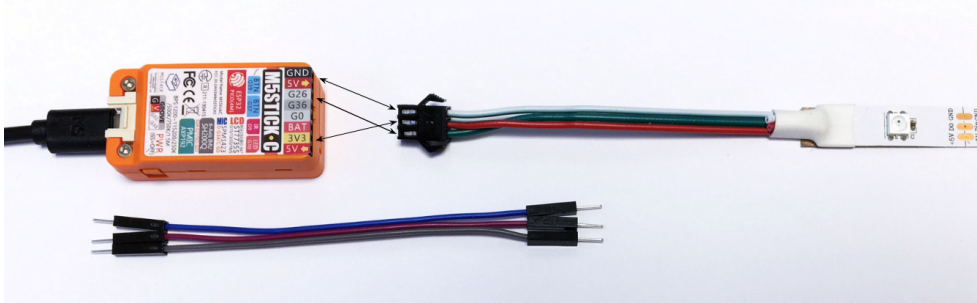


Tilslut LED-strip

Tilslut jeres LED-strip som vist på billedet. Brug et mellemlid (3 x jumper kabel).



Forbindelserne er:

Type	Microcontroller	LED-strip ledning
Jord	GND	Hvid
Strøm (3,3 volt)	3V3	Rød
Data / kontrolsignal	G26	Grøn

Første MicroPython program

Åbn Mu-editoren. Opret en fil "intro.py" med dette indhold:


```
import machine
import tinkerlib

# 30 LED'er tilsluttet pin 26
ledstrip = tinkerlib.LEDStrip(machine.Pin(26), 30)

# Indstil LED'ernes farver
ledstrip[0] = (255, 0, 0)
ledstrip[9] = (0, 0, 255)

# Opdater LED'erne ved at kalde ledstrip.write()
ledstrip.write()
```

Afprøv programmet

- Tryk på -knappen i Mu for at køre programmet på Microcontrolleren.
- Tjek, at den første diode lyser rødt (diode 0) og den tiende lyser blå (diode 9).

Øvelse

- Udvid programmet, så hver anden diode farves rød og hver anden farves blå for de første 10 dioder.

Navngivning

Vi kan gøre programmet nemmere at læse, ved at give navne til farvekoderne. Navngiv farverne i intro.py:

```
# Opret nye navne 'red' og 'blue'
red = (255, 0, 0)
blue = (0, 0, 255)

# Brug navne i koden - det gør det nemt at læse og ændre
ledstrip[0] = red
ledstrip[1] = blue
ledstrip[2] = red
...
ledstrip[9] = blue
ledstrip.write()
```

Animationer

For at lave animationer skal vi bruge funktionen `sleep_ms` fra biblioteket `time`.

```
import machine
import tinkerlib
import time
ledstrip = tinkerlib.LEDStrip(machine.Pin(26), 30)

# Tænd den første diode
ledstrip[0] = (255, 0, 0)
ledstrip.write()
time.sleep_ms(200) # Vent 200 millisekunder

# Tænd den næste diode
ledstrip[1] = (0, 0, 255)
ledstrip.write()
time.sleep_ms(200) # Vent 200 millisekunder

# Fortsæt selv ...
```

Øvelser

- Fortsæt mønsteret for de første 10 LED'er.
- Lav en variabel til at styre hastigheden (i stedet for at gentage „200“)
- En LED slukkes ved at sætte den til (0, 0, 0). Sluk den forrige LED i hvert trin, så der kun er en LED tændt ad gangen.

Flere farver

Her er nogle flere farver I kan bruge. Lav fx en regnbue, en istap, en animation af ild, eller noget andet i selv finder på.

Rød	(255, 0, 0)	Gul	(255, 255, 0)
Grøn	(0, 255, 0)	Lilla	(127, 0, 255)
Blå	(0, 0, 255)	Tyrkis	(0, 255, 255)

Flere muligheder

I kan også gøre brug af funktionerne `ledstrip.fill(farve)`, `ledstrip.clear()` eller `ledstrip.fillN(farve, antal)`, der hhv. tænder alle, slukker alle eller tænder et specifikt antal LED'er.

Tilslut fugtighedssensor

Fugtighedssensoren, tilsluttes via et specielt stik, ved siden af USB-C stikket. Farverne på de fire ledninger har betydning, se tabellen nedenfor.

Type	Ledningsfarve	Fugtighedssensor ben
Jord	Sort	-
Strøm (3,3 volt)	Rød	+
Data / kontrolsignal	Gul	out



Vigtigt: Den må ikke sættes forkert på, tjek evt. farverne en ekstra gang. (Den hvide ledning bruges ikke og kan evt. klippes af.)


Måling

Opret en ny fil i Mu-editoren og gem den som „measure.py“. Indtast følgende program for at lave en måling med sensoren:

```
import machine
import dht

# Fugtighedssensor tilsluttet via pin 33 (gul ledning)
sensor = dht.DHT11(machine.Pin(33))

sensor.measure()           # 1) Foretag måling
luftfugtighed = sensor.humidity() # 2) Gem luftfugtighed i variabel
print(luftfugtighed)       # 3) Vis den aflæste værdi
```

Afprøv programmet (tryk på ). Der bør blive vist et tal mellem 0 og 100 i konsollen (formentlig omkring 40-55). Det er luftfugtigheden i procent.

Fejlsøgning

Hvis I får fejlmeddelelsen ETIMEDOUT, kan det være tegn på tre forskellige ting:

- Kablerne er sat forkert i.
- Stikket er ikke helt trykket i bund. Det skal sige klik når stikket sættes i micro-controlleren.
- Sensoren er langsom, og I har kørt programmet for mange gange hurtigt efter hinanden.

Gentag måling


Vi skal nu ændre programmet, så det bliver ved med at aflæse værdierne. Ændr programmet til at bruge en „while True“, som nedenfor, det gentager programmet igen og igen. Det er vigtigt at linjerne under „while True:“ bliver rykket ind som vist.

Hold en pause (time.sleep_ms) mellem hver måling, sensoren er lidt langsom.

```
import machine
import dht
import time

# Fugtighedssensor tilsluttet via pin 33 (gul ledning)
sensor = dht.DHT11(machine.Pin(33))

while True:
    sensor.measure()           # 1) Foretag måling
    luftfugtighed = sensor.humidity() # 2) Gem luftfugtighed i variabel
    print(luftfugtighed)      # 3) Vis den aflæste værdi
    time.sleep_ms(1000)       # 4) Vent 1 sekund
```

Afprøv programmet (tryk på )

Fugtighed og LED-strip


Tilføj i starten af filen:

```
import tinkerlib
ledstrip = tinkerlib.LEDStrip(machine.Pin(26), 30)
red = (100, 0, 0)
green = (0, 100, 0)
```

Tilføj inde i „while True:“ konstruktionen (husk fire mellemrum før hver linje):

```
color = green
if luftfugtighed > 80:
    color = red # Udføres kun når luftfugtigheden er > 80%

# Tænd 8 LED'er i den valgte farve
ledstrip.fillN(color, 8)
```

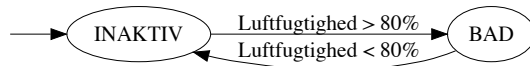
Afprøv programmet (tryk på )

Opgaver

- Ændr ottetallet til beregningen `luftfugtighed*3/10`, hvad sker der?

Tilstande

For at kunne tage tid på længden af et brusebad, skal vi først holde øje med tilstanden, som skifter mellem „INAKTIV“ og „BAD“, afhængig af luftfugtigheden:



Tilføj først en ny variabel ovenover „while“

```
tilstand = "INAKTIV"
```

```
while True:
    # ... samme som før ...
```

Erstat dernæst „if-else“ blokken fra før, med følgende der ikke ændrer på LED-strippen, men bare skifter tilstand:

```
if luftfugtighed > 80 and tilstand == "INAKTIV":
    tilstand = "BAD"
```

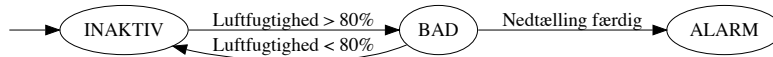
```
if luftfugtighed <= 80 and tilstand == "BAD":
    tilstand = "INAKTIV"
```

Opgave

- Brug `print()` til at følge med i om tilstanden ændrer sig, som den skal

Tidtagning og nedtælling

Vi skal udvide med endnu en tilstand, så vi kan lave en alarm der går af efter 3 minutter.



For at tælle hvor lang tid der er gået, skal vi bruge en ny variabel til at holde styr på tiden med.

Tilføj en variabel `timer` ovenover „while“:

```
timer = 0
```

Inde i `while`-løkken begynder vi så at tælle op. Hver gang hele `while`-løkken er blevet kørt er der gået cirka et sekund, pga. vores `time.sleep_ms{1000}`.

```
if tilstand == "BAD":
    timer = timer + 1
```

For at skifte over til alarm tilstanden, skal vi nu bare tilføje endnu en `if`:

```
if timer > 180 and tilstand == "BAD":
    tilstand = "ALARM"
```

Prøv det af

- Hvornår skal den skifte tilbage fra „ALARM“?
- Husk at sætte timeren tilbage til 0 på et passende sted

Alarm på LED-strip

- Brug print til at følge med i værdien af timeren
- Gør så LED-strippen viser hvor mange sekunder der er gået
- Gør så LED-strippen viser hvor mange minutter der er gået
- Gør så LED-strippen tæller ned fra 5, når et bad starter (spørg evt. om et hint)

Brug af højttaler

Erstat LED-strippen med højttaler og lav en ny fil `speaker_demo.py`, hvor I indtaster:

```
import hat
speaker = hat.get(hat.SPEAKER)

speaker.tone(1800, 200)
```

Det første tal er frekvensen, det andet tal længden af tonen i millisekunder.

Man kan desværre ikke have LED-strip og højttaler tilsluttet samtidig.

**Alarm**

- Gå tilbage til jeres `measure.py`-program, og indsæt kode så højttaleren lyder, når tilstanden er "ALARM".

Oprettelse på Adafruit

Åbn hjemmesiden `io.adafruit.com`, og opret en bruger ved at trykke på:

[Get Started for Free](#)

Udfyld navn, email, brugernavn og password og tryk på „Create account“.

FIRST NAME	<input type="text" value="Martin"/>
LAST NAME	<input type="text" value="Dybdal"/>
EMAIL	<input type="text" value="dybber@di.ku.dk"/>
USERNAME	<input type="text" value="dybber"/>
<small>Username is viewable to the public on the forums, Adafruit IO, and elsewhere.</small>	
PASSWORD	<input type="password" value="*****"/>
<input type="button" value="CREATE ACCOUNT"/>	

Opret feed

Gå tilbage til `io.adafruit.com`. Vi skal oprette et *feed* vi kan sende vores luftfugtigheds-data til.

- Tryk på følgende fire knapper i rækkefølge:



- Indtast et navn, fx *humidity* og tryk „Create“.

Forbind til WiFi fra microcontroller

Opret en ny fil med følgende indhold og gem som „`adafruit_send.py`“:

```
import time
import wifi
import requests
import json

wifi.connect("DIKU1", "PeterNaur")
```

Når I kører filen, burde den skrive følgende (og et par andre linjer):

```
Connecting to WiFi network...
Successfully connected to "DIKU1".
```

Indtast brugernavn og AIO key

Vi skal bruge jeres brugernavn og en speciel 'nøgle' for at logge på `io.adafruit.com` fra vores program.

Opret to variabler `username` og `AIO_KEY`, ved at tilføje følgende:

```
username = "JERES BRUGERNAVN"
AIO_KEY = "JERES NØGLE"
```

I finder jeres *AIO Key* ved at trykke på **AIO Key** inde på Adafruit-hjemmesiden. Kopier indholdet af feltet „Active Key“. Der kan fx stå `"aio_6f1752f7441f8b01fbd54b949e3b"`. Det sætter I ind i stedet for `"JERES NØGLE"`.

Send data til Adafruit IO

Næste trin er at få sendt noget data afsted. Til det skal I bruge følgende funktion:

```
def send_to_feed(feed, value):  
    url = "https://io.adafruit.com/api/v2/"  
    url = url + username + "/feeds/" + feed + "/data/"  
    headers = {'X-AIO-Key': AIO_KEY}  
    data = { "value" : value }  
    r = requests.post(url, headers=headers, json=data)  
    r.close()
```

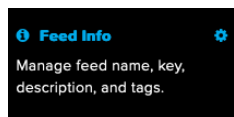
(Kan kopieres fra dette link: kortlink.dk/24mhr)


For at sende en værdi, fx tallet 17, skriver I:

```
send_to_feed("NAVNET PÅ JERES FEED", 17)
```

I skal bruge navnet på jeres feed, som I oprettede tidligere. Hvis det ikke virker, så prøv følgende:

- Find i stedet jeres *Feed key* ved at trykke på:



Hvis I ikke kan finde det, skal I måske først åbne „hamburger“-menuen: 

Tjek at det virker

- Åbn io.adafruit.com og klik videre til jeres feed. ("Feeds-> "Humidity", fx)
- Tjek at grafen viser tallet 17

Send luftfugtighedsdata

Nu er det jeres opgave at kombinere de to dele, så data'en I aflæser fra jeres fugtighedssensorer, bliver sendt til jeres feed på Adafruit.

Kopier de relevante dele af `adafruit_send.py` over i `measure.py`

- Import af modulerne `wifi`, `requests`, `json`
- Kode der forbinder til WiFi
- Definition af `username` og `AIO_KEY`
- Funktionen `send_to_feed`

Overfør program til microcontroller

Programmet kører kun så længe microcontrolleren sidder i computeren.

For at overføre programmet skal I omdøbe det til „`main.py`“, og bruge „Files“ menuen i Mu, hvor I skal trække „`main.py`“ fra HØJRE side til venstre side.

Genstart microcontrolleren ved at holde knappen på siden, nærmest USB-porten inde i 6 sekunder.