Arquitetura de Sistemas Computacionais

Trabalho 1 - 2022

Prof. Rodrigo Machniewicz Sokulski

Valor

O trabalho tem peso de 2 na nota bimestral do primeiro bimestre (do total de 5 pontos).

Quantidade de alunos

O trabalho deve ser desenvolvido individualmente (1 aluno) ou em duplas (2 alunos).

Data de entrega e arquivos pedidos

A data limite para a entrega é dia 26/09/2022 às 23:59.

O trabalho deve ser desenvolvido na linguagem C, e deve ser compilado utilizando as flags -Wall e -Werror.

Os arquivos .c e .h do projeto devem ser colocados em um arquivo tar.gz, que deve ser entregue pelo Blackboard.

Junto aos arquivos .c e .h, também deve ser enviado um relatório em pdf descrevendo as estruturas de dados e funções utilizadas, assim como o objetivo das estruturas e funções mais importantes.

- O objetivo do relatório é explicar a implementação realizada de forma que não seja necessário abrir o código fonte desenvolvido.
- Seja sucinto no relatório, colocando apenas os pontos e decisões importantes durante o projeto.
- Coloque instruções de como seu projeto deve ser compilado.

Defesa

A defesa será realizada dia 27/09/2022, durante o horário de aula.

Compilação

O trabalho deve ser desenvolvido na linguagem C, e deve ser compilado utilizando as flags -Wall e -Werror.

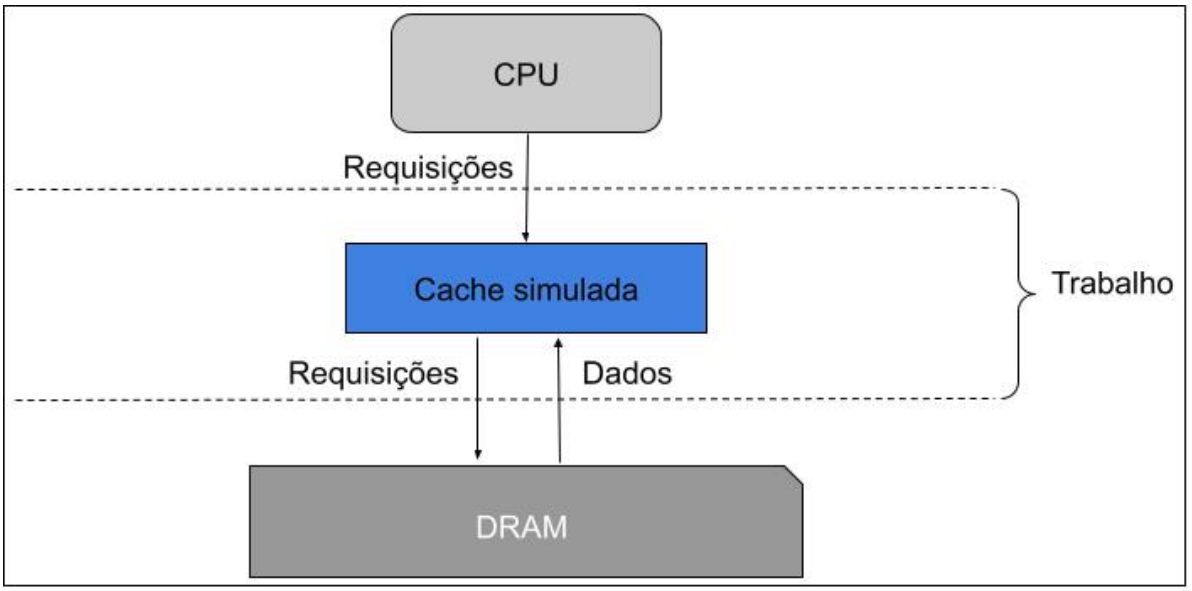
Seu trabalho deve poder ser compilado nos computadores da Positivo.
Durante a compilação não podem ser gerados warnings.

- Cada warning representa a perda de 0.1 ponto na nota do trabalho.

Não devem ser utilizadas flags para a supressão de warnings, caso sejam utilizadas, elas serão removidas antes da compilação para a defesa.

Descrição

- O trabalho consiste na implementação de um simulador de memória cache.
- A memória cache simulada terá apenas 1 nível, recebendo requisições do processador e realizando acessos à memória principal para a obtenção de blocos de dados, como ilustrado na figura a seguir.



- Esse simulador deve armazenar os endereços presentes na cache bem como o tempo total para a resposta das requisições.

Consideramos que, **ao fim da execução**, o sistema é desligado, assim **é** necessário atualizar a DRAM com escritas feitas à cache, caso os dados delas não tenham sido escritos na DRAM.

Entrada

>> Como entrada, seu programa receberá como parâmetro o nome de um arquivo.

A **primeira linha** deste arquivo contém:

- Número inteiro **S**, indicando o tamanho da cache simulada, em bytes
- Número **B**, indicando o tamanho de cada bloco da cache.
- Tempo de acesso à cache (**Tc**)
- Tempo de leitura da DRAM (**TRM**)
- Tempo de escrita à DRAM (**TWM**)

Cada um dos valores é separado por um espaço.
A **segunda linha** deste arquivo contém um número inteiro **N**, indicando o número de requisições da CPU que devem ser simuladas.

As **N linhas seguintes** contêm, cada uma, uma requisição.

Cada requisição é caracterizada por 3 valores, separados por um espaço:
Ciclo Tipo Endereço

- Ciclo: Ciclo de clock em que a requisição de leitura ou escrita chegou na cache.
- Tipo: Tipo de Requisição, **W** indica uma escrita, **R** indica uma leitura.
- Endereço: Número inteiro, de 64 bits, indicando o endereço de memória escrito ou lido pela requisição

As requisições são apresentadas em ordem, assim o ciclo de cada requisição seguinte sempre será maior que o da anterior.

Exemplo de entrada:

```
4 1 1 300 500
3
100 R 10057
130 R 10061
180 W 10062
```

Saída

A saída esperada contém:

- Parâmetros específicos da cache simulada
- Total de ciclos para a resposta da cache às requisições
- Conteúdo da cache ao fim da simulação

A saída deve ser impressa na saída padrão (stdout).

A **primeira linha** deve conter os parâmetros da cache, separados por um espaço:

1. Tamanho da cache (apenas número de Bytes)
2. Associatividade
3. Tamanho de cada bloco (apenas número de Bytes)
4. Política de substituição (FIFO, LRU)
5. Mecanismo de coerência de escritas (WT, WB)
6. Política de alocação durante escritas (WA, WNA)
7. Tempo de acesso à cache (Tc)
8. Tempo de leitura da DRAM (TRM)
9. Tempo de escrita na DRAM (TWM)

A **segunda linha** deve conter apenas 1 número, indicando o tempo total gasto para responder às requisições.

A **terceira linha** deve apresentar o número de **HITS** e **MISSES** ocorridos na cache, separados por um espaço.

A quarta linha deve apresentar o número **A** de conjuntos associativos presentes na cache.

As **próximas A linhas** contém os endereços armazenados nos **A** conjuntos associativos da cache, ou seja, o endereço que fez cada bloco ser levado à cache.
Os endereços em um mesmo conjunto associativo devem ser separados por um espaço.
Caso uma linha esteja inválida, escreva a palavra NULL.

Descrição da saída (Dê um zoom):

<Tam. Cache> <Associatividade> <Tam. Bloco> <FIFO|LRU> <WT|WB> <WA|WNA> <Tempo acesso cache> <Tempo leitura DRAM> <Tempo escrita DRAM>
<Tempo total para atender requisições>
<Número de HITS> <Número de MISSES>
<A>
<Endereço na via 0 do conjunto 0> <Endereço na via 1 do conjunto 0> ….
<Endereço na via 0 do conjunto 1> <Endereço na via 1 do conjunto 1> ….
….

Exemplo de saída:

4 2 1 LRU WT WA 1 300 500
1403
0 3
2
10062 NULL
10057 10061

Variações específicas de cada aluno

Os parâmetros da cache implementada por cada aluno/grupo variam de acordo com o RGM da seguinte maneira:

- No caso de uma dupla, o RGM utilizado é a soma dos RGMs de cada membro.

Dígito menos significativo do RGM	Associatividade	Política de substituição	Mecanismo de coerência de escritas	Política de alocação durante escritas
0	4	FIFO	WT	WA
1	8	LRU	WB	WNA
2	16	FIFO	WB	WA
3	4	LRU	WT	WNA
4	8	FIFO	WT	WNA
5	16	LRU	WB	WA
6	4	FIFO	WB	WNA
7	8	LRU	WT	WA
8	16	FIFO	WT	WA
9	4	LRU	WB	WNA

Ex:
RGM: 8121101169**9**
Dígito menos significativo: **9**
=> Associatividade: 4
=> Política de substituição: LRU
=> Mecanismo de coerência de escritas: WB
=> Política de alocação durante escritas: WNA

Dúvidas comuns:

Bit de Write Back:

- Quando utilizando a política de Write Back (WB), não implemente o bit para verificar se uma página foi escrita ou não, apenas escreva cada bloco da cache na DRAM assim que for substituído.