Universidade Federal do Rio Grande do Norte
Instituto Metrópole Digital
IMD0041 – Introdução a Organização e Arquitetura de Computadores
Descrição do Simulador

- Projeto da Unidade III
- Valor: 4,0
- Atividade INDIVIDUAL
- Prazo de entrega: 17/06

# Simulador de Hierarquia de Memória

1. Descrição

Implementar uma ferramenta que simule o comportamento interno de uma cache L1 e da memória principal. O simulador recebe como entrada uma sequencia de comandos que podem ser de leitura ou escrita e o endereço solicitado. A cache simulada deve verificar a presença ou não do bloco contendo o dado solicitado (acarretando em um Hit ou Miss). O simulador deve ser configurável no que diz respeito aos aspectos de projeto de cache discutidos em sala: Mapeamento, Politica de Substituição e Política de Escrita.

1.1. Mapeamento:

O mapeamento escolhido pode ser de 3 tipos: Direto, Totalmente Associativo e Parcialmente associativo. Dependendo do mapeamento escolhido e após receber o comando de leitura ou escrita o simulador deve informar o resultado da operação (hit ou miss) e a linha da cache que contém o bloco solicitado.

1.2. Política de Substituição:

O algoritmo de substituição de blocos podem ser de 4: Aleatório, FIFO, LFU e LRU. Dependendo da política de substituição escolhida e após receber o comando de leitura ou escrita o simulador deve adicionalmente informar se houve substituição e qual bloco foi retirado de que linha da cache.

1.3. Política de Escrita:

Existem 2 políticas de escrita disponíveis: Write-back e Write-through. Ao ser realizada uma leitura, o simulador deve mostrar que bloco foi modificado e qual o novo valor.

2. Comandos

O simulador aceita como entrada 3 comandos: *read*, *write* e *show*. O comando *read* recebe como parâmetro o endereço a ser lido. Como resposta o simulador deve informar o resultado (hit ou miss) e a linha da cache que contém o bloco solicitado. O comando *write* funciona de maneira similar, entretanto além do endereço do dado a ser escrito ele tem como parâmetro o novo valor a ser escrito. Por fim, o comando *show* mostra o conteúdo de toda a cache e de toda a memória principal.

3. Arquivo de Configuração

O simulador precisa gerenciar uma memória cache com características pré-definidas. Em um arquivo de configuração (config.txt) devem ficar armazenadas as escolhas do usuário para as seguintes características (nesta ordem). No início da simulação este arquivo é lido e suas configurações entram em vigor durante a execução.

*Tamanho do bloco (em número de palavras)*
*Numero de linhas da cache*

*Numero de blocos da memória principal*
*Mapeamento (1 – Direto; 2 – Totalmente Associativo; 3 – Parcialmente Associativo)*
*Numero de conjuntos (caso não seja Parcialmente Associativo, ler o valor normalmente mas desconsidere-o)*
*Política de substituição (1 – Aleatório; 2 – FIFO; 3 – LFU; 4 – LRU)*
*Política de Escrita (1 – Write-back; 2 – Write-Through)*

4. Exemplo de arquivo de configuração

| **Arquivo** *config.txt***:** | **O que significa:** |
|---|---|
| *4* | *Cada bloco contém 4 palavras* |
| *8* | *A cache tem 8 linhas (ou seja, comportam até 8 blocos vindos da memória)* |
| *16* | *A memória principal tem 16 blocos* |
| *1* | *Mapeamento Direto* |
| *2* | *Não significa NADA pois o mapeamento não é parcialmente associativo* |
| *4* | *Política de substituição LRU* |
| *1* | *Política de escrita Write-back* |

5. Exemplo de funcionamento:

| **Entrada:** | **Saída:** |
|---|---|
| *Command> Read 25*<br>*Command> Read 28*<br>*Command>Write 25 41*<br>*Command> Show* | *Read 25 -> HIT linha 2*<br>*Read 28 -> MISS -> alocado na linha 3 -> bloco 7 substituido*<br>*Write 25 -> HIT linha 2 -> novo valor do endereço 25=41*<br>*Show ->*<br>*CACHE L1*<br>*Linha –Bloco-Endereço -Conteúdo*<br>*0 – 4– 16 – 23*<br>*0 – 4 – 17 - 32*<br>*0 – 4 – 18 – 0*<br>*0 – 4 – 19 – 12*<br>*1 – 1 – 4 - 7*<br>*1 – 1 – 5 - 12*<br>*1 – 1 – 6 – 12*<br>*1 – 1 – 7 - 7*<br>*2 – 6 – 24 – 0*<br>***2 – 6 – 25 - 41***<br>*2 – 6 – 26 - 0*<br>*2 – 6 – 27 – 0*<br>*3 – 7 – 28 - 3*<br>*3 – 7 – 29 - 99*<br>*3 – 7 – 30 - 0*<br>*3 – 7 – 31 - 1*<br><br>*MEMORIA PRINCIPAL*<br>*Bloco-Endereço -Conteúdo*<br>*0 – 0 – 7*<br>*0 – 1 - 2*<br>*0 – 2 – 0*<br>*0 – 3 – 0*<br>*1 – 4 – 7*<br>*1 – 5 – 12*<br>*1 – 6 – 12*<br>*1 – 7 – 7*<br>*2 – 8 – 0*<br>*2 – 9 – 0*<br>*2 – 10 – 0*<br>*2 – 11 – 0*<br>*3 – 12 – 0*<br>*3 – 13 – 0*<br>*3 – 14– 0*<br>*3 – 15 –1*<br>... |

6. Instruções para o desenvolvimento do trabalho
   6.1. O que deve ser enviado?
   - Arquivos de código fonte contendo tudo necessário para a execução do programa.
   - Documento especificando a função de cada arquivo e como executá-lo.

   6.2. Como será testado:
   Todas as ferramentas serão testadas com um caso de teste criado especificamente para avaliar a corretude. Se não funcionar ou a leitura/escrita de dados ou a exibição estiver errada, a nota tende a **zero**.

   6.3. Considere:
   - A memória cache simulada diz respeito somente a uma cache de dados.
   - O CONTEÚDO das memórias não importa.
   - O endereçamento da memória é por PALAVRA e não por BYTE.
   - As solicitações (entrada do sistema) são feitas considerando o ENDEREÇO. Entretanto, a transação entre cache e memória principal é por meio de BLOCOS.

7. **IMPORTANTE**: o professor reserva-se ao direito de modificar partes deste documento estabelecendo novos requisitos e demandas para o trabalho.