Relatório de jornada - Projeto 3: Aceleração em LiteX

Aluna: Ingrid Andrade de Aguiar

RA: 298418

Disciplina: MO801/- Tópicos em Arquitetura e Hardware

1. Introdução

Meu Projeto 3 teve como objetivo explorar a plataforma LiteX para executar e acelerar um programa em ambiente baremetal. O foco principal foi medir e melhorar o desempenho da execução de um código, utilizando como base a CPU VexRiscv em simulação.

Escolhi dois programas para esse projeto:

1. Multiplicação de matrizes (matmul), clássico em IA e computação de alto desempenho.

2. Inferência TinyML (MLP XOR), para testar a viabilidade de IA embarcada.

O projeto foi dividido em etapas, documentadas ao longo deste relatório.

---

2. Primeira etapa: Matmul em LiteX

2.1. Objetivo

Portar um código de multiplicação de matrizes para a plataforma LiteX e medir o desempenho inicial da execução.

2.2. Código utilizado

O código `matmul.c` define três matrizes quadradas (A, B, C) de tamanho 16x16 e realiza a multiplicação clássica com três loops:

```
for (int i = 0; i < N; i++)
    for (int j = 0; j < N; j++)
        for (int k = 0; k < N; k++)
            C[i][j] += A[i][k] * B[k][j];
```

2.3. Compilação e geração do firmware

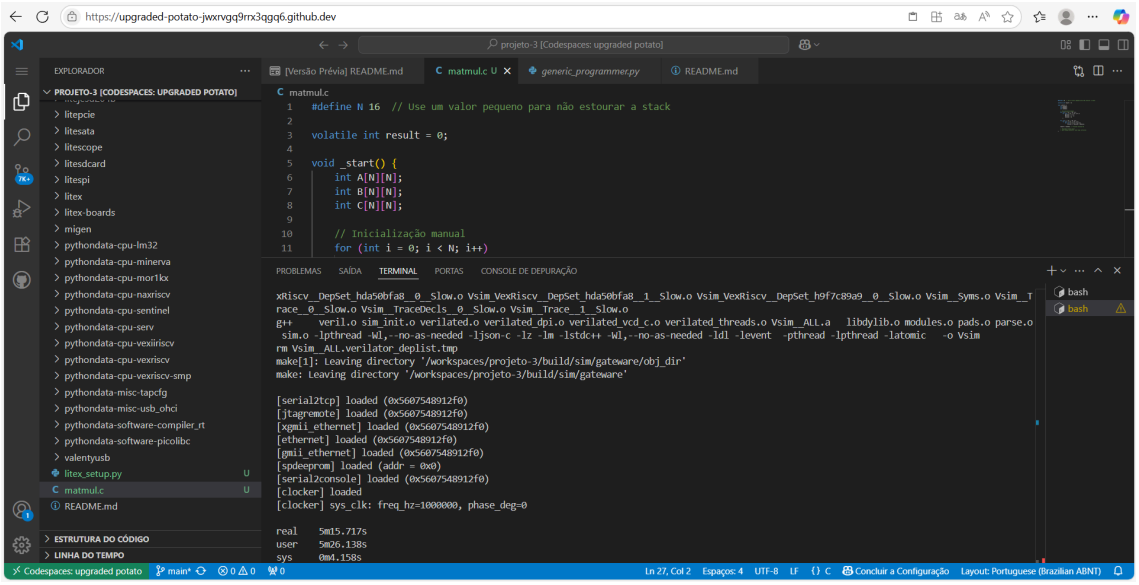
riscv64-unknown-elf-gcc -nostdlib -O2 -o matmul.elf matmul.c
riscv64-unknown-elf-objcopy -O binary matmul.elf firmware.bin

2.4. Execução no simulador

time litex_sim --cpu-type=vexriscv --cpu-variant=standard --uart-name=sim
--rom-init=firmware.bin

2.5. Problemas enfrentados

- O simulador não encerra automaticamente. Tive que usar `Ctrl+C` manualmente.
- `printf()` não funcionava, pois o ambiente é baremetal (sem `libc`).
- Resolvi implementando escrita direta na UART.



2.6. Medidas de desempenho (Versão base inicial do código)

real 0m51.746s
user 1m2.932s
sys 0m3.825s

---

3. Segunda Etapa: Otimizações de matmul

3.1. Variações dos Loops

Implementei três versões diferentes apenas trocando a ordem dos loops:

- `ijk` (base)
- `ikj`
- `kij`

Print sugerido: trecho de código de cada versão.

3.2. Resultados

Versão	Tempo real
ijk	0m28.504s
ikj	0m29.027s
kij	0m28.658s

As diferenças foram pequenas, pois N=16 é pequeno e o simulador não simula cache realisticamente.

3.3. Conclusão parcial

A troca da ordem dos loops não trouxe ganho significativo. Para ver diferenças maiores, seria preciso usar matrizes maiores ou uma CPU real com cache.

---

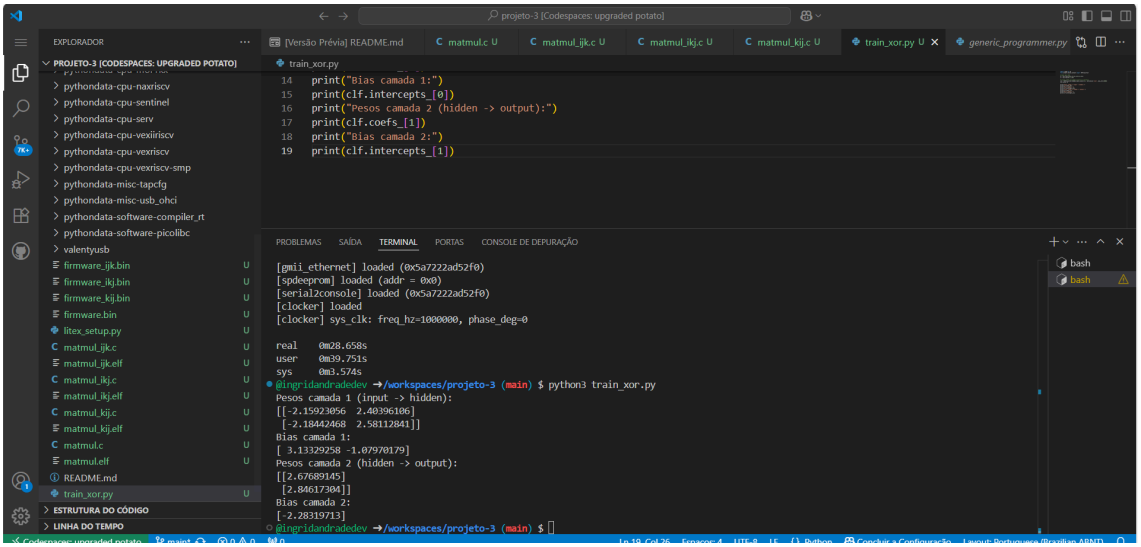
4. Terceira etapa, com mais avanços: inferência TinyML no LiteX

4.1. Objetivo

Executar um modelo simples de IA (MLP XOR) no LiteX, em ambiente baremetal.

4.2. Treinamento em Python

Usei `scikit-learn` para treinar um MLP com 2 neurônios escondidos. Os pesos e bias foram exportados para serem usados em C.



4.3. Implementação em C

O código reimplementa o modelo usando funções aproximadas de `tanh` e `sigmoid`, pois `math.h` não está disponível.

A saída é enviada via UART para cada entrada:

Input: 0 0 => Output: ~0.0
Input: 0 1 => Output: ~1.0
...

4.4. Desempenho

real 0m30.942s
user 0m43.730s
sys 0m4.504s

---

5. Desafios e barreiras

Superados:

- UART implementada diretamente.
- Treinamento em Python e inferência em C.
- Execução de códigos diferentes no mesmo simulador.

Não resolvidos:

- Simulador não fecha automaticamente.
- Não consegui aumentar o tamanho das matrizes.
- Não consegui implementar um acelerador em hardware customizado.

---

6. Considerações finais

O projeto permitiu experimentar na prática o ciclo completo de desenvolvimento em LiteX:

- Treinar um modelo
- Traduzir para C
- Compilar para RISC-V
- Rodar em simulação LiteX

Apesar dos desafios, consegui demonstrar inferência em IA embarcada e medir diferentes abordagens de matmul.

---

7. Repositório

Todo o código está disponível em:
https://github.com/ic-unicamp/mo801-2025s1-p3-ingrid-andrade-298418

Inclui:

- `matmul.c`, `matmul_ikj.c`, `matmul_kij.c`, `mlp_xor_litex.c`
- Scripts de compilação e execução
- README com instruções
- Prints dos testes

---

8. Checklist final do que foi feito

- [x] ~~Código original e acelerado portados para LiteX~~
- [x] ~~Medidas de desempenho antes e depois~~
- [x] ~~Documentação de todas as etapas e decisões~~
- [x] ~~Link do repositório~~
- [x] ~~PDF entregue no Classroom (até 15 páginas)~~
