Multi-Room Sound Adapter

Nico Lang, Philipp Immler

Februar 2025

1 Projekt

1.1 Projektteam

Nico Lang
Wirtschaftsingenieure/Betriebsinformatik
Grießau
6651 Häselgehr AT
Nico.Lang@hak-reutte.ac.at

Philipp Immler
Wirtschaftsingenieure/Betriebsinformatik
Hoheneggweg 21a
6682 Vils AT
Philipp.Immler@hak-reutte.ac.at

1.2 Eidesstattliche Erklärung

Hiermit versichere ich, dass ich die vorliegende Arbeit selbstständig verfasst und keine anderen Hilfsmittel als die angegebenen benützt habe. Die Stellen, die anderen Werken (gilt ebenso für Werke aus elektronischen Datenbanken oder aus dem Internet) wörtlich oder sinngemäß entnommen sind, habe ich unter Angabe der Quelle und Einhaltung der Regeln wissenschaftlichen Zitierens kenntlich gemacht. Diese Versicherung umfasst auch in der Arbeit verwendete bildliche Darstellungen, Tabellen, Skizzen und Zeichnungen. Für die Erstellung der Arbeit habe ich auch folgende Hilfsmittel generativer KI-Tools (ChatGPT 3.5) zu folgendem Zweck verwendet: Inspiration und allgemeine Information. Auch Übersetzer (DeepL) wurden zur Hilfe genommen. Die verwendeten Hilfsmittel wurden vollständig und wahrheitsgetreu inkl. Produktversion und Prompt ausgewiesen.

.....	.....
Ort, Datum	Unterschrift Schüler/in
	.....
	Unterschrift Schüler/in

1.3 Abstract Deutsch

1.4 Abstract English

1.5 Danksagung

Inhaltsverzeichnis

2 Einleitung

Hier befindet sich die allgemeine Einleitung der Diplomarbeit.

2.1 Einleitung Hardware

Der Teil der Hardware für folgende Diplomarbeit beschäftigt sich damit, einen sinnvollen internen Aufbau des Geräts zu erzielen, die am besten geeigneten Hardware-Komponenten zu finden, das System bzw. die einzelnen Komponenten zusammenzubauen (Platine) und zu testen. Dieser Teil der Diplomarbeit wird von Nico Lang übernommen. Zudem beschäftigt sich dieser Teil mit dem optischen Design des Geräts (Gehäuse) und bestimmt die technischen Anforderungen (Schnittstellen), die der Adapter letztendlich haben soll. Bei der Planung soll darauf geachtet werden, möglichst viele Kosten einzusparen, ohne dabei die Faktoren der Sicherheit und Qualität zu vergessen.

2.2 Einleitung Software

Der Teil der Software für folgende Diplomarbeit beschäftigt sich damit, einerseits die Software des Adapters, andererseits die Software der Smartphoneapp zu entwickeln. Dieser Teil der Diplomarbeit wird von Philipp Immler übernommen. Die Software des Adapters wird mit der Programmiersprache C++ codiert. Die Software der Smartphoneapp wird mit JavaScript codiert. Um eine bestmögliche Leistung und Effizienz zu garantieren, werden bei der Programmierung zahlreiche Bibliotheken und Frameworks verwendet. Bei der Entwicklung der Software wird ein großes Augenmerk auf Sicherheit und Effizienz gelegt.

3 Planung

3.1 Festlegung Funktionsweise

GEMEINSAM

3.2 Auswahl Hardwarekomponenten

Zur Auswahl der Hardwarekomponenten des Adapters wird zu aller erst die externe Ausstattung des Adapters überlegt. Das bedeutet praktisch alles, mit dem ein Endverbraucher letztendlich zu tun hat. Dann kann der interne Teil, also die Technik dahinter, individuell auf die Anforderungen des externen Teils designet und entwickelt werden.

3.2.1 Auswahl externe Hardware

Der Adapter soll ein möglichst kompakt konstruiertes und stabiles Gehäuse bekommen. An diesem wird ein einfacher Taster zur Interaktion angebracht. Mit dem Taster sollen einige Funktionen des Adapters ermöglicht werden. Beispielsweise per Klick, Doppelklick oder kurzem Halten. Da sich die Aufgaben des Tasters selbst gering halten werden (Verbindungsvorgang, Ein- und Ausschalten, ...) wird nur ein einziger Taster verwendet, um die Komplexität des Gesamtsystems zu senken. Die weitaus komplizierteren Funktionen sollen alle samt in der Smartphone-Applikation ermöglicht werden. Zusätzlich werden eine oder mehrere Leuchtdioden zur Statusanzeige verbaut, um beispielsweise den aktuellen Verbindungsstatus zum Mobilgerät und zum Internet oder den aktuellen Akkustand anzuzeigen.

Gehäuse
Das Gehäuse soll alle Komponenten auf möglichst kleinem Raum zusammenhalten. Da sich Komponenten und möglicherweise auch das Design selbst laufend ändern, wird dieses deshalb erst gegen Ende des Projekts finalisiert werden können.

Taster
Für den Taster wird ein herkömmlicher Knopf in das Gehäuse eingelassen.

LED (Light-Emitting Diode)
Es ist durchaus bekannt, dass Leuchtdioden im Vergleich zu anderer Lichttechnik effizienter sind.
Laut
`https://ledtipps.net/wirkungsgrad/#wirkungsgrad-und-lichtausbeute-im-vergleich`
"Grundsätzlich fällt der Wirkungsgrad bei höheren Leistungen immer etwas besser aus."
In der folgenden Tabelle aus
`https://ledtipps.net/wirkungsgrad/#wirkungsgrad-und-lichtausbeute-im-vergleich`
wird der Wirkungsgrad sowie die Lichtausbeute unter herkömmlicher Lichttechnik verglichen.

Leuchtmittel	**Wirkungsgrad**	**Lichtausbeute**
LED	25-40%	80 – 150 lm/W
Energiesparlampe	15-25%	40 – 60 lm/W
Halogenlampe	8-12%	15 – 20 lm/W
Glühlampe	3-5%	10 – 15 lm/W

Für dieses Projekt bedeutet das, dass die Leuchtdiode wohl die effizienteste Methode für eine Statusanzeige auf dem Adapter ist. Dies trägt unter anderem zu einer längeren Akkulaufzeit bei.

3.2.2 Auswahl interne Hardware

3.3 Auswahl Technologien

3.3.1 Protokolle

In diesem Kapitel geht es um die Recherche und Auswahl von Protokollen, die für den Austausch von Daten verwendet werden.

HTTP
Das Hyper Text Transfer Protocol ist ein weitverbreitetes Protokoll im Web und wird größtenteils für die Kommunikation zwischen Browsern und Webservern eingesetzt. Dabei basiert das Protokoll auf sogenannten "Requests"(auf Deutsch: Anfragen). Es gibt zahlreiche Anwendungen für HTTP. Wir nutzen es einerseits als REST-API und andererseits für das Audio-Streaming. (vgl. URLPI02)

Anwendung als REST-API
Der Begriff "REST-APIßetzt sich aus den Abkürzungen "RESTünd ÄPIßusammen. Dabei steht "REST"für "Representational State Transfer"(auf Deutsch: "gegenständliche Zustandsübertragung") und ÄPI"für Äpplication Programming Interface"(auf Deutsch: Änwendungsprogrammierschnittstelle"). Eine REST-API zeichnet sich dadurch aus, dass sie eine einheitliche Schnittstelle zwischen Server und Client bietet. Dies wird durch die "Routes"(auf Deutsch: "Routen") geschaffen. Wenn der Client Requests an diese Routen sendet werden Aktionen auf dem Server ausgeführt. Es ist auch möglich, dass anschließend der Client Daten vom Server erhält. (vgl. URLPI04)

In unserer Diplomarbeit stellt der Microcontroller als Webserver eine REST-API zur Verfügung um so einheitlich Daten mit dem Client (Smartphone) auszutauschen. Folgende Routen sind dabei auf dem Webserver aufrufbar:

Route	Anfragen-Typ	Funktion
/getInfo	GET	Client bekommt Infos vom Microcontroller
/getAvailableNetworks	GET	Client bekommt eine Liste im JSON-Format, gefüllt mit SSID und RSSI (Stärke) von verfügbaren Netzwerken in der Nähe des Microcontrollers
/setWiFiCredentials	POST	Client sendet SSID und Passwort des gewünschten Netzwerks an den Microcontroller
/setStreamUrl	POST	Client sendet die URL des gewünschten Audio-Streams an den Microcontroller

Anwendung fürs Audio-Streaming
HTTP wird oft zum Streamen von Daten eingesetzt. Dies können Audio- oder auch Videodaten sein.

Beim Streaming wird grundsätzlich zwischen Live-Streaming und On-Demand-Streaming unterschieden. Beim Livestreaming werden die Daten gleich nach dem Erstellen an den Client gesendet. Ein Beispiel dafür ist das Streaming eines Live-Events. Das Video, welches von der Kamera eingefangen wird, wird direkt an die Clients gesendet. Das Gegenteil zum Livestreaming ist das On-Demand-Streaming. Dabei werden fertige Daten (z.B. Filme, Musik) auf einem Server gespeichert. Auf Anfrage eines Clients, werden diese in Teilstücke zerlegt und Stück für Stück an den Client gesendet. Dabei hängt der Client diese Stücke wieder zusammen und kann sie somit als Ganzes wiedergeben. Dies hat den Vorteil, dass die Daten nicht (oder nur kurz) auf dem Client gespeichert werden und es somit ressourcenschonend ist. Der Nachteil dabei ist, dass gerade bei größeren Datenmengen eine hohe Bandbreite benötigt wird. Dabei werden die Daten auch meist nicht (bzw. nur kurz) auf dem Client gespeichert. (vgl. URLPI05)

In unserer Diplomarbeit muss der Microcontroller fähig sein, Audiodaten von Livestreams und auch von On-Demand-Streams zu erhalten.Livestreams könnten dabei von Radiosendern stammen und On-Demand-Streams von Musikanbietern.

I2S
Das Inter IC Sound Protocol wird verwendet, um Stereo-Audio-Daten zwischen ICs auszutauschen. Es benötigt für die Datenübertragung folgende Leitungen:

- Taktleitung
- Wortauswahl

- mindestens eine Datenleitung

Die Datenübertragung erfolgt seriell und synchron. Seriell bedeutet, dass die Daten nur durch eine Leitung (die Datenleitung) übertragen werden. Synchron bedeutet, dass die Daten in einem bestimmten Takt übertragen werden. Dieser Takt wird von der Taktleitung vorgegeben. Die Leitung für die Wortauswahl wählt den Stereokanal aus (links oder rechts). (vgl. https://www.mikrocontroller.net/articles/I2S)
In unserem Projekt wird das I2S Protokoll verwendet, um die digitalen Stereo-Audio-Daten vom Microcontroller an den Digital-Analag-Wandler zu übertragen. Dabei werden die digitalen Buffer, die der Microcontroller vom Audio-Stream erhält, mittels I2S an den Digital-Analog-Wandler gesendet, welcher die digitalen Daten in analoge Daten umwandelt, so dass diese dann anschließend auf der Lautsprecherbox ausgegeben werden können.

3.4 Auswahl Softwaretools

3.4.1 Einleitung

In diesem Kapitel geht es um die Recherche und Auswahl von geeigneten Softwaretools, welche für die App-Entwicklung, als auch für die Entwicklung der Software des Microcontrollers verwendet werden. Zusätzlich werden auch die Softwaretools zum Schreiben dieser Diplomarbeit kurz beschrieben.
LaTeX
"LaTeX ist ein hochwertiges Schriftsatzsystem, das Funktionen für die Erstellung technischer und wissenschaftlicher Dokumentationen enthält. LaTeX ist der Defacto-Standard für die Kommunikation und Veröffentlichung von wissenschaftlichen Dokumenten."(Übersetzung des englischen Originals von: https://www.latex-project.org/)
Wir haben uns für das Schreiben unserer Diplomarbeit in LaTeX entschieden, weil es sich sehr gut für wissenschaftliche Arbeiten eignet und wir somit schon damit vertraut sind, wenn wir es in der Zukunft brauchen.
draw.io "draw.io ist eine kostenlose Online-Diagrammsoftware zur Erstellung von Flussdiagrammen, Prozessdiagrammen, Organigrammen, UML, ER und Netzwerkdiagrammen."(Übersetzung des englischen Originals von: https://app.diagrams.net/)
Wir haben alle Diagramme unserer Diplomarbeit in draw.io erstellt, weil es einfach zu handhaben ist und es eine große Auwahl an Diagrammtypen und Formen gibt.
Visual Studio Code
"Visual Studio Code ist ein leichtgewichtiger, aber leistungsstarker Quellcode-Editor, der auf Ihrem Desktop läuft und für Windows, macOS und Linux verfügbar ist. Er bietet integrierte Unterstützung für JavaScript, TypeScript und Node.js und verfügt über ein umfangreiches Ökosystem von Erweiterungen für andere Sprachen und Laufzeiten (wie C++, C#, Java, Python, PHP, Go, .NET)."(Übersetzung

des englischen Originals von: https://code.visualstudio.com/docs)
Wir haben Visual Studio Code als IDE für unsere Diplomarbeit gewählt, weil durch die unzähligen Erweiterungen viele verschieden Programmiersprachen und Bibliotheken unterstützt und wir auch schon etwas Erfahrung damit haben.
GitHub
"GitHub ist eine webbasierte Schnittstelle, die Git verwendet, die Open-Source-Software zur Versionskontrolle, mit der mehrere Personen gleichzeitig separate Änderungen an Webseiten vornehmen können. Wie Carpenter anmerkt, fördert GitHub die Zusammenarbeit von Teams bei der Erstellung und Bearbeitung von Website-Inhalten, da es eine Zusammenarbeit in Echtzeit ermöglicht."(Übersetzung des englischen Originals von: https://digital.gov/resources/an-introduction-github/)
Wir verwenden GitHub für die Verwaltung unseres Codes und unserer Dokumente. Der Vorteil dabei ist, dass jedes Projektmitglied auf seinem lokalen PC an den Dokumenten arbeiten kann und die Änderungen dann per GitHub synchronisiert werden können.
DeepL Wir verwenden DeepL um englische Texte, welche für unsere Diplomarbeit relevant sind, ins Deutsche zu übersetzen. Wir haben uns für DeepL entschieden weil dieser einer der genauesten Übersetzer ist und man dießen außerdem kostenlos nutzen kann.

3.4.2 Bibliotheken Microcontroller

Im folgenden werden die verwendeten Bibliotheken im Code des Microcontrollers aufgezählt und kurz beschrieben:
Arduino
Die Arduino-Bibliothek wird verwendet um den ESP32 ähnlich wie einen Arduino programmieren zu können. Es erleichtert dabei die Programmierung enorm, vorallem dann, wenn man schon Vorerfahrung mit der Programmierung von Arduinos hat.
WiFi
`https://github.com/arduino-libraries/WiFi`
Die WiFi-Bibliothek wird verwendet, um die Funktionen der eingebauten WiFi-Antenne des ESP32 zu verwenden. Der ESP32 kann dabei entweder als Access Point oder als Client fungieren. Wenn er als Access Point fungiert, stellt er ein eigenes WiFi-Netzwerk bereit, mit dem sich andere Geräte verbinden können und der ESP32 somit einen Host darstellt. Als Client kann er sich mit anderen WiFi-Netzwerken bzw. Access Points verbinden. In unserem Projekt fungiert der ESP32 sowohl als Access Point, als auch als Client.
ArduinoJson
Die ArduinoJson-Bibliothek wird verwendet, um Daten in das JSON vormat zu kodieren. Der Vorteil dabei ist, dass JSON ein weit verbreitetes Format in der Informatik ist und deshalb mit vielen Schnittstellen funktioniert. In unserem Projekt wird die ArduinoJson-Bibliothek für den einheitlichen Datenaustausch

zwischen Webserver (ESP32) und Client (Smartphone) verwendet.
WebServer
Die WebServer-Bibliothek wird verwendet, um einen Webserver auf dem ESP32 bereitzustellen. Dieser ist wichtig für den Datenaustausch mittels HTTP, zwischen ESP32 und Smartphone. Mithilfe des Webserver wird einerseits das WiFi-Passwort des gewählten WLANs, als auch die URL des Audiostreams an den ESP32 geleitet. In die andere Richtung, werden grundlegende WiFi-Informationen und verfügbare WiFi-Netzwerke vom ESP32 bereitgestellt.
Audio
Die Audio-Bibliothek wird verwendet, um die Audio-Daten mittels I2S-Protokoll an den Digital-Analog-Wandler zu senden und diesen zu konfigurieren. Der ESP32 verfügt bereits standartmäßig über Funktionen, mit deren Hilfe man Audiodaten mittels I2S übertragen kann. Allerdings bereitet dies einen viel höheren Aufwand für Konfiguration usw. Daher verwenden wir in unserem Projekt die Audio-Bibliothek.

3.4.3 Softwaretools Smartphoneapp

Für die Entwicklung der Smartphoneapp wurde das "React Native Framework verwendet. Mithilfe von React Native ist es möglich eine zentrale Applikation zu entwickeln und diese dann auf mehreren Plattformen wie IOS, Android und auch im Web zu verwenden. React Native basiert auf React, welches ein Framework für die Frontend-Entwicklung ist. Außerdem wird die Radio-Browser-API für die bereitstellung diverser Internetradios verwendet.

4 Entwicklung

4.1 Design Platine

4.2 Bestückung Platine

4.3 Entwicklung Software Adapter

In diesem Kapitel wird der Übergang der Planung in die Entwicklung der Software des Adapters beschrieben. Zur Entwicklung der Software des Microcontrollers wird die IDE Visual Studio Code in Verbindung mit PlatformIO verwendet. Um die Entwicklung einfacher und übersichtlicher zu gestalten, wird die Arduino-Bibliothek in Verbindung mit C++ verwendet.

4.3.1 Programmablauf

Der Ablauf des Programmes wird mit folgendem UML-Ablaufdiagramm veranschaulicht:

Start
Der Microcontroller wird durch einmaliges Drücken auf den Knopf aus dem Standby (Deep Sleep) Modus erweckt.

Lesen der WiFi-Zugangsdaten
Es wird im EEPROM des Microcontrollers nach einer gespeicherten SSID und nach einem gespeicherten Passwort gesucht. Wenn die Zugansdaten nicht vorhanden sind wird die Status-LED auf rot geschalten. Anschließend werden keine weiteren Prozesse ausgeführt, bis der/die Benutzer/in mit Druck auf den Knopf in den Konfigurationsmodus schaltet. Wenn die Zugangsdaten allerdings gespeichert sind versucht der Microcontroller sich als Client mit dem WLAN zu verbinden. Wenn dies erfolgreich ist, wird in den Standardmodus gewechselt. Wenn die Verbindung allerdings fehlschlägt, wird in den Konfigurationsmodus gewechselt.

Standardmodus
Sobald der Standardmodus aktiviert wird, wechselt die Farbe der Status-LED auf grün. Anschließend wird aus dem EEPROM ausgelesen, ob bereits eine Stream-URL gespeichert ist. Wenn dies der Fall ist, fungiert der Microcontroller als HTTP-Client und empfängt den Stream. Anschließend wird dieser mittels I2S-Protokoll an den Digital-Analog-Wandler übertragen. Wenn allerdings noch keine Stream-URL gespeichert ist, wartet der Microcontroller bis eine Stream-URL vom Client gesendet wird.

Konfigurationsmodus
Wird der Konfigurationsmodus mit Druck auf den Taster aktiviert, wechselt die Status-LED ihre Farbe auf blau. Anschließend wird der WiFi-Chip in den AP-Modus gestellt. Somit bietet der Microcontroller einen Access Point, auf den sich andere Geräte als Clients verbinden können. Dies wird benötigt, dass die Clients in weiterer Folge WiFi-Zugangsdaten an den Microcontroller senden können. Um die Daten zu empfangen wird ein WebServer auf dem Microcontroller gestartet. Dieser erhält dann die WiFi-Zugangsdaten mittels HTTP-Protokoll.

Wechsel in normalen Modus
Sobald der Microcontroller die WiFi-Zugangsdaten vom Client erhalten hat, schreibt er diese in den EEPROM. Anschließend wird der Microcontroller neu gestartet und der Programmablauf fängt wieder von vorne an.

4.3.2 Klassen

Bei der Entwicklung der Microcontroller-Software wurde aufgrund der Übersichtlichkeit und um die Design-Patterns der Softwareentwicklung einzuhalten, auf objektorientierte Programmierung gesetzt. Das folgende UML-Klassendiagramm veranschaulicht die Beziehung der verschiedenen Klassen zueinander: hier kommt das UML-Klassendiagramm her Im folgenden Teil werden die Klassen und deren Funktionen noch näher beschrieben:

StatusLED
Mithilfe der Klasse StatusLED wird die RGB-LED, welche am ESP32 angeschlossen ist, gesteuert.
NetworkManager
Die Klasse NetworkManager kümmert sich um alle Funktionen, die mit dem WiFi des ESP32 zu tun haben.
AudioManager
Die Klasse AudioManager regelt hauptsächlich das Senden der Audiodaten, vom ESP32 an den Digital-Analog Wandler, mittels I2S-Protokoll.
Log
Die Klasse Log beinhaltet ein Json Dokument in dieses Logs geschrieben werden können. Dies können Status-Logs oder auch Fehler-Logs sein.

4.4 Entwicklung Smartphone-App

In diesem Kapitel wird der Übergang der Planung in die Entwicklung der Smartphone-App beschrieben.

4.5 Design Adaptergehäuse

4.6 Fertigung Adaptergehäuse

5 Testen und Fehlerbehebung

5.1 Testen des Gesamtsystems

5.2 auftretende Fehler beheben

5.3 Test auf Cybersecurity

In diesem Kapitel wird der gesamte Code auf Sicherheitslücken getestet.

5.4 Auftretende Sicherheitslücken schließen

In diesem Kapitel werden die bei den Tests aufgetretenen Sicherheitslücken geschlossen.

6 Einzelnachweise

6.1 Literaturverzeichnis

6.2 Abbildungsverzeichnis

6.3 Anhang