مستشعر المسافة بالموجات فوق الصوتية HC-SR04 مع ESP32

مختبر مستشعرات OmArTronics · المشروع 01 · دليل طاولة العمل

رفيق مطبوع للدرس الكامل. فيه كل ما تحتاجه على طاولة العمل، وليس فيه ما لا تحتاجه إلا على الأريكة.

حقائق سريعة

المستشعر	وحدة مسافة بالموجات فوق الصوتية HC-SR04 (تصميم ElecFreaks، ونسخ مقلّدة كثيرة)
المبدأ	زمن رحلة دفعة فوق صوتية بتردد 40 كيلوهرتز
الواجهة	رقمية: نبضة إطلاق 10 ميكروثانية دخولاً، وعرض نبضة Echo خروجاً
التغذية	5 فولت تيار مستمر، نحو 15 ملي أمبير
المنطق	Trig يقبل تشغيلاً بـ 3.3 فولت · **Echo يُخرج 5 فولت ولا بد من تقسيمه**
المدى	2 إلى 400 سم (ورقة البيانات) · تمييز 0.3 سم · ±3 مم في أفضل الحالات
الحزمة	مخروط بنحو 15 درجة
زمن الدورة	60 ملي ثانية كحد أدنى بين نبضتين
الحجم	45 × 20 × 15 مم
المتحكم المستخدم هنا	ESP32-WROOM-32، وTrig على GPIO 5، وEcho على GPIO 18

1. كيف يعمل، في 150 كلمة

تطلق الوحدة ثماني دورات من صوت بتردد 40 كيلوهرتز من أسطوانة المرسِل حين ترفع Trig 10 ميكروثانية. تنتقل تلك الدفعة عبر الهواء، وترتد عمّا يقف أمامها، وتعود إلى أسطوانة المستقبِل. ثم ترفع الوحدة دبوس Echo وتبقيه مرتفعاً بقدر زمن الرحلة تماماً. ويقيس متحكمك زمن تلك النبضة ويجري الحساب.

الكلمة المهمة هي ذهاباً وإياباً: طار الصوت إلى الهدف ثم عاد، فالزمن المقاس يغطي ضعف المسافة التي تهمّك.

المعادلة

```
distance_cm = (echo_time_us × 0.0343) / 2
```

`0.0343` هي سرعة الصوت بالسنتيمتر لكل ميكروثانية عند 20 درجة مئوية. واختصار ورقة البيانات `distance_cm = echo_time_us / 58` هو الشيء نفسه، لأن 1 / (0.0343 / 2) = 58.3.

مثال محلول — يبقى Echo مرتفعاً 1160 ميكروثانية: 1160 × 0.0343 = 39.79 سم من مسار الصوت الكلي، ÷ 2 = **19.9 سم**. وللتحقق: 1160 / 58 = 20.0 سم.

ترتفع سرعة الصوت نحو 0.6 متر في الثانية لكل درجة مئوية، فالقراءة غير المعوَّضة تنحرف نحو 0.17 % لكل درجة ابتعاداً عن 20 درجة مئوية.

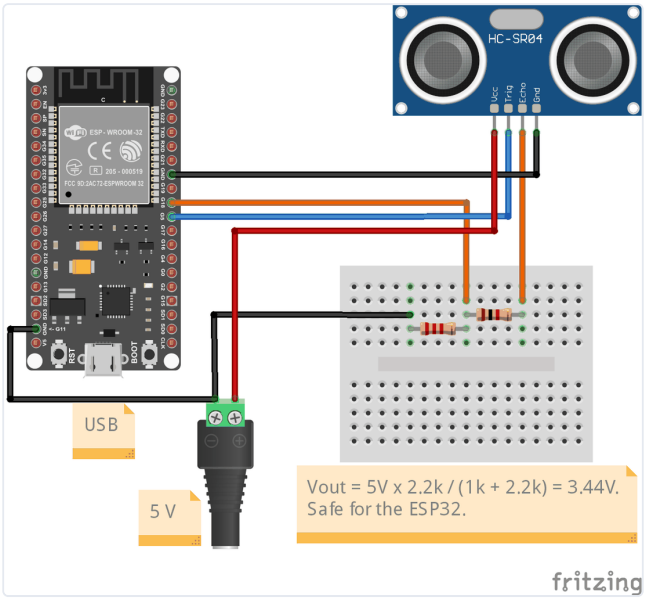
2. قائمة المواد

المكوّن	الطراز / القيمة	الكمية	الغرض
ESP32 DevKit	ESP32-WROOM-32	1	يقيس زمن نبضة Echo، ويقدّم لوحة التحكم
مستشعر فوق صوتي	HC-SR04	1	يطلق الدفعة، ويعيد نبضة Echo
مقاومة	1 كيلوأوم، ¼ واط	1	R1 في مقسّم Echo (من Echo إلى العقدة)
مقاومة	2.2 كيلوأوم، ¼ واط	1	R2 في مقسّم Echo (من العقدة إلى GND)
لوحة تجارب	400 أو 830 نقطة توصيل	1	توصيل بلا لحام
أسلاك توصيل	M-M / M-F	نحو 6	التوصيلات

نحو 6 إلى 9 يورو في المجموع، أكثرها ثمن ESP32. أما HC-SR04 نفسه فبـ 1 إلى 2 يورو.

عن قيمتي المقاومتين: 2 كيلوأوم ليست قيمة قياسية في سلسلة E12، فلا تجدها في معظم الأطقم. وتعطي 1 كيلوأوم / 2.2 كيلوأوم جهد 3.44 فولت على العقدة. وإن أردت البقاء تحت خط 3.3 فولت، فإن 2.2 كيلوأوم / 3.3 كيلوأوم تعطي 3.0 فولت، والقيمتان موجودتان في كل طقم.

3. التوصيل



دبوس HC-SR04	دبوس ESP32	السلك	ملاحظة
VCC	5V / VIN	أحمر	5 فولت كاملة — على 3.3 فولت يتقلص المدى ويكثر ضجيج الإشارة
Trig	GPIO 5	أزرق	تشغيل بـ 3.3 فولت من ESP32 يكفي
Echo	من عقدة المقسّم إلى GPIO 18	أصفر	خرج 5 فولت — **لا بد** من تقسيمه أولاً
GND	GND	أسود	أرضي مشترك، وصّله أولاً

المقسّم

```
Echo ──[ R1 = 1 kOhm ]──┬──[ R2 = 2.2 kOhm ]── GND
                        │
                        └── ESP32 GPIO 18
```

جهد العقدة = 5 فولت × 2.2 / 3.2 = 5 × (R1 + R2) / R2 = **3.44 فولت**

ترتيب التركيب: USB مفصول، ثم GND، ثم VCC، ثم Trig، ثم ابنِ المقسّم، ثم مدّ العقدة إلى GPIO 18، ثم تأكد أن VCC وGND غير متبادلين، ثم وصّل USB.

ملاحظات السلامة

- لا تصل Echo مباشرة بدبوس GPIO أبداً. المقسّم ليس اختيارياً؛ فـ 5 فولت على دبوس 3.3 فولت تجهده وتفسده مع الوقت.
- لا تستخدم GPIO 6 حتى 11 — فـ ESP32 يوصّلها بشريحة الفلاش. أما GPIO 5 و18 فدبوسان عامّان آمنان.
- أعطِ المستشعر تغذية 5 فولت حقيقية. وتغذية USB عبر دبوس 5V/VIN في ESP32 تكفي.
- صوّبه نحو سطح مستوٍ مساحته 0.5 م² على الأقل حتى يعود صدى نظيف.

---

4. سكتش الاختبار — القائمة الكاملة

`firmware/hc-sr04_test/hc-sr04_test.ino` · في Arduino IDE، واللوحة = **ESP32 Dev Module**، وSerial Monitor بسرعة **115200 باود**.

```
/*
 * HC-SR04 Ultrasonic Distance — Test Sketch
 * OmArTronics · Project 01 · 2026-07-25 · MIT License
 *
 * Prints distance in cm over Serial at 115200 baud.
 * Wiring: Trig -> GPIO 5, Echo -> GPIO 18 through a 1k/2.2k divider (5V -> 3.44V),
 *         VCC -> 5V, GND -> GND.  See wiring.md.
 *
 * Single file — just open in the Arduino IDE and upload. Change pins below.
 */

// ===== Settings (edit these) =====
#define TRIG_PIN            5       // ESP32 GPIO 5  -> HC-SR04 Trig
#define ECHO_PIN            18      // ESP32 GPIO 18 <- Echo via 1k/2.2k divider
#define READING_INTERVAL_MS 60      // datasheet suggests >= 60 ms between pings
#define ECHO_TIMEOUT_US     25000   // ~430 cm round trip; pulseIn gives up after this
#define SERIAL_BAUD         115200
#define SOUND_CM_PER_US     0.0343f // speed of sound at 20 C, cm per microsecond
#define DIST_MIN_CM         2.0f    // datasheet minimum
#define DIST_MAX_CM         400.0f  // datasheet maximum
#define CALIBRATION_OFFSET  0.0f    // cm added after conversion (tune vs a ruler)
#define CALIBRATION_SCALE   1.0f    // multiply reading (1.0 = no change)
// ================================

// One measurement. Returns distance in cm, or NAN if the reading is invalid.
float readDistanceCm() {
  // 10 us trigger pulse tells the module to fire an 8-cycle 40 kHz burst.
  digitalWrite(TRIG_PIN, LOW);
  delayMicroseconds(2);
  digitalWrite(TRIG_PIN, HIGH);
  delayMicroseconds(10);
  digitalWrite(TRIG_PIN, LOW);

  // pulseIn measures how long Echo stays HIGH, in microseconds.
  // It returns 0 if no echo arrives before the timeout.
  unsigned long echo_us = pulseIn(ECHO_PIN, HIGH, ECHO_TIMEOUT_US);
  if (echo_us == 0) return NAN;                 // timeout: nothing in range

  // Convert time of flight to distance (divide by 2 for the round trip).
  float cm = (echo_us * SOUND_CM_PER_US) / 2.0f;
  cm = cm * CALIBRATION_SCALE + CALIBRATION_OFFSET;
```

```
  // Reject readings outside the datasheet range.
  if (cm < DIST_MIN_CM || cm > DIST_MAX_CM) return NAN;
  return cm;
}

void setup() {
  Serial.begin(SERIAL_BAUD);
  pinMode(TRIG_PIN, OUTPUT);
  pinMode(ECHO_PIN, INPUT);
  digitalWrite(TRIG_PIN, LOW);
  delay(50);
  Serial.println(F("HC-SR04 test ready. Distance in cm:"));
}

void loop() {
  float cm = readDistanceCm();
  if (isnan(cm)) {
    Serial.println(F("-- out of range / no echo --"));
  } else {
    Serial.print(cm, 1);
    Serial.println(F(" cm"));
  }
  delay(READING_INTERVAL_MS);
}
```

الخرج المتوقع على Serial، مع تحريك يدك نحو المستشعر:

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

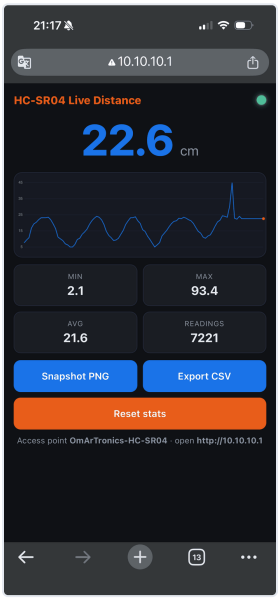
إن كان هذا خاطئاً فالتوصيل خاطئ، ولن يصلحه أي خادم ويب.

---

5. لوحة التحكم المباشرة على الويب — خمس خطوات

`firmware/hc-sr04_plotter/hc-sr04_plotter.ino` · يحتاج مكتبة **ArduinoJson**.

1. افتح سكتش الراسم وارفعه. ملف واحد ورفعة واحدة — فلوحة التحكم مترجَمة داخل السكتش في `PROGMEM`، فلا نظام ملفات ولا إضافة ولا رفعة ثانية.
2. على هاتفك، انضم إلى شبكة WiFi باسم **`OmArTronics-HC-SR04`** (كلمة المرور `ultrasonic`).
3. افتح **`http://10.10.10.1`** في المتصفح. `http` عادي، لا `https`.
4. اقرأ القيمة المباشرة، والمخطط المتدحرج لآخر 100 قراءة، وبطاقات الحد الأدنى والحد الأقصى والمتوسط والعدد. تستعلم الصفحة خمس مرات في الثانية.
5. استخدم الأزرار: **اللقطة** تحفظ المخطط صورة PNG، و**التصدير** ينزّل القراءات المخزّنة ملف CSV، و**التصفير** يمسح الإحصاءات على ESP32.



ما فعله المستشعر فعلاً

مقيس على هذا البناء، من تصدير CSV ومن إحصاءات لوحة التحكم — لا من ورقة البيانات.

الملاحظة	المقيس
جدار ساكن على بعد نحو 89 سم، 24 قراءة في 1.55 ثانية	88.7 إلى 89.6 سم · وسيط 89.1 · انحراف معياري 1.8 مم · **9 مم بين القمة والقاع**
مدى قريب، نافذة 2.4 ثانية عند نحو 8 سم	متوسط 7.67 سم · مدى 6 مم
قيمة شاذة في عينة مفردة داخل تصدير من 100 عينة	22.3 ثم **32.7** ثم 23.5 سم، والهدف لم يتحرك
الحد الأدنى في الجلسة	2.1 سم (جزئياً بفعل مرشّح `DIST_MIN_CM` في السكتش نفسه)
الحد الأقصى في الجلسة	93.4 سم (وبلغت جلسة سابقة ذروة عند 101.4 سم)
تباعد عينات Serial	وسيط 78 مللي ثانية مقابل فترة مضبوطة على 60 مللي ثانية
تباعد عينات CSV	وسيط 205 مللي ثانية — فذلك الملف يحمل ما استعلم عنه المتصفح

المدى 9 مم يساوي ثلاثة أضعاف التمييز الذي تعلنه ورقة البيانات، على هدف لم يتحرك قط. وذكر قراءة HC-SR04 بمنزلتين عشريتين خيال. **إن كان مشروعك يتصرف بناء على المسافة، فخذ وسيط ثلاث أو خمس قراءات.**

وما يلي غير مقيس هنا، بل متوقع من ورقة البيانات: الأهداف الطرية أو المائلة تبعثر الصدى فتنقطع القراءة؛ والمخروط بنحو 15 درجة يبلّغ عن أقرب جسم في أي موضع داخله؛ والثابت المثبّت 0.0343 ينحرف مع الحرارة؛ ووحدتان تطلقان معاً قد تسمع كل منهما الأخرى.

6. حل المشكلات

العَرَض	السبب المرجّح	الحل
القراءة دائماً `-- out of range / no echo --`	Trig أو Echo على دبوس خاطئ، أو عقدة المقسّم لا تصل إلى GPIO 18	راجع GPIO 5 وGPIO 18؛ وافحص الاستمرارية من وصلة R1/R2 إلى GPIO 18
القراءات 0 أو بلا معنى	VCC على 3.3 فولت، أو VCC وGND متبادلان	انقل VCC إلى 5V/VIN؛ وتأكد من أرضي مشترك
القيم تقفز عدة سنتيمترات على هدف ساكن	ضجيج العينة المفردة؛ أو هدف طري أو مائل	رشّح 3 إلى 5 عينات بالوسيط؛ وصوّب عمودياً نحو سطح مستوٍ صلب
شوكة منفردة بين حين وآخر (32.7 سم مثلاً)	صدى كاذب واحد	ارفض العينات التي تبتعد عن الوسيط الجاري بأكثر من عتبة معيّنة
لا شيء دون نحو 2 سم	المحوّل ما زال يرنّ من دفعته هو، إضافة إلى مرشّح `DIST_MIN_CM`	سلوك متوقع — صمّم مشروعك حول المنطقة الميتة
`ArduinoJson.h: No such file or directory`	المكتبة غير مثبّتة	ثبّت **ArduinoJson** من Library Manager
الرفع يفشل والمنفذ مشغول	Serial Monitor يحتجز منفذ COM	أغلق Serial Monitor، وأعد وصل الكابل، ثم ارفع مجدداً
الهاتف متصل بنقطة الوصول لكن الصفحة لا تُحمّل	الهاتف رجع إلى بيانات الجوّال، أو كُتب `https`	أطفئ بيانات الجوّال أثناء الاختبار؛ واستخدم `http://10.10.10.1`

7. الملفات والروابط

الملف	ما هو
`firmware/hc-sr04_test/`	سكتش الاختبار عبر Serial (القسم 4)
`firmware/hc-sr04_plotter/`	الراسم البياني المباشر على الويب، ولوحة التحكم مضمّنة، ورفعة واحدة
`firmware/hc-sr04_plotter/web_src/`	لوحة التحكم كملفات HTML وCSS وJS قابلة للتحرير
`electronics/schematic.svg`	مخطط التوصيل مع مقسّم Echo
`electronics/wiring.md`	جدول الدبابيس، وحساب المقسّم، وترتيب التركيب
`cad/OmBlock HC-SR04.stl`	حامل المستشعر القابل للطباعة ثلاثية الأبعاد لشبكة OmGrid
`animation/hc-sr04_animation.html`	رسم متحرك تفاعلي لزمن الرحلة

- **الدرس الكامل:** https://omartronics.com/hc-sr04-ultrasonic-sensor-esp32/
- **ورقة البيانات:** https://cdn.sparkfun.com/datasheets/Sensors/Proximity/HCSR04.pdf
- **التالي في السلسلة:** PIR HC-SR501 — حركة دون موجة صوت واحدة.

البرنامج الثابت مرخّص بترخيص *MIT* — استخدمه في ما تشاء. أما ملفات التصميم والتوثيق فللاستعمال الشخصي. © 2026 *OmArTronics* ·
omartronics.com