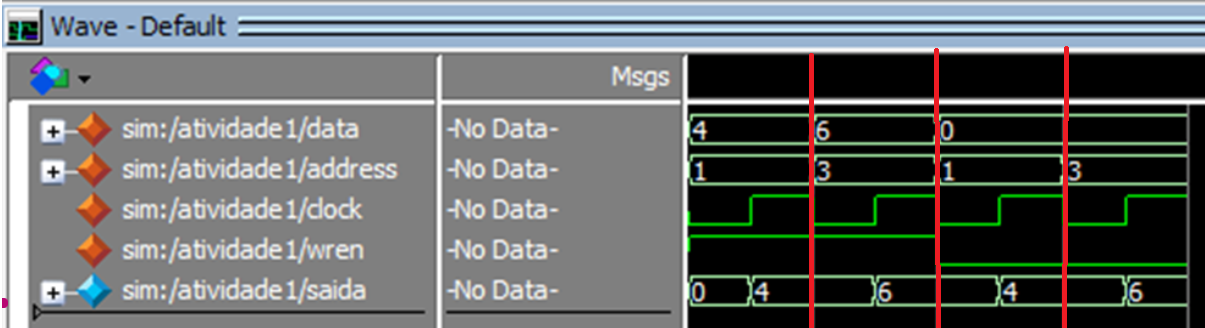
**Forma de onda parte I**



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

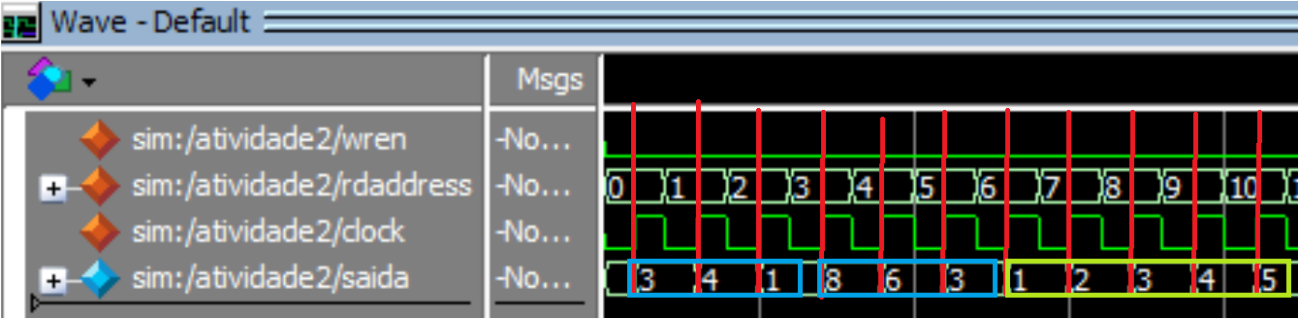
A imagem acima foi separada em 4 regiões sequenciais.

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 (2⁵) linhas e cache L1 de 4 (2²) linhas

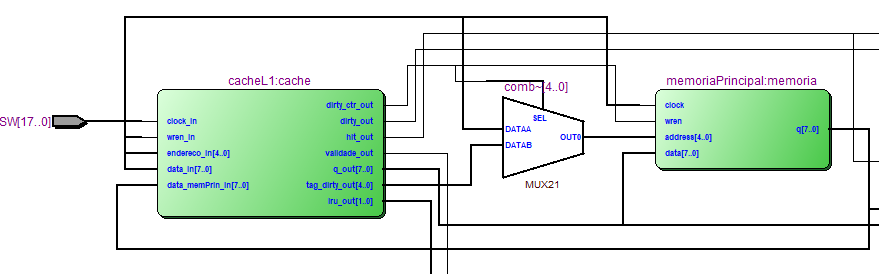
Mapeamento de Endereços em Memória Cache: Mapeamento Associativo

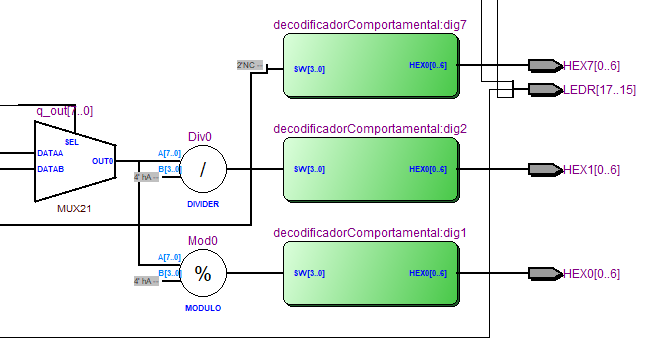
|  |
| --- |
| Endereço 5 bits |
| Tag (5 bits) |

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) |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |





Modulo cacheL1:

As entradas clock\_in, wren\_in, endereco\_in e data\_in são recebidos 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.

q\_out é encaminhado para o primeiro MUX e