Tutorial 5: Smart lights

1. Introduction

1.1 What am I learning here and why?

One of the most important pillars of the smart home is saving energy. This reason is mentioned time and again when it comes to why people opt for smart home technologies. After all, saving energy not only has an ecological advantage, but above all a cost advantage.

There are multiple ways for saving energy:

- Intelligent lights ensure that they are only switched on when a person is in the room
- Intelligent heating ensures that the heating is lowered at a certain time
- A smart washing machine can be installed in a way that it washes your clothes at the time it is the cheapest to do so

In this tutorial we will focus on smart lighting and show how you can implement it in your SmartHome4Seniors house model.

Learning objectives

In this tutorial you will get to know one opportunity how to control your lightning using a motion sensor combined with a potentiometer. The tutorial is based on the scenario that the presence of a person is registered and the light is controlled based on predefined settings.

After you have completed the tutorial you will

- be able to connect and control a LED light
- be able to connect and control the PIR motion sensor
- be able to set a potentiometer

1.2 What do I need?

So that you can carry out the installations shown in this tutorial you should have downloaded the Thonny programming environment on your device. Also, you need to have installed the firmware of MicroPython on your Raspberry Pi Pico. The extended modifications (see p 42 in manual) including the extra components and their connectivity must also be made on the breadboard.

As regards physical skills you should be able to count off holes on the breadbord and insert components to it.

For implementation on the SmartHome4Seniors house model you need the following material:

- Raspberry Pi Pico
- Full size breadboard
- Micro-USB cable
- several male-to-male jumper wires
- LEDs (any colours)
- PIR motion sensor
- Potentiometer
- 220 Ohm resistor

This project has been funded with support from the European Commission. This publication [communication] reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

2. Learning content

2.1 Theoretical background

To understand the content of this tutorial well, you will now get an introduction to the most important terms and contexts.

Definitions

- A **PIR motion sensor** have a so-called PIR sensor (passive infrared sensor) and uses thermal radiation of living beings for detection. As a result, they detect even the smallest movements. (source: Hager Vertriebsgesellschaft mbH & Co. KG: Bewegungs- und Präsenzmelder. Die Helfer im Dunkeln. Online available at: https://hager.com/de/wissen/e-volution/fachwissen-elektrotechnik/bewegungs--und-praesenzmelder (called at 19 Nov 2023))
- With the help of a **potentiometer** you can specify several settings. For example, how long the light should stay on when a person is detected or that the light should only come on at a certain ambient brightness (i.e. at night).

2.2 Step-by-step guide

Now let's move on to the implementation. To do this, take the SmartHome4Seniors house model or just look at the instructions and recordings.

At least three installation steps are required for installing and controlling an LED with a PIR motion sensor. These are:

1. Install and control of the LED lights
2. Install and control of the PIR motion sensor
3. Install and set the potentiometer

The installations shown include connecting the electronic materials in the house model and writing the program. The steps come with instructional videos showing how it should be executed. If you are not sure about the components used in this tutorial, please advise the SmartHome Kit Manual (link) and the relevant chapter in the SmartHome Guide (link).

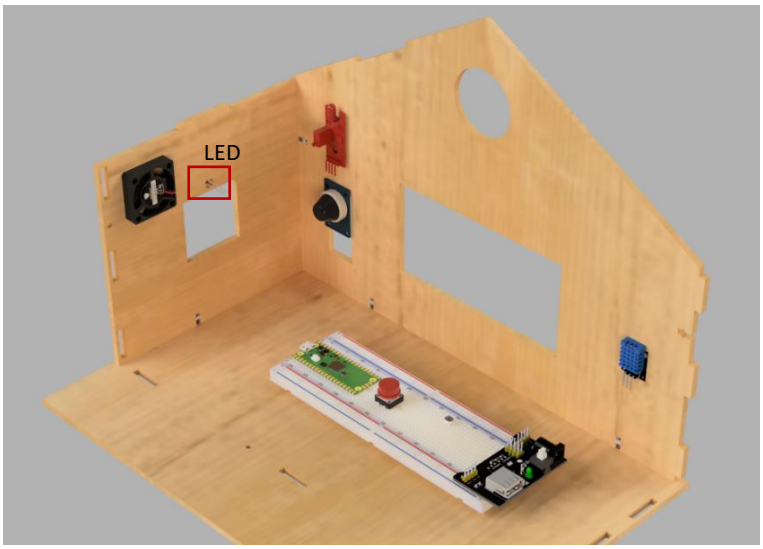
NOTE: Since the experiments involved are all circuit experiments, a wrong connection or short circuit may damage your RaspberryPi Pico development board. Please, always check the circuit again before connecting the power supply.

2.2.1 Install the sensors

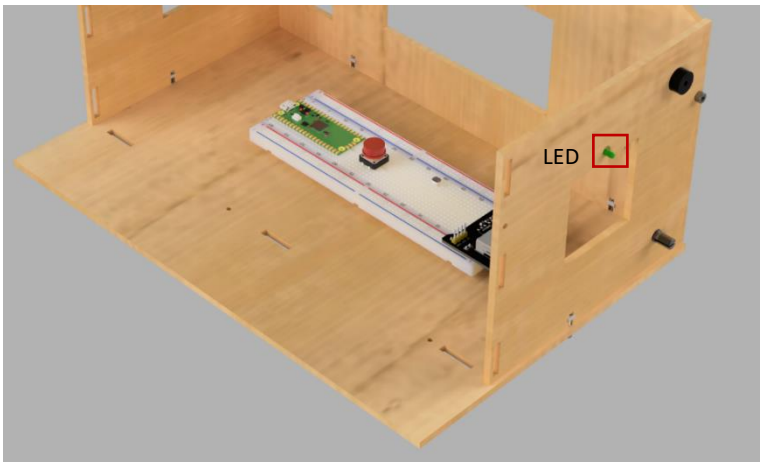
1. Mount the LEDs

The first LED light needs to be mounted on the left-side piece of the SmartHome4Seniors house model. Insert the **LED** to the mounting hole and friction will keep it in place.

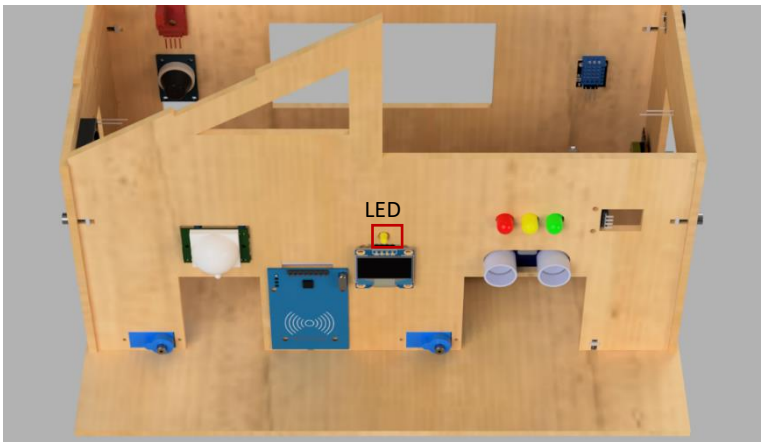
This project has been funded with support from the European Commission. This publication [communication] reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.



The second LED light (green) needs to be inserted in the right-side piece in the mounting hole and friction will keep it in place.



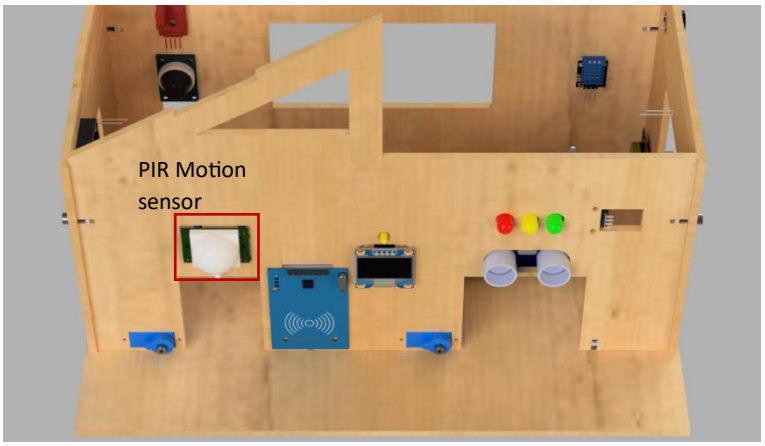
The third LED (yellow) needs to be mounted in the front-side of the house model. Insert the LED in the mounting hole and friction will keep it in place.



2. PIR Motion Sensor

To insert it in the SmartHome4Seniors house model you two bolt and two nuts. Mount the sensor through the top left and bottom right mounting holes.

This project has been funded with support from the European Commission. This publication [communication] reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.



2.2.2 Connect the electronics

Connect the cables as described and shown in the diagram.

LEDs

– connect the longer end (+) of the LED to 220Ohm resistors

– connect the resistors to GPIO10 (red), GPIO11 (blue), GPIO12 (yellow), GPIO13 (green)

– connect the shorter end (-) of the LEDs to the GND(-) rail

PIR Motion Sensor

– connect the VCC (red cable) to the 5V rail (+)

– connect the GND (black cable) to the GND rail (-)

– connect the OUT (yellow cable) to GPIO22 pin

Potentiometer

– black cable should be connected the GND rail (-)

– orange cable should be connected to GPIO27 ADC pin

– red cable should be connected to 3V3 power pin

– turn the potentiometer to the left so it's off

Photoresistor

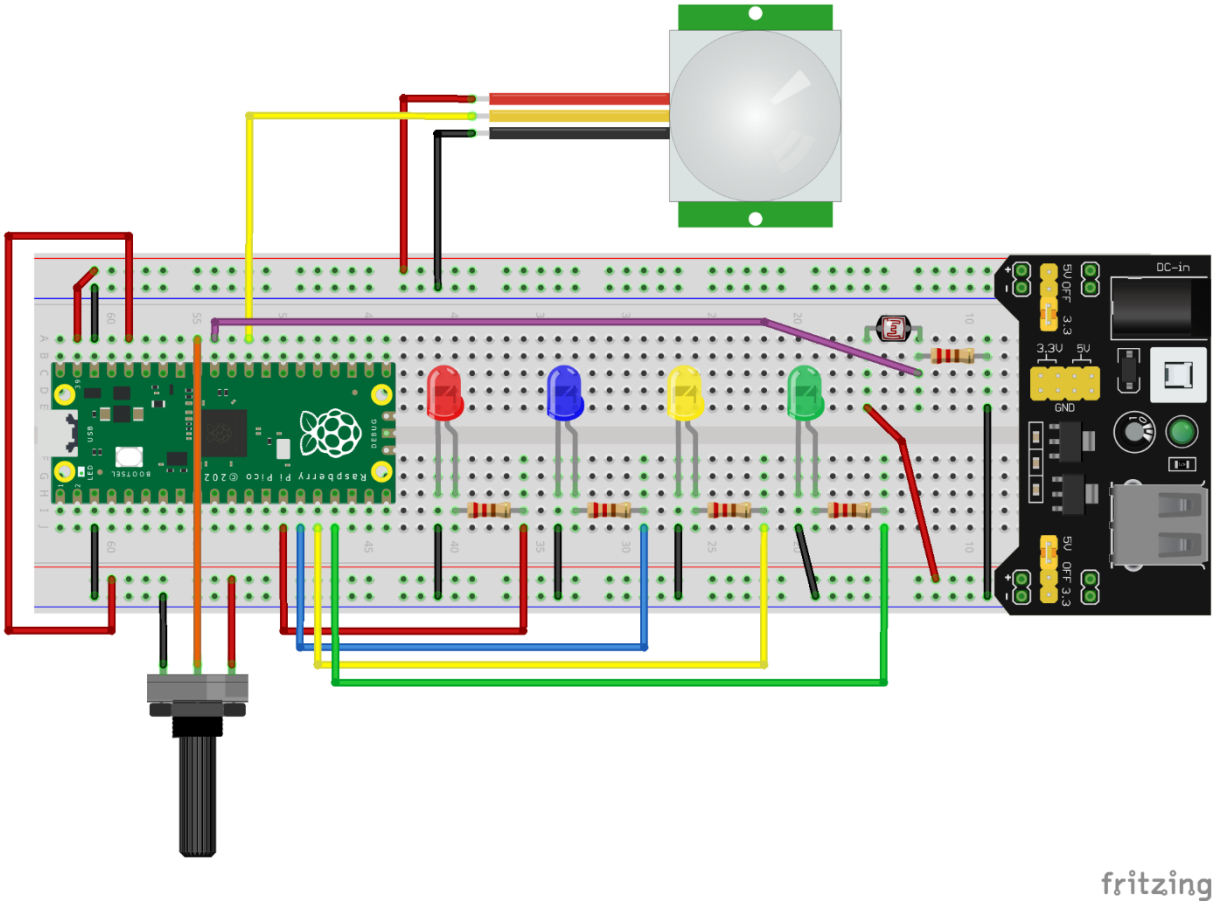
– connect the right leg of the photoresistor to a 220 Ohm resistor

– connect the other side of the resistor to the GND rail (-)

– connect the right leg of the photoresistor to the GPIO26 ADC pin

– connect the left leg of the photoresistor to 3V3 (+) rail

This project has been funded with support from the European Commission. This publication [communication] reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.



2.2.3 Code

Open Thonny from your device. Then go to File -> Save as…, choose Raspberry Pi Pico, and save your file under the name lighting.py.

Write the following code in the editor page of Thonny.

```
from machine import Pin, ADC, PWM
from time import sleep

#Define pins for each component
PIN_PIR = 22
PIN_LED_1 = 10
PIN_LED_2 = 11
PIN_LED_3 = 12
PIN_LED_4 = 13
PIN_LDR = 26
PIN_POT = 27

#Setup global variables for u16
MAX = 65535
MID = 32768
MIN = 0

#Setup PIR sensor
PIR = Pin(PIN_PIR, Pin.IN, Pin.PULL_UP)

#Setup LED lights
led_1 = PWM(Pin(PIN_LED_1))
```


This project has been funded with support from the European Commission. This publication [communication] reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

```
led_2 = PWM(Pin(PIN_LED_2))
led_3 = PWM(Pin(PIN_LED_3))
led_4 = PWM(Pin(PIN_LED_4))

led_1.freq(1000)
led_2.freq(1000)
led_3.freq(1000)
led_4.freq(1000)

led_1.duty_u16(MIN)
led_2.duty_u16(MIN)
led_3.duty_u16(MIN)
led_4.duty_u16(MIN)
sleep(5)

#Setup LDR photoresistor
LDR = ADC(Pin(PIN_LDR))
LDR_MAX = 4000
LDR_MIN = 50
LDR_THRESHOLD = int((LDR_MAX + LDR_MIN)/2)

#Setup POT potentiometer
POT = ADC(Pin(PIN_POT))

while True:
    POT_value = POT.read_u16()
    #print(POT_value)
    led_1.duty_u16(POT_value)
    led_2.duty_u16(POT_value)
    led_3.duty_u16(POT_value)
    led_4.duty_u16(POT_value)
    sleep(0.1)

    if PIR.value():
        led_1.duty_u16(MAX)
        led_2.duty_u16(MAX)
        led_3.duty_u16(MAX)
        led_4.duty_u16(MAX)
        sleep(3)

    LDR_value = LDR.read_u16()
    print (LDR_value)
    if LDR_value > LDR_THRESHOLD:
        led_1.duty_u16(MAX)
        led_2.duty_u16(MAX)
        led_3.duty_u16(MAX)
        led_4.duty_u16(MAX)
        sleep(0.1)
```

Save the program by clicking the Save icon on the top left-hand side, or by pressing Ctrl+S on your keyboard.

Thonny will ask you where you want your program to be saved. Choose the Raspberry Pi Pico. Save the file as led.py and click OK. You always need to add the .py extension so that Thonny recognises the file as a Python file.

This project has been funded with support from the European Commission. This publication [communication] reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

2.2.4 Application

Now that you have connected everything you should see the LED turning on when detected movement and staying on for the predefined amount of time. And after a while it goes off again.

3. Summary

In this tutorial you have learned one option how to control your lightning using the method of motion detection. The setup of this scenario included the installation of three sensors, namely:

- LED lights
- PIR motion sensor
- Potentiometer to define the duration of lighting

4. References

Hager Vertriebsgesellschaft mbH & Co. KG: Bewegungs- und Präsenzmelder. Die Helfer im Dunkeln. Online available at: https://hager.com/de/wissen/e-volution/fachwissen-elektrotechnik/bewegungs--und-praesenzmelder (called at 19 Nov 2023)

This project has been funded with support from the European Commission. This publication [communication] reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.