Kapittel 21

Programmering av praktisk utstyr

21.1 Innledning

Hvis du (eller dine elever eller studenter, hvis du har noen) vil bruke Python i praktiske eksperimenter, kan Microbit og/eller Raspberry Pi være verdt å vurdere. Både Microbit og Raspberry Pi er elektroniske kort som inneholder en mikroprosessor som kan kjøre Python-kode (Microbit kan riktignok kun kjøre MicroPython, som er en forenklet utgave av Python). Microbit har flere sensorer på selve kortet, mens Raspberry Pi ikke har noen. Til både Microbit og Raspberry Pi fins tilleggsutstyr med sensorer og aktuatorer.

Vi gir en kort beskrivelse og noen eksempler på bruk av Microbit og Raspberry Pi i dette kapitlet. Det fins veldig mye informasjon, inkl. prosjekter med kodeeksempler, på de respektive hjemmesidene.

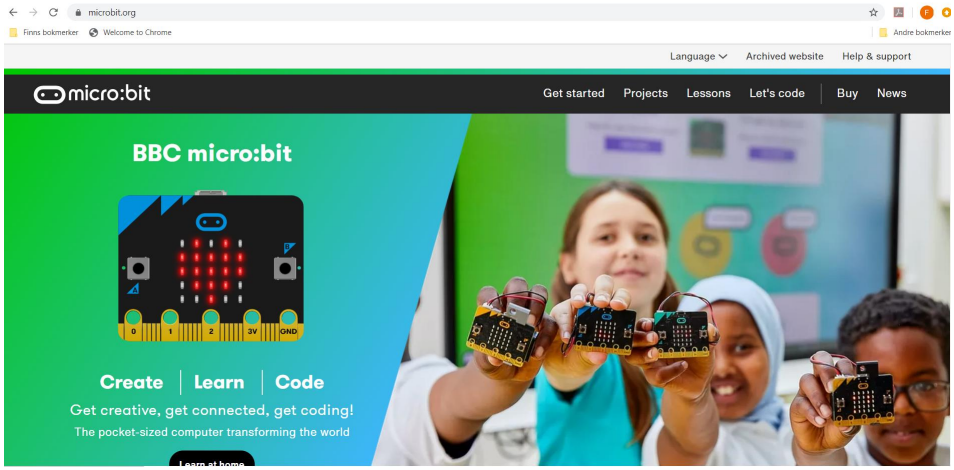
21.2 Microbit

21.2.1 Innledning

Microbit, som også skrives Micro Bit og micro:bit, er et elektronikkort med en mikroprosessor for praktiske aktiviteter i undervisning særlig i grunnskolen. Hensikten med Microbit er å motivere for programmering ved at elever og andre opplever at programmer kan brukes til å manipulere (styre) og monitorere (måle på) praktiske systemer, eller med enklere ord: virke sammen med den praktiske virkeligheten. Microbit gir også økt forståelse og interesse for hardware (maskinvare og fysiske komponenter) innen datateknologi.

Microbit er utviklet av BBC (England) og ble klar til bruk i 2015. Figur 21.1 viser Microbits hjemmeside.

På Microbits hjemmeside fins en omfattende samling av prosjekter inkl. oppskrifter for programmeringen.



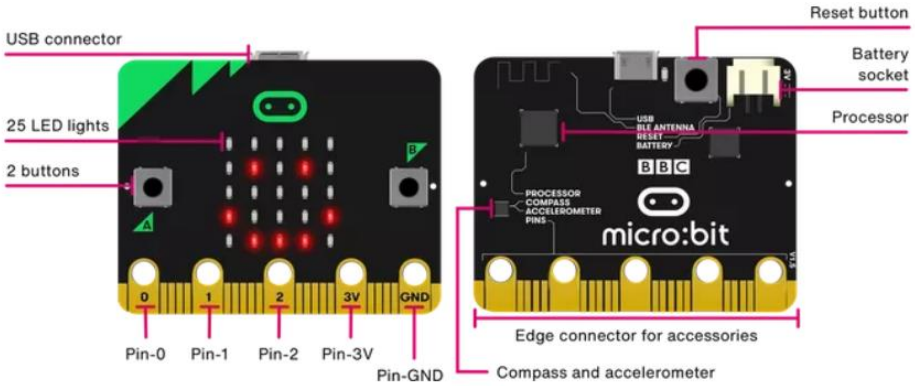
Figur 21.1: Microbits hjemmeside på http://microbit.org.

Det fins mange forhandlere av Microbit, f.eks. Skolehuset, Kjell & Company, Elfa Distrelec, RS, Komplett, n00b.

21.2.2 Mikrobeskrivelse av Microbit

21.2.2.1 Oversikt

Figur 21.2 gir en oversikt over Microbit-kortet (dets to sider). Kortet har dimensjon 52 x 43 mm.



Figur 21.2: Microbit-kortet. (Kilde: http://microbit.org.)

21.2.2.2 Programmeringsspråk

De offisielle programmeringsspråkene for Microbit er:

- MakeCode – et grafisk språk utviklet av Microsoft
- MicroPython – et tekstbasert språk som er en forenklet utgave av Python

På Microbits hjemmeside på http://microbit.org fins webbaserte editorer for MakeCode (der det også fins en fane for tilsvarende JavScript-kode) og MicroPython. Det er muligheter for programmering også med andre språk, se https://microbit.org/code.

Siden denne boken handler om Python, konsentrere vi oss om programmering med MicroPython.

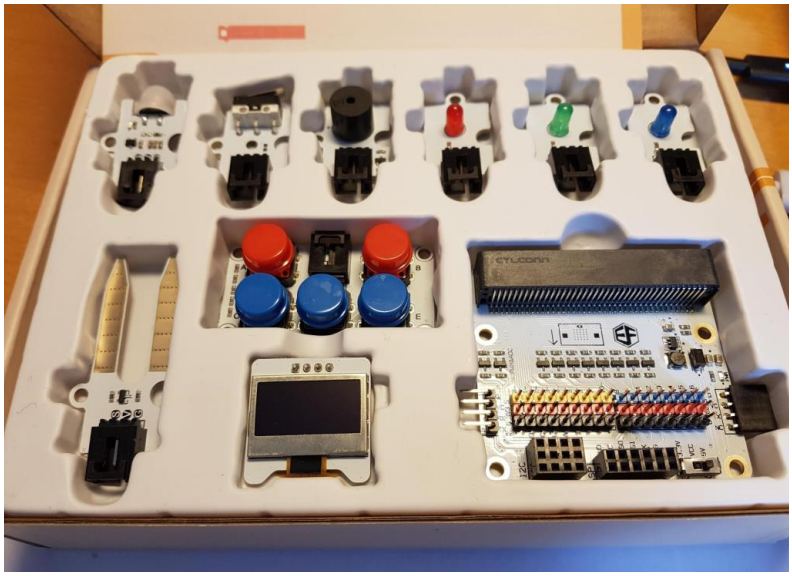
21.2.2.3 Maskinvare

Microbits maskinvare omfatter:

- Innebygde inngangselementer (engelsk: inputs):
  - Trykknapper (2 stk.)
  - Akselerometer
  - Kompass
  - Temperatursensor (på mikroprosessoren)
- Innebygde utgangselementer (outputs):
  - Lysdioder eller LEDs (light emitting diodes) (25 stk.)
- 25 pinners konnektor for IO-kopling (IO = input/output) til eksternt utstyr
- Kommunikasjon:
  - USB (samme kabel som for kraftforsyning, se nedenfor)
  - Radio
- Krafttilførsel:
  - USB (samme kabel som for kommunikasjon, se ovenfor)
  - Batteri (3V), som trengs bare dersom Microbit-en ikke skal være tilkoplet PC.

21.2.2.4 Tilleggsutstyr til Microbit

Mange produsenter har laget forskjellig slags utstyr i form av sensorer og aktuatorer (f.eks. motorer) og koplingsbrett som kan koples til Microbit. Et eksempel er Tinker Kit, se figur 21.3. Detaljerte kodeeksempler fins på https://tinkercademy.com.



Figur 21.3: Tinker Kit – tilleggsutstyr til Microbit.

21.2.3 Prosedyre for programmering av Microbit

Programmeringen av Microbit foregår typisk iht. denne prosedyren:

1. Du skriver koden (MicroPython eller MakeCode, ev. JavaScript) i den aktuelle editoren, f.eks. en av de webbaserte editorene som er tilgjengelige på http://microbit.org.

2. Du lagrer koden som en hex-fil på din PC. Dette er «å laste ned» eller *download* på godt norsk. Hex-filen inneholder kode som mikroprosessoren på Microbit-en kan kjøre. Hex-filen består av maskinkode som er «kryptisk» for oss, men fullt forståelig for mikroprosessoren.

3. Du kopierer eller «trekker over» hex-filen til Microbit-stasjonen som vises i (den vanlige) filutforskeren på din PC. Dette kalles også å *flashe* hex-filen. Dette gjør du i den vanlige filutforskeren. Ordet flashe stammer fra at datalageret på Microbit er et såkalt flash-lager.

4. Når flashingen fra din PC til Microbit er ferdig, kjører Microbit programmet automatisk.

Merk: I MicroPython-editoren Mu kombineres punktene 2 og 3 ovenfor, dvs. download og flash, når du klikker Flash-knappen i Mu. Mer om Mu i kap. 21.2.4.3.

21.2.4 MicroPython

21.2.4.1 Innledning

MicroPython er et tekstbasert språk som er en forenklet utgave av Python. Du kan programmere Microbit med MicroPython.

Full beskrivelse av MicroPython fins på

http://micropython.org.

Bibliotekene (engelsk: libraries) i MicroPython er ikke de samme som i Python, jf. følgende beskrivelse av MicroPython-biblioteker på https://docs.micropython.org/en/latest/library:

- *MicroPython implements a subset of Python functionality for each module.*
- *To ease extensibility, MicroPython versions of standard Python modules usually have u (≪micro≫) prefix.*
- *Any particular MicroPython variant or port may miss any feature/function described in this general documentation (due to resource constraints or other limitations).*

MicroPythons funksjonalitet er beskrevet på:

https://docs.micropython.org/

Bruk av MicroPython sammen med Microbit er beskrevet på:

https://microbit-micropython.readthedocs.io/

21.2.4.2 MicroPython-programmering i Microbits editor

På Microbits hjemmeside fins en editor for MicroPython. Den direkte adressen er:

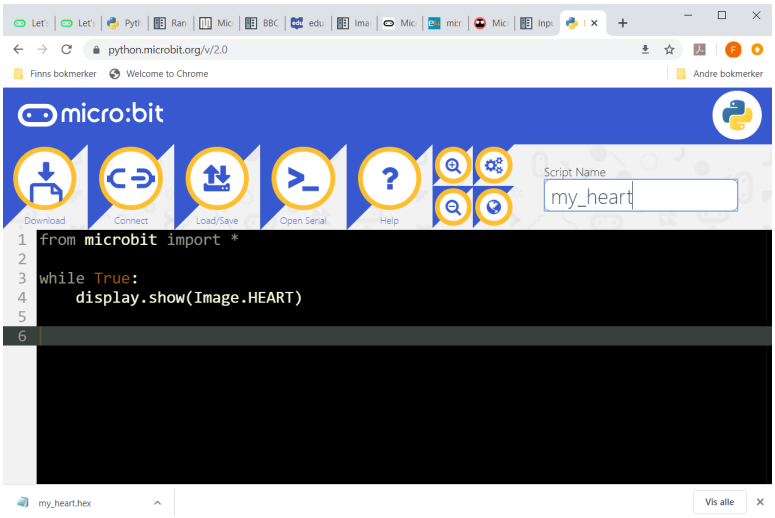
https://python.microbit.org

Eksempel 21.1 viser bruken av denne editoren.

Eksempel 21.1 *MicroPython-programmering i Microbits editor*

Figur 21.4 viser editoren med kode for presentasjon av et hjertesymbol på Microbits LED-display.

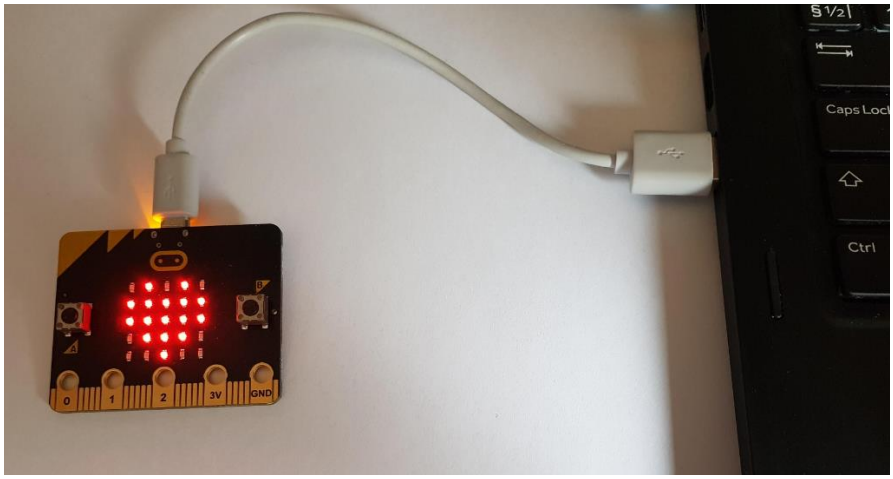
Kommentarer til noen av knappene/funksjonene i editorens verktøylinje:



Figur 21.4: Microbits webbaserte editor for MicroPython for visning av hjertesymbol på Microbits LED-display.

- **Download**: Ei hex-fil lagres i Nedlast-mappa på din PC.
- **Connect**: Her aktiverer du koplingen mellom PC og Microbit. Dersom du får feilmelding (kanskje fordi du allerede har aktivert en kopling i en annen editor), kan du kople ut og inn igjen ledningen mellom PC og Microbit.
- **Load/Save** gir mulighet til å åpne/lagre programfiler – både py-fil og hex-fil.

Når programmet er utviklet og flashet iht. prosedyren beskrevet i kap. 21.2.3, vises et vakkert hjertesymbol på Microbits LED-display, se figur 21.5. Programmet kjører helt til nytt program flashes.

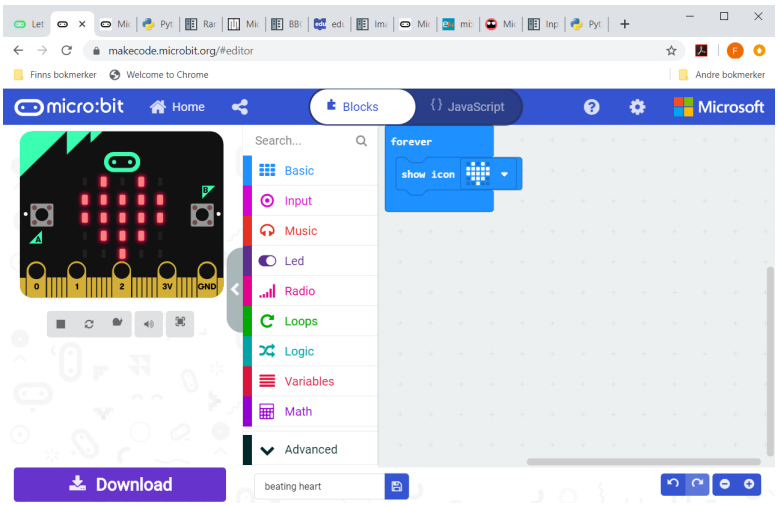


Figur 21.5: Hjertesymbol på Microbits LED-display.

Det er jo litt artig å se hvordan MakeCode-koden ser ut for det samme programmet. MakeCode-editoren er tilgjengelig på:

https://makecode.microbit.org/#editor

Figur 21.6 viser koden i MakeCode-editoren.



Figur 21.6: Kode i den webbaserte MakeCode-editoren for visning av hjertesymbol på Microbits LED-display.

I MakeCode-editoren brukes Download-knappen til å laste ned (lagre) hex-filen til Nedlast-mappa på PC-en. Du må selv flashe denne hex-filen til Microbit, som forklart i kap. 21.2.3.

[Slutt på eksempel 21.1]

21.2.4.3 MicroPython-programmering i editoren Mu

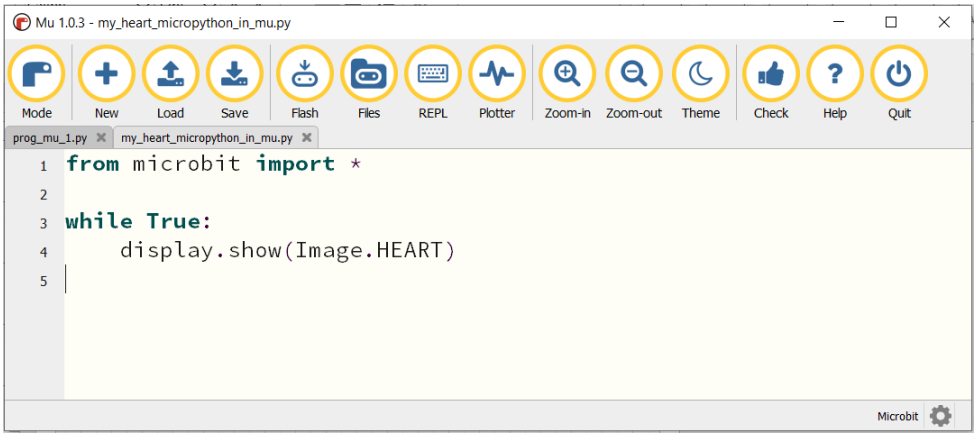
Mu er en kraftigere editor for MicroPython-programmering enn den webbaserte editoren på http://microbit.org, så vi anbefaler Mu. Mu er tilgjengelig fra nedlasting og installering på https://codewith.mu.

Eksempel 21.2 *MicroPython-programmering i Mu*

Figur 21.7 viser Mu med MicroPython-koden for hjerteeksempelet presentert i eksempel 21.1.

Kommentarer til noen av knappene/funksjonene i verktøylinjen i Mu:

- **Mode**: Her velger du hvilken ekstern enhet Mu skal koples til (slik at du kan flashe koden direkte til enheten). Enheten vil være Microbit i vårt tilfelle.
- **Load** og **Save**: Her kan du åpne og lagre py-filer.



Figur 21.7: Mu med MicroPython-koden for hjerteeksempelet presentert i eksempel 21.1.

- **Flash**: Her kan du flashe programmet direkte til Microbit (i vårt tilfelle). Flash-knappen kombinerer punktene 2 og 3 i prosedyren beskrevet i kap. 21.2.3. Etter at et program er flashet (overført i kjørbar form) til Microbiten, starter programmet å kjøre automatisk på Microbiten.
- **REPL** (read–eval–print loop): Åpner en kommandolinje a la konsollen i Spyder.
- **Plotter**: Åpner et plottevindu inne i Mu, men det er ikke støtte for Matplotlib i MicroPython eller Mu. Plotter-vinduet er et *enkelt* plottevindu, uten tallverdier langs x-aksen (tidsaksen), men med automatisk skalering langs aksene. Plotter-vinduet plotter fortløpende de *tuppelverdiene* som skrives til (vises i) REPL vha. print()-funksjonen, altså print(tuppel).

[Slutt på eksempel 21.2]

Eksempel 21.3 viser hvordan vi kan logge måledata kontinuerlig med Microbit og til slutt få dataene lagret i en csv-fil, som er en tekstfil der dataene er skilt i kolonner med komma (comma-separated values). Csv-filer er hendige fordi de kan leses av «alle» programmer, bl.a. Python, Excel, Matlab, LabVIEW, Word, Notisblokk m.m.

Eksempel 21.3 *Datalogging med Microbit*

Informasjon om datalogging med Microbit:

- Dataloggeprogrammet skal kjøre syklisk med en for- eller while-løkke der syklustiden (tiden mellom hver iterasjon av løkken) styres med en vent-funksjon, f.eks. sleep(ventetid_millisek), der ventetid er i antall millisekunder. Alternativt kan du bruke time.sleep(ventetid_sekunder), hvis du har importert time-modulen.
- Du må bruke print()-funksjonen til å overføre data fra Microbit til PC-en fortløpende. Du legger print()-funksjonskallet inne i løkken i ditt MicroPython-program. Argumentet til print() *må* være en tuppel, som skal inneholde dataene.

  - Hvis kun *én* verdi skal vises, må argumentet være tuppelen (verdi,), altså komma uten noe etter.
  - Hvis print skal overføre *to* verdier, må argumentet være tuppelen (verdi1, verdi2).

- Dataene vises fortløpende i REPL-vinduet (konsollen) og/eller i Plotter-vinduet i Mu-editoren på PC-en dersom disse er åpnet i Mu, men merk:

- Overføring av data fra Microbit til Mu starter *ikke automatisk* etter flashingen/programstarten. Overføringen kommer i gang først etter at du har trykket Reset-knappen på Microbit.

- Etter at dataoverføringen er igangsatt (med Reset-knappen på Microbit), kan du sette i gang lagring av dataene som overføres og vises i REPL i en csv-fil med tuppelverdiene som rader, med én rad (tuppelverdi) for hver gang print() kjøres. Obs: Du setter i gang fillagringen ved å åpne Plotter-vinduet *etter* at dataoverføringen er igangsatt (med Reset-knappen).

- Når filoverføringen er avsluttet ved at programmet har stoppet eller at du har lukket Plotter-vinduet, er filen med dataene tilgjengelig i mappen Bruker / mu_code / data_capture. Du kan så plotte dataene i csv-filen på skikkelig vis f.eks. i Python/Spyder. Filens innhold kan leses med f.eks. loadtxt()-funksjonen, jf. kap. 9.3.

- Datalogging er også beskrevet på https://microbit.org/projects/make-it-code-it/python-wireless-data-logger.

En oppgave gitt til studenter (løsningen følger):

- Finn ut hvor selve temperatursensoren befinner seg på Microbit.

- Lag et program i MicroPython som logger temperaturen detektert av Microbit hvert sekund i ett minutt (kan realiseres med en for- eller while-løkke). Sørg for at temperaturen varierer under eksperimentet. (Bør ikke måle temperaturen direkte i vann e.l.)

- Når eksperimentet er ferdig, skal du plotte temperaturen i Python i Spyder (eller et annet Python-verktøy med mulighet for plotting) som funksjon av tiden (med $t = 0$ sek. som starttidspunkt). Du skal altså lage to programmer i denne oppgaven – ett MicroPython-program som skal kjøre på Microbit og ett Python-program som skal kjøre på PC-en. (Det er ingen skikkelige plottemuligheter med MicroPython, så den oppgaven må vi utføre i f.eks. Spyder.

Løsning på oppgaven:

Temperatursensoren befinner seg inne i mikroprosessoren, og temperaturen vil kunne bli høyere enn lufttemperaturen (siden mikroprosessoren utviklet varme).

Program 21.1 er Mu-programmet som flashes til Microbiten.

http://techteach.no/python/files/prog_mu_temperature.py

Listing 21.1: prog_mu_temperature.py

```
from microbit import *

t_step = 1  # sec
t_start = 0
t_stop = 60
N = int((t_stop - t_start)/t_step) + 1

for k in range(0, N):
    sleep(int(t_step*1000))
    t_k = k*t_step
    data_tuple = (t_k, temperature())
    print(data_tuple)
```

Figur 21.8 viser utsnitt av csv-filen.

	A	B
1	5,28	
2		
3	6,28	
4		
5	7,28	
6		
7	8,28	
8		
9	9,28	
10		
11	9,28	
12		
13	10,28	
14		

Figur 21.8: Utsnitt av csv-filen.

Program 21.2 leser csv-filen og plotter temperaturen vs. tid i Spyder. (Under eksperimentet ble mikroprosessoren berørt med fingeren, hvilket fikk temperaturen til å stige noe.)

http://techteach.no/python/files/prog_read_from_csv_with_loadtxt.py

Listing 21.2: prog_read_from_csv_with_loadtxt.py

```
import numpy as np
import matplotlib.pyplot as plt

data = np.loadtxt('20200520-175917.csv', delimiter=',')
```

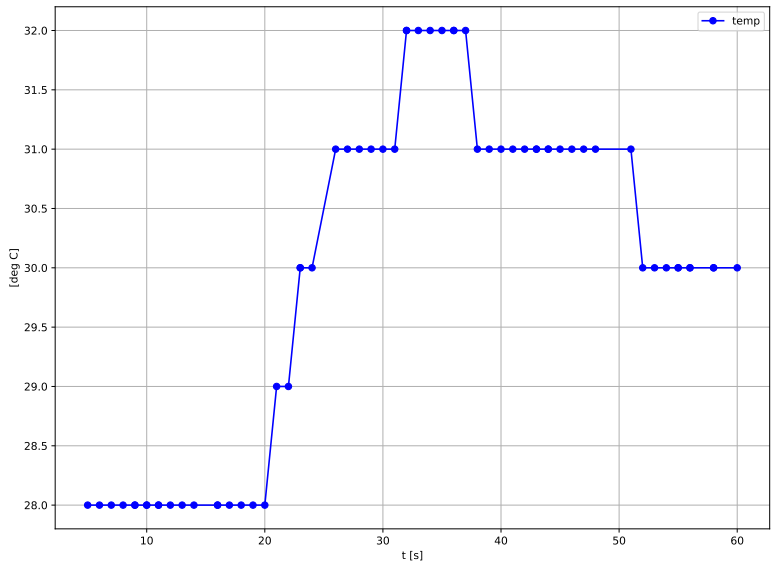
```
time = data[:, 0]
temp = t = data[:, 1]

plt.close('all')
plt.figure(1, figsize=(12, 9))

plt.plot(time, temp, 'b-o')
plt.grid()
plt.xlabel('t [s]')
plt.ylabel('[deg C]')
plt.legend(labels=('temp',))

plt.show()
plt.savefig('plot_temp_microbit.pdf')
```

Figur 21.9 viser plottet av temperaturen i Spyder. Overføringen av data fra Microbit til PC er ikke helt konsistent: Det ser ut til at det kan være enkelte iterasjoner av løkken der dataoverføringen ikke finner sted, se figur 21.9, men stort sett fungerer dataoverføringen bra.



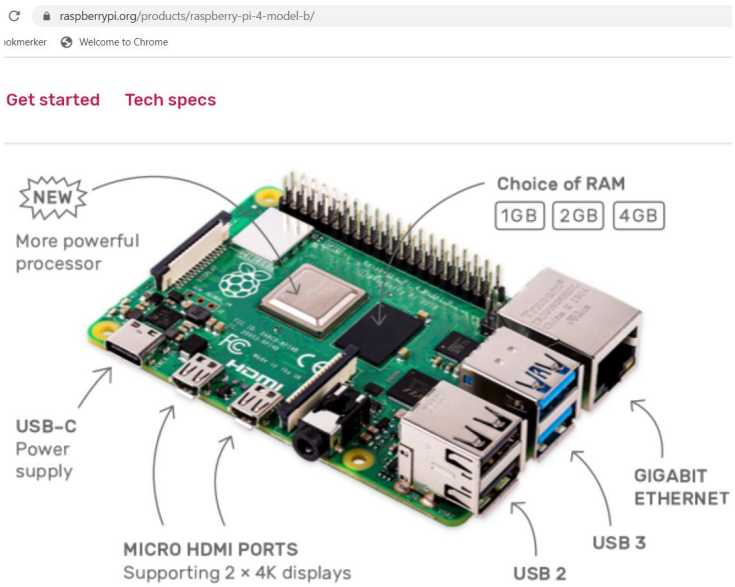
Figur 21.9: Plottet av temperaturen.

[Slutt på eksempel 21.3]

21.3 Raspberry Pi

21.3.1 Hva er Raspberry Pi?

Raspberry Pi er en liten, men fullblods, ett-korts datamaskin. Figur 21.10 viser et bilde av den nyligste versjonen, 4, på Raspberry Pis hjemmeside på http://raspberrypi.org. Som figur 21.10 viser, fins en rekke tilkoplingsmuligheter. Raspberry Pi har innebygd WiFi, så du kommer på internett, og på versjon 4 er det dessuten mulighet for Ethernet-tilkopling.



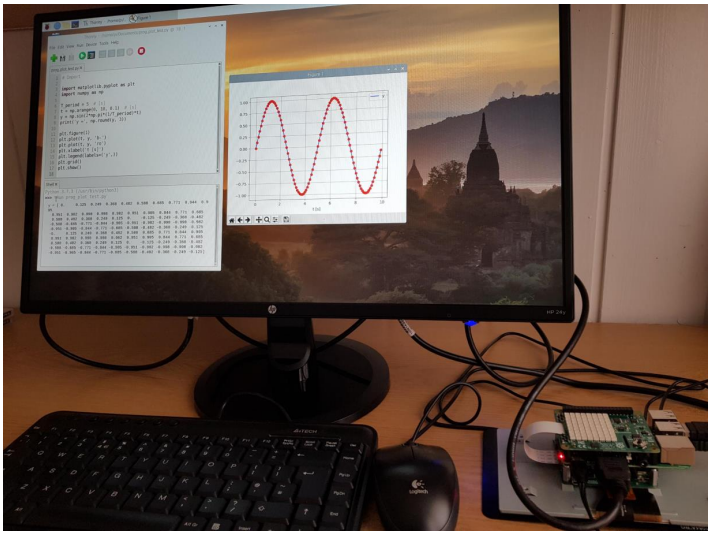
Figur 21.10: Ett-korts datamaskinen Raspberry Pi 4. (Fra http://raspberrypi.org.)

Raspberry Pi kjører et operativsystem kalt Raspbian, som er basert på Linux-distribusjonen Debian. Raspbian har et grafisk brukergrensesnitt som minner om Microsoft Windows.

21.3.2 Hvilket tilleggsutstyr trenger du?

21.3.2.1 Standard skjerm, tastatur og mus

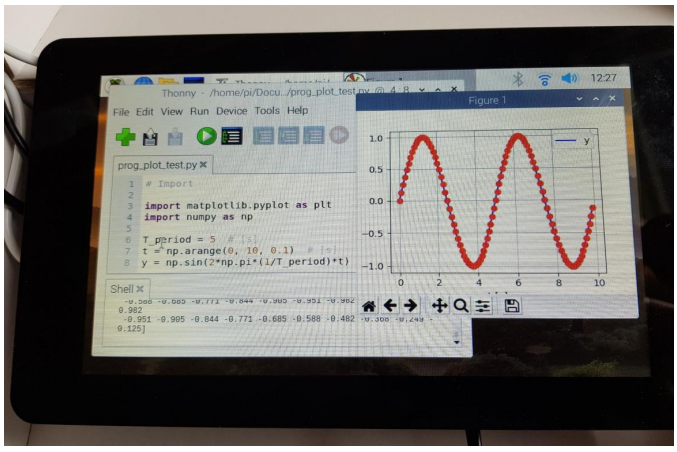
Raspberry Pi er kortet vist i figur 21.10, og ikke noe annet følger med (bortsett fra en plastboks der du kan plassere kortet). Du må selv skaffe utstyr for å kunne kommunisere med Raspberry Pi. Figur 21.11 viser utstyr som forfatter FH har skaffet seg for å bruke Rasberry Pi-en, nemlig skjerm, tastatur og mus. Raspberry Pi-en er vist nederst til høyre i bildet. (Den befinner seg der i midten av en stabel med i alt tre kort. De to andre er et skjermkort og sensorkortet SenseHAT. Disse omtales senere i kapitlet.) Det er også mulig å sette opp Raspberry Pi til å bruke skjerm, tastatur og mus på en bærbar PC.



Figur 21.11: Utstyr anskaffet for bruk av Raspberry Pi: Skjerm. Tastatur. Mus.

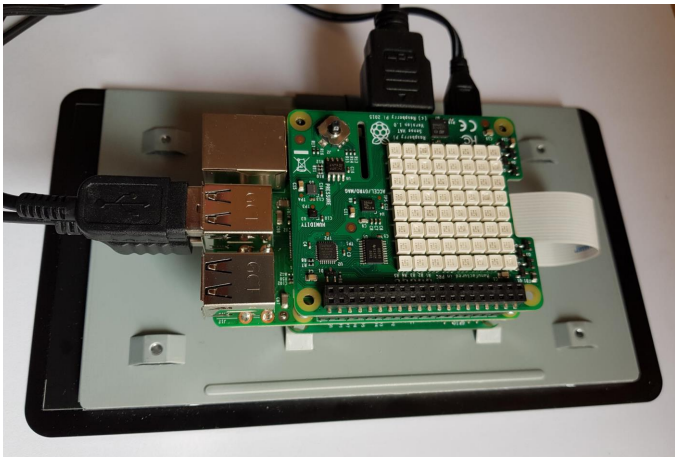
21.3.2.2 Mini-berøringsskjerm

Hvis du skal lage et system med minimal utrusting med skjerm, tastatur og mus, kan du vurdere å anskaffe en mini-berøringsskjerm, se figur 21.12, som viser 7-tommers mini-skjermen Touch Screen Display (nærmere informasjon fins bl.a. på http://raspberrypi.org).



Figur 21.12: Mini-skjerm for bruk sammen med Raspberry Pi.

Figur 21.13 viser baksiden av skjermen. Raspberry Pi-en (i midten) er plassert oppå mini-skjermens skjermkort, og oppå Raspberry Pi-en er tilleggsenheten SenseHAT montert (SenseHAT omtales nedenfor).



Figur 21.13: Raspberry Pi (i midten) sammen med kortet for mini-skjermen (nederst) og SenseHAT (øverst).

21.3.2.3 Tilleggsutstyr

Det fins mye tilleggsutstyr til Rasberry Pi fra ulike produsenter, f.eks. ulike sensorer og aktuatorer slik at du kan bruke Raspberry Pi til måling og styring. Et eksempel er SenseHAT[1], som er et kort med diverse sensorer montert på kortet. SenseHAT er det øverste kortet vist i figur 21.13. SenseHAT har følgende sensorer:

- Gyroskop
- Akselerometer
- Magnetometer
- Temperatursensor
- Trykksensor
- Fuktighetssensor

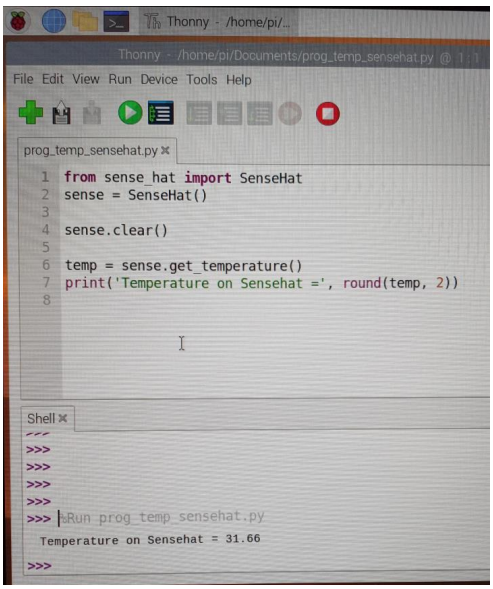
Figur 21.14 viser et eksempel på et Python-program for lesing av temperaturen på SenseHAT.

21.3.3 Programmering av Raspberry Pi

Raspberry Pi kommer med en Python-kompilator og bl.a. programmeringsmiljøet Thonny. Det er støtte for full Python (mens Microbit kun kan kjøre en enklere Python-versjon: MicroPython, jf. kap. 21.2.4). Du kan installere Python-pakker, bl.a. Matplotlib, men det er egne kommandoer for nedlasting og installering (andre enn ≪pip install≫). Du finner greit fram til informasjon om dette på nettet.

[1]HAT = Hardware Attached on Top

Figur 21.14 viser et eksempel på et Python-program i Thonny for engangs lesing av temperaturen på SenseHAT. (Resultat på et solvendt arbeidsrom en dag i mai: 31,66 grader C.)



Figur 21.14: Et Python-program for registrering av temperaturen på SenseHAT.

Et annet eksempel på et Python-program som kjører på Raspberry Pi er vist i figur 21.11 (PC-skjerm) og figur 21.12 (mini-skjerm). Det er det samme som kjører. I programmet er både matplotlib-pakken og numpy-pakken importert. Et sinussignal genereres med np.sin()-funksjonen og plottes med plt.plot()-funksjonen.

21.4 Oppgaver til kapittel 21 – ingen

Her er det tomt. Vi har ikke laget oppgaver til dette kapitlet. Grunnen er at det fins mange programeksempler og programmeringsoppgaver med komplette løsninger på hjemmesidene til Microbit (http://microbit.org) og Raspberry Pi (https://www.raspberrypi.org/) og på hjemmesidene til produsenter av tilleggutstyr.