Staatliche Technikerschule Berlin

Projektierung, Aufbau und Inbetriebnahme eines automatisierten Gewächshauses

Projektarbeit von

Graf, Torsten
Roth, Fabian
Nawroth, Stephanie

betreut durch

Herrn Kurt Jankowski-Tepe

Fachrichtung: Elektrotechnik
Schwerpunkt: Automatisierungs-/Energietechnik

Berlin, den 22.September 2019

Kurzbeschreibung/ Accept

Die vorliegende Dokumentation befasst sich mit der Projektierung, dem Aufbau und der Inbetriebnahme eines automatisierten Gewächshauses.

Ziel der Arbeit ist es, ein funktionstüchtiges Gewächshaus im Kleinformat herzustellen, das wie in dem vorliegenden Fall auf einem Balkon beziehungsweise auf kleinem Raum betrieben werden kann.
Gleichzeitig soll es als Grundlage/ Modell für einen Aufbau in größerem Maßstab dienen, beispielsweise für Haus- und Gartenbesitzer, mit der Option, vorhandene Photovoltaik-Anlagen als Energiequelle nutzen zu können.

This documentation discusses the design, construction and commissioning of an automated greenhouse.

The aim of the work is to produce a fully functional small-format greenhouse that can be operated on a balcony or in a small space, as in the present case. At the same time, it is intended to serve as a basis or model for a larger-scale construction, for example for house and garden owners, with the option of being able to use existing photovoltaic systems as an energy source.

Inhaltsverzeichnis

1 Projektierung

1.1 Überblick

Für die Umsetzung der Aufgabenstellung ist es erforderlich sich einen Überblick über die zur Verfügung stehenden Mittel, Bedingungen am Aufbauort und technischen Voraussetzungen zu machen.
Das Gewächshaus soll auf einem Balkon Platz finden.
Der Platzbedarf ist also bei einer nutzbaren Fläche von ca.4m x 1,5m gering zu halten. Im Fall dieses Projekts kommt erschwerend hinzu, dass es sich um einen Balkon im Dachgeschoss, Südseite handelt.
Im Sommer ist also mit viel Sonneneinstrahlung, ergo viel UV-Licht zu rechnen. Es ist demnach bei der Wahl der Materialien darauf zu achten, dass sie den Anforderungen gerecht werden und einen optimalen Pflanzenschutz und Pflanzenwuchs gewährleisten.
Natürlich hängt die Auswahl der Materialien auch immer von den finanziellen Mitteln ab die einem zur Verfügung stehen.

1.2 Motivation für das Projekt

Die Intention des Projekts war es zum einen eine Möglichkeit zu finden kleinere Pflanzen-, Gemüse-, Obstarten ziehen zu können unter Berücksichtigung des geringen gegebenen Platzes.
Zum anderen soll die Gewächshausversion im Kleinformat auch als Vorlage dienen für die Anwendung im größeren Maßstab, im Bereich der Haus- und Gartenbesitzer. Unter anderem mit der Möglichkeit Solarenergie zu nutzen, wenn zum Beispiel eine Photovoltaik-Anlage am Einsatzort zur Verfügung steht.

2 Inbetriebnahme

In diesem Kapitel geht es darum, das in der Planungsphase entworfene Modell (*siehe 2.1*) aufzubauen und in Betrieb zunehmen.
Nähere Informationen und Details zu den verwendeten Komponenten, sowie zur Programmierung der Steuerung sind ab *Kapitel 3* zu finden.

2.1 Vorbereitung / Planung

Wie unter Punkt *1.1 Überblick* bereits erwähnt, müssen einige Vorgaben in die Projektplanung mit einbezogen werden.
Am Installationsort steht nur begrenzt Platz zur Verfügung, dadurch muss das Grundgerüst des Gewächshauses entsprechend dimensioniert werden und darf in diesem Fall ein Höchstmaß von 800mm x 400mm x 400mm (B x H x T) nicht überschreiten.

Des Weiteren muss man sich überlegen wie man die Elektroinstallation vornimmt. Ob die Steuerungselektronik mit in das Gewächshaus integriert werden, oder zum Beispiel als separater Schaltkasten ausgelagert werden soll.
Außerdem muss man bedenken welche äußeren Einflüsse, unter anderem Wetter, auf das Gewächshaus und die verwendeten Komponenten wirken können.
Auch die Energieversorgung sollte im Vorfeld betrachtet werden.
Ist beispielsweise eine Außensteckdose vorhanden oder muss anderweitig für eine Stromversorgung gesorgt werden. Und wenn ja welche Optionen gibt es?

Aus der Planungsphase geht folgender erster Entwurf für dieses Projekt hervor:

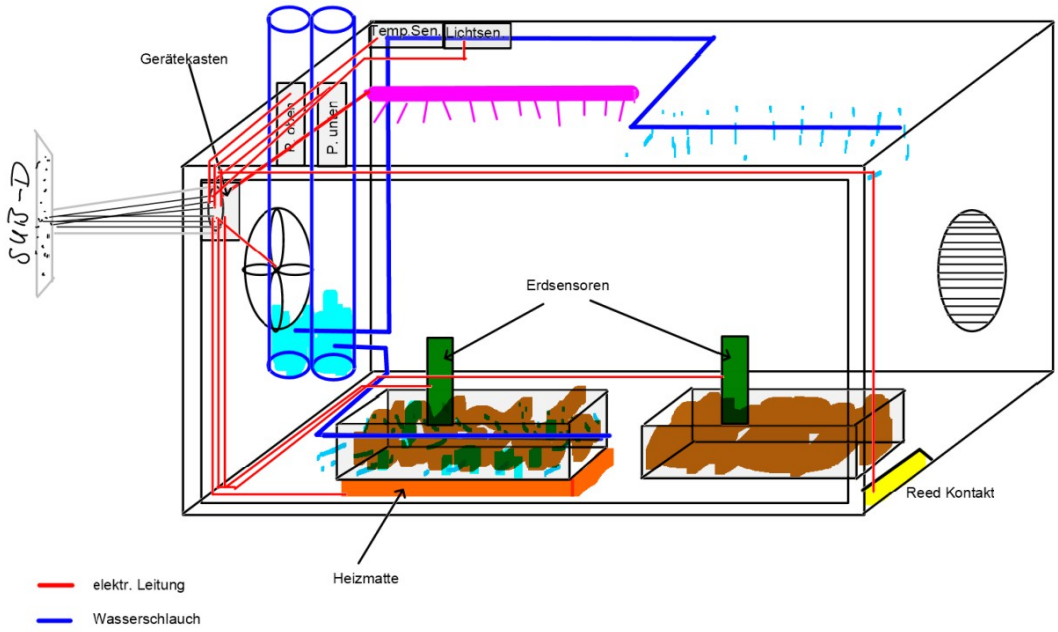

Abbildung 1: Erster Entwurf Gewächshaus

2.2 Durchführung

Nach der Planungsphase wird der Aufbau des Gewächshauses vorgenommen..
Im ersten Schritt erfolgt die Mechanische Bearbeitung.
Die Montage des Korpus und Bearbeitung der Acrylglasplatten. Hier werden Aussparungen und Löcher gesetzt unter anderem für die Belüftung und Bewässerung. Anschließend werden die entsprechenden Komponenten wie Lüfter, Schläuche, Pflanzbecken installiert.

Schritt zwei umfasst die Elektroinstallationsarbeiten.
Erstellen eines Schaltkastens für die Steuerungselektronik und der entsprechenden Verbindungselemente zum Korpus des Gewächshauses (z.B. Sub-D Stecker).

Im dritten Schritt erfolgt die Programmierung der Steuerung.
Beim vorliegenden Fall handelt es sich um einen Arduino-Mega, welcher den Erfordernissen entsprechend programmiert und konfiguriert wird.
Näheres dazu im *Kapitel 3.2.3*.

Nach Abschluss der Montage- und Installationsarbeiten erfolgen diverse Testläufe die dazu beitragen die einzelnen Funktionen beispielsweise der Sensorik zu überprüfen und gegebenenfalls anzupassen oder zu verbessern. Zudem hat man die Möglichkeit eventuell auftretende Probleme auszumerzen.

3 Aufbau

In diesem Abschnitt wird ein kurzer Überblick über verwendete Materialien, deren Funktion und Zusammenspiel aufgezeigt.

3.1 Mechanisch

3.1.1 Gewächshaus

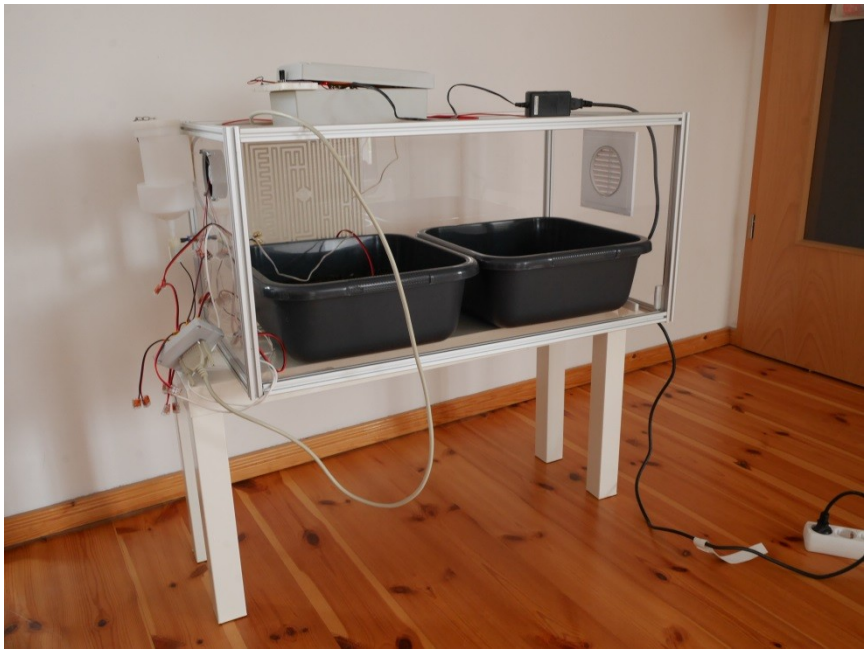

Abbildung 2: Aufbau Gewächshaus - Testphase

Zur Durchführung des Projektes wird ein Gewächshaus benötigt (siehe Abb.3) welches von allen Seiten gut lichtdurchflutet ist. Es soll sich von mindestens einer Seite öffnen lassen, ohne jedoch den Korpus vollständig entfernen zu müssen. Damit ist gewährleistet das Messwerte wie Luftfeuchtigkeit und Temperatur, innerhalb des Gewächshauses, weiterhin ermittelt und aufrechterhalten werden.

Für das vorliegende Projekt wird ein Gewächshaus mit den Maßen 800mm x 400mm x 400mm (B x H x T) mit Acrylglasscheiben und Aluminiumprofilen selbst hergestellt.
Des Weiteren werden für die Bepflanzung zwei Viereckschüsseln mit den Maßen 390mm x 150mm x 390mm (B x H x T) verwendet, welche den Vorteil bieten, dass man die Bepflanzung außerhalb des Gewächshauses vornehmen kann.

3.1.2 Bewässerungssystem

Um die Bodenfeuchtigkeit erhöhen zu können sind zwei separat voneinander getrennte Systeme integriert.

Zum einen Bewässerung von oben, welche Sprühnebel ausgibt.
Zum zweiten eine Bewässerung direkt in der Erde als Ring verlegt.
Dafür gibt es zwei getrennte Wassertanks, die mit unterschiedlichen Pumpen das Wasser mittels 6 mm² Schlauch in das Gewächshaus und anschließend in die Viereckschüsseln befördert.
Da Erde einen höheren Widerstand als Luft hat, wird für die Bodenbewässerung eine stärkere Pumpe benötigt.
Auf diese Weise hat man mehr Optionen in der Art der Bepflanzung, denn nicht alle Pflanzen vertragen eine Bewässerung von oben.

3.2 Elektroinstallation

3.2.1 Sensorik

3.2.1.1 Bodenfeuchtesensor

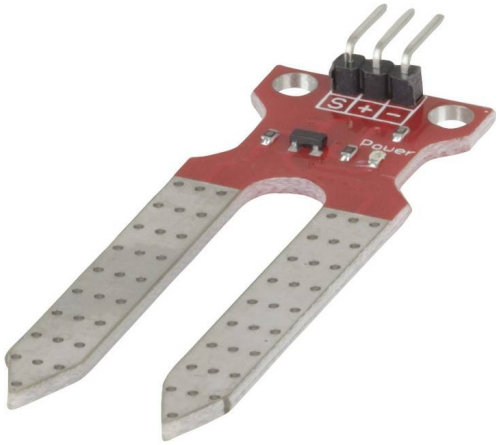

Abbildung 3: Feuchtigkeitssensor ME110

Der dargestellte Feuchtigkeitssensor (Abb. 4) wird verwendet, um die Bodenfeuchte bei Pflanzen messen zu können. Im Falle von Trockenheit wird eine der beiden elektrischen Wasserpumpen aktiviert, je nachdem, wo der Boden trocken ist.
Der Sensor wird vom Arduino mit einer 5V Spannung versorgt.
Je höher die Feuchtigkeit an den beiden Kontakten ist, desto besser kann der Strom von einem Kontakt zum anderen fließen.
Dieser Wert wird im Sensor elektronisch aufbereitet und in Form eines analogen Signals an einen analogen Eingang des Arduino übermittelt.
Da dieser keine elektrische Spannung als solche messen kann, wandelt er die am analogen Pin anliegende Spannung in einen Zahlenwert um.
0 bis 5 Volt entspricht einem Zahlenwert von 0 bis 1023 (Das sind 1024 Zahlen, da die Null als erster Zahlenwert gezählt wird).
Bei dem verwendeten Feuchtigkeitssensor liegt der obere maximale Wert jedoch beim Zahlenwert 800, wenn der Sensor komplett in Wasser eingetaucht ist. Das bedeutet, dass man beim Programmieren auf die Kalibrierung achten muss, die jedoch abhängig von der Feuchtigkeit/Flüssigkeit ist.
Da durch Wasser bzw. Feuchtigkeit an dem Sensor eine Elektrolyse stattfindet, ist es ratsam den Abstand zwischen den Messungen nicht im Sekundentakt, sondern im Abstand von mindestens 15 Minuten durchzuführen.
Es kann sonst schon nach ca. 24h zu Schäden am Sensor kommen.
[Funduino GmbH, o.J.]

3.2.1.2 Reedkontakt

Abbildung 4: Magnetkontakt MK30

Der dargestellte Magnetkontakt MK30 (Abb. 4) dient zur Öffnungsüberwachung von Türen, Fenstern, Verteilern usw.
Er besteht aus einem Reedschalter in einem zylinderförmigen Kunststoffgehäuse und einem Rundstab-Dauermagneten.
Öffnet sich die Frontscheibe des Gewächshauses, öffnet der Reedschalter und verhindert, dass das Bewässerungssystem betrieben werden kann.
[Telenot]

3.2.1.3 Temperatur- und Luftfeuchtesensor

DHT22 pins	
1	VCC
2	DATA
3	NC
4	GND

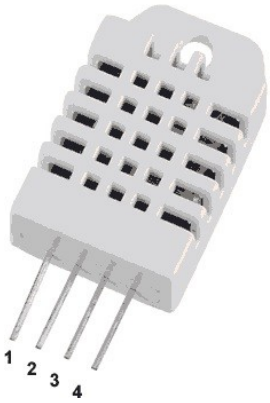

Abbildung 5: DHT22 - Temperatur- und Luftfeuchtigkeitssensor

Der dargestellte Sensor DHT22 in Abbildung 5 bietet die Möglichkeit mittels Arduino die Temperatur zwischen -40°C und 125°C mit einer Genauigkeit von 0,5°C messen zu können. Der messbare Bereich für die Luftfeuchte liegt zwischen 0% und 100% mit einer Genauigkeit zwischen 2% und 5%.

Der Arduino kommuniziert mit dem Sensor über einen digitalen Pin, welcher als Ein- und Ausgang dient. Er sendet über diesen Pin zuerst eine Anfrage an den Sensor, dieser ändert die Einstellung des Pins und der Sensor schickt seine aktuellen Werte an den Arduino zurück.

Da es sich um einen digitalen Sensor handelt, werden Temperatur und Luftfeuchtigkeit mittels 8-Bit-Auflösung übertragen.
Das heißt es werden pro Wert 8 Signale über den digitalen Pin übermittelt.
Da über digitale Pins nicht die eingehende Spannung gemessen werden kann, sondern nur die Signale HIGH (**1**) und LOW (**0**) gesendet werden, wird der entsprechende Wert in einer binären Darstellung übermittelt.
Der übermittelte Wert kann anschließend mittels der Spezifikation des Sensors in die entsprechende Temperatur oder Luftfeuchtigkeit übersetzt werden.
[ITEAD Wiki, 2015]

3.2.1.4 Helligkeitssensor

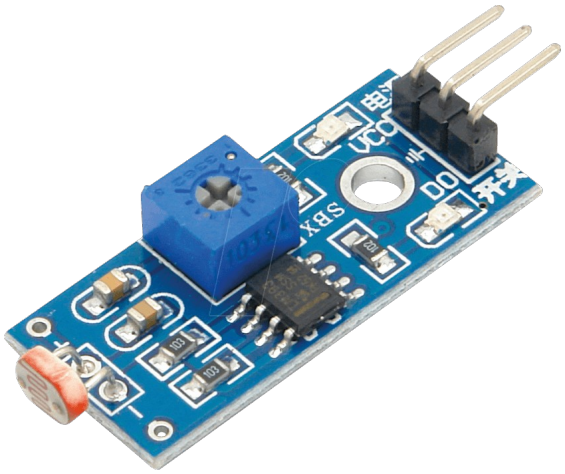

Abbildung 6: Der Helligkeitssensor

Der dargestellte Helligkeitssensor in Abbildung 6 besitzt einen Fotowiderstand auf der linken, schmalen Seite der Platine, der seinen Widerstand in Abhängigkeit vom Lichteinfall ändert.
Fällt mehr Licht ein, bei zunehmender Helligkeit, wird der Widerstandswert kleiner. Ist die Stelle an welcher der Sensor angebracht ist dunkel wird der Widerstandswert größer.

Dieser Effekt wird zusammen mit einem in Reihe geschalteten Widerstand dazu genutzt, die Helligkeit bzw. Dunkelheit anhand der am Fotowiderstand anliegenden Spannung zu ermitteln.
Bei einer Reihenschaltung teilt sich die Gesamtspannung auf die Widerstände auf und nach dem Ohmschen Gesetz fällt dabei die höchste Spannung am größten Widerstand ab.

Das heißt, dass die 5V-Spannung, die vom Arduino bereitgestellt wird, im Stromkreis des Helligkeitssensors auf den Fotowiderstand und auf den verbauten Widerstand aufgeteilt wird.
Also gilt: Je mehr Licht auf den Fotowiderstand fällt, desto geringer ist dessen Widerstand, und somit auch die an ihm abfallende Spannung.

Um die am Fotowiderstand anliegende Spannung zu messen, wird der Sensor an einen analogen Pin des Arduinos angeschlossen.
Dieser wandelt daraufhin den gemessenen Spannungswert in eine Zahl zwischen 0 (0V) und 1023 (5V) um. Der Wert ‚0' entspricht somit der höchsten messbaren und der Wert ‚1023' der geringsten messbaren Helligkeit.
[Coding World UG, o.J.]

3.2.1.5 Einspeisung Solarmodul

Abbildung 7: Solarpanel SPR-E-Flex-50

Das dargestellte Solarpanel zeigt die vorhandene elektronische Versorgung in Verbindung mit einem Akku des gesamten Gewächshauses.
Die Anlage wird als Inselbetrieb betrieben. Dazu wird das Solarpanel parallel zum Akku und zur Elektronik des Gewächshauses geschaltet.
Man spricht von einer Photovoltaik-Inselanlage, wenn der vorhandene Stromspeicher auch dann von der Photovoltaik-Anlage mit Strom versorgt und somit aufgeladen werden kann, wenn keine Verbindung (mehr) zum öffentlichen Stromnetz besteht.

Dies führt dazu, dass die Anlage einzig und allein den von den Solarmodulen produzierten Strom verbrauchen kann, ohne auf den Zukauf von normalem Strom angewiesen zu sein.
Des Weiteren bietet es den Vorteil mobil zu bleiben.
Das SunPower Solarmodul hat die Maße (B) 653mm (L) 556mm und ist maximal biegsam bis zu 30°.
Somit ist es flexibel einsetzbar. Die SunPower-Solarzelle besitzt eine Effizienz von bis zu 23%. Diese ist abhängig von Einstrahlwinkel der Sonne, Verschattung des Moduls, Umgebungstemperatur.

Solarzellen werden meist in Dick- und Dünnschichtzellen unterschieden. Für die Spannungserzeugung ist die Grundvoraussetzung ein Halbleitermaterial.

Schaut man sich einmal eine Solarzelle genauer an, kann man sagen, dass diese in drei Zonen eingeteilt werden kann.
Oben der n+ -Emitter, darunter eine Raumladungszone und unten eine Diffusionszone. Tritt also eine zugeführte Energie (elektromagnetische Strahlung der Sonne) auf die Solarzelle werden durch Photonen Elektronen aus der Diffusionszone in die Raumladungszone gehoben.
Es entstehen freibewegliche Elektronen und sogenannte Löcher in der n-Basis. Diese Elektronen werden vom positiven Pol der Diffusionsspannung in den n+ -Emitter beschleunigt. Die defekten Elektronen (Löcher) werden durch den negativen Pol der Diffusionsspannung in die p-Basis beschleunigt. Es entsteht eine Spannung, die zum Leistungsumsatz genutzt werden kann.

Da das Solarpanel eine Vmpp von 17,7V, ein Impp von 2,8A besitzt und die Elektronik über 12V versorgt wird, benötigt man für dieses System keinen Solarladeregler. Dieser hätte in diesem Fall nur den negativen Effekt die Effektivität der Anlage zu verschlechtern.

3.2.2 Aktorik

3.2.2.1 UV-LED

Pflanzen brauchen Licht zum Wachsen, steht dieses nicht in Form von Sonnenlicht zur Verfügung kann man sein Gewächshaus zusätzlich mit UV-LEDs ausstatten, die je nach Erfordernissen hinzugeschaltet werden.

Optimal wäre eine Mischung aus rotem und blauem Licht, da Photosynthese von Pflanzen hauptsächlich in den Wellenlängenbereichen zwischen 400-500 nm (blau) und 600-700nm (hellrot) stattfindet.
Für dieses Projekt wurden lediglich rote LEDs verwendet, die beispielhaft für die entsprechende UV-LED-Beleuchtung stehen.
Das Hinzu- oder Ausschalten der LEDs erfolgt über die Programmierung des Arduino in Verbindung mit dem Helligkeitssensor.
Je nach anliegender Spannung entsprechen 0,3V (**Low**) dem Zustand ‚Hell' und ‚Dunkel' entspricht einer Spannung von 5V (**High**).

3.2.2.2 Wasserpumpen

Es kommen zwei verschiedene Typen von Pumpen zum Einsatz.
Eine 5V-Pumpe mit wenig Leistung und eine 12V-Pumpe mit viel Leistung. Für die Bewässerung von der Gewächshausdecke ist die schwächere Pumpe vorgesehen. Die Pflanzen sollen von oben sachte beregnet werden, es ist also kein hoher Wasserdruck von Nöten.

Für die Bewässerung von unten wird etwas mehr Pumpenleistung benötigt, weil der im Boden verlegte Schlauch deutlich länger ist und mehr Widerstand beim Pumpen des Wassers entsteht.

Abhängig von dem Bodenfeuchtewert, der vom Arduino ausgewertet wird, setzt der Pumpvorgang ein bzw. wird eingestellt.
Die Intensität der Bewässerung wird über die Dauer des Pumpvorgangs geregelt. Zusätzlich wird der Abstand zwischen den Gießimpulsen in die Programmierung mit integriert, damit eine Überschwemmung der Pflanzen/Überwässerung des Bodens vermieden wird.

Abb.

Pumpe Oben: Typ: Werte

Abbildung 8: Zahnrad-Wasserpumpe

Pumpe Bodenbewässerung:
Die für die Bodenbewässerung eingesetzte Pumpe vom Hersteller Walfront, mit einem Spannungsbereich zwischen 3 und 12V, hat eine Fördermenge von etwa 1,2 Liter Wasser/ min bei einer Spannung von 12V.

3.2.2.3 Lüfter

Abbildung 9: Lüfter DC 12V (Ruilian Science)

Als Lüfter wird ein 12V PC-Lüfter verwendet, da er für die gegebenen Zwecke ausreicht.
Man muss beim Einbau auf die Richtung des Luftstromes achten. Je nachdem ob man möchte das Luft angesaugt oder herausgeblasen wird.

Wenn die Temperatur im Gewächshaus einen vom Nutzer im Programm festgelegten Wert überschreitet, wird der Lüfter eingeschalten und befördert die warme Luft über ein Entlüftungsgitter welches sich an der gegenüberliegenden Seite befindet aus dem Raum.
Diese Temperaturschwelle stellt man zusätzlich über ein Potentiometer ein.

3.2.2.4 Heizmatte

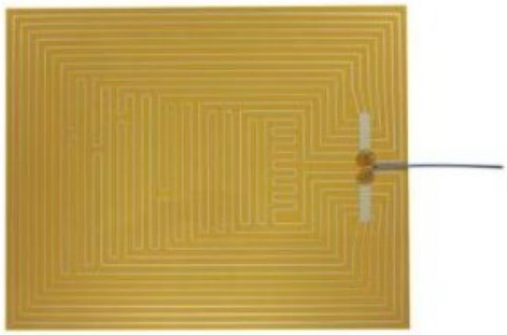

Abbildung 10: Heizmatte

Die Heizung ist eine 12V/55W-Heizfolie (von Conrad).
Sie soll den Innenraum beheizen, wenn notwendig. Es ist sinnvoll eine große Fläche zu wählen, um eine gleichmäßige Wärmeabstrahlung zu erhalten.

Bei der verwendeten Heizmatte (siehe Abbildung 8) handelt es sich um eine Folienheizung vom Typ Thermo Polyester Heizfolie selbstklebend 12 V/DC, 12 V/AC 50W Schutzart IPX4 (L x B) 500mm x 400mm.

Die selbstklebende Heizfolie aus Polyester dient dem Schutz des Gehäuses vor Minusgraden. Die Heiztemperatur der Matte ist von unterschiedlichen äußeren Einflüssen abhängig, wie zum Beispiel dem Untergrund, der Luftfeuchte und der Betriebsspannung.

Tatsächlich sollte man gut überlegen, ob die Installation einer Heizmatte/Heizung notwendig ist. Es hängt von der Verwendung des Gewächshauses ab, welche Pflanzen man dort anpflanzen und ziehen möchte. Für das hier erstellte Modell im Kleinformat dient die Heizmatte unter anderem Probezwecken für die größere Bauform und dem Frostschutz des Gehäuses. Sowie zum Beispiel dem Pflanzenschutz bei einer Frühaussaat, welche ab Februar eingebracht werden kann.

3.2.3 Arduino

3.2.3.1 Aufbau

Für den Aufbau der Steuerungselektronik des Gewächshauses bietet es sich an fertige Module, die für den Arduino angepriesen werden, zu besorgen.
Hauptargumente, die für diese Entscheidung sprechen, sind eine bereits mitgelieferte Library und zugehörige Dokumentation.
Dadurch reduziert sich der Zeitaufwand für die Elektroinstallation.
Denn je bekannter die Module sind, desto mehr Beispielcode für die Programmierung ist zu finden, den man studieren kann.

Bekannte Vertriebsfirmen sind z.B. Adafruit oder Az-Delivery.
Bei diesen Anbietern findet man Datenblätter und andere Hilfestellung zu den Produkten.
Im Allgemeinen wird Jemandem der wenig Vorerfahrung auf dem Gebiet der Mikrocontroller-Programmierung hat somit der Einstieg in das Thema erleichtert.
Für die Montage bietet es sich an Verdrahtungspläne zu erstellen *(siehe Anhang).*

Verwendet wird:

Tabelle 1:Stückliste elektronische Hardware für Arduino

Anzahl	Komponente
1x	Arduino-Mega
1x	20x4 Display mit i2c-Modul
2x	Relaismodul
1x	Tiefsetzsteller
2x	Erdfeuchtesensor-Modul
1x	Lichtsensor-Modul
1x	Temperatursensor
1x	Wasserpumpe für Bodenbewässerung
1x	Wasserpumpe für Deckenbewässerung
4x	Potentiometer
1x	Taster
1x	Hauptschalter
1x	Lüfter
1x	Heizmatte
2x	Leitungsfilter (Eigenbau)

Gehäusebox:
Beim Aufbau werden die elektronischen Komponenten in eine Kunststoffbox eingebaut, um sie vor Spritzwasser und Umwelteinflüssen zu schützen, aufgrund des Installationsortes des Gewächshauses.

Um den Arduino programmieren zu können existiert an der Vorderseite eine Abdeckung hinter der ein USB-Anschluss leicht zugänglich ist.
Die Zugänglichkeit des USB-Anschlusses ist wichtig, um die Arbeit während der Testphasen zu erleichtern und unnötiges Ausbauen oder Neuverdrahten der Komponenten zu vermeiden.
Der Arduino soll „im Einsatz befindlich“ programmiert werden können.

Um eventuelle Einstrahlung seitens der Leistungsleitungen in die Signalleitungen zu verhindern ist es sinnvoll das bei der Anordnung der Komponenten im Schaltkasten zu berücksichtigen.
Zum Beispiel den Leistungsteil auf der einen Seite und die Steuer-/Signalteile auf der entgegengesetzten Seite zu installieren.

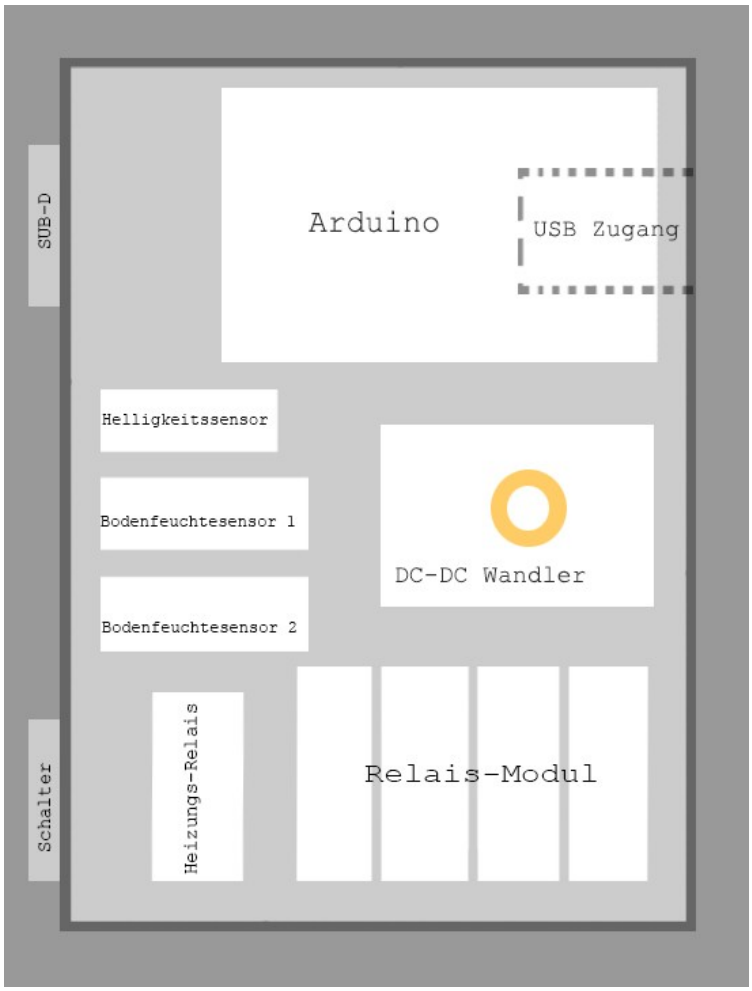

Abbildung 11: Aufbau Schaltkasten Steuerungselektronik

3.2.3.2 Arduino-Mega:

Zuerst ist es zu empfehlen sich das Datenblatt des Controllers ATmega2560 und ein Pin-Out-Diagramm zu organisieren.

Da keine Register gesetzt werden müssen für das vorliegende Projekt und die Arduino-Funktionen vieles abgedeckt haben, wurde das Datenblatt nur in geringem Maße benötigt.

Das Pin-Out-Diagramm ist wichtig, um einen schnellen Überblick über die Ports und deren Funktion zu bekommen.
Zum Beispiel welcher Port interruptfähig oder zur Pulsweitenmodulation geeignet ist.

Dislpay:
Beim Display gibt es verschiedene Optionen, hier wird ein 20x4Display verwendet. Es bietet 20 Zeichen und 4 Zeilen, um möglichst viele Parameter auf einmal anzeigen zu können.
Außerdem wird eine i2c-Ansteuerung statt einer parallelen Ansteuerung gewählt, um Leitungen zu sparen.
Dazu wird ein i2c-Adapter verwendet, welcher 8 Datenleitungen, zu zwei Leitungen kompensiert, einer Takt- und einer Datenleitung.
Die Module verwenden einen IC „NXD PCF 8574T“.
Das ist ein Portexpander von parallel zu i2c.

Relais-Module:
Verwendet werden 5 Relais. Ein Vierer-Modul und ein einzelnes Relaismodul. Das Modul wird mit 5V versorgt und über den Arduino werden die Relais an- und ausgeschaltet.
Die hier verwendeten Module haben eine Besonderheit.
Sie sind Low-aktiv.
Will man also das Relais einschalten, muss es auf **Low** gesetzt werden und will man es ausschalten, muss man es auf **High** setzen.
Das ist am Anfang gewöhnungsbedürftig, da es entgegen der üblichen Logik ist.

Tiefsetzsteller:
Der Tiefsetzsteller macht aus der 12V Eingangsspannung 5V. Mit diesen 5V wird der Arduino versorgt und die komplette Sensorik. Über das Potentiometer stellt man die gewünschte Spannung ein.

Lichtsensor-Modul:
Für den Lichtsensor wird eine Komperator-Schaltung mit dem IC LM393 und einem Fotowiderstand genutzt. Der LM393 ist ein low-power Dual-Komperator. Er schaltet bei einer einstellbaren Schwelle um zwischen **High** und **Low**.
‚Hell' entspricht einer Spannung von 0,3V (Low) und ‚Dunkel' entspricht 5V (High). Der Arduino wertet dies aus und das Licht ändert seinen Zustand „AN/AUS" je nach Vorgabe.

Erdfeuchtesensor-Modul:
Der YL-69-Feuchtesensor nutzt ebenfalls eine Komperator-Schaltung mit dem IC LM393. Aber statt einem Fotowiderstand, werden hier zwei Elektroden genutzt über die eine Spannung anliegt.
Sind diese Elektroden in der Luft, ist der Widerstand unendlich groß und es fließt kein Strom. Steckt man diese Elektroden in ein Medium z.B. Erde, fließt ein kleiner Strom. Ist die Erde feucht, fließt ein größerer Strom.
Dieses Verhalten wird am analogen Port des Arduino , welcher ein ADC ist, durch das Programm ausgewertet und schaltet je nach Trockenheit des Bodens, die Wasserpumpe zu oder aus.

Temperatursensor:
Als Temperatursensor wird der DHT22 verwendet.
In diesem Modul ist ein Temperatursensor und ein Luftfeuchtesensor in einem enthalten. Mit der dazugehörigen Library ist er einfach auszuwerten. Mit dem DHT22 wird die Temperatur im Gewächshaus gemessen.

Potentiometer:
Die Potentiometer haben einen Wert von 10kOhm.
Sie sind an den analogen Eingängen angebracht, hinter dem sich ein ADC befindet. Hier wird die Schleiferstellung ausgewertet und im Programm können die Schwellenwerte eingestellt werden.

Taster:
Der Taster schaltet das Display um.
Um einen definierten Pegel am Eingangs-Pin zu haben, wird der interne Pullup-Widerstand benutzt.

Leitungsfilter:
Der Leitungsfilter ist sehr wichtig für die korrekte Funktionsweise der Steuerung. Er muss so nah wie möglich an die Wasserpumpen angeschlossen werden, um eine Einbringung von hochfrequenter Spannung ins System zu verhindern.
Er besteht aus zwei Spulen mit je100µH, einem Widerstand von 1kOhm, einem Keramikkondensator 100nF und einer Freilaufdiode für den Elektromotor der Wasserpumpe.
(Eventuell Grafik Aufbau?)

Lüfter:
Siehe Kapitel 3.2.2.1

Wasserpumpe:
Siehe Kapitel 3.2.2.2

Heizmatte:
Siehe Kapitel 3.2.2.4

3.3 Funktionen

Der Arduino ist das Herzstück der Steuerung. Er liest die Daten der Sensoren ein und gibt entsprechende Befehle an die Aktoren.

Hauptfunktion ist es die Pflanzen bei zu trockenem Boden zu wässern.
Über die Erdfeuchtesensoren erkennt der Arduino wie trocken der Boden ist.
Ist er zu trocken, wird die Wasserpumpe eingeschaltet.
Die Schwelle, ab wann der Boden zu trocken ist, stellt der Anwender selbst ein.
Das kann man beispielsweise manuell über die Potentiometer tun.

Man stellt den Zustand her, den der Boden haben soll wenn er zu trocken ist und stellt über das Potentiometer die Schwelle so ein, dass die Pumpe anfängt zu wässern.

Ist es im Gewächshaus zu warm schaltet der Arduino einen Lüfter ein, der die warme Luft über ein Entlüftungsgitter nach draußen befördert.
Ist es zu kühl für die Pflanzen, schaltet sich die Heizung zu, um den Raum zu wärmen. Entsprechende Temperaturwerte besorgt man sich zum Beispiel vom Botaniker seines Vertrauens.

Pflanzen benötigen Licht. Dazu wertet der Arduino einen Fotowiderstand aus und ab einer gewissen einstellbaren Schwelle, geht im Gewächshaus UV-Licht an, damit die Pflanzen optimale Wachstumsbedingungen haben.

Eine Sicherheitsfunktion ist auch integriert, um einen größeren Wasserschaden zu vermeiden. Sollte die Frontplatte offen sein, kann die Wasserpumpe nicht pumpen. Hier wird ein REED-Kontakt als Sicherheitsschalter verwendet.

Des Weiteren gibt es einen Taster.
Dieser wird benötigt, um das Display umschalten zu können.
Dieses hat vier Zeilen, es werden für dieses Projekt allerdings acht Zeilen gebraucht. Daher werden zwei Ebenen eingerichtet, zwischen denen mit dem Taster umgeschaltet werden kann.
In der ersten Ebene, die auch standartmäßig beim Start angezeigt wird, sind die Ist-Werte zu sehen, wie Temperatur und Erdfeuchte.
Ebene zwei zeigt die Soll-Werte an, also die einstellbaren Schwellen.

3.4 Programmierung

Es gibt mehrere Vorgehensweisen für die man sich entscheiden kann, um den Arduino zu Programmieren.
In diesem Fall kommt die gebräuchlichste und einfachste Methode zum Einsatz: die Benutzung der Arduino-IDE.

Die Arduino-IDE ist sehr spartanisch, wenn man sie mit Atmel-Studio oder Eclipse vergleicht, dafür ist sie sehr einfach gehalten und man findet sich auch ohne große Vorkenntnisse gut zurecht.

Man muss nur drei Sachen beherrschen:
‚Überprüfen' und ‚Hochladen', sowie ‚Speichern' und unter Werkzeuge-Board das richtige Arduino-Board auswählen.
Sobald man das verinnerlicht hat, kann man mit der Programmierung beginnen und Testen.
Ein weiterer Vorteil, man muss nichts Einrichten, es funktioniert alles „Out of the box".

3.4.1 Die Hauptdatei

Ein kleiner Exkurs:
In C ist die Haupdatei immer die Main-Datei.
Beim Arduino hat man für gewöhnlich keinen Zugriff auf die „echte" Main-Datei.
Es gibt sie unter:

$HOME/arduino/hardware/arduino/avr/cores/arduino/main.cpp

Aber diese Main-Datei ist schon vorgegeben und man sollte sie nicht verändern.

Die main.cpp enthält zwei Funtionen „*setup()*" und „*loop()*".
Legt man in der Arduino-IDE nun sein Projekt an, hier heißt es „abschlussprojekt.ino", sind beim ersten Start nur diese zwei Funktionen zu sehen, ansonsten ist die Datei leer.

Es müssen auch beide Funktionsnamen im eigenen Projekt stehen, sonst bekommt man vom Debugger später eine Fehlermeldung.
Die Befehle die im *Setup()* stehen, werden einmalig ausgeführt und die im *loop()* stehen, werden in einer Endlosschleife ausgeführt

Abschlussprojekt.ino:

Was passiert in dieser Datei?
Als erstes werden externe Libraries und ausgelagerte Funktionen inkludiert.
Hier ist zu erwähnen, dass für den Temperatursensor und für das i2c-LCD erst die richtige Bibliothek heruntergeladen werden muss.

Unter Windows war das einfacher, da man unter ‚*Sketch-Bibliothek einbinden-Bibliotheken verwalten*' alles bequem per *wizard* erledigen konnten.
Da aber unter Linux gearbeitet wird, muss das Projekt ausfindig gemacht werden, die Biliothek bei *Github* herunterladen und manuell in den Ordner

$HOME/Arduino/library/

kopiert werden.
Dann ist alles richtig eingerichtet und man kann unter Linux und Windows gleichermaßen arbeiten.

Danach werden Variablen deklariert und einige auch initialisiert.
Dabei sollte auf sprechende Namen geachtet werden.
Sprechende Namen bedeutet, dass man schon durch lesen des Variablennamens erkennen kann, was die Variable für eine Aufgabe/Funktion hat. Man sollte sich für die sprechenden Namen Zeit nehmen und alles genau durchdenken, dadurch spart man sich später Arbeit, in Form von nachträglichen Änderungen.
Man kann beispielsweise den Entwurf der ersten Codezeilen zuerst auf Papier schreiben, gegebenenfalls streichen und neuschreiben. Sprechende Namen sorgen im Idealfall auch dafür, dass man seinen Programmcode schon beim Lesen versteht und sich Kommentare sparen kann.

Dann kommen die Funktionsdefinitionen.
Beim modularisieren des Projekts, werden alle selbstgeschriebenen Funktionen der Übersichthalber in extra Dateien geschrieben, welche am Anfang der Datei inkludiert werden.

Es folgt der *Setup*, die erste eigentliche Arduino-Funktion.
Hier wird das Display initialisiert und eine erste LCD-Ausgabe getätigt.
Der Temperatursensor wird initialisiert. Dann werden die Pins konfiguriert.
Dazu verwendet man die Methode pinMode() und über das Argument INPUT oder OUTPUT setzt man sie als Ein- oder Ausgang.
Für den Taster und den REED-Kontakt werden die internen PullUp-Widerstände aktiviert und ersparen so eine externe Beschaltung.

Der *Timer* für die Zeiten wird gesetzt.
Hierzu wurde die Funktion *millis()* genutzt.
Wenn man das Arduino-Board einschaltet startet auch ein Timer, der die Zeit in Millisekunden zählt. Durch Verwendung von *millis(),* kann man den Wert abrufen beziehungsweise in eine Variable schreiben und den Wert verarbeiten.
Durch genaues Lesen der Arduino-Referenz erfährt man, dass die Funktion nur 50 Tage zählt und dann wieder auf Null zurückgesetzt wird.
Das sollte man im Hinterkopf behalten.

Zum Schluss kommt noch eine LCD-Ausgabe „`...Complete`“.
Es ist gleichzeitig eine Überprüfung, ob alles korrekt im Setup durchlaufen wird.
Erscheint „`Complete`“ nicht, hängt etwas im Setup fest und man muss eine Fehlersuche durchführen.

Die zweite Arduino-Funktion „*loop*“. Hier werden die Aufgaben programmiert, die endlos abgearbeitet werden sollen.

Zuerst werden die Potentiometer eingelesen, die an den analogen Ports angeschlossen sind. Mit ihnen stellt man die Schwellen ein.
Näheres im *Kapitel 3.2.3.2* - Potentiometer.

Nun wird der Taster-Zustand eingelesen.
Wird der Taster gedrückt, ändert sich das Verhältnis zwischen *vorheriger_taster* und *aktueller_taster* und die Nummer des Displays schaltet zwischen **0** und **1** hin und her.

Als nächstes erfolgt die Display-Ausgabe.
Wenn 2 Sekunden verstrichen sind (dt_display_ms = 2000) geht man in die Anweisung und ändert die Display-Anzeige.
In der nächsten Anweisung liest man die Messwerte der Sensoren ein (Temperatur, Erdfeuchte und Luftfeuchte).
Danach wird die Raumtemperatur mit der Schwellentemperatur verglichen, ist die Raumtemperatur zu niedrig, geht die Heizung an.
Ist die Temperatur zu hoch, geht die Heizung aus.
Das gleiche Verfahren wird beim Lüfter angewandt, bloß umgekehrt.
Es ist aber auf eine gewisse Totzeit zwischen Heizen und Lüften zu achten, da sonst der Lüfter gegen die Heizung arbeitet. Das wäre kontraproduktiv und Energietechnisch gesehen katastrophal.

Danach wird der Helligkeitssensor ausgewertet.
Ist es draußen Dunkel gehen die LEDs an, sonst sind sie aus.

Zum Schluß werden die Pumpenfunktionen aufgerufen.
Es sind zwei, eine für die Bewässerung von oben und eine für die Bewässerung von unten. Diese Pumpkreise arbeiten unabhängig voneinander.

3.4.2 Die Unterprogramme

Um den Code übersichtlicher zu gestalten, wird das Programm modularisiert.
Im Folgenden sind die einzelnen ausgelagerten Funktionen aufgeführt:

```
makros.h
make_string.h
make_string.cpp
pumpe_1.h
pumpe_1.cpp
pumpe_2.h
pumpe_1.cpp
tuer_zu.h
tuer_zu.cpp
update_lcd.h
update_limits.h
update_limits.cpp
update_messwerte.h
```

Dazu wurden zwei Dateien erstellt. Eine CPP-Datei in welcher der Code steht, die Funktionsdefinition. Und eine Header-Datei in welcher der Prototyp steht, die Deklaration.
Die Arduino-IDE unterstützt nur .CPP Dateien. Es reicht an dieser Stelle nicht, eine .C-Datei zu erstellen, da bekommt man eine Fehlermeldung. Die Dateien „update-messewerte“ und „update_lcd“ existieren nur als Header-Datei. Möchte man nach dem gleichen Verfahren hier eine .CPP und .H-Datei erstellen, bekommt man eine Fehlermeldung, dass in der Hauptdatei bestimmte Objekte fehlen. Um die Funktion trotzdem auslagern zu können, wurde an dieser Stelle der C-Standard umgangen und der komplette Code in die Header-Datei geschrieben.

Auch hier sollte wieder auf sprechende Dateinamen geachtet werden, damit sofort klar wird welche Funktion in der Datei steckt.

In der Datei `makros.h` sind alle Makros enthalten, welche im Wesentlichen die Pin-Belegung des Arduino durch Makronamen ersetzen, die dann wiederum im

Code stehen. Dadurch erkennt man sofort, welche Hardware an welchem Pin angeschlossen werden muss.

Die Datei `make_string.cpp` misst die Länge des Strings, der auf unserem Display ausgegeben wird.
Ist der String kürzer als zwanzig Zeichen, wird hinten solange ein Leerzeichen angefügt, bis die Anzahl zwanzig erreicht ist.
Somit können dynamisch alte Zeichen gelöschen werden, wenn das Display aktualisiert wird.

`Pumpe_1.cpp` und `Pumpe_2.cpp` sind vom Aufbau identisch.
Der Ablauf des Pumpens wird hier durch einen Zustandsautomaten (case-Struktur) gelöst.
Das Problem ist, die Einsickerzeit zu berücksichtigen, denn das Wasser braucht Zeit um sich im Boden zu verteilen.
Im ersten Zustand (`zustand_warten_zu_trocken`) wird geprüft, ob die Tür zu ist und die Erde feucht oder trocken ist. Ist die Erde feucht, springt man heraus aus der ‚`case`' und kommt im nächsten Zyklus wieder im ersten Zustand an.
Ist es diesmal trocken, springt die Pumpe an und der nächste Zustand (`zustand_pumpt`) wird gesetzt.
Hier wird geprüft, ob die Zeit welche die Pumpe laufen soll, vorüber ist (`dt_pumpendauer_1_ms = 5000;`), oder ob die Tür in der Zwischenzeit aufgemacht wurde.

In beiden Fällen wird die Pumpe ausgeschaltet und der nächste Zustand wird gesetzt (`zustand_warten_durchfeuchtung`).
In diesem und letzten Zustand wird gewartet, dass sich das Wasser im Boden richtig verteilt. Diese Zeit kann man im Code einstellen.
Für dieses Projekt wird hier eine Zeit von 10 Sekunden gewählt (`dt_warten_durchfeuchtung_1_ms = 10000`).

Die Funktion `tuer_zu.cpp` wertet den REED-Kontakt aus.
Ist der Kontakt geschlossen wird der Arduino-Pin „REEDKONTAKT“ gegen Masse gezogen und die Funktion liefert ein „*true*“ zurück, wenn nicht ein „*false*“.
Das wird dann in den Pumpenfunktionen ausgewertet. Für den Arduino-Pin „REEDKONTAKT“ verwenden wir an dieser Stelle den internen Pullup-Widerstand, um einen definierten Pegel zu haben (und auch nicht mit dem Brumfinger zu schalten).

Die letzten drei Dateien, `update_lcd.h`, `update_limits.cpp` und `update_messwerte.h`, machen im Prinzip dasselbe.

Sie werden in jedem Programmzyklus abgefragt. `Update_limits` wird immer abgefragt und `update_messwerte` und `update_lcd` nur, wenn eine gewisse Zeit abgelaufen ist.

Interessant war die *map*-Funktion. Damit wurden die ADC-Werte des Arduino in lesbare Werte, in Grad Celsius-Werte umgewandelt (*gemappt*).

Die zweite wichtige Sache ist die Verwendung von „extern“ bei den Variablennamen. Durch Verwendung des Deklarationsspezifizierers „*extern*“ kann man gleiche Variablennamen mehrfach verwenden.
[„Mikrocontroller verstehen und Anwenden“, Clemes Valens]

https://www.roboternetz.de/community/threads/65187-Tutorial-Erstellen-einer-Arduino-Bibliothek

4 Fazit

Kritische Auseinandersetzung mit Aufgabenstellung, was wurde geschafft, was nicht, und warum nicht (Probleme etc.)

EMV-Problem bei den Wasserpumpen, beim Einschalten der Pumpen über ein Relais, schaltete sich der Arduino aus. Bei Fehlersuche ergab sich, immer wenn das Kabel in der Nähe des µC lag, strürzte er ab, lag es weit weg, dann funktionierte es gut. Nach Messungen mit dem Oszi, konnte man die HF-Störung sehen und diese mit einem Leitungsfilter (Spule im Hin- und Rückzweig, Keramikkondensator parallel, Entladewiderstand) soweit reduzieren, das sie keinen Einfluß auf die Arbeit des Arduino mehr haben. Quelle: „EMV-Störungen/Prozessankopplung“ Skript von Kurt Jankowski-Tepe (S.21) Da die Heizmatte 4,5A zieht, brauchen wir einen extra Stecker und extra Netzteil, die SubD-Stecker dürfen max mit 1A belastet werden.

Es gibt bei den von uns verwendeten Erdfeuchtesensoren ein großes, aber bekanntes Problem: die elektro-chemische Metallabscheidung der Elektrode durch den galvanischen Effekt oder einfach ausgedrückt, der Sensor löst sich nach kurzer Zeit auf. Die Elektroden sind nur zwei Leiterbahnen aus einer 35µm dicken Kupferschicht. Dabei wandern die Kupferelektronen der Minuselektrode zur Pluselektrode. Irgendwann ist die Minuselektrode so weit abgetragen, dass man den Erdfeuchtesensor austauschen muss. Wird der Erdfeuchtesensor dauerhaft bestromt, ist dieser schon nach wenigen Tagen erudiert, das ist schlecht. Uns sind zwei Lösungen eingefallen, die wir aber noch nicht realisiert haben. Lösung 1: Den Sensor nur bestromen, wenn er gebraucht wird. Das müßte man per Software erledigen. Den Sensor bestromen, dann messen, dann den Sensor wieder abschalten. Lösung 2: Nach Altervativen umschauen. Es gibt einen kapazitiven Erdfeuchtesensor von at- delivery → den Bodenfeuchtsensor „Modul V1.2 kapazitiv“.

Quelle: http://www.kupferspirale.info/mikro-galvanischer-effekt
https://www.wotech-technical-media.de/womag/lexikon/Metallabscheidung/Allgemeines/ Metallabscheidung_galvanische_Besonderheiten_5444.php

https://www.az-delivery.de/products/bodenfeuchte-sensor-modul-v1-2

Wir mußten auch das Problem bewältigen, dass einer die Elektronik bei sich zu Hause hatte und ein Anderer das Gewächshaus in dem sich die Sensorik und Aktorik befand bei sich zu Hause. Hier mußte man sich überlegen, wie man Änderungen am Programm vornehmen kann und gleichzeitig die Änderungen am Gewächshaus sehen kann, ohne ständig eine Entfernung von Erkner nach Karlshorst zurückzulegen. Es mußte der Bau eines Programmieradapters, der

das Gewächshaus simuliert, erfolgen. Dazu haben wir einen weiteren 25pol SubD-Stecker verwendet. Der REED- Kontakt wurde durch einen Schalter ersetzt, die Pumpen, der Lüfter, die UV-Beleuchtung durch LED‘s, die den Schaltzustand anzeigen konnten, einem Fotowiderstand und einem Temperatursensor. Das ganze wurde einmal abgeglichen und von da an konnten Tests und Änderungen an einer Stelle vorgenommen werden.

- Wir starteten mit dem Arduino Uno, aber da waren wir schnell an die Grenzen gekommen, was die Anzahl der Ports betrifft. Deshalb sind wir zum nächst größeren Modell übergegangen,

5 Ausblick

Mögliche Erweiterung und Anwendungen

(Auch wenn schon einiges vorhanden ist, kann man sicherlich noch einiges mehr hinzufügen. Mir sind folgende Punkte eingefallen, wovon ich bei Bedarf auch das ein oder andere umsetzen kann. Wer weitere Ideen hat, kann gerne einen Kommentar hinterlassen.

- **Genauere Erdmessung** mit Xiaomi Mi Flora: Die Feuchtigkeitssensoren haben den Nachteil, dass sie leicht erodieren und nicht all zu genau sind. Inzwischen hat Xiaomi einen Pflanzensensor vorgestellt, den MiFlora, der kabellos viele weitere Daten erheben kann.
- **Wasserfüllstand**: Zwar bewässern wir die Pflanzen automatisch, allerdings muss immer noch per Hand Wasser in den Eimer nachgegossen werden. Mit einer Füllstandsmessung könnte man sich z.B. per Telegram App benachrichtigen, sobald Wasser nachgefüllt werden muss. Alternativ kann man auch eine Warnleuchte (LED) anbringen.
- **Luftfeuchte messen**: Der DHT11 bzw. DHT22 Sensor kann neben der Temperatur auch die Luftfeuchtigkeit messen. Dies nutzen wir allerdings im Moment nicht. Denkbar wäre, dass man die Luftfeuchtigkeit mit Wasserspritzern auf ein bestimmtes Level bringt. Wobei dies natürlich abhängig von den Pflanzen ist.
- **Heizen**: Gerade im kälteren Frühjahr oder im Herbst kann es auch mal kühler werden. Um optimale Verhältnisse für die Pflanzen zu schaffen, kann man auch die Luft im Gewächshaus heizen.
- **Aktiv kühlen**: Bisher kühlen wir nur passiv (Fenster auf/zu). Durch einen Lüfter könnten wir bei hohen Temperaturen zusätzlich auch noch aktiv kühlen.)

➔ Füllstandsanzeige?

6 Quellen

Abbildung 3: Bodenfeuchtesensor
https://www.conrad.de/de/p/bodenfeuchtesensor-me110-iduino-me1101616242.html?WT.mc_id=google_pla&WT.srch=1&ef_id=EAIaIQobChMIuc6p0cu65QIViiaCh1Eog3SEAQYASABEgLm5vD_BwE:G:s&gclid=EAIaIQobChMIuc6p0cu65QIViiaCh1Eog3SEAQYASABEgLm5vD_BwE&hk=SEM&s_kwcid=AL!222!3!367270211499!!!g!!

Abbildung 4: Reedkontakt
https://www.telenot.com/fileadmin/user_upload/glossar/MK30-Datenblatt_100091712.pdf

Abbildung 5: Luftfeuchtesensor
https://www.electroschematics.com/arduino-dht22-am2302-tutorial-library/

Abbildung 6: Helligkeitssensor
https://www.reichelt.de/entwicklerboards-lichtsensor-mit-high-low-ausgang-lm393-debo-light-sens-p224222.html?PROVID=2788&gclid=EAIaIQobChMI8bK0956S3AIVDdwZCh2Nrw96EAQYAyABEgL0XfD_BwE&&r=1

Abbildung 7: Solarpanel
https://www.sunpower.de/sites/default/files/2019-10/sp_E_Flex_50W_UK.pdf

Abbildung 8: Heizmatte
https://www.digitalo.de/products/645685/Thermo-Polyester-Heizfolie-selbstklebend-12-V-DC-12-V-AC-50W-Schutzart-IPX4-L-x-B-500mm-x-400mm.html

3.2.3.1 Katalog für elektronische Module
https://www.az-delivery.de/
https://www.adafruit.com/

3.2.2.1 UV-LED
https://www.hswt.de/forschung/wissenstransfer/2013/april-2013/belichtung.html

3.4.2 Die Unterprogramme
https://www.mikrocontroller.net/articles/Umstieg_von_Arduino_auf_AVR
https://www.arduino.cc/reference/de/language/functions/time/millis/

6.1 Literaturverzeichnis

Im aktuellen Dokument sind keine Quellen vorhande

6.2 Abbildungsverzeichnis

6.3 Tabellenverzeichnis

7 Anhänge

Bildmaterial
Tabellen
Protokolle
Stücklisten
Programmquelltexte (Courier New)
Selbstständigkeitserklärung