

TP final. Data de entrega 03/12/2021 às 23:55h via colabweb.

Objetivo: implementar uma multiplicação matricial no EduMips64 e estudar a relação entre acesso aos elementos de uma matriz e as possíveis falhas da cache.

Descrição geral

Criar um programa .s para EduMips64 realiza a multiplicação de matrizes $X=Y*Z$ ilustrada na página 77 do livro de Arquitetura, quinta edição do Patterson (primeiro algoritmo da página).

Atividade:

- 1) Criar um programa equivalente ao programa em C apresentado no livro do Patterson em assembly do EduMips que executa a multiplicação entre matrizes quadradas de tamanho **10x10** com elementos inteiros de **32bits** (.word32). As matrizes podem ser inicializadas manualmente na memória com valores inteiros aleatórios positivos ou negativos. **Comentar cada linha do programa e explicar o funcionamento de forma geral.** Indicar a região da memória do edumips onde o professor pode verificar que a matriz resultante está armazenada. As matrizes devem ser organizadas por linhas na memória e não por coluna (abordagem similar a criar uma matriz usando vetores linha em C). 4pts
- 2) **Discutir e explicar o desempenho do programa**, apontando todos os possíveis hazards que possam existir (**os três tipos de hazards**, estruturais, de dados e de controle). **Ilustrar o funcionamento do pipeline** de pelo menos uma iteração do loop mais externo. Calcular a **CPI** do programa. 2pts
- 3) Estudar a seção chamada **“Oitava otimização: otimizações de compilador para reduzir a taxa de falta”** referente a hierarquia de memória apresentada no livro e responder:
 - a) Quais são as considerações do ponto de vista do programador que limitam o desempenho da cache quando a memória é acessada por linha ou por coluna de uma matriz? Ilustrar como os dados são organizados na memória do computador e como são enviados e transferidos para a cache. Explicando quando e como acontecem as falhas. 1pts
 - b) Explicar porque ocorrem falhas da cache nos algoritmos da página 76 na seção de permuta de loops. Ilustrar os conceitos de localidade temporal e espacial considerando uma hierarquia de memória com unicamente uma cache L1. 1pts
 - c) Explicar como o algoritmo por blocos (página 77) funciona e porque melhora o desempenho em relação à multiplicação tradicional. Ilustrar os acessos à memória. 1pts
 - d) Qual seriam as taxas de miss-cache se considerarmos hipoteticamente que existe uma memória cache L1 de mapeamento direto com três blocos de tamanho 40 bytes e executarmos os dois algoritmos da página 77 em matrizes 10x10 com dados do tipo inteiro de 32 bits? 1pts

Entregar num arquivo .zip com o nome do aluno(a) o programa em .s e um relatório com as respostas.

O pdf do livro pode ser encontrado neste link

<https://drive.google.com/file/d/1aleu-jwZGUyGw3o3yu7jsYU6vNK6I74a/view?usp=sharing>

Dica: os elementos das matrizes na memória do Edumips podem ser acessados utilizando a conta tradicional que é aprendida em AED 1, por exemplo $k=i*n+j$. A figura seguinte ilustra como os elementos devem ser organizados por linha na memória.

Matrizes

Para que o compilador identifique o espaço de memória associado a determinado elemento $m[i][j]$, é feita uma **conta de endereçamento**: o elemento com índices i e j é armazenado no endereço k da memória associado à matriz, onde $k = i*n+j$.

Supondo $mat[m][n] \rightarrow m=3$ e $n=4$

