Sistemas Microcontrolados
Experimento 07 - Controle de Temperatura

Aluno: Deivid da Silva Galvão RA: 2408740
Aluno: Eduardo Yuji Yoshida Yamada RA: 2320606

1 Introdução

No presente relatório, desenvolvemos um programa capaz de monitorar a temperatura e ativar ou desativar automaticamente a ventoinha ao atingir um valor pré-definido. Posteriormente, removemos a ventoinha e passamos a controlar o aquecedor através de um controlador PID, calculando os valores de Kp e Ki.

2 Materiais e Métodos

2.1 Materiais Utilizados

- 1 LED branco;
- Jumpers (fios para conexão);
- Placa Arduino Mega 2560;
- Protoboard para montagem do circuito;
- Sensor de temperatura;
- Aquecedor;
- 2 Displays de 7 seguimentos;
- Ponte H.

2.2 Código Fonte Comentado

```
#include <OneWire.h>
#include <DallasTemperature.h>
#define DADOS 2
#define LED_ALARME 12
#define TEMPERATURA_ALARME 40  // Temperatura do alarme em
   graus Celsius

// Pinos do display de 7 segmentos
```

```
int ledA = 22;
int ledB = 24;
int ledC = 26;
int ledD = 28;
int ledE = 30;
int ledF = 32;
int ledG = 34;
int jumper1 = 36;  // Display da dezena
int jumper2 = 38;  // Display da unidade
int In1 = 4;
int In2 = 5;
int ventiladorIn3 = 6;
int ventiladorIn4 = 7;
int vel = 0;
float erro, erroant, saida, kp, ki;

// Variaveis do display
int unidade, dezena, valor, pot_ts, status;

OneWire oneWire(DADOS);
DallasTemperature sensors(&oneWire);

// Tabela de segmentos para os numeros 0-9
byte segmentos[10][7] = {
  { 1, 1, 1, 1, 1, 1, 0 },  // 0
  { 0, 1, 1, 0, 0, 0, 0 },  // 1
  { 1, 1, 0, 1, 1, 0, 1 },  // 2
  { 1, 1, 1, 1, 0, 0, 1 },  // 3
  { 0, 1, 1, 0, 0, 1, 1 },  // 4
  { 1, 0, 1, 1, 0, 1, 1 },  // 5
  { 1, 0, 1, 1, 1, 1, 1 },  // 6
  { 1, 1, 1, 0, 0, 0, 0 },  // 7
  { 1, 1, 1, 1, 1, 1, 1 },  // 8
  { 1, 1, 1, 1, 0, 1, 1 }   // 9
};

void setup() {
  pinMode(LED_ALARME, OUTPUT);
  sensors.begin();

  // Segmentos
  pinMode(ledA, OUTPUT);
  pinMode(ledB, OUTPUT);
  pinMode(ledC, OUTPUT);
  pinMode(ledD, OUTPUT);
  pinMode(ledE, OUTPUT);
  pinMode(ledF, OUTPUT);
```

```
  pinMode(ledG, OUTPUT);
  pinMode(jumper1, OUTPUT);
  pinMode(jumper2, OUTPUT);
  //ventilador ar condicionado
  pinMode(ventiladorIn3, OUTPUT);
  pinMode(ventiladorIn4, OUTPUT);
  //hotend
  pinMode(In1, OUTPUT);
  pinMode(In2, OUTPUT);
  //butao
  pinMode(47, INPUT_PULLUP);
  pinMode(49, INPUT_PULLUP);

  pinMode(6, OUTPUT);

  Serial.begin(9600);

  // Timer para multiplexacao
  TCCR1A = 0;
  TCCR1B = 0;
  TCCR1B |= (1 << CS10);  // sem prescaler
  TCNT1 = 0xC180;         // preload para ~2ms
  TIMSK1 |= (1 << TOIE1);
}

void loop() {
  if (digitalRead(47) == 0) {
    vel += 10;
    if (vel > 255) {
      vel = 255;
    }
  }
  if (digitalRead(49) == 0) {
    vel -= 10;
    if (vel < 0) {
      vel = 0;
    }
  }

  analogWrite(ventiladorIn3, 0);
  analogWrite(ventiladorIn4, vel);

  sensors.requestTemperatures();
  float temp = sensors.getTempCByIndex(0);
  valor = constrain(int(temp), 0, 99);  //entre 0 e 99

  dezena = valor / 10;
```

```
  unidade = valor % 10;

  kp = 5;
  ki = 0.5;
  erroant = erro;
  erro = 40-temp;
  saida = kp * erro + ki * (erro+erroant);
  status = 1;
  vel = 0;
  if (saida < 0){
    saida = 0;
    vel = 255;// Por precaucao
    status = 0;
  }
  analogWrite(4, saida);
  analogWrite(5,0);

  Serial.print("Saida:");
  Serial.println(saida);

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

  Serial.print("Velocidade:");
  Serial.println(vel);

  Serial.print("Esquentando:");
  if (status == 1){
    Serial.println("Sim");
  } else {
    Serial.println("Nao");
  }

/*
  if (valor > TEMPERATURA_ALARME) {
    digitalWrite(LED_ALARME, HIGH);
    analogWrite(In1, 0);
    analogWrite(In2, 0);
    vel = 255;
    status = 0;
  } else if (valor < TEMPERATURA_ALARME) {
    digitalWrite(LED_ALARME, LOW);
    analogWrite(In1, 255);
    analogWrite(In2, 0);
    vel = 0;
    status = 1;
  }
```

```
*/
}
// Interrupcao do TIMER1 para multiplexar os displays
ISR(TIMER1_OVF_vect) {
  TCNT1 = 0xC180;  // Reinicia TIMER para intervalo de ~2ms
  pot_ts++;

  if (pot_ts % 2 == 0) {
    // Mostra dezena
    digitalWrite(jumper1, HIGH);
    digitalWrite(jumper2, LOW);
    mostrarNumero(dezena);
  } else {
    // Mostra unidade
    digitalWrite(jumper1, LOW);
    digitalWrite(jumper2, HIGH);
    mostrarNumero(unidade);
  }
}

// Mostra o digito no display
void mostrarNumero(int numero) {
  digitalWrite(ledA, segmentos[numero][0]);
  digitalWrite(ledB, segmentos[numero][1]);
  digitalWrite(ledC, segmentos[numero][2]);
  digitalWrite(ledD, segmentos[numero][3]);
  digitalWrite(ledE, segmentos[numero][4]);
  digitalWrite(ledF, segmentos[numero][5]);
  digitalWrite(ledG, segmentos[numero][6]);
}
```

Listing 1: Código

3 Resultados

Durante o experimento, o sistema monitorou eficientemente a temperatura ambiente com um sensor conectado ao Arduíno. Os valores foram exibidos em tempo real nos dois *displays* de 7 segmentos.

Quando a temperatura ficava abaixo de 40°C, o aquecedor era ativado via ponte H. O controle PI ajustava a saída conforme o erro entre a temperatura medida e a desejada, garantindo estabilidade. À medida que a temperatura se aproximava do valor de referência, a velocidade do ventilador aumentava proporcionalmente, controlada por *PWM*, ajudando na dissipação de calor. Caso a temperatura ultrapassasse 40°C, o LED de alarme era acionado, sinalizando superaquecimento.

Já na segunda parte do experimento, com a ventoinha removida, o circuito utilizava os valores de kp e ki, juntamente com a temperatura e, dessa forma, regulava o quanto ele iria esquentar ou não. Dessa forma, quanto mais perto de 40, menor o valor da saída. Caso ele ultrapassasse o valor de 40°C, para o valor da saída não ficar negativo, deixamos como 0; dessa forma, ele irá desligar o aquecedor e irá resfriar naturalmente.

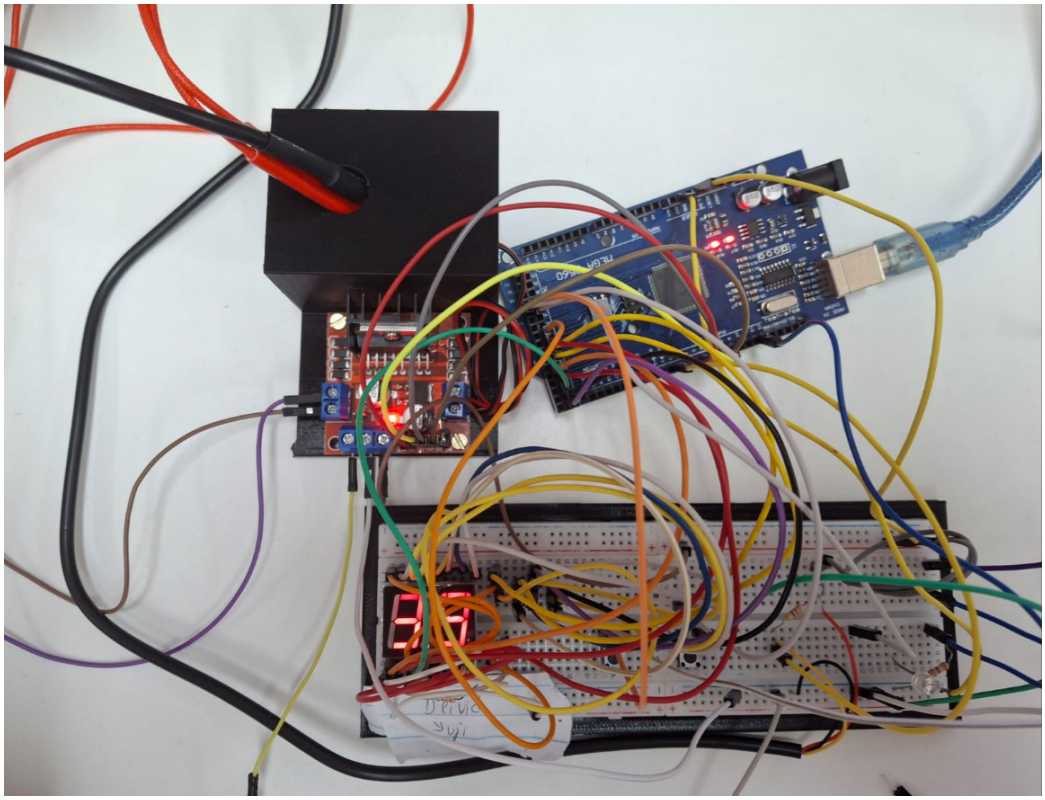

Figure 1: Montagem física do circuito no laboratório

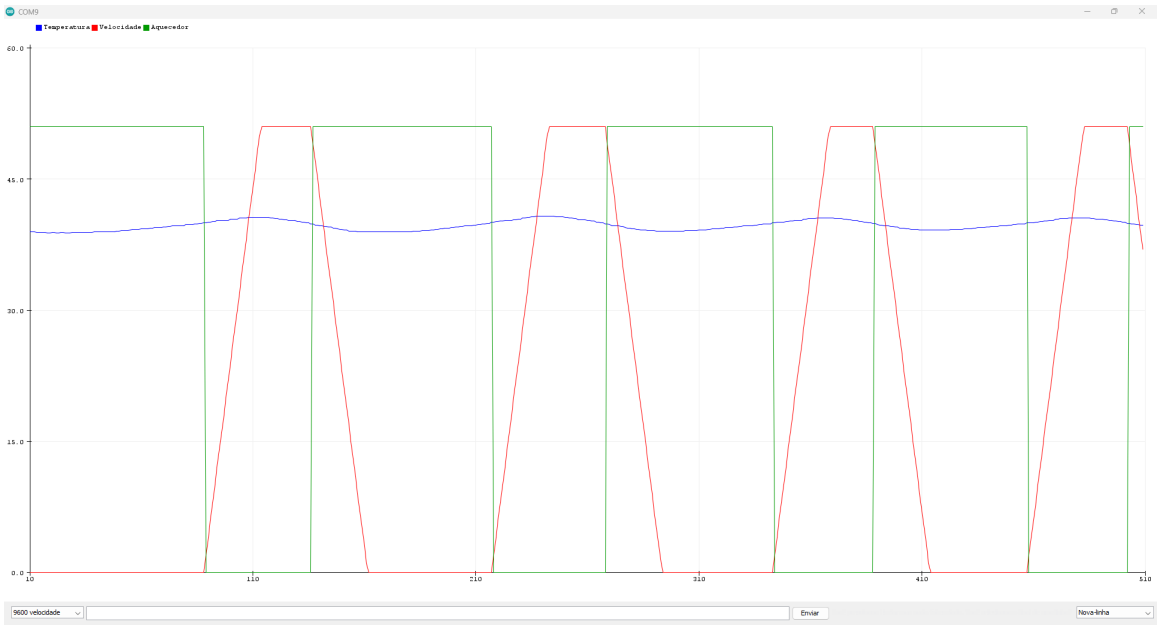

Figure 2: Gráfico Temperatura, Velocidade e Aquecedor ligado

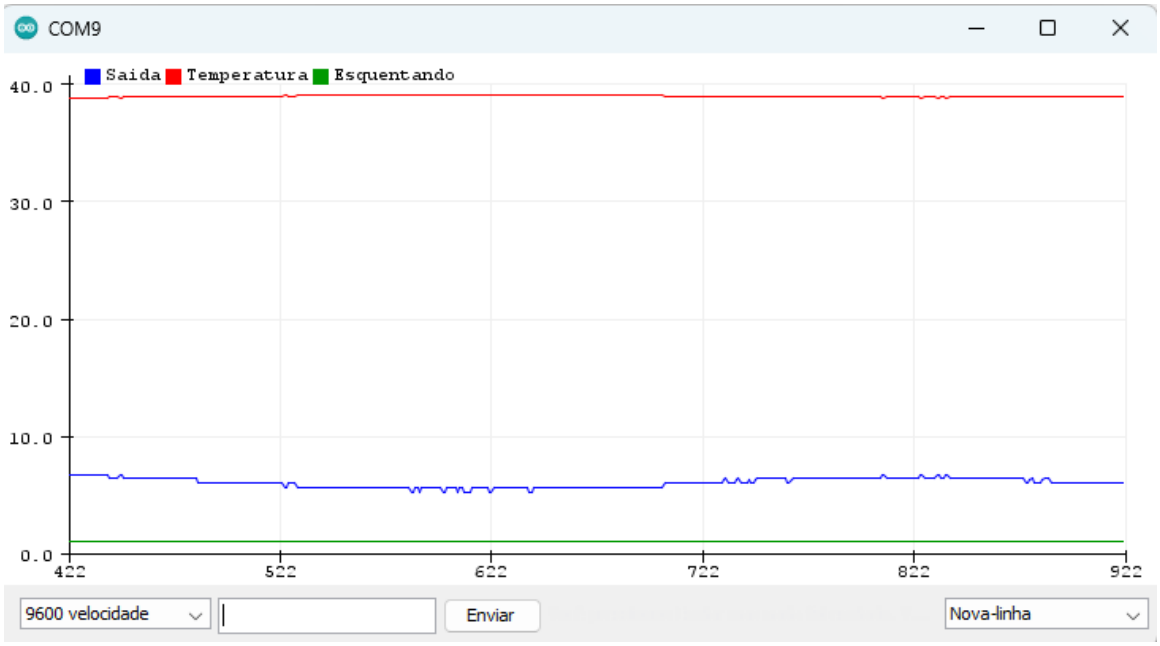

Figure 3: Gráfico com o controlador PID

4 Conclusão

O experimento implementou com sucesso um sistema de controle de temperatura usando Arduíno Mega 2560, sensor digital, ventilador e aquecedor. Com controle

proporcional-integral (PI), o sistema reagiu dinamicamente às variações térmicas, ativando ou desativando os atuadores conforme necessário. A exibição nos *displays* de 7 segmentos e a comunicação serial permitiram o monitoramento em tempo real, enquanto a multiplexação otimizou o uso dos pinos digitais. O ventilador foi acionado ao atingir 40°C, e o aquecedor desligado, com um LED indicando sobreaquecimento.