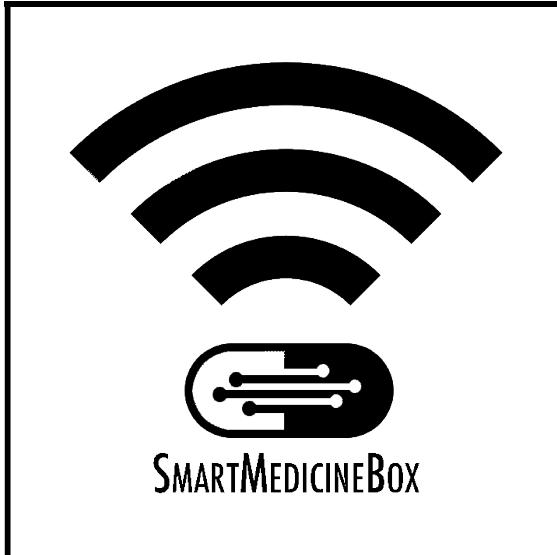


htl bildung mit zukunft	HÖHERE TECHNISCHE BUNDES-LEHR- UND VERSUCHSANSTALT SALZBURG
Abteilung:	Elektronik
Ausbildungsschwerpunkt: Technische Informatik	

DIPLOMARBEIT

SmartMedicineBox



Ausgeführt im Schuljahr 2018/2019

von:

Maximilian Busse 5CHEL

Gregor Maximilian Tockner 5CHEL

Betreuer:

Prof. Dipl.-Ing.

Siegbert Schrempf

Salzburg, am 05.04.2019

Erklärung

Ich erkläre eidesstattlich, dass ich die vorliegende Diplomarbeit mit Ausnahme der ausdrücklich als zitiert oder verwendet gekennzeichneten Teile selbständig erarbeitet und verfasst habe.

Datum/Ort: _____

Unterschrift(en): _____

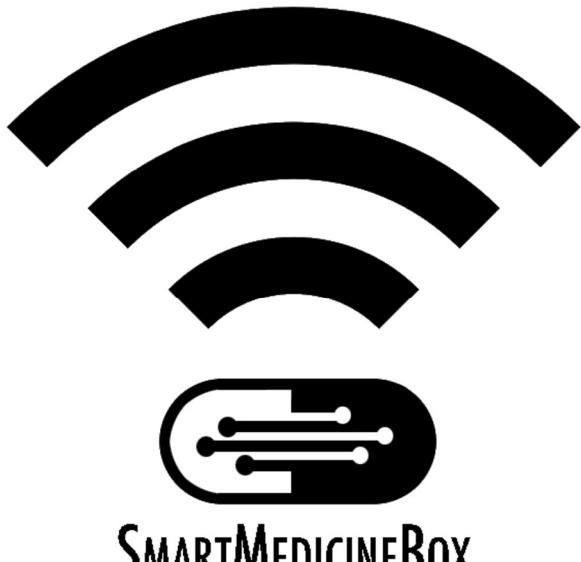
Maximilian Busse

Gregor Maximilian Tockner

DIPLOMARBEIT

Zusammenfassung

Namen der Verfasser/innen	Maximilian Busse Gregor Maximilian Tockner
Jahrgang	5CHEL
Schuljahr	2018/19
Thema der Diplomarbeit	SmartMedicineBox - Entwicklung einer automatisierten Medikamentenversorgung
Aufgabenstellung	Viele Menschen sind im hohen Alter auf Medikamente, Tabletten und Pillen angewiesen, welche täglich in richtiger Reihenfolge und Menge eingenommen werden müssen. Die Dosierung als auch die Einnahmezeiten variieren mit jedem Medikament. Sich die genauen Abläufe zu merken, bereitet manchen Pflegebedürftigen große Probleme. Daher müssen Angehörige, beziehungsweise Pfleger, die tägliche Einnahme kontrollieren und protokollieren.
Realisierung	Es soll eine automatisierte Medikamenten-Box entwickelt werden, welche mittels Smartphone-Steuerung Medikamente zu eigens festgelegten Zeiten ausgeben soll und die Entnahme dieser kontrolliert.
Ergebnisse	Über ein Embedded System wird die Ausgabe der Medikamente gesteuert und die Ausgabezeiten dieser mit Werten aus einer Datenbank abgeglichen, welche über eine Smartphone-Applikation festgelegt werden können.

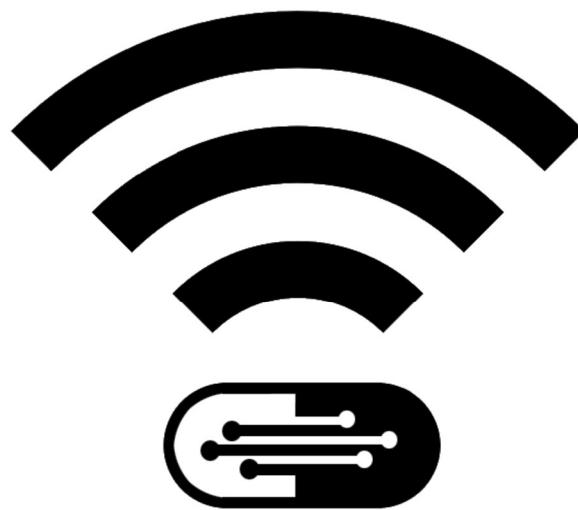
Typische Grafik, Foto etc. (mit Erläuterung)	
Teilnahme an Wettbewerben, Auszeichnungen	AXawards
Möglichkeiten der Einsichtnahme in die Arbeit	Schulbibliothek der HTBLuVA Salzburg
Approbation (Datum / Unterschrift)	Prüferin / Prüfer Direktorin / Direktor Abteilungsvorständin / Abteilungsvorstand

DIPLOMA THESIS

Summary

Author(s)	Maximilian Busse Gregor Maximilian Tockner
Form	5CHEL
Academic year	2018/19
Topic	SmartMedicineBox – Development of an automated medicine supply
Assignment of tasks	A lot of old people are reliant on all sorts of medicine and pills which have to be consumed in correct order and amount on a daily basis. The dosage as well as the time of consumption vary with most medications. Remembering the exact order of consumption often creates problems for people in need for care and supervision. Therefore care workers have to monitor and log the daily intake.
Realization	An automated box that supplies people in need for care should be developed. The box should provide the user with medicine at customizable times and check if the medicine was taken.
Results	An Embedded System controls the distribution of the medicine and compares the time the medicine should be issued with an online-database. The distribution time is customizable via an smartphone-application.

Illustrative graph, photo
(incl. explanation)



Participation in
competitions

Awards

AXawards

Accessibility of
diploma thesis

Library of the HTBLuVA Salzburg

Approval
(Date / Signature)

Examiner

Head of College
Head of Department

Vorwort

In der vorliegenden Diplomarbeit beschäftigen wir uns mit der Entwicklung einer automatisierten Medizinversorgung von hilfsbedürftigen Personen.

Die Projektidee wurde geboren, als wir in einem Zeitungsartikel gelesen haben, dass in Kranken- und Pflegeheimen immer weniger Pfleger zur Verfügung stehen, beziehungsweise diese zu wenig Zeit für ihre Patienten zur Verfügung haben. Wir wollten den Pflegern einen Teil der Arbeit abnehmen und ihnen somit mehr Zeit für ihre restlichen Arbeitsaktivitäten lassen. Die Idee der SmartMedicineBox war somit geboren. Das Ziel war es, eine Medizinbox zu konstruieren, welche Patienten automatisch mit Medikamenten versorgt und die Einnahme kontrolliert und protokolliert. Das Ganze sollte zudem über ein Smartphone individualisierbar und von Familienangehörigen einsehbar sein.

Für die Umsetzung dieser Diplomarbeit wurden verschiedene individuelle Aufgabenstellungen festgelegt. Maximilian beschäftigte sich mit der Ansteuerung des Schrittmotors, welcher die Medizinbox präzise bewegt und somit die Medikamente ausgibt. Eine weitere Aufgabe war es zu überprüfen, ob die Medikamente auch wirklich entnommen wurden und diese Information an ein Smartphone zu senden. Parallel dazu widmete sich Gregor der Steuerung der Medizinbox über eine Smartphone Applikation und der Gehäusekonstruktion.

In diesem Zusammenhang danken wir insbesondere:

- unserem Projektbetreuer **Prof. Dipl.-Ing. Siegbert Schrempf** für seine tatkräftige und fachtheoretische Unterstützung,
- den verantwortlichen Leitern des Printlabors der Elektronikabteilung, Herrn **BEd; FL Ing. Dipl.-Päd. Herbert Pölzer** und **FL Ing. Dipl.-Päd. Norbert Wen** für die Unterstützung bei der Erstellung der notwendigen Leiterplatten,
- Herrn **FL Ing. BEd Robert Pöttinger** und **FL Ing. Dipl.-Päd. Wolfgang Straßl** für ihr Engagement und die Hilfestellung in der Entwicklungsphase des Projekts.

Inhaltsverzeichnis

1 Systemspezifikation.....	11
1.1 Zielbestimmungen.....	11
1.1.1 Musskriterien.....	11
1.1.2 Wunschkriterien	12
1.1.3 Abgrenzungskriterien	12
1.2 Produkteinsatz	12
1.2.1 Anwendungsbereiche.....	12
1.2.2 Zielgruppen.....	12
1.2.3 Betriebsbedingungen	12
1.3 Produktumgebung.....	13
1.3.1 Software.....	13
1.3.2 Hardware	13
1.4 Produktfunktionen	13
1.5 Produktdaten.....	14
1.6 Produktleistungen	14
1.7 Benutzeroberfläche	14
1.7.1 Dialogstruktur.....	14
1.8 Qualitätsbestimmungen.....	15
2 Projektmanagement.....	15
2.1 Überblick	15
2.2 GANTT-Diagramme.....	15
3 Grundlagen und Methoden.....	18
3.1 Schrittmotor.....	18
3.1.1 Verwendung	19
3.1.2 Aufbau	19
3.1.3 Funktionsweise	19
3.1.4 Vollschrittbetrieb.....	20
3.2 Mikrocontroller.....	23
3.2.1 Wahl des Mikrocontrollers	23
3.2.2 Taktzeugung mittels PIC	24
3.2.2.1 Ansteuerung des PIC	24
3.2.3 Programmierung	25
3.3 Motoransteuerung	27
3.3.1 Wahl des Motortreibers	27
3.3.2 Hardware zur Motorsteuerung	28

Diplomarbeit	SmartMedicineBox	2018/2019
3.3.3 Software zur Motoransteuerung	30	
3.3.3.1 Schrittmotorbetriebsart.....	30	
3.3.3.2 Enable und Sleep.....	31	
3.3.4 Zusammenfassung der Variabel-Einstellungen.....	32	
3.4 Sensorik	32	
3.4.1 Fotodiode & IR-LED	32	
3.4.1.1 Verwendung	32	
3.4.1.2 Funktionsweise	33	
3.4.1.3 Beschaltung	35	
3.4.1.4 Programmierung	35	
3.4.2 Hallsensor	36	
3.4.2.1 Funktionsweise	36	
3.4.2.2 Beschaltung	38	
3.5 Gehäusebau	39	
3.5.1 Hauptgehäuse.....	39	
3.5.2 Medizin-Trommel.....	41	
3.5.2.1 Solid Edge.....	41	
3.6 Spannungsversorgung	42	
3.6.1 Erzeugung der Betriebsspannungen	42	
3.6.2 Buck-Converter	43	
3.6.2.1 Funktionsweise	43	
3.6.2.2 Beschaltung	44	
3.6.2.3 Verwendung	44	
3.6.3 Low-Drop-Out Voltage Regler	45	
3.6.3.1 Funktionsweise	45	
3.6.3.2 Beschaltung	45	
3.6.3.3 Verwendung	46	
3.6.4 Basisplatine.....	46	
3.6.4.1 Funktion der Basisplatine	46	
3.7 Software.....	47	
3.7.1 Arduino IDE.....	47	
3.7.2 Android Studio.....	47	
3.7.2.1 Wahl des Programms.....	47	
3.7.2.2 Datenbanken.....	47	
3.7.2.3 Firebase	48	
3.7.2.4 Benutzeroberfläche	50	

Diplomarbeit	SmartMedicineBox	2018/2019
3.7.2.5	Datensendung.....	52
3.7.2.6	Datenempfang am Mikrocontroller	55
3.7.2.7	Datenempfang in der Applikation.....	56
4	Ergebnisse.....	58
4.1	Prototyp.....	58
4.1.1	Steckbrettaufbau	58
4.2	Gefertigte Basisplatine	60
4.3	Gefertigtes Gehäuse	61
5	Literaturverzeichnis	62
6	Abbildungsverzeichnis	63
7	Tabellenverzeichnis.....	64
8	Glossar.....	65
9	Kostenaufstellung.....	65
10	Schlusswort	66
11	Begleitprotokoll gemäß § 9 Abs. 2 PrO	66
11.1	Begleitprotokoll Busse.....	66
11.2	Begleitprotokoll Tockner	68
12	Anhang	69
12.1	Konstruktionspläne.....	69
12.1.1	Motortreiberplatine	69
12.1.2	Basisplatine.....	69

1 Systemspezifikation

Am Smartphone werden die Ausgebzeiten der Medikamente eingegeben und online gespeichert. Die SmartMedicineBox fragt diese Daten nun über Wlan ab und gibt die Medikamente aus.

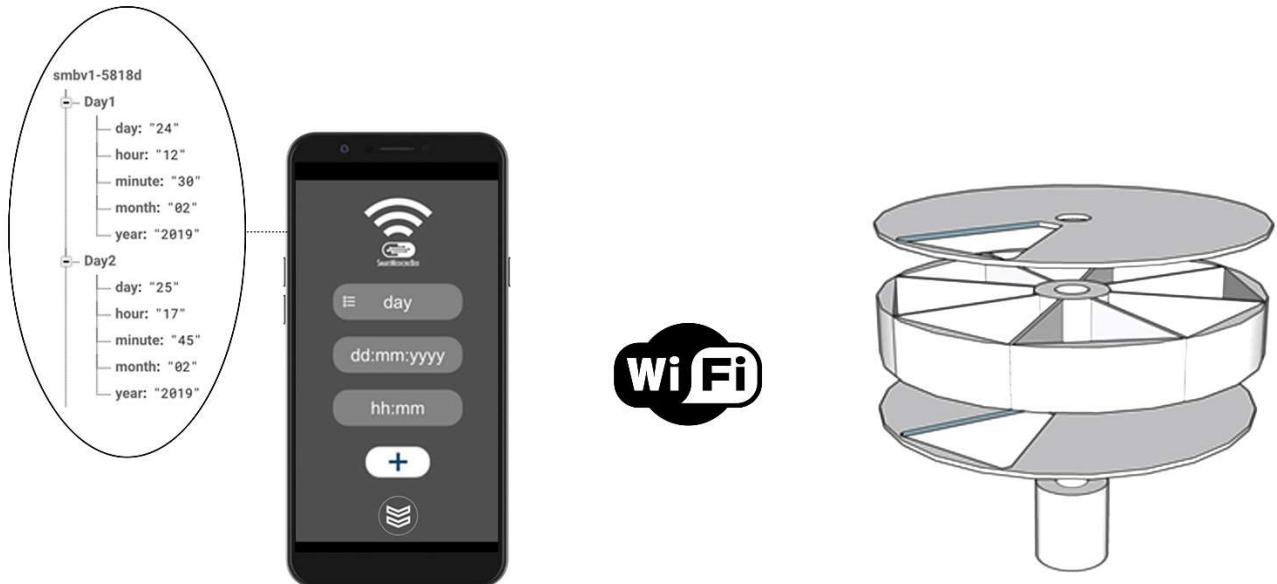


Abbildung 1: Vereinfachtes Blockschaltbild

1.1 Zielbestimmungen

Um zu garantieren, dass Hilfsbedürftige zum richtigen Zeitpunkt die korrekte Anzahl an Medikamenten einnehmen, soll ein Gerät entwickelt werden, welches die Medikamente nach bestimmten Kriterien zur Verfügung stellt und zusätzlich kontrolliert, ob diese auch tatsächlich eingenommen werden.

1.1.1 Musskriterien

- Ein Schrittmotor muss die Medikamentenboxen bewegen können und die richtige Anzahl an Schritten fahren, um die einzelnen Fächer richtig zu positionieren.
- Der Motor muss über eine Smartphone Applikationen gesteuert werden können.
- Die Ausgabezeiten der Medikamente müssen über die Smartphone Applikation frei wählbar sein.
- Die Kommunikation zwischen Hard- und Software muss fehlerfrei verlaufen.

1.1.2 Wunschkriterien

- Die Einnahme der Medikamente sollte überprüft und protokolliert werden können.
- Die Smartphone Applikation soll die Möglichkeit besitzen, mehrere Boxen ansteuern zu können.

1.1.3 Abgrenzungskriterien

- Die SmartMedicineBox soll Patienten keine Medikamente verabreichen. Diese sollen lediglich zur Verfügung gestellt werden.
- Die Box soll nicht automatisiert befüllt werden. Dies muss über eine dafür zuständige Fachkraft erfolgen.

1.2 Produkteinsatz

1.2.1 Anwendungsbereiche

Mit der Medizinbox kann die Medikamentenversorgung für Pflegebedürftige, wie beispielsweise in einem Pflege- oder Altersheim, automatisiert, kontrolliert und leichter von Pflegern gesteuert werden.

1.2.2 Zielgruppen

Medikamentenabhängige müssen sich Reihenfolge und Einnahmezeiten ihrer Medikamente nicht mehr auswendig merken. Durch die SmartMedicineBox werden diese automatisch darauf hingewiesen, wann Medikamente eingenommen werden sollen.

1.2.3 Betriebsbedingungen

- Für den Betrieb wird genügend freier Platz für die Box und eine Steckdose in der Nähe benötigt.
- Eine Smartphone Applikation muss installiert werden.
- Für den aktiven Verwender der Medikamente wird lediglich eine kurze Erklärung der Funktionsweise benötigt.

1.3 Produktumgebung

1.3.1 Software

Zur Steuerung der Medizinbox muss die Smartphone Applikation „MediBox“ installiert werden. Für diese wird eine Internetverbindung benötigt.

1.3.2 Hardware

Die fertig entwickelte Medikamentenbox und ein Smartphone werden benötigt. Um die Box betreiben zu können, wird eine 230V Steckdose benötigt.

1.4 Produktfunktionen

- Motorfahrt
 - Der Motor fährt in beide Richtungen. Die Richtung und Distanz werden in der Smartphone Applikation gewählt und an das Gerät gesendet.
- Referenzpositionierung des Motors
 - Es wird eine Referenzposition über einen Sensor abgefragt und stetig erneuert, um Fahrten immer vom selben Punkt zu starten.
- Ausgabe
 - Die Ausgabe der Medikamente muss präzise durch die Schrittanzahl des Motors bestimmt sein. Die Medikamente werden aus dem Sortierbehälter in eine separat entnehmbare Box befördert.
- Kontrolle der Entnahme
 - Die Entnahme der Medikamente wird über einen Sensor kontrolliert. Erkennt dieser die Entnahme, wird darauffolgend eine Bestätigung der Entnahme an die Smartphone-Applikation gesendet.
- Hardware-Software-Kommunikation
 - Die Smartphone Applikation kann Ausgabezeiten bestimmen. Ein Sensor bei der Ausgabe schickt eine Bestätigung an die Applikation, sobald die Medikamente entnommen werden.

1.5 Produktdaten

- Profile
 - In der Smartphone Applikation kann ein benutzerdefiniertes Profil erstellt werden. In diesem können Ausgabezeiten für den Anwender ausgewählt werden.

1.6 Produktleistungen

- Echtzeit
 - Eingestellte Daten der Smartphone Applikation werden in Echtzeit an die Medizinbox übermittelt und andersherum die Bestätigung der Medikamentenentnahme an die Applikation.

1.7 Benutzeroberfläche

Das User Interface für die Pfleger sollte einfach zu bedienen und überschaubar sein. Ebenfalls ist dieses über jedes Android-fähige Smartphone bedienbar.

1.7.1 Dialogstruktur

In der Applikation können über einen Regler individuelle Zeiten für den Benutzer der Medizinbox angelegt werden. Zudem kann der Status der Entnahme der Medikamente abgefragt und aus einer Datenbank ausgelesen werden.

1.8 Qualitätsbestimmungen

Die folgende Tabelle veranschaulicht Prioritäten in der Qualitätsanforderungen.

	sehr wichtig	wichtig	weniger wichtig	unwichtig
Robustheit		x		
Zuverlässigkeit	x			
Korrektheit	x			
Benutzerfreundlichkeit	x			
Effizienz			x	
Portierbarkeit			x	
Kompatibilität			x	

Tabelle 1: Qualitätsanforderungen

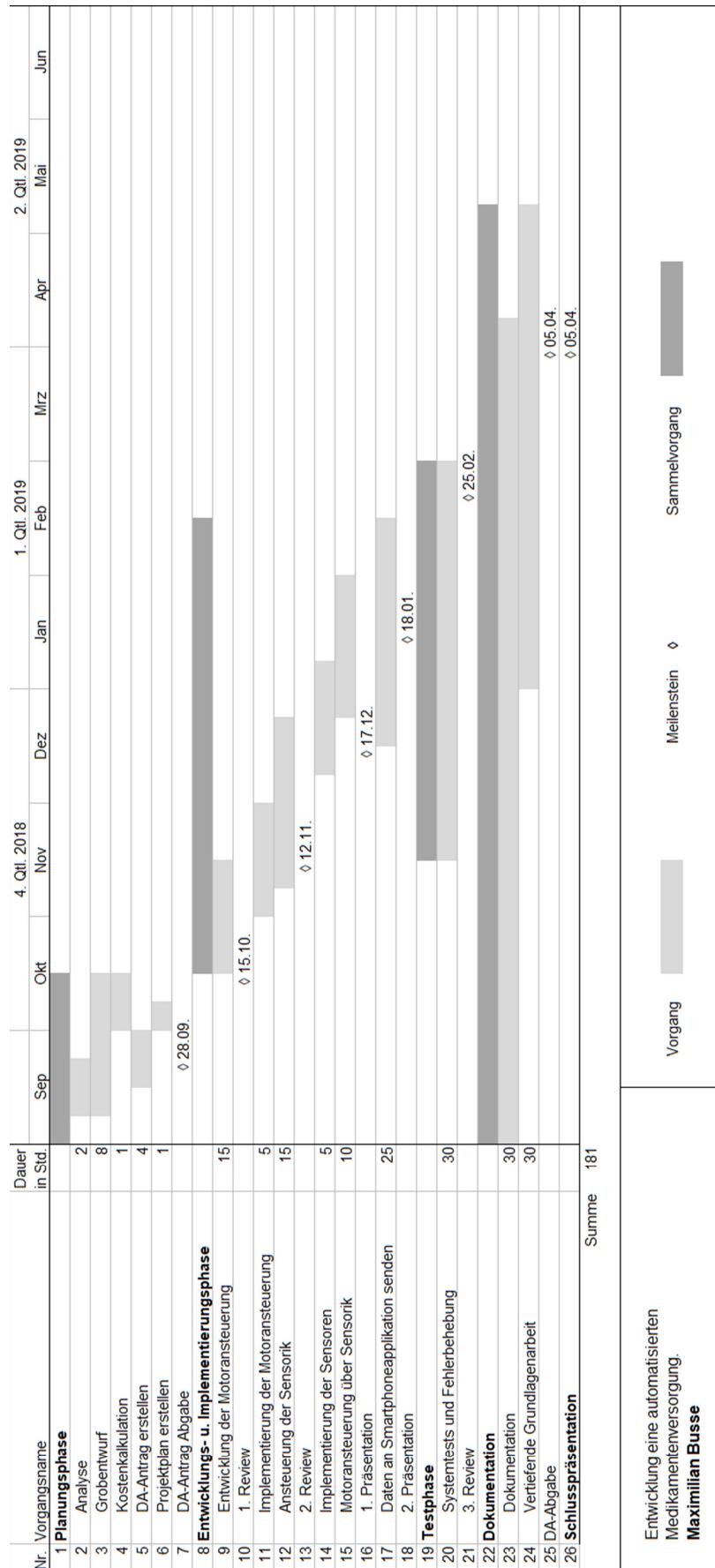
2 Projektmanagement

2.1 Überblick

- Maximilian Busse:
 - Auswertung der Sensordaten
 - Programmierung der Software zur Motoransteuerung
 - Auswertung der Signale der Handyapplikation
 - Projektmanagement
- Gregor Tockner:
 - Entwicklung der Smartphone Applikation
 - Mechanische Konstruktion
 - Schaltungsentwicklung und Implementierung
 - Gesamtaufbau

2.2 GANTT-Diagramme

Zur Veranschaulichung der Arbeitseinteilung wird pro Projektteilnehmer jeweils ein GANTT-Diagramm erstellt.

**Abbildung 2: GANTT-Diagramm Busse Max**

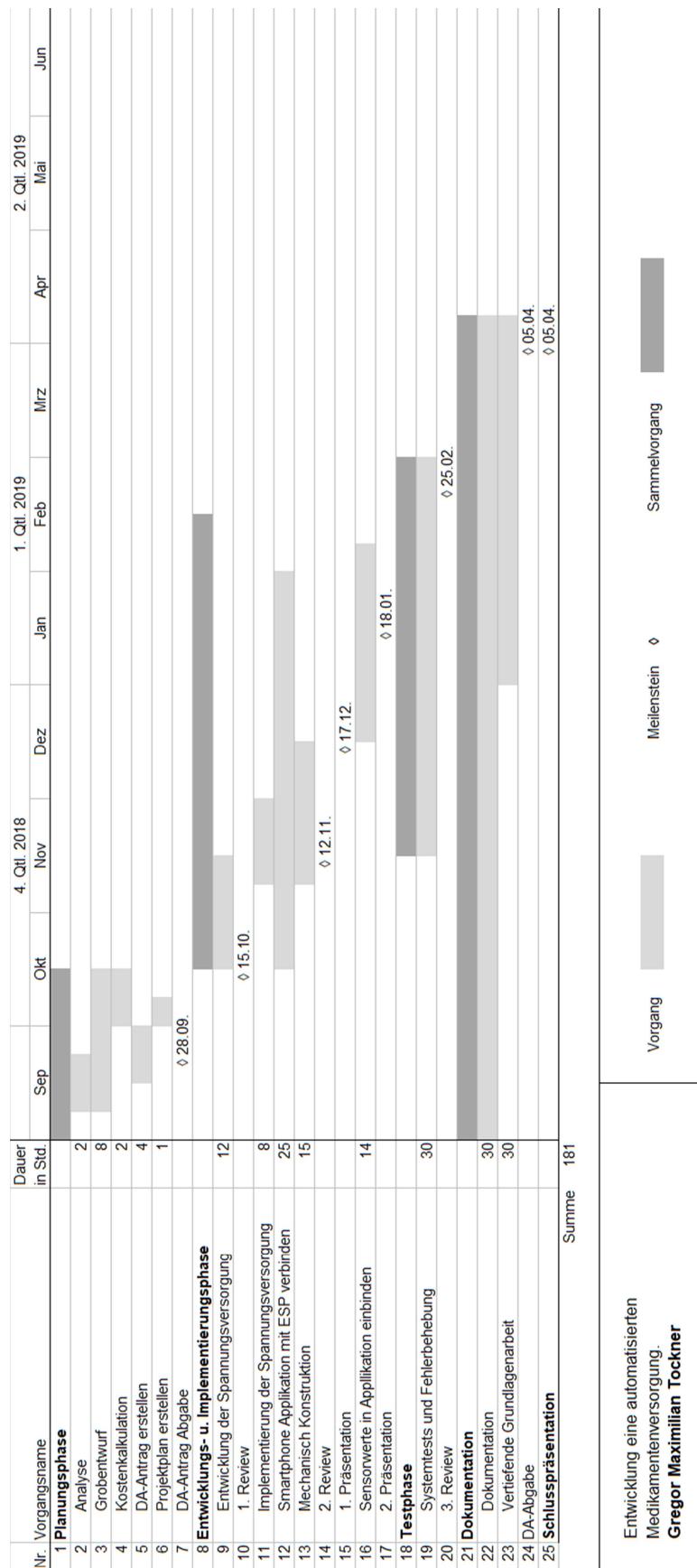


Abbildung 3: GANTT-Diagramm Tockner Gregor

3 Grundlagen und Methoden

Die SMB (SmartMedicineBox) wird über eine Smartphone-Applikation gesteuert. In dieser Applikation wird festgelegt, wann Medikamente ausgegeben werden sollen. Diese Daten werden in einer Datenbank gespeichert. Ein Mikrocontroller mit WiFi-Verbindung lädt diese Daten in seinen Speicher. Über eigens festgelegte Programmfunctionen, wird ein Schrittmotor angesteuert. Dieser dreht eine Trommel mit medikamentenbefüllten Fächern, sobald die Uhrzeit und das Datum der in der Datenbank angegebenen Zeit entsprechen. Die Medikamente fallen bei übereinstimmenden Daten in eine entnehmbare Schachtel. Über Sensoren wird überprüft, ob die Schachtel entnommen wird. Informationen über die Entnahme der Schachtel werden über den Mikrocontroller zurück an die Datenbank und somit an das Smartphone gesendet. Die nachstehende Abbildung veranschaulicht dies exemplarisch.

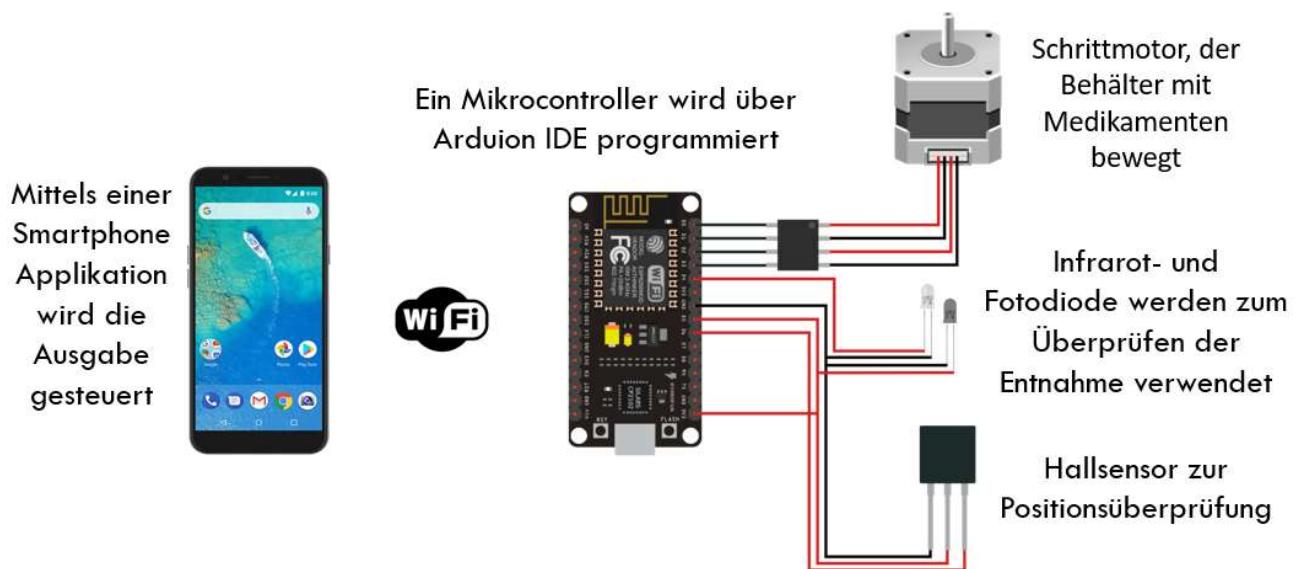


Abbildung 4: Systemkomponenten der SmartMedicineBox

3.1 Schrittmotor

Als besonders ausschlaggebend ist die Schrittmotoransteuerung bei diesem Projekt zu betrachten, da für diese eine eigens gefertigte Platine und Programmcode erforderlich sind.

3.1.1 Verwendung

Um eine präzise Positionierung einzelner Fächer der SmartMedicineBox zu ermöglichen, wird ein Schrittmotor verwendet. Dieser kann eine bestimmte Anzahl an Schritten, welche im Programmcode bestimmt werden, drehen. Gegenüber einem Servomotor kann der Schrittmotor somit viel exakter angesteuert werden.

3.1.2 Aufbau

Permanentmagnetschrittmotoren besitzen einen Stator - dieser ist fest verbaut und somit nicht beweglich - und einen beweglichen Rotor, welcher sich innerhalb des Stators dreht. Eine Rotation wird über den Stromfluss innerhalb der Stator-Spulen und den durch diesen Strom angesprochenen Magneten des Rotors ermöglicht. Diese Magneten weisen abwechselnd einen Nord- und Südpol auf. Über dieses Magnetfeld kann somit der Rotor in beide Richtungen und verschieden schnell gedreht werden.¹

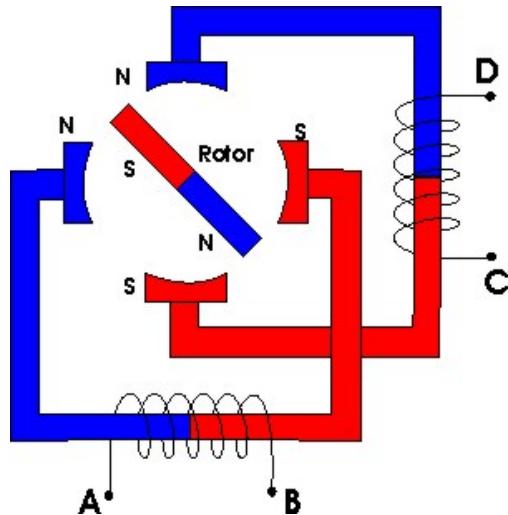


Abbildung 5: Bipolar-Schrittmotor (Quelle: mikrocontroller.net)

3.1.3 Funktionsweise

Wird an den Spulen des Motors Spannung angelegt, so dreht sich der Motor ein kleines Stück. Beim Bipolar-Schrittmotor gibt es zwei Spulen und somit vier Anschlüsse, wovon jeweils immer zwei gleichzeitig angesteuert werden und den Motor somit um 1,8 Grad

¹ Vgl.: <https://www.mikrocontroller.net/mc-project/Pages/Robotik/Mechanik/mechanik.html>; S. 2; [05.11.2018 08:32]

drehen. Um eine volle Drehung zu bekommen, muss der Motor also 200 Schritte drehen ($200 * 1,8$ Grad = 360 Grad). Nach jedem Schritt wird eine andere Kombination der Anschlussmöglichkeiten gewählt, wobei immer zwei Anschlüsse aktiv – also unter Spannung – und die anderen zwei auf null Volt sein müssen. Diese Beschaltung wird als Bipolar-Betrieb bezeichnet, da für die vier Anschlüsse des Motors bipolare (zweipolige) Wechselschalter verwendet werden.

3.1.4 Vollschrittbetrieb

Im Vollschrittbetrieb (oder auch Normal-Betrieb genannt) werden immer jeweils zwei Anschlüsse des Motors unter Strom gesetzt. Dies veranschaulicht die folgende Abbildung:

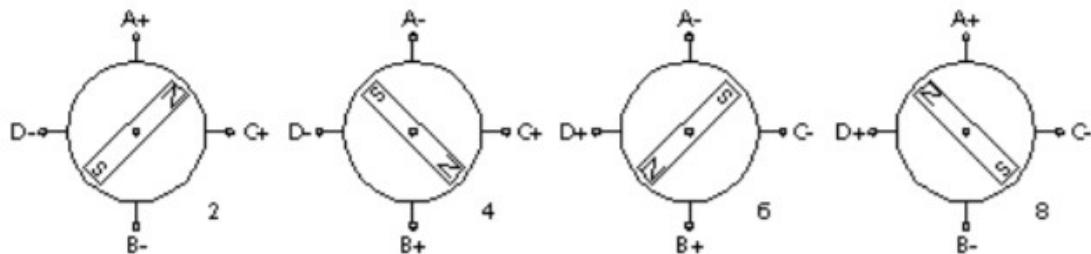


Abbildung 6: Schrittmotor Normal-Betrieb²

In folgender Tabelle sind die sich aus Abbildung 6 ergebenen Signale mit „0“ für 0V und „1“ für 12V beschrieben. In der nachstehenden Tabelle sind die typischen Phasenübergänge des bipolaren Schrittmotors von einem Zustand zum nächsten veranschaulicht.

Schritt	Phasen			
	A	B	C	D
1	1	0	1	0
2	0	1	1	0
3	0	1	0	1
4	1	0	0	1
1 ...	1	0	1	0

² Vgl.: https://wiki.ntb.ch/infoportal/_media/hardware/sysp/bauteile/schrittmotor_kurz_erklaert_d.pdf
[05.11.2018 14:38]

Tabelle 2: Phasen des Schrittmotors

Die Signale, welche an die vier Eingänge des Schrittmotors (A+, A-, B-, B+) gesendet werden, sehen folgendermaßen aus:

**Abbildung 7: Zeitliniendiagramm des bipolaren Schrittmotors**

Wie im rechten Bereich der Abbildung 7 unter dem Menüpunkt „Messungen“ zu sehen ist, wurde die gewünschte Spannung von 12V mit einem Toleranzspielraum annähernd erreicht mit 11,7V.

Abbildung 8 zeigt zusätzlich noch das „Step-Signal“, welches vom ESP an den IC-Treiber gesendet wird. Für erste Testzwecke wurde eine Periodendauer von 6ms gewählt, da der Motor mit dieser Periodendauer „flüssig“ läuft und ein pulsweitenmoduliertes Signal nicht zwingend notwendig ist. (Über ein pulsweitenmoduliertes Signal kann der Motor beschleunigen und abbremsen.) Somit ändern sich bei jeder steigenden Flanke des Step-Signals zwei Zustände der vier Zuleitungen zum Motor. Jede steigende Flanke entspricht

also auch einem Schritt des Motors von $1,8^\circ$, somit ergibt sich nach 200 Schritten eine volle Umdrehung von 360° ($= 1,8^\circ * 200 \text{ steps}$).

Die Medizinbox verfügt über acht Fächer, also eine 45° Drehung pro Fach. Damit ergibt sich die folgende Schrittzahl pro Fach bei einer 1:1-Übersetzung:

$$\frac{360^\circ}{200 \text{ steps}} = \frac{45^\circ}{x \text{ steps}} \Rightarrow x \text{ steps} = \frac{45^\circ * 200 \text{ steps}}{360^\circ} = 25 \text{ steps}$$

Ein Fach der Medizinbox ist somit 25 Schritte „lang“.

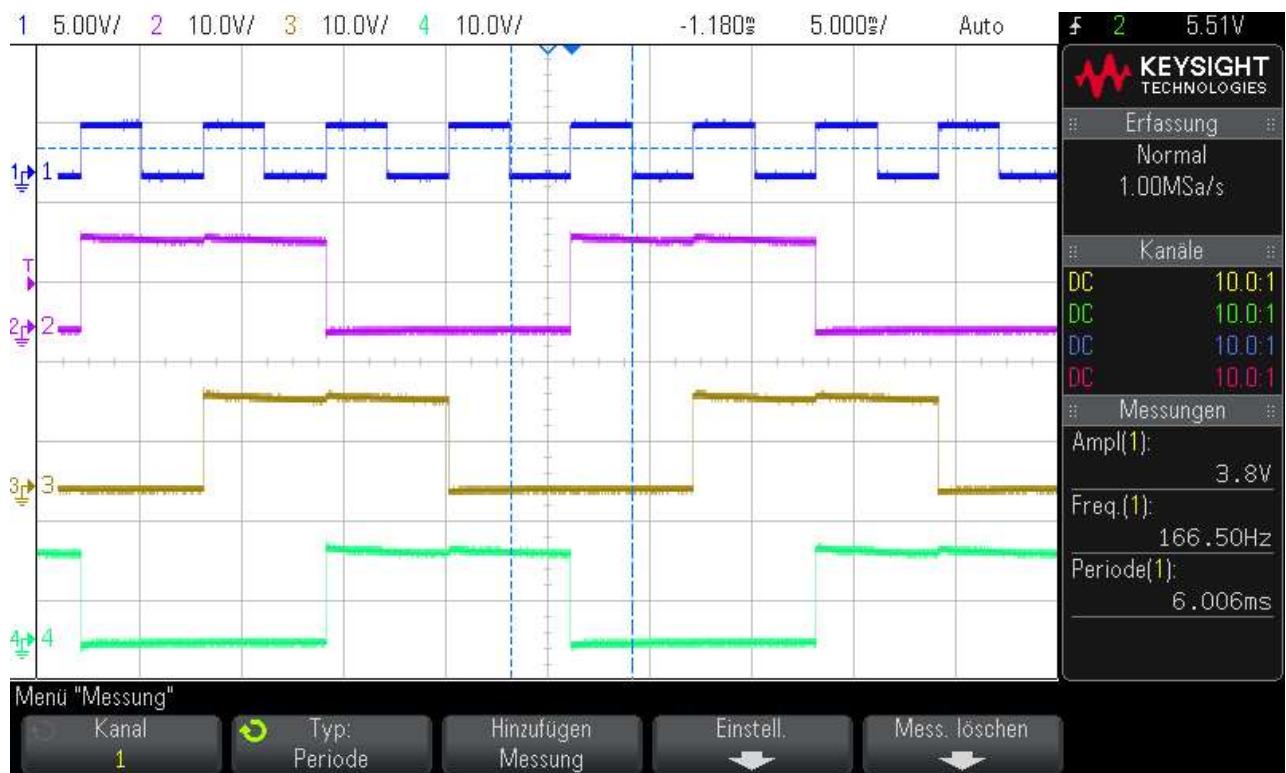


Abbildung 8: Zeitliniendiagramm des Step-Signals

3.2 Mikrocontroller

3.2.1 Wahl des Mikrocontrollers

Für die Ansteuerung des Motors wird ein Mikrocontroller mit speziellen Kriterien benötigt:

- Um den Motor über ein Smartphone zu steuern, muss dieser über Bluetooth oder WiFi angesteuert werden können.
- Das Modul muss bis zu 3,3V liefern können, um eine Fotodiode, Infrarot-LED und einen Hallsensor anzusteuern und auswerten zu können.
- Zur Auswertung der Sensordaten wird zusätzlich auch noch ein ADC-Anschluss (Analog zu Digital Konverter) benötigt.
- Für die Verbindung der Sensoren, dem Schrittmotor und der Spannungsversorgung dieser, werden einige GPIO-Pins am Mikrocontroller benötigt.

Das ESP8266 NodeMCU ist ein Arduino-Mikrocontroller mit eingebautem WiFi-Modul. Dieser Mikrocontroller ist uns bereits aus Schulprojekten an der HTBLuVA Salzburg der vorherigen Jahre bekannt und die Umgebung daher vertraut. Dieser Mikrocontroller liefert die gewünschte Versorgungsspannung von 3,3V an den Ausgängen, verfügt über ein ADC-Modul und hat genügend GPIO-Anschlüsse, um den Motortreiber-IC, die Fotodiode, IR-LED und den Hallsensor ansteuern zu können. Zusätzlich ist der Controller mit 6,50€ (Stand 03.12.2018, laut Amazon) sehr kostengünstig und perfekt zur Umsetzung des Projekts geeignet. In Abbildung 9 ist das Pinout des ESPs veranschaulicht. Benutzte Pins wurden hierbei mit einem Punkt versehen.

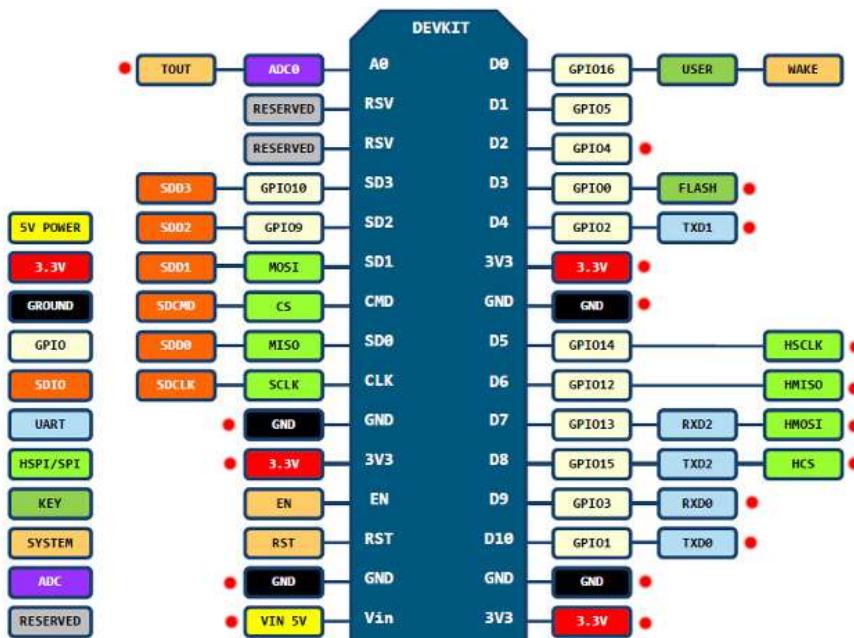


Abbildung 9: Pinout des ESP8266

3.2.2 Takterzeugung mittels PIC

Da das ESP-Modul seinen Timer bereits für die WLAN-Verbindung zum Smartphone benutzt, wird ein weiteres Taktsignal benötigt, welches den Motortreiber und damit auch den Motor steuert. Dieses Taktsignal muss eine möglichst gleichmäßige Periodendauer haben, da der Schrittmotor bei unregelmäßigem Ansteuern ins Stocken kommen kann. Um ein passendes Signal zu erhalten, wird der PIC12F1572-Microchip verwendet. Dieser bietet drei pulsweitenmodulierbare und voneinander unabhängige Taktgeber. Von diesen wird einer zur Taktgenerierung für den Motor verwendet.

3.2.2.1 Ansteuerung des PIC

Die Beschriftung des PIC als Taktgenerator entspricht dem Datenblatt des „PIC12F1572“ von Microchip.

Von den acht Pins des PIC werden lediglich V_{DD}, V_{SS}, R_{A4} und R_{A1} benötigt, wobei die Spannung von 3,3V für V_{DD} direkt vom ESP-Modul geliefert wird und V_{SS} ebenfalls mit dem ESP auf Ground verbunden wird. R_{A1} wird als Eingang benutzt. An diesen PIN wird die Anzahl der Schritte in einem String (mit ASCII-Zeichen) gesendet, um die sich der Motor drehen soll, also beispielsweise 200 Schritte für eine volle Umdrehung. R_{A4} dient als Ausgangs-PIN. Hier wird das Taktsignal an den Motortreiber übergeben, welcher dann den Motor dreht.

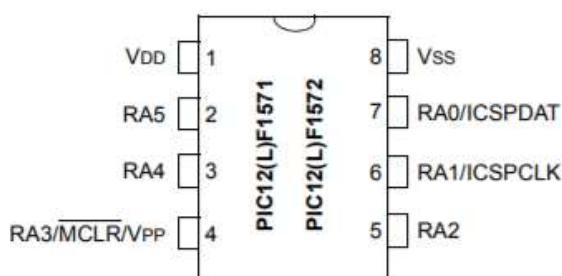


Abbildung 10: PIN Diagramm des PIC³

Programmintern kann nun justiert werden, welche Periodendauer gewünscht ist. Für dieses Projekt wurde eine Periodendauer von 3 Millisekunden gewählt. (Siehe: 3.1.4 Vollschrittbetrieb)

³ Vgl.: <http://ww1.microchip.com/downloads/en/DeviceDoc/40001723D.pdf>; S. 3; [26.11.2018 17:05]

3.2.3 Programmierung

In der nachfolgenden Abbildung sind die einzelnen Programmabschnitte des ESPs mittels Blockschaltbild vereinfacht visualisiert. Einzelne Funktionen sind miteinander verknüpft und benötigen intern noch weitere kleinere Funktionen, daher dient die folgende Abbildung lediglich als Orientierungshilfe.

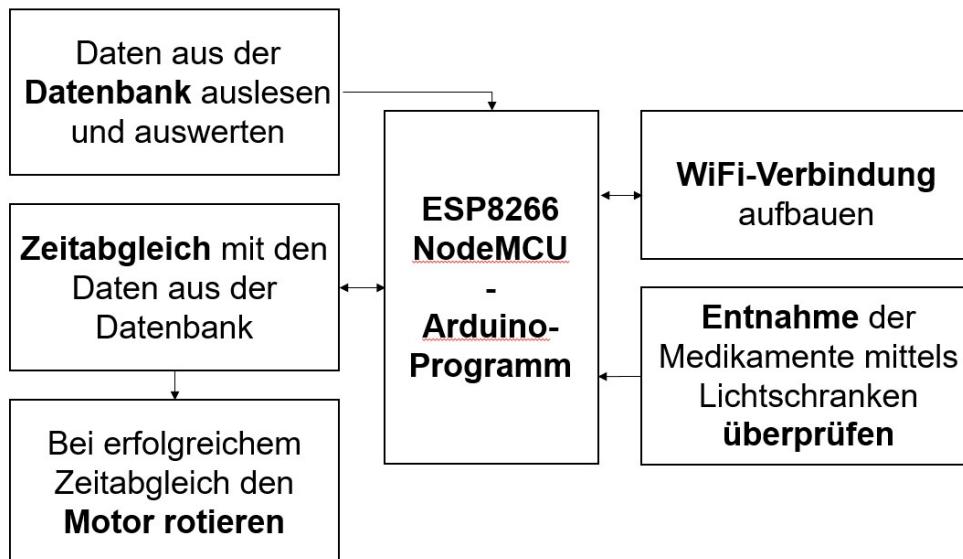


Abbildung 11: Visualisierung der Programmkomponenten am ESP

Das Programm, welches am ESP läuft, wartet darauf, dass vom Smartphone ein Array geschickt wird. In diesem Array befinden sich fünf Zahlenwerte. Diese Werte beschreiben Tag, Monat, Jahr, Stunde und Minute und sollen dem ESP mitteilen, wann sich der Schrittmotor drehen und damit die Medikamente ausgeben soll. Die fünf Werte werden mit dem aktuellen Datum und Uhrzeit über WLAN verglichen. Sobald die Werte mit der aktuellen Zeit übereinstimmen, wird eine Funktion aufgerufen, welche den Schrittmotor aus dem „Schlaf-Modus“ in den Betriebszustand versetzt und dann die Medizinbox um genau ein Fach weiterdreht. Zusätzlich wird eine LED im Sekudentakt ein- und ausgeschaltet um dem Benutzer der Box zu signalisieren, dass Medikamente zur Einnahme bereitstehen. Diese Funktion kann zusätzlich mit einem akustischen Signal stärker signalisiert werden, dies hängt jedoch vom Benutzer ab und davon, wie dringend die Medikamente eingenommen werden müssen, beziehungsweise wie dringend die zeitgenaue Einnahme dieser ist. Nachdem die Medikamente in die Schachtel zur Entnahme gefallen sind, wird auch der Lichtschranken zwischen Fotodiode und IR-LED eingeschaltet, damit die

Entnahme der Schachtel ab sofort überprüft werden kann. Sobald die Schachtel mit ihren Medikamenten entnommen wird, sendet das ESP einen Wert an die Smartphone-Applikation, welche dem Benutzer mitteilt, dass die Medikamente entnommen wurden. Das ESP begibt sich in einen Initialzustand zurück und wartet auf die nächste Ausgabezeit. Anzumerken ist, dass der Lichtschranken nur dann aktiviert ist, wenn erwartet wird, dass die Schachtel entnommen wird. So können Fehlmeldungen vermieden und Strom gespart werden.

Die nachfolgende Abbildung 12 stellt einen Codeabschnitt der Zeitabfrage am Mikrocontroller dar. In Zeile 84 bis 91 wird überprüft, ob alle fünf Werte der aktuellen Zeit („pill_time“) mit der von der App angegebenen Zeit („app_time“) übereinstimmen. Sind alle fünf Werte identisch, ist die Hilfsvariable „help“ gleich fünf und somit wird die Funktion in Zeile 92 bis 96 aufgerufen, welche den Motor um eine gewünschte Anzahl an Schritten in eine gewünschte Richtung dreht. Zudem wird eine kurze Benachrichtigung in der Konsole ausgegeben.

```
--  
84 int help = 0;  
85 if (!fertig) {  
86     for (int i = 0; i <= 5; i++) {  
87         if (pill_time[i] == app_time[i]) {  
88             help++;  
89         }  
90     }  
91 }  
92 if (help == 5) {  
93     turn(200, 0); //steps; direction  
94     fertig = 1;  
95     Serial.printf("\nZEIT: %d.%d.%d - %d:%d\n"), day, month, year, hour, min;  
96 }
```

Abbildung 12: Zeitvergleich am ESP

3.3 Motoransteuerung

3.3.1 Wahl des Motortreibers

Um ein, wie in dieser Arbeit unter 3.1.4 Vollschriftbetrieb beschriebenes, Signal zu bekommen, werden zwei H-Brücken für die vier Ausgangssignale benötigt. Durch die H-Brücken können Ströme, welche diese durchfließen, umgepolt werden und somit die passenden Signale für den Motor zur Verfügung stellen. Ebenfalls sollte der IC die gewünschte Geschwindigkeit und Drehrichtung steuern können. Der „DRV8825 Stepper Motor Controller IC“ von Texas Instruments erfüllt all diese Kriterien und wird daher für diese Arbeit verwendet.

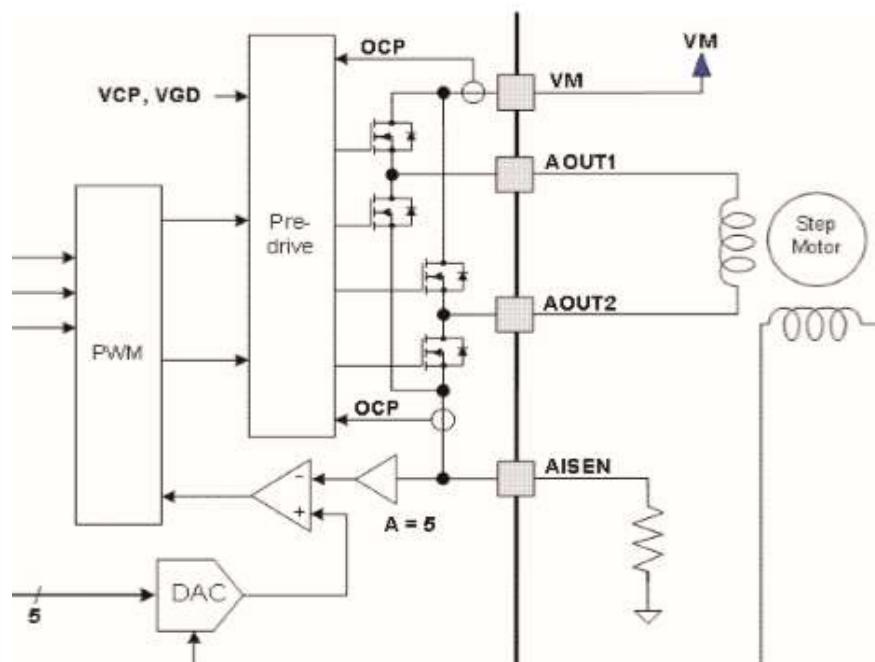


Abbildung 13: Mottransteuerung über H-Brücken nach Datenblattauszug (Quelle: ti.com)

Abbildung 13 zeigt die interne Beschaltung des Motortreiber-ICs. Die beiden Pins AOUT1 und AOUT2 sind dabei an die einzelnen H-Brücken angeschlossen.

3.3.2 Hardware zur Motorsteuerung

Der DRV8825 benötigt zum Betreiben des Schrittmotors zusätzliche externe Bauteile, welche in der folgenden Abbildung veranschaulicht sind.

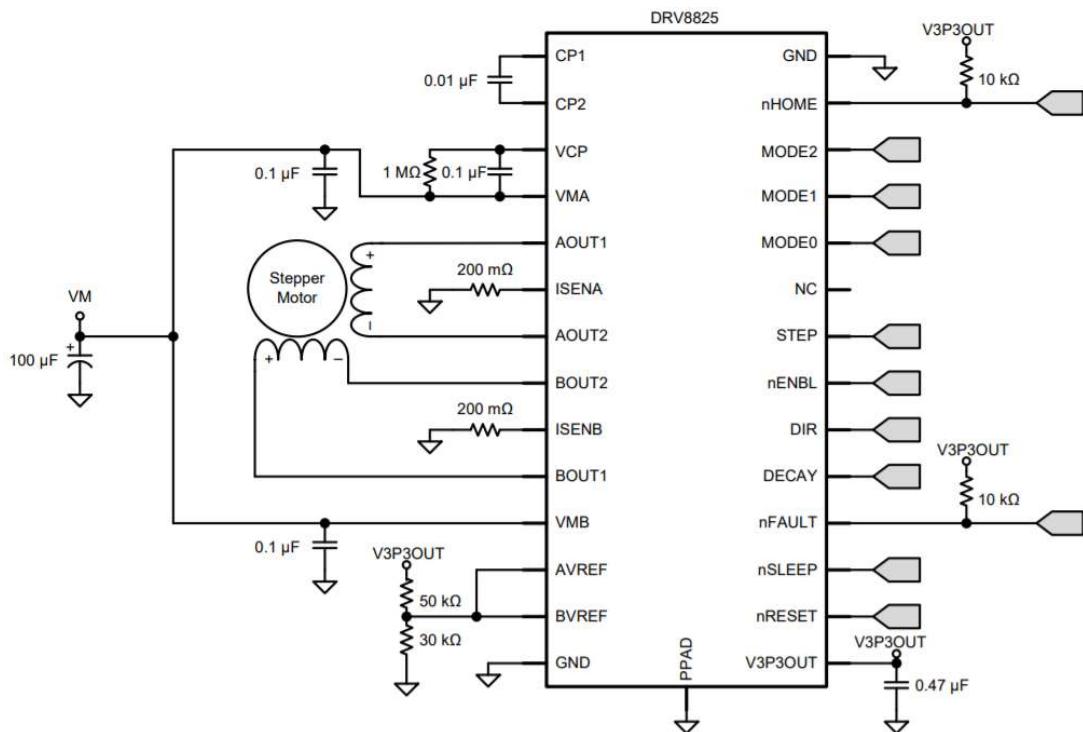


Abbildung 14: Beschaltung des Motortreiber-IC nach Datenblattauszug⁴

Diese Beschaltung ist dem Datenblatt von Texas Instruments für Bipolar-Schrittmotoren entnommen. Einzelne Bauteilwerte müssen für den gewählten Schrittmotor der SMB jedoch abgeändert werden. Um kurzzeitig die interne IC-Gate-Spannung, welche höher als die an VCP/VMA angelegten 12V ist, an den MOSFETs zu erreichen, wird zwischen CP1 und CP2 ein Kondensator von 0,01uF angeschlossen. (Siehe Abbildung 14; links oben)

Für den Betrieb des Schrittmotors muss der Strom, der ihn durchfließt, begrenzt werden. Ansonsten kann der Motor oder dessen Treiber beschädigt werden. Es wurde ein Maximalstrom von 0,5A gewählt, damit noch ein ausreichendes Haltemoment gegeben ist, jedoch ohne eine übermäßige Wärmeentwicklung.

⁴ Vgl.: <http://www.ti.com/lit/ds/symlink/drv8825.pdf>; S.18f; [12.11.2018 12:35]

Um den Maximalstrom durch den Motor auf 0,5A zu begrenzen, liefert das Datenblatt folgende Formel:

$$I_{CHOP} = \frac{V_{REF}}{5 * R_{ISENSE}}$$

R_{ISENSE} wird mit 0,1 Ohm angenommen. Dieser befindet sich jeweils an den Pins ISENA und ISENB. Somit ergibt sich folgende Umformung der vorherigen Formel auf V_{REF} .

$$\frac{V_{REF}}{5 * 0,1\Omega} = 0,5A \Rightarrow V_{REF} = 0,5A * 5 * 0,1\Omega = 0,25V$$

Dieser Strom liegt nun zwischen AVREF und BVREF an und somit ergibt sich folgender Spannungsteiler für die Widerstände R6 und R7 bei einer Spannung von 3,3V und der Annahme von $R7 = 10k\Omega$ (E12):

$$\frac{R6 + R7}{R7} = \frac{3,3V}{0,25V} = \frac{R6 + 10k\Omega}{10k\Omega} \Rightarrow \frac{3,3V}{0,25V} * 10k\Omega - 10k\Omega = R6 = 122k\Omega$$

Um der E12-Widerstandsreihe - eine genormte Folge von Werten elektronischer Bauelemente - zu entsprechen, wurden für R6 daher 120k Ω gewählt.

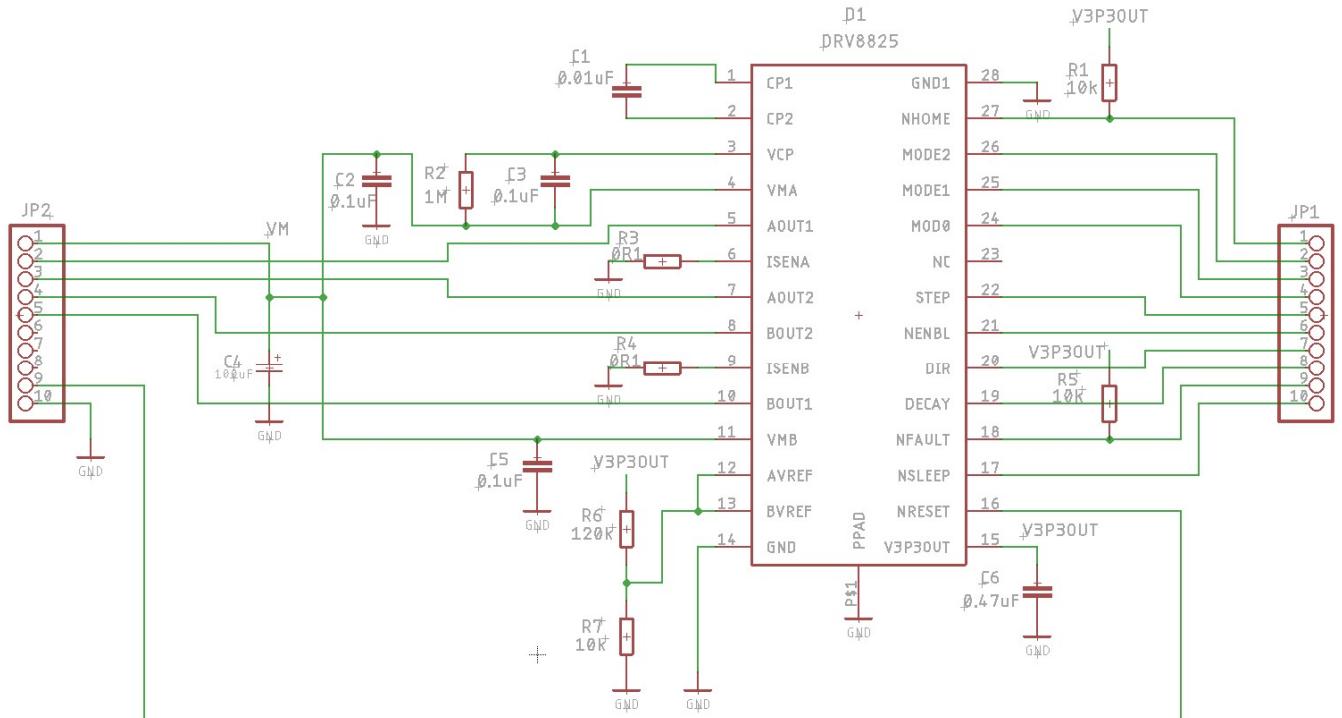


Abbildung 15: Individuelle Beschaltung des Motortreiber-IC

Über die Steckleisten JP1 und JP2 kann der Motortreiber vorerst auf einem Steckbrett getestet und später mit einer Basisplatine, auf welcher dann auch der Mikrokontroller zur Steuerung zu finden ist, verbunden werden. Beim Betrieb der Platine auf dem Steckbrett ist unbedingt darauf zu achten, dass Motortreiber, Mikrocontroller und Spannungsversorgung dasselbe Potential haben; also alle GND-Leitungen miteinander verbunden sind.

Die folgende Abbildung zeigt die fertige Motortreiber-Platine, dessen Aufbau im Anhang unter „12.1.1 Motortreiberplatine“ zu finden ist.

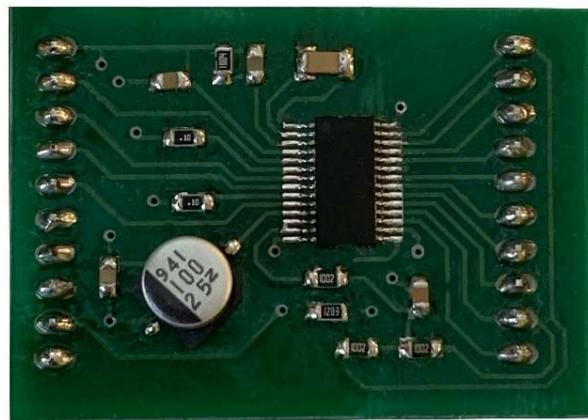


Abbildung 16: Gefertigte Motortreiber-Platine

3.3.3 Software zur Motoransteuerung

Vor Beginn der Programmierung werden dem Datenblatt noch Werte für Variablen entnommen, welche zur Motoransteuerung benötigt werden.

3.3.3.1 Schrittmotorbetriebsart – Mode0/1/2

Über drei Variablen kann bestimmt werden, ob der Schrittmotor über Voll-, Halb-, Viertel-, Achtel-, 16tel- oder 32stel-Schritte betrieben wird. Mode0, Mode1 und Mode2 werden dabei bei dem gewählten Schrittmotor dazu verwendet. Die einzelnen Betriebsarten sind in der folgenden Tabelle 3 veranschaulicht.

MODE2	MODE1	MODE0	STEP MODE
0	0	0	Full step (2-phase excitation) with 71% current
0	0	1	1/2 step (1-2 phase excitation)
0	1	0	1/4 step (W1-2 phase excitation)
0	1	1	8 microsteps/step
1	0	0	16 microsteps/step
1	0	1	32 microsteps/step
1	1	0	32 microsteps/step
1	1	1	32 microsteps/step

Tabelle 3: Schrittweite des Motors nach Datenblattauszug⁵

Wie aus der Tabelle zu entnehmen ist, werden Mode0, Mode1 als auch Mode2 für den Vollschrittbetrieb auf „0“ gesetzt.

3.3.3.2 Enable, Sleep und Indexer

Über den Sleep-Pin des Motortreibers kann dieser in einen Ruhezustand gebracht werden. Die Problematik hierbei ist, dass sich somit auch alle internen Taktgeneratoren deaktivieren und erst wieder weiterlaufen, sobald der Sleep-Pin wieder zurückgesetzt wird. Ebenfalls wird ein Indexer wieder auf null zurückgesetzt. In diesem wird festgehalten, in welchem der vier Ansteuerungsmöglichkeiten der Motor sich momentan befindet (siehe „Tabelle 2: Phasen des Schrittmotors“).

Da über den Indexer die Position des Motors bestimmbar ist, darf dieser nicht einfach zurückgesetzt werden, sobald die Box ausgeschaltet wird. Der Enable-Pin lässt interne Taktgeneratoren weiterlaufen und setzt auch den das interne Register, in dem sich der Indexer befindet, nicht zurück.

Aus genannten Gründen wird der Sleep-Pin dauerhaft auf null gesetzt und der Enable-Pin immer dann unter Spannung gesetzt, wenn auch der Motor gedreht wird. Solange die Funktion, welche den Motor dreht, nicht aufgerufen wird, fließt somit auch kein Strom. Da der Indexer dazu verwendet wird, die Position des Schrittmotors zu bestimmen, wird nicht zwingend ein Hallsensor, wie in „3.4.2 Hallsensor“ beschrieben, zur Positionskontrolle des Motors benötigt. Dank dem Indexer kann programmintern geregelt werden, dass der Schrittmotor darauf ausgelegt ist, immer nur 25 Schritte zu fahren (Siehe 3.1.4 Vollschrittbetrieb).

⁵ Vgl.: <http://www.ti.com/lit/ds/symlink/drv8825.pdf>; S.13f; [19.11.2018 17:30]

Sollte der Motor - aus nicht gewünschten Gründen - also beispielsweise 23, statt 25 Schritten fahren, erkennt die Indexer-Funktion dies automatisch und dreht auf die „vollen“ 25 Schritte weiter. Somit ist garantiert, dass das Ausgabefach der Medizibox am richtigen Platz steht.

3.3.4 Zusammenfassung der Variabel-Einstellungen

Die Belegungen der Variablen sind in der folgenden Tabelle kurz zusammengefasst:

Variable	Wert	Erklärung
Mode0	0	Bestimmen die Betriebsart des Schrittmotors
Mode1	0	
Mode2	0	
Enable	LOW/HIGH	Versorgt die Motortreiber-Platine immer dann mit Spannung, wenn der Motor gedreht wird. Ansonsten wird Enable auf LOW gesetzt.
Sleep	LOW	Bei HIGH werden alle internen Abläufe der Motortreiber-Platine deaktiviert. Bei LOW werden alle Abläufe wieder reaktiviert beziehungsweise bleiben diese dauerhaft aktiv, solange eine Spannungsversorgung vorhanden ist.

Tabelle 4: Variabel-Einstellungen

3.4 Sensorik

3.4.1 Fotodiode & IR-LED

3.4.1.1 Verwendung

Wird der Inhalt eines Fachs der Medizibox ausgegeben, so fällt dieser in einen Ausgabebehälter, welcher vom Pflegebedürftigen entnommen werden kann. Sobald dieser Behälter aus seiner Position entnommen wird, scheint IR-Licht auf die Fotodiode, wodurch ESP-intern eine Funktion aufgerufen wird (Siehe 3.4.1.2). Zusätzlich wird gespeichert, dass das Fach des jeweiligen Tages korrekterweise entnommen wurde.

Abbildung 17 und 18 veranschaulichen, wie der Lichtschranken zur Überprüfung der Entnahme der Medizin fungieren soll.



Abbildung 17: IR-LED - Behälter - Fotodiode

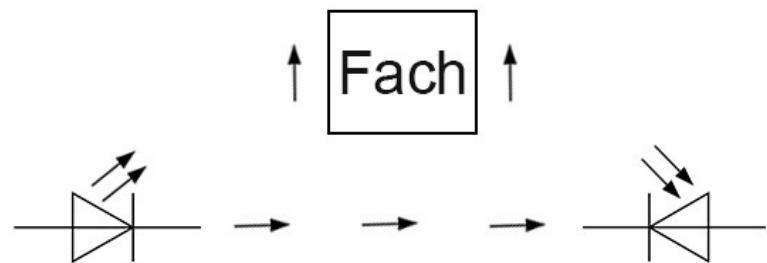


Abbildung 18: IR-LED - kein Behälter - Fotodiode

3.4.1.2 Funktionsweise

„Infrarotstrahlung (IR-Strahlung), die umgangssprachlich auch als Wärmestrahlung bezeichnet wird, ist eine elektromagnetische Strahlung im Wellenbereich von 780 nm bis 1 mm und befindet sich somit im elektromagnetischen Spektrum zwischen dem sichtbaren Licht und den Mikrowellen. Natürliche Quellen sind z.B. das Sonnenlicht und Feuer.“⁶ Für dieses Projekt wird eine Infrarot-LED des Herstellers VISHAY verwendet, da diese eine Wellenlänge von 950nm erreicht. Zusätzlich wird eine IR-Fotodiode verwendet, welche IR-Strahlung in eine elektrische Spannung umwandelt. Diese Spannung kann nun vom Mikrocontroller über ein internes ADC-Modul als digitaler Wert interpretiert werden. (Siehe 3.4.1.4 Programmierung)

⁶ Vgl.: https://www.awmf.org/uploads/tx_szleitlinien/002-010I_S1_Infrarotstrahlung_W%C3%A4rmestrahlung_2012.pdf; [19.12.2018 10:38]

Die Infrarot-LED wird mit 1,3V und maximal 150mA betrieben (Siehe Abbildung 19). Da das ESP 3,3V liefert und ein Strom von 100mA gewählt wurde, müssen also 2V am Vorwiderstand abfallen.

$$R_V = \frac{V_{CC} - U_F}{100mA} = \frac{2V}{100mA} = 20\Omega$$

Der Vorwiderstand der Infrarot-LED wurde mit $R_V = 20\Omega$ (E12) gewählt.

BASIC CHARACTERISTICS ($T_{amb} = 25^\circ C$, unless otherwise specified)						
PARAMETER	TEST CONDITION	SYMBOL	MIN.	TYP.	MAX.	UNIT
Forward voltage	$I_F = 100\text{ mA}, t_p = 20\text{ ms}$	V_F		1.3	1.7	V

Abbildung 19: Betriebsspannung Infrarot-LED⁷

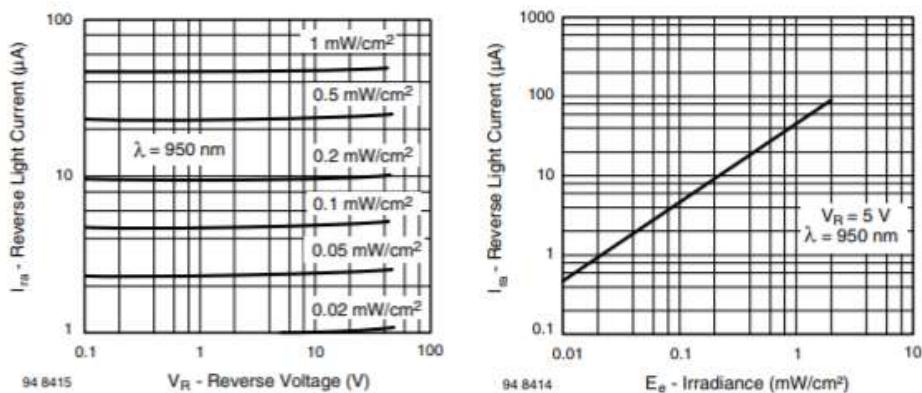


Abbildung 20: Strom und Spannungsverlauf Fotodiode⁸

Da laut Datenblatt mit einer IR-LED zwischen 0,2-1mW/cm² bei voller Helligkeit zu rechnen ist, ergibt sich ein Strom von ca. 10-40uA. Bei voller Helligkeit soll die Foto-Diode zwischen 2-3V für den ADC-Pin am ESP liefern, damit dieser die anliegende Spannung als digitalen Wert interpretieren kann. Die Spannung wird in der nachfolgenden Rechnung mit 2,5V angenommen.

$$R_V = \frac{U_e}{I_{max}} = \frac{2,5V}{40uA} = 62,5k\Omega$$

Der Vorwiderstand der Fotodiode wurde mit $R_V = 68k\Omega$ (E12) gewählt.

⁷ Vgl.: <https://www.vishay.com/docs/81055/tsus520.pdf>; S.2; [27.01.2019 16:00]

⁸ Vgl.: <https://www.vishay.com/docs/81522/bpw41n.pdf>; S. 2; [27.01.2019 16:30]

3.4.1.3 Beschaltung

Die IR-LED und IR-Fotodiode werden demnach wie folgt verkabelt:

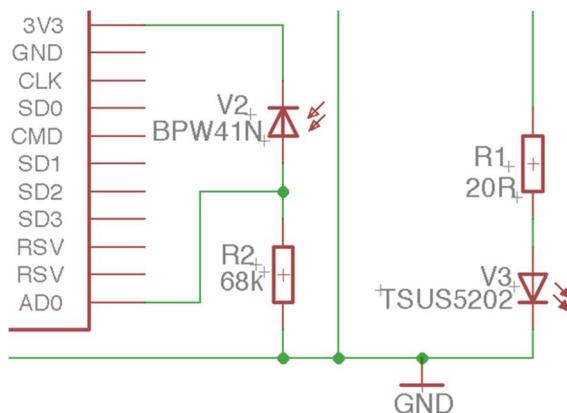


Abbildung 21:Beschaltung IR-LED & Fotodiode

Die Fotodiode und IR-LED werden im Endprodukt mit einem Abstand von circa drei bis fünf Zentimetern zueinander gerichtet. Ist die Luftlinie dazwischen nun frei, liefert die Fotodiode eine Spannung an den Mikrocontroller. Wird die Luftlinie jedoch unterbrochen, fließt (nahezu) kein Strom. Wie dieser Ablauf genau realisiert wird, ist im Absatz „3.4.1.1 Verwendung“ genauer beschrieben.

3.4.1.4 Programmierung

Über die Funktion „checkIR“ und dem Analog Digital Converter am ESP wird der von der Fotodiode ausgegebene Spannungswert ausgewertet. Der ADC-Pin des ESPs hat eine Auflösung von 10bit und eine Messbereich von 0 bis 3,3V. Eine maximale Spannung von 3,3V würde daher einen Wert von 1024 ergeben. Bei den Lichtverhältnissen innerhalb der Box ergeben sich, sobald die Luftbrücke zwischen Fotodiode und IR-LED vorhanden ist, Werte im Bereich von 600-900. Befindet sich ein Objekt innerhalb des Lichtschrankens, so liefert die Funktion null.

Ein Wert zwischen 600-900 (in der folgenden Rechnung als 750 gemittelt) ergibt daher folgende Spannung:

$$\frac{3,3V}{1024} = \frac{x}{750} \Rightarrow 750 * \frac{3,3V}{1024} = 2,42V$$

Im Ruhemodus befindet sich eine Medizinschachtel zwischen IR-LED und Fotodiode. Daher ist keine Verbindung vorhanden. Sobald die Schachtel entnommen wird, erkennt die Funktion, dass der Wert sich von „hell“ (≈ 750) zu „dunkel“ (≈ 20) geändert hat, die Differenz zwischen altem und neuem Wert also, nach Definition, größer als zehn ist. (Siehe nachfolgende Abbildung, Zeile 26)

```

21 boolean checkIR() {
22     static int old = 0;
23     int pd = analogRead(A0);
24     int dif = pd-old;
25     old = pd;
26     return dif>10;
27 }
```

Abbildung 22: Programmcode - Lichtschranken

Diese Funktion liefert immer dann eine Rückmeldung, sobald die Medizinschachtel entnommen wurde. Diese Daten können dann zurück an das Smartphone geschickt werden und dort den Benutzer über die Geschehnisse informieren.

3.4.2 Hallsensor

3.4.2.1 Funktionsweise

Ein Hallsensor wird verwendet, um die genaue Position eines Fachs der Medizinbox zu bestimmen. Würde der Motor immer eine bestimmte Anzahl an Schritten fahren, besteht die Möglichkeit, dass die Position der Fächer nicht mehr genau genug ist und sich um mehrere Millimeter verschiebt. Um diese Ungenauigkeit zu vermeiden, wird an jedem Fach ein kleiner Magnet auf der Außenseite befestigt. Wenn der Schrittmotor nun von einem Fach zum nächsten wechseln soll, dreht er nur so lange, bis er den nächsten Magneten – und damit auch das nächste Fach – erkennt und stoppt dann.

Damit der Hallsensor nur dann Spannung liefert, wenn auch ein Magnetfeld von der Medizintrommel erkannt wird, wird ein Non-Latching Hall-Effect-Sensor verwendet. (Bei einem Latching Hall-Sensor könnte solange Spannung fließen, bis erneut ein Magnetfeld erkannt wird.) Dieser ist auch noch bipolar, somit spielt es keine Rolle, ob der Süd- oder Nordpol des Magneten näher am Hallsensor ist, da dieser über beide Pole steuerbar ist. Abbildung 23 veranschaulicht die drei allgemeinen Betriebsarten bipolarer Schalter und deren Schaltschwellen. Die magnetische Flussdichte „B“ beschreibt dabei die Eigenschaft eines Magnetfelds, welches zur Bestimmung der Schaltpunkte von Hallsensoren verwendet wird. Beschrieben wird dieses in Gauß „G“, wobei $1\text{G} = 0,1\text{mT}$ entspricht. Bipolare Schalter haben normalerweise einen positiven magnetischen Operationspunkt „BOP“ und einen negativen Freigabepunkt „BRP“.⁹

- Im Latch-Modus schaltet der Hallsensor mit positiven BOP und negativem BRP. Das bedeutet, dass mit dem Südpol des Magneten ein- und mit dem Nordpol ausgeschaltet wird.
- Im unipolaren Modus wird sowohl bei BOP als auch BRP im positiven Bereich geschaltet. Somit schaltet der Hallsensor nur dann, wenn ein Südpol anliegt.
- Der negative unipolare Modus schaltet sowohl bei BOP als auch BRP im negativen Bereich, also dem Nordpol.

Kombiniert man nun den unipolaren und negativ unipolaren Modus, so wird im positiven als auch negativem Bereich der Flussdichte geschalten – also bipolar. Dieser Modus wird für den Hallsensor in diesem Projekt verwendet.

⁹ Vgl.: <https://www.allegromicro.com/en/Design-Center/Technical-Documents/Hall-Effect-Sensor-IC-Publications/Bipolar-Switch-Hall-Effect-ICs.aspx>; [21.03.2019 12:05]

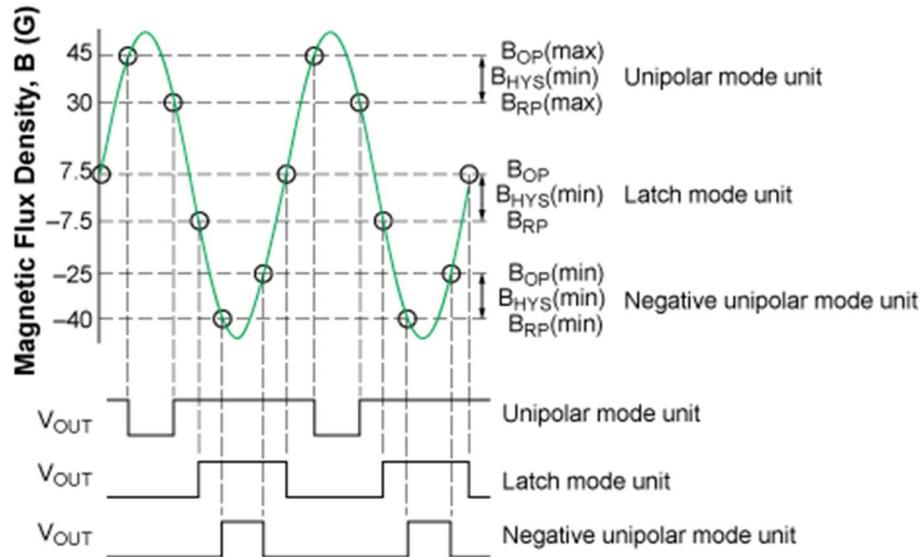


Abbildung 23: Spannungsverlauf und Schaltschwellen beim Hallsensor (Quelle: allegro.com)

3.4.2.2 Beschaltung

Da der gewählte Hallsensor ein SMD Bauteil ist, wird eine kleine Platine angefertigt, welche an der Box befestigt werden kann. Die Bauteilwerte von $C1 = 47\text{nF}$ und $R1 = 10\text{k}\Omega$ ergeben sich nach dem Datenblatt „Diodes Incorporated“. Dieses empfiehlt Werte von $10\text{nF}-100\text{nF}$ für den Kondensator und $10\text{k}\Omega$ bis $100\text{k}\Omega$ für den Widerstand.

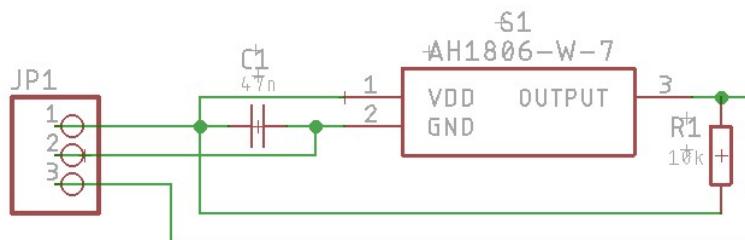


Abbildung 24: Hallsensor Platine

Solange sich kein Magnet nah genug am Hallsensor befindet, wird eine Spannung ausgegeben. Sobald ein Magnetfeld erkannt wird, geht die Spannung gegen Null. Die Spannungswerte werden somit entweder als „0“ für keine Spannung beziehungsweise „1“ für Spannung an das ESP gesendet, über welches dann programmintern ausgewertet werden kann, um wie viele Fächer sich die Medizintrommel weitergedreht hat.

Entgegen der ursprünglichen Planung wurde statt dem Hallsensor jedoch ein Indexer verwendet, welcher programtechnisch bestimmen kann, an welcher Position der Schrittmotor sich befindet. Dieser ist im Absatz „3.3.3.2 Sleep, Enable und Indexer“ jedoch

genauer erläutert. Die Verwendung des Hallsensors empfiehlt sich für Anwendungen, wo Störungen des Motors häufiger eintreten können, als bei der Verwendung in einem geschlossenen Gehäuse, wie es bei der SMB der Fall ist.

3.5 Gehäusebau

3.5.1 Hauptgehäuse

Das Hauptgehäuse wird dazu verwendet die Basisplatine mit ihren Aufsteckplatinen aufzubewahren. Zusätzlich wird hier auch der Schrittmotor mit der Medizin-Trommel montiert, wobei auf der Vorderseite des Gehäuses eine Aussparung frei bleibt, durch welche die Medikamente aus der Box in eine Schachtel fallen können.

Gefertigt wird das Gehäuse aus Polystyrol, welches von einer CNC-Fräse zurechtgeschnitten wird. Die Box wird in der Online-Software „Inventables-Easel“ erstellt (Siehe Abbildung 25). Die einzelnen Segmente des Gehäuses werden hier modelliert und können in einer 2D-Ansicht hier betrachtet werden.

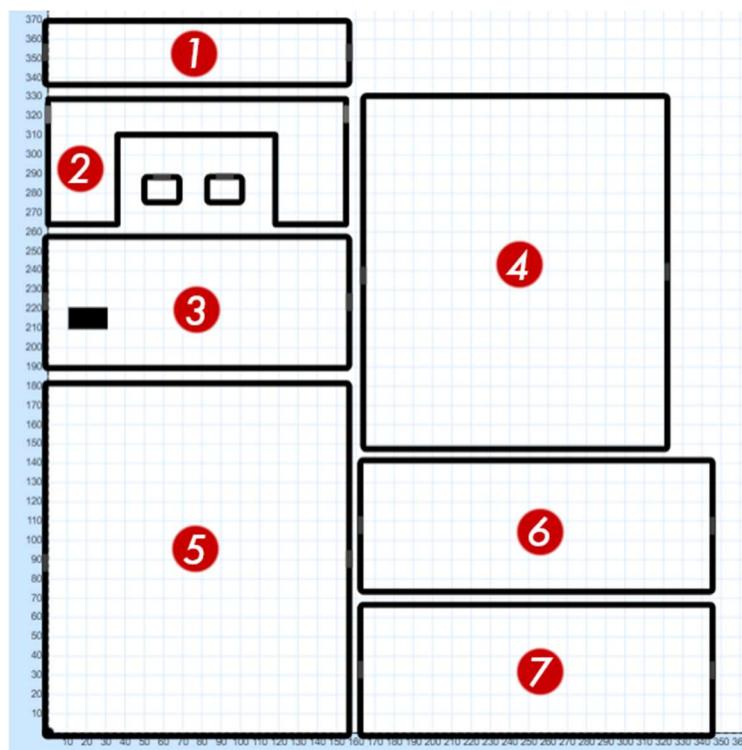


Abbildung 25: Gehäuseaufbau in Easel

- Der mit „1“ markierte Bauteil aus der vorherigen Abbildung fungiert als Stütze innerhalb des Gehäuses, um für Stabilität zu sorgen und gleichzeitig um eine Platine daran befestigen zu können.
- Bauteil „2“ ist die Vorderseite der Box. Hier werden die Medikamente ausgegeben.
- Bauteil „3“, „6“ und „7“ sind die Seitenteile der Box, wobei auf der Rückseite eine Aussparung ausgefräst wird, durch welche die Kabel in die Box eingeführt werden können.
- Übrig bleiben somit noch Bauteil „5“ und „6“, welche Bodenplatte und Deckel der Box werden.

Dieses 2D-Modell der Box kann nun gefräst werden und muss anschließend noch zurechtgeschliffen und zusammengebaut werden.

Das fertige Gehäuse sieht wie in Abbildung 26 veranschaulicht aus.

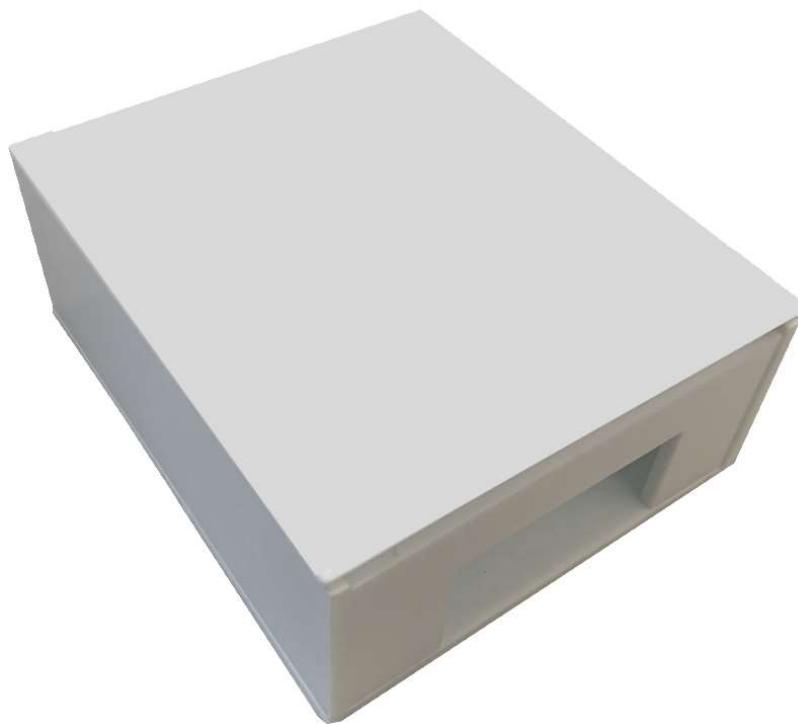


Abbildung 26: Assemblieretes Gehäuse

3.5.2 Medizin-Trommel

Da die Medizinbox selbst kreisförmig und der innere Aufbau der Box etwas komplizierter ist, wird diese mit dem hauseigenen 3D-Drucker der HTBLuVA Salzburg hergestellt. Die Trommel wird in dem Programm Solid Edge gefertigt und sieht wie folgt aus:

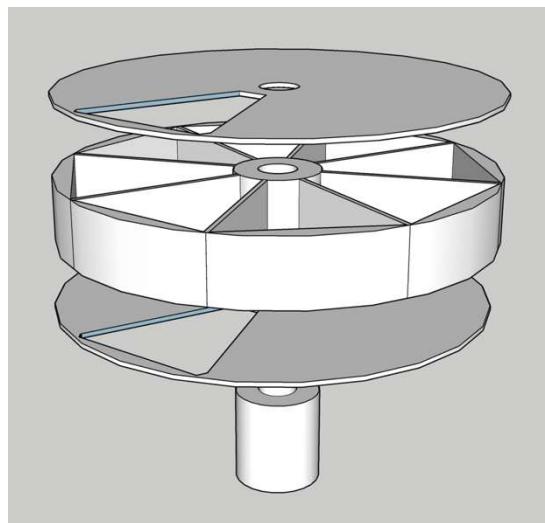


Abbildung 27: 3D-Design der Trommel

3.5.2.1 Solid Edge

Zur Realisierung dieses Projekts wurde das CAD Programm Solid Edge gewählt, da dies bereits in der HTBLuVA Salzburg im Unterricht in den Schuljahren davor verwendet wurde. Da das Programm weitgehend bereits vertraut ist, ist auch das erneute Einarbeiten in die Software schneller möglich. Solid Edge ist eine sehr bekannte und beliebte CAD Software. Ein CAD-System ist im Grunde nichts anderes als eine Kombination aus Software und Hardware – sozusagen wie eine Art Computer oder Workstation – die Ingenieure bei der Entwicklung von Produkten und Systemen unterstützt. Eingesetzt werden kann CAD dabei in der Entwicklung, Bearbeitung, Analyse oder Optimierung eines Designs. Mit CAD können Kurven und Formen in zwei Dimensionen (2D), sowie Kurven, Oberflächen und Gegenstände in drei Dimensionen (3D) entworfen werden. Letzteres wird auch als 3D-CAD bezeichnet.^{“10}

Im Fall dieses Projektes wird der 3D-Drucker der HTBLuVA verwendet um die Hardware zu erzeugen. Mit CAD kann in zwei Dimensionen aber auch in drei Dimensionen entworfen

¹⁰ Vgl.: <https://de.rs-online.com/web/generalDisplay.html?id=infozone&file=automation/cad>; S.1 [16.03.2019]
Tockner Gregor

und modelliert werden. Für dieses Projekt wird ein Dreidimensionaler Gegenstand entworfen, welcher auch als 3D-CAD bezeichnet wird.

3.6 Spannungsversorgung

3.6.1 Erzeugung der Betriebsspannungen

Die Schaltung wird mit einer 12 Volt Eingangsspannung versorgt. Die 12 Volt Spannung wird benötigt, um den Schrittmotor zu versorgen. Weiter wird für die Schaltung ein 3,3 Volt Spannungspegel benötigt. Diese 3,3 Volt versorgen den Motortreiber, den PIC für den Motortreiber und die Sensorik.

Um die 12 Volt auf 3,3 Volt herunter zu regeln, wurde eine Kombination aus Buck-Converter und Low Drop Out Regler gewählt. Der Buck Converter bringt die Spannung möglichst effizient von 12 auf 5 Volt. Der Buck liefert allerdings kein konstantes 5 Volt Signal. Da dies Probleme am Motortreiber verursachen kann, wird ein LDO verwendet, um aus den 5 Volt ein möglichst stör- und rippelfreies 3,3 Volt Signal zu erhalten.

Der LDO, der in diesem Projekt verwendet wird, ist der AMS1117. Der AMS1117 ist bereits auf dem ESP8266 verbaut und eignet sich dadurch sehr gut für eine möglichst unkomplizierte Schaltung. Abbildung 28 zeigt die Gesamtschaltung der Spannungsversorgung, der Eingang des LDO ist der Pin VIN auf dem ESP12E.

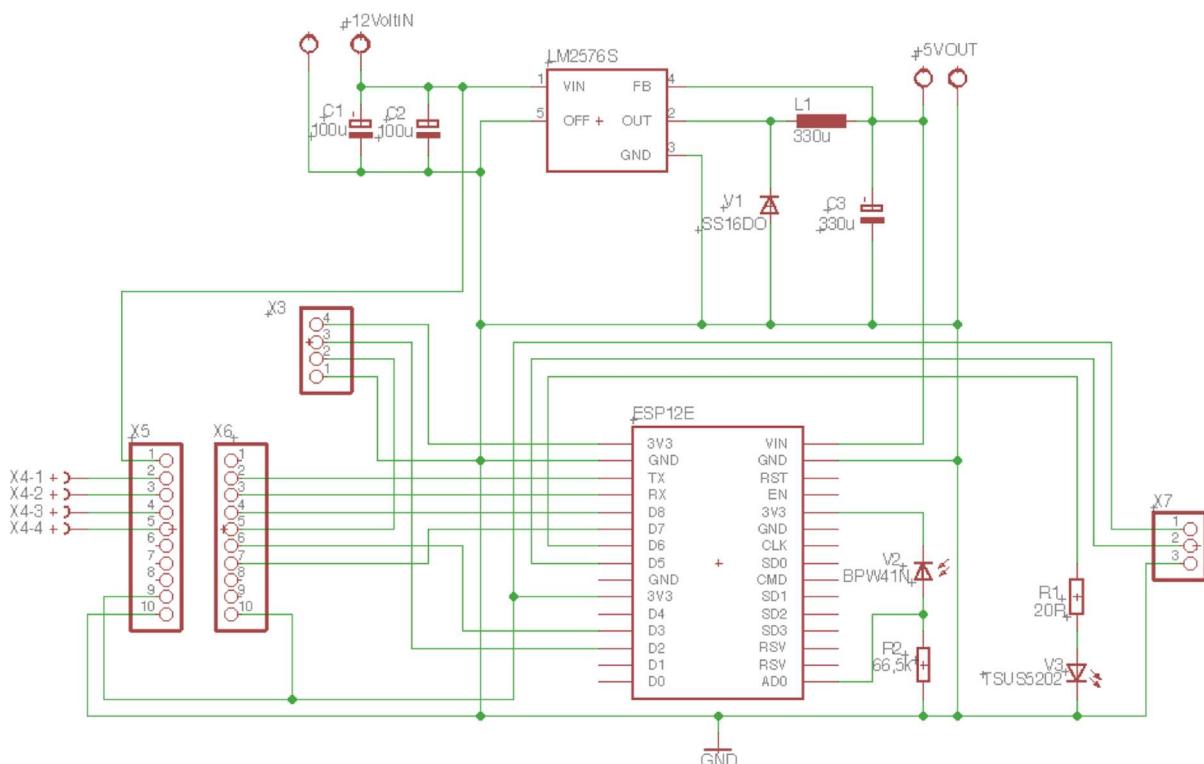


Abbildung 28: Schaltung des Projekts inklusive Spannungsversorgungseinheit

3.6.2 Buck-Converter

3.6.2.1 Funktionsweise

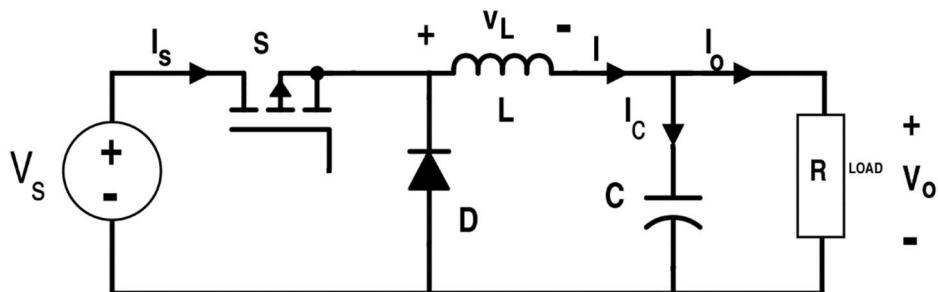


Abbildung 29: Aufbau des Buck-Converters (Quelle: meilleureimage.eu)

Der Transistor „S“ wird ein- und ausgeschalten. Dies passiert üblicherweise einige hundert bis mehrere Millionen Mal pro Sekunde. Dadurch wird elektrische Energie von der Spannungsquelle zum Lastwiederstand überliefert.

Die Spule und der Kondensator speichern Energie und ermöglichen die Versorgung der Last, R_{LOAD} in Abbildung 29, während der Schalter geöffnet ist. Die Ausgangsgröße kann durch die Formel: $U_a = U_e \times \frac{t_1}{T}$ eingestellt werden. Diese Steuerung erfolgt üblicherweise mit einem Regler, damit die Ausgangsspannung und der Ausgangsstrom auf dem gewollten Wert erhalten werden können.

Während der Schalter Eingeschalten ist, fließt der Laststrom durch die Spule und durch die Last. Während der Ausschaltphase wird die in der Spule gespeicherte Energie abgebaut. Der Strom durch die Last fließt weiter, allerdings fließt der Strom nun aus dem Kondensator durch die Diode.

Die Spule und der Kondensator bilden einen Tiefpass zweiter Ordnung. Die Abwärtswandlung wird dadurch erreicht, dass aus der Rechteckspannung der Gleichanteil herausgefiltert wird. Wie hoch dieser übrigbleibende Gleichanteil ist, kann durch das Tastverhältnis eingestellt werden.

3.6.2.2 Beschaltung

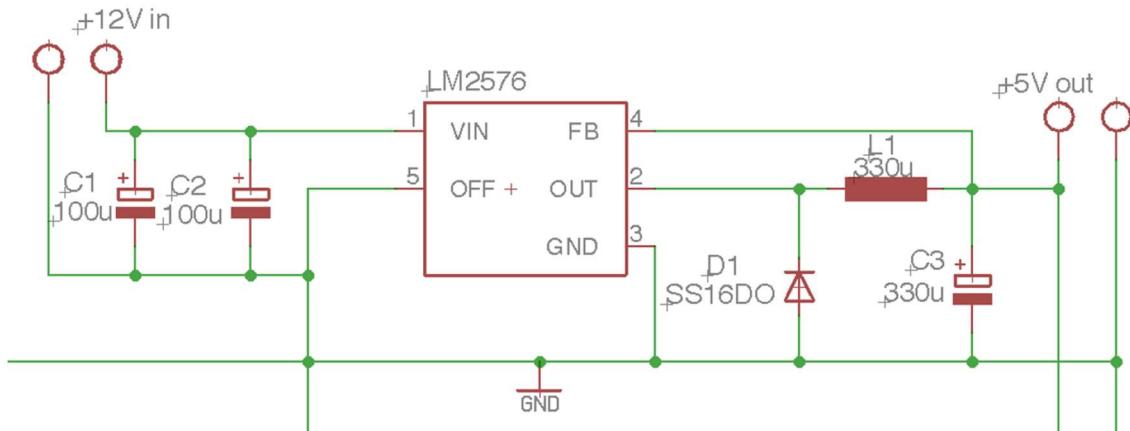


Abbildung 30: Buck Converter

Die 12V Eingangsspannung wird an +12V in angeschlossen. „+5V out“ wird verwendet um zu überprüfen, ob der Buck-Converter die 12V erfolgreich zu 5V umgewandelt hat. Zur Beschaltung wurden die für diesen Anwendungsfall benötigten Werte aus dem Datenblatt entnommen. Abbildung 31 zeigt einen Auszug aus dem Datenblatt, in dem die Bauteilwerte für eine Eingangsspannung von 7 Volt bis 40 Volt und eine fixe Ausgangsspannung von 5 Volt festlegt sind.

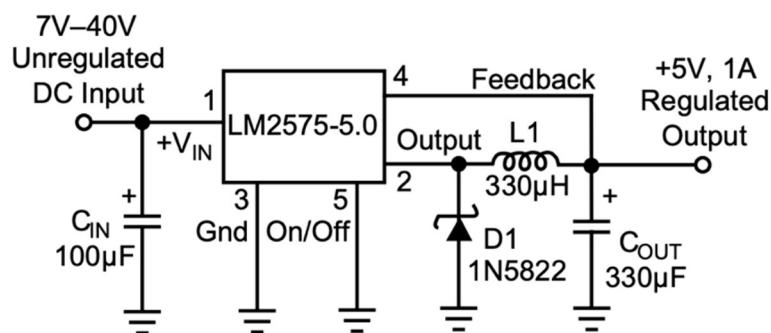


Abbildung 31: Typische Anwendung des LM2575-5.0 (Quelle: Datenblatt LM2576, Micrel)

3.6.2.3 Verwendung

Der Buck-Converter wird verwendet um die 12 Volt Eingangsspannung auf 5 Volt zu regulieren. Dies wird benötigt da der Schrittmotor mit einer Spannung von 12 Volt betrieben wird und das ESP-8266 mit einer Spannung von 5 Volt versorgt wird.

Für diesen Anwendungsfall wurde ein LM2576-5.0 verwendet.

3.6.3 Low-Drop-Out Voltage Regler

Low Drop Out Regler werden als kostengünstige Methoden zur Regelung Ausgangsspannungen verwendet. Low Drop Out Regler werden bei den meisten Anwendungen in erster Linie verwendet, um eine stabile Gleichspannung zu erzeugen. Diese geglättete Spannung soll empfindliche Lasten vor störbehafteten Energiequellen schützen.

3.6.3.1 Funktionsweise

Abbildung 32 zeigt den Grundaufbau eines LDO. Die stabile Referenzspannung und die Ausgangsspannung werden an die Eingänge des Differenzverstärkers geschaltet. Der Differenzverstärker verstärkt dann die Spannungsdifferenz zwischen der Referenzspannung und der Spannungsabweichung der Ausgangsspannung. Diese Spannungsdifferenz steuert den Feldeffekttransistor. Der Feldeffekttransistor ist an den Ausgang des Differenzverstärkers geschlossen und wird von dessen Ausgangsspannung gesteuert. Heutzutage kommen jedoch einige Linearregler ohne den Feldeffekttransistor als Schaltregler aus, da sie nur noch mit sehr geringen Spannungen arbeiten.¹¹

3.6.3.2 Beschaltung

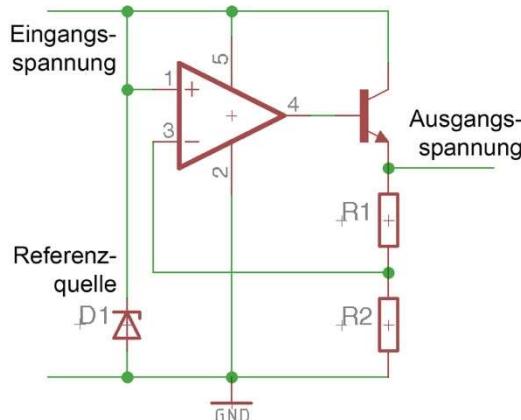


Abbildung 32: Aufbau Low Drop Out Regler

In Abbildung 32 ist zu sehen, dass die Ausgangsspannung und die Referenzspannung auf den Eingang des Low Drop Out Reglers rückgeführt werden um die Ausgangsspannung zu erzeugen. Der OPV wird intern von der Eingangsspannung des LDO versorgt.

¹¹ Vgl.: https://www.renesas.com/eu/en/www/doc/whitepapers/linear-regulator/understanding-ldos_de.pdf [19.01.2019 19:45]

3.6.3.3 Verwendung

Der Low Drop Out Regler bekommt die 5 Volt Spannung des Buck-Converters eingespeist. Diese 5V Spannung wird von dem Low Drop Out Regler auf 3,3 Volt gesetzt. Der Low Drop Out Regler dient nicht nur zur Verringerung der Spannung, sondern liefert gleichzeitig auch eine rippelfreie Gleichspannung für die Motoransteuerung.

3.6.4 Basisplatine

3.6.4.1 Funktion der Basisplatine

Die fertiggestellte Basisplatine ist modular aufgebaut. Das bedeutet, dass die Motortreiberplatine, der PIC, sowie das ESP8266 auf die Basisplatine einfach aufgesteckt werden können. Neben der Verbindung der einzelnen modularen Komponenten beinhaltet die Basisplatine außerdem die Hauptspannungsversorgung und die Ansteuerung für die Infrarot- und Fotodiode sowie den Hall-Sensor. Der Buck Converter befindet sich in der rechten unteren Ecke der Basisplatine, das ESP8266 auf der linken Seite, der PIC und die Motortreiberplatine werden in der Mitte der Basisplatine aufgesteckt.

Ebenfalls befinden sich die verschiedenen Eingänge und Ausgänge in Form von Schraubklemmen für die 12 Volt Eingangsspannung, 5 Volt Ausgangsspannung, den Schrittmotor, die Infrarotdiode, Fotodiode und den Hallsensor auf der Platine.

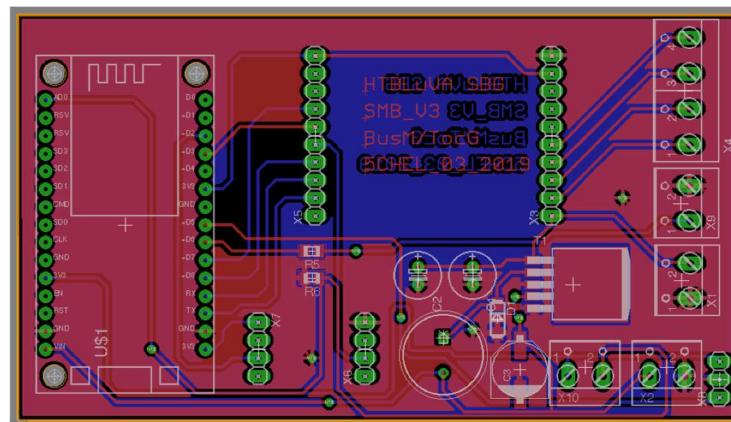


Abbildung 33: Basisplatine

3.7 Software

3.7.1 Arduino IDE

3.7.2 Android Studio

3.7.2.1 Wahl des Programms

Zur Programmierung der Smartphone Applikation wird Android Studio verwendet, welches speziell für Android Applikationen entwickelt wurde und damit eine zuverlässige Basis für dieses Projekt bietet. Neben Android Studio besteht auch die Möglichkeit, die Applikation in Eclipse zu programmieren, welche die zweite große Entwicklungsumgebung zum Erstellen von Android Applikationen ist. Android Studio wurde wie Android selbst von Google entwickelt und bietet daher die beste Kompatibilität zwischen den beiden Komponenten. Eclipse ist nicht nur zur Android Entwicklung verwendbar, sondern kann ebenfalls für Programmiersprachen wie C, C++ und andere verwendet werden, was die Navigation oft unnötig kompliziert macht und das Finden der relevanten Funktionen oft erschwert.

3.7.2.2 Datenbanken

Bei der Smart Medicine Box werden Datenbanken verwendet um die Ausgabezeiten online zu speichern. Da Datenbanken oftmals benutzt werden, um die Übertragung von Daten zwischen verschiedenen Systemen wie zum Beispiel einem Computer und einem Smartphone zu erleichtern. Sie ermöglichen eine dauerhafte zentrale Speicherung von Daten und bieten die Möglichkeit diese zu verwalten. Bei den meisten Datenbanken handelt es sich um sogenannte relationale Datenbanken, welche auf Standard SQL basieren. Je nach Hersteller gibt es verschiedene Abwandlung der Standard SQL Datenbanken. Im Allgemeinen wird bei Datenbanken von Datenverwaltung gesprochen, generell handelt es sich jedoch um ein Datenbankensystem. Dieses Datenbankensystem besteht aus einer Datenbank und einem Datenbankenmanagementsystem. Das Datenbankenmanagementsystem folgt den Anweisungen des Entwicklers, verwaltet und organisiert die Dateneinträge in der Datenbank.

Neben den relationalen Datenbanken gibt es auch so genannte noSQL Datenbanken. NoSQL bedeutet einfach gesagt, dass sie keine SQL Datenbanken sind, sie werden verwendet um Objekte und deren Eigenschaften zu speichern. NoSQL Datenbanken machen es einfacher große Mengen an Daten zu speichern und auszuwerten.

Im Vergleich zu noSQL Datenbanken eignen sich relationale Datenbanken nicht zum Speichern großer Datenmengen, da diese eine Integrationsüberprüfung durchführen und die Daten Normalisieren was dazu führt, dass sie länger brauchen um die Daten zu speichern. Normalisierung bedeutet, die Struktur einer Datenbank nach vorgegebenen Normalisierungsregeln zu verändern. Die Normalisierungsregeln bewirken, dass die Datenbank in einer Form dargestellt wird, in welcher keine vermeidbaren Redundanzen mehr existieren. Relationale Datenbanken werden daher hauptsächlich in Anwendungsfällen bei denen es auf Genauigkeit, Integrität oder Nachvollziehbarkeit ankommt verwendet. Daher werden sie meistens bei beispielsweise Banksystemen oder Transaktionssystemen eingesetzt.¹²

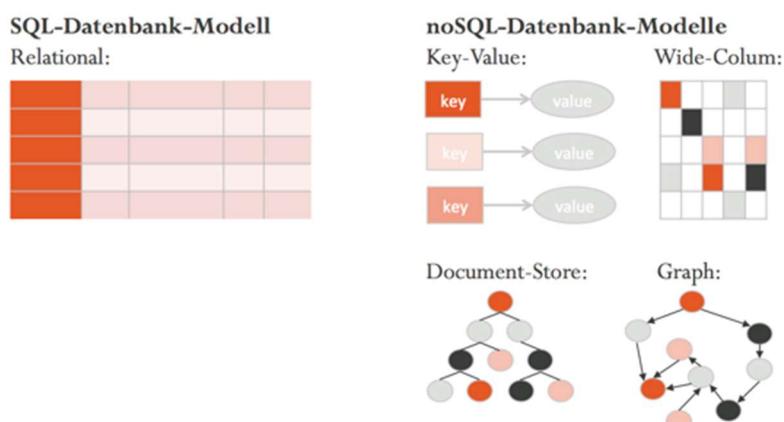


Abbildung 34:Relational und NoSQL (Quelle:schwarze-consulting.de)

Für dieses Projekt wird Googles Firebase Online Datenbankensystem verwendet. Firebase speichert Daten in der noSQL Form.

3.7.2.3 **Firebase**

Über Firebase können Web- und Smartphone Anwendungen entwickelt werden. Neben dem in diesem Projekt verwendetem Datenbankservice bietet Firebase noch weitere

¹² Vgl.: <http://www.datenbanken-verstehen.de/datenbank-grundlagen/einsatz-von-datenbanken/> [11.03.2019 9:15]

Produkte für Entwickler. Unter diese Produkte fällt zum Beispiel Analytics, welches kostenfrei die Möglichkeit bietet Daten zu analysieren, Cloud Speicher und viele mehr.

Firebase bietet verschiedene Pläne an, die von Gratis bis 25 Euro pro Monat reichen, für die in diesem Projekt benötigte Datenmenge reicht der gratis Spark Plan jedoch leicht.

Um nun eine Datenbank in Firebase anzulegen muss zuerst ein Account erstellt werden beziehungsweise kann auch ein Google Account verwendet werden um sich anzumelden und die die Firebase Services zu benutzen. Nun muss ein neues Projekt erstellt werden. In dem neu angelegten Projekt kann dann eine Realtime Database erstellt werden.

Bei Datenbanken können Accounts angelegt werden. Der Benutzer müsste sich daher bei jedem Öffnen der Applikation einloggen. Damit das Einloggen in die Applikation so einfach wie möglich gestaltet wird, müssen zwei Parameter in den Security Regeln von Firebase geändert werden, wodurch kein Login benötigt wird. Unter dem Eintrag "rules" müssen ".read" und ".write" von false auf true geändert werden. Dies führt dazu, dass jeder, der den Link für die Firebase Realtime Database und das Database Secret besitzt, die Datenbank benutzen kann.

```
1 ▾  {
2 ▾   "rules": {
3     ".read": true,
4     ".write": true
5   }
6 }
```

Abbildung 35:Firebase Security Einstellungen

Der Link zur Realtime Database kann direkt von der Seite mit den Datenbank Einträgen entnommen werden. Das Database Secret kann unter den Projekteinstellungen ausgelesen werden.¹³

Firebase kann über Android Studio relativ unkompliziert unter dem Menüpunkt Tools -> Firebase mit der Smartphone Applikation verbunden werden. Dafür muss im Firebase Untermenü "Realtime Database" gewählt werden und eine Verbindung zu der Firebase Datenbank erstellt werden.

¹³ Vgl.: <https://www.droidwiki.org/wiki/Firebase> [15.03.2019 18:30]

Die Verbindung mit dem ESP8266 wird erzeugt in dem das Database Secret und der Link zur Realtime Database in Programmcode als FIREBASE_HOST und FIREBASE_AUTH definiert werden und das ESP866 mit dem Internet verbunden wird. Wichtig ist auch dass die Firebase Libraries in den Programmcode importiert werden, damit die nachfolgenden Funktionen verwendet werden können. In weiterer Folge muss mit der Funktion `Firebase.begin(FIREBASE_HOST, FIREBASE_AUTH)` die Verbindung zum Firebase Server hergestellt werden.

3.7.2.4 Benutzeroberfläche

Nach dem Starten der Applikation wird der Benutzer von einem Startup Screen begrüßt. Ein Klick auf den Startup-Bildschirm leitet den Benutzer zur Dateneingabe der Applikation weiter.



Abbildung 36: Startup Screen und Dateneingabe

Sobald der Benutzer auf dem Dateneingabe Bildschirm gelandet ist können Werte eingegeben werden. Folgende Werte müssen nun angegeben werden um eine neue Ausgabezeit festzulegen: Es muss ein Tag von Tag1 bis Tag7 ausgewählt werden, das

Datum des Ausgabetags muss im Format [Tag:Monat:Jahr] angegeben werden, die Uhrzeit muss als [Stunde:Minute] festgelegt werden.

Sobald alle gerade genannten Werte eingegeben wurden, können sie durch drücken des plus Symbols in die Datenbank eingegeben werden.



Abbildung 37: Eintrag Hinzufügen & Liste öffnen

Eingegebene Werte werden nun Online in die Datenbank eingetragen und können vom Benutzer im Listen Menüpunkt ausgelesen und überprüft werden. Um auf die Liste zuzugreifen wird auf das Symbol am unteren Bildschirmrand gedrückt.

Unter dem Menüpunkt „Liste“ können die bereits eingegebenen Werte vom Benutzer überprüft werden.



Abbildung 38: Liste der eingegebenen Werte

Die einzelnen Tage werden mit den eingegebenen Uhrzeiten und Daten aufgelistet. Durch Betätigen der Pfeiltaste in der linken oberen Ecke kann der Benutzer wieder zum Eingabe-Bildschirm zurückkehren.

3.7.2.5 Datensendung

Die Daten müssen vom Smartphone an das ESP gesendet werden, das passiert jedoch nicht auf direktem Weg.

Der Datenaustausch erfolgt über die Google Plattform „Firebase“. Firebase bietet die Möglichkeit Daten (zum Beispiel einzelne Werte oder Wörter) in einer online Datenbank zu speichern. Die Werte werden nun vom Smartphone in die Datenbank geschrieben damit sie dann vom ESP wieder ausgelesen werden können.

Zuerst wird online unter www.firebaseio.google.com eine neue Datenbank erstellt.

Diese Datenbank bekommt automatisch ein sogenanntes Database Secret und eine API URL zugewiesen. Da Firebase sowie Android Studio von Google entwickelt worden sind, ist die Verbindung der Datenbank zur Smartphone Applikation relativ einfach. In Android Studio ist unter dem Menüpunkt „Tools“ Firebase auszuwählen und den weiteren Anweisungen zu folgen. Das Database Secret und die API URL werden weiterverwendet, um eine Verbindung zwischen dem ESP und der Datenbank herzustellen.

Es können nun in der Applikation Einträge in die Datenbank erstellt werden.

```
final DatabaseReference Day1dateD = myRef.child("Day1").child("day");
final DatabaseReference Day1dateM = myRef.child("Day1").child("month");
final DatabaseReference Day1dateY = myRef.child("Day1").child("year");
final DatabaseReference Day1timeH = myRef.child("Day1").child("hour");
final DatabaseReference Day1timeM = myRef.child("Day1").child("minute");
```

Abbildung 39: Datenbankeinträge Erstellen

Die Variable myRef steht hier für die Datenbank, in die Einträge gemacht werden sollen.

Mit dem „.child“-Befehl werden untergeordnete Einträge in der Datenbank erstellt. Die Datenbank sieht in Firebase, wie in Abbildung 40 veranschaulicht, aus.

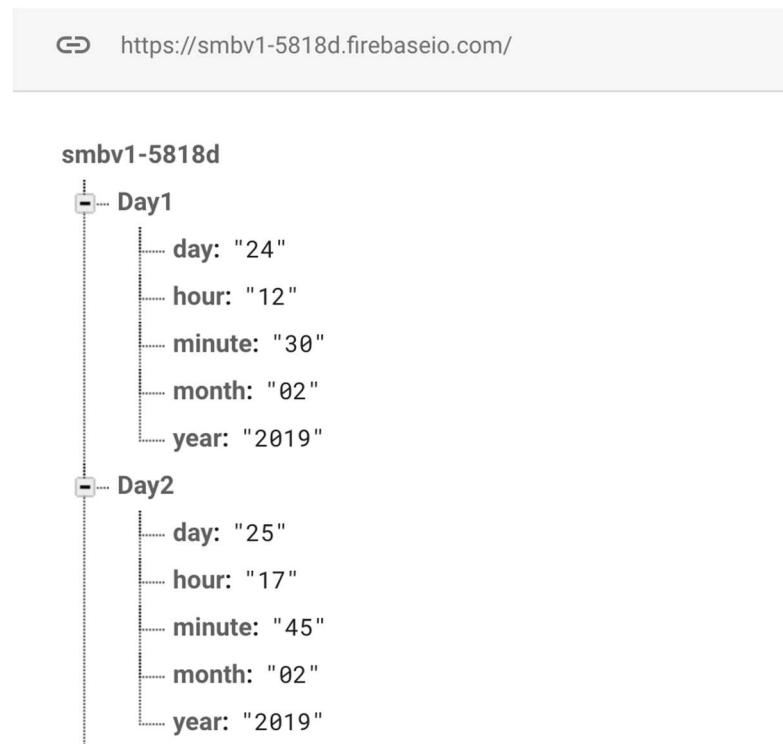


Abbildung 40: Datenbank in Firebase

Die Werte können jetzt in die Datenbank eingelesen werden. Die vom Benutzer eingegebenen Werte werden jeweils als ein String für das Datum und ein String für die Uhrzeit eingelesen. Die Strings werden dann an den Doppelpunkten aufgeteilt und in Arrays zerlegt. Dies liefert drei Teile für das Datum: date[0], date[1], date[2] und zwei Teile für die Uhrzeit: time[0], time[1]. Die jeweiligen Arrays werden dann mittels des Kommandos „setValue“ in die bereits erstellte Datenbank eingetragen.

```

if(day.equals("Day1")) {
    Day1dateD.setValue(date[0]);
    Day1dateM.setValue(date[1]);
    Day1dateY.setValue(date[2]);
    Day1timeH.setValue(time[0]);
    Day1timeM.setValue(time[1]);
}

else if(day.equals("Day2")){
  
```

Abbildung 41: Werte Eingabe in Datenbank

Um die Eingegebenen Werte in die Datenbank zu schreiben muss der Benutzer den „Hinzufügen“ Button betätigen.

```
<ImageButton  
    android:id="@+id/addplus"  
    android:layout_width="wrap_content"  
    android:layout_height="wrap_content"  
    android:layout_marginStart="8dp"  
    android:layout_marginLeft="8dp"  
    android:layout_marginTop="8dp"  
    android:layout_marginEnd="8dp"  
    android:layout_marginRight="8dp"  
    android:layout_marginBottom="8dp"  
    android:src="@drawable/add"  
    android:background="#505050"  
  
    app:layout_constraintBottom_toBottomOf="parent"  
    app:layout_constraintEnd_toEndOf="parent"  
    app:layout_constraintStart_toStartOf="parent"  
    app:layout_constraintTop_toTopOf="parent"  
    app:layout_constraintVertical_bias="0.719" />
```

Abbildung 42: Image Button

Der „Hinzufügen“ Button sowie die meisten verwendeten Buttons, wurde in Android Studio als Image Button definiert. Ein Image Button ermöglicht es anstatt Text ein Bild anzuzeigen mit welchem der Benutzer interagieren kann. Standardmäßig sieht ein Image Button wie ein normaler Button mit einer vordefinierten Farbe, welche sich mit den verschiedenen Zuständen ändert, aus. Mit android:src“@drawable/add“ kann nun das Bild für den Button definiert werden. „@drawable“ beschreibt in welchem Directory sich das zu verwendende Bild befindet, in diesem Fall befinden sich alle Bilder im Unterordner “drawable“ des “resources“ Ordners.

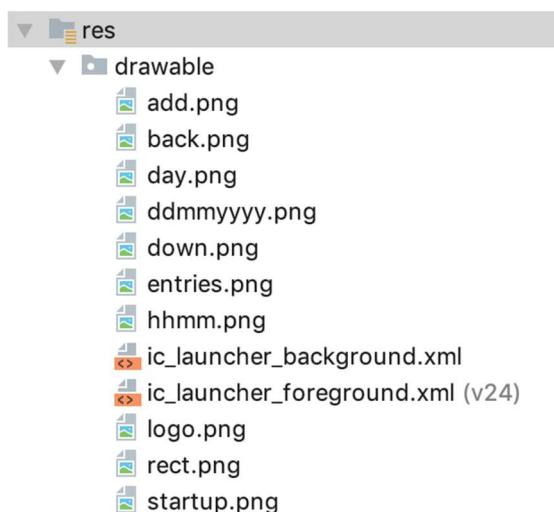


Abbildung 43: Bilder in Android Studio

Mit "add" wird nun das Gewünschte Bild ausgewählt, in diesem Fall "add.png". Wichtig ist hierbei, dass die verwendeten Bilder PNG Dateien sind, da andere Dateiformate wie zum Beispiel JPEG nicht unterstützt sind und weder von Image View noch als Image Button dargestellt werden können.

3.7.2.6 Datenempfang am Mikrocontroller

Um die Daten aus der Datenbank am ESP auslesen zu können wird eine von Firebase zur Verfügung gestellte Arduino Library benutzt, welche das Schreiben und das Auslesen von Daten ermöglicht.

```
1 #include <FirebaseCloudMessaging.h>
2 #include <Firebase.h>
3 #include <FirebaseHttpClient.h>
4 #include <FirebaseArduino.h>
5 #include <FirebaseError.h>
6 #include <FirebaseObject.h>
7
8 #include <FirebaseCloudMessaging.h>
9 #include <Firebase.h>
10 #include <FirebaseHttpClient.h>
11 #include <FirebaseArduino.h>
12 #include <FirebaseError.h>
13 #include <FirebaseObject.h>
```

Abbildung 44: included Firebase Library

Damit die Daten aus der richtigen Datenbank ausgelesen werden können, müssen zuerst zwei wichtige Variablen festgelegt werden. Die benutzerdefinierte URL der Datenbank und das sogenannte „database secret“ für die zu verwendende Datenbank werden dazu mittels „#define“ definiert.

```
25 //firebase definitions
26 #define FIREBASE_HOST "smbv1-5818d.firebaseio.com"
27 #define FIREBASE_AUTH "VBKNZv92C4CGGjVLFUXMBBjS4tr3xnMr0rT1yalz" // database secret auf firebase
28
```

Abbildung 45: Firebase Host & Database Secret

Um die Werte nun aus Firebase einlesen zu können, müssen zuerst noch die Pfade, unter denen die einzelnen Variablen in Firebase gespeichert sind, definiert werden.

```
29 //firebase paths to strings
30 #define day1dateD "/Day1/day"
31 #define day1dateM "/Day1/month"
32 #define day1dateY "/Day1/year"
33 #define day1timeH "/Day1/hour"
34 #define day1timeM "/Day1/minute"
```

Abbildung 46: Pfaddefinierung in Arduino

Die Werte der Variablen können nun als String oder Integer gespeichert werden.

Mit der Funktion `Firebase.getString(day1dateD);` kann der Wert, welcher in der Datenbank unter dem oben definierten Pfad für `day1dateD` liegt, ausgelesen und als String gespeichert werden.

```
104     dayIN = Firebase.getString(day1dateD);
105     monthIN = Firebase.getString(day1dateM);
106     yearIN = Firebase.getString(day1dateY);
107     hourIN = Firebase.getString(day1timeH);
108     minuteIN = Firebase.getString(day1timeM);
109
```

Abbildung 47: In Android aus der Datenbank lesen

Die Ausgelesenen Werte können nun zum Vergleichen des Datums weiterverwendet werden. Dies erfolgt in dem sich das ESP immer wieder die aktuelle Uhrzeit und das Datum aus dem Internet besorgt.

3.7.2.7 Datenempfang in der Applikation

Die Applikation soll nicht nur die Werte in die Datenbank schreiben, sondern auch Informationen über die Entnahme der Tabletten erhalten. Daher müssen die Inhalte der Datenbank von der Applikation auch ausgelesen werden können. Zum Auslesen der Daten wird die Funktion “`readFromDatabase()`“ verwendet. Diese Funktion wird einmal beim Öffnen der Applikation und dann sobald sich ein Wert in der Datenbank ändert ausgeführt. Bei einem Aufruf der Funktion bekommt der String “`value`“ den in der Datenbank eingetragenen Wert mittels der `getKey()` Funktion überliefert. In weiterer Folge wird dann der Wert der Variable `value` über `textView.setText(value)` als Text unter dem Menüpunkt “Liste“ ausgegeben.

```
private void readFromDatabase(){
    // Read from the database
    myRef.addValueEventListener(new ValueEventListener() {
        @Override
        public void onDataChange(DataSnapshot dataSnapshot) {
            String value = dataSnapshot.child("Day3").child("day").getKey();
            Log.v( tag: "read", msg: "Value is: " + value);
            textView1.setText(value);
        }
        @Override
        public void onCancelled(DatabaseError error) {
            // Failed to read value
            Log.v( tag: "read", msg: "Failed to read value.", error.toException());
        }
    });
}
```

Abbildung 48: In Java aus der Datenbank lesen

day1  

Abbildung 49: Listeneintrag

Da die Liste für alle sieben Tage aktuell sein muss erfolgt dies unabhängig davon welcher Tag oder welches Datum sich geändert hat. Dies garantiert, dass die Liste immer den aktuellsten Wert beinhaltet.

4 Ergebnisse

4.1 Prototyp

4.1.1 Steckbrettaufbau

Ein allerersten Prototypenaufbau erfolgt auf einem Steckbrett. Hier sind einzelne Komponenten wie Motortreiber-Platine mit dem ESP über Kabel verbunden. Der Aufbau auf einem Steckbrett empfiehlt sich sehr, da hier falsch gelegte oder vergessene Verbindungen ganz einfach auf- und umgesteckt werden können.

Die Spannungsversorgung erfolgt vorerst über ein schulinternes Netzteil, welches auf 12V und 1Ampere-Strombegrenzung eingestellt wird. Mit diesem Aufbau kann getestet werden, ob der Motor sich nach Vorgaben vom ESP drehen lässt. Hierbei können die Schrittanzahl und Drehrichtung frei justiert werden. Sobald die Ansteuerung des Schrittmotors über den Taktzeugenden PIC problemlos erfolgt, kommen nach und nach mehr Komponenten mit in den Prozess. Der Lichtschranken wird parallel zur Motoransteuerung getestet. Fotodiode und Infrarot-LED werden gemäß der Beschaltung aus 3.4.1 Fotodiode & IR-LED miteinander beziehungsweise mit dem ESP verbunden. Getestet werden kann die Funktion dieser über den ADC-Pin am ESP. Indem man die Luftbrücke zwischen den beiden Dioden unterbricht und wieder frei lässt, kann man in der Arduino-Konsole die unterschiedlichen Spannungswerte ablesen, welche die Fotodiode dem ESP übergibt. Ist keine Luftbrücke vorhanden, sollten die Werte annähernd Null sein, ansonsten ist mit Werten von 200-1000 zu rechnen. (Siehe: 3.4.1.4 Programmierung)

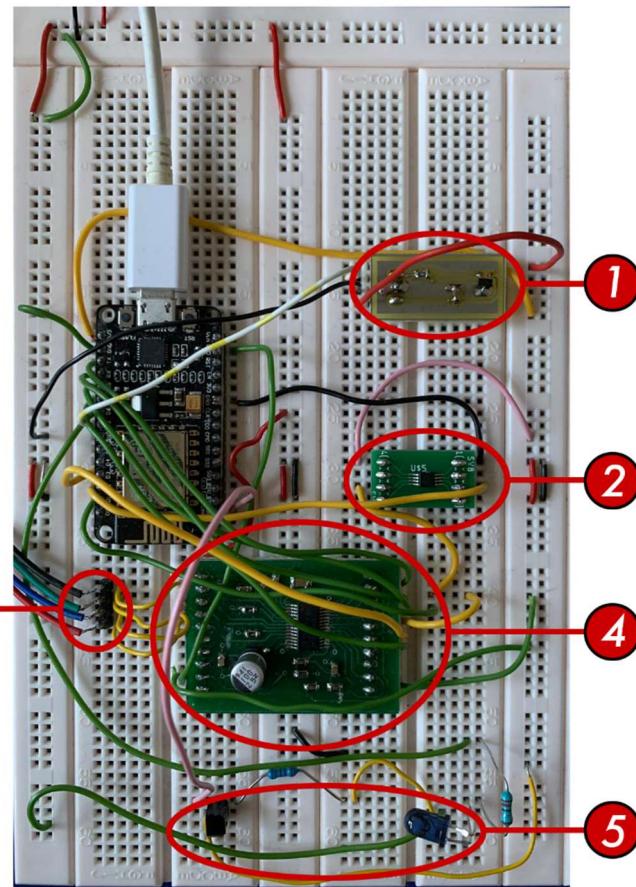


Abbildung 50: Prototyp am Steckbrett

In Abbildung 50 mit „1“ markiert ist der Hallsensor, welcher im späteren Verlauf des Projekts jedoch nicht implementiert wird (Siehe 3.4.2 Hallsensor). Mit „2“ markiert ist der PIC, welcher die Schnittstelle von ESP zu Motortreiberplatine „4“ herstellt und dem Motor „3“ somit mitteilt, um wie viele Schritte sich dieser drehen soll. An unterster Stelle vom Steckbrett ist die Lichtschranke „5“, bestehend aus IR-LED und IR-Fotodiode, montiert. Mit diesem Aufbau kann getestet werden, ob der Motor sich nach Vorgaben vom ESP drehen lässt.

4.2 Gefertigte Basisplatine

Sobald alle Verbindungen der einzelnen Komponenten zum ESP auf dem Steckbrett auf ihre Richtigkeit überprüft wurden, können diese auf der Basisplatine eingesetzt werden. (Siehe Kapitel 3.6.4 Basisplatine)

Ein Prototyp der gefertigten Platine mit all ihren Komponenten sieht demnach wie folgt aus.

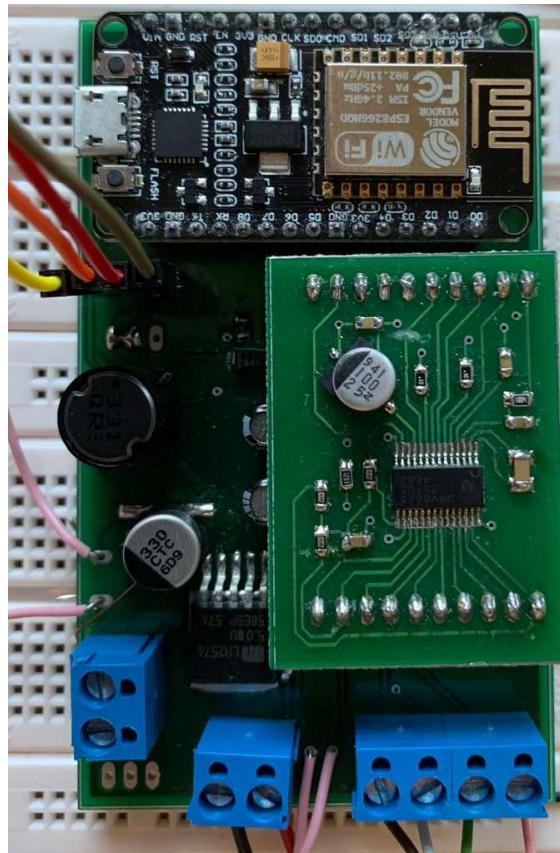


Abbildung 51: Prototyp der Basisplatine mit Modular-Komponenten

Die Platine verfügt mehrere Schraubklemmen, an welche Spannungsversorgung, Hallsensor, Fotodiode, Infrarot-LED, PIC-Taktgenerator und Schrittmotor angeschlossen werden. Ebenfalls wird hier das ESP und die Motortreiber-Platine aufgesteckt. Zu beachten ist hierbei, dass das ESP nun über die Platine mit 5V Spannung versorgt wird und somit keine zusätzliche Spannungsquelle benötigt wird. Das ESP wird nur an einen Computer angeschlossen, wenn Änderungen am Programm vorgenommen werden müssen. Das ESP startet von selbst, sobald es mit Spannung versorgt wird, verbindet sich mit dem Internet und holt sich relevante Daten aus der Datenbank.

Da dies nur ein Prototyp der Basisplatine ist, sind der PIC, Fotodiode und IR-LED nur provisorisch angelötet. Ebenfalls ist die Anordnung des Mikrocontrollers und der Motortreiber-Platine hier nicht optimal gewählt, da sich diese überlappen. Die endgültige Version der Basisplatine wurde vergrößert, damit die aufzusteckenden Komponenten genug Platz haben.

4.3 Gefertigtes Gehäuse

Im CNC-gefrästen Gehäuse wird die Basisplatine mit ihren Komponenten untergebracht. Durch eine Aussparung auf der Rückseite des Gehäuses, kann das Kabel zur Spannungsversorgung der Platine eingeführt werden. Fotodiode und IR-LED sind fest mit dem Gehäuse verbunden und bilden einen Lichtschranken um die Entnahme der Medikamente zu überprüfen.

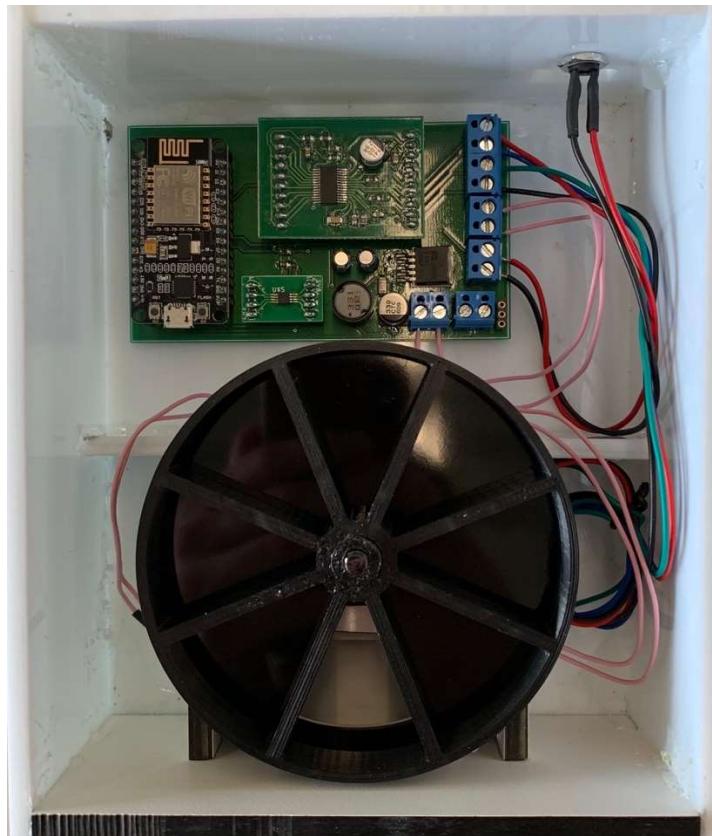


Abbildung 52: Innenleben der SmartMedicineBox

5 Literaturverzeichnis

-
- [1] [Online]. Verfügbar unter: <https://www.mikrocontroller.net/mc-project/Pages/Robotik/Mechanik/mechanik.html>; S. 2; [Zugriff: 05.11.2018 08:32]
 - [2] Deltron AG, „Schrittmotor kurz erklärt“. [Online]. Verfügbar unter: https://wiki.ntb.ch/infoportal/_media/hardware/sysebauteile/schrittmotor_kurz_erklaert_d.pdf [Zugriff: 05.11.2018 14:38]
 - [3] Microchip, „PIC12(L)F157/2“. [Online]. Verfügbar unter: <http://ww1.microchip.com/downloads/en/DeviceDoc/40001723D.pdf>; S. 3; [Zugriff: 26.11.2018 17:05]
 - [4] Texas Instruments, „DRV8825“. [Online]. Verfügbar unter: <http://www.ti.com/lit/ds/symlink/drv8825.pdf>; S.18f; [Zugriff: 12.11.2018 12:35]
 - [5] Texas Instruments, „DRV8825“. [Online]. Verfügbar unter: <http://www.ti.com/lit/ds/symlink/drv8825.pdf>; S.13f; [Zugriff: 19.11.2018 17:30]
 - [6] [Online]. Verfügbar unter: https://www.awmf.org/uploads/tx_szleitlinien/002-010I_S1_Infrarotstrahlung_W%C3%A4rmestrahlung_2012.pdf; [Zugriff: 19.12.2018 10:38]
 - [7] Vishay, „TSUS5200“. [Online]. Verfügbar unter: <https://www.vishay.com/docs/81055/tsus520.pdf>; S.2; [Zugriff: 27.01.2019 16:00]
 - [8] Vishay, „BPW41N. [Online]. Verfügbar unter: <https://www.vishay.com/docs/81522/bpw41n.pdf>; S. 2; [Zugriff: 27.01.2019 16:30]
 - [9] Allegro, „Bipolar Switch Hall-Effect ICs“. [Online]. Verfügbar unter: <https://www.allegromicro.com/en/Design-Center/Technical-Documents/Hall-Effect-Sensor-IC-Publications/Bipolar-Switch-Hall-Effect-ICs.aspx>; [Zugriff: 21.03.2019 12:05]
 - [10] RS components, „Was ist Cad und wo wird es eingesetzt?“. [Online]. Verfügbar unter: <https://de.rs-online.com/web/generalDisplay.html?id=infozone&file=automation/cad>; S.1 [Zugriff: 16.03.2019]
 - [11] Renesas, „Understanding Idos“. [Online]. Verfügbar unter: https://www.renesas.com/eu/en/www/doc/whitepapers/linear-regulator/understanding-idos_de.pdf [Zugriff: 19.01.2019 19:45]
 - [12] Mandy Goram, „Einsatz von Datenbanken“. [Online]. Verfügbar unter: <http://www.datenbanken-verstehen.de/datenbank-grundlagen/einsatz-von-datenbanken/> [Zugriff: 11.03.2019 9:15]

[13] DroidWiki, „Firebase“. [Online]. Verfügbar unter:

<https://www.droidwiki.org/wiki/Firebase> [Zugriff: 15.03.2019 18:30]

6 Abbildungsverzeichnis

Abbildung 1: Vereinfachtes Blockschaltbild.....	11
Abbildung 2: GANTT-Diagramm Busse Max.....	16
Abbildung 3: GANTT-Diagramm Tockner Gregor.....	17
Abbildung 4: Systemkomponenten der SmartMedicineBox	18
Abbildung 5: Bipolar-Schrittmotor (Quelle: mikrocontroller.net).....	19
Abbildung 6: Schrittmotor Normal-Betrieb.....	20
Abbildung 7: Zeitliniendiagramm des bipolaren Schrittmotors.....	21
Abbildung 8: Zeitliniendiagramm des Step-Signals.....	22
Abbildung 9: Pinout des ESP8266.....	23
Abbildung 10: PIN Diagramm des PIC.....	24
Abbildung 11: Visualisierung der Programmkomponenten am ESP	25
Abbildung 12: Zeitvergleich am ESP	26
Abbildung 13: Motortransteuerung über H-Brücken nach Datenblattauszug (Quelle: ti.com)	27
Abbildung 14: Beschaltung des Motortreiber-IC nach Datenblattauszug	28
Abbildung 15: Individuelle Beschaltung des Motortreiber-IC	29
Abbildung 16: Gefertigte Motortreiber-Platine.....	30
Abbildung 17: IR-LED - Behälter - Fotodiode.....	33
Abbildung 18: IR-LED - kein Behälter - Fotodiode	33
Abbildung 19: Betriebsspannung Infrarot-LED	34
Abbildung 20: Strom und Spannungsverlauf Fotodiode	34
Abbildung 21:Beschaltung IR-LED & Fotodiode	35
Abbildung 22: Programmcode - Lichtschranken	36
Abbildung 23: Spannungsverlauf und Schaltschwellen beim Hallsensor (Quelle: allegro.com)....	38
Abbildung 24: Hallsensor Platine	38
Abbildung 25: Gehäuseaufbau in Easel.....	39
Abbildung 26: Assembliertes Gehäuse	40
Abbildung 27: 3D-Design der Trommel.....	41
Abbildung 28: Schaltung des Projekts inklusive Spannungsversorgungseinheit	42
Abbildung 29: Aufbau des Buck-Converters (Quelle: meilleureimage.eu)	43
Abbildung 30: Buck Converter	44
Abbildung 31: Typische Anwendung des LM2575-5.0 (Quelle: Datenblatt LM2576, Micrel)	44

Diplomarbeit	SmartMedicineBox	2018/2019
Abbildung 32: Aufbau Low Drop Out Regler	45	
Abbildung 33: Basisplatine	46	
Abbildung 34: Relational und NoSQL (Quelle:schwarze-consulting.de).....	48	
Abbildung 35: Firebase Security Einstellungen.....	49	
Abbildung 36: Startup Screen und Dateneingabe	50	
Abbildung 37: Eintrag Hinzufügen & Liste öffnen.....	51	
Abbildung 38: Liste der eingegebenen Werte	51	
Abbildung 39: Datenbankeinträge Erstellen.....	52	
Abbildung 40: Datenbank in Firebase.....	53	
Abbildung 41: Werte Eingabe in Datenbank	53	
Abbildung 42: Image Button	54	
Abbildung 43: Bilder in Android Studio.....	54	
Abbildung 44: included Firebase Library.....	55	
Abbildung 45: Firebase Host & Database Secret.....	55	
Abbildung 46: Pfaddefinierung in Arduino	56	
Abbildung 47: In Android aus der Datenbank lesen	56	
Abbildung 48: In Java aus der Datenbank lesen.....	57	
Abbildung 49: Listeneintrag	57	
Abbildung 50: Prototyp am Steckbrett	59	
Abbildung 51: Prototyp der Basisplatine mit Modular-Komponenten.....	60	
Abbildung 52: Innenleben der SmartMedicineBox	61	

7 Tabellenverzeichnis

Tabelle 1: Qualitätsanforderungen.....	15
Tabelle 2: Phasen des Schrittmotors	21
Tabelle 3: Schrittweite des Motors nach Datenblattauszug	31
Tabelle 4: Variabel-Einstellungen.....	32
Tabelle 5: Kostenaufstellung.....	65
Tabelle 6: Begleitprotokoll Busse Maximilian	67
Tabelle 7: Begleitprotokoll Tockner Gregor	69

8 Glossar

- SMB: „SmartMedicineBox“; Projekttitel der Medizinbox
- ESP: WLAN-Modul ESP8266-Platine mit WIFI-erzeugendem Modul
- GANTT: Diagramm zur zeitlichen Planung eines Projekts
- Datenbank: System zur Speicherung und zum Abrufen von großen Datenmengen
- SQL: Programmiersprache, mit der Datenbanken programmiert werden
- Array: Eine bestimmte Anordnung von Objekten oder Daten in einem Container

9 Kostenaufstellung

Kostenstelle	Preis
Schrittmotor	13,00€
ESP8266 Node MCU	6,50€
Widerstände, Kondensatoren und Spulen	5,00€
Motortreiber DRV8825PWP	4,60€
Buck Converter	2,00€
Fotodiode 950nm	0,50€
Infrarotdiode 950nm	0,25€
Basisplatine; HTL-Eigenbau	0,00€
Motortreiber-Platine; HTL-Eigenbau	0,00€
Summe	31,85€

Tabelle 5: Kostenaufstellung

10 Schlusswort

11 Begleitprotokoll gemäß § 9 Abs. 2 PrO

11.1 Begleitprotokoll Busse

Name:

Maximilian Busse

Diplomarbeitstitel:

SmartMedicineBox - Entwicklung einer automatisierten Medikamentenversorgung

KW	Beschreibung	Zeitaufwand
38	Diplomarbeitsantrag erstellt	2h
39	Diplomarbeitsantrag eingereicht	1h
40	Projektplanung, Aufgabenverfeinerung, Blockschaltbild, Zeitplan	4h
41	Diplomarbeit begonnen	2h
42	Platine zur Motoransteuerung in EAGLE entworfen	8h
43	Motor-Platine in Auftrag gegeben	8h
44	Software zur Motoransteuerung programmieren	8h
45	Arduino-Sketch der Motoransteuerung testen; ADC-Pin mit Potentiometer testen	8h
46	Datenblatt DRV8255 ausarbeiten + Timing Diagramm	8h
47	Platine löten, testen und Motor damit drehen lassen; Richtung und Drehgeschwindigkeit variieren	8h
48	Programmerweiterung; Schrittmotorbeschreibung in Diplomarbeit	8h
49	Beschreibung des Motortreiber-ICs und ESP8266; Fotodiode/IR-LED Werte auswerten	8h
50	Lichtschranken mit IR-LED und Fotodiode fertiggestellt; Platine für Hallsensor entworfen	8h
51	Projektpräsentation vor der Klasse; Platine für Hallsensor in Auftrag gegeben	8h
52	(Weihnachtsferien) Schriftliche Arbeiten an der Diplomarbeit	1h
1	(Weihnachtsferien) Schriftliche Arbeiten an der Diplomarbeit	1h
2	Schriftliche Arbeiten an der Diplomarbeit	8h

3	Programmierung des PIC (Taktsignal für Schrittmotor)	8h
4	Enable-Pin angesteuert für Stromsparmodus und Hitzereduktion beim Schrittmotor	8h
5	Hallsensorplatine gelötet und programmiert	8h
6	Schriftliche Arbeiten an der Diplomarbeit;	8h
7	(Semesterferien) Schriftliche Arbeiten an der Diplomarbeit	1h
8	Daten der Smartphone-Applikation auswerten	8h
9	Daten der Smartphone-Applikation auswerten	8h
10	Schriftliche Arbeiten an der Diplomarbeit; Gehäusebau; Lichtschranken in Gehäuse implementieren; Entnehmbare Schachtel gefertigt	8h
11	Fehlersuche (Verbindung zwischen ESP und Smartphone)	8h
12	Fehlersuche und Korrektur der Basisplatine	8h
13	Gehäuseassembly und Einbau der Sensorik; Schriftliche Arbeiten an der Diplomarbeit	8h
14	Gehäuseassembly und finale Korrekturen	8h

Tabelle 6: Begleitprotokoll Busse Maximilian

11.2 Begleitprotokoll Tockner

Name:

Gregor Maximilian Tockner

Diplomarbeitstitel:

SmartMedicineBox - Entwicklung einer automatisierten Medikamentenversorgung

KW	Beschreibung	Zeitaufwand
38	Diplomarbeitsantrag erstellt	2h
39	Diplomarbeitsantrag eingereicht	1h
40	Projektplanung, Aufgabenverfeinerung, Blockschaltbild, Zeitplan	4h
41	Diplomarbeit begonnen	2h
42	Entwurf der Schaltung für die Spannungsversorgung	8h
43	Entwurf der Schaltung für die Basisplatine	8h
44	Bauteilwertberechnung und Dokumentation der Berechnungen	8h
45	Suchen der benötigten Bauteile; Erstellen der Stücklisten	8h
46	Entwicklung der Basisplatine in Eagle	8h
47	Implementierung und Inbetriebnahme der Motoransteuerung	8h
48	Überarbeitung der Schaltung; Besorgen der Bauteile	8h
49	Entwicklung des Konzepts der Smartphone Applikation	8h
50	Entwurf der Basisplatine in Eagle	8h
51	Projektpräsentation vor der Klasse	8h
52	(Weihnachtsferien) Schriftliche Arbeiten an der Diplomarbeit	8h
1	(Weihnachtsferien) Schriftliche Arbeiten an der Diplomarbeit	8h
2	Überarbeitung des Konzepts der Smartphone Applikation	8h
3	Basisentwurf der Smartphone Applikation	8h
4	Verbindung der Smartphone Applikation und Firebase	8h
5	Entwicklung der Smartphone Applikation	8h
6	Design der Smartphone Applikation	8h
7	(Semesterferien) Schriftliche Arbeiten an der Diplomarbeit	1h
8	Applikations Design in Applikation einbinden	8h
9	Daten in Smartphone Applikation einlesen	8h

10	Verbindung zwischen ESP und Smartphone	8h
11	Fehlersuche (Verbindung zwischen ESP und Smartphone)	8h
12	Fehlersuche Basisplatine	8h
13	Fehlersuche und Korrektur der Basisplatine	8h
14	Gehäuseassembly und finale Korrekturen	8h

Tabelle 7: Begleitprotokoll Tockner Gregor

12 Anhang

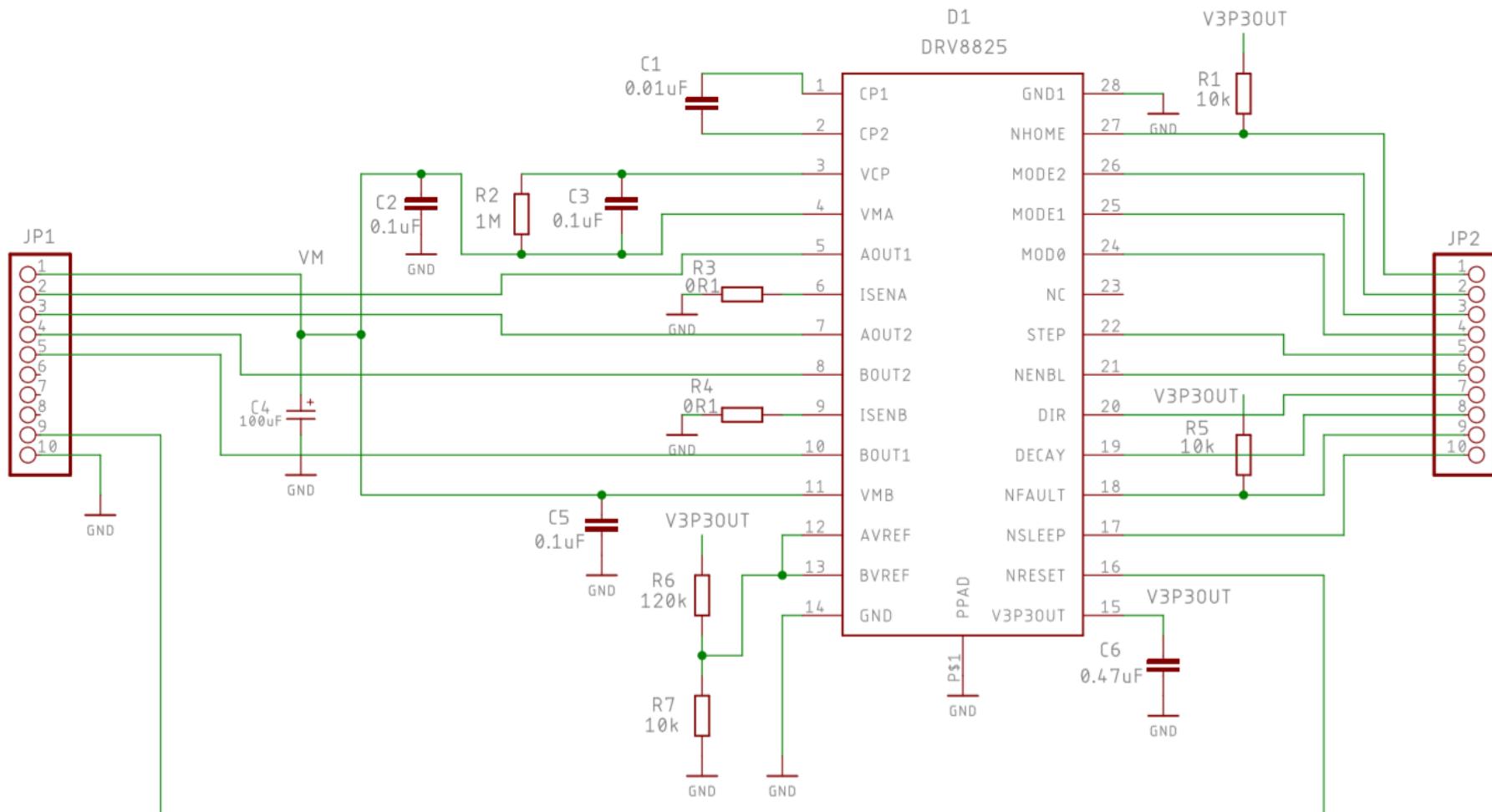
12.1 Konstruktionspläne

12.1.1 Motortreiberplatine

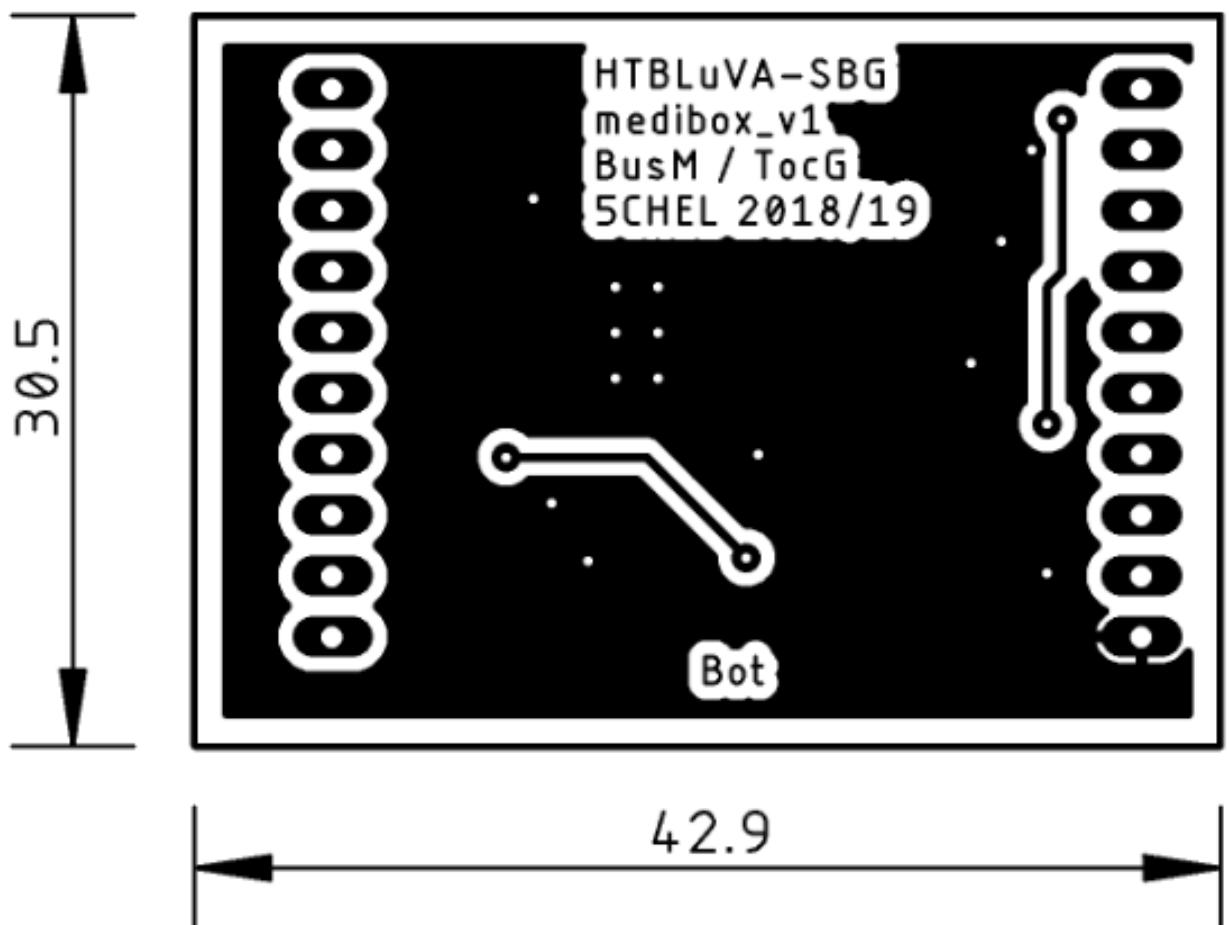
- i. Schaltplan
- ii. Layout Bottom
- iii. Layout Top
- iv. Bauteile Top
- v. Bohrungen
- vi. PCB Bottom
- vii. PCB Top

12.1.2 Basisplatine

- i. Schaltplan
- ii. Layout Bottom
- iii. Layout Top
- iv. Bauteile Top
- v. Bohrungen
- vi. PCB Bottom
- vii. PCB Top

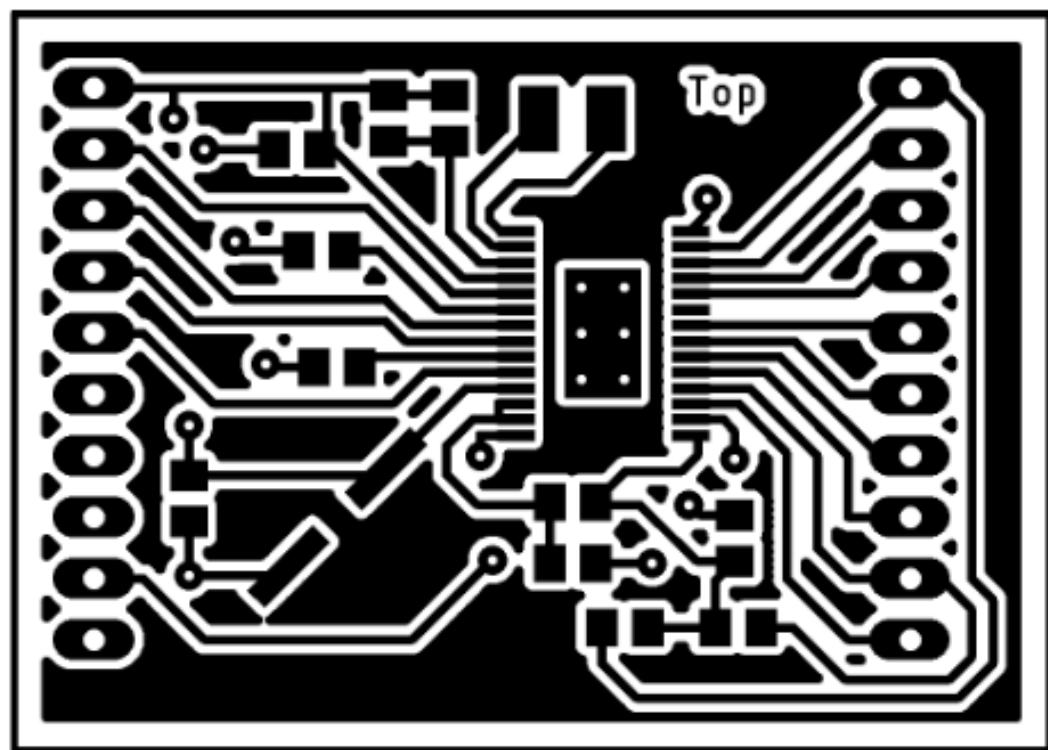


Sachnummer, Dateiname: Medibox_schaltplan.docx	Name: Busse Maximilian	Toleranz: keine	Werkstoff:
 HTBLuVA Salzburg Elektronik	ID-Nr.: 5CHEL	Gez.: BusM	Geprüft: BusM
Benennung: SMB	Betreuer: Sres	Dokumentart: Schaltplan	Freigabe:
Version: 1.0	Revision:	Dokumentstatus:	
Maßstab: ohne	Spr.: DE	Datum: 27.03.2019	Blatt: 1
Blätter: 1			



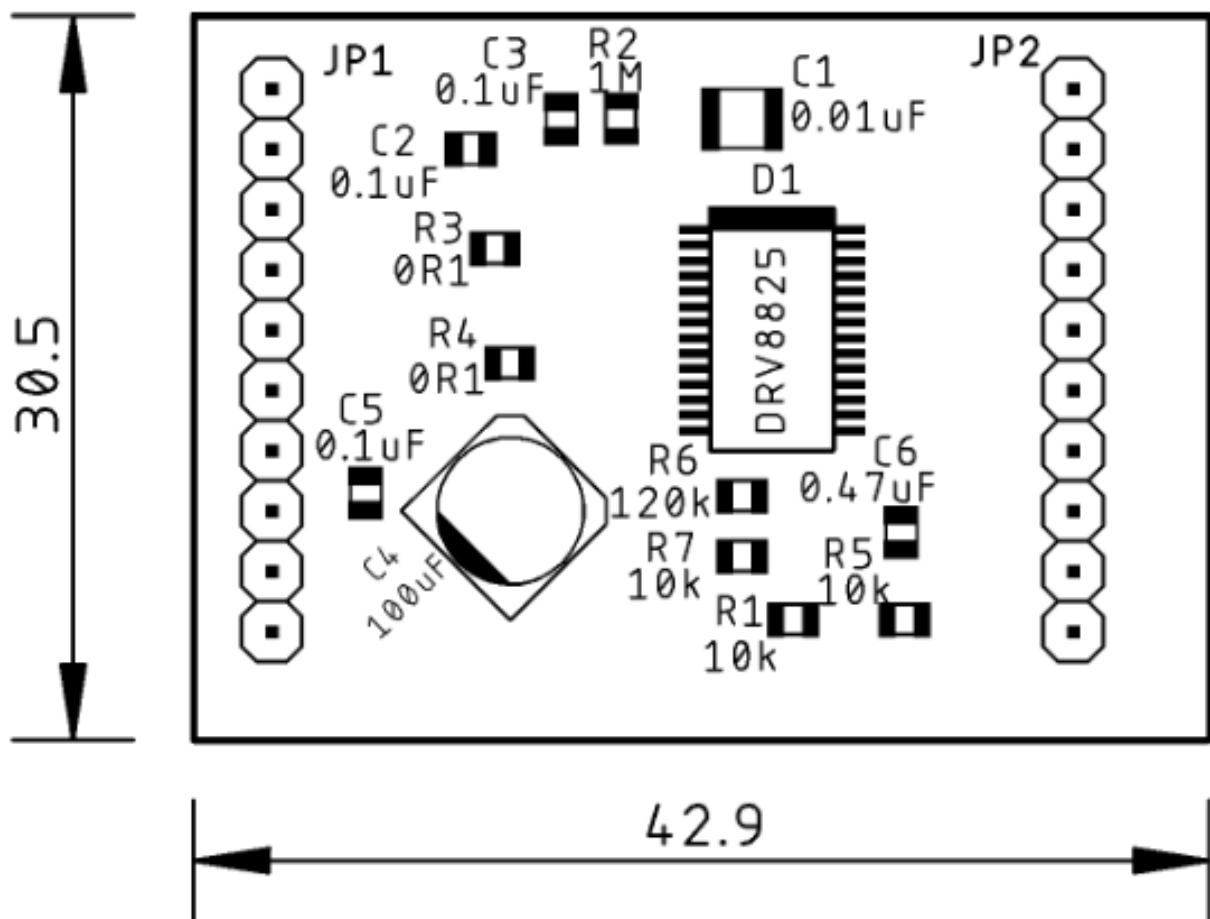
Sachnummer, Dateiname: Medibox_layout_bot.docx	Name: Busse Maximilian				Toleranz: keine	Werkstoff:	
	ID-Nr.: 5CHEL	Gez.: BusM	Geprüft: BusM	Betreuer: SreS	Dokumentart: Layout Bottom	Freigabe:	
Benennung: SMB	Version: 1.0		Revision:	Dokumentstatus: Fertigung			
	Maßstab: ohne	Spr.: DE	Datum: 27.03.2019	Blatt: 1	Blätter: 1		

30.5

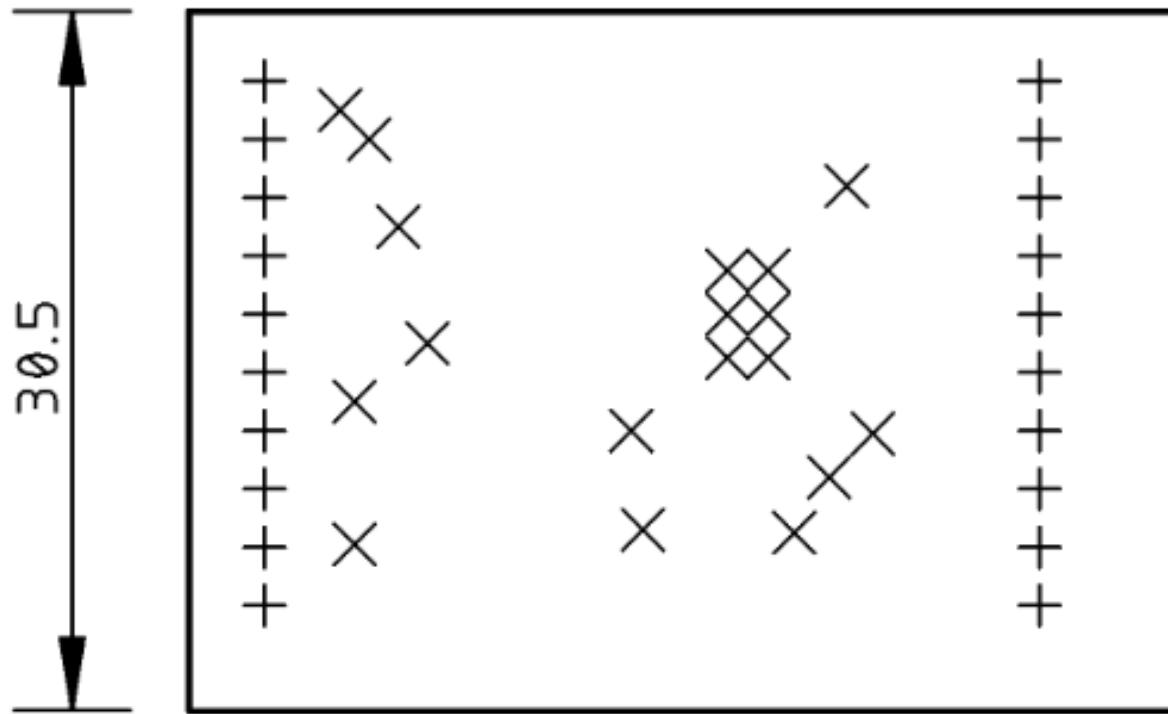


42.9

Sachnummer, Dateiname: Medibox_layout_top.docx	Name: Busse Maximilian	Toleranz: keine	Werkstoff:
 HTBLuVA Salzburg	ID-Nr.: 5CHEL	Gez.: BusM	Geprüft: BusM
	Betreuer: SreS	Dokumentart: Layout Top	Freigabe:
	Benennung: SMB	Version: 1.0	Revision: Dokumentstatus: Fertigung
		Maßstab: ohne	Spr.: DE
		Datum: 27.03.2019	Blatt: 1
			Blätter: 1

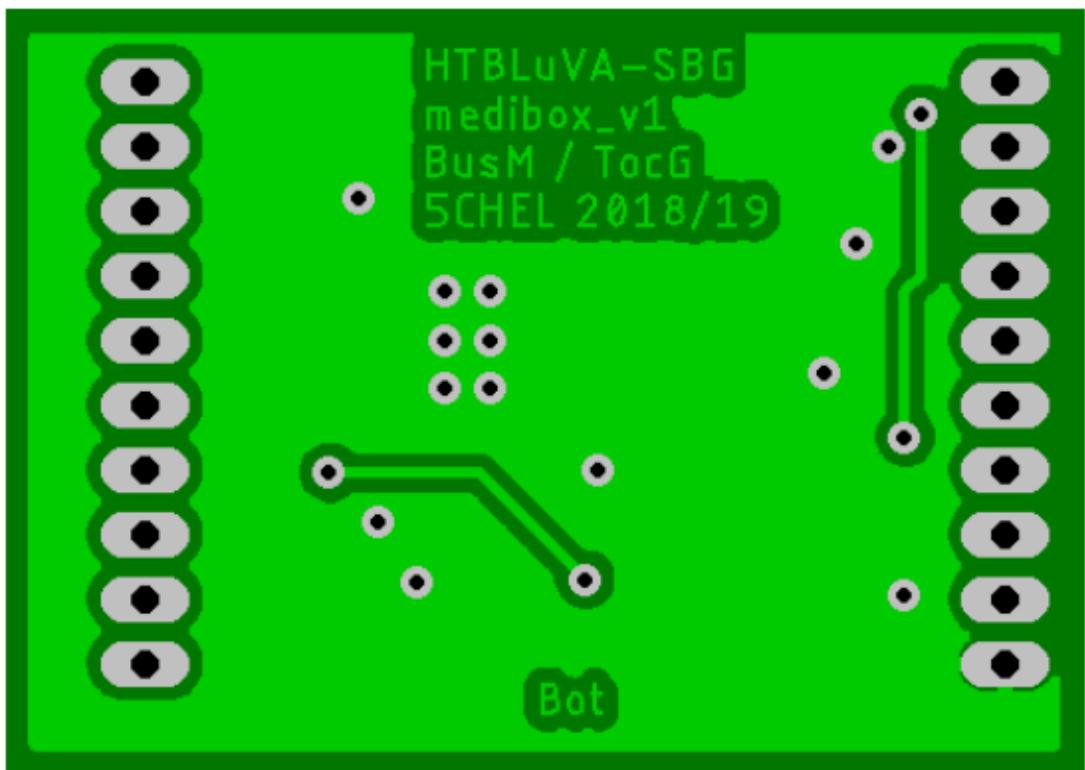


Sachnummer, Dateiname: Medibox_bestplan.docx	Name: Busse Maximilian				Toleranz: keine	Werkstoff:	
	ID-Nr.: 5CHEL	Gez.: BusM	Geprüft: BusM	Betreuer: SreS	Dokumentart: Bestückungsplan	Freigabe:	
HTBLuVA Salzburg Elektronik	Benennung: SMB				Version: 1.0	Revision:	Dokumentstatus: Fertigung
	Maßstab: ohne	Spr.: DE	Datum: 27.03.2019	Blatt: 1	Blätter: 1		

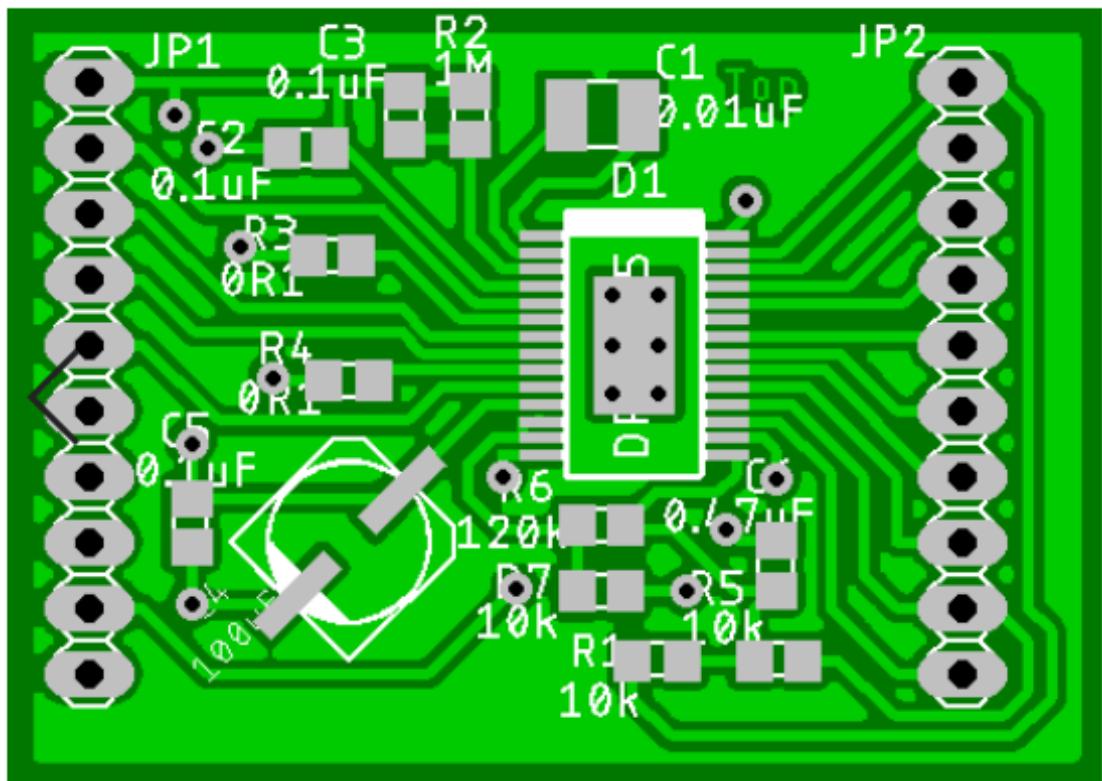


Sym.	Nr.	Durchm.	Anzahl	Durchk.
X	1	0.5	18	Ja
+	2	1.0	20	Ja

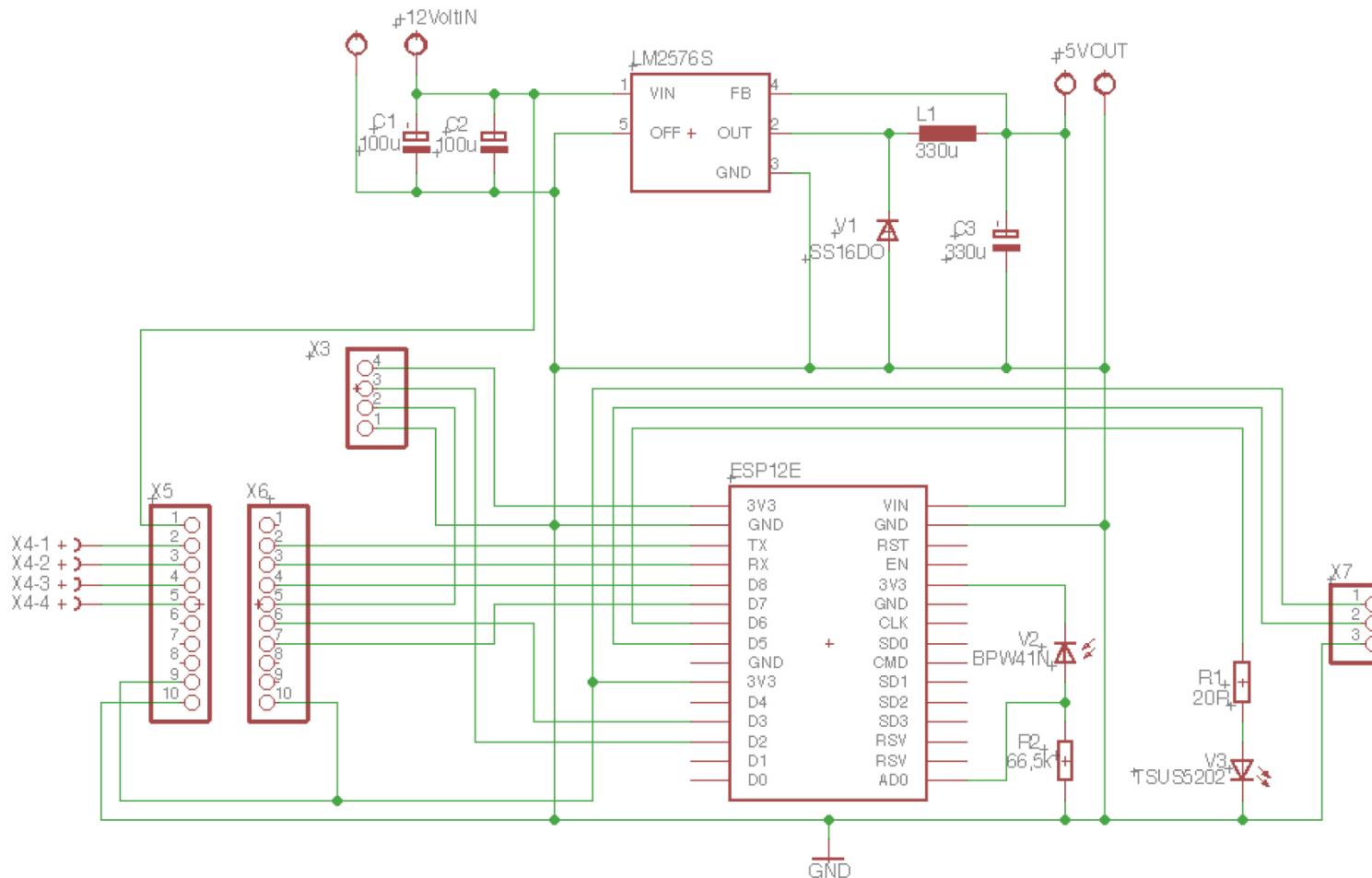
Sachnummer, Dateiname: Medibox_bohrungen.docx	Name: Busse Maximilian				Toleranz: keine	Werkstoff:	
 HTBLuVA Salzburg HTBLuVA Salzburg Elektronik	ID-Nr.:	Gez.:	Geprüft:	Betreuer:	Dokumentart:	Freigabe:	
	5CHEL	BusM	BusM	SreS	Bohrplan		
	Benennung: SMB				Version: 1.0	Revision:	Dokumentstatus: Fertigung
	Maßstab:	Spr.:	Datum:		Blatt:	Blätter:	
	ohne	DE	27.03.2019		1	1	



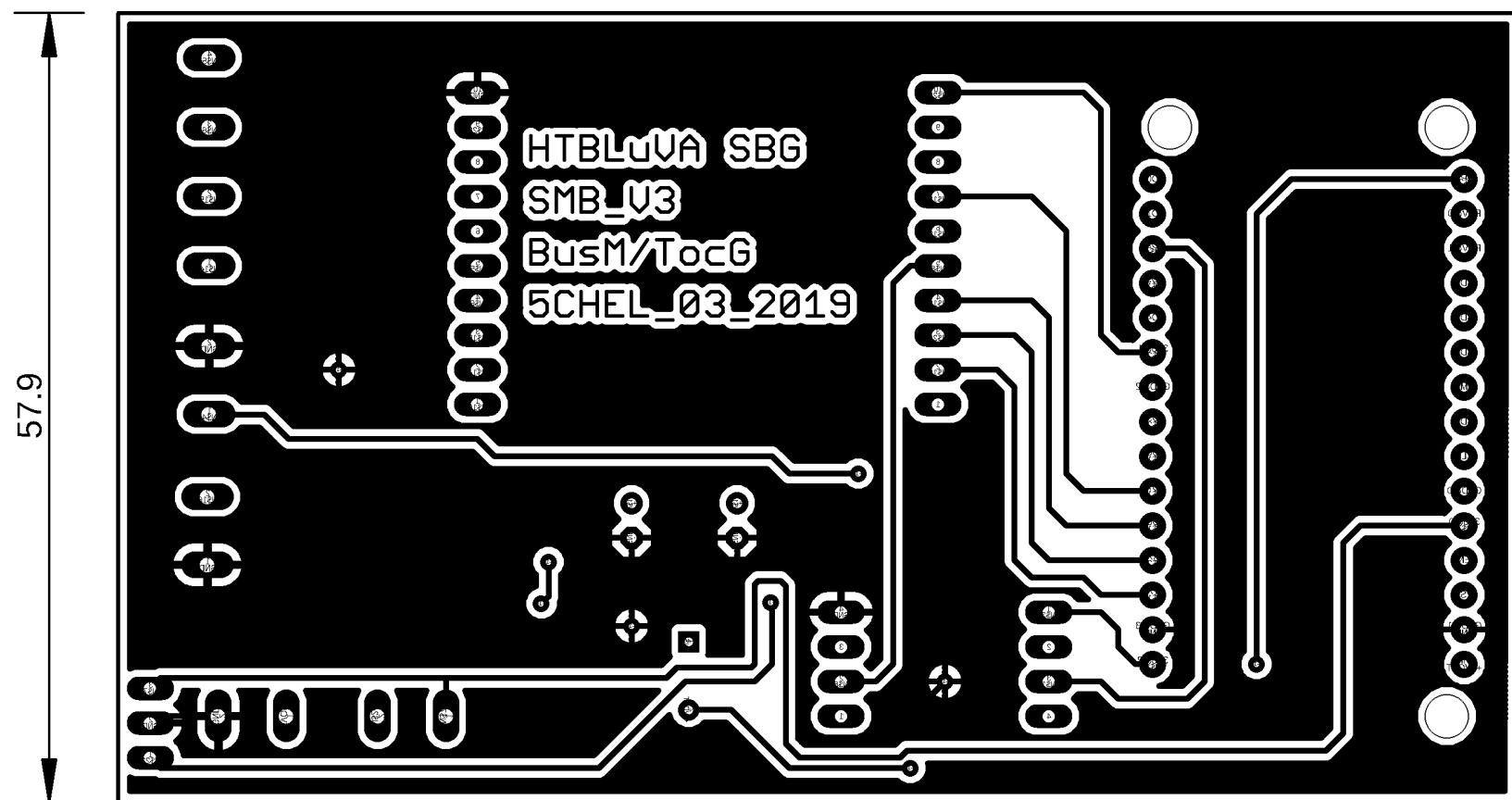
Sachnummer, Dateiname: Medibox_bcb_bot.docx	Name: Busse Maximilian				Toleranz: keine	Werkstoff:	
	ID-Nr.:	Gez.:	Geprüft:	Betreuer:	Dokumentart: PCB Bot	Freigabe:	
	5CHEL	BusM	BusM	SreS			
HTBLuVA Salzburg Elektronik	Benennung: SMB				Version: 1.0	Revision:	Dokumentstatus: Fertigung
	Maßstab:	Spr.:	Datum:		Blatt:	Blätter:	1 1
	ohne	DE	27.03.2019				



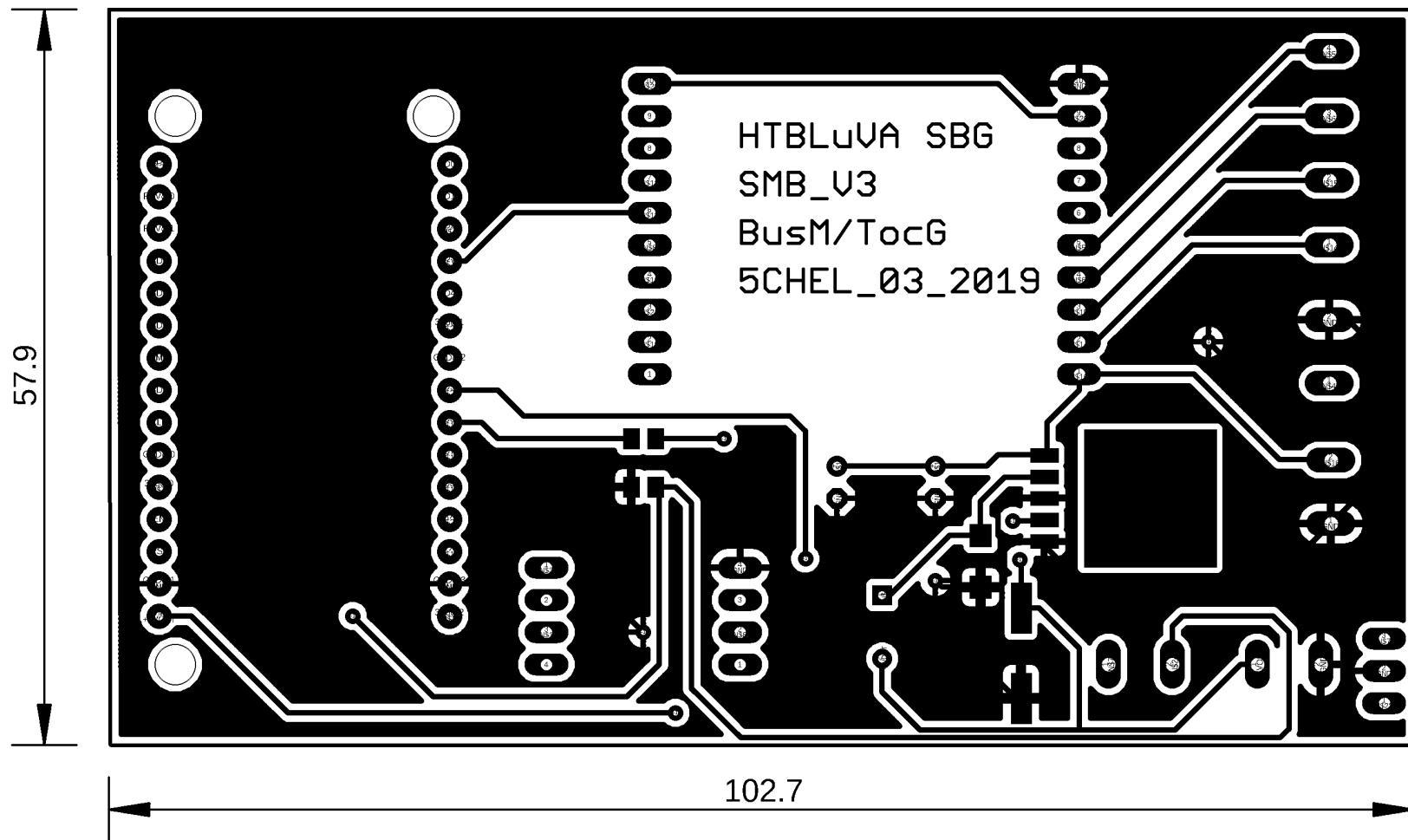
Sachnummer, Dateiname: Medibox_pcb_top.docx	Name: Busse Maximilian				Toleranz: keine	Werkstoff:	
 HTBLuVA Salzburg Elektronik	ID-Nr.:	5CHEL	Gez.:	BusM	Geprüft:	BusM	Betreuer: SreS
	Benennung: SMB				Dokumentart: PCB Top	Freigabe:	
	Version: 1.0		Revision:		Dokumentstatus: Fertigung		
		Maßstab:	Spr.:	Datum: 27.03.2019		Blatt:	Blätter: 1 1



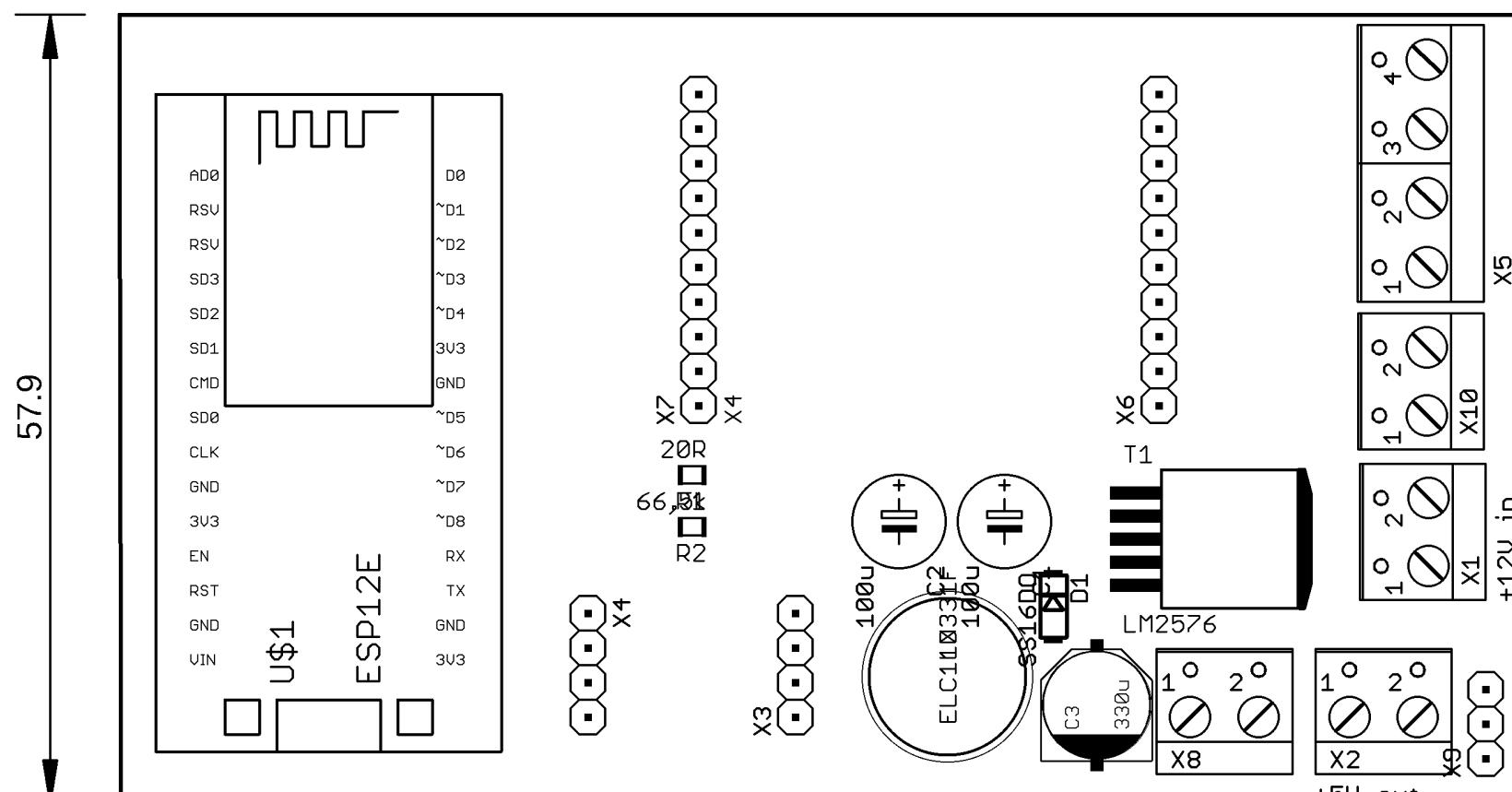
Sachnummer, Dateiname: smb_schaltplan.docx	Name: Tockner Gregor	Toleranz: keine	Werkstoff:
 HTBLuVA Salzburg	ID-Nr.: 5CHEL	Gez.: TocG	Geprüft: TocG
	Betreuer: SreS	Dokumentart: Schaltplan	Freigabe:
Benennung: SMB	Version: 1.0	Revision:	Dokumentstatus:
	Maßstab: DE	Spr.: DE	Datum: 27.03.2019
	Blatt: 1	Blätter: 1	



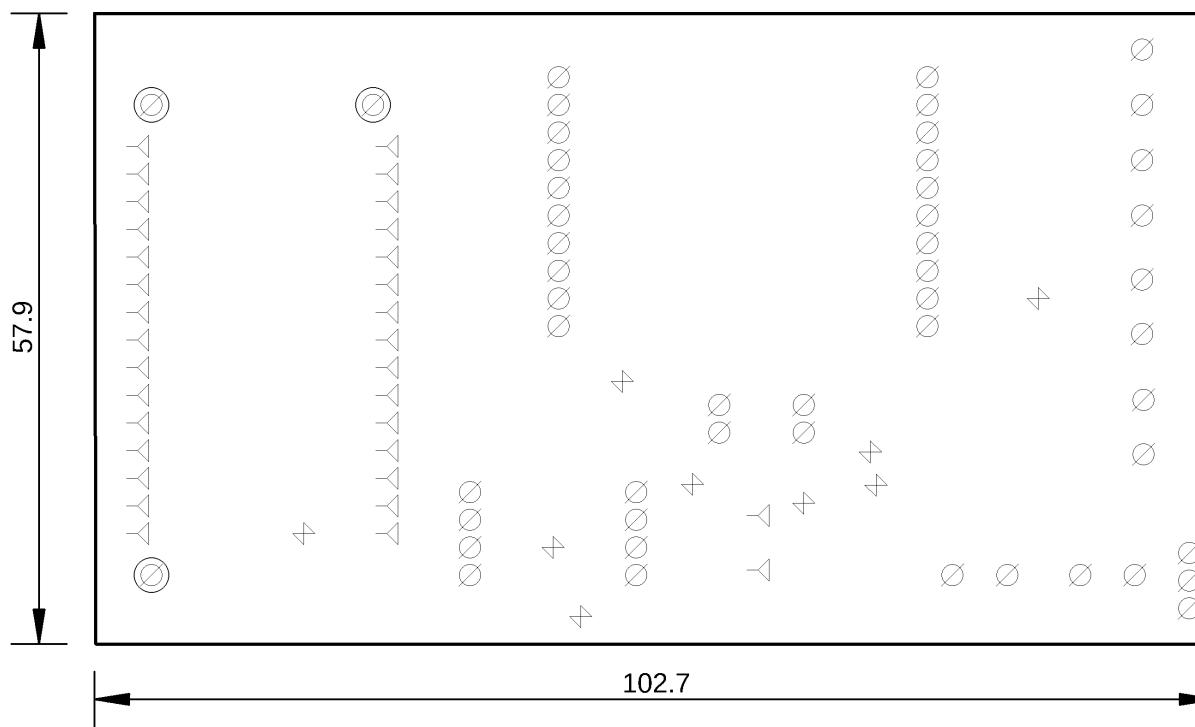
Sachnummer, Dateiname: smb_layout_bot.docx	Name: Tockner Gregor				Toleranz: keine	Werkstoff:
 HTBLuVA Salzburg	ID-Nr.: 5CHEL	Gez.: TocG	Geprüft: TocG	Betreuer: SreS	Dokumentart: Layout_bot	Freigabe:
Benennung: SMB	Version: 1.0			Revision:	Dokumentstatus:	
	Maßstab:	Spr.:	Datum:	27.03.2019	Blatt.:	1
					Blätter:	1



Sachnummer, Dateiname: smb_layout_top.docx	Name: Tockner Gregor	Toleranz: keine	Werkstoff:
 HTBLuVA Salzburg	ID-Nr.: 5CHEL	Gez.: TocG	Geprüft: TocG
	Betreuer: SreS	Dokumentart: Layout_top	Freigabe:
Benennung: SMB	Version: 1.0	Revision:	Dokumentstatus:
	Maßstab: DE	Spr.: DE	Datum: 27.03.2019
	Blatt:	Blätter:	1

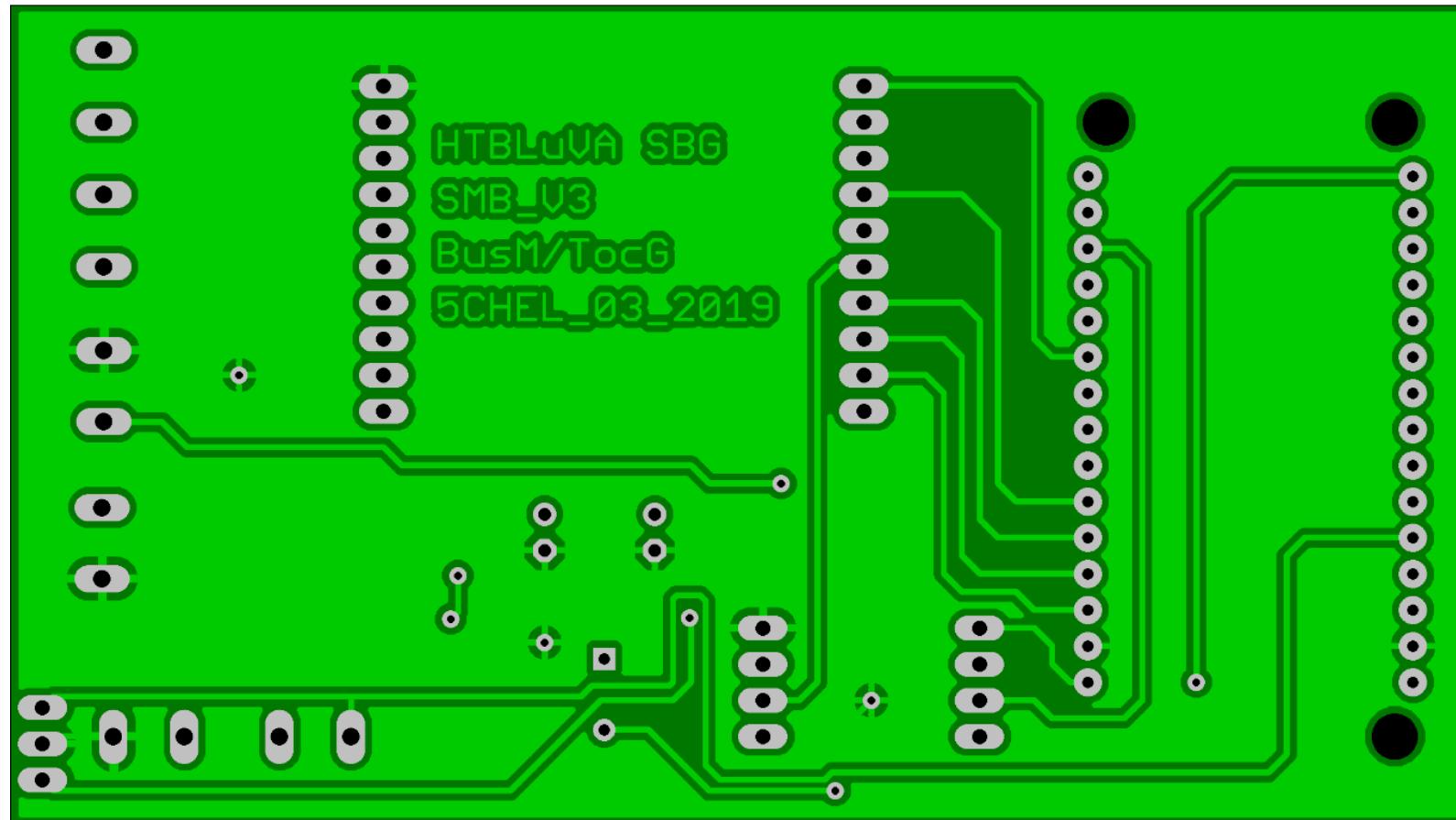


Sachnummer, Dateiname: smb_bauteile_top.docx	Name: Tockner Gregor	Toleranz: keine	Werkstoff:
 HTBLuVA Salzburg HTBLuVA Salzburg Elektronik	ID-Nr.: 5CHEL	Gez.: TocG	Geprüft: TocG
	Betreuer: SreS	Dokumentart: Bestückungsplan	Freigabe:
Benennung: SMB	Version: 1.0	Revision:	Dokumentstatus:
	Maßstab: DE	Spr.:	Datum: 27.03.2019
		Blatt:	1
		Blätter:	1

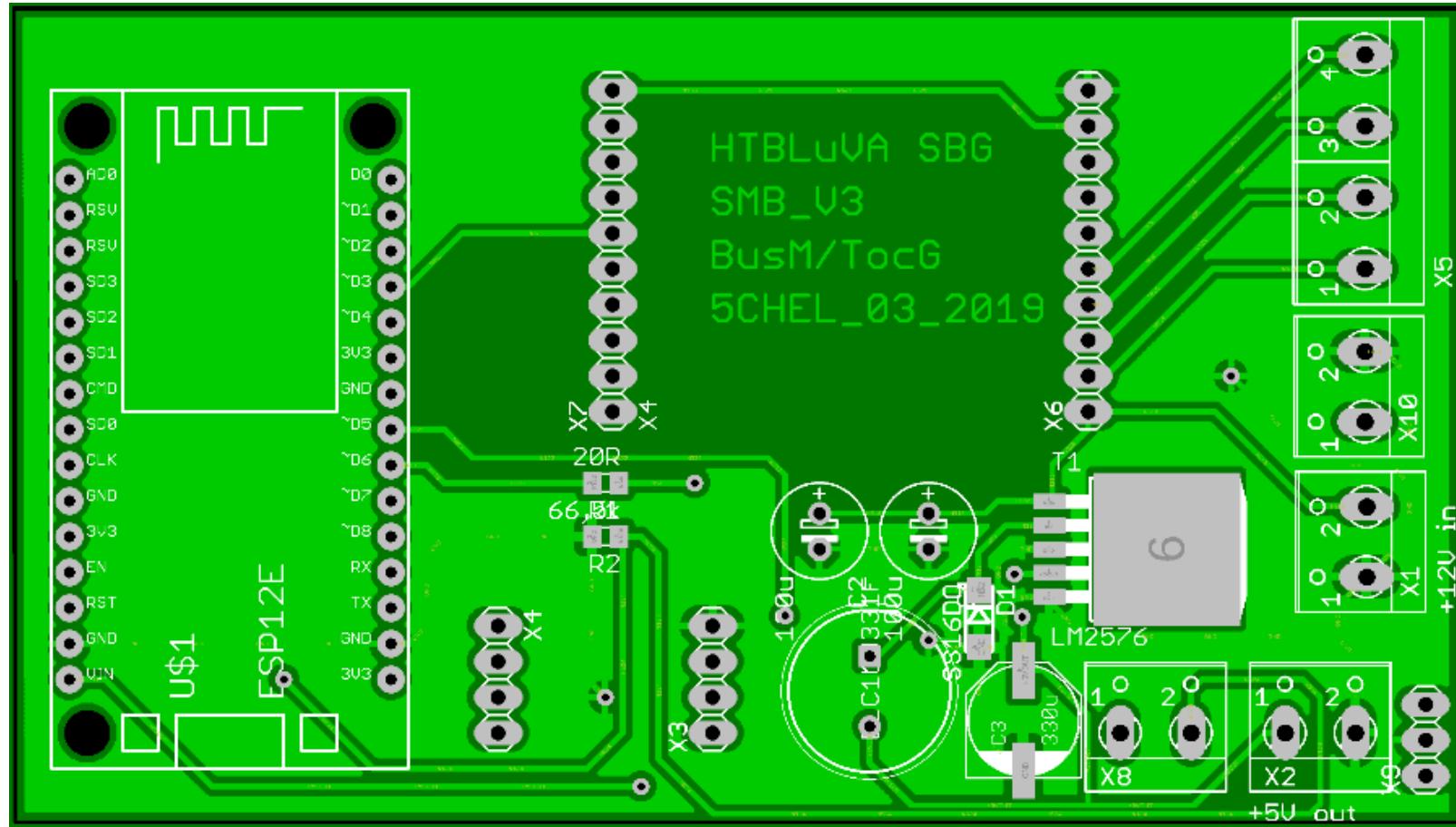


Sym.	Nr.	Durchm.	Anzahl	Durchk.
↑	1	0.5	9	Ja
↖	2	0.8	32	Ja
Ø	3	0.8	4	Ja
Ø	4	1.0	31	Ja
Ø	5	1.2	12	Ja
Ø	6	3.2	3	Nein

Sachnummer, Dateiname: smb_bohrungen.docx	Name: Tockner Gregor	Toleranz: keine	Werkstoff:				
 HTBLuVA Salzburg Elektronik	ID-Nr.: 5CHEL	Gez.: TocG	Geprüft: TocG	Betreuer: SreS	Dokumentart: Bohrplan	Freigabe:	
	Benennung: SMB				Version: 1.0	Revision:	Dokumentstatus:
	Maßstab:	Spr.:	Datum:	Blatt:	Blätter:		
	DE	27.03.2019	1	1			



Sachnummer, Dateiname: Smb_pcb_bot.docx	Name: Tockner Gregor				Toleranz: keine	Werkstoff:
 HTBLuVA Salzburg	ID-Nr.: 5CHEL	Gez.: TocG	Geprüft: TocG	Betreuer: SreS	Dokumentart: PCB_bot	Freigabe:
Benennung: SMB	Version: 1.0		Revision:	Dokumentstatus:		
	Maßstab:	Spr.:	Datum:	Blatt:	Blätter:	
		DE	27.03.2019	1	1	



Sachnummer, Dateiname: Smb_pcb_top.docx	Name: Tockner Gregor	Toleranz: keine	Werkstoff:
 HTBLuVA Salzburg	ID-Nr.: 5CHEL	Gez.: TocG	Geprüft: TocG
	Betreuer: SreS	Dokumentart: PCB_top	Freigabe:
Benennung: SMB	Version: 1.0	Revision:	Dokumentstatus:
	Maßstab: DE	Spr.:	Datum: 27.03.2019
		Blatt:	1
		Blätter	1