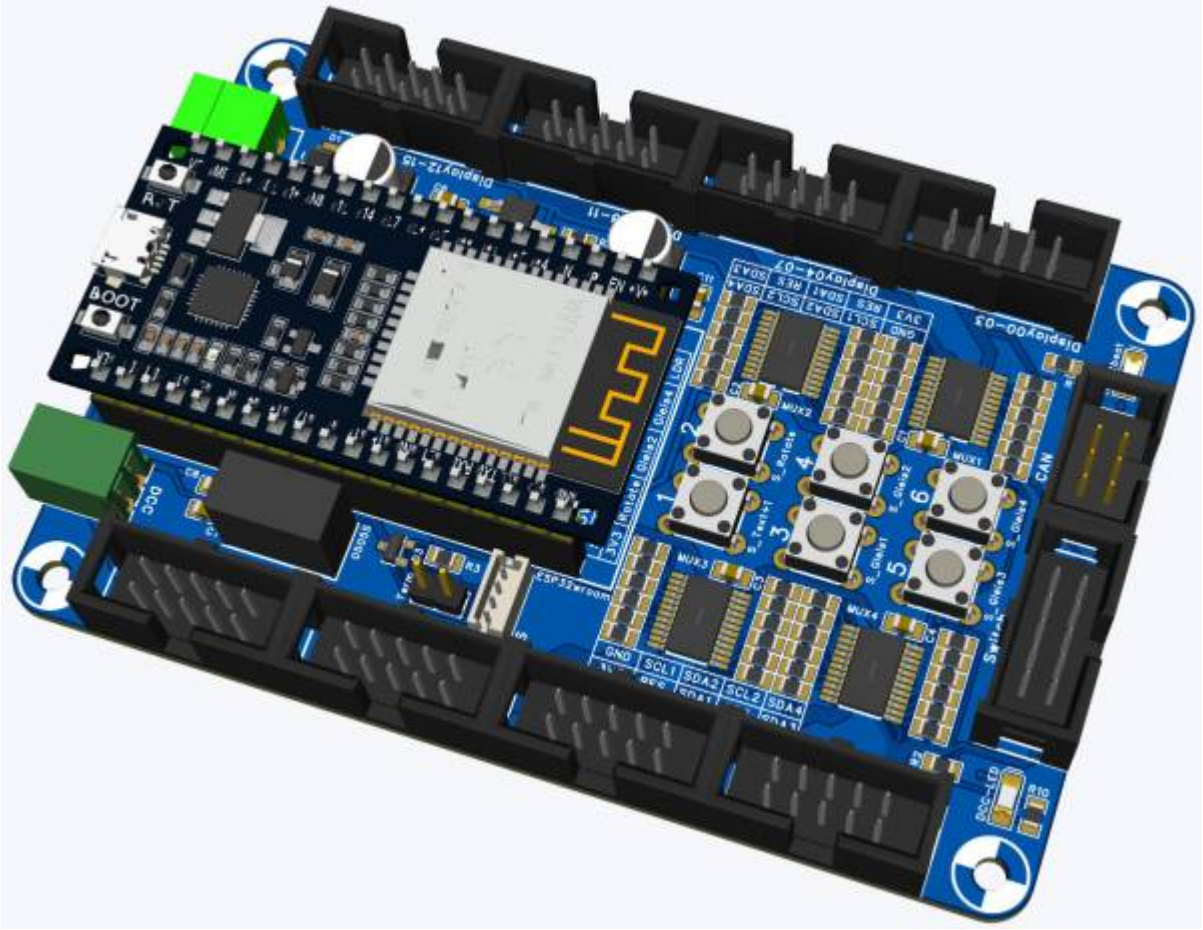
# 740 Display-Steuerung Zugzielanzeiger v3.0

von Michael

Zur alten Anleitung (v2.1): zur Vorgängerplatine

## Aller guten Dinge sind drei

Mit der ZZA Steuerung v3 wird aus den einzelnen Komponenten endlich ein vollständiges System. Basierend auf Ideen von Tobias, Klaus, und Fredddy entwickelte Hardi 2019 einen optimierten Sketch, mit dem es erstmals möglich war, 0,87“ und 0,91“ Monochrom-Displays per DCC-Befehl zu aktualisieren.




## Funktionsweise der ZZA-Steuerung v3

Über vier 8-Fach Multiplexer wechselt der ESP32 ca. alle 4 ms zum nächsten OLED Display. So benötigt er knapp 135 ms, um 32 Lauftexte zu aktualisieren. Das gelingt, weil nur das obere Viertel des Displays aktualisiert wird (Danke, Hardi).
**Ja, richtig gelesen:** Alle Gleise können gleichzeitig mit Lauftext versorgt werden, wenn mal wieder alle Züge mit Verspätung fahren.

Das Layout der Wannenstecker wurde beibehalten. So kann die neue Platine Plug & Play an

bestehende Displays angeschlossen werden.
Bei zukünftigen Installationen kann auf die zweite RST-Leitung verzichtet werden.
Die beiden SCL Leitungen sollten wegen der Pull Up Widerstände weiterhin verwendet werden, allerdings sind sie nicht mehr bestimmten SDA Leitungen zuzuordnen.

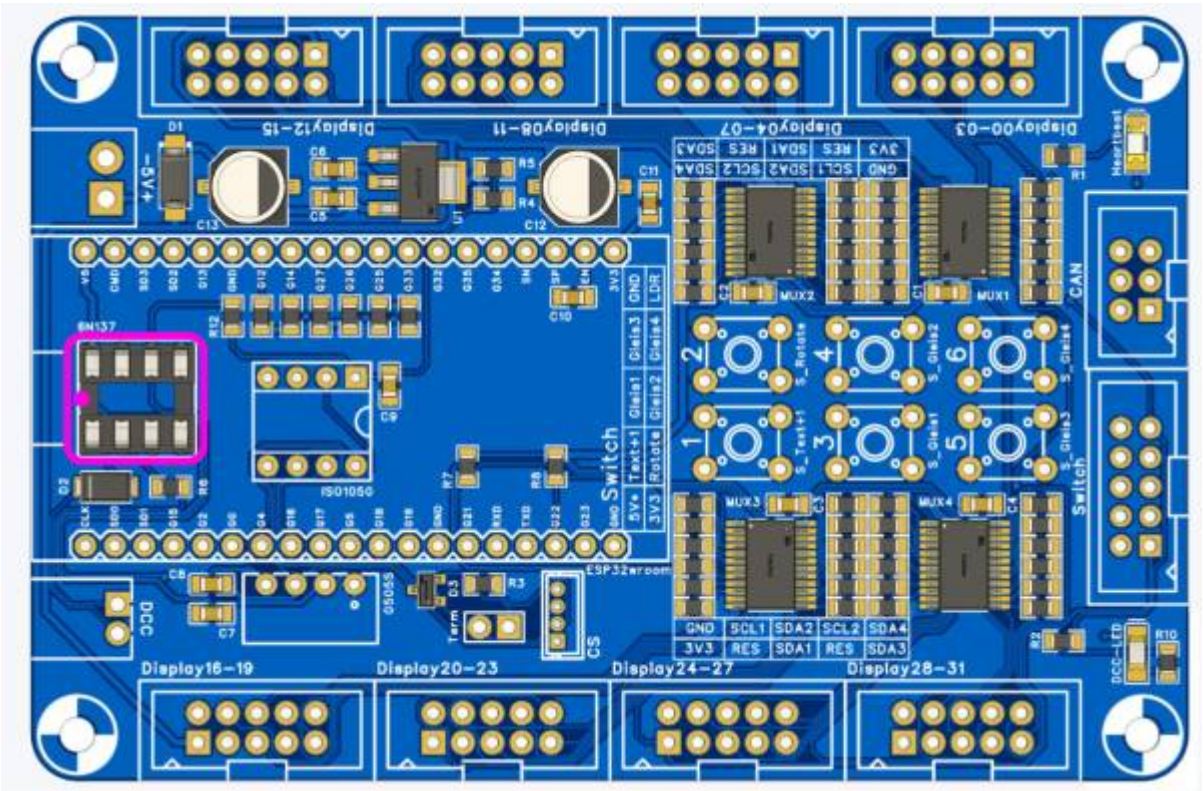
Neu ist auch die Funktion zur Bildung von Display-Gruppen, wodurch mehrere Displays mit nur einem DCC Befehl dasselbe Zugziel empfangen, egal, ob links- oder rechtsseitige Ausrichtung.
Was im alten Sketch der Befehl „Vorheriges/Nächstes OLED wählen“ war, wird somit zu „Vorherige/Nächste Gleisgruppe wählen“.

Gegenüber dem Vorgänger können nun endlich alle vorhandenen OLED-Adapter verwendet werden. Dazu gehören die Display-Adapter (750) in zwei- und vierfacher Ausführung sowie die Kofferplatine (760).

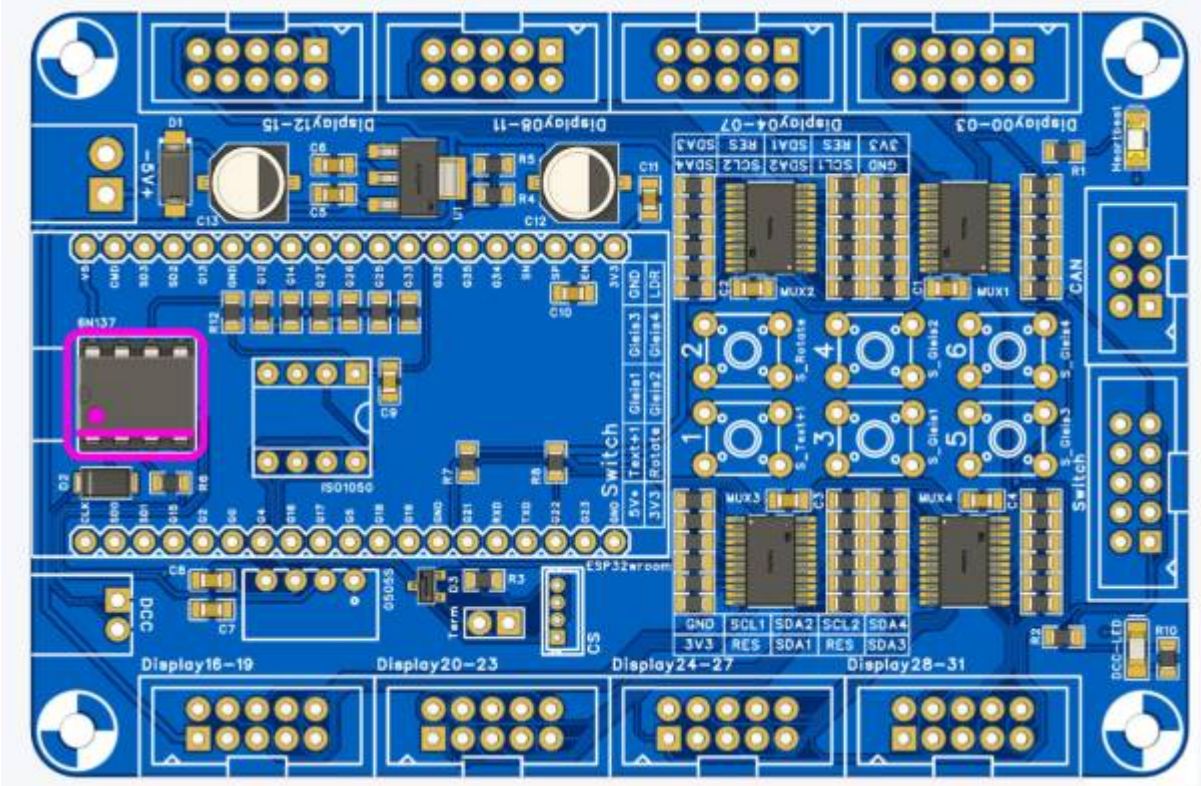
# Aufbau der Platine

Zum Aufbau gibt es nicht viel zu sagen. Die Platine ist nahezu komplett vorbestückt. Lediglich 21 Teile müssen gelötet werden und abgesehen von der Richtung kann hier prinzipiell nichts vertauscht werden.

**Schritt 1**
Als erstes wird die IC Fassung eingelötet. Dabei bitte auf die linksseitige Kerbe achten.

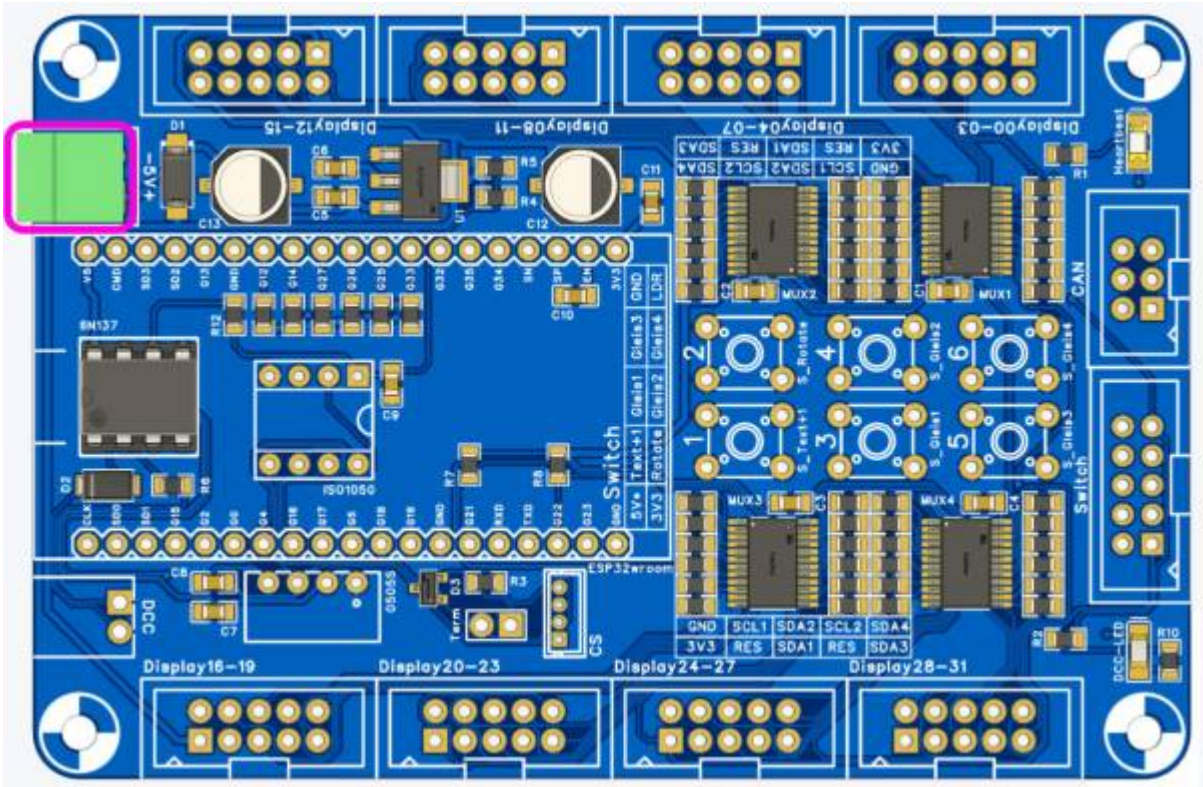



**Schritt 2**
Im Anschluss kann direkt der Optokoppler eingesteckt werden. Hier bitte unbedingt auf korrekte Position von Pin 1 achten. Der Punkt auf dem IC zeigt wie im Bild nach unten links.



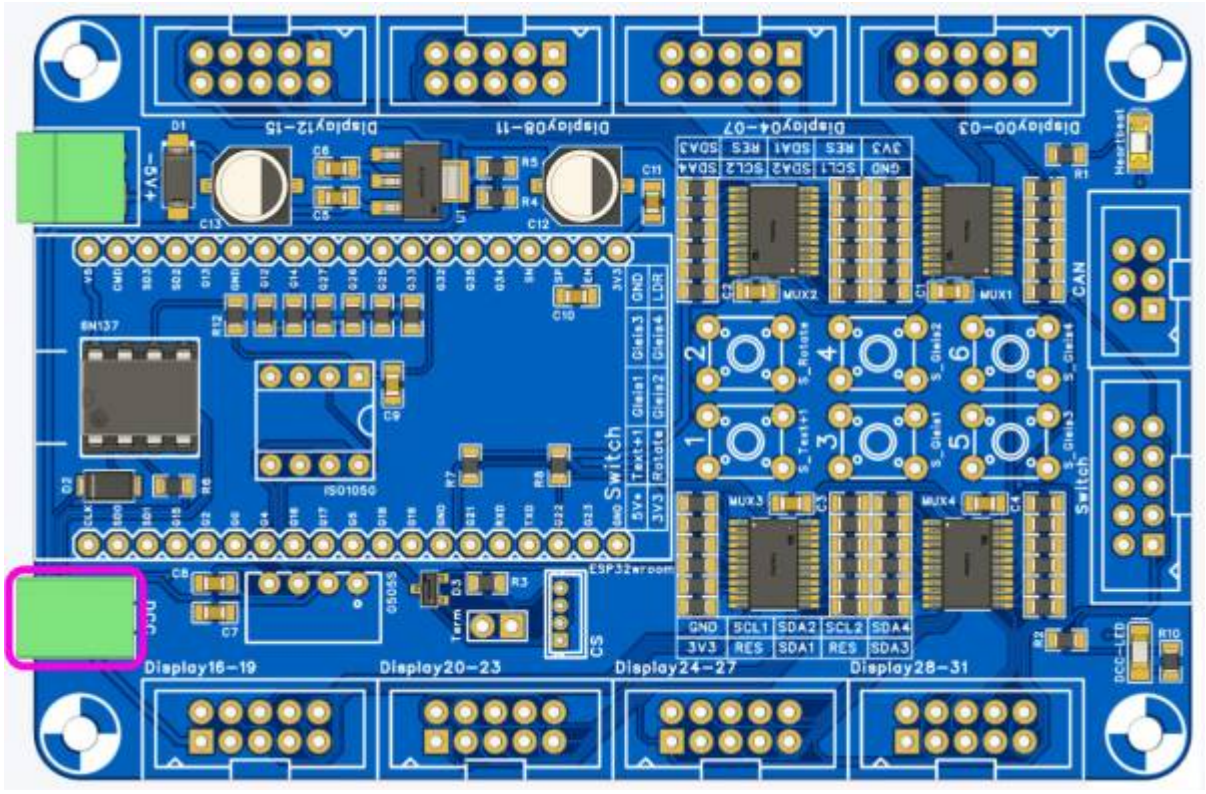

**Schritt 3**
Die Versorgungsspannung (5V) erfolgt über den grünen Stecker mit 3,5 mm Rastermaß.

**Achtung:** Die Polarität wurde an den MobaLedLib-Standard angepasst.
Bei der Vorgängerplatine war die Polarität vertauscht.



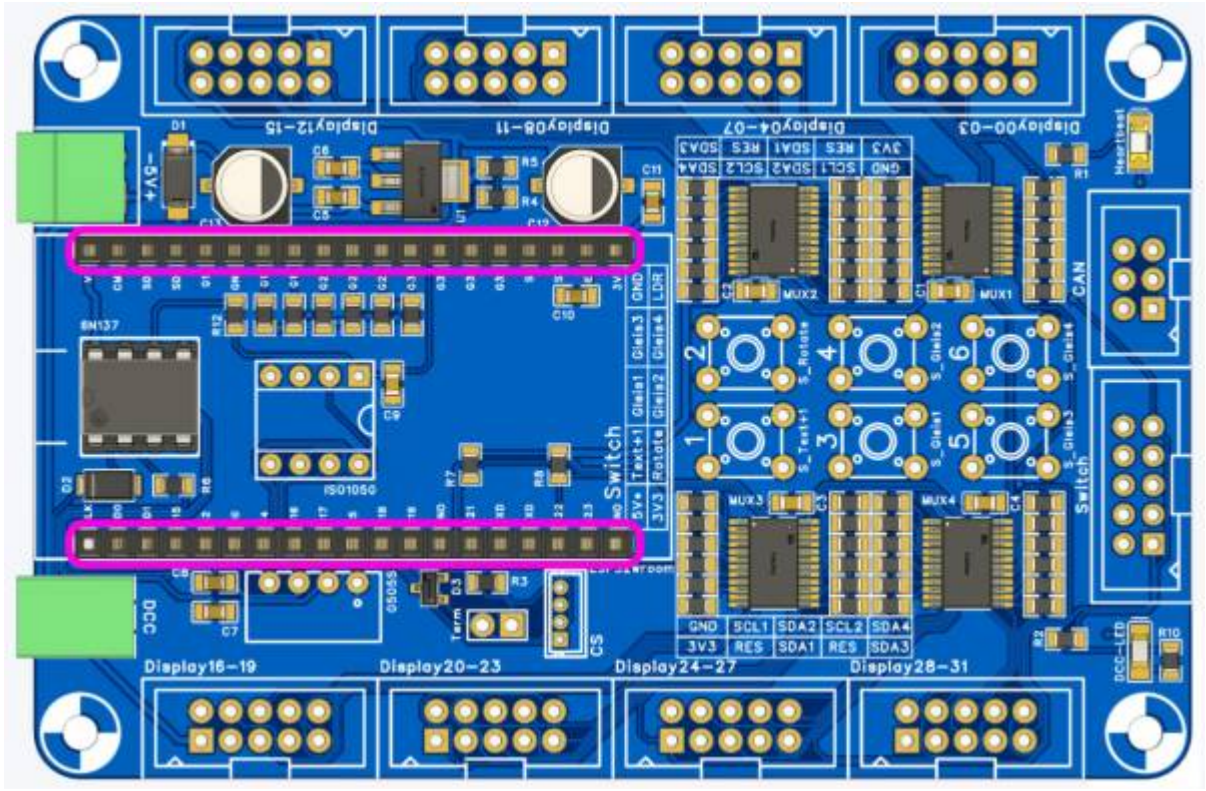

**Schritt 4**

Das DCC-Signal wird über den kleineren grünen Stecker im Rastermaß 2,54 mm eingespeist.



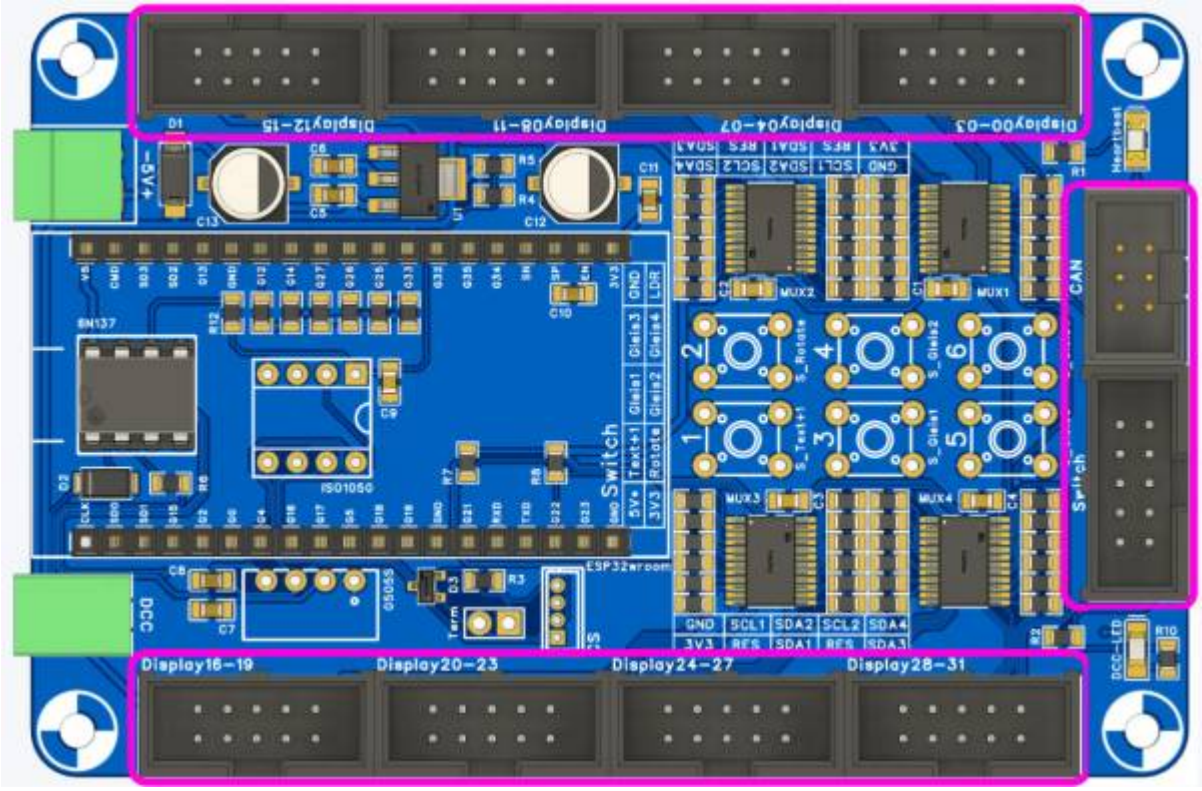

**Schritt 5**

Um die Buchsenleisten rechtwinklig und plan einzulöten, montiert man sie am besten mit eingestecktem ESP32.
Nachdem der jeweils vordere, mittlere und hintere PIN der Buchsenleisten gelötet ist, wird dieser entfernt und die restlichen Lötpunkte verzinnt.

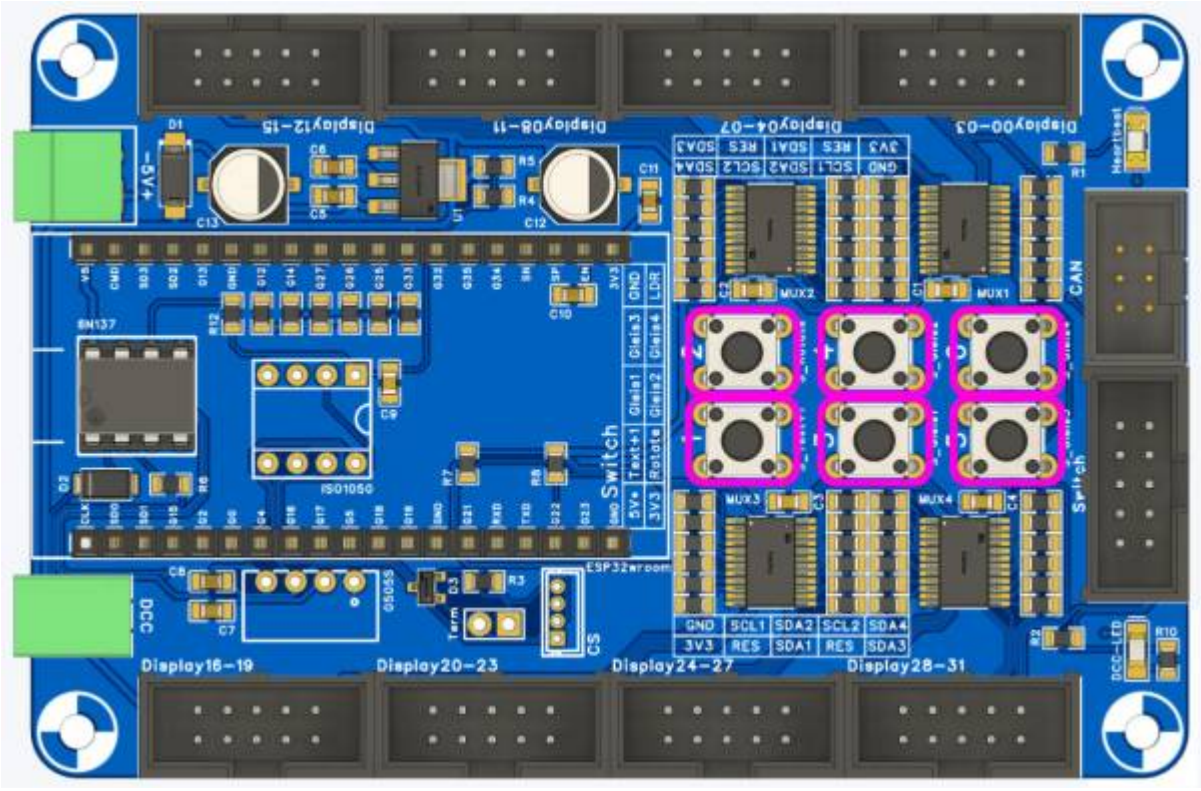



**Schritt 6**

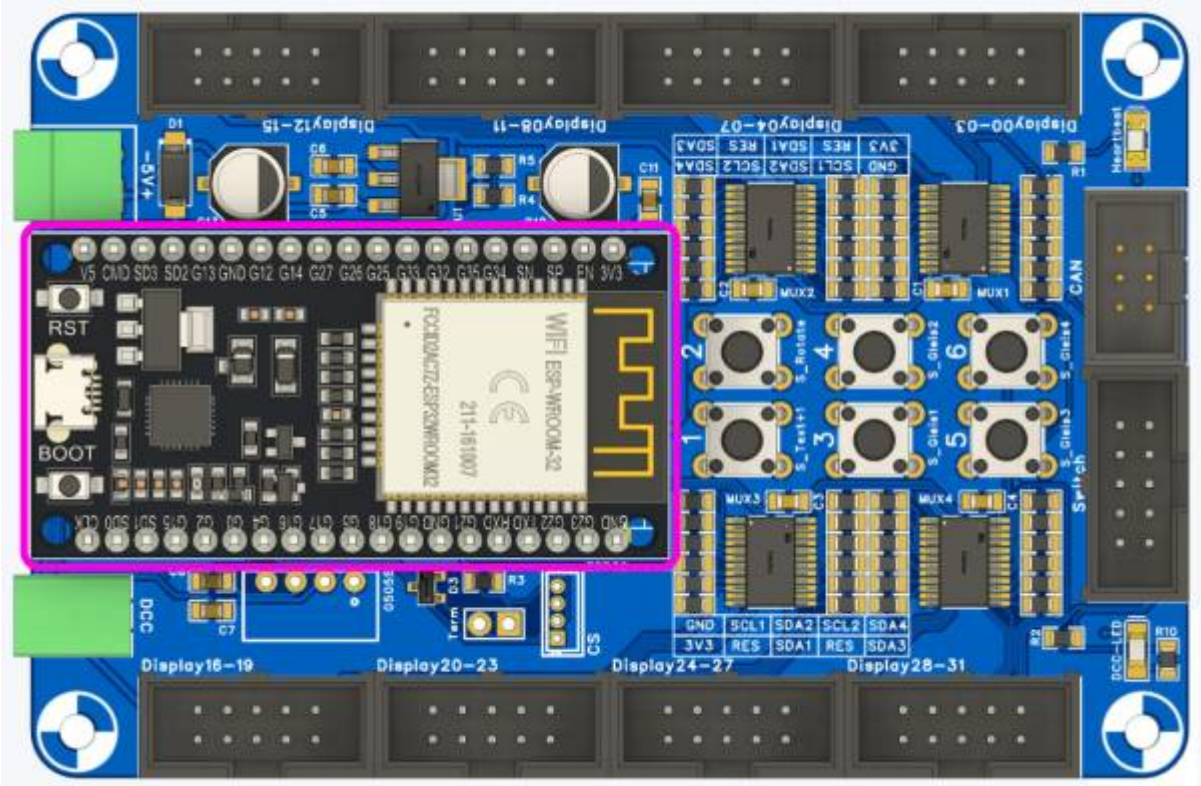
Bei den Wannensteckern empfiehlt es sich, zunächst nur jeweils einen Kontakt zu löten. So kann man nach einer Sichtkontrolle die einzelnen Stecker durch Erhitzen dieses Pins noch korrigieren.
Im Anschluss verzinnt man die restlichen Pins. Der 6-polige Wannenstecker ist zur Ansteuerung per CAN vorgesehen und derzeit noch ohne Funktion. Um das Gehäuse später nicht anpassen zu müssen, wird er hier vorsorglich eingelötet.




**Schritt 7**
Nach dem Einlöten der sechs Taster ist Vorsicht geboten. Die langen Hebel neigen abzubrechen. Ohne entsprechendes Gehäuse sollten hier Taster mit kürzerem Hub verwendet werden.

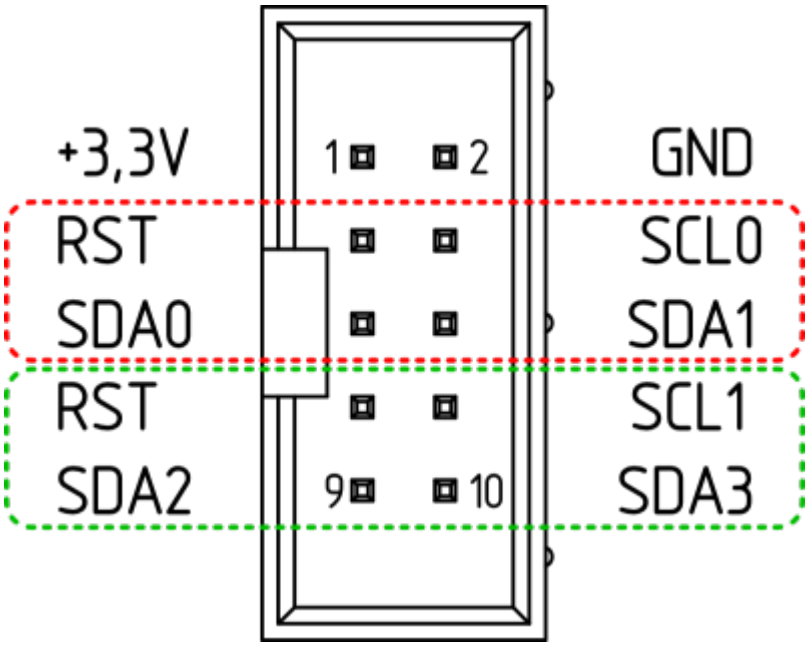


**Schritt 8**
Der letzte Schritt ist das Einstecken des Mikrocontrollers.




## Anschluss der Displays

### Pin-Belegung des Wannensteckers




Der Wannenstecker ist für den Anschluss eines Vierfach-Displays optimiert.
Hier teilen sich dann vier OLEDs auf einem OLED-Adapter (750) die Anschlüsse Plus, Minus, Rest und SCL.

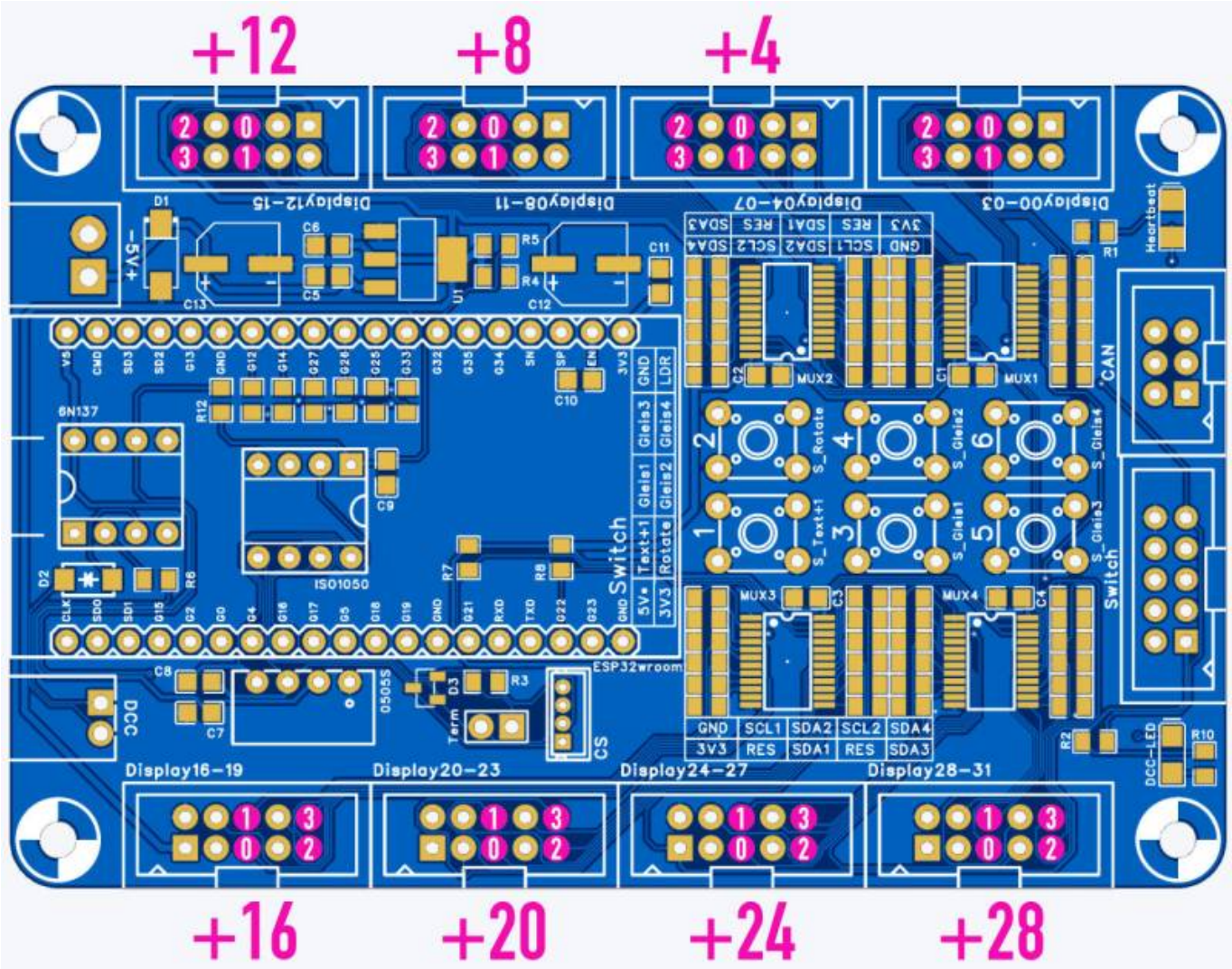
Die RST-Pins sind identisch. Es reicht, wenn einer davon zum OLED-Adapter (750) geführt wird.

Die SCL-Pins haben eigene Pull Up Widerstände und sollten separat zum OLED-Adapter (750) geführt werden.
SCL0 und SCL1 sind keinen bestimmten SDA-Pins zugeordnet. Die Reihenfolge spielt daher keine Rolle.

Bei Zweifach-Displays können die Leitungen SCL, 3V3 und GND mit Y-Verbindungen aufgeteilt werden.
So können je Wannenstecker zwei Zweifach-Displays, also vier OLEDs betrieben werden (rot/grün).

**Reihenfolge der Displays**

Dank der Gleisgruppen ist es bei der neuen Steuerung nicht entscheidend, an welchen SDA-Pin die Displays angeschlossen werden.
Es empfiehlt sich lediglich, alle Displays auf dieselbe Weise zu verkabeln, weil dann die Zuordnung im TextMessages Tool einfacher wird.
Das folgende Schema verdeutlicht die Vorgehensweise:
In jedem Wannenstecker gibt es für vier Displays vier SDA-Anschlüsse (0 bis 3). Mit jedem weiteren Wannenstecker erhöht sich die Nummer des Pins um vier.
Somit stehen 32 SDA-Anschlüsse zur Verfügung (0 bis 31).



## Warum diese Lösung und nicht MQTT?

MQTT gilt als moderne, serverbasierte Lösung, bei der das Steuerungsprogramm (z. B. RocRail) die Befehle an einen MQTT Broker sendet (typischerweise ein Raspberry Pi), auf dem die Verwaltung der Zugziele läuft.
Sobald der Broker einen Befehl von der Steuerungssoftware bekommt, schickt er die Daten zum ESP.

### Die Vorteile von MQTT

- Die Zugziele müssen nicht im Arduino Sketch gepflegt werden. Sie werden bequem über die Weboberfläche WYSIWYG eingepflegt.
- Deutsche Umlaute können direkt eingegeben werden.
- Es ist nur ein einziger Befehl der Steuerungssoftware an den MQTT Broker nötig, um das Zugziel an das richtige Gleis zu senden.

### Die Nachteile von MQTT

- Der Installationsaufwand stellt für den ungeübten Bastler mit wenig IT-Kenntnissen eine zusätzliche Hürde dar.
- Das Umschalten der Displays setzt zwingend eine Steuerungssoftware (z. B. RocRail) voraus oder einen PC in unmittelbarer Nähe, zum Schalten per Weboberfläche.

Als ich Ende 2023 mit diesem Projekt begann, war nur RocRail in der Lage, MQTT Nachrichten zu senden.
Der per DCC steuerbare Arduino Sketch war also schon damals universeller einsetzbar, wenn auch noch mit vielen Einschränkungen verbunden.
Doch mit der ZZA Steuerung v3 sind die meisten dieser Einschränkungen Geschichte.

### Die Vorteile des Sketchs

- Steuerbar per serieller Schnittstelle, DCC Handregler und Steuerungssoftware.
- Kein zusätzlicher Server nötig.
- Steuerbar mit iTrain (Aktionen), RocRail (ergänzen?), TrainController Gold (Bahnwärter) und WinDigipet (Stellwerkswärter)
- Bis zu 32 OLED Displays mit einer Steuerung möglich **(NEU ab v3)**
- Deutsche Umlaute können direkt eingegeben werden. **(NEU ab v3)**
- Nahezu unbegrenzte Anzahl der Zugziele **(NEU ab v3)**
- Bequeme Eingabe der Zugziele inkl. Suchfunktion, Lokverwaltung und Steuerbefehlen **(NEU ab v3)**
- Bequemer Upload des geänderten Sketchs per WLAN **(NEU ab v3)**
- Über 200 vordefinierte Zugziele als Beispielsammlung **(NEU ab v3)**

### Die Nachteile des Sketchs

- Es sind immer zwei direkt aufeinanderfolgende DCC-Befehle nötig, um das richtige Gleis zu wählen und den passenden Text zu senden. Das können nicht alle Steuerungen. ⇒ Link

## Das TextMessages Tool

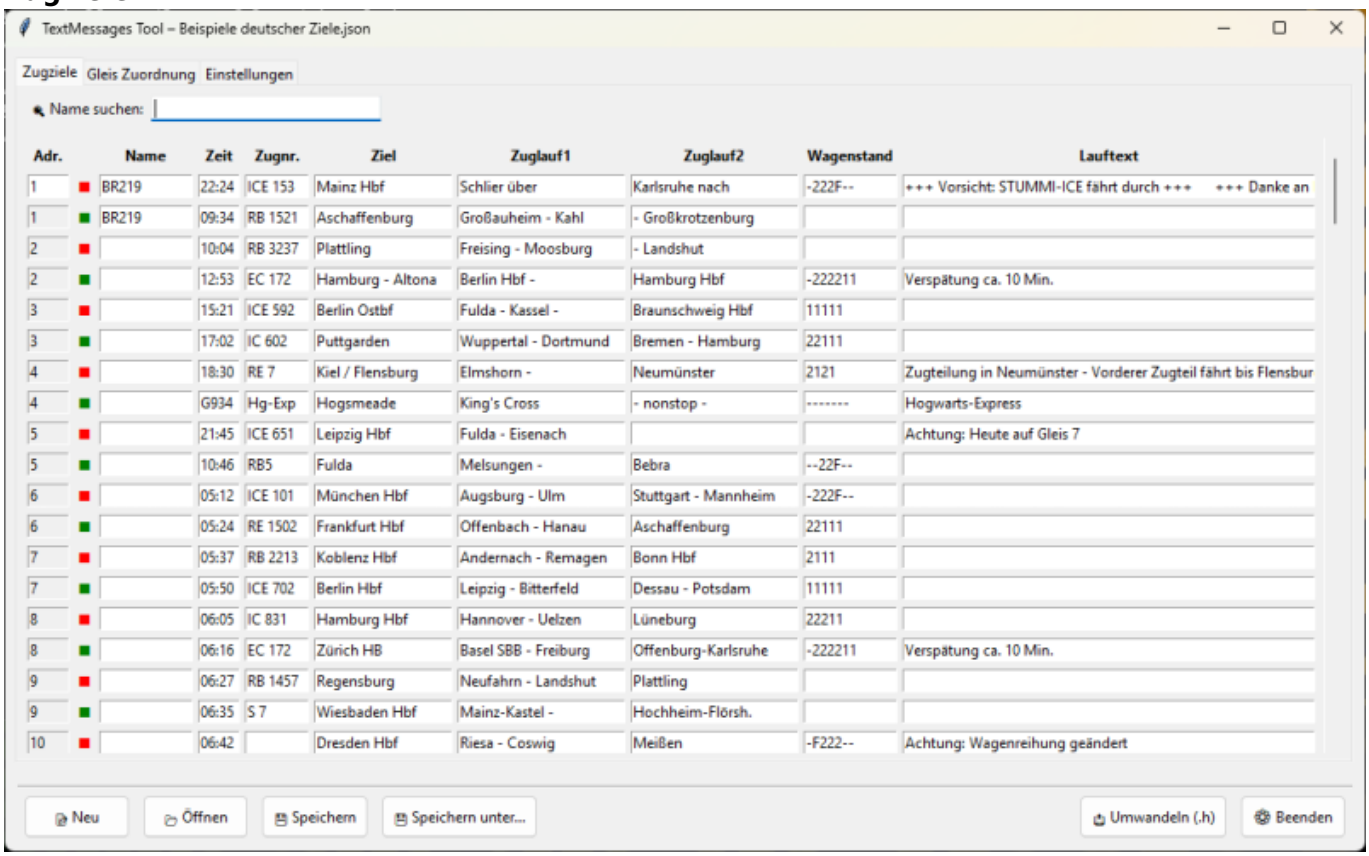
Das TextMessages Tool unterstützt bei der Eingabe der erforderlichen Daten. Dazu gehören in erster Linie die Zugziele, WLAN- und DCC-Einstellungen sowie die Zuordnung der Displays an die jeweiligen

Gleise. Nach der Eingabe in die grafische Oberfläche wandelt das Tool die Daten in die benötigte Arduino-Sprache um.

Mit dem Tool können mehrere JSON-Dateien angelegt, bearbeitet und in den Sketch umgewandelt werden. So lassen sich beispielsweise unterschiedliche Ziele je Bahnhof verwalten. Dabei kann der Sketch immer derselbe sein, bei dem lediglich die Text_Messages.h vor dem Upload ersetzt wird.

Um das TextMessages Tool nutzen zu können, ist - sofern durch die MobaLedLib noch nicht geschehen - die Installation von Python erforderlich.
⇒ https://www.python.org/downloads/

### Zugziele




| Adr. | Name | Zeit | Zugnr. | Ziel | Zuglauf1 | Zuglauf2 | Wagenstand | Lauftext |
|---|---|---|---|---|---|---|---|---|
| 1 | BR219 | 22:24 | ICE 153 | Mainz Hbf | Schlier über | Karlsruhe nach | -222F-- | +++ Vorsicht: STUMMI-ICE fährt durch +++ +++ Danke an |
| 1 | BR219 | 09:34 | RB 1521 | Aschaffenburg | Großauheim - Kahl | - Großkrotzenburg | | |
| 2 | | 10:04 | RB 3237 | Plattling | Freising - Moosburg | - Landshut | | |
| 2 | | 12:53 | EC 172 | Hamburg - Altona | Berlin Hbf - | Hamburg Hbf | -222211 | Verspätung ca. 10 Min. |
| 3 | | 15:21 | ICE 592 | Berlin Ostbf | Fulda - Kassel - | Braunschweig Hbf | 11111 | |
| 3 | | 17:02 | IC 602 | Puttgarden | Wuppertal - Dortmund | Bremen - Hamburg | 22111 | |
| 4 | | 18:30 | RE 7 | Kiel / Flensburg | Elmshorn - | Neumünster | 2121 | Zugteilung in Neumünster - Vorderer Zugteil fährt bis Flensbur |
| 4 | | G934 | Hg-Exp | Hogsmeade | King's Cross | - nonstop - | ------- | Hogwarts-Express |
| 5 | | 21:45 | ICE 651 | Leipzig Hbf | Fulda - Eisenach | | | Achtung: Heute auf Gleis 7 |
| 5 | | 10:46 | RB5 | Fulda | Melsungen - | Bebra | --22F-- | |
| 6 | | 05:12 | ICE 101 | München Hbf | Augsburg - Ulm | Stuttgart - Mannheim | -222F-- | |
| 6 | | 05:24 | RE 1502 | Frankfurt Hbf | Offenbach - Hanau | Aschaffenburg | 22111 | |
| 7 | | 05:37 | RB 2213 | Koblenz Hbf | Andernach - Remagen | Bonn Hbf | 2111 | |
| 7 | | 05:50 | ICE 702 | Berlin Hbf | Leipzig - Bitterfeld | Dessau - Potsdam | 11111 | |
| 8 | | 06:05 | IC 831 | Hamburg Hbf | Hannover - Uelzen | Lüneburg | 22211 | |
| 8 | | 06:16 | EC 172 | Zürich HB | Basel SBB - Freiburg | Offenburg-Karlsruhe | -222211 | Verspätung ca. 10 Min. |
| 9 | | 06:27 | RB 1457 | Regensburg | Neufahrn - Landshut | Plattling | | |
| 9 | | 06:35 | S 7 | Wiesbaden Hbf | Mainz-Kastel - | Hochheim-Flörsh. | | |
| 10 | | 06:42 | | Dresden Hbf | Riesa - Coswig | Meißen | -F222-- | Achtung: Wagenreihung geändert |

Im Reiter Zugziele werden alle gewünschten Ziele je Zuggarnitur definiert. Es empfiehlt sich, die Ziele an eine bestimmte Lokomotive zu binden, da das Steuerungsprogramm (z. B. iTrain) diese Lok abfragen kann. Um einen Überblick der zugeordneten Ziele zu erhalten, kann zu jedem Ziel der Name der jeweiligen Lok hinzugefügt werden. Nach diesem Namen kann man später auch suchen. Es empfiehlt sich, hier die Baureihe einzutragen.

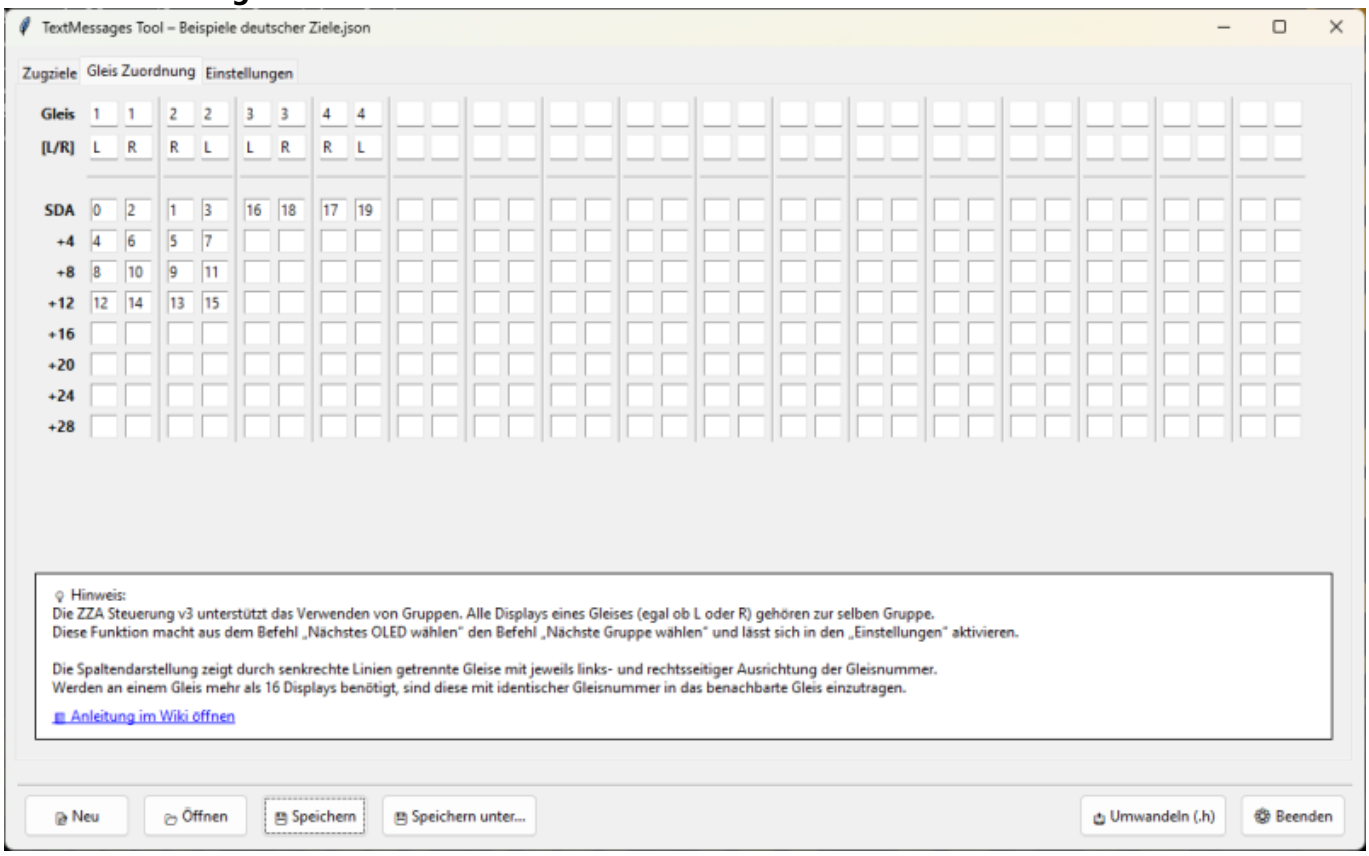
Die DCC Adressen werden vom Sketch fortlaufend vergeben. Daher bietet das Tool keine Möglichkeit zur Sortierung der Ziele (z. B. nach Baureihe), da die Neusortierung zu Abweichungen mit der Programmierung im Steuerungsprogramm führen würde.

Die hier einzutragende Adresse für das erste Zugziel richtet sich nach der DCC Startadresse (n), die im Reiter „Einstellungen“ eingetragen wird und der Anzahl der angeschlossenen Gleise. Sie dient als Orientierung und hat keinen Einfluss auf das Programm.

| Gleise | DCC erstes Ziel |
|---|---|
| bis zu 2 | n+4 |
| bis zu 4 | n+6 |

| Gleise | DCC erstes Ziel |
|---|---|
| bis zu 6 | n+8 |
| ... | |
| bis zu 32 | n+34 |

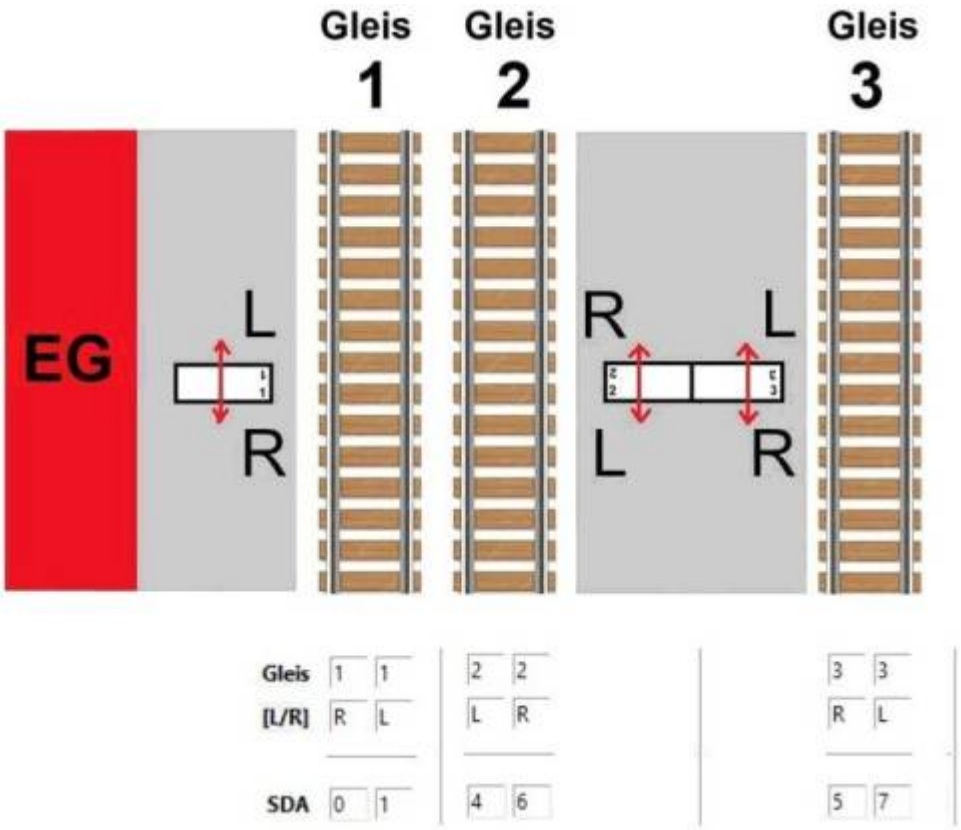
### Gleis Zuordnung



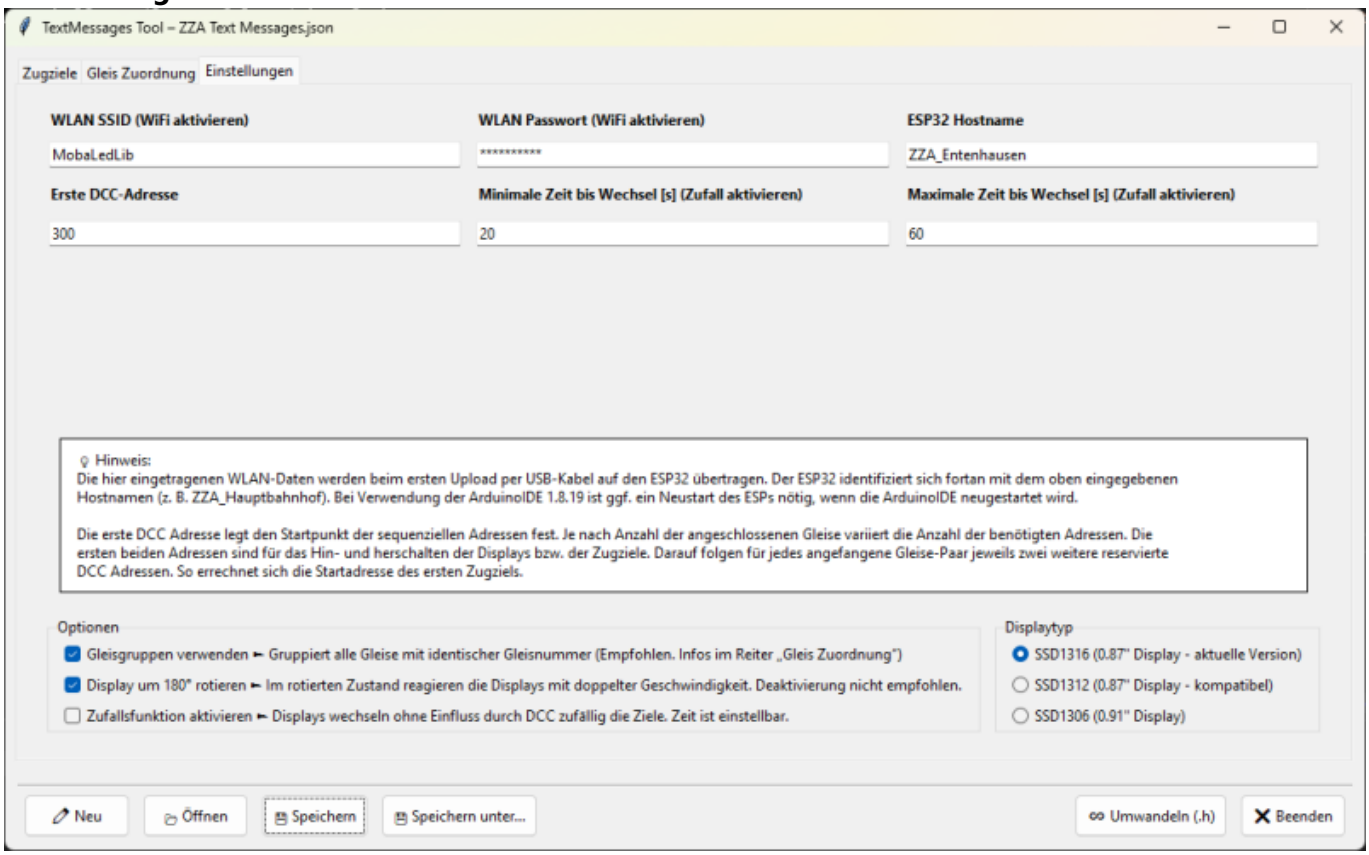

Der Reiter „Gleis Zuordnung“ ist so aufgebaut, dass er die Gleisgruppen selbst vorgibt. So ist keine Anpassung der Reihenfolge im Sketch mehr erforderlich. Jede Doppelspalte entspricht einem Gleis. Werden mehr als 16 Displays für ein Gleis benötigt, müssen diese in zwei benachbarten Doppelspalten eingetragen werden, damit sie zur selben Gleisgruppe gehören. Dieser Fall ist äußerst selten, weil für das zweite Gleis weniger als die Hälfte aller Displays übrig blieben. Mit der Ausrichtung [L/R] wird angegeben, ob die Gleisnummer links oder rechts im Display erscheinen soll.

Die SDA Anschlüsse werden entsprechend der vorgenommenen Verdrahtung angegeben (siehe „Reihenfolge der Displays“). Für ein vierfaches Doppeldisplay sind das in der Regel zwei Gleise mit jeweils zwei Ausrichtungen. Dieses vierfache Doppeldisplay belegt jeweils eine Reihe zweier Doppelspalten (z. B. SDA 0 und 2 für Gleis 1 L/R und SDA 3 und 1 für Gleis 2 L/R). Die Gleisseite lässt sich für zwei Doppelspalten auch in der Reihenfolge L/R/R/L eintragen, was von der Logik eher dem Aufbau der Adapterplatine (750) entspricht.

Zum besseren Verständnis hier ein Beispiel für drei Gleise, die am Empfangsgebäude des Bahnhofs liegen. Gleis 1 wird mit einem doppelseitigen Einzeldisplay bedient, Gleis 2 und 3 werden über ein doppelseitiges Zweifachdisplay bedient. Das Einzeldisplay nutzt nur zwei der vier Anschlüsse des ersten Wannensteckers, das Zweifachdisplay nutzt alle vier Anschlüsse des zweiten Wannensteckers.




## Einstellungen




Im Reiter Einstellungen werden als erstes die WLAN Zugangsdaten eingegeben, um den Sketch kabellos übertragen zu können. Die erste Übertragung muss selbstverständlich per USB Kabel erfolgen, da die Zugangsdaten noch nicht übermittelt wurden. Da die MobaLedLib nach wie vor die Arduino IDE 1.8.19 voraussetzt, ist es zwingend erforderlich, dass der ESP32 erst nach dem Öffnen der Arduino IDE gestartet wird. Es empfiehlt sich daher, die ZZA Steuerung über ein Relais schaltbar

zu machen, sofern man die Übertragung per WLAN nutzen möchte.

Mit der ersten DCC Adresse legt man den Startpunkt für die Steuerung fest. Diese muss selbstverständlich mit den Einstellungen im Steuerungsprogramm übereinstimmen.

Die Option „Zufall“ ist standardmäßig deaktiviert. Wer das zufällige Umschalten der DCC Steuerung vorzieht, kann das hier aktivieren und auch die minimale und maximale Zeit bis zum nächsten Umschalten einstellen. Die DCC Steuerung **kann** in dem Fall deaktiviert werden.

Die Gleisgruppen sind standardmäßig aktiviert. Ohne diese müsste jedes OLED Display einzeln aktiviert werden, um einen Text zu übertragen. Für jedes doppelseitige Display wären somit vier aufeinanderfolgende DCC Befehle erforderlich. Mit aktiven Gleisgruppen werden alle OLEDs eines Gleises gleichzeitig aktualisiert.

Die Display-Rotation ist standardmäßig aktiviert, weil es zum einen der Ausrichtung auf der Adapterplatine (750) entspricht und zum anderen die schnellste Aktualisierung der Displays gewährleistet. Die Option ist für den Notfall gedacht, wenn das Display versehentlich über Kopf eingebaut wurde. Es wird daher empfohlen, die Ausrichtung der Displays vor dem festen Einbau zu prüfen.

**Umwandeln**
Sind alle Einstellungen korrekt eingetragen, wandelt man mit dem Button „Umwandeln (.h)“ die Werte in die benötigte „Text_Messages.h“-Datei um und ersetzt diese im Verzeichnis „users/user/documents/Arduino/Zugzielanzeiger“. Die „Text_Messages.h“ kann übrigens ohne Bedenken ersetzt werden, da das Programm automatisch ein Backup der letzten 20 Konfigurationen erstellt und die ältesten automatisch löscht. Ist die Arduino IDE in einem der beiden Standard-Verzeichnisse installiert, öffnet das Programm nach erfolgreicher Umwandlung sogar den Sketch in der Arduino IDE. Nach einem kurzen Neustart des ESP32 kann der veränderte Sketch nun per WLAN übertragen werden.

# Download und Installation des ESP Sketches

Der Sketch ist hier zu finden:
https://github.com/raily74/MobaLedLib/blob/main/OLED/Zugzielanzeiger/Sketch/Zugzielanzeiger.zip

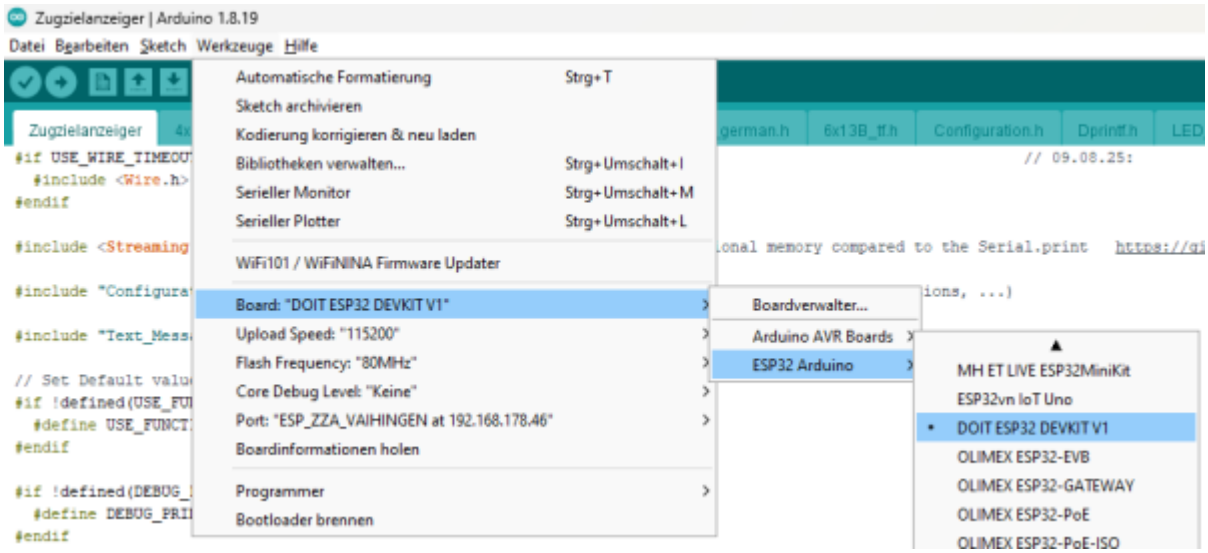
## Erste Übertragung auf den ESP32

Es empfiehlt sich, vor dem ersten Übertragen **mindestens** die WLAN-SSID, das WLAN Passwort und den gewünschten ESP-Hostname im Text Messages Tool einzugeben und den Sketch mit diesen Daten **per USB-Kabel** auf den ESP32 zu übertragen. Nach dem Umwandeln der Eingaben in die „TextMessages.h“ öffnet sich auf Wunsch die Arduino IDE. Vor dem ersten Upload auf den ESP32 müssen wie gewohnt über „Werkzeuge > Bibliotheken verwalten...“ folgende Bibliotheken installiert werden:

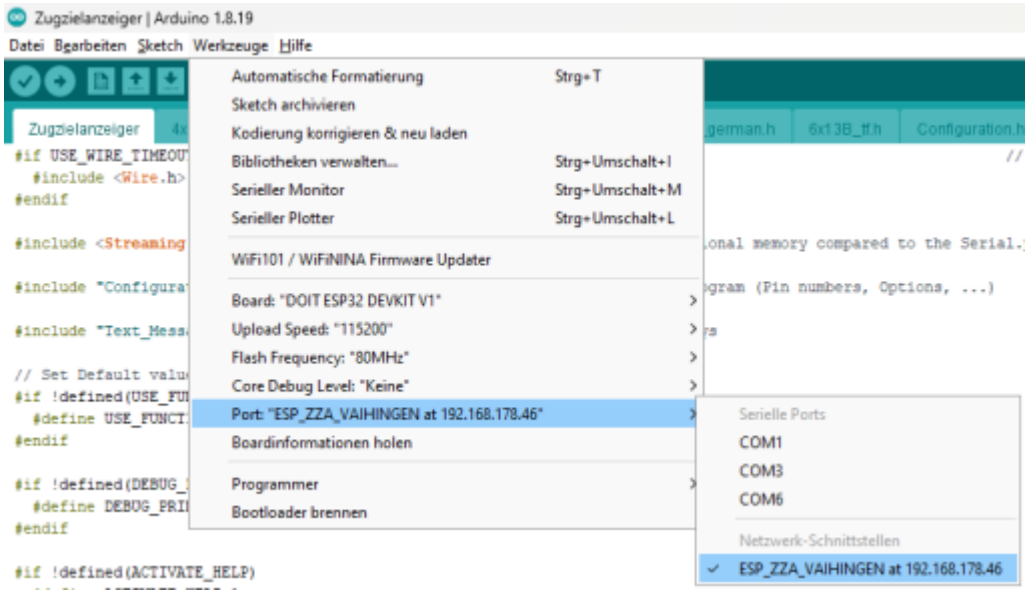
- SparkFun I2C Mux Arduino Library
- Streaming
- U8g2
- WiFi

- Arduino_ESP32_OTA
- NmraDcc (wird bei Installation der MobaLedLib bereits installiert)

Nachdem die Bibliotheken installiert wurden, muss das richtige Board ausgewählt werden, damit die Arduino IDE den Sketch passend für den ESP32 kompilieren kann.
Das geht über Werkzeuge > Board > ESP32 Arduino > DOIT ESP32 DEVKIT V1




Der erste Upload muss per USB-Kabel erfolgen, weil der ESP32 die WLAN-Daten noch nicht kennt und sich somit noch nicht im WLAN melden kann. Dazu muss unter „Werkzeuge > Port“ der richtige serielle COM-Port eingestellt werden. Ab dem folgenden Upload kann hier dann auch unter Port auch die Netzwerk-Schnittstelle ausgewählt werden. Hier sieht man den ESP32 dann mit dem zuvor angegebenen ESP-Hostname.




# Steuerung per DCC

Der größte Clou der Zugzielanzeiger ist die Möglichkeit, den Anzeigentext vom einfahrenden Zug steuern zu lassen. Die Anzahl der benötigten DCC Adressen hängt von der Anzahl der Gleise und der Anzahl der Zugziele ab. Beides zählt der Sketch selbst und erzeugt damit die fortlaufenden Adressen.

**Achtung**
Zur automatischen Steuerung sind die Befehle „Gleisgruppe x wählen“ und „Textblock x auf aktueller Gleisgruppe anzeigen“ wichtig. Die folgende Tabelle zeigt ein Beispiel mit vier Gleisgruppen. Mit nur zwei statt vier Gleisgruppen würden die Zeilen 6/7 und 10/11 entfallen und der erste direkte Textblock stünde in Zeile 8 statt in 12. Mit fünf Gleisgruppen stünde der erste direkte Textblock in Zeile 16. Die Zeilen 9 und 15 wären leer, weil die sechste Gleisgruppe nicht belegt ist und bei zwei Adressen die Zustände „grün“ nicht verwendet würden. Das ist zu bedenken, wenn man im TextMessages Tool die erste DCC Adresse vergibt und die Startadresse für die direkten Textblöcke im Reiter „Zugziele“ festlegt.

| Aspekt* | DCC-Adr. | Zustand | Funktion | Bemerkungen |
|---|---|---|---|---|
| 0 | n+0 | Rot | Vorangegangenen Textblock auf aktueller Gleisgruppe anzeigen | wird zur Automatiksteuerung nicht benötigt |
| 1 | n+0 | Grün | Nächsten Textblock auf aktueller Gleisgruppe anzeigen | wird zur Automatiksteuerung nicht benötigt |
| 2 | n+1 | Rot | Vorangegangene Gleisgruppe wählen | wird zur Automatiksteuerung nicht benötigt |
| 3 | n+1 | Grün | Nächste Gleisgruppe wählen | wird zur Automatiksteuerung nicht benötigt |
| 4 | n+2 | Rot | Nächsten Textblock auf Gleisgruppe 1 anzeigen | Wichtige Funktion nach Verlassen von Gleis 1 |
| 5 | n+2 | Grün | Nächsten Textblock auf Gleisgruppe 2 anzeigen | Wichtige Funktion nach Verlassen von Gleis 2 |
| 6 | n+3 | Rot | Nächsten Textblock auf Gleisgruppe 3 anzeigen | Wichtige Funktion nach Verlassen von Gleis 3 |
| 7 | n+3 | Grün | Nächsten Textblock auf Gleisgruppe 4 anzeigen | Wichtige Funktion nach Verlassen von Gleis 4 |
| 8 | n+4 | Rot | Gleisgruppe 1 wählen | Wird als erstes von einfahrendem Zug auf Gleis 1 ausgelöst |
| 9 | n+4 | Grün | Gleisgruppe 2 wählen | Wird als erstes von einfahrendem Zug auf Gleis 2 ausgelöst |
| 10 | n+5 | Rot | Gleisgruppe 3 wählen | Wird als erstes von einfahrendem Zug auf Gleis 3 ausgelöst |
| 11 | n+5 | Grün | Gleisgruppe 4 wählen | Wird als erstes von einfahrendem Zug auf Gleis 4 ausgelöst |
| 12 | n+6 | Rot | Textblock 1 auf aktueller Gleisgruppe anzeigen | Wird anschließend mit Verzögerung (<1s) an das aktuelle Display gesendet |
| 13 | n+6 | Grün | Textblock 2 auf aktueller Gleisgruppe anzeigen | |
| 14 | n+7 | Rot | Textblock 3 auf aktueller Gleisgruppe anzeigen | |
| 15 | n+7 | Grün | Textblock 4 auf aktueller Gleisgruppe anzeigen | |
| … | | | | |
| 110 | n+55 | Rot | Textblock 99 auf aktueller Gleisgruppe anzeigen | |
| 111 | n+55 | Grün | Textblock 100 auf aktueller Gleisgruppe anzeigen | |

Mit der oben gezeigten Tabelle ist das Erstellen einer Regel ganz einfach. Die Verknüpfung wird hier

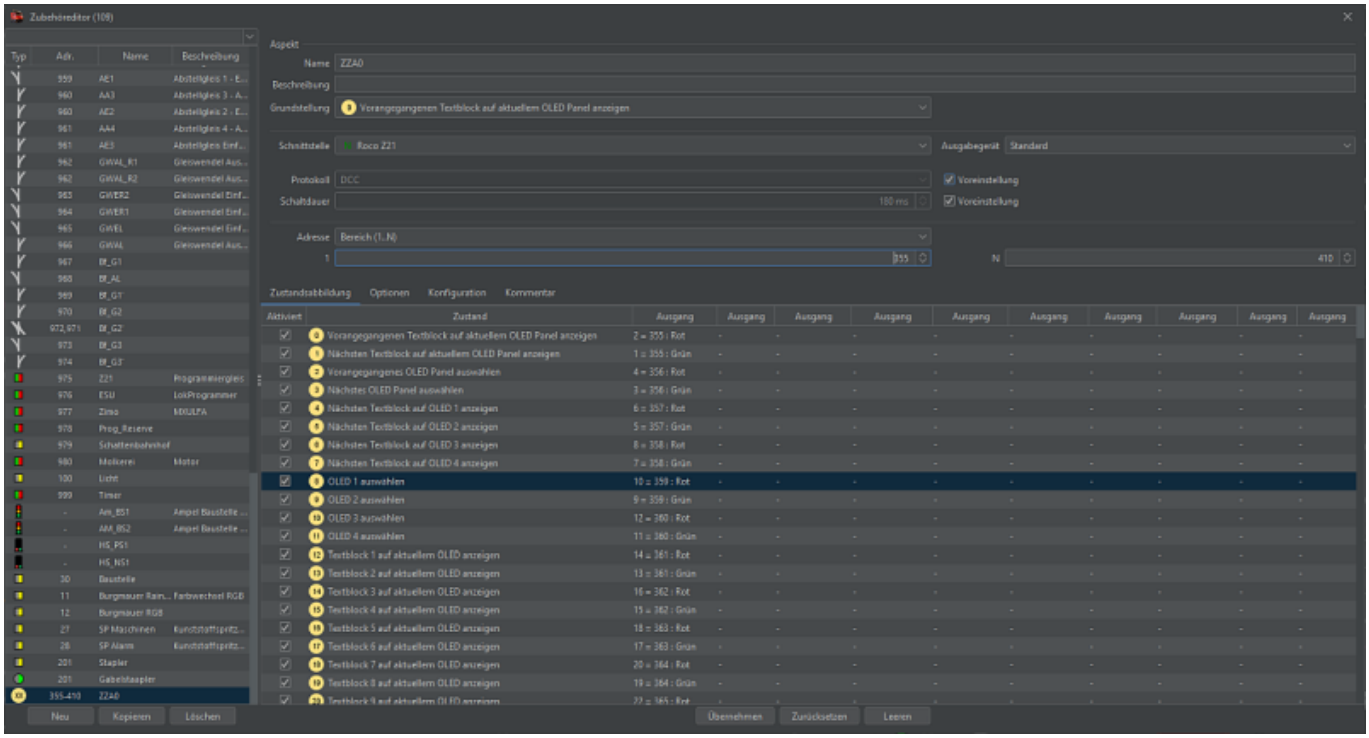
am Beispiel von iTrain gezeigt. Zur Erstellung einer „Wenn/Dann-Regel“ in anderen Programmen muss deren Anleitung zu Rate gezogen werden.

# Verwendung im Steuerungsprogramm

## iTrain

### Neuen Aspekt* als Zubehör in iTrain definieren

- Im Zubehöreditor von iTrain (Strg+F8) wird ein neues Zubehör vom Typ „Aspekt“ erstellt. Diesem gibt man einen frei wählbaren, sinnvollen Namen (z.B. „ZZA Hauptbahnhof“ oder wie im Bild „ZZA0“).
- Als Adresse wählt man „Bereich (1…N)“ und trägt bei 1 die zuvor in der Arduino IDE definierte Adresse „355“ und bei N die Adresse 410 ein. Es werden automatisch 56 DCC Adressen reserviert.
- Als nächstes müssen alle 112 Aspekte aktiviert werden. Ich kenne keinen Weg, wie man alle auf einmal aktivieren kann. Am schnellsten geht es daher, mit den Pfeiltasten zur zweiten Zelle in der Spalte „Aktiviert“ zu navigieren und die Leertaste zu drücken. Dann kann man im Wechsel den Pfeil nach unten und die Leertaste drücken, bis alle 112 Aspekte aktiviert sind.
- Hat man bis hierher alles richtig gemacht, sollte der Zustand „Aspekt A0“ auf Ausgang „1 = 355 : Grün“ liegen und „Aspekt A1“ auf Ausgang „2 = 355 : Rot“. Wir benötigen es aber genau umgekehrt. „Aspekt A0“ muss auf Ausgang „2 = 355 : Rot“ und „Aspekt A1“ auf Ausgang „1 = 355 : Grün“. Das Vertauschen ist einfach. Man wählt zunächst den „Aspekt A0“ per Mausklick aus, dann bei gedrückter Umschalt-Taste (Shift) den letzten Aspekt („Aspekt A111“) und drückt die Taste „S“. Alternativ kann man über die rechte Maustaste das Kontextmenü aufrufen und „Vertausche die Ausgänge (S)“ wählen.
- Wer jetzt noch die Muse hat, kann die vorgegebenen Bezeichnungen „Aspekt A0“ bis „Aspekt A111“ in der Spalte „Zustand“ per Doppelklick umbenennen. Hier empfiehlt es sich, die Namen der Funktion zu verwenden. Diese befinden sich im MobaLedLib-Wiki. Das Umbenennen muss nur einmal gemacht werden. Für eine zweite Display-Steuerung mit anderem Adress-Bereich kann das Zubehör „ZZA0“ kopiert werden. Im Anschluss kann einfach ein anderer Adressbereich für das Duplikat definiert werden.

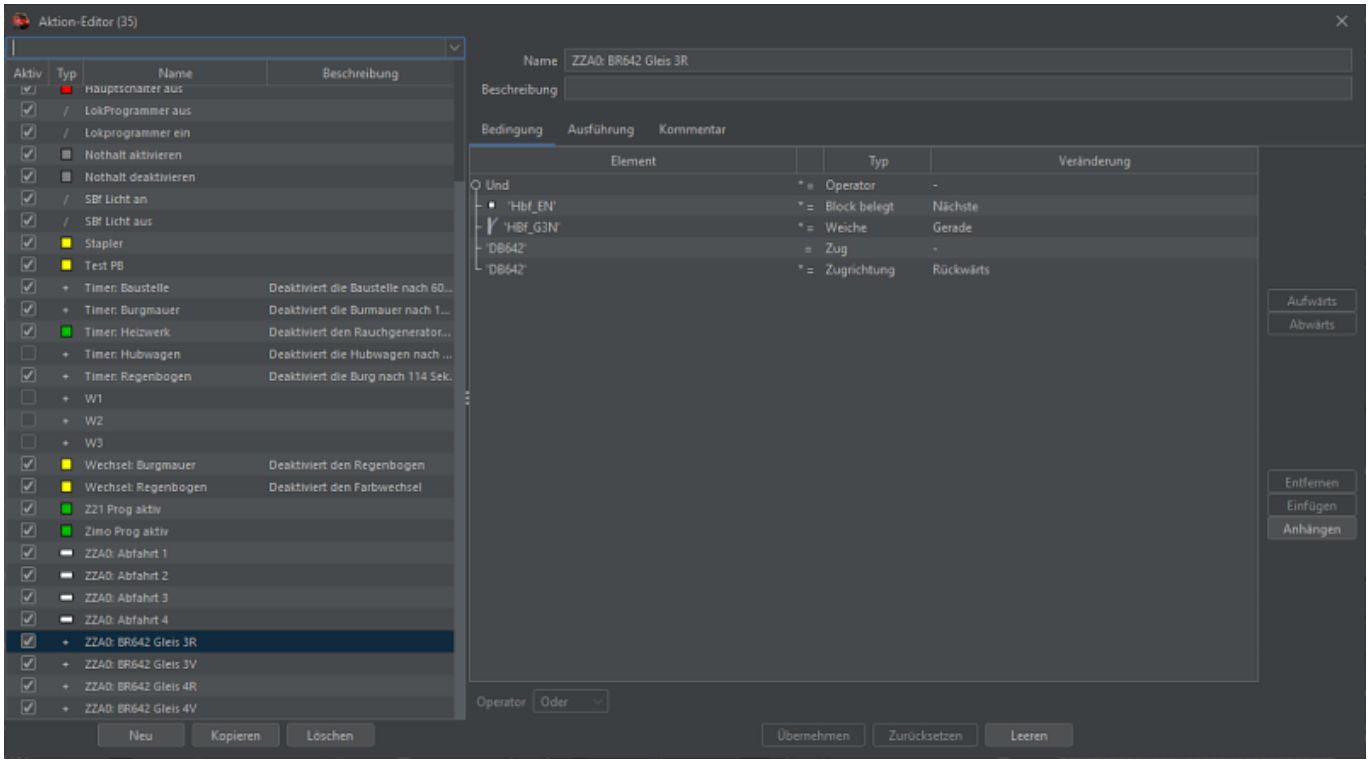


**Aktion mit Bedingungen erstellen**
In einer Aktion werden nun die Bedingungen und die Ausführung miteinander verknüpft.

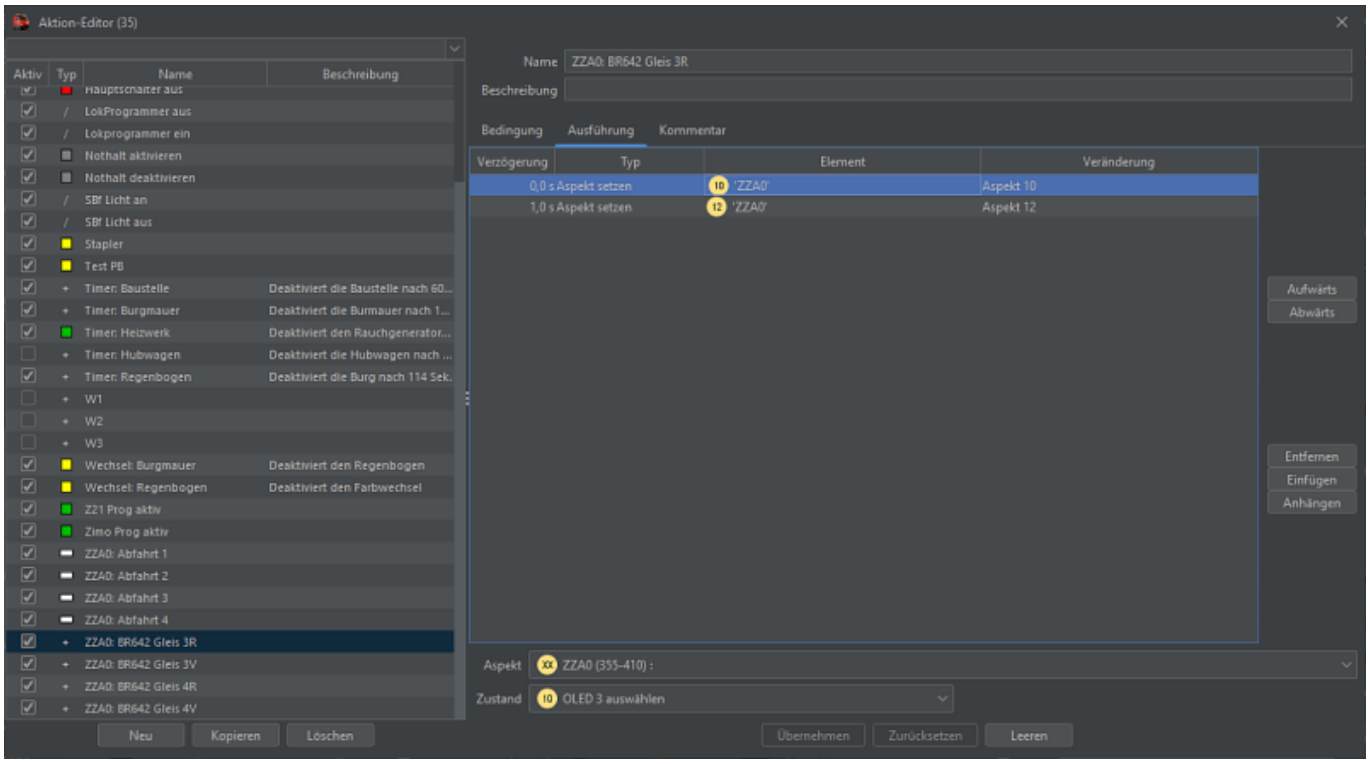
Beispiele für die Bedingung:
- ein bestimmter Block ist belegt
- im Block befindet sich ein bestimmter Zug (DCC-Adresse)
- die Fahrtrichtung ist vorwärts/rückwärts
- die folgenden Weichen führen zu einen bestimmten Gleis

Alle oben genannten Bedingungen sollten in einer „Und-Bedingung“ erfasst werden, sodass nur ausgeführt wird, wenn alle Anfragen mit „Ja“ beantwortet werden.

Die Ausführung schaltet lediglich zwei DCC Adressen. Auch hier ein Beispiel: An Gleis 3 soll der Text Nr. 1 angezeigt werden. Dazu wird zunächst die DCC Adresse n+5 auf Rot gesetzt (entspricht Aspekt 10) und mit einer kurzen Verzögerung die Adresse n+7 auf Rot (entspricht Aspekt 12). Im folgenden Beispiel wird für den Zug „DB642“ die Bedingung definiert, dass der Zug rückwärts fahrend den Block „Hbf_EN“ in Richtung Bahnhof belegt und die Weiche „Hbf_G3N“ auf gerade steht.




Sind alle vier Bedingungen erfüllt, wird der Aspekt „ZZA0“ auf den Zustand „10 (OLED3 auswählen)“ und mit einer Sekunde Verzögerung auf den Zustand „12 (Textblock 1 auf aktuellem OLED anzeigen)“ eingestellt. Das sorgt im Hintergrund dafür, dass die angeschlossene Zentrale die DCC-Adresse 360 auf Rot setzt und eine Sekunde später die DCC-Adresse 361 auf Rot. Der Arduino interpretiert diese beiden Signale, schaltet auf Display 3 um und zeigt dort Text 1 an. Für den vorwärts fahrenden „DB642“ wählt man beispielsweise Zustand „13 (Textblock 2 auf aktuellem OLED anzeigen)“. Auf Gleis 4 wählt man Zustand 14 und 15. So fährt derselbe Zug nicht immer zum selben Ziel.




*) Der Aspekt ist eine spezielle Funktion in iTrain. Dazu wird ein beliebiger DCC Adressbereich mit maximal 128 Adressen in einen virtuellen „Drehschalter“ mit

bis zu 256 Schaltstellungen verwandelt. Das macht das Senden der Display- und Textwahl einfacher. Die Programme WinDigipet, TrainController und RocRail werden ähnliche Möglichkeiten bieten, wenn auch unter anderem Namen. Zur Not kann auch einfach die jeweilige DCC-Adresse als Ausführung gesendet werden.
Hinweise zur Vorgehensweise dieser Programme bitte gern im Forum posten.

### WinDigipet

Ich bitte um eure Unterstützung

### Train Controller

Ich bitte um eure Unterstützung

### RocRail

Ich bitte um eure Unterstützung

---

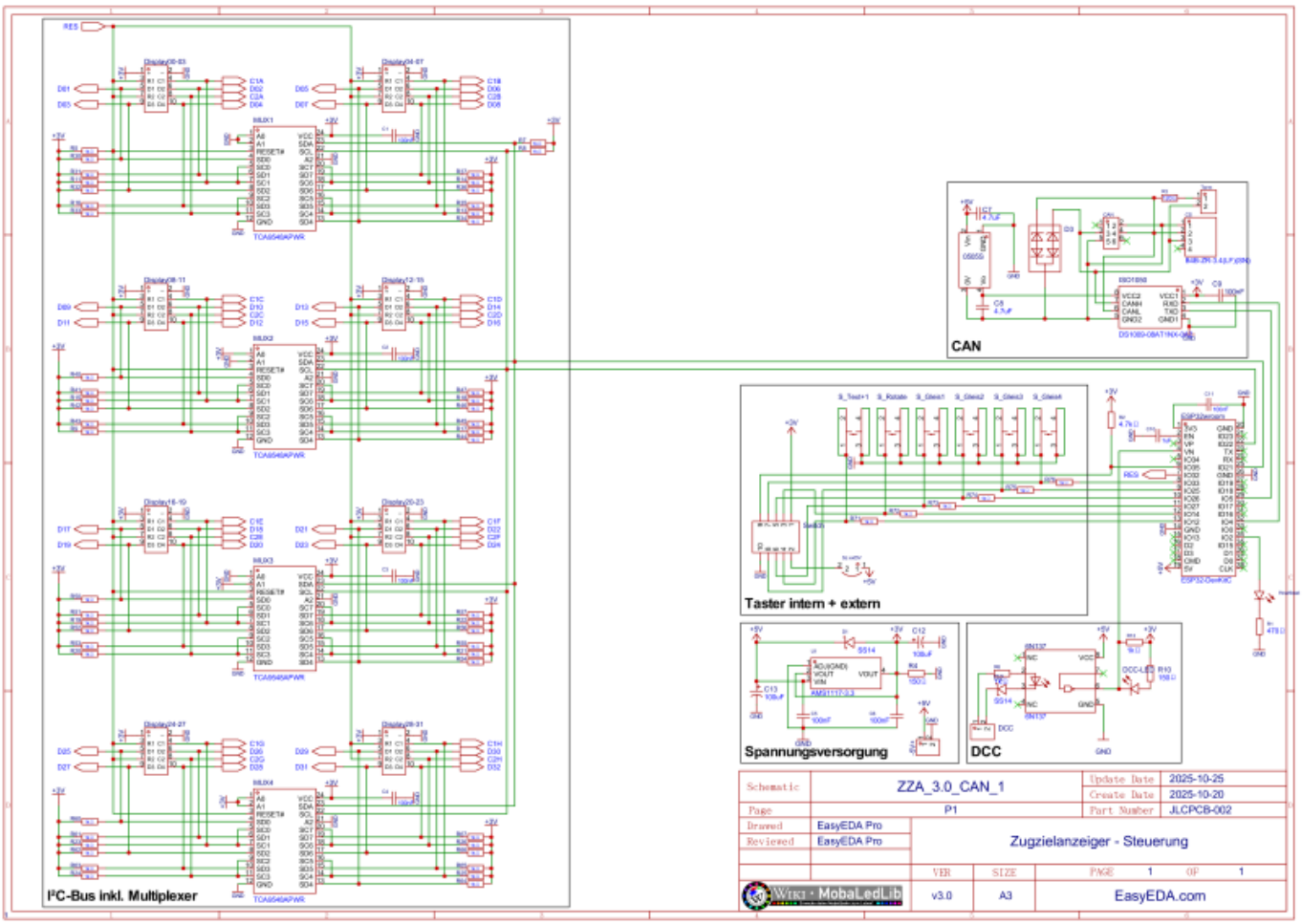
## 3D-Gehäuse - ZZA-Steuerung

Eignung für 3D-Drucker: FFF / FDM ★★★★ SLA / STL ★★★☆




Die Druckdaten sind hier zu finden:
https://github.com/Hardi-St/MobaLedLib_Docu/tree/master/3D_Daten_fuer_die_MobaLedLib/Gehaeuse-740_Display-Steuerung_Zugzielanzeiger

## Schaltplan




From:
https://wiki.mobaledlib.de/ - **MobaLedLib Wiki**

Permanent link:
**https://wiki.mobaledlib.de/anleitungen/oled/display-steuerung**

Last update: **2026/03/01 13:10**