**Lista de Memória  
Infra-Estrutura de Hardware**

**Grupo:  
Antonio Carlos da Silva Junior(acsj)  
Bruno Florencio Pinheiro(bfp)  
Danilo Torres Ferreira(dtf)  
Edilson Ferreira da Silva(efs2)**

**Felipe Farias**

1- **Analise o código cuidadosamente de forma que você entenda em qual ordem os elementos da matriz são chamados. Explique como este posicionamento se reflete na memória principal .**

Os elementos das matrizes são chamados dentro das sub rotinas. Na primeira, os elementos são chamados variando-se, primeiramente, as colunas e, em seguida, as linhas.Já na segunda, os elementos são chamados variando-se, primeiro as linhas e depois as colunas.  
Há uma grande vantagem em variar primeiramente as colunas. Tal vantagem se dá pois ocorre o principio da localidade espacial, visto que, elementos de uma mesma linha encontram-se de maneira consecutiva na memória principal

2-  **Crie um novo projeto, digite (copie) o código C acima, compile-o e carregue para o simulador. Execute o programa no simulador e analise como a memória cache de instruções funciona. Considere uma cache de 32 palavras com tamanho de bloco igual a 4. A seguir, responda as seguintes questões:**

**a. Como está configurado o endereço de 32 bits usado na memória cache?  
b. O que ocorre quando há uma falta na cache?  
c. O que ocorre quando há um acerto na cache?  
d. Qual a função da tag ou rotulo de memória?**

a)

[0...1]: Deslocamento da palavra

[2...3]: Bits usados para escolher a palavra na memória cache.

[4...6]: Bits usados para escolher o bloco na memória cache.

[7...31]: TAG -Bits usados para representar a palavra no bloco.

b)

1-O processador é paralisado;

2-O controlador da cache solicita a leitura na memória principal do endereço físico da informação;

3-Quando a informação é retornada pela memória principal, ela é armazenada na memória cachê e o processador volta a executar normalmente, partindo da instrução que tinha levantado a falta.

c)

O dado é, imediatamente, lido da cachê e, logo em seguida, enviado ao processador.Quando um dado já está na cachê, é eliminada a penalidade por falta, e o dado não precisa ser lido da memória principal

d)

O rótulo é uma fração do endereço que é utilizado para a localização de uma informação na cache. Essa localização é feita comparando o rótulo fornecido com a palavra que está na cache.

**3- Os parâmetros da memória cache podem ser alterados para testar os efeitos dos diferentes casos. Investigue os efeitos dos diferentes parâmetros de configuração.**

**a. Explique de forma concisa os seguintes parâmetros: tamanho da cache, tamanho do bloco da cache, número de conjuntos de cache, políticas de escrita e políticas de substituição.**

**b. Se a cache for grande suficiente tal que todo o código dentro do loop caiba na cache, quantas faltas de cache irão existir durante a execução do loop? Isto é bom ou ruim?**

**c. Que estilo deve apresentar o código para tirar melhor proveito de uma cache com blocos grandes?**

a)

-Tamanho da cachê: O tamanho da cachê é a capacidade de dados q podem ser armazenados em uma cachê. Geralmente medido em palavras ou bytes.

-Tamanho do bloco: O tamanho do bloco da cachê é o número de palavras que cada entrada pode acomodar.

-Número de conjuntos da cachê: O número de conjuntos da cachê indica o grau de associatividade da cachê ,ou seja, a quantidade de grupos de blocos q podem ser localizados em paralelo

-Políticas de escrita: Existem duas possíveis maneiras de se escrever na cachê, a write-throught e a write-back. Na write-throught, a medida q num dado é modificado na cachê ele é atualizado na memória principal. No write-back, a atualização da memória principal só ocorre quando há a necessidade de se substituir um bloco na cachê

-Políticas de substituição: A mais usada é a utilização do algoritmo LRU. A idéia do LRU é que a substituição se dará trocando o bloco que foi referenciado a mais tempo pelo desejado.

b)

Nenhuma. Pois todo o código estará na cachê quando requisto. Isso é muito bom, pois elimina o tempo de tratamento de falta, fator que atrasa muito a execução de um programa.

c)

Uma cache com blocos grandes tem o benefício do princípio da localidade. Devido a isso, um código sem muitos desvios(ex: jump) melhora o proveito dessa cache.

**4- Compile o programa em C com a opção de otimização high (No MipsIt, vá em Project, opção Settings, Optimization Level : 3 High). Execute e analise a cache de dados. Faça uso de uma cache de 32 palavras, com tamanho de bloco 4 e mapeamento direto. Estude as subrotinas SumByColRum e SumByRowCol. Explique cuidadosamente em qual ordem os endereços de memória são visitados pelas duas subrotinas.**

**Execute o programa e analise quantos acertos de cache cada subrotina possui. Existe alguma diferença entre elas? Por que?**

**SumByColRow:**

A idéia dessa sub-rotina é, para cada coluna visitada na matriz, todas as linhas serem visitadas. Devido a essa idéia, ocorrerão visitas a blocos consecutivos. Por exemplo, se última posição visitada foi L[0]C[0] a seguinte será L[1]C[0], ambas de blocos diferentes.

Acertos

D-Cache: 0,03 I-Cache: 1,00

**SumByRowCol:**

A idéia dessa sub-rotina é, para cada linha visitada na matriz, todas as colunas serem visitadas. Devido a essa idéia, ocorrerão visitas a todos os elementos de um bloco antes de se mudar de bloco. Por exemplo, se a última posição visitada foi L[0]C[0} a seguinte será L[0]C[1], ambas no mesmo bloco.

Acertos

D-Cache: 0,75 I-Cache: 1,00