5 Temperatuur meten

5.1 Einddoel

Je gaat de temperatuur meten in een warmhoud bak en erbuiten.

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

- De definitieve python code waar de verschillende opdrachten in verschillende functies staan.
- Een foto van de opstelling waarop duidelijk te zien is hoe alles in aangesloten.
- 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.

5.2 Kennis

Voor deze opdracht heb je kennis nodig van microPython.

5.3 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
- 2X TC74A?
- Ventilator
- 2x dht22

5.4 Voorbeeldcode en libraries

In It's Learning kan me de onderstaande voorbeeldcode vinden om je op weg te helpen.

5.5 Opdracht

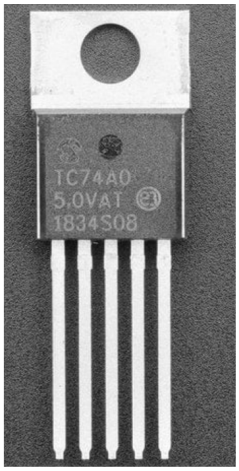
We gaan met 4 sensoren de temperatuur in een yoghurt-maker warmhoud bak meten. In de bak en buiten de bak zit een temperatuur sensor van de TC74A_ en een dht22 sensor. Deze sensoren worden op een andere manier uitgelezen. We gaan een opstelling maken, en er na meerdere keren meten. Er is ook een vennetje in de deksel van de bak die we in de ene meting wel gebruiken, en in de volgende meting niet.

De gemeten waarden worden met een tijd in een bestand op de pico opgeslagen. Dit bestand wordt aan het einde van de pico afgehaald zodat we het in Excel kunnen importeren en er een grafiek van kunnen maken.

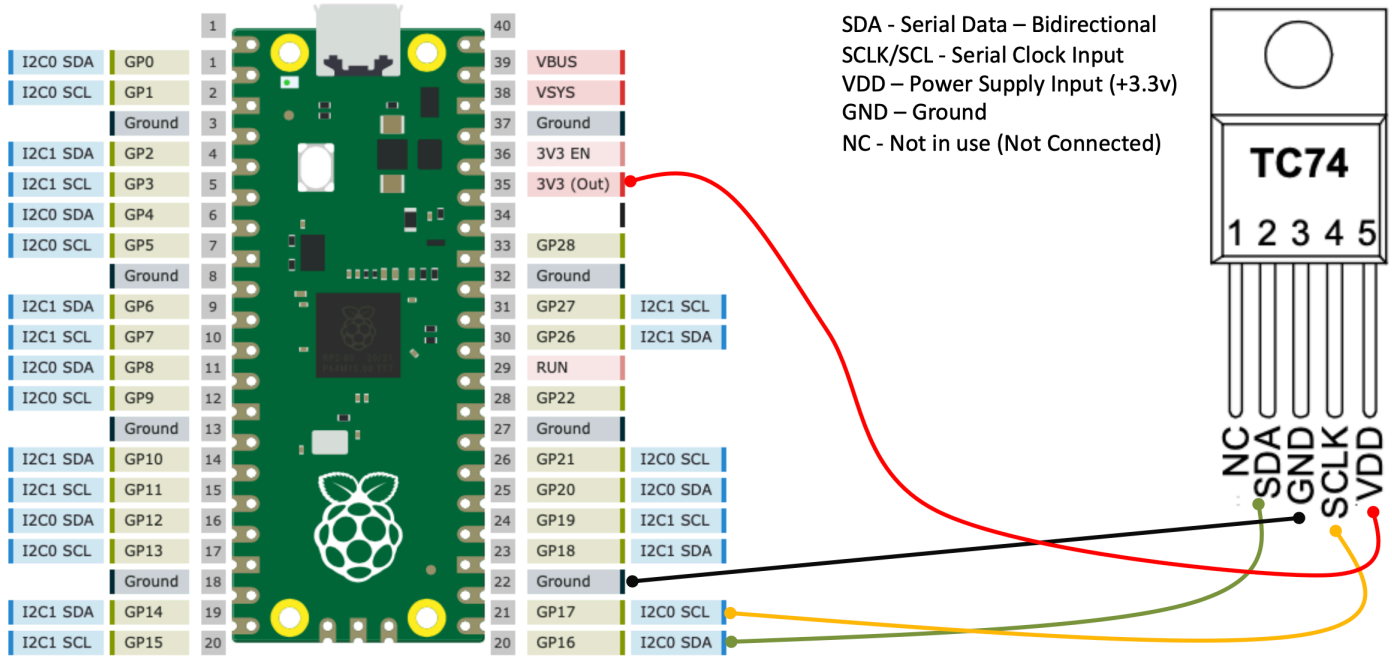
5.5.1 TC74A_

De TC74A_ temperatuur sensoren waarmee we praten met i2c. Dit is een serieel communicatieprotocol. Een serieel communicatieprotocol heeft als kracht dat je tot 256 apparaten kan aansturen met slechts 2 GP-pinnen op je microcontroller. We hoeven alleen de GND en VCC aan te sluiten op respectievelijk GND en 3.3V en we gebruiken de SCL en SDA voor data. Als we naar de Pico kijken zien we in de blauwe vakjes dat dit i2c ook daar ondersteund wordt.

```
from machine import I2C, Pin

i2c = I2C(id = 0, sda=Pin(0), scl=Pin(1))
devices = i2c.scan()

print('Scan i2c bus...')
if len(devices) == 0:
  print("No i2c device !")
else:
  print('i2c devices found:',len(devices))

  for device in devices:
    print("Decimal address: ",device," | Hexa address: ",hex(device))

```



Iedere sensor moet een unieke naam hebben. Het is daarom aan te raden om eerst van de ene sensor het adres van op te vragen, en daarna de andere sensor. De verschillende sensoren kunnen op dezelfde twee poorten aangesloten worden en met de twee adressen weten de sensoren wie de temperatuur moet doorsturen.



Met de onderstaande code kan je de temperatuur van een sensor uitlezen.

```
from machine import Pin, I2C
from time import sleep

i2c_interface = 0

sdapin = Pin(16)
sclpin = Pin(17)

i2c =I2C(i2c_interface, scl=sclpin, sda=sdapin, freq=100000)

tc74address = 0x48

while True:
    data = i2c.readfrom(tc74address, 1, True)
    temp = int.from_bytes(data, "big")
    print(temp)
    sleep(5)
```

hell ×

```
>> %Run -c $EDITOR_CONTENT

27
27
27
28
```

5.5.2 Dht11

In it's learning kan je onderstaande code vinden. Met de methode measure lees je de temperatuur uit van de DHT11. Maak de code zo dat je beide sensoren uitleest.

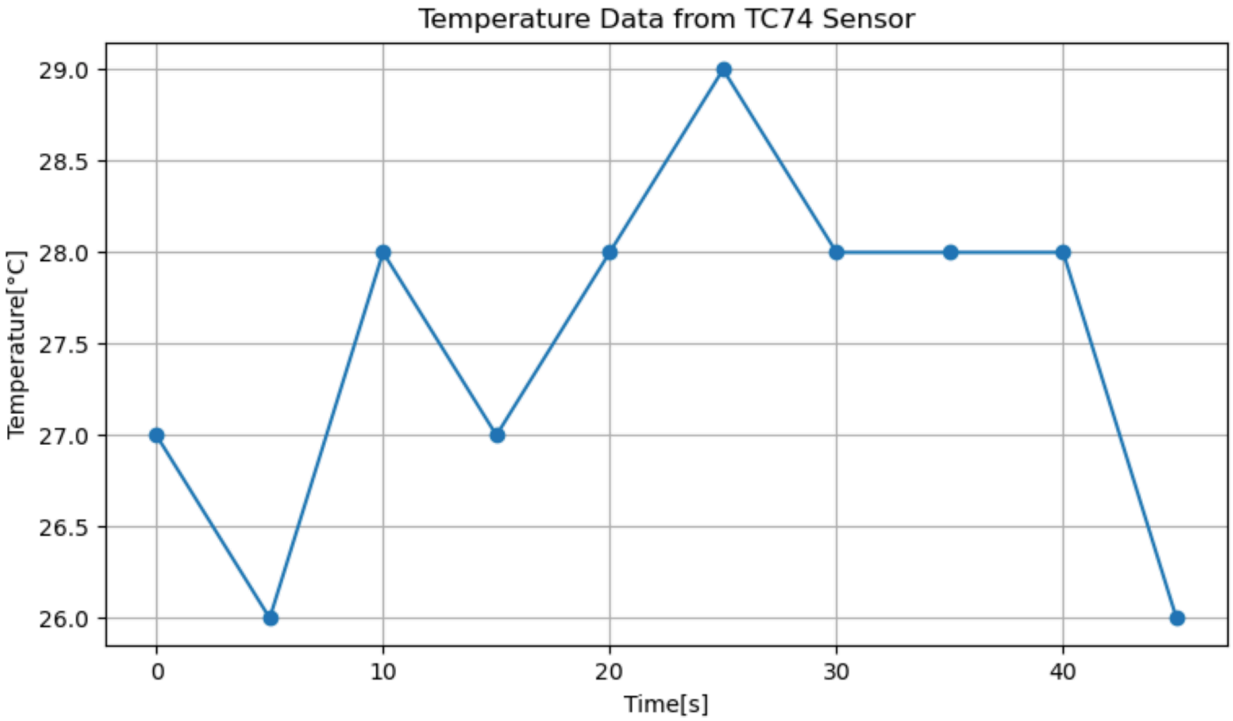
```
# credits to https://github.com/ikornaselur/pico-libs/tree/master/src/dht11

import array
import micropython
import utime
from machine import Pin
from micropython import const

class InvalidChecksum(Exception):
    pass

class InvalidPulseCount(Exception):
    pass

MAX_UNCHANGED = const(100)
MIN_INTERVAL_US = const(200000)
HIGH_LEVEL = const(50)
EXPECTED_PULSES = const(84)

class DHT11:
    _temperature: float
    _humidity: float

    def __init__(self, pin):
        self._pin = pin
        self._last_measure = utime.ticks_us()
        self._temperature = -1
        self._humidity = -1
```

5.5.3 Data opslaan op een pico

```
from machine import Pin, I2C
from time import sleep
#I2C Initialization
tc74address = 0x48
i2c_interface = 0
sdapin = Pin(16)
sclpin = Pin(17)

i2c =I2C(i2c_interface, scl=sclpin, sda=sdapin, freq=100000)

# Open File
file = open("tempdata.txt", "w")

# Write Data to File Function
def writefiledata(time, value):
    file.write(f"{time;{round(value, 2)}\n")

counter=0
SLEEP = 5 # sleep 5 seconds

while True:
    data = i2c.readfrom(tc74address, 1, True) temp = int.from_bytes(data, "big") print(temp)
    writefiledata(k*SLEEP, temp)
    counter += 1
    sleep(SLEEP)

file.close()
```

5.5.4 Data tonen in grafiek

```
import matplotlib.pyplot as plt
# Open File
f = open("tempdata.txt", "r")
Data Analysis Example
# Transform File Data into x Array and y Array that can be used for plotting x = []
y = []
k=0
for record in f:
    record = record.replace("\n", "")
    record = record.split(";")
    x.append(int(record[0]))
    y.append(int(record[1]))
    k=k+1
f.close()

plt.plot(x,y, '-o')
plt.title('Temperature Data from TC74 Sensor')
plt.xlabel('Time[s]')
plt.ylabel('Temperature[°C]')
plt.grid()
plt.show()
```


Temperature Data from TC74 Sensor
Temperature[°C]
29.0
28.5
28.0
27.5
27.0
26.5
26.0
0
10
20
30
40
Time[s]