HTBLuVA St. Pölten

Höhere Lehranstalt für Maschinenbau
Ausbildungsschwerpunkt Automatisierungstechnik

DIPLOMARBEIT

Gesamtprojekt

Berechnung, Konstruktion und Fertigung einer Münzprägemaschine

Berechnung u. Konstruktion
Hochecker Fabian 5BHMBA Betreuer: Dipl.-Ing. Günter Schmatz
Günter Amstätter

Walzen u. Motive
Grafeneder Gabriel 5BHMBA Betreuer: Dipl.-Ing. Günter Schmatz
Günter Amstätter

Sensorik
Schmid Patrick 5BHMBA Betreuer: Dipl.-Ing. Günter Schmatz
Günter Amstätter

Programm
Schüler Florian 5BHMBA Betreuer: Dipl.-Ing. Günter Schmatz
Günter Amstätter

ausgeführt im Schuljahr 2021/22

Eidesstattliche Erklärung

Ich erkläre an Eides statt, dass ich die vorliegende Diplomarbeit selbstständig und ohne fremde Hilfe verfasst, andere als die angegebenen Quellen und Hilfsmittel nicht benutzt und die den benutzten Quellen wörtlich und inhaltlich entnommenen Stellen als solche erkenntlich gemacht habe.

St. Pölten, am 31.03.2022

Gabriel Grafeneder

Fabian Hochecker

Patrick Schmid

Florian Schüler

Vorwort

Unsere Diplomarbeit entwickelte sich aus dem internationalen Schülerprojekt der Organisation „Erasmus+“. Mit der Chance an diesem teilzunehmen, wurde uns die Möglichkeit geboten, unser Wissen zu erweitern und mit Schülern aus anderen Ländern zusammen zu arbeiten, sowie internationale Freundschaften zu schließen. Obwohl uns dies des Öfteren vor Herausforderungen stellte, war es eine Erfahrung die wir in Zukunft nicht missen wollen. An dieser Stelle gilt unser Dank der HTBLuVA St.Pölten, insbesonders unseren Fachlehrer Herrn Günter Amstätter, welcher die Teilnahme an diesem Projekt erst ermöglichte. Aufgrund seines tatkräftigen Vorantreibens konnte die Arbeit nicht nur auf Papier, sondern auch in der Praxis realisiert werden. Hierbei stand er uns vor allem mit seiner langjährigen Erfahrung im Bereich des Werkzeugbaus zur Seite. Des Weiteren möchten wir uns bei unserem hauptverantwortlichen Betreuer Herrn Dipl. -Ing. Günter Schmatz bedanken. Dieser hat uns bei jeglichen Herausforderungen unterstützt und uns bei der technischen Umsetzung beraten.

Zusammenfassung

Im Zuge des Erasmus+ - Projekts wird eine vollautomatisierte Münzprägemaschine berechnet und konstruiert. Es sollen 5 Cent-Münzen mittels zwei Prägewalzen beidseitig geprägt werden. Auf der Rückseite der Münze ist immer das gleiche Motiv zu erkennen. Auf der Vorderseite kann der Nutzer aus 6 verschiedenen Motiven wählen. Dieser Vorgang erfolgt dabei über ein Touch-Display. Durch eine Münzprüfeinheit wird sichergestellt, dass nur 5 Cent-Stücke in die Maschine gelangen. Eine Steuerung soll die Münze zeitgerecht zu den Walzen geführt werden, um das gewünschte Motiv zu erhalten.

Um das erforderliche Drehmoment zu erreichen, wird ein Antriebssystem berechnet und konstruiert. Weiters werden sechs unterschiedliche Motive, welche zur Auswahl stehen, auf die Walzen übertragen. Dabei wird das Motiv der Rückseite der Münzen einheitlich gehalten. Dazu abgestimmt wird eine Steuerung per Raspberry Pi und Arduino entworfen, welche die Münze mittels eines Schrittmotors zeitgerecht zu den Walzen führt. Diese verfügt über eine ansprechende Benutzeroberfläche, um das Bedienen der Münzprägemaschine so einfach wie möglich zu gestalten.

Die Berechnung sowie die Konstruktion als auch die Programmierung aller Komponenten ist abgeschlossen. Der Großteil, der mechanischen und elektronischen Teile, ist bereits vollständig gefertigt und bereit für den Zusammenbau. Dieser wird in den folgenden Wochen fertiggestellt. Die Auswahl der verschiedenen Motive, sowie die Steuerung des Motors und der Münzzuführung funktionieren bereits.

Abstract

In the course of the Erasmus+ project, a fully automated coin pressing machine is being calculated and constructed. Five-cent coins are to be minted on both sides using two embossing rollers. The reverse side of the coin will always show the same motif. On the front side, the user can choose from six different motifs. This process is carried out via a touch display. A coin verify unit ensures that only five-cent coins enter the machine. A control system is used to guide the coin to the rollers in time to obtain the desired motif.

To achieve the required torque, a drive system is calculated and designed. Furthermore, six different motifs, which can be selected, are transferred to the rollers. The motif on the reverse side of the coins is kept uniform. In addition, a control system is designed using Raspberry Pi and Arduino, which guides the coins to the rollers in time using a stepper motor. This has an appealing user interface to make operating the coin minting machine as simple as possible.

The calculation as well as the construction and programming of all components has been completed. The majority of the mechanical and electronic parts are already completely manufactured and ready for assembly. This will be completed in the following weeks. The selection of the different motifs as well as the control of the motor and the coin feed are already working.

Inhaltsverzeichnis

1. Einführung 12

1.1. Ausgangssituation 12

1.2. Zielsetzung 13

1.3. Teilnahme am Erasmus+ Projekt „PPM4EU“ 13

1.4. Aufgabenstellung 15

1.4.1. Auslegung der Walzen 15
1.4.2. Auslegung des Getriebes 15
1.4.3. Entwurf einer Münzzuführung 15
1.4.4. Auswahl der Sensorik 15
1.4.5. Programmierung 15

2. Projektplanung 16

2.1. Projektstruktur 16

2.2. Terminplan 16

2.3. Individuelle Aufgabenstellungen 17

2.3.1. Aufgabenstellung von Kandidat 1 (Hochecker Fabian) 17
2.3.2. Aufgabenstellung von Kandidat 2 (Grafeneder Gabriel) 17
2.3.3. Aufgabenstellung von Kandidat 3 (Schmid Patrick) 18
2.3.4. Aufgabenstellung von Kandidat 4 (Schüler Florian) 18

3. Auslegung der Prägewalzen 19

3.1. Ermittlung der Festigkeit einer 5-Cent Münze 19

3.2. Konzepte für die Ausführung der Walzen 25

3.2.1. Konzept 1: kleiner Durchmesser und geringe Breite 25
3.2.2. Konzept 2: kleiner Durchmesser und große Breite 26
3.2.3. Konzept 3: großer Durchmesser und geringe Breite 27
3.2.4. Entscheidungsfindung 28

3.3. Berechnung von Walzendurchmesser und Drehmoment 30

3.3.1. Berechnung des Durchmessers 30
3.3.2. Berechnung des erforderlichen Walzmoments 33

4. Auslegung und Konstruktion des Getriebes 38

4.1. Konzepte Getriebe 38

4.1.1. Konzept 1: Getriebe für Walzen mit kleinem Durchmesser 38

4.1.2. Konzept 2: Getriebe für Walzen mit großem Durchmesser 39

Konzepte Antrieb 41

4.1.3. Konzept 1: Getriebemotor 41

4.1.4. Konzept 2: Gleichstrommotor 42

4.1.5. Konzept 3: Antrieb eines Akkuschraubers 43

4.2. Konstruktionsstruktur 44

4.2.1. Einfügen eines neuen Teils 45

4.2.2. Funktion der Exceltabelle 46

4.2.3. Filtern 46

4.2.5. Verwaltung 47

4.3. Konstruktion Getriebe 48

4.3.1. Allgemein 48

4.3.1.1. Verwendete Programme 48

4.3.1.2. Steuerskizze 48

4.3.2. Hauptzusammenbau 50

4.3.3. Theoretische Übersetzung 51

4.3.4. Übersicht der Getriebestufen 52

4.3.5. Antriebsstrang 54

4.3.5.1. Unterbaugruppe – Antrieb ZSB 55

4.3.5.1.1. Antrieb 55

4.3.5.1.2. Gehäuse 57

4.3.5.1.3. Kupplung 59

4.3.5.1.4. Zusammenbau 61

4.3.5.2. Antriebswelle – Welle1 ZSB 62

4.3.5.2.1. Welle1 64

4.3.5.2.2. Zahnrad 66

4.3.5.2.3. Lager 67

4.3.5.2.4. Passfeder 68

4.3.6. Unterbaugruppe – Welle 2 ZSB 69

4.3.6.1. Ritzelwelle 71

4.3.6.2. Lager und Passfeder 72

4.3.7. Unterbaugruppe – Achse 3 ZSB ... 73
4.3.7.1. Achse 3 ... 75
4.3.7.2. Lagerung ... 77
4.3.7.3. Walze 3 ... 78
4.3.7.4. Motive der Prägewalzen ... 81
4.3.7.5. Zahnkranz ... 89
4.3.8. Unterbaugruppe – Achse 4 ZSB ... 91
4.3.9. Hinterer Lagerblock ... 94
4.3.10. Vorderer Lagerblock ... 95
4.4. Zusammenbaureihenfolge ... 97
4.5. Gehäuse ... 103
4.6. Technische Zeichnungen ... 105
5. Weg der Münze ... 106
5.1. Position des Münzeinwurfs ... 106
5.2. Münzführung ... 107
5.3. Einschubeinheit ... 111
5.3.1. Konzepte ... 111
5.3.1.1. Fallend ... 111
5.3.1.2. Schiebend ... 111
5.3.1.3. Schwenkend ... 111
5.3.2. Schwenkeinheit ... 112
5.3.2.1. Schwenkarm und Schieber ... 112
5.3.2.2. Leitschiene ... 113
5.3.2.3. Schrittmotor ... 114
5.4. Abführen der Münze ... 114
6. Steuerung ... 115
6.1. Auswahl der Steuerung ... 115
6.1.1. SPS ... 115
6.1.2. Arduino ... 116
6.1.3. Raspberry Pi ... 117
6.1.4. Lösungsansatz ... 118

6.2. Kommunikation der Steuerungen 119

7. Sensoren Aktoren 120

7.1.1.1. Münzerkennung 120

7.1.2. Mechanisch 121

7.1.3. Elektronisch 122

7.1.4. CH-926 123

7.2. Detektion der Walzposition 124

7.2.1. Konzepte 124

7.2.1.1. Optischer inkrementaler Drehgeber 124

7.2.1.2. Optischer Sensor 126

7.2.1.3. Zahnflankenerkennung 127

7.2.1.4. Schraubenerkennung 128

7.2.2. Lösungsansatz 129

7.3. Schrittmotor 129

7.3.1. Auswahl 129

7.3.2. Treiberplatine 130

7.4. Motorsteuerung 131

7.4.1. Relais 131

7.4.2. Motortreiberplatine 131

7.5. Schaltung 133

7.5.1. Galvanische Trennung 133

7.5.2. Versuche 134

7.5.3. Schaltplan 135

7.5.4. Schaltplatine 137

8. Ablaufsteuerung 140

8.1. C++ 140

8.1.1. Funktion: 140

8.1.2. Switch 140

8.2. Setup des Arduino Programmes 141

8.2.1. Bibliotheken 141

8.2.2. Pin-Belegung festlegen 141

8.2.3. Variablen 142

8.2.4. Ein- und Ausgänge Setzen ... 143
8.2.5. RFID-Leser Initialisieren ... 144
8.2.6. Stepper Referenzieren ... 144

8.3. Lesen und schreiben von Signalen ... 145

8.3.1. Positions-Sensor Walzen ... 145
8.3.2. Referenz-Sensor Walzen ... 146
8.3.3. Endlage Stepper ... 146
8.3.4. Münzprüfeinheit ... 147
8.3.5. RFID ... 148
8.3.6. Serial ... 149

8.4. Ablauf des Arduino-Programmes ... 151

8.4.1. Schritt 10 ... 151
8.4.2. Schritt 20 ... 151
8.4.3. Schritt 30 ... 152
8.4.4. Schritt 40 ... 152
8.4.5. Schritt 50: ... 153
8.4.6. Schritt 60 ... 153
8.4.7. Schritt 70 ... 154

9. Interface ... 155

9.1. Python ... 155

9.1.1. Klassen ... 155
9.1.2. Funktion ... 155
9.1.3. Vererbung ... 155

9.2. Auswahl der GUI Bibliothek ... 156

9.2.1. Tkinter ... 156
9.2.2. Kivy ... 156
9.2.3. Lösungsansatz ... 157

9.3. .kV ... 157

9.3.1. Layouts ... 157
9.3.1.1. FloatLayout ... 158
9.3.1.2. RelativeLayout ... 158
9.3.1.3. GridLayout ... 158
9.3.1.4. AnchorLayout ... 158

9.3.1.5. BoxLayout ... 158
9.3.1.6. StackLayout ... 158
9.3.1.7. PageLayout ... 159
9.3.2. Widgets ... 159
9.3.2.1. Label ... 159
9.3.2.2. Button ... 159
9.3.2.3. Toggle Button ... 160
9.3.2.4. Switch ... 160
9.3.2.5. Slider ... 160

9.4. Setup des Interfaces ... 161
9.4.1. Erzeugung des Interfaces ... 161
9.4.2. Bibliotheken ... 161
9.4.3. Serielle Kommunikation ... 162
9.4.4. Globale Variablen ... 162
9.4.5. Fenster erstellen ... 163
9.4.6. Initialisierung von Objekten ... 163

9.5. Ablauf des Interfaces ... 164
9.5.1. Wartebildschirm ... 165
9.5.2. Münze eingeworfen ... 166
9.5.3. Designauswahl ... 166
9.5.4. Popup ... 168
9.5.5. Datenübertragung ... 169
9.5.6. Prozessbildschirm ... 170
9.5.7. Maschine fertig ... 171

9.6. Einstellungen ... 171
9.6.1. Vollbild ... 173
9.6.2. Animationen ... 174
9.6.3. Geschwindigkeiten und Nachlaufzeit ... 175
9.6.4. Manuelle Motorsteuerung ... 176
9.6.5. Zeiteingabefeld ... 176
9.6.5.1. Schreiben ... 176
9.6.5.2. Löschen ... 177
9.6.5.3. Fokus ... 177
9.6.5.4. Verhindern eines Bereichsfehlers ... 178
9.6.5.5. Nullen löschen ... 178

9.6.5.6. Enter 179
9.6.5.7. Speichern 179
9.6.5.8. Plus / Minus 1 180

10. Fertigung 181

1. Einführung

1.1. Ausgangssituation

Im Schuljahr 2019/2020 nahmen zwei Schüler des Projektteams gemeinsam mit anderen Klassenkollegen an einem internationalem Schulprojekt, gesponsert von Erasmus+ mit dem Titel „PPM4EU – Penny Press Machine for Europe“, teil. Ebenfalls Teil des Projekts waren Schülerinnen und Schüler aus Deutschland, Luxemburg, Spanien, Norwegen und der Schweiz.

Im Zuge dieser Zusammenarbeit sollte eine funktionierende Münzprägemaschine entworfen, berechnet, konstruiert und anschließend gefertigt werden. Zu diesem Zeitpunkt war eine gemeinsame Planung und Dokumentation in der Projektsprache Englisch, sowie eine spätere Fertigung vorgesehen.

Jede der involvierten Bildungsanstalten sollte nach Abschluss dieses Projekts über eine eigene „Penny Press“ verfügen. Während der beiden Meetings in Luxemburg und Spanien wurden Konzepte für das Getriebe, das Walzwerk und für die Steuerung gesammelt. Aufgrund der COVID 19 – Pandemie wurden im März 2020 schulische Auslandsreisen untersagt und infolgedessen das Projekt „PPM4EU“ bis auf weiteres stillgelegt.

Das Projekt wurde daraufhin im Abschlussjahr 2021/2022 das Thema dieser Diplomarbeit.

1.2. Zielsetzung

Die Maschine soll mithilfe eines Walzwerkes auf beide Seiten der Münze ein Motiv prägen. Jedes der sechs Motive symbolisiert eine Partnerschule im Projekt „PPM4EU“. Die Auswahl der Motive, soll über ein integriertes Touch Display getätigt werden.

1.3. Teilnahme am Erasmus+ Projekt „PPM4EU“

Nachdem im Herbst des Jahres 2021 die Auflagen während der Corona Pandemie gelockert wurden, konnte das internationale Schülerprojekt „PPM4EU“ wieder aufgenommen und weitergeführt werden. Da sich das Projektteam im Rahmen der Diplomarbeit schon ausführlich mit dem Thema „Penny Press“ beschäftigt hatte, wurde entschieden, nicht das ursprüngliche Team, sondern die Diplomarbeitsgruppe zum nächsten Meeting zu schicken. Dieses Treffen fand im November 2021 in der norwegischen Stadt Trondheim statt. Viele Schüler aus den vorherigen Treffen haben ihre Ausbildung bereits abgeschlossen und konnten somit nicht mehr am Projekt teilnehmen. Dies führte zu einer großen Herausforderung, da die meisten neuen Teilnehmer projektfremd waren und somit das technische „Know-How“ fehlte. Die erste Aufgabe dieses Meetings war es, die bereits getane Arbeit der verschiedenen Delegationen vorzustellen. Daraus ergab sich, dass seit dem vorherigen Treffen in Barcelona, die technische Entwicklung größtenteils im österreichischen Team (Projektteam dieser Diplomarbeit), vorangetrieben wurde. Das Projekt wird von der deutschen Partnerschule organisiert. Somit wird die Fertigstellungsfrist auch von der deutschen nationalen Agentur festgelegt. Dieser Termin liegt im August des Jahres 2022. Um diese Frist einzuhalten, wurde die österreichische Delegation zum technischen Leiterteam ernannt. Aufgabe der Diplomarbeitsgruppe ist es somit, die anderen Länder mit Berechnungen, Konstruktionen und Fertigungszeichnungen zu versorgen. Nach dem Treffen in Norwegen wurde mittels Microsoft-Teams miteinander kommuniziert. Diese Plattform wurde auch dazu verwendet, die Zeichnungen zu publizieren und offene Fragen zu beantworten. Um den gewünschten Arbeitsfortschritt zu gewährleisten und Probleme zu klären, wurden Online-Meetings abgehalten.

Abbildung 1: Projektteam PPM4EU

(https://ppm4eu.cnc-network.eu/activities/meetings/meeting-no/58-third-meeting-in-trondheim)

1.4. Aufgabenstellung

1.4.1. Auslegung der Walzen

- Ermittlung der Festigkeit einer 5 Cent-Münze
- Berechnung der Kräfte im Walzspalt und des erforderlichen Drehmoments am Walzwerk
- Entwurf der Prägemotive

1.4.2. Auslegung des Getriebes

- Berechnung und Konstruktion eines Getriebes zur Übertragung des erforderlichen Drehmoments
- Auswahl eines passenden Motors
- Entwurf eines Gehäuses

1.4.3. Entwurf einer Münzzuführung

- Konstruktion einer Münzführung vom Einwurf bis zwischen die Walzen

1.4.4. Auswahl der Sensorik

- Sensoren zur Überwachung der Walzenposition
- Sensoren zum Referenzieren der Motoren

1.4.5. Programmierung

- Auswertung der Sensorik
- Programmierung einer Ablaufsteuerung
- Erstellen eines Userinterface

2. Projektplanung

2.1. Projektstruktur

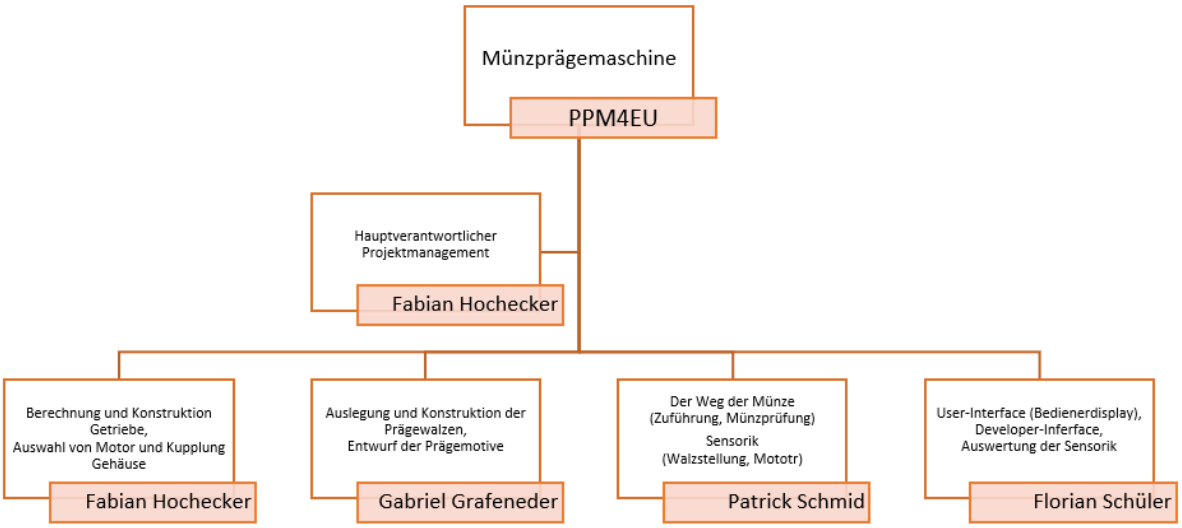


Abbildung 2: Organigramm

2.2. Terminplan

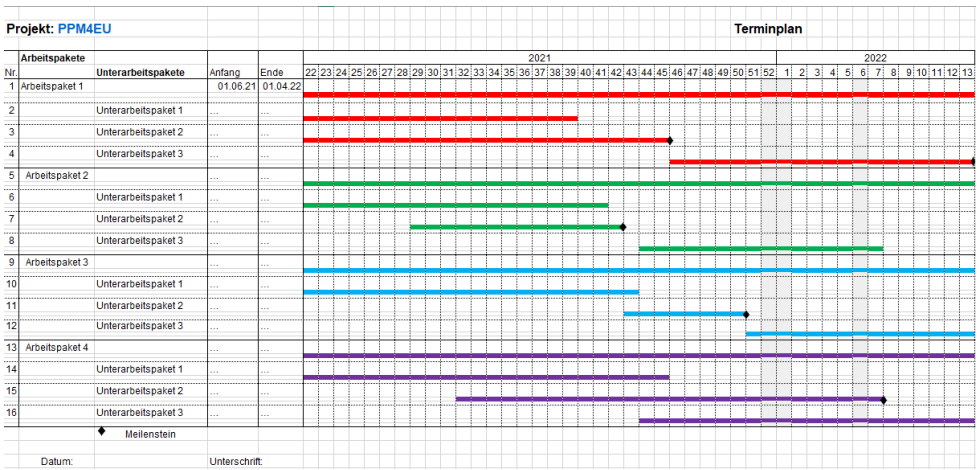


Abbildung 2: Terminplan

2.3. Individuelle Aufgabenstellungen

2.3.1. Aufgabenstellung von Kandidat 1 (Hochecker Fabian)

Das Hauptaugenmerk der Aufgabenstellung von Kandidat 1 liegt in der Berechnung und Konstruktion eines Getriebes. Dieses soll das passende Abtriebsdrehmoment auf den Prägewalzen gewährleisten. Dazu muss ebenfalls ein passender Motor und eine Kupplung dimensioniert bzw. ausgesucht werden. Neben den Getriebebauteilen soll auch eine Einhausung für die fertige Münzprägemaschine konstruiert werden.
Nach einer Berechnung mit dem System „Mathcad“ ist ein 3D – Modell (SolidWorks) der mechanischen Komponenten der Münzprägemaschine zu erstellen. Danach sind die einzelnen Bauteile, in 2D – Fertigungszeichnungen abzuleiten.

2.3.2. Aufgabenstellung von Kandidat 2 (Grafeneder Gabriel)

Der Kandidat 2 ist bei diesem Projekt für die Auslegung sowie konstruktive Entwicklung der Prägewalzen und deren Motive zuständig. Für die erfolgreiche Auslegung der Prägewalzen muss zuerst die Festigkeit der Münze bestimmt werden. Die Auslegung des Walzenradius sowie die Bestimmung der auftretenden Walzkräfte erfolgt mit dem Programm „Mathcad“. Mit den aus der Berechnung gewonnen Werten soll ein passender Werkstoff für die Walzen ausgewählt werden. Weiters dienen diese Ergebnisse der Auslegung des Getriebes, welche Kandidat 1 durchführt.
Es sollen 6 unterschiedliche Motive zur Auswahl stehen. Anschließend werden die Motive im CAD Programm „SolidWorks“ auf den Walzen dargestellt.

2.3.3. Aufgabenstellung von Kandidat 3 (Schmid Patrick)

Die Hauptaufgabe des Kandidat 3 liegt in der Konstruktion (SolidWorks) in einer optisch ansprechenden Münzzuführung. Diese soll die Münze vom Einwurf bis zur Prägung und aus der Maschine transportieren. Hierzu muss eine Einheit konstruiert werden, die die Münze zum richtigen Zeitpunkt zwischen die Walzen führt. Weiters sollen die für die Maschine benötigten Sensoren und Aktoren gewählt werden und eine Schaltung für das Verbinden der Sensoren und Aktoren mit der Steuerung erstellt werden.

2.3.4. Aufgabenstellung von Kandidat 4 (Schüler Florian)

Aufgabe von Kandidat 4 ist es eine Steuerungseinheit auszuwählen, welche es ermöglicht, die Münzprägemaschine möglichst zuverlässig sowie kostengünstig zu steuern. Weiters soll ein grafisches Interface sowie eine Ablaufsteuerung programmiert werden. Das Interface soll es ermöglichen, die verschiedenen Motive per Touch auszuwählen und in Auftrag zu geben. Außerdem soll das Interface über eine Developer Seite verfügen, welche das Konfigurieren und Managen der Maschine ermöglicht. Die Ablaufsteuerung soll die Informationen des Interfaces auswerten und anhand dessen die Maschine richtig steuern.

3. Auslegung der Prägewalzen

3.1. Ermittlung der Festigkeit einer 5-Cent Münze

Um die mechanischen Eigenschaften, des Münzmaterials zu bestimmen, werden diverse Recherchen im Internet durchgeführt. Aus diesen Ermittlungen kann nur der bereits bekannte Hinweis gefasst werden, dass die Münzen aus einem mit Kupfer beschichteten Stahlkern bestehen. Dies kann man auch dadurch beweisen, dass Kupfer allein nicht magnetisch ist, die Münzen jedoch magnetisch sind. Weitere brauchbare Schlüsse kann man jedoch aus dem World Wide Web (WWW) nicht ziehen. Grund für das Nichtauffinden des Münzenwerkstoffes ist der Fälschungsschutz, damit kein Falschgeld in Umlauf gebracht werden kann.

Da die HTBLuVA St. Pölten eine ausreichende Ausstattung im Bereich der Werkstoffprüfgeräte besitzt, wird der Entschluss gefasst, einen Druckversuch zur Ermittlung der Streckgrenze der Münzen durchzuführen. Bei diesem Versuch wird keineswegs die genaue Bezeichnung des Werkstoffes ermittelt, sondern lediglich die Kraft, welche zur plastischen Umformung benötigt wird.

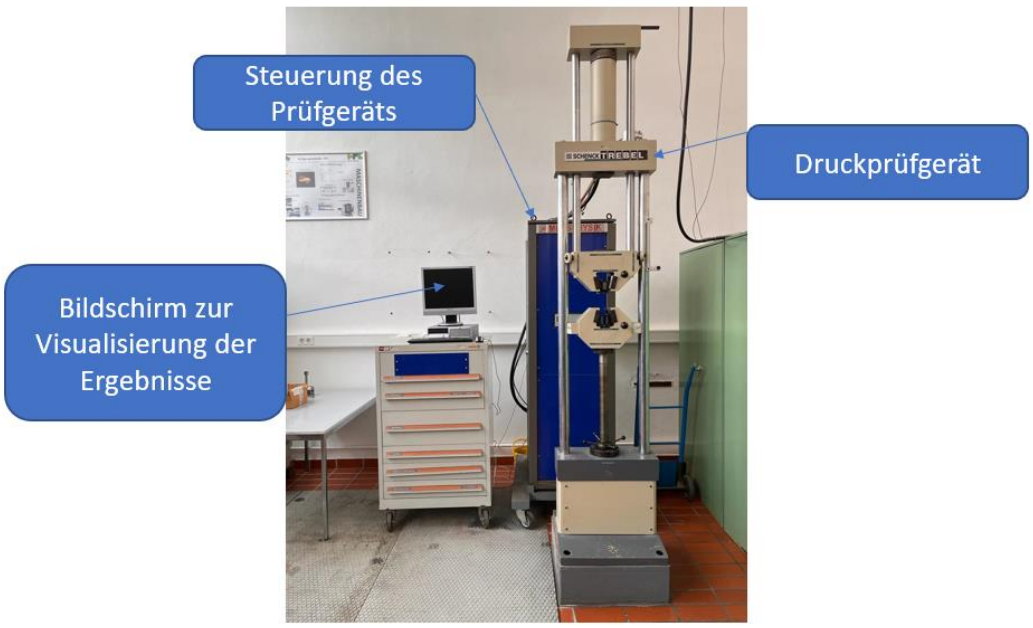


Abbildung 3: Druckprüfgerät

Ein Druckversuch funktioniert folgendermaßen:
Das zu prüfende Werkstück wird zwischen die beiden Backen des Prüfgeräts gelegt. Anschließend wird das zu prüfende Werkstück vom Prüfgerät zwischen den Backen verformt. Von einem Computer, welcher das Prüfgerät steuert, wird eine Kennlinie aufgezeichnet, welche im Nachhinein Auskunft über die maximale Kraft gibt.
Um die Streckgrenze des Werkstoffs mittels Druckprüfung zu ermitteln, ist die Dicke einer Münze allein nicht ausreichend, daher wurde die Entscheidung getroffen worden, mehrere Münzen übereinander zu stapeln. Um ein wegrutschen der Münzen, während des Versuches zu verhindern, wird eine Aluminium-Hülse genutzt, welche ebenfalls mit zerdrückt wird. Die Aluminiumhülsen sind in diversen Ausführungen gefertigt worden um zu untersuchen, welche sich am besten für den Versuch eigenen würden und welche zu keiner Verfälschung des Ergebnisses führen würden. Bei der Ausführung der Hülsen mit einem dicken Ring an der Unterseite stellt sich möglicherweise das Problem, dass das Material der Münzen während des Prüfvorgangs nur wenig Platz zum Fließen hat (siehe Abbildung 4) und somit das Ergebnis verfälscht wird. Bei der Ausführung als normale zylindrische Hülse ist es jedoch kein Problem, da das Aluminium voraussichtlich um einiges weicher ist als der Stahlkern der Münze. Es sollte deshalb nicht in das Ergebnis des Versuches mit einfließen.

Abbildung 4: Hülse, dicker Ring

Beim ersten Anlauf den Versuch durchzuführen wird kein Ergebnis erzielt, da das Prüfgerät nicht auf 0 zusammengefahren werden kann, da der Endlagenschalter auslöst, bevor die Münzen erwischt werden.
Vor dem zweiten Anlauf wird eine Distanzbacke aus einem harten Werkstoff (C45) mit einer Länge von 280 od. 300 mm gefertigt, welche weiters auch einen Zentrieransatz hat, damit diese beim Druckversuch nicht verrutscht. Als Vorbereitung wird die

angefertigte Distanzbacke mittels des Zentrieransatzes positioniert, die Aluminiumhülse mit Münzen gefüllt und mittig auf die Distanzbacke positioniert (siehe Abbildung 5).

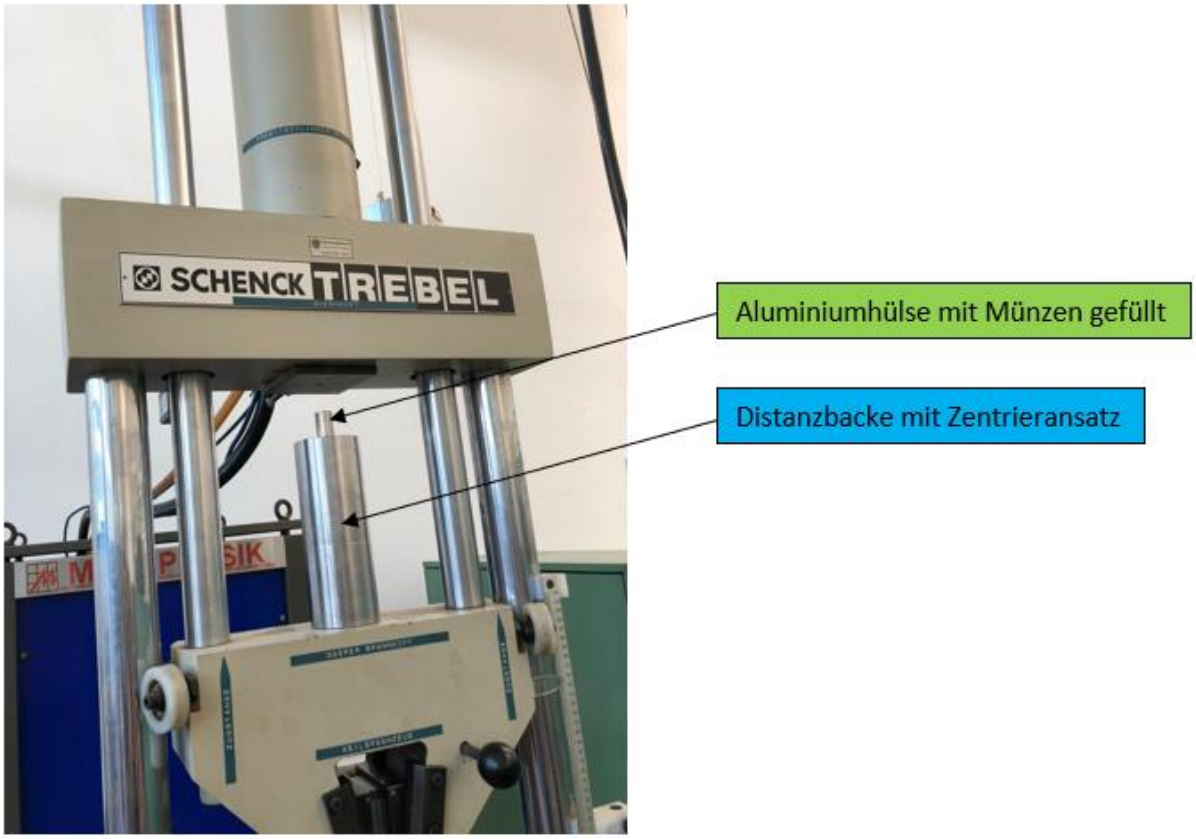
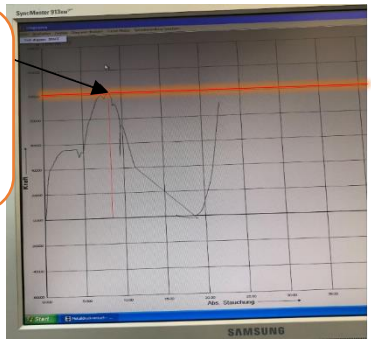


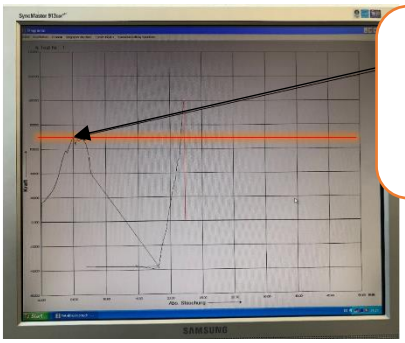
Abbildung 5: Vorbereitung Druckversuch

Der Druckversuch kann nun dank der gedrehten Unterlegbacke mit Zentrieransatz durchgeführt werden. Während des Versuches stellt sich jedoch heraus, dass die dafür angefertigten Hülsen brechen und sich nicht fließend verformen. Dadurch können die Münzen während des Zusammendrückens auf die Seite rutschen. Letztendlich ist der Münzenstapel unter der Krafteinwirkung weggerutscht und die Maschine ist ruckartig mit den Backen zusammengeschlagen. Da dies die Maschine beschädigen könnte, werden nur drei Versuche durchgeführt.

Aus den bei den Versuchen aufgezeichneten Diagrammen kann man nun die maximal wirkende Kraft auslesen (siehe Abbildung 6). Die maximale Kraft, welche ausgeübt wurde, ist knapp an der Grenze des Prüfgerätes. Da die Münzen aus dem Prüfgerät gerutscht sind, kann man jedoch die Versuche nicht als zuverlässig werten.

Größte gemessene Kraft 100 000N





Größte gemessene Kraft 85 000N

Abbildung 6: Diagram Druckversuch

Da die Druckprüfung gewissermaßen fehlgeschlagen ist, wird beschlossen, eine Härteprüfung nach Vickers durchzuführen. Das notwendige Prüfgerät besitzt ebenfalls die HTBLuVA ST. Pölten und stellt es zur Verfügung.

Die Härteprüfung nach Vickers funktioniert folgendermaßen: Das Härteprüfverfahren nach Vickers unterliegt den Standards der Norm ISO 6507 und zählt sich zu den statischen Prüfverfahren. Dabei wird ein pyramidenförmiger Prüfkörper, welcher aus einem hochfesten Werkstoff besteht, in das zu prüfende Werkstück hineingedrückt. Aus der aufzubringenden Kraft und der Diagonale des bleibenden Eindruckes im Werkstück wird danach vom Prüfgerät selbstständig die Härte berechnet.

Damit das Ergebnis nicht verfälscht wird, muss das zu prüfende Werkstück eine bestimmte Dicke haben. Um diesem Effekt vorzubeugen, verwenden die Diplomanden zum Prüfen zwei aufeinander- gelegte Münzen.

Um eine mögliche Streuung der ermittelten Werte geringer zu halten, werden zwei Prüfungen durchgeführt. Der erste Prüfvorgang hat eine Härte nach Vickers (HV) von 106 ergeben, was ca. einer Streckgrenze von 338 N/mm^2 entspricht. Der zweite Prüfvorgang hat eine Härte nach Vickers (HV) von 108 ergeben, was ca. einer Streckgrenze von 342 N/mm^2 entspricht.

Diese beiden Werte sind vergleichbar mit den Werten der Druckprüfung.

Bei dem Erasmus+ Projekt nehmen sowohl die Schweiz als auch Norwegen teil. Diese beiden Länder haben nicht den Euro als Währung, sondern Franken und Kronen. Aus diesem Grund wird auch eine Härteprüfung mit diesen beiden Münzen durchgeführt. Daraus ergab sich das Ergebnis von 114HV, was einer Streckgrenze von ca. 368N/mm^2 erweist. Da der Werkstoff der beiden Münzen ident ist, gilt auch das

Ergebnis der Härteprüfung für beide Münzen. Auf Grund dessen, dass an diesem Projekt mehr Länder mit dem Euro als Währung teilnehmen, wurde die Maschine auf die Werte der 5 Cent Münze ausgelegt, da jedoch eine gewisse Sicherheit mit berücksichtigt wird, ist auch der Schweizerfranken sowie die norwegische Krone kein Problem für die Maschine.

Zusätzlich zu den beiden Prüfverfahren wird auch ein Versuch zur Verformung der Münze durchgeführt, bei welchem zugleich ein Gefühl für die Richtigkeit der beiden Versuche gefunden wird. Bei diesem Versuch werden mehrere 5 Cent Münzen sowie auch Schweizerfranken mittels einer Blechwalze im Stahlleichtbau der HTBLuVA St. Pölten verformt. Dieser Versuch erweist sich als sehr kräftezehrend, da das Betätigen der Kurbel große Anstrengung erforderte. Zudem muss auch das Gestell der Blechwalze gehalten werden, da es nicht fest mit dem Boden verbunden ist und somit leicht wegrutscht.

Abbildung 7: Blechwalze

Abbildung 8: Gewalzte Münzen

Aus diesem Versuch bestätigt sich die Härte der Münzen, da die Münzen erst nach 10-15-maligem Durchkurbeln durch die Walzen, die auf der Abbildung 8 gezeigte Form

erreichen. Auch auf die Dicke der gewalzten Münze wird geachtet, welche sich gegen Ende hin auf ca. 0,8mm befinden sollte. Deshalb wird zwischendurch immer wieder gemessen (siehe Abbildung 9).

Abbildung 9: Zwischenstandmessung Münze

3.2. Konzepte für die Ausführung der Walzen

Der Diplomand Grafeneder arbeitet verschiedene Konzepte zur Ausführung der Prägewalzen aus. Daraus ergaben sich viele verschiedene Möglichkeiten.

3.2.1. Konzept 1: kleiner Durchmesser und geringe Breite

Bei dieser Methode zur Ausführung der Walzen ist die Idee, dass die Motive radial angeordnet werden, also am Umfang verteilt. Es sollen also die 6 auszuwählenden Motive knapp hintereinander auf der Walze angeordnet werden (siehe Abbildung 10). Durch die richtige Steuerung dahinter soll die Münze zum richtigen Zeitpunkt zu der Walze zugeführt werden.

Zwei Vorteile bei diesem Konzept sind, dass die Steuerung sehr simpel gehalten werden kann, da nur eine radiale Bewegung der Zuführung erforderlich ist, und weiters die Kosten gering sind, da nur wenig Material benötigt wird.

Der Nachteil könnte hier sein, dass die sechs Motive am Umfang zu wenig Platz haben, und es somit nicht möglich ist, alle sechs gewünschten Motive aufzutragen.

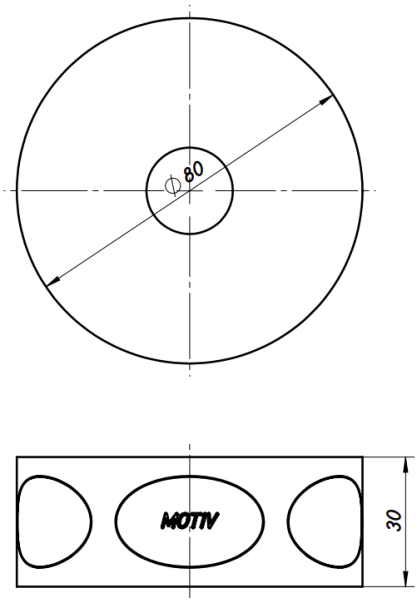


Abbildung 10: Walze_klein_schmal

3.2.2. Konzept 2: kleiner Durchmesser und große Breite

Bei der zweiten Methode sollen die Walzen ebenfalls mit einem kleinen Durchmesser, jedoch mit einer großen Breite ausgeführt werden. Somit können die Motive axial, also in Längsrichtung nebeneinander verteilt werden (siehe Abbildung 11). Diese Ausführung erfordert jedoch einen größeren steuerungstechnischen Aufwand, da die Zuführung der Münzen nicht nur radial verfahrbar und positionierbar sein muss, sondern auch in axialer Richtung positionierbar sein soll. Angesichts dieser Ausführung ist es ein Vorteil, dass die Motive ausreichend Platz haben und es zu keinen Überschneidungen kommen wird. Nachteilig hierbei ist jedoch die höhere Beanspruchung während des Prägevorgangs, denn immer, wenn das mittlere Motiv geprägt wird, entsteht ein hohes Biegemoment auf den beiden Lagerstellen. Zudem bedeutet dies einen größeren Materialaufwand, da die Walzen breiter sind und somit höhere Kosten anfallen.

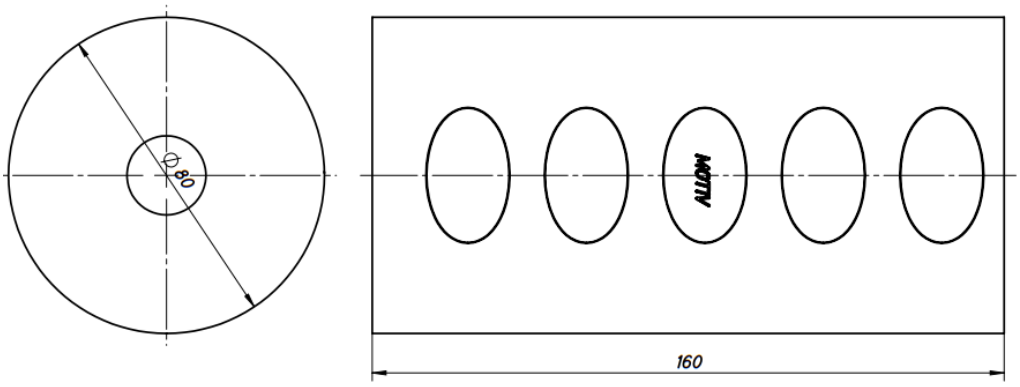


Abbildung 11: Walze_klein_breit

3.2.3. Konzept 3: großer Durchmesser und geringe Breite

Als dritte Methode zur Ausführung der Prägewalzen kommen Walzen mit einem großen Durchmesser und einer geringen Breite in Frage (sieh Abbildung 12). Die Motive können hier wiederum mit ausreichendem Platz radial angeordnet werden. Dies würde ebenfalls einen geringeren steuerungstechnischen Aufwand bedeuten, da wie bereits in Konzept 1 genannt, die Münzzuführung nur radial verschiebbar und positionierbar sein muss. Zusätzlich haben die Motive ausreichend Platz, um sich am Umfang zu verteilen. Nachteilig sind hier wiederum die Kosten, da mehr Material benötigt wird.

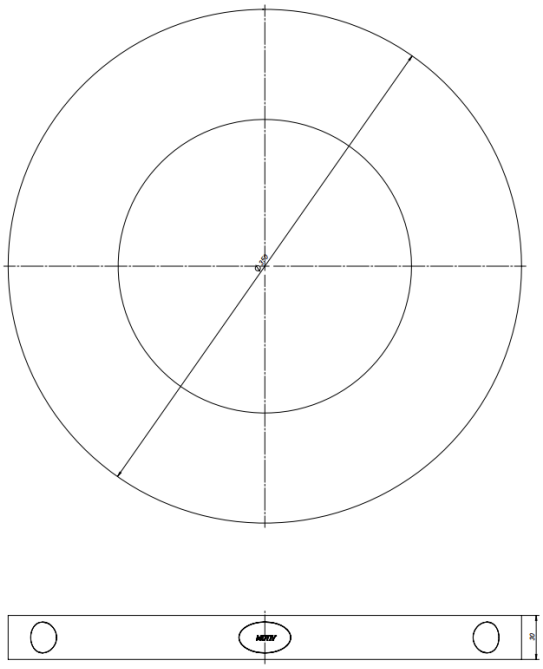


Abbildung 12: Walzen_groß_schmal

3.2.4. Entscheidungsfindung

Nach genauerer Betrachtung einer herkömmlichen, händisch betriebenen, Münzprägemaschine, wie sie beispielsweise im Tierpark Schönbrunn und an vielen weiteren Standorten vorzufinden ist, stellt der Diplomand fest, dass bei diesen Maschinen kleine kompakte Walzen verwendet werden. Diese Ausführung würde dem Konzept 1 dieser Diplomarbeit entsprechen. Bei der Betrachtung einer Münze aus einer herkömmlichen Maschine wird festgestellt, dass auf der Münze eine Art kleine „Nase“ mit eingeprägt ist (siehe Abbildung 13).

Abbildung 13: Beispielmünze

Diese Nase scheint als Mitnehmer, um die Münze beim Vorbeidrehen des Motives auf den Walzen einzuziehen und durchzuziehen, zu dienen (siehe Abbildung 14).

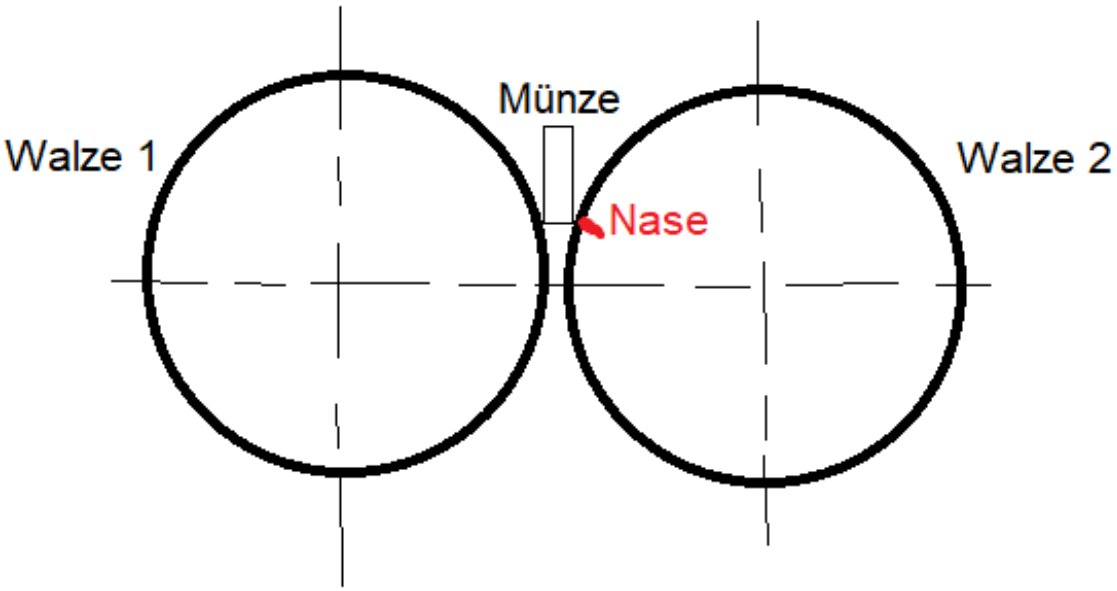


Abbildung 14: Schemaskizze_Nase

Nach einer Besprechung mit den Diplomandenkollegen wurde beschlossen, dass eine Möglichkeit gefunden werden sollte, die „Nase“ auf der Münze zu vermeiden. Wie bekannt ist, ist auch bei einem großen industriellen Walzwerk keine Kerbe nötig, um zum Beispiel das zu walzende Blech einzuziehen.

3.3. Berechnung von Walzendurchmesser und Drehmoment

3.3.1. Berechnung des Durchmessers

Um die „Nase“ zu vermeiden, muss die Münze mittels Reibschluss zwischen die Walzen eingezogen werden, das heißt sobald sie zwischen die Walzen trifft, soll sie ohne Mithilfe der Prägung, also des Motives, eingezogen werden.

Um dies zu gewährleisten, entwickelt der Diplomand Grafeneder ein Berechnungsmodell zur Ermittlung des erforderlichen Walzendurchmessers. Der Großteil der bei diesem Berechnungsmodell angewandten Methoden basiert auf der Dissertation von Christian Overhagen namens "Modelle zum Walzen von Flach- und Vollquerschnitten“ und auf der Datei „Begriffe und Kenngrößen der Umformtechnik von J. Dietrich und H. Tschätsch“.

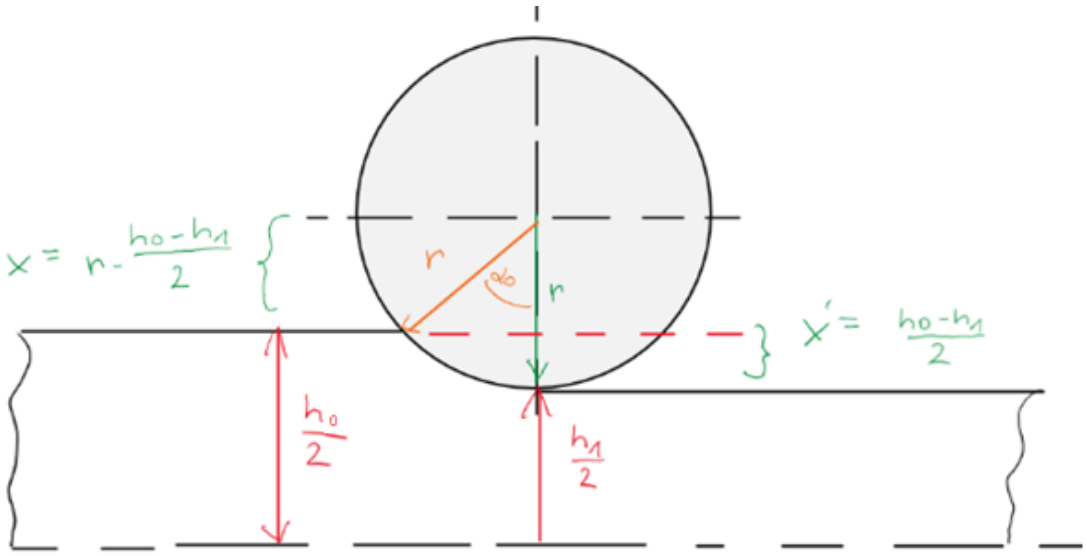


Abbildung 15: Skizze zur Berechnung des Walzenradius

Anhand der hier in Abbildung 15 gezeigten Skizze kann man ein Gleichungssystem, bestehend aus drei Gleichungen (siehe Abbildung 16), aufstellen.

$$\text{I:} \quad x - x' = r$$

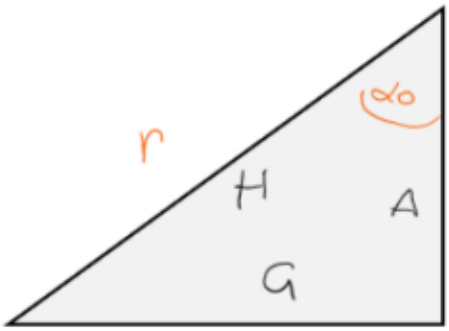
$$\text{II:} \quad x = r \cdot \cos(\alpha_0)$$

$$\text{III:} \quad x' = \frac{h_0 - h_1}{2}$$

Abbildung 16: Gleichungssystem zur Walzenradiusberechnung

Dieses Gleichungssystem wird nach der Variablen „r“ aufgelöst, welche den Radius der Walzen beschreibt. Der Winkel α_0 (Alpha 0) beschreibt den Reibwinkel welcher mit hoher Wahrscheinlichkeit zutrifft.

Bei der Auflösung dieses Gleichungssystems kommt man auf einen mindestens erforderlichen Walzenradius von r = 178,575 mm. Das bedeutet also, dass die Walzen einen Durchmesser von mindestens 357,15 mm haben müssen.



$$\frac{h_0 - h_1}{2}$$

Tasächlicher Greifwinkel:

$$\alpha_0 := \operatorname{acos}\left(1 - \frac{(h_0 - h_1)}{2\,r}\right) = 4.07\ \mathbf{deg}$$

Abbildung 17: Greifwinkel

Aus dem hier dargestellten rechtwinkeligen Dreieck (siehe Abbildung 17) kann man durch den Einsatz des Pythagoreischen Lehrsatzes den tatsächlichen Eingriffswinkel beziehungsweise Reibwinkel errechnen. Würde man die Berechnung nochmals mit dem tatsächlichen Greifwinkel durchführen, so käme man auf einen Walzendurchmesser von ca. 344 mm.

$$\mu_{min.} := \frac{1-\cos(\alpha_0)}{\sin(\alpha_0)} = 0.036$$

minimalst erforderlicher Reibkoeffizient (4.3/19)

Abbildung 18: Mindestreibwert

Aus der in Abbildung 18 abgebildeten Formel errechnet sich der mindestens erforderliche Reibungskoeffizient, welcher zwischen den Walzen und der Münze gegeben sein muss, damit die Münze eingezogen wird. Der Reibungskoeffizient beschreibt wie stark die Reibung zwischen zwei Werkstoffen ist. In diesem Fall benötigt man den Reibwert von Stahl auf Kupfer. Übliche Werte für das Reibverhalten von Kupfer zu Stahl liegen zwischen 0.10 und 0.17.

<u>Auswahl des Konzepts</u>

Da die Diplomanden sich dazu entschlossen haben, das Einziehen der Münze mittels Reibschluss zu verwirklichen, wird nun vom Diplomand Grafeneder das passende Konzept dafür ausgewählt.

Weil sich aus der Berechnung ein Walzendurchmesser von 344 mm ergeben hat, steht nur mehr ein Konzept zur Auswahl. Es wird beschlossen, dass die Walzen mit einem großen Durchmesser, einer geringen Breite und mit radial angeordneten Motiven ausgeführt werden.

3.3.2. Berechnung des erforderlichen Walzmoments

Als Grundlage für alle weiteren Berechnungen des Getriebes und der Lager, sowie für die Auslegung des Motors, welche Diplomand Hochecker durchführt, benötigt man das Drehmoment, das erforderlich ist, um die Münzen durch die Walzen hindurchzuziehen. Diese Berechnungen basieren ebenfalls auf der Dissertation von Christian Overhagen namens "Modelle zum Walzen von Flach- und Vollquerschnitten."

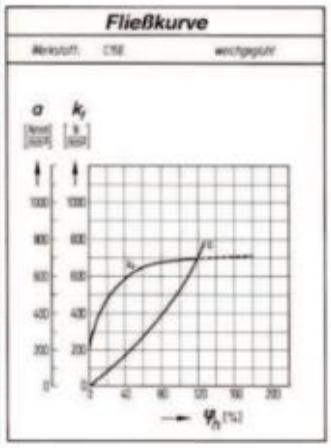
Basierend auf der in Abbildung 15 dargestellten Skizze wird der Verformungswirkungsgrad berechnet (siehe Abbildung 19). Dieser gibt an, wie gut sich das Material zur Verformung eignet.

$$\varphi := \left|\ln\left(\frac{h_1}{h_0}\right)\right| = 0.736$$

Verformungswirkungsgrad

Abbildung 19: Verformwirkungsgrad

Mit Hilfe des in Abbildung 20 dargestellten Diagrammes und der dazugehörigen Tabelle aus dem Skriptum „Praxis der Umformtechnik" von J. Dietrich, H. Tschätsch (Springer Fachmedien Wiesbaden 2013) kann die voraussichtliche Fließspannung des Münzenwerkstoffs ermittelt werden. Da die genaue Werkstoffbezeichnung nicht bekannt ist, wird eine Fließspannung von einem Werkstoff mit ähnlichem Rm-Wert für die Berechnung herangezogen. Der Rm-Wert beschreibt die Zugfestigkeit eines Materials.



Werkstoff		k_{f0}	k_{f100}	n
alt	neu	N/mm²	N/mm²	
Ck 10	C10E	260	740	0,216
Ck 15, Cq 15	C15E, C15C	280	760	0,165
Ck 22, Cq 22	C22E, C22C	320	760	0,157
Ck 35, Cq 35	C35E, C35C	340	950	0,178
Ck 45, Cq 45	C45E, C45C	390	1000	0,167
Cf 53		430	1140	0,170
34 Cr 4	34Cr4	410	970	0,118
42 Cr Mo 4	42CrMo4	420	1100	0,149
Al 99,5	EN AW-1050A	60	150	0,222
Al Mg Si 1	EN AW-6082	130	260	0,197
Cu Zn 10	CuZn10	250	600	0,331
Cu Zn 15	CuZn15	250	600	0,331
Cu Zn 30	CuZn30	250	880	0,497
Cu Zn 37	CuZn37	280	880	0,433

Werkstoff: C35E
(der Rm Wert, ist den unseren Ermittelten am nächsten)

$$k_{f100} := 950 \frac{N}{mm^2}$$

$$n := 0.178$$

Abbildung 20: Fließkurven

Mit den zuvor aus der Tabelle herausgelesenen Werten k_{f100} und dem Exponenten n, sowie dem errechneten Verformungswirkungsgrad aus Abbildung 19 kann nun mit der in Abbildung 21 dargestellten Formel die Formänderungsfestigkeit k_f berechnet werden.

Tabelle 2.1
Fließspannungen k_{f_0} und $k_{f_{100\%}}$ und Exponent n

$$k_{f_1} = k_{f_{100\%}} \cdot \varphi_h^n \qquad \varphi_h\,(-)$$

Formänderungsfestigkeit

$$k_f := k_{f100} \cdot \varphi^n = 899.548 \frac{N}{mm^2}$$

Abbildung 21: Formänderungsfestigkeit

Daraus ergibt sich dann mit der Formel aus Abbildung 22 die mittlere Formänderungsfestigkeit.

$$k_{fm} := \left| \int_{\varphi^n}^{\varphi} k_f \, d\varphi \right| = 189.738 \frac{N}{mm^2}$$

mittlere Formänderungsfestigkeit

Abbildung 22: mittlere Formänderungsfestigkeit

Für die Berechnung des mittleren Umformwiderstands k_{wm} (siehe Abbildung 23) wird der Umformwirkungsgrad η_U benötigt. Dieser liegt laut Skriptum zwischen 0.4 und 0.8, weshalb bei dieser Berechnung 0.6 angenommen wird.

$$k_{wm} := k_{fm} \cdot \frac{1}{\eta_U} = 316.23 \ \frac{N}{mm^2}$$ Mittlerer Umformwiderstand (Glg. 4.3/22)

Abbildung 23: mittlerer Umformwiderstand

Abgeleitet von der Skizze, welche in Abbildung 15 dargestellt ist, wird die gedrückte Länge der Münze sowie die Fläche, welche während des Prägevorgangs im Eingriff ist, berechnet (siehe Abbildung 24). Die gedrückte Länge der Münze ist jene Länge, um welche sich die Münze in Walzrichtung verformt. Das heißt, dass die Münze letztendlich ca. 34 mm lang sein wird. Die gedrückte Fläche ist jene Fläche, die während des Prägevorgangs insgesamt im Eingriff, heißt zwischen den Walzen, sein wird.

$$l_d := \sqrt{r \cdot (h_0 - h_1) - \frac{(h_0 - h_1)^2}{4}}$$ gedrückte Länge

$$l_d = 12.243 \ mm$$

$$A_d := l_d \cdot d_{Münze} = 260.159 \ mm^2$$ Fläche im Eingriff

Abbildung 24: gedrückte Länge, Fläche im Eingriff

Ausgehend von der Fläche im Eingriff wird nun die Kraft, welche zur Umformung der Münze benötigt wird, berechnet (siehe Abbildung 25).

$$F := A_d \cdot k_{wm} = 82.27 \ kN$$ Notwendige Umformkraft (Glg. 4.3/23)

Abbildung 25: benötigte Umformkraft

Die hier errechnete Umformkraft kann annähernd mit den aus den Druckversuchen ermittelten Kräften verglichen werden. Hier wurde eine Kraft von 82.27kN berechnet und beim Druckversuch wurde eine Kraft von ca. 85kN ermittelt. Da jedoch ein Drehmoment und keine Kraft benötigt wird, muss nun die errechnete Kraft mit dem Radius der Walzen und dem beim Walzen vorhandenen Hebelarmbeiwert multipliziert werden, um das Drehmoment zu erhalten.

$$a_H := \left(\frac{\frac{2}{3} \cdot h_1 + \frac{1}{3} h_0}{(h_0 - h_1)} \right) \cdot \varphi = 0.922$$

Hebelarmbeiwert (Glg. 4.3/31)

Abbildung 26: Hebelarmbeiwert

Das errechnete Drehmoment entspricht dem Moment, welches insgesamt auf den beiden Walzen vorhanden sein muss. Da zur Auslegung nur das Moment auf eine Walze benötigt wird, wird das erforderliche Walzenmoment durch 2 geteilt (siehe Abbildung 27).

$$M_d := F \cdot l_d \cdot a_H = 928.726\ Nm$$

erforderliches Moment für beide Walzen (Glg. 4.3/31)

$$M := \frac{M_d}{2} = 464.363\ Nm$$

Moment pro Walze

Abbildung 27: erforderliches Walzmoment

Bei Konstruktionen und Auslegungen von Bauteilen wird immer mit einer Sicherheit gerechnet. Je nach Anwendungsfall kann diese variieren. Bei dieser Arbeit wird eine Sicherheit von 1.5 gewählt. Um diese mit in die Berechnung einfließen zu lassen, muss das Moment je Walze mit der Sicherheit multipliziert werden (siehe Abbildung 28).

$\nu := 1.5$	Sicherheit
$M_{erf.} := M \cdot \nu = 696.544\ Nm$	Moment inkl. Sicherheit

Abbildung 28: Walzmoment inklusive Sicherheit

Mit dem hier berechneten Walzmoment kann nun in die Auslegung des Getriebes übergegangen werden.

4. Auslegung und Konstruktion des Getriebes

4.1. Konzepte Getriebe

Parallel zu den Konzepten für die Prägewalzen, werden zeitgleich dazu passende Getriebe-Konzepte ausgearbeitet.

4.1.1. Konzept 1: Getriebe für Walzen mit kleinem Durchmesser

Das erste Konzept für das Getriebe wurde vom internationalen Projekt „PPM4EU“ übernommen. Da die technische Leitung des Projekts bei der österreichischen Delegation liegt, sind alle dazu angestellten Überlegungen vom Projektteam dieser Diplomarbeit.

Das erste Konzept des Getriebes basiert auf dem ersten Konzept der Prägewalzen. Die Umformung passiert dabei mit zwei Walzen mit eher kleinen Durchmessern.

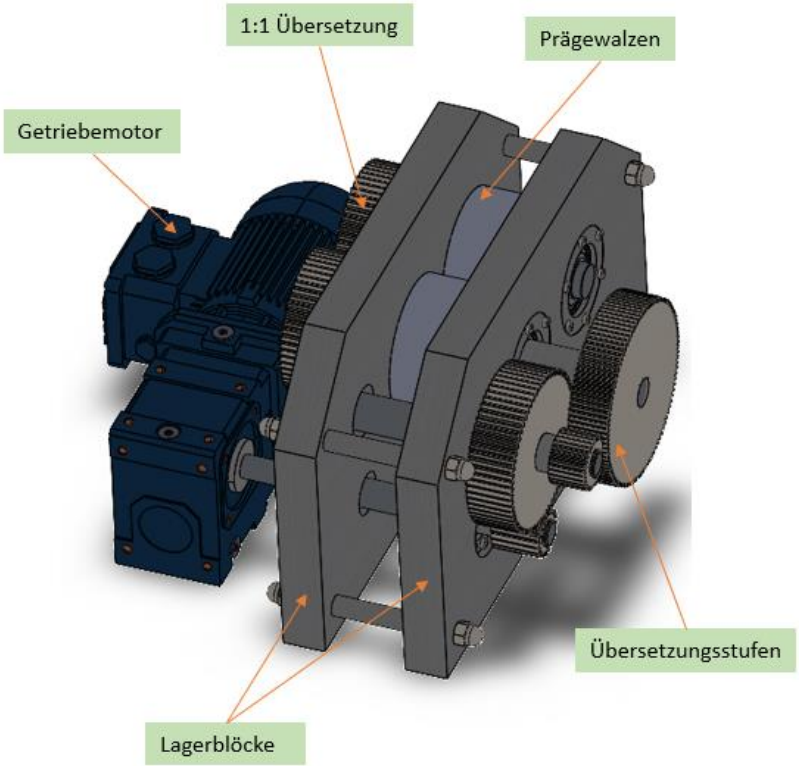


Abbildung 29: Schematische Konstruktion des Getriebes

In „Abbildung 2“ ist eine schematische Konstruktion des Getriebes zu sehen. Ein Getriebemotor von der Firma „Watt-Drive“ treibt das Getriebe zur Münzprägung an. Um den Maschinenbediener einen Einblick in die Mechanik zu gewähren, sind die beiden Getriebestufen vor dem Lagerblock positioniert. Die Prägung selbst findet zwischen den beiden Lagerblöcken statt.
Die Prägewalzen werden als Ringe gefertigt, welche auf den Getriebewellen montiert werden können. Auf diese Weise sind bei Verschleißerscheinungen nur die Prägeringe und nicht die ganzen Getriebewellen zu tauschen.
Um die passende Drehrichtung und das gleiche Drehmoment auf den Walzen zu erhalten, sind die Wellen mit dem hinteren Zahnradpaar (Übersetzung = 1:1) verbunden.
Um die gewünschte Positionierung sicher zu stellen, werden die beiden Lagerblöcke und der Motor auf einer gemeinsamen Grundplatte verschraubt.

4.1.2. Konzept 2: Getriebe für Walzen mit großem Durchmesser

Nach der Entscheidung, die Walzen mit einem großen Durchmesser und als Ringe zu fertigen, wurde ein neues Konzept entwickelt. Die Walzen sitzen nun nicht mehr auf rotierenden Getriebewelle, sondern auf stillstehenden Achsen (siehe Abbildung 3).

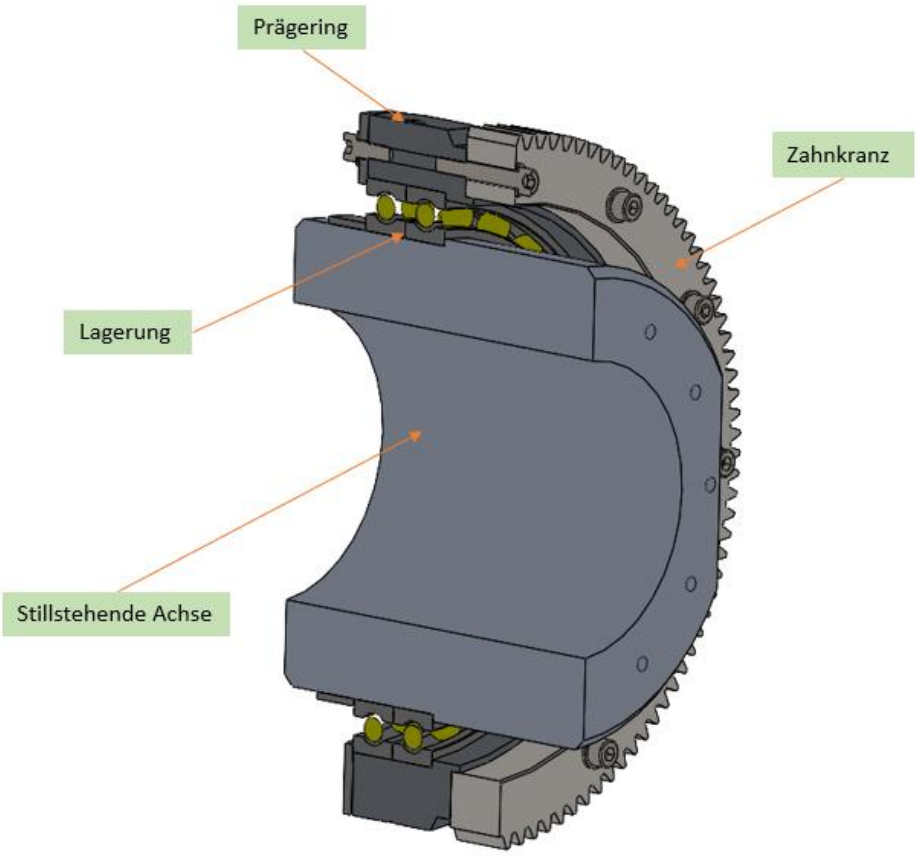


Abbildung 30: Aufbau mit stillstehender Achse

Anders als beim ersten Konzept, ist nicht die Welle gelagert. Die Lagerung befindet sich direkt unter den Walzen, welche auf einer stilstehenden Achse laufen. Diese Achse wird mit den Lagerblöcken verstiftet und verschraubt.

Ein großer Vorteil der innen hohlen Achsen ist der Platzgewinn. Somit kann beispielsweise der Antrieb oder die Elektronik für die Steuerung im Inneren der Achsen positioniert werden.

Konzepte Antrieb

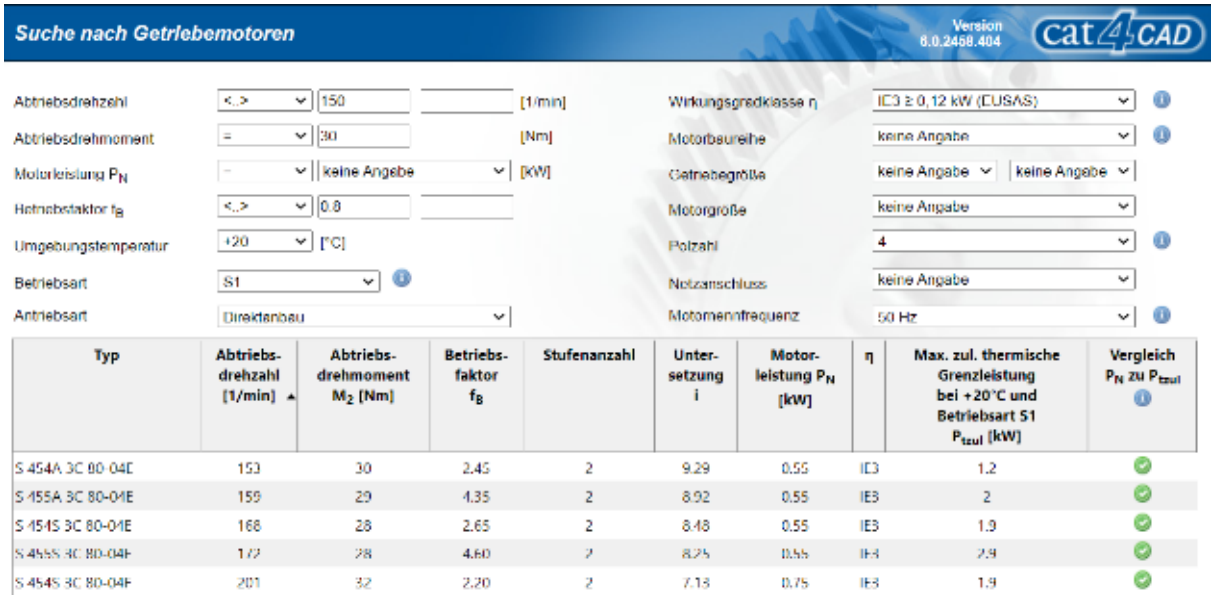
Für den Vorgang des Münzprägens ist auf den Prägewalzen ein hohes Drehmoment sowie eine sehr geringe Drehzahl erforderlich.
Um diese gewünschten Parameter zu erhalten, wird ein Getriebe, welches von einem Motor angetrieben wird, eingesetzt.
Für die Art des Antriebs kommen verschiedene Motortypen infrage. Um den passenden Motor zu wählen, wurden drei Konzepte erstellt und diese miteinander verglichen.

4.1.3. Konzept 1: Getriebemotor

Im ersten Konzept wird ein zweistufige Stirnradgetriebe von einem Getriebemotor von der Firma „Watt Drive“ angetrieben.

Abbildung 31: Getriebemotor v. Watt Drive (http://www.wattdrive.com/wg20/)

Der große Vorteil an dieser Lösung ist, dass auf der Internetseite des Unternehmens ein Motor-Konfigurator zu Verfügung steht. So kann aus dem gewünschten Abtriebsmoment und der Drehzahl auf den Walzen sowie der Übersetzung des Getriebes, die Abtriebsparameter des Motors errechnet werden. Diese Werte können anschließend in den Konfigurator eingegeben und für die Anwendung der beste Motor ausgewählt werden.



Typ	Abtriebs-drehzahl [1/min]	Abtriebs-drehmoment M_2 [Nm]	Betriebs-faktor f_B	Stufenanzahl	Unter-setzung i	Motor-leistung P_N [kW]	η	Max. zul. thermische Grenzleistung bei +20°C und Betriebsart S1 P_{tzul} [kW]	Vergleich P_N zu P_{tzul}
S 454A 3C 80-04E	153	30	2.45	2	9.29	0.55	IE3	1.2	✓
S 455A 3C 80-04E	159	29	4.35	2	8.92	0.55	IE3	2	✓
S 454S 3C 80-04E	168	28	2.65	2	8.48	0.55	IE3	1.9	✓
S 455S 3C 80-04E	172	28	4.60	2	8.25	0.55	IE3	2.9	✓
S 454S 3C 80-04E	201	32	2.20	2	7.13	0.75	IE3	1.9	✓

Abbildung 32: Beispiel für Motorauswahl (https://cat4cad.wattdrive.com/cat4cad1/)

4.1.4. Konzept 2: Gleichstrommotor

Eine weitere Möglichkeit die „Penny Press“ anzutreiben, wäre ein Gleichstrommotor aus dem Modellbau.

Der Preis wäre im Vergleich zum fertigen Getriebemotor wesentlich günstiger und die Bauform wesentlich kompakter. Der Gleichstrommotor liefert jedoch enorme Drehzahlen (ab ca. 3000 U/min), welche zumindest eine dritte Getriebestufe erfordern würden. Zudem müsste bei den hohen Drehzahlen die erste Getriebestufe geschmiert werden. Die zusätzliche Getriebestufe und die erforderliche Schmierung bringen höheren Konstruktionsaufwand, Material- und Wartungskosten mit sich.

4.1.5. Konzept 3: Antrieb eines Akkuschraubers

Im dritten Konzept werden die Vorteile der Ideen aus den beiden vorherigen Konzepten kombiniert.

Es wird überlegt den Antrieb (Motor und Getriebe) eines Akkuschraubers auszubauen und für den Anwendungsfall einer Münzprägemaschine zu nutzen.

Die Drehzahl, des sehr kompakt gehaltenen Motors, wird durch das Planetengetriebe gemindert sowie das Drehmoment erhöht. Dies ermöglicht eine kleinere Übersetzung im Hauptgetriebe der Prägeeinheit.

Da ein Akkuschrauber im Regelfall über zwei Gänge verfügt, ist das Drehmoment und die Drehzahl über einen gewissen Bereich regelbar. Zudem ist der Preis verglichen mit einem Getriebemotor recht günstig.

Aus Zufall ist bereits ein Akkuschrauber vorhanden. Deswegen wird entschieden dieses Konzept zu wählen und dessen Vorteile zu nutzen.

Verbaut werden die Komponenten eines Akkuschraubers der Firma „Bosch“ mit der Bezeichnung „Professional GSR 18V-28“.

4.2. Konstruktionsstruktur

Um während dem Arbeiten an der Diplomarbeit stets den Überblick zu behalten, wurde die Excel-Datei „Braugruppenstruktur“ erstellt (siehe Abbildung 33).

In dieser Datei sind alle verbauten Baugruppen, Unterbaugruppen, Einzelteile und Normteile aufgelistet.

Baugruppenstruktur_PPM

Baugruppen: Achse3_ZSB, Achse4_ZSB, Antrieb_ZSB, ASM Stepper, ASM Zuführung, PPM_ZSB, Welle1_ZSB, Welle2_ZSB

Zeichnung vh: 3D-Druck, Ja, Nein

Status: 25, 50, 75

Baugruppen	Unterbaugruppen / Einzelteile	Teilenummern	Zeichnungsnummern	Zeichnung vh	Check Schüler	Check Professor	Status Produktion	Status	Zeichnung	Kosten	Rechnung
PPM_ZSB	Welle2_ZSB	5BHMBA-DA-Asm-W2		Nein	Ja	Nein	-				
PPM_ZSB	Achse3_ZSB	5BHMBA-DA-Asm-A3		Ja	Nein	Nein	-				
PPM_ZSB	Achse4_ZSB	5BHMBA-DA-Asm-A4		Ja	Nein	Nein	-				
PPM_ZSB	Antrieb_ZSB	5BHMBA-DA-Asm-Ant		Nein	Nein	Nein	-				
PPM_ZSB	ASM Stepper	5BHMBA-DA-Asm-St		Nein	Nein	Nein	-				
PPM_ZSB	ASM Zuführung	5BHMBA-DA-Asm-Zuf		Nein	Nein	Nein	-				
PPM_ZSB	Lagerblock_vorne	5BHMBA-DA-P-Lbvorne	5BHMBA-DA-D-LBvorne	Ja	Nein	Nein	-			€ 409,45	
PPM_ZSB	Lagerblock_hinten	5BHMBA-DA-P-Lbhinten	5BHMBA-DA-D-LBhinten	Ja	Nein	Nein	-			€ 163,50	
Welle1_ZSB	Welle1	5BHMBA-DA-P-W1	5BHMBA-DA-D-W1	Ja	Ja	Ja	Fertig			€ 45,00	
Welle1_ZSB	Distanzhülse_W1	5BHMBA-DA-P-DHW1	5BHMBA-DA-D-DHW1	Ja	Ja	Ja	Fertig				
Welle1_ZSB	Wellendeckel_W1	5BHMBA-DA-P-WDW1	5BHMBA-DA-D-WDW1	Ja	Ja	Ja	Fertig				
Welle1_ZSB	Zahnrad1	5BHMBA-DA-P-Z1	5BHMBA-DA-D-Z1	Ja	Ja	Ja	Fertig			€ 32,14	
Welle2_ZSB	Welle2	5BHMBA-DA-P-W2	5BHMBA-DA-D-W2	Ja	Ja	Ja	Freigegeben				
Welle2_ZSB	Distanzhülse1_W2	5BHMBA-DA-P-DH1W2	5BHMBA-DA-D-DH1W2	Ja	Ja	Ja	Fertig				

Abbildung 33: Grobübersicht d. Baugruppenstruktur (Datei: Baugruppenstruktur)

Jeder Fertigungsteil enthält dabei:

- Eine Zuordnung zur Baugruppe
- Einen Teilenamen
- Eine Teil- sowie Zeichnungsnummer
- Den Produktionsstatus
- Eine Fertigungszeichnung
- Die Kosten (inkl. Rechnungen)
- Fertigungszeit

Mit dieser Datei kann das Projektteam in den Produktionsstatus der gesamten Diplomarbeit einsehen. Es sollte somit nicht passieren, dass auf das Erstellen von einzelnen Zeichnungen oder den Rohteilkauf vergessen wird.

4.2.1. Einfügen eines neuen Teils

Baugruppen	Unterbaugruppen / Einzelteile	Teilenummern	Zeichnungsnummern
Welle1_ZSB	Welle1	5BHMBA-DA-P-W1	5BHMBA-DA-D-W1
Welle1_ZSB	Distanzhülse_W1	5BHMBA-DA-P-DHW1	5BHMBA-DA-D-DHW1

Abbildung 34: Neuen Teil einfügen

Wird ein neuer Teil eingefügt, muss dieser einer Baugruppe zugewiesen werden (Spalte 1). Danach wird ein Name sowie eine Teile- und Zeichnungsnummer vergeben (Spalte 2, 3, 4).

Diese Parameter sind vor allem beim Erstellen von neuen Zeichnungen sehr wichtig. In der Datei können die Zeichnungsnummern übersichtlich angelegt- und anschließend in die Zeichnung kopiert werden. Somit vergibt man keine Nummer doppelt und behält den Überblick.

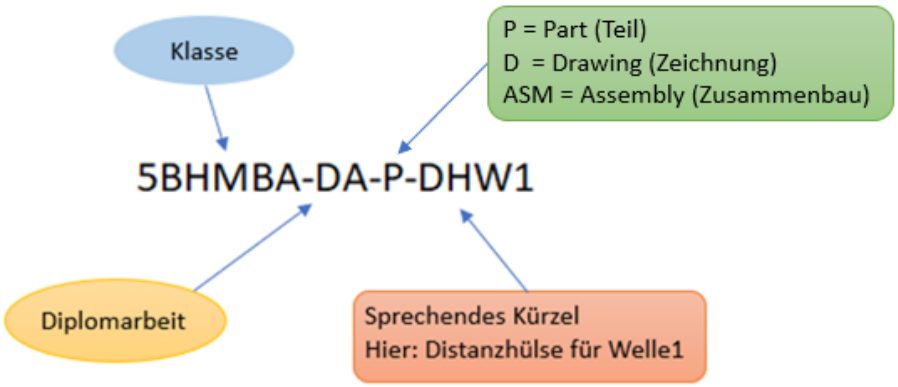


Abbildung 35: Aufbau der Zeichnungsnummern

4.2.2. Funktion der Exceltabelle

4.2.3. Filtern

Um das Arbeiten mit den Zeichnungsnummer zu erleichtern, können die Teile nach den Baugruppen und dem Fertigungsstatus gefiltert werden.

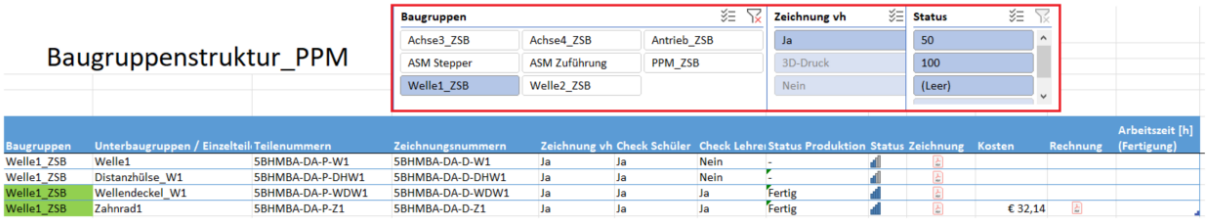


Abbildung 36:Filtern nach Baugruppen

4.2.4. Fertigungsstand

Nachdem eine Zeichnung erstellt wird, soll diese vor der Fertigung von anderen Schülern und einem Professor kontrolliert werden. Dieser Status wird in der Tabelle notiert (siehe Abbildung 37).

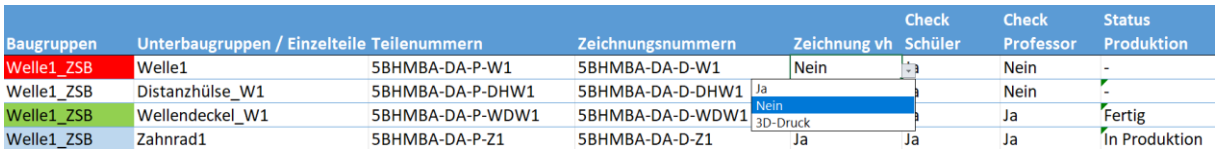


Abbildung 37: Fertigungszustand

Abhängig vom Stadium der Zeichnung wird die erste Spalte des Teils eingefärbt:

- Rot = keine Zeichnung vorhanden
- Blau = zur Fertigung freigegeben
- Grün = Werkstück ist fertig

4.2.5. Verwaltung

Am Ende der Zeilen sind Zeichnungen zum Werkstück und Rechnungen für das Rohmaterial verlinkt. Diese Dateien können direkt aus der Excel-Tabelle geöffnet werden (Klick auf PDF-Symbol).

Mithilfe dieser Funktionen können anfallende Rechnungen sowie bereits ausgegebenes Geld im Überblick gehalten werden.

Zeichnung	Kosten	Rechnung	Arbeitszeit [h] (Fertigung)
	€ 45,00		5
			3
	€ 32,14		4

Abbildung 38: Verwaltung

4.3. Konstruktion Getriebe

4.3.1. Allgemein

4.3.1.1. Verwendete Programme

Zur Konstruktion des Getriebes der Münzprägeeinheit wurde die CAD-Software „SolidWorks 2020“ verwendet. Während der Konstruktion wurden die belasteten Bauteile und deren Festigkeit mit dem Programm „Math CAD“ nachgerechnet.

4.3.1.2. Steuerskizze

Zum Erstellen der Einzelteile sowie der Baugruppen wurde die Funktion einer Steuerskizze genutzt. Das bedeutet, alle Teile sind auf eine Skizze referenziert. Wird diese Skizze verändert, so werden automatisch alle Änderungen in den Einzelteilen aktualisiert. Dies erleichtert vor allem den späteren Zusammenbau der Einzelteile.

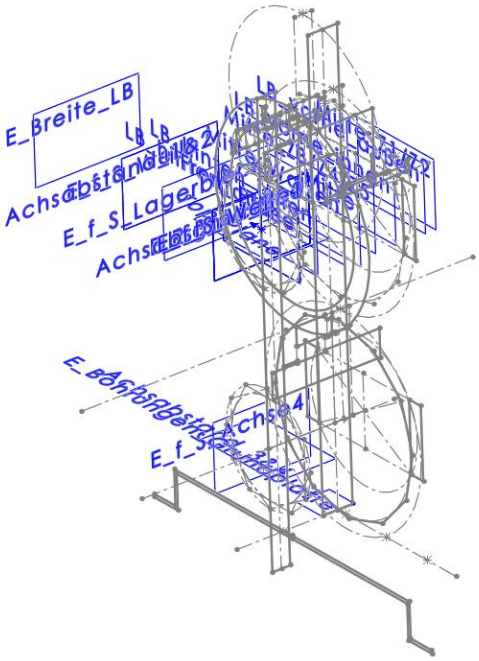


Abbildung 39: Grobansicht Steuerskizze

Eine Steuerskizze besteht im Groben aus Ebenen (blau) und Skizzen (grau).

Wenn die Skizzen direkt auf die Ebenen referenziert werden, fixiert dies deren Lage im Raum. Mit dem Verschieben der Skizzen können Maße und Positionen verändert werden.

In der Steuerskizze wird für jedes verbaute Bauteil eine eigene Skizze eingefügt. In dieser können Bemaßungen direkt auf andere Bauteile oder Ebenen gesetzt werden. Besonders wichtig ist es, jede Skizze mit dem Namen des späteren Bauteils zu versehen (siehe Abbildung 40).

Beim Erstellen eines neuen Bauteils wird die gesamte Steuerskizze eingefügt. Anschließend wird die gewünschte Bauteilskizze ausgewählt und beispielsweise um eine Achse rotiert. Werden die Skizzen davor in der Steuerskizze nicht benannt, so verliert man den Überblick.

Abbildung 40: Achsen, Ebenen und Skizzen

4.3.2. Hauptzusammenbau

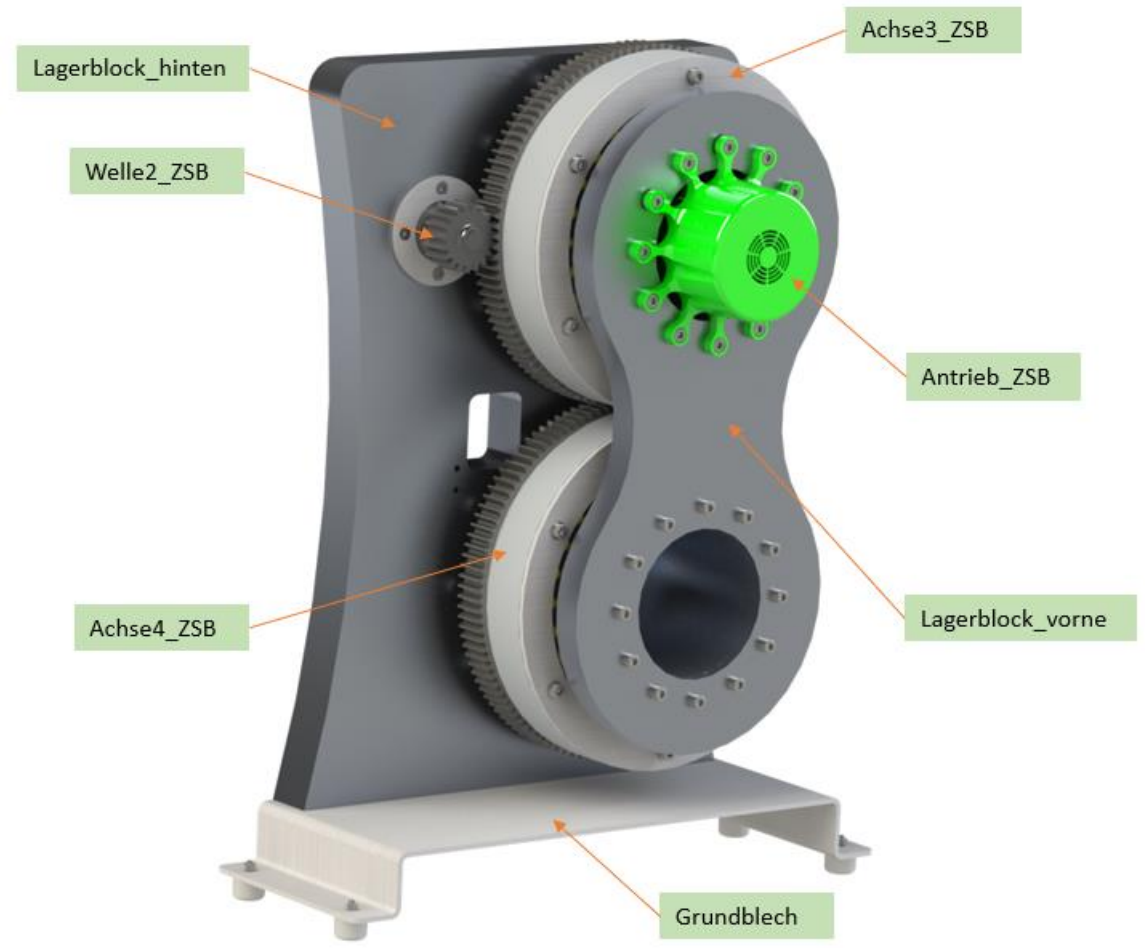


Abbildung 41:Hauptzusammenbau mit Unterbaugruppen

In „Abbildung 41“ ist der Hauptzusammenbau der Münzprägeeinheit inkl. Getriebe und Antrieb abgebildet. Dazu sind alle Unterbaugruppen angeführt.

Auf den folgenden Seiten wird jede Unterbaugruppe eigens abgebildet und die Konstruktion sowie die dazugehörige Berechnung beschrieben. Die Schilderung wird beim Antrieb begonnen und entlang der Getriebestufen weitergeführt.

Nach der Beschreibung aller Baugruppen wird erklärt, wie und in welcher Reihenfolge die Hauptbaugruppe zusammengesetzt wird.

4.3.3. Theoretische Übersetzung

Aus der Datei „Berechnung_Walzmoment“ kann das erforderliche Drehmoment auf den Walzen entnommen werden. Aus der Drehzahl des Akkuschraubers und einer gewünschten Abtriebsdrehzahl wird die erforderliche Gesamtübersetzung des Getriebes berechnet (siehe Abbildung 42).

Berechnung Verzahnung

$Nm := N \cdot m$

$M_{Ab} := 750\ Nm$	gefordertes Abtriebsmoment auf den Walzen
$n_{Ab} := 10\ min^{-1}$	gewünschte Drezahl der Walzen
$n_{An} := 380\ min^{-1}$	Antriebsdrezahl des Motors
$\eta := 0.9$	Wirkungsgrad des Getriebes
Verzahnungsqualität:	8
Zahnradwerkstoff:	C45 (Zahnräder v. Norelem) 42CrMo4 (Ritzelwelle)

Auswahl der Übersetzungen:

$i_{ges} := \frac{n_{An}}{n_{Ab}} = 38$	theoretische Gesamtübersetzung

Abbildung 42: Berechnung der erforderlichen Gesamtübersetzung (Datei: „Berechnung_Getriebe)

Um die theoretische Gesamtübersetzung zu realisieren, wird ein zweistufiges Stirnradgetriebe ausgelegt und berechnet.

4.3.4. Übersicht der Getriebestufen

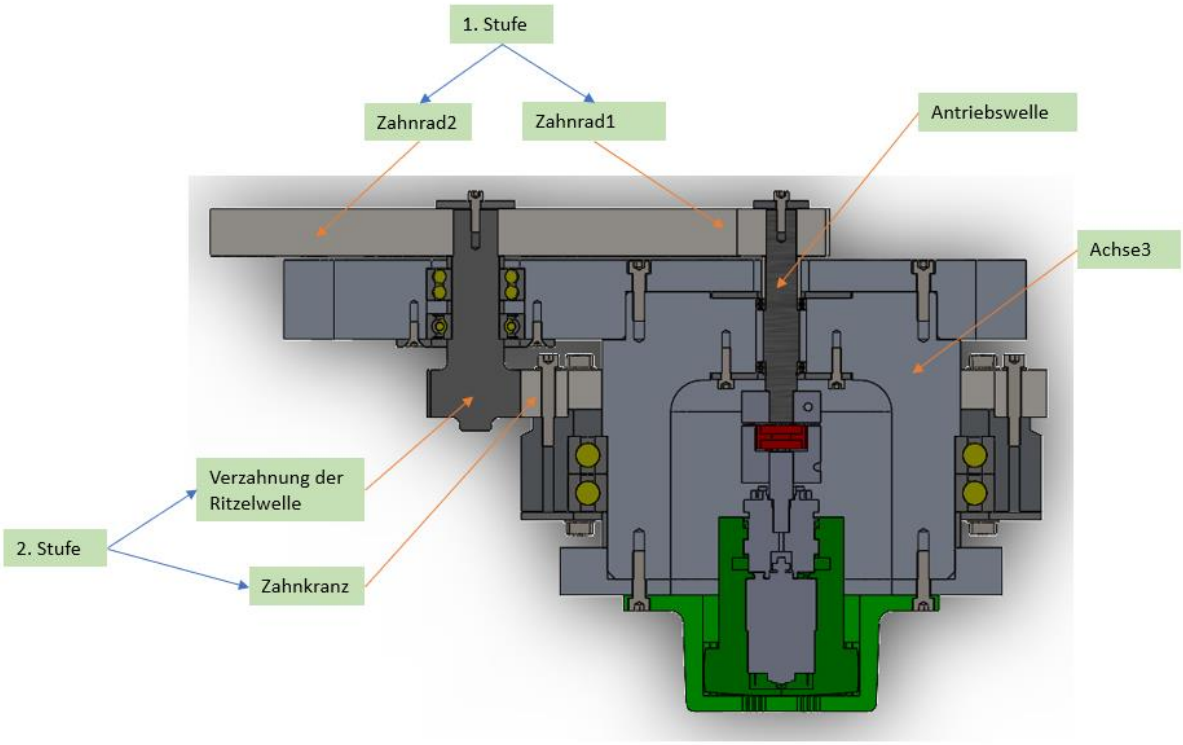


Abbildung 43: Überblick Getriebestufen 1 & 2

Auf Abbildung 43 ist ein Überblick der ersten zwei Getriebestufen zu sehen. Die Tatsache, dass die Antriebswelle konzentrisch in „Achse3“ sitzt, erschwert die Auswahl der Zahnräder.

Damit alle Verzahnungen korrekt ineinandergreifen, müssen jeweils beide Zahnräder der ersten zwei Getriebestufen exakt denselben Achsabstand aufweisen. Der passende Achsabstand der Zahnräder ergibt sich aus deren Teilkreisdurchmessern. Dieser ergibt sich wiederum aus der Zähnezahl und dem Modul (siehe Abbildung 44).

$$a_1 := \frac{(m_{12} \cdot z_1) + (m_{12} \cdot z_2)}{2} = 193.5\ mm$$ Achsabstand Stufe 1

$$a_2 := \frac{(m_{34} \cdot z_3) + (m_{34} \cdot z_4)}{2} = 193.5\ mm$$ Achsabstand Stufe 2

Abbildung 44: Berechnung Achsabstand 1 & 2

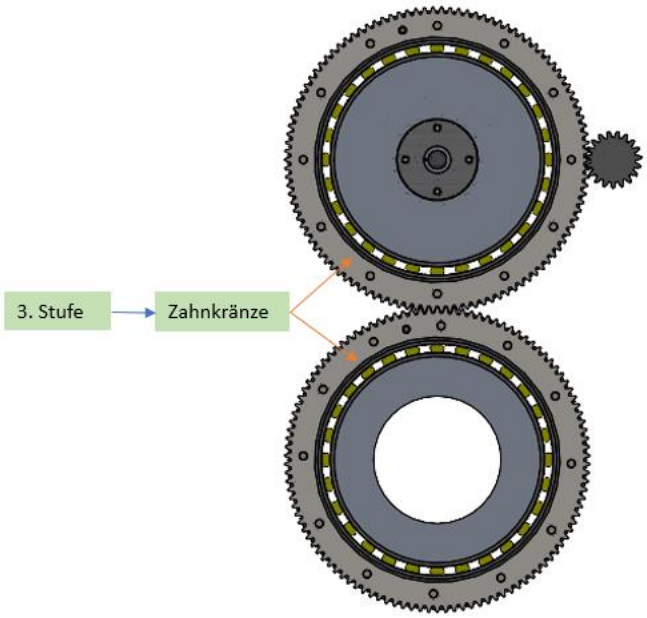


Abbildung 45: Überblick Getriebestufe 3

$$a_3 := \frac{(m_{34} \cdot z_4) + (m_{34} \cdot z_4)}{2} = 330\ mm$$ Achsabstand Stufe 3

Abbildung 46: Berechnung Achsabstand 3

Der Achsabstand der dritten Getriebestufe muss keine besonderen Anforderungen erfüllen. Der Achsabstand und der Teilkreis der Zahnkränze sind nur zur Auslegung des Außendurchmessers der Prägewalzen zu berücksichtigen.

4.3.5. Antriebsstrang

Der Antriebsstrang reicht vom Motor, welcher die Münzprägemaschine antreibt, bis zur ersten Getriebewelle.

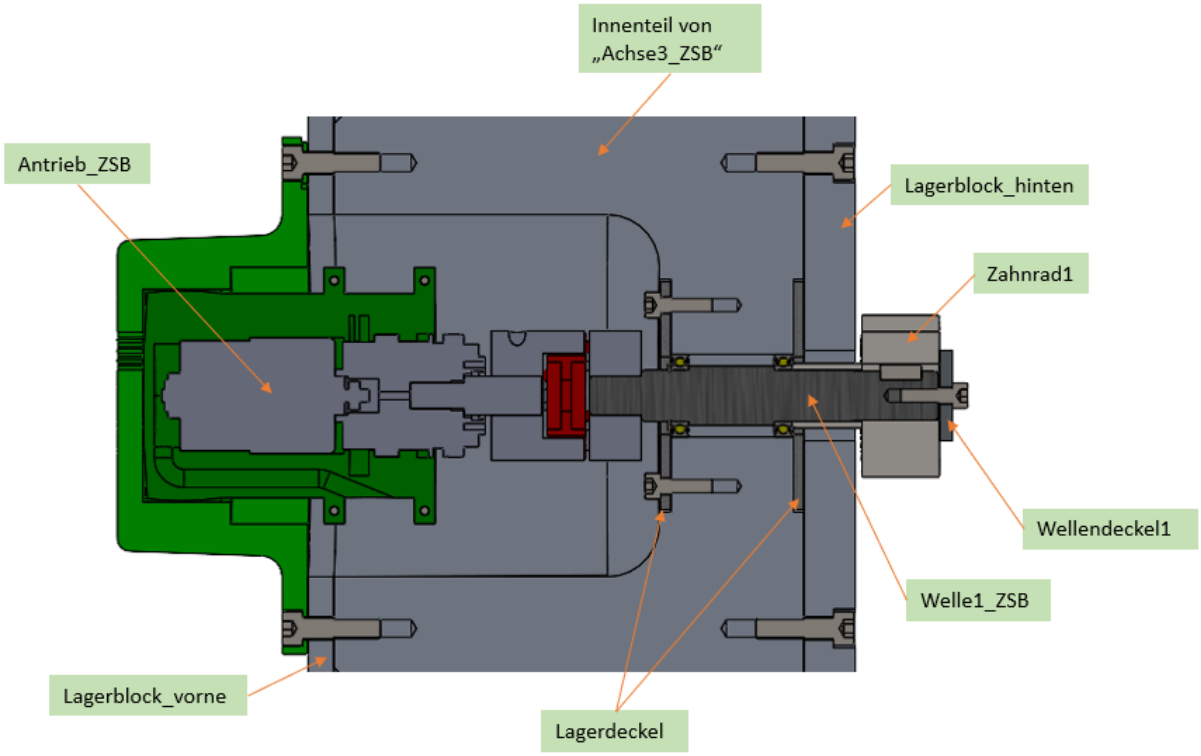


Abbildung 47: Antriebsstrang (geschnitten)

Der Antriebsstrang ist in die Baugruppen „Antrieb_ZSB“, „Achse3_ZSB“ und „Welle1_ZSB“ unterteilt.
Die Unterbaugruppen „Antrieb_ZSB“ und „Welle1_ZSB“ werden mit einer Elastomer-Klauenkupplung verbunden.

4.3.5.1. Unterbaugruppe – Antrieb ZSB

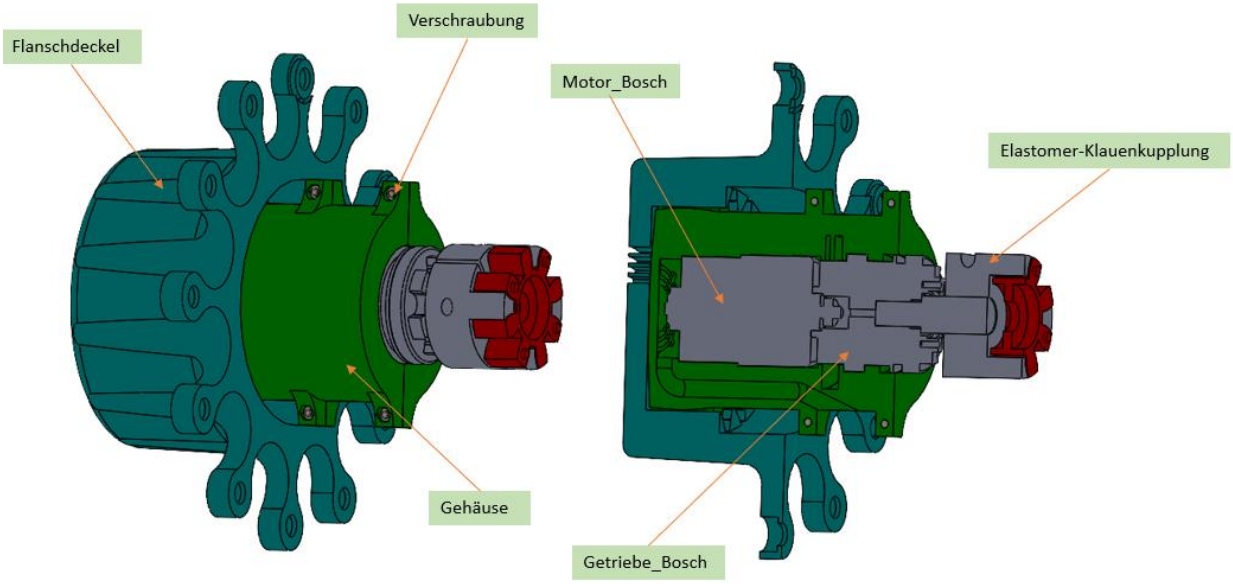


Abbildung 48: Unterbaugruppe-Antrieb_ZSB

Auf „Abbildung 48“ ist ein Überblick über die Unterbaugruppe „Antrieb_ZSB“ zu sehen. Diese Unterbaugruppe besteht aus fünf Einzelteilen und beinhaltet den Antrieb der Münzprägemaschine.

4.3.5.1.1. Antrieb

Zum Betreiben der Münzprägemaschine wird der Antrieb (Motor und Planetengetriebe) des Akkuschraubers GSR 18V-26 der Firma Bosch verwendet.
Dieser Akkuschrauber wird gekauft und anschließend in seine Einzelteile zerlegt. Auf „Abbildung 49“ ist das Innenleben dieses Produkts zu sehen.

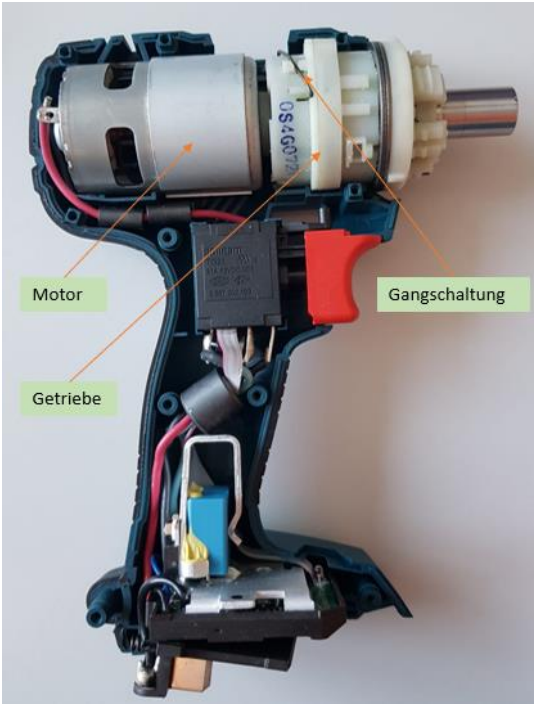


Abbildung 49: Innenleben Akkuschrauber

Da keine 3D-Zeichnung des verwendeten Motors inkl. Getriebe im Internet zu finden ist, werden die Teile aus dem Akkuschrauber ausgebaut, vermessen und abgezeichnet. Diese Arbeit ist notwendig, um sämtliche Abmessungen umliegender Bauteile auf den Antrieb abzustimmen.

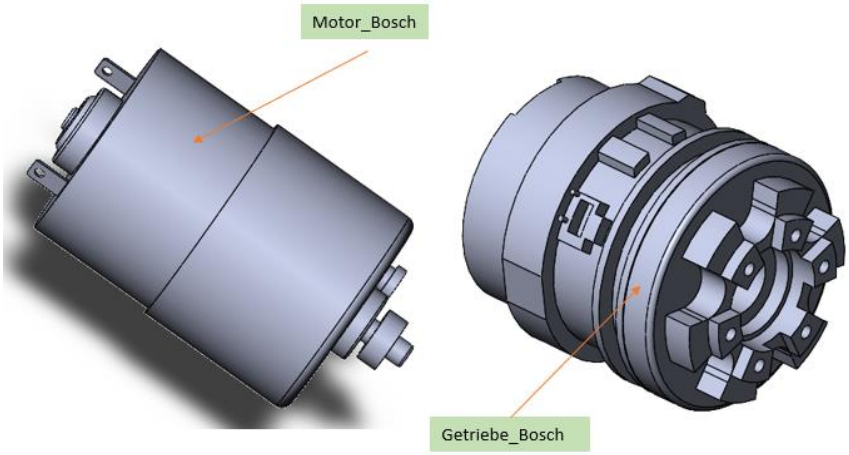


Abbildung 50: Motor und Getriebe (Bosch)

Vor allem beim Vermessen und Konstruieren des Getriebes muss sehr genau gearbeitet werden. Alle Absätze und „Noppen“ am Getriebe dienen zum genauen Positionieren und zum Sichern gegen Verdrehen im Gehäuse.

4.3.5.1.2. Gehäuse

Der Antrieb des Akkuschraubers wird bei der Münzprägemaschine in einem Gehäuse fixiert (siehe Abbildung 48). Dieses Gehäuse besteht aus zwei Hälften welche mittels 3D-Druck hergestellt werden. Die beiden Gehäusehälften werden nach dem Einsetzten des Antriebes miteinander verschraubt.

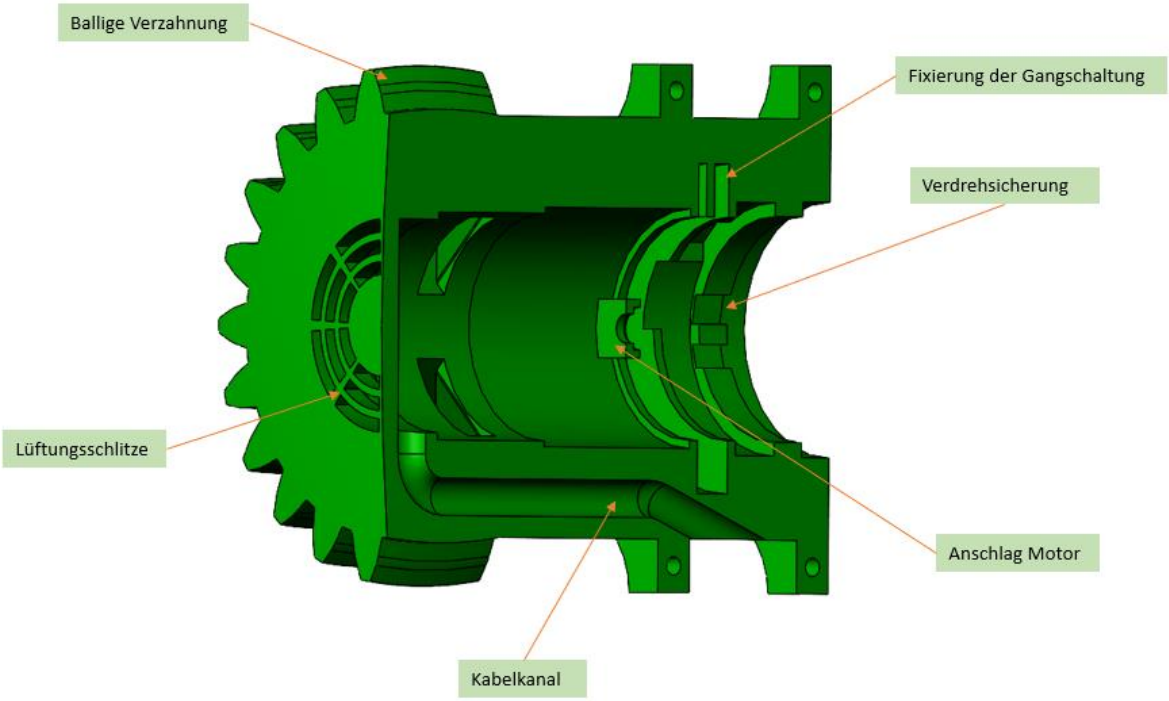


Abbildung 51: Gehäusehälfte

Die Innenkontur des Gehäuses wird nach den Maßen bzw. nach dem CAD-Modell des Antriebes von Bosch gezeichnet. Ist die Gehäusehälfte fertig gedruckt wird zuerst der Motor eingesetzt.

Mithilfe der Positionsschraube vom Motor wird die Lage sowie die Verdrehsicherung im Gehäuse positioniert (siehe Abbildung 52).

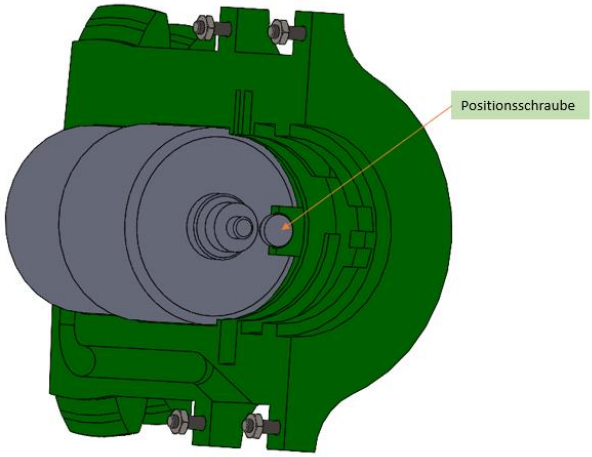


Abbildung 52: Positionsschraube Motor

Um im zusammengebauten Zustand zu den Anschlussklemmen des Motors zu gelangen, wird ein Kabelkanal ins Gehäuse gedruckt. Durch diesen werden die Kabel gelegt.

Um den Motor im Betrieb nicht zu überhitzten werden an der Rückseite und am Umfang des Gehäuses Lüftungsschlitze vorgesehen. Durch die Lüftungsschlitze an der Rückseite kann der Motor die benötigte Luft ansaugen und durch jene am Umfang wieder ausblasen.

Bevor das Getriebe in das Gehäuse eingelegt wird, muss der passende Gang eingelegt werden. Dies geschieht über eine Drahtklammer am Getriebe. Wird das Getriebe nun eingelegt, sitzt diese Klammer in der Ausnehmung. Durch diese konstruktive Lösung wird ein unerwünschtes Umschalten des Gangs ausgeschlossen.

Zwischen dem Antrieb des Akkumotors und der ersten Welle des Getriebes der Münzprägemaschine muss ein minimaler Winkelausgleich möglich sein, außerdem auch zwischen Gehäuse und Flanschdeckel. Bei diesem Ausgleich darf das Gehäuse nicht verdreht werden. Die konstruktive Lösung dazu ist eine ballige Verzahnung auf dem Gehäuse. Mithilfe des Gegenstücks im Deckel kann ein auftretender Winkelversatz ausgeglichen und das Gehäuse nicht verdreht werden (siehe Abbildung 53).

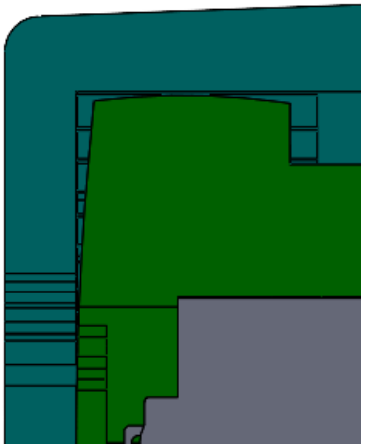


Abbildung 53: Ballige Verzahnung

4.3.5.1.3. Kupplung

Um einen möglichen Winkelversatz zwischen der Abtriebswelle des Akkuschrauber-Antriebs und der Antriebwelle des Getriebes der Münzprägemaschine auszugleichen, wird eine Elastomer-Klauenkupplung eingesetzt.

Aus der Dateien „Berechung_Getriebe“ stammt das erforderliche Antriebsmoment des Getriebes der Münzprägeeinheit.

Auswahl des Motors

Tatsächliche Momente auf den Wellen:

$M_{T4} := M_{Ab} = 750\ Nm$ erforderliches Abtriebsmoment der Prägewalzen

$M_{T3} := M_{T4} = 750\ Nm$

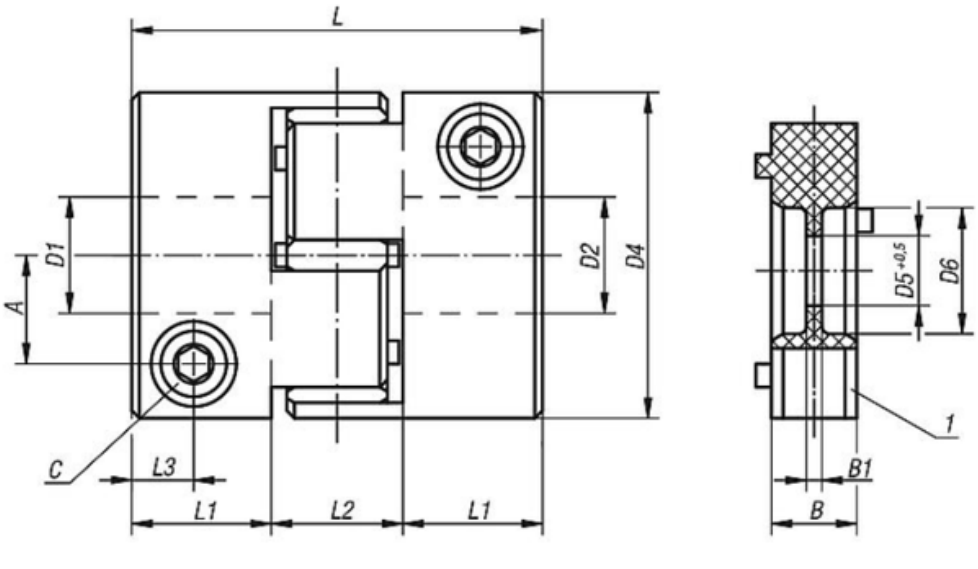
$M_{T2} := \frac{M_{T3}}{i_{34} \cdot \eta} = 143.939\ Nm$

$M_{T1} := \frac{M_{T2}}{i_{12} \cdot \eta} = 27.625\ Nm$ erforderliches Antriebsmoment auf Welle 1

Abbildung 54: Erforderliches Antriebsmoment

Der Motor der Firma Bosch wird so angesteuert, dass auf der Abtriebswelle des Planetengetriebes ein Drehmoment von 30Nm vorhanden ist. Die Auswahl der

Kupplung erfolgt ebenfalls über dieses Drehmoment. Die Kupplung wird aus dem Online-Katalog der Firma Norelem ausgewählt (Abbildung 55).



Artikelübersicht (komprimiert) » ARTIKELAUSWAHL/-FILTER

Reset

Bestellnummer	Größe	Nenn-moment Nm	Trägheits-moment (10^{-3} kgm²)	Statische Torsionssteife Nm/arcmin	max. Wellenversatz axial ±	max. Wellenversatz lateral	radiale Federsteife N/mm	Anziehdreh-moment der Schrauben Nm	CAD	Zub.	Preis	Bestellen
23022-002	2	2	0,00016	0,007	0,6	0,1	405	0,5			50,60 €	0
23022-005	5	5	0,0011	0,016	0,8	0,1	510	1			58,78 €	0
23022-008	8	8	0,01	0,04	0,5	0,1	600	4			80,83 €	0
23022-015	15	15	0,03	0,24	0,5	0,1	2100	8			98,09 €	0
23022-030	30	30	0,09	0,41	0,5	0,1	2500	14			117,01 €	0
23022-060	60	60	0,18	0,61	0,5	0,1	2600	35			138,79 €	0

Abbildung 55: Auswahl der Kupplung (Online: https://www.norelem.at)

4.3.5.1.4. Zusammenbau

Die gewählte Kupplung verfügt über eine radiale Klemmnabe. Eine der beiden Kupplungshälften wird inkl. Elastomer-Einsatz auf die Abtriebswelle des Planetengetriebes geklemmt. Anschließend steckt man den Motor mit den Planetengetriebe zusammen und setzt die Teile in eine Gehäusehälfte ein. Nach dem Verschrauben der beiden Gehäusehälften ist der Antrieb fixiert. Das Gehäuse mit dem bereits verbauten Antrieb wird nun in den Deckel eingesetzt welcher mit dem vorderen Lagerblock verschraubt wird (siehe Abbildung 56).

Abbildung 56: Antrieb_ZSB - Zusammenbau

4.3.5.2. Antriebswelle – Welle1 ZSB

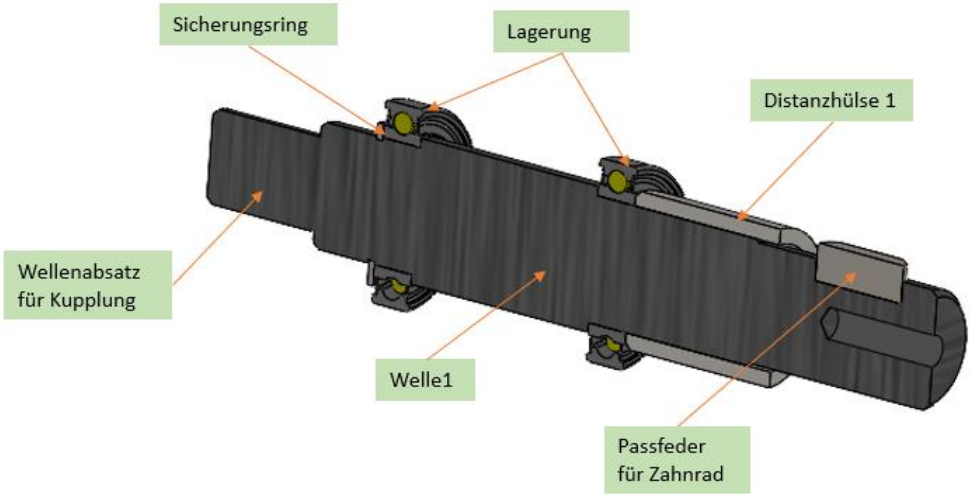


Abbildung 57: Unterbaugruppe Welle1_ZSB

Anordnung der Bauteile

Um die Position der Lager zu fixieren wir auf der Seite des Antriebes ein Sicherungsring verwendet. Auf der anderen Seite wird das Lager über die Distanzhülse, und später über das Zahnrad und den Wellendeckel gehalten. Der Wellendeckel ist mit dem Wellenende verschraubt (Gewindebohrung am Wellenende). Ist der Zusammenbau der Unterbaugruppe, wie in „Abbildung 57“ zu sehen, montiert, kann diese in die Unterbaugruppe „Achse3_ZSB“ eingesetzt werden.

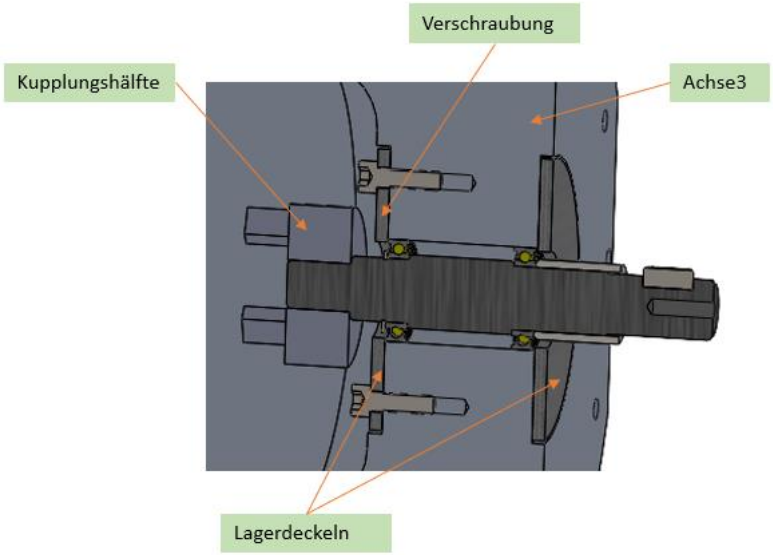


Abbildung 58: Welle1_ZSB (eingesetzt)

Durch die beiden Lagerdeckel werden die Außenringe der beiden Lager gehalten. Die Lagerung hätte auch als Fest-Loslagerung ausgeführt werden können, um den Fertigungsaufwand zu reduzieren wird jedoch eine schwimmende Lagerung gewählt. Bei einer schwimmenden Lagerung ist die Welle in axialer Richtung minimal verschiebbar. Dies stört in diesem Fall nicht, da über die Kupplung bzw. über das Zahnrad ein gewisser Längenversatz ausgeglichen werden kann. Zwischen dem Zahnrad und dem Deckel sitzt im Hauptzusammenbau ein Lagerblock. Somit wird der Deckel eingeklemmt und am Verlassen seiner Position gehindert (siehe Abbildung 47). Um das Mitdrehen des Deckels zu verhindern, wird dieser zusätzlich eingeklebt.

Nachdem die Welle mit den Deckeln positioniert wurde kann am hinteren Wellenstummel die zweite Hälfte der Elastomer-Klauenkupplung montiert werden.

4.3.5.2.1. Welle1

Mit den auf der Welle auftretenden Kräften wurden die Auflagerkräfte bestimmt und der Wellendurchmesser auf Biegung sowie auf Torsion dimensioniert.

Zu diesem Zweck wird die Welle freigemacht und alle Kräfte eingezeichnet (siehe Abbildung 60).

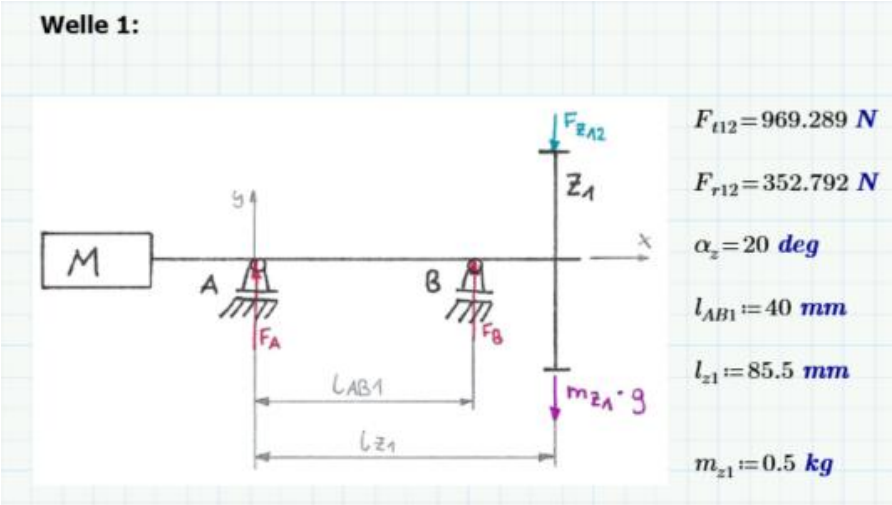


Abbildung 59: Auflagerreaktionen Welle 1

Anschließend werden Gleichungssysteme für Kräfte und Drehmomente aufgestellt und gelöst. Daraus ergeben sich die wirkenden Kräfte auf den Lagern (siehe Abbildung 6).

$$\Sigma M_{iA} = 0 \qquad \Sigma F_{iy} = 0$$

$$F_{B1} \cdot l_{AB1} - (F_{z12} + m_{z1} \cdot g) \cdot l_{z1} = 0 \qquad F_{A1} + F_{B1} - F_{z12} - m_{z1} \cdot g = 0$$

$$F_{B1} := \frac{(F_{z12} + m_{z1} \cdot g) \cdot l_{z1}}{l_{AB1}} = 2.215\ kN \qquad F_{A1} := -F_{B1} + F_{z12} + m_{z1} \cdot g = -1.179\ kN$$

Abbildung 60: Lagerkräfte

Auftretende Spannungen

Nach dieser Berechnung wird der Wellendurchmesser mit dem auftretenden Torsionsmoment vordimensioniert. Anschließend wir das größte auftretende Biegemoment bestimmt und die Biegespannung berechnet. Steigt die errechnete

Biegespannung über den zulässigen Wert (entnommen aus Tabellenbuch Roloff/Matek) ist der Durchmesser anzupassen.

Welle 1:

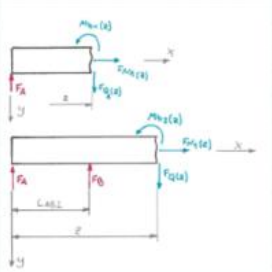
$d_{T1} = 15\ mm$ (auf Torison dimensioniert)

$d_{W1} := 15.5\ mm$ (gewählter Wellendurchmesser)

$\sigma_{bWN} := 550\ \frac{N}{mm^2}$ (Werkstoff 42CrMo4 RM TB 1-1)

$\nu := 1.25$ (Sicherheit für Biegung)

Bestimmung des maximalen Biegemoments:



1) $z_1 := l_{AB1} = 40\ mm$

$M_{b1} := |-F_{A1} \cdot z_1| = 47.156\ Nm$

2) $z_2 := l_{z1} = 85.5\ mm$

$M_{b2} := |-F_{A1} \cdot (z_2 - l_{AB1}) - F_{B1} \cdot z_2|$

$M_{b2} = 135.768\ Nm$

$M_{b1_max} :=$ if $M_{b1} > M_{b2}$
$\quad M_{b1}$
else
$\quad M_{b2}$

$M_{b1_max} = 135.768\ Nm$ Maximales Biegemoment Welle 1

Spannungswerte

$W_{x1} := \frac{d_{W1}^3 \cdot \pi}{32}$ $\sigma_{b1max} := \frac{M_{b1_max}}{W_{x1}} = 371.367\ \frac{N}{mm^2}$ $\sigma_{zul} := \frac{\sigma_{bWN}}{\nu} = 440\ \frac{N}{mm^2}$

Abbildung 61: Berechnung der auftretenden Spannungen (Datei: Berechnung_Getriebe)

Laut „Abbildung 61“ liegt der Wert der tatsächlich auftretenden Biegespannung unter dem maximal zulässigen Wert. Um Beschädigungen der Welle wie beispielsweise Bruch durch Ermüdung auszuschließen, wurde bei der Berechnung der zulässigen Spannung ein Sicherheitsfaktor von 1.25 miteingerechnet.

4.3.5.2.2. Zahnrad

Zur Auswahl der Zahnräder werden die berechneten Werte für Zähnezahlen, Teilkreisdurchmessern und Moduli aus der Datei „Berechnung_Getriebe“ entnommen. Mithilfe dieser Parameter wird im Internet nach geeigneten Normzahnrädern gesuchte. Die im Getriebe der Münzprägemaschine verwendeten Zahnräder stammen aus dem Online-Katalog des Normteileanbieters „Norelem“.
Um sicher zu gehen, dass die gewählten Zahnräder den auftretenden Belastungen standhalten, werden die Zahnfuß- und Zahnflankenspannungen der Evolventenverzahnung berechnet.

Diese Berechnungen basieren auf der Online-Datei „Stirnradvezahnungen“ von Prof. Dr. Ing Gruss und dem Buch Roloff / Matek Maschinenelemente.

Die Zahnräder der „Firma“ Norelem bestehen aus dem Werkstoff „C45“. Die zulässigen Spannungen werden somit mit diesem Werkstoffdaten berechnet.

4.3.5.2.3. Lager

Mit den nun bekannten Lagerkräften und einer gewünschten Lebensdauer kann die erforderliche Tragfähigkeit C der Lager berechnet werden.

Auswahl der passendenLager

Welle 1:

$F_{A1} = -1.179 \cdot 10^3\ N$ $\quad n_{An} = 380\ min^{-1}$

$F_{B1} = (2.215 \cdot 10^3)\ N$

Geforderte Lebensdauer: $L_{10h} := 500\ hr$

Lager A:

$p := 3$ Kugellagerexponent

$P := |F_{A1}| = 1.179\ kN$ Dynamisch äquivalente Belastung

$C := \sqrt[p]{\frac{L_{10h} \cdot n_{An}}{10^6}} \cdot P = 2.653\ kN$ Dynamische Tragzahl

Lager: 61804-2RS1 (SKF)

Abbildung 62: Berechnung Lagerauswahl

Mit diesem Kennwert und den gewünschten Wellenabmessungen wird im Online-Shop der Firma „SKF“ ein passendes Lager ausgesucht.

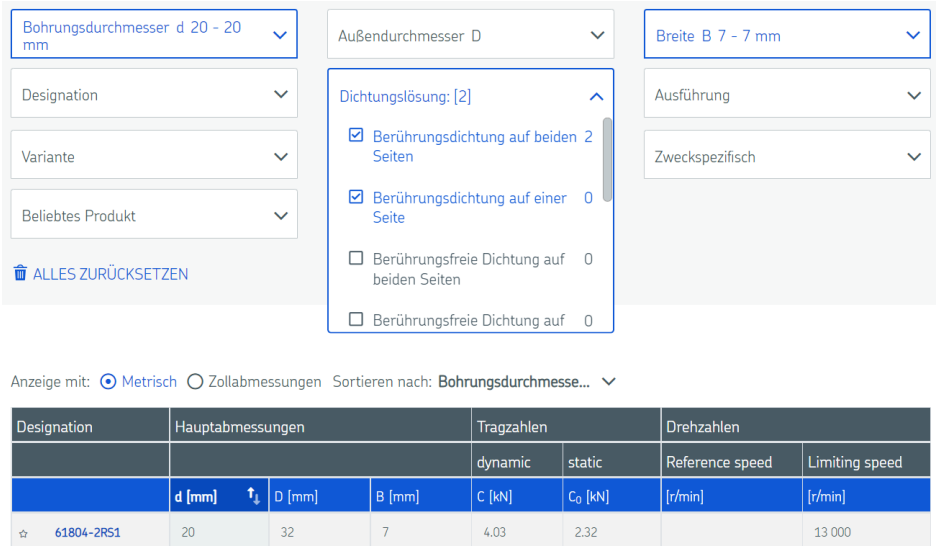


Abbildung 63: SKF-Shop (Online: https://www.skf.com/at)

4.3.5.2.4. Passfeder

Das vom Motor erzeugt Drehmoment wird mittels Passfeder, welche auf Flächenpressung beansprucht wird, auf das Zahnrad übertragen. Die Wahl der richtigen Passfeder basiert ebenfalls auf Berechnungen. Ein Beispiel dazu siehe „Abbildung *64*“.

Passfeder Rad 1:

Werkstoff: E295 $L := 16\ mm$ (Länge)

$b := 6\ mm$ (Breite) $n := 1$ (Anzahl)

$h := 6\ mm$ (Höhe) $\varphi := 1$ (= 1 bei n=1)

$d_{W1} = 15.5\ mm$ (Wellendurchmesser) $M_{T1max} = 30\ Nm$

$S_f := 1.3$ (gewünschte Sicherheit)

$h' := 0.4 \cdot h = 2.4\ mm$

$L' := L - b$

$Re := 295\ \frac{N}{mm^2}$

$p_{zul} := \frac{Re}{S_f} = 226.923\ \frac{N}{mm^2}$

$p := \frac{2 \cdot M_{T1max}}{d_{W1} \cdot L' \cdot h' \cdot n \cdot \varphi} = 161.29\ \frac{N}{mm^2}$

Abbildung 64: Berechnung Passfeder

4.3.6. Unterbaugruppe – Welle 2 ZSB

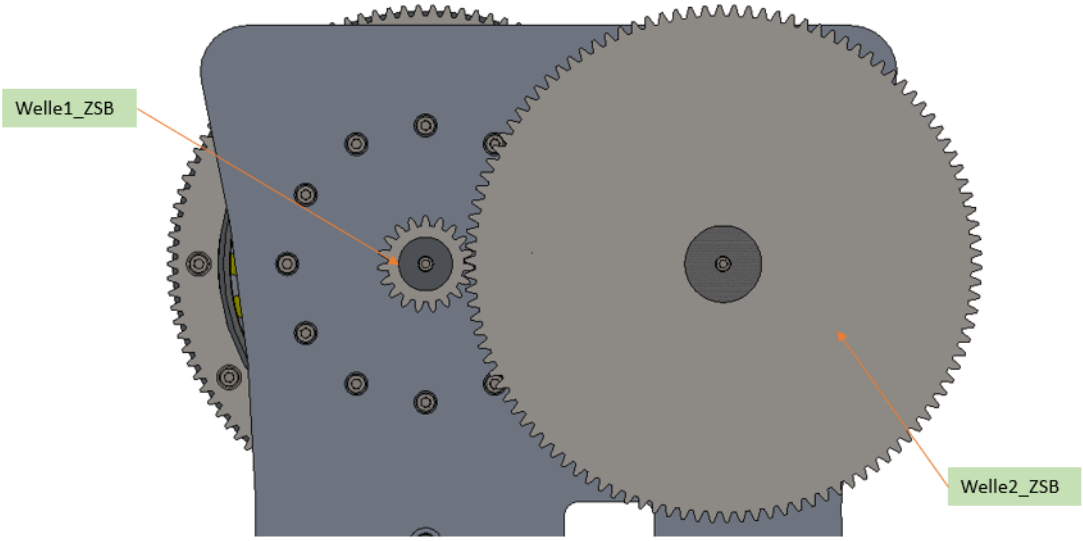


Abbildung 65: Erste Übersetzungsstufe

Auf der Hinterseite der Münzprägemaschine befindet sich die erste Übersetzungsstufe des Getriebes (siehe „Abbildung 65“). Hier wird das Moment von der Antriebswelle auf die zweite Welle übertragen. Durch die Übersetzung zwischen den Zahnrädern wird dabei das Drehmoment höher und die Drehzahl langsamer.

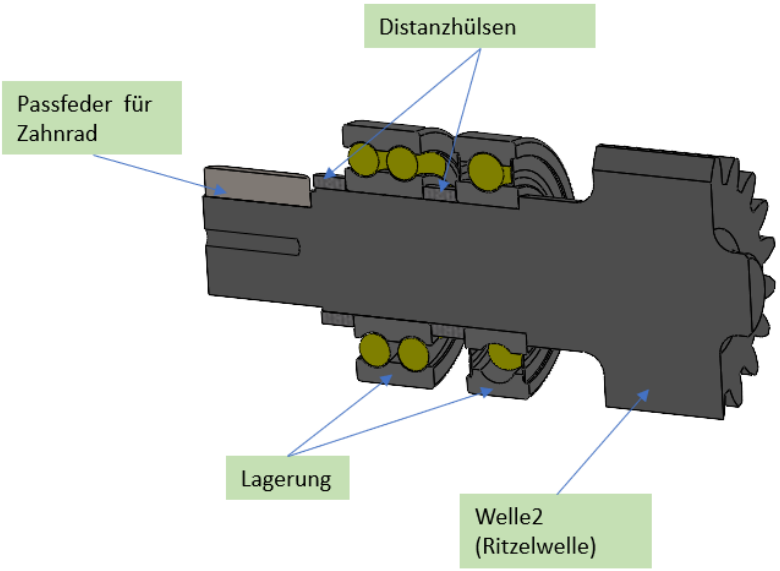


Abbildung 66: Unterbaugruppe Welle2_ZSB

Anordnung der Bauteile

Die beiden Lager werden von der Seite des großen Zahnrades (Absatz mit Passfeder) auf die Welle aufgeschoben. Der Innenring des einreihigen Kugellagers liegt am Wellenabsatz an und ist somit einseitig gegen Verrutschen gesichert. Um Abstand zu halten, wird zwischen den beiden Lagern eine Distanzhülse beigelegt.

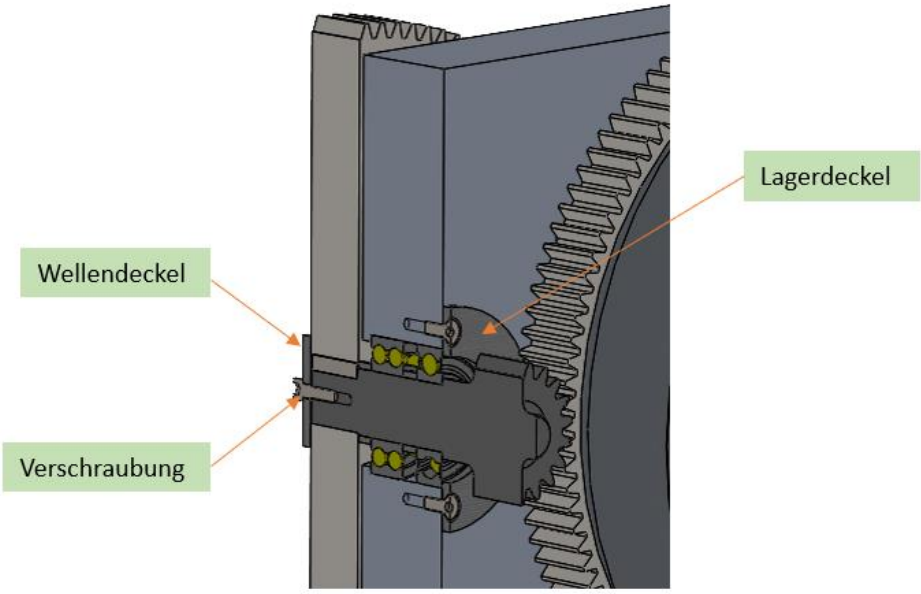


Abbildung 67: Welle2_ZSB (verbauter Zustand)

Im verbauten Zustand wird der Außenring des einreihigen Lagers durch einen Lagerdeckel in Position gehalten. Das andere Lager wird über den Wellendeckel, das Zahnrad und die hintere Distanzhülse fixiert. Der Wellendeckel ist mit der Ritzelwelle (Welle2) verschraubt.

In diesem Fall ist die Lagerung wieder „schwimmend“ ausgeführt. Aber auch hier ist der minimale Versatz in axialer Richtung kein Problem. Eine klassische Fest-Loslagerung würde einen höheren baulichen Aufwand mit sich bringen.

4.3.6.1. Ritzelwelle

Ein aufsteckbares Ritzel der Firma „Norelem“ würde der Belastung in der zweiten Getriebestufe nicht standhalten. Des Weiteren bräuchte man eine zweite am Umfang angeordnete Passfeder um das Torsionsmoment zu übertragen. Würde man eine zusätzliche Passfedernut in die Welle fräsen, wäre aufgrund der Querschnittminderung ein größerer Wellendurchmesser notwendig.
Ein Lösungsansatz für die Verwendung eines aufsteckbaren Ritzels, wäre das Zahnrad zu härten. Das Problem dabei ist jedoch, dass der Zahnradwerkstoff „C45“ dennoch nicht auf die erforderlichen Parameter modifiziert werden kann.

Um die zulässige Spannung zu erhöhen und das Moment einfach zu übertragen, wird die zweite Getriebewelle als Ritzelwelle ausgeführt. Der Werkstoff hierzu ist der Vergütungsstahl „42CrMo4“, welcher gut wärmebehandelbar ist.
Mittels „Nitrierverfahren“ zum Härten der Oberfläche können die gewünschten Festigkeitswerte erreicht werden.

Um den Durchmesser der Ritzelwelle zu bestimmen, wird identisch zur Welle 1 der Durchmesser grob auf Torsion vordimensioniert. Anschließend werden die auftretenden Auflagerkräfte bestimmt.
Da auf der zweiten Getriebewelle zwei Zahnräder sitzen, muss ein Gleichungssystem für ein dreidimensionales Kräftesystem aufgestellt und gelöst werden. Dazu wird zuerst die Welle freigemacht und die wirkenden Kräfte eingezeichnet (siehe Abbildung 68).

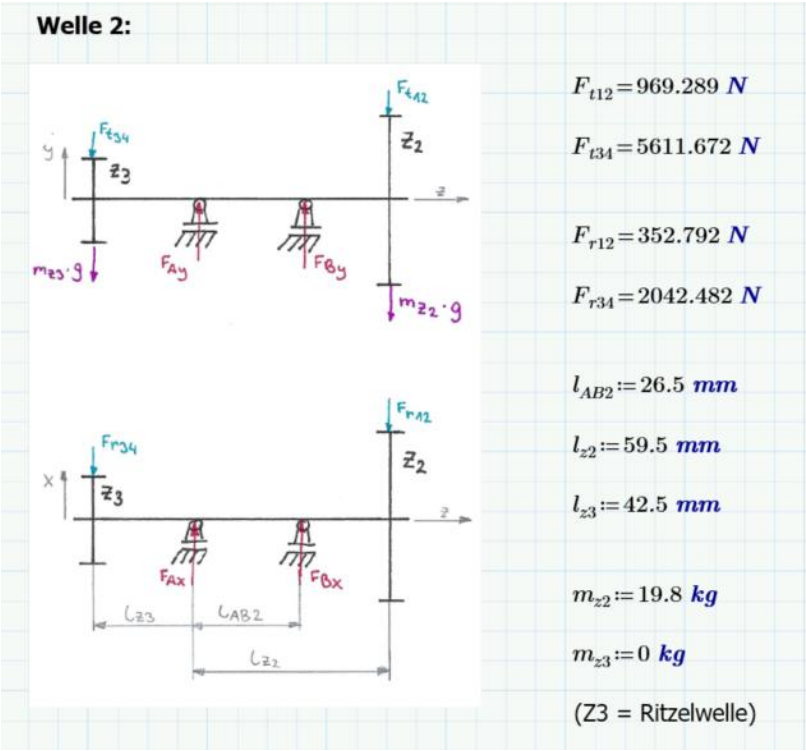


Abbildung 68: Auflagerreaktionen Welle2

Sind die Auflagerkräfte bekannt, kann das maximale Biegemoment ermittelt und der Wellendurchmesser auf Biegung überprüft werden (gleich wie bei Welle1; siehe Abbildung 61).

4.3.6.2. Lager und Passfeder

Die Passfeder und die erforderlichen Lager werden ebenfalls, wie bei der Unterbaugruppe „Welle1_ZSB“, berechnet aus dem Katalog ausgesucht und eingefügt.

4.3.7. Unterbaugruppe – Achse 3 ZSB

Das Drehmoment wird von der Ritzelwelle auf den Zahnkranz der Unterbaugruppe „Achse3_ZSB“, wie auf „Abbildung 68“ abgebildet, übertragen.

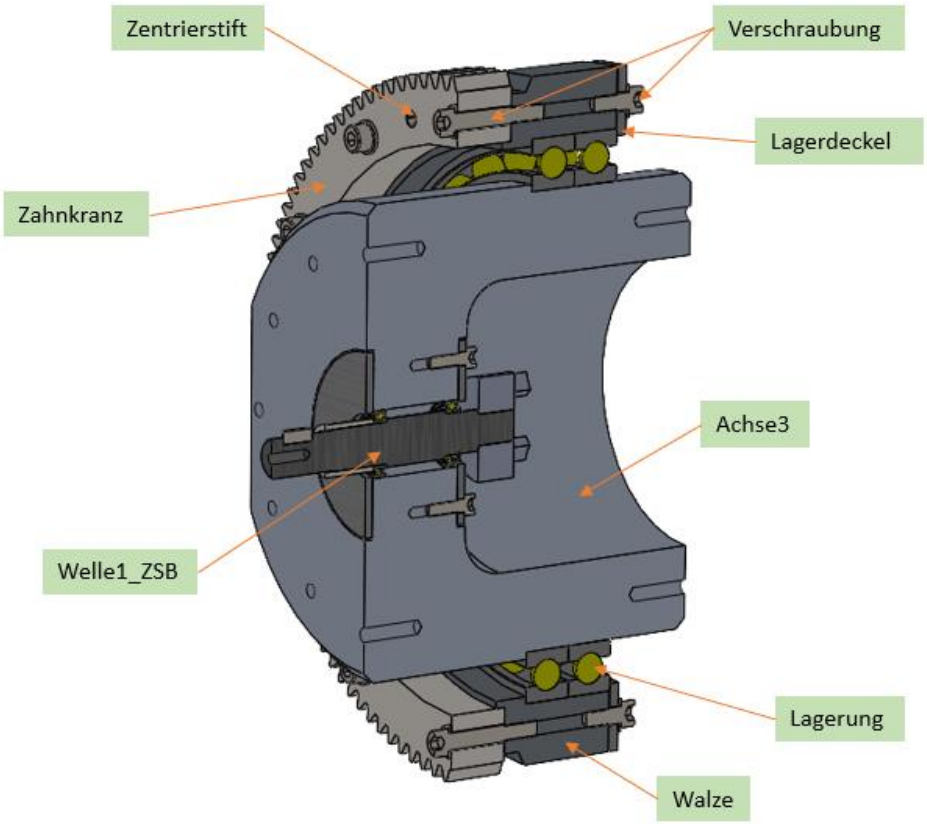


Abbildung 69: Unterbaugruppe "Welle3_ZSB"

Anordnung der Bauteile

Wie bereits beschrieben ist die Unterbaugruppe „Welle1_ZSB“ in das Innere der „Achse3“ eingesetzt.

In der Unterbaugruppe werden zuerst die beiden Lager in die Walze eingepresst und anschließend auf die Achse geschoben. Diese Lagerung ist dabei als Festlager ausgeführt. Auf der linken Seite werden die Rillenkugellager von einem Absatz auf der Welle und in der Walze gehalten. Der Außenring des rechten Lagers wird mittels eines Lagerdeckels gegen eine axiale Verschiebung gesichert. Um auch den Innenring in

Position zu halten, ist eine Distanzhülse zwischen Lager und dem vorderen Lagerblock beigelegt (siehe Abbildung 70). Dies passiert jedoch erst im Hauptzusammenbau.
Um die Drehbewegung des Walzvorgangs zu realisieren sind die Walzen mit einem Zahnkranz verschraubt und verstiftet.

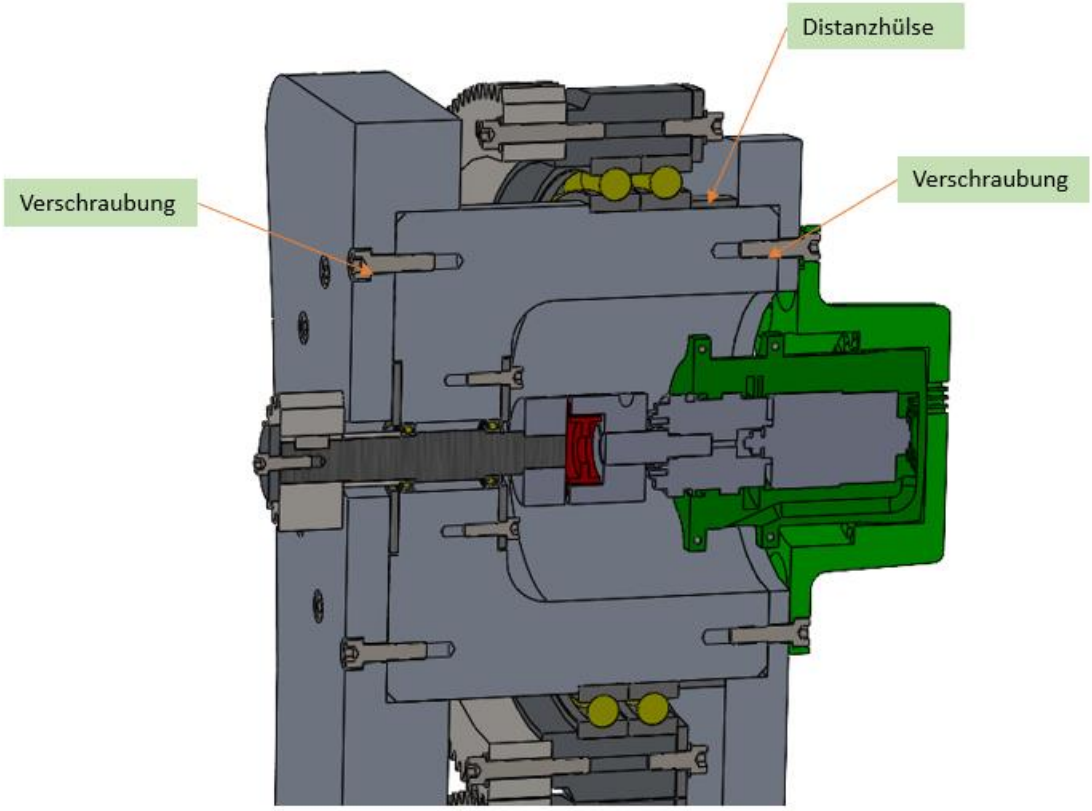


Abbildung 70: Achse3_ZSB (verbauter Zustand)

Die Unterbaugruppe „Achse3_ZSB“ wird mit dem vorderen sowie hinteren Lagerblock verschraubt.
An „Abbildung 70“ ist zu erkennen, dass diese Unterbaugruppe die komplexeste in der gesamten Konstruktion ist. Sie wird im Zusammenbau mit den Lagerblöcken, der Unterbaugruppe „Welle1_ZSB“ und dem Zusammenbau des Antriebs kombiniert.

4.3.7.1. Achse 3

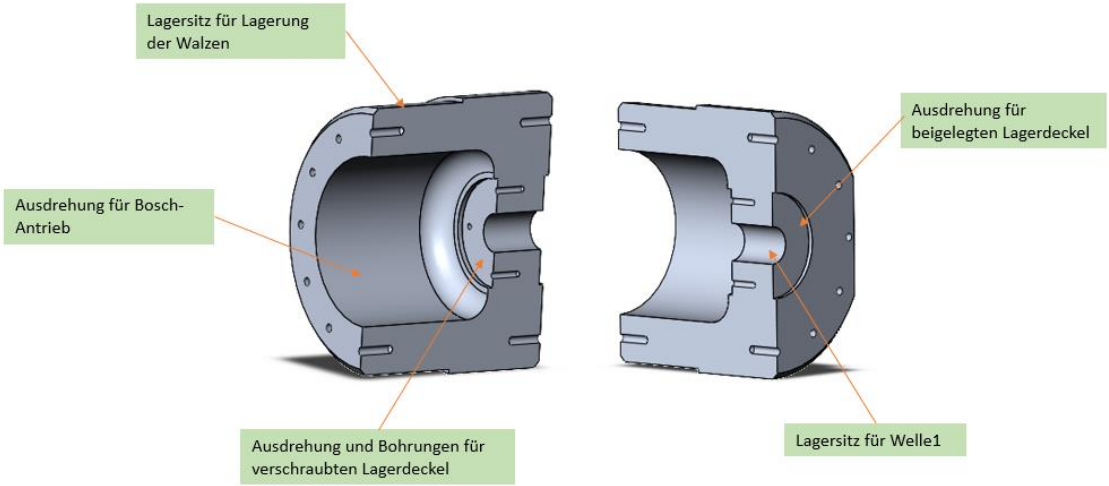


Abbildung 71: Achse3

Die Hauptaufgabe der „Achse3“ ist das Tragen der ersten Prägewalze. Hierzu befindet sich am kleineren Außendurchmesser eine Passung für einen genauen Lagersitz. Der Absatz am Außendurchmesser dient als Anschlag für die Positionierung des Lagers.
Im Inneren der Achse wird die Unterbaugruppe „Welle1_ZSB“ verbaut. Um den Motor des Akkuschraubers und das Planetengetriebe unterzubringen, ist die große Ausdrehung nötig. Über die aufgepressten Lager werden die Umformkräfte des Walzvorganges vollständig auf die Achse übertragen. Um die dabei entstehende Kerbwirkung auszugleichen, wurde die innere Ausdrehung mit einem großen Radius versehen. Mit dieser konstruktiven Lösung sollen Ermüdungserscheinungen, aufgrund der hohen Walzkräfte, vermieden werden.
Die Antriebswelle des Getriebes („Welle1“ aus Unterbaugruppe „Welle1_ZSB“) wird in der kleinen inneren Ausdrehung gelagert. Da hier die Wahl auf eine schwimmende Lagerung gefallen ist, kann die Ausdrehung durchgehend und ohne einen Absatz gefertigt werden. Der große Vorteil daran ist der geringere Fertigungsaufwand im Vergleich zu einer Fest-Loslagerung, die einen zusätzlichen Absatz benötigen würde.
Mit den stirnseitig angeordneten Bohrbildern wird die Achse im Zusammenbau mit dem vorderen sowie hinteren Lagerblock verschraubt. Die Flachstellen am

Außendurchmesser verhindern eine unerwünschte Verdrehung und gewährleisten eine exakte Positionierung (siehe Abbildung 72).

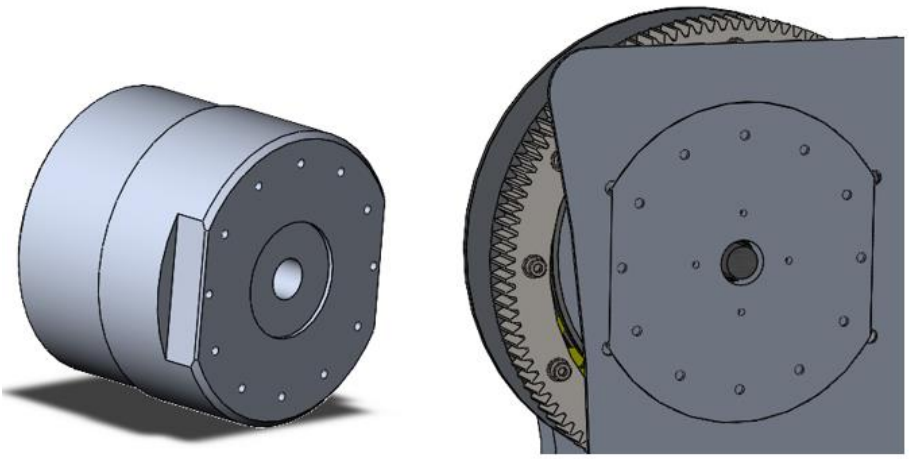


Abbildung 72: Flachstelle Achse 3

4.3.7.2. Lagerung

Um das passende Lager zu berechnen, wird eine Skizze mit allen wirkenden Kräften gezeichnet. Zur Bestimmung der statischen Tragfähigkeit des Lagers muss die Umformkraft des Walzprozesses sowie die, in der Verzahnung auftretenden Radialkräfte, miteinberechnet werden.

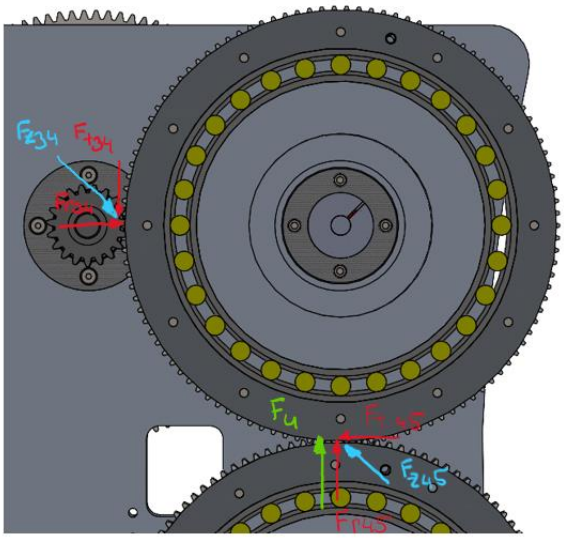


Abbildung 73: Achse3 - Lagerkräfte

$F_U = 170\ kN$ $\qquad F_{radial} := F_U + F_{r34} + F_{r45} = 174.085\ kN$

$F_{r34} = 2.042\ kN$ $\qquad F_{axial} := 0\ kN$

$F_{r45} = 2.042\ kN$

$e := \frac{F_{axial}}{F_{radial}} = 0$ $\qquad$ Wenn der Wert e<0,8 ist, dann P0 = Radialbeslastung

$P_0 := F_{radial} = 174.085\ kN$ $\qquad$ Statisch äquivalente Belastung des Lagers

$fs := 1.2$ $\qquad$ Statische Kennzahl

$C_0 := P_0 \cdot fs = 208.902\ kN$ $\qquad$ Statische Tragfähigkeit

$n_{Lager} := 2$ $\qquad$ Anzahl der verwendeten Lager

$C_{0_proLager} := \frac{C_0}{n_{Lager}} = 104.451\ kN$

Gewählte Lager: 2x 61844

Non-commercial use only

Abbildung 74: Achse3 - Lagerberechnung

Die Kräfte aus dem freigemachten System werden zu einer Ersatzkraft addiert. Durch diese erfährt das Lager eine radiale Belastung. Axiale Lagerbelastungen treten in diesem Fall keine auf. Dadurch ergibt sich, dass die statisch äquivalente Belastung des Lagers gleich der Radialkraft ist.
Mit diesem Wert und der statischen Kennzahl (typischen Wert angenommen) wird die erforderliche Tragfähigkeit errechnet. Um diese zu mindern, werden zwei Lager nebeneinander verbaut, was die Tragzahl halbiert. Anschließend sind aus dem Online-Katalog der Firma SKF die passenden Lager auszuwählen (siehe Abbildung 63).

4.3.7.3. Walze 3

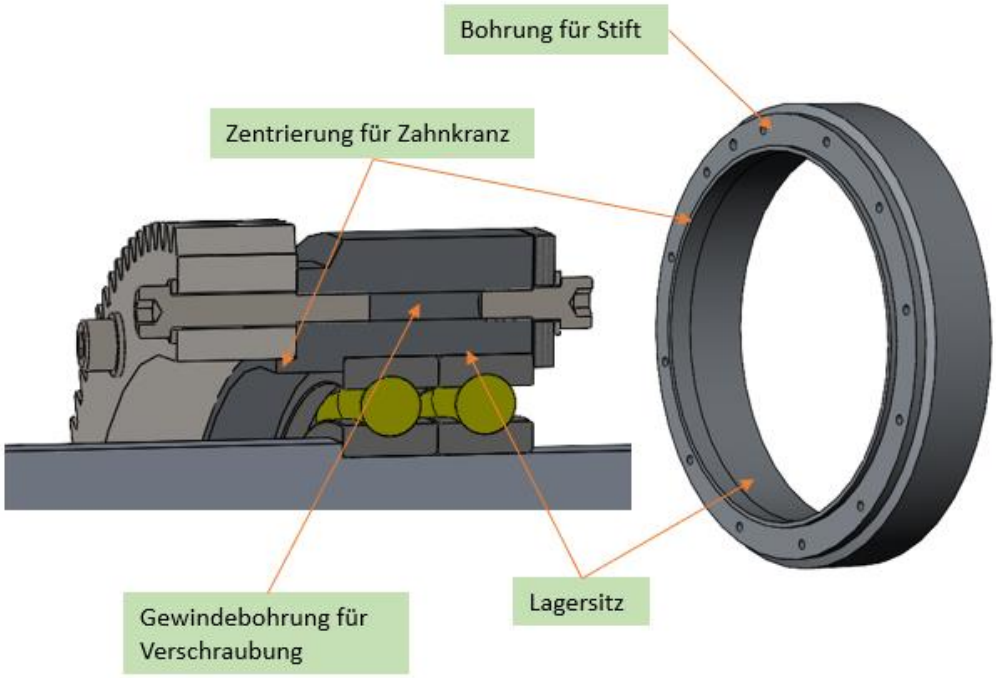


Abbildung 75: Achse3

Die Prägewalze wird aus einem Ring bestehend aus vergütetem Werkzeugstahl realisiert. Der Außendurchmesser ist dabei vom Teilkreisdurchmesser der Zahnkränze abhängig (Abbildung 76).

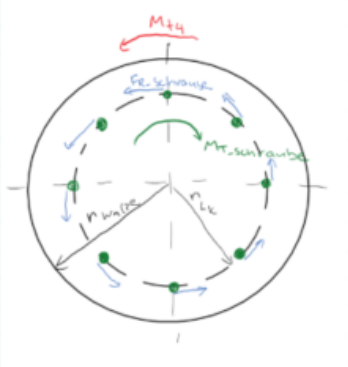
$h_1 := 0.8\ mm$	Breite des Walzspalts
$d_{Walze} := d_4 - h_1 = 329.2\ mm$	Walzendurchmesser

Abbildung 76: Berechnung Walzendurchmesser

Im Inneren wird der Passungssitz für die Lagerung ausgedreht. Um den Lagerdeckel zu montieren sind auf der Stirnseite der Walze 12 Gewindebohrungen vorgesehen. Die Bohrungen und Gewinde sind durchgängig ausgeführt, um eine Verschraubung des Zahnkranzes von der anderen Seite ebenfalls zu ermöglichen.

Über diese Schraubenverbindung wird das Drehmoment auf die Prägewalzen übertragen. Um das Versagen der Schrauben zu verhindern, wird die Verbindung nachgerechnet. Dabei ist darauf zu achten, dass das Drehmoment nur mittels Reibschluss zwischen den Kontaktflächen übertragen werden kann. Ansonsten würden die Schrauben über längere Zeit beschädigt werden.
Nach der Berechnung der erforderlichen Vorspannkraft, können die passenden Schrauben sowie das richtige Anzugsmoment bestimmt werden (siehe Abbildung 77).

Berechnung Verschraubung Walze - Zahnkranz



$r_{Walze} := 164.6\ mm$

$M_{T4max} = 814.488\ Nm$

$M_{T_Walze} := \mathrm{Ceil}(M_{T4max}, 10\ Nm) = 820\ Nm$

$d_{LK} := 287\ mm$

$M_{T_Walze} \cdot r_{Walze} = M_{T_Schrauben} \cdot \frac{d_{LK}}{2}$

$M_{T_Schrauben} := \frac{M_{T_Walze} \cdot r_{Walze}}{\frac{d_{LK}}{2}} = 940.571\ Nm$

Erforderlichen Vorspannung der Schrauben: (TBM S.320)

$F_{V_erf} = \frac{\varphi \cdot F_{R_Schraube}}{\mu}$

$N := 8$ Anzahl der Schrauben

$\mu := 0.12$ Reibkoeffizient

$\varphi := 2$ Sicherheit gegen Rutschen

$F_{R_Schraube} := \frac{M_{T_Schrauben}}{N \cdot \frac{d_{LK}}{2}} = 819.313\ N$ Reibkraft pro Schraube

$F_{Verf} := \frac{\varphi \cdot F_{R_Schraube}}{\mu} = 13.655\ kN$ Erfordderliche Vorspannkraft

$M_A := 32\ Nm$ Anzugsmoment (TBM S.232)

Aus TB "Erfahrungswerte Vorspannkraft" (TM S.230) --> gewählt M8

Abbildung 77: Berechnung Schraubenverbindung

Während des Prägevorgangs muss die aktuelle Lage des Motives auf der Walze stets bekannt sein. Deswegen ist eine genau Positionierung zwischen der Prägewalze und dem Zahnkranz erforderlich. Um diese zu gewährleisten, werden die beiden Bauteile über einen Spannstift zueinander ausgerichtet. Zusätzlich gibt es auf der Walze einen eigenen Zentrierabsatz, auf den der Zahnkranz aufgesetzt wird.
Die Motive werden in radialer Richtung am Umfang der Walze angeordnet.

4.3.7.4. Motive der Prägewalzen

Eines der Motive, welches in die Walze gefräst werden soll, stellt das Logo der HTBLuVA St. Pölten dar. Das Logo wird zuerst in SolidWorks entworfen und anschließend mit Maßen versehen, um die Größe genau zu definieren und eine bessere Vorstellung zur Größe zu bekommen. Das Logo ist schlicht gehalten, da die Fertigung ein großes Problem darstellen könnte.

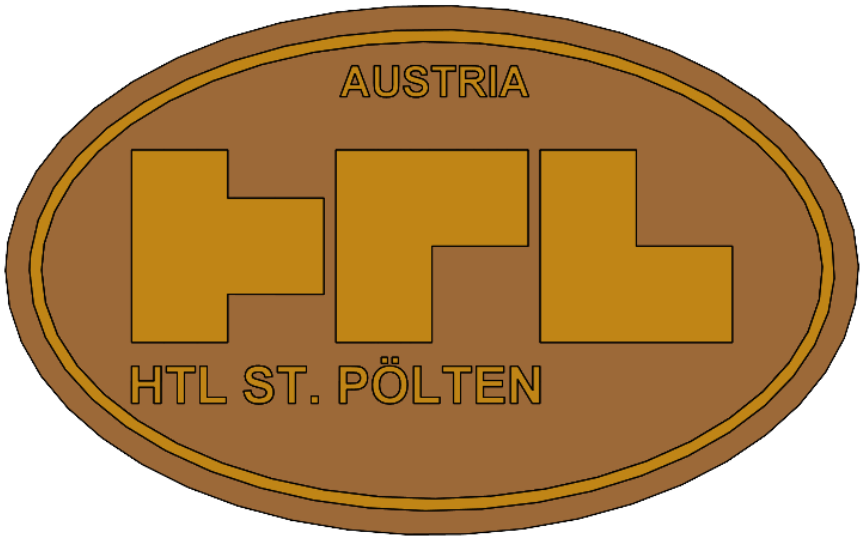


Abbildung 78: HTL St. Pölten Logo in SolidWorks

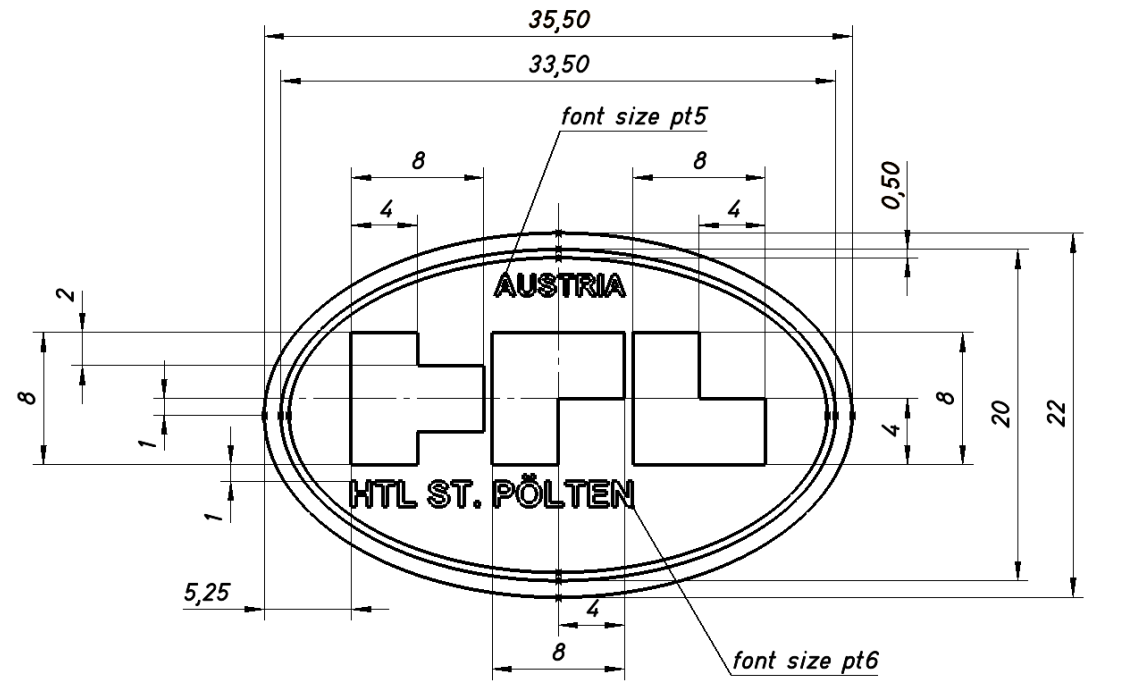


Abbildung 79: Bemaßung HTL Logo

Die zuvor im 3D angefertigte Skizze wird nun in das 3D-Modell der Walze übernommen. Durch Kopieren der Skizze ist dies einfach möglich.

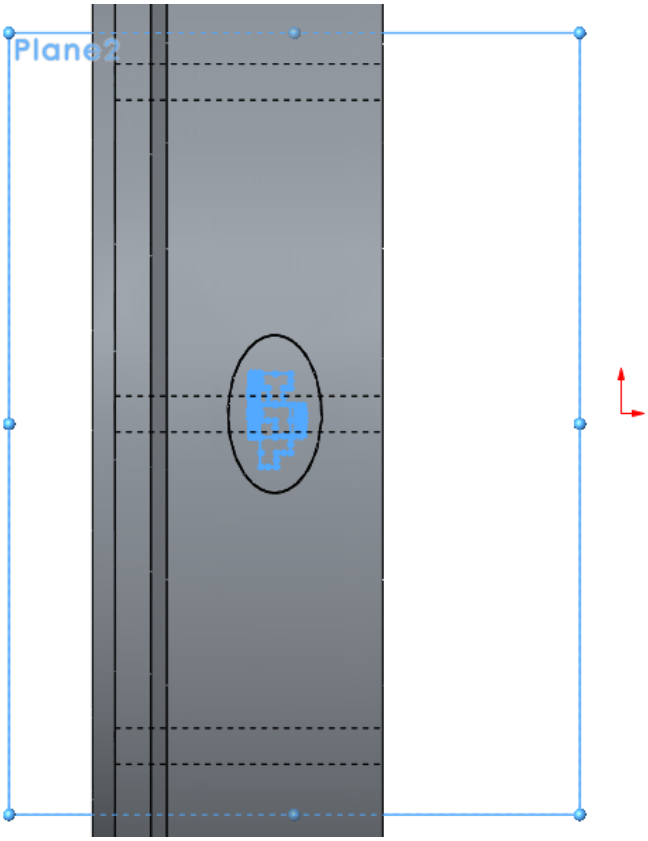


Abbildung 80: Skizze HTL Logo Walzen

Nachdem die Skizze richtig positioniert ist, kann diese mit Hilfe des Wrap-Features genau auf die runde Oberfläche der Walze übertragen werden. Da die Münze als Ausgangsmaterial eine Dicke von 1,7mm hat und durch einen Walzspalt von 0.8mm gezogen werden soll ergibt sich eine Motivtiefe von 0,45mm um ein beidseitiges aufprägen zu ermöglichen.

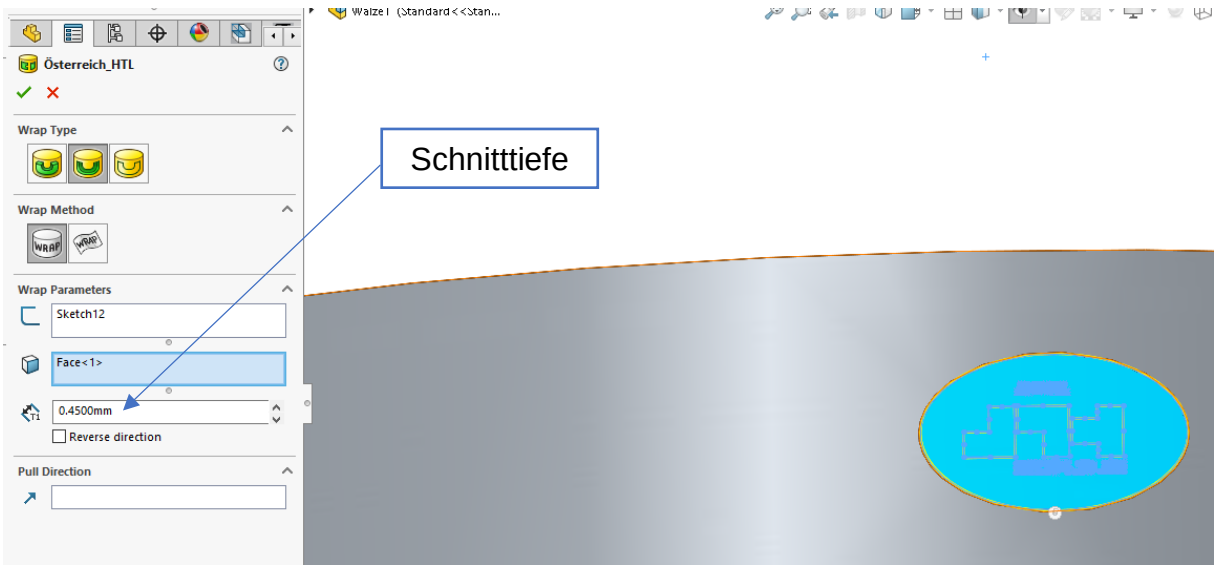


Abbildung 81: Wrap-Feature

Da diese Diplomarbeit als internationales Projekt durchgeführt wird, werden die weiteren fünf Logos von den anderen teilnehmenden Ländern entworfen. Diese landesspezifischen Motive werden ebenfalls auf die Walzen übertragen.

Die Motive der anderen Länder werden als SVG also Vektorgrafik hochgeladen. Mit dem Programm Inkscape (siehe Abbildung 79) können diese geöffnet werden, und in ein DXF-Format umgewandelt werden.

Abbildung 82: Inkscape

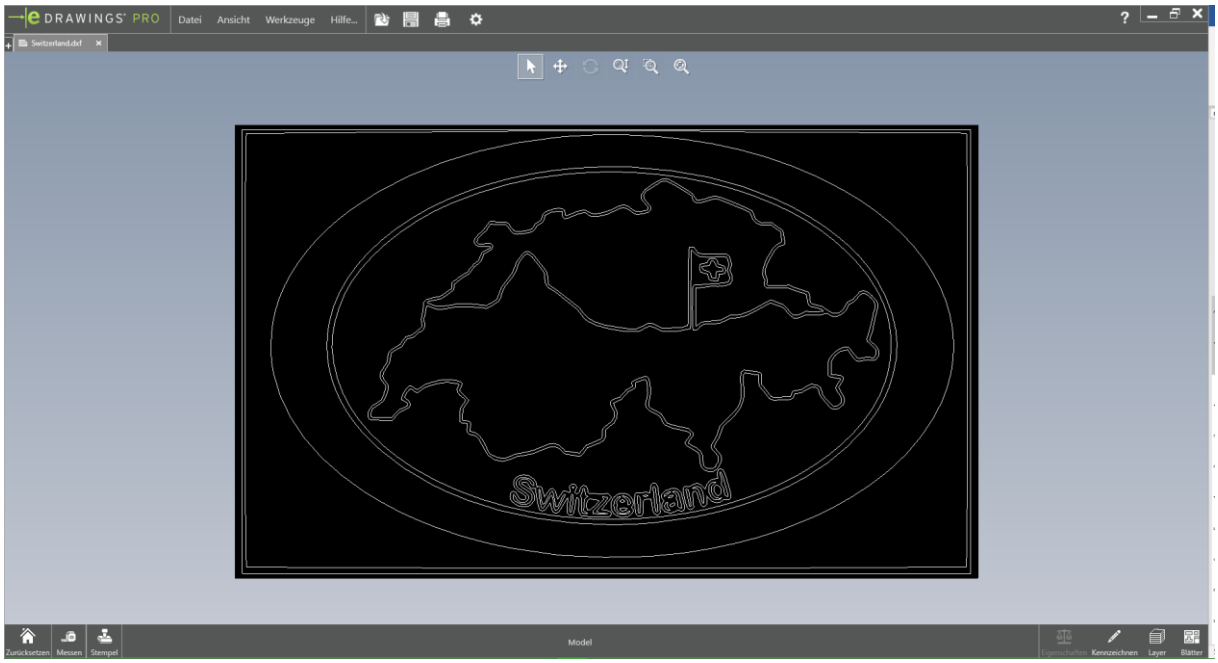


Abbildung 83: DXF-Format

Diese Zeichnung im DXF-Format wird wiederum als Skizze in SolidWorks eingefügt. Diese Skizze kann anschließend wie bereits zuvor beschrieben wieder mit dem Warp-Feature auf die Walzen übertragen werden.
Da einige Konturen zu komplex sind, um sie in einem Schritt auf die Walzen zu übertragen, müssen sie geteilt werden und in zwei Schritten in die Walzen geschnitten werden. Diese vorgangsweise stellt bei einzelnen Motiven eine große Herausforderung dar und gestaltet sich als sehr aufwändig.

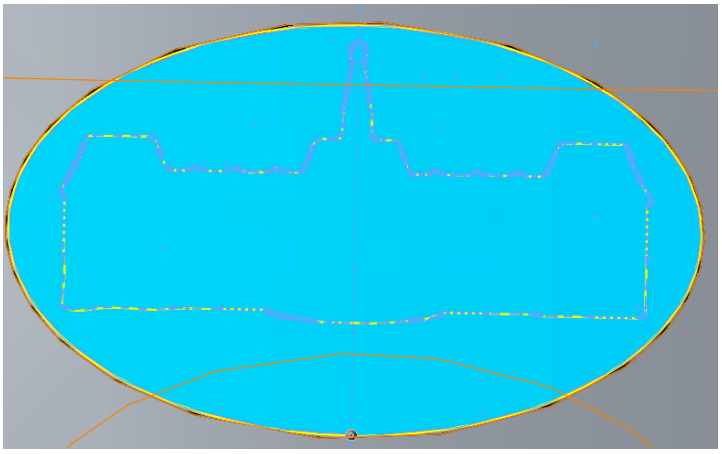


Abbildung 84: Luxemburg Schloss Kontur

Wie in Abbildung 84 zu sehen, wird zuerst die äußere Kontur des Logos in die Walze geschnitten. Im Anschluss daran, wird wie in Abbildung 85 gezeigt die innere Kontur, also die Fenster des Schlosses wieder aus dem Schnitt heraus extrudiert. Sowohl der subtraktive als auch der additive Vorgang könne mit dem Wrap-Feature durchgeführt werden (siehe Abbildung 85).

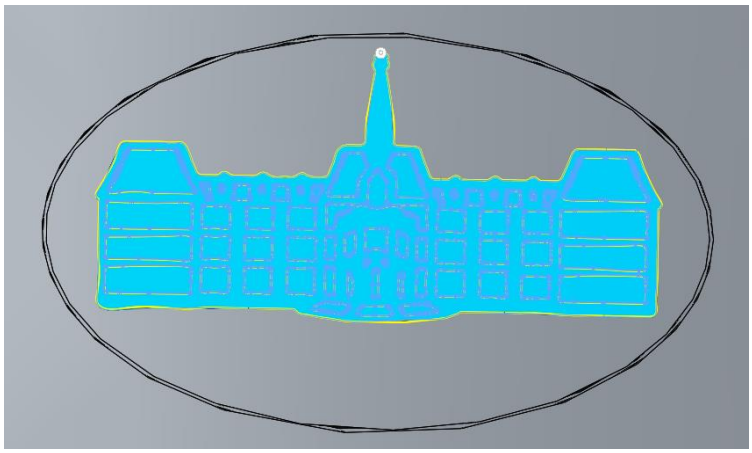


Abbildung 85: Luxemburg Schloss Fenster

Zuletzt wird die dazugehörige Schrift eingefügt und in die Walze geschnitten (siehe Abbildung 86).

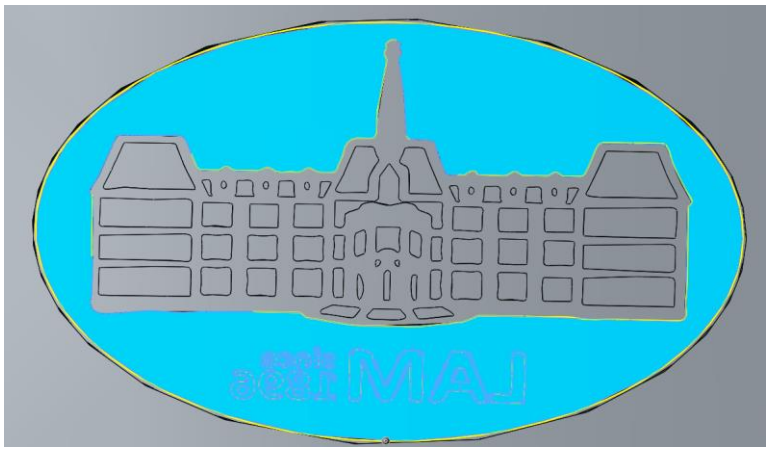


Abbildung 86: Luxemburg Schrift

Nach der Fertigstellung aller Motive auf den Walzen sieht diese wie in Abbildung 87 gezeigt aus.

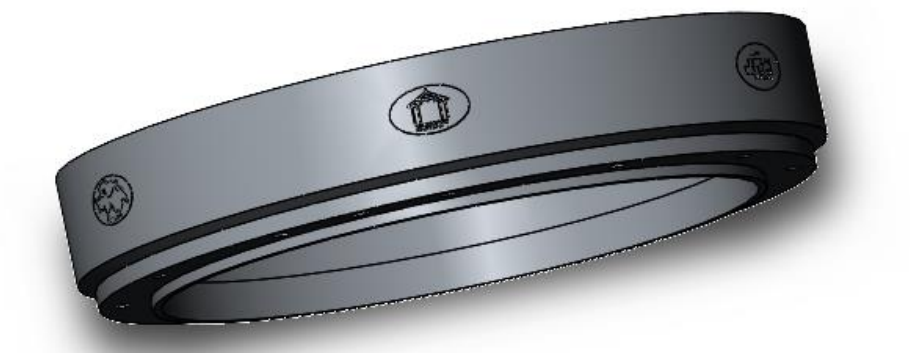


Abbildung 87: Walze 1 mit Motiven

Die Münzen sollen wie zuvor kurz angerissen, nicht nur einseitig, sondern beidseitig geprägt werden. Vom Projektteam wurde entschieden auf der Münzrückseite ein einheitliches Motiv zu verwenden. Um dieses Logo auf die Walze zu übertragen, müssen die Konturen des Logos in SolidWorks als Skizze nachgezogen werden (siehe Abbildung 88). Dazu fügt man in SolidWorks auf einer Ebene das Bild ein, welches man abpausen möchte und legt danach eine Skizze darüber und zieht die Konturen des Bilds mit normalen Skizzenlinien nach.

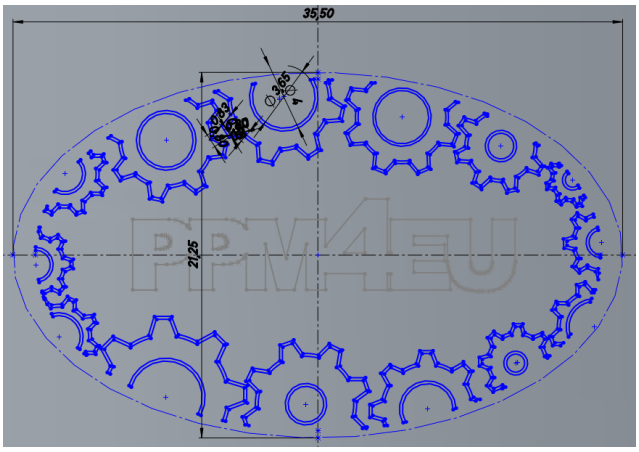


Abbildung 88: Nachziehen der Motive des PPM4EU Logos

Das fertige Logo soll 6-mal auf die Walze 2 übertragen werden, um zu jedem anderen Motiv auf der Walze 1 ein Motiv für die zweite Seite der Münze zu haben, andernfalls wären zwei voneinander unabhängige Getriebe bzw. Antriebe benötigt worden, um diesen Vorgang nur mit einem Motiv auf der zweiten Walze zu verwirklichen.

Abbildung 89: PPM4EU Logo auf der Walze 2

4.3.7.5. Zahnkranz

Da Zahnkränze in dieser Größe sehr teuer sind, wird ein Normteil gewählt und dieses an die Einbauerfordernisse angepasst

Hierbei fällt, wie auch bei den anderen Zahnrädern, die Wahl auf den Normteilkatalog der Firma Norelem.

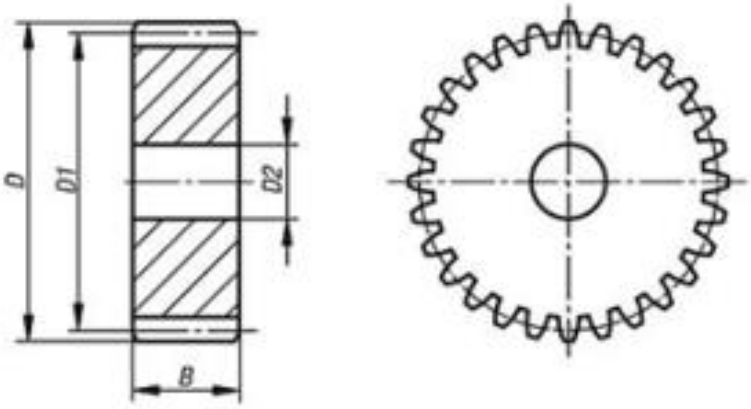


Abbildung 90: Zahnkranz (https://www.norelem.at)

Für die Fertigung des Zahnkranzes wird ein Normzahnrad der Firma Norelem gekauft und innen auf den erforderlichen Durchmesser, unter Berücksichtigung der erforderlichen Passung und der, für einen gleichmäßigen Wälzvorgang, erforderlichen Konzentrizität ausgedreht.

Danach wird das passende Bohrbild für die Verschraubung mit der Walze, sowie die Bohrung für den Spannstift eingebracht.

Um sicherzustellen, dass die beiden Zahnkränze zusammenpassen, muss das erste Bohrbild zur Mittellinie des Zahnfußes referenziert werden (siehe Abbildung 91).

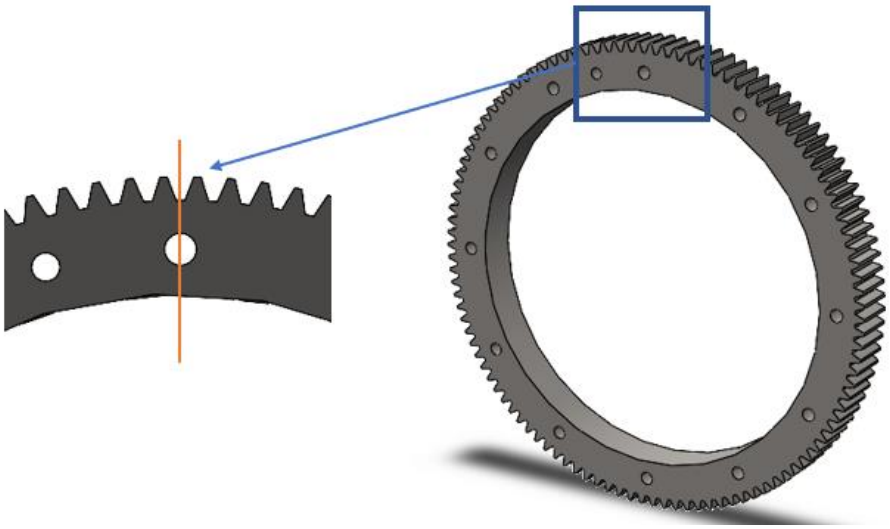


Abbildung 91: Referenzieren der Bohrung

4.3.8. Unterbaugruppe – Achse 4 ZSB

Die Unterbaugruppe „Achse4_ZSB“ ist nahezu ident der Unterbaugruppe „Achse3_ZSB“ ausgeführt.

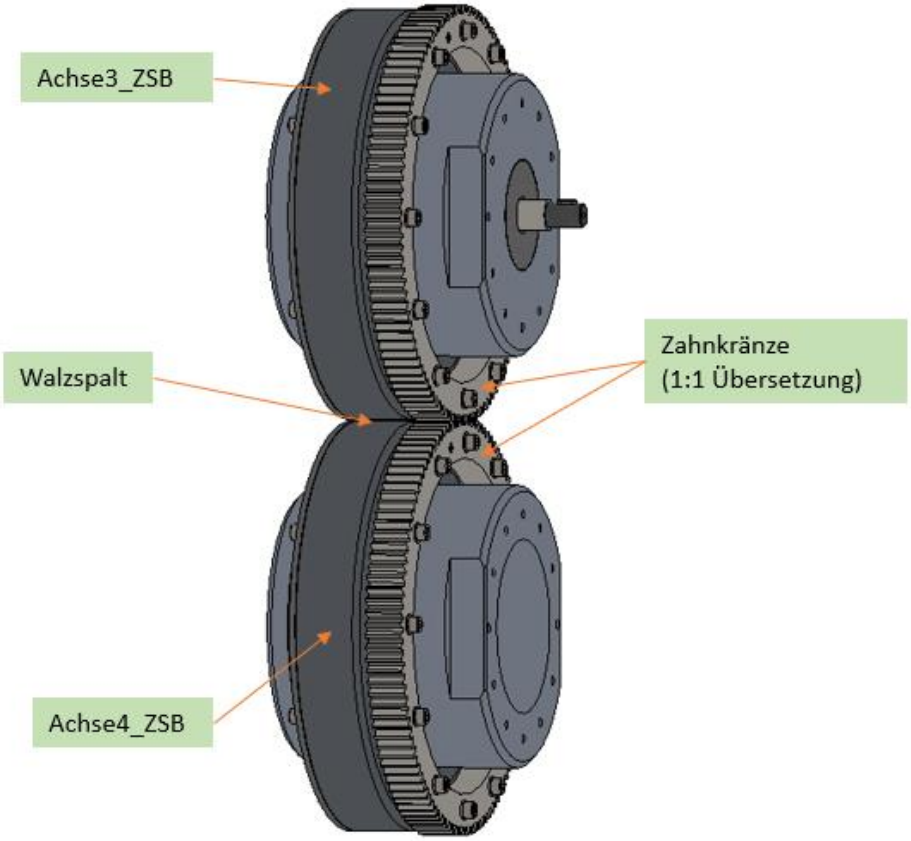


Abbildung 92: Achse3_ZSB & und Achse4_ZSB

Während des Prägevorganges müssen sich die Walzen in entgegengesetzter Richtung drehen und dabei über dasselbe Drehmoment sowie dieselbe Drehzahl verfügen. Hierzu haben die Zahnkränze eine Übersetzung von 1:1 (siehe Abbildung 92).

Die einzigen Unterschiede zur „Achse3_ZSB“ finden sich in der stillstehenden Achse.

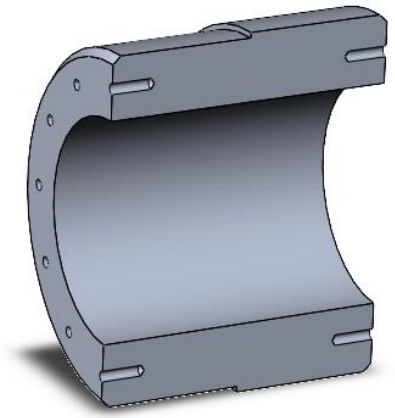


Abbildung 93: Achse4

Im Gegensatz zur „Achse3“ wird das Innere der „Achse4“ auf einen durchgehenden Durchmesser ausgedreht.
Wird nur die Funktion des Getriebes betrachtet, müsste die Achse nicht ausgedreht werden. In diesem Fall könnte einiges an Arbeits- bzw. Zerspanungszeit eingespart werden. Dennoch wird eine innen hohle Achse vorgesehen, da der entstehende Platz anderwärtig genutzt werden kann. Es können beispielsweise Komponenten der elektronischen Steuerung in dem Hohlraum platziert werden.

Der zweite Unterschied zur Baugruppe „Achse3_ZSB“ befindet sich in der zweiten Walze.
Um eine Münze auf beiden Seiten zu prägen, müssen die Motive der Walzen genau zueinander passen. Werden jedoch auf beiden Walzen und auf beiden Zahnkränzen die Bohrbilder gleich gefertigt, kommt es zu einer Überschneidung der Zähne (siehe Abbildung 94).

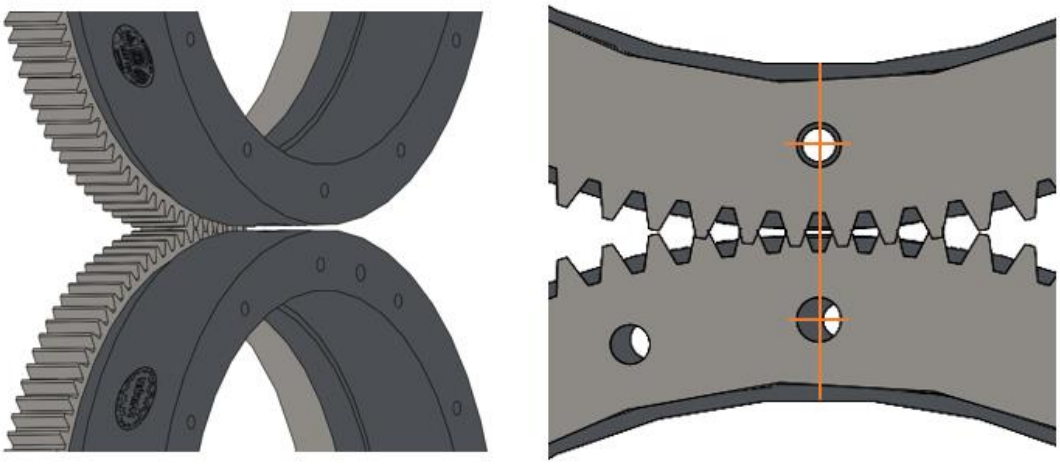


Abbildung 94: Zahnüberschneidung

Um dieses Problem zu beheben, muss das untere Bohrbild, um den Winkel der halbe Zahnteilung, verdreht werden. Dies könnte entweder auf der „Walze2“ oder auf dem Zahnkranz geschehen. Da es naheliegender ist zwei identische Zahnkränze zu fertigen, wurde das Bohrbild auf der Walze verdreht.

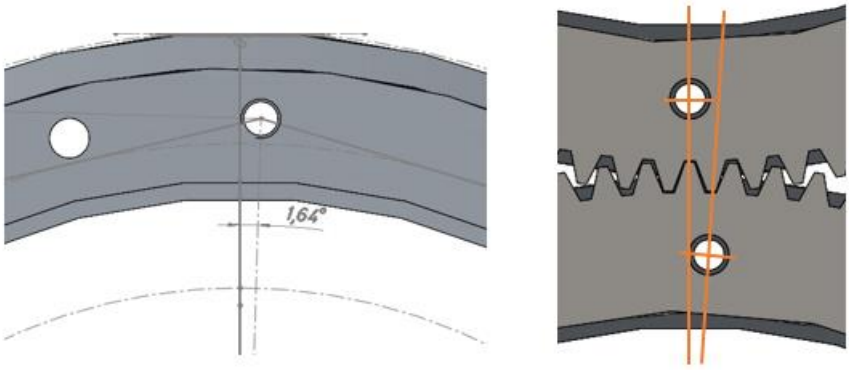


Abbildung 95: Verdrehtes Bohrbild

Somit stimmt der Zahneingriff und die Motive passen zueinander.

4.3.9. Hinterer Lagerblock

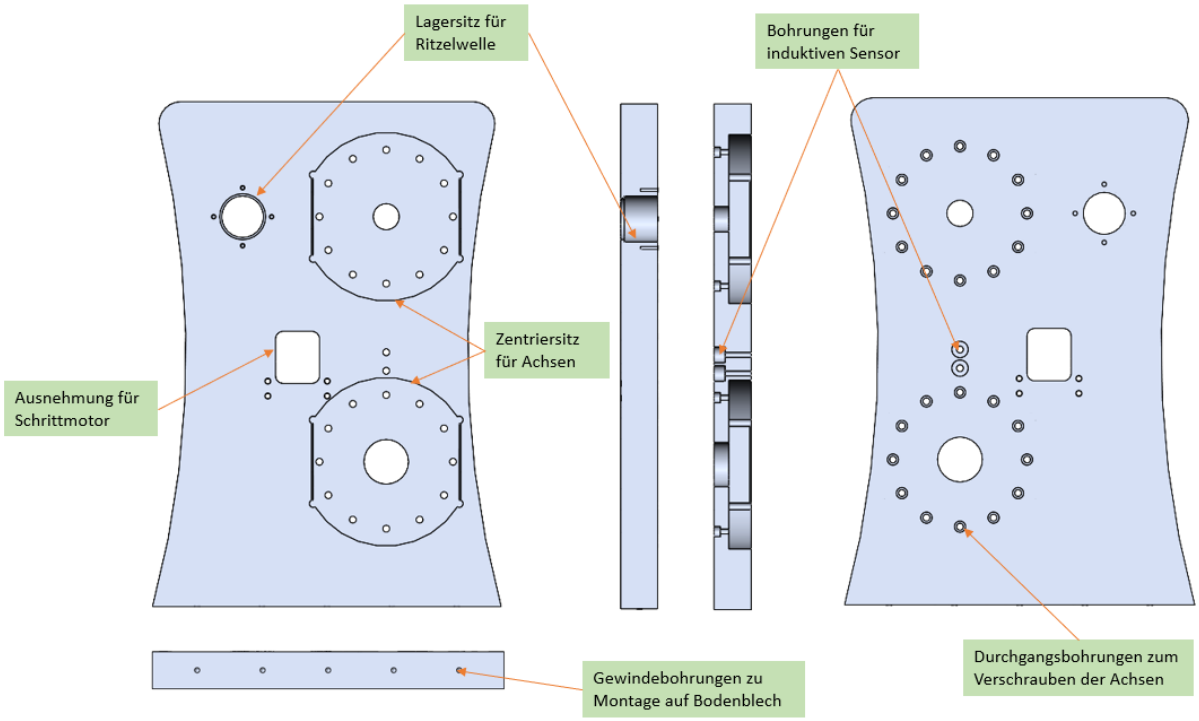


Abbildung 96: Hinterer Lagerblock

Auf der Unterseite des Lagerblocks befinden sich fünf Gewindebohrungen, welche zur Montage auf dem Bodenblech gedacht sind.

Wie bereits im Kapitel (4.4.7.1) beschrieben, verfügen die Zentriersitze für die Achsen über Flachstellen. Diese verhindern das Verdrehen der Achsen und sichern somit die richtige Position. An den Übergängen zwischen den Flachstellen und dem Radius befinden sich vier Bohrungen. Dies sind Ausnehmungen zum Fräsen, da mit einem Schaftfräser keine Innenecken gefräst werden können.

Die zwei Bohrbilder beinhalten insgesamt 24 plangesenkte Durchgangsbohrungen. Diese dienen zum Verschrauben der Achsen mit dem hinteren Lagerblock.

Im Lagersitz neben den Zentriersitzen wird die Ritzelwelle (Welle2) schwimmend gelagert.

Für die Münzzuführung ist ein Schrittmotor notwendig, welcher in der Ausnehmung darunter montiert werden kann (siehe Kapitel 5.3.2.3).

Des Weiteren befinden sich Bohrungen für die Unterbringung von induktiven Sensoren im Lagerblock. Diese sind Teil der Steuerung und werden im Kapitel 7.2.2 genauer beschrieben.
Die geschwungene Außenkontur trägt nicht zur Funktion des Bauteils bei, sondern verbessert „nur“ das Erscheinungsbild.

4.3.10. Vorderer Lagerblock

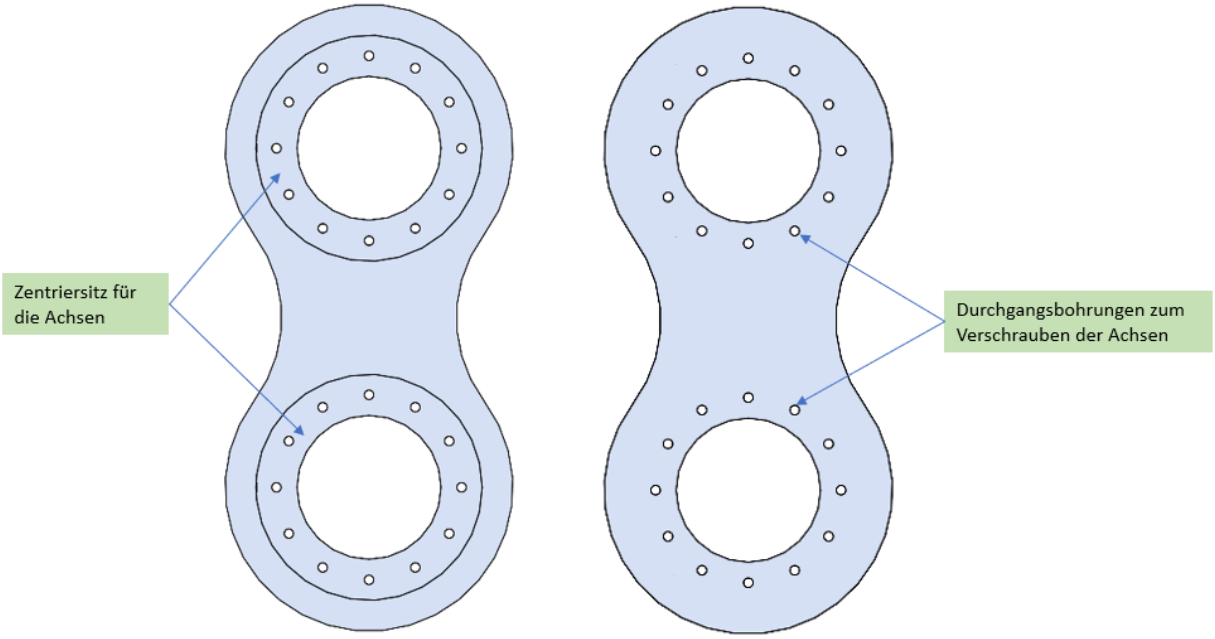


Abbildung 97: Lagerblock vorne

Die Bauteilbezeichnung „Lagerblock vorne“ kommt aus dem ersten Konzept der Münzprägemaschine. Hier gab es noch keine stillstehenden Achsen, sondern umlaufende Wellen, welche in beiden Lagerblöcken gelagert wurden. In der ausgeführten Konstruktion hat dieses Bauteil keine Lagerungsaufgaben mehr, dennoch wurde der Name übernommen.
Die Funktion des vorderen Lagerblocks ist, die beim Prägen auftretenden Kräfte aufzunehmen. Wäre dieser nicht vorhanden, müsste die gesamte Last vom hinteren Lagerblock übernommen werden. Dabei würde in der Achse, in den Schrauben und in

der Einspannung ein Biegemoment wirken, welches zu einer großen Biegespannung führen würde (siehe Abbildung 98).

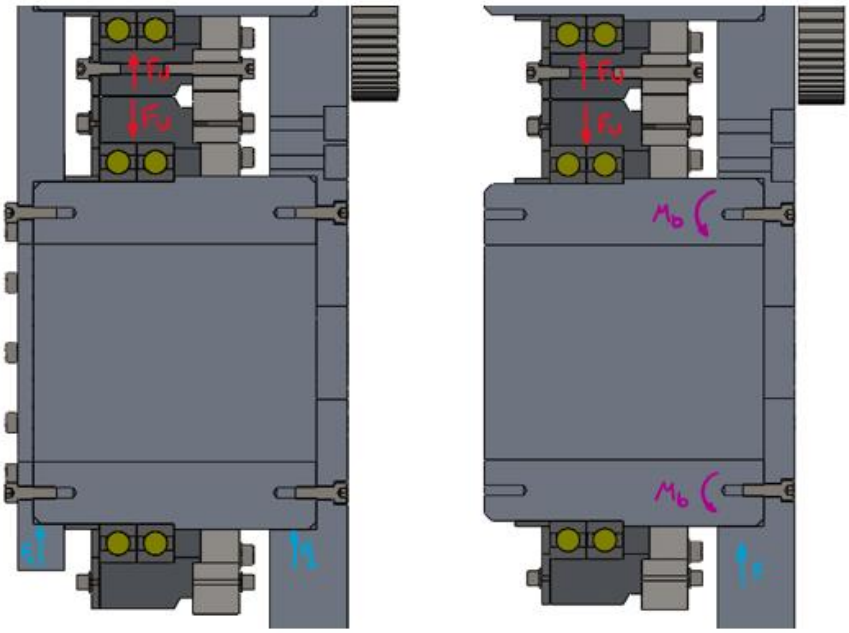


Abbildung 98: Vorderer Lagerblock – Funktion 1

Davon abgesehen hält der vordere Lagerblock über die Distanzhülsen die Lager und somit die Walzen und die Zahnkränze in der richtigen Position (siehe Abbildung 101).

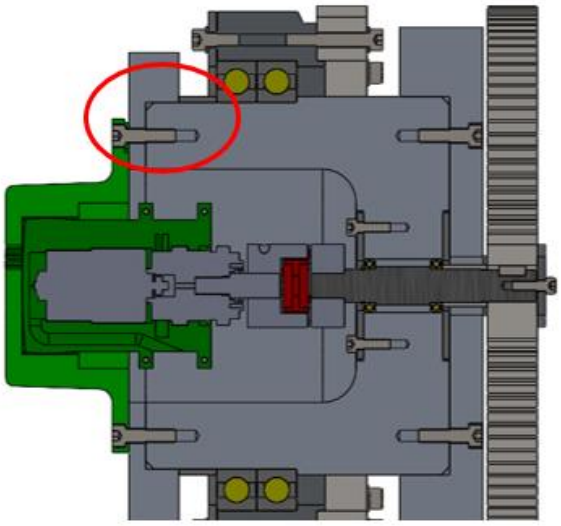


Abbildung 99: Lagerblock vorne - Funktion 2

Da der vordere Lagerblock über die Achsen mit dem hintern Lagerblock verschraubt ist, ist es nicht notwendig, dass dieser bis zur Bodenplatte reicht. Aus optischen Gründen wird der Lagerblock „fliegend“ und die Außenkontur geschwungen ausgeführt.

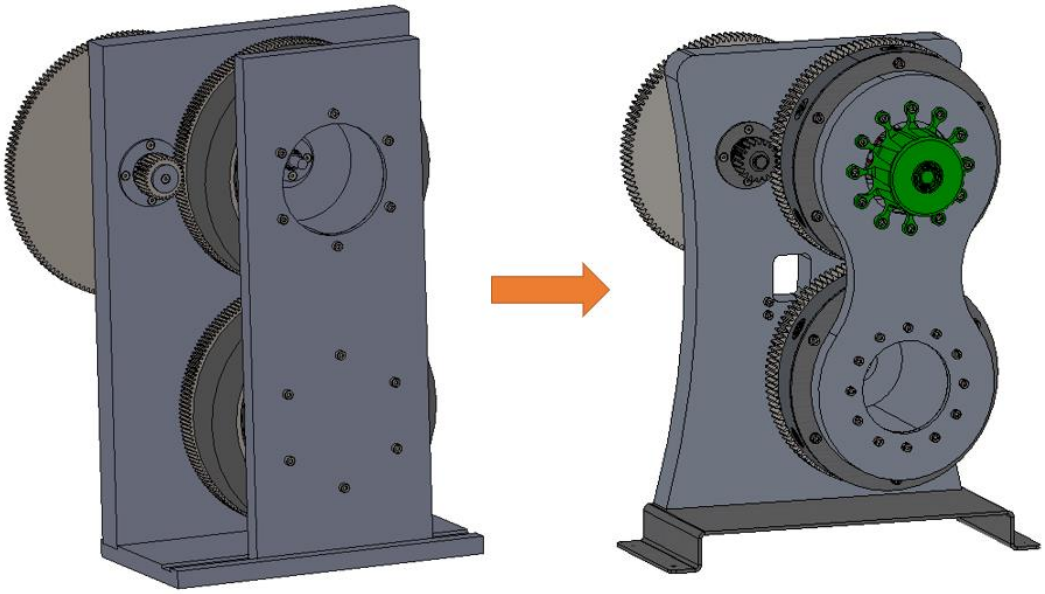


Abbildung 100: Erstes und finales Design der Lagerblöcke

4.4. Zusammenbaureihenfolge

Im Bodenblech befinden sich vier Bohrungen welche zur Montage auf ein Grundgestell oder auf eine Bodenplatte gedacht sind (siehe Abbildung 101). Das Bodenblech dient nur zur Montage und wird später wieder entfernt.

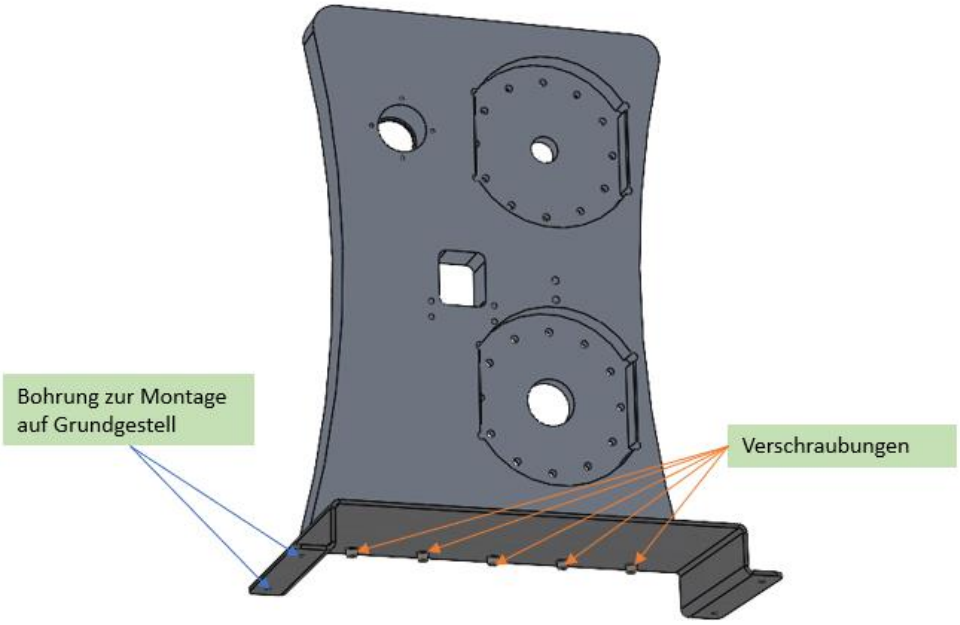


Abbildung 101: Montage Schritt 1

Im nächsten Schritt werden die beiden Unterbaugruppen „Achse3_ZSB“ und „Achse4_ZSB“ montiert. Diese Baugruppen sind die Komplexesten und zugleich die Wichtigsten, da sie das Walzwerk beinhalten.

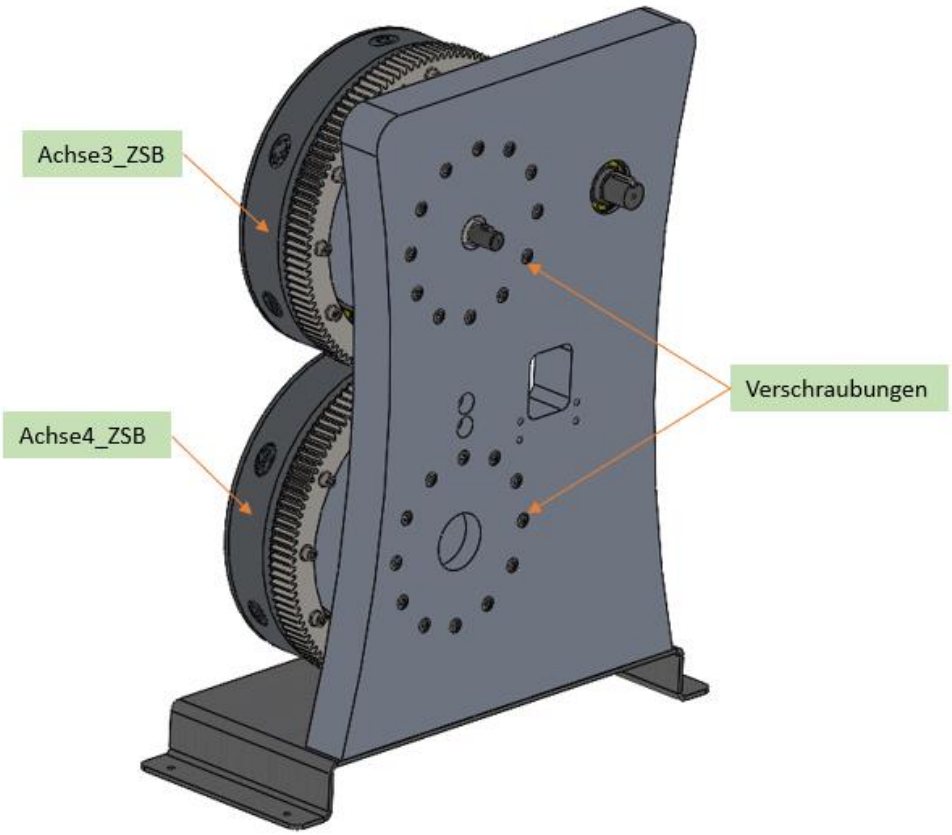


Abbildung 102: Montage Schritt 2

Ist dies geschehen, kann der Zusammenbau der „Welle2_ZSB“ im hinteren Lagerblock eingebaut und mithilfe der Lagerdeckel fixiert werden (siehe Abbildung 103)

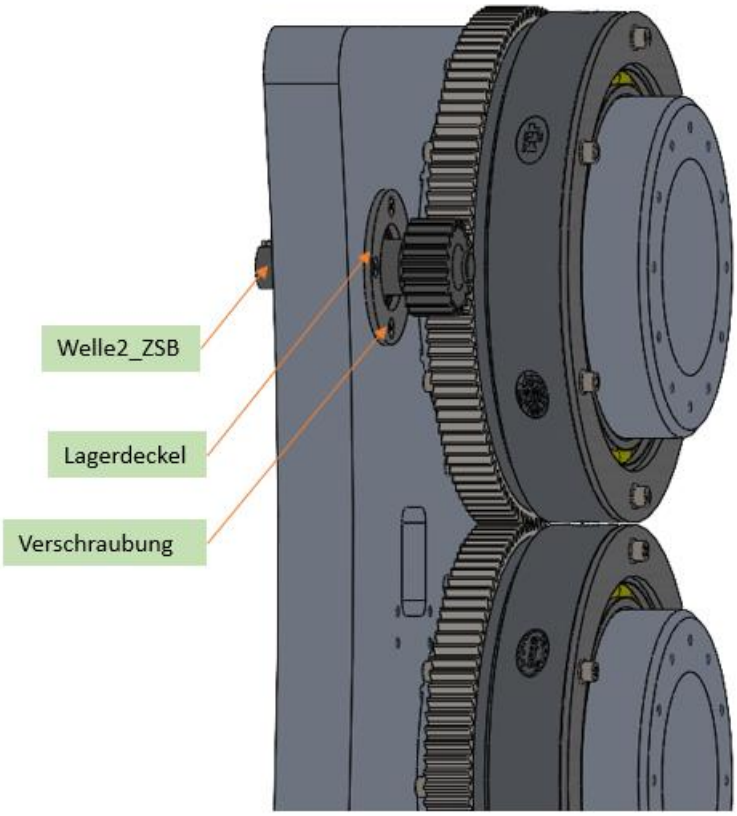


Abbildung 103: Montage Schritt 3

Als nächsten können die beiden Zahnräder der ersten Getriebestufe aufgesteckt und mit den Wellendeckeln fixiert werden (siehe Abbildung 104).

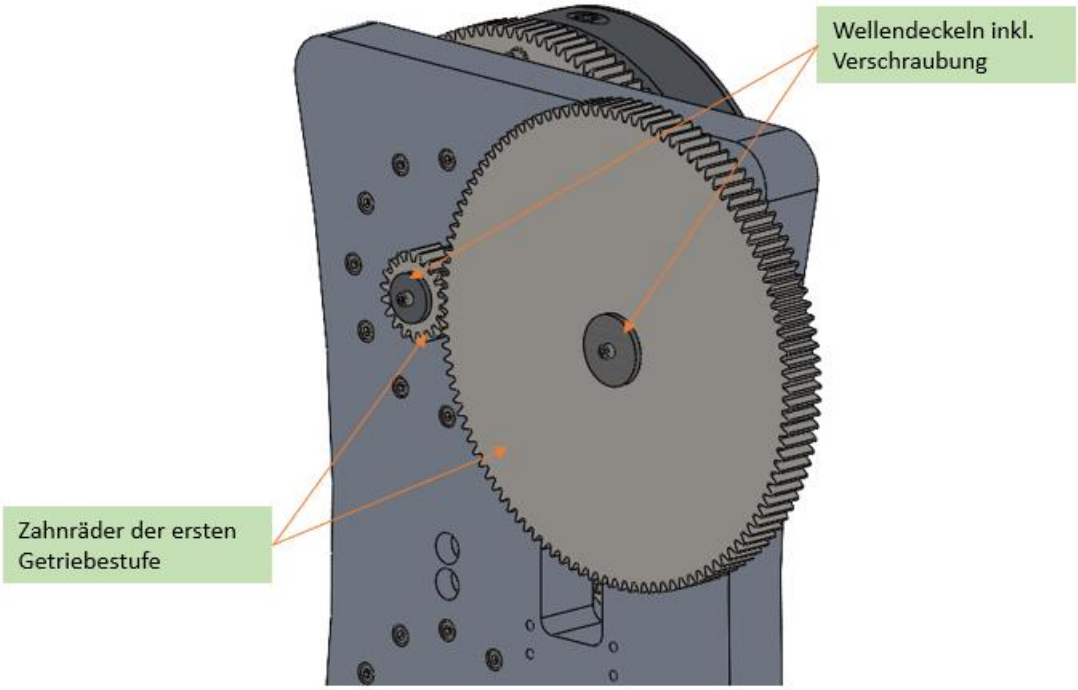


Abbildung 104: Montage Schritt 4

Anschließend werden die beiden Distanzhülsen beigelegt und der vordere Lagerblock mit den Achsen verschraubt. Bei der oberen Achse werden etwas längere Schrauben verwendet. Dies ermöglicht, dass die Unterbaugruppe „Antrieb_ZSB“ gleich mitgeschraubt werden kann (siehe Abbildung 105).

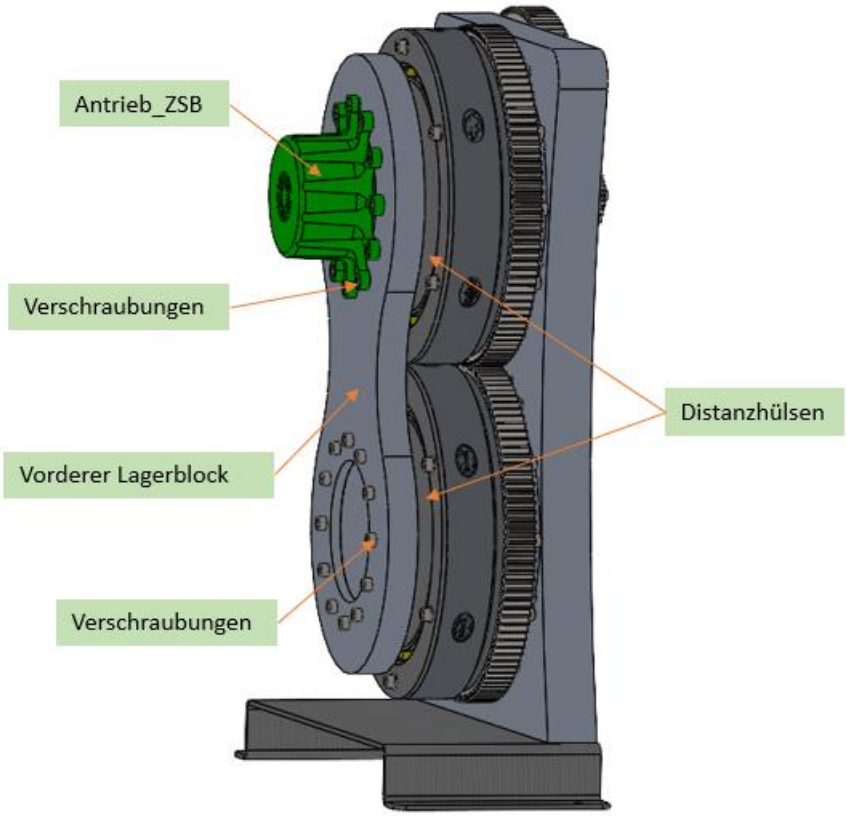


Abbildung 105: Montage Schritt 5

Mit dem Montieren des vorderen Lagerblocks ist der Zusammenbau des Getriebes der Münzprägeeinheit beendet.

4.5. Gehäuse

Aufgrund der kritischen Zeitschiene wird von der Entwicklung einer ästhetisch hochwertigen Einhausung abgesehen und stattdessen eine einfache Lösung vorgesehen. Der ebenfalls angedachte Verzicht auf diese Einhausung wird aus sicherheitstechnischen Gründen verworfen.

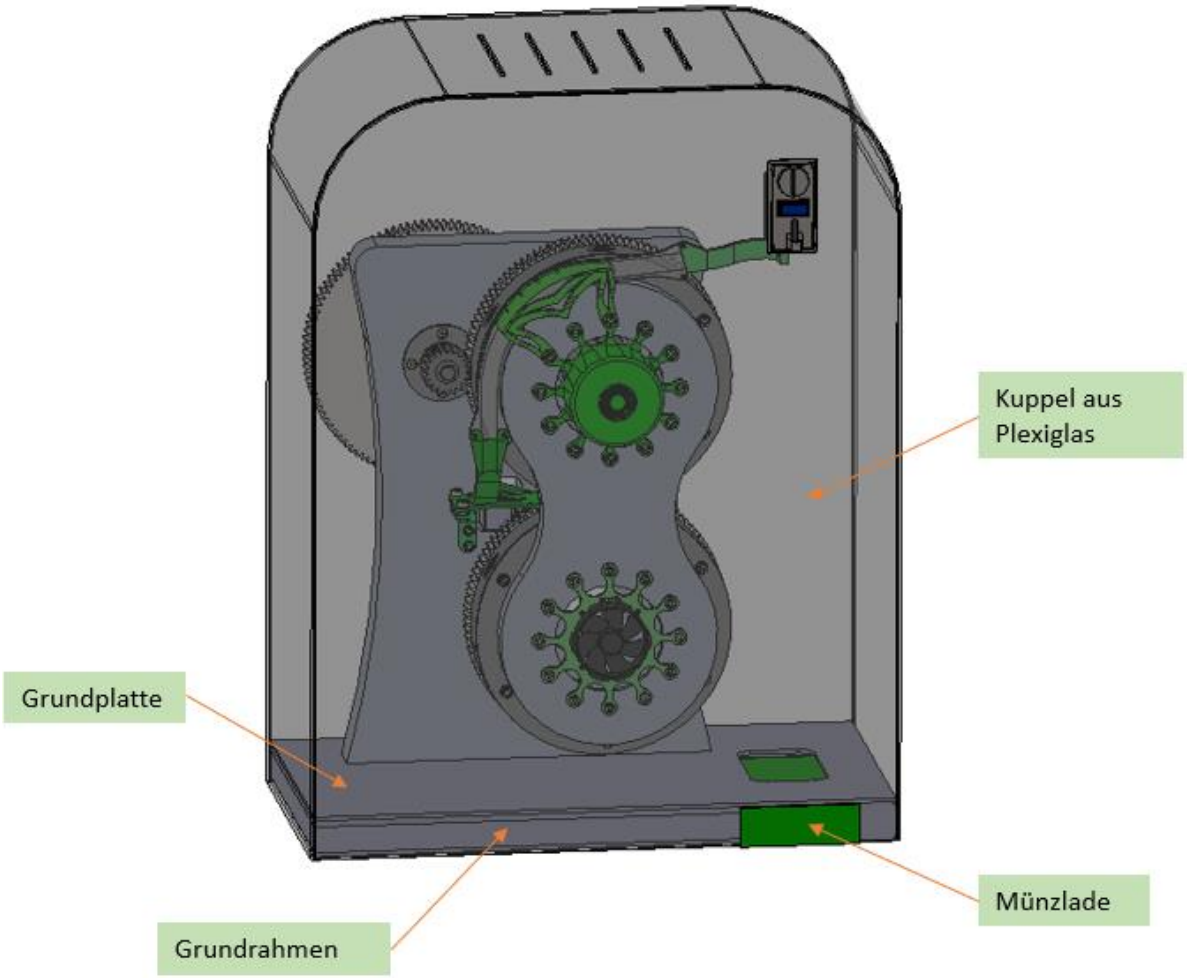


Abbildung 106: Gehäuse

Um das Hineingreifen in das Walzwerk zu verhindern, wird eine Kuppel aus Plexiglas über die Münzprägemaschine gestülpt.
Der hintere Lagerblock wird mit der Grundplatte verschraubt. Um dies zu realisieren, ist diese mit dem Grundrahmen verschweißt. In diesem Grundrahmen ist auch die „Münzlade“ integriert (siehe Abbildung 107: Grundrahmen).

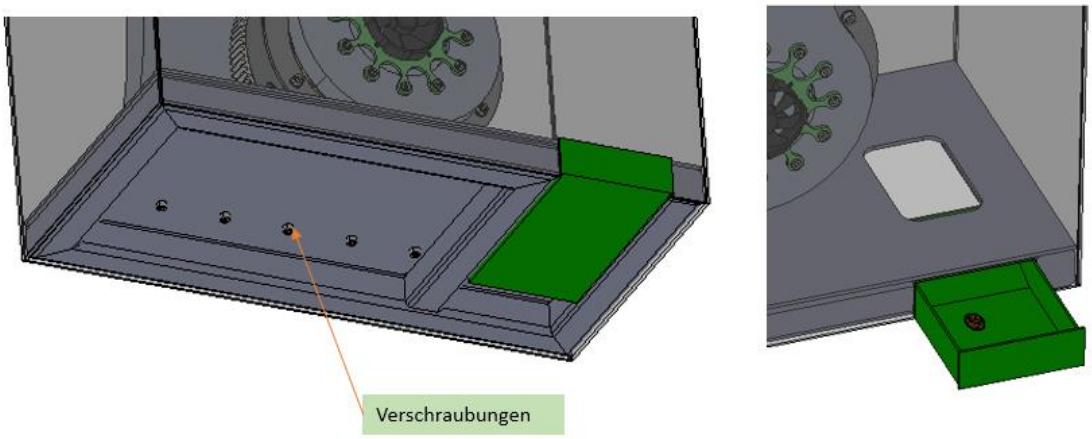


Abbildung 107: Grundrahmen

4.6. Technische Zeichnungen

Um die Einzelteile der Münzprägemaschine zu produzieren, werden nach Fertigstellung der CAD-Dateien, Fertigungszeichnungen angefertigt.

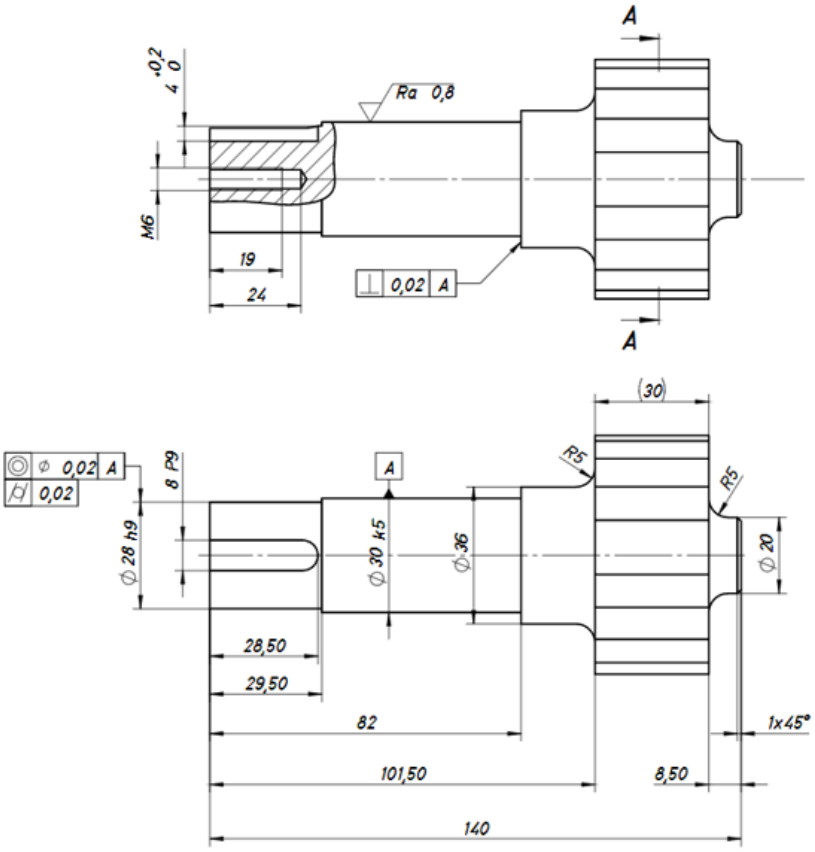


Abbildung 108: Bsp. Zeichnung

Die auf den Zeichnungen angegebenen Passungen, Toleranzen und Oberflächenangaben werden dem Tabellenbuch Rolof/Matek bzw. direkt den Passungsempfehlungen von SKF (Online: https://www.skf.com/at/products/rolling-bearings/principles-of-rolling-bearing-selection/bearing-selection-process/bearing-interfaces/seat-tolerances-for-standard-conditions) entnommen.

5. Weg der Münze

Die Münze muss von dem Münzeinwurf (siehe Kapitel 7.1.4), zur Einschubeinheit (siehe Kapitel 5.3), zwischen die Walzen und bis aus der Maschine transportiert werden.

5.1. Position des Münzeinwurfs

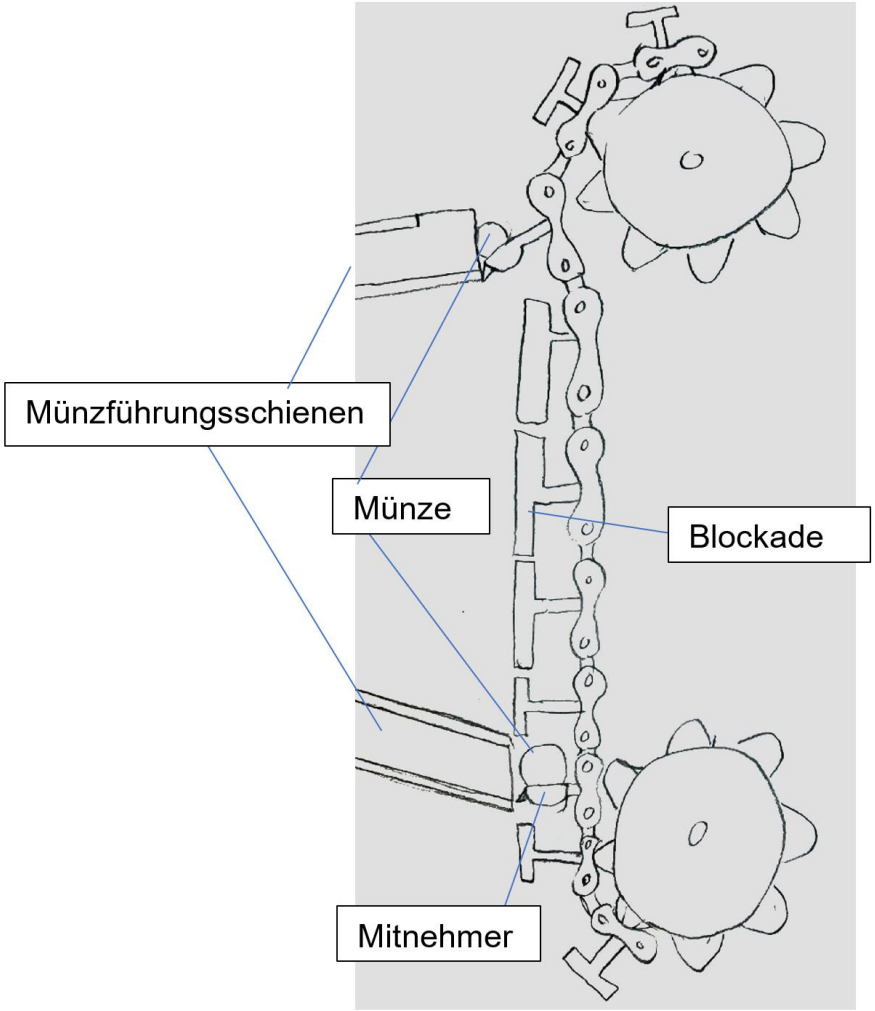


Abbildung 109: Münzaufzug

Der Münzeinwurf kann oben oder unten am Gehäuse befestigt werden. Eine Befestigung oben hat den Vorteil, dass die Münze mittels Schwerkraft bis zur Einschubeinheit transportiert werden kann. Jedoch wird das Display unten am

Gehäuse angebracht. Es ist komfortabler, wenn sich der Münzeinwurf neben dem Display befindet.

Die Münze muss danach von unten nach oben bis zur Einschubeinheit transportiert werden. Hierfür könnte ein Aufzug wie in Abbildung 109 verwendet werden. Hier wird die Münze mit dem Mitnehmer nach oben transportiert. Wenn der Mitnehmer die obere Schiene erreicht, kippt dieser nach vorne und die Münze rollt auf die Schiene. Damit die Münze nicht frühzeitig die untere Münzschiene verlässt wird diese durch die Blockaden blockiert. Dies ist komplex und benötigt viele weitere Bauteile deswegen wird der Münzeinwurf oben positioniert.

5.2. Münzführung

Die Münzzuführung muss die Münze vom Einwurf bis zu Einschubeinheit transportieren.

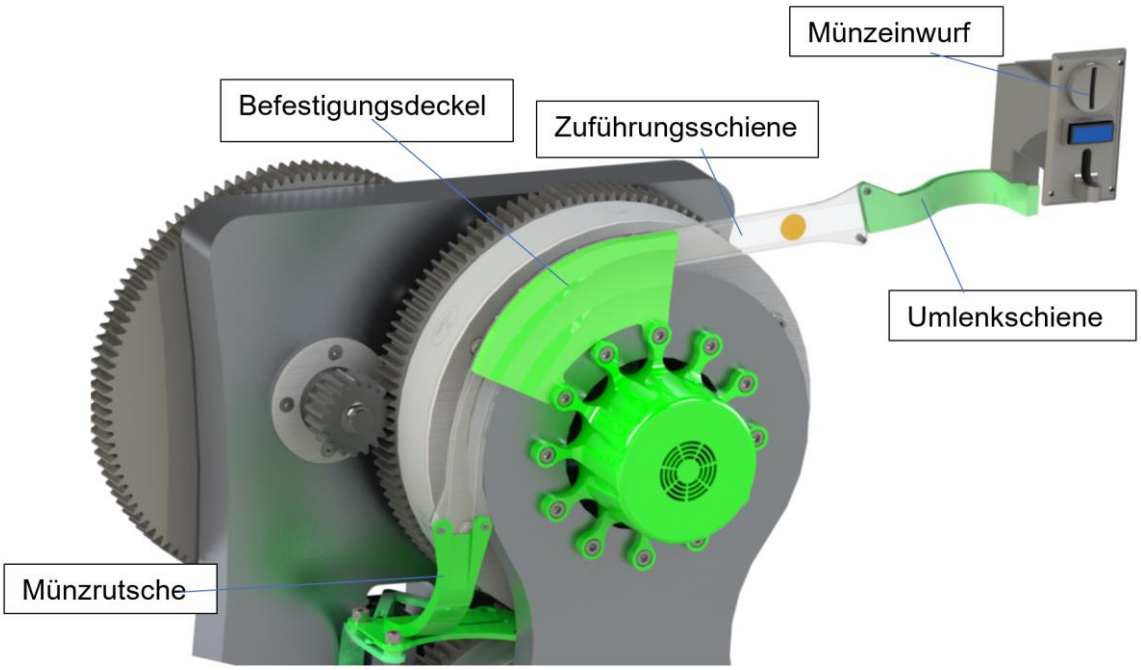


Abbildung 110: Münzzuführung

Die Münze wird in den, mit vier Schrauben an der Abdeckung befestigten, Münzeinwurf eingeworfen (siehe Abbildung 111). Sie verlässt den Münzeinwurf nach unten auf die Umlenkschiene.

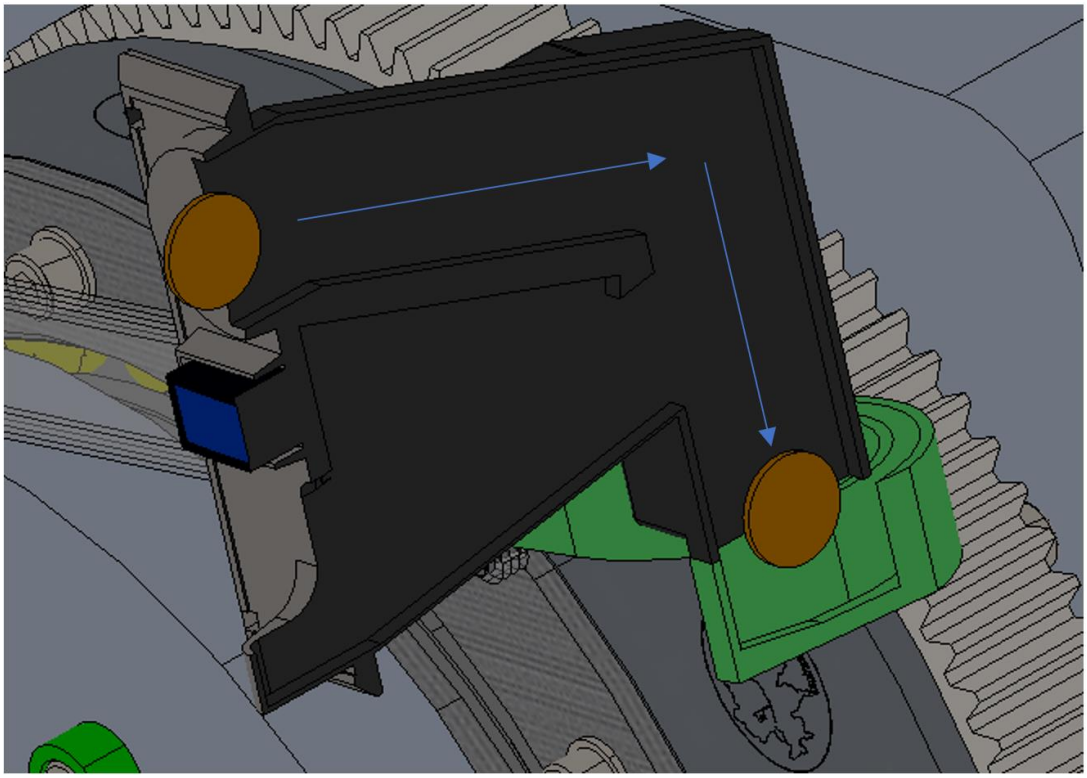


Abbildung 111: Münzeinwurf

Auf der Umlenkschiene (siehe Abbildung 112) wird die Münze in die Münzführschiene umgelenkt. Befestigt wird die Umlenkschiene an der Münzführschiene mit einer Schraube. Die Geschwungene Ausführung dient der Optik und verringert die Breite des Gehäuses. Durch die geringen Anforderungen an die Festigkeit und die hohe Komplexität wird dieser Teil aus PETG 3D-gedruckt.

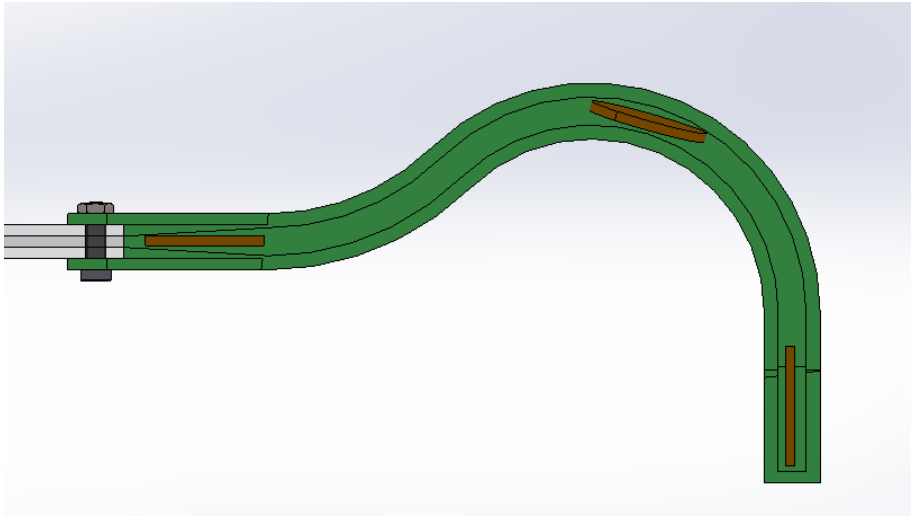


Abbildung 112: Umlenkschiene

Die Führungsschiene wird 4-teilig ausgeführt. Mit 2 identen Seitenteilen und 2 Abstandshaltern (siehe Abbildung 113**)** auf denen die Münze rollt. Diese werden an den Enden und durch den Befestigungsdeckel verschraubt. Dieser wird mit dem Motorgehäusedeckel mitgeschraubt, um die Zuführung zu befestigen (siehe Abbildung 114). Um die Münze während des Rollens beobachten zu können, wird die Schiene aus Plexiglas gefertigt. Ausgeschnitten werden die Teile der Zuführschiene per Laserschneider.

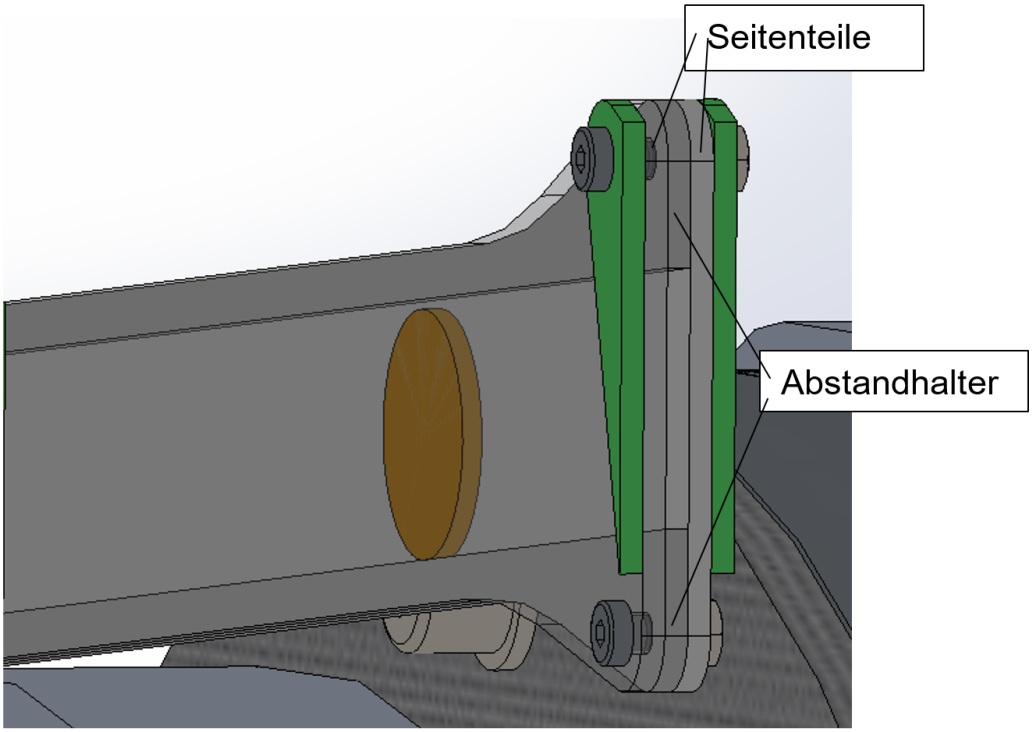


Abbildung 113: Zuführschiene

Um die Münze Einschieben zu können, muss sie in eine horizontale Lage gebracht werden. Hierfür wird die Münzrutsche benutzt (siehe Abbildung 115). Diese ist aufgrund ihrer komplexen Kontur aus PETG 3D-gedruckt. Die Münzrutsche wird mit der Zuführschiene verschraubt und mit der Leitschiene verklebt.

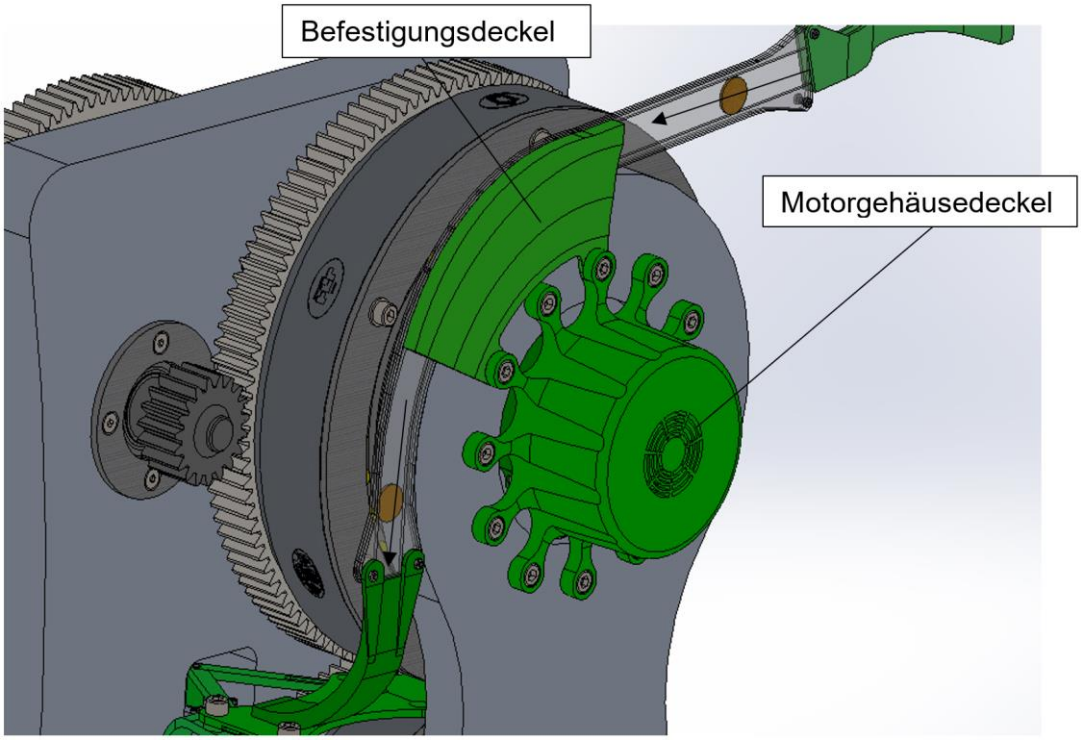


Abbildung 114: Zuführung

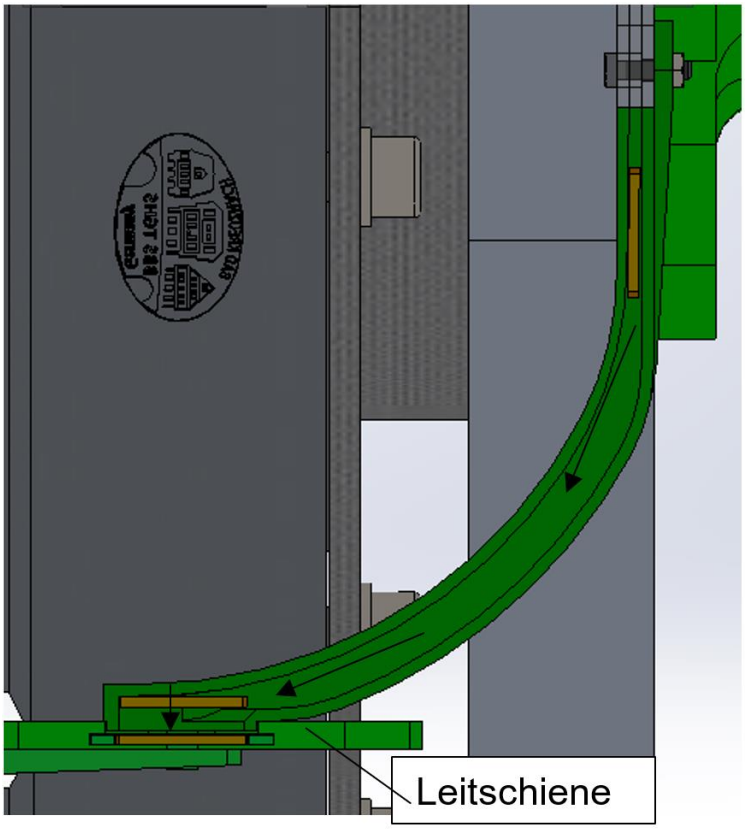


Abbildung 115: Münzrutsche

5.3. Einschubeinheit

Die Einschubeinheit muss die Münze zum richtigen Zeitpunkt zwischen die Walzen schieben.

5.3.1. Konzepte

5.3.1.1. Fallend

Bei diesem Konzept sind die Walzen horizontal nebeneinander angeordnet. Die Münze wird mit einem Stößel zurückgehalten. Im richtigen Moment fährt der Stößel ein und die Münze fällt zwischen die Walzen. Dieser Aufbau besteht aus wenigen Komponenten und ist einfach zu steuern. Jedoch kann die Geschwindigkeit des Einschiebens nicht angepasst werden. Dadurch könnte es zu Problemen mit der zeitlichen Genauigkeit kommen. Weiters ist dieser Vorgang optisch nicht spektakulär.

5.3.1.2. Schiebend

Hier schiebt ein Zylinder die Münze linear zwischen die Walzen. Der Zylinder wird an einem Lagerblock befestigt. Am Ende des Zylinders wird ein Gestänge befestigt, auf dessen Ende sich eine Aufnahme für die Münze befindet. Das Gestänge wird mit einem Gleitlager für erhöhte Stabilität gelagert. Im eingefahrenen Zustand kann die Münze auf die Aufnahme rutschen. Beim Ausfahren des Zylinders wird die Münze zwischen die Walzen geschoben, um von den Walzen eingezogen zu werden. Vorteilhaft ist, dass die Geschwindigkeit kontrolliert werden kann. Der Zylinder benötigt jedoch viel Platz.

5.3.1.3. Schwenkend

Die Münze wird mit einem Motor zwischen die Walzen geschwenkt. Hierbei lässt sich die Geschwindigkeit und der Schwenkwinkel einstellen. Weiters können der Motor und der Schwenkarm kompakt verbaut werden. Aus diesen Gründen wird die Münze per Schwenkeinheit eingeschoben.

5.3.2. Schwenkeinheit

5.3.2.1. Schwenkarm und Schieber

In dem Schwenkarm befindet sich ein Langloch, für den Schieber (siehe Abbildung 116). Der Schieber schiebt die Münze nach vorne. Er wird in dem Langloch des Schwenkarms und in der Leitschiene (siehe 5.3.2.2**)** geführt. Diese Führung wandelt die rotatorische Bewegung des Schwenkarms in eine translatorische Bewegung des Schiebers.

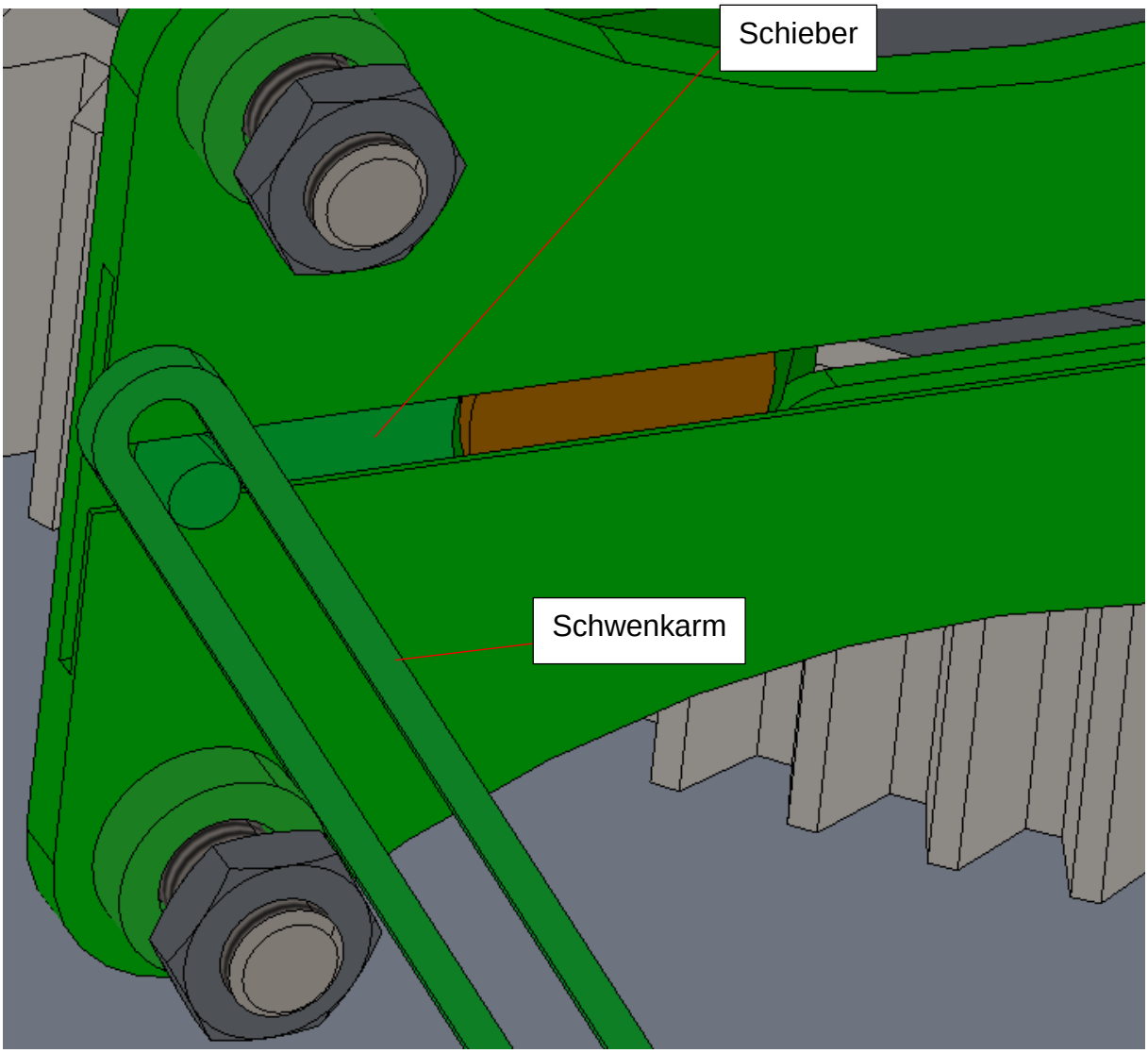


Abbildung 116 Schwenkarm

5.3.2.2. Leitschiene

Damit die Münze nicht vomSchieber fällt, wird sie in einer Leitschiene geführt. Die Leitschiene wird mit der Münzrutsche verklebt und mit einem Winkel am Lagerblock befestigt. Ihre geschwungene Form dient zur Optik und zur Befestigung des Winkels.

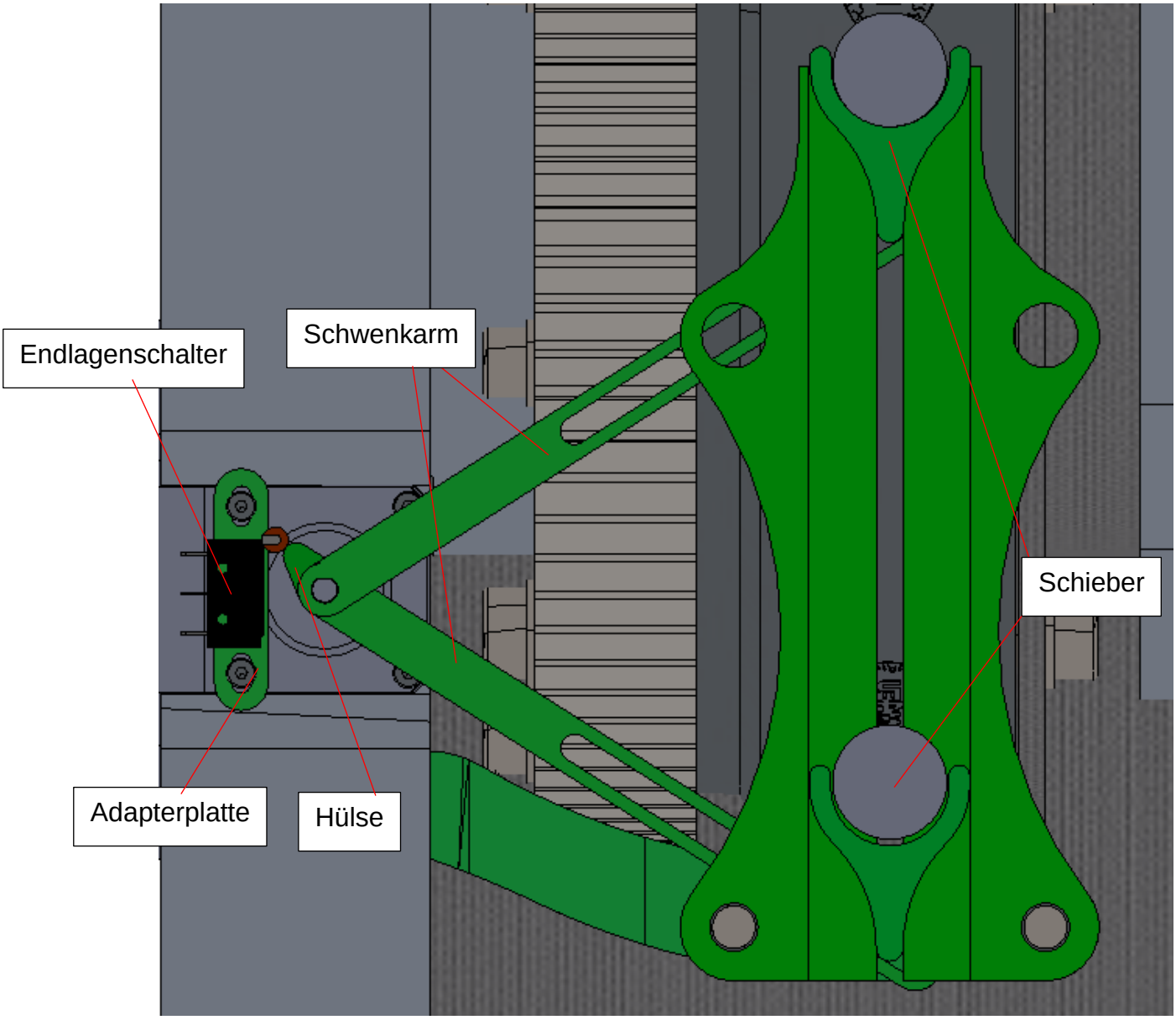


Abbildung 117 Leitschiene

5.3.2.3. Schrittmotor

Die Drehbewegung zum Schwenken wird durch einen Schrittmotor (NEMA 16) realisiert. Dieser bietet sowohl eine Anpassbarkeit der Geschwindigkeit als auch die Möglichkeit den Schwenkwinkel einzustellen. Befestigt wird der Schrittmotor mithilfe eines gebogenen Bleches, welches mit dem Lagerblock verschraubt wird. Der Schrittmotor muss referenziert werden, damit ein immer gleichbleibender Nullpunkt bekannt ist. Hierfür wird ein Endlagenschalter benutzt, der durch die Distanzhülse des Schwenkarms ausgelöst wird. Die Position des Endlagensensors lässt sich durch die Langlöcher der Adapterplatte anpassen (siehe Abbildung 117).
Um die Schwenkeinheit an die restliche Konstruktion anpassen zu können, werden ihre Komponenten 3D-gedruckt. So können die Komponenten schnell neu gefertigt werden.

5.4. Abführen der Münze

Die Münze fällt nach dem Prägen, durch in Loch im Bodenblech, in eine Lade. Diese Lade kann herausgezogen werden, um die Münze zu entnehmen (siehe Abbildung 118).

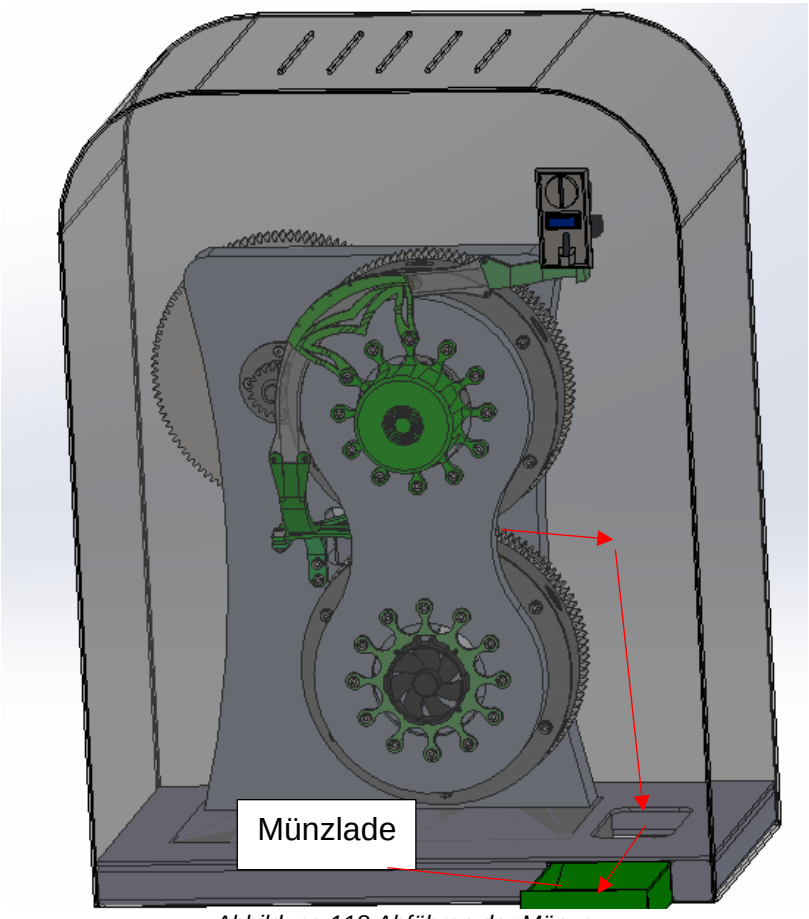


Abbildung 118 Abführen der Münze

6. Steuerung

6.1. Auswahl der Steuerung

6.1.1. SPS

Abbildung 119: SPS (https://de.wikipedia.org/wiki/Logo_(SPS)#)

Eine SPS (siehe Abbildung 119) auch Speicherprogrammierbare Steuerung genannt wird zur Steuerung beziehungsweise Regelung von Maschinen und Anlagen verwendet. Dies wird durch mehrere Eingänge sowie Ausgänge und einem Betriebssystem ermöglicht, welche den Anweisungen des Anwenderprogrammes folgen.

Vorteile:

- Einfaches modifizieren und ändern des Ablaufes
- Einfache Vervielfachung der Steuerung durch Kopieren
- Hohe Zuverlässigkeit
- Niedriger Energieverbrauch
- Hohe Flexibilität

- Schnelle Verarbeitung der Daten
- Netzwerkfähig

Nachteile:

- Zu teuer für einfache Anwendungen
- Ungeeignet für komplexe Interfaces

Vgl. (Juschkat, 2019)

6.1.2. Arduino

Abbildung 120: Arduino (https://www.reichelt.at/at/de/arduino-uno-rev-3-smd-variante-atmega328-usb-arduino-uno-p119045.html)

Ein Arduino (siehe Abbildung 120) ist eine Physical-computing Plattform, welche aus einer Soft- und Hardware besteht, die zur Realisierung von Hobbyanwendungen genutzt wird. Die Hardware besteht aus einem Eingabe/Ausgabe - Board welches mit einem Mikrocontroller und Ein- und Ausgängen bestückt ist. Die Software des Chips basiert auf einer C++ ähnlichen Programmiersprache.
Vgl. (Arduino, 2022)

Vorteile:

- Einfaches modifizieren und ändern des Ablaufes
- Einfache Vervielfachung der Steuerung durch Kopieren
- Niedriger Energieverbrauch
- Kostengünstig
- Schnelle Verarbeitung der Daten
- Hohe Flexibilität
- Hohe Zuverlässigkeit
- Einfach programmierbar durch eigene Entwicklungsumgebung

Nachteile:

- Ungeeignet für Interfaces
- Nicht Netzwerkfähig

Vgl. (Arduino vs Raspberry Pi, 2018)

6.1.3. Raspberry Pi

Abbildung 121: Raspberry Pi (https://upload.wikimedia.org/wikipedia/commons/f/f1/Raspberry_Pi_4_Model_B_-_Side.jpg)

Ein Raspberry Pi (siehe Abbildung 121) ist ein Einplatinencomputer welcher über diverse Anschlüsse wie USB, HDMI, Video, Ton und GPIO Pins verfügt. Als

Betriebssystem kann das Chip eigene Betriebssystem Raspbian verwendet werden, aber auch verschiedenste Linux sowie Windows Distributionen. Als Haupt-Programmiersprache verwendet der Raspberry Pi Python.
Vgl. (Raspberry Pi, 2022)

Vorteile:

- Kostengünstig
- Einfaches modifizieren und ändern des Ablaufes
- Einfache Vervielfachung der Steuerung durch Kopieren
- Niedriger Energieverbrauch
- Hohe Flexibilität
- Geeignet für Interfaces
- Großer Aufgabenbereich

Nachteile:

- Nicht geeignet für zeitabhängige Anwendungen
- Unzuverlässigkeit der GPIO Pins

6.1.4. *Lösungsansatz*

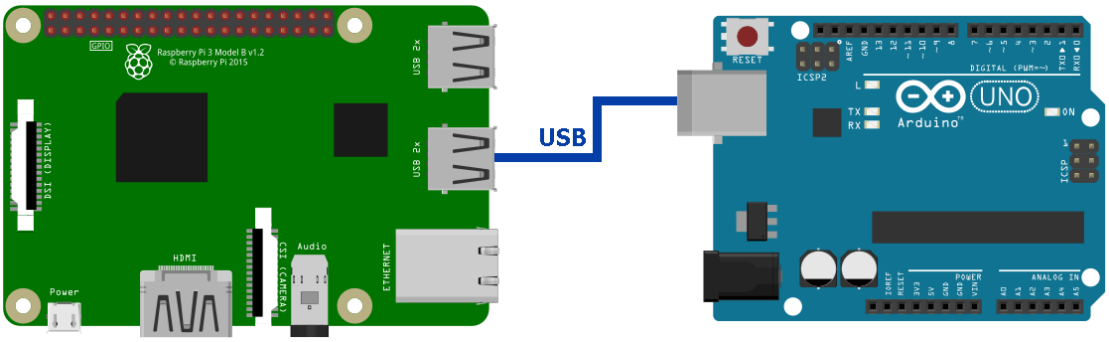


Abbildung 122: Raspberry Pi + Arduino (https://elektro.turanis.de/html/prj176/index.html)

Als Lösungsansatz wird von dem Projektteam keines der oben genannten Konzepte im Einzelnen verwendet. Eine SPS ist für die Münzprägemaschine nicht rentabel. Der Arduino hingegen ist nicht geeignet für ein User Interface. Der Raspberry Pi ist unter den genannten die beste Option da er ein User Interface unterstützt und Ein- und Ausgangs Pins besitzt. Diese Pins sind aber zu unzuverlässig und ungenau. Um die

gesamten Anforderungen bestmöglich zu erreichen hat sich das Projektteam für einen Verbund von Raspberry Pi und Arduino (siehe Abbildung 122) in einer „Master-Master Replication“ entschieden. Es vereint die Vorteile beider Komponenten und eliminiert so die negativen Aspekte. Weiters sind beide Komponenten relativ kostengünstig und einfach in der Anschaffung.

6.2. Kommunikation der Steuerungen

Zum Kommunizieren der beiden Geräte wird die serielle Schnittstelle, auch COM Port genannt, verwendet. Die Bezeichnung seriell kommt daher, dass die 8-Bits eines Bytes hintereinander übertragen werden. Vor jedem und nach jedem Block aus 8-Bits wird außerdem ein Start-bit und Stopp-bit gesendet, um die Grenzen des Bytes zu definieren. Die Kommunikation erfolgt im Vollduplex-Modus, hierbei werden zum Senden und Empfangen verschieden Pins verwendet, was den Vorteil bringt, dass Informationen in beide Richtungen gleichzeitig übertragen werden können.
Vgl. (Tyson)

7. Sensoren Aktoren

7.1.1.1. Münzerkennung

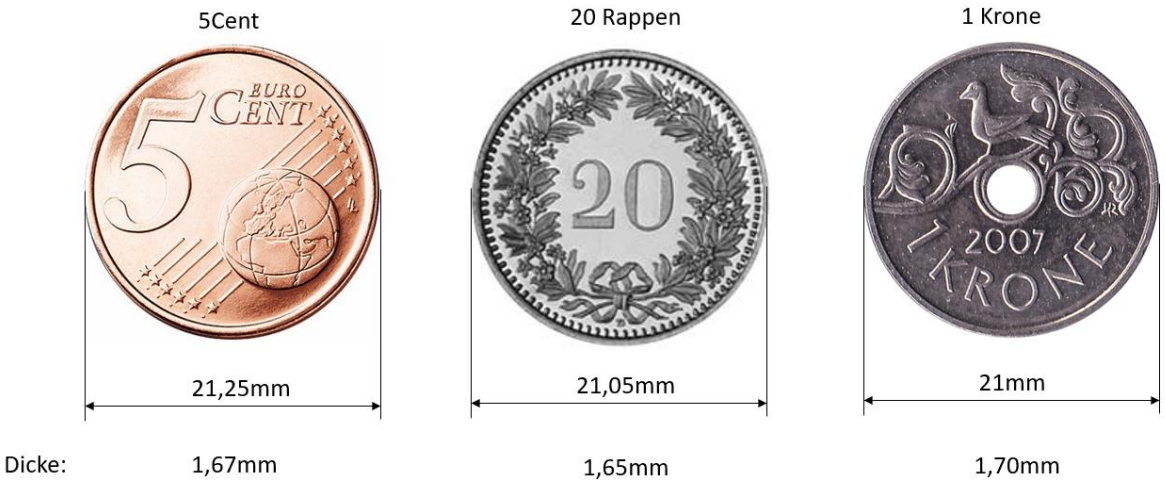


Abbildung 123: Münzen https://www.muenzen.eu/euro-umlaufmuenzen.htm

Es sollen nicht nur 5 Cent Münzen, sondern auch 1 Norwegische Krone oder 20 Schweizer Rappen geprägt werden. Dies kommt dadurch zustande, dass bei dem Projekt PPM4EU nicht nur Euro-Länder teilnehmen, sondern auch die Schweiz und Norwegen. Da es umständlich wäre immer 5 Cent Münzen neben der Schweizer und der Norwegischen Münzpräge bereitzustellen, hat man sich dafür entschieden die Maschine auch für norwegische Kronen und schweizer Rappen auszulegen. Diese Münzen wurden gewählt da sie einen ähnlichen Durchmesser und eine ähnliche Dicke haben siehe Abbildung 123.

Um einen erfolgreichen Prägevorgang gewährleisten zu können dürfen nur diese Münzen verwendet werden, da bei kleineren Durchmessern und Dicken das Motiv nicht richtig abgebildet werden kann und bei härteren und dickeren Münzen, die Maschine beschädigt werden könnte. Deswegen muss überprüft werden, ob eine der zu verwendenden Münzen eingeworfen wurde und falsche müssen aussortiert werden. Um dies zu bewerkstelligen, wurden zwei Konzepte erstellt.

7.1.2. Mechanisch

Das Konzept für die Mechanische Münzerkennung prüft die Münze auf ihren Durchmesser. Ist die Münze zu klein fällt sie durch den kleinen Schlitz des Halteblechs (siehe Abbildung 124), ist diese zu groß passt sie nicht in den Schieber. Der Schieber wird zum Einwerfen der Münze zurückgeschoben. Erreicht der Schieber seine hintere Endlage aktiviert er einen Endlagenschalter, der signalisiert, dass eine Münze eingeworfen wurde. Die Münze wird nicht auf Dicke oder Material geprüft.

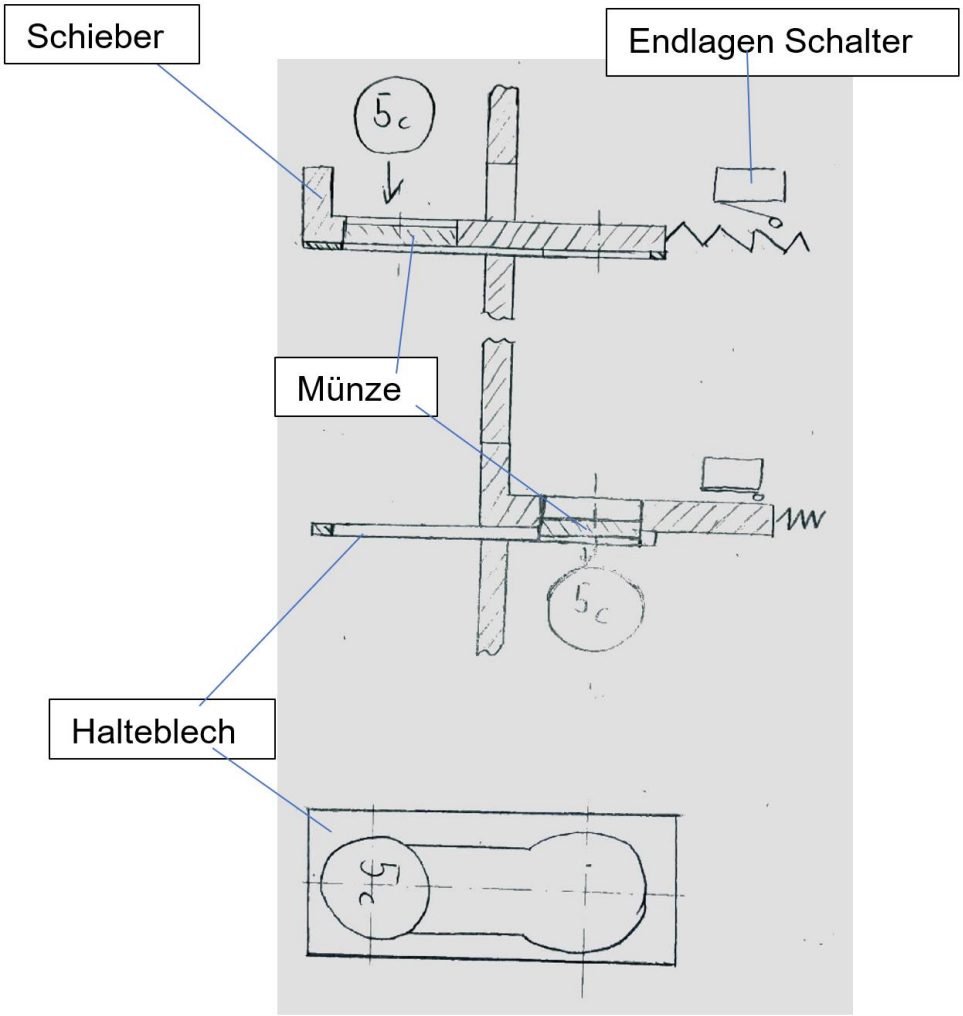


Abbildung 124: mechanische Münzerkennung

7.1.3. Elektronisch

Elektronische Münzerkennungseinheiten können die Münze auf Material, Dicke und Durchmesser prüfen. Hierfür fließt durch Spule A (siehe Abbildung 125) ein Strom, wodurch ein Feld erzeugt. Dieses Feld fließt durch die Münze bis zur Spule B. Je nach Durchmesser, Dicke und Material der Münze wird die Dauer und Stärke des Felds verändert. Das Feld in Spule B induziert dann einen Strom, der in Dauer und Stärke variiert. Der Strom wird dann mit einem vorgespeicherten Strom verglichen. Sind die 2 Ströme ähnlich genug wird die Münze akzeptiert.

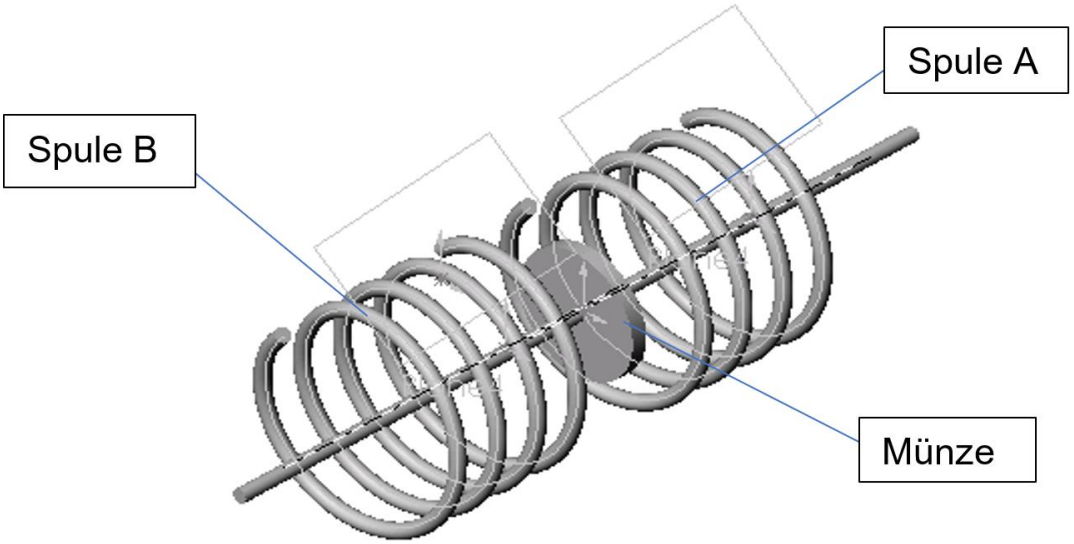


Abbildung 125: elektronische Münzerkennung https://web.mit.edu/2.972/www/reports/coin_tester/coin_tester.htm

7.1.4. CH-926

Vom Projekt PPM4EU wird die Münzerkennungseinheit CH-926 zur Verfügung gestellt. Sie ist eine elektronische Münzerkennungseinheit (siehe Kapitel 7.1.3). Es können bis zu 6 verschiedene Münzen einprogrammiert werden. Diese senden jeweils unterschiedlich viele Impulse damit die Steuerung sie unterscheiden kann. Wenn eine der einprogrammierten Münzen eingeworfen und erkannt wird, wird die Spule des elektromagnetischen Schiebers (siehe Abbildung 126), mit Strom beaufschlagt. Dadurch öffnet sich dieser und die Münze fällt aus dem rechten Münzauslass. Wenn die eingeworfene Münze nicht erkannt wird, wird sie durch den linken Münzauslass zurückgegeben.

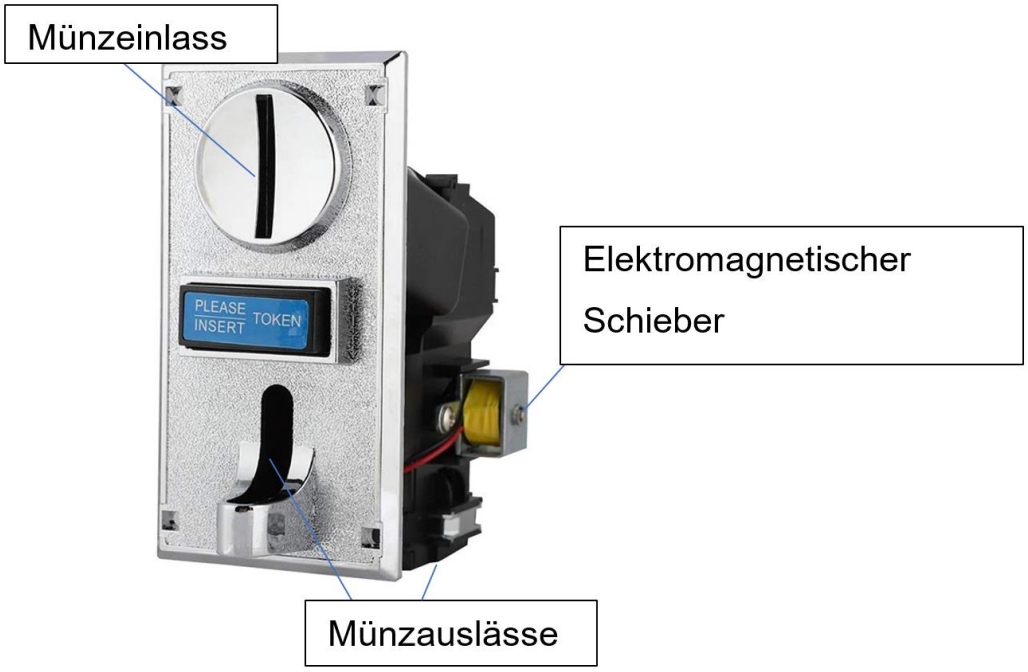


Abbildung 126: 117 CH-926 https://ae01.alicdn.com/kf/Hf91e64b446834eb990b7a008d09ab770t/2020-Multi-M-nzpr-fer-Elektronische-Rolle-Unten-M-nzpr-fer-Selector-Mechanismus-Automaten-Arcade-Spiel.jpg_Q90.jpg_.webp

7.2. Detektion der Walzposition

Die Position der Walzen muss detektiert werden, um gewährleisten zu können, dass die Münze zum richtigen Zeitpunkt eingeschoben wird.

7.2.1. Konzepte

7.2.1.1. Optischer inkrementaler Drehgeber

Ein optischer inkrementaler Drehgeber besteht aus einer Lichtquelle, einem Photorezeptor und einer Codescheibe (siehe Abbildung 127). Der Lichtstrahl der LED wird durch die Codescheibe blockiert. Dreht diese sich weiter, kann der Lichtstrahl durch die Lücke den Photorezeptor aktivieren. Dadurch ergibt sich ein Pulssignal, mit dem man bestimmen kann, wie weit sich der Drehgeber relativ zum Startpunkt gedreht hat.

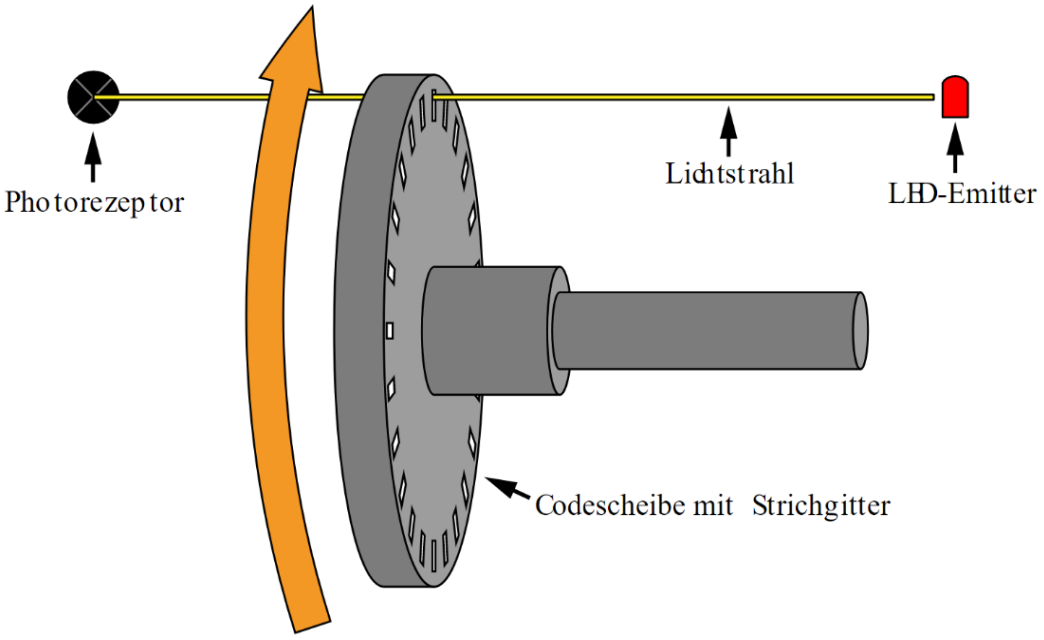


Abbildung 127: Drehgeber https://www.waycon.de/produkte/encoder-winkelgeber/messprinzip-encoder-drehgeber/

Durch den großen Durchmesser der Walzen, kann die Codescheibe nicht direkt an der Walze angebracht werden. Man muss sie an Welle 2 anbringen (siehe Abbildung 128). Hier befinden sich der Emitter und Empfänger in einer Zange, die am Lagerblock befestigt wird. Der Vorteil dieses Konzepts ist die Genauigkeit durch die vielen Messpunkte. Jedoch ist das Auslesen durch die hohe Frequenz des Signals kompliziert. Weiters wird eine weiterer Sensor als Nullpunkt, zum Referenzieren, benötigt.

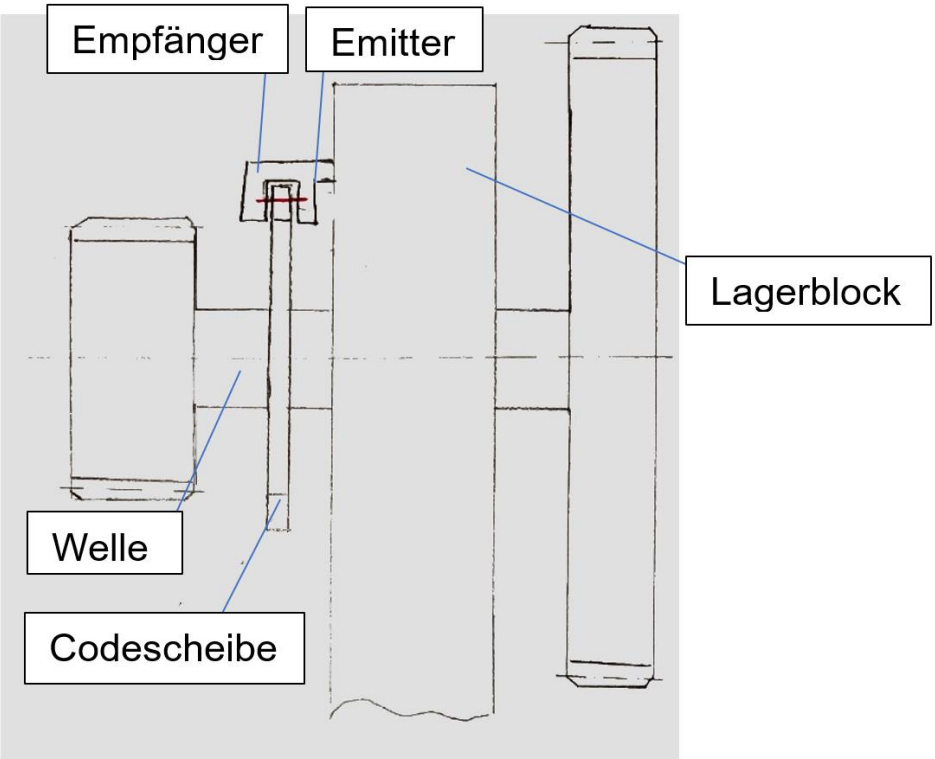


Abbildung 128: Konzept Inkrementalgeber

7.2.1.2. Optischer Sensor

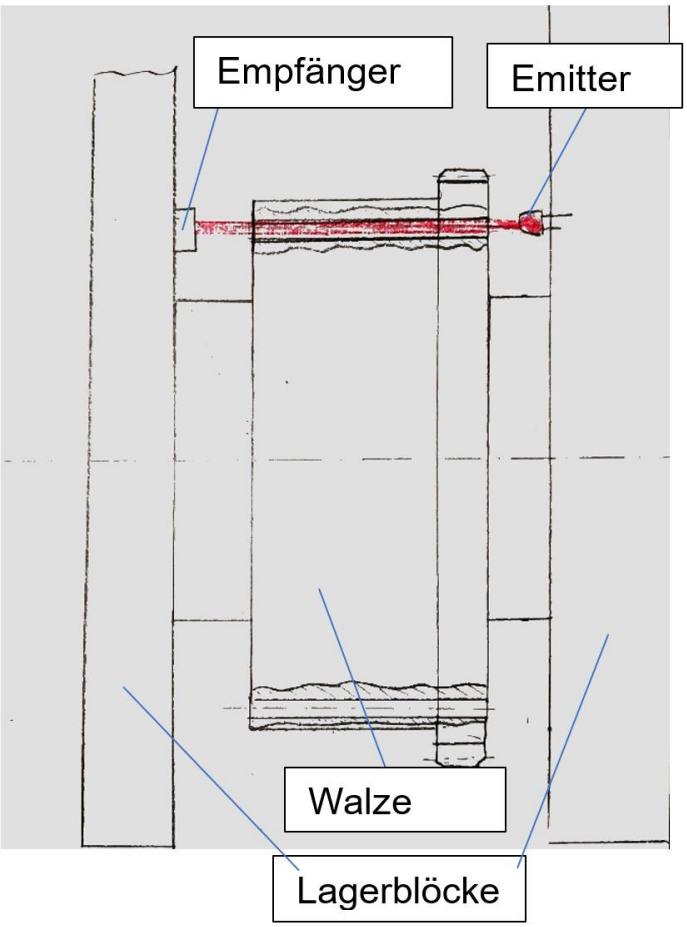


Abbildung 129: Konzept optischer Sensor

Es wird wieder ein optischer Emitter und ein Empfänger benutzt (siehe Kapitel 7.2.1.1). Hier wird die Codescheibe durch am Umfang angeordnete Bohrungen durch die Walze und den Zahnkranz, ersetzt (siehe Abbildung 129). Den Löchern und Motiven wird eine Nummer zugeteilt. Wenn die Nummer des ausgewählten Motives und des Lochs übereinstimmen, beginnt der Einschiebevorgang.

Bei diesem Konzept ist die Frequenz der Pulse durch die geringe Anzahl an Löchern sehr gering. Daraus folgt, dass das Auslesen des Signals einfacher ist. Optische Sensoren sind jedoch schmutzempfindlich und müssen deshalb gereinigt werden. Weiters könnte der Empfänger durch externe Lichtquellen fehlausgelöst werden, wodurch wiederum die Position falsch bestimmt werden würde. Zusätzlich benötigt man auch hier einen weiteren Sensor als Nullpunkt, zum Referenzieren.

7.2.1.3. Zahnflankenerkennung

Der induktive Sensor erkennt hier ähnlich wie bei einer Codescheibe (siehe Kapitel 7.2.1.1) die Zahnflanken und Zahnlücken des Zahnkranzes (siehe Abbildung 130). Dadurch ergibt sich ein Pulssignal, aus dem man die Position der Walze ermitteln kann.

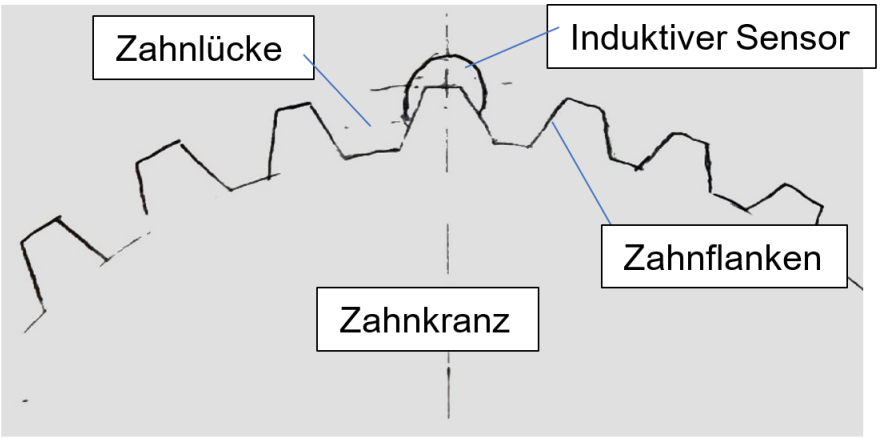


Abbildung 130: Zahnflankenerkennung

Die Position der Walze kann durch die vielen Messpunkte sehr genau bestimmt werden. Der Einbau ist einfacher als bei optischen Sensoren, da kein separater Emitter und Empfänger benötigt wird. Weiters müssen keine Veränderungen an den Walzen und Zahnkränzen vorgenommen werden. Jedoch ist die Frequenz des Signales sehr hoch, dadurch ist es schwierig auszulesen und es wird ein weiterer Sensor zum Referenzieren des Nullpunkts benötigt.

7.2.1.4. Schraubenerkennung

Hier werden die Schrauben, die zur Befestigung des Zahnkranzes dienen, zum Auslösen des Induktiven Sensors verwendet (siehe Abbildung 131).

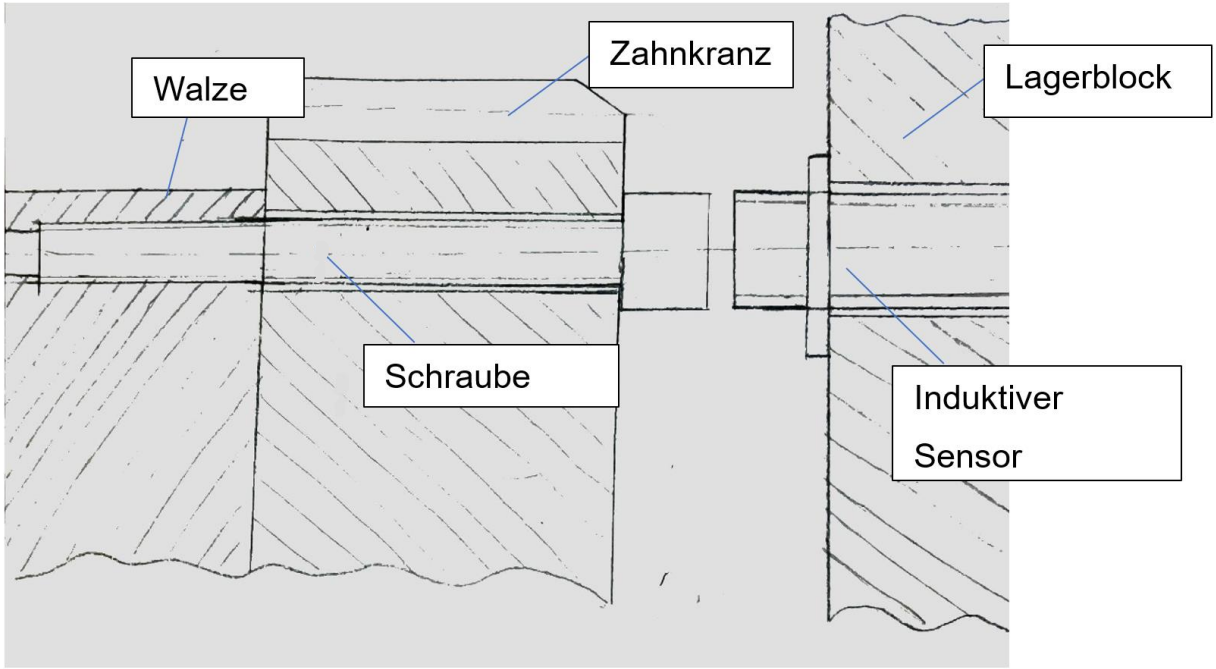


Abbildung 131: Schraubenerkennung

Der Einbau ist einfacher als bei optischen Sensoren. Es müssen keine Veränderungen an den Walzen oder Zahnkränzen vorgenommen werden. Die Frequenz ist durch die wenigen Messpunkte niedrig, dadurch ist es leicht die Position auszuwerten. Jedoch wird einen Sensor zum Setzten des Nullpunkts, als Referenz, benötigt.

7.2.2. Lösungsansatz

Es werden 2 induktive Sensoren verwendet. Der erste detektiert wie in Kapitel 7.2.1.4 beschrieben die Position der Walze. Der zweite Sensor wird verwendet um einen Nullpunkt zu setzten. Hierfür wird ein gebogenes Blech, dass mit den Schrauben mitgeschraubt wird als Auslöser verwendet (siehe Abbildung 132.)

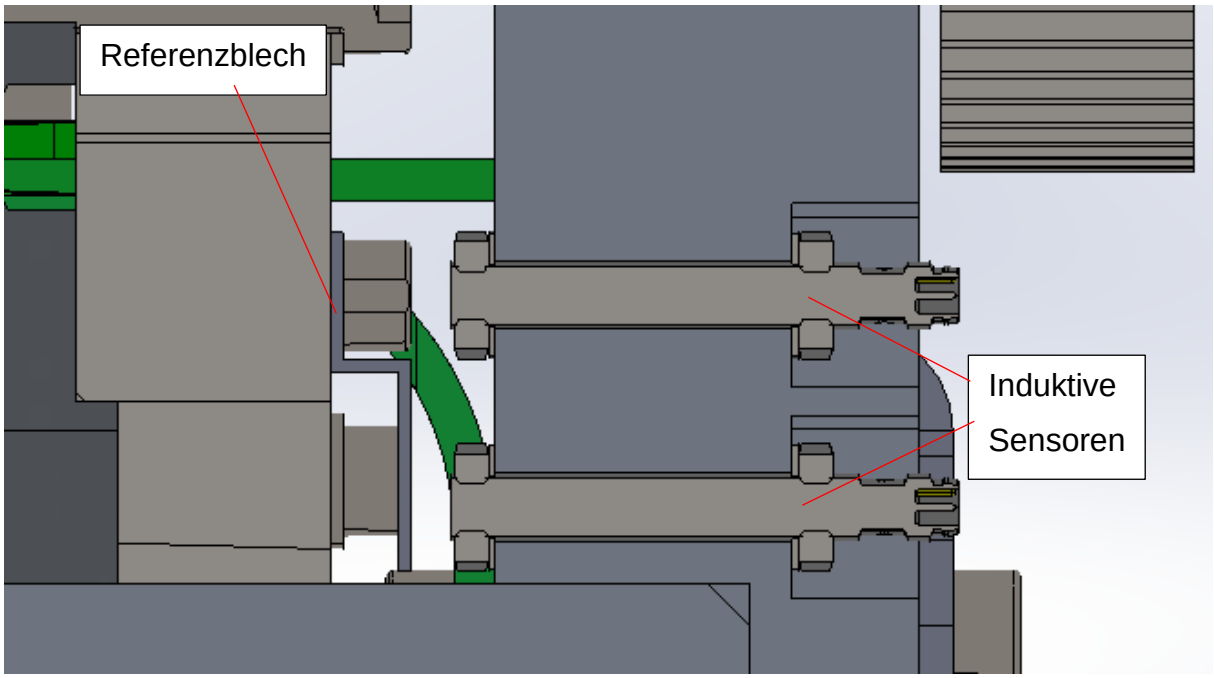


Abbildung 132 Schraubenerkennung mit Referenz

7.3. Schrittmotor

7.3.1. Auswahl

Es wird der NEMA 16HS3404P1 verwendet. Er wurde gewählt, da dieser, wie viele andere verwendeten Komponenten, mit 12 Volt betrieben werden kann und dadurch kein separates Netzgerät benötigt. Weiters ist er kompakt, günstig und besitzt eine Flachstelle auf der Welle, um den Schwenkarm zu befestigen.

7.3.2. Treiberplatine

Der Schrittmotor wird nicht direkt über eine Verstärkerschaltung mit dem Arduino verbunden, da dies die Schaltung unnötig verkomplizieren würde. Es wird stattdessen die Schrittmotortreiberplatine DRV8825 verwendet. Gewählt wurde diese, da sie allen Anforderungen des Schrittmotors und der Steuerung gerecht wird. Die maximale Versorgungsspannung der Platine beträgt 45V (12V werden benötigt), der maximale Strom beträgt 2,2A (0,4A werden benötigt), dieser kann über ein Potentiometer eingestellt werden. Die Ansteuerung der Treiberplatine erfolgt direkt über die 5V Ausgänge des Arduinos. Außerdem lässt sich die Schrittauflösung über Signale an den Eingängen M0, M1 und M2 (siehe Abbildung 133) auf bis zu 1/32 Schritte einstellen. Weiters verfügt die Platine über Sicherheitsmaßnahmen, wie ein Abschalten bei zu großer Temperatur, zu großem Strom oder bei Kurzschluss. Um zu Verfahren wird ein PWM Signal an den STEP Eingang der Treiberplatine angelegt und ein High oder Low Signal, an den DIR Eingang, um die Richtung festzulegen. Die Versorgungsspannung für den Schrittmotor wird mit VMOT und GND verbunden. Hier sollte ein 100µF Kondensator parallel geschalten werden, um Spannungsspitzen zu glätten. Der Schrittmotor wird mit den Ausgängen B2 B1 A1 A2 verbunden, A und B stehen hier jeweils für die Spulen des Schrittmotors. Will man den Treiber ausschalten, legt man ein Signal an den ENABLE Eingang an.

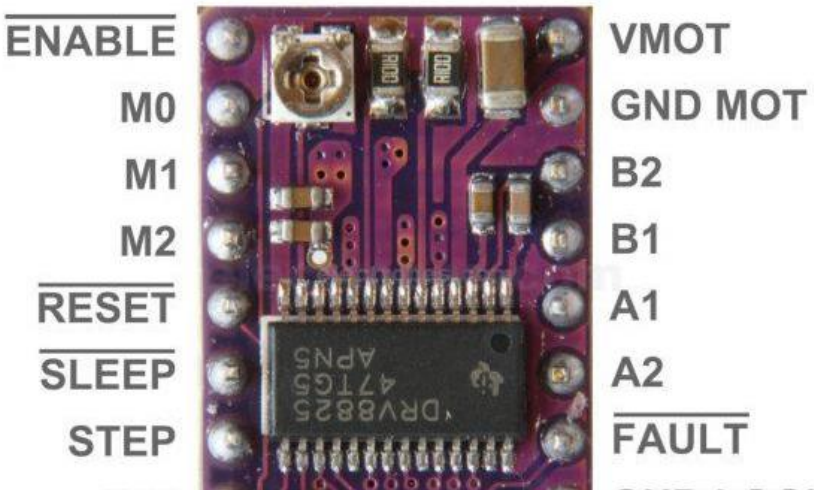


Abbildung 133 Schrittmotortreiber https://electrobes.com/product/drv8825-stepper-motor-driver-with-heat-sink/

7.4. Motorsteuerung

Der Motor, der die Walzen antreibt, muss durch die Steuerung kontrollierbar sein, um den Prägevorgang automatisieren zu können. Prinzipiell existieren dafür zwei mögliche Ausführungsvarianten.

7.4.1. Relais

Durch die Erfassung der Walzposition mittels induktiver Sensoren (siehe Kapitel 7.2.2), könnte die Zeitverzögerung zwischen Erreichen des Vorpunktes zum Prägen und des Startens des Einschubvorgangs, an die Drehzahl anpassen. Hierbei muss nur der Walzmotor gestartet und gestoppt werden, eine Regelung im eigentlichen Sinne ist nicht erforderlich und somit reicht ein einfaches Relais zum Ansteuern des Antriebmotors.

7.4.2. Motortreiberplatine

Der Vorteil einer Motortreiberplatine ist, dass die Drehzahl eingestellt werden kann. Dadurch kann die Prägezeit so eingestellt werden, dass der Prägevorgang optisch am ansprechendsten wirkt. Verwendet wird der Cytron MD30C Motor Treiber. Dieser kann die 30A des Netzteils dauerhaft übertragen und Spitzenströme von bis zu 80A über Kondensatoren bereitstellen. Angesteuert wird die Platine über zwei 5V Eingänge und einen Ground. Die gewünschte Geschwindigkeit wird über ein PWM-Signal gesteuert. Die Richtung über ein, an den DIR Eingang angelegtes HIGH oder LOW Signal. Über Jumper kann die Platine konfiguriert werden, ein internes PWM-Signal, einstellbar über ein internes Potentiometer oder ein externes PWM-Signal zu verwenden.

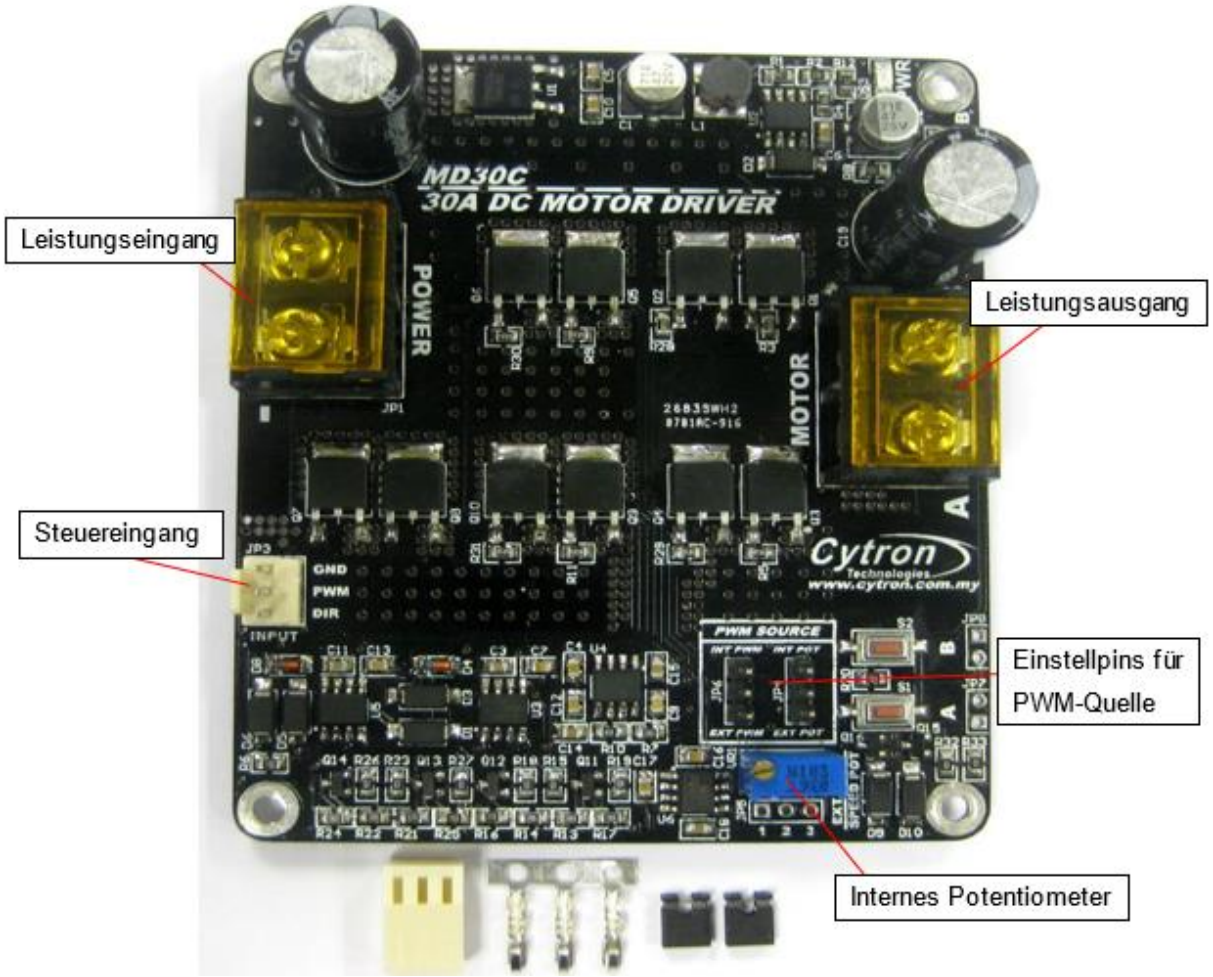


Abbildung 134: Motortreiberplatine (aus Datenblatt md30c 1,4-usersmanual)

7.5. Schaltung

In diesem Kapitel wird behandelt, wie die Komponenten miteinander verbunden werden.

7.5.1. Galvanische Trennung

Die Induktiven Sensoren und die Münzerkennungseinheit werden mit 12V betrieben und geben ein 12V Signal aus. Jedoch beträgt das Spannungsniveau des Arduinos 5V. Um 12V Signale auslesen zu können, ohne den Arduino zu beschädigen, müssen die Sensoren und der Arduino, galvanisch voneinander getrennt werden. Hierfür werden Optokoppler verwendet. Auf der Sensorseite werden die Sensoren und die Münzzuführung angeschlossen (siehe Abbildung 135). Senden diese ein Signal leuchtet die LED im Optokoppler. Das Licht der LED löst den Fototransistor aus und schließt dadurch den Stromkreis auf der Steuerungsseite. Daraus folgt ein 5V Signal an einen Eingang des Arduinos.

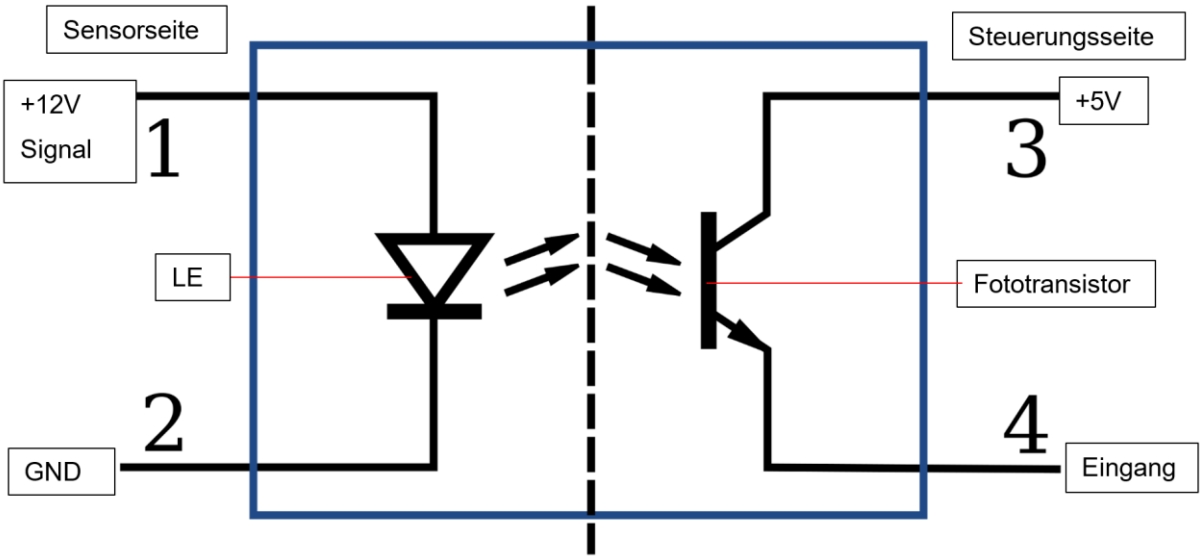


Abbildung 135: Optokoppler https://de.wikipedia.org/wiki/Optokoppler

7.5.2. Versuche

Um die Komponenten, das Programm und deren Funktion zu testen wurde die Schaltung zuerst auf Steckbrettern aufgebaut (siehe Abbildung 136).

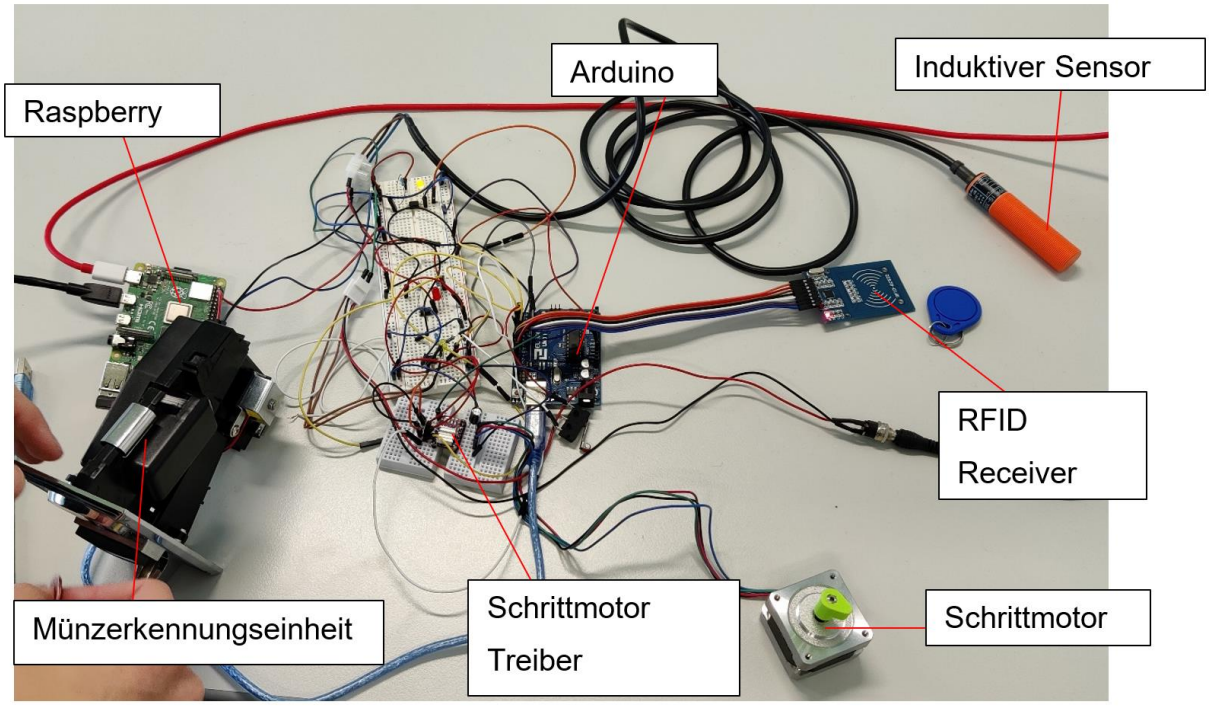


Abbildung 136: Versuchsschaltung

Diese Schaltung wurde Stück für Stück aufgebaut. Zuerst wurde der induktive Sensor verbunden (siehe Abbildung 137) [hier mit Raspberry statt Arduino]. Wie in Kapitel 7.5.1 beschrieben wird der Induktive Sensor mit der 12V Seite verbunden und das Signalkabel (schwarz) mit dem Optokoppler. Die 5V Seite des Optokopplers wird mit der Steuerung verbunden. Um das Signal zu visualisieren, wurde eine LED zwischen den Optokoppler und der Steuerung geschalten.

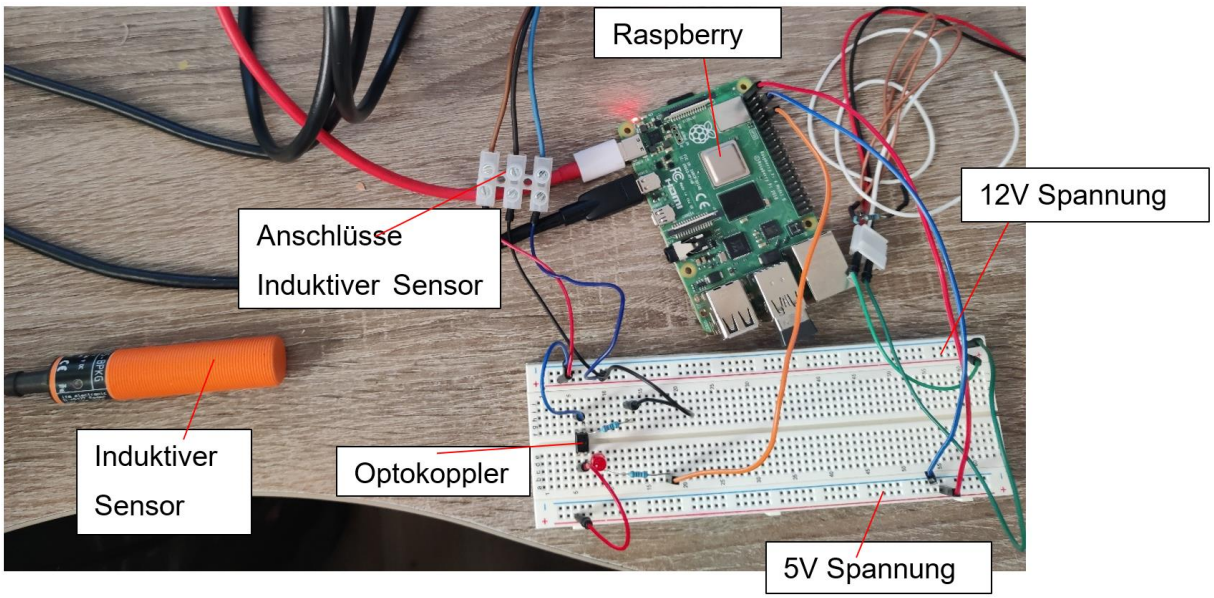


Abbildung 137: Schaltung Induktiver Sensor

Die nächsten Schritte waren das Verbinden der Münzerkennungseinheit, des RFID Receivers und des Schrittmotors (siehe Abbildung 136).

7.5.3. Schaltplan

Um diese komplexe Schaltung immer wieder aufbauen zu können, wurde sie in Tinkercad gezeichnet. Tinkercad ist ein Programm, in dem sich online Arduino Schaltungen aufzeichnen und simulieren lassen. Es wurde gewählt da es kostenlos und einfach zu benutzen ist. Aufgrund der mangelhaften Bauteilliste dieses Programms, wurden nicht verfügbare Bauteile durch ähnliche ersetzt. Die Sensoren und Münzerkennungseinheit werden durch Schalter dargestellt. Die Anschlüsse des RFID Receivers sind als Pinleiste ausgeführt und die DIP4 Optokoppler wurden durch DIP6 Optokoppler ersetzt. Um verhindern zu können, dass Münzen während des Prägevorgangs in die Maschine eingeworfen werden können, wird die Stromversorgung der Münzerkennungseinheit über ein Relais geschalten.

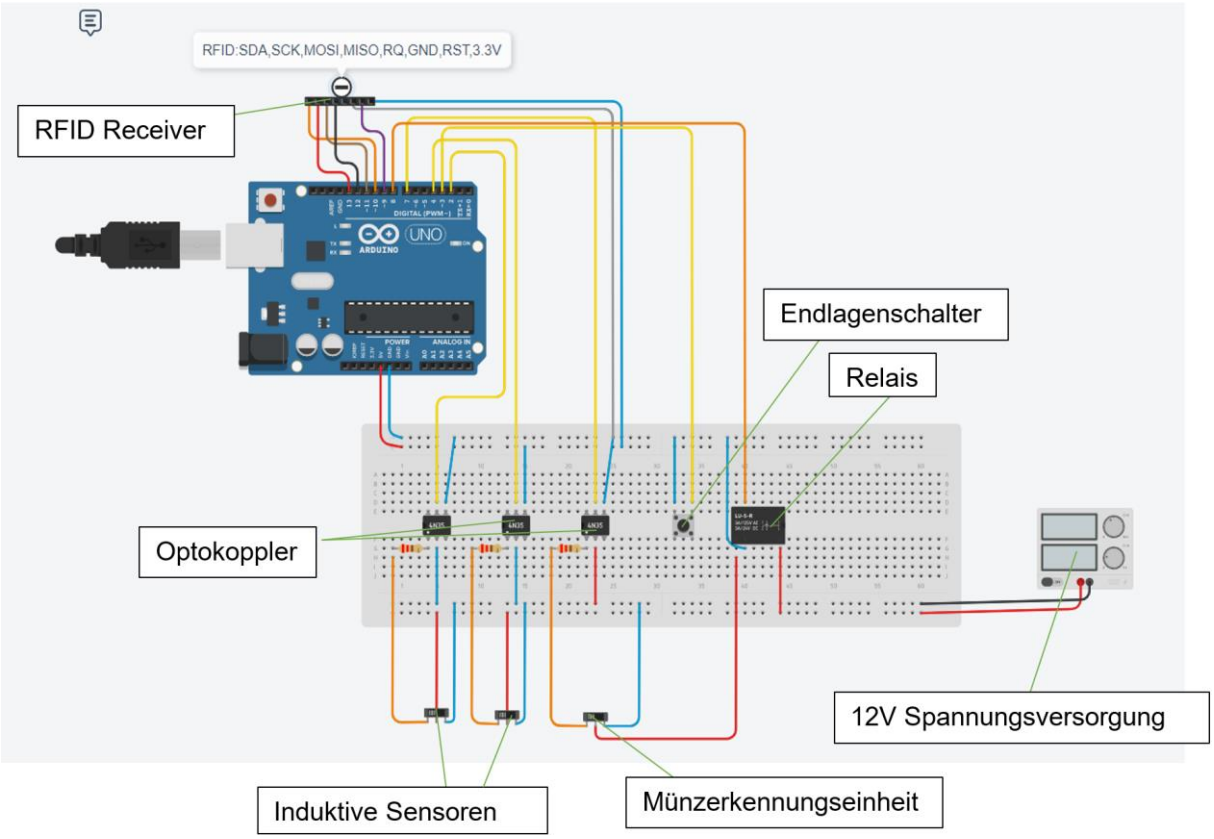


Abbildung 138: Tinkercad Schaltplan

Der Schrittmotor und dessen Treiber wurden nicht zum Schaltplan hinzugefügt da diese Komponenten schwer durch ähnliche zu ersetzen sind. Aus diesem Grund wurde die Schaltung auf einem eigenen Steckbrett aufgebaut und zwischen den Versuchen nicht abgebaut.

7.5.4. Schaltplatine

Um die Schaltung zu vereinfachen und die zur verbindung benötigten Kabel zu minimieren, wird eine Schaltplatine geätzt. Diese ist mit dem Programm Proteus gezeichnet und in der Schule gefertigt. Die Platine ist zweiseitig und verfügt über eine Massefläche, welche die Massen verbinden und durch ihre Impedanz Störungen verringert (siehe Abbildung 140). Sensoren und Aktoren werden mit Stiftleisten an der Platine sowie mit Buchsen an den Kabeln miteinander verbunden. Der Arduino, der Schrittmotor Treiber und die Optokoppler werden in Buchsenleisten gesteckt. Printklemmen verbinden die 5V und 12V Spannungsversorgung mit der Platine (siehe Abbildung 139). Die Komponenten müssen von dem Diplomarbeitsteam auf die Platine gelötet werden (siehe **Fehler! Verweisquelle konnte nicht gefunden werden.** & Abbildung 142).

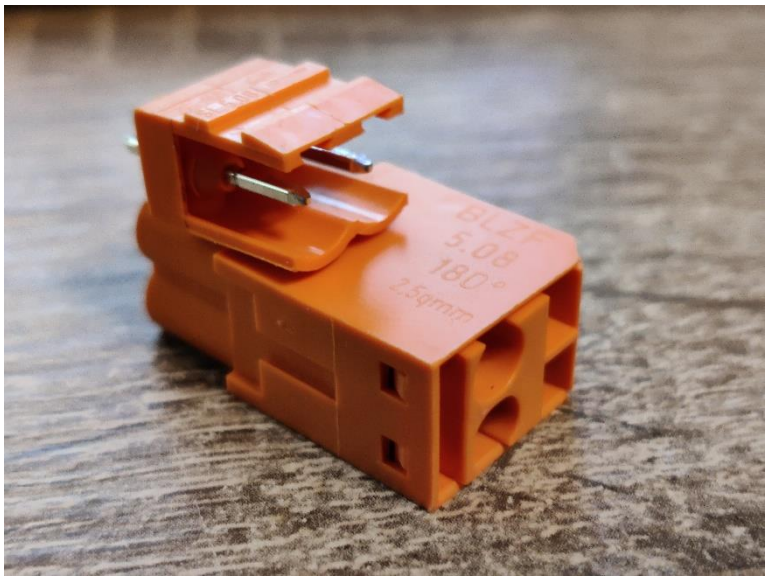


Abbildung 139 Printklemmen

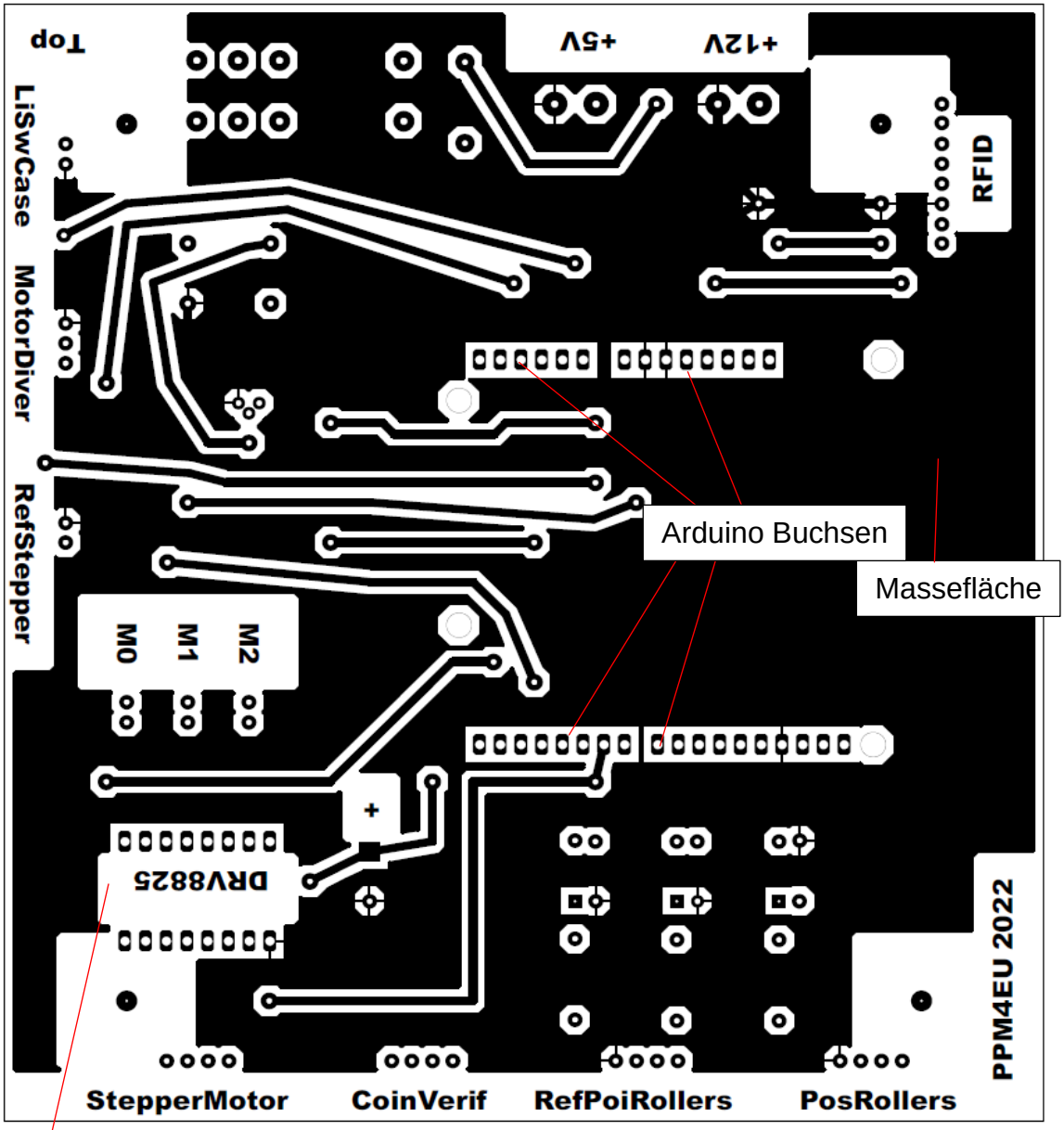


Abbildung 140 Zeichnung Platine

Schrittmotortreiber Buchsen

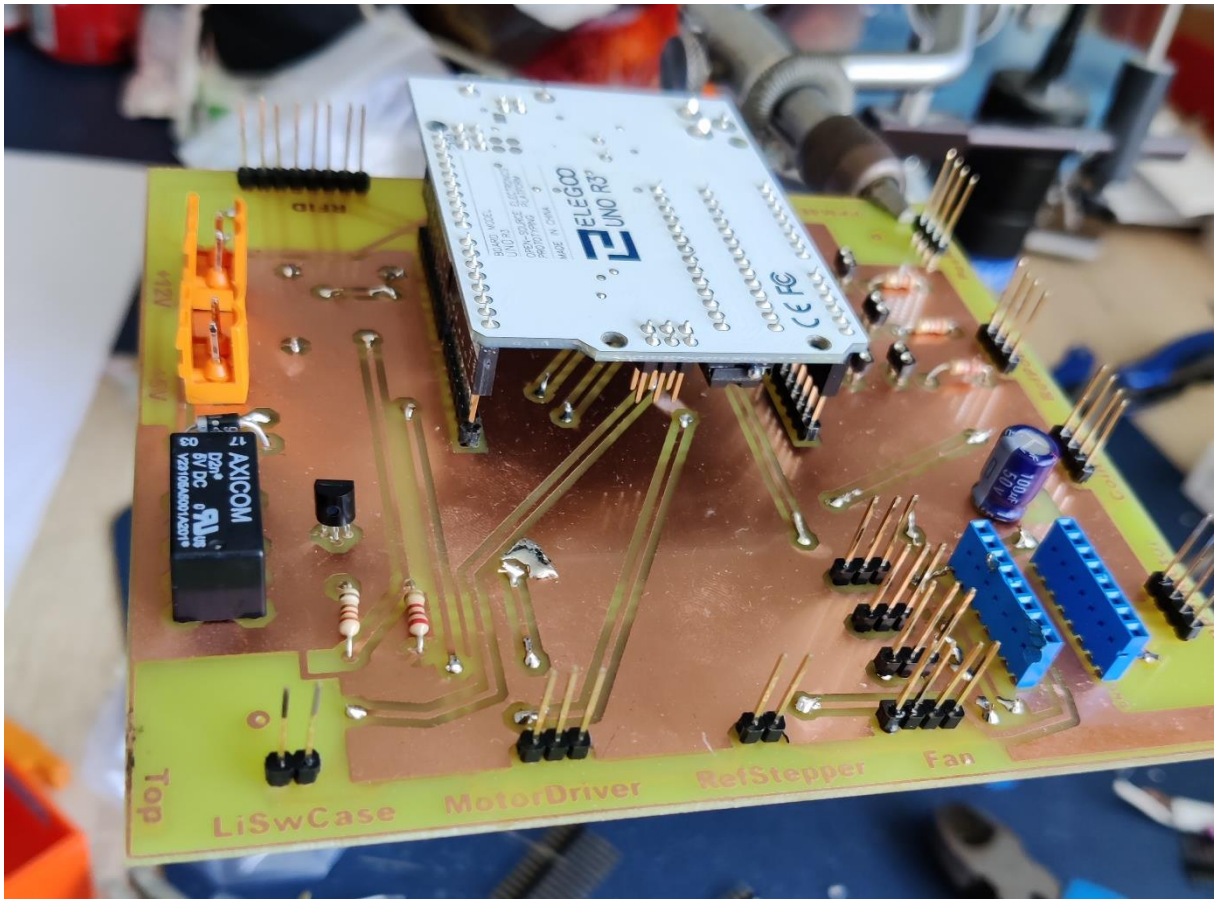


Abbildung 141 Schaltplatine

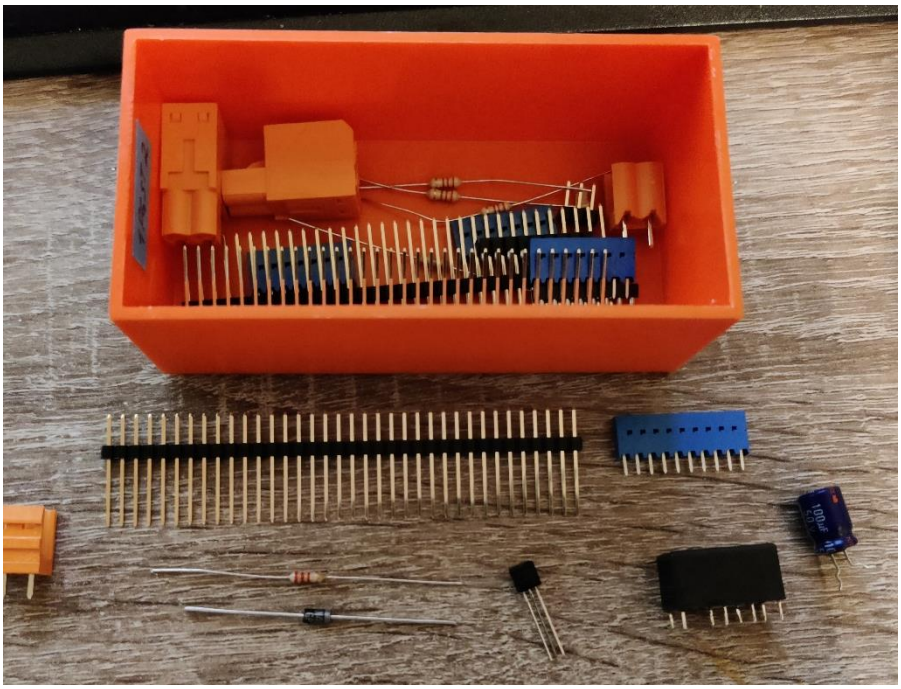


Abbildung 142 Komponenten Platine

8. Ablaufsteuerung

8.1. C++

Zur Programmierung des Arduino wird eine C++ ähnliche Programmiersprache verwendet. C++ ist eine objektorientierte Programmiersprache, diese zeichnet sich aus, da sie sowohl effizient als auch maschinennahe ist. C++ wird hauptsächlich zur Anwendungsprogrammierung als auch für die Systemprogrammierung verwendet.
Vgl. (C++ Definition, 2016)

8.1.1. Funktion:

Eine Funktion ist ein Codeblock der Operationen ausführt. Durch Definition von optionalen Eingabeparametern ist es möglich Argumente an die Funktion weiterzugehen. Eine Funktion kann auch einen Wert ausgeben. Wird das Schlüsselwort „void“ vor der Funktion verwendet so gibt diese keine Werte zurück. Funktionen werden verwendet um gängige Operationen in einen wiederverwendbaren Block zu Kapseln.
Vgl. (Functions, 2021)

8.1.2. Switch

Die switch-Anweisung besteht aus einer Reihe von Fallbezeichnungen oder auch „Cases“. Hierbei muss die Bezeichnung des „Case“ einen ganzzahligen Typ haben oder ein Klassentyp sein. Weiters beinhaltet der Switch eine Standardbezeichnung, auch „Default“. Außerdem gibt es das „break“ Statement, ist dieses vorhanden wird an die Anweisung nach dem Switch übergeben. Switch-Anweisung prüfen, ob ein übereinstimmender Ausdruck vorhanden ist und führen den zugeordneten code aus.
Vgl. (Switch Statement, 2021)

8.2. Setup des Arduino Programmes

8.2.1. Bibliotheken

```
//Libraries
#include <SPI.h>
#include <MFRC522.h>
#include <AccelStepper.h>
```

Abbildung 143: Arduino Bibliotheken

Damit Peripheriegeräte mit höheren Anforderungen genutzt werden können, ist es wichtig, deren Bibliotheken zu integrieren. Bevor diese integriert werden können, müssen sie aber noch installiert werden. In der oben zu sehenden Abbildung wurden die Bibliotheken für den RFID Leser sowie für den Stepper Driver inkludiert (siehe Abbildung 143).

8.2.2. Pin-Belegung festlegen

```
#define RST_PIN 9 //reset Pin RFID
#define SS_PIN 10 //SDA PIN RFID

#define IND1 2    //First Inductive Sensor
#define IND2 4    //Second Inductive Sensor
#define COIN 7    //Coin Verify Unit
#define END 3     //final position Switch
#define RELAI A2  //Relai

#define dirPin 8    //Stepper direction Pin
#define stepPin 5   //Stepper step Pin
#define motorInterfaceType 1  //Stepper Interface Type
#define sleep A1    //Stepper sleep Pin

#define MOTOR 6     //Motor Pin
#define dirPIN A5   //Motor direction Pin
```

Abbildung 144: Arduino Pin festlegen

Um die Pins am Arduino nutzen zu können muss im Vorhinein definiert werden, welche Pins genutzt werden. Außerdem muss den Pins ein Name zugewiesen werden. Dies erfolgt wie in der oberen Abbildung (siehe Abbildung 144). Es ist auch möglich analoge Pins als digitale Pins zu nutzen.

8.2.3. Variablen

```
//========================================== Sensor input declarations

 bool COINmem=0;     //Coin Verify Unit Memory
 bool COINval=0;     //Coin Verify Unit Value
 bool IND1mem=0;     //First Inductive Sensor Memory
 bool IND1val = 0;   //First Inductive Sensor Value
 bool IND2mem=0;     //Second Inductive Sensor Memory
 bool IND2val = 0;   //Second Inductive Sensor Value
 bool ENDmem=0;      //Final position Memory
 bool ENDval=0;      //Final position Value
 bool IND1reset=0;   //First Inductive Sensor Reset

 //========================================== Raspberry Pi data declaration

 double design=0;                      //selected Design
 long unsigned int offset=0;           //selected Offset
 long unsigned int MOTORtimeout = 0;   //selected Motortimeout
 int MOTORspeed = 0;                   //Motor speed
 int STEPPERspeed = 0;                 //Stepper speed

//========================================== Counter and time dely declaration

 double INDcounter=0;
 int steps=0;
 long unsigned int previousMillis = 0;
 long unsigned int currentMillis = 0;
 long unsigned int previousMillis_timeout = 0;
 long unsigned int currentMillis_timeout = 0;

//========================================== RFID declarations

 byte sector         = 1;
 byte blockAddr      = 4;
 byte trailerBlock   = 7;
 byte buffer[18];
 byte size = sizeof(buffer);
 MFRC522::StatusCode status;
 MFRC522 mfrc522(SS_PIN, RST_PIN);
 MFRC522::MIFARE_Key key;

//========================================== RFID output declaration

 String code = "";
 String addzero = "";
 String pass = "";

//========================================== Serial declarations

 String data = "";
 String subAblauf = "";
 String subLoop = "";
 String subLoop2 = "";

 AccelStepper stepper = AccelStepper(motorInterfaceType, stepPin, dirPin);
```

Abbildung 145: Arduino Variablen

Damit die Variablen im gesamten Ablaufprogramm genutzt werden können, müssen diese als global deklariert werden. Dies geschieht durch das deklarieren der Variablen

außerhalb einer Klasse oder Funktion wie in der oben zu sehenden Abbildung (siehe Abbildung 145).

8.2.4. Ein- und Ausgänge Setzen

```
//========================================== Setup

void setup() {

 pinMode(IND1, INPUT_PULLUP);
 pinMode(IND2, INPUT_PULLUP);
 pinMode(COIN, INPUT_PULLUP);
 pinMode(END, INPUT_PULLUP);
 pinMode(MOTOR, OUTPUT);
 pinMode(RELAI, OUTPUT);
 pinMode(dirPIN, OUTPUT);
 stepper.setMaxSpeed(1000);
 stepper.setAcceleration(200);

 digitalWrite(RELAI, HIGH);

 Serial.begin(9600);
 SPI.begin();
 for (byte i = 0; i < 6; i++) {
        key.keyByte[i] = 0xFF;
    }
    dump_byte_array(key.keyByte, MFRC522::MF_KEY_SIZE);
    Serial.println();

 RFID();
 ref();
}
```

Pin-Art zuweisen

Serial Port aktivieren

RFID einrichten

Abbildung 146: Arduino Ein- und Ausgänge

Um die deklarierten Pins benutzen zu können muss ihnen noch die Betriebsart zugewiesen werden (siehe Abbildung 146). Als Betriebsarten gibt es INPUT, INPUT_PULLUP und OUTPUT. Beim INPUT_PULLUP wird der im Arduino interne Pull-Up Widerstand aktiviert, weswegen kein externer Widerstand von Nöten ist. Beim INPUT ist dies nicht der Fall. Weiters wird der serielle Port aktiviert, was das Kommunizieren mit dem Raspberry Pi ermöglicht. Damit der RFID Leser funktioniert muss dieser neu eingerichtet werden, wie im oberen Bild unten zu sehen ist (siehe Abbildung 146).

8.2.5. RFID-Leser Initialisieren

```
//====================

void RFID() {

  mfrc522.PCD_Init();
}
```

Abbildung 147: RFID Initialisieren

Damit der RFID einsatzbereit ist muss er vorher initialisiert werden (siehe Abbildung 147). Dies bewirkt, dass der RFID mit Lesen von Chips oder Karten beginnt, sofern diese in der Nähe sind.

8.2.6. Stepper Referenzieren

```
//==================================

void ref() {
  ENDval = digitalRead(END);
  if (!ENDval) {
    while (!ENDval) {
      ENDval = digitalRead(END);
      stepper.setSpeed(-50);
      stepper.runSpeed();
    }
    digitalWrite(A1, HIGH);
    stepper.setCurrentPosition(0);
    RFID();
  }
  digitalWrite(A1, HIGH);
}
```

Abbildung 148: Stepper Referenzieren

Um den Stepper verwenden zu können muss dieser vorher referenziert werden. Dies wird erreicht, indem sich der Stepper so lange zurückdreht, bis dieser gegen die Endlage fährt und so den Kontakt schließt. Beim Schließen des Kontakts wird der Stepper gestoppt und seine aktuelle Position auf 0 gesetzt (siehe Abbildung 148).

8.3. Lesen und schreiben von Signalen

8.3.1. Positions-Sensor Walzen

```
//=====================================================

  IND1val = digitalRead(IND1);
  if (!IND1val & !IND1mem)
    {
      if(INDcounter == 12 || IND1reset)
        {
      INDcounter = 0;
      IND1reset = 0;
        }
      IND1mem = 1;
      INDcounter += 1;
      Serial.println("IND1-" + String(INDcounter));
    }
  if(IND1val)
    {
      IND1mem = 0;
    }
```

Abbildung 149: Position Walzen lesen

Um die aktuelle Stellung der Walzen zu erkennen wird das entsprechende Sensorsignal durch die Funktion digitalRead() vom Arduino Pin gelesen (siehe Abbildung 149). Beim Erhalten eines Signals wird der Zähler um 1 erhöht so lange, bis eine Umdrehung erreicht ist. Damit nach dem Start der Maschine die Position festgelegt werden kann wird der Positions-Sensor durch den Referenz-Sensor kalibriert.

8.3.2. Referenz-Sensor Walzen

```
//=================================

  IND2val = digitalRead(IND2);
  if (!IND2val & !IND2mem)
    {
      Serial.println("IND2");
      IND1reset = 1;
      IND2mem = 1;
    }
  if(IND2val)
    {
      IND2mem = 0;
      IND1reset = 0;
    }
```

Abbildung 150: Walzen Referenz lesen

Damit die Position der Walzen nach dem Start der Maschine festgelegt werden kann braucht es einen Referenz-Sensor. Das Signal dieses Sensors erhält man wie jenes in Kapitel 8.3.1. Solange der Sensor ein Signal gibt, ist die reset-Variable aktiv (siehe Abbildung 150). Erhält nun der Positions-Sensor ein Signal, während die Rücksetzvariable aktiv ist wird der Positions-Sensor referenziert.

8.3.3. Endlage Stepper

```
//=====================================

  ENDval = digitalRead(END);
  if (ENDval & !ENDmem)
    {
      Serial.println("END1");
      ENDmem = 1;
      stepper.setCurrentPosition(0);
    }
  if(!ENDval & ENDmem)
    {
      ENDmem=0;
      delay(1);
    }
```

Abbildung 151: Endlage Stepper lesen

Um die Position des Steppers nach jedem Einsatz zu kalibrieren, wird ein Endlagensensor eingesetzt. Das Lesen des Signals erfolgt gleich wie in Kapitel 8.3.1. Bei Erhalt eines Signales wir die aktuelle Position des Steppers zurückgesetzt (siehe Abbildung 151).

8.3.4. Münzprüfeinheit

```
//=========================================

  COINval = digitalRead(COIN);
  if(COINval and steps==0)
    {
      Serial.println("Coin inserted");
      previousMillis_timeout = millis();
      steps=10;
    }
```

Abbildung 152: Münzprüfeinheit lesen

Damit erkannt wird, wann eine Münze eingeworfen wurde, wird eine Münzprüfeinheit benötigt welche ermittelt, ob eine Münze eingeworfen wurde. Die Ermittlung des Signales erfolgt gleich wie in Kapitel 8.3.1. Wird ein Signal gelesen, so wird der Ablauf gestartet, weiters wird der Timer für den Motor zurückgesetzt (siehe Abbildung 152).

8.3.5. RFID

```
//========================================== Authenticate RFID

  if ( ! mfrc522.PICC_IsNewCardPresent())
    return;

  if ( ! mfrc522.PICC_ReadCardSerial())
    return;

  status = ((MFRC522::StatusCode) mfrc522.PCD_Authenticate
  (MFRC522::PICC_CMD_MF_AUTH_KEY_A, trailerBlock, &key, &(mfrc522.uid)));
  status = (MFRC522::StatusCode) mfrc522.MIFARE_Read(blockAddr, buffer, &size);

  dump_byte_array(buffer, 16);

  mfrc522.PICC_HaltA();
  mfrc522.PCD_StopCrypto1();
}

//========================================== RFID read

void dump_byte_array(byte *buffer, byte bufferSize)
{
  for (byte i = 0; i < bufferSize; i++)
    {
      addzero = (buffer[i] < 0x10 ? " 0" : " ");
      code = (buffer[i]);
      pass += addzero;
      pass += code;

      if (i ==  (bufferSize-1))
        {
        Serial.println(pass);
        pass = "";
        }
    }
}
```

Abbildung 153: RFID lesen

Um eine Karte oder einen Chip mit dem RFID Leser lesen zu können muss zuerst geprüft werden, ob ein neuer RFID-Tag in der Nähe ist. Ist dies der Fall wird der RFID-Tag authentifiziert, wobei der Buffer von der ausgewählten Adresse des Tags gelesen wird (siehe Abbildung 153). Damit man die Daten aus dem Buffer erhält wird über alle Items des Buffers iteriert. Beim Iterieren werden die Daten in eine Variable geschrieben, welche ausgelesen werden kann.

8.3.6. Serial

```
//=========================================

void serial()
{
  if (Serial.available()>0)
   {
    data = Serial.readStringUntil('\n');
    Serial.println(data + "A");
   }
}
```

definiertes Symbol

Abbildung 154: serielle Schnittstelle lesen

Damit die Daten vom Raspberry verwendet werden können müssen diese vom Serial Port gelesen werden. Dies erfolgt mit der Funktion Serial.readUntil(), welche die Daten bis zu einem definierten Symbol liest (siehe Abbildung 154).

```
//=================================================

  serial();
  subLoop = data.substring(0, 2);
  if (subLoop == "TO")
    {
      MOTORtimeout = (data.substring(3)).toInt();
      data ="";
    }

  else if (subLoop == "SM")
    {
      MOTORspeed = (data.substring(3)).toInt();
      data ="";
    }

  else if (subLoop == "SS")
    {
      STEPPERspeed = (data.substring(3)).toInt();
      data ="";
    }

  else if (data == "Motor-right" && steps == 0)
    {
      digitalWrite(dirPIN, HIGH);
      analogWrite(MOTOR, MOTORspeed);
      data = "";
    }

  else if (data == "Motor-left" && steps == 0)
    {
      digitalWrite(dirPIN, LOW);
      analogWrite(MOTOR, MOTORspeed);
      data = "";
    }

  else if (data == "Motor-stop" && steps == 0)
    {
      analogWrite(MOTOR, 0);
      data = "";
    }
```

Aufteilung in Substrings

Abbildung 155: Arduino Daten auswerten

Um die gelesenen Daten verwenden zu können muss der Datenstring in kleine Substrings aufgeteilt werden, sofern mehrere Informationen gleichzeitig gesendet werden. Dies erfolgt mit der Funktion .substring() wie in der obigen Abbildung (siehe Abbildung 155). Die aufbereiteten Daten werden dann in Variablen gespeichert.

8.4. Ablauf des Arduino-Programmes

8.4.1. Schritt 10

```
switch(steps)
  {
    case 10:
      Serial.println("choose design");
      analogWrite(MOTOR, MOTORspeed);
      digitalWrite(dirPIN, HIGH);
      digitalWrite(RELAI, LOW);
      steps=20;
    break;
```

Abbildung 156: Schritt 10

Wird eine Münze eingeworfen so startet der Ablauf in Schritt 10 (siehe Abbildung 156). In diesem Schritt wird der Motor zum Prägen der Münzen eingeschaltet. Außerdem wird das Relais deaktiviert. Dies hat zur Folge, dass keine weiteren Münzen eingeworfen werden können.

8.4.2. Schritt 20

```
case 20:
  serial();
  subAblauf = data.substring(0, 4);
  if (subAblauf == "Data")
    {
      design = (data.substring(5, 6)).toInt();
      offset = (data.substring(7)).toInt();
      Serial.println("design " + String(design) +" chosen | offset: "+ String(offset));
      steps=30;
      data = "";
    }
break;
```

Abbildung 157: Schritt 20

Wird ein Design auf dem Interface ausgewählt, so werden die Daten an den Arduino gesendet. Dieser liest wie in der obigen Abbildung (siehe Abbildung 157) die Daten und ordnete sie den entsprechenden Variablen zu.

8.4.3. Schritt 30

```
case 30:
  if(INDcounter/2 == design)
    {
      previousMillis = millis();
      steps=40;
    }
break;
```

Abbildung 158: Schritt 30

Wenn die Position der Walzen dem ausgewählten Design entspricht, so wird ein Timer gestartet. Dieser misst wie viele Millisekunden seit der Erfüllung der Bedienung vergangen sind (siehe Abbildung 158).

8.4.4. Schritt 40

```
case 40:
  currentMillis = millis();
  if (currentMillis - previousMillis >= offset)
    {
      steps = 50;
    }
break;
```

Abbildung 159. Schritt 40

Wenn die vergangen Millisekunden seit der Erfüllung der Bedienung mit dem vorgesehenen Offset des jeweiligen Designs übereinstimmen so wird der nächste Schritt aufgerufen (siehe Abbildung 159).

8.4.5. Schritt 50:

Definierte Position [Angabe in Stepper-schritte]

```
case 50:
  digitalWrite(A1, LOW);
  while (stepper.currentPosition() != 288)
    {
      stepper.setSpeed(STEPPERspeed);
      stepper.runSpeed();
    }
  previousMillis = millis();
  steps = 60;
break;
```

Abbildung 160: Schritt 50

Ist der Timer abgelaufen, so schwenkt der Stepper zu den Walzen. Dies tut der Stepper so lange bis er eine definierte Position erreicht hat (siehe Abbildung 160). Ist die Position erreicht wird ein neuer Timer gestartet und der nächste Schritt aufgerufen.

8.4.6. Schritt 60

```
case 60:
  currentMillis = millis();
  if (currentMillis - previousMillis >= 5000)
    {
      previousMillis = millis();
      steps = 70;
    }
break;
```

Abbildung 161: Schritt 60

Der in der obigen Abbildung gezeigte Schritt (siehe Abbildung 161) erfüllt den gleichen Zweck wie jener in Kapitel 8.4.4.

8.4.7. Schritt 70

```
case 70:
  while (!ENDval)
    {
      ENDval = digitalRead(END);
      stepper.setSpeed(-STEPPERspeed);
      stepper.runSpeed();

      if (stepper.currentPosition() == -10)
        {
          digitalWrite(A1, HIGH);
          Serial.println("Stepper fertig");
          digitalWrite(RELAI, HIGH);
          steps = 0;
          break;
        }
    }

  if (ENDval)
    {
      Serial.println("Stepper fertig");
      digitalWrite(A1, HIGH);
      stepper.setCurrentPosition(0);
      digitalWrite(RELAI, HIGH);
      steps=0;
    }
  break;

  default:
  break;
}
```

Fehlerfall

Normalfall

Abbildung 162: Schritt 70

Ist der Timer abgelaufen so schwenkt der Stepper wieder zurück. Der Stepper schwenkt so lange zurück, bis er den Endschalter erreicht und so den Kontakt schließt (siehe Abbildung 162). Würde in einem Fehlerfall der Endlagenschalter nicht funktionieren fährt der Stepper bis zu einer, durch einen Anschlag, definierten Position zurück.

9. Interface

9.1. Python

Zur Programmierung des Raspberry Pi's wird die Programmiersprache Python verwendet. Python ist eine interpretierte, objektorientierte Programmiersprache mit dynamischer Semantik. Durch die einfache Syntax ist Python leicht zu erlernen und gut lesbar. Es unterstützt eine Vielzahl von Bibliotheken und ist auf allen wichtigen Plattformen erhältlich.

9.1.1. Klassen

Eine Klasse dient zu Beschreiben von gemeinsamen Strukturen und dessen Verhalten. Sie helfen den Code zu ordnen, indem reale Eigenschaften in Klassen eingeteilt werden. Klassen dienen als Baupläne, welche wiederverwendet werden können.
Vgl. (Python Klassen, 2021)

9.1.2. Funktion

Eine Funktion ist eine Gruppe von Anweisungen, die eine bestimmte Aufgabe erfüllen. Funktionen dienen zum Aufteilen des Codes in kleine wiederverwendbare Module. Sie fördern die Übersichtlichkeit und die Handhabung.
Vgl. (Function)

9.1.3. Vererbung

Vererbung ermöglicht das Verwenden von Methoden und Objekten von definierten Klassen. Es können übergeordnete Klassen sowie untergeordnete Klassen vererbt werden. Vererbung dient zur Vereinfachung des Codes durch Verwendung von bereits erstellten Bauplänen.
Vgl. (Python Inheritance)

9.2. Auswahl der GUI Bibliothek

9.2.1. Tkinter

Tkinter ist die Standard GUI-Toolkit Bibliothek für Python zum Erstellen von grafischen Benutzeroberflächen.

Vorteile:

- Kostenlos
- Python Standard Bibliothek
- Einfache und schnelle Handhabung
- Stabil

Nachteile:

- GUI schwer skalierbar
- Wenig Features
- Veraltet

Vgl. (PYTHON-GUI - PYQT VS TKINTER)

9.2.2. Kivy

Kivy ist eine cross Plattform Python Bibliothek zum Erstellen von Interfaces und multi-touch apps. Kivy verbindet Python mit der eigenen Kivy Programmiersprache. Dies ermöglicht es einfach Widgets und diverse Features in Python zu integrieren.

Vorteile:

- Kostenlos
- Cross Plattform kompatibel auch IOS und Android
- Skalierbare GUIs
- Viele Features
- Stabil

Nachteile:

- Mischung aus Python und kV-Sprache
- Komplexere Einrichtung

9.2.3. Lösungsansatz

Als Lösung hat sich das Projektteam für Kivy entschieden da es über viele neue und innovative Features verfügt, welche ein modernes Design ermöglichen. Außerdem ist das Interface nicht an eine definierte Auflösung gebunden. Dies ermöglicht es verschieden Bildschirme zur Darstellung zu verwenden.

9.3. .kV

.Kv ist die Kivy eigene Programmiersprache. Sie ermöglicht es durch einfache Befehle Widgets in einem vordefinierten Layout zu platzieren. Weiters können die Widgets beliebig gestaltet und entsprechend dem gewählten Layout platziert werden. Dies wird auch verwendet um das Interface zu designen, der Code hierfür wird aber nicht näher erläutert da er zu unübersichtlich und unrelevant ist.

9.3.1. Layouts

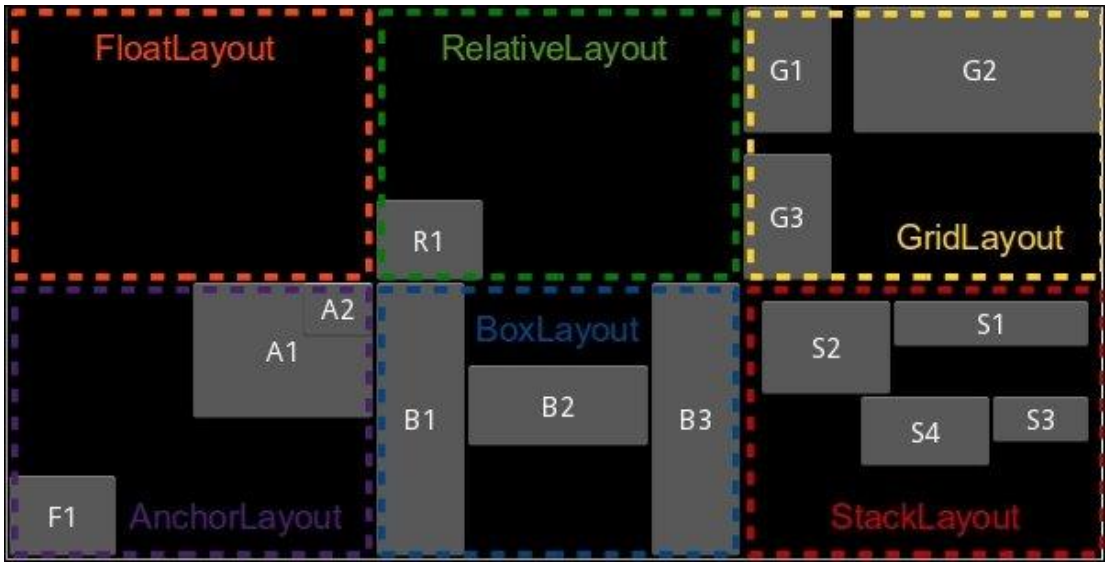


Abbildung 163. Layouts (https://subscription.packtpub.com/book/application-development/9781785286926/1/ch01lvl1sec11/embedding-layouts)

9.3.1.1. FloatLayout

In einem FloatLayout sind die Widgets frei bewegbar (siehe Abbildung 163). Sie können aber anhand von Pixeln fest positioniert werden.
vgl. (Float Layout)

9.3.1.2. RelativeLayout

Auf dem RelativeLayout lassen sich Widgets frei platzieren (siehe Abbildung 163). Die Widgets werden relativ zum übergeordneten Objekt platziert anhand von prozentuellen Angaben.
vgl. (Relative Layout)

9.3.1.3. GridLayout

Das GridLayout ordnet die Widgets anhand einer Matrix an (siehe Abbildung 163). Die Matrix wir durch Angabe von Reihen und Zeilen definiert.
vgl. (Grid Layout)

9.3.1.4. AnchorLayout

Das AnchorLayout richtet die Widgets anhand von Bildschirmgrenzen an (siehe Abbildung 163). Dies sind beispielweise die Bildschirm Ober- und Untergrenze.
vgl. (Anchor Layout)

9.3.1.5. BoxLayout

In einem Boxlayout werden die Widgets anhand einer Box horizontal oder vertikal angeordnet (siehe Abbildung 163).
vgl. (Box Layout)

9.3.1.6. StackLayout

Das StackLayout ordnet so viele Widgets wie möglich horizontal oder vertikal an, dabei werden die Widgets nicht verändert (siehe Abbildung 163).
vgl. (Stack Layout)

9.3.1.7. PageLayout

Das PageLayout ermöglicht es mehrere Seiten zu erstellen, indem Widgets oder Layouts zu einzelnen Seiten konvertiert werden.

9.3.2. Widgets

9.3.2.1. Label

Abbildung 164: Label (https://kivy.org/doc/stable/api-kivy.uix.label.html)

Das Label (siehe Abbildung 164) ist eine TextBox die zum darstellten von Texten verwendet wird. Die Texte können in der TextBox beliebig platziert sowie formatiert werden.

Vgl. (Label)

9.3.2.2. Button

Abbildung 165: Button (https://kivy.org/doc/stable/api-kivy.uix.button.html)

Der Button (siehe Abbildung 165) ist ein Knopf, welcher beim Drücken oder Loslassen eine Aktion aufruft. Durch Bilder kann der Knopf auch beliebig gestaltet werden.

Vgl. (Button)

9.3.2.3. Toggle Button

Abbildung 166: Toggle Button (https://kivy.org/doc/stable/api-kivy.uix.togglebutton.html)

Der Toggle Button (siehe Abbildung 166) ist ein Ankreuzfeld, welches beim Anklicken seinen Status ändert. Mehrere Ankreuzfelder können auch zu einem Optionsfeld gruppiert werden. Den Ankreuzfeldern kann ein beliebiges Design zugeordnet werden mithilfe von Bildern.

Vgl. (Toggle Button)

9.3.2.4. Switch

Abbildung 167: Switch (https://kivy.org/doc/stable/api-kivy.uix.switch.html)

Der Switch (siehe Abbildung 167) ist ein Schalter, welcher durch Wischen EIN beziehungsweise AUS geschaltet werden kann. Das Aussehen des Schalters kann durch Grafiken verändert werden.

Vgl. (Switch)

9.3.2.5. Slider

Abbildung 168: Slider (https://kivy.org/doc/stable-1.11.1/api-kivy.uix.slider.html)

Der Slider (siehe Abbildung 168) ist ein Schieber mit dem Werte in einem vorher fest definierten Bereich eingestellt werden können. Durch Bilder kann das Aussehen des Schiebers verändert werden.

Vgl. (Slider)

9.4. Setup des Interfaces

9.4.1. Erzeugung des Interfaces

```
# ====================================

class WindowManager(ScreenManager):
    Window.fullscreen = "auto"

# ====================================

class Interface(App):
    def build(self):
        pass


if __name__ == "__main__":
    Interface().run()
```

WindowManager

Builder

Abbildung 169: Raspberry Pi Builder und Windowmanager

Damit ein Interface ensteht braucht man einen Builder (siehe Abbildung 169). Dieser erzeugt mithilfe der .kV File das Interface (siehe 9.3). Der WindowManager hat die Aufgabe die verschiedenen Fenster und ihre Animationen zu verwalten.

9.4.2. Bibliotheken

```
# ==========================================================================

from kivy.app import App
from kivy.properties import StringProperty, BooleanProperty, NumericProperty
from kivy.uix.screenmanager import ScreenManager, Screen
from kivy.core.window import Window
from kivy.clock import Clock
import serial
```

Abbildung 170: Raspberry Pi Bibliotheken

Um Bibliotheken in Python verwenden zu können müssen sie zuerst installiert werden. Ist dies geschehen müssen die verschiedenen Bibliotheken in das Interfaceprogramm geladen werden (siehe Abbildung 170). Im Fall des Projektteams werden „Kivy“

Bibliotheken sowie die „Serial“ Bibliothek gebraucht, um das Interface zu erstellen und um die Kommunikation mit dem Arduino zu ermöglichen.

9.4.3. Serielle Kommunikation

```
# ==========================================================

ser = serial.Serial('/dev/ttyACM0', 9600, timeout=1)
ser.reset_input_buffer()
```

Abbildung 171: Raspberry Pi serielle Kommunikation

Als Steuerungshardware der Münzprägemaschine wird wie bereits in Punkt 6.1.4 beschrieben eine Kombination aus Arduino und Raspberry Pi verwendet. Die Kommunikation dieser beiden Komponenten wird über einen seriellen Bus ermöglicht. Hierfür muss dem Raspberry Pi der Port sowie die Baudrate zugewiesen werden (siehe Abbildung 171**)**. Den Port erhält man durch die Eingabe des Befehls „ls /dev/tty*“ in der Befehlszeile, die Baudrate ist selbst zu wählen muss aber bei beiden Geräten übereinstimmen. Die Baudrate sagt aus wie oft ein Signal pro Sekunde seinen Zustand ändert vgl. (Eckert, 2015).

9.4.4. Globale Variablen

```
# ==============================================================

select = 1
coin = False
process = False
if str(open(f"temp/GIF", "r").read()).strip() == "True":
    GIF = True
else:
    GIF = False
```

Abbildung 172: Raspberry Pi globale Variablen

Damit die Variablen in jedem Abschnitt des Programmes benutzt werden können müssen sie als gloable Variablen definiert werden (siehe Abbildung 172). In Python werden diese erst später an der gwünschten Stelle globalisiert mit dem Befehl „global“.

9.4.5. Fenster erstellen

```
# ==================================================================

class MainPage(Screen):

# ==================================================================

    design_selected = StringProperty("designs/design_Austria.png")
    background_design = StringProperty("images/background_Austria.png")
```

Abbildung 173: Raspberry Pi Fenster erstellen

Damit ein Fenster erstellt wird, braucht es eine Klasse, die von der Bibliothek „Screen“ erbt (siehe Abbildung 173). Dies ermöglicht es den Fenstermanager das erstellte Fenster MainPage zu verwalten. Klassen verfügen über Klassenvariablen wie in Abbildung 173 unten, das sind Variablen, die von jeder Stelle der Klasse aufgerufen werden können.

9.4.6. Initialisierung von Objekten

```
def __init__(self, **kwargs):
    super(DevPage, self).__init__()

    Clock.schedule_interval(self.serial, 0.1)
    Clock.schedule_once(self.serial, 5)

    Clock.schedule_once(self.slider_init, 1)

    if GIF:
        self.toggle1_state = "normal"
    else:
        self.toggle1_state = "down"

    self.on_toggle_button_state(self.ids.toggle_1)
```

Abbildung 174: Raspberry Pi Initialisierung von Objekten

Wenn ein Objekt in einer Klasse erstellt wird, wird die „__init__“ Funktion aufgerufen (siehe Abbildung 174). Die Funktion ermöglicht es der Klasse die Attribute des Objektes zu initialisieren. Dies geschieht beispielsweise beim Start des Programmes.

Damit die „__init__“ Funktion auf die Objekte der übergeordnete Klasse „DevPage“ zugreifen kann ohne sie zu referenzieren wird die Funktion „super()“ gebraucht. Diese ermöglicht es die Objekte der übergeordneten Klasse zu vererben.

9.5. Ablauf des Interfaces

```
==========================================================================

def serial(self, dt):
    global coin

==========================================================================

    if ser.inWaiting() > 0:
        self.data = ser.readline().decode('utf-8', errors="ignore").strip()
        self.data = str(self.data)
        print(self.data)

==========================================================================

    if self.data == self.encoding:
        self.data = ""
        if self.dev == False:
            App.get_running_app().root.current = "dev"
            self.dev = True
        else:
            self.rfid_disabled()
            self.dev = False

==========================================================================

    if self.data == "Coin inserted":
        self.data = ""
        coin = True
        Waiting().coin_check()

==========================================================================

    if self.data == "Stepper fertig":
        Processing().timeover(self)
    data2 = self.data.split("-")
```

Daten lesen

Daten dekodieren und Auswerten

Abbildung 175:Raspberry Pi serielle Schnittstelle lesen

Um den Interfaceablauf realisiert zu realisieren müssen Ablaufdaten vom Arduino über den seriellen Port übernommen werden. Dies geschieht mithilfe der Methode ser.readline(). Da die Daten noch mit UTF-8 kodiert sind, müssen diese dekodiert werden. Die Daten werden im Anschluss in der Funktion „serial“ mit verschiedenen Parametern verglichen (siehe Abbildung 175). Stimmt der Parameter mit den Daten

überein werden die Information an die entsprechenden Funktionen weitergegeben (siehe Abbildung 175). Diese Funktionen steuern den Interface Ablauf können aber auch die Einstellungen aktivieren.

9.5.1. Wartebildschirm

```
# ==========================================================================================

    def gif_Waiting(self, dt):

        global GIF

        if GIF and App.get_running_app().root.current == "wait":

            if self.i <= 6:
                self.i += 1
            else:
                self.i = 0

            self.Waiting_gif = "images/gif_Waiting/Coin_waiting-" + str(self.i) + ".jpg"
        else:
            self.Waiting_gif = "images/gif_Waiting/Coin_waiting-0.jpg"
```

Abbildung 176: Wartebildschirm

Während die Maschine auf eine Münze wartet, wird auf dem Display ein GIF abgespielt, sofern es nicht in den Einstellungen deaktiviert wurde. Das GIF wird durch eine Aneinanderreihung von Bildern erzeugt. Dies wird durch das Feature Kivy Clock realisiert, welches die Funktion „gif_Waiting“ in einem definierten Intervall aufruft (siehe Abbildung 176).

9.5.2. Münze eingeworfen

```
# ==========================================================

    def coin_check(self):

        if App.get_running_app().root.current == "wait":
            App.get_running_app().root.current = "main"
```

Abbildung 177: Münze eingeworfen

Wird eine Münze eingeworfen so wird dies über den seriellen Port an den Raspberry Pi weitergegeben. Dieser ruft die Funktion „coin_check“ auf, welche die Anzeigeauf die Designauswahl umschaltet (siehe Abbildung 177).

9.5.3. Designauswahl

Abbildung 178: MainPage

```
# ==================================================================

    def show_img(self, direction):
        global select

# ==================================================================

        if direction == "minus":
            if select == 1:
                select = 6
            else:
                select -= 1
        if direction == "plus":
            if select == 6:
                select = 1
            else:
                select += 1

# ==================================================================

        if select == 1:
            self.design_selected = "designs/design_Austria.png"
            self.background_design = "images/background_Austria.png"

        elif select == 2:
            self.design_selected = "designs/design_Germany.png"
            self.background_design = "images/background_Germany.png"

        elif select == 3:
            self.design_selected = "designs/design_Spain.png"
            self.background_design = "images/background_Spain.png"

        elif select == 4:
            self.design_selected = "designs/design_Norway.png"
            self.background_design = "images/background_Norway.png"

        elif select == 5:
            self.design_selected = "designs/design_Switzerland.png"
            self.background_design = "images/background_Switzerland.png"

        elif select == 6:
            self.design_selected = "designs/design_Luxenburg.png"
            self.background_design = "images/background_Luxenburg.png"
```

Neuen Wert ermitteln

Design und Hinter-grund ändern

Abbildung 179: Designauswahl

Wird eine der beiden Knöpfe auf der MainPage (siehe Abbildung 179) gedrückt so wird die Funktion „show_img“ aufgerufen (siehe Abbildung 179). Die Funktion wertet den übermittelten Wert für die Richtung aus und schreibt ihn auf die Variable „select“.

Durch die Wertänderung der Variable wird ein neues Design und der entsprechende Hintergrund ausgewählt.

9.5.4. Popup

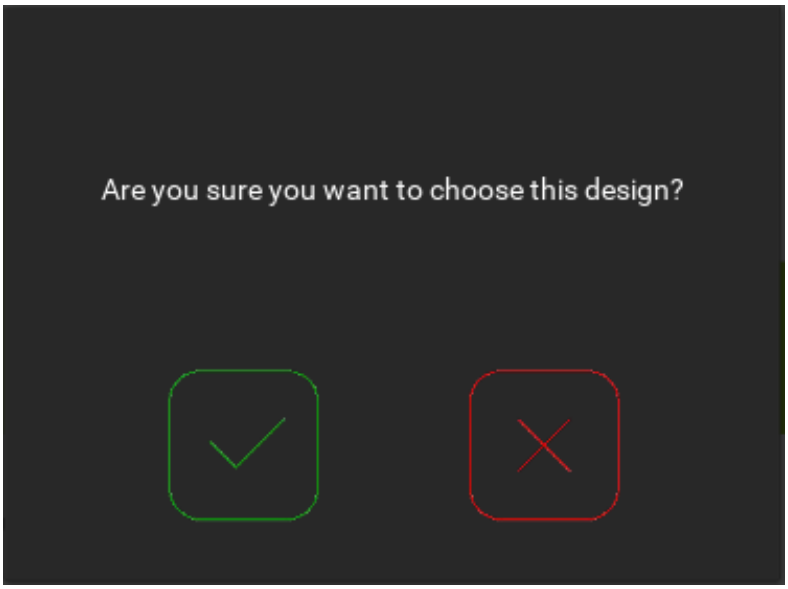


Abbildung 180: Popup

Wird der Startknopf auf der MainPage (siehe Abbildung 178) gedrückt so erscheint ein Popup (siehe Abbildung 180). Wenn das Kreuz gedrückt wird, erscheint wieder die MainPage (siehe Abbildung 178). Beim Druck auf das Häkchen wird fortgefahren.

9.5.5. Datenübertragung

```
# ==========================================================================

    def cycle_start(self):
        global select
        global process

        process = True
# ==========================================================================

        if select == 1:
            delay = (str(open(f"temp/Input1", "r").read())).strip()
        elif select == 2:
            delay = (str(open(f"temp/Input2", "r").read())).strip()
        elif select == 3:
            delay = (str(open(f"temp/Input3", "r").read())).strip()
        elif select == 4:
            delay = (str(open(f"temp/Input4", "r").read())).strip()
        elif select == 5:
            delay = (str(open(f"temp/Input5", "r").read())).strip()
        elif select == 6:
            delay = (str(open(f"temp/Input6", "r").read())).strip()

# ==========================================================================
        data = ("Data-" + str(select) + "-" + str(delay) + "\n").encode()

        ser.write(data)
```

Abbildung 181: Datenübertragung

Wird das Häckchen auf dem Popup (siehe Abbildung 180) gedrückt so wird die Funktion „cycle_start“ aufgerufen (siehe Abbildung 181). Die Funktion ermittelt das ausgewählte Design und bestimmt die dazugehörige Zeitverzögerung. Damit die Daten über den seriellen Port übermittelt werden können, müssen sie vorher mit UTF-8 kodiert werden, ansonsten würde ein UnicodeError entstehen. UTF-8 steht für 8-Bit Universal Coded Character Set Transformation Format und dient zum kodieren von Unicode Zeichen vgl. (UTF-8, 2021). Dies wird durch die Methode „.encode()“ ermöglicht.

9.5.6. Prozessbildschirm

```
==========================================================================

def gif_Processing(self, dt):

    global GIF

==========================================================================
==========================================================================

    if GIF and App.get_running_app().root.current == "pro":

        if self.i <= 58:
            self.i += 1
        else:
            self.i = 0

        self.Processing_gif = "images/gif_Process/Process-" + str(self.i) + ".jpg"
    else:
        self.Processing_gif = "images/gif_Process/Process-0.jpg"
```

Abbildung 182: Prozessbildschirm

Während die Maschine eine Münze prägt, wird auf dem Display ein GIF abgespielt (siehe Abbildung 182), sofern es nicht in den Einstellungen deaktiviert wurde. Dies wird wie in Kapitel 9.5.1 realisiert.

9.5.7. Maschine fertig

```
# ==================================================

    @staticmethod
    def timeover(self):
        global coin
        global process

# ==================================================

        App.get_running_app().root.current = "wait"
        coin = False
        process = False
```

Abbildung 183: Maschine fertig

Ist die Münzprägemaschine mit dem Prägevorgang fertig, so wird die Funktion „timeover“ aufgerufen (siehe Abbildung 183). Diese setzt die Hilfsvariablen zurück und wechselt das Fenster auf den Wartebildschirm.

9.6. Einstellungen

Abbildung 184: DevPage Page 1

Abbildung 185:DevPage Page 2

Damit äußere Einflüsse sowie Fehler leicht behoben werden können, gibt es eine developer Seite (siehe Abbildung 184 und Abbildung 185). Auf dieser können die Geschwindigkeiten des Steppers und auch die des Motors variabel angepasst werden. Weiters lässt sich einstellen, wie lange der Motor nachlaufen soll, bis er abgedreht wird. Bei Leistungsproblemen lassen sich hier auch die Animationen deaktivieren. Um Fehler wie feststeckende Münzen leicht zu beheben, gibt es auf der developer Seite Knöpfe zum manuellen Verfahren des Motors. Da durch Toleranzen und äußere Einflüsse Fehler beim Timing der Münzeinführung in die Walzen entstehen können, lassen sich außerdem variable Verzögerungen einstellen.

9.6.1. Vollbild

```
# ==========================================

    @staticmethod
    def fullscreen_enabled(widget):
        if widget.active:
            Window.fullscreen = "auto"
        else:
            Window.fullscreen = False
```

Abbildung 186: Vollbild

Die Funktion „fullscreen_enabled“ ermöglicht es durch einen Schalter auf der Devpage den standardmäßigen Fullscreen Modus EIN beziehungsweise AUS zu schalten (siehe Abbildung 186).

9.6.2. Animationen

```
# ==================================================
# ==================================================

    def on_toggle_button_state(self, widget):

        global GIF

        if widget.state == "down":
            self.Label1_enabled = False
            self.Label1_text = "Disabled"
            GIF = False

            with open("temp/GIF", "w") as file:
                file.write(str(GIF))
            file.close()

        else:
            self.Label1_enabled = True
            self.Label1_text = "Enabled"
            GIF = True

            with open("temp/GIF", "w") as file:
                file.write(str(GIF))
            file.close()
```

Abbildung 187: Animationen

Durch das Umschalten des GIF Umschaltknopfes auf der Devpage wird die Funktion „on_toggle_button_state“ aufgerufen (siehe Abbildung 187). Diese Funktion prüft, ob der GIF Schalter deaktiviert oder aktiviert ist und wechselt anhand dessen den Status der GIFs. Weiters wird der neue Status in einer Datei gespeichert, um den Status nach einem Neustart abfragen zu können.

9.6.3. Geschwindigkeiten und Nachlaufzeit

```
# ==========================================================================

    def save_slider(self, slider):
        if slider == "TO":
            with open("temp/Slider_TO", "w") as file:
                file.write(str(int(self.ids.TO.value)))
            file.close()
            status = ("TO-" + str(int(self.ids.TO.value)) + "\n").encode()

        elif slider == "SM":
            with open("temp/Slider_SM", "w") as file:
                file.write(str(int(self.ids.SM.value)))
            file.close()
            status = ("SM-" + str(int(self.ids.SM.value)) + "\n").encode()

        elif slider == "SS":
            with open("temp/Slider_SS", "w") as file:
                file.write(str(int(self.ids.SS.value)))
            file.close()
            status = ("SS-" + str(int(self.ids.SS.value)) + "\n").encode()
        ser.write(status)
```

Abbildung 188: Geschwindigkeiten und Nachlaufzeit

Wird einer der Slider auf der Devpage verschoben, so wird die Funktion „save_slider“ aufgerufen (siehe Abbildung 188). Diese wählt anhand der übertragenen Daten den richtigen Slider aus und speichert den Wert in eine Datei. Weiters wird der Wert kodiert und an den Arduino gesendet. Außerdem wir bei jedem Neustart der Maschine der kodierte Wert an den Arduino gesendet.

9.6.4. Manuelle Motorsteuerung

```
# ==================================================

    def motor_turn_l(self):
        command = ("Motor-left" + "\n").encode()
        ser.write(command)

# ==================================================

    def motor_turn_r(self):
        command = ("Motor-right" + "\n").encode()
        ser.write(command)

# ==================================================

    def motor_stop(self):
        command = ("Motor-stop" + "\n").encode()
        ser.write(command)
```

Abbildung 189: Manuelle Motorsteuerung

Wird einer der beiden Knöpfe zum manuellen Verfahren des Motors auf der Devpage gedrückt, wird die entsprechende Funktion „motor_turn_l“ oder „motor_turn_r“ aufgerufen (siehe Abbildung 189). Diese Funktionen senden den Befehl direkt an den Arduino. Wenn der gedrückte Knopf ausgelassen wird, wird die Funktion „motor_stop“ ausgeführt, welche den Befehl auf den seriellen Port schreibt.

9.6.5. Zeiteingabefeld

9.6.5.1. Schreiben

```
# ==================================================================

    def write(self, Number):
        if self.Input1_enabled and len(self.Input1_text) < 5:
            if self.Input1_text == "0":
                self.Input1_text = self.Input1_text[:-1]
            self.Input1_text += Number
```

Abbildung 190: Zeiteingabefeld schreiben

Wird auf dem Numpad, welches sich auf der Devpage befindet, einer der Zahlenknöpfe gedrückt wird die Funktion „write“ aufgerufen (siehe Abbildung 190). Die Funktion prüft welche der Inputboxen aktiviert ist und ob die maximale Zeichenanzahl schon erreicht ist. Falls bei der entsprechenden Inputbox eine 0 eingetragen ist wird diese gelöscht. Danach wird die Zahl welche übermittelt wurde in die Inputbox eingetragen.

9.6.5.2. Löschen

```
# ==================================================================

    def erase(self):
        if self.Input1_enabled and len(self.Input1_text) >= 1:
            self.Input1_text = self.Input1_text[:-1]
```

Abbildung 191: Zeiteingabefeld löschen

Wird auf dem Numpad welches sich auf der Devpage befindet der Knopf „C“ gedrückt, so wird die Funktion „erase“ ausgeführt (siehe Abbildung 191) . Diese prüft welche Inputbox aktiviert ist und ob sich Zeichen in ihr befinden. Trifft dies zu wird das letzte Zeichen der Inputbox gelöscht.

9.6.5.3. Fokus

```
# ==================================================

    def focus(self, Input):
        self.prevent_range()

        if Input != "Input1_enabled":
            self.Input1_enabled = False
            with open("temp/Input1", "w") as file:
                file.write(self.Input1_text)
            file.close()
        else:
            self.Input1_enabled = True
```

Abbildung 192. Zeiteingabefeld Fokus

Wenn eine der Inputboxen angeklickt wird, wird die Funktion „focus“ ausgeführt (siehe Abbildung 192). Diese Funktion setzt die Box auf aktiviert. Wird eine andere Box

angeklickt oder der Enter Knopf gedrückt so verliert diese Box den focus. Die Funktion erfasst dies und deaktiviert die Box, weiters wird der Inhalt der Inputox in eine Datei gespeichert.

9.6.5.4. Verhindern eines Bereichsfehlers

```
# ====================================

    def prevent_range(self):
        if self.Input1_text == "":
            self.Input1_text = "0"
```

Abbildung 193: Zeiteingabefeld Verhindern eines Bereichsfehlers

Die Funktion „prevent_range“ wird aufgerufen durch die Funktionen „enter“ und „focus“. Sie überprüft ob sich in der Inputbox Character befinden. Wenn nicht wird eine 0 eingefügt (siehe Abbildung 193). Dies wird gebraucht damit beim Auswerten kein TypeError entsteht.

9.6.5.5. Nullen löschen

```
# ==========================================================================

    def erase_zeros(self):
        while self.Input1_text[0] == "0" and len(self.Input1_text) > 1:
            self.Input1_text = self.Input1_text[1:]
```

Abbildung 194: Zeiteingabefeld Nullen löschen

Die Funktion „erase_zeros“ wird durch das Drücken des Enter Knopfes ausgeführt. Die Funktion überprüft ob sich vorangestellte Nullen vor der ersten Signifikaten Stelle befinden (siehe Abbildung 194). Ist dies der Fall werden diese entfernt.

9.6.5.6. Enter

```
# ==================================================================

def enter(self):

    self.prevent_range()
    self.erase_zeros()
    self.Input_save()

    self.Input1_enabled = False
    self.Input2_enabled = False
    self.Input3_enabled = False
    self.Input4_enabled = False
    self.Input5_enabled = False
    self.Input6_enabled = False

    if (not self.Input1_enabled and not self.Input2_enabled
        and not self.Input3_enabled and not self.Input4_enabled
        and not self.Input5_enabled and not self.Input6_enabled):
        self.pad_enabled = False
```

Abbildung 195: Zeiteingabefeld Enter

Wenn der Knopf Enter auf dem Numpad gedrückt wird, wird die Funktion „enter“ aufgerufen (siehe Abbildung 195). Sie optimiert die Werte in den Inputboxen und deaktiviert diese. Weiters wird das Numpad deaktiviert.

9.6.5.7. Speichern

```
# ==============================================

def Input_save(self):
    with open("temp/Input1", "w") as file:
        file.write(self.Input1_text)
    file.close()
```

Abbildung 196: Zeiteingabefeld speichern

Die Funktion „Input_save“ wird durch die Funktionen „enter“ und „Input_buttons“ aufgerufen und speichert die Werte der Inputboxen in Dateien ab (siehe Abbildung 196).

9.6.5.8. Plus / Minus 1

```
# ==================================================================

    def Input_buttons(self, direction, Input):
        if direction == "minus":
            if Input == "Input1" and int(self.Input1_text) > 0:
                self.Input1_text = str(int(self.Input1_text) - 1)
        elif direction == "plus":
            if Input == "Input1" and int(self.Input1_text) < 99999:
                self.Input1_text = str(int(self.Input1_text) + 1)
```

Abbildung 197: Zeiteingabefeld Plus/Minus 1

Wird einer der Knöpfe neben der Inputbox gedrückt so wird die Funktion „Input_buttons“ aufgerufen. Diese ermittelt welcher der beiden Knöpfe gedrückt wurde und addiert oder subtrahiert anhand dessen 1, sofern der Maximal- oder Minimalwert nicht erreicht sind (siehe Abbildung 197).

10. Fertigung

Die Walzen für diese Münzprägemaschine werden in der Schweiz gefertigt. Die Schweizerdelegation, hat sich beim Meeting, in Norwegen stattgefunden hat, dazu bereit erklärt, die Walzen in ihrer Firma zu fertigen. Diese beiden Walzen sind jedoch die einzigen Teile, welche zur Gänze auswärts und nicht von dem Diplomanden an der HTBLuVA St. Pölten gefertigt wurden.

Da die Schweizer eine 5-Achsfräse mit CAM-System besitzen, können sie die Walzen einfach als fertige STEP-Datei auf die Fräse hochladen und müssen die Motive nicht eigenhändig programmieren. Damit dies reibungslos funktioniert, müssen auf allen Motiven noch Radien hinzugefügt werden, da die Fräse keine 90°-Ecken fertigen kann. Das fertige Motiv für die Bundeslehr- und Versuchsanstalt St. Pölten sieht dann Folgendermaßen aus (siehe Abbildung 198).

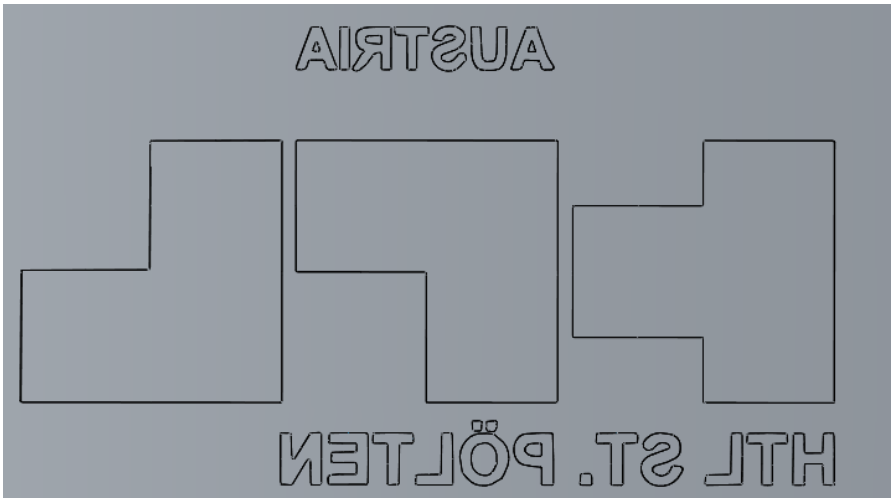


Abbildung 198: Logo mit leichten Radien

Die hinzugefügten Radien sind nur sehr klein ausgeführt und sind somit auch nicht gut sichtbar. In der Abbildung 199 sind die Radien nochmals in einer besseren Ansicht dargestellt.

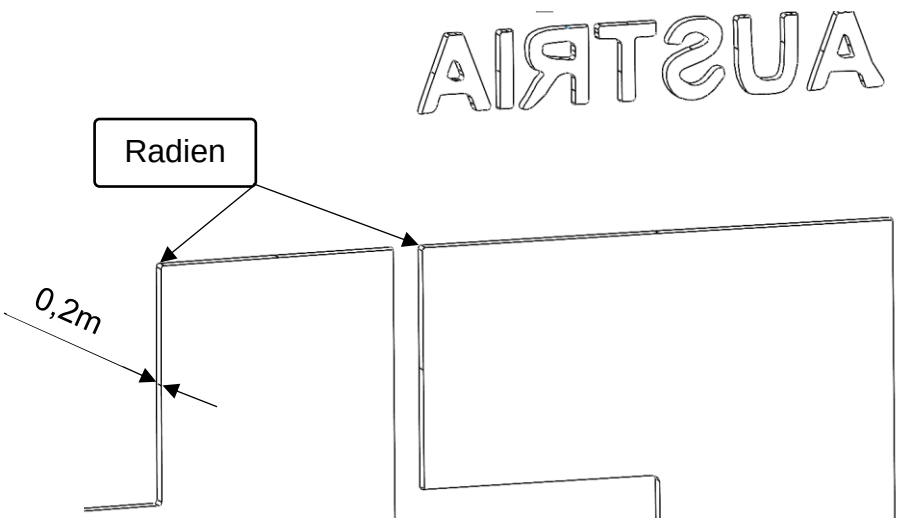


Abbildung 199: Radien sichtbar

Weiters wird beschlossen, einen Gravierfräser an Stelle eines Fingerfräsers mit einem Durchmesser von 0,1mm zu verwenden, da der Chef der Schweizerfirma der Meinung ist, dass die Fräser sofort abbrechen würden, wenn man damit das Werkstück berührt. Da der Gravierfräser einen Spitzenwinkel von 60° besitzt, wären die Motive um einiges Breiter geworden, wenn dieselbe Tiefe der Motive beibehalten werden sollte. Um dies zu verhindern, wird die Tiefe der Motive auf 0,2mm anstatt 0,45mm gesenkt (siehe Abbildung 199).

Die am häufigsten angewandten Verfahren zur Herstellung der Teile sind Drehen und Bohren (siehe Abbildung 200), sowie das Fräsen (siehe Abbildung 201).

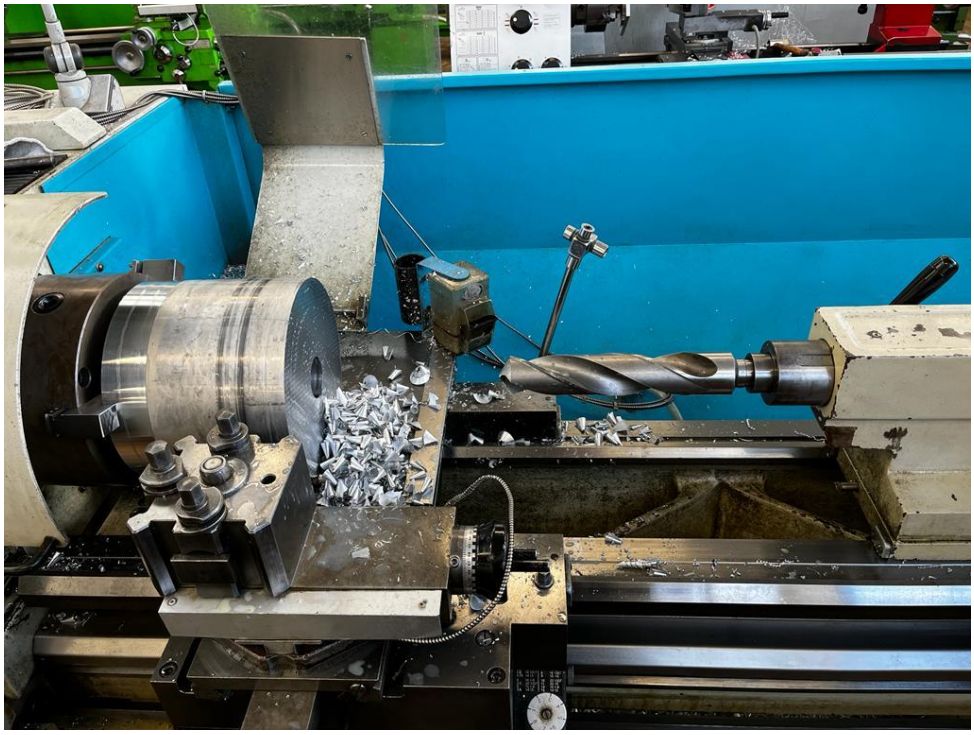


Abbildung 200: Drehen und Bohren Achse

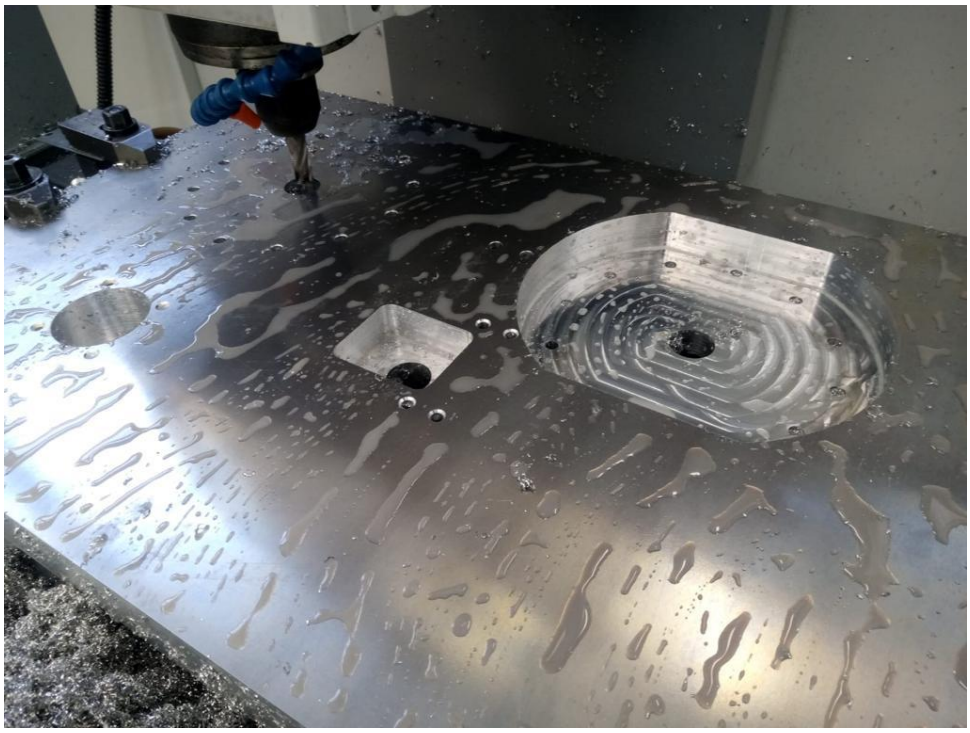


Abbildung 201: Fräsen große Platte

Ein für die Diplomanden noch eher unvertrautes Werkzeug ist ein sogenannter Teilapparat (siehe Abbildung 202), mit welchem immer eine bestimmte Anzahl an Schritten, meistens nur wenige Grad, weitergedreht werden kann.

Abbildung 202: Teilapparat

Funktionsmäßig ist der Teilapparat so aufgebaut, dass auf der in Abbildung 202 sichtbaren Lochscheibe eine gewisse Teilung aufgetragen ist. Diese ist abhängig von der Zähnezahl des Zahnrades. In diesem Fall wird ein Zahnrad mit 19 Zähnen gefertigt. Da keine Scheibe mit einer Teilung von 19 Zähnen vorhanden ist, wird eine Scheibe mit einer Teilung von 57 Zähnen verwendet, da diese vorhanden war und 57 ein Vielfaches von 19 ist. Dies muss beim Weiterdrehen des Apparats berücksichtigt werden. Es muss immer 2 Umdrehungen und 7 Löcher weitergedreht werden.

Dieser Teilappart wird in Kombination mit einem Modulfräser für die Fertigung der Ritzelwelle verwendet. Der Modul des verwendeten Zahnradpaars ist 3, deshalb wird auch ein dreier Modulfräser für die Fertigung verwendet (siehe Abbildung 203). Der

Modulfräser besitzt genau die Form einer Zahnlücke und erzeugt so mit jedem Fräsvorgang zwei Zahnflanken (siehe Abbildung 204).

Abbildung 203: Modulfräser

Abbildung 204: Kontur Modulfräser

Für die Fertigung muss der Fräser genau auf der Welle, welche mit einer Verzahnung versehen werden soll, ausgemittelt werden. Von diesem Mittel ausgehend können in diesem Fall immer genau 6,5mm zugestellt werden und anschließend in Längsrichtung durchgefräst werden. Der Fräsvorgang wurde auf zwei Schnitte aufgeteilt, einmal mit 5,5mm, um möglichst viel Material weg zu nehmen und einmal 1mm, um eine glatte Oberfläche zu bekommen. Die Ritzelwelle kann so Zahnlücke für Zahnlücke gefräst werden, wie Abbildung 205 bis Abbildung 207 zeigen.

Abbildung 205: 1.Zahn Ritzelwelle

Abbildung 206: Zwischenstand Ritzelwelle

Abbildung 207: Fertige Ritzelwelle

Bei der Fertigung der doch recht großen Teile sind einige Probleme aufgetreten, da die, an der HTBLuVA St. Pölten vorhandenen, Messmittel und Maschinen nicht für

Teile dieser Größe ausgelegt sind. Durch Kontakte zu befreundeten Firmen konnten die benötigten Messmittel bereitgestellt werden. Weiters mussten zum Einspannen der Achsen die Spannbacken getauscht werden, da auf dieser Maschine (Colchester) im Normalfall nicht so große Teile bearbeitet werden.

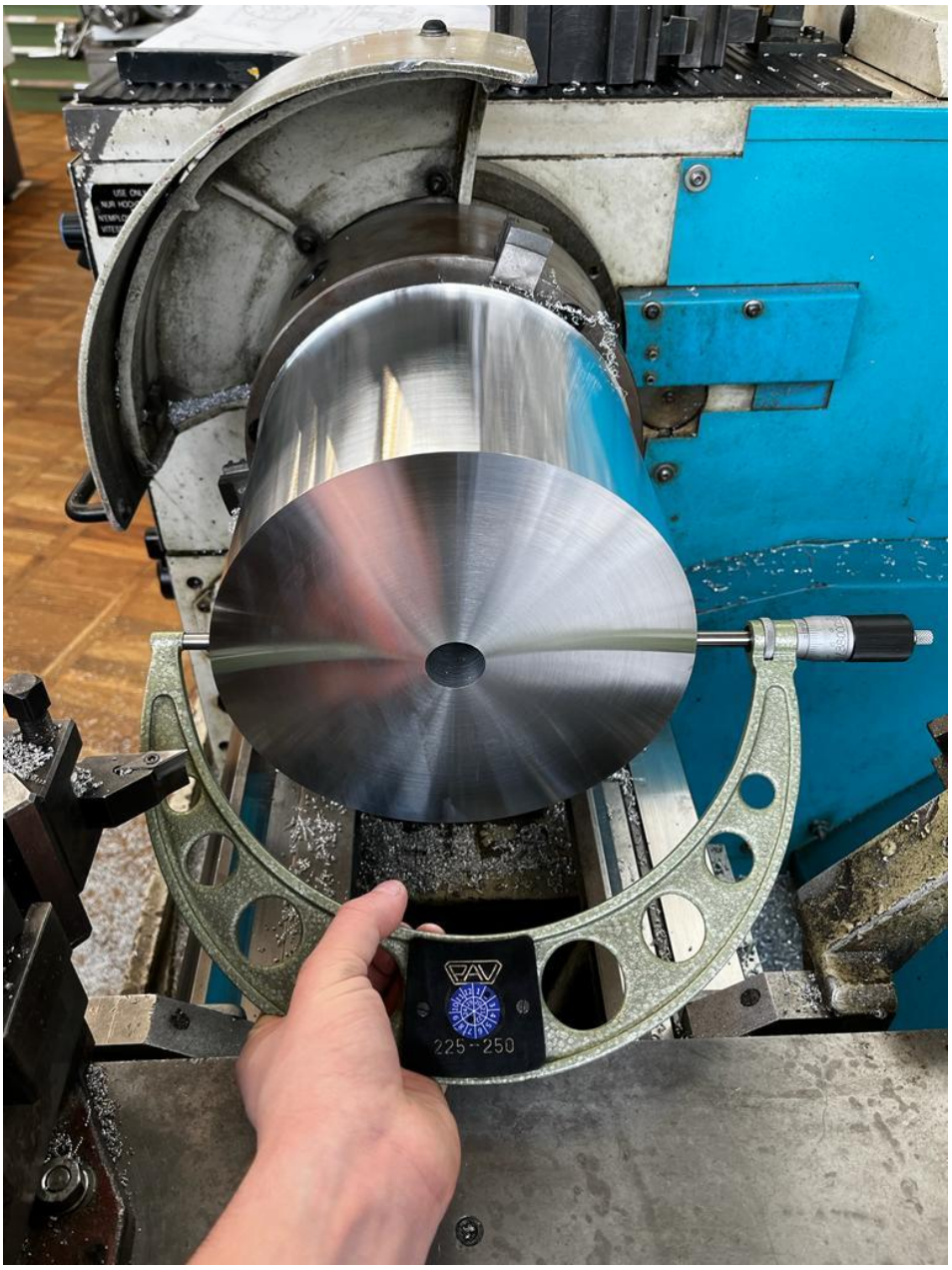


Abbildung 208: Mikrometer, Achse

Für die Fertigung der Beiden Achsen 3 und 4 müssen jeweils zwei Passungen gedreht werden, die durch ausreichend große Mikrometerschrauben (siehe Abbildung 208) vermessen werden.
Das Bohren der Lochkreise erfolgte auf den HAAS-Fräsmaschinen, jedoch können nur die Löcher gebohrt werden und nicht die Gewinde geschnitten werden, da kein funktionstüchtiges Ausgleichsfutter für die Gewindebohrer vorhanden ist. Die Gewinde wurden deshalb nachträglich per Hand geschnitten.

Um das Ausdrehen der Zahnräder (siehe Abbildung 209), welche zu Zahnkränzen werden sollen, zu erleichtern, werden diese zuerst mit dem Wasserstrahlschneider (siehe Abbildung 210) auf ein passenderes Maß gebracht. Anschließend werden die vorgeschnittenen Zahnkränze mittels einer Planscheibe, mit einem 4-Backenfutter, auf eine dafür geeignete Maschine gespannt.

Abbildung 209: Ausdrehen Zahnkranz

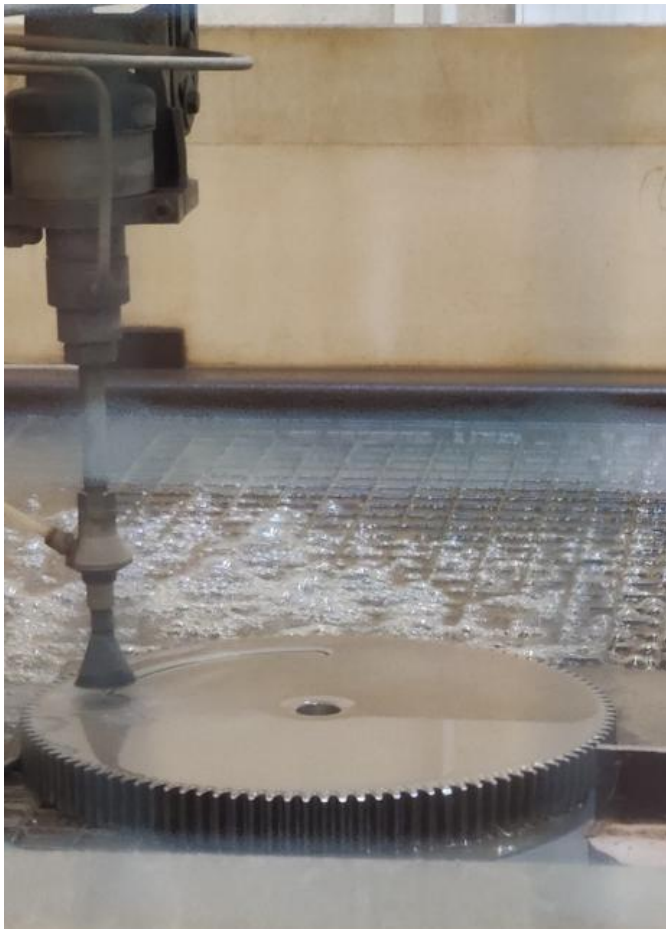


Abbildung 210: Wasserstrahlschneiden

Hier muss ebenfalls auf möglichst genauen Rundlauf geachtet werden, da die beiden Zahnkränze mittels einem Zentriersitz auf den Walzen zentriert werden sollen. Dies wird ebenfalls wie bei den Teilen zuvor mittels einer Messuhr realisiert. Da die Oberfläche auf Grund der Zähne nicht glatt ist, ist dies eine größere Herausforderung als bei den beiden Achsen.

Auch das Fertigen der großen Platte stellt sich als Herausforderung dar, da der Verfahrweg der Fräse nicht ausreicht, um die ganze Platte in einer Aufspannung zu fräsen. Deshalb müssen zuerst die beiden Sitze für die Achsen, sowie die dazugehörigen Lochkreise in der ersten Aufspannung in die Platte gefräst werden. Anschließend wird die Platte umgespannt und die restliche Bearbeitung dieser Plattenseite durchgeführt.

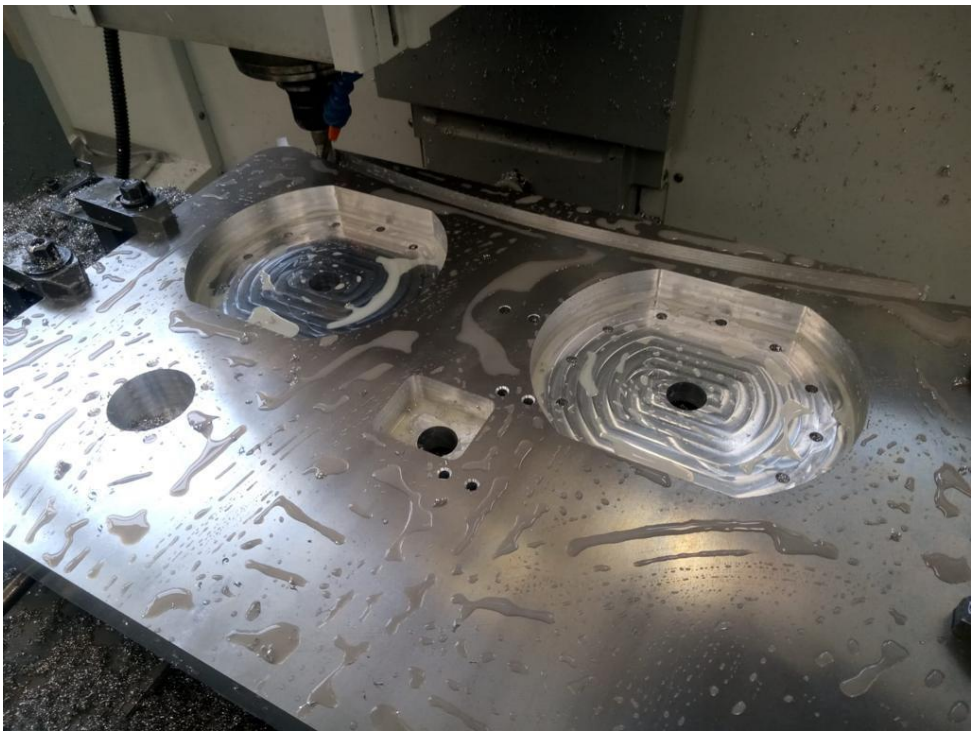


Abbildung 211: Große Platte

Die kleine Platte hingegen kann in einer Aufspannung gefräst werden und bereitetet auch keine Probleme. Mittels eines kleinen CNC-Programms, welches direkt auf der Fräse geschrieben werden kann, können die Lochkreise sowie die Ausnehmungen für die Walzen als auch die Außenkonturen der Platten bearbeitet werden.

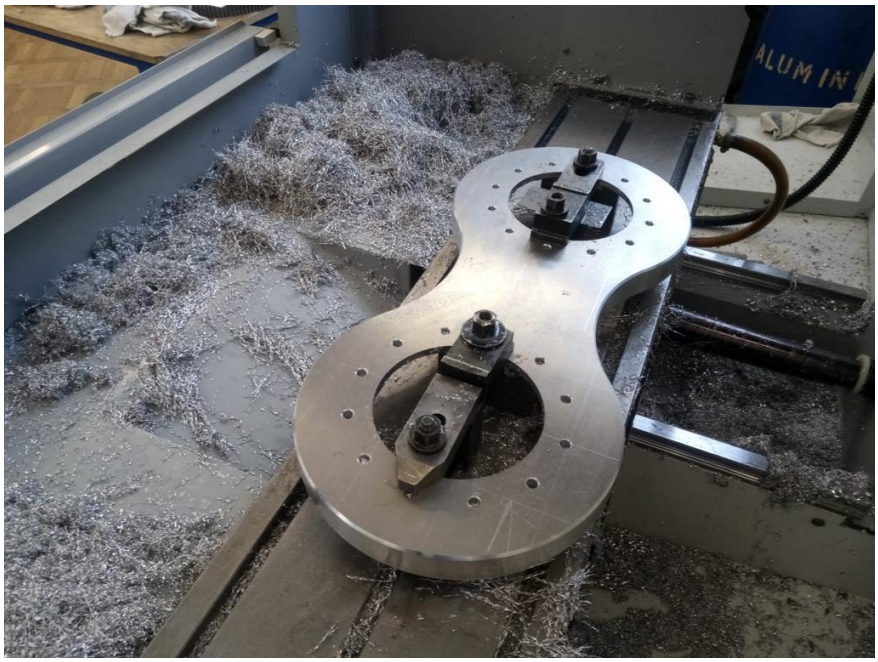


Abbildung 212: Kleine Platte

Es wird jedoch nicht nur auf den Maschinen der HTBLuVA St. Pölten produziert, sondern auch zuhause bei den Diplomanden. Um die Funktionsfähigkeit des entworfenen Gehäuses, für den Motor und das dazugehörige Getriebe, zu überprüfen, druckt Diplomand Grafeneder zwei Prototypen mit jeweils kleinen Veränderungen (siehe Abbildung 213).

Abbildung 213: 3D-Druck Prototyp

Nach der erfolgreichen Funktionsprüfung des Gehäuses wird dieses von Diplomanden Schüler ebenfalls mit seinem privaten 3D-Drucker gedruckt. Folglich werden alle weiteren Teile von ihm gedruckt. Das daraus resultierende Endprodukt des Gehäuses und des dazugehörigen Deckels sieht folgendermaßen aus:

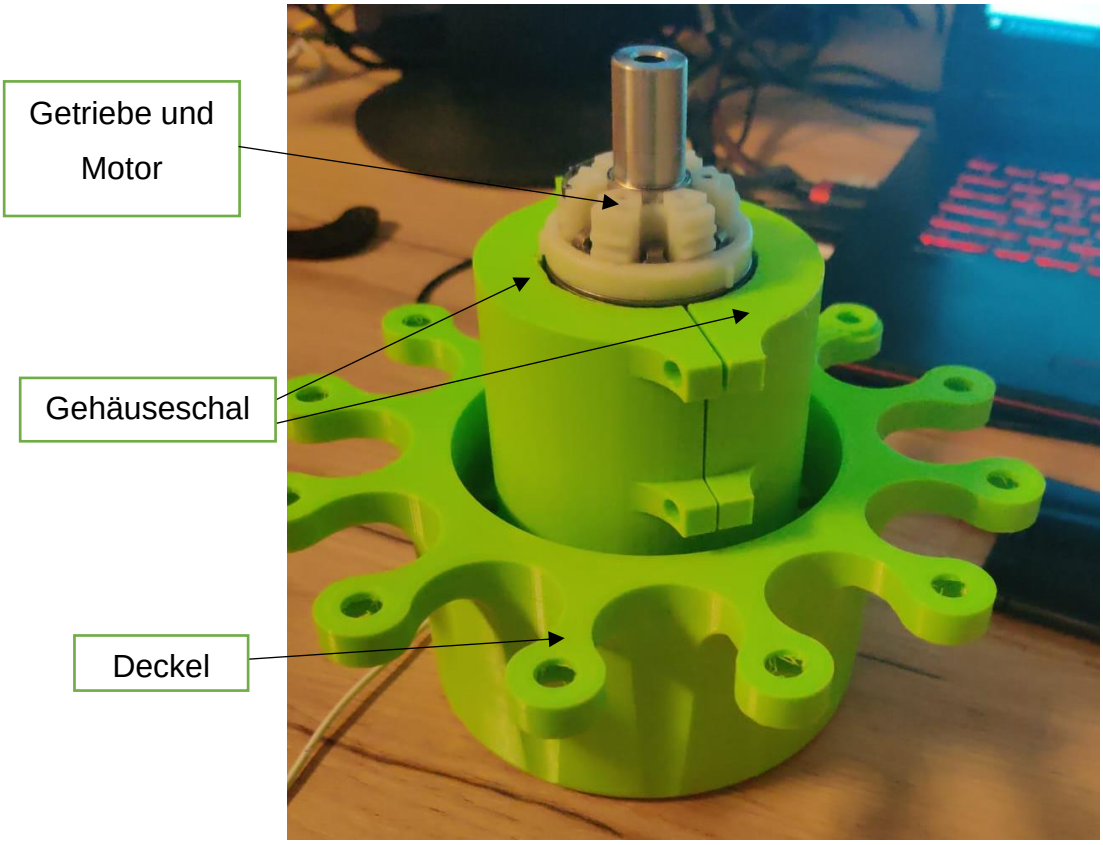


Abbildung 214: 3D-Druck, Motorgehäuse und Deckel

Abbildung 1: Projektteam PPM4EU (https://ppm4eu.cnc-network.eu/activities/meetings/meeting-no/58-third-meeting-in-trondheim) 14
Abbildung 2: Organigramm 16
Abbildung 3: Druckprüfgerät 19
Abbildung 4: Hülse, dicker Ring 20
Abbildung 5: Vorbereitung Druckversuch 21
Abbildung 6: Diagram Druckversuch 22
Abbildung 7: Blechwalze 23
Abbildung 8: Gewalzte Münzen 23
Abbildung 9: Zwischenstandmessung Münze 24
Abbildung 10: Walze_klein_schmal 25
Abbildung 11: Walze_klein_breit 26
Abbildung 12: Walzen_groß_schmal 27
Abbildung 13: Beispielmünze 28
Abbildung 14: Schemaskizze_Nase 29
Abbildung 15: Skizze zur Berechnung des Walzenradius 30
Abbildung 16: Gleichungssystem zur Walzenradiusberechnung 31
Abbildung 17: Greifwinkel 31
Abbildung 18: Mindestreibwert 32
Abbildung 19: Verformwirkungsgrad 33
Abbildung 20: Fließkurven 34
Abbildung 21: Formänderungsfestigkeit 34
Abbildung 22: mittlere Formänderungsfestigkeit 34
Abbildung 23: mittlerer Umformwiderstand 35
Abbildung 24: gedrückte Länge, Fläche im Eingriff 35
Abbildung 25: benötigte Umformkraft 35
Abbildung 26: Hebelarmbeiwert 36
Abbildung 27: erforderliches Walzmoment 36
Abbildung 28: Walzmoment inklusive Sicherheit 37
Abbildung 29: Schematische Konstruktion des Getriebes 38
Abbildung 30: Aufbau mit stillstehender Achse 40
Abbildung 31: Getriebemotor v. Watt Drive (http://www.wattdrive.com/wg20/) 41
Abbildung 32: Beispiel für Motorauswahl (https://cat4cad.wattdrive.com/cat4cad1/) 42
Abbildung 33: Grobübersicht d. Baugruppenstruktur (Datei: Baugruppenstruktur) 44
Abbildung 34: Neuen Teil einfügen 45
Abbildung 35: Aufbau der Zeichnungsnummern 45
Abbildung 36:Filtern nach Baugruppen 46
Abbildung 37: Fertigungszustand 46
Abbildung 38: Verwaltung 47
Abbildung 39: Grobansicht Steuerskizze 48
Abbildung 40: Achsen, Ebenen und Skizzen 49
Abbildung 41:Hauptzusammenbau mit Unterbaugruppen 50
Abbildung 42: Berechnung der erforderlichen Gesamtübersetzung (Datei: „Berechnung_Getriebe). 51
Abbildung 43: Überblick Getriebestufen 1 & 2 52

Abbildung 44: Berechnung Achsabstand 1 & 2 ... 52
Abbildung 45: Überblick Getriebestufe 3 ... 53
Abbildung 46: Berechnung Achsabstand 3 ... 53
Abbildung 47: Antriebsstrang (geschnitten) ... 54
Abbildung 48: Unterbaugruppe-Antrieb_ZSB ... 55
Abbildung 49: Innenleben Akkuschrauber ... 56
Abbildung 50: Motor und Getriebe (Bosch) ... 56
Abbildung 51: Gehäusehälfte ... 57
Abbildung 52: Positionsschraube Motor ... 58
Abbildung 53: Ballige Verzahnung ... 59
Abbildung 54: Erforderliches Antriebsmoment ... 59
Abbildung 55: Auswahl der Kupplung (Online: https://www.norelem.at) ... 60
Abbildung 56: Antrieb_ZSB - Zusammenbau ... 61
Abbildung 57: Unterbaugruppe Welle1_ZSB ... 62
Abbildung 58: Welle1_ZSB (eingesetzt) ... 63
Abbildung 59: Auflagerreaktionen Welle 1 ... 64
Abbildung 60: Lagerkräfte ... 64
Abbildung 61: Berechnung der auftretenden Spannungen (Datei: Berechnung_Getriebe) ... 65
Abbildung 62: Berechnung Lagerauswahl ... 67
Abbildung 63: SKF-Shop (Online: https://www.skf.com/at) ... 67
Abbildung 64: Berechnung Passfeder ... 68
Abbildung 65: Erste Übersetzungsstufe ... 69
Abbildung 66: Unterbaugruppe Welle2_ZSB ... 69
Abbildung 67: Welle2_ZSB (verbauter Zustand) ... 70
Abbildung 68: Auflagerreaktionen Welle2 ... 72
Abbildung 69: Unterbaugruppe "Welle3_ZSB" ... 73
Abbildung 70: Achse3_ZSB (verbauter Zustand) ... 74
Abbildung 71: Achse3 ... 75
Abbildung 72: Flachstelle Achse 3 ... 76
Abbildung 73: Achse3 - Lagerkräfte ... 77
Abbildung 74: Achse3 - Lagerberechnung ... 77
Abbildung 75: Achse3 ... 78
Abbildung 76: Berechnung Walzendurchmesser ... 79
Abbildung 77: Berechnung Schraubenverbindung ... 80
Abbildung 78: HTL St. Pölten Logo in SolidWorks ... 81
Abbildung 79: Bemaßung HTL Logo ... 82
Abbildung 80: Skizze HTL Logo Walzen ... 83
Abbildung 81: Wrap-Feature ... 84
Abbildung 82: Inkscape ... 84
Abbildung 83: DXF-Format ... 85
Abbildung 84: Luxemburg Schloss Kontur ... 85
Abbildung 85: Luxemburg Schloss Fenster ... 86
Abbildung 86: Luxemburg Schrift ... 86
Abbildung 87: Walze 1 mit Motiven ... 87

Abbildung 88: Nachziehen der Motive des PPM4EU Logos ... 87
Abbildung 89: PPM4EU Logo auf der Walze 2 ... 88
Abbildung 90: Zahnkranz (https://www.norelem.at) ... 89
Abbildung 91: Referenzieren der Bohrung ... 90
Abbildung 92: Achse3_ZSB & und Achse4_ZSB ... 91
Abbildung 93: Achse4 ... 92
Abbildung 94: Zahnüberschneidung ... 93
Abbildung 95: Verdrehtes Bohrbild ... 93
Abbildung 96: Hinterer Lagerblock ... 94
Abbildung 97: Lagerblock vorne ... 95
Abbildung 98: Vorderer Lagerblock – Funktion 1 ... 96
Abbildung 99: Lagerblock vorne - Funktion 2 ... 96
Abbildung 100: Erstes und finales Design der Lagerblöcke ... 97
Abbildung 101: Montage Schritt 1 ... 98
Abbildung 102: Montage Schritt 2 ... 99
Abbildung 103: Montage Schritt 3 ... 100
Abbildung 104: Montage Schritt 4 ... 101
Abbildung 105: Montage Schritt 5 ... 102
Abbildung 106: Gehäuse ... 103
Abbildung 107: Grundrahmen ... 104
Abbildung 108: Bsp. Zeichnung ... 105
Abbildung 109: Münzaufzug ... 106
Abbildung 110: Münzzuführung ... 107
Abbildung 111: Münzeinwurf ... 108
Abbildung 112: Umlenkschiene ... 108
Abbildung 113: Zuführschiene ... 109
Abbildung 114: Zuführung ... 110
Abbildung 115: Münzrutsche ... 110
Abbildung 116 Schwenkarm ... 112
Abbildung 117 Leitschiene ... 113
Abbildung 118 Abführen der Münze ... 114
Abbildung 119: SPS (https://de.wikipedia.org/wiki/Logo_(SPS)#) ... 115
Abbildung 120: Arduino (https://www.reichelt.at/at/de/arduino-uno-rev-3-smd-variante-atmega328-usb-arduino-uno-p119045.html) ... 116
Abbildung 121: Raspberry Pi (https://upload.wikimedia.org/wikipedia/commons/f/f1/Raspberry_Pi_4_Model_B_-_Side.jpg) ... 117
Abbildung 122: Raspberry Pi + Arduino (https://elektro.turanis.de/html/prj176/index.html) ... 118
Abbildung 123: Münzen https://www.muenzen.eu/euro-umlaufmuenzen.htm ... 120
Abbildung 124: mechanische Münzerkennung ... 121
Abbildung 125: elektronische Münzerkennung https://web.mit.edu/2.972/www/reports/coin_tester/coin_tester.htm ... 122
Abbildung 126: 117 CH-926 https://ae01.alicdn.com/kf/Hf91e64b446834eb990b7a008d09ab770t/2020-Multi-M-nzpr-fer-

Elektronische-Rolle-Unten-M-nzpr-fer-Selector-Mechanismus-Automaten-Arcade-Spiel.jpg_Q90.jpg_.webp 123
Abbildung 127: Drehgeber https://www.waycon.de/produkte/encoder-winkelgeber/messprinzip-encoder-drehgeber/ 124
Abbildung 128: Konzept Inkrementalgeber 125
Abbildung 129: Konzept optischer Sensor 126
Abbildung 130: Zahnflankenerkennung 127
Abbildung 131: Schraubenerkennung 128
Abbildung 132 Schraubenerkennung mit Referenz 129
Abbildung 133 Schrittmotortreiber https://electrobes.com/product/drv8825-stepper-motor-driver-with-heat-sink/ 130
Abbildung 134: Motortreiberplatine (aus Datenblatt md30c 1,4-usersmanual) 132
Abbildung 135: Optokoppler 133
Abbildung 136: Versuchsschaltung 134
Abbildung 137: Schaltung Induktiver Sensor 135
Abbildung 138: Tinkercad Schaltplan 136
Abbildung 139 Printklemmen 137
Abbildung 140 Zeichnung Platine 138
Abbildung 141 Schaltplatine 139
Abbildung 142 Komponenten Platine 139
Abbildung 143: Arduino Bibliotheken 141
Abbildung 144: Arduino Pin festlegen 141
Abbildung 145: Arduino Variablen 142
Abbildung 146: Arduino Ein- und Ausgänge 143
Abbildung 147: RFID Initialisieren 144
Abbildung 148: Stepper Referenzieren 144
Abbildung 149: Position Walzen lesen 145
Abbildung 150: Walzen Referenz lesen 146
Abbildung 151: Endlage Stepper lesen 146
Abbildung 152: Münzprüfeinheit lesen 147
Abbildung 153: RFID lesen 148
Abbildung 154: serielle Schnittstelle lesen 149
Abbildung 155: Arduino Daten auswerten 150
Abbildung 156: Schritt 10 151
Abbildung 157: Schritt 20 151
Abbildung 158: Schritt 30 152
Abbildung 159. Schritt 40 152
Abbildung 160: Schritt 50 153
Abbildung 161: Schritt 60 153
Abbildung 162: Schritt 70 154
Abbildung 163. Layouts (https://subscription.packtpub.com/book/application-development/9781785286926/1/ch01lvl1sec11/embedding-layouts) 157
Abbildung 164: Label (https://kivy.org/doc/stable/api-kivy.uix.label.html) 159
Abbildung 165: Button (https://kivy.org/doc/stable/api-kivy.uix.button.html) 159

Abbildung 166: Toggle Button (https://kivy.org/doc/stable/api-kivy.uix.togglebutton.html) 160
Abbildung 167: Switch (https://kivy.org/doc/stable/api-kivy.uix.switch.html) 160
Abbildung 168: Slider (https://kivy.org/doc/stable-1.11.1/api-kivy.uix.slider.html) 160
Abbildung 169: Raspberry Pi Builder und Windowmanager 161
Abbildung 170: Raspberry Pi Bibliotheken 161
Abbildung 171: Raspberry Pi serielle Kommunikation 162
Abbildung 172: Raspberry Pi globale Variablen 162
Abbildung 173: Raspberry Pi Fenster erstellen 163
Abbildung 174: Raspberry Pi Initialisierung von Objekten 163
Abbildung 175:Raspberry Pi serielle Schnittstelle lesen 164
Abbildung 176: Wartebildschirm 165
Abbildung 177: Münze eingeworfen 166
Abbildung 178: MainPage 166
Abbildung 179: Designauswahl 167
Abbildung 180: Popup 168
Abbildung 181: Datenübertragung 169
Abbildung 182: Prozessbildschirm 170
Abbildung 183: Maschine fertig 171
Abbildung 184: DevPage Page 1 171
Abbildung 185:DevPage Page 2 172
Abbildung 186: Vollbild 173
Abbildung 187: Animationen 174
Abbildung 188: Geschwindigkeiten und Nachlaufzeit 175
Abbildung 189: Manuelle Motorsteuerung 176
Abbildung 190: Zeiteingabefeld schreiben 176
Abbildung 191: Zeiteingabefeld löschen 177
Abbildung 192. Zeiteingabefeld Fokus 177
Abbildung 193: Zeiteingabefeld Verhindern eines Bereichsfehlers 178
Abbildung 194: Zeiteingabefeld Nullen löschen 178
Abbildung 195: Zeiteingabefeld Enter 179
Abbildung 196: Zeiteingabefeld speichern 179
Abbildung 197: Zeiteingabefeld Plus/Minus 1 180
Abbildung 198: Logo mit leichten Radien 181
Abbildung 199: Radien sichtbar 182
Abbildung 200: Drehen und Bohren Achse 183
Abbildung 201: Fräsen große Platte 183
Abbildung 202: Teilapparat 184
Abbildung 203: Modulfräser 185
Abbildung 204: Kontur Modulfräser 185
Abbildung 205: 1.Zahn Ritzelwelle 186
Abbildung 206: Zwischenstand Ritzelwelle 186
Abbildung 207: Fertige Ritzelwelle 186
Abbildung 208: Mikrometer, Achse 187
Abbildung 209: Ausdrehen Zahnkranz 189

Abbildung 210: Wasserstrahlschneiden 189
Abbildung 211: Große Platte 190
Abbildung 212: Kleine Platte 191
Abbildung 213: 3D-Druck Prototyp 192
Abbildung 214: 3D-Druck, Motorgehäuse und Deckel 193

Literaturverzeichnis

Relative Layout. (kein Datum). Von Kivy: https://kivy.org/doc/stable/api-kivy.uix.relativelayout.html abgerufen

Anchor Layout. (kein Datum). Von Kivy : https://kivy.org/doc/stable/api-kivy.uix.anchorlayout.html abgerufen

Arduino. (19. Jänner 2022). Von Wikipedia: https://de.wikipedia.org/wiki/Arduino_(Plattform) abgerufen

Arduino vs Raspberry Pi. (6. März 2018). Von IONOS: https://www.ionos.at/digitalguide/server/knowhow/arduino-vs-raspberry-pi/ abgerufen

Box Layout. (kein Datum). Von Kivy: https://kivy.org/doc/stable/api-kivy.uix.boxlayout.html abgerufen

Button. (kein Datum). Von Kivy: https://kivy.org/doc/stable/api-kivy.uix.button.html abgerufen

C++ Definition. (Mai 2016). Von ComputerWeekly: https://www.computerweekly.com/de/definition/C abgerufen

Eckert, M. (29. Jänner 2015). *Was ist der Unterschied zischen Bitrate und Baudrate*. Von ComputerWeekly: https://www.computerweekly.com/de/antwort/Was-ist-der-Unterschied-zwischen-Bitrate-und-Baudrate#:~:text=Die%20Baudrate%20ist%20per%20Definition,manchmal%20auch%20als%20Symbolrate%20bezeichnet. abgerufen

Float Layout. (kein Datum). Von Kivy: https://kivy.org/doc/stable/api-kivy.uix.floatlayout.html abgerufen

Function. (kein Datum). Von Programiz: https://www.programiz.com/python-programming/function abgerufen

Functions. (August 2021). Von Microsoft: https://docs.microsoft.com/en-us/cpp/cpp/functions-cpp?view=msvc-170 abgerufen

Grid Layout. (kein Datum). Von Kivy: https://kivy.org/doc/stable/api-kivy.uix.gridlayout.html abgerufen

Gruss, P. D.-I. (10 2010). *Berechnungsgrundlagen Stirnradverzahnung.* Von Hochschule Anhalt: https://www.hs-

anhalt.de/fileadmin/Dateien/FB6/personen/voigt_st/Lehrunterlagen/02_MA/Berechnungsgrundlagen_Stirnradverzahnungen_2010-10.pdf abgerufen

Herbert, W., Dieter, j., Joachm, V., & Christian, S. (2017). *Roloff / Matek Maschinenelemente Auflage 23.* Wießbaden: Springer vieweg.

Hu, D. (kein Datum). *Web MIT*. Von Coin tester: https://web.mit.edu/2.972/www/reports/coin_tester/coin_tester.htm abgerufen

J. Dietrich, H. T. (2. September 2013). *Media/blfa_files.* Von Ingenieur-Buch: https://www.ingenieur-buch.de/media/blfa_files/9783658019952-Leseprobe.pdf abgerufen

Juschkat, K. (8. September 2019). *Was ist eine SPS Definition Grundlagen und Funktion*. Von elektrotechnik Vogel: https://www.elektrotechnik.vogel.de/was-ist-eine-sps-definition-grundlagen-und-funktion-a-773404/ abgerufen

Label. (kein Datum). Von Kivy: https://kivy.org/doc/stable/api-kivy.uix.label.html abgerufen

Overhagen, D.-I. (2. November 2018). *Receive/Duepublico_mods.* Von DuEPublico 2: https://duepublico2.uni-due.de/receive/duepublico_mods_00047540 abgerufen

Python Inheritance. (kein Datum). Von w3schools: Python Inheritance- https://www.w3schools.com/python/python_inheritance.asp abgerufen

Python Klassen. (22. April 2021). Von Sivakids: https://www.sivakids.de/python-klassen/ abgerufen

PYTHON-GUI - PYQT VS TKINTER. (kein Datum). Von Acervolima: https://de.acervolima.com/python-gui-pyqt-vs-tkinter/ abgerufen

Raspberry Pi. (2. März 2022). Von Wikipedia: https://de.wikipedia.org/wiki/Raspberry_Pi abgerufen

Slider. (kein Datum). Von Kivy: Slider- https://kivy.org/doc/stable/api-kivy.uix.slider.html abgerufen

Stack Layout. (kein Datum). Von Kivy : https://kivy.org/doc/stable/api-kivy.uix.stacklayout.html abgerufen

Switch. (kein Datum). Von Kivy: Switch- https://kivy.org/doc/stable/api-kivy.uix.switch.html abgerufen

Switch Statement. (17. August 2021). Von Microsoft: https://docs.microsoft.com/en-us/cpp/cpp/switch-statement-cpp?view=msvc-170 abgerufen

Toggle Button. (kein Datum). Von Kivy: https://kivy.org/doc/stable/api-kivy.uix.togglebutton.html abgerufen

Tyson, J. (kein Datum). *Serial Port.* Von Howstuffworks: https://computer.howstuffworks.com/serial-port.htm abgerufen

UTF-8. (22. Dezember 2021). Von Wikipedia: https://de.wikipedia.org/wiki/UTF-8 abgerufen

Waycon. (kein Datum). Von Messprinzip Encoder Drehgeber: https://www.waycon.de/produkte/encoder-winkelgeber/messprinzip-encoder-drehgeber/ abgerufen