1. 数据类型和容器

立即数：一个常数，也就是一个普通的数字，不能作为容器。

寄存器：CPU中的临时储存单元，80X86的32位平台平台上CPU寄存器仅有14个，寄存器速度比内存快得多，存储的可以是从0x00000000到0xFFFFFFFF之间的一个整数，一个int类型的存储大小，占32个bit，即一个DoubleWord的长度。

内存：操作系统分配给程序的虚拟内存，并不是真实内存。80X86的32位平台每个进程都有4G的虚拟内存，也可以说是4GB寻址能力，地址从低位到高位分别是0x00000000到0xFFFFFFFF，占32个bit，即一个DoubleWord的长度，总共4GB。

1. 数据储存方式：

在内存中，一个数比如0x11223344会被分个储存，每个被分成的部分是八个bit，也就是一个byte。所以0x11223344会被分成0x11,0x22,0x33,0x44四个部分，数字的低位存在内存的高位，数据的高位存在内存的低位，这种模式被称为“小端模式”。

从内存条上看，数据是这样的：

Address: 0x00000000 | 0x00000001 | 0x00000002 | 0x00000003

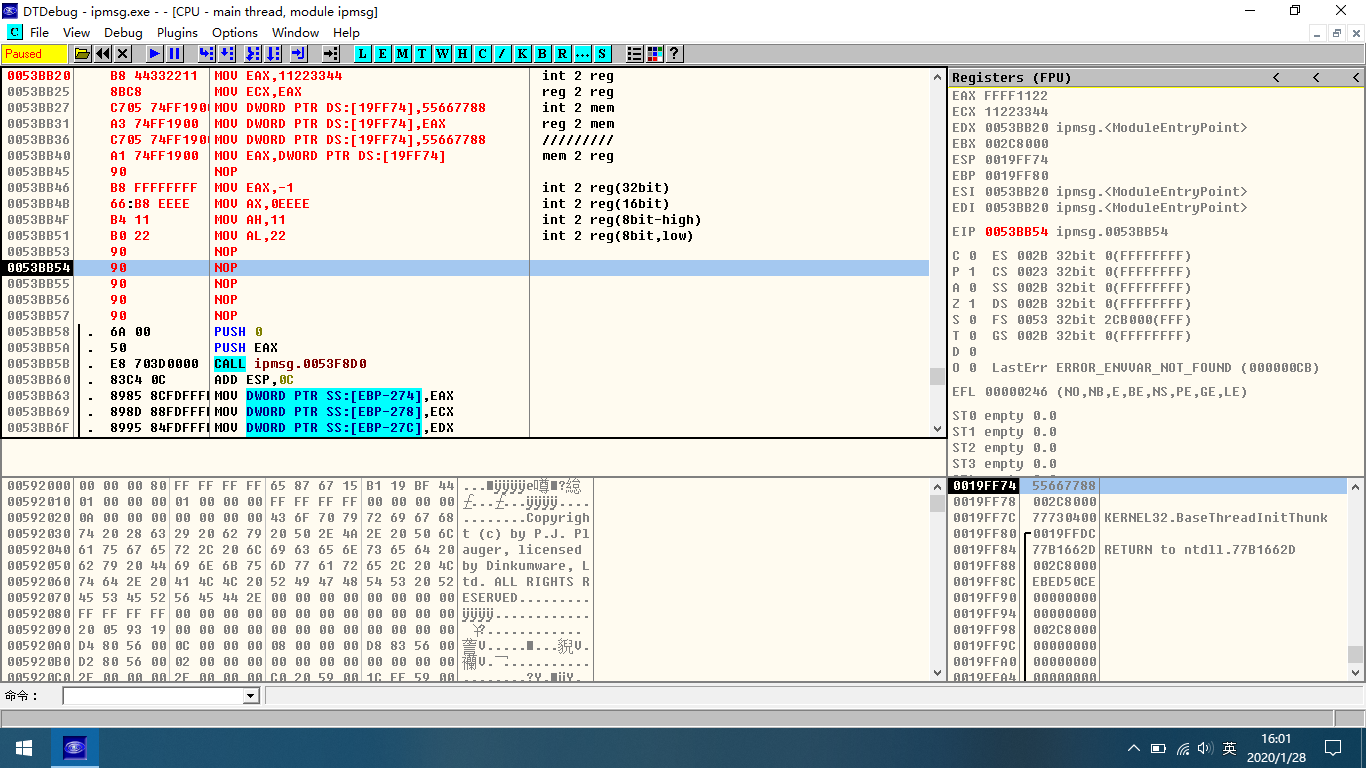
..Value: 0x44.......| 0x33.......| 0x22.......| 0x11......

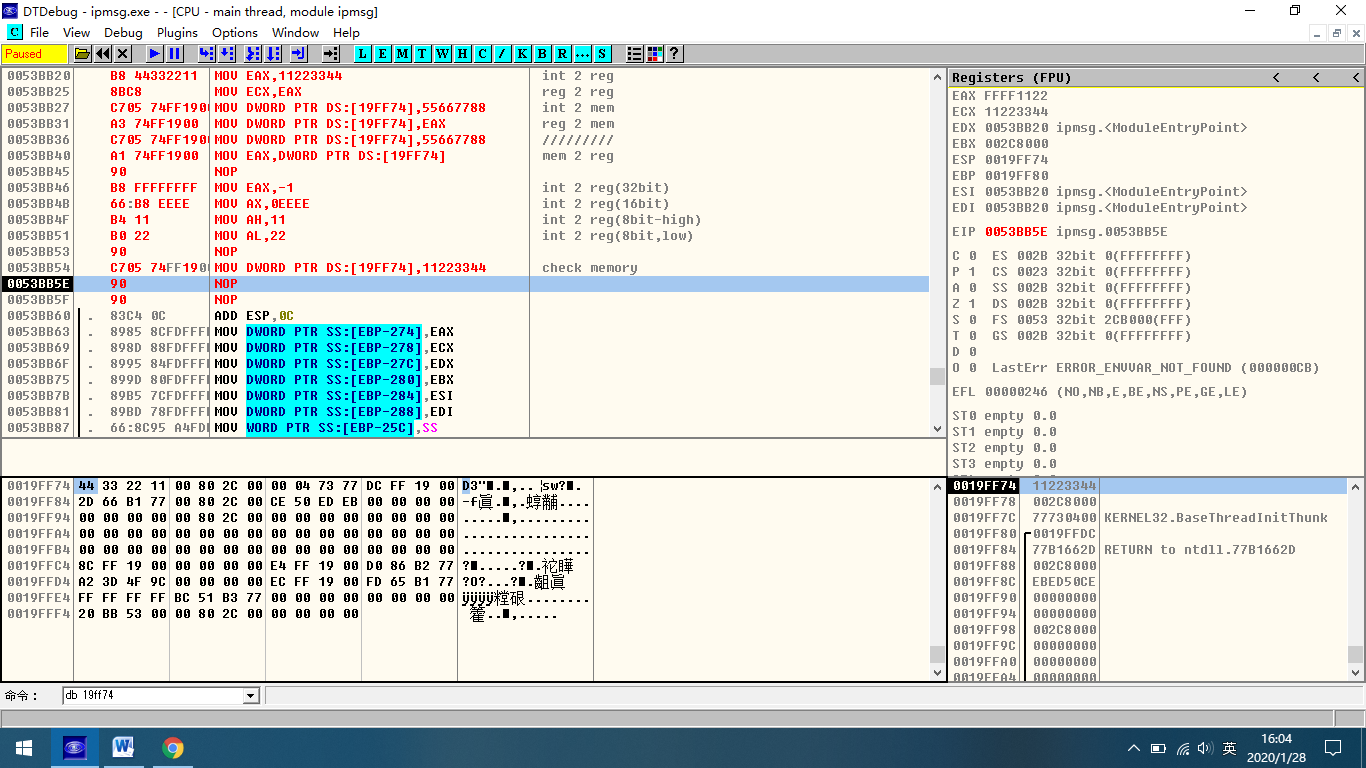
三、80X86汇编指令

*注意：只要没有特别提到，下列这些指令的参数都可以是三十二位立即数、三十二位寄存器、三十二位内存、十六位立即数、十六位寄存器、十六位内存、八位立即数、八位寄存器、八位内存。*

1. MOV a,b

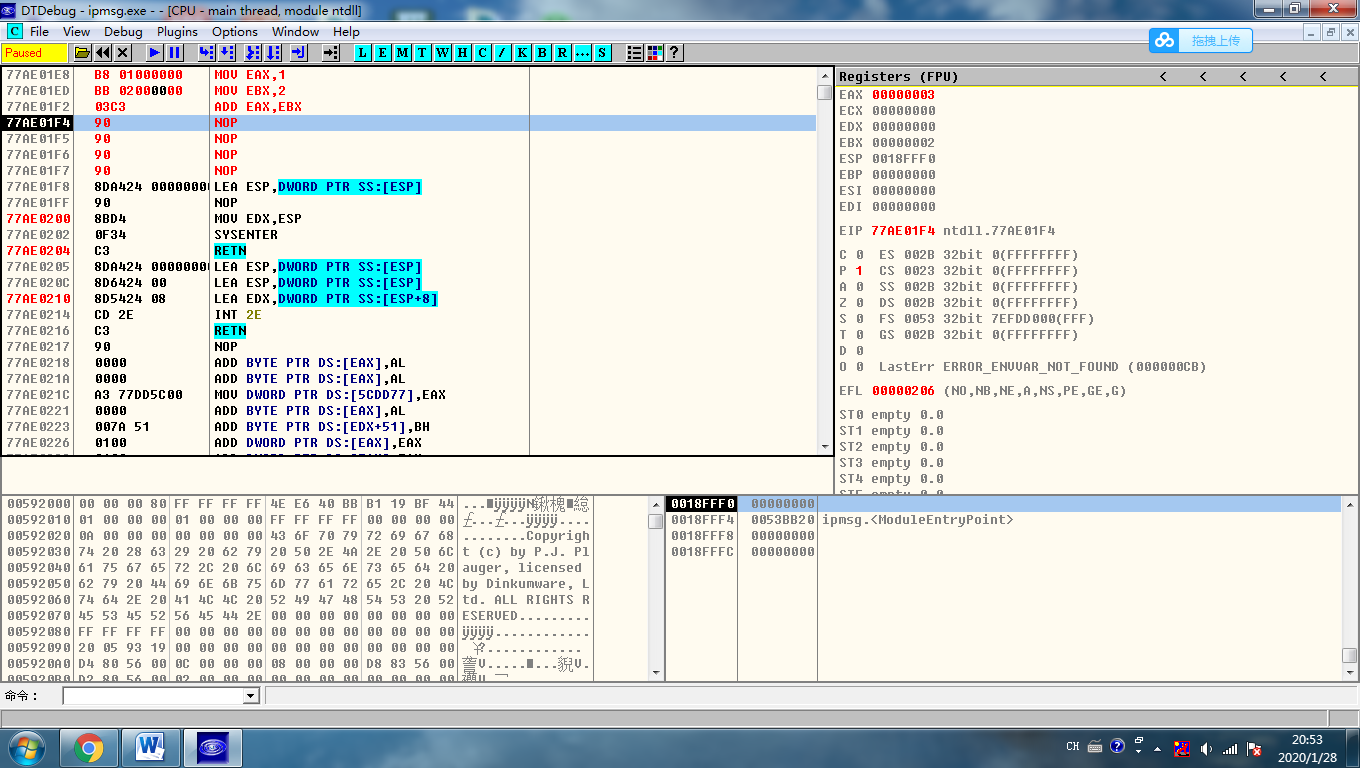
把b写入到a



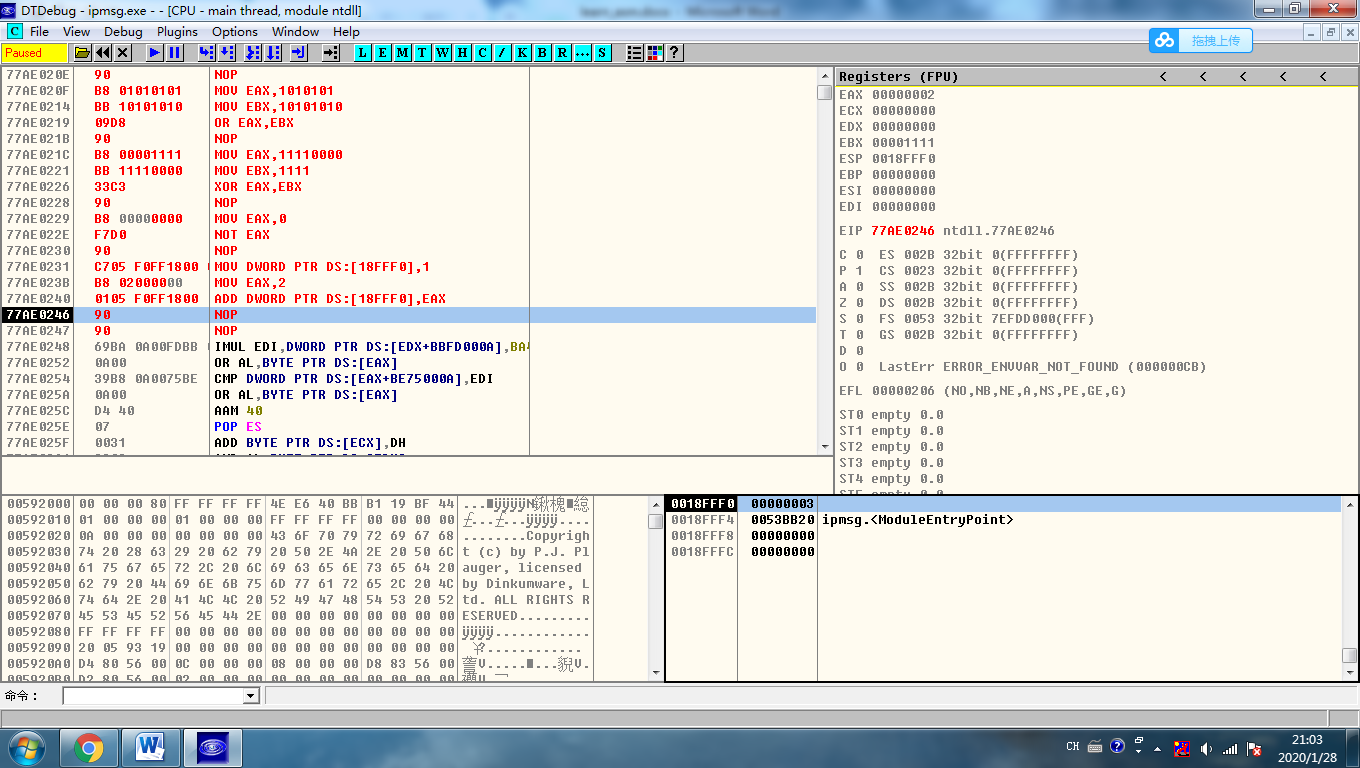


1. ADD a,b

a加b，结果存到a

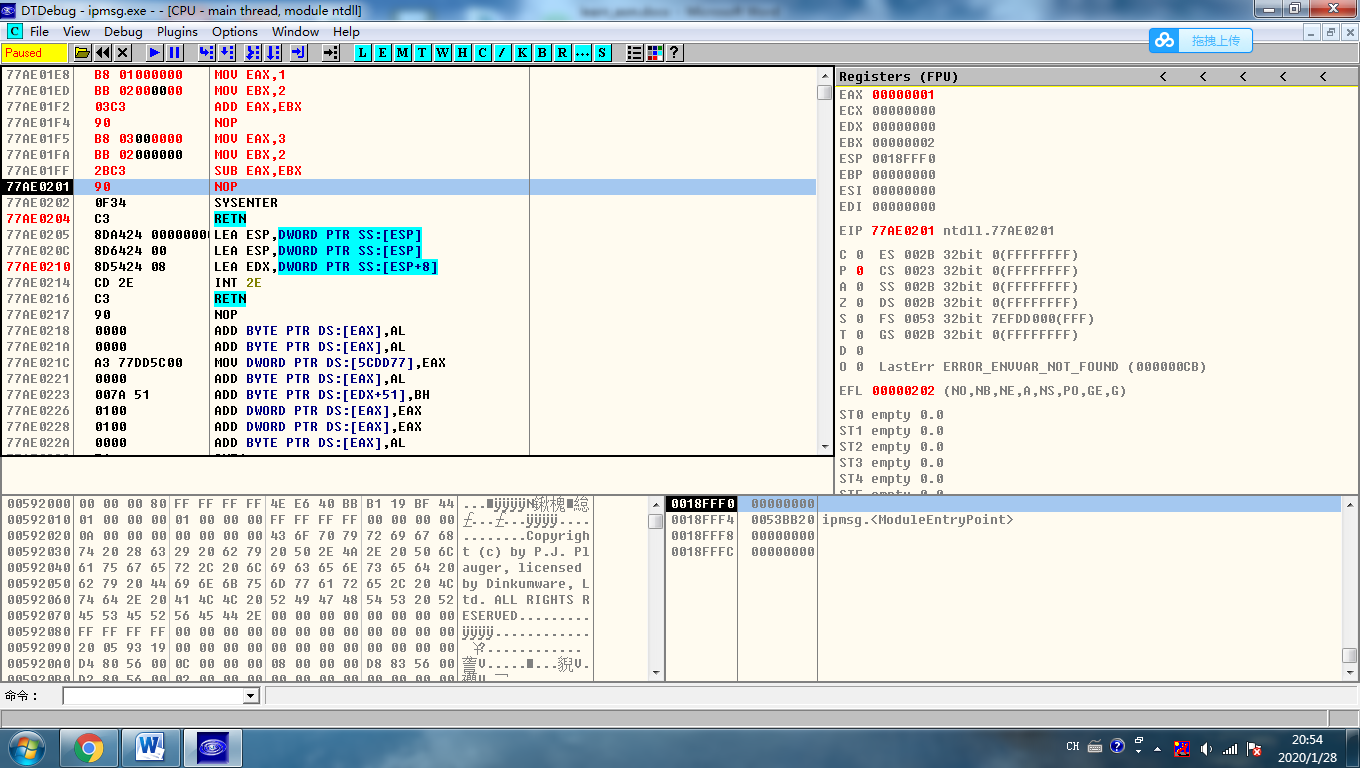


PS:使用内存的ADD。之后的指令如果没有特别说明，也可以有一个参数使用内存地址，但不能两个参数都用内存地址。



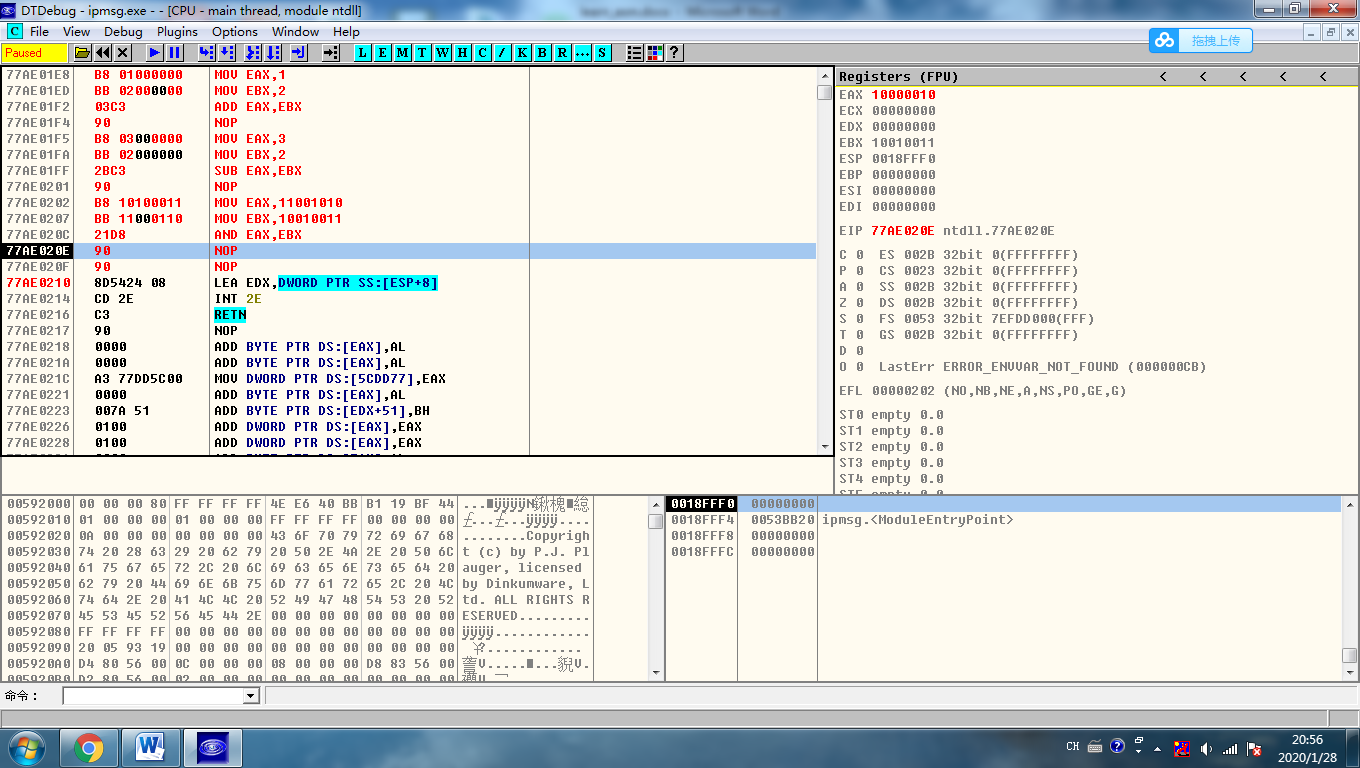
1. SUB a,b

a减b，结果存到a



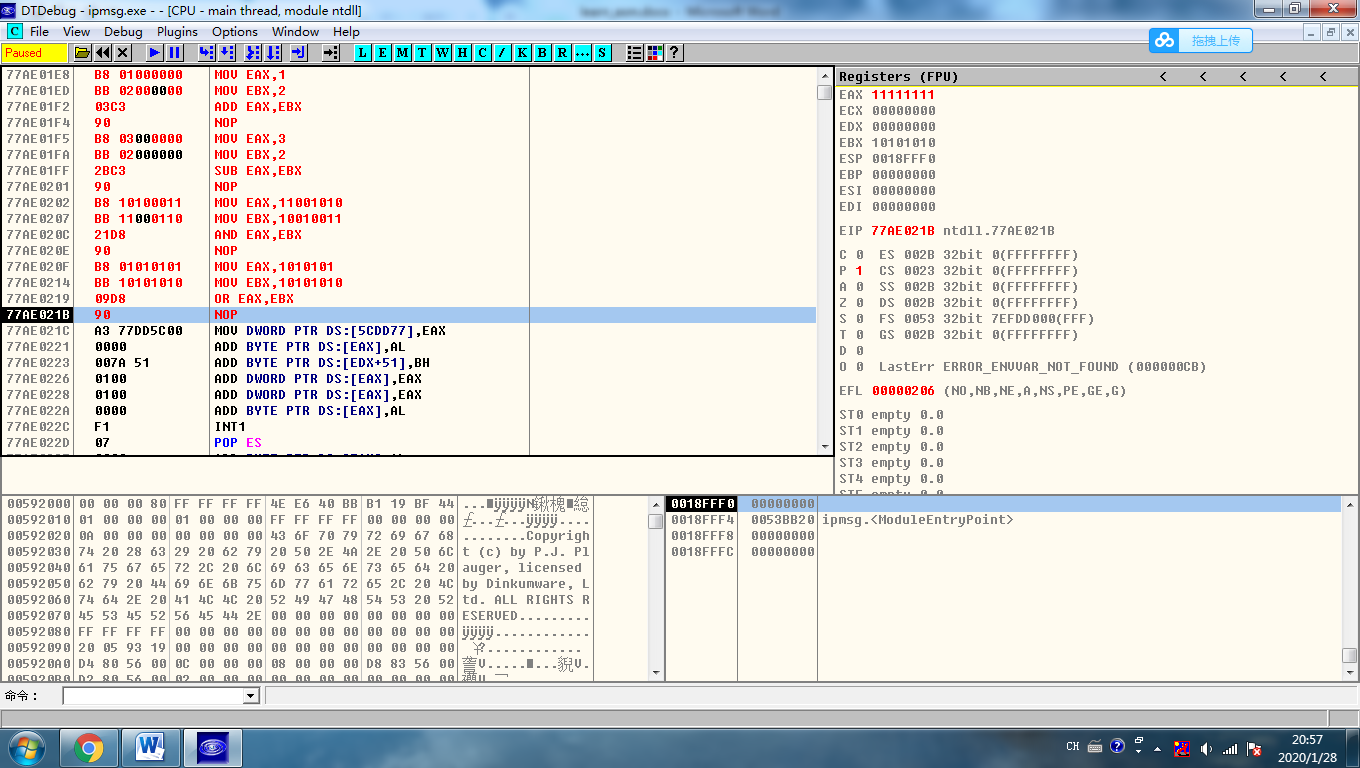
1. AND a,b

a与b，结果存到a



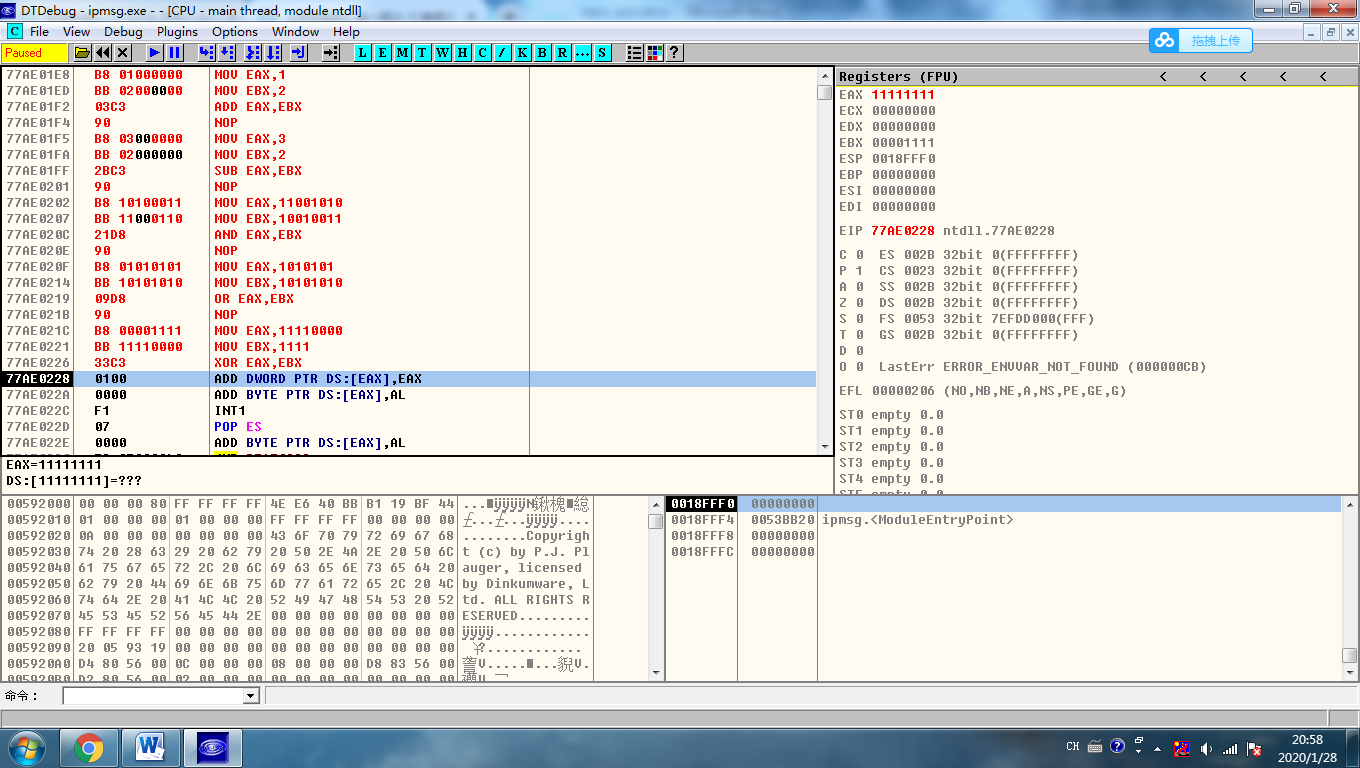
1. OR a,b

a或b，结果存到a



1. XOR a,b

a异或b，结果存到a



1. NOT a

非a，结果存到a

