**Robot agricol**

Nume: Anastasiu Andreea Valentina

Grupa:30237

**Cuprins**

1.Introducere

2.Prezentare functionalitate

2.1. Prezentare cod senzori

2.2.Prezentare cod motoare

3. Schema electrica si poze fizice

**1.Introducere**

Proiectul propune utilizarea unui sistem robotic avansat pentru a sprijini fermierii în luarea deciziilor informate, evitând zonele sensibile din punct de vedere climatic și condițiile meteorologice instabile.

Robotul este dotat cu o serie de senzori care măsoară umiditatea solului, temperatura, umiditatea aerului și concentrația de dioxid de carbon. Aceste date sunt transmise către un Raspberry Pi, care analizează parametrii în timp real. Dacă valorile colectate se încadrează în limitele optime, sistemul declanșează o funcție suplimentară: realizarea de fotografii ale zonei monitorizate. Acest mecanism nu doar că optimizează utilizarea resurselor, ci contribuie la monitorizarea atentă și la dezvoltarea unui mediu agricol mai inteligent și mai eficient.

**2.Prezentare functionalitate**

**2.1. Prezentare cod senzori**

Codul implementează controlul unui sistem robotic care monitorizează condițiile de mediu utilizând senzori pentru CO2, temperatură, umiditate a aerului și umiditate a solului. Sistemul transmite datele către un Raspberry Pi 4 și cu ajutorul unui mechanism mecanic cu servo motor, senzorul de umiditate a solului este pus in pamant.

**a. Configurări și inițializări**

* **Senzor de CO2:**  
  Configurat pe pinul analogic A1. Calibrarea este realizată în funcția setup(): CO2Sensor co2Sensor(CO2\_PIN, 0.99, 100);
* **Senzor DHT11:**  
  Configurat pentru măsurarea temperaturii și umidității aerului pe pinul digital 4.
* **Senzor de umiditate a solului:**  
  Configurat pe pinul digital 7 cu un prag (thresholdValue) pentru a determina dacă solul este uscat sau umed.
* **Motor servo:**  
  Controlat pe pinul digital 9, utilizat pentru a insera senzorul de umiditate sol.
* **Timpi de control:**  
  Diferite intervale temporale sunt utilizate pentru gestionarea stărilor (pauză, citire și revenire).

**b. Stări ale sistemului**

Codul utilizează o mașină de stări finite pentru a controla sistemul:

* **PAUSE:**  
  Așteaptă o perioadă de pauză (pauseDuration) înainte de a începe citirea datelor, deoarece atunci se va misca robotul.

if (currentMillis - previousMillis >= pauseDuration) {

currentState = READING;}

* **READING:**  
  Citirea datelor senzorilor și mișcarea servo-ului.
* **RETURNING:**  
  Revine la poziția inițială a servo-ului și trece la starea de pauză.

**c. Citirea senzorilor**

Funcția readSensors() colectează date de la senzori:

* **CO2:**  
  Valoarea este citită prin co2Sensor.read() și reprezintă concentrația de CO2.
* **DHT11:**  
  Temperatură și umiditate citite prin dht.readTemperature() și dht.readHumidity().
* **Umiditate sol:**  
  Valoarea analogică a senzorului este comparată cu un prag (thresholdValue) pentru a determina dacă solul este "Dry" (uscat) sau "Wet" (umed).

**d.** **Transmiterea datelor**

Datele colectate sunt formate într-un șir care include valorile pentru CO2, temperatură, umiditate aer, umiditate sol și starea solului. Acestea sunt transmise prin portul serial către Raspberry Pi.

String data = "CO2:" + String(co2Value) +

",Temp:" + String(temperature, 2) +

",Humidity:" + String(humidity, 2) +

",SoilMoisture:" + String(soilValue) +

",SoilStatus:" + soilStatus;

Serial.println(data);

**2.2.Prezentare cod motoare**

Controlează o mașină cu 4 motoare DC, un senzor ultrasonic și un servo motor, care evită obstacolele autonom. Mașina detectează distanțele față de obstacole utilizând un senzor ultrasonic și ia decizii de direcție pe baza datelor colectate.

#### a.Configurări și Inițializări

* **Senzor ultrasonic:**  
  Detectează distanțele până la obstacole folosind pinii TRIG\_PIN (A1) și ECHO\_PIN (A0).

NewPing radar(TRIG\_PIN, ECHO\_PIN, MAX\_DISTANCE);

* + MAX\_DISTANCE: Distanța maximă detectată de senzor (200 cm).
* **Motoare DC:**  
  Controlate cu placa Adafruit Motor Shield. Cele patru motoare sunt inițializate pe canalele 1, 2, 3 și 4.

AF\_DCMotor motor1(1);

AF\_DCMotor motor2(2);

AF\_DCMotor motor3(3);

AF\_DCMotor motor4(4);

* **Servo motor:**  
  Controlează direcția senzorului ultrasonic pentru a măsura distanțele în diferite direcții (stânga, dreapta, față).

Servo myservo;

myservo.attach(9);

**b. Timpi și stări**

Codul gestionează funcționarea mașinii folosind două intervale de timp:

* **Durată de funcționare (runDuration)**: 1 minut și 30 de secunde.
* **Durată de pauză (pauseDuration)**: jumatate de minut cand va prelua date de la senzori

Variabila isRunning controlează dacă mașina este activă sau în pauză.

**c. Funcționalități principale**

**Citirea distanței**  
Funcția checkDistance() măsoară distanța până la obstacole în cm. Dacă senzorul nu detectează nimic (0 cm), valoarea este setată la 250 cm pentru a indica "fără obstacole".

int checkDistance() {

int cm = radar.ping\_cm();

if (cm == 0) {

cm = 250;

}

return cm;

}

**Detectarea laterală**  
Servo motorul mută senzorul ultrasonic spre stânga sau dreapta pentru a măsura distanțele.

* **Stânga:** for (pos = 100; pos <= 170; pos += 1) { myservo.write(pos); }
* **Dreapta:** for (pos = 100; pos >= 30; pos -= 1) { myservo.write(pos); }

**Logica de evitare a obstacolelor**

* Dacă un obstacol este detectat la o distanță mai mică de 15 cm:
  + Oprește mișcarea (stopMovement()).
  + Merge înapoi (moveBackward()).
  + Compară distanțele la stânga și dreapta.
  + Se întoarce în direcția cu mai mult spațiu liber (turnLeft() sau turnRight()).
* Dacă nu există obstacole, continuă înainte (moveForward()).

if (distance <= 15) {

stopMovement();

moveBackward();

if (distanceRight >= distanceLeft) {

turnRight();

} else {

turnLeft();

}

} else {

moveForward();

}

**Mișcarea motoarelor**

* **Înainte:**

motor1.run(FORWARD);

motor2.run(FORWARD);

* **Înapoi:**

motor1.run(BACKWARD);

motor2.run(BACKWARD);

* **Stânga:**

motor1.run(FORWARD);

motor2.run(BACKWARD);

* **Dreapta:**

motor1.run(BACKWARD);

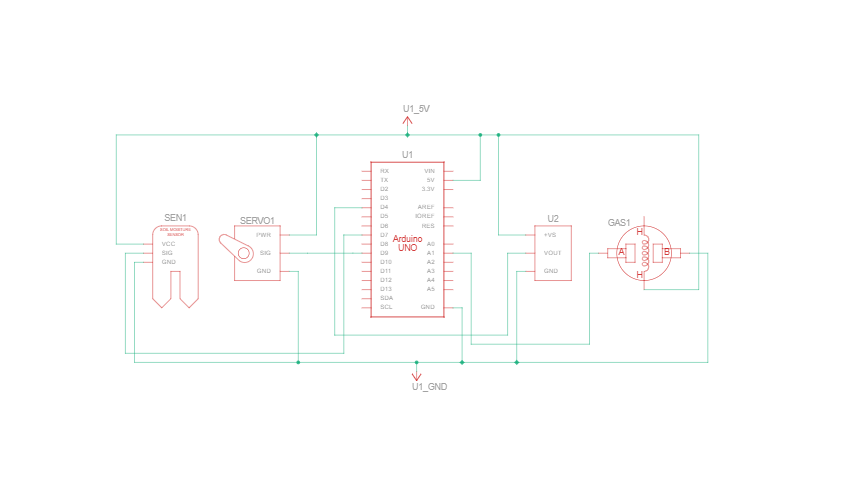
motor2.run(FORWARD);

* **Oprire:**

motor1.run(RELEASE);

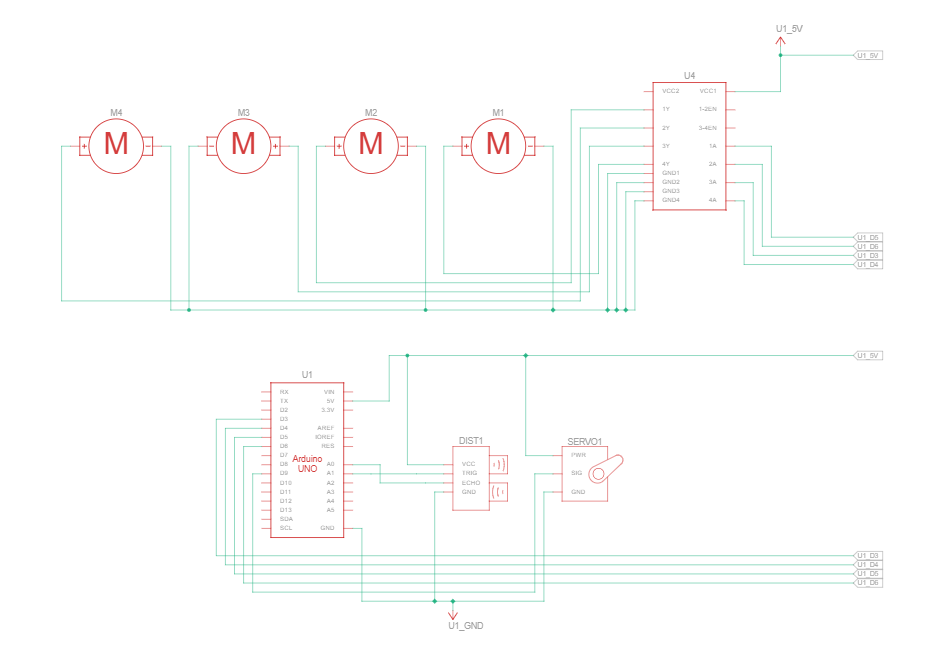
motor2.run(RELEASE);

**3. Schema electrica si poze fizice**



Aceasta prezintă circuitul cu senzori

1. **Arduino UNO (U1)** - Controlerul central al sistemului care primește și procesează semnalele.
2. **Senzor de gaz (GAS1)** - Detectează prezența dioxidului de carbon. Este conectat la pini analogici pentru a oferi o valoare proporțională cu concentrația gazului.
3. **Servo motor (SERVO1)** – E folosit pentru inserarea senzorului de umiditate a solului in pamant.
4. **Senzor umiditate sol** (SEN1)
5. **Senzor umiditate si temperatura(U2)**
6. **Conexiuni de alimentare (U1\_5V și U1\_GND)** - Circuitul este alimentat cu 5V, iar masa comună (GND) este distribuită tuturor componentelor.



**Arduino UNO (U1)** - Controlerul principal al sistemului.

**Motoare (M1-M4)** - Patru motoare DC pentru mișcare, conectate printr-un driver H-Bridge.

**Driver motor (U4)** - Circuitul L293D sau similar care permite controlul direcției și vitezei motoarelor.

**Senzor de distanță (DIST1)** - Senzor ultrasonic care măsoară distanța până la obstacole și este conectat la pini digitali (TRIG și ECHO).

**Servo motor (SERVO1)** – Care e utilizat pentru a roti senzorul de distanță.

**Conexiuni de alimentare (U1\_5V și U1\_GND)**

