Universidade Estadual de Maringá – UEM
Departamento de Informática - DIN

Trabalho Prático – Simulador Extensões Protocolo MESI em uma aplicação

Disciplina: Arquitetura e Organização de Computadores II **Código**: 12029
Curso: Ciência da Computação **Turma**: 02
Professora: Sandra Cossul
Semestre: 2025/02

Obs.: Trabalho baseado na proposta elaborada pelo Prof. Maurilio Martins Campano Junior.

OBJETIVO

Implementar um programa que **simule uma memória cache gerenciada utilizando uma extensão do protocolo MESI** (Modify, Exclusive, Shared, Invalid) como MESIF ou MOESI, implementado em uma **aplicação real.**

DESCRIÇÃO

O trabalho consiste em realizar uma simulação de pelo menos **três processadores**, cada um com um nível de memória cache dedicada e uma memória RAM compartilhada entre os processadores.

O programa deve conter **dois espaços de endereçamento**, um para simular a **memória principal** e outro para simular a **memória cache** alocada a cada um dos três processadores.

O programa deve solicitar acessos aos dados contidos na memória. O espaço da memória principal deve ter pelo menos 50 posições, e o espaço da memória cache deve ter pelo menos 5 posições. O tamanho do bloco da memória RAM deve ser levado em conta, pois um bloco da memória RAM é alocado a uma linha da memória cache.

As solicitações podem ocorrer em cada um dos processadores, ou seja, o processador 1 pode solicitar um dado da posição X da memória RAM e o processador 2 também pode solicitar dados de qualquer posição da memória.

A representação da memória e das caches devem ser similares as representações abaixo:

Memória principal	
Linha	**Dado**
0	3434
1	24121
2	232
3	5028
4	81092
5	91639
6	54432
...	...

Memória cache – P1	Tags – P1

Memória cache – P2	Tags – P2

O programa no seu início deve preencher o espaço da memória principal com valores aleatórios (numéricos, alfa numéricos, ou outras informações de acordo com a aplicação). Em seguida, o programa deve possibilitar ao usuário selecionar um processador, e realizar a solicitação de um dado da memória principal, sendo essa solicitação com ou sem alteração de seu valor (leitura ou escrita).

A função de mapeamento da memória cache deve preencher as linhas em ordem aleatória (mapeamento associativo - cada bloco da memória principal pode ser carregado em qualquer linha da cache), o **algoritmo de substituição** a ser utilizado deve ser o **FIFO** e a **política de escrita** deve ser o **write-back**.

Quando a solicitação é realizada, cada linha da memória cache pode estar associada aos valores de acordo com o **protocolo MESI** (modify – exclusive – shared – invalid –> modificada, exclusiva, compartilhada e inválida) **e o estado adicional** (a depender da extensão do protocolo MESI escolhido – MESIF ou MOESI). Esta informação deve estar visível ao visualizar a representação da memória cache.

Os seguintes tipos de transações devem estar definidos:

- **RH (read hit – leitura com acerto)** – quando a leitura de um dado é solicitada e este já está na cache do processador solicitante – (o estado é mantido – se está modificado fica modificado, se está compartilhado fica compartilhado, e se está exclusivo fica exclusivo).
- **RM (read miss – leitura com falha)** – quando a leitura de um dado é solicitada e este não está na cache do processador solicitante – (se o dado está presente em outra cache a linha é marcada como compartilhada, e se o dado não está presente em outra cache é marcado como exclusiva).
- **WM (write miss – escrita com falha)** – quando a escrita de um dado é solicitada e este dado não está na cache do processador solicitante - (após a linha ser carregada é marcada como modificada, se a linha estiver presente em outra cache, é marcada como inválida na outra).
- **WH (write hit – escrita com acerto)** – quando a escrita de um dado é solicitada e este dado está na cache do processador solicitante – (se a linha é compartilhada muda-se para modificada e todas as outras para inválida, se a linha é exclusiva muda-se para modificada, e se a linha já está marcada como modificada o seu estado é mantido)

Ao executar qualquer solicitação da memória deve ser destacado se ocorreu um RH, RM, WM ou WH, destacando ainda o **estado da linha**.

O campo TAGS da memória cache deve armazenar as informações relevantes sobre a linha da memória cache, como a posição correspondente da memória RAM (bloco) e as informações referentes a extensão do protocolo MESI. Além disso estas informações NÃO DEVEM estar armazenadas na memória RAM, somente na memória cache.

- **O simulador deve estar inserido em uma aplicação real qualquer**, tal como um **jogo**, **um sistema de reserva/compra de passagens, de bilhetes de cinema etc.** Cada grupo deve escolher a sua aplicação, e a mesma deve ser diferente das outras equipes. Assim, é necessário indicar no link dos temas a aplicação a ser desenvolvida.

Assim, as funções da aplicação devem definir leituras e escritas em dados compartilhados, tal como exemplo um jogo com um mundo compartilhado, que pode ser acessado por mais de um usuário. No jogo por exemplo, algumas destas informações compartilhadas podem ser lidos e/ou escritos e a gerência destas leituras e escritas é feita com base em uma extensão do protocolo MESI.

Na aplicação, o usuário deve ser capaz de selecionar "um processador" (P1, P2 ou P3), escolher a operação (leitura ou escrita), e qual informação deseja solicitar, sendo exibido então as informações associadas a extensão do protocolo MESI.

Deve ser possível visualizar ou solicitar o estado da memória principal e de cada cache a qualquer momento ao longo da execução da aplicação (em tempo real).

- Extra (opcional): a aplicação pode incluir um log de eventos, com o histórico das operações. Isso ajuda na demonstração e entendimento do protocolo.

INSTRUÇÕES GERAIS

- O trabalho poderá ser realizado de forma **individual**, em **duplas** ou **trios.**

- Serão aceitos trabalhos nas seguintes **linguagens de programação**: C, C++, C#, Pascal, Delphi, Java, JavaScript, Python.

- O código fonte deve estar **comentado** em suas partes principais, e **bem estruturado**.

- Cada dupla deve entregar **três arquivos**, um **relatório técnico**, o(s) **arquivos fonte do programa**, e um **arquivo executável**, bem como as **instruções para compilar e rodar o programa**.

- A **entrega** será realizada **pelo Classroom** até as 23:59 do dia **30/11/2025**, com data da apresentação prevista para os dias 01/12/2025, 03/12/2025 e 08/12/2025.

- Coloque seu nome (ou nome da dupla) no nome do arquivo. Um integrante do grupo fazer a entrega é suficiente. O outro integrante pode enviar um comentário: "Entregue por tal pessoa".

- O relatório deve conter:
    - Introdução
    - Objetivos/Justificativa
    - Funcionamento da memória cache
    - Funcionamento do protocolo MESI e extensão
    - Funcionamento da aplicação (regras de negócio)
    - Decisões de projeto para a implementação
        - Estruturas de dados utilizadas
        - Gerenciamento do protocolo escolhido
        - A aplicação (questões relativas à implementação)
    - Conclusão (destacar o aprendizado sobre coerência de cache)
    - Referências

- O trabalho vale **nota 10**, correspondendo a **terceira avaliação periódica** (que prevê Trabalhos Práticos), conforme critério de avaliação da disciplina, seguindo o detalhamento abaixo:
    - Código fonte - 0 a 6,0
    - Relatório – 0 a 2,0
    - Apresentação - 0 a 2,0

- A apresentação será somente para a professora, que solicitará a execução de comandos e acessos à aplicação.

- **Pontos de avaliação**:
    - implementação correta
    - uso na prática dos conceitos discutidos em aula
    - qualidade e originalidade da implementação
    - estrutura do código

- **Não serão avaliados os trabalhos que**:
    - forem entregues fora do prazo (ou por email);
    - não tiverem identificação;
    - não atendam as especificações deste documento;
    - for identificada cópia (nesse caso, ambos os trabalhos copiados serão desconsiderados).