

Bezzera Duo Bridge – Build guide

main-1eaf1f7 · github.com/larszu/bezzera-duo-protocol

Contents

1. Flashing the ESP32
2. Wiring
3. Pinout of both boards
4. Replacement display shows no values
5. Remote access

Flashing the ESP32

The bridge runs on two boards:

Board	What works
Waveshare ESP32-S3-ETH (ESP32-S3, 8 MB PSRAM, 16 MB flash, Ethernet), recommended	everything: Matter, Ethernet, update via the web UI, 24 h history
classic ESP32 (ESP32-D0WD/WROOM-32, 4 MB flash, USB via CP2102/CH340)	everything except Matter, Ethernet and update via the web UI; 5 min history

The flash page detects the board by itself. Build both locally with `bash tools/firmware_bauen.sh`. Three ways, from simplest to most flexible.

1. In the browser (recommended)

1. Open **Chrome or Edge** on a computer (Safari, Firefox and phones have no Web Serial).
2. Open the flash page: <https://larszu.github.io/bezzera-duo-protocol/>
3. Plug in the ESP32 via USB-C, click **Installieren** (install), pick the port (usually “USB JTAG/serial debug unit”).
4. After flashing the page asks for your home Wi-Fi (Improv). Pick the network, enter the password.
5. The bridge is then reachable at <http://duo.local>.

GitHub rebuilds the page on every change to `bridge/` (`.github/workflows/firmware.yml`); each version tag also gets the file as a release.

2. With Arduino CLI (from source)

```
brew install arduino-cli # macOS; Linux/Windows:
arduino.github.io/arduino-cli
arduino-cli core install esp32:esp32@3.3.12
FQBN=esp32:esp32:esp32s3:CDCOnBoot=cdc,FlashSize=16M,PartitionScheme=custom,PSRAM=opi
arduino-cli compile --fqbn $FQBN bridge/duo_bridge
arduino-cli upload --fqbn $FQBN -p /dev/cu.usbmodem... bridge/duo_bridge # Windows: -p COM5
```

- `FlashSize=16M,PartitionScheme=custom`: the own layout `bridge/duo_bridge/partitions.csv` (two 6 MB app slots, room for Bluetooth and Matter). The settings storage stays where it was, stored settings survive the change.
- `PSRAM=opi`: the ESP32-S3R8 on this board. Without PSRAM everything works, the history just covers 30 minutes instead of 24 hours.
- `CDCOnBoot=cdc`: output and commands over the USB-C port.

3. With esptool (ready-made file from a release)

```
pip install esptool
esptool --chip esp32s3 --port /dev/cu.usbmodem... write-flash 0x0 duo_bridge-vX.Y.bin
```

If the ESP32 is not detected

- Try another USB cable: many only charge.
- Force download mode: hold **BOOT**, tap **RESET**, release BOOT, then flash. Press RESET afterwards.
- macOS asks “Allow accessory to connect?” the first time – allow it.

First setup without a computer

1. Connect your phone to the Wi-Fi **duo-bridge**, password **espresso1**.
2. The page opens as a sign-in portal by itself (otherwise open any address, e.g. <http://espresso.maschine>).
3. **Einstellungen** → **Heim-WLAN** → **Netze suchen** (settings → home Wi-Fi → search), tap your network, enter the password.
4. Under **Zugang** (access) change the bridge's Wi-Fi password and set a password for the page (user **duo**). The default password is public in this repo.

Over USB: `n <ssid> <password>` (see README, USB commands).

Afterwards

- Wiring to the machine: README, ESP32 bridge → wiring.
- Scale: **Einstellungen** → **Waage** → **Bluetooth-Waage** → **Waage suchen**, tap the scale.
- From outside: [fernzugriff.en.md](#).

Wiring

Wiring plan with hole positions: [docs/breadboard_hybrid.png](#) . Which pin that is on the ESP32-S3-ETH or a classic ESP32 DevKit, and how to move from one to the other: [docs/pinbelegung.en.md](#) .

Connection	how
Mainboard black (replacement cable brown) + display brown	ESP32 GND
Mainboard red (green) ↔ display green	directly, not to the ESP32
Mainboard pink (white)	10 kΩ → GPIO17 , GPIO17 20 kΩ → GND
ESP32 GPIO18	→ display white
Display yellow	→ ESP32 GPIO15
ESP32 GPIO16	→ mainboard green (yellow)
ESP32 5V	unconnected

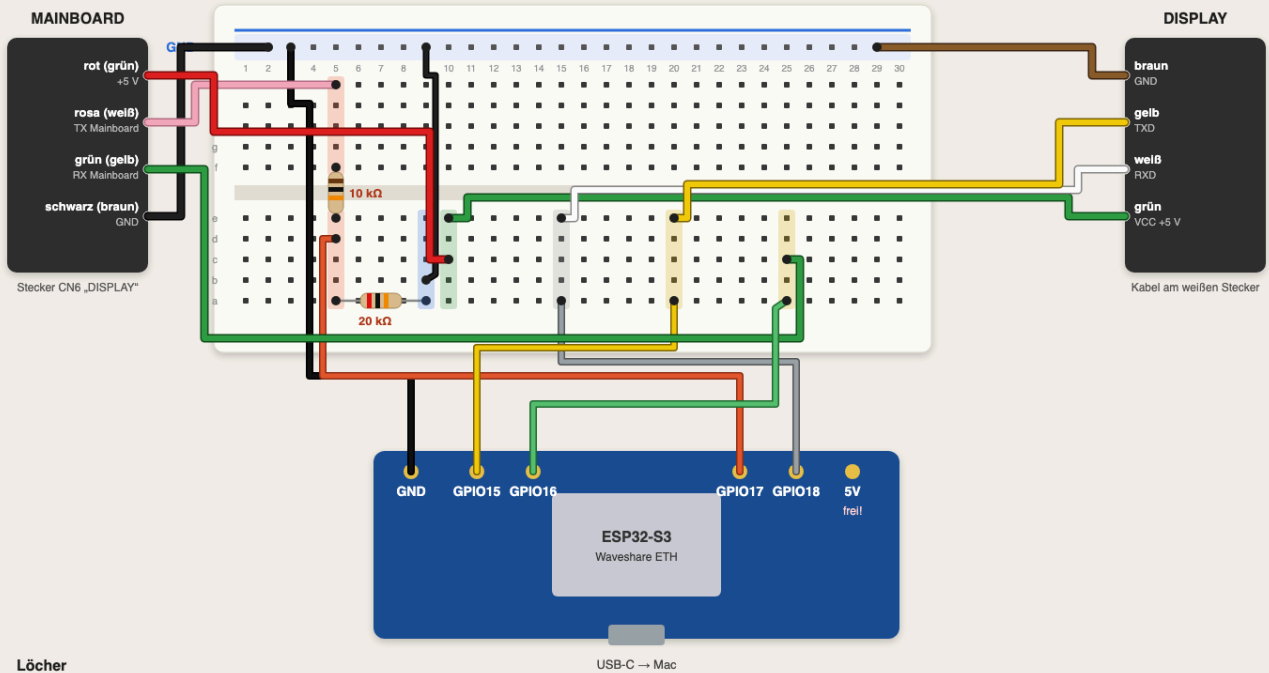
Display only, without the machine (cycling through pages, reading the flash): display green to ESP32 **5V**, brown to GND, yellow to GPIO15, white to GPIO18.

Pitfalls when building

- **Swapping display brown and display green reverses the display's polarity.** Its protection diode then shorts the machine's 5 V (~0 V on the supply). The display survived it.
- **Without a common ground** only garbage bytes arrive: connect machine GND directly to a GND pin of the ESP32.
- **Swapping 10 kΩ and 20 kΩ** gives ~1.6 V at GPIO17, right in the threshold region — garbled data. Correct: 10 kΩ between the line and GPIO17, 20 kΩ to GND.
- **Never also plug the display directly into the machine** while the ESP32 is in the line: two transmitters on one line, 5 V from two sources.

ESP32-Bridge zwischen Mainboard und Display (Hybridmodus)

Netzstecker gezogen · ESP32 per USB am Mac · 5V-Pin des ESP32 bleibt frei · Farben: Originalkabel am Mainboard (neues Kabel in Klammern)



Löcher

Spalte 5 oben (j): Mainboard rosa (neu: weiß) = TX Mainboard

5f → 5e: 10 kΩ über die Mittelrinne (braun-schwarz-orange)

5d: Kabel zu ESP32 GPIO17

5a → 9a: 20 kΩ (rot-schwarz-orange), 9b Draht zur GND-Schiene

Spalte 10: Mainboard rot (neu: grün) in 10c + Display grün in 10e → +5 V, NICHT zum ESP32

USB-C → Mac

Spalte 15: Display weiß in 15e, 15a → ESP32 GPIO18

Spalte 20: Display gelb in 20e, 20a → ESP32 GPIO15

Spalte 25: Mainboard grün (neu: gelb) in 25c, 25a → ESP32 GPIO16

GND-Schiene: Mainboard schwarz (neu: braun), Display braun, ESP32 GND

Pinout of both boards

The firmware uses **the same GPIO numbers** on both boards. When moving from one board to the other the wires stay on the same signals; only position and silkscreen differ.

Signal	GPIO	Waveshare ESP32-S3-ETH	classic ESP32 (DevKit, 30 pins)	Wire
RX from display	15	pin "15"	D15 (IO15/G15), right row, 13th from top	display yellow
TX to mainboard	16	pin "16"	RX2 (GPIO16), right row, 10th from top	mainboard green (spare cable yellow)
RX from mainboard	17	pin "17"	TX2 (GPIO17), right row, 9th from top	mainboard pink (white) via 10 kΩ, 20 kΩ to GND
TX to display	18	pin "18"	D18 (IO18), right row, 7th from top	display white
Ground	GND	"GND"	GND , right row, 14th from top (or bottom left)	mainboard black + display brown
5 V (only without machine)	–	"5V"	VIN , left row bottom	display green, only without machine

Classic DevKit, antenna up, USB down; right row from top: D23, D22, TX0, RX0, D21, D19, **D18**, D5, **TX2**, **RX2**, D4, D2, **D15**, **GND**, 3V3. Other variants (38 pins, DevKitC) carry the same GPIO names elsewhere: follow the labels, not the position.

Classic ESP32 caveats: "RX2"/"TX2" are just names (GPIO16 sends to the mainboard, GPIO17 receives); boards with PSRAM (WROVER) use GPIO16/17 internally and do not work; keep the 10 kΩ/20 kΩ divider on the mainboard input; no Ethernet, no Matter, updates only via the flash page; little RAM, so enter ntfy as `http://ntfy.sh/...` (unencrypted); calendar and Telegram need TLS and may fail.

Switching boards: machine off, move each wire to the same GPIO on the new board, flash it via the [flash page](#), set Wi-Fi, password and profiles again.

Replacement display shows no values

Fitted a new display (spare part 2.2), pages appear, touch beeps, but **no temperatures, no clock?** Then the display uses a different frame header (5A A5) than the mainboard (C6 A5). An SD card switches it over, no bridge, no software:

1. Download [display_kopf_c6.zip](#), copy the folder DWIN_SET onto an empty microSD (FAT32, 4 KB clusters, at most 8 GB).
2. Switch the machine on, insert the card into the display, pull it after a few seconds.
3. Switch the machine off and on at the main switch. The values now appear.

The card only changes one display setting (register R3 from 5A to C6), no pages or images. **Only use it if the display shows no values:** on a mainboard that itself speaks 5A A5 it would make the display silent (undo with R3=5A). Tested with display 2.2 on mainboard 2.1, background and backup first in [docs/versionen.en.md](#). No warranty.

Remote access

The bridge is a small web server on your home network. Only a secure tunnel should reach it from outside; **never** forward a router port – the bridge can switch the machine on.

First: set a password

Einstellungen → Zugang → Passwort für diese Seite (settings → access → password for this page). Without it anyone on the network can operate the machine.

Option 1: Tailscale via a device that is always on (recommended)

There is no mature Tailscale for the ESP32 itself (not enough memory for WireGuard plus the Tailscale control plane). A computer on the home network acting as a **subnet router** is enough, e.g. a home server, Raspberry Pi or NAS:

Easiest with the script from this repo, it detects the home network itself:

```
sudo tools/tailscale_heimnetz.sh
```

Or by hand:

```
## on the device in the home network (Linux)
curl -fsSL https://tailscale.com/install.sh | sh
echo 'net.ipv4.ip_forward = 1' | sudo tee /etc/sysctl.d/99-tailscale.conf && sudo
sysctl -p /etc/sysctl.d/99-tailscale.conf
sudo tailscale up --advertise-routes=192.168.0.0/24    # your home network
```

1. In the Tailscale admin (login.tailscale.com → Machines) approve the route under “Edit route settings”.
2. Install the Tailscale app on the phone and sign in.
3. Open the bridge's **IP address** in the browser (shown under Einstellungen → Heim-WLAN).
`duo.local` does not work over Tailscale because mDNS only works on the local network.

Tip: give the bridge a fixed IP in the router so the bookmark keeps working.

Option 1b: the machine as its own device in Tailscale (Docker)

Instead of exposing the whole home network, the bridge appears as its own device “bezzera” in the Tailscale app, with a fixed address for a home-screen icon. On a computer in the home network with Docker:

```
sudo docker run -d --name bezzera-tailscale --restart unless-stopped \
  -e TS_HOSTNAME=bezzera -e TS_STATE_DIR=/var/lib/tailscale -e TS_USERSPACE=true \
  -v /opt/bezzera-tailscale/state:/var/lib/tailscale tailscale/tailscale:latest
sudo docker logs bezzera-tailscale          # open the login link and confirm
sudo docker exec bezzera-tailscale tailscale serve --bg --tcp 80 tcp://<bridge IP>:80
sudo docker exec bezzera-tailscale tailscale ip -4    # address for the phone
```

--tcp instead of --http so the page answers under the name and the 100.x address. On the iPhone: connect Tailscale, open the address in Safari, Share → Add to Home Screen. Give the bridge a fixed IP in the router.

Option 2: Home Assistant

If Home Assistant has remote access (Nabu Casa or your own tunnel), the bridge is there as device “Bezzera Duo” with the switch “Maschine”. Switching on and off then works from anywhere in the Home Assistant app, without a second tunnel.

Option 3: Calendar

For fixed coffee times: **Maschine** → **Ein- und Ausschalten** → **Kalender**. Events with the keyword in the title switch the machine on beforehand. The calendar is read over the internet, so this works while you are away too.