汇编语言的准备知识-给初次接触汇编者2 PDF转换可能丢失图片或格式，建议阅读原文
https://www.100test.com/kao_ti2020/138/2021_2022__E6_B1_87_E7_BC_96_E8_AF_AD_E8_c98_138270.htm 汇编指令的操作数可以是内存中的数据， 如何让程序从内存中正确取得所需要的数据就是对内存的寻址。 INTEL 的CPU 可以工作在两种寻址模式:实模式和保护模式。 前者已经过时，就不讲了，WINDOWS 现在是32位保护模式的系统， PE 文件就基本是运行在一个32位线性地址空间， 所以这里就只介绍32位线性空间的寻址方式。 其实线性地址的概念是很直观的， 就想象一系列字节排成一长队，第一个字节编号为0， 第二个编号位1，。。。。。 一直到4294967295(十六进制FFFFFFFF，这是32位二进制数所能表达的最大值了)。 这已经有4GB的容量!足够容纳一个程序所有的代码和数据。 当然， 这并不表示你的机器有那么多内存。 物理内存的 管理 和分配是很复杂的内容， 初学者不必在意， 总之， 从程序本身的角度看， 就好象是在那么大的内存中。 在INTEL系统中， 内存地址总是由"段选择符:有效地址"的方式给出。段选择符(SELECTOR)存放在某一个段寄存器中， 有效地址则可由不同的方式给出。 段选择符通过检索段描述符确定段的起始地址， 长度(又称段限制)， 粒度， 存取权限， 访问性质等。 先不用深究这些， 只要知道段选择符可以确定段的性质就行了。 一旦由选择符确定了段， 有效地址相对于段的基地址开始算。 比如由选择符1A7选择的数据段， 其基地址是400000， 把1A7 装入DS中， 就确定使用该数据段。 DS:0 就指向线性地址400000。 DS:1F5278 就指向线性地址5E5278。 我们在一般情况下， 看

不到也不需要看到段的起始地址， 只需要关心在该段中的有效地址就行了。 在32位系统中， 有效地址也是由32位数字表示， 就是说， 只要有一个段就足以涵盖4GB线性地址空间，为什么还要有不同的段选择符呢? 正如前面所说的， 这是为了对数据进行不同性质的访问。 非法的访问将产生异常中断， 而这正是保护模式的核心内容， 是构造优先级和多任务系统的基础。 这里有涉及到很多深层的东西， 初学者先可不必理会。 有效地址的计算方式是: 基址 间址*比例因子 偏移量。 这些量都是指段内的相对于段起始地址的量度， 和段的起始地址没有关系。 比如， 基址=100000， 间址=400， 比例因子=4， 偏移量=20000， 则有效地址为: 100000 400*4 20000=100000 1000 20000=121000。 对应的线性地址是400000 121000=521000。 (注意， 都是十六进制数)。 基址可以放在任何32位通用寄存器中， 间址也可以放在除ESP外的任何一个通用寄存器中。 比例因子可以是1， 2， 4 或8。 偏移量是立即数。 如: [EBP EDX*8 200]就是一个有效的有效地址表达式。 当然， 多数情况下用不着这么复杂， 间址，比例因子和偏移量不一定要出现。 内存的基本单位是字节(BYTE)。 每个字节是8个二进制位， 所以每个字节能表示的最大的数是11111111， 即十进制的255。 一般来说， 用十六进制比较方便， 因为每4个二进制位刚好等于1个十六进制位，11111111b = 0xFF。 内存中的字节是连续存放的， 两个字节构成一个字(WORD)， 两个字构成一个双字(DWORD)。 在INTEL架构中， 采用small endian格式， 即在内存中，高位字节在低位字节后面。 举例说明:十六进制数803E7D0C， 每两位是一个字节， 在内存中的形式是: 0C 7D 3E 80。 在32位

寄存器中则是正常形式，如在EAX就是803E7D0C。 当我们的形式地址指向这个数的时候，实际上是指向第一个字节，即0C。 我们可以指定访问长度是字节， 字或者双字。 假设DS:[EDX]指向第一个字节0C: mov AL， byte ptr DS:[EDX] .把字节0C存入AL mov AX， word ptr DS:[EDX] .把字7D0C存入AX mov EAX， dword ptr DS:[EDX] .把双字803E7D0C存入EAX 在段的属性中，有一个就是缺省访问宽度。如果缺省访问宽度为双字(在32位系统中经常如此)，那么要进行字节或字的访问，就必须用byte/word ptr显式地指明。 缺省段选择：如果指令中只有作为段内偏移的有效地址，而没有指明在哪一个段里的时候，有如下规则： 如果用ebp和esp作为基址或间址，则认为是在SS确定的段中； 其他情况，都认为是在DS确定的段中。 如果想打破这个规则，就必须使用段超越前缀。举例如下： mov eax， dword ptr [edx] .缺省使用DS，把DS:[EDX]指向的双字送入eax mov ebx， dword ptr ES:[EDX] .使用ES:段超越前缀，把ES:[EDX]指向的双字送入ebx 