This code assumes that the vibration motor is connected to GPIO pin 18 on the Raspberry Pi. You may need to adjust the pin number to match your specific setup.

The code defines a **vibrate** function that turns the vibration motor on, waits for a specified duration (in seconds), and then turns the motor off. The main loop of the code repeatedly calls the **vibrate** function with a duration of 1 second and then waits for 1 second before repeating the process.

You can modify the code to customize the vibration pattern by changing the duration and timing of the calls to the **vibrate** function.

import RPi.GPIO as GPIO

import time

# Set GPIO pin number for vibration motor

MOTOR\_PIN = 18

# Set GPIO mode to BCM

GPIO.setmode(GPIO.BCM)

# Set up motor pin as output

GPIO.setup(MOTOR\_PIN, GPIO.OUT)

def vibrate(duration):

# Turn motor on

GPIO.output(MOTOR\_PIN, GPIO.HIGH)

# Wait for duration in seconds

time.sleep(duration)

# Turn motor off

GPIO.output(MOTOR\_PIN, GPIO.LOW)

# Vibrate motor for 1 second, then pause for 1 second, and repeat

while True:

vibrate(1)

time.sleep(1)

Vibrates===============================================

===========Ultrasonic======================================

import RPi.GPIO as GPIO

import time

# Set GPIO pin numbers

TRIG = 23

ECHO = 24

# Set GPIO mode to BCM

GPIO.setmode(GPIO.BCM)

# Set up TRIG and ECHO pins

GPIO.setup(TRIG, GPIO.OUT)

GPIO.setup(ECHO, GPIO.IN)

def measure\_distance():

# Send 10us pulse to trigger pin

GPIO.output(TRIG, True)

time.sleep(0.00001)

GPIO.output(TRIG, False)

# Wait for echo high and low

while GPIO.input(ECHO) == 0:

pulse\_start = time.time()

while GPIO.input(ECHO) == 1:

pulse\_end = time.time()

# Calculate pulse duration and distance

pulse\_duration = pulse\_end - pulse\_start

distance = pulse\_duration \* 17150

distance = round(distance, 2)

return distance

# Measure distance every second and print it

while True:

distance = measure\_distance()

print(f"Distance: {distance} cm")

time.sleep(1)

This code assumes that the ultrasonic sensor's TRIG pin is connected to GPIO 23 and the ECHO pin is connected to GPIO 24 on the Raspberry Pi. You may need to adjust the pin numbers to match your specific setup.

The code uses the **measure\_distance** function to send a 10 microsecond pulse to the TRIG pin, wait for the ECHO pin to go high and low, and calculate the duration of the pulse. It then converts the pulse duration to distance in centimeters using the speed of sound in air (17150 cm/s). The function returns the distance in centimeters.

The main loop of the code calls the **measure\_distance** function once per second and prints the distance to the console. You can modify this loop to do something else with the distance value, such as controlling a robot or triggering an alarm.