



**Hochschule für Technik
und Wirtschaft Berlin**

University of Applied Sciences

Projektdokumentation

„PoC Grüne-Welle-Assistent / Urban Compass“

Modul:

Big Data Analytics

Semester:

Sommersemester 2022

Dozent:

Prof. Dr. Olga Willner

Projektteam:

Konstantin Groll

Lucas Oldenburg

Per Ejsmont

Nico Teichert

Paul Felix Kühn

1	Projektziel.....	3
1.1	PoC Grüne-Welle-Assistent ESP32	3
1.2	Urban Compass	3
2	Hardware.....	4
3	Fachkonzeption	5
3.1	PoC Grüne-Welle-Assistent ESP32	5
3.2	Urban Compass	6
3.2.1	LED-Inhalte	6
3.2.2	Konzeptskizzen	8
3.2.2.1	Bezirke	9
3.2.2.2	Sehenswürdigkeiten	10
3.2.2.3	ÖPNV	11
3.2.2.4	Uhrzeit	12
3.2.2.5	Logos	13
3.2.3	Storyboard	14
4	Projektvorgehen.....	15
4.1	Hardwareentwicklung	15
4.1.1	Elektronik-Bauteile	15
4.1.2	Modellierung und 3D-Druck Gehäuse	24
4.2	Softwareentwicklung	25
4.2.1	Grüne Welle Assistent	25
4.2.1.1	Anzeige Balken	25
4.2.1.2	Anzeige Fahrrad-Symbol	26
4.2.2	Urban Compass	27
5	Projektergebnis	30
6	Ausblick	32

Inhaltsverzeichnis

1 Projektziel

Die Ziele dieses Projekts lassen sich in zwei Unterziele unterteilen, die im Folgenden beschrieben werden.

1.1 PoC Grüne-Welle-Assistent ESP32

Im Rahmen des „Leetzenflow“-Projekts entwickelte das Institut für Verkehrswissenschaften der WWU Münster im Auftrag der Stadt Münster einen prototypischen Grüne-Welle-Assistenten. Dieser unterstützt Fahrradfahrende dabei, die eigene Geschwindigkeit anzupassen, um grüne Ampelphasen optimal zu nutzen und somit an weniger Ampeln anhalten zu müssen. Ziel dieses Projekts ist die Erarbeitung eines Proof of Concept (PoC), das belegt, dass der Grüne-Welle-Assistent ebenso mit abweichender Hardware unter Verwendung eines ESP32 als Mikrocontroller umgesetzt werden kann.

1.2 Urban Compass

Das zweite Projektziel besteht darin, die im Rahmen des PoC entwickelte Hardwarelösung als Grundlage zu verwenden, um eine andere Anzeige als den Ampelstatus auf den LEDs als „Urban Compass“ zu realisieren. Konkret handelt es sich dabei um Infotainment-Inhalte, die auf den LEDs angezeigt werden können. Diese Inhalte sollen dabei das Potenzial haben, Fahrradfahrende bei der Navigation durch die Stadt zu unterstützen und darüber hinaus eine positive psychologische Wirkung auf das Fahrerlebnis zu erzeugen. Die Inhalte werden im Rahmen des Projekts in einem fachlichen Konzept entwickelt und prototypisch als Anzeige auf den LEDs umgesetzt.

2 Hardware

Die folgende Tabelle zeigt alle im Projekt verwendeten Hardwarekomponenten inklusive Shopping-Links und Kostenaufstellung.

Komponente	Produktnname	Hersteller	Link	Preis
LED 1	64x32 RGB LED MATRIX – 5MM Pitch	Adafruit Industries LLC	https://www.digikey.de/de/products/detail/adafruit-industries-llc/2277/7035035	59,99 €
LED 2	32x32 RGB LED MATRIX – 5MM Pitch	Adafruit Industries LLC	https://www.digikey.de/de/products/detail/adafruit-industries-llc/2026/7035028?s=N4IgTCBcDallwFYwA4COY6NQOQCIGFOBfIA	41,98 €
Microcontroller	ESP32WROOM32E	Espressif Systems	https://www.digikey.de/de/products/detail/espressif-systems/ESP32-DEVKITC-32E/12091810	12,00 €
Jumper-Kabel	Jumper Wire Kabel 40 STK. je 20 cm F2F Female to Female	AZ-Delivery	https://www.kaufland.de/product/342455919/?utm_source=shopping&utm_medium=non-paid&utm_campaign=pricecomparison&id=42345840	3,99 € (40 Stück)
Netzteil	AC-Adapter (Output: 5V, 3A)	Leicke	https://www.otto.de/p/leicke-ull-netzteil-15w-5v-3a-netzteil-besonders-leicht-kurzschluss-ueberspannungs-und-ueberhitzungsschutz-SOC1G0DZ/#variationId=S0C1G0DZBSMT	8,99 €
Netzteil-Adapter	Sara fragen	-	-	-
Micro-USB-Kabel	Micro-USB-Kabel	Liour	https://amzn.to/3qPDoVF	4,99 €
Stromkabel LED	HUB75-Power Cable	Adafruit Industries LLC	Im Lieferumfang der LED-Matrix enthalten	-
HUB75-Verbindungs kabel	HUB75-Data Cable	Adafruit Industries LLC	Im Lieferumfang der LED-Matrix enthalten	-
DC-Buchse	DC Buchse Stecker 5,5 x 2,1mm	LitaElek	https://www.amazon.de/dp/B019HAC6V4/	6,49 € (5 Paar)
-	-	-	-	Summe: ???

Zusätzlich wurde im Rahmen des Projekts folgendes Werkzeug verwendet:

- 3D-Drucker für die Case (Anycubic Mega X Drucker) (https://de.anycubic.com/products/mega-x?_pos=14&_sid=d5e44558a&_ss=r)
- Messer (Case)
- Pfeile (Case)
- Bohrer (Case)
- Seitenschneider für das zuschneiden der Kabel
- Lötkolben
- Schrumpfschlauch

Die genaue Beschreibung der Zusammensetzung der Komponenten findet sich in Kapitel 4.1 „Elektronik-Bauteile“.

3 Fachkonzeption

Im Folgenden wird jeweils die vorangehende Planung und Konzeption zur Erreichung der in Kapitel 1 beschriebenen Projektziele erläutert.

3.1 PoC Grüne-Welle-Assistent ESP32

Der Grüne-Welle-Assistent stellt, wie bereits in Kapitel 1.1 beschrieben, ein Produkt dar, dass aus dem „Leetzenflow“-Projekt der Stadt Münster heraus entstanden ist. Er soll Fahrradfahrende, die sich einer Kreuzung mit Ampelanlage nähern, mit Informationen bzgl. der aktuellen Ampelphase versorgen. Ziel dessen ist, dass die Fahrradfahrenden ihre Geschwindigkeit auf Basis dieser Information so regulieren können, dass sie an der aufkommenden Ampel nicht anhalten sowie absteigen müssen und so im „Flow“ durch die Stadt kommen.

Realisiert wurde der Grüne Welle Assistent durch eine digitale Anzeige (zwei verbundene LEDs) vor einer aufkommenden Straßenkreuzung, die anhand eines Farbverlaufs die verbleibende Dauer der aktuellen Ampelphase darstellt. Dabei wird bei einer grünen Ampelphase ein grüner Balken angezeigt, der mit abnehmender Grün-Phase immer kleiner wird. Wechselt die Ampel auf „Rot“, wechselt analog dazu auch die LED-Anzeige auf einen roten Balken, der mit abnehmender Rot-Phase immer kleiner wird. Zusätzlich wird während einer Rot-Phase ein statisches und bei einer Grün-Phase ein sich bewegendes Fahrrad-Symbol angezeigt. Dies symbolisiert den Fahrradfahrenden, dass sie sich von der Anzeige angesprochen fühlen sollen und verdeutlicht zudem, ob Fahrradfahrende an der aufkommenden Ampel aktuell anhalten müssen oder fahren können.



Quelle: <https://smartcity.ms/leetzenflow/>

Die vollständige fachliche und technische Beschreibung dieses Vorgängerprojekts kann unter <https://smartcity.ms/leezenflow/> eingesehen werden. Bei dem Projekt handelt es sich zudem über ein Open-Source-Projekt, bei dem alle Vorgehensweisen und Komponenten der Implementierung frei öffentlich verfügbar gemacht wurden. Diese können unter <https://github.com/bCyberGmbH/leezenflow-doku> abgerufen werden. Die drei relevantesten verwendeten Komponenten zur Anzeige des Ampelstatus umfassen dabei:

Komponente	Produktnname	Hersteller	Link
LED 1	64x32 RGB LED MATRIX – 5MM Pitch	Adafruit Industries LLC	https://www.digikey.de/de/products/detail/adafruit-industries-llc/2277/7035035
LED 2	32x32 RGB LED MATRIX – 5MM Pitch	Adafruit Industries LLC	https://www.digikey.de/de/products/detail/adafruit-industries-llc/2026/7035028?s=N4IgTCBcDallwFYwA4C0Y6NQOQCIGFOBfIA
Raspberry Pi	Raspberry 1373331 Pi 3 Modell B+ Mainboard, 1GB	Raspberry	https://www.amazon.de/Raspberry-1373331-Modell-Mainboard-1GB/dp/B07BFH96M3/ref=as_li_ss_til?dchild=1&keywords=raspberrypi+pi+3&qid=1625566866&sr=8-4&th=1&linkCode=sl1&tag=bcyber-21&linkId=62525177043a4efc0c5861fe447f3e6c&language=de_DE

Ziel dieses Projekts ist die ist die technische Nachstellung des Projekts bzw. der Anzeige des Ampelstatus. Dabei wird anstatt der oben aufgeführten Raspberry Pi Komponente jedoch ein ESP32 als Microcontroller eingesetzt, wobei die gleichen LED-Komponenten verwendet werden (siehe auch Kapitel 2 „Hardware“). Das erste Projektziel beinhaltet also die Entwicklung eines Proof of Concept (PoC), das zeigen soll, dass die Anzeige des Ampelstatus auch unter Verwendung abweichender Hardware (ESP32) möglich ist.

Der ESP32 ist im Gegensatz zum Raspberry Pi deutlich günstiger und verfügt zum Zeitpunkt des Projekts über eine höhere Verfügbarkeit, was die Implementierung des Grüne Welle Assistenten skalierbarer machen würde. Im Vergleich zum Raspberry Pi verfügt der ESP32 jedoch über eine geringere Performanz, weshalb innerhalb des PoC geprüft werden soll, ob eine Umsetzung unter diesem Nachteil trotzdem möglich ist. Eine Verbindung zur Ampelanlage, wie sie im Leetzenflow-Projekt umgesetzt wurde, ist dabei noch nicht vorgesehen.

Für die Umsetzung des PoC innerhalb dieses Projekts wird dabei folgendes Abnahmeszenario definiert:

1. Genereller Nachweis der Betriebsfähigkeit der verfügbaren Hardware-Komponenten (siehe Kapitel 4.1.1).
2. Konstante Anzeige eines statischen Fahrrad-Symbols auf der LED-Anzeige.
3. Anzeige eines grünfarbigen Balkens auf der LED-Anzeige, der mit der Zeit in konstanter Geschwindigkeit abnimmt bzw. sich verkleinert und schließlich vollends verschwindet.
4. Automatischer Wechsel der Anzeige auf einen rotfarbigen Balken, sobald der grünfarbige Balken komplett ausgeblendet ist.
5. Anzeige des rotfarbigen Balkens auf der LED-Anzeige, der ebenfalls mit der Zeit in konstanter Geschwindigkeit abnimmt bzw. sich verkleinert und schließlich vollends verschwindet.
6. Erneuter automatischer Wechsel der Anzeige auf einen grünfarbigen Balken, sobald der rotfarbige Balken komplett ausgeblendet ist.

Gelingt es zum Ende des Projekts, das oberhalb beschriebene bzw. geplante Szenario unter Einsatz der in Kapitel 2 aufgeführten Hardware-Komponenten zu realisieren, gilt das PoC als erfolgreich umgesetzt.

3.2 Urban Compass

In diesem Kapitel werden die auf dem Urban Compass anzuzeigenden Inhalte, deren Gestaltung sowie die angestrebte Nutzung des Urban Compass aus Sicht eines Fahrradfahrenden beschrieben.

3.2.1 LED-Inhalte

Hauptziel der Anzeige von Infotainment-Inhalten ist in erster Linie, das Fahrerlebnis von Fahrradfahrenden positiv zu beeinflussen, indem auf den LEDs verschiedene Informationen angezeigt werden. Einerseits soll dabei die Navigation durch die Stadt erleichtert werden mit dem Ziel, dass Fahrradfahrende nicht anhalten bzw. nicht vom Fahrrad absteigen müssen, um sich über den weiteren Weg zu informieren. Dies ist insbesondere in Verbindung mit dem

Grüne-Welle-Assistenten sinnvoll, der ebenfalls das Ziel verfolgt, Fahrradfahrenden einen guten „Flow“ im Verkehrsfluss zu ermöglichen. Zum anderen sollen die angezeigten Informationen dazu anregen, die Stadt und ihre Sehenswürdigkeiten besser kennen zu lernen und Menschen zur Entschleunigung des Alltags zu bewegen.

Für die Identifikation der anzuzeigenden Informationen wurde ein Brainstorming-Teamworkshop durchgeführt, in dem acht verschiedene Kategorien von Informationen bestimmt wurden, die für die Anzeige auf der LED sinnvoll und geeignet wären:

#	Kategorie	Beschreibung / Inhalt	Nutzen
1	Bezirke	Stadtbezirke wie z.B. „Friedrichshain“, „Kreuzberg“, etc.	Ermöglichung grober Orientierung für Fahrradfahrende, damit diese angenehmer durch die Stadt navigieren können und dabei weniger auf Navigationsdienste / Karten zurückgreifen müssen.
2	ÖPNV	Zentrale Linien des öffentlichen Personennahverkehrs wie z.B. „U5“, „S41/S42“, etc.	Ermöglichung grober Orientierung analog zu „1“, zusätzlich Förderung des multimodalen Verkehrsverhaltens von Fahrradfahrenden. Außerdem nützlich für Fahrradfahrende, um bei plötzlich auftretenden schlechten Wetterverhältnissen auf die öffentlichen Verkehrsmittel zu wechseln.
3	Kieze	Bekannte Stadtkieze wie z.B. „Bergmannkiez“, „Wrangelkiez“, etc.	Ermöglichung grober Orientierung analog zu „1“, zusätzlich Ermöglichung der Erkundung des Stadtbildes für Fahrradfahrende.
4	Plätze	Bekannte Stadtplätze wie z.B. „Hermannplatz“, „Moritzplatz“, etc.	Ermöglichung grober Orientierung analog zu „1“, zusätzlich Ermöglichung der Erkundung des Stadtbildes für Fahrradfahrende.
5	Sehenswürdigkeiten	Bekannte Sehenswürdigkeiten wie Gebäude, Denkmale, Brücken, Museen, etc. wie z.B. „Checkpoint Charlie“, „East Side Gallery“, „Jüdisches Museum“ etc.	Ermöglichung grober Orientierung analog zu „1“, zusätzlich Ermöglichung der Erkundung des Stadtbildes für Fahrradfahrende.
6	Freizeitmöglichkeiten	Diverse Arten von öffentlichen Freizeiteinrichtungen wie z.B. Schwimmbäder, Kinos, Theater, etc.	Ermöglichung der Erkundung des Stadtbildes für Fahrradfahrende.
7	Parks & Seen	Parkanlagen, z.B. „Viktoriapark“, „Görlitzer Park“, „Plötzensee“, etc.	Ermöglichung grober Orientierung analog zu „1“, zusätzlich Anregung von Fahrradfahrenden zur vermehrten Nutzung von Grünflächen für die Förderung psychischer Gesundheit.
8	Uhrzeit	Anzeige der aktuellen Uhrzeit.	Ermöglichung zeitlicher Orientierung von Fahrradfahrenden, um Navigation ggf. anzupassen oder neue Ziele bzw. Zwischenziele einzuplanen.

Um die Relevanz der einzelnen Informationskategorien für Fahrradfahrende im Einzelnen zu bewerten und eine finale Auswahl zu treffen, wird an dieser Stelle eine weitere Befragung von Radfahrern im Rahmen eines zukünftigen Nachfolgeprojekts empfohlen. Aus zeitlichen Gründen wird dies in diesem Projekt nicht durchgeführt und stattdessen die drei Kategorien „Bezirke“, „Sehenswürdigkeiten“, „ÖPNV“ und „Uhrzeit“ exemplarisch ausgewählt.

Zusätzlich zu der Anzeige der Elemente (z.B. „Friedrichshain“, „Fernsehturm“, etc.) soll hinter diesen eine Anzeige der Richtung erfolgen, die ein Fahrradfahrender ab der nächsten Kreuzung einschlagen soll, um sich in die Richtung des jeweiligen Elements zu begeben. Für die Richtungsanzeige werden dabei fünf Pfeilarten definiert:



Die jeweiligen Kategorien inkl. der Richtungspfeile sollen dabei immer im Wechselseitigen Dauerschleife angezeigt werden. Dies bedeutet, dass nach der Anzeige einer Kategorie für einige Sekunden die nächste Kategorie auf der LED angezeigt werden soll und dieser Vorgang kontinuierlich wiederholt wird. Die genaue Zeitdauer für die Anzeige einer Kategorie wird dabei in diesem Projekt auf fünf Sekunden festgelegt, da dies für Test- & Demonstrationszwecke einen guten Wert darstellt. Für eine spätere Umsetzung sollte nach der Bestimmung der finalen Kategorien eine Anzeigedauer gewählt werden, bei der mit einer durchschnittlichen Fahrgeschwindigkeit von Fahrradfahrenden alle Kategorien während der Fahrt erkannt und wahrgenommen werden können.

Als weitere LED-Inhalte kam im Projektverlauf zudem die Idee auf, passende Icons zu den verschiedenen Kategorien zu definieren und auf der LED-Matrix anzuzeigen. Dies könnte für Fahrradfahrende den Mehrwert schaffen, die angezeigten Informationen schneller und besser einordnen zu können. Aus zeitlichen Gründen wird darauf im Rahmen dieses Projekts bei der Umsetzung verzichtet.

3.2.2 Konzeptskizzen

Als Grundlage für die Umsetzung des Urban Compass wurden Konzeptskizzen entwickelt, die sich hinsichtlich der Gestaltung an denen des Reallabor Radbahn Berlin orientieren. Die Konzeptskizzen wurden mit PowerPoint und dem Online-Tool Pixilart (<https://www.pixilart.com>) erstellt. Für die Erstellung der Bilder der in Kapitel 3.2.1 konzeptionierten LED-Inhalte wurde Pixilart ausgewählt, weil es sehr einfach und schnell ermöglicht, ein Bild mit der gleichen Auflösung wie die der LED-Matrix zu erstellen. Diese können anschließend in verschiedenen Bildgrößen exportiert werden. Anschließend wurden die erstellten Bilder der LED-Inhalte mit PowerPoint in die Fotos von der Radbahn eingesetzt, um die Konzeptskizzen zu erstellen, die in den folgenden Unterkapiteln dargestellt werden. Darüber hinaus wurden erste Logos entworfen, die zur Bewerbung und Gestaltung des finalen Produkts eingesetzt werden können.

3.2.2.1 Bezirke



3.2.2.2 Sehenswürdigkeiten



3.2.2.3 ÖPNV



3.2.2.4 Uhrzeit



3.2.2.5 Logos

Logo Version 0:



Logo Version 1:

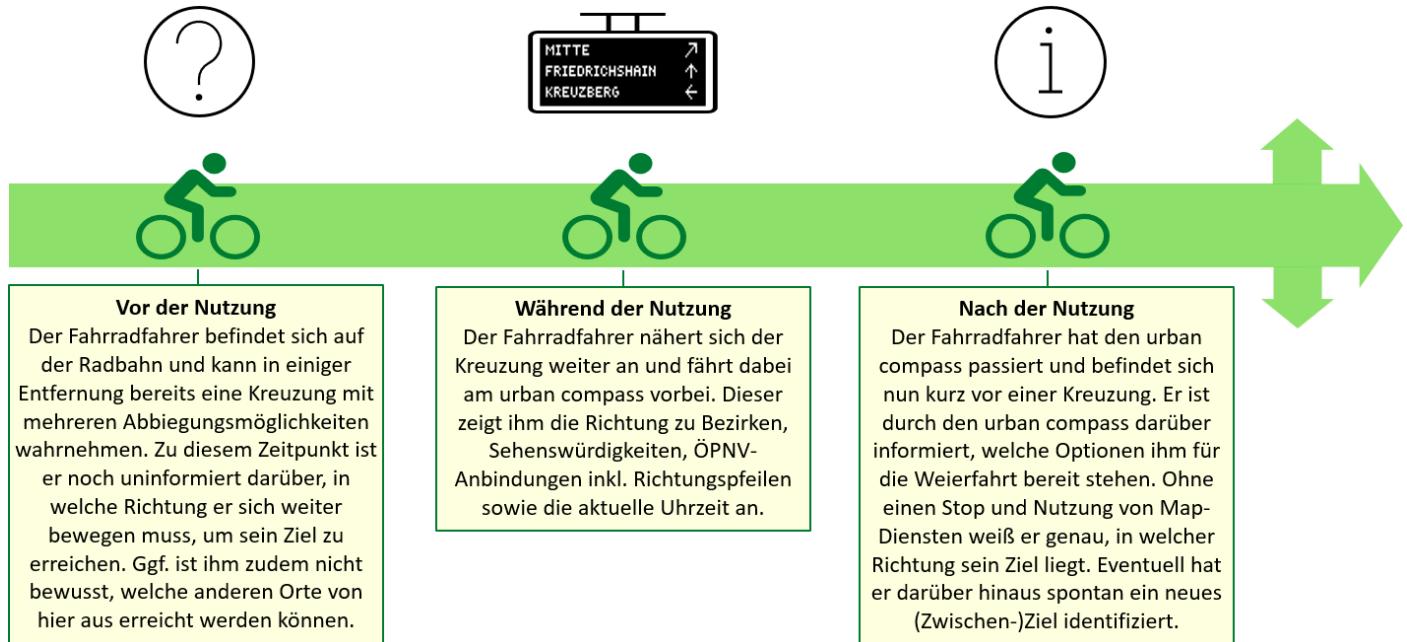


Logo Version 2 (final):



3.2.3 Storyboard

Das unterhalb aufgeföhrte Storyboard beschreibt die Idealsituation, in der Fahrradfahrende als zukünftige Nutzer mit dem urban compass interagieren. Die beschriebenen Verhaltensweisen und Entwicklungen dieser sind somit auch als Zielstellung zu verstehen, die als Grundlage für die weitere Entwicklung gelten.



4 Projektvorgehen

In diesem Kapitel werden die einzelnen Vorgehensschritte des Projekts unterteilt nach Hardware- und Softwareentwicklung beschrieben und dokumentiert.

4.1 Hardwareentwicklung

Dieses Kapitel umfasst eine Beschreibung über den (Zusammen-)Bau der eingesetzten Hardwarekomponenten.

4.1.1 Elektronik-Bauteile

Zu Beginn wurde versucht, die Zusammensetzung und Funktionsfähigkeit der verfügbaren Hardware-Komponenten mittels verschiedener Simulatoren zu testen. Dadurch sollte Aufwand bei der Inbetriebnahme der Komponenten durch aufwändiges Testen vermieden werden. Dafür wurde beispielsweise das Tool Wokwi (<https://wokwi.com/>) verwendet, mit dem IoT-Projekte simuliert werden können.

The screenshot shows the Wokwi web-based development environment. At the top, there's a toolbar with 'SAVE', 'SHARE', and a library search bar. Below that is a navigation bar with 'hub75-poc.ino' and 'by urish'. The main area has tabs for 'hub75-poc.ino', 'diagram.json', 'libraries.txt', and 'Library Manager'. The code editor contains the following C++ code:

```
1 // testshapes demo for RGBmatrixPanel library.
2 // Demonstrates the drawing abilities of the RGBmatrixPanel library.
3 // For 32x64 RGB LED matrix.
4
5 // WILL NOT FIT on ARDUINO UNO -- requires a Mega, M0 or M4 board
6
7 #include <RGBmatrixPanel.h>
8
9 // Most of the signal pins are configurable, but the CLK pin has some
10 // special constraints. On 8-bit AVR boards it must be on PORTB...
11 // Pin 11 works on the Arduino Mega. On 32-bit SAMD boards it must be
12 // on the same PORT as the RGB data pins (D2-D7)...
13 // Pin 8 works on the Adafruit Metro M0 or Arduino Zero,
14 // Pin A4 works on the Adafruit Metro M4 (if using the Adafruit RGB
15 // Matrix Shield, cut trace between CLK pads and run a wire to A4).
16
17 // #define CLK 8 // USE THIS ON ADAFRUIT METRO M0, etc.
18 // #define CLK A4 // USE THIS ON METRO M4 (not M0)
19 #define CLK 11 // USE THIS ON ARDUINO MEGA
20
21 #define OE 9
22 #define LAT 10
23 #define A A0
24 #define B A1
25 #define C A2
26 #define D A3
27
28 RGBmatrixPanel matrix(A, B, C, D, CLK, LAT, OE, false, 64);
29
30 void setup() {
31
32     matrix.begin();
33
34     // draw a pixel in solid white
35     matrix.drawPixel(0, 0, matrix.Color333(7, 7, 7));
36 }
```

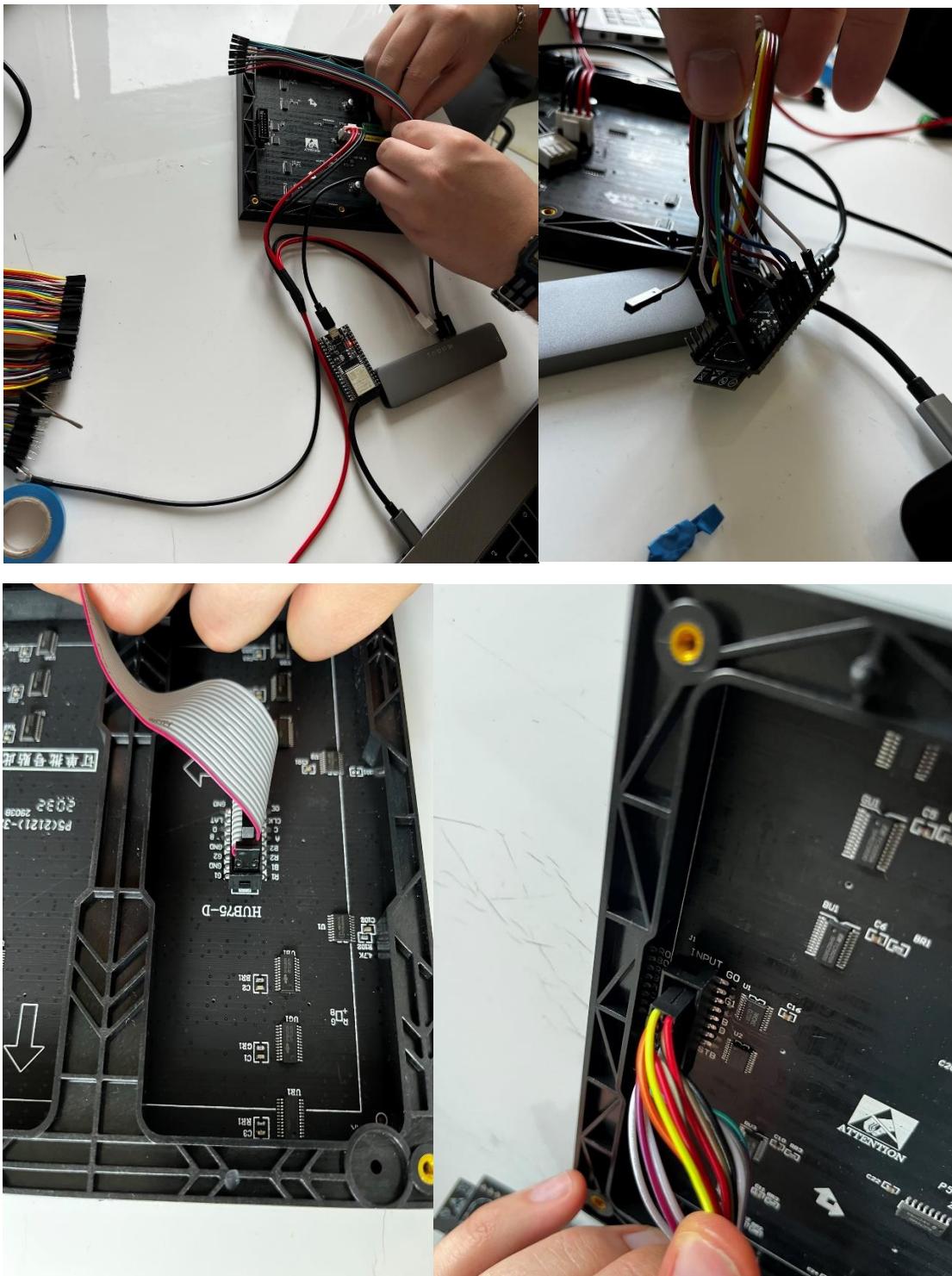
To the right of the code editor is a simulation window titled 'Simulation'. It shows a 32x64 grid of pixels with a red 'X' drawn across it. Below the grid is a 3D model of an Arduino MEGA board. The status bar at the bottom right indicates '00:01.833' and '16%'. There are also play and pause buttons for the simulation.

Quelle: <https://wokwi.com/projects/308108890422116928>

Leider standen bei allen Simulatoren einzelne in diesem Projekt verfügbare Komponenten nicht zur Verfügung. So ist beispielsweise bei Wokwi das Matrix LED-Panel nicht verfügbar. Es gibt lediglich eine sehr ähnliches Projekt (<https://wokwi.com/projects/308108890422116928>), bei dem ein Arduino MEGA Mikrocontroller in Verbindung mit einem Matrix LED-Panel verwendet wird. Leider spiegelt dies jedoch nicht die in diesem Projekt verfügbaren Komponenten wider, weshalb kein Simulator verwendet werden konnte. Aufgrund dessen wurden die Komponenten nach dem „try and error“-Prinzip Schritt für Schritt verbunden und getestet. Die einzelnen Schritte inkl. Probleme werden im Folgenden aufgeführt.

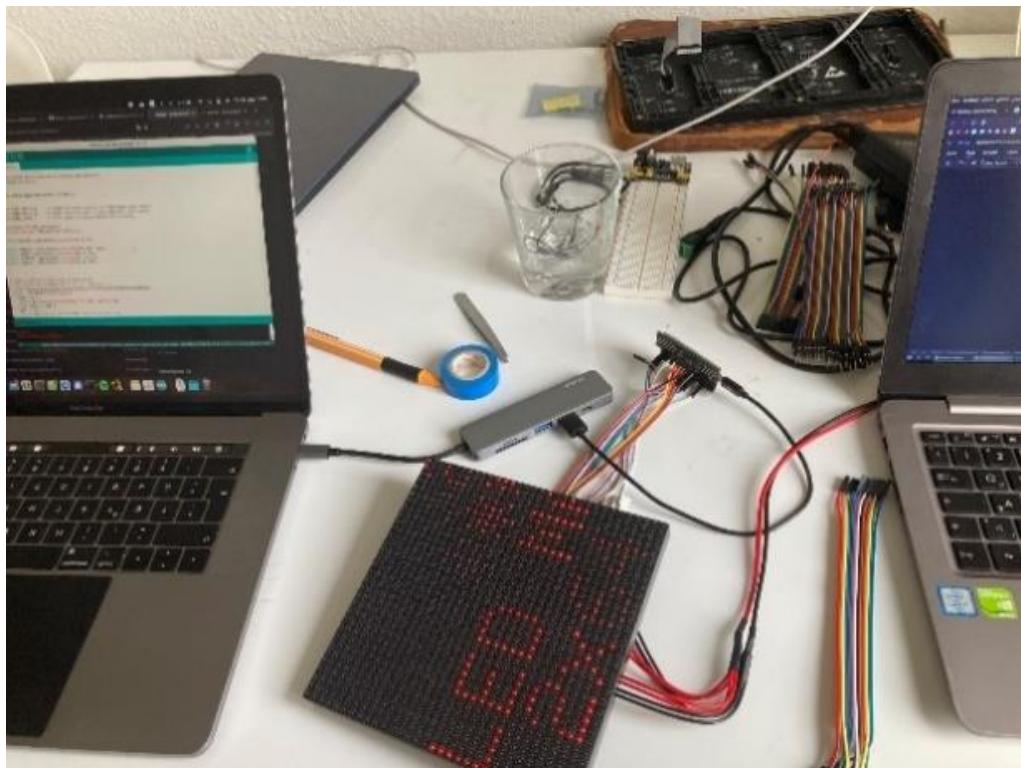
1. Verbindung ESP mit LED 32x32

- über Jumper-Kabel
- Probleme/Herausforderungen:
 - Jumper-Kabel physisch schwierig einzustecken (ggf. schlechte Qualität)
 - Mapping-Beschriftung auf LED leicht anders als in Doku (<https://github.com/mrfaptastic/ESP32-HUB75-MatrixPanel-I2S-DMA>): Nummerierung beginnend mit „0“ statt mit „1“
 - Code Tests können mithilfe von Wokwi durchgeführt werden (<https://wokwi.com/projects/342872544243614291>)



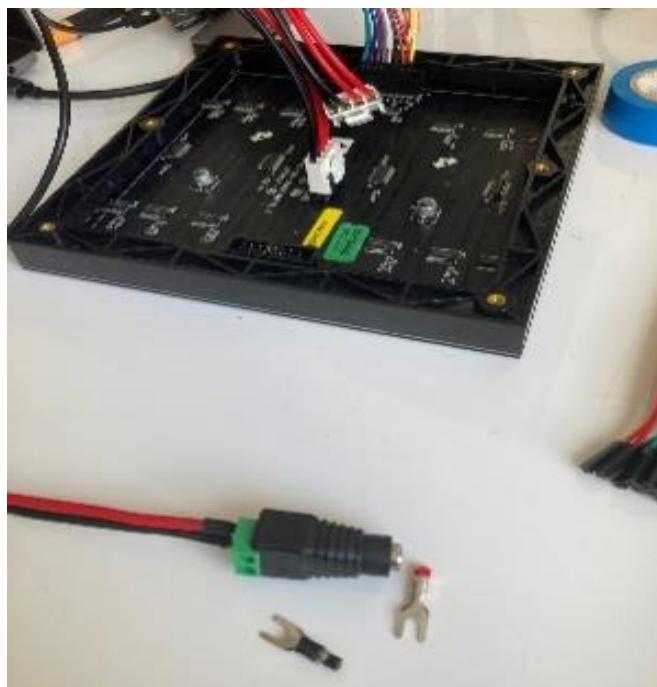
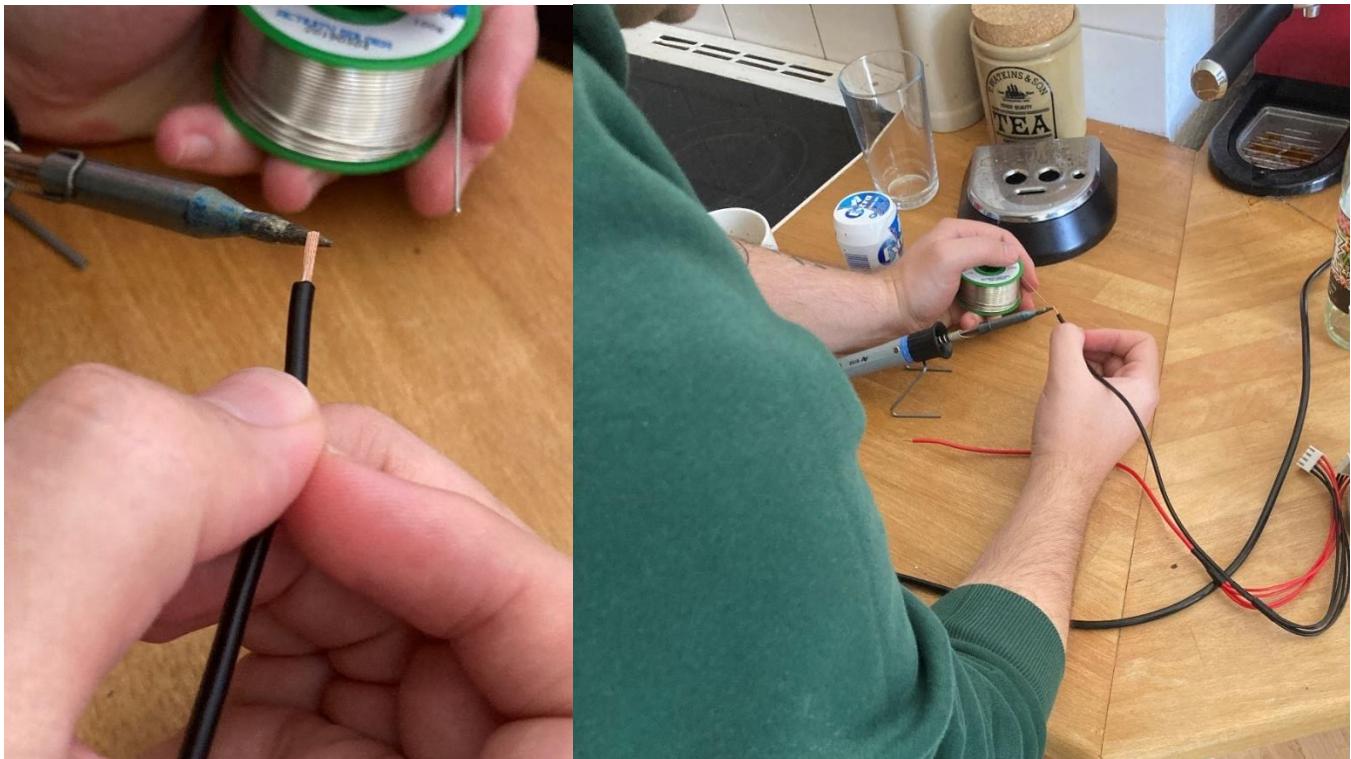
2. Verbindung ESP mit Laptop über Micro-USB-Kabel

- Allgemeine Verbindungsherstellung & Test ESP mit LED 32x32
- Test der LED 32x32 mit „Simple Test Shapes“ (https://github.com/mrfaptastic/ESP32-HUB75-MatrixPanel-I2S-DMA/tree/master/examples/1_SimpleTestShapes) → fehlerfrei
- Noch ohne Netzteil, Stromversorgung LED nur über ESP → LED-Anzeige zu schwach



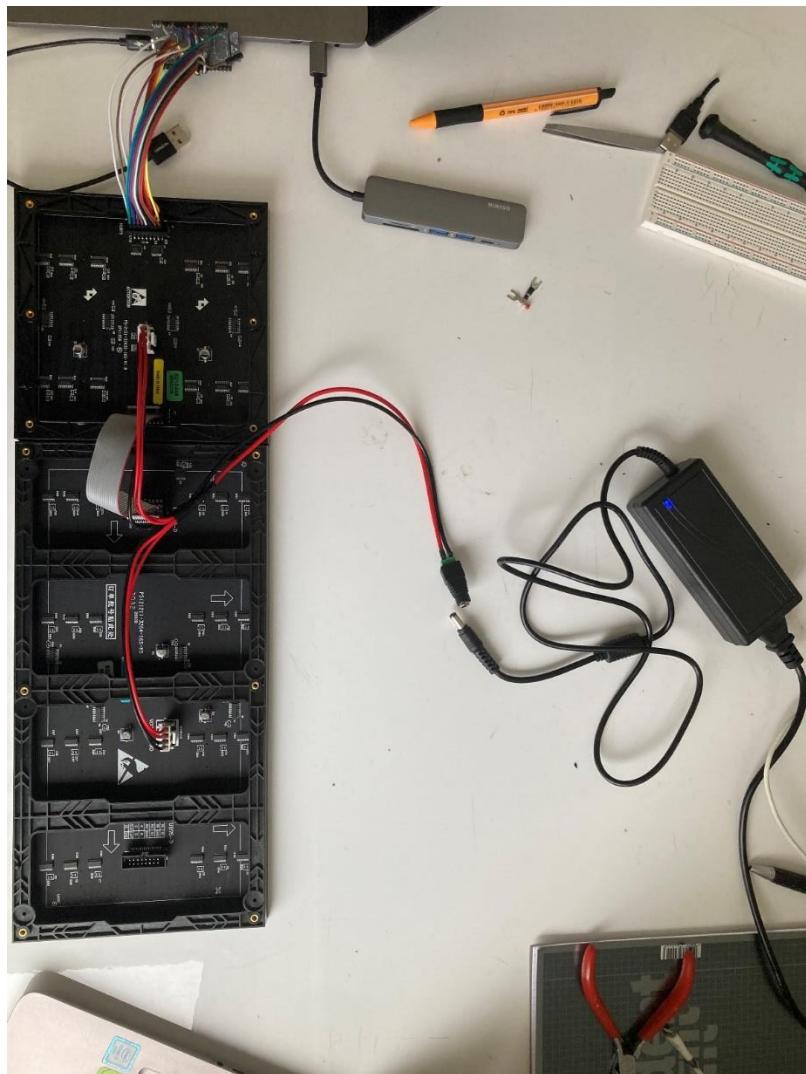
3. Test Stromversorgung: ESP mit LED 32x32

- Verbindung: ESP → LED 32x32
- Verbindung Stromkabel des LED 32x32 über Hohlstecker Buchse zu 5V-Netzteil
- Verbindung ESP über Micro-USB-Kabel an Laptop
- Anzeige-Test ok, aber: Floating-Ground-Problem aufgetreten → Flackern des LED 32x32



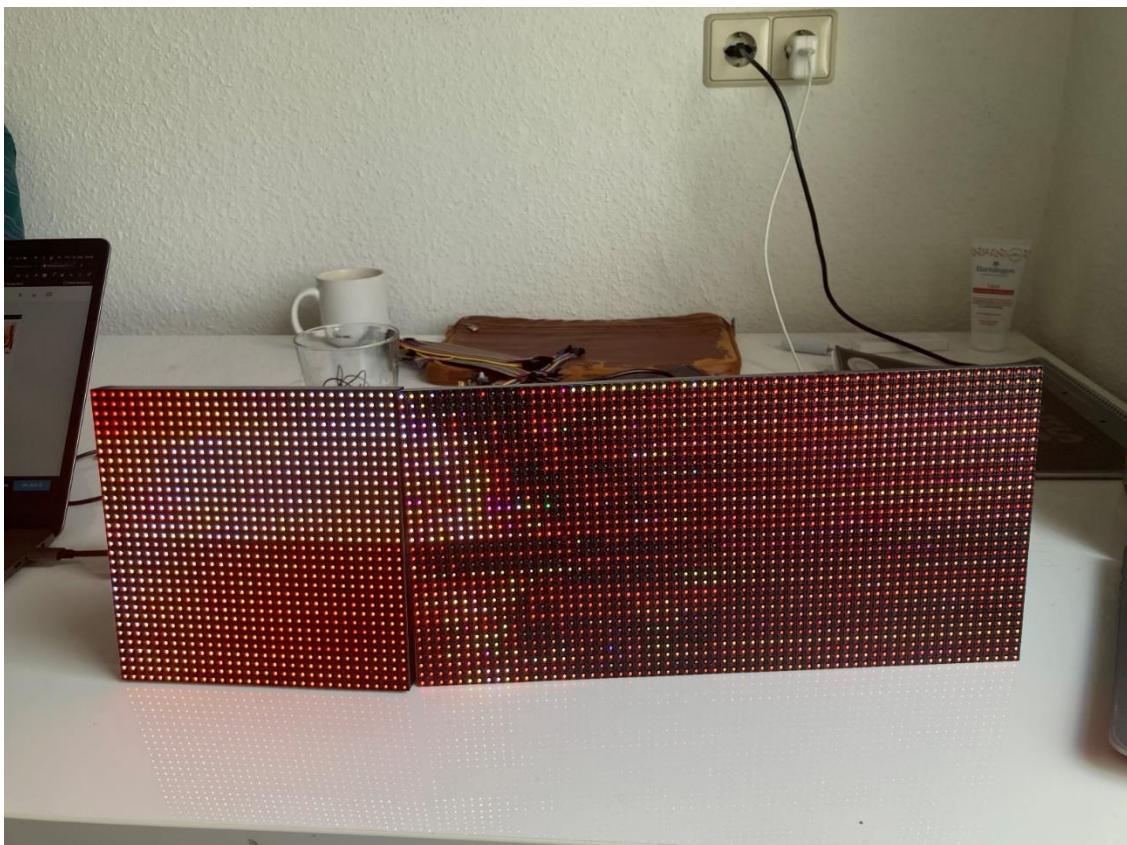
4. Test Stromversorgung: ESP mit LED 32x32 + LED 64x32

- Verbindung: ESP → LED 32x32 → LED 64x32
- Verbindung der Stromkabel der LED 64x32 & LED 32x32 über Hohlstecker Buchse zu 5V-Netzteil
- Verbindung ESP über Micro-USB-Kabel an Laptop
- Test ok, aber: Floating-Ground-Problem aufgetreten → Flackern des LED 64x32



5. Test Anzeige: ESP mit LED 32x32 + LED 64x32

- Verbindung: ESP → LED 32x32 → LED 64x32
- HUB75-Verbindungsleitung für Verbindung LED 32x32 zu LED 64x32
- Test beider LEDs zusammengeschaltet, erneut nach „Simple Test Shapes“ (https://github.com/mrfaptastic/ESP32-HUB75-MatrixPanel-I2S-DMA/tree/master/examples/1_SimpleTestShapes) → fehlerhafte Anzeige

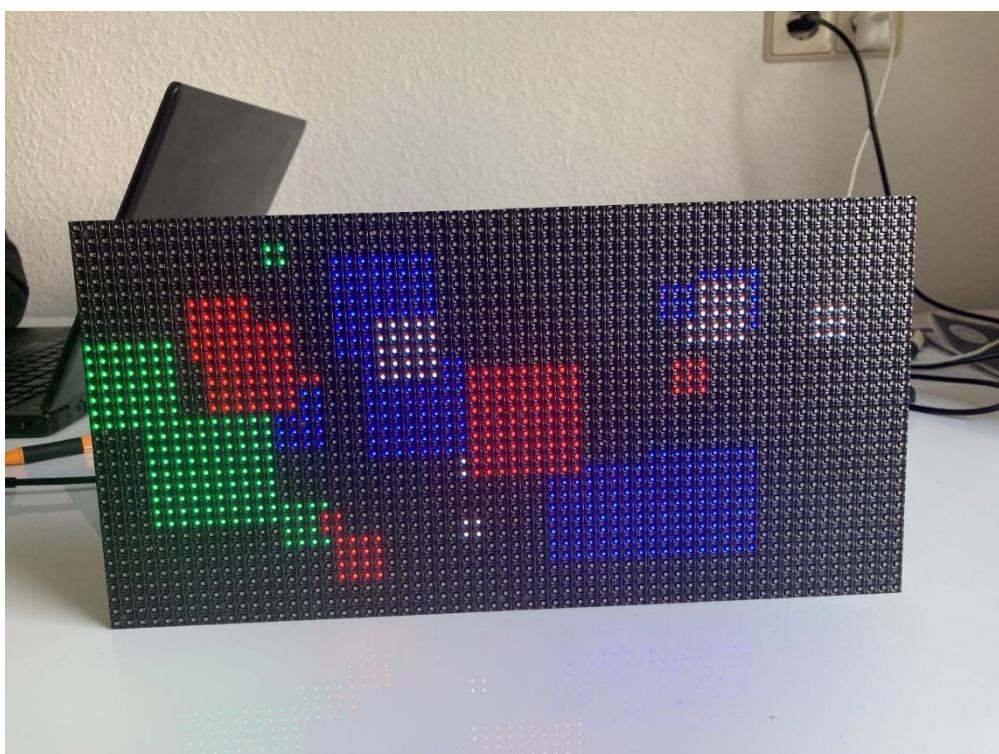
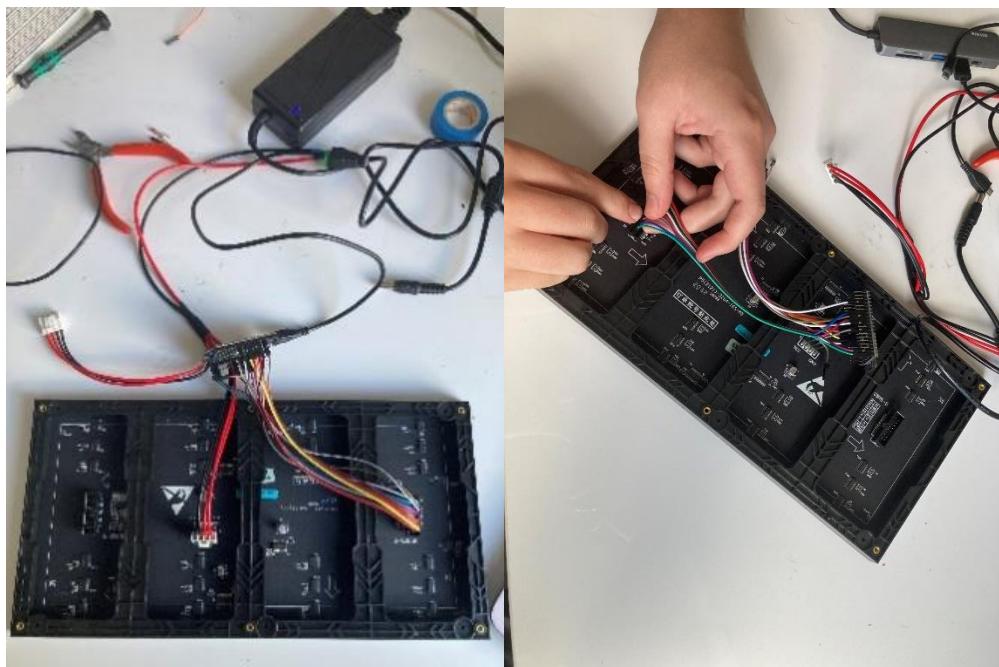


6. Test Anzeige: ESP mit LED 32x32

- Verbindung ESP → LED 32x32
- Test mit Bouncing Squares (<https://github.com/mrfaptastic/ESP32-HUB75-MatrixPanel-I2S-DMA/tree/master/examples/BouncingSquares>) → fehlerhafte Anzeige

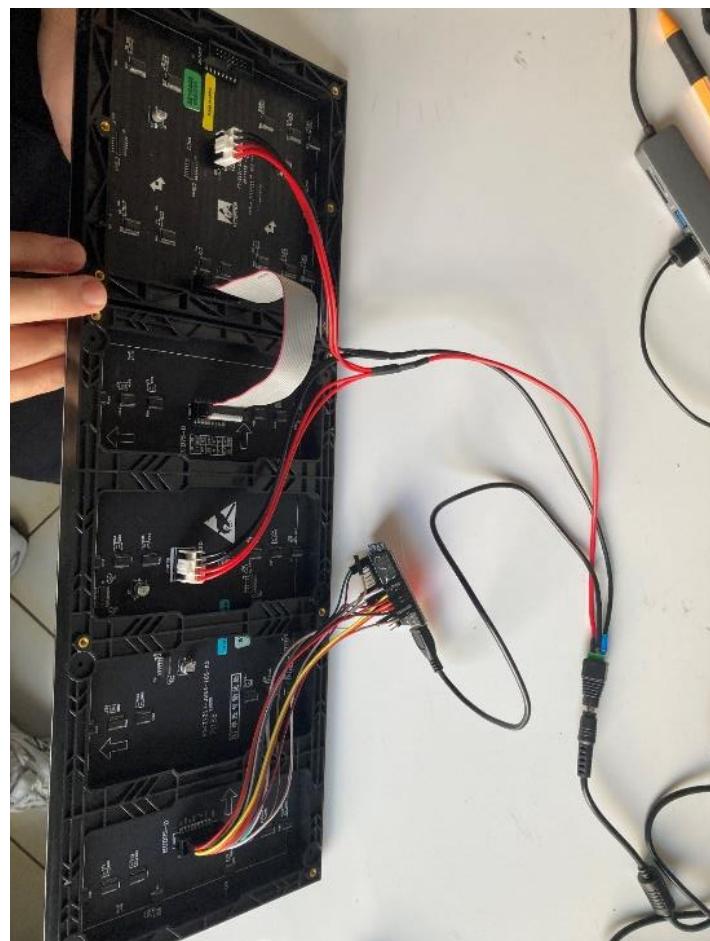
7. Test Anzeige: ESP mit LED 64x32

- Verbindung: ESP → LED 64x32
- Test mit „Bouncing Squares“ (<https://github.com/mrfaptastic/ESP32-HUB75-MatrixPanel-I2S-DMA/tree/master/examples/BouncingSquares>) → in Ordnung, fehlerfreie Anzeige



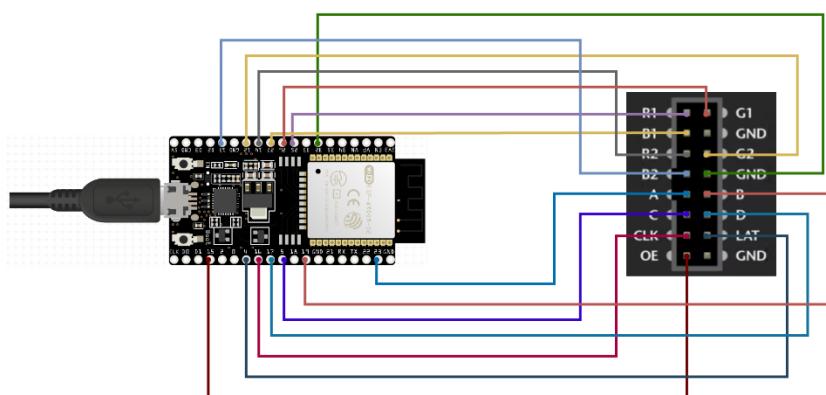
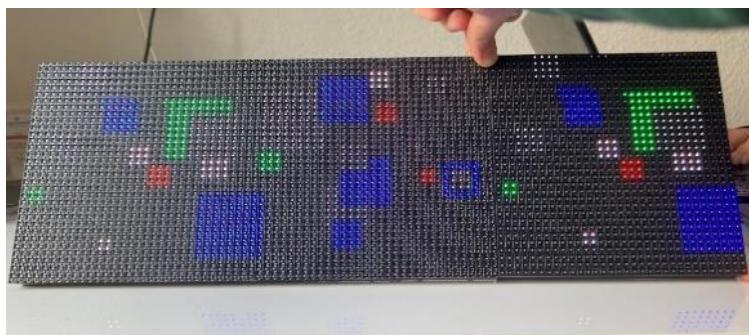
8. Test Anzeige: ESP mit LED 64x32 + LED 32x32

- Änderung LED-Reihenfolge
- Verbindung: ESP → LED 64x32 → LED 32x32
- HUB75-Verbindungskabel für Verbindung LED 64x32 zu LED 32x32
- Test mit „Bouncing Squares“ (<https://github.com/mrfaptastic/ESP32-HUB75-MatrixPanel-I2S-DMA/tree/master/examples/BouncingSquares>) → keine verbundene Anzeige auf LEDs, jeweils nur einzeln
- Anzeige auf LED 64x32 fehlerfrei wie zuvor
- Anzeige auf LED 32x32 fehlerhaft: erkannt als LED 64x32, deshalb nur halbe Anzeige, sonst korrekte Anzeige



9. Test Anzeige: ESP mit LED 64x32 + LED 32x32 ohne separaten Ground

- Verbindung: ESP → LED 64x32 → LED 32x32
- Wesentliche Änderungen zu Schritt 8:
 - Entfernung der Ground-Verbindung zwischen ESP und LED 64x32
 - Verwendung eines anderen Netzteils mit wirklicher 5V-Spannung (zuvor leicht über 5V)
- Test mit „Bouncing Squares“ (<https://github.com/mrfaptastic/ESP32-HUB75-MatrixPanel-I2S-DMA/tree/master/examples/BouncingSquares>) → Anzeige fehlerfrei & über beide Panels
- Test mit „Aurora Demo“ (<https://github.com/mrfaptastic/ESP32-HUB75-MatrixPanel-I2S-DMA/blob/master/examples/AuroraDemo/AuroraDemo.ino>) → Anzeige fehlerfrei & über beide Panels
- Finale Komponenten-Konstellation
- Schaltplan: siehe letztes Bild



4.1.2 Modellierung und 3D-Druck Gehäuse

Im Rahmen des Projekts wurde mithilfe eines 3D-Druckers ein Gehäuse konzipiert und gedruckt. Dieses dient dazu, die zwei LEDs als hauptsächliche Komponenten statisch zu verbinden und ihnen Stabilität zu verleihen. Zusätzlich wurde an der Vorderseite eine Halterung gedruckt, in die eine Plexiglasscheibe für weiteren Schutz der LEDs eingesetzt werden kann. Die einzelnen Arbeitsschritte umfassten dabei folgendes:

1. Maße nehmen und Suche nach Druckvorlagen

Zunächst wurden die Maße der Halterungen der LED-Matrizen genommen. Da in diesem Projekt beide Matrizen zusammengebaut werden müssen, erschwerte dies die Suche nach einer passenden Druckvorlage. Letztendlich wurden einzelne Vorlagen für die Modellierung des Gehäuses in Betracht gezogen, jedoch wichen die Maße der gewählten Gehäuse zu weit von unseren Matrizen ab.

2. Konzeption einer eigenen Lösung (UseCase-spezifisch)

Da keine passende Druckvorlage gefunden wurde, konzipierten wir eine eigene Lösung, welche für unseren Nutzen zugeschnitten wurde. Wichtig war dabei die Möglichkeit der Justierung der beiden Matrizen in einem Rahmen, sowie die Anbringung einer Frontscheibe zum Schutz der LEDs.

3. Modellierung der Einzelteile

Die Einzelteile wurden mit der Website <https://www.tinkercad.com/> modelliert. Da der Rahmen für die LEDs größer ist, als die Platte des 3D-Druckers, muss dieser in zwei Teile geteilt werden. Gleiches gilt für die Halterung der Plexiglasscheibe.

4. Druckeinstellungen anpassen

Damit der Druck eine ausreichende Qualität hat, müssen unterschiedliche Druckeinstellungen angepasst werden.

5. Testdruck durchführen: 7-mal bis gewünschtes Ergebnis erreicht

Um die Druckeinstellungen zu verproben wurden Testdrucke durchgeführt. Diese Testdrucke dauerten aufgrund der relativ großen Rahmengröße zwischen 3 und 15 Stunden. Nach einigen Iterationen standen die optimalen Druckeinstellungen für jedes Einzelteil fest.

6. Finale Drucke

Anschließend wurden die finalen Drucke für den Rahmen, als auch für die Plexiglashalterung durchgeführt.

7. Beschaffung Plexiglasscheibe

Die Plexiglasscheibe wurde nach der Fertigstellung des Drucks im Baumarkt für die Rahmengröße zugeschnitten, sodass diese bestmöglich in die Halterung hineinpasst.

8. Zusammenbau der Einzelteile

Im Anschluss wurden die Einzelteile zusammengesetzt und die Hardware wurde erneut für Bohrung der Löcher im Rahmen vermessen, damit diese fest montiert werden kann.

9. Einsetzen der Hardwarekomponenten

Zum Schluss wurde die Hardware in den Rahmen eingesetzt und festgeschraubt. Die Plexiglasscheibe wurde in der entsprechenden Halterung eingeklebt. Diese wurde dann wiederum mithilfe von Magneten auf den Rahmen der LED-Matrizen gesetzt.

4.2 Softwareentwicklung

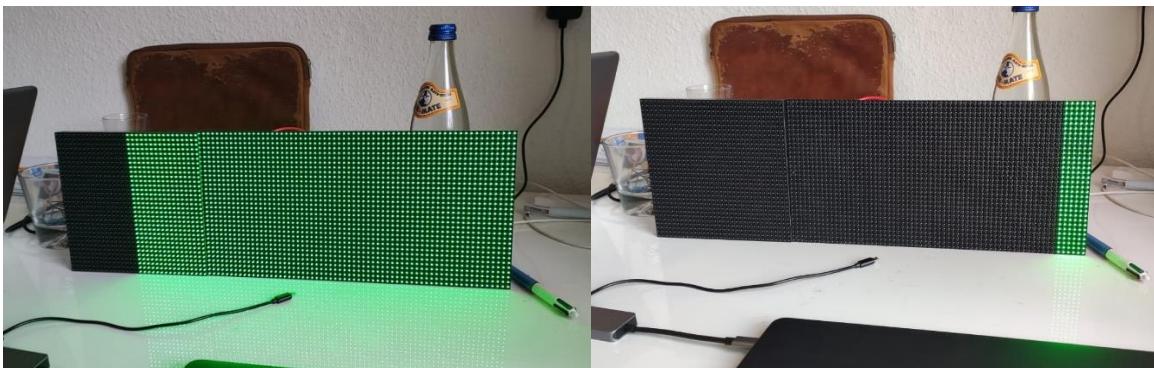
Dieses Kapitel umfasst eine Beschreibung über die einzelnen Schritte der Softwareentwicklung. Dabei werden die einzelnen Entwicklungsschritte des Grüne Welle Assistenten und des urban compass im Groben erläutert, um ein generelles Verständnis für das gewählte Vorgehen zu vermitteln. Der vollständige Code kann aus der technischen Dokumentation bzw. dem GitHub Repository zum Projekt unter <https://github.com/lucasoldenburg/urbancompass> entnommen werden.

4.2.1 Grüne Welle Assistent

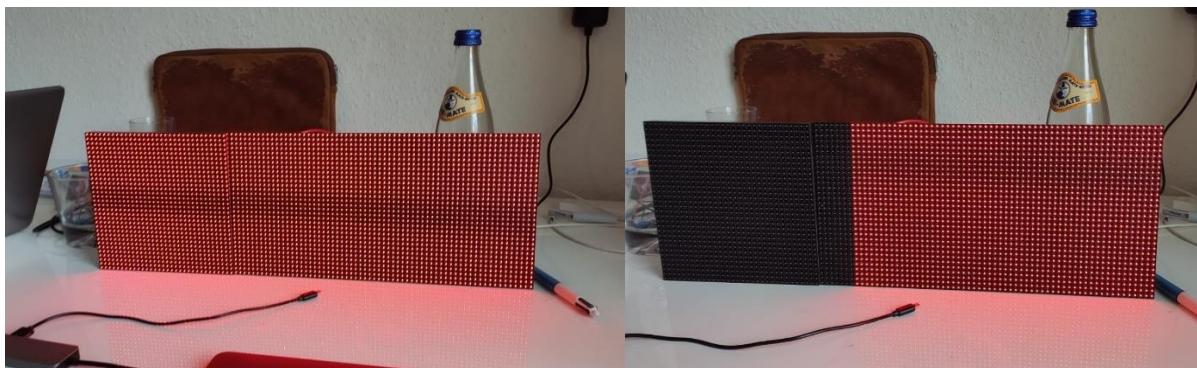
Im Folgenden werden die Entwicklungsschritte des Grüne Welle Assistenten erläutert.

4.2.1.1 Anzeige Balken

- Verwendung der Library Adafruit GFX für alle Methoden
- Matrix einfarbig füllen mit *fill color()*, Farbe grün
- Spaltenweise überschreiben mit „leeren Pixeln“, um Füllung schrittweise zu entfernen



- Matrix erneut einfarbig füllen mit *fill color()*, Farbe rot
- Spaltenweise überschreiben mit „leeren Pixeln“, um Füllung schrittweise zu entfernen



4.2.1.2 Anzeige Fahrrad-Symbol

- Zeichnen eines Fahrrad-Piktogramms mit Pixilart (<https://www.pixilart.com/>) und speichern als Bitmap
- Anschließend Konvertierung in HEX-Code für Verwendung in Arduino-Code (<https://github.com/mrfaptastic/ESP32-HUB75-MatrixPanel-I2S-DMA/blob/master/examples/BitmapIcons/bmp2hex.py>)
- Zeichnen des Fahrrad-Symbols auf LEDs mit drawBitmap()



4.2.2 Urban Compass

Folgende Entwicklungsschritte wurden im Rahmen der Softwareentwicklung des urban compass durchgeführt:

- Prototypisches Zeichnen einer der in Kapitel 3.2 (Fachkonzeption) erstellten Infotainment-Inhalte mit Pixilart (<https://www.pixilart.com/>) und speichern als Bitmap
- Anschließend Konvertierung in HEX-Code für Verwendung in Arduino-Code (<https://github.com/mrfaptastic/ESP32-HUB75-MatrixPanel-I2S-DMA/blob/master/examples/BitmapIcons/bmp2hex.py>)
- Zeichnen der Inhalte mit *drawXbm565()*: Draw-Funktion von <https://github.com/mrfaptastic/ESP32-HUB75-MatrixPanel-I2S-DMA/blob/master/examples/BitmapIcons/BitmapIcons.ino>
- Ergebnis: fehlerhafte Anzeige (siehe folgende Bilder)



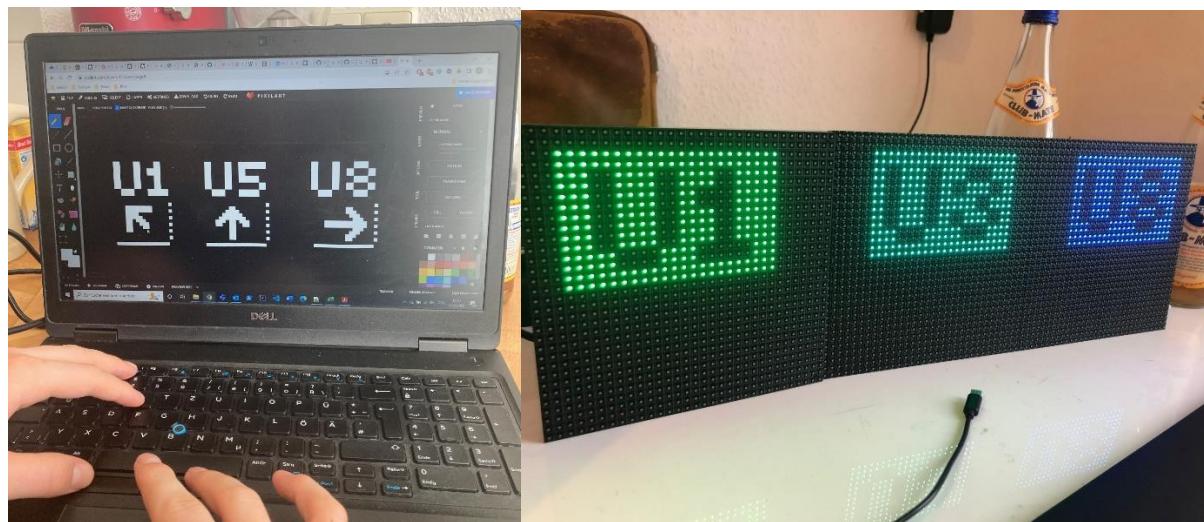
- auf `drawBitmap()` Funktion der Adafruit GFX Library gewechselt
- Änderung:
 - Umwandlung der Bitmap in HEX (muss nicht mehr gespiegelt/geflipped sein)
 - Änderung des Aufrufs von Skript (<https://github.com/mrfaptastic/ESP32-HUB75-MatrixPanel-I2S-DMA/blob/master/examples/BitmapIcons/bmp2hex.py>)
- Bitmaps für alle vier Kategorien der Infotainment-Inhalte erstellt und optimiert (Bezirke, Sehenswürdigkeiten, Uhrzeit, ÖPNV)
- Code-Anpassung: Iteration durch verschiedene Bitmaps nacheinander → Wechsel der Anzeige
- Ergebnis: Kategorien Uhrzeit, Sehenswürdigkeiten und Bezirke werden fehlerfrei angezeigt (siehe folgende Bilder)



- Anzeige der Kategorie ÖPNV noch fehlerhaft: Fehler bei der Konvertierung zu Schwarz-Weiss



- Erstellung Bitmaps für einzelne Elemente der ÖPNV-Anzeige (farbige Umrandungen, Schrift und Pfeile)
- Einzelne Bitmaps der farbigen Umrandungen und Linienkennung sowie Pfeile mit *drawBitmap()* in einzelnen Farben übereinander gezeichnet, um mehrfarbiges Bild zusammen zu setzen
- Ergebnis: auch fehlerfreie Anzeige der Kategorie ÖPNV



5 Projektergebnis

Im Folgenden wird das Projektergebnis als Bildergalerie dargestellt. Beide Projektziele, sowohl die Umsetzung des PoC Grüne-Welle-Assistenten als auch der urban compass, konnten vollständig erreicht werden. Der PoC gilt als erfolgreich umgesetzt, da alle in Kapitel 3.1 definierten Schritte im Rahmen des Abnahmeszenarios fehlerfrei durchgeführt werden können. Auch die Anzeige der Infotainment-Inhalte im Rahmen des „urban compass“ ist vollständig (alle Kategorien und Inhalte) und fehlerfrei (korrekte Anzeige), weshalb auch dieses Ziel als erreicht gilt.





6 Ausblick

In Bezug auf den urban compass wurden im Verlauf der prototypischen Umsetzung bereits einige offene Aspekte und Optimierungspotenziale identifiziert, die im Folgenden aufgeführt werden:

- Optimierung der Anzeige & Inhalte für den urban compass:
 - Identifikation weiterer Infotainment-Inhalte sowie Bestimmung der relevantesten Inhalte bzw. Kategorien. Umsetzung ggf. über Befragung von Fahrradfahrenden, Sichtung von Studien und durch den Einsatz interdisziplinärer Teams (z.B. Psychologen, Stadtplaner, Designer, etc.).
 - Hinzufügen von sprechenden Icons, sodass die angezeigten Informationen von Fahrradfahrenden besser eingeordnet werden können.
 - Je nach Standort eines urban compass Angabe von Kilometern für die Distanz zu Orten wie Bezirken oder Sehenswürdigkeiten und Angabe von Minuten für nächste Anbindungen des ÖPNV.
 - Ermittlung der optimalen Anzeigedauer je Kategorie über die durchschnittliche Geschwindigkeit der Fahrradfahrenden.
- Weitere LED-Anzeige in Außenrichtung der Radbahn: Gegebenenfalls wäre die Umsetzung einer zweiten Anzeige sinnvoll, die in die Außenrichtung der Radbahn zeigt. Auf dieser könnten andere Infotainment-Inhalte angezeigt werden, die zum Beispiel Fußgänger über die Radbahn informieren und zur Nutzung dieser anregen.
- Solar-Betrieb: Gegebenenfalls könnte der Betrieb der Hardware-Komponenten über Solarzellen und Akkus ermöglicht werden und so die Abhängigkeit zu gesonderter Stromversorgung eliminiert werden.
- Witterungsfestigkeit: Das 3D-gedruckte Gehäuse könnte erweitert bzw. vervollständigt werden, indem eine Rückwand konzipiert und Dichtungen für die Abweisung von Wasser eingesetzt werden.

Für eine potenzielle Weiterführung des Projekts wird eine genaue Prüfung dieser o.g. Ansätze empfohlen, um den urban compass bzw. den Grüne Welle Assistenten weiter zu optimieren.