Fachhochschule
Nordwestschweiz

Fachbericht

Team 1

Projekt 4

Fachhochschule Nordwestschweiz FHNW

5. 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

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 **?? ??** behandelt die Verarbeitung und Ausgabe der Audiofiles. Das nächste Kapitel **?? ??** 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 **?? ??** detailliert. Die Validierung der einzelnen Blöcke wird im jeweiligen Kapitel abgehandelt.

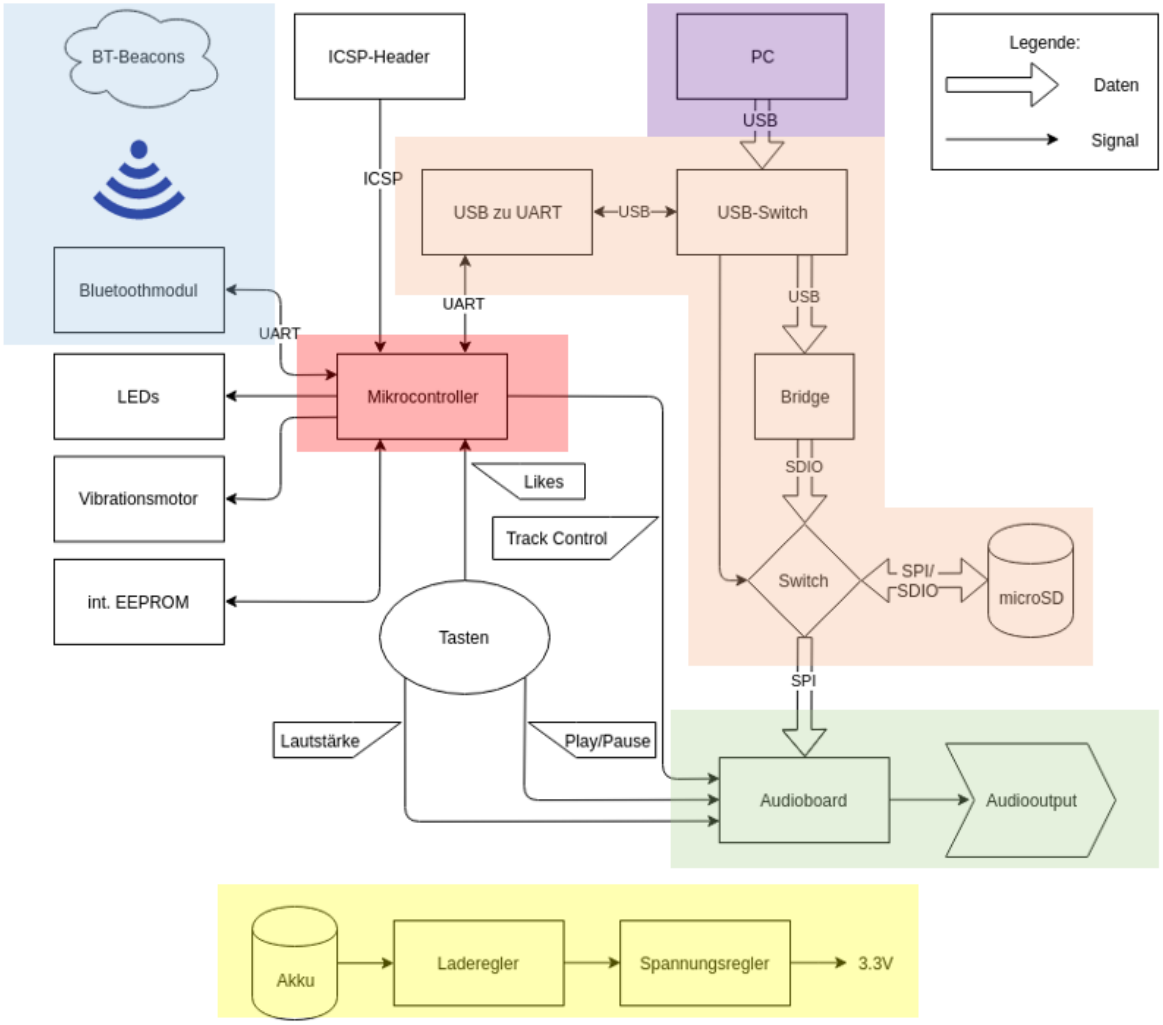

Abbildung 1.1: Blockschaltbild Gesamtsystem

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

```
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

```
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

`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 eingehalten wird 2.3.

2.4 Validierung

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

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.

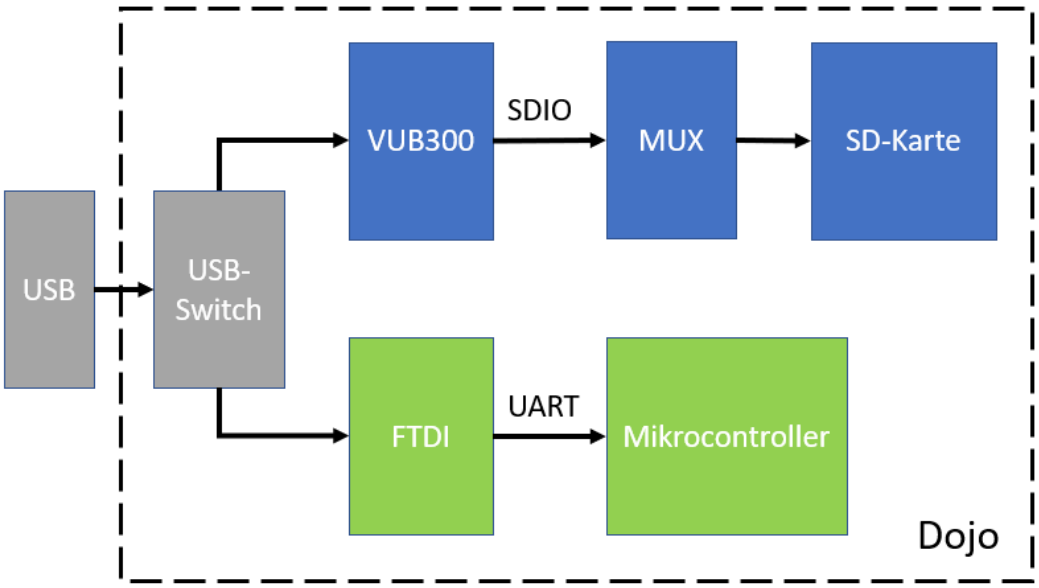

Abbildung 3.1: Konzept USB

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 **?? ??**. 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 Verzeichnisse

Abbildungsverzeichnis