



Universidade do Estado do Rio de Janeiro

Centro de Tecnologia e Ciências

Faculdade de Engenharia

Departamento de Sistemas e Computação

Controle de Processos por Computador

Turma 1

Philippe Macêdo Borges Ramos

Trabalho Final

Parte 1

Professor: Luigi Maciel Ribeiro

Data de Envio: 28/11/2025

Rio de Janeiro

2025

1 Funcionalidade

A funcionalidade do sistema foi projetada para o gerenciamento térmico de um servidor de processamento de dados com IoT, garantindo a integridade do hardware através da automatização do controle térmico e monitoramento contínuo.

1.1 Descrição do Sistema

O projeto consiste no desenvolvimento de um sistema de controle térmico para servidores em malha fechada focada especificamente na regulação de temperatura dos processadores (CPUs). O microcontrolador usado é o ESP32, que é responsável pela aquisição de sinais e execução do algoritmo de controle.

A interação do sistema com o ambiente ocorre pelos seguintes componentes:

- **Variável Analógica:** A temperatura do processador é monitorada por um sensor digital, e em seguida este sensor digitaliza o sinal térmico e comunica o valor em graus Celsius ao ESP32 permitindo leituras contínuas.
- **Atuação Analógica:** O controle de resfriamento é realizado por uma ventoinha acionada via sinal PWM. O algoritmo de controle PID ajusta a velocidade de rotação por meio da largura de pulso, de modo que mantenha a temperatura no *Setpoint* desejado.
- **Entradas Digitais:** As entradas digitais do sistema são:
 1. Sensor de Intrusão (Chave): Responsável por monitorar a abertura do gabinete do servidor, o que comprometeria a eficiência do fluxo de ar;
 2. Sensor de Vibração (Botão Simulado): Responsável por detectar vibrações no equipamento.
 3. Comando Local (Botão de Turbo): Botão físico que permite o operador realizar um acionamento manual de resfriamento máximo.
- **Saídas Digitais:** As saídas digitais do sistema são:
 1. Relé de Segurança: Atua cortando a alimentação do servidor caso a temperatura atinja níveis destrutivos;
 2. Sinalização Visual: LEDs indicadores de status para operação normal (Verde), falha/alerta (Vermelho) e ventoinha (Azul).

1.2 Fluxo de Operações

O fluxo de dados está dividido em controle local e supervisão remota:

- **Aquisição Digital:** O ESP32 envia um comando de solicitação ao sensor. O sensor processa a temperatura e retorna um valor digital preciso, e junto disso, o controlador lê os estados lógicos dos botões e chaves de segurança.
- **Controle PID:** O firmware calcula o erro do sistema e com base nos parâmetros Proporcional, Integral e Derivativo, o algoritmo determina a correção necessária na velocidade da ventoinha.

- **Segurança e Lógica Combinacional:** Antes de aplicar o sinal à ventoinha, o sistema realiza uma verificação nas restrições de segurança. Caso o sensor de intrusão indicar abertura, um alerta é gerado e caso a temperatura ultrapassar o limite crítico, o relé de segurança é acionado desligando a carga.
- **Comunicação IoT:** Periodicamente os dados de temperatura, Duty Cycle e status dos alarmes, são enviados em um pacote JSON.
- **Supervisão:** O pacote JSON é enviado via requisição HTTP POST para o servidor backend em DJANGO (hospedado no PythonAnywhere), que armazena o histórico e atualiza o *Dashboard*.

1.3 Interface do Usuário

A interação com o usuário ocorre através de uma aplicação web com as principais funcionalidades:

- **Monitoramento:** Visualização de gráficos de temperatura e velocidade da ventoinha.
- **Painel de Alarmes:** Indicando visualmente as violações de segurança (exemplo: "Gabinete Aberto")
- **Controle Setpoint:** Formulário para alteração do Setpoint da temperatura desejada via API, permitindo ajuste remotamente.

1.4 Integração com Outros Sistemas

Para integração com outros sistemas, é utilizada a arquitetura RESTful API. O ESP32 envia os dados ao servidor Django separando a lógica de controle crítica da lógica de supervisão e isso permite que o sistema continue operando e protegendo o hardware mesmo se a conexão com a internet for perdida.

2 Desempenho

Considerando as características físicas do sensor digital e a inércia térmica do processo, o desempenho do sistema foi dimensionado.

2.1 Requisitos de Tempo de Resposta

- **Controle Local:** Considerando o tempo de conversão interna do sensor utilizado, o período de amostragem do controle PID foi fixado em 1 segundo. Pois, como sistemas térmicos apresentam dinâmica lenta, este tempo garante estabilidade sem sobrecarregar o barramento de dados.
- **Latência de Atuação de Segurança:** O relé opera com prioridade máxima, ou seja, quando uma temperatura crítica é lida, o corte de energia ocorre imediatamente independente do cálculo do PID.
- **Latência de Telemetria:** A atualização do painel web ocorre a cada 15 segundos para que as cotas de requisição da hospedagem sejam respeitadas e evite congestionamento de rede.

2.2 Capacidade de Processamento

O sistema utiliza um dos Core do ESP32 para gerenciamento do protocolo 1-wire, leitura de portas e cálculo do PID, enquanto utiliza outro Core para gerenciar a pilha TCP/IP e criptografia SSL para que o envio dos dados para a nuvem não bloqueie o loop de controle da temperatura.

2.3 Eficiência Energética

Para a otimização energética, o sistema utiliza o controle feito pelo PID. Pois, diferente de sistemas de refrigeração que usam o algoritmo "liga-desliga" para o controle, no qual, operam com a ventoinha sempre em 0% ou 100%, o controle feito pelo algoritmo PID modula a potência necessária através do PWM, ou seja, na maior parte do tempo a ventoinha estará operando em regimes intermediários reduzindo o consumo elétrico do atuador.

2.4 Escalabilidade

O sistema foi projetado para permitir o crescimento sem a necessidade de refatoração do código. Com a arquitetura RESTful no servidor Django, novos controladores ESP32 podem ser adicionados à rede bastando configurar uma ID de dispositivo único no firmware.

3 Segurança

Visando a proteção dos equipamentos, foram pensadas nas seguintes seguranças.

3.1 Segurança Física

- **Relé de proteção:** O acionamento do relé de emergência é configurado para cortar fisicamente a energia do sistema monitorado caso o sistema de controle falhe na contenção da temperatura e ela atinja o limite crítico.
- **Watchdog Timer (WTD):** O temporizador de vigilância do firmware reinicia o microcontrolador caso o sistema trave por mais de 3 segundos, garantindo recuperação autônoma.

3.2 Segurança de Dados

- **Autenticação de API:** A comunicação entre o Django e o ESP32 é protegida por *Tokens* de autenticação no cabeçalho das requisições HTTP, impedindo que dados falsos sejam enviados por terceiros.
- **Validação:** O ESP32 ignora comandos remotos que coloquem o sistema em risco.

3.3 Segurança Operacional

- **Modo de Falha Segura (*Fail-Safe*):** Caso o sensor digital pare de responder, o sistema detecta a ausência de comunicação no barramento 1-Wire e entra em modo de emergência acionando a ventoinha em 100% da potência e o operador é notificado do erro via LED.
- **Prevenção:** A integração do sensor de vibração permite desligar componentes ao detectar queda ou impacto.

3.4 Gestão de risco

A gestão de riscos é importante para que os pontos críticos de vulnerabilidade sejam identificados e contramedidas sejam implementadas. A Tabela 1 resume os principais riscos identificados e quais as estratégias de mitigação adotadas.

Tabela 1: Matriz de Gestão de Riscos e Mitigação

Riscos	Impacto Potencial	Medida de Mitigação Adotada
Falha do Sensor térmico	O sistema perde a referência de temperatura, podendo causar superaquecimento sem detecção.	Lógica de Fail-Safe: O código verifica se a leitura é válida. Se inválida (NaN ou erro), força a ventoinha para 100% e aciona o LED de erro.
Travamento do Microcontrolador	O controle PID congela, mantendo a ventoinha em velocidade fixa mesmo se a temperatura subir.	Watchdog Timer (WDT): Um temporizador de hardware reinicia o ESP32 automaticamente se o loop principal travar por mais de 3 segundos.
Perda de Conexão Wi-Fi	O sistema perde a supervisão remota e a capacidade de receber comandos do operador.	Operação Autônoma (Local): O controle PID e a segurança (Relé) rodam no ESP32 independentemente da conexão. O sistema reconecta automaticamente.
Ataque de Replay ou Injeção	Um atacante envia dados falsos para o servidor ou comandos maliciosos para o ESP32.	Autenticação via Token: Uso de chaves de API nos cabeçalhos HTTP. O servidor rejeita pacotes sem credenciais válidas.
Falha Mecânica da Ventoinha	O atuador queima ou trava, impedindo a remoção efetiva de calor.	Redundância Passiva (Relé): Se a temperatura subir acima do limite crítico mesmo com comando de 100%, o Relé corta a energia da carga (CPU).