Infraestrutura de Hardware - 3. EE - Memória e Multiprocessadores 2017-1  
  
***1. Considere um sistema computacional que possui um processador superescalar que trabalha na frequência de 2,0 GHz. O espaço de endereçamento virtual é de 8192 Gbytes. A quantidade de discos corresponde ao espaço de endereçamento que é organizado em páginas de 16 Kbytes, porém o sistema possui uma memória principal com 1/1024 da capacidade da memória virtual e duas memórias cache, uma para instruções e outra para dados, uma com 1/512 e outra com 1/1024 da capacidade de armazenamento da memória principal. O sistema ainda inclui uma TLB associativa de 2048 slots.***

1. ***[0,5] Quais as vantagens e desvantagens de se ter um sistema hierárquico de memória?***

**Vantagens:**

Tira proveito do princípio da localidade espacial e temporal: dados usados recentemente e dados próximos aos usados recentemente provavelmente serão usados de novo.

Otimiza as diferentes tecnologias que relacionam custo e tempo de acesso: memórias mais rápidas e menores são mais caras, e memórias grandes porém lentas são baratas. O que se deseja atingir com a hierarquia é o custo e tamanho de uma memória grande como o HD, mas ter o tempo de acesso de uma SRAM (memória do tipo mais caro)

Dado a essa otimização, o desempenho da CPU aumenta: as memórias não se desenvolveram no mesmo passo dos processadores, o que faz com que, sem um planejamento do sistema de memória, a capacidade do processador tenha um gargalo, sendo impedido de operar com máxima eficiência.

**Desvantagens:**

Maior complexidade na implementação e custo elevado.

***2. Para o sistema da questão 1, a cache de instruções é associativa por conjunto com grau de associatividade igual a 8 e a política de escrita é write-through. A cache de dados possui metade da associatividade da cache de instruções e a política de escrita é write-back. Para as duas caches a transferência de informação é feita considerando-se blocos de 8 palavras.***

*g* Descrição do sistema especificado na questão 1:

* Processador de 2.0GHz
* 8192 GB de Espaço de Endereçamento Virtual
* Página de 16Kb
* Memória Principal de 8GB ~> 8192 MB
* Cache de Instruções de 16MB
* Cache de Dados de 8MB
* TLB Fully Associative com 2048 Slots.

1. ***[1,0] Qual o layout e tamanho em (bytes) das caches de instrução e dados?***

**Cache de Instruções:**

* 8-way Set Associative
* Write-through
* Block size 8

Isso nos deixa com uma cache de 25 bytes cada bloco. Como a cache é de 16MB, ela possui 224 bytes. Dividindo 224/25, temos que nossa cache terá 219 slots.

Como é 8-way associative, temos 219/23 = 216. Para endereçar esses sets, precisamos de **16 bits (set)**

Para achar a word dentro do bloco de tamanho 8, precisamos de **3 bits (block offset)**

E para words de 4 bytes, precisamos de **2 bits (byte offset).**

Os bits restantes são usados para formar a **Tag (11 bits).**

Layout de Endereço da Cache de Instruções:

|  |  |  |  |
| --- | --- | --- | --- |
| Tag (11) | Set (16) | Block Offset (3) | Byte Offset (2) |

Layout da Cache de Instruções:

Cada **set** será da seguinte forma:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1 bit | 11 bits | 256 bits x 8 (cada conjunto tem 8 espaços para blocos) | | | | | | | |
| Valid | Tag | Data | Data | Data | Data | Data | Data | Data | Data |
| x 216 Slots \* 8 sets (n sei oq isso quer dizer) / eu faria: [ 2¹⁶ \* 2³ \* (1 + 11 + 256) ] / 2³ | | | | | | | | | |

ou apenas 219 (número de slots) \* 268 (bits por slot)/8 (transformar pra byte)

= 17.563.648 bytes desculpa por ser prolixo

na verdade eu sou só um lixo

Te amo Lucky (não Luke)

****

**Cache de Dados:**

* 4-way Set Associative
* Write-back
* Block size 8

Isso nos deixa com uma cache de 25 bytes cada bloco. Como a cache é de 8MB, ela possui 223 bytes. Dividindo 223/25, temos que nossa cache terá 218 **blocos.**.

Como é 4-way associative, temos 218/22 = 216. Para endereçar esses sets, precisamos de **16 bits (set)**

Para achar a word dentro do bloco de tamanho 8, precisamos de **3 bits (block offset)**

E para words de 4 bytes, precisamos de **2 bits (byte offset).**

Os bits restantes são usados para formar a **Tag (11 bits).**

Layout de Endereço da Cache de Dados:

|  |  |  |  |
| --- | --- | --- | --- |
| Tag (11) | Set (16) | Block Offset (3) | Byte Offset (2) |

Layout da Cache de Dados:

Cada **slot** será da seguinte forma:

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1 bit | 1 bit | 11 bits | 256 bits x 8 espaços | | | | | | | |
| Dirty | Valid | Tag | Data | Data | Data | Data | Data | Data | Data | Data |
| x 216 Slots \* 4 sets (nao sei pq isso dnv kk) | | | | | | | | | | |

1. ***[1,0] Para uma determinada aplicação o CPI (ideal) é de 1,5. No entanto a taxa de acertos da cache de instruções e da cache de dados é de 95% e 92%, respectivamente, e o tempo para transferir um bloco é de 100ns. Qual o impacto das caches no CPI se as instruções de acesso à memória para a aplicação correspondem a 32% das instruções?***

Como 32% das instruções são de acesso a memória, apenas 32% estão sujeitas ao miss rate de 8% da cache de dados. Como a cache de instruções é sempre acessada primeiro, ela está sujeita a 100% do miss rate de 5% da cache de instruções.

Sabemos que:

Então

-- pra mim isso de cima ta errado --

15,12/16,62 = 90% da utilização, passa 90% do tempo utilizado em stall na memória

1. ***[0,5] Descreva duas técnicas de otimização que reduzem a taxa de faltas. Informe qual o tipo de falta que a técnica impacta diretamente (compulsória, capacidade ou conflito)***

Aumento de associatividade, pois agora blocos de memória são mapeadas para posições diferentes, possibilitando uma maior comutação(Miss rate). Aumento tamanho bloco(Miss rate).

1. ***[0,5] Descreva duas técnicas de otimização de memória cache que pode reduzir a penalidade, informe qual o tipo de penalidade reduzida pela técnica.***

Early restart, palavra requisitada do bloco quando é carregada na cache **(a palavra toda por ordem de endereçamento)** é enviada para a CPU, diminuir Miss Penalty.

Critical Word First, requisita palavra **específica** primeiro da memória e envia para cpu depois de carregada (Miss Penalty).

Memórias mais largas para acesso paralelo a mais de uma palavra por bloco.

Aumentar largura de banda entre cache e memória.

***3. Numa segunda versão do projeto decidiu-se colocar um segundo nível de cache (cache L2) com o dobro do tamanho da cache de instruções, organizada como mapeamento direto e 10 vezes mais rápida que a memória principal. Com esta nova configuração, a taxa de faltas da cache de instruções (com relação à memória principal) passou a ser de 3% e a de dados passou a ser de 6%. Quais as implicações desta mudança no desempenho do sistema? [1,0]***

Em Instruções, da L1 para L2, ainda temos 5% de misses, afinal, a L1 não mudou de tamanho, porém, agora a penalidade é 100. Da L2 para Principal, temos 3%, com penalidade de 100ns, pois a P não mudou de tecnologia. O mesmo vale para Dados, porém com taxas de 8%\*32% e 6%\*32% para L1 e L2, respectivamente.

Então

Melhora de 1,29x

***4. Considerando o sistema da questão 1 responda as questões a seguir.***

1. ***[1,0] Qual o layout e tamanho da tabela usada na tradução do endereço?***

Com uma página de 16KB, 24.210= 214, precisamos de 14 bits para endereçar o byte dentro da página. Ou seja, o page offset será de 14 bits.

Com 8192GB de memória virtual, precisamos de 213.230 = 243 = 43 bits para endereçar a memória virtual. Desses 43 bits, 14 serão de Page Offset e 29 bits para Virtual Page Number.

Com 8GB de memória fisica, precisamos de 33 bits para endereçá-la completamente. Desses 33 bits, 14 serão de Page offset e 19 de Physical Page Number.

Endereço de Memória Virtual:

|  |  |  |  |
| --- | --- | --- | --- |
| Virtual Page Number(29) | | Page Offset(14) | |

Endereço de Memória Física

|  |  |  |  |
| --- | --- | --- | --- |
| Physical Page Number(19) | | Page Offset(14) | |

Layout da Tabela de Tradução:

|  |  |  |  |
| --- | --- | --- | --- |
| Valid(1) | Dirty(1) | Physical Page Number(19) | |

1. ***[1,0] Qual o tamanho e layout da TLB e quais vantagens e desvantagens de se ter este mecanismo?***

Como nossa TLB tem 2048 (211 ) Slots e é fully associative, não necessitamos de um campo Index no nosso layout de endereço, ficando com o seguinte layout:

Layout da TLB:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Valid(1) | Dirty(1) | Tag(29) | Physical Page Number(19) | |

***5. Descreva o mecanismo de RAID, bem como vantagens e desvantagens deste mecanismo num sistema hierárquico de memória. [1,0]***

Esse mecanismo consiste em conjuntos de discos baratos e redundantes onde os dados persistentes são armazenados e distribuídos. Temos vários níveis:

**RAID 1,** com N discos normais + N discos de redundância, onde a redundância é por dado, isto é, cada dado tem uma cópia, temos um sistema caro e ineficiente.

**RAID 10 (1+0),** funciona como a RAID 1, porém os discos podem ser acessados paralelamente.

**RAID 3,** com N discos + 1 (de redundância), que armazena bits de paridade, permitindo recuperar informações de um bloco que der problema. Porém, se dois ou mais blocos apresentarem problema, o disco de paridade não conseguirá resolver.

**RAID 4,** parecida com a RAID 3, porém, a leitura é apenas no disco contendo o bloco, e não em todos os discos.

**RAID 5,** também possui N+1 discos, mas a paridade é distribuída entre os discos, de forma que seja possível verificar a paridade e realizar escritas simultâneas em vários discos.

***6. Processos concorrentes que se comunicam através de variáveis compartilhadas é um mecanismo suportado por várias linguagens paralelas permitindo explorar paralelismo de thread.***

1. ***[0,5] Por que processadores multi-core são mais adequados para se explorar paralelismo de thread em comparação com processadores superescalares?***

Processadores multi core suportam várias threads por núcleo, onde cada núcleo também pode ser um pipeline superescalar. Então, aplicações que exploram de paralelismo de thread terão muito mais suporte e eficiência, uma vez que podem ser distribuídas entre os vários cores do processador. Se a abordagem superscalar fosse mantida, haveria um gargalo nessa eficiência, uma vez que o número de threads suportada pelo processador seria limitada, devido ao próprio paralelismo do superscalar ser limitado ao paralelismo a nível de instrução.

1. ***[1,0] O que é o problema de coerência de memória e porque este problema existe?***

Em um sistema de multi processadores, cada processador possui sua própria cache privada, isso significa que que mais de um processador pode ter o mesmo dado da memória principal em sua cache. Se um dos processadores alterar esse dado compartilhado, a informação salva nas outras caches se tornam inconsistentes, e esse é o problema de coerência de memória. Basicamente, coerência ocorre quando uma leitura último valor escrito, e não um valor desatualizado que ainda não foi escrito na cache que está sendo acessada.

1. ***[1,0] Descreva a melhor técnica para resolver o problema da coerência se a memória for centralizada. Quais as vantagens e desvantagens desta técnica?***

Como a memória é centralizada podemos utilizar a técnica snooping, que consiste em ligar todas as caches por um barramento e cada controlador de cache “presta atenção” nos dados que passam pelo barramento. No Write Update, a escrita acontece junto com a atualização das cópias. No Write Invalidate, quando um dos controladores vê um dado que foi requisitado que ele já possui na própria cache, ele pode invalidar, atualizar ou fornecer o valor. Ou seja, cada bloco da cache mantém o estado de compartilhamento do dado.

A desvantagem dessa técnica é que ela se torna inviável se o barramento for muito complexo ou se a memória não for centralizada.

***7. Complete a tabela abaixo explicando se é possível acontecer o conjunto descrito e como ele pode ser reproduzido:***

|  |  |  |  |
| --- | --- | --- | --- |
| **Cache** | **TLB** | **Memória** | **Explicação** |
| miss | hit | hit | Possível, não está presente na cache, porém já poderia ter sido referenciada anteriormente e ter sido substituída na cache, onde apenas a TLB teria espaço para continuar mantendo o endereço da memória. |
| hit | miss | hit | Possível, TLB substituída |
| miss | miss | hit | Possível, TLB substituída e falta de cache |
| miss | miss | miss | Possível, o dado não está presente na memória. |
| miss | hit | miss | Impossível |
| hit | hit | miss | Impossível |
| hit | miss | miss | Impossível |

***EE3 20XX.? (É a do Black Market, mas transcrita, foto tava podre.)***

**01) Considere um sistema computacional que executa aplicações em um processador RISC, mas quais 35% das instruções são de acesso à memória, 50% das instruções são aritméticas e 15% são de desvio. O espaço de endereçamento virtual é de 1024 GBytes. O número de discos corresponde ao espaço de endereçamento que é organizado como páginas de 16 KBytes, porém o sistema possui uma memória principal com 1/256 da capacidade da memória virtual e duas memórias cache, uma para instruções e outra para dados, uma com 1/64 e outra com 1/128 de capacidade de armazenamento de memória. O sistema ainda inclui uma TLB associativa de 512 entradas.**

1. **Descreva DUAS vantagens e UMA desvantagem de se ter duas caches em vez de uma cache apenas para dados e instruções.**

Vantagens : Maior largura de banda e evita hazard estrutural  
 Desvantagem : Aumenta o número de faltas

1. **Descreva como acontece um acesso a memória cache desde que um endereço é colocado no barramento pelo processador.**

**02) Para o sistema da questão anterior, a cache de instr. é associativa por conjunto com grau de associatividade igual a 8 e a política de escrita é write-back. A cache de dados possui metade da associatividade da cache de instr. e a política de escrita é write-through. PAra as duas caches a transferência de informação é feita considerando-se blocos de 8 palavras.**

1. **Para cada cache do sistema, qual layout de endereço, e quantos bytes possuem cada uma das caches?**

End. virtual é 1024GB = 2¹⁰ \* 2³⁰, tamanho da memória será 2² \* 2³⁰ (4GB).

Para a ICache: Teremos 8 palavras de 4 bytes, totalizando 32 bytes, logo serão 3 e 2 bits para referenciar o block offset e byte offset, respectivamente.

Para os conjuntos são necessários 3 bits.

Vale lembrar que o tamanho desta cache é de 64MB, **implicando em 26 bits para endereçamento**, mas desses bits, **3 são desnecessários por conta da associatividade** e ( 3 + 2 = ) 5 são para os offsets, nos levando a 18 bits para representar cada linha. Como são necessários 32 bits para endereçar a memória principal, precisaremos de 18 bits para a tag.

|  |  |  |  |
| --- | --- | --- | --- |
| Tag(9) | Set(18) | Block Offset(3) | Byte Offset(2) |
| 32 bits | | | |

Layout da cache de instruções(write back):

|  |  |  |  |
| --- | --- | --- | --- |
| Validate(1) | Dirty(1) | Tag(9) | Dados(32\*8) |

seu tamanho será calculado multiplicando a quantidade de bits de uma “linha” pelo número de linhas:

Para o DCache: também serão 8 palavras de 4 bytes.

Porém, o tamanho desta cache será a metade da cache acima, então serão necessários 25 bits para endereçar cada byte. O grau de associatividade será 4.

|  |  |  |  |
| --- | --- | --- | --- |
| Tag(9) | Set(18) | Block Offset(3) | Byte Offset(2) |
| 32 bits | | | |

Layout da cache de instruções(write through):

|  |  |  |
| --- | --- | --- |
| Validate(1) | Tag(9) | Dados(32\*8) |

seu tamanho será calculado multiplicando a quantidade de bits de uma “linha” pelo número de linhas:

1. **Para este sistema sabe-se que a taxa de hit da cache de instruções e da cache de dados é de 95% e 93%, respectivamente. Se o hit time para a cache de instruções e de dados é de 1 ciclo e a penalidade é de 32 ciclos, qual o tempo médio de acesso de cada cache?**

**03) Para o sistema da questão anterior decidiu-se duplicar o grau de associatividade da cache de dados e tamanho do bloco. Quais as implicações destas mudanças no custo do sistema de memória (tamanho da cache) e no tempo médio de acesso de cada cache? Justifique quanti/qualitativamente.**

Quantitativamente:

Como a quantidade de Slots não muda (2²⁰) e a quantidade de espaços num conjunto agora será 2³, a quantidade de conjuntos será 2²⁰/2³ = 2¹⁷ -> serão necessário 17 bits para endereçar os Sets.

Como o tamanho do bloco agora será de 16 palavras, serão necessário 4 bits para endereçar a palavra em questão.

O byte offset não muda, já que cada palavra por padrão tem 4 bytes.

|  |  |  |  |
| --- | --- | --- | --- |
| Tag(9) | Set(17) | Block Offset(4) | Byte Offset(2) |
| 32 bits | | | |

Qualitativamente:

Como visto na questão anterior, o número de bits para o Set do endereço é igual ao número de slots/associatividade, isso significa que, quanto maior a associatividade, menos bits serão necessários para o Set e mais bits poderão ser usados na Tag. Isso implica no aumento da cache com o aumento da associatividade, já que é necessário salvar a Tag junto dos dados. No entanto, com o aumento da associatividade, reduzimos a quantidade de misses por conflitos de endereçamento, ao preço de um aumento no custo, já que todas as tags devem ser comparadas simultaneamente em todos os sets, aumentando a complexidade e o Hardware necessário.

Mas o aumento do bloco causa o efeito contrário do aumento da associatividade na tag, isso é, precisamos aumentar o bit de block offset em 1, logo temos que diminuir a tag em 1 também.

No final das contas, dobrar a associatividade e o número de blocos não causa mudanças no tamanho da cache. Ao custo de mais comparadores e um hardware mais complexo, pode haver uma redução na taxa de miss. Com o aumento de blocos, haverá maior aproveitamento da localidade espacial, porém ao preço de um aumento na penalidade caso um miss ocorra, já que mais dados terão que ser transferidos da memória principal.

**04) Para o sistema da questão 1 responda as questões a seguir:**

1. **qual o percentual da memória será ocupado pela tabela de páginas?**
2. **qual o tamanho da TLB e quais as vantagens e desvantagens desse mecanismo?**

Layout de Endereço Físico (4GB = 2^32 bits)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Physical Page Number (18) | | | | Page Offset(14) |

Layout de Endereço Virtual (2^40)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Virtual Page Number (26) | | | | Page Offset(14) |

Layout da Tabela de Tradução

|  |  |  |  |
| --- | --- | --- | --- |
| Valid(1) | Dirty(1) | PPN(18) | |

Layout da TLB

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Valid(1) | Dirty(1) | Tag(26) | PPN(18) | | | |

1. O page offset necessitará de 14 bits para endereçá-lo.
2. Para endereçar a mem. virtual, serão necessários 40 bits, resultando num VPN de 26 bits.
3. Para a memória física, serão necessários 32 bits, resultando num PPN de 18 bits.

no

**Vantagens da TLB:**

Aproveitamento da localidade temporal, reduz a penalidade de consultar a tabela de páginas na memória toda vez que houver uma troca de página.

**Desvantagens da TLB:**

Uma TLB completamente associativa é pequena e cara de ser implementada. O uso de uma TLB aumenta significativamente a complexidade do sistema para lidar com misses e penalidades, já que o lookup deve ser feito como um estado intermediário antes do acesso a memória principal.

**05) No caso de uma falta de página, a mesma é carregada na memória pelo DMA, como funciona esta técnica e quais suas vantagens e desvantagens?**

A técnica DMA serve para transportar dados pelo barramento sem que o processador precise ficar verificando se a operação já foi finalizada. No caso de falta de página, é necessário carregar a página do disco para a memória. Com o DMA, a CPU dá sinal que necessita desses dados mas não executa a transferência, normalmente deixando o DMA como controlador do barramento para realizá-la. Enquanto o DMA está ocupado transferindo os dados, o processador está livre para realizar outras tarefas, sendo essa a principal vantagem dessa técnica. Quando a transferência é finalizada, o DMA envia uma interrupção para sinalizar que o dado está pronto para uso.

As desvantagens de se implementar um DMA são a necessidade de um hardware específico (a unidade de DMA), que aumenta o custo total do sistema e a complexidade, além de contribuir para os problemas de coerência de cache.

**06) Descreva algumas características de barramentos na realização de operações de E/S considerando os aspectos abaixo:**

1. **quem sincroniza a transferência de E/S**

Sobre sincronia, podemos ter dois tipos de barramentos:

1. Síncrono: o barramento síncrono é o mais simples de ser implementado, porque todo os dispositivos estarão atuando pelo mesmo clock.
2. Assíncrono: este barramento possibilita a individualidade dos dispositivos, isso é, cada dispositivo atua na sua própria velocidade. Para isso ele utiliza de bits de checagem, que sinalizam se o dispositivo quer utilizar o barramento e se ele finalizou de usar o barramento.

**b. como os dispositivos conseguem acesso ao barramento**

No quesito acesso, normalmente no barramento é necessário uma técnica de arbitragem, mais popularmente duas são discutidas:

1. Centralizado: Nesse tipo de arbitragem, é necessário um dispositivo “árbitro” que decide quem deve usar o barramento ou não. Uma das implementações é dar prioridade aos dispositivos de acordo com sua proximidade, quanto mais perto do árbitro, maior sua prioridade.
2. Descentralizado: Já nesse tipo, não é necessário um novo componente que decide tudo, todos os dispositivos são ligados por barramento e conseguem saber se alguém está usando ou não e decidem entre si quem será o próximo a utilizá-lo.

**c. como melhorar a taxa de transferência no barramento**

Para melhorar a taxa de transferência, podemos melhorar o barramento das seguintes formas:

1. Podemos dividir o barramento em linhas de dados e de endereços.
2. Transferência por blocos.
3. Aumentar o barramento de dados.