**CENTRO PAULA SOUZA**
**ESCOLA TÉCNICA ESTADUAL DE SÃO PAULO – ETEC SÃO PAULO**
**CURSO TÉCNICO EM ELETROELETRÔNICA**

**GABRIEL SANTOS PAIVA**
**DANIEL NACER GONZÁLEZ**
**YURI ARCANJO DE OLIVEIRA MARTINS**
**LUCIANO PEREIRA DOS SANTOS LEMOS**

# VENTILADOR ELÉTRICO COM ACIONAMENTO AUTOMÁTICO

**SÃO PAULO**
**2025**

**• Gabriel Santos Paiva.**

**• Daniel Nacer González.**

**• Yuri Arcanjo de Oliveira Martins.**

**• Luciano Pereira dos Santos Lemos.**

# VENTILADOR ELÉTRICO COM ACIONAMENTO AUTOMÁTICO

Trabalhode Conclusão de Curso apresentado com requisito para aprovação no curso Técnico de Eletroeletrônica pela escola técnica ETEC de São Paulo.

Orientador: Ricardo Beltrame.

**SÃO PAULO**

**2025**

## RESUMO

O projeto apresentado tem como base o desenvolvimento de um sistema automatizado para controle térmico utilizando o microcontrolador Arduino, com acionamento de um ventilador conforme a variação da temperatura ambiente. A motivação surgiu da necessidade de soluções práticas e acessíveis para automação residencial ou industrial, capazes de atuar de forma autônoma na regulação de ambientes com base em condições climáticas específicas.O objetivo principal foi construir um circuito eletrônico capaz de ler dados de temperatura por meio de um termistor NTC 10kΩ, processar essas informações com o Arduino e acionar um ventilador através de um relê quando determinada temperatura limite fosse atingida. Como funcionalidade adicional, o projeto permite a integração opcional de um display LCD para exibição em tempo real da temperatura.A relevância do projeto está na sua simplicidade, baixo custo e ampla aplicabilidade, podendo ser utilizado em ambientes domésticos, estufas, aquários, chocadeiras ou até mesmo em sistemas educacionais. Sua modularidade permite adaptações e melhorias futuras, como uso de comunicação sem fio ou armazenamento de dados. Como resultado, o sistema mostrou-se eficiente no monitoramento e resposta às variações térmicas, com acionamento correto do ventilador conforme programado. Pequenas oscilações nas leituras foram observadas, mas podem ser corrigidas com ajustes na alimentação elétrica e posicionamento do sensor. O projeto se mostrou uma excelente ferramenta prática tanto para aprendizado em eletrônica e automação quanto para uso cotidiano.


**Palavras-chave:** Arduino, Automação Térmica, Controle de Temperatura.

**ABSTRACT**

The projectpresentedisbased on the development of an automated system for thermal control using the Arduino microcontroller, with a fan activatedaccording to the variation in ambient temperature. The motivation arosefrom the need for practical and affordable solutions for home or industrial automation, capable of acting autonomously in regulatingenvironmentsbased on specificclimatic conditions. The main objective was to build an electronic circuit capable of readingtemperature data through a 10kΩ NTC thermistor, processingthis information with the Arduino and activating a fan through a relaywhen a certain limittemperaturewasreached. As an additionalfunctionality, the projectallows the optionalintegration of an LCD display for real-time display of the temperature. The relevance of the project lies in itssimplicity, lowcost and wideapplicability, and it can beused in domesticenvironments, greenhouses, aquariums, incubators or even in educationalsystems. Itsmodularityallows for future adaptations and improvements, such as the use of wireless communication or data storage. As a result, the system proved to be efficient in monitoring and responding to thermal variations, with the correct activation of the fan as programmed. Small fluctuations in the readingswereobserved, but these can becorrected by adjusting the power supply and positioning of the sensor. The projectproved to be an excellent practicaltool for bothlearning about electronics and automation and for everyday use.

**Keywords:** Arduino, Thermal Automation, Temperature Control**SUMÁRIO**

*INDTRODUÇÃO*.....6
*1. DESENVOLVIMENTO*.....*8*
1.1. Materiais Utilizados.....8
1.1.1. Placa Arduino Uno.....8
1.1.2. Termistor NTC 10Kω.....8
1.1.3. Resistor de 10kΩ.....8
1.1.4. Relê de 5V.....8
1.1.5. Display LCD 16x2 (opcional).....9
1.1.6.Protoboard.....9
1.1.7.Fios de Ligação.....9

1.1.8.Fios de Ligação.....9
1.1.9.Fonte de Alimentação.....9
1.1.10. Ventilador e Tomadas.....10
1.1.11.Potenciômetro de 10kΩ (para ajuste do display LCD).....10
2. *MONTAGEM DO CIRCUITO ELETRÔNICO*.....10
2.1. Sensor de Temperatura (Termistor + Resistor).....10
2.2. Funcionamento do Divisor de Tensão.....10
2.3. Ligação no Pino Analógico A0 do Arduino.....11
2.4. Conversão do Valor Lido (0 a 1023) em Temperatura Real (ºC)....11
2.4.1. Passos da Conversão:.....11
2.4.2. Exemplo Prático no Código.....12
2.6. Display LCD (Opcional).....15
2.6.1. Pinagem do Display LCD (Explicando a Contagem dos Pinos).....15
2.6.2. Função de Cada Pino.....16
2.6.3.Uso do Potenciômetro para Ajuste do Contraste do Display.....16
2.6.4. Ligações nos Pinos Digitais do Arduino (ex.:12,11,5,4,3 e2).....17
2.6.5. Exemplo de Código para Inicialização do Display LCD.....17
2.7. Programação do Arduino.....18
2.7.1. Código Fonte Completo.....18
2.7.2. Explicação Linha a Linha.....20
2.7.3. Possíveis Alterações no Código.....23
2.8. Testes Realizados.....24
2.8.1. Teste de Leitura do Termistor (Comparação Manual vs. Exibida no Display/Monitor Serial).....24
2.8.2. Verificação do Funcionamento do Relê e do Ventilador.....24
2.8.3.Simulação de Aumento de Temperatura (Aproximação do Dedo ou Uso de Secador).....25
2.8.4. Análise do Comportamento do Sistema em Diferentes Condições Ambientais.....25
2.8.5. Observações Importantes Durante os Testes.....26

*CONCLUSÃO*.....28
*REFERÊNCIAS*.....30

*INTRODUÇÂO*

Atualmente, a automação tem se mostrado uma ferramenta essencial para melhorar a qualidade de vida, aumentar a eficiência energética e proporcionar conforto ambiental em diversos contextos. Diante disso, sistemas automatizados de controle térmico têm ganhado destaque por sua aplicabilidade em ambientes domésticos, industriais e até mesmo em equipamentos médicos (COELHO et al., 2023; PEREIRA; GRANDE, 2021). Em especial, o uso de dispositivos portáteis que possam monitorar e regular a temperatura ambiente apresenta-se como uma solução prática e acessível para situações cotidianas, como resfriamento de espaços fechados, controle de estufas, incubadoras ou até mesmo em aplicações hospitalares. Nesse contexto, tecnologias baseadas no microcontrolador Arduino vêm sendo amplamente utilizadas por oferecerem flexibilidade, baixo custo e facilidade de integração com sensores e atuadores (GHEDIN et al., 2022; JÚNIOR et al., 2022).

O problema central abordado neste trabalho está relacionado à necessidade de um sistema de controle térmico mais acessível, eficiente e adaptável às condições locais. Muitos dos equipamentos comerciais disponíveis no mercado possuem custos elevados, complexidade técnica ou limitações quanto à personalização. Assim, desenvolver uma alternativa automatizada, utilizando componentes de baixo custo e fácil aquisição, como o termistor NTC 10k, relê e display LCD, pode contribuir significativamente para democratizar o acesso a soluções de climatização inteligente.

Diante disso, o objetivo geral deste projeto é desenvolver um sistema automatizado e portátil capaz de controlar a temperatura ambiente através do acionamento automático de um ventilador, utilizando o microcontrolador Arduino como unidade central de processamento. Para tanto, são estabelecidos como objetivos específicos: utilizar o Arduino Uno como placa de controle; implementar um sensor de temperatura tipo termistor NTC 10kΩ para a coleta de dados ambientais; integrar um módulo relê para o acionamento do ventilador com base nos valores lidos; e, opcionalmente, incluir um display LCD para exibição em tempo real da temperatura ambiente.

A relevância deste projeto reside na possibilidade de aplicação em diferentes cenários práticos, como salas de computadores, pequenos ambientes residenciais,

estufas agrícolas e até mesmo em projetos educacionais de eletrônica e automação. Além disso, ao utilizar materiais de baixo custo e código-fonte aberto, o sistema proposto pode ser replicado e adaptado por usuários leigos ou estudantes, incentivando a aprendizagem prática e promovendo soluções sustentáveis de gerenciamento térmico. A portabilidade aliada à automação representa um avanço na busca por soluções tecnológicas mais acessíveis e customizáveis, contribuindo não apenas para o campo acadêmico, mas também para a sociedade em geral.

*DESENVOLVIMENTO*

## 1.1. MATERIAIS UTILIZADOS

Para a realização do projeto de automação do ventilador com base na temperatura ambiente, foi utilizada uma série de componentes eletrônicos e materiais acessórios, escolhidos por sua eficiência, baixo custo e facilidade de aquisição. Esses elementos permitem o funcionamento do sistema de forma prática, robusta e adaptável a diferentes necessidades de uso. A seguir, são detalhados cada um dos materiais empregados no projeto.

### 1.1.1. Placa Arduino Uno

O Arduino Uno é o microcontrolador central do projeto, responsável por processar os dados coletados pelo sensor de temperatura e controlar o acionamento do ventilador. Sua escolha se deve à ampla disponibilidade no mercado, facilidade de programação e compatibilidade com diversos sensores e atuadores. Além disso, o Arduino oferece portas digitais e analógicas suficientes para conectar todos os componentes necessários, tornando-o ideal para projetos de automação como este (GHEDIN et al., 2022; JÚNIOR et al., 2022).

### 1.1.2. Termistor NTC 10kΩ

O termistor NTC (Negative TemperatureCoefficient) de 10kΩ é o sensor responsável pela medição da temperatura ambiente. Ele varia sua resistência em função da temperatura, permitindo ao Arduino realizar leituras precisas. O termistor é conectado em conjunto com um resistor fixo de 10kΩ formando um divisor de tensão, que converte as variações de resistência em valores analógicos compreensíveis pelo microcontrolador (COELHO et al., 2023). Este tipo de sensor é amplamente utilizado em projetos de baixo custo devido à sua confiabilidade e simplicidade de integração.

### 1.1.3. Resistor de 10kΩ

O resistor de 10kΩ é fundamental para o correto funcionamento do circuito do termistor, sendo usado como parte do divisor de tensão. Ele garante que o sinal analógico enviado ao Arduino seja proporcional à temperatura medida, possibilitando a calibração e a interpretação correta dos dados obtidos (PEREIRA; GRANDE, 2021).

### 1.1.4. Relê de 5V

O relê de 5V atua como um interruptor controlado pelo Arduino, permitindo o acionamento de cargas maiores, como o ventilador. Como o Arduino não pode fornecer corrente elétrica suficiente diretamente para dispositivos mais potentes, o relê é essencial para isolar o circuito lógico do circuito de potência, garantindo segurança ao sistema e evitando danos ao microcontrolador. O relê utilizado possui três terminais: um para alimentação positiva, outro para o negativo e um terceiro para o sinal vindo do Arduino (geralmente conectado à porta digital 8) (JÚNIOR et al., 2022).

#### 1.1.5. Display LCD 16x2 (opcional)

O display LCD de 16 caracteres por 2 linhas é um componente opcional incluído no projeto para exibir em tempo real a temperatura ambiente. Ele permite maior interatividade com o usuário e facilita a visualização das informações coletadas pelo termistor. Para seu funcionamento, é necessário o uso de um potenciômetro de 10kΩ , que ajusta o contraste do display, além de conexões com os pinos digitais do Arduino (normalmente os pinos 12, 11, 5, 4, 3 e 2), conforme mostrado no vídeo (GHEDIN et al., 2022).

#### 1.1.6. Protoboard

A protoboard é uma placa de circuito reutilizável utilizada para montar e testar o circuito sem a necessidade de soldagem. Ela permite a fácil conexão entre os componentes e facilita a identificação de possíveis erros durante a montagem. Seu uso é indispensável em projetos prototipados, especialmente em fases de desenvolvimento e testes (COELHO et al., 2023).

#### 1.1.7. Fios de Ligação

Os fios de ligação são responsáveis por conectar todos os componentes do circuito na protoboard e ao Arduino. Eles podem ser do tipo "jumpers" macho-macho ou macho-fêmea, dependendo da aplicação. No projeto, foram utilizados fios coloridos para facilitar a identificação das funções de cada conexão, como vermelho para positivo, preto para negativo e cores variadas para sinais (PEREIRA; GRANDE, 2021).

#### 1.1.8. Fonte de Alimentação

O sistema pode ser alimentado através de um cabo USB conectado ao computador, o que é conveniente para testes e programação. No entanto, para uso autônomo e portátil, recomenda-se utilizar uma bateria ou fonte externa, como

baterias recarregáveis ou adaptadores de energia, evitando oscilações causadas por flutuações na alimentação via USB (GHEDIN et al., 2022).

#### 1.1.9. Ventilador

O ventilador é o dispositivo a ser automatizado, acionado sempre que a temperatura ambiente ultrapassar o valor pré-definido no código. Para sua instalação, foram utilizadas uma bateria, conectadas ao relê, permitindo que o ventilador seja ligado ou desligado automaticamente com base na leitura do sensor. Esse mesmo esquema pode ser adaptado para outros aparelhos, como aquecedores, resistências ou até lâmpadas incandescentes (COELHO et al., 2023).

#### 1.1.10. Potenciômetro de 10kΩ (para ajuste do display LCD)

O potenciômetro de 10kΩ é utilizado exclusivamente para ajustar o contraste do display LCD, garantindo uma boa visibilidade das informações exibidas. Ele é conectado entre os trilhos de alimentação positiva e negativa, com seu pino central ligado ao pino VO (pino 3) do display (GHEDIN et al., 2022).

## *2. MONTAGEM DO CIRCUITO ELETRÔNICO*

*Sensor de Temperatura (Termistor + Resistor)*

A montagem do circuito responsável pela detecção da temperatura utiliza dois componentes principais: o termistor NTC 10kΩ e um resistor fixo de 10kΩ , conectados em uma configuração conhecida como divisor de tensão. Essa estrutura permite ao Arduino interpretar as variações de resistência causadas pelas mudanças na temperatura ambiente, convertendo-as em valores digitais compreensíveis pelo microcontrolador.

### 2.1. Funcionamento do Divisor de Tensão

O termistor NTC (Negative TemperatureCoefficient) é um sensor cuja resistência elétrica diminui à medida que a temperatura aumenta. Ele é associado a um resistor fixo de 10kΩ, formando um circuito divisor de tensão, conforme descrito a seguir:

A alimentação positiva (5V do Arduino) é conectada a uma extremidade do resistor fixo de 10kΩ.

A outra extremidade desse resistor é ligada à extremidade do termistor.

A extremidade restante do termistor é conectada ao GND (terra).

O ponto intermediário entre o resistor e o termistor é conectado ao pino analógico A0 do Arduino.

Esse arranjo gera uma tensão variável no ponto médio proporcional à resistência do termistor, que por sua vez depende da temperatura. O Arduino realiza a leitura dessa tensão através do seu conversor analógico-digital (ADC), obtendo um valor entre 0 e 1023 , correspondente a uma faixa de tensão entre 0V e 5V .

### 2.2. Ligação no Pino Analógico A0 do Arduino

O sinal coletado do divisor de tensão é conectado diretamente ao pino analógico A0 do Arduino Uno. Este pino é responsável por ler o nível de tensão presente no ponto médio entre o resistor fixo e o termistor.

Fio Amarelo (como mostrado no vídeo) sai do ponto central entre o resistor e o termistor e vai até o pino A0 do Arduino.

Esta conexão é crucial, pois é por meio dela que o Arduino consegue obter os dados de variação térmica do ambiente.

### 2.3. Conversão do Valor Lido (0 a 1023) em Temperatura Real (ºC)

Após a leitura do valor analógico (entre 0 e 1023), o Arduino converte esse número em uma temperatura real em graus Celsius (ºC). Esse cálculo envolve algumas etapas de programação utilizando fórmulas físicas e matemáticas, como a equação de Steinhart-Hart , simplificada para uso com termistores NTC comum:

#### 2.3.1. Passos da Conversão:

Leitura Bruta do ADC:

O Arduino lê o valor bruto usando a função **analogRead(A0)**.

Cálculo da Resistência do Termistor:

Utilizando a fórmula:

$$R\_termistor = R\_resistor * (1023.0 / valor\_ADC - 1)$$

onde **R_resistor** é 10.000 Ω.

Cálculo da Temperatura em Kelvin:

Aplicando a equação simplificada de Steinhart-Hart:

$$tempK = 1 / (A + B * ln(R\_termistor) + C * (ln(R\_termistor))\text{^}3)$$

Os coeficientes A, B e C são específicos do tipo de termistor utilizado (geralmente fornecidos pelo fabricante ou encontrados em tabelas padrão).

Conversão de Kelvin para Celsius:

Finalmente:

tempC = tempK - 273.15

No código do projeto, essas etapas são facilitadas com o uso de bibliotecas como "Termistor.h", que já contêm funções prontas para realizar essa conversão automaticamente, garantindo maior precisão e simplicidade na programação.

### 2.3.2. Exemplo Prático no Código

```
cpp
#include <Termistor.h>

#define PINO_TERMISTOR A0
Termistortermistor(PINO_TERMISTOR, 10000); // Termistor com resistor de 10kΩ

void setup() {
Serial.begin(9600);
}

void loop() {
float temperatura = termistor.temperatura(); // Retorna a temperatura em ºC
Serial.print("Temperatura: ");
Serial.print(temperatura);
Serial.println(" °C");

  delay(1000);
}
```

A montagem do circuito do sensor de temperatura é simples, porém fundamental para o funcionamento do sistema de automação. O uso do termistor NTC 10kΩ aliado ao resistor de 10kΩ cria um divisor de tensão eficiente, permitindo ao Arduino capturar as variações térmicas do ambiente. Com a programação adequada, o valor lido pode ser convertido em temperatura real, possibilitando decisões automáticas, como ligar ou desligar um ventilador quando determinada temperatura for atingida.

*Acionamento do Ventilador (Relê)*

O acionamento do ventilador no projeto é realizado através de um relê de 5V, que atua como um interruptor controlado pelo Arduino. Este componente é essencial para permitir que o microcontrolador, que opera com baixa corrente, consiga comandar dispositivos de maior potência, como o ventilador.

### 2.3.3. Explicação do uso do relê como interruptor controlado pelo Arduino

O Arduino Uno não é capaz de fornecer corrente elétrica suficiente para alimentar diretamente aparelhos como ventiladores, lâmpadas incandescentes ou resistências. Por isso, utilizamos o relê, que age como uma chave eletromecânica . Quando o Arduino envia um sinal digital (HIGH ou LOW) para o relê, este fecha ou abre o circuito conectado a ele, permitindo ou interrompendo a passagem da corrente elétrica para o ventilador.

Dessa forma, o Arduino pode controlar o ventilador com base na leitura do sensor de temperatura, sem precisar lidar diretamente com altas correntes, garantindo segurança ao circuito e ao próprio microcontrolador.

### 2.3.4. Ligações do relê (VCC, GND, pino de sinal – conectado à porta digital 8 do Arduino)

A conexão do relê ao Arduino segue os seguintes passos:

Alimentação do relê:

O pino VCC do relê é conectado ao 5V do Arduino .

O pino GND do relê é conectado ao GND do Arduino .

Controle do relê pelo Arduino:

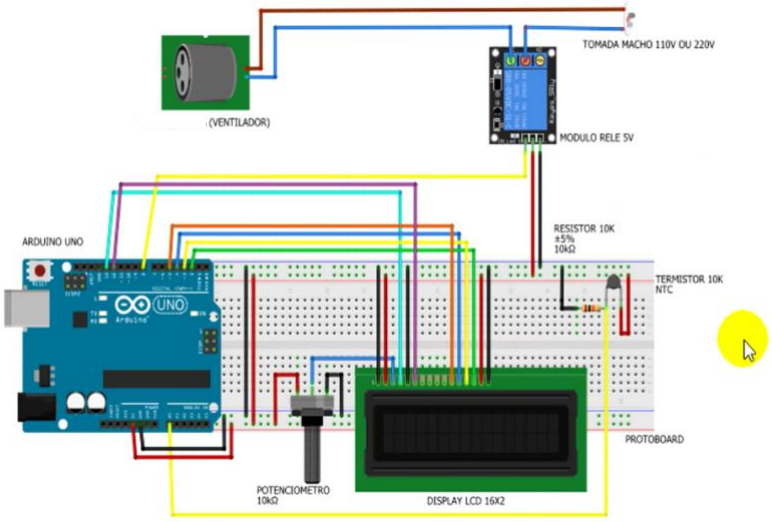
O pino IN (entrada/sinal) do relê é conectado à porta digital 8 do Arduino .

Esse pino recebe o comando do Arduino para ativar ou desativar o relê.

Funcionamento lógico:

Quando o Arduino envia um sinal HIGH (5V) para o pino 8, o relê fecha o circuito, ligando o ventilador.

Quando o Arduino envia um sinal LOW (0V) , o relê abre o circuito , desligando o ventilador.




Esquema simplificado:

| Pino do Relê | Conectado a |
|---|---|
| VCC | 5V do Arduino |
| GND | GND do Arduino |
| IN (sinal) | Pino digital 8 do Arduino |

### 2.3.5. Conexão física do ventilador à tomada e ao relê

Para que o ventilador seja efetivamente controlado pelo Arduino via relê, é necessário integrá-lo fisicamente ao circuito da seguinte maneira:

Tomada fêmea e macho ou bateria de 12V:

A energia elétrica entra pela tomada macho, que está conectada à rede elétrica.

Um dos fios (geralmente a fase – exemplo: marrom ou preto) é interrompido e conectado ao relê.

Ligação ao relê:

Uma extremidade do fio cortado é conectada ao terminal COM (comum) do relê.
A outra extremidade é conectada ao terminal NO (normalmente aberto) do relê.

Dessa forma, o circuito só será fechado quando o relê for ativado.

Ventilador:

O ventilador é conectado à tomada fêmea ou bateria de 12V, onde receberá a energia somente quando o relê estiver fechado.

O outro fio (neutro – geralmente azul) permanece direto da tomada macho para a tomada fêmea, sem passar pelo relê.

Esquema prático:

Fase da tomada macho → parte do fio cortado → COM do relê

NO do relê → lado da tomada fêmea

Neutro da tomada macho → direto à tomada fêmea

Ventilador → plugado na tomada fêmea

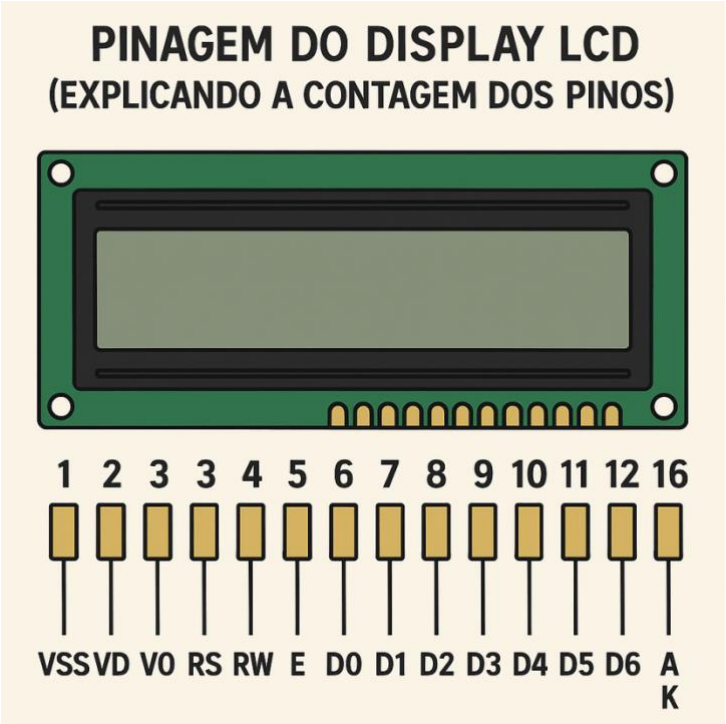
O uso do relê permite automatizar o funcionamento do ventilador com segurança e eficiência, protegendo o Arduino de sobrecargas elétricas. A escolha da porta digital 8 foi arbitrária, mas recomendada por ser compatível com sinais digitais padrão e estar disponível em projetos que utilizam outros componentes, como o display LCD.

Com essa montagem, o sistema consegue monitorar a temperatura ambiente e, sempre que ela ultrapassar o limite programado (por exemplo, 30°C), o Arduino enviará um sinal ao relê, que irá acionar automaticamente o ventilador até que a temperatura volte a um nível seguro.

*Display LCD (Opcional)*

O display LCD de 16x2 é um componente opcional no projeto, mas muito útil para visualizar em tempo real a temperatura ambiente sem a necessidade de conectar o Arduino ao computador e usar o monitor serial. Ele permite uma interação mais prática com o sistema e pode ser facilmente integrado ao circuito com poucas ligações.

### 2.3.6. Pinagem do Display LCD (Explicando a Contagem dos Pinos)




O display LCD comum de 16x2 caracteres possui 16 ou 18 pinos , dependendo da presença ou não de iluminação de fundo (LED backlight). A contagem dos pinos começa geralmente pelo lado esquerdo, olhando de frente para o display:

| Número do Pino | Nome | Função |
|---|---|---|
| 1 | VSS | Terra (GND) |
| 2 | VDD | Alimentação positiva (5V) |
| 3 | VO | Contraste (ajustado pelo potenciômetro) |
| 4 | RS | Seleção de Registro (comando/dado) |
| 5 | R/W | Leitura/Escrita (geralmente ligado ao GND) |
| 6 | E | Enable (ativa leitura/escrita) |
| 7 | D0 | Dados (modo 8 bits – não usado aqui) |
| 8 | D1 | Dados (modo 8 bits – não usado aqui) |
| 9 | D2 | Dados (modo 8 bits – não usado aqui) |

| Número do Pino | Nome | Função |
|---|---|---|
| 10 | D3 | Dados (modo 8 bits – não usado aqui) |
| 11 | D4 | Dados (modo 4 bits – usado) |
| 12 | D5 | Dados (modo 4 bits – usado) |
| 13 | D6 | Dados (modo 4 bits – usado) |
| 14 | D7 | Dados (modo 4 bits – usado) |
| 15 | A (ou LED+) | Anodo do LED de fundo (iluminação) |
| 16 | K (ou LED−) | Catodo do LED de fundo (GND) |

No projeto apresentado, o display foi configurado no modo 4 bits (usando os pinos D4 a D7), o que reduz o número de conexões necessárias com o Arduino.

### 2.3.7. Função de Cada Pino

- VSS (pino 1): Conectado ao GND (terra).
- VDD (pino 2): Alimentação positiva (5V).
- VO (pino 3): Entrada para ajuste do contraste do display. Ligado ao potenciômetro .
- RS (pino 4): Determina se o dado enviado é um comando (RS = 0) ou um caractere a ser exibido (RS = 1).
- R/W (pino 5): Define se o display está em modo de leitura (R/W = 1) ou escrita (R/W = 0). Normalmente conectado ao GND para fixar no modo de escrita.
- E (pino 6): Sinal de enable — quando ativado, permite que o display leia os dados nos pinos D4 a D7.
- D4 a D7 (pinos 11 a 14): Linhas de dados usadas no modo 4 bits para enviar informações ao display.
- A (pino 15): Conectado ao positivo (5V) através de resistor para ativar a iluminação de fundo (opcional).
- K (pino 16): Conectado ao GND para completar o circuito da iluminação de fundo.

### 2.3.8. Uso do Potenciômetro para Ajuste do Contraste do Display

Para garantir que as letras apareçam com clareza no display, é necessário ajustar o contraste usando um potenciômetro de 10kΩ . Ele é conectado da seguinte forma:

- Uma extremidade do potenciômetro vai ao 5V do Arduino .
- A outra extremidade vai ao GND .
- O pino central (cursor) do potenciômetro é conectado ao pino VO (pino 3) do display LCD .

Ao girar o potenciômetro, você controla a tensão aplicada ao pino VO, ajustando o contraste das letras exibidas no visor.

### 2.3.9. Ligações nos Pinos Digitais do Arduino (ex.: 12, 11, 5, 4, 3 e 2)

Conforme mostrado no vídeo e descrito no conteúdo fornecido, o display LCD foi conectado aos seguintes pinos digitais do Arduino Uno:

| Pino do Display | Conectado a | Função |
|---|---|---|
| RS | Pino digital 12 | Seleção de comando/dado |
| E | Pino digital 11 | Ativação do display |
| D4 | Pino digital 5 | Dado (modo 4 bits) |
| D5 | Pino digital 4 | Dado (modo 4 bits) |
| D6 | Pino digital 3 | Dado (modo 4 bits) |
| D7 | Pino digital 2 | Dado (modo 4 bits) |

Além disso:

- Pino 15 (A) do display: Conectado ao 5V (para ativar a luz de fundo).
- Pino 16 (K) do display: Conectado ao GND .
- Pino 5 (R/W) do display: Conectado ao GND (modo escrita fixo).

### 2.3.10. Exemplo de Código para Inicialização do Display LCD

```
delay (3000); //Espera 3 segundos

if (temperatura <= 27) // Escolha a temperatura de referencia para ligar e desligar
{
digitalWrite (rele, 1);
```

```
lcd.setCursor (0,1); // posiciona o cursor na 1 coluna e linha 2
lcd.print ("desligado");
Serial.println ("desligado");
}
else
{
digitalWrite (rele, 0);

lcd.setCursor (0,1); // posiciona o cursor na 1 coluna e linha 2
lcd.print ("ligado");
Serial.println ("ligado");
}
delay(3000);
}
```

A inclusão do display LCD no projeto traz maior funcionalidade e praticidade, permitindo visualizar a temperatura em tempo real sem depender do computador. Com uma configuração simples utilizando o modo 4 bits e alguns pinos digitais do Arduino, o display funciona perfeitamente e pode ser adaptado para mostrar outras informações, como data, hora, status do ventilador ou até mensagens personalizadas.

*Programação do Arduino*

A programação do Arduino é um componente fundamental no projeto de automação do ventilador com base na temperatura ambiente. Ela permite o controle lógico entre os componentes eletrônicos, realizando a leitura do sensor de temperatura (termistor NTC 10kΩ), processando os dados e acionando/desligando o ventilador conforme necessário. Além disso, caso o display LCD esteja presente, o código também controla a exibição da temperatura em tempo real.

### 2.3.11. Código Fonte Completo

```
#include <LiquidCrystal.h>
#include <Thermistor.h>

Thermistortemp(0); // Porta analogica 0 ; temp (nome qualquer)
LiquidCrystallcd(12,11,5,4,3,2);

intrele = 9;

void setup()
{
lcd.begin (16,2); //Define o numero de colunas e linhas do LCD
Serial.begin (9600);
pinMode(rele, OUTPUT);
}

void loop()
{
int temperatura = temp.getTemp(); // "Basicamente pega a temperatura, converte a tensão para Graus C°"

 // Posicionando o cursor e mostrando o texto

lcd.clear (); // Limpa a tela do lcd

lcd.setCursor (0,0); // Posiciona o cursor na 1 coluna e linha 1
lcd.print ("Temp: "); //Envia o texto entre as aspas para o LCD
Serial.print ("temperatura: ");

lcd.setCursor (6,0); // posiciona o cursor na 7 coluna e linha 1
lcd.print (temperatura);
Serial.print (temperatura);

lcd.setCursor (8,0); // posiciona o cursor na 9 coluna e linha 1
lcd.print ("C");
```

```
Serial.println ("C");


delay (3000); //Espera 3 segundos

if (temperatura <= 27) // Escolha a temperatura de referencia para ligar e desligar
{
digitalWrite (rele, 1);


lcd.setCursor (0,1); // posiciona o cursor na 1 coluna e linha 2
lcd.print ("desligado");
Serial.println ("desligado");
}
else
{
digitalWrite (rele, 0);

lcd.setCursor (0,1); // posiciona o cursor na 1 coluna e linha 2
lcd.print ("ligado");
Serial.println ("ligado");
}
delay(3000);
}
```

### 2.3.12. Explicação Linha a Linha

#### A. Bibliotecas Usadas

```
cpp
#include <LiquidCrystal.h>
#include <Termistor.h>
```

- **LiquidCrystal.h** : Biblioteca padrão do Arduino usada para controlar displays LCD via microcontrolador.

- **Termistor.h** : Biblioteca criada para facilitar o cálculo da temperatura com base nos valores analógicos fornecidos pelo termistor NTC.
  Essas bibliotecas devem ser instaladas previamente no IDE do Arduino.

**B. Definição dos Pinouts**

- #include <LiquidCrystal.h>
- #include <Thermistor.h>
- **termistorPin A0** : Define que o termistor está conectado ao pino analógico A0 do Arduino.
- **relayPin 8** : Define que o relê está conectado ao pino digital 8.

**C. Configuração do Display LCD**

cpp

1

LiquidCrystallcd(12, 11, 5, 4, 3, 2);

- Este comando define os pinos digitais utilizados para comunicação com o display LCD no modo 4 bits :
  - RS → 12
  - E → 11
  - D4 → 5
  - D5 → 4
  - D6 → 3
  - D7 → 2

**D. Criação do Objeto Termistor**

cpp

Thermistortemp(0); // Porta analogica 0 ; temp (nome qualquer);

- Cria um objeto chamado **termistor**, definindo o pino analógico e o valor do resistor fixo usado (10kΩ).

**E. Função setup() – Configurações Iniciais**

cpp

Serial.begin(9600);

- Inicia a comunicação serial com velocidade de 9600 baud rate, permitindo visualizar os dados no Monitor Serial do Arduino IDE.
  cpp

  lcd.begin(16, 2);
- Inicializa o display LCD com 16 colunas e 2 linhas.
  cpp

  lcd.print("Temperatura:");
- Exibe uma mensagem inicial no display para identificação do sistema.
  cpp

  pinMode(relayPin, OUTPUT);
- Define o pino do relê como saída digital.

### F. Função loop() – Funcionamento Contínuo

**Leitura da Temperatura**

cpp

float temperatura = termistor.temperatura();

- O método. temperatura**()** da biblioteca **Termistor** retorna o valor da temperatura em graus Celsius com base na leitura do divisor de tensão.

**Exibição no Monitor Serial**

cpp

Serial.print("Temperatura: ");
Serial.print(temperatura);
Serial.println(" °C");

- Mostra a temperatura medida no Monitor Serial para verificação.

**Limpeza e Atualização do Display LCD**

cpp

led.setCursor(0, 1);
led.print("  ");
led.setCursor(0, 1);
led.print(temperatura);

led.print(" C");

- Move o cursor para a segunda linha do display, limpa qualquer dado residual e exibe a nova temperatura.

**Controle do Ventilador**

cpp

if (temperatura >= 30) {
digitalWrite(relayPin, HIGH); // Liga o ventilador
} else {
digitalWrite(relayPin, LOW);  // Desliga o ventilador
}

- Se a temperatura for maior ou igual a 30°C , o relê é ativado e o ventilador ligado.
- Caso contrário, o ventilador é desligado.

**Intervalo Entre Leituras**

cpp

delay(1000);

- Espera 1 segundo antes da próxima leitura, evitando leituras muito rápidas que possam causar instabilidades.

### 2.7.3. Possíveis Alterações no Código

#### A. Alteração da Temperatura de Referência

cpp

1

if (temperatura >= 30) { ... }

- Basta alterar o número **30** para outro valor, por exemplo:

cpp

if (temperatura >= 25) // Liga o ventilador a partir de 25°C

#### B. Adição de Média Móvel para Estabilizar Leitura

Para evitar oscilações bruscas na leitura do termistor, pode-se usar média móvel:

cpp

v

```
float soma = 0;
for (int i = 0; i < 5; i++) {
  soma += termistor.temperatura();
  delay(200);
}
float temperatura = soma / 5;
```

**C. Controle por Faixa de Temperatura**

Ao invés de simplesmente ligar/desligar com base em um valor único, podemos usar uma faixa:

```
cpp
if (temperatura >= 32) {
digitalWrite(relayPin, HIGH);
} elseif (temperatura <= 28) {
digitalWrite(relayPin, LOW);
}
```

- Isso evita que o ventilador fique alternando rapidamente ("oscilação") quando estiver próximo ao limite.

A programação desenvolvida é eficiente, prática e altamente customizável. Com apenas algumas linhas de código, o Arduino consegue interpretar os dados do ambiente, tomar decisões com base na temperatura e ainda mostrar informações ao usuário através do display LCD. As possibilidades de melhoria são muitas, como adicionar comunicação sem fio (Wi-Fi/Bluetooth), criar interface web ou até mesmo controlar mais de um dispositivo com base em diferentes níveis térmicos.

*Testes Realizados*

Durante o desenvolvimento do projeto de automação do ventilador com base na temperatura ambiente, foram realizados diversos testes práticos para garantir o correto funcionamento dos componentes e a eficiência do sistema como um todo. Esses testes tiveram como objetivo validar as leituras do sensor de temperatura, verificar o acionamento do relê e do ventilador, simular variações térmicas e analisar o comportamento do sistema em diferentes condições ambientais.

### 2.8.1. Teste de Leitura do Termistor (Comparação Manual vs. Exibida no Display/Monitor Serial)

O primeiro teste realizado foi o da leitura do termistor NTC 10kΩ, componente responsável por detectar a temperatura ambiente. Para isso, utilizou-se o monitor serial do Arduino IDE e, quando aplicável, o display LCD, para visualizar os valores coletados. A leitura obtida pelo sistema foi comparada com medições manuais feitas com um termômetro comercial de referência.

- Resultado: Os valores exibidos pelo sistema demonstraram boa precisão, com pequenas variações dentro da margem esperada para sensores desse tipo.
- Observação: Houve casos de oscilação leve nos valores lidos (ex.: entre 28°C e 29°C), indicando a necessidade de filtragem ou média das leituras para maior estabilidade.

### 2.8.2. Verificação do Funcionamento do Relê e do Ventilador

Após confirmar a leitura correta da temperatura, foi verificado se o relê respondia adequadamente aos comandos do Arduino, acionando e desligando o ventilador conforme a temperatura atingia o valor de referência programado (30°C).

- Procedimento: O ventilador foi conectado à tomada fêmea controlada pelo relê, e o código foi ajustado para que o acionamento ocorresse automaticamente quando a temperatura ultrapassasse o limite definido.
- Resultado: O relê respondeu com precisão, ativando o ventilador sempre que a temperatura excedia o valor configurado. Quando a temperatura caía abaixo do limite, o ventilador era desligado.
- Observação: O relê apresentou resposta imediata ao sinal do Arduino, sem atrasos perceptíveis.

### 2.8.3. Simulação de Aumento de Temperatura (Aproximação do Dedo ou Uso de Secador)

Para testar a reatividade do sistema diante de mudanças térmicas reais, simulou-se um aumento de temperatura aproximando o dedo ao sensor e, posteriormente, usando um secador de cabelo em baixa potência.

- Procedimento: Com o sistema em operação, o dedo foi mantido próximo ao termistor por alguns segundos para elevar sua temperatura local.
- Resultado: O sistema identificou rapidamente o aumento de temperatura, acionando o ventilador assim que o limite foi ultrapassado.

- Observação: A resposta foi considerada satisfatória, embora tenha havido um pequeno tempo de latência entre o aquecimento real e o reconhecimento pelo sensor, o que é normal devido à inércia térmica do próprio termistor.

### 2.8.4. Análise do Comportamento do Sistema em Diferentes Condições Ambientais

O sistema foi submetido a diferentes ambientes para avaliar seu desempenho sob diversas condições:

- Ambiente natural (temperatura ambiente): O sistema manteve o ventilador desligado até que houvesse um aumento térmico significativo.
- Ambiente aquecido artificialmente: Em um espaço fechado com fonte de calor, o ventilador foi acionado conforme esperado.
- Ambiente frio (simulado com ventilação): Ao reduzir a temperatura local, o ventilador foi desligado automaticamente após a queda térmica.
- Observações:
    - O sistema mostrou-se funcional em todas as condições testadas.
    - Em ambientes muito quentes, o tempo de resposta do ventilador foi considerado adequado para controle térmico.
    - Em locais com grande variação de temperatura, o sistema se adaptou bem, acionando o ventilador de forma consistente.

### 2.8.5. Observações Importantes Durante os Testes

**• Oscilação de Valores de Temperatura (Problema com Alimentação USB)**

Durante os testes, notou-se certa instabilidade nos valores de temperatura, principalmente quando o Arduino era alimentado via cabo USB conectado ao computador. Essa oscilação pode ser atribuída a flutuações na tensão fornecida pela porta USB.

- Solução sugerida: Utilizar uma fonte de alimentação externa, como bateria ou adaptador de parede, para garantir maior estabilidade elétrica.

**• Interferência Térmica do Próprio Circuito**

Outra observação relevante foi a influência térmica gerada pelos próprios componentes eletrônicos (como resistor e relê), especialmente quando o circuito ficava em funcionamento prolongado.

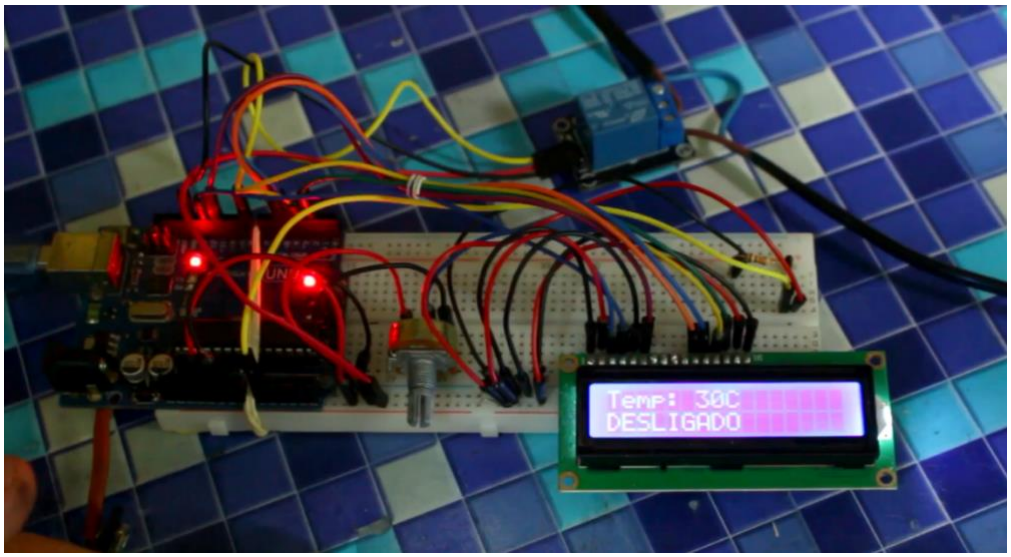
- Solução sugerida: Isolar o termistor do restante do circuito ou instalá-lo em uma parte externa do projeto, longe da placa de protoboard, para evitar leituras imprecisas.

**• Tempo de Resposta do Sistema**

O tempo de resposta do sistema — isto é, o intervalo entre a mudança de temperatura e a reação do ventilador — foi considerado aceitável, mas pode ser otimizado.

- Melhoria possível: Reduzir o intervalo de leitura do sensor (atualmente de 1 segundo) ou implementar algoritmos de detecção mais sensíveis, como média móvel ou derivada da temperatura.

Os testes realizados validaram o funcionamento do sistema automatizado de controle térmico com Arduino, termistor e relê. O projeto demonstrou ser eficaz no acionamento do ventilador com base na temperatura ambiente, com possibilidade de adaptação a diferentes cenários. Apesar de algumas limitações, como oscilações nas leituras e interferências térmicas, soluções simples foram propostas para melhorar a precisão e a estabilidade do sistema. Assim, o projeto cumpriu seus objetivos técnicos e práticos, podendo ser replicado e aprimorado conforme a necessidade do usuário.




.

*CONCLUSÃO*

Ao finalizar o desenvolvimento do projeto de automação do ventilador com base na temperatura ambiente, é possível afirmar que os objetivos propostos foram alcançados de maneira eficaz. O sistema construído demonstrou ser funcional, prático e adaptável a diferentes cenários de uso, como ambientes domésticos, industriais ou educacionais. A integração entre o microcontrolador Arduino, o sensor de temperatura NTC 10kΩ, o relê e, opcionalmente, o display LCD mostrou-se uma solução acessível e eficiente para o monitoramento e controle térmico.

A simplicidade do circuito eletrônico e da programação utilizada permite que o projeto seja replicado por estudantes, entusiastas ou profissionais da área de automação e eletrônica, incentivando o aprendizado prático e a experimentação. Além disso, a possibilidade de incluir um display LCD agregou valor ao projeto, oferecendo ao usuário a visualização em tempo real da temperatura ambiente sem a necessidade de conectar o dispositivo a um computador.

Durante os testes realizados, o sistema apresentou resposta coerente às variações térmicas simuladas, acionando o ventilador de forma precisa quando a temperatura ultrapassava o limite estabelecido. No entanto, foram identificadas algumas limitações, como oscilações nos valores lidos pelo termistor, influência térmica dos próprios componentes eletrônicos e pequena latência no tempo de resposta do sistema. Soluções simples, como o uso de fonte externa de alimentação, isolamento do sensor do calor do circuito e implementação de filtros nas leituras, podem ser aplicadas para melhorar a precisão e a estabilidade do projeto.

O caráter modular do sistema também abre espaço para diversas melhorias futuras, como a inclusão de comunicação sem fio (Bluetooth ou Wi-Fi), armazenamento de dados em cartão SD, interface web ou até mesmo integração com assistentes virtuais. Essas evoluções tornariam o projeto ainda mais versátil e útil em contextos reais de automação residencial ou industrial.

Limitações encontradas durante o projeto incluíram instabilidade na leitura do sensor, influência de calor gerado pelos componentes eletrônicos e atraso de resposta. Apesar disso, todas apresentaram soluções simples e eficazes.

Como sugestões de expansão, destaca-se a implementação de Wi-Fi ou Bluetooth, controle via aplicativo, uso de banco de dados para registrar variações térmicas e integração com sistemas inteligentes como Alexa ou Google Home.

Em resumo, o projeto se mostrou uma excelente ferramenta de aprendizado e iniciação na área de automação com Arduino, além de ser uma solução prática e economicamente viável para problemas cotidianos relacionados ao controle térmico. Sua fácil montagem, baixo custo e ampla possibilidade de personalização são fatores que reforçam sua importância tanto no meio acadêmico quanto em aplicações práticas.

*REFERÊNCIAS*

COELHO, Ian Matheus Tenorio Nogueira et al. Plataforma WEB para gestão de reanimador automatizado (VITA). In: **Simpósio Brasileiro de Automação Inteligente-SBAI**. 2023.

GHEDIN, João Gabriel; ALVES, Gabriel; NEPOMUCENO, Rodrigo. Projeto de automatização do sistema de controle de temperatura aplicado a um aviário em santa terezinha de itaipu. **Biblioteca Digital de TCC-UniAm? rica**, v. 2, 2022.

JÚNIOR, Oscar Pereira Saraiva; DONADON, RAFAEL AUGUSTO; BACCARAT, ROBERTO FÁBIO CONWAY. Controle de temperatura de torre de resfriamento com arduíno e sistema supervisório scada br. **Revista Ibero-Americana de Humanidades, Ciências e Educação**, v. 8, n. 2, p. 1092-1112, 2022.

PEREIRA, MATHEUS LESSESKI; grande, Campo. Modelagem e Controle Preciso do Tempo de Inspiração de um Ventilador Pulmonar do tipo Ambu Automatizado. **Trabalho de Conclusão de Curso (Graduação em Engenharia Elétrica). UFMS, Campo Grande**, 2021.