Fachhochschule
Nordwestschweiz

Fachbericht

Team 1

Projekt 4

Fachhochschule Nordwestschweiz FHNW

10. Juni 2018

Studiengang:	Elektro- und Informationstechnik EIT
Auftraggeber/in:	Prof. Hans Gysin Jana Kalbermatter
Fachexperten:	Matthias Meier Prof. Dr. Pascal Schleuniger Pascal Buchschacher Dr. Roswitha Dubach Dr. Anita Gertiser Bonnie Domenghino
Projektteam:	Adrian Annaheim Benjamin Ebeling Jonas Rosenmund Michael Schwarz Samuel Wey Andres Minder

Inhaltsverzeichnis

1 Gesamtsystem **3**

2 Software **4**
2.1 Konzept . . . 4
2.2 Datenverwaltung . . . 4
2.3 Auslesen, Schreiben . . . 5
2.4 Validierung . . . 5

3 USB **7**
3.1 Technische Grundlagen . . . 7
3.2 Konzept . . . 7
3.3 USB zu UART . . . 8
3.4 USB zu SDIO . . . 8
3.5 Validierung UART-Pfad . . . 9
3.6 Validierung SDIO-Pfad . . . 9

4 Audioausgabe **10**
4.1 Technische Grundlagen . . . 10
4.2 Konzept . . . 10
4.3 Hardware . . . 11
4.4 Firmware . . . 12
4.5 Validierung . . . 13

5 Bluetooth **14**
5.1 Technische Grundlagen . . . 14
5.2 IBeacons auslesen und identifizieren . . . 14
5.2.1 Technische Daten . . . 14
5.2.2 Bluetooth-Konfiguration . . . 15
5.3 Beacons im Museum . . . 15
5.3.1 Minew E7 . . . 15
5.3.2 Protokollbeschreibung . . . 15
5.3.3 Receiver Signal Strength Indicator (RSSI) . . . 15
5.4 Konzept . . . 16
5.5 Hardware . . . 16
5.6 Firmware . . . 16
5.7 Validierung . . . 16

6 Firmware **17**
6.1 Konzept . . . 17
6.2 Statemachine . . . 17
6.3 Datenverwaltung . . . 17
6.4 Validierung . . . 17

7 Verzeichnisse **18**

8 Anhang **19**

1 Gesamtsystem

Abbildung 1.1 zeigt das grobe Gesamtsystem des Dojos. Die einzelnen Kapitel dieses Berichts orientieren sich an den Blöcken und deren Funktion im Gesamtsystem. In Kapitel **2 Software** wird die Computersoftware erklärt, welche verwendet wird, um den Dojo zu konfigurieren. Das Kapitel **?? ??** befasst sich mit dem Akku, der Lade -schaltung und Überwachung, sowie der Spannungsversorgung. Im Kapitel **3 USB** wird erläutert, wie über die USB-Schnittstelle mit dem Mikrocontroller kommuniziert wird und wie Audiofiles auf die SD-Karte übertragen werden. Das Kapitel **4 Audioausgabe** behandelt die Verarbeitung und Ausgabe der Audiofiles. Das nächste Kapitel **5 Bluetooth** befasst sich mit dem Bluetooth. Das heisst, mit den Beacons zur Erkennung der Kunstobjekte und dem Bluetoothmodul im Dojo für den Empfang. Wie die Firmware als Gesamtsystem funktioniert, erfährt man in Kapitel **6 Firmware** detailliert. Die Validierung der einzelnen Blöcke wird im jeweiligen Kapitel abgehandelt.

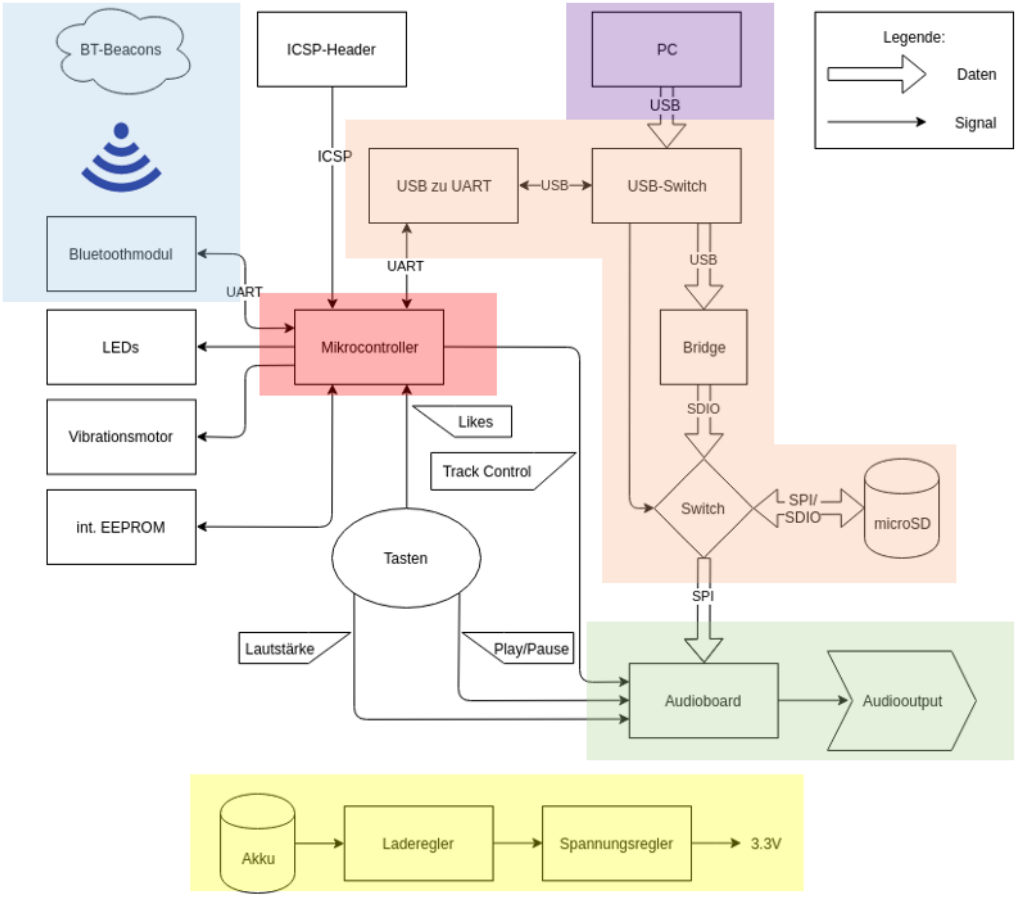

Abbildung 1.1: Blockschaltbild Gesamtsystem

Software	Energieversorgung	USB	Audioausgabe	Bluetooth	Firmware

Tabelle 1.1: Legende Gesamtsystem

2 Software

Im Kapitel Software wird erläutert, wie die Audiodateien auf das Dojo gelangen und die Korrespondenztabelle angepasst werden kann. Es wird erklärt, wie die Likes ausgewertet und verarbeitet werden können.

2.1 Konzept

In diesem unter Kapitel wir das erarbeitete Konzept für die Datenübertragung von einem Computer auf das Dojo dargelegt.

2.2 Datenverwaltung

Alle Ausstellungsobjekte sind in einer Textdatei aufgelistet, in der vermerkt wird, welche Audio- und Textdatei zu welchem Beacon gehört. Die Form dieser Inventardatei ist in Abbildung 2.1 zu sehen.
Zudem wird in einer weiteren Textdatei ein Preset definiert, mit der die SD-Karte beschrieben werden kann. Die Form dieser Presetdatei ist in Abbildung 2.2 zu sehen.

```
exhibition romans{
    233(
        de  roemischerhelm.mp3  roemischerhelm.tex
        en  romanhelmet.mp3     romanhelmet.tex
    )
    10(
        de  becher_rom.mp3      becher_rom.tex
        en  cup_rome.mp3        cup_rome.tex
    )
}

exhibition middle_ages{
    15(
        de  mittelalterlicher_schaedel.mp3  mittelalterlicher_schaedel.tex
        en  skull_medieval.mp3              skull_medieval.tex
    )
    11(
        de  morgenstern.mp3         morgenstern.tex
        en  primitive_weapon.mp3    primitive_weapon.tex
    )
}
```

Abbildung 2.1: Inventardatei für eine Testausstellung

Der CLI-Befehl `makepackage` nimmt die Inventardatei und Presetdatei und generiert ein Ordner in dem alle Audiodateien die in der Inventardatei aufgelisteten sind sich befinden. Die Dateien

```
languages: de, en
exhibitions: romans, middle_ages
```

Abbildung 2.2: Presetdatei für eine Testausstellung

Anzahl gespeicherter Beacons	Sprachcode
ID Beacon1	Like Beacon1, Zutritt Beacon1
ID Beacon2	Like Beacon2, Zutritt Beacon2
⋮	⋮

Abbildung 2.3: EEPROM Adressierungsmuster

des generierten Audiopakets haben als namen nur noch eine Zahl, die bestimmt an welcher Stelle sie auf der SD-Karte abgespeichert wird.
Der CLI-Befehl `loadsd` nimmt das vorher generierte Audiopaket und schreibt es auf die SD-Karte des Audioguides. Dazu schickt es dem Mikrocontroller den Befehl, dass er den Multiplexer umschaltet. Das Betriebsystem des PC sollte dann die SD-Karte mounten und dem Python Skript verfügbar machen, worauf dieses die Daten kopiert. Es wird dabei angenommen, dass nur ein Audioguide am PC angeschlossen ist und dass alle SD-Karten "dojo-sd"heissen.
Der CLI-Befehl `maketicket` nimmt die Inventardatei und die Presetdatei als Argumente und fragt den Nutzer, welche Ausstellungen und Sprache dem Besucher zur Verfügung stehen sollen. Mit diesen Inputs wird ein Ticket generiert, das über die Serielle Schnittstelle auf das EEPROM des Mikrocontrollers geladen wird.
Auf dem EEPROM hat jeder Beacon zwei Bytes, die die Beacon-ID, Like und Zutritt beinhalten. Die Likes sind zu Beginn des Besuchs alle auf Null und können vom Besucher gesetzt werden. Das Zutrittsbits der bezahlten Ausstellungen werden zu Beginn des Besuchs gesetzt und können vom Besucher nicht verändert werden.

2.3 Auslesen, Schreiben

Das EEPROM des Mikrocontrollers wird über eine serielle Schnittstelle beschrieben, wobei alle Befehle das gleiche Muster haben:

```
[1 char] op-code | [3 char] arg1 | [3 char] arg2 ;
```

Um ein Byte mit Wert 42 in die Adresse 2 zu schreiben, muss der String `p002042` auf die serielle Schnittstelle geschrieben werden. Um das Byte an der Adresse 2 auszulesen, muss der String `r002 ;` auf die serielle Schnittstelle geschrieben werden. Wichtig ist, dass alle Befehle mit einem Semikolon terminiert werden.
Das Python Modul beinhaltet neben Schreib- und Lesefunktionen auch noch Funktionen, die ein ganzes Ticket auf das EEPROM schreiben können, bzw. wieder einlesen können. Hier ist es wichtig, dass das vordefinierte Muster des EEPROMs in Abbildung 2.3 eingehalten wird.

2.4 Validierung

Für die Validierung der Datenverwaltung wurden Python Unittests eingesetzt, die mit einer Testdatenbank alle Funktionen testen. Neben den einzelnen Funktionen wurde auch ein kompletter

Durchlauf mit der Testdatenbank gemacht.
Die EEPROM-Operationen werden mit einem Arduino Uno-Board getestet, wobei das EEPROM vom PC aus beschrieben und wieder eingelesen wird. Die Unittests stellen sicher, dass das EEPROM die gewünschten Daten beinhaltet und dass diese mit der spezifizierten Geschwindigkeit übertragen werden.
Es werden zudem noch Benchmarks durchgeführt, welche sicherstellen, dass nicht unnötig gewartet wird. Ein Test, der misst, wie lange das beschreiben von verschieden lange Tickets dauert zeigt, dass das beschreiben eines vollen Tickets mit 250 Beacons 6.17 dauert.

3 USB

Im folgenden Kapitel wird erklärt, wie über die USB-Schnittstelle Daten auf den Dojo geladen werden und ausgelesen werden. In einem Unterkapitel wird sogleich die Validierung beschrieben.

3.1 Technische Grundlagen

Nachfolgend werden einige Begriffe zur Datenübertragung kurz erklärt, welche im Kapitel von Bedeutung sind.

UART Universal Asynchronous Receiver Transmitter (UART) realisiert eine digitale serielle Schnittstelle. Über die UART-Schnittstelle können Daten über einen Datenstrom gesendet und empfangen werden. Die Daten sind in einem fixen Rahmen, bestehend aus Start-Bit, fünf bis neun Datenbits, optionalem Parity-Bit zur Fehlererkennung und einem Stopp-Bit.

SDIO Secure Digital Input Output (SDIO) bezeichnet ein Interface für die Datenübertragung zwischen SD-Karten. Die Daten werden wahlweise im SPI, 1-Bit oder im 4-Bit Modus übertragen.

3.2 Konzept

Zur Kommunikation mit dem Mikrocontroller im Gerät besteht eine USB-UART-Schnittstelle. Um neue Audio-Dateien auf die SD-Karte zu schreiben, wurde ausserdem eine USB-SDIO-Schnittstelle realisiert. Folgende Abbildung 3.1 zeigt das Konzept mit den beiden Schnittstellen.

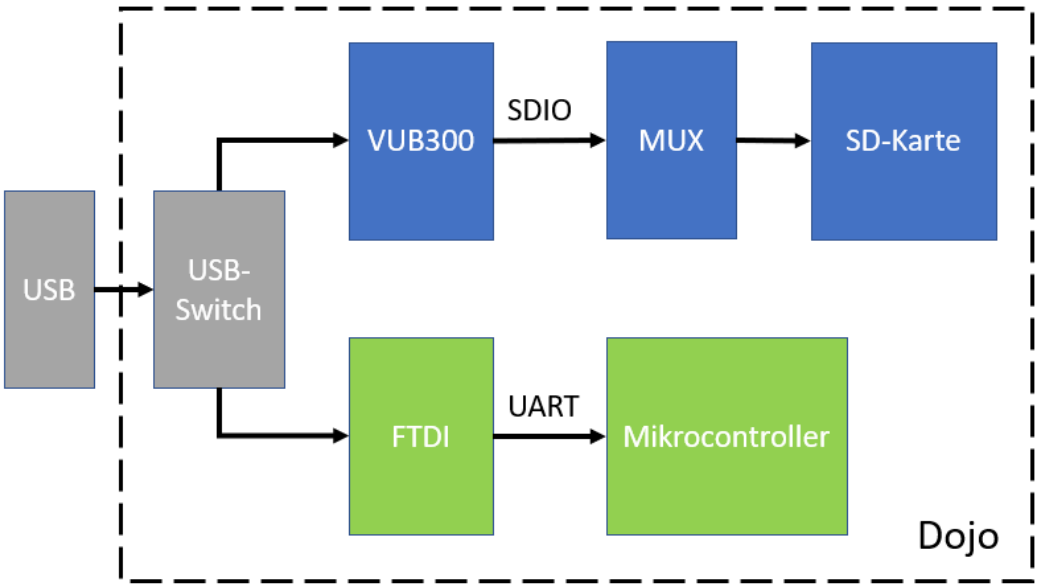

Abbildung 3.1: Konzept USB

Eine micro USB-Typ B Schnittstelle ist die gemeinsame Schnittstelle nach aussen. Dahinter schaltet ein USB-Switch, gesteuert durch den Mikrocontroller, einen der beiden Pfade durch. Es kann jeweils entweder der SDIO-Pfad oder UART-Pfad durchgeschaltet werden, nie beide Pfade gleichzeitig.
Als USB-Switch wird der Baustein TS3USB30E von Texas Instruments verwendet. Dieser Chip wurde dafür entwickelt, um high-speed USB 2.0 Signale in portablen Geräten und Konsumelektronik zu schalten. Gesteuert wird der USB-Switch über zwei Pins. Über den Enable-Pin kann der Baustein in einen hochohmigen Zustand gebracht werden, um den Bus nicht zu belasten und den Stromverbrauch zu senken. Mit dem Select-Pin wird der durchzuschaltende Pfad ausgewählt. Folgende Tabelle 3.1 zeigt die Wahrheitstabelle für die entsprechenden Pfade:

Select (Pin24)	$\overline{Enable}$ (Pin23)	Funktion
X	HIGH	hochohmig
LOW	LOW	Pfad = UART
HIGH	LOW	Pfad = SDIO

Tabelle 3.1: Wahrheitstabelle USB-Pfade

3.3 USB zu UART

Der Baustein FT231X von FTDI stellt das Interface von USB zu UART bereit. Das ganze USB-Protokoll wird dabei auf dem Chip abgehandelt und es sind kaum zusätzliche Bauteile nötig (siehe Schema). Über zwei Pins (Rx und Tx) werden die Daten seriell übertragen.
Bei aktivem UART-Pfad wird der Dojo am Computer als serielle Schnittstelle erkannt. Über diese Schnittstelle kommuniziert nun die Software auf dem Computer mit dem Mikrocontroller im Gerät, um beispielsweise die ‹Likes›auszulesen.

3.4 USB zu SDIO

Der VUB300 Chip vom Hersteller elan stellt das USB zu SDIO Interface bereit. Damit kann die SD-Karte im Dojo über den USB-Anschluss beschrieben und ausgelesen werden. Wird mit dem USB-Switch der SDIO-Pfad durchgeschaltet, erscheint die SD-Karte am Computer als Datenträger und kann gelesen sowie beschrieben werden. Zu beachten ist, dass zwischen dem VUB300 Chip und der SD-Karte noch ein weiterer Multiplexer (MUX) eingebaut ist. Mit diesem wird die SD-Karte zwischen dem VUB300 und dem Audio-Chip umgeschaltet. Genaueres dazu findet man in Abschnitt **4 Audioausgabe**. Damit die SD-Karte mit dem SDIO-Pfad verbunden ist, muss der MUX TS3A27518 (siehe Schema im Anhang) folgendermassen angesteuert werden:

$\overline{Enable}$ (Pin25)	IN1 (Pin26)	IN2 (Pin27)	Funktion
HIGH	X	X	hochohmig
LOW	LOW	LOW	Pfad = SDIO(VUB300)
LOW	HIGH	HIGH	Pfad = Sound-Modul

Tabelle 3.2: Wahrheitstabelle SD-Pfade

3.5 Validierung UART-Pfad

Um die korrekte Funktion der seriellen Schnittstelle zwischen dem USB-Anschluss und dem Mikrocontroller zu überprüfen, wurde eine kleine Testsoftware auf den Mikrocontroller geladen. Sobald der Mikrocontroller am RX-Pin Zeichen empfängt, sendet er diese über den TX-Pin wieder zurück. Mit dem Serial Monitor der Arduino IDE wurden die Zeichenketten erfolgreich gesendet und empfangen.

3.6 Validierung SDIO-Pfad

Mit einem weiteren Testprogramm wurde die Funktionsfähigkeit des SDIO-Pfades überprüft. Dafür wurde der USB-Switch und der MUX mit dem Mikrocontroller entsprechend angesteuert. Die SD-Karte wurde daraufhin von einem Computer mit Linux Betriebssystem als Datenträger erkannt und konnte erfolgreich ausgelesen und beschrieben werden. Leider stellte sich nach weiteren Tests heraus, dass auf einem Windows-Rechner keine SD-Karte erkannt wird. Die Gründe dafür sind nicht bekannt und das Problem konnte auch nicht behoben werden. Somit muss die Computer-Software auf einem Linux Betriebssystem gestartet werden.

Anmerkung
Durch die begrenzte Verfügbarkeit des VUB300 Chips, sowie der geringen Informationsdichte über den Baustein und dem Kompatibilitätsproblem mit Windows, ist dieser nicht für weitere Projekte oder Serienproduktionen zu empfehlen.

4 Audioausgabe

In diesem Kapitel werden die technischen Grundlagen welche sich auf das verwendete Audio-Modul beziehen, das Konzept sowie die Funktion der Audioausgabe beschrieben. Zudem wird die Validierung erläutert.

4.1 Technische Grundlagen

Für den Prototyp des Dojo's wurde unter anderem ein WTV020 Chip verwendet. Das WTV020 Modul wird in diesem Kapitel beschrieben, da diese Informationen für die nachfolgenden Kapitel benötigt werden.
Das WTV020 Modul ist ein Soundmodul welches es ermöglicht Audiodateien auf einem Aktor abzuspielen. Auf einer maximal 1GB grossen μSD-Karte können bis zu 512 Audiodateien abgespeichert werden. Die Audiodateien auf der μSD-Karte müssen jedoch dem .wav oder.ad4 Format entsprechen. Die Dateien müssen gemäss Vorgabe: 0000; 0001; 0002; ... nummeriert werden.
Der WTV020 Chip kann in zwei verschiedenen Modes betrieben werden, dem MP3 Mode und dem Two Line Serial Mode. Im MP3 Mode können direkt 6 Pins angesteuert werden. Durch die sechs Pins können folgende Funktionen umgesetzt werden: Reset, ±Volume , next, previous und play, pause. Für das Dojo wird jedoch der two line serial mode genützt. Dieser kann das Modul mit nur 3 Pins betreiben. Der Mikrocontroller muss an den Clock-, den Data- und den Busy-Pin angeschlossen werden. Im two line serial mode können die Audiodateien welche sich auf der μSD-Karte befinden abgespielt werden. Er ermöglicht zudem, ähnlich wie im MP3 Mode ein Lied zu pausieren und neu zu starten, sowie eine Lautstärkenregulation.

4.2 Konzept

Das Konzept der Audioausgabe ist wie folgt aufgebaut: Die auf einer μSD-Karte abgespeicherten

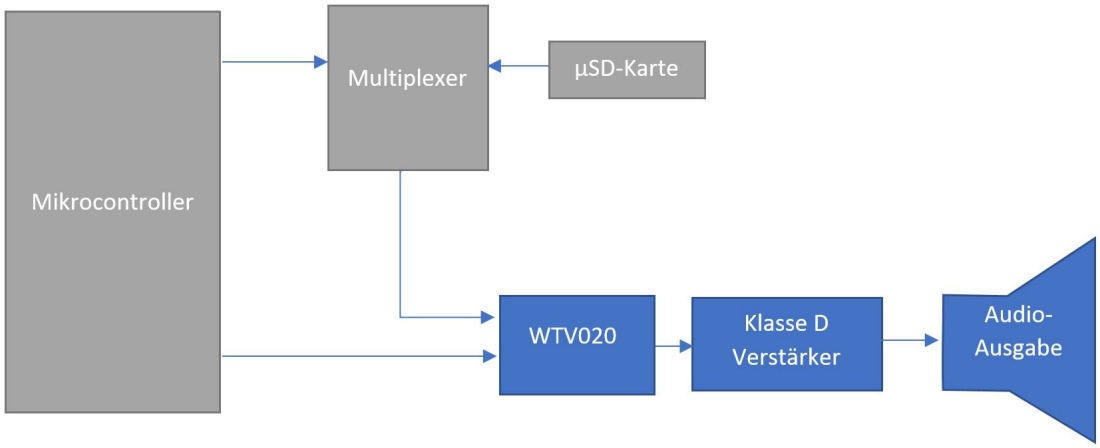

Abbildung 4.1: Audio Konzept

Audiodateien werden über einen Multiplexer, welcher vom Mikrocontroller gesteuert wird, an

einen WTV020 Chip übertragen. Dieser wird in einem Serial Mode [Kapitel WTv020] betrieben und kann ebenfalls vom Mikrocontroller angesteuert werden. Der Audiochip entschlüsselt die Daten und gibt diese an einen Klasse D Verstärker weiter. Dieser wird benötigt um eine gut hörbare Lautstärke zu erreichen. Das Audiosignal wird schliesslich an einem Bone Conductor ausgegeben. Nachfolgend wird die Anordnung auf dem Print der einzelnen Komponenten dargelegt.

4.3 Hardware

Wie im Konzept beschrieben, muss man, um Daten von der μSD-Karte auszulesen zuerst den Multiplexer ansteuern. Dies geschieht über den Mikrocontroller. Im ungesteuerten Zustand ist der MUX auf den VUB300 geschaltet. Sobald der MUX entsprechend angesteuert wird schalten die benötigten Kontakte. Der WTV020SD-20S Chip kann nun auf die μSD-Karte zugreifen. Über vier Pins werden die Audiofiles ausgelesen und weitergegeben. Das Audiosignal wird auf den Klasse-D Verstärker gegeben. Dieser ist wie folgt aufgebaut. Um ein Rauschen an der Spei-

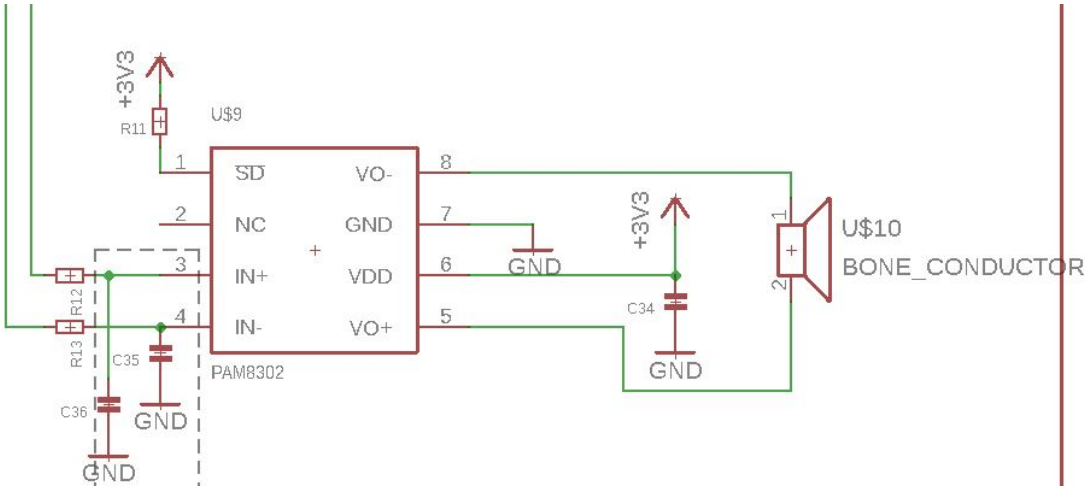

Abbildung 4.2: Klasse D Verstärker

sung zu vermeiden wird ein einμ Farad grosser Kondensator benötigt. In diesem Projekt wurde eine konstante Verstärkung mit dem Faktor ca.15 verwendet, da die Lautstärken Regelung über den WTV020-Chip geregelt werden soll. Dieser Faktor entsteht aus dem Verhältnis der Widerstände. Um das Rauschen der Eingänge sowie zu hohe Frequenzen zu filtern wurden zuerst wie die Abbildung 4.2 zeigt, Kapazitäten eingeplant. Auf dem Prototyp wurden aufgrund von Tests welche in der Validierung aufgeführt werden, die Kapazitäten weggelassen.
Wie erwähnt wird die Lautstärkenregelung über den WTV020-Chip geregelt. Dies ist jedoch im serial mode fehleranfällig. Aufgrund dessen wurde für die Tests ein 50 Ohm grosser Widerstand vor den Bone Conductor geschaltet da bei grossen Strömen dieser eine zu kleine Innenimpedanz aufweist.
Wie erwähnt wird die Lautstärkenregelung über den WTV020-Chip geregelt. Dies ist jedoch im serial mode fehleranfällig. Aufgrund dessen wurde für die Tests ein 50 Ohm grosser Widerstand vor den Bone Conductor geschaltet da bei grossen Strömen dieser eine zu kleine Innenimpedanz aufweist. Die Audioausgabe erfolgt wie beschrieben über einen Bone Conductor. Der Bone Conductor besteht aus einem kleinen Metallstab welcher mit einer Kupfer Spule umwickelt ist. Sobald ein pulsförmiger Strom durch die Spule fliest, dehnt sich ein Magnetfeld aus welches ein

zusammenziehen auslöst. Der Bone Conductor ermöglicht es durch die Vibrationen das eine Audio Datei über den Schädelknochen abgespielt wird und so nur für eine Person hörbar ist, diese jedoch immer noch die Umgebungsgeräusche war nimmt.

4.4 Firmware

Wie im Kapitel Hardware beschrieben muss der MUX angesteuert werden um dem WTV-Chip den Zugriff auf die μSD-Karte zu gewähren. Die folgende Tabelle zeigt wie der MUX angesteuert werden muss, damit die benötigten Pins durgeschalten sind.

$\overline{EN}$	IN1	IN2	NC1/2/3 TO COM1/2/3, COM1/2/3 TO NC1/2/3	NC4/5/6 TO COM4/5/6, COM4/5/6 TO NC4/5/6	NO1/2/3 TO COM1/2/3, COM1/2/3 TO NO1/2/3	NO4/5/6 TO COM4/5/6, COM4/5/6 TO NO4/5/6
H	X	X	OFF	OFF	OFF	OFF
L	L	L	ON	ON	OFF	OFF
L	H	L	OFF	ON	ON	OFF
L	L	H	ON	OFF	OFF	ON
L	H	H	OFF	OFF	ON	ON

Abbildung 4.3: MUX, TS3A27518 Funktionstabelle

Für die Audioausgabe müssen alle Öffner geschlossen werden. Dies wird über die Pins 25 – 27 (Port C) realisiert. Sobald Musik abgespielt werden muss, werden die benötigten Ausgänge gesetzt. Sobald am Pin 9 über einen Taster Play ein low Signal erkannt wird, wird folgende Funktion aufgerufen:

```
void sendWTVcommand(unsigned int command){
digitalWrite(WTV_CLK, LOW);
_delay_us(1900);
for (byte i = 0; i < 16; i++)
{
_delay_us(100);
digitalWrite(WTV_CLK, LOW);
digitalWrite(WTV_DOUT, LOW);
if ((command & 0x8000) != 0)
{
digitalWrite(WTV_DOUT, HIGH);
}
_delay_us(100);
digitalWrite(WTV_CLK, HIGH);
command = command<<1;
}
}
```

Abbildung 4.4: Audio Datei abspielen

Die benötigten Befehle welche sendWTVcommand mitgegeben werden müssen sind aufgrund des WTV020-Chips vorgefertigt.

Nachfolgend werden die verwendeten Befehle aufgeführt.
OPCODEPLAYPAUSE: OPCODEPLAYPAUSE muss default mässig auf 0xFFE gesetzt werden. Dies ermöglicht es eine Datei abzuspielen und zu pausieren. Um die Ausgabe zu starten muss immer am Anfang die gewünschte File Nummer mitgegeben werden (0-512). Ohne File Nummer wird die Audioausgabe pausiert und kann wieder gestartet werden.
OPCODESTOP: Dies stoppt die laufende Datei und setzt diese zurück.
OPCODEVOL: OPCODEVOL muss am Anfang auf 0xFFF0 gesetzt werden. Dies ermöglicht die Lautstärken Regulierung wobei 0xFFF0 den Aktor auf stumm schaltet und 0xFFF7 die maximale Lautstärke ausgibt.
Für die Audioausgabe wurden PLAYPAUSE und OPCODEVOL verwendet. Durch diese ist es möglich die gewünschten Funktionen umzusetzen.

4.5 Validierung

Um die Audioausgabe zu testen, wurden verschiedene Versuche durchgeführt. Zuerst wurde der Bone Conductor über eine kleine Verstärkerschaltung direkt an einem Laptop angeschlossen, um die benötigte Leistung und die Lautstärke abschätzen zu können. Bei diesen Messungen ergab sich, bei einer sehr gut hörbaren Lautstärke, eine maximale Scheinleistung von 0.28 VA.
Da das WTV020 kleine Eigenheiten besitzt musste das Versuchsboard in dem im Kapitel (WTV tech Grundlagen) beschriebenen MP3 Mode betrieben werden. So konnten nicht kompatible μSD-Karten aussortiert werden.
Alle Funktionen des WTV020-Chips wurden separat überprüft. Alle verlangten Funktionen des Chips laufen wunschgemäss. Zudem wurde ein Versuchsaufbau mit dem D-Klasse Verstärker durchgeführt, mit welchem die aktuell maximal benötigte Leistung von XXXW gemessen wurde. Trotz der funktionierenden Tests ist aktuell am Prototyp die Lautstärkenregelung nicht möglich. Das Audiomodul übernimmt die eingestellte Lautstärke, setzt diese jedoch meist wieder auf den vorherigen Wert. Zudem ist zu erwähnen, dass für eine Massenproduktion ein WTV020 Modul nicht geeignet wäre, aufgrund der erwähnten Eigenheiten. Das Modul nimmt trotz gleicher Formatierung und gleichem Hersteller nicht jede μSD-Karten an. Zudem treten beim serial mode immer wieder kleiner Ungereimtheiten auf, wie z.B. die Lautstärkenregelung.

5 Bluetooth

5.1 Technische Grundlagen

«««< Updated upstream

5.2 IBeacons auslesen und identifizieren

Mittels der Funktion scan() im State SCAN wird mit dem hm-11 nach den verfügbaren IBeacons in der Umgebung gesucht. Über eine serielle Schnittstelle (UART) wird vom MCU der Befehl AT+DISI? dem hm-11 geschickt, wobei als Antwort dann alle umliegenden IBeacons in einem String zurückkommen (siehe Abbildung 5.1). Dafür wird vom MCU jeder char, rsp. jedes

Send: AT+DISI?

Recv: OK+DISCS (Scan start)

Recv: OK+DIS[P0:P1:P2:P3:P4] (if have one device)

Recv: OK+DIS[P0:P1:P2:P3] (if have two devices)

.....

Recv: OK+DISCE (Scan end)

Abbildung 5.1: P0: Factory ID (8 Byte); P1: IBeacon UUID (32 Byte); P2: Major-, Minorvalue, Measured Power (10 Byte); P3: Media-Access-Control-Adresse MAC (12 Byte); P4: RSSI (4 Byte) [?]

Byte vom Datenbuffer ausgelesen und verwertet. Es wird zuerst nach einer UUID eines IBeacons gefiltert und dann die ersten drei Zahlen in einen Integer gecastet. Anschließend soll das Majorvalue[1], welches einen bestimmten, selbst wählbaren Wert hat, abgeglichen werden, damit nur die IBeacons berücksichtig werden, die relevant sind. Ist der empfangene RSSI-Wert grösser als -90dbm, wird der IBeacon eingespeichert. Dies wird mit jedem empfangenen IBeacon wiederholt und immer die RSSI-Werte miteinander verglichen, wobei dann der IBeacon mit dem grösseren RSSI-Wert ins System gespeichert und als Returnvalue von der scan() Funktion zurückgegeben wird. ======= Da eine drahtlose Verbindung mit Beacons hergestellt werden muss, wird dazu ein Bluetooth-Modul, das HM-11, verwendet. Es ermöglicht das Übertragen der Identifikationsnummer vom Beacon auf das Dojo. Der Vorteil eines vorgefertigten Moduls liegt in der einfachen Ansteuerung Hardware-, sowie Firmware-mässig.

5.2.1 Technische Daten

Das HM-11 Bluetooth-Modul benötigt eine Versorgungsspannung von 3.3 V DC und verfügt über das Kommunikationsprotokoll UART. Die Standard Baudrate beträgt 9600bps und nimmt beim Senden einen Strom von 15 mA auf. In offener Umgebung beträgt die Reichweite bis zu 30 m und unterstützt AT Kommandos, um das Bluetooth-Modul zu konfigurieren.

[1] Major- und Minorvalues dienen hauptsächlich zur zusätzlichen Identifikation eines IBeacons

5.2.2 Bluetooth-Konfiguration

Damit der Prototyp die Beacons erkennt, wird ein Bluetooth-Modul, das HM-11, verwendet. Es ermöglicht das Übertragen der Identifikationsnummer vom Beacon auf die Printplatte. Um eine reibungslose Kommunikation zu gewährleisten, müssen einige Konfigurationen beim Bluetooth-Modul vorgenommen werden. Das Bluetooth-Modul wird mit einem FTDI verkabelt und am PC angeschlossen. Mit den AT-Kommandos wie AT+Name, AT+ROLE1, AT+IMME1 und AT+DISI? kann das Bluetooth-Modul nun eingestellt werden. Falls das Bluetooth-Modul bereits auf dem Print bestückt ist, kann das Modul über die UART-Schnittstelle konfiguriert werden.

5.3 Beacons im Museum

Beacons sind kleine und kompakte Bluetooth-Sender die auf der Low Energy Spezifikation basieren. Sie können entweder batteriebetrieben oder mit permanentem Stromanschluss an Räumen oder Objekten installiert werden. Bewegt sich ein Museumsbesucher in die Nähe eines Beacons mit diesem Dojo, so kann er Audio- und Video-Beiträge empfangen und mit dem Dojo abspielen. Zudem kann an der Kasse des Museums entschieden werden, welche Zonen aktiviert werden sollen. Das Dojo beinhaltet somit auch die Zutrittsberechtigungen. Der Vorteil dieses Systems ist der Preis. Es wird nur so viel bezahlt, was den Kunden interessiert.

5.3.1 Minew E7

Mit dem Beacon Minew E7 konnte ein Museum simuliert werden, um den Prototypen auf die Funktionalität zu prüfen. Dieser Beacon lässt sich über die gratis BeaconSET+ App konfigurieren. Dieser Beacon bringt einige Vorteile mit sich. Zum einen kriegt er eine maximale Reichweite von 100 m hin, doch noch viel wichtiger ist das Protokoll. Der Minew E7 unterstützt iBeacon und Eddystone Protokolle die entweder separat oder gleichzeitig benutzt werden können.

5.3.2 Protokollbeschreibung

Das iBeacon-Protokoll wurde von Apple entwickelt und baut auf der Bluetooth-Low-Energy-Spezifikation auf. Das Paket das ausgestrahlt wird hat eine Länge von 30 Byte. In diesem Paket stecken die ganzen Informationen wie Geräte-ID, Beacon-Typ, Referenzempfangsleistung und noch viele weitere Daten drin. Um die Beacons zu scannen, wurde die Software so geschrieben, dass es nur die UUID, den Majorvalue und den RSSI-Wert benötogt. Nachfolgend wird der Aufbau des iBeacons-Paket graphisch dargestellt.

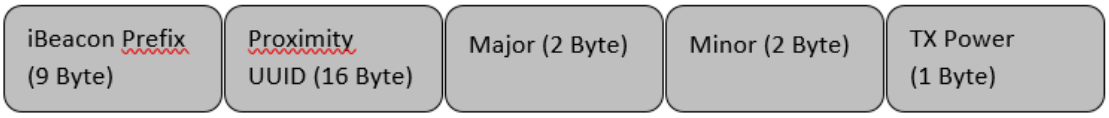

Abbildung 5.2: Aufbau eines iBeacon-Pakets

5.3.3 Receiver Signal Strength Indicator (RSSI)

Der RSSI ist ein dimensionsloser Wert. Er beschreibt die empfangene Leistung bei kabelloser Kommunikation. Angegeben wird der Wert in dBm, was das Verhältnis in Dezibel zwischen der

Leistung mW und der Referenzleistung 1 mW darstellt.

5.4 Konzept

5.5 Hardware

5.6 Firmware

»»»> Stashed changes

5.7 Validierung

Für die Validierung wurden zwei IBeacons mit Androidhandys simuliert (Beacon Simulator App). Dafür wurden diese mit einer Distanz von ungefähr $6m$ zueinander auf einer Höhe von ca. $2m$ platziert. Um die detektierten IBeacons direkt auszuwerten, wurde die UUID auf einen Emulator (Putty) über eine serielle Schnittstelle (USB Typ micro B) herausgeschrieben. Somit konnte verifiziert werden, dass der vom Dojo detektierte IBeacon auch der sich am Dojo nächsten befindliche IBeacon war.

Eine definitive Distanzbestimmung zwischen dem IBeacon und dem Dojo ist schwierig, da die Sendeleistung der simulierten IBeacons etwas schwanken. Auch die Auslegung des Raumes, sowie die Einrichtung kann das Signal abschwächen, was direkten Einfluss auf die Erkennungsdistanz hat. Es kann aber festgehalten werden, dass ein IBeacon in einem Umkreis von $3m$ garantiert erkennt wird, solange sich keine Gegenstände unmittelbar zwischen den beiden Objekten befinden.

6 Firmware

6.1 Konzept

6.2 Statemachine

Auf dem Mikrocontroller läuft eine Mealy-Statemachine mit fünf Zuständen:

[leftmargin=3.2cm]Es wird vom Dojo nach IBeacons in naher Umgebung gesucht. Falls einer gefunden wurde, dann wird direkt das dazugehörige Audiofile bereitgestellt. Wird der Playbutton gedrückt, ändert sich der State zu PLAY. Ansonsten wird weiter gescanned. Hier werden die bereitgestellten Audiofiles abgespielt. Wird der Playbutton nochmals gedrückt, wird das Abspielen abgebrochen und der State wechselt wieder zu SCAN.

Wird das Dojo an den Computer angeschlossen, ist der Zustand des Dojos abhängig von der gewünschten Tätigkeit. Dafür wird vom Computer aus ein Befehl über das USB-Kabel gesendet, woraufhin sich der State ändert:

[leftmargin=3.2cm]Die vom Besucher getätigten Likes werden vom EEPROM auf den Computer transferiert. Die microSD-Karte wird mit den gewünschten Audiofiles beschrieben. Ein Ticket wird auf das interne EEPROM geladen.

6.3 Datenverwaltung

In diesem Kapitel wird die Datenverwaltung auf dem internen EEPROM erklärt.

6.4 Validierung

Hier wird die Validierung der kompletten Firmware sowie dessen Ergebnisse dargelegt.

7 Verzeichnisse

Abbildungsverzeichnis

1.1 Blockschaltbild Gesamtsystem 3

2.1 Inventardatei für eine Testausstellung 4
2.2 Presetdatei für eine Testausstellung 5
2.3 EEPROM Adressierungsmuster 5

3.1 Konzept USB 7

4.1 Audio Konzept 10
4.2 Klasse D Verstärker 11
4.3 MUX, TS3A27518 Funktionstabelle 12
4.4 Audio Datei abspielen 12

5.1 Rückgabe des AR+DISI? Befehls 14
5.2 Aufbau eines iBeacon-Pakets 15

8.1 Abmessungen WTV020 19

8 Anhang

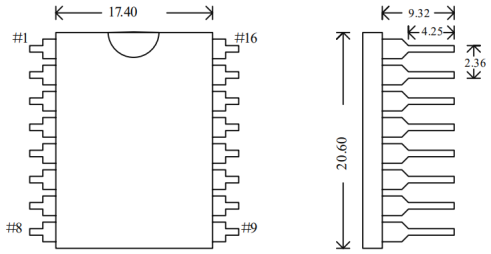

Abbildung 8.1: Abmessungen WTV020