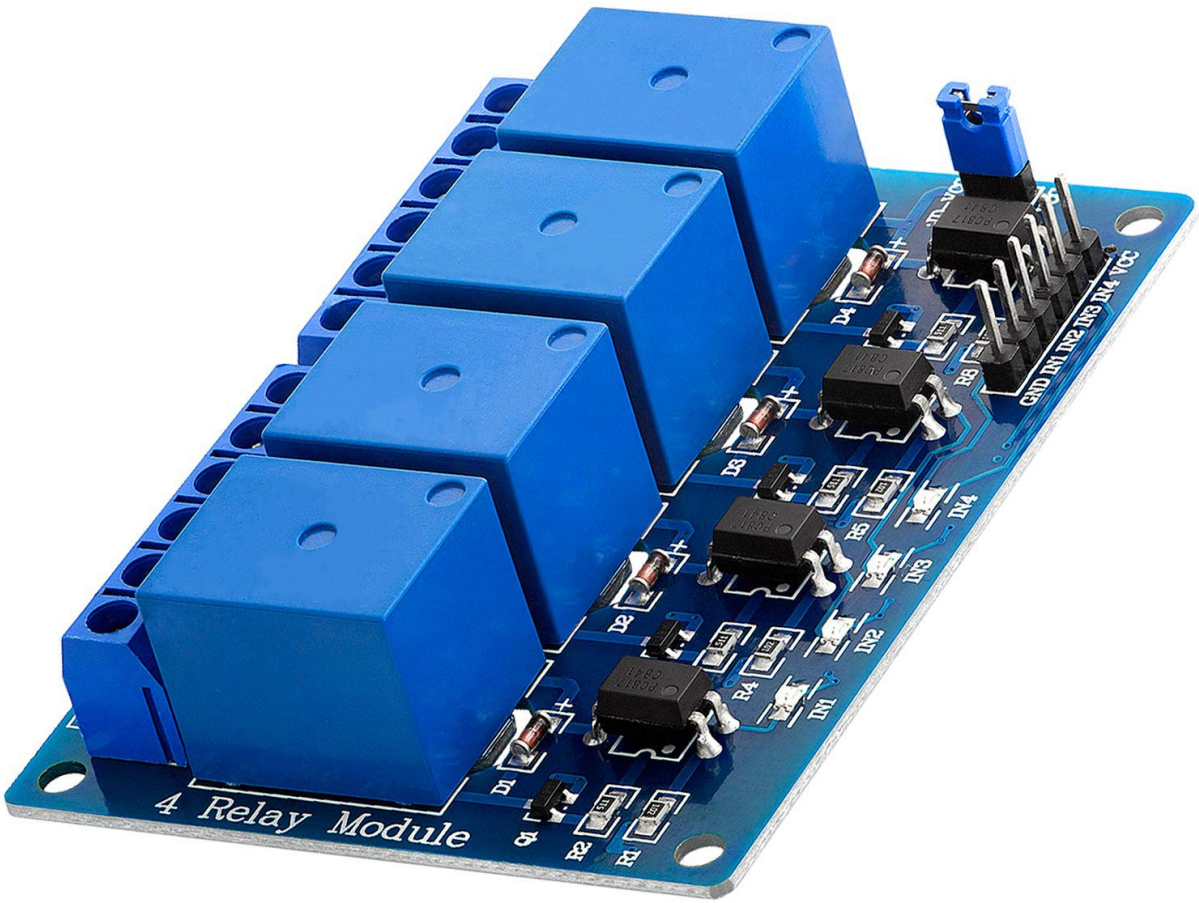
Willkommen!

Und herzlichen Dank für den Kauf unseres AZ-Delivery 4 Kanal Relais Moduls. Auf den folgenden Seiten gehen wir mit dir gemeinsam die ersten Programmierschritte durch.

Viel Spaß!



Mit den 4 Relais können größere Lasten (bis 5A) an einem Mikrocontroller usw. betrieben werden.

WARNUNG!! ES BESTEHT LEBENSGEFAHR DURCH EINEN ELEKTRISCHEN SCHLAG BEI BETRIEB ÜBER 30V ODER 50V NETZSPANNUNG. ACHTEN SIE AUF ENTSPRECHENDE ISOLIERUNG UND SCHUTZVORKEHRUNGEN.

Ansteuern des Relais:

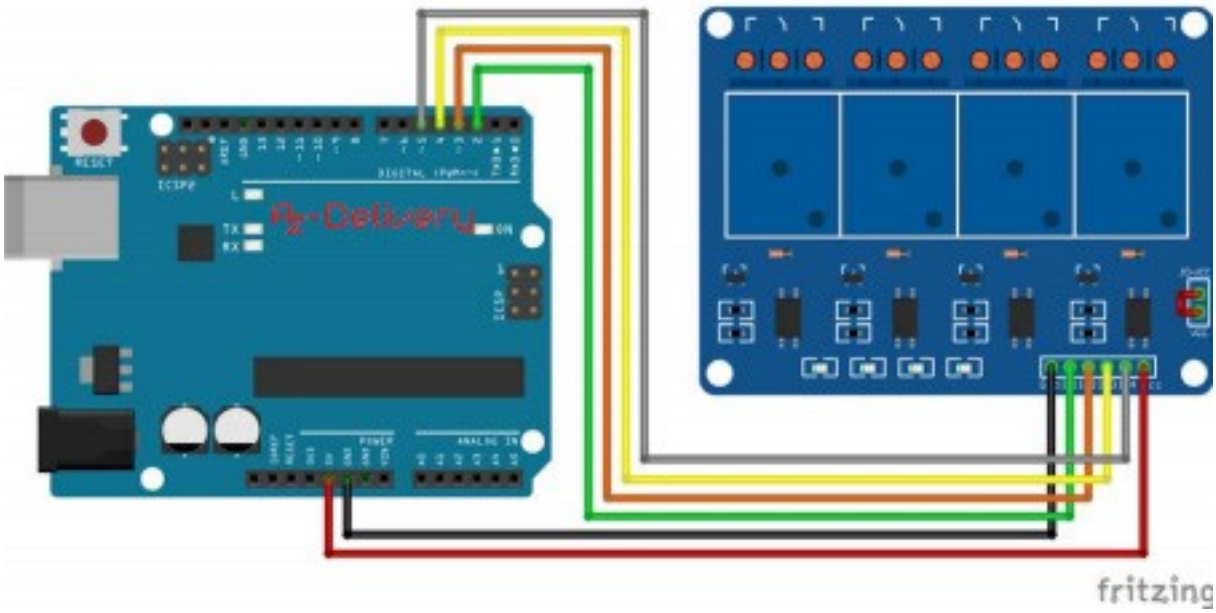
Das Relais wird ganz einfach angesteuert, wird der Ausgangspegel auf **LOW** geschaltet, so zieht das Relais an und wird eingeschaltet.

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 Mikrocontroller

Verdrahten des Moduls mit einem ATmega328p Mikrocontroller:

VCC wird mit **5V** am ATmega328p MC verbunden Rote Leitung **GND** wird mit **GND** verbunden Schwarze Leitung **IN1** wird mit **PIN 2** verbunden Grüne Leitung **IN2** wird mit **PIN 3** verbunden Orange Leitung **IN3** wird mit **PIN 4** verbunden Gelbe Leitung **IN4** wird mit **PIN 5** verbunden Graue Leitung



Vorbereiten der Software:

Die Arduino-IDE 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 Mikrocontroller:

```
const int RELAIS1 = 2; //Arduino Pin 2
const int RELAIS2 = 3; //Arduino Pin 3
const int RELAIS3 = 4; //Arduino Pin 4
const int RELAIS4 = 5; //Arduino Pin 5

void setup() {
  pinMode(RELAIS1, OUTPUT);
  pinMode(RELAIS2, OUTPUT);
  pinMode(RELAIS3, OUTPUT);
  pinMode(RELAIS4, OUTPUT);
}

void loop() {
  digitalWrite(RELAIS1, LOW);
  digitalWrite(RELAIS2, HIGH);
  digitalWrite(RELAIS3, HIGH);
  digitalWrite(RELAIS4, HIGH);
  delay(5000);
  digitalWrite(RELAIS1, HIGH);
  digitalWrite(RELAIS2, LOW);
  digitalWrite(RELAIS3, HIGH);
  digitalWrite(RELAIS4, HIGH);
  delay(5000);
  digitalWrite(RELAIS1, HIGH);
  digitalWrite(RELAIS2, HIGH);
  digitalWrite(RELAIS3, LOW);
  digitalWrite(RELAIS4, HIGH);
  delay(5000);
  digitalWrite(RELAIS1, HIGH);
  digitalWrite(RELAIS2, HIGH);
  digitalWrite(RELAIS3, HIGH);
  digitalWrite(RELAIS4, LOW);
  delay(5000);
}
```

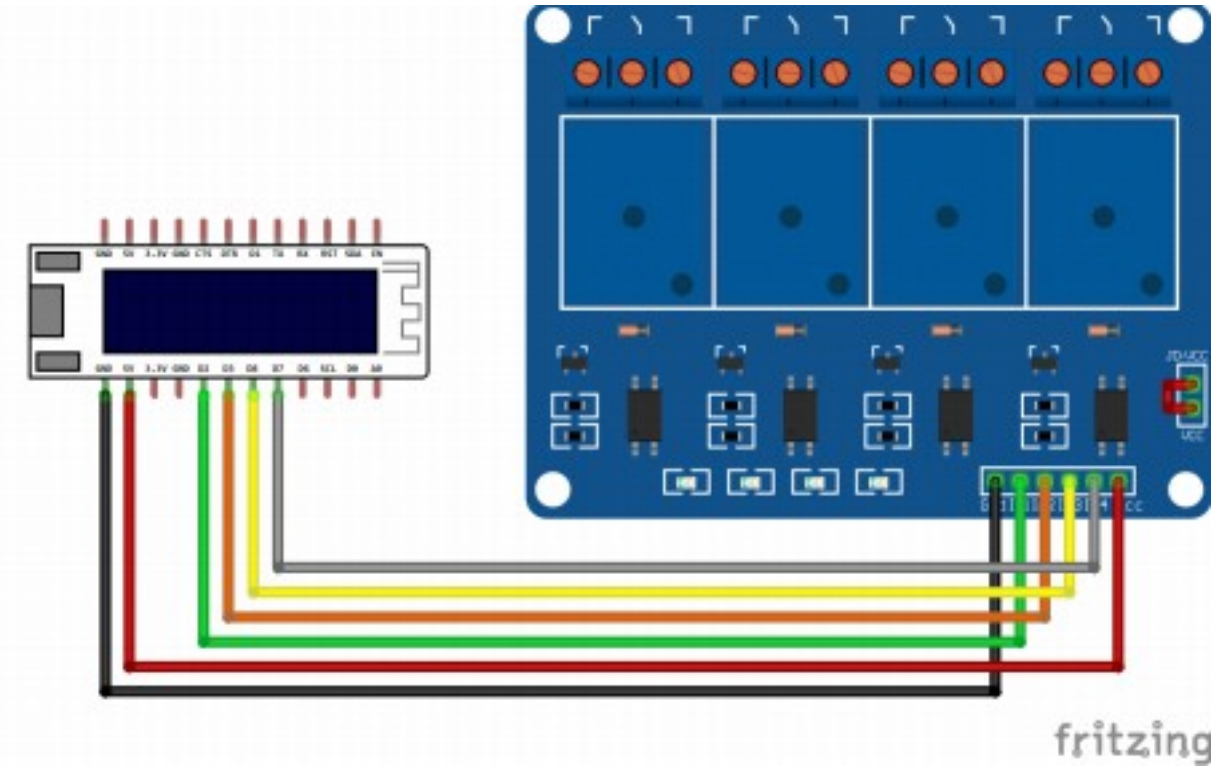
Nach dem übertragen  werden die Relais für 5 Sekunde nacheinander eingeschaltet und wieder ausgeschaltet.

Verwenden der Relais

an einem ESP8266

Verdrahten des Moduls mit einem ESP8266 (mit OLED):

VCC wird mit **5V** am ESP8266 verbunden Rote Leitung **GND** wird mit **GND** verbunden Schwarze Leitung **IN1** wird mit **PIN D2** verbunden Grüne Leitung **IN2** wird mit **PIN D3** verbunden Orange Leitung **IN3** wird mit **PIN D8** verbunden Gelbe Leitung **IN4** wird mit **PIN D7** verbunden Graue Leitung



Der Code für ESP8266:

```
const int RELAIS1 = D2; //ESP GPIO Pin 2
const int RELAIS2 = D3; //ESP GPIO Pin 3
const int RELAIS3 = D8; //ESP GPIO Pin 8
const int RELAIS4 = D7; //ESP GPIO Pin 7

void setup() {
  pinMode(RELAIS1, OUTPUT);
  pinMode(RELAIS2, OUTPUT);
  pinMode(RELAIS3, OUTPUT);
  pinMode(RELAIS4, OUTPUT);
}

void loop() {
  digitalWrite(RELAIS1, LOW);
  digitalWrite(RELAIS2, HIGH);
  digitalWrite(RELAIS3, HIGH);
  digitalWrite(RELAIS4, HIGH);
  delay(1000);
  digitalWrite(RELAIS1, HIGH);
  digitalWrite(RELAIS2, LOW);
  digitalWrite(RELAIS3, HIGH);
  digitalWrite(RELAIS4, HIGH);
  delay(1000);
  digitalWrite(RELAIS1, HIGH);
  digitalWrite(RELAIS2, HIGH);
  digitalWrite(RELAIS3, LOW);
  digitalWrite(RELAIS4, HIGH);
  delay(1000);
  digitalWrite(RELAIS1, HIGH);
  digitalWrite(RELAIS2, HIGH);
  digitalWrite(RELAIS3, HIGH);
  digitalWrite(RELAIS4, LOW);
  delay(1000);
}
```

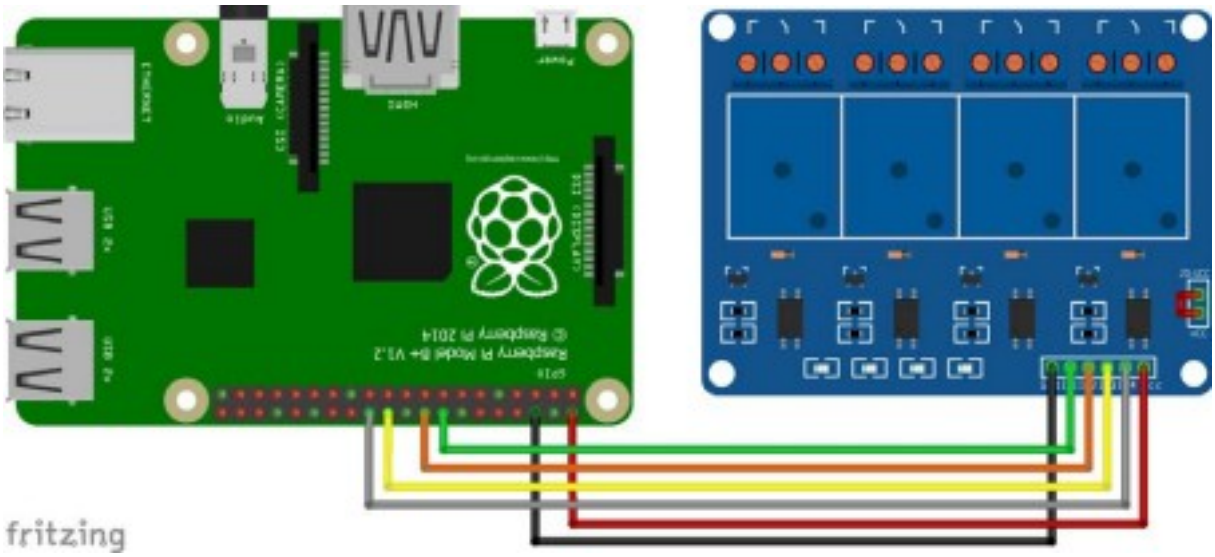
Nach dem übertragen  werden die Relais für 1 Sekunde nacheinander eingeschaltet und wieder ausgeschaltet.

.

Verwendung der Relais
an einem Raspberry Pi

Verdrahten des Moduls mit einem Raspberry Pi:

VCC wird mit **5V** am Raspberry verbunden Rote Leitung **GND** wird mit **GND** verbunden Schwarze Leitung **IN1** wird mit **GPIO 4** verbunden Grüne Leitung **IN2** wird mit **GPIO 5** verbunden Orange Leitung **IN3** wird mit **GPIO 6** verbunden Gelbe Leitung **IN4** wird mit **GPIO 10** verbunden Graue 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` Version prüfen `sudo apt-get purge wiringPi` Deinstallieren von alter Version `sudo apt-get install git-core` Installieren von git `git clone git://git.drogon.net/wiringPi` wiringPi herunterladen `cd ~/wiringPi` in das Verzeichnis wechseln `git pull origin` prüfen auf aktuelle Version `cd ~/wiringPi` in das Verzeichnis wechseln `./build` wiringPi Kompilieren `sudo ./build` wiringPi Installieren `sudo reboot` 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 # Module einbinden
GPIO.setmode(GPIO.BCM) # Pin Beschreibung auf BCM
GPIO.setup(23,GPIO.OUT) # GPIO4 als Ausgang festlegen
GPIO.setup(24,GPIO.OUT) # GPIO5 als Ausgang festlegen
GPIO.setup(25,GPIO.OUT) # GPIO6 als Ausgang festlegen
GPIO.setup(8,GPIO.OUT) # GPIO10 als Ausgang festlegen while
True: # Schleife generieren  GPIO.output(23,GPIO.LOW) # Relais 1
EIN
 GPIO.output(24,GPIO.HIGH) # Relais 2 AUS
  GPIO.output(25,GPIO.HIGH) # Relais 3 AUS
 GPIO.output(8,GPIO.HIGH) # Relais 4 AUS
 sleep(1) # warte 1s
 GPIO.output(23,GPIO.HIGH) # Relais 1 AUS
 GPIO.output(24,GPIO.LOW) # Relais 2 EIN
  GPIO.output(25,GPIO.HIGH) # Relais 3 AUS
 GPIO.output(8,GPIO.HIGH) # Relais 4 AUS
 sleep(1) # warte 1s
  GPIO.output(23,GPIO.HIGH) # Relais 1 AUS
 GPIO.output(24,GPIO.HIGH) # Relais 2 AUS
  GPIO.output(25,GPIO.LOW) # Relais 3 EIN
 GPIO.output(8,GPIO.HIGH) # Relais 4 AUS
 sleep(1) # warte 1s
  GPIO.output(23,GPIO.HIGH) # Relais 1 AUS
 GPIO.output(24,GPIO.HIGH) # Relais 2 AUS
  GPIO.output(25,GPIO.HIGH) # Relais 3 AUS
 GPIO.output(8,GPIO.LOW) # Relais 4 EIN
 sleep(1) # warte 1s
```

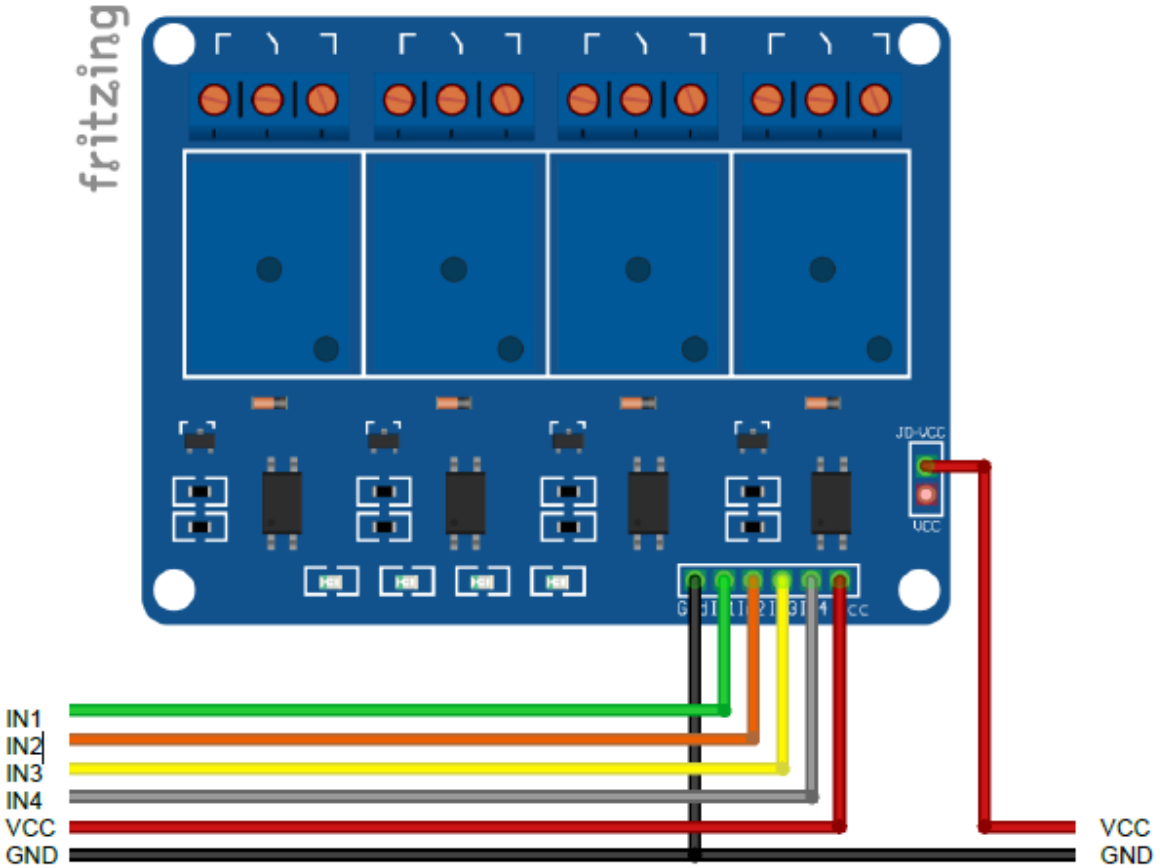
Das Programm führen mit dem Befehl

`python relais.py`

aus. Die Relais für 1 Sekunde nacheinander eingeschaltet und wieder ausgeschaltet, solange bis wir das Programm mit STRG+C beenden.

Externe Stromversorgung für das Modul:

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



Mikrocontroller **Externe Stromversorgung ESP mit 5V**

Es besteht keine Galvanische Trennung zwischen dem Mikrocontroller, 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 0,36A. Das ist für manche µC-Module schon zu viel!

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

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