

06.06.2018
Ayşe İnci Çiftçi
Gamze Keçibaş
Olgu Şirin

MECH 100 INTRODUCTION TO MECHANICAL ENGINEERING
Spring 2018
Homework 14
Raspberry Pi Assignment

In this assignment, we decide to measurement light density with LDR and Raspberry Pi 3. We use one LDR, 1 microfarad capacitor and jumper cable.



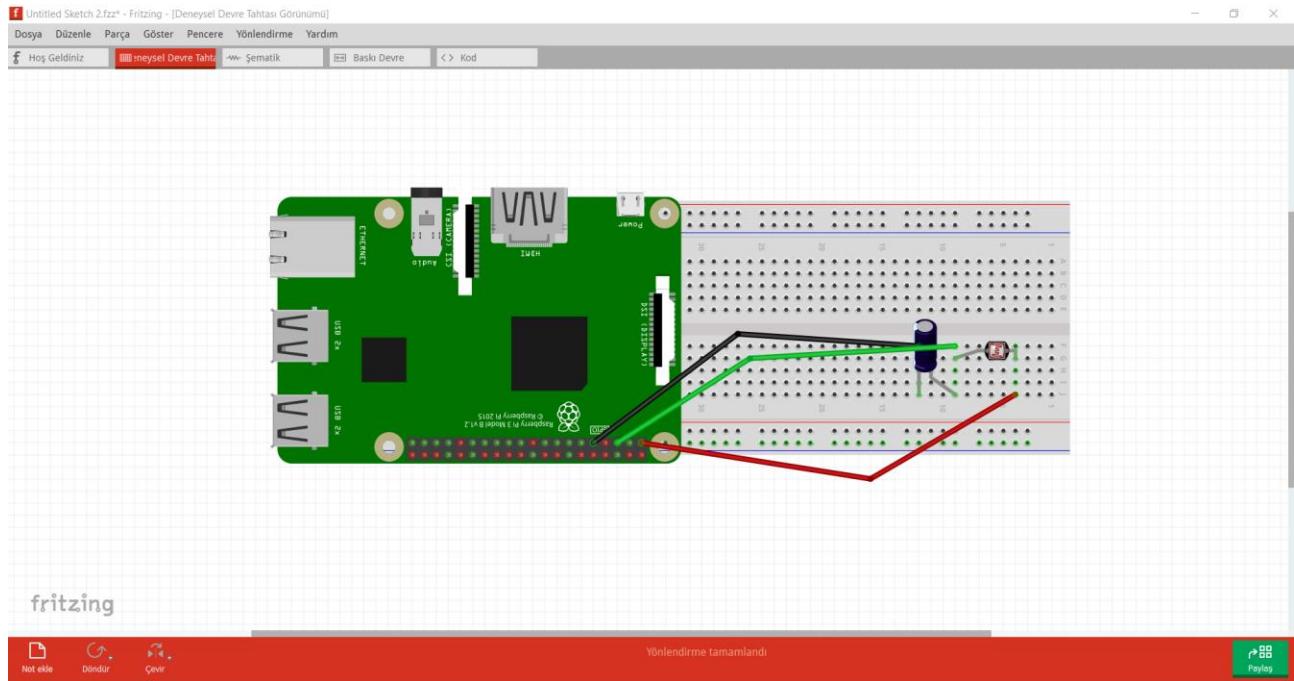
(LDR)1



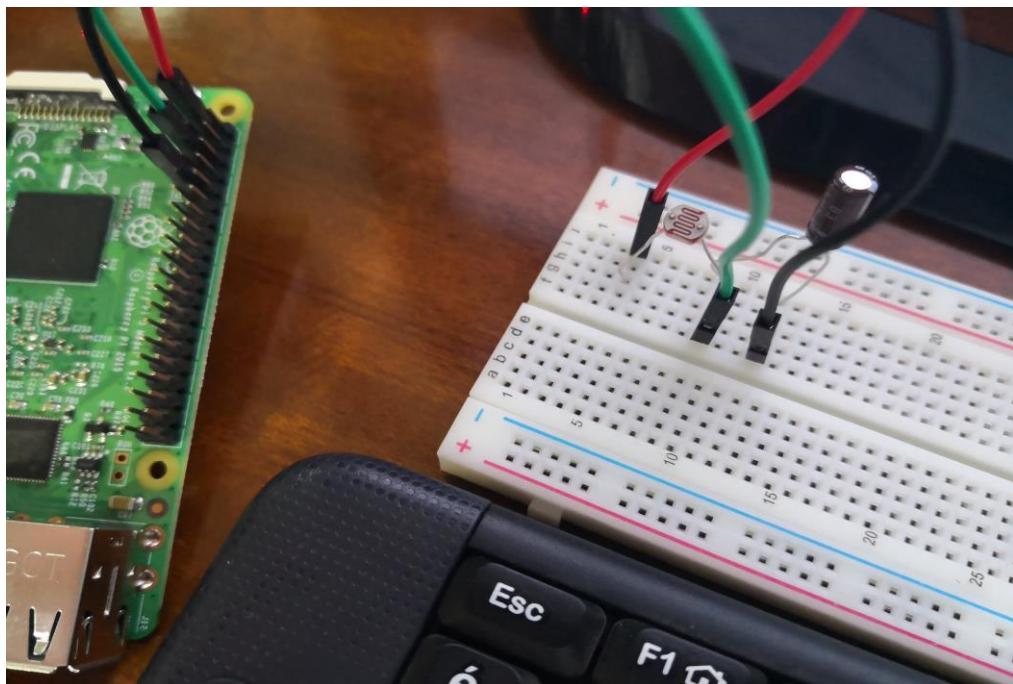
(1 microfarad capacitor)2

LDR is called photocell in daily life. LDR is a circuit element used to measure light density that has two points. The resistance between the two ends changes according to the amount of light falling on it.

LDR is working if it is connected analog enter. However, Raspberry Pi does not have analog enter, so we use a capacitor and setup a RC circuit to work the LDR. RC circuit means a circuit includes capacitor and resistor.



(Model of our RC circuit with Raspberry Pi on Fritzing)



(Real circuit)

The task of the capacitors is to keep the charge. The current passing through the circuit charges the capacitor and causes the voltage to rise. Resistance reduces the current to the capacitor and get longer the charging time. There is a direct relationship between charging time and resistance. Since the resistance of LDR is connected to the direct light, the charging time of

the capacitor is determined by the environment light. When the capacitor is charged to 1.8V, Raspberry Pi will perceive it as logical 1 and GPIO pin logical 1 level will tell us the light level. In our project, x and LDRReading variables in our code demonstrate the time. After the decide charging time of capacitor, it decide environment brightness. If time is smaller or equal than 100 seconds, it tweets 'Is it bright?'. If the time is greater than 100 but smaller than 200 seconds, it tweets 'Normal light :). If the time is greater or equal to 200 seconds, it tweets 'Is it dark?'. Additionally, each parts tweet to charging time.

Our code basically measures the time from the GPIO3 pin to the logical 1. As the amount of light on the LDR increases, the resistance will decrease and the capacitor will charge faster, resulting in a shorter time measurement. After the measurement, Raspberry Pi sends averaged data via Twitter 1 minute for LDR. Data refreshes every 12 seconds. Raspberry Pi gets average of last 5 data and tweet it with installed Twython3. Twython helps us to tweet our data automatically. We enter our keys and access tokens of our Twitter account. Our Twitter account link is https://twitter.com/kume_Olingas18. Our code and its screenshots is below:

```

from twython import *                                     #After install the code from site,
import tweepy                                         #import to our main program
OAUTH_TOKEN = "998223469732139008-g0o0RaE1yisHULWyWYRpMrhkOtIGfm4"
OAUTH_TOKEN_SECRET =
"gmCnefshYhfB7nDCQGJdeAEYLvpZhNToa7pTAiK5Zcz8c"
APP_KEY = "YFfPXg6vhw9KTPQdKQ7CxAq06"
APP_SECRET = "qMUYiRoESAb8INDzmyoraJ5XA40RQMT8tXsTXae9CLKiv7LKYZ"
twitter = Twython(APP_KEY, APP_SECRET, OAUTH_TOKEN,
OAUTH_TOKEN_SECRET)

import RPi.GPIO as GPIO                                #They are used terms in code
and
import time

GPIO.setwarnings(False)
GPIO.setmode(GPIO.BCM)

ldr_pin = 3                                           #It demonstrates LDR is connected
which pins of Raspberry Pi
x=0                                                    #aim of x is add to time end of tweet,aim
of counter is average to last 5 data
counter=0

def RCtime (RCpin):
    reading = 0
    GPIO.setup(RCpin, GPIO.OUT)
    GPIO.output(RCpin, GPIO.LOW)

```

```

time.sleep(12)

GPIO.setup(RCpin, GPIO.IN)
while (GPIO.input(RCpin) == GPIO.LOW):
    reading += 1                                #Calculation to light density
return reading

while True:
    LDRReading = RCtime(3)
    x=x+LDRReading
    if (counter==5):
        x=x/5
        if (x <= 100):                         #Decide to light density and tweet to
brightness of environment
            my_message=('Is it bright?(%g)' % x)
            twitter.update_status(status=my_message)
            counter=0
            x=0
            time.sleep(60)
    elif (x <=200):
        my_message=('Normal light :)(%g)' % x)
        twitter.update_status(status=my_message)
        counter=0
        x=0
        time.sleep(60)
    else:
        my_message=('Is it dark?(%g)' % x)
        twitter.update_status(status=my_message)
        counter=0
        x=0
        time.sleep(60)
    counter=counter+1                           #Power cuts during 5 seconds. After
five seconds program runs again to be stopped.

```

```

from __future__ import *
import RPi.GPIO as GPIO
import time
GPIO.setwarnings(False)
GPIO.setmode(GPIO.BCM)

ldr_pin = 3
counter=0

def RCTime (RCpin):
    reading = 0
    GPIO.setup(RCpin, GPIO.OUT)
    GPIO.output(RCpin, GPIO.LOW)
    time.sleep(0.1)
    GPIO.setup(RCpin, GPIO.IN)
    while (GPIO.input(RCpin) == GPIO.LOW):
        reading += 1
    return reading

while True:
    LDRreading = RCTime(3)
    x=LDRreading
    if (counter==5):
        x=x/5
        if (x <= 100):
            my_message='Is it bright?(%g)' % x
            twitter.update_status(status=my_message)
        counter=0
        x=0
        time.sleep(60)
    elif (x >=200):
        my_message='Normal light :)(%g)' % x
        twitter.update_status(status=my_message)
        counter=0
        x=0
        time.sleep(60)
    else:
        my_message='Is it dark?(%g)' % x
        twitter.update_status(status=my_message)
        counter=0
        x=0
        time.sleep(60)
    counter=counter+1

```

The code is a Python script named ldr_mech.py. It uses the RPi.GPIO library to control an LDR sensor connected to pin 3. The script initializes the LDR pin and sets up a loop. Inside the loop, it calls the RCTime function to read the LDR value. The reading is then averaged over five consecutive measurements. Based on the average brightness, a message is constructed and posted to a Twitter account using the tweepy library. The script includes a sleep command of 60 seconds between each measurement cycle. A final comment indicates a power cut detection mechanism.

(Our code screenshots)

The screenshot shows a Linux desktop environment with a terminal window open in the background. The terminal window has three lines of text:

```
pi@Garn...mech100 ~
File Edit Tabs Help
pi@Garn...mech100 $ scrot
pi@Garn...mech100 $ scrot
pi@Garn...mech100 $ scrot
```

The main window is the Twitter Application Management interface for the application "kume_Olingas18". It displays the following information:

Application Settings

Consumer Key (API Key): YFIPXg6hw9KTPQdKQ7CxA6
Consumer Secret (API Secret): qMUYfRoE5Ab8lNDzmyoraJ5XA40RQMT8tXae9CLKv7LKYZ
Access Level: Read and write (modify app permissions)
Owner: kume_Olingas18
Owner ID: 998223469732139008

Application Actions

Regenerate Consumer Key and Secret | Change App Permissions

Your Access Token

This access token can be used to make API requests on your own account's behalf. Do not share your access token secret with anyone.

Access Token	998223469732139008-gb00RaFLyshHULWVYRpMrhkOtiGfm4
Access Token Secret	gmCnefhSYhfB7nDCQQJdeAEYLvpZhNToa7pTAik5Zcz8c

The terminal window also shows the same three lines of text as above.

(Our tweet account information)

Reference

1. <https://www.cytron.io/p-sn-ldr-s>
2. <https://www.robomart.com/01-microfarad-50v-electrolytic-capacitor>
3. <https://twython.readthedocs.io/en/latest/usage/install.html>
4. <http://maker.robotistan.com/etiket/raspberry-pi-dersleri/>