HC-SR04 + ESP32 — Quick Start

One page to get from a box of parts to a live distance reading. The full explanation lives in the blog post; this is the bench version.

What you need

Part	Value	Qty
ESP32 DevKit	ESP32-WROOM-32	1
Ultrasonic sensor	HC-SR04	1
Resistor	1 kOhm	1
Resistor	2.2 kOhm	1
Breadboard	400 or 830 tie-point	1
Jumper wires	M-M / M-F	~6

Software: Arduino IDE with the ESP32 board package. The plotter sketch also needs the **ArduinoJson** library (Library Manager → "ArduinoJson").

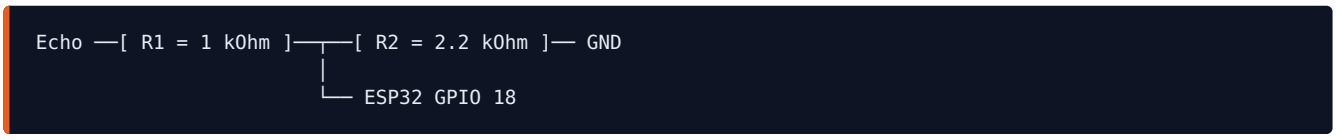
Wire it — 3 steps

USB unplugged for all of this.

1. Power and ground. HC-SR04 GND → ESP32 GND (black). Then HC-SR04 VCC → ESP32 **5V / VIN** (red). Not 3.3 V — on 3.3 V the range shrinks and the readings get noisy.

2. Trigger. HC-SR04 Trig → ESP32 GPIO 5 (blue). The ESP32's 3.3 V output is enough to trigger the module.

3. Echo, through the divider. This is the step you cannot skip. Echo pulses to 5 V and the ESP32 is a 3.3 V part.



The node sits at 5 V × 2.2 / 3.2 = **3.44 V** — above the 3.3 V rail, below the ESP32's 3.6 V absolute maximum. If you want it under the rail, use 2.2 kOhm for R1 and 3.3 kOhm for R2 and you land at 3.0 V.

HC-SR04	ESP32	Wire
VCC	5V / VIN	red
Trig	GPIO 5	blue
Echo	divider node → GPIO 18	yellow
GND	GND	black

Avoid GPIO 6–11: the ESP32 uses them for its flash chip. Check VCC and GND are not swapped, then plug in USB.

Upload

Serial test first — it proves the wiring before anything else can hide a mistake.

1. Open `firmware/hc-sr04_test/hc-sr04_test.ino` in the Arduino IDE.
2. Tools → Board → **ESP32 Dev Module**, and pick your COM port.
3. Upload, then open the Serial Monitor at **115200 baud**.

Expected output, moving a hand toward the sensor:

```
HC-SR04 test ready. Distance in cm:
34.2 cm
28.9 cm
19.9 cm
11.3 cm
-- out of range / no echo --
```

Then the live dashboard.

1. Open `firmware/hc-sr04_plotter/hc-sr04_plotter.ino` and upload it. One file, one upload — the web page is compiled into the sketch as `PROGMEM`, so there is no filesystem step and no second upload.
2. On your phone, join the WiFi network `OmArTronics-HC-SR04`, password `ultrasonic`, and open `http://10.10.10.1`.

You get a live number, a rolling chart of the last 100 readings, min/max/average cards, and buttons for PNG snapshot, CSV export and stats reset.

Troubleshooting

Symptom	Cause	Fix
Every reading is `-- out of range / no echo --`	Trig or Echo on the wrong pin, or Echo not reaching GPIO 18 through the divider	Re-check GPIO 5 / GPIO 18; measure continuity from the R1/R2 node to GPIO 18
Readings are 0 or wild garbage	VCC on 3.3 V instead of 5 V, or VCC/GND swapped	Move VCC to the 5V/VIN pin; verify GND is common
Numbers jump by several cm on a motionless target	Normal single-sample noise, plus soft or tilted target	Median-filter 3–5 readings; aim at a flat, hard surface at least 0.5 m²
Readings look fine but stop below ~2 cm	The 2 cm dead zone plus the sketch's `DIST_MIN_CM` filter	Expected — the transducer is still ringing from its own burst that close
Phone joins the AP but `10.10.10.1` does not load	Phone fell back to mobile data, or you typed `https`	Disable mobile data for the test; the dashboard is plain `http://`
Compile error: `ArduinoJson.h: No such file`	Library missing	Library Manager → install **ArduinoJson**
Upload fails / port busy	Serial Monitor holding the COM port	Close the Serial Monitor, unplug and replug, upload again

Files

`firmware/hc-sr04_test/` · `firmware/hc-sr04_plotter/` · `electronics/wiring.md` · `electronics/schematic.svg` · `cad/` (OmBlock STL) · `animation/hc-sr04_animation.html`