

## Willkommen!

Und herzlichen Dank für den Kauf unseres AZ-Delivery 2 Kanal Relais Moduls. Auf den folgenden Seiten gehen wir mit dir gemeinsam die ersten Programmierschritte durch.

Viel Spaß!



Mit den 2 Relais können größere Lasten (bis 10A) an einem Arduino, Raspberry Pi usw. betrieben werden.

**WARNUNG!! ES BESTEHT LEBENSGEFAHR DURCH EINEN ELEKTRISCHEN SCHLAG BEI BETRIEB ÜBER 30V ODER 230V NETZSPANNUNG. ACHTEN SIE AUF ENTSPRECHENDE ISOLIERUNG UND SCHUTZVORKEHRUNGEN.**

## Ansteuern des Relais:

Das Relais wird ganz einfach angesteuert, wird der Ausgangspegel auf **LOW** geschaltet, so zieht das Relais an und wird eingeschaltet.

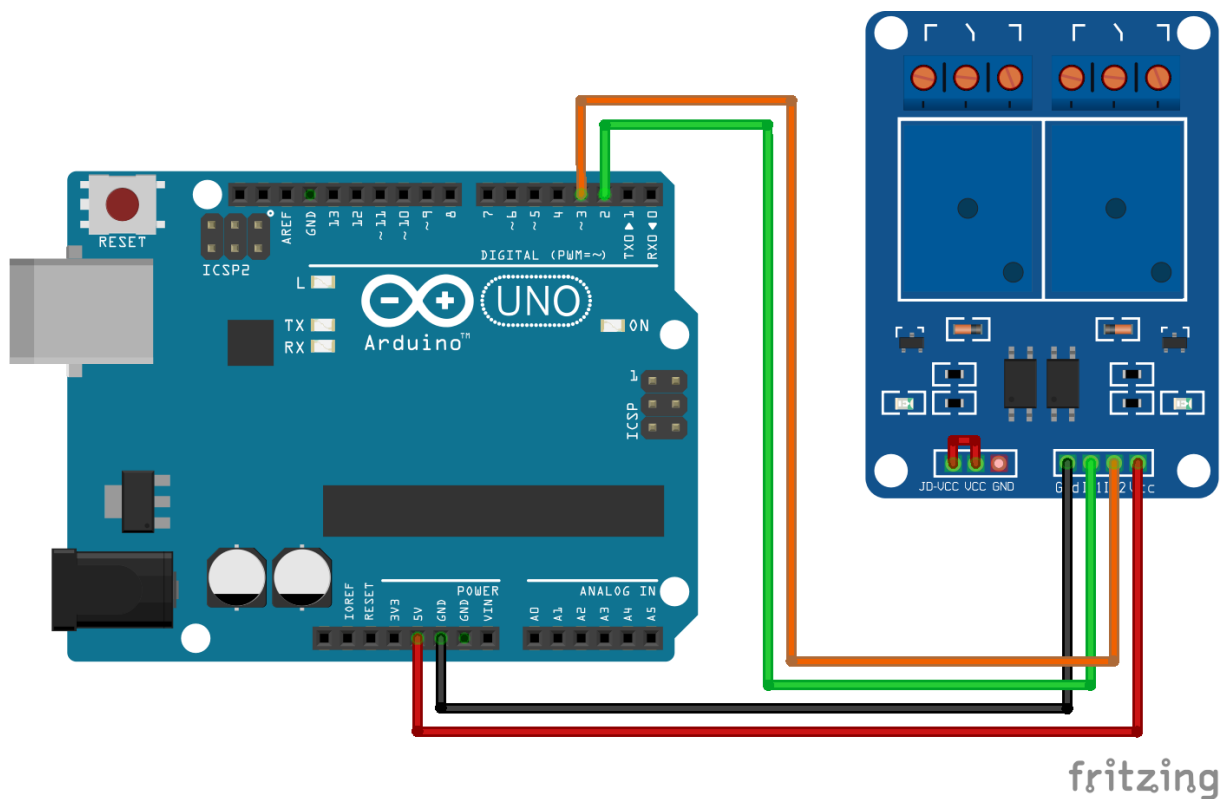
Das Relais hat einen Wechsler-Ausgang mit einem Öffner (NC) und einen Schließer (NO). Je nachdem an welchen Ausgang die Last angeschlossen wird, kann die Last aus bzw. eingeschaltet werden.

## Verwendung der Relais an einem Arduino

### Verdrahten des Moduls mit einem Arduino Uno:

**VCC** wird mit **5V** am Arduino verbunden  
**GND** wird mit **GND** verbunden  
**IN1** wird mit **PIN 2** verbunden  
**IN2** wird mit **PIN 3** verbunden

Rote Leitung  
Schwarze Leitung  
Grüne Leitung  
Orange Leitung



### Vorbereiten der Software:


Die Arduino Software sehen wir in diesem Schritt als Installiert an, sollte diese bei dir noch fehlen, so kannst du diese unter <https://www.arduino.cc/en/Main/Software#> herunterladen und auf deinen PC installieren.

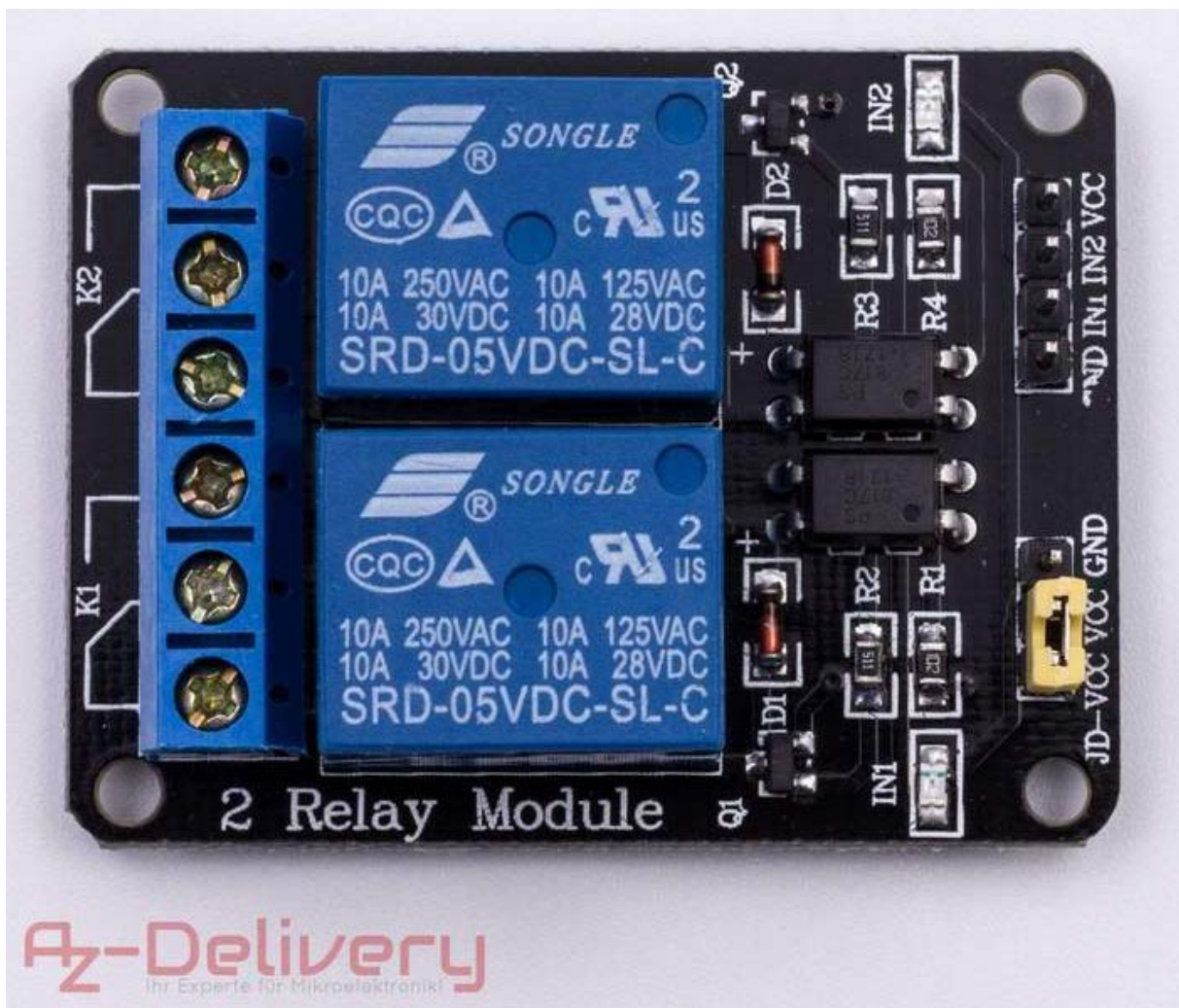
## Der Code für einen Arduino:

```
const int RELAIS1 = 2;           //Arduino Pin 2
const int RELAIS2 = 3;           //Arduino Pin 3

void setup() {
  pinMode(RELAIS1, OUTPUT);
  pinMode(RELAIS2, OUTPUT);
}

void loop() {
  digitalWrite(RELAIS1, HIGH);
  digitalWrite(RELAIS2, LOW);
  delay(1000);
  digitalWrite(RELAIS1, LOW);
  digitalWrite(RELAIS2, HIGH);
  delay(1000);
}
```

Nach dem übertragen  werden die Relais für 1 Sekunde abwechselnd eingeschaltet und wieder ausgeschaltet.

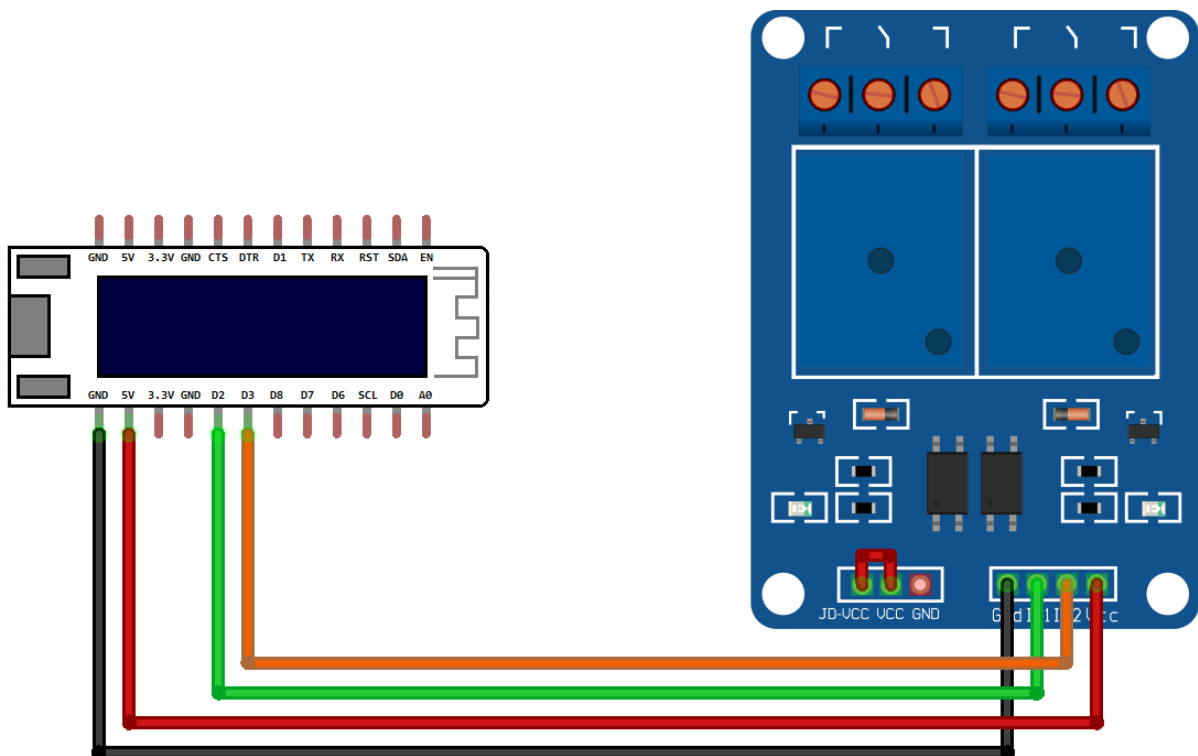


## Verwenden der Relais an einem ESP8266

### Verdrahten des Moduls mit einem ESP8266 (mit OLED):

VCC wird mit **5V** am ESP8266 verbunden  
GND wird mit **GND** verbunden  
IN1 wird mit **PIN D2** verbunden  
IN2 wird mit **PIN D3** verbunden

Rote Leitung  
Schwarze Leitung  
Grüne Leitung  
Orange Leitung




### Der Code für ESP8266:

fritzing

```
const int RELAIS1 = D2;           //ESP GPIO Pin 2
const int RELAIS2 = D3;           //ESP GPIO Pin 3

void setup() {
  pinMode(RELAIS1, OUTPUT);
  pinMode(RELAIS2, OUTPUT);
}

void loop() {
  digitalWrite(RELAIS1, HIGH);
  digitalWrite(RELAIS2, LOW);
  delay(1000);
  digitalWrite(RELAIS1, LOW);
  digitalWrite(RELAIS2, HIGH);
  delay(1000);
}
```

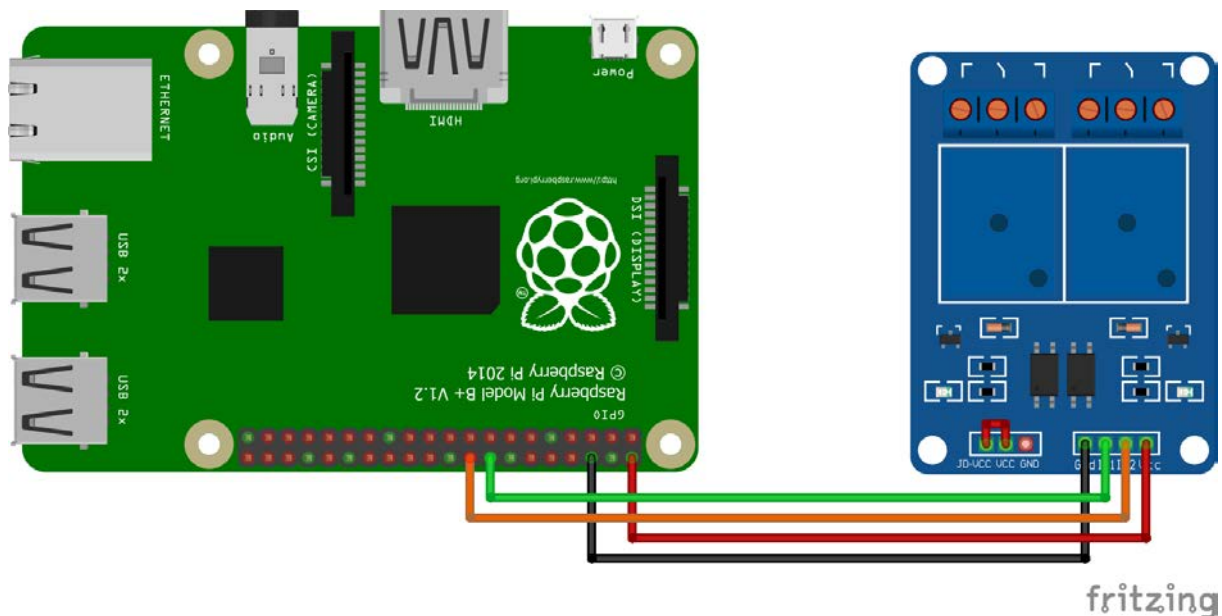
Nach dem Übertragen  werden die Relais für 5 Sekunden abwechselnd eingeschaltet und wieder ausgeschaltet.

## Verwendung der Relais an einem Raspberry Pi

### Verdrahten des Moduls mit einem Raspberry Pi:

VCC wird mit **5V** am Raspberry verbunden  
GND wird mit **GND** verbunden  
IN1 wird mit **GPIO 4** verbunden  
IN2 wird mit **GPIO 5** verbunden

Rote Leitung  
Schwarze Leitung  
Grüne Leitung  
Orange Leitung



### Vorbereiten der Software:

Der Raspberry sollte entsprechend dem eBook für Raspberry Pi vorbereitet werden und aktualisiert werden.

Installation von wiringPi falls **gpio -v** keine Informationen ausgibt:

```
gpio -v
sudo apt-get purge wiringPi
sudo apt-get install git-core
git clone git://git.drogon.net/wiringPi
cd ~/wiringPi
git pull origin
cd ~/wiringPi
./build
sudo ./build
sudo reboot
```

Version prüfen  
Deinstallieren von alter Version  
Installieren von git  
wiringPi herunterladen  
in das Verzeichnis wechseln  
prüfen auf aktuelle Version  
in das Verzeichnis wechseln  
wiringPi Kompilieren  
wiringPi Installieren  
Raspberry Pi neustarten

Nun legen wir ein neues Programm an:



**touch relais.py**

Anlegen einer neuen Datei „relais.py“

**nano relais.py**

Datei im Editor öffnen

Dann fügen wir folgenden Code in den Editor ein:

```
#!/usr/bin/python
from time import sleep
import RPi.GPIO as GPIO
GPIO.setmode(GPIO.BCM)
GPIO.setup(23,GPIO.OUT)
GPIO.setup(24,GPIO.OUT)
while True:
    GPIO.output(23,GPIO.HIGH)
    GPIO.output(24,GPIO.LOW)
    sleep(1)
    GPIO.output(23,GPIO.LOW)
    GPIO.output(24,GPIO.HIGH)
    sleep(1)
```

# Module einbinden  
# Pin Beschreibung auf BCM  
# GPIO4 als Ausgang festlegen  
# GPIO5 als Ausgang festlegen  
# Schleife generieren  
# Relais 1 AUS  
# Relais 2 EIN  
# warte 1s  
# Relais 1 EIN  
# Relais 2 AUS  
# warte 1s

Das Programm führen mit dem Befehl

**python relais.py**

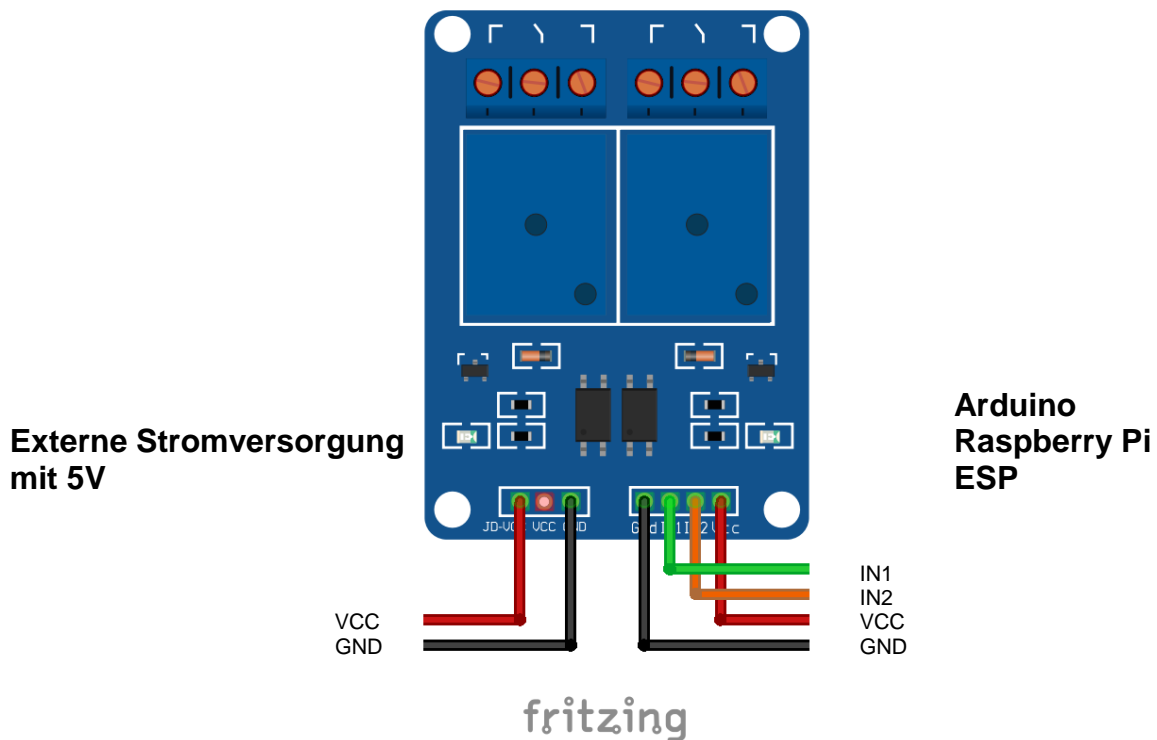
aus. Die Relais werden für 5 Sekunde abwechselnd eingeschaltet und wieder ausgeschaltet, solange bis wir das Programm mit STRG+C beenden.

**Du hast es geschafft, du kannst nun in deinen Projekten ein Relais verwenden und größere Verbraucher schalten!**



## Externe Stromversorgung für das Modul:

Sollte die Stromversorgung über einen Raspberry Pi, Arduino, ESP usw. nicht ausreichend sein, so auch eine externe Stromversorgung von 5V verwendet werden.



Es besteht keine Galvanische Trennung zwischen dem Raspberry Pi, Arduino, ESP oder ähnlichem. Die Masse (GND) ist verbunden. JD-VCC und VCC dürfen nicht miteinander verbunden werden und der Jumper muss entfernt werden.

Ein Relais benötigt 0.45W, was einen Strombedarf von 0,09A bedeutet. Das Relaismodul benötigt bei beiden eingeschalteten Relais somit 0,18A. Das kann für manche  $\mu$ C-Module zu viel sein.

Ab jetzt heißt es Experimentieren.

Und für mehr Hardware sorgt natürlich dein Online-Shop auf:

<https://az-delivery.de>

Viel Spaß!

Impressum

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