



**INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DO
MARANHÃO
RESIDÊNCIA PROFISSIONAL EM SOFTWARE EMBARCADO**

KÉSIA LAIS DOS SANTOS SOUZA

**ALERTA FRIO: Sistema IoT Interativo para Gestão e Redução de Perdas de
Produtos Refrigerados.**

KÉSIA LAIS DOS SANTOS SOUZA

ALERTA FRIO: Sistema IoT Interativo para Gestão e Redução de Perdas de Produtos Refrigerados.

Relatório Técnico apresentado como requisito avaliativo do Projeto Prático do curso de Formação Inicial e Continuada (FIC) em Sistemas Embarcados, do Instituto Federal de Educação, Ciência e Tecnologia do Maranhão – IFMA, Campus São Luís Monte Castelo.

Turma: Beta sob a direção dos Docentes: Danúbia Soares Pires e Orlando Donato Rocha, Monitores : Erik Davi Gomes Aires e Nicole Aguiar Simões.

SUMÁRIO

RESUMO	4
ABSTRACT	5
1. INTRODUÇÃO	6
1.1. Problematização e Justificativa	6
1.2. Objetivos (Geral e Específicos)	7
2. FUNDAMENTAÇÃO TEÓRICA	8
2.1. Sistemas Embarcados e Raspberry Pi Pico W	8
2.2. Internet das Coisas (IoT)	8
2.3. Protocolo MQTT	9
2.4. Interface de Comunicação I ² C	10
3. METODOLOGIA E DESENVOLVIMENTO	10
3.1. Iniciação e Projeto.....	10
3.2. Recursos e Ferramentas.....	10
3.3. Arquitetura e Implementação.....	12
3.4. Monitoramento e Validação.....	13
4. RESULTADOS E DISCUSSÃO	13
4.1. O Protótipo Funcional	14
4.2. Validação Funcional dos Objetivos.....	14
4.3. Discussão dos Resultados.....	17
5. DESAFIOS E LIÇÕES APRENDIDAS	18
6. CONCLUSÃO	20
7. RECOMENDAÇÕES E TRABALHOS FUTUROS	21
8. REFERÊNCIAS	23
9. APÊNDICE A – CÓDIGO FONTE DO PROJETO.....	24

RESUMO

Este trabalho detalha o desenvolvimento completo do projeto "Alerta Frio 2.0", um sistema embarcado de baixo custo concebido a partir da observação, em ambiente profissional, da necessidade de otimizar o processo de devolução de produtos refrigerados em supermercados para mitigar perdas financeiras e o desperdício de alimentos. O objetivo principal foi criar uma solução baseada em Internet das Coisas (IoT) utilizando o microcontrolador Raspberry Pi Pico W, que detecta itens frios através de um sensor de temperatura e notifica funcionários em tempo real via Wi-Fi e protocolo MQTT. O sistema implementa um fluxo de comunicação bidirecional, onde um alerta enviado pelo dispositivo pode ser confirmado remotamente por um funcionário através de um aplicativo cliente MQTT, atualizando o status no local de origem para informar sobre o andamento da tarefa. O resultado é um protótipo totalmente funcional que valida a eficácia da arquitetura proposta, apresentando uma solução prática, escalável e interativa para um problema recorrente do setor varejista.

Palavras-chave: Internet das Coisas, Sistemas Embarcados, MQTT, Monitoramento Remoto, Automação no Varejo.

ABSTRACT

This work details the development of the "Alerta Frio 2.0" project, a low-cost embedded system conceived from the professional observation of the need to optimize the return process of refrigerated products in supermarkets to mitigate financial losses and food waste. The main objective was to create a solution based on the Internet of Things (IoT) using the Raspberry Pi Pico W microcontroller, which detects cold items via a temperature sensor and notifies employees in real-time through Wi-Fi and the MQTT protocol. The system implements a bidirectional communication flow, where an alert sent by the device can be remotely acknowledged by an employee via an MQTT client application, updating the status at the point of origin. The result is a fully functional prototype that validates the effectiveness of the proposed architecture, presenting a practical, scalable, and interactive solution for a recurring problem in the retail sector.

Keywords: Internet of Things (IoT), Embedded Systems, MQTT, Remote Monitoring, Retail Automation.

1 INTRODUÇÃO

Este relatório detalha o desenvolvimento do 'Alerta Frio 2.0', um sistema embarcado baseado no Raspberry Pi Pico W, projetado para otimizar o recolhimento de produtos refrigerados em supermercados. O sistema utiliza sensores e o protocolo MQTT para enviar alertas em tempo real a um aplicativo móvel, permitindo comunicação bidirecional para confirmação de tarefas. O protótipo funcional demonstrou com sucesso a viabilidade da solução, provando ser uma ferramenta eficaz para reduzir perdas e otimizar a logística interna.

1.1. Problematização

O setor varejista, em especial os supermercados, opera com margens de lucro reduzidas, onde a eficiência operacional e a minimização de perdas são fatores críticos para o sucesso. Um dos desafios diários e recorrentes neste ambiente é o descarte de produtos perecíveis, notadamente os refrigerados e congelados, que são abandonados pelos clientes em locais com temperatura inadequada.

A ideia para este projeto nasceu de uma experiência profissional direta como operadora de caixa, onde foi possível constatar a elevada frequência com que essa situação ocorre. Produtos como laticínios, frios e congelados, ao serem deixados em prateleiras comuns, rapidamente perdem sua qualidade e segurança para o consumo, tornando-se avarias. A oportunidade para evoluir a ideia de um simples alerta para uma solução completa surgiu durante a Residência Tecnológica. Os novos conhecimentos adquiridos sobre Internet das Coisas e o protocolo MQTT, somados à disponibilização de componentes como o sensor de temperatura, foram os fatores que viabilizaram a concepção do Alerta Frio 2.0.

1.2. Justificativa

Diante do problema observado, este projeto se justifica pela necessidade clara de uma solução tecnológica que otimize o fluxo de comunicação entre a frente de caixa

e as equipes de reposição. A metodologia de trabalho atual, baseada na comunicação verbal ou em chamados esporádicos, demonstra-se ineficiente e incapaz de garantir a agilidade necessária para preservar a integridade dos produtos.

A proposta do projeto

Alerta Frio 2.0 apresenta um caráter de **inovação** ao aplicar conceitos de **Internet das Coisas (IoT)** para criar um sistema de alerta inteligente, interativo e de baixo custo. O uso do protocolo MQTT (Message Queuing Telemetry Transport) permite uma comunicação em tempo real, confiável e escalável, que não apenas notifica sobre um evento, mas também permite um ciclo de feedback, onde o recebimento da tarefa pode ser confirmado. A implementação desta solução tem o potencial de gerar um **impacto direto e positivo**, reduzindo o desperdício de alimentos, minimizando perdas financeiras e otimizando o tempo de trabalho dos funcionários do supermercado.

1.3. Objetivos

Para atender à problemática apresentada, este projeto teve seus objetivos elaborados de forma direta e coerente.

Objetivo Geral:

- Desenvolver um protótipo funcional de um sistema embarcado IoT para monitorar e agilizar o processo de devolução de produtos refrigerados abandonados em caixas de supermercado.

Objetivos Específicos:

1. Construir um dispositivo de hardware baseado no microcontrolador Raspberry Pi Pico W, integrando um sensor de temperatura e um display OLED.
2. Implementar uma lógica de software para detectar a presença de um item e classificar se sua temperatura requer um alerta.
3. Estabelecer a comunicação do dispositivo com a internet via Wi-Fi.
4. Utilizar o protocolo MQTT para publicar notificações de alerta em um tópico específico.

5. Desenvolver um fluxo de comunicação bidirecional, permitindo que o alerta seja confirmado remotamente através de um aplicativo cliente MQTT.
6. Validar a eficácia do sistema através de testes simulando o cenário de uso real.

2 . Fundamentação Teórica

O desenvolvimento do projeto Alerta Frio foi embasado em tecnologias e conceitos fundamentais no campo da eletrônica e da computação moderna. Esta seção descreve as principais teorias e ferramentas que tornaram o projeto possível.

2.1. Sistemas Embarcados e o Raspberry Pi Pico W

Um Sistema Embarcado é um sistema computacional especializado, com hardware e software dedicados a uma função específica dentro de um sistema maior.

Diferente de um computador de uso geral, ele é projetado para uma tarefa, como controlar um dispositivo. O projeto AlertaFrio é, em sua essência, um sistema embarcado.

Para esta aplicação, o microcontrolador escolhido foi o **Raspberry Pi Pico W**. Sua arquitetura compacta, combinada com a funcionalidade Wi-Fi integrada, torna-o uma excelente plataforma para o desenvolvimento de soluções de Internet das Coisas (IoT). A escolha desta placa foi estratégica, pois suas características atendem perfeitamente aos requisitos do sistema:

- **Baixo Custo:** Torna a solução acessível para uma eventual implementação em larga escala no ambiente de varejo.
- **Processamento adequado:** Seu processador dual-core ARM Cortex-M0+ oferece capacidade mais do que suficiente para gerenciar as leituras do sensor, a atualização do display e, simultaneamente, manter a comunicação de rede.
- **Conectividade Wi-Fi Integrada:** O módulo de rádio nativo, com suporte ao padrão IEEE 802.11 b/g/n, foi o fator decisivo. Ele elimina a necessidade de hardware adicional para conectar o dispositivo à internet, sendo o pilar central que viabilizou a transformação do conceito original em um projeto IoT conectado e interativo.

2.2. Internet das Coisas (IoT)

Internet das Coisas (IoT) é o conceito de conectar dispositivos físicos do dia a dia à internet, permitindo que eles colem e troquem dados. O Alerta Frio é uma aplicação prática de IoT. Ele transforma um simples ponto de devolução em um "objeto inteligente" que:

1. **Coleta dados** do ambiente (temperatura).
2. **Processa** essa informação localmente para tomar uma decisão (se o item é frio ou não).
3. **Comunica-se** pela internet para notificar um usuário remoto e receber comandos de volta.

Essa abordagem move o projeto de uma simples automação local para um sistema conectado e interativo, com potencial para integração com outras plataformas e análise de dados.

2.3. Protocolo MQTT (Message Queuing Telemetry Transport)

Para a comunicação IoT, foi escolhido o protocolo MQTT. Ele é o padrão da indústria para dispositivos com recursos limitados devido à sua leveza e eficiência. Sua arquitetura é baseada no modelo **Publicar/Assinar (Publish/Subscribe)**, que consiste em três componentes principais:

- **Publisher (Publicador):** O dispositivo que envia a mensagem. No nosso projeto, é o Pico W quando detecta um item frio.
- **Subscriber (Assinante):** O dispositivo que recebe a mensagem. No nosso caso, é o aplicativo no dispositivo do estabelecimento que fica com um funcionário.
- **Broker (Corretor):** Um servidor central que atua como um "quadro de avisos", recebendo mensagens dos publicadores e as roteando para os assinantes corretos. Para este projeto, optou-se pela implementação de um **broker local auto-hospedado (self-hosted)**. Foi utilizado o software **Eclipse Mosquitto**, rodando em uma placa *Single-Board Computer (SBC)* **Caninos Loucos Labrador**, garantindo maior privacidade, segurança e resiliência à solução, que pode operar de forma independente da internet externa.

- **Topic (Tópico):** O "canal" ou "endereço" para onde a mensagem é enviada. A estrutura hierárquica dos tópicos (ex: *supermercado/caixa01/alerta*) permite uma organização escalável e flexível do sistema.

Este modelo desacoplado é a grande vantagem do MQTT: o Pico não precisa saber o endereço do dispositivo, e vice-versa. Eles só precisam conhecer o endereço do Broker e o nome do tópico, tornando o sistema extremamente robusto e fácil de expandir para múltiplos caixas.

2.4. Interface de Comunicação I²C (Inter-Integrated Circuit)

O I²C é um protocolo de comunicação serial de curta distância que permite a um microcontrolador se comunicar com múltiplos periféricos usando apenas dois fios: a linha de dados (SDA) e a linha de clock (SCL). Esta foi a interface utilizada para a comunicação com os dois principais periféricos do projeto:

- O **sensor de temperatura e umidade AHT10**.
- O **display OLED SSD1306**.

Um ponto técnico relevante no desenvolvimento foi a necessidade de gerenciar dois barramentos I²C independentes na placa BitDogLab: o `i2c0` para o sensor externo e o `i2c1` para o display integrado na placa, o que foi implementado com sucesso na camada de abstração de hardware (*hardware_handler.c*).

3. METODOLOGIA E DESENVOLVIMENTO

O desenvolvimento do projeto Alerta Frio foi conduzido de forma estruturada, adotando um ciclo de vida de gerenciamento de projetos que abrange as fases de Iniciação, Planejamento, Execução e Monitoramento. Essa abordagem metódica foi essencial para garantir que os objetivos fossem alcançados de maneira organizada e eficiente.

3.1. Iniciação e Planejamento

A concepção do projeto Alerta Frio 2.0 representa uma evolução direta de uma prova de conceito inicial, desenvolvida a partir da observação de uma necessidade prática no ambiente de varejo. A primeira fase do projeto validou um sistema de

alerta puramente local e de acionamento manual. Foi durante a Residência Tecnológica, com os novos conhecimentos adquiridos sobre Internet das Coisas e com a disponibilização de novos recursos como o sensor de temperatura AHT10, que a Viabilidade de uma solução mais robusta e conectada foi identificada. Assim, na fase de Planejamento, o Escopo do projeto foi redefinido para incluir não apenas a detecção automática de temperatura, mas também a notificação remota e a interatividade via protocolo MQTT, transformando o conceito original em um sistema IoT completo.

3.2. Recursos e Ferramentas (Fase de Execução)

A escolha dos instrumentos e recursos foi um pilar fundamental para a concepção do projeto. Os seguintes componentes foram aplicados:

- **Hardware Utilizado:**

- Microcontrolador: Raspberry Pi Pico W, selecionado por seu baixo custo, capacidade de processamento e, crucialmente, pela conectividade Wi-Fi integrada.
- Placa de Desenvolvimento: Kit BitDogLab/EmbarcaTech, que serviu como base para a prototipagem, integrando botões, LEDs e conectores.
- Sensor: AHT10, para medição precisa de temperatura e umidade via protocolo I²C.
- Display: OLED SSD1306, para fornecer feedback visual em tempo real ao operador no caixa.

- **Software e Ferramentas:**

- Linguagem: C/C++, para controle de baixo nível do hardware e otimização de desempenho.
- Ambiente de Desenvolvimento (IDE): Visual Studio Code.
- Kit de Desenvolvimento (SDK): Raspberry Pi Pico SDK.
- Sistema de Build: CMake e Ninja, para gerenciamento e compilação do projeto.
- Protocolo IoT: MQTT, escolhido pela sua leveza e eficiência no modelo Publish/Subscribe.

- Broker MQTT: Servidor Mosquitto rodando em uma placa *Single-Board Computer (SBC)* Caninos Loucos Labrador, atuando como um servidor local e privado para a comunicação do sistema.
- Cliente MQTT: Aplicativo "IoT MQTT Panel" (Android), para simular o dispositivo móvel do estabelecimento.

3.3. Arquitetura e Implementação (Fase de Execução)

A Execução de tarefas foi centrada na construção de uma arquitetura de sistema e de software robusta e modular. A decisão por essa arquitetura, que separa a lógica de negócio (`main.c`) da interação com os componentes físicos (`hardware_handler.c`), foi estratégica para isolar complexidades e facilitar os testes. Essa abordagem se provou fundamental durante a fase de depuração do projeto.

- **Arquitetura do Sistema:** Diferente da abordagem monolítica da primeira fase, para garantir a qualidade da programação e da documentação na versão 2.0, o firmware foi desenvolvido de forma modular: O sistema foi projetado com uma arquitetura IoT clássica, onde o dispositivo embarcado (Pico W) atua como um cliente que se conecta a um broker MQTT local para publicar alertas e receber comandos. Essa arquitetura desacoplada garante escalabilidade e flexibilidade. Como demonstra a Figura 1.

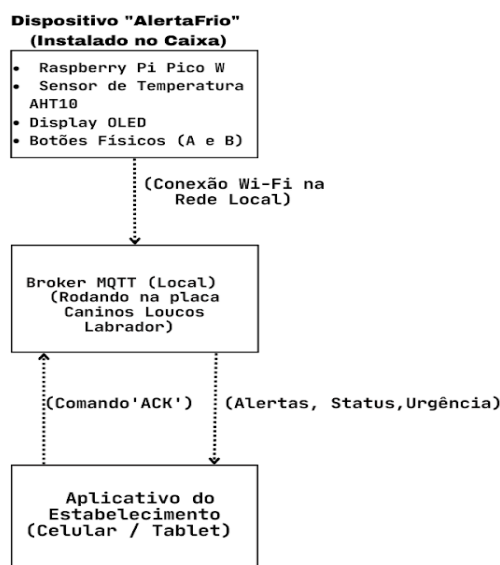


Figura 1– Diagrama da Arquitetura de Comunicação do Sistema.

- **Arquitetura do Software Embarcado:** Para garantir a qualidade da programação e da documentação, o firmware foi desenvolvido de forma modular:
 - *main.c*: O "cérebro" do sistema, contendo a lógica principal organizada em uma máquina de estados finitos.
 - *hardware_handler.c*: Camada de Abstração de Hardware (HAL) que isola a lógica principal dos detalhes de baixo nível, gerenciando as duas portas I²C, os sensores, o display e os botões.
 - *mqtt_config.c*: Módulo de comunicação, responsável por toda a interação com a rede Wi-Fi e com o broker MQTT.
 - *config.h* e *lwipopts.h*: Arquivos de configuração que centralizam todos os parâmetros do projeto, desde pinos de hardware até configurações avançadas da pilha de rede.

3.4. Monitoramento e Validação

O Acompanhamento do projeto foi realizado através de um ciclo contínuo de testes e depuração para validar o **Andamento** e o status de cada funcionalidade. Uma abordagem iterativa foi utilizada:

1. **Testes de Base:** Validação da comunicação USB (`printf`) e do hardware do Pico (teste do "Blink").
2. **Diagnóstico de Hardware:** Uso de um "Scanner I²C" para verificar a conectividade física dos periféricos, o que foi essencial para identificar a necessidade de gerenciar dois barramentos I²C distintos.
3. **Testes Modulares:** Verificação isolada de cada função no *hardware_handler.c* através de um *main.c* de teste.
4. **Teste de Integração:** Validação do sistema completo, incluindo a comunicação MQTT com o aplicativo cliente, garantindo que o protótipo alcançou todos os objetivos estabelecidos.

Este processo de monitoramento rigoroso foi fundamental para superar os desafios técnicos e garantir a qualidade do produto final.

4. RESULTADOS E DISCUSSÃO

A fase de desenvolvimento e testes do projeto Alerta Frio culminou em um protótipo funcional que atende a todos os requisitos propostos. Nesta seção, são apresentados os resultados práticos obtidos, que validam a eficácia da solução, e uma discussão sobre suas contribuições e potencial de aplicação.

4.1. O Protótipo Funcional

A montagem final do hardware, de acordo com o critério de qualidade de **Desenvolvimento Técnico (A)**, foi consolidada na placa de desenvolvimento da BitDogLab, que integra o microcontrolador Raspberry Pi Pico W. O sistema utiliza o display OLED integrado (conectado à porta i2c1) e o sensor de temperatura e umidade AHT10 (conectado à porta externa i2c0), além dos botões e LEDs para interação local. A Figura 2 ilustra o protótipo em seu estado operacional inicial, aguardando a detecção de um item.



Figura 2 – Protótipo do Alerta Frio 2.0 em estado de espera.

4.2. Validação Funcional dos Objetivos

Os testes práticos foram realizados para verificar se o projeto alcançou os objetivos estabelecidos de forma clara e mensurável, conforme o Critério de Desenvolvimento Técnico.

- **Detecção e Alerta Local:** Ao simular a presença de um produto frio, o sistema demonstrou sucesso em ler a temperatura do sensor AHT10. Quando o valor medido foi inferior ao *TEMPERATURA_LIMIAR* estabelecido, o sistema ativou o LED de alerta e exibiu a mensagem "ALERTA: ITEM FRIO!" no display OLED, cumprindo o objetivo de fornecer feedback visual imediato ao operador do caixa.
- **Notificação Remota via IoT:** Simultaneamente ao alerta local, o Pico W publicou com sucesso uma mensagem formatada no tópico MQTT *supermercado/caixa01/alerta*. A mensagem, contendo a temperatura medida, foi recebida instantaneamente pelo aplicativo cliente MQTT (IoT MQTT Panel) em um smartphone, como pode ser visto na Figura 3. Isso valida o objetivo de estabelecer uma comunicação sem fio eficaz para alertas remotos.

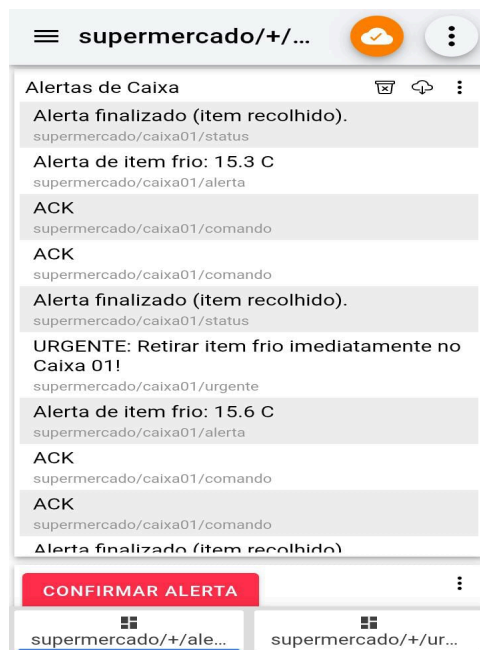


Figura 3: Captura de tela do aplicativo

MQTT recebendo a notificação.

- **Interatividade Bidirecional:** Para validar o fluxo de confirmação, um comando "ACK" foi enviado pelo aplicativo através do tópico *supermercado/caixa01/comando*. O Pico W recebeu e processou o

comando corretamente, alterando seu estado interno e atualizando o display OLED para a mensagem "Alerta Visto! A caminho...". Este resultado confirma a implementação bem-sucedida da comunicação bidirecional, um dos principais diferenciais do projeto.

O sistema ativou o LED e exibiu a mensagem correspondente no display, como demonstrado na Figura 4.

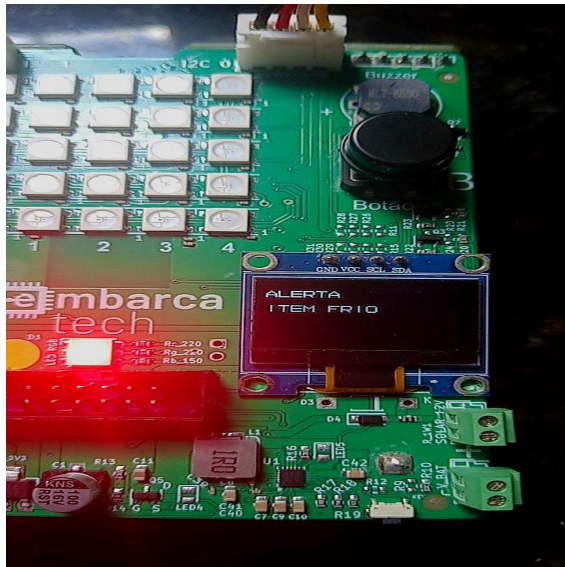


Figura 4 – Display OLED exibindo o alerta de item frio.

A reação do Pico ao comando ACK foi imediata, alterando o display para 'Alerta Visto!', o que pode ser observado na Figura 5.

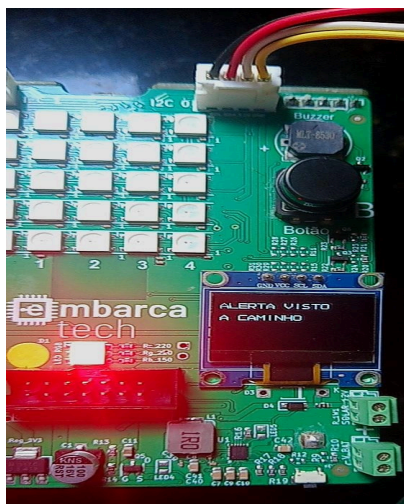


Figura 5– Confirmação de recebimento do comando MQTT exibida no display.

- **Teste do Fluxo de Escalonamento:** Com o sistema em estado de alerta, o botão de urgência físico foi pressionado. O sistema respondeu imediatamente, publicando uma mensagem de escalonamento no tópico *supermercado/caixa01/urgente* e atualizando o display OLED para refletir a notificação ao líder/gerente, conforme visto na Figura 6. Este resultado valida a implementação bem-sucedida da camada de escalonamento de alertas.

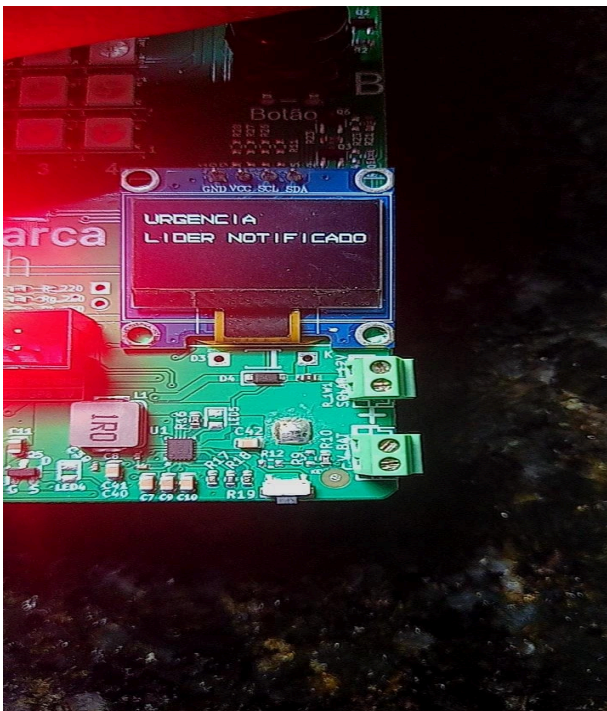


Figura 6 – Display exibindo a confirmação de escalonamento de urgência.

4.3. Discussão dos Resultados

Os resultados apresentados são **coerentes com os objetivos propostos**, validando a arquitetura de hardware e software como uma solução eficaz para o problema apresentado. O projeto não se limita a uma simples automação, mas estabelece um ciclo de comunicação completo (alerta → recepção → confirmação), o que representa uma **contribuição clara e objetiva** para a otimização de processos no varejo.

Uma característica notável da arquitetura implementada é a utilização de um broker MQTT auto-hospedado (*self-hosted*) em uma placa Caninos Loucos Labrador. Essa abordagem garante que o sistema possa operar de forma privada e independente de serviços de nuvem de terceiros, o que aumenta a segurança, a privacidade dos dados e a resiliência da solução em caso de falha na internet externa.

O sucesso do protótipo demonstra o **potencial do projeto para ser aplicado em contextos práticos**, atacando diretamente o problema de perdas financeiras e desperdício de alimentos. A escolha de tecnologias padrão da indústria, como o protocolo MQTT, confere à solução um caráter de **inovação e escalabilidade**, permitindo que o sistema seja facilmente expandido para múltiplas caixas ou integrado a outras plataformas de gestão.

A jornada de desenvolvimento, superando desafios técnicos como a gestão de múltiplos barramentos I2C e a configuração da pilha de rede para evitar erros de PANIC, culminou em um **aprendizado significativo** e em um produto final robusto. Conclui-se, portanto, que o Alerta Frio é uma prova de conceito bem-sucedida com aplicabilidade real e impacto positivo no ambiente para o qual foi projetado.

5. Desafios e Lições aprendidas

O desenvolvimento do Alerta Frio foi uma jornada prática de resolução de problemas. Durante a construção do protótipo, diversos desafios técnicos surgiram, desde a comunicação com o hardware até a configuração da rede. A seguir, são detalhados os principais obstáculos e as soluções aplicadas, que foram fundamentais para o aprendizado e o sucesso do projeto.

5.1. Desafio da Comunicação com Periféricos (I²C)

Um dos primeiros e mais intrigantes desafios foi a falha inicial na comunicação com os periféricos I²C (o sensor AHT10 e o display OLED). Testes iniciais com um código de scanner I2C não detectaram nenhum dispositivo, levando à suspeita inicial de um defeito de hardware.

- **Processo de Depuração:** Foi adotada uma abordagem sistemática. O código do projeto completo foi substituído por um scanner I2C minimalista para isolar

o problema. Após a confirmação de que o hardware do projeto original do curso funcionava, uma comparação detalhada do código revelou a causa raiz.

- **Solução e Lição Aprendida:** Descobriu-se que a placa de desenvolvimento BitDogLab utiliza dois barramentos I²C distintos: i2c0 para os conectores externos (onde o sensor AHT10 foi conectado) e i2c1 para o display OLED integrado. O software foi refatorado no módulo *hardware_handler.c* para inicializar e gerenciar ambas as portas I2C de forma independente. A lição aprendida foi a importância crucial de se verificar o esquemático e a pinagem específica da placa de desenvolvimento, em vez de assumir as configurações padrão do microcontrolador.

5.2. Desafio da Configuração do Ambiente de Rede (O Erro de "PANIC")

Após a integração da conectividade Wi-Fi e MQTT, o sistema começou a apresentar um erro crítico de `*** PANIC ***` com a mensagem `pool MEMP_SYS_TIMEOUT is empty`.

- **Processo de Depuração:** A mensagem de pânico indicava um esgotamento de um recurso específico da pilha de rede LwIP. A pesquisa sobre o erro apontou que as configurações padrão de alocação de memória para "timeouts" do sistema no Pico SDK eram insuficientes para uma aplicação que mantém uma conexão MQTT persistente.
- **Solução e Lição Aprendida:** A solução foi criar um arquivo de configuração customizado, o `lwipopts.h`, e adicionar uma diretiva para aumentar o tamanho do pool de memória para os timeouts (`#define MEMP_NUM_SYS_TIMEOUT 16`). Isso resolveu o travamento instantaneamente. A lição aprendida foi que, para aplicações de rede robustas, é necessário um entendimento mais profundo das configurações da pilha de rede, indo além das configurações padrão fornecidas pelo SDK.

A superação deste desafio foi fundamental para o sucesso do projeto e para a solidificação do conhecimento em depuração de hardware, configuração de software de baixo nível e automação de processos de desenvolvimento.

6. Conclusão

O presente trabalho detalhou o desenvolvimento do projeto Alerta Frio, uma iniciativa que partiu de uma necessidade prática observada em ambiente profissional e evoluiu para uma solução tecnológica completa no campo da Internet das Coisas. O objetivo central de criar um sistema interativo, de baixo custo e escalável para mitigar perdas de produtos refrigerados em supermercados foi o pilar que norteou todas as fases do projeto.

Ao término do ciclo de desenvolvimento, conclui-se que todos os objetivos específicos foram plenamente alcançados. A entrega final consiste em um protótipo de hardware totalmente funcional, capaz de detectar a temperatura de um item, e um sistema de software embarcado que gerencia estados, exibe informações em um display OLED e se comunica em tempo real com a internet via Wi-Fi e protocolo MQTT. A validação do fluxo de comunicação bidirecional, permitindo que um funcionário confirme remotamente o recebimento de um alerta, representa a principal inovação e o maior sucesso técnico do projeto.

As lições aprendidas ao longo da jornada foram imensuráveis. A superação de desafios complexos, como a depuração de múltiplos barramentos I2C, a resolução de erros de "PANIC" através da configuração da pilha de rede LwIP e a reestruturação de bibliotecas de software, proporcionou um aprendizado prático profundo em sistemas embarcados que transcende a teoria. Este processo rigoroso de resolução de problemas foi essencial para garantir a robustez da solução final.

Portanto, o projeto Alerta Frio não apenas representa a aplicação bem-sucedida dos conhecimentos adquiridos durante a Residência Tecnológica em Sistemas Embarcados, mas também se consolida como uma prova de conceito com genuíno potencial de aplicação e inovação para o setor varejista, demonstrando como a tecnologia IoT pode ser empregada de forma eficaz para resolver problemas do mundo real.

7. Recomendações e trabalhos futuros

O protótipo funcional do Alerta Frio serve como uma base sólida e validada para futuras expansões. As lições aprendidas durante o desenvolvimento abrem caminho para uma série de melhorias que podem transformar o protótipo em um produto comercialmente viável e ainda mais robusto. As seguintes recomendações são propostas:

7.1. Evoluções no Software Embarcado (Firmware)

- **Gestão Avançada de Energia:** Introduzir modos de baixo consumo de energia (*sleep modes*) para aumentar a eficiência energética do dispositivo.
- **Configuração via Ponto de Acesso (Access Point):** Aprimorar o firmware para que o Pico W crie sua própria rede Wi-Fi na primeira inicialização, permitindo a configuração das credenciais da rede local e do broker através de uma página web.

7.2. Melhorias no Hardware

- **Placa de Circuito Impresso (PCB) Customizada:** Migrar o circuito para uma PCB dedicada para otimizar o tamanho, reduzir custos e aumentar a robustez.
- **Design de um Invólucro (Case):** Desenvolver um case para proteger a eletrônica e conferir ao produto uma aparência profissional.
- **Inclusão de Alerta Sonoro:** Adicionar um buzzer para emitir um sinal sonoro junto ao alerta visual do LED, garantindo a notificação em ambientes ruidosos.

7.3. Expansão do Sistema e do Aplicativo

- **Desenvolvimento de Aplicativo Móvel Nativo:** Substituir o cliente MQTT genérico por um aplicativo customizado para Android e iOS, com interface e notificações otimizadas.

- **Criação de um Dashboard Gerencial:** Desenvolver uma aplicação web para supervisores, exibindo um mapa da loja com o status de todos os caixas e gerando estatísticas de desempenho.

7.4. Recomendações de Implantação e Processos

- **Elaboração de um Plano de Treinamento para Colaboradores:** Para garantir o sucesso e a adoção da tecnologia no ambiente do supermercado, recomenda-se a criação de um plano de treinamento focado em dois grupos:
 - **Para Operadores de Caixa:** Um treinamento rápido para que entendam o significado de cada mensagem no display OLED ("Aguardando", "Alerta", "Alerta Visto") e saibam quando e como utilizar o botão de urgência.
 - **Para a Equipe de Recolhimento:** Um treinamento sobre como instalar e configurar o aplicativo cliente MQTT no dispositivo móvel, como interpretar as notificações de alerta e, principalmente, sobre a importância de usar o botão de "Confirmar" para manter o sistema de status sempre atualizado.

8. REFERÊNCIAS

RASPBERRY PI FOUNDATION. Raspberry Pi Pico SDK. Repositório de Software. Disponível em: <https://github.com/raspberrypi/pico-sdk> Acesso em: 11 de julho de 2025

GOOGLE PLAY STORE. IoT MQTT Panel. Aplicativo Cliente MQTT para Android. Disponível em: <https://play.google.com/store/apps/details?id=com.iotpanel.generic.app> Acesso em: 11 de julho de 2025

CANINOS LOUCOS. "Labrador - Single Board Computer". Disponível em: <https://www.caninosloucos.org/labrador> Acesso em: 11 de julho de 2025.

ECLIPSE FOUNDATION. "Eclipse Mosquitto - An open source MQTT broker". Disponível em: <https://mosquitto.org>. Acesso em: 11 de julho de 2025.

9. APÊNDICE A – CÓDIGO FONTE DO PROJETO

O código fonte completo do firmware desenvolvido para o microcontrolador Raspberry Pi Pico W, bem como os arquivos de configuração do projeto, estão disponíveis publicamente para consulta no seguinte Drive:

[Alerta Frio 2.0](#)