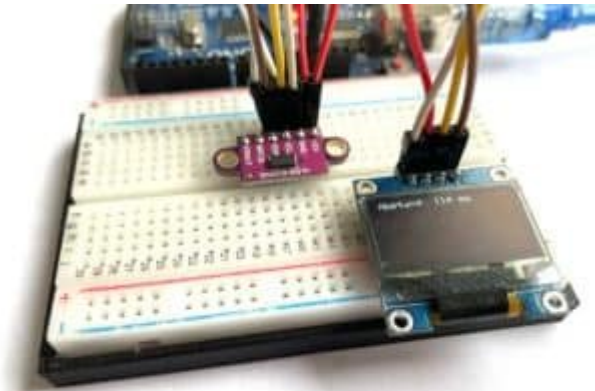


So misst du Entfernungen mit dem VL53L0X-Sensor und dem Arduino UNO



In diesem Blogartikel möchten wir Dir zeigen, wie Du ganz einfach den VL53L0X-Sensor mit Deinem Arduino UNO verbindest und verwendest. Mit diesem faszinierenden Sensor kannst Du präzise Entfernungen messen, und das mit **einer Reichweite von bis zu 2 Metern**.

Was ist der VL53L0X-Sensor?

Der VL53L0X ist ein **Time-of-Flight (ToF)-Laserabstandssensor**, der mithilfe von Infrarot-Licht die Entfernung zu einem Objekt misst. Er ist kompakt und leicht zu integrieren, was ihn ideal für die Anwendung in Roboterprojekten, Drohnen oder zur Objekterkennung macht.

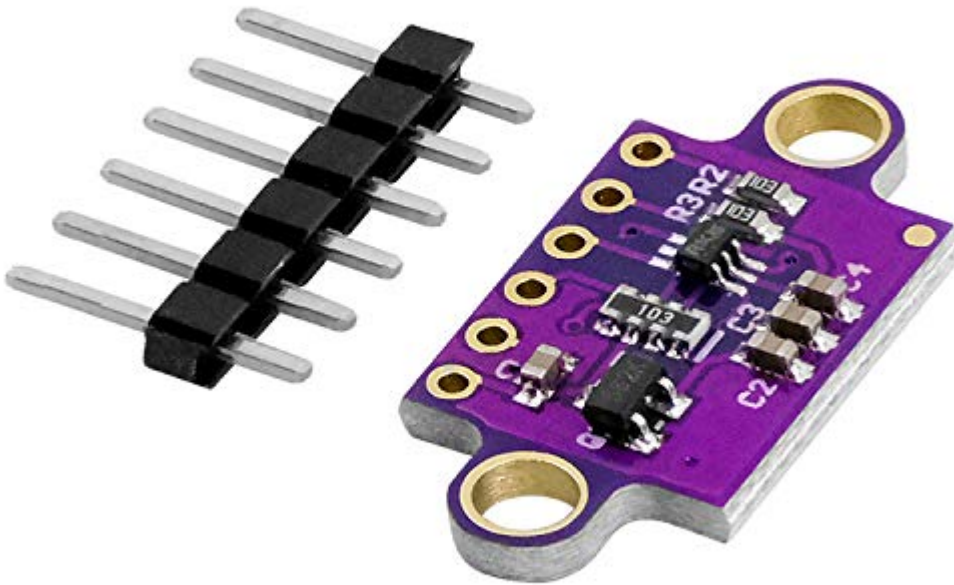
Was benötigst Du für dieses Projekt?

Neben dem Sensor benötigst du nur einen Microcontroller – in diesem Tutorial verwenden wir den Arduino UNO. Selbstverständlich kannst du auch jedes andere verwenden, sofern es über I²C verfügt.



[AZDelivery 1 x Mikrocontroller Board AZ- ATmega328 - Board | Der Mikrocontroller ist kompatibel mit Arduino IDE und auch mit dem Arduino UNO | Das AZ Delivery Board mit USB-Kabel inklusive E-Book!](#)

10,99 €



[AZDelivery VL53L0X Time of Flight Sensor 940nm kompatibel mit Arduino inklusive E-Book!](#)

□ Schnelle Reaktionszeit: 50 ms.

7,49 €

Verkabelung des Sensors mit dem Arduino UN0

Bevor wir anfangen, den Sensor zu programmieren, müssen wir ihn erst einmal mit dem Arduino UN0 verbinden. Die Verkabelung ist denkbar einfach:

1. Verbinde den VCC (oder auch VIN)-Pin des VL53L0X mit dem 5V-Pin des Arduino UN0.
2. Verbinde den GND-Pin des Sensors mit dem GND-Pin des Arduino UN0.

3. Verbinde den SDA-Pin mit dem A4-Pin (SDA) des Arduino UNO.
4. Verbinde den SCL-Pin mit dem A5-Pin (SCL) des Arduino UNO.

Programmierung des Arduino UNO

Um den VL53L0X-Sensor verwenden zu können, musst Du zunächst die passende Bibliothek in Deiner Arduino IDE installieren. Hierzu gehst Du wie folgt vor:

1. Öffne die Arduino IDE auf Deinem Computer.
2. Klicke im Menü auf „Sketch“ > „Bibliothek einbinden“ > „Bibliotheken verwalten...“.
3. Suche im Bibliotheksmanager nach „VL53L0X“ und installiere die Bibliothek „VL53L0X von Pololu“.

Jetzt können wir mit dem Programmieren beginnen! Erstelle einen neuen Sketch und füge den folgenden Code ein:

```
#include <Wire.h>
#include <VL53L0X.h>
```

```
VL53L0X sensor;
```

```
void setup() {
  Serial.begin(9600);
  Wire.begin();

  sensor.init();
  sensor.setTimeout(500);

  Serial.println("VL53L0X-Sensor bereit!");
}
```

```
void loop() {
  int distance = sensor.readRangeSingleMillimeters();
```

```
    if (sensor.timeoutOccurred()) {  
        Serial.print("Timeout! ");  
    }  
  
    Serial.print("Abstand: ");  
    Serial.print(distance);  
    Serial.println(" mm");  
  
    delay(1000);  
}
```

Lade den Sketch auf Deinen Arduino UNO hoch und öffne den seriellen Monitor. Du solltest nun alle Sekunden die gemessene Entfernung in Mill

imetern angezeigt bekommen.

Wie funktioniert der Code?

Lass uns einen Blick auf die einzelnen Bestandteile des Codes werfen, um besser zu verstehen, wie das Programm funktioniert: Zuerst binden wir die benötigten Bibliotheken ein, nämlich Wire.h für die I2C-Kommunikation und VL53L0X.h für den Sensor.

```
#include <Wire.h>  
#include <VL53L0X.h>
```

Danach erstellen wir ein Objekt sensor der Klasse VL53L0X.

```
VL53L0X sensor;
```

Im setup()-Funktionsteil initialisieren wir die serielle Kommunikation mit einer Baudrate von 9600, die [I2C-Kommunikation](#) und den Sensor. Wir setzen außerdem ein Timeout von 500 ms für den Sensor, damit das Programm nicht ewig hängt, falls es Probleme bei der Messung geben sollte.

```
void setup() {  
  Serial.begin(9600);  
  Wire.begin();  
  
  sensor.init();  
  sensor.setTimeout(500);  
  
  Serial.println("VL53L0X-Sensor bereit!");  
}
```

In der loop()-Funktion lesen wir die Entfernung in Millimetern mit sensor.readRangeSingleMillimeters() und speichern den Wert in der Variable distance. Wir überprüfen, ob ein Timeout aufgetreten ist, und geben eine entsprechende Meldung aus. Anschließend geben wir den gemessenen Abstand auf dem seriellen Monitor aus und warten eine Sekunde, bevor wir die nächste Messung durchführen.

```
void loop() {  
  int distance = sensor.readRangeSingleMillimeters();  
  
  if (sensor.timeoutOccurred()) {  
    Serial.print("Timeout! ");  
  }  
  
  Serial.print("Abstand: ");  
  Serial.print(distance);  
  Serial.println(" mm");  
  
  delay(1000);  
}
```

Jetzt weißt Du, wie Du den Time-of-Flight-Sensor mit Deinem Arduino UNO verbinden und verwenden kannst, um präzise Entfernungsmessungen durchzuführen. Mit dieser neuen Fähigkeit kannst Du nun kreative Projekte entwickeln, bei denen Entfernungen eine wichtige Rolle spielen, wie beispielsweise

Parkassistenten, Hinderniserkennung für Roboter oder sogar interaktive Kunstinstallationen.

Die Entfernung auf einem OLED-Display anzeigen

Um das Projekt zu erweitern und die gemessene Entfernung **auf einem OLED-Display anzuzeigen**, benötigst Du zusätzlich:

- Ein I2C-OLED-Display (128×64 Pixel, z.B. SSD1306)
- Ein paar zusätzliche Jumper-Kabel

Verkabelung des OLED-Displays mit dem Arduino UNO

Die Verkabelung des OLED-Displays ist ähnlich einfach wie beim VL53L0X-Sensor:

1. Verbinde den VCC-Pin des OLED-Displays mit dem 5V-Pin des Arduino UNO.
2. Verbinde den GND-Pin des OLED-Displays mit dem GND-Pin des Arduino UNO.
3. Verbinde den SDA-Pin des OLED-Displays mit dem A4-Pin (SDA) des Arduino UNO.
4. Verbinde den SCL-Pin des OLED-Displays mit dem A5-Pin (SCL) des Arduino UNO.

Da das OLED-Display und der VL53L0X-Sensor beide I2C-Geräte sind, können sie gemeinsam an den I2C-Bus des Arduino UNO angeschlossen werden.

Programmierung des Arduino UNO

Um das OLED-Display zu verwenden, musst Du zunächst die passenden Bibliotheken in Deiner Arduino IDE installieren. Hierzu gehst Du wie folgt vor:

1. Öffne die Arduino IDE auf Deinem Computer.
2. Klicke im Menü auf „Sketch“ > „Bibliothek einbinden“ > „Bibliotheken verwalten...“.
3. Suche im Bibliotheksmanager nach „Adafruit SSD1306“ und installiere die Bibliothek „Adafruit SSD1306 by Adafruit“.
4. Suche im Bibliotheksmanager nach „Adafruit GFX“ und installiere die Bibliothek „Adafruit GFX Library by Adafruit“.

Jetzt können wir den Code erweitern, um das OLED-Display zu nutzen. **Erstelle einen neuen Sketch oder bearbeite den bestehenden und füge den folgenden Code ein:**

```
#include <Wire.h>
#include <VL53L0X.h>
#include <Adafruit_SSD1306.h>
#include <Adafruit_GFX.h>
```

```
#define SCREEN_WIDTH 128
#define SCREEN_HEIGHT 64
```

```
VL53L0X sensor;
Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire);
```

```
void setup() {
  Serial.begin(9600);
  Wire.begin();

  sensor.init();
  sensor.setTimeout(500);
```

```
display.begin(SSD1306_SWITCHCAPVCC, 0x3C);
```



```

display.clearDisplay();
display.setTextSize(1);
display.setTextColor(WHITE);
display.setCursor(0, 0);

Serial.println("VL53L0X-Sensor bereit!");
}

void loop() {
  int distance = sensor.readRangeSingleMillimeters();

  if (sensor.timeoutOccurred()) {
    Serial.print("Timeout! ");
  }

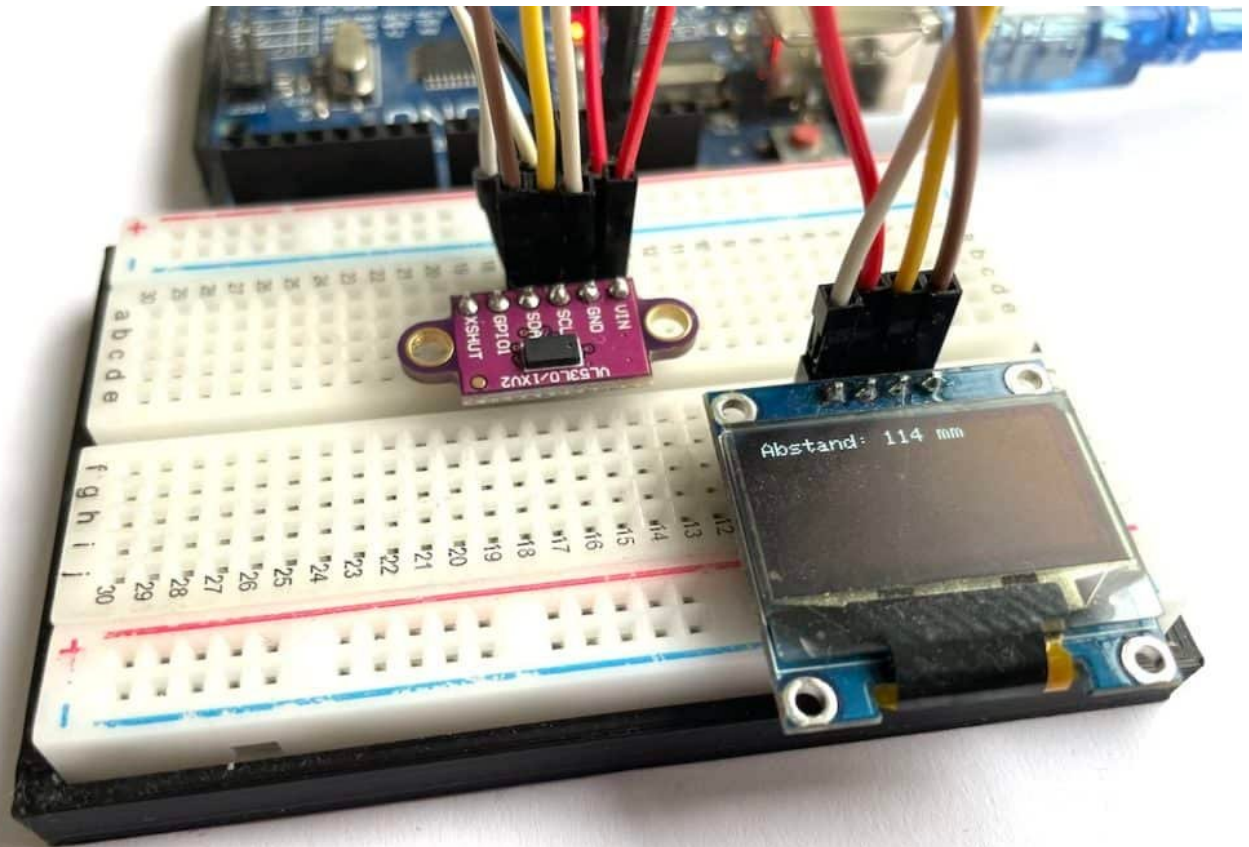
  display.clearDisplay();
  display.setCursor(0, 0);
  display.print("Abstand: ");
  display.print(distance);
  display.print(" mm");
  display.display();

  Serial.print("Abstand: ");
  Serial.print(distance);
  Serial.println(" mm");

  delay(1000);
}

```

Lade den Sketch auf Deinen Arduino UNO hoch. Die gemessene Entfernung sollte nun auf dem OLED-Display angezeigt werden.



Möchtest du die Entfernung lieber mit Ultraschall messen? [Bei uns erfährst du auch, wie das geht.](#)