



Trabalho Prático Final

SSC0114 - Arquitetura de Computadores

Cezar Guimarães (Nº 9763209)

Leonardo Moreira Kobe (Nº 9778623)

Matheus Carvalho Raimundo (Nº 10369014)

Pedro Pastorello Fernandes (Nº 10262502)

Orientadora: Prof. Sarita Mazzini Bruschi

USP - São Carlos

Dezembro/2018

Índice Analítico

1. Introdução	2
2. Desenvolvimento	2
3. Guia de Execução	2
3. Utilizando a Interface Gráfica	3
4. Conclusão	8
5. Bibliografia	8

1. Introdução

O objetivo desse documento é descrever o projeto que foi implementado como trabalho prático final da disciplina de Arquitetura de Computadores. O projeto implementado é um **REA** (Recurso Educacional Aberto) que funciona como **simulador** de protocolos de coerência de cache vistos em sala de aula (VI, MSI e MESI).

A simulação é feita em interface gráfica, mostrando visualmente todas as alterações nas memórias cache, RAM, e no barramento de uma arquitetura abstrata.

2. Desenvolvimento

O projeto realiza simulação de execução de três protocolos de coerência de cache multiprocessadores: **MSI (Write-back e Write-invalidate)**, **MESI (Write-back e Write-invalidate)** e **VI (Write-through e Write-invalidate)**. Sua interface foi implementada por meio da Web e, para tal, as tecnologias utilizadas foram:

- Linguagens de marcação front-end, **HTML** e **CSS**;
- Linguagem de programação **JavaScript**;
- Bibliotecas da linguagem **jQuery** e **jQueryUI**.

Além disso, durante o desenvolvimento do projeto, o sistema de versionamento **Git** também foi utilizado.

3. Guia de Execução

O trabalho pode ser executado abrindo-se o arquivo principal (*index.html*) em qualquer navegador moderno (botão direito do mouse → abrir com → escolher navegador). Se o projeto estiver compactado em um arquivo *.ZIP, talvez seja necessário descompactar todos os arquivos em um diretório do computador. Para evitar problemas de compatibilidade, recomenda-se o uso dos navegadores Google Chrome ou Mozilla Firefox, visto que os testes do projeto foram realizados nos mesmos.

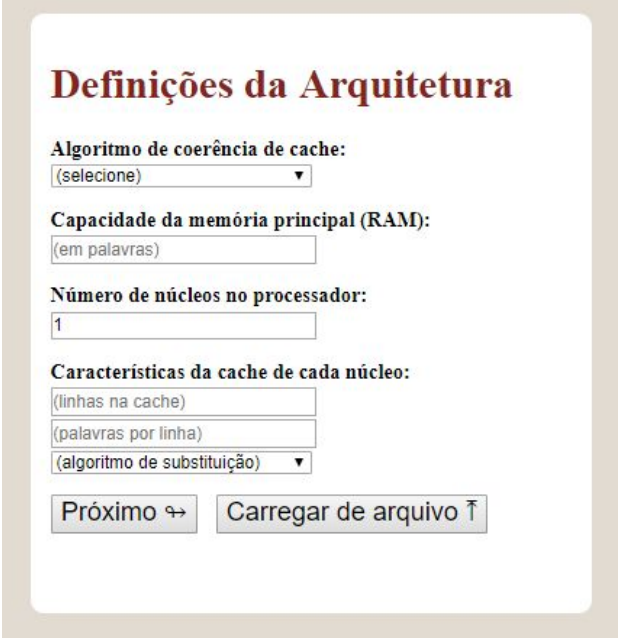
Para a execução, é mandatório que a tela do navegador possua, ao menos, 800:650 pixels de dimensão. Obviamente, também é necessário que o JavaScript esteja habilitado.

4. Utilizando a Interface Gráfica

A interface gráfica é intuitiva e autoexplicativa, mas abaixo seguem algumas noções básicos de uso desta. Ela utiliza muitos efeitos visuais.

4.1 Definindo a Arquitetura

Ao iniciar a aplicação, se todos os *scripts* necessários foram carregados adequadamente, a tela de definição da arquitetura é a primeira a aparecer:



Definições da Arquitetura

Algoritmo de coerência de cache:
(selecione) ▼

Capacidade da memória principal (RAM):
(em palavras)

Número de núcleos no processador:
1

Características da cache de cada núcleo:
(linhas na cache)
(palavras por linha)
(algoritmo de substituição) ▼

Próximo ⇌ Carregar de arquivo ↗

Nela, é possível definir algumas características da arquitetura. Entre elas, a política/algoritmo de coerência de cache (MSI, MESI ou VI), a capacidade da memória principal/RAM (em palavras), o número de núcleos dentro do processador que pode requisitar palavras e a capacidade da memória cache presente em cada um desses núcleos (linhas de cache e palavras por linha).

As seguintes características da arquitetura são pré-definidas e não podem ser modificadas:

- arquitetura simétrica, 32 bits;
- memória endereçada a byte;
- todos os núcleos e a memória utilizam um único barramento.

Por fim, é possível também carregar todas essas definições de um arquivo, podendo assim salvar essas características para uso em futuras simulações sem o trabalho de reinserir as informações. Este arquivo é gerado e pode ser obtido em tempo de simulação (segue abaixo).

4.2 Valores Iniciais da Memória Principal

Para simulação, é necessário inserir alguns valores que vêm de início presentes na memória principal (RAM). Esta é a função da próxima tela:

Memória RAM

Auto-Preencher

Posição	Valor Inicial
0x0	(valor inicial)
0x4	(valor inicial)
0x8	(valor inicial)
0xC	(valor inicial)
0x10	(valor inicial)
0x14	(valor inicial)
0x18	(valor inicial)
0x1C	(valor inicial)
0x20	(valor inicial)
0x24	(valor inicial)

Anterior ← Próximo →

Os valores devem estar entre 0-4294967295, ou seja, devem ser valores válidos que podem ser armazenados em 32 bits de forma a desconsiderar sinal. É possível utilizar um autopreenchimento nesses valores, caso sejam irrelevantes para a simulação, clicando no botão “Auto-Preencher”.

4.3 Requisições dos Núcleos do Processador

A próxima tela permite que as requisições dos núcleos sejam adicionadas:

O núcleo	lê/escreve	no endereço
(N)	(selecione) ▼	(0, 4, ...)
(N)	(selecione) ▼	(0, 4, ...)

(adicionar requisição)

Anterior ← Simular →

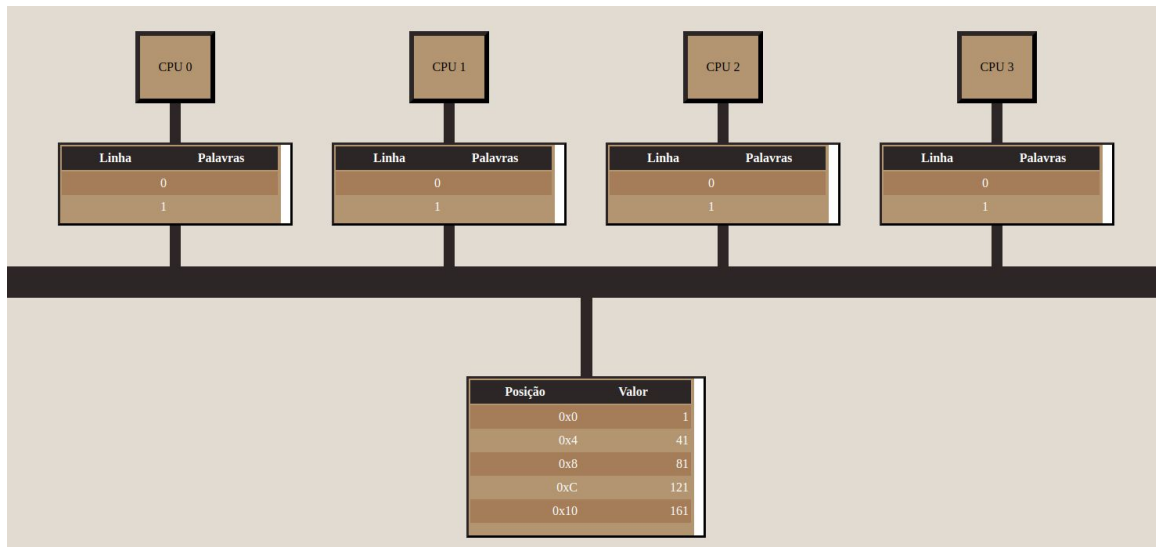
Para adicionar uma requisição, basta clicar em “(adicionar requisição)”. Na tabela, o primeiro campo indica qual núcleo fará a requisição (P0, P1, ...), o segundo qual tipo de requisição (read/write) e o terceiro em qual endereço de memória (0, 4, 8, 12, ...).

O núcleo que faz a requisição deve ser algo entre 0 e aquele número de núcleos definido na primeira tela subtraído de 1 (0, 1, ..., N-1). No caso de uma requisição de escrita (*write*), o valor a ser escrito deve ser algo entre 0 e 4294967295. A memória é endereçada a byte.

É possível reordenar as requisições com as setas para cima e para baixo, e remover determinada requisição caso desejável.

4.4 Simulação

A próxima tela já é a tela de simulação:

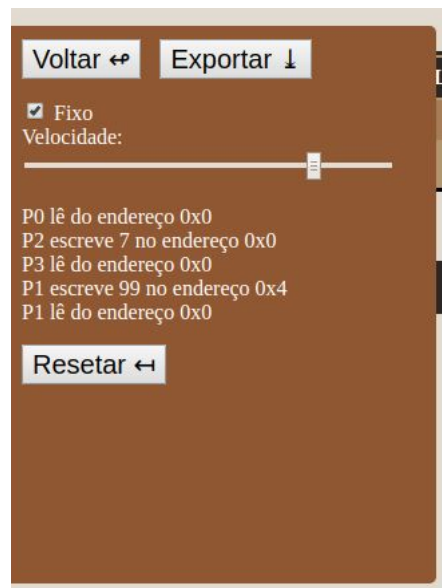


A memória principal (RAM) é exibida no canto inferior (com os valores iniciais definidos na segunda tela), o barramento ao centro da tela, e em cima cada um dos núcleos do processador e suas respectivas memórias cache.

A priori, todas as memórias cache começam com todas as linhas inválidas e sem nenhum valor armazenado.

Caso a tela seja muito pequena para exibir todos os núcleos, é possível usar as setas (canto superior esquerdo e superior direito) para rolar entre os núcleos. Durante a simulação, essa rolagem é feita automaticamente. Caso a tela seja grande o bastante para exibir todos os núcleos, isso não é um problema.

A barra de ferramentas (canto esquerdo) possibilita realizar a simulação:



Nela é possível alterar a velocidade das animações, e se a própria barra de ferramentas deve ficar fixada para exibição ou se esconder quando perder o foco do mouse. Também é possível baixar as definições da arquitetura atual para um arquivo para uso em simulações futuras clicando em “Exportar”. A funcionalidade “Exportar” é compatível com poucos navegadores atuais, mas tem funcionamento testado e garantido nos navegadores recomendados (Chrome e Firefox).

Para realizar a simulação em si, clique na primeira requisição. Visualmente, será mostrado tudo o que acontecerá, de acordo com os algoritmos escolhidos. Para continuar, clique na segunda requisição, e assim por diante. Para restaurar a arquitetura ao seu estado inicial, basta clicar em “Resetar”. A partir de uma determinada requisição, não é possível retornar ao estado anterior (voltar uma requisição) sem antes restaurar a arquitetura ao seu estado inicial, mas é possível avançar dentre as requisições o quanto for necessário. Enquanto as animações de uma simulação estão sendo feitas, não é possível pular para outra requisição ou restaurar a arquitetura.

5. Conclusão

Com o desenvolvimento deste projeto, conclui-se que diferentes políticas de coerência de cache podem afetar significativamente o funcionamento de uma arquitetura.

Ao testar diferentes políticas de coerência de cache para uma mesma arquitetura, nota-se que dependendo da forma como as requisições são feitas, uma pode ser mais rápida ou mais lenta que outra. Além disso, em situação real deve-se considerar também que os processadores devem concorrer pelo próprio barramento para estas requisições.

6. Bibliografia

- Em geral, apenas o conteúdo e anotações de sala de aula foram utilizados no desenvolvimento deste projeto.
- O website <<https://www.w3schools.com/>> foi utilizado para relembrar conceitos de web.