BESONDERE LERNLEISTUNG

Automatisiertes Gewächshaus

Betreuende Lehrkräfte: Herr Friedrich Wilhelm Ernst & Herr Uwe Homm

Xuan Hung Truong

21. März 2019

Inhaltsverzeichnis

1 Einführung

Dieses Projekt beschreibt eine modellhafte Modifizierung von Gewächshäusern mit Regelungs- und Kommunikationstechnik.

Für ein optimales Wachstum von Pflanzen empfiehlt sich eine konstante Temperatur zwischen 20 bis 25 °C. Um diese dauerhaft konstant gewährleisten zu können, müssen Arbeitskräfte die Temperatur durchgehend manuell steuern, wodurch Lohnkosten aufkommen. Parameter wie Bodenfeuchtigkeit, Luftfeuchtigkeit und Sonnenlicht sind ebenfalls schwer zu kontrollieren.

Es stellt zudem eine große Schwierigkeit für normale Gewächshäuser dar, wenn sich Wetterverhältnisse plötzlich ändern. Nach einer Änderung muss eine Arbeitskraft manuell am Gewächshaus beispielsweise die Jalousien herablassen, wodurch Arbeitskosten wie Anfahrtszeiten aufkommen. Ebenso kann der Betreiber des Gewächshauses keine Änderungen vornehmen, wenn sich dieser nicht im näheren Umfeld des Gewächshauses aufhält.

Dieses Projekt zeigt einen möglichen Lösungsweg für die oben beschriebene Problematik.

Auf dem Markt gibt es bereits einige automatisierte Gewächshäuser. Es gibt automatisierte Container mit Pflanzen, autonome Gewächshäuser in der Stadt und sogar ein automatisiertes Gewächshaus in der Antarktis, welches als Forschungsprojekt für potenzielle Gewächshäuser im All dient. Im Internet findet man viele Anleitungen mit einem Arduino[1] oder aus einem Raspberry Pi[2] ein automatisiertes Gewächshaus zu bauen.

Der Unterschied zwischen den bereits existierenden automatisieren Gewächshäusern und dem in dieser Arbeit vorgestellten Modell ist, dass Letzteres über eine Applikation[3] gesteuert werden kann, in welcher man ein Pflanzenlexikon erstellt. Dadurch können in einem Gewächshaus mehrere Module gebaut werden, welche verschiedene Pflanzen beinhalten, die verschieden angesteuert werden. Die Soll-Werte für die Parameter können mittels dieser App auf einfache und effektive Art und Weise verändert werden.

[1] Open Source Soft-/Hardware, mit einem Micro Controller und analogen bzw. digitalen Ein- und Ausgängen
[2] Einplatinencomputer - https://www.raspberrypi.org/
[3] Anwendungssoftware

Eine Motivation für dieses Projekt war, das Problem eines Freundes zu lösen: aufgrund seiner langen Arbeitszeiten und häufigen Geschäftsreisen ist es ihm nicht möglich, einen Gemüsegarten anzubauen und diesen im geforderten Maß zu pflegen.

Bilder bzw. Grafiken ohne Quellenangaben sind selbst aufgenommen bzw. gezeichnet. Verwendete Programme werden im Quellenverzeichnis angegeben.

Alle Zeitangaben der Quellen geben den letzten Zeitpunkt des Zugriffs an.

1.1 Zielsetzung

Ziel dieses Projekts ist es, ein Gewächshaus-Modell zu entwickeln, welches vier Parameter regelt: Temperatur, Luftfeuchtigkeit, Bodenfeuchtigkeit und Lichtstärke. Dieses Gewächshaus soll über eine Applikation[3] steuerbar sein, welche dem Gewächshaus eine Pflanze zuordnet. Verschiedene Pflanzen besitzen zahlreiche unterschiedliche Informationen, wie zum Beispiel Soll-Werte bezüglich der optimalen Luftfeuchtigkeit, Temperatur etc. Diese Werte werden zunächst einmal für die jeweilige Pflanzenart definiert und können jederzeit überarbeitet oder gelöscht werden. Ebenso sollen eingelesene Parameter gespeichert werden, um eine graphische Darstellung der Parameter zu verwirklichen. Eine weitere Funktion soll die direkte Live-Steuerung werden, die es dem Nutzer ermöglicht, das Gewächshaus in Echtzeit zu steuern.

Die Kommunikation läuft dabei über ein lokales Netzwerk[4].

In dieser schriftlichen Ausarbeitung werden wichtige Hardware- und Softwarekomponenten abgehandelt. Viele dieser Komponenten habe ich erst durch die Arbeit an diesem Projekt kennengelernt.

[4] Local Area Network (LAN) - ein Netzwerk in Haushalten oder Unternehmen

2 Voraussetzungen für pflanzliches Wachstum

Für das Wachstum jeder Pflanze sind viele unterschiedliche Faktoren wichtig. Besonders Sonnenenergie, Kohlenstoffdioxid, Wasser und Wärme spielen eine wichtige Rolle für die Energiegewinnung der Pflanze.

Für das Wachstum der Pflanze wird Energie benötigt, welche sie aus der Photosynthese gewinnt. Die Photosynthese ist ein physiologischer Vorgang, in dem Licht zu chemischer Energie transformiert wird. Für diesen Prozess wird Kohlenstoffdioxid, Wasser, Wärme und Sonnenenergie benötigt. Die entstehende Wärme beim Energieverbrauch für das Pflanzenwachstum wird an die Umgebung abgegeben. Bei zu hoher Umgebungstemperatur kann die Pflanze die Wärme nicht abgeben und somit die Energie für ihr Wachstum nicht verbrauchen. Ebenso können Enzyme bei zu hoher Umgebungstemperatur zerstört werden und der Wasserverlust steigt. Bei zu geringer Umgebungstemperatur kann der Prozess der Photosynthese zum Erliegen kommen, wodurch in der Folge Energie zum Wachsen fehlt. Ebenso stoppt die Photosynthese, wenn zu wenig Wasser vorhanden ist.

Wasser wird auch als Lösungs- und Transportmittel für Nährstoffe benötigt und ist somit überlebenswichtig für die Pflanze. Wasser sorgt ebenfalls für die Stabilität der Pflanze, da es den Zellinnendruck reguliert.

Wie oben genannt ist Sonnenenergie ebenfalls notwendig für die Photosynthese. Diese Energie gelangt in Form von Sonnenstrahlen zur Pflanze, welche die energiereichen Teilchen der Sonnenstrahlen (Photonen) absorbiert. Für diesen Prozess benötigt die Pflanze den Farbstoff Chlorophyll, welcher bei einem zu großen Vorkommen von Sonnenenergie beschädigt wird. Um den Farbstoff Chlorophyll in den Blättern zu schützen, transpiriert die Pflanze. Transpiration ist die Verdunstung von Wasser über Spaltöffnungen in der Blattunterfläche. Durch die Transpiration entsteht ebenfalls ein Unterdruck, wodurch Mineralien in die Spitzen der Pflanze transportiert werden und die Pflanze stabil steht.

Der eigentliche Grund für Spaltöffnungen ist der Gasaustausch von Kohlenstoffdioxid und Sauerstoff mit der Luft, welcher essentiell für die Photosynthese ist. Dieser essentielle Gasaustausch kann jedoch nicht stattfinden, wenn die relative Luftfeuchtigkeit zu hoch ist. Folglich kann die Pflanze nicht transpirieren, wodurch die Spitzen der Pflanzen nicht versorgt werden und die Spaltöffnungen sich verschließen. Bei zu geringer Luftfeuchtigkeit wird die Pflanze schlechter vor Sonnenenergie geschützt, was zu Verbrennungen an den Blättern der Pflanze führt.

Zusammengefasst braucht die Pflanze Kohlenstoffdioxid, Wärme, Wasser und Sonnenenergie für die Energiegewinnung durch die Photosynthese. Weitere Faktoren für Pflanzenwachstum sind Luftfeuchtigkeit und Nährstoffe im Boden. All diese Faktoren haben Ober- und Untergrenzen, die bei jeder Pflanze verschieden sind. Bei Überschreitung dieser Grenzen wird die Energieproduktion gestört, was letztendlich zu Schäden an der Pflanze führt.

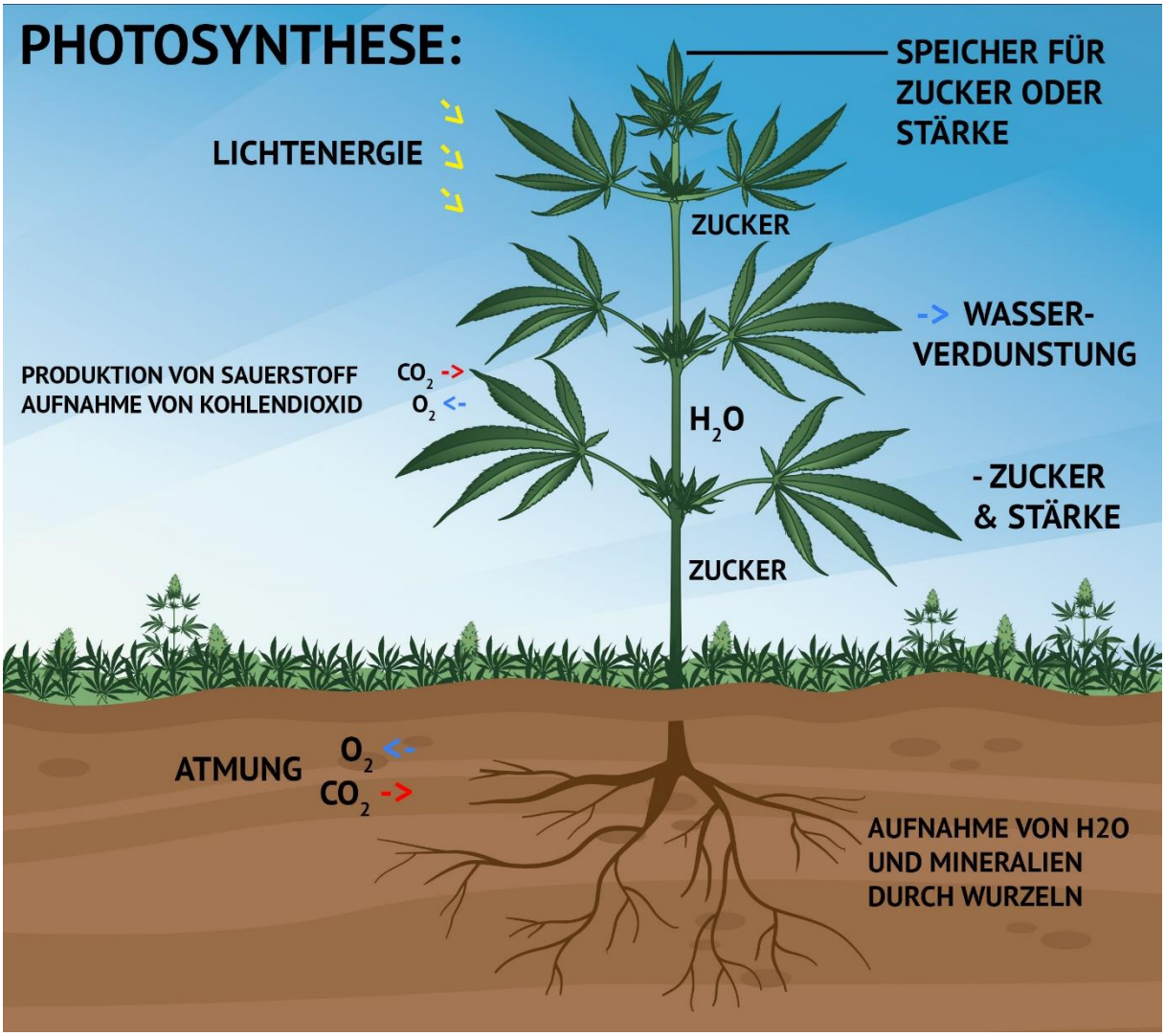


Abbildung 1: graphische Darstellung der Photosynthese[5]

[5] https://sensiseeds.com/de/blog/photosynthese-die-lichtphase/ (17:50, 18.02.19)

2.1 Projektbezug

Die im Rahmen dieses Projekts erarbeitete Technologie reguliert vier der oben genannten Faktoren.

Durch einen Photowiderstand wird die Helligkeit im Gewächshaus gemessen, um die benötigte Sonnenenergie zu regulieren. Bei zu starker oder zu geringer Helligkeit werden Jalousien bzw. ein UV-Licht aktiviert.

Um die Wasserversorgung der Pflanze zu regulieren, misst ein Bodenfeuchtigkeitssensor den Wassergehalt im Boden. Falls nötig, wird die Wasserzugabe dann reguliert.

Ein Temperatur-Luftfeuchtigkeitssensor übermittelt dem Regelungsgerät Werte bezüglich der Parameter Temperatur und Luftfeuchtigkeit, welche durch die Aktoren Luftbefeuchter, Heizung und Klimaanlage in den gewünschten Bereich navigiert werden.

3 Hardware

Die dem Projekt zugehörige Hardware umfasst ein Gewächshausmodell mit mehreren elektronischen Komponenten einem Datenbankserver und einem Smartphone mit Android-Betriebssystem.

Mithilfe des Datenbankservers werden die Daten des Systems verwaltet, welche durch das Smartphone mit der Gewächshaus-Applikation oder dem Mikrocontroller im Gewächshausmodell verändert werden können. Für diesen Datenaustausch ist ein WLAN-Netzwerk vorgesehen.

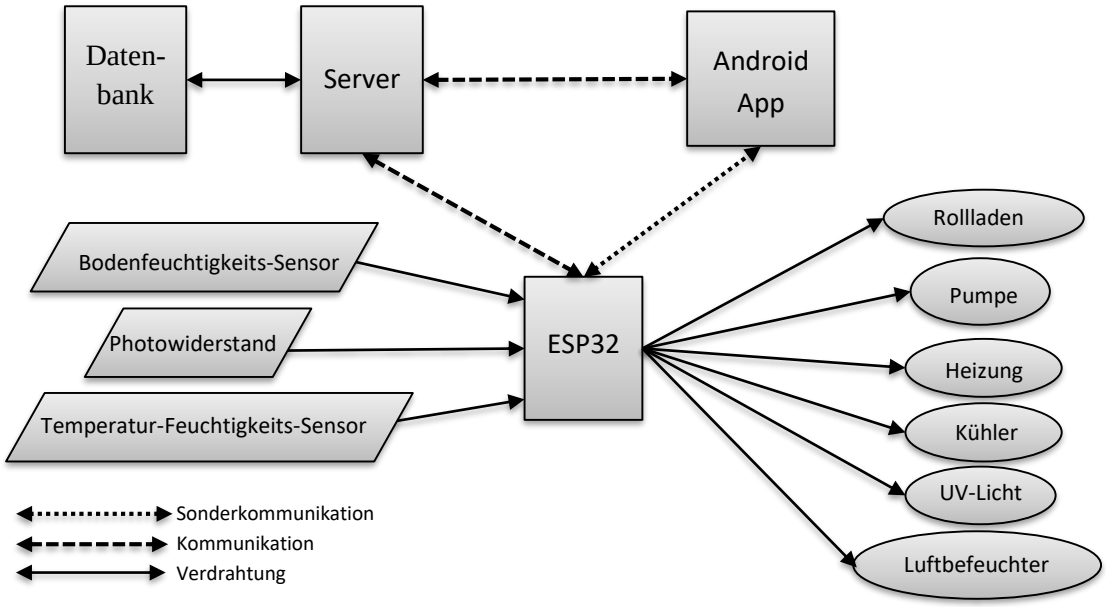


Abbildung 2: Schematische Darstellung des Gesamtsystems

Abbildung 3: Foto des Modells

3.1 Das Gewächshausmodell

Als Gewächshausmodell wurde das IKEA-Gewächshaus Socker[6] in weiß ausgewählt, da dieses preiswert, schnell zu besorgen und leicht zu bearbeiten ist. Die Maße des Gewächshauses betragen 45x22x35 cm und die Materialien sind Stahl, Pulverlack und Polystyrol.

Abbildung 4: Socker Gewächshaus[2]

Wie unter 2.2 beschrieben, gibt es Regelkreisläufe, um die vier Parameter Temperatur, Bodenfeuchtigkeit, Luftfeuchtigkeit und Lichtstärke zu kontrollieren. Dafür besitzt dieses Modell drei Sensoren, sechs Aktoren und den Mikrocontroller ESP32 als Regler.

3.1.1 Sensoren

Sensoren (lat. sentire z.dt. fühlen, empfinden) sind Bauteile, die chemische oder physikalische Eigenschaften oder stoffliche Beschaffenheit ihrer Umgebung erfassen können[7]. Diese werden in viele Untergruppen aufgeteilt, abhängig von gemessener Größe, Energiequelle, Signalart oder Änderungsart des elektrischen Stroms.

Ein Photowiderstand ist ein passiver, resistiver, lichtempfindlicher, analoger Sensor, welcher den elektrischen Widerstand mit der Lichtmenge ins Verhältnis setzt. Dieser Fotowiderstand ist im Funduino Starterpack vorhanden, weshalb ich mich für diesen Sensor entschieden habe.

Die Website Funduino[8] ermöglicht Einsteigern der Mikrocontroller-Programmierung einen guten Einstieg in das Thema. Die Website besitzt eine gute Dokumentation von Übungsprojekten und bietet viele verschiedene Packs an, welche zu den Dokumentationen passen. Diese Packs unterscheiden sich vom Umfang der elektronischen Komponenten und vom Mikrocontroller, welcher im Pack vorhanden ist. Die Website bietet vor allem Bildungseinrichtungen die Möglichkeit, kostengünstig den Schülern das Mikrocontroller-Programmierern beizubringen.

[6] https://www.ikea.com/de/de/catalog/products/70186603/ (16:17, 18.02.19)
[7] https://www.elektro4000.de/magazin/induktive-sensoren-aktive-sensoren-passive-sensoren/ (15:46, 27.01.19)
[8] https://funduino.de/ (16:22, 19.03.19)

Zudem beinhaltet das Projekt einen Verbundsensor, welcher die Parameter Temperatur und Luftfeuchtigkeit misst. Dieser DHT11 Sensor der Marke Adafruit ist ein passiver, digitaler Sensor, welcher mit einer Versorgungsspannung von 3 bis 5 V arbeitet. Der Sensor gibt Feuchtigkeitswerte im Bereich von 20 bis 80 % mit einer Abweichung von 5 % an und im Temperaturintervall 0 bis 50 °C mit 2 °C Abweichung. Ich habe mich für diesen Sensor entschieden, da dieser preiswert ist und direkt zusammen mit dem Mikrocontroller geliefert wurde. Ebenso ist eine Bibliothek und eine gute Dokumentation für den Sensor vorhanden, was für ihn spricht.[9]

Zuletzt wird noch der Bodenfeuchtigkeitssensor benannt, welcher ebenfalls im Funduino-Starterpack vorhanden war. Dieser Sensor besitzt zwei Kontakte, an denen Spannung anliegt. Abhängig vom Wassergehalt, leitet dieser den Strom besser bzw. schlechter zwischen den beiden Kontakten. Dies wird durch den Sensor und die Versorgungsspannung in ein analoges Signal umgewandelt, welches der Mikrocontroller deuten kann.

Abbildung 5: Fotowiderstand[10]

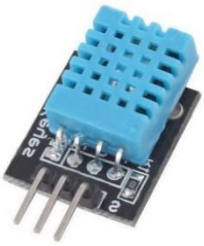


Abbildung 6: Temperature-Humidity-Sensor[11]

Abbildung 7: Ground Humidity Sensor[12]

3.1.2 Aktoren

Aktoren bewirken das Gegenteil von Sensoren. Sie setzen elektrische Signale in mechanische Bewegung oder in andere physikalische Größen um.

Da dies nur ein Modell ist, werden die Aktoren nur beispielsweise verdeutlicht!

Für die Änderung des Parameters Temperatur besitzt das Projekt einen 5-V-Grafikkartenlüfter zum Herabsetzen der Temperatur und eine LED als Symbol für die Heizung. Der Lüfter besitzt drei Pins: VCC, GROUND und Statusausgabe der derzeitigen Position.

Der Parameter Luftfeuchtigkeit hat nur den Luftbefeuchter als Aktor, welcher wie die Heizung durch eine LED gekennzeichnet wird.

Der Aktor Pumpe wird angesteuert, wenn der Parameter Bodenfeuchtigkeit unter dem Soll-Wert liegt. Ich habe mich für diese 5-V-Pumpe entschieden, da sie wie die anderen Sensoren preiswert ist und ich sie direkt mit dem Mikrocontroller zusammen bestellen konnte.

[9] https://cdn-learn.adafruit.com/down loads/pdf/dht.pdf (16:47, 16.02.19)
[10] https://articulo.mercadolibre.com.mx/MLM-558210202-sensor-luminosidad-ldr-photocell-gl5528-_JM (16:36, 25.02.19)
[11] https://www.mouser.com/ds/2/758/DHT11-Technical-Data-Sheet-Translated-Version-1143054.pdf (16:37, 25.02.19)
[12] https://www.miniinthebox.com/de/p/bodenfeuchtesensor-bodenfeuchtesensor-modul_p903362.html (16:37, 25.02.19)

Um den letzten Parameter Lichtstärke zu regeln, besitzt das Modell eine UV-Lampe, welche ebenso als LED vertreten wird und Rollläden, die über einen unipolaren Schrittmotor angesteuert werden. Dieser Schrittmotor war im Funduino-Starterpack vorhanden und wird deswegen von mir benutzt. Zudem überzeugte mich die geringe Versorgungsspannung von 5 V, die exakte Ansteuerung durch 2048 Einzelschritte und die einfache Ansteuerung durch die vorhandene Bibliothek. Der einzige Nachteil ist die langsame maximale Drehgeschwindigkeit des Schrittmotors.

Abbildung 8: Pumpe

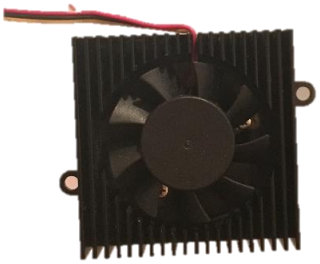


Abbildung 9: Kühler

Abbildung 10: Rolladen

3.1.2.1 Sperrschicht-Feldeffekttransistoren

Da die meisten Aktoren mit einer Versorgungsspannung von 5 V arbeiten und der Mikrocontroller ESP32 maximal 3,3V ausgibt, werden diese Aktoren mit Hilfe von unipolaren Transistoren angesteuert.

Transistoren sind allgemein Halbleiter-Bauteile, die als Schalter fungieren können. Diese können durch elektrische Signale gesteuert werden.[13] Die Gruppe der Feldeffekttransistoren ist eine der zwei wichtigsten Gruppen der Transistoren. Sie steuern den Stromfluß durch ein elektrisches Feld, welches durch die angelegte Spannung beeinflusst wird. Sie haben drei Pins: Drain, Gate und Source. Gate ist die Steuerelektrode, Source die Quelle des zusteuernden Stromflusses und Drain der Ausgang des Stromflusses, an den die Last kommt.[14] Sperrschicht-Feldeffekttransistoren (im Folgenden: JFET) sind besondere Feldeffekttransistoren, die eine weitere Sperrschicht besitzen. Sie werden in zwei Gruppen aufgeteilt, n-Kanal-Typ und p-Kanal-Typ. Der Unterschied der Gruppen ist die Dotierung der halbleitenden Kristallstrecke. Beim n-Kanal-Typ ist diese n-dotiert, was bedeutet das Elektronen der Halbleiterschicht hinzugefügt wurden, damit diese negativ geladen ist.[15] Die Steuerung funktioniert durch einen Feldeffekt, welcher durch die Spannung am Gate gesteuert wird, wie in Abbildung 11 zu sehen.

[13] https://www.grund-wissen.de/elektronik/bauteile/transistor.html (22:03, 16.02.19)
[14] https://www.elektronik-kompendium.de/sites/bau/0207011.htm (08:20, 17.02.19)
[15] https://www.elektronik-kompendium.de/sites/bau/1101211.htm (08:24, 17.02.19)

In diesem Projekt werden nur zwei JFETs mit dem n-Kanal-Typ benutzt, um den Lüfter und die Pumpe anzusteuern. Dafür wird an die Source-Pin eine Versorgungsspannung von 5 V angehängt und an das Gate der entsprechende Pin des Mikrocontrollers angeschlossen. Der Lüfter bzw. die Pumpe wird stattdessen an Drain angeschlossen.

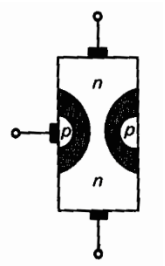


Abbildung 11: Feldeffekt[16]

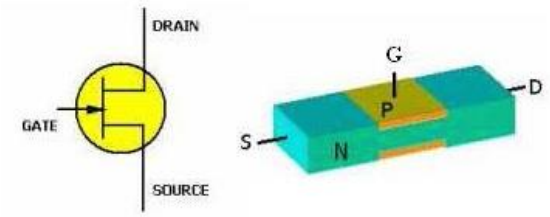


Abbildung 12: n-Kanal JFET[11]

3.1.3 Mikrocontroller ESP32

Der Mikrocontroller ESP32 ist der Nachfolger des ESP8266 und somit ein 3,3–V-Logikboard. Der ESP32 besitzt eine Micro-USB-Typ-B-Schnittstelle für die Stromversorgung und den Informationsaustausch mit dem Rechner. Zudem besitzt der Mikrocontroller einen ESP-WROOM-32-Prozessor, welcher eine WLAN- und Bluetooth Schnittstelle besitzt, eine 160-MHz-32-Bit-Dual-Core-CPU hat und auf dem 512-KB-SRAM wie 16-MB-Flashspeicher vorhanden sind. Es gibt 32 digitale I/O-Pins mit 3,3 V und 6 Analog-zu-Digital-Pins.

Besonders wird der Mikrocontroller im Bereich „Internet of Things“ angewendet, da dieser sowohl eine WLAN-, als auch eine Bluetooth-Schnittstelle aufweist und zudem sehr preiswert ist.

Aufgrund der WLAN-Schnittstelle, dem Preis und den verfügbaren Ports habe ich mich für diesen Mikrocontroller entschieden. Es standen ebenfalls der Arduino Mega 2560 und der ESP8266 zur Auswahl. Diese hatten entweder keine WLAN-Schnittstelle oder besaßen zu wenige Pins.

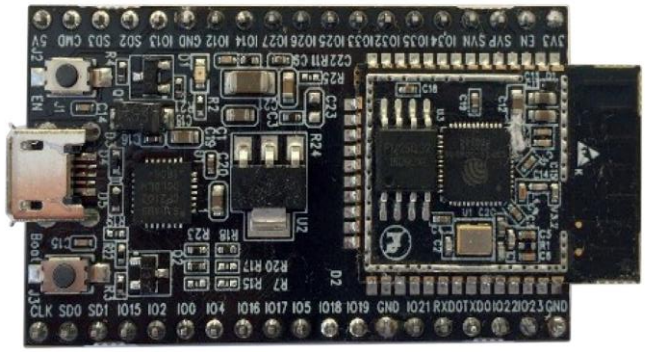


Abbildung 13: Mikrocontroller ESP32

[16] https://web.sonoma.edu/users/m/marivani/es231/units/experiment_07_2.shtml (16:10, 25.02.19)

3.1.4 Regelung

Eine Regelung liegt vor, wenn ein Soll-Intervall festgelegt wird, indem sich der Ausgangsparameter befinden soll und dieses durch den Regelkreis mit Rückkopplung eingestellt wird. Um dies zu gewährleisten, werden Sensor, Aktor und ein Regelgerät benötigt. Das benötigte Intervall wird in Form der Sollgröße dem Regelkreislauf vorgeschrieben. Diese Sollgröße wird mit der Rückführung, also dem eingelesenen Wert des Sensors, verglichen und die Information der Regelabweichung dem Regelgerät übermittelt. Dieses wertet die Information aus und erstellt eine Handlungsanweisung für den Aktor. Dieser führt die die Ausgangsgröße beeinflussende Aktion aus. Diese Größe ist wiederum die Eingangsgröße des Sensors und gleichzeitig die Ausgangsgröße des gesamten Regelkreises.

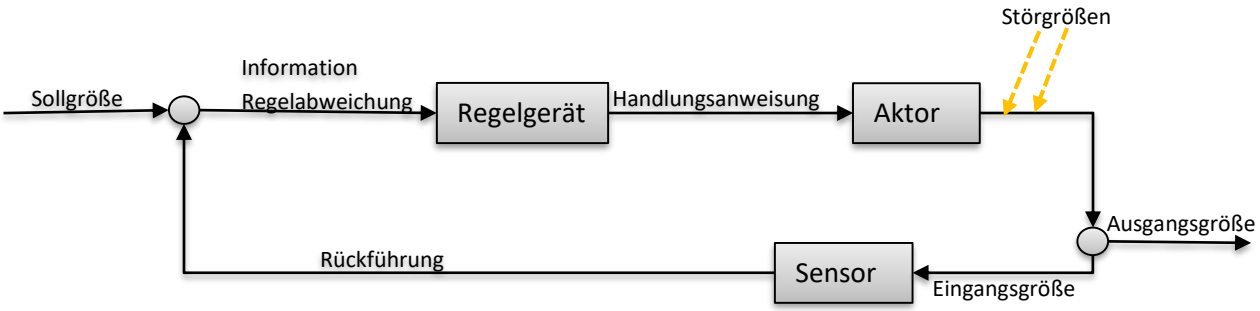


Abbildung 14: Allgemeiner Regelkreislauf

Um den Parameter Temperatur zu steuern, stehen zwei Aktoren und ein Sensor zur Verfügung. Der DHT11-Sensor, welcher Temperatur sowie die relative Luftfeuchtigkeit einließt, ist der Sensor des Regelkreislaufs. Dieser übermittelt die eingelesenen Werte an den ESP32, welcher Regelgerät und Recheneinheit darstellt, welcher wiederum Sollgrößen mit der Rückführung vergleicht. Bei Regelabweichung sendet der ESP32 eine Handlungsanweisung an einen der beiden Aktoren Heizung oder Klimaanlage. Bei Unterschreitung der Min-Soll-Temperatur erhält die Heizung die Handlungsanweisung, beim Übertreten der Max-Soll-Temperatur hingegen die Klimaanlage. Störgrößen wären in diesem Fall Wärmeaustausch mit dem äußeren System.

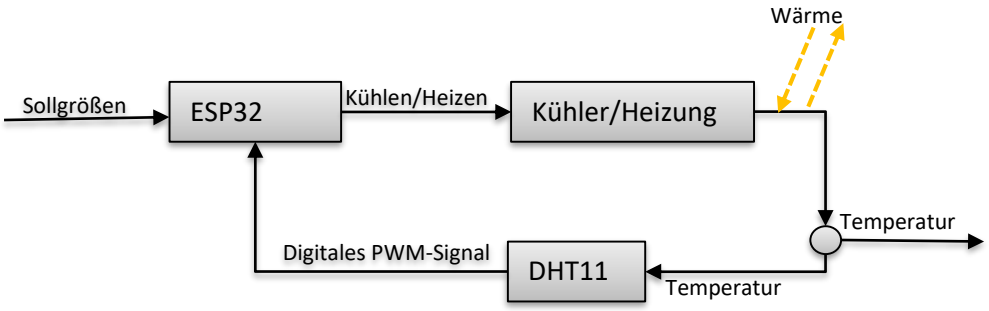


Abbildung 15: Regelkreislauf des Parameters Temperatur

3.1.5 Schaltplan des Modells

Ein Schaltplan ist eine grafische Darstellung einer elektrischen Schaltung. In diesem Projekt wird die Schaltung im Modell zwischen Aktoren, Sensoren und Regler gezeigt.

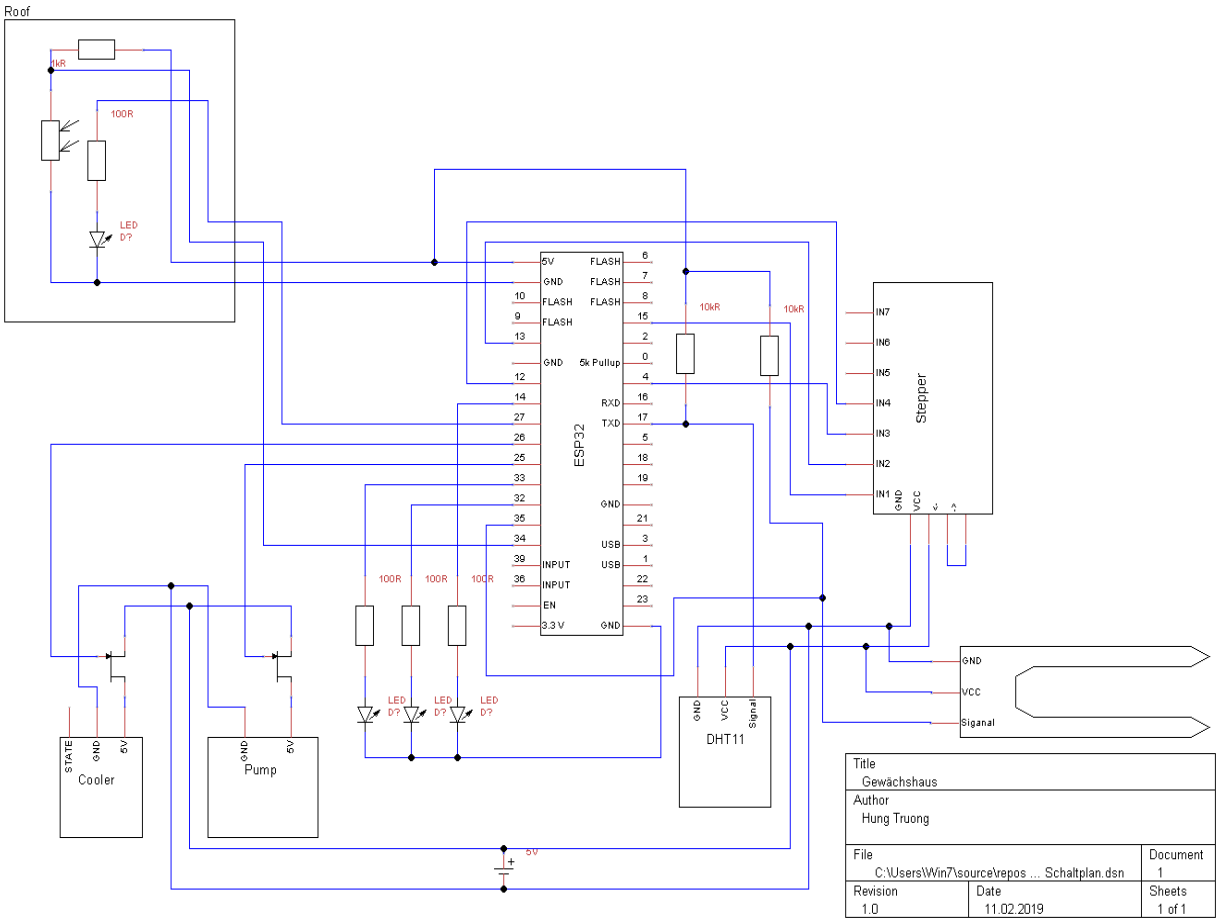


Abbildung 16: Schaltplan des Modells

Der Schaltplan gibt in der oberen rechten Ecke den speziellen Ort „Roof“ an, an dem sich der Fotowiderstand und die LED, welche das UV-Licht repräsentiert, zusammen aufhalten. Andere Sensoren und Aktoren haben ihren eigenen Platz. Nur noch der Mikrocontroller und einige LEDs sind zusammen in einem Plastikbehälter.

3.2 Das Smartphone

Das Smartphone muss mindestens die Version Android 5.0 (Lollipop) besitzen.

4 Software

Die Software für diese Projekt wurde in den Programmiersprachen C#, C für ArduinoIDE, Java , XML und SQL geschrieben. Als Entwicklungsumgebungen wurde für den Datenbank-Server Microsoft Visual Studio Enterprise 2017[17], für den Mikrocontroller ArduinoIDE[18] und für die Android-Applikation Android Studios[19] verwendet. Für die Planung und Gestaltung der Schaltpläne diente das kostenlose Tool TinyCAD[20] und für die Programmablaufpläne das kostenlose Tool PapDesigner[21]. Zudem wurde git[22] zum praktischen Verwalten des Codes benutzt.

Alle Grafiken, die in der Android-Applikation verwendet werden, stammen von der Internetseite material.io[23], sowie der Website www.flaticon.com und sind somit frei zur Verfügung gestellt.

4.1 Erfassung der Messwerte

Der Regler hat 18 Analog-zu-Digital-Konverter-Eingänge (im Folgenden: ADC-Eingänge), die mit einer Auflösung von 12 Bit funktionieren. Somit wandeln sie analoge Werte von 0 bis 3,3 V in ein digitales Signal von 0 bis 4095 um, wie in Abbildung 17 zu sehen ist. Ebenso ist zu beachten, dass die Eingänge nicht exakt linear AD-wandeln. Speziell zwischen 0 und 0,1 V und zwischen 3,2 und 3,3 V kann der Eingang nicht zwischen nebeneinanderliegenden Werten unterscheiden.

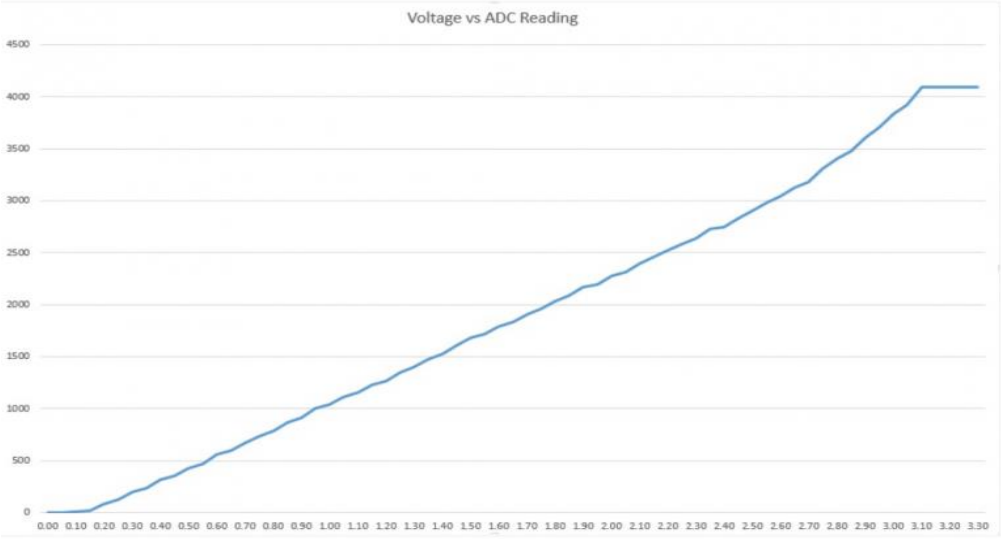


Abbildung 17: Kennlinie der ADC Eingänge des ESP32[24]

[17] https://visualstudio.microsoft.com/de/vs/enterprise/ (08:34, 17.02.19)
[18] https://www.arduino.cc/en/Main/Software (08:36, 17.02.19)
[19] https://developer.android.com/studio/ (08:36, 17.02.19)
[20] https://www.tinycad.net/ (08:37, 17.02.19)
[21] https://papdesigner.de.jaleco.com/ (08:37, 17.02.19)
[22] https://git-scm.com/ (08:37, 17.02.19)
[23] https://material.io/tools/icons/ (08:37, 17.02.19)
[24] https://randomnerdtutorials.com/esp32-pinout-reference-gpios/ (08:38, 17.02.19)

Diese ADC-Eingänge werden im Projekt zum Einlesen von zwei Parametern benötigt. Die Sensoren für diese Parameter liefern ein analoges Signal, welches umgewandelt werden muss. Die betroffenen Parameter sind Bodenfeuchtigkeit und Lichtstärke, wobei der Bodenfeuchtigkeitssensor eine Arbeitsspannung von 5V hat.

Der Verbund-Sensor liefert ein digitales Signal mit Pulsweitenmodulation, welches mit der entsprechenden Bibliothek gedeutet werden kann. Dieser Sensor arbeitet ebenfalls mit 5V, kann aber mit einer 3.3-V-Logik gedeutet werden.[25]

4.1.1 Pulsweitenmodulation

Die Pulsweitenmodulation (kurz: PWM) ist eine Modulationsart, die zwischen zwei elektrischen Spannungen oszilliert. Bei dieser Modulation wird gemessen, wie lange das entstehende Rechtecksignal sich beim HIGH-Wert befindet. Die Dauer bis zum Flankenwechsel wird als Pulsweite bezeichnet. Nach der abfallenden Flanke bleibt das Signal LOW, bis die nächste Periode beginnt. Die Periodendauer ist konstant. Um mit dem Signal arbeiten zu können, wird der Tastgrad[26] p berechnet. Dieser ist das Verhältnis zwischen Pulsweite und Periodendauer.

$$p = \frac{t_{ein}}{T} = \frac{t_{ein}}{t_{ein} + t_{aus}}$$

Bei einer kurzen Periodendauer und einer glättenden Tiefpasswirkung verhält sich das Signal der PWM wie ein analoges Signal, womit es möglich ist, Aktoren anzusteuern. Die mittlere Spannung lässt sich dann wie folgt berechnen:

$$U_m = U_{aus} + (U_{ein} - U_{aus}) * p$$

PWM wird ebenfalls in der Messtechnik benutzt. Sie ermöglicht es, analoge Signale ohne Übertragungsfehler zu übermitteln.

[25] https://cdn-learn.adafruit.com/downloads/pdf/dht.pdf?timestamp=1551247302 (11:02, 02.03.19)
[26] DIN IEC 60469-1

4.2 Steuerung der Aktoren

Die Steuerung der Aktoren ist in diesem Projekt sehr simpel. Um den Aktor laufen zu lassen, benötigt dieser ein HIGH Signal.

Diese Ansteuerung betrifft alle Aktoren des Projektes bis auf den Schrittmotor des Rollladens. Die Steuerplatine des Schrittmotors hat vier Eingänge zum Steuern des Schrittmotors. Diese Eingänge müssen verschieden angesteuert werden, damit sich der Schrittmotor dreht. Da die Arduino-IDE eine Bibliothek besitzt, die die Ansteuerung übernimmt, wird im Projekt nur angegeben, um wie viel Grad der Schrittmotor sich weiterbewegen soll.

Step	wire 1	wire 2	wire 3	wire 4
1	High	low	high	low
2	low	high	high	low
3	low	high	low	high
4	high	low	low	high

Abbildung 18: Ansteuerung eines bipolaren Schrittmotor für eine 360° Drehung[27]

4.3 Kommunikation …

Ohne digitale Kommunikation läuft in unserer heutigen hoch digitalen Welt nichts mehr. Fernsehausstrahlung, WLAN oder Bluetooth wären nicht mehr wegzudenken. Auch in diesem Projekt ist Kommunikation wichtig. Sie ermöglicht es dem Gartenhausbesitzer von überall auf der Welt auf das Gewächshaus zuzugreifen und es zu steuern. Diese digitale Kommunikation ist in diesem Projekt auf der Server-Client-Architektur aufgebaut.

4.3.1 … mit Server-Client Architektur

Diese Architektur beschreibt einen Informationsaustausch zwischen zwei Teilnehmer eines Netzwerkes. Vorteil dieser Architektur ist, dass Daten zentral gespeichert werden können, auf die man von überall zugreifen kann. Bei der Kommunikation mit dieser Architektur nimmt dabei ein Teilnehmer die Rolle des Servers und der andere die Rolle des Clients an.

Ein Client ist ein Teilnehmer am Netzwerk, welcher eine Dienstleistung beantragt. Neben dem Client wird auch eine Anfrage (Request) erstellt, die an den anderen Teilnehmer adressiert ist. Nach dem Senden wartet der Client auf eine Antwort des anderen Teilnehmers und wertet diese dann aus. Nach Beendigung des Dienstes wird der Client wieder zerstört.

[27] http://www.tigoe.com/pcomp/code/circuits/motors/stepper-motors/ (11:03, 02.03.19)

Der Server ist das Gegenstück zum Client. Dem Server wird ein Port zugeordnet, an dem er auf Anfragen wartet. Bei Erreichen einer Anfrage verarbeitet er diese und sendet die angeforderten Informationen an den Sender zurück. Nach der Kommunikation wartet der Server weiter auf Anfragen und wird nicht zerstört.

Um mit der Server-Client-Architektur zu arbeiten, stellt Microsoft die Socket-Klasse zur Verfügung. Sockets sind bereitgestellte Objekte, die als Kommunikationsendpunkt dienen, wobei Stream-Sockets eine Untergruppe bilden. Bei Stream-Sockets läuft es folgendermaßen ab:

Auf Serverseite wird ein Server-Socket erstellt und an einen Port gebunden. Dieser wartet dort auf Anfragen. Bei einer einkommenden Anfrage erstellt dieser einen Socket, an den er die Verbindung der Anfrage übergibt. Der Server-Socket ist somit fast durchgehend erreichbar. Der erstellte Socket führt die Dienstleistung aus und zerstört sich beim Ende der Kommunikation.

Die Clientseite erstellt, falls eine Kommunikation zum Server notwendig ist, einen Socket mit der Server-Adresse. Dieser verbindet sich dann mit dem Server und kommuniziert mit ihm bidirektional, also kann er senden und empfangen. Nach Benötigung der Kommunikation wird ein Ende-Signal an den Server gesendet und der Socket zerstört sich selbst.

Im gesamten Projekt wird für die Kommunikation die Server-Client-Architektur mit Protokoll TCP benutzt.

4.3.1.1 TCP/IP-Protokoll

Das TCP/IP-Protokoll ist ein Zusammenschluss aus Protokollen, in dessen Kern das Transmission Control Protocol (TCP), das User Datagram Protocol (UDP) und das Internet Protocol (IP) steht. Das Internet Control Message Protocol wird in diesem Fall vernachlässigt, da es ein Teil des IP ist. Zusammen beschreiben die Protokolle die TCP/IP-Schichten Transport und Internet und ermöglichen es Daten über ein Kommunikationsnetz zu übertragen.

Anwendungsschicht
Transportschicht
Internetschicht
Netzschicht

Abbildung 19: TCP/IP-Referenzmodell

Das Internet Protocol beschreibt dabei den Weg der Kommunikation. Das Protokoll findet den Weg zum richtigen Ziel und wieder zurück. Durch dieses Protokoll ist Routing möglich, was bedeutet, dass Pakete im Netzwerk weitergeleitet werden können und dabei immer den schnellsten möglichen Weg beschreiten. Es beschreibt somit die Internetebene des TCP/IP-Referenzmodells.

Das Transmission Control Protocol ist für die Datensicherheit zuständig. Das TCP stellt durch Sockets eine Verbindung zwischen zwei Endpunkten eines Netzwerkes her, worüber beidseitige Datenübertragung möglich ist. Zudem teilt das TCP die Daten in einzelne Pakete auf, welche an das Ziel geschickt werden. Dort wird mit einer Prüfsumme überprüft, ob alle Daten verlustfrei angekommen sind. Falls nicht, versucht das Protokoll eine erneute Kommunikation. Sie ist somit in der Transportschicht der TCP/IP-Referenzmodells tätig.

Das User Datagramm Protocol ist eine vereinfachte Version des TCP und somit ebenso in der Transportschicht. Problem des TCP ist die Datensicherung, die zu Verzögerungen führen kann. Dadurch wurde UDP entwickelt, was ohne diese Sicherung funktioniert.

In Abbildung 20 ist ein solches Paket zu sehen. Es hat einen IP-Kopf der 4. Version, indem die IP-Adresse des Zieles und der Quelle gespeichert ist. Ebenso ist die Protokollart, die komplette Länge und die Lebenszeit gespeichert, nachdem sich das Paket zerstören soll, falls es in der Zeit nicht am Ziel angekommen ist. Die Daten, welche übertragen werden soll, sind ebenfalls in dem Header gespeichert.

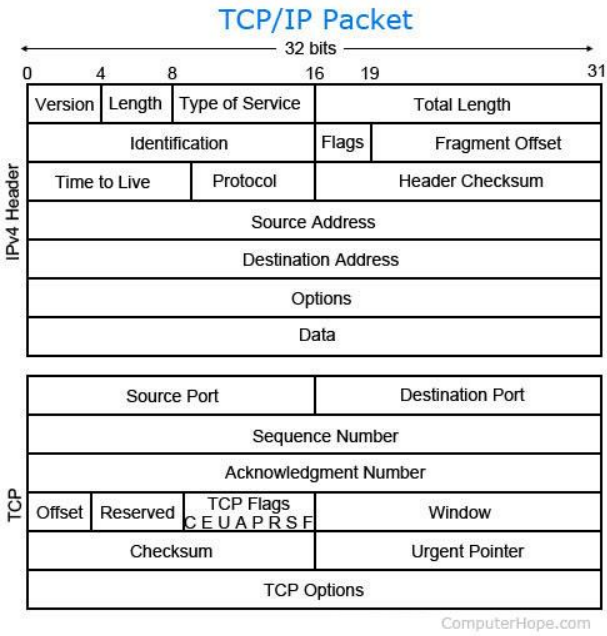


Abbildung 20: TCP/IPv4-Paket[23]

Unter dem Header finden sie den TCP-Block. In diesem sind Ziel- und Quell-Port gespeichert. Ebenso die Prüfsumme, mit der überprüft wird, ob alle Pakete angekommen sind. Insgesamt sind es 13 mal 32 bit Blöcke und somit ist ein Paket 52 Byte groß. [28]

[28] https://www.computerhope.com/jargon/t/tcpip.htm (09:03, 03.03.19)

4.3.2 Zwischen Regler und Datenbank-Server

Der Regler und der Datenbank-Server, im Folgenden Server genannt, kommunizieren miteinander und tauschen dabei meist Informationen über die eingetragene Pflanze am Regler. Beim allerersten Booten sendet der Regler den „new arduino“-Befehl an den Server, welcher mit der neuen Mikrocontroller-ID antwortet. Diese ID wird auf dem Regler gespeichert. Bei Neubooten liest der Regler diese Information aus und sendet mit der gelesenen ID ein reconnect-Befehl an den Server. Falls dem Regler eine Pflanze zugeordnet ist oder wird, sendet der Server die Sollwerte an den Regler. Bei Änderung der Sollwerte der Pflanze passiert dies ebenso.

Hat der Regler die Sollwerte empfangen, sendet dieser alle 500 ms die durchschnittlichen Werte der letzten 20 eingelesenen Istwerte an den Server, welcher die Werte in der Datenbank abspeichert.

Zuletzt gibt es noch die Kommunikationsmöglichkeit, dass der Server dem Regler den Befehl „live_“ mit einer IP-Adresse sendet. Dieser Befehl startet die sogenannte „Live“-Funktion, die später im Abschnitt 4.3.4 „Kommunikation zwischen App und Regler“ thematisiert wird.

4.3.3 Zwischen App und Datenbank-Server

Zwischen Android-Applikation, im Folgenden App genannt, und dem Datenbank-Server, im Folgenden Server genannt, werden alle Daten kommuniziert, also Pflanzendaten, Mikrocontroller-Daten und die eingelesenen Werte der Sensoren.

Für diese Kommunikation stehen der App viele Befehle zur Verfügung, welche einen ähnlichen Aufbau haben. Sie fangen mit einem Funktions-Schlüsselwort an, welches beschreibt, was der Befehl machen soll. Zur Auswahl stehen:

- „new“ zum Hinzufügen einer neuen Zeile in der Datenbank,
- „set“ zum Verändern eines Wertes in der Datenbank,
- „get“ zum Anfragen nach einem Wert aus der Datenbank und
- „delete“ zum Löschen einer Zeile in der Datenbank.

Nach dem Funktions-Schlüsselwort folgt das Tabellen-Schlüsselwort, also welche Datenbanktabelle benutzt werden soll. Hierbei ist die Entscheidung zwischen „arduino“, „plant“ und „data“ möglich.

Zudem kann jetzt noch ein Zusatz-Schlüsselwort hinzugefügt werden. Dabei stehen bestimmte Spalten der Tabelle sowie „all“ und „display“ zur Verfügung.

Nach diesen Schlüsselwörtern folgt ein Unterstich als Trennung zwischen Befehlskopf und den folgenden Parametern, welche ebenfalls durch Unterstriche getrennt werden.

Als Beispiel fungiert der Befehl „set arduino plantid“. Dieser Befehl ordnet dem Micro Contoller eine Pflanze zu. Dieser Befehl benötigt weitere zwei Parameter, die Mikrocontroller-Nummer als Ganzzahl (Integer) und die Pflanzen-Nummer ebenfalls als Ganzzahl(Integer).

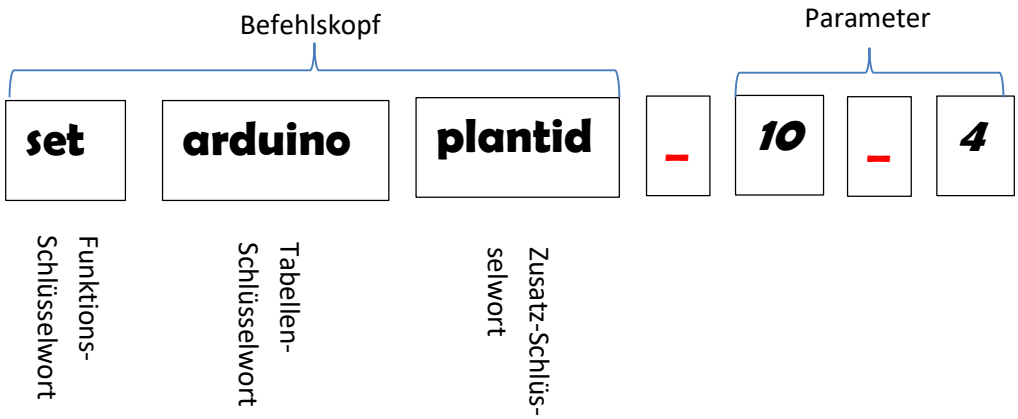


Abbildung 21: Aufbau eines Befehls am Beispiel „set arduino plantid"

Alle Befehle mit Beschreibung und Parametern sind im Anhang zu finden.

Die App fragt somit alles Mögliche an Werten an und bekommt vom Server die dazugehörige Antwort mit den Werten. Bei Anfrage von mehreren Werten werden Zeilen mit „;“ getrennt und einzelne Werte mit „_“.

Eine volle Dokumentation aller Befehle sind im Anhang zu finden.

4.3.4 Zwischen App und Regler

Die Kommunikation zwischen Applikation und Regler besteht nur für einen bestimmten Zeitraum und ist somit eine spezielle Kommunikation. Sie findet nur statt, wenn „Live“-Modus/ „Live“- Funktion aktiviert ist. Die sogenannte Live-Funktion ermöglicht es, dem Benutzer die eingelesenen Parameter direkt zu sehen und um die Aktoren manuell zu steuern. Um diese Funktion zu starten, sendet die Android-Applikation den Befehl „live“ an den C#-Server. Dieser leitet mit der IP-Adresse des Android-Handys, auf dem die Applikation läuft, an den zu steuernden Mikrocontroller. Dieser sendet die aktuellen Werte und den Betriebszustand der Aktoren an die Android-Applikation. Diese übermittelt dann in Dauerschleife bis zum Abbruch den gewünschten Zustand der Aktoren.

Diese Kommunikation läuft direkt zwischen Android App und Mikrocontroller, wobei der C#-Server nur am Anfang gebraucht wird. Zum Beenden sendet die Android App einen Ende-Befehl an den Mikrocontroller und diese Art der Kommunikation ist beendet.

Ein Sequenzdiagramm ist im Anhang vorhanden.

4.4 Verwaltung der Daten

Da dieses Projekt viele Daten zum Speichern und Verwalten hat, wird hierfür dies eine Datenbank verwendet.

Eine Datenbank ist ein elektronisches Verwaltungssystem, welches große Datenmengen zentral speichert und verwaltet. Vorteile sind, dass Redundanzen und Inkonsistenzen verhindert werden. Dadurch werden mehrfache Speicherungen der gleichen Information und das Problem der Aktualisierung verhindert. Ebenso löst es das Problem der zentralen Datenspeicherung, wodurch nun mehrere Programme auf die Datenbank Zugriff haben können. Um diese Daten zu ändern, wird eine Skriptsprache benötigt. In diesem Fall wird der Standard SQL benutzt.

Das Projekt benutzt den Microsoft-SQL-Server um Pflanzenwerte, Mikrocontrollerwerte und Messwerte zu speichern. Dieser Server läuft lokal auf dem Rechner und nur der C#-Server kann auf diesen zugreifen. Die genannten drei Abteilungen haben jeweils eine eigene Tabelle in der Datenbank. Die Tabelle der Pflanzenwerte speichert den Namen, die ID und die Min/Max-Sollwerte der Pflanze. Die Mikrocontrollertabelle speichert nur ihre ID, die IP-Adresse des Mikrocontrollers und die Pflanzen-ID, die dem Mikrocontroller zugeteilt ist. Die Messwert-Tabelle speichert, wie der Name sagt, die Messwerte und die Zeit, zu der sie gemessen wurden.

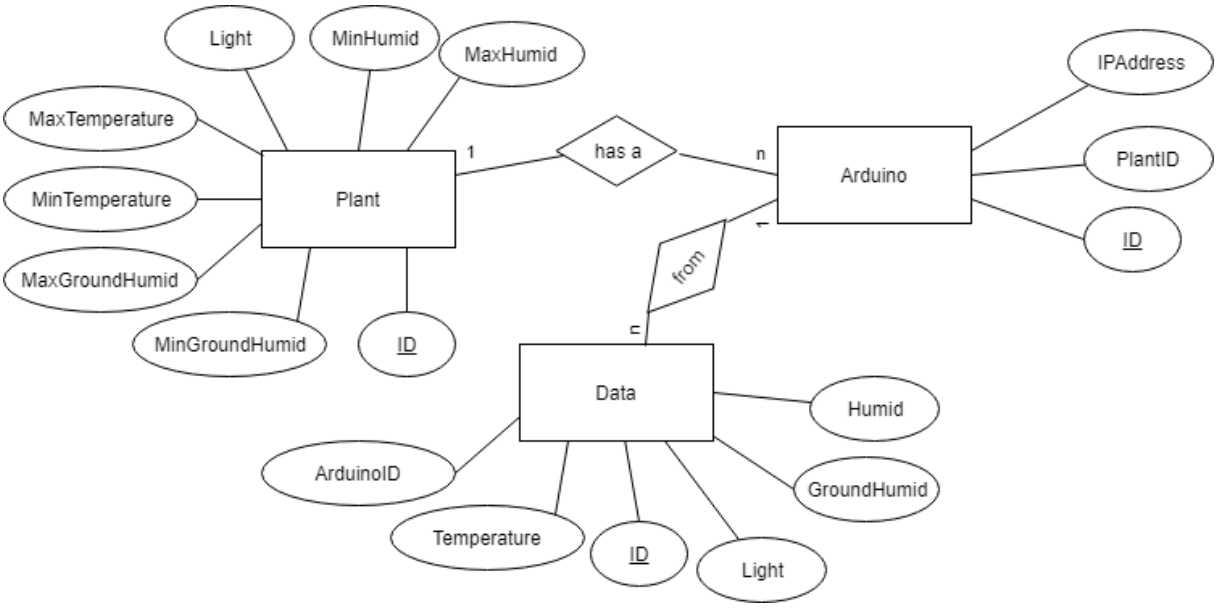


Abbildung 22: ER-Modell der Datenbank

4.4.1 SQL

Die Datenbanksprache SQL wird von den meisten Datenbanksystemen unterstützt. Mit dieser Datenbanksprache kann man Daten definieren, hinzufügen, bearbeiten, gruppieren, auslesen und löschen. Die Befehle ähneln der englischen Umgangssprache und die Syntax der Befehle ist sehr simpel. Diese Datenbanksprache wird im Projekt verwendet, um zwischen Server und Datenbank zu kommunizieren.

Normalerweise wird der SQL-Befehl zum Server gesendet und verarbeitet. Ich habe mich dagegen entschieden und eigene Befehle für die Kommunikation mit dem Server definiert, um mehr Einfachheit zu schaffen. Ebenso können mehrere Daten gleichzeitig übermittelt werden.

4.5 Darstellung der Parameterhistorie

Die Datenbank besitzt, wie im vorherigen Abschnitt beschrieben, eine Tabelle der gemessenen Werte an den Reglern. Da dies viele Zahlen sind, welche tabellarisch aufgelistet sehr komplex wirken, werden die Werte graphisch dargestellt.

Diese graphische Darstellung findet in der Android-Applikation statt und zeigt immer nur die Werte eines Parameters eines Reglers an. Für diese graphische Darstellung wurde die graphView-Bibliothek[29] benutzt. Diese Bibliothek stellt Klassen zur Verfügung, Graphen vereinfacht in einer Android Applikation in einem Koordinatensystem darzustellen. Zudem ermöglicht die Klasse das Durchscrollen und das Zoomen in dem Koordinatensystem. Einziger Nachteil ist die beschränkte Einteilung der Achsen auf Zahlen, Texte und Zeitangaben.

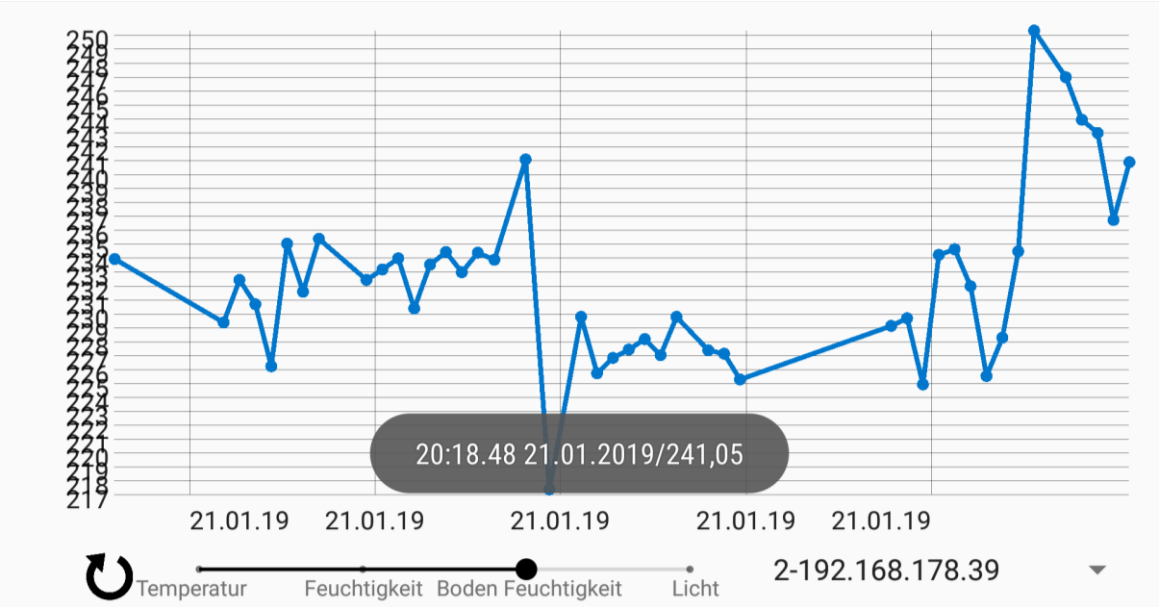


Abbildung 23: Historie-Darstellung des Parameter Bodenfeuchtigkeit des Regler 2

[29] http://www.android-graphview.org/ (21:27, 19.02.19)

4.6 Graphische Oberfläche

Die graphische Oberfläche ist die Schnittstelle zwischen Projekt und Benutzer. Dabei wurde der Fokus auf Einfachheit und instinktive Benutzung gelegt. Diese graphische Oberfläche besteht im Android-Betriebssystem aus der App-Bar und dem Layout. Die graphische Oberfläche wird dabei durch die Auszeichnungssprache XML beschrieben, die man in Android Studios sehr simpel mit dem Layout Editor ändern kann.

Das Layout mit den Benutzerelementen besteht aus dem Grundformat Relative Layout. Dieses Format ermöglicht eine Definition des Platzes für Objekte in Relation zu anderen Objekten oder zu der Form selbst. In dieser Grundform wurden andere Formen benutzt, um die Objekte zu gruppieren.

Die Menüleiste im unteren Teil der App und dient zur Navigation zwischen dem Startfenster, dem Pflanzenfenster und dem Reglerfenster.

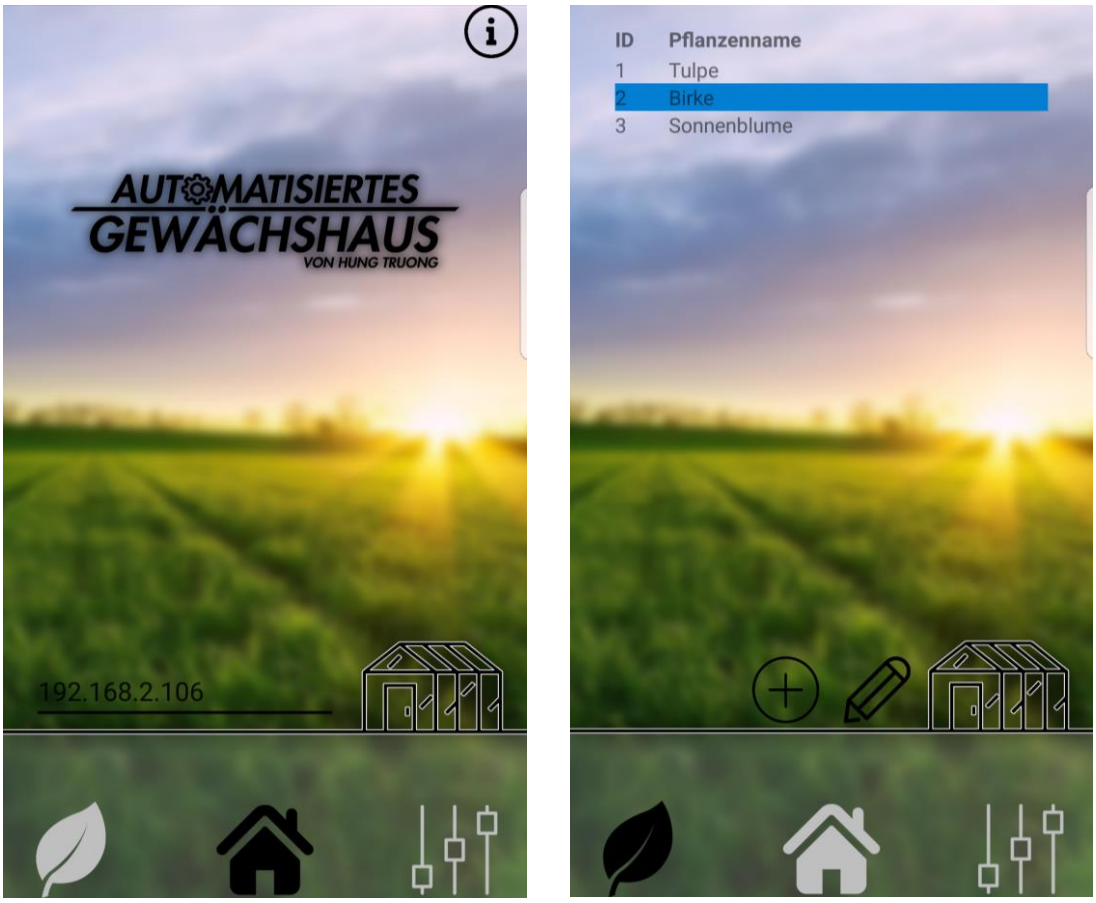


Abbildung 24: Screenshots der Android Applikation

5 Bedienungsanleitung

Im Folgenden wird erklärt, wie dieses Projekt bedient wird.

5.1 Erstkonfiguration

Vor dem Benutzen des Mikrocontrollers muss der Code auf den Controller geladen werden. In diesem Code muss man noch den WLAN-Namen, das WLAN-Passwort und die Server-IP eintragen, damit dieser funktioniert.

Ebenso muss man die Server-IP in der App eintragen. Das Feld dafür findet man direkt auf dem Start-Fenster der App.

5.2 Hinweise

Um zum Start-Fenster zu kommen, muss unten in der Menüleiste das Haus-Symbol angetippt werden.

Ebenso gibt es die Möglichkeit über die untere Menü-Leiste zum Regler- oder zum Pflanzenfenster zu wechseln. Dafür muss das jeweilige Symbol angetippt werden.

Bei einigen Fenstern kann das Laden dauern, da eine vorherige Kommunikation zum Server vonnöten ist.

Falls die Kommunikation misslingt, wird das neue Fenster trotzdessen geöffnet. Dieses Fenster wird dann nur das Schlüsselwort „Error“ beinhalten. In diesem Fall kann das Fenster neu geladen werden, indem das jeweilige Symbol in der Menüleiste angetippt wird. In einigen Fenstern ist ebenfalls ein Neulade-Symbol vorhanden.

5.3 Festlegung einer Pflanze

Bevor ein Gewächshaus eine Pflanze zugeordnet bekommen kann, muss eine Pflanze deklariert werden. Um dies zu tun, muss das Pflanzen-Fenster über die Menü-Leiste geöffnet werden. In diesem Fenster ist ein Hinzufüge-Symbol (Kreis mit Plus innendrin) vorhanden, welcher nach Betätigung zu einem neuen Fenster führt. In diesem Fenster müssen die Soll-Werte und der Name der Pflanze eintragen werden und mit dem Speicher-Symbol am unteren Ende bestätigt werden.

Soll eine vorhandene Pflanze verändert oder gelöscht werden, muss das Pflanzen-Fenster geöffnet werden. In diesem wird die zu bearbeitende Pflanze in der Tabelle ausgewählt. Nach Auswahl der Pflanze, wird die Zeile blau hinterlegt und ein Stift-Symbol ist am unteren Ende des Bildschirms zu finden. Nach Antippen dieses Stiftes wird ein neues Fenster geöffnet, in dem die Änderungen durchgeführt werden können. Diese müssen zum Schluss mit dem Speicher-Symbol bestätigt werden. Zum Löschen muss das Mülleimer-Symbol betätigt werden.

5.4 Zuordnung Pflanze Gewächshaus

Damit ein Gewächshaus eine Pflanze zugeordnet bekommt, um arbeiten zu können, muss das Regler-Fenster aufgerufen werden. Dieses wird wieder über die Menüleiste geöffnet. In diesem Regler-Fenster werden alle Regler tabellarisch aufgelistet, die sich mindestens einmal mit dem Server verbunden haben.

In dieser Tabelle wird der richtige Regler durch Tippen ausgewählt. Die Zeile des ausgewählten Reglers erleuchtet blau und im unteren Teil des Bildschirmes findet man Bearbeitungsmöglichkeiten. Beim Auswählen des Reglers kann es dabei zu einigen Sekunden Wartezeit kommen Nun wird die Pflanze ausgewählt, wobei das Feld „keine“ Pflanze auch ausgewählt werden kann. und mit dem Speicher-Symbol wird die Änderung bestätigt.

5.5 Manuelle Steuerung des Gewächshauses

Um die manuelle direkte Steuerung des Gewächshauses zu übernehmen, muss das Regler-Fenster geöffnet und der zu steuernde Regler ausgewählt werden. Nach der Auswahl erscheint im unteren Teil des Bildschirmes ein Fernbedienungs-Symbol, welches angetippt werden muss. Dieses führt zum „Live“-Fenster, welches in drei Abschnitte unterteilt ist.

Der erste Abschnitt ist die Anzeige der live eingelesenen Parameter des Gewächshauses. Der zweite Abschnitt ist der Steuerungsbereich für die Aktoren und der dritte Abschnitt beinhaltet den Start-Stopp-Knopf. Der Start-Knopf muss betätigt werden, um die „Live“-Übertragung zu starten. Erst nach dem Bestätigen werden Abschnitt eins und zwei aktiv. Nun können die Aktoren live gesteuert werden.

Um die Übertragung zu stoppen, muss der Stopp-Knopf betätigt werden, welcher ebenso das Fenster schließt.

5.6 Anzeige der Parameterhistorie

Um die Historie der Parameter graphisch anzeigen zu lassen, müssen Sie zum Regler-Fenster. In diesem ist im unteren Teil ein Graph-Symbol, welcher nach Betätigung zu einem neuen Fenster führt. In diesem Fenster ist ein Koordinatensystem und eine Steuerungsleiste vorhanden. In der Steuerungsleiste wird der Regler und der Parameter ausgewählt, welcher angezeigt werden soll. Daraufhin wird ein Graph in das Koordinatensystem gezeichnet, welches durch Zoomen und Scrollen untersucht werden kann. Durch Antippen der Punkte des Graphen können weitere Informationen angezeigt werden.

5.7 Entfernen der Applikation

Zum Entfernen der Applikation muss diese, wie andere Applikationen vom Smartphone gelöscht werden.

6 Ergebnis

Die Arbeit an diesem Projekt ist sehr interessant und hat mir viele neue Einblicke in verschiedene Themengebiete der Informatik eröffnet.

Es war eine neue Erfahrung, das theoretisch Gelernte praktisch an einem Modell zu erarbeiten. Die Herausforderung war dabei sowohl Software und Hardware als auch die schriftliche Dokumentation zu erarbeiten, wodurch ich mir eine bessere Vorstellung davon erschafft habe, wie eine Bachelor-Arbeit an der Universität aussehen könnte.

Ich bin zufrieden mit meinem Projekt, obwohl es Zeiten gab, in denen es hoffnungslos schien. Es gab verschiedene Probleme und viele Wissenslücken, die zunächst durch umfangreiche Recherchearbeit geschlossen werden mussten. Durchgebrannte LEDs, eine chaotische und unverständlich gewordene erste Version des Projektes und die nicht aufbauende Kommunikation zwischen zwei Teilnehmern waren einige der Hindernisse.

Schlussendlich habe ich dennoch ein zufriedenstellendes Modell erarbeiten können und bin erfreut, mein persönliches Lernziel erreicht zu haben.

6.1 Resümee

Die anfangs festgelegten Ziele konnten während der Projektarbeit verwirklicht werden. Es ist mir gelungen, ein Modell zu erstellen, in dem man eine mögliche Automatisierung eines Gewächshauses darstellen kann. Dabei sind einige Punkte besonders hervorzuheben:

- Durch den kompakten Aufbau des Reglers mit den Sensoren und Aktoren können diese als Modul verkauft werden. Somit kann ein großes Gewächshaus mehrere Regler für jeweils eine Pflanze besitzen, welche in verschiedenen Beeten gepflanzt sind. Diese verschiedenen Regler können über ein System gesteuert werden, wodurch das System benutzerfreundlicher ist.
- Durch die einfache graphische Oberfläche kann der Nutzer dieses Systems sehr schnell und einfach alle Funktionen ansteuern, die dieses Projekt beinhaltet.
- Durch die objektorientierte Programmierung ist dieses System einfach und schnell zu warten oder zu erweitern. Einzig die vielen Programmiersprachen, die verwendet werden, wie C#, SQL, C für Arduino IDE, Java und XML, stellen ein Hindernis dar.
- Die Materialbeschaffung war insgesamt kostengünstig, da einige Bestandteile aus alten, kaputten Geräten recycelt werden konnten. Dabei beliefen sich die Kosten ohne Computer auf etwa 50€.

6.2 Ausblick

Während der Projektarbeit sind mir zahlreiche Ideen gekommen, wie man dieses Projekt noch erweitern und optimieren könnte. Diese wären jedoch sehr zeitaufwändig gewesen und in der stressigen Prüfungsphase des Abiturs nicht umsetzbar. Aus diesem Grund musste ich mich zunächst gegen die Einbindung der Ideen einsetzen.

Nach der allgemeinen Hochschulreife werde ich höchstwahrscheinlich weiter an dem Projekt arbeiten und die folgenden Ideen einbauen:

- Bluetooth-Steuerung des Gewächshauses mit der Applikation
- Regelung des Kohlenstoffdioxids in der Luft
- Regelung der Nährstoffe im Erdreich
- eine Pflanzenkamera mit Videoübertragung zur Applikation
- Applikation für andere Betriebssysteme (z. B. IOS)
- eine Online-Datenbank mit bereits vordefinierten häufig genutzten Pflanzen
- Einsatz von mehreren Sensoren zum genaueren Einlesen der Parameter
- eine Benutzeroberfläche der Android Applikation, die sich an der jeweiligen Displaygröße des Smartphones orientiert

Mein Ziel ist zunächst der Einsatz im eigenen Haushalt. Eine weitere Möglichkeit wäre der Verkauf des Produktes, wobei die Konkurrenz in diesem Sektor auf dem Markt sehr groß und stark ist.

7 Selbstständigkeitserklärung

Hiermit versichere ich, dass die vorliegende Besondere Lernleistung selbstständig und nur unter Verwendung der angegebenen Quellen und Hilfsmittel angefertigt wurde. Mir ist bekannt, dass bei nachgewiesenem Täuschungsversuch die Prüfung als „nicht bestanden“ erklärt werden kann.

X

Datum, Unterschrift von Hung Truong

8 Quellenverzeichnis

Theis, Thomas (2017): Einstieg in C# mit Visual Studio 2017. Ideal für Programmiereinsteiger, 5. Auflage, Bonn: Rheinwerk Computing

https://www.pflanzenforschung.de/de/journal/journalbeitrage/home-sweet-home-fuenf-faktoren-machen-aus-einem-standor-10385 (19:24, 26.01.19)

https://arts.bio/de/pflanzenkunde/wachstumsfaktoren-der-pflanzen (19:24, 26.01.19)

https://www.pflanzenforschung.de/de/themen/lexikon/transpiration-846/ (19:50, 26.01.19)

https://www.pflanzenforschung.de/de/themen/lexikon/photosynthese-285 (19:52, 26.01.19)

https://www.fremdwort.de/suchen/bedeutung/aktuator# (14:47, 28.01.19)

https://www.tutorialspoint.com/unix_sockets/what_is_socket.htm (15:36, 28.01.19)

https://docs.microsoft.com/en-us/windows/uwp/networking/sockets (15:59, 28.01.19)

https://docs.oracle.com/javase/tutorial/networking/sockets/definition.html (16:07, 28.01.19)

http://www.datenbanken-verstehen.de/ (18:55, 28.01.19)

https://www.uni-ulm.de/fileadmin/website_uni_ulm/iui.inst.050/vorlesungen/wise1011/les/Uebung_2.pdf (10:28, 02.03.19)

http://digital.ni.com/public.nsf/allkb/9C90B9A416A509DD86257EF2007C1926 (10:29, 02.04.19)

https://developer.android.com/guide/topics/ui (11:04, 04.03.19)

https://developer.android.com/studio/write/layout-editor#create-layout (11:04 04.03.19)

http://www.sqlcourse.com/intro.html (11:07, 04.03.19)

https://de.wikipedia.org/wiki/SQL(11:07, 04.03.19)

https://de.wikipedia.org/wiki/Internet_Protocol (16:56, 04.03.19)

https://de.wikipedia.org/wiki/Internetprotokollfamilie#TCP/IP-Referenzmodell (16:59, 04.03.19)

https://de.wikipedia.org/wiki/Transmission_Control_Protocol (17:02, 04.03.19)

https://de.wikipedia.org/wiki/Transmission_Control_Protocol/Internet_Protocol (17:03, 04.03.19)

https://de.wikipedia.org/wiki/User_Datagram_Protocol (17:03, 04.03.19)

https://www.computerhope.com/jargon/t/tcpip.htm (17:34, 04.03.19)

https://www.webopedia.com/TERM/I/IP.html (17:45, 04.03.19)

Abbildung 2, 14, 15, 19, 21, 22 mit Microsoft Word 2016 erstellt

Abbildung 8, 9, 10 mit Microsoft Word 2016 bearbeitet

Abbildung 16 mit TinyCAD erstellt - https://www.tinycad.net/ (08:37, 17.02.19)

9 Anhang

Im Anhang finden Sie den Code, sowie Datenblätter und weitere Informationen.

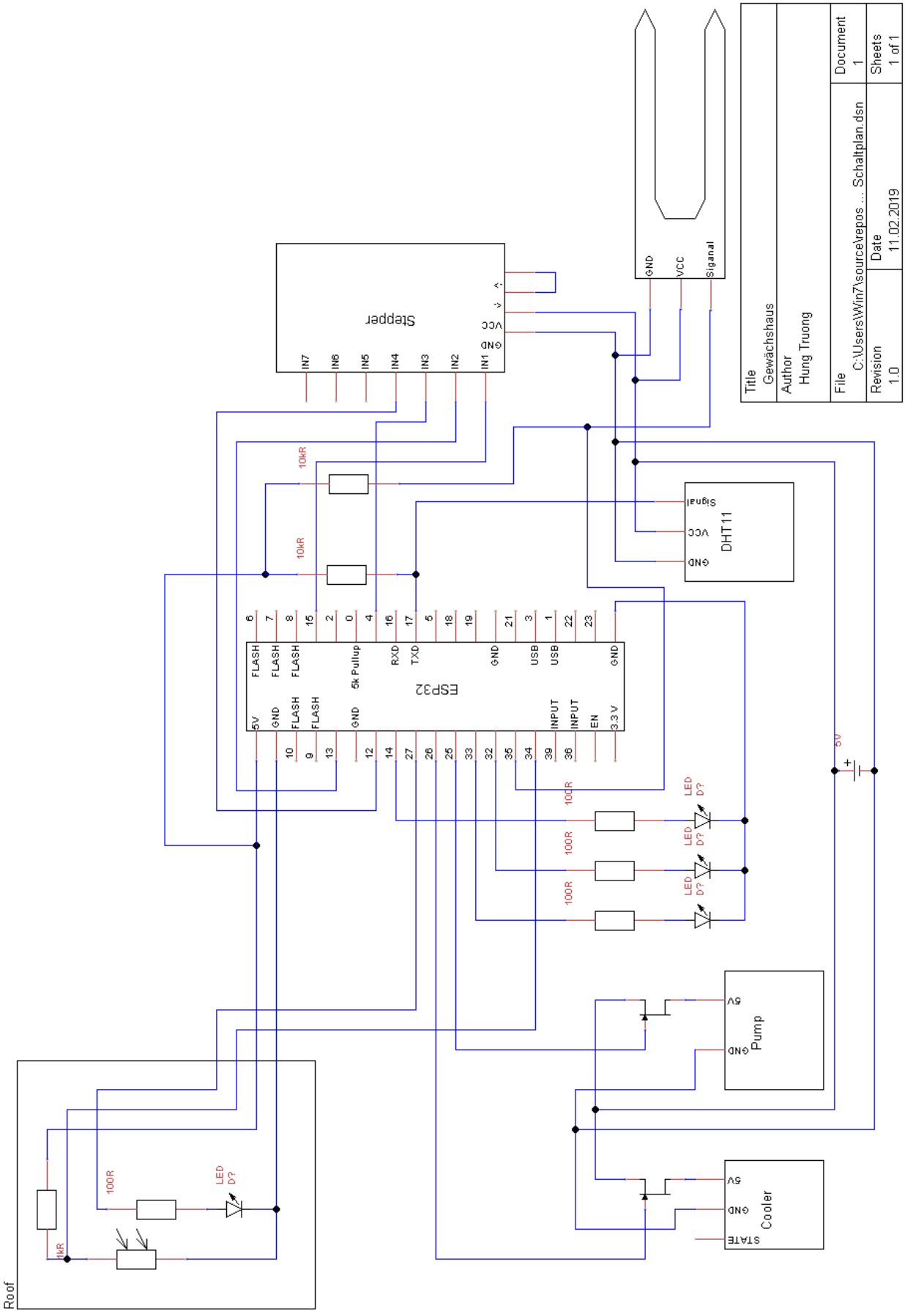
9.1 Code

Im Folgenden finden sie eine CD, auf welcher der Programmcode sowie einige Programme und Datenblätter zu finden sind. Diese sind nicht in ausgedruckter Form vorzufinden, da ihr Umfang zu groß ist.

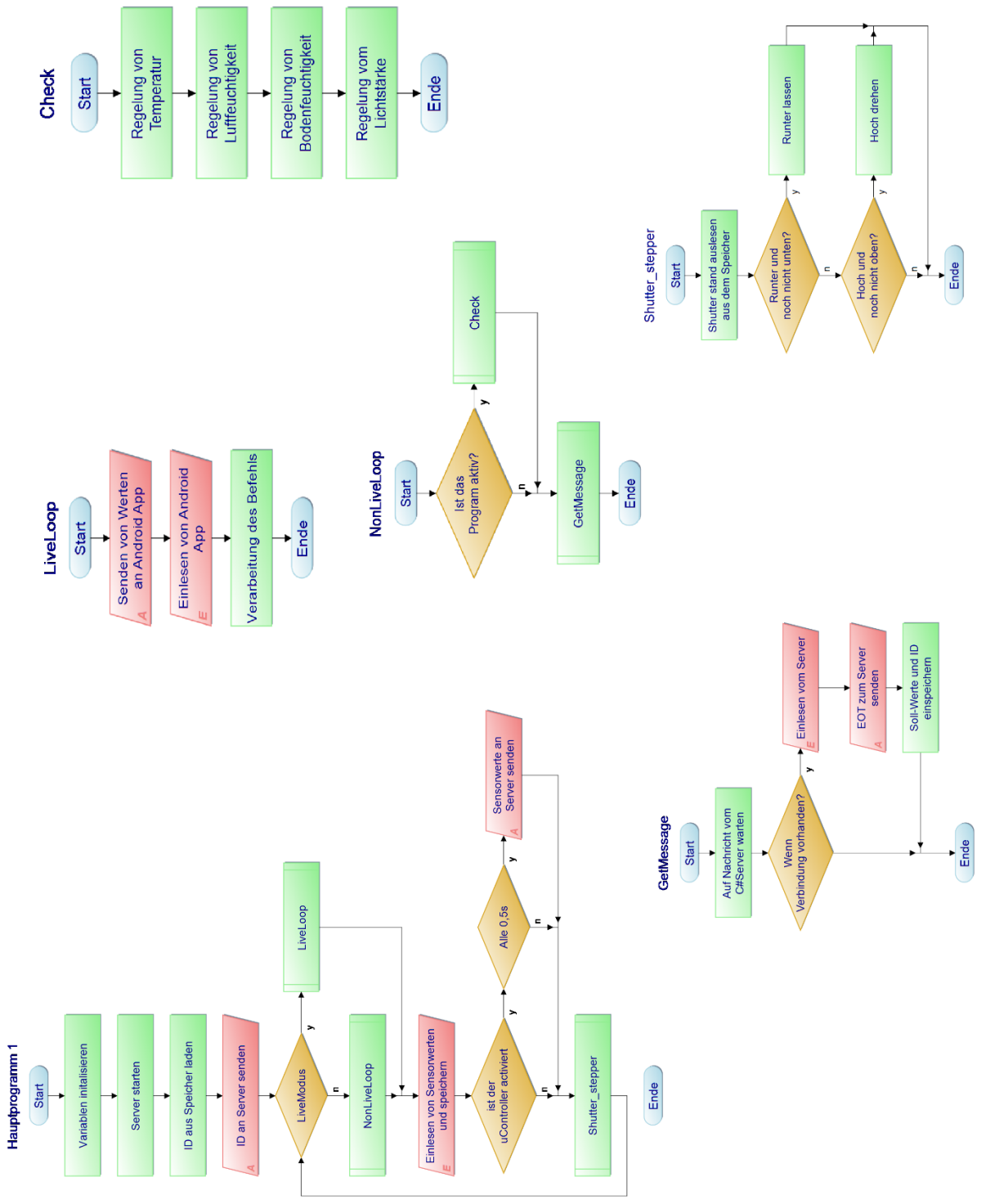
Ebenso können Sie den Code online abrufen. Sie finden ihn unter dem Link:

https://github.com/Gnuhry/Gartenhaus

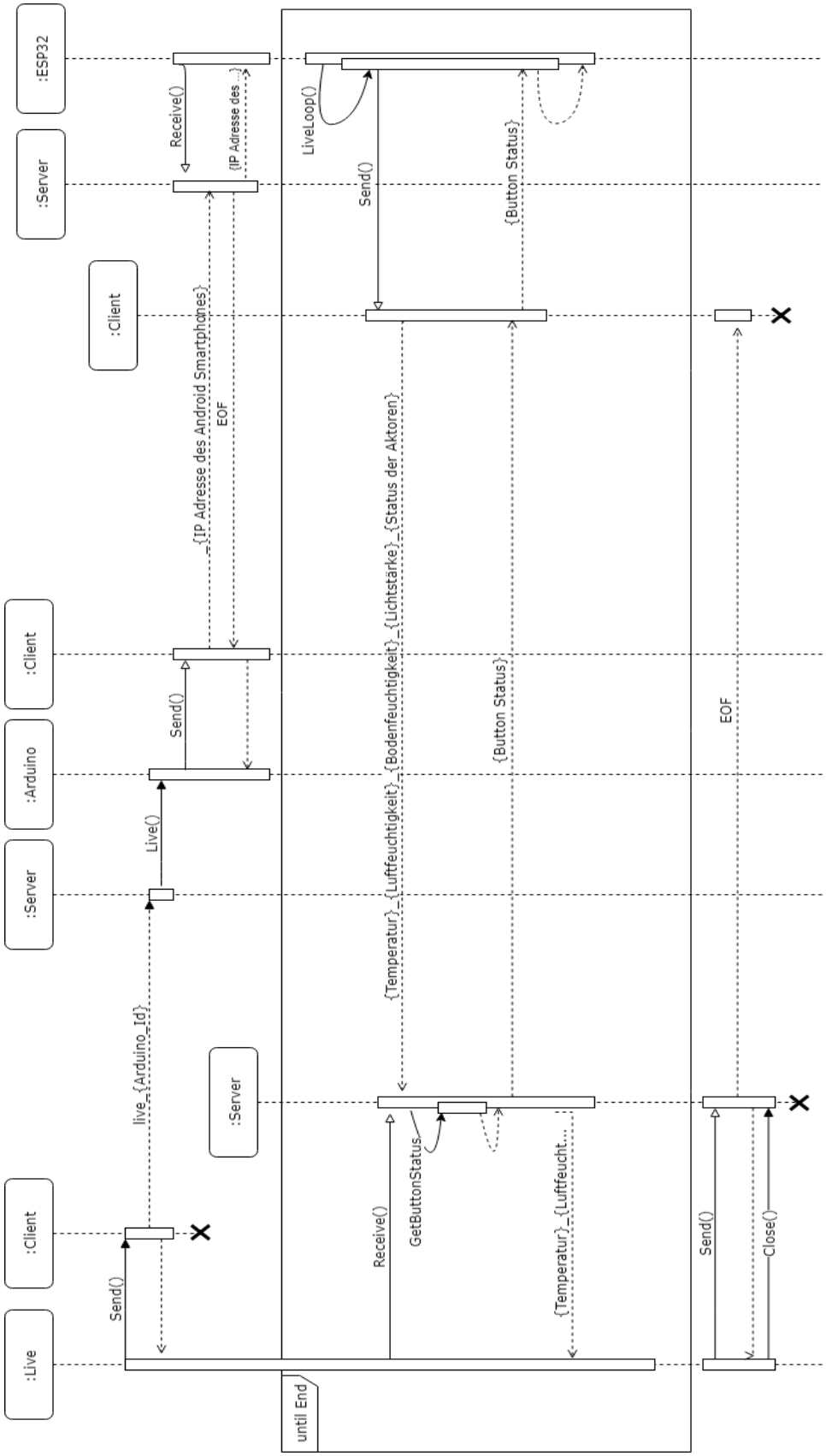
9.2 Schaltplan



9.3 Programmablaufplan des Reglers

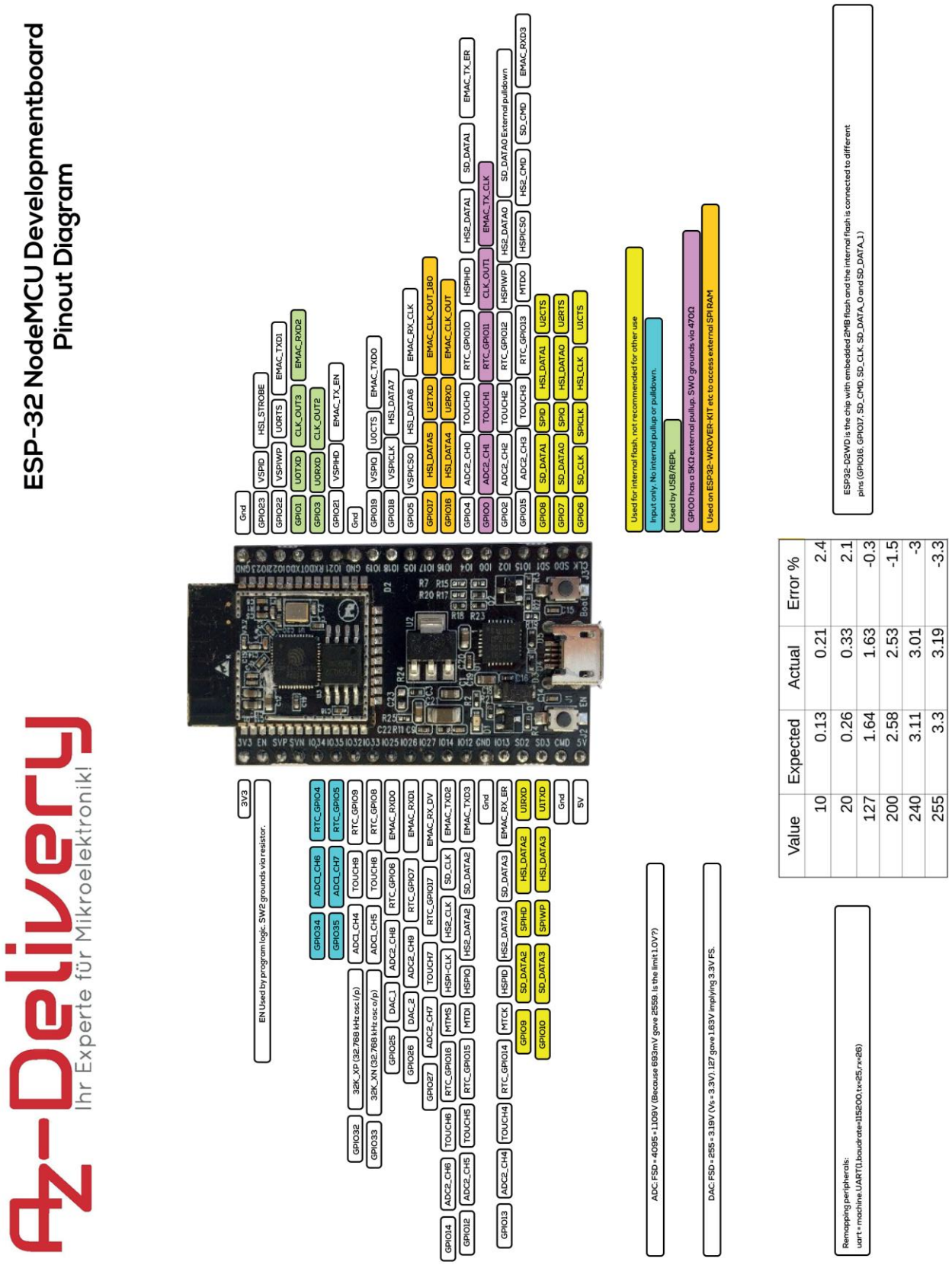


9.4 Sequenzdiagramm der „Live“-Funktion



9.5 Liste alle Befehle der Kommunikation mit dem C# Server

9.6 Pinout Diagramm des Reglers



Value	Expected	Actual	Error %
10	0.13	0.21	2.4
20	0.26	0.33	2.1
127	1.64	1.63	-0.3
200	2.58	2.53	-1.5
240	3.11	3.01	-3
255	3.3	3.19	-3.3

9.7 Datenblätter