalten  
-----  
sudo a2dissite default-ssl  
sudo apache2ctl graceful  
-----
```

Folgt man der offiziellen Dokumentation für die Installation von Certbot unter Debian Wheezy, so soll der Certbot heruntergeladen werden. Danach soll er als ausführbare Datei für das System kenntlich gemacht werden und danach ausgeführt werden. Die folgenden Befehle sind unten gegeben: [Q3]

**Anmerkung:** Certbot wurde hier unter Raspbian Wheezy installiert. Anfang April führte ich ein Upgrade von Raspbian Wheezy auf Raspbian Jessie durch. Daher ist unten die offizielle Dokumentation für Debian Wheezy zu sehen.

```
-----  
2: Offizielle Anleitung für die Installation  
-----  
wget https://dl.eff.org/certbot-auto  
chmod a+x certbot-auto  
./certbot-auto --apache  
-----
```

Hier kommt es zu einer Fehlermeldung. Der Raspberry Pi B kann den Certbot nicht ausführen. Das Problem liegt nicht am Certbot selber, sondern an Programmpaketen, die Certbot benötigt. Diese Programmpakete kann der Raspberry Pi B jedoch nicht ausführen. Dies liegt daran, dass der Raspberry Pi B einen ARMv6-Prozessor besitzt, während folgenden Modelle des Raspberry Pi einen ARMv7-Prozessor besitzen. Dies hat zur Folge, dass Programmpakete für die einzelnen Prozessoren anders kompiliert werden müssen, damit der Prozessor den Programmcode ausführen kann. Ein ARMv6-Prozessor kann keinen Programmcode ausführen, welcher für einen ARMv7-Prozessor kompiliert wurde.

### Installation des Certbot beim Raspberry Pi B

Den Blogeinträgen von Benny Wydooghe und Rajnish Bhaskar zufolge sollen diejenigen Programmpakete entfernt werden, welche nicht für ARMv6-Prozessoren kompiliert wurden. Für diese müsste ein Downgrade notwendig sein. Augeas 1.4 wird sodann vom Raspberry Pi selber für ARMv6 kompiliert.

```
-----  
3: Programmpakete entfernen, neu kompilieren # Quelle: [Q2], [Q1]  
-----  
sudo apt-get remove libaugeas0 augeas-tools libaugeas-dev  
sudo apt-get install libreadline6-dev  
sudo wget http://download.augeas.net/augeas-1.4.0.tar.gz  
sudo tar -zxf augeas-1.4.0.tar.gz  
cd augeas-1.4.0 \  
  
sudo CFLAGS='`-march=armv6 -mfpu=vfp -mfloat-abi=hard`' CPPFLAGS='`-march=armv6  
-mfpu=vfp -mffloat-abi=hard`' ./configure  
  
sudo CFLAGS='`-march=armv6 -mfpu=vfp -mffloat-abi=hard`' CPPFLAGS='`-march=armv6  
-mfpu=vfp -mffloat-abi=hard`' make  
  
sudo make install  
-----
```

Zum Schluss soll es helfen, die Logdatei vom Certbot mit den richtigen Rechten zu versehen.

```
-----  
3.1: Dateirechte ändern # Quelle: [Q1]  
-----  
sudo chmod 755 /var/log/letsencrypt
```

Mit der Ausführung dieser Befehle in der gegebenen Reihenfolge gelang es mir, den Certbot beim Raspberry Pi B einzurichten. Noch ist aber die Installation nicht vollständig. Zu achten ist aber darauf, dass **keine** Softwareaktualisierung des Certbots durchgeführt wird. Die Bibliothek wird dadurch unbrauchbar gemacht, eine Neuinstallation wäre dann notwendig. Der folgende Parameter darf also **nicht** ausgeführt werden. (Dieser Fall kam vor.)

```
-----  
3.2: NICHT beim Raspberry Pi B auszuführen!  
-----
```

```
./certbot-auto renew  
-----
```

```
4: Certbot starten  
-----
```

```
sudo ./certbot-auto -apache # [Q3]  
-----
```

Noch ist der Server über SSL nicht verfügbar. Folgende Fehlermeldung erscheint noch.

**SSL\_ERROR\_RX\_RECORD\_TOO\_LONG**

Das Problem wird durch Finetuning der Dateien der Virtual Host gelöst:

```
-----  
5: Finetuning der Datei /etc/apache2/sites-available/ssl.conf  
-----  
SSLEngine on  
ServerName mypi.dedyn.io  
SSLCertificateFile /etc/letsencrypt/live/mypi.dedyn.io/cert.pem  
SSLCertificateKeyFile /etc/letsencrypt/live/mypi.dedyn.io/privkey.pem  
SSLCertificateChainFile /etc/letsencrypt/live/mypi.dedyn.io/chain.pem  
-----
```

Nun müssen noch die richtigen Parameter eingesetzt werden. Ein RSA-Schlüssel mit der Bitlänge 4096 eines Apache-Webservers für die Domain mypi.dedyn.io kompiliert werden, ohne dass eine Ausgabe von jeglichem Text im Terminal und ohne dass Aktualisierung des Certbot erfolgt. Nach einiger Fehlersuche war folgender Befehl zielführend.

```
-----  
6: Certbot starten  
-----  
../certbot-auto certonly -n --apache --rsa-key-size 4096  
--quiet --no-self-upgrade -d mypi.dedyn.io  
-----
```

Certbot speichert die nun erstellten Dateien unter folgendem Dateipfad:

„/etc/letsencrypt/live/myraspi.dedyn.io/fullchain.pem“

Dort ist das CA-Zertifikat mit entsprechendem Schlüsselpaar gespeichert.

Nun lässt sich die Webseite unter <https://mypi.dedyn.io> öffnen.

#### Quellenverweis für Befehle:

[Q1] <https://lordofthemoon.com/blog/2016/08/lets-encrypt-for-the-raspberry-pi-model-b/>

[Q2] <https://i-net.be/techblog/?p=6>

[Q3] <https://certbot.eff.org/#debianwheezy-apache>

#### 4.3.4 Firewall einrichten

Als Firewall wurde fail2ban eingerichtet. Fail2ban analysiert die Fehlermeldungen aus den Logdateien von Apache. Auf deren Basis werden sogenannte Jails aktiviert, welche unerwünschte Besucher blockieren. Im Folgenden soll nur die Konfigurationsdatei des lokalen Jails aufgeführt werden. [Q4]

```
sudo apt-get install fail2ban

-----
/etc/fail2ban/jail.local
-----

# Apache: banning basic authentication failures

[apache]
enabled = true
port = http,https
filter = apache-auth
logpath = /var/log/apache*/error.log
maxretry = 6

# Apache: baning scripts searching for scripts on the website. Since I use PHP:

[apache-noscript]
enabled = false

# Apache: block clients attempting to request unusually long and suspicious URLs.

[apache-overflows]
enabled = true
port = http,https
filter = apache-overflows
logpath = /var/log/apache*/error.log
maxretry = 2

# Der folgende Jail wurde neu eingefügt:

[apache-badbots]
enabled = true
port = http,https
filter = apache-badbots
logpath = /var/log/apache*/error.log
maxretry = 2

# Do not use Apache to provide access to web content within home directories

[apache-nohome]
enabled = true
port = http,https
filter = apache-nohome
```

```

logpath  = /var/log/apache*/error.log
maxretry = 2

# PHP: blocks attempts to use certain PHP behavior for malicious purposes

[php-url-fopen]
enabled  = true
port     = http,https
filter   = php-url-fopen
logpath  = /var/log/apache*/access.log

```

Quelle: [Q4] <https://www.digitalocean.com/community/tutorials/how-to-protect-an-apache-server-with-fail2ban-on-ubuntu-14-04>

## 4.4 Systemsteuerung: Automatisierung, Aktualisierungen und Systemüberwachung

### 4.4.1 Softwareupdates durchführen

Für ein Betriebssystem ist es äußerst wichtig, aktuelle Programme zu verwenden. Wichtige Patches, Bugfixes, Verbesserungen und Erweiterungen werden damit auf dem Computer installiert. Die Aktualisierung der Programmpakete vom Raspberry Pi lässt sich mithilfe folgender drei Befehle einfach ausführen. Mit dem ersten wird nach aktueller Software gesucht, mit dem zweiten installiert. Mit dem dritten Befehl werden größere Updates durchgeführt.

```

sudo apt-get update
sudo apt-get upgrade -y
sudo apt-get dist-upgrade -y

```

### Logwatch als „Wachhund“ - Kurzfassung

Logwatch ist ein Intrusion Detection System. Man kann sich Logwatch wie einen Wachhund vorstellen. Ich als Administrator bin nicht täglich beim Raspberry Pi eingeloggt und weiß nicht immer darüber Bescheid, ob und wann jemand den Webserver angreiften hat.

Logwatch macht also folgendes: Er analysiert verschiedene System-Logdateien auf Ereignisse, wo Fehlermeldungen auftrafen, und sendet mir einen Bericht per E-Mail zu. Beispielsweise wertet er die Datei des Hintergrundprozesses vom Webserver (httpd) aus und zeigt, dass folgende Adresse versucht wurde aufzurufen:

```
//phpmyadmin/scripts/setup.php: 1 Time(s)
```

## **Cronjob: Automatisierung von Befehlen**

In den vorherigen Kapiteln wurden verschiedene Programmpakete installiert. Einige davon müssen regelmäßig ausgeführt werden, so beispielsweise das Aktualisieren des Betriebssystems. Damit diese Befehle automatisiert ausgeführt werden, lassen sich sogenannte Cronjobs erstellen. In der Cronjob-Datei wird eingetragen, wann welcher Befehl ausgeführt werden soll. Unten sind alle aktuellen Cronjobs dargestellt.

```
-----  
sudo crontab -e # Cronjob erstellen  
-----  
# Logwatch: Send daily email to report logs  
# @Raspberry Pi || Time: Every 24 hours  
0 18 * * * sudo /usr/sbin/logwatch --output mail --mailto [E-Mail]@[E-Mail].[E-Mail]  
--detail high --range 1 days --format html  
  
# Update system, then restart. Do not send a message  
# @Raspberry Pi || Time: every day at 5 am  
0 5 * * * sudo apt-get update && sudo apt-get upgrade -y && sudo shutdown -r 5 > /dev/null  
  
# Start firewall when RPi boots  
@reboot sudo fail2ban-client start  
  
# DynDNS via ddclient when RPi boots  
@reboot /usr/sbin/ddclient  
  
# Refresh SSL certificate monthly, restart apache  
@weekly /wpi/letsencrypt/certbot-auto certonly -n --apache --rsa-key-size 4096 --quiet  
--no-self-upgrade -d mypi.dedyn.io && sudo service apache2 restart  
  
# Create table/database for readings  
1 0 * * * sudo php /wpi/php/createDB.php  
-----
```

## Kapitel 5

# Ausblick auf Erweiterungen

Der jetzige Stand ist noch nicht endgültig. Weiterhin gibt es Erweiterungen, die noch bevorstehen, aber aus Zeitgründen noch nicht in Angriff genommen worden.

### **Graph erweitern: Monatsüberblick, Archiv**

Nach jetzigem Stand wird auf der Webseite ein Graph angezeigt, welcher die Messungen der letzten drei Stunden desselben Tages darstellt. Dies soll in Zukunft durch Buttons auf der Webseite erweitert werden können. Ein größerer Zeitraum soll dadurch ausgewählt werden können. Auch soll die Möglichkeit hinzugefügt werden, dass ältere Messungen angezeigt werden.

Derzeit muss auch für jeden neuen Graphen die gesamte Seite neu geladen werden. Dies wird mit AJAX in Zukunft vorraussichtlich geändert werden. Nur der Graph würde aktualisiert werden.

### **Wettertendenzen aufzeigen**

Durch den Verlauf des Luftdrucks lässt sich eine Wettertendenz ableiten. Diese ist nicht immer zutreffend. Jedoch reichen diese aus, um eine Schätzung der Wetterentwicklung für den folgenden Tag zu machen.

Anwendung und Idee dahinter: Die Wetterstation würde mich per E-Mail benachrichtigen, sollte es die Tendenz für eine gute Witterung zum Tischtennisspielen am folgenden Tag sprechen.

### **Lichtsensor hinzufügen**

Ich hatte bereits mit dem Lichtsensor A9013 den Lichteinfall innerhalb meines Zimmers messen können. Für die Ergänzung an der Wetterstation fehlt jedoch eine Kalibrierung und/oder eine Herleitung der Beziehung zwischen gemessenen Widerstand und dem Lichteinfall. Ohne Interpretation wären diese Messungen wertlos.

# Appendices

# Literaturverzeichnis

- [1] Rebekka Krampitz. Was ist die Feuchtkugeltemperatur? *Wetterkanal Kachelmannwetter*, 22.12.2015.
- [2] Simon Long. A security update for Raspbian PIXEL. *Raspberry Pi Foundation (Blogbeitrag)*, 30.11.2016.
- [3] Joanna Moorhead. Raspberry Pi device will 'reboot computing in schools'. *The Guardian*, 09.01.2012.
- [4] J. B. Rohregger. Methoden zur Bestimmung der Schneefallgrenze. Master's thesis, Universität Wien, 2008.
- [5] D. Sonntag. Important new Values of the Physical Constants of 1986, Vapour Pressure Formulations based on its-90, and Psychrometer Formulae. *Zeitschrift für Meteorologie. Band. 40, Nr. 5, 1990, S. 340–344. ISSN: 0084-5361*, 1946.
- [6] Eben Upton. Ten millionth Raspberry Pi, and a new kit. *Raspberry Pi Foundation (Blogbeitrag)*, 08.09.2016.

## Anhang A

# Verwendete Bibliotheken

Bibliothek	Autor(en)	Lizenz
BMP Library	Tony DiCoda, Adafruit	Copyright-Notiz
DHT Library	Tony DiCoda, Adafruit	Copyright-Notiz
MCP3008 Library	Tony DiCoda, Adafruit	Copyright-Notiz
ChartJS	Nick Downie	MIT-Lizenz
jQuery	jQuery Foundation	MIT-Lizenz
Bibliothek in EAGLE	Sparkfun Industries	CC SA-BY 4.0 International

## Anhang B

# Erstellte Skripte

Aus Gründen der Transparenz soll hier für jedes einzelne Skript die Autorenschaft und die Lizenz für die jeweilige Datei aufgelistet sein.

Dateipfad	Autor(en)	Lizenz
/wpi/getData.sh /wpi/thingspeak_API.py	Alexandar Vesović Mahesh Vekitachalam, Alexandar Vesović	eigenes Werk lizenzfrei
/wpi/lib/bmp.py /wpi/lib/dht.py	Tony DiCola Tony DiCola	Copyright-Notiz Copyright-Notiz
/wpi/run/raindrop.py /wpi/run/winddirection.py /wpi/run/windspeed.py	Alexandar Vesović Alexandar Vesović MagPi, Alexandar Vesović	eigenes Werk eigenes Werk lizenzfrei
/var/www/impressum.php /var/www/index.php	Alexandar Vesović Alexandar Vesović	eigenes Werk eigenes Werk
/var/www/css/main.css /var/www/css/desktop.css /var/www/css/mobile.css	Alexandar Vesović Alexandar Vesović Alexandar Vesović	eigenes Werk eigenes Werk eigenes Werk
/var/www/func/button.js /var/www/func/Chart.js /var/www/func/jQuery.js /var/www/func/head.php /var/www/func/header.php /var/www/func/menu.php /var/www/func/meta.php /var/www/func/footer.php	Alexandar Vesović Nick Downie jQuery Foundation, Weitere Alexandar Vesović Alexandar Vesović Alexandar Vesović Alexandar Vesović Alexandar Vesović	eigenes Werk MIT-Lizenz MIT-Lizenz, Copyright-Notiz eigenes Werk eigenes Werk eigenes Werk eigenes Werk eigenes Werk

Dateipfad	Autor(en)	Lizenz
/var/www/pic/cloud.png		gemeinfrei (CC0)
/var/www/pic/head.png	Alexandar Vesović	eigenes Werk
/var/www/pic/menu.png	Slomox, WikiMedia	gemeinfrei
/var/www/WeatherPi/index.php	Alexandar Vesović	eigenes Werk
/var/www/WeatherPi/meteocalc.php	Alexandar Vesović	eigenes Werk
/var/www/WeatherPi/mysql_graph.php	Alexandar Vesović	eigenes Werk
/var/www/WebPi/index.php	Alexandar Vesović	eigenes Werk
/wpi/php/createDB.php	Alexandar Vesović	eigenes Werk
/wpi/php/options.php	Alexandar Vesović	eigenes Werk
/wpi/php/lastline.php	Alexandar Vesović	eigenes Werk

#### Copyright-Notiz von Adafruit für die verwendeten Bibliotheken:

```

# Copyright (c) 2014 Adafruit Industries
# Author: Tony DiCola

# Permission is hereby granted, free of charge, to any person obtaining a copy
# of this software and associated documentation files (the "Software"), to deal
# in the Software without restriction, including without limitation the rights
# to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
# copies of the Software, and to permit persons to whom the Software is
# furnished to do so, subject to the following conditions:

# The above copyright notice and this permission notice shall be included in all
# copies or substantial portions of the Software.

# THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
# IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
# FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
# AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
# LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,
# OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE
# SOFTWARE.

```

## Anhang C

# Begriffsverzeichnis

Fachbegriff	Beschreibung
Anemoskop	Windrichtungsmesser
Anemometer	Windmesser
Pulviometer	Niederschlagsmesser
Terminal	Kommandozeile bei UNIX-basierten Systemen (Linux)
Jail	Regelwerk (Konfigurationsdatei) für fail2ban. Darin wird eingestellt, bei welcher Art Logeinträge welche Aktion („Banaction“) erfolgt.
Superuser	Administrator unter Linux-basierten Betriebssystemen
Stylesheet	Eine CSS-Datei, welche das Layout der Webseite beinhaltet
Open-Source	Programm, dessen Quelltext bearbeitet werden kann („quelloffen“).
Subdomain	Teil einer Domain.
Port	Eine „Eingangstür“ für Datenpakete zum Computer.
IPv4, IPv6	Bestimmten Ports ist eine Funktion zugewiesen
Canvas-Element	Zwei verschiedene Versionen der IP-Adresse. Beide werden aktuell unterstützt.
Patch	IPv4 ist aktuell Standard, soll jedoch in Zukunft durch IPv6 ersetzt werden
Nano-Editor	Ein HTML-Befehl, mit dem Formen angezeigt werden
Ubuntu	Codeschnippel, welcher einen Programmfehler behebt („Bugfix“)
Distribution	Ein Programm Paket. Damit lässt sich über dem Terminal eine Datei bearbeiten
UART	Linux-basiertes Betriebssystem. Eine Distribution von Debian.
	Linux-basiertes Betriebssystem. Eine Distribution von Debian.
	Betriebssystem, welches vom Distributor mit bestimmter Software und Linux-Kernel zusammengestellt wurde. Auch: Eine (bestimmte) Version eines Linux-Betriebssystems
	serielle Schnittstelle an PCs und Mikrocontrollern

Abkürzung	Fachbegriff	Beschreibung
API	Application Programming Interface	Schnittstelle für Programmierung
CA	Certificate Authority	Zertifizierungstelle, welche CA-Zertifikate vergibt
RSA	(Rivest-Shamir-Adleman)	moderner, weit verwendeter Verschlüsselungsalgorithmus
DNS	Domain Name Server	Server, der Domain und IP-Adresse speichert.
DoS-Angriff	Denial-of-Service-Angriff	Ein Angriff auf einen Computer (oft Server). Durch eine zu hohe Anzahl an zu Anfragen wird dieser überlastet.

## Anhang D

### Abkürzungsverzeichnis

Formelzeichen	Bezeichnung	Einheit / Wert
$\vartheta$	Umgebungstemperatur	Grad Celsius
T	Umgebungstemperatur	Kelvin
rf	relative Luftfeuchtigkeit	%
aF	absolute Luftfeuchtigkeit	$\frac{g}{m^3}$
p	Luftdruck	hPa
$\tau$	Taupunkttemperatur	Grad Celsius
$t_f$	Feuchttemperatur	Grad Celsius
$e_s$	Sättigungsdampfdruck	hPa
$p_L$	Partialdruck der trockenen Luft	hPa
$p_w$	Partialdruck des Wasserdampfes	hPa
$\rho$	Dichte	$\frac{g}{m^3}$
$h_{Cumulus}$	Höhe einer Schönwetterwolke (Cumulus)	m
$N_A$	Avogardo-Konstante	$N_A = 6.022142 \cdot 10^{23} \frac{1}{mol}$
$k_A$	Boltzmann-Konstante	$k_A = 1.38065 \cdot 10^{-23} \frac{J}{K}$
R	allgemeine Gaskonstante	$R = k_A * N_A = 8.314472 \frac{J}{K \cdot mol}$
$R_w$	spezifische Gaskonstante für Wasserdampf	$R_w = 461.4 \frac{J}{kg \cdot K}$

Tabelle D.1: Verwendete Formelzeichen

## Anhang E

# Liste aller Materialien

### Hardware:

Raspberry Pi:	<b>Modell B</b>	<b>Zero</b>
Netzteil:	5V, 2A	5V, 0.7A
Zusätzlich:	Gehäuse	Micro-USB-Adapter
Zusätzlich:	Kühlkörper	WLAN-Adapter

### Schaltung:

- Sensoren: DHT22, BMP180 und WH14C (vgl. Kapitel 3)
- Steckbrett und Steckbrücken für die Realisierung der Schaltung
- Metallwiderstände (1kOhm, 4.7kOhm), Kohlewiderstände (10kOhm)
- zwei RJ11-Adapter zum Verbinden mit Kabeln des WH14C
- abisolierte Kabel zum Verbinden mit RJ11-Adapter

### Löten:

- Lötkolben, Lötzinn, Brille und Unterlage zum Löten. Angelötet wurden:
  - > die Pins am Raspberry Pi Zero,
  - > die Pins am BMP180
  - > am P6-Header des Raspberry Pi B; (für Projekt unerheblich)

### Zum Anbringen des Raspberry Pi Zero auf dem Balkon:

- Eine mymuesli-Verpackung als Gehäuse
- Moosgummi in blau und schwarz für Gestaltung
- Heißkleber zum Aufkleben des Moosgummis
- Kabelbinder in verschiedenen Größen zum Befestigen der Kabel und des Gehäuses