

1. 简述现代计算机系统需要存储层级的原因。

- ① CPU的运算速度比内存,硬盘等存储设备的速度快得多,如果不使用一下存储模块,会使CPU浪费很多等待时间,影响运算效率。
- ② 由于存储介质需要成本,不可能给所有数据都使用最快的存储,需要根据具体数据而分配不同层级的存储。
- ③ 计算机系统处理任务具有并发性,为了避免任务的干扰和冲突,需要将任务分配到不同的内存空间和磁盘空间。

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 如果页过大,只有当前页全部装入内存后才会被访问,会造成内存浪费。如果出现缺页中断,如果页很大,需要花更长时间读取,影响系统性能。
- ② 如果页过小,每个进程会产生很多页目录和页表项,会导致内存开销过大。过小的页还会增加缺页中断的可能。

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:

31	20 19		10 9		8	7	6	5	4	3	2	1	0	
PPN[1]				PPN[0]		RSW	D	A	G	U	X	W	R	V
12				10		2	1	1	1	1	1	1	1	1

- 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?

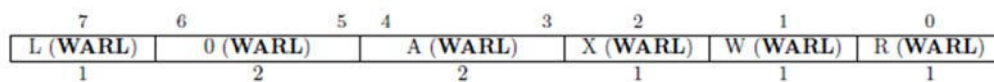
1). 0: V 位标志表示 PTE 是否有效,如果 V=0, 可以不关心其他位。

- 1. R: page 是否可读 2. W: page 是否可写 3. X: page 是否可执行
- 4. U: u-mode 是否可访问该页面 5. G: 全局映射
- 6. A: 自上次读以来,虚拟 page 已被读取或 fetched
- 7. D: 自上次读以来,虚拟 page 已被写入

2) 如果用户可以自由修改页表,那么页表可能会映射到内存中的全部位置,产生安全问题。

3) 当 X/W/R 都为 0 时,表示此 PTE 指向下级的 page table, 否则它就是 leaf PTE, pointer to next level of page table

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) R, W, X 分别对应读、写、执行权限。

RISC-V 提供三种权限模式 M, S, U, 为限制不可信的代码使其只能访问自己的内存。

2) A 0 PMP 未启用, 不匹配地址

1 TOR Top of range

2 NA4 Naturally aligned four-byte region

3 NAPOT Naturally aligned power-of-two region

L 1 M, S, U 模式下必须遵循配置寄存器的权限设置

0 M 模式下任何匹配到 PMP entry 的操作即成功

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) 64 位虚拟地址寻址 2^{64} B $\frac{2^{64}}{2^{12}} \times 8 = 2^{55}$ 个字节空间

2) 48 位虚拟地址寻址 2^{48} B $\frac{2^{48}}{2^{12}} \times 8 = 2^{39}$ 个字节空间

3) 多级页表只记录进程实际使用的虚拟地址内存区请求页表, 从而减少内存使用量