Relatório Lab 1 – OC - 2024/2025

Grupo: 47

Alunos: David Carvalho (107221), Geovani Rocha (102050), Tomás Macieira (100596)

Tarefa 4.1 – Cache L1

Modo de endereçamento

Offset = 6 bits (Palavra * tamanho do bloco), index = 8 bits (Tamanho da cache L1 / número de blocos), restantes bits para a tag

Resumo do código

Utilizando o código fornecido pelos docentes, efetuámos as seguintes alterações:

- A estrutura de dados “cache“ utiliza agora uma lista de linhas do tipo “CacheLine”;
- Utilizamos mascaras de bits para isolar e retirar os bits necessários para o offset, index e a tag respetivamente. De seguida usamos o index para aceder à linha que corresponde ao endereço;
- A lógica para determinar se obtivemos um cache hit ou miss decorre de modo idêntico ao do código base. No entanto, para endereçarmos algo na memória da cache (array L1Cache) multiplicamos o tamanho do bloco pelo index, para descobrir a posição equivalente à linha, e fazemos um deslocamento com base no offset. Para aceder a DRAM, restauramos o seu endereço tendo em conta os bits da tag e do index.

Tarefa 4.2 – Cache L2

Modo de endereçamento

Offset = 6 bits, index = 9 bits (tamanho da cache L2 / número de blocos), restantes bits para a tag.

Resumo do código

Utilizando o código da cache L1, efetuámos as seguintes alterações:

- A execução da cache L1 decorre de modo idêntico, com exceção dos acessos à DRAM (cache miss), que foram substituídos por acessos à cache L2, respeitando assim a hierarquia;
- A lógica da cache L2 funciona de forma semelhante à de L1. No entanto as mascaras para isolar os bits do index e da tag, e o endereço de memória de L2 (array L2Cache), têm em conta as novas dimensões da cache;
- No caso de cache miss em L1, seguimos uma política de “write-back”, onde a cache L2, caso aceda à DRAM ou não, procede por escrever a palavra também na posição de memória da cache L1, que lhe foi passada como argumento.

Tarefa 4.3 – Cache L2 associativa duas vias

Modo de endereçamento

Offset = 6 bits, index = 8 bits (Número de linhas / 2), restantes bits para a tag.

Resumo do código

Utilizando o código da cache L2, efetuámos as seguintes alterações:

- Criação de uma estrutura de dados “Set” que contém duas linhas de cache do tipo CacheLine, e um bit “LRU” (utilização de política “least recently used”), que determina qual a linha que foi usada menos recentemente (bit a 0 informa que a a linha usada menos recentemente foi a primeira linha do set, e vice-versa). No início do programa LRU é 0;
- A estrutura de dados CacheL2, utiliza agora uma lista de Sets. Através deste set, retiramos as suas duas linhas e calculamos os seus respetivos endereços (index, lineTwoIndex). O endereço da segunda linha tem em consideração o bit que fora removido do index, devido à existência dos sets. Este bit será necessário para endereçar a memória da cache;
- Caso ambas as linhas forem inválidas, acedemos à DRAM e escrevemos a palavra na primeira linha, colocando o bit LRU a 1 (segunda linha);
- Caso uma das linhas tenha uma tag correspondente à do endereço, será essa linha a substituída, ficando a restante como LRU;
- Caso ambas tenham uma tag diferente da requerida, é substituída a linha LRU, ficando a restante definida como usada menos recentemente;