E-Technik Bericht

Ole Stein, Glen, Daniel Holle

June 2024

Chapter 1

Aufbau

1.1 Übersicht

Dieser Bericht beschreibt unser diesjähriges E-Technik Praxisprojekt. Hierbei handelt es sich um die Steuerung eines Hydroponik-Systems unter Verwendung eines Arduino-Microcontrollers.

1.2 Grundlagen eines Hydroponik-Systems

Eine sogennante Hydrokultur, oder Hydroponic System, ist eine Art des Pflanzenanbaus. Hierbei wird anstatt von Erde ein inertes Wachstumsmedium und eine Nährstofflösung genutzt. Durch genaue Kontrolle verschiedener Faktoren lässt sich eine ideale Umgebung für schnelles Wachstum erschaffen.

1.2.1 Nutrient Film Technique

Unser Projekt verwendet die Nutrient Film Technique, kurz NFT. Diese nutzt einen dünnen, kontinuierlich fließenden Film aus Nährstofflösung, der am Boden eines Rohres verläuft, um die Wurzeln der Pflanzen mit Nährstoffen zu versorgen. Diese befinden sich in mit einem Wachstumsmedium (z.B. Tonkügelchen) gefüllten Korb in der Decke des Rohres.

Durch den simplen Aufbau ist dieses System bestens für die Verwendung zuhause geeignet.

1.3 Sensorik

Relevant für eine gute Wachstumsumgebung sind folgende Werte:

- Umgebungstemperatur
- Luftfeuchtigkeit

- Wassertemperatur
- PH-Wert des Wassers
- PPM-Wert des Wassers(um die Menge der Nährstoffe zu bestimmen)

Um diese Werte zu bestimmen, wird eine Sammlung von Sensoren genutzt.

1.3.1 Umgebungstemperatur und Luftfeuchtigkeit

Umgebungstemperatur und Luftfeuchtigkeit werden von einem weit verbreiteten DHT-22/AM2302 Sensor gemessen. Bei diesem handelt es sich um einen digitalen Temperatur- und Luftfeuchtigkeitssensor der eine einzelne Serielle Schnittstelle hat.

1.3.2 Wassertemperatur

Die Wassertemperatur wird mit einem DS18B20 Temperatursensor gemessen. Auch dieser ist ein digitaler Sensor und verfügt über einen "One-Wire-Bus" der es ermöglicht mehrere dieser Sensoren an den selben Datenpin anzuschließen.

1.3.3 PH-Wert

Um den PH-Wert zu bestimmen, wird ein analoges PH-Messgerät mit einer Glaselektrode genutzt. Eine an dieser stattfindende Halbzellenreaktion erzeugt eine Potentialdifferenz zu einer in einer Referenzlösung aufbewahrten Elektrode. Diese kann gemessen werden, und hängt größtenteils linear mit dem PH-Wert zusammen.

1.3.4 PPM-Wert

Die im Wasser gelösten Teilchen lassen sich über den Leitwert des Wassers bestimmen. Um diesen zu berechnen wird ein selbstgebautes EC-Messgerät genutzt. An einer späteren Stelle in diesem Bericht wird die Funktionsweise genauer erläutert.

1.3.5 Wasserstand

Abschließend werden mehrere Feuchtigkeitssensoren genutzt, um die Füllstände der verschiedenen Behälter zu bestimmen. Hierbei kommen Sensoren verschiedener Bauweise zum Einsatz.

Bauweise A

Diese Bauweise nutzt einfach nebeneinandergelegene Leiter mit sehr geringem Abstand. An einen dieser Leiter wird eine Spannung angelegt, welche gemessen wird. Der andere wird mit dem Erdleiter verbunden. Im trockenen Zustand genügt der Abstand zwischen den Bahnen um diese zu isolieren. Kommt

nun Wasser in Kontakt mit beiden Leitern entsteht ein Kurzschluss und die gemessene Spannung fällt plötzlich ab.

Bauweise B

Auch dieser Aufbau nutzt nebeneinander liegende Leiterbahnen, allerdings wird außerdem noch ein Transistor verwendet. Die unter Spannung liegende Leiterbahn ist diesmal mit dem Emitter-Anschluss des Transistors verbunden, der an Erde liegende Leiter mit dem Base-Anschluss und der Signalausgang mit dem Collector. Wenn nun eine Verbindung zwischen den Leiterbahnen entsteht wird der Transistor durchlässig und die Eingangsspannung kann gemessen werden.

1.3.6 Display

Ein kleines OLED-Display wird zur Ausgabe der wichtigsten Werte (PH, PPM, Temperatur) genutzt.

Chapter 2

Anschluss und Steuerung des Arduinos

2.1 Anschlüsse

Die digitalen Sensoren (Temperatursensoren) werden direkt an die digitalen Pins des Arduinos angeschlossen. Des weiteren wird die Spannungsversorgung des PPM-Sensors, sowie die verschiedenen Transistoren und Relais zur Kontrolle der Pumpen und des Lichtes, an digitale Pins angeschlossen.

Die analogen Sensoren (PH-,PPM-,Wasserstandsensoren) werden dementsprechend mit den analogen Eingangs-Pins verbunden.

Das Display verfügt über eine I2C-Schnittstelle, welche hier auch verwendet wird.

Alle Sensoren und das Display erhalten ihre Strom- und Spannungsversorgung vom Arduino selber, da keine hohen Ströme erforderlich sind.

2.2 Steuerung

Der Code zur Kontrolle das Arduinos kann unter

`https://github.com/CatraMyBeloved/E_Tech/blob/main/complete/complete.ino`

gefunden werden.

In diesem bericht sollen nun einige Aspekte dieses Codes genauer erläutert werden.

Main Loop

In der "loop"-Funktion des Arduinos befindet sich Code, welcher wiederholt ausgeführt wird. Zuerst wird die momentane Zeit mithilfe der eingebauten realtime clock (RTC) überprüft, um bei Bedarf den das Licht-Relais steuernde Signal zu ändern.

Listing 2.1: Abfrage der RTC und Aufruf der control lights Funktion

```
// Use real-time clock to control PIN_LIGHTS, get current time
    RTCTime currentTime;
    RTC.getTime(currentTime);

// Get current hour
    int currentHour = currentTime.getHour();

// Control lights based on current hour
    control_lights(currentHour);
```

Listing 2.2: control lights Funktion

```
void control_lights(int currentHour) {
// Turn off lights if current hour is outside the range
  if ((currentHour < LIGHTS_ON_HOUR || currentHour >
  LIGHTS_OFF_HOUR) && light_state == 1) {
    digitalWrite(PIN_LIGHTS, LOW);
    light_state = 0;
  }
// Turn on lights if current hour is within the range
  else if ((currentHour >= LIGHTS_ON_HOUR && currentHour <=
  LIGHTS_OFF_HOUR) && light_state == 0) {
    digitalWrite(PIN_LIGHTS, HIGH);
    light_state = 1;
  }
}
```

Die Variable `light_state` speichert den momentanen Status des Pins, damit dieser nicht bei jedem Loop aktualisiert wird. Als nächstes werden die verschiedenen Messwerte erhoben, und in einem Array eingetragen.

Listing 2.3: Mess- und Arrayeintrag(Beispiel)

```
ppm = get_PPM();
measured_values[3] = ppm;
```

Danach werden die Werte mit vorher definierten "glaubhaften" Werten abgeglichen, um sicherzustellen das es keine fehlerhaften Sensordaten gibt.

Listing 2.4: "Validate" Funktion

```
void validate(float measured_values[]){
  for(int i = 0; i < NUMBER_SENSORS; i++){
    if(measured_values[i] >= valid_values[i][0] && measured_values[i]
    <= valid_values[i][1]){
      validated[i] = 1;
    }
  }
```

Als letztes wird überprüft, ob die steuerbaren Werte eine Anpassung nötig haben. Falls dies der Fall ist, werden die jeweiligen Steuer-Funktionen aufgerufen.

Listing 2.5: Aufruf Steuer-Funktionen

```
    // Check if pH needs adjustment
  if (ph > PH_MAX) {

    adjust_PH();

    return;
  }

  // Check if PPM needs adjustment
  if (ppm < PPM_MIN) {

    adjust_PPM();

    return;
  }
```