

## 1. 简述现代计算机系统需要存储层级的原因。

在快速和高容量间找到平衡，提高计算机系统的性能和效率。计算机系统的存储层次一般包括多级缓存、主存储器和辅助存储器。存储层次的设计是基于存储介质的特性和访问速度的不同。越接近处理器的存储介质速度越快，但成本也越高。因此，通过在处理器和主存之间增加多级缓存，可以利用高速缓存快速缓存数据，从而减少对主存储器访问，提高访问速度。

此外，辅助存储器（如硬盘、固态硬盘等）容量大价格便宜，但速度比主存储器慢很多，因此常用作长期存储和备份。操作系统通常使用虚拟内存的机制来管理主存储器和辅助存储器之间数据传输，以支持大型程序和多任务处理。

## 2. 在页式虚拟存储中，过大或过小的页分别会引起什么问题？

① 页太大：1° 内存碎片化，由于大页需要连接物理内存空间，当物理内存中没有足够大的连续空闲内存时，无法分配大页，这将导致内存碎片化。

2° 交换开销增加，页越大，读取磁盘次数越少，但每次读写数据量也越大，交换开销大。

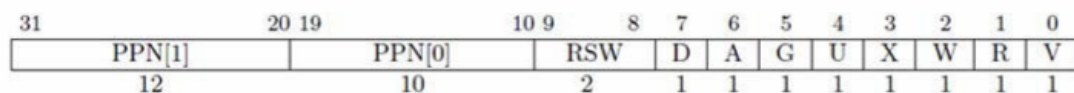
3° 数据不够利用，若一个页中只有少数数据被使用，那么使用大页会导致更多内存浪费。

② 页太小：1° 页表过大：页太小，需要更多页表项，将

导致页表过大, 占用过多内存.

2° 上下文切换开销增加: 页小, 一个进程可能需要大量的页, 这意味更频繁进行上下切换, 系统开销增加

3. 页表条目除了保存物理页号外, 一般还包含各种状态和权限标记位。它们为内存访问提供了各种细粒度的控制。例如, RISC-V 指令集的 Sv32 页表条目具有如下的形式:



- 1) 查阅 RISC-V 规范, 简要描述上述条目中的位 7 至位 0 具有什么功能。
- 2) 结合上述功能讨论: 如果用户进程能够自由修改自己的页表, 会发生什么问题?
- 3) 在 RISC-V 的虚拟内存管理中, 一个 X/W/R 位全部为 0 的有效页表条目有什么含义?

(1) V: Valid(有效位), 用于指示当前页表是否有效, 如果该位为 0, 则表示该页表项无效, 虚拟地址对应的 4kB 空间还没有被操作系统映射到物理内存中。若该位为 1, 则该页存在。

R: Read (读权限位), 用于指示当前页的读权限, 0 不可读, 1 可读。

W: Write (写权限位), 0 不可写, 1 可写

X: Execute (执行权限位), 0 不可执行, 1 可执行

U: User (用户权限位), 0 表示当前页只能被内核访问, 1 表示当前页可以被用户程序访问。

G: Global (全局位), 0 表示当前页不是全局页, 即当前页只在进程的地址空间中有效, 1 则相反。

A: Accessed (访问位), 0 表示未被访问过, 1 则相反

D: Dirty (脏页位), 用于指示当前页是否被修改过  
0表示未修改过, 1相反

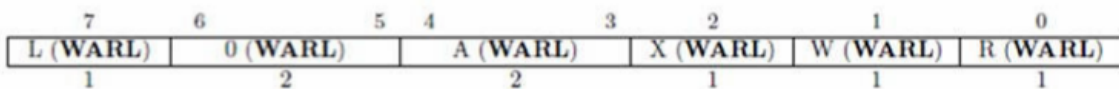
(2) 安全问题是, 用户进程通过修改页表, 获取对内核或其他进程的非法访问权限;

内存问题, 修改页表可能会导致内存泄漏或内存溢出, 如可能会将自己的栈区或堆区的页面标记为可执行, 从而使得栈或堆中数据被错误当作代码执行, 导致程序崩溃或泄漏敏感信息.

系统稳定性问题. 若将某些页面权限设置为无效或者非法, 从而导致程序访问这些页面异常.

(3) X/R/W全为0表示该页面是一个已经被释放的页面, 但对应的页表项仍然存在于页表中

4. RISC-V 的物理内存保护 (PMP) 机制允许硬件线程为特定的物理内存区域指定访问权限, 其配置寄存器有如下的形式:



查阅 RISC-V 规范, 回答以下问题:

- 1) 在页表条目中已经存在 X/W/R 位的情况下, PMP 控制寄存器中的 X/W/R 位有什么作用?
- 2) 说明 PMP 配置寄存器中的 L 和 A 位有什么作用。

1) 用来控制物理内存区域的访问权限。若页表中 X/W/R 位与 PMP 机制控制寄存器中 X/W/R 不同时, 根据二者权限取较小值。

(2) L (Lock) 位 L 为 1, PMP 将所描述内存区域将

被锁定，可以防止PMP 页已被非法篡改。

A (Address Matching) 位。A为1。该PMP条目所描述内存区域将与物理地址进行匹配，而不是与虚拟地址进行匹配。

5. 回答以下问题：

- 1) 如果页大小为4KB，每个页表条目使用8字节空间，内存系统按字节寻址。则使用完整的64位虚拟地址时，一个单级页表系统需要多大的空间用于存储页表？
- 2) 实际上，多数真实系统仅限制使用64位系统的一部分位作为有效的访存空间，例如Sv48即仅使用48位的虚拟地址空间，则保持其他假设不变时，一个单级页表系统存储页表所需要的空间被降低到多少？
- 3) 多级页表为什么可以降低虚拟内存系统的实际页表存储开销？

(1) 虚拟地址的寻址空间是 $2^{64}$ 字节。

为了能够映射整个 $2^{64}$ B空间，需要表项的个数为 $\frac{2^{64}}{2^{12}} = 2^{52} \text{ B} = 2^{22} \text{ GB}$

页表大小就是 $8 \text{ B} \times 2^{22} \text{ GB} = 2^{25} \text{ GB}$

(2) 需要表项个数 $\frac{2^{48}}{2^{12}} = 2^6 \text{ GB}$

页表大小是 $8 \text{ B} \times 2^6 \text{ GB} = 2^9 \text{ GB}$

(3) 空间局部性：多级页表通过将页表划分为多个层级。每个层级只包含一部分页表项，使得每个页表的大小更小。当进程访问一个连续的虚拟地址范围时，只需加载相应的页表层级，而不是整个页表，从而减少了存储开销。

延迟分配：只在需要时才分配物理内存，当进程创建新的虚拟页时，只需在页表中创建对应的页表项，不需要为未使用的虚拟页分配实际物理空间。