UNIVERSITATEA TEHNICĂ DIN CLUJ-NAPOCA

Solar-tracker cu monitorizare a energiei

Proiectare cu Microprocesoare

Autor: Rodanciuc Tiberiu-Gabriel
Grupa: 30237

Facultatea de Automatica
si Calculatoare

15 Ianuarie 2024

Cuprins

1 Descriere

Proiectul nostru de programare cu microprocesoare se concentrează pe utilizarea plăcii Arduino Uno pentru a dezvolta un sistem de sun-tracking și monitorizare a producției de energie solară. Acest proiect combină tehnologia Arduino cu conceptele de energie solară și urmărire a soarelui pentru a optimiza colectarea energiei solare.

Sun-tracking-ul este realizat prin intermediul a 4 fotorezistente și doua motoare servo, care ajustează poziția panoului solar pentru a urmări mișcarea soarelui pe parcursul zilei. Astfel, panoul solar este întotdeauna orientat spre soare, maximizând astfel producția de energie.

În plus, proiectul include și un sistem de monitorizare a producției de energie, care colectează date despre cantitatea de energie solară produsă de panoul solar si o trimite prin internet folosind un modul ESP8266. Aceste date pot fi vizualizate și analizate printr-o interfață de utilizator sau transmise către o platformă online pentru monitorizare în timp real.

Prin acest proiect, ne propunem să demonstrăm eficiența și utilitatea tehnologiei solare și a urmăririi soarelui în producția de energie verde. Acesta reprezintă o soluție inovatoare pentru utilizarea eficientă a energiei solare, contribuind la reducerea amprentei de carbon și la promovarea energiei regenerabile în lumea modernă.

2 Componentele folosite

Creierul proiectului sta in placa de dezvoltare Arduino UNO. Am ales aceasta placa deoarece are toti pinii de care am nevoie pentru realizarea trackerului. Pinii analogici sunt folositi pentru masurarea energiei produse de panoul solar, si citirea fotorezistentelor, pentru ca ulterior sa facem decizii pentru miscarea servomotoarelor, prin pinii digitali. De asemenea, folosim comunicare UART cu modulul ESP8266 pentru transmiterea datelor.

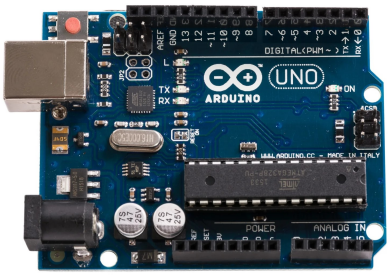

Figura 1: Placa Arduino Uno R3

Pentru a decide directia in care sa se miste panoul solar, folosim 4 fotorezistente. Fotorezistențele, cunoscute și sub denumirea de rezistoare foto sau celule fotoconductoare, sunt dispozitive electronice sensibile la lumină. Acestea au o rezistență electrică care variază în funcție de intensitatea luminii la care sunt expuse. Cand observam, citindu-le valorile folosind pinii analogici ale placii, ca unele sunt mai expuse luminii decat celelalte, decidem directia si unghiul cu care invartim servomotarele care alcatuiesc sistemul pan-tilt.

Figura 2: Fotorezistenta

Servomotoarele 9g sunt dispozitive care transformă semnalele electrice, în special semnalele PWM (modularea lățimii impulsurilor), în mișcare mecanică controlată și precisă. Ele includ un motor electric, un potențiometru sau un encoder pentru feedback și un circuit de control electronic. Semnalul PWM determină poziția dorită, iar servomotorul se rotește până când feedback-ul îl plasează în poziția corectă.

Figura 3: Servomotor 9g

Folosind 2 servomotoare, am creat, dupa printarea 3D a unor parti, un sistem Pan-Tilt, ca in figura 6. Acesta foloseste un servomotor pentru miscare orizontala si altul pentru inclinare verticala, pentru a ne indrepta mereu panoul solar spre soare.

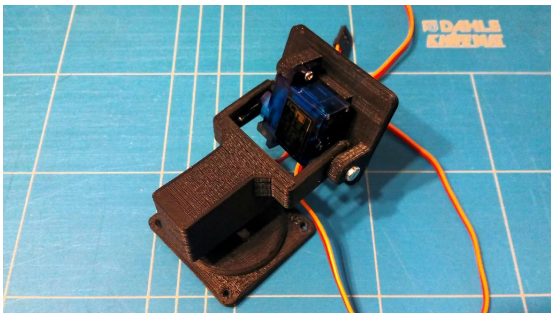

Figura 4: Sistemul Pan-Tilt cu 2 servomotoare

ESP8266-01 este un modul de dezvoltare bazat pe cipul ESP8266, oferind conectivitate WiFi și posibilitatea de a controla dispozitivele și senzorii utilizând comunicații wireless. Acesta are porturi GPIO limitate, spațiu de stocare limitat și este folosit în proiecte IoT și comunicații fără fir. Placuta Arduino UNO comunica cu acest modul prin protocolul de comunicare UART.

Figura 5: Modul IoT ESP8266

3 Functionalitati

Modul de functionare a proiectului, si modul de organizare a codului Arduino poate fi actualizat in 2 sub-sectiuni: logica pentru urmarirea luminii/soarelui, si transmiterea de date prin Internet.

3.1 Sun - Tracking

Codul Arduino controlează două servo-motoare pentru mișcările de pan (orizontal) și tilt (vertical). Aceste servo-motoare sunt de obicei utilizate în aplicații precum urmărirea sau orientarea camerei. Iată o explicație detaliată a codului:

1. **Include și declară variabile**:
   - Se include biblioteca `Servo.h`, care furnizează funcționalitatea pentru controlul servo-motoarelor.
   - Se declară două obiecte `Servo` pentru servo-motoarele de pan (orizontal) și tilt (vertical).
   - Variabilele `panAngle` și `tiltAngle` reprezintă unghiurile curente de pan și tilt ale servo-motoarelor.
   - Variabila `waittime` specifică intervalul de întârziere în milisecunde dintre mișcările servo-motoarelor.
   - Variabilele `topLeft`, `topRight`, `downLeft`, și `downRight` sunt utilizate pentru a stoca valorile citite de la fotorezistoare.
2. **Funcția `setup()`**:
   - Se atașează pinurile la care sunt conectate servo-motoarele de pan și tilt.
   - Se configurează pinurile A0, A1, A2 și A3 ca intrări (prin `pinMode`), pentru a citi valorile de la fotorezistoare.
3. **Bucle Principale (`loop()`)**:
   - Se citesc valorile de la fotorezistoarele aflate în colțurile superioare și inferioare ale dispozitivului.
   - În funcție de comparații între aceste valori, unghiurile de pan și tilt ale servo-motoarelor sunt ajustate pentru a orienta corect camera.
   - Există și limitări pentru a preveni depășirea unghiurilor maxime.

- Se utilizează funcțiile `write()` ale obiectelor servo pentru a controla poziția servomotoarelor.

```
#include <Servo.h>

Servo panServo; // Pan servo
Servo tiltServo; // Tilt servo
int panAngle = 80;
int tiltAngle = 80;
int waittime = 10;
int topLeft, topRight, downLeft, downRight;

void setup()
{
  panServo.attach(6);
  tiltServo.attach(5);
  pinMode(A0, INPUT);
  pinMode(A1, INPUT);
  pinMode(A2, INPUT);
  pinMode(A3, INPUT);
}

void loop()
{
  // Read values from photoresistors
  topLeft = analogRead(A2);
  topRight = analogRead(A1);
  downLeft = analogRead(A0);
  downRight = analogRead(A3);

  if (topLeft > topRight){
   panAngle = panAngle - 1;
   delay(waittime);
  }

  if(downLeft > downRight){
   panAngle = panAngle - 1;
   delay(waittime);
  }

  if(topLeft < topRight){
   panAngle = panAngle + 1;
   delay(waittime);
  }

  if(downLeft < downRight){
   panAngle = panAngle + 1;
   delay(waittime);
  }
```

```
47
48    if(panAngle > 160){
49     panAngle = 160;
50    }
51
52    if(panAngle < 0){
53     panAngle = 0;
54    }
55
56    if (topLeft > downLeft) {
57      tiltAngle = tiltAngle - 1;
58      delay(waittime);
59    }
60    if (topRight > downRight) {
61      tiltAngle = tiltAngle - 1;
62      delay(waittime);
63    }
64    if (topLeft < downLeft) {
65      tiltAngle = tiltAngle + 1;
66      delay(waittime);
67    }
68    if (topRight < downRight) {
69      tiltAngle = tiltAngle + 1;
70      delay(waittime);
71    }
72    if (tiltAngle >120) {
73      tiltAngle = 120;
74    }
75    if (tiltAngle < 90) {
76      tiltAngle = 90;
77    }
78
79    panServo.write(tiltAngle);
80    tiltServo.write(panAngle);
81
82  }
```

3.2 Trimiterea Datelor la API-ul ThingSpeak

Acest cod este o schiță Arduino pentru un modul ESP8266 (probabil un ESP-01 sau similar) care este folosit pentru a se conecta la o rețea Wi-Fi și a trimite date către platforma ThingSpeak, probabil pentru a înregistra datele de la senzori. Iată o descriere mai simplificată a componentelor și funcționalităților principale:

1. **Biblioteci și constante**: Codul începe prin includerea bibliotecii `SoftwareSerial` și definirea constantelor pentru pini RX și TX. Acești pini sunt utilizați pentru comunicarea serială cu modulul ESP8266. De asemenea, sunt definite câteva șiruri de caractere pentru datele de autentificare ale rețelei Wi-Fi (SSID și parola), cheia API ThingSpeak, gazdă și port, precum și un identificator de câmp pentru ThingSpeak. Mai sunt și variabile întregi

pentru a număra comenzile reușite și timpul de execuție al comenzii, și un indicator boolean `found` pentru a verifica dacă a fost primit un răspuns.

2. **Inițializare Serială și ESP8266**: În funcția `setup()`, se inițiază comunicarea serială cu o viteză de 9600 pentru Arduino și 115200 pentru ESP8266. Apoi, se trimit mai multe comenzi AT pentru a configura ESP8266:
   - `AT`: Verifică răspunsul ESP8266.
   - `AT+CWMODE=1`: Setează ESP8266 în modul stație.
   - `AT+CWJAP`: Conectează la rețeaua Wi-Fi specificată folosind SSID și parola furnizate.
3. **Bucle Principale (`loop()`)**: În funcția `loop()`, sarcina principală este să citească în mod repetat datele senzorului (nu este implementată în codul furnizat) și să le trimită la ThingSpeak. Iată ce se întâmplă în buclă:
   - `valSensor` este actualizat cu datele senzorului (trebuie să implementați `getSensorData()`).
   - Se construiește un șir de caractere `getData` cu datele senzorului, cheia API și identificatorul de câmp.
   - Se trimit comenzi AT pentru a:
     - Activa conexiunile multiple (`AT+CIPMUX=1`).
     - Iniția o conexiune TCP cu ThingSpeak (`AT+CIPSTART`).
     - Trimite datele (`AT+CIPSEND`), cu lungimea datelor de trimis.

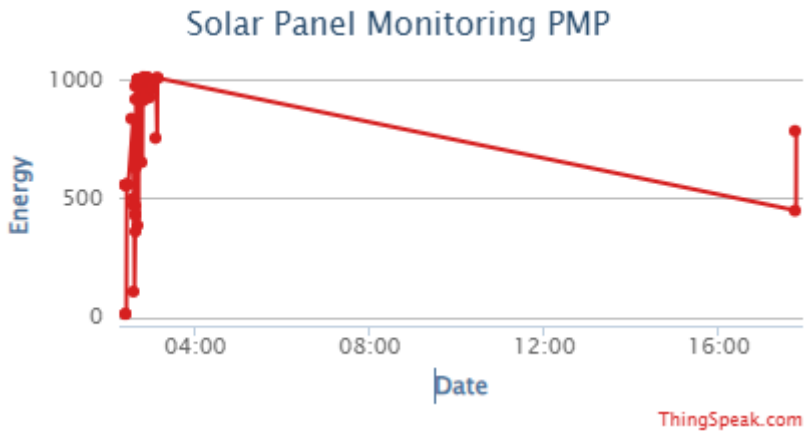

Figura 6: Chart-ul generat de datele transmise in API-ul ThingSpeak (random values in imagine)

```
#include <SoftwareSerial.h>
#define RX 2
#define TX 3
String AP = "Internet Wifi Name";       // AP NAME
String PASS = "Internet Wifi Password"; // AP PASSWORD
String API = "YOUR API WRITE KEY";   // Write API KEY
String HOST = "api.thingspeak.com";
String PORT = "80";
String field = "field1";
int countTrueCommand;
int countTimeCommand;
boolean found = false;
int valSensor = 1;
SoftwareSerial esp8266(RX,TX);

```

```
16
17  void setup() {
18    Serial.begin(9600);
19    esp8266.begin(115200);
20    sendCommand("AT",5,"OK");
21    sendCommand("AT+CWMODE=1",5,"OK");
22    sendCommand("AT+CWJAP=\""+ AP +"\",\""+ PASS +"\"",20,"OK");
23  }
24
25  void loop() {
26   valSensor = getSensorData();
27   String getData = "GET /update?api_key="+ API +"&"+ field +"="+String(valSensor);
28  sendCommand("AT+CIPMUX=1",5,"OK");
29   sendCommand("AT+CIPSTART=0,\"TCP\",\""+ HOST +"\","+ PORT,15,"OK");
30   sendCommand("AT+CIPSEND=0," +String(getData.length()+4),4,">");
31   esp8266.println(getData);delay(1500);countTrueCommand++;
32   sendCommand("AT+CIPCLOSE=0",5,"OK");
33  }
34
35  int getSensorData(){
36    //here goes the solar panel readings
37  }
38
39  void sendCommand(String command, int maxTime, char readReplay[]) {
40    Serial.print(countTrueCommand);
41    Serial.print(". at command => ");
42    Serial.print(command);
43    Serial.print(" ");
44    while(countTimeCommand < (maxTime*1))
45    {
46      esp8266.println(command);//at+cipsend
47      if(esp8266.find(readReplay))//ok
48      {
49        found = true;
50        break;
51      }
52
53      countTimeCommand++;
54    }
55
56    if(found == true)
57    {
58      Serial.println("OYI");
59      countTrueCommand++;
60      countTimeCommand = 0;
61    }
62
63    if(found == false)
```

```
  {
    Serial.println("Fail");
    countTrueCommand = 0;
    countTimeCommand = 0;
  }

  found = false;
 }
```

4 Schema electrica a componentelor

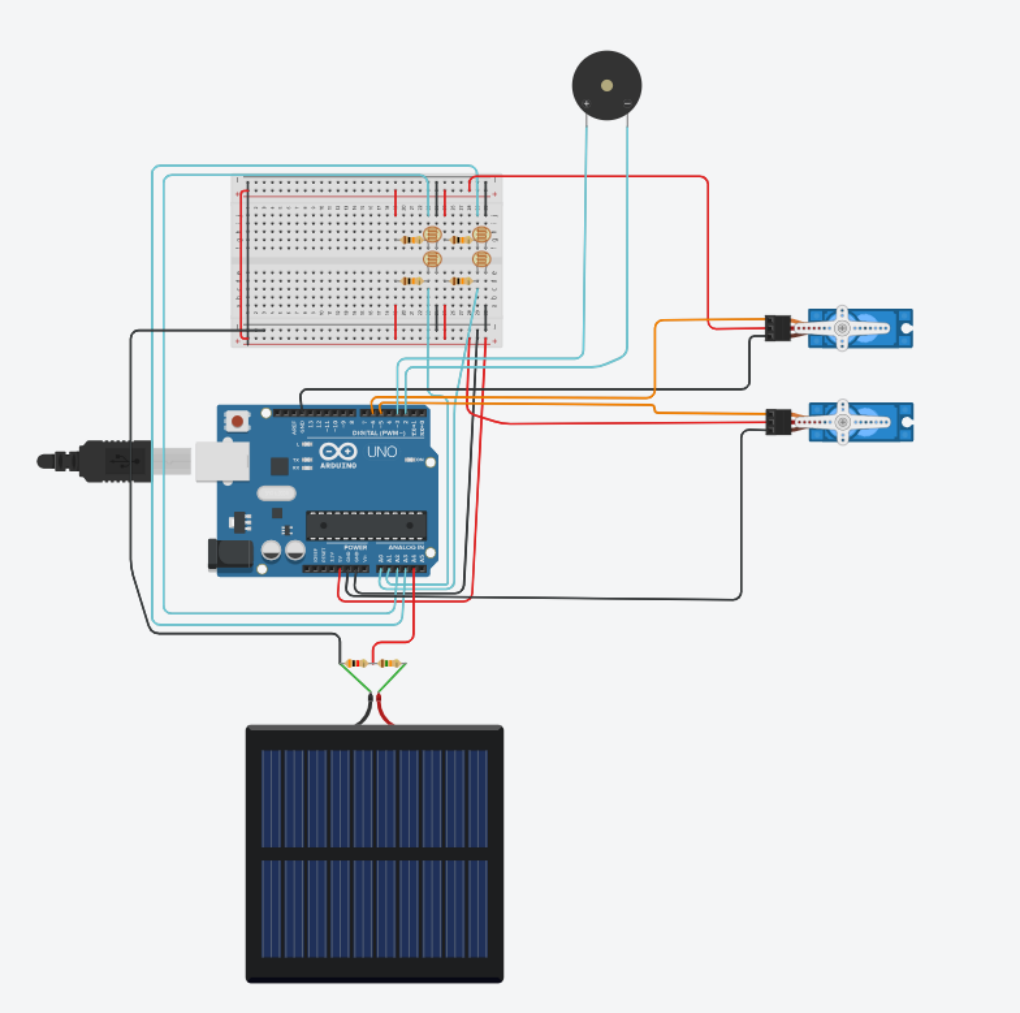

Figura 7: Schema electrica (ESP8266 inlocuit cu Piezo din cauza Tinkercad)

5 Bibliografie

Solar Tracker - GreatScott! Instructables

Solar Tracker YouTube Video

Sending Data to ThingSpeak API using ESP8266

Actual ThingSpeak page of the project's data