Trabalho II de OAP - Hierarquia de Memória

Dhruv Babani, João Pedro da Cunha, Miguel Warken e Vitor Caus

1. Histórico

A hierarquia de memória é uma estrutura que organiza e gerencia diferentes tipos de memória em um sistema de computador. Ela foi desenvolvida para atender às necessidades de desempenho e capacidade de armazenamento dos computadores, oferecendo uma combinação de memória rápida e cara, e memória mais lenta e barata.

No início da história dos computadores, a memória principal era baseada em tubos de vácuo e depois em transistores individuais. Essas memórias eram caras e tinham capacidades muito limitadas. Com o advento dos circuitos integrados e das memórias de núcleo magnético na década de 1950 e 1960, foi possível aumentar a capacidade de armazenamento da memória principal. Na década de 1970, as memórias de acesso aleatório (RAM) de chips dinâmicos (DRAM) começaram a ser amplamente utilizadas como memória principal. Essas memórias eram mais baratas e tinham maiores capacidades, mas eram mais lentas em comparação com as memórias de núcleo magnético. A hierarquia de memória começou a se desenvolver nessa época, com a introdução de caches de memória entre o processador e a memória principal.

Com o avanço da tecnologia e a necessidade de processadores mais rápidos, a hierarquia de memória evoluiu ainda mais. A memória cache se tornou mais sofisticada, com algoritmos de substituição inteligentes e hierarquias de múltiplos níveis. Além disso, novas tecnologias de memória foram introduzidas, como a memória estática (SRAM) mais rápida, que passou a ser usada nos níveis de cache mais próximos do processador.

Nos últimos anos, surgiram novas tecnologias de memória, como a memória de acesso aleatório não volátil (NVRAM), que combina a velocidade da RAM com a capacidade de reter dados mesmo quando a energia e´ desligada. A NVRAM tem potencial para substituir as memórias secundárias tradicionais, como discos rígidos e unidades de estado sólido (SSDs), devido à sua velocidade e capacidade de armazenamento.

A hierarquia de memória continua a evoluir para atender às demandas dos aplicativos modernos, que exigem cada vez mais velocidade e capacidade de armazenamento. Novas tecnologias, como a memória persistente de acesso aleatório (PMEM), estão sendo pesquisadas e desenvolvidas para fornecer uma camada intermediária entre a memória principal e o armazenamento em massa, oferecendo maior capacidade e velocidade em comparação com os dispositivos de armazenamento tradicionais.

Em resumo, a hierarquia de memória evoluiu ao longo do tempo para fornecer uma combinação de memória rápida e cara, e memória mais lenta e barata, a fim de otimizar o desempenho e a capacidade de armazenamento dos sistemas de computador. As tecnologias de memória continuam a se desenvolver para acompanhar as demandas crescentes da computação moderna.

2. Motivação

A hierarquia da memória é motivada por duas principais considerações: capacidade e velocidade. Essas duas características são fundamentais para o desempenho eficiente de um sistema de computador.

A capacidade refere-se à quantidade total de informações que um sistema pode armazenar. As informações em um sistema de computador são armazenadas em diferentes tipos de memória, desde a memória principal até dispositivos de armazenamento em massa, como discos rígidos ou unidades de estado sólido (SSDs). Cada nível da hierarquia da memória tem uma capacidade diferente, com os níveis mais altos geralmente tendo menos capacidade do que os níveis mais baixos. Isso ocorre porque o custo por bit de armazenamento aumenta à medida que a velocidade de acesso diminui. Portanto, é mais econômico e prático ter níveis de memória com capacidades diferentes para atender às necessidades de armazenamento de um sistema.

A velocidade de acesso é o tempo necessário para ler ou escrever dados em uma determinada memória. A velocidade de acesso é inversamente proporcional à capacidade - quanto mais rápida é a memória, menor é a capacidade que pode ser fornecida por um determinado custo. Portanto, sistemas de computador utilizam diferentes níveis de memória para equilibrar a velocidade de acesso e a capacidade de armazenamento. Os níveis mais altos da hierarquia, como o cache de nível 1 (L1) e o cache de nível 2 (L2) integrados no processador, são os mais rápidos, mas também têm menor capacidade. A memória principal (RAM) é mais lenta, mas tem uma capacidade maior. Os níveis mais baixos, como os discos rígidos ou SSDs, têm velocidades de acesso ainda mais lentas, mas oferecem capacidades ainda maiores.

A hierarquia de memória busca equilibrar essas duas considerações conflitantes, combinando diferentes tipos de memória em uma hierarquia que oferece uma combinação ideal de velocidade e capacidade. Nos níveis mais baixos da hierarquia, estão os dispositivos de armazenamento mais rápidos, mas de menor capacidade, como a memória cache. Em níveis superiores, estão dispositivos com maior capacidade, mas tempos de acesso mais lentos, como a memória principal (RAM) e o armazenamento em disco.

Ao organizar a memória dessa forma, a hierarquia permite que o sistema acesse rapidamente os dados mais frequentemente utilizados, armazenando-os nos n´ıveis mais rápidos e próximos do processador. Dessa forma, é possível reduzir o tempo de acesso aos dados mais cr´ıticos para o desempenho do sistema. Ao mesmo tempo, a hierarquia de memória oferece uma capacidade de armazenamento suficiente para a maioria dos dados, mesmo que o acesso a eles seja um pouco mais lento.

A localidade espacial refere-se à tendência de um programa acessar dados próximos entre si. Por exemplo, em um loop de repetição, é comum que os elementos de um array sejam acessados sequencialmente. A localidade espacial explora esse padrão, armazenando blocos de dados adjacentes em níveis mais rápidos da hierarquia de memória, como a memória cache. Dessa forma, quando um dado é acessado, é provável que os dados vizinhos também sejam necessários em breve, reduzindo o tempo de acesso total.

A localidade temporal, por sua vez, refere-se à tendência de um programa acessar repetidamente os mesmos dados em um curto período de tempo. Por exemplo, em um

loop, uma variável pode ser acessada várias vezes. A localidade temporal explora esse padrão mantendo os dados recentemente acessados nos níveis mais rápidos da hierarquia de memória. Assim, se um dado é necessário novamente em um curto intervalo de tempo, ele pode ser prontamente recuperado, sem a necessidade de acessar níveis mais lentos da hierarquia de memória.

A exploração da localidade espacial e temporal é fundamental para o desempenho eficiente da hierarquia de memória. Ao armazenar e gerenciar os dados de acordo com esses princípios, é possível reduzir significativamente o tempo de acesso a memória, aproveitando os padrões de acesso típicos dos programas. Isso resulta em um melhor desempenho geral do sistema e uma utilização mais eficiente dos recursos de memória disponíveis.

Em resumo, a motivação por trás da hierarquia de memória é equilibrar a velocidade e a capacidade de armazenamento em um sistema de computador, garantindo um desempenho eficiente e econômico. Essa abordagem permite otimizar o tempo de acesso aos dados mais importantes, enquanto mantém a capacidade de armazenar grandes quantidades de informação.

3. Fundamentos

A hierarquia de memória é uma organização de diferentes níveis de armazenamento de dados em um sistema de computador. Cada nível da hierarquia possui características distintas em termos de capacidade, velocidade de acesso e custo por bit armazenado. Essa estrutura e´ projetada para otimizar o desempenho geral do sistema, fornecendo um equilíbrio entre custo e velocidade.

Os fundamentos da hierarquia de memória incluem:

1. Registradores: São os níveis mais rápidos e caros da hierarquia de memória. Os registradores estão localizados diretamente na Unidade de Processamento Central (CPU) e são usados para armazenar os dados e as instruções que o processador precisa acessar rapidamente. Eles têm a menor capacidade de armazenamento em comparação com os outros níveis da hierarquia.
2. Cache: O cache é um nível intermediário da hierarquia de memória, posicionado entre os registradores e a memória principal (RAM). É projetado para reduzir a latência de acesso à memória, armazenando dados frequentemente usados. Existem diferentes níveis de cache, como L1, L2 e L3, sendo o L1 o mais próximo da CPU e, portanto, mais rápido, mas também com menor capacidade de armazenamento em comparação com níveis superiores.
3. Memória Principal (RAM): A memória principal, geralmente chamada de RAM (Random Access Memory), é o nível seguinte na hierarquia. É mais lenta do que o cache, mas oferece uma capacidade de armazenamento significativamente maior. A RAM é usada para armazenar dados e instruções que são acessados com frequência pelo processador.
4. Memória Secundária: A memória secundária é composta por dispositivos de armazenamento de dados de longo prazo, como discos rígidos (HDDs) e unidades de estado sólido (SSDs). Esses dispositivos têm uma capacidade ainda maior do que a memória principal, mas são mais lentos em termos de velocidade de acesso. A memória secundaria é usada para armazenar dados e programas permanentemente, mesmo quando o computador é desligado.

A hierarquia de memória é projetada de forma hierárquica para otimizar o desempenho do sistema. Dados e instruções são armazenados nos níveis mais próximos da CPU, onde podem ser acessados rapidamente. Se o dado não estiver presente em um nível mais próximo, o sistema procura no próximo nível da hierarquia, aumentando a latência de acesso. O objetivo é minimizar a quantidade de tempo gasto em acessos mais lentos à memória secundária, aproveitando a velocidade e a capacidade dos níveis superiores da hierarquia.

4. Memorias Cache

A memória cache é um tipo de memória de alta velocidade que atua como uma área de armazenamento temporário entre a unidade de processamento central (CPU) e a memória principal em um computador. Ela é projetada para melhorar o desempenho do sistema, reduzindo o tempo necessário para acessar dados frequentemente utilizados pela CPU.

A principal função da memória cache é armazenar cópias dos dados mais frequentemente acessados pela CPU, a fim de reduzir a latência de acesso. Isso é possível porque a memória cache é construída com chips de memória muito mais rápidos do que a memória principal, o que permite que os dados sejam recuperados e entregues à CPU em velocidades muito mais altas.

A memória cache é organizada em vários níveis, geralmente chamados de cache L1, L2 e L3. O cache L1 é o mais próximo da CPU e também o menor, mas possui a menor latência de acesso. O cache L2 é um pouco maior e mais lento, enquanto o cache L3 é o maior e mais lento dos três. No entanto, o tamanho e a velocidade de cada nível de cache podem variar dependendo da arquitetura do processador.

Quando a CPU precisa acessar um determinado dado, ela verifica primeiro o cache L1. Se os dados estiverem presentes nesse nível, ocorre um ”acerto de cache” (cache hit),e os dados são imediatamente fornecidos à CPU. Isso acelera significativamente o tempo de acesso, pois a CPU não precisa esperar que os dados sejam buscados na memória principal.

No entanto, se os dados não estiverem presentes no cache L1, ocorre um ”erro de cache” (cache miss), e a CPU precisa procurar nos níveis de cache subsequentes ou, eventualmente, na memória principal. Se os dados estiverem presentes em um nível de cache superior, eles são copiados para o cache inferior para uso futuro, melhorando o desempenho subsequente.

A estratégia de gerenciamento de cache mais comum é conhecida como ”princípio de localidade”, que explora o fato de que os programas tendem a acessar dados próximos aos dados recentemente acessados. Isso significa que, se um determinado dado foi buscado recentemente, é provável que seja acessado novamente em um futuro próximo. Portanto, a memória cache armazena esses dados, esperando que sejam necessários novamente em breve.

Além disso, a memória cache também possui diferentes níveis de associatividade, que determinam como os dados são organizados dentro dela. A associatividade define quantos blocos de dados podem ser armazenados em uma determinada posição da memória cache. A associatividade direta armazena apenas um bloco de dados por posição, enquanto a associatividade totalmente associativa permite que qualquer bloco de dados seja armazenado em qualquer posição. A associatividade por conjunto, que é a mais comumente utilizada, permite que um conjunto de posições armazene vários blocos de dados, reduzindo a probabilidade de ocorrerem erros de cache.

Em resumo, a memória cache é uma parte essencial da arquitetura de um computador moderno. Ela melhora o desempenho do sistema armazenando cópias dos dados frequentemente utilizados pela CPU, reduzindo a latência de acesso e aproveitando o princípio de localidade. Com sua rápida velocidade de acesso e técnicas eficientes de gerenciamento de dados, a memória cache contribui para a execução eficiente de programas e o desempenho geral do computador.

5. Mapeamentos de Hierquia de Memória

O mapeamento de endereços em sistemas de memória cache é uma técnica utilizada para determinar onde os dados devem ser armazenados na cache. Existem três principais tipos de mapeamento de endereços: direto, totalmente associativo e associativo por conjuntos. Cada um deles possui suas próprias características e políticas de substituição de dados.

1. Mapeamento direto:No mapeamento direto, cada bloco de dados da memória principal é mapeado para uma localização específica na cache. A correspondência é feita por meio do uso de bits de endereço. Por exemplo, se uma cache tiver 2N blocos, os N bits menos significativos do endereço são usados para indexar a cache. A organização típica de um bloco de cache inclui: o bloco de dados em si (que armazena os dados do endereço principal), uma tag (que armazena o endereço principal completo) e bits de validade (que indicam se o bloco contém dados válidos ou está vazio).

A política de substituição de dados em um mapeamento direto é simples quando um novo bloco precisa ser colocado na cache, ele é armazenado na posição determinada pelo mapeamento direto, substituindo o bloco que ocupava originalmente essa posição. Isso significa que não há escolha na substituição de dados, uma vez que cada bloco tem uma localização fixa na cache.

2. Mapeamento totalmente associativo: No mapeamento totalmente associativo, cada bloco de dados pode ser armazenado em qualquer posição livre na cache. Isso significa que não há restrição quanto à localização do bloco na cache. A organização típica de um bloco de cache totalmente associativo é semelhante à do mapeamento direto, com um bloco de dados, uma tag e bits de validade.

Quando um novo bloco precisa ser armazenado na cache em um mapeamento totalmente associativo, é necessário escolher uma posição livre na cache. Se todas as posições estiverem ocupadas, é necessário utilizar uma política de substituição de dados para decidir qual bloco será substituído. As políticas comuns de substituição de dados incluem a política de substituição menos recentemente utilizado (LRU), a política de substituição mais recentemente utilizado (MRU), entre outras.

3. Mapeamento associativo por conjuntos: No mapeamento associativo por conjuntos, a cache é dividida em conjuntos, e cada bloco da memória principal é mapeado para um conjunto especifíco. Dentro de cada conjunto, o mapeamento é direto. A organização típica de um bloco de cache em um mapeamento associativo por conjuntos inclui um conjunto de blocos de dados, tags e bits de validade.

A escolha do conjunto em que um bloco será armazenado é determinada pelos bits do meio do endereço. Uma vez que o conjunto é determinado, o mapeamento direto é utilizado para decidir a posição exata dentro do conjunto onde o bloco será armazenado. A política de substituição de dados em um mapeamento associativo por conjuntos é semelhante à do mapeamento totalmente associativo, onde é necessário escolher uma posição livre dentro do conjunto e, se todas estiverem ocupadas, utilizar uma política de substituição para decidir qual bloco será substituído.

Em resumo, o mapeamento de endereços direto, totalmente associativo e associativo por conjuntos são técnicas utilizadas para determinar onde os dados devem ser armazenados na cache. Cada tipo possui sua própria organização de blocos de cache, tags e bits de validade, além de políticas de substituição de dados específicas para lidar com a ocupação da cache.

6. Integridade de dados e problemas de coerência em memoria Cache

A integridade de dados em uma cache refere-se à garantia de que os dados armazenados na cache estão atualizados e correspondem aos dados presentes na memória principal. Uma cache é um tipo de memória intermediária que armazena cópias dos dados frequentemente acessados da memória principal, permitindo um acesso mais rápido a esses dados pelo processador.

No entanto, a presença de caches em sistemas multiprocessadores pode levar ao problema de coerência de cache. Esse problema ocorre quando vários processadores têm cópias de uma mesma região de memória em suas caches e realizam operações de leitura e escrita nessas cópias de forma assíncrona.

Quando um processador realiza uma operação de escrita em um dado que está presente em sua cache, ele atualiza apenas a cópia local na cache, sem atualizar imediatamente a cópia na memória principal. Isso cria a possibilidade de que outros processadores, que possuem cópias desatualizadas desse dado em suas caches, continuem a utilizar essas cópias desatualizadas, o que pode levar a resultados inconsistentes e violações da integridade dos dados.

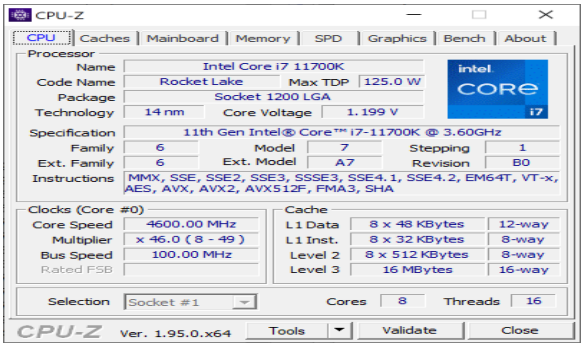
Para resolver o problema de coerência de cache, são necessários mecanismos que garantam a consistência dos dados compartilhados entre os processadores. Esses mecanismos podem ser baseados em protocolos de coerência de cache, que definem regras para as operações de leitura e escrita realizadas pelos processadores.

Existem diferentes protocolos de coerência de cache, como o protocolo de coerência de cache MSI (Modified, Shared, Invalid), o protocolo MESI (Modified, Exclusive, Shared, Invalid) e o protocolo MOESI (Modified, Owned, Exclusive, Shared, Invalid). Esses protocolos estabelecem regras para as transições de estados de uma linha de cache (por exemplo, de compartilhada para modificada) e para a troca de informações de controle entre os processadores.

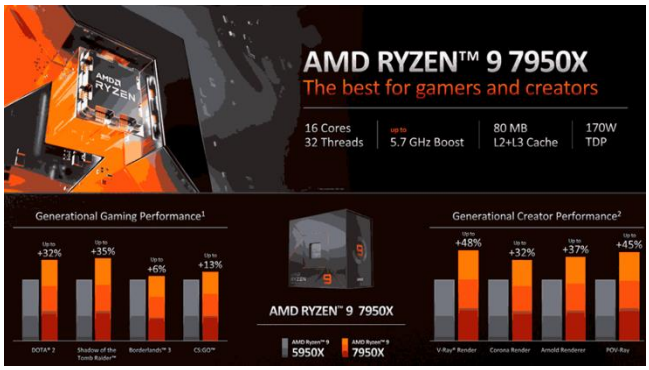
7. Exemplos de processadores com sua hieraquia de memoria

É importante observar que essas configurações de memória podem variar dependendo do modelo específico do processador e da geração em que foi lançado. Além disso, a hierarquia de memória também pode incluir outras camadas, como memória de registro, memória de alto desempenho em placas gráficas (GPUs) e memória em sistemas distribu´ıdos.Mas aqui embaixo esta´ alguns exemplos de processadores:

1. Processador Intel Core i7-11700K:
   Cache L1: 512 KB (256 KB por núcleo)
   Cache L2: 4 MB (1 MB por núcleo)
   Cache L3: 16 MB (compartilhado entre todos os núcleos)
   Memória principal (RAM): DDR4, geralmente variando de 8 GB a 64 GB ou mais.

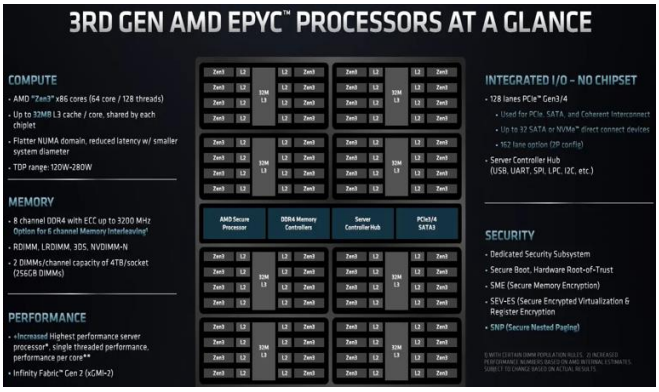


2. Processador AMD Ryzen 9 5900X:
   Cache L1: 512 KB (64 KB por núcleo)
   Cache L2: 8 MB (1 MB por núcleo)
   Cache L3: 64 MB (compartilhado entre todos os núcleos)
   Memória principal (RAM): DDR4, geralmente variando de 8 GB a 128 GB ou mais.



3. Processador Apple M1:
   Cache L1: 192 KB (64 KB de dados, 128 KB de instruções, por núcleo)
   Cache L2: 12 MB (compartilhado entre todos os núcleos)
   Cache L3: Não especificado publicamente.
   Memória principal (RAM): Integrada no chip, geralmente variando de 8 GB a 16 GB em dispositivos Mac.

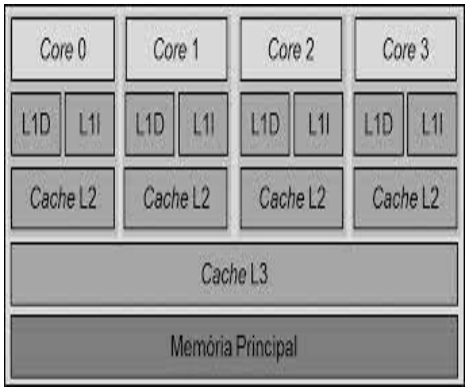
4. Processador Qualcomm Snapdragon 888:
   Cache L1: 256 KB (128 KB de dados, 128 KB de instruções, por núcleo)
   Cache L2: 3 MB (compartilhado entre todos os núcleos)
   Cache L3: Não especificado publicamente.
   Memória principal (RAM): LPDDR5, geralmente variando de 6 GB a 16 GB em dispositivos móveis.



8 Imagens e Tabelas referentes do Trabalho

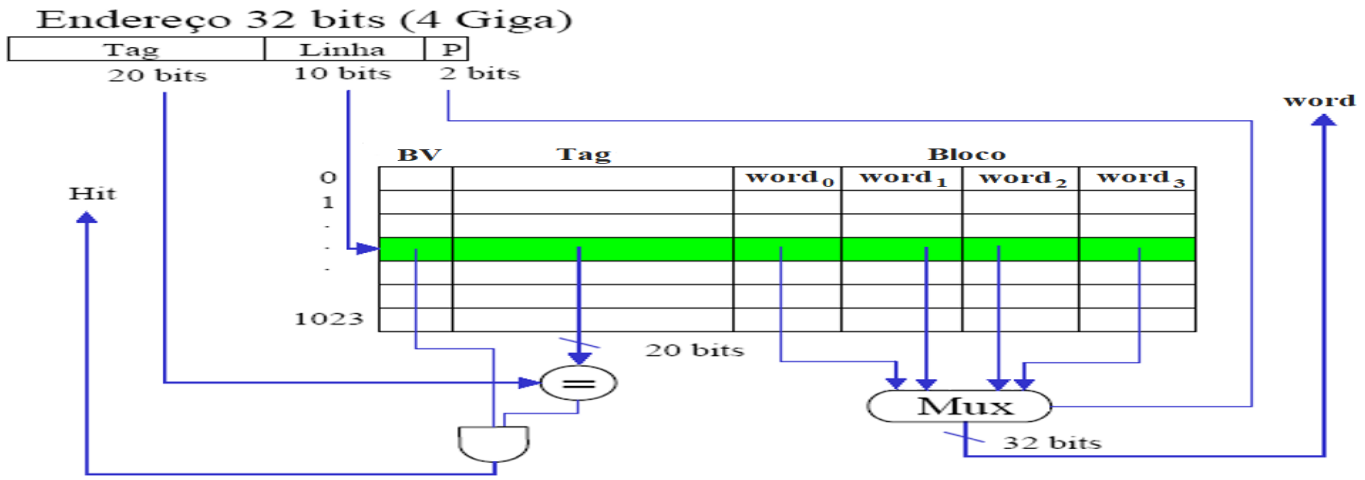
1)Hierarquia de Memoria:

2) Memorias de Cache

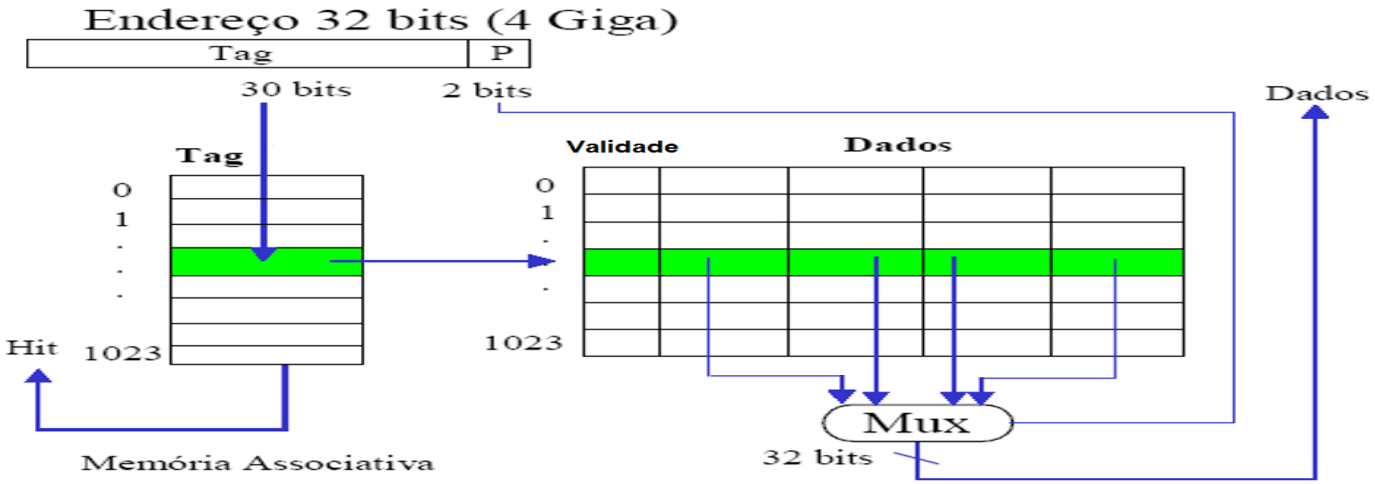


3) Mapeamentos de endereço em cache

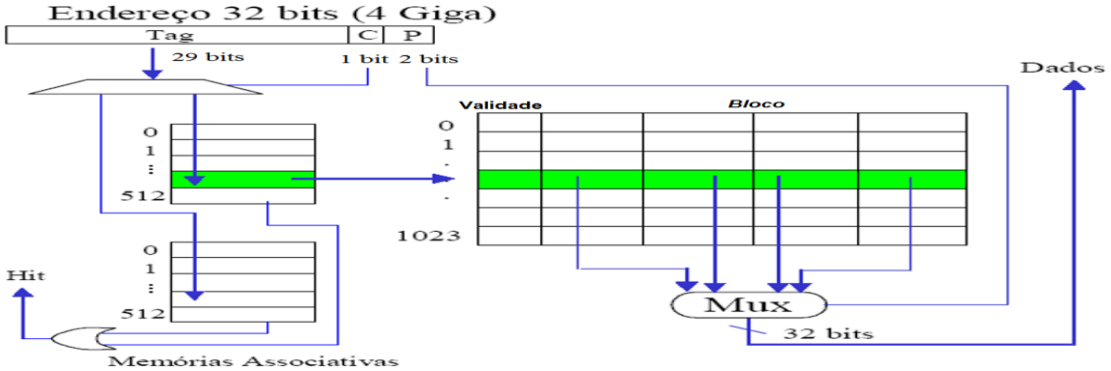
-Mapeamento Direto:



-Mapeamento associativo:



-Mapeamento Conjunto Associativo:



9 Considerações Finais

Durante a exploração do tema "hierarquia de memória", foi possível obter aprendizados significativos sobre a organização e o funcionamento dos sistemas de memória em computadores. A hierarquia de memória é um conceito fundamental na arquitetura de computadores, que busca otimizar o desempenho do sistema ao utilizar diferentes níveis de memória com características distintas.

A aplicação prática da hierarquia de memória é ampla e afeta diretamente o desempenho de diferentes tipos de sistemas computacionais, desde dispositivos móveis até supercomputadores. A hierarquia de memória visa reduzir o gargalo de desempenho entre a velocidade de processamento da CPU e a velocidade de acesso à memória, aproveitando as características de diferentes níveis de memória para armazenar e acessar dados de forma eficiente.

Ao compreender a hierarquia de memória, é possível tomar decisões de projeto e implementação que resultam em sistemas mais rápidos e eficientes. Algumas das conclusões práticas sobre o aprendizado obtido na exploração desse tema incluem:

1. **Aproximação de acesso rápido: Quanto mais próximo a memória estiver do processador, menor será o tempo de acesso aos dados. Portanto, é essencial ter memórias mais rápidas e de menor latência nos níveis mais próximos à CPU.**

2. **Capacidade de armazenamento: À medida que se avança para níveis mais distantes do processador, a capacidade de armazenamento aumenta, mas o tempo de acesso aos dados também aumenta. Portanto, é necessário encontrar um equilíbrio entre a capacidade de armazenamento e o tempo de acesso para cada nível de memória.**

3. **Princípio da localidade: Os programas de computador têm a tendência de acessar um conjunto limitado de dados de forma repetida. Esse princípio é conhecido como localidade temporal e localidade espacial. Ao aproveitar esse padrão, os sistemas de memória podem ser projetados para armazenar os dados mais frequentemente acessados próximos ao processador, melhorando o desempenho geral.**

4. **Cache: O uso de memórias cache é uma técnica amplamente empregada na hierarquia de memória. Os caches são memórias de acesso rápido que armazenam cópias dos dados mais frequentemente utilizados. A utilização eficiente de caches pode reduzir significativamente o tempo de acesso à memória principal, melhorando o desempenho geral do sistema.**

5. **Evolução tecnológica: Com o avanço da tecnologia, novas arquiteturas de memória estão sendo desenvolvidas, como as memórias não voláteis (como as memórias flash) e memórias de acesso aleatório persistentes (NVRAM). Essas tecnologias oferecem desafios e oportunidades para o design da hierarquia de memória, pois podem alterar as características dos diferentes níveis de memória.**