# Hierarquia de Memória Cache – Versão 23

Este documento descreve o terceiro trabalho da disciplina de Arquitetura de Computadores II, que consiste na implementação de um sistema de gerenciamento da hierarquia de memória. Este gerenciamento deve ser descrito em VHDL, e simulado junto com o processador utilizado no Trabalho 2 ou similar.

A seguir está apresentada uma hierarquia de memória com três níveis: (i) cache de nível 1 (L1), (ii) cache de nível 2 (L2) e (iii) Memória Principal (MP).



O processador se comunica com a hierarquia de memória através das portas de: (i) **endereço -** aponta para o endereço de um dado ou instrução; (ii) **controle** - habilita o acesso à memória e define se este acesso é de leitura ou escrita; (iii) **status** - indica se a informação já foi lida ou escrita na/da memória; e (iv) dados ou instruções - porta bidirecional no caso da memória ser de dados (efetua escrita e leitura), e porta unidirecional no caso da memória ser de instruções (só leitura).

Independente da implementação, pois o aluno tem algumas liberdades de escolha, os sinais provenientes do processador devem ir diretamente para o nível logo abaixo da hierarquia de memória, e cada nível será responsável por gerar os endereços para o nível subsequente.

Cada nível de cache tem um conjunto de sinais que reportam o resultado da operação de leitura, tais como miss ou hit. Para cada nível, também, devem ser direcionados sinais de controle, bem como endereços.

***O objetivo deste trabalho é implementar uma hierarquia de dados com 3 níveis (L1, L2 e MP)***, sendo a hierarquia de instruções implementada apenas com um nível (MP)

A memória de instruções deve ser implementada através de um arquivo contendo um código executável do processador. **Este código deve ser descrito de forma a verificar a funcionalidade da hierarquia da memória que está sendo implementada**. Assim, um programa que testa a hierarquia de memória deve fazer diversos acessos, forçando que ocorram alguns casos de cache miss e outros de cache hit de forma a explorar a localidade espacial e temporal do programa. O arquivo que contém o código executável do processador pode ser obtido com um programa *assembly* do MIPS sendo entrada para o montador do processador (e.g., MARS).

Na hierarquia de dados, considere que a cache de nível 1 tem 4 linhas, e cada bloco da linha tem 4 palavras. A cache de nível 2 tem 8 linhas e cada bloco da linha tem 4 palavras.

Entre cada nível da hierarquia de memória deve ser considerado um atraso parametrizável. A cache L1 deve operar na mesma frequência do processador (acesso com um ciclo de relógio). A cache L2 tem tempo de acesso de 2 ciclos de relógio, enquanto que a MP tem tempo de acesso de 4 ciclos de relógio. Para implementação das caches de dados devem ser consideradas um tipo de mapeamento e um mecanismo para manter a integridade de dados, tal como descrito a seguir: (i) **Cache L1 - mapeamento direto como *write-through***; (ii) **Cache L2 - mapeamento associativo como *write-back***, e ***regra de substituição com contador*** no caso de miss com cache tendo todas as posições ocupadas. Entre cada nível, sempre utilize bordas de relógio invertidas. Demais características de implementação, não definidas aqui, estão livres para serem escolhidas.

Ao final do trabalho o aluno terá uma arquitetura semelhante a que segue na figura abaixo. Note que o relógio não é apresentado, mas deve ter os valores descritos acima.



## Realização do trabalho

O trabalho deverá ser realizado em ***no máximo*** grupos de até 3 (três) alunos, e ser entregue até o dia descrito na agenda da disciplina. ***Deve*** ***ser entregue um relatório descrevendo as atividades feitas, juntamente com os arquivos fonte VHDL da implementação do trabalho e os arquivos com os experimentos utilizados***. É importante descrever a validação da gerência da hierarquia de memória. Note que validar este tipo de sistema implica diversos experimentos: tais como sucessivos miss e hits.

Espera-se que os grupos, apresentem gráficos com as latências de cinco programas (feitos pelos alunos, eventualmente copiados de livros ou Internet) para quatro arquiteturas alvo, tal como exemplificado no gráfico abaixo. *Um destes cinco programas têm que ser acesso a vetor/matriz de forma a privilegiar localidade temporal/espacial; enquanto o outro deve operar exatamente de forma inversa, ou seja, prejudicando a localidade*. Embora os programas possam ser adquiridos de terceiros, espera-se que os grupos validem os programas no MARS e obtenham os arquivos de código e de dados para usar no simulador VHDL.



As arquiteturas alvo consideradas são:

* MP-0: A arquitetura básica contendo um processador MIPS (e.g., MR2) com a memória implementada sem considerar atraso; ou seja, uma arquitetura usada em Organização de Computadores, Arquitetura de Computadores I, ou no T2 de Arquitetura de Computadores II. Para este caso é necessário apenas simular os cinco programas e coletar os tempos de execução;
* MP: A arquitetura MP-0 inserido atraso na memória principal. A sugestão é inserir um módulo entre o processador e a memória principal que implemente o atraso definido acima. Neste caso, o VHDL que implementa o módulo de memória propriamente dito será igual ao MP-0. Adicionalmente, toda a vez que o processador acessar a hierarquia de memória de dados, deve aguardar uma resposta de que o dado está disponível. Este procedimento pode ser feito com um sinal de ack, em um modelo tipo o semi-síncrono. Note que o processador terá que ser parcialmente modificado para não seguir a sua execução enquanto o ack não for ativado;
* MP+L1: A arquitetura MP com a interposição de uma cache de nível 1 entre o processador e a memória principal;
* MP+L1+L2: A arquitetura MP+L1 com a interposição de uma cache de nível 2 entre a cache de nível 1 e a memória principal.