

FACHBERICHT

FS18 - PRO4E - TEAM 5

8. Juni 2018

AUFTAGGEBER:	H. GYSIN
	J. KALBERMATTER
BETREUER:	M. MEIER
	A. GERTISER
	R. DUBACH
	B. DOMENGHINO
	P. SCHLEUNIGER
PROJEKTLEITUNG:	SIMON ZOLLER
TEAMMITGLIEDER:	SEVERIN HUNZIKER
	MISCHA KNUPFER
	LUKAS LOOSLI
	JOSHA GIAMBONINI
	ELIAS VON DÄNIKEN
	GIANLUCA PICCIOLA
STUDIENGANG:	ELEKTRO- UND INFORMATIONSTECHNIK

Abstract

Aimlessly wandering around a museum because of too little knowledge is one of the main negative reactions to a museum visit. That's why no one can remember an individual work of art. To counteract this, an audio guide called Dōjō was designed to provide visitors with key information about a work of art so they can enjoy their visit more. This project should design the circuits for this Dōjō, which electronically recognises the artwork the visitor is standing in front of. Audio information stored on an internal SD-Card over a Class-D Amplifier is provided through an integrated bone sound sensor produced by adafruit. Buttons for audio control have also been integrated. In addition, the Dōjō allows the visitor to „like“ an artwork by simply pressing a button. Information about these „liked“ works are made available to the visitors either by email or a hardcopy is provided at the end of their visit. Each work of art has a Bluetooth Low Energy (BLE) Beacon that continuously sends its ID to the Dōjō. The Dōjō's internal microcontroller (nrf52832) then scans for this BLE-Signals and plays the file that belongs to the ID with the strongest signal. The Dōjō can identify beacons at XX meters. The inductive rechargeable battery has a capacity of 800mAh and can provide energy for about 6 hours when the Dojo is playing audiofiles and be recharged in normal charging mode within 13 hours. To protect the battery from damage, a deep discharge protection turns off the Dōjō when the battery is under 3V of its nominal voltage.

Key Words: audio guide, Bluetooth low energy, inductive charge, bone sound sensor

Inhaltsverzeichnis

1 Einleitung	4
2 Gesamtkonzept	5
2.1 Funktionsweise	5
2.2 Anwendung	6
3 Hardware	9
3.1 Induktive Ladung	9
3.2 Li-Ion-Batterie	12
3.3 Ladestation	13
3.4 Mikrocontroller	14
3.5 Filterstufe	14
3.6 Verstärkerstufe	16
3.7 Knochenschallaktor	16
4 Software	18
4.1 State-Machine	20
4.2 Nordic Software Development Kit	27
4.3 Bluetooth	28
4.4 SD-Karte	32
4.5 Audiowiedergabe durch PWM	34
4.6 Lizenzen	36
5 Validierung	38
5.1 Testkonzept Hardware	38
5.2 Testkonzept Software	44
6 Schlusswort	48
7 Ehrlichkeitserklärung	50
8 Literaturverzeichnis	51
9 Abbildungsverzeichnis	52
10 Tabellenverzeichnis	53
A Anhang	54
A.1 Messresultate	54

1 Einleitung

Museen bieten die Möglichkeit unterschiedlichste Ausstellungsobjekte unter einem Dach zu betrachten. Die Art der ausgestellten Kunst ist hier von Austellung zu Austellung unterschiedlich, was jedoch bleibt ist die kreative Wahrnehmung der Besucher. Wohl nirgends kann man so gut in seinen eigenen Gedanken versinken und sich Gedanken über ein Ausstellungsobjekt machen wie in einem Museum. Um Besucher anzulocken, sind Museen auf Innovation angewiesen, welche zum einen die Übergabe von Informationen möglichst benutzerfreundlich gestaltet, aber auch ein angenehmes Ambiente schafft. Hierbei kommt auch vermehrt der Einsatz von Smartphones zum Zuge, wobei die Problematik darin besteht, dass man der Aussenwelt gefährlich nahe kommt und dadurch abgelenkt wird. Um diesem Problem entgegenzuwirken, wird ein von Frau J. Kalbermatter designter Audioguide namens Dōjō entwickelt, welcher Kunstobjekte drahtlos erkennen und darüber gespeicherte Informationen via Körperschalltechnik zum Museumsbesucher bringen kann. Durch eine Vielzahl von weiteren Funktionen ist ein neuer Informationsaustausch an den Museumsbesucher möglich, und Ablenkungen zur Aussenwelt sind trotz modernster Technik in weiter Ferne.

Ziel im Projekt 4, Studiengang Elektro- und Informationstechnik an der Fachhochschule Nordwestschweiz, war es das funktionelle Konzept von Frau Kalbermatter durch die Verwendung von elektrotechnischen Bauteilen zu realisieren. Dazu wurde die drahtlose Erkennung der Kunstobjekte mittels Bluetooth Low Energy (BLE) Beacons erreicht. Genannte Beacons müssen in unmittelbare Nähe der Kunstobjekte angebracht sein. Die Informationen zu den Kunstobjekten wurden als Audiofiles auf einer herausnehmbaren SD-Karte gespeichert und werden zum Abspielen via PWM-Ausgang des Mikrocontrollers (nRF52832) über einen Klasse D Verstärker auf den Knochenschallaktor gegeben. Tasten für die Wiedergabekontrolle (Start, Stopp, Lauter, Leiser) wurden implementiert sowie die erwähnte „Like“-Taste. Außerdem verfügt der Dōjō über einen Li-Ionen-Akku, welcher induktiv geladen wird. Damit der Dōjō gänzlich drahtlos bleibt, erfolgen Datendownload und Konfiguration ebenso über Bluetooth. Der integrierte Mikrocontroller beinhaltet die Software und übernimmt somit die Erkennung, Ansteuerung und Koordination der Hardware.

Es wurde ein Prototyp der Elektronik realisiert. Er kann zu vier verschiedenen Sprachen XXX Stunden Audioausgabe speichern. Die Ansteuerung der Audiofiles erfolgt über Bluetooth-Beacons, welche bis zu einer Distanz von XXX m erkannt werden. Die eingebaute «Like»-Taste ermöglicht, favorisierte Kunstobjekte zu vermerken und die dazugehörigen Informationen am Ende des Besuches digital oder in Form einer Broschüre beim Ausgang als Erinnerung mitzunehmen. Außerdem besitzt der Dōjō einen integrierten Akku mit einer Kapazität von 800 mAh, welcher bei pausenloser Audioausgabe genug Energie für rund 6 Stunden liefert. Die Induktionsladung lädt den Akku im Normalbetrieb zu 100% innerhalb 13 Stunden. Außerdem sorgt ein Tiefenentladungsschutz der Batterie dafür, dass der Dōjō bei einer Betriebsnennspannung von unter 3V ausgeschaltet wird.

Der nachfolgende Bericht umfasst drei Hauptbereiche. Der erste Bereich (Kapitel 2) umfasst das Gesamtkonzept, welcher die gesamte Anwendung auslegt. Die nachfolgenden zwei Hauptbereiche sind in Hardware (Kapitel 3) und Software (Kapitel 4) gegliedert. Die Hardware teilt sich wiederum in die Themengebiete Energieübertragung (Kapitel 3.1), Energiespeicherung (Kapitel 3.2) und Audioausgabe über den Knochenschallaktor (Kapitel 3.5) auf. Die Software beinhaltet die Unterbereiche der State Machine (Kapitel 4.1), Bluetooth (Kapitel 4.3), sowie die gesamte Programmstruktur der SD-Karte (Kapitel 4.4) und Audioausgabe über PWM (Kapitel 4.5). In Kapitel 5 befindet sich die Validierung.

2 Gesamtkonzept

Bevor auf die einzelnen Komponenten des Dōjōs eingegangen werden kann, ist ein Einblick in das Gesamtkonzept notwendig. Hauptbestandteil werden in diesem Kapitel der Aufbau, die Funktionsweise des Gerätes, wie auch die Anwendung im Museum sein. Beim Aufbau ist die Kommunikation der einzelnen Komponenten untereinander ersichtlich. Anschliessend werden die jeweiligen Funktionsweisen erklärt. Der letzte Teil gibt einen Einblick in die Anwendung vor und während dem Museumsbesuch und welche technischen Möglichkeiten dem Besucher zur Verfügung stehen.

2.1 Funktionsweise

Der Dōjō ist eine Art Audioguide, welcher für Museen designt wurde und diverse Funktionen beinhaltet. Abbildung 1 zeigt den von der Auftraggeberin designten Prototypen. Einer der grössten Unterschiede zu einem herkömmlichen Audioguide ist die Sprachausgabe mittels einem Knochenschallaktor und nicht wie gewöhnlich mit einem Lautsprecher. Eine weitere Eigenheit ist der integrierte „Like“-Button, mit dem man die entsprechenden Ausstellungsstücke „liken“ kann. Dadurch lässt sich für jeden Besucher individuell eine persönliche Kunstobjektliste zusammenstellen, die nur noch die entsprechenden Objekte mit einem „Like“ enthält. Das ermöglicht es, am Ende des Besuchs jeder Person eine Liste mit den jeweiligen Informationen zum Ausstellungsobjekt auszuhändigen. Ansonsten verfügt der Dōjō über die gleichen Funktionen, die man von einem üblichen Audioguide erwarten würde, wie z.B. der Audiowiedergabe, dem Haltemodus oder der Lautstärkeregelung.



Abbildung 1: Dōjō als Modell

Das Herzstück des Dōjōs ist ein zentraler NRF52-Mikrocontroller von Nordic Semiconductor mit integriertem und low-Energy fähigem Bluetooth-Stack. Die verwendeten Daten werden auf einer SD-Karte gespeichert. Bei Bedarf werden die Audiodateien an den Verstärker weitergegeben und schliesslich über den Knochenschallaktor wiedergegeben. Die Energieversorgung wird mit einer Li-Ion-Batterie sichergestellt, welche über ein induktives Ladesystem versorgt wird. Abbildung 2 zeigt das technische Lösungskonzept im Überblick.

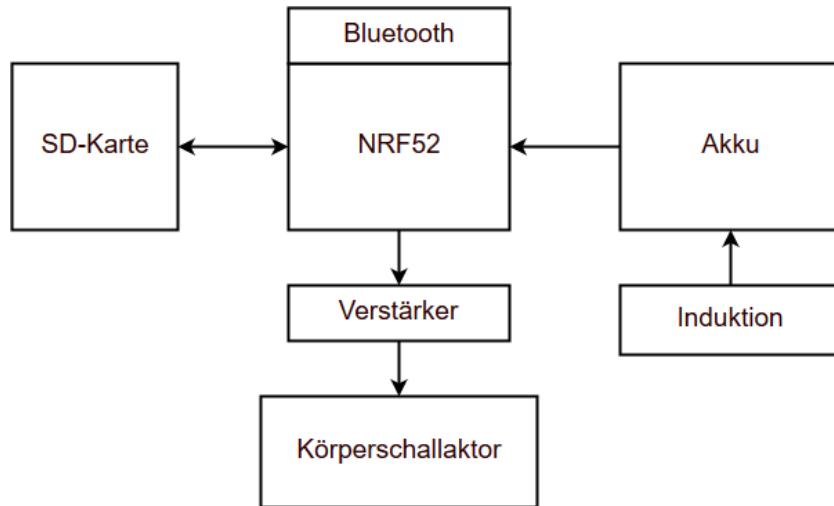


Abbildung 2: Teilsysteme des Dōjōs

Die Funktionen des Dōjōs werden nachfolgend in zwei Bereiche unterteilt. Zuerst wird die Funktionalität aus Sicht des Users beschrieben. Anschliessend werden die relevanten Funktionen aus Sicht der Betreiber erläutert.

Der **User** geht mit dem Dōjō durch das Museum. Sobald die Bluetooth Beacons genügend nahe sind, erhält der User durch eine aufleuchtende LED ein Signal. Jetzt kann er entscheiden, ob er sich das zugehörige Audio-File anhören will. Trifft dies zu, wird die Audiodatei durch das Betätigen des Play-Buttons abgespielt. Die Lautstärke kann über die Volume-Buttons justiert werden. Falls das Ausstellungsstück dem User gefallen hat, kann die „Like“-Taste gedrückt werden. Dadurch wird das Ausstellungsstück auf einer Liste in der internen SD-Karte gespeichert. Am Ende des Museumsbesuches kann diese Liste ausgewertet werden, wobei die Verwendung dieser Daten nicht Bestandteil der Projektarbeit ist und aus diesem Grund nicht weiterverfolgt wird.

Der **Betreiber** hat die Aufgabe den Dōjō zu konfigurieren. Dies erfolgt über die dafür vorgesehene SD-Karte, welche mit dem Computer beschrieben wird. Anschliessend kann die Speicherkarte in den Dōjō eingeführt werden. Die Energieversorgung erfolgt über eine induktive Ladestation. Die nächsten beiden Funktionen sind Wunschziele, die vor allem mit Rücksicht auf die Laufzeit realisiert werden. Den Bluetooth-Receiver könnte man kurzzeitig auf ein Bluetooth Beacon umschalten. Der Betreiber müsste nur noch einen Receiver pro Raum installieren. Damit könnte man die gewünschte Lokalisierung der Besucher umsetzen. Das zweite wäre die Möglichkeit per Bluetooth einzelne Audiofiles auf den Dōjō zu übertragen, um im Falle einer Änderung der Ausstellung, die Liste anzupassen und damit aktuell halten zu können.

2.2 Anwendung

Nachdem die Unterteilung zwischen Betreiber und User gemacht wurde, kann nun die Betriebsebene verdeutlicht werden. Um einen lückenlosen Betrieb zu gewährleisten, ist ein Konzept für den Ablauf von Vorteil. Nachfolgend werden die Hauptschritte erklärt. Abbildung 3 dient zur Übersicht und zeigt einen möglichen Anwendungsablauf.

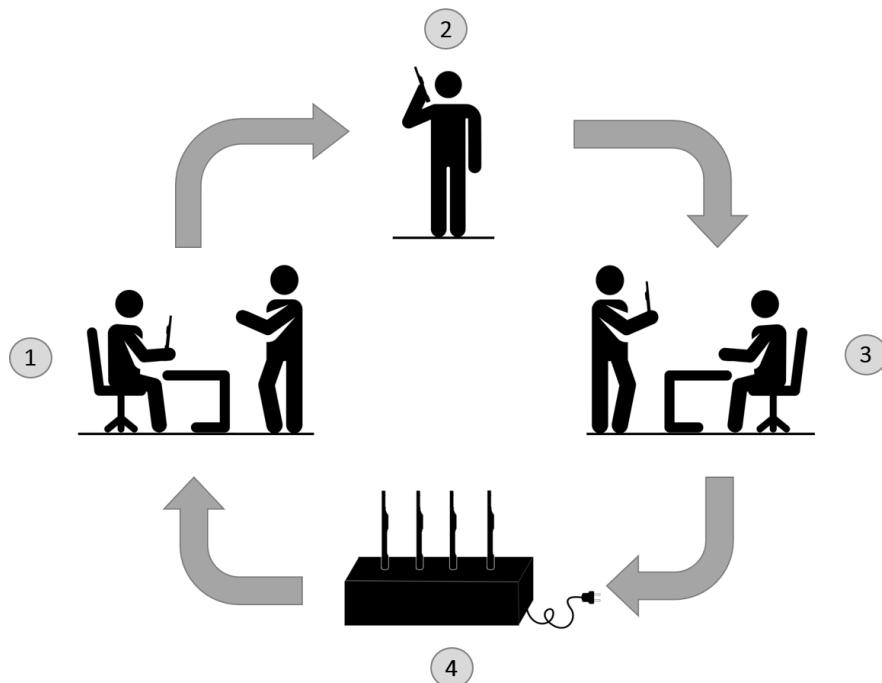


Abbildung 3: Anwendungsablauf im Museum

Schritt 1 beinhaltet die Ausgabe des Dōjōs am Empfang. Es werden jeweils die Geräte mit dem höchsten Ladestatus abgegeben. Ein lückenloser Betrieb wird erreicht, wenn die Stückzahl der Audio-Guides in etwa der Anzahl Besucher pro Tag entspricht. Die Zutrittsberechtigung wird gemäss dem Wunsch des Besuchers festgelegt. Anschliessend wird die Sprache durch den Besucher selbst gewählt. Hierbei stehen ihm vier Bluetooth-Beacons zur Verfügung, zu welchen er sein Dōjō hinhalten kann. Die gewünschte Sprache ist hierbei durch die Landesflagge gekennzeichnet. Abbildung 4 zeigt das Prinzip. Damit eine der vier Sprachen auf dem Dōjō aktiviert wird, muss das Gerät an den richtigen Beacon gehalten werden und gleichzeitig die Play-Taste gedrückt werden. Wurde die Sprache erfolgreich ausgewählt, wird ein kurzes Audio-Sample zur Überprüfung der gewählten Sprache abgespielt. Sobald die gewünschte Sprache geladen und getestet wurde, kann der Museumsbesuch gestartet werden.

In **Schritt 2** befindet sich der Besucher auf dem Rundgang mit dem Dōjō. Dabei hat er die Möglichkeit, während dem Rundgang Bilder zu „liken“ und sich die zugehörigen Audio-Files der Kunstobjekte anzuhören.

Die Abgabe der Gerätes erfolgt in **Schritt 3**. Hier hat der Besucher die Möglichkeit die Kunstobjekte, welche mit einem „Like“ versehen wurden, als Broschüre oder per Mail zu erhalten. Der entgegengenommene Dōjō kann nun für den nächsten Besucher gereinigt werden.

In **Schritt 4** werden alle benötigten Informationen extrahiert. Danach wird das Gerät mit der induktiven Ladestation geladen. Für weitere Informationen zur Ladestation wird auf das Kapitel 3.1 verwiesen. Nun kann der Zyklus wieder von vorne beginnen.

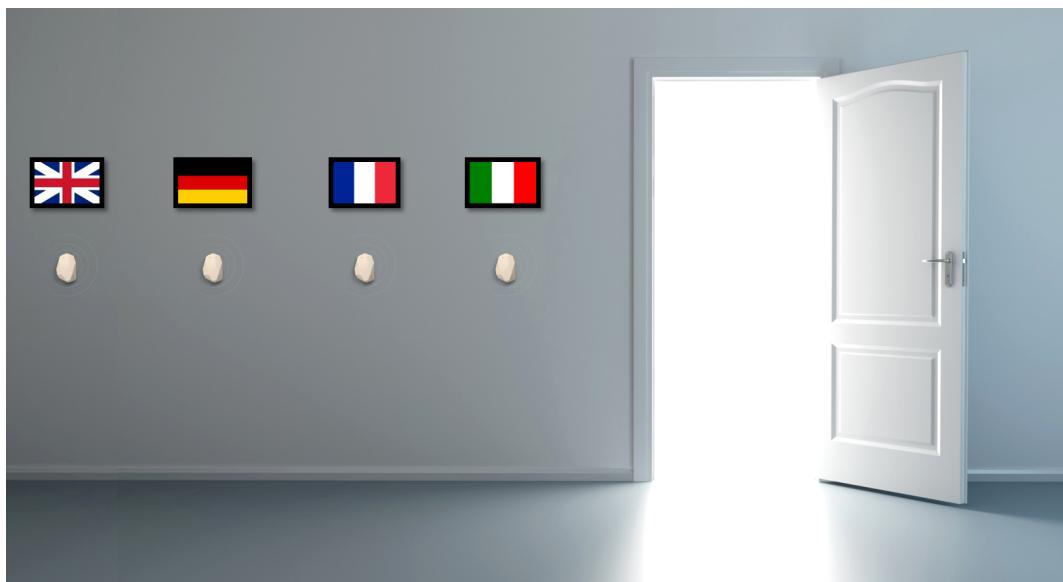


Abbildung 4: Sprachauswahl mittels Bluetooth-Beacon

3 Hardware

Nachdem die ganze Struktur und der Funktionsablauf der Anwendung detailliert erklärt wurde, soll nun an dieser Stelle der Dōjō in seine Komponenten zerlegt und genauer erläutert werden. Die einzelnen Systeme sind logisch geordnet und werden in den nachfolgenden Kapiteln erklärt. Zuerst wird die induktive Energieübertragung, die Energiespeicherung und die dazu notwendige Regelung erläutert. Anschliessend werden die Komponenten Mikrocontroller, Filterstufe, Verstärkerstufe und Knochenschallaktor beschrieben.

3.1 Induktive Ladung

Der Dōjō wird mit Hilfe des Induktionsprinzips geladen. Der Ladeprozess wird gestartet, sobald der Dōjō auf die Ladestation gestellt wird. Beim Induktionsprinzip wird die Energie mithilfe von Spulen über eine kurze Distanz zwischen zwei Schaltungen transportiert. Die erste Schaltung wird Transceiver genannt und sendet die Energie. Sie besteht aus einem Pulsgenerator, mit welchem das LC-Glied gepulst wird. Sie macht den Hauptanteil einer solchen induktiven Ladeschaltung aus. Die zweite Schaltung wird Receiver genannt und empfängt die Energie. Sie besteht ebenfalls aus einem LC-Glied, hat jedoch zusätzlich einen Gleichrichter. Nachfolgend werden die Transceiver- und Receiverschaltung beschrieben, welche sich auf Abbildung 5 beziehen.

Transceiver

Als Pulsgenerator der Transceiver-Schaltung wird die Timer-Schaltung des NE555 verwendet, welcher mit 10V gespiesen wird. Diese Timer-Schaltung bietet den Vorteil, dass sie ohne grossen Aufwand zu erstellen und einzustellen ist. Diese Eigenschaften haben diesen Timer-Baustein zum Standard IC für Zeitschaltungen gemacht. Der NE555 enthält eine monolithisch integrierte Zeitgeberschaltung, die sich aufgrund ihrer Eigenschaften als Taktgeber, Oszillator und für Zeitverzögerungen verwenden lässt. Der NE555 ist so universell einsetzbar, dass er als wichtigster integrierter Schaltkreis gilt. Das Funktionsprinzip beruht auf der Programmierung durch externe Komponenten. Das heisst, dass mithilfe von verschiedenen Verhältnissen der extern angeschlossenen Komponenten (bestehend aus Widerständen und Kondensatoren), die Pulsdauer, Frequenz und der Duty cycles eingestellt werden kann. Das dabei entstehende Pulssignal wird benutzt, um das LC-Glied zu pulsieren, welches aus der Parallelschaltung unserer $14.9\mu H$ Transceiverspule ($U\$1$) und $220nF$ Kondensators C_3 besteht (Abbildung 5). Um dies zu ermöglichen wird dieses Glied an die Versorgungsspannung gehängt und in Serie dazu der Kollektor Anschluss eines 2N3055 Leistungs NPN-Transistor. An dessen Emitter wird nun ein Niederohmiger Widerstand (R_4) auf GND gehängt. Dieser dient als Strombegrenzung, da die Spannung welche über ihm Abfällt von einem zweiten Transistor (2N2222) überwacht wird. Dieser 2N2222 Transistor wird zwischen dem Pulssignal, welches den 2N3055 Leistungs-Transistor steuert, und GND gehängt. Wird nun der Strom und somit auch die Spannung über dem Widerstand R_4 zu hoch, so schliesst der 2N2222 Transistor das Pulssignal kurz. Der 2N3055 Leistungstransistor wird nicht mehr sauber durch gesteuert und der Strom wird begrenzt. Die Strombegrenzung ist von den verwendeten Komponenten des LC-Gliedes abhängig. Es ist darauf zu achten, welches der Beiden Elemente (LC-Glied/Leistungstransistor) weniger Strom verträgt. In diesem Fall ist dies die Spule, welche aufgrund ihrer geringen Baugrösse nur einen Strom von $0.6A$ verträgt. Würden grössere Spulen verwendet, so gilt spätestens bei einem Strom von $15A$ zu begrenzen, da dies die Belastungsgrenze für den 2N3055 Leistungstransistor ist.

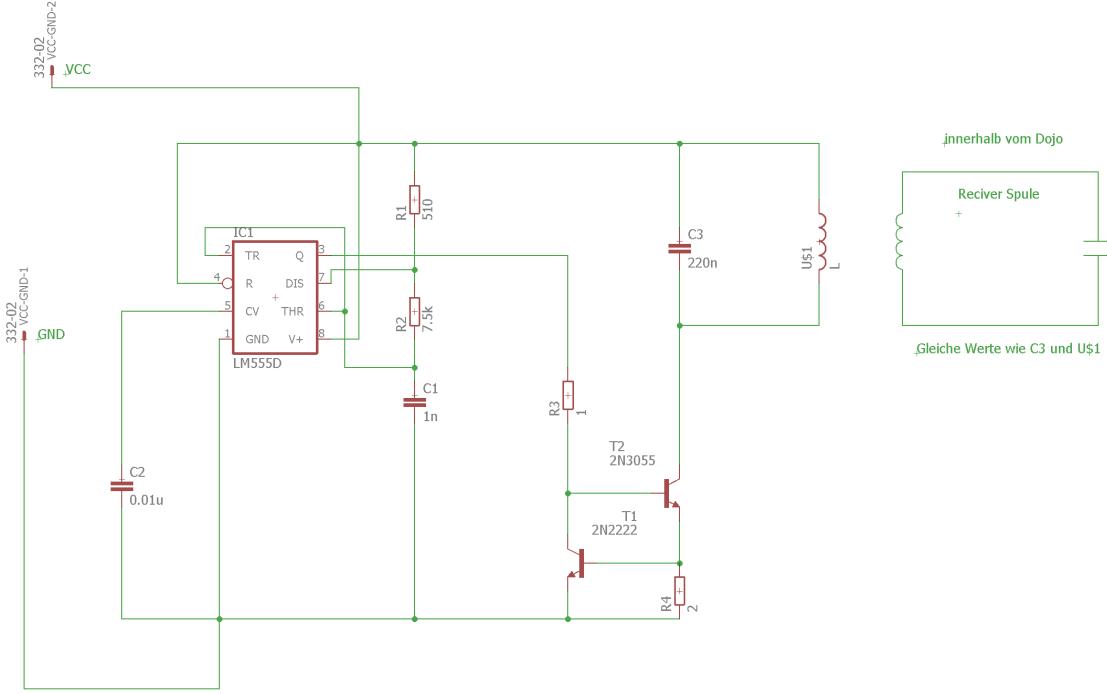


Abbildung 5: Verwendete Timerschaltung als Pulsquelle

Das Pulssignal selber muss verschiedene Kriterien erfüllen. Zum einen sollte der Duty Cycle so nahe wie möglich an 50% sein. Um dies zu erreichen muss $R_2 >> R_1$ gelten. Das andere Kriterium ist die Erreichung der Resonanzfrequenz (f_{res}) des LC-Gliedes. Diese wird wie folgt aus den Komponenten des LC-Gliedes berechnet.

$$f_{res} = \frac{\sqrt{L \cdot C}}{2 \cdot \pi} \quad (3.1)$$

Die ideale Frequenz würde 93kHz betragen. Jedoch musste diese auf 79kHz angepasst werden da die nachfolgende Ladeschaltung nicht ideal dimensioniert wurde. Genaueres dazu ist in der Validierung der Hardware zu finden. Um das Pulssignal theoretisch optimal einzustellen, können folgende Richtlinien betrachtet werden:

- **C** beeinflusst die Zeiten (Frequenz/High-Time/Low-Time)
- **R₁** beeinflusst die High-Time, lässt jedoch die Low-Time unverändert.
- **R₂** beeinflusst die High- und Low-Time und beeinflusst somit den Duty Cycle.

Die verwendeten Komponenten C , R_1 und R_2 , welche in der Abbildung 5 rechts vom Timer Baustein ($LM555D$) zu sehen sind, wurden durch nachfolgende Formeln 3.2 bis 3.7 berechnet.

Zuerst wollen wir unsere ideale Taktfrequenz berechnen. Dabei werden wir zuerst einen Kondensator auswählen. Da dieser die Zeit massgebend beeinflusst, ist darauf zu achten, dass dieser eher klein zu wählen ist, da sonst die Widerstandswerte ins unermässliche gehen.

$$f = \frac{1}{T} = \frac{1.44}{((R_1 + 2 \cdot R_2) \cdot C)} \quad (3.2)$$

Nun bleiben noch die Widerstände zu bestimmen. Wir wollen einen Duty Cycle von ca 50%. Das heisst, dass Low- und High-Time beinahe identisch sind. Daraus lässt sich schliessen, dass

R_1 vernachlässigbar klein ist. Dieser könnte allenfalls sogar als 0 angenommen werden, jedoch lässt sich so nicht jede beliebige Frequenz einstellen mit den vorhandenen Widerstandswerten für R_2 . Dieser müsste aus mehreren Widerständen zusammengesetzt werden. Daher entschieden wir uns dafür einen kleinen Wert für R_1 anzunehmen, was sich beim Duty-Cycle kaum bemerkbar machte.

$$LowTime = 0.693 \cdot (R_2) \cdot C \quad (3.3)$$

$$HighTime = 0.693 \cdot (R_1 + R_2) \cdot C \quad (3.4)$$

$$D = DutyCycle = \frac{R_1 + R_2}{R_1 + 2 \cdot R_2} \quad (3.5)$$

Somit wären alle Komponenten bestummen. Möchte man eine andere Vorgehensweise wählen, so stünden noch die folgenden Formeln 3.6 und 3.7 zur Verfügung.

$$R_1 = 1.44 \cdot \left(\frac{2 \cdot D - 1}{f \cdot C} \right) \quad (3.6)$$

$$R_2 = 1.44 \cdot \left(\frac{1 - D}{f \cdot C} \right) \quad (3.7)$$

Für die Berechnung der effektiven Werte, müssen die ersten Werte angenommen werden. Deshalb lohnt es sich einen Wert für den Kondensator C anzunehmen, da dessen verfügbare Werte durch die E-Reihen begrenzt sind. Es ist zu erwähnen, dass es sich bei den bereits eingefügten Zahlenwerten um Konstanten handelt, welche in jedem handelsüblichen Datenblatt eines 555 Timer-Bausteins zu finden sind. Unsere Berechnungen ergaben für die angepasste Tranceiver Schaltung mit einer verringerten Taktfrequenz von $79.12kHz$ folgende Werte:

$$\begin{aligned} C_1 &= 1nF \\ R_1 &= 200\Omega \\ R_2 &= 9k\Omega \end{aligned}$$

Diese Bauteile sind in Abbildung 5 zu sehen, wobei die neu berechneten Werte der Widerstände nicht mehr mit den ursprünglichen Werten im Schema korrelieren. Nach dem die Werte berechnet wurden, können die einzelnen Komponenten ausgewählt und implementiert werden. Dabei gilt es zu beachten, dass minimale Abweichungen bereits zu einer Änderung der Frequenz, Pulsdauer und Duty-Cycle des Pulssignales führen.

Receiver

Der Receiver besteht primär aus einem LC-Glied und einem Gleichrichter. Speziell ist, dass bei der Tranceiverschaltung das selbe LC-Glied verwendet wurde wie bei der Receiverschaltung. (Wie in Abbildung 5 rechts oben zu sehen besteht das LC-Glied aus der Flachspule L mit einem Wert von $14.9uH$ und einem Koppelkondensator C_3 mit $220nF$). Dies wurde aus dem Grund gewählt, weil das Energiefeld bei einem identischen LC-Glieder Paar am effektivsten ausgenutzt werden kann. Das L-Glied (Flachspule) lässt sich mit einer Dimension von $\varnothing15mm$ Durchmesser und $0.2mm$ Höhe gut im inneren des Dōjō's platzieren. Die Positionierung findet am Boden statt, da dieser eben ist und somit als Standfläche für den Dōjō genutzt wird. Somit kann dieser schlussendlich in der Ladestation platziert werden, ohne davon zu rollen. Die hochfrequente

Wechselspannung welche nun am Dōjō anliegt, muss für die Speisung der Batterie noch gleichgerichtet werden. Hierbei können beliebige Dioden verwendet und zu einem Brückengleichrichter verschaltet werden. Wir entschieden uns für Schotky-Dioden, da diese mit einer Durchlassspannung von $0.2V$ die übertragene Spannung nur leicht abschwächen. Anschliessend wird die Ladeschaltung gespiesen, welche den gesamten Ladeprozess des Akkus übernimmt. Einen Einblick in diesen Ladeprozess gibt nachfolgendes Kapitel.

3.2 Li-Ion-Batterie

Da nun die Energieübertragung verdeutlicht wurde, kann an dieser Stelle die Batterie erklärt werden. Die gesamte Energiespeicherung wird mit einem Lithium-Ionen-Akkumulator des Typs Emmerich LI14500 realisiert. Dieser weist bei einer Nominalspannung von $3.7V$ eine Kapazität von $800mAh$ auf und verfügt über interne Schutzeinrichtungen. Um eine Abschätzung über die Betriebszeit des Dōjōs zu erhalten, sind Faktoren wie der maximale Verbrauch, Nominalspannung und Kapazität notwendig. Die maximale Leistung des Dōjōs lässt sich durch die Leistung des Verstärkers bzw. Knochenschallgebers (P_{Kn}) und des Mikrocontrollers (P_{MC}) beschreiben. Alle anderen Komponenten können aufgrund des geringen Betriebsstrom mit einem Sicherheitsfaktor (P_{zus}) von $0.1W$ dazu gerechnet werden. Die Leistung des Verstärkers weist eine maximale RMS-Leistung von $471.9mW$ auf. Gerechnet wird mit 80% des RMS-Wertes, da der Knochenschallgeber nicht permanent beansprucht wird. Die Leistungsaufnahme des Mikrocontrollers beträgt $27.36mW$ und wurde durch Messungen ermittelt. Die Gesamtleistung wird wie folgt berechnet:

$$P_{max} = (0.8 \cdot P_{Kn}) + P_{MC} + P_{zus} = (0.8 \cdot 0.472W) + (0.02736W) + 0.1W = 0.506W \quad (3.8)$$

Die Gesamtleistung beträgt somit rund $0.506W$. Nun kann basierend auf der Gesamtleistung die Betriebszeit t_{max} berechnet werden:

$$t_{max} = \frac{W \cdot U}{P_{tot}} = \frac{800mAh \cdot 3.7V}{506mW} = 5.85h \approx 5h\ 51min \quad (3.9)$$

Es ist somit ersichtlich, dass bei permanentem Einsatz eine maximale Betriebszeit von knapp 6 Stunden erzielt werden kann. Allerdings kann ein lückenloser Betrieb durch einen geschickten Ladeprozess erzielt werden.

Schutzeinrichtungen

Um den verwendeten Akkumulator zu schützen, sind diverse Schutzeinrichtung notwendig. Damit die Ladespannung und der maximale Ladestrom nicht überschritten werden, muss der Ladevorgang überwacht werden. Für die Laderegelung wurde ein Lade-IC vom Hersteller Microchip des Typs MCP73831 verwendet. Dieser übernimmt die gesamte Spannungs- und Stromregelung während des Ladeprozesses und kann parallel dazu eine LED zur Ladesignalisation ansteuern. Der Ladezyklus ist in der Abbildung 6 ersichtlich. Hierbei wurde der Akku im Schnelllademodus mit einem maximalen Strom von $400mA$ geladen. Dieser Strom ergibt sich aus dem Datenblatt der Batterie [?], wobei sowohl der Entladestrom, als auch der Ladestrom $0.5 \cdot Q$ pro Stunde beträgt. Q entspricht der Batteriekapazität. Somit lässt sich der Ladestrom I_c folgendermassen berechnen:

$$I_c = 0.5 \cdot Q = 0.5 \frac{1}{h} \cdot 800mAh = 400mA \quad (3.10)$$

Betrachtet man die Abbildung 6, so wird ersichtlich, dass die Spannung rund $2.5h$ geregelt wird. Sobald der Spannungswert $4.2V$ erreicht hat, beginnt der Lade-IC mit der Stromregelung. Für diesen Prozess wurden beim Versuch noch einmal rund 30 Minuten benötigt, wodurch die letzten rund 20% der Batteriekapazität geladen werden konnten.

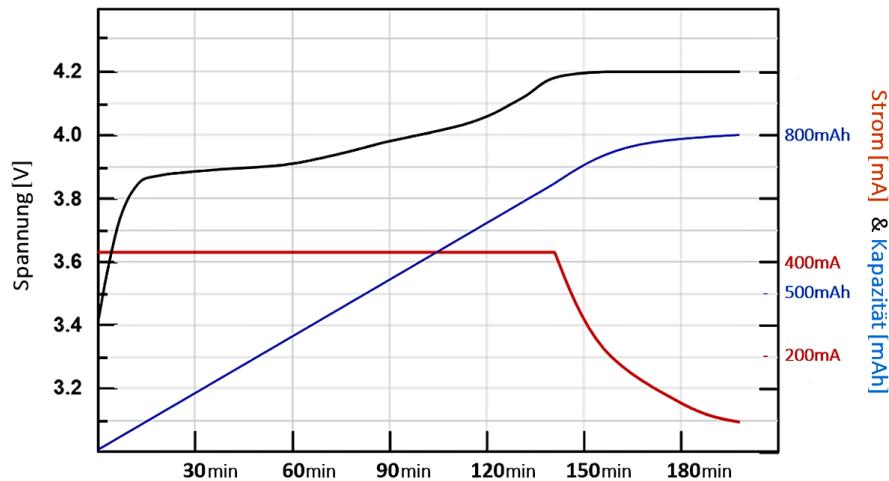


Abbildung 6: Laderegelung Li-Ion Akku

Ein weiterer Schutzbaustein ist das Protection Circuit Module (PCM), das einerseits einen Überladeschutz von $4.25V \pm 0.025V$, aber auch einen Tiefentladungsschutz von $2.5V \pm 0.063V$ regelt. Weiter ist der Akku gegen Überströme ab einer Höhe von $4.8A$ geschützt.

3.3 Ladestation

Nachdem die gesamte induktive Ladeschaltung und Energiespeicherung beschrieben wurde, folgt nun die Beschreibung der Ladestation. Hierfür wurde ein Prototyp erstellt, welcher die gesamte primäre induktive Ladeschaltung beinhaltet. Für den Prototypen wurde eine .stl Datei erstellt, welche mit einem 3D-Drucker gedruckt wurde. Wichtig ist hierbei zu erwähnen, dass es sich nur um einen Prototypen handelt und es bei einer Weiterentwicklung noch Anpassungen geben kann. Da die Ladestation für Versuchszwecke erstellt wurde, ist das Design so gewählt, dass nur ein Dōjō geladen werden kann. Dies könnte in einem weiteren Schritt auf mehrere Ladeaussparungen erweitert werden, wodurch mehrere Dōjōs gleichzeitig pro Ladestation geladen werden können. Abbildung 7 zeigt die Ladestation aus verschiedenen Perspektiven.

Die Öffnung dient zum einen als Standhalterung und zum anderen als korrekte Positionierung für die induktive Ladeschaltung. Die richtige Positionierung ist eines der wichtigsten Kriterien für einen optimalen Ladezyklus, da die Transceiver- und Receiverspule direkt übereinanderliegend den besten Wirkungsgrad erzielen. Der Hohlraum im Inneren der Ladestation dient zur Platzierung des Primärkreises und hat die Abmessung $(80 \times 70.7 \times 30)\text{mm}$. Für den Prototypen wurde eine Lochrasterplatine mit allen nötigen Komponenten gefertigt, welche genau in diese Aussparung passt. Weiter sind runde Löcher ($5\text{mm } \varnothing$) ersichtlich, welche in die Kammer des Schriftzuges führen. Sie sind für LEDs vorgesehen, welche den Dōjō-Schriftzug bei angeschlossener Versorgungsspannung zum Leuchten bringen.

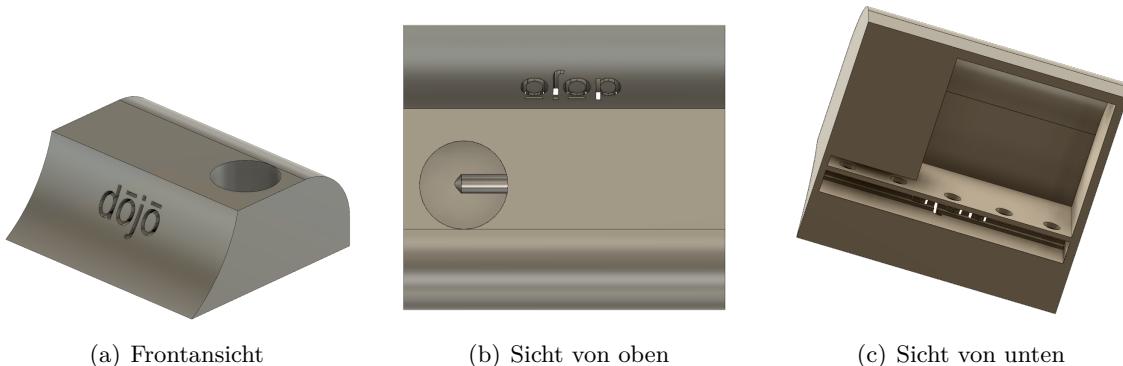


Abbildung 7: Prototyp der Ladestation aus verschiedenen Perspektiven

3.4 Mikrocontroller

Für die Steuerung der Anwendung wird nun noch ein zentraler Controller benötigt. Die Wahl fiel auf den Mikrocontroller nRF52832 von Nordic Semiconductors. Seine hohe Performance ermöglicht es ein System aufzubauen, welches nur eine zentrale Schnittstelle beinhaltet. Abbildung 8 zeigt den Controller als Einzelbauteil und als Entwicklungskit.

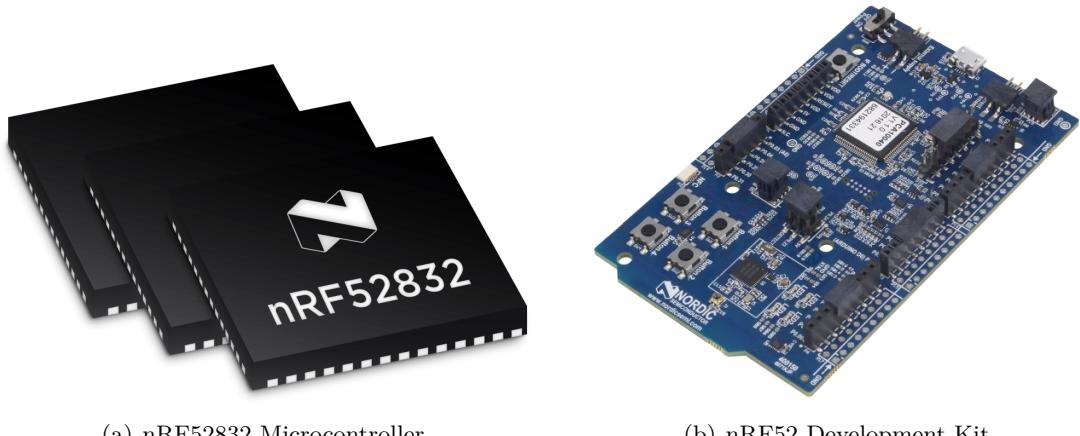


Abbildung 8: Mikrocontroller als Einzelement und als Entwicklungsanwendung

Der nRF52832 weist eine Versorgungsspannung zwischen $1.7V$ und $3.6V$ und einen Betriebssstrom von $5.4mA$ auf. Die niedrigen Versorgungswerte ermöglichen einen dauerhaften Betrieb durch die integrierte Batterie. Die Speicherkapazität ist durch $512kB$ flash/ $64kB$ RAM Speicher gegeben. Für die Entwicklung des Prototypen wurde das Entwicklungsboard verwendet. Dies hat verschiedene Vorteile, wie z.B. die integrierte Bluetooth-Antenne, die einfache Verwendung von sogenannten Shields und das einfache Handling für Messungen und provisorische Verbindungen.

3.5 Filterstufe

Nachdem die Steuerungseinheit gezeigt wurde, kann nun die Ausgangsstufe erläutert werden. Bevor das Signal verstärkt wird, durchläuft es eine Filterstufe, bestehend aus einer Spule L und einem Kondensator C . Das Filter hat die Aufgabe das PWM-Signal des Mikrocontrollers zu

filtern und den DC-Anteil zu entfernen. Zusätzlich wird der Mikrocontroller noch vom System entkoppelt. Das wird mit $100\mu F$ -Elektrolytkondensatoren an jedem PWM-Ausgang erzielt. Sie werden jeweils seriell geschaltet. Die Dimensionierung der Komponenten wurde mit der nachfolgenden Grundgleichung 3.11 durchgeführt:

$$f_g = \frac{1}{2 \cdot \pi \cdot \sqrt{L \cdot C}} \quad (3.11)$$

Nun wird die Grenzfrequenz f_g mit $20kHz$ festgelegt und ein Kondensator C mit dem Wert $1\mu F$ gewählt. Jetzt kann die Gleichung 3.11 nach der Spule L umgestellt werden:

$$L = \frac{1}{(2 \cdot \pi \cdot f_g)^2 \cdot C} \quad (3.12)$$

Durch das Einsetzen der Werte für C und f_g in Gleichung 3.12 erhält man die folgende Induktivität für die Spule:

$$L = \frac{1}{(2 \cdot \pi \cdot f_g)^2 \cdot C} = \frac{1}{(2 \cdot \pi \cdot 20kHz)^2 \cdot 1\mu F} = 63.33\mu H \quad (3.13)$$

Unter Einhaltung der „E12-Reihe“ wird nun der berechnete Wert von $63.33\mu H$ auf den nächstliegenden Wert der Reihe von $68\mu H$ gesetzt. Da es nun zu Abweichungen in der Berechnung kommt, muss die Grenzfrequenz f_g noch mit den bestimmten Bauteilwerten von L und C zurückgerechnet werden. Durch das Einsetzen der Werte in die Gleichung 3.11, erhält man die nachfolgende Grenzfrequenz f_g :

$$f_g = \frac{1}{2 \cdot \pi \cdot \sqrt{L \cdot C}} = \frac{1}{2 \cdot \pi \cdot \sqrt{68\mu H \cdot 1\mu F}} = 19.3kHz \quad (3.14)$$

Es ist eine Differenz zur ursprünglichen Annahme von $700Hz$ zu erkennen, die jedoch in dieser Anwendung vernachlässigt werden kann. In Abbildung 9 ist der Aufbau zu sehen. Dabei ist C_1 der Entkopplungskondensator und L_1 und C_2 bilden zusammen die Filterstufe. Rechts ist der Ausgang zur Verstärkerstufe zu sehen, welche im nächsten Kapitel beschrieben wird.

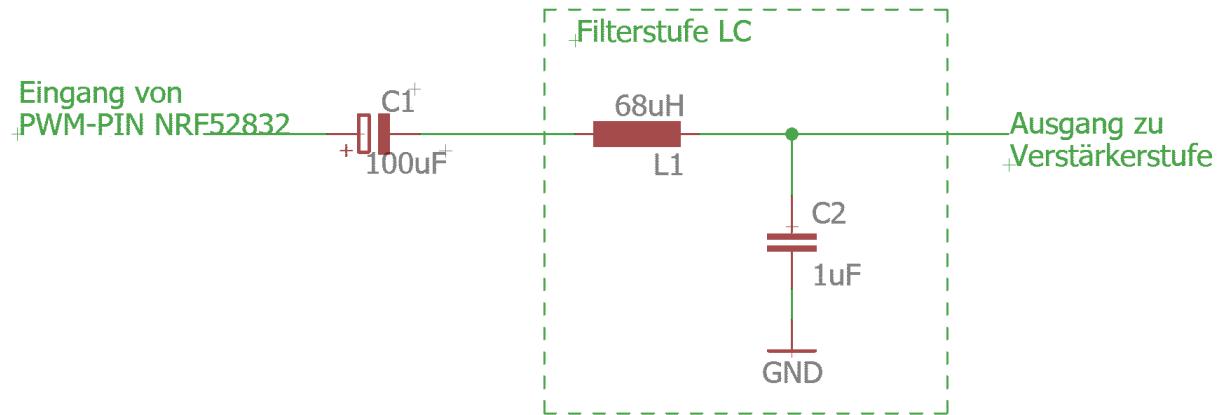


Abbildung 9: Schema der LC-Filterstufe

3.6 Verstärkerstufe

Mit einer Verstärkerstufe lassen sich auf einfache Art und Weise Signale jeglicher Form verstärken. Sie eignen sich bestens, um den Ausgang eines Mikrocontrollers entsprechend aufzubereiten, da die Ausgangsseite meist sehr niedrige Ströme aufweist. Dadurch kann dem Knochenschallaktor genügend Energie zur Verfügung gestellt werden. Prinzipiell gibt es zwei Arten von Verstärkern. Entweder erfolgt die Umsetzung digital oder analog. Beide erfüllen die gleiche Aufgabe, weisen jedoch bezüglich Wirkungsgrad einen deutlichen Unterschied auf. Die digitale Variante weist ungefähr einen Wirkungsgrad von 90% auf [?], während die analoge Variante einen maximalen Wirkungsgrad im Bereich der Leistungsanpassung erzielt [?]. Aus diesem Grund wird ein digitaler Verstärker (Class-D-Verstärker) in der Anwendung implementiert. Die Wahl fiel auf den Stereo-Amplifier MAX 98306. Der Verstärker hat in Betrieb einen Stromverbrauch von $143mA$ und eine Speisespannung von $3.3V$. Somit hat er einen Leistungsverbrauch von $471.9mW$. Im Standby benötigt er lediglich $2mA$ und somit $6.6mW$ [?]. In Abbildung 10 ist der schematische Aufbau des Verstärkers dargestellt.

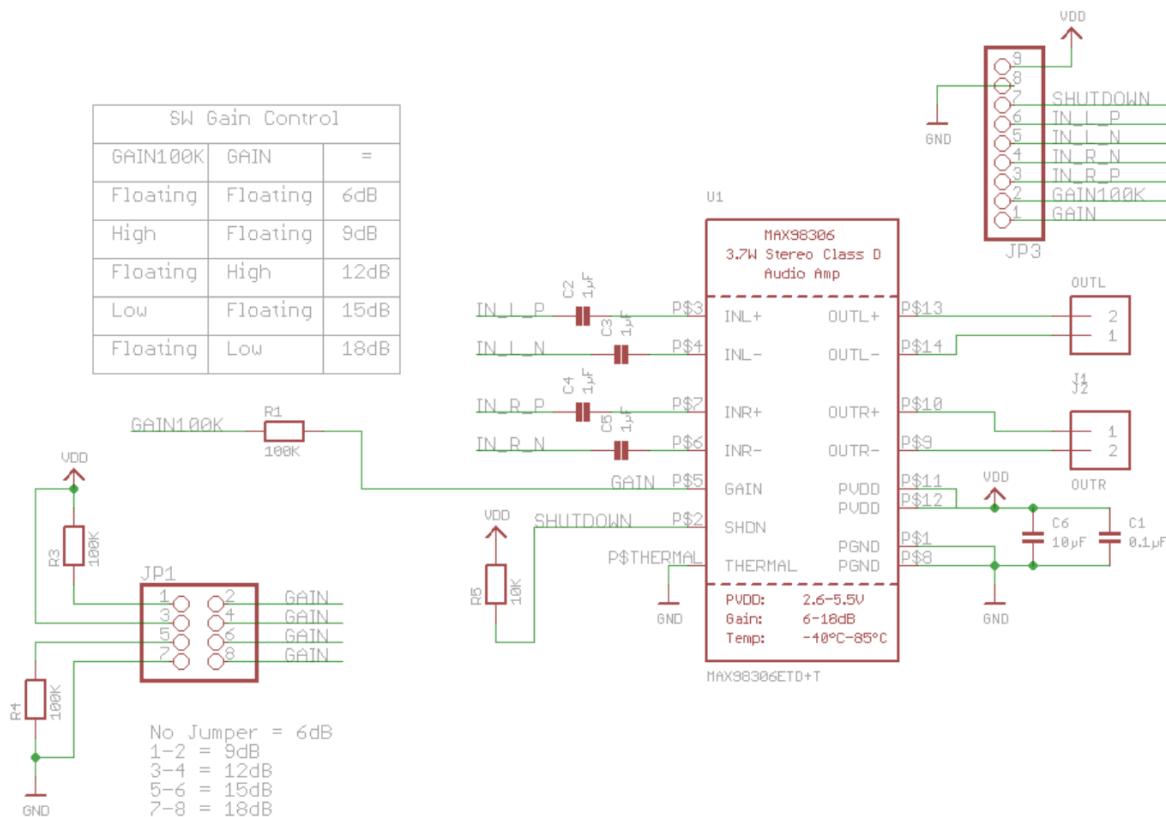


Abbildung 10: Schema der Verstärkerstufe

3.7 Knochenschallaktor

Nachdem das vom Mikrocontroller ausgegebene Audio-File über die Filter- und Verstärkerstufe entsprechend aufbereitet wurde, kann nun die Audiodatei über einen sogenannten Knochenschallaktor ausgegeben werden. Der Aktor arbeitet nach dem Prinzip der Weiterleitung von Schall-Schwingungen oder auch Vibrationen. Dadurch lässt sich der ursprüngliche Gehörgang umgehen und die Schwingungen werden über den Schädelknochen an das Innenohr übertragen. Dies verbessert auch die Hygiene der Anwendung, da kein direkter Kontakt mit dem Gehörgang

stattfindet[?]. Falls weitere Informationen zur Knochenschalltechnologie gewünscht werden, wird an dieser Stelle auf die Quelle [?] verwiesen. Für die Anwendung im Dōjō wird ein Knochenschallaktor des Herstellers Adafruit verwendet, welcher in der Abbildung 11 ersichtlich ist.

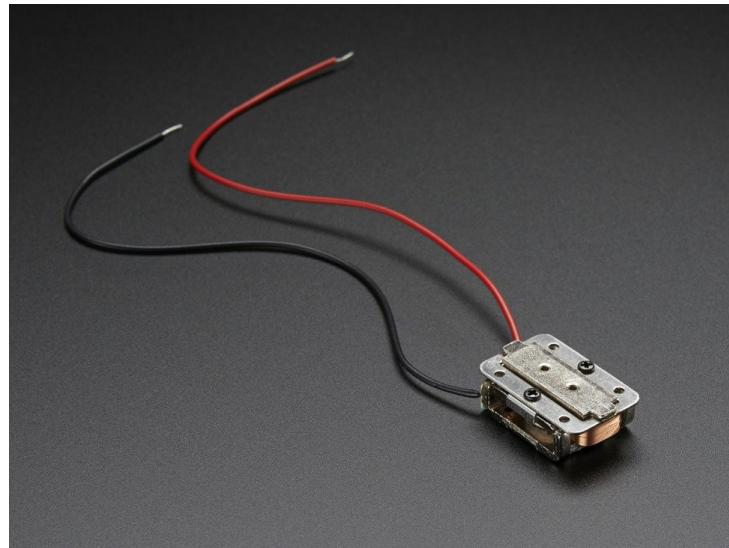


Abbildung 11: Knochenschallaktor von Adafruit

Das ausgewählte Bauteil eignet sich bestens für die Verwendung im Dōjō. Mit einem Gewicht von $9.6g$ und den Dimensionen $(14x21,5x7,9)mm$ lässt sich der Aktor gut in das bestehende Gehäuse implementieren [?]. Weiter ist das Bauteil relativ kostengünstig im Handel erhältlich und kann $1W_{RMS}$ Leistung liefern, was sich dann in der Lautstärke bemerkbar macht. Nach ausführlichen Recherchearbeiten konnten keine wirklichen Alternativen ausgemacht werden. Meist befindet sich die Technologie noch in der Entwicklungsphase oder fällt aufgrund des Preises aus der Auswahlmöglichkeit.

4 Software

Nachdem das Kapitel Hardware ausführlich behandelt wurde, muss nun noch die Software dazu beschrieben werden. Der verwendete Mikrocontroller legt die Verwendung des Software Development Kit [?] (SDK) nahe. Dies ist eine Sammlung von Beispielen, Librarys und vorcompilierten Codes. Verwendet wurde die Version v12.3.0. Die SDK ist aufgrund des verwendeten integrierten Bluetooth-Stack wichtig. Dieser ist im sogenannten Softdevice enthalten. Um ihn nutzen zu können, verwenden wir den S132. Dieser und die nötige Initialisierungen des Bluetooth-Stacks waren in dem Beispielprojekt Uartc in der Central-Rolle vorhanden. Dadurch wurde das ganze Projekt auf diesem Beispiel aufgebaut. Der Softdevice und die SDK legen einige Abstraktions-Layer auf die Hardware. Diese sind in Abbildung 12 visualisiert. Die wichtigsten Module des SDK werden im Kapitel 4.2 erklärt, falls weitere Informationen gewünscht sind, wird auf die offizielle Dokumentation verwiesen [?]. Wichtig für dieses Kapitel ist, dass der beschriebene Software-Zustand dem Soll-Zustand entspricht. Die Abweichungen sind im Kapitel Validierung genauer beschrieben.

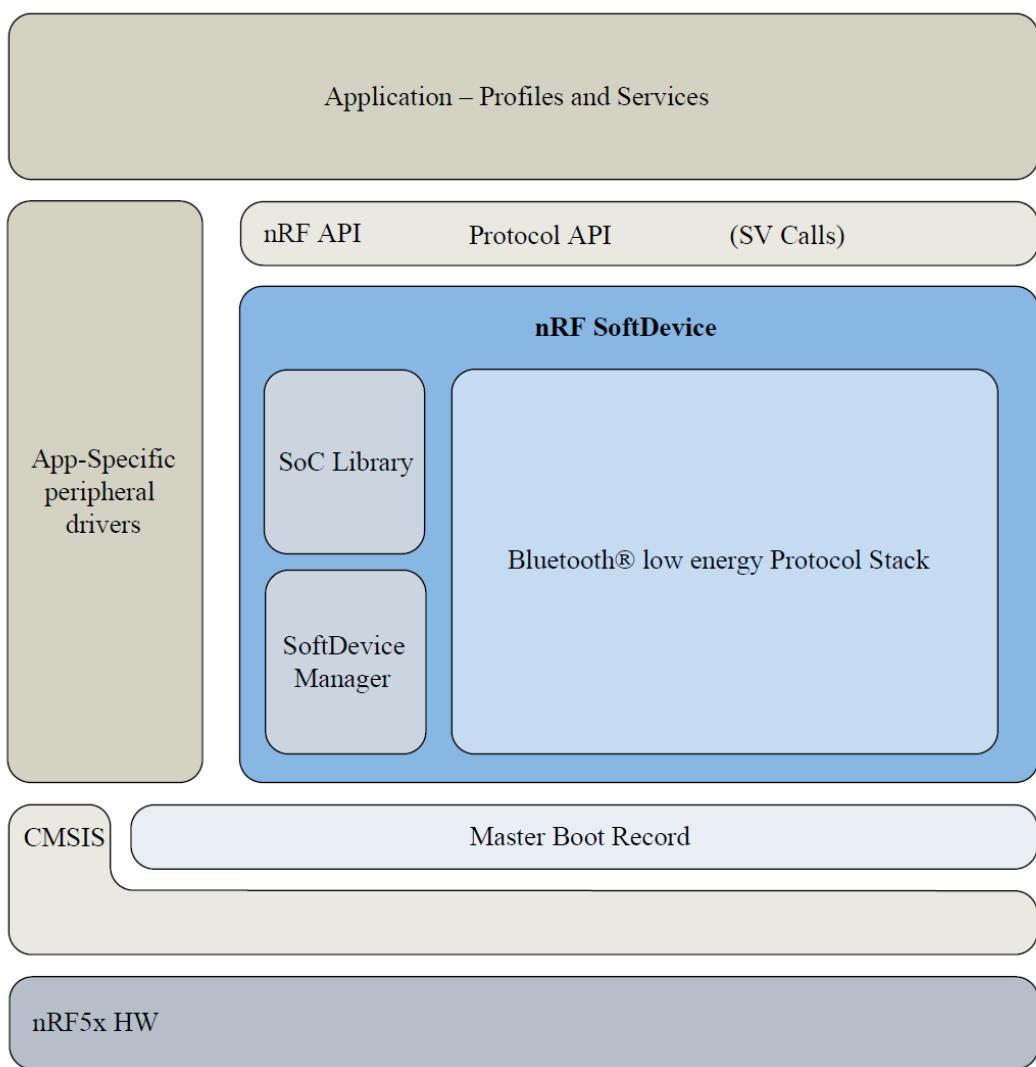


Abbildung 12: Aufbau der Software auf der Hardware

Der eigentliche Programmaufbau ist eine State machine, welche so aufgebaut ist, dass alle Events möglichst kurz gehalten wurden. Jedoch hat dies relativ viele Flags zur Folge. Anschliessend

werden die Events entsprechend ihrer Priorität verarbeitet. Die Prioritäten ergeben sich aus der Else IF im Wait-State, genaueres dazu wird im Kapitel 4.1 beschrieben. Dadurch entsteht ein pollendes Programm mit Prioritäten, wie in Abbildung 13 dargestellt.

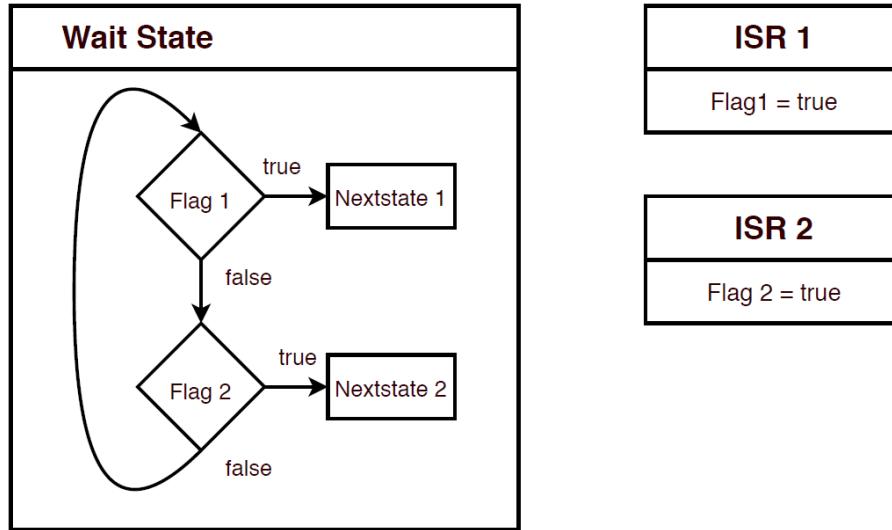


Abbildung 13: Konzept der pollenden Software

Das Programm wurde in verschiedene Module unterteilt, um die Leserlichkeit zu verbessern. Die Unterteilung wurde entsprechend der Funktionen gemacht. Im Hauptmodul (Main) sind nur die Statemachine und die Interrupthandler vorhanden. Folglich wurde ein Modul für die BLE-Funktionalitäten geschrieben und eines für die SD-Karte. Ein weiters Modul existiert für die Batterie-Funktionalitäten. Dieses ist jedoch nicht implementiert und enthält nur Dummy-Funktionen. Nachfolgend wird die Statemachine mit den einzelnen States beschrieben und teilweise grafisch dargestellt.

4.1 State-Machine

Nachdem mit dem Einleitungskapitel ein kurzer Überblick geschaffen wurde, kann nun der Hauptteil der Software genauer betrachtet werden. Der gesamte Ablauf basiert auf einer klassischen State machine, die aufgrund von unterschiedlichen Parametern in die entsprechenden nächsten States springt. Das hat den Vorteil, dass sich das Programm stets in einem definierten Zustand befindet und mittels entsprechenden Parametern jeweils den nächsten Arbeitsschritt vordefiniert. Die Abbildung 14 zeigt das Gesamtkonzept der State-Machine. Anschliessend werden die einzelnen States genauer definiert und beschrieben.

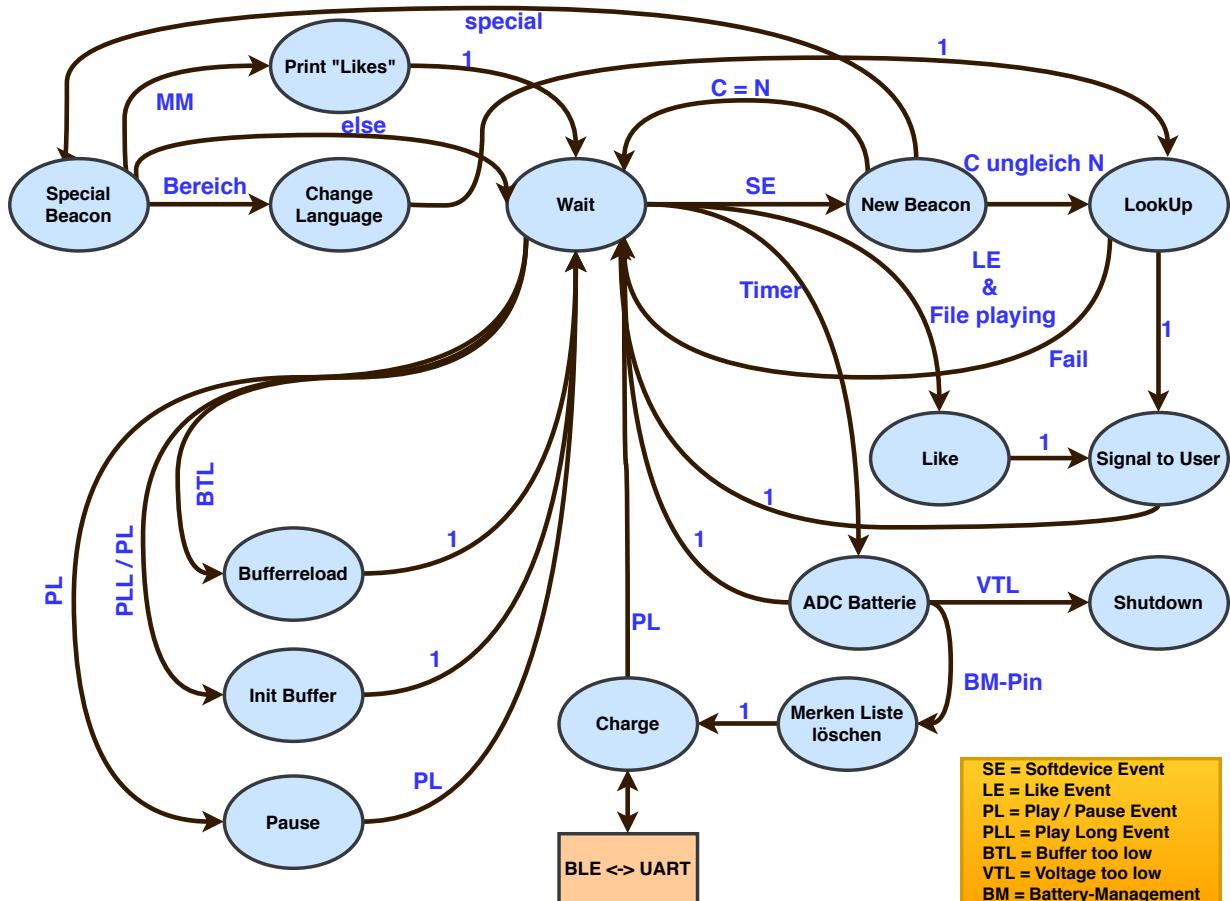


Abbildung 14: Statemachine mit den einzelnen States und den Parametern im Überblick

State: Lookup

In diesem State verschafft sich das Programm über die entsprechende Initialisierung Zugriff auf die SD-Karte der Anwendung. Falls der Mikrocontroller nicht auf die SD-Karte zugreifen kann, wird eine Fehlermeldung ausgegeben und die Funktion wird beendet. Andernfalls wird dem Mikrocontroller signalisiert, dass der Zugriff geglückt ist und die eigentliche Funktion wird gestartet. Dazu werden die beiden Minor- und Majorzahlen in ein hexadezimales Zahlensystem gewandelt, welche dann als Vergleichskriterium verwendet werden. Falls die Nummer gefunden wird, kann das entsprechend zugehörige Audio-File über den Mikrocontroller ausgegeben werden. Verglichen wird jeweils zeilenweise, weshalb auch ein Fehlerhandling eingebaut wurde. Damit wird erkannt, ob sich das Textfile am Ende befindet. Somit lässt sich dann die Suche wiederholen, oder einen Fehler ausgeben. Die nachfolgende Abbildung 15 zeigt den detaillierten

Funktionsablauf im Lookup-State. Wichtig ist an dieser Stelle die Erkenntnis, dass der Lookup-State die Verbindung zwischen Bluetooth und dem Audio-File repräsentiert.

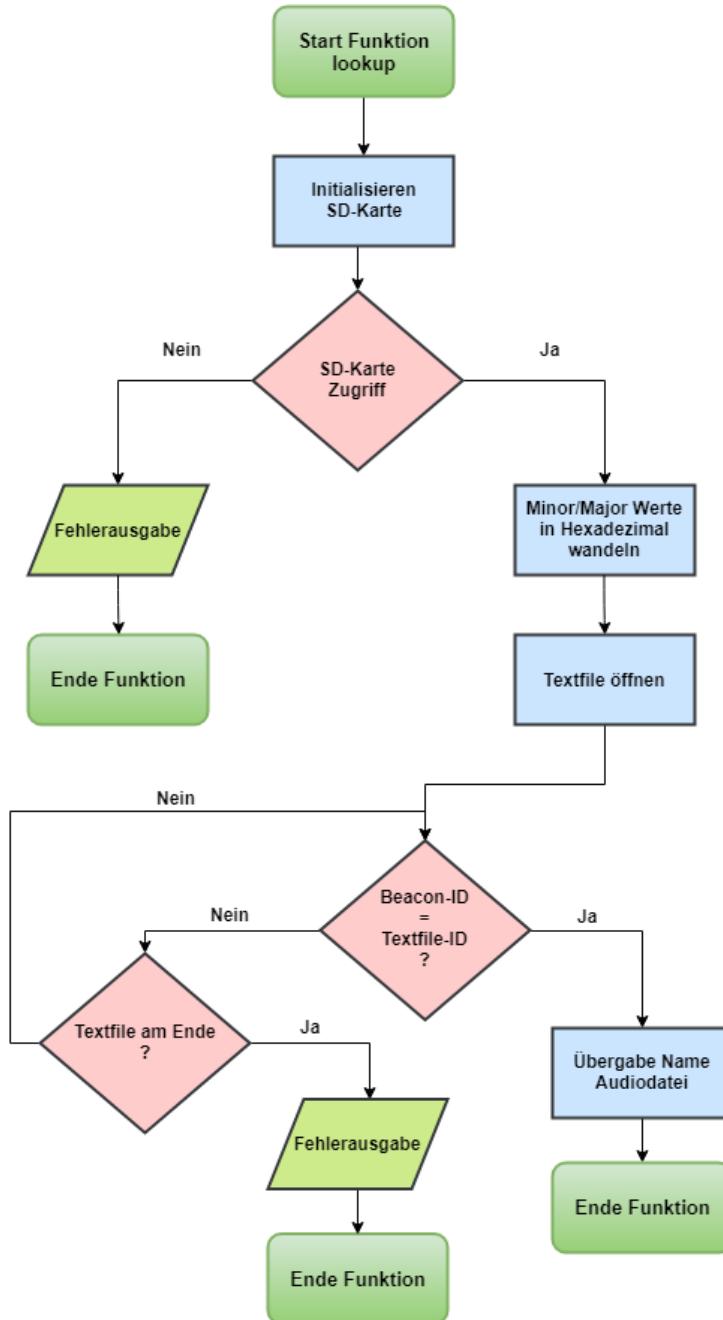


Abbildung 15: Funktionsablauf im lookup-State

State: Print „Likes“

Der State Print „Likes“ wurde nicht implementiert. Das hat zur Folge, dass das Hauptprogramm direkt in den Wait-State springt. Es ist eine optionale Möglichkeit, die in einem weiteren Entwicklungsschritt bearbeitet werden kann. Die Software ermöglicht es aber diese Funktion noch einzubetten. Dabei könnte die Broschüre mit den interessanten Objekten direkt gedruckt werden und bereits für den Besucher am Ausgang des Museums bereit liegen.

State: Special Beacon

Special Beacon dient hauptsächlich zur Unterscheidung der verschiedenen Beacons für die Sprache, das Drucken und weitere Features die in einem weiteren Ansatz implementiert werden können. Aus diesem Grund wurde dieser State auch relativ einfach gehalten. Zuerst wird verglichen, ob es sich dabei um die Sprachkonfiguration handelt. Ist dies zutreffend, so springt das Programm in den Change Language State. Andernfalls wird überprüft, ob gerade der Like-Button gedrückt wird und entsprechend in den Print „Likes“ State gewechselt wird. Ist keine der beiden Zustände zutreffend, springt das Programm in den Wait-State. Die nachfolgende Abbildung 16 zeigt den Ablauf der Funktion.

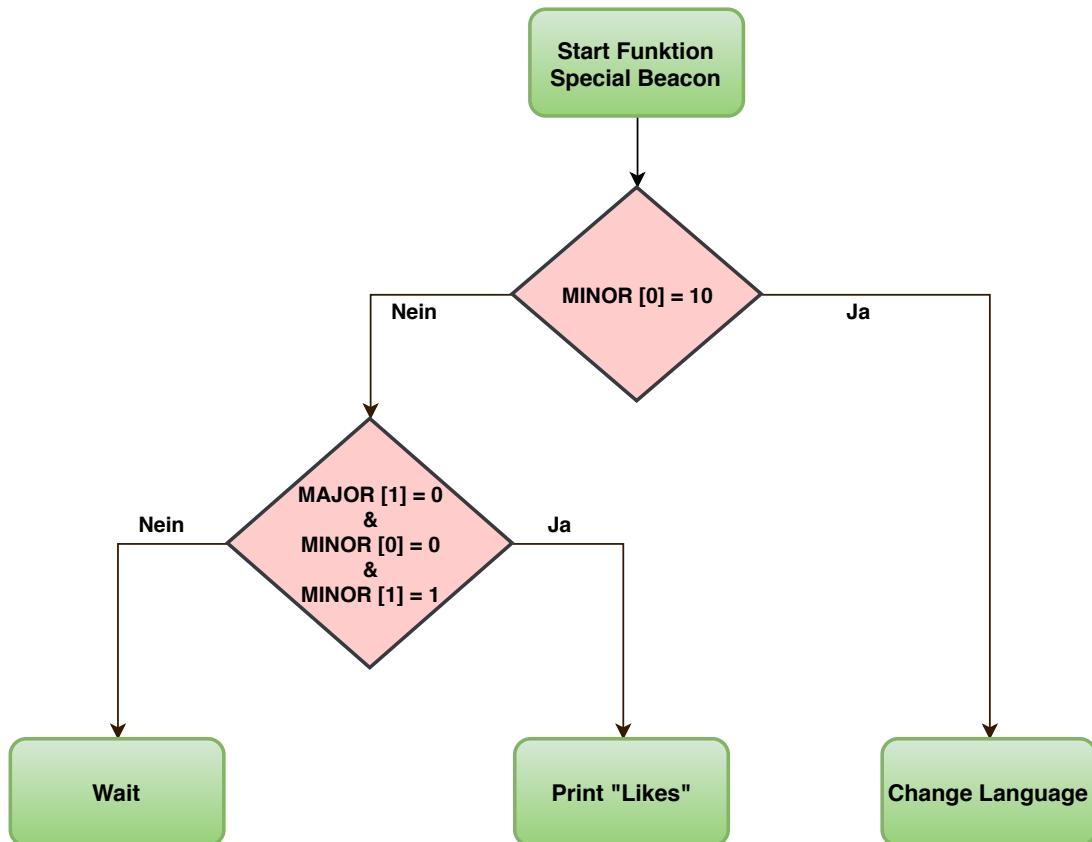


Abbildung 16: Funktionsablauf im Special Beacon State

State: Pause

Dieser State pausiert das Sound-File und wartet auf ein weiteres Play-Event, um die Audiodatei wieder abzuspielen.

State: Init Buffer und Bufferreload

Diese beiden States sind für das Initialisieren des Buffers und für den Start der PWM-Funktion verantwortlich. Für weitere Informationen dazu, wird auf das Kapitel 4.4 verwiesen.

State: Merken Liste löschen

Dieser State löscht die Likes, um für den nächsten User bereit zu sein. Anschliessend folgt der Charge State.

State: Change Language

Dieser State ist für die Sprachauswahl verantwortlich. Aufgrund des Zahlenwertes in Minor [1] wird zwischen den Landessprachen der Schweiz ausgewählt. Dabei wird die Vergleichstabelle in Form einer Datei kopiert und mit einem Kürzel entsprechend der Sprache versehen. Danach springt das Programm wieder in den Lookup-State.

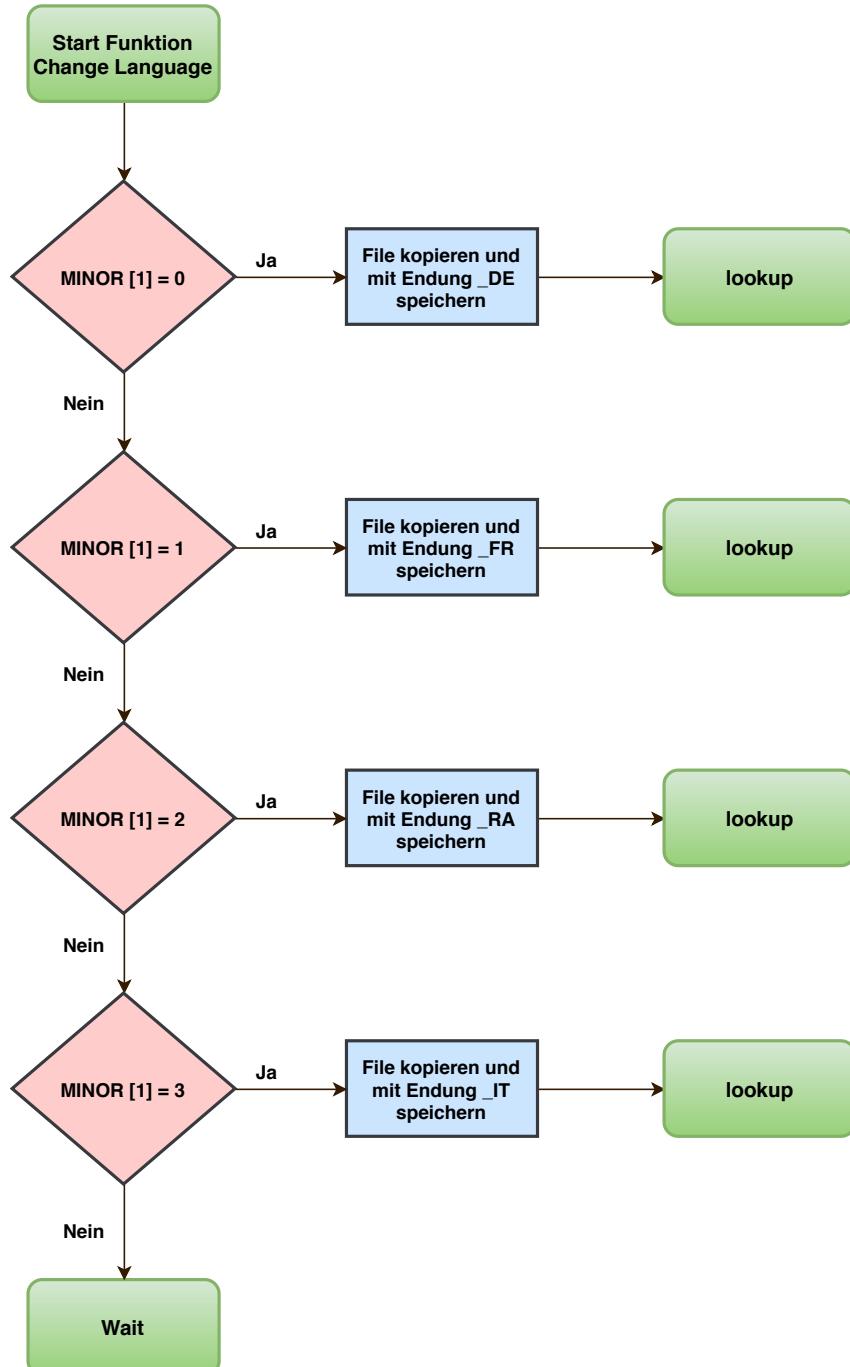


Abbildung 17: Funktionsablauf im Change Language State

State: New Beacon

Wird ein Beacon erkannt, dann wird der New Beacon State aufgerufen. Ist das aktuelle Beacon immer noch das gleiche wie das alte Beacon, springt das Programm in den Wait-State. Wenn es sich nicht um das gleiche Beacon handelt, wird über den Lookup-State überprüft, ob eine entsprechende Übereinstimmung vorhanden ist.

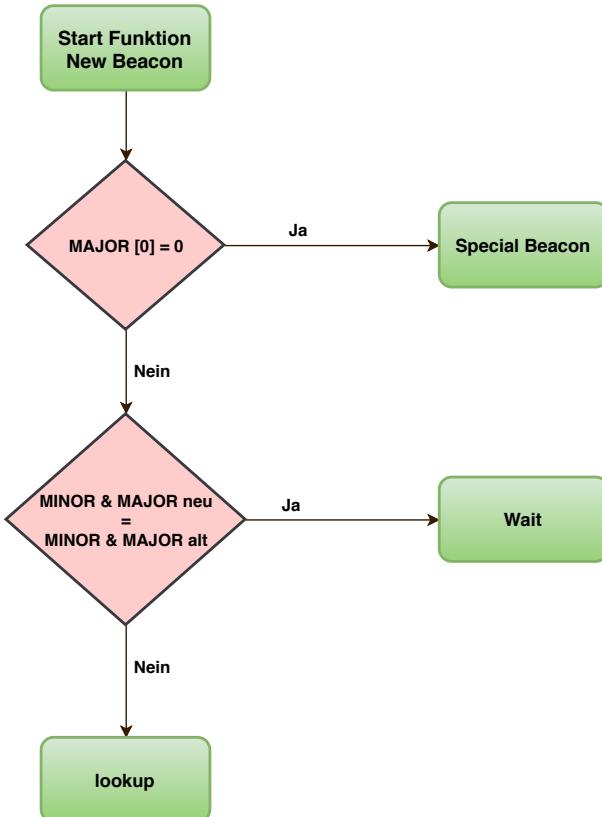


Abbildung 18: Funktionsablauf im New Beacon State

State: Like

Wird in diesem State über einen Button ein Like-Ereignis ausgelöst, springt das Programm in den Signal to User State und speichert den entsprechenden Namen des Audio-Files auf die SD-Karte. Findet kein Ereignis statt, springt das Programm in den Wait-State.

State: Signal to User

Dieser State hat die Aufgabe, dem Benutzer mitzuteilen, dass ein neues Audio-File verfügbar ist, oder ob ein Kunstobjekt gespeichert wurde. Dabei blinkt eine dafür vorgesehene LED. Anschliessend springt die Software in den Wait-State.

State: Wait

Dieser State ist die Ausgangslage des kompletten Ablaufs. Aufgrund von den in der Abbildung 19 ersichtlichen Bedingungen und sogenannten Flags, entscheidet das Hauptprogramm in welchen State zu switchen ist. Die Reihenfolge der Abfrage erfolgt nach dem Prioritätsprinzip.

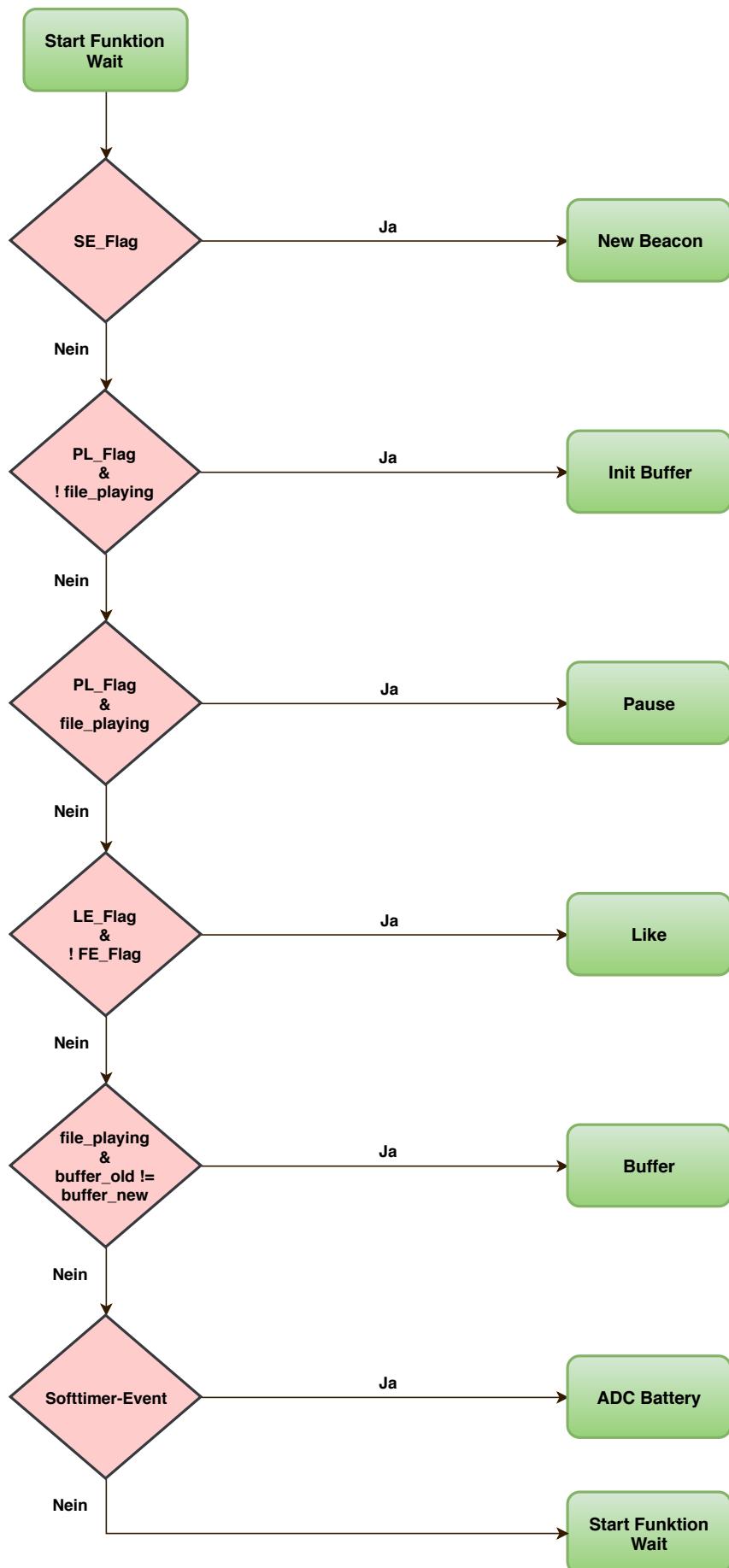


Abbildung 19: Funktionsablauf im Wait State

State: ADC Battery

ADC-Battery wird zyklisch durch einen Softtimer im Wait-State aufgerufen, jedoch mit einer tiefen Priorität. Funktional wird hier die Batterie auf ihren Ladezustand überprüft und gemäss Abbildung 20 entsprechend gehandelt. Wird die Batterie geladen, so wird die Like-Liste gelöscht. Andernfalls wird das Programm in den Wait-State wechseln, oder bei ungenügendem Ladezustand das System herunterfahren.

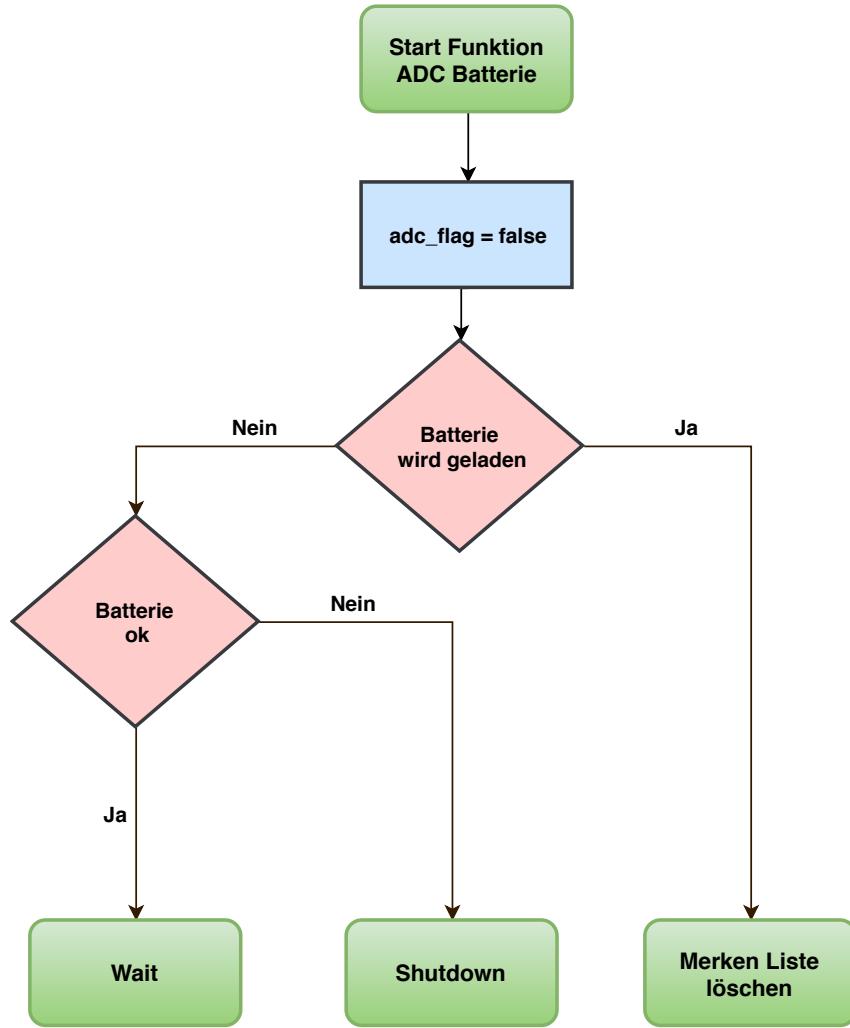


Abbildung 20: Funktionsablauf im ADC Battery State

State: Shutdown

Ist der Ladezustand des Geräts zu tief, wird das System gezwungenermassen heruntergefahren. Somit sollen Schäden an der Hardware und an der Software vermieden werden.

State: Charge

Hier wird überprüft, ob ein Pause- oder Playevent vorliegt. Falls ein Event vorliegt, springt die Software in den Wait-State und ist bereit für den nächsten User. Falls kein Event vorliegt, befindet sich das Programm in einer Schlaufe. Darin ist es möglich den Dojo mit der Ladestation zu verbinden und über UART spezifische Anpassungen auf der SD-Karte vorzunehmen.

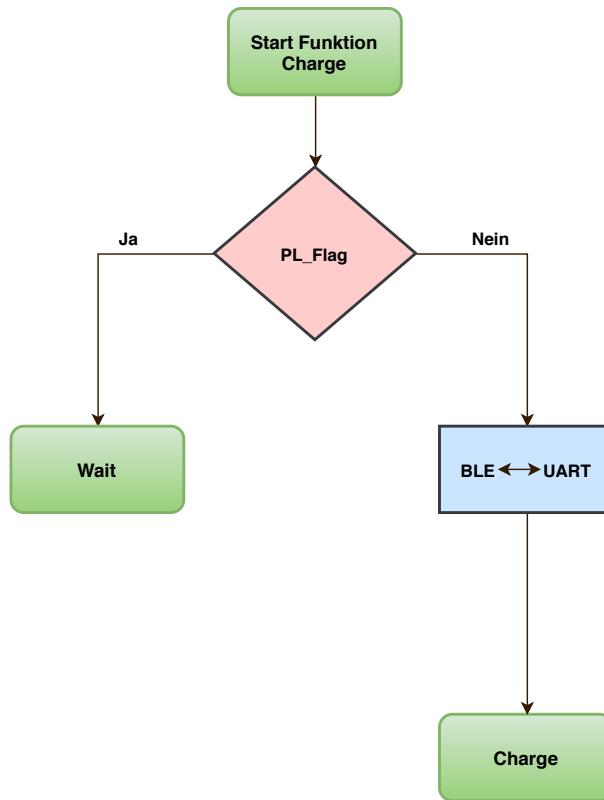


Abbildung 21: Funktionsablauf im Charge State

4.2 Nordic Software Development Kit

Das Software Development Kit (SDK) ist eine Sammlung von nützlichem Code für die NRF-Chip Familie. Sie ist sehr umfangreich und ein absolutes Muss für unser Projekt. Hier wird jedoch nur auf ausgewählte Module innerhalb der SDK eingegangen, welche für das Projekt entscheidend sind. Zuerst wird auf das Softdevice S132 eingegangen. Danach folgt das Board Support Package und zum Schluss wird noch das NRF Log Modul beschrieben. Für weitere Informationen wird auf die offizielle Dokumentation [?] verwiesen.

Softdevice S132

Der Softdevice 132 ist ein Protokoll Stack nach Bluetooth 5. Er stellt alle Funktionen des Bluetooth-Low-Energy (BLE) Stacks zur Verfügung, was für das Projekt entscheidend war. Der Stack wird in der Central-Rolle verwendet. Jedoch ist der Softdevice nicht als Code erhältlich, es liegen nur die sogenannten Binarys bei.

Board Support Package (BSP)

Das Board Support Package ist eine Abstraktions-Schicht, die das Ziel hat, die Hardware von der Software zu trennen und die Portierbarkeit zu erhöhen. Sie arbeitet mit Callbacks und bietet viele Funktionen für typische Hardware, zum Beispiel für Tasten. Dadurch lässt sich für kurze und lange Betätigungszeiten der Tasten jeweils verschiedene Callbacks einrichten. Die Verlinkung zwischen der Hardware und den entsprechenden Defines wird mit dem File boards.h gemacht und muss bei der Verwendung eines anderen Prints entsprechend angepasst werden.

NRF Log

Das Modul NRF Log ist ein mächtiges Debug-Werkzeug. Damit ist es möglich aus der laufenden Software auf eine Konsole zu schreiben, welche frei wählbar ist. Das Projektteam hat sich auf die Variante mit Putty geeinigt. Weiter können auch hilfreiche Warnungs-Stufen festgelegt werden. Damit wird ein entsprechendes Feedback generiert.

4.3 Bluetooth

Ein weiterer wichtiger Baustein der Softwareentwicklung ist das Bluetooth. Aus diesem Grund wird an dieser Stelle das Bluetooth ausführlich behandelt und detailliert erklärt. Gestartet wird mit den Grundlagen. Anschliessend wird das GAP und das GATT beschrieben. Danach folgen die Kapitel BLE-Beacons und RSSI. Am Schluss wird noch die Major- und Minorvergabe behandelt.

4.3.1 Grundlagen

Standard-Bluetooth-Geräte senden in einem lizenzenfreien Band mit einer Frequenz zwischen 2.402 und 2.480 GHz. Dabei können Störungen durch diverse andere Geräte auftreten, die im selben Frequenzband arbeiten. Um eine Robustheit gegenüber Störungen zu erhalten, wird ein Frequenzsprungverfahren eingesetzt. Dadurch wird das Frequenzband in 79 Kanäle eingeteilt und bis zu 1600-Mal in der Sekunde gewechselt [?].

Derzeitiger Bluetooth-Standard ist Bluetooth 5. Es zeichnet sich im Gegensatz zu seinen Vorgängern durch eine höhere Reichweite (bis zu 100 m statt 25 m) und eine schnellere Datenrate (bis zu 2 Mbit/s statt 1 Mbit/s) aus. Ausserdem enthält dieser Standard ebenso den im Standard 4 bereits eingeführten „Low Energy“-Modus, welcher den schon moderaten Energieverbrauch zusätzlich senkt, was für dieses Projekt sehr wichtig ist [?]. Im Low-Energy-Modus wird das Frequenzband für das Verfahren nicht in 79 sondern in 40 Kanäle unterteilt. Ausserdem wird durch die Datenrate Energie gespart, indem eine Geschwindigkeit von 1 Mbit/s statt 2 Mbit/s erreicht werden kann. Die typische Reichweite im Low-Energy-Modus beträgt 40 m, was die Mindestanforderung von 5 Metern für die Anwendung im Projekt überschreitet und folglich genügt [?]. Damit über Bluetooth Low Energy (BLE) Verbindungen aufgebaut werden können, benötigt es sogenannte Profile. Beim Bluetooth sind dies das GAP (Generic Access Profile) und das GATT (Generic Attribute Profile), auf welche in den nachfolgenden beiden Kapiteln näher eingegangen wird.

4.3.2 Generic Access Profile

Das GAP kontrolliert die Verbindungen und die Authentifizierungen. Es beschreibt grundsätzlich die Art und Weise wie Geräte miteinander kommunizieren. Dazu werden zwei verschiedene Rollen für die Geräte definiert. Zum einen die Rolle des zentralen Gerätes und zum anderen die Rolle des Peripheriegerätes [?]. Damit eine Verbindung aufgebaut wird, muss das Peripheriegerät in einem bestimmten Intervall das Payload-Datenpaket senden. Empfängt ein zentrales Gerät das Payload-Paket, kann es für zusätzliche Daten ein Antwortpaket (Scan Response Payload) anfordern, welches daraufhin vom Peripheriegerät gesendet wird [?]. Meistens senden die Peripheriegeräte ihre Authentifizierung und das GATT. Die Authentifizierung dient dem Verbindungsauflaufbau und das GATT dem grösseren Datentransfer. Solange die Verbindung aktiv ist, kann das Peripheriegerät keine weiteren Verbindungen eingehen. Wird jedoch nur eine kleine Datenmenge benutzt, kann die Datenmenge in das Payload integriert werden, wodurch alle zentralen Geräte in der Nähe einen Zugriff auf die Daten haben. Dieses Verfahren nennt man Broadcasting und ist für dieses Projekt von zentraler Bedeutung. Dennoch wird zuerst im nächsten Kapitel das GATT erläutert [?].

4.3.3 Generic Attribute Profile

Das GATT definiert die Art und Weise, wie Peripherie und zentrales Gerät miteinander Daten austauschen. Dies wird mit Hilfe des Attribute Protocol (ATT) [?] umgesetzt. Das ATT speichert Services, Characteristics und dazugehörende Daten. Services sind logische Sammlungen von zusammengehörenden Characteristics, wobei Characteristics als Datenpunkte betrachtet werden können. Demzufolge besteht das ATT aus einer in Services geordneten Datensammlung [?]. Funktional betrachtet speichern Peripheriegeräte das GATT und dienen somit als GATT-Server. Zentrale Geräte verbinden sich mit Hilfe des GAP mit dem GATT-Server und stellen eine entsprechende Anfrage. Das zentrale Gerät wird somit zum GATT-Client. In einem Intervall werden vom Client gesendete Anfragen vom Server mit Datenpaketen beantwortet. So können grössere Datenpakete ausgetauscht werden, jedoch kein Broadcasting stattfinden [?]. Das genannte zentrale Gerät bzw. GATT-Client ist in dieser Anwendung der Dōjō. Die genannten Peripheriegeräte sind die BLE-Beacons, welche in der Nähe der jeweiligen Kunstobjekte angebracht sind und im nächsten Kapitel genauer beschrieben werden [?].

4.3.4 BLE-Beacons

BLE-Beacons (Bluetooth Low Energy Beacons) sind kleine Peripheriegeräte, die kleine Informationsmengen übertragen. Wird z.B. ein Temperaturverlauf über ein ganzes Jahr hinweg gemessen, so kommen BLE-Beacons zum Einsatz. Der Grund dafür liegt im Energieverbrauch. Sie können mit einer Knopfzellenbatterie über Jahre hinweg in Betrieb bleiben [?].

Generell können BLE-Beacons vier Rollen einnehmen, welche in zwei verbindungsfähige Rollen und 2 nicht-verbindungsfähige Rollen eingeteilt werden. Die verbindungsfähigen Rollen sind diejenigen des Peripheriegerätes und des zentralen Gerätes. Die nicht-verbindungsfähigen Rollen sind diejenigen des Senders und des Beobachters. Als Peripheriegerät fungiert das BLE-Beacon als Slave und wartet somit auf einen Input des zentralen Gerätes (Master). Als zentrales Gerät fungiert das BLE-Beacon als Master und kann Verbindungen mit einem oder mehreren anderen Geräten eingehen. Als Sender (Broadcaster) kann das BLE-Beacon zwar keine Verbindungen eingehen, aber vordefinierte Werte senden. Wird das BLE-Beacon als Beobachter (Observer) benutzt, so kann der Observer von einem anderen BLE-Beacon Werte empfangen und an einem angeschlossenen Display anzeigen. Die übermittelten Signale der BLE-Beacons werden gemäss Abbildung 22 formatiert [?].

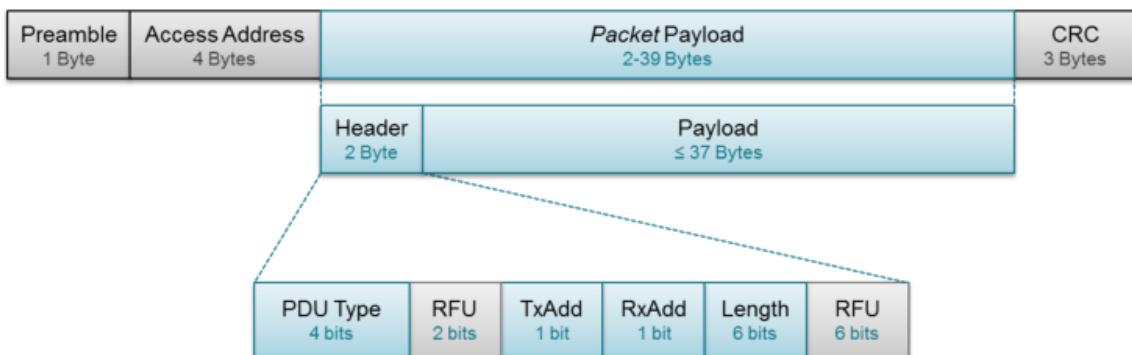


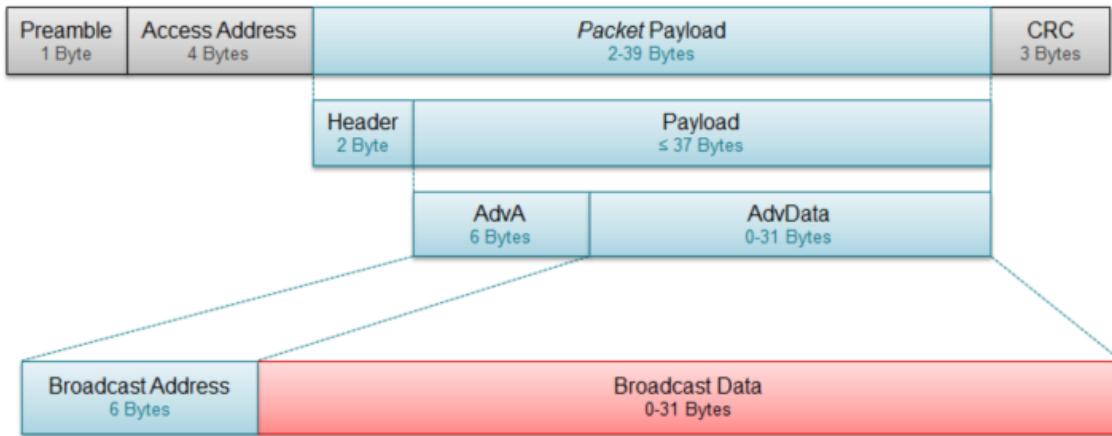
Abbildung 22: PacketPayload Header

Die Preamble wird zur Synchronisierung und Zeitschätzung benötigt und ist für BLE-Beacons, welche in der Rolle eines Broadcasters sind, immer 0xAA. Auch die Access Address ist für Broadcaster immer gleich, nämlich 0x8E89BED6. Das Payload-Paket beinhaltet Header und Payload. PDU bestimmt den Sendekanaltyp des sendenden Beacons gemäss Tabelle 1.

PDU Type	Packet Name	Description
0000	ADV_IND	Connectable undirected advertising event
0010	ADV_NONCONN_IND	Non-connectable undirected advertising event
0110	ADV_SCAN_IND	Scannable undirected advertising event

Tabelle 1: PDU Type

RFU (Reserved for Future Use) steht, wie der Name schon sagt, als Reserve für zukünftige weitere Implementationen und wird deshalb derzeit nicht gebraucht. TxAdd definiert, ob die Sendeadresse des Beacons öffentlich (TxAdd = 0) oder zufällig (TxAdd = 1) ist. Das RxAdd tangiert die Beacons nicht und wird deshalb nicht weiter erwähnt. Das CRC (Cyclic Redundancy Check) dient der Erkennung von Fehlübertragungen. Das Payload ist gemäss Abbildung 23 aufgebaut [?].

**Abbildung 23:** PacketPayload Payload

Es ist zu sehen, dass es mit der Sendeadresse und den Sendedaten gefüllt ist. Die Sendeadresse kann öffentlich oder zufällig sein, wobei eine öffentliche Sendeadresse eine OUI (Organizationally Unique Identifier) nutzt, welche von der IEEE Registration Authority vergeben wird. Die Sendedaten können gemäss der Tabelle 2 formatiert werden [?].

AD Data Type	Data Type Value	Description
Flags	0x01	Device discovery capabilities
Service UUID	0x02 - 0x07	Device GATT services
Local Name	0x08 - 0x09	Device name
TX Power Level	0xA	Device output power
Manufacturer Specific Data	0xFF	User defined

Tabelle 2: AD Data Type

Die Manufacturer Specific Data können vom Benutzer definiert werden. Hier können Werte gespeichert werden, die das BLE-Beacon senden soll, wie z.B. den RSSI-Wert. Dieser wird im nächsten Kapitel erläutert [?]. Die Flags definieren die Fähigkeiten des BLE-Beacons und sind gemäss der Tabelle 3 definiert.

Byte	Bit	Flag/Value	Description
0	•	0x02	Length of this data
1	•	0x01	GAP AD Type Flags
2	0	LE Limited Discoverable Mode	180 s advertising
•	1	LE General Discoverable Mode	Indefinite advertising time
•	2	BR/EDR Not Supported	•
•	3	Simultaneous LE and BR/EDR (Controller)	•
•	4	Simultaneous LE and BR/EDR (Host)	•
•	5-7	•	Reserved

Tabelle 3: Flags

4.3.5 RSSI

RSSI steht für Received Signal Strength Indicator und ist ein Indikator für die Empfangsfeldstärke bei der drahtlosen Kommunikation. Je höher dieser Wert ist, desto stärker ist das empfangene Signal [?]. Die drahtlose Kommunikation erfolgt über das Senden von elektromagnetischen Wellen. Trifft eine solche elektromagnetische Welle auf eine Antenne, so führt dies zu einer messbaren Selbstinduktion, welche dem RSSI-Wert entspricht. Dieser Wert muss abhängig von der jeweiligen Anwendung interpretiert werden, da diverse Faktoren bezüglich Transmitter, Receiver und Übertragungsmedium den Wert beeinflussen können.

Im Falle des Museums mit gleichwertigen Beacons bedeutet dies, dass der Besucher sich eher in der Nähe des Beacons mit stärkerem RSSI Wert befindet. Dadurch kann das richtige Audiofile abgespielt werden. Zur Unterscheidung der Beacons werden UUID (Universal Unique Identifier), Major und Minor verwendet. UUID ist im Falle des Projekts eine gewählte Identifikationsnummer für das Museum. Major beschreibt eine eindeutig definierte Zimmernummer und Minor ist die definierte Nummer des Beacons im gleichen Raum. Im nächsten Kapitel wird die Minor- und die Majorvergabe beschrieben.

4.3.6 Major Minor Vergabe

Um die Major- und Minornummern mit dem Dōjō zu benutzen, wurden sie im Verlaufe des Projektes standardisiert. Die Sprachauswahl funktioniert über solche Nummern. Beacons mit diesen speziellen Nummern lösen auf dem Dōjō entsprechende Funktionen aus. Abbildung 24 zeigt die Definitionen. Die Nummernräume sind so gestaltet, dass genug Platz für zusätzliche Funktionen vorhanden ist, wie zum Beispiel für weitere Sprachen oder Zugangskontrollen.

**Abbildung 24:** Default Definitionen von speziellen Major- und Minornummern.

In Abbildung 25 ist zu sehen, dass die erste Major Nummer 0x00 sein muss, damit ein spezieller Beacon erkannt wird. Das vereinfacht die Software, weil bei einem neuen BLE-Package nur jeweils der erste Eintrag im Array betrachtet werden muss, um die speziellen Beacons zu erkennen. Der Nummernraum für spezielle Beacons umfasst ca. 16 Mio. Adressen, was für zusätzliche Funktionen reichen sollte. Somit bleiben ca. 4.2 Mia. Adressen für Kunstwerke übrig.

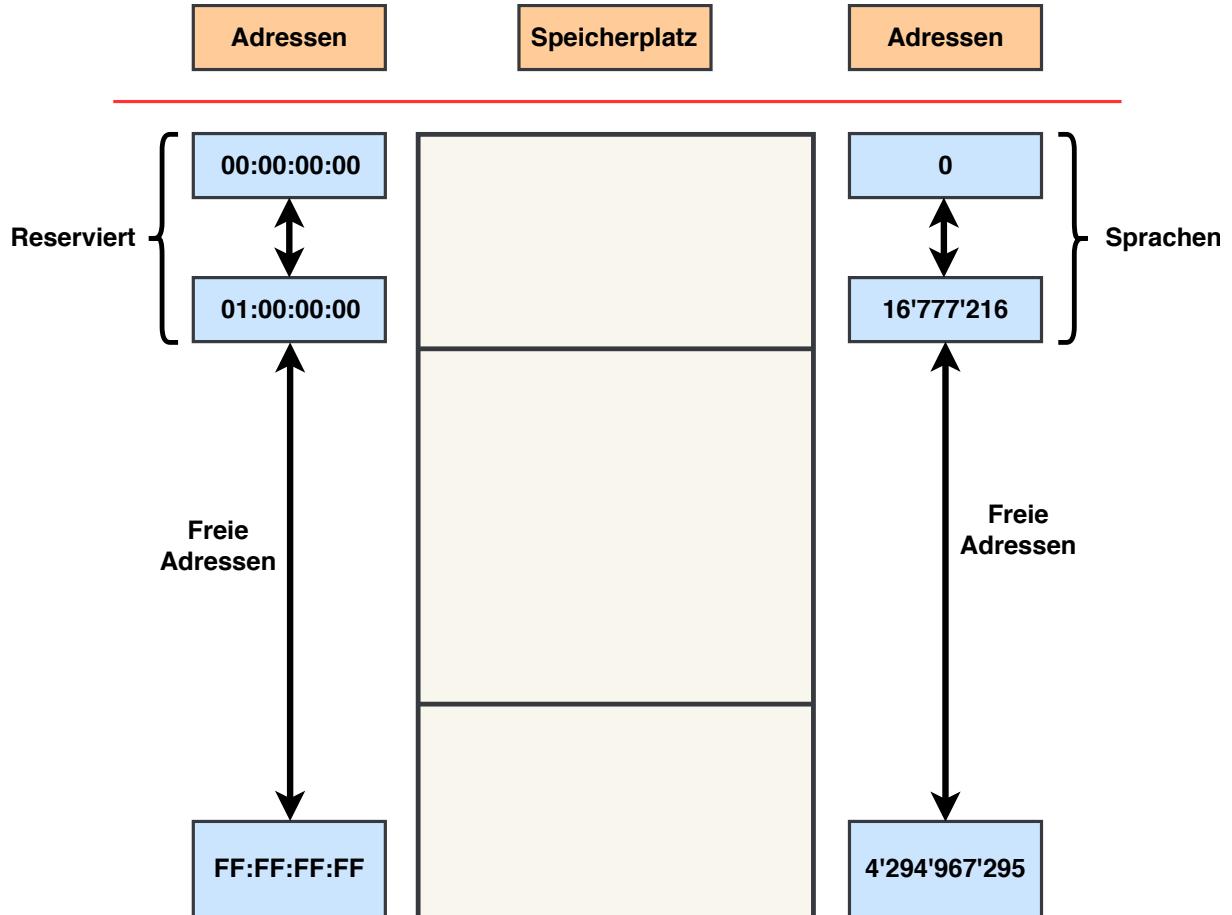


Abbildung 25: Definition der Major Minor Vergabe

4.4 SD-Karte

Nachdem nun die ganze Datenübertragung und Kommunikationsebene erklärt wurde, muss jetzt noch die Datenspeicherung und der Datenzugriff verdeutlicht werden. Damit ein Zugriff auf die SD-Karte möglich ist, wurde die Library fatfs verwendet. Sie ermöglicht einen einfachen Zugriff auf die Daten mit einer SPI-Schnittstelle und wird im ersten Kapitel beschrieben. Anschliessend wird die Bündelung des Codes (Modul) in Zusammenhang mit der SD-Karte erklärt. Das letzte Kapitel beschreibt noch die benötigten Files auf der SD-Karte, um die Funktionen des Dōjōs zu gewährleisten.

4.4.1 FatFs

Für den Zugriff auf die SD-Karte wird ein Generic FAT Filesystem Module verwendet. In dieser Library sind die Funktionen für die Initialisierung und für den Zugriff auf die SD-Karte enthalten. Die FatFs Library untersteht einer 1-clause BSD Lizenz. Sie darf verwendet werden, muss aber in jedem Fall den Lizenztext beinhalten. Der Lizenztext ist im Anhang vorhanden.

Spezifikationen FatFs

Dateisystem Typ: FAT, FAT32(rev0.0) und exFAT(rev1.0)

Anzahl geöffneter Dateien: unlimitiert (hängt vom verfügbaren Speicher ab)

Anzahl Datenträger: bis zu 10

Datenträgergrösse: bis zu 2TB bei 512Bytes/Sektor

Dateigrösse: bis zu 4GB – 1 auf FAT-Volume und praktisch unbegrenzt auf exFAT-Volume

Clustergrösse: Bis zu 128 Sektoren auf FAT-Volume und bis zu 16 MB auf exFAT-Volume

Sektorgrösse: 512, 1024, 2048 und 4096 Bytes [?]

4.4.2 SD-Karte Modul

Das SD-Karten-Modul besteht hauptsächlich aus den Funktionen Merken und SD-Karten-Initialisierung. **Merken** beinhaltet zwei Funktionen. Die Erste speichert das aktuelle Kunstwerk auf eine Liste, auf welcher die verschieden gemerkten Kunstwerken gesammelt werden. Die zweite Funktion löscht die Liste wieder, um den Dōjō für den nächsten Nutzer betriebsbereit zu machen. Der Dateiname der Liste lässt sich im Header-File konfigurieren. **SD-Karte Initialisieren** ruft eine Reihe fatfs-Befehle auf, um die SD-Karte zu mounten. Diese wird über einen SPI-Bus an den Mikrocontroller angeschlossen. Die Pins für den Bus lassen sich im Header-File definieren.

4.4.3 Lookup

Diese Funktion ist die eigentliche Implementierung des in Abbildung 15 beschrieben Algorithmus und hat mit `next_Value` und Sprachwechsel 2 wichtige Funktionen, die nachfolgend beschrieben werden.

`next_Value` dient dazu, die nächsten Werte in den Buffer zu laden. Sie ist folgendermassen definiert:

```
static void next_value(int bufix)
```

Sie öffnet zuerst die entsprechende Audiodatei. Dies geschieht nur zu Beginn der Datei. Sie bleibt geöffnet bis die Audiodatei zu Ende ist, oder der Benutzer das Abspielen unterbricht. Ebenfalls wird der Lesezeiger beim ersten Aufruf der Funktion auf das 44 Byte geschoben, da die ersten 44 Bytes eines WAV-Files keine Audiodaten enthalten. Die Funktion liest nun 4096 Bytes in einen Hilfsbuffer ein. Die Werte werden dann daraus skaliert und in die Sequenzen geladen. Um eine invertierte Sequenz zu erzeugen, wird das Bit 15 auf 1 gesetzt.

Die Funktion **Sprachwechsel** ist für Sprachwechsel des Dōjōs verantwortlich. Die Sprachauswahl geschieht über verschiedene Files. Es existiert für jede Sprache ein eigenes Lookup-File. Dieses verbindet die Majo-Minor-Kennzeichnungen mit den Wav-Files der jeweiligen Sprache. Dadurch muss für einen Sprachwechsel nur der Zeiger auf das Lookup-File geändert werden.

4.4.4 Benötigte Files

Im Header-File des Moduls SD-Karte müssen verschiedene Files definiert werden. Sie müssen auch auf der SD-Karte im richtigen Format vorhanden sein. Die Merken-Liste ist eine normale Textdatei (Endung .txt). Sie sollte leer sein. Des weiteren müssen die Lookup-Files als CSV (Comma separated Value) Dateien vorhanden sein. Sie sollten dem in Abbildung 26 definierten Format entsprechen. Zu beachten ist, dass die X Symbole für zwei stellige Hex-Zahlen stehen. Somit hat es eine Referenz auf ein Kunstwerk pro Zeile. Weiter gilt, dass die Sprachkürzel nur zwei Zeichen beinhalten sollten.

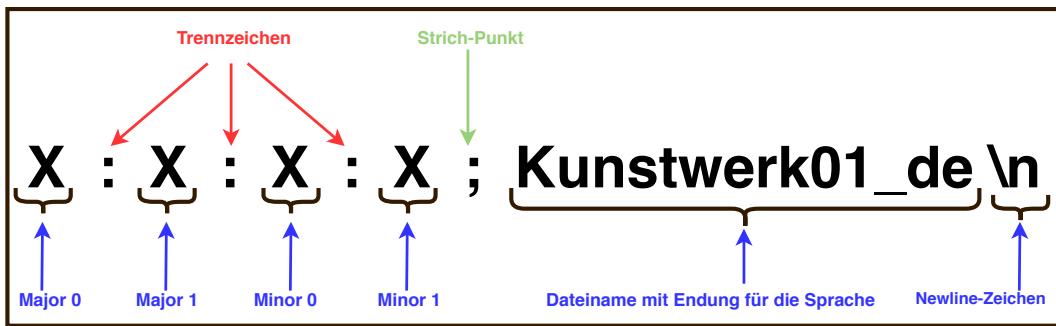


Abbildung 26: Formatdefinition Lookup-File

Die Audiodateien sind im Format WAV Unsigned 8-bit PCM auf der SD-Karte abzulegen. Dabei muss beachtet werden, dass die Sample Frequenz $32kHz$ beträgt und es sich um eine Mono-Kanal Datei handelt.

4.5 Audiowiedergabe durch PWM

Nachdem der Zugriff und die Kommunikation mit der SD-Karte erklärt wurde, kann an dieser Stelle nun die Audiowiedergabe beschrieben werden. Damit ein Audiosignal abgespielt werden kann, sind zwei PWM-Signale Voraussetzung. Dabei ist das erste Signal das invertierte Signal des zweiten und umgekehrt. Abbildung 27 zeigt das Prinzip.

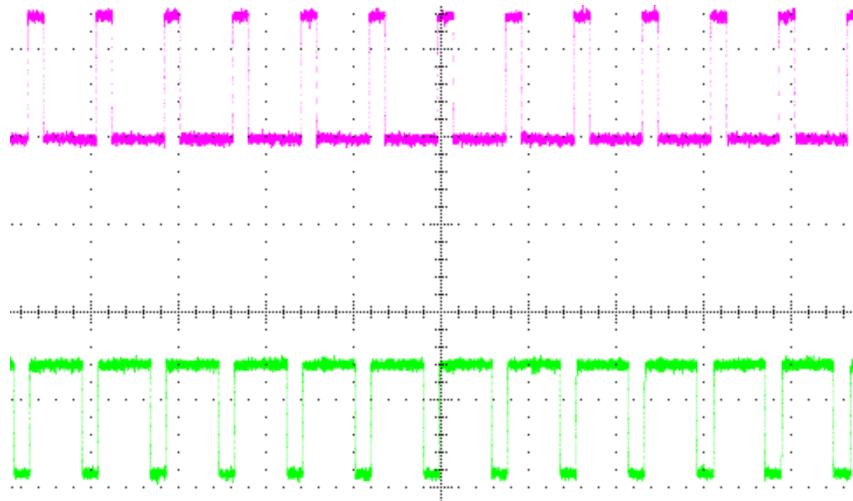


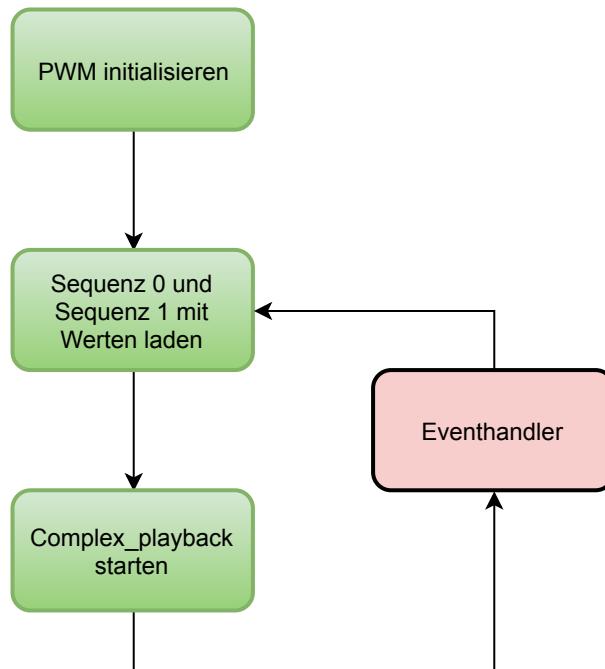
Abbildung 27: PWM-Ausgang des NRF52: Das violette Signal entspricht dem invertierten grünen Signal

Das PWM-Signal wird mit dem NRF52-Controller generiert. NORDIC SEMICONDUCTOR stellt dafür die PWM HAL and driver Bibliothek zur Verfügung. Die genutzten Funktionen sind in der Tabelle 4 aufgelistet. Der Controller bietet vier PWM-Instanzen mit je vier Kanälen. Für die Audioausgabe wurden zwei PWM-Instanzen mit je einem Kanal genutzt.

Beschreibung	Funktion	Argumente
PWM initialisiern	nrf_drv_pwm_init	nrf_drv_pwm_t const *const p_instance nrf_drv_pwm_config_t const *p_config nrf_drv_pwm_handler_t handler
Audio abspielen	nrf_drv_pwm_complex_playback	nrf_drv_pwm_t const *const p_instance nrf_pwm_sequence_t const *p_sequence_0 nrf_pwm_sequence_t const *p_sequence_1 uint16_t playback_count uint32_t flags

Tabelle 4: Funktionen der PWM HAL and driver Bibliothek

Abbildung 28 zeigt den Ablauf des PWM-Unterprogramms. Zuerst findet die Initialisierung statt. Anschliessend können die beiden Sequenzen 0 und 1 generiert werden, in welchen die Daten des Audio-Files abgelegt werden. Die Funktion complex_playback generiert dann aufgrund der Sequenzen das entsprechende PWM-Signal am Ausgang. Nachfolgend werden die einzelnen Schritte noch detaillierter beschrieben.

**Abbildung 28:** Ablauf PWM-Ausgabe

PWM Initialisieren

Bei der PWM-Initialisierung wurde die PWM-Instanz, die Config und der Eventhandler mitgegeben. Die Config des PWM musste anhand der Angaben des Wavefiles generiert werden. Die Werte wurden gemäss Tabelle 5 gewählt.

Parameter	Gewählter Wert	Bemerkung
output_pins	Pin 17	Pin 17 für PWM Modul 1 Pin 27 für PWM Modul 2
irq_priority	APP_IRQ_PRIORITY_LOWEST	Wurde so gewählt, da keine anderen Interruptroutinen vorhanden
base_clock	NRF_PWM_CLK_16MHz	Es wurde die maximale clock Frequenz gewählt
count_mode	NRF_PWM_MODE_UP	PWM zählt bis zu top_value
top_value	500	Siehe Berechnung top_value
load_mode	NRF_PWM_LOAD_COMMON	Wurde gewählt, da alle PWM Kanäle den selben Wert ausgeben
stepp_mode	NRF_PWM_STEP_AUTO	Jedes Sample wird gemäss der Anzahl definierter Wiederholungen abgespielt

Tabelle 5: Gewählte Config Werte

Berechnung top_value

Da die clock Frequenz (in Tabelle 5 base_clock) des PWM Moduls auf $16MHz$ gesetzt wurde, beträgt der top_value für eine Audiodatei mit Abtastfrequenz von $32kHz$ 500. Dies berechnet sich wie folgt:

$$32kHz = \frac{16MHz}{top_value} \Rightarrow top_value = \frac{16MHz}{32kHz} = 500 \quad (4.1)$$

Sequenzen laden

Jetzt können die Sequenzen geladen werden. Mit der Funktion next_Value werden neue Werte in die entsprechenden Sequenzen geladen. Diese Funktion ist im Kapitel 4.4 genauer beschrieben.

Complex playback

Die Funktion generiert anhand der Sequenz ein entsprechendes PWM-Signal. Der Vorteil zu simple playback ist, dass zwei Sequenzen mitgegeben werden können. Wenn die Sequenz 0 fertig abgespielt wurde, startet automatisch die zweite Sequenz und der Eventhandler wird ausgelöst. In dieser Zeit kann die erste Sequenz wieder neu beladen werden. Die Beladung der Sequenzen wird mit dem Eventhandler dieser Funktion gesteuert.

4.6 Lizenzen

Die Lizenztexte sind alle in Englisch veröffentlicht. Dadurch kann das Projektteam nicht haftbar gemacht werden für unwissentlich gemachte Übersetzungsfehler.

SDK

Nordic Semiconductor hat an Anfang eines jeden Files ihren Lizenztext [?] hinterlegt. In diesem ist Beschrieben was mit dem Code gemacht werden darf und was nicht. Einige wichtige Punkte:

- Am Anfang jedes Files muss der Lizenztext stehen.
- Weiterverteilung muss unter dem Selben Copyright erfolgen wie bisher.
- Mann soll nicht ungefragt Werbung machen für Nordic und Co.
- Die Software darf nur für NRF-Chips verwendet werden.

Auch werden alle Verantwortungen für Support, Sicherheit usw. abgestritten. Damit sind sie nicht Haftbar für Schäden, die die Software anrichtet.

Der S132 ist speziell, weil dieser nicht in Source-Form vorliegt. Er ist nur in Binary-Form enthalten. Er hat einen leicht anderen Lizenztext welcher in der SDK neben den Biarys gefunden werden kann. Er besagt zusätzlich, dass der Sourcecode der Softdevice nicht reverse engineered, decompilert, modifiziert und disassembliert werden darf.

5 Validierung

Bei der Validierung werden nachfolgend die Testkonzepte der Software und Hardware erläutert. Die Hardware beinhaltet hierbei hauptsächlich das Testing der Batterie wie auch der Induktions-Ladeschaltung. Das Testkonzept der Software beinhaltet das Testing des PWM-Signals, der Statemachine, des Bluetooths und das Testing der SD-Karte.

5.1 Testkonzept Hardware

Damit ein reibungsloser Betrieb möglich ist, müssen die einzelnen Hardware Komponenten auf Herz und Nieren geprüft werden. Nachfolgend werden die Testverfahren von Batterie, induktiver Übertragung und LC-Filter genauer beschrieben und die jeweiligen Testergebnisse aufgelistet.

Schutzmechanismen Batterie

Die Batterie weist einige integrierte Schutzmechanismen auf, welche alle getestet werden müssen. Als erstes wurde der Tiefentladungsschutz geprüft. Um dies zu testen wurde die Batterie entladen, bis der Schutzmechanismus die Versorgungsspannung abstellt. Da der maximale Entladestrom gleich dem maximalen Ladestrom ist, beträgt dieser gemäss der Berechnung im Kapitel „Li-Ion-Batterie“ (Abschnitt Schutzeinrichtungen) 400mA. Der Widerstand ergibt sich dabei mittels nachfolgender Berechnung 5.1.

$$R = \frac{U}{I} = \frac{3.7V}{0.4A} = 9.25\Omega \quad (5.1)$$

Es wurde somit ein Widerstand der Dimension 9.25Ω angeschlossen.

Während dem Entladevorgang wurde stets die Spannung überwacht, wobei die Spannung von 3.7V auf bis 2.5V absank. Nachdem die 2.5V Schwellenspannung unterschritten wurde, brach der integrierte Batterieschutz die Spannungsversorgung ab. Die Widerstände wurden abgehängt und der gesamte Vorgang wurde kurze Zeit danach mit dem selben Ergebnis wiederholt.

Als nächstes wurde ein Kurzschlusstest durchgeführt. Der Schwellenstrom liegt gemäss Datenblatt [?] bei 4.8A. Durch das $U = R \cdot I$ Gesetz, wurde ein Widerstand der Grösse $700m\Omega$ verwendet, um den Schwellenstrom zu überschreiten. Auch bei diesem Versuch riegelte das PCM den hohen Entladestrom ab und schaltete die Versorgungsspannung der Batterie ab. Dieser Versuch wurde ebenfalls bei selbem Ergebnis wiederholt.

Ladeschaltung der Batterie

Für die Ladeschaltung der Batterie wurde wie im Kapitel ?? bereits erwähnt ein Lade-IC verwendet. Dieser reguliert zuerst die Spannung wobei nach Erreichung des Schwellenwertes von 4.2V der Strom auf 0.4A herunter reguliert wird. Dieser Vorgang wurde während einem gesamten Ladevorgang der Batterie beobachtet und dokumentiert und ist in Abbildung 29 ersichtlich. Hierbei ist wichtig zu erwähnen, dass dieses Testing nicht die induktive Energieübertragung verwendete, sondern der Fokus auf der Funktionalität des Lade-ICs lag und somit das Netzgerät „Power Supply“ der Firma „K. Witmer“ für die Versorgungsspannung verwendet wurde. Aus diesem Grund ist auch ein Strom von $400mA$ wie auch eine Ladezeit von lediglich rund 270 Minuten ($\cong 4.5h$) ersichtlich, was die effektiven Ladewerte mittels induktiver Ladung deutlich unterbietet. Ersichtlich ist nachfolgend die Regulierung der Spannung (blaue Kurve) wie auch die Regulierung des Stromes (rote Kurve).

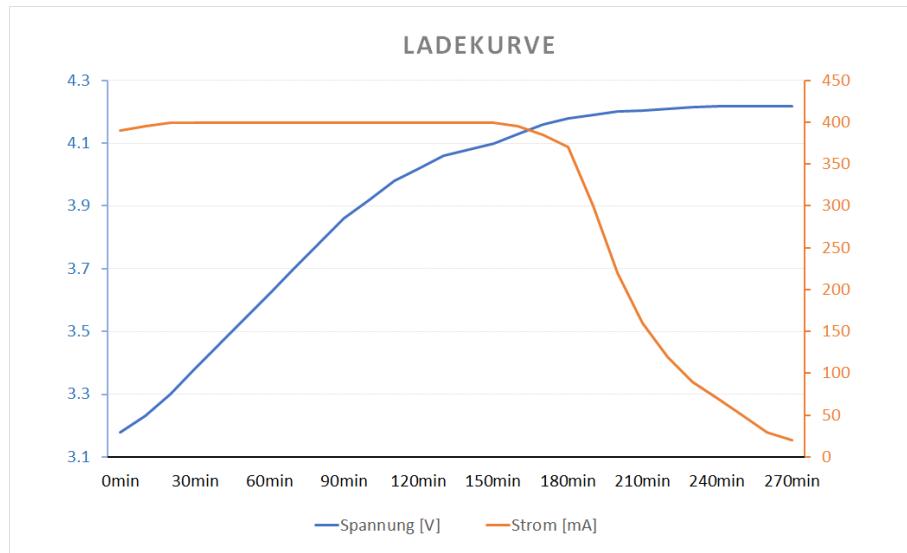


Abbildung 29: Ladekurve Emmerich LI14500

Die oben genannten Vorgänge der Spannungs- und Stromregulierung des Lade-ICs sind in dieser Grafik gut ersichtlich, wobei der Ladevorgang nach dem erreichen von rund 20mA abgebrochen wurde.

Induktive Ladeschaltung

In diesem Abschnitt werden die Ergebnisse der induktiven Ladeschaltung präsentiert. Nachfolgend zeigt Abbildung 30 die Abhängigkeit zwischen induziertem Strom und Spannung in Abhängigkeit zur Distanz „z“ zwischen den Induktionsspulen.

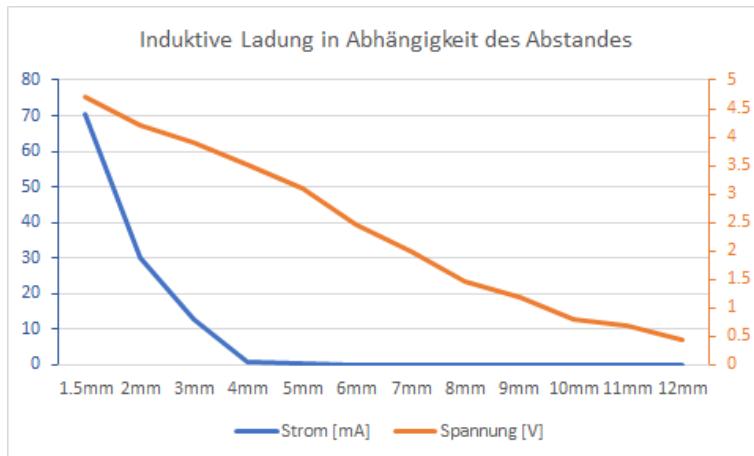


Abbildung 30: Induzierter Strom in Abhängigkeit der Distanz

In der Abbildung ist gut sichtbar, dass die Spannung fast linear zur Distanz abnimmt, hingegen der Ladestrom der Batterie extrem schnell abfällt. Aufgrund dieser Erkenntnis sind wir gezwungen eine möglichst kurze Entfernung zwischen den Spulen einzuhalten um eine möglichst effiziente Energieübertragung zu erreichen. Um dies zu erreichen, haben wir eine Ladestation entworfen, welche die bestmöglichen Induktionswerte garantiert. Der minimale Abstand welcher somit während dem Ladezyklus erreicht werden kann ist abhängig von der Wanddicke des Dōjōs.

Alles in allem beträgt der Abstand somit rund 1.5mm . Bei diesem Abstand resultiert ein an die Batterie gelieferter Strom von maximal $70mA$, wobei die Ladezeit direkt von diesem Strom abhängt. Der gesamte Ladezyklus wird in zwei Etappen unterteilt. Bei der ersten Etappe wird die Batterie mit konstantem Strom geladen. Dies hat zur Folge, dass die Spannung von ihrem Minimalwert $3.3V$ auf den Schwellenwert von $4.2V$ reguliert wird. Um auf die notwendige Ladezeit des ersten Ladezyklus zu kommen, kann die Zeit während einer beliebig grossen Spannungsdifferenz gestoppt werden. Generell gilt: Umso grösser die Spannungsdifferenz, desto genauer die Approximation. Die Endzeit der Spannungsregelung kann linear hochgerechnet werden. Da jedoch der Ladezyklus nach vollständiger Spannungsregelung noch nicht abgeschlossen ist, folgt noch die benötigte Zeit für die Stromregelung. Da diese nicht einfach berechnet werden kann, wurde dieser Prozess im Labor durchgeführt und gestoppt. Diese zwei Ladezyklen zusammen gerechnet ergeben die gesamte Ladezeit. Für den Ladezyklus wurden drei unterschiedlich hohe Ströme definiert, welche die Zeit für den Ladezyklus vorgeben. Sie beinhaltet einen schonenden Ladezyklus ($I_{charge} = 30mA$), einen normalen Ladezyklus ($I_2 = 50mA$) und einen schnellen Ladezyklus ($I_3 = 70mA$). Die Erreichung dieser drei Ladestufen wird in dieser Arbeit nicht weiter definiert, könnte jedoch durch drei unterschiedliche Spannungsstufen, welche die Primärspule versorgen, reguliert werden. Nachfolgend sind die Berechnungen für die jeweiligen Ladezeiten $t_{voltage}$ und $t_{current}$ der drei Stufen, wie auch die gesamte Ladezeit t_{tot} ersichtlich.

Ladezyklus schonend

$$t_{voltage} = \frac{1.1V}{\left(\frac{0.006V}{10\text{Minuten}}\right)} = 1'833.3\text{Minuten} = 30h\ 33min \quad (5.2)$$

$$t_{current} = 2h\ 7min \quad (5.3)$$

Die gesamte Ladezeit ergibt sich durch Addition der Berechnungen 5.2 und 5.3.

$$t_{tot} = t_{slow_{voltage}} + t_{current} = 32h\ 27min \sim 32.5h \quad (5.4)$$

Es ergibt sich somit eine gesamte Ladezeit von 32.5h. Dieser Zyklus braucht sehr lange um die Batterie zu laden, er ist aber auch der schonendste. Es wird empfohlen, ihn bei Ruhetagen anzuwenden.

Ladezyklus normal

$$t_{voltage} = \frac{1.1V}{\left(\frac{0.0016V}{10\text{Minuten}}\right)} = 687.5\text{Minuten} = 11h\ 28min \quad (5.5)$$

$$t_{current} = 46min \quad (5.6)$$

Die gesamte Ladezeit des normalen Ladezyklus ist nachfolgend in 5.7 ersichtlich.

$$t_{tot} = t_{voltage} + t_{current} = 12h46min \sim 13h \quad (5.7)$$

Der normale Ladezyklus benötigt eine Zeit von rund 13h. Diese Zeit ist schon massiv kürzer als bei der schonenden Ladung. Sie kann für den Ladevorgang während der Nacht oder bei wenigen

Besuchern verwendet werden.

Ladezyklus schnell

$$t_{voltage} = \frac{1.1V}{\left(\frac{0.0265V}{10\text{Minuten}}\right)} = 415.1\text{Minuten} = 6h\ 55min \quad (5.8)$$

$$t_{current} = 1h\ 18min \quad (5.9)$$

Die gesamte Ladezeit gemäss Berechnung 5.10 ergibt sich durch Addition der Berechnungen 5.8 und 5.9.

$$t_{tot} = t_{voltage} + t_{current} = 7h\ 41min \sim 8h \quad (5.10)$$

Der Ladezyklus mit einem Versorgungsstrom von $70mA$ benötigt eine Ladezeit von knapp 9h und ist somit der schnellste Zyklus. Es wird jedoch empfohlen, dass dieser Zyklus nur bei grossen Besucherzahlen verwendet wird.

Allgemein ist die kurze Ladezeit von lediglich rund 8 Stunden beim schnellen Ladezyklus (Berechnung 5.10) augenfällig. Es gilt jedoch zu beachten, dass bei einem Strom von $70mA$ die Spule sich mehr erwärmt als beim Ladezyklus mit $50mA$. Es wird somit im Allgemeinen vorgeschlagen, dass bei genügend Zeit der normale Zyklus mit $50mA$ gestartet wird. Dieser ist zwar um rund 5 Stunden langsamer, jedoch nachhaltiger für die eingebauten Materialien.

Inbetriebnahme der induktiven Ladeschaltung

Der erste Prototyp der Pulsschaltung für die Induktion wurde mithilfe eines NE555 realisiert, wie sie in Abbildung ?? zu sehen ist. Dabei wurden als Widerstände Potentiometer eingebaut, mit welchen die Resonanzfrequenz des LC-Gliedes gesucht werden sollte. Der allererste Versuch ergab eine Übertragung von $5mA$ bei einer Taktfrequenz von $10kHz$. Es stellte sich relativ schnell heraus, dass die Taktfrequenz viel zu tief war. Das konnte durch die Veränderung der Taktfrequenz der Pulsschaltung festgestellt werden. Bei höherer Taktfrequenz wurde die Übertragung besser. Da die Taktfrequenz der NE555-Prototypenschaltung aufgrund der Wahl des Potentiometers auf $10kHz$ begrenzt war, wurde nun in weiteren Schritten mit dem Frequenzgenerator der 2n3055 Leistungstransistor angesteuert. Des Weiteren wurde das C aus dem LC-Glied ausgebaut, um so das spezifische Maximum der Spule L zu finden. So wurde nun mit konstanter Eingangsspannung und variabler Taktfrequenz die Spule getestet. Mit höher werdender Frequenz wurde die Übertragung besser. Um diese noch weiter zu verbessern empfiehlt sich bei der Taktfrequenz einen Duty Cycle von möglichst 50% zu erzielen. Es erwies sich, dass bei der Taktfrequenz von $93kHz$ die beste Übertragung zustande kam. Bei höheren Frequenzen war diese wieder rückläufig. Da nun die optimale Frequenz gefunden und die Induktivität der Spule bekannt war ($14.9\mu H$), konnte der ideale Kopplungskondensator berechnet werden. Dies ergab, unter Berücksichtigung der E-Reihen, einen $220nF$ Kondensator als ideal. Damit die Spule nicht zu heiss werden und der Isolierlack nicht zu schmelzen beginnt, wurde der Strombegrenzungswiderstand des 2n2222 mit 2Ω definiert. Die Berechnung hierzu ist im oberen Text zu finden. Somit sind alle benötigten Bauteile der Schaltung aus Abbildung 31 bekannt.

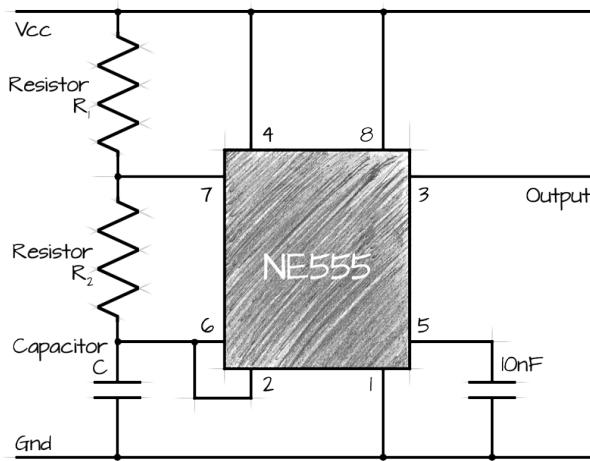


Abbildung 31: Verwendete Timerschaltung NE555 als Pulssquelle

$$C = 1nF$$

$$R_1 = 510\Omega$$

$$R_2 = 7.5k\Omega$$

Mit diesen Werten konnte nach dem Gleichrichter, bei einer Übertragungsdistanz von einer Standard-Platinen dicke, eine Leerlaufspannung von ca. 17V und ein Kurzschlussstrom ca. 300mA gemessen werden. Es gilt jedoch zu beachten, dass diese Werte ohne Last gemessen wurden.

Da nun die Induktionsstufe funktioniert wird diese nun um die Ladeschaltung erweitert. Hier ergab sich folgendes Problem:

Die Eingangsspannung des Lade-IC's darf nicht mehr als 6.5 Volt betragen. Da die Receiverspule sich im inneren des Dojōs befindet und dadurch nur eine sehr geringe Baugröße besitzen darf, kann diese zwangsläufig keine grossen Leistungen erbringen. Dies bedeutet, dass akzeptabel Ladeströme nur mit genügend hohen Spannungen erzeugt werden können.

Um dies zu erzielen wurde versucht, einen Spannungsregler einzubauen, um den übertragenen Strom beizubehalten, jedoch die Spannung zu begrenzen. Das Ergebnis davon war jedoch ziemlich bescheiden. Der so entstehende Energieverlust ist ziemlich gross und ausserdem kam es vereinzelt vor, dass der Lade-IC trotzdem kaputt ging, da der Spannungsregler nicht sauber regelte.

Die Ideale Lösung wäre natürlich einen geeigneteren Lade-IC zu nehmen. Da jedoch die Validierung desjenigen bereits durchgeführt wurde und nicht genügend Zeit vorhanden war diesen erneut zu bestellen wurde sich entschieden die Transceiver Schaltung ein wenig anzupassen.

Dabei gibt es folgende Möglichkeiten. Man kann die Eingangsspannung absenken. Dies beeinflusst jedoch auch den übertragenen Strom exponentiell, da einerseits eingangsseitig mit weniger Spannung auch weniger Strom durch die Spule fliesst und andererseits auch der Leistungstransistor nicht sauber durchgesteuert wird da die Pulsschaltung ebenfalls an der gleichen Versorgungsspannung hängt und somit sein Pulssignal beeinflusst wird. Die zweite Möglichkeit ist die Taktfrequenz verringern. Dadurch wird die Induzierte Spannung ebenfalls abgesengt was ebenfalls den Strom exponentiell beeinflusst. Die dritte Möglichkeit wäre die Übertragungsdistanz zu verringern. Da aber sichergestellt werden wollte, dass der Dojō-Boden dennoch eine gewisse Stabilität aufweisen sollte wurde diese Distanz der Platinendicke beihalten.

Somit wurde die Schaltung lange ausgetestet um die beste Kombination zu erzielen. Das beste Resultat ergab die Kombination zwischen einer Taktfrequenz von ca 80kHz und einer Eingangs Spannung von 8.3V. Dies lässt sich damit erklären das leichtes Absenken der beiden Möglichkeiten den Strom nicht so stark beeinflusst. Des Weiteren wurde der Strombegrenzungswiderstand auf 1.5Ω verringert da sich die Eingangs Spannung verringert hat und ausserdem der Stromfluss durch die Spule damit erhöht wurde.

Diese Anpassungen fielen ebenfalls zu Gunsten des Energiemanagements, jedoch ist dies eine stabile Variante für die ersten Prototypen. Durch die Anpassungen wird an der Transceiver Spule die überschüssige Energie in Form von Wärme abgegeben. Die Tests ergaben, dass so die Batterie bei einer Spannung von 4.7V mit 70mA geladen werden können. Es ist ebenfalls möglich kurzzeitig mit einem höheren Strom zu laden, wenn die Eingangs Spannung nur geringfügig erhöht wird. (bis zu 120mA). Jedoch ist die Hitzeentwicklung hierbei so gross, dass nach einer gewissen Zeit die Übertragung zusammenbricht. Deshalb ist zu empfehlen die angegebenen Werte nicht zu überschreiten um eine dauernde Funktionalität zu gewährleisten.

Die Werte der oben beschriebenen angepassten Pulsschaltung werden ebenfalls wie in Abbildung 31 aufgebaut und sind folgende:

C: $1nF$

R1: 200Ω

R2: $9 k\Omega$

LC-Filter

Die Validierung des LC-Filters wurde mittels eines $500Hz$ Sinus Signals erledigt. Dabei wurde sowohl ohne Filter wie auch mit Filter eine KO Bild aufgenommen. Abbildung ?? zeigt das Signal ohne Filter. Rot angezeigt ist die FFT des Signals. In Abbildung ?? ist das Signal und die FFT davon mit dem LC-Filter.

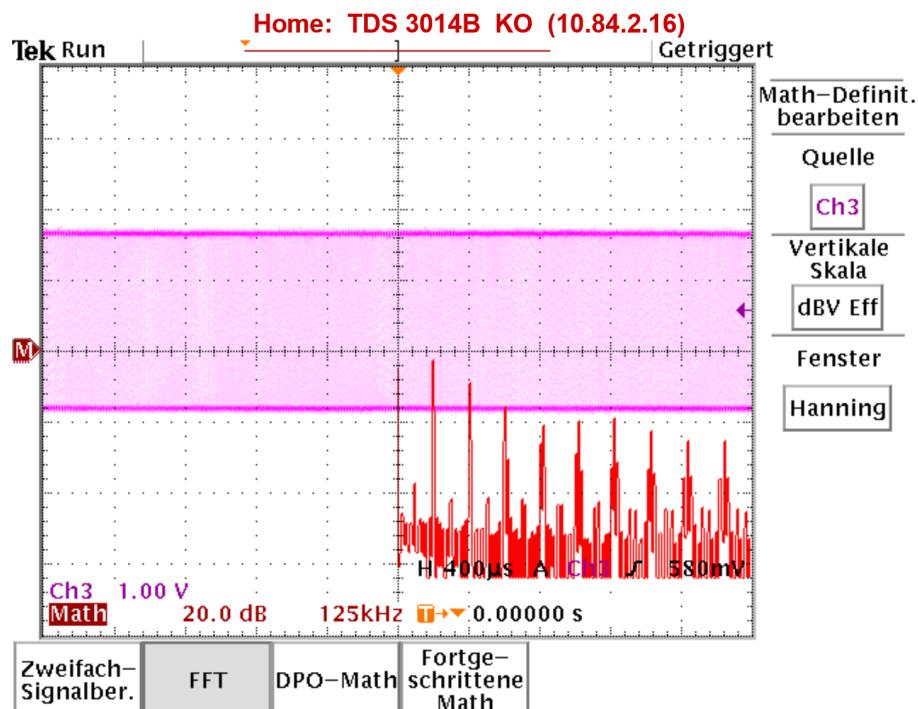


Abbildung 32: 500Hz Sinus Signal und FFT ohne LC-Filter

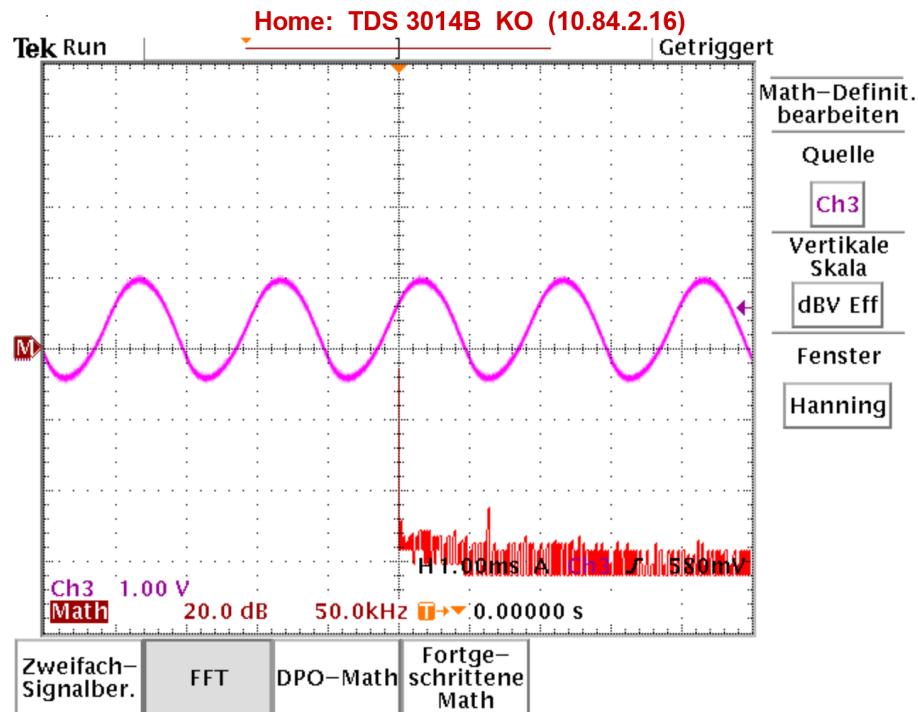


Abbildung 33: 500Hz Sinus Signal und FFT mit LC-Filter

Durch die beiden KO Aufnahmen wurde ersichtlich, dass durch den Filter ein deutlich besseres und weniger verrausches Signal rekonstruiert wird. Somit ist der LC-Filter notwendig, da mit ihm die Schaltfrequenzen des PWM Ausgangs unterdrückt werden.

5.2 Testkonzept Software

Für einen einwandfreien Betrieb ist das Testing der Software einer der wichtigsten Bestandteile der Validierung. Dieses Kapitel wird in die Abschnitte „PWM Signal“, „Verhalten der Statemachine“, „Kommunikation zwischen Bluetooth und Beacon“ wie auch „Funktionalität der SD-Karte“ geteilt.

5.2.1 PWM Signal

Das PWM Signal wurde mit dem Gleichen Signal wie das LC-Filter validiert. Dabei wurden sowohl die beiden PWM Kanäle mit dem KO aufgezeichnet wie in ?? zu sehen ist, sowie das gefilterte Signal.

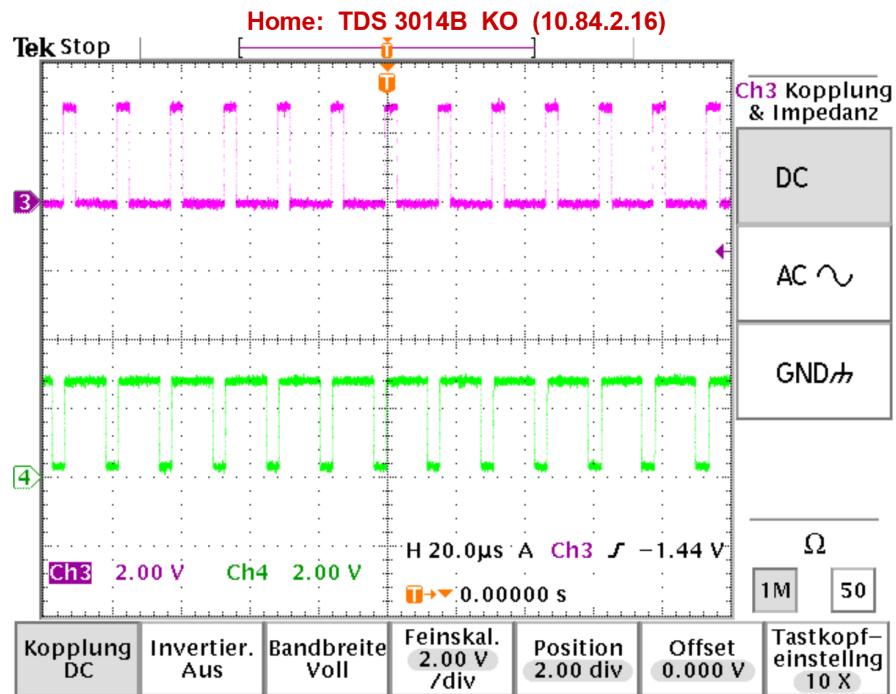
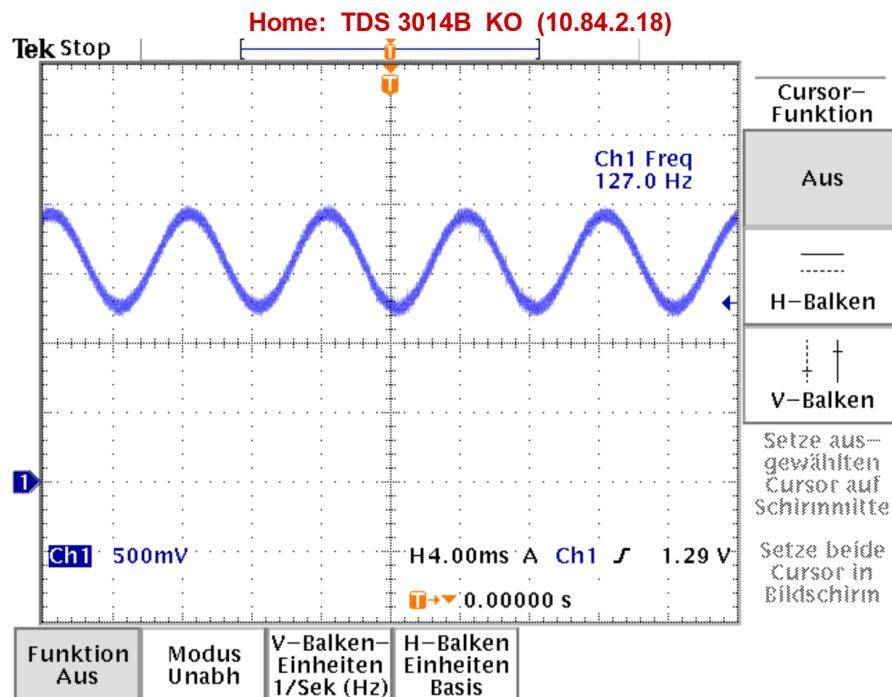


Abbildung 34: PWM Signal beider PWM Ausgänge

Aus 34 ist ersichtlich, dass die beiden PWM Signale zwar invertiert zu einander stehen aber auch zeitlich (knapp knapp $4\mu s$) verschoben sind. Diese Zeitverschiebung hat jedoch keinen hörbaren Effekt und kann somit vernachlässigt werden.

Um die Einstellungen des PWM-Moduls aus 4.5 zu validieren, wurde ein $500Hz$ Sinus Signal mit dem KO abgebildet. Das entstandene Resultat ist in nachfolgender Abbildung 35 ersichtlich.

Abbildung 35: PWM auf $32kHz$ eingestellt

Es ist augenfällig, dass die Frequenz des Signals nun nur 127Hz anstatt den angelegten 500Hz beträgt, was einem Faktor von rund 4 entspricht. Um die gewünschten 500Hz zu erreichen, wurde der top value aus Kapitel 4.5 von 500 auf 125 herunter gesetzt. Das entstandene Resultat ist in Abbildung 36 ersichtlich.

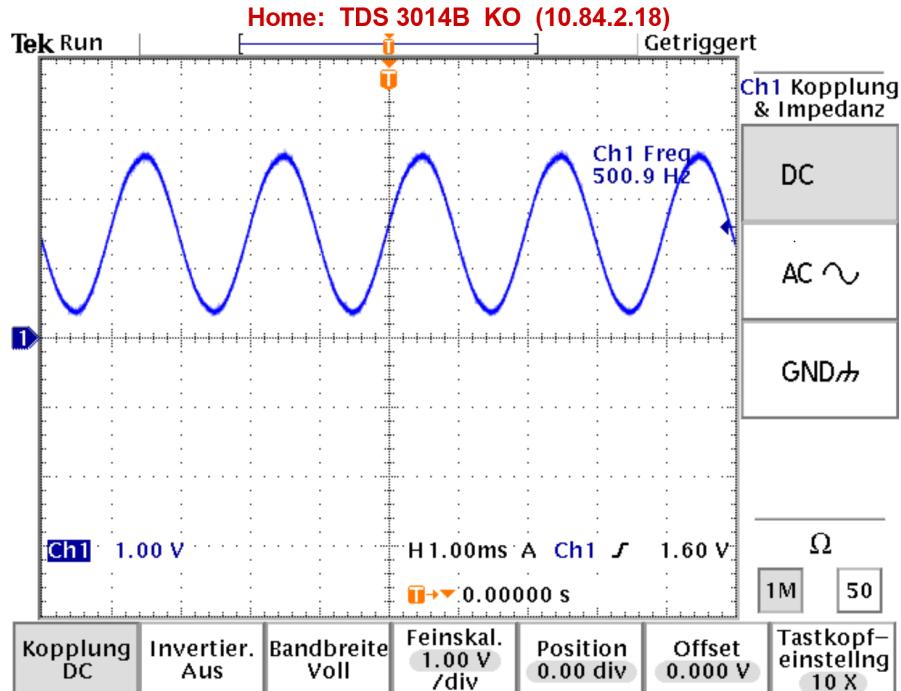


Abbildung 36: PWM auf 128kHz eingestellt

Es ist ersichtlich, dass nun das Sinus Signal mit 500Hz schwingt. Da die Berechnungen einen top value von 500 ergeben, deutet dies auf einen Fehler im Code, in der Audiodatei oder auf einen Überlegungsfehler hin. Es wurde herausgefunden, dass die Tests fälschlicherweise mit einer stereo Audiodatei durchgeführt wurden. Da jedoch nur ein Kanal angesteuert wird, ergibt sich ein Faktor 2 unterschied zu den Berechnungen. Dieser Test wurde folglich noch einmal mit einer mono Audiodatei durchgeführt, wobei der top value gemäss unseren Erwartungen um den Faktor 2 vom berechneten Wert abweicht.

5.2.2 Statemachine

Die Statemachine wurde auf ihr Verhalten hin getestet. Dabei wurden alle States mithilfe von Flags einmal erzwungen, wobei folgende States nicht korrekt funktionierten:

- ADC-Batterie
- Shutdown
- Merken Liste Löschen
- Charge

Es gilt anzumerken, dass diese „fehlerhaften States“ in der Statemachiene eigentlich funktionieren, jedoch ihre Aufgabe nicht erfüllen können, da sie aus zeitlichen Gründen nicht implementiert wurden. Die Implementierung dieser States würden bei der Weiterverfolgung dieses Projektes als zusätzlicher Arbeitsschritt implementiert werden.

5.2.3 Bluetooth

Das Bluetooth Modul wurde mit verschiedenen Beacons getestet. Dabei ist aufgefallen, dass bei der gleichzeitigen Verwendung von mehreren Beacons, welche mit unterschiedlichen Sendeintervallen senden, dies zu grösseren Problemen führt. Es dominieren die Beacons mit schnellerer Sendefrequenz, wobei die langsameren überblendet werden. Aus diesem Grund ist es zu empfehlen, dass die Abstände der Beacons gemäss Abbildung 37 angewendet werden. Die Radien können für jeden Beacon einzeln konfiguriert werden. Die grössze des Radius kann zwischen 0m und 40m variiieren. Default mässig ist dieser Radius auf 3m eingestellt.

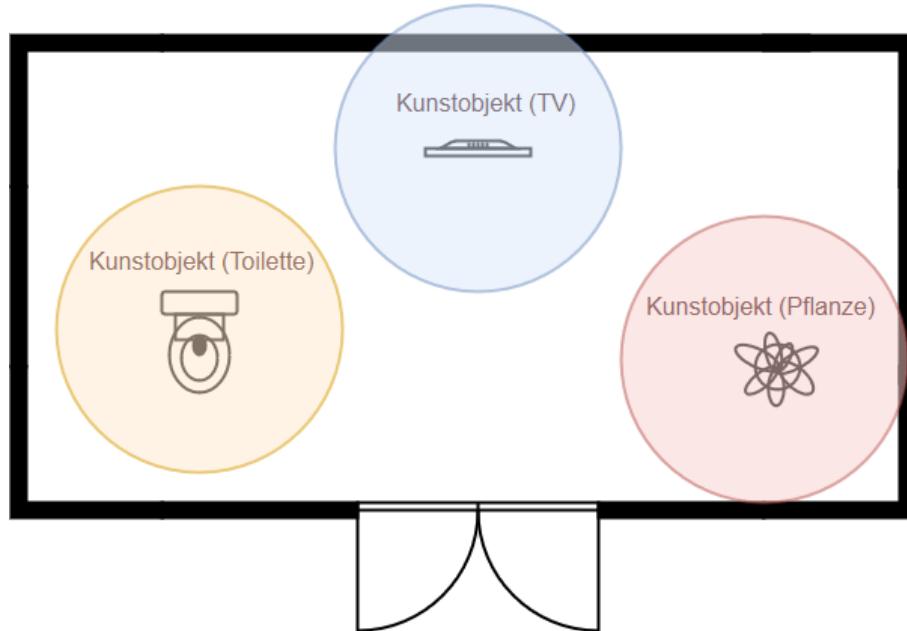


Abbildung 37: Überlappung der Referenz-Felder

5.2.4 SD-Karte

Das SD-Karten Modul hat alle Anforderungen erfüllt. Das lesen und schreiben ist problemlos möglich und funktionierte ohne weiteres. Einziger Makel weist die Funktion „Merken“ (welche vom „Like Button“ aufgerufen wird) auf. Hierbei ist es nicht möglich, ein gemerktes Kunstobjekt wieder zu löschen. Dafür ist es notwendig die SD-Karte aus dem Gerät zu nehmen und sie neu zu konfigurieren. Zudem kann man ein Kunstobjekt gleich mehrmals in die selbe „Merkliste“ einfügen, was zwar keine Probleme im Gerät auslöst, aber vom Handlich her nicht sehr schön ist.

6 Schlusswort

Dieses Kapitel besteht noch aus einzelnen Argumenten und Abschnitten und ist noch nicht miteinander verknüpft

a) Was läuft? Was wurde im Projekt erreicht? Welche in der Aufgabenstellung gestellten Anforderungen wurden realisiert? Wo wurden diese übertroffen? Der Dōjō wurde ursprünglich in einem kleinen hohlen Stab konzeptioniert. In dieser ersten Entwicklungsphase wurde ein Prototyp realisiert, welcher optisch an und für sich nichts mit dem Dōjō zu tun hat, sondern lediglich die Funktionalität der Elektronik im Vordergrund steht. Der grosszügige Aufbau hat den Vorteil, dass sich einzelne Komponenten wie z.B. die Ladeschaltung für die Energiespeicherung, Mikrocontroller auf einem Evaluation Board oder der Knochenschallaktor als einzelne Teilsystem getestet und in einem weiteren Schritt zu einem Gesamtsystem zusammenbauen lassen. Diese Vorgehensweise ermöglichte eine saubere und effiziente Arbeitstrennung. Nachdem alle Komponenten dimensioniert und getestet wurden, konnte ein fertiges in den Dojo passendes PCB Design erstellt werden, welches in einem weiteren Schritt in den Dōjō eingebaut werden könnte.

Das Gesamtsystem umfasst hierbei die Hardwarekomponenten wie induktive Ladeschaltung, Batterie, Microcontroller, sowie notwendige Komponenten für die Audioausgabe. Die Software beinhaltet alle programmierbaren Ebenen wie die State Machine, aber auch die Audio PWM Ausgabe, das Bluetooth und die Programmierung des Nordic SDK. Die Erkennung der Bluetooth-Beacons funktioniert einwandfrei, solange nicht mehr als 10 Beacons in unmittelbarer Nähe sind. Das Detektieren erfolgt über die Software. Mit einem geeigneten Algorithmus wird jeweils das Beacon mit dem stärksten Signal als das „korrekte“ Kunstobjekt identifiziert. Das integrierte Bluetooth-Modul im Mikrocontroller ermöglicht das Empfangen der entsprechenden Beacon-ID und kann diese über eine geeignete Funktion in eine Hexadezimal-Zahl wandeln. Anschliessend wird mit einer ID-Liste auf der SD-Karte verglichen und bei Übereinstimmung das entsprechende Audio-File abgespielt.

Die induktive Ladung funktioniert einwandfrei und ermöglicht eine Stromübertragung von maximal 70mA. Für nachfolgende Weiterentwicklungen der Ladestufe empfiehlt es sich, einen Lade-IC zu verwenden, welcher eine grössere Eingangsspannung verarbeiten kann. Die Verwendung eines LT1512 hat den Vorteil, dass die Eingangsspannung zwischen 2.7V – 25V variieren kann. Dieser grosse Spannungsbereich hat jedoch zur Folge, dass die Grösse des Bauteils ($L \times B \times H$) mit $(6.2 \times 5 \times 1.7)mm$ eher gross ausfällt. Trotz seiner grossen Bauform überwiegt hierbei aber der Vorteil mit der hohen Versorgungsspannung.

Des Weiteren würde es sich lohnen die Spulen zu optimieren. Die eingebauten Flachspulen sind zwar platzsparend, kommen jedoch aufgrund ihrer Bauform ziemlich schnell an ihr Leistungsmaximum. Hier würden sich solche Spulen sich empfehlen, wie sie z.B. auch in modernen elektrischen Zahnbürsten verwendet werden. Hierbei handelt es sich um zwei unterschiedlich grosse Ringspulen, von welchem die kleinere Tranceiver-Spule in der Ladeschaltung in die grössere Receiverspule im Dojo gesteckt wird. Diese Ringspulen wären zwar ein wenig grösser, doch durch die grössere Bauform könnte die Ladeschaltung mit grösseren Strömen betrieben und damit auch der Dojo noch schneller geladen werden.

Mit der Verwendung des komplexen und leistungsfähigen Mikrocontrollers NRF52832 können alle Prozesse zentral gesteuert werden und dadurch entfallen weitere Controller. Ebenfalls kann dadurch der Energieverbrauch gesenkt werden und die Verwendung eines zusätzlichen Bluetooth-Moduls entfällt.

b) Was läuft nicht? Wo besteht Verbesserungsbedarf? Was konnte nicht realisiert werden? Hier ist eine kurze Angabe der Gründe sinnvoll. Mit dem Verzicht einer USB-Schnittstelle besteht keine Möglichkeit die SD-Karte ohne weiteres zu aktualisieren oder den

Inhalt im Allgemeinen zu verändern. Daher sieht das Konzept vor, die SD-Karte manuell zu entnehmen und dann über einen extra dafür vorgesehenen SD-Karten-Hub zu aktualisieren. Dadurch lassen sich mehrere SD-Karten gleichzeitig bearbeiten und führt damit zu einer signifikanten Zeiteinsparung für die Betreiber. Weiter wird die SD-Karte im Gehäuse auch so angebracht, dass eine einfache Entnahme ohne Werkzeug oder Ähnliches möglich ist. Theoretisch wäre es möglich gewesen eine USB-Schnittstelle zu implementieren. Allerdings muss man sich dann auch unmittelbar die Frage stellen ob es dann noch Sinn macht, den Dōjō induktiv zu laden. Beide Eigenschaften sind unmittelbar miteinander verknüpft. Die Entscheidung fiel dann auf die induktive Ladung der Anwendung.

c) Optimierungs- / Weiterentwicklungsmöglichkeiten: Welche Optimierungsmöglichkeiten bestehen? Was könnte wie besser gemacht werden? Was muss bei einer Weiterentwicklung bedacht werden? Eine Weiterentwicklung des aktuellen Konzepts ist durchaus sinnvoll. Nachdem die Basis des Dōjō umgesetzt und getestet wurde, ist viel technisches „Know-How“ vorhanden. Mögliche Weiterentwicklungsmöglichkeiten sind nachfolgend aufgelistet.

Einerseits kann eine Aktualisierung der SD-Karte über ein drahtloses Netzwerk wie z.B Bluetooth oder W-LAN umgesetzt werden. Dadurch liesse sich jeglicher Kontakt mit der SD-Karte durch einen Mitarbeiter vermeiden. Fraglich ist an dieser Stelle jedoch in welchem Umfang Aktualisierungen vorgenommen werden, da die drahtlose Datenübertragung nicht unbegrenzt Daten übermitteln kann.

Eine weitere Möglichkeit ist die Umsetzung eines integrierten Tickets. Dieses wurde im Verlaufe des Berichts weder erläutert noch umgesetzt. Die automatische Zutrittsberechtigung zu verschiedenen Ausstellungsbereiche, ermöglicht einen individuellen Museumsbesuch. Die Software dazu, ist jedoch im Moment noch nicht für dieses Szenario ausgelegt und benötigt bei einer Implementation noch einiges an Aufwand.

Ein weiterer Punkt welcher mit weitaus geringerem Aufwand implementiert werden könnte, ist die Signalisation mittels LED Leiste. Im Moment wird noch eine LED ohne rgb angesteuert. Das Ansprechen einer RGB Leuchtdiode ist softwaretechnisch jedoch programmiert und müsste für die Funktionalität nur noch im Dojo eingebaut werden.

7 Ehrlichkeitserklärung

Der Projektleiter bestätigt mit der Unterschrift, dass der Bericht selbst verfasst und alle Quellen sauber und korrekt deklariert wurden.

Ort, Datum

Unterschrift Projektleiter

8 Literaturverzeichnis

9 Abbildungsverzeichnis

1	Dōjō als Modell	5
2	Teilsysteme des Dōjōs	6
3	Anwendungsablauf des Dōjōs	7
4	Sprachauswahl mittels Bluetooth-Beacon	8
5	Verwendete Timerschaltung als Pulsquelle	10
6	Laderegelung Li-Ion Akku	13
7	Prototyp Ladestation	14
8	Mikrocontroller NRF52832 [?] [?]	14
9	Schema der LC-Filterstufe	15
10	Verstärkerstufe [?]	16
11	Knochenschallaktor [?]	17
12	Software:Layers	18
13	Software:Pollend	19
14	Statemachine-Diagramm	20
15	Statemachine: lookup	21
16	Statemachine: Special Beacon	22
17	Statemachine: Change Language	23
18	Statemachine: New Beacon	24
19	Statemachine: Wait	25
20	Statemachine: ADC Battery	26
21	Statemachine: Charge	27
22	PacketPayload Header [?]	29
23	PacketPayload Payload [?]	30
24	Software:Definierte MM	31
25	Software:MM Vergabe	32
26	Formatdefinition Lookup-File	34
27	PWM-Ausgang des NRF52	34
28	Ablauf PWM-Ausgabe	35
29	Ladekurve Emmerich LI14500	39
30	Induzierter Strom in Abhängigkeit der Distanz	39
31	Verwendete Timerschaltung NE555 als Pulsquelle	42
32	500Hz Sinus Signal und FFT ohne LC-Filter	43
33	500Hz Sinus Signal und FFT mit LC-Filter	44

34	PWM Signal beider PWM Ausgänge	45
35	PWM auf $32kHz$ eingestellt	45
36	PWM auf $128kHz$ eingestellt	46
37	Überlappung der Referenz-Felder	47

10 Tabellenverzeichnis

1	PDU Type [?]	30
2	AD Data Type [?]	30
3	Flags [?]	31
4	Funktionen der PWM HAL and driver Bibliothek	35
5	Gewählte Config Werte	36

A Anhang

A.1 Messresultate

bcsjdbcsjbvsvb