

# Elektro<sub>xy</sub>

## ESP32 NodeMCU

WLAN- und Bluetooth-Entwicklungsboard mit ESP32-WROOM-32

Kurzanleitung

ElektroXY · Bach IT Service UG (haftungsbeschränkt)

Lindenstr. 7 · 53506 Kesseling · [www.elektroxy.de](http://www.elektroxy.de)

## Worum es geht

Der ESP32 ist ein Mikrocontroller mit eingebautem WLAN und Bluetooth. Auf diesem Board sitzt er als fertiges Funkmodul ESP32-WROOM-32 mit Antenne, daneben ein USB-Seriell-Wandler und ein Spannungsregler. Mehr braucht es nicht: USB-Kabel anstecken, Programm übertragen, fertig.

Gegenüber dem Vorgänger ESP8266 bringt er zwei Rechenkerne mit bis zu 240 MHz, deutlich mehr Speicher, Bluetooth zusätzlich zum WLAN und erheblich mehr Anschlüsse. Programmiert wird er wie ein Arduino — dieselbe Entwicklungsumgebung, dieselben Bibliotheken, nur ein anderes Board im Menü.

Typische Einsätze: Messwerte ins heimische Netz melden, eine kleine Weboberfläche bereitstellen, Geräte über MQTT steuern oder per Bluetooth mit dem Handy sprechen.

## Technische Daten

| Eigenschaft         | Wert                                                                     |
|---------------------|--------------------------------------------------------------------------|
| Funkmodul           | ESP32-WROOM-32 mit Leiterbahn-Antenne                                    |
| Prozessor           | Xtensa LX6, zwei Kerne, 80 bis 240 MHz                                   |
| Arbeitsspeicher     | 520 KB SRAM                                                              |
| Programmspeicher    | 4 MB Flash                                                               |
| WLAN                | 802.11 b/g/n, 2,4 GHz                                                    |
| Bluetooth           | Version 4.2, Classic und Low Energy                                      |
| USB-Anschluss       | Micro-USB über CP2102-Wandler                                            |
| Nutzbare Anschlüsse | je nach Bauform 30 oder 38 Stiftleisten-Pins                             |
| Schnittstellen      | I <sup>2</sup> C, SPI, I <sup>2</sup> S, UART, CAN, ADC, DAC, Touch, PWM |
| Abmessungen         | ca. 55 × 28 mm                                                           |

## Stromversorgung und Spannungspegel

Versorgen können Sie das Board auf zwei Wegen:

| Anschluss | Spannung | Hinweis                                               |
|-----------|----------|-------------------------------------------------------|
| Micro-USB | 5 V      | der Normalfall, auch zum Programmieren                |
| VIN       | 5 V      | geht auf denselben Regler wie der USB-Anschluss       |
| 3V3       | 3,3 V    | Ausgang des Reglers, versorgt angeschlossene Sensoren |

Achten Sie auf eine ausreichend kräftige Quelle. Im Ruhezustand genügen wenige Milliampere, beim Senden über WLAN zieht das Modul jedoch kurzzeitig einige hundert Milliampere. Ein Netzteil mit mindestens 500 mA ist Pflicht — viele sporadische Neustarts gehen auf eine zu schwache Versorgung zurück und nicht auf einen Fehler im Programm.

**Alle Ein- und Ausgänge arbeiten mit 3,3 V und sind nicht 5-Volt-fest. Ein 5-V-Signal direkt an einen GPIO zerstört den Anschluss. Für 5-V-Sensoren brauchen Sie einen Pegelwandler oder einen Spannungsteiler.**

## Anschlüsse

### Pins mit Besonderheiten

Die meisten Anschlüsse sind frei verwendbar. Diese hier brauchen Aufmerksamkeit — sie sind die häufigste Ursache für Projekte, die scheinbar grundlos nicht starten:

| Pin                 | Besonderheit                                    | Empfehlung                                              |
|---------------------|-------------------------------------------------|---------------------------------------------------------|
| GPIO 6 – 11         | fest mit dem Flash-Speicher verbunden           | keinesfalls verwenden, auch wenn herausgeführt          |
| GPIO 34, 35, 36, 39 | nur Eingang, keine internen Pull-up-Widerstände | für Taster einen externen Widerstand vorsehen           |
| GPIO 0              | bestimmt beim Start die Betriebsart             | beim Einschalten nicht auf Masse ziehen                 |
| GPIO 2              | wirkt beim Start mit, meist die Onboard-LED     | beim Einschalten nicht fest auf High legen              |
| GPIO 12             | wirkt beim Start mit                            | beim Einschalten keinen Pull-up anschließen             |
| GPIO 1, 3           | belegt durch die USB-Verbindung                 | nicht anderweitig nutzen, solange USB angeschlossen ist |

### Standardbelegung der Schnittstellen

| Schnittstelle    | Signal                    | Pin                            |
|------------------|---------------------------|--------------------------------|
| I <sup>2</sup> C | SDA / SCL                 | GPIO 21 / GPIO 22              |
| SPI (VSPI)       | MOSI / MISO / SCK / CS    | GPIO 23 / 19 / 18 / 5          |
| UART0            | TX / RX                   | GPIO 1 / GPIO 3 (USB)          |
| DAC              | zwei echte Analogausgänge | GPIO 25 / GPIO 26              |
| ADC1             | Analogeingänge            | GPIO 32 bis 39                 |
| ADC2             | Analogeingänge            | GPIO 0, 2, 4, 12 – 15, 25 – 27 |

Die Zuordnung von I<sup>2</sup>C und SPI ist nur eine Voreinstellung — beim ESP32 lassen sich die meisten Funktionen frei auf andere Pins legen.

**Wichtig für Analogmessungen:** Die Eingänge der Gruppe ADC2 lassen sich nicht benutzen, solange das WLAN aktiv ist. Wer misst und gleichzeitig funkt, sollte ausschließlich Pins aus ADC1 verwenden.

## Arduino-IDE einrichten

### 1. Treiber

Die USB-Brücke CP2102 wird von Windows 10 und 11 in der Regel selbst erkannt. Taucht kein neuer COM-Port auf, installieren Sie den CP210x-Treiber von Silicon Labs. Unter Linux und macOS ist kein Treiber nötig.

## 2. Boardpaket hinzufügen

Öffnen Sie in der Arduino-IDE die Einstellungen und tragen Sie unter „Zusätzliche Boardverwalter-URLs“ diese Adresse ein:

```
https://espressif.github.io/arduino-esp32/package_esp32_index.json
```

Danach im Boardverwalter nach „esp32“ suchen und das Paket von Espressif installieren. Das dauert ein paar Minuten, es werden mehrere hundert Megabyte geladen.

## 3. Board auswählen

| Einstellung  | Wert                                           |
|--------------|------------------------------------------------|
| Board        | ESP32 Dev Module                               |
| Upload Speed | 921600 (bei Problemen 115200)                  |
| Flash Size   | 4MB (32Mb)                                     |
| Port         | der COM-Port, der beim Anstecken neu erscheint |

## Erstes Programm

Der folgende Sketch lässt die Onboard-LED blinken und meldet sich über die serielle Schnittstelle. Damit prüfen Sie in einem Rutsch, ob Übertragung, Ausführung und Rückkanal funktionieren:

```
#define LED_PIN 2          // Onboard-LED der meisten NodeMCU-Boards

void setup() {
  Serial.begin(115200);
  pinMode(LED_PIN, OUTPUT);
  Serial.println();
  Serial.println("ESP32 gestartet.");
  Serial.print("Taktfrequenz: ");
  Serial.print(getCpuFrequencyMhz());
  Serial.println(" MHz");
  Serial.print("Freier Arbeitsspeicher: ");
  Serial.print(ESP.getFreeHeap() / 1024);
  Serial.println(" KB");
}

void loop() {
  digitalWrite(LED_PIN, HIGH);
  delay(500);
  digitalWrite(LED_PIN, LOW);
  delay(500);
  Serial.print("Laufzeit: ");
  Serial.print(millis() / 1000);
  Serial.println(" s");
}
```

Stellen Sie den seriellen Monitor auf 115200 Baud. Erscheinen statt Text nur wirre Zeichen, stimmt die Baudrate nicht.

Nicht jede Bauform hat eine Onboard-LED. Bleibt es dunkel, während der serielle Monitor ordentlich zählt, ist alles in Ordnung — dann fehlt schlicht die LED.

## Ins WLAN bringen

Das hier ist der Sketch, den Sie in der Praxis als Grundgerüst immer wieder verwenden werden:

```
#include <WiFi.h>

const char* WLAN_NAME    = "MeinNetzwerk";
const char* WLAN_PASSWORT = "geheim";

void setup() {
    Serial.begin(115200);
    delay(500);

    WiFi.mode(WIFI_STA);          // nur als Teilnehmer, kein eigener Hotspot
    WiFi.begin(WLAN_NAME, WLAN_PASSWORT);

    Serial.print("Verbinde");
    unsigned long start = millis();
    while (WiFi.status() != WL_CONNECTED) {
        if (millis() - start > 20000) {          // nach 20 s aufgeben
            Serial.println(" - keine Verbindung. Neustart.");
            ESP.restart();
        }
        Serial.print(".");
        delay(500);
    }

    Serial.println();
    Serial.print("Verbunden. IP-Adresse: ");
    Serial.println(WiFi.localIP());
    Serial.print("Signalstaerke: ");
    Serial.print(WiFi.RSSI());
    Serial.println(" dBm");
}

void loop() {
}
```

Zur Signalstärke: Werte um -50 dBm sind sehr gut, ab etwa -80 dBm wird die Verbindung unzuverlässig. Liegt Ihr Wert darunter, hilft ein anderer Aufstellort mehr als jede Änderung am Programm.

**Der ESP32 funkt ausschließlich auf 2,4 GHz. Ein reines 5-GHz-Netz sieht er nicht. Bei Routern, die beide Bänder unter demselben Namen betreiben, funktioniert es in der Regel trotzdem.**

## Batteriebetrieb und Tiefschlaf

Für Messstationen, die nur alle paar Minuten einen Wert melden, lohnt sich der Tiefschlaf. Der Prozessor wird dabei fast vollständig abgeschaltet und über einen Zeitgeber wieder geweckt. Nach dem Aufwachen startet das Programm von vorn — der Ablauf gehört deshalb komplett in setup():

```
#define SCHLAFDAUER_SEKUNDEN 300

void setup() {
    Serial.begin(115200);
```

```

Serial.println("Wach. Messung laeuft ...");

// hier messen und den Wert verschicken

Serial.println("Gehe schlafen.");
Serial.flush();
esp_sleep_enable_timer_wakeup(SCHLAFDAUER_SEKUNDEN * 1000000ULL);
esp_deep_sleep_start();
}

void loop() {
  // wird nie erreicht
}

```

Beachten Sie, dass das nackte Funkmodul im Tiefschlaf nur wenige Mikroampere benötigt, das komplette Board aber deutlich mehr: Spannungsregler und USB-Wandler laufen weiter. Für echten Langzeitbetrieb aus einer Batterie ist daher ein Board ohne USB-Wandler oder ein direkt beschaltetes Modul die bessere Wahl.

## Fehlersuche

| Beobachtung                                 | Wahrscheinliche Ursache                                  | Abhilfe                                                      |
|---------------------------------------------|----------------------------------------------------------|--------------------------------------------------------------|
| Kein COM-Port sichtbar                      | Treiber fehlt oder reines Ladekabel ohne Datenadern      | CP210x-Treiber installieren, anderes USB-Kabel probieren     |
| Upload bleibt bei „Connecting ...“ stehen   | Board wechselt nicht selbsttätig in den Programmiermodus | BOOT gedrückt halten, kurz EN/RST tippen, BOOT loslassen     |
| Upload bricht mit Prüfsummenfehler ab       | Übertragungsrate zu hoch oder Kabel zu lang              | Upload Speed auf 115200 stellen                              |
| Serieller Monitor zeigt wirre Zeichen       | Falsche Baudrate                                         | Auf 115200 stellen                                           |
| Board startet immer wieder neu              | Stromversorgung zu schwach, bricht beim Senden ein       | Netzteil mit mindestens 500 mA, kurzes Kabel                 |
| Programm startet gar nicht                  | GPIO 0, 2 oder 12 beim Einschalten falsch beschaltet     | Diese Pins beim Start freilassen, siehe Abschnitt Anschlüsse |
| Analogwert bleibt auf 0 oder springt        | Pin gehört zu ADC2 und das WLAN ist aktiv                | Auf einen Pin aus ADC1 wechseln (GPIO 32 bis 39)             |
| WLAN wird nicht gefunden                    | Netz funkt nur auf 5 GHz                                 | 2,4-GHz-Band am Router aktivieren                            |
| Ein Eingang reagiert nicht auf einen Taster | GPIO 34 bis 39 haben keine internen Pull-ups             | Externen Widerstand von 10 kOhm nach 3,3 V setzen            |

Angaben ohne Gewähr. Alle Werte beziehen sich auf das Funkmodul ESP32-WROOM-32; maßgeblich ist dessen jeweils aktuelles Datenblatt. Die Belegung der Stiftleisten kann je nach Bauform des Boards abweichen.