# Nibo Burger Dokumentation

[1]Dokumentation über das Automatiker Lehrlings Projekt Nibo Burger, welchen wir programmiert und zusammengebaut haben


Abteilung: Elektroabteilung

Verfasser: Flavio Knobel

Pamasol Willi Mäder AG

Driesbüelstrasse 2

Postfach 157 8808

Pfäffikon SZ

Switzerland

Phone +41 55 417 40 40

Fax +41 55 417 40 44

www.pamasol.com info@pamasol.com

[1] https://cdn-reichelt.de/bilder/web/xxl_ws/C160/NIBO_BURGER_01.png

# 1 Inhaltsverzeichnis

# 2 Abbildungsverzeichnis

# 3 Roboter

## 3.1 Mikrocontroller[2]

Mikrocontroller sind Halbleiter Elemente aus der Elektronik. Auf diesen Chips sind verschiedene Sachen meist bereits eingebaut wie zum Beispiel Arbeitsspeicher, Programmspeicher und Peripheriefunktionen. So kann man mithilfe von Analogen und Digitalen ein und Ausgängen fast jedes elektrische Gerät mit solchen Chips versehen werden. Man findet diese Mikrocontroller heute fast überall. Erkennen kann man sie darin das die meisten viele Anschlüsse besitzen und wenn sie keine Kühlung benötigen, eine Schwarze hülle haben wie zum Beispiel auf der Abbildung 1.

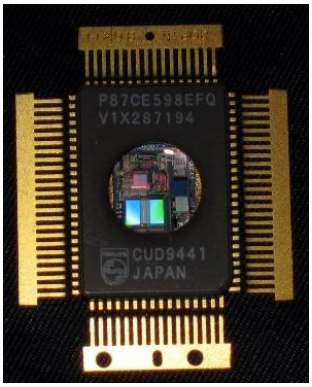


*Abbildung 1 Microcontroller*

## 3.2 3.3 Volt Erzeugung auf der Platine[3]

Der Mikrokontroller IC3 erzeugt die 3.3 Volt. Durch den Kondensator C4 werden die 3.3 Volt noch geglättet. Schaltung ist unten auf dem Schema (Abbildung 2) ersichtlich. Dieses umwandeln hat einen Verlust, welcher in Wärme umgewandelt wird. Allerdings bringt dieser Spannungsregler immer 3.3V am Ausgang da er sich anpasst. Das heisst er verhält sich nicht wie ein Widerstand, sondern er misst seine Eigenspannung und passt sich dann an, sodass am Ausgang immer 3.3V sind.

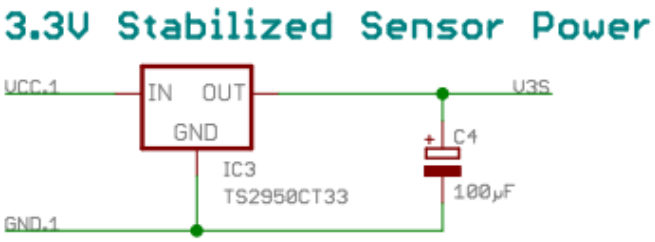


*Abbildung 2: 3.3V Spannungsregler*

[2] https://de.wikipedia.org/wiki/Mikrocontroller#/media/Datei:87CE598.JPG

https://en.wikipedia.org/wiki/Microcontroller

[3] https://github.com/pamasol/Lehrlingsprojekt-Nibo-Burger#3-programming-with-microchip-studio-and-git

## 3.3 PWM Signal[4]

PWM Signal ist ein Signal, das in einer gewissen Frequenz immer Liner ansteigt und abfällt das heisst es gibt nur 2 Zustände Ein und Aus. Durch das ein und ausschalten kann man kleiner Spannung erzeugen und so den Motor de Burgers langsamer drehen lassen. Wenn die Hälfte der Zeit 5V am Motor anliegt und die andere Hälfte der Zeit 0V hat man noch eine Spannung von 2.5V. Dieses Verfahren wird auf dem Abbild 4 gezeigt. Dieser Prozess passiert mehrfach in der Sekunde mit einer Frequenz von 14,7 kHz. Das PWM Signal wird im IC1 und IC2 erzeugt. Das Vorwärts und Rückwärtsfahren wird durch das Kehren der Polarität ermöglicht. Diese wird in einer H Brücke (Abbildung 3) gesteuert. In dieser H Brücke hat es ausserdem noch Dioden verbaut. Diese dienen dazu, falls der Motor im ausgeschalteten Zustand anfängt zu drehen und dann Spannung erzeugt wird diese dort vernichtet, um die Platine zu schützen.

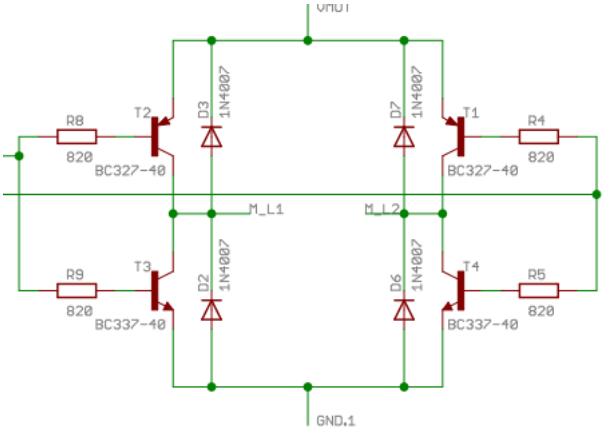


*Abbildung 3 H Brücke*

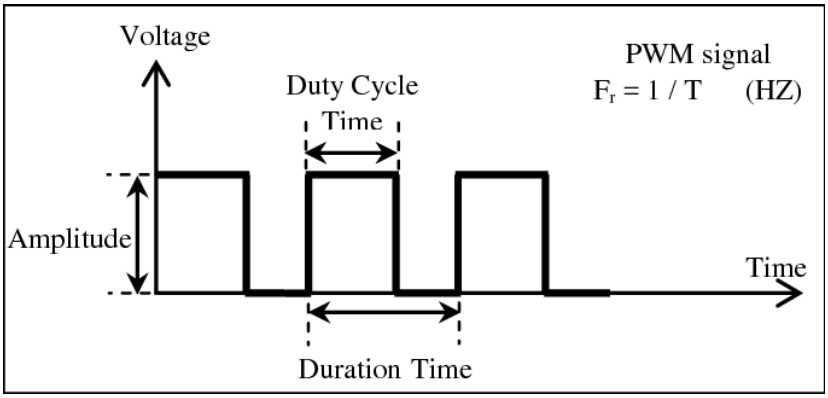


*Abbildung 4 PWM Signal*

[4] https://de.wikipedia.org/wiki/Pulsdauermodulation

https://www.researchgate.net/figure/PWM-signal-with-its-two-basic-time-periods_fig4_271437313

## 3.4 IR Sensor [5]

IR steht für Infrarot das heisst diese Sensoren Funktionieren durch das Unterbrechen oder Beleuchten von Infrarot Licht auf eine Optoelektronische Diode. Diese Technik wird oft bei Lichtschranken eingesetzt. Es kann aber auch sein, dass das Infrarotlicht von etwas reflektiert wird wie auf Abbildung 5 umso zum Beispiel ein Objekt zu erkennen. Durch das kann ein Objekt, welches zwischen dem Licht ist, erfasst werden, indem sie das Licht ganz unterbrechen oder indem sie das Licht an einem Gegenstand reflektiert wird oder auch eine Drehzahl gemessen werden da dann das Licht in regelmässigen Abständen unterbrochen wird. Infrarot Licht ist für das Menschliche Auge unsichtbar.

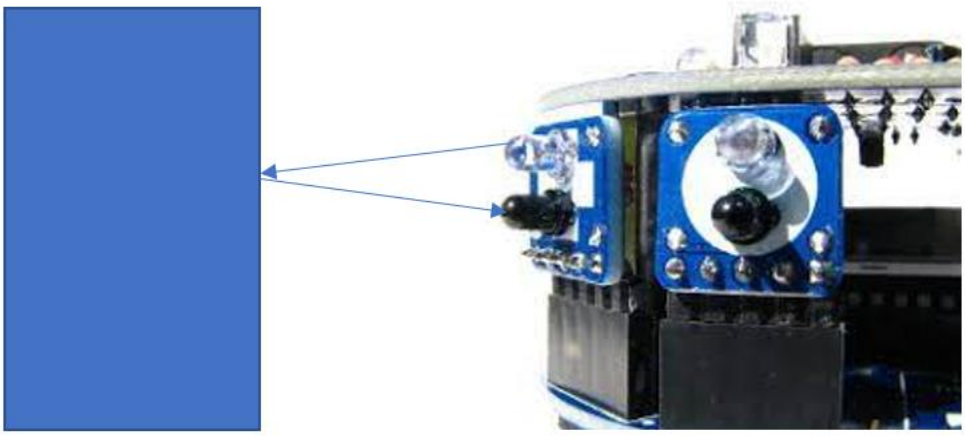

*Abbildung 5 IR Sensor*

## 3.5 RGB Sensoren

R Rot, G Grün und B Blau stehen für die 3 Grundfarben, die zusammen jede Farbe, die der Mensch sehen kann, erzeugen kann. Diese Sensoren, die auf Abbild 6 sind, können auch all diese Farben erkennen, indem sie den Wert von diesen 3 Farben ermitteln und dann ein Analoges Signal ausgeben, welches dann zu Zahlen umgerechnet wird.

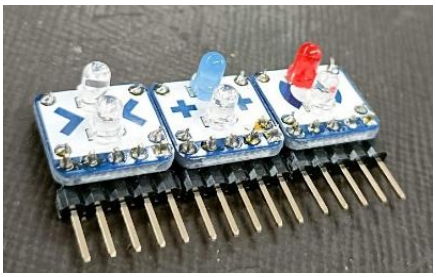

*Abbildung 6 RGB Sensoren*

[5] http://www.nicai-systems.com/de/nibo-burger-spezifikationen

## 3.6 Unterschied der Speicher

### 3.6.1 ROM=Read Only Memory

Der Speicherinhalt kann nur gelesen werden und bleibt nach Wegnahme der Betriebsspannung erhalten. Der Speicherinhalt wird beim Fertigungsprozess (nach Angaben des Anwenders) festgeschrieben. Dies ist nur sinnvoll bei grossen Stückzahlen.

### 3.6.2 PROM=Programmable Read Only Memory

Der Speicherinhalt wird mit speziellen Programmiergeräten eingeschrieben. Der Inhalt ist dann unveränderlich festgehalten. Lange Lebensdauer. Sinnvoll bei kleinen Stückzahlen.

### 3.6.3 EPROM=Ereasable Programmable Read Only Memory

Der Speicherinhalt wird mit speziellen Programmiergeräten eingeschrieben. Die Bausteine können mittels Bestrahlung durch UV-Licht gelöscht und neu beschrieben werden. Der Inhalt der Speicherzellen bleibt für ca. 10 Jahre erhalten. Sinnvoll bei Prototypenentwicklung und Geräten mit geringer Lebensdauer.

### 3.6.4 EEPROM=Electrical Ereasable Programmable Read Only Memory

Wie EPROM, jedoch kann der Speicherinhalt mittels elektrischer Signale gelöscht und anschliessend neu programmiert werden. Es gibt EEPROMs, welche zur Programmierung aus der Schaltung herausgenommen werden müssen, wie auch solche, die in der Schaltung verbleiben können. Nur sinnvoll bei Geräten, welche gelegentliche Programm oder Datenänderungen erfordern (z.B. Speicherprogrammierbare Steuerungen)

### 3.6.5 Flash-EEPROM

Flash-Speicher können beliebig oft beschrieben werden (wie EEPROM). Sie verbleiben grundsätzlich in der Schaltung. Das Lesen und Schreiben erfolgt seriell durch eine Schnittstelle. Sie haben damit die gleiche Funktion wie eine Festplatte. Sinnvoll bei Anwendungen, wo grosse Datenmengen dauerhaft gespeichert werden müssen (z.B. Audiorecorder) oder Inhalte nur gelegentlich geändert werden (z.B. Flash Speicherinder SPS).

### 3.6.6 RAM=Random Access Memory

Das sind die System Schreib und Lesespeicher. Der gesamte Speicherinhalt geht aber nach Wegnahme der Betriebsspannung verloren. Die Speicherung der Daten erfolgt durch Flip Flop Schaltungen. Dies ist ein Speicher für sehr schnelle Anwendungen (z.B. Cache Speicher). Die Speicherung der Daten erfolgt auf der Basis von Kondensatoren. Damit die Daten nicht verloren gehen, ist eigenständiger Refresh des Speichers erforderlich. Geringe Kosten und hohe Speicherdichte pro Chip. Eine typische Anwendung ist der Arbeitsspeicher in PCs. Weiterentwicklungen, vor allem mit dem Ziel einer höheren Geschwindigkeit, sind EDO, SDRAM, DDR, RDRAM und SLDRAM.

### 3.6.7 Speicher im Burger[6]

Der Nibo Burger hat 16KB Programmspeicher (früher ROM, EPROM, EEPROM, heute FLASH) und 1KB Sram. Im Programmspeicher wird das Programm gespeichert um dann während dem Betrieb bearbeitet und ausgeführt werden. Diesen Speicher kann man mehrfach beschreiben und löschen. Im Sram wird das Programm geladen und diese Daten werden nur temporär gespeichert das heisst wenn der Roboter ausgeschaltet wird, sind die Daten weg. Der Burger hat ausserdem einen EEPROM Speicher von 512 Byte, indem die Kalibrierung Informationen gespeichert werden.

[6] https://octopart.com/atmega16a-aur-microchip-77760041?gclid=CjwKCAjw4ayUBhA4EiwATWyBrvnPMhS_ThREaIQW6aqlnQJE8SUWqzLO_RvHP616EPzjvAnrdwoWXhoCnKgQAvD_BwE

# 4 Programmieren

## 4.1 Programmsprache für unseren Nibo Burger[7]

Unser Burger haben wir in C Programmiert. Die der Informatiker Dennis Ritchie in den frühen 1970er Jahren an den Bell Laboratories entwickelte. Seitdem ist sie eine der am weitesten verbreiteten Programmiersprachen.

## 4.2 Variabel[8]

Eine Variable ist In der Programmierung ist ein abstraktes Volumen für eine Größe, welche im Verlauf eines Rechenprozesses auftritt. In diesem Volumen kann ein Wert oder eine Zahl gespeichert werden. Dieser Wert kann auch während, dass Programm läuft, verändert werden. Der Programmierer gibt dieser Variabel einen Namen und für die Maschine ist es dann, wie eine Adresse wo sie Sachen hinschickt.

## 4.3 For Loop[9]

Eine For Schlaufe rechnet so lange eine Zahl hoch oder runter bis die Kondition nicht mehr gegeben ist. Das heisst wenn i<=10 nichtmehr gegeben ist, weil i den Wert 10 hat, hört die For Schleife auf zu rechnen. Rechnen kann man Plus, Minus, Mal oder geteilt. Der Ablauf einer solchen Schleife wird auf Abbildung 7 nochmal gezeigt.

```
for (i=0; i<=10; i++) {}
```

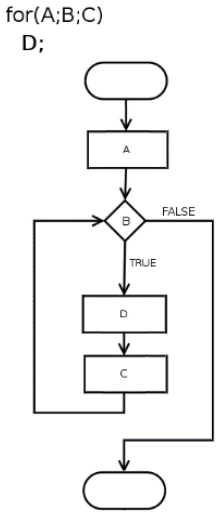


*Abbildung 7 For Loop*

## 4.4 Funktion in C[10]

Eine Funktion ist ein Programmabschnitt, den das Programm nur zu bestimmten Zeiten oder mehrfach ablaufe, soll. Andere Programmsprachen bezeichnen Funktionen auch als Module, Unterprogramme oder Labels.

---

[7] https://de.wikipedia.org/wiki/C_(Programmiersprache)

[8] https://de.wikipedia.org/wiki/Variable_(Programmierung)

[9] https://de.wikipedia.org/wiki/For-Schleife
https://en.wikipedia.org/wiki/For_loop#/media/File:For-loop-diagram.png

[10] https://de.wikibooks.org/wiki/C-Programmierung:_Funktionen

## 4.5 Was macht IC5 im Burger

Ist ein Speicherchip auf ihm wird das Programm gespeichert. Er ist direkt mit dem USB-Port verbunden und regelt die eingehenden Daten. Er ist auch für das Lesen der Daten die vom USB kommen verantwortlich und speichert diese anschliessend. Durchh den USB Port kommen 2 Signal leitungen (im schema Abbildung 9 auf dem Pin PA2 und PA3 des IC5) die Daten übertragen. Da Man mit 2 Leitungen nicht den ganzen Speicher ansteuern darum übernimmt der IC5 diese Aufgaben und speichert die Daten am richtigen Ort. Der IC5 ist der Chip mittig in der Abbildung 8.

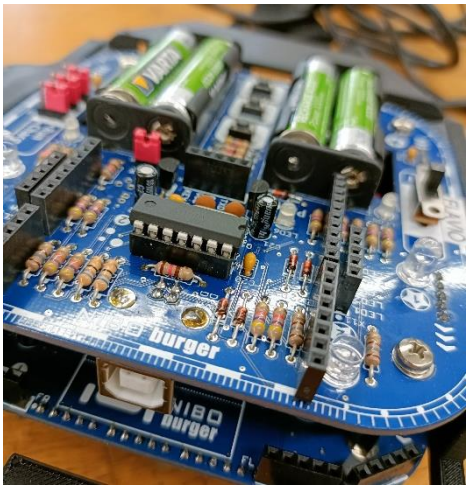


*Abbildung 8: IC5 Platine*

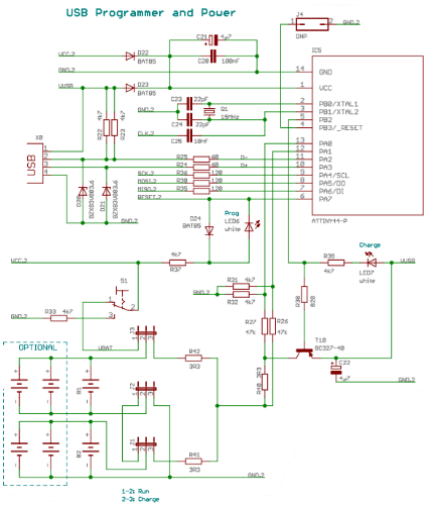


*Abbildung 9: IC5 Schema*

## 4.6 Für was ist Q1[11]

Gibt an IC5 einen Takt vor das dieser in der Richtigen Geschwindigkeit arbeitet ( Im Schema auf Abbildung 10 mit Q1 15MHz eingezeichnet). Die in Quarzoszillator Schaltungen verwendeten Schwingquarze sind meist Kristallplättchen, Stäbe oder Gabeln, die durch elektrische Spannung zu mechanischen Formänderungen gebracht werden können, die wiederum eine elektrische Spannung erzeugen. Der IC5 hat selbst einen 8MHz Takt. Da wir ihn aber mit 15MHz Takten läuft er schneller. Maximal wären 16MHz möglich.

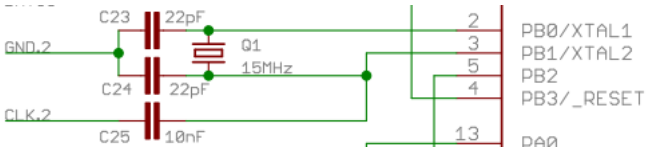


*Abbildung 10: Q1 Schema*

## 4.7 Was ist ein Compiler[12]

Ein Compiler übersetzt die Programmiersprache in einen Maschinen Code, um in anschliessend auf die Maschine zu laden. Wenn ein Programm einen Fehler drin hat, wird dann oft ein Fehler vom Compiler angezeigt. Ein früher Compiler wurde 1949 von der Mathematikerin Grace Hopper konzipiert. Bis zu diesem Zeitpunkt mussten Programmierer direkt Maschinencode erstellen.

[11] https://de.wikipedia.org/wiki/Quarzoszillator

https://github.com/pamasol/Lehrlingsprojekt-Nibo-Burger

[12] https://de.wikipedia.org/wiki/Compiler

# 5 Persönliche Erfahrung

## 5.1 Wie war der Aufbau

Der Aufbau war durch die Anleitung logisch und gut zu verstehen. Auch der Roboter selbst ist sehr übersichtlich aufgebaut, sodass man nicht viel Zeit verliert die richtigen Steckplätze für die einzelnen Bauteile zu suchen.

### 5.1.1 Materialkontrolle

Wir haben am Anfang alle Bauteile sortiert auf ein Blatt geklebt, um einen Überblick zu bekommen (Abbildung 11). Danach muss man die Platinen voneinander trennen (Abbildung 12) bei diesem Schritt muss man sehr vorsichtig sein, dass man nicht einen wichtigen Teil der Platine zerstört. Die Sensoren lässt man am besten in einer Reihe da dann, dass Löten deutlich einfacher fällt.

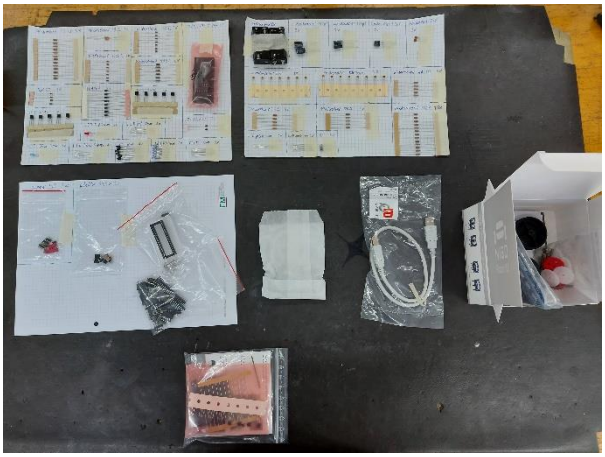

*Abbildung 11 Materialkontrolle Bauteile*

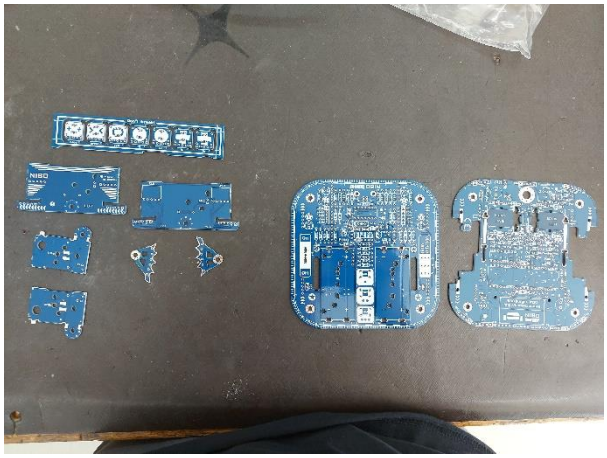

*Abbildung 12 Materialkontrolle Platine*

### 5.1.2 Löten

Beim Löten muss stets beachtet werden das alle Bauteile am richtigen Ort sind und gut verlötet wurden. Ansonsten ist das Löten mit etwas Übung nicht sehr schwierig da man von der Anleitung gut geführt wird und auch die Platinen logisch aufgebaut sind (Beispiel auf Abbildung 13 und 14).

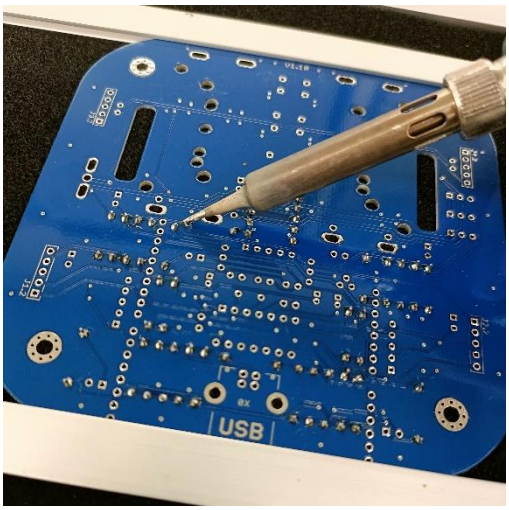


*Abbildung 13 Löten*

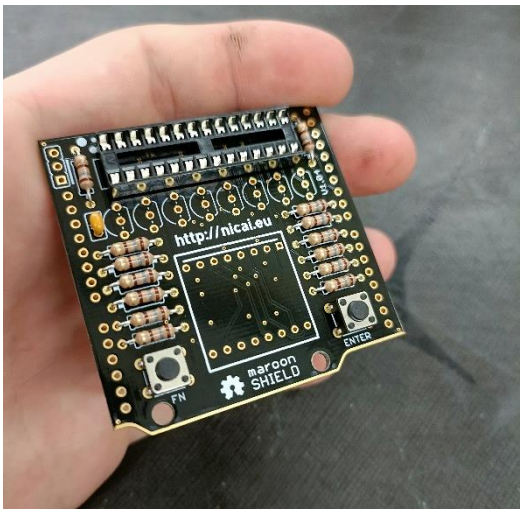


*Abbildung 14 8x8 Anzeige*

### 5.1.3 Aufbau

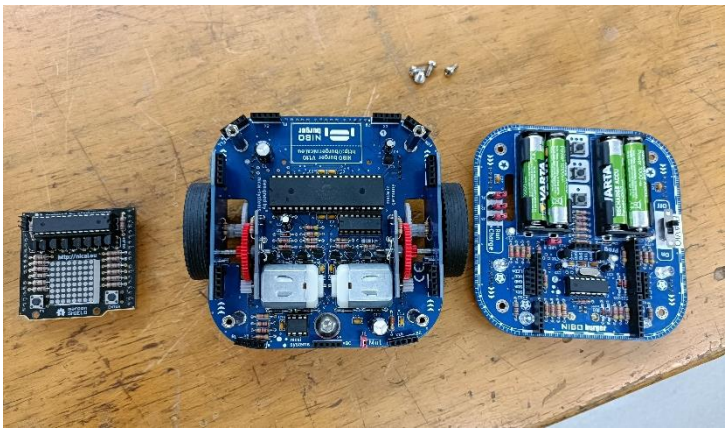

*Abbildung 15 Aufbau*

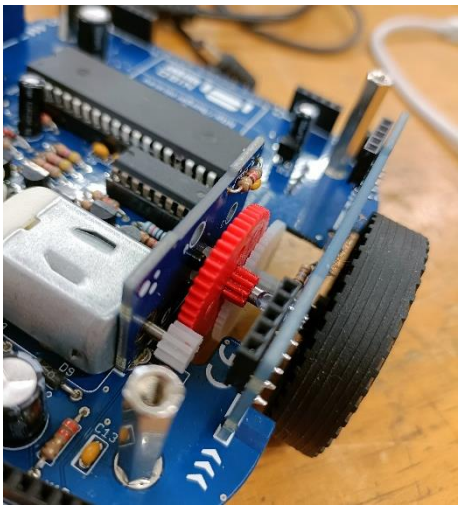

*Abbildung 16 Getriebe*

Der generelle Aufbau ist einfach da wiederum die Anleitung einen Logischen Ablauf hat. Beim Getriebe und Einbau der Motoren ist es etwas schwierig den richtigen Abstand zu erreichen, dass das Getriebe flüssig läuft und auf beiden Seiten gleichschnell dreht. Ich habe die 25:1 Übersetzung genutzt, um schneller fahren zu können. Alternativ wäre noch die 125:1 Übersetzung möglich (Abbildung 15 und 16).

### 5.1.4 Zusatzaufgabe

Unsere Zusatzaufgabe war das optische Verbessern des Roboters mithilfe eines 3D Druckers. Gezeichnet wurde das Model in Autodesk Fusion 365. Ich habe daher einen Umkippschutz, Stossstange und Spoiler an meine Konstruktion gemacht. Durch die Stossstange, die einmal um den Burger rum geht, schütze ich die Sensoren und Räder vor Stössen. Die Beiden Spoiler haben nur einen Optischen nutzen (Abbildung 17).

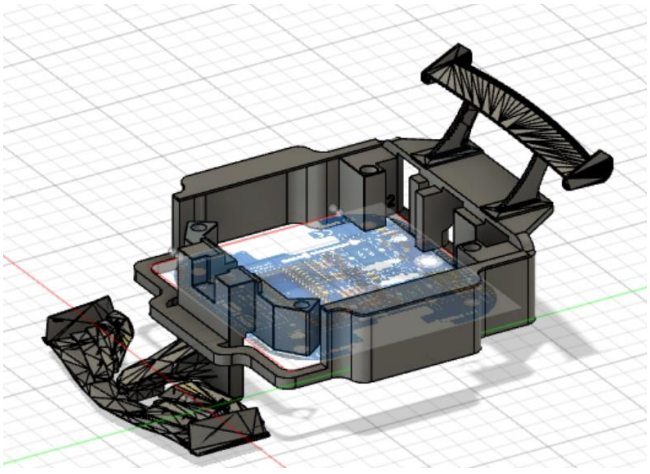

*Abbildung 17 3D Model*

## 5.2 Was muss man beim Programmieren beachten

Farbsensoren müssen vor den Messen kalibriert werden da sonst keine brauchbaren Werte entstehen. Beim gerade fahren am besten etwas programmieren das er sich wieder ausrichtet da die Motoren auf dem rauen Boden nie gleichschnell drehen. Auch immer wieder Speichern und ausprobieren, ob der neue Programmteil funktioniert, kann helfen das man am Schluss nicht in einem grossen Programm Fehler suchen muss.

# 6 Schluss

Das Projekt Nibo Burger hat mir Spass gemacht und es hat mir geholfen das Programmieren in C zu verstehen. Die Installation und Einrichtung der Programme am Anfang waren etwas umständlich aber hat am Schluss dann gut funktioniert. Auch fand ich gut, dass wir mit der Software Git gearbeitet haben, diese habe ich nämlich bis anhin noch nicht gekannt. Obwohl es manchmal Schwierigkeiten gab, ja jeder Roboter seine eigenen Kinderkrankheiten hatte und man diese lösen musste war das Projekt großartig und ein Mehrwert für mein Wissen im Programmieren.