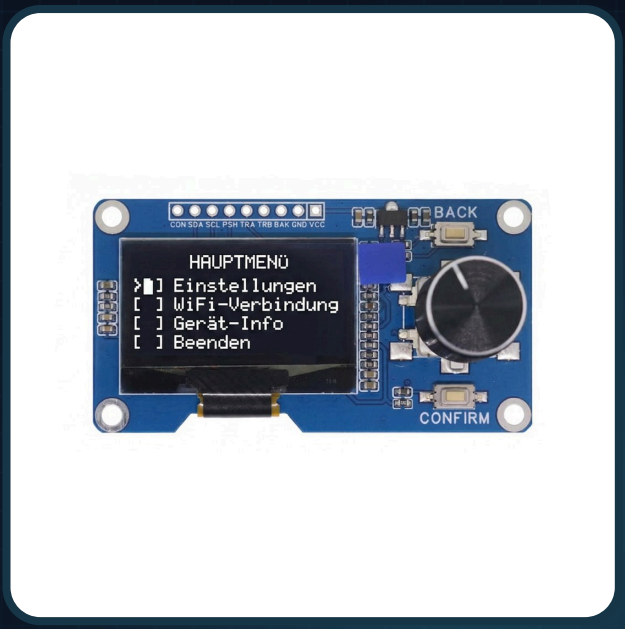
BEDIENPANEL · ESP32

OLED mit Drehgeber

1,3-Zoll-OLED, EC11-Drehgeber und zwei Tasten auf einer Platine. Erste Bedienung in rund 25 Minuten – mit dem ESP32 DevKit.



AUFLÖSUNG 128 × 64 Pixel	CONTROLLER SH1106
BEDIENUNG EC11 + 2 Tasten	LOGIK 3,3 V

01
Verkabeln
9 Leitungen zum ESP32

02
Einrichten
IDE, Board-Paket, Bibliothek

03
Hochladen
Beispiel-Sketch

04
Bedienen
Drehen und drücken

01 Auf einen Blick

Anzeige und komplette Bedienung auf einer Platine: drehen zum Navigieren, drücken zum Auswählen, dazu eigene Tasten für BACK und CONFIRM. Das Modul ist für 3,3-V-Logik gebaut – deshalb steuern wir es mit dem ESP32.

Technische Daten

Display	**OLED, weiß, 1,3 Zoll**
Auflösung	**128 × 64 Pixel**
Controller	**SH1106**
Schnittstelle	**I2C, Adresse 0x3C**
Drehgeber	**EC11 mit Druckfunktion**
Tasten	**BACK und CONFIRM, aktiv LOW**
Versorgung	**3,3 V DC**

Lieferumfang

1 × **1,3" OLED-Display mit EC11-Drehgeber**

Nur 3,3 V
Das Modul ist für 3,3-V-Logik ausgelegt. Am UNO bräuchtest du Pegelwandler in allen Leitungen – am ESP32 passt alles direkt.

SH1106-Treiber
Wie beim 1,3-Zoll-OLED: Adafruit SH110X statt SSD1306, sonst ist das Bild verschoben.

Pinbelegung

PIN	FUNKTION	SPANNUNG	HINWEIS
CON	Taste CONFIRM	aktiv LOW	Rechts unten auf der Platine.
SDA	I2C Daten	3,3 V	ESP32 GPIO21.
SCL	I2C Takt	3,3 V	ESP32 GPIO22.
PSH	Druck auf den Knopf	aktiv LOW	Taster in der Achse.
TRA	Encoder A	3,3 V	Quadratursignal A (CLK).
TRB	Encoder B	3,3 V	Signal B (DT) – ergibt die Richtung.
BAK	Taste BACK	aktiv LOW	Rechts oben auf der Platine.
GND	Masse	0 V	Gemeinsame Masse.
VCC	Versorgung	3,3 V DC	An den 3V3-Pin des ESP32.

Reihenfolge laut Aufdruck: CON · SDA · SCL · PSH · TRA · TRB · BAK · GND · VCC.

Was du zusätzlich brauchst

ESP32 DevKit USB-C
WROOM-32, CH340C – 3,3-V-Logik
MKR-ESP32-005

Breadboard + Jumper-Set
400 Kontakte, 40 Dupont-Kabel
MKR-SET-BB400-JUMPER

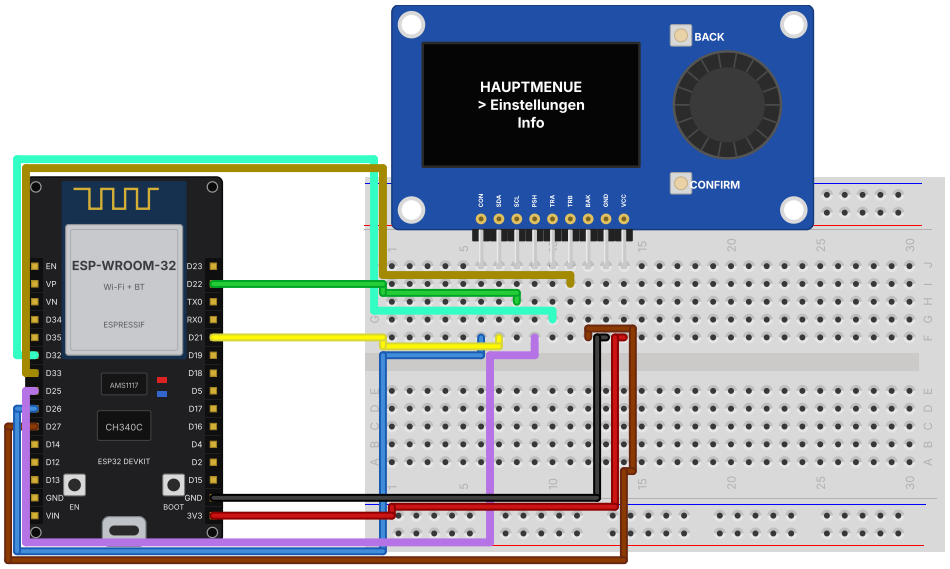
USB-C-Datenkabel
Kein reines Ladekabel – es muss Daten übertragen

02

Verkabeln

Neun Leitungen – zwei für I2C, fünf für Drehgeber und Tasten, zwei für Strom.

STECKPLATINEN-ANSICHT

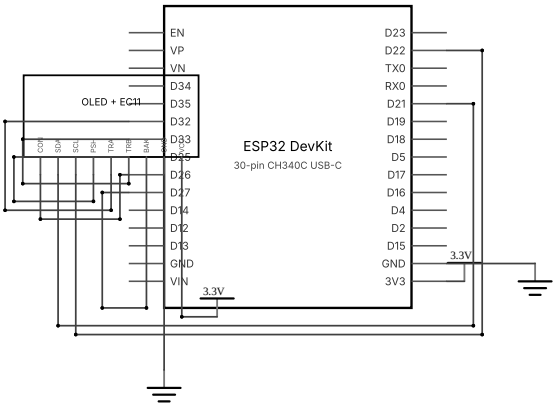


Verbindungen

MODUL		ESP32 DEVKIT	LEITUNG
CON	→	D26	Taste CONFIRM
SDA	→	D21	I2C Daten
SCL	→	D22	I2C Takt
PSH	→	D25	Knopf drücken
TRA	→	D32	Encoder A
TRB	→	D33	Encoder B
BAK	→	D27	Taste BACK
GND	→	GND	Masse
VCC	→	3V3	Versorgung 3,3 V

VCC an 3V3
Niemals an VIN oder 5 V – das Modul ist für 3,3 V gebaut.

SCHALTPLAN



Verdrahtung erstellt mit Fritzing

03 Einrichten & Hochladen

Schritte 1 bis 3 machst du nur ein einziges Mal – danach ist dein ESP32 für alle weiteren Projekte startklar.

1 Arduino IDE installieren

Lade die aktuelle Arduino IDE 2 von `arduino.cc/en/software` herunter und installiere sie.

2 ESP32-Board-Paket installieren

Im Boardverwalter (Board-Symbol links) nach **esp32** suchen und **esp32 von Espressif Systems** installieren. Den CH340C-Treiber bringen Windows 10/11 und macOS meist mit. Ausführlich mit Bildern: unser Guide zum ESP32 DevKit.

3 Board und Port wählen

ESP32 per USB-C anschließen. In der Board-Auswahl **ESP32 Dev Module** wählen und den neuen COM-Port (Windows) bzw. `/dev/ttyUSB0` (Linux) / `/dev/cu.wchusbserial…` (macOS).

4 Bibliothek installieren

Bibliotheksverwalter öffnen (`Strg` + `Umschalt` + `I`), nach **Adafruit SH110X** von Adafruit suchen und installieren. Abhängigkeiten (u. a. *Adafruit GFX Library, Adafruit BusIO*) mit **Alle installieren** bestätigen.

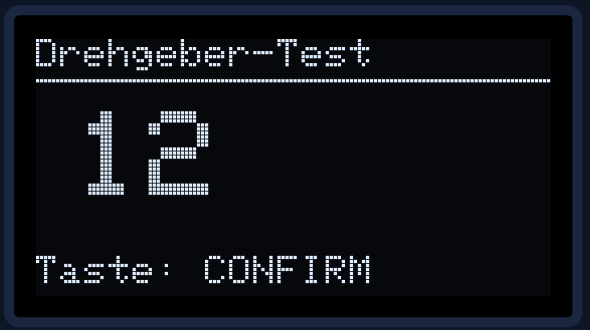
5 Sketch hochladen

Neuen Sketch anlegen, den Code von Seite 5 einfügen und mit dem Pfeil-Symbol hochladen (`Strg` + `U`). Unten erscheint „Hochladen abgeschlossen".

6 Seriellen Monitor öffnen (optional)

`Strg` + `Umschalt` + `M`, unten rechts **115200 Baud**. Bei jeder Raste erscheint der neue Wert, z. B. „Wert: 12".

SO SIEHT ES AUF DEM DISPLAY AUS



Nachbildung der Anzeige – berechnet aus dem Sketch auf Seite 5, Pixel für Pixel.

i Schnelltest

Dreh am Knopf: Die große Zahl zählt hoch oder runter. Die unterste Zeile zeigt die zuletzt gedrückte Taste, BACK setzt den Wert auf 0.

04 Der erste Sketch

Liest Drehgeber und alle drei Tasten aus und zeigt den Wert groß an – die Basis für jedes Menü.

MAKEROO_OLEDENC_Start.ino

```
// MAKEROO · 1,3" OLED SH1106 mit EC11-Drehgeber · Erster Start (ESP32)
#include <Wire.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SH110X.h>

const int TRA = 32, TRB = 33;                  // Drehgeber A und B
const int PSH = 25, CON = 26, BAK = 27;        // Druck, CONFIRM, BACK
Adafruit_SH1106G oled(128, 64, &Wire, -1);
int wert = 0;                                  // wird per Drehen veraendert
int letzterA = HIGH;
const char* taste = "-";                       // zuletzt gedrueckte Taste

void setup() {
  Serial.begin(115200);
  int pins[] = {TRA, TRB, PSH, CON, BAK};
  for (int p : pins) pinMode(p, INPUT_PULLUP);
  Wire.begin(21, 22);                          // SDA, SCL
  if (!oled.begin(0x3C, true)) {
    Serial.println("Display nicht gefunden!");
    while (true) delay(10);
  }
}

void loop() {
  int a = digitalRead(TRA);
  if (a != letzterA && a == LOW) {             // eine Raste weiter
    if (digitalRead(TRB) == HIGH) wert++; else wert--;
    Serial.printf("Wert: %d\n", wert);
  }
  letzterA = a;
  if (digitalRead(PSH) == LOW) taste = "PUSH";
  if (digitalRead(CON) == LOW) taste = "CONFIRM";
  if (digitalRead(BAK) == LOW) { taste = "BACK"; wert = 0; }

  oled.clearDisplay();
  oled.setTextColor(SH110X_WHITE);
  oled.setTextSize(1);
  oled.setCursor(0, 0);
  oled.print("Drehgeber-Test");
  oled.drawLine(0, 10, 127, 10, SH110X_WHITE);
  oled.setTextSize(3);                         // 18 x 24 Pixel je Zeichen
  oled.setCursor(10, 18);
  oled.print(wert);
  oled.setTextSize(1);
  oled.setCursor(0, 54);
  oled.print("Taste: ");
  oled.print(taste);
  oled.display();
}
```

Zeile 6-7
Pins
Drehgeber an D32/D33, Tasten an D25, D26, D27.

Zeile 16
Pull-ups
Alle fünf Eingänge bekommen den internen Pull-up.

Zeile 26-29
Drehrichtung
Fällt A auf LOW, verrät B die Richtung – eine Raste, ein Schritt.

Zeile 33
BACK
Merkt die Taste und setzt den Wert zurück.

Zeile 41-43
Großer Wert
Größe 3: 18 × 24 Pixel je Ziffer.

Zeile 48
Anzeigen
Überträgt das fertige Bild in einem Rutsch.

Geprüft: kompiliert für ESP32 Dev Module (Board-Paket esp32 3.3.12) mit Adafruit SH110X 2.1.15 (300.1 kB Flash = 23 %, 23 kB RAM).

05 Wenn es nicht klappt

SYMPTOM	LÖSUNG
„Display nicht gefunden!“	SDA an D21, SCL an D22? VCC an 3V3? Mit dem I2C-Scanner aus dem ESP32-Guide nach 0x3C suchen.
Wert zählt falsch herum	TRA und TRB tauschen – oder im Sketch `wert++` und `wert--` vertauschen.
Wert springt um 2 oder unregelmäßig	Langsamer drehen und prüfen, ob GND fest steckt; bei Bedarf A/B je 100 nF nach GND.
Bild um 2 Pixel verschoben	SSD1306-Bibliothek erwischt – SH1106G verwenden.
Upload bricht ab	Pins D2, D12, D15 frei lassen; BOOT-Taste beim „Connecting…“ halten.

06 Wie geht's weiter?

Mit ESP32 & ESPHome

Gleiche Verdrahtung wie hier – Drehgeber und Display direkt in ESPHome:

```
i2c:
  sda: GPIO21
  scl: GPIO22
display:
  - platform: ssd1306_i2c
    model: "SH1106 128x64"
    address: 0x3C
sensor:
  - platform: rotary_encoder
    name: "Drehgeber"
    pin_a: { number: GPIO32, mode: INPUT_PULLUP }
    pin_b: { number: GPIO33, mode: INPUT_PULLUP }
```

Projektideen

Menü-Steuerung
Einträge wählen, Werte einstellen

Lautstärkeregler
Drehen statt Tippen

Smart-Home-Panel
Szenen per Knopf in Home Assistant

Reflow-/Sous-Vide-Regler
Solltemperatur einstellen

Passt dazu

ESP32 DevKit USB-C
WROOM-32, CH340C – 3,3-V-Logik
`MKR-ESP32-005`

ESP32 Gehäuse aus Acryl
Schützt das Board
`MKR-ESP32-HOUSING-045`

Relais KY-019
Verbraucher schalten
`MKR-KY019-034`

BMP280 Luftdruck & Temperatur
Messwerte zum Anzeigen
`MKR-GY-BMP280-024`

Sicherheitshinweise. Elektronikmodul für Ausbildungs-, Entwicklungs- und Experimentierzwecke. Nur mit Kleinspannung betreiben, niemals an Netzspannung. Nicht geeignet für Kinder unter 14 Jahren; Nutzung nur unter Aufsicht. Elektroaltgeräte gehören nicht in den Hausmüll – Rückgabe kostenlos bei kommunalen Sammelstellen.

Verantwortlicher Wirtschaftsakteur (GPSR): Alexander Prez · makeroo · Hinterm Bahnhof 4 · 90513 Zirndorf · support@makeroo.de
Fragen? Schreib uns mit einem Foto deiner Verkabelung.
Arduino ist eine Marke der Arduino SA und dient nur der Kompatibilitätsangabe.