Universitatea
Transilvania
din Brașov
FACULTATEA DE INGINERIE ELECTRICĂ
ȘI ȘTIINȚA CALCULATOARELOR

Casă inteligentă

Studenți:
Sârcă Florin-Sabin
Zaharia George-Andrei
An: IV
Grupa: 4LF772
Email:
george.zaharia@student.unitbv.ro
florin.sarca@student.unitbv.ro

Conținut

1 Rezumatul Lucrării 3
2 Introducere 4
3 Arhitectura hardware 5
3.1 Arduino UNO 5
3.1.1 Tensiune de alimentare 6
3.1.2 Memorie 6
3.1.3 Input și Output 7
3.2 Senzorul de temperatură 8
3.2.1 Detalii tehnice 8
3.2.2 Aplicație practică 9
3.2.3 Alimentare 9
3.3 Senzorul de mișcare 9
3.3.1 Detalii tehnice 10
3.3.2 Aplicație practică 10
3.4 Senzorul de lumină 11
3.4.1 Detalii tehnice 11
3.4.2 Aplicație practică 11
3.5 Modulul Bluetooth HC-05 12
3.5.1 Detalii tehnice 12
3.5.2 Aplicație practică 12
3.6 Led RGB 13
3.6.1 Detalii tehnice 13
3.6.2 Aplicație practică 14
3.7 Senzor Gaz MQ2 14
3.7.1 Aplicație practică 14
3.8 Senzor CO MQ7 15
3.8.1 Detalii tehnice 15
3.8.2 Aplicație practică 15
3.9 Modul LCD I2C 16
3.9.1 Detalii tehnice 16
3.9.2 Aplicație practică 16

3.10 Power Supply Pentru Breadboard 17
4 Bill of Materials 18
5 Arhitectura Software / Firmware 19
5.1 Sistem de Operare 19
5.2 Limbaj de Programare 19
5.3 Medii De Dezvoltare 19
5.4 Compilator 19
5.5 Librării 19
5.6 Schemă Logică 20
5.1 Cod Arduino 21
5.2 Statistică 22
5.3 Structurarea Lucrării 22
5.3.1 Schemă Bloc 22
5.3.2 Schemă Electrică 23
5.3.3 Schemă de Conectare 23
6 Montaj 24
7 Concluzii 24
8 Bibliografie 25
9 Cod Sursă 26

1 Rezumatul Lucrării

În acest proiect am dorit să realizam o casă modernă care are un nivel de confort mai ridicat decât al unei locuințe clasice.

Implementarea a fost realizată pe placa de dezvoltare Arduino UNO unde s-au efectuat măsurători și cu ajutorul IDE-ului am putut programa și testa senzorii.

În funcție de anumiți parametri măsurați de acești senzori, se pot activa anumite echipamente auxiliare.

Prin intermediul modulului Bluetooth am creat o aplicație de telefon prin care putem schimba afișarea ecranului LCD al casei și putem controla lumina ambientală.

2 Introducere

O casă inteligentă reprezintă automatizarea locuinței noastre și ne permite să controlăm de la distanță aspecte ce țin de confortul ambiental și nu numai. Unele echipamente pot lua decizii în funcție de anumiți parametri, pentru a le crea siguranța si ușurarea vieții proprietarilor.

Senzorii utilizați sunt: senzor de temperatură, senzor de mișcare, senzor de lumină, senzor de gaz și fum, senzor de monoxid de carbon.

O casă inteligentă poate să: detecteze dacă într-o încapere există pericol de incendiu, monoxid de carbon, aprindă lumina dintr-o încapere din aplicația de pe telefon prin intermediul modulului Bluetooth, detecteze mișcarea unei persoane ce va face ca lumina să se aprindă automat.

3 Arhitectura hardware

3.1 Arduino UNO

în realizarea proiectului am ales folosirea unei plăcuțe de dezvoltare de tip Arduino UNO cu următoarele specificații:

- Microcontroller – Atmega328P
- Tensiune de lucru – 5V
- Tensiune de intrare – 7-20V
- Numărul pinilor digitali de I/O – 14 (6 asigură ieșire PWM)
- UART – 1
- I2C – 1
- SPPI – 1
- Numărul pinilor analogici – 6
- Memorie Flash – 32KB
- SRAM – 2KB
- EEPROM – 1KB
- Frecvență – 16 MHz
- Lungime – 68.6 mm
- Lățime – 53.4 mm
- Greutate – 25 g

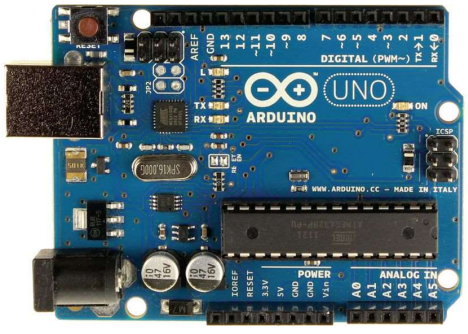


Figure 1. Placa de dezvoltare Arduino UNO

3.1.1 Tensiune de alimentare

Arduino UNO poate fi alimentat direct de la USB sau prin intermediul unui alimentator extern de 9V – 1A.

Plăcuța poate fi alimentată extern la o tensiune cuprinsă între 6 și 20V. Dacă alimentarea este mai mică decât 7V, pinul de 5V va livra o tensiune mai mică decât 5V și placa poate deveni instabilă. Dacă folosim o tensiune de alimentare a plăcii mai mare decât 12V, putem deteriora plăcuța. Recomandat este folosirea unei tensiuni cuprinse între 7-12V.

Pini de alimentare folosiți:

- VIN – Acest pin este folosit pentru înlocuirea pinului de 5V când plăcuța este alimentată extern.
- 5V – Acest pin dă o tensiune de 5V de pe plăcuță. Plăcuța poate fi alimentată de la un alimentator extern (7-12V), de la USB (5V), ori de la pinul VIN (7-12V). Adăugând mai mult voltaj pinilor de 5V sau 3.3V vor ignora regulatorul ceea ce poate duce la avarierea plăcii.
- 3.3V – Acest pin livrează 3.3V generată de regulator. Curentul maxim este de 50mA.
- GND – Pini de împământare.

3.1.2 Memorie

Microcontroller-ul ATmega238P are o memorie 32KB (cu 0.5KB folosiți pentru BIOS). Are deasemenea 2KB SRAM și un 1KB de EEPROM.

3.1.3 Input și Output

Cei 14 pini digitali pot fi folosiți ca pini de intrare sau ieșire prin utilizarea funcțiilor pinMode (), digitalRead () și digitalWrite () în programarea arduino. Fiecare pin acționează la 5V și poate furniza sau primi un curent maxim de 40mA. Din acești 14 pini, unii au funcții specifice, așa cum sunt cei enumerați mai jos:

- Pinii seriali 0 (Rx) și 1 (Tx): pinii Rx și Tx sunt folosiți pentru a primi și transmite date seriale TTL. Aceștia sunt conectați cu USB-ul.
- Pinii de întrerupere externi 2 și 3: Acești pini pot fi configurați pentru a declanșa o întrerupere atunci când au o valoare low, un front crescător sau descrescător sau o modificare a valorii.
- Pinii 3, 5, 6, 9 și 11 PWM: Acești pin furnizează o ieșire PWM pe 8 biți folosind funcția analogWrite ().
- Pinii SPI 10 (SS), 11 (MOSI), 12 (MISO) și 13 (SCK): Acești pini sunt folosiți pentru comunicarea SPI.
- Pinul LED încorporat 13: Acest pin este conectat cu un LED încorporat, când pinul 13 este HIGH - LED-ul este pornit și când pinul 13 este LOW, este oprit.

Pe lângă cei 14 pini digitali, există și 6 pini de intrare analogici, fiecare având o rezoluție de 10 biți, adică 1024 de valori diferite. Aceștia măsoară o tensiune de la 0 la 5 volți, dar această limită poate fi mărită folosind pinul AREF cu funcția de analogReference().

Pinul analogic 4 (SDA) și pinul 5 (SCA), de asemenea pot fi utilizați pentru comunicarea TWI folosind biblioteca Wire.

3.2 Senzorul de temperatură

Unii din senzorii folosiți în acest proiect este senzorul de temperatură și umiditate DHT11 cu un semnal digital de ieșire calibrat. Coeficienții de calibrare sunt stocați în memoria OTP a senzorului, ce sunt folosiți de procesul de detectare a semnalului intern al senzorului.

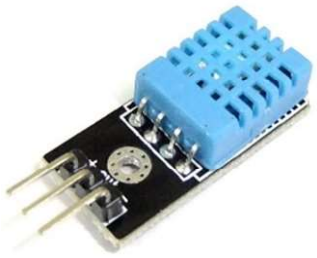


Figure 2. Senzor Temperatura DHT11

3.2.1 Detalii tehnice

Item	Measurement Range	Humidity Accuracy	Temperature Accuracy	Resolution	Package
DHT11	20-90%RH 0-50 ℃	±5%RH	±2℃	1	4 Pin Single Row

Parameters	Conditions	Minimum	Typical	Maximum
Humidity				
Resolution		1%RH	1%RH	1%RH
			8 Bit	
Repeatability			±1%RH	
Accuracy	25℃		±4%RH	
	0-50℃			±5%RH
Interchangeability	Fully Interchangeable			
Measurement Range	0℃	30%RH		90%RH
	25℃	20%RH		90%RH
	50℃	20%RH		80%RH
Response Time (Seconds)	1/e(63%)25℃, 1m/s Air	6 S	10 S	15 S
Hysteresis			±1%RH	
Long-Term Stability	Typical		±1%RH/year	
Temperature				
Resolution		1℃	1℃	1℃
		8 Bit	8 Bit	8 Bit
Repeatability			±1℃	
Accuracy		±1℃		±2℃
Measurement Range		0℃		50℃
Response Time (Seconds)	1/e(63%)	6 S		30 S

3.2.2 Aplicație practică

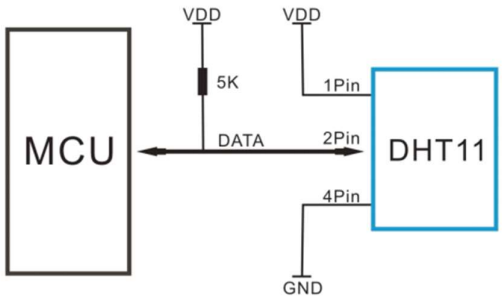


Figure 3. Schemă electrică DHT11

Dacă vom conecta un fir mai mic de 20m, vom avea nevoie de o rezistență de 5K.

Dacă folosim un fir mai mare de 20m, se va alege o valoare a rezistenței necesară.

3.2.3 Alimentare

DHT11 se alimentează de la plăcuța Arduino UNO prin intermediul pinului 3.3V sau 5V DC. Când alimentăm senzorul, nu trebuie trimisă nicio instrucțiune timp de o secundă pentru a trece de starea instabilă.

3.3 Senzorul de mișcare

Cel de-al doilea senzor folosit în proiect este senzorul de mișcare PIR. Senzorul permite detectarea mișcării folosit pentru a vedea dacă o persoană trece prin raza acestuia.

Senzorul PIR conține un senzor piroelectric ce detectează nivele de radiații infraroșu. Acest senzor piroelectric emite radiații de intensitate mică și în momentul în care o persoană trece în raza lui, el va emite radiații de intensitate mare în urma căldurii degajate de persoană.

Figure 4. Senzor de mișcare PIR

3.3.1 Detalii tehnice

Recommended Model	D204B
Encapsulation Type	TO-5
IRReceiving Electrode	2×1mm, 2 elements
Window Size	5×3.8mm
Spectral Response	5-14μm
Transmittance	≥75%
Signal Output[Vp-p]	≥3500mV
Sensitivity	≥3300V/W
Detectivity (D*)	$\geq 1.4 \times 10^{8}$ cmHz$^{1/2}$/W
Noise[Vp-p]	<70mV
Output Balance	<10%
Offset Voltage	0.3-1.2V
Supply Voltage	3-15V
Operating Temp.	-30-70°C
Storage Temp.	-40-80°C

3.3.2 Aplicație practică

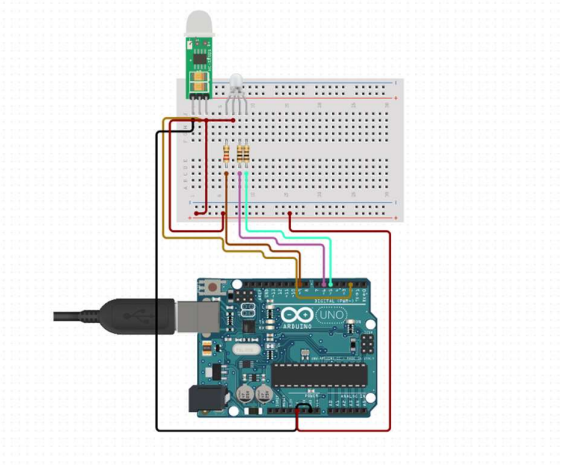


Figure 5. Exemplu practic Senzor de mișcare PIR

În fig 5. este reprezentat un circuit simplu în care vom aprinde un led RGB prin intermediul senzorului de mișcare PIR și al plăcuței Arduino.

3.4 Senzorul de lumină

Senzorul TSL2561 este un convertor de lumină către digital ce transformă intensitatea luminii într-un semnal digital capabil de a fi afișat pe un I2C sau o interfață SMBus.

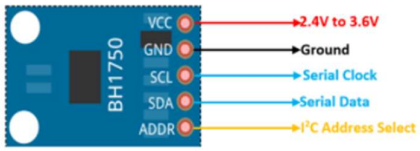


Figure 6. Senzor de lumină TSL2561

3.4.1 Detalii tehnice

Recommended Operating Conditions

	MIN	NOM	MAX	UNIT
Supply voltage, V_{DD}	2.7	3	3.6	V
Operating free-air temperature, T_A	–30		70	°C
SCL, SDA input low voltage, V_{IL}	–0.5		0.8	V
SCL, SDA input high voltage, V_{IH}	2.1		3.6	V

Electrical Characteristics over recommended operating free-air temperature range (unless otherwise noted)

PARAMETER		TEST CONDITIONS	MIN	TYP	MAX	UNIT
I_{DD}	Supply current	Active		0.24	0.6	mA
		Power down		3.2	15	µA
V_{OL}	INT, SDA output low voltage	3 mA sink current	0		0.4	V
		6 mA sink current	0		0.6	V
I_{LEAK}	Leakage current		–5		5	µA

3.4.2 Aplicație practică

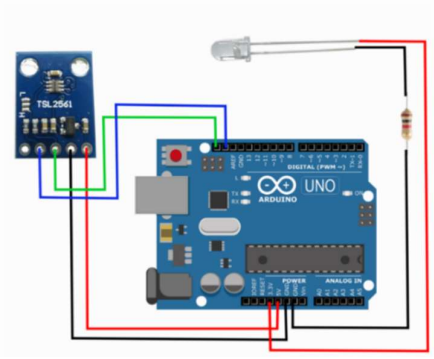


Figure 7. Senzor de lumină TSL2561

3.5 Modulul Bluetooth HC-05

HC-05 este un modul ce se conectează la un port serial (Rx -> Tx, Tx -> Rx) de la Atmega328P, ce permite microcontroller-ului să comunice cu alte device-uri prin intermediul Bluetooth-ului.

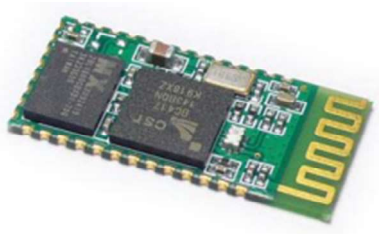


Figure 8. Modul Bluetooth HC-05

3.5.1 Detalii tehnice

- Length: 28 mm (1 in)
- Width: 15 mm (⅔ in)
- Height: 2.35 mm (0.1 in)
- Typical price: Around 8$
- Supply voltage: 3.3V to 6.0V
- Operating voltages: 3.3V (all other pins, except VCC)
- Working current: 30mA
- Operating range: max. 10m (33 ft)
- Default password: 0000 or 1234 (depends on model/manufacturer)
- Supported baud rate: 9600,19200,38400,57600,115200,230400,460800
- Follows IEEE 802.15.1

Modulul poate rula în ambele moduri, atât ca master cât și slave, și poate fi folosit în diferite aplicații cum ar fi: casă înteligentă, robotică, control la distanță, etc.

3.5.2 Aplicație practică

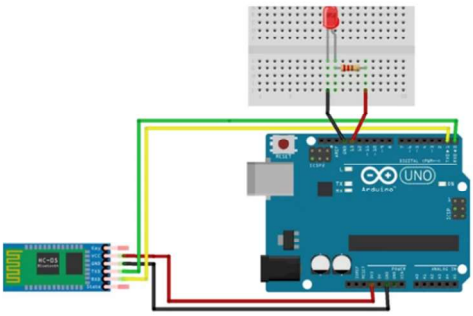


Figure 9. Modul Bluetooth HC-05

3.6 Led RGB

Ledul RGB este o componentă electronică des folosită pentru verificarea funcționalității unui modul/circuit sau iluminarea ambientală.

În cazul ledurilor RGB acestea pot fi de două feluri:

- cu anod comun – se conectează la Vcc;
- sau catod comun – se conectează la GND;

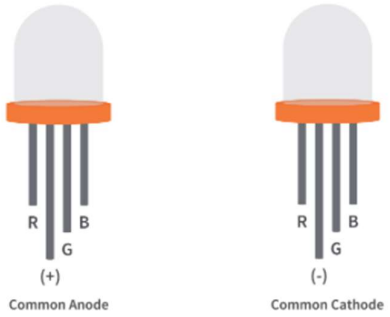


Figure 10. Led RGB

3.6.1 Detalii tehnice

- Low Thermal Resistance
- No UV rays
- Super High flux Output and High luminance
- Forward Current for Red, Blue and Green color: 20mA
- Forward Voltage
  - Red: 2v (typical)
  - Blue: 3.2(typical)
  - Green: 3.2(typical)
- Luminous Intensity
  - Red: 800 mcd
  - Blue: 4000 mcd
  - Green: 900 mcd
- Wavelength
  - Red: 625 nm
  - Blue: 520 nm
  - Green: 467.5 nm
- Operating Temperature: -25 °C to 85 °C
- Storage Temperature: -30 °C to 85 °C

3.6.2 Aplicație practică

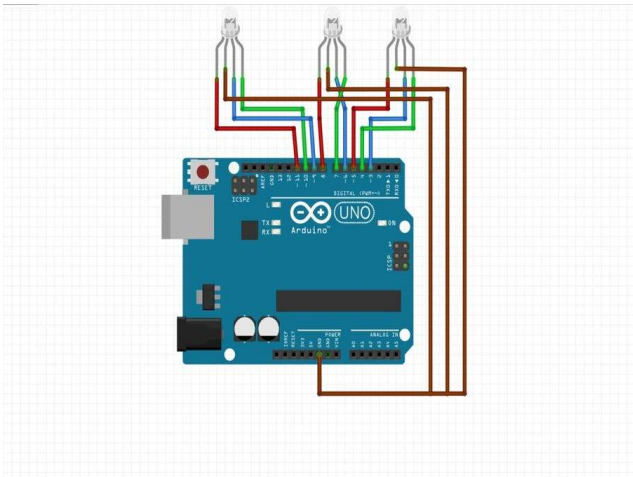


Figure 11. Led RGB conectat la Arduino Uno

3.7 Senzor Gaz MQ2

MQ2 este un senzor ce detectează gazul și fumul dintr-un spațiu închis.

Acest modul vine cu un pin digital ce poate opera fără un microcontroller și este ușor de folosit în cazul în care vrem doar să detectăm.

Când vine vorba de măsurarea gazului intră în acțiune pinul analogic alimentat la 5V și poate fi folosit cu microcontroller.

Figure 12. MQ2 Sensor

3.7.1 Aplicație practică

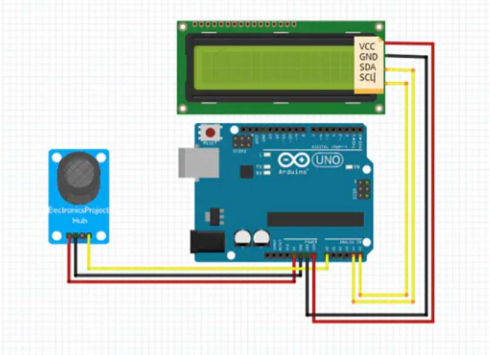


Figure 13. MQ2 Application

3.8 Senzor CO MQ7

Senzorul MQ7 este utilizat în detectarea monoxidului de carbon din interiorul locuințelor.

Figure 14. Senzor MQ7

3.8.1 Detalii tehnice

Symbol	Parameter name	Technical condition	Remark
Vc	circuit voltage	5V±0.1	Ac or Dc
VH (H)	Heating voltage (high)	5V±0.1	Ac or Dc
VH (L)	Heating voltage (low)	1.4V±0.1	Ac or Dc
RL	Load resistance	Can adjust	
RH	Heating resistance	33 Ω ±5%	Room temperature
TH (H)	Heating time (high)	60±1 seconds	
TH (L)	Heating time (low)	90±1 seconds	
PH	Heating consumption	About 350mW	

3.8.2 Aplicație practică

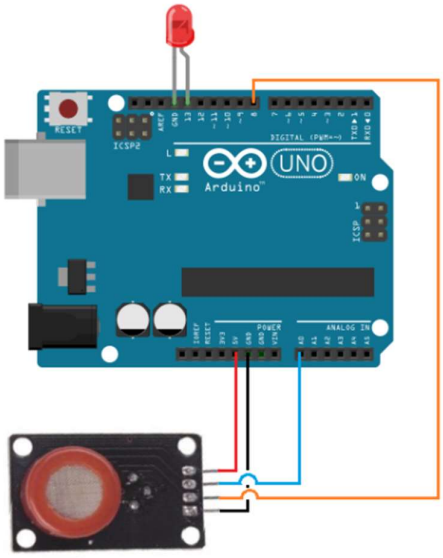


Figure 15. MQ7 Application

3.9 Modul LCD I2C

Modulul LCD 1602 I2C conține 2 linii a câte 16 caractere fiecare afișate pe interfața display-ului. Interfața I2C necesită doar 2 conexiuni pentru operare, +5Vdc și GND.

Figure 16. Module LCD 1602 I2Cd

3.9.1 Detalii tehnice

Pin #	Name	Type	Description
1	GND	Power	Supply & Logic ground
2	VCC	Power	Digital I/O 0 or RX (serial receive)
3	SDA	I/O	Serial Data line
4	SCL	CLK	Serial Clock line
A0	A0	Jumper	Optional address selection A0 - see below
A1	A1	Jumper	Optional address selection A1 - see below
A2	A2	Jumper	Optional address selection A2 - see below
Backlight		Jumper	Jumpered - enable backlight, Open - disable backlight
Contrast		Pot	Adjust for best viewing

	2 lines by 16 character
I2C Address Range	0x20 to 0x27 (Default=0x27, addressable)
Operating Voltage	5 Vdc
Backlight	White
Contrast	Adjustable by potentiometer on I2c interface
Size	80mm x 36mm x 20 mm
Viewable area	66mm x 16mm

3.9.2 Aplicație practică

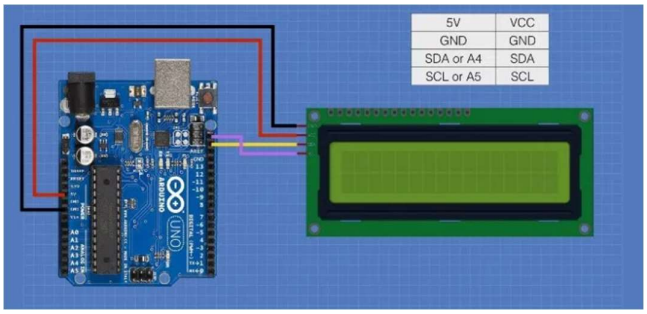


Figure 17. Module LCD 1602 I2C Application

3.10 Power Supply Pentru Breadboard

Am folosit un alimentator extern in acest proiect deoarece placa Arduino nu ar fi putut alimenta toti senzorii necesari realizarii intregului montaj. In figura de mai jos (Figure 18.) Power Supply-ul alimenteaza in partea de sus cu 5V, iar in partea de jos cu 3.3V.

Figure 18. Power Supply pentru Breadboard

4 Bill of Materials

Nr. crt.	Denumire	Preț	Sursa
1	Arduino UNO R3	35 lei	robofun.ro
2	Senzor temperatură DHT11	10 lei	optimusdigital.ro
3	Senzor lumină TSL2561	16 lei	optimusdigital.ro
4	Senzor de mișcare PIR	10 lei	optimusdigital.ro
5	Modul senzor gaz MQ-2	12.5 lei	optimusdigital.ro
6	Modul senzor CO MQ-7	15 lei	optimusdigital.ro
7	Modul afișaj LCD l2C	18 lei	optimusdigital.ro
8	LED-uri RGB	3 lei	optimusdigital.ro
9	Modul Bluetooth	22 lei	optimusdigital.ro
10	Ventilator	5 lei	optimusdigital.ro
11	Placuță Breadboard	10 lei	optimusdigital.ro
12	Breadboard Power Supply	5 lei	optimusdigital.ro
13	Fire	16 lei	optimusdigital.ro
	Total	177.5 lei	

5 Arhitectura Software / Firmware

5.1 Sistem de Operare

Sistemul de operare folosit în acest proiect este Windows 10.

5.2 Limbaj de Programare

Limbajul folosit pentru programarea senzorilor este C++.

5.3 Medii De Dezvoltare

Mediul de dezvoltare al acestui proiect este Arduino IDE (versiunea 1.8.13) folosit pentru crearea, încărcarea și testarea programelor aferente pentru plăcuța de dezvoltare Arduino.

5.4 Compilator

Compilatorul este AVR-GCC (versiunea 1.8.3) ce traduce codul sursă în codul mașinii ca să poată fi încărcat în microcontroller.

5.5 Librării

- Wire.h :

  Librăria permite comunicarea cu device-uri I2C/TWI. La plăcile Arduino cu revizia 3, pinii SDA (data line) și SCL (clock line) se găsesc lângă pinul AREF situat în partea de sus a plăcii.

- LiquidCrystal_I2C.h :

  Librăria permite ca microcontroller-ul să acceseze și să controleze LCD bazat pe un chipset Hitachi HD44780, care este cel mai întâlnit la LCD ce afișează text.

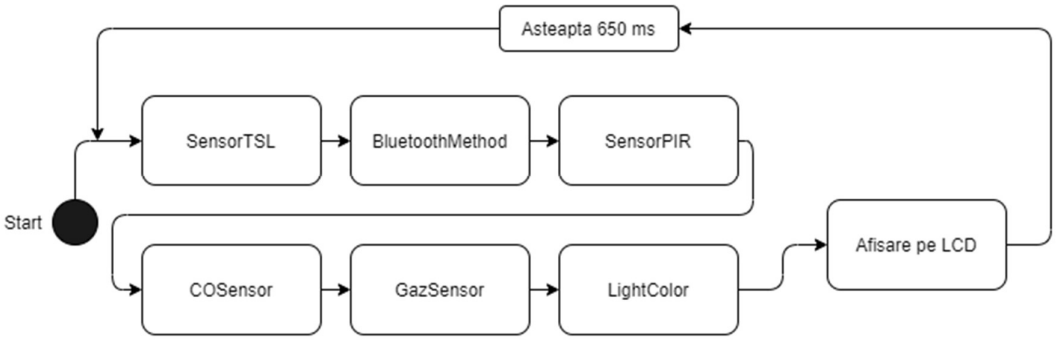
- DHT.h :

  Librăria asigură preluarea datelor de la senzorul de temperatură și transmiterea acestora către microcontroller.

- SparkFunTSL2561.h :

  Librăria permite preluarea valorilor de la senzorul de lumină TSL2561 și prelucrarea acestora de către microcontroller.

5.6 Schemă Logică



Pseudocod:

```
citeste LED_B_BL
citeste LED_G_BL
citeste LED_BL
citeste LED_PIR
citeste INP_PIR
citeste LED_TSL
citeste LED_B_TSL
citeste LED_G_TSL
citeste DHT_PIN
citeste COOLER_PIN
citeste MQ7_analog_IN
citeste MQ2_analog_IN
citeste humidity
citeste tempC
citeste tempF

Functie COSensor()

        daca Value >= 100 atunci
                porneste cooler
        altfel
                cooler oprit
        sfarsit daca

SfFunctie

Functie MonoxideSensorPrint()
        afiseaza pe LCD CO Value
SfFunctie

Functie GasSensorPrint()
        afiseaza pe LCD Gas Value
SfFunctie

Functie GasSensor()

        daca Value >= 170 atunci
                porneste cooler
        altfel
                cooler oprit
        sfarsit daca

SfFunctie

Functie DHTSensor()
        afisare pe LCD tempC
SfFunctie

Functie TSLSensor()

citeste visible

        daca detectez lumina atunci
                daca visible > 20000 atunci
                        stinge LED_RGB
                altfel
                        aprinde LED_RGB
                sfarsit daca
        altfel
                afiseaza erroare
        sfarsit daca

SfFunctie

Functie PIRSensor()

        daca detectez miscare atunci
                aprinde LED_RGB
        altfel
                LED_RGB stins
        sfarsit daca
```

SfFunctie

Functie light_color()

daca state = 1 atunci
aprinde LED_RGB
altfel
LED_RGB stins
sfarsit daca
SfFunctie

Functie Bluetooth()

citeste incoming_value

daca incoming_value = 1 atunci
aprinde LED_RGB
altfel
LED_RGB stins
sfarsit daca

daca incoming_value = a atunci
afiseaza pe LCD DHTSensor
altfel daca incoming_value = b atunci
afiseaza pe LCD GasSensorPrint
altfel daca incoming_value = c atunci
afiseaza pe LCD MonoxideSensorPrint
altfel daca incoming_value = 1 atuci
afiseaza pe LCD DHTSensor
sfarsit daca
SfFunctie

5.1 Cod Arduino

```
void BluetoothMethod()
{
      if(Serial.available() > 0)
      {
            char incoming_value = Serial.read();
            Serial.print("State2: ");
            Serial.println(state);
            if (incoming_value == '1')
                {
                  state = 1;
                }
            else if (incoming_value == '0')
                {
                  state = 0;
                }
            Serial.print("State3: ");
            Serial.println(state);
            Serial.print(incoming_value);
            Serial.print("\n");

            switch(incoming_value)
            {
              case 'a':
                DHTSensor();
                break;
              case 'b':
                GasSensorPrint();
                break;
              case 'c':
                MonoxideSensorPrint();
                break;
              default:
                DHTSensor();
                break;
            }
      }
}

void loop() {
  SensorTSL();
  BluetoothMethod();
  SensorPIR();
  CoSensor();
  GazSensor();
  light_color();

  Serial.print("\n");
  //delay(1500);
}
```

```
void SensorTSL()
{
  //  delay(ms);
    unsigned int visible, infrared;

    if (light.getData(visible,infrared))
    {
      Serial.print("Visible: ");
      Serial.print(visible);
      Serial.print("\n");

      if (visible > 20000)
      {
      analogWrite(LED_TSL, 255);
      analogWrite(LED_B_TSL, 255);
      analogWrite(LED_G_TSL, 255);
      }
      else
      {
      analogWrite(LED_TSL, visible/3);
      delay(50);
      analogWrite(LED_B_TSL, visible/3);
      delay(50);
      analogWrite(LED_G_TSL, visible/3);
      delay(50);
      }

    }
    else
    {
      byte error = light.getError();
      Serial.println(error);
    }
}

void SensorPIR(){
   if(digitalRead(INP_PIR) == HIGH)
   {
    Serial.println("Motion Detected");
      digitalWrite(LED_PIR,LOW);
     // delay(2000);
   }
  else{
    Serial.println("Motion not detected");
      digitalWrite(LED_PIR,HIGH);
     // delay(2000);
  }
}
```

5.2 Statistică

Denumirea senzorului	Timp necesar	Dezvoltator
Senzor DHT11	6h	Zaharia George-Andrei
Senzor PIR	5h	Zaharia George-Andrei
Senzor MQ2	1.3h	Zaharia George-Andrei
Senzor MQ7	1h	Sarca Florin-Sabin
Senzor TSL2561	3h	Sarca Florin-Sabin
Modul Bluetooth (+ aplicatie)	10h	Sarca Florin-Sabin
LCD I2C	6h	Zaharia George-Andrei

5.3 Structurarea Lucrării

5.3.1 Schemă Bloc

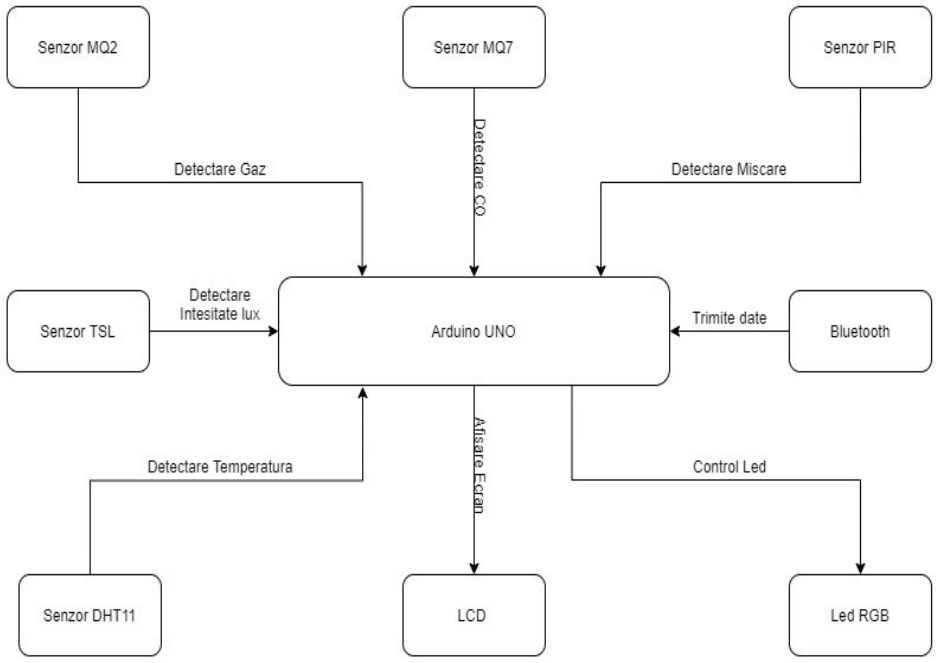


Figure 19. Schemă bloc a casei inteligente

5.3.2 Schemă Electrică

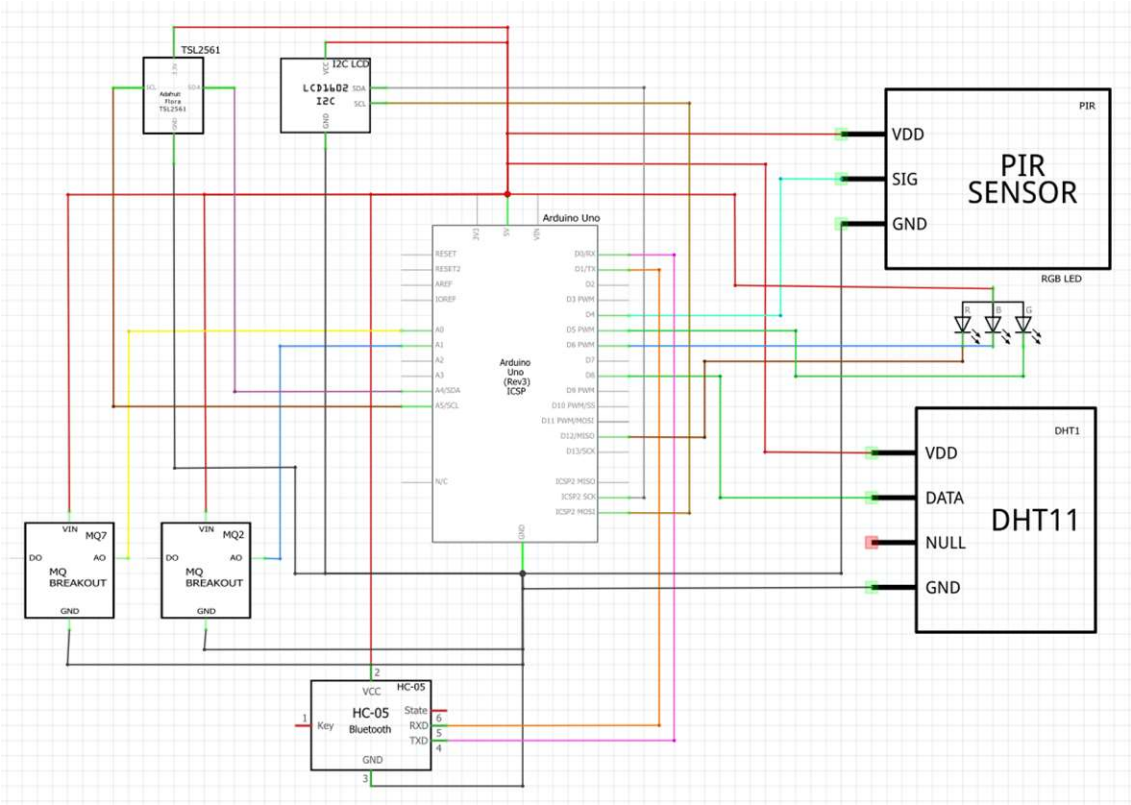


Figure 20. Schemă electrică a casei inteligente

5.3.3 Schemă de Conectare

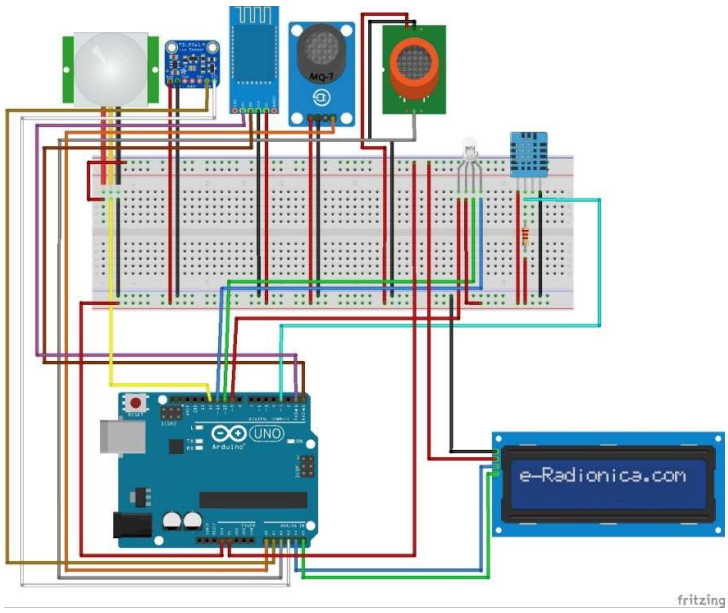


Figure 21. Schemă de conectare a casei inteligente

6 Montaj

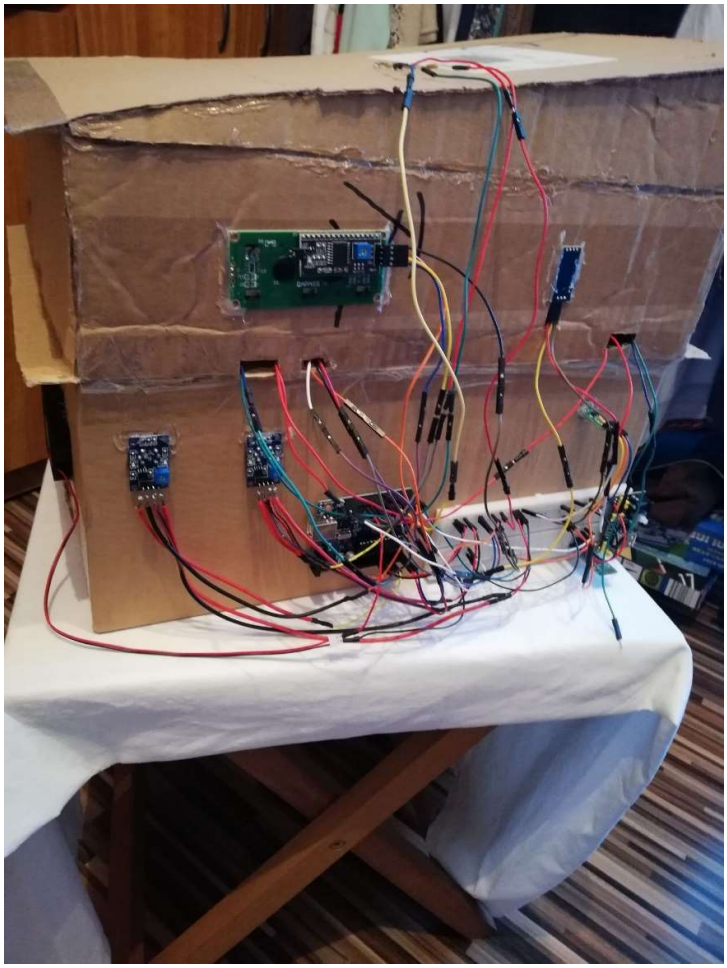


Figure 22. Montaj

7 Concluzii

În urma acestui proiect am acumulat informații noi despre cum se programează, cum se testează și cum măsoară senzorii prezentați mai sus. Tehnologia cere ne-a ajutat la îndeplinirea proiectului este Arduino IDE.

Problemele întâmpinate au fost legate de adaptarea codului pentru fiecare senzor în parte, la realizarea unor conexiuni între mai mulți senzori, cât și la realizarea aplicației mobile folosită pentru testarea modulelor.

În cuida tuturor dificultăților care au apărut în realizarea Casei Inteligente, aceasta a fost și este un success.

8 Bibliografie

- https://ardushop.ro/1329-thickbox_default/dht11-digital-temperature-and-humidity-sensor-module.jpg
- https://www.mouser.com/datasheet/2/758/DHT11-Technical-Data-Sheet-Translated-Version-1143054.pdf
- https://www.farnell.com/datasheets/1682209.pdf
- https://cdn-learn.adafruit.com/downloads/pdf/pir-passive-infrared-proximity-motion-sensor.pdf
- https://www.homemade-circuits.com/pir-sensor-datasheet-pinout-specification-working/
- https://www.distrelec.ro/Web/WebShopImages/landscape_large/9-/01/arduino-a000066.jpg
- https://cdn-shop.adafruit.com/datasheets/TSL2561.pdf
- https://cdn-learn.adafruit.com/downloads/pdf/tsl2561.pdf
- https://maker.pro/custom/tutorial/hc-05-bluetooth-transceiver-module-datasheet-highlights
- https://clubarcrobotica.wordpress.com/arduino-remote-controlled-led-using-hc-05-bluetooth/
- https://makersportal.com/blog/2018/4/19/arduino-light-sensor-tsl2561-and-experiments-with-infrared-and-visible-light
- https://www.circuitbread.com/tutorials/how-rgb-leds-work-and-how-to-control-color
- https://components101.com/rgb-led-pinout-configuration-circuit-datasheet
- https://create.arduino.cc/projecthub/AlexanderVaughn/not-your-typical-rgb-led-356854
- https://components101.com/mq2-gas-sensor
- https://create.arduino.cc/projecthub/Junezriyaz/how-to-connect-mq2-gas-sensor-to-arduino-f6a456
- http://www.learningaboutelectronics.com/Articles/MQ-7-carbon-monoxide-sensor-circuit-with-arduino.php
- https://www.sparkfun.com/datasheets/Sensors/Biometric/MQ-7.pdf
- https://opencircuit.shop/resources/file/da88acc1702a90667728fcf4ac9c75c455475706466/I2C-LCD-interface.pdf
- https://dronebotworkshop.com/lcd-displays-arduino/
- https://components101.com/sensors/bh1750-ambient-light-sensor
- https://www.arduino.cc/en/reference/wire

9 Cod Sursă

```
#include <SparkFunTSL2561.h>
#include <Wire.h>
#include <LiquidCrystal_I2C.h>
#include <DHT.h>
#define Type DHT11

LiquidCrystal_I2C lcd(0x27, 20, 4);

SFE_TSL2561 light;

boolean gain;
unsigned int ms = 1500;

volatile int state = 0;
int LED_B_BL = 6;
int LED_G_BL = 5;
int LED_BL = 12;
int LED_PIR = 11;
int INP_PIR = 4;
int LED_TSL = 3;
int LED_B_TSL = 10;
int LED_G_TSL = 9;
int DHT_PIN = 8;
int COOLER_PIN = 13;
int MQ7_analog_IN = A0;
int MQ2_analog_IN = A1;
float humidity;
float tempC;
float tempF;

DHT dht(DHT_PIN,Type);

void setup() {
  lcd.init();
  lcd.backlight();
  lcd.clear();
  lcd.setCursor(0,0);
  lcd.print("Which sensor do");
  lcd.setCursor(0,1);
  lcd.print("you want to use?");
  Serial.begin(9600);
  pinMode(LED_BL,OUTPUT);
  pinMode(LED_PIR, OUTPUT);
  pinMode(INP_PIR, INPUT);
  pinMode(COOLER_PIN, OUTPUT);
  pinMode(MQ7_analog_IN, INPUT);
  pinMode(MQ2_analog_IN, INPUT);
  digitalWrite(LED_BL, HIGH);
  digitalWrite(LED_PIR, HIGH);
  analogWrite(LED_TSL, 255);

  dht.begin();
  delay(500);
  light.begin();

  gain = 1;
  unsigned char time = 2;
  light.setTiming(gain,time,ms);
  light.setPowerUp();
}
```

```
void MonoxideSensorPrint()
{
 int Value = analogRead(MQ7_analog_IN);
 lcd.clear();
 lcd.setCursor(0,0);
 lcd.print("CO value:   ");
 lcd.print(Value);
}

void GasSensorPrint()
{
 int Value = analogRead(MQ2_analog_IN);
 lcd.clear();
 lcd.setCursor(0,0);
 lcd.print("Gas value:  ");
 lcd.print(Value);
}

void CoSensor()
{
 int Value = analogRead(MQ7_analog_IN);
 Serial.print("Monoxid value: ");
 Serial.println(Value);

 if(Value >= 100)
 {
  digitalWrite(COOLER_PIN, HIGH);
 }
 else
 {
  digitalWrite(COOLER_PIN, LOW);
 }
}
// delay(1500);

void GazSensor()
{
 int Value = analogRead(MQ2_analog_IN);
 Serial.print("Gsz value: ");
 Serial.println(Value);

 if(Value >= 170)
 {
  digitalWrite(COOLER_PIN, HIGH);
 }
 else
 {
  digitalWrite(COOLER_PIN, LOW);
 }
}

void DHTSensor()
{
  humidity = dht.readHumidity();
  tempC = dht.readTemperature();
  tempF = dht.readTemperature(true);

  lcd.clear();
  lcd.setCursor(0,0);
  lcd.print("TempC:");
  lcd.print(tempC);
  lcd.print(" C ");
  lcd.setCursor(0,1);
```

```
  lcd.print("Humidity:");
  lcd.print(humidity);
  lcd.print(" %");

}

void SensorTSL()
{
 //  delay(ms);
  unsigned int visible, infrared;

  if (light.getData(visible,infrared))
  {
   Serial.print("Visible: ");
   Serial.print(visible);
   Serial.print("\n");

   if (visible > 20000)
   {
   analogWrite(LED_TSL, 255);
   analogWrite(LED_B_TSL, 255);
   analogWrite(LED_G_TSL, 255);
   }
   else
   {
   analogWrite(LED_TSL, visible/3);
   delay(50);
   analogWrite(LED_B_TSL, visible/3);
   delay(50);
   analogWrite(LED_G_TSL, visible/3);
   delay(50);
   }
  }
  else
  {
   byte error = light.getError();
   Serial.println(error);
  }
}

void SensorPIR(){
  if(digitalRead(INP_PIR) == HIGH)
  {
   Serial.println("Motion Detected");
    digitalWrite(LED_PIR,LOW);
  }
 else{
   Serial.println("Motion not detected");
    digitalWrite(LED_PIR,HIGH);
 }
}

void light_color()
{
 Serial.print("State1: ");
 Serial.println(state);
 if(state == true)
       {
          digitalWrite(LED_BL, LOW);
          delay(50);
          digitalWrite(LED_BL, HIGH);
          delay(50);
          analogWrite(LED_B_BL, 0);
          delay(50);
          analogWrite(LED_B_BL, 255);
```

```
    delay(50);
    analogWrite(LED_G_BL, 0);
    delay(50);
    analogWrite(LED_G_BL, 255);
    delay(50);
    digitalWrite(LED_BL, LOW);
    analogWrite(LED_B_BL, 0);
    delay(50);
    digitalWrite(LED_BL, HIGH);
    analogWrite(LED_B_BL, 255);
    delay(50);
    digitalWrite(LED_BL, LOW);
    analogWrite(LED_G_BL, 0);
    delay(50);
    digitalWrite(LED_BL, HIGH);
    analogWrite(LED_G_BL, 255);
    delay(50);
    digitalWrite(LED_BL, LOW);
    analogWrite(LED_G_BL, 0);
    analogWrite(LED_B_BL, 0);
    delay(50);
    digitalWrite(LED_BL, HIGH);
    analogWrite(LED_G_BL, 255);
    analogWrite(LED_B_BL, 255);
    delay(50);
  }
  else
  {
    digitalWrite(LED_BL, HIGH);
    analogWrite(LED_B_BL, 255);
    analogWrite(LED_G_BL, 255);
  }
}

void BluetoothMethod()
{
  if(Serial.available() > 0)
  {
    char incoming_value = Serial.read();
    Serial.print("State2: ");
    Serial.println(state);
    if (incoming_value == '1')
      {
       state = 1;
      }
    else if (incoming_value == '0')
      {
       state = 0;
      }
    Serial.print("State3: ");
    Serial.println(state);
    Serial.print(incoming_value);
    Serial.print("\n");

    switch(incoming_value)
    {
     case 'a':
      DHTSensor();
      break;
     case 'b':
      GasSensorPrint();
      break;
     case 'c':
      MonoxideSensorPrint();
      break;
     default:
```

```
            DHTSensor();
            break;
          }
      }
}

void loop() {
  SensorTSL();
  BluetoothMethod();
  SensorPIR();
  CoSensor();
  GazSensor();
  light_color();

  Serial.print("\n");

}
```