

**Aprinderea cu diferite intensități a unui LED în funcție de
lumină ambientală și de mișcare cu o placă de dezvoltare
Raspberry Pi 4**

Student: Péntek Tamás

Profesor îndrumător: Lișman Dragoș-Florin

Grupa: 30238

Data: 11.01.2020

Cuprins

1. Rezumat	3
2. Introducere	4
2.1 Interfața de comunicare serială și sincronă	4
2.2 Interfața de comunicare I^2C	5
2.3 Obiective	6
3. Fundamentare Teoretică	7
3.1 Placa de dezvoltare Raspberry Pi 4	7
3.2 Senzor de mișcare	9
3.3 Senzor de lumină ambientală	10
3.4 PWM (Pulse Width Modulation)	11
4. Proiectare și implementare	12
4.1 Conectarea componentelor	12
4.2 Prezentarea programului	14
4.3 Manual de utilizare	15
5. Rezultate experimentale	16
6. Concluzii	19
Bibliografie	19
Anexe	20
Anexa 1	20
Anexa 2	23

1. Rezumat

În ziua de azi suntem înconjurați cu foarte mulți senzori care colectează informații. Acest lucru este folositor pentru că folosind senzori avem posibilitatea de a automatiza niște procese, lucru care a fost imposibil în trecut.

Obiectivul principal al acestui proiect era crearea unui sistem de iluminare automatizat care în funcție de gradul de iluminare și în funcție de mișcare detectată să aprinde un LED cu intensitatea potrivită. Pentru acest sistem am folosit un senzor de mișcare, un senzor de lumină ambientală, niște LEDuri și un buton.

Datele de la senzori au fost prelucrate de placa de dezvoltare Raspberry Pi 4 și limbajul de programare utilizat a fost Python. Folosind aceste componente am reușit să creez un sistem care poate fi utilizat de oricine, fără experiențe în acest domeniu. Folosind senzorii disponibile pe piață și o placă de dezvoltare la un preț redus putem să construim niște sisteme destul de complexe care îmbunătățesc viața fiecărui om și garantează o viață mai confortabilă.

FACULTATEA DE AUTOMATICĂ ȘI CALCULATOARE - CATEDRA CALCULATOARE

2. Introducere

Tema proiectului constă în realizarea unui sistem de iluminare care aprinde un LED cu o anumită intensitate în funcție de lumină și de mișcare. Sensorii și alte componentele se află pe un breadboard, adică pe o placă expansiune pentru prototipuri. Breadboard-ul și placa de dezvoltare Raspberry Pi 4 sunt conectate ușor cu niște fire, iar placa de dezvoltare este alimentată prin cablu USB-Type C. Pe breadboard se află și un buton, care se folosește pentru pornirea și oprirea sistemului, acest lucru este confirmat printr-un LED, care dacă este aprins înseamnă că sistemul funcționează și dacă este stins, sistemul este oprit.

Domeniul de studiu este tehnologia informației care este o tehnologie folosită pentru prelucrarea datelor, informației folosind calculatoare, și automatizarea care are ca și scop construirea unor sisteme care funcționează automat, independent. Un astfel de sistem se poate construi ca un sistem înglobat sau sistem embedded care este o combinație de hardware (de exemplu Raspberry Pi 4) și software specializat pentru rezolvarea sarcinii (de exemplu programul scris în limbajul de programare Python). Acest domeniu este foarte important în ziua de azi pentru că tehnologia s-a dezvoltat foarte mult în ultimii ani și prin acesta au apărut foarte multe oportunități pentru a crea niște sisteme mult mai complexe care rezolvă niște probleme mai avansate, lucru care ajută la dezvoltarea științei și îmbunătățește viața fiecărui om și oferă o viață mai confortabilă pentru fiecare om.

2.1 Interfața de comunicare serială și sincronă

Pentru ca circuitele să-și poată transfera informații trebuie să conțină o modalitate de comunicare comună. Există mai multe modalități de comunicare pe care se poate împărți în două categorii: comunicare serială și paralelă.

Interfețele paralele transferă mai mulți biți în același timp și au nevoie de magistrale de date care transmit mai multe linii. Interfețele seriale transferă informația bit cu bit. Aceste interfețe folosesc doar un singur fir.

O altă modalitate de clasificare a interfețelor de comunicare este după modalitatea de comunicare: sincronă sau asincronă. O interfață de comunicare sincronă folosește un semnal de ceas la ambele capete ale comunicației pentru a sincroniza informația transmisă. În cazul interfeței de comunicare asincronă datele sunt transferate fără folosirea unui semnal de ceas. Cele mai importante interfețe de comunicare serială și sincronă sunt SPI (Serial Peripheral Interface) și I²C (Inter-integrated Circuit).

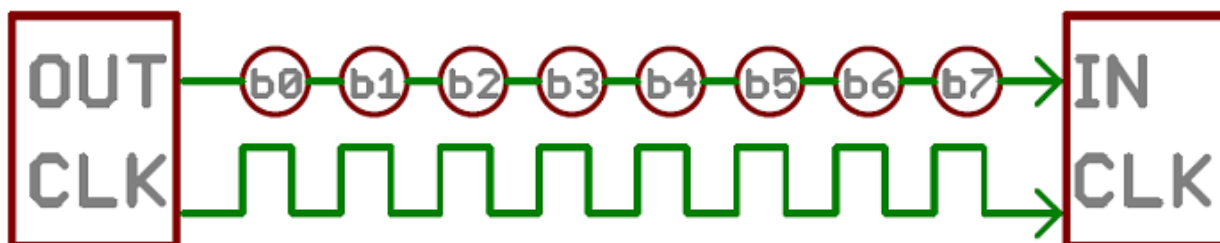


Figura 1. Transmisia serială și sincronă

2.2 Interfața de comunicare I^2C

Interfața de comunicare I^2C a fost inventat și dezvoltat de Philips și este a fost creat pentru a permite mai multe circuite integrate „slave” să comunice cu unul sau mai multe chip-uri „master”. Acest tip de interfață este destinat numai comunicațiilor pe distanțe mici și are nevoie doar de 2 fire de semnal pentru a trimite/primi informația. I^2C permite comunicarea între mai mult de două dispozitive. Unul dintre dispozitive va fi master, iar el va comunica cu un slave, dar ulterior rolurile se pot schimba.

Fiecare magistrală I^2C este compus din 2 semnale: SCL și SDA. SCL reprezintă semnalul de ceas, iar SDA semnalul de date. Semnalul de ceas este generat întotdeauna la partea de master. Ambele semnale sunt bidirecționale și sunt conectate prin rezistențe Pull-Up la o tensiune de alimentare pozitivă, înseamnănd că atunci când magistrala este liber, ambele linii sunt pe „1” (High) logic.

O transmisie de date începe cu o condiție de START și se termină cu o condiție de STOP. O condiție de START este o trecere al lui SDA de la 1 logic la 0 logic, cât timp SCL are valoare 1 logic. Condiția de STOP este o trecere al lui SDA de la 0 logic la 1 logic, cât timp semnalul SCL este menținut pe 1 logic. Pentru fiecare impuls de ceas se transmite un singur bit. Semnalul SDA se poate modifica numai când SCL este în starea 0 logic, iar când SCL este în starea 1 logic datele de pe linia SDA trebuie să fie stabile. Informația transmisă este spartă în două tipuri de cadre (frames): cadre de adresă, în acest caz dispozitivul master indică dispozitivul slave la care informația va fi trimisă; unul sau mai multe cadre de date care conțin informația pe 8 biți (octeți) trimise de la dispozitivul master la dispozitivul slave sau invers.

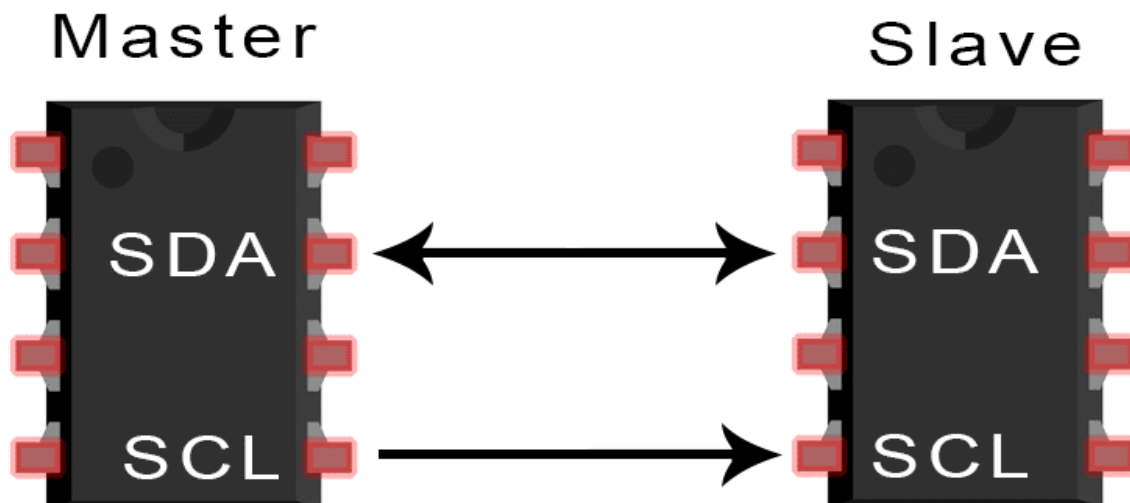


Figura 2. Comunicația I^2C

2.3 Obiective

Obiectivul principal al proiectului este aprinderea unui LED cu o anumită intensitate în funcție de intensitatea măsurată de senzorul de lumină ambientală și în funcție de mișcare detectată de senzorul de mișcare.

După ce sistemul este pornit cu un buton aflat pe breadboard, sistemul devine activ și senzorul de mișcare trimite către placa de dezvoltare Raspberry Pi 4 un semnal de 1 (High) logic dacă a detectat o mișcare. În același timp senzorul de lumină ambientală măsoară intensitatea luminoasă, gradul de iluminare și trimite către placa de dezvoltare, folosind interfața de comunicație serială și sincronă, numită I^2C , valoare măsurată în lux. După aceea programul de pe placa de dezvoltare calculează pe baza valorii măsurată de senzorul de lumină ambientală intensitatea potrivită cu care se aprinde LEDul. Modificarea intensității LEDului se face folosind tehnica PWM (Pulse Width Modulation).

În ceea ce urmează, următorul capitol va conține detalii despre placa de dezvoltare, despre senzori, despre modul cum este modificat intensitatea LEDului. În capitolul „Proiectare și implementare” vom discuta despre pașii necesari pentru implementarea proiectului, despre soluția aleasă și despre algoritmi implementați. Capitolul „Rezultate experimentale” va descrie

FACULTATEA DE AUTOMATICĂ ȘI CALCULATOARE - CATEDRA CALCULATOARE

rezultatele obținute și instrumentele de proiectare utilizate, iar ultimul capitol va fi o concluzie, un sumar al proiectului și a rezultatelor obținute.

3. Fundamentare Teoretică

După cum am menționat anterior, placa de dezvoltare aleasă a fost Raspberry Pi 4, la care s-au conectat mai mulți senzori. Se poate găsi mai multe tipuri de senzori de mișcare și de lumină ambientală, în cadrul acestui proiect a fost ales senzorul de lumină ambientală MAX44009, care folosește protocolul I^2C , și senzorul de mișcare HC-SR501. Acest capitol va conține detalii despre aceste dispozitive, folosite la implementarea proiectului.

3.1 Placa de dezvoltare Raspberry Pi 4

Raspberry Pi este o serie de SBC (Single-board computer) de dimensiunile unui card de credit și este produs de către Raspberry Pi Foundation. Scopul a fost acela de a crea un dispozitiv cu costuri reduse care să îmbunătățească abilitățile de programare și înțelegerea hardware. Raspberry Pi este mai lent decât un laptop sau PC, dar poate oferi majoritatea aplicațiilor acestora precum conectare la internet, redare de conținut video/audio, jocuri video, la un nivel de consum redus de energie[3].

În plus, Raspberry Pi are o caracteristică specială: port generic de intrare/ieșire (General-Purpose Input/Output)(GPIO). Acesta oferă posibilitatea de a conecta diverse componente electronice specifice sistemelor înglobate: de exemplu senzori, butoane, ecran LCD [3].

Raspberry Pi 4 (Figura 3.) are 40 de pini GPIO (Figura 4.) care permite conectarea mai multor componente electronice în același timp care ajută la crearea unor proiecte mai complexe. Varianta folosită la acest proiect are 4 GB de RAM, un procesor Broadcom BCM2711, un slot card microSD pentru rularea sistemului de operare și pentru stocare de date și două porturi de micro-HDMI. Alimentarea plăcii se poate face conectând la un calculator/ laptop și folosind un cablu USB-Type C.

Placa are două pini de alimentare 5V, două pini de alimentare 3,3 V și opt pini de masă (Ground) ceea ce permite alimentarea mai multor componente. Pe lângă acesta, Raspberry Pi 4 are pini pentru interfața de comunicare de tip I^2C , SPI, UART (Universal Asynchronous Receiver/Transmitter), din care rezultă că această placă este compatibilă cu orice componentă electronică, cu orice senzor care folosește una dintre interfețele de comunicare descrise mai sus.

FACULTATEA DE AUTOMATICĂ ȘI CALCULATOARE - CATEDRA CALCULATOARE



Figura 3. Raspberry Pi 4


 Raspberry Pi Pinout		
3v3 Power	1	2 5v Power
BCM 2 (SDA)	3	4 5v Power
BCM 3 (SCL)	5	6 Ground
BCM 4 (GPCLK0)	7	8 BCM 14 (TXD)
Ground	9	10 BCM 15 (RXD)
BCM 17	11	12 BCM 18 (PWM0)
BCM 27	13	14 Ground
BCM 22	15	16 BCM 23
3v3 Power	17	18 BCM 24
BCM 10 (MOSI)	19	20 Ground
BCM 9 (MISO)	21	22 BCM 25
BCM 11 (SCLK)	23	24 BCM 8 (CE0)
Ground	25	26 BCM 7 (CE1)
BCM 0 (ID_SD)	27	28 BCM 1 (ID_SC)
BCM 5	29	30 Ground
BCM 6	31	32 BCM 12 (PWM0)
BCM 13 (PWM1)	33	34 Ground
BCM 19 (MISO)	35	36 BCM 16
BCM 26	37	38 BCM 20 (MOSI)
Ground	39	40 BCM 21 (SCLK)

Figura 4. Pinii GPIO

3.2 Senzor de mișcare

Senzorul de mișcare folosit în acest proiect este un senzor PIR HC-SR501. Acest senzor este utilizat pentru a detecta prezența oamenilor. Este des utilizat în aprinderea sau stingerea automată a luminii într-o încăpere atunci când o persoană ajunge sau părăsește o încăpă.

Senzorul este bazat pe tehnologia IR (Infrared Radiation): senzorul detectează radiații în infraroșu din mediu. Odată ce există o radiație în infraroșu emisă de corpul uman, senzorul generează un semnal de ieșire 1 logic, iar când nu există nicio mișcare, senzorul generează un semnal de ieșire 0 logic. Acesta este format dintr-un senzor piroelectric, care este capabil să detecteze diferite niveluri de radiații infraroșii.

Suportă o tensiune de alimentare mai ridicată (5-20 V) și are un consum mai mare, iar raza de sensibilitate este de 110 grade și detectează obiecte până la o distanță de 7 m. Capacul alb funcționează ca o lentilă divergentă ce îi permite senzorului să detecteze obiecte pe o rază mai mare. Senzorul face citiri la intervale de timp, cuprinse între 0,3 s și 5 min. Distanța și timpul pot fi ajustate folosind șuruburile de pe senzor.

Senzorul de mișcare se conectează la placa de dezvoltare Raspberry Pi 4 cu trei fire: un fir trebuie legat la Vcc (5 V), un alt fir la GND (0 V) și al treilea fir la un pin digital, de pe care se face citirea valorii transmise de senzor.



Figura 5. Senzorul de mișcare HC-SR501

3.3 Senzor de lumină ambientală

Senzorul folosit pentru măsurarea gradului de iluminare, a luminii ambientală este un senzor MAX44009. Acest senzor detectează lumina ambientală și are o interfață de comunicare I^2C pentru a putea fi folosit ușor cu diverse microcontrolere și are un consum redus de energie. Acest senzor detectează lumina asemenea ochiului uman deoarece dispune de filtre IR și UV (Ultraviolet).

Acest senzor are o gamă dinamică foarte largă de 22 de biți de la 0,045 lux până la 188.000 lux. Cea mai frecventă utilizare în viața noastră de zi cu zi al acestui senzor este în telefoanele mobile și tablete. Cele mai multe telefoane mobile au acum senzori de lumină ambientală folosiți pentru a regla luminozitatea. Dacă dispozitivul se află într-un loc întunecat, acesta scade luminozitatea ecranului pentru a economisi energie.

Senzorul de lumină ambientală folosește o componentă numită fotodiodă pentru a măsura iluminarea. O fotodiodă este un fotodetector capabil să convertească lumina în curent. În funcție de intensitatea și lungimea de undă a radiației, fotodioda dă un răspuns în curent. Acest curent este prelucrat de un circuit de putere, care salvează rezultatul într-un bitstream. După aceea rezultatul este procesat digital și stocat într-un registru de ieșire din care este citit prin interfața de comunicare I^2C .

În interiorul acestui senzor avem mai mulți regiștri, dintre care cei mai importanți sunt registrul Lux High Byte și registrul Lux Low Byte. Registrul Lux Low Byte oferă patru cel mai puțin semnificativi biți a mantisei, iar Registrul Lux High Byte oferă patru biți al exponentului și patru cel mai semnificativi biți a mantisei. După citirea acestor regiștri aplicăm o formulă și avem rezultatul final în lux.

Acest senzor are 6 pini: Vcc (5 V), GND (0 V), SCL, SDA, INT și A0. Pinul Vcc și GND se conectează la placa de dezvoltare pentru a alimenta senzorul. Pinul SCL este semnalul de ceas, iar pinul SDA este semnalul de date a interfeței de comunicare I^2C . Pinul INT se folosește pentru întreruperi, dar în acest proiect nu avem nevoie, de aceea nu conectăm la placa de dezvoltare. Pinul A0 se folosește pentru a selecta adresa dispozitivului slave: dacă se leagă la GND, adresa selectată va fi 1001 010x și dacă se leagă la Vcc, adresa selectată va fi 1001 011x.

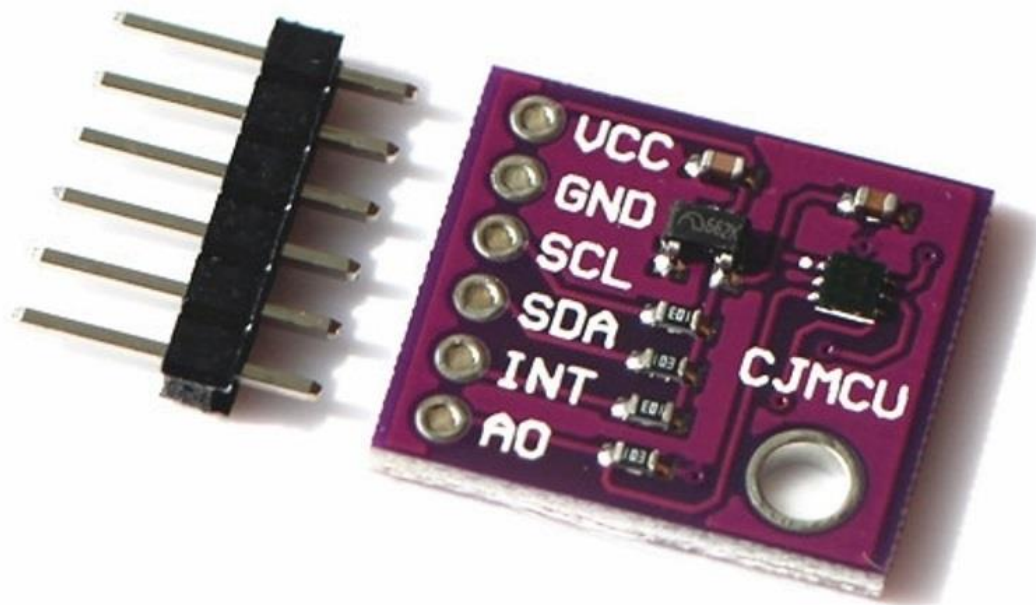


Figura 6. Senzorul de lumină ambientală MAX44009

3.4 PWM (Pulse Width Modulation)

PWM (modulație prin lățimea pulsului) este o metodă de a obține un semnal analogic prin intermediul unui semnal digital, alternând periodic ieșirea între nivelul logic 1 și nivelul logic 0. PWM este o tehnică de modulare care este utilizată pe scară largă pentru controlul puterii. Este frecvent utilizat în aplicațiile de control a motorului pentru a controla viteza unui motor sau luminozitatea, intensitatea unui LED. Un semnal PWM este o formă de undă pătrată și are doi parametri importanți: frecvența și factor de umplere. Frecvența este numărul de cicluri repetate pe care le face o formă de undă într-o secundă. Frațiunea de perioadă cât semnalul este activ (1 logic) se numește factor de umplere (duty cycle).

Există două tipuri de PWM: hardware PWM și software PWM. Hardware PWM-urile sunt restricționate de generatoarele PWM incluse în placa de dezvoltare Raspberry Pi 4. Acest lucru înseamnă că doar anumiți pini au posibilitatea de a genera un semnal PWM, deoarece numărul generatoarelor de PWM este limitat. Cu software PWM teoretic, fiecare pin de ieșire poate fi utilizat ca semnal PWM. În acest caz generarea semnalelor se rezolvă din program, din software și de aceea nu sunt limitați, orice pin digital are posibilitatea de a genera un semnal PWM.

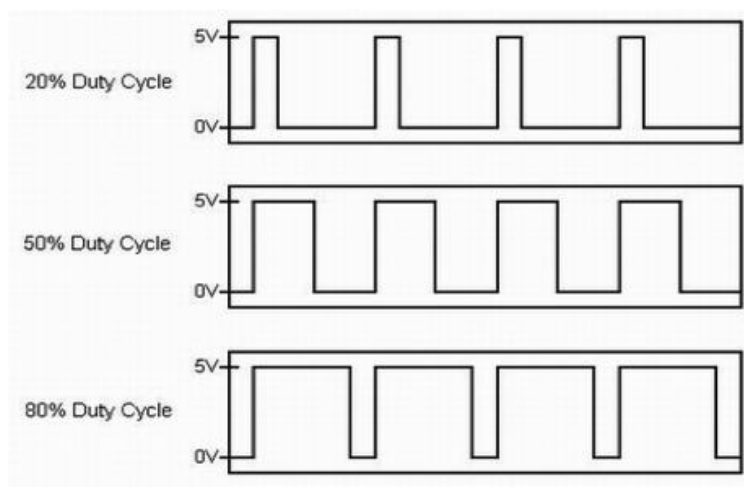


Figura 7. Diferiți factori de umplere unui semnal PWM

4. Proiectare și implementare

În acest capitol se prezintă implementarea hardware și software a proiectului ales: modul cum s-a făcut conectarea componentelor, prezentarea programului care este executat pe placa de dezvoltare Raspberry Pi 4 și schema bloc cu toate semnale de comandă.

4.1 Conectarea componentelor

Primul pas a fost conectarea plăcii de dezvoltare Raspberry Pi 4 cu un calculator/laptop, aceasta se face folosind un cablu USB Type C. După ce placa este conectată și este pregătit sistemul de operare pe cardul microSD facem conexiunea cu placa cu ajutorul software-ului VNC Viewer, prin introducerea adresei de IP și parola plăcii de dezvoltare. Dacă conexiunea s-a făcut cu succes, apare Desktop-ul sistemului de operare în programul menționat mai sus.

Toți senzorii, LEDurile și butonul punem pe un breadboard pentru a avea o conexiune mai bună și mai sigură cu placa de dezvoltare. Senzorul de mișcare are trei pini: pinul Vcc conectăm cu un pin Vcc de pe placă, pinul GND conectăm cu un pin GND, iar pinul de ieșire a sensorului conectăm cu un pin digital a plăcii. În cazul sensorului de lumină ambientală procedăm la fel: alimentăm sensorul cu pinii Vcc și GND, dar trebuie să avem grijă, pentru că

FACULTATEA DE AUTOMATICĂ ȘI CALCULATOARE - CATEDRA CALCULATOARE

acest senzor are nevoie o tensiune mai mică decât 5V, de aceea pinul de Vcc legăm la un pin de Vcc de 3,3 V ca să nu ardem senzorul. După aceea legăm pinii interfeței de comunicare I^2C : pinul SCL legăm la pinul SCL a plăcii, adică la pinul 5, iar pinul SDA legăm la pinul SDA a plăcii, adică la pinul 3. Rămână doar să legăm ultimul pin al acestui senzor, pinul A0 care este folosit pentru alegerea adresei dispozitivului slave: prin legarea acestui pin la GND alegem adresa slave-ului pentru scriere: 1001 0100 și pentru citire: 1001 0101. Pinul INT nu conectăm, deoarece nu avem nevoie de întreruperi în cadrul acestui proiect.

Urmează să conectăm LEDul la locul potrivit. Fiindcă vom folosi Software PWM pentru a regla intensitatea LEDului, putem să conectăm LEDul la orice pin digital. Catodul LEDului legăm la un pin de tip GND, iar anodul este conectat prima dată la o rezistență de 330 Ohm, pentru ca să protejăm LEDul, și după aceea este conectat la un pin digital al plăcii de dezvoltare. Facem același conexiune și în cazul celui alt LED, numai că conect la un alt pin digital. Rămână doar să conectăm butonul cu care pornim și oprim sistemul: butonul are 4 pini, dintre care folosim doar două, un pin va fi conectat la GND și celălalt la un pin digital. După conectarea tuturor componentelor montajul arată ca în Figura 8, iar schema completă, detaliată este prezentată în Anexa 2.

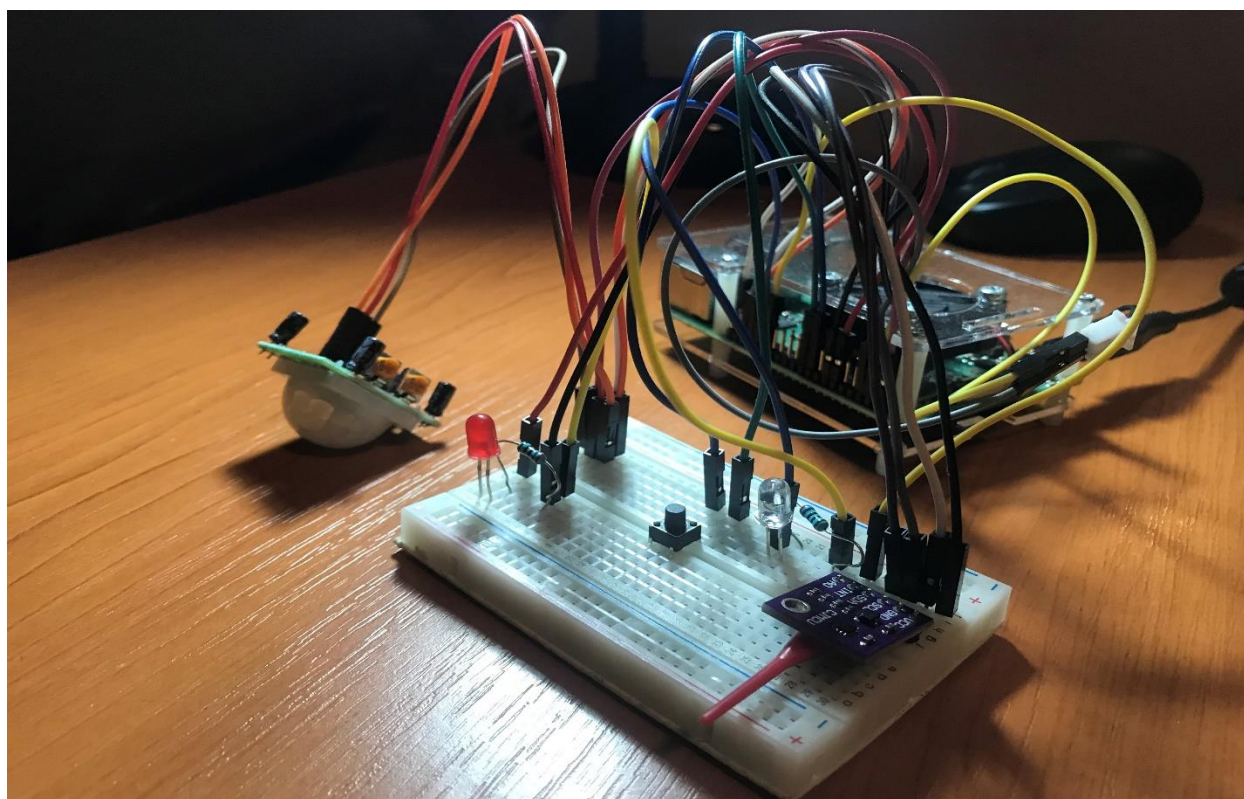


Figura 8. Montajul complet

4.2 Prezentarea programului

Codul sursă complet se poate găsi la sfârșitul acestei documentații în Anexa 1. Programul care rulează pe placa de dezvoltare a fost scris în limbajul de programare Python, care este limbajul de programare inițial al plăcii de dezvoltare. Python este un limbaj de programare destul de puternic, ușor de utilizat (ușor de citit și de scris) și simplifică implementarea proiectelor făcute cu placa de dezvoltare Raspberry Pi. Sintaxa Python este foarte curată, cu mare accent pe lizibilitate.

Codul începe cu importare unor biblioteci care sunt esențiale pentru funcționarea corectă a programului. Biblioteca `smbus` este folosită pentru interfața de comunicare I^2C , biblioteca `time` se folosește pentru delay-uri, iar `RPi.GPIO` pentru a accesa fiecare pin de pe placa de dezvoltare.

În codul sursă sunt definite trei metode importante. Metoda **`calculateLux`** calculează și returnează valoarea de lux măsurată de senzorul ambientală de lumină. În variabila `lux_HIGH` salvăm exponentul și partea cea mai semnificativă a mantisei, iar în variabila `lux_LOW` salvăm partea cea mai puțin semnificativă a mantisei. Metoda `I2C_bus.read_i2c_block_data` citește de pe senzorul cu adresa `SENSOR_ADDRESS` valoarea din registrul High și apoi din registrul Low. Fiecare registru are o dimensiune de 8 biți. Fiindcă exponentul se află pe cele mai semnificative patru biți, trebuie deplasat la dreapta cu 4 poziții. Partea cea mai semnificativă a mantisei și partea cea mai puțin semnificativă a mantisei se află pe cele mai puțin semnificative patru biți de aceea prima dată se face un ȘI logic între această valoare și `0x0f` pentru a obține valoarea corectă. După aceea partea cea mai semnificativă a mantisei este deplasată cu 4 poziții la stânga după care se face un ȘI logic cu partea cea mai puțin semnificativă a mantisei și în acest fel se obține valoarea finală a mantisei în variabila `mantisa_FULL`. După ce programul a calculat exponentul și mantisa se aplică formula următoare $2^{\text{exponent}} * \text{mantisa} * 0,045$, iar rezultatul acestei formule este valoarea finală măsurată de senzorul de lumină ambientală în lux.

Metoda **`calculateDutyCycle`** are un parametru care reprezintă valoarea măsurată de senzorul de lumină ambientală în lux. Această metodă calculează factorul de umplere cu care se aprinde LEDul. Dacă lumina ambientală are o valoare mare, metoda returnează un factor de umplere mai mic, iar dacă valoarea luminii ambientale este mai scăzută, factorul de umplere devine mai mare.

Metoda **`turnOffSystemLed`** este folosită pentru LEDul care arată dacă sistemul este pornit sau nu. Când utilizatorul oprește sistemul, acest LED are o animație controlată de această metodă.

FACULTATEA DE AUTOMATICĂ ȘI CALCULATOARE - CATEDRA CALCULATOARE

După aceste metode urmează partea propriu-zisă, partea de main a programului. Prima dată este inițializat fiecare pin digital utilizat în cadrul acestui proiect și este creat o magistrală de comunicare de tip I^2C . Apoi este setat modul de numerotare folosit în acest program. Fiecare pin poate fi numerat în două moduri: BOARD și BCM, în acest program este folosit modul BCM. După aceea este setat tipul fiecărui pin: de tip INPUT sau OUTPUT și este creat o variabilă de tip Software PWM: pwmLED. La pinul butonului atașăm un rezistor de tip Pull-Up, acest rezistor este conectat la 5V (Vcc). Asta înseamnă că valoarea citită de la buton, când nu este apăsat va fi 1 logic, adică o cantitate mică de curent se scurge între VCC și pinul de intrare, așadar acest pin de intrare citește o valoare aproape de VCC. În momentul în care apăsăm butonul, pinul de intrare se conectează direct la ground. Curentul trece prin rezistor și ajunge la ground făcând pinul de intrare să citească valoarea 0 logic.

Urmează o buclă while care este partea principală a programului. Aici se citește valoarea de la pinul de ieșire a senzorului de mișcare și valoarea pinului butonului. Dacă butonul a fost apăsat, se setează un flag și sistemul pornește. Cât timp nu este apăsat butonul încă o dată, sistemul citește continuu ieșirea senzorului de mișcare și valoare luminii ambientale. Când senzorul de mișcare detectează o mișcare este calculată valoarea factorului de umplere și în acest moment este aprins LEDul cu intensitatea potrivită și se afișează un mesaj " ! MOTION DETECTED !" . LEDul rămâne aprins câteva milisecunde după care se stinge dacă senzorul nu detectează nicio mișcare. Această buclă se repetă până când utilizatorul apasă butonul. Acest lucru este verificat cu flag-ul countFlag. În acest caz este setat flagul turnOFF la 1, ceea ce are ca și efect ieșirea programului din bucla while după care este apelată metoda turnOffSystemLed care execută animația de oprire a LEDului sistemului.

Programul poate fi rulat folosind terminalul sistemului de operare de pe placa de dezvoltare sau prin rularea fișierului executabil denumit „SSCproject” de pe Desktop.

4.3 Manual de utilizare

Pentru ca utilizatorul să poată rula și verifica funcționalitatea proiectului, acesta trebuie să urmeze următorii pași:

- a) Utilizatorul trebuie să instaleze programul VNC Viewer în care deschide sistemul de operare a plăcii sau trebuie să conecteze placa la un monitor
- b) Dacă utilizează VNC Viewer trebuie să introducă la username „pi” și la password „raspberryl23”
- c) Se deschide de pe Desktop-ul sistemului fișierul „SSCproject” și se alege butonul „Execute in Terminal”
- d) Se apasă butonul care se află pe mijlocul breadboard-ului după care sistemul pornește și LEDul albastru se aprinde

FACULTATEA DE AUTOMATICĂ ȘI CALCULATOARE - CATEDRA CALCULATOARE

- e) Se verifică funcționalitatea programului
- f) Se apasă butonul pentru a opri programul
- g) Pentru a deconecta placa de la calculator, se închide programul VNC Viewer și apoi se deconectează cablul

5. Rezultate experimentale

În cazul acestui proiect nu avem un simulator sau un program în care este posibilă testarea funcționalității programului. De aceea testarea propriu-zisă și completă se poate face numai când programul și sistemul este gata făcut și este pus în funcționare. În ceea ce urmează se va prezenta câteva screenshot-uri făcute în programul VNC Viewer când placa de dezvoltare a fost conectată la un laptop și a fost rulat programul.

Pe Figura 9. este prezentat starea programului după ce a fost pornit sistemul și a fost rulat fișierul SSCproject. În acest moment apare în program un mesaj care spune utilizatorului să apese butonul de pe placa de dezvoltare pentru a porni sistemul.

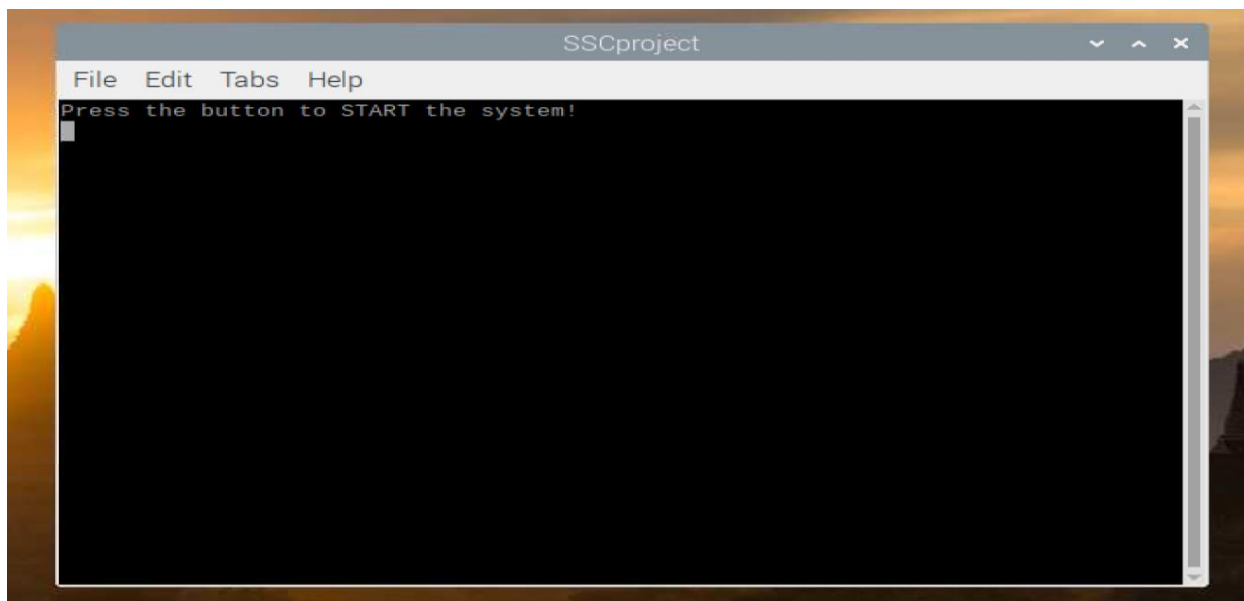


Figura 9. Starea programului imediat după ce a fost rulat

Pe Figura 10. se vede starea programului după ce utilizatorul a apăsă butonul de pe breadboard. În acest moment programul începe să măsoare valoarea luminii ambientale și afișează pe ecran această valoare măsurată în lux.

FACULTATEA DE AUTOMATICĂ ȘI CALCULATOARE - CATEDRA CALCULATOARE

Pe Figura 11. este prezentat mometul când senzorul de mișcare a detectat o mișcare. Acest lucru este evidențiat prin afișarea mesajului „! MOTION DETECTED !”.

Figura 12. prezintă starea programului după ce utilizatorul a apăsă din nou butonul, acest lucru însemnând oprirea programului. Programul, înainte de a opri, mai afișează un mesaj: „Sistem powered off!”.

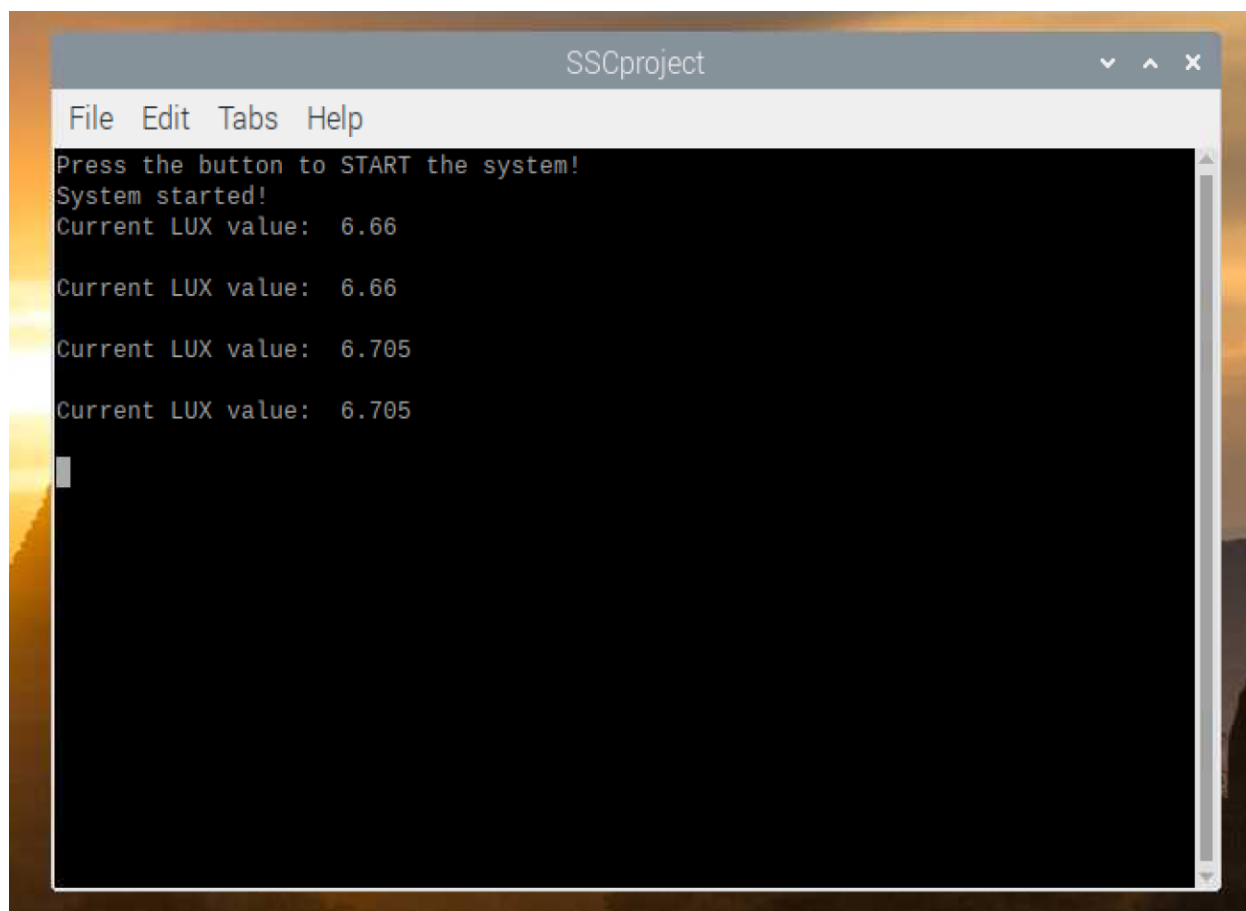


Figura 10. Starea programului după ce utilizatorul a apăsă butonul

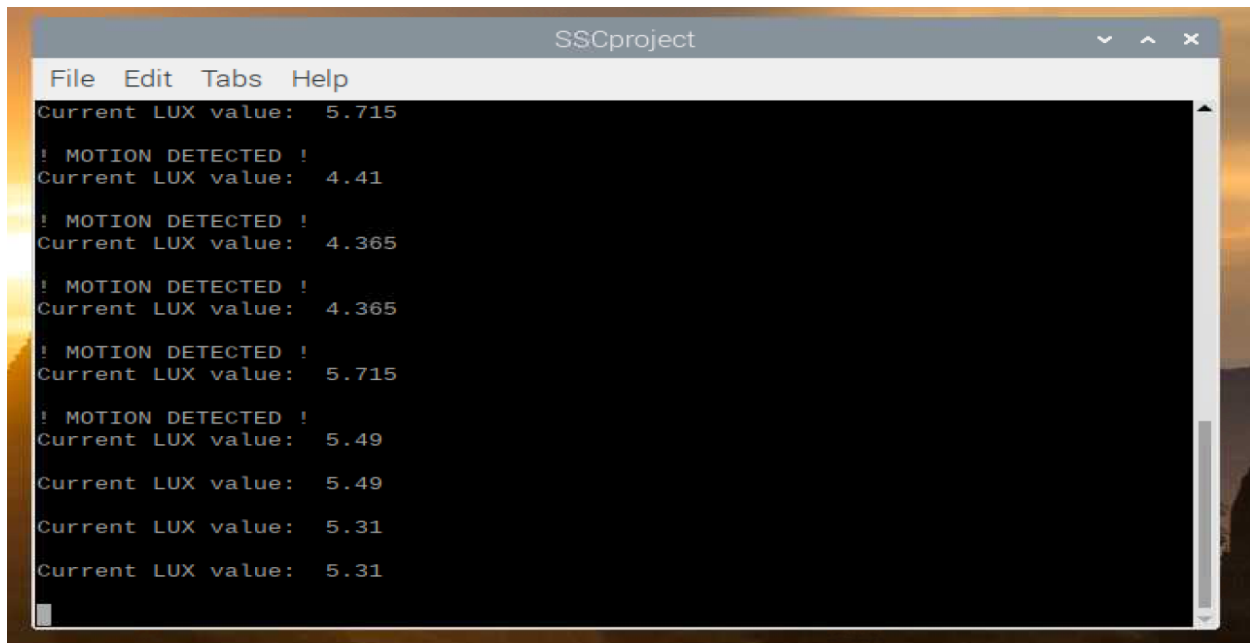


Figura 11. Momentul când senzorul de mișcare detectează o mișcare

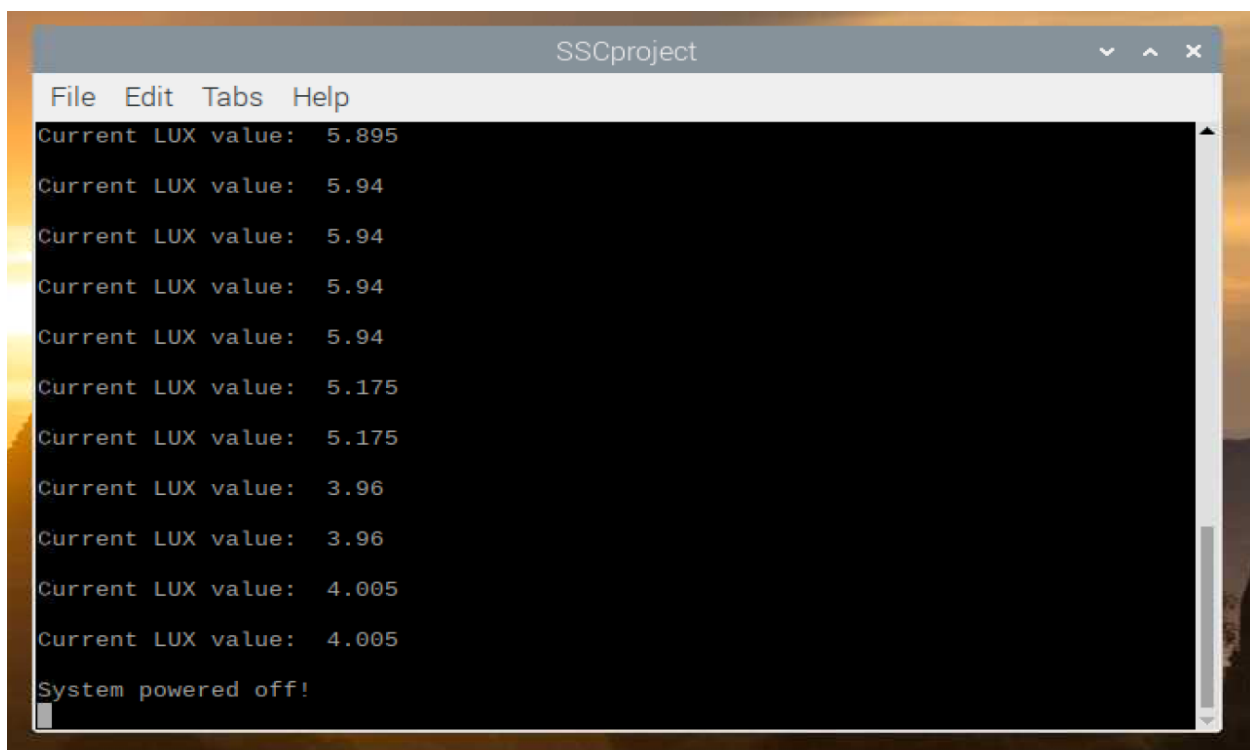


Figura 12. Momentul când utilizatorul apasă din nou butonul

6. Concluzii

În cadrul acestui proiect s-a realizat implementarea unui sistem care este folosit pentru măsurarea luminii ambintelă, detectarea mișcării și în funcție de aceste două lucruri acest sistem aprinde un LED cu intensitatea potrivită. Prin implemetarea acestui proiect am dobândit următoarele cunoștințe: folosirea plăcii Raspberry Pi 4, conectarea componentelor electronice cu o placă de dezvoltare și controlarea acestora cu un program, preluarea de date de la un senzor de mișcare și de lumină ambiantă, aprofundarea folosirii limbajului de programare Python și înțelegerea modului de funcționare a interfeței de comunicare I^2C .

Acest sistem, implementat în cadrul acestui proiect rezolvă o problemă din viața de zi cu zi: consumarea excesivă a energiei electrice. Prin implemetarea acestui sistem automat folosim doar energia necesară în timpul zilei și în timpul nopții. Fiindcă acești senzori au un consum redus de energie, nici sistemul în sine nu consumă o cantitate mare de energie. Dezavantajul este că trebuie să cumpărăm placa de dezvoltare, care este o sumă mai mare, dar dacă se alege o placă mai ieftină, problema se rezolvă.

Ca dezvoltare ulterioară s-ar putea realiza un sistem mult mai complex, cu o cameră video și cu o aplicație Android. Camera video ar fi folosită pentru a face o poză despre persoana care a apărut în fața senzorului de mișcare și această poză ar fi trimisă de sistem cu ajutorul modului de Bluetooth integrat către un telefon mobil care cu ajutorul unei aplicații Android ar afișa această poză.

Bibliografie

- [1] Interfață serială de comunicare
<https://learn.sparkfun.com/tutorials/serial-communication>
- [2] Interfața de comunicare I^2C
<https://learn.sparkfun.com/tutorials/i2c>
<http://www.circuitbasics.com/basics-of-the-i2c-communication-protocol/>
- [3] Raspberry Pi
https://ro.wikipedia.org/wiki/Raspberry_Pi
- [4] Raspberry Pi 4
<https://www.raspberrypi.org/products/raspberry-pi-4-model-b/specifications/>
- [5] Pinii pe Raspberry Pi 4
<https://maker.pro/raspberry-pi/tutorial/raspberry-pi-4-gpio-pinout>

FACULTATEA DE AUTOMATICĂ ȘI CALCULATOARE - CATEDRA CALCULATOARE

[6] Senzor de mișcare

<https://learn.adafruit.com/pir-passive-infrared-proximity-motion-sensor/overview>

<https://www.seeedstudio.com/blog/2019/08/03/pir-sensor-introduction-and-how-pir-motion-sensor-works-with-arduino-and-raspberry-pi/>

[7] Senzor de lumină ambinetală

<https://blog.endaq.com/how-light-sensors-work>

[8] PWM

<https://www.embedded-computing.com/embedded-computing-design/pwm-control-with-the-raspberry-pi-2>

Anexe

Anexa 1

Codul sursă complet

```
import smbus
import time
import sys
import RPi.GPIO as GPIO

def calculateLux():
    LUX_HIGH_BYTE_REGISTER = 0x03
    LUX_LOW_BYTE_REGISTER = 0x04
    SENSOR_ADDRESS = 0x4a

    lux_HIGH = I2C_bus.read_i2c_block_data(SENSOR_ADDRESS,
    LUX_HIGH_BYTE_REGISTER)
    exponent = (0xf0 & lux_HIGH[0]) >> 4
    lux_LOW = I2C_bus.read_i2c_block_data(SENSOR_ADDRESS,
    LUX_LOW_BYTE_REGISTER)
    mantisa_HIGH = 0x0f & lux_HIGH[0]
    mantisa_LOW = 0x0f & lux_LOW[0]
    mantisa_FULL = (mantisa_HIGH << 4) | mantisa_LOW
    lux_final = 2 ** exponent * mantisa_FULL * 0.045
    return lux_final
```

FACULTATEA DE AUTOMATICĂ ȘI CALCULATOARE - CATEDRA CALCULATOARE

```
def calculateDutyCycle(luxValue):
    value = 0
    unit = 1880

    if luxValue >= 0.045 and luxValue <= 188000:
        value = luxValue / unit
    elif luxValue < 0.045:
        value = 0
    elif luxValue > 188000:
        value = 100
    return (100 - int(round(value)))

def turnOffSystemLed():
    GPIO.output(12, GPIO.LOW)
    time.sleep(0.5)
    GPIO.output(12, GPIO.HIGH)
    time.sleep(0.1)
    GPIO.output(12, GPIO.LOW)
    time.sleep(0.2)
    GPIO.output(12, GPIO.HIGH)
    time.sleep(0.1)
    GPIO.output(12, GPIO.LOW)

systemLED = 12
functionalLED = 18
PIRSensorPIN = 16
buttonPIN = 24
I2C_bus = smbus.SMBus(1)
GPIO.setmode(GPIO.BCM)
GPIO.setwarnings(False)
GPIO.setup(functionalLED, GPIO.OUT)
GPIO.setup(systemLED, GPIO.OUT)
GPIO.setup(PIRSensorPIN, GPIO.IN)
pwmLED = GPIO.PWM(functionalLED, 100)
pwmLED.start(0)
GPIO.setup(buttonPIN, GPIO.IN, pull_up_down=GPIO.PUD_UP)
countFlag = 0
turnOFF = 0
turnON = 0
```

FACULTATEA DE AUTOMATICĂ ȘI CALCULATOARE - CATEDRA CALCULATOARE

```
print "Press the button to START the system!"

while not turnOFF:
    PIRSensorInput = GPIO.input(16)
    button = GPIO.input(24)
    if not button:
        countFlag += 1
        time.sleep(0.5)
    if countFlag == 1:
        if turnON == 0:
            print "System started!"
            GPIO.output(systemLED, GPIO.HIGH)
            time.sleep(1)
            turnON = 1
        lux = calculateLux()
        print "Current LUX value: ", lux, "\n"
        time.sleep(0.4)

        if PIRSensorInput == 1:
            value = calculateDutyCycle(lux)
            print "! MOTION DETECTED !"
            GPIO.output(systemLED, GPIO.LOW)
            time.sleep(0.2)
            GPIO.output(systemLED, GPIO.HIGH)
            pwmLED.ChangeDutyCycle(value)
        else:
            pwmLED.ChangeDutyCycle(0)

    if countFlag == 2:
        time.sleep(0.5)
        print "System powered off!"
        GPIO.output(systemLED, GPIO.LOW)
        countFlag = 0
        turnOFF = 1

turnOffSystemLed()
```

Anexa 2

Schema detaliată

