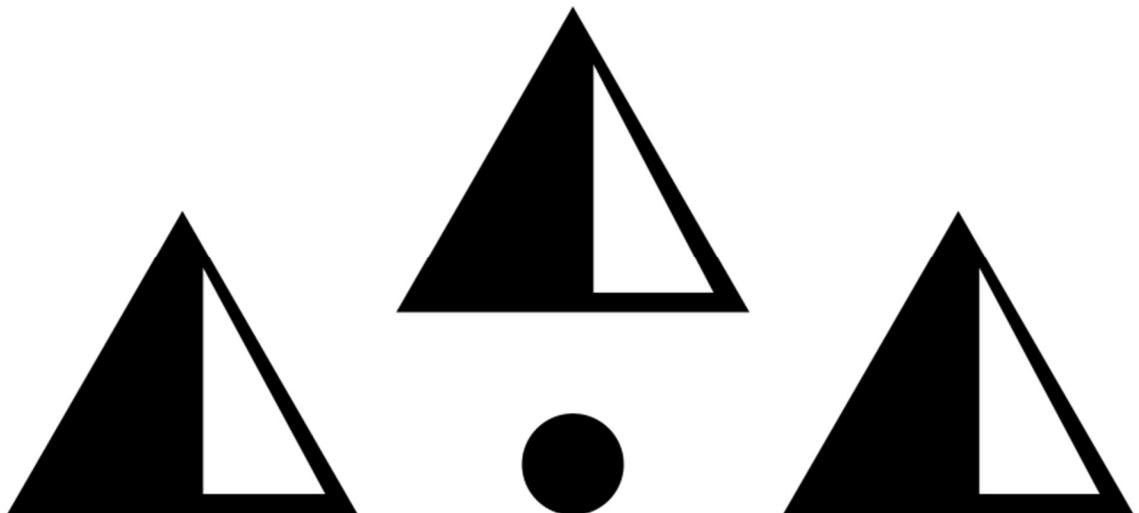


2020

Glück

Ein Glücksspiel aus reinem Glück

Berufsmaturitätsarbeit



Lionardo Wernli und Marco Stauber

Berufsmaturitätsschule Zürich

Technik, Architektur, Life Sciences

Begleitet durch: R. Suter

BEL17A

Abgabe: 24. November 2020

1. Inhaltsverzeichnis

| | |
|---|-----------|
| 2. ABSTRACT | 2 |
| 3. DAS HÜTCHENSPIEL | 3 |
| 4. ZUSÄTZLICHE INFOS..... | 4 |
| 5. ENTWICKLUNG..... | 5 |
| 5.1. INFORMIEREN | 6 |
| 5.2. ENTSCHEIDEN | 6 |
| 5.3. PLANEN..... | 7 |
| 5.4. REALISIEREN | 8 |
| 5.4.1. Stromversorgung | 9 |
| 5.4.2. Random Number Generator (<i>Noise Generator</i>) | 10 |
| 5.4.3. OLED-Display + Jogpad | 11 |
| 5.4.4. USB-C Kommunikation..... | 12 |
| 5.4.5. Elektromagnet, Schritt- und Servomotor | 13 |
| 5.4.6. LED-Kontroller | 13 |
| 5.4.7. Gehäuse | 14 |
| 5.5. KONTROLLIEREN..... | 17 |
| 6. SCHLUSSWORT | 18 |
| 7. GLOSSAR | 19 |
| 9. QUELLENVERZEICHNIS | 21 |
| 9.1. ABBILDUNGSVERZEICHNIS | 21 |
| 9.2. ZITATVERZEICHNIS..... | 22 |
| 10. DANK | 22 |
| 11. ANHANG | 23 |
| 11.1. SCHEMA..... | 23 |
| 11.2. LAYOUT..... | 32 |
| 12. BESCHEINIGUNG | 34 |

2. Abstract

Unser Ziel ist es, ein angebliches Glücksspiel zu einem echten Glücksspiel zu machen. Wir wollen wissen, ob dieses Spiel dann mehr Spass macht, da man nicht abgezogen werden kann, oder ob es dadurch den Anreiz verliert. Darum entwickeln wir eine elektronische Variante des bekannten Hütchenspiels, das nicht von uns beeinflusst werden kann. Da es in der Elektrotechnik keinen Zufall gibt, verwenden wir einen Mikrocontroller mit eingebauten Random Number Generator von STM, der durch einen Noise Generator eine dezimale Zahl berechnet, die den Spielablauf bestimmt. Mit LEDs oder einem Servomotor stellen wir die Hütchen dar. Die LEDs werden über einen LED-Controller angesteuert, mit dem wir lange Probleme hatten, da das Datenblatt dazu sehr unklar war und man viel hineininterpretieren konnte. Am Ende haben wir es geschafft beide Methoden zum Laufen zu bekommen und haben jetzt ein fertiges elektrisches Hütchenspiel mit simuliertem Glück

3. Das Hütchenspiel

Das Hütchenspiel ist ein vermeintliches Glücks- und Geschicklichkeitsspiel, bei dem ein Gegenstand unter einer Reihe gleich aussehender Becher oder Schalen versteckt wird. Der Leiter des Spiels vertauscht möglichst schnell und oft diese Schalen, um zu erreichen, dass der Spieler nicht mehr weiss, unter welcher Schale sich der Gegenstand befindet. Dieses Spiel wird oft als Trickbetrug verwendet. Durch Tricks und Ablenkung wird versucht, den Gegenstand komplett verschwinden zu lassen, sodass keine Möglichkeit besteht, ihn zu finden. Diese Trickbetrüger lassen die Spieler trotzdem immer wieder gewinnen, um eine Gewinnchance vorzutäuschen und den Spieler motiviert zu halten. In der Schweiz ist solch ein Betrug verboten.

«Wer in der Absicht, sich oder einen anderen unrechtmässig zu bereichern, jemandem durch Vorspielen oder Unterdrückung von Tatsachen arglistig irreführt oder ihn in einem Irrtum arglistig bestärkt und so den Irrenden zu einem Verhalten bestimmt, wodurch dieser sich selbst oder einen anderen am Vermögen schädigt, wird mit Freiheitsstrafe bis zu fünf Jahren oder Geldstrafe bestraft» (Schweizer-Bund 1937, StGB Art. 146 Abs. 1)

Es wird hauptsächlich in organisierten Gruppen gespielt. Da wir aber versuchen, echtes Glück zu simulieren, distanzieren wir uns von diesen Machenschaften.

(Wikipedia Hütchenspiel 2020)

4. Zusätzliche Infos

Alle Dateien wie Dokumentation, Printdateien, Code und sonstige Daten findet man im folgendem Link oder QR-Code. Dieser Link führt zu einem GitHub, auf welchem alle Dateien hochgeladen wurden. Dieser QR-Code ist auch zu finden, wenn man im Menü unseres Produkts auf „QR“ drückt.

https://github.com/MXACE/BMA_2020

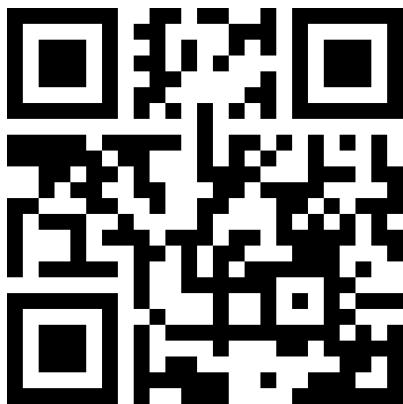


Abbildung 1: QR-Code Github

Das Produkt ist bei der Abgabe komplett funktionstüchtig und ist unbeschädigt.

/

5. Entwicklung

Wenn man an Glück denkt, denkt man schnell an Glücksspiel, dabei sind die meisten Glücksspiele nicht unbedingt Glückssache. Sie simulieren dem Spieler lediglich eine Glücks- oder Pechsträhne vor, um seine Motivation aufrechtzuerhalten. Doch wie sieht ein Spiel aus, das echtes Glück beinhaltet? Genau das haben wir uns gefragt. Wir wollen wissen, wie viel Spass ein Glücksspiel macht, wenn es auf echtem Glück basiert. Unsere Vermutung ist, dass ein Glücksspiel keinen Anreiz zum Weiterspielen mehr liefert, wenn das Gewinnen reines Glück ist. Um dies zu beweisen, wollen wir ein elektronisches Glücksspiel entwickeln, das möglichst nur auf reinem Glück basiert. Wir kennen den Betrug aus Videospielen, welche sogenannte «Lootboxen» beinhalten. Diese versprechen eine hohe Chance darauf, seltene Gegenstände zu erhalten, was man aber trotzdem nie tut. Somit kann die Firma, die dahinter steckt, von dem Geld profitieren. Obwohl man von dem Betrug weiß, sind Lootboxen nach wie vor verlockend, da jeder, der Videospiele spielt, gerne einzigartige Gegenstände besitzt. (Trotzdem sind diese verlockend, weil die Gegenstände im Spiel schön aussehen und jeder, der Videospiele spielt, gerne etwas Einzigartiges hat.)

5.1. Informieren

Zum Überthema «Glück» sind uns sehr schnell Glücksspiele in den Sinn gekommen und damit verbunden der Gedanke, dass die meisten bekannteren Glücksspiele eher Betrug und Abzocke sind. Deshalb haben wir uns überlegt, wie wir ein Glücksspiel entwickeln können, das möglichst auf reinem Glück basiert und immer noch Spass macht.

5.2. Entscheiden

Als klassisches Glückspiel, das gerne zum Betrug verwendet wird, haben wir uns für das Hütchenspiel entschieden, da es allbekannt ist und wir eine klare Vorstellung hatten, wie wir das realisieren könnten. In der Elektronik gibt es keinen echten Zufall. STelectronics wirbt mit einem Mikrocontroller, der einen eingebauten «Random Number Generator» hat. Dieser erzeugt mit einem Noise Generator eine 20-stellige Dezimalzahl, mit der wir einen Spieleablauf realisieren können. Da wir uns nicht entscheiden konnten, ob wir dieses Hütchenspiel mit Bechern oder einfach mit LED darstellen wollten, haben wir eine Hybrid-Variante erstellt.

5.3. Planen

Um ein Glücksspiel aus reinem Glück zu entwerfen, haben wir uns überlegt, dass der Mensch das Spiel nicht steuern darf. Daher wäre es naheliegend, einen Mikrocontroller entscheiden zu lassen, was passiert, da dieser nur logische und mathematische Zusammenhänge versteht. Einer unserer ersten Schritte war es, zu planen, was das Produkt kann und was man als Input geben kann. Entschieden haben wir uns für zwei Methoden. Bei der einen kann man das Ganze nur optisch beobachten, bei der anderen nur akustisch mitverfolgen. Letztere beinhaltet einen Motor.

Wir haben skizziert, wie das Produkt nach unseren Vorstellungen aussehen sollte. Um einen klaren Ablauf zu haben, haben wir ein Blockschema mit dem groben Aufbau des ganzen Projekts erstellt. Die ersten Bauteile wurden gesucht und das dazugehörige Datenblatt gelesen. Wenn es gepasst hat, haben wir das Bauteil in eine Stückliste aufgenommen, und falls nicht, wurde eine Alternative gesucht.

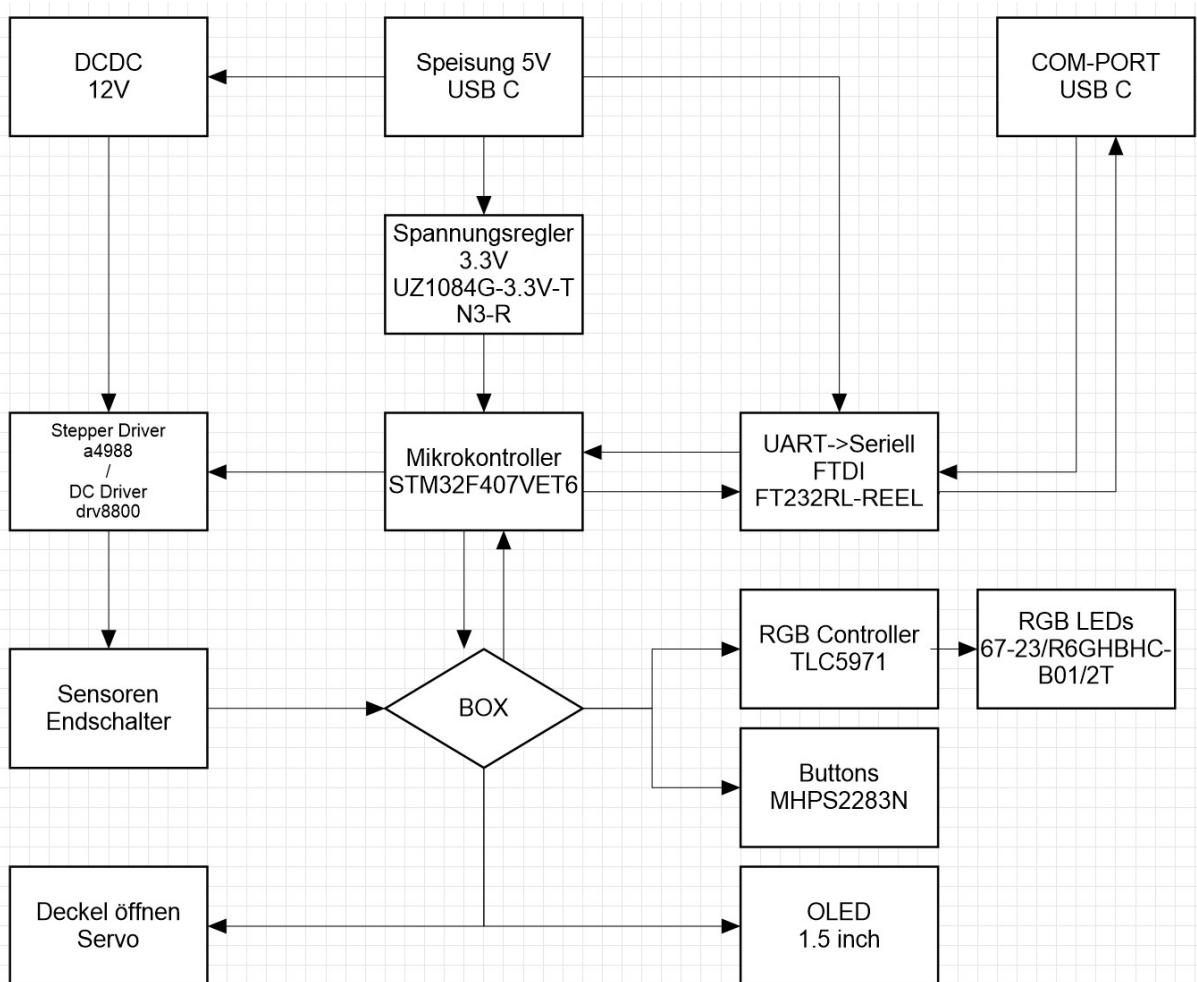


Abbildung 2: Blockschaubild

5.4. Realisieren

Aus dem Blockschema (Abbildung 2) wurde zuerst der Print gezeichnet. Für das Schema und das Layout wurde Altium und für das Zeichnen des Gehäuses wurde Solidworks genutzt. Das Schema und das Layout konnten wir ohne Testaufbau zeichnen, da wir bis auf den LED Controller nur Bauteile nutzten, die wir schon kannten und bei denen wir wissen, wie sie funktionieren. Dadurch, dass wir die bestückte Leiterplatte schnell hatten, konnten wir früh mit dem Programmieren anfangen und uns Zeit lassen. Um möglichst wenig Fehler beim Programmieren zu verursachen, haben wir den Code für die Module einzeln geschrieben. Damit können wir sicherstellen, dass allfällige Fehler von einem einzelnen Modul kommen. In unserem Fall haben wir den Mikrocontroller in der Programmiersprache C programmiert, welche wir in der Ausbildung erlernt haben. In dieser ist es möglich, mehrere Dateien miteinander zu einem Programm zu verknüpfen. So können wir beispielsweise eine Datei für den Bildschirm und eine andere für den USB-C Konverter erstellen. Das gibt uns eine klare Struktur, falls wir etwas am Code ändern oder hinzufügen müssen. Für die interne Peripherie des Mikrocontrollers haben wir HAL-Bibliotheken verwendet. Sie geben uns eine Grundlage des Mikrocontrollers, sodass wir nicht das über 1700-seitige Manual lesen mussten.

```
7 void Servo_init()
8 {
9     htim4.Instance->CCR1 = 30; //30-120
10    HAL_TIM_PWM_Start(&htim4,TIM_CHANNEL_1);
11 }
12
13 void Servo_angle(uint16_t angle)
14 {
15     uint16_t DutyCycle = 0;
16
17     if(angle > 180) angle = 180;
18
19     if(angle == 0) DutyCycle = 30;
20     else DutyCycle = 30 + angle/2;
21
22     htim4.Instance->CCR1 = DutyCycle; //30-120
23 }
24
```

Abbildung 3 Funktionen für den Servomotor in der Programmiersprache C

5.4.1. Stromversorgung

Um den Print mit Strom zu versorgen, mussten wir zunächst wissen, wie viel Strom der ganze Aufbau braucht. Wir haben uns für eine USB-C Schnittstelle entschieden, da diese bis zu 5 Ampere ausgelegt ist, was mehr als ausreichend ist. Eine USB-Schnittstelle eines Computers kann niemals so viel liefern, doch wir wollten eine Kommunikationsmöglichkeit haben. Die einfachste Methode war es, zwei USB-C Anschlüsse zu haben. Damit sich der Strom nicht aufteilt, brauchten wir dafür ein Relais, welches ein Master-Slave System umsetzt. Strom kann von beiden Anschlüssen gezogen werden, jedoch wird der Versorgungsanschluss priorisiert, während die Kommunikation immer noch vorhanden ist. Im Fall, dass nur der Computeranschluss verbunden ist, kriegt der Motor keine Speisung.

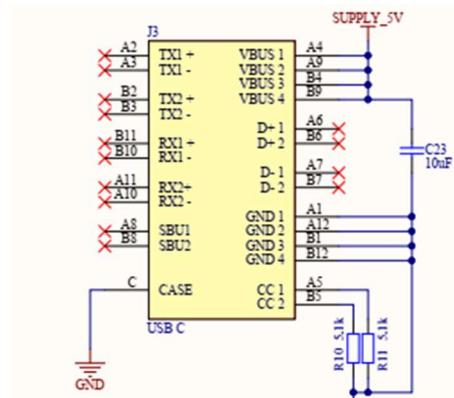


Abbildung 4: Versorgungs-USB-C

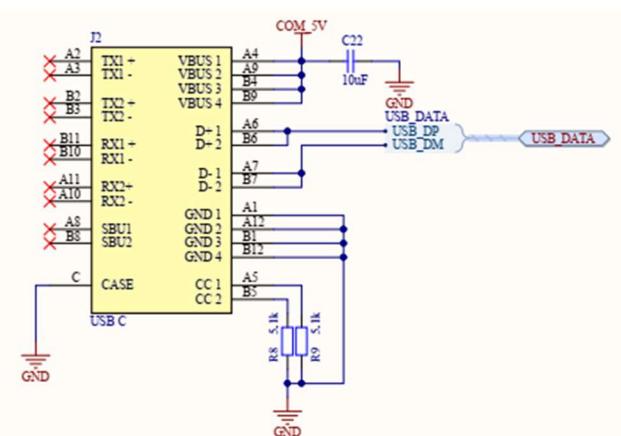


Abbildung 5: Kommunikations-USB-C

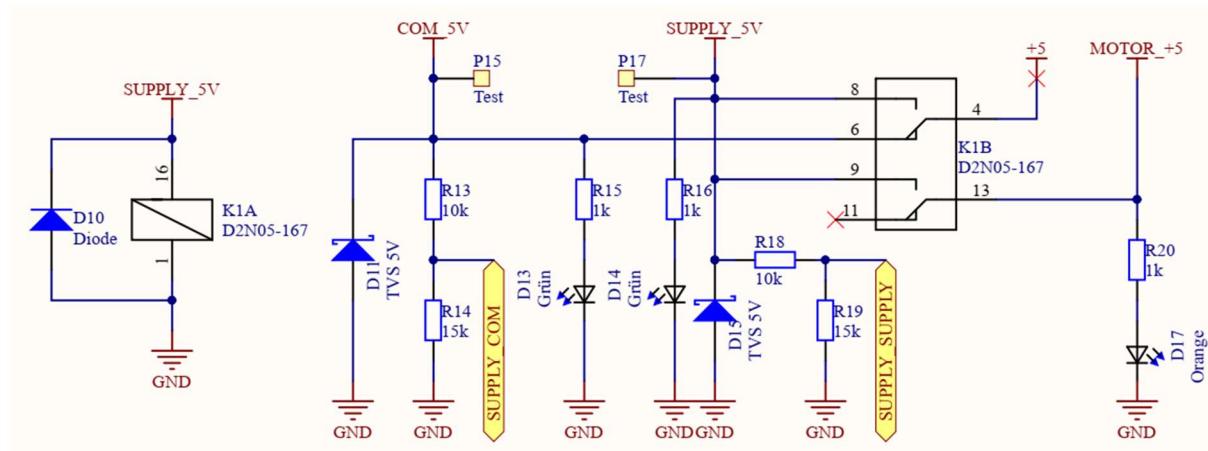


Abbildung 6: Schema Relaissteuerung

5.4.2. Random Number Generator (Noise Generator)

Ein Random Number Generator (kurz RNG) ist eine elektrische Schaltung, die ein Rauschen erzeugt und auch als Noise Generator bezeichnet wird. Rauschen an sich ist in der Elektronik ein ungewollter Nebeneffekt, der Signale verunreinigen kann und sie unlesbar macht, doch diese Schaltung ist aufgebaut, um genau dies zu erzeugen. Das Rauschen wird in diesem Modul gebraucht, um einen zufälligen Spannungswert zu erhalten. Der Spannungswert wird kurzzeitig gespeichert und mit einem Analog/Digital-Wandler in einen digitalen beziehungsweise binären Code umgewandelt. So erhält man dann eine Zufallszahl.

Unser Random Number Generator liest dieses Rauschen mit einem Analog/Digital-Wandler. Von den vorgegebenen Funktionen kann unser Mikrocontroller eine 20-stellige Dezimalzahl generieren. Da es fast nicht sein kann, dass bei einem Rauschen zweimal die gleiche Zahl herauskommt, ist eine so generierte Zahl sehr zufällig. Unserer Mikrocontroller unterstützt nur einen 10 Bit grossen Wert. Für unseren Nutzen genügt die Grösse der Zufallszahl auf jeden Fall.

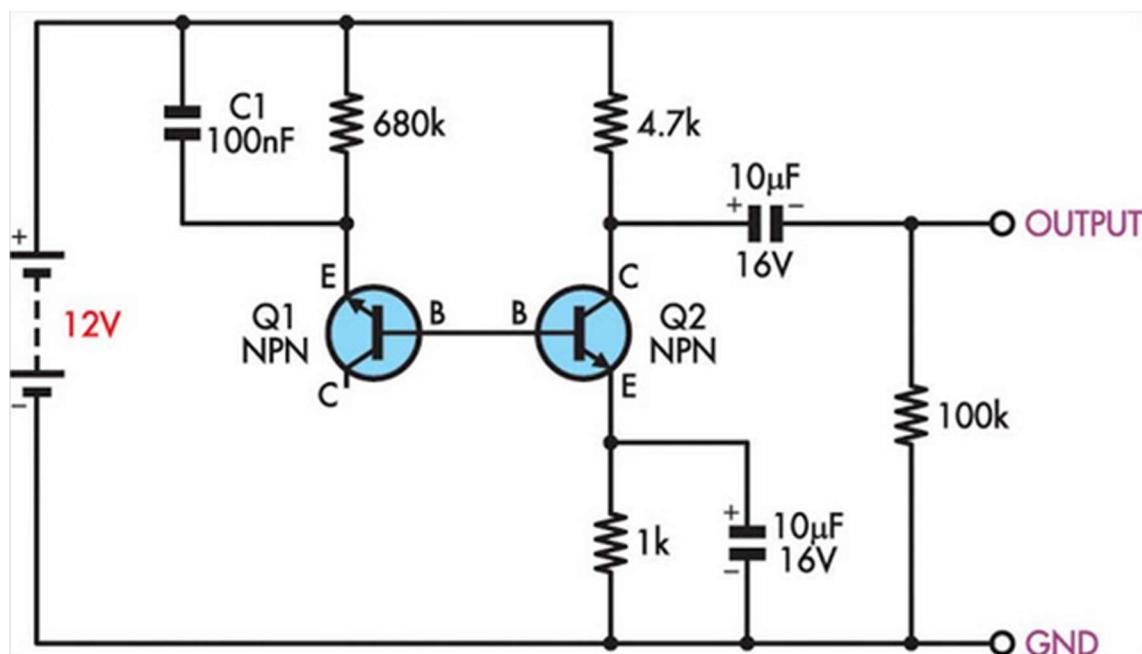


Abbildung 7: White Noise Generator

5.4.3. OLED-Display + Jogpad

Mit unserem OLED-Display haben wir eine direkte visuelle Informationsansicht des Mikrocontrollers. Auf diesem stellen wir in erster Linie Information dar, wir können aber auch den QR-Code anzeigen oder die einzelnen Methoden starten. Je nach ange schlossener Speisung ist die Motor-Taste ausgeblendet oder nicht. Das Menu wird durch fünf Tasten gesteuert, die in einem Kreis angeordnet sind, wobei die vier äusseren der Navigierung und der mittlere der Bestätigung dienen. Der Code für das grafische Zeichnen und die Buchstaben auf dem OLED Display basiert auf der Bibliothek von L. Heizer. Das effektive Design und die Funktionen der grafischen Tasten haben wir selbst programmiert.

Das Display soll bei der Steuerung des Produktes helfen.

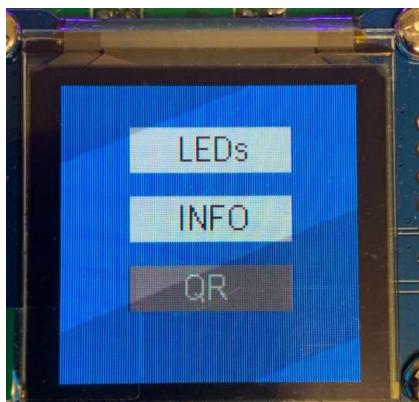


Abbildung 8 Hauptmenu mit Speisung über Kommunikations USB-C



Abbildung 9 Hauptmenu mit Speisung über den Versorgungs USB-C

Die dunklen Balken kommen von der Aktualisierungsrate des Bildschirmes.

5.4.4. USB-C Kommunikation

Für uns war es wichtig, dass wir in Echtzeit Informationen abfragen können, um mögliche Fehler zu finden. Dafür verwenden wir die UART-Ausgänge des Mikrocontrollers. UART ist ein etwas älteres serielles Kommunikationsprotokoll, das mit zwei Leitungen funktioniert. Um das UART-Signal des Mikrocontrollers für einen Computer verständlich zu machen, benötigt man einen UART-zu-Seriell Konverter-Chip. Ein sehr häufig verwendetes Modul, welches wir auch benutzen, ist der FTDI-Chip. Dieser ist präzise steuerbar, indem man zusätzlich den Datenfluss kontrollieren kann. Zwei LEDs zeigen für jeweils eine Richtung an, ob Daten fliessen. Wenn man nun das USB-Kabel an einen Computer und an der USB-C-Kommunikation anschliesst und ein Terminal aufmacht, kann man mit den richtigen Einstellungen Daten, die der Mikrocontroller schickt, auslesen oder Daten an diesen senden. In unserem Fall wollen wir nur Daten lesen – wir brauchen diese Schnittstelle, um am Computer erkennen zu können, wann der Mikrokontroller in einen neuen Codeabschnitt kommt, welche zufällige Zahl generiert wurde oder welche Taste gedrückt wurde. Für das Terminal benötigt man die Standardeinstellungen 115200 Baud und 8n1.

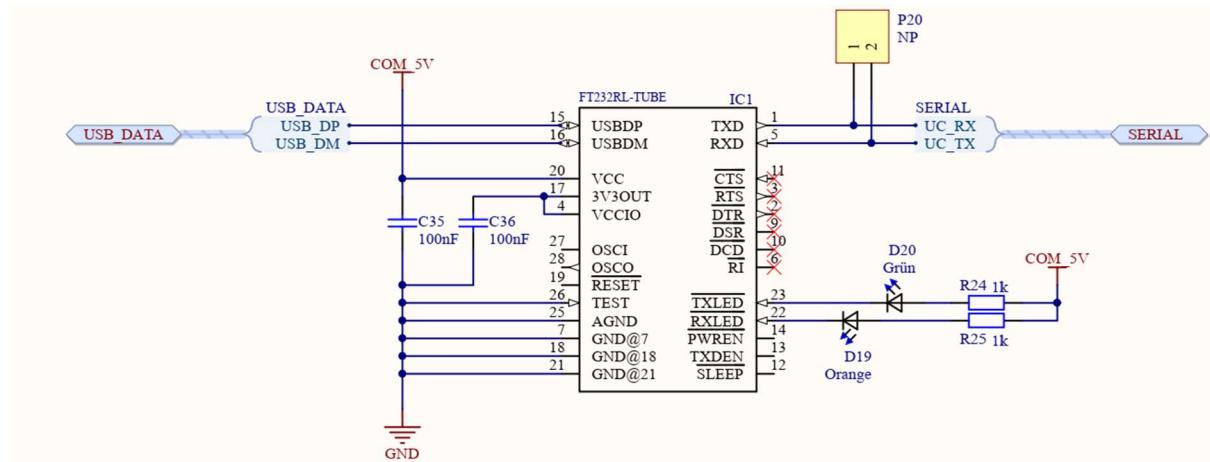


Abbildung 10: Schema UART-zu-Seriell Konverter-Chip, FTDI

5.4.5. Elektromagnet, Schritt- und Servomotor

Für die Motorenmethode wird eine Kugel zu einer zufälligen Box und zu einem zufälligen Zeitpunkt während des Ablaufes bewegt. Der Schrittmotor zieht eine Plattform, die auf einer linearen Schiene montiert ist. Diese Plattform kann sich ein wenig mehr bewegen als die Distanz zwischen den zwei äusseren Boxen. Der Motor wird durch ein austauschbares a4988-Modul gesteuert. Dieses erlaubt uns, die Schrittgrösse, die Schrittrichtung und die einzelnen Schritte zu steuern. Auf der Plattform ist ein Elektromagnet montiert, der die Kugel aufnehmen und fallen lassen kann, um die Kugel so zu transportieren. Damit man den ganzen Vorgang nicht sieht, verdeckt eine gedruckte Kunststoffplatte die Sicht auf die Kugel. Der Servomotor kann diese Platte bewegen, um die Sicht freizugeben.

5.4.6. LED-Kontroller

Der LED-Controller, den wir verwendet haben, kann vier RGB-LEDs ansteuern. Da wir sechs Boxen haben, bleiben zwei übrig. Trotzdem haben wir die übrigen zwei miteingeplant, da diese als Feedback dienen können, falls wir es brauchen. Wir vermuteten ursprünglich, dass die Ansteuerung sehr einfach sei, da man mehrere Controller in Serie anschliessen kann und diese über die zwei Leitungen (Clock und Daten) seriell angesteuert werden können. Schlussendlich sassen wir jedoch sehr lange dran, bis es funktionierte.

Das Problem war, dass unser Code nicht schnell genug war. Somit kam der nötige Code nie rechtzeitig an. Das Timing ist daher sehr wichtig: Bei zu grossen oder zu kurzen Pausen kann der Controller mit den Daten nichts anfangen. In diesen 224 Bit, die man an einem einzigen RGB-Treiber IC schickt, stellt man verschiedene Einstellungen, die einzelnen Farben sowie die Helligkeit der LEDs ein.

| MSB | LSB | | | | | | | | | | | |
|--------------------------------|------------------------------|-------------------------|--------------------------|------------------------|---------------------------|---------------------------|---------------------------|-------------|---------------------------|---------------------------|---------------------------|--|
| Write Command (6 bits, 25h) | Function Control (5 bits) | BC for BLUE (7 bits) | BC for GREEN (7 bits) | BC for RED (7 Bits) | GS for OUTB3 (16 Bits) | GS for OUTG3 (16 Bits) | GS for OUTR3 (16 Bits) | 16 Bits × 6 | GS for OUTB0 (16 Bits) | GS for OUTG0 (16 Bits) | GS for OUTR0 (16 Bits) | |

Abbildung 11:LED-Controller Daten

5.4.7. Gehäuse

Für das Erstellen des Gehäuses waren die Grössen des Prints und die des Testaufbaus für den Schrittmotor nötig. Die Printgrösse wussten wir schon seit dem Layout, also fehlte nur der Aufbau des Motors. Für diesen haben wir eine lineare Aluminiumschiene und ein Openbuilds Vslot-Kit auf eine Holzplatte geschraubt, die Funktionen getestet und nachgemessen, wie viele Schritte mit dem Schrittmotor nötig waren, um von einer Box zu der anderen zu kommen.

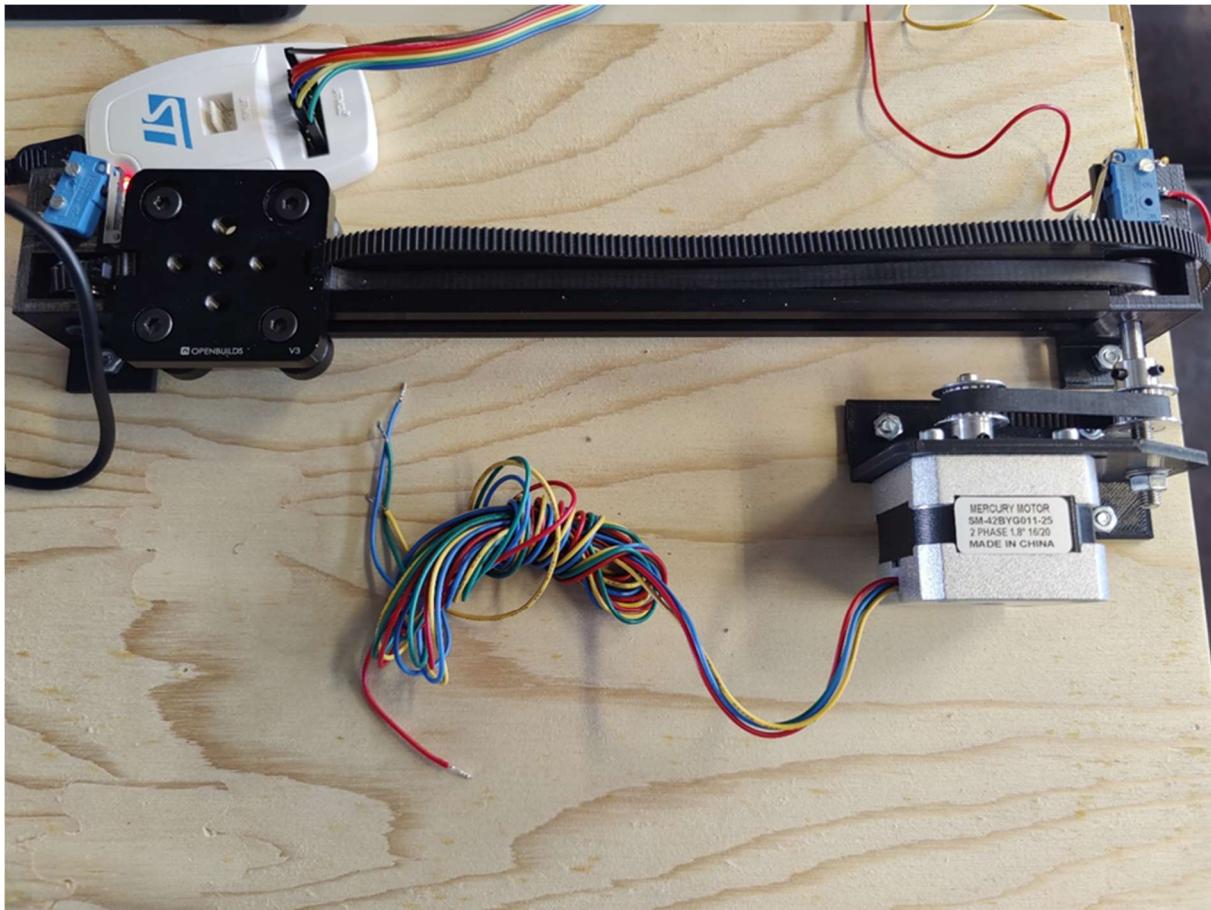


Abbildung 12: Testaufbau

Dieser Aufbau war funktionstüchtig und nun konnten wir das Gehäuse mit der Software Solidworks zeichnen. Die Löcher konnten wir direkt aus den Massen vom Aufbau und dem Print nehmen. Zusätzlich benötigt der kleine Ventilator Lüftungsschlitz, die wir in den Wänden eingeplant haben. An der Vorderseite sind zwei Aussparungen für die USB-C Anschlüsse und auf dem Deckel gibt es ein kleines Fenster, das Löcher für die Tasten hat und das durchsichtig ist, um die LEDs und die Kugel sehen zu können. Gleichzeitig ist unser Name wie auch die Klasse und der Text „BMA 2020“ eingraviert. Halter für die Kugel, den Elektromagnet, den Print, den Spannungswandler und die Gehäuseecken wurden aus PETG 3D-gedruckt.

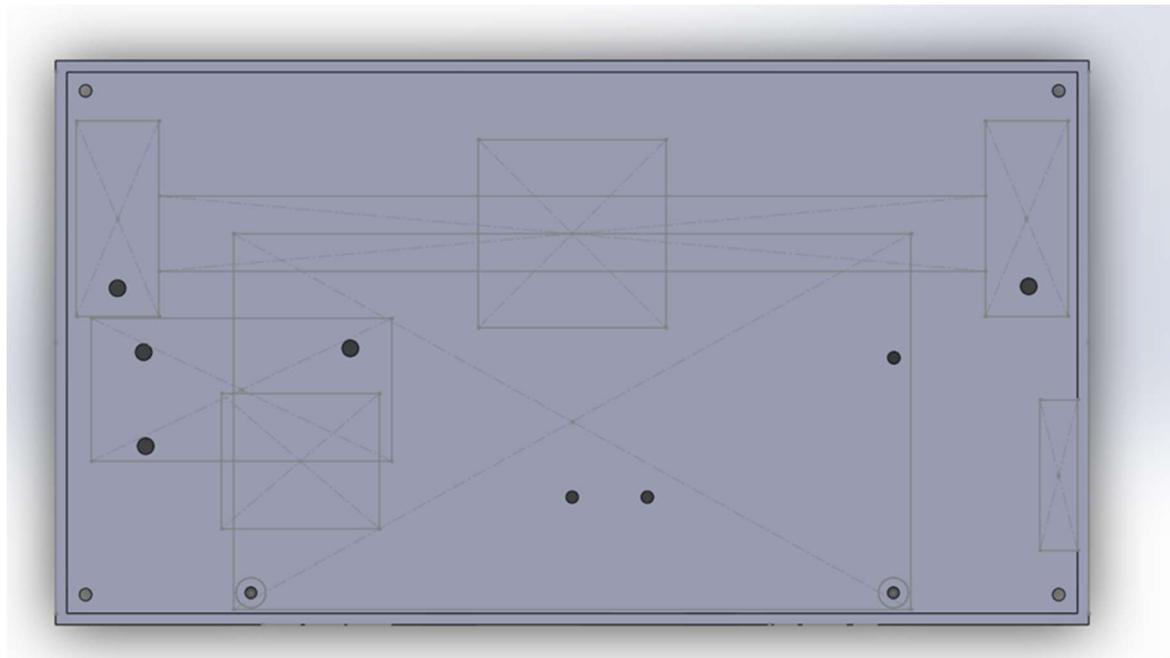


Abbildung 13: Solidworks Gehäuse Unten

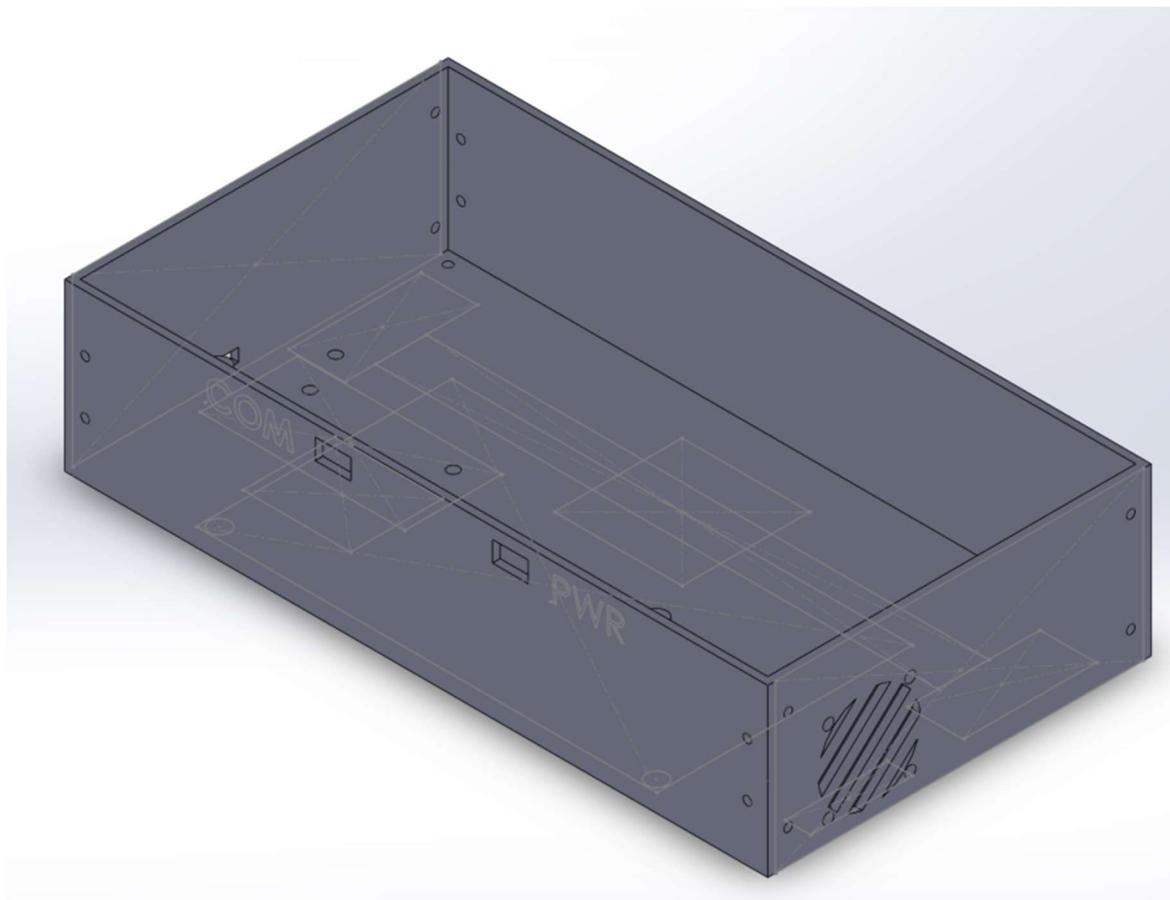


Abbildung 14: Solidworks Gehäuse Seite

Die Wände, der Boden und der Deckel, welche mit dem Lasercutter zugeschnitten wurden, wurden an 3D-gedruckten Ecken montiert. Nachdem alles montiert und getestet war, war unser Produkt bereit dafür, dass andere ihr Glück versuchen können und dabei merken, ob sie genauso viel Spass haben wie bei herkömmlichen Glücksspielen.



Abbildung 15: Fertiges Gehäuse mit einer Schutzfolie

5.5. Kontrollieren

Die Tests haben wir mit Freunden, Familie und Mitarbeitern im Betrieb gemacht. Mit dem Ergebnis können wir zufrieden sein, denn für die meisten war es verlockend, zu versuchen, das richtige Kästchen zu erraten. Im Allgemeinen war die LED-Methode beliebter als die Motor-Methode. Wir denken, dass dies an der Herausforderung liegt, etwas Schnelles mit den Augen mitzuverfolgen, da es sich belohnender anfühlt, das richtige Ergebnis zu finden. Leider war der Reiz trotzdem nicht so groß wie bei einem Lootcase, da es nichts zu gewinnen gibt.

Bei der Motor-Methode ist die Trefferquote um zwei Prozent höher als bei der LED-Methode. Das erklärt sich dadurch, dass man den Motor dabei hören kann, wie er die Plattform verschiebt, und so schätzen kann, wo sie sich befindet. Die meisten Spieler wurden durch das Blinken der LEDs abgelenkt, was verlockend war, da man dachte, man könnte die LED verfolgen, obwohl die letzte Bewegung der LED nicht sichtbar ist. Bei insgesamt 124 Tests, wurde 22-mal die richtige Box gewählt. Die Durchschnittsquote beweist hiermit, dass die richtige Wahl einer Box der Chance eins zu sechs gleicht.

Es ist möglich, ein Glückspiel zu gestalten, das nur auf Glück basiert, und immer noch Spass machen kann – auch wenn dahinter kein Preis steckt, den man gewinnen kann.

6. Schlusswort

Rückblickend auf unsere Arbeit im vergangenen halben Jahr sind einige Probleme ersichtlich, die sich jedoch schnell wieder gelöst haben.

Zeitlich wurde es gegen Ende recht knapp, obwohl wir absichtlich früh begonnen haben. Die Programmtests, für die bei mehreren Tests zur gleichen Zeit ein bis zwei Wochen eingeplant waren, verzögerten sich zum Teil um einige Wochen. Auch beim Zeichnen des Schemas und der Erstellung des Layouts kamen kleine Fehler zustande, die jedoch schnell wieder behoben werden konnten. Da wir parallel daran arbeiten konnten und eine Zeit lang nicht voneinander abhängig waren, haben wir uns gegenseitig nicht gestört. Gegen Ende kam dann langsam Stress und Druck auf. Dass Mitschüler über ihre fertigen Arbeiten erzählt haben, hat es nicht besser gemacht. Trotz allem wurden wir rechtzeitig fertig und haben viel über Projektplanung und Realisierung solcher Projekte gelernt. Nicht nur das wurde erreicht, sondern wir sind auch stolz auf unser Endresultat. Dass unsere Klasse beeindruckt war, hat uns gezeigt, dass wir stolz sein dürfen.

Einige Dinge waren echte Herausforderungen, da diese meistens nur teilweise funktionierten und uns immer wieder Probleme verursachten. Einerseits die Stromversorgung für den Motor, die RGB-LEDs oder auch ein Teil des Codes.

In der Projektwoche nach den Herbstferien konnten wir die Zeit gut nutzen und waren sehr produktiv. Dies half uns zum Beispiel, den ganzen Code für den Schrittmotor zu testen, das Gehäuse zu designen, den Code zusammenfügen und die ersten Fehler zu beheben.

7. Glossar

8N1: Einstellung des UART-Protokolls, 8 Datenbits, No Paritybit (Zur Kontrolle der Quersumme), 1 Stopbit

AD-Wandler: Analog/Digital-Wandler, gibt eine analoge Spannung als digitalen Wert aus.

Altium: Programm zum Zeichnen und Entwerfen von Leiterplatten.

BC: Brightness Control: Gebündelte Helligkeitssteuerung

Binär: Zahlensystem aus 0 und 1

Clock: Pulsierendes Rechtecksignal, das als Takt für Daten verwendet wird.

GitHub: Cloud mit öffentlichen Zugriff zum Herunterladen von Dateien. Wird meist für die Versionsverwaltung von Software-Entwicklungsprojekten genutzt.

GS: Gray Scale: Einzelne Helligkeitssteuerung

HAL: Hardware Abstraction Layer, Hardwarenahe genormte Programmierbibliothek.

Jogpad: Kontrolltasten für ein Menu.

LED: Light Emitting Diode. Elektrisches Bauteil, das leuchtet.

Lineare Schiene: linear rail. Eine Schiene für eine bewegende Plattform.

Lootcase: Werden in der Onlinespielwelt verwendet. Es sind Kisten mit Inhalt, der einen Wert hat. Die Chance etwas mit hohem Wert zu bekommen ist geringer als die Chance, etwas Günstiges zu bekommen. Viele Onlinespiele verwenden Lootcases, um mehr Geld mit ihren Spielen einzunehmen, da diese mit Spielzeit oder gegen Geld gekauft werden können.

LSB: Least Significant Bit; Niederwertigstes Bit einer Binärzahl.

Mikrokontroller: Programmierbare Steuereinheit.

MSB: Most Significant Bit; Höchstwertigstes Bit einer Binärzahl.

OLED-Display: Display aus Leuchtenden organischen Flüssigkristallen

Openbuilds: Website, um seine mechanischen Projekte öffentlich zu teilen.

PETG: Material zum Drucken für 3D-Drucker.

Print, PCB: Leiterplatte, auf ihr werden elektrische Leitungen gezogen.

QR Code: ein zweidimensionaler Code, der Text, Link, Telefonnummer, GPS Position oder Kontaktdata beinhaltet. Kann mit aktuellen Mobiltelefonen per Kamera gescannt werden.

Rauschen, Noise: physikalisches Phänomen, messbare unregelmässige Stromschwankungen.

Relais: Elektromagnetisch betriebener Schalter

RGB: Rot Grün Blau, meist bei LEDs.

Seriell: Informationen werden nacheinander auf einer Leitung gesendet.

Solidworks: 3D CAD-Tool

Steppermotor; Schrittmotor; Servomotor; Motor mit einstellbarem Winkel. Meistens in Schritten unterteilt.

STM: **STMicroelectronics**, Hersteller von Halbleiterbausteinen.

UART: Universal Asynchronous Receiver Transmitter. Serielles Protokoll für Kommunikation.

9. Quellenverzeichnis

9.1. Abbildungsverzeichnis

- Abb.1 QR-Code GitHub
Abb.2 Blockschaltbild
Abb.3 Funktionen für den Servomotor in der Programmiersprache C
Abb.4 Versorgungs-USB-C
Abb.5 Kommunikations-USB-C
Abb.6 Schema Relaissteuerung
Abb.7 White Noise Generator
Simple White Noise Generator
<https://www.eeweb.com/simple-white-noise-generator/>
[Abrufdatum: 21.10.2020]
Abb.8 Hauptmenu mit Speisung über Kommunikations USB-C
Abb.9 Hauptmenu mit Speisung über den Versorgungs USB-C
Abb.10 Schema UART-zu-Seriell Konverter-Chip, FTDI
Abb.11 LED-Controller Daten
Datenblatt TLC5971 Texas Instruments Seite 27
Abb.12 Testaufbau
Abb.13 Solidworks Gehäuse Unten
Abb.14 Solidworks Gehäuse Seite
Abb.15 Fertiges Gehäuse mit einer Schutzfolie
Abb.16 Schema Buttons & Jogpad
Abb.17 Schema Anschlüsse
Abb.18 Schema LED Kontroller
Abb.19 Schema DC Motor Treiber
Abb.20 Schema OLED Display
Abb.21 Schema USB & Speisung
Abb.22 Schema Reed Sensor
Abb.23 Schema Serial/UART Konverter
Abb.24 Schema Mikrokontroller
Abb.25 Bestückungsplan Bottom
Abb.26 Bestückungsplan Top

9.2. Zitatverzeichnis

Wikipedia (2020), Hütchenspiel,
<https://de.wikipedia.org/wiki/Hütchenspiel>

[Abrufdatum: 17.11.2020]

Schweizer Bund, Kanton Zürich Sicherheitsdirektion Sozialhilfe, Betrugsgesetz,
<http://www.sozialhilfe.zh.ch/Lists/Gesetzestexte/DispItem.aspx?ID=795#:~:text=1%20Wer%20in%20der%20Absicht,ndern%20am%20Vermögen%20schädigt%2C%20wird>,

[Abrufdatum: 21. Oktober 2020]

10. Dank

Für die erhaltene Hilfe danken wir:

Herrn Enrico Malacarne, der sich die Zeit genommen hat, uns beim LED-Controller zu helfen.

Daniel Lütscher für die Möglichkeit, während der Bearbeitung seine Werkzeuge und Materialien zu verwenden.

Johannes Blanck für die Korrektur der Arbeit.

11. Anhang

11.1. Schema

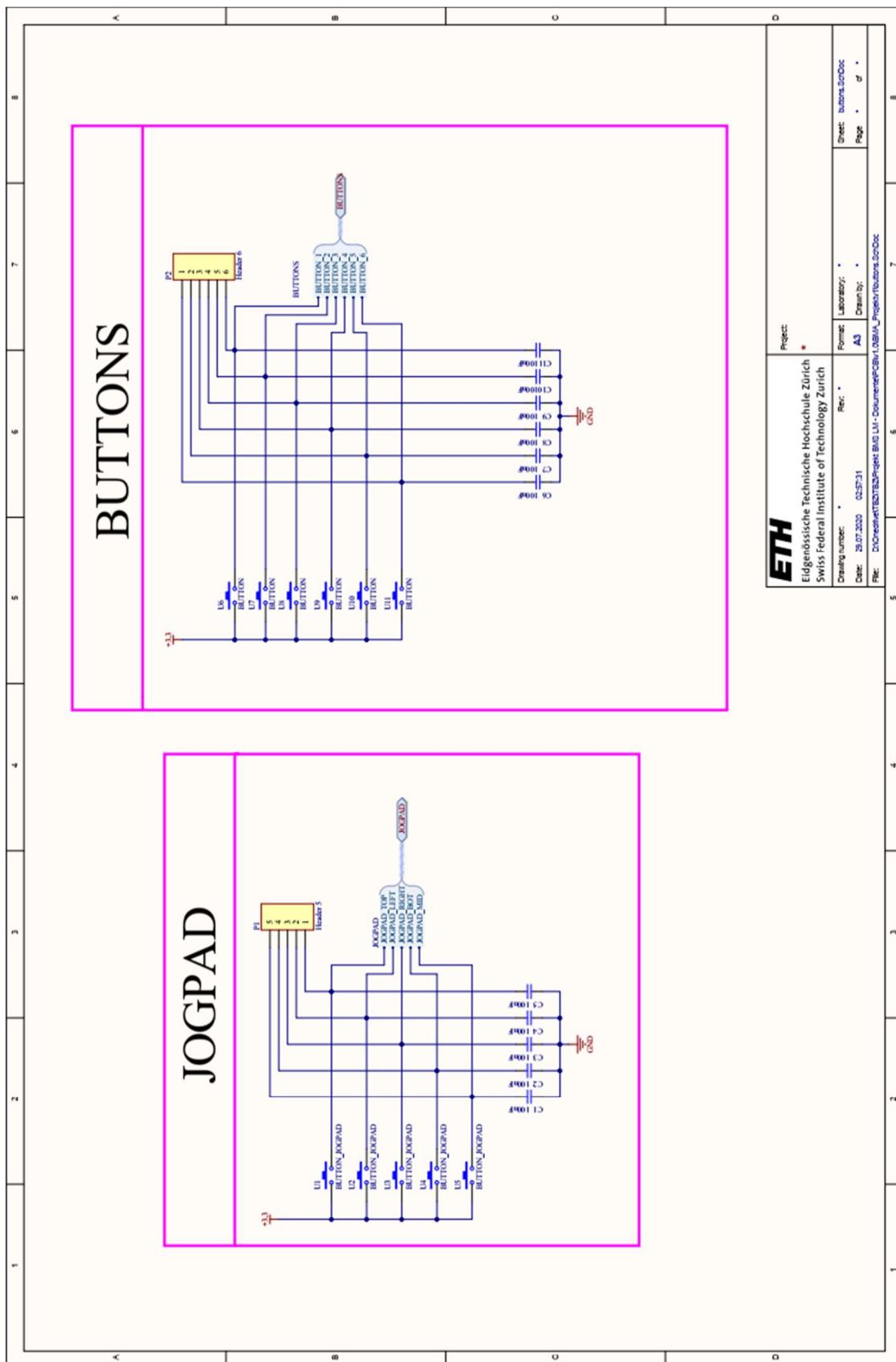


Abbildung 16: Schema Buttons & Jogpads

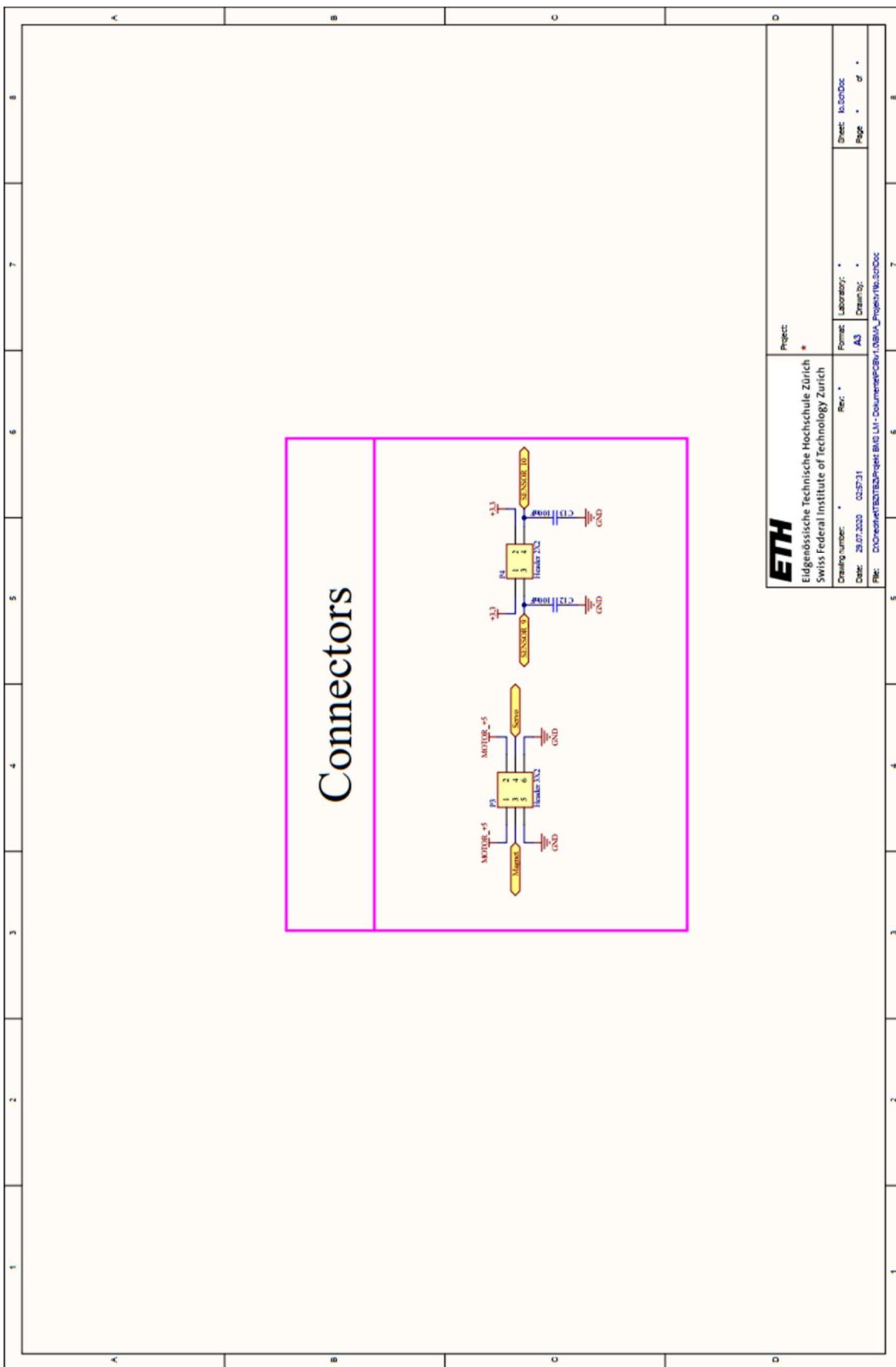


Abbildung 17: Schema Anschlüsse

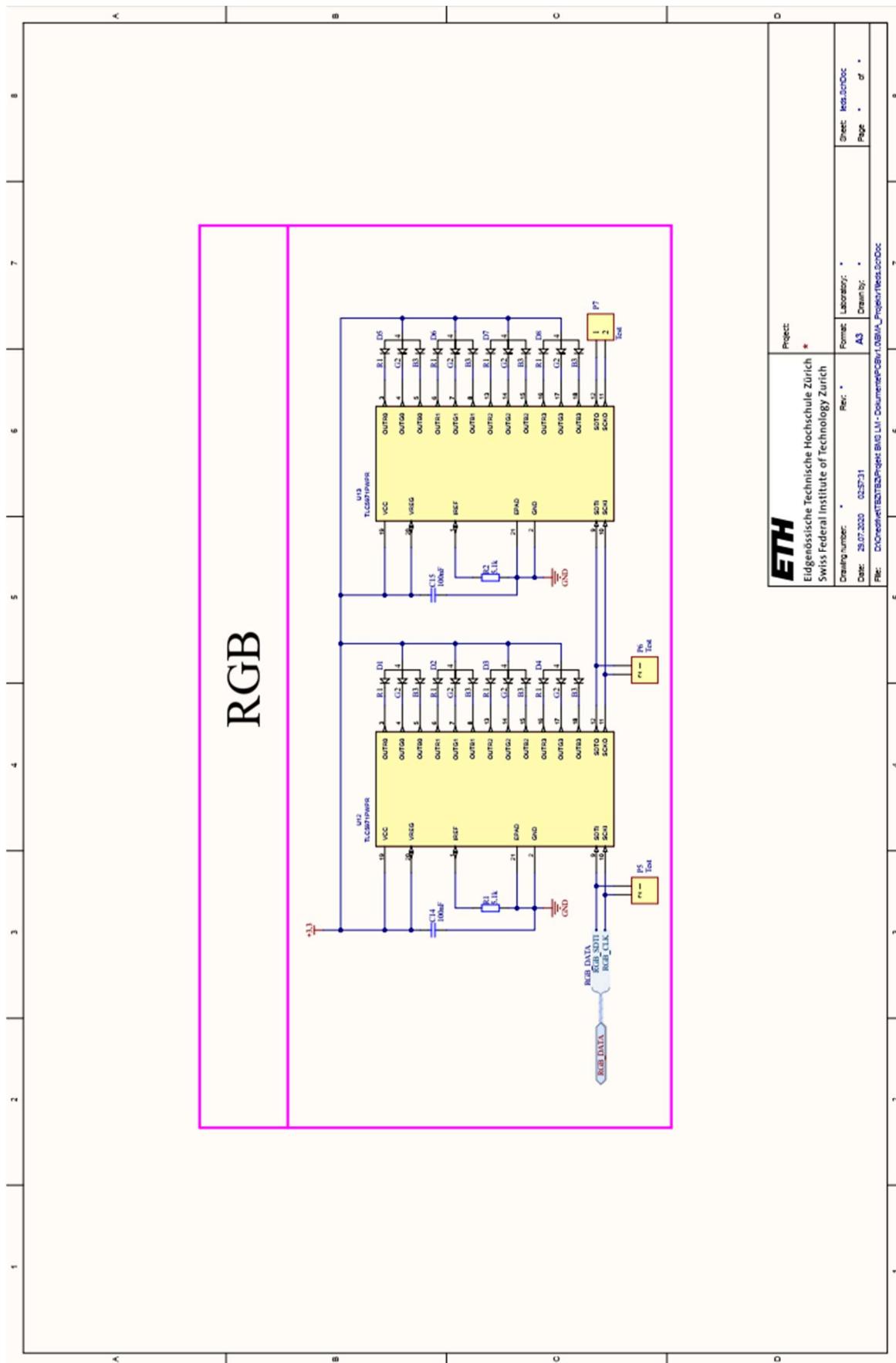


Abbildung 18: Schema LED Kontroller

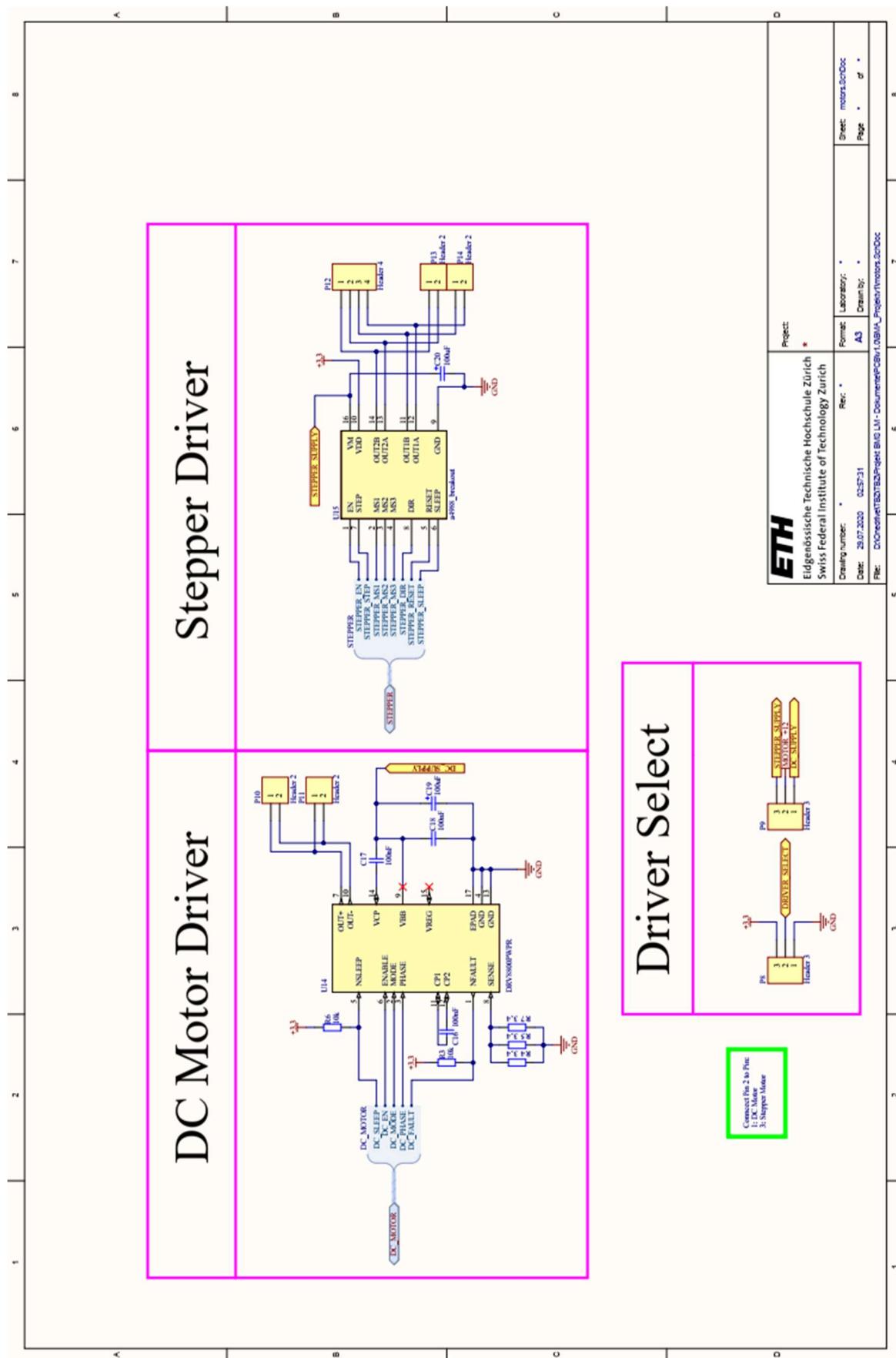


Abbildung 19: Schema DC Motor Treiber

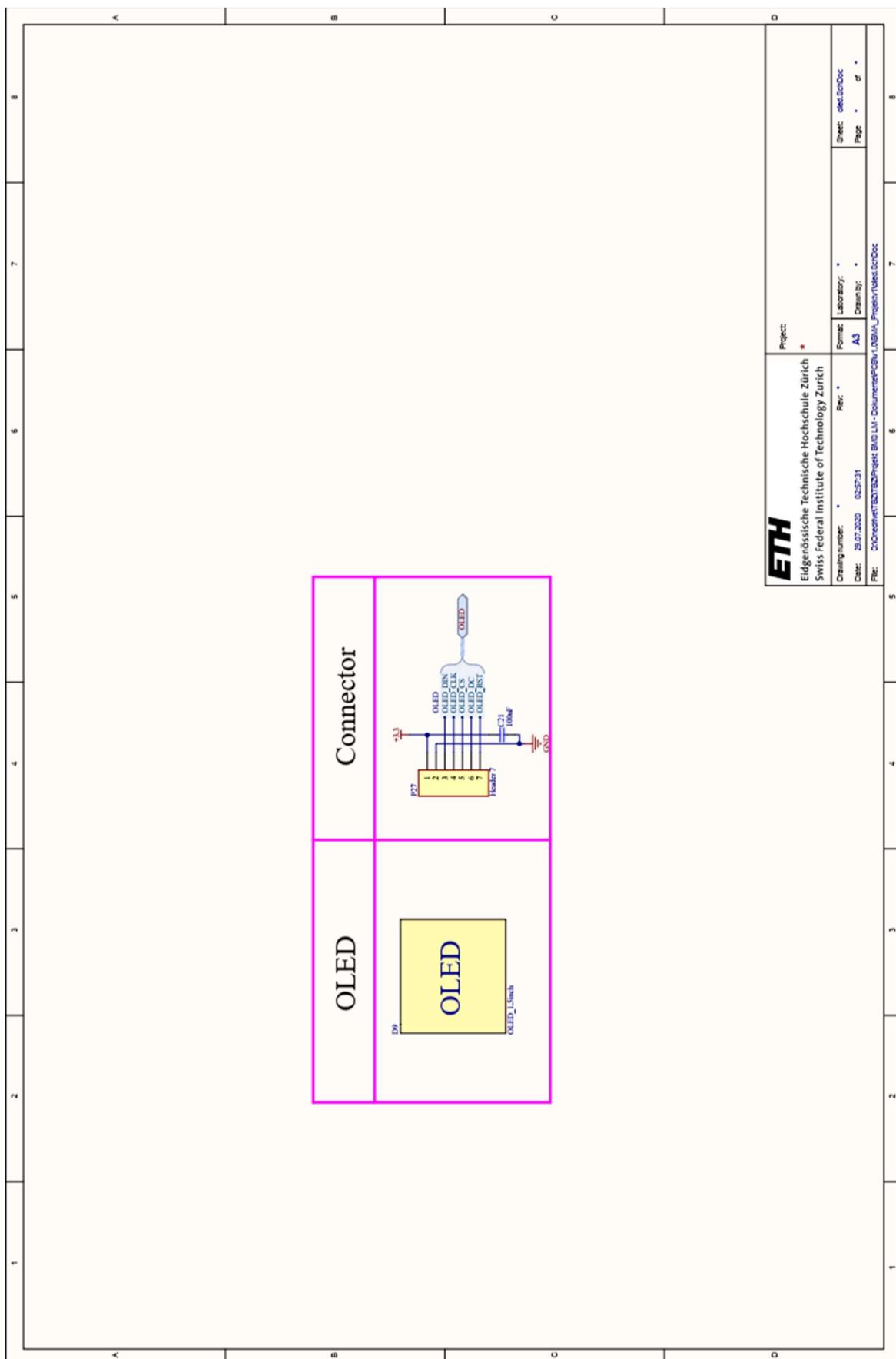


Abbildung 20: Schema OLED Display

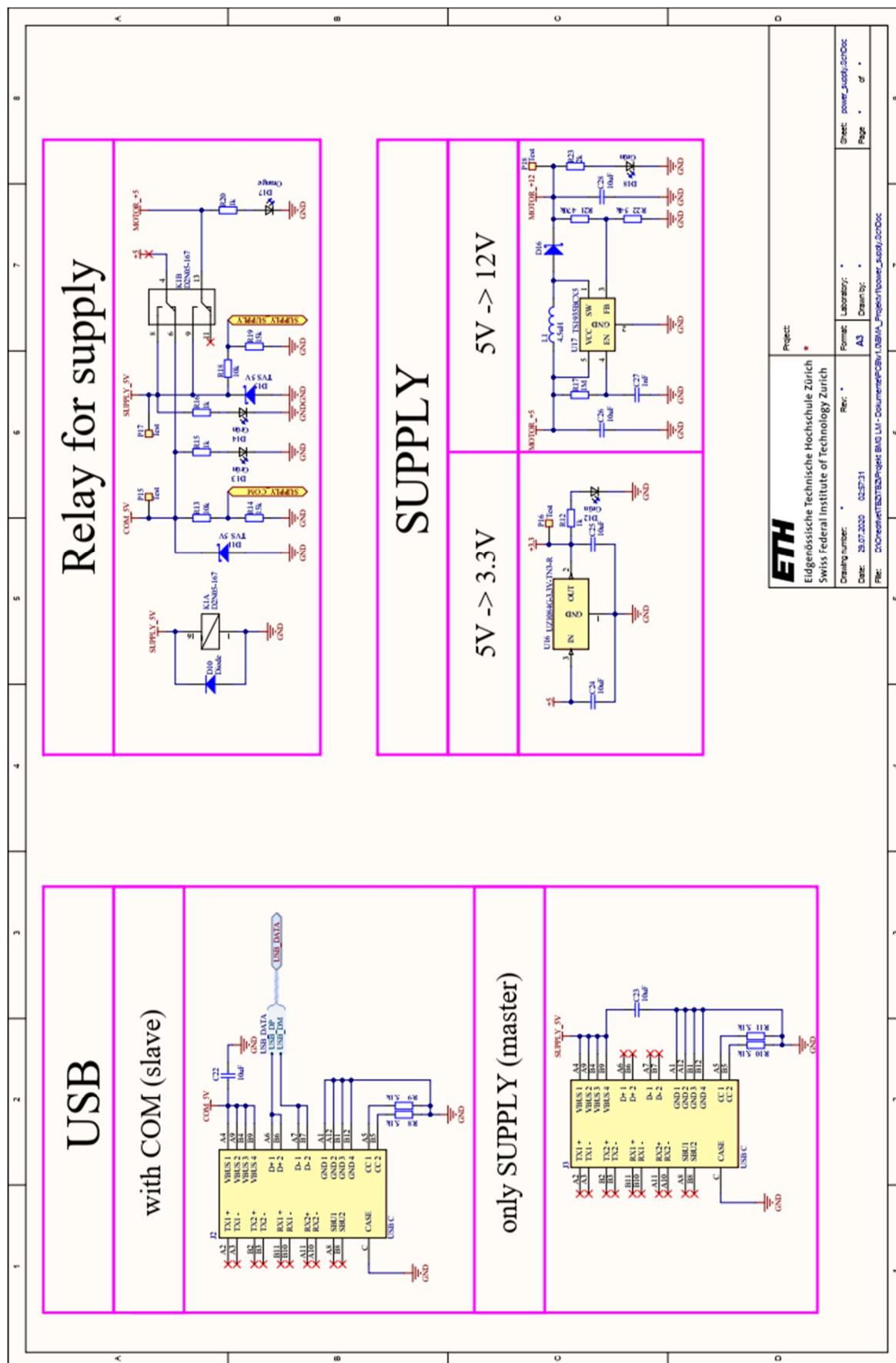


Abbildung 21: Schema USB & Speisung

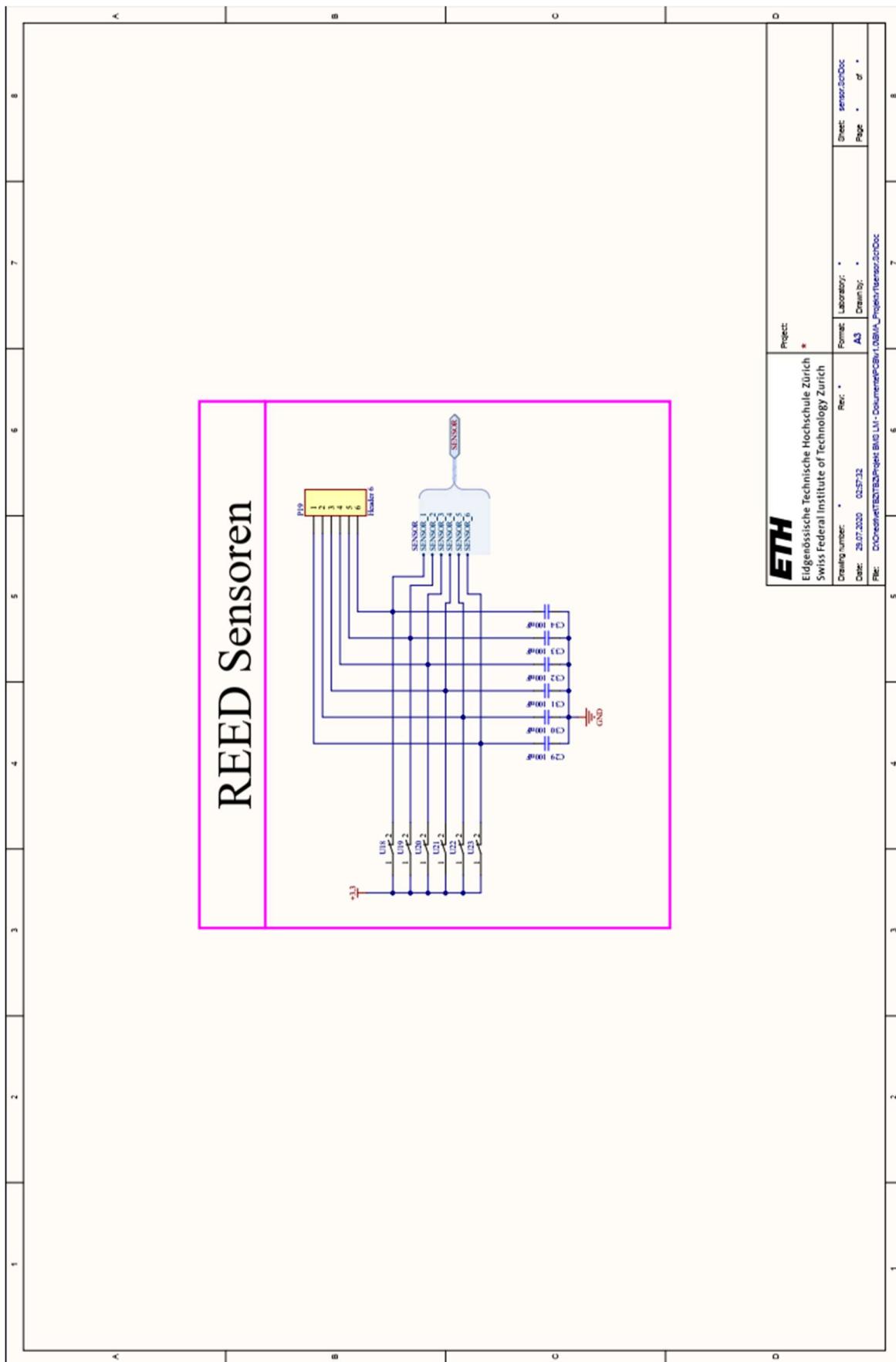


Abbildung 22: Schema Reed Sensor

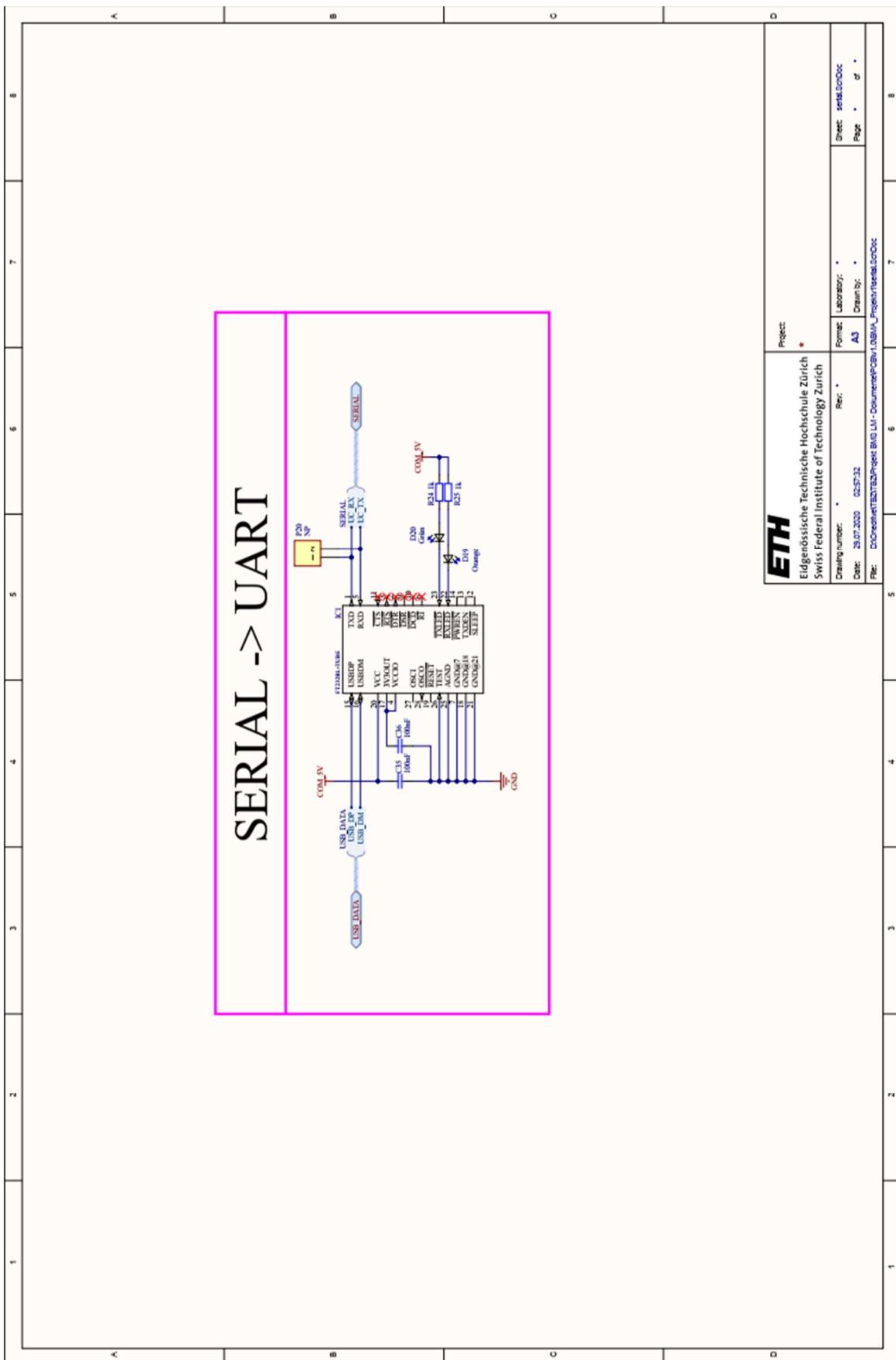


Abbildung 23: Schema Serial/Uart Konverter

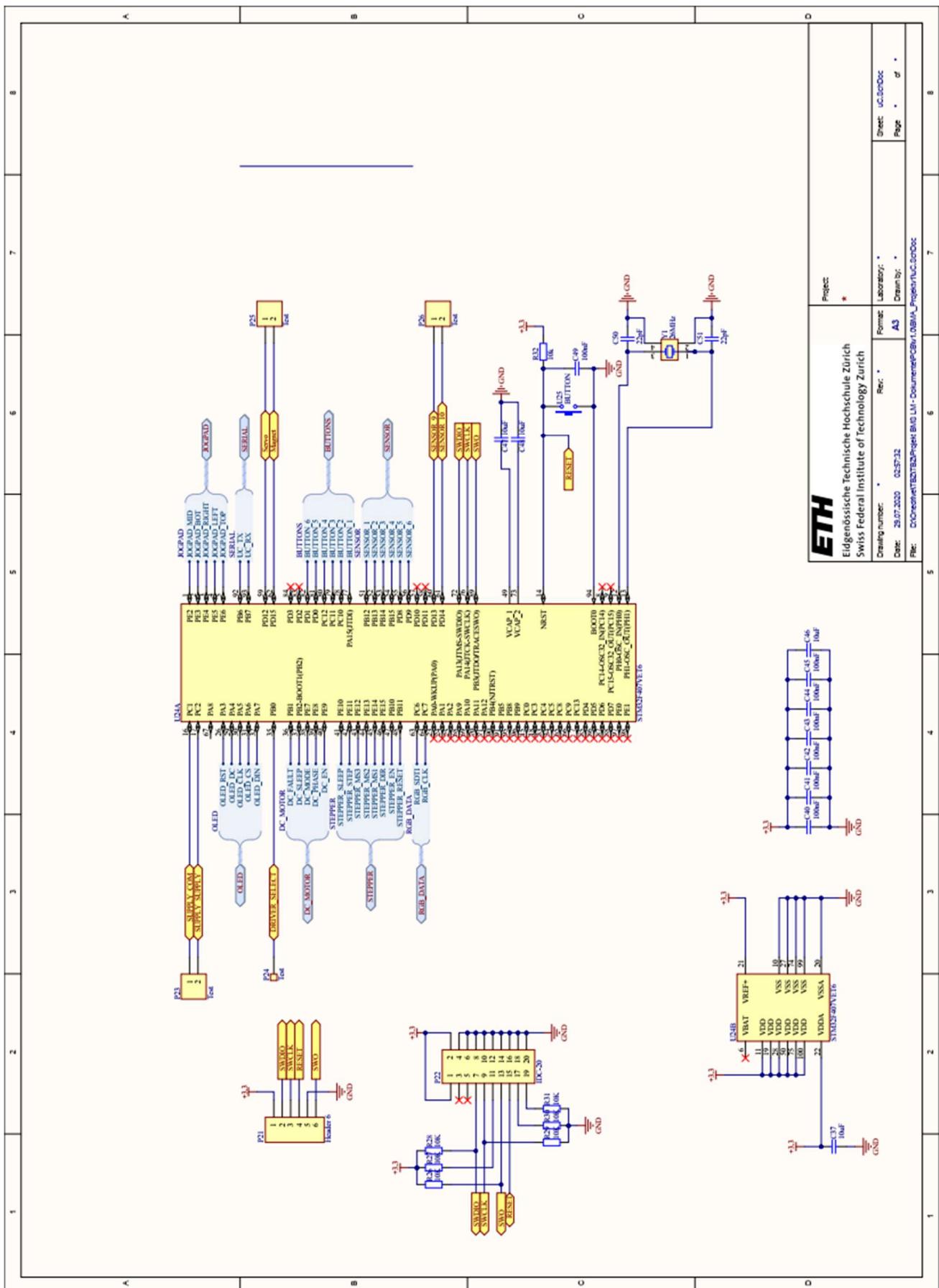


Abbildung 24: Schema Mikrokontroller

11.2. Layout

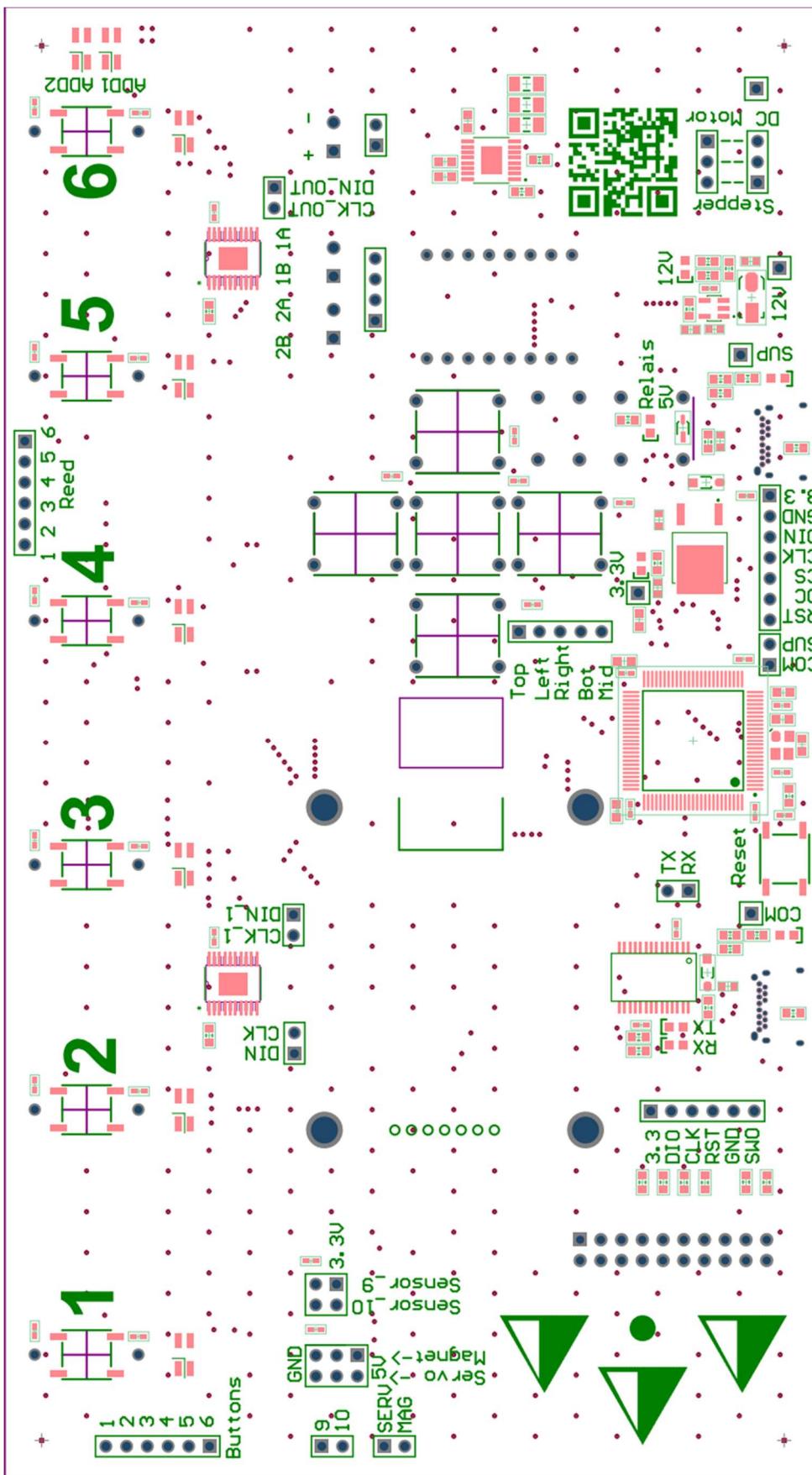


Abbildung 25: Bestückungsplan Bottom

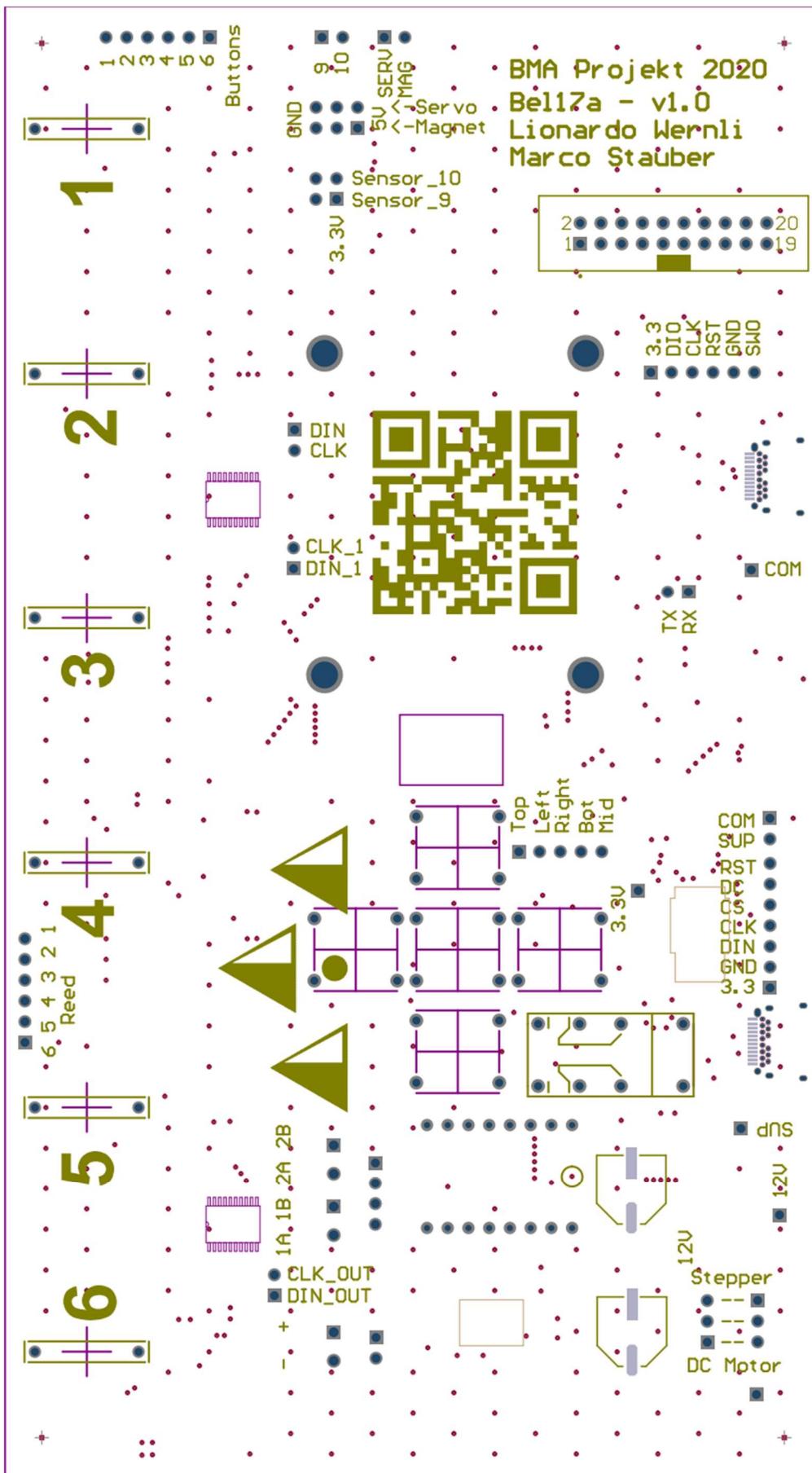


Abbildung 26: Bestückungsplan Top

12. Bescheinigung

Bescheinigung

Name: Leonardo Wernli, Marco Stauber

Klasse: Bel17a

Hiermit bestätige ich, die vorliegende Berufsmaturitätsarbeit mit dem Titel

« Ein Glücksspiel aus reinem Glück »

selbst verfasst zu haben. Informationen aus fremden Quellen sind stets durch die entsprechenden Angaben (Zitate, Quellenverzeichnis) gekennzeichnet.

Ort und Datum: _____

Unterschrift: _____