

1. 主要原因是为了^在提供更快访问速度、提供更大容量、所需成本更低三者矛盾中达到平衡, 可较好地解决容量要大、速度要快、成本又较合理的矛盾

1. 访问速度上, CPU中寄存器的速度远快于内存, 而内存速度又远快于外存. 多层存储层次结构引入了高速缓存等中间层次, 缩小各层间的速度差异使每层都相对较快, 以提供更快的数据访问速度.

2. 容量上, 通过层次化使容量速度匹配, 逐级提升以达更大容量.

3. 成本上: 不同类型的存储介质在性能和成本方面存在着权衡. 例如高速缓存成本高而外存成本很低. 通过存储层次对不同速度的存储设备在性能和成本间找到一个平衡点.

2. 页如果太大, 虽然可以减小页表大小, 但每个进程需要访问的页数会减少, 这会导致许多页内部的数据未被充分利用, 每次页面装入或换页操作会变得非常缓慢, 进而影响系统性能.

页如果太小, 会造成页表项太多, 页表体积过大, 其自身占内存过大, 导致系统效率大幅下降.

3. (1) 第0位V: valid ^{有效} 是否有效 第1位R: readable 此页是否可读
第2位W: writeable 此页是否可写 第3位X: executable 此页是否可执行
第4位U: 访问权限等级, 是否可在用户模式下访问 第5位G: 是否指定全局映射
第6位A: accessed 此页是否被访问过 第7位D: dirty 此页是否被写脏.

(2) 将可以改变自己页表的U、G位, 提升权限等级, 如把非全局映射标记为全局的将是一个错误; 将用户进程的页表项修改为指向其他进程的内存位置而无法实现权限机制. 无法隔离等.

(3) X/W/R 均0 为无效的有效页表条目是下级页表的指针.



4. 表示访问实际内存区域时是否允许执行/写入/读取。若PMP控制寄存器中X/W/R位与页表条目的相冲突,则以PMP中的权限为准。

(2) L: 用于锁定PMP表项,被设置时任何写入将被忽略

A: 用于指定地址匹配模式(NAPOT, NA4, TOR, OFF),该PMP寄存器将依此应用于对应的地址范围内(灵活)

5. (1) 页大小4KB \rightarrow 索引位数为 $\log_2(4 \times 2^{10}) = 12$

使用64位虚拟地址 共要表项 $2^{64-12} = 2^{52}$ 个 单位

最多 $2^{52} \times 8 \text{ Byte} = 2^{55} \text{ Byte} = 2^{25} \text{ MB}$.

(2) 使用48位虚拟地址 共要表项 $2^{48-12} = 2^{36}$ 个

$2^{36} \times 8 \text{ Byte} = 2^{39} \text{ Byte} = 0.5 \text{ GB}$.

13) 多级页表能随着进程占用内存空间的增大而动态地增多属于该进程的页表数目(减少也是同理),可以动态调整,省却未映射页表项。另外可将后级页表的数据先放在磁盘,使用时再加载,这些等都可节省存储开销。

