

# Buzzer-System

## Eingebettete Systeme

des Studiengangs Technische Informatik  
an der Hochschule Bochum Campus Velbert Heiligenhaus

von  
Jan Küpper

Fachsemester: Wintersemester 2020/21

Prüfer: Prof. Dr. rer. nat. Peter Gerwinski

Matrikelnummer: 018100841

Ort, Datum: Remscheid, 16.10.2023

# Inhaltsverzeichnis

<b>1. Einleitung</b>	<b>4</b>
1.1. Motivation	4
1.2. Aufgabenstellung und Zielsetzung	4
1.3. Projektkontext	4
1.4. Vorgehensweise und Aufbau der Arbeit	5
<b>2. Konzeption des Buzzer-System</b>	<b>6</b>
<b>3. Anwendungsfälle</b>	<b>7</b>
3.1. Verbindung herstellen	8
3.2. Anzahl Verbindungen anzeigen	8
3.3. Spiel starten	9
3.4. Spiel 2 steuern	9
3.5. Spiel 2 Status anzeigen	10
3.6. Spiel beenden	10
<b>4. Aufbau und Implementierung des Buzzer-Systems</b>	<b>11</b>
4.1. Hardware Zusammenstellung	11
4.1.1. Auswahl eines Mikrocontrollers	11
4.1.2. Entwicklung des Schaltplans	11
4.1.3. Auswahl der Buzzer-Komponenten	12
4.2. Software-Entwicklung und Integration	13
4.2.1. Eigenständige Softwareentwicklung für den Buzzer	13
4.2.2. Aufbau einer Kommunikation mit Teilnehmern	13
4.2.3. Nutzerfreundliche Schnittstelle und Funktionalitäten des Buzzers	14
<b>5. Qualitätssicherung</b>	<b>16</b>
<b>6. Fazit und Ausblick</b>	<b>17</b>
<b>7. Anhänge</b>	<b>18</b>
<b>Literatur- und Quellenverzeichnis</b>	<b>19</b>
<b>Abkürzungsverzeichnis</b>	<b>20</b>
<b>Abbildungsverzeichnis</b>	<b>21</b>
<b>Eidesstattliche Erklärung</b>	<b>23</b>

# 1. Einleitung

Diese Ausarbeitung beschäftigt sich mit dem Entwicklungsprozess eines Buzzer-Systems und erläutert die Funktionsweise im Detail. Die Funktionalität des Buzzer-Systems geht dabei über die herkömmlichen Funktionen eines Buzzers hinaus. Es wird insbesondere auf die technischen Aspekte der Umsetzung eingegangen, außerdem werden Anwendungsmöglichkeiten und Erweiterungen des Systems vorgestellt.

## 1.1. Motivation

Oftmals kann das mühsame Durchführen von Spielrunden, insbesondere das Auswerten von Punkten und das Verteilen von Aufgaben, den Spielspaß mindern. Ein Buzzer soll hier Abhilfe schaffen, um das Erlebnis von Gesellschaftsspielen zu verbessern. Er soll als digitaler Spielleiter fungieren und viele Aufgaben automatisieren. Dadurch soll das Spielerlebnis noch angenehmer und flüssiger gestaltet werden. Darüber hinaus bietet ein Buzzer auch die Möglichkeit, neue Spiele zu entwickeln und bestehende Spiele zu erweitern. Dieses Projekt soll zeigen, wie einfach und preiswert es sein kann, ein eigenes digitales Spiel zu entwickeln und somit das Spielerlebnis zu verbessern.

## 1.2. Aufgabenstellung und Zielsetzung

Das Ziel dieses Projekts ist die Entwicklung eines interaktiven Spiels, das entweder alleine oder als Gemeinschaftsspiel gespielt werden kann. Der Buzzer soll verschiedene Funktionen wie ein Reaktionsspiel oder Quiz steuern. Die Spieler haben die Möglichkeit, an einer Spiel-Session teilzunehmen und ihre Spielstände über das Netzwerk auszutauschen, um ein umfassendes Spiel aufzubauen und auszuwerten. Durch diese Funktion können die Spieler nicht nur ihr eigenes Spiel verbessern, sondern auch gegen andere Spieler antreten und ihr Können unter Beweis stellen. Das Spiel kann entweder als Reaktionsspiel alleine oder als Quiz in einer Gruppe gespielt werden. Die Technik soll ihren Platz in einem 3D gedruckten Gehäuse finden.

## 1.3. Projektkontext

Um ein möglichst spannendes und herausforderndes Spielerlebnis zu gewährleisten, braucht es mehrere Buzzer, Spielmodi sind Varianten oder Regelsätze, um verschiedene Spielerlebnisse zu bieten. Das Spiel kann verändert, die Schwierigkeit angepasst oder der Fokus des Spiels verlagert werden. Die Spielmodi sind Einzelspieler-, Mehrspielermodi. Diese Vielfalt an Spielmodi ermöglicht es, verschiedene Spielergruppen anzusprechen und das Spielerlebnis abwechslungsreich zu gestalten. Ein entscheidender Schritt bei der Entwicklung dieses interaktiven Gerätes ist die Programmierung eines Arduino-Mikrocomputers. Der Arduino bietet eine programmierbare Plattform, auf der sich maßgeschneiderte Funktionen erstellen lassen. Arduino ist eine Programmiersprache ähnlich wie C++, um die Verhaltensweisen des Geräts festzulegen. Der Arduino kann Sensoren auslesen, Aktoren steuern und Benutzereingaben verarbeiten.

## 1.4. Vorgehensweise und Aufbau der Arbeit

Im zweiten Kapitel folgt die Beschreibung der Grundlagen und der Anwendung des Buzzer-System. Kapitel 3 beinhaltet die Anwendungsfälle. Darauf aufbauend stellt Kapitel 4 einen Überblick der Realisierung der Anwendung dar, wobei insbesondere auf die softwareseitige Umsetzung eingegangen wird. Kapitel 5 beschreibt die Sicherstellung der Funktionalität und der gewünschten Anforderungen. Ein Fazit der Arbeit und ein kurzer Ausblick auf die mögliche Weiterentwicklung des Buzzer-System beschließen die Arbeit in Kapitel 6.

## 2. Konzeption des Buzzer-System

Digitale Spielgeräte haben in den letzten Jahren eine immer wichtigere Rolle im Bereich des Unterhaltungswesen eingenommen. Sie bieten interaktive Möglichkeiten, um Spaß zu haben, Fähigkeiten zu entwickeln und soziale Interaktionen zu fördern. Diese Geräte reichen von einfachen Konsolen bis hin zu komplexen Virtual-Reality-Systemen (s. [demodern\_Games]). In diesem Kapitel ist die grundlegende Konzeption des Buzzer-Systems beschrieben, das sich als vielseitiges und interaktives Spielgerät in diese Reihe digitaler Unterhaltungsprodukte einreicht. Im Wesentlichen besteht das Buzzer-System aus drei grundlegenden Phasen: der Eingabe, der Verarbeitung und der Ausgabe. Diese Phasen sind entscheidend für die reibungslose Funktionsweise des Systems und ermöglichen es den Spielern, ein spannendes und unterhaltsames Spiel zu genießen. In der Eingabe Phase muss das System Interaktionen der Spieler erfassen. Das Herzstück des Systems ist der Taster im Buzzer, der als Eingabegeräte dient. Die Spieler drücken auf die Buzzer, um ihre Antwort oder ihre Auswahl kundzutun. Das System erfasst diese Eingaben präzise und sorgt dafür, dass sie für die Verarbeitung bereitstehen. Nachdem die Eingaben erfasst wurden, erfolgt die Validierung und Verarbeitung. Das System überprüft die eingegangenen Signale und sorgt für einen fairen und korrekten Ablauf des Spiels. Dies umfasst auch die Aktualisierung aller anderen Buzzer und die Steuerung des Spielablaufs. Die Ausgabe-Phase ist entscheidend für die Spielerfahrung. Hier werden die Ergebnisse der Verarbeitung präsentiert. Dies umfasst die Anzeige von Signalen, um den Spielverlauf zu begleiten, und die Aktualisierung des Spielstands, damit die Spieler ihre Leistung nachvollziehen können. Die sorgfältige Konzeption und Umsetzung dieser Eingabe-Verarbeitung-Ausgabe-Phasen gewährleisten, dass das Buzzer-System eine reibungslose und unterhaltsame Spielerfahrung bietet. In den folgenden Kapiteln wird genauer auf die technischen Details und Designüberlegungen eingegangen, die in die Entwicklung dieses Systems eingeflossen sind. Dabei werden auch die verschiedenen Anwendungsfälle beschrieben.

### 3. Anwendungsfälle

In diesem Kapitel wird eine Beschreibung der allgemeinen und für Spiel 2 spezifischen Anwendungsfälle des Buzzer-Systems vorgenommen. Das System bietet die Möglichkeit, Spielsitzungen zu starten und unterschiedliche Spiele durchzuführen. Die Anwendungsfälle werden in Abbildung 2 dargestellt. Dies ermöglicht einen klaren Überblick über die Funktionsweise und die vielfältigen Einsatzmöglichkeiten dieses Systems.

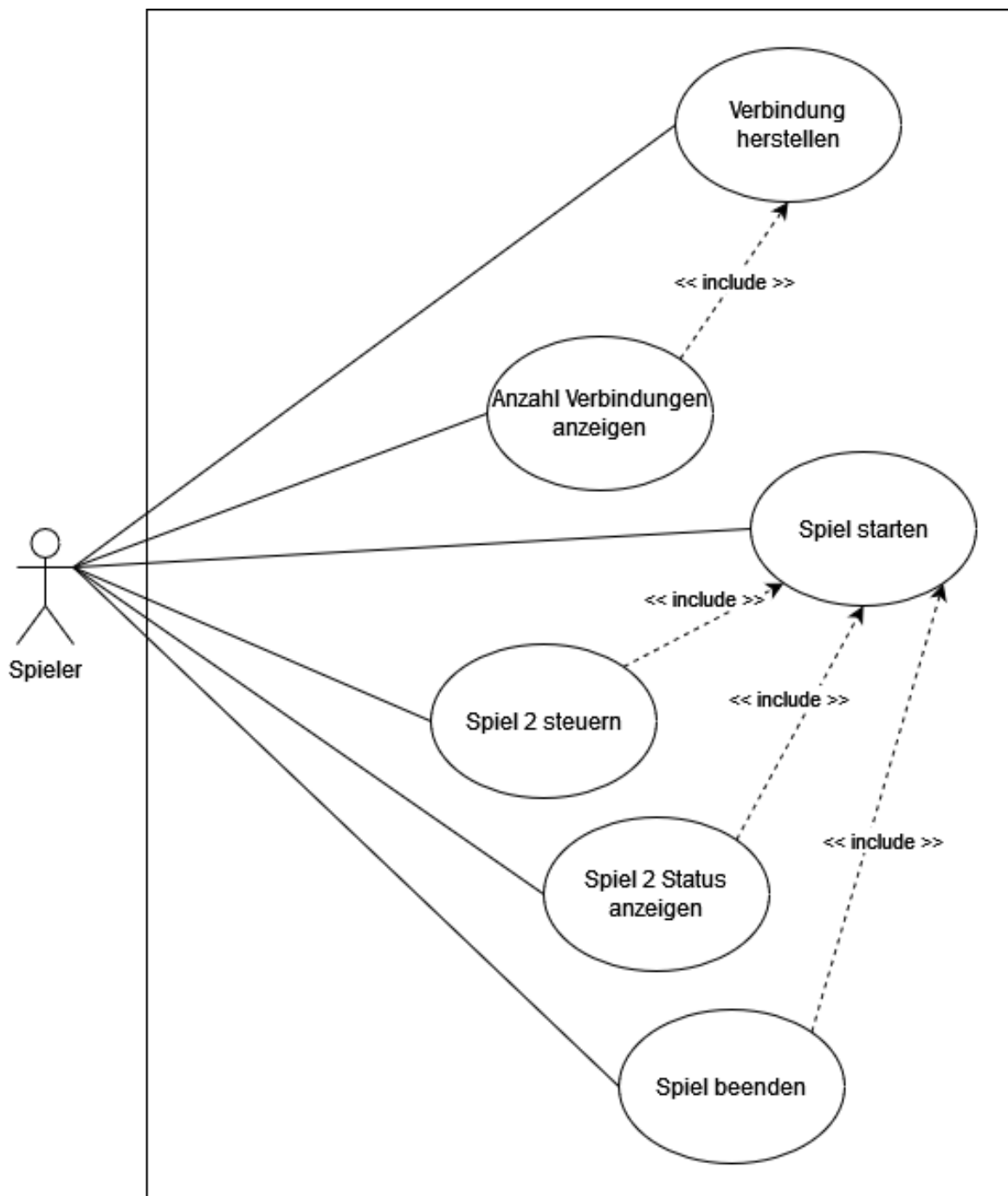


Abb. 1 Anwendungsfalldiagramm

### 3.1. Verbindung herstellen

Ein Spiel besteht aus mehreren Buzzern, um zu spielen, müssen diese mit dem Netzwerk des Host-Buzzer verbunden werden.

**Name:** Verbindung herstellen

**Ziel(e):** Verbindung mit dem Netzwerk vom Host-Buzzer und diesem herstellen

**Vorbedingung:** Host-Buzzer und Client-Buzzer sind eingeschaltet und Client blinkt schnell

**Nachbedingung Erfolg:** Verbindung mit Host-Buzzer ist hergestellt

**Primärer Akteur:** Spieler

**Weitere Akteure:** Host-Buzzer, Client-Buzzer

**Auslösendes Ereignis:** Lust etwas zu spielen

**Beschreibung:**

1. Smartphone mit Client-Buzzer Netzwerk verbinden
2. Verbindung zum Host-Buzzer über die Webseite des Client-Buzzer herstellen

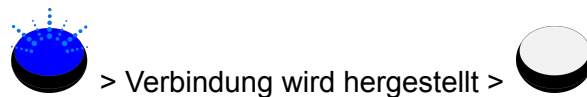


Abb. 2 Verbindung herstellen

### 3.2. Anzahl Verbindungen anzeigen

Die Anzahl der verbundenen Buzzer kann an dem Host-Buzzer angezeigt werden.

**Name:** Anzahl Verbindungen anzeigen

**Ziel(e):** Anzahl der verbundenen Clients mit dem Host anzeigen

**Vorbedingung:** Gerät ist eingeschaltet

**Nachbedingung Erfolg:** die blaue LED zeigt durch aufblinken die Anzahl der Clients an

**Primärer Akteur:** Spieler

**Weitere Akteure:** Host-Buzzer

**Auslösendes Ereignis:** informieren über Anzahl Spieler

**Beschreibung:**

1. Host-Buzzer für 10 Sekunden drücken

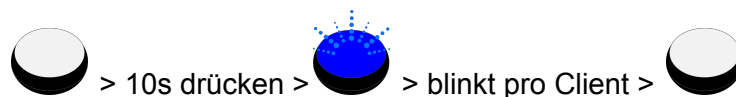


Abb. 3 Anzahl Verbindungen anzeigen

### 3.3. Spiel starten

Mit den Buzzern kann in mehreren Spielmodi gespielt werden, an dem Host-Buzzer kann ein beliebiges Spiel ausgewählt werden.

**Name:** Spiel starten

**Ziel(e):** ein Spiel starten um es zu spielen

**Vorbedingung:** Host-Buzzer muss eingeschaltet sein

**Nachbedingung Erfolg:** ein Spiel wurde ausgewählt und kann gespielt werden

**Primärer Akteur:** Spieler

**Weitere Akteure:** Host-Buzzer

**Auslösendes Ereignis:** es besteht der Wunsch nach einem bestimmten Spiel

**Beschreibung:**

1. Host-Buzzer für 5 Sekunden drücken
2. mit kurzem Druck auf den Host-Buzzer Index des Spiel auswählen
3. Host-Buzzer für 5 Sekunden ausgewähltes Spiel starten

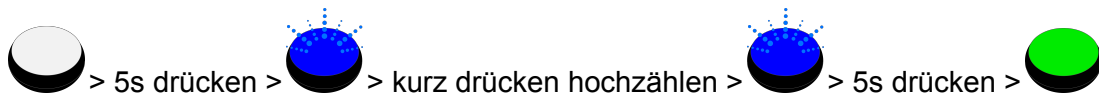


Abb. 4 Spiel starten

### 3.4. Spiel 2 steuern

Spiel 2 ist ein Quiz, wer zuerst drückt darf eine Antwort geben, ein Punkt wird auf dem Buzzer angerechnet, auf den zurückgesetzt wird.

**Name:** Spiel 2 steuern

**Ziel(e):** einen Spieldurchlauf für sich entscheiden

**Vorbedingung:** Spiel 2 muss ausgewählt worden sein

**Nachbedingung Erfolg:** alle Buzzer sind zurückgesetzt und einer hat einen Punkt bekommen

**Primärer Akteur:** Spieler

**Weitere Akteure:** Host-Buzzer oder Client-Buzzer

**Auslösendes Ereignis:** Spieler möchte die Antwort auf eine Frage geben

**Beschreibung:**

1. Host-Buzzer oder Client-Buzzer kurz drücken
2. Host-Buzzer oder Client-Buzzer 2 Sekunden drücken zum zurücksetzen und Punkt anzurechnen



Abb. 5 Spiel 2 steuern



### 3.5. Spiel 2 Status anzeigen

Der Spielstatus von Spiel 2 kann durch die LED am jeweiligen Buzzer angezeigt werden.

**Name:** Spiel 2 Status anzeigen

**Ziel(e):** herausfinden wie der Spielstand ist

**Vorbedingung:** Spiel 2 muss ausgewählt worden sein

**Nachbedingung Erfolg:** der Buzzer zeigt durch Blinken die Punktzahl an

**Primärer Akteur:** Spieler

**Weitere Akteure:** Host-Buzzer oder Client-Buzzer

**Auslösendes Ereignis:** informieren über aktuellen Spielstand

**Beschreibung:**

1. Buzzer für 4 Sekunden drücken

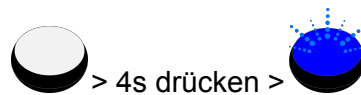


Abb. 6 Spielstatus abrufen

### 3.6. Spiel beenden

Um ein Spiel neu starten oder zu ändern, muss das aktuelle Spiel beendet werden.

**Name:** Spiel beenden

**Ziel(e):** das aktuell ausgewählte Spiel beenden

**Vorbedingung:** ein Spiel muss ausgewählt worden sein

**Nachbedingung Erfolg:** es ist kein Spiel mehr ausgewählt, alle Buzzer sind zurückgesetzt

**Primärer Akteur:** Spieler

**Weitere Akteure:** Host-Buzzer

**Auslösendes Ereignis:** Spieler möchte ein anderes Spiel spielen

**Beschreibung:**

1. Buzzer für 8 Sekunden drücken



Abb. 7 Spiel beenden

## 4. Aufbau und Implementierung des Buzzer-Systems

Die folgenden Abschnitte beschreiben die praktische Umsetzung des Buzzer-Systems. Hierbei liegt der Fokus darauf, das entwickelte Konzept in die Realität umzusetzen. Die Implementierung gliedert sich in mehrere spezifische Bereiche. Beginnend mit der Hardware-Zusammenstellung, gefolgt von der Software-Entwicklung und Integration wird der gesamte Ablauf im folgenden beschrieben.

### 4.1. Hardware Zusammenstellung

Der erste Schritt in der Realisierung des Buzzer-Systems konzentriert sich auf die hardwareseitige Umsetzung. Dies beinhaltet die Auswahl des optimalen Mikrocontrollers, die Entwicklung eines passenden Schaltplans und die Auswahl der Buzzer-Komponenten. Die sorgfältige Planung und Zusammenstellung dieser Hardware-Elemente legt den Grundstein für das Buzzer-System.

#### 4.1.1. Auswahl eines Mikrocontrollers

Für die Umsetzung des Projekts ist die Auswahl des passenden Mikrocontrollers von großer Bedeutung. Ein wichtiges Kriterium bei der Wahl des Mikrocontrollers ist die Möglichkeit, ihn sowohl als Client als auch als Server agieren zu lassen. Hierfür wird eine periphere Baugruppe im Mikrocontroller benötigt. Aus diesem Grund wurde der ESP32 als passender Mikrocontroller ausgewählt. Der ESP32 verfügt über ein integriertes Wireless Local Area Network und ist somit in der Lage, sowohl als ein WLAN Access Point konfiguriert zu werden, als auch eine Verbindung zu einem WLAN herzustellen (s. [Jen17]). Zusätzlich verfügt er über Bluetooth sowie mehrere Analogeingängen und IO-Pins, beispielsweise um Signale wie einen Tastenanschlag zu übertragen und visuelle Rückmeldungen über LEDs darzustellen. Den ESP32 und weitere Komponenten gibt es verbaut auf einem Entwicklerboard, das auch die Programmierung des Mikrocontrollers über USB ermöglicht. Das Board ist mit der Arduino IDE, Lua, MicroPython und Node kompatibel und eignet sich aufgrund seines geringen Ruhestroms von weniger als 5  $\mu\text{A}$  auch für den Einsatz mit Batterie oder Akku. Insgesamt bietet der D1 Mini ESP32 alle erforderlichen Funktionen und Eigenschaften, um das Projekt erfolgreich umzusetzen.

#### 4.1.2. Entwicklung des Schaltplans

Der Schaltplan des Buzzers besteht aus drei LEDs und einem mechanischen Taster, der mit einem Pulldown-Widerstand verbunden ist. Der Taster fungiert als Auslöser für das Spiel und gibt ein Signal an den Mikrocontroller, wenn er gedrückt wird. Der Pulldown-Widerstand mit einem Wert von 10 k $\Omega$  sorgt dafür, dass der Eingangspin des Mikrocontrollers ein eindeutiges Signal erhält und nicht durch Störungen verfälscht wird. Die LEDs sind ebenfalls mit dem Mikrocontroller verbunden, sie dienen zur Anzeige des Status.

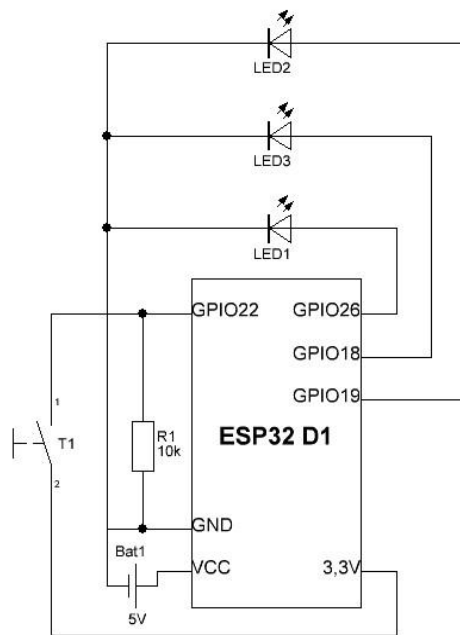


Abb. 8 Schaltplan

#### 4.1.3. Auswahl der Buzzer-Komponenten

Für den Buzzer werden verschiedene Komponenten benötigt, um ein funktionsfähiges Gerät zu erstellen. Das Gehäuse des Buzzers ist aus dem 3D-Drucker, dieser druckt in mehreren Schichten 3D-CAD Daten aus PLA.. Das Gehäuse bietet der Hardware Schutz und ist gleichzeitig ein mechanischer Taster, der als Signalgeber dient. Um die notwendigen elektronischen Bauteile wie im vorherigen Abschnitt beschrieben zu verbinden, wurde eine Lochrasterplatine und Kupferkabel verwendet. Der Buzzer verfügt über drei LEDs in den Farben Grün, Orange und Blau, die in das Oberteil eingelassen sind, sie zeigen den aktuellen Status an.

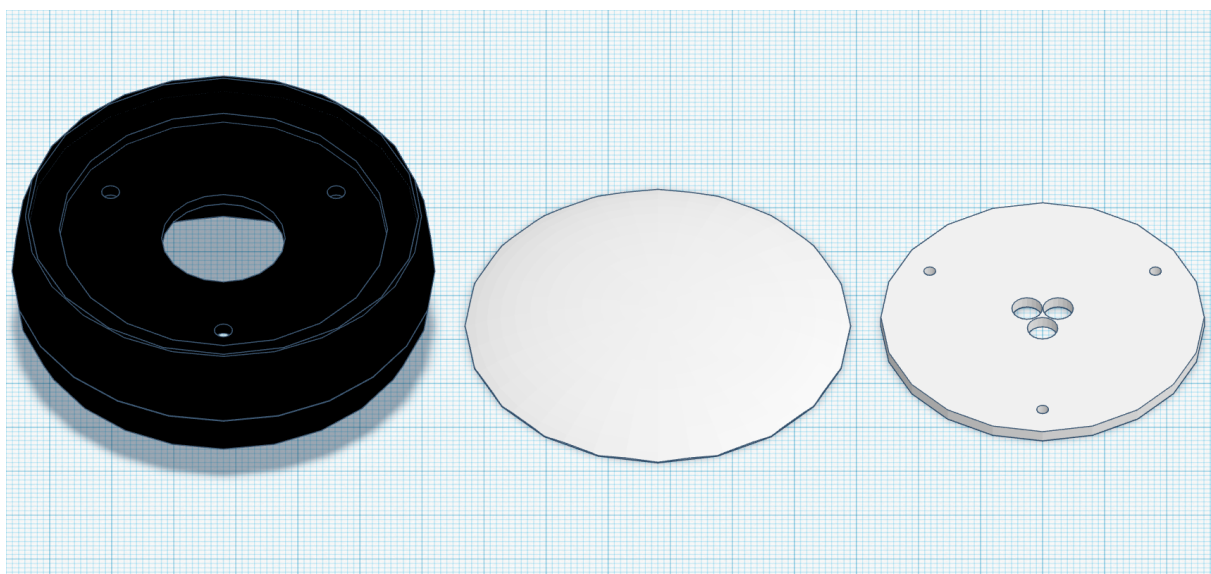


Abb. 9 3D-Gehäuse

## 4.2. Software-Entwicklung und Integration

Ein weiterer essentieller Teil des Buzzer-Systems ist die Software, die es steuert und mit anderen Komponenten interagieren lässt. In diesem Abschnitt wird die Softwareentwicklung und Integration beschrieben. Hierbei werden die Schritte zur Entwicklung der Software für das Buzzer-System erläutert. Zudem wird beleuchtet, wie die Kommunikation mit anderen Teilnehmern hergestellt wird und wie die Bedienung und Funktionalitäten des Buzzers in die Software integriert wurde, um das Gerät benutzerfreundlich zu machen.

### 4.2.1. Eigenständige Softwareentwicklung für den Buzzer

Um das Produkt zu realisieren, muss die Software für den Mikrocontroller entwickelt werden. Die Programmierung erfolgt in der Arduino IDE, der D1 Mini ESP32 ist mit dieser IDE kompatibel. Die Arduino-Plattform ist ein Open-Source-Software-Framework, das es Entwicklern ermöglicht, schnell und einfach elektronische Schaltungen sowie Mikrocontroller-Anwendungen zu programmieren und diesen auf den Mikrocontroller hochzuladen. Das Programm für den Mikrocontroller wird in der Arduino IDE geschrieben und basiert auf der Arduino-Sprache, die eine Variante von C++ ist (s. [EXP18]). Mit Hilfe der Arduino-Sprache und der Arduino-IDE können Entwickler Funktionen wie `digitalWrite()` oder `pinMode()` verwenden, um GPIOs des Mikrocontrollers zu steuern. Die `Setup()`-Funktion wird beim Start des Programms einmal ausgeführt und die `Loop()`-Funktion wird kontinuierlich ausgeführt, solange der Mikrocontroller aktiv ist. Das Programm ist in der Lage, das Signal des Tasters auf dem Eingangspin des Mikrocontrollers zu erkennen. Dafür wird der Eingangspin dauerhaft ausgelesen, bei einer steigenden Flanke des Signals wird das Signal aufgezeichnet. Bei fallender Flanke wird das Signal ausgewertet und das Ergebnis an die anderen Geräte im Netzwerk gesendet. Gleichzeitig wird dem Spieler ein Feedback durch die LEDs angezeigt. Um dem Spieler Feedback über den aktuellen Spielstand zu geben, werden verschiedene LEDs verwendet.

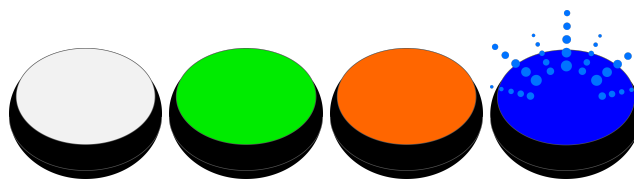


Abb. 10 LED Zustände

### 4.2.2. Aufbau einer Kommunikation mit Teilnehmern

Die Kommunikation mit anderen Teilnehmern ist ein wichtiger Aspekt des Buzzer-Systems. In der IDE können verschiedene Bibliotheken für die WiFi-Verbindung und die Kommunikation mit anderen Geräten eingebunden werden. Dafür wird eine Verbindung über IP und Port genutzt. Ein Gerät wird als Access Point programmiert und dient gleichzeitig als Server für eine Sitzung. Für die Inbetriebnahme ist eine direkte Verbindung notwendig, um eine Konfiguration durchzuführen, da die Zugangsdaten zum Netzwerk zu Beginn nicht bekannt sind. Nach einem Neustart versucht der Client, eine Verbindung mit dem letzten Server herzustellen und verwendet das Netzwerk der letzten Verbindung. Falls das Gerät erstmals verbunden werden soll oder der Verbindungsaufbau scheitert, stellt das System

einen Access Point bereit. Über eine Benutzeroberfläche auf einem Smartphone oder PC kann man sich dann mit dem richtigen Netzwerk verbinden. Die gesamte Funktion wird von der WIFI Manager-Bibliothek bereitgestellt. Die Kommunikation erfolgt über die Arduino WiFi Bibliothek. Ein- und ausgehende Verbindungen werden durch die Bibliothek behandelt. Darüber hinaus wurde ein benutzerdefiniertes Protokoll entworfen, um Befehle zum Setzen des Status zu übermitteln. Da die Verbindung nicht bemerkt, wenn sie zusammenbricht, wurde ein Keepalive hinzugefügt. Dies ist ein Mechanismus bei der Datenübertragung, der darauf abzielt, eine Netzwerkverbindung aufrechtzuerhalten und sich selbst von der Erreichbarkeit und Funktion eines Kommunikationspartners zu überzeugen (s. [Mar17]). Damit diese immer erreichbar sind, erfolgen Nebenläufigkeiten in separaten Tickern. Es gibt eine Bibliothek zum Erstellen von Tickern, die wiederholende Funktionen aufrufen können. Ziel dabei ist es, `delay()` durch nicht blockierende Funktionen zu ersetzen. Mit der Arduino-Tickerbibliothek können einfach Ticker-Rückrufe erstellt werden, mit denen eine Funktion in einem vorgegebenen Intervall aufgerufen wird. Dies funktioniert wie ein Thread, in dem bei Bedarf eine sekundäre Funktion ausgeführt wird.

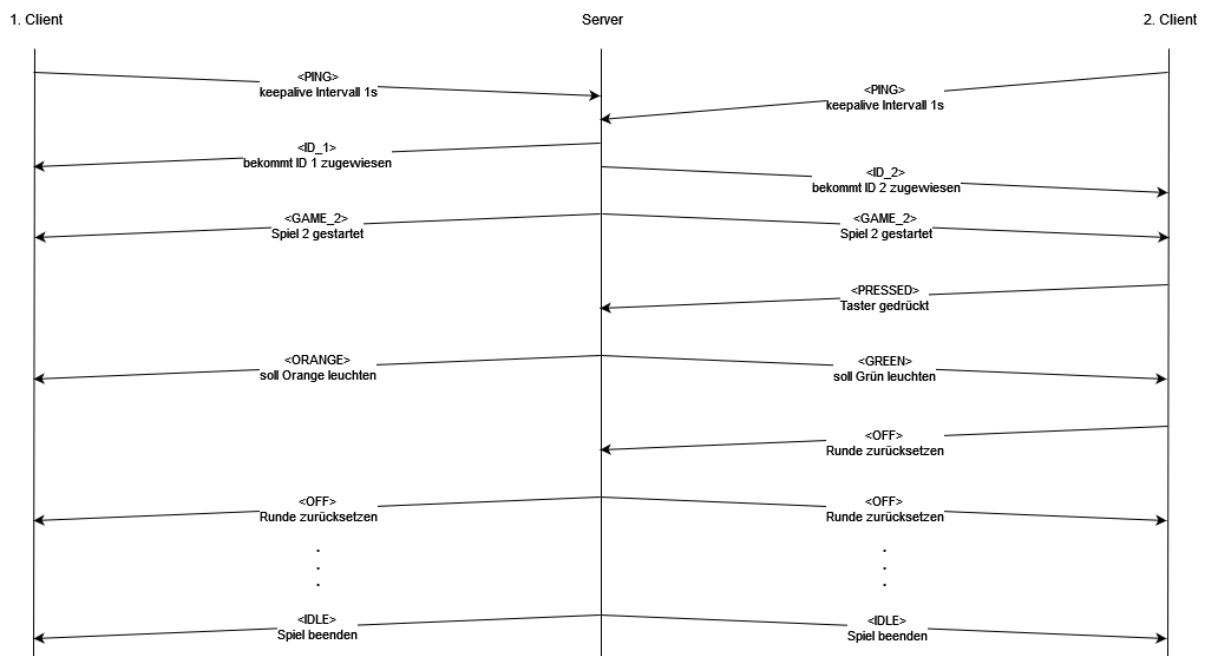


Abb. 11 Spiel 2 benutzerdefiniertes Protokoll

#### 4.2.3. Nutzerfreundliche Schnittstelle und Funktionalitäten des Buzzers

Die Interaktion mit dem Buzzer-System wird durch verschiedene, auf die Druckdauer des Taster abgestimmte Bedienoptionen ermöglicht. Dabei ergeben sich unterschiedliche Funktionen basierend auf der Länge des Tasten-Druck. Um versehentliche Auslösungen zu vermeiden, wurde besonderes Augenmerk darauf gelegt, dass die Benutzer diese Optionen gezielt auswählen können. Beim Betätigen des Tasters mit einem langen Druck > 15 Sekunden wird das Gerät neu gestartet, eine Funktion, die aufgrund ihrer Bedeutung vor unbeabsichtigter Aktivierung geschützt werden muss. Die Anzeige der Anzahl der angeschlossenen Clients erfolgt durch das Halten der Taste für 10 Sekunden, wodurch eine übersichtliche Informationsquelle geschaffen wird. Bei einem 5 Sekunden-Druck auf den Taster öffnen sich die Optionen, die dem Nutzer zur Verfügung stehen, und können nach

erfolgt Auswahl wieder geschlossen werden. Insbesondere für die Verwendung im Kontext von Spielen wurde die Funktionalität differenziert gestaltet. Ein Tastendruck von 0,05 - 4 Sekunden Dauer ermöglicht spezifische Aktionen im Spiel, die durch die Dauer des Auslösens abgegrenzt sind. Die Grundlage für diese vielfältigen Funktionalitäten bildet das vom Taster generierte Signal, das zwischen den Zuständen HIGH und LOW wechselt. Die anschließende Aktion wird maßgeblich von der Dauer des HIGH-Pegels bestimmt, der durch den Taster-Druck repräsentiert wird. Diese Art der Bedienung ist intuitiv und einfach. Im Falle eines des Reaktions-Spiel müssen die Spieler möglichst schnell auf bestimmte Reize wie ein Aufleuchten eines Buzzer reagieren. Beim Quiz-Spiel müssen die Spieler hingegen den Buzzer betätigen, wenn sie eine Antwort geben möchten, der schnellste Spieler darf antworten und über das Zurücksetzen der Runde wird der Punkt an den der zurücksetzt vergeben.

```

buttonState = digitalRead(BUTTON);
if(lastButtonState == LOW && buttonState == HIGH){ // button gedrückt
    pressedTime = millis();

    if(gameState == GAME_1){
        client.println("<PRESSED>");
    }
}
else if(lastButtonState == HIGH && buttonState == LOW) { // button losgelassen
    elapsedTime = millis() - pressedTime;

    if(elapsedTime > 15000){ //neustarten
        wifiManager.resetSettings();
        ESP.restart();
        delay(1000);
    }
    else if(elapsedTime >= 6000){ //id anzeigen
        tickerToggle.attach(0.5, toggle, LED1);
        tickerEnd.once(id, end);
    }
    else if(elapsedTime >= 4000 && gameState == GAME_2){ // Spiel 2 aktuelle Punktzahl
        tickerToggle.attach(0.5, toggle, LED1);
        tickerEnd.once(gameCount, end);
    }
    else if(elapsedTime >= 2000 && gameState == GAME_2){ // Spiel 2 Rücksetzen, wer zurücksetzt bekommt einen Punkt
        gameCount++;
        client.println("<OFF>");
    }
    else if(elapsedTime >= 50 && gameState == GAME_2){ // Spiel 2 spielen
        client.println("<PRESSED>");
    }
}

lastButtonState = buttonState;

```

Abb. 12 Code State Machine Client

## 5. Qualitätssicherung

Um eine zuverlässige Funktion und eine hohe Qualität des Buzzer-Systems sicherzustellen, wurden verschiedene Maßnahmen zur Qualitätssicherung ergriffen. Im Rahmen des Projekts ist die Qualitätssicherung ein wichtiger Bestandteil, um die Funktionalität und Sicherheit der Anwendung zu gewährleisten. Zunächst wird eine Versionierung in unterschiedlichen Branches durchgeführt. Nachdem der Code getestet und für gut befunden wurde, wird er in den Master-Branch gemerged. Dabei wird geprüft, ob die Anforderungen erfüllt wurden und die Software einwandfrei funktioniert. Ein weiterer wichtiger Aspekt ist der Integrationstest. Zunächst werden verschiedene Tests auf Hard- und Softwareebene durchgeführt, um die korrekte Funktionsweise des Systems zu gewährleisten. Hier wird das gesamte System getestet und nicht nur einzelne Funktionen. Dabei werden mögliche Probleme und Schwachstellen identifiziert und behoben. Durch diese Tests wird auch sichergestellt, dass die Anwendung den Anforderungen entspricht. Zusätzlich wurde während des Entwicklungsprozesses auf eine sorgfältige Dokumentation und eine klare Strukturierung des Codes geachtet. Dadurch soll die Wartbarkeit und Erweiterbarkeit des Systems erleichtert werden. Um eine hohe Benutzerfreundlichkeit zu erreichen, wurde das System auch von Testpersonen ausprobiert. Das Feedback der Testpersonen wurde ausgewertet und in die weitere Entwicklung des Systems einbezogen. Durch diese Maßnahmen zur Qualitätssicherung wird eine hohe Zuverlässigkeit und Qualität des Buzzer-Systems gewährleistet.

## 6. Fazit und Ausblick

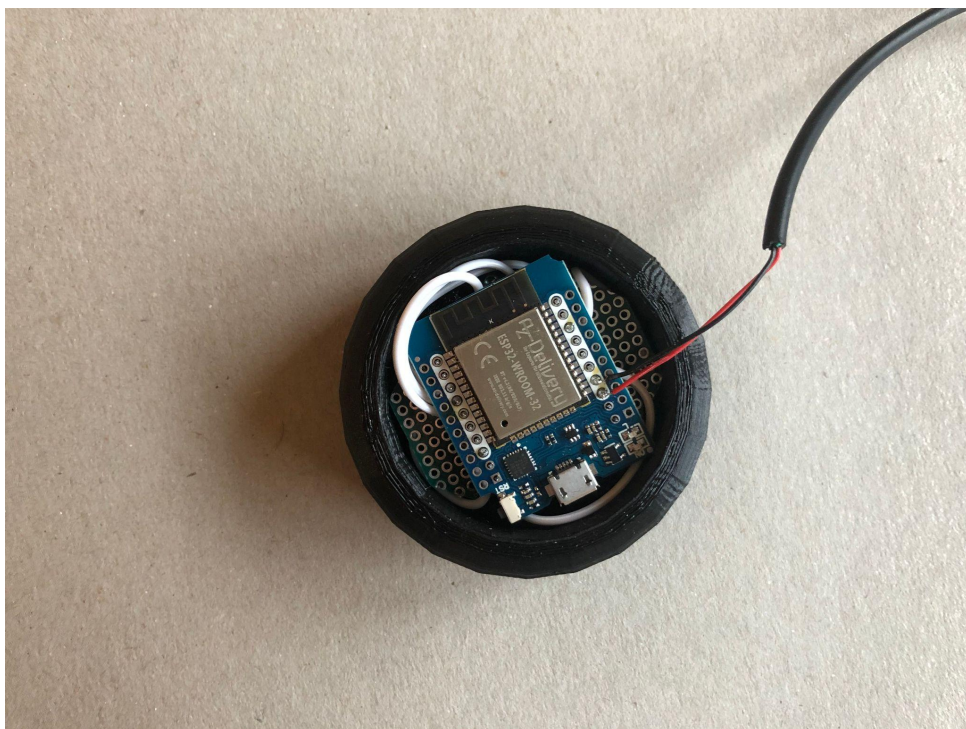
Das Projekt hat die Anforderungen erfüllt. Obwohl das Buzzer-System noch Verbesserungspotenzial aufweist, sind die Anwender begeistert von den Möglichkeiten, die ihnen die Anwendung bietet. Es konnte ein funktionstüchtiger Buzzer mit Gehäuse entwickelt werden, der alle notwendigen Funktionen unterstützt. Der Einsatz eines Mikrocontrollers mit WiFi-Peripherie ermöglicht eine einfache Verbindung zu anderen Teilnehmern und verbessert das Spielerlebnis.

Obwohl alle bisherigen Anforderungen erfüllt wurden, bietet die Modularität der Anwendung eine gute Grundlage für zukünftige Erweiterungen und Anpassungen. In Zukunft können weitere Erweiterungen und Anpassungen vorgenommen werden. Als nächste Schritte können Feedbackmöglichkeiten für den Nutzer hinzugefügt werden, beispielsweise durch eine Weboberfläche, über die der Nutzer den Buzzer steuern und Rückmeldungen bekommen kann. Eine weitere Möglichkeit zur Verbesserung der Nutzerfreundlichkeit ist der Einsatz eines Akkubetriebs mit einem 3,7 V Akku, um den Buzzer mobiler zu gestalten. Neue Spiele können auf ähnliche Weise implementiert werden, wodurch das Produkt deutlich vielfältiger einsetzbar wird. Aufgrund des im Kapitel 4 beschriebenen modularen Aufbaus des Projektes lassen sich solche Anpassungen beziehungsweise Erweiterungen sehr einfach vornehmen. Die Modularität der Anwendung ermöglicht somit eine gute Wartbarkeit und Erweiterbarkeit.



## 7. Anhänge

### A.1 Bilder



## Literatur- und Quellenverzeichnis

[demodern \_Games] Games & Gamification Marketing, Demodern, URL:  
<https://demodern.de/services/games> (Abgerufen: 18. September 2023, 18:00 UTC)

[Jen17] Jens Nickel: Mein Weg in das IoT (20): Ein eigenes WLAN-Netzwerk mit dem ESP32, Fassung vom 15. November 2017, URL:  
<https://www.elektormagazine.de/news/mein-weg-in-das-iot-20-ein-eigenes-wlan-netzwerk-mit-dem-esp32> (Abgerufen: 05. September 2023, 08:00 UTC)

[EXP18] EXP-Tech: Arduino Tutorial: Integrated Development Environment (IDE), Fassung vom 15. Januar 2018, URL:  
<https://www.exp-tech.de/blog/arduino-tutorial-integrated-development-environment-ide>  
(Abgerufen: 05. September 2023, 10:00 UTC)

[Mar17] Margaret Rouse: Keepalive, Fassung vom 30. März 2017, URL:  
<https://www.techopedia.com/definition/28979/keepalive> (Abgerufen: 14. Oktober 2023, 13:00 UTC)

## Abkürzungsverzeichnis

IDE	“Integrated Development Environment”
TCP/IP-Protokoll	“Transmission Control Protocol/Internet Protocol”
CAD	“Computer Aided Design”
PLA	“Polylactide”
LED	“light-emitting diode”
GPIO	“General Purpose Input/Output”

## Abbildungsverzeichnis

Abb. 1 Anwendungsfalldiagramm	6
Abb. 2 Verbindung herstellen	7
Abb. 3 Anzahl Verbindungen anzeigen	7
Abb. 4 Spiel starten	8
Abb. 5 Spiel 2 steuern	8
Abb. 6 Spielstatus abrufen	9
Abb. 7 Spiel beenden	9
Abb. 8 Schaltplan	11
Abb. 9 3D-Gehäuse	11
Abb. 10 LED Zustände	12
Abb. 11 Spiel 2 benutzerdefiniertes Protokoll	13
Abb. 12 Code State Machine Client	14

## Eidesstattliche Erklärung

Ich versichere, dass ich die Arbeit selbstständig verfasst und keine anderen als die angegebenen Quellen Hilfsmittel benutzt sowie Zitate kenntlich gemacht habe.

Die Regelung der geltenden Prüfungsordnung zu Versäumnis, Rücktritt, Täuschung und Ordnungsverstoß habe ich zur Kenntnis genommen.

Diese Arbeit hat in gleicher oder ähnlicher Form keiner Prüfungsbehörde vorgelegen.

Remscheid, den 16.10.2023



Jan Küpper