

**FHV**



Vorarlberg University  
of Applied Sciences

# Light-Pong Projektdokumentation

Matthias Hefel, Elias Sohm

Lehrveranstaltung: Embedded Programming

Betreut von: Prof.(FH) DI Patrick Ritschel

# Inhaltsverzeichnis

<b>Inhaltsverzeichnis</b>	<b>2</b>
<b>Idee</b>	<b>3</b>
<b>Team</b>	<b>3</b>
<b>Architektur</b>	<b>4</b>
Client/Tischtennisschläger	5
Softwarearchitektur	5
Server/Controller	6
Softwarearchitektur	6
Key-Features	7
Komponenten	7
Dmx_driver	7
Mh_x25_driver	7
Espnow_comm	7
Light_effects	7
Game	7
<b>Anhang</b>	<b>8</b>
Demo Video	8
VOL.AT Artikel zum ÖH-Hackathon	8

## Idee

Tischtennis mit Licht. Das ist die Idee von Light-Pong. Anstatt einem Physischen Tischtennisball, schlagen die Spieler mit ihrem Schläger in die Luft und bewegen einen Ball aus Licht, welcher auf das Spielfeld projiziert wird.



## Team



Elias Sohm (Links im Bild)

Matthias Hefel (Mittig im Bild)

Betreut von: Patrick Ritschel (Rechts im Bild)

# Architektur

Es werden 3 ESP32c3 und ein MH-X25 LED Spot verwendet. Ein ESP dient als "Server" und empfängt Accelerometer-Daten, welche die zwei "Clients" (die anderen ESPs), via ESP-NOW an diesen senden. Der Server nutzt diese für die Game Loop.

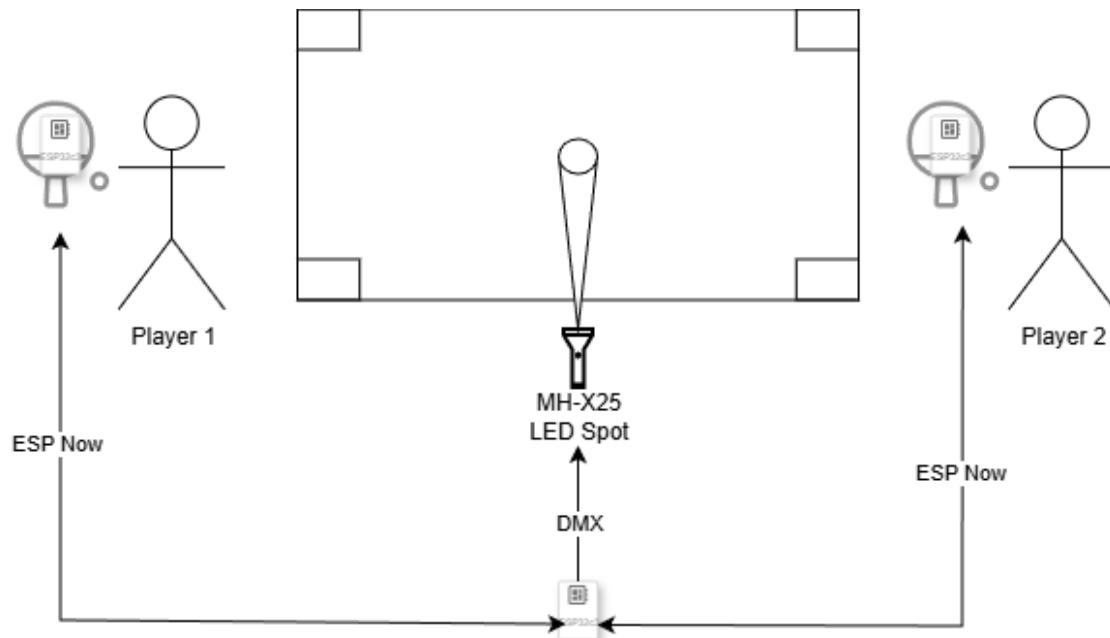


Abbildung 1: Light-Pong Architektur

In Abbildung 1 zu sehen ist die physische Architektur des Systems. Jeder Tischtennisschläger ist mit einem ESP32c3 ausgestattet. Der Tisch besteht in der Realität dabei aus Papier auf Tischhöhe und der MH-X25 LED Spot steht in der Mitte unter dem Tisch.

## Client

Die auf den Schlägern angebrachten ESPs schicken, wenn ein gewisser Beschleunigungs-Threshold überschritten wird, ein Signal an den Server. Um dieses Signal zu senden, wird ESP-NOW ein auf MAC-Adressen basierendes verbindungsloses Protokoll verwendet, das Datenpakete über das 2,4-GHz-Band sendet. Das Signal enthält Informationen über Sender, die momentane Beschleunigung des Schlägers zum Sendezeitpunkt sowie ob der Knopf für den Spezialschuss (Feuerball) beim Schlag betätigt wurde. Für den Feuerball wurde ein 10 Sekunden Cooldown implementiert, somit kann dieser nicht konstant eingesetzt werden.

Die ESPs verwenden die LED Matrix, um die Punktzahl des jeweiligen Spielers anzuzeigen (als Zahl). Sowie ein Grün -> Rot Verlauf, welcher den Feuerball Cooldown anzeigt.

## Client - Softwarearchitektur

Die Client-Software ist modular aufgebaut und in mehrere funktionale Komponenten unterteilt. Eine Sensorkomponente verarbeitet die Daten des ICM-42688-P sowie die Eingaben der Buttons. Eine separate Kommunikationsschicht übernimmt die drahtlose Übertragung der Ereignisse über ESP-NOW. Zusätzlich ist eine Feedback-Komponente für die Ansteuerung der LED-Matrix zuständig, um Spielinformationen in Echtzeit darzustellen.

Diese Struktur ermöglicht eine klare Trennung zwischen Sensorverarbeitung, Spiel-Logik, Kommunikation und visueller Rückmeldung.

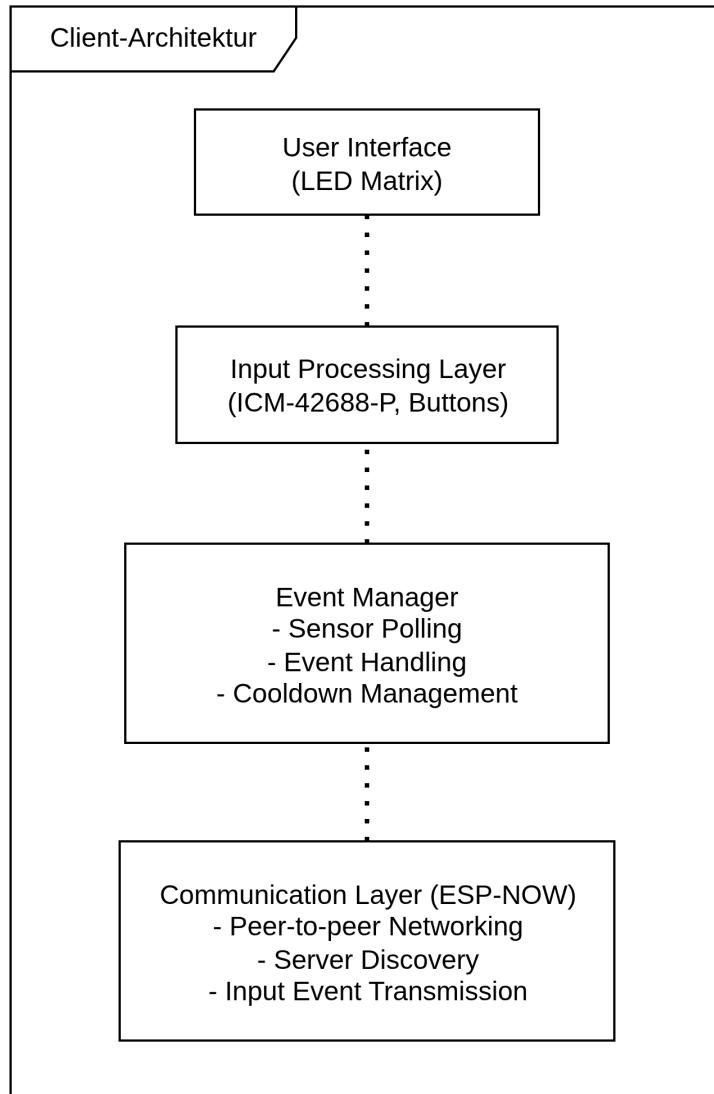


Abbildung 2: Light-Pong Client Architektur

## Server/Controller

Der Server bedient mehrere Funktionalitäten. Er dient zum einen zum Senden und Empfangen von Nachrichten der zwei ESP32c3, welche an den Tischtennisschlägern angebracht sind. Diese senden Nachrichten wie "Schlage zu" und "Feuerball JA/NEIN". Welche dann vom Server bearbeitet werden. Der Server entscheidet dann anhand des Gamestates (welcher Spieler ist am Zug; ist der Lichtball schon nahe genug am Rand des Spielfeldes) darüber, ob der Schlag durchgeht und sich der Lichtball in die andere Richtung bewegt, oder nichts passiert. Dazu werden Informationen, wie der Punktestand an die Schläger gesendet, welche dann auf den Schlägern angezeigt werden.

## Softwarearchitektur

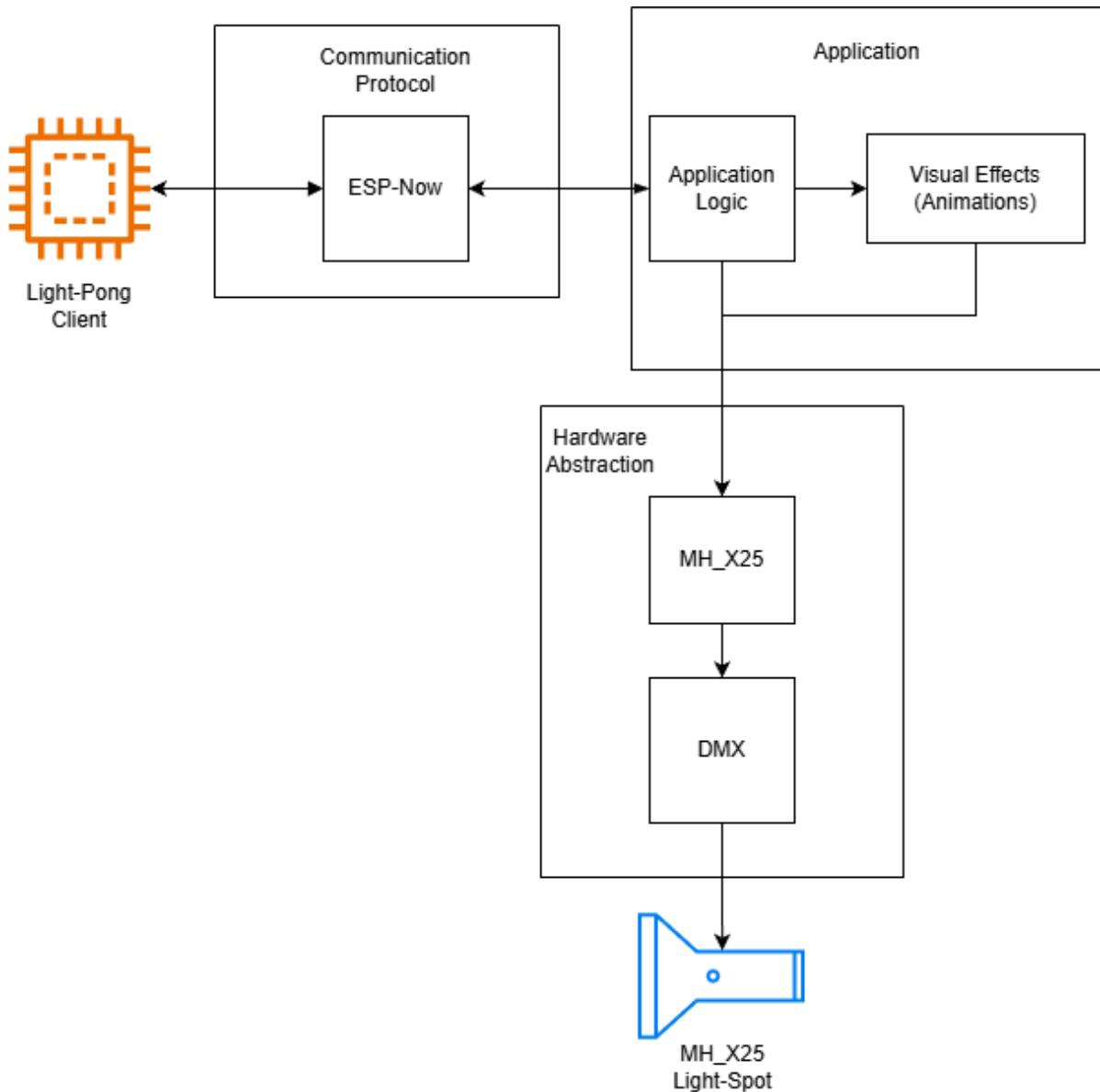


Abbildung 2: Light-Pong Server Architektur

In Abbildung 2 ist die Softwarearchitektur des Light-Pong-Servers zu sehen. Sie besteht im Großen und Ganzen aus drei Bereichen. Der Bereich für Communication, welcher die Kommunikation zu den Clients via ESP-NOW implementiert. Der Application Bereich, welcher die Spiellogik und Steuerung des MH-X25 moving heads steuert. Sowie dem Hardware Abstraction Bereichs, welcher DMX512 implementiert und eine einfach zu verwendende Abstraktion für den MH-X25 moving head bereitstellt, welche von der Game und Lighting\_Effects Komponente genutzt wird, um den moving head anzusteuern.

## Key-Features

**Layered Component Architecture:** Strikte Trennung zwischen Hardware-Abstraktion (DMX, MH-X25), Protokoll-Layer (ESP-NOW), Visual Effects und Game Logic.

**Dependency Injection Pattern:** Komponenten erhalten Handles über Context-Setter, keine direkte Abhängigkeit zwischen Layern. Ermöglicht einfaches Testing und Austausch von Komponenten.

**FreeRTOS Multi-Task Architecture:** Separate Tasks für ESP-NOW Receiver (Priorität 5) und Game Controller Loop, synchronisiert über Event Groups für Paddle-Hit-Events.

## Komponenten

### Dmx\_driver

Implementation des DMX512 Protokolls über RS-485

### Mh\_x25\_driver

Abstraktionslayer für MH-X25 LED moving head. Betrieben im 12-channel DMX mode. Ermöglicht das Ansteuern von Pan, Tilt, Gobo, Color, ...

### Espnow\_comm

Kommunikation mit Clients über ESP-Now.

- Dynamische Peer Discovery und Registrierung (Keine statischen MAC-Adressen benötigt)
- Player-ID Assignment
- Receiving Client "Hit-event"
- Broadcast Player-Scores

### Light\_effects

Komponente für Lichteffekte für Win-Animation.

### Game

Zentrale Spiellogik.

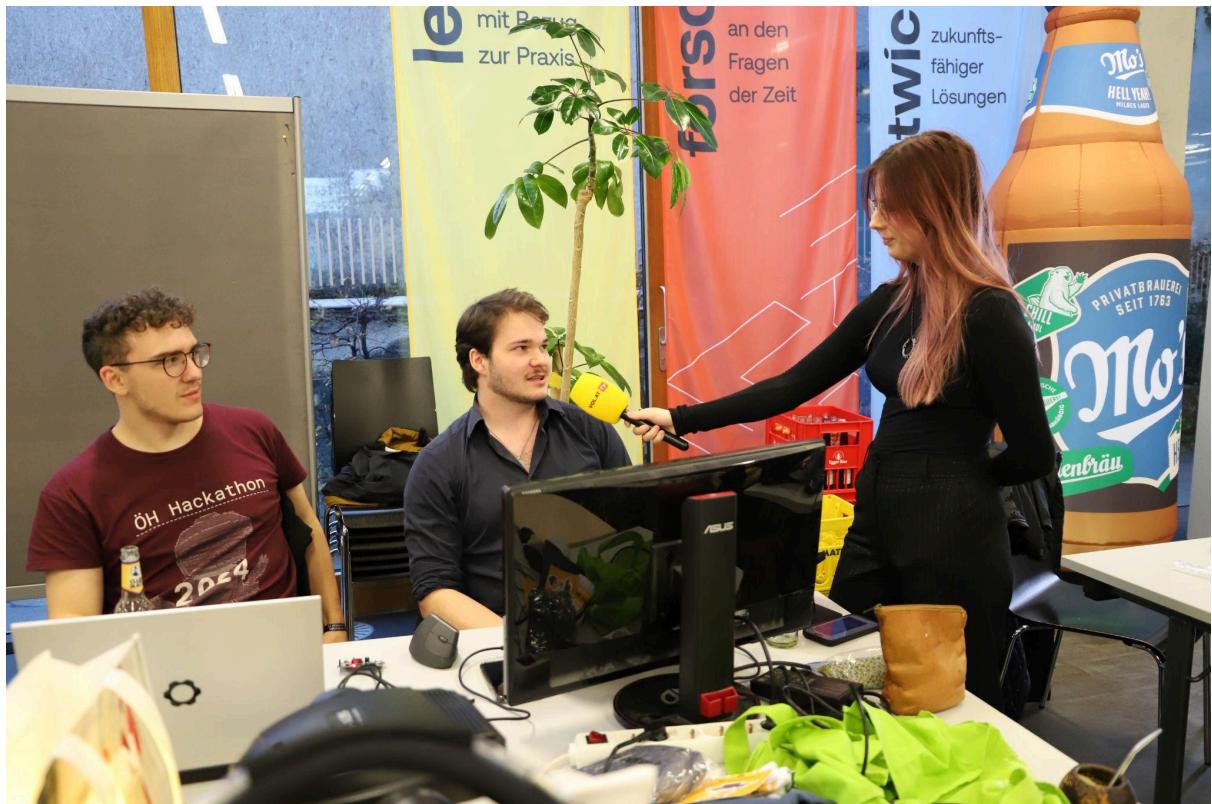
- Client-Hit-Detection
- Client-Hit-Timeout
- Score management und wind condition checking
- Fireball mode handling

# Anhang

Demo Video

[Light-Pong-Demo.MOV](#)

[VOL.AT Artikel zum ÖH-Hackathon](#)



<https://www.vol.at/hackathon-100-studierende-48-stunden-eine-menge-kreativitaet/9806525>