Course notes, module 4 week 8
I/O interfacing to sensors and actuators

Kjeld Jensen kjen@sdu.dk

Agenda

1. Recap of the exercises from previous module
2. Preparation for next module
3. Exercises
4. Optional exercises

1. Recap of exercises from previous module

In the previous module we configured RPi as both wifi client and as wifi access point. We also set up routing between the wifi access point clients and the internet. We created a script that is run each time the RPi boots to automatize the startup process, and we wrote some simple PhP scripts to share data.

We will briefly review the results and findings of the exercises and the optional exercises in class.

2. Preparation for next module

In module 5 Henrik Midtiby will teach you about git. Please read this document before the module: https://missing.csail.mit.edu/2020/version-control/

3. Exercises

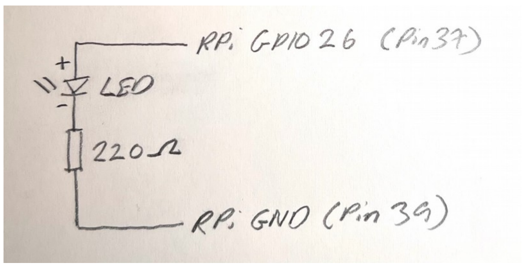
In these exercises you will be using the breadboard with the Raspberry Pico installed.

Before conducting the exercises please get the updated LEO1 kit documentation from itslearning and read slide 7 concerning the use of the breadboard. You should also read slide 3 (first paragraph) and slide 2.

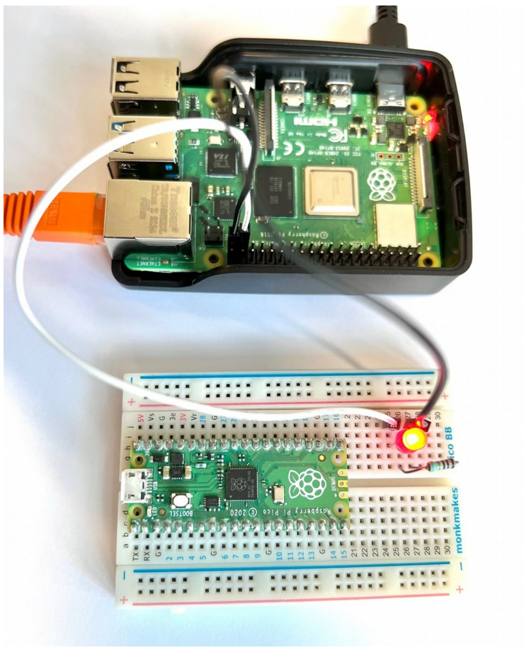
3.1 Blinking LED indicating that the RPi is running

On the breadboard make a simple circuit of a LED in series with a 220 Ohm resistor.

Please notice that the LED has a an anode (+) and a cathode (-), see LEO1 kit slide 9 for more information.



Connect the positive end of the circuit to `GPIO 26` (pin 37) and the negative end of the circuit to the `GND` (pin 39). You can find an overview of the GPIO pins on slide 5.



Make a Shell script `led_blink.sh` that makes the LED blink at 1 Hz using the GPIO commands below

Initialize the GPIO Pin
```
$ echo "26" > /sys/class/gpio/export
$ echo "out" > /sys/class/gpio/gpio26/direction
```

Set the GPIO pin state
```
$ echo "1" > /sys/class/gpio/gpio26/value
$ echo "0" > /sys/class/gpio/gpio26/value
```

Reset the GPIO pin
```
$ echo "26" > /sys/class/gpio/unexport
```

Run the script and use the `htop` command to ensure that the script only consumes insignificant CPU resources. If you haven't installed htop already, run this command:

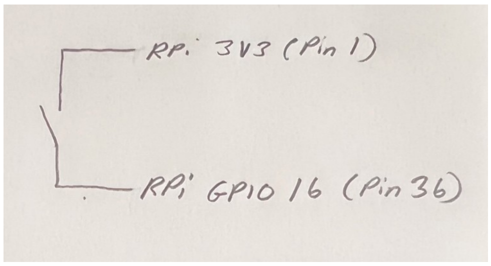
```
$ sudo apt install htop
```

Add `led_blink.sh` to the RPi startup script `boot.sh` created in the module 3 exercise to run it when the RPi boots. You now have a LED indicating that you RPi is up and running.

Remember to put a `&` after the `led_blink.sh` to start the script as a background process.

3.2 RPi shutdown switch

Using one female-female and one female-male testlead create a "switch" like in the diagram below.



Make a shell script `read_switch.sh` that reads and outputs the status of the switch as `ON` or `OFF` using the GPIO commands below:

Initialize the GPIO Pin

```
$ echo "16" > /sys/class/gpio/export
$ echo "in" > /sys/class/gpio/gpio16/direction
```

Set the GPIO pin state

```
$ cat /sys/class/gpio/gpio16/value
```

Reset the GPIO pin

```
$ echo "16" > /sys/class/gpio/unexport
```

Make another shell script `power_off.sh` that uses `read_switch.sh` to read the switch status once per second. If the switch returns `ON` for 3 seconds, the script should power off the RPi.

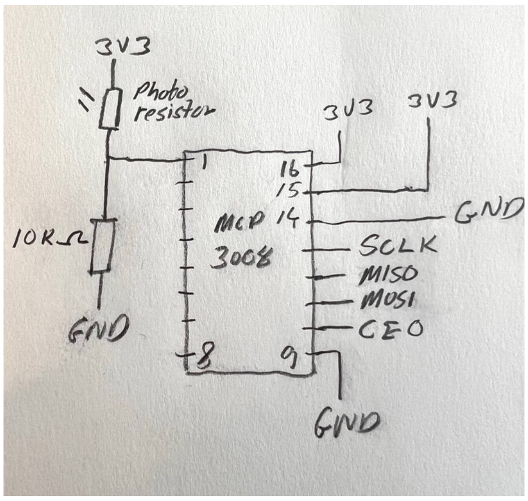
Add `power_off.sh` to the RPi startup script `boot.sh` created in the module 3 exercise to run it when the RPi boots. You now have a switch that properly powers off the RPi without the risk of creating a file system error.

3.3 RPi analog light sensor

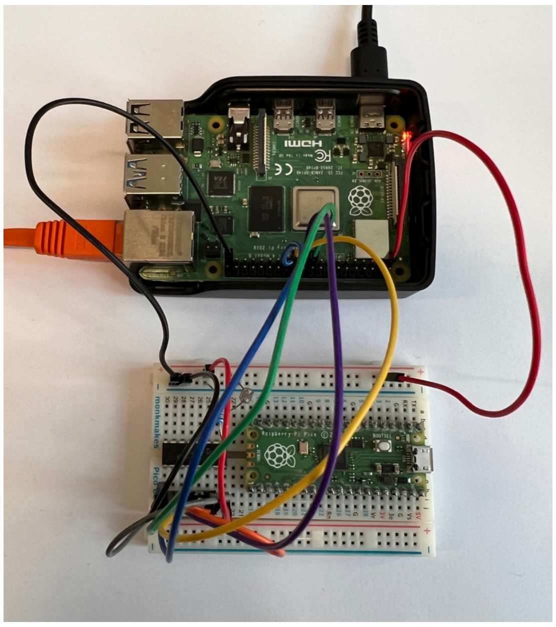
Using the RPi for digital I/O is valuable. The Rpi unfortunately does not directly support analog input, but this can be mitigated by adding an external ADC such as the MCP3008 in your LEO1 kit. Slide 9 shows the pinout of the MCP3008.

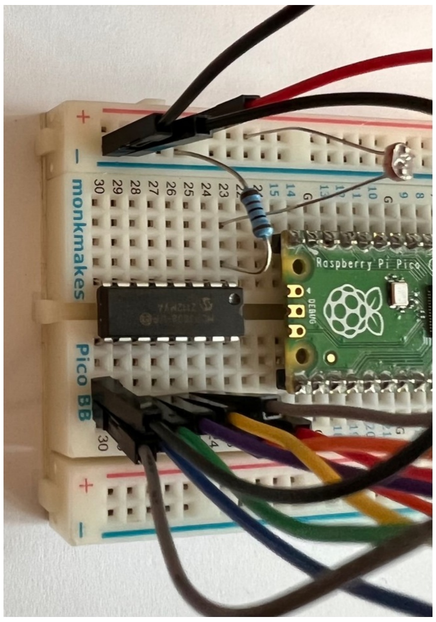
The MCP3008 communicates via a SPI bus. RPi supports both hardware and software SPI. We will be using the hardware SPI.

Set up the circuit below to connect the light sensor to the RPi via the MCP3008.



Make sure that the RPi is powered off before doing so and be very careful to triplecheck all connections before turning on the power.





In `raspi-config` under `Interface options` enable `SPI`

Create a Python script (text file) named `read_mcp3008.py` with the Python code below:

```
#!/usr/bin/env python3

import spidev

spi = spidev.SpiDev()
spi.open (0,0)
spi.max_speed_hz =1000000

def read_adc(channel):
     adc = spi.xfer2 ([1 ,(8 + channel) << 4 , 0])
     data = ((adc[1] & 3) << 8) + adc[2]
     return data

print (read_adc(0))
```

Run the Python script using this command:

```
$ python3 read_mcp3008.py
```

You should now be able to read the value from the ADC channel 0. Try covering top of the photoresistor and run the script againg to see a lower value.

If you have problems with a missing SPI library, then run:

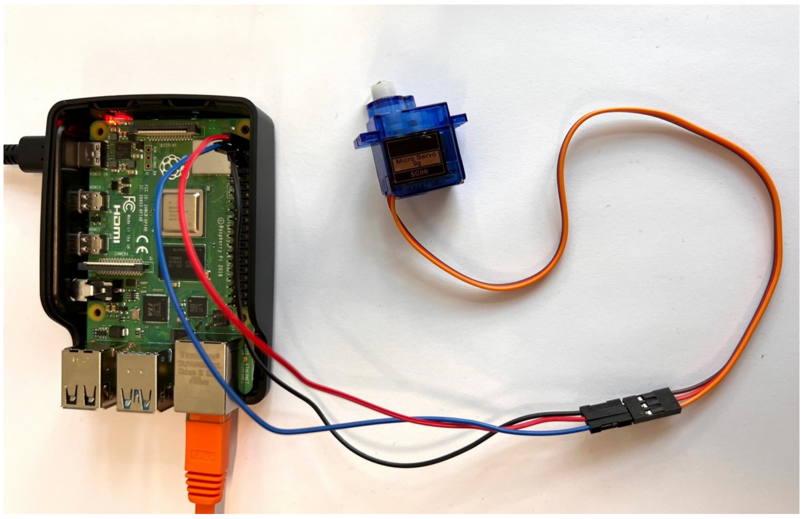
```
$ sudo apt update
$ sudo apt upgrade
$ sudo apt install python-pip
$ pip install spidev
```

3.4 RPi servo control

The RPi can also control motors and servos using a PWM signal. In this exercise we will control the servo in the LEO1 kit.

Connect the SERVO to the RPi using three male-female test leads

Servo	**RPi**
Brown	GND
Red	5V
Orange	GPIO 2 (pin 3)



Create a Python script named `servo_control.py` with the Python code below:

```
#!/usr/bin/env python3

PIN_SERVO = 3

import RPi.GPIO as GPIO
from time import sleep
GPIO.setmode(GPIO.BOARD)
GPIO.setup(PIN_SERVO, GPIO.OUT)
pwm=GPIO.PWM(PIN_SERVO, 50) # 50 Hz
pwm.start(0)

def servo_angle(servo_angle):
     duty = servo_angle / 18 + 2
     GPIO.output(PIN_SERVO, True)
     pwm.ChangeDutyCycle(duty)
     sleep(1)
     GPIO.output(PIN_SERVO, False)
     pwm.ChangeDutyCycle(0)

def servo_stop():
     pwm.stop()
     GPIO.cleanup()

servo_angle (0)
sleep (1)
servo_angle (90)
sleep (1)
servo_angle (180)
servo_stop()
```

Now try to run the `servo_control.py` and see the servo being actuated according to the script.

4. Optional exercises

4.1 RPi LED signal operational state

Extend the LED script created in exercise 3.1 to signal the current state of the RPi. Use the LED to produce "counts"

1 count means all ok
2 means no internet access from the RPi
3 means means high CPU temperature

4.2 Recap of previous optional exercises

You likely haven't worked your way through the optional exercises from last time and maybe some of those before. Please complete those.