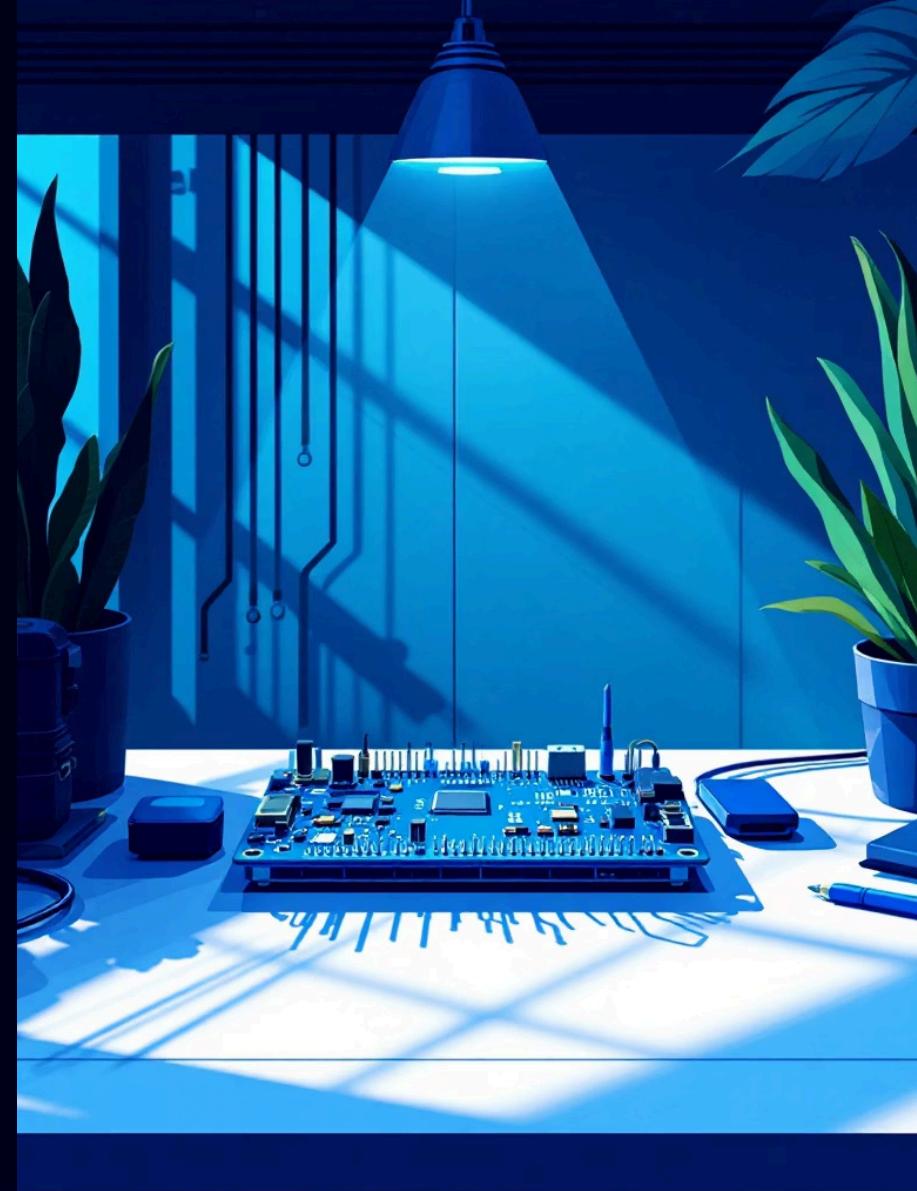


# Configuração do Ambiente Zephyr OS

Ajustes de Ferramentas e Planejamento Futuro

Nicolas Jordani – UFSM | Novembro 2025



# Visão Geral do Projeto

Este relatório documenta o processo completo de instalação e configuração do ambiente de desenvolvimento Zephyr OS em máquina virtual, incluindo desafios encontrados e soluções implementadas.

O projeto visa estabelecer uma base sólida para desenvolvimento de sistemas embarcados com comunicação BLE e rádio.

01

## Configuração do Ambiente

Instalação e ajustes iniciais

02

## Resolução de Problemas

Correção de incompatibilidades

03

## Implementação Futura

BLE e comunicação por rádio



# Ambiente de Desenvolvimento



## Sistema Operacional

Ubuntu 24.04 LTS (Noble) em VirtualBox com particionamento manual



## Python

Versão 3.10.x após ajustes de compatibilidade



## Zephyr OS

Versão v4.3.99 com west v1.5.0



## Toolchain

Zephyr SDK 0.16.x para compilação

# Desafios Técnicos Encontrados

1

## Incompatibilidade Python 3.12

Ubuntu 24.04 usa Python 3.12 por padrão, mas Zephyr requer versões 3.8-3.10. Solução: instalação via Deadsnakes e criação de venv dedicado.

2

## West Bloqueado em 1.5.0

PyPI oferece apenas west 1.5.0, enquanto Zephyr main requer  $\geq 1.12$ . Impossível atualizar via pip.

3

## Branch Main Incompatível

Funcionalidades removidas causaram erros: falta de runners.yaml, remoção do comando west run, mudanças na simulação.

4

## Espaço em Disco Insuficiente

Particionamento original inadequado para SDK, árvore Zephyr e cache. Expansão para  $\sim 40$  GB necessária.

# Processo de Execução Validado

## Ativar Ambiente Virtual

```
source .venv-310/bin/activate
```

## Acessar Workspace

```
cd /mnt/dados/zephyr
```

## Compilar Exemplo

```
west build -b qemu_x86 zephyr/samples/hello_world -d build
```

## Executar no QEMU

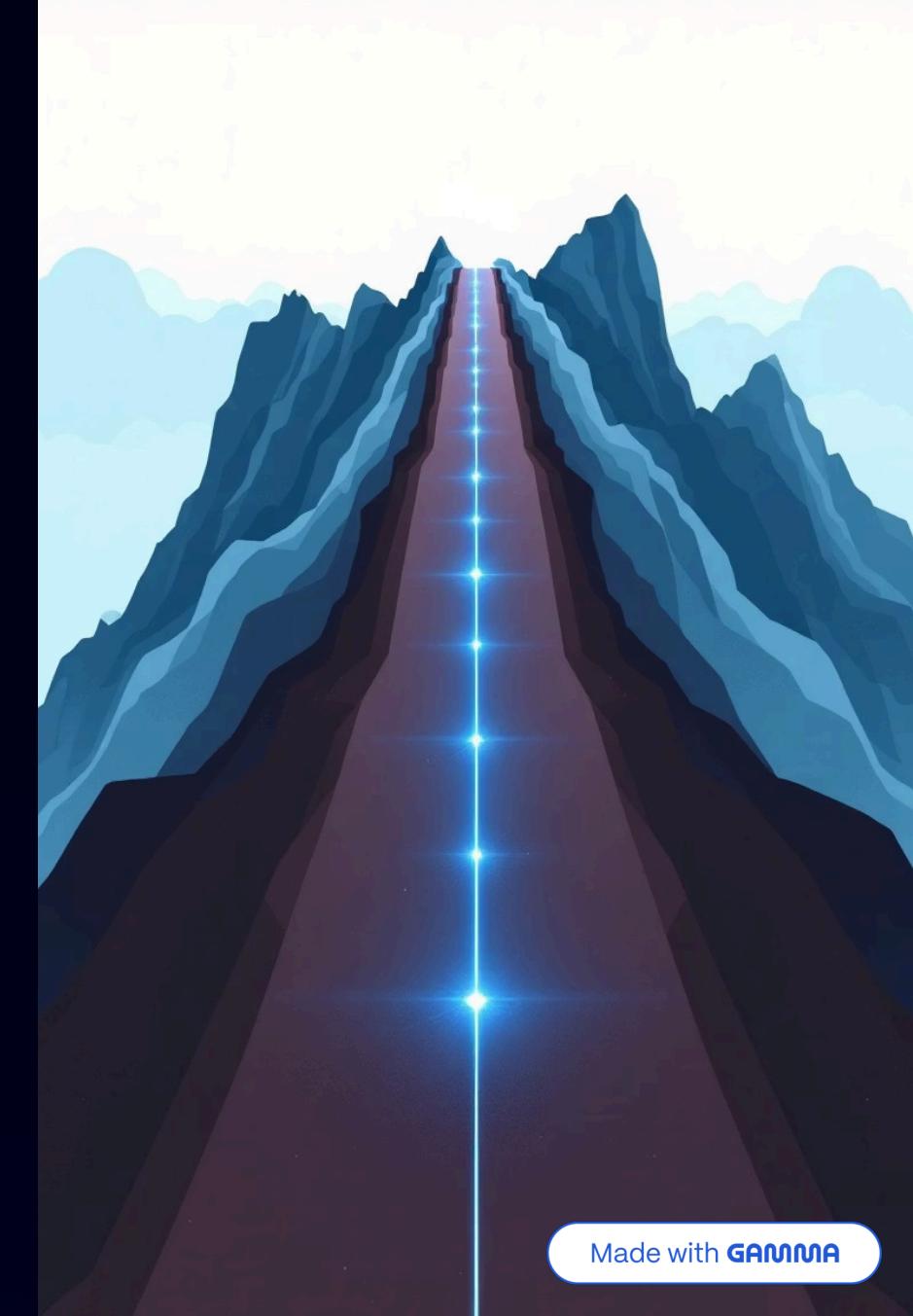
```
west build -t run
```

## Encerrar QEMU

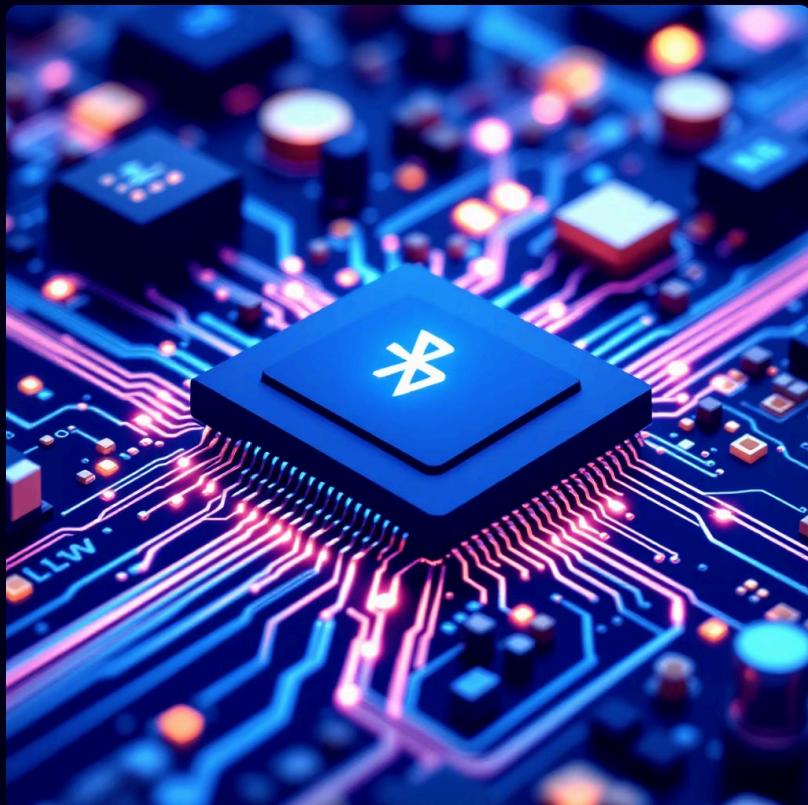
```
Ctrl + A, X
```

- ☐ Este fluxo foi testado e comprovadamente funcional após todas as correções implementadas.

# Próximos Passos



# Implementação de BLE



## Acesso ao Log via Celular

Integração de Bluetooth Low Energy permitirá visualização de logs da placa via smartphone, envio de comandos de depuração e interação com aplicativos BLE em Android e iOS.



### Habilitar Subsistema BLE

CONFIG\_BT e CONFIG\_BT\_PERIPHERAL no Zephyr



### Criar Serviço GATT

Serviço customizado com característica de log



### Testar Conectividade

Validação com nRF Connect no smartphone

# Comunicação Entre Placas por Rádio



Após sucesso no BLE, o próximo passo integra comunicação por rádio entre duas placas. O objetivo é adaptar aplicativo de comunicação existente, criando arquitetura híbrida: Celular → BLE → Placa 1 → Rádio → Placa 2.

# Conquistas e Aprendizados



## Problemas Críticos

Resolvidos com sucesso

O processo exigiu correções significativas devido a mudanças recentes no Zephyr OS e restrições nas versões do west disponíveis no PyPI.



## Ambiente Funcional

Totalmente estabilizado

Após ajustes no Python, particionamento da VM e método de execução, o ambiente foi estabilizado com sucesso, preparando o terreno para avanços funcionais.



## Próximas Fases

BLE e comunicação rádio

# Conclusão e Perspectivas

O ambiente de desenvolvimento Zephyr OS foi estabelecido com sucesso, superando desafios técnicos complexos e criando base sólida para evolução do projeto.

## 1 Fase Atual

Ambiente estável e funcional

## 2 Curto Prazo

Implementação BLE para logs via celular

## 3 Médio Prazo

Comunicação híbrida BLE + rádio entre placas

## 4 Visão Futura

Arquitetura robusta de comunicação entre dispositivos

