2023

1. Doe-het-zelf auto back-up sensor

Projectnummer: **2021-1-FR01-KA220-SCH-000031617**

Co-funded by the European Union

SCRAPY Partnerschap

31/05/2023

Inhoud

Experiment 1: Doe-het-zelf auto back-upsensor

Korte beschrijving

Maak een achteruitrijsysteem voor auto's met Raspberry Pi Pico en ultrasone sensor.

Uitgebreide beschrijving

Vind je achteruitrijden een beetje lastig? Maak je je zorgen dat je iets of iemand raakt tijdens het achteruitrijden? Maak je geen zorgen meer! Met behulp van het Raspberry Pi Pico-bord en een HC-SR04 ultrasone sensor kun je je eigen achteruitrijradarsysteem bouwen waarmee het inparkeren van je auto een fluitje van een cent wordt.

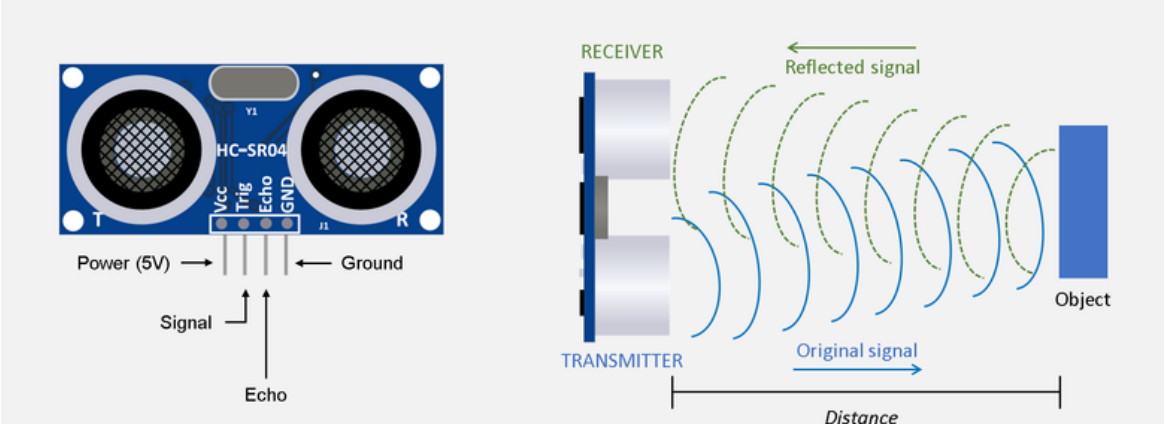
- Dit doe-het-zelf-project is perfect voor iedereen die geïnteresseerd is in elektronica en techniek, en het is geschikt voor alle niveaus. Deze achteruitrijradar zal je zeker uitdagen en verrukken.
- Naast de Raspberry Pi Pico en de HC-SR04 sensor heb je voor dit project ook een zoemer, drie LED's (groen, geel en rood), drie weerstanden van 220 ohm, een rapid prototyping bord en aansluitdraden nodig. Met deze onderdelen kun je een betrouwbare en nauwkeurige achteruitrijradar maken waarmee je gemakkelijk kunt inparkeren.
- Door de stapsgewijze instructies te volgen, leer je hoe je de componenten aansluit, het Raspberry Pi Pico-bord programmeert en het systeem test om er zeker van te zijn dat het correct werkt. Je leert ook over de fysica van ultrasone sensoren en hoe ze kunnen worden gebruikt om afstanden te meten.

De werkingsprincipes van de HC-SR04 ultrasone sensor?

De HC-SR04 ultrasone sensor zendt hoogfrequente geluidsgolven uit die onhoorbaar zijn voor menselijke oren. Deze geluidsgolven verplaatsen zich door de lucht en weerkaatsen tegen objecten op hun pad. Wanneer de geluidsgolven een object raken, worden ze teruggekaatst naar de sensor, die meet hoe lang het duurt voordat de golven terugkaatsen. Door de geluidssnelheid en de tijd die de golven nodig hebben om naar het object en terug te reizen te kennen, kan de sensor de afstand tot het object berekenen.

Dit zijn de belangrijkste stappen in het werkingsprincipe van de HC-SR04 ultrasone sensor:

- De sensor zendt een geluidsgolf uit met een hoge frequentie (meestal rond 40 kHz) vanaf de zender.
- De geluidsgolf beweegt door de lucht en weerkaatst tegen een voorwerp.
- De weerkaatste geluidsgolf wordt gedetecteerd door de ontvanger van de sensor.
- De sensor meet de tijd die de geluidsgolf nodig heeft om naar het object en terug te reizen.
- De sensor berekent de afstand tot het object op basis van de tijd die de geluidsgolf nodig had om naar het object en terug te reizen.



Over het geheel genomen is de HC-SR04 ultrasone sensor een betrouwbare en nauwkeurige manier om afstanden te meten, en wordt hij veel gebruikt in toepassingen zoals robotica, automatisering en afstandsdetectie.

Doelstellingen

Door middel van deze activiteit zal de gebruiker experimenteren met het bouwen van een omkeerradarsysteem met behulp van het Raspberry Pi Pico-bord en een HC-SR04 ultrasone sensor. De gebruiker zal kennis opdoen over:

- De fysica van ultrasone golven en hoe deze gebruikt kunnen worden om afstanden te meten.
- De basisprincipes van programmeren in Python en hoe je code schrijft om het Raspberry Pi Pico-bord aan te sturen.
- De principes van circuitontwerp en hoe je componenten samenvoegt op een rapid prototyping board om een functioneel omkeerradarsysteem te maken.

Door dit project te voltooien, krijgt de gebruiker meer inzicht in elektronica, techniek en programmeren. Ze zullen ook een praktisch en nuttig apparaat hebben dat ze kunnen gebruiken om hun auto veiliger en gemakkelijker te parkeren.

Te gebruiken materialen

- 1 x Raspberry Pi Pico
- 1 x Pico breadboard kit
- 1 x volledig breadboard
- 1 x HC-SR04 ultrasone sensor
- 1 x zoemer
- 3 x LED's (groen, geel en rood)
- 3 x 220 ohm weerstanden
- Jumper draden

Te volgen stappen

De belangrijkste stappen om het experiment met de omkeerradar uit te voeren met het Raspberry Pi Pico-bord en de HC-SR04 ultrasone sensor:

1. Sluit de HC-SR04 ultrasone sensor aan op het Raspberry Pi Pico-bord met behulp van aansluitdraden.
2. Sluit de zoemer en LED's aan op het Raspberry Pi Pico-bord met aansluitdraden en de 220 ohm weerstanden om de stroom te beperken.
3. Schrijf een Python-programma om het Raspberry Pi Pico-bord te besturen en de HC-SR04 sensor te gebruiken om afstanden te meten.
4. Programmeer het Raspberry Pi Pico-bord om de groene LED aan te zetten als er geen obstakel is, de gele LED als het obstakel binnen een bepaald bereik is en de rode LED als het obstakel te dichtbij is.
5. Programmeer het Raspberry Pi Pico-bord om de zoemer te activeren wanneer het obstakel te dichtbij is.
6. Test het achteruitrijradarsysteem door obstakels op verschillende afstanden en hoeken achter de radar te plaatsen en zorg ervoor dat de LED's en zoemer de juiste feedback geven.

Voor stap 4: Om de werking van het systeem te bestuderen, moet het gedrag worden geprogrammeerd. De afstandswaarden kunnen worden aangepast aan specifieke vereisten. De programmering moet de volgende regels volgen:

- De groene LED moet continu branden als de afstand tussen de sensor en een obstakel groter is dan 20cm (1 meter).
- Het oranje lampje moet branden als de afstand tussen de sensor en een obstakel tussen 20 cm en 5 cm is.
- Het rode lampje moet branden als de afstand tussen de sensor en een obstakel kleiner is dan 5 cm.

Door het systeem zo te programmeren dat het deze regels volgt, geeft het duidelijke visuele feedback over de afstand tussen de sensor en een obstakel.

Deze regels kunnen naar behoefte worden aangepast aan specifieke vereisten of gebruikssituaties.

Bedradingsschema

Hier is een bedradingsschema voor het radarproject met Raspberry Pi Pico-bord, HC-SR04 ultrasone sensor, zoemer en drie LED's:

Raspberry Pi Pico-bord:

- GP15: Triggerpen van de HC-SR04-sensor
- GP14: Echo pin van de HC-SR04 sensor
- GP10: Positieve pin van de groene LED

- GP11: Positieve pin van de oranje LED
- GP12: Positieve pin van de rode LED
- GP2: Positieve pin van de zoemer
- GND: Massapin van de printplaat

HC-SR04 Sensor:

- VCC: Verbinden met 5V voedingsbron
- GND: Verbinden met GND van het Raspberry Pi Pico-bord
- Trig: Verbinden met GP15 van het Raspberry Pi Pico-bord
- Echo: Verbinden met GP14 van het Raspberry Pi Pico-bord

Groene LED:

- Positieve poot: Verbinden met GP10 van het Raspberry Pi Pico-bord via een weerstand van 220 ohm
- Negatief been: Verbinden met GND van het Raspberry Pi Pico-bord

Oranje LED:

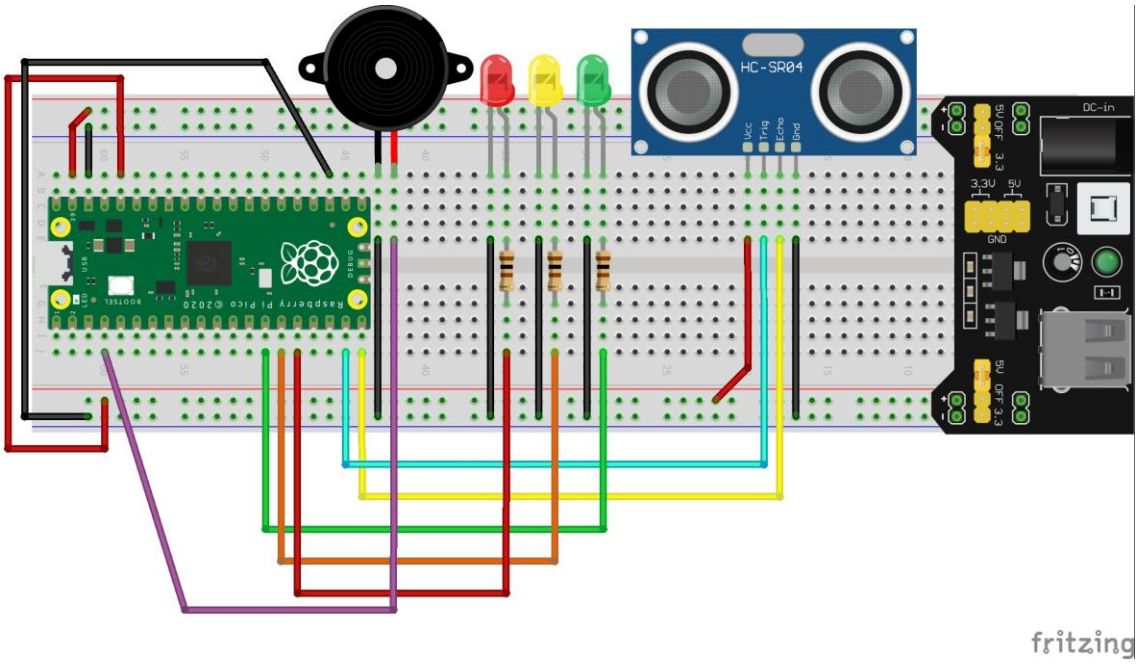
- Positieve poot: Verbinden met GP11 van het Raspberry Pi Pico-bord via een weerstand van 220 ohm
- Negatief been: Verbinden met GND van het Raspberry Pi Pico-bord

Rood LED:

- Positieve poot: Verbinden met GP12 van het Raspberry Pi Pico-bord via een weerstand van 220 ohm
- Negatief been: Verbinden met GND van het Raspberry Pi Pico-bord

Zoemer:

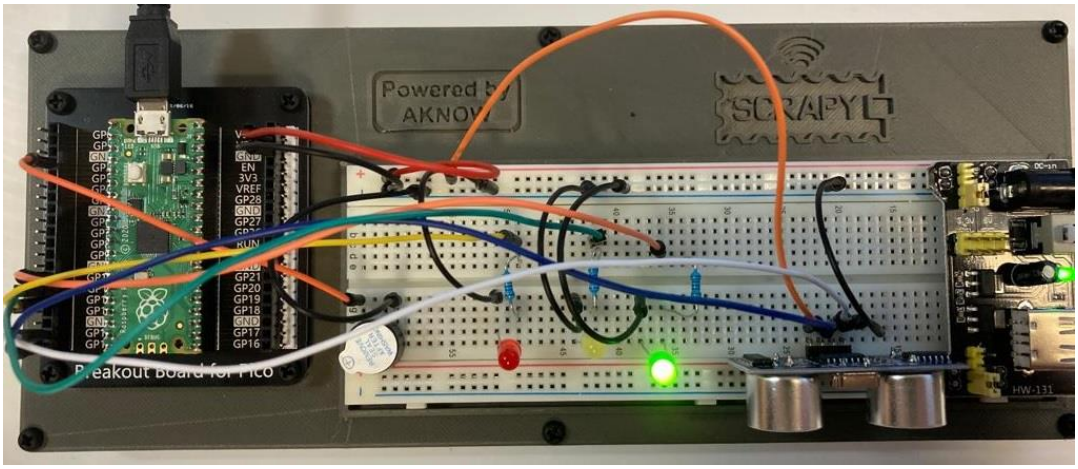
- Positieve poot: Verbinden met GP2 van Raspberry Pi Pico-bord
- Negatief been: Verbinden met GND van het Raspberry Pi Pico-bord

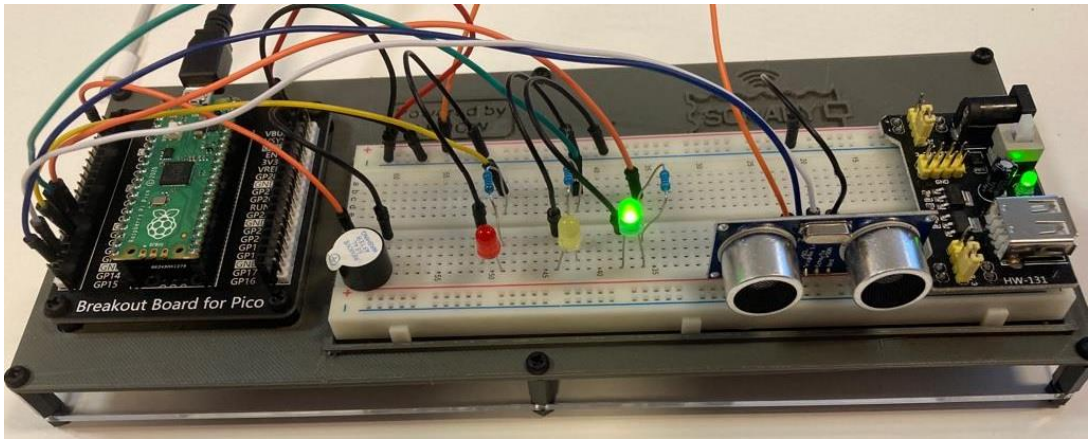


Code

```
# Import required libraries
from machine import Pin, time_pulse_us
import time
# Define pins for the components
trigger_pin = Pin(15, Pin.OUT)
echo_pin = Pin(14, Pin.IN)
green_led = Pin(10, Pin.OUT)
orange_led = Pin(11, Pin.OUT)
red_led = Pin(12, Pin.OUT)
buzzer = Pin(2, Pin.OUT)
# Define the distance thresholds for each LED
green_threshold = 20  # cm
orange_threshold = 5 # cm
# Define the function to calculate the distance from the HC-SR04 sensor
def get_distance():
    # Send a 10us pulse to trigger the sensor
    trigger_pin.low()
    time.sleep_us(2)
    trigger_pin.high()
    time.sleep_us(10)
    trigger_pin.low()
    # Measure the duration of the echo signal
    duration = time_pulse_us(echo_pin, 1, 10000)
    # Calculate the distance from the duration using the speed of sound (343
m/s)
    distance = duration / 2 / 1000000 * 343 * 100
    return distance
# Define the main loop to read the distance and control the LEDs and buzzer
while True:
    # Get the distance from the sensor and print it
    distance = get_distance()
    print(f"Distance : {distance} cm")
    # Turn on the green LED if the distance is greater than the threshold
    if distance > green_threshold:
        green_led.on()
        orange_led.off()
        red_led.off()
        buzzer.off()
    # Turn on the orange LED if the distance is between the thresholds
    elif distance > orange_threshold:
        green_led.off()
        orange_led.on()
        red_led.off()
        buzzer.off()
    # Turn on the red LED and buzzer if the distance is less than the threshold
    else:
        green_led.off()
        orange_led.off()
        red_led.on()
        buzzer.on()
        time.sleep(0.5)  # Buzz for 0.5 seconds
    # Wait for 0.1 second before the next measurement
          time.sleep(0.1)
```

Voorbeeldfoto's





Conclusie

Samenvattend: dit project bestond uit het gebruik van een Raspberry Pi Pico-bord, een HC-SR04 ultrasone sensor en enkele extra onderdelen om een achteruitrijradar voor auto's te maken. We leerden over de werkingsprincipes van de HC-SR04 sensor, hoe we de componenten met elkaar konden verbinden en hoe we het gedrag van het systeem konden programmeren met MicroPython. Door de stappen in deze tutorial te volgen, konden we een systeem maken dat obstakels kan detecteren en de bestuurder visuele en hoorbare feedback kan geven.

Er zijn veel manieren om dit project verder uit te breiden. Enkele suggesties zijn:

- Meer LED's of een display toevoegen om meer gedetailleerde afstandsinformatie te geven.

- Algoritmen voor machinaal leren gebruiken om obstakeldetectie en nauwkeurigheid te verbeteren.
- Een draadloze interface maken zodat het systeem kan communiceren met een mobiel apparaat of een ander extern apparaat.
- Extra sensoren of componenten toevoegen om een uitgebreider voertuigveiligheidssysteem te maken.

Over het geheel genomen biedt dit project een geweldige introductie in de wereld van elektronica en programmeren en dient het als startpunt voor verdere verkenningen en experimenten.