

# 3 Afstand meten

## 3.1 Einddoel

Je gaat een apparaat maken dat de omgeving scant en op het scherm toont waar er objecten staan in de omgeving.

Als je klaar bent met de opdracht lever je in It's learning in:

- De definitieve python code.
- Een mp4-filmpje met een demo van je programma.

Als dit gedaan is laat je de opdracht zien aan de docent zodat deze de opdracht kan goedkeuren.

## 3.2 Kennis

Voor deze opdracht heb je kennis nodig van microPython.

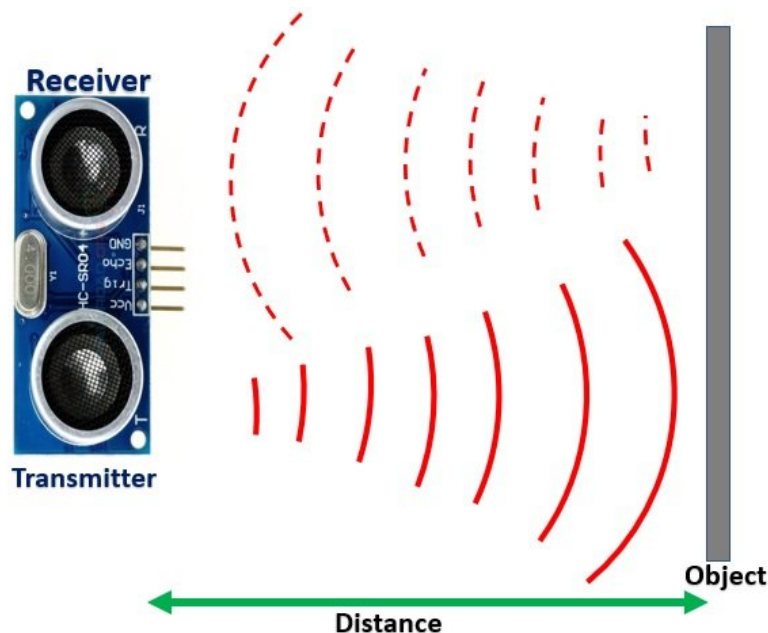
## 3.3 Benodigdheden

Voor deze opdracht heb je de onderstaande materialen nodig. Controleer aan het begin of al deze spullen aanwezig zijn. Bij het opruimen dien je weer te controleren of alles aanwezig is. Indien er iets defect is geraakt moet je de docent op de hoogte brengen.

- Raspberry Pi Pico – H
- UBS-usb mini kabel
- Ultrasoon sensor HC-SR04
- Micro servomotor SG90
- 

## 3.4 Opdracht

Met de HC-SR04 kunnen we door middel van geluidsgolven de afstand bepalen.

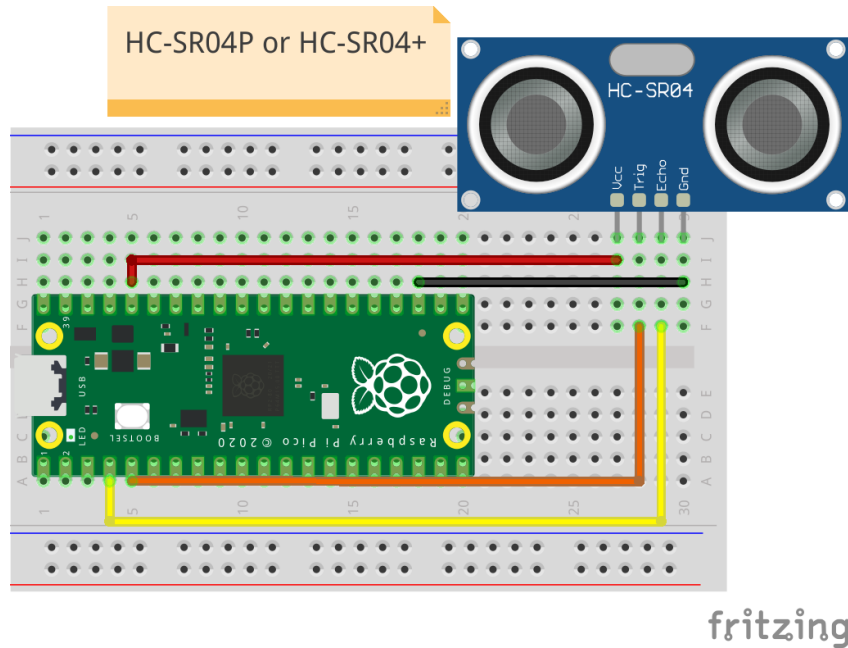


Een ultrasoon sensor meet de tijd dat een signaal is verzonden totdat deze is ontvangen. Dit is dus tweemaal de afstand die we uiteindelijk moeten hebben. De snelheid van geluid door lucht bij 20°C is 340 m/s. Met deze informatie kan de afstand berekend worden.

De tijd die we meten is in microseconden, 1 microseconden 1  $\mu\text{s}$  = 0.000001 seconden.

$$2s = v * t \rightarrow s = \frac{340 \text{ in } \frac{m}{s}}{2} * t \text{ in } s = 170 * t = 0.17 \left[ \frac{mm}{\mu s} \right] * t [\mu s]$$

We sluiten de ultrasoon sensor als volgt aan. (De Vcc staat hieronder op 3.3V, dit zou beter op de Vbus aangesloten kunnen worden.)



De berekening die we hierboven hebben gezien hoeven we niet zelf te programmeren. We gaan hiervoor een librarie gebruiken. We kunnen deze van GitHub downloaden:

<https://github.com/rsc1975/micropython-hcsr04>

Product Solutions Open Source Pricing

rsc1975 / micropython-hcsr04
Public

Code Issues Pull requests Actions Projects Security Insights

master 2 branches 1 tag
Go to file Code

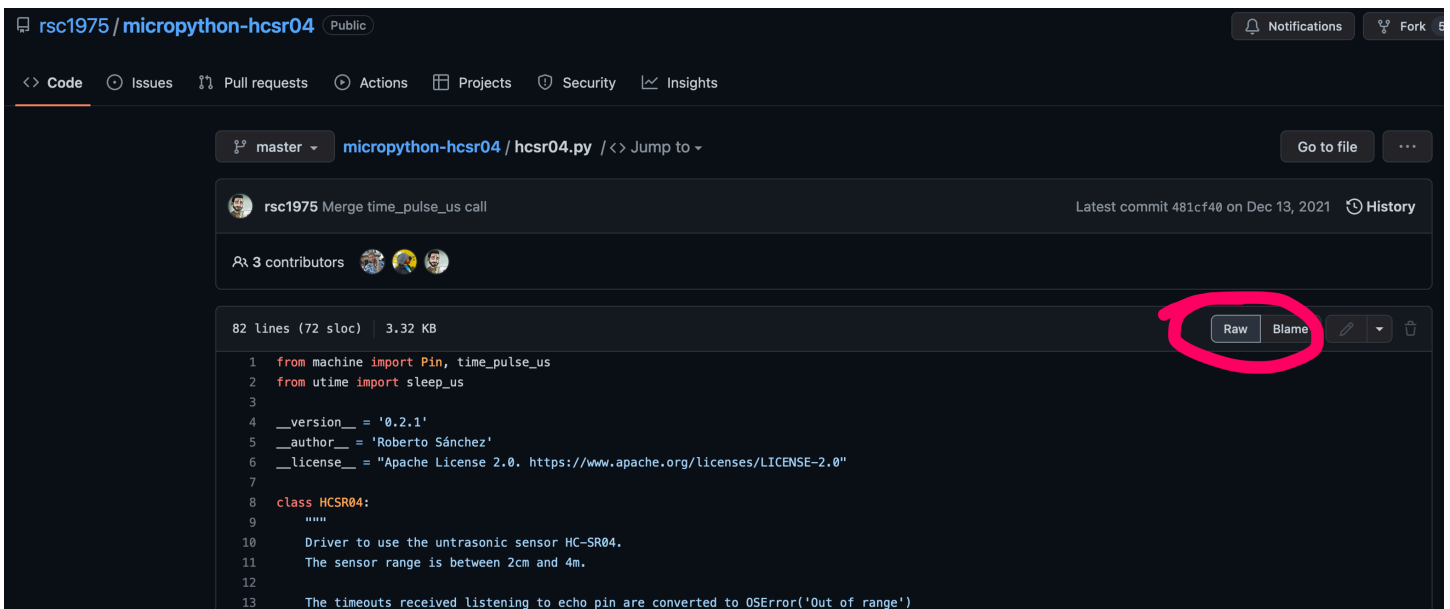
rsc1975 Merge branch 'jfdona23-tweak-imported-libraries'
17eafe3 on Dec 13, 2021 14 commits

LICENSE	Initial commit	6 years ago
README.md	Fixed broken link to sensor datasheet	4 years ago
hcsr04.py	Merge time_pulse_us call	11 months ago

README.md

## HC-SR04 Sensor driver in micropython

Micropython driver for the well-known untrasonic sensor [HC-SR04](#)



Het bestand moet op je Pico opgeslagen worden met de naam hcsr04.py

Met de onderstaande code kan meet je de afstand. Let op dat de correcte trigger en echo pinnen nog ingevoerd moeten worden.

```
1 from machine import Pin
2 from hcsr04 import HCSR04
3 from time import sleep
4
5 SLEEP = 1
6 TRIGGER_PIN = 21
7 PIN_ECHO = 20
8
9 _sensor = HCSR04(trigger_pin=TRIGGER_PIN, echo_pin=PIN_ECHO, echo_timeout_us=10000)
10
11 def measure():
12     return _sensor.distance_cm()
13
14 if __name__ == "__main__":
15     while True:
16         print('Distance:', measure(), 'cm', '|')
17         time.sleep(SLEEP)
18
```

### 3.4.1 Opdracht 1: Afstand weergeven met Leds

Voor iedere 4 cm dat een voorwerp verwijderd is gaat er een led uit op het break-out board. Dus als een voorwerp op 4cm wordt gemeten gaat er 15 led aan. Als het voorwerp op 8 cm wordt gemeten gaan er 14 leds aan. Wordt een voorwerp op 60 cm of meer gemeten dan gaan er geen leds aan.

Als een voorwerp binnen de 4cm is zal de sensor deze niet detecteren. Er zullen dan geen leds aangaan.

### 3.4.2 Opdracht 2: Servomotor en afstandsmeting

We laten een servomotor steeds van 0 naar 180 graden in stapjes van 5 graden draaien, en weer terug. Bij iedere stap meten we de afstand naar een eventueel voorwerp. De gemeten afstanden en de hoek worden op het scherm getoond. Op de leds kan je weer zien of er een object gedetecteerd wordt.

Met de onderstaande code kan je een servomotor waarvan de data pin op 28 is aangesloten draaien naar 90 graden.

```
1 import machine
2 from time import sleep
3
4 class Servo:
5     def __init__(self, MIN_DUTY=300000, MAX_DUTY=2300000, pin=0, freq=50):
6         self.pwm = machine.PWM(machine.Pin(pin))
7         self.pwm.freq(freq)
8         self.MIN_DUTY = MIN_DUTY
9         self.MAX_DUTY = MAX_DUTY
10
11     def rotateDeg(self, deg):
12         deg = max(0, min(deg, 180))
13         duty_ns = int(self.MAX_DUTY - deg * (self.MAX_DUTY - self.MIN_DUTY) / 180)
14         self.pwm.duty_ns(duty_ns)
15
16 servo = Servo(pin=28)
17 servo.rotateDeg(90)
```

### 3.4.3 Opdracht 3: Knoppen gebruiken

De hoek van de servomotor is in de vorige opdracht steeds in stappen ingesteld. Verander de code zodat je met 2 drukknoppen de servomotor of maar links of naar rechts kan sturen.