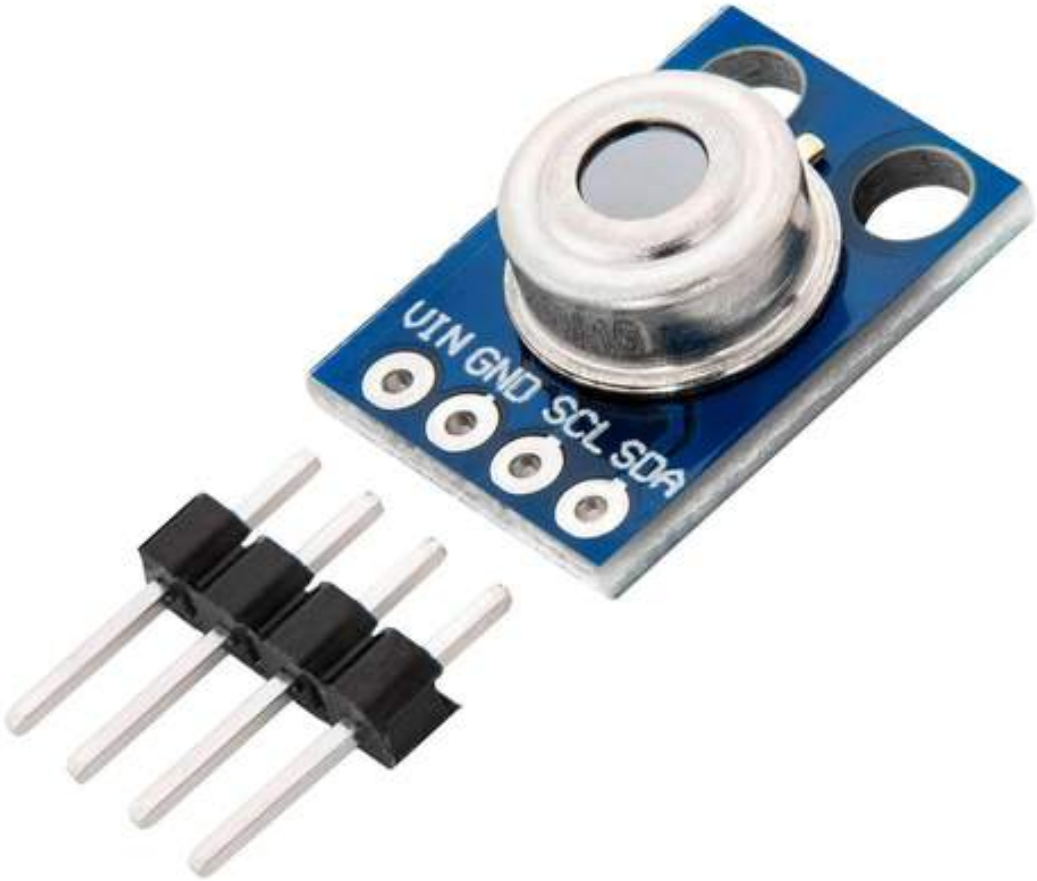
Willkommen!

Vielen Dank, dass Sie sich für unser *GY-906 IR-Temperaturmodul von AZ-Delivery* entschieden haben. In den nachfolgenden Seiten werden wir Ihnen erklären wie Sie das Gerät einrichten und nutzen können.

Viel Spaß!



Inhaltsverzeichnis

Einführung

Im Herzen des GY-906 Moduls befindet sich der Infrarotsensor MLX90614. Es handelt sich dabei um ein Infrarot-Thermometer für berührungslose Temperaturmessungen. Sowohl der IR-empfindliche Thermosäule-Detektorchip als auch die Signalkonditionierungs-ASIC sind im gleichen *TO-39*-Gehäuse integriert. Ein rauscharmer Verstärker, ein 17-Bit-ADC und eine leistungsstarke DSP-Einheit, die in den MLX90164 integriert sind, sorgen für eine hohe Genauigkeit und Auflösung des Thermometers.

Ein IR-Thermometer ist ein Gerät, das die Temperatur mit der von einem Objekt emittierten Infrarotstrahlung misst. In jedem Material, dessen Temperatur über dem absoluten Nullpunkt (0K, -273,13°C, -459,67°F) liegt, bewegen sich Moleküle in seinem Inneren. Wenn die Temperatur höher ist, bewegen sich die Moleküle schneller. Diese Moleküle geben Infrarotstrahlung ab, die tatsächlich die Temperatur ist, und je heißer sie werden, desto mehr Strahlung geben sie ab. IR-Thermometer erfassen und messen diese Strahlung.

Infrarot-Thermometer verwenden eine Linse, um Infrarotlicht von einem Objekt auf einen als Thermosäule bezeichneten Detektor zu fokussieren. Die Funktion der Thermosäule besteht darin, die Infrarotstrahlung zu absorbieren und in Wärme umzuwandeln. Sie wird immer heißer, da sie immer mehr Infrarotenergie absorbiert. Die überschüssige Wärme wird in Elektrizität umgewandelt. Diese Elektrizität wird an einen Detektor übertragen, der die Temperatur des Objekts bestimmt.

Das heißt, jedes Objekt strahlt in Abhängigkeit von seiner Wärme infrarotes Licht aus, und dieses Licht wird mit einer Thermosäule erfasst, die immer heißer wird und gleichzeitig die überschüssige Wärme in Elektrizität umwandelt.

Der Sensor misst Infrarotlicht, das von den Objekten ausgestrahlt wird, so dass er die Temperatur erfassen kann, ohne die Objekte physisch berühren zu müssen. Richten Sie den Sensor einfach auf das Objekt, dessen Temperatur Sie bestimmen möchten, und er erfasst die Temperatur, indem er das emittierte Infrarotlicht absorbiert.

Hinweis: Bitte halten Sie bei jeder Messung einen Messabstand von 10 mm zwischen dem Modul und dem Objekt ein.

Technische Daten

» Betriebsspannungsbereich:	3.3V bis 5V DC
» Max. Strom	2mA
» Kommunikations-Protokoll:	I2C
» Temperaturbereich der Umgebung:	-40°C to 125°C [-40 to 257°F]
» Temperaturbereich von Objekten:	-70 to 380°C [-94 to 716°F]
» Genauigkeit:	±0.5°C für Obj. und Umgebung
» Dimensionen:	11 x 17 x 6mm [0.4 x 0.7 x 0.24in]
» Sichtfeld:	90 Grad

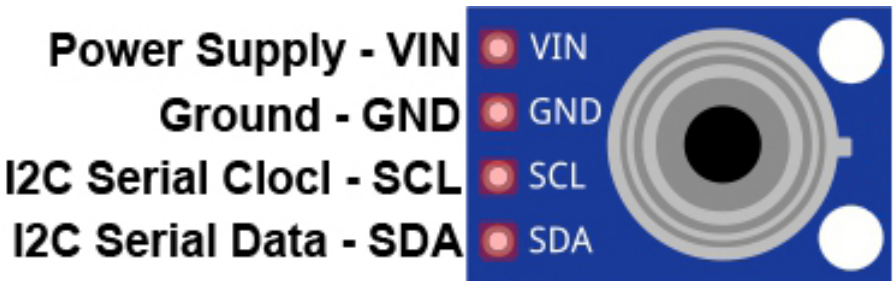
Das Modul kann einen breiteren Temperaturbereich als die meisten digitalen Sensoren erfassen (von *-70*°C bis *+380*°C), da es das Objekt, dessen Temperatur es misst, nicht berühren muss. Halten Sie bei der Temperaturmessung einen Abstand von 10 mm ein, da sich die Messungen stark unterscheiden können, wenn das Modul weiter vom Objekt entfernt ist.

Das Modul unterstützt die serielle Schnittstelle I2C. Das Modul hat eine Standard-I2C-Adresse, die *0x5a* lautet. Das Modul hat eine feste I2C-Adresse, so dass nur ein Modul an der gleichen I2C-Schnittstelle verwendet werden kann. Um mehr Module an der gleichen I2C-Schnittstelle zu verwenden, verwenden Sie den I2C-Multiplexer.

Das Modul kann zur Bestimmung der Durchschnittstemperatur eines Bereichs nützlich sein, da es ein Sichtfeld von 90 Grad misst.

Pinbelegung

Das GY-906 Infrarot-Temperaturmodul hat vier Pins. Die Pinbelegung ist wie folgt:



Die Pins des Moduls können ohne Gefahr für den Sensor selbst an eine 3,3V- oder 5V-Stromversorgung angeschlossen werden. Das Modul hat einen on-Board 3,3V-Spannungsregler.

Wie man die Arduino IDE einrichtet

Falls die Arduino-IDE nicht installiert ist, folgen Sie dem *link* und laden Sie die Installationsdatei für das Betriebssystem Ihrer Wahl herunter.

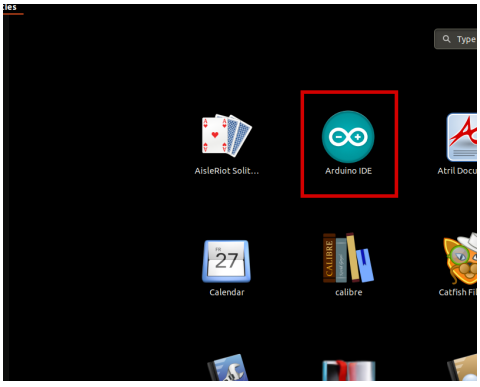
Für `Windows` Benutzer: Doppelklicken Sie auf die heruntergeladene `.exe`-Datei und folgen Sie den Anweisungen im Installationsfenster.

Für *Linux* Benutzer, laden Sie eine Datei mit der Erweiterung *.tar.xz* herunter, die extrahiert werden muss. Wenn sie extrahiert ist, gehen Sie in das extrahierte Verzeichnis und öffnen Sie das Terminal in diesem Verzeichnis. Zwei *.sh* Skripte müssen ausgeführt werden, das erste namens *arduino-linux-setup.sh* und das zweite heißt *install.sh.*

Um das erste Skript im Terminal auszuführen, öffnen Sie das Terminal im extrahierten Ordner und führen Sie den folgenden Befehl aus:
sh arduino-linux-setup.sh user_name
user_name - ist der Name eines Superusers im Linux-Betriebssystem. Ein Passwort für den Superuser muss beim Start des Befehls eingegeben werden. Warten Sie einige Minuten, bis das Skript vollständig abgeschlossen ist.

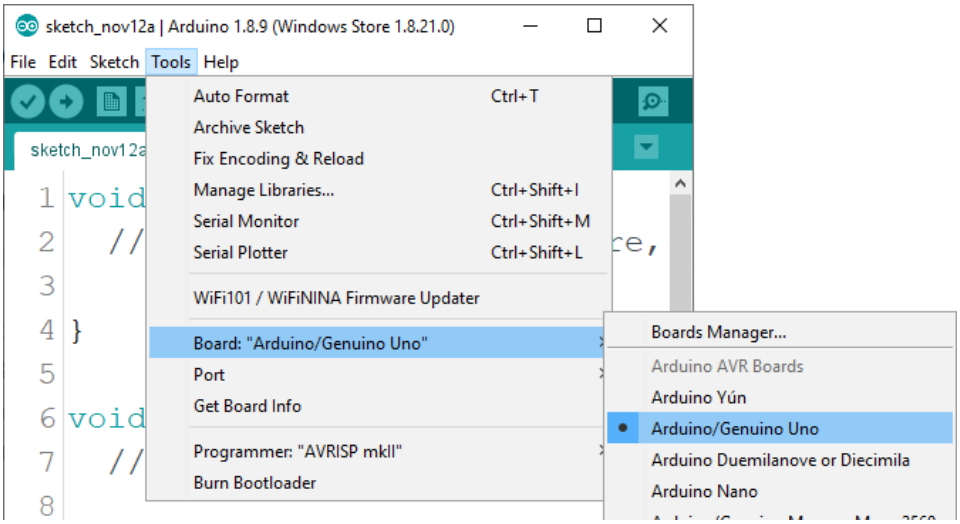
Das zweite Skript mit der Bezeichnung *install.sh*-Skript muss nach der Installation des ersten Skripts verwendet werden. Führen Sie den folgenden Befehl im Terminal (extrahiertes Verzeichnis) aus: **sh install.sh**

Nach der Installation dieser Skripte gehen Sie zu *All Apps*, wo die *Arduino-IDE* installiert ist.



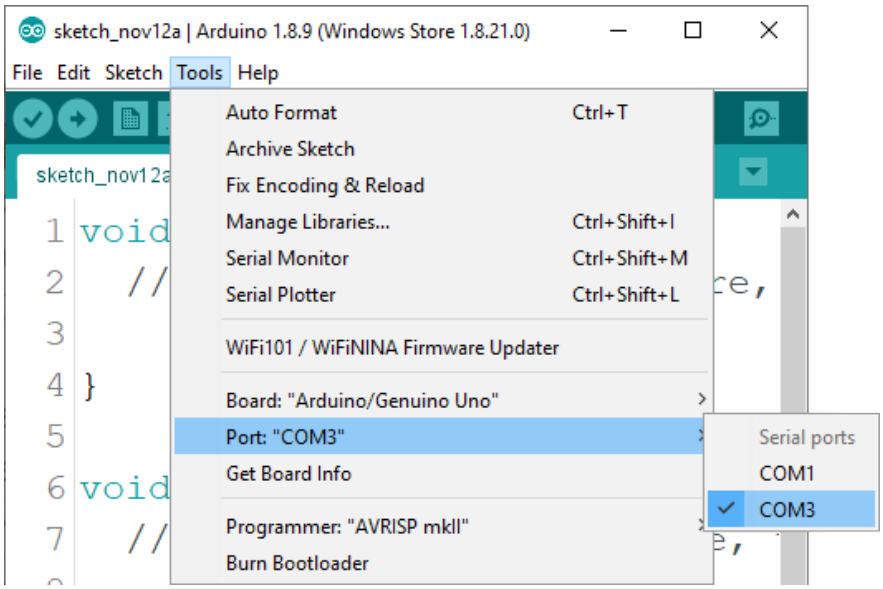
Fast alle Betriebssysteme werden mit einem vorinstallierten Texteditor ausgeliefert (z.B. *Windows* mit *Notepad*, *Linux* Ubuntu mit *Gedit*, *Linux Raspbian* mit *Leafpad* usw.). Alle diese Texteditoren sind für den Zweck des eBooks vollkommen in Ordnung.

Zunächst ist zu prüfen, ob Ihr PC ein Arduino-Board erkennen kann. Öffnen Sie die frisch installierte Arduino-IDE, und gehen Sie zu:
Tools > Board > {your board name here}
{your board name here} sollte der *Arduino/Genuino Uno* sein, wie es auf dem folgenden Bild zu sehen kann:



Der Port, an den das Arduino-Board angeschlossen ist, muss ausgewählt werden. Gehe zu: *Tools > Port > {port name goes here}* und wenn das Arduino-Board an den USB-Port angeschlossen ist, ist der Portname im Drop-down Menü auf dem vorherigen Bild zu sehen.

Wenn die Arduino-IDE unter Windows verwendet wird, lauten die Portnamen wie folgt:



Für *Linux* Benutzer, ist zum Beispiel der Portname */dev/ttyUSBx*, wobei x für eine ganze Zahl zwischen *0* und *9* steht.

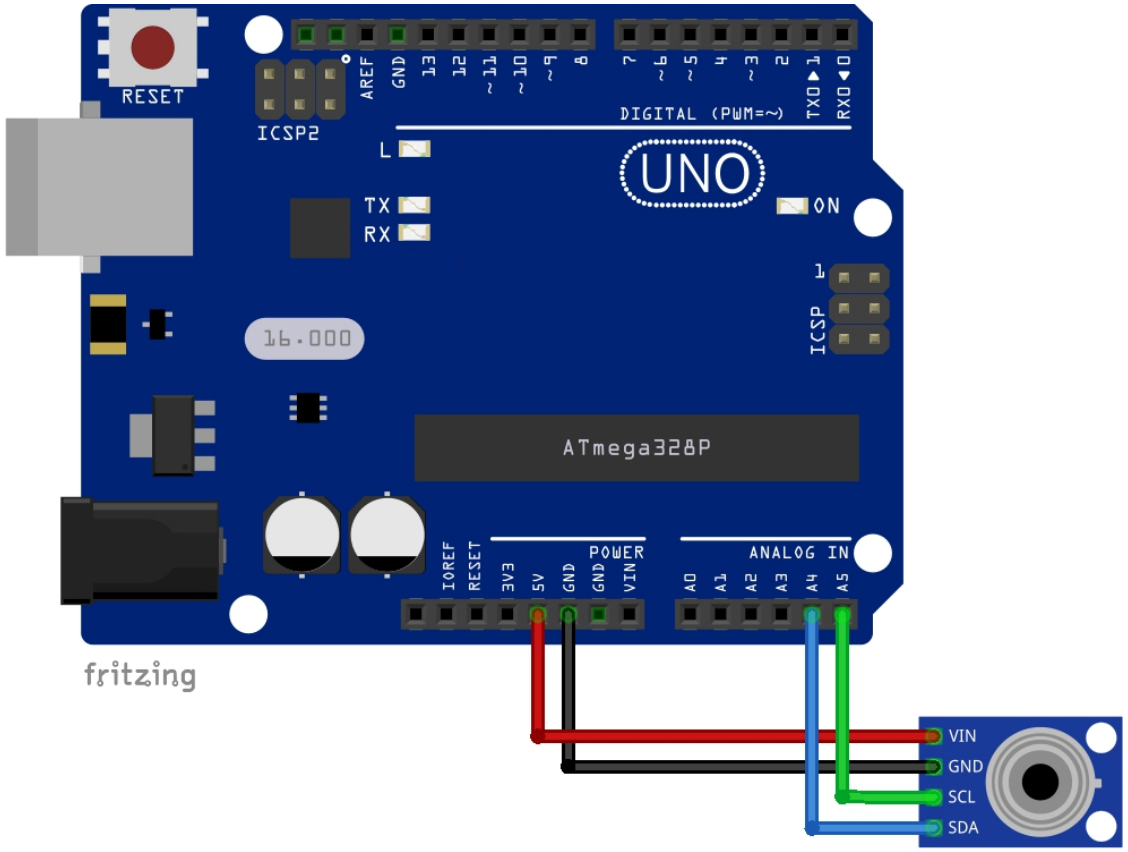
Wie man den Raspberry Pi und Python einrichtet

Für den Raspberry Pi muss zuerst das Betriebssystem installiert werden, dann muss alles so eingerichtet werden, dass es im *Headless*-Modus verwendet werden kann. Der *Headless*-Modus ermöglicht eine Fernverbindung zum Raspberry Pi, ohne dass ein PC-Bildschirm, eine Maus oder eine Tastatur erforderlich ist. Die einzigen Dinge, die in diesem Modus verwendet werden, sind der Raspberry Pi selbst, die Stromversorgung und die Internetverbindung. Das alles wird in dem kostenlosen eBook ausführlich erklärt:
Raspberry Pi Quick Startup Guide

Das Betriebssystem Raspbian wird mit vorinstalliertem *Python* ausgeliefert.

Verbindung des Moduls mit dem Uno

Verbinden Sie das GY-906 Modul mit dem Uno, wie unten abgebildet:

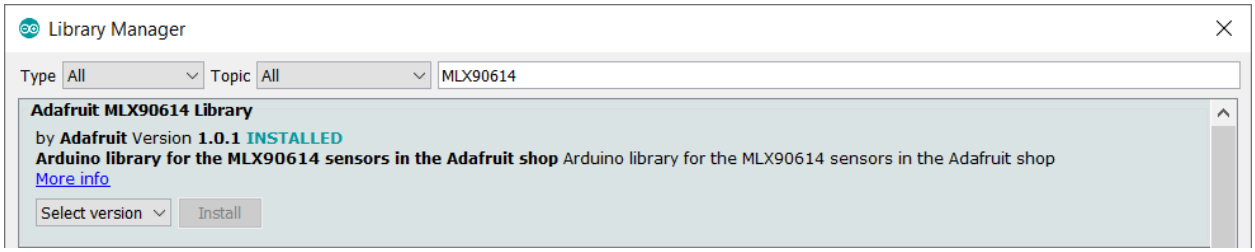


MLX90164 Pin	Uno Pin	Drahtfarbe
VIN	5V	**Roter Draht**
GND	GND	**Schwarzer Draht**
SCL	A5	**Grüner Draht**
SDA	A4	**Blauer Draht**

Library für Arduino IDE

Um das Modul mit der Arduino IDE zu verwenden, wird empfohlen, eine externe Library dafür herunterzuladen. Die in diesem eBook verwendete Library heißt *Adafruit_MLX90614*. Um Sie herunterzuladen und zu installieren, öffnen Sie die Arduino IDE und gehen Sie zu:
Tools > Manage Libraries

Wenn sich ein neues Fenster öffnet, geben Sie *MLX90614* in das Suchfeld ein und installieren die *Adafruit MLX90164 Library* von *Adafruit*, wie unten abgebildet:



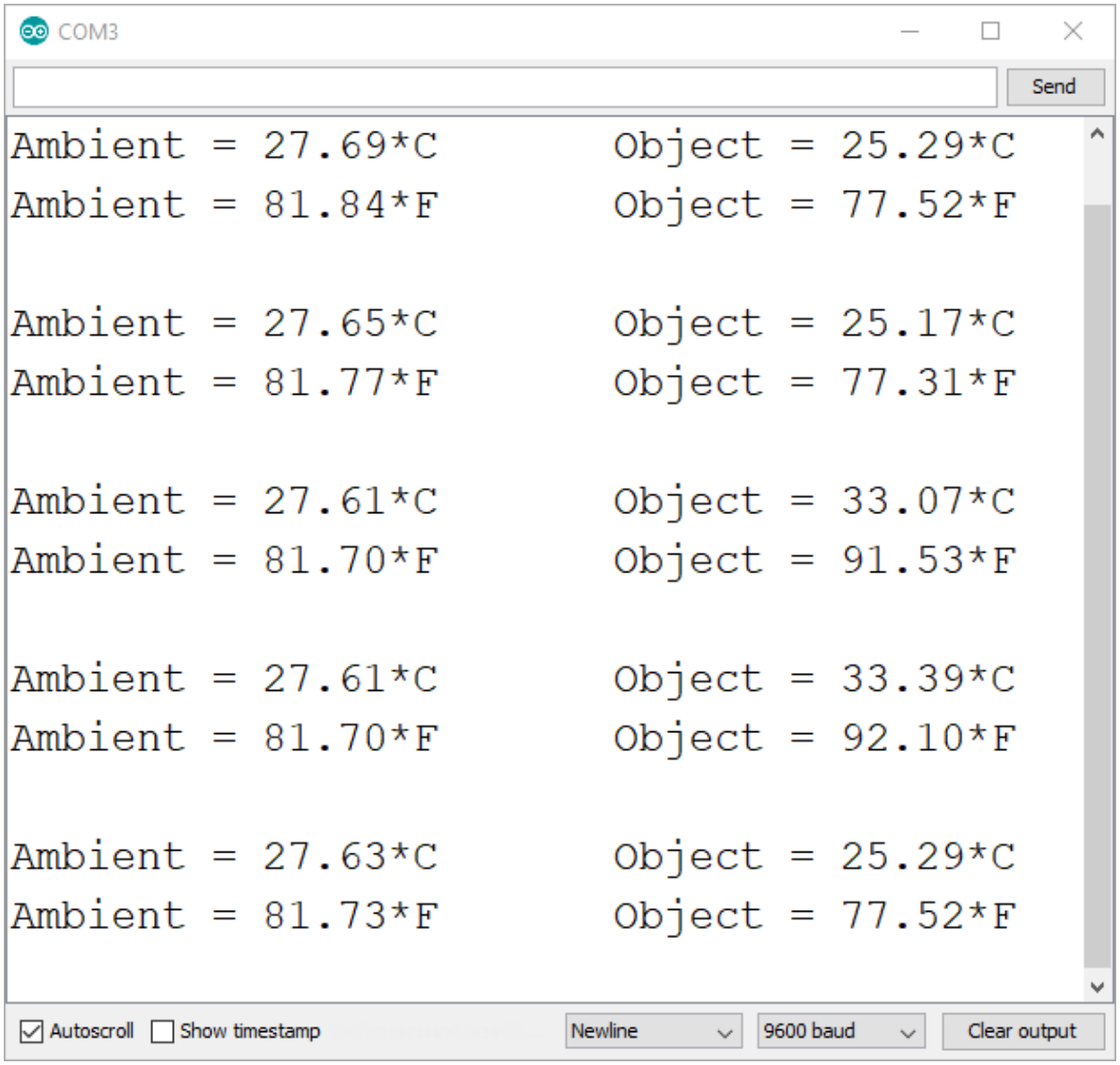
Sketch-Beispiel

Das folgende Sketch-Beispiel ist ein modifizierter Sketch aus der *Adafruit MLX90614 Library*:
File > Examples > Adafruit MLX90614 > mlxtest

```
#include <Draht.h>
#include <Adafruit_MLX90614.h>
Adafruit_MLX90614 itemp = Adafruit_MLX90614();
void setup() {
   Serial.begin(9600);
   Serial.println("Adafruit MLX90614 test");
   itemp.begin();
}
void loop() {
  Serial.print("\nAmbient = ");
  Serial.print(itemp.readAmbientTempC());
  Serial.print("*C\tObject = ");
  Serial.print(itemp.readObjectTempC());
  Serial.println("*C");
  Serial.print("Ambient = ");
  Serial.print(itemp.readAmbientTempF());
  Serial.print("*F\tObject = ");
  Serial.print(itemp.readObjectTempF());
  Serial.println("*F");
  Serial.println();
  delay(2500);
}
```

Laden Sie den Sketch in den Serial Monitor (*`Tools > Serial Monitor`*).

Die Ausgabe sollte wie folgt aussehen:



Der Sketch beginnt mit der Einbeziehung zweier Libraries: *Wire* und *Adafruit_MLX90614*. Die erste wird für die I2C-Kommunikation und die zweite für die Funktionen zur Kontrolle des Sensors verwendet.

Als Nächstes wird das Objekt *itemp* mit folgender Codezeile erstellt:

```
Adafruit_MLX90614 itemp = Adafruit_MLX90614;
```

Das Objekt *itemp* stellt das Modul dar und wird verwendet, um Sensordaten auszulesen.

In der *setup()* Funktion, wird die serielle Kommunikation mit einer Baudrate von *9600bps gestartet.*

Dann wird das Objekt *itemp* mit der folgenden Codezeile initialisiert:

```
itemp.begin()
```

In der *loop()* Funktion werden die Daten des Moduls ausgelesen und auf dem Serial Monitor angezeigt.

Um die Umgebungstemperatur in Celsius zu lesen, verwenden wir folgende Codezeile:

```
itemp.readAmbientTempC()
```

Um die Objekttemperatur in Celsius auszulesen (Objekt direkt vor dem Sensor), verwenden wir folgende Codzeile:

```
itemp.readObjectTempC()
```

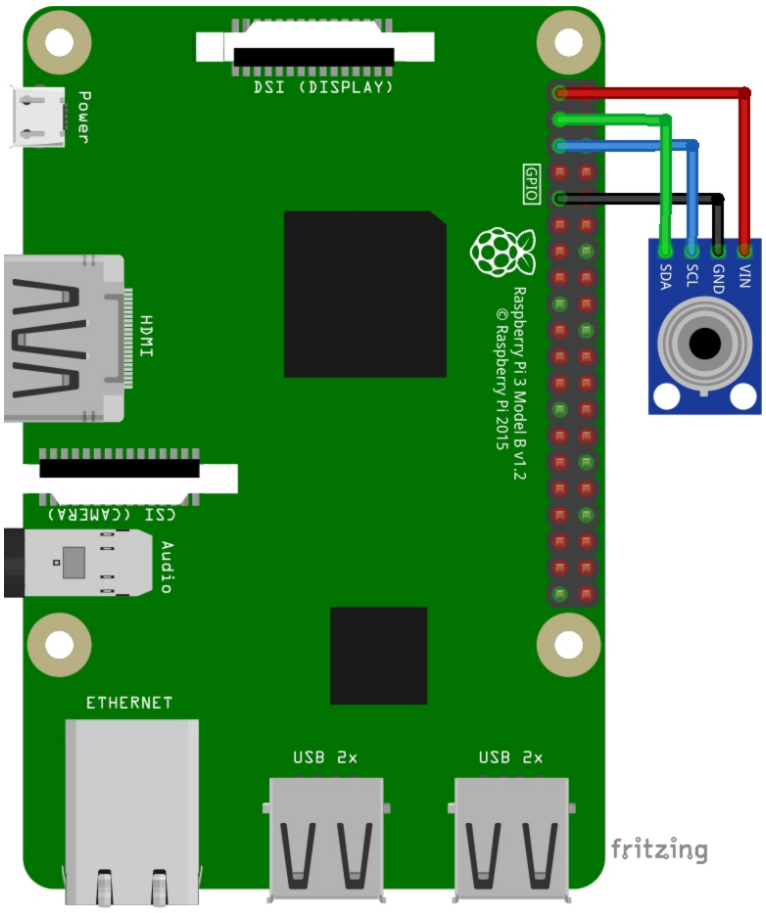
Um die Umgebungstemperatur in Fahrenheit auszulesen, verwenden wir:
`itemp.readAmbientTempF()`

Um die Objekttemperatur in Fahrenheit auszulesen (Objekt direkt vor dem Sensor), verwenden wir folgende Codzeile:
`itemp.readObjectTempF()`

Am Ende der *`loop()`* Funktion wird eine Verzögerungspause von 2,5 Sekunden (2500 Millisekunden) eingelegt, welche mit folgender Codezeile geändert werden kann:
`delay(2500);`

Verbindung des Moduls mit dem Raspberry Pi

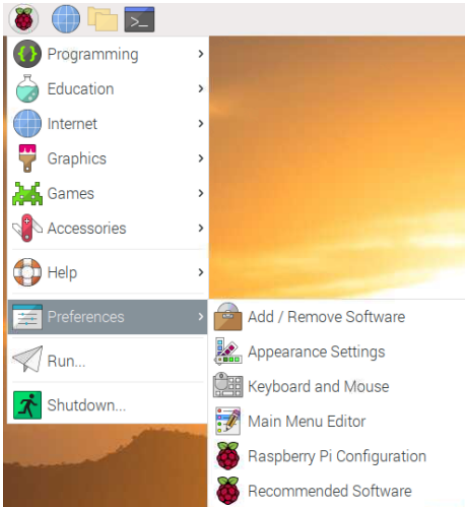
Verbinden Sie das GY-906 Modul mit dem Raspberry Pi, wie unten abgebildet:



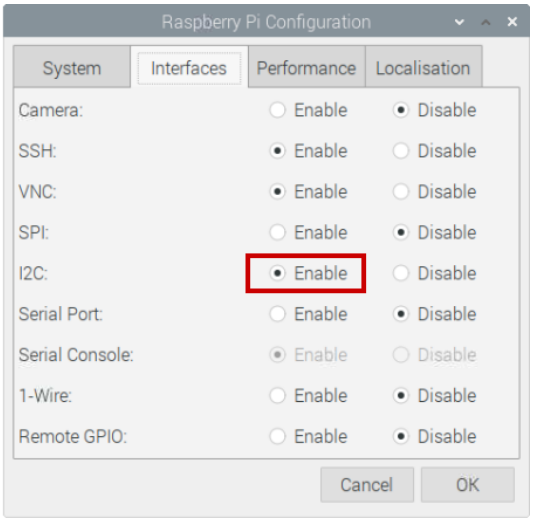
MLX90164 Pin	Raspberry Pi Pin	Physischer Pin No.	Drahtfarbe
GND	GND	9	**Schwarzer Draht**
VIN	3V3	1	**Roter Draht**
SCL	GPIO3	5	**Blauer Draht**
SDA	GPIO2	3	**Grüner Draht**

Aktivieren der I2C-Schnittstelle

Um den Bildschirm mit einer Raspberry Pi zu verwenden, muss zunächst die SPI-Schnittstelle in Raspbian aktiviert werden. Gehen Sie dafür zu: *Application Menu* > *Preferences* > *Raspberry Pi Configuration*



Als nächstes öffnen Sie die Registerkarte *`Interfaces/Schnittstelle`*, stellen Sie die SPI-Radiobuttons auf *Ok* und aktivieren Sie es, wie auf dem folgenden Bild gezeigt:



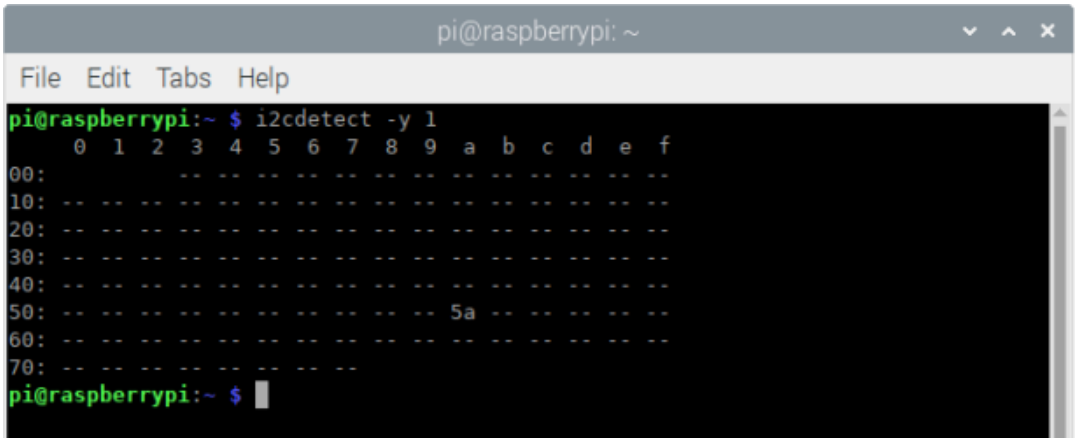
Um die I2C-Adresse des I2C-Adapters zu ermitteln, muss das *i2c-tools* installiert sein. Falls es nicht bereits installiert ist, führen Sie den folgenden Befehl im Terminal aus:

```
sudo apt-get update
sudo apt-get install i2c-tools
```

Danach öffnen Sie das Terminal und führen folgenden Befehl aus:

```
i2cdetect -y 1
```

Die Ausgabe sollte wie folgt aussehen:



wobei 0x5a die I2C-Adresse des I2C-Adapters darstellt.

Sollte die I2C-Schnittstelle nicht aktiviert und der vorherige Befehl ausgeführt worden sein, wird Ihnen folgende Fehlermeldung angezeigt:

Python-Skript

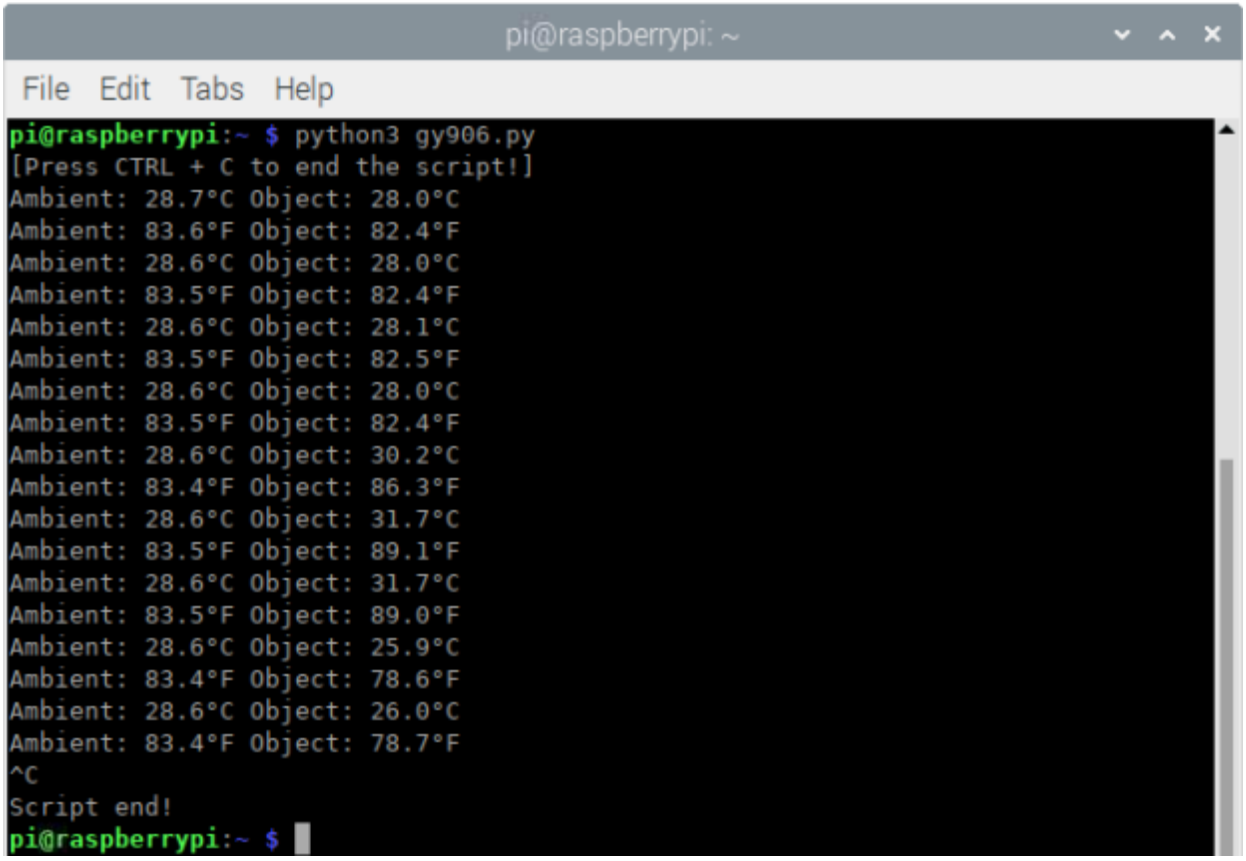
```
import time
import smbus
ds = u'\xb0' # UTF-8 degree sign
i2c_address = 0x5a # I2C address of the GY-906 sensor

# Temperature registers addresses
MLX90614_TA = 0x06 # Ambient temp
MLX90614_TOBJ1 = 0x07 # Object temp

# Read temperature registers and calculate Celsius
def ambC():
    val = bus.read_i2c_block_data(i2c_address, MLX90614_TA, 2)
    temp = val[1] << 8
    temp |= val[0]
    temp *= .02
    temp  -= 273.15
    return temp

def objC():
    val = bus.read_i2c_block_data(i2c_address, MLX90614_TOBJ1, 2)
    temp = val[1] << 8
    temp |= val[0]
     temp *= .02
    temp  -= 273.15
    return temp
```

```
def ambF():
    return ambC() * 9.0 / 5.0 + 32.0

def objF():
    return objC() * 9.0 / 5.0 + 32.0

# Initialize I2C (SMBus)
bus = smbus.SMBus(1)

print('[Press CTRL + C to end the script!]')
try:
    while True:
        print('Ambient: {:.1f}{}C\tObject: {:.1f}{}C'.
            format(ambC(), ds, objC(), ds))
        print('Ambient: {:.1f}{}F\tObject: {:.1f}{}F'.
            format(ambF(), ds, objF(), ds))
        time.sleep(1)

except KeyboardInterrupt:
    print('\nScript end!')
```

Speichern Sie das Skript unter dem Namen *gy906.py*. Um das Skript auszuführen, öffnen Sie das Terminal in dem Verzeichnis, in dem das Skript gespeichert wurde, und führen Sie den folgenden Befehl aus:

`python3 gy906.py`

Die Ausgabe sollte wie folgt aussehen:



Um das Skript zu stoppen, drücken Sie 'Strg + C' auf der Tastatur.

Das Skript beginnt mit der Einbeziehung folgender Libraries: *time* und *SMBus*. Die *time* Library wird für die Zeitfunktionalität verwendet und der *SMBus* (System Management Bus) wird zum Lesen/Schreiben von Informationen auf der I2C-Schnittstelle verwendet.

Als Nächstes wir die Variable *ds* erstellt. Der UTF-Grad-Zeichenwert wird hier gespeichert. Er wird benötigt, um das Ergebnis genauer anzuzeigen.

Dann werden die Adressen von zwei Registern des Sensors in den Variablen mit den Namen *MLX90614_TA* and *MLX90614_TOBJ1,* mit der folgenden Codezeile gespeichert:

```
MLX90614_TA = 0x06
MLX90614_TOBJ1 = 0x07
```

Im Register *MLX90614_TA* werden die Umgebungsdaten gespeichert und in dem Register *MLX90614_TOBJ1* die Objektdaten.

Als nächstes werden zwei Funktionen erstellt, die Temperaturdaten aus dem Sensor lesen. Die erste Funktion heißt *ambC()*. Sie hat keine Argumente und gibt einen Float-Wert zurück. Die Funktion liest das Register, das die Umgebungstemperaturdaten enthält, und wandelt die Rohdaten in lesbare Temperatur in Celsius um. Der zurückgegebene Float-Wert stellt die Umgebungstemperaturdaten in Celsius dar.

Die zweite Funktion heißt *objC().* Sie besitzt keine Argumente und gibt einen Float-Wert zurück. Die Funktion liest das Register, das die Objekttemperaturdaten enthält und wandelt die Rohdaten in lesbare Temperatur in Celsius um. Der zurückgegebene Float-Wert stellt die Objekttemperaturdaten in Celsius dar.

Um die Temperatur von Celsius in Fahrenheit umzuwandeln, werden zwei Funktionen namens *ambF()* und *objF()* erstellt. Die *ambF()* Funktion verwendet eine Formel, um die von der *ambC()* Funktion gelesenen Celsius-Daten in Fahrenheit umzuwandeln. Die *objF()* Funktion verwendet die Formel, um die von der *objC()* Funktion gelesenen Celsius-Daten in Fahrenheit umzuwandeln.

Als Nächstes wird die I2C-Kommunikation initialisiert:
bus = smbus.SMBus(1)

Dann wird ein *try-except-finally* Codeblock erstellt. Im *try* block wird ein indefinite loop block (*while True:*) erstellt. Innerhalb dieses Blocks werden Daten gelesen und im Terminal angezeigt. Zwischen beiden loops des indefinite loop Blocks gibt es eine Pause von einer Sekunde. Das Intervall kann mit foldender Codezeile verändert werden: *time.sleep(1)*

Der *except* Codeblock wird mit *Strg+C* ausgeführt. Das wird *KeyboardInterrupt* genannt Wenn dieser Codeblock ausgeführt wurde, wird im Terminal die Nachricht *Script end!* Angezeigt.

Jetzt sind Sie dran! Entwickeln Sie Ihre eigenen Projekte und Smart-Home Installationen. Wie Sie das bewerkstelligen können, zeigen wir Ihnen unkompliziert und verständlich auf unserem Blog. Dort bieten wir Ihnen Beispielskripte und Tutorials mit interessanten kleinen Projekten an, um schnell in die Welt der Mikroelektronik einzusteigen. Zusätzlich bietet Ihnen auch das Internet unzählige Möglichkeiten, um sich in Sachen Mikroelektronik weiterzubilden.

Falls Sie nach weiteren hochwertigen Produkten für Arduino und Raspberry Pi suchen, sind Sie bei AZ-Delivery Vertriebs GmbH goldrichtig. Wir bieten Ihnen zahlreiche Anwendungsbeispiele, ausführliche Installationsanleitungen, E-Books, Bibliotheken und natürlich die Unterstützung unserer technischen Experten.

https://az-delivery.de

Viel Spaß!

Impressum

https://az-delivery.de/pages/about-us