Centro Federal de Educação Tecnológica de Minas Gerais
Departamento de Computação – Laboratório de arquitetura e organização de computadores II

Relatório da Prática 1

Ulisses Andrade Carvalho

27 de agosto de 2023

Forma de onda parte I

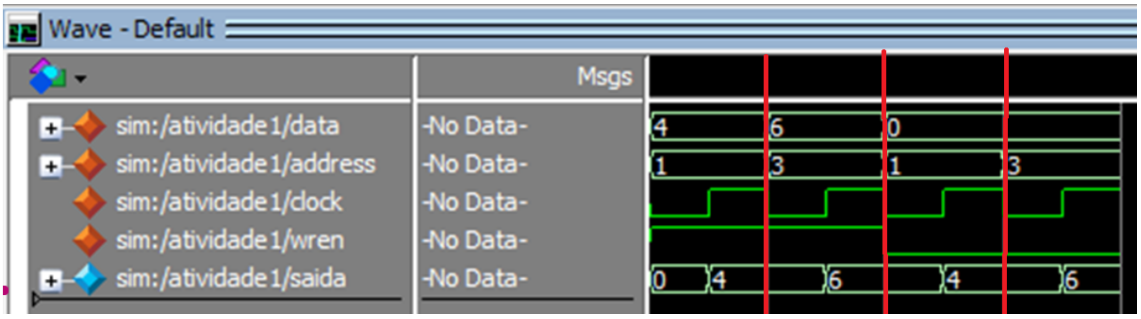

IMAGEM 1: Parte 1 - Simulação de escrita e leitura da memória.

A imagem acima foi separada em 4 regiões sequenciais.

A primeira se refere ao momento que será gravado na memória, na posição 1, como mostrado pelo sinal **address**, o valor 4, presente no sinal **data**. Como o sinal **wren** está com valor 1, o valor 4 foi efetivamente gravado na posição 1. Como mostrado no sinal de saída, após o dado ser gravado na memória, no momento em que há uma borda de subida, o valor presente na posição 1 da memória é lido e exibido no sinal de **saída**.

A segunda região simula um comportamento semelhante ao da região anterior, porém o valor 6, como mostrado no sinal **data**, é armazenado na posição de memória 3, presente no sinal **address**. A partir do momento que ocorre uma borda de subida, o valor é escrito na memória e exibido no sinal de **saída**.

As terceira e quarta posições tem o sinal **wren** como 0, portanto não ocorrerá escrita, apenas leitura. Isso se torna evidente pois o sinal **data** foi definido como 0 nesse restante de tempo, mas quando ocorre uma borda de subida, não haverá escrita nas posições 1 e 3, apenas leitura dos sinais ali presentes. Portanto, a leitura da posição 1 ocasionou o valor 4, e a leitura da posição 3 resultou no valor 6.

Forma de onda parte II

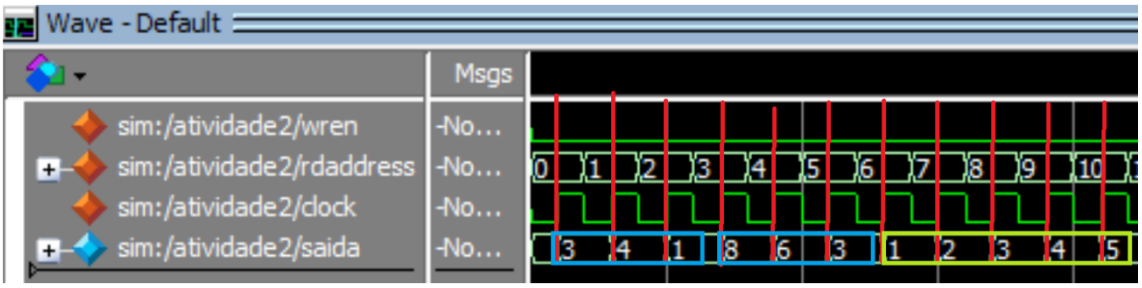

IMAGEM 2: Parte 2 - Simulação de leitura da memória previamente inicializada.

Nessa segunda simulação todos os dados já foram previamente gravados na memória através do arquivo .mif. Assim, o valor do **raddress** foi variado da posição inicial da memória (raddress=0) até o seu final e, a cada borda de subida do clock, o valor da saída recebeu o valor armazenado naquela posição da memória.

Na imagem foi destacado cada borda de subida do **clock**, podendo assim observar a alteração do valor da saída, que é antecedido pela alteração do valor **rdaddress**.

Os três dígitos finais da matrícula de cada aluno foram circulados com a cor azul e os números em ordem crescente foram destacados em amarelo.

Projeto parte III

Palavras: 8 bits

Blocos: 1 palavra

Memoria principal de 32 (2^5) linhas e cache L1 de 4 (2^2) linhas

Mapeamento de Endereços em Memória Cache: Mapeamento Associativo

Endereço 5 bits
Tag (5 bits)

Cache L1 utilizara a política de Write-Back e terá 1 bit de validade, 1 de Dirty, 2 bits de LRU, 5 bits para Tag e 8 bits para o bloco com uma palavra, segue imagem ilustrativa:

Cache de dados 4x17

Validade (1 bit)	Dirty (1 bit)	LRU (2 bits)	Tag (5 bits)	Bloco (8 bits)

Diagrama de módulos

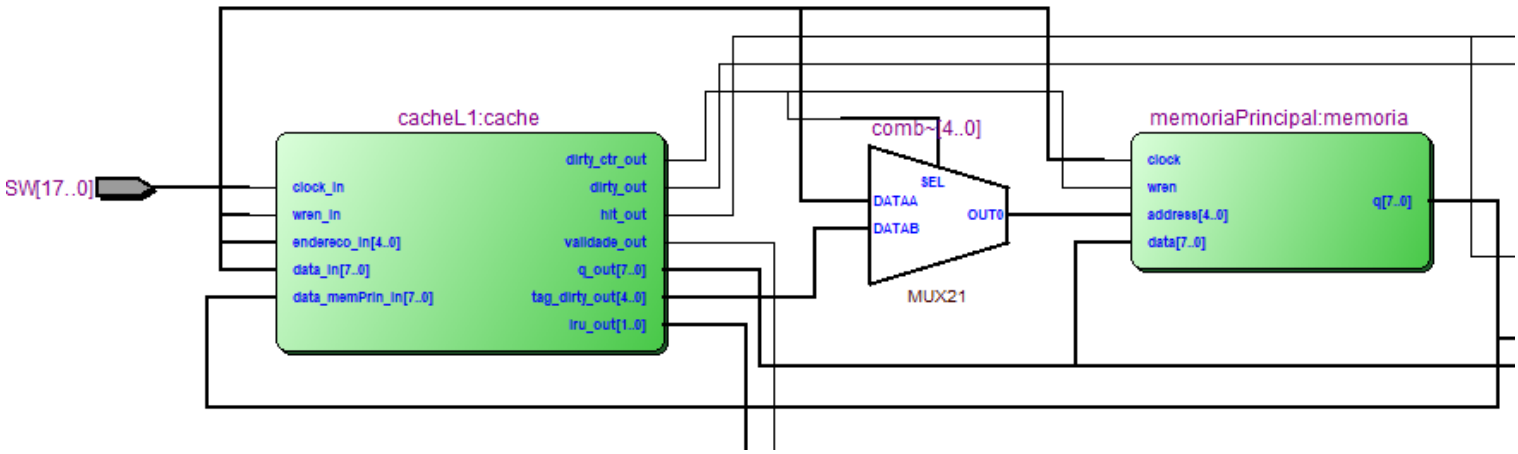

IMAGEM 3: Parte 3 – Primeira parte do diagrama de módulos.

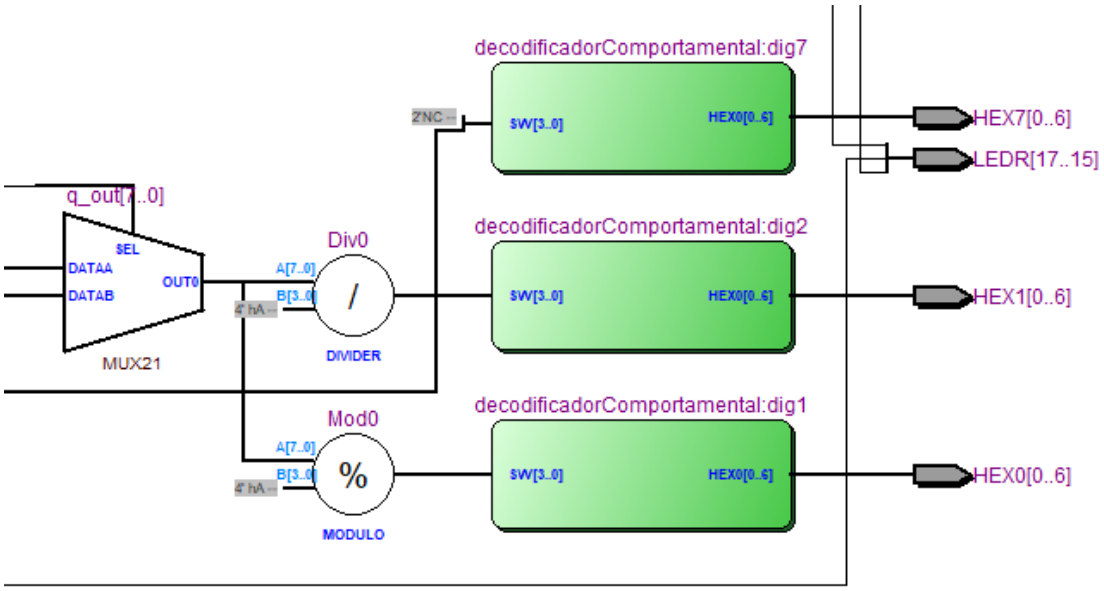

IMAGEM 4: Parte 3 – Segunda parte do diagrama de módulos.

Entradas e saídas do modulo da cacheL1:

- As entradas clock_in, wren_in, endereco_in e data_in são recebidas da placa FBGA através das chaves SW;
- A entrada data_memPrin_in é recebida da memória principal e é usado para trazer um dado da memória para cache no caso em que ocorrer um miss;
- A saída dirty_ctr_out é passada para o wren da memória principal e é usado para controlar o primeiro MUX;
- O dirty_out, hit_out e validade_out são usados para controlar um led na placa, o hit_out também é passado para o segundo MUX para selecionar qual dado lido (cache ou memoria principal) será mostrado nos displays;
- q_out é encaminhado para o segundo MUX que seleciona a saída para ser mostrada na placa FBGA e para a entrada de dados da memória principal para escrita no caso de write-back;
- tag_dirty_out é passada para o primeiro MUX que seleciona qual será o endereço que a memória principal acessara;
- O lru_out é usado para controlar o valor apresentado um display da FBGA;

Modulo memoriaPrincipal:

- clock recebido da placa FBGA
- wren é recebido do sinal dirty_ctr_out do modulo cacheL1;
- address recebe a saída do primeiro MUX, que seleciona se o endereço a ser usado na memória principal será o que veio da placa FBGA ou o tag_dirty_out;
- A entrada data recebe seu valor da saída da cacheL1 q_out, já que a escrita na memória principal ocorre somente de acordo com a política de write-back;
- q é a saída de dados da memória principal, esse sinal é passado tanto para o segundo MUX quanto para a entrada data_memPrin_in da cacheL1;

Os outros componentes são decodificadores para o display de sete segmentos e os módulos de divisão e resto de divisão.

Formas de onda parte III

Para facilitar a escrita vou me referir a cada intervalo separado pelas retas vermelhas de intervalo.

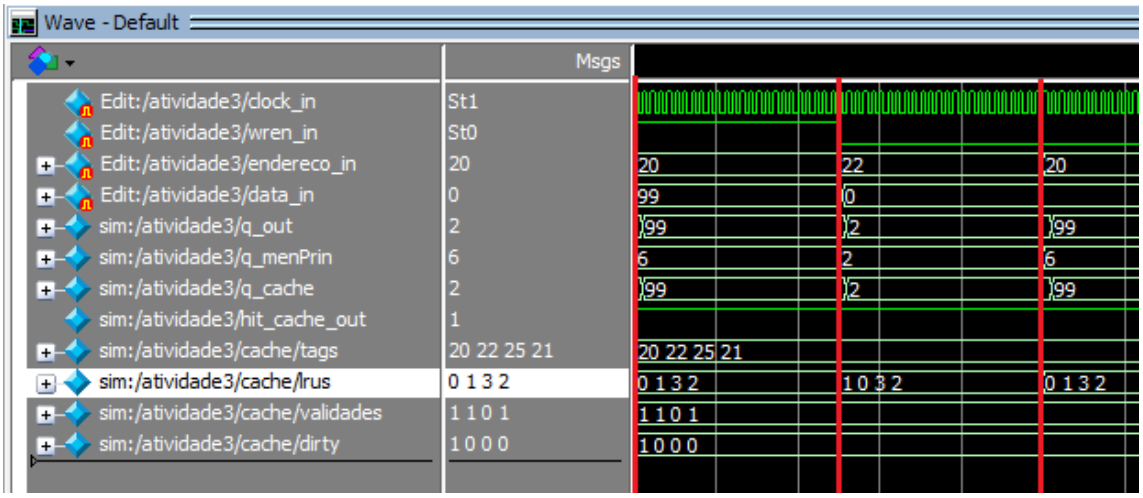

IMAGEM 5: Parte 3 – Simulação de escrita e leitura com hit.

No primeiro intervalo tem-se uma escrita (wren ativo) no endereço 20, que está na cache como pode ser observado no array de tags da cache, do valor 99. Pode-se observar que o sinal q_out é atualizado com o valor passado para a escrita, o sinal de hit é ativado e os lrus e o dirty são atualizados.

No segundo intervalo ocorre a leitura do valor no endereço 22, já que o sinal wren está desativado. Novamente o valor de q_out é atualizado, dessa vez com o valor presente na inicialização feita com o arquivo .mif. O lru também é atualizado. No ultimo momento é feita uma outra leitura, apenas para mostrar que o valor escrito no primeiro estagio foi realmente gravado na cache.

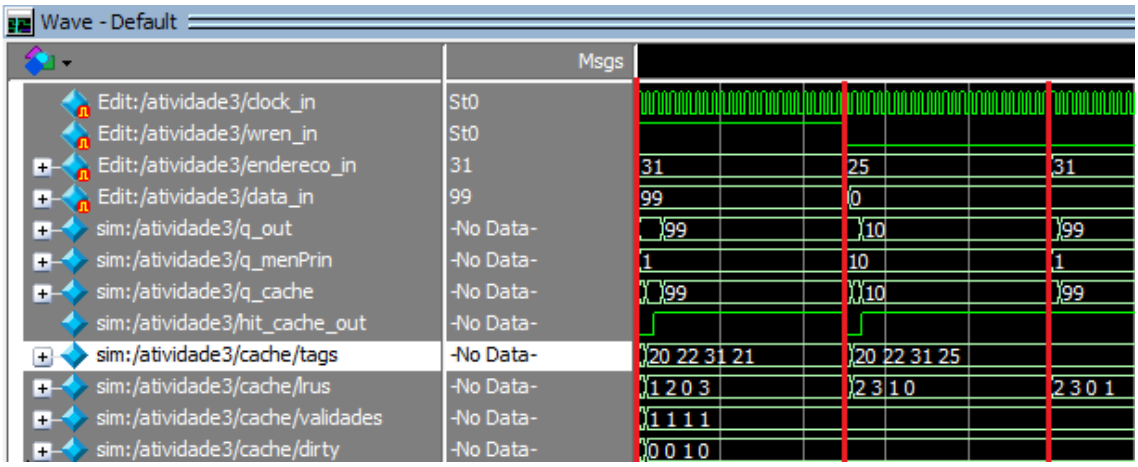

IMAGEM 6: Parte 3 – Simulação de escrita e leitura sem hit.

No primeiro intervalo tem-se a escrita do valor 99 na posição 31, esse processo demora alguns ciclos de clock a mais, pois primeiro a cache tem que detectar o miss, em seguida a memoria principal deve ser acionada para buscar o valor desejado e passa-lo para a cache na posição, com validade desativada ou, caso não exista nenhuma validade desativada, a que tiver maior LRU. Depois de encontrar a posição a ser substituída na cache, essa faz a escrita do novo valor no endereço buscado. Nesse processo o sinal de hit é desabilitado, a tag do novo endereço é salva, os lrus são atualizados e o bit de sujeira é ativado.

No segundo intervalo, acontece uma leitura do endereço 25, o qual está na cache, porem com o bit de validade desabilitado. Dessa forma, o mesmo processo de busca na memória principal que ocorre na escrita é realizado na leitura, porém o bit de sujeira não é ativado. No ultimo intervalo é realizado outra leitura a fim de mostrar que a escrita do primeiro intervalo realmente funcionou.

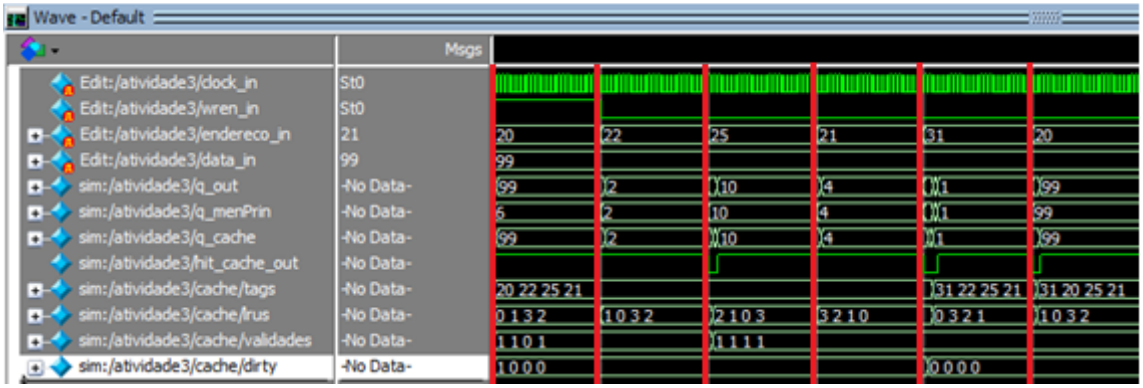

IMAGEM 7: Parte 3 – Simulação do funcionamento do write-back.

O funcionamento do write-back pode ser visto no quinto e sexto intervalo, os demais intervalos estão apenas “configurando” a cache para que o write-back ocorra. No primeiro intervalo é escrito o valor 99 na posição 20 com hit, assim o bit de sujeira é

ativado, nos demais intervalor até o quarto, são acessadas outras posições para que o próximo endereço a ser retirado da cache no caso de um miss seja o 20.

O quinto estagio é uma leitura com miss, o qual é mais detalhado na próxima imagem:

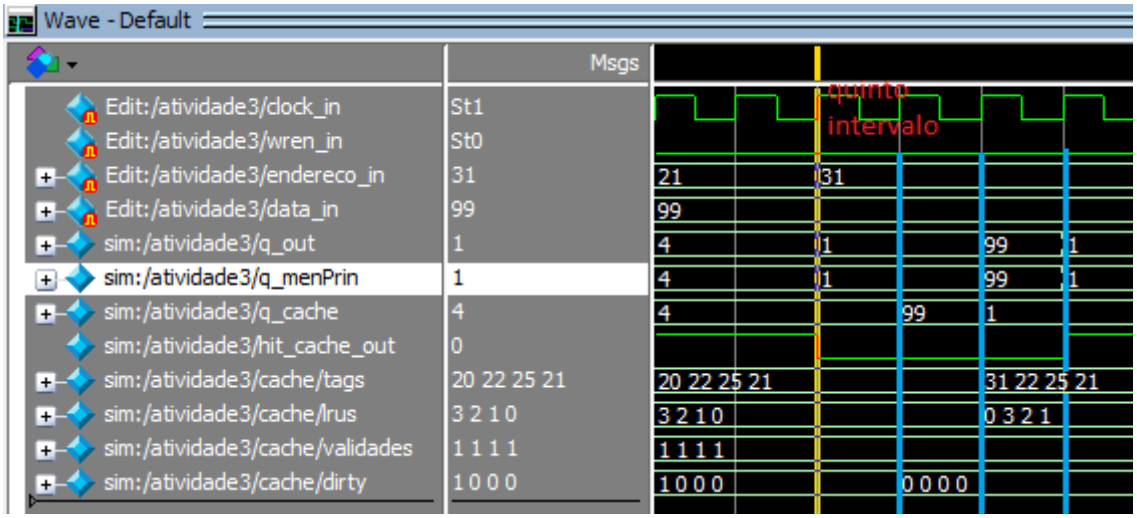

IMAGEM 8: Parte 3 – Simulação do funcionamento do write-back no quinto intervalo.

Cada separação em azul será referida como estagio.

No primeiro estágio a cache detecta o miss e avisa a memória principal, a qual busca o valor no endereço especificado e o encaminha para a saída q_out. No segundo a cache detecta que o bit de sujeira está ativado, então o desativa e busca o valor que deve ser atualizado na memoria principal. Agora no terceiro estagio a memoria principal realiza a escrita do valor lido pela cache, e a cache depois de passar o valor a ser atualizado para a memória principal, realiza a escrita do valor informado pela memoria principal na posição adequada de acordo com os bits de validade e lru. Ainda no terceiro estagio, os lrus e as tags são atualizados. No ultimo estagio muda apenas o valor a ser encaminhado para a saída q_out que agora é o q_cache, pois o valor do bit de hit agora é 1.

No sexto intervalo ocorre apenas uma leitura com miss no endereço 20, para mostrar que a politica de write-back realmente está funcionando.