Răcitor frunte

Introducere

Nume: Subțirică Florin-Ioan
Grupa: 332CD
Îndrumător: Florin Stancu

Descriere

Proiectul constă în crearea unui sistem de monitorizare a temperaturii camerei. Când acesta detectează o creștere majoră a temperaturii, atunci ventilatorul atașat va porni și va începe procesul de răcire. Pe lângă funcția de răcire, acesta va mai avea și led-uri atașate.

Motivație

Ideea a pornit de la sesiunile de învățare pentru examenele finale din iunie, atunci când temperaturile de afară urcă foarte mult, astfel și corpul nostru resimțind o creștere de temperatură bruscă, dar și de la stresul provocat de volumul mare de informații care trebuiesc acumulate pentru examene (generând astfel 'încingerea creierului').

Cred că proiectul va fi util pentru toți studenții de la această facultate, dar mai ales pentru cei de anul 1, care nu știu ce-i așteaptă mai departe.

Descriere generală

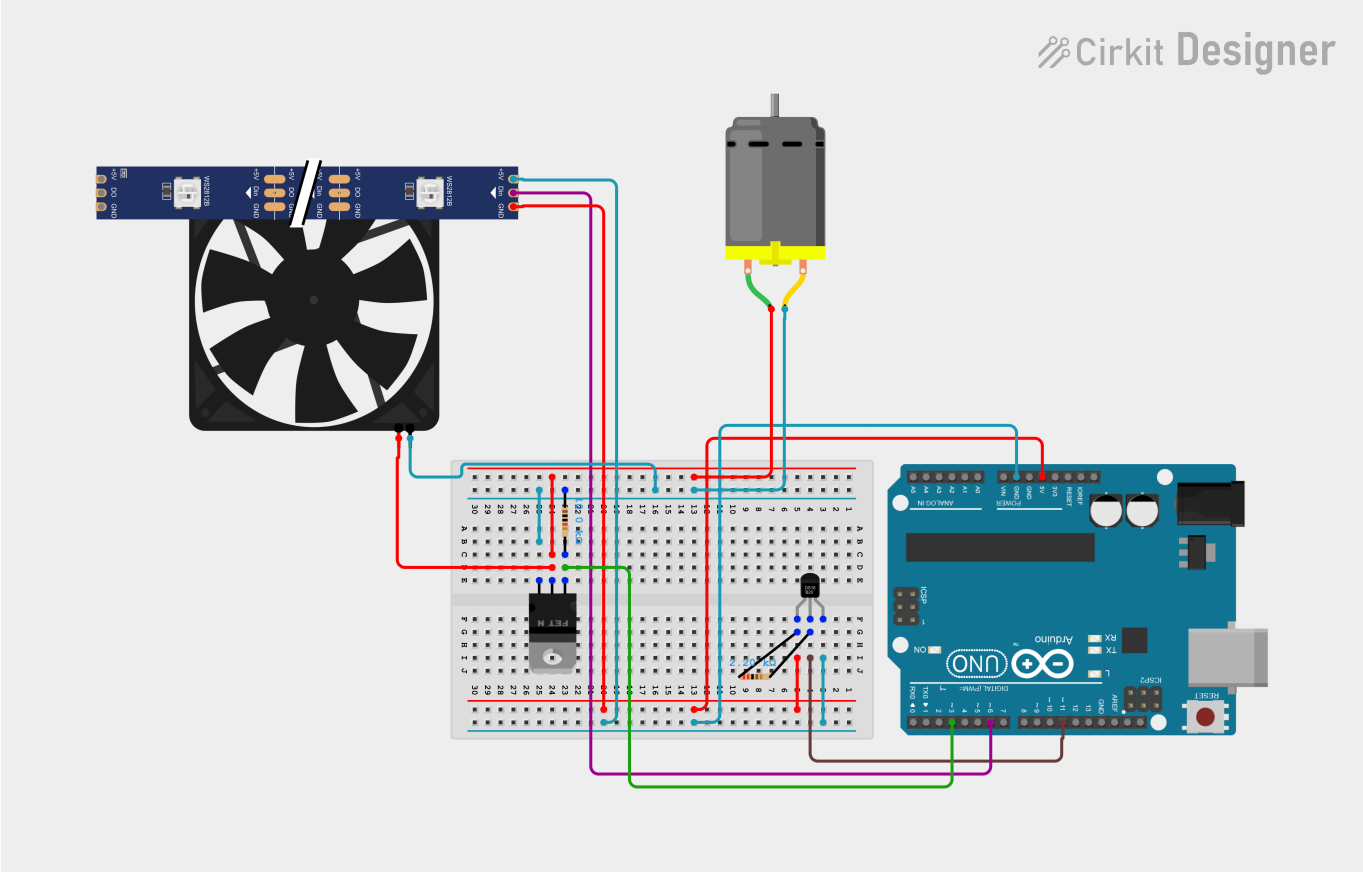
Senzorul de temperatură transmite prin ADC datele colectate către Arduino, iar Arduino transmite mai departe un semnal PWM către MOSFET, acesta controlând viteza ventilatorului (0%, 30%, 60%, 100%). În același timp, folosind I2C, Arduino va controla banda RGB LED, acesta modificându-i culoarea în funcție de viteza ventilatorului / temperatura indicată de senzor (0% - alb, 30% - galben, 60% - portocaliu, 100% - roșu).

Hardware Design

Lista componente

1 x Arduino UNO 16U2
1 x Senzor de temperatură DS18B20
1 x Ventilator PC 12V
1 x Rezistență 2.2kΩ
1 x Rezistență 10kΩ
1 x 15 cm bandă LED RGB WS2812
1 x Breadboard 400 puncte
1 x Set Jumper breadboard 140
1 x Tranzistor MOSFET N-MOS IRF540N

Diagrama circuit



1. Conectarea senzorului de temperatură (DS18B20):
- +Vs la 5V de pe Arduino.
- GND la GND de pe Arduino.
- Vout la un pin digital (D11) pentru citirea temperaturii.

2. Conectarea ventilatorului:

- GND la GND de pe Arduino.
- 5V la drain al MOSFET-ului.
- source al MOSFET-ului la GND.
- gate al MOSFET-ului la un pin digital PWM (D3) pentru controlul vitezei.

3. Conectarea benzilor LED RGB:
- GND la GND de pe Arduino.
- 5V la 5V de pe Arduino.
- DIN al primei benzi la un pin digital (D6).

4. Rezistențe:
- O rezistență de pull-down (2.2kΩ) între Digital OUT și GND
- O rezistență de pull-down (10kΩ) între gate și GND la MOSFET pentru a asigura că ventilatorul rămâne oprit când pinul de gate nu este activat.

Software Design

Codul pentru proiect este scris în **Arduino IDE** și folosește bibliotecile **Adafruit_NeoPixel** și **OneWire** pentru a controla un LED STRIP RGB și pentru a citi datele de la un senzor de temperatură DS18B20. În plus, codul controlează un ventilator în funcție de temperatura citită.

Biblioteca și definiții

#include <Adafruit_NeoPixel.h>

#include <OneWire.h>

#define LED_STRIP_PIN 6

#define NUM_LEDS 9

#define TEMPERATURE_PIN 11

#define FAN_PIN 3

- Adafruit_NeoPixel.h: Biblioteca pentru controlul LED-urilor NeoPixel.
- OneWire.h: Biblioteca pentru comunicarea cu senzorii OneWire, precum DS18B20.
- LED_STRIP_PIN: Pinul la care este conectat la LED STRIP.
- NUM_LEDS: Numărul de LED-uri din șir.
- TEMPERATURE_PIN: Pinul la care este conectat senzorul de temperatură DS18B20.
- FAN_PIN: Pinul la care este conectat ventilatorul.

Obiecte globale

Adafruit_NeoPixel strip = Adafruit_NeoPixel(NUM_LEDS, LED_STRIP_PIN, NEO_GRB + NEO_KHZ800);

OneWire ds(TEMPERATURE_PIN);

- strip: Un obiect pentru controlul șirului de LED-uri NeoPixel.
- ds: Un obiect pentru comunicarea cu senzorul de temperatură.

Funcția setup

void setup() {

```
Serial.begin(9600);
pinMode(TEMPERATURE_PIN, INPUT);
pinMode(FAN_PIN, OUTPUT);
pinMode(LED_STRIP_PIN, OUTPUT);
analogReference(INTERNAL);
strip.begin();
strip.show();
```

}

- Serial.begin(9600): Inițializează comunicarea serială la 9600 baud.
- pinMode(TEMPERATURE_PIN, INPUT): Configurează pinul pentru senzorul de temperatură ca intrare.
- pinMode(FAN_PIN, OUTPUT): Configurează pinul pentru ventilator ca ieșire.
- pinMode(LED_STRIP_PIN, OUTPUT): Configurează pinul pentru LED-uri ca ieșire.
- analogReference(INTERNAL): Setează referința analogică internă (1.1V pe majoritatea plăcilor Arduino).
- strip.begin(): Inițializează șirul de LED-uri.
- strip.show(): Resetează LED-urile, setând toate culorile la negru (oprit).

Funcția loop

void loop() {

```
byte i;
byte present = 0;
```

```
byte data[12];
byte addr[8];
```

```
if ( !ds.search(addr)) {
  ds.reset_search();
  delay(250);
  return;
}
```

```
present = ds.reset();
ds.select(addr);
ds.write(0x44);         // Start temperature conversion
```

```
delay(1000);

present = ds.reset();
ds.select(addr);
ds.write(0xBE);         // Read Scratchpad
```

```
for ( i = 0; i < 9; i++) {           // Read 9 bytes
  data[i] = ds.read();
}
uint16_t val = data[0] | ( (uint16_t) data[1] << 8);
float temperature = val * 0.0625; // Convert ADC value to temperature in
Celsius
```

```
Serial.print("\nTemperature: ");
Serial.print(temperature);
```

```
if (temperature < 24) {
  analogWrite(FAN_PIN, 255);
  setColor(255, 255, 255); // White
} else if (temperature >= 24 && temperature < 26) {
  analogWrite(FAN_PIN, 120); // 30% duty cycle
  setColor(255, 255, 0);    // Yellow
} else if (temperature >= 26 && temperature < 28) {
  analogWrite(FAN_PIN, 60); // 60% duty cycle
  setColor(255, 165, 0);     // Orange
} else {
  analogWrite(FAN_PIN, 0); // 100% duty cycle
  setColor(255, 0, 0);       // Red
}
```

```
delay(1000); // Read temperature and update fan speed every second
```

```
}
```

- Funcția loop este executată continuu și face următoarele:
  - Caută un dispozitiv OneWire (senzorul de temperatură) conectat.
  - Dacă nu găsește un senzor, resetează căutarea și așteaptă 250 ms.

- Dacă găsește un senzor, inițiază o conversie de temperatură și așteaptă 1 secundă.
- Citește datele de temperatură de la senzor.
- Convertește valoarea citită într-o temperatură în grade Celsius.
- Afișează temperatura pe monitorul serial.
- Controlează ventilatorul și LED-urile în funcție de temperatura citită:
  - Sub 24°C: Ventilatorul este pornit la putere maximă, LED-urile sunt albe.
  - Între 24°C și 26°C: Ventilatorul funcționează la 30% din putere, LED-urile sunt galbene.
  - Între 26°C și 28°C: Ventilatorul funcționează la 60% din putere, LED-urile sunt portocalii.
  - Peste 28°C: Ventilatorul este oprit, LED-urile sunt roșii.
- Așteaptă 1 secundă înainte de a repeta ciclul.

Funcția setColor

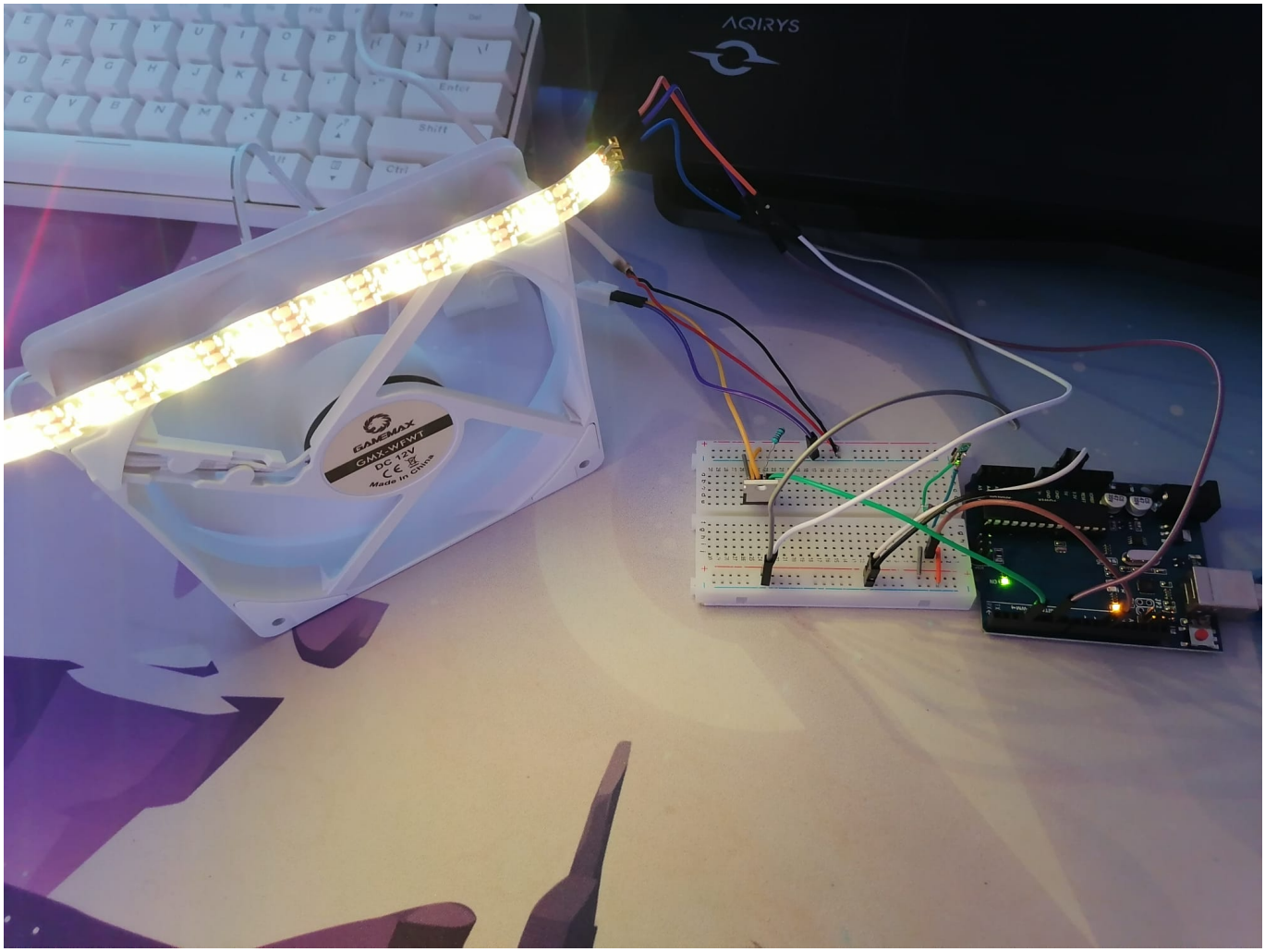
void setColor(uint8_t r, uint8_t g, uint8_t b) {

```
for (int i = 0; i < NUM_LEDS; i++) {
  strip.setPixelColor(i, strip.Color(r, g, b));
}
strip.show();
```

}

- Setează culoarea tuturor LED-urilor în funcție de valorile RGB primite ca parametri.
- strip.setPixelColor(i, strip.Color(r, g, b)): Setează culoarea LED-ului i la culoarea specificată.
- strip.show(): Actualizează șirul de LED-uri pentru a reflecta modificările de culoare.

Rezultate obţinute



Video demonstrativ: https://www.youtube.com/shorts/d3L1eB4awSk

Concluzii

În concluzie, a fost o adevărată plăcere să lucrez la acest proiect, unde am avut oportunitatea de a combina cunoștințele de hardware cu cele de software și de a fi creativ în găsirea soluțiilor pentru controlul cooler-ului și a led strip-ului rgb. Integrarea senzorului de temperatură și a mosfet-ului a fost amuzantă, mai ales că mi s-au stricat 2 senzori. Sper să mai am oportunitatea să lucrez la astfel de proiecte în viitor, însă poate de data asta nu se mai strică senzorii :))!

Download

proiect_pm.zip

Bibliografie/Resurse

1. **Arduino Libraries**
2. **Adafruit_NeoPixel Library Documentation**
3. **OneWire Library Documentation**
4. **Datasheet DS18B20**
5. **Datasheet MOSFET N-MOS IRF540N**
6. **Datasheet LED Strip RGB WS2812**