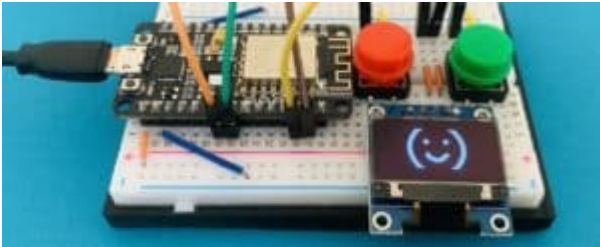


Ein OLED-Display (Adafruit SSD1306) am Arduino oder ESP8266 anschließen



Willst Du Deinem Arduino-Projekt einen Hauch von Eleganz verleihen? Wie wäre es mit einem coolen OLED-Display? **Sie sind flach, leicht, energieeffizient und bieten ein gestochen scharfes Bild.** In diesem Blog-Artikel zeigen wir Dir, wie Du ein OLED-Display (Adafruit SSD1306) an Deinen Arduino anschließt und verwendest. Also, lass uns direkt loslegen!

Benötigte Materialien:

1. Arduino (z.B. Arduino Uno, Mega oder Nano)
2. OLED-Display (z.B. 128×64 Pixel, I2C-Kommunikation)
3. Jumper-Kabel (männlich-männlich)
4. Breadboard (optional)

Schritt 1: Verkabelung

Bevor wir mit der Programmierung beginnen, müssen wir das Display mit dem Arduino verbinden. Hier ist, wie Du es verkabelst:

1. Schließe GND (Ground) des OLED-Displays an den GND-Pin des Arduino an.
2. Verbinde VCC (Power) des Displays mit dem 5V-Pin des Arduino.

3. Schließe SDA (Daten) des Displays an den A4-Pin (SDA) des Arduino Uno oder Mega, bzw. an den A4-Pin (SDA) des Arduino Nano an.
4. Verbinde SCL (Clock) des Displays mit dem A5-Pin (SCL) des Arduino Uno oder Mega, bzw. an den A5-Pin (SCL) des Arduino Nano.

Wenn Du ein Breadboard verwendest, stecke das -Display auf das Breadboard und verbinde es mit dem Arduino über Jumper-Kabel.

Schritt 2: Bibliotheken installieren

Um das OLED-Display mit dem Arduino zu verwenden, **benötigen wir zwei Bibliotheken**: die Adafruit-GFX-Bibliothek und die Adafruit-SSD1306-Bibliothek. Du kannst sie direkt aus der Arduino-IDE heraus installieren:

1. Öffne die Arduino-IDE und gehe zu „Sketch“ > „Bibliothek einbinden“ > „Bibliotheken verwalten...“.
2. Suche nach „Adafruit GFX Library“ und installiere die neueste Version.
3. Suche anschließend nach „Adafruit SSD1306“ und installiere auch hier die neueste Version.

Schritt 3: Beispielcode laden und anpassen

Jetzt, wo die Verkabelung und die Bibliotheken bereit sind, können wir den Beispielcode laden und anpassen:

1. Öffne die Arduino-IDE und gehe zu „Datei“ > „Beispiele“ > „Adafruit SSD1306“ > „ssd1306_128x64_i2c“.
2. Suche im Code nach der Zeile `#define OLED_RESET 4` und

ändere sie in `#define OLED_RESET -1`, um den Reset-Pin zu deaktivieren (wir benötigen ihn nicht).

3. Suche im Code nach der Zeile `Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire, OLED_RESET);` und ändere sie in `Adafruit_SSD1306 display(128, 64, &Wire, -1);` für ein 128×64 Pixel OLED-Display.

Schritt 4: Code hochladen und Display testen

Es ist Zeit, den angepassten Beispielcode auf Deinen Arduino zu laden und das OLED-Display zu testen:

1. Stelle sicher, dass Dein Arduino über USB mit Deinem Computer verbunden ist.
2. Wähle in der Arduino-IDE unter „Werkzeuge“ > „Board“ das richtige Arduino-Board aus (z.B. Arduino Uno, Mega oder Nano).
3. Wähle unter „Werkzeuge“ > „Port“ den richtigen USB-Anschluss für Deinen Arduino aus.
4. Klicke auf den „Hochladen“-Button (Pfeilsymbol) in der Arduino-IDE, um den Code auf Deinen Arduino zu übertragen.

Sobald der Code erfolgreich hochgeladen wurde, sollte das Display verschiedene Muster und Text anzeigen. Herzlichen Glückwunsch, Du hast erfolgreich ein [OLED](#)-Display an Deinen Arduino angeschlossen und verwendet!

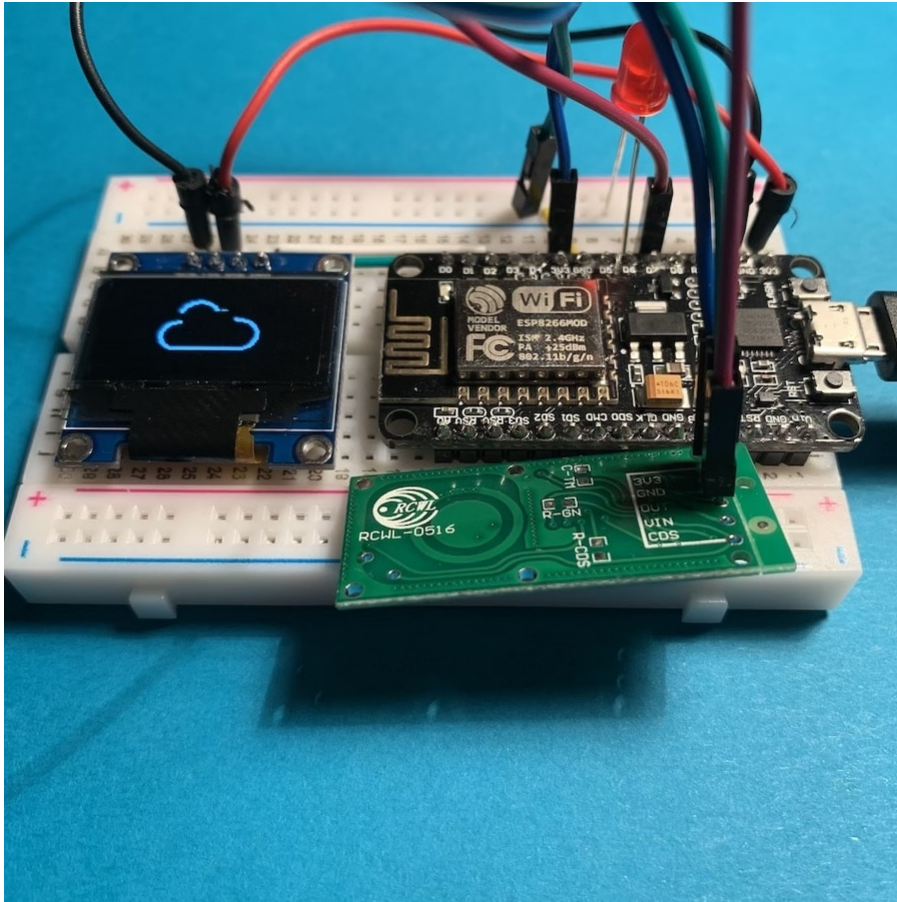
Schritt 5: Dein eigenes Display-Design erstellen

Nun, da Du weißt, wie Du ein OLED-Display anschließt und den Beispielcode verwendest, kannst Du Deine eigenen Designs und

Texte erstellen! Hier sind einige grundlegende Funktionen, die Du nutzen kannst:

- `display.clearDisplay();` – Löscht den Display-Inhalt.
- `display.setTextSize(1);` – Setzt die Textgröße (1, 2, 3...).
- `display.setTextColor(WHITE);` – Setzt die Textfarbe (OLED-Displays sind normalerweise monochrom, daher ist Weiß die Standardfarbe).
- `display.setCursor(0, 0);` – Setzt den Cursor an die angegebene Position (x, y).
- `display.println("Hallo, Maker!");` – Gibt Text auf dem Display aus.
- `display.display();` – Aktualisiert das Display, um die vorgenommenen Änderungen anzuzeigen.

Fühle Dich frei, mit diesen Funktionen zu experimentieren und Dein eigenes Display-Design zu erstellen. Denke daran, den Befehl `display.display();` nach Deinen Anweisungen zu verwenden, um die Änderungen auf dem Display sichtbar zu machen. Übrigens, in diesem Tutorial lernst du, wie du [Bilder auf dem OLED-Display anzeigst](#).



Der Anschluss am ESP8266

Der Anschluss eines OLED-Displays an einen ESP8266 ist ebenso einfach wie beim Arduino. Hier ist, wie Du es verkabelst:

Benötigte Materialien:

1. ESP8266 (z.B. NodeMCU, Wemos D1 Mini)
2. OLED-Display (z.B. 128×64 Pixel, I2C-Kommunikation)
3. Jumper-Kabel (männlich-männlich)
4. Breadboard (optional)

Schritt 1: Verkabelung und Code

Um das OLED-Display mit dem ESP8266 zu verbinden, führe die folgenden Schritte aus:

1. Schließe GND (Ground) des OLED-Displays an den GND-Pin des ESP8266 an.
2. Verbinde VCC (Power) des Displays mit dem 3.3V-Pin des ESP8266.
3. Schließe SDA (Daten) des Displays an den D2-Pin (GPIO4, SDA) des ESP8266 an.
4. Verbinde SCL (Clock) des Displays mit dem D1-Pin (GPIO5, SCL) des ESP8266.

Wenn Du ein Breadboard verwendest, stecke das OLED-Display auf das Breadboard und verbinde es mit dem ESP8266 über Jumper-Kabel.

Die restlichen Schritte zur Installation der Bibliotheken und zum Anpassen des Beispielcodes bleiben dieselben. Du musst lediglich eine kleine Änderung am Beispielcode vornehmen, um die Pin-Belegung für den ESP8266 festzulegen:

1. Füge im Code die Zeile `#include <Wire.h>` hinzu, um die Wire-Bibliothek einzubinden.
2. Füge direkt nach der Zeile `Wire.begin();` die folgenden Zeilen hinzu, um die I2C-Pins für den ESP8266 festzulegen:

```
#if defined(ESP8266)
  Wire.begin(4, 5); // SDA, SCL
#endif
```

Das war's! Jetzt kannst Du den angepassten Code auf Deinen ESP8266 hochladen und das Display wie zuvor beschrieben verwenden.