

17.

(1) 根据 TLB 的内容, 查找虚拟地址 0x05a4 的物理地址:

虚拟地址长度为 14 比特, 页大小为 64 字节, 因此低 12 位为页内偏移。

虚拟地址的高 2 位为页号, 即 0x05。

TLB 的索引通过取虚拟地址的中间 10 位 (14 位-12 位) 的值, 即 0x02。

在索引为 2 的 TLB 条目中, 标签匹配, 且有效位为 1, 因此发生了 TLB 命中。

物理页号为 0x1C。

物理地址为: $0x1C \ll 6 \mid 0x0A4 = 0x1C0A4$ 。

(2) 该系统的页表条目数可以通过虚拟地址长度和页大小计算得出:

虚拟地址长度为 14 比特, 其中 12 比特用于页内偏移, 剩余 2 比特用于页号。

由于页大小为 64 字节, 每个页对应一个物理页框。

物理地址长度为 12 比特, 对应 2^{12} 个物理页框。

因此, 页表条目数为 $2^2 = 4$ 个条目。

(3) 根据 L1 缓存的内容, 判断虚拟地址 0x05a4 的访存请求是否命中缓存和访存结果:

虚拟地址的高 2 位为页号, 即 0x05。

虚拟地址的低 4 位为块内偏移, 即 0x4。

L1 缓存为直接映射, 组号通过取虚拟地址的中间 4 位 (14 位-2 位) 的值, 即 0x05。

在组号为 5 的 L1 缓存中, 标签匹配, 有效位为 1, 且块偏移匹配, 因此发生了缓存命中。

访存结果为 L1 缓存中对应块偏移的值, 即 0x00。

综上所述, 根据给定的 TLB 和 L1 缓存的内容, TLB 发生命中, 物理地址为 0x1C0A4, 而 L1 缓存也发生命中, 访存结果为 0x00。

18.

(1)

访存地址	A	B	C	D	A	B	C	D
way 0	-	A	C	D	D	B	A	C
命中?	N	N	N	N	Y	N	Y	N
way 1	-	-	-	-	-	-	-	-
命中?	N	N	N	N	N	N	N	N

当程序长时间运行时, 缓存的命中率为 $2/8 = 25\%$ 。

(2)

为了使上述程序在该缓存中拥有最大的命中率, 可以使用 FIFO (First-In, First-Out) 替换策略。

使用 FIFO 替换策略时, 无论程序如何循环访问四个地址, 缓存中总是保持最新的四个访问, 即使访存顺序为 A-B-C-D 或 D-C-B-A。因此, 该缓存的命中率为 $4/8 = 50\%$ 。

19.

(1) 低位标签在同一缓存组内通常被要求是唯一的, 原因如下:

缓存组内的每个缓存块都具有相同的组索引, 即它们共享同一个组索引位。因此, 为了区分不同的缓存块, 需要使用唯一的低位标签。

低位标签的作用是在组内进行快速匹配, 以确定是否存在命中。如果不同的缓存块具有相同的低位标签, 就无法准确地确定哪个缓存块被命中, 从而导致错误的

预测和数据前馈。

(2) 引入微标签技术对于通常的缓存替换策略可能会有影响，主要体现在以下方面：

缓存替换策略通常基于缓存块的标签信息进行决策。由于微标签技术将标签拆分为高位标签和低位标签，传统的替换策略需要相应地进行调整，以充分利用微标签提供的更多信息。例如，可以结合高位标签和低位标签来评估缓存块的热度和访问频率，从而更准确地进行替换决策。

微标签技术可能需要更复杂的硬件支持和额外的控制逻辑来处理低位标签和高位标签的比较、命中判断和数据前馈。因此，在引入微标签技术时，需要对缓存控制器的设计进行相应的改进和优化，以适应新的缓存结构和操作流程。

(3) 对于 16KB 页大小的内存系统和一个 8KB 大小的四路组相联缓存，我们可以计算低位标签的位数如下：

缓存大小为 8KB，即 2^{13} 字节。

组相联缓存的路数为 4 路，因此每个组中有 4 个缓存块。

每个缓存块的大小为 $2^{13} \text{ 字节} / 4 = 2^{11} \text{ 字节}$ 。

低位标签用于区分不同的缓存块，因此需要能够表示 4 个不同的缓存块，即 2^2 个不同的值。

因此，低位标签的位数为 $\log_2(2^2) = 2$ 比特。

因此，一个 8KB 大小的四路组相联缓存可以拥有 2 比特的低位标签。

20.

监听一致性和目录一致性是两种常见的缓存一致性协议，它们各有优缺点。

监听一致性 (Snooping Consistency) 是一种基于总线的缓存一致性协议。在监听一致性协议中，每个缓存都能够监视总线上的读写操作，并根据操作的地址和类型进行相应的响应和处理。优点包括：

优点：

实现简单：监听一致性协议不需要引入专门的目录表或中央控制器，每个缓存都能够通过监听总线进行状态的更新和维护，因此实现相对简单。

响应快速：由于每个缓存都能够即时监测到其他缓存的操作，并根据需要进行数据的更新或失效，因此响应速度较快。

缺点：

总线带宽限制：在监听一致性协议中，所有的缓存都需要监听总线上的读写操作，这可能导致总线带宽的瓶颈，特别是在大规模多处理器系统中。

高能耗：由于需要在总线上广播和监听大量的读写操作，监听一致性协议可能会导致较高的能耗。

目录一致性 (Directory-based Consistency) 是另一种常见的缓存一致性协议。在目录一致性协议中，引入了一个专门的目录表来记录缓存块的状态和位置信息。

优点包括：

优点：

总线带宽优化：目录一致性协议通过目录表来记录缓存块的状态和位置信息，减少了在总线上广播和监听操作的需求，从而优化了总线带宽的使用。

灵活性：由于目录表的存在，目录一致性协议可以更灵活地管理缓存块的状态和一致性信息，包括精确控制缓存块的共享状态和缓存块的失效。

缺点：

实现复杂：目录一致性协议需要引入专门的目录表，以及相关的控制逻辑和算法，

因此实现相对复杂。

响应延迟：由于需要进行目录表的查询和更新操作，目录一致性协议可能会引入额外的延迟，导致响应时间相对较长。

缓存一致性的实现代价主要体现在以下几个方面：

硬件成本：实现缓存一致性可能需要增加额外的硬件支持，包括总线监听逻辑、目录表和相关控制电路等。这些硬件成本可能增加系统的复杂性和成本。

软件复杂性：缓存一致性协议需要在软件层面进行支持和管理，包括对共享数据的同步和一致性保证。这可能增加软件开发和维护的复杂性。

性能开销：为了实现缓存一致性，可能需要引入额外的延迟和开销，例如在读写操作时进行缓存状态的查询和更新，以及在共享数据访问时进行同步操作。这些额外的开销可能会影响系统的性能和响应时间。

综上所述，监听一致性和目录一致性各有优缺点，选择适合的缓存一致性协议需要考虑具体的应用场景和系统要求。同时，实现缓存一致性需要在硬件、软件 and 性能方面进行权衡和优化。