Grupo: Davi Sousa Soares

Victor Kauan da Silva Miranda

Professor: Ivan Silva Saraiva

MAIO

2025

Conteúdo

1. Introdução

2. Metodologia

2.1. Implementação em Assembly RISC-V

2.2. Implementação em Python

3. Resultados

3.1. Comparação Visual

3.2 Análise

4. Conclusão

## **1. Introdução**

O presente trabalho tem como objetivo implementar um programa que calcule o histograma de um canal (neste caso, o canal vermelho) de uma imagem no formato QCIF. O histograma foi implementado tanto em linguagem de baixo nível (Assembly RISC-V) utilizando o simulador RARS, quanto em linguagem de alto nível (Python). Ao final, os resultados dos dois métodos são comparados para verificar a consistência das implementações.

## **2. Metodologia**

### **2.1 Implementação em Assembly RISC-V**

O código trabalhoArq.asm foi desenvolvido para:

Ler os dados da imagem já convertida para um formato interpretável (lista de valores de pixels do canal vermelho).

Calcular o histograma de frequências (ocorrências de cada valor de pixel de 0 a 255).

Imprimir na tela no formato:

Pixel X - Acorrencia Y

O histograma é salvo em um arquivo de texto histograma\_rars.txt, posteriormente usado para comparação com o histograma em alto nível.

### **2.2 Implementação em Python**

O arquivo frequencia.py realiza as seguintes tarefas:Abre a imagem original 1.jpg, localizada no diretório imagens/.

Extrai o canal vermelho de todos os pixels.

Gera um histograma interativo utilizando a biblioteca plotly.express.

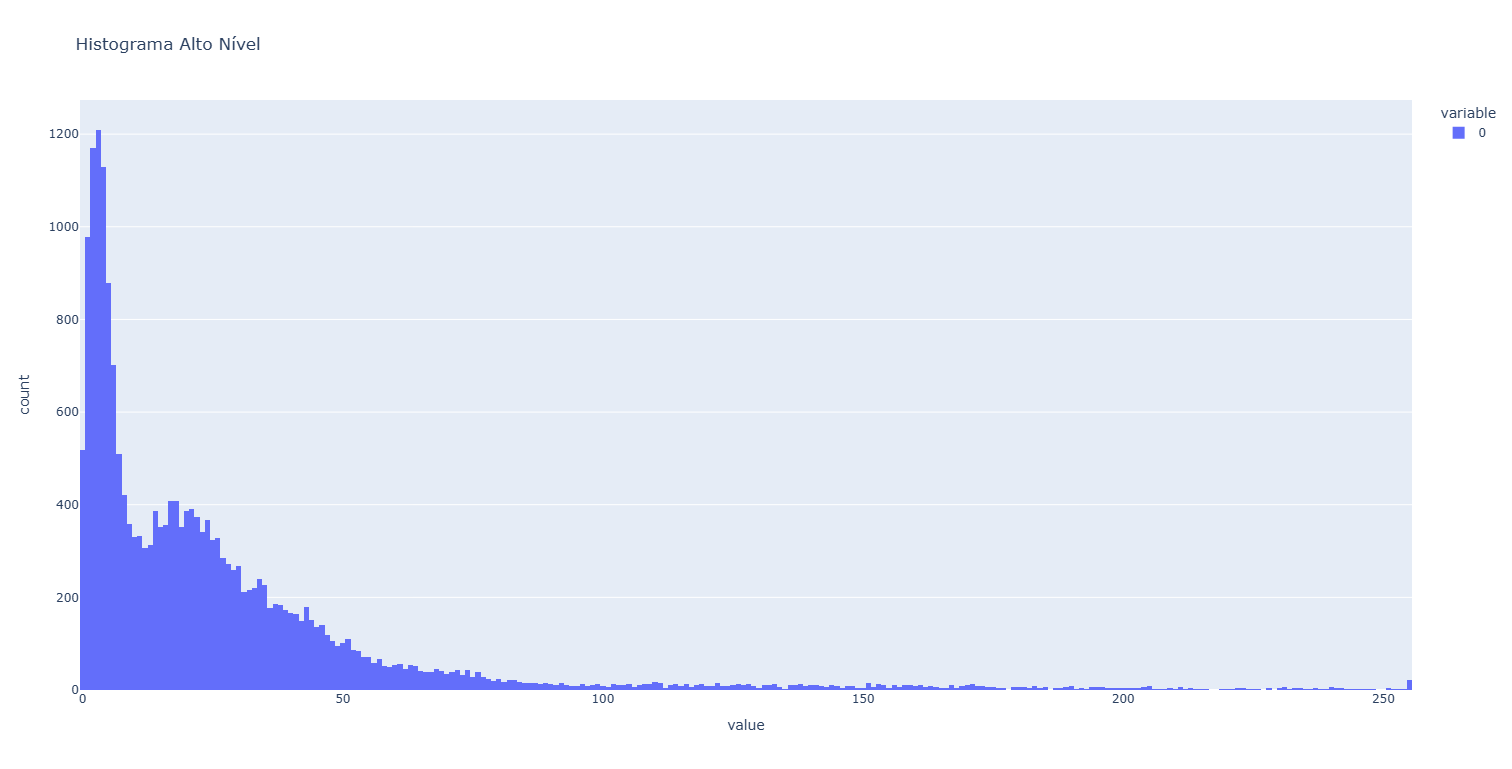
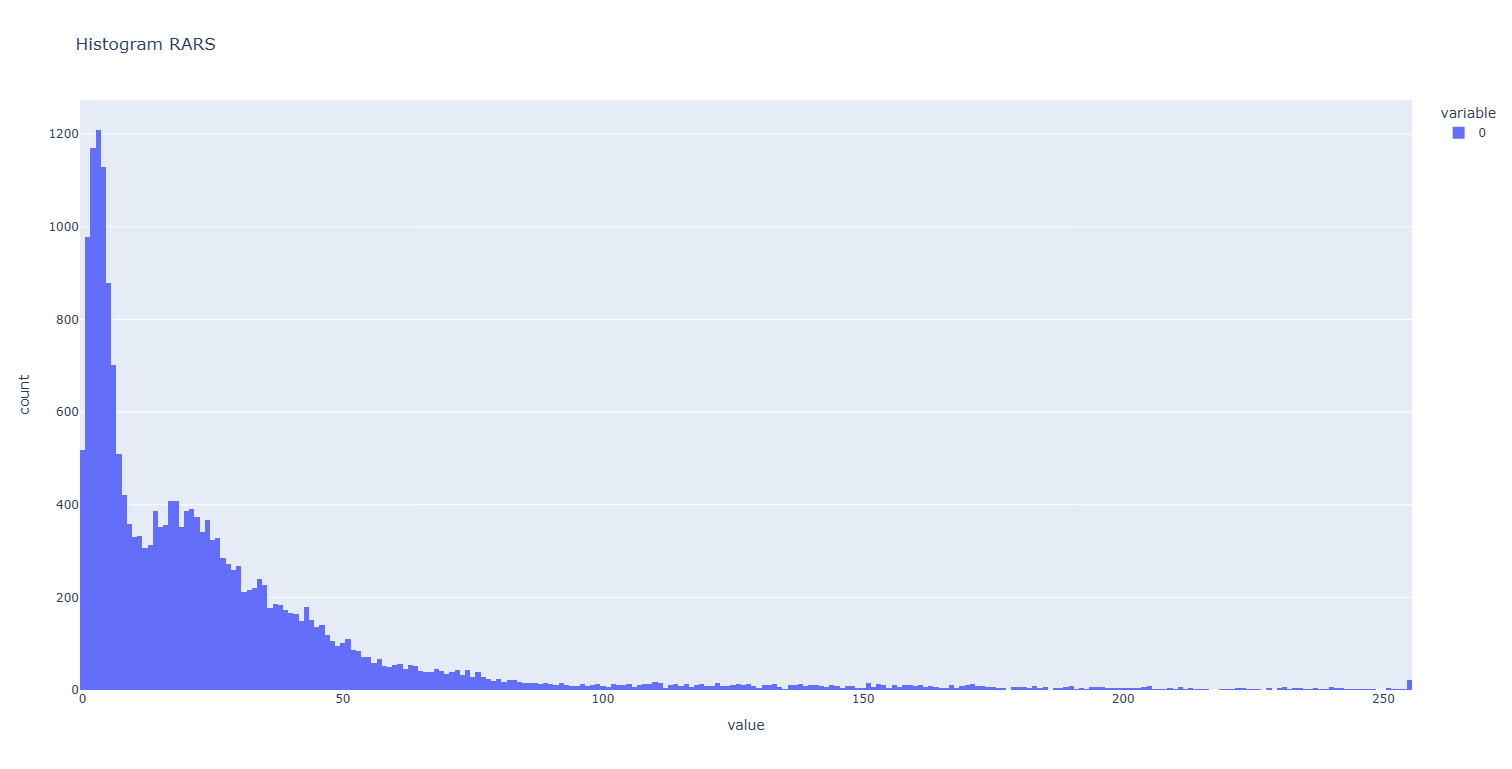
Lê os dados gerados pelo programa em Assembly e recria a distribuição para comparação.

Exibe dois gráficos de histograma: um com os dados em alto nível, outro com os dados obtidos da execução Assembly.

## **3. Resultados**

### **3.1 Comparação Visual**

Os histogramas gerados pelas duas abordagens foram visualmente comparados. Ambos apresentam distribuição idêntica, confirmando que a lógica de contagem em Assembly reproduziu corretamente os dados da imagem.

### **3.2 Análise**

A implementação em Assembly mostrou-se mais trabalhosa devido ao controle manual de memória e ausência de bibliotecas para manipulação de imagens, exigindo a conversão prévia da imagem em dados puros. Em contrapartida, a implementação em Python foi mais ágil e intuitiva, com bibliotecas que facilitam a manipulação e visualização dos dados.

Apesar das diferenças de complexidade, os resultados finais mostraram-se consistentes entre ambas as abordagens.

## **4. Conclusão**

O objetivo de implementar o histograma de uma imagem QCIF em dois níveis de abstração foi atingido com sucesso. A equivalência entre os histogramas confirma a precisão do código Assembly. Essa atividade proporcionou um exercício prático de integração entre hardware e software, com foco na compreensão de estruturas de dados em nível de máquina.