Um den **28BYJ-48 Schrittmotor** mit einem **Raspberry Pi Zero** zu steuern, benötigst du den **ULN2003 Treiber**. Hier ist eine Schritt-für-Schritt-Anleitung:

**Benötigte Komponenten**

* Raspberry Pi Zero (mit Raspbian OS)
* 28BYJ-48 Schrittmotor
* ULN2003 Motortreiber-Board
* Jumper-Kabel

**Schaltplan (Verdrahtung)**

Der **28BYJ-48** wird am **ULN2003 Treiber** angeschlossen, dieser wiederum mit dem **Raspberry Pi Zero** verbunden.

| **ULN2003 Pin** | **Raspberry Pi Zero GPIO** |
| --- | --- |
| IN1 | GPIO 17 (Pin 11) |
| IN2 | GPIO 18 (Pin 12) |
| IN3 | GPIO 27 (Pin 13) |
| IN4 | GPIO 22 (Pin 15) |
| VCC (5V) | 5V (Pin 2 oder 4) |
| GND | GND (Pin 6) |

Der Motor selbst wird mit der 5-poligen Buchse am ULN2003 Treiber angeschlossen.

**Python-Code zur Steuerung**

Installiere zuerst **RPi.GPIO**, falls nicht vorhanden:

sh

KopierenBearbeiten

sudo apt update

sudo apt install python3-rpi.gpio

Dann ein einfaches Python-Skript zur Steuerung des Motors:

python

KopierenBearbeiten

import RPi.GPIO as GPIO

import time

# GPIO-Modus setzen

GPIO.setmode(GPIO.BCM)

# Pins für den ULN2003 Motor Driver

IN1 = 17

IN2 = 18

IN3 = 27

IN4 = 22

# Pin-Setup

motor\_pins = [IN1, IN2, IN3, IN4]

for pin in motor\_pins:

GPIO.setup(pin, GPIO.OUT)

GPIO.output(pin, 0)

# Schrittsequenz für den Motor (Halbschritt)

seq = [

[1, 0, 0, 0],

[1, 1, 0, 0],

[0, 1, 0, 0],

[0, 1, 1, 0],

[0, 0, 1, 0],

[0, 0, 1, 1],

[0, 0, 0, 1],

[1, 0, 0, 1]

]

def motor\_steuern(steps, delay=0.002):

"""Bewegt den Motor um eine bestimmte Anzahl von Schritten"""

for \_ in range(steps):

for step in seq:

for pin, val in zip(motor\_pins, step):

GPIO.output(pin, val)

time.sleep(delay)

try:

print("Motor dreht sich vorwärts...")

motor\_steuern(512) # 512 Schritte = ca. eine Umdrehung

time.sleep(1)

print("Motor dreht sich rückwärts...")

motor\_steuern(-512)

finally:

GPIO.cleanup()

**Erklärung**

* Der **28BYJ-48** benötigt **4096 Schritte** für eine volle Umdrehung (bei Halbschrittmodus).
* **512 Schritte** entsprechen ca. **45° Drehung**.
* **delay=0.002** beeinflusst die Geschwindigkeit – größere Werte = langsamer.
* **GPIO.cleanup()** stellt sicher, dass die Pins nach Programmende freigegeben werden.