Low Power Room Display

Dokumentation des LPRD-Projekts

LPRD Maintainers

Table of contents

1. Einleitung

Jannis Gröger

1.1 Motivation

Ob es bei Meetingräumen einer Firma, Klassenzimmern einer Schule, Seminarräumen einer Tagungsstätte oder Vorlesungssälen der Hochschule ist - anzuzeigen, ob dieser Raum belegt ist und was dort gerade stattfindet, ist eine Herausforderung für den Betreiber und die Lösung für Schüler, Mitarbeiter oder Tagende nicht gerade benutzerfreundlich. Das Austauschen von Papieren, die die Raumbelegungsdaten anzeigt, bedeutet einen hohen Mehraufwand für Zuständige und einen nicht nachhaltigen Ressourcenverbrauch, während sich bei der Verwendung von digitalen Lösungen immer wieder die Frage nach der benötigten Infrastruktur zur Stromversorgung dieser Anzeigen stellt.

Aus diesem Grund sind so genannte Low Power Raumanzeigen eine interessante Alternative zu oben genannten Lösungen, da sie durch Optimierung des Stromverbrauchs nicht nur sehr energieeffizient, sondern durch Darstellen der Informationen auf einem Display eine umweltschonende Lösung für das Problem bieten. Darüber hinaus können solche Raumanzeigen meist über eine zentrale Software verwaltet und gesteuert werden und besitzen neben den oben genannten Nutzungsmöglichkeiten ein breites Anwendungsspektrum.

1.2 Ziel der Arbeit

Die vorliegende Projektarbeit beschäftigt sich umfassend mit der Entwicklung eines energieeffizienten Raumdisplays, das darüber hinaus als generische Informationsanzeige genutzt werden kann. Ziel dieser Arbeit ist es, detailliert aufzuzeigen, wie eine solche Anzeige von der Konzeptionsphase bis hin zur finalen Umsetzung realisiert werden kann. Dabei wird nicht nur auf die technischen und theoretischen Grundlagen eingegangen, sondern es werden auch die praktischen Vorgehensweisen der Entwicklung ausführlich beschrieben und für den Leser anschaulich dargestellt. Es wird erklärt, welche Technologien und Materialien verwendet werden, welche Herausforderungen während der Entwicklungsphase auftreten und wie diese gelöst werden können. Des Weiteren wird beleuchtet, wie das Display hinsichtlich seiner Energieeffizienz und Steuerung optimiert werden kann, um eine nachhaltige und einfache Nutzung zu gewährleisten. Dem Leser soll ein umfassendes Verständnis über den gesamten Entwicklungsprozess gegeben werden, sodass dieser in der Lage ist, ähnliche Projekte und Konzepte eigenständig umzusetzen.

1.3 Aufbau der Arbeit

Zu Beginn der Arbeit werden vorhandene Technologien der einzelnen Komponenten einer energieeffizienten Raumanzeige analysiert, um ein grundlegendes technisches Verständnis über die Funktionsweise dieser zu schaffen. Ebenso werden dem Leser mögliche Softwareframeworks und Bibliotheken, näher gebracht die zur Umsetzung eines solchen Projektes genutzt werden können. Im Anschluss werden bereits auf dem Markt vorhandene Produkte beleuchtet und der aktuelle Stand der Technik aufgezeigt. Desweiteren wird das Projektteam vorgestellt und die Organisation und Arbeitsweise intern kurz beschrieben.

Im Hauptteil der Arbeit wird zunächst das Systemkonzept zusammen mit den Anwendungsszenarien und Anforderungen an das Produkt vorgestellt. Anschließend wird die Entwicklung der Low Power Raum Anzeige erläutert und anhand der vorher geschaffenen Grundlagen die Auswahl der verwendeten Komponenten erklärt. Hierbei spielen energiesparende Display-Technologien, sowie auch Optimierung des Stromverbrauchs und Minimierung des Datenaustauschs zwischen dem Displaymodul und Endgerät des Benutzers eine Rolle. Darauf folgend wird die Firmware der Raumanzeige tiefgehend analysiert und ihre Funktion dem Leser erläutert. Ein besonderer Fokus liegt hierbei auf der Implementierung der Displayansteuerung, sowie der Optimierung des Energierverbrauchs durch optimales Hardwaremanagement. Im letzten Kapitel des Hauptteils wird auf die Webentwicklung des Projekts eingegangen. Diese umfasst die Gestaltung eines User Interfaces zur Steuerung des Displays vom Endgerät des Nutzers, aber auch eine serverseitige Webanwendung zur Verwaltung und Steuerung mehrerer Displaymodule. Durch diese Maßnahmen wird sichergestellt, dass das Low Power Raumdisplay nicht nur eine lange Akkulaufzeit aufweist, sondern auch eine einfache und benutzerfreundliche Bedienung zur Verfügung stellt.

Zum Schluss der Arbeit werden die Ergebniss aus der Ausarbeitung und Entwicklung der Low Power Raumanzeige aufgezeigt und ein Fazit über die Arbeit gezogen. Zusätzlich wird ein Ausblick auf mögliche Verbesserungen und Erweiterungen gegeben und auf bereits begonnene Implementierung dieser hingewiesen.

2. Praktische und Theoretische Grundlagen

2.1 Mikrocontroller

Stasa Lukic

Das Gehirn des Projektes ist ein Mikrocontroller, es gibte verschiedene arten von Mikrocontrollern die man vergleichen muss um einen für sich passenden Mikrocontroller zu finden. Im folgendem Unterkapitel werden die von uns ausgewählten Mikrocontroller genauer dargestellt, ihre Eigenschaften, nachteile und auch vorteile.

Für das Projekt haben wir uns Prioritäten für den Mikrocontroller gesetzt. Als ersters sollte er genug Speicher haben sodas kein weiterer externer Speicher genutzt werden muss, dies würde unsere Kosten und gleichzeitig den Strombedarf senken da ein externes Speicher Modul Strom verbraucht und eine Latenz verursacht die längere arbeitszeiten bedeuten.
WLan sollte intern auf dem Board verbaut seien da unser Mikrocontroller unter keiner so hohen Last arbeiten wird, das unsere Netzwerk Last auf einen sub Mikrocontroller verlagert werden müsste.
Als letztes sollte der Mikrocontroller über einen deep sleep verfügen[STA_06].
So würden wir am meisten Strom sparen da unserer Mikrocontroller nach beenden einer aufgabe (wie z.b. ein neues Bild darstellen) wahrscheinlich für längere Zeit keine Aufgaben hat oder bekommt. Der Mikrocontroller sollte eine Schnittstelle für SPI besitzen für eine mögliche schnelle datenübertragung mit dem Display.

In Tabelle 2.1 sind die Mikrocontroller mit ihren Eigenschaften aufgelistet. Es wurden von insgesamt 3 Marken recharchiert. Diese währen Espressif, STMicroelectronics und Raspberry Pi.

Die Mikrocontroller wurden nicht als SMD Chip gekauft sondern als development boards. Für Espressif recharchierten wir die XIAO ESP32 S3 und XIAO ESP32 C3 Boards von Seeed Studio und Die D1 Mini ESP8266 Boards von AZ-Delivery. Bei dem STM32 guckten wir uns den STM32WL55 an. Der RP2040 ist der RASP PI PICO von Raspberyy Pi.

Eigenschaften	**ESP32 S3 [STA_01]**	**ESP32 C3[STA_02]**	**ESP8266 [STA_03]**
Prozessor	Dual-core 240MHz	Single core 160MHz	Singlecore 160MHz
Speicher	8MB Flash 8MB PSRAM	400KB SRAM 4MB Flash	4MB Flash
Wifi	2,4GHz WLAN	2,4GHz WLAN 802.11b/g/n	2,4GHz WLAN 802.11n
Spannung	3,3V	3,3V	3,3V
Idle Strom	22mA	24.4mA	Maximum 500mA *
Sleep Strom	14 µA	43µA	
Schnittstellen	SPI, UART, IIC, IIS, 11xGPIO, 9xADC	SPI, IIC, UART, 11xGPIO, 9xADC	SPI, IIC, UART, 9xGPIO, ADC
Weitere Eigneschaften	Reset Button Boot button Battery Charge Chip für 100mA Externe Antenne	Boot button Reset Button Battery Charge Chip für 100mA Externe Antenne	Reset button möglich Antenne in Platine verbaut

Eigenschaften	STM32 [STA_04]	RP2040 [STA_05]
Prozessor	Singlecore 48MHz	Dual-core 133MHz
Speicher	256KB Flash 64KB RAM	2MB Flash 256KB RAM
Wifi	2,4GHz LoRa	Keins
Spannung	5V	5V
Idle Strom	15 mA	24mA
Sleep Strom	360nA	1,3mA
Schnittstellen	2xSPI, 3xIIC, 2xUART, 43xGPIO	SPI, 2xIIC, 2xUART, 30xGPIO
Weitere Eigneschaften		

Tabelle 2.1: Eigenschaften der Mikrocontroller

*Keine Angaben außer Maximum

2.2 Stromsparende Display-Technologien

Stasa Lukic

Für dieses Projekt brauchen wir ein Display das so wenig Strom verbraucht wie möglich, damit unser Room display solange wie möglich mit einer Akku Ladung aushält. Wir fokussierten unsere Recherche auf ePaper, bistabile LCDs und OLED displays.

2.2.1 OLED

OLED Bildschirme haben eine sehr gute Bildqualität und einen weiten betrachtungswinkel. Anstatt Halbleiter werden in den LEDs Organische Molekühle verwendet die Licht erzeugen. Diese Molekühle befinden sich zwischen ander Anode und einer Kathode. Fließt nun Strom wird blau und gelbes Licht erzeugt, mit Farbfiltern entstehen die Restlichen Farben. OLED Bildschirme können sehr helle Bilder zu erzeugen mit starkem Kontrast oder Farbstarke Bilder, dies verbraucht aber mehr Strom mit mehr Helligkeit, was für dieses Projekt ein Nachteil ist. Ein Vorteil ist, dass schwarze Pixel kein Strom verbrauchen gegenüber älteren Displays[STA_09].

2.2.2 Bistabile LCD

Bistabile LCD Displays haben den Vorteil das sie im gegensatz zu LCD keinen durchgehenden Stromverbrauch haben, Strom wird nur dann verbraucht wenn das Bild gewechselt wird. Dafür benutzt das Display kleine Kristalle die mit Hilfe von einem Stromfluss einen von zwei Zuständen annehmen kann, der Kristall lässt entweder Licht durch oder reflektiert es zurück, womit man ein Schwarz und Weiß Bildschirm bauen kann. Benutzt man mehrere Ebenen kann man sogar verschiedene Farben darstellen. Wenn das Bild des Displays nur selten geändert wird, bedeutet es einen sehr niedrigen durchschnittlichen Strom verbrauch. Der Nachteil ist, dass das Aktualisieren des Bildes zwei bis drei Sekunden dauert [STA_08].

2.2.3 ePaper

ePaper hat ein ähnliches Konzept wie Bistabile LCD Displays, sie verbrauchen nur dann Strom wenn das Bild aktualisiert wird. Hier werden geladene Farbpartikel in sehr kleinen Zellen eingesperrt. Je nachdem welche Ladung auf eine Zelle ausgeübt wird, werden manche Farbpartikel angezogen oder weggestoßen. Der Nachteil ist, dass die Aktualsierung eines Bildes je nach ePaper Display-Technologie sehr lange dauern kann. Das ist weil, wie man sich vorstellen kann, muss man einen Pixel mehrmals laden, um wirklich alle Farbpartikel abzustoßen, die man abstoßen will oder genau umgekehrt. Das resultiert in einer Aktualiesierungsrate von bis zu 26 Sekunden oder kürzer je nach Farbleistung. Ein weiter Nachteil ist, dass Farbpartikel nur negativ oder positiv geladen werden können, was in nur zwei Farben resultiert. Durch mehrere Ebenen kann das Display

mehrere Farben annehmen, aber bei weitem nicht so viele wie beim Bistablem LCD. Der Vorteil ist, dass sie noch weniger Strom verbrauchen beim Aktualsieren als die Bistablien LCDs [STA_07].

2.3 Akku-Technologien

Benjamin Klaric

Um das System möglichst autark zu machen, wurde es mit einem Akku betrieben. In Vorgang mit der Auswahl von passenden Akku-Technologien wurden die drei meistverbreiteten verglichen, nämlich die drei bedeutendsten Typen aus der Lithium-Ionen-Akku Familie: klassische Lithium-Ionen-Akkus, oft benutzten Lithium-Polymer-Akkus und die sicheren Lithium-Eisen-Phosphat-Akkus. Genauer gesagt sollte man die Vor- und Nachteile von jeweiligen Akku-Typen genauer anschauen und dementsprechend eine Technologie auswählen. In den nächsten Abschnitten werden die drei genannten Technologien genauer angeschaut und die jeweiligen Vor- und Nachteile genannt.

Lithium-Ionen-Akkus (Li-Ion)
Die meistverbreiteten Li-Ion-Akkus bieten relativ hohe Energiedichte, sind kostengünstig und langlebig. Die Li-Ion-Akkus kommen in verschiedenen Bauformen, wie z. B. 18650, 21700 usw. Dieser Art von Akkus besteht aus einer negativer Elektrode, bzw. Kathode aus Grafit und einer positiven Elektrode bzw. Anode aus Lithiumcobaltoxid ($LiCoO_2$), Lithiumnickeldioxid ($LiNiO_2$) oder Lithiummanganatoxid ($LiMn_2O_4$). Diese Wahl von Chemie bietet eine gute Zyklenfestigkeit, was natürlich der Langlebigkeit entspricht. Im geladenen Lithium-Ionen-Akku erzeugt ein elektrochemischer Prozess Spannung zwischen den Elektroden. Lithium-Ionen (Li+) bewegen sich dabei durch den Elektrolyten zwischen festen Übergangsmetall- und Grafitstrukturen der Elektroden, getrennt durch einen Separator. Das Funktionsprinzip von Li-Ion-Akkus bezieht sich dabei auf die Verschiebung von Lithium-Ionen. So wird die elektromotorische Kraft erzeugt. [BK_01]
Mit einer Nennspannung von 3,7V und der Entladekurve, die stabil über einen Großteil der Entladung bei Nennspannung bleibt, was in der Abbildung 2.1 zu sehen ist.
In der Abbildung sind vier verschiedene Kurven dargestellt. Die dunkelgrüne Kurve repräsentiert die Spannung während des Ladevorgangs, während die dunkelblaue Kurve die Spannung während des Entladens zeigt. Die hellgrüne Kurve stellt den Strom während des Ladevorgangs dar, und die hellblaue Kurve zeigt den Strom während des Entladens. Alle vier Kurven sind gegen die Zeit geplottet.

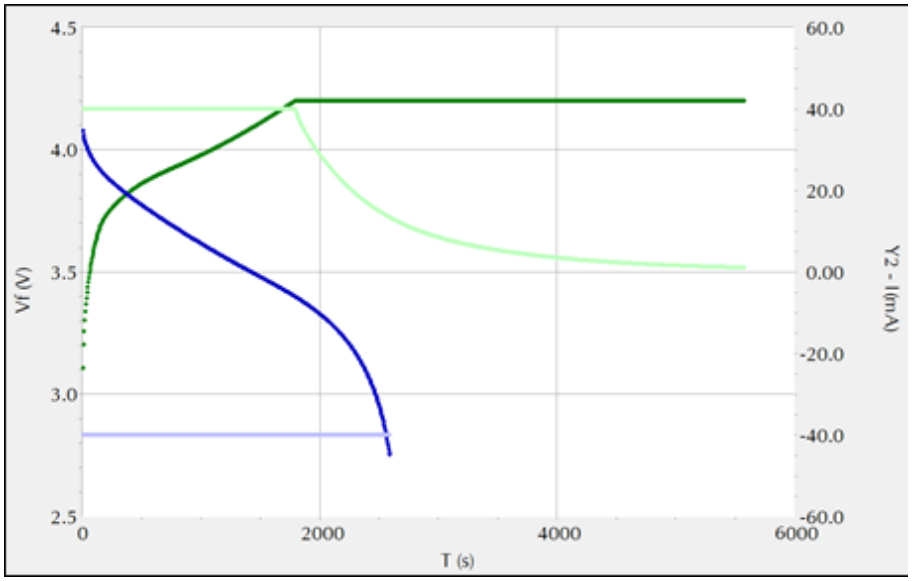


Abbildung 2.1: Lade- (dunkelgrün) und Entladekurve(dunkelblau) von einer Li-Ion-Knopfzelle [BK_02]

Li-Ion-Akkus können aber nicht selber in Betrieb genommen werden, da die sich ohne eine Art von Überwachung tief entladen wurden oder beim Laden überladen geworden sind. Dazu werden Li-Ion-Akkus immer mit einem Batteriemanagementsystem benutzt, kurz BMS gesagt. Ein BMS kümmert sich darum, dass die Li-Ion-Akkus einen Cut-Off-Spannungspegel beim Laden und beim Entladen haben. Das BMS bietet auch Kurzschlussschutz und meistens eine Überstromerkennungsfunktionalität. [BK_03]
Li-Ion-Akkus, voll geladen, liegen bei ungefähr 4,25V und entladen sich bis 2,5V, wenn die an einen BMS angeschlossen sind.

Ein Merkmal von Li-Ion-Akkus ist die Art des Ladens, da sie eine spezielle Ladetechnik benötigen, nämlich die sogenannte CC-CV-Methode (Constant Current - Constant Voltage). Genauer gesagt, werden die Akkus die ersten ~80% mit einem konstanten Strom geladen und die restlichen ~20% mit einer konstanten Spannung. Dieses Verfahren ist in Abbildung 2.1 zu sehen, dargestellt durch die grüne und hellgrüne Kurve, allerdings nicht im Verhältnis 80%-20%. Dafür wird ein besonderes Ladegerät benötigt. Mit richtiger Nutzung von Ladeverfahren haben die Akkus keinen Memory-Effekt und dementsprechend auch sehr geringe Selbstentladung. Die Li-Ion-Akkus sind meistens ein bisschen teurer als die anderen Akku-Technologien und benötigen die genannten speziellen Ladegeräte. [BK_01]

Lithium-Polymer-Akkus (LiPo)

Die LiPo-Akkus sind eine spezielle Bauform der klassischen Li-Ion-Akkus und nutzen daher die gleiche Zellchemie. Als eine der neuesten Akkutechnologien bieten sie eine der höchsten Energiedichten in Relation zum Gewicht. Sie behalten alle Vorteile der Li-Ion-Akkus, da sie Teil der Lithium-Ionen-Akku-Familie sind, und verwenden dasselbe Ladeverfahren wie klassische Li-Ion-Akkus. Oft sind sie mit einem eingebauten Batteriemanagementsystem (BMS) ausgestattet, was sie etwas teurer macht. Diese Akkus sind typischerweise in Mobiltelefonen zu finden. LiPo-Akkus verwenden flexible Kunststoffbeutel anstelle der zylindrischen Bauweise von Li-Ion-Akkus. Diese Flexibilität ermöglicht es, sie an die spezifischen Anforderungen des Gerätedesigns anzupassen und bietet eine größere Designfreiheit. Sie nutzen den verfügbaren Raum in elektronischen Geräten effizient aus.
Ein Nachteil liegt jedoch in den Sicherheitsaspekten. LiPo-Akkus sind sehr empfindlich gegenüber Überhitzung, was in Rauch- und Gasentwicklung oder im Extremfall sogar zu Selbstentzündung führen kann. Dieses Risiko erhöht sich besonders bei unsachgemäßer Handhabung, übermäßiger Belastung oder mechanischer Beschädigung der Akkuzellen. [BK_04]

Lithium-Eisen-Phosphat-Akkus ($LiFePO_4$)

Die $LiFePO_4$-Akkus gehören zur Familie der Lithium-Ionen-Technologie, verzichten jedoch auf das Kathodenmaterial Lithiumcobaltdioxid ($LiCoO_2$), das in klassischen Li-Ion-Akkus verwendet wird. Im Vergleich zu Li-Ionen-Akkus mit LiCoO2 bieten LiFePO4-Akkus eine höhere Sicherheit, da sie keine exothermen Reaktionen unter extremen Bedingungen zeigen, was das Risiko von Bränden und Explosionen erheblich verringert. Diese Akkus sind ideal für Anwendungen in Elektrofahrzeugen aufgrund ihrer höheren Entladestromkapazität, Nicht-Toxizität und längerer Lebensdauer im Vergleich zu Li-Ionen-Akkus mit $LiCoO_2$. [BK_05] Im Vergleich zu Li-Ion-Akkus liegt die Nennspannung von $LiFePo_4$-Akkus bei 3,3V und bleibt stabil über einen Großteil der Entladung bei Nennspannung, was auf Abbildung 2.2 zu sehen ist.

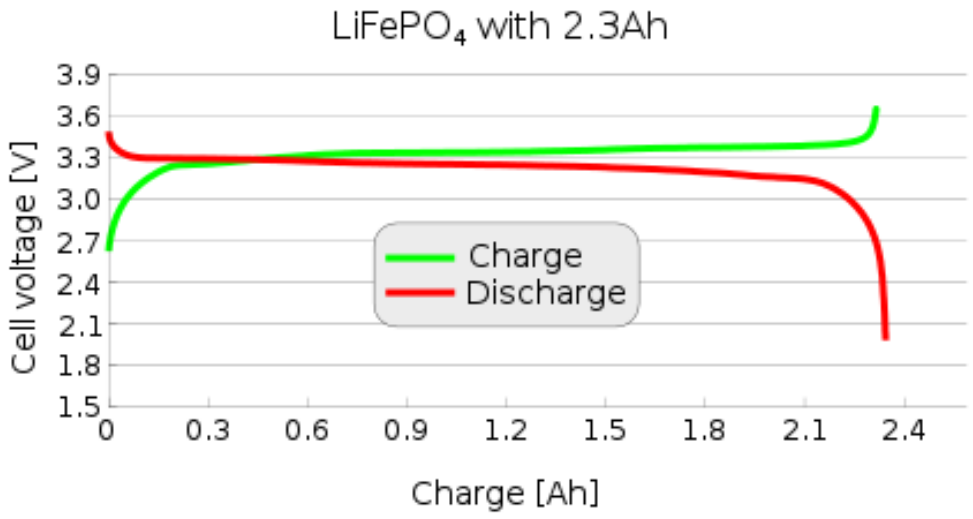


Abbildung 2.2: Lade- (grün) und Entladekurve (rot) von $LiFePO_4$-Akkus [BK_06]

Wie Li-Ion-Akkus benötigen die $LiFePO_4$-Akkus auch eine Schutzschaltung in Form von BMS, allerdings besondere Art für $LiFePO_4$-Akkus. Da sie auch zur Familie von Li-Ion-Akkus gehören, werden sie auch über gleiches Ladeverfahren wie Li-Ion-Akkus geladen. Die Entladeschlussspannung von diesen Akkus liegt typischerweise bei 2,0V und die Ladeschlussspannungen liegen in der Regel bei 3,6V. Die Werte sind aber von BMS abhängig.

Der einzige Nachteil dieser Akku-Technologie ist, dass im Vergleich zu den reinen Li-Ion-Akkus die Nennspannung niedriger liegt, bei 3,3V im Vergleich zu 3,7V. [BK_05]

2.4 Funk-Technologien

Stasa Lukic

2.4.1 LoRa

Low-Power-Wide-Area-Network-Technologie, auch genannt LoRa ist eine stromsparende Technologie, um Daten über eine längere Distanz zu schicken. Dafür benutz LoRa die CSS-Modulationstechnik für eine Funkreichweite von mehreren Kilometern. Durch den niedrigen Stromverbrauch ist LoRa geeignet für mobile IoT-Projekte. Diese können batteriebetrieben werden für eine längere Zeit. Die Netzwerktopologie von LoRa sind Punkt-zu-Punkt oder Punkt-zu-Mehrpunkt, oft in Kombination mit Gateways [STA_12].

2.4.2 WLAN

WLAN ist die weitverbreiteste Netzwerk-Technologie für den normalen Haushalt. Es erlaubt Geräten oder Clients über Radiofrequenzen sich mit einem Access Point zu verbinden. Diese nutzen oft Stern- oder Baumtopologien. Durch die Schnelle Geschwindigkeit von 2,4GHz oder 5GHz wird jedoch die Reichweite gekürzt [STA_10].

2.4.3 Zigbee

Zigbee wurde spezifisch dafür entwickelt, wenn man geringe Datenmengen verschicken will und einen geringen Stromverbrauch haben möchte. Ihre geringe Reichweite ist kein Problem, da Zigbee sehr schnell und zuverlässig ein Mesh-Netzwerk aufbauen kann, das heist, dass Zigbee selbstständig Netzwerkpfade baut und sogar bei einzelnen Ausfällen selber einen neuen Netzpfad findet [STA_11].

2.4.4 Bluetooth

Bluetooth ist eine 1990 entwickelte Punkt-zu-Punkt Technologie, die ultrahochfrequente Radiowellen nutzt, um über eine sehr kleine Reichweite Daten zu verschicken.

2.4.5 Vergleiche der Funktechnologien

Bei der Reichweite hat LoRa die längste Reichweite mit bis zu 15 Kilometern, aber mit einer niedrigen Datenübertragung von nur 0,3 kpbs bis 50 kbps, abhängig von der Reichweite. WLAN hat eine Reichweite bis zu 100 m mit Wänden und bis zu 300 m ausserorts, hat dafür aber eine sehr schnelle Datenübertragung, mit neueren Technologien wie Wi-Fi 6 sind mehrere Gbps möglich. Zigbee hat die gleiche Reichweite wie ein WLAN-Netzwerk, hat aber dafür den Vorteil, sehr zuverlässige Mesh-Netzwerke aufzubauen. Wie schon erwähnt besitzt sie eine niedrige Datenübertragung von bis zu 250 kbps, aber dafür ist sie sehr leistungseffizeint. Bluetooth hat eine Reichweite von nur 10 m, besitzt aber eine mittlere und zuverlässige Datenübertragung von 2 Mbps bis 3 Mbps und macht sich dafür sehr nützlich für persönliche zwecke.

2.5 Bildverarbeitung

Stasa Lukic

2.5.1 Bitmaps

Bitmaps speichern Bilder in einem Format, sodass jeder Pixel einzeln Farbwerte zugeschrieben bekommt. Umso mehr Bits man einem Pixel zuweist, umso genauer kann man die Farbe darstellen.
Dies wird Farbtiefe genannt, wird einem Pixel zum Beispiel nur ein Bit zugewiesen, kann der Pixel entweder die Farbe Schwarz (1) oder die Farbe Weiß (0) anzeigen. Heutzutage werden mehrere Formate genutzt, darunter sind die bekanntesten 8 Bit grey-scale, RGB 565, 24-Bit RGB und 32-Bit RGBA.

8-Bit grey scale erlaubt es einem Pixel 256 verschiedene Graustufen anzunehmen.
RGB 565 teilt jeweils Rot 5 Bits, Grün 6 Bits und 5 Bits für Blau für 65536 verschiedene Farbmöglichkeiten, auch genannt High color.
* 24-Bit RGB, auch genannt true color, lässt einen Pixel 16777216 verschiedene Farben annehmen, jedem RGB channel wert wird jeweils ein Byte zugewiesen [STA_14]. Dies wird einmal genauer in Abbildung 2.3 dargestellt.

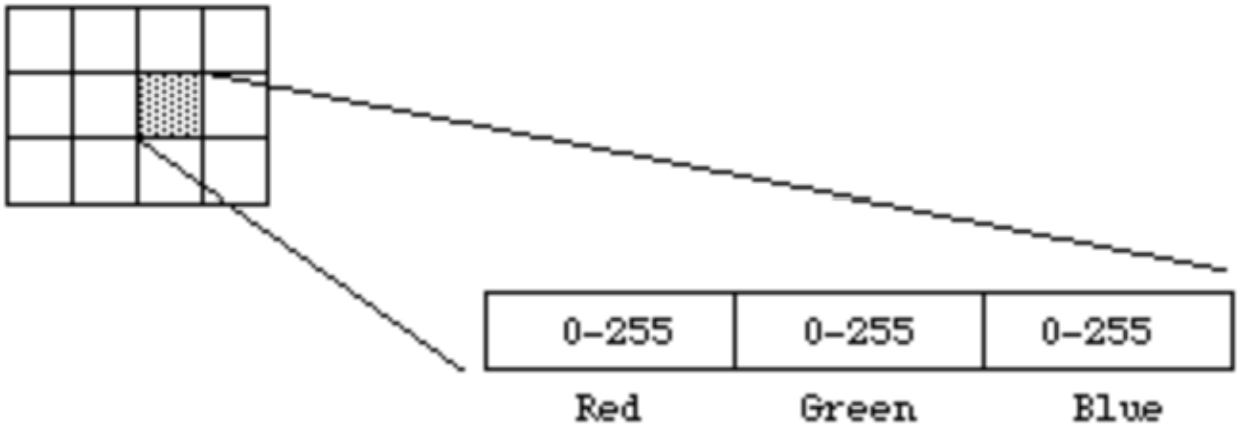


Abbildung 2.3: 24 Bit RGB Pixel

32-Bit RGBA ergänzt noch einen alpha channel. Dieser lässt den Pixel weitere Informationen zur Transparenz der Farbe wissen. Mit RGBA sind 4294967296 verschiedene Farben mit Transparenz möglich.

Diese Informationen werden als Hexadezimahlzahlen gespeichert, es gibt aber auch Indexed Bitmaps. Indexed Bitmaps können einen Farbwert eine Zahl zuweisen, benutzt ein Bild zum Beispiel eine Color depth von 24 Bit aber insgesamt weniger als 256 Farben, Kann man jede Farbe eine Zahl zuweisen. Damit können wir das Bild anstatt mit 24 bit pro Pixel mit 8 Bit pro Pixel darstellen, behält aber die color depth von 24 Bit bei [STA_15]. Ein beispiel kann man in Abbildung 2.4 sehen.

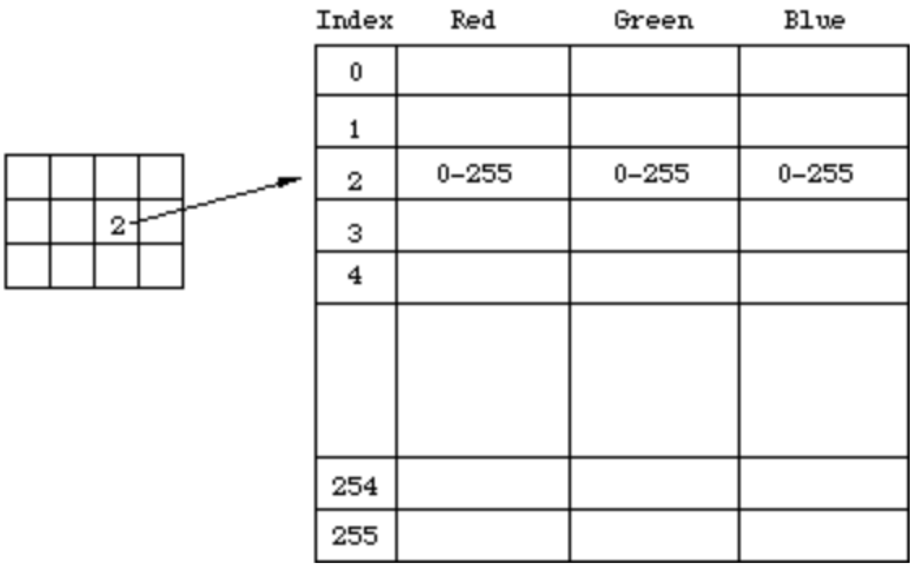


Abbildung 2.4: Indizierte Bitmap

2.5.2 Dithering

Dithering kann dann verwendet werden, um Farbtiefe in einem Bild zu imitieren, wenn wenig Farben in einem Bild verwendet werden. Ein Beispiel, in einer Indiziert Bitmap gibt es nur eine begrenzte Zahl an Farben, dann kann Dithering einen Übergang zwischen zwei Farben erzeugen. Dadurch erscheint das Bild geschmeidiger oder als hätte es einen höhere Farbtiefe.

Dies geschieht durch verschiedene Algorithmen, je nachdem welches Verfahren von Dithering genutzt wird. Es gibt Diffusion Dither, Pattern Dither und Noise Dither. Jedes Verfahren hat eine unterschiedliche Vorangehensweise, wie die Farben übergehen sollen. In Abbildung 2.5 sind Ditherverfahren zu sehen.

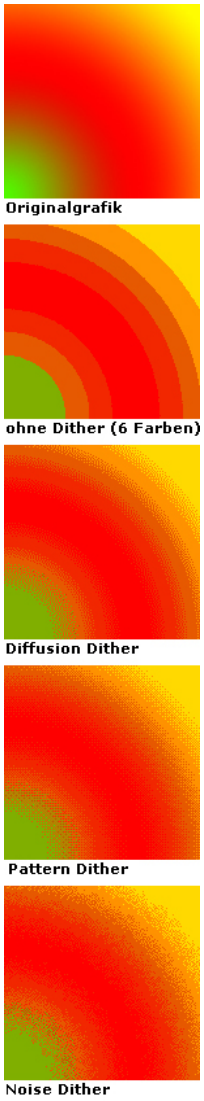


Abbildung 2.5 Verschiedene Ditherverfahren [STA_16]

2.5.3 PNG dekodieren

PNG-Dateien sind Bilder, die komprimiert wurden und dabei keinen Verlust an Bildqualität haben. Die Bilder werden durch eine Substitutionskompressionsmethode und eine Entropiekodierung komprimiert. Die Substitutionskompressionsmethode versucht wiederkehrende Muster im Bild zu erkennen und diese dann Abzuspeichern. Die Entropiekodierung baut Wahrscheihnlichkeitswerte für Farben im Bild auf, PNG kann dann die Farben beim Dekompriemieren nach Wahrscheinlichkeit zurück sortieren.

Um die PNG zu dekodieren, kann man die PNG-Datei als Hex-Format auslesen lassen. Die ersten acht Hexzahlen sind immer die gleichen und signalisieren, dass der folgende Abschnitt an Hexzahlen einer PNG angehören. Diese 8 Hexzahlen werden auch Magische Zahlen genannt, das wären "89 50 4E 47 0D 0A 1A 0A" [STA_17].

Beim Auslesen sind die Hexwerte natürlich noch komprimiert, mit Hilfe von einer Bibliothek wie `zLib` kann man die Farbwerte für jeden Pixel der PNG-Datei auslesen lassen [STA_18].

2.6 Energiemanagement

Mario Wegmann

Computer verbrauchen auch ohne aktive Aufgabe Strom. Ohne spezielle Anweisungen an den Computer wird der Stromverbrauch nicht gedrosselt, wodurch Energie verschwendet wird. Bei PCs fallen hierbei ACPI sleep states ein, ACPI steht dabei für Advanced Configuration and Power Interface und ist ein offener Industriestandard für die Energieverwaltung bei Computern [MW_02]. Die sogenannten ACPI sleep states ermöglichen es dem Betriebssystem zu steuern, welche Komponenten gedrosselt oder gar komplett abgeschaltet werden, um den Stromverbrauch zu minimieren. Auch bei Mikrocontrollern gibt es mehrere Mechanismen, womit der Stromverbrauch reduziert werden kann. Da ein geringer Stromverbrauch eine der wichtigsten Kriterien dieses Projekts ist, werden hier beispielhaft die Sleep States eines ESP32 erläutert.

2.6.1 Einfaches Delay

Die einfachste Möglichkeit, einem ESP32 Mikrocontroller keine konkrete Aufgabe zu geben, ist mit einem Delay. Die Funktion heißt in der Arduino IDE `Delay()` und lässt einen Parameter zu, welcher angibt, wie viele Mikrosekunden der Mikrocontroller warten soll. In der ESP-IDF heißt die Funktion `vTaskDelay()`, hier kann auch ein Parameter für die Wartedauer übergeben werden. Der Parameter nimmt jedoch nicht Millisekunden, sondern Ticks an. Die Ticks sind abhängig von der Tick-Frequenz, daher müssen die Millisekunden mit der Funktion `pdMSTOTICKS()` in Ticks umgewandelt werden. Während dem aktiven Wartens werden jedoch keine Komponenten aktiv abgeschaltet oder gedrosselt, lediglich der Prozessor läuft nicht auf Vollast [MW_06].

2.6.2 Der Light-sleep Modus

Deutlich besser ist da schon der Light-sleep. Wird dieser Modus aktiviert, so werden die teilweise Clock-Signale der digitalen Peripherie, der meisten Teile des RAMs und der CPUs ausgeschalten und die Versorgungsspannung reduziert, dieses Verfahren nennt sich Clock-Gating [MW_03]. Dadurch kann der Stromverbrauch deutlich reduziert werden, wie hoch die Reduktion ist, wird im Kapitel 7.9 Strommessung von Mikrocontroller und Display gemessen. Beim ESP32 gibt es nun verschiedene Wakeup Sources, um den Light-sleep wieder zu beenden. Unter anderem sind folgende Wakeup Sources möglich:

- Timer
- Touchpad
- GPIO
- ULP Coprocessor (Ultra Low Power)
- UART

Nach einem Wakeup Event wird der State wiederhergestellt und der Mikrocontroller arbeitet an der Stelle weiter, wo er durch den Light-sleep aufgehört hat. [MW_06]

2.6.3 Der Deep-sleep Modus

Noch mehr Stromeinsparung kann mit dem Deep-sleep realisiert werden. Dabei werden die oben genannten Komponenten nicht nur clock-gated, sondern komplett abgeschaltet, somit bleibt nur noch der RTC-Controller und dessen Memory und der ULP Coprocessor aktiv. Viele der zuvor genannten Wakeup Sources können auch verwendet werden, um den ESP32 aus dem Deep-sleep heraus aufzuwecken. Ausnahmen sind hierbei jedoch die meisten GPIO Pins und UART. Des Weiteren ist anzumerken, dass nach dem Aufwecken aus dem Deep-sleep der ESP32 kein State wiederhergestellt wird. Die Firmware fängt also von vorne an und auch Daten, die während dem Deep-sleep nicht verloren gehen dürfen, müssen separat in nicht-flüchtigen Speicher abgelegt werden. [MW_06]

2.7 Firmwarebibliotheken

Ahmet Emirhan Göktas

Firmware-Bibliotheken sind wesentliche Werkzeuge für die Entwicklung embedded Systemen. Sie ermöglichen es Entwicklern, vordefinierte Funktionen und Klassen zur Interaktion mit Hardwarekomponenten zu verwenden, anstatt jedes Register und Bit manuell steuern zu müssen. Diese werden als "Hardware Abstraction Layers" oder "HALs" bezeichnet, die typischerweise vom Mikrocontroller-Hersteller oder der Community bereitgestellt und in C oder C++ geschrieben werden.

HALs bieten mehrere Vorteile:

- **Portabilität**: Derselbe Code kann mit minimalen Änderungen auf verschiedenen Mikrocontrollern verwendet werden.
- **Einfachheit**: Der Code ist leichter zu lesen und zu verstehen, da Entwickler sich nicht mit niedrigen Details befassen müssen.
- **Konsistenz**: HALs bieten eine konsistente API, was die Lernkurve beim Wechsel zwischen verschiedenen Mikrocontrollern verringert. [AEG_01]

2.7.1 Arduino

Ahmet Emirhan Göktas

Arduino ist eine beliebte Open-Source-Elektronikplattform, die auf benutzerfreundlicher Hardware und Software basiert. Sie kommt mit einer eigenen integrierten Entwicklungsumgebung (IDE) und einem Satz von Bibliotheken, die das Schreiben und Hochladen von Code auf einen Mikrocontroller erleichtern. Die Projektdateien, bekannt als "Skizzen", haben die Erweiterung `.ino`, was eine von der Arduino-IDE verwaltete C++-Datei ist.

Der größte Vorteil von Arduino ist die große Community und die Vielzahl verfügbarer Bibliotheken. Dies macht es einfach, ein neues Projekt zu starten, Rapid Prototyping durchzuführen und Lösungen für gängige Probleme zu finden, da wahrscheinlich bereits jemand ähnliche Probleme hatte und gelöst hat.

Da Arduino jedoch eine High-Level-Plattform ist, die auf verschiedenen Mikrocontrollern funktioniert, ist es möglicherweise nicht die beste Wahl für leistungskritische Anwendungen. Darüber hinaus fehlt der Arduino-IDE, die für Anfänger entwickelt wurde, Funktionen wie Multi-File-Projekte oder Versionskontrolle, was in größeren Projekten hinderlich sein kann. [AEG_02]

2.7.2 PlatformIO

Ahmet Emirhan Göktas

PlatformIO ist ein vielseitiges Entwicklungsökosystem, das mehr als 1500 eingebettete Boards und über 40 Entwicklungsplattformen unterstützt, darunter Arduino, Espressif IDF, STM32Cube und mehr. Es ist als Plugin für Visual Studio Code, Atom und JetBrains IDEs verfügbar und bietet eine einheitliche Schnittstelle zum Erstellen, Hochladen und Debuggen von Code auf verschiedenen Plattformen.

PlatformIO verfügt über einen Bibliotheksmanager, der es ermöglicht, Bibliotheken aus einem zentralen Repository zu suchen und zu installieren, was die Verwaltung von Abhängigkeiten erleichtert. Es integriert sich auch gut mit gängigen CI/CD-Tools und erleichtert automatisiertes Testen und Deployment. Dies macht PlatformIO zu einer ausgezeichneten Wahl für professionelle Entwicklungsumgebungen, in denen Skalierbarkeit und Wartbarkeit entscheidend sind. [AEG_03]

2.7.3 Espressif IoT Development Framework (ESP-IDF)

Ahmet Emirhan Göktas

ESP-IDF ist das offizielle Entwicklungsframework für die ESP32- und ESP32-S-Serie von SoCs. Es basiert auf FreeRTOS und bietet eine umfassende API-Sammlung zur Konfiguration und Interaktion mit der Hardware. Geschrieben in C und zur Verwendung mit der Xtensa-Toolchain konzipiert, bietet ESP-IDF Bibliotheken für gängige Aufgaben wie Netzwerke, Dateisysteme und Peripheriegeräte. Es ist auch das Framework, das vom ESP32 Arduino Core verwendet wird.

Das ESP-IDF ist gut dokumentiert und hat eine große Community, was es einfach macht, Hilfe zu Problemen bekommen. Sein Hauptvorteil ist die Leistung und der niedrige Zugriff auf die Hardware, was es zu einer guten Wahl für leistungskritische Anwendungen macht. Es hat jedoch eine steilere Lernkurve im Vergleich zu Arduino und PlatformIO. [AEG_04] [AEG_05]

2.8 Gehäuseentwicklung

Jannis Gröger

Um das Gehäuse für ein elektronisches Gerät zu entwickeln, müssen zunächst die Anforderung festgelegt werden. Hierbei kann man in zwischen funktionalen und ästhtischen und ergonomischen Anforderungen unterscheiden. Grundsätzlich soll ein Gehäuse die Komponenten und die Elektronik nicht nur vor äußeren Einflüssen wie Staub, Schmutz oder mechanischen Stößen schützen, sondern trägt auch zur Wärmeableitung bei und spielt eine wichtige Rolle bei der Benutzung des Geräts. Im Folgenden werden die Grundlagen und die essentiellen Überlegungen bei der Entwicklung eines Gehäuses behandelt.

2.8.1 Funktionale Anforderungen

Die Hauptfunktion eines Gehäuses ist der Schutz der einzelnen Bauteile, wobei hier auf verschiedene Faktoren zu achten ist. Zum einen muss das Gehäuse stabil genug sein, um die internen Komponenten vor mechanischen Einflüssen wie Stöße oder Druck zu schützen. Das Material muss dementsprechend passend ausgewählt werden, wobei auch die Geometrie des Gehäuses berücksichtigt werden muss. Häufig verwendete Materialien bei Elektrogeräten sind Kunststoffe oder Metalle wie bspw Aluminium. [JG_01]

Zudem ist das Gehäuse dazu da, die Elektronik darin vor äußeren Einflüssen wie beispielsweise Staub oder Wasser zu bewahren. Bei der Frage, vor welchen sogenannten Umwelteinflüssen ein Gehäuse schützt, hilft die Einteilung in IP-Schutzklassen, wobei das IP für International Protection steht und die verschiedenen Klassen nach der DIN 60529 spezifiziert sind. Die Bezeichnung der Klassen besteht hierbei aus zwei Kennziffern nach der IP-Abkürzung: Die erste Ziffer macht erkenntlich, gegen welche Arten von Berührung und Fremdkörpern das Gehäuse schützt, während die zweite Ziffer den Schutz gegen Wasser benennt. Eine Gehäuse mit IP68 Spezifikation ist beispielsweise vollständig gegen Staubeintritt und vor andauerndem Untertauchen geschützt. Eine solche Zertifizierung kann jedoch nur durch bestimmte Instititionen ausgestellt werden. [JG_02]

Ein weitere Funktion, der bei der Entwicklung des Gehäuses berücksichtigt werden muss, ist die Wärmeregulation, da die Elektronik sowohl vor Überhitzung, als auch vor zu starker Unterkühlung geschützt werden muss. Da elektronische Komponenten selbst Wärme erzeugen, spielt die Wärmeableitung eine größere Rolle, als der Schutz vor zu geringen Temperaturen. Diese kann durch Einsatz von Lüftern oder Kühlkörpern oder Einbau von Lüftungsschlitzen und die damit verbundene Konvektion bewerkstelligt werden. [JG_02]

Eine letzte funktionale Anforderung an das Gehäuse ist die elektromagnetische Verträglichkeit. Darunter versteht man, dass elektronische Geräte so aufgebaut sind, dass sie vor elektromagnetischer Störungen anderer Geräte geschützt sind und ihre eigenen elektromagnetischen Felder auf ein Minimum reduzieren, um selbst nicht andere Geräte zu stören. Dies kann durch die verwendung leitfähiger Materialien und speziellen Dichtungen erreicht werden. [JG_02]

2.8.2 Ästhetische und ergonomische Anforderungen

Neben der Funktionalität stellen auch Ästhetik und Ergonomie gewisse Ansprüche an ein Gehäuse. Bei der Bedienung eines elektronischen Gerätes steht die Benutzerfreundlichkeit im Vordergrund, die bei tragbaren Geräten beispielsweise durch ein geringes Gewicht beeinflusst wird, wohingegen bei stationären Geräten eher auf Zugänglichkeit der Bedienelemente und Anschlüsse geachtet werden sollte. Des Weiteren beeinflusst das Design die Wahrnehmung und das Erscheinungsbild des Geräts, was maßgeblich zur Nutzung und Akzeptanz des Geräts beim Verbraucher beiträgt.

2.8.3 Auswahl des Materials und der Fertigungstechnik

Bei der Auswahl des Materials sind die oben genannten Aspekte von großer Bedeutung, da verschiedene Materialien andere Eigenschaften besitzen und dementsprechend nicht jedes Material die gleichen Anforderungen erfüllt. Abhängig von der Materialauswahl ist im nächsten Schritt dann die Auswahl der Fertigungstechnik, wobei hier zusätzlich auf die Anforderungen an die Geometrie des Gehäuses, die anzufertigende Stückzahl und der Material- sowie der Fertigungspreis geachtet werden muss.

2.8.4 Design und Verfahren des Rapid Prototyping

Sind Material und Fertigungstechnik gewählt, kann mit dem eigentlichen Design des Gehäuses begonnen werden. Heutzutage wird das Design häufig durch CAD-Software (computer-aided design Software) unterstützt, was Gehäuseentwicklern und -

herstellern ermöglicht, ein präzises 3D-Modell des gewünschten Produkts zu Erstellen. Durch Finite-Elemente-Modellierung, kurz FEM, können Simulationen zum Testen der mechanischen oder thermischen Eigenschaften verwendet werden.[JG_03]

Nach dem Design folgt dann die Fertigung eines oder mehrerer Prototypen, die auf gewünschte Eigenschaften getestet werden, um eventuelle Schwachstellen auszumachen und Verbesserungen am Design vornehmen zu können. Hierbei werden immer häufiger Verfahren des Rapid Prototyping, im Folgenden auch RPT genannt, verwendet, was das "Erstellen von Prototypen aus einfach verarbeitbaren Materialien" [JG_03] bezeichnet. Hierbei wird das 3D-Modell aus der CAD-Software in ein "trianguliertes Oberflächenmodell überführt" [JG_03] und anschließend in einzelne Schichten mit einer festgelegten Dicke zerlegt. In unterschiedlichen Verfahren, von denen einige im Folgenden genannt werden, werden diese Schichten dann wieder zusammengefügt, wodurch ein Prototyp entsteht. [JG_03]

- Das Stereolithografie-Verfahren (STL-Verfahren) ist ein RPT-Verfahren, bei dem mit einem Laser in einem Bad aus "polymere[r] Flüssigkeit die Konturen der Scheiben des Modells schichtweise [...] nachgefahren [...] werden"[JG_03], wodurch ein räumliches Modell entsteht.
- Beim Laminated Object Manufacturing Verfahren (LOM-Verfahren) dagegen, fährt der Laser die Konturen auf Papier nach, das die vorher festgelegte Dicke aufweist. Diese Papierschichten werden dann mittels eines "Schmelzkleber[s] aufeinander geklebt" [JG_03]. Das entstehende Modell kann dann ähnlich "wie Holz nachbearbeitet werden" [JG_03].
- Das Fused Deposition Modelling-Verfahren (FDM-Verfahren) erstellt ein Prototyp, indem ein Kunststofffaden durch eine beheizbare Düse auf einer Plattform ausgelegt wird. Diese Plattform lässt sich dabei frei im Raum bewegen und der Durchmesser des Fadens entsricht dabei der vorher festgelegten Schichtdicke. [JG_03]
- Ein weiteres Verfahren ist das Solidier-Verfahren, auch Solid Ground Verfahren oder kurz SGC-Verfahren genannt. Das Modell entsteht hierbei aus mehreren dünnen Fotopolymerschichten, wovon jede einzelne "mit einer fotografischen Maskentechnik unter UV-Licht aushärtet" [JG_03]. Um das fertige Modell herum wird ein Wachskörper gebildet, wodurch Überhänge nicht mit Stützen fixiert werden müssen. [JG_03]
- Auch das Selective Laser Sintering ist ein RPT-Verfahren, bei dem die einzelnen Schichten aus pulverförmigen Material mit einem Laser an gewollten Stellen geschmolzen wird. Durch schichtweises Herunterfahren der Fläche, worauf sich die unterste Pulverschicht befindet, liegt das fertige Modell dann in einem Pulverbett. [JG_03]
- Das 3D-Printing-Verfahren nutzt auch ein Schicht Pulver, wobei hier das Material mithilfe eines Binders verbunden wird. Auch hier wird das Modell an der obersten Schicht aufgebaut und der fertige Prototyp liegt in einem Pulverbett. [JG_03]

Die einzelnen RPT-Verfahren unterscheiden sich in den möglichen Materialien, der Genauigkeit der Modelle, der Qualität der Oberfläche und der verbundene Aufwand für die Nachbearbeitung. Diese Unterschiede sind der Abbildung 2.6 zu entnehmen.

	STL	LOM	FDM	SGC	SLS	3DP
Material	Foto-polymer	Papier, Kunststoff, Metallfolie	Wachs, Kunststoff	Fotopoly-mer	ABS, Wachs, Sintermetalle	Gipspulver, Binder
Genauigkeit [mm]	0,06	0,12	0,13	0,1	0,12	0,2
Oberflächengüte	hoch	eingeschränkt	eingeschränkt	hoch	materialab-hängig	gering
Aufwand für die Nachbearbeitung	gering	sehr hoch	mittel	sehr hoch	hoch	gering

Abbildung 2.6: Eigenschaften der verschiedenen Rapid-Prototyping-Verfahren [JG_03]

Nach dem Prototyping und ausführlichen Testen, kann das überzeugende Modell schließlich in Serienproduktion übergehen womit die Entwicklung des Gehäuses ebenfalls abgeschlossen ist.

2.9 Webanwendungen

Mario Wegmann

Aufgrund der Anforderungen und der Komplexität von größeren Webseiten ist es nicht mehr praktikabel, Inhalte komplett händisch in HTML, CSS und JavaScript zu erstellen. Zu groß und fehleranfällig ist die Wartung eines solchen manuell erstellten Konstrukts. Ein Beispiel hierfür wäre das Erweitern einer neuen Unterseite in der Menüleiste. Hier müsste jede andere Unterseite angepasst werden, um die neue Unterseite von allen anderen Unterseiten aus zu erreichen.

Stattdessen hat es sich etabliert, für Websites mit viel statischem Inhalt sogenannte Static Site Generators zu nutzen. Hierbei liegt der Fokus nur noch auf der Erstellung des eigentlichen Inhalts über eine simple Textdatei. Nach Fertigstellung des Inhalts erzeugt der Generator dann das restliche Gerüst der Website, so werden die Unterseiten generiert, das Navigationsmenü erstellt, Bilder optimiert und das Layout anhand des angegebenen Themes angewendet.

Neben Websites mit statischen Inhalten gibt es auch Websites, deren Inhalt dynamisch erstellt wird. In Social Media ist die Anzahl an Inhalten so enorm, dass es nicht praktikabel wäre, allen Personen die gleiche Timeline anzuzeigen. Stattdessen wird pro Benutzer eine auf ihn spezialisierte Timeline beim Aufruf generiert.

Zuletzt gibt es auch Anwendungen, die über den Webbrowser laufen und somit keine lokale Installation benötigen, ein Beispiel wäre hierfür die Nextcloud, womit sich Dateien abspeichern, erstellen, ordnen und teilen lassen. Dank vieler Erweiterungen.

Die letzten beiden genannten Kategorien erfordern viele Komponenten, die zusammenarbeiten müssen, um dem Benutzer eine performante und benutzerfreundliche Erfahrung zu ermöglichen. Daher haben sich mehr und mehr verschiedene Technologien im Web entwickelt, die mit unterschiedlichen Herangehensweisen versuchen, die Anforderungen zu erfüllen.

Dennoch gibt es auch noch Anwendungsfälle, wo es durchaus sinnvoll ist, bewusst auf weitere Technologien zu verzichten und nur mit HTML, CSS und JavaScript zu arbeiten. Ein Anwendungsfall wäre der Embedded Bereich, da dort die Ressourcen sehr knapp sind und somit die Last für Anfragen möglichst auf den zugreifenden Client ausgelagert werden sollten.

2.9.1 Webtechnologien

Mario Wegmann

Moderne Webbrowser haben sich auf die drei Programmiersprachen HTML, CSS und JavaScript geeinigt, um Webseiten darzustellen.

HyperText Markup Language (HTML)

HTML definiert die Struktur und den Inhalt einer Webseite durch die Verwendung von HTML-Tags. HTML bildet das Gerüst einer Webseite, indem es Text, Bilder, Links, Videos und andere Elemente einbindet und organisiert.

Cascading Style Sheets (CSS)

CSS ist eine Stylesheet-Sprache, die verwendet wird, um das Aussehen und Layout von HTML-Dokumenten zu gestalten. Mit CSS können Entwickler die visuellen Aspekte von HTML-Elementen steuern. CSS ermöglicht eine Trennung von Struktur und Design, was die Wartung und Anpassung von Webseiten erleichtert.

JavaScript

JavaScript ist eine Programmiersprache, welche im Browser ausgeführt werden kann und somit für Interaktivität auf der Clientseite sorgt. JavaScript nutzt die Technik der DOM-Manipulation, um den Inhalt des geladenen HTML nachträglich zu modifizieren. Das Document Object Model (DOM) beschreibt den Aufbau der einzelnen HTML-Elemente als Baumstruktur, diese Elemente können mit JavaScript gelesen, hinzugefügt, verändert und gelöscht werden. TypeScript ist eine Erweiterung von JavaScript und erweitert die Sprache um statische Typen, damit können Fehler im Programmcode früher erkannt und die Codequalität verbessert werden.

React

React ist eine JavaScript-Bibliothek zur Erstellung von Benutzeroberflächen. Sie verwendet eine komponentenbasierte Architektur, die es Entwicklern ermöglicht, wiederverwendbare UI-Komponenten zu erstellen und den Status von Anwendungen

effizient zu verwalten. React nutzt einen virtuellen DOM zur Optimierung von Updates und zur Verbesserung der Performance. Neben React sind Vue, Angular und Svelte weitere bekannte Frontend-Bibliotheken [MW_12].

Structured Query Language (SQL)

SQL ist eine standardisierte Programmiersprache, die zur Verwaltung und Manipulation von Daten in relationalen Datenbanken verwendet wird. Mit SQL können Benutzer Datenbanken erstellen, ändern, abfragen und verwalten [MW_13].

Object Relational Mapper (ORM)

ORMs sind nützlich, weil sie die Arbeit mit Datenbanken durch typsichere, deklarative Datenmodellierung und effiziente Datenbankmigrationen erheblich vereinfachen und optimieren. Prisma ORM ist beispielsweise ein ORM, welches automatisch TypeScript-Typen generiert. Dies ermöglicht Typensicherheit und die Nutzung von Autovervollständigung im Editor, wodurch die Produktivität und Codequalität verbessert wird. ORMs reduzieren die Notwendigkeit für manuelles SQL-Schreiben und minimieren somit potenzielle Fehler. Zudem unterstützen sie verschiedene Datenbanksysteme, somit ist ein Wechsel des RDBMS mit nur wenig Änderungen am Code möglich [MW_14].

PostgreSQL

PostgreSQL ist ein relationale Datenbankmanagementsystem (RDBMS). Ein RDBMS ist eine Software, die zur Verwaltung von Datenbanken verwendet wird, die auf dem relationalen Modell basieren. In diesem Modell werden Daten in Tabellen organisiert, die aus Zeilen und Spalten bestehen. Jede Tabelle repräsentiert eine Entität, und die Beziehungen zwischen den Tabellen werden durch Primär- und Fremdschlüssel definiert. PostgreSQL ist bekannt für seine Erweiterbarkeit, Standardkonformität und fortgeschrittene Features wie komplexe Abfragen und Transaktionen. MySQL ist ein weiteres bekanntes und verbreitetes RDBMS [MW_15].

Webserver

Es gibt verschiedene Möglichkeiten, wie HTML, CSS und JavaScript bei einer Anfrage an den Webserver generiert werden können. Der Code kann komplett statisch auf dem Webserver abgelegt sein oder auch dynamisch bei der Anfrage generiert werden. Prinzipiell lässt sich für den letzten Fall fast jede Programmiersprache verwenden, beliebt sind jedoch PHP, Python, Ruby, C# und JavaScript (Node.JS) [MW_04].

Für die Umsetzung werden häufig Frameworks genutzt, ähnlich wie bei Static Site Generators, unterstützen Frameworks bei der Realisierung von Websites, indem es wiederkehrende Aufgaben vereinfacht und die Wiederverwendung von Code fördert. Durch die Abstraktion und Strukturierung von Code sowie die Objektorientierung der Daten wird die Entwicklungszeit verkürzt und die Wartbarkeit großer Anwendungen erheblich verbessert. NextCloud verwendet beispielsweise das PHP Framework Symfony [MW_05]

Reverse Proxy

Ein Reverse Proxy ermöglicht das Anbieten von mehreren Webanwendungen auf demselben Server. Typischerweise nutzen Browser für HTTP-Anfragen den Port 80 und für HTTPS den Port 443. Da mehrere Webanwendungen auf dem gleichen System die Ports nicht mehrfach verwenden können, müssten andere Ports verwendet werden, was aber unpraktikabel für den Benutzer der Webanwendung ist. Der Einsatz eines Reverse Proxies bietet hier mehr Flexibilität. Dieser hört als einziger auf die beiden Ports und nimmt die Anfragen entgegen, anhand verschiedener Eigenschaften, wie z.B. Domainname im HTTP-Header, kann dann die Entscheidung getroffen werden, an welchen Webserverdienst die Anfrage weitergeleitet wird. Reverse Proxies bieten hierbei oft noch weitere Funktionen an, wie die Lastverteilung auf mehreren Dienste, das Verschlüsseln der Verbindung mit HTTPS, oder auch das Schützen der Webanwendung durch nur authentifizierten Zugriff über HTTP Basic Auth.

3. Marktanalyse aktueller Trends und Technologien

Julia Reuter

Die Motivation des Projektteams lag bei erster Betrachtung des Projektes Low-Power Raumanzeige für die THA hauptsächlich in der Steigerung der Effizienz und des eigenen Komforts. Das Projekt wurde also eher als Nischenprodukt für den internen Gebrauch angesehen. Diese Einstellung änderte sich jedoch schnell nach einiger Recherche. Nicht nur ist der potenzielle Markt für eine Low-Power Raumanzeige aufgrund der vielfältigen Einsatzgebiete in unter anderem Büroräumen, Krankenhäusern oder Bildungseinrichtungen relativ groß. Sondern auch im Hinblick auf das Thema Umwelt, Nachhaltigkeit und Digitalisierung stellt die Raumanzeige eine attraktive Lösung dar. So wird beispielsweise eine effiziente Nutzung von Räumen durch klare Erkennbarkeit der Belegung ermöglicht. Durch intelligente Verwaltungsmechanismen könnten sogar unnötige Heiz- und Beleuchtungskosten reduziert werden, wenn der Raum gerade nicht genutzt wird. Darüber hinaus kann auf papierbasierte Stundenpläne und Türschilder verzichtet werden. Der damit verbundene personelle Koordinationsaufwand zur Instandhaltung wird deutlich verringert und Ressourcen wie Papier und Druckertinte werden eingespart.

All dieses Aspekte sind Merkmale für ein durchaus zukunfttaugliches Produkt und gerade deshalb gibt es bereits einige Unternehmen, die dieses Potenzial erkannt und entsprechende Produkte entwickelt haben. Im Folgenden werden nun vier hauptsächlich deutsche Unternehmen analysiert, welche als Inspiration für die eigene Raumanzeige dienen und durch ihr breites Spektrum dem Team zu neuen Ansätzen und Ideen verhelfen.

3.1 Übersicht einiger Konkurrenzprodukte

Bei den ausgewählten Unternehmen und Produkten handelt es sich um:

- **Wizepanel** der Firma Wilke Technology in Aachen [JR_01]
- **Digitales Türschild** der Firma digitalSIGNAGE in Hamburg [JR_02]
- **ROOMZ Display** der Firma ROOMZ in Freiburg (Schweiz) [JR_03]
- **Touchscreen-Display** der Firma Beetronics mit Sitz in Düsseldorf [JR_04]

Im Hinblick auf das eigene Projekt lag der Fokus der Recherche auf folgenden Aspekten, welche im Anschluss tabellarisch (s. Tabelle 4.2) aufgelistet sind:

- Display Art (LCD oder E-Paper)
- Farbunterstützung
- Ansteuerung/Netzprotokoll
- Batterie und geschätzte Laufzeit
- Betrieb (Schnittstelle zum Kunden)
- Besondere Features

3.2 Produktvergleich ausgewählter Unternehmen

Kategorie	Wizepanel [JR_05]	Digitales Türschild [JR_06]	ROOMZ [JR_03]	Beetronics [JR_07]
Display Art	E-Paper	Touch-Screen (kein E-Paper)	E-Paper	High Brightness Full HD Touchscreen
Farbunterstützung	Schwarz-Weiß, 16 Graustufen	Farbe	Schwarz-Weiß	Farbe
Ansteuerung/ Netzprotokoll	868 MHz Ultra Low Power Funk, eigene Funksender	Ethernet, NFC-Option verfügbar	WLAN	Display Port, HDMI, VGA, USB-C, Unterstützung aller gängigen Betriebssysteme
Batterie und geschätzte Laufzeit	10 x AA-Lithium-Primärzellen, bis zu 20 Jahre	Power over Ethernet (PoE) → Kabelanschluss nötig	8000mAh, 4 Jahre	Netzteil (d.h. Stromanschluss) denn braucht 12.7W (Betrieb)/ 0.4W (Standby)
Betrieb (Kundenschnittstelle)	Zentrale Serverkomponente: Inhaltsübertragung und Geräteverwaltung	Keine Angabe	MyRoomz-App zur Buchung einzelner definierter Arbeitsbereiche [JR_10]	Einfaches Plug-and-Play, sonst eigener Treiber für verschiedene Betriebssysteme verfügbar
Besondere Features	Bereits in den Server integrierte Adapter für MS Exchange, iCal/ ICS, Google Calendars, Excel,... Auch WebUntis wird unterstützt [JR_08]	LEDs rund um Display (rot, gelb, grün) zur Hervorhebung der Raumbelegung, Sofortbuchung über Touch/NFC	MyRoomz-App bietet Lageplanansicht des Gebäudes mit Übersicht aller Räume mit Markierung ob sie belegt sind oder nicht [JR_10], ROOMZ Sensor zur Vermeidung von Ghost-Meetings [JR_09], Sofortbuchung über Touchleiste am Gehäuse	Keine spezifischen Features angegeben, aber allgemein sehr vielfältig universell einsetzbar

Tabelle 3.1: Übersicht und Vergleich der 4 ausgewählten Raumanzeigen

3.3 Auswertung der Rechercheergebnisse

3.3.1 Vor- und Nachteile der einzelnen Produkte

Hersteller	Vorteile	Nachteile
Wizepanel	• E-Paper-Technologie: Geringer Energieverbrauch • Lange Akkulaufzeit: Bis zu 20 Jahre • Ultra Low Power Funk: Energiesparende und drahtlose Kommunikation • Serverintegration: Unterstützung für gängige Kalender- und Verwaltungssysteme	• Schwarz-weiß Display: Keine Farbdarstellung möglich • Eigene Funksender zur Datenübertragung werden benötigt: Abhängigkeit von zusätzlicher Hardware
digitalSIGNAGE	• Touch-Screen und Farbe: Interaktive und ansprechende Benutzeroberfläche • NFC-Option und Ethernet: Flexibilität bei der Ansteuerung • LEDs zur Raumbelegung: Visuelle Hinweise zur Raumverfügbarkeit	• Power over Ethernet (PoE): Benötigt eine Kabelverbindung, externe Abhängigkeit • Keine Angaben zur Akku-Laufzeit: Unklarheit über Energieeffizienz, vermutlich aber eher gering aufgrund der Displaytechnologie
ROOMZ	• E-Paper-Technologie: Geringer Energieverbrauch • WLAN: Einfache Integration in bestehende Netzwerke • Lange Akkulaufzeit: Bis zu 4 Jahre • MyRoomz-App: Umfassende Funktionen zur Raumbuchung und -verwaltung, auch über mobile Endgeräte	• Schwarz-weiß Display: Keine Farbdarstellung möglich
Beetronics	• High Brightness Full HD Touchscreen: Hervorragende Bildqualität und interaktive Möglichkeiten • Vielseitige Anschlussmöglichkeiten: Unterstützung für Display Port, HDMI, VGA, USB-C; Flexibiliät in der Ansteuerung	• Verhältnismäßig sehr hoher Energieverbrauch • Kabelgebunden: Abhängigkeit von ständiger Stromquelle

Tabelle 3.2: Vor-und Nachteile der vier Displayprodukte

3.3.2 Abgeleitete Kriterien für das eigene Produkt

Basierend auf der Analyse der Vor- und Nachteile der Konkurrenzprodukte (vgl. Tabelle 3.1) sollten folgende Eigenschaften in die eigene Low-Power Raumanzeige integriert werden, um auf dem aktuellen Markt konkurrenzfähig zu sein:

1. **E-Paper-Technologie:** Für minimalen Energieverbrauch und lange Akkulaufzeit
2. **Lange Akkulaufzeit:** Nutzung von effizienten Batterien/Akkus für eine wartungsarme Lösung und unabhängigen Betrieb
3. **Drahtlose Kommunikation:** WLAN oder Funk für flexible und einfache Installation sowie geringem Energieverbrauch
4. **Integration in bestehende Systeme:** Unterstützung für gängige Kalender- und Verwaltungssysteme (z.B. iCal/ICS oder WebUntis) über zentrale Serverkomponenten zur effizienten Verwaltung und Nutzung

3.3.3 Ansätze zur Verbesserung der Konkurrenzfähigkeit auf dem Markt

Um sich von den zahlreichen Konkurrenzprodukten abzuheben, gibt es einige Ansätze, welche nach Möglichkeit in das System integriert werden.

- **Interaktivität:** Auch wenn es sich um ein E-Paper Display handelt, besteht die Möglichkeit eine gewisse Benutzerinteraktion für beispielsweise spontane Raumbuchungen zu integrieren. Eine Option, die energiesparend und auch ohne Touchfunktion umsetzbar ist, ist das Integrieren von Knöpfen am Gehäuserand. Per Knopfdruck könnte so das Anzeigebild oder die Ansicht gewechselt werden.
- **Farbunterstützung:** Ein E-Paper-Display mit Farbunterstützung trägt dazu bei, das eher schlichtes Produkt für den Kunden optisch ansprechender zu gestalten und sich dabei von der Masse abzuheben. Auch ein E-Paper-Display mit Farbunterstützung würde dazu beitragen das eher schlichtes Produkt für den Kunden optisch ansprechender zu gestalten und sich von der Masse abzuheben.
- **Flexibilität:** Das zentrale Alleinstellungsmerkmal ist das Entwickeln eines möglichst generischen Konzeptes, das sehr flexibel einsetzbar ist. Denn eine Einschränkung der obigen Systeme ist, dass alle zentral über eine Serverkomponente gesteuert werden und teilweise sogar eigene Funkverteiler brauchen (vgl. Wizepanel). Dies ist für den Benutzer mit viel Aufwand und Kosten verbunden, um die eigenen Infrastruktur so zu erweitern, damit Display-Module intergriert werden können. Diese Maßnahmen lohnen sich nur für sehr große Organisationen mit ausreichend Budget. Die Zielgruppe ist also entsprechend gering. Zudem ist man stets von einer externen Softwarekomponente abhängig. Die Display Ansteuerungsschnittstelle wird ebenfalls vom Hersteller in Form von einem Server (vgl. Wizepanel) oder einer App (vgl. ROOMZ) geliefert. So ist das System bei Ausfall oder Fehlverhalten nicht mehr einsatzfähig, bis das Problem durch eine externe Partie, sprich dem Hersteller, behoben wurde.

Ziel ist es also ein Konzept umzusetzten, indem die Benutzergruppe möglichst breit ist und externe Abhängigkeiten so weit wie möglich minimiert werden. Die eigene Raumanzeige soll also nicht zwingend eine zusätzliche externe Infrastruktur benötigen. Sprich es wird eine Option für eine Standalone-Funktionalität geben, sodass das Low-Power Display völlig unabhängig von jeglicher zusätzlicher Soft- und Hardware ist und autark funkionieren kann. Gleichzeitig besteht weiterhin die Möglichkeit eine zentrale Verwaltungskomponente zu integrieren, um auch mehrere Displays verwalten zu können. Diese Flexibilität gekoppelt mit der Prämisse, das System Open-Source zu gestalten, sorgt für eine hohe Konkurrenzfähigkeit auf dem Markt. Außerdem wird versucht, ein möglichst großes Potenzial für Erweiterbarkeit zu schaffen, sodass beispielsweise ein externer Raumsensor (vgl. ROOMZ Sensor) eingebunden werden kann. Dem Kunden werden somit alle Mittel an die Hand gegeben, um das System nach Belieben anzupassen und zu erweitern.

Die Low-Power Raumanzeige ist somit nicht nur für die Hochschule attraktiv, sondern kann auch in kleinern Kreisen bis hin zum privaten Gebrauch eingesetzt werden.

4. Teamorganisation

4.1 Kommunikation und Organisation

Stasa Lukic

4.1.1 Einleitung

Für das Projekt wurden verschiedenste Programme genutzt, um eine gute Kommunikation und Organisation innerhalb der Teammitglieder zu halten. Für eine effektive und effiziente Kommunikation und Organisation wurde für jeden Bereich ein Programm verwendet, die folgend angesprochen wird.

4.1.2 Notion

Für Notizen, Protokolle, Timelines und Dokumentierungen benutzen wir Notion. Dieses Programm erlaubt es uns schnell und einfach verschiedenste Sachen aufzuschreiben, egal ob es sich um eine Tabelle, Aufgaben, Timeline oder anderes handelt. Der Vorteil von Notion ist, dass man einen Workspace erschafft, den man sich wie ein großes Dokument vorstellen kann, aber mehrere Leute können gleichzeitig auf dieses Dokument zugreifen. Dies nutzen wir, um eine Timeline mit wichtigen Terminen oder ein Gantt Chart für das Projekt aufzubauen, wo wir sehen können, wie wir im Zeitplan stehen, sind wir einem Arbeitspunkt der Zeit voraus oder zu spät. Updates können wir uns aufschreiben, um uns gegenseitig den Stand der Dinge zu kommunizieren ohne auf eine Antwort warten zu müssen.

4.1.3 Nextcloud

Nextcloud ist die Hochschule eigene kostenlose Cloud, diese nutzen wir, um wichtige Dokumente zu sichern und für später einfach zugänglich zu machen.

4.1.4 Zoom, Discord und Whatsapp

Zoom, Discord und WhatsApp waren unsere Programme für eine direkte Kommunikation. Über WhatsApp kann man schnell ein Projektmitglied etwas fragen, Termine vereinbaren oder anderes. Zoom und Discord wurden für wöchentliche Meetings genutzt, aber auch für längere Arbeitssessions. Der Vorteil hierbei liegt, dass man von verschiedensten Orten eine direkte Kommunikation aufbauen kann und so gleichzeitig miteinander weiter arbeiten kann.

4.1.5 Github

Um unseren Programmcode miteinander zuteilen, nutzen wir Github. Github lässt mehrere Personen an einem Programmcode schreiben, mithilfe von sehr einfachen und schlauen Verfahren. Gedownloadete Projekte von Github können direkt bearbeitet werden. Branches lassen mehrere Personen an einem Projekt arbeiten. Wir nutzen einen Dev Branch von Main wo wir persönliche Feature-Branche gemerged haben, sobald sie geschrieben und getestet wurden. Sobald das jedes Feature fertig ist und alles auf dem Dev Branch gemerged ist, kann man den Dev Branch auf Main mergen und hat somit einen fertigen Code.

Commits sind Codeupdates, die man auf die Github "Cloud" hochlädt. Sie erlauben die Möglichkeit ältere Commits anzusehen und mit dem neu geschrieben Code zu vergleichen, dies hat den Vorteil, dass wenn ein Fehler im Code nicht gefunden werden kann, man auch den Stand des älteren Commits wiederzuverwenden. Issues erleichtert die Organisation von Fehler im Code. Github besitzt ein System, das es flexibel erlaubt sogenannte Issues zu eröffnen. Diese Issues werden dann genutzt, um Arbeitspakete zu verfolgen, Probleme anzusprechen oder Feedback zu erhalten.

4.1.6 MkDocs

Für unsere technische Abgabe nutzen wir MkDocs in Verbindung mit Github. MkDocs ist ein benutzerfreundliches Programm, das mithilfe von Markdown gut strukturierte Dokumentationen erstellt. Die MkDocs Dokumentation kann einfach konfiguriert werden, indem man in der Konfigurationsdatei mkdocs.yml gewollte Einstellungen reinschreibt. In dieser Datei können auch

Plugins hinzugefügt werden, einer dieser Plugins ist "with-pdf", die es uns erlaubt, aus der Dokumentation eine PDF zu exportieren.

4.2 Rollenaufteilung

Stasa Lukic

Unsere Teammitglieder wurden jeweils in Teams aufgeteilt, in diesen Teams haben sie die zugehörigen Aufgaben dann bearbeitet. Wir haben von unseren 7 Mitgliedern (später 6) 2 Personen ins Softwareteam eingetragen, 2 Personen ins Hardwareteam und 3 Personen ins Firmwareteam. Jedes Team hat auch einen Teamleader ausgewählt, diese Person kann in Meeting Updates geben und fürs Team sprechen. Die sorgt für eine sehr effiziente und organisierte Kommunikation zwischen den Teams. Teamleader können diese Rolle auch nutzen, um klare Ziele zu setzen. Rollenaufteilung innerhalb der Teams macht auch dann Sinn, um Stärken einzelner Teammitglieder, in einem Themenbereich zu nutzen. So erhält man nicht nur bessere Resultate, die Person selbst kann ihre Stärken noch einmal feinschleifen. Arbeitspackete können fair auf ein Team verteilt werden, um eine unfaire Arbeitsaufteilung zu verhindern.

4.3 Zeitplan

Stasa Lukic

Zeitpläne können über verschiedene Arten erstellt werden, es gibt die Gantt-Diagramme, Pfad-Diagramme, Meilensteinziele und weiteres. Wir haben uns für die Gantt-Diagramme entschieden. Gantt-Diagramme erstellen Balken auf einer Timeline mit einem Start- und Enddatum. Diese Balkan können Aufgaben zugewiesen bekommen oder sub Aufgaben / Balken. Zur Organisationen müssen die Balken klar gekennzeichnet werden und einem Plan folgen, sie müssen aufeinander aufbauen. Dies können wir in der Abbildung 4.1 sehen.

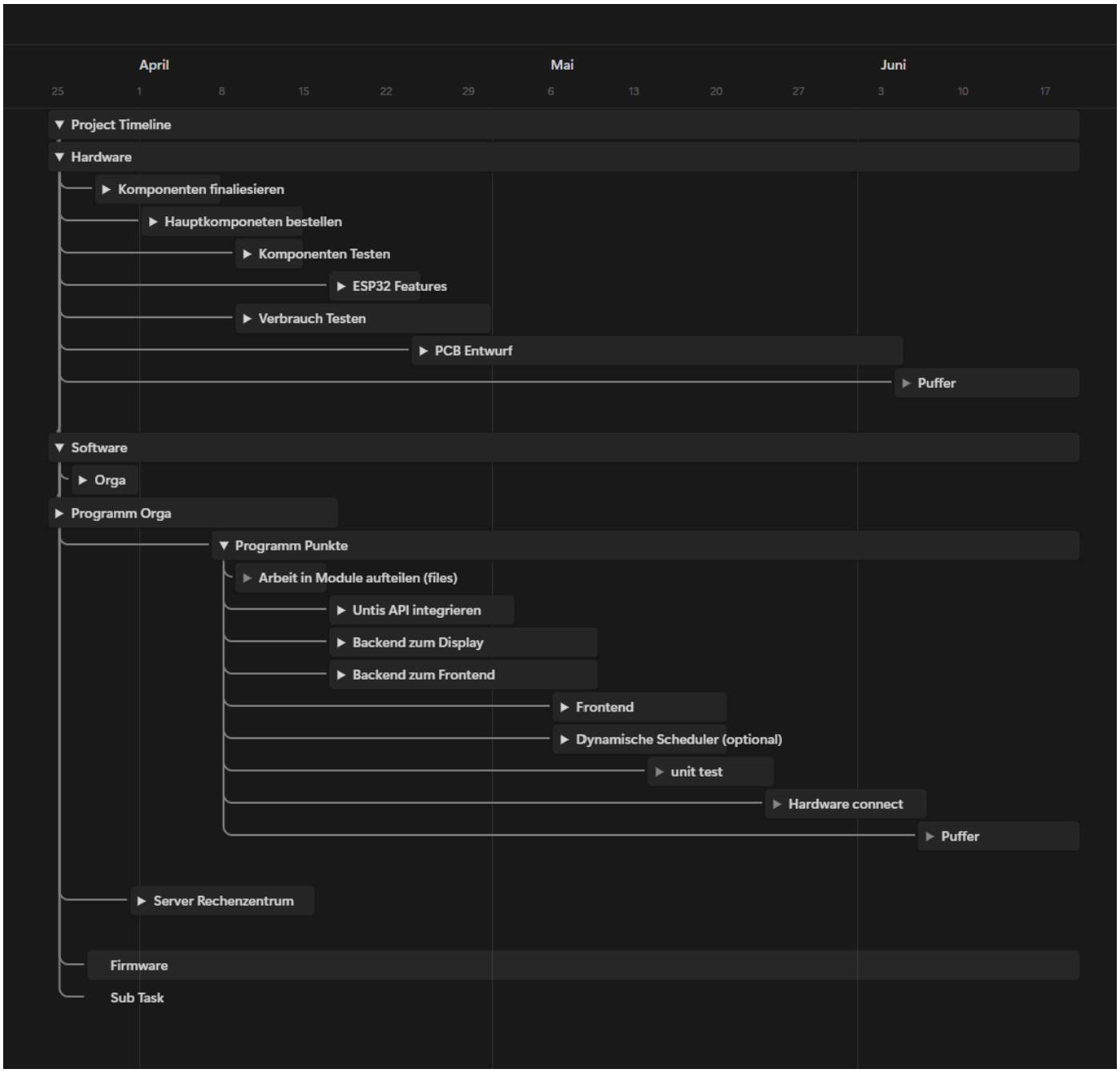


Abbildung 4.1: Timeline Beispiel Projekt

Diese Timeline kennzeichnet klare Ziele und visualisiert den Plan von Projektanfang zu Ende. Natürlich ist es schwer genau zusagen, wie lange eine Aufgabe dauern wird zu erledigen, Probleme können auch nicht vorausgesagt werden, dadurch bauen wir am Ende einen Zeitpuffer ein. Dauert eine Aufgabe länger als angegeben, können wir diese Zeit vom Puffer "klauen". Durch die Visualisierung und Struktur können Projektmitglieder ihre Zeit besser einplanen und haben Eigenverantwortung, diese Deadlines zu erreichen, dadurch haben wir einen durchgehenden Fortschritt am Projekt.

5. Systemkonzept

Julia Reuter

Basierend auf den Rechercheergebnissen aus Kapitel 3 hat das Team folgendes Systemkonzept für die Low-Power Raumanzeige entwickelt.

5.1 Grundaufbau der Hardware

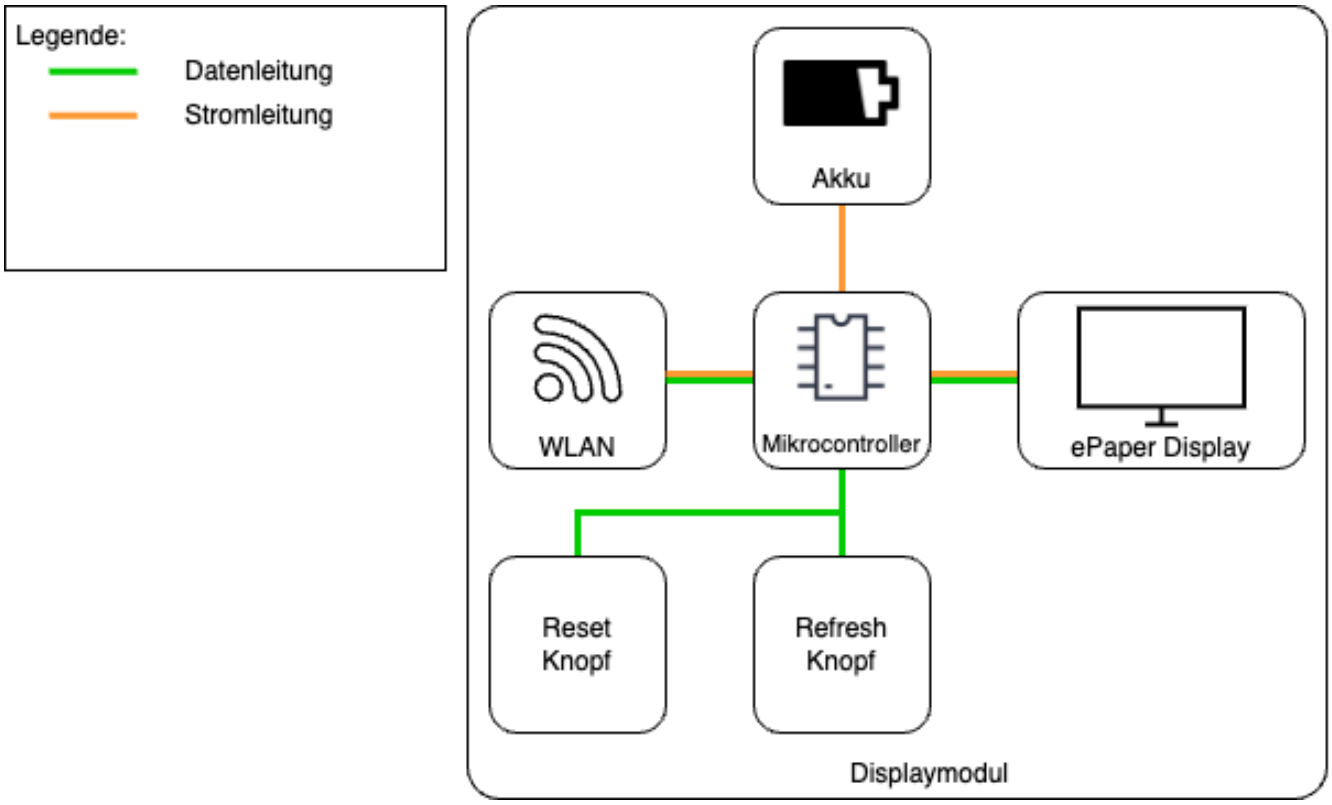


Abbildung 5.1 Visualisierung des Grundkonzepts des Systems

Wie in Abbildung 5.1 veranschaulicht, besteht das Low-Power Display Modul aus:

- Einem Akku, der die Stromversorgung des gesamten Systems gewährleistet
- Einem Mikrocontroller, der die zentrale Steuereinheit bildet und die anzuzeigenden Informationen über WLAN empfängt, entsprechend aufbereitet und weitergibt
- Einem E-Paper Display, das über eine geeignete Schnittstelle die Daten von dem Mikrocontroller erhält und im Anschluss direkt anzeigen kann

Angedacht sind zudem zwei Knöpfe, die Benutzerinteraktionen ermöglichen. Zum Beispiel kann das System nach einem Fehler neu gestartet, oder sogar wieder auf Werkseinstellungen zurückgesetzt werden. Auch das angezeigte Bild könnte einfach per Knopfdruck geändert werden.

Verbaut werden die einzelnen Komponenten in einem kompakten 3D-gedruckten Gehäuse, welches sich einfach und ohne externe Kabel montieren lässt.

5.2 Allgemeine Software-Funktionen

Da die gesamte Ansteuerung des Systems über einen Mikrocontroller erfolgt, muss dieser entsprechend programmiert werden. Er bildet die zentrale Steuereinheit, die sich mit entsprechender Firmware um das Empfangen und Senden von Daten, die Displayansteuerung und das Powermanagement kümmert. Durch ihn soll es dem Benutzer ermöglicht werden, über eine einfache und intuitive Schnittstelle, wie beispielsweise eine Website, das Display mit Inhalten zu versorgen. So können ohne viel Aufwand Bilddateien auf der Website hochgeladen und mit nur mit einem Klick über eine WLAN-Verbindung an das Modul geschickt werden. Auch die Displayverwaltung und diverse Einstellungen werden über diese Art von Interface bequem gesteuert.

5.3 Potentielle Einsatzgebiete der Low-Power Raumanzeige

In der Projektdefinition ist die Hochschule als konkreter Kunde genannt, weshalb der Fokus vor allem auf diesen spezifischen Anwendungsfall gerichtet ist. Hier besteht die Hauptidee darin, die Raumverfügbarkeit in Echtzeit anzuzeigen und über das Stattfinden von Veranstaltungen, Vorlesungen und Seminaren zu informieren.

Dies bedeutet, dass Studenten schnell und unkompliziert herausfinden können, welche Räume gerade frei sind, um ungestört zu lernen oder Gruppenarbeiten durchzuführen. Dozenten können sicherstellen, dass ihre geplanten Veranstaltungen in den vorgesehenen Räumen stattfinden und bei Bedarf kurzfristige Änderungen kommunizieren. Auch das Verwaltungspersonal profitiert von einer effizienten Raumverwaltung, die durch das Anzeigen zusätzlicher Informationen, wie beispielsweise das Melden defekter Beamer, die Wartung der Räumlichkeiten verbessert.

Um ein solches Netz an Displays an der Hochschule reibungslos betreiben zu können, ist, nach Vorbild der Konkurrenz (vgl. Kapitel 3), die zentrale Verwaltung über eine Serverkomponente sinnvoll. Es bietet sich hierbei an, je nach Standort bzw. Raum, IDs zu vergeben, sodass alle Inhalte korrekt zugeordnet werden können. Mithilfe einer Schnittstelle zu gängigen Kalendersystemen wie ICS oder WebUntis wird nicht nur über die Raumverfügbarkeit informiert, sondern auch der gesamte Tagesstundenplan der jeweiligen Hörsäle ist sofort ersichtlich.

Das Einsatzgebiet der Low-Power Raumanzeige ist jedoch nicht nur auf den Hochschulbereich beschränkt. Vor allem für den mobilen Einsatz oder für Privatpersonen ist das Einrichten und Verwalten eines Servers unpraktisch und zeitaufwändig. Deshalb ist das System so flexibel und generisch gehalten, dass es die Bedürfnisse unterschiedlicher Benutzergruppen erfüllt.

So kann es auch von kleineren Firmen zur effizienten Verwaltung von einzelnen Besprechungsräumen oder Shared-Desk-Systemen eingesetzt werden. Auch auf Messeständen oder in Tagungsräumen ist eine universelle Low-Power Anzeige denkbar. In diesen Anwendungsfällen ist häufig keine geeignete Infrastruktur vorhanden, um ein komplexes System zu integrieren. Deshalb ist es möglich, das Display nicht nur in einem "Server-Modus" zu betreiben, sondern auch in den zwei weiteren Betriebsmodi "Standalone" und "Netzwerk".

Im Standalone-Modus kann das Display am flexibelsten eingesetzt werden, da keinerlei Grundaustattung am Montageort nötig ist. Der im System integrierte Mikrocontroller öffnet einen WLAN Access-Point, sodass sich ein Client-Gerät, wie beispielsweise das eigene Handy, mit der Raumanzeige verbindet. Der Benutzer kanndarufhin direkt über sein Gerät den gewünschten Inhalt an das Display übermitteln. Dieser Modus erweitert das Low-Power Raumdisplay zu einer völlig autarken Anzeigetechnologie.

Im Netzwerkmodus ist das Konzept ähnlich, mit dem Unterschied, dass sich sowohl das Displaymodul als auch das Client-Gerät im gleichen WLAN, wie zum Beispiel dem eigenen Heimnetzwerk, befinden und kommunizieren.

5.4 Die drei Betriebsmodi der Raumanzeige im Vergleich

5.4.1 Standalone-Modus

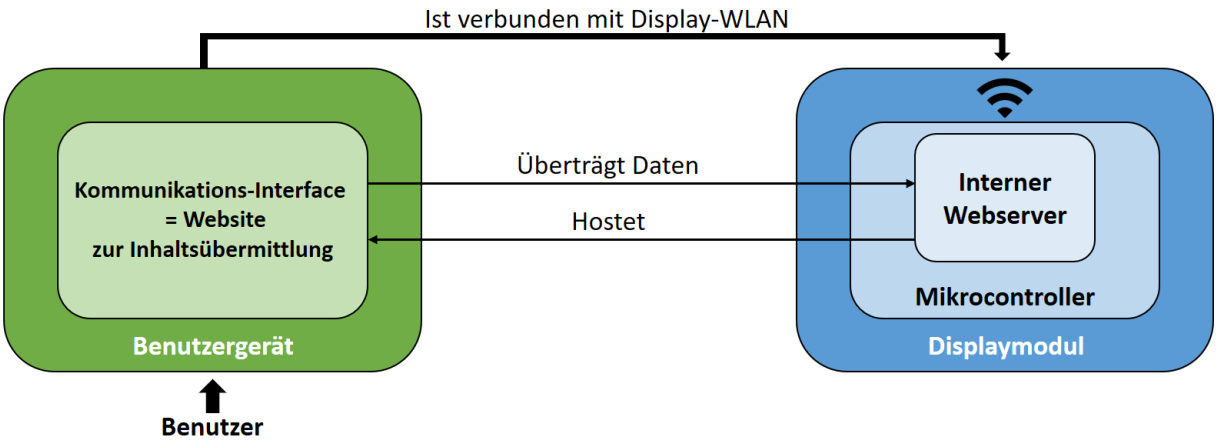


Abbildung 5.2: Standalone-Modus

Modus	Vorteile	Nachteile
Standalone	• Keine zusätzliche Infrastruktur erforderlich	• Eingeschränkte Funktionalität im Vergleich zu anderen Modi
	• Einfache Einrichtung und Nutzung	• Begrenzte Reichweite, Benutzer muss vor Ort sein
	• Hohe Flexibilität, da es an jedem Ort eingesetzt werden kann	• Keine zentrale Verwaltung oder automatisierte Darstellung möglich
	• Ideal für den mobilen Einsatz und Privatpersonen	

Tabelle 5.1: Standalone-Modus

5.4.2 Netzwerk-Modus

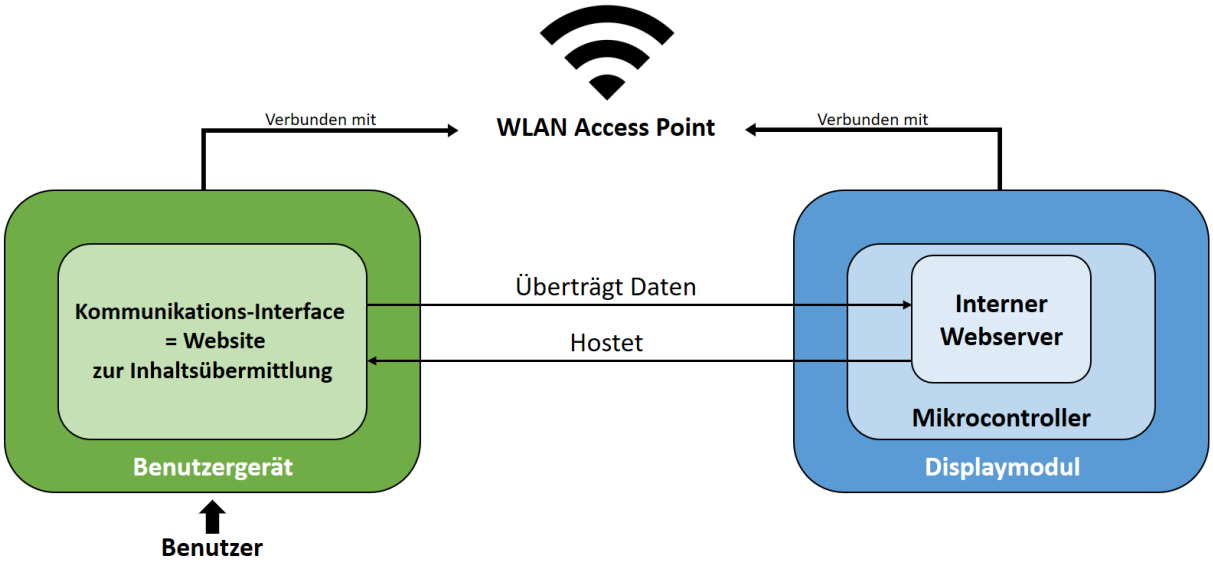


Abbildung 5.3: Netzwerk-Modus

Modus	Vorteile	Nachteile
Netzwerk	• Nutzung bestehender WLAN-Infrastruktur	• Erfordert ein funktionierendes WLAN-Netzwerk
	• Mehrere Geräten innerhalb des gleichen Netzwerks können mit dem Display kommunizieren	• Abhängigkeit von bestehendem Netzwerk
	• Benutzer muss nicht vor Ort sein	• Zugeteilte IP-Adresse des Moduls muss bekannt sein, um sich zu verbinden

Tabelle 5.2: Netzwerk-Modus

5.4.3 Server-Modus

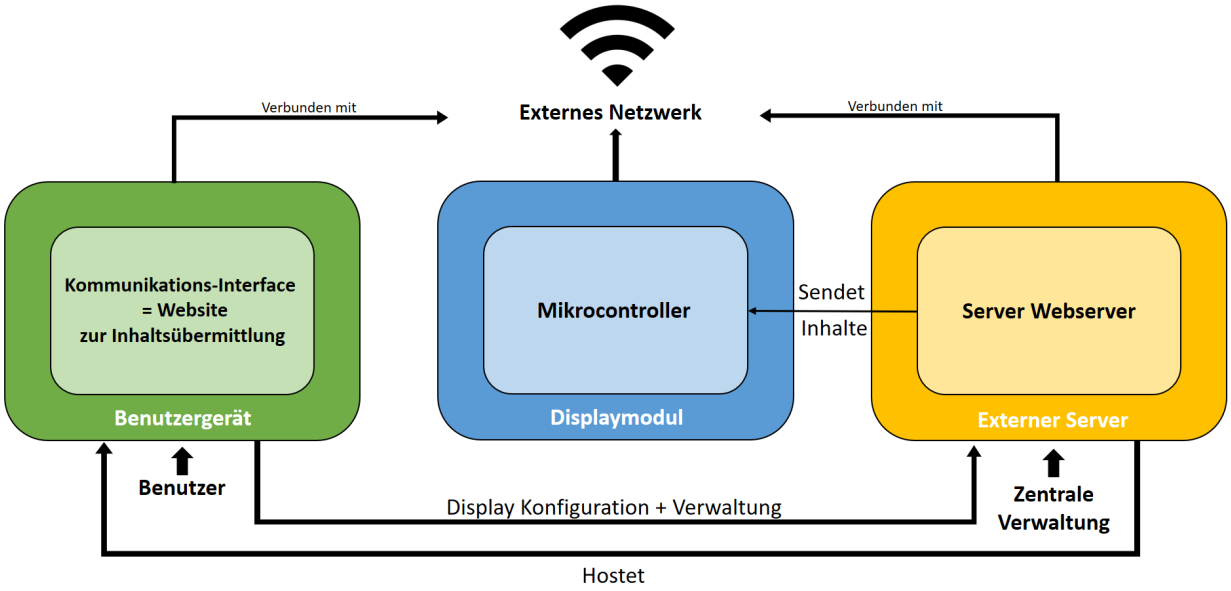


Abbildung 5.4: Server-Modus

Modus	Vorteile	Nachteile
Server	• Zentrale Verwaltung und Steuerung vieler Geräte	• Etwas höherer Einrichtungs- und Wartungsaufwand
	• Möglichkeit zur Integration mit Kalender- und Verwaltungssystemen	• Erfordert Server-Hardware und weitere IT-Ressourcen
	• Geeignet für große Organisationen und Institutionen	• Abhängigkeit von der Serververfügbarkeit und Netzwerkinfrastruktur
	• Ermöglicht komplexe Funktionen wie automatische Updates, Synchronisation und umfangreiche Datenanalyse	

Tabelle 5.3: Server-Modus

6. Hardware

6.1 Auswahl des Mikrocontrollers

Benjamin Klaric

In der Tabelle in Kapitel 2.1 Mikrocontroller wurden schon die verschiedenen Möglichkeiten zur Auswahl von Mikrocontrollern dargestellt. Wenn man die Anforderungen, wie Anzahl GPIO Pins, Stromverbrauch in Sleep Modi, Bauform, interne Speicher und mögliche Vorteile über den anderen, wie z. B. intern gebauten Ladegerät, erschien schon ein eindeutiger Gewinner, in Form von XIAO ESP32-S3. Die verschiedenen STM32 Mikrocontroller waren auch lange Zeit in Betrachtung, wurden aber herausgelassen aufgrund der Tatsache, dass sie kein eingebautes Wi-Fi-Modul haben.
Der XIAO ESP32-S3 mit seinem kleinen Formfaktor, sehr geringen Stromverbrauch in Deep Sleep Modus, enormen Internen speicher (im Vergleich mit anderen Kandidaten) und Möglichkeit, die Akku über den USB-C-Anschluss zu laden, machen den Mikrocontroller sehr flexibel und für das Systemanforderungen passend. [BK_07]
Der Prozessor, der auf dem Mikrocontroller läuft, ist der leistungsstarke Xtensa LX7 Dual-Core, 32-Bit-Prozessor, mit bis zu 240 MHz Taktfrequenz. Mit internen 512 KB SRAM-Speichern ist er in der Lage, viele Operationen und Funktionen lokal durchzuführen. Neben dem SRAM-Speicher ist der Mikrocontroller auch extern mit 8 MB Flash und 8 MB PSRAM Speicher ausgestattet. Der Mikrocontroller ist auch mit interner Hardware Watchdog Timers ausgestattet, die in dem Fall, wo der Mikrocontroller hängend geblieben ist, eine Möglichkeit von Neustart des Systems anbietet. [BK_08]
Aufgrund der niedrigen Idle Strom von 22 mA und noch niedrigere Deep Sleep Strom von 14 µA stellt sich das Modul perfekt für ein autarkes System. Die Möglichkeit von Laden der Akkus durch das USB-C ist natürlich auch von Vorteil, allerdings mit einem niedrigen Ladestrom von 100 mA. Und das alles befindet sich auf einer kleinen Platine mit Maßen 21x17,5 mm, der ungefähre Große von einer 2-Euro-Münze. [BK_07]
Der XIAO ESP32-S3 stellt elf GPIO Pins zur Verfügung, was auf Abildung 6.1 zu sehen ist. Von diesen elf GPIO-Pins sind neun analoge Pins, an die man analoge Signale anschließen kann. Neben den UART-Pins, die für die serielle Kommunikation mit dem Mikrocontroller genutzt werden, bietet der XIAO ESP32-S3 auch eine SPI- und eine I2C-Schnittstelle. Diese Schnittstellen sind serielle synchrone Datenbusse, die zur Kommunikation mit verschiedenen Peripheriegeräten verwendet werden können. Der 5V-Pin ist nur dann aktiv, wenn das Modul über den USB-C-Anschluss betrieben wird. Andererseits ist der 3,3V-Pin ist immer aktiv und dient als regulierter Ausgang mit einem maximalen Strom von 700 mA. [BK_07]

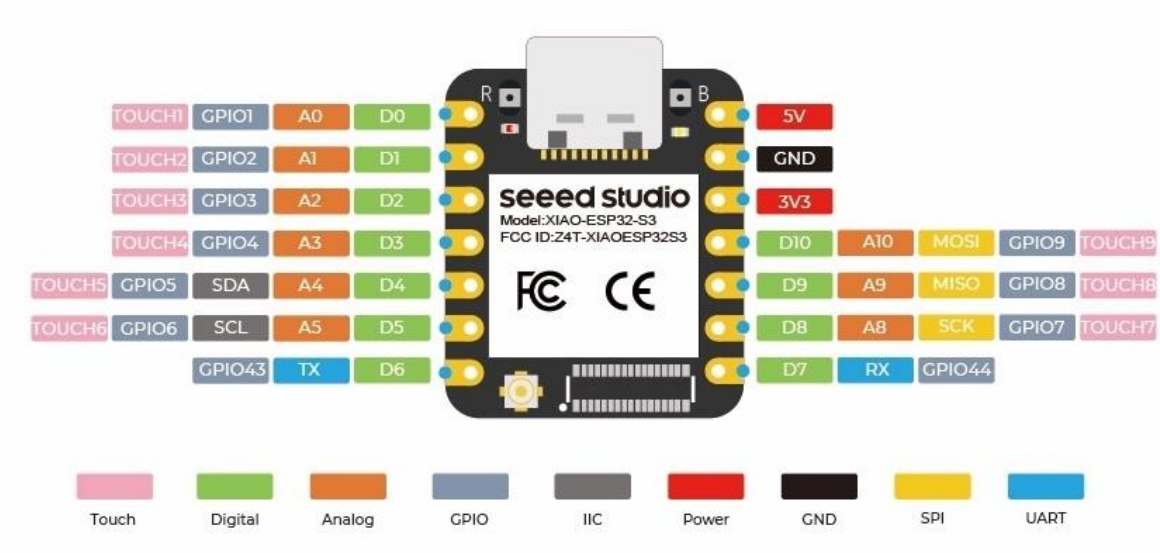


Abbildung 6.1: Pinout von XIAO ESP32-S3 [BK_07]

6.2 Auswahl der Displaytechnologie

Stasa Lukic

Unsere Display auswahl hat sich letztlich auf die ePaper Technologie und bistabile LCD-Technologie beschränkt, durch ihren Vorteil der deutigen Stromeffizienz. Zwischen diesen beiden Technologien haben wir uns für ePaper entschieden, der Vorteil, mit ePaper haben wir die Möglichkeit noch mehr Strom zu sparen und durch das Papierartige Display ist besitzt es eine bessere Lesbarkeit als Bistabile LCD. ePaper besitzt weniger Farbmöglichkeiten, dadurch entschieden wir uns 2 Displays für das Projekt zu testen, ein Display ohne Farben, aber dafür schnelle Refresh-Time und Display mit Farben, aber mit einer längeren Refresh-Time. Das erste Display kann man in Abbildung 6.3 sehen, es kommt mit einer Refresh-Time von 0,5 Sekunden, besitzt dafür aber nur 8 mögliche Greyscales. Das zweite Display besitzt 4 Farben, Weiß, Schwarz, Gelb und Rot und kann mit in Abbildung 7.2.1 sehen. Dies kommt aber mit dem Nachteil von einer Refresh-Time von 16 Sekunden.

Abbildung 6.2: Waveshare 7.3inch (G) ePaper Display

Abbildung 6.3: Waveshare 9.7inch ePaper Display

Die erwähnten geringen Stromverbrauch haben wir selber nochmal getestet und kann man im im Kapitel 7.9 Strommessung von Mikrocontroller und Display genauer nachlesen. Ein kleiner Nachteil der Auswahl ist, dass verschiedene Displays verschiedene Anforderungen haben. Die Displays besitzen einen unterschiedlichen Spannungsverbrauch. Das 7.3 inch Display benutzt 3.3V und das größere 9.7 inch Display benutzt 5V. Dadurch haben wir uns entschieden, einen Schalter auf der Platine einzubauen, der dieses Problem löst und zwischen diesen Spannungen wechselt, genaueres kann man im Kapitel 7.5 Schaltungsentwurf nachlesen.

6.3 Auswahl der Akkukomponenten

Benjamin Klaric

Akku
Der ausgewählte Akku, aus verschiedenen Akku-Technologien, die in 3.3 Akku-Technologien genannt waren, war der klassische Li-Ion-Akku aus mehreren Gründen. Aufgrund der Anzahl und Position des Systems, die von Akku betrieben sein wurde, war der Sicherheitsaspekt ein entscheidender Faktor. Deswegen wurden die LiPo-Akkus nicht mehr betrachtet.
Der Vergleich zwischen Li-Ion-Akkus und $LiFePo_4$-Akkus hatte eindeutige Unterschiede geliefert. Die $LiFePo_4$-Akkus repräsentieren die sicherste Variante, andererseits hatten die Li-Ion-Akkus eine höhere Nennspannung von 3,7V.
Aus Sicherheitsblick betrachtet wurden die $LiFePo_4$-Akkus ausgewählt. Die Nennspannung von 3,2V hat jedoch gegen die $LiFePo_4$-Akkus gesprochen, da der ausgewählte Mikrocontroller, nämlich den XIAO ESP32-S3, Betriebsspannung zwischen 3,2V und 4,2V angefordert hat. Obwohl die $LiFePo_4$-Akkus die Betriebsspannung liefern konnten, konnten sie das System nicht betreiben, da die genannte Betriebsspannung in Fall ohne schwere Last gemeint ist. Wegen dies wurden allerdings die Li-Ion-Akkus gewählt.
Der nächste Schritt war, eine vernünftige Kapazität zu finden, sodass die Akkus das System lang genug betreiben konnten. Dazu hat man die Schätzungen mit Verbrauchmessungen unterstützt und ungefähr die benötigte Akku-Kapazität geschätzt. Eine Akkulaufzeit von einem Jahr war gewünscht und dementsprechend war die Große des Akkus gewählt, nämlich ~10000 mAh. Ein

Akku mit so einer Kapazität war aber nicht einfach zu finden. Deswegen wurde das Akkupack aus zwei ~5000 mAh Akkus gebaut. Um von zwei ~5000 mAh auf die gewünschte Kapazität zu kommen, wurden die Akkus in parallel verbunden, da so die Kapazität sich verdoppelt; im Vergleich, wenn man die in Reihe anschließt, so wird die Spannung verdoppelt.
Bei der Suche nach Akkus muss man einige Sachen beachten. Der Formfaktor der Akku, minimale Kapazität, ob der Akku schon ein eingebautes BMS hat und der Entladestrom.
Es würde die BAK N21700CD-53E ausgewählt, die auf Abildung 6.4 zu sehen ist.

Abbildung 6.4: Li-Ion-Akku [BK_09]

Aus der Tabelle der technischen Daten auf der Abildung 6.5 des ausgewählten Akkus ist es klar zu sehen, dass alle benötigten Parameter erfüllt sind. Der Formfaktor ist aus der Durchmesser und Höhe (die ersten zwei Ziffern von Durchmesser und normalerweise erste drei Ziffern von Höhe → 21700).

Technische Daten zu der BAK N21700CD-53E

Kapazität	5300mAh
Minimale Kapazität	5150mAh
Nennspannung	3,6V - 3,7V
Ladeschlussspannung	4,2V ± 0,05
Entladeschlussspannung	2,5V
max. Entladestrom (konstant)	2C 10,3A (10300mA)
max. Entladestrom (Puls)	ca. 15A - 20A
max. Ladestrom	1C 5,150A (5150mA)
empfohlene Ladestrom	0,5C 2,5A (2500mA)
Schutzelektronik BMS	keine
Pluspol	flach (Flat Top)
Durchmesser	21,25 mm ± 0,15 mm
Höhe	70,50 mm ± 0,25 mm
Gewicht	70g ± 1 g
Ladeverfahren	CC-CV
Datenblätter und Zertifikate	siehe technische Daten

Abbildung 6.5: Technische Daten von BAK N21700CD-53E [BK_09]

BMS
Um den Akkupack sicher betreiben zu können, wurde ein BMS benötigt. Wie bereits in 3.3 Akku-Technologien erwähnt, schützt ein BMS die Akkus vor Tiefentladung, Überladung, Kurzschluss und Überstrom.

Beim genaueren Blick auf ein 1S 2MOS BMS, auf Abildung 6.6, wo 1S die Anzahl der Zellen in Serie repräsentiert, erkennt man einen IC und zwei MOSFET Transistoren auf der Vorderseite. Die MOSFET Transistoren sind nämlich die NMOS-Transistoren und dienen als Schalter, die vom IC durch Gate-Spannung angesteuert werden.

Überladungsschutz: Wenn die Ladespannung 4,2V überschreitet, sendet der IC ein Signal, das die Gate-Spannung eines NMOS-Transistors steuert, wodurch dieser den Stromkreis unterbricht.

Tiefentladungsschutz: Bei Tiefentladung wird ein ähnlicher Mechanismus aktiviert, allerdings bei 2,5V. Der IC sendet ein Signal an den zweiten NMOS-Transistor, um den Stromfluss zu unterbrechen, sobald die Spannung einen kritischen Wert erreicht.

Überstromschutz: Der IC überwacht den Stromfluss kontinuierlich. Wenn der Strom über den zulässigen Grenzwert steigt, schaltet der IC einen der NMOS-Transistoren ab, um den Stromfluss zu unterbrechen und die Batterie vor Schäden zu schützen.

Kurzschlussschutz: Auch bei einem Kurzschluss, bei dem sehr hohe Ströme fließen, erkennt der IC diese Situation und unterbricht sofort den Stromfluss durch Abschalten der NMOS-Transistoren.

Das BMS, das auf Abildung 7.xx dargestellt ist, liefert alle diese Features und wurde für den Aufbau von dem Akkupack ausgewählt.

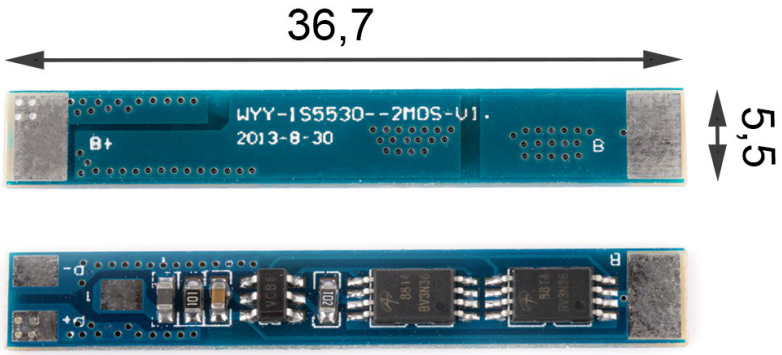


Abbildung 6.6: 1S 2MOS Batteriemanagementsystem [BK_10]

Ladegerät
Wie bereits in 3.3 Akku-Technologien, benötigen die Li-Ion-Akkus ein spezielles Ladegerät, nämlich ein Ladegerät, der in die Lage ist beim Laden der Akkus erste 80% konstanter Strom zu liefern und danach 20% die Akkus mit konstanter Spannung zu laden.
Man kann solche Ladegeräte ohne viele Schwierigkeiten finden, sind allerdings teuer. Bei der Suche nach einem muss man eine Sache beachten, nämlich die Zellenanzahl. Das ausgewählte Ladegerät ist auf der Abildung 6.7 zu sehen.

Abbildung 6.7: Ladegerät für Li-Ion-Akkus [BK_11]

6.4 Zusammenbau der Akkupacks

Mario Wegmann

6.4.1 Konzeptioneller Aufbau

Bei der Auswahl der Akkukomponenten wurde die Entscheidung getroffen, mehrere Lithium-Ionen-Zellen zu verwenden. Da die Kapazität des Akkupacks vergrößert wird, die Batteriespannung jedoch weiterhin zwischen 2,5 V und 4,2 V liegen soll, werden die beiden Lithium-Ionen-Zellen parallel miteinander verbunden. Dadurch verdoppelt sich die Kapazität, während die Spannung gleich bleibt. Zudem wird der gemeinsame Pluspol mit dem Plus des Battery Management System verbunden und ebenso mit dem Minuspolen verfahren. Zum Verbinden von den Akkuzellen untereinander und mit dem BMS eignet sich ein Nickelband, welches auch als Hiluminband bekannt ist[MW_07]. Der Aufbau des Akkus ist auch in den Abbildungen 6.8 und 6.9 ersichtlich.

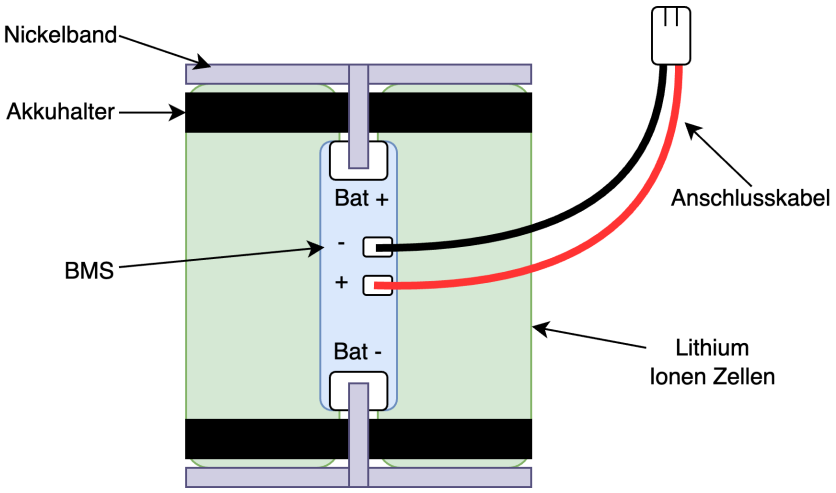


Abbildung 6.8: Konzeptioneller Aufbau des Akkus

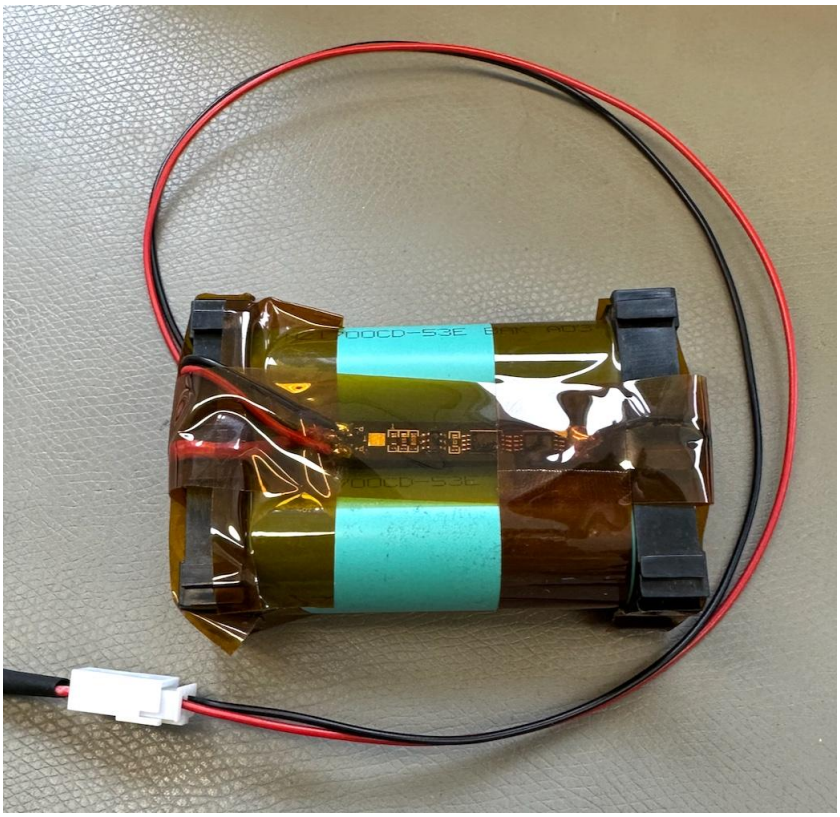


Abbildung 6.9: Foto des aufgebauten Akkus

6.4.2 Battery Management System anschließen

Als Erstes sollte das BMS verdrahtet werden. In diesem Projekt wurde sich dafür entschieden, dass das Akkupack über einen Stecker mit dem Mainboard verbunden werden kann und somit modular ist. Somit wurde an den Ausgängen des BMS ein 2 Pin JST-X2 Stecker gelötet. Hierbei wurde sich im Team intern darauf geeinigt, dass der Pluspol rechts ist, wenn von der Drahtseite auf den Stecker geschaut wird und die Nase des Steckers nach oben zeigt. An den Pads für die Batteriepole wurde das Nickelband ebenso verlötet.

6.4.3 Vorbereiten der Lithium-Ionen-Zellen

Für einen sicheren Umgang mit Lithium-Ionen-Zellen ist es wichtig, mehrere Dinge zu beachten, bevor mehrere Zellen miteinander verbunden werden können. Bereits beim Beschaffen von den Zellen sollte darauf geachtet werden, das identische Modell und eine gleiche Charge zu verwenden. Zudem sollten Zellen, die verbunden werden sollen, möglichst gleich alt und auch gleich belastet werden oder im Optimalfall komplett neu sein. Zuletzt sollte vor dem Verbinden darauf geachtet werden, dass die Zellen die gleiche Zellspannung aufweisen, um einem schlagartigen Ladungswechsel beim Verbinden vorzubeugen.

Nachdem die einzelnen Zellen vorbereitet waren, wurden diese in die Plastikhalter eingelegt. Diese Plastikabstandshalter ergeben zusammen mit den Zellen ein stabiles Gesamtsystem und halten die Zellen davor ab, sich direkt zu berühren.

6.4.4 Punktschweißverfahren bei Lithium-Ionen-Zellen

Lithium-Ionen-Zellen sind wärmeempfindlich und daher ist Weichlöten kein geeignetes Verfahren, um die Zellen mit dem Hiluminband zu verbinden, stattdessen eignet sich das Punktschweißverfahren. Hierbei wird das Nickelband an einem Pol einer Zelle gepresst und dann die zwei Elektroden des Punktschweißgerätes auf das Nickelband gedrückt. Bei dem in diesem Projekt verwendeten Gerät fließen durch die beiden Elektroden 650 Ampere bei 4,2 Volt, dieser hohe Strom führt zu einem Schweißpunkt, welcher das Nickelband und den Pol fest verbindet, jedoch aufgrund der kurzen Dauer des Prozesses von circa 5 Millisekunden kaum eine Wärmebelastung für die Akkuzelle darstellt. Das Nickelband, welches bereits mit dem BMS verbunden ist, wurde um ein weiteres Nickelband orthogonal dazu erweitert. Dadurch ergibt sich eine Nickelverbindung in T-Form. An beiden offenen Enden wurde jeweils ein Batteriepol angeschlossen. Dies wurde für die andere Polseite wiederholt [MW_08].

6.4.5 Isolieren des Akkupacks

Abschließend wurde Kapton Klebeband verwendet, um das Akkupack mit einer isolierenden Schicht zu umhüllen. Dadurch wird vermieden, dass unkontrolliert die Batteriepole mit anderen leitfähigen Materialien in Berührung kommen und sämtlicher Strom über das BMS geleitet wird. Kapton Klebeband ist dabei ein sehr guter elektrischer und thermischer Isolator.

6.5 Schaltungsentwurf

Benjamin Klaric

Um das gesamte System optimal zu gestalten, wurde eine Platine entworfen, die alle benötigten Komponenten und Schaltungen integriert, um die normale Funktionalität des Systems zu gewährleisten. Für die Entwicklung der Platine war ein präziser Schaltungsentwurf mit den richtigen Komponenten erforderlich. Der Schaltungsentwurf sowie der Platinentwurf wurden mithilfe der ECAD-Software KiCad erstellt. Die Auswahl der geeigneten Komponenten sowie eine detaillierte Beschreibung der Schaltung sind im folgenden Abschnitt beschrieben.
Alle Komponenten, die in der Schaltung beinhaltet sind, befinden sich in dem sogenannten Bill of Materials (kurz BOM).

Ganze Schaltung
Auf der Abildung 6.10 ist die vollständige Schaltung mit allen Teilschaltungen dargestellt.

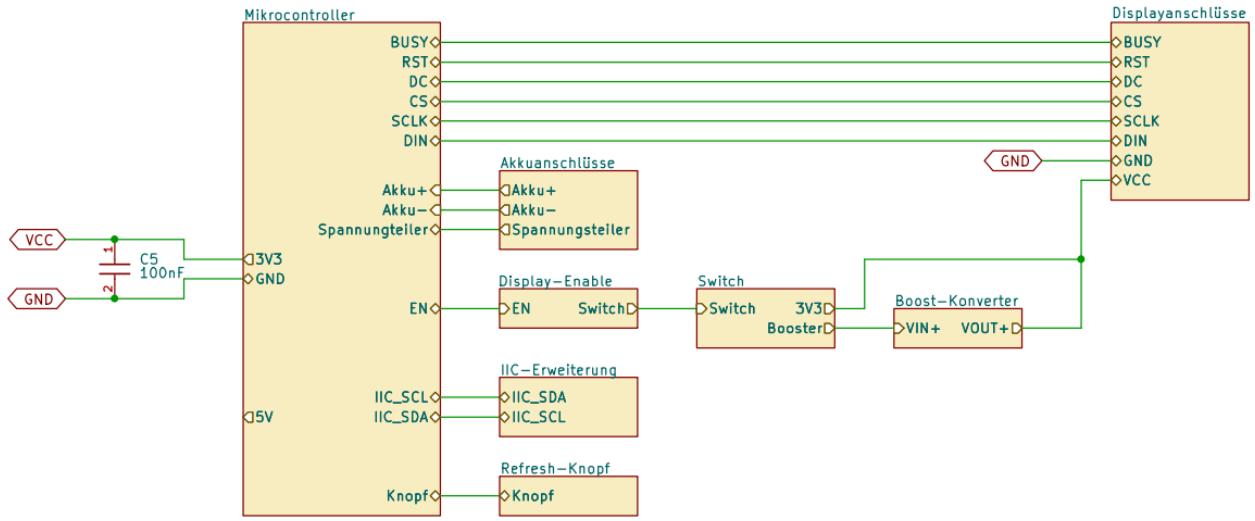


Abbildung 6.10: Ganze Schaltung für das Mainboard des Systems

Wenn man die Abildung 6.10 betrachtet, sind die Verbindungen zwischen den Teilschaltungen und der Gesamtverdrahtung der Schaltung sichtbar. Spätere Abschnitte behandeln jede Teilschaltung einzeln, und es wird empfohlen, zur Gesamtabbildung der Schaltung zurückzukehren, um die Verbindung der erklärten Teilschaltungen im Gesamtkontext zu sehen.
Ganz links in der Abbildung ist ein 100 nF Kondensator, speziell C5, zwischen 3,3V (V_{CC}) und Ground (GND) parallel geschaltet. Dieser dient als Entkopplungskondensator, um Spannungsschwankungen zu reduzieren und die Stabilität der Stromversorgung sicherzustellen. Der 100 nF Kondensator fängt hochfrequente Störungen ab und glättet diese. Dadurch verbessert er die Betriebssicherheit des Systems, insbesondere während schneller Schaltvorgänge, indem er eine stabile Spannungsversorgung gewährleistet und die Signalqualität erhöht. Dieser Wert von 100 nF ist ein Standardwert für Entkopplungskondensatoren bei niedrigen Frequenzen. [BK_12] [BK_13]

Mikrocontroller
Wenn man die Teilschaltung des Mikrocontrollers auf der Abildung 6.11 betrachtet, fällt auf, dass außer dem Mikrocontroller keine weiteren Komponenten zu sehen sind. Die gesamte Schaltung wurde in Teilschaltungen oder sogenannte Subsheets, wie das Feature in KiCad genannt ist, aufgeteilt, um sie übersichtlicher darzustellen.

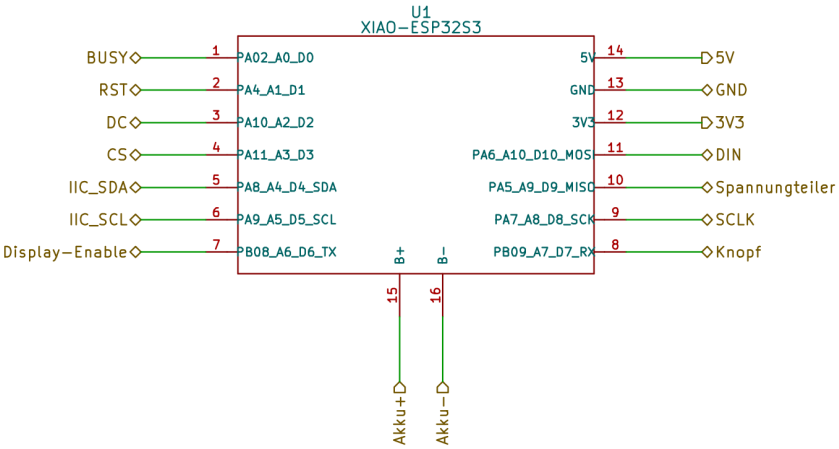


Abbildung 6.11: Ausgänge des Mikrocontrollers

In dieser Teilschaltung sind die Ausgänge des Mikrocontrollers hierarchischen Labels mit entsprechenden Namen zugeordnet.

Akkuanschlüsse
Die Akkuanschlüsse in der Teilschaltung beinhalten einen Stecker, konkret den J1, sowie einen Spannungsteiler, der aus R5 und R6 besteht. Diese Komponenten sind auf der Abildung 6.12 dargestellt.

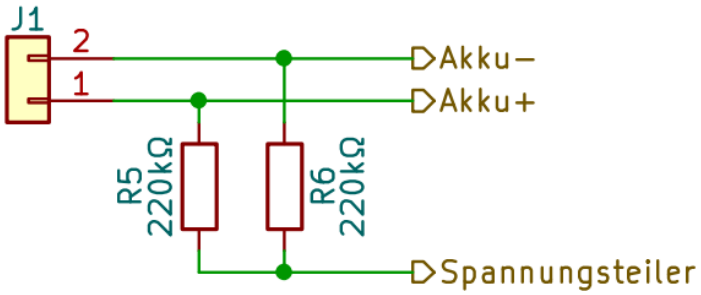


Abbildung 6.12: Schaltung von Akkuanschlüsse

Der Akkupack ist über den Stecker mit den Akkuanschlüssen des Mikrocontrollers verbunden. Parallel dazu ist ein Spannungsteiler aufgebaut. Der Zweck dieses Spannungsteilers besteht darin, die Spannung des Akkupacks zu messen, indem ein analoger GPIO-Pin des XIAO ESP32-S3 verwendet wird. Dadurch wird der Zustand des Akkupacks genau überwacht, um sicherzustellen, dass ausreichend Spannung für den Mikrocontroller bereitgestellt wird.

Display-Enable

Auf der Abildung 6.13 ist die sogenannte Display-Enable Schaltung zu sehen. Man kann drei Widerstände, nämlich den R1, R2 und R3 erkennen, neben den zwei MOSFETs, einen NMOS und einen PMOS Transistor.

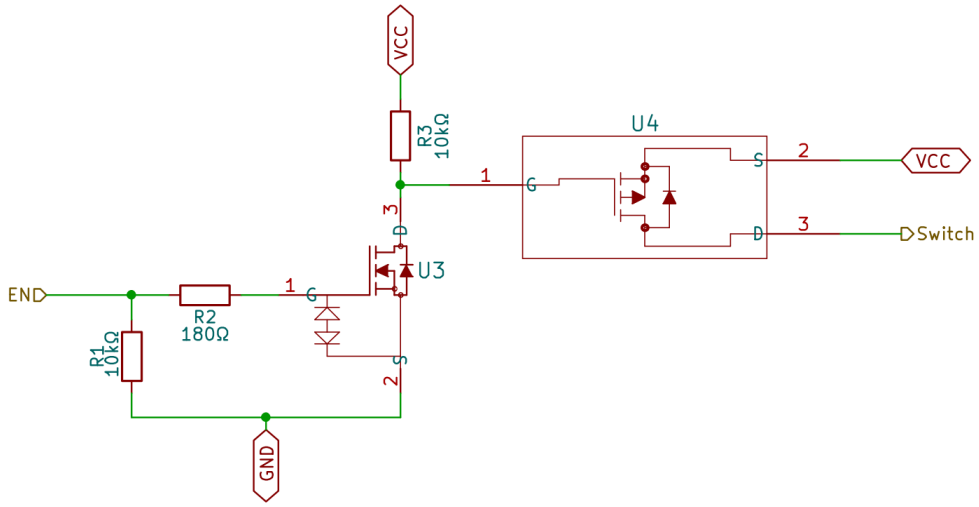


Abbildung 6.13: Schaltung von Display Enable

Die E-Paper-Technologie von Displays verbraucht tatsächlich keinen Strom im Idle-Zustand. Das Gleiche kann aber nicht für die kleinen Zwischenplatinen gesagt werden, die Parallelport-Kommunikationsverfahren zu SPI übersetzen. Diese verbrauchen ständig Strom, was eine unerwünschte Folge ist. Um den Stromverbrauch zu optimieren, wurde eine Schaltung entworfen, die die Stromversorgung des Displays durch Code trennt.
Um die Schaltung per Code ansteuern zu können, wurden zwei MOSFETs verwendet: ein N-Channel MOSFET und ein P-Channel MOSFET. Der Ausgang des Mikrocontrollers geht über den 180 Ω Widerstand, nämlich R2, auf das Gate des N-Channel MOSFET (U3). Der 180 Ω Widerstand dient dazu, den Gate-Strom zu begrenzen und das Schalten des Transistors zu stabilisieren. Dazu wird auch ein Pulldown-Widerstand (R1) eingesetzt, da beim Einschalten des Mikrocontrollers der EN-Ausgang nicht sofort auf 0V gesetzt sein könnte. Ein Wert von 10 kΩ ist ein Standardwert, weil er ausreichend ist, um das Gate sicher auf 0V zu ziehen, ohne signifikanten Strom zu verbrauchen. Um ein unnötiges Einschalten des NMOS zu vermeiden, wurde der Pulldown-Widerstand gesetzt.
Die Source-Leitung des NMOS ist mit dem Ground verbunden, da dies dem Funktionsprinzip des NMOS entspricht. Die Drain-Leitung ist mit dem Gate des P-Channel MOSFET (U4) verbunden und auch mit einem Pull-up-Widerstand (R3). Der Pull-up-Widerstand sorgt dafür, dass das Gate des PMOS auf 3,3V gehalten wird, wenn der NMOS nicht leitend ist, wodurch der PMOS ausgeschaltet bleibt.
Der NMOS-Transistor (U3) schaltet ein, wenn der EN-Pin des Mikrocontrollers auf High (3,3V) geht. Dadurch wird das Gate des PMOS-Transistors auf 0V gezogen. Der PMOS-Transistor (U4) schaltet ein, wenn das Gate auf 0V gezogen wird, wodurch die Verbindung zwischen VCC und Switch geschlossen wird.

Die ausgewählten MOSFETs wurden aufgrund ihrer niedrigen Schwellenspannung und ihres geringen Drain-Source-Widerstands gewählt, was eine effiziente und zuverlässige Schaltung ermöglicht.

Switch
Man kann auf der Abildung 6.14 ein Switch erkennen, der dafür zuständig ist, die Betriebsspannung des Displays auszuwählen, zwischen direkten 3,3V oder dem Ausgang von Booster Schaltung.

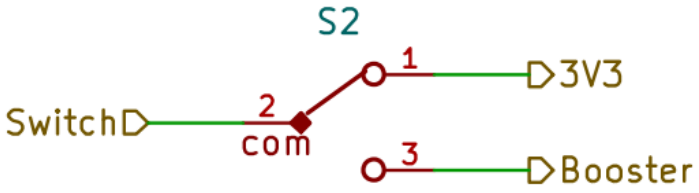


Abbildung 6.14: Switch für Ansteuerung der Betriebsspannung

Der Switch wird per Bedarf mittels menschlichen Einfluss die Leitung zur Betriebsspannung von Display umleiten, zu Booster Schaltung oder direkt mit dem 3,3V Ausgang von Mikrocontroller verbinden. Diese Funktionalität ist so ausgedacht, sodass man die Platine generisch halten kann, um verschiedene Displays mit verschiedenen Betriebsspannungen anzusteuern.

Boost-Konverter
Die Boost-Konverter Schaltung, die auf der Abildung 6.15 dargestellt ist, hat die Aufgabe, die Betriebsspannung zu erhöhen.

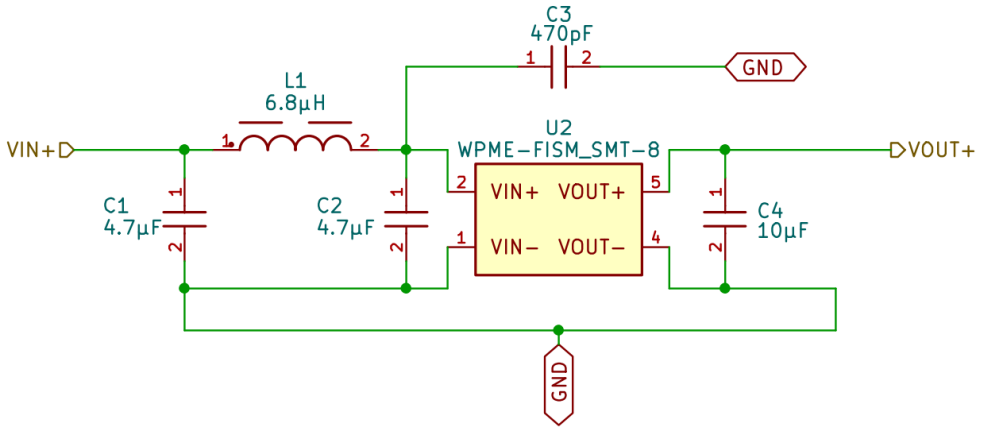


Abbildung 6.15: Booster Schaltung

Der Boost-Konverter befindet sich in der Mitte der Schaltung. Die L1 speichert Energie in ihrem Magnetfeld, wenn Strom durch sie fließt. Während der Schaltzyklen des Boost-Konverters wird diese Energie verwendet, um die Spannung am Ausgang zu erhöhen. Wenn der Schalter innerhalb des Moduls geschlossen ist, fließt Strom durch die Induktivität und speichert Energie. Wenn der Schalter öffnet, wird die gespeicherte Energie freigesetzt, wodurch die Spannung über die Induktivität ansteigt und die Ausgangsspannung erhöht wird. Der Ausgangskondensator, nämlich den C3, glättet die Ausgangsspannung und reduziert das Rauschen am Ausgang des Boost-Konverters. Die L1 zusammen mit C3 nennt sich ein LC-Glied und dient als ein Tiefpassfilter, das hochfrequente Störungen und Rauschen herausfiltert und eine gleichmäßige Gleichspannung liefert.
Der Eingangskondensator C1 reduziert Spannungsrauschen und glättet die Eingangsspannung, bevor sie in die Schaltung eintritt. Der sogenannte Bypass-Kondensator C2 dient dazu, hochfrequente Störungen und Rauschen herauszufiltern. Der C4 Kondensator wird verwendet, um elektromagnetische Interferenzen (EMI) zu reduzieren. Er hilft dabei, Rauschen und Störungen zu filtern, die durch die Schaltung erzeugt werden könnten, und sorgt für eine saubere Ausgangsspannung.
Der Boost-Konverter hat intern auch eine galvanische Trennung. Diese galvanische Trennung dient dazu, elektrische Isolation zwischen dem Eingang und dem Ausgang zu gewährleisten. Sie schützt vor elektrischen Störungen und sorgt dafür, dass keine direkten elektrischen Verbindungen zwischen den beiden Seiten bestehen. In diesem speziellen Fall wurde jedoch entschieden, die galvanische Trennung nicht zu nutzen, wegen niedriger Leistungsanforderungen des Systems und der Tatsache, dass die elektrische Isolation in Form von galvanischer Trennung nicht kritisch ist. [BK_14]

IIC-Erweiterung
Auf der Abildung 6.16 sind die IIC Ausgänge des Mikrocontrollers dargestellt, mit jeweils einem Pull-up Widerstand von 10 kΩ.

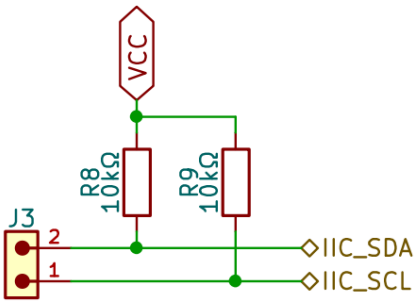


Abbildung 6.16: IIC-Ausgänge

Die IIC Pins von dem Mikrocontroller wurden auf die Platine frei zugänglich gemacht, sodass die Erweiterungsmöglichkeiten in Form eines Sensors oder Ähnliches einfacher zu implementieren sind. Momentan bleiben die unverbunden.

Refresh-Knopf
Der benötigte Knopf ist auf der Abildung 6.17 dargestellt. Die wird für manuellen Refresh des Displays benutzt.

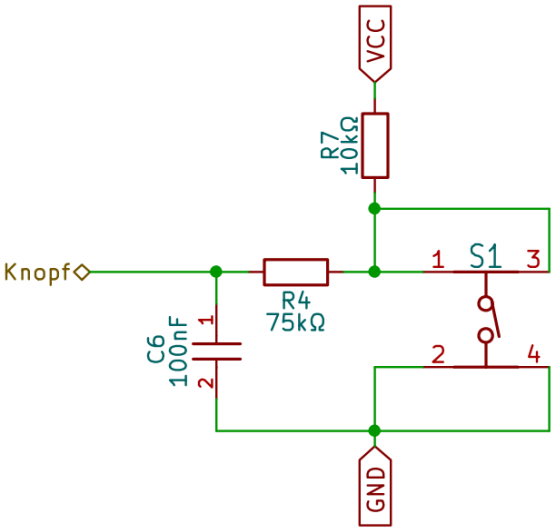


Abbildung 6.17: Schaltung von Refresh-Knopf

Wenn ein mechanischer Schalter oder Taster betätigt wird, kann es zu kurzen ungewollten Kontakten kommen, die als Prellen bezeichnet werden. Diese entstehen durch die mechanische Federung und Vibrationen des Schalters. In elektronischen Schaltungen kann dieses Prellen zu Fehlfunktionen führen, insbesondere wenn der Taster als Trigger dient.
Der verwendete Knopf benötigt eine Entprellschaltung, um zuverlässig zu funktionieren, da er eine maximale Prellzeit von 10 ms aufweist. [BK_15]
Um das Prellen zu beseitigen, wird ein RC-Glied eingebaut, das als Hochpassfilter fungiert. Wenn der Taster offen ist, lädt sich der Kondensator C6 über die Widerstände R4 und R7 auf, wodurch die Spannung langsamer ansteigt. Ist der Taster geschlossen, wird der Kondensator über R4 mit einer kontrollierten Geschwindigkeit entladen. Der Widerstand R7 hat einen Wert von 10 kΩ, da es sich um einen Pull-up-Widerstand handelt. Für R4 wird ein Wert von 75 kΩ gewählt, um in Kombination mit dem Kondensator von 100 nF die gewünschte Zeitkonstante zu erreichen. [BK_16]

Displayanschlüsse
Die dargestellten Ausgänge auf der Abildung 6.18, nämlich den J2-Komponente, sind die Ausgänge, die man direkt mit dem Display über Dupont Pins verbinden kann.

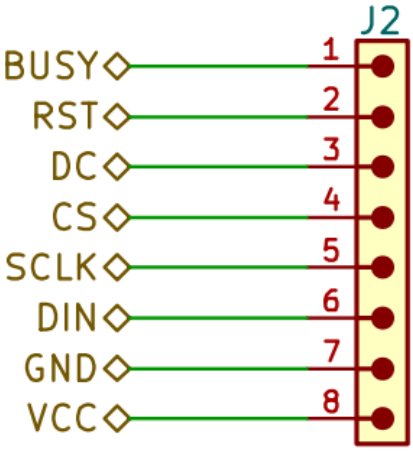


Abbildung 6.18: Ausgänge für das Display

Die ersten sieben Ausgänge kommen direkt von dem Mikrocontroller, wobei der achte Ausgang, konkret den V_{cc} Ausgang, von dem Switch abhängig ist.

6.6 Platinendesign

Benjamin Klaric

Mit der in 7.5 Schaltungsentwurf entwickelten Schaltung kann eine Platine entworfen werden. Hierfür wurde, wie bereits erwähnt, das ECAD-Programm KiCad verwendet, um die entworfene Schaltung auf einer Platine umzusetzen.
Für die ausgewählten Komponenten müssen jedoch die entsprechenden *Footprints* gefunden werden. Ein Footprint definiert die Fläche und die Bohrungen, die eine Komponente benötigt, um auf der Platine montiert zu werden. Darüber hinaus werden 3D-Modelle ausgewählt, um eine bessere 3D-Ansicht in KiCad zu ermöglichen und mögliche Überlappungen der Komponenten besser zu erkennen.
Dafür werden die Webseiten der Hersteller besucht, um die benötigten Dateien zu finden. Diese sind oft bereits verfügbar, jedoch nicht immer. In solchen Fällen ist es notwendig, mehrere Webseiten zu durchsuchen, um die erforderlichen Dateien zu finden, beispielsweise über Plattformen wie Ultra Librarian und SnapMagic. [BK_17] [BK_18] Auf diesen Plattformen können Symbole, Footprints und 3D-Modelle für viele Komponenten gefunden werden.
Wenn man aus einer Schaltung eine Platine erstellen möchte, werden die Komponenten mit den dazugehörigen Footprints im PCB-Design-Programm bereits platziert angezeigt. Es ist dann erforderlich, sie an die richtigen Positionen zu bewegen, die Verdrahtung vorzunehmen und die Grenzen der Platine festzulegen. Anschließend muss ein Ground Polygon sowohl auf der oberen als auch auf der unteren Seite der Platine gezogen werden. Dabei sind die Designregeln des jeweiligen PCB-Herstellers zu beachten. In diesem Fall werden die Designregeln des Herstellers Aisler berücksichtigt, wie beispielsweise die Größe der Platine, die Größe der Vias usw. [BK_19] Die einzelnen Schritte werden in den folgenden Abschnitten erklärt.

Platzierung
Das Platzieren der Komponenten an den richtigen Stellen ist eine wichtige Aufgabe, bei der einige wesentliche Aspekte berücksichtigt werden müssen.
Es ist wichtig sicherzustellen, dass platzierte Komponenten sich nicht gegenseitig mit ihren Gehäusen stören. Ebenso spielt der Abstand zwischen den Komponenten eine entscheidende Rolle. Insbesondere für das manuelle Löten ist es vorteilhaft, wenn die Komponenten ausreichend Abstand zueinander haben.
Ein weiterer wichtiger Aspekt ist der Einfluss der Platzierung der Komponenten auf das Routing bzw. die Verdrahtung der Verbindungen innerhalb der Schaltung.

Routing
Die Komponenten müssen gemäß dem Schaltungsentwurf verbunden werden. Für das System wird eine Zweischichtplatine entworfen, die zwei Ebenen bietet, auf denen das Routing durchgeführt werden kann.
Obwohl das Routing manuell durchgeführt werden könnte, wurde ein Autorouter verwendet, da es mit 25 Komponenten schwierig wäre, alles von Hand zu routen. Die Nutzung eines Autorouters reduziert zudem das Risiko von Fehlern, da er nach festgelegten Regeln stets optimal und korrekt routet. Für die Platine wurde der Autorouter mit dem Namen Freerouting in KiCad verwendet. [BK_20]

Um Freerouting zu nutzen, muss zunächst aus dem Platinen-Designfenster eine .DSN-Datei exportiert und in Freerouting importiert werden. Das Ergebnis ist eine .SES-Datei (Specctra Session File), die anschließend wieder in KiCad importiert wird, um das Routing des Autorouters zu übernehmen. Eine detaillierte Anleitung zum Umgang mit Freerouting findet sich im offiziellen GitHub. [BK_20]
Der Autorouter kann gut konfiguriert werden, z.B. bezüglich der Routenbreite, der Anzahl der Vias (Vias sind kleine Bohrungen, die beim Wechsel zwischen der oberen und unteren Seite der Platine verwendet werden), usw.

Grenzen und Ground Polygon
Um die Platine auf die gewünschte Größe zu bringen, müssen die Grenzen definiert werden. Dies kann entweder vor oder nach dem Routing erfolgen. Wie bereits erwähnt, wird auch ein Ground Polygon über beiden Seiten gezogen. Dies dient mehreren Zwecken: Eine gute Masseverbindung reduziert Störungen und verbessert die Signalintegrität, da die Ground-Fläche als niederohmiger Pfad für Rückströme fungiert. Ein großflächiges Ground-Polygon kann elektromagnetische Störungen (EMI) reduzieren, indem es als Abschirmung fungiert und Hochfrequenzsignale ableitet. Zudem hilft ein großes Ground-Polygon, Potenzialunterschiede auf der Platine zu minimieren, was wichtig ist, um unerwünschte Spannungsabfälle oder Spannungsspitzen zu vermeiden. [BK_21]

Ergebnis
Die fertige Platine sieht am Ende so aus, wie auf Abildung 6.19 dargestellt ist. Die rote Fläche repräsentiert das Ground-Polygon auf der oberen Seite und die blaue Fläche das Ground-Polygon auf der unteren Seite der Platine. Die zahlreichen Vias dienen dazu, die Ground-Polygone beider Seiten miteinander zu verbinden. Dies ist besonders wichtig, wenn auf einer Seite keine direkte Verbindung zum Ground möglich ist, zum Beispiel aufgrund anderer Leiterbahnen oder Komponenten.

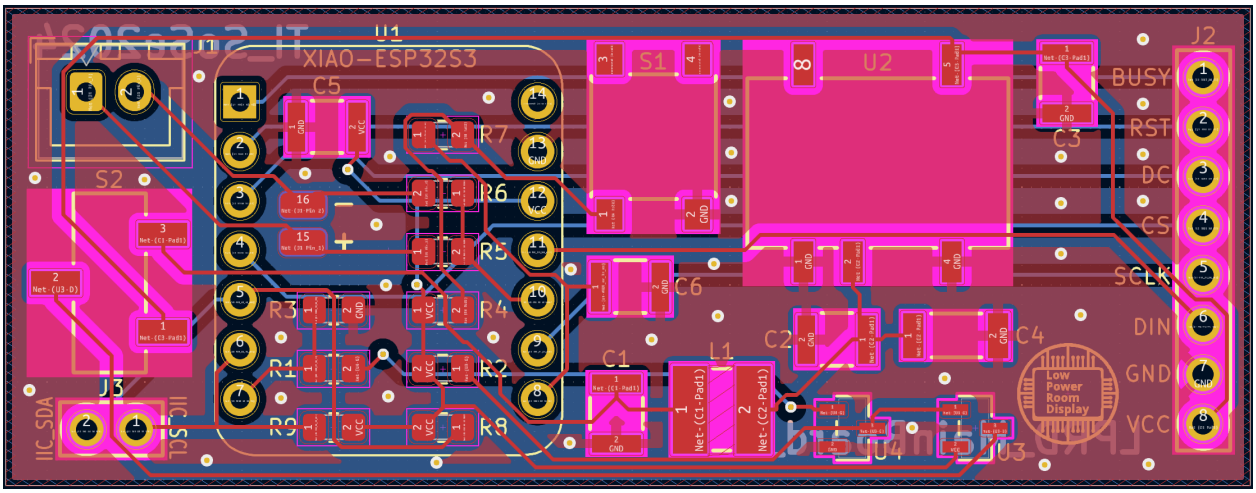


Abbildung 6.19: Fertiges Platinendesign

Wenn man auf 3D-Ansicht wechselt, wurden alle Bauteile, mit seinen 3D-Modellen sichtbar, wie auf Abildung 6.20 zu sehen ist. Es wurde ein Raytracing-Effekt verwendet, um die Bauteile realistischer darzustellen.

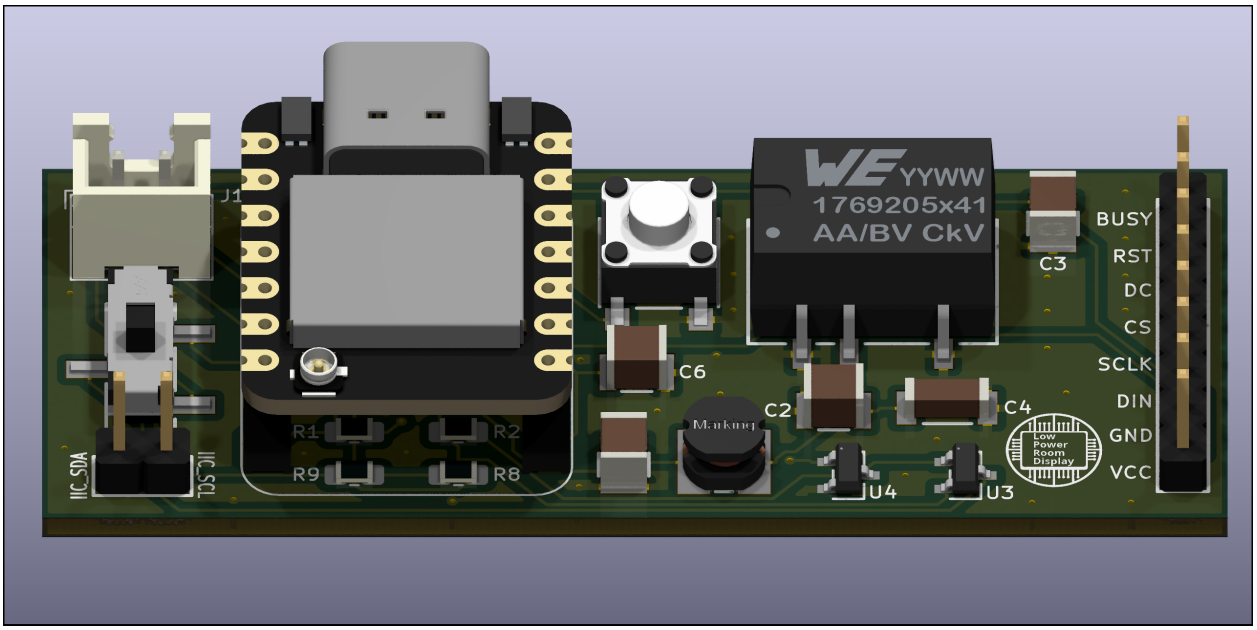


Abbildung 6.20: 3D-Ansicht von der Platine

6.7 Zusammenbau der Platinen

Benjamin Klaric

Nachdem die Platine und alle anderen Bauteile angekommen sind, wurde mit dem Zusammenbau der Platinen begonnen.
Da viele Bauteile mittels SMT (Surface Mounted Technology) mit der Platine verbunden sind, ist es normalerweise sinnvoll, das Löten mit einem Reflow-Ofen durchzuführen. Dafür benötigt man jedoch ein sogenanntes Stencil. Ein Stencil ist ein kleines Metallteil mit Löchern, die den Pads (Kontakten für die Bauteile auf der Platine) entsprechen. Mit einem Stencil kann die Lötpaste viel einfacher und präziser auf die Pads aufgetragen werden. Leider konnte aus Kostengründen kein Stencil bestellt werden, und das manuelle Auftragen der Paste auf 50 Pads ist sehr zeitaufwendig.
Deshalb wurde das Löten von Hand durchgeführt. Die verlötete Platine ist in Abildung 6.21 zu sehen.

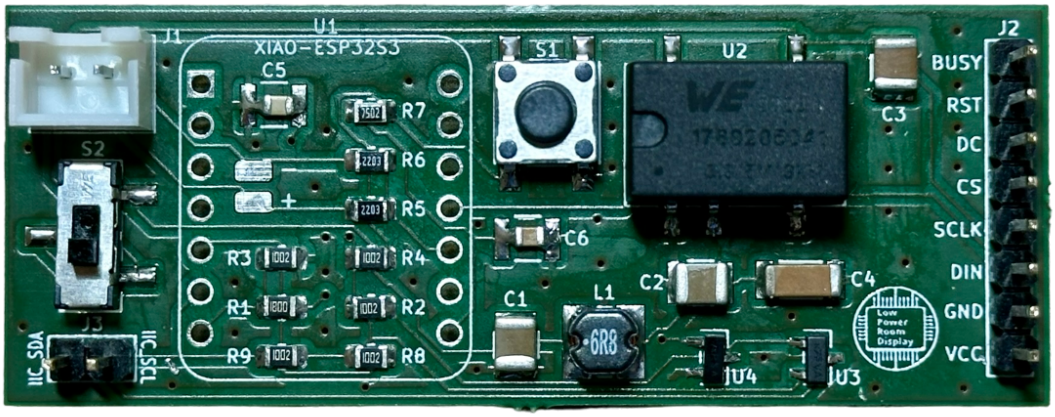


Abbildung 6.21: Fertige Platine ohne Mikrocontroller und Pin Headers

6.8 Gehäuse

Jannis Gröger

6.8.1 Anforderungen

Beim Entwurf des Gehäuse für das Low Power Raumdisplay sind die funktionalen Anforderungen einfach gehalten. Da das Gerät im Einsatz später stationär platziert ist, ist es weder mechanischen Stößen noch einer Art von Vibration zu schützen. Zudem wird es nicht in der Hand bedient, weshalb es auch keinen mechanischen Druck ausgesetzt ist. Daher ist nicht auf ein besonders robustes Material zu achten.

Da der Einsatzort hauptsächlich im Gebäudeinneren vorgesehen ist, ist ein Schutz gegen jegliche Art von Wasser zu vernachlässigen. Da das Display als Bendienung zwei Benutzerknöpfe besitzt, muss das Gehäuse gegen Berühren mit Fingern geschätzt werden, also sollte der Schutz gegen Berührung mindestens der Schutzart zwei zugeordnet werden können. [JG_02]

Die Low Power Raumanzeige besitzt elektronische Komponenten, die keine großen Wärmemengen produzieren. Lediglich beim Laden der Akkupacks könnte ein kritische Wärmemenge entstehen, weswegen diese leicht aus dem Gehöuse entfernbar sein sollten, dass beim Laden genug Abwärme durch natürliche Konvektion gegeben ist.

Da die Informationsanzeige mit WLAN mit dem Gerät des Betreibers kommunziert, sollte das Gehäuse die Komponenten nicht zu sehr elektromagnetisch von außen abschirmen und auch elektromagnetische Felder nach außen hin zulassen, da ansonten die Kommunkation gestört werden und das Displaymodul nicht mehr seine volle Funktionsfähigkeit aufweisen kann.

6.8.2 Design

Zum Entwickeln des 3D-Modells wird die CAD-Software "Autodesk Fusion" genutzt. Das Vorgehen hierbei besteht primär aus der Erstellung einer Skizze, aus der dann verschiedene Fläche extrudiert werden. Um komplexere Konstruktionen zu erlauben, wird dieses Vorgehen wiederholt, wobei die Skizzenfläche eine Oberfläche einer bereits vorhandenen Extrusion ist. Die beiden Gehäuse bestehen jeweils aus drei Teilen:

- Einer Front, in der das Display sitzt
- Einer Halterung für die Platinen des Displays, die an der Front angebracht wird
- einer Rückseite, in der an der Rückwand die Halterungen für Akkupack und selbst entworfener Platine befestigt sind.

Diese Teile werden jeweil mit Gewindeschrauben mit 2,5mm durchmesser verbunden. Die Rückseite kann konstruktionstechnisch nochmal in einen Quader für elektronische Bauteile und einer Schale für das Display unterteilt werden. Diese Teile sind jedoch fest an ihren Kanten verbunden. Im folgenden werden die einzelnen Merkmale, die dem Gehäuse hinzugefügt werden, erläutert.

Um die einzelnen Komponenten im Inneren des Gehäuses zu fixieren, müssen für jede einzelne eine spezifische Halterung angefertigt werden. Die Akkupacks werden durch vier L-förmige Säulen, die jeweils eine Ecke der Packs umschließen, vor dem Hin und Herrutschen bewahrt. Das Herausfallen nach Vorne verhindert ein Steg, der mit zwei Schrauben über dem Akkupack angebracht wird. Die Platine wird durch zwei gegenüberliegende Nuten in der Unterseite des Gehäuses und einer konstruierten Wand im Gehäuse gehalten, indem sie seitlich in die Nuten eingeschoben wird. Hierbei wird an der Seite des Gehäuses eine Bohrung angebracht, um auch im montierten Zustand der Low Power Raumanzeige Zugang zum USB-C Anschluss des Mikrocontrollers zu haben.

Da das Displaymodul wie oben erwähnt nicht in der Hand gehalten wird sondern fest vor bspw einem Hörsaal einer Hochschule platziert werden soll, werden an der Rückwand des Gehäuses zwei Langloch-Einhängeöffnungen angebracht. Um das Display dann an der Wand aufzuhängen, müssen einfach zwei Schrauben in der Wand eingeschraubt werden. Diese werden in die Öffnungen eingefädelt, dann wird das Displaymodul nach unten geschoben und ist damit an der Wand fixiert. Damit das Displaymodul auch mobil eingesetzt werden kann, wie beispielsweise bei einem Messestand, wird die untere Fläche des Gehäuses angeschrägt, dass man das Display stabil auf einen Tisch stellen kann.

Wie bereits genannt, soll die Bedienung des Low Power Raumdisplays über einen Bentzerknopf gesteuert werden. Um die Benutzerfreundlichkeit zu gewährleisten, wird dieser an der Front des Displaymoduls angebracht, damit er leicht zugänglich und für alle sichtbar ist. Zusätzlich sollen die eingebauten "Boot"- und "Reset"-Knöpfe des ESP32 ebenso benutzbar bleiben, allerdings nur für den System-Administrator. Deshalb werden hierfür kleine Löcher an der Rückseite des Gehäuses angebracht, dass die Knöpfe ähnlich wie beim SIM Karten Slot eines Handys mit Hilfe eines schmalen Werkzeugs gedrückt werden können. Durch die Platzierung auf der Rückseite sind die Löcher zunächst für die Person, die vor dem Display steht, nicht sichtbar, sondern erst dann erreichbar, wenn das Displaymodul von der Wand genommen wird.

Damit die Software des ESP geflasht werden kann, beispielsweise nach Aktualisierung der Firmware, ohne dabei das Displaymodul auseinanderzubauen, wird an der Seite des Gehäuses eine Bohrung vorgenommen, sodass der USB-C Port des Mikrocontrollers jederzeit zugänglich ist.

Die Halterungen für die Platinen der Beiden Epaper-Displays sind einfache Konstruktionen aus schmalen Stegen, die zylinderförmige Extrusionen besitzen, um die Platinen festzuschrauben. Hierbei unterscheiden sich die beiden Halterungen in ihren Dimensionen, da nicht nur die beiden Displays selbst, sondern auch die zugehörigen Platinen unterschiedliche Maße besitzen.

Die Front des Gehäuses besitzt einen rechteckigen Ausschnitt mit einer Falz nach innen, in die das Display dann gelegt wird. Um das Display vor hineinfallen ins Gehäuse zu schützen, wird einmal die Platinenhalterung direkt an der Rückseitige des Displays angebracht und zusätzlich noch kleine Überhänge konstruiert, die das Display von hinten stützen. Desweiteren ist ein rechteckiger Ausschnitt für den Benutzerknopf in der Front vorhanden.

Da im Laufe der Gehäuseentwicklung Protoypen designt und anschließend getestet werden, entstehen mehrere Entwürfe für die Umsetzung. Die folgenden drei Abbildungen Abbildung 6.22, Abbildung 6.23 und Abbildung 6.24 zeigen die verschiedenen Versionen der Rückseite des 7.3-Zoll-Gehäuses und jeweils die Veränderungen zur vorherigen Version. Die Rückseite des 9.7-Zoll-Gehäuse wurde anschließend nach dem Vorbild des kleineren Gehäuses designt, wie man in Abbildung 6.25 sieht. Die beiden Fronten sind zusammen mit dem Platinenhalter und der Abdeckung des Benutzerknopfes schließlich in Abbildung 6.26 und Abbildung 6.27 zu sehen.

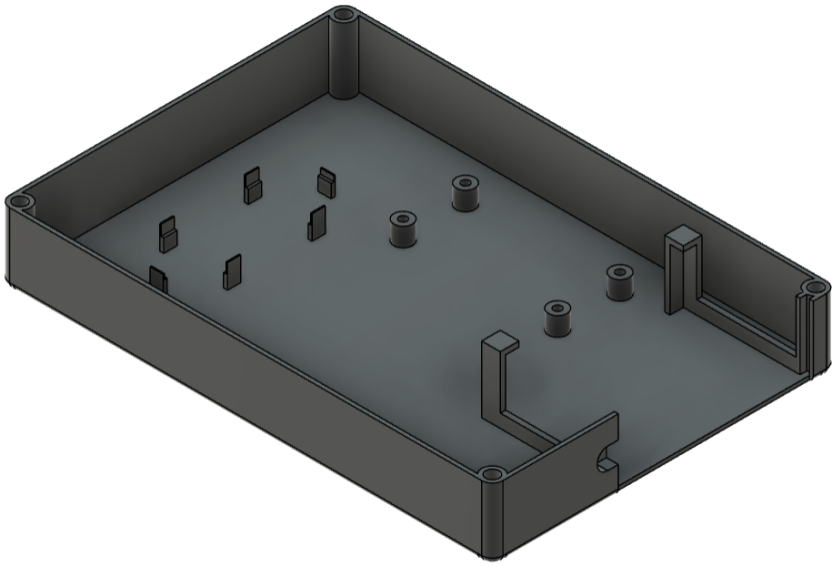


Abbildung 6.22: Die erste Version der 7.3-Zoll-Rückseite.

Nach der ersten Version des Gehäuses wurde das Design nochmal grundlegend verändert und auch die Befestigung der Akkupacks überdacht. Wie in Abbildung 6.22 zu erkennen ist, wurde zunächst vorgesehen, die Akkupacks von außen zugänglich zu machen, um einen leichteren Austausch zu ermöglichen ohne das Gehäuse von der Wand nehmen zu müssen.

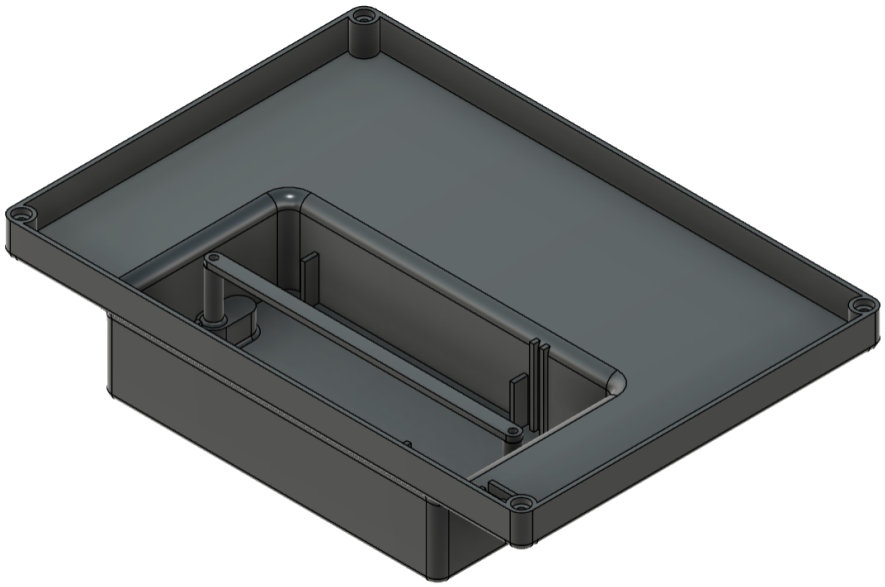


Abbildung 6.23: Die zweite Version der 7.3-Zoll-Rückseite.

In der zweiten Version ist nun schon mehr das finale Design wiederzufinden. Das Gehäuse wurde an den Stellen, wo nicht die Elektronik im inneren behalten wird, auf einen minimale Dicke reduziert und die Akkupacks befinden sich nun im Inneren, sodass das Gehäuse geöffnet werden muss, um diese zu tauschen.

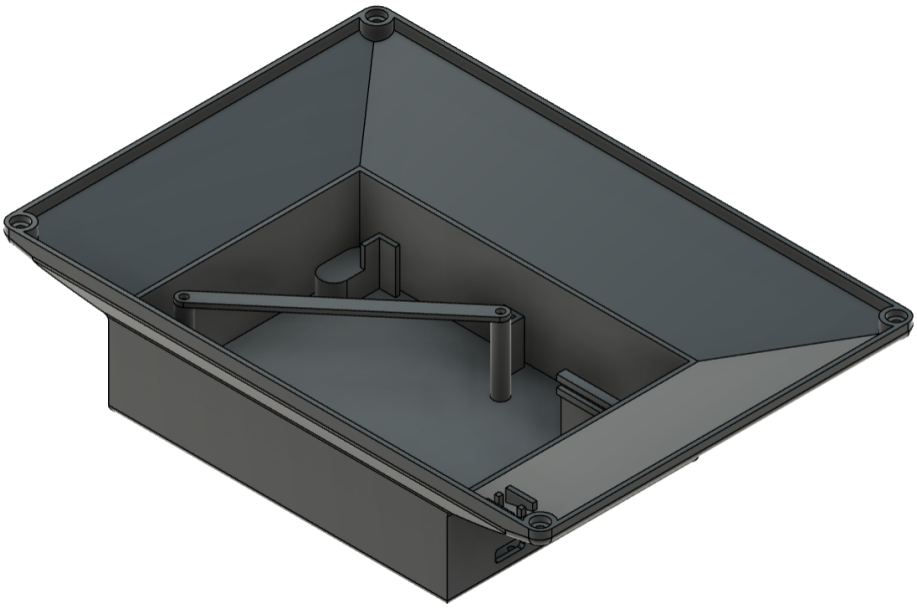


Abbildung 6.24: Die dritte und finale Version der 7.3-Zoll-Rückseite.

In Abbildung 6.24 sieht man nun das endgültige Design der Gehäuserückseite. Im Vergleich zur vorherigen Version wurden die Seiten abgeschrägt, um ein noch schmaleren Eindruck des Gehäuses zu erwecken. Zudem wurden die Aufhängelöcher noch versetzt, um den Schwerpunkt des Gehäuses besser auszugleichen.

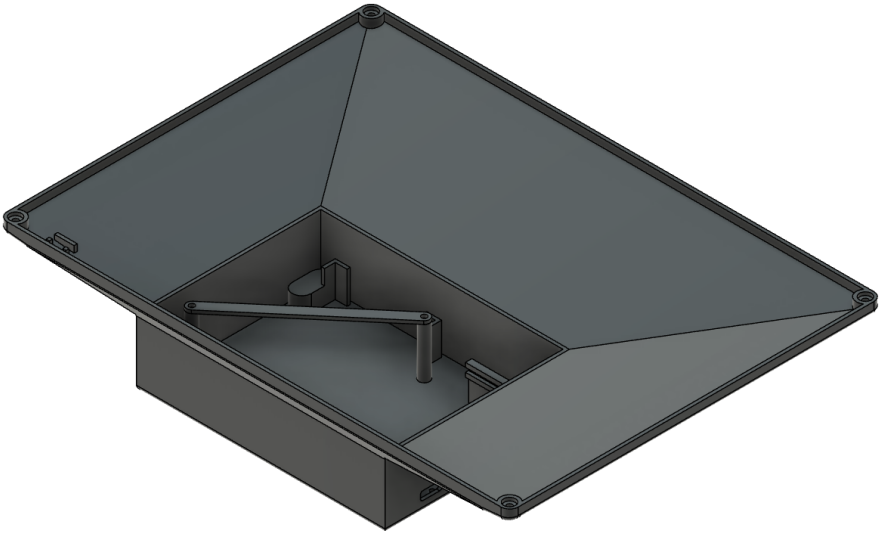


Abbildung 6.25: Die Rückseite des Gehäuses für das 9.7-Zoll-Epaper-Display.

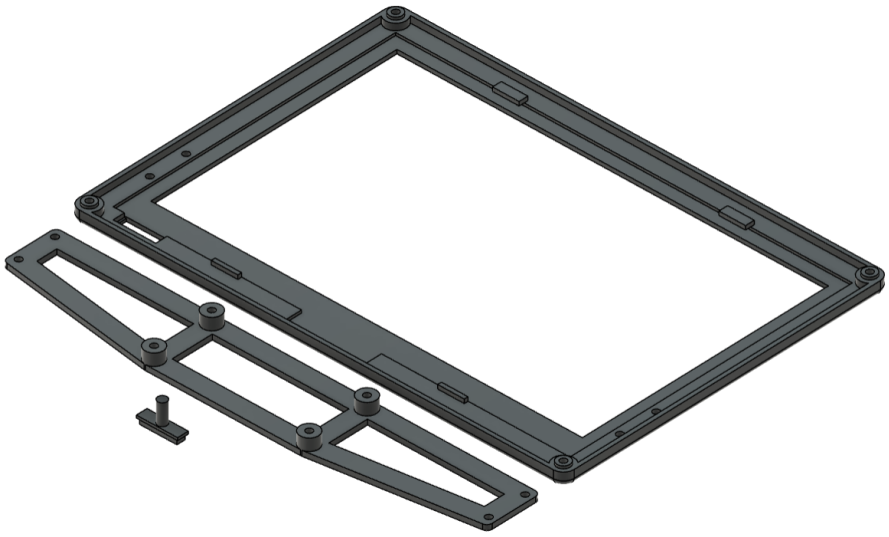


Abbildung 6.26: Die Front des 7.3-Zoll-Gehäuses mit Platinenhalter und Knopfabdeckung.

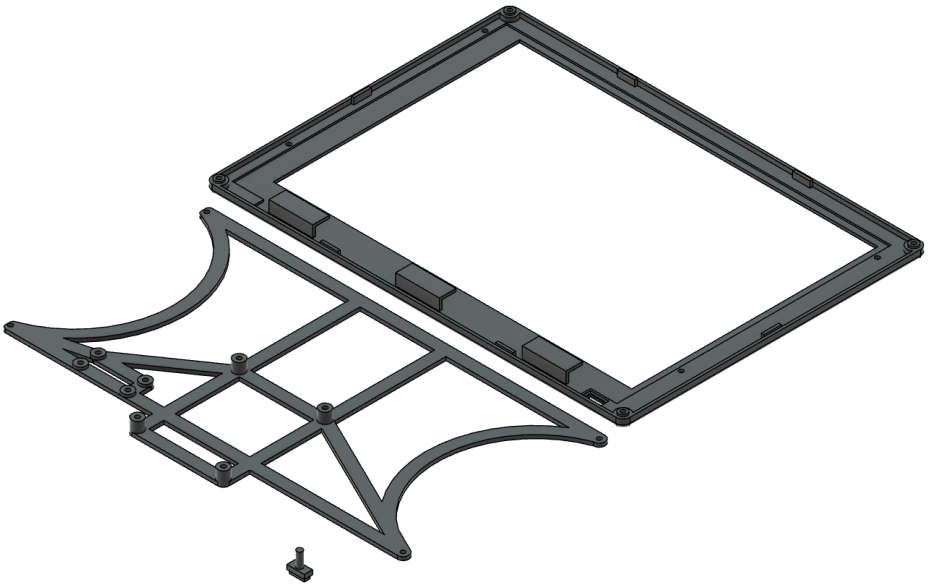


Abbildung 6.27: Die Front des 9.7-Zoll-Gehäuses mit Platinenhalter und Knopfabdeckung

Bei der Front des 9.7-Zoll-Gehäuses wurde an der unteren Kante noch eine extra Stützstruktur angebracht, um das Flachbandkabel des Displays kontrolliert zu biegen und ein Ab- oder Einreißen zu verhindern. Zudem deckt der Platinenhalter im Gegensatz zu dem des kleineren Gehäuses einen größeren Teil des Displays ab, da es hier die einzige Art der Unterstützung des Displays von hinten ist.

6.8.3 Prototyping

Da es sich bei dem LoW Power Raumdisplay um eine Projektarbeit handelt und zunächst keine Serienproduktion vorgesehen ist, beschränkt sich die Entwicklung des Gehäuses auf den Entwurf zweier Prototypen, einmal mit einem 1.3-Zoll-Display und einmal mit einem 9.7-Zoll-Display. Hierbei wird das Rapid-Prototyping-Verfahren des Fused-Deposition-Modelling angewandt, da es neben geringen Kosten auch genug Stabilität für die genannten Anforderungen aufweist. Zudem ist es dem Projektteam möglich,

direkt im eigenen Labor der Technischen Hochschule Augsburg dieses Verfahren anzuwenden. Es werden mehrere sogenannte FDM-Drucker zur Verfügung gestellt, ebenso wie die CAD-Software Inventor-Softwarepaket der Firma Autodesk. [JG_04]

Die verwendeten FDM-Drucker, im Folgenden auch 3D-Drucker genannt, beschränken sich auf die Modelle 2 Extended+, 3 und S5 der Marke UltiMaker. Diese Drucker verwenden Filamente aus Polyactiden, kurz PLA, oder Polyethylenterephtalat mit Glykolmodifikation, auch PETG genannt mit einem Durchmesser von 2,85mm^2. Die Materialien unterscheiden sich nur leicht in ihren Eigenschaften, wobei PETG stabiler und haltbarer ist, PLA dagegen ist hitzebeständiger und biologisch abbaubar.[JG_05] Beide Materialien sind in ihrer Stabilität und Hitzebeständigkeit für den Zweck des Low Power Displays ausreichend.

Nachdem die Modelle der Gehäuse im CAD Programm entworfen wurden, werden sie als .stl Dateien in einen sogenannten Slicer exportiert. Ein Slicer übernimmt die Aufgabe des Slicings, wobei ein 3D-Modell in einen für 3D Drucker verständlichen Maschinencode Namens G-Code umgewandelt wird. Zusätzlich können verschiedene Einstellungen vorgenommen werden, die die Druckqualität beeinflussen, wie beispielsweise Druckgeschwindigkeit oder Schichtdicke. [JG_06]

Bei diesem Projekt wurde die Slicing Software Cura verwendet, welche ebenfalls von der Marke UltiMaker stammt. Als Schichthöhe werden Werte zwischen 0.1 und 0.2 mm bevorzugt. Bei der Schichtdicke gilt, je kleiner, desto besser ist die Oberflächengüte des Modells, allerdings wird dann auch mehr Zeit für den Druck benötigt.[JG_06]

Ein weiterer einstellbarer Wert der mit der Schichtdicke zusammenhängt, ist die Dicke des Bodens und der Decke. Hier wird festgelegt, wie viele Schichten flächendeckend gedruckt werden sollen, bevor die Füllung des Modells beginnt. Ein ähnlicher Wert ist die Wandstärke, wobei hier die Anzahl der vertikalen Schichten ausgewählt wird.[JG_06]

Bei der Füllung von Hohlräumen des Modells kann aus der Art und geometrischen Form der Füllung und der prozentualen Menge gewählt werden. Zudem kann noch die Art der Stützen bei Überhängen, sowie die Platzierung und der Winkel des Überhangs gewählt werden, der noch ohne Stütze gedruckt werden kann, bestimmt werden.[JG_06]

Um zu einen besseren Halt der Modells auf der Grundfläche des Drucker zu gewährleisten, gibt es verschieden Möglichkeiten zur Auswahl. Ein sogenannter Brim fügt beim Drucken den Außenkanten der Grundfläche eine einzelne Schicht bestimmter Breite hinzu, was die Verbindungsfläche zwischen Druck und Grundfläche erhöht. Ein Raft ist eine Art Polster zwischen dem gesamten Modell und der Druckplatte, was ein nachträgliches Verziehen des Modells durch Erkaltung des Materials verhindert. Eine letzte Variante ist ein Skirt, eine Linie auf der ersten Schicht um die Konturen der Grundfläche. Da diese nicht mit dem Modell verbunden ist, hat diese Variante keinen Einfluss auf Stabilität oder Verziehen des Drucks.[JG_06]

Zusätzlich zu den bisher genannten Einstellungen, können noch weitere Parameter für den Druck bestimmt werden, diese werden jedoch nicht genauer erläutert und es werden die gegebenen Standardwerte der Slicer-Software genutzt. Die gewählten Werte der genannten Steuergrößen können der Tabelle 6.1 entnommen werden.

Parameter	Wert
Schichthöhe	0.2 mm
Boden-/Deckenstärke	1.2 mm
Wandstärke	0.8 mm
Füllung	Gyroid, 20%
Stützsturktur	Baum, überall, 60°
Verbdg Druckplatte	Skirt

Tabelle 6.1: Parameterwerte für 3D Druck der Gehäuseprototypen.

Nach dem Slicen wird der G-Code an den 3D-Drucker übergeben. Hier muss dann nur noch das passende Filament einsetzen und der Druck kann starten. Nach dem Ablösen des fertigen Modells von der Druckplatte wird teilweise noch mit Hilfe eines Skalpells nachgearbeitet, um die Ungenauigkeiten des Druckers auszugleichen. Anschließend werden die Teile auf Passgenauigkeit für die Komponenten getestet und dann die bereits oben genannten Änderungen vorgenommen. In Abbildung 6.28 sieht man den Zwischenstand eines Drucks. Neben dem Gehöuse selbst ist auch die Stützstruktur in Form der Bäume sehr gut zu erkennen.

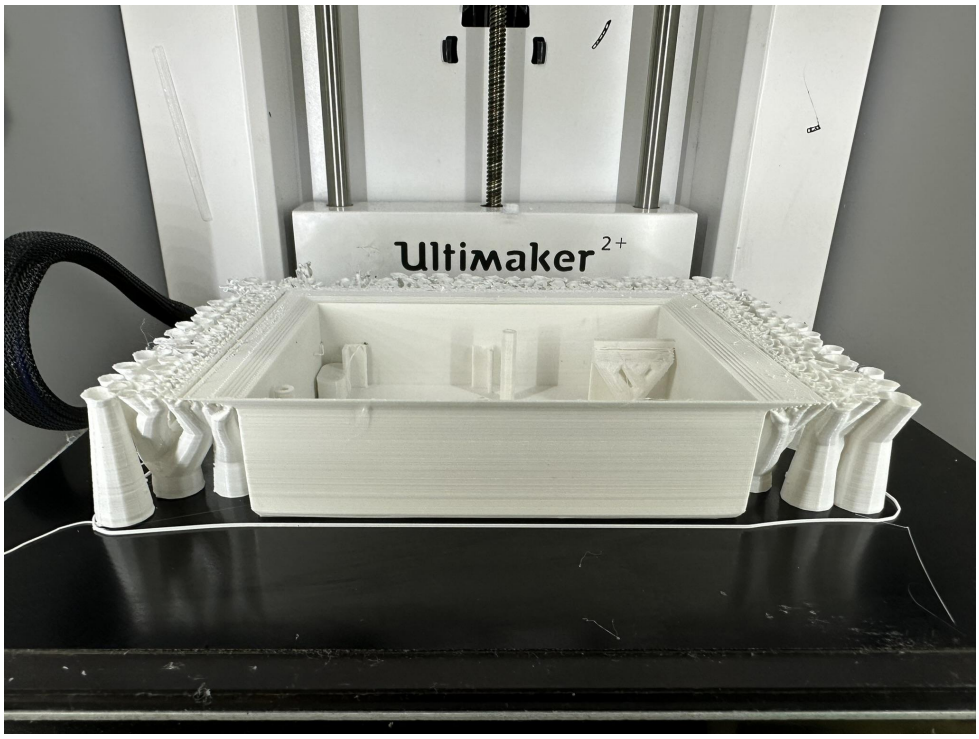


Abbildung 6.28: Zwischenstand beim Drucken des 7.3-Zoll-Gehäuses.

Nach dem Fertigen des finalen Prototyp Gehäuses werden die elektronischen Komponenten eingesetzt und das Displaymodul zusammengebaut. Die fertigen Prototypen kann man in Abbildung 6.29 und Abbildung 6.30 betrachten.

Abbildung 6.29: Das finale Prototyp-Gehäuse des 9.7-Zoll-Gehäuses.

Abbildung 6.30: Das finale Prototyp-Gehäuse des 7.3-Zoll-Gehäuses.

6.9 Strommessung von Mikrocontroller und Display

6.9.1 Versuchsaufbau

Mario Wegmann

Der Prozess, um ein Bild per WLAN zu empfangen, zu verarbeiten und es auf einem ePaper Display darzustellen, ist sehr umfangreich und während der verschiedenen Phasen ist der Stromverbrauch sehr schwankend. Daher kann mit einer statischen Strommessung kein aussagekräftiges Messergebnis produziert werden. Um den Strom dynamisch zu messen und dabei schnelle Änderungen sichtbar zu machen, eignet sich daher ein Oszilloskop als Messinstrument. Da das Oszilloskop jedoch nur Spannungen messen kann, muss hier der Umweg über einen Shunt-Widerstand gemacht werden. Der Shunt-Widerstand wird dabei in Reihe zwischen der zu messenden Last und der Masse geschaltet, mit einem Tastkopf an beiden Enden des Shunt-Widerstands verbunden, kann nun der Spannungsabfall über den Shunt-Widerstand gemessen werden. Abschließend kann über das Ohm'sche Gesetz aus dem Widerstandswert und der Spannung der durchflossene Strom berechnet werden. Viele Oszilloskope bieten daher auch die Möglichkeit an, die Achsenbeschriftung auf mA umzuschalten, um Messergebnisse mit der korrekten Einheit festhalten zu können. Es muss jedoch beachtet werden, dass das Oszilloskope nicht automatisch das Ohm'sche Gesetz anwendet, da es keine Kenntnis über den Widerstandswert hat. Somit muss dies bei den Messergebnissen mit verrechnet werden oder alternativ darauf geachtet werden, dass der Widerstand wert des Shunt-Widerstands 1 Ω so genau wie möglich erreicht. Des Weiteren ist unbedingt darauf zu achten, dass der Shunt-Widerstand zwischen Last und Masse hängt, wenn es sich um eine netzbetriebene Schaltung handelt. Zwar wäre das Messergebnis theoretisch auch korrekt, jedoch würde durch die Erdung der Masse am Tastkopf ein Kurzschluss entstehen, wenn die Masse vom Tastkopf mit einem anderen Spannungspegel, als der Masse der Schaltung, verbunden wird. Dieser Kurzschluss könnte das Oszilloskop beschädigen.

In Abbildung 6.31 erkennt man den Versuchsaufbau für die Messung. Als Spannungsquelle dient ein Labornetzteil. Vom Pluspol des Labornetzteils führt ein Draht zum V_{CC} Pin des ESP32. Von GND des ESP32 führt ein Draht zum 1 Ω Shunt Widerstand. An diesem ist auch der Tastkopf des Oszilloskopes angeschlossen. Nach dem Shunt Widerstand führt ein Draht zu einem Multimeter. Das Multimeter dient zur Kontrolle, ob die vom Oszilloskop ausgegebenen Werte plausibel sind. Abschließend ist der zweite Kontakt vom Multimeter mit dem Minuspol des Labornetzteils verbunden.

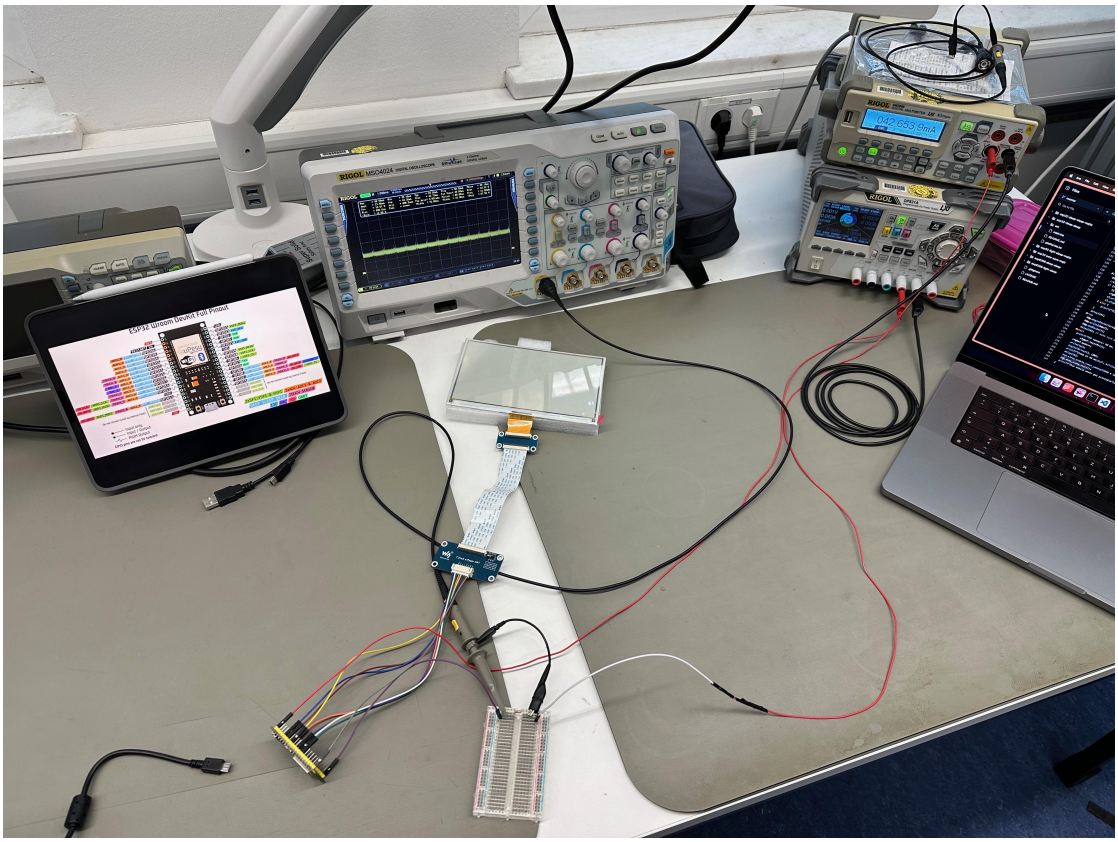


Abbildung 6.31: Der Versuchsaufbau mit Oszilloskop, Multimeter und Labornetzteil

6.9.2 Messergebnisse

Mario Wegmann

In Tabelle 6.2 kann man einen kompletten Refresh des 7,3 Zoll großen Displays erkennen. Hierbei wird auf dem ePaper Display zuerst weiß gestellt, anschließend wird nach einer Pause eine in der Firmware hart codierte Bitmap auf dem Display angezeigt. Zum Abschluss wird erneut ein weißes Bild dargestellt. In diesem Teilversuch wurde der Stromverbrauch vom WLAN noch nicht

gemessen. Für diesen Versuch wurde Democode von Hersteller WaveShare verwendet, welcher in der Quelle zu finden ist [MW_09].

Bereich	Zustand	Avg. Strom [mA]	Zeit [s]
1.	Bitmap an ePaper Display übertragen	66,13	1,032
2.	Bildschirminhalt leeren	64.05	1,624
3.	Bildschirminhalt leeren	74,64	2,608
4.	Bildschirminhalt leeren	73,67	2,689
5.	Bildschirminhalt leeren	66,06	3,675
6.	Bildschirminhalt leeren	62,05	2,214
7.	Delay	48,99	0,994
8.	Bitmap an ePaper Display übertragen	66,77	2,196
9.	Bitmap darstellen	70,51	1,847
10.	Bitmap darstellen	82,34	2,653
11.	Bitmap darstellen	82,03	2,703
12.	Bitmap darstellen	70,94	3,695
13.	Bitmap darstellen	66,15	2,009
14.	Delay	49,19	3,001
15.	Bitmap an ePaper Display übertragen	67,69	2,021
16.	Bildschirminhalt leeren	62,90	1,785
17.	Bildschirminhalt leeren	73,83	2,728
18.	Bildschirminhalt leeren	73,51	2,604
19.	Bildschirminhalt leeren	65,99	3,646
20.	Bildschirminhalt leeren	61,93	2,120
21.	Delay	49,09	2,008
22.	Deep-sleep	9,777	5,001

Tabelle 6.2: Der Stromverbrauch aufgeteilt in Zeitabschnitte beim 7,3" Display

Benjamin Klaric

In Abbildung 6.32 sind die geplotteten Werte von der Tabelle 6.2 graphisch dargestellt.

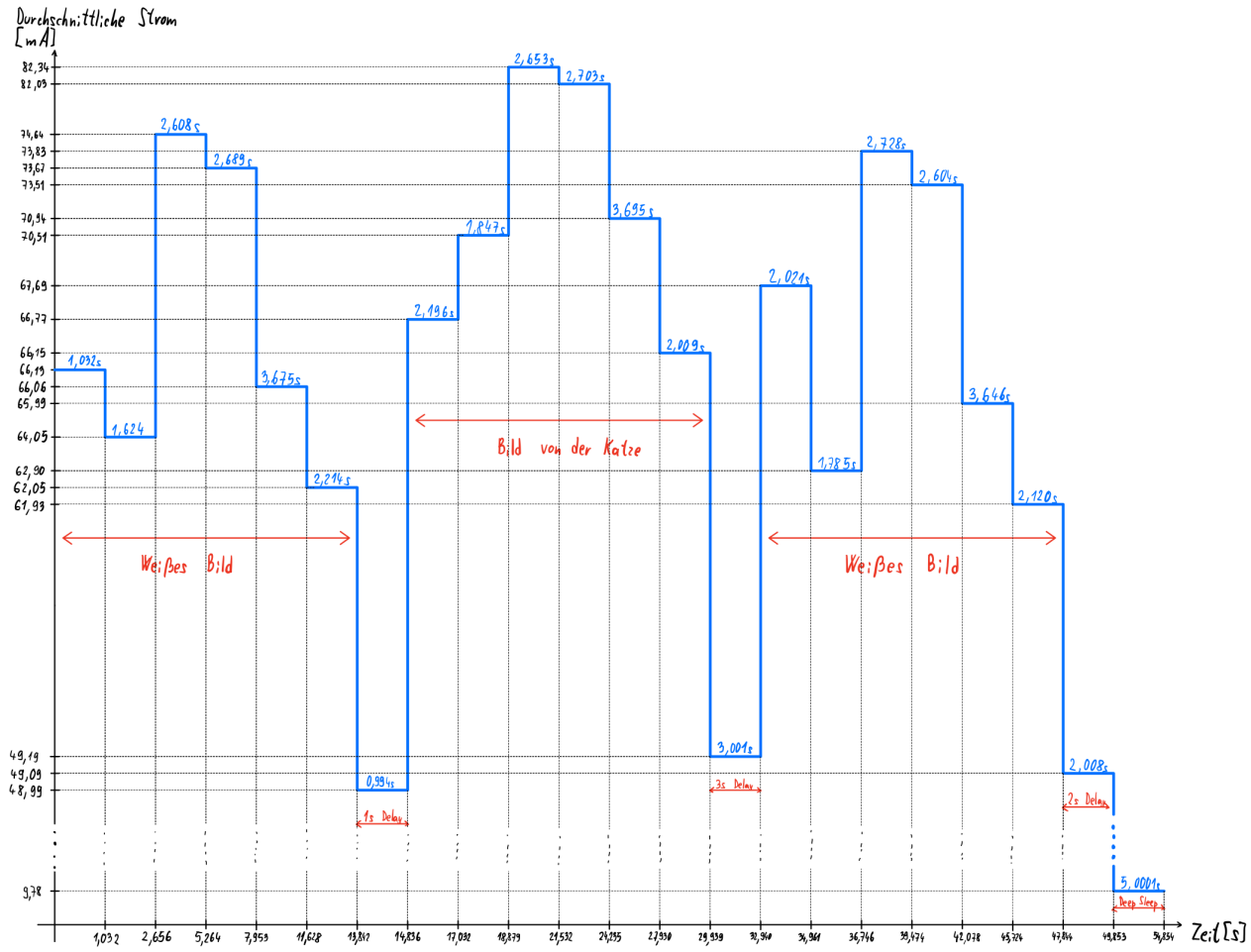


Abbildung 6.32: Plot von dem Stromvebrauch in Zeitabschnitten beim 7,3" Display

Mario Wegmann

In Tabelle 6.3 sieht man die Messwerte vom Stromverbrauch beim 9,7 Zoll Display. Hierbei wird der ESP32 und das zugehörige Treiber-Board mit Strom versorgt, anschließend das Treiber-Board initialisiert und nach den Eigenschaften des angeschlossenen ePaper Displays gefragt. Erst danach wird die Bitmap zuerst an das Treiber-Board übertragen und dann vom Treiber-Board auf das Display. Abschließend wird das Treiber-Board in den Idle wieder gesetzt und zum Schluss der ESP32 zusätzlich in den Deep-sleep. Für diesen Versuch wurde Democode von GitHub verwendet, welcher in der Quelle zu finden ist [MW_10].

Bereich	Zustand	Avg. Strom [mA]	Zeit [s]
1.	Idle mit angeschalteten Treiber-Board	67,54	0,929
2.	Init des Treiber-Boards	94,87	0,948
3.	Auslesen der Informationen des angeschlossenen ePaper Displays	112,2	0,497
4.	Übertragen der Bitmap über SPI an das Treiberboard	103,4	0,460
5.	Übertragen der Bitmap an das Display	105,2	0,570
6.	Bildschirminhalt auf dem ePaper Display darstellen	109,9	0,865
7.	Idle mit angeschalteten Treiber-Board	66,48	0,954
8.	Deep-sleep angeschalteten Treiber-Board	29,22	5,002

Tabelle 6.3 Der Stromverbrauch aufgeteilt in Zeitabschnitte beim 9,7" Display

Benjamin Klaric

In Abbildung 6.33 sind die geplotteten Werte von der Tabelle 6.3 graphisch dargestellt.

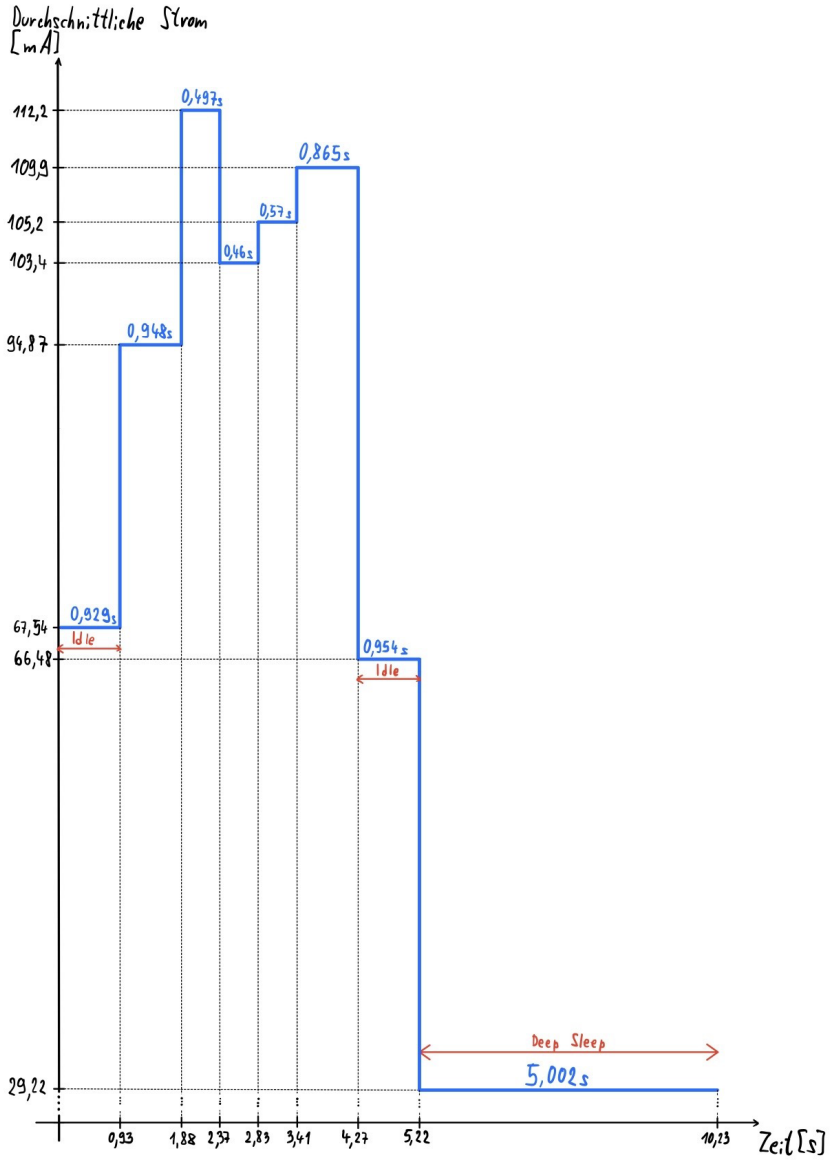


Abbildung 6.33: Plot von dem Stromvebrauch in Zeitabschnitten beim 9,7" Display

Mario Wegmann

In Tabelle 6.4 sind die Messwerte vom Stromverbrauch bei der Verwendung des ESP32 WLAN-Modems erkennbar. Hierbei ist zu beachten, dass der Verlauf nicht periodisch verläuft, sondern die relevanten Bereiche innerhalb einer Übertragung angegeben sind.

Bereich	Zustand	Avg. Strom [mA]	Zeit [s]
1.	ESP im Idle, WLAN Modem ausgeschaltet	49,28	5,001
2.	ESP im Idle, WLAN Modem eingeschaltet	121,64	1,004
3.	WLAN Modem aktiv (Pakettransport)	603,44	$76 * 10^{-6}$

Tabelle 6.4: Die relevanten Stromverbrauchsbereiche bei der Verwendung vom WLAN-Modem

6.9.3 Interpretation

Benjamin Klaric

Der Stromverbrauch sowohl des 7,3-Zoll- als auch des 9,7-Zoll-Displays ist während eines vollen Refreshs angemessen. Es gibt jedoch einige Dinge zu berücksichtigen, nämlich dass ein voller Refresh nicht jedes Mal durchgeführt werden muss und dass der Stromverbrauch in Verbindung mit der Wi-Fi-Nutzung nicht in dieser Konfiguration gemessen wurde.
Die Erklärung dafür, dass nicht bei jeder Gelegenheit ein voller Refresh erforderlich ist, liegt an den Spezifikationen des Displays. Gemäß diesen Spezifikationen soll das Display nach einer längeren Zeit ohne Refresh einen vollständigen Refresh durchführen. [BK_22] Dies reduziert die Zeitspanne, in der das System unter Volllast arbeitet, und damit auch den durchschnittlichen Verbrauch.
Der Wi-Fi-Verbrauch wurde nicht berücksichtigt, da dieser von vielen Parametern abhängt, die zum Zeitpunkt der Strommessung noch nicht optimiert waren. Dazu gehören die Code-Optimierung, die Steuerung der Wi-Fi-Nutzung und die Tatsache, dass das Modul, das für die Strommessung verwendet wurde, später nicht mehr genutzt wurde aufgrund des hohen Stromverbrauchs im Deep Sleep Modus.
Im nächsten Abschnitt erfolgt ein genauerer Blick auf die gemessenen Werte und deren Bedeutung.

Stromverbrauch von 7,3" Display
Das 7,3" Display hat eine längere Refresh-Zeit, was darauf zurückzuführen ist, dass es vier anstatt zwei Farben darstellen kann. Diese zusätzliche Farbunterstützung führt zu einer längeren Arbeitszeit und somit zu einem erhöhten Stromverbrauch. Der durchschnittliche Stromverbrauch des Displays kann anhand des Plots aus Abbildung 6.32 berechnet werden. Dieser liegt bei 62 mA über einen Zeitraum von 54,853 Sekunden, wobei der Verbrauch stark von der Länge der Arbeitszeit abhängt.
Durch den Verzicht auf einen vollständigen Refresh und die Nutzung eines Mikrocontrollers wie dem XIAO ESP32-S3, der im Deep Sleep Modus einen sehr geringen Stromverbrauch von 14 µA im Vergleich zu 9,78 mA aufweist, kann der durchschnittliche Stromverbrauch des Systems signifikant reduziert werden. Dies führt zu einer effizienteren Nutzung und einer Verringerung des Gesamtenergiebedarfs über längere Zeiträume.

Stromverbrauch von 9,7" Display
Das 9,7-Zoll-Display zeichnet sich durch eine sehr kurze Refresh-Zeit aus, was jedoch zu einem höheren Stromverbrauch bei jedem Refresh führt. Über einen Zeitraum von 4,885 Sekunden beträgt der durchschnittliche Stromverbrauch 97 mA. Dieser erhöhte Durchschnittsverbrauch wird durch die verkürzte Betriebszeit ausgeglichen.
In Tabelle 6.3 wird verdeutlicht, dass trotz des höheren Stromverbrauchs des 9,7-Zoll-Displays der Gesamtenergieverbrauch aufgrund der schnellen Refresh-Zeit kompensiert wird.

Stromverbrauch von WLAN

Mario Wegmann

Wie in der Tabelle 6.4 ersichtlich, fließen bei der Verwendung vom WLAN-Modem deutlich höhere Ströme als beim reinen Betrieb mit einem Display. Jedoch ist auch erkennbar, dass gerade die aperiodischen Spitzen beim Senden mit 76 µS verhältnismäßig sehr kurz sind. Man kann somit den Beitrag der Spannungen vernachlässigen und von einem Durchschnittsverbrauch von circa 121 mA ausgehen, wenn das WLAN-Modem aktiv ist und Daten übertragen werden sollen. Dennoch empfiehlt es sich, die Nutzung vom WLAN-Modem möglichst gering zu halten.

6.10 A-D-Wandler Messung

Benjamin Klaric

Wie bereit in 6.5 Schaltungsentwurf unter **Akkuanschlüsse** erwähnt, wurde ein analoger GPIO-Pin des XIAO ESP32-S3 verwendet, um den Spannungspegel des Akkupacks zu überwachen. Der XIAO ESP32-S3 bietet viele analoge Pins, wovon einer, nämlich der A9-Pin, für die oben genannte Überwachung genutzt wird. Dazu wird an diesem Pin ein Spannungsteiler aufgebaut, bei dem zwei 220 kΩ Widerstände parallel zu den Akku+ und Akku- Eingängen des Mikrocontrollers verbunden werden. Der Wert für die Widerstände wurde so gewählt, weil der Spannungsteiler immer Strom zieht, auch wenn sich der Mikrocontroller im Deep Sleep befindet, da die Widerstände direkt mit dem Akkupack verbunden sind. Mit zwei 220 kΩ Widerständen und einer Spannung zwischen 3,5V und 4,2V ergibt sich ein Verbrauch zwischen 8 µA und 9,5 µA gemäß dem Ohm'schen Gesetz, was akzeptabel ist. Allerdings muss man darauf achten, dass die Widerstandswerte möglichst nah beieinander liegen, damit sich die

Spannung möglichst genau halbiert. Falls die Widerstände nicht exakt gleich sind, kann man den Multiplikationsfaktor so anpassen, dass die ungenaue Aufteilung der Spannung berücksichtigt wird, indem man die Spannungsteiler-Gleichung aufstellt und anpasst, wie in Abbildung 6.34 zu sehen ist. Dabei entspricht das Widerstandsverhältnis dem Multiplikationsfaktor.

$$V_{akku} = V_{gemessen} \cdot \frac{R2}{R1+R2} \quad \rightarrow \quad V_{gemessen} = V_{akku} \cdot \frac{R1+R2}{R2}$$

Abbildung 6.34: Umgeformte Spannungsteilerformel

Die Messung muss man auch in Code implementieren, wobei von dem Hersteller der Code zur Verfügung gestellt ist. [BK_23] Nach der Testmessung wurde festgestellt, dass die Messung nicht ganz genau ist, mit einem Fehler von etwa 100 mV. Dies könnte auf mehrere Gründe zurückzuführen sein, die sich aus dem Prinzip der Messung über einen analogen Pin mit Analog-Digital-Wandler (A-D-Wandler) ergeben. Der A-D-Wandler im Mikrocontroller verwendet einen Abtastmechanismus, der kleine, schnelle Stromstöße verursacht. Diese können zu momentanen Spannungsabfällen aufgrund des hohen Widerstands der Spannungsteilerwiderstände führen. Daher beeinflussen hochfrequentes Rauschen oder Schwankungen in der Batteriespannung direkt den gemessenen Wert. Bei der Abtastung wird ein interner Kondensator kurzzeitig mit dem analogen Eingangspin verbunden. Dieser interne Kondensator muss sich schnell auf die Eingangsspannung aufladen. Wenn der Widerstand des Spannungsteilers hoch ist, kann er sich möglicherweise nicht schnell genug aufladen, was zu Messfehlern führt. Um dieses Problem zu lösen, wurde ein Pufferkondensator von 100 nF parallel zu Akku+ und Akku- hinzugefügt. Dieser Wert ist ein typischer Wert für das Puffern in solchen Anwendungen. Dieser Kondensator fungiert als kleiner Energiespeicher, der dem A-D-Wandler die erforderliche Stromversorgung ohne signifikante Spannungsabfälle ermöglicht. Darüber hinaus filtert er hochfrequentes Rauschen heraus und reduziert die Auswirkungen von Schwankungen in der Batteriespannung. Wenn man Abildung 6.35 betrachtet, ist eine verbesserte und stabilere Kurve der Messdaten mit dem Kondensator zu erkennen. Für jeden Spannungspegel wurden 50 Datenpunkte sowohl mit als auch ohne Kondensator gemessen.

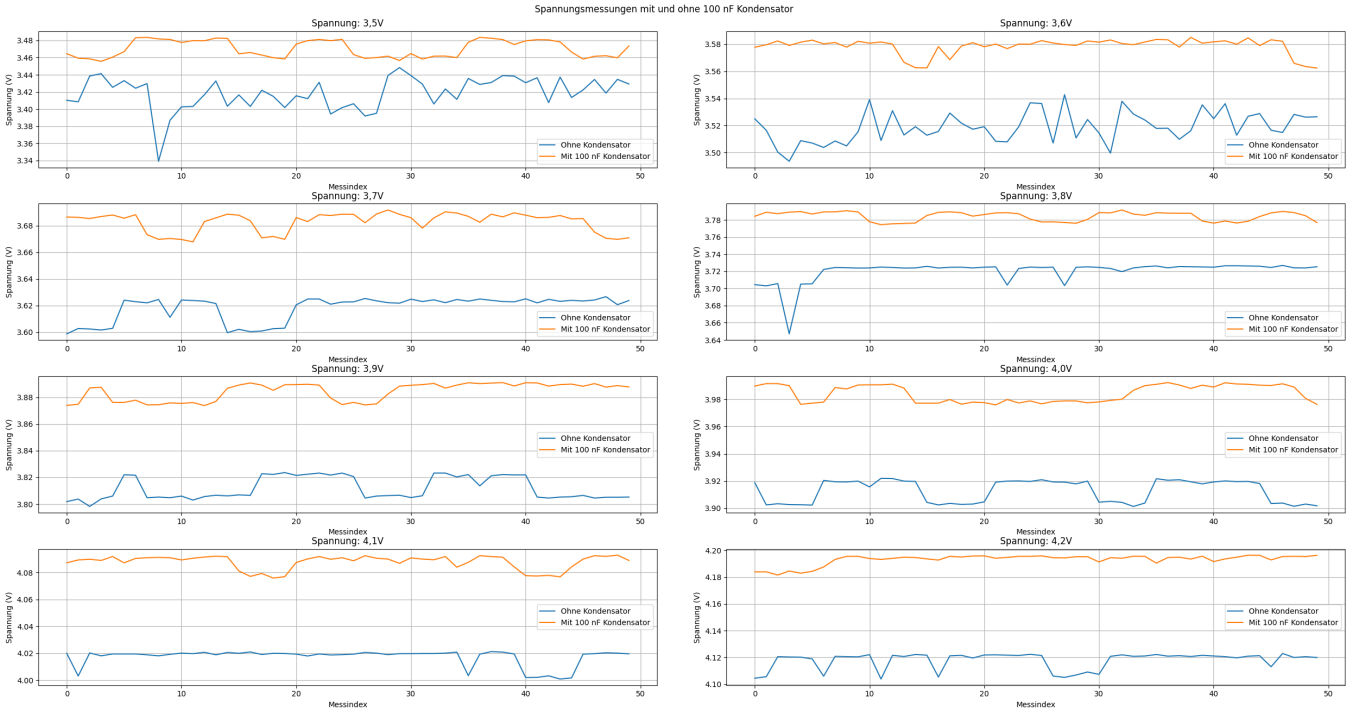


Abbildung 6.35: Plot von 50 Messwerten bei jedem Spannungspegel

Allerdings besteht trotz der Verringerung des Fehlers noch ein Restfehler. Dies liegt, wie bereits erwähnt, an den hohen Widerstandswerten, die aufgrund des hohen Stromverbrauchs nicht weiter reduziert werden können. Die Werte zeigen weniger Spitzen und bleiben stabiler bei einem bestimmten Wert, was von Vorteil ist. Zusätzlich wird in der Software ein kleiner Offset hinzugefügt.
Eine andere Art der Messung könnte in Betracht gezogen werden, nämlich nicht die genaue Messung des aktuellen Spannungswerts, sondern die Messung der Änderung in Bezug auf den zu Beginn gemessenen Wert. Genauer gesagt könnte zu

Beginn der Spannungswert des Akkupacks mit einem A-D-Wandler gemessen werden, und dann könnte der Abfall in Bezug auf diesen Wert überwacht werden, anstatt den aktuellen Spannungswert zu messen. Möglicherweise könnte dies bessere Ergebnisse liefern. Es ist bekannt, dass die Spannung bei einem voll geladenen Akkupack von 4,2V auf 3,5V abfallen darf, also um 0,7V. Dieser Abfall könnte auch bei der A-D-Wandler-Messung überwacht werden. Leider war diese Art von Messung aus Zeitgründen nicht ausgetestet.

7. Firmware

7.1 Entwicklungsumgebung

Ahmet Emirhan Göktas

PlatformIO wurde als Entwicklungsumgebung für die LPRD-Firmware ausgewählt. Da die Arduino IDE für die Komplexität des Projekts nicht geeignet ist, wird PlatformIO als fortschrittlichere Alternative verwendet.

7.1.1 PlatformIO mit Visual Studio Code

Ahmet Emirhan Göktas

Wie im Abschnitt Grundlagen erwähnt, kann PlatformIO mit verschiedenen IDEs verwendet werden. Die meisten unserer Teammitglieder nutzen Visual Studio Code als ihre IDE. Daher möchten wir zeigen, wie man PlatformIO für Visual Studio Code einrichtet.

Installation

Ahmet Emirhan Göktas

1. Installieren Sie Visual Studio Code.
2. Öffnen Sie das Erweiterungspanel entweder über die Seitenleiste oder durch Drücken von `^ Ctrl` + `⇧ Shift` + `X` und geben Sie `PlatformIO IDE` ein.
3. Klicken Sie auf die `Install`, um die PlatformIO IDE-Erweiterung zu installieren.
4. Nachdem die Erweiterung installiert ist, können Sie die PlatformIO-Seitenleiste öffnen, indem Sie auf das PlatformIO-Symbol in der Seitenleiste klicken. Dies kann einige Minuten dauern, da PlatformIO das Backend herunterlädt und die Umgebung einrichtet. [AEG_06]

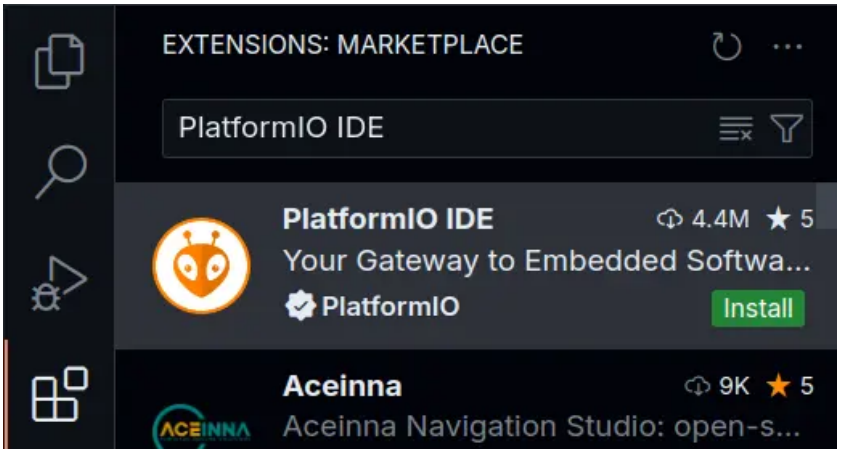


Abbildung 7.1: PlatformIO-Erweiterung in Visual Studio

Projekt erstellen

Ahmet Emirhan Göktas

1. Öffnen Sie die VS Code-Befehlszeile durch Drücken von `^ Ctrl` + `⇧ Shift` + `P` und geben Sie `>PlatformIO:PlatformIO Home` ein.
2. Drücken Sie `Enter`, um die PlatformIO-Startseite zu öffnen.
3. Klicken Sie auf der linken Seite der PlatformIO-Startseite auf die Registerkarte `Projects`.
4. Klicken Sie auf die `New Project`.
5. Wählen Sie das Board, das Sie verwenden. In unserem Fall verwenden wir das Board `Seeed Studio XIAO ESP32S3`.
6. Wählen Sie das Framework. In unserem Fall verwenden wir das `Arduino`-Framework.
7. Drücken Sie abschließend die `Finish`, um das Projekt zu erstellen. [AEG_07]

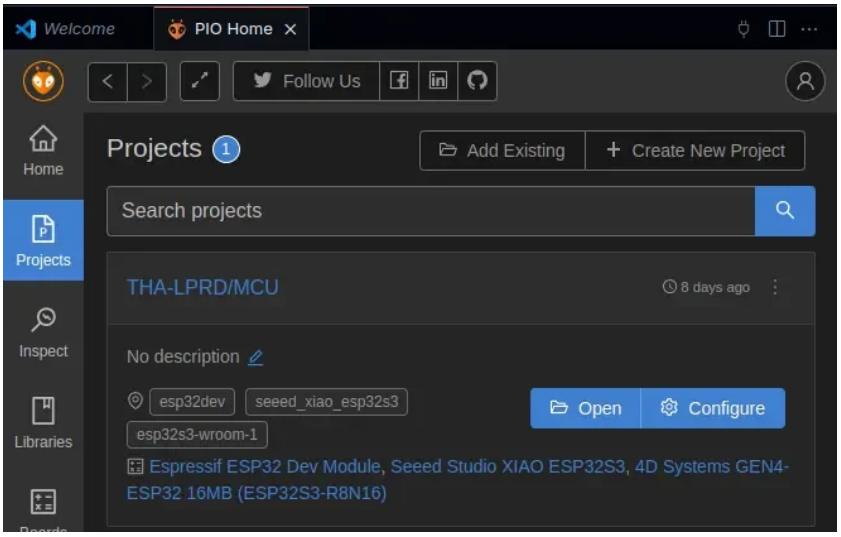


Abbildung 7.2: Neues PlatformIO-Projekt erstellen

Mikrocontroller flashen

Ahmet Emirhan Göktas

1. Verbinden Sie den Mikrocontroller mit Ihrem Computer.
2. Öffnen Sie die PlatformIO-Seitenleiste, indem Sie auf das PlatformIO-Symbol in der Seitenleiste klicken.
3. Klicken Sie auf die Projektumgebung im Abschnitt `Project Tasks`.
4. Klicken Sie auf die `Upload` im Abschnitt `General`, um die Firmware auf den Mikrocontroller zu flashen.
5. Die `Upload File System Image` Taste im Abschnitt `Platform` kann verwendet werden, um das Dateisystem-Image auf den Mikrocontroller hochzuladen.

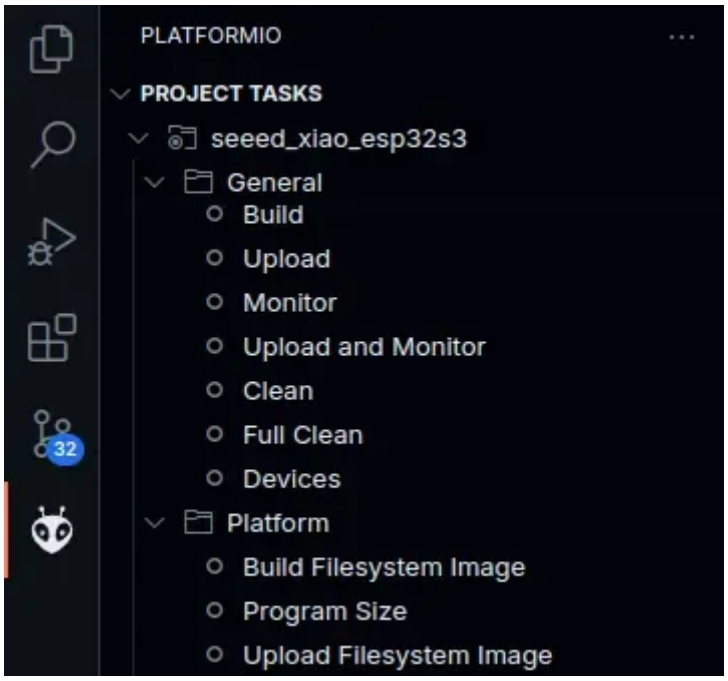


Abbildung 7.3: Firmware auf den Mikrocontroller hochladen

7.1.2 Versionskontrolle

Ahmet Emirhan Göktas

In diesem Projekt haben wir git als unser Versionskontrollsystem verwendet und dem Git-Flow-Modell gefolgt. Das Git-Flow-Modell ist ein von Vincent Driessen erstelltes Branching model für git. Es ist eine Reihe von Regeln, die verwendet werden, um die Verzweigungen in einem git-Repository zu verwalten. Die haupt Branches sind `master` und `dev`. Der `master`-Branch wird für produktionsbereiten Code verwendet, während der `dev`-Branch für Code verwendet wird, der sich in der Entwicklung befindet. Der `dev`-Branch wird abgezweigt, um Feature-Branches zu erstellen. Sobald das Feature abgeschlossen ist, wird der Feature-Branch wieder in den `dev`-Branch integriert. Wenn der Code im `dev`-Branch bereit für die Produktion ist, wird er in den `master`-Branch integriert. [AEG_08]

7.2 Anforderungen

Ahmet Emirhan Göktas

Wie bereits erwähnt, muss das LPRD bestimmte Anforderungen erfüllen. Diese Anforderungen bilden die Grundlage für die Firmware-Entwicklung. Die folgenden Anforderungen ergeben sich aus den Hardwareanforderungen:

7.2.1 Drahtlose Kommunikation

Ahmet Emirhan Göktas

Das Gerät soll drahtlos gesteuert werden, um eine einfache Installation und Konfiguration zu ermöglichen und die Steuerung des Geräts von einem zentralen Server aus zu erlauben. Es gibt zwei Möglichkeiten, dies zu erreichen:

- Verbindung mit einem bestehenden drahtlosen Netzwerk.
- Erstellung eines eigenen drahtlosen Netzwerks.

7.2.2 Dateisystem

Ahmet Emirhan Göktas

Das Displaymodul benötigt einen nichtflüchtigen Speicher, um die angezeigten Bilder sowie die Konfigurationen wie Netzwerkeinstellungen und Betriebsmodi zu speichern. Der ESP32-S3-Mikrocontroller bietet 8 MB Flash-Speicher, der mit dem LittleFS-Dateisystem genutzt wird.

7.2.3 Konfiguration

Ahmet Emirhan Göktas

Das Displaymodul ist flexibel gestaltet und hat keine fest kodierte Konfiguration in der Firmware. Die Konfiguration kann vom Benutzer geändert werden und umfasst:

- WLAN-Netzwerkkonfiguration (SSID, Passwort)
- E-Paper-Display-Konfiguration (Modell)
- Betriebsmodus (Standalone, Netzwerk, Server)
- Serverkonfiguration (URL)
- Log-Level (für Debugging)
- HTTP-Server-Konfiguration (HTTP, HTTPS)
- Authentifizierung (Benutzername, Passwort)

7.2.4 Betriebsmodi

Ahmet Emirhan Göktas

Das Displaymodul unterstützt verschiedene Betriebsmodi:

- **Standalone**: Funktioniert ohne ein bestehendes WLAN-Netzwerk.
- **Netzwerk**: Funktioniert mit einem bestehenden WLAN-Netzwerk.
- **Server**: Wird von einem zentralen Server gesteuert.
- **Default**: Der Standardmodus bei der ersten Inbetriebnahme oder bei nicht behebbaren Fehlern. Dieser Modus entspricht dem Standalone-Modus mit fest kodierten Einstellungen.

7.2.5 SPI-Schnittstelle

Ahmet Emirhan Göktas

Das E-Paper-Display ist über die SPI-Schnittstelle mit dem ESP32-S3-Mikrocontroller verbunden. Diese Schnittstelle wird zur Kommunikation und Anzeige von Bildern auf dem E-Paper-Display genutzt und muss entsprechend initialisiert und konfiguriert werden.

7.2.6 Niedriger Energieverbrauch

Ahmet Emirhan Göktas

Die Akkulaufzeit ist ein wichtiger Faktor. Das Displaymodul muss in den Schlafmodus gehen können, wenn es nicht in Gebrauch ist, und bei Bedarf wieder aufwachen, um Energie zu sparen und die Akkulaufzeit zu verlängern.

7.3 Klassendiagramm

Ahmet Emirhan Göktas

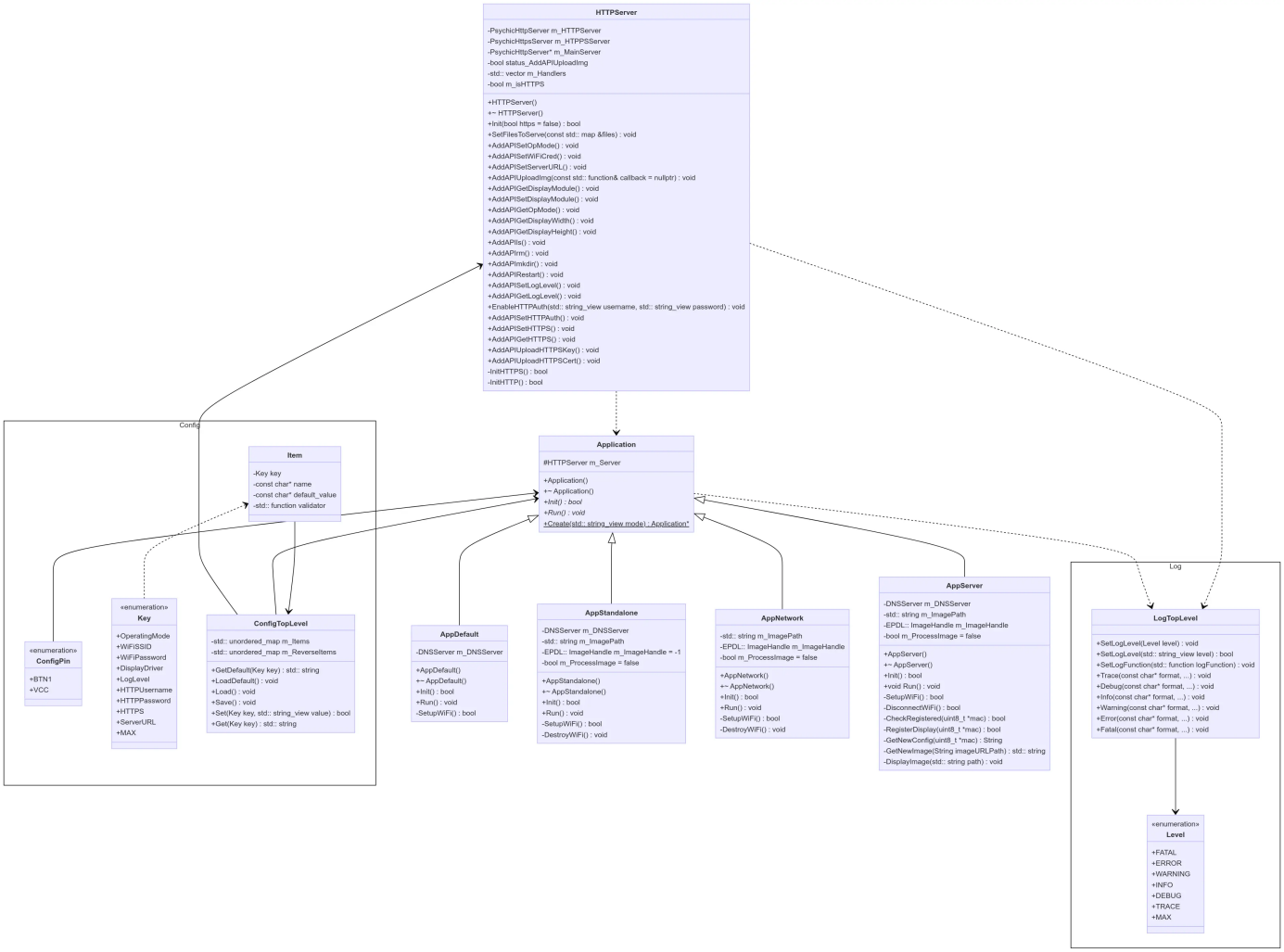


Abbildung 7.4: Klassendiagramm der Application

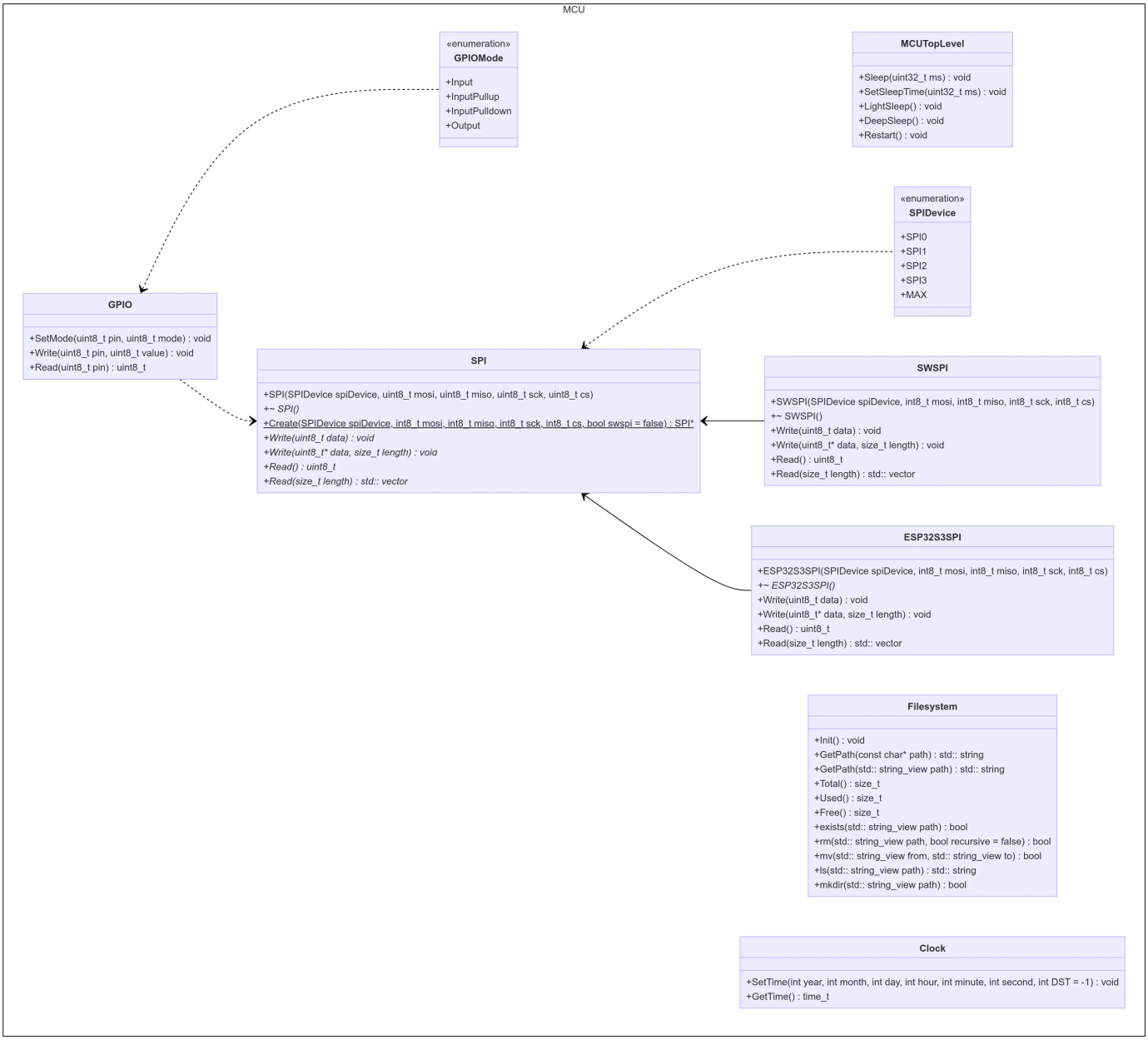


Abbildung 7.5: Klassendiagramm der Systemansteuerung

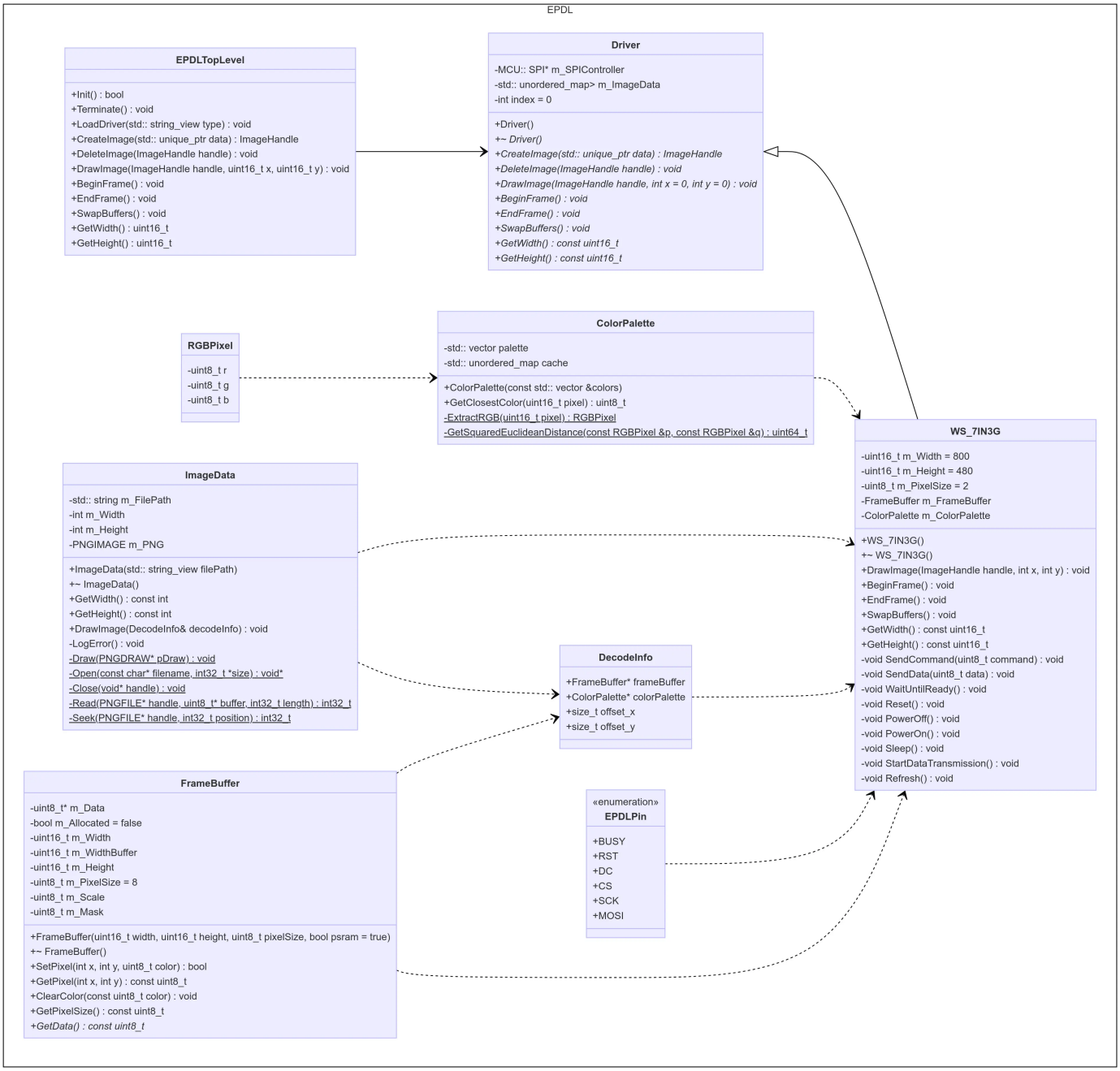


Abbildung 7.6: Klassendiagramm der E-Paper-Display-Library

7.4 Application

Ahmet Emirhan Göktas

Die Applikation ist der Hauptteil der Firmware. Sie ist verantwortlich für die Konfiguration des Displaymoduls, die drahtlose Kommunikation, die Betriebsmodi und die SPI-Schnittstelle. Die Applikation ist in verschiedene Klassen unterteilt, die jeweils für unterschiedliche Teile der Applikation verantwortlich sind. Diese sind:

- `Application`: Die Hauptklasse der Applikation. Sie ist eine Schnittstelle, die von den verschiedenen Betriebsmodi implementiert wird.
- `AppStandalone`: Der Standalone-Betriebsmodus. Das Displaymodul funktioniert ohne ein bestehendes WLAN.
- `AppNetwork`: Der Netzwerk-Betriebsmodus. Das Displaymodul funktioniert mit einem bestehenden WLAN.
- `AppServer`: Der Server-Betriebsmodus. Das Displaymodul wird von einem zentralen Server gesteuert.
- `AppDefault`: Der Standard-Betriebsmodus. Das Displaymodul befindet sich im Standardmodus, wenn es zum ersten Mal eingeschaltet wird oder wenn ein nicht behebbarer Fehler auftritt.
- `Log`: Ein Namensraum, der für das Loggen von Nachrichten auf den seriellen Port verantwortlich ist.
- `Config`: Ein Namensraum, der für die Konfiguration des Displaymoduls verantwortlich ist.

7.4.1 Ablauf der Applikation

Ahmet Emirhan Göktas

Die `Application`-Klasse ist die Hauptklasse der Anwendung. Der grundlegende Ablauf der Anwendung ist wie folgt:

1. Initialisierung des Dateisystems.
2. Lesen der Konfiguration aus dem Dateisystem.
3. Initialisierung des entsprechenden Modus.
4. Ausführen der Anwendung.

7.4.2 Betriebsmodi

Standardmodus

Ahmet Emirhan Göktas

In diesem Modus wird das WLAN im Access-Point-Modus initialisiert, um dem Benutzer die direkte Verbindung mit dem Gerät zu ermöglichen. Nachdem das WLAN initialisiert ist, wird der HTTP-Server gestartet. Die Standardseite ist die Einstellungsseite, wobei keine Endpunkte für den Bild-Upload enthalten sind.

Der Zweck dieses Modus besteht darin, dem Benutzer die Konfiguration des Geräts zu ermöglichen, unabhängig davon, ob etwas schiefgelaufen ist oder ein Werksreset durchgeführt wurde.

Standalone-Modus

Ahmet Emirhan Göktas

In diesem Modus arbeitet das Gerät unabhängig von bestehenden Infrastrukturen. Es beginnt mit der Initialisierung des WLANs im Access-Point-Modus. Anschließend wird der HTTP-Server mit Endpunkten für alle Einstellungen sowie für den Bild-Upload initialisiert. Eine Callback-Funktion wird übergeben, die aufgerufen wird, wenn ein Bild hochgeladen wird, sodass das Bild auf dem E-Paper-Display angezeigt werden kann. Schließlich wird das E-Paper-Display initialisiert, um bereit zu sein, die Bilder anzuzeigen.

Nachdem all diese Schritte erfolgreich abgeschlossen sind, ist das Gerät einsatzbereit. Man verbindet sich mit dem WLAN-Netzwerk des Geräts und ruft die Weboberfläche unter `http://192.168.4.1/index.html` auf. Das Gerät bleibt wach, bis ein Bild hochgeladen wird. Nach dem Hochladen wird das Bild auf dem E-Paper-Display angezeigt und das Modul geht in den Tiefschlaf, um die Akkulaufzeit zu verlängern und den Stromverbrauch zu minimieren.

Netzwerkmodus

Ahmet Emirhan Göktas

Dieser Modus bietet einfachen Zugriff auf das Gerät, indem es sich mit einem bestehenden WLAN-Netzwerk verbindet. Das Gerät stellt die Verbindung zum WLAN-Netzwerk her, indem es die in der Konfigurationsdatei gespeicherten Anmeldeinformationen verwendet. Nach der Herstellung der Verbindung wird der HTTP-Server genauso wie im Standalone-Modus initialisiert. Nachdem das Gerät im Netzwerk gefunden wurde, kann es über die IP-Adresse des Geräts ähnlich wie im Standalone-Modus aufgerufen werden.

Wenn während der Initialisierung ein Fehler auftritt, wie z. B. falsche WLAN-Anmeldeinformationen, die dazu führen, dass das Gerät nicht mehr erreichbar ist, wechselt das Gerät zurück in den Standardmodus, um dem Benutzer die Neukonfiguration des Geräts zu ermöglichen. Andernfalls bleibt die Routine dieselbe wie im Standalone-Modus.

Servermodus

Mario Wegmann

In diesem Modus wird mit einem Webbrowser nicht direkt auf das Displaymodul selbst zugegriffen, stattdessen holt sich jedes Displaymodul die notwendigen Informationen von einem zentralen Linux Webserver ab. Beim Konfigurieren im Standard-Modus wird neben den Zugangsdaten vom bestehenden WLAN-Netzwerk auch eine Server-URL mit angegeben. Bei jedem Aufwecken aus dem Deep-sleep verbindet sich das Displaymodul dann mit dem bestehenden WLAN und versucht den Server über HTTP(S) zu erreichen. Als Erstes wird überprüft, ob das Displaymodul dem Server bekannt ist, falls dies noch nicht der Fall ist, dann ruft das Displaymodul eine API-Route auf, um sich selbst als neues Display zu registrieren. Falls das Display bereits dem Server bekannt ist, wird die API-Route aufgerufen, um eine neue Darstellungs-Konfiguration anzufragen. In dieser Konfiguration meldet der Server dem Displaymodul, unter welcher URL das Displaymodul das aktuelle Asset heruntergeladen werden kann und wie lange dieses Asset angezeigt werden soll. Nach der erfolgreichen Kommunikation mit dem Server setzt das Displaymodul einen Wakeup Timer auf die Dauer der übergebenen Anzeigedauer und geht in den Deep-sleep Modus.

7.4.3 Konfiguration

Ahmet Emirhan Göktas

Der `Config`-Namensraum ist für die Konfiguration des Moduls verantwortlich. Die Konfiguration wird in einer JSON-Datei im Dateisystem gespeichert. Beim Booten wird die `Load`-Methode verwendet, um die Konfiguration aus dem Dateisystem zu laden, sodass die Konfiguration direkt aus dem RAM verfügbar ist. Die `Save`-Methode kann dann verwendet werden, um die Konfiguration zurück in das Dateisystem zu speichern. Dabei verwenden wir die ArduinoJson-Bibliothek, um die JSON-Datei zu lesen und zu schreiben.

7.4.4 Logging

Ahmet Emirhan Göktas

Der `Log`-Namensraum ist für das Loggen von Nachrichten auf den seriellen Port verantwortlich. Das Log-Level kann in der Konfigurationsdatei festgelegt werden. Die Protokollstufe kann auf `TRACE`, `DEBUG`, `INFO`, `WARN`, `ERROR` oder `FATAL` eingestellt werden. Das Log-Level wird über die Konfigurationsdatei festgelegt und die Nachrichten werden nur geloggt, wenn die Protokollstufe auf eine Stufe eingestellt ist, die gleich oder höher ist als die Stufe der Nachricht.

7.5 HTTP-Server

Ahmet Emirhan Göktas

The HTTP-Server is responsible for handling incoming HTTP requests. It is implemented using the `PsychicHttp` library, a lightweight and easy-to-use HTTP server library for the ESP32. We utilize the following features from the library:

- Handling HTTP requests such as GET and POST with file upload support
- Basic authentication
- HTTPS support [AEG_09]

7.5.1 HTTP Requests

Ahmet Emirhan Göktas

Der HTTP-Server ist dafür verantwortlich, eingehende HTTP-Anfragen zu bearbeiten. Er wird mit der `PsychicHttp`-Bibliothek implementiert, einer leichtgewichtigen und einfach zu verwendenden HTTP-Server-Bibliothek für den ESP32. Wir nutzen die folgenden Funktionen der Bibliothek:

- Bearbeitung von HTTP-Anfragen wie GET und POST mit Unterstützung für Dateiupload
- Basis-Authentifizierung
- HTTPS-Unterstützung

7.5.2 HTTP-Anfragen

Ahmet Emirhan Göktas

HTTP definiert Methoden, um die gewünschte Aktion an einer gegebenen Ressource anzugeben. Die Methodennamen sind groß- und kleinschreibungssensitiv und müssen in Großbuchstaben geschrieben werden, im Gegensatz zu Header-Namen. Diese Methoden können wie folgt kategorisiert werden:

- **Sicher**: Methoden, die nichts auf dem Server ändern.
- **Idempotent**: Methoden, die mehrfach mit dem gleichen Ergebnis aufgerufen werden können.
- **Cachefähig**: Methoden, deren Antworten gespeichert und später wiederverwendet werden können.

[AEG_10] [AEG_11]

GET-Anfrage

Ahmet Emirhan Göktas

Die GET-Methode fordert eine Darstellung der angegebenen Ressource an. Anfragen mit GET sollten nur Daten abrufen und keine anderen Auswirkungen auf die Daten haben. Zum Beispiel sendet der Browser eine GET-Anfrage an den Server, um die HTML-Datei der Website abzurufen, wenn eine Website aufgerufen wird.

Die Anfrage unten ist eine einfache GET-Anfrage, um die `index.html`-Datei vom Server abzurufen. Dies sollte die Hauptseite der Website zurückgeben.

```
GET /index.html HTTP/1.1
Host: 192.168.4.1
```

Ein weiteres Beispiel für eine GET-Anfrage ist unten gezeigt, bei dem die aktuelle Konfiguration des Geräts angefordert wird. Dies sollte den Betriebsmodus des Geräts (z.B. `Standalone`, `Netzwerk` oder `Server`) im Klartext zurückgeben.

```
GET /api/v1/GetOpMode HTTP/1.1
Host: 192.168.4.1
```

[AEG_12]

Derzeit hat der Server folgende Endpunkte:

GET `/api/v1/GetDisplayModule`

Gibt das angeschlossene E-Paper-Display-Modell zurück.

Mögliche Rückgabewerte:

- `WS_7IN3G`
- `WS_9IN7`

GET `/api/v1/GetOpMode`

Gibt den aktuellen Betriebsmodus des Geräts zurück.

Mögliche Rückgabewerte:

- `Standalone`
- `Network`
- `Server`

GET `/api/v1/GetDisplayWidth`

Gibt die Breite des angeschlossenen E-Paper-Displays in Pixel zurück.

Mögliche Rückgabewerte:

- `Positive integer`

GET `/api/v1/GetDisplayHeight`

Gibt die Höhe des angeschlossenen E-Paper-Displays in Pixel zurück.

Mögliche Rückgabewerte:

- `Positive integer`

GET `/api/v1/GetLogLevel`

Gibt das aktuelle Log-Level des Geräts zurück.

Mögliche Rückgabewerte:

- `Trace`
- `Debug`
- `Info`
- `Warn`
- `Error`
- `Fatal`

GET `/api/v1/GetHTTPS`

Gibt zurück, ob das Gerät HTTPS verwendet.

Mögliche Rückgabewerte:

- `true`
- `false`

Und diese Dateien sind derzeit auf dem Server verfügbar:

GET `/index.html`

Gibt die Hauptseite der Website zurück.

GET `/settings.html`

Gibt die Einstellungsseite der Website zurück.

GET `/utils.js`

Gibt die Hilfs-JavaScript-Datei zurück.

GET `/LPRD-Logo.webp`

Gibt das LPRD-Logo-Bild zurück.

GET `/html2canvas.min.js`

Gibt die html2canvas-Bibliothek zurück.

GET `/icons88-settings-25-w.png`

Gibt das Einstellungs-Icon-Bild zurück.

GET `/script.js`

Gibt die Haupt-JavaScript-Datei zurück.

GET `/style.css`

Gibt die Haupt-CSS-Datei zurück.

GET `/layouts/various_templates`

Gibt die Seite mit verschiedenen Vorlagen der Website zurück.

Hinweis: Dieses Verzeichnis enthält mehrere HTML-Vorlagen, die zum Generieren von Bildern verwendet werden, nicht zum direkten Seitenserver.

POST Request

Ahmet Emirhan Göktas

Die POST-Methode wird verwendet, um eine Entität an die angegebene Ressource zu übermitteln, was oft eine Änderung des Zustands des Servers zur Folge hat. Zum Beispiel, wenn ein Benutzer ein Formular auf einer Website absendet, sendet der Browser eine POST-Anfrage mit den Formulardaten an den Server. Der Typ der Anfrage wird durch den `Content-Type`-Header definiert.

Die untenstehende Anfrage ist eine einfache POST-Anfrage, um den Betriebsmodus des Geräts festzulegen. Der Anfrage-Body sollte den neuen Betriebsmodus des Geräts enthalten (z.B. `Standalone`, `Network` oder `Server`). Der Server sollte mit einer Erfolgsmeldung oder einer Fehlermeldung antworten, wenn der angegebene Modus ungültig ist.

```
POST /api/v1/SetOpMode HTTP/1.1
Host: 192.168.4.1
Content-Type: application/x-www-form-urlencoded
Content-Length: 16

mode=Standalone
```

[AEG_13]

Derzeit hat der Server folgende Endpunkte:

POST `/api/v1/SetOpMode`

Legt den Betriebsmodus des Geräts fest.

Anfrage-Body:

- `mode` - Der neue Betriebsmodus des Geräts.

Mögliche Werte:

- `Standalone`
- `Network`
- `Server`

POST `/api/v1/SetWiFiCred`

Legt die WLAN-Zugangsdaten des Geräts fest.

Anfrage-Body:

- `ssid` - Die SSID des WLAN-Netzwerks.
- `password` - Das Passwort des WLAN-Netzwerks.

POST `/api/v1/SetServerURL`

Legt die Server-URL des Geräts fest.

Anfrage-Body:

- `url` - Die URL des Servers.

POST `/api/v1/SetDisplayModule`

Legt das angeschlossene E-Paper-Display-Modell des Geräts fest.

Anfrage-Body:

- `model` - Das Modell des E-Paper-Displays.

Mögliche Werte:

- `WS_7IN3G`
- `WS_9IN7`

POST `/api/v1/restart`

Startet das Gerät neu.

POST `/api/v1/SetLogLevel`

Legt das Log-Level des Geräts fest.

Anfrage-Body:

- `level` - Das Log-Level des Geräts.

Mögliche Werte:

- `Trace`
- `Debug`
- `Info`
- `Warn`
- `Error`
- `Fatal`

POST `/api/v1/SetHTTPAuth`

Legt die HTTP-Authentifizierung des Geräts fest.

Anfrage-Body:

- `username` - Der Benutzername für die HTTP-Authentifizierung.
- `password` - Das Passwort für die HTTP-Authentifizierung.

POST `/api/v1/SetHTTPS`

Legt fest, ob das Gerät HTTPS verwendet oder nicht.

Anfrage-Body:

- `https` - Ob das Gerät HTTPS verwendet oder nicht.

Mögliche Werte:

- `true`
- `false`

POST `/api/v1/mkdir`

Erstellt ein neues Verzeichnis im Dateisystem.

Anfrage-Body:

- `path` - Der Pfad des neuen Verzeichnisses.

Hinweis: Dieser Endpunkt dient nur zu Debugging-Zwecken und ist nicht in der finalen Version enthalten.

POST `/api/v1/ls`

Listet die Dateien im angegebenen Verzeichnis auf.

Anfrage-Body:

- `path` - Der Pfad des Verzeichnisses.

Hinweis: Dieser Endpunkt dient nur zu Debugging-Zwecken und ist nicht in der finalen Version enthalten.

POST `/api/v1/rm`

Entfernt die angegebene Datei aus dem Dateisystem.

Anfrage-Body:

- `path` - Der Pfad der zu entfernenden Datei.

Hinweis: Dieser Endpunkt dient nur zu Debugging-Zwecken und ist nicht in der finalen Version enthalten.

7.5.3 File Upload

Ahmet Emirhan Göktas

Datei-Uploads werden typischerweise mit der POST Methode behandelt. Für eine Datei-Upload-Anfrage muss der `Content-Type` Header auf `multipart/form-data` gesetzt sein und der Body entsprechend formatiert werden.

`Multipart/form-data` ist ein Medientyp, der das Senden mehrere verschiedene Inhaltsteile in einer einzigen HTTP-Anfrage ermöglicht. Jeder Teil wird durch eine einzigartige Begrenzungszeichenfolge getrennt und besteht aus einem Header und einem Body. Dieses Format ermöglicht die gleichzeitige Übertragung von Dateien und anderen Daten.

Die Begrenzungszeichenfolge ist ein einzigartiger Trennstrich, der verwendet wird, um jeden Teil des multipart-Inhalts zu trennen. Der Server analysiert den Body der Anfrage und behandelt jeden Teil einzeln. Dies stellt sicher, dass Dateien, auch große, hochgeladen werden können, ohne typische Größenbeschränkungen für einzelne HTTP-Anfragen zu überschreiten.

Nachfolgend ist ein Beispiel für eine POST-Anfrage zum Hochladen einer Datei, die "Hello, World!" enthält, an den Server. Der Server sollte mit einer Erfolgsmeldung antworten, wenn die Datei erfolgreich gespeichert wird, oder mit einer Fehlermeldung, wenn dies fehlschlägt.

```
POST /upload HTTP/1.1
Host: example.com
Content-Type: multipart/form-data; boundary=----WebKitFormBoundary7MA4YWxkTrZu0gW
Content-Length: 210

------WebKitFormBoundary7MA4YWxkTrZu0gW
Content-Disposition: form-data; name="file"; filename="example.txt"
Content-Type: text/plain

Hello World!
------WebKitFormBoundary7MA4YWxkTrZu0gW--
```

[AEG_14] [AEG_15] [AEG_16]

Der Server hat derzeit folgende Endpunkte:

POST `/api/v1/UploadImg`

Lädt eine Bilddatei auf den Server hoch.

Anfrage-Body:

- `file` - Die hochzuladende Bilddatei.

POST `/api/v1/UploadHTTPSKey`

Lädt die HTTPS-Schlüsseldatei auf den Server hoch.

Anfrage-Body:

- `file` - Die hochzuladende HTTPS-Schlüsseldatei.

POST `/api/v1/UploadHTTPSCert`

Lädt die HTTPS-Zertifikatsdatei auf den Server hoch.

Anfrage-Body:

- `file` - Die hochzuladende HTTPS-Zertifikatsdatei.

7.5.4 Basic Authentication

Ahmet Emirhan Göktas

HTTP Basic Authentication ist ein einfaches Authentifizierungsschema, das im HTTP-Protokoll integriert ist. Der Client sendet HTTP-Anfragen mit einem Authorization-Header, der das Wort `Basic` gefolgt von einem Leerzeichen und einer base64-codierten Zeichenkette `username:password` enthält. Der Server decodiert die base64-Zeichenkette und extrahiert den Benutzernamen und das Passwort zur Authentifizierung. Dann überprüft der Server die Anmeldeinformationen in seiner Datenbank und antwortet mit einem `401 Unauthorized`-Statuscode, wenn die Anmeldeinformationen ungültig sind. Das untenstehende Beispiel zeigt eine Anfrage mit grundlegender Authentifizierung.

```
GET /api/v1/GetDisplayModule HTTP/1.1
Host: 192.168.4.1
Authorization: Basic YWRtaW46YWRtaW4K=
```

[AEG_17]

Alle oben aufgeführten Endpunkte sind mit Basic Authentication geschützt. Der Standardbenutzername und das Standardpasswort sind jeweils `admin`.

7.5.5 HTTPS Unterstützung

Ahmet Emirhan Göktas

HTTPS ist eine Erweiterung von HTTP, um es sicherer zu machen, indem die zwischen dem Client und dem Server ausgetauschten Daten verschlüsselt werden. HTTPS verwendet SSL/TLS zur Verschlüsselung der Daten, was das Abhören und Manipulieren der Daten verhindert. Die Unterstützung von HTTPS kann einfach aktiviert werden, indem dem Server die Zertifikats- und Schlüsseldateien zur Verfügung gestellt werden. Der Server verwendet diese Dateien dann, um nach einem Neustart eine sichere Verbindung mit dem Client herzustellen. [AEG_18]

7.6 Systemansteuerung

Ahmet Emirhan Göktas

Um das Gerät einfach steuern zu können, haben wir den MCU-Namespace implementiert. Er bietet uns Methoden zum Neustarten des ESP, zum Versetzen in den Schlafmodus oder zum Einrichten des SPI-Busses.

7.6.1 Allgemeine Steuerung und GPIO

Ahmet Emirhan Göktas

Es gibt verschiedene General Purpose Input/Output (GPIO)-Pins auf dem ESP32, die zur Steuerung externer Geräte verwendet werden können. Diese Pins können als Eingänge oder Ausgänge konfiguriert werden und können verwendet werden, um digitale Signale zu lesen oder zu schreiben, um externe Geräte zu steuern. Insgesamt stehen 48 GPIO-Pins auf dem ESP32S3 zur Verfügung. Es wird jedoch generell nicht empfohlen, alle zu verwenden, da einige für andere interne Funktionen wie Flash, RAM usw. benötigt werden. Es gibt auch RTC-GPIO-Pins, die verwendet werden können, um den ESP32 aus dem Tiefschlaf zu wecken.

Zum Beispiel konfigurieren wir den GPIO-Pin 2 zu Beginn des Programms als Eingang, um zu überprüfen, ob ein Werksreset angefordert wird oder nicht. Später konfigurieren wir den Pin jedoch, um den ESP32 aus dem Tiefschlaf zu wecken.

Ein weiteres Beispiel ist der GPIO 43, der zur Steuerung der Stromversorgung des E-Paper-Displays verwendet wird. Wir konfigurieren den Pin als Ausgang und schreiben ein hohes Signal auf den Pin, um das Display einzuschalten.

Es gibt auch praktische Methoden wie Neustart, Tiefschlafmodus und Einstellen eines Weckers. Es gibt auch eine Verzögerungsmethode, die es uns ermöglicht, die Ausführung des Programms zu pausieren, ohne den Watchdog-Timer auszulösen.

[AEG_19]

7.6.2 SPI

Ahmet Emirhan Göktas

Das Serial Peripheral Interface (SPI) ist eine synchrone serielle Kommunikationsschnittstelle, die es Geräten ermöglicht, miteinander zu kommunizieren. Es wird häufig verwendet, um Mikrocontroller mit Peripheriegeräten wie Sensoren, Displays und Speichergeräten zu verbinden. Es besteht aus vier Signalen: MOSI (Master Out Slave In), MISO (Master In Slave Out), SCK (Serial Clock) und CS (Chip Select). Die Steuerung des SPI-Busses ist etwas mühsam, daher haben wir eine eigene Wrapper-Bibliothek um den esp-idf-SPI-Treiber erstellt. [AEG_21]

Zum Beispiel ist das E-Paper-Display über den SPI-Bus mit dem ESP32 verbunden. Wir verwenden den SPI-Bus, um Befehle und Daten an das Display zu senden, um den Bildschirm zu aktualisieren. Nachdem die Pins 9 für MOSI, 3 für MISO, 7 für SCK und 4 für CS an den SPI-Treiber übergeben wurden, können wir Daten einfach an das Display senden, ohne uns um die low-level Details kümmern zu müssen. [AEG_20]

7.7 E-Paper Display API

Ahmet Emirhan Göktas

Die E-Paper Display API ist verantwortlich für die Steuerung des E-Paper-Displays. Die API stellt Methoden zur Initialisierung des Displays und zur Aktualisierung des Displays mit neuen Bildern bereit. Sie abstrahiert die low-level Details des Displays und bietet eine einfache Schnittstelle zur Interaktion mit dem zugrundeliegenden Treiber. Jedes Displaymodell benötigt seinen eigenen Treiber. Derzeit haben wir Treiber für die Waveshare 7,3 Zoll (G) und 9,7 Zoll Displays. [AEG_22] [AEG_23]

7.7.1 Framebuffer

Ahmet Emirhan Göktas

Der Framebuffer ist ein Speicherpuffer, der direkt aus dem PSRAM des ESP32S3 zugewiesen wird. Der Framebuffer wird verwendet, um die Bilddaten an das Display zu senden. Er speichert die gegebenen Daten bereit zur Übertragung an das Display, während er seinen Speicherbedarf so klein wie möglich hält. Zum Beispiel unterstützt das 7,3 Zoll Display 4 Farben, daher benötigt es nur 2 Bits pro Pixel, wie auch das Display die Daten erwartet. Wenn wir also die ersten vier Pixel auf Schwarz setzen wollen, sollten die Daten `0000 0000` im Binärformat sein, wobei jedes 2 Bits einen Pixel darstellen. Die Anzahl der Bits pro Pixel kann beim Initialisieren des Framebuffers übergeben werden, der diese dann zur Kompression der Daten so weit wie möglich nutzt, um Speicher zu sparen, während dennoch ein einfacher Zugriff auf jeden Pixel möglich bleibt. Die Kompression für das 7,3 Zoll Display beträgt 2 Bits pro Pixel und ergibt eine Speicherersparnis von 75 % im Vergleich zu 8 Bits pro Pixel. Anstatt also 375 KB (480 × 800 / 1024) zuzuweisen, benötigen wir nur 93,75 KB (480 × 800 / 1024 / 4).

7.7.2 ImageData

Ahmet Emirhan Göktas

Die ImageData Struktur wird verwendet, um die Bilddaten zu speichern, die auf dem E-Paper Display angezeigt werden sollen. Da wir begrenzte Mengen an RAM im ESP haben, erwartet die Struktur, dass Bilder in einer Datei im Dateisystem gespeichert werden. Das unterstützte Bildformat ist PNG, welches bereits Informationen wie Breite, Höhe und Farbtiefe enthält. Die Struktur wird vom Treiber gesteuert. Nachdem sie generiert wurde, können wir sie beim Treiber registrieren und ein Handle zur späteren Verwendung erhalten. Der Treiber kümmert sich dann um den Rest.

Später können wir die Draw-Methode mit den gewünschten x- und y-Offsets aufrufen, um das Bild auf den Framebuffer zu zeichnen, das dann auf dem Bildschirm angezeigt wird. Der Treiber übergibt intern dem Bilddecoder die Farbpalette und den Framebuffer zum Zeichnen des Bildes. Der Bilddecoder verwendet die PNGdec Bibliothek zum Dekodieren des Bildes und erhält die Pixeldaten. Diese Daten werden dann in tatsächlich darstellbare Farben umgewandelt, indem die nächstgelegene Farbe aus der Farbpalette ermittelt wird. [AEG_24] [AEG_25]

7.7.3 Colorpalette

Ahmet Emirhan Göktas

Dies ist eine Struktur, die wir verwenden können, um die Farben zu definieren, die wir haben möchten. Es kann später ein 16-Bit-Farbwert (RGB565) übergeben werden, um die nächstgelegene Farbe aus der Palette zu erhalten. Es verwendet den quadrierten euklidischen Abstand, um die nächstgelegene Farbe zu finden. Der euklidische Abstand ist die Quadratwurzel der Summe der quadrierten Differenzen zwischen den beiden Farben. Der quadrierte Abstand wird verwendet, um die Quadratwurzeloperation zu vermeiden, die rechnerisch aufwändig ist. Nach Berechnung jedes Abstands wird die Farbe mit dem kleinsten Abstand im Cache gespeichert, um später verwendet zu werden, und der Index der Farbe in der Palette wird zurückgegeben. Wenn wir zum Beispiel eine Palette mit 4 Farben haben: schwarz (0, 0, 0), weiß (255, 255, 255), rot (255, 0, 0) und gelb (255, 255, 0) und wir übergeben die Farbe (204, 0, 70) an die Palette, wird der Abstand zwischen jeder Farbe wie folgt berechnet:

$$d = (R_1 - R_2)^2 + (G_1 - G_2)^2 + (B_1 - B_2)^2$$

$$d_{\text{black}} = (204 - 0)^2 + (0 - 0)^2 + (70 - 0)^2 = 46516$$

$$d_{\text{white}} = (204 - 255)^2 + (0 - 255)^2 + (70 - 255)^2 = 101851$$

$$d_{\text{red}} = (204 - 255)^2 + (0 - 0)^2 + (70 - 0)^2 = 7501$$

$$d_{\text{yellow}} = (204 - 255)^2 + (0 - 255)^2 + (70 - 0)^2 = 72526$$

Aus diesen Berechnungen ergibt sich, dass die Farbe (204, 0, 70) der Farbe Rot in der Palette am nächsten ist. [AEG_26]

7.7.4 Treiber

Ahmet Emirhan Göktas

Der Treiber ist der Hauptteil der E-Paper Display API. Er ist verantwortlich für die Initialisierung des Displays, die Aktualisierung des Displays mit neuen Bilddaten und die Handhabung der low-level Details des Displays. Es gibt eine Interface-Klasse, die jeder Treiber erben muss. Die Interface-Klasse stellt die grundlegenden Methoden bereit, die der Treiber implementieren muss. Der Treiber ist verantwortlich für die Einrichtung des Displays, das Senden der Befehle an das Display und die Aktualisierung des Displays mit neuen Bilddaten. Der Treiber kümmert sich auch um die low-level Details des Displays, wie das Einrichten des SPI-Busses, das Senden der Daten an das Display und die Aktualisierung des Displays mit neuen Bilddaten.`

8. Webentwicklung

8.1 Framework und Library Auswahl

Julia Reuter

Die Interaktion zwischen Benutzer und Raumanzeige soll so einfach und unkompliziert wie möglich gehalten werden und dennoch genügend Funktionalitäten unterstützen. Sie erfolgt deshalb über eine selbsterstellte Webseite, die mithilfe von HTML5, CSS sowie JavaScript designt wurde und über den ESP32-Webserver gehostet wird. Als externe Bibliothek verwendet die Benutzerwebseite nur die Javascript Library html2canvas, um äußere Abhängigkeiten so weit wie möglich zu vermeiden. Mit über 182.000 Downloads und 45 Releases (zuletzt am 22.01.2022) [JR_19], wurde sich aufgrund der Popularität und der guten Pflege bewusst für diese Bibliothek entschieden. Der Javascript-Code kann heruntergeladen werden, sodass er lokal auf dem ESP32 gespeichert und eingebunden wird.

8.1.1 Beschreibung der html2canvas Bibliothek

Julia Reuter

Die JavaScript Library html2canvas ermöglicht es, "Screenshots" von Webseiten oder Teilen davon direkt im Browser des Benutzers zu erstellen. Im Gegensatz zu herkömmlichen Screenshots, die das tatsächliche Aussehen der Webseite pixelgenau erfassen, basiert html2canvas auf dem DOM (Document Object Model). Das Document Object Model ist eine plattform- und sprachunabhängige Schnittstelle, die HTML- und XML-Dokumente als strukturierte Baumdarstellungen repräsentiert. In einem DOM-Baum ist jedes Element, Attribut und Stück Text in einem Dokument als ein "Node" (Knoten) dargestellt und kann über unter anderem auch über eine vergebene ID einzeln angesprochen werden. Diese Knoten sind durch Beziehungen wie Eltern-Kind und Geschwisterknoten miteinander verbunden, wodurch eine hierarchische Struktur entsteht [JR_12].

Auf Grundlage der im DOM verfügbaren Informationen erstellt die Html2Canvas-Bibliothek eine Repräsentation der Webseite. Hierfür durchläuft der Scrip die DOM-Struktur und sammelt Informationen über alle vorhandenen Elemente. Diese Informationen nutzt es, um eine visuelle Darstellung der Seite zu generieren. Dabei werden diverse Eigenschaften der HTML-Elemente und CSS-Stile berücksichtigt, um die Struktur und das Erscheinungsbild der Webseite so genau wie möglich nachzubilden [JR_11].

Um einen Screenshot eines spezifischen Teils einer Webseite zu erstellen, kann ein bestimmtes div-Element übergeben werden. Der Script durchläuft dann nur den Teil der DOM-Struktur, der mit dieser spezifischen ID verbunden ist. Dies ermöglicht es, präzise und gezielt nur bestimmte Abschnitte einer Webseite zu erfassen und darzustellen, anstatt die gesamte Seite zu berücksichtigen.

Einschränkungen von html2canvas

html2canvas unterstützt eine Vielzahl von CSS-Stilen und kann beispielsweise background-color, border-radius, font-family und text-shadow korrekt rendern. Allerdings werden komplexere Eigenschaften wie filter, mix-blend-mode oder zoom nicht unterstützt.

Der wesentlichste Punkt bei der Verwendung von html2canvas ist jedoch die Behandlung von Bildern und anderen Medieninhalten. Bilder müssen unter derselben Herkunft (Same-Origin Policy) wie die Webseite liegen, damit die Bibliothek darauf zugreifen und sie korrekt darstellen kann. Die Same-Origin Policy ist ein Sicherheitskonzept, das in Webbrowsern implementiert ist, um zu verhindern, dass Skripte von einem Ursprung auf Inhalte eines anderen Ursprungs zugreifen. Ein Ursprung wird durch das Protokoll, den Host und den Port definiert. Diese Richtlinie soll verhindern, dass bösartige Skripte auf sensible Daten zugreifen können, die von einer anderen Webseite stammen [JR_13]

Da html2canvas auf Informationen aus dem DOM angewiesen ist, kann es Dateien aller Art, die von einem anderen Ursprung stammen, nicht direkt rendern. Dies führt dazu, dass eingebundene Bilder im erzeugten Screenshot fehlen oder unvollständig dargestellt werden [JR_11].

Um trotzdem Abbildungen im html2canvas Screenshot darzustellen, besteht die Möglichkeit, diese in Base64 kodiert direkt in das HTML-Dokument einzubetten. Diese Kodierung ermöglicht es, das Bild als Zeichenkette darzustellen, die vom Browser

unabhängig von der ursprünglichen Quelle verarbeitet werden kann. Dies umgeht die Same-Origin-Beschränkungen, führt jedoch zu einer erhöhten Datenmenge (33-37% Overhead im Vergleich zu der Originaldatei) und entsprechend längeren Ladezeiten [JR_14]

Browserkompatibilität

Die Bibliothek ist mit den meisten modernen Browsern kompatibel, einschließlich Firefox (ab Version 3.5), Google Chrome, Opera (ab Version 12), Internet Explorer (ab Version 9), Edge und Safari (ab Version 6) [JR_11].

8.1.2 Beschreibung der node-html-to-image

Mario Wegmann

Die node-html-to-image Bibliothek baut auf dem selben Prinzip, wie die html2canvas Bibliothek auf. Diese benötigt jedoch keinen bestehnden Browser mit GUI Oberfläche, sondern nutzt die headless-Variante von Chromium, um aus dem HTML ein Screenshot zu erstellen. [MW_11]

8.2 Benutzerwebseite

Julia Reuter

8.2.1 Aufbau und Funktionalitäten

Die Webseite zur Steuerung der Anzeigeinhalte unterstützt folgende zwei Funktionalitäten: PNG-Upload und HTML-Design

PNG-Upload

Nutzer können per Knopfdruck ein PNG-Bild auswählen und hochladen, sodass es im Anschluss direkt auf dem verbundenen Display angezeigt wird. Wichtig zu beachten ist hierbei, dass das PNG exakt auf die Pixeldimensionen des Displays abgestimmt ist, da es sonst nicht auf dem Display dargestellt werden kann. Um unnötigen Ressourcenverbrauch durch Senden fehlerhafter Dateien zu vermeiden, wird vor dem Upload überprüft, ob es sich bei der hochgeladenen Datei um ein PNG handelt und ob die Breite und Höhe, der des angeschlossenen Displays entspricht. Falls die Abmessungen nicht stimmen, oder ein anderes Dateiformat ausgewählt wurde, wird der Upload abgelehnt und der Nutzer erhält eine entsprechende Fehlermeldung. Erst wenn die Bildgröße korrekt ist, wird die Datei per HTTP POST (vgl. Kapitel) an den Server gesendet, wo es anschließend weiter verarbeitet wird.

HTML-Design

Ein weitere Möglichkeit, das Display mit Inhalten zu versorgen, ist mithilfe von HTML und CSS ein Design zu erstellen und hochzuladen. Der Benutzer kann in ein Textfeld HTML-Code eingeben und sich in Echtzeit eine Vorschau anzeigen lassen. Hierfür rendert die Webseite den Code bei jeder Eingabe in das Textfeld, sodass das Programmieren durch diese Art von Liveübertragung der Designfortschritt stets ersichtlich ist. Natürlich kann der Code auch außerhalb in einer Entwicklungsumgebeung wie VS-Code geschrieben und anschließend in das vorhandene Textfeld kopiert werden. Damit nach Fertigstellung des Designs der HTML-Code an den Server gesendet werden kann, muss das HTML-Design zunächst in ein PNG konvertiert werden. Diese Konvertierung erfolgt per Knopfdruck durch die in Kapitel 9.1.1 beschriebene Bibliothek html2canvas. Nachdem der erfolgreichen Umwandlung wird auch dieses PNG auf die korrekten Dimensionen überprüft und an den Server übermittelt. Es liegt hier also am Benutzer, sein Design im korrekten Pixelformat zu erzeugen.

Zusätzlich stehen dem Benutzer vordefinierte HTML-Vorlagen zur Auswahl, um den gesamten Prozess etwas zu verkürzen. Für bestimmte Anwendungsfälle wie bespielsweise ein Pausenschild vor der Bürotüre oder eine Stundenplananzeige vor dem Hörsal wurde ein Layout erstellt,dessen Inhalt über ein zusätzliches Textfeld personalisiert werden kann. Natürlich kann das Design auch direkt über den jweiligen HTML-Code angepasst werden, was dem Nutzer noch zusätzliche Felxibilität bietet. Hintergrund dieses Vorlagen-Konzeptes ist, wenn mehrere Displays beispielsweise als Raumanzeigen vor Hörsälen eingesetzt werden, könnten ICS-Kalenderdaten aus WebUntis automatisiert in ein bestehendes Layout integriert werden. Dieser Schritt erspart viel Zeit und ist spätestens bei der Verwaltung von vielen Modulen im Server-Modus essentiell.

Settings Page

Neben den verschiedenen Design-Upload Varianten, bietet die Benutzerwebseite noch eine Einstellungsseite mit einigen Konfigurationsmöglichkeiten.

1. Betriebsmodus
2. Verbundenes Display-Modul
3. WiFi-Konfiguration
4. Serielles Konsolen Log-Level
5. HTTP Authentifikation
6. HTTPS Setting
7. Server URL

Betriebsmodus Hier kann der Nutzer den Betriebsmodus der Anzeige (Standalone, Netzwerk, Server) über ein Dropdown-Menü ändern. Der Defaultmodus ist der Standalone, in dem sich das System auch befindet, wenn es auf Werkseinstellungen zurückgesetzt wurde.

Display-Modul Alle zur Verfügung stehenden Display-Module sind hier aufgelistet und können ausgewählt werden. Per Default gibt es momentan nur das Waveshare 7.3" 4 color und 9.7" 2 color Modul, wobei dies natürlich nach Belieben erweitert werden kann.

WiFi-Konfiguration Um die Low-Power Anzeige im Netzwerkmodus zu betreiben können hier die WLAN-Zugangsdaten (SSID und Passwort) eingegeben werden. Nach der Datenübertragung an den Server versucht sich das Modul beim nächsten Neustart mit den entsprechenden WLAN zu verbinden.

Serielles Konsolen Log-Level Diese Einstellung ist hauptächlich als Debugging-Maßnahme oder für Entwickler hilfreich. In der Browserkonsole wird die detaillierte Verfolgung von Ereignissen und Fehlern ermöglicht, die während der Laufzeit der Anwendung auftreten. Durch die Anpassung des Log-Levels (s. Tabelle 9.2.1) kann die Tiefe und Art der Protokollierung bestimmt werden, um die Menge und Art der Informationen, die das System auf der seriellen Konsole ausgibt zu steuern. So können spezifische Probleme zu diagnostiziert, oder das allgemeine Systemverhalten zu überwacht werden [JR_16].

Tabel: Tabelle 8.1 Übersicht über die verscheidenen Log-Levels [JR_17] { #_tab_8_1 }

Level	Beschreibung	Verwendung
Fatal	Kritische Fehler, die dazu führen, dass das System nicht mehr weiterarbeiten kann.	Wird verwendet, um schwerwiegende Probleme zu identifizieren, die sofortige Aufmerksamkeit erfordern.
Error	Schwerwiegende Fehler, die das System zwar nicht vollständig stoppen, aber zu signifikanten Problemen führen.	Ideal zur Erkennung und Behebung von Fehlern, die die Funktionalität beeinträchtigen.
Warning	Warnungen über potenzielle Probleme, die zu Fehlern führen könnten.	Hilfreich, um mögliche zukünftige Probleme zu erkennen und frühzeitig zu beheben.
Info	Allgemeine Informationen über den Systembetrieb.	Nützlich für ein allgemeines Verständnis des Systemverhaltens und zur Überwachung des normalen Betriebs.
Debug	Detaillierte Debugging-Informationen, die bei der Fehlersuche helfen.	Wird hauptsächlich von Entwicklern genutzt, um tiefergehende Probleme zu diagnostizieren und den Code zu debuggen.
Trace	Sehr detaillierte Protokolle, die fast alle Aktivitäten des Systems aufzeichnen.	Wird verwendet, um tiefgreifende Einblicke in die Systemoperationen zu erhalten und komplexe Probleme zu analysieren.

HTTP-Authentifikation Wenn das System konfiguriert werden soll, nachdem es sich in den Werkseinstellungen befand, muss der Benutzer mit Zugangsdaten anmelden, um auf die Webseite zugreifen zu können. Diese Zugangsdaten können hier geändert werden.

HTTPS Settings Die HTTPS-Einstellungen erlauben es dem Nutzer, HTTPS-Verschlüsselung der Webseite zu aktivieren und eine externe Zertifikatsdatei hochzuladen. Diese kann einfach und kostenfrei über beispielsweise den Dienst Let's Encrypt erstellt werden [JR_18].

Nachdem alle gewünschten Einstellungen ausgewählt und gespeichert wurden, werden sie per HTTP POST an den Server gesendet, sodass die Änderungen nach einem Neustart des Systems in Kraft treten.

8.2.2 Maßnahmen zur Steigerung der Benutzerfreundlichkeit

Allgemein ist die Konfiguration des Displays sehr stark modulabhängig. Um trotzdem ausreichend Benutzerfreundlichkeit zu garantieren, werden beim Starten der Webseite gewisse Funktionen aufgerufen, um die Attribute des verbundenen Displays abzufragen. Die Serverkommunikation erfolgt kommunizieren mit dem Server über asynchrone HTTP-Anfragen, um die notwendigen Informationen zu erhalten und die Benutzeroberfläche entsprechend anzupassen.

So werden beim Aufrufen der Webseite die Dimensionen des verbundenen Display-Moduls abgefragt und auf der Uploadseite angezeigt. Zusätzlich wird der Benutzer auf der Settings-Page über den eingestellten Display-Modus und die allgemeine Modulart informiert.

Kann keine Verbindung zum Server aufgebaut werden, oder ist der Bilduplоad fehlgeschlagen, so wird der Benutzer informiert.

8.3 Linux Webanwendung

Mario Wegmann

8.3.1 Verwendete Technologien

Wie im Kapitel 2.9.1 Webtechnologien bereits erläutert, ist es für die Entwicklung einer komplexeren Webanwendung sehr sinnvoll, ein Webframework einzusetzen. Hierbei gibt es eine große Auswahl an möglichen Frameworks und auch serverseitigen Programmiersprachen. Da JavaScript bereits im Standalonemodus, wie auch im Netzwerk-Modus, verwendet wird, um die Interaktion mit dem Displaymodul zu realisieren, ist es naheliegend, auch auf dem Server JavaScript zu nutzen. Diese Entscheidung ermöglichte eine einheitliche Codebasis sowohl im Frontend als auch im Backend, was die Entwicklung effizienter und die Wartung der Anwendung einfacher macht und zuletzt auch das Erlernen neuer Programmiersprachen auf eine reduziert. Zudem wurde die Möglichkeit genutzt, TypeScript anstatt JavaScript zu verwenden, um die Vorteile zu nutzen, welche ebenso bereits im Kapitel 2.9.1 Webtechnologien erwähnt wurden.

Für das Frontend wurde React ausgewählt. Durch React lassen sich Komponenten realisieren, welche modular wiederverwendet werden können. Dadurch wird es vermieden, doppelten Code zu verfassen und Änderungen an einer Komponente werden global in der gesamten Webanwendung widergespiegelt.

Als Webframework wurde Next.js ausgewählt, da es React unter anderem um serverseitiges Rendering erweitert. Darüber hinaus bietet Next.js eine nahtlose Integration von API-Routen und Middleware, was die Entwicklung von Full-Stack-Anwendungen erleichtert.

Für das Speichern der Daten wurde PostgreSQL als Datenbank ausgewählt. Hierbei wird Prisma ORM als Zwischenschicht eingesetzt, damit die Modelle der relationanlen Datenbank als Objekte mit statischen Typen generiert werden.

Für den Webserver wurden Linux und teilweise Docker verwendet. Das quelloffene Linux ist bekannt für seine Stabilität, Sicherheit und Performance, was es zu einer idealen Wahl für den Einsatz als Webserver macht. Docker ergänzt diese Vorteile durch die Bereitstellung einer containerisierten Umgebung, die eine konsistente und isolierte Ausführung von Anwendungen ermöglicht. Dies erleichtert die Skalierung und Verwaltung der Anwendung erheblich und sorgt dafür, dass sie in unterschiedlichen Umgebungen gleichbleibend funktioniert. Docker-Container bieten zudem eine einfache Möglichkeit, Abhängigkeiten zu verwalten und die Bereitstellung von Updates zu automatisieren.

Die Kombination dieser Technologien ermöglicht effizient eine benutzerfreundliche, performante und wartbare Webanwendung zu entwickeln.

Wie die Einzelkomponenten zusammenspielen, ist in Abbildung 8.1 ersichtlich.

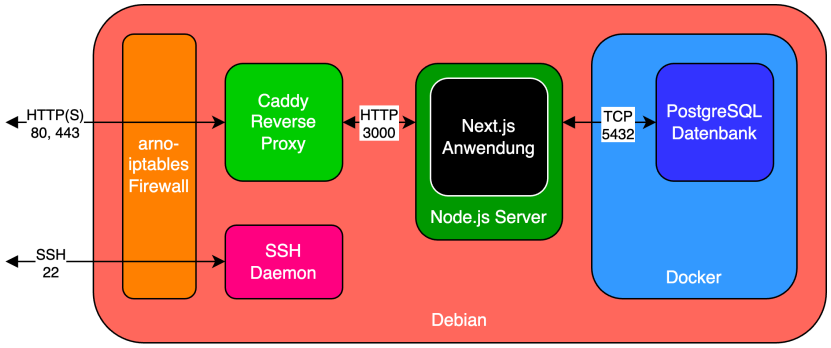


Abbildung 8.1: Der Techstack der Webanwendung

Der Code wurde größtenteils in VSCode verfasst und mit der Versionskontrolle git auf GitHub verwaltet.

8.3.2 Aufbau der Webanwendung

Datenbankstruktur

In der PostgreSQL Datenbank gibt es zwei Tabellen, die erste Tabelle enthält alle Displaymodule, der Aufbau einer Displaymodul-Entität ist in Tabelle 8.2 erkennbar. Hierbei wird als Primary Key die MAC-Adresse verwendet, da diese bei der Produktion vom Hersteller eindeutig vergeben wird, eignet sich diese sehr gut als Primary Key. Des Weiteren werden ein frei definierbarer Anzeigename, die Auflösung des verbauten ePaper Displays, der Zeitpunkt, wann sich das Displaymodul das letzte Mal mit dem Server verbunden hat und das aktuell festgelegte Asset hinterlegt. In Tabelle 8.3 ist die Asset-Entität aufgelistet, diese besteht aus einer zufällig generierten cuid für den Primary Key, einer Typenangabe, ob statisch oder dynamisch, einem frei definierbaren Anzeigename, der Dateipfad auf dem Server Dateisystem, dem HTML Code für das Erstellen des Assets, sofern es über HTML generiert wird und die Anzeigedauer. Die PNGs werden lokal auf dem Dateisystem des Servers abgelegt, somit kann die Datenbank einerseits schlank bleiben und der Webserver kann die PNGs einfach ausliefern.

Die meisten Eigenschaften wurden als optional deklariert, damit die Entitäten auch schon erstellt werden können, auch wenn noch nicht alle Eigenschaften bekannt sind.

Key	Type	Eigenschaft
mac_adr	String	MAC-Adresse, Primary Key
friendly_name	String	Anzeigename, Optional
width	Int	Breite des Displays
height	Int	Höhe des Displays
last_seen	DateTime	Zuletzt gesehen, Optional
currentAsset	String	Aktuell anzuzeigendes Asset, Optional
currentAssetType	String	Typ des aktuell anzuzeigenden Assets, Optional

Tabelle 8.2: Die Display-Entität

Key	Type	Eigenschaft
id	String	Primary Key, cuid Algorithumus
type	Type	Art des Assets, Optional, Kann "STATIC" oder "DYNAMIC" sein
friendly_name	String	Anzeigename, Optional
file_path	String	Pfad auf dem Dateisystem des Servers, Optional
html	String	HTML Code zum generieren des PNGs, Optional
valid_for	Int	Anzeigedauer des Assets, Optional

Tabelle 8.3: Die Asset-Entität

Backend

Für eine standardisierte Verwendung der Daten wurde eine REST-API-Schnittstelle umgesetzt. So können mit GET-Anfragen Daten angefragt, mit PUT-Anfragen Daten bearbeitet und mit DELETE-Anfragen Daten auch gelöscht werden. Sowohl das Frontend als auch die Displaymodule verwenden diese REST-API.

Eine vollständige Übersicht der möglichen REST-API-Aufrufe ist in Tabelle 8.4 ersichtlich. Platzhalter in eckigen Klammern werden dabei dynamisch in der Anfrage berücksichtigt.

URL	Methode	Body	Rückgabe	Zweck
api/v1/assets	GET	-	JSON	Gibt alle vorhandenen Assets in JSON zurück.
api/v1/assets	PUT	HTML Form	JSON	Erstellt ein neues Asset. Falls im HTML Form ein Key `html` enthalten ist, wird der Inhalt als HTML interpretiert und ein PNG davon erzeugt und lokal abgespeichert. Falls im HTML Form eine PNG-Datei enthalten ist, so wird diese lokal abgespeichert.
api/v1/assets/[ID]	GET	-	JSON	Gibt das Asset mit der übergebenen ID als JSON zurück.
api/v1/assets/[ID]	PUT	HTML Form	HTTP Status	Aktualisiert die Daten des Assets mit der passenden ID, falls HTML mitgesendet wird, so wird das PNG neu genriert und lokal abgespeichert.
api/v1/assets/[ID]	DELETE	-	HTTP Status	Löscht das Asset mit der übergebenen ID.
api/v1/displays	GET	-	JSON	Gibt alle vorhandenen Displaymodule in JSON zurück.
api/v1/displays/[MAC]	GET	-	JSON	Gibt das Displaymodule mit der passenden MAC-Adresse in JSON zurück.
api/v1/displays/[MAC]	PUT	HTML Form oder JSON	HTTP Status	Aktualisiert die Infos des Displaymoduls mit der MAC-Adresse oder setzt das zugewiesene Asset neu.
api/v1/displays/[MAC]	DELETE	-	HTTP Status	Löscht das Displaymodul mit der MAC-Adresse aus der Datenbank.
api/v1/displays/config/[MAC]	GET	-	JSON	Gibt als JSON zurück, welches Asset gerade für das Displaymodul mit der MAC-Adresse zugewiesen ist und wie lange es angezeigt werden soll.
api/v1/displays/register/[MAC]	PUT	JSON	JSON	Erstellt das Displaymodul mit der MAC-Adresse neu in der Datenbank und gibt das neu erstellte Displaymodul als JSON zurück.

Tabelle 8.4: Die Unterstützen REST-API Aufrufe der Webanwendung

Frontend

Im Frontend gibt es fünf Unterseiten. Die Startseite enthält das Menü zu den anderen beiden Hauptunterseiten »Displays« und »Assets«. In der Unterseite »Displays« befindet sich eine Auflistung aller dem Server bekannten Displaymodulen. Ein Displaymodul registriert sich dabei beim Server selbst, da es im Servermodus die Register-URL aufruft und sich dem Server so bekannt macht. In Abbildung 8.2 ist erkennbar, dass jedes Displaymodul seinem mit seinem Anzeigenamen und auch einer Vorschau des aktuell zugewiesenen Assets dargestellt wird. Ein Displaymodul kann in der Übersicht angeklickt werden, wodurch

sich eine Detailansichtsseite öffnet. In dieser Detailseite, welche Abbildung 8.3 zeigt, können die Metadaten des Displays bearbeitet und gespeichert werden. Zusätzlich werden alle Assets angezeigt, sobald ein Asset angeklickt wird, wird dieses als neues Asset dem Display zugewiesen.

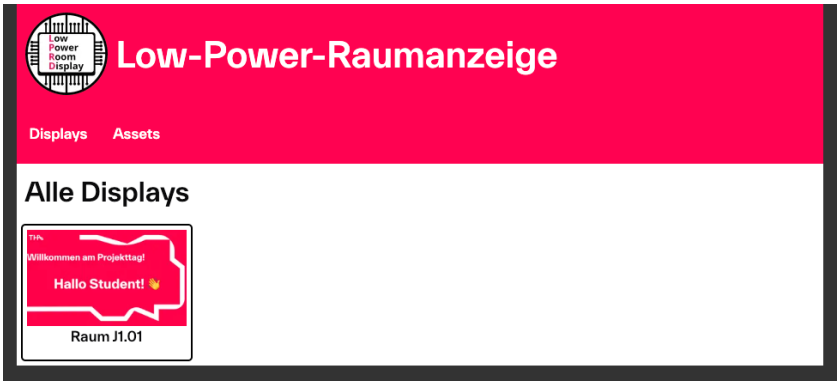


Abbildung 8.2: Die Übersichtsseite aller Displays

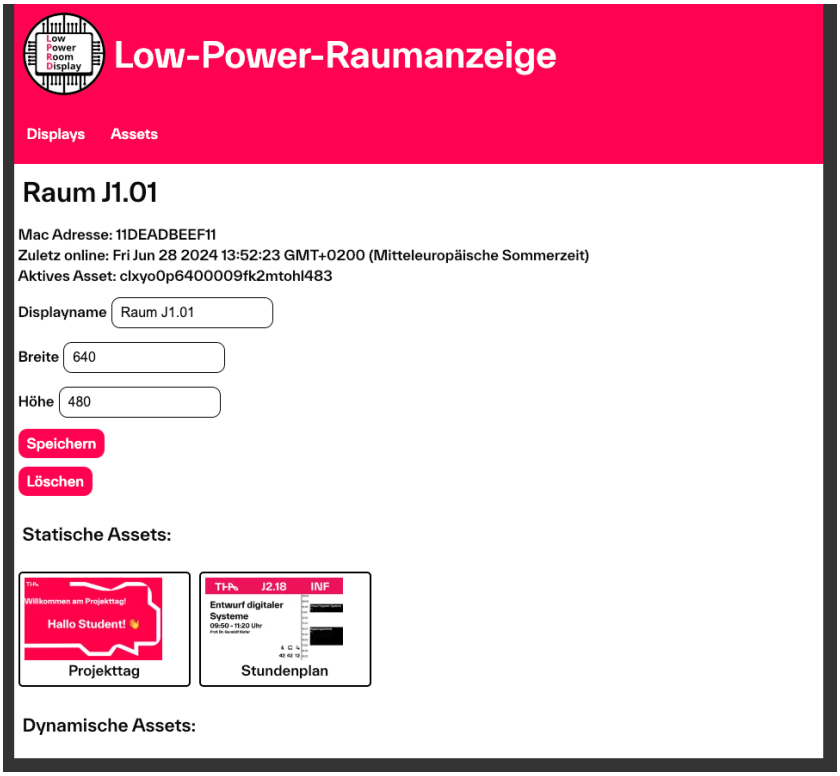


Abbildung 8.3: Die Detailansicht eines Displays

Ähnlich verhält es sich mit der zweiten Hauptunterseite »Assets«, die unter Abbildung 8.4 erkennbar ist. Auch hier werden alle vorhandenen Assets aufgelistet und können ausgewählt werden, um eine Detailansicht, wie in Abbildung 8.5 gezeigt, zu öffnen und die Metadaten zu bearbeiten und das Asset zu löschen. Im Gegensatz zu Displaymodulen können Assets händisch angelegt werden, entweder kann dafür eine PNG-Datei hochgeladen oder HTML-Code eingetragen werden, welcher abschließend wieder ein PNG produziert. Die Abbildungen Abbildung 8.6 und Abbildung 8.7 zeigen jeweils die beiden Vorgänge.

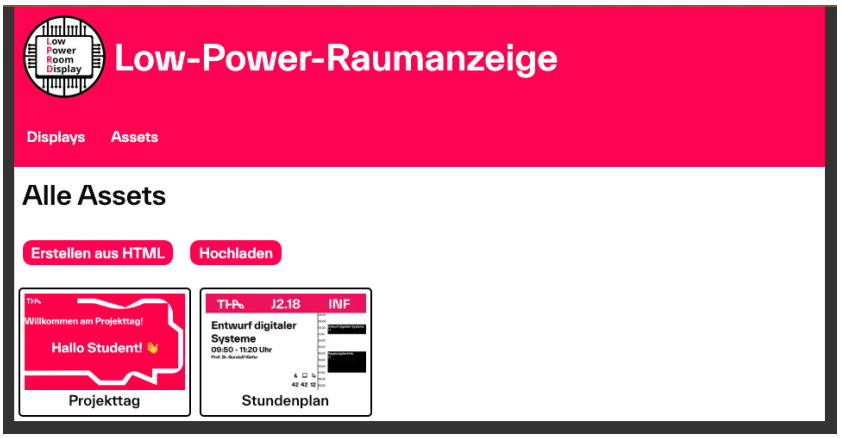


Abbildung 8.4: Eine Übersicht über alle vorhanden Assets

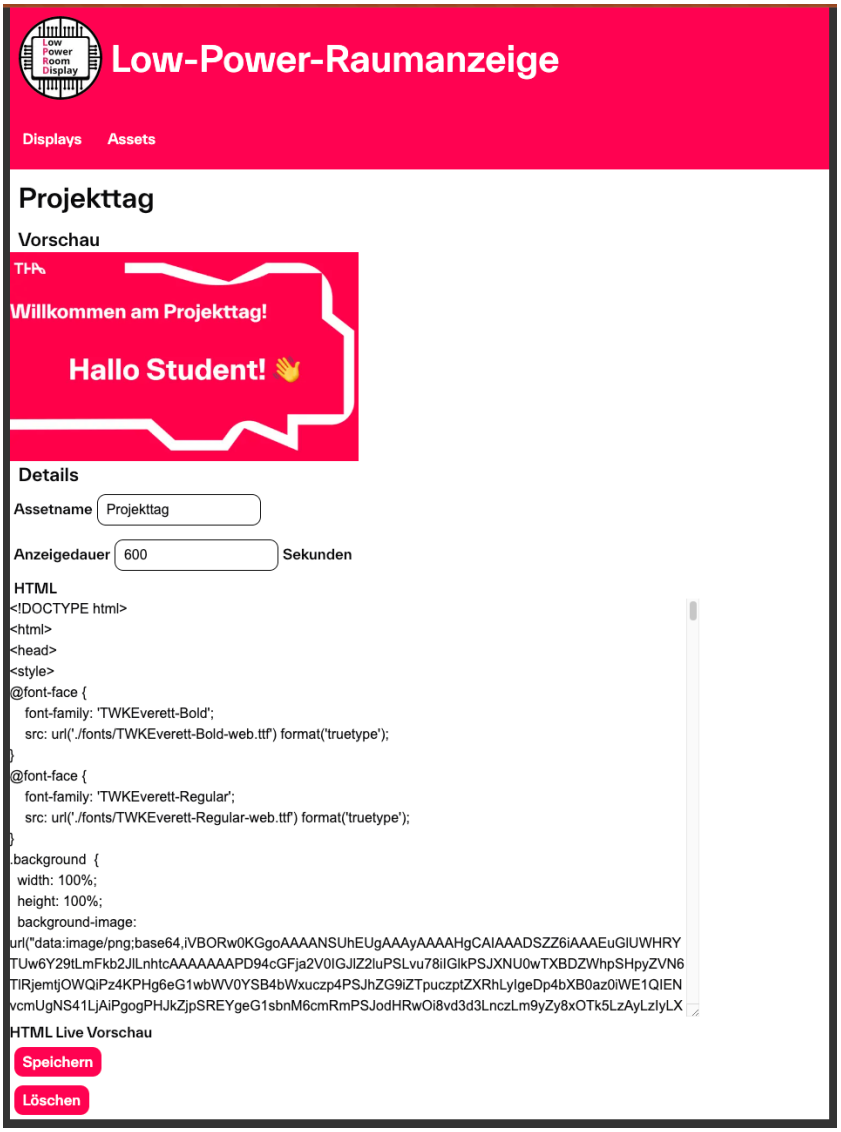


Abbildung 8.5: Die Detailansicht eines Assets

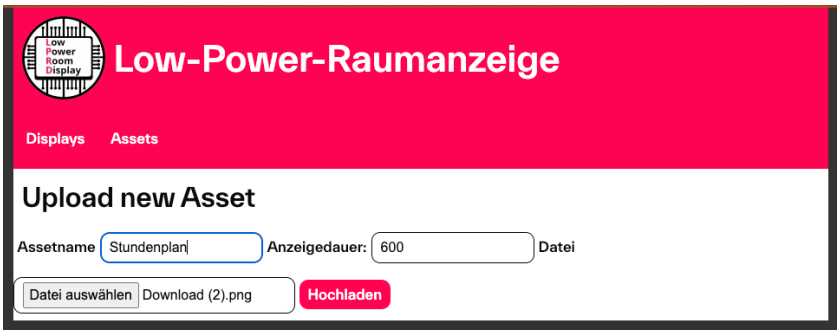


Abbildung 8.6: Das Formular zum hochladen von bestehenden PNGs

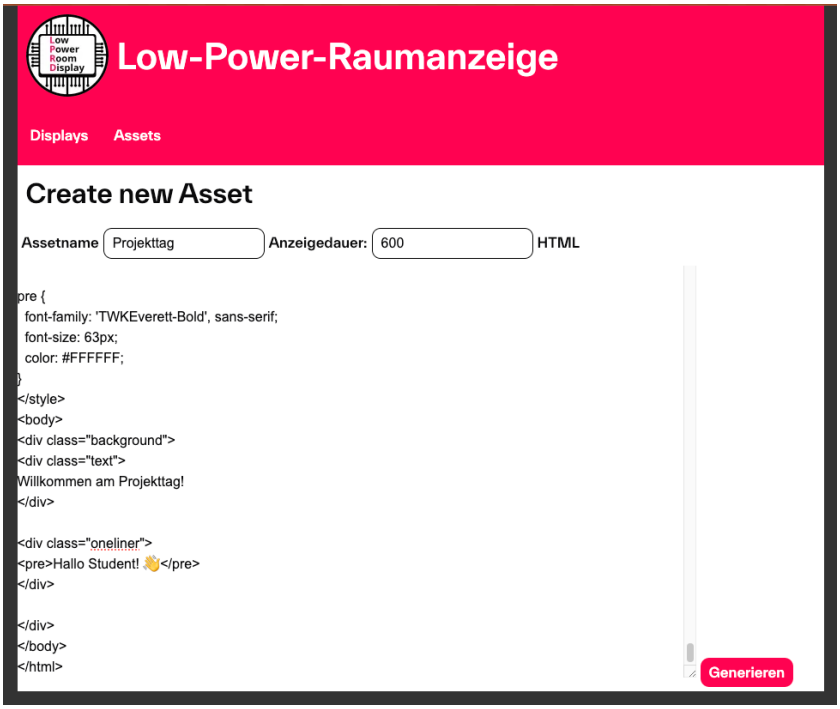


Abbildung 8.7: Das Formular zum erstellen von PNGs aus HTML Code

8.3.3 Testumgebung

Als Testumgebung wurde beim Rechenzentrum (RZ) eine Linux-VM beantragt. Die vom Rechenzentrum erhaltene VM hat die in Tabelle 8.5 genannten Eigenschaften.

Eigenschaft	Wert
Betriebssysrtem	Debian 12 (bookworm)
Prozessor-Cores	2x 2,9 GHz
Arbeitsspeicher	2 GB
Hauptspeicher	30 GB
Hostname	lprd

Tabelle 8.5: Die Systemeigenschaften der vom RZ erhaltenen VM.

Zudem wurde vom RZ auch gleich ein DNS-Eintrag gesetzt und die Domain lprd.informatik.tha.de verweist auf diese VM. Neben dem Betriebssystem liefert die VM auch gleich noch eine Firewall standardmäßig mit, hier setzt das RZ die Variante arno-iptables ein [MW_01].

8.3.4 Ersteinrichtung

Der administrative Zugriff auf die VM erfolgt über SSH. Nach der Übergabe wurden zuerst die Standardzugangsdaten der VM durch neue Zugangsdaten ersetzt und die SSH-Anmeldung des Root-Benutzers gesperrt. Ebenso wurden für die Personen, welche Zugriff auf die VM benötigen, jeweils neue Benutzer angelegt. Dabei wurden als Benutzerauthentifizierung SSH-Keys als Authentifizierungsmethode verwendet. Anschließend wurden die installierten Pakete mit dem Advanced Packaging Tool (apt) Paketmanager auf die neueste Version aktualisiert.

8.3.5 Einrichtung der Webanwendung

Nach der erfolgreichen Grundinstallation kann mit dem Aufsetzen der Testumgebung der Webanwendung begonnen werden. Dafür werden zuerst die Pakete git und docker über den apt Paketmanager installiert.

Die PostgreSQL-Datenbank wurde in einem Docker Container eingerichtet. Anschließend konnte das GitHub-Projekt lokal auf dem Server geklont werden. Vor dem ersten Start der Webanwendung kann mit Prisma ORM die Datenbank konfiguriert und initialisiert werden, Prisma ORM verwendet dabei die Informationen, die in der schema.prisma Datei angegeben sind. Danach lässt sich mit einem npm Befehl ein Development Server starten.

Änderungen am Code werden nach dem Speichern der Quelldatei sofort in den laufenden Development Server eingepflegt, was eine schnelle und effiziente Entwicklung ermöglicht.

8.3.6 Datensicherung und Monitoring

Der Zweck der VM ist primär die Entwicklung der Webanwendung und somit nicht der produktive Einsatz. Da der Quellcode der Webanwendung über Git verwaltet wird und die Daten innerhalb der Datenbank reine Testdaten sind, wurde auf das Einrichten einer Datensicherung verzichtet.

Für eine produktive Umgebung ist eine sorgfältig überlegte Backupstrategie unabdingbar. Hierbei sollte genaustens überlegt werden, wie der Backupprozess implementiert und automatisiert wird und wie erstellte Backups auf Konsistenz und Wiederherstellbarkeit überprüft werden können. Auch der Speicherort von Backups sollte bedacht werden und der 3-2-1 Regel folgen.

Auch wurde keine Monitoringlösung verwendet, um den Zustand der VM und der darauf laufenden Dienste zu überwachen, auch hier ist es sinnvoll, ein Konzept für die Produktivumgebung auszuarbeiten.

9. Fazit

Jannis Gröger

9.1 Zusammenfassung

Das Projekt zur Entwicklung eines Low Power Raumdisplays war in mehrern Hinsichten ein klarer Erfolg. Das Hauptziel, eine energieeffizientes Informationsanzeige zu entwickeln, wurde erfüllt und kann, wie in den Anforderungen vgl Kapitel 5.3 beschrieben, nicht nur an Hochschulen, sondern auch in Firmen, an Messeständen und in Privathaushalten genutzt werden.

Das Display zeichnet sich mit einer langen Akkuklaufzeit und sehr geringer Leistungsaufnahme durch die durchdachte Auswahl stromsparender Komponenten wie das E-Paper Display und softwareseitige Hardwareverwaltung wie die Implementierung des Deepsleeps. Durch die Aufbauen eines eigenes WLAN Acces Points des Mikrocontrollers und die Stromversorgung per Akku ist das Displaymodul komplett autark von Stromnetz und Netzwerk, was den Einsatzbereich und die Anwendungsszenarien deutlich erweitert.

Zudem lässt sich die Raumanzeige durch den frontseitigen Benutzerknopf und die implementierte Website sehr leicht steuern und einfach bedienen. Die Verwaltung mehrerer Displays im Server-Modus geschieht ebenfalls über eine Website, die sich in der Struktur sehr ähnlich sind, was die Benutzerfreundlichkeit steigert.

Das schlanke und durchdachte Design des Gehäuses ermöglicht nicht nur eine einfache Montage, sondern auch ein leichtes Austauschen der Akkupacks beim Laden, ein Zurücksetzen des Displaymoduls ohne viel Aufwand und ein Debugging oder Systemflash durch die Zugänglichkeit zum Mikrocontroller.

Insgesamt bietet das entwickelte Low Power Raumdisplay eine zukunftsweisende Lösung für verschiedene Einsatzszenarien, von Bürogebäuden über öffentliche Einrichtungen bis hin zu privaten Haushalten. Es trägt zur Reduzierung des Energieverbrauchs bei und unterstützt damit nachhaltige und umweltfreundliche Technologien.

Abschließend kann festgehalten werden, dass die Projektarbeit nicht nur zur Weiterentwicklung technischer Kompetenzen beigetragen hat, sondern auch einen praktischen Nutzen bietet, der im Einklang mit den heutigen Anforderungen an Energieeffizienz und Benutzerfreundlichkeit steht. Die positiven Testergebnisse und das Feedback aus ersten praktischen Anwendungen bestätigen den Erfolg des Projekts und zeigen das Potential für eine breite Markteinführung.

9.2 Ausblick

Da die Projektarbeit in ihrem zeitlichen Rahmen begrenzt ist, ist eine Beschränkung auf eine begrenzte Anzahl an Features nicht nur sinnvoll, sondern auch essentiell. Im Folgenden werden mögliche Erweiterungen und Verbesserungen der Low Power Raumanzeige genannt, die bei einer Weiterentwicklung berücksichtigt werden können.

Eine interessante Information über das Display selbst ist der momentane Akkustand, dass verhindert werden kann, dass das Display nicht mehr mit Strom versorgt wird und seine Funktion damit verliert. Denkbar wäre hier die Messung der Akkukapazität und das Anzeigen dieser auf dem Display, sodass der Betreiber einen einfachen Überblick darüber bekommt, wann er die Akkus eines Moduls tauschen oder laden muss.

Zudem ist eine Verbesserung der entworfenen Platine eine Erweiterung, die in Betracht gezogen werden kann. Durch das Anbringen von so genannten Mounting Holes, könnte die Platine leichter im Gehäuse befestigt werden, und das feste Verlöten des Mikrocontrollers auf der Platine würde die gesamte Dicke des PCBs verringern. Dadurch könnten entsprechende Veränderungen am Design des Gehäuses vorgenommen werden, um ein noch dünneres und robusteres Modul zu bekommen.

Eine weitere mögliche Verbesserung ist die Implementierung eines Light Sleeps, der es ermöglicht, das Display jeder Zeit erreichbar zu machen. So kann beispielsweise im Netzwerk Modus nicht nur eine Bildaktualisierung vorgenommen werden, wenn der Benutzer auf den Refresh Knopf drückt, sondern auch dann, wenn der Betreiber ein neues Bild über die Website hochlädt.

Die momentane Erstellung von Anzeigebilder aus vorhanden Templates, in die der Betreiber des Displays Daten händisch einträgt, könnte in Zukunft automatisiert passieren. Damit könnte eine Erstellung eines Zeitplans aus externen Quellen wie zum

Beispiel einem Kalender per CalDAV realisiert werden. Zudem könnte auch aus Raumbelegungsdaten einer Software wie Beispielsweise WebUntis ein Zeitplan erstellt, was für Bildungseinrichtungen von großem Interesse sein kann.

Die Funktion des frontseitigen Benutzerknopfes sind auch vielseitig erweiterbar. Beispielsweise das Wechseln zwischen mehreren Anzeigebildern oder das Buchen eines Raumes, welches dann auch in einem Raumbelegungsplan angezeigt wird, sind vorstellbare Verbesserungen.

Das momentane unterstützte 7.3-Zoll-Display benötigt für jede Bildaktualisierung eine längere Zeit und flackert stark, um die Pigmente des E-Papers richtig zu platzieren. Dies ist nicht nur umständlich für den Benutzer, sondern kann auch ein möglicher Trigger für Menschen mit photosensitiver Epilepsie sein. Beim 9.7-Zoll-Display ist die Aktualisierung um einiges schneller mit eine Zeit von unter einer Sekunde, jedoch werden hierbei keine Farben unterstützt. Eine Verbesserung wäre die Nutzung eines Displays, welches die sogenannte "Fast Refresh" Technologie des 9.7-Zoll-Displays wie auch eine Farbunterstützung ähnlich wie bei dem 7.3-Zoll-Display besitzt.

Die genannten Verbesserungs- und Erweiterungsmöglichkeiten bieten die Chance auf eine noch bessere Benutzerfreundlichkeit und leichtere Bedienung, um die Anwendungsfälle zu optimieren und das Einsatzspektrum zu erweitern.

10. Glossar

Displaymodul

Komplettes Set bestehend aus Display, Mikrocontroller, Akku, Platine und Gehäuse.

Display

Das Display, welches angesteuert und mit Strom und Daten versorgt werden muss. Ein solches Display kann Beispielweise in Abbildung 10.1 gesehen werden.

Abbildung 10.1: Ein exemplarisches ePaper Display

Mikrocontroller

Im Displaymodul verbauter Computer, welcher drahtlos mit Clients kommunizieren kann und die Ansteuerung des Displays übernimmt.

Mikrocontroller Webserver

Kann immer Bitmaps über HTTP empfangen und im (Standalonemodus) zusätzlich die Client-Website an einen Client ausliefern.

Client

Gerät (Smartphone, Tablet, Computer) worüber der Benutzer, mittels eines Browsers, mit dem Displaymodul direkt (Standalonemodus) oder alternativ mit dem zentralen Webserver (Servermodus) kommuniziert.

Client Website

Die im Client sichtbare Website bestehend aus HTML, CSS und JavaScript.

Server

Linux-Container, welcher den zentralen Webserver betreibt.

Server Webserver

Die Anwendung läuft auf dem Server 24/7. Kann sich Daten von externen Quellen holen, diese von HTML in Bitmaps konvertieren und die Bitmaps an Displaymodule verteilen.

Server Website

Website für den Client, welche auf dem zentralen Webserver läuft und womit sich mehrere Displaymodule steuern lassen.

Standalonemodus

Der Client kommuniziert direkt mit dem Mikrocontroller, dabei erstellt der Mikrocontroller selber ein WLAN. Der Mikrocontroller arbeitet also als WLAN Access-Point.

Netzwerkmodus

Der Client kommuniziert direkt mit dem Mikrocontroller, dabei wird ein vorhandenes WLAN benutzt. Der Mikrocontroller arbeitet also als WLAN-Client.

Servermodus

Der Client kommuniziert nur über den Server Webserver mit dem Mikrocontroller, dabei wird ein vorhandenes WLAN benutzt. Der Mikrocontroller arbeitet also als WLAN-Client.

Asset

Den Inhalt, den ein Displaymodul anzeigen soll. Ein Asset wird als PNG dem Displaymodul zur Verfügung gestellt. Die Bezeichnung Asset unterscheidet sich von einem Bild in dem Aspekt, dass ein Asset eine Gültigkeitsdauer hat und auch aus HTML-Code generiert werden kann. Erst für die Übertragung an ein Displaymodul wird ein Asset in ein Bild gewandelt.

11. Quellenverzeichnis

[AEG_01] Mehrere Autoren, Hardware Abstraction Layers, https://en.wikipedia.org/w/index.php?title=Hardware_abstraction&oldid=1229915402, 2024-06-28, 2024-06-19

[AEG_02] Mehrere Autoren, Arduino IDE, https://en.wikipedia.org/w/index.php?title=Arduino&oldid=1231008379, 2024-06-28, 2024-06-25, S 1, 7-9

[AEG_03] Mehrere Autoren, PlatformIO, http://Platform.io, 2024-06-28, 2024-06-02

[AEG_04] Mehrere Autoren, ESP-IDF, https://docs.platformio.org/en/stable/frameworks/espidf.html, 2024-06-28, 2024-03-24

[AEG_05] Mehrere Autoren, ESP-IDF vs Arduino, https://www.espboards.dev/blog/esp-idf-vs-arduino-core/, 2024-06-28, 2023-03-18

[AEG_06] Mehrere Autoren, Installieren von PlatformIO extension, https://platformio.org/install/ide?install=vscode, 2024-06-29, 2024-05-24

[AEG_07] Mehrere Autoren, Konfiguration von XIAO ESP32S, https://docs.platformio.org/en/latest//boards/espressif32/seeed_xiao_esp32s3.html, 2024-06-29, 2024-03-06

[AEG_08] Vincent Driessen, Git flow, https://nvie.com/posts/a-successful-git-branching-model/, 2024-06-29, 2024-03-05

[AEG_09] Mehrere Autoren, PsychicHttp Bibliothek und die Wiki, https://github.com/hoeken/PsychicHttp, 2024-06-29, 2024-05-28

[AEG_10] Mehrere Autoren, HTTP Methoden MDN, https://developer.mozilla.org/en-US/docs/Web/HTTP/Methods, 2024-06-29, 2024-06-29

[AEG_11] Mehrere Autoren, HTTP Methoden Wikipedia, https://en.wikipedia.org/w/index.php?title=HTTP&oldid=1231005162#HTTP/1.1_request_messages, 2024-06-29, 2024-06-25, S. 10-11

[AEG_12] Mehrere Autoren, HTTP GET Request, https://developer.mozilla.org/en-US/docs/Web/HTTP/Methods/GET, 2024-06-29, 2024-05-21

[AEG_13] Mehrere Autoren, HTTP POST Request, https://developer.mozilla.org/en-US/docs/Web/HTTP/Methods/POST, 2024-06-29, 2024-06-09

[AEG_14] L. Masinter, RFC7578 über multipart/form-data, https://www.rfc-editor.org/rfc/pdfrfc/rfc7578.txt.pdf, 2024-06-29, 2015-06

[AEG_15] Mehrere Autoren, HTTP MIME types, https://developer.mozilla.org/en-US/docs/Web/HTTP/Basics_of_HTTP/MIME_types, 2024-06-29, 2024-06-26

[AEG_16] Mehrere Autoren, multipart/form-data MDN, https://developer.mozilla.org/en-US/docs/Learn/Forms/Sending_and_retrieving_form_data, 2024-06-29, 2024-05-22

[AEG_17] Mehrere Autoren, Basic Auth MDN, https://developer.mozilla.org/en-US/docs/Web/HTTP/Authentication, 2024-06-29, 2024-05-22

[AEG_18] Mehrere Autoren, Was ist HTTPS, https://en.wikipedia.org/w/index.php?title=HTTPS&oldid=1231015807, 2024-06-29, 2024-06-26, S. 1

[AEG_19] Mehrere Autoren, Espressif dokumentation über GPIO, https://docs.espressif.com/projects/esp-idf/en/v5.2.2/esp32s3/api-reference/peripherals/gpio.html, 2024-06-29, 2023-10-12

[AEG_20] Mehrere Autoren, Espressif dokumentation über SPI, https://docs.espressif.com/projects/esp-idf/en/v5.2.2/esp32s3/api-reference/peripherals/spi_master.html, 2024-06-29, 2024-10-08

[AEG_21] Mehrere Autoren, Was ist SPI, https://en.wikipedia.org/w/index.php?title=Serial_Peripheral_Interface&oldid=1230957841, 2024-06-29, 2024-06-25, S. 1-3

[AEG_22] Mehrere Autoren, Waveshare wiki 9.7" Display, https://www.waveshare.com/wiki/9.7inch_e-Paper_HAT, 2024-06-29, 2024-05-25

[AEG_23] Mehrere Autoren, Waveshare wiki 7.3" Display, https://www.waveshare.com/wiki/7.3inch_e-Paper_HAT_(G)_Manual, 2024-06-29, 2023-12-08

[AEG_24] Mehrere Autoren, Was ist PNG, https://en.wikipedia.org/w/index.php?title=PNG&oldid=1222529480, 2024-06-29, 2024-05-06, S. 1

[AEG_25] Mehrere Autoren, PNGdec Bibliothek, https://github.com/bitbank2/PNGdec, 2024-06-29, 2024-04-19

[AEG_26] Mehrere Autoren, Euclidean distance, https://en.wikipedia.org/w/index.php?title=Euclidean_distance&oldid=1214273566, 2024-06-29, 2024-03-17

[BK_01] Mehrere Authoren, Lithium-Polymer-Akkumulator, https://de.wikipedia.org/wiki/Lithium-Ionen-Akkumulator, Zugriffsdatum: 2024-06-27, Herausgabedatum: 2024-06-21, Seiten 1-12, 12-14

[BK_02] Testing Lithium-ion Batteries, https://www.gamry.com/application-notes/battery-research/testing-lithium-ion-batteries, Zugriffsdatum: 2024-06-27, Herausgabedatum: 2024-06-21, Seite 6

[BK_03] Mehrere Authoren, Batteriemanagementsystem, https://de.wikipedia.org/wiki/Batteriemanagementsystem, Zugriffsdatum: 2024-06-28, Herausgabedatum: 2024-04-06, Seiten 1-4

[BK_04] Mehrere Authoren, Lithium-Polymer-Akkumulator, https://de.wikipedia.org/wiki/Lithium-Polymer-Akkumulator, Zugriffsdatum: 2024-06-27, Herausgabedatum: 2024-05-19, Seiten 1-4

[BK_05] Mehrere Authoren, Lithium-Eisenphosphat-Akkumulator, https://de.wikipedia.org/wiki/Lithium-Eisenphosphat-Akkumulator#Eigenschaften, Zugriffsdatum: 2024-06-27, Herausgabedatum: 2024-06-11, Seiten 1-5

[BK_06] Walter Dvorak, LIFePO4 charge discharge diagram, https://de.wikipedia.org/wiki/Datei:LIFePO4_charge_discharge_diagram.svg, Zugriffsdatum: 2024-06-27, Herausgabedatum: 2016-12-28, Seiten 1-2

[BK_07] Citric, Seeed Studio XIAO ESP32S3, https://de.wikipedia.org/wiki/Datei:LIFePO4_charge_discharge_diagram.svg, Zugriffsdatum: 2024-06-27, Herausgabedatum: 2023-03-23, Seiten 1-4, 6

[BK_08] Espressif Systems, ESP32-S3 Series Datasheet, https://de.wikipedia.org/wiki/Datei:LIFePO4_charge_discharge_diagram.svg, Zugriffsdatum: 2024-06-27, Herausgabedatum: 2023-11-24, Seiten 2-4

[BK_09] BAK N21700CD-53E mit 5300mAh 10A 3,6V - 3,7V Li-Ionen-Akku, https://www.akkuteile.de/lithium-ionen-akkus/21700/bak/bak-n21700cd-53e-mit-5300mah-10a-3-6v-3-7v-li-ionen-akku_100625_3323, Zugriffsdatum: 2024-06-27, Herausgabedatum: 2024-02-28, Seite 1, 2-3

[BK_10] 1S PCB/PCM - Keeppower 1S-5530 (Schutzelektronik) 2MOS, https://www.akkuteile.de/1s-pcb-keeppower-xzd-1s5530-schutzelektronik-7a_200501_1407, Zugriffsdatum: 2024-06-27, Herausgabedatum: 2023-11-30, Seite 1

[BK_11] Fuyuang 1S 3,6V - 3,7V (4,2V) Li-Ion-Ladegerät 2A + DC Stecker, https://www.akkuteile.de/fuyuang-enerpower-1s-3-6-3-7v-4-2v-li-ion-ladegeraet-2a-dc-stecker_400619_2570, Zugriffsdatum: 2024-06-27, Herausgabedatum: 2024-02-28, Seite 1

[BK_12] Walt Kester, Decoupling, https://www.analog.com/en/resources/analog-dialogue/studentzone/studentzone-april-2017.html, Zugriffsdatum: 2024-06-29, Herausgabedatum: 2024-04-13, Seiten 1-3

[BK_13] Analog Devices Inc., Decoupling Techniques, https://www.analog.com/en/resources/analog-dialogue/studentzone/studentzone-april-2017.html, Zugriffsdatum: 2024-06-29, Herausgabedatum: 2009-03, Seiten 1-6

[BK_14] Würth Elektronik, MagI3C Power Module WPME-FISM, https://www.we-online.com/components/products/datasheet/1769205041.pdf, Zugriffsdatum: 2024-06-29, Herausgabedatum: 2023-08, Seiten 20-23

[BK_15] Würth Elektronik, WS-TASV SMT Tact Switch, https://www.we-online.com/components/products/datasheet/430182050816.pdf, Zugriffsdatum: 2024-06-29, Herausgabedatum: 2020-06-04, Seiten 2-3

[BK_16] Würth Elektronik, Kontakt-Entprellschaltung für Taster, https://www.we-online.com/catalog/media/o185485v410%20SN015b%20DE.pdf, Zugriffsdatum: 2024-06-29, Herausgabedatum: 2020-08-31, Seiten 1-2

[BK_17] Ultra Librarian, https://www.ultralibrarian.com, Zugriffsdatum: 2024-06-30, Herausgabedatum: 2024-06-19, Seite

[BK_18] SnapMagic, https://www.snapeda.com/home/, Zugriffsdatum: 2024-06-30, Herausgabedatum: 2024-06-25, Seite

[BK_19] Thea, 2 Layer Simple Design Rules, https://community.aisler.net/t/2-layer-simple-design-rules/3735, Zugriffsdatum: 2024-06-30, Herausgabedatum: 2024-05-22, Seite 4

[BK_20] Mehrere Authoren, Freerouthing, https://freerouting.org, https://github.com/freerouting/freerouting, Zugriffsdatum: 2024-06-30, Herausgabedatum: 2024-06-29, Seiten 1-6

[BK_21] Tim Williams, The ground plane: Lord of the Board, http://elmac.co.uk/Lord_of_the_board.pdf, Zugriffsdatum: 2024-06-30, Herausgabedatum: 2007-09, Seiten 1-6

[BK_22] 7.3inch e-Paper HAT (G) Manual, https://www.waveshare.com/wiki/7.3inch_e-Paper_HAT_(G)_Manual#Resource, Zugriffsdatum: 2024-06-29, Herausgabedatum: 2024-02-28, Seite 4

[BK_23] msfujino, Battery voltage monitor and AD conversion for XIAO_ESP32C, https://wiki.seeedstudio.com/XIAO_ESP32C3_Getting_Started/#check-the-battery-voltage, Zugriffsdatum: 2024-06-30, Herausgabedatum: 2023-12-15, Seite

[JG_01] Prof. Dr. D. Wöhrle, Prof. Dr. Hendrik Wöhrle, Materialien in Rechnern und digitalen Computern, https://onlinelibrary.wiley.com/doi/full/10.1002/ciuz.201900004, Zugriffsdatum: 2024-06-20, Herausgabedatum: 2020-02-05

[JG_02] Jens Lienig, Hans Brümmer, Elektronische Gerätetechnik, Springer Vieweg Berlin, Heidelberg, Herausgabedatum: 2024-05-03, ISBN: 978-3-662-68707-9, S.44f, S. 92ff, S.149ff, S481ff,

[JG_03] Sandor Vanja, Christian Weber, Helmut Bley, Klaus Zeman, CAx für Ingenieure, Springer-Verlag Berlin Heidelberg, 2. Auflage 2009, ISBN: 978-3-540-36038-4, S.75ff, S. 242

[JG_04] Hubert Högl, Fablab, https://hhoegl.informatik.hs-augsburg.de/hhwiki/Fablab, Zugriffsdatum: 2024-06-27, Herausgabedatum: 2023-02-01

[JG_05] Thomas Jackson, Vollständig vergleichende Analys PLA vs. PETG, https://sunlu.com.de/blogs/products-knowledge/vollstandige-vergleichende-analyse-pla-vs-petg#:~:text=PLA%20ist%20ein%20biologisch%20abbaubarer,sich%20aber%20leichter%20drucken%20l%C3%A4sst., Zugriffsdatum: 2024-06-27, Herausgabedatum: 2024-02-21

[JG_06] Ondřej Grundlagen des 3D-Drucks mit Josef Prusa, Prusa Research a.s., 1. Ausgabe 2020, S. 35f

[JR_01] Wilke Technology GmbH, Wizepanel Digitale Türschilder https://wizepanel.de/produkte/#schilder, Zugriffsdatum: 28-06-2024, Letzte Bearbeitung: 27-09-2023

[JR_02] digitalSIGNAGE.de GmbH, Digitale Türschilder https://www.digitalsignage.de/digitale-tuerschilder.html, Letzte Bearbeitung: 01-03-2024, Zugriffsdatum: 28-06-2024

[JR_03] ROOMZ SA, ROOMZ Display https://roomz.io/roomz_display, Zugriffsdatum: 28-06-2024, Letzte Bearbeitung: 10-06-2024

[JR_04] Beetronics B.V., Touchscreen-Displays https://www.beetronics.de/c-touchscreens, Zugriffsdatum: 28-06-2024, Letzte Bearbeitung: 14-04-2024

[JR_05] Wilke Technology GmbH, Wizepanel 9"7 classic Datenblatt https://wizepanel.de/downloads/DATA_Sheet_WizePanel_97_Classic_DE.pdf, Zugriffsdatum: 28-06-2024, Letzte Bearbeitung: 03-08-2024

[JR_06] digitalSIGNAGE.de GmbH, Digitales Signboard https://www.digitalsignage.de/portfolio-hardware/xds-1078d-digital-signage-signboard.html, Zugriffsdatum: 28-06-2024, Letzte Bearbeitung: nicht bekannt

[JR_07] Beetronics B.V., 10 Zoll Touchscreen Datenblatt https://www.beetronics.de/datasheets/product/341, Zugriffsdatum: 28-06-2024, Letzte Bearbeitung: nicht bekannt

[JR_08] Wilke Technology GmbH, Wizepanel Handbuch, S.52 https://wizepanel.de/downloads/WizePanel_Manual_EN.pdf, Zugriffsdatum: 28-06-2024, Letzte Bearbeitung: 03-08-2024

[JR_09] ROOMZ SA, ROOMZ Sensor https://roomz.io/roomz_sensor/?lang=de, Zugriffsdatum: 28-06-2024, Letzte Bearbeitung: 04-10-2023

[JR_10] ROOMZ SA, myRoomz App https://roomz.io/myroomz-arbeitsplatzmanagement/?lang=de, Zugriffsdatum: 28-06-2024, Letzte Bearbeitung: 04-10-2023

[JR_11] Niklas von Hertzen, Html2Canvas Dokumentation https://html2canvas.hertzen.com/documentation, Zugriffsdatum: 28-06-2024, Letzte Bearbeitung: 06-06-2024

[JR_12] Diverse Autoren, Document Object Level (DOM) https://de.wikipedia.org/w/index.php?title=Document_Object_Model&oldid=235351737#Standardisierung, Zugriffsdatum: 28-06-2024, Letzte Bearbeitung: 10-07-2023

[JR_13] Diverse Autoren, Same-origin Policy https://en.wikipedia.org/w/index.php?title=Same-origin_policy&oldid=1229277634#References, Zugriffsdatum: 28-06-2024, Letzte Bearbeitung: 15-06-2024

[JR_14] Diverse Autoren, Base64 https://en.wikipedia.org/w/index.php?title=Base64&oldid=1230643470, Zugriffsdatum: 28-06-2024, Letzte Bearbeitung: 23-06-2024

[JR_15] Niklas von Hertzen, Html2Canvas Features https://html2canvas.hertzen.com/getting-started, Zugriffsdatum: 28-06-2024, Letzte Bearbeitung: 06-12-2023

[JR_16] Diverse Autoren, Web development tools https://en.wikipedia.org/w/index.php?title=Web_development_tools&oldid=1231254703, Zugriffsdatum: 29-06-2024, Letzte Bearbeitung: 27-06-2024

[JR_17] Diverse Autoren, Log4j https://en.wikipedia.org/w/index.php?title=Log4j&oldid=1230768614, Zugriffsdatum: 29-06-2024, Letzte Bearbeitung: 24-06-2024

[JR_18] Internet Security Research Group (ISRG), Über Let's Encrypt https://letsencrypt.org/de/about/, Zugriffsdatum: 29-06-2024, Letzte Bearbeitung: 25-06-2024

[JR_19] Niklas Von Hertzen, Github html2canvas https://github.com/niklasvh/html2canvas, Zugriffsdatum: 30-06-2024, Letzte Bearbeitung: 29-06-2024

[MW_01] Andreas Gärtner, Nutzungsbedingungen virtuelle Maschinen (VM), https://howto.informatik.hs-augsburg.de/index.php?title=Nutzungsbedingungen_virtuelle_Maschinen_(VM)&oldid=389, 2024-06-26, 2024-06-17, S. 2

[MW_02] Mehrere Autoren, Advanced Configuration and Power Interface, https://de.wikipedia.org/w/index.php?title=Advanced_Configuration_and_Power_Interface&oldid=229877761, 2024-06-27, 2023-01-15, S. 1

[MW_03] Mehrere Autoren, Clock-Gating, https://de.wikipedia.org/w/index.php?title=Clock-Gating&oldid=222280351, 2024-06-26, 2022-04-22, S. 1

[MW_04] Mehrere Autoren, Introduction to the server side, https://developer.mozilla.org/en-US/docs/Learn/Server-side/First_steps/Introduction#what_can_you_do_on_the_server-side, 2024-06-27, 2024-01-01, S. 5

[MW_05] Mehrere Autoren, Nextcloud Server Sourcecode, https://github.com/nextcloud/server/tree/master, 2024-06-27, 2024-06-27

[MW_06] Mehrere Autoren, ESP32 Sleep Modes, https://docs.espressif.com/projects/esp-idf/en/stable/esp32/api-reference/system/sleep_modes.html?highlight=deep%20sleep, 2024-06-27, 2024-05-09, S. 1 - 6

[MW_07] Korthauer, Reiner: "Handbuch Lithium-Ionen-Batterien", Berlin, Heidelberg: Springer Berlin / Heidelberg, 2024-06-28, 2013, S. 237 - 247

[MW_08] Mehrere Autoren, Gebrauchsanweisung Kerpu Punktschweißgerät, https://m.media-amazon.com/images/I/91dqZ7Y1-TL.pdf, 2024-06-28, 2022-11-15, S. 4

[MW_10] Mehrere Autoren, E-Paper ESP32 Driver Board Demo Code, https://files.waveshare.com/upload/5/50/E-Paper_ESP32_Driver_Board_Code.7z, 2024-06-29, 2024-04-22

[MW_10] Mehrere Autoren, Quick hack to port the IT8951 display driver to Arduino , https://github.com/clashman/it8951, 2024-06-29, 2019-11-14

[MW_11] Frin, Yvonnick, node-html-to-image , https://github.com/frinyvonnick/node-html-to-image, 2024-06-30, 2023-08-05

[MW_12] Mehrere Autoren, React , https://de.wikipedia.org/w/index.php?title=React&oldid=246001260, 2024-06-30, 2024-06-17

[MW_13] Mehrere Autoren, SQL , https://de.wikipedia.org/w/index.php?title=SQL&oldid=239068065, 2024-06-30, 2023-11-23

[MW_14] Mehrere Autoren, Objektrelationale Abbildung , https://de.wikipedia.org/w/index.php?title=Objektrelationale_Abbildung&oldid=230015918, 2024-06-30, 2023-01-19

[MW_15] Mehrere Autoren, PostgreSQL , https://de.wikipedia.org/w/index.php?title=PostgreSQL&oldid=244163584, 2024-06-30, 2024-04-18

[STA_01] https://wiki.seeedstudio.com/xiao_esp32s3_getting_started/

[STA_02] https://wiki.seeedstudio.com/XIAO_ESP32C3_Getting_Started/

[STA_03] https://cdn.shopify.com/s/files/1/1509/1638/files/Betriebsanleitung-AZ-D1miniV1.2_2.pdf?v=1590603445

[STA_04] https://www.st.com/resource/en/datasheet/stm32wl55cc.pdf

[STA_05] https://cdn-reichelt.de/documents/datenblatt/A300/RASPBERRY_PI_PICO_DB_EN.pdf

[STA_06] https://files.seeedstudio.com/wiki/Seeed-Studio-XIAO-ESP32/Low_Power_Consumption.pdf

[STA_07] https://www.glyn.de/produkte/displays/epaper-displays/wie-funktionieren-epaper/

[STA_08] https://www.orientdisplay.com/de/knowledge-base/lcd-basics/bistable-lcd/#:~:text=Bistabiles%20LCD%20kann%20ein%20Bild,wenn%20die%20Stromversorgung%20unterbrochen%20wird

[STA_09] https://www.orientdisplay.com/de/knowledge-base/oled-basics/how-does-oled-work/#:~:text=OLED%20funktioniert%20wie%20eine%20LED,Materialien%20sind%20Gelb%20und%20Blau.

[STA_10] https://www.proofpoint.com/de/threat-reference/wifi

[STA_11] https://de.wikipedia.org/wiki/ZigBee

[STA_12] https://www.lora-wan.de/

[STA_14] https://paulbourke.net/dataformats/bitmaps/

[STA_15] https://en.wikipedia.org/wiki/Color_depth

[STA_16] https://de.wikipedia.org/wiki/Dithering_(Bildbearbeitung)

[STA_17] https://de.wikipedia.org/wiki/Portable_Network_Graphics#Technische_Details

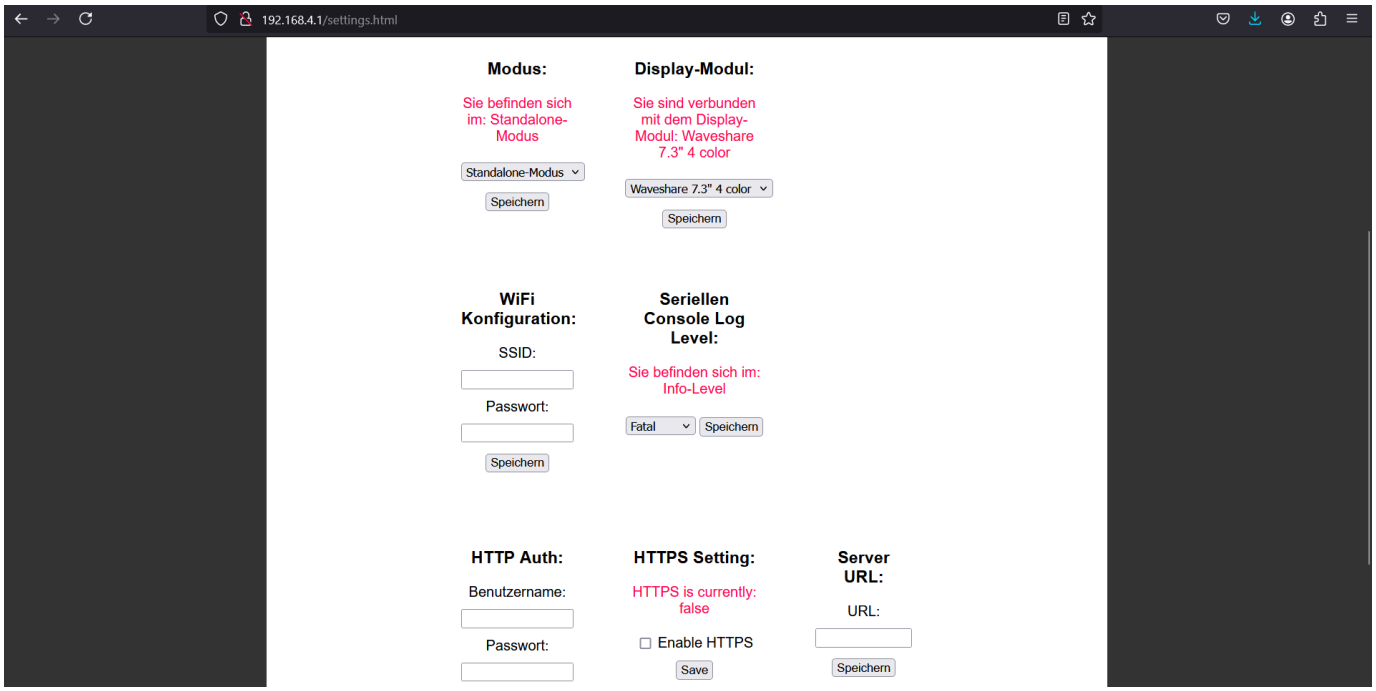
[STA_18] https://www.zlib.net/manual.html

12. Anhang 1: Handbuch

12.1 Erste Schritte und Display Inbetriebnahme

Julia Reuter

1. Aufwecken des Display-Moduls per Knopfdruck am unteren Gehäuserand, damit es seinen Access-Point öffnet
2. Verfügbare WLANs checken und "THA-LPRD-..." (Endung je nach Modul) auswählen und mit dem Default Passwort: "password" verbinden
3. Im Browser die Adresse 192.168.4.1/index.html aufrufen. Sie werden sich vermutlich beim ersten Aufrufen authentifizieren müssen. Default-Benutzer: "admin" und Passwort: "admin"
4. Nun sollten Sie auf diese Settings Page gelangen, wo Sie folgendes einstellen können:
    - **Modus:** Festlegen des Betriebsmodus (Standalone ist der Default)
    - **Displaymodul:** Auswählen des gerade verbundenen Display-Moduls
    - **WiFi-Konfiguration:** Standlone: Konfigurieren der WLAN eigenen Access-Points, Netzwerk- und Servermodus: Zugangsdaten des bestehenden WLAN-Netzwerks
    - **Serielles Konsolen Log Level:** Für Debugging können verschiedene Konsolennachrichten aktiviert werden
    - **HTTP Auth:** Benutzerauthentifizierung (Festlegen Benutzerpasswort und Zugangsdaten der Benutzeroberfläche)
    - **HTTPS-Setting:** Über Hochladen einer Zertifikatsdatei kann die Benutzeroberfläche verschlüsselt werden
    - **Server URL:** Einfügen einer Server URL für den Servermodus



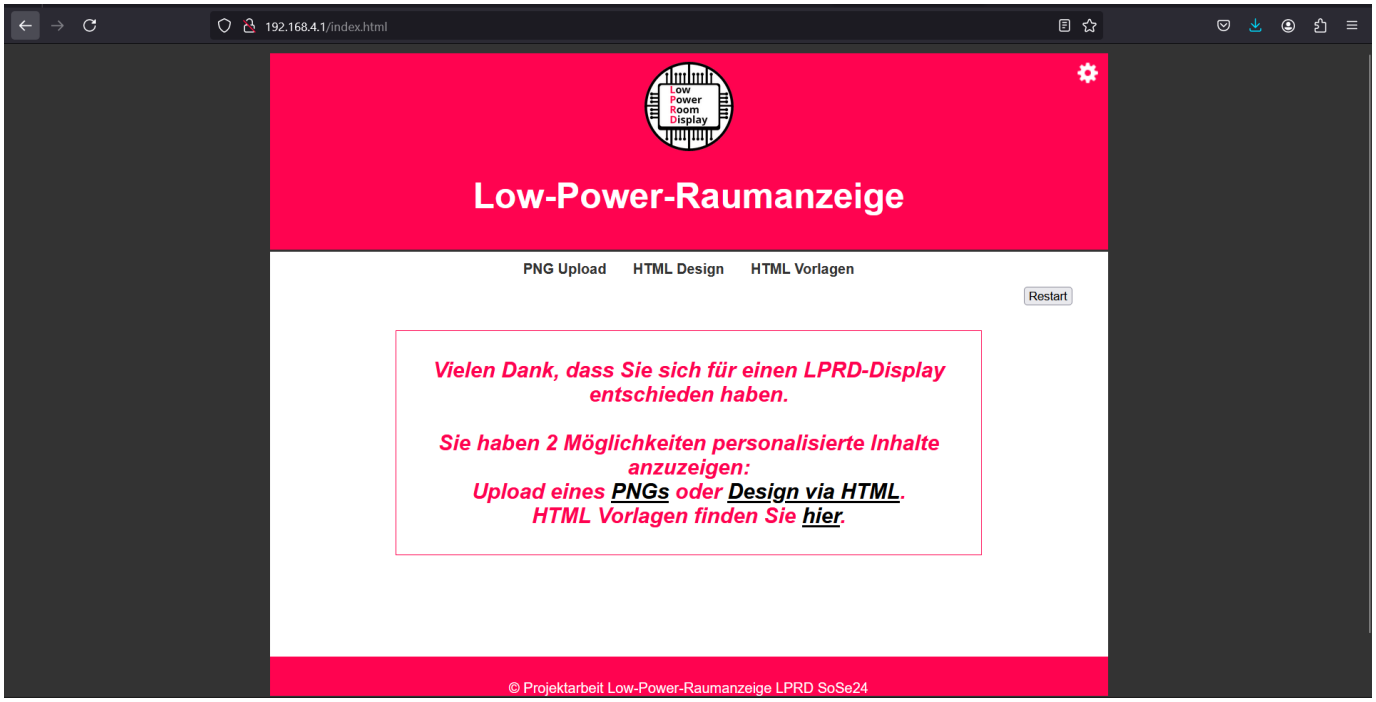
Alle geänderten Einstellungen müssen jeweils gespeichert werden. Über den Restart-Button wird das System neu gestartet, damit die Änderungen übernommen werden.
Bitte haben Sie hier einige Minuten Geduld. Falls das Verbinden mit einem externen WLAN oder Server nicht funktioniert hat, so öffnet das Display automatisch wieder nach ca. 5 min den eigenen Access Point und Sie können entweder die Konfiguration erneut versuchen oder fahren im Standalone-Modus fort.

Natürlich können die Einstellungen jederzeit geändert werden.

12.2 Hochladen von Inhalten

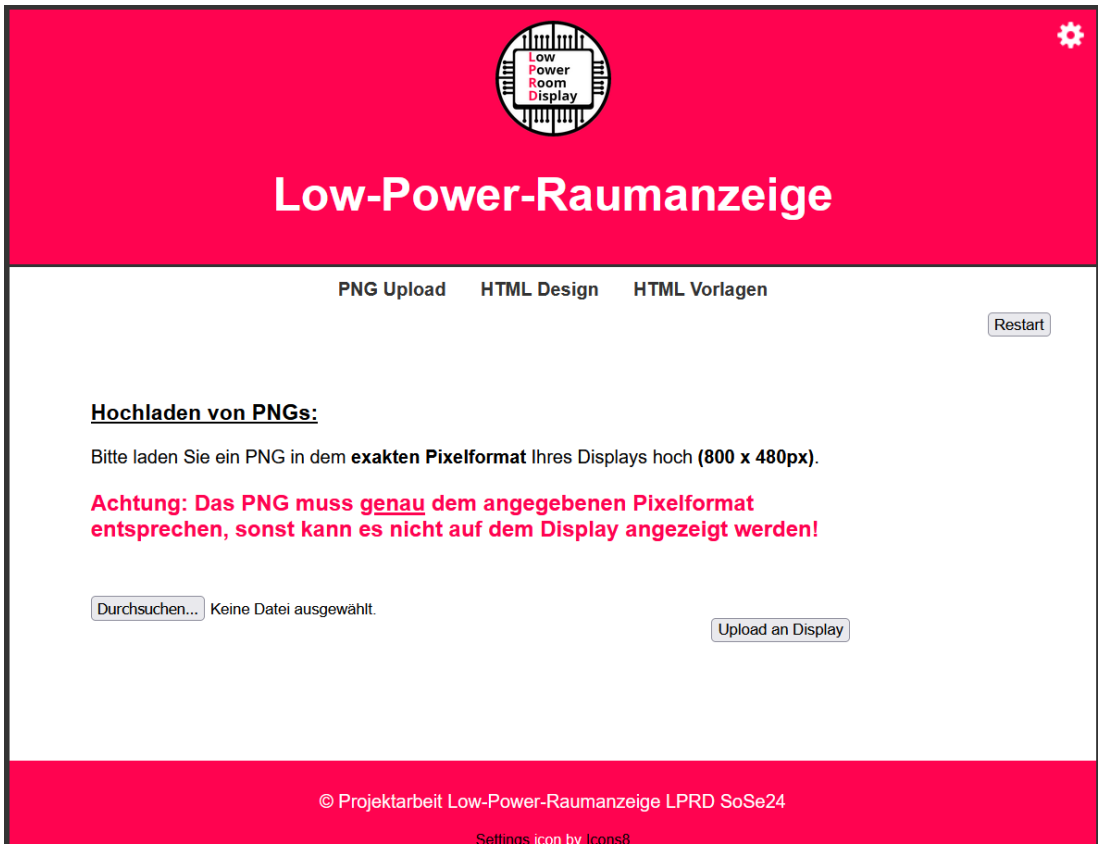
Julia Reuter

Wenn Sie das System erfolgreich neu gestartet haben, sollten Sie auf folgende Seite (Abbildung 3) gelangen und können Sie mit dem Upload fortfahren

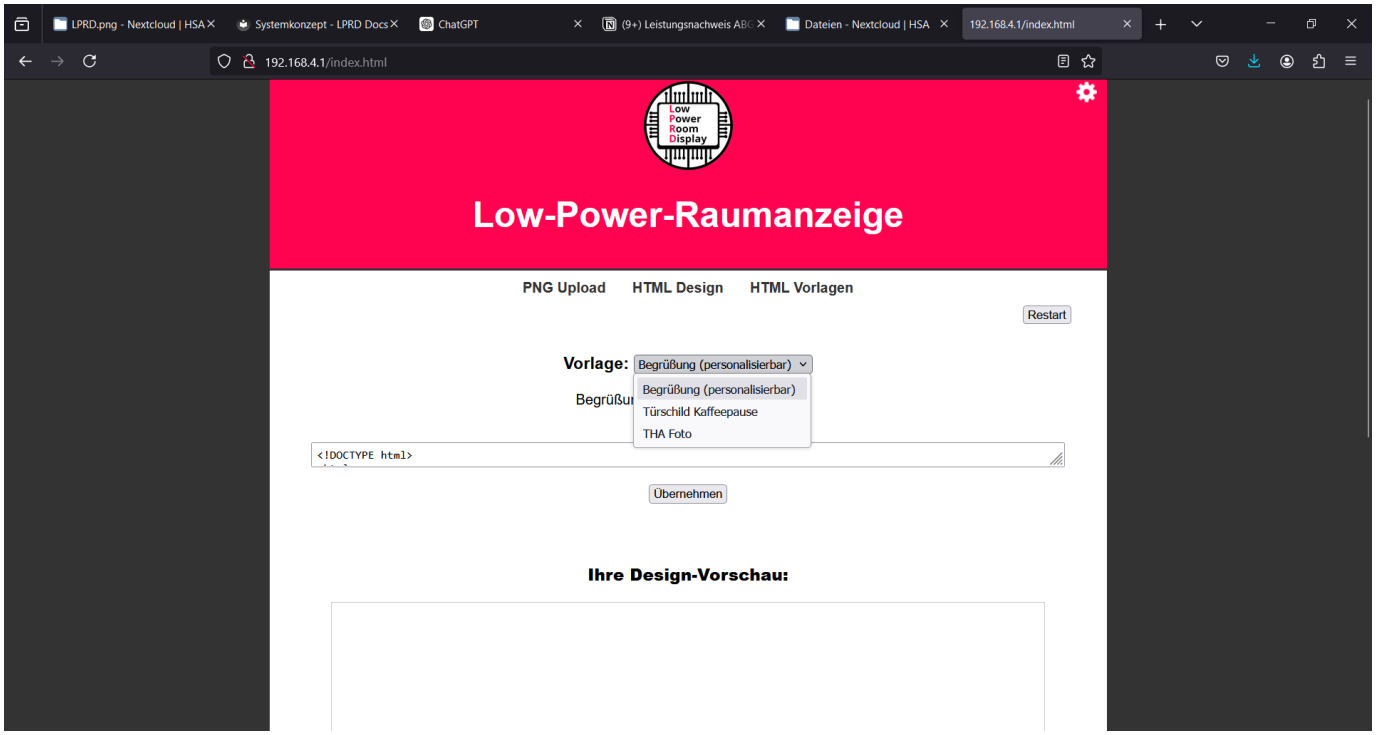


Hierfür stehen zwei Optionen zur Verfügung:

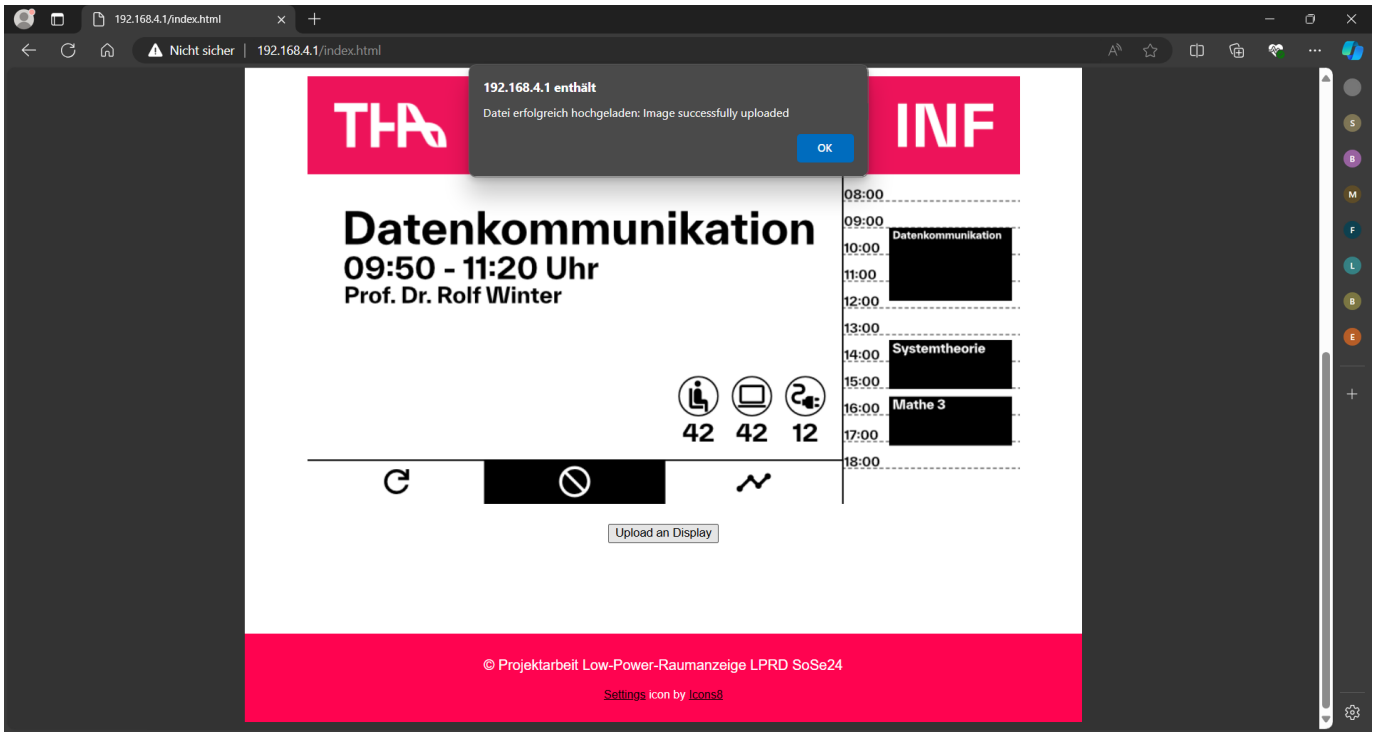
- **PNG-Upload** (Achtung: die Pixelauflösung des PNGs muss der Pixelauflösung des verbundenen Displays entsprechen)
  - Die Displayauflösung wird direkt beim Aufrufen an die Seite übermittelt und sollte angezeigt werden.
- **HTML-Design:** In dem Textfeld kann mithilfe von HTML Code ein Layout erstellt werden, was von der Webseite automatisch zu einem Bild konvertiert und an das Display geschickt wird.



- **HTML-Vorlagen:** Es können auch bereits vorgefertigte HTML-Vorlagen verwendet werden.



War der Upload erfolgreich, so werden Sie benachrichtigt und das Bild sollte nach wenigen Sekunden auf dem Display erscheinen.



Denken Sie daran, dass nach jedem erfolgreichen Bild-Upload das System in den deep-sleep gesetzt wird und nicht mehr erreichbar ist. Um den Inhalt zu ändern oder Einstellungen vorzunehmen, drücken Sie bitte den Knopf am unteren Display-Rand.

12.3 Netzwerkmodus

Julia Reuter
Nachdem Sie in den Einstellungen unter dem Punkt WiFi Konfiguration ihre WLAN-Daten eingeben und nach dem Speichern, das Display neu gestartet haben, sollte sich Ihr Modul automatisch mit dem entsprechenden WLAN verbinden.
Unter der zugewiesenen IP: …/index.html können Sie mit dem Upload wie gehabt fortfahren.

12.4 Aufsetzen des Linux Servers

Mario Wegmann

12.4.1 Installation

In der folgenden Anleitung werden Platzhalter mit `<Platzhaltername>` markiert.

Zu Beginn sollte ein Reverseproxy installiert werden, dies ist kein absolutes Muss, aber für weitere Funktionalitäten, wie HTTP BasicAuth und SSL-Zertifikate, durchaus hilfreich. Hier wurde der Caddy Reverse Proxy verwendet, aber auch Nginx oder Apache können problemlos eingesetzt werden.

Um Caddy zu installieren, wird das Repository von den Caddy Maintainern hinzugefügt.

```
sudo apt install -y debian-keyring debian-archive-keyring apt-transport-https curl
curl -1sLf 'https://dl.cloudsmith.io/public/caddy/stable/gpg.key' | sudo gpg --dearmor -o /usr/share/keyrings/caddy-stable-archive-keyring.gpg
curl -1sLf 'https://dl.cloudsmith.io/public/caddy/stable/debian.deb.txt' | sudo tee /etc/apt/sources.list.d/caddy-stable.list
```

Anschließend können die Repositorys neu geladen und Caddy installiert werden.

```
sudo apt-get update
sudo apt-get install caddy
```

Der Caddy Reverseproxy wird über eine Konfigurationsdatei verwaltet, diese sollte sich unter '/etc/caddy/Caddyfile' befinden. Mit dem folgenden Code wird ein einfacher Reverseproxy auf Port :3000 der Next.js Anwendung eingerichtet, dabei wird ein lokal liegendes SSL-Zertifikat verwendet und eine Basic Auth Authentifizierung.
Den Passwort Hash kann man direkt mit Caddy generieren, indem `caddy hash-password` verwendet wird, diese Hilfsfunktion fragt nach dem Passwort und gibt dann den Hash aus, welcher in der Caddyfile eingetragen werden kann.

```
lprd.informatik.tha.de {
        tls /etc/ssl/certs/lprd-fullchain.pem /etc/ssl/priavte/lprd-privkey.pem
        basicauth * {
                <Benutzername> <Passwort Hash>
        }
        reverse_proxy 127.0.0.1:3000
}
```

Danach muss Node.js installiert werden. Da über den apt Paketmanager nur sehr langsam aktuelle Versionen von Node.js bereitgestellt werden, wird hier der Node Version Manager (nvm) verwendet.

Dieser lässt sich über folgendes Bash Skript installieren.

```
curl -o- https://raw.githubusercontent.com/nvm-sh/nvm/v0.39.7/install.sh | bash
```

Nach der Installation empfiehlt es sich, die SSH-Sitzung neu zu starten, damit der `nvm` Befehl auch erkannt wird. Anschließend kann mit `nvm install 20` die LTS-Version Node.js 20 installiert werden. Mit Node.js kommt auch der `npm` Paketmanager. Mit diesem lässt sich React und Next.js installieren.

```
npm install next@latest react@latest react-dom@latest
```

Als Nächstes wird `git` installiert, um den Sourcode herunterladen zu können.

```
sudo apt-get install git
```

Danach muss ein SSH Key für Github hinterlegt werden, damit das Repository geklont werden kann. Dafür wird als Erstes ein SSH Key generiert. Es empfiehlt sich dabei auch, eine Passphrase zu vergeben und beim Speicherort einen eindeutigen Namen zu vergeben.

```
ssh-keygen -t ed25519 -C "<Github E-Mail Adresse>"
```

Als Nächstes wird der SSH Agent einmal gestartet, um anschließend den SSH Key beim SSH Agent hinterlegen zu können.

```
eval "$(ssh-agent -s)"
```

Beim Hinzufügen muss auch die zuvor festgelegte Passphrase eingegeben werden.

```
ssh-add <Speicherort des zuvor erstellten SSH Keys angeben>
```

Der Public Teil des SSH Keys muss auch im Github Account hinterlegt werden. Dafür die URL https://github.com/settings/keys aufrufen und einen neuen SSH Key hinzufügen.

Nun kann das Repository der Webanwendung geklont werden.

```
git clone git@github.com:THA-LPRD/web.git lprd
```

Danach wird das Verzeichnis von der Next.js Anwendung betreten.

```
cd lprd/lprd-server
```

Die Anwendung verwendet die offizielle Schriftart der THA. Die Schriftart TWK Everett darf, wegen der von der THA erworbenen, Lizenz jedoch nur Studierenden und Mitarbeitern der Technischen Hochschule Augsburg zur Verfügung gestellt werden. Die Schriftart lässt sich unter folgender URL herunterladen: https://www.tha.de/Kommunikation/Corporate-Design.page Anschließend müssen die zwei Schriftfamilien TWKEverett-Bold-web.ttf und TWKEverett-Medium-web.ttf TWKEverett-Medium-web.ttf im Ordner lprd-server/components/fonts kopiert werden.

Als Nächstes kann die PostgreSQL-Datenbank aufgesetzt werden. Dafür kann das Docker Image oder auch ein bestehender PostgreSQL Server verwendet werden.

```
sudo docker run --name lprd-postgres -v pgdata:/var/lib/postgresql/data -p 5432:5432 -e
POSTGRES_PASSWORD=<SicherersDatenbankPasswort> -d postgres
```

Die URL der Datenbank muss auch noch der Next.js Anwendung bekannt gemacht werden, dies geschieht über die .env Datei. Diese sollte neu angelegt werden und den folgenden Inhalt enthalten:

```
DATABASE_URL="postgresql://<PostgresUser>:<PostgresUserPassword>@<PostgresServerIP/Hostname>:5432"
```

Nach der Installation der Datenbank kann diese mit Prisma mit den notwendigen Tabellen befüllt werden.

```
npx prisma migrate dev --name init
```

Für die Konvertierung von HTML zu PNG wird die Library node-html-to-image verwendet. Diese nutzt Puppeteer als Basis für die Konvertierung. Beim direkten Testen auf dem Server kann es vorkommen, dass Puppeteer nicht startet, da es weitere Abhängigkeiten hat. Diese fehlenden Abhängigkeiten können mit dem folgenden Befehl herausgefunden werden:

```
ldd /home/<USER>/.cache/puppeteer/chrome/linux-<VERSION>/chrome-linux64/chrome | grep not
```

Im Fall der hier verwendeten Debian Installation mussten folgende Pakete noch zusätzlich installiert werden.

```
sudo apt-get install libgbm1 libasound2 libxkbcommon0 libatk-bridge2.0-0 libnss3
```

Die Vorbereitungen sind somit abgeschlossen und der Developmentserver kann gestartet werden.

```
npm run dev
```

Unter http://localhost:3000 ist der Server nun erreichbar.

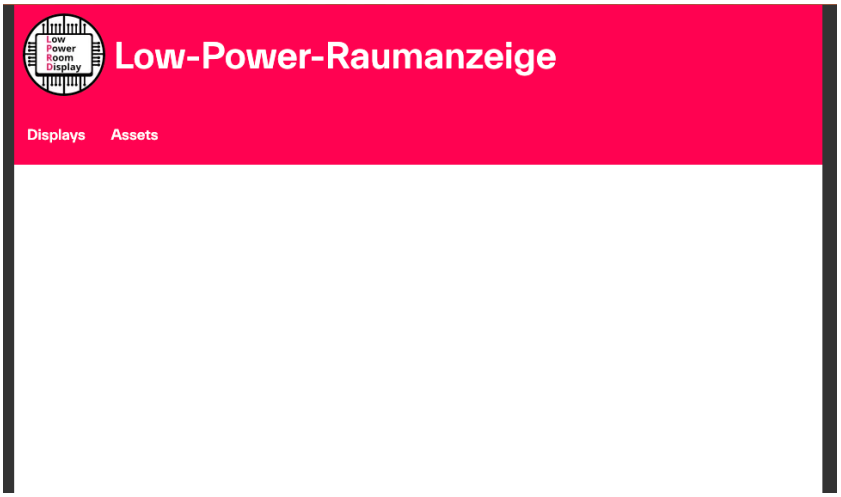
Der Developmentserver kann mit dem Tastenkürzel Ctrl+C beendet werden.

12.5 Verwenden der Webanwendung

Mario Wegmann

12.5.1 Unterseiten

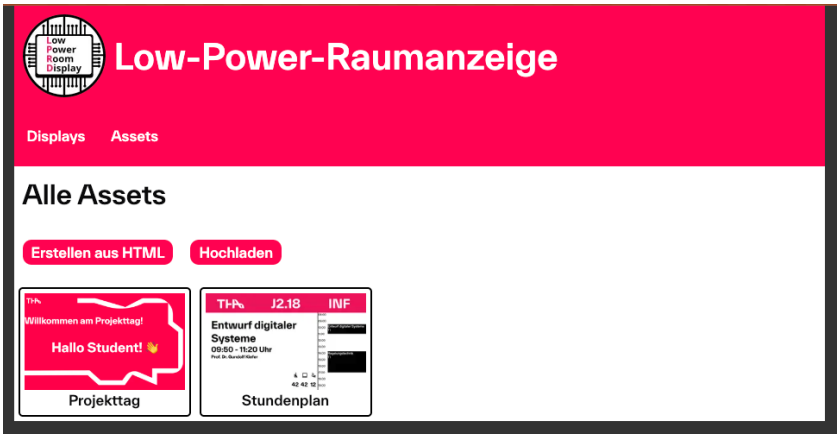
Nachdem die Anwendung gestartet wurde, kann die Adresse im Browser eingegeben werden. Die Startseite, wie in ersichtlich, zeigt das Menü für die Unterseiten an. Es gibt zwei Unterseiten, unter »Displays« werden alle bereits bekannten Displays aufgelistet. In der Unterseite »Assets« werden alle vorhandenen Assets angezeigt.



Die Startseite der Webanwendung

12.5.2 Neues Asset erstellen

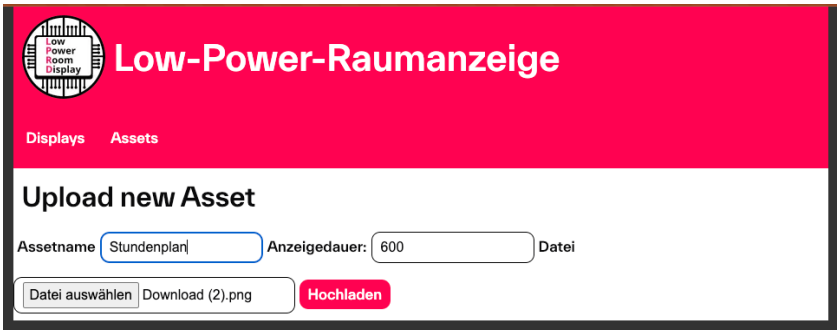
Es gibt zwei Möglichkeiten, ein Asset zu erstellen, es kann entweder ein fertiges PNG hochgeladen werden oder ein PNG aus einem HTML-Code erstellt werden. Dazu kann in der »Assets«-Unterseite, welche in ersichtlich ist, einer der beiden Vorgänge über die Knöpfe gestartet werden.



Eine Übersicht über alle vorhandenen Assets

PNG hochladen

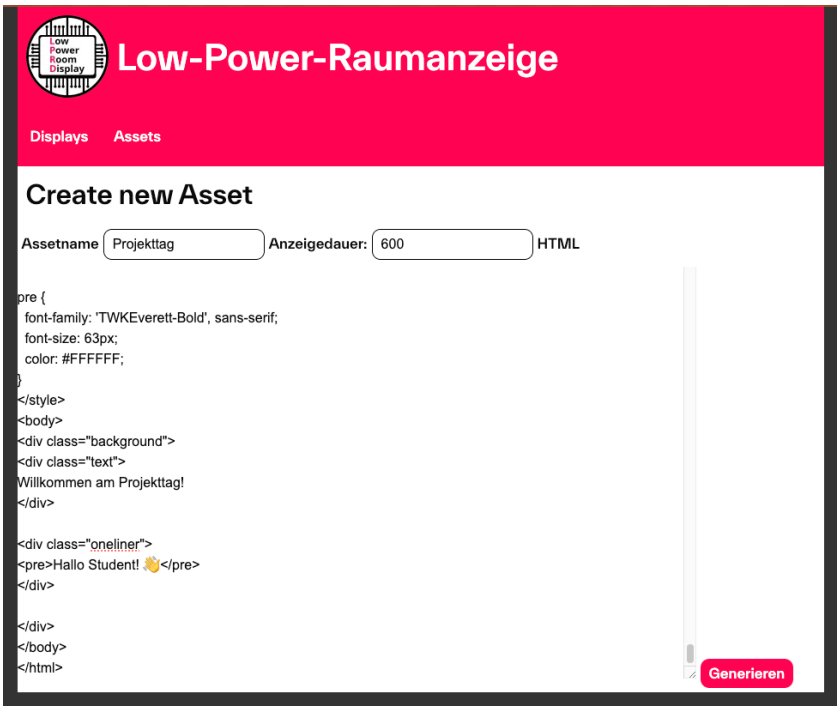
Ein PNG kann auf der Unterseite Assets über den »Hochladen«-Knopf hochgeladen werden. Beim Hochladen muss darauf geachtet werden, dass das PNG in der richtigen Auflösung für das Display ist, welches später das Asset anzeigen soll. Ebenso kann ein Name für das Asset und wie lange es angezeigt werden soll, angegeben werden. Das Hochladen beginnt mit einem Druck auf den »Hochladen«-Knopf. Die zeigt das Formular.



Das Formular zum Hochladen von bestehenden PNGs

PNG aus HTML erstellen

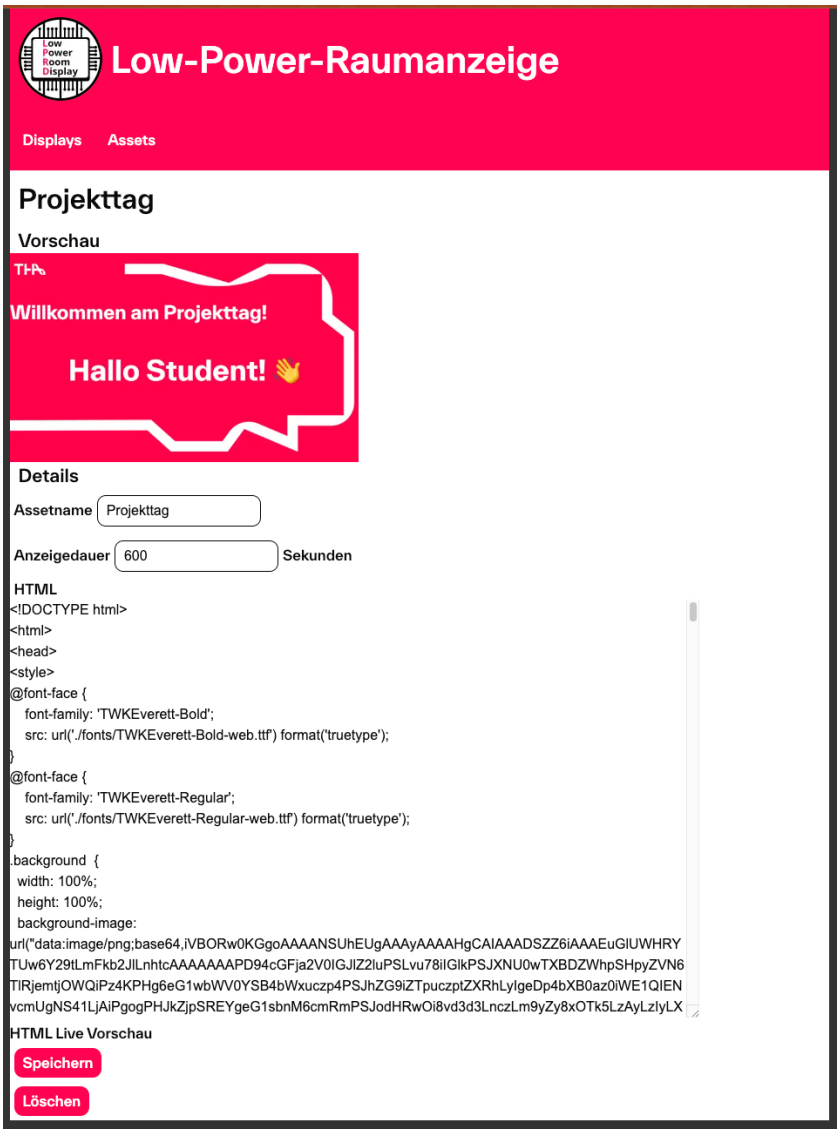
Auf der Unterseite Assets gibt es auch den »Erstellen aus HTML«-Knopf, dieser öffnet ein Formular, in dem der Assetname, die Anzeigedauer und der HTML-Code eingegeben werden kann. Nach dem Druck auf den »Generieren«-Knopf wird ein PNG erstellt und lokal abgelegt. Die zeigt das Formular.



Das Formular zum Erstellen von PNGs aus HTML Code

12.5.3 Vorhandenes Asset bearbeiten

In der »Assets« Übersichtsseite kann jedes vorhandene Asset ausgewählt werden, um eine Detailansicht, welche in ersichtlich ist, anzuzeigen. Dort können der Name, die Anzeigedauer und der HTML-Code bearbeitet werden. Ebenso kann ein Asset auch gelöscht werden.



Die Detailansicht eines Assets

12.5.4 Neues Displaymodul verbinden

Beim erstmaligen Einschalten eines Displaymoduls, nach dem Flashen, kann in der Einrichtungsseite des Displaymoduls der Servermodus als Betriebsmodus ausgewählt werden. Ebenso kann die Server URL im dazugehörigen Textfeld angegeben werden.

Die weiteren Einstellungen verhalten sich gleich, wie auch bei der Einrichtung des Netzwerkmodus bereits erläutert wurde.

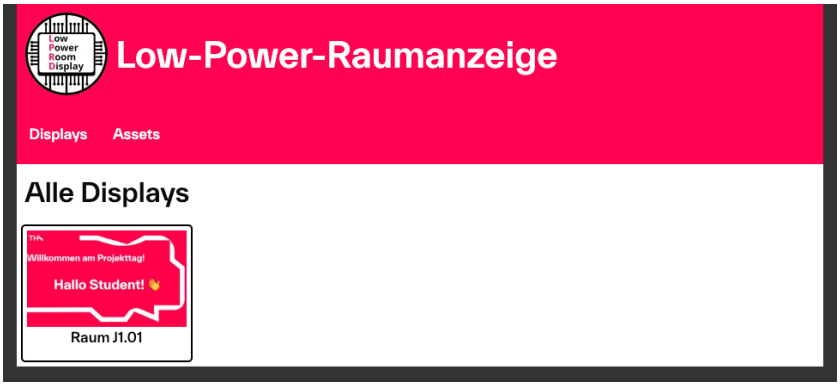
Nachdem die Erstkonfiguration gespeichert und das Displaymodul neu gestartet wurde, versucht es sich automatisch mit dem eingetragenen WLAN und anschließend mit dem eingetragenen Server zu verbinden.

Falls das Displaymodul sich zum ersten Mal mit diesem Server verbindet, wird es automatisch in der Datenbank eingetragen, anschließend geht das Display in den Deep-sleep Modus.

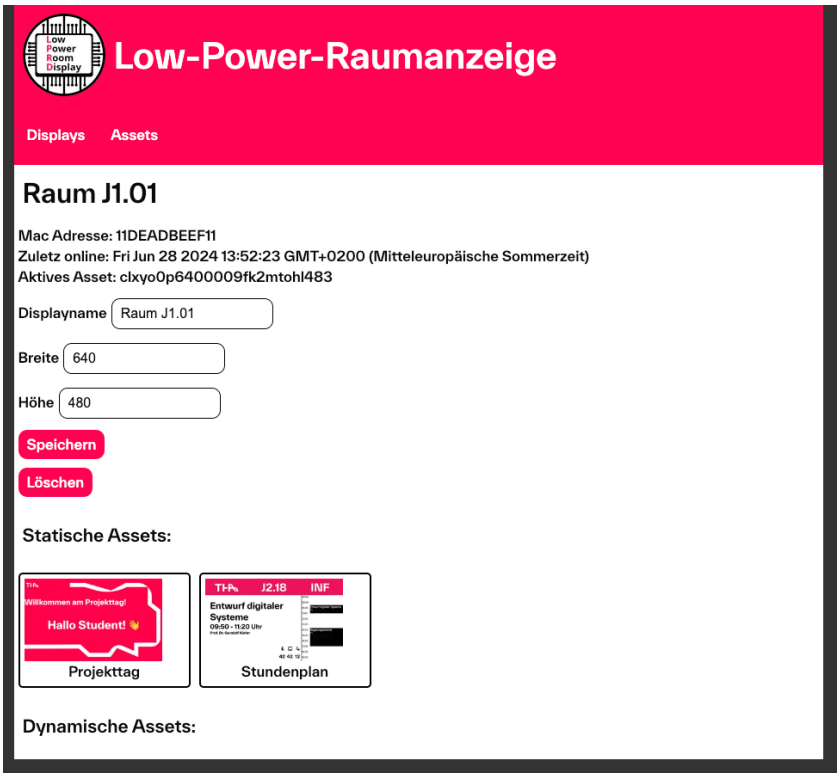
12.5.5 Vorhandenes Display bearbeiten

Ähnlich wie bei der Assets-Übersicht können auch die Displays in der »Display«-Unterseite, welche in dargestellt ist, bearbeitet werden. In der Detailansicht, diese ist in zu erkennen, eines Displays kann die MAC-Adresse und der Zeitpunkt, zu dem sich das Display zum letzten Mal beim Server gemeldet hat, eingesehen werden. Der Name des Displays und die Auflösung können bearbeitet werden.

Zuletzt kann ein Display auch wieder gelöscht werden.



Die Übersichtsseite aller Displays



Die Detailansicht eines Displays

12.5.6 Neues Asset auf dem Display anzeigen

Sobald ein Displaymodul sich in der Webanwendung registriert hat, kann in der Webanwendung das Display ausgewählt werden. Hier können grundlegende Informationen zum Display bearbeitet und ein neues aktives Asset gesetzt werden.

Mit einem Tastendruck auf den vorderseitigen Knopf kann das Display aus dem Deep-sleep Modus wieder aufgeweckt werden. Immer nach dem Aufwecken versucht das Displaymodul, sich mit dem Server zu verbinden und eine neue Konfiguration zu erhalten. Die Konfiguration enthält die URL des aktuell anzuzeigenden Assets und eine Dauer in Sekunden, die angibt, wie lange das Asset angezeigt werden soll.

Das Displaymodul lädt anschließend das Asset von der angegebenen URL herunter, zeigt es an und versetzt sich anschließend wieder in den Deep-sleep Modus. Zusätzlich wird noch ein Wakeup Timer auf die angegebene Dauer gesetzt. Dieser Timer weckt somit das Displaymodul automatisch nach Ablauf der Anzeigedauer auf, damit sich das Displaymodul ein neues anzuzeigendes Asset herunterlädt.

12.6 Arbeiten an der Displaymodul-Firmware

Ahmet Emirhan Göktas

Um an diesem Projekt zu arbeiten, müssen folgende Werkzeuge auf Ihrem Computer installiert sein:

- PlatformIO
- Ein Code-Editor (z.B. Visual Studio Code)
- Git
- Verfügbarer USB-Anschluss (zum Flashen der Firmware)

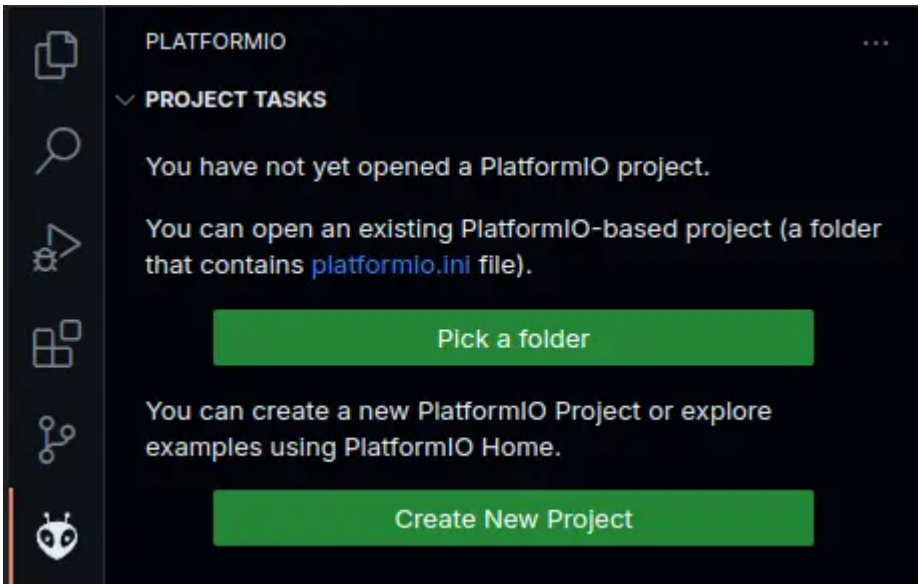
12.6.1 Einrichten des Projekts

Hier ist eine Schritt-für-Schritt-Anleitung, wie Sie das Projekt auf Ihrem lokalen Rechner für Visual Studio Code einrichten.

1. Installieren Sie die PlatformIO-Erweiterung für Visual Studio Code. Sie können die Erweiterung finden, indem Sie im Erweiterungstab nach `PlatformIO IDE` suchen.
2. Klonen Sie das Repository auf Ihren lokalen Rechner:

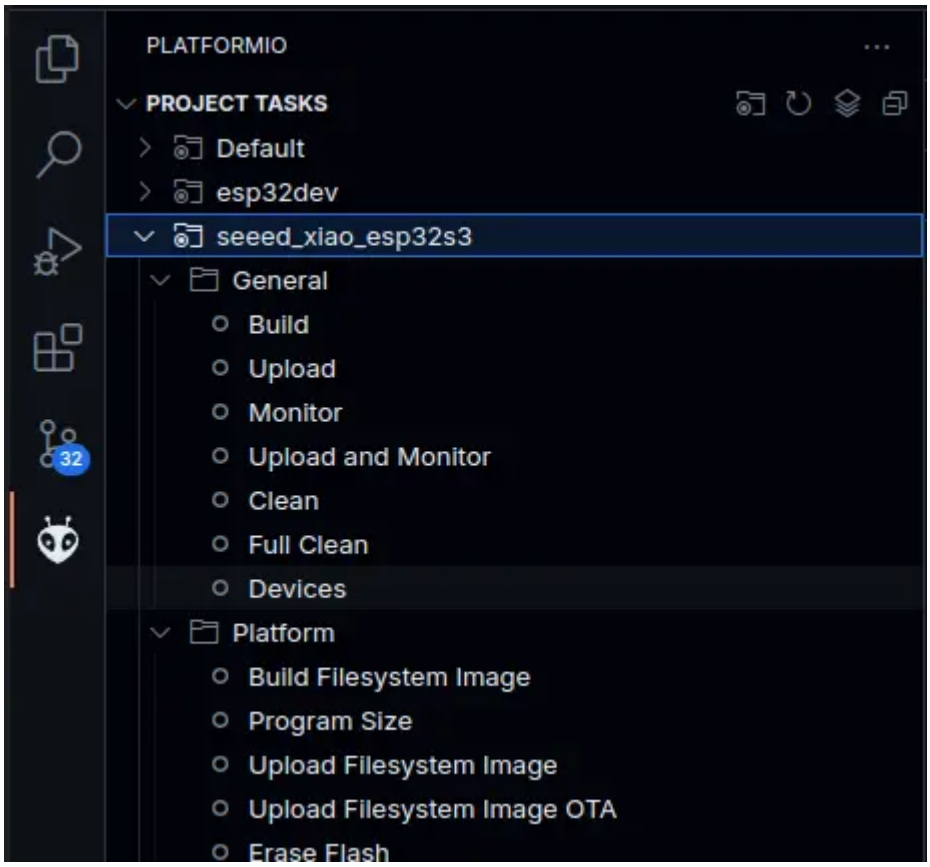
```
git clone --recursive https://github.com/THA-LPRD/MCU.git
```

3. Klicken Sie auf das PlatformIO-Symbol auf der linken Seite von Visual Studio Code. Dort sollten Sie eine Taste namens `Pick a folder` sehen. Klicken Sie darauf und wählen Sie den Ordner aus, in den Sie das Repository geklont haben.



PlatformIO Projekt öffnen

4. PlatformIO sollte das Projekt automatisch erkennen und die Umgebung für Sie konfigurieren. Nachdem Sie die richtige Umgebung ausgewählt haben, können Sie mit der Arbeit am Projekt beginnen.
5. Sie können die Firmware auf den ESP32 flashen, indem Sie auf das PlatformIO-Symbol auf der linken Seite von Visual Studio Code klicken und dann auf die Taste `Upload` klicken.
6. Wenn Sie die Ausgabe des seriellen Monitors sehen möchten, können Sie auf das PlatformIO-Symbol auf der linken Seite von Visual Studio Code klicken und dann auf die Taste `Serial Monitor` klicken.
7. Sie können auch die Taste `Upload File System Image` verwenden, um die Dateien im Ordner `data` auf den ESP32 hochzuladen.



ESP32 flashen

> **Note**
>
> Abhängig vom verwendeten ESP32-Board kann es notwendig sein, die Boot-Taste zu drücken, bevor der ESP32 an den USB-Anschluss angeschlossen wird.

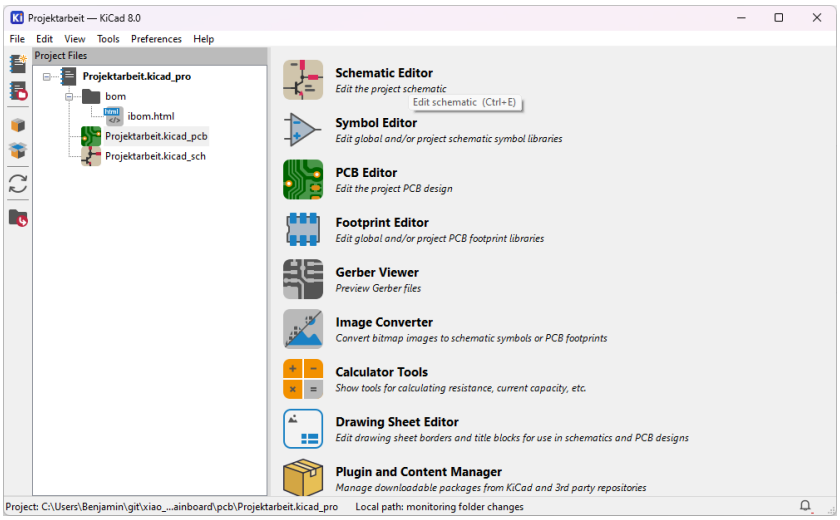
12.7 Bestellung von der Platine und benötigten Bauteilen

Benjamin Klaric

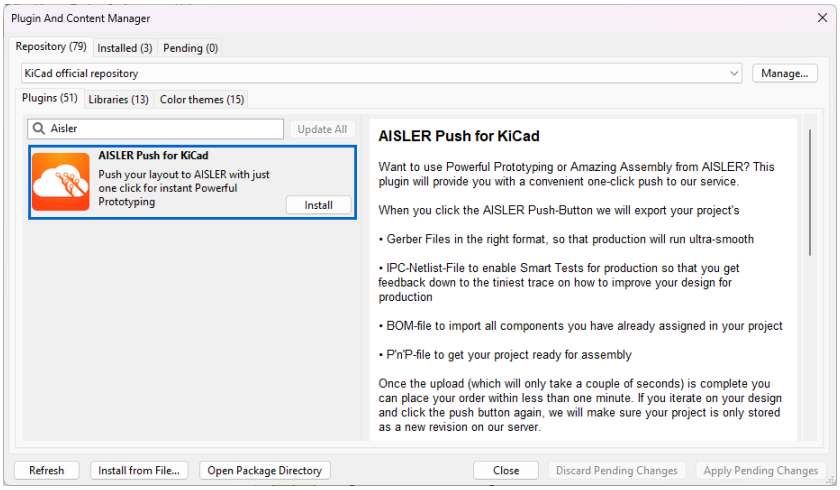
Um die Platine zu bestellen muss man das erstens das GitHub Repo, auf dem sich die Files von der Platine befinden, clonen.

```
git clone https://github.com/bklaric1/xiao_esp32_s3_mainboard.git
```

Auf dem Repo gibt es zwei Branches, nämlich den **main** und den **feature_subsheets**. Beide Branches können für das Bestellen von der Platine benutzt sein werden.
Nachdem soll man das ECAD Programm KiCad installieren, unter dem https://www.kicad.org/download/ Link. Man soll Version 8+ installieren, da die Files auf KiCad 8 erzeugt werden und sind nicht backwards kompatibel.
Man soll einfach den *.kicad_pro* File öffnen und es soll sich automatisch KiCad öffnen.
Unter den `Plugin and Content Manager`, die auf dargestellt ist, kann man ein Plugin für Aisler (oder JLCPCB) zu KiCad hinzufügen, um die Bestellung einfacher zu machen. Für Aisler sieht der Plugin so aus, wie es auf dargestellt ist.

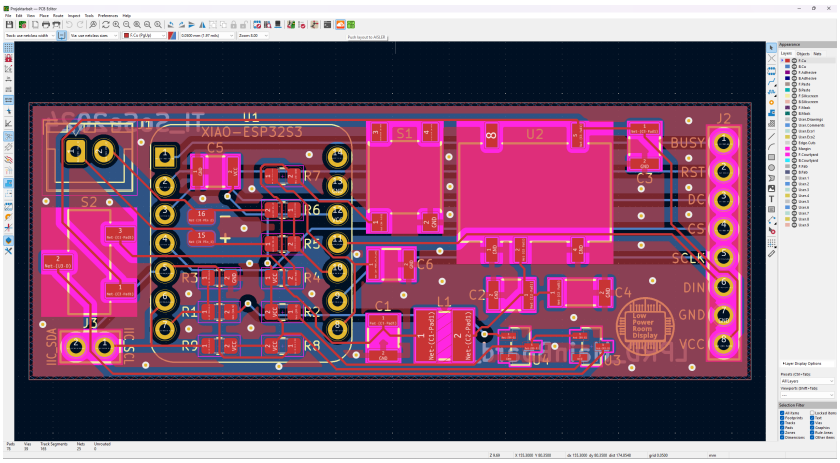


KiCad Fenster



Aisler Plugin

Nachdem die Installation erfolgreich war, erschiet ein neues Knopf in PCB Editor Fenster, oben mittig, wie auf zu sehen ist.



Knopf für Aisler Plugin

Wenn man darauf druckt, wird man auf die Webseite von Aisler genommen, mit schon angelegten Projekt für das Bestellung. Nachher soll man einfach die Schritten von Aisler Webseite folgen um die Platine zu bestellen.

Die Bauteilen kann man unter `/pcb/bom/ibom.html` sehen. Es öffnet sich ein Browser Fenster wo man interaktiv eine BOM Tabelle und die Platine sehen kann. Wenn man den Maus auf die Bauteile hinbewegt, wurde dementsprechenden Bauteilen auf die Platine in rot markiert sein werden.

12.8 Quellen

https://docs.github.com/en/authentication/connecting-to-github-with-ssh/generating-a-new-ssh-key-and-adding-it-to-the-ssh-agent?platform=linux#generating-a-new-ssh-key https://docs.github.com/en/authentication/connecting-to-github-with-ssh/adding-a-new-ssh-key-to-your-github-account

https://tha-lprd.github.io/Docs/v.1.0.0