UNIVERSITATEA TEHNICĂ DIN CLUJ-NAPOCA

DOCUMENTAȚIE PROIECT

TEMA PROIECTULUI: SISTEM DE CLIMATIZARE (RĂCIRE)

La materia:

MĂSURATORI ELECTRICE ȘI SENZORI

NUME STUDENT: CREȚ MARIA-MAGDALENA

GRUPA: 30223, ANUL 2

FACULTATEA DE AUTOMATICĂ ȘI CALCULATOARE, SPECIALIZAREA CALCULATOARE ȘI TEHNOLOGIA INFORMAȚIEI

Descriere temă poiect

Un controller de temperatură bazat pe o placă Arduino este un dispozitiv electronic care reglează temperatura într-un mediu specific, cum ar fi o cameră de creștere a plantelor, sistem de climatizare, interioriorul unui dispozitiv de tipul laptop sau calculator. Acesta funcționează utilizând un senzor de temperatură pentru a măsura temperatura ambientală și apoi ajustează un dispozitiv de control, cum ar fi un un sistem de încălzire/răcire (în cazul de față, un ventilator), pentru a menține temperatura într-un interval dorit.

Avantajele ale utilizării acestui tip de controller de temperatură

a) Arduino este o platformă programabilă, ceea ce înseamnă că putem personaliza controller-ul pentru a satisface nevoile specifice ale aplicației. Puteți ajusta setările și algoritmul de control în funcție de cerințele pe care le aveam, cee ace ne oferă flexibilitate.
b) Un controller de temperatură poate contribui la economisirea energiei prin oprirea sau reducerea dispozitivelor de încălzire/răcire atunci cand nu sunt necesare.
c) Controlerele de temperatură bazate pe Arduino pot oferi o precizie ridicată în menținerea temperaturii dorite, ceea ce este esențial în aplicații precum culturile de plante sau incubatoare.

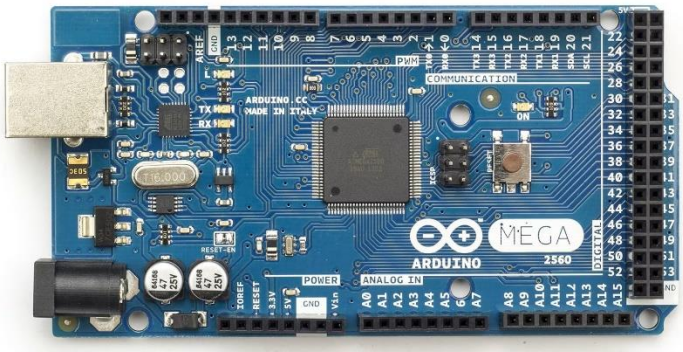
Descriere Proiect

Proiectul constă în dezvoltarea unui sistem de climatizare utilizând o placă Arduino Mega, un ventilator de 12 volți și un senzor de temperatură. Sistemul oferă două moduri de racire: automat și manual, cu posibilitatea de a regla viteza ventilatorului (pentru modul manual). În modul automat viteza ventilatorului crește odată cu creșterea temperaturii peste pragul de pornire a ventilatorului, de 25 de grade. Acest lucru se realizeaza pe baza senzorului de temperatură utilizat.

Componente

1. Arduino Mega
2. Ventilator de 12V
3. Senzor de temperatură
4. Buton de pornire/întrerupere
5. Buton de mărire viteză ventilator 10 kOhm

6. Buton de micșorare viteză ventilator 10 kOhm
7. 2 Rezistențe de 220 Ohm
8. Transistor NPN (baza-emițător-colector)
9. LED pentru modul manual (de culoare galbena, pornit atunci cand ventilatorul funcționează pe modul manual)
10. LED pentru indicarea vitezei ventilatorului (de culoare albastră)
11. Adaptor la curent de 12V pentru ventilator



Conexiuni

- Ventilatorul de 12V

- Conectează firul roșu la pinul de alimentare 12V al ventilatorului.

- Conectează firul negru la GND al ventilatorului.

- Senzorul de Temperatură

- Conectează pinul de semnal la unul dintre pinii analogici ai Arduino.

- Conectează firul de alimentare la Arduino.

- Conectează firul de GND la GND al Arduino.

- Butoanele de Control

- Conectează butonul de pornire/întrerupere la un pin digital al Arduino.

- Conectează butoanele de mărire și micșorare viteză la pini digitali separați ai Arduino.

- LED-uri

- Conectează LED-ul pentru modul manual la un pin digital al Arduino.

- Conectează LED-ul pentru indicarea vitezei ventilatorului la un alt pin digital al Arduino.

- Adaptor la Curent de 12V

- Conectează adaptorul la curent de 12V la o sursă de alimentare.

Cod Arduino

```
// Librarie Umiditate
#include <DHT.h>
#define TEMP_TYPE DHT11


// Pini
int dhtPin = 7;
int increaseButtonPin = 5;
int decreaseButtonPin = 4;
int manualButtonPin = 6;
int ledPowerFanPin = 10;
int ledManualPin = 11;
int fanPin = 9;

// Variabile
bool manual = false;
int powerFanMin = 30;
```

```
int powerFanMax = 255;
int stepIncrement = 51;

int powerManualFan = 0;
int powerTempFan = 0;
int powerFan = 0;
int ledPower = 0;

int okManualButton = 0;
int okIncreaseButton = 0;
int okDecreaseButton = 0;

float temp = 0.0;
float tempValue = 25.0;

int stateManualButton = 0;
int stateIncreaseButton = 0;
int stateDecreaseButton = 0;

// Pentru counter
int nr = 0;

DHT dht(dhtPin, TEMP_TYPE);

void setup()
{
  // Debug console
  Serial.begin(9600);
  delay(10);

  dht.begin();
  delay(500);

  pinMode(ledPowerFanPin, OUTPUT);
  pinMode(ledManualPin, OUTPUT);
  pinMode(fanPin, OUTPUT);

  pinMode(increaseButtonPin, INPUT);
  pinMode(decreaseButtonPin, INPUT);
  pinMode(manualButtonPin, INPUT);

  digitalWrite(fanPin, LOW);

  // Setari pentru counter intern - Nu modific
  TCCR1A = 0;
```

```
  TCCR1B = 0;
  OCR1A = 20;
  TCCR1B |= (1 << WGM12);
  TCCR1B |= (1 << CS10);
  TCCR1B |= (1 << CS12);
  TIMSK1 |= (1 << OCIE1A);
}

void loop()
{
  readTemp();
  stateManualButton = digitalRead(manualButtonPin);
  stateIncreaseButton = digitalRead(increaseButtonPin);
  stateDecreaseButton = digitalRead(decreaseButtonPin);

  if (stateManualButton == HIGH) {
    if (okManualButton == 0)
    {
      manual = !manual;
    }
    okManualButton = 1;
  } else {
    okManualButton = 0;
  }

  if (stateIncreaseButton == HIGH) {
    if (okIncreaseButton == 0)
    {
      if (powerManualFan + stepIncrement <= powerFanMax)
      {
        powerManualFan += stepIncrement;
      }
    }
    okIncreaseButton = 1;
  } else {
    okIncreaseButton = 0;
  }

  if (stateDecreaseButton == HIGH) {
    if (okDecreaseButton == 0)
    {
      if (powerManualFan - stepIncrement >= 0)
      {
        powerManualFan -= stepIncrement;
      }
```

```
    }
    okDecreaseButton = 1;
  } else {
    okDecreaseButton = 0;
  }


  if (manual == true) {
    digitalWrite(ledManualPin, HIGH);
    powerFan = powerManualFan;
  } else {
    powerFan = powerTempFan;
    powerManualFan = 0;
    digitalWrite(ledManualPin, LOW);
  }

  ledPower = map(powerFan, 0, powerFanMax, 0, 255);
  analogWrite(ledPowerFanPin, ledPower);

  Serial.print("Putere Ventilator : ");
  Serial.println(powerFan);
  delay(200);
}

void readTemp()
{
  temp = dht.readTemperature();

  if (temp > tempValue)
  {
    powerTempFan = map(temp, 15, 35, powerFanMin, powerFanMax);
  } else {
    powerTempFan = 0;
  }

  Serial.print("Temperatura : ");
  Serial.println(temp);
}

// Simulare scriere analog
ISR(TIMER1_COMPA_vect)
{
  if(nr<powerFan)
  {
    digitalWrite(fanPin, HIGH);
```

```
  }
  else{
    digitalWrite(fanPin, LOW);
  }

  nr++;
  if (nr == 255)
  {
    nr = 0;
  }
}
```

Codul de mai sus implementează un sistem de climatizare controlat de un senzor de temperatură, cu posibilitatea de control manual.

O explicație a principalelor funcționalități ale codului, pentru o mai bună înțelegere:

Bibliotecă și Definiții

#include <DHT.h>

#define TEMP_TYPE DHT11

- Se folosește o bibliotecă pentru senzorul de temperatură DHT.

- Se definește tipul senzorului de temperatură ca DHT11.

Pini și Variabile

// Definire pini

int dhtPin = 7;

int increaseButtonPin = 5;

int decreaseButtonPin = 4;

int manualButtonPin = 6;

int ledPowerFanPin = 10;

int ledManualPin = 11;

int fanPin = 9;

```
// Variabile
bool manual = false;
int powerFanMin = 30;
int powerFanMax = 255;
int stepIncrement = 51;

int powerManualFan = 0;
int powerTempFan = 0;
int powerFan = 0;
int ledPower = 0;

int okManualButton = 0;
int okIncreaseButton = 0;
int okDecreaseButton = 0;

float temp = 0.0;
float tempValue = 25.0;

int stateManualButton = 0;
int stateIncreaseButton = 0;
int stateDecreaseButton = 0;
```

- Se definesc pini pentru senzorul de temperatură, butoanele de control, LED-urile și ventilatorul.
- Se definesc variabile pentru controlul manual, setări ale ventilatorului, și altele.

Explicații cod

```
void loop()
{
  // Citire temperatură
  readTemp();

  // Citire stări butoane
  stateManualButton = digitalRead(manualButtonPin);
  stateIncreaseButton = digitalRead(increaseButtonPin);
  stateDecreaseButton = digitalRead(decreaseButtonPin);

  // Control manual
  if (stateManualButton == HIGH) {
    if (okManualButton == 0)
    {
      manual = !manual;
    }
    okManualButton = 1;
  } else {
    okManualButton = 0;
  }
  // Incrementare viteză manuală
  if (stateIncreaseButton == HIGH) {
    if (okIncreaseButton == 0)
    {
      if (powerManualFan + stepIncrement <= powerFanMax)
      {
```

```
      powerManualFan += stepIncrement;
    }
  }
  okIncreaseButton = 1;
} else {
  okIncreaseButton = 0;
}
// Decrementare viteză manuală
if (stateDecreaseButton == HIGH) {
  if (okDecreaseButton == 0)
  {
    if (powerManualFan - stepIncrement >= 0)
    {
      powerManualFan -= stepIncrement;
    }
  }
  okDecreaseButton = 1;
} else {
  okDecreaseButton = 0;
}
// Control ventilator în funcție de mod
if (manual == true) {
  digitalWrite(ledManualPin, HIGH);
  powerFan = powerManualFan;
} else {
  powerFan = powerTempFan;
  powerManualFan = 0;
  digitalWrite(ledManualPin, LOW);
```

```
  }

  // Actualizare LED viteză
  ledPower = map(powerFan, 0, powerFanMax, 0, 255);
  analogWrite(ledPowerFanPin, ledPower);

  // Afisare putere ventilator și temperatură
  Serial.print("Putere Ventilator : ");
  Serial.println(powerFan);
  delay(200);
}
```

- Se citește temperatura.
- Se verifică stările butoanelor și se efectuează acțiunile corespunzătoare.
- Se controlează ventilatorul în funcție de modul manual sau automat.
- Se afișează puterea ventilatorului și temperatura pe consola serială.

Funcție pentru Citirea Temperaturii

```
void readTemp()
{
  temp = dht.readTemperature();
  if (temp > tempValue)
  {
    powerTempFan = map(temp, 15, 35, powerFanMin, powerFanMax);
  } else {
    powerTempFan = 0;
  }
  Serial.print("Temperatura : ");
```

Serial.println(temp); }

- Funcție pentru citirea temperaturii de la senzorul DHT.

- Se ajustează puterea ventilatorului în funcție de temperatura citită.

Mod de utilizare

1. Pornire/Apăsare Buton Auto (Mod Automat):

- La pornirea sistemului, acesta va fi în modul automat implicit.

- Senzorul de temperatură monitorizează temperatura ambientală.

- Dacă temperatura depășește 25 de grade Celsius, ventilatorul pornește automat pentru a răci camera.

2. Schimbare la Mod Manual:

- Dacă se dorește preluarea controlului manual, se apasă butonul pentru modul manual.

- În modul manual, se pot utiliza butoanele de mărire și micșorare a vitezii ventilatorului pentru a ajusta nivelul de ventilație dorit.

3. Buton de Mărire și Micșorare a Vitezei Ventilatorului:

- În modul manual, utilizează butoanele dedicate pentru a regla viteza ventilatorului.

- Un LED albastru indică nivelul de viteză: cu cât mai intens este LED-ul, cu atât mai mare este viteza ventilatorului.

4. Buton de Mod:

- Acest buton permite schimbarea între modul automat și modul manual.

- Un LED indică modul ales: dacă LED-ul manual este aprins, ventilatorul este în modul manual; dacă este stins, ești în modul automat. (LED-ul de culoare galbenă)

5. LED pentru Modul Manual:

- Un LED dedicat indică dacă sistemul este în modul manual sau automat. Aprins înseamnă modul manual, stins înseamnă modul automat.

Prin combinarea acestor funcționalități, utilizatorul poate personaliza și controla experiența de climatizare în funcție de nevoile sale specifice. Sistemul oferă atât un mod automat inteligent, cât și un control manual, oferind un echilibru flexibil între confort și eficiență energetică. Utilizatorul are posibilitatea să schimbe între aceste moduri pentru a se adapta la schimbările de mediu sau preferințele personale.

Concluzii

Prin implementarea acestui sistem de climatizare, s-a creat o soluție eficientă care poate răspunde la cerințele de răcire atât automat, în funcție de temperatura ambientală, cât și manual, prin intermediul butoanelor. Integrarea unui senzor de temperatură și a unui controler Arduino Mega adaugă un grad ridicat de flexibilitate și automatizare în gestionarea climatizării într-o cameră.

Îmbunătățiri posibile: Este important să se mențină codul și conexiunile hardware bine documentate pentru a ușura întreținerea și eventuala extindere a sistemului. De asemenea se poate explora posibilitatea de a adăuga funcții suplimentare, cum ar fi programarea pentru anumite intervale de timp sau utilizarea unui ecran LCD pentru afișarea informațiilor despre sistem.

Cu toate acestea, acest proiect are scopul de a oferi o soluție practică pentru gestionarea climatizării într-o cameră, utilizând atât informații de programare hardware cât și informații cu referința la materia pentru care s-a realizat proiectul: *Măsuratori electrice și senzori*.