**内存机制**

逻辑地址->线性地址->物理地址

  逻辑地址：程序产生的与段相关的偏移地址部分。

  线性地址：32位无符号数，可以用来表达4GB地址。

  物理地址：内存芯片级内存单元寻址。

**1.分段**

逻辑地址由段选择符和偏移量组成，偏移量指明了从段开始的地方到实际地址之间的距离。

**段寄存器：**处理器提供段寄存器用于存放段选择符。包括cs（代码），ss（程序栈），ds（静态数据或全局数据）等6个寄存器。x86系统还提供一种

非编程寄存器用于保存上述六种寄存器中的段选择符，然后段的逻辑地址转换就不需要访问描述符表，可以直接从这个寄存器中获取。

**段描述符**：8字节，描述段的特征，放在全局描述符表或局部描述符表中。

      Base：段手字节的线性地址   Limit：决定段的大小  Type：类型   D/B（取决于数据还是代码段）：段偏移量的长度

**过程:**先检查逻辑地址的段选择符已查看该段的段描述符放在哪个描述表中,再根据描述表的base地址加上偏移量得到了线性地址.

**LInux中的分段:**Linux中有四个段,内核数据段,内核代码段,用户数据段,用户代码段,而这四个段的Base都是0x00000000,也就是说Linux的逻辑地址与线性地址是一致

的,逻辑地址的偏移量就是线性地址的值.这样分段，避免了逻辑地址到线性地址的转换步骤(逻辑地址就等于线性地址)，但仍然保留了段的等级这层最基本保护。

**2.分页**

线性地址被分为固定长度的组,称为页.页内部连续的线性地址被映射到连续的物理地址中,内核可以直接指定整个页的存取权限.

    把线性地址映射到物理地址中的数据结构称为页表,页表也存放在主存中.

    线性地址的转换:32位的线性地址被分为三个域 目录(最高10位) 页表(10位) 偏移量(最低12位)  正在使用的页目录的物理地址存放在寄存器cr3中,

线性地址内的Directory字段决定页目录的目录项,目录项指向适当的页表,table字段又决定具体的表项,表项含有页的物理地址,再根据偏移量确定物理

地址.每个活动进程都有一个页目录,内核不会马上为进程用到的所有页表都分配RAM,只有进程实际需要一个页时才给该页分配RAM.

    64位系统会采用多级分页机制.

**Linux:**

    每个进程有4G虚拟内存空间(线性地址)，每次访问内存空间的某个地址，都需要把地址翻译为实际物理内存地址

 Linux中每个进程都会有各自不同的页表，不同进程的映射函数互不相同，保证每个进程虚拟地址不会映射到相同的物理地址上。

    所有进程共享同一物理内存，每个进程只把自己目前需要的虚拟内存空间映射并存储到物理内存上。

    进程要知道哪些内存地址上的数据在物理内存上，哪些不在，还有在物理内存上的哪里，需要用页表来记录

    页表的每一个表项分两部分，第一部分记录此页是否在物理内存上，第二部分记录物理内存页的地址

    当进程访问某个虚拟地址，去看页表，如果发现对应的数据不在物理内存中，则缺页异常

    缺页异常的处理过程，就是把进程需要的数据从磁盘上拷贝到物理内存中，如果内存已经满了，没有空地方了，那就找一个页覆盖，如果被覆盖的页曾经被修改过，需要将此页写回磁盘

**3.实现虚拟内存**

　　虚拟内存是将系统硬盘空间和系统实际内存联合在一起供进程使用,给进程提供了一个比内存大得多的虚拟空间。在程序运行时，只要把虚拟地址空间的一小部分映射到内存，其余都存储在硬盘上（也就是说程序虚拟空间就等于实际物理内存加部分硬盘空间）。当被访问的虚拟地址不在内存时，则说明该地址未被映射到内存，而是被存贮在硬盘中，因此需要的虚拟存储地址随即被调入到内存；同时当系统内存紧张时，也可以把当前不用的虚拟存储空间换出到硬盘，来腾出物理内存空间。页机制可以很好的实现这种换入换出。