

}

假设系统中 int 变量为 4 字节，input 数组的起始地址为 0x0。系统仅在对 input 数组的读写时会访问缓存，其他变量全部位于寄存器中。则：

- 1) 如果存在一个 512 字节大小的缓存，该缓存两路组相联，块大小为 16 字节，使用 LRU 替换策略，缓存初始为空。则执行上述程序时缓存的命中率为多少？
- 2) 在其他条件不变的情况下，增加缓存的总大小可以改善该程序的命中率吗？请说明理由。
- 3) 在其他条件不变的情况下，增加缓存的块大小可以改善该程序的命中率吗？请说明理由。

17. 假设一个使用虚拟内存和 L1 缓存的存储系统具有以下特征：

- a) 内存系统按字节寻址，访存请求每次仅传递一个字节给处理器。
- b) 虚拟地址长度 14 比特，物理地址长度 12 比特。  
块内偏移：2 位  
索引：4 位  
标签：8 位
- c) 页大小 64 字节，使用单级页表。
- d) TLB 拥有 16 个条目，四路组相联。  
4 组  
标签：6 位
- e) L1 缓存物理寻址，块大小 4 字节，共 16 个组，直接映射。  
0x05a4: 000/0110 100/00

现在 CPU 发起了一次对虚拟地址 0x05a4 的单字节内存加载请求，回答以下问题。

- 1) 若请求发起时，TLB 的部分内容如下表所示。则 TLB 是否发生命中？如果命中，此次内存访问的物理地址是多少？

组号	标签	物理页号	有效位	标签	物理页号	有效位
0	0x0B	—	0	0x1F	—	0
	0x07	0x0D	1	0x02	0x2F	1
1	0x01	0x05	1	0x05	0x0D	1
	0x14	—	0	0x2A	0x16	1
2	0x03	—	0	0x05	0x1C	1
	0x0B	0x07	1	0x00	0x1B	1
3	0x26	0x34	1	0x02	—	0
	0x19	0x2F	1	0x38	—	0

- (2) 该系统的页表有多少个条目？

- 3) 如果 TLB 命中，则使用 1) 得到的物理地址，否则使用物理地址 0x1e4。如果 L1 缓存的内容如表所示，则此次访存请求是否命中缓存？如果命中，访存结果是多少？

组号	标签	有效位	块偏移			
			0x0	0x1	0x2	0x3
0	0x1F	0	—	—	—	—
1	0x05	1	0x02	0x09	0xCB	0xA3
2	0x1C	1	0x09	0x55	0x01	0x08
3	0x0D	0	—	—	—	—
4	0x1B	1	0x9B	0xEE	0xE2	0x86
5	0x2F	1	0x00	0x00	0x01	0x00
6	0x07	0	—	—	—	—
7	0x05	1	0x6F	0x23	0xAB	0xD0
8	0x16	0	—	—	—	—

9	0x1C	1	0x63	0x2F	0x1B	0x00
10	0x1C	1	0x28	0x34	0x01	0xC4
11	0x16	1	0x29	0xC8	0x56	0x99
12	0x34	0	—	—	—	—
13	0x34	0	—	—	—	—
14	0x0D	0	—	—	—	—
15	0x07	1	0xE8	0x59	0x04	0x45

18. 一段程序循环往复地按顺序访问 A、B、C、D 四个地址上的数据。考虑一个拥有 2 条目的全相联缓存，回答以下问题。

- 1) 使用 LRU 替换策略时，填写下表。当程序长时间运行时，缓存的命中率为多少？

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

- 2) 提出一种缓存替换策略，使得上述程序可以在该缓存中拥有最大的命中率，并计算该命中率。*Random replacement 随机替换*

$$\text{命中率为 } \frac{1}{2} \times \frac{1}{2} \times \frac{1}{2} \times \frac{1}{2} = \frac{1}{16}$$

19. 一些处理器引入了“微标签”(microtag)的技术来降低组相联缓存标签匹配过程的时序压力。该技术将地址的标签部分进一步拆分为高位标签(HTag)和低位标签(LTag)，在判断缓存命中与否时，控制器仅取出低位标签进行比较，将匹配的缓存块预测为一次命中并把数据前馈给处理器。在随后的剩余周期内，高位标签被取出并进一步用于判断该预测最终是否构成真正的命中。回答以下问题：

- 1) 低位标签在同一缓存组内通常被要求是唯一的，试说明原因。
- 2) 基于对 1) 的讨论，简要说明该技术的引入对于通常的缓存替换策略有什么影响。
- 3) 考虑到虚拟页偏移和物理页偏移是一致的，为了提高访存性能，系统可以进一步要求地址的低位标签和组索引位完全位于页偏移字段内，这样低位标签的匹配过程就完全不需要经过地址翻译而可以直接进行，后续的高位标签则使用页表翻译后的结果判断是否构成真实命中。基于上述过程，对于 16KB 页大小的内存系统，一个 8KB 大小的四路组相联缓存至多可以拥有几比特的低位标签？

20. 监听一致性和目录一致性各有什么优缺点？简述缓存一致性的实现代价体现在哪些方面？

19. 1) 如果低位标签在同一缓存组内不唯一，即多个缓存块具有相同的低位标签，那么在进行标签匹配时，会存在冲突和不确定性，无法准确识别哪个缓存块是真正匹配的。这会导致缓存命中率下降，性能受到影响。低位标签的唯一性保证了在同一缓存组内每个缓存块都有唯一的标识，可以准确地匹配对应的数据块。

2) 当缓存要写入某个位置的数据时，拥有相同低位标签的数据必须被替换。

3) 块大小是多少？没法确定呀。

1) 监听一致性是基于缓存之间相互监听和通信的机制。当一个缓存对共享数据进行修改时，它会向其他缓存发送失效通知，让其他缓存将对应的数据置为无效状态。这样可以保证缓存中的数据始终保持一致。优点是实时性好，只有发生共享数据修改时才进行通信，可以减少不必要的时间开销。缺点是通信开销较大，当系统中缓存数量较多时，通信负载会增加。

目录一致性是通过维护一个全局的目录表来实现缓存一致性。目录表记录了每个缓存块的状态信息，包括是否被修改、是否被共享等。当一个缓存对共享数据进行修改时，它需要先向目录表发送请求，并等待目录表的响应，才能进行数据的修改。目录表会根据请求的类型和当前的缓存状态来更新相应的信息。优点是通信开销相对较小，只需要与目录表进行通信，不需要与其他缓存直接通信。缺点是目录表的维护和更新可能引入较大的开销，并且访问目录表的延迟可能会影响整体性能。

2) 缓存一致性的实现代价体现在以下几个方面：

需要额外的硬件支持：实现缓存一致性需要在处理器和缓存之间增加一些额外的硬件支持，例如监听机制或目录表。这些硬件的设计和实现会增加成本和复杂性。

通信开销：缓存之间的通信是实现一致性的关键，但它也会引入一定的通信开销。通信开销包括发送请求、等待响应、传输数据等，这些都会占用系统的带宽和延迟。

同步延迟：为了保证一致性，需要在访问共享数据之前进行同步操作，例如等待缓存失效或目录表的响应。这些同步操作会引入一定的延迟，可能会降低系统的吞吐量。