Del 4 – Spara data på SD-kort

Hur man sparar och läser data från minneskort

Data genereras av mikrokontrollern och skickas till datorn via en USB-sladd. Men hur gör vi om man inte kan vara uppkopplad till datorn direkt via en USB-sladd? Om en mikrokontroller används för att samla in data och den inte är kopplad till en dator behöver den ett sätt att antingen skicka eller lagra datan tills den kan hämtas. I denna labb ska vi lära oss hur man lagrar data på ett SD-kort med en Feather. SD-kort är enkla minneskort som finns i många olika apparater t.ex. telefoner och kameror.

Material

- Adafruit Feather RP2040
- Kopplingsdäck
- USB-sladd för att koppla Feather till datorn
- Dator
- BMP180 breakout
- MicroSD-kort-breakout modul samt MicroSD-kort
- Sladdar för koppling

Protokoll och koppling

Kommunikationen mellan feathen och SD-kortet sker med ytterligare ett protokoll, kallat SPI (Serial Peripheral Interface). SPI består av en huvudenhet (master) och en underenhet (slave). Flera underenheter kan vara anslutet på samma SPI gränssnitt likt I2C protokollet i tidigare laboration, en viktigt skillnad är dock att varje underenhet behöver en egen CHIP-SELECT signal koppling som berättar för underenheter att det är just den enhetet som huvudenheten försöker tala med.
Kopplingarna som behöver göras är:

- **SCK** eller ibland kallad CLK
- **MOSI** vilket står för Master Out & Slave In, kan ibland heta bara MO eller bara SI
- **MISO** vilket står för Master In & Slave Out, kan ibland heta bara MI eller bara SO
- **CS** dvs chip select. Ibland benämnd Slave Select, SS. Denna kopplas från CS på underenheten till valfri digitalt styrbar pin på Feather.

E-post: cansat@au.se
Telefon: 070-000 90 56
Senast uppdaterad: 2024-11-19

Astronomisk Ungdom

Strömförsörjning behövs även här och kopplas med jord från GND och positiv spänning 3,3V.

Programmering

Det första vi behöver göra är att importera de moduler vi kommer använda, dessa finns inbyggda och behöver ej installeras.

```
1 import board
2 import busio
3 import sdcardio
4 import storage
```

Nästa steg är att starta igång SPI samt specificera vilken pin vi valt för chip select. Exemplet nedan utgår från att pin 10 är vald, men alla digitala pinnar fungerar.

```
6 spi = board.SPI()
7 cs = board.D10
```

Sedan är det dags att starta igång filsystemet på SD-kortet, och göra det åtkomligt för oss. Det gör vi genom följande:

```
9 sdcard = sdcardio.SDCard(spi, cs)
10 vfs = storage.VfsFat(sdcard)
11 storage.mount(vfs, "/sd")
```

Vi kan nu använda samma funktioner som i vanliga Python för att läsa och skriva till filer. Exempel på detta kommer nedan.

Skriva:

```
13 with open("/sd/test.txt", "w") as f:
14     f.write("Hello world!\r\n")
```

Läsa:

```
13 with open("/sd/test.txt", "r") as f:
14     print("Read line from file:")
15     print(f.readline())
```

Vill vi skriva ny data till en fil utan att radera den gamla kan vi använda följande:

```
13 with open("/sd/test.txt", "a") as f:
14     f.write("This is another line!\r\n")
```

Uppgift

Använd koden från den förra laboration, men istället för att bara skriva mätvärdena till serie terminalen, så spara även värdena i en fil på ett SD-kort för senare analys.
För att göra det tydligare när olika mätningar tagits i förhållande till varandra kan man använda funktionen time.monotonic(). Denna ger tid i sekunder från en okänd starttid. Du kan alltså endast använda den för att ta reda på tiden mellan två olika mätningar, ej något klockslag. Men känner du till tiden du startade mätningarna bör du kunna räkna ut resten.
Exempel på dess användning:
print(time.monotonic())