

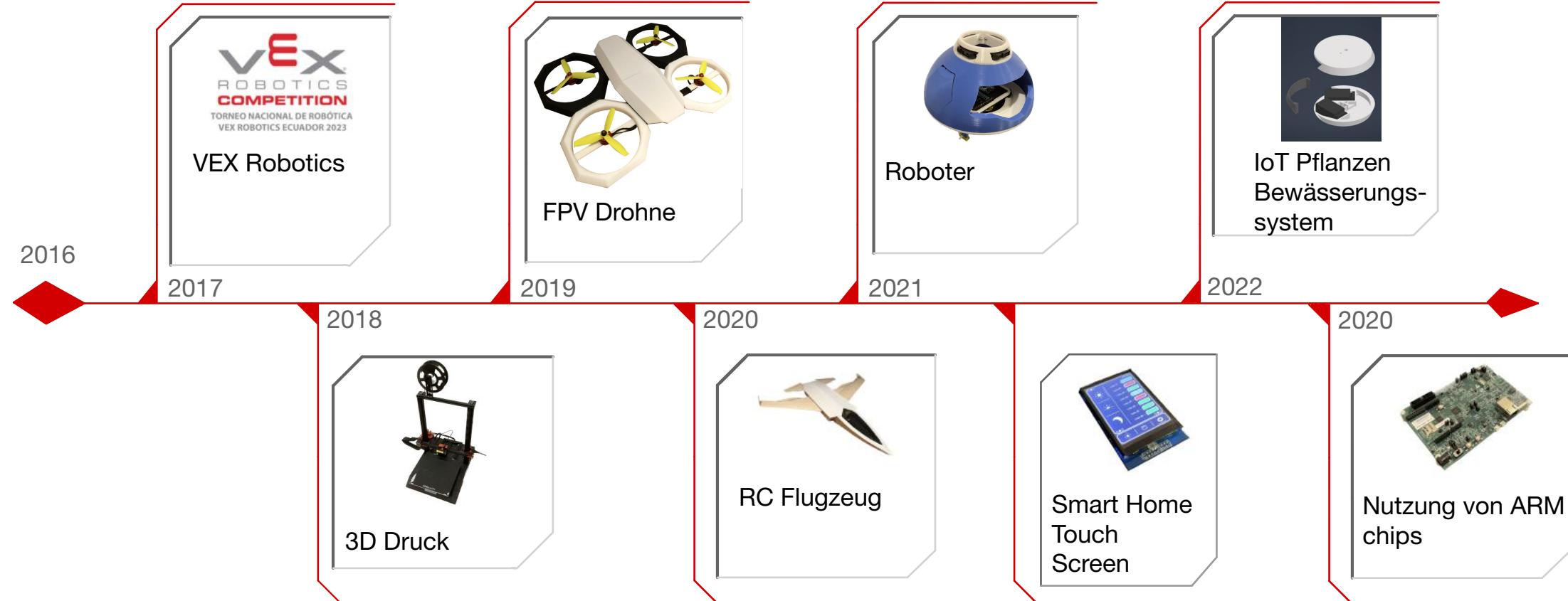
# Projekt Portfolio

---

Von Christopher Jung

# Meine Projekte:

Von 2016 bis 2022





# VEX Robotics

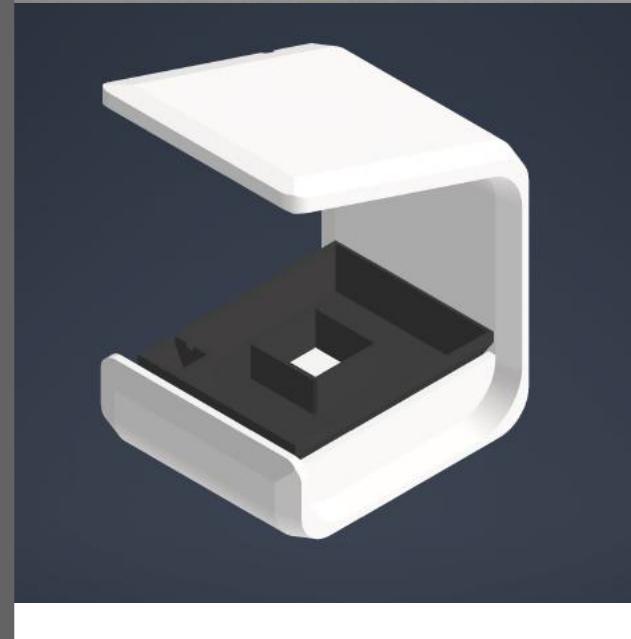
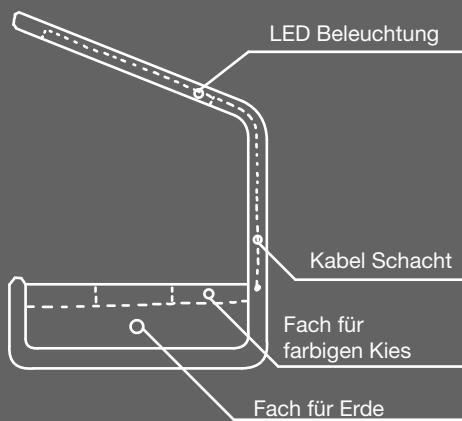
## Competition

Das VEX Robotics ist ein internationales STEM Programm, in dem Schulen oder privat organisierte Teams in Events gegeneinander antreten und mit selbstgebauten Robotern in Arenen Aufgaben erledigen müssen, um Punkte zu erzielen. Die Roboter müssen nach Regeln gebaut werden, wie z.B.: Es dürfen nur VEX Teile verwendet werden und der Roboter darf in seiner kompaktesten Form eine bestimmte Größe nicht überschreiten. Mehr infos hier: <https://www.vexrobotics.com>

# 3D Druck

## Blumentopf Mit Beleuchtung

In dem STEM Programm habe ich auch die Grundkenntnisse des CAD modellieren erlernt sowie das Ausdrucken der modellierten Teile mit dem 3D Drucker. Dieser Blumentopf mit Beleuchtung ist eines meiner Dekorationsprojekte, die heute meinen Wohnraum dekorieren.



### 3D Druck Details:

Die Teile für diesen Blumentopf wurden auf dem Creality Cr10s Pro in weißem und schwarzen PLA gedruckt. Der gesamte Blumentopf besteht aus 5 Einzelteilen, die in 4 Gängen gedruckt wurden. Danach wurde ein LED Streifen als Beleuchtung eingesetzt, das obere Fach mit blauem Aquarium Kies gefüllt und dekorierende Elemente aus Holz geschnitten und eingebaut.

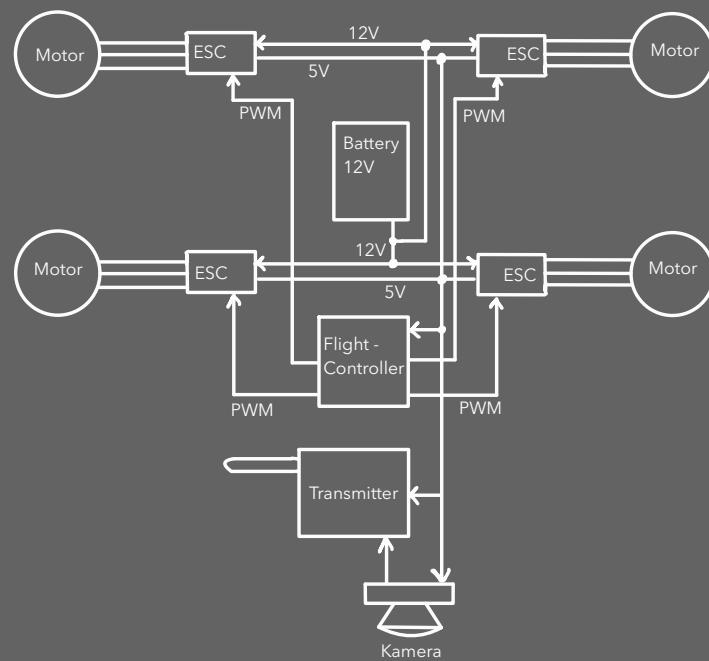


# FPV Drohne

Dieser FPV Quadcopter wurde aus selbst 3D modellierten und 3D gedruckten PLA Teilen zusammengebaut.

Für die Elektronik wurde ein Beta FPV Flightcontroller verwendet, zusammen mit 4 ESCs und einer FPV Kamera mit Video-Transmitter.

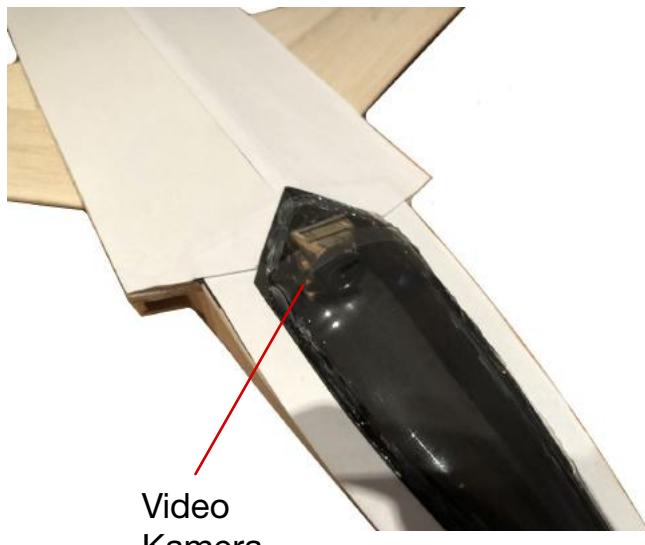
Block Diagramm:



# Selbstgebautes Flugzeug

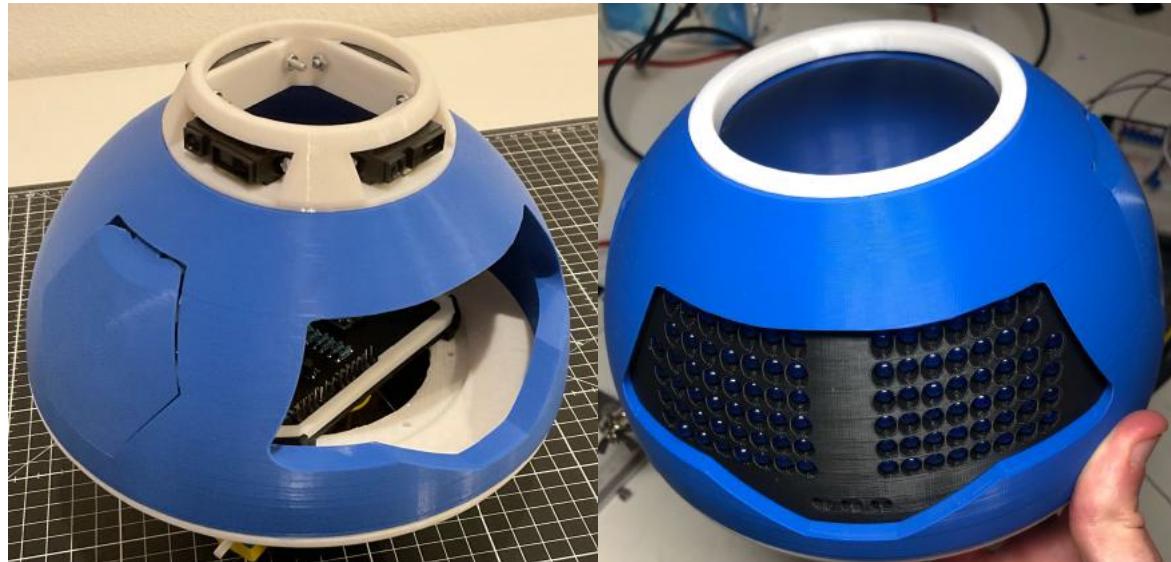
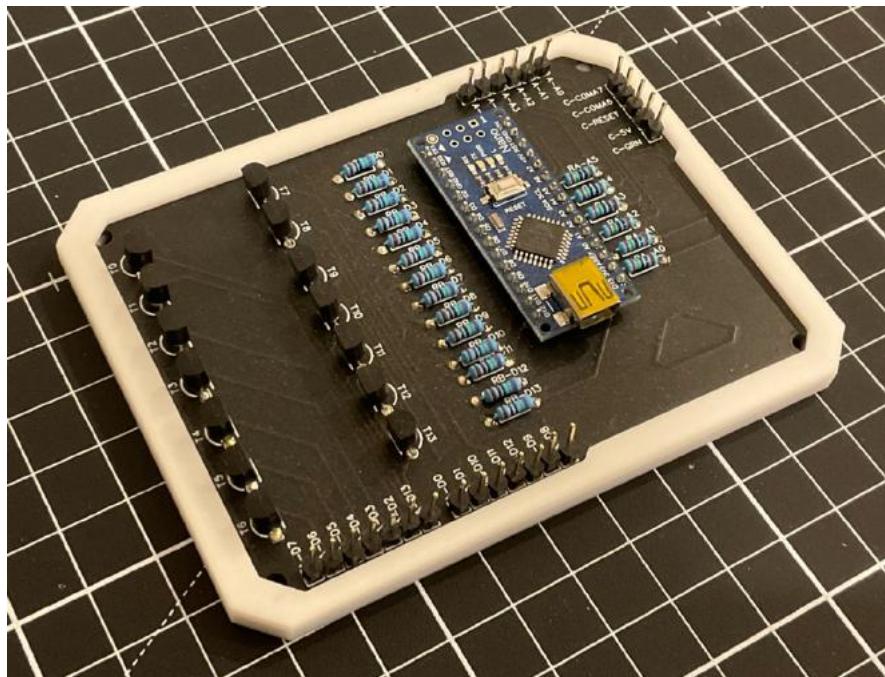
Hier sehen Sie eines meiner selbst gebauten RC-Flugzeuge. Das Flugzeug wurde selbst geplant und aus Balsaholz gebaut.

Das Flugzeug hat 4 Control-Surfaces und ist innen mit einer Videokamera mit Live-Verbindung zum Boden ausgestattet.



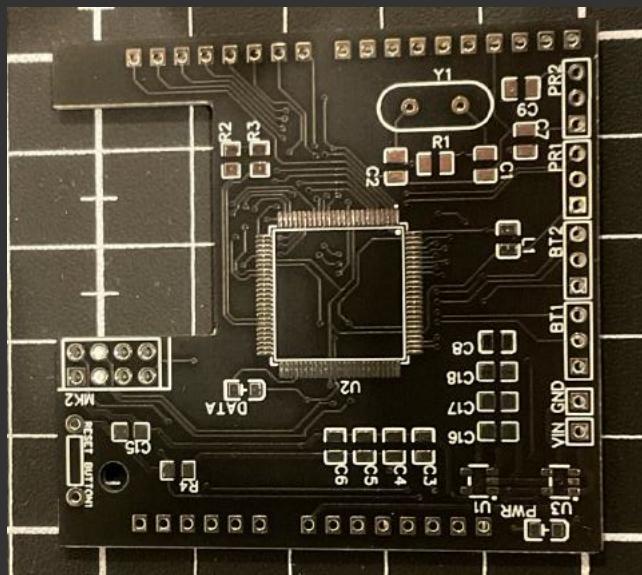
# 3D gedruckter Roboter

In diesem Projekt wurde ein Roboter aus 3D gedruckten Teilen gebaut. Der Roboter wurde mit einer selbstgebauten LED Matrix und Infrarotsensoren zur Navigation ausgestattet. Die LED Matrix wurde mit Hilfe einer selbstentworfenen Leiterplatte angesteuert, die die Signale des Arduinos verstärkt.

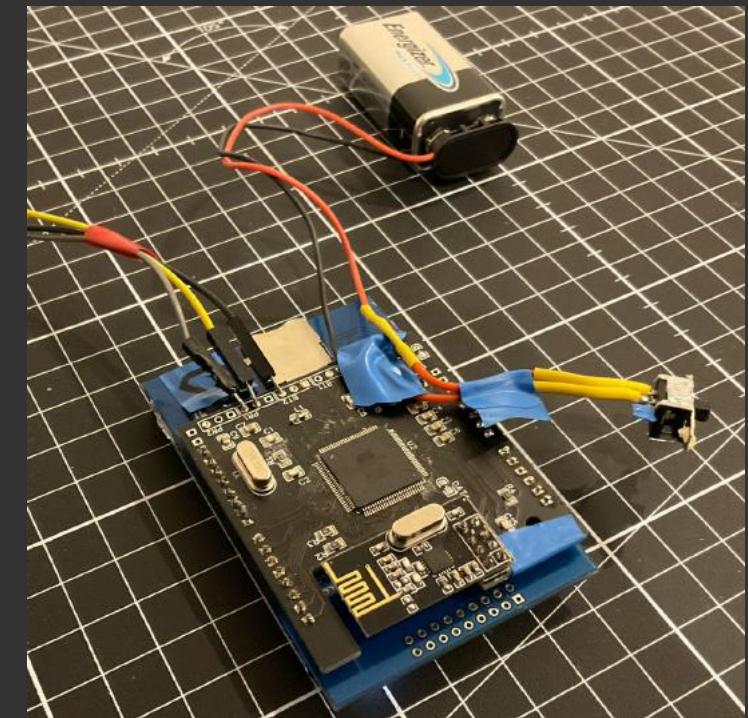


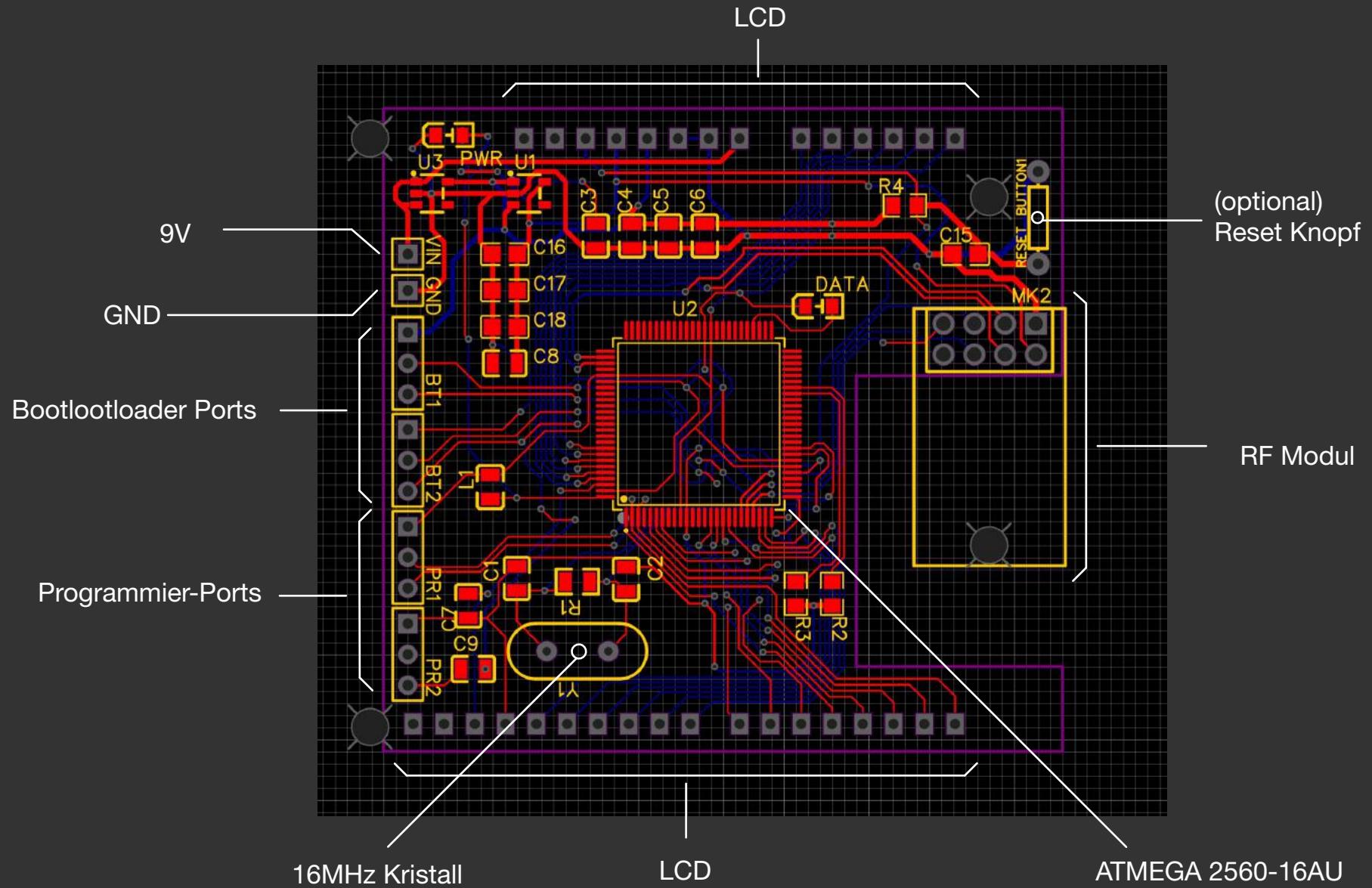
# Touch Screen Smart Home Transceiver

In diesem Projekt wurde ein Touch Screen Modul auf eine selbst entworfene Platine gelötet. Die Platine hat eine AVR MCU [ATmega2560] als Prozessor und ein NRF24L01 Modul zur Kommunikation. Auch das Receiver Modul wurde selbst von mir entworfen und basiert auf dem ATMEGA 328. Hierbei wurde diese extern gefertigt und bestückt. Die Platine des Touch Screen wurde auch extern gefertigt aber selbst bestückt. Sie wurde dann mit Hilfe eines FTDI-Adapters mit einem eigenen Script programmiert.



Receiver Modul



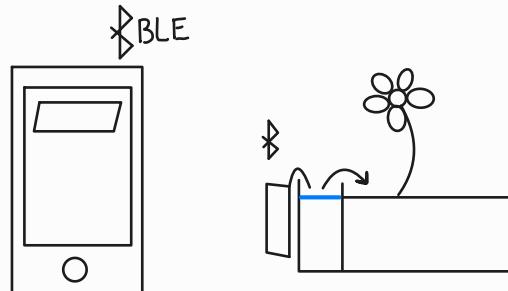


# IoT Pflanzen Bewässerungs-System

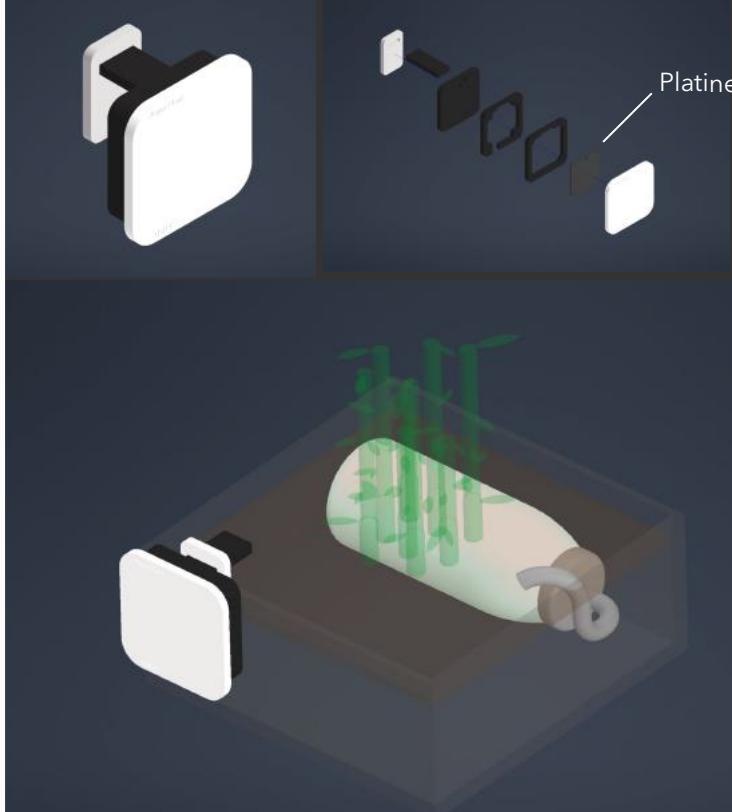
## Die Idee

Die Idee war, ein Gerät zu bauen, das eine Pflanze in Abwesenheit der Pflegeperson am Leben erhält und mit Wasser versorgt.

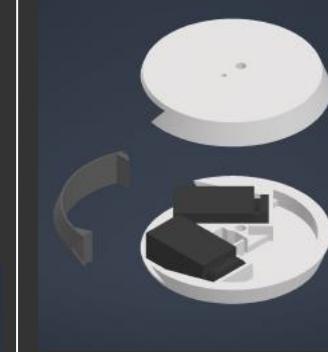
Es wurden zwei verschiedene Varianten für das System geplant.  
Variante 1: Die Steuereinheit wird außen angebracht und Wasser aus einer vergrabenen Flasche in die Erde gepumpt. Variante 2: Ein Ventil gibt Wasser aus einer auf dem Gerät stehenden Flasche in die Erde.



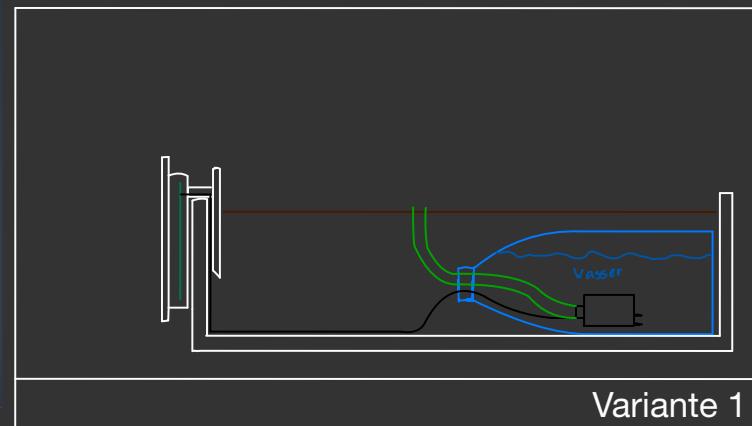
Variante 1



Variante 2



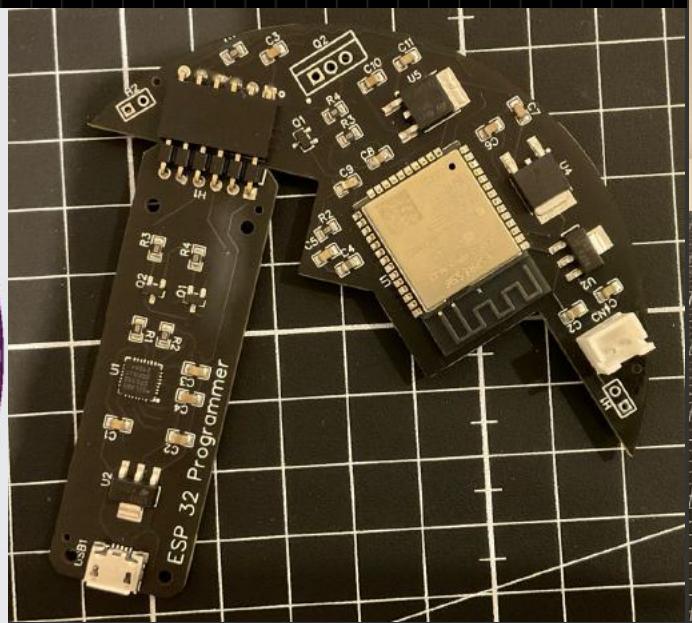
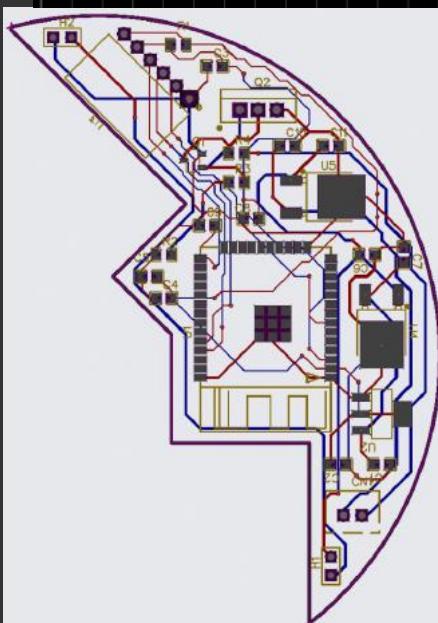
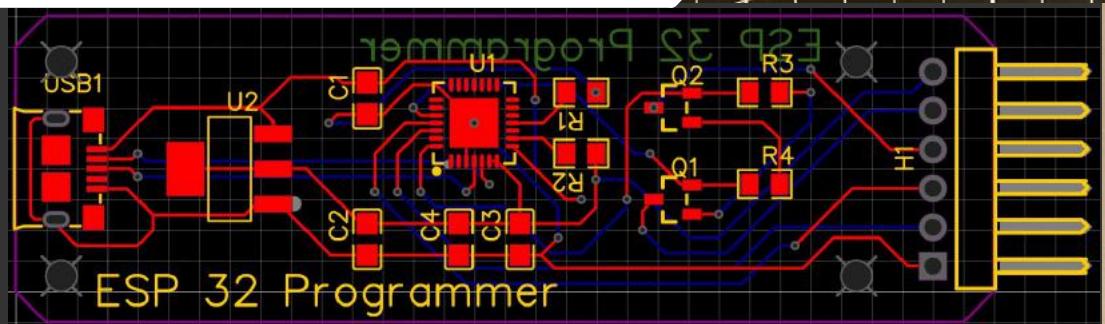
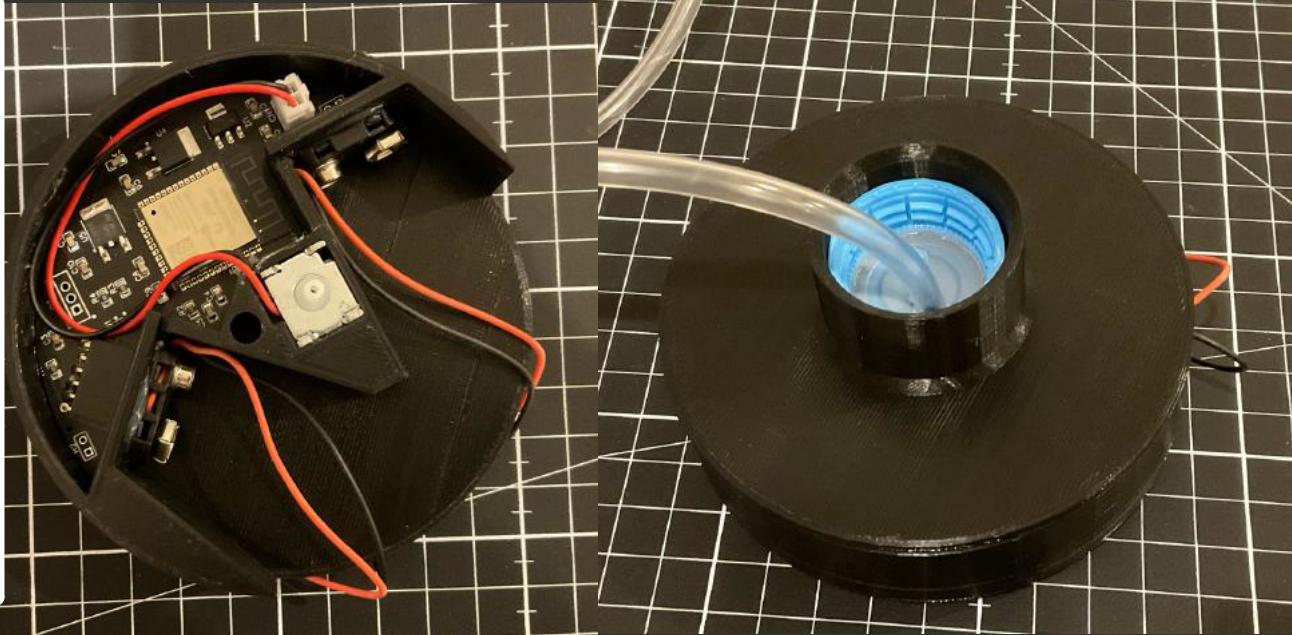
Variante 2



Variante 1

# Ausführung

Zur Ansteuerung wurde ein Controller-Board entworfen, das einen ESP32 benutzt, um eine Bluetooth Verbindung zu einem Handy zu erstellen. Zusätzlich zum Controller-Board wurde ein passendes Programmier-Board basierend auf einem CP 2102 gebaut, um den ESP über USB zu programmieren.



Damit der Nutzer das Gerät komfortabel benutzen kann, wurde in Swift eine App programmiert, bei der man Bewässerungszeiten festlegen kann und den Bewässerungsplan an das Gerät weitergibt.

The screenshot shows the Xcode interface with the following details:

- Project Navigator:** Shows a file named "ContentView.swift".
- Editor:** Displays the Swift code for the `ContentView`. The code initializes a `BLE` object and defines state variables for two watering sessions, each with time, start time, and volume.
- App Preview:** Shows a simulated interface titled "Watering Plan". It includes a "My Device" section with a "Connect Device" button, a weekly calendar grid for setting times and volumes, and a "Set Plan" button.

```
1 import SwiftUI
2
3 struct ContentView: View {
4     // Setup BLE
5     @StateObject var MyBLE = BLE()
6
7
8     @State var isConnected: Bool = false
9
10    @State var Time1: Int = 1
11    @State var T1sta: String = "Am"
12    @State var T1ml: Float = 300
13    @State var T1mlFinal1: Int = 0
14    @State var T1mlFinal2: Int = 0
15
16    @State var Time2: Int = 1
17    @State var T2sta: String = "Am"
18    @State var T2ml: Float = 50
19    @State var T2mlFinal1: Int = 0
20    @State var T2mlFinal2: Int = 0
21
22
23
```

# Nutzung von Arm Chips

Um stetig bessere und intelligenter Projekte umzusetzen ist eine fähige Elektronikplattform Notwendig.

Um eine solche Plattform zu verwirklichen werden entsprechende Chips benötigt. Hier eignen sich ARM Chips besonders gut da die Cortex Architekturen Standardisiert sind. Die Spitzenhersteller dieser Chips sind Firmen wie NXP und STMicroelectronics. Dabei wurden die Nucleo Boards von ST erfolgreich programmiert und sind nun für den Einsatz in der Plattform bereit. Aktuell wird NXPs i.MX8M Cortex M7 für den Einsatz in der Plattform vorbereitet. Das Ziel dieses Projektes ist es, eine Microkontrollerplattform zu erstellen, die es mir ermöglicht, einfach und flexibel leistungsstarke Cortex Module für zukünftige Robotik und andere Projekte zu fertigen um, diese besser und einfacher umsetzen zu können. Für die Zukunft plane ich nicht nur ARM Cortex M4 und M7 Microcontroller in die Plattform einzubringen sondern auch Mikroprozessoren wie NXPs i.MX8M und Layerscape Prozessoren, die auf den ARM Cortex A53 und A72 Architekturen basieren.

