INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
CAMPUS: CAMPUS FORTALEZA
DISCIPLINA: PROJETO INTEGRADOR
EMBARCATECH - ETAPA 2 - RESIDÊNCIA

CARLOS DELFINO CARVALHO PINHEIRO
202510110980368

RACK INTELIGENTE: SISTEMA AIOT DE GERENCIAMENTO E SEGURANÇA AMBIENTAL PARA RACKS EM INFRAESTRUTURAS DE PEQUENO PORTE

Trabalho de Conclusão de Curso apresentado ao Instituto Federal como requisito parcial do projeto Embarcatech - Etapa 2 - Residência.

FORTALEZA - CE
2025

RELATÓRIO DE EVIDÊNCIA DE DESENVOLVIMENTO (RED)

1. VISÃO GERAL DO PROJETO

Este projeto implementa um **Rack Inteligente** voltado para **monitoramento ambiental** e **segurança física** em infraestruturas de pequeno porte. A solução integra:

- **Firmware embarcado** (C/C++ com FreeRTOS), responsável por leitura de sensores, atuação e comunicação.
- **Camada IoT** via **MQTT**, para transporte de telemetria e comandos.
- **Dashboard** (Python), para visualização, controle manual e integração com recursos de IA.
- **AIoT**, com uso de **LLM** e **modelos de séries temporais** para apoiar decisões e previsões.

A ideia principal é separar bem as responsabilidades para facilitar manutenção, testes e evolução do sistema.

2. ESTADO ATUAL (MARCOS E ESTABILIDADE)

O projeto está em **estágio avançado**, com os marcos principais **praticamente entregues**. Nesta etapa eu estou focado em:

- **Revisão geral do código** (refatorações pontuais e padronização).
- **Correção de bugs menores**.
- **Reatividade do sistema**, onde testo resposta a comandos e simulação de múltiplos Racks.
- **Testes de resiliência e estabilidade** (execução contínua por longos períodos).

Evidências observadas até aqui:

- **Firmware**: executando por dias com boa estabilidade.
- **Dashboard**: responde bem ao monitoramento simultâneo de **dezenas a centenas de racks**.
- **Broker MQTT (Mosquitto)** em VPS: serviço estável há **mais de 3 meses**, sem incidentes relevantes.

3. ARQUITETURA DO FIRMWARE (C/C++ COM FREERTOS)

3.1. ESTRATÉGIA DE CONCORRÊNCIA: UMA TASK POR RESPONSABILIDADE

Foi adotada a tática de **uma task por responsabilidade**. Essa decisão melhora:

- **Diagnóstico** (fica mais fácil isolar gargalos e falhas).
- **Manutenção** (alterações ficam localizadas).
- **Evolução** (troca de protocolos ou drivers sem reescrever o sistema inteiro).

Um exemplo importante: para publicação de dados via MQTT, há tasks dedicadas por parâmetro monitorado. Isso permite, se necessário, substituir apenas a parte de transporte (por exemplo, trocar MQTT por outro protocolo) **sem interferir** na coleta e no cálculo dos parâmetros ambientais.

3.2. ENTRADA DO TECLADO E CONTROLE DO MENU

As tasks relacionadas ao teclado utilizam **notificações (task notify)** com **flags** para representar as teclas pressionadas. Há duas responsabilidades principais:

- **Task de leitura de GPIOs** do teclado.
- **Task de processamento** das teclas e controle do menu.

O menu é estruturado como uma **máquina de estados**. Embora ainda não esteja totalmente finalizado, ele demonstra uma abordagem orientada a eventos, típica de sistemas embarcados robustos.

3.3. PROCESSAMENTO DE COMANDOS: FILAS (QUEUES) E ACK

Para comandos recebidos (por exemplo, **abrir porta**, **acionar alarme** e **acionar ventilação**), foi adotada a abordagem com **queues**. Com isso:

- Os comandos são enfileirados em ordem **FIFO**.
- A execução ocorre de forma previsível, reduzindo risco de concorrência.
- Ao final, é gerado um **ACK** confirmando que o comando foi recebido e processado.

3.4. SENSORES E COMUNICAÇÃO COM MQTT

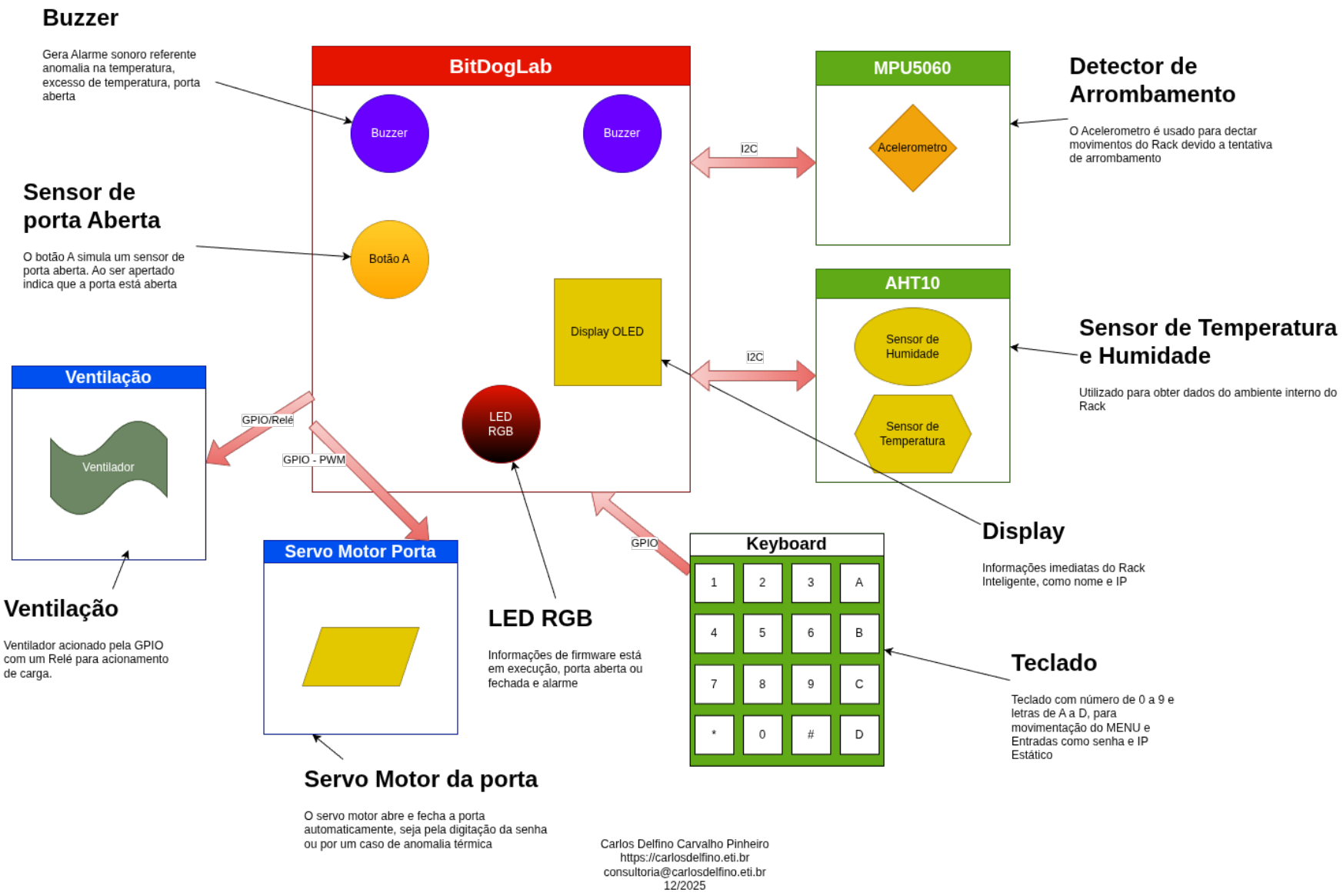
Para sensores (movimento, temperatura e umidade), o firmware notifica componentes responsáveis por publicar telemetria. A escolha de múltiplas classes/módulos, mesmo quando seria possível concentrar lógica em um único ponto, foi intencional para reforçar:

- Modularização.
- Separação de responsabilidades.
- Organização do sistema em um formato mais próximo de um produto evolutivo.

4. HARDWARE

O hardware foi desenvolvido conforme o pré-projeto. O diagrama de blocos abaixo resume a estrutura e o funcionamento:

Hardware do Rack Inteligente

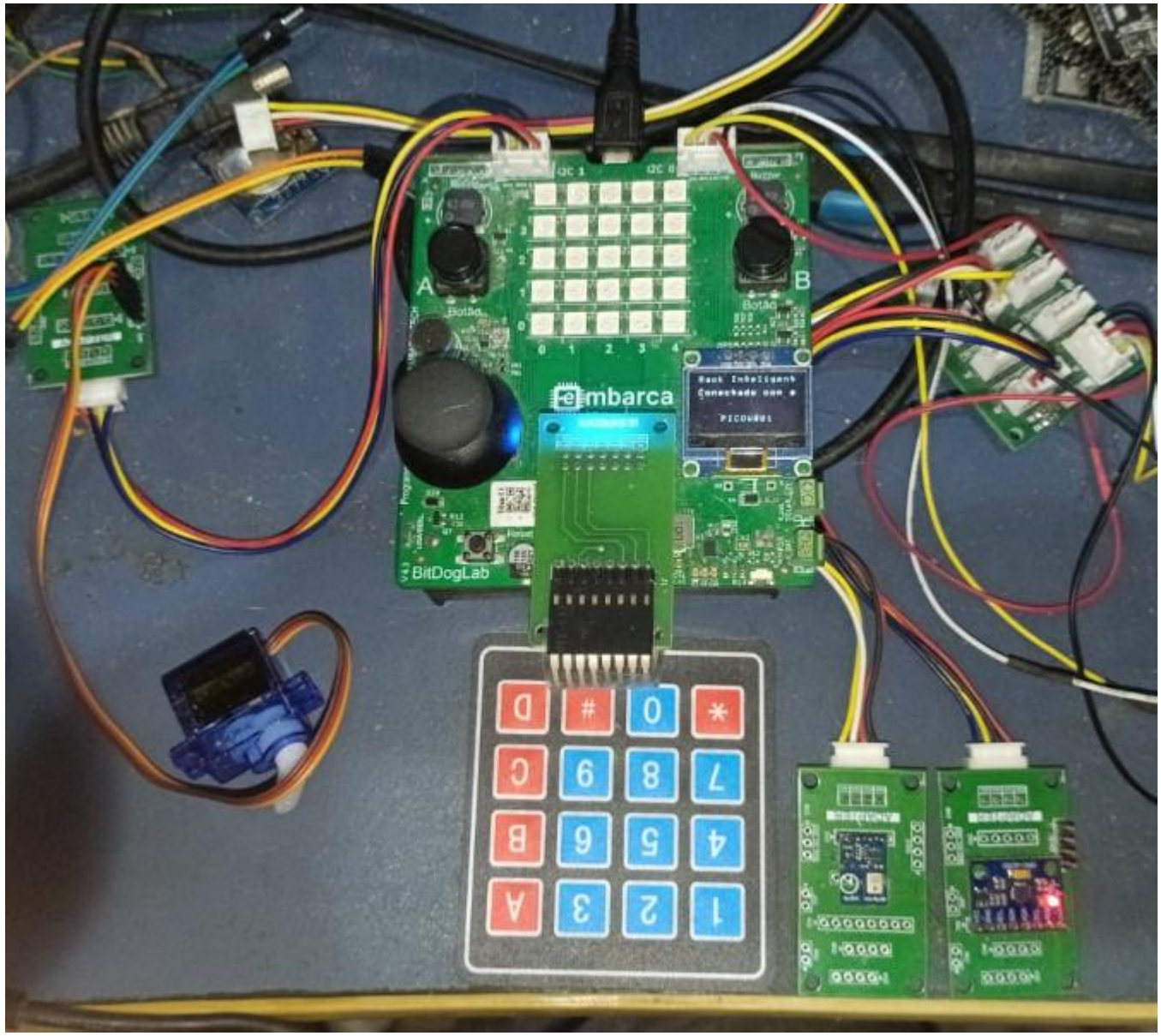


Carlos Delfino Carvalho Pinheiro
https://carlosdelfino.eti.br
consultoria@carlosdelfino.eti.br

12/2025

A imagem a seguir mostra o protótipo montado. Nela:

- Adaptador e **servo motor** (acionamento da porta).
- Placa de distribuição **I2C**.
- Sensor de movimento (**MPU5060**).
- Sensor de temperatura e umidade (**AHT10**).
- Teclado.
- A **BitDogLab** ao centro, já integrando LEDs RGB e display OLED.


A
Botão
B
Botão
embarca
BitDogLab
D
#
0
*
C
9
8
7
B
6
5
4
A
3
2
1

ESQUEMÁTICO DAS CONEXÕES

No esquemático vemos o I2C usado pelo sensor de movimento (MPU6050) e sensor de temperatura e umidade (AHT10) conectado aos pinos GPIO0 e GPIO1 para o canal I2C0 do I2C externalizado através do conector de mesmo nome .

Já o PWM para controle do servo motor está no pino GPIO2, externalizado pelo conector de nome I2C1, mas neste uso não tem relação alguma com o I2C.

Os pinos de conexão do Display OLED são GPIO14 e GPIO15 conectados ao canal I2C1, como usam outros pinos não conflitam com o conector de mesmo nome.

Os LEDs RGB estão conectados nos pinos GPIO12, GPIO13 e GPIO15, respectivamente.

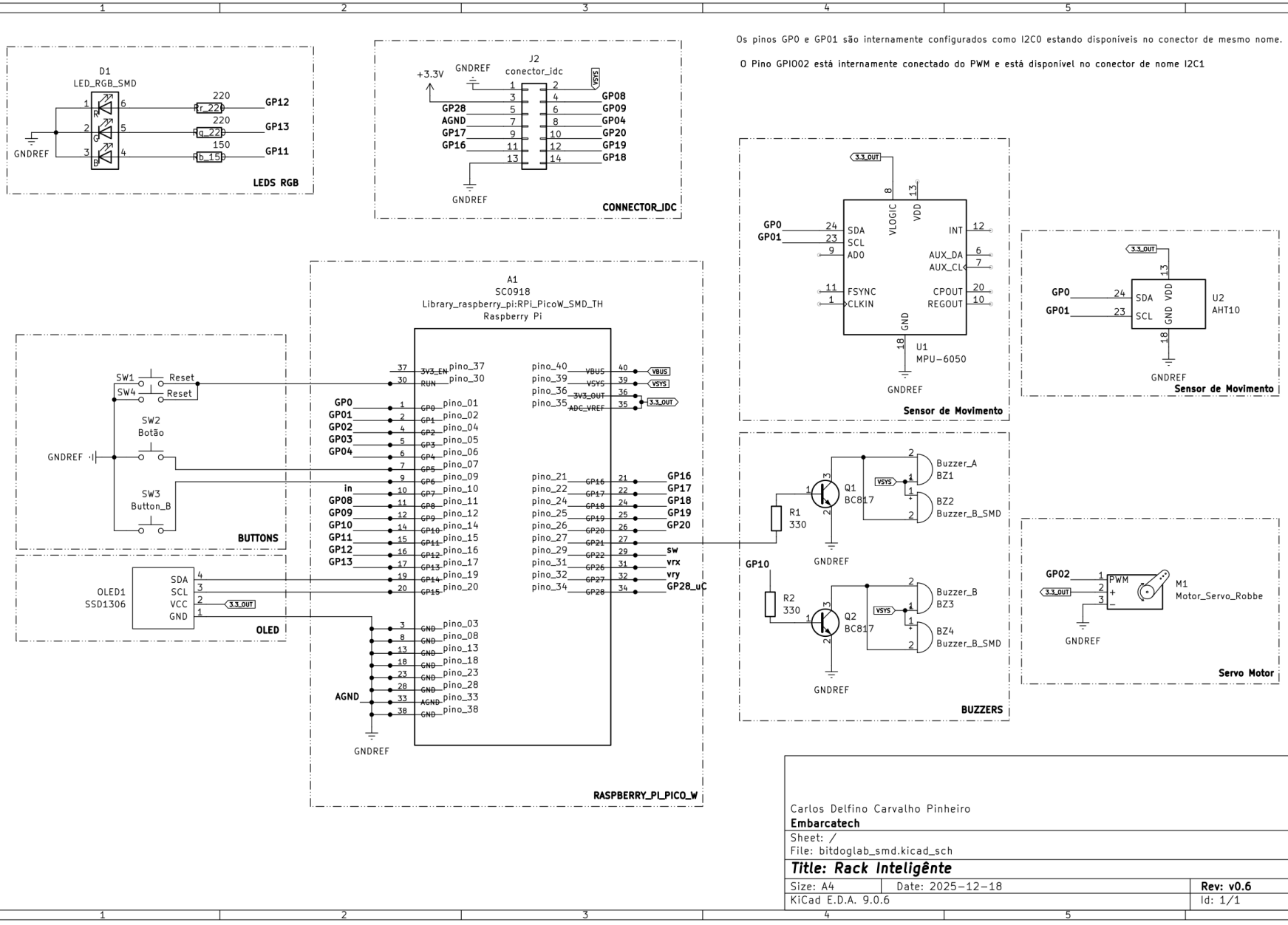
Os buzzer estão conectados ao GPIO10 e GPIO27, sendo controladores por PWM.

Os botões, botão A está conectado ao pino GPIO4 e o botão B GPIO5.

O Ventilador está representado pelo LED Verde, O LED Azul pisca indicando funcionamento do firmware corretamente, durante o boot rápidamente e durante o funcionamento normal muda de estado a cada 1 segundo.

A detecção de porta aberta está sendo feita pelo botão A, ao ser apertado a porta está aberta. em posição normal está fechada.

O botão B, desativa o alarme até a próxima anomalia alertada.



Carlos Delfino Carvalho Pinheiro
Embarcatech

Sheet: /
File: bitdoglab_smd.kicad_sch

Title: Rack Inteligênte

Size: A4	Date: 2025-12-18	**Rev: v0.6**
KiCad E.D.A. 9.0.6		Id: 1/1

5. CAMADA IOT (MQTT)

Foi adotado o protocolo **MQTT** para comunicação entre o firmware e o dashboard, por ser leve e adequado a telemetria e comandos em tempo real.

O broker foi implantado em um **servidor VPS**, com:

- Regras de **firewall** para reduzir superfície de ataque.
- Autenticação por **usuário/senha**.

Isso fornece uma base adequada para segurança operacional no cenário do projeto.

6. AIOT (IA APLICADA AO SISTEMA)

6.1. INTEGRAÇÃO COM LLM VIA OPENWEBUI/OLLAMA

Foi adotado o **OpenWebUI** como interface de acoplamento entre o dashboard e a engine de LLMs (via **Ollama**). Essa estratégia permite realizar requisições usando compatibilidade com a biblioteca OpenAI, facilitando a troca de engine GenAI sem mudanças grandes no código (normalmente basta alterar URL e chave de API).

6.2. TOOL CALLING PARA DECISÃO E ATUAÇÃO

Foram usados recursos de **Tool Calling** para apoiar decisões relacionadas a:

- Temperaturas em níveis críticos.
- Acionamento de ventilação.
- Disparo de alarmes.

O modelo utilizado como orquestrador de ferramentas é o **Granite 4** (3 bilhões de parâmetros).

6.3. PREVISÃO DE TEMPERATURA E UMIDADE (TTM)

Foi adicionado um mecanismo de previsão com o **Granite TTM (Tiny Time Series Model)**, utilizando dados históricos para estimar valores futuros de temperatura e umidade. Isso ajuda a antecipar situações de risco (por exemplo, tendência de aquecimento contínuo).

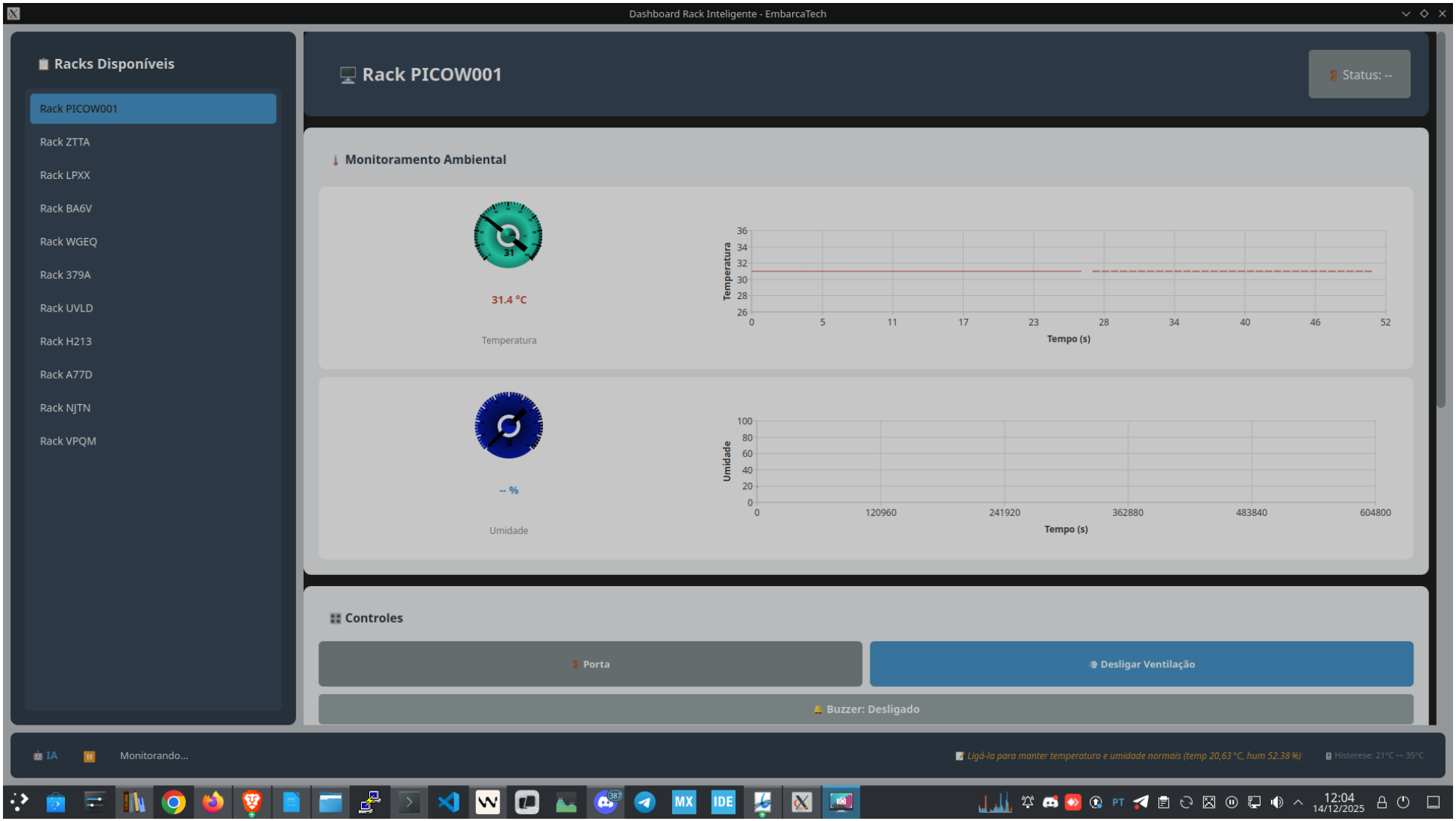
7. TREINAMENTO E PARAMETRIZAÇÃO DA IA

- O prompt principal para o comportamento da GenAI está na pasta `prompts/`.
- As variáveis de execução do dashboard e parâmetros de treinamento/uso do Granite TTM são configuradas via arquivo `.env` na raiz do projeto.

8. DASHBOARD (PYTHON)

O dashboard foi desenvolvido em **Python** para facilitar a integração com a LLM e com o TTM (ambos modelos Granite da IBM) e acelerar os ciclos de teste.

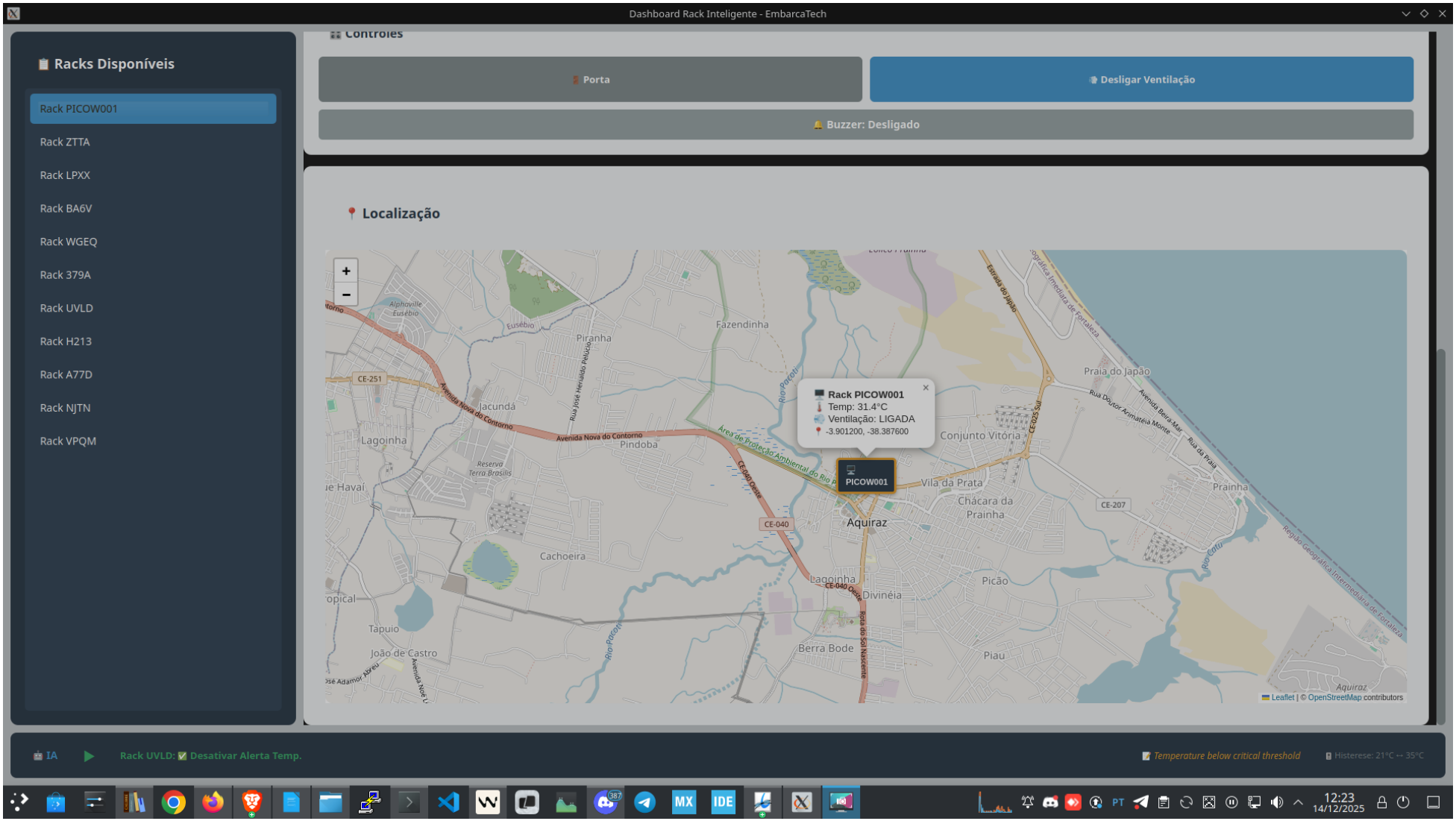
A seguir, a janela principal:


Dashboard Rack Inteligente - EmbarcaTech
Racks Disponíveis
Rack PICOW001
Rack ZTTA
Rack LPXX
Rack BA6V
Rack WGEQ
Rack 379A
Rack UVLD
Rack H213
Rack A77D
Rack NJTN
Rack VPQM
Rack PICOW001
Status: --
Monitoramento Ambiental
31
31.4 °C
Temperatura
Temperatura
36
34
32
30
28
26
0
5
11
17
23
28
34
40
46
52
Tempo (s)
-- %
Umidade
Umidade
100
80
60
40
20
0
0
120960
241920
362880
483840
604800
Tempo (s)
Controles
Porta
Desligar Ventilação
Buzzer: Desligado
IA
Monitorando...
Ligá-la para manter temperatura e umidade normais (temp 20,63 °C, hum 52.38 %)
Histerese: 21°C ↔ 35°C
PT
12:04
14/12/2025

Componentes principais da interface:

- **Lista de racks (lado esquerdo)**: alimentada dinamicamente conforme as telemetrias são publicadas no MQTT. O nome do rack pisca quando algum parâmetro é atualizado.
- **Painel de telemetria (lado direito)**: temperatura e umidade, traço contínuo valor histórico coletado, traço descontinuo previsão com baseno histórico.
- **Controles manuais**: botões para abrir porta e acionar ventilação, além de sinalização do estado do alarme.

Ao rolar o conteúdo, é exibido o mapa de localização do rack, com dados obtidos pelo parâmetro de GPS enviado pelo Rack Inteligente:


Dashboard Rack Inteligente - EmbarcaTech
Racks Disponíveis
Rack PICOW001
Rack ZTTA
Rack LPXX
Rack BA6V
Rack WGEQ
Rack 379A
Rack UVLD
Rack H213
Rack A77D
Rack NJTN
Rack VPQM
Controles
Porta
Desligar Ventilação
Buzzer: Desligado
Localização
Rack PICOW001
Temp: 31.4°C
Ventilação: LIGADA
-3.901200, -38.387600
PICOW001
Aquiraz
Fazendinha
Piranha
Jacundá
Lagoinha
Pindoba
Cachoeira
Tapuio
João de Castro
Berra Bode
Lagoinha
Divinéia
Picão
Piau
Vila da Prata
Chácara da Prainha
Conjunto Vitória
Praia do Japão
Prainha
CE-251
CE-040
CE-207
Avenida Nova do Contorno
Leaflet | © OpenStreetMap contributors
IA
Rack UVLD: Desativar Alerta Temp.
Temperature below critical threshold
Histerese: 21°C ↔ 35°C
12:23
14/12/2025

No rodapé, são exibidas as **reflexões e decisões** produzidas pela GenAI a partir dos parâmetros de telemetria.

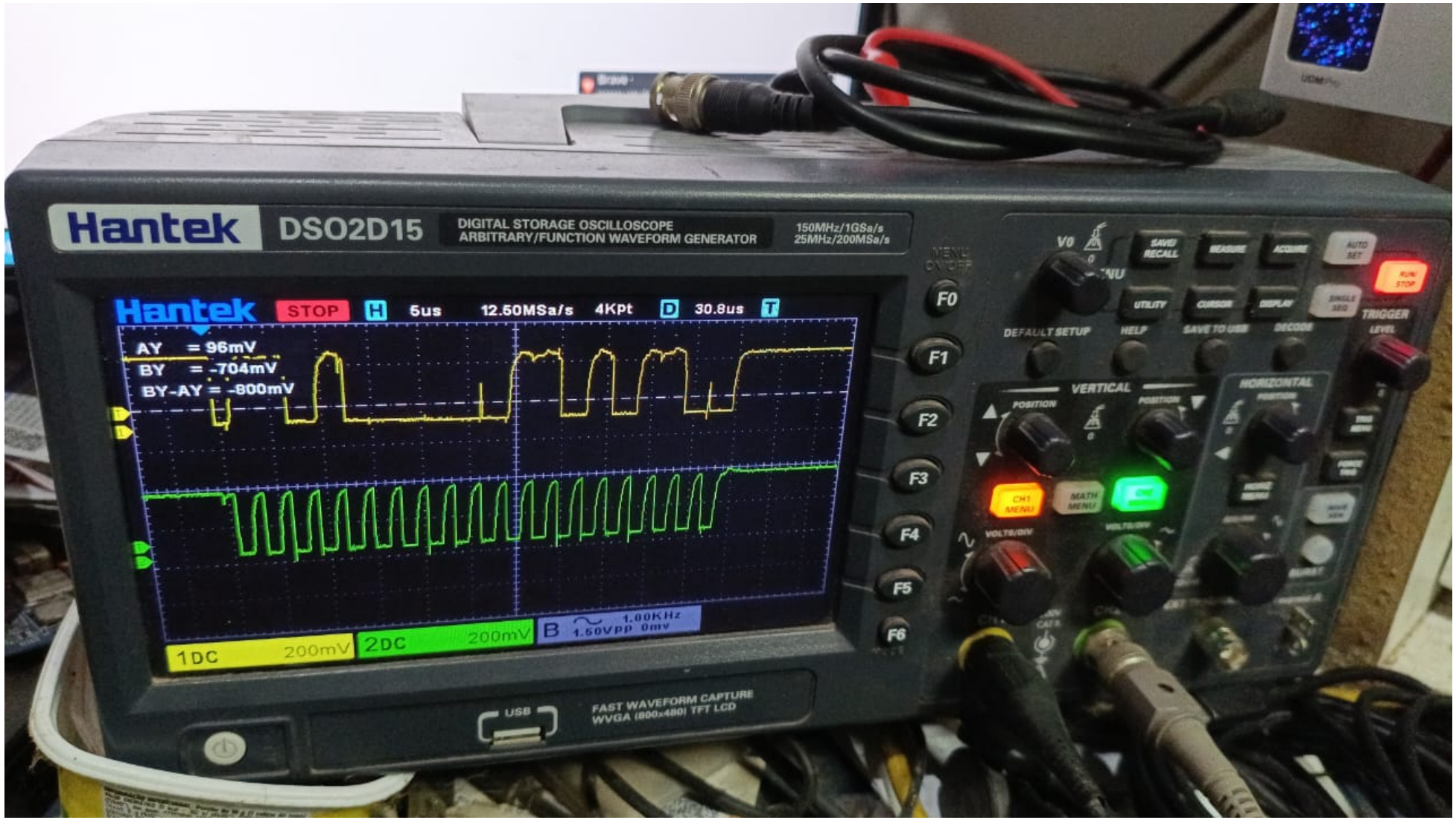
9. TESTES REALIZADOS (EVIDÊNCIAS)

Além do uso com o hardware disponível, foram feitos testes para validar escala e estabilidade:

- Teste de **execução contínua** (firmware + dashboard ligados por dias).
- Teste de **carga lógica** no dashboard usando um script Python que simula racks extras publicando dados aleatórios (telemetria e localização), permitindo avaliar:
- Desempenho da interface.
- Estabilidade do broker MQTT.
- Comportamento do sistema sob múltiplas fontes de dados.

9.1. MEDIÇÕES E ANÁLISE DO CIRCUITO

Foram feitas medições e analises do circuito para identificar ponto de falhas. Apesar da deformação do sinal no barramento I2C (deveria ser uma onda quadrada mais retilinea) a comunicação funciona perfeitamente bem.


Hantek
DSO2D15
DIGITAL STORAGE OSCILLOSCOPE
ARBITRARY/FUNCTION WAVEFORM GENERATOR
150MHz/1GSa/s
25MHz/200MSa/s
Hantek
STOP
H
5us
12.50MSa/s
4KPt
D
30.8us
T
AY = 96mV
BY = -704mV
BY-AY = -800mV
1DC 200mV
2DC 200mV
B 1.00KHz
1.50Vpp 0mv
USB
FAST WAVEFORM CAPTURE
WVGA (800x480) TFT LCD
F0
F1
F2
F3
F4
F5
F6
VERTICAL
HORIZONTAL
TRIGGER
DEFAULT SETUP
HELP
SAVE TO USB
DECODE
SAVE/ RECALL
MEASURE
ACQUIRE
AUTO SET
UTILITY
CURSOR
DISPLAY
RUN/ STOP
CH1 MENU
MATH MENU
VOLTS/DIV

10. PROBLEMAS ENCONTRADOS E SOLUÇÕES ADOTADAS

- **Limitação de bancada (ausência de um rack real)**
- Para simular cenários, foram usados métodos práticos para elevar temperatura/ umidade, permitindo observar respostas do sistema.
- **Necessidade de testar vários racks com apenas um kit**
- Foi criado um simulador que injeta dados de múltiplos racks (com geolocalizações variadas e grande variação de temperatura/umidade), viabilizando testes mais completos do dashboard.
- **Sensor de movimento aparentemente não funcional**
- Está em avaliação a possibilidade de troca do componente junto ao EmbarcaTech.
- **Falha no envio de umidade (intermitente)**
- Próxima ação: revisar sincronismo entre a task de coleta e a task de publicação MQTT, garantindo consistência na atualização do valor.

11. PRÓXIMOS PASSOS

- Adicionar um **watchdog timer** para detectar travamentos e reiniciar o hardware quando necessário.
- Finalizar o **menu local** para troca de senha.
- Implementar armazenamento dinâmico e seguro de configurações (por exemplo, senha e dados de rede Wi-Fi).
- Evoluir a coleta e uso de geolocalização do rack, incluindo opções de configuração.
- Adicionar no dashboard recursos para **recuperação de senha remota** (com segurança e auditoria).
- Melhorar o treinamento e calibragem dos componentes de IA, tanto do **Granite TTM** quanto do **Granite LLM**.

12. PROJETO NO GITHUB

O projeto pode ser clonado, lembre-se de clonar recursivamente, pois o firmware está em outro repositório. Assim poderá ser analisado em seu estágio atual:

- Projeto Geral: https://github.com/ArvoreDosSaberes/Embacathec-Etapa-2---Projeto-Final.git.
- Projeto Firmware: https://github.com/ArvoreDosSaberes/Embarcatech-Etapa-2---Projeto-Final-firmware
- Em breve Dashboard e Simulador estarão em seus repositórios próprios para facilitar minha gerencia do código.