访问 k 个 PTE，第一眼看上去昂贵而不切实际。然而，这里 TLB 能够起作用，正是通过将不同层次上页表的 PTE 缓存起来。实际上，带多级页表的地址翻译并不比单级页表慢很多。

9.6.4　综合：端到端的地址翻译

在这一节里，我们通过一个具体的端到端的地址翻译示例，来综合一下我们刚学过的这些内容，这个示例运行在有一个 TLB 和 L1 d-cache 的小系统上。为了保证可管理性，我们做出如下假设：

- 内存是按字节寻址的。
- 内存访问是针对 1 字节的字的(不是 4 字节的字)。
- 虚拟地址是 14 位长的($n=14$)。
- 物理地址是 12 位长的($m=12$)。
- 页面大小是 64 字节($P=64$)。
- TLB 是四路组相联的，总共有 16 个条目。
- L1 d-cache 是物理寻址、直接映射的，行大小为 4 字节，而总共有 16 个组。

图 9-19 展示了虚拟地址和物理地址的格式。因为每个页面是 $2^6=64$ 字节，所以虚拟地址和物理地址的低 6 位分别作为 VPO 和 PPO。虚拟地址的高 8 位作为 VPN。物理地址的高 6 位作为 PPN。

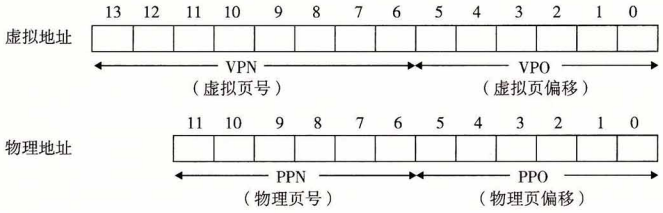

图 9-19　小内存系统的寻址。假设 14 位的虚拟地址($n=14$)，12 位的物理地址($m=12$)和 64 字节的页面($P=64$)

图 9-20 展示了小内存系统的一个快照，包括 TLB(图 9-20a)、页表的一部分(图 9-20b)和 L1 高速缓存(图 9-20c)。在 TLB 和高速缓存的图上面，我们还展示了访问这些设备时硬件是如何划分虚拟地址和物理地址的位的。

- TLB。TLB 是利用 VPN 的位进行虚拟寻址的。因为 TLB 有 4 个组，所以 VPN 的低 2 位就作为组索引(TLBI)。VPN 中剩下的高 6 位作为标记(TLBT)，用来区别可能映射到同一个 TLB 组的不同的 VPN。
- 页表。这个页表是一个单级设计，一共有 $2^8=256$ 个页表条目(PTE)。然而，我们只对这些条目中的开头 16 个感兴趣。为了方便，我们用索引它的 VPN 来标识每个 PTE；但是要记住这些 VPN 并不是页表的一部分，也不储存在内存中。另外，注意每个无效 PTE 的 PPN 都用一个破折号来表示，以加强一个概念：无论刚好这里存储的是什么位值，都是没有任何意义的。
- 高速缓存。直接映射的缓存是通过物理地址中的字段来寻址的。因为每个块都是 4 字节，所以物理地址的低 2 位作为块偏移(CO)。因为有 16 组，所以接下来的 4 位就用来表示组索引(CI)。剩下的 6 位作为标记(CT)。