Breadboard Power Adapter

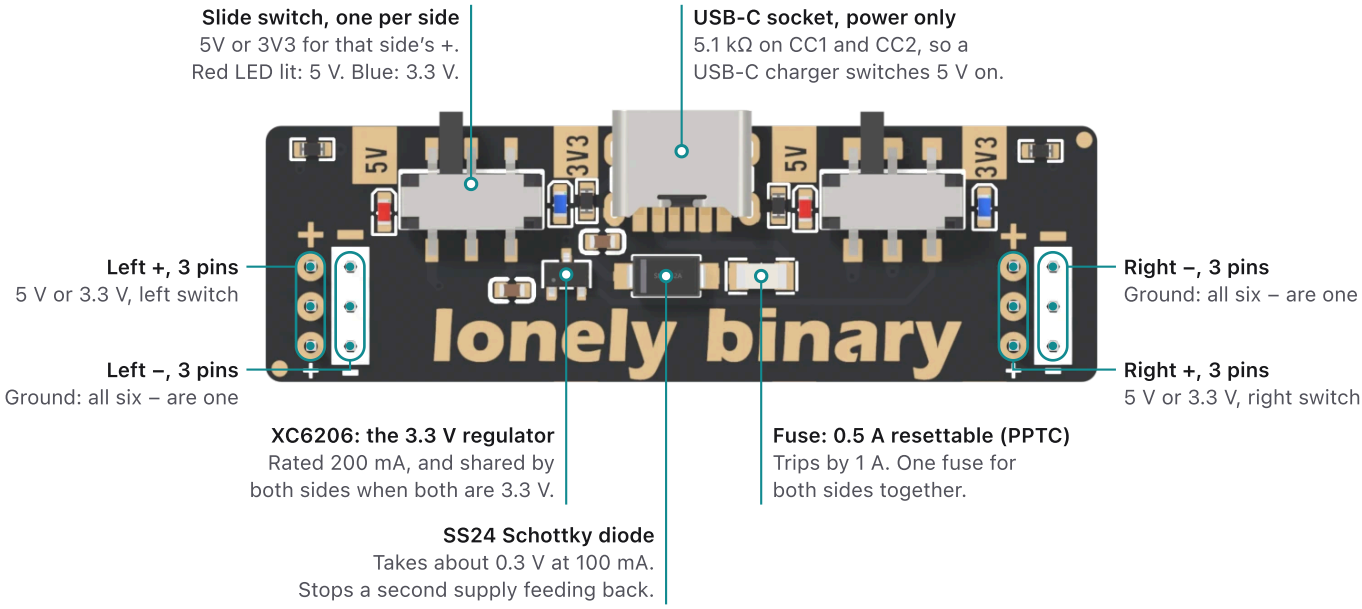
Two breadboard rails from one USB-C socket: 5 V or 3.3 V on each side, by its own switch.

Line each + up with a +

Both ends print **+ –** reading left to right, so + is the outer column at the left end and the inner one at the right: the board fits a breadboard one way round. On a breadboard printed mirror-image, one side is backwards whichever way round it goes. Nothing on the adapter shorts, **but every part wired by the printing gets its supply reversed.**

Twelve pins, four groups of three

Drawn from above with the USB-C socket at the top. Each end prints **+ –** left to right: + is the outer column at the left end and the inner column at the right. The three pins in a column are one net.



A 5 V rail that reads 4.7 V is working

The diode and the fuse take a few tenths of a volt between USB and the rail, more as the current rises. 5 V parts accept about 4.5 to 5.5 V. The 3.3 V side is regulated and holds 3.3 V.

What the 5 V rail reads

10 mA

4.83 V on the 5 V rail

4.82 V with the fuse at 0.7 Ω. The diode takes 0.17 V.

100 mA

4.69 V on the 5 V rail

4.63 V with the fuse at 0.7 Ω. The diode takes 0.30 V.

300 mA

4.60 V on the 5 V rail

4.44 V with the fuse at 0.7 Ω. The diode takes 0.35 V.

Every milliamp costs a little: the diode takes more as the current rises, and the fuse its resistance times the current. **Under 4.5 V on a 5 V rail** is the reading that resets things: too much load, or a weak charger.

Worked out, not measured: a 5.00 V port, less the SS24's typical forward voltage read off onsemi's curve, less the current times the fuse's resistance, 0.15 Ω in the big number and 0.7 Ω beneath it: two boards off one reel can differ by that much. The fitted parts are second sources, so treat every figure as about. A USB port may sit anywhere from 4.75 to 5.25 V, which moves every number by the same amount.

Watch the rail with an ESP32

Arduino IDE · Board: ESP32 Dev Module · No libraries

```
// Watch a breadboard power rail. Wiring: table on the right.
const int PIN = 34;              // ADC1, input only
const float TOP_OHMS = 10000;  // 100000 for a booster
const float BOTTOM_OHMS = 10000;
const float SCALE = (TOP_OHMS + BOTTOM_OHMS) / BOTTOM_OHMS;

float lo = 99, hi = 0, sum = 0;
int n = 0;
uint32_t last = 0;

void setup() {
  Serial.begin(115200);
  delay(500);
  Serial.println("rail watch: average, lowest, highest");
}

void loop() {
  float v = analogReadMilliVolts(PIN) / 1000.0 * SCALE;
  sum += v;
  n++;
  if (v < lo) lo = v;
  if (v > hi) hi = v;

  if (millis() - last >= 1000) {
    last = millis();
    Serial.printf("%.2f V   low %.2f   high %.2f\n",
                  sum / n, lo, hi);
    lo = 99; hi = 0; sum = 0; n = 0;
  }
  delay(1);
}
```

```
rail watch: average, lowest, highest
4.71 V   low 4.69   high 4.73
4.71 V   low 4.52   high 4.73
```

Serial Monitor at 115200, once a second: the average of about a thousand readings, then the lowest and the highest. The second line caught a burst of current a meter would average away.

Wiring

	FROM	TO
1	Rail +	10 kΩ, first end
2	10 kΩ, second end	the junction
3	The junction	10 kΩ to rail –
4	The junction	GPIO 34
5	ESP32 GND	Rail –

Power the ESP32 from its own USB cable, and plug it in before the adapter. Share ground only: never wire the rail to the ESP32's 5V or 3V3 pin as well. GPIO 34 is on ADC1 and input only, so a slip in the sketch cannot drive the rail. Two 10 kΩ halve the rail, so 5 V arrives as 2.5 V.

Two sides, one ground

Each switch sets its own side's + to 5 V or 3.3 V, so the two sides can differ. All six – pins are one ground, the USB supply's, which is what lets a 5 V part on one side and a 3.3 V part on the other talk. Both 3.3 V settings share one XC6206 rated for 200 mA. For an ESP32 or an Arduino, set a side to 5 V and wire it to the dev board's 5V or VIN pin: its own regulator makes its 3.3 V and is built for Wi-Fi's bursts. Slide a switch with the cable unplugged, not under a running circuit.

When it does not work

I plugged it in the wrong way round

The adapter is almost certainly fine: nothing shorts, and + and – swap names. Parts wired to those rails may not be. Turn it round and check each part.

Neither LED on one side is lit

A slide switch left between its two positions connects neither. Push it firmly to one end. Both sides dark: check the cable and the fuse.

The rail died and came back a minute later

The fuse tripped and reset: something drew more than it holds. If it keeps happening, measure what the circuit draws, or look for the short.

My 5 V rail reads 4.3 V

Heavy load, a weak charger, or both. Measure with the load disconnected: back to about 4.8 V means the adapter is fine and the load is the question.

An ESP32 on the 3.3 V side resets on Wi-Fi

Wi-Fi bursts pull well over 300 mA and the regulator is rated 200 mA. Feed the ESP32's 5V or VIN pin from a side set to 5 V.

Nothing lights and the rail reads 0 V

Try a cable that charges a phone, then the charger. 5 V at one end of the fuse and 0 V at the other means it has tripped: find the short.

Specifications

Input	USB-C, power only; 5.1 kΩ on CC1, CC2	**Current**	0.5 A in total, both sides; trips by 1 A
Diode	SS24 Schottky, about 0.3 V at 100 mA	**Outputs**	5 V or 3.3 V per side, by slide switch
5 V rail	About 4.5 to 4.9 V under ordinary loads	**3.3 V rail**	XC6206, 200 mA, shared by both sides
Pins	2 × (3 +, 3 –), 2.54 mm pitch, one way round	**Board**	53.34 × 16.36 mm, pins 7.9 mm below