Höhere Technische Bundeslehranstalt Wien 3, Rennweg
Mechatronik
HTL Rennweg: Rennweg 89b
A-1030 Wien, Tel +43 1 24215-10, Fax DW 18

Diplomarbeit

Backflip Basti

ausgeführt an der
Höheren Abteilung für
Mechatronik
der Höheren Technischen Lehranstalt Wien 3 Rennweg

im Jahr 2020/21

durch
Dominik Eminger

unter der Anleitung von

Martin Sommer

Wien, März 2021

1 Kurzfassung

Das Ziel des Backflip Basti Projekts war es, die Idee der Spielemodifizierung „Salto Sammy“ für den Spiele-Klassiker „Looping Louie“ fortzuführen und das Originalspiel im Arcade-Stil zu modernifizieren und digitalisieren.

Dies sollte mit einer eigenen Motoransteuerung, neu entworfenen, 3D-gedruckten Teilen mit einer Anzeige, Tastern und einem Sensor funktionieren, der die Treffer automatisch erkennt und anzeigen lässt.

Die wichtigsten Anforderungen hierbei waren, den Motor mittels eines Motorcontrollers ansteuern zu können und damit die Geschwindigkeit zu ändern, das Einbauen eines Anzeigesystems für die derzeitigen Lebenspunkte des Spielers und das Erkennen eines Treffers, sowie die Umsetzung eines visuellen Hinweises bei einem Treffer.

Das Projekt wurde von Dominik Eminger umgesetzt, unter der Anleitung von Martin Sommer und durch die Unterstützung von Lukas Eminger als externem Betreuer.

Das Projekt wurde in 3 Hauptgruppen unterteilt, die **Mechanik und der 3D-Druck** für die Herstellung der Teile, die **Elektronik** für die Verknüpfung und Funktion der Hardware und die **Programmierung** für die Erstellung von Software, mit der die Elektronik angesteuert werden konnte. Diese konnten dann jeweils in zwei Untergruppen aufgeteilt werden, die Motoransteuerung und das Anzeigesystem. Das Anzeigesystem wurde mittels einer Lichtschranke realisiert, die einen Treffer erkennt und dies an einen Arduino Nano weiterleitet, welcher dann Leuchtdioden aufblinken lässt, die die Lebenspunkte des Spielers darstellen. Die Motorsteuerung funktioniert ebenfalls mit einem Arduino Nano. Dieser wurde in eine Elektronikbox verbaut, auf der sich Taster befinden, mit denen die Geschwindigkeit des Dreharms ausgewählt werden kann. Der Arduino steuert den Motor mittels eines Motorcontrollers dann mit der dementsprechenden Geschwindigkeit an.

Das Hauptproblem des Projekts war ein schlecht eingeschätztes Risiko, die Lieferung des 3D-Druckers verzögerte sich um 2 Monate. Zusätzlich dazu wurden manche Arbeitspakete vernachlässigt oder falsch geschätzt, was zu Zeitverlust und damit nicht erfüllten optionalen Zielen führte. Trotzdem wurden alle Hauptziele erfüllt und das Projekt damit erfolgreich abgeschlossen.

2 Abstract

The objective of the Backflip Basti project was to expand on the idea of the game modification “Salto Sammy”, which originally modified the classic game of “Looping Louie”. This should be done by modernizing and digitalizing the original game in an arcade-like style.

The project was supposed to have its own motor control, newly designed, 3D printed parts which have a display, buttons, and a sensor, which automatically detects hits and shows them to the player.

The most important goals were, that the motor was to be controlled by a motor controller to change its speed, the implementation of a display system, which shows the current health points, and of a visual cue, to notify the player that they have been hit.

The project was executed by Dominik Eminger under the instructions of Martin Sommer and with the help of Lukas Eminger as an external supervisor.

The project was split into 3 main groups, the **mechanics and 3D printing** for the manufacturing of parts, the **electronics** for the connection and functionality of the project’s hardware and the **programming** for the creation of software, which controls the electronic components. These groups were then split into 2 subgroups each, the motor control, and the display system. The display system was realized via the use of a light barrier, which could detect a hit and give this information to an Arduino nano. The Arduino could then let light emitting diodes light up and blink, which represent the player’s current health points. The motor control also uses an Arduino nano, which was placed in an electronics box. On its front side are buttons, with which the speed of the spin arm could be selected. The Arduino then sends a signal to the motor controller, which spins the motor at the selected speed.

The main problem of the project was a wrongly calculated risk, the delivery of the 3D printer was delayed by 2 months. Additionally, some work packages were neglected or misjudged, which led to a time loss and therefore not fulfilled optional goals. Nonetheless, all of the main goals were fulfilled and the project was successfully completed.

3 Ehrenwörtliche Erklärung

Ich versichere,

- dass ich die gesamte Diplomarbeit selbstständig verfasst habe,
- dass ich keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe
- und mich auch sonst keiner unerlaubten Hilfe bzw. Hilfsmittel bedient habe.

Wien, am

Dominik Eminger: _____

4 Präambel

Die Inhalte dieser Diplomarbeit entsprechen den Qualitätsnormen für „Ingenieurprojekte“ gemäß § 29 der Verordnung des Bundesministers für Unterricht und kulturelle Angelegenheiten über die Reife- und Diplomprüfung in den berufsbildenden höheren Schulen, BGBl. Nr. 847/1992, in der Fassung der Verordnungen BGBl. Nr. 269/1993, Nr. 467/1996 und BGBl. II Nr. 123/97.

Betreuender Lehrer:

Martin Sommer

Inhaltsverzeichnis

Inhalt

5 Projektidee

5.1 Ausgangssituation

Das Projekt „Backflip Basti“ basiert auf einem bereits bestehenden System, „Salto Sammy“, das den Spiele-Klassiker „Looping Louie“ modifiziert, um dem Spiel eine gewisse Unberechenbarkeit zu geben, was zu mehr Spannung und Spielspaß führt. Backflip Basti nimmt die Grundidee von Salto Sammy auf und führt diese noch weiter, indem das Spiel mittels Anzeigen und visuellem Feedback digitalisiert wird.

5.2 Beschreibung der Idee

Die Digitalisierung des Spiels erfolgt in mehreren Schritten. Zuerst soll das handelsübliche „Looping Louie“ Spiel mit einem Motorcontroller ausgestattet werden, um die Geschwindigkeit des Dreharms steuern zu können. Anschließend sollen die Lebenspunkte der Spieler, sogenannte HP, elektronisch erfasst und ausgewertet werden, um dann auf einer Anzeige zu erscheinen. Zusätzlich soll ein visueller Hinweis den Spieler alarmieren, wenn er HP verliert.

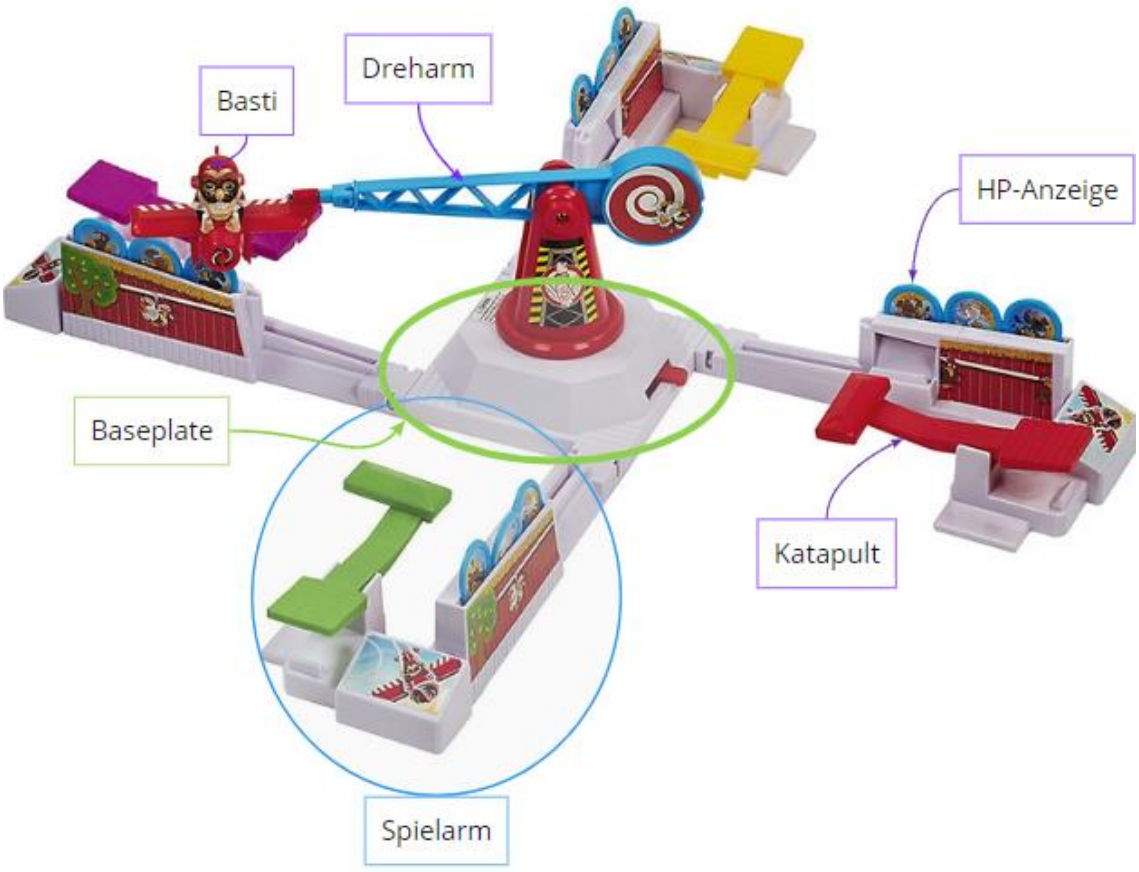


Abbildung 1: Visuelle Darstellung des Grundspiels und dessen Komponenten

6 Projektplanung

6.1 Projektziele

6.1.1 Hauptziele

RE-M 1	**Einstellen unterschiedlicher Geschwindigkeiten des Dreharms mittels eines Motorcontrollers**
RE-M 2	**Elektronische Erfassung der Lebenspunkte (HP) der Spieler**
RE-M 3	**Einbau einer visuellen Anzeige der derzeitigen HP**
RE-M 4	**Einbau eines visuellen Hinweises bei Verlust von HP**

Alle vier Hauptziele des Projekts wurden erfüllt. Angefangen mit RE-M 1 wurde ein Motorcontroller verwendet, der den Motor mittels einer H-Brücke ansteuert und so die Geschwindigkeit regeln kann. Dies lässt sich mittels Tastern auf der Elektronikbox einstellen. RE-M 2 wurde mittels einer Lichtschranke realisiert, die einen Verlust der HP erkennt und an einen Mikrocontroller weiterleitet. RE-M 3 und RE-M 4 wurden durch LEDs im Spielarm erfüllt, RE-M 4 spezifisch ist hierbei ein bemerkbares Blinken der LEDs.

6.1.2 Optionale Ziele

RE-O 1	Umbau des Katapults zu einer elektronischen Variante
RE-O 2	Einbau von Audiofeedback
RE-O 3	Implementation von verschiedenen Spielmodi oder Schwierigkeiten
RE-O 4	Implementation von Handicaps und Boni
RE-O 5	Einbau von Displays
RE-O 6	Implementation von Minispielen
RE-O 7	Erhöhung der Spieleranzahl

Die optionalen Ziele wurden zum Teil erfüllt. RE-O 1 hätte den größten Ressourcen-Aufwand bedeutet und wurde aus zeitlichen Gründen nicht erfüllt. RE-O 2 wurde durch einen kleinen Lautsprecher in der Elektronikbox realisiert. RE-O 3 wurde teilweise erfüllt, da mit den 3

einstellbaren Geschwindigkeiten auch theoretisch 3 verschiedene Schwierigkeitsstufen existieren, jedoch wurde dieses Ziel nicht noch weiterverfolgt. Da RE-O 5 nicht erfüllt wurde, wurden RE-O 4 und RE-O 6 ebenfalls nicht erfüllt, da das Display hierfür notwendig gewesen wäre. RE-O 7 wäre neben RE-O 2 das nächste optionale Ziel gewesen, da nur das 3D-Drucken und Verkabeln neuer Arme und eines Adapters für zusätzliche Arme notwendig gewesen wäre. Dies wurde auch aus zeitlichen Gründen nicht realisiert.

6.1.3 NICHT Ziele

RE-N 1	Einbau eines Online-Modus
RE-N 2	Entwicklung einer App
RE-N 3	Veränderung des Dreharms
RE-N 4	Rückwärtskompatibilität

6.2 Projektteam

Dominik Eminger ist Projektleiter und verantwortlich für die Projektplanung, die Mechanik, Elektronik und Programmierung. Er wird extern unterstützt durch Lukas Eminger und als persönlicher Betreuer steht Martin Sommer zur Verfügung, für deren Unterstützung ich mich sehr herzlich bedanke.

Abbildung 2: Dominik Eminger

6.3 Objekt-Struktur-Plan und Projekt-Struktur-Plan

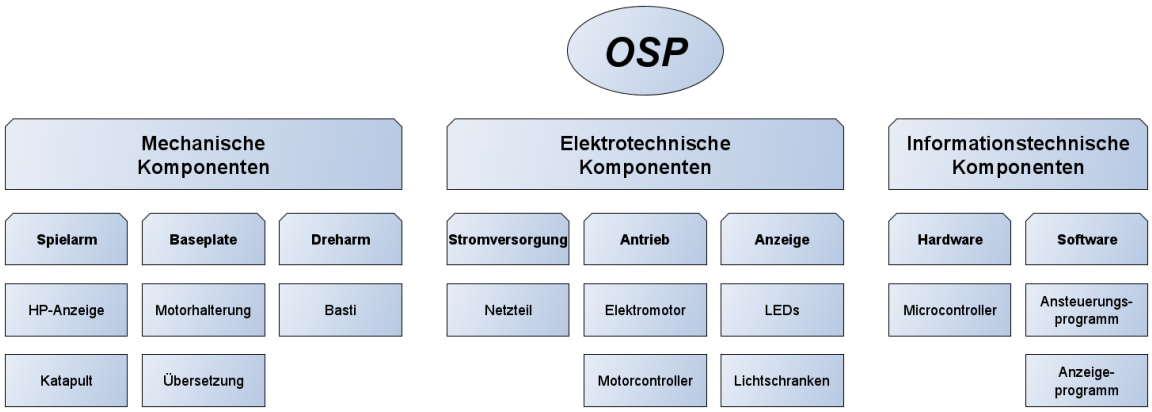


Abbildung 3: OSP Objekt-Struktur-Plan

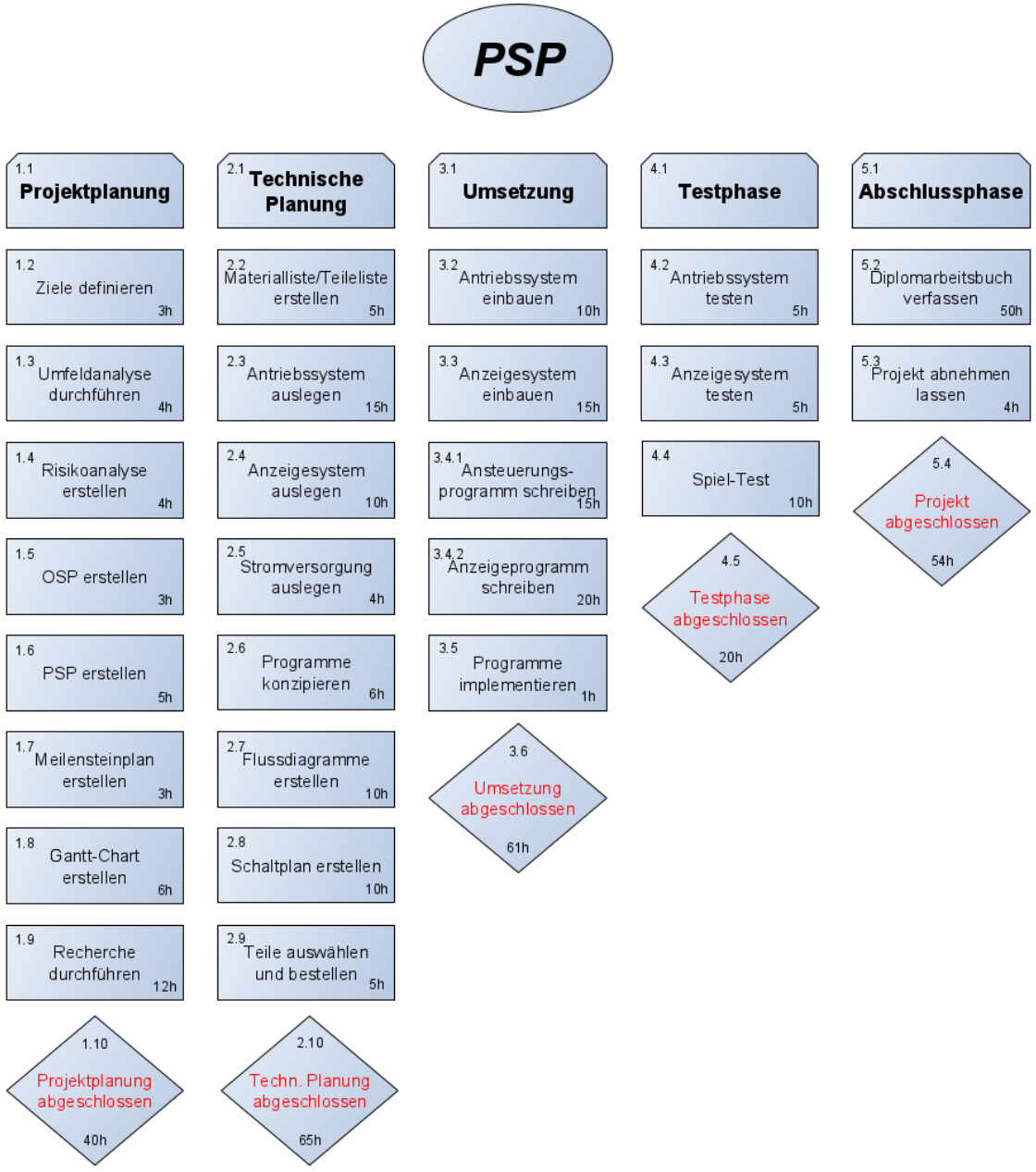


Abbildung 4: PSP Projekt-Struktur-Plan

6.4 Umfeldanalyse

6.4.1 Grafische Darstellung

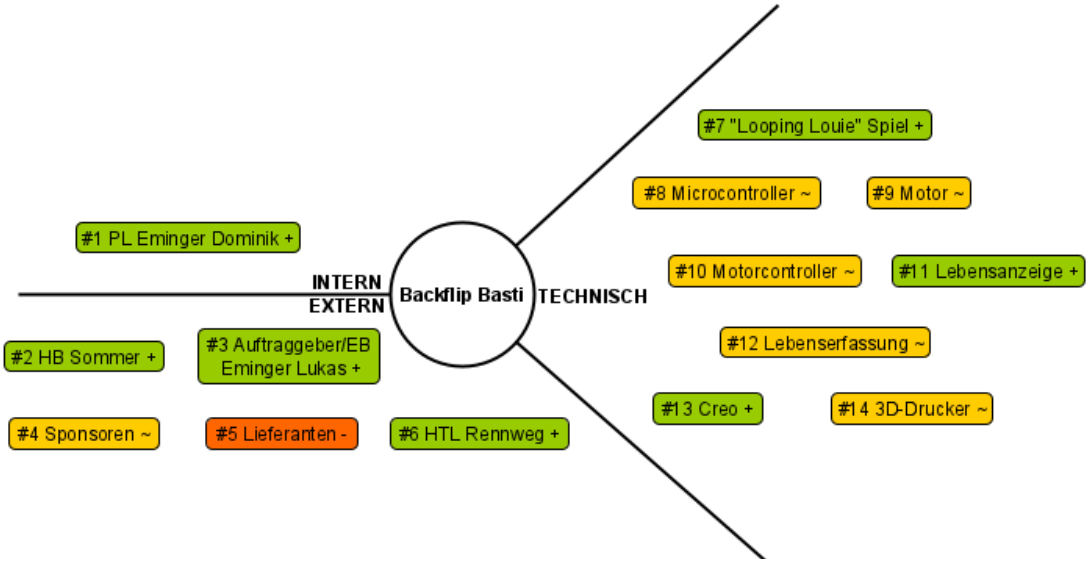


Abbildung 5: Umfeldanalyse

6.4.2 Beschreibung der wichtigsten Umfelder

#	Bezeichnung	Beschreibung	Bewertung
1	PL Eminger	Planung und Durchführung des Projekts	+
2	HB Sommer	Unterstützung bei Elektrotechnik und Programmierung	+
3	Auftraggeber/EB Eminger	Externe Betreuung bei 3D-Druck und Projektmanagement	+
4	Sponsoren	Eventuell kein Sponsoring	~
5	Lieferanten	Eventuell lange Lieferzeiten	-
6	HTL Rennweg	Bereitstellung von Räumlichkeiten	+
7	„Looping Louie“-Spiel	Wichtiger Bestandteil	+
8	Microcontroller	Wichtiger Bestandteil	~
9	Motor	Wichtiger Bestandteil	~
10	Motorcontroller	Wichtiger Bestandteil	~
11	Lebensanzeige	Wichtiger Bestandteil	+
12	Lebenserfassung	Wichtiger Bestandteil	~
13	Creo	CAD-Modellierung von 3D-Druck-Teilen	+
14	3D-Drucker	Wichtiges Werkzeug	~

Tabelle 1: Beschreibung der Umfelder

6.5 Zeitplan

6.5.1 Meilensteine

Geplantes Datum	Tatsächliches Datum	Meilenstein
15.11.2020	15.11.2020	Projektplanung abgeschlossen
25.11.2020	23.11.2020	Diplomarbeitsantrag eingereicht
09.12.2020	07.12.2020	Anzeigesystem ausgelegt
23.12.2020	07.01.2021	Technische Planung abgeschlossen
13.01.2021	01.01.2021	Anzeigesystem eingebaut
27.01.2021	03.01.2021	Programme implementiert
27.01.2021	09.03.2021	Umsetzung abgeschlossen
10.02.2021	12.03.2021	Testphase abgeschlossen
03.03.2021	21.03.2021	Diplomarbeitsbuch komplett verfasst
17.03.2021	24.03.2021	Projektabnahme
26.03.2021	26.03.2021	Abgabe des Korrekturexemplars
19.05.2021	19.05.2021	Defensio

Tabelle 2: Meilensteine

Die Meilensteine wurden bis zur Implementierung der Programme weitestgehend eingehalten, wonach sich ein größerer Abstand zum nächsten Meilenstein befindet. Dieser Abstand ist hauptsächlich durch die große Lieferverzögerung des 3D-Druckers entstanden, der das Prototyping und Erstellen der 3D-Druck Teile sowie den Einbau der Elektronik in diese Teile nach hinten verschob. Jedoch konnte dies durch das frühzeitige Fertigstellen der Testprogramme in einem Testaufbau zumindest teilweise aufgeholt werden.

6.5.2 Arbeitsstunden pro Arbeitspaket

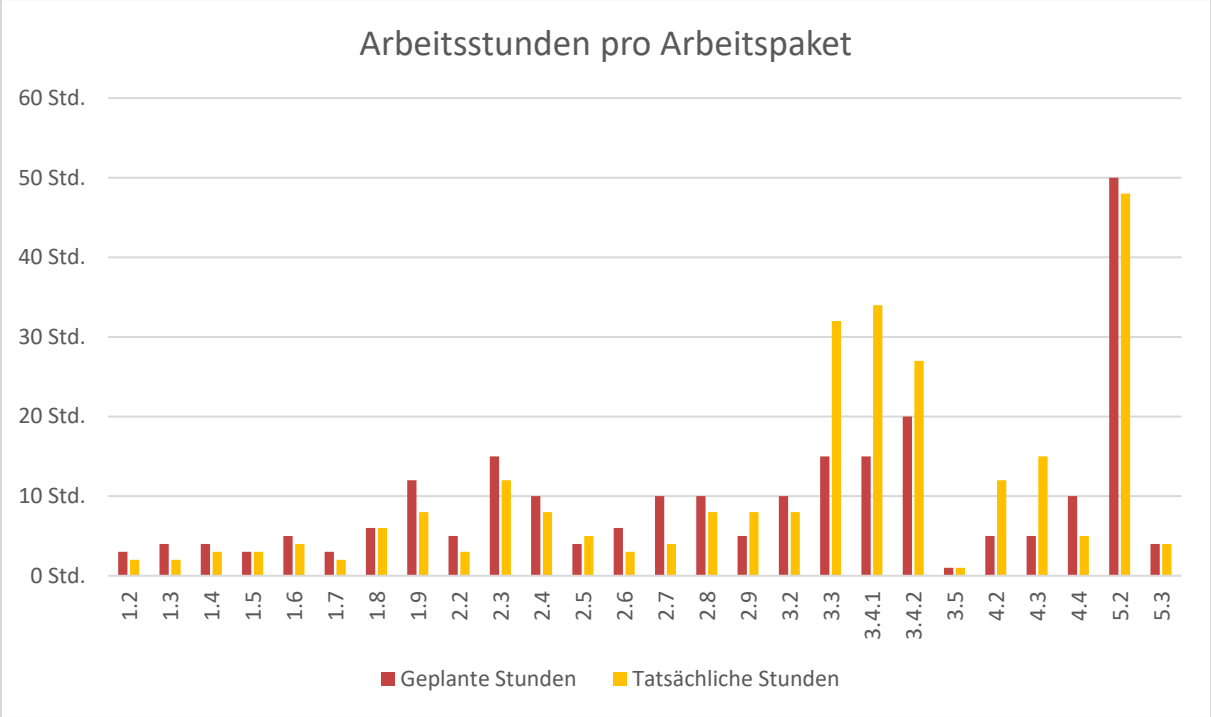


Abbildung 6: Arbeitsstunden pro Arbeitspaket

Die im Diagramm dargestellten Arbeitspakete sind im *Kapitel 6.3 OSP und PSP* in *Abbildung 4: PSP* zu finden. Die meisten Arbeitspakete wurden recht gut eingeschätzt und größtenteils schneller als geplant abgeschlossen, mit einigen Ausnahmen. Am auffälligsten sind hierbei Arbeitspakete 3.3, 3.4.1 und 3.4.2, das Schreiben der Ansteuerungs- und Anzeigeprogramme und das Einbauen des Anzeigesystems. Die Programmierung unterlief mehreren Versionen pro Programm und war durch mehrmaliges Korrigieren des Codes zeitaufwändiger als geplant. Beim Einbauen des Anzeigesystems zeigte sich ein Denkfehler in der Planung, da nicht bedacht worden war, dass dieses Arbeitspaket einmal pro Spielarm, also in Summe viermal, zu machen ist.

Für Arbeitspakete 4.2 und 4.3, das Testen der Systeme, wurde ebenfalls etwas mehr Zeit in Anspruch genommen, um sicherzustellen, dass keine Fehler vorliegen.

Hierbei ist noch erwähnenswert, dass das Designen und Drucken der 3D-Druck Teile ebenfalls ein sehr umfangreiches Arbeitspaket darstellte, welches in der Planung nicht vernachlässigt wurde, jedoch einen großen Teil des Projekts ausmachte.

6.5.3 Arbeitsstunden gesamt

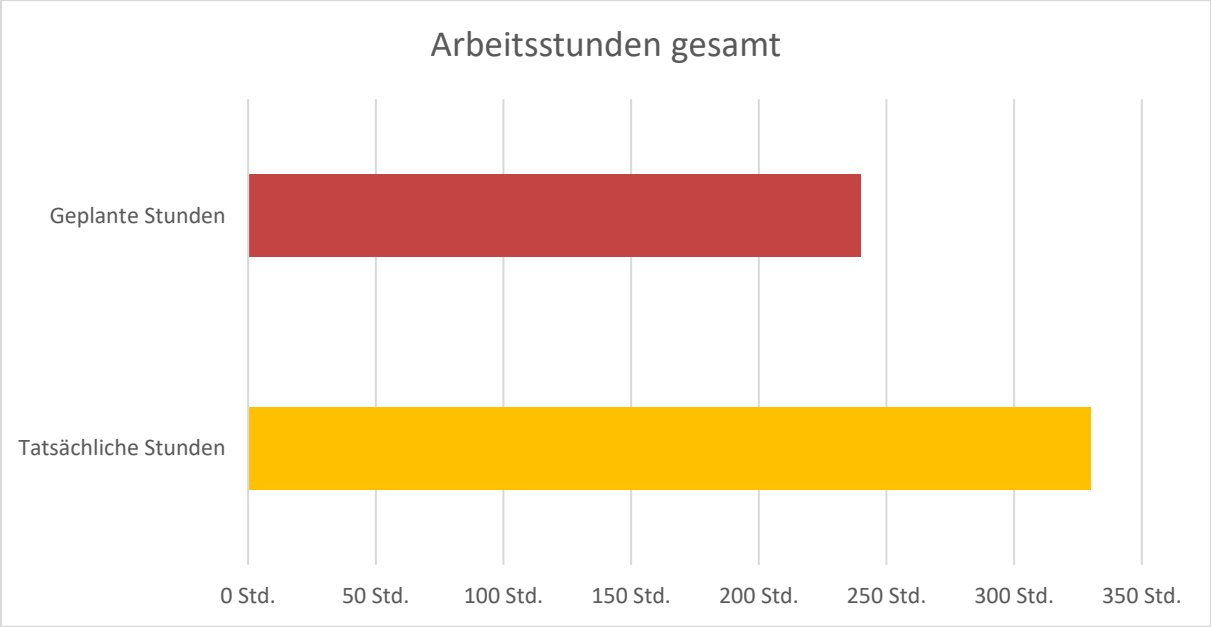


Abbildung 7: Arbeitsstunden gesamt

Die Planung der Arbeitsstunden war grundsätzlich recht realistisch, wenn nicht das Arbeitspaket des Designens und Druckens der 3D-Druck Teile in der Planung gefehlt hätte. Dieses hat zirka 60 Stunden in Anspruch genommen, die Druckzeit selbst wurde hierbei nicht mitgerechnet. Aber unabhängig davon wurde die Planung um ungefähr 20 Stunden überschritten.

6.6 Risikomanagement

6.6.1 Beschreibung der wichtigsten Risiken

#	Bezeichnung	Beschreibung des Risikos	P	A	RF
4	Sponsoren	Höhere Kosten, falls kein Sponsoring	70	20	1400
5	Lieferanten	Verspätungen bei Lieferzeiten	60	80	4800
8	Microcontroller	Komplikationen mit Programmierung/Elektrotechnik	65	75	4875
9	Motor	Unpassender Motor	35	80	2800
10	Motorcontroller	Komplikationen mit Ansteuerung	50	70	3500
12	Lebenserfassung	Eventuelle Hardware-Probleme	60	30	1800
14	3D-Drucker	Eventuelle Störungen oder Ungenauigkeiten	40	80	3200

Tabelle 3: Beschreibung der wichtigsten Risiken

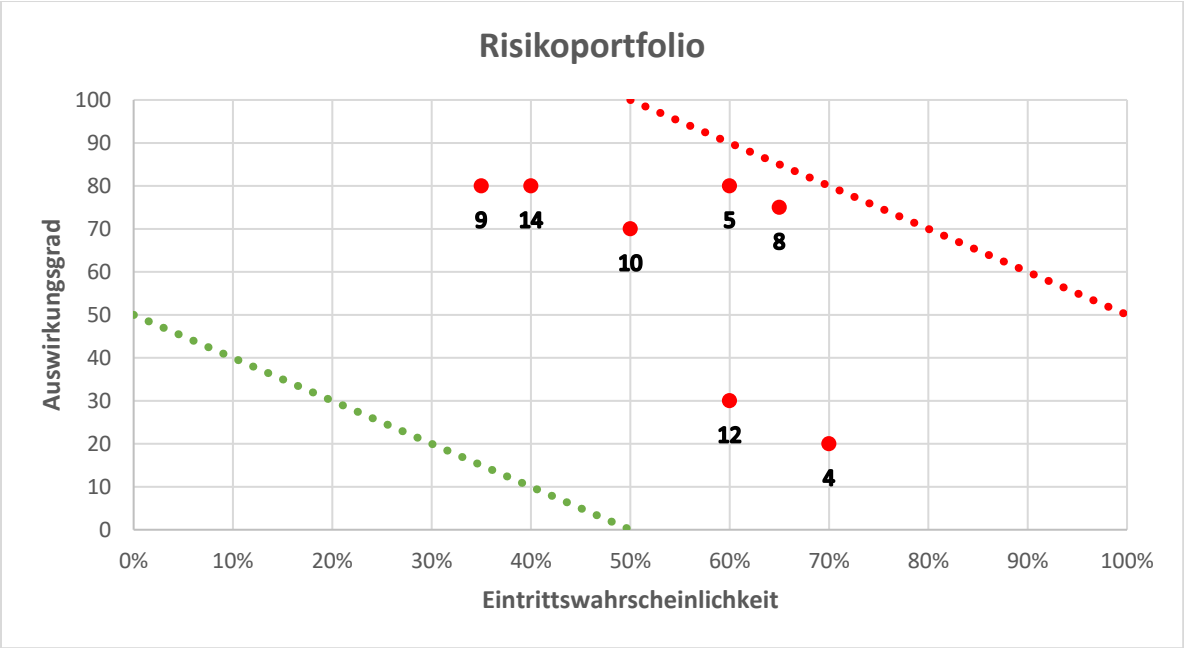


Abbildung 8: Risikoportfolio

Von den geplanten Risiken sind nur wenige tatsächlich aufgetreten. Das Hauptproblem war die Lieferzeit des Druckers, welches unter Risiko 5 fällt. Risiko 1 trat ebenfalls ein, jedoch war dies kein Problem, da das etwas höhere Budget durch Eigenkosten abgedeckt werden konnte.

6.6.2 Risiko Gegenmaßnahmen

#	Bezeichnung	Gegenmaßnahme
4	Sponsoren	Viele Sponsoren anfragen bzw. Budget gut einteilen
5	Lieferanten	Frühzeitig bestellen und zuverlässige Lieferservices benutzen, Pufferzeiten miteinberechnen
8	Microcontroller	Gute Planung bei problematischen Gebieten und schnelles Reagieren, falls Probleme auftreten
9	Motor	Rechtzeitige und gut überlegte Auslegung und Auswahl
10	Motorcontroller	Simplen Motorcontroller auswählen, um Komplikationen zu vermeiden
12	Lebenserfassung	Frühe und häufige Testphasen, um Hardware-Probleme schnellstmöglich zu entfernen
14	3D-Drucker	Anderen 3D-Drucker verwenden

Tabelle 4: Risiko-Gegenmaßnahmen

6.7 Kosten

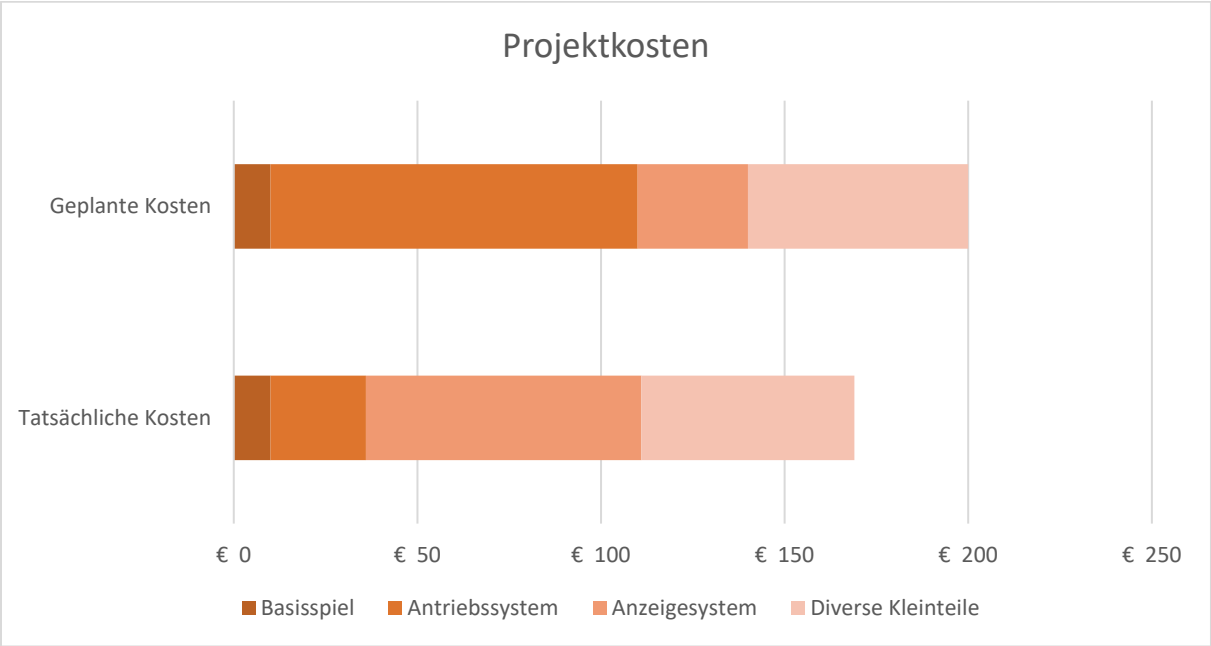


Abbildung 9: Projektkosten

Die Kostenplanung wurde eher grob gerechnet, da zum Zeitpunkt der Planung noch nicht klar war, welche Teile tatsächlich Verwendung finden würden. Daher kommen auch die Abweichungen sowohl im Antriebssystem als auch im Anzeigesystem. Die Planung wurde auf einen neu gekauften Motor im Antriebssystem ausgelegt, welcher im Endeffekt nicht notwendig war und auf einen Mikrocontroller im Anzeigesystem, wovon im Endeffekt mehrere verwendet wurden. Die Schätzung der Basisspiel- und Kleinteilkosten war jedoch ziemlich akkurat.

7 Mechanik und 3D-Druck

7.1 Problemstellung

Es musste einerseits der Spielarm neu entworfen werden, um die Elektronik darin unterbringen und den bestehenden Hebel darauf montieren zu können. Zusätzlich musste eine Möglichkeit gefunden werden, den neuen Spielarm auf der modifizierten Baseplate anzubringen. Die Spielarme sollten einfach von der Baseplate abnehmbar sein, um den Transport des gesamten Spiels zu vereinfachen. Zuletzt musste noch Platz für die Elektronik der Motorsteuerung gefunden werden.

Diese Bestandteile sollten mittels eines Prusa Mini 3D-Druckers gedruckt werden. Wie der Name impliziert, hat der Mini eine recht kleine Druckfläche mit nur 18x18cm, dementsprechend musste dies im Design der Teile berücksichtigt werden.

7.2 Lösungsansätze

Es stellte sich rasch heraus, dass ein neuer Motor und somit eine neue oder modifizierte Baseplate nicht erforderlich sind. Dadurch entstand der erste Lösungsansatz, die neuen Spielarme direkt mit der ursprünglichen Baseplate zu verbinden. Da dies die Verkabelung jedoch extrem erschwert hätte, wurde diese Idee wieder verworfen und nach Alternativen gesucht.

Somit entstand die Idee eines Rahmens, der die bestehende Baseplate einfasst und leichte Verbindungspunkte für die Spielarme als auch Platz für deren Verkabelung bietet. Da die Baseplate nicht mehr verändert wurde, musste ein neuer Platz für die Elektronik der Motorsteuerung gefunden werden. Dies wurde mittels einer schlichten Elektronikbox realisiert, die ebenfalls am Rahmen angebracht werden sollte.

7.3 Verwendete 3D-Druck Techniken

Es wurde beim Entwurf der 3D-Druck Teile Wert auf leichte Druckbarkeit und Minimierung des verbrauchten Druckmaterials gelegt, um das Prototyping einfach zu halten. Dies wurde auch teilweise mit dem externen Betreuer Lukas Eminger besprochen, der hierbei gute Tipps geben konnte.

Als Hauptverbindungsart zwischen den 3D-Druck Teilen wurde eine simple Schwalbenschwanz-Steckverbindung gewählt. Durch die präzisen Eigenschaften des Druckers war es möglich, verschiedene Passungen im 0.05mm Bereich zu testen und für die dementsprechende Anwendung zu wählen.

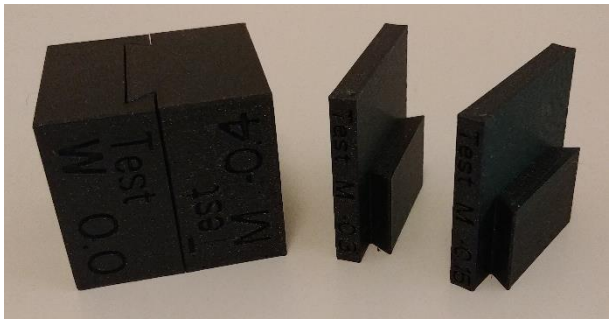


Abbildung 10: Verschiedene Tests der Schwalbenschwanzverbindungen

So wurde beispielsweise für die Verbindung zwischen Spielarm und Rahmen, welche oft getrennt und wieder verbunden werden soll, eine Spielpassung gewählt, während für die Elektronikbox am Rahmen, welche nicht wieder getrennt werden sollte, eine Presspassung besser war.

Der große Vorteil der Fertigung der Teile durch 3D-Druck war das Prototyping und die Fähigkeit, schnell in der Entwicklung voranzukommen, ohne einen bemerkbaren Arbeitsaufwand in der Herstellung zu haben. Dies wurde vor allem in den Spielarmen und im dazugehörigen Hebel stark genutzt, da diese nicht zu sehr vom Original abweichen konnten, aber auch viele Veränderungen mit sich bringen mussten, um Platz für die Elektronik zu schaffen.

Ein weiteres Feature, das beim 3D-Druck eher einzigartig ist, ist die Möglichkeit, Teile in einen laufenden Druck einzubauen und diese damit einfach in den Druck zu integrieren. Das Slicer-Programm der Firma Prusa *(PrusaSlicer)* bietet eine Druckpause als Tool für genau solche Anwendungen. Dies wurde beim Projekt hauptsächlich in der Elektronikbox verwendet, um Sechskantmuttern in den Druck zu integrieren, welche als Basis für Schraubverbindungen dienen.

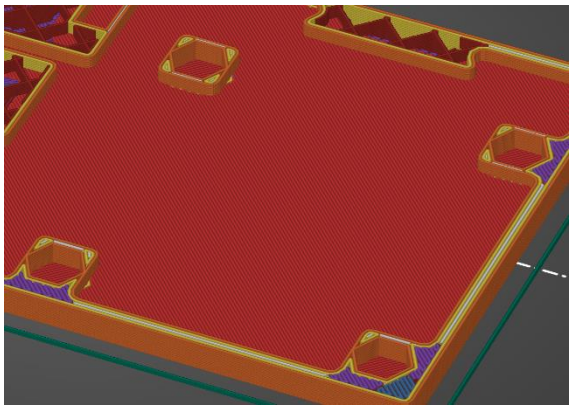


Abbildung 11: Schnitt der Elektronikbox im PrusaSlicer

7.4 Rahmen

Zuerst wurde der Rahmen entworfen, in dem Platz für die Verkabelung vorhanden sein sollte und der mit der Baseplate verbunden wird.

Die erste Version war recht nah an der finalen Version. Die Verbindungen für die Spielarme wurden mittels einer Schwalbenschwanzverbindung realisiert, ebenso wie die Verbindung für die Elektronikbox, jedoch wurde für die Elektronikbox eine engere Toleranz gewählt, da diese nicht leicht entfernbar sein sollte. Das Design der Verbindungspunkte der Baseplate basiert auf den Steckverbindungen der Arme des Originalspiels.

Es wurden unterhalb der Verbindungspunkte für die Baseplate Öffnungen für die Kabel entworfen, um als Kabelkanäle zu dienen, die die Verbindungsstecker der Spielarme nach außen führen.

Diese Version wurde jedoch verworfen, da festgestellt wurde, dass die Verbindungspunkte für die Baseplate und somit auch die der Spielarme falsch positioniert waren.

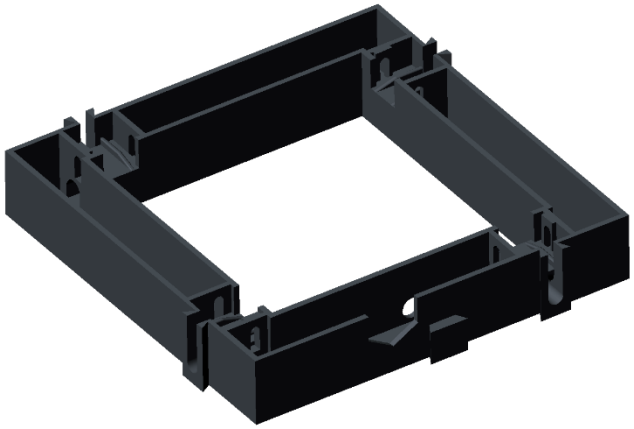


Abbildung 12: Rahmen Version 1

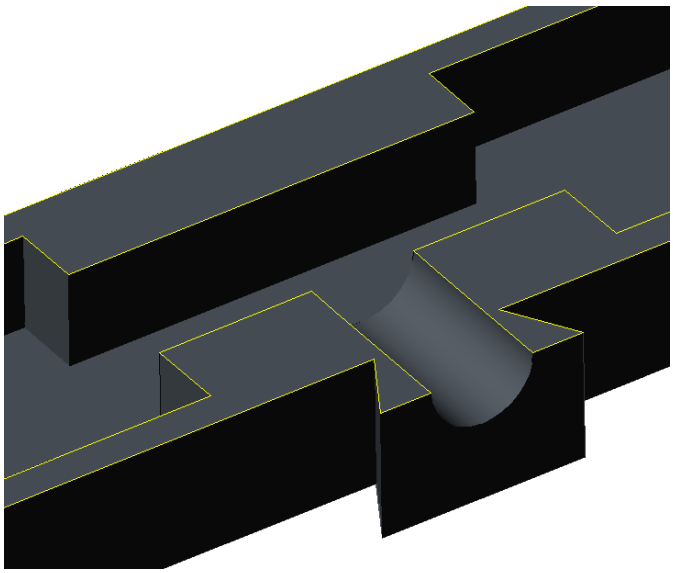


Abbildung 13: Rahmen Schnittansicht Kabelkanäle

Die zweite Version behob dieses Problem, ebenso wurden leichte Veränderungen in den Toleranzen der Schwalbenschwanzverbindungen vorgenommen.

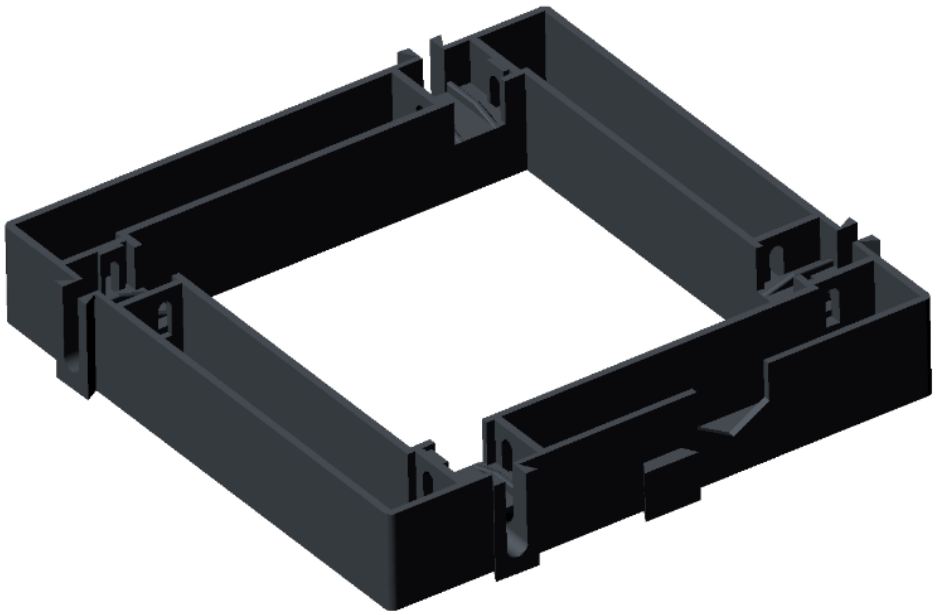


Abbildung 14: Rahmen Version 2

7.5 Elektronikbox und Deckel

7.5.1 Elektronikbox

Die Elektronikbox war der zweite Teil, der entworfen wurde. Es waren durch den Drehradius des Dreharms und der Maße des Spielarms bereits maximale Maße bekannt, die Breite der Elektronikbox wurde dem dem Rahmen angepasst. Da in der Elektronikbox die meisten Bauteile auf eher engem Raum eingebaut waren, war diese vom Designstandpunkt das aufwändigste Teil. Deswegen wurde bei der Elektronikbox auch eine Baugruppe mit den Elektronikbauteilen erstellt, die zirka die Maße der Originalteile hatten, um sicherzustellen, dass alle Teile ohne Probleme in der Box Platz finden.

Die erste Version der Elektronikbox wurde nach der ersten, nicht passenden Version des Rahmens entworfen, dementsprechend musste die Box ebenfalls umgeändert werden, um mit dem neuen Rahmen kompatibel zu sein.

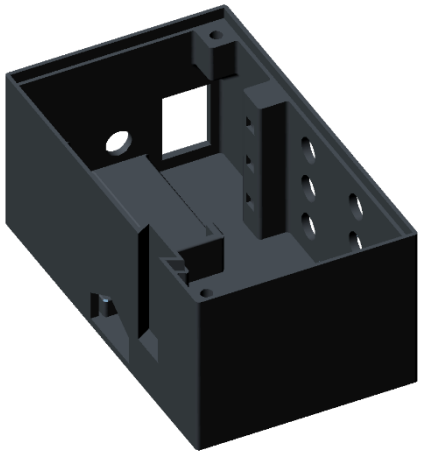


Abbildung 15: Elektronikbox Version 1

Mit der zweiten Version der Elektronikbox wurde dies wieder behoben. Da die erste Version nie gedruckt wurde, fanden auch keine Ausbesserungen statt, die auf den Drucker spezifisch ausgerichtet waren.

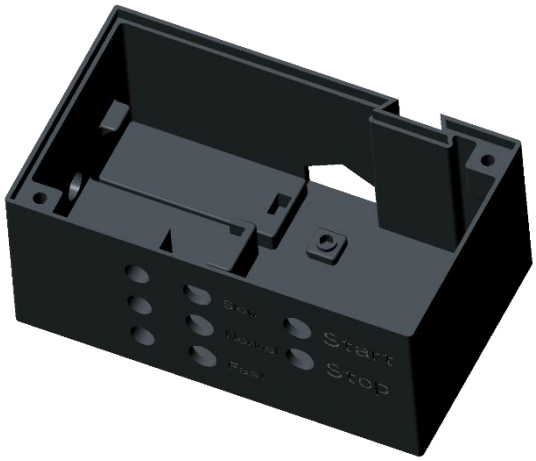


Abbildung 16: Elektronikbox Version 2

Die Box wurde größtenteils so entworfen, dass die Elektronikteile darin einfach einzubauen sind und es wurde vermieden, diese zusätzlich befestigen zu müssen, beispielsweise mit Klebmaterial oder Schrauben. Die Ausnahmen hierfür sind die Leuchtdioden neben den Tastern *(Abbildung 17, violett)*, welche mit Heißkleber in den dafür vorgesehenen Öffnungen befestigt wurden sowie der Motorcontroller *(Abbildung 18, grün)*, welcher mit den eingesetzten M3 Muttern und Schrauben befestigt wurde.

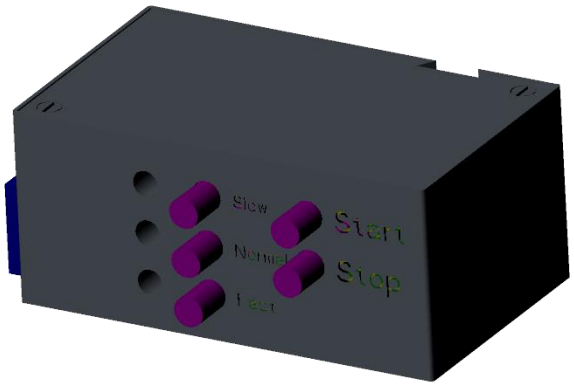


Abbildung 17: Elektronikbox mit groben Teilen und Deckel

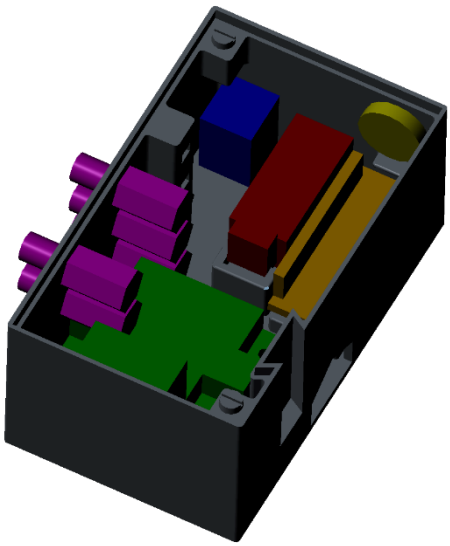


Abbildung 18: Elektronikbox mit groben Teilen und Einsicht in Box

7.5.2 Deckel

Der Deckel der Elektronikbox hatte bei der ersten Version keine gröberen Funktionsprobleme. Es fiel allerdings auf, dass die Lautstärke des Lautsprechers nicht optimal durchdrang und dass der Deckel teilweise etwas schwierig zu entfernen war.

Somit wurden in der zweiten Version ein Gitter im Deckel für den Lautsprecher sowie ein seitlicher Schlitz für leichteres Anheben des Deckels eingefügt. Ebenso wurden die Öffnungen für die M3 Schrauben etwas vergrößert, damit die Schrauben leichter gedreht werden können.

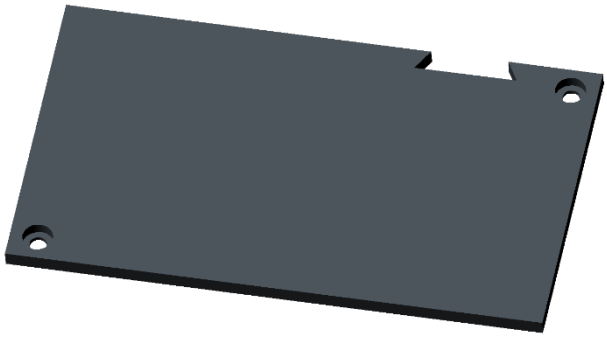


Abbildung 19: Deckel der Box, Version 1

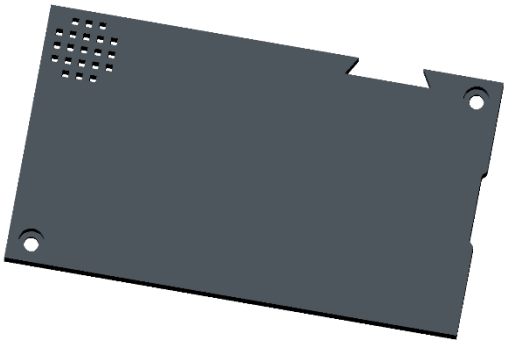


Abbildung 20: Deckel der Box, Version 2

7.6 Spielarm und Hebel

7.6.1 Spielarm

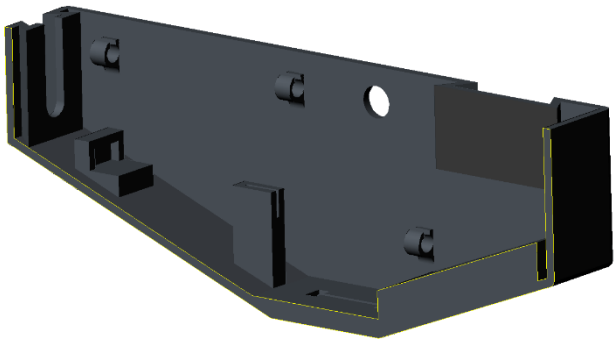


Abbildung 21: Spielarm Version 1

Der Spielarm wurde zwar als erstes Teil entworfen, aber die fertige Version 1 wurde erst nach den anderen Teilen konstruiert. Es wurde hierbei mit verschiedenen Konzepten gearbeitet, schlussendlich wurde jedoch das Konzept eines dünnen Spielarms umgesetzt, an dem der Hebel mittels einer Schwalbenschwanzverbindung angebracht wird. Dieses war auch das Konzept mit dem geringsten Materialverbrauch.

Nach dem ersten Druck wurde festgestellt, dass die Kabelführungsschlaufen zu dünn waren und leicht abbrachen. Auch war die Position für die Öffnung des Tasters schlecht gewählt, da dieser quer nicht in den Spielarm passen würde.

In der zweiten Version des Spielarms wurden basierend auf den Erfahrungen des ersten Drucks die Schlaufen für die Kabel vergrößert und verstärkt, sowie die Öffnung für den Taster an die Vorderseite des Arms gesetzt. Zusätzlich dazu wurden noch kleinere Veränderungen vorgenommen, wie das Verlängern des Schwalbenschwanzes auf der Innenseite, um unnötiges Stützmaterial im Druck zu vermeiden.

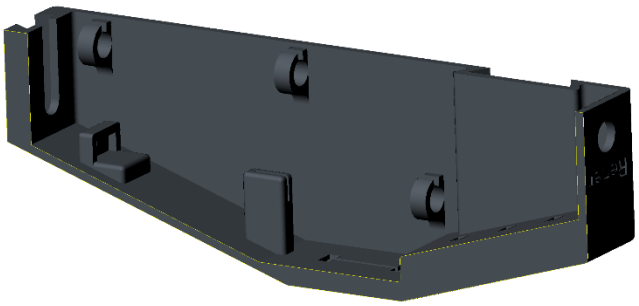


Abbildung 22: Spielarm Version 2

7.6.2 Hebel

Die Halterung des Hebels wurde sehr stark an das Design des Originalspiels angelehnt, da dieser Teil mit dem originalen Spielehebel funktionieren sollte. Es wurden daher hauptsächlich die Maße des Originals übernommen und an den Spielarm angepasst.

Jedoch wurden für den Hebel allein im Endeffekt 3 verschiedene Versionen gedruckt, da die geringe Wandstärke beim Hebel für Probleme sorgte. Das im Spiel verwendete ABS ist wesentlich stärker als 3D-gedrucktes PLA, deswegen musste hier mit verschiedenen Wandstärken getestet werden, bis eine passende gefunden wurde.

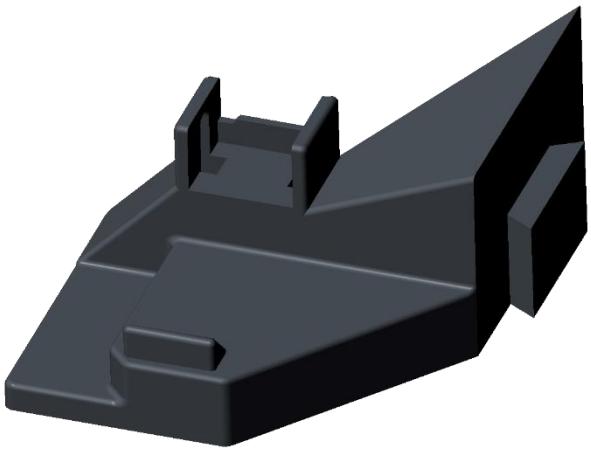


Abbildung 23: Hebel Version 3

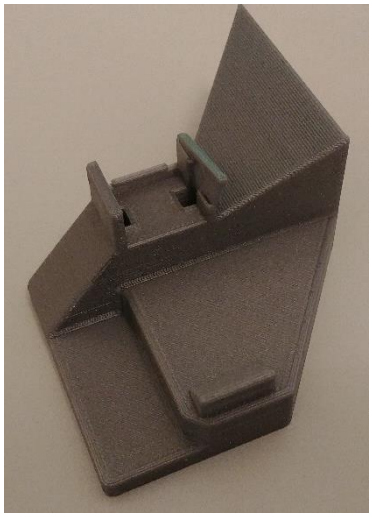


Abbildung 24: Druck des Hebels, Version 2; geringe Wandstärke führte zu Bruch

8 Elektronik

8.1 Problemstellung

Um den Motor ansteuern zu können, musste ein passender Motorcontroller gewählt werden. Es musste auch ein Konzept für die visuelle Anzeige und den visuellen Hinweis erstellt und getestet werden, sowie für die elektronische Erfassung der HP.

Zuletzt mussten die fertigen Systeme in die 3D-Druckteile eingebaut werden.

8.2 Lösungsansätze

Zuerst wurde an einer Lösung für den Spielarm gearbeitet, da dieser für das Testen des Konzepts vorerst wichtiger war. Es wurde entschieden, die visuelle Anzeige vorerst mit Leuchtdioden zu realisieren und später eventuell noch auf ein anderes Display umzusteigen. Der visuelle Hinweis sollte dann über das Programm hinzugefügt werden. Die elektronische Erfassung sollte mittels einer Lichtschranke geschehen, durch den sich der „fliegende Basti“ auf dem Dreharm bewegt.

Da beim Originalspiel der Lebensverlust mit einer Münze dargestellt ist, die nach Kontakt mit dem Flieger nach unten fällt, wurde die Münze für den Testaufbau durch die Lichtschranke ersetzt und Leuchtdioden am Arm angebracht, um die visuelle Anzeige zu testen.

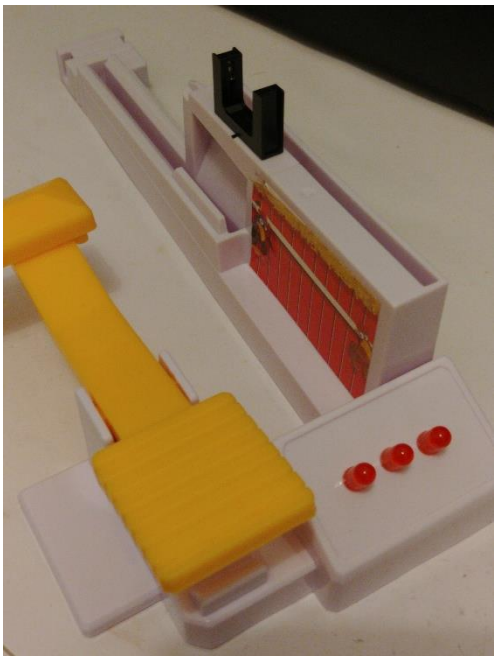


Abbildung 25: Testaufbau des Spielarms

Der Testaufbau war auch für die Mechanik sehr hilfreich, da die genaue Position der Lichtschranke für den neuen Spielarm somit gemessen und festgelegt werden konnte. Andere Maße, wie die Position des Hebels, die sich ebenfalls nicht ändern sollte, konnten auf diese Weise auch gemessen werden.

Nach dem Testaufbau wurde die Elektronikbox entwickelt. Sie führte den Motorcontroller mit dessen Ansteuerung zusammen, welche mittels Tastern realisiert wurde, mit denen die Geschwindigkeit auswählbar ist. All dies sollte über einen Arduino Nano gesteuert werden.

8.3 Elektronikbox

Die Elektronikbox ist das Herzstück der Elektronik. Auf der linken Seite befinden sich der Hauptschalter und die Buchse für den Netzteilstecker, welcher das System mit 7,5 V Gleichstrom versorgt. Der Motorcontroller wandelt diesen für den Arduino und die Spielarme mittels eines integrierten Spannungsreglers in 5 V um. An der Vorderseite befinden sich die Taster für die Motorsteuerung sowie Leuchtdioden, die als Anzeige für die jeweils gewählte Geschwindigkeit dienen.

Abbildung 26: Elektronikbox

8.4 Spielarm

Der Spielarm wurde ähnlich wie die Elektronikbox mit einem Arduino Nano für das Auslesen der Werte und Steuern der Leuchtdioden ausgestattet. Es war ursprünglich geplant, nur einen Arduino für das gesamte Projekt zu verwenden und die Datenübertragung der Sensoren und LEDs anders zu regeln. Da der Arduino aber zu wenige I/O Pins hat, wurde pro Spielarm ein Arduino verbaut, der für den jeweiligen Arm zuständig ist, um auch Zeit in der Programmierung zu sparen.

Auf der Oberseite des Spielarms befindet sich eine Lichtschranke, die den Verlust von HP erkennt, wenn „Basti“ die Lichtschranke durchkreuzt. Somit wird ein Signal an den Arduino geschickt, welcher über die LEDs die derzeitigen Leben ausgibt. Zusätzlich dazu ist auf der Vorderseite des Spielarms ein „Reset-Button“, der die Lebenspunkte auf 3 HP zurücksetzt. Dies wurde der Einfachheit halber eingefügt, um das Testen zu erleichtern. Die Alternative wäre gewesen, die Leben automatisch zurückzusetzen, nachdem sie null erreicht haben, jedoch wurde beim Testen empfunden, dass dies den Spielfluss unterbricht, also wurde der Taster auch für die finale Version übernommen.

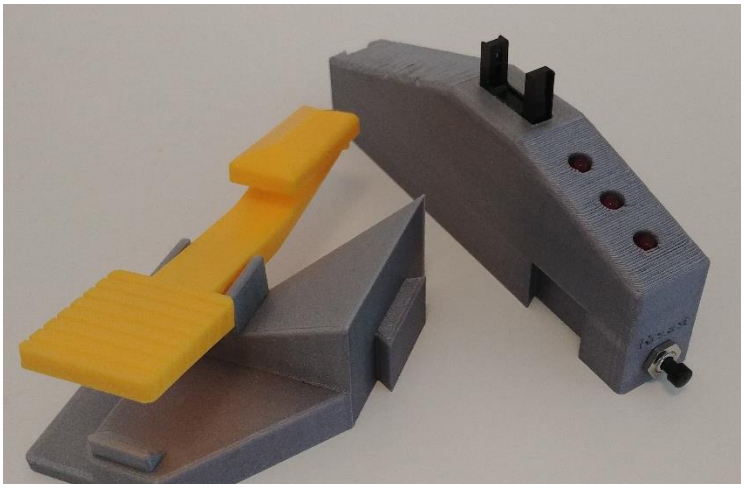


Abbildung 27: Spielarm mit Hebel

8.5 Komponenten

8.5.1 Arduino Nano

Der Arduino Nano ist der wichtigste Teil in der Elektronik, da er über einen Mikrocontroller verfügt, mit dem das ganze System angesteuert wird. Aufgrund seiner einfachen Programmierweise und ausreichenden Anzahl an In- und Output-Pins wurde er bevorzugt gewählt.

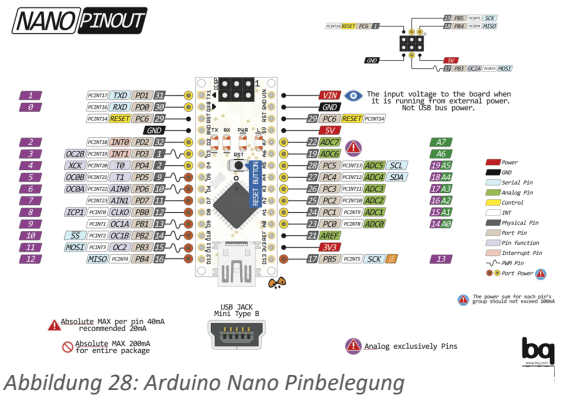


Abbildung 28: Arduino Nano Pinbelegung

Technische Daten	
Mikrocontroller	ATMega328P-AU
Betriebsspannung	5 V
Versorgungsspannung	7-12 V
Digitale I/O Pins	22 (6 davon PWM)
Analoge I/O Pins	8
Speicher	32 kB/ EEPROM 1 kB
Taktfrequenz	16 MHz

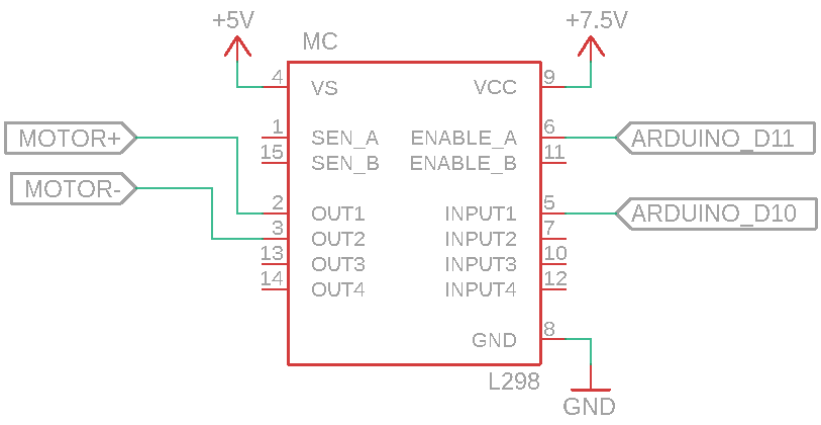
Tabelle 5: Technische Daten Arduino Nano

8.5.2 Motorcontroller

Für den Motorcontroller wurde der L289N gewählt, da dieser bereits vorhanden war und in einem ähnlichen Projekt verwendet wurde. Dieser Motorcontroller wird öfter mit Arduinos verwendet, da ein eingebauter Spannungsregler 5 V für den Mikrocontroller ausgibt.

Der Motorcontroller ist darauf ausgelegt, Motoren von 5-35 V anzusteuern. Der Motor des Looping-Louie Spiels wird mit 2 AA-Batterien betrieben und bekommt somit zirka 3 V Spannung. Da der Motor über ein PWM-Signal angesteuert wird und die Geschwindigkeit einstellbar ist, ist dies ausreichend für das Projekt.

Abbildung 29: Motorcontroller L28



Technische Daten	
Versorgungsspannung	3,2-40 V
Ausgangsspannung	5-35 V
Betriebsstrombereich	0-36 mA
Maximalstrom	2 A

Tabelle 6: Technische Daten L289N

8.5.3 Taster

Für die Taster wurde der TC-MT250 gewählt, da er passende Maße hat und für die Anwendung im Projekt mehr als ausreichend ist.

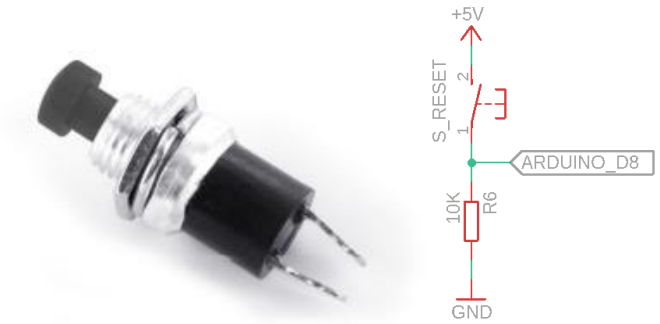


Abbildung 30: Taster TC-MT250 und Anschluss

Der Taster benötigt für eine optimale Funktionsweise einen Pull-Down Widerstand von 10 kΩ, der das Signal auf das Erdpotential *(GND)* zieht, damit der Pin am Arduino nur dann ein Signal vom Taster bekommt, wenn dieser tatsächlich betätigt wird.

Technische Daten	
Max. Betriebsspannung	250 V
Max. Betriebsstrom	0.5 A
Kontakt-Widerstand	50 mΩ

Tabelle 7: Technische Daten TC-MT250

8.5.4 Leuchtdioden

Als Leuchtdioden wurden übliche 5mm 5 V LEDs verwendet. Sie werden einerseits in der Elektronikbox verwendet, um die derzeitige Geschwindigkeit anzuzeigen, aber auch im Spielarm zur Lebensanzeige. Als Vorwiderstand wurde ein 330 Ω Widerstand verwendet.

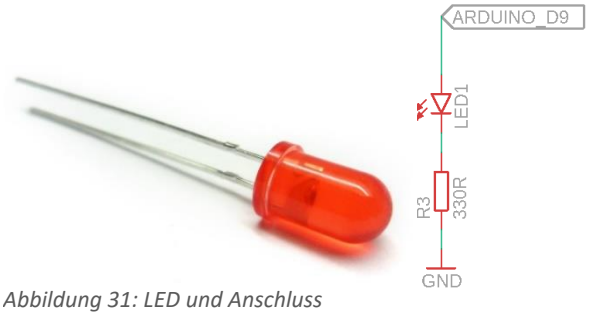


Abbildung 31: LED und Anschluss

Technische Daten	
Empfohlene Betriebsspannung	2 V
Max. Betriebsstrom	20 mA

Tabelle 8: Technische Daten LED

8.5.5 Lichtschranke

Die Lichtschranke, die für den Spielarm gewählt wurde, ist eine LTH-301-32. Sie entspricht den Maßen, die bei der Auslegung des Spielarms festgelegt wurden und ist mit einer Versorgungsspannung von 5V mit dem Rest der Elektronik kompatibel.

Die Lichtschranke sendet je nachdem, welcher Widerstand am Collector *(siehe Abbildung 32, „COL"-Pin)* hängt, ein Signal an den Arduino. Mit dem gewählten 10 kΩ - Widerstand liegt die Reaktionszeit der Lichtschranke bei zirka 100 µs, also 0.1 ms.

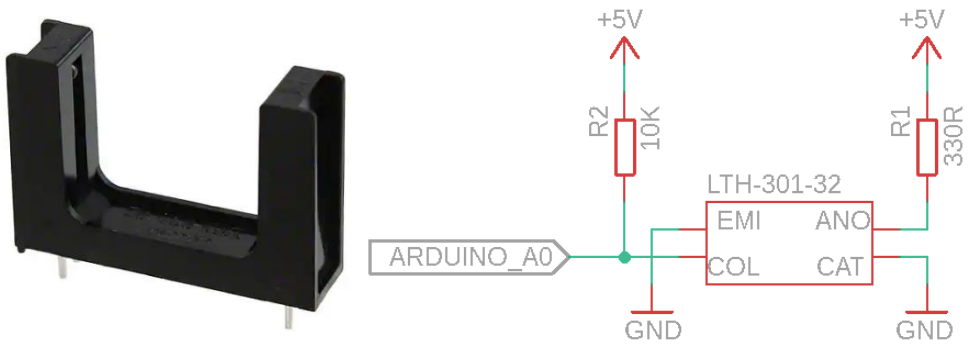


Abbildung 32: Lichtschranke LTH-301-32 und Anschluss

Technische Daten	
Maximale Spannung an Diode	5 V
Max. Betriebsstrom Diode	60 mA
Maximale Spannung Transistor	30 V
Max. Betriebsstrom Transistor	20 mA

Tabelle 9: Technische Daten LTH-301-32

8.5.6 Lautsprecher

Der Lautsprecher, der für die Audio-Signale ausgewählt wurde, ist ein Visaton K-16 Miniaturlautsprecher.

Er wurde wegen seiner kleinen Maße und ausreichenden Lautstärke gewählt.

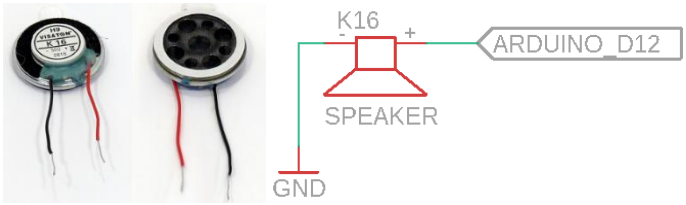


Abbildung 33: Lautsprecher K16 und Anschluss

Technische Daten	
Nennbelastbarkeit	500 mW
Impedanz	50 Ω
Übertragungsbereich	700 – 20000 Hz
Mittlerer Schalldruckpegel	71 dB

Tabelle 10: Technische Daten K-16

8.6 Verkabelung

Nach dem Testen der Programme mit der Hardware wurden die Löt- und Verkabelungsarbeiten durchgeführt.

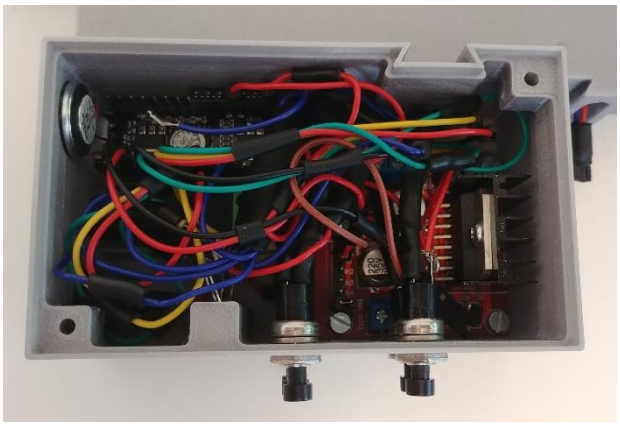


Abbildung 34: Verkabelung Elektronikbox

Es wurde recht früh gegen eine Platine entschieden, da das Designen und Fertigen einer oder mehrerer Platinen mit der recht geringen Bauteilanzahl und dem Platzmangel in den Spielarmen und in der Elektronikboxkeine gute Idee gewesen wäre. Einerseits musste so keine Rücksicht auf die Platzierung der Platine gelegt werden, andererseits mussten dafür mehr Halterungen für die verschiedenen Elektronikteile konstruiert werden und der Lötaufwand wurde dadurch erhöht.

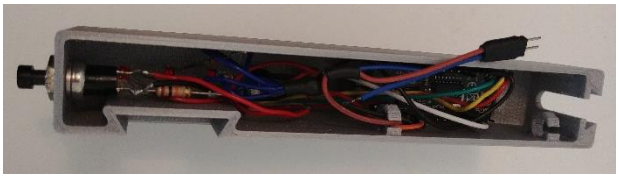


Abbildung 35: Verkabelung Spielarm

Es wurden beim Löten, wie in Abbildung 34 zu sehen ist, viele Schrumpfschläuche verwendet, um sicherzustellen, dass Kontakte zwischen Kabeln, und somit auch Kurzschlüsse, nicht geschehen.

Um den Motor zu verkabeln, mussten an der Baseplate einige Modifikationen gemacht werden, wie das Entfernen der bestehenden Elektronik und das Bohren einer Öffnung, um die Kabel von der Elektronikbox zum Motor führen zu können.

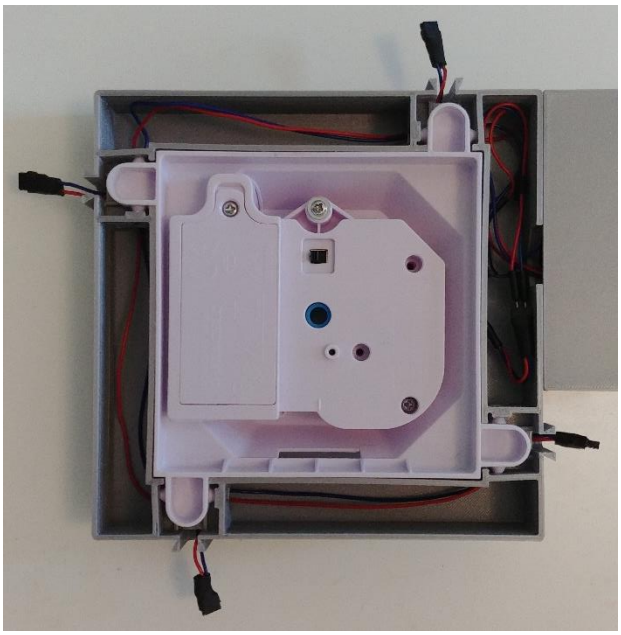


Abbildung 36: Verkabelung Rahmen

9 Programmierung

Der Quellcode der beiden Programme befindet sich in ausgedruckter Form im Anhang.

9.1 Problemstellung

Es musste ein Programm für das Anzeigesystem und für das Antriebssystem konzipiert und geschrieben werden. Für das Anzeigesystem müssten Daten von der Lichtschranke ausgelesen und verarbeitet werden, um den Verlust eines Lebenspunkts zu erkennen, welcher dann mittels der LEDs angezeigt werden soll.

Für das Antriebssystem sollten verschiedene Geschwindigkeiten mittels Tastern eingestellt werden können und diese dann an den Motorcontroller weitergeleitet werden, um somit den Motor anzusteuern.

9.2 Lösungsansätze

Anfänglich wurde getestet, ob das Anzeigesystem mit einfachen if-Schleifen realisiert werden könnte, jedoch wurde diese Idee recht bald verworfen und stattdessen ein endlicher Zustandsautomat für die Durchführung gewählt.

Dieser erlaubt es, die Lichtschranke dauerhaft abzufragen und nach einem Signal einfach in den nächsten Zustand zu wechseln. Dieses Programm wurde schon für den Testaufbau *(siehe Abbildung 25: Testaufbau des Spielarms)* verwendet, um das Konzept des fertigen Spiels zu testen. Mit dem Testaufbau konnte auch die Geschwindigkeit des Originalspiels überprüft werden. Es wurde also ein kurzes Test-Programm geschrieben und somit festgestellt, wie lange eine Runde durchschnittlich dauert, zirka 3 Sekunden.

```
505, Timer 1: 8387 ms
525, Timer 2: 11423 ms
Roundtime = 3036 ms
528, Timer 1: 14456 ms
500, Timer 2: 17481 ms
Roundtime = 3025 ms
514, Timer 1: 20500 ms
525, Timer 2: 23518 ms
Roundtime = 3018 ms
502, Timer 1: 26529 ms
529, Timer 2: 29535 ms
Roundtime = 3006 ms
```

Abbildung 37: Testergebnisse des Geschwindigkeits-Tests

Für das Antriebssystem wurde ebenfalls ein endlicher Zustandsautomat gewählt, da oft zwischen verschiedenen Zuständen gewechselt werden müsste. Es wurde mit dem Geschwindigkeits-Test überprüft, welche PWM ungefähr welcher Geschwindigkeit entspricht, bis die Rundenzeit dem des Originalspiels entsprach, dieser Wert wurde dann für die „Normal“-Geschwindigkeit gewählt.

9.3 Motorsteuerung

9.3.1 Flussdiagramm

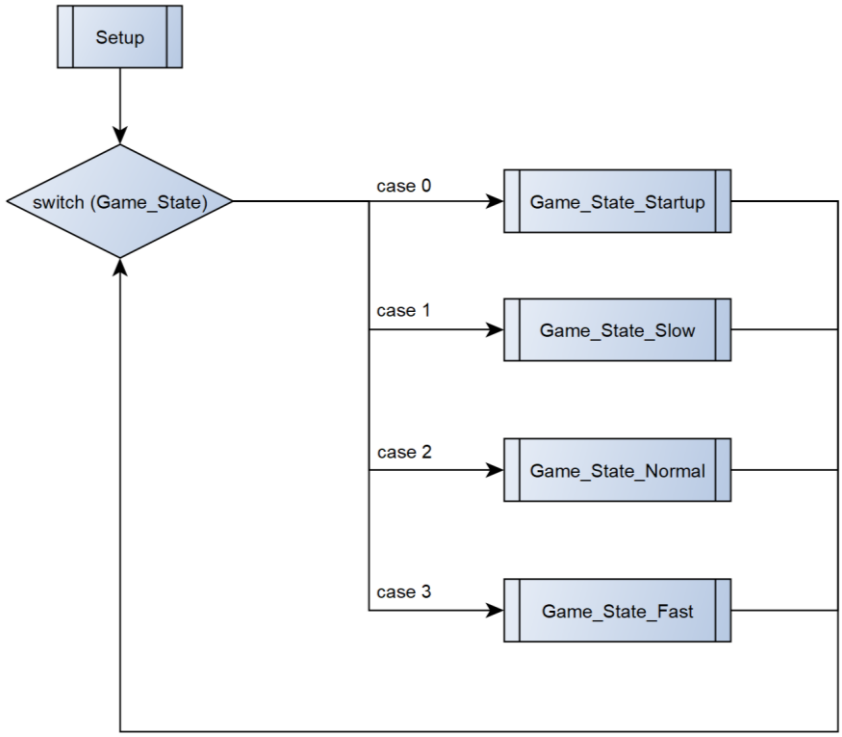


Abbildung 38: Flussdiagramm der Motorsteuerung

9.3.2 Zustandsdiagramm

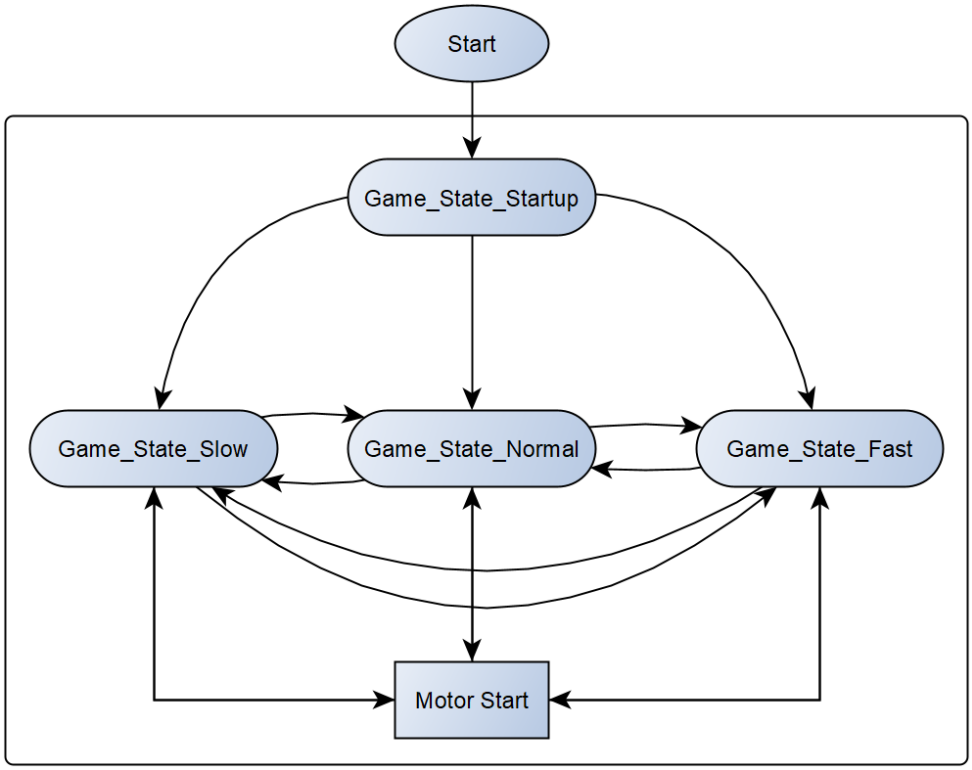


Abbildung 39: Zustandsdiagramm der Motorsteuerung

Wie in Abbildung 38 zu sehen ist, wurde das Motorsteuerungs-Programm so konzipiert, dass nach dem Startup jederzeit in einen der drei Hauptzustände, beziehungsweise zwischen den Zuständen selbst gewechselt werden kann. Mit Betätigung des Start-Buttons kann auch von den Zuständen aus der Motor mit der dementsprechenden Geschwindigkeit gestartet werden.

9.3.3 Zustände

9.3.3.1 Startup-Zustand

```
int Game_State_Startup(){                       //Startup-Zustand (Zustand 0), wird nach Einschalten sofort ausgeführt

  digitalWrite(Led_Slow, LOW);                  //Ausschalten der LEDs für den Fall, dass noch welche aktiv waren
  digitalWrite(Led_Normal, LOW);
  digitalWrite(Led_Fast, LOW);

  i = 0;                                        //Setzen der Counter-Variable auf 0

  Slow_Value = digitalRead(Button_Slow);        //Auslesen der Taster und Zuschreiben der Werte
  Normal_Value = digitalRead(Button_Normal);
  Fast_Value = digitalRead(Button_Fast);

  if(Slow_Value == HIGH){                       //Falls der "Slow" Taster gedrückt wird, auf Zustand 1 wechseln
    return Game_State = 1;
  }
  else if(Normal_Value == HIGH){                //Falls der "Normal" Taster gedrückt wird, auf Zustand 2 wechseln
    return Game_State = 2;
  }
  else if(Fast_Value == HIGH){                  //Falls der "Fast" Taster gedrückt wird, auf Zustand 3 wechseln
    return Game_State = 3;
  }
}
```

Abbildung 40: Programmausschnitt der Motorsteuerung, Startup-Zustand

Im Startup-Zustand werden zuerst alle LEDs der Elektronikbox ausgeschalten, für den Fall, dass durch einen Fehler noch eine LED leuchtet. Anschließend werden die Taster überprüft und die Counter-Variable i auf 0 gesetzt. Diese verhindert, dass später im Programm die Tonfolgen wiederholt werden. Sollte einer der Taster gedrückt werden, wird in den jeweiligen Zustand gewechselt, Zustand 1 entspricht hierbei dem „Slow“-Zustand, Zustand 2 dem „Normal“-Zustand und Zustand 3 dem „Fast“-Zustand.

9.3.3.2 Hauptzustände

```
int Game_State_Slow(){                          //Slow-Zustand (Zustand 1)

  digitalWrite(Led_Slow, HIGH);                 //Einschalten der "Slow" LED, Ausschalten der "Normal"- und "Fast" LED
  digitalWrite(Led_Normal, LOW);
  digitalWrite(Led_Fast, LOW);

  while(i < 1){                                 //Schleife, die nur einmal ausgeführt wird, um "Slow"-Tonfolge zu spielen
    tone(speaker, 80, 150);
    delay(150);
    tone(speaker, 90, 150);
    i++;
  }

  Start_Value = digitalRead(Button_Start);      //Auslesen der Taster und Zuschreiben der Werte
  Stop_Value = digitalRead(Button_Stop);
  Slow_Value = digitalRead(Button_Slow);
  Normal_Value = digitalRead(Button_Normal);
  Fast_Value = digitalRead(Button_Fast);

  if(Stop_Value != HIGH){                       //Führt Programm aus, solange der "Stop"-Taster nicht gedrückt wird

    if(Start_Value == HIGH){                    //Falls der "Start"-Knopf gedrückt wird, wird der Motor mit niedriger Geschwindigkeit angeschalten
      analogWrite(MotorPWM, 50);
      digitalWrite(MotorIN, HIGH);
    }
    else if(Normal_Value == HIGH){              //Falls der "Normal"-Knopf gedrückt wird, wird der Motor ausgeschalten, die Counter-Variable auf 0 gesetzt und auf Zustand 2 gewechselt
      digitalWrite(MotorIN, LOW);
      analogWrite(MotorPWM, 0);
      i = 0;

      return Game_State = 2;
    }
    else if(Fast_Value == HIGH){                //Falls der "Fast"-Knopf gedrückt wird, wird der Motor ausgeschalten, die Counter-Variable auf 0 gesetzt und auf Zustand 3 gewechselt
      digitalWrite(MotorIN, LOW);
      analogWrite(MotorPWM, 0);
      i = 0;

      return Game_State = 3;
    }
  }
  else if(Stop_Value == HIGH){                  //Falls der "Stop"-Knopf gedrückt wird, wird der Motor ausgeschalten und wieder zum Anfang von Zustand 1 gewechselt
    digitalWrite(MotorIN, LOW);
    analogWrite(MotorPWM, 0);

    return Game_State = 1;
  }
}
```

Abbildung 41: Programmausschnitt der Motorsteuerung, „Slow“-Zustand

Die Hauptzustände „Slow“, „Normal“ und „Fast“ verhalten sich alle gleich, bis auf minimale Unterschiede, daher wird hier nur einer der Zustände veranschaulicht.

Im „Slow“-Zustand werden zuerst alle LEDs, bis auf die „Slow“ LED ausgeschalten, um anzuzeigen, dass dieser Zustand ausgewählt wurde. Darauf folgt eine kleine Tonfolge, die dem „Slow“-Zustand eigen ist.

Nun werden ähnlich wie im Startup-Zustand alle Taster ausgelesen. Als Sicherheitsmaßnahme wurde ein Ungleich-Operator (!=) für den „Stop“-Taster eingefügt, damit das Programm unterbrochen wird, sollte dieser gedrückt werden. Der „Start“-Taster startet den Motor mit der dementsprechenden Geschwindigkeit, in diesem Fall mit niedriger Geschwindigkeit.

Nun kann der Motor mit dem „Stop“-Taster jederzeit wieder gestoppt werden, aber es kann auch währenddessen in andere Zustände gewechselt werden, wobei der Motor sich immer zuerst ausschaltet. Die Zustände können auch problemlos vor Betätigen des „Start“-Tasters gewechselt werden.

9.4 Anzeigesystem

9.4.1 Flussdiagramm

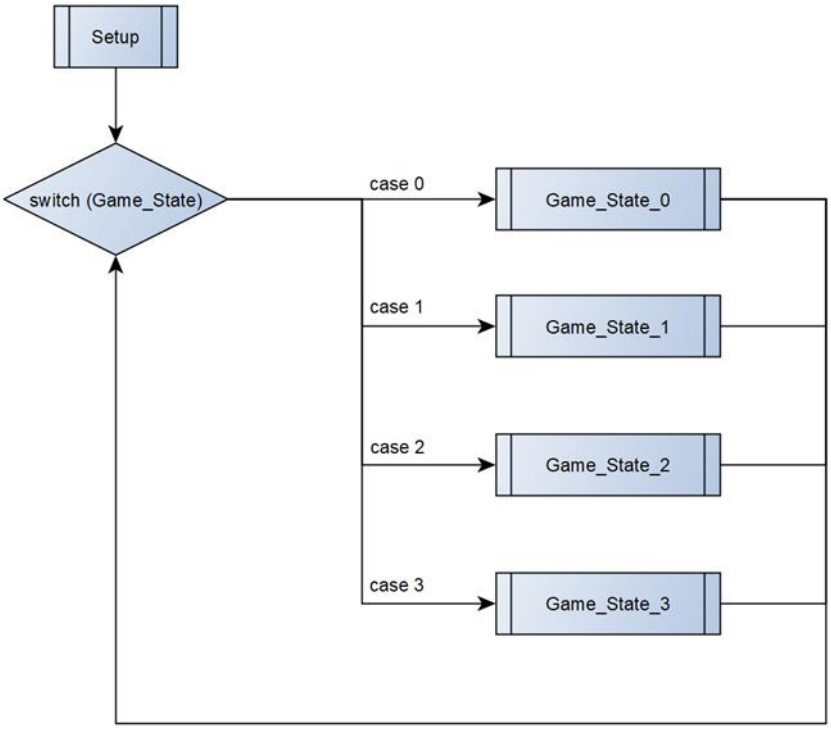


Abbildung 42: Flussdiagramm des Anzeigesystems

9.4.2 Zustandsdiagramm

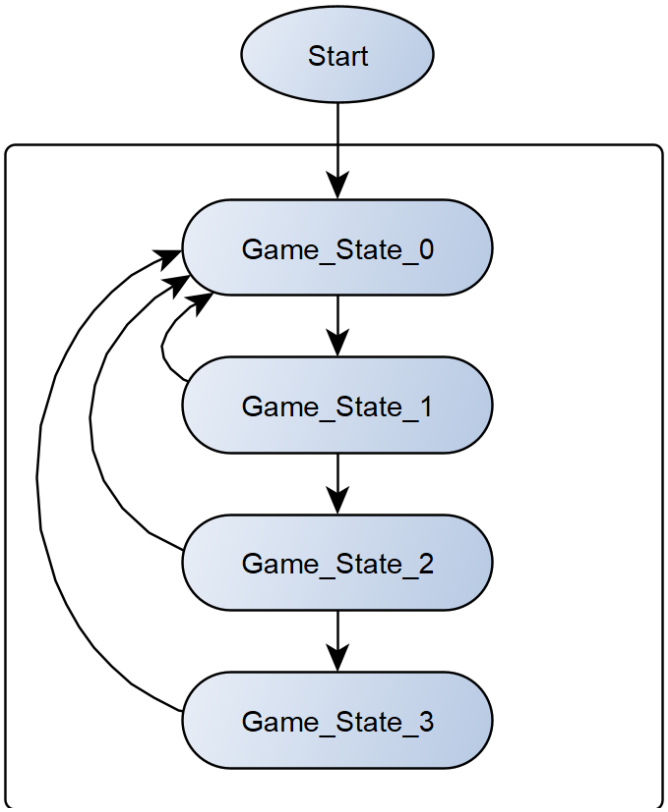


Abbildung 43: Zustandsdiagramm des Anzeigesystems

Im Vergleich zur Motorsteuerung ist der Ablauf des Anzeigesystems viel linearer und einfacher. Es wird von Zustand 0 bis zu Zustand 3 gewechselt, mit der Option jederzeit zu Zustand 0 zurückzusetzen.

9.4.3 Zustände

```
int Game_State_0(){
  sensorValue = analogRead(sensorPin);        //Auslesen des Sensors und Zuschreiben des Wertes
  ButtonValue = LOW;                          //Wert des Reset-Buttons auf 0 setzen

  analogWrite(led_1Pin, LedLow);              //Alle LEDs auf mittlere Helligkeit setzen
  analogWrite(led_2Pin, LedLow);
  analogWrite(led_3Pin, LedLow);

  ButtonValue = digitalRead(ResetButton);     //Auslesen des Tasters und Zuschreiben des Wertes
  if(ButtonValue == HIGH){                    //Falls Taster gedrückt wird, wieder auf Zustand 0 zurückgehen
    return Game_State = 0;
  }

  if(sensorValue >= 500){                     //Falls Lichtschranke ein Signal bekommt, erste LED blinken lassen
    analogWrite(led_1Pin, LedHigh);
    delay(280);
    analogWrite(led_1Pin, LedOff);
    delay(280);
    analogWrite(led_1Pin, LedHigh);
    delay(280);
    analogWrite(led_1Pin, LedOff);
    delay(280);
    analogWrite(led_1Pin, LedHigh);
    delay(280);

    return Game_State = 1;                    //Übergang zu Zustand 1
  }
}
```

Abbildung 44: Programmausschnitt des Anzeigesystems, Zustand 0

Beim Anzeigesystem sind die Zustände ebenfalls alle gleich aufgebaut, als Beispiel hier wird Zustand 0 dargestellt.

Zustand 0 ist der Anfangszustand und entspricht 3 Lebenspunkten, signalisiert dadurch, dass alle 3 LEDs leuchten. Zuerst wird die Lichtschranke ausgelesen und der Wert des Reset-Buttons auf LOW, also gleich Null gesetzt.

Entsprechend dem Zustand werden anschließend darauf die LEDs an- beziehungsweise ausgeschalten. Wie in Abbildung 43 zu sehen ist, werden die LEDs auf die mittlere Helligkeitsstufe gesetzt. Dies wurde so gewählt, damit das Blinken der LED mit voller Helligkeit auffälliger ist, während immer noch angezeigt wird, wieviele HP der Spieler derzeit hat.

Vor dem Auslösen der Lichtschranke kann nun noch der Reset-Button gedrückt werden, um von jedem Zustand in Zustand 0 zu wechseln und die HP so essenziell wieder auf 3 zurückzusetzen.

Falls die Lichtschranke ein Signal bekommt, blinkt die oberste LED dreimal, bevor sie mit dem Wechsel in den nächsten Zustand erlischt.

10 Lessons Learned

10.1 Allgemein

Das Projekt war, als mein nun dritter Versuch für die Diplomarbeit, das erste Projekt, das ich tatsächlich machen wollte. Auch wenn ich zuvor die Chance hatte, ein eigenes Projekt durchzuführen und auch ein Thema gewählt habe, das mich interessiert, war es immer noch eine Mühe, mich selbst zu motivieren. Mit diesem Projekt hatte ich, auch durch Unterstützung meiner Familie, viel mehr Motivation, es zu Ende zu bringen und daran zu arbeiten. Mir ist klar geworden, dass um Hilfe zu bitten nicht zwangsweise schlecht ist, vor allem wenn es nur kleine Dinge sind, bei denen ich mich nicht vollständig ausgekannt habe.

Ich hätte zwar gerne mehr der optionalen Ziele erfüllt und das Projekt in diesem Zeitrahmen zu dem gemacht, was anfangs geplant war, aber leider war die starke Verspätung des Druckers mit fast 2 Monaten nicht vorherzusehen. Es waren jedoch genug Puffer eingeplant, um das Projekt rechtzeitig abzuschließen. Ich würde also sagen, dass das Projekt im Großen und Ganzen ein Erfolg war.

10.2 Mechanik

Trotz verspäteter Lieferung konnten mit dem 3D-Drucker sehr schnell und einfach Prototypen erstellt werden. Durch meinen externen Betreuer Lukas Eminger konnte ich auch einige Techniken über das FDM-Druckverfahren lernen und in dem Projekt anwenden.

Da ich zuvor persönlich nur mit einem älteren Drucker gearbeitet habe, war das Arbeiten mit dem Prusa Mini erstaunlich einfach. Einmal aufgesetzt konnte praktisch sofort gedruckt werden, ohne großartig viel zu kalibrieren oder einzustellen und der Prusa Slicer ist ebenfalls ein sehr gutes Tool, das ich auch in Zukunft oft verwenden werde.

10.3 Elektronik

Die Elektronik beim Projekt wurde recht simpel gehalten, da dies nicht gerade mein stärkstes Gebiet ist. Dennoch war vor allem gegen Ende des Projekts beim Einbau der Elektronik recht viel Aufwand durch das Löten jeder einzelnen Verbindung vorhanden.

Da für das Anzeigesystem und für die Anzeige der Elektronikbox kein Display, sondern LEDs gewählt wurden, mussten einige optionale Ziele damit weggestrichen werden, weil man dafür ein gewisses Interface bräuchte. Die 3 LEDs am Spielarm limitieren einen auch nur zu 3 Lebenspunkten maximal. Dies ist definitiv eines der Ziele, das ich rückblickend gerne erfüllt hätte.

Bei der Planung des Audiofeedbacks war auch anfangs unklar, wie es gemacht werden sollte. Zuerst war die Idee, in jeden Spielarm einen Lautsprecher zu verbauen, doch diese wurde wieder verworfen, da es vermutlich zu verwirrend für die Spieler wäre und die dauernden Audiosignale eventuell ablenkend wären, demnach wurde im Endeffekt nur in die Elektronikbox ein Lautsprecher verbaut.

10.4 Programmierung

Für die Programmierung wurde ein Arduino ausgewählt, da ich mit diesem System am meisten Erfahrung habe und keine „unnötige“ Arbeit in das Erlernen einer neuen Entwicklungsumgebung beziehungsweise einer neuen Programmiersprache stecken wollte.

Dies hat sich meiner Meinung auch ausgezahlt, da ich nur den Zustandsautomaten wieder erlernen musste, was kein Problem war. Der Rest der Programmierung hat unerwarteterweise auch Spaß gemacht, da ich keine Fehler hatte, an denen ich tagelang festhing, und der Arbeitsfluss damit auch angenehm war.

11 Quellenverzeichnis

https://www.3d-grenzenlos.de/3d-drucker-arten/#Fused_Deposition_Modeling_Fused_Filament_Fabrication_FDM_FFF
https://howtomechatronics.com/tutorials/arduino/arduino-dc-motor-control-tutorial-l298n-pwm-h-bridge/
https://www.arduino.cc/reference/de/language/functions/advanced-io/tone/
https://www.arduino.cc/reference/de/language/structure/control-structure/switchcase/
https://forum.arduino.cc/index.php?topic=373897.0 (Eagle Library)
https://optoelectronics.liteon.com/upload/download/ds-55-96-0005/lth-301-32ds.pdf
https://asset.conrad.com/media10/add/160267/c1/-/en/001666921DS01/datenblatt-1666921-barthelme-led-sortiment-rot-rund-5-mm-800-mcd-35-20-ma-2-v.pdf
https://asset.conrad.com/media10/add/160267/c1/-/en/001589396DS00/datenblatt-1589480-tru-components-tc-mt250asw-drucktaster-tastend-1-st.pdf
https://asset.conrad.com/media10/add/160267/c1/-/gl/000710803DS01/datenblatt-710803-visaton-2816-miniatur-lautsprecher-geraeusch-entwicklung-71-db-0500-w-1-st.pdf
https://components101.com/modules/l293n-motor-driver-module

12 Tabellenverzeichnis

13 Abbildungsverzeichnis

14 Anhang

Abnahmeprotokoll (4 Seiten)

Programm der Motorsteuerung (2 Seiten)
Programm des Anzeigesystems (2 Seiten)

Schaltplan der Motorsteuerung (1 Seite)
Schaltplan des Anzeigesystems (1 Seite)

Datenblatt der Lichtschranke (6 Seiten)
Datenblatt des Motorcontrollers (7 Seiten)
Datenblatt des Drucktasters (1 Seite)
Datenblatt der Leuchtdiode (1 Seite)
Datenblatt des Lautsprechers (1 Seite)

Sämtliche Programmcodes, Schaltpläne, STL-Dateien der 3D-Druckteile und Datenblätter sind auch auf https://github.com/DominikEminger/BackflipBasti/ zu finden, oder durch Scannen dieses QR-Codes: