第三讲 IA32 内存寻址机制

在硬件工程师和普通用户看来，内存就是插在或固化在主板上的内存条，它们有一定的容量，比如128MB。但在应用程序员看来中，并不过度关心插在主板上的内存容量，而是他们可以使用的内存空间，比如，他们可以开发一个占用1 GB 内存的程序，并让其在操作系统下运行，哪怕这台机器上只有128 MB 的物理内存条。而对于操作系统开发者而言，则是介于二者之间，它既需要知道物理内存的地址，也需要提供一套机制，为应用程序员提供另一个内存空间，这个内存空间的大小可以和实际的物理内存大小之间没有多大关系。

我们将主板上的物理内存条所提供的内存空间定义为**物理内存空间，**其中每个内存单元的实际地址就是**物理地址**；将应用程序员看到的内存空间（因为高级语言不涉及内存空间，因此，这里指的是从汇编语言的角度看）定义为**虚拟地址空间(或地址空间)**，其中的地址就叫**虚拟地址(或虚地址)**， 一般用“**段：偏移量**”的形式来描述，比如在8086 中 A815:CF2D 就代表段首地址为 A815，段内偏移位为 CF2D 的虚地址。

在任何一台计算机上，都存在一个程序能产生的内存地址的集合。当程序执行这样一条指令时：

```
MOVE  REG, ADDR
```

它把地址为ADDR（假设为10000）的内存单元的内容复制到REG中，地址ADRR可以通过索引、基址寄存器、段寄存器和其它方式产生。

在 8086 的实模式下，把某一段寄存器左移 4 位，然后与地址 ADDR 相加后被直接送到内存总线上，这个相加后的地址就是内存单元的**物理地址**，而程序中的这个地址就叫**逻**

辑地址（或叫虚地址）。在 IA32 的保护模式下，这个逻辑地址不是被直接送到内存总线而是被送到内存管理单元（MMU）。MMU 由一个或一组芯片组成，其功能是把逻辑地址映射为物理地址，即进行地址转换，如图 2.7 所示。

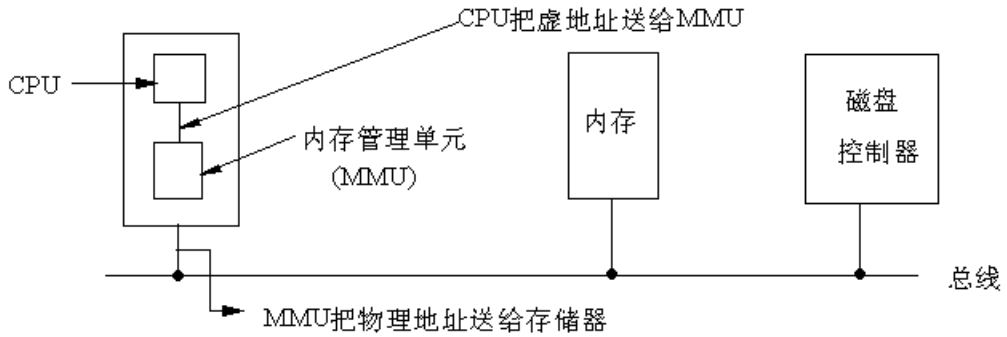

图 2－7　MMU的位置和功能

当使用IA32时，我们必须区分以下三种不同的地址：

逻辑地址：

机器语言指令仍用这种地址指定一个操作数的地址或一条指令的地址。这种寻址方式在Intel的分段结构中表现得尤为具体，它使得MS-DOS或Windows程序员把程序分为若干段。每个逻辑地址都由一个段和偏移量组成。

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2^{32}（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：

也就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。

从图2.7可以看出，MMU是一种硬件电路，它包含两个部件，一个是分段部件，一个是分页部件，在此，我们把它们分别叫做分段机制和分页机制，以利于从逻辑的角度来理解硬件的实现机制。分段机制把一个逻辑地址转换为线性地址；接着，分页机制把一个线性地址转换为物理地址，如图2.8所示。

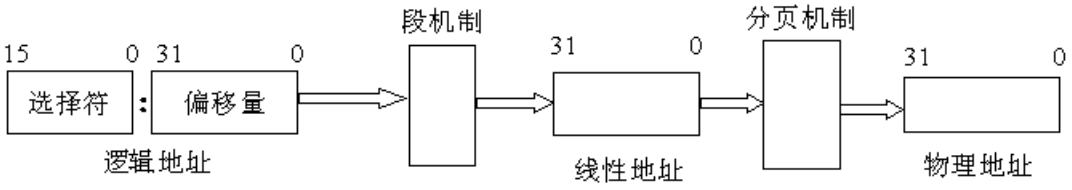

图2.8 MMU把逻辑地址转换为物理地址