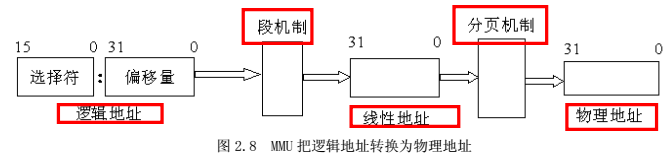
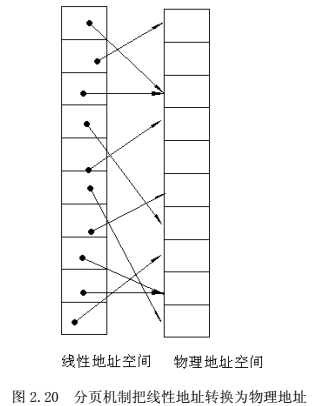
# 内存地址





逻辑地址

每个逻辑地址都由一个段和偏移量组成：CS + offset

线性地址

线性地址是一个 32 位的无符号整数，可以表达高达 4GB的地址。通常用 16 进制表示线性地址，其取值范围为 0x00000000～0xffffffff。

物理地址

物理地址是内存单元的实际地址，用于芯片级内存单元寻址。物理地址也由 32 位无符

号整数表示。

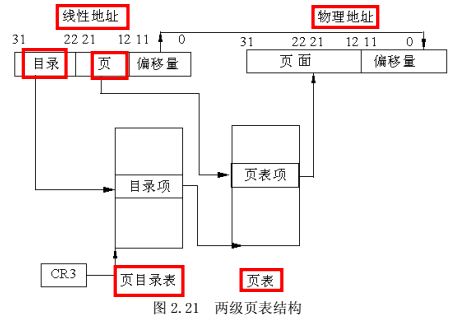
## 分页机制

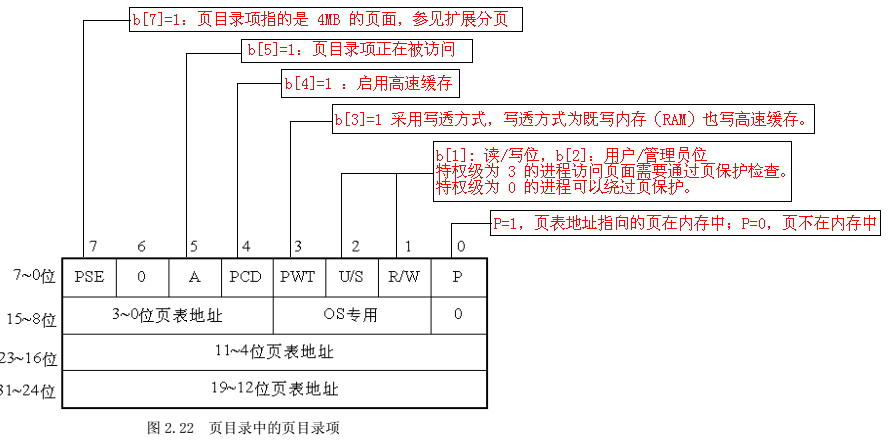
分页机制把整个线性地址空间及整个物理地址空间都看成由页组成，在线性地址空间中的任何一页，可以映射为物理地址空间中的任何一页。

分页是将程序分成若干相同大小的页， 每页 4K 个字节。 如果不允许分页 （CR0的最高位置 0），那么经过段机制转化而来的 32 位线性地址就是物理地址。但如果允许分页（CR0 的最高位置 1），就要将 32 位线性地址通过一个**两级表格结构**转化成物理地址。

两级表结构的第一级称为**页目录**，存储在一个 4K 字节的页面中。页目录表共有 **1K 个表项**，每个**表项为 4 个字节**，并指向第二级表。线性地址的最高 10 位（即位 31～位 22）用来产生第一级的索引，由索引得到的表项中，指定并选择了 1K 个二级表中的一个表。

两级表结构的第二级称为**页表**，也刚好存储在一个 4K 字节的页面中，包含 **1K 个字节的表项**，每个表项包含一个页的物理基地址。第二级页表由线性地址的中间 10 位（即位 21~位12）进行索引，以获得包含页的物理地址的页表项，这个物理地址的高 20 位与线性地址的低12 位形成了最后的物理地址， 也就是页转化过程输出的物理地址。



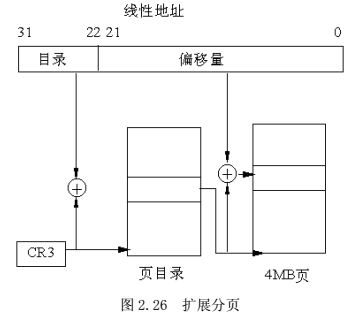




## 扩展分页

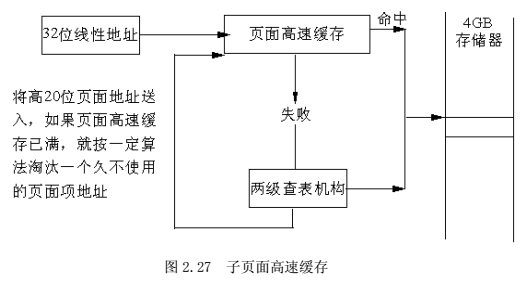
允许页的大小为 4MB。非扩展分页的也大小为4K。

在扩展分页的情况下，分页机制把 32 位线性地址分成两个域：最高 10 位的目录域和其余 22 位的偏移量。



## 页面高速缓存

由于在分页情况下，每次存储器访问都要存取两级页表，这就大大降低了访问速度。所以，为了提高速度，在 386 中设置一个最近存取页面的高速缓存硬件机制，它自动保持32 项处理器最近使用的页面地址，因此，可以覆盖 128K 字节的存储器地址。当进行存储器访问时，先检查要访问的页面是否在高速缓存中，如果在，就不必经过两级访问了，如果不在，再进行两级访问。平均来说，页面高速缓存大约有 98%的命中率，也就是说每次访问存储器时，只有 2%的情况必须访问两级分页机构。这就大大加快了速度，页面高速缓存的作用如图 2.27 所示。



## 三级分页模式

针对64 位结构的处理器，Linux 采用三级分页模式。

* 总目录 PGD（Page Global Directory）
* 中间目录 PMD（Page Middle Derectory）
* 页表 PT（Page Table）

PGD、PMD 及 PT 表的表项都占 4 个字节

