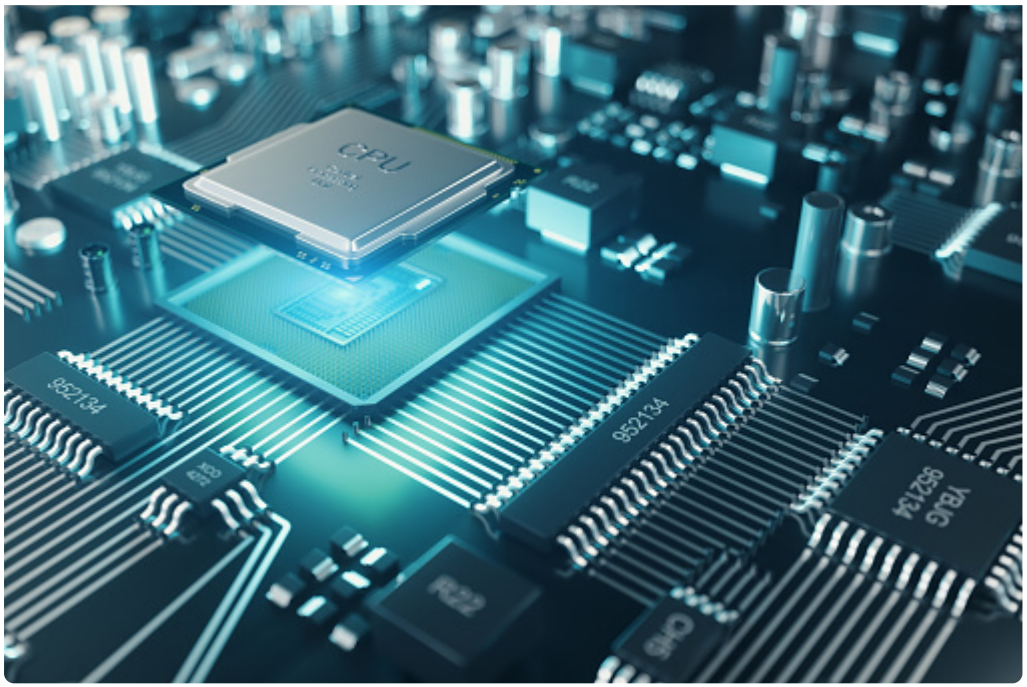
CPU瞒着内存竟干出这种事

还记得我吗，我是阿Q，CPU一号车间的那个阿Q。

今天忙里偷闲，来到厂里 地址翻译部门 转转，负责这项工作的小黑正忙得满头大汗。

看到我的到来，小黑指着旁边的座椅示意让我坐下。



坐了好一会儿，小黑才从工位上忙完转过身来，“实在不好意思阿Q，今天活太多，没来得及招待你”

“刚忙什么呢，看你满头大汗的”，我问道。

“嗨，别提了，老是发现内存页面错误，不停地要通知操作系统那边去处理，真是怀念以前啊，没有这么多破事儿要管”，小黑叹了口气。

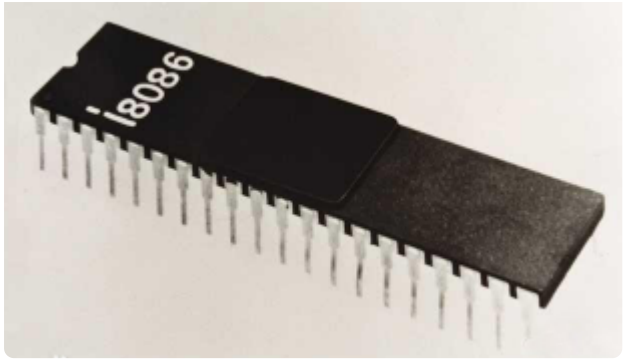
我一听来了兴趣，“小黑你给我说说你们的工作呗，地址翻译是怎么一回事儿，为什么怀念以前呢？”

小黑调整了下坐姿，咕噜咕噜喝了几口水说道，“这话说来可就话长了”

接下来小黑开始给我讲起了历史故事……

8086

原来咱们的祖先叫8086，小黑还给我看了他的照片



那是一个纯真质朴的年代，虽然工作性能不高，不过那个年代的程序都很简单，我们的祖先一问世就成为了明星，称得上那个时代的顶流了。

看到照片中的那些金属针脚了吗？那是我们CPU和外界打交道的触角，每一根都有不同的作用。

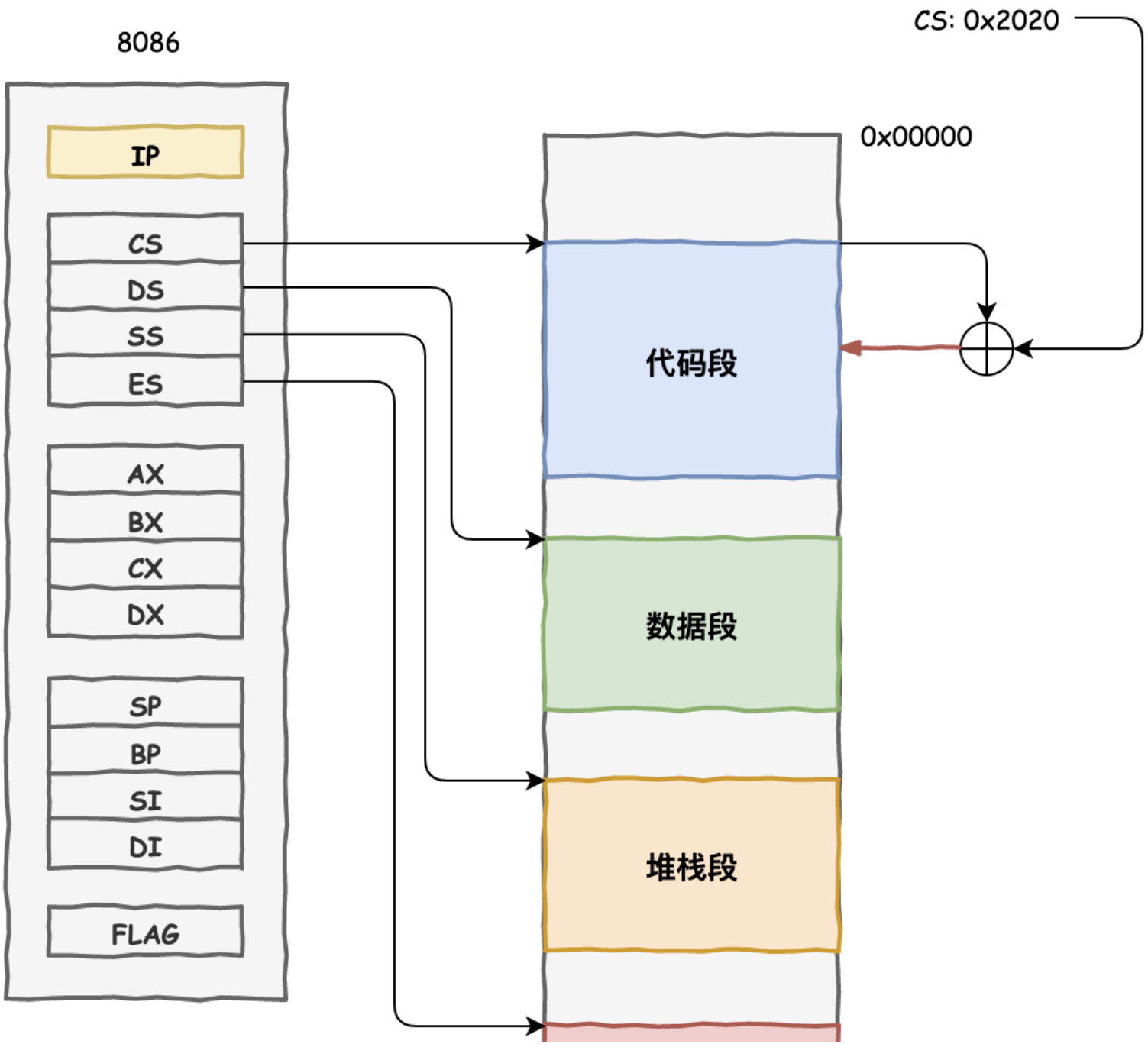
通过这些触角，CPU就可以跟内存打交道，获取指令和数据，辛勤的干活啦。

那个年代，条件比较差，能凑合的就凑合，能共用的就共用。这不，你看祖先CPU的地址总线针脚和数据总线针脚就共用了。

祖先是一个16位的CPU，数据(Data)总线就有16位，一次性可以传输16个比特位。和地址(Address)总线凑合着一起共用，于是就取名AD0-AD15。

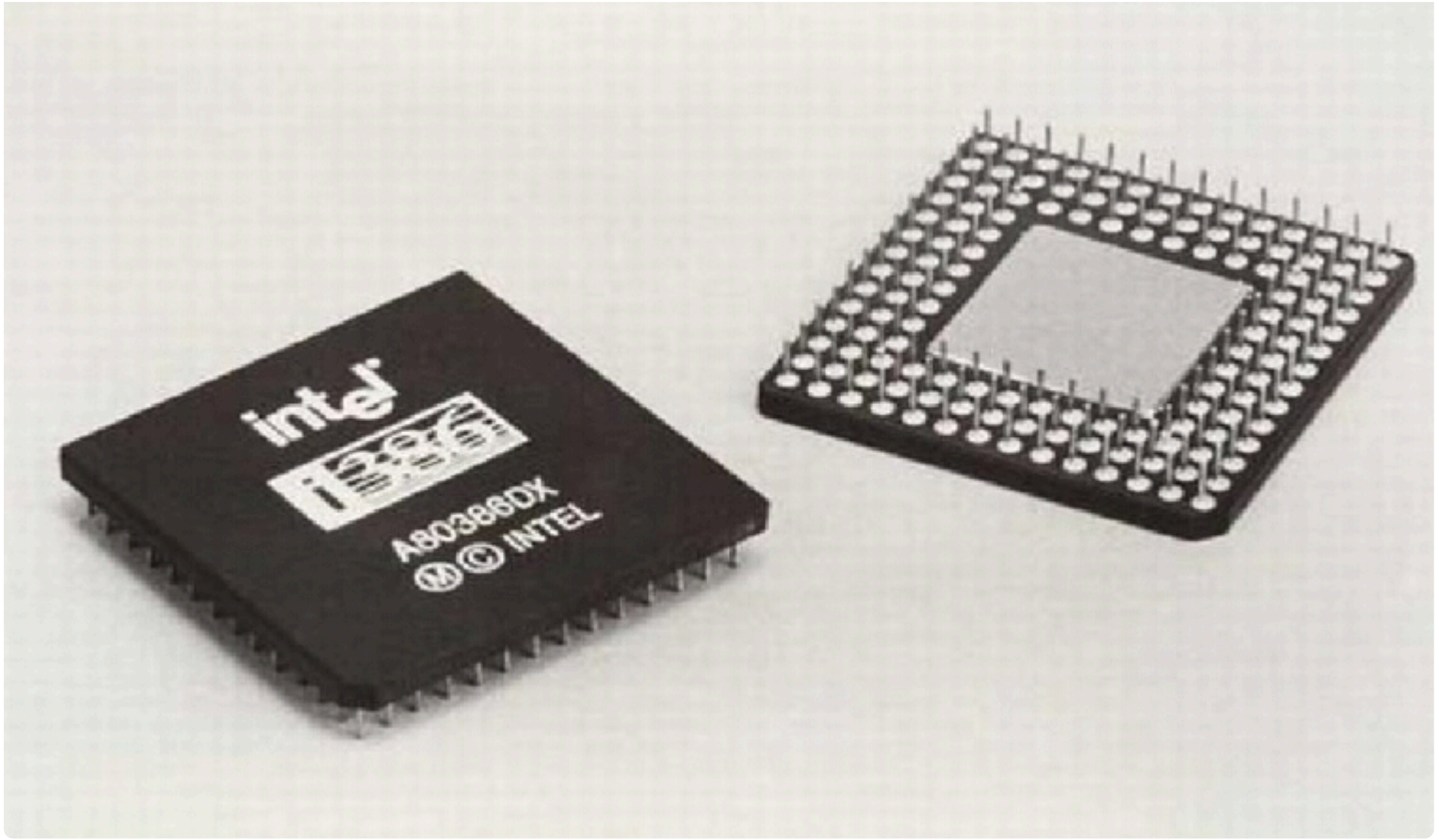
不过祖先的地址总线却不止16个，还多出了A16-A19整整4个呢！这样有20个地址线，可以寻址1MB的内存了！

但是祖先的寄存器都是16位的啊，只能存放16位的地址。不过他们很聪明，发明了一个叫 分段式存储管理 的方法，把内存划分为最大64KB的小块，为什么是64KB呢，因为16位地址最多只能寻址这么大了。然后又加了几个叫做段寄存器的东西，指向这些块的开头，这样，通过段地址+段内偏移地址的方式，就能访问更多的内存了。

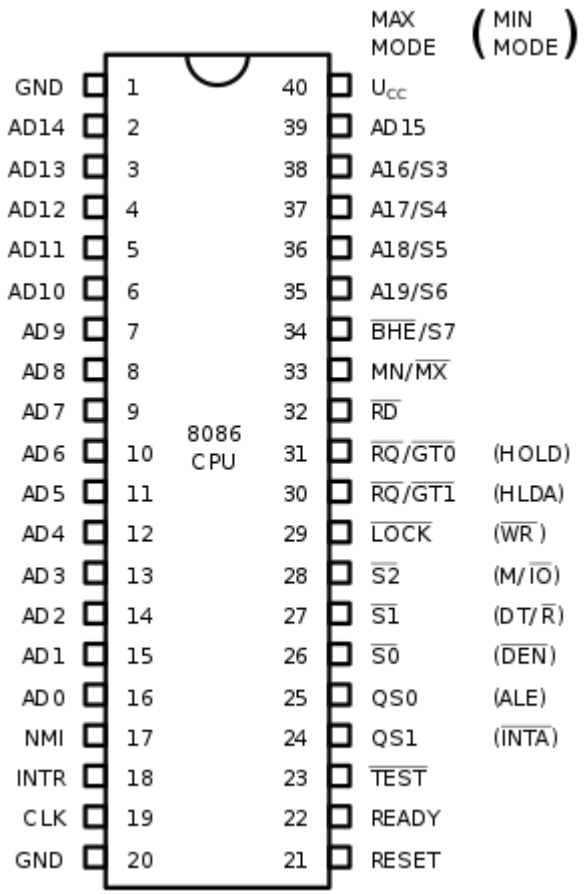

8086
IP
CS
DS
SS
ES
AX
BX
CX
DX
SP
BP
SI
DI
FLAG
CS: 0x2020
0x00000
代码段
数据段
堆栈段

32位时代

后来啊，祖先的那点计算能力越来越捉襟见肘，实在是跟不上时代了。家族中的年轻一代开始挑大梁，80286和80386CPU相继问世，尤其是80386，成为了划时代的存在。



到了80386时代，我们与外界通信的引脚就更多了，并且变成了32位的CPU，那个时候，生活条件就变好了，地址线和数据线再也不用共享引脚了。



后来，人类变得越来越贪心，想要一边听音乐，一边还要上网，同时还要编辑文档，这就同时需要运行多个程序。

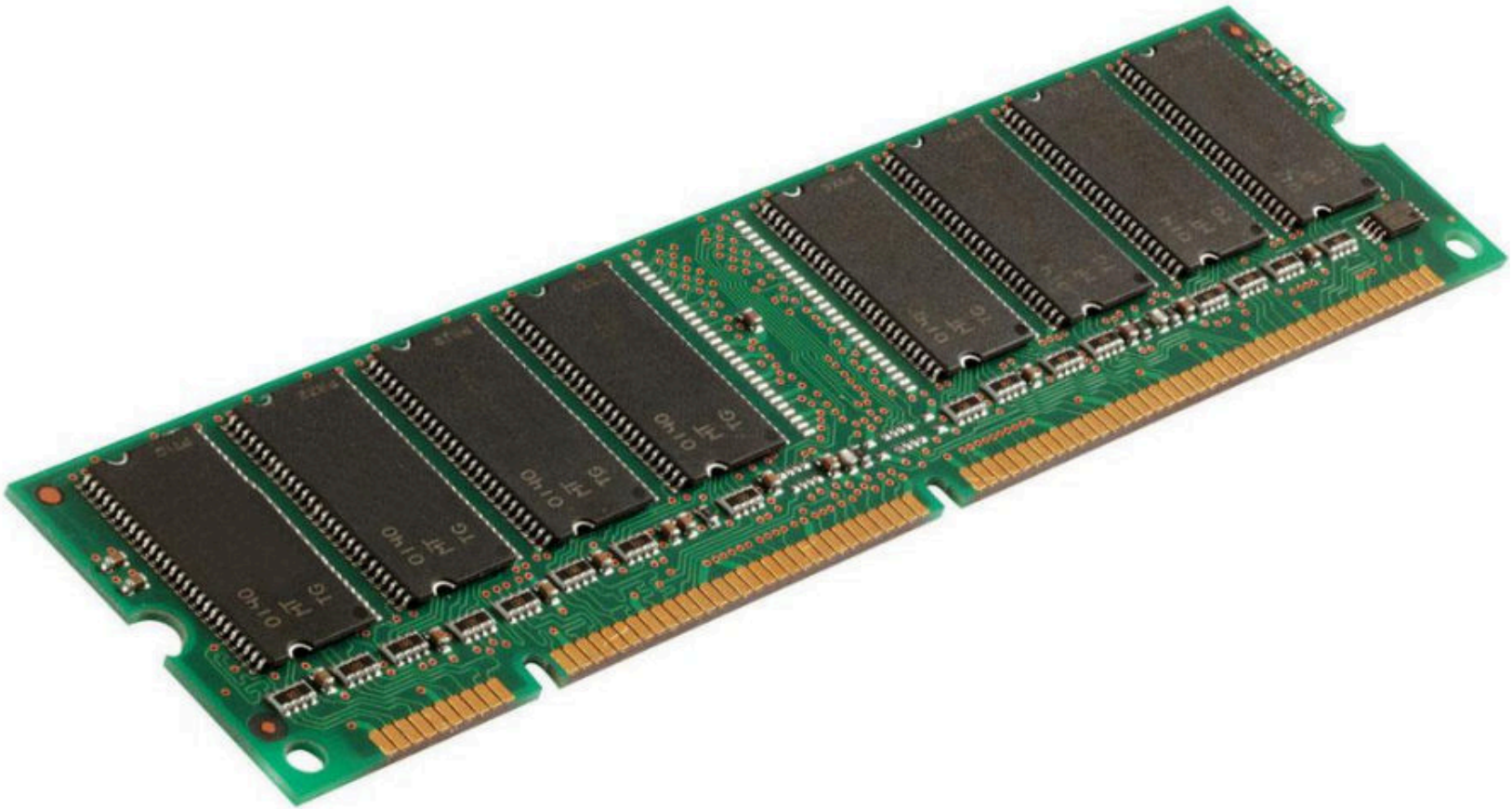
这个时候，有人发现了商机，开发了一个叫 操作系统 的东西，原来那些程序不再直接和我们CPU打交道了，而是和操作系统打交道，操作系统再和我们打交道，中间商赚差价说的就是他们！

操作系统这玩意儿很聪明啊，通过时间片划分让我们CPU来轮流执行多个程序，一会儿让我们执行音乐播放，一会儿让我们执行浏览器程序，一会儿又让我们执行文档编辑程序。我们是无所谓啊，给什么代码不是代码

啊，我们不挑，埋头苦干就是了。人类的反应速度跟我们就差得远了，他们还以为这些程序真的是同时执行的呢。

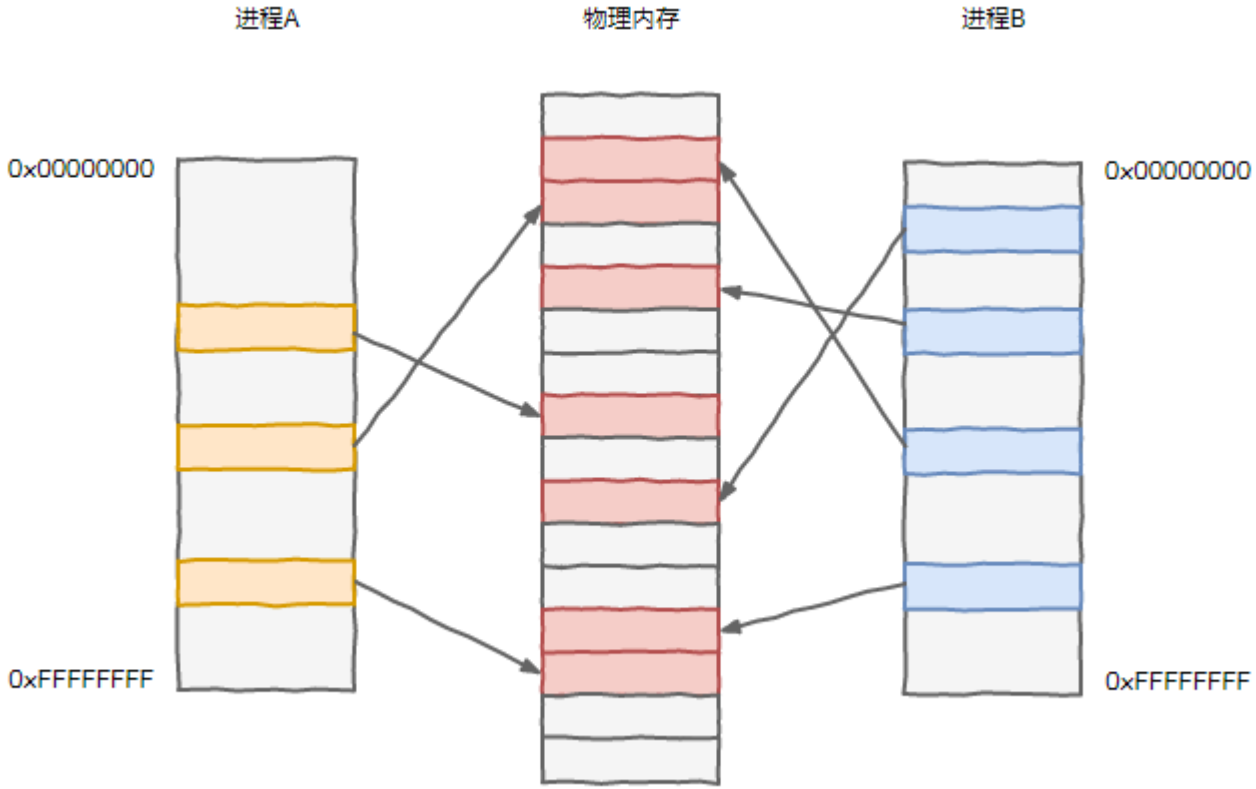
虚拟内存

不过随之而来出现了一个大问题，这么多程序都要运行，大家挤在一个内存里，经常发生摩擦，冲突不断。

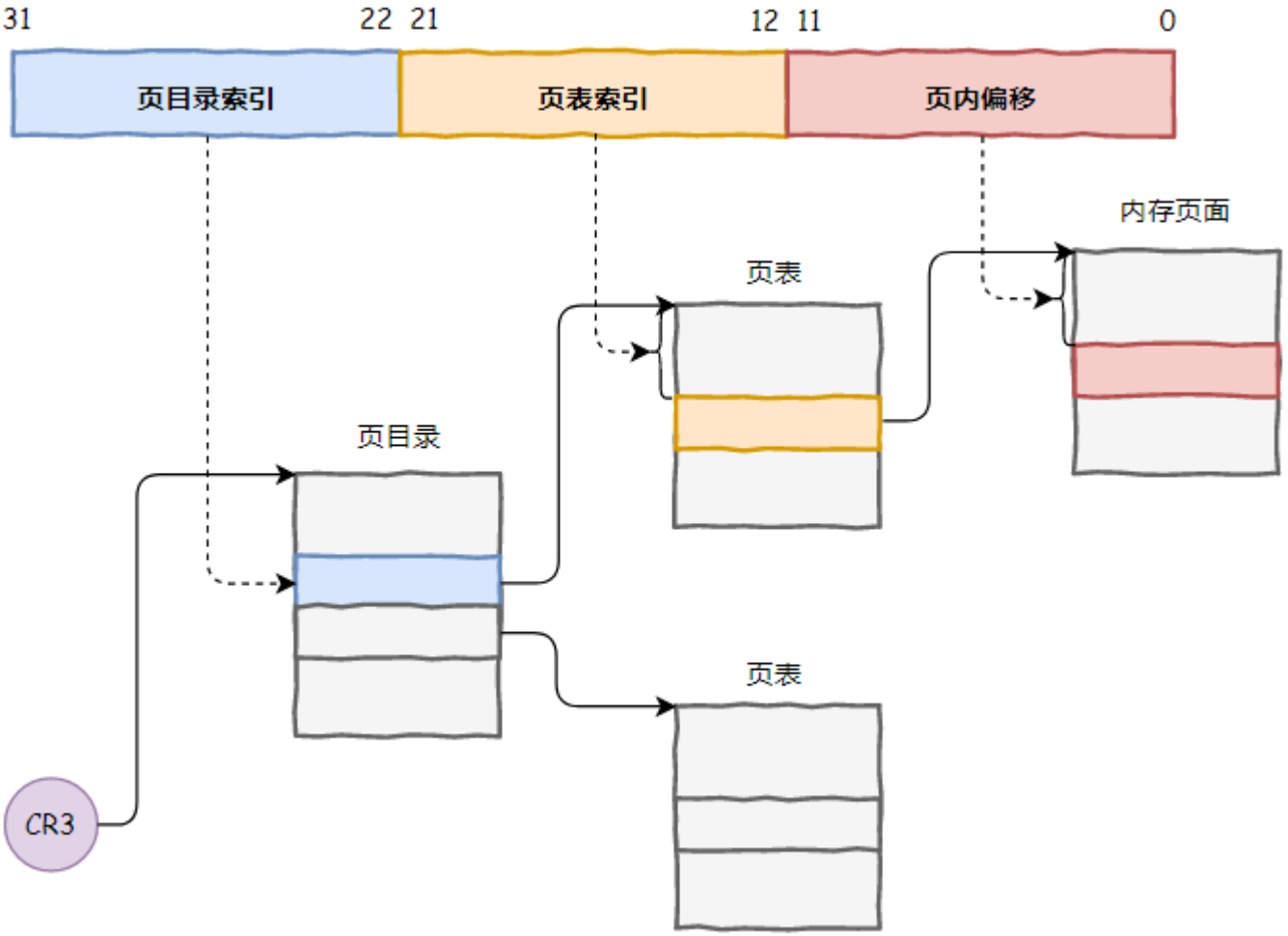


先祖们为了此事殚精竭虑，终于想出了一个好办法，一直沿用至今。

他们提出了一个 虚拟地址 的东西，所有程序使用的地址都是一个虚拟的地址，在真正和内存打交道的时候，咱们CPU内部工作人员再给翻译成真实的内存地址，关于这事儿，内存那家伙一直被我们蒙在鼓里。



这样一来，每个程序都可以用的是0x00000000到0xffffffff总共4GB这么大范围的地址空间，当然不会真的给他们那么多空间，内存那家伙总共才4GB呢，而是要按需申请分配。分配的单元是按照 页 来进行的，32位的CPU一个页是4KB。这些分配管理的累活就让操作系统来干了，中间商不能光拿好处不干正事，至于我们CPU，做好地址翻译的工作就好了。



为此，在我们寄存器内部专门添置了一个新的寄存器CR3，用来指向一个地址翻译查询字典，字典划分了两级目录。我们把一个32位的地址划分了3部分，前面两部分分别指向两级目录中的条目，用来定位这个地址在物理内存的哪个页面，最后一部分就是指向物理内存页面的偏移，这样就完成了地址的翻译工作。

每个进程有不同的地址空间，切换进程的时候，把CR3的内容换一下就使用新进程的翻译字典，特别的方便。

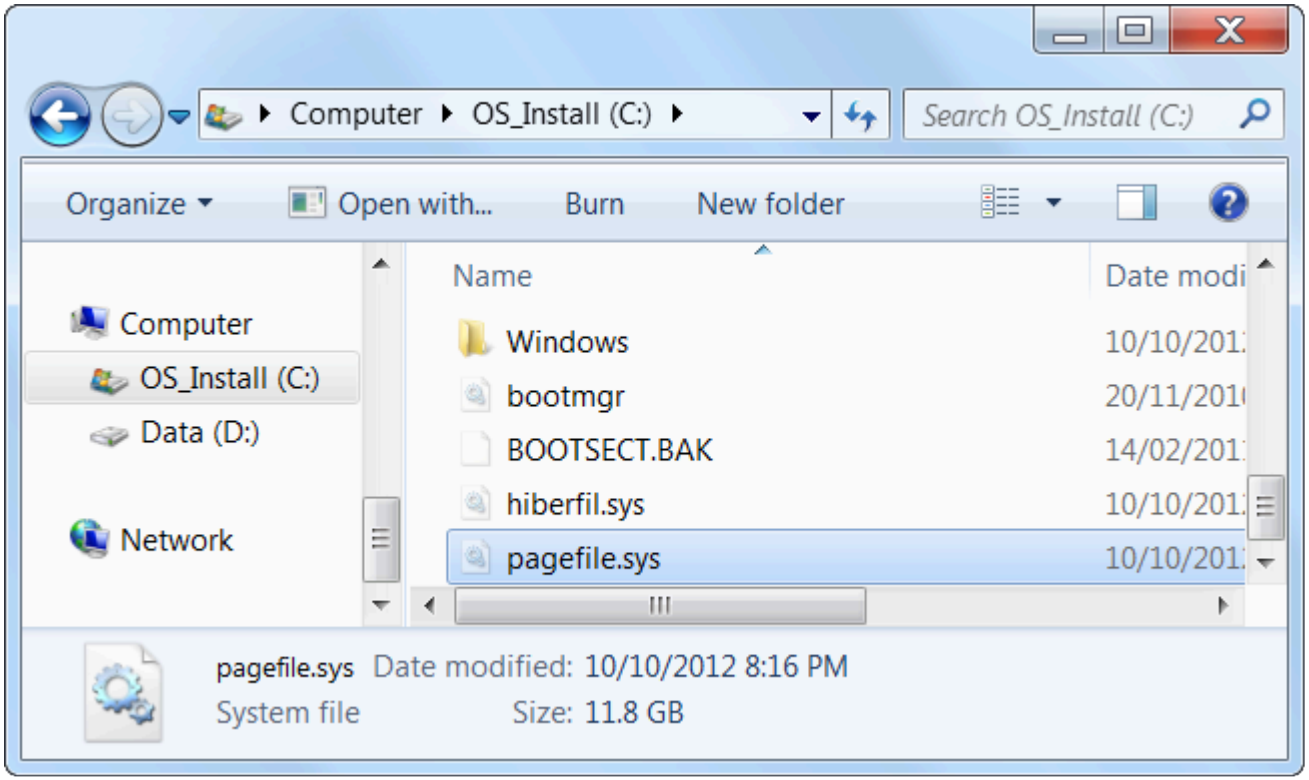
我们把这种内存管理方式叫做 分页式内存管理 。

真佩服先祖们的智慧，这样巧妙的把各个程序隔离开来，后来我们把这种工作模式叫做 保护模式 ，把之前那种直接使用真实内存地址的工作模式叫做 实地址模式 。

分页交换

人类变得越来越贪婪，程序变得越来越多，对内存的需求也越来越大。随着这些程序都不断申请内存页面，内存空间很快就要耗尽了。

我们看在眼里，急在心里，后来找操作系统协商，看看这问题该怎么办。



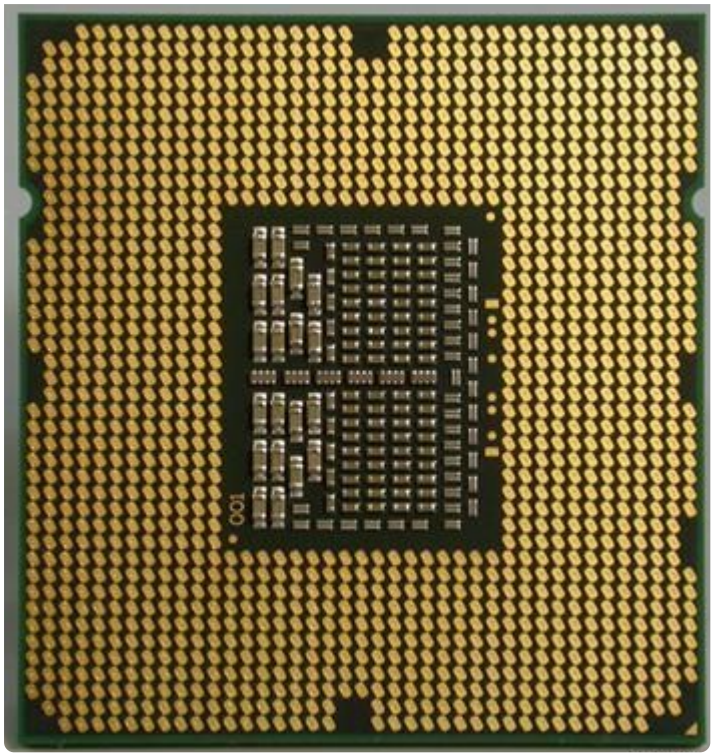
操作系统那家伙也不赖，想出了一个好办法。内存的大小有限，但是硬盘给力啊，硬盘空间大的多，去硬盘上划一块区域来，把内存里长时间没有用到的页面给换到这块区域里去，然后做个标记。如果后面谁要访问那个页面，咱们CPU就检查如果有这个标记，就发送一个页错误的中断信号告诉操作系统去把这个页面换回来。

通过我们之间的配合，解决了内存紧张的危机。后来我们把这个技术叫做 内存分页交换 。

现在

时间过得很快，到了我们这一辈，内存变得更大了，16GB都是小case，32GB也很常见。

除了内存，我们CPU本身也更先进了，别的不说，你光看看咱们现在的引脚数那比先祖们那几辈就不可同日而语。



我们不仅从32位变成了64位，还从单核变成了多核，像我所在的CPU就有8个车间，8核并行执行，比起先祖那个年代简直有云泥之别。