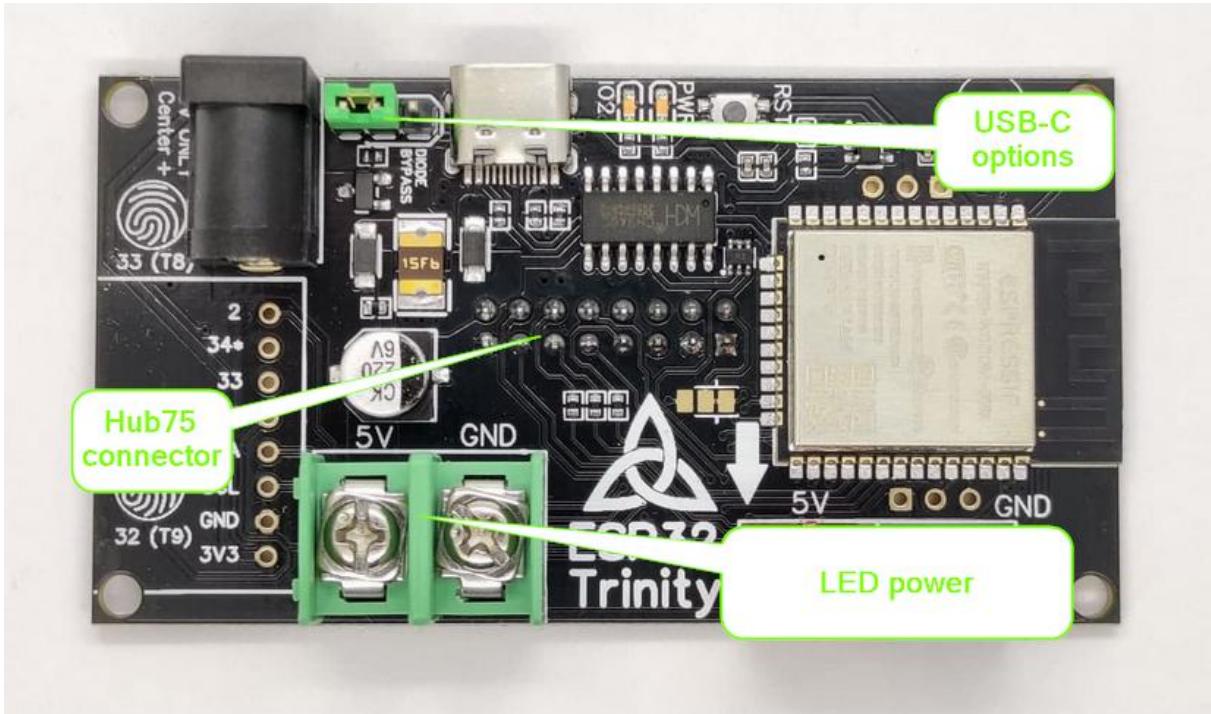


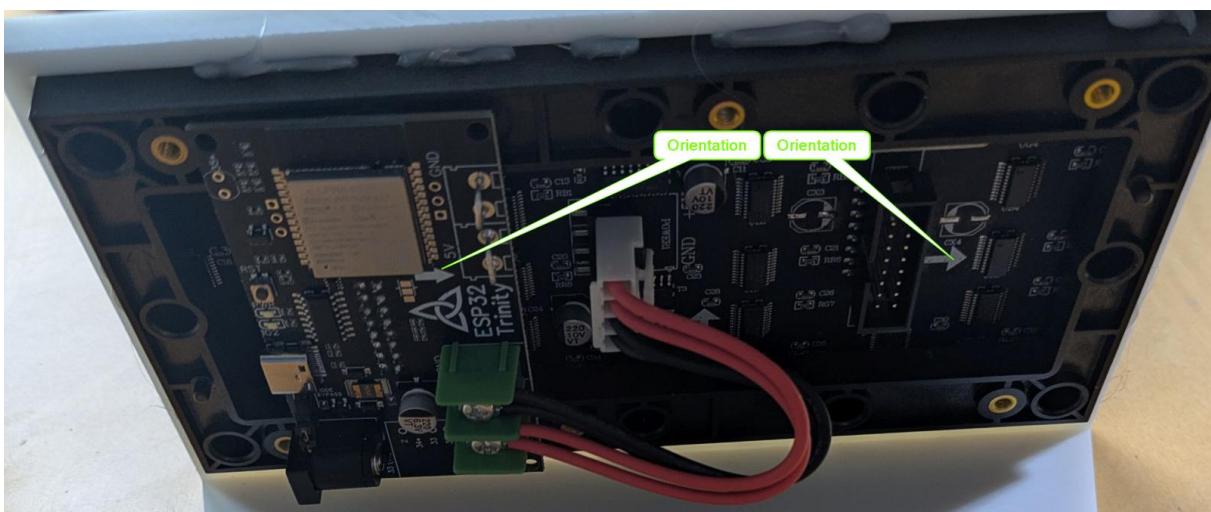
Kurzanleitung ESP32 Trinity WiFi Tetris Uhr

Basierend auf dem Open Source Projekt <https://esp32trinity.com/>

Zusammenbau



- 1) Die HUB75 Buchse auf der Rückseite der Platine anlöten.
- 2) Den 3Pin Anschluss (USB-C options) oben links anlöten.



- 3) Die Platine auf das LED Modul setzen. Dabei beachten dass die Pfeile auf Platine und Modul die gleiche Ausrichtung haben. **Beim Einsetzen sicherstellen dass die Buchse mittig im HUB75 Konnektor landet!**
- 4) Den ,LED Power' Anschluss mit dem LED Modul verbinden.
- 5) Das Led Modul in den Rahmen kleben.

Hardware config

Über die Position des Jumpers (USB-C Options) kann man festlegen ob das LED Modul auch über den USB-C Anschluss mit Spannung versorgt wird. Mit Jumper links wird nur der ESP32 versorgt; Jumper rechts (Diode Bypass) legt fest dass auch das LED Modul versorgt wird.

Alternativ kann man das LED Modul über die Rundbuchse mit 5Volt versorgen.

Bei meinen Tests zog das Modul maximal 300mA, daher sollte die Versorgung über USB-C kein Problem darstellen.

Inbetriebnahme

Beim ersten Start (config Modus) stellt die Uhr ein WiFi Netzwerk mit der Kennung ‚WiFiTetris‘ zur Verfügung. (Kennwort: ‚clock123‘). Über dieses kann man sein eigenes WiFi Netzwerk eintragen und einige Parameter der Uhr konfigurieren (Zeitzone, Geschwindigkeit der Animation, ..)

Nach erfolgter Konfiguration speichert die Uhr die Netzwerkkennung und verbindet sich automatisch nach dem Start.

Über den Reset Knopf und erneutetem Reset in der Anfangsphase kommt man wieder in den config Modus.

In WiFiTetris anmelden
192.168.4.1

WiFiManager

WifiTetris

Configure WiFi

Info

Exit

Update

No AP set

SSID

Password

Show Password

Time Zone

Europe/Dublin

24 hour Clock

Force every digit reset

Animation Delay (smaller == faster)

50

Set clkphase false

Set Driver to FM6126A

64x64 panel

Save

Programmierung

Mittels der Sourcen auf dem ESP32 Trinity Github und der Arduino IDE mit ‚ESP32 Extension‘ kann die Platine über den USB-C Anschluss programmiert werden. Dabei am besten das Board ‚Wemos D1 Min ESP32‘ auswählen.

Software updates ohne Arduino IDE

Auf meinem Server sind sowohl die Sourcen als auch die binaries für die Uhr abgelegt.

Mit Chrome, Edge oder dem Opera Browser (Firefox funktioniert nicht!) kann man über folgende Webseite: <https://espressif.github.io/esptool-js/> den ESP32 der Uhr programmieren.

Nach dem Verbinden des USB Anschlusses mit dem Computer und auswählen des richtigen COM Ports Adresse ,0x0‘ und eins der binaries vom Server auswählen. Dann ,Programm‘ klicken.

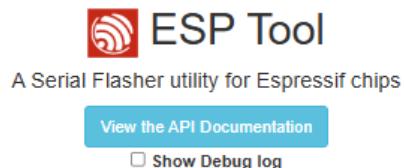
Warten bis

Hash of data verified.

Leaving...

Hard resetting via RTS pin...

erscheint. Dann Uhr vom USB trennen. Beim nachfolgenden Boot kann es sein dass der ,RST‘ Taster auf dem Trinity Board gedrückt werden muss. Das Board ist dann wieder im ,config mode‘



Program

Connected to device: ESP32-D0WD (revision 1)

Copy Trace Disconnect Erase Flash

Flash Address File

0x0 Datei auswählen WiFiTetrisC...ager_5.bin

Add File

Flash Mode: keep ▾

Flash Frequency: keep ▾

Flash Size: keep ▾

Program

```
esptool.js
Serial port WebSerial VendorID 0x1a86 ProductID 0x7523
Connecting...
Detecting chip type... ESP32
Chip Revision: 1
Chip is ESP32-D0WD (revision 1)
Features: Wi-Fi, BT, Dual Core, 240MHz, VRef calibration in efuse, Coding Scheme None
Crystal is 40MHz
MAC: 1c:69:20:b5:c2:84
Uploading stub...
Running stub...
Stub running...
Changing baudrate to 921600
Changed
If the chip does not respond to any further commands, consider using a lower baud rate.
Flash ID: 164046
```