Raspberry Pi Wetterstation

Von Leon Lepa, Oliver Winkler und Marvin Johanning
IT-Systemelektroniker der ISE9a des CSBME

April 2020

Zusammenfassung

Dieses Dokument dient zur Dokumentation des Aufbaus sowie der Programmierung einer Wetterstation, die mithilfe eines Raspberry Pis verwirklicht wurde. Zudem wird die Durchführung sowie die verwendeten Hard- und Softwarekomponenten beschrieben.

Inhaltsverzeichnis

1 Einleitung und Übersicht

Als Projekt wurde eine Wetterstation deshalb ausgewählt, da private Wetterstationen die Wetterverhältnisse an einem bestimmten Standort besser ermitteln können, als beispielsweise diejenigen Wetterstationen, die von verschiendensten Online-Diensten wie z. B. „Google“ oder „wetter.de“ genutzt werden.

Die Daten dieser Wetterstationen stammen oftmals aus einem anderen Ort, der gegebenenfalls sogar mehrere Kilometer weit entfernt liegt, wodurch die Genauigkeit dieser Daten oftmals schwankt. Ein bekanntes Problem ist beispielsweise die inkorrekte Anzeige von Regen an einem Standort, obwohl dort die Sonne scheint.

Der Besitz einer privaten Wetterstation jedoch macht es möglich, die genauen Wetterverhältnisse an einem exakt festgelegten Standort zu ermitteln. Die dadurch entstehenden Möglichkeiten sind vielfältig, wie beispielsweise die Errechnung der Median- oder Durchschnittstemperatur oder -luftfeuchtigkeit.

Ein weiterer Vorteil besteht darin, die Temperatur- und Luftfeuchtigkeitsdaten in einem geschlossenen Raum ermitteln zu können und bei Überschreiten eines Grenzwertes z. B. einen Alarm erschallen zu lassen. Dies könnte beispielsweise in Gewächshäusern oder auch Serverräumen zum Einsatz kommen und somit möglicherweise teuren Hardwaretausch — aufgrund von Überhitzung — entgegenkommen.

Die „Raspberry Pi Wetterstation“ bietet hierfür eine geeignete Grundlage; sie ermittelt die Wetterdaten anhand von Sensoren — dessen genaue Bezeichnungen und Funktionen im späteren Verlaufe dieser Dokumentation noch ausführlicher erklärt werden — und gibt diese auf einem LCD-Bildschirm aus. Zudem ist es möglich, die Wetterdaten anhand einer Webseite auszulesen, welche über die IP-Adresse des Raspberry Pis erreichbar ist. Weitere Funktionen lassen sich aufgrund des einfach zu programmierenden Raspberry Pi ohne großen Aufwand hinzufügen.

Im Folgenden werden nun die für die Verwirklichung des Projektes verwendeten Hard- und Softwarekomponenten genauer beschrieben; daraufhin wird die Durchführung und Planung des Projektes erläutert.

1.1 Verwendete Hardware

Im Folgenden Abschnitt werden die für das Projekt eingesetzten Hardwarekomponenten aufgelistet; zudem folgen Erläuterungen über deren Funktionalität sowie der Grund für das Einsetzen dieser Komponente in dem Projekt.

1.1.1 Raspberry Pi 3B+

Bei der Hauptkomponente des Projekts handelt es sich, wie der Name bereits vermuten lässt, um ein Raspberry Pi 3B+. Das Raspberry Pi ist ein sogenannter „Einplatinencomputer“ mit einer ARM-CPU. Somit ist die Installation von gängigeren Betriebssystemem, welche hauptsächlich nur unter x86-Technologie funktionieren, nicht möglich. Stattdessen bedient man sich einer Reihe von GNU/Linux-Distributionen, die für ARM-Prozessoren ausgelegt sind; so auch das von uns eingesetzte Betriebssystem „Raspbian 10 (buster)“. Bei „Raspbian“ handelt es sich um ein speziell für das Raspberry Pi entwickeltes Debian-Derivat welches auch von den Entwicklern des Raspberry Pis empfohlen wird.

Auch von den Computern selbst gibt es eine Vielzahl von unterschiedlichen Versionen von denen der Großteil für dieses Projekt in Frage kommt; die Entscheidung, die „3B+“ Variante einzusetzen stammt daher, dass uns diese bereits zur Verfügung stand. Bei der Auswahl eines geeigneten Raspberry Pi — oder auch eines anderen Einplatinencomputers — muss jedoch darauf geachtet werden, dass diese auch „GPIO-Pins“ (sogenannte *general-purpose input output pins*) besitzen; diese werden nämlich dazu eingesetzt, die Daten der einzelnen Sensoren an den Computer zu übermitteln.

Wir haben uns bewusst für diesen Einplatinencomputer entschieden — und nicht einen der vielen anderen — da es sich bei dem Raspberry Pi um den meistverbreiten Einplatinencomputer handelt; dadurch lassen sich eventuell auftretende Probleme der Soft- oder Hardwarekomponenten einfacher beheben, denn es existiert bereits eine große Online-Community mit vielen, äußerst hilfreichen, Benutzern.

1.1.2 MCP3008 Analog-Digital-Konverter (ADC)

Da die vorhin bereits kurz angesprochenen GPIO-Pins leider nur digitale Signale auslesen können, war es vonnöten, einen Analog-Digital-Konverter einzusetzen, welcher die analogen Signale des Regen- und Wassersensors in digitale umwandelt, mit welchen das Raspberry Pi umgehen kann.

Der *MCP3008* wurde ausgewählt, da er von vielen Leuten zur Verwendung mit einem Raspberry Pi empfohlen wurde und wir uns

somit sicher sein konnten, dass dieser auch funktioniert.

1.1.3 LM393 Regen-/Wassersensor

Bei einem LM393 handelt es sich einen sogenannten *Komparator*, welcher lediglich einen Spannungsunterschied misst; sobald ein Spannungsunterschied vorliegt, wird ein Signal ausgesendet, welches von dem Raspberry Pi ausgelesen werden kann.

Wir haben uns für den LM393 entschieden, da er perfekt für unsere Anwendung geeignet ist.

1.1.4 HD44780 LCD-Display

Um dem Benutzer Informationen zu der aktuellen Wetterlage anzuzeigen — was vor allem im Betrieb innerhalb eines Gebäudes geeignet ist — wurde auch ein LCD-Bildschirm an den Raspberry Pi angeschlossen. Hier haben wir uns für das einfache 2 x 16 Zeichen Display *HD44780* entschieden.

1.1.5 DHT22 Feuchtigkeits-/Temperatursensor

Die allerwichtigste Komponente einer Wetterstation ist der Temperatur- und Luftfeuchtigkeitssensor. Wir entschieden uns hier für den sehr weitverbreiteten *DHT22*-Sensor.

Dieser kann in einem Temperaturbereich von -40°C bis zu +80°C arbeiten und ist somit auch in Gebieten und Räumen mit extremen Temperaturbedingungen geeignet. Zudem ist der Sensor alle zwei Sekunden auslesbar und kann dadurch für sehr zeitnahe Temperatur- und Feuchtigkeitsmessungen eingesetzt werden.

1.2 Verwendete Software

Um das Projekt zu verwirklichen kommen verschiedenste Softwarekomponenten zum Einsatz; diese werden im Folgenden aufgelistet und deren Funktion kurz erläutert.

1.2.1 Ruby als Webserver und zum Auslesen des DHT22

Um eine einfache Webseite mit den Temperatur- und Luftfeuchtigkeitsverhältnissen zu erstellen, wurde die Programmiersprache *Ruby* unter Zuhilfenahme eines sogenannten *Gems* — also eines Pakets, dass die Funktionen der Programmiersprache erweitert — benutzt. Dieses Paket wurde mit der eingebauten Ruby *socket*-Funktion verbunden um einen HTTP-Server zu betreiben, der die aus dem Sensor ausgelesenen Daten auf einer Webseite präsentiert.

Dieser Gem, welcher den Namen *dht-sensor-ffi* trägt, stellt einfache Mittel dafür bereit, den digitalen DHT22-Sensor auszulesen.
So ist es möglich, mithilfe der `DhtSensor`-Klasse die Temperatur und Luftfeuchtigkeit auszulesen; es muss lediglich die `read()`-Methode benutzt und dieser mitgeteilt werden, an welchem Pin der Sensor angeschlossen ist und zudem auch welcher Sensor benutzt wird (es gibt unter anderem auch noch den DHT11).

Soll beispielsweise die Temperatur eines DHT22-Sensors ausgelesen werden, der mit Pin Nr. 4 des Raspberry Pis verbunden ist, so genügt folgende Codezeile: `DhtSensor.read(4,22).temperature`. Zudem muss der Gem auch in der Datei geladen werden; dies ist mithilfe der `require`-Funktion folgendermaßen möglich: `require 'dht-sensor-ffi'`.

Damit es der Programmiersprache jedoch überhaupt möglich ist, die einzelnen GPIO-Pins des Raspberry Pis auszulesen, ist eine in C geschrieben Bibliothek notwendig, welche Zugriff auf diese Pins gewährt; eine Kopie dieser Bibliothek ist auch im Projektordner im *lib*-Unterverzeichnis anzufinden. Weitere Informationen über die Bibliothek und dessen Installation sind dort anzutreffen.

1.2.2 Python zum Auslesen des LM393

Um den Status des LM393 auszulesen wurde Python verwendet; die `regen.py`-Datei liest den GPIO-Pin Nr. 13 aus und falls dessen Status *high* — also falls auf diesem 5 V anliegen — ist, so gibt die Datei „Es regnet“ aus; andernfalls wird „Es regnet nicht“ ausgegeben.

Diese Datei wurde mit in die Datei eingebunden, welche den Server betreibt, sodass auch auf der Webseite diese Regeninformationen angezeigt werden können.

1.2.3 Ruby zum Betreiben des LCD

Auch um Informationen auf dem LCD anzuzeigen wurde Ruby verwendet; auch hier war ein Gem namens *i2c-ss1602* vonnöten, der mit dem I^2C-Bus des Bildschirms kommunizieren kann. Ein Display kann mithilfe von `display = I2C::Drivers::SS1602::Display.new ('/dev/i2c-1', 0x27)` initialisiert werden und Text wird auf dem LCD unter Zuhilfenahme von der `text()`-Methode auf den Bildschirm geschrieben.

Dies wurde mit dem *dht-sensor-ffi*-Gem sowie mit dem Python-Script `regen.py` verknüpft, sodass der Bildschirm Zugriff auf die Wetterinformationen hat und dann in der Lage dazu ist, diese auszugeben.

1.2.4 VNC für Fernzugriff

Um auf das Raspberry Pi auch außerhalb des eigenen Netzwerkes zugreifen zu können wird VNC benutzt; dies hat den Vorteil, dass keine Netzwerkeinstellungen geändert werden müssen. Es wird bloß ein VNC-Account mit Zugriff auf die von uns angelegte Gruppe benötigt.

Nach einer erfolgreichen Verbindung muss lediglich der Browser geöffnet und `localhost` aufgerufen werden; die Wetterinformation werden dann angezeigt auf einer einfachen HTML-Seite angezeigt.

1.2.5 Cronjobs

Es wurden außerdem zwei sogenannte *Cronjobs* angelegt, welche die nötigen Scripts automatisch beim Starten des Raspberry Pi mitstarten; somit ist es nicht nötig, diese manuell zu starten.

Ein neuer Cronjob wird mithilfe des Kommandos `sudo crontab -e` erstellt — dieses öffnet einen Editor — und es wurden folgende Einträge erstellt (wobei `/pfad/zu/` mit dem tatsächlichen Pfad ersetzt werden muss): `@reboot sudo ruby /pfad/zu/display.rb` und `@reboot sudo ruby /pfad/zu/temp-server.rb`

2 Projektplanung

Wegen der, zu dem Zeitpunkt der Projektdurchführung laufenden, Kontaktsperre aufgrund von Sars-CoV-2 / COVID-19, musste die Projektplanung hauptsächlich dezentral geschehen. Es wurde daher auf diverse Online-Tools zurückgegriffen, um diese dezentrale Planung einfacher zu bewerkstelligen.

2.1 Eingesetzte Online-Tools

Im Folgenden werden die eingesetzten Tools aufgelistet und kurz beschrieben.

2.1.1 WhatsApp

Um Informationen zu sammeln und uns untereinander auszutauschen wurde der Instant-Messenger-Dienst „WhatsApp“ eingesetzt. Dort werden Dinge besprochen, für welche eine Videokonferenz nicht sinnvoll wäre, beispielsweise kurze Informationen oder interessante Links.

2.1.2 Jitsi

Videokonferenzen werden meist mithilfe von „Jitsi“ bewerkstelligt.

Der Vorteil von *Jitsi* anderen Videokonferenzdiensten gegenüber ist, dass es sich hierbei um eine Ende-zu-Ende-Verschlüsselung handelt; hierdurch wird das Abhören von Gesprächen erschwert und ist somit so gut wie unmöglich. Zudem ist das Erstellen eines Benutzerkontos nicht vonnöten, wodurch weniger persönliche Daten weitergegeben werden müssen. Ein weiterer Vorteil ist, dass es sich bei *Jitsi* um quelloffene (*open source*) Software handelt — also Software, dessen Quellcode eingesehen werden kann — und somit das Risiko einer „Backdoor“ sehr gering ist.

2.1.3 Git / GitLab

Für die Planung von Aufgaben und das Verwalten des Quellcodes der für die Wetterstation benötigten Programme sowie für die Dokumentation wird GitLab eingesetzt.

Bei Git handelt es sich um ein sogenanntes *Versionskontrollsystem*, das die Verwaltung und Kollaboration der Code-Base vereinfacht. Das gesamte Projekt kann als *Repository* auf einer Git-Seite — in unserem Falle GitLab — angelegt werden; dort werden die Daten dann zentral gespeichert. Zudem hat jede an dem Projekt arbeitende Partei eine lokale Kopie dieses Repositorys, an welchem Veränderungen getätigt

und dann auf das *Remote Repository* — also die auf einem fernen Server liegende Version des Repositorys — *gepusht*, also hochgeladen, werden können.

Die lokalen Veränderungen an der Code-Base werden von der jeweiligen Person in einem sogenannten *Commit* gespeichert und kurz beschrieben; dies macht es möglich, ohne Mühe auf eine ältere Version der Code-Base umzuschalten, falls sich die neuen Änderungen als schlecht ausgegeben haben.

2.2 Projektdurchführung

Da uns für die Projektdurchführung aufgrund der bereits beschriebenen Viruspandemie mehr Zeit verschafft wurde, war es uns möglich, die Arbeit an dem Projekt auf mehrere Tage zu verteilen.

Zuallererst wurde mit dem Schreiben der Dokumentation sowie mit dem Zusammenstellen des Raspberry Pi mit dem DHT22-Sensor begonnen. Dies ging ohne größere Probleme vonstatten.

Das Herausfinden von Informationen über das LCD-Display gestaltete sich allerdings schwieriger. Der Link zu einer Online-Anleitung, welche dem LCD-Display beilag, war leider nicht erreichbar; dies führte dazu, dass wir gezwungen waren, uns die nötigen Informationen anderweitig zu beschaffen.

Auch das Beschaffen von Informationen über den Regensensor gestaltete sich etwas schwierig; vor allem, da wir uns sehr unsicher waren, wie der Analog-Digital-Konverter mit dem Regensensor verknüpft werden muss. Nach weiterer Recherche wurden jedoch Informationen gefunden und es wurde klar, dass der Analog-Digital-Konverter nicht gebraucht wurde; dies liegt daran, dass der Regensensor bereits einen digitalen Ausgang zur Verfügung stellt. Somit war es möglich, den Regensensor direkt auszulesen, ohne die Signale in digitale umzuwandeln.

Nachdem sich herausstellte, dass das von uns gekaufte Display eine I^2C-Schnittstelle besitzt, wurde das Finden von relevanten Informationen deutlich einfacher und bald darauf wurde ein passender Ruby-Gem gefunden. Zuallererst musste jedoch der Pin-Header, der dem LCD beilag und der die I^2C-Schnittstelle ansprechen kann, angelötet werden.

3 Fazit

Die Projektdurchführung verlief nicht ganz ohne Probleme; diese konnten jedoch zügig erkannt und ausgebessert werden.

Allerdings stellt sich heraus, dass sich die Arbeit an einem Gruppenprojekt deutlich schwieriger gestaltet, wenn es nicht möglich ist, sich persönlich zu treffen; und auch wenn die beschriebenen Tools sehr dabei geholfen haben, so ist es doch ziemlich ungewohnt, damit zu arbeiten.

Jedoch ist uns nun klar, dass bei dem nächsten Projekt mehr Recherchen durchgeführt werden sollten, bevor Hardwarekomponenten gekauft werden; so war es beispielsweise unnötig, den Analog-Digital-Konverter zu kaufen, da der Regensensor bereits, anders als zuerst angenommen, einen digitalen Ausgang besitzt.

Unserer Meinung nach war dieses Projekt jedoch erfolgreich; es wurde eine funktionierende Wetterstation gebaut, welche auch aus dem Internet heraus erreichbar ist. Zudem wurden hilfreiche Dinge gelernt, welche die kommenden Projektdurchführungen stark erleichtern sollten.