

#### 第4章习题

##### 1. 简述现代计算机系统需要存储层级的原因

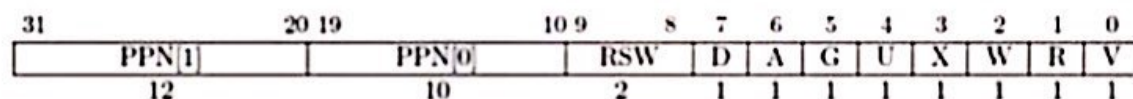
随着现代处理器性能不断攀升，处理器和内存之间的速度差距不断扩大，形成“内存墙”的问题。为此，现代计算机使用多级分层存储结构，并引入存储层级，利用数据的时间局部性和空间局部性来改善访存性能。

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?

页太大: 每个进程需要巨大的内存空间用于存储页表, 对于拥有数以百计的进程的计算机来说, 硬件开销巨大, 内存浪费较大

页太小: 降低访问效率, 如果频繁翻译页地址, 系统运行会耗费较多时间; 虽然可能利用了存储, 但不利于数据的连续访问

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) D: dirty bit, 当页被写后, 置对应 PTE 中 D 位

A: access bit, 当页被读, 写或正被获取, 置对应 PTE 中 A 位

G: global mapping, 标记全局映射

U: 是否对用户模式可访问

PTE: page table entry

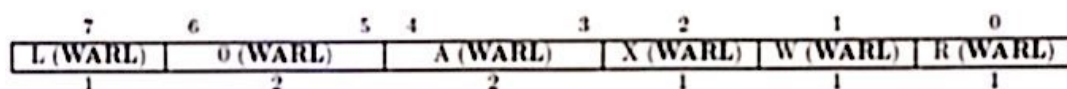
X, W, R: 标记是否可以读、写、执行

V: 标记 PTE 是否有效

2) 如果用户进程能自由修改自己的页表, 很有可能破坏其他进程的物理数据存储器, 甚至可通过 r, w, a 等位锁定某些内存禁止访问造成严重后果

3) 该页不可读、不可写、不可访问

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) Bits Description

0	R: Read permissions	{ 0x0: No read permission 0x1: read permission
1	W: Write permissions	{ 0x0: No write permission 0x1: Write permission
2	X: execute permissions	{ 0x0: No execute permission 0x1: execute permissions

如果内存映射的非保留区域没有应用 PMP 权限, 则默认情况下, 管理员或用户访问模式将失败, 而机器模式访问将允许。访问设备内存映射中的保留区域读取时将返回 0x0, 写入将被忽略。在没有 PMP 保护的情况下访问设备内存之外的保留区域将导致总线错误。

2) L: Locked Bit: { 0x0: Entry Unlocked  
0x1: Entry locked

A: Address matching code { 0x0: PMP 禁止访问  
0x1: TOR 区域被定义为两个相邻的寄存器  
0x2: Support only a four-byte region with only a four-byte granularity  
0x3: Naturally aligned power-of-two region

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) 页大小为 4KB, 则页内索引位数 12 位, 虚拟页号占剩余 52 位  
每个页表条目占用 8 字节,  $\therefore$  所需存储空间  $2^{52} \times 8 = 2^{55}$  字节

2) 虚拟页号占剩余的 36 位, 同 1) 一个单级页表所需存储空间为  $2^{39} = 512G$

3) 多级页表的优势在于能跟随进程占用内存空间的增大, 对地址增多属于该进程的页表数目; 而当进程占用空间很小, 页表数目也小, 以此减小页表占用的内存空间, 大大减少了硬件开销