《漏洞利用及渗透测试基础》实验报告

姓名：孙蕗 学号：2112060 班级：信安1班

**实验名称：**

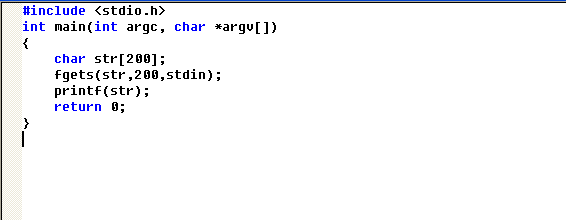
格式化字符串漏洞

**实验要求：**

以第四章示例4-7代码，完成任意地址的数据获取，观察Release模式和Debug模式的差异，并进行总结。

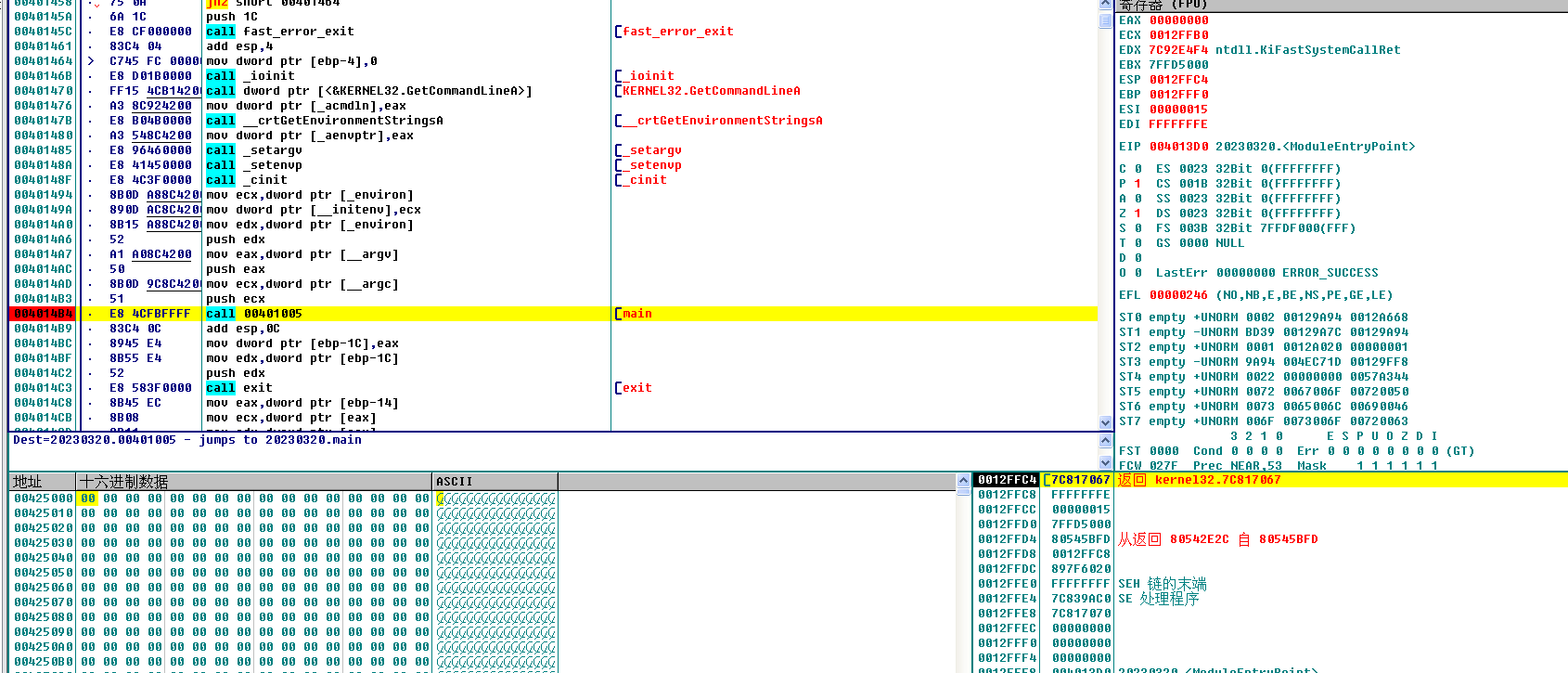
**实验过程：**

1. **debug模式**
2. 编写代码

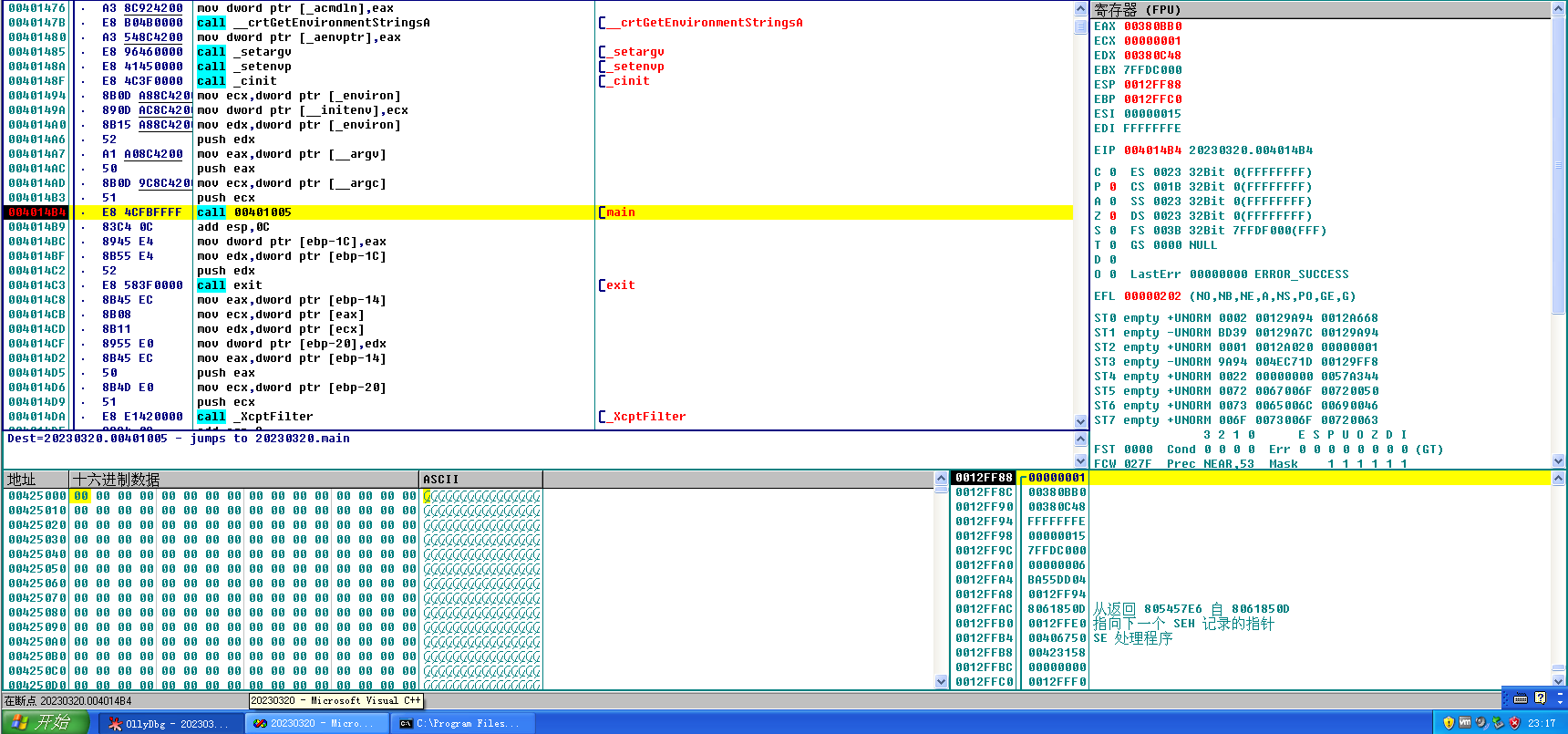


代码表示的含义是有一个200大小的字符数组，fgets的结果会存在字符数组里，printf输出。

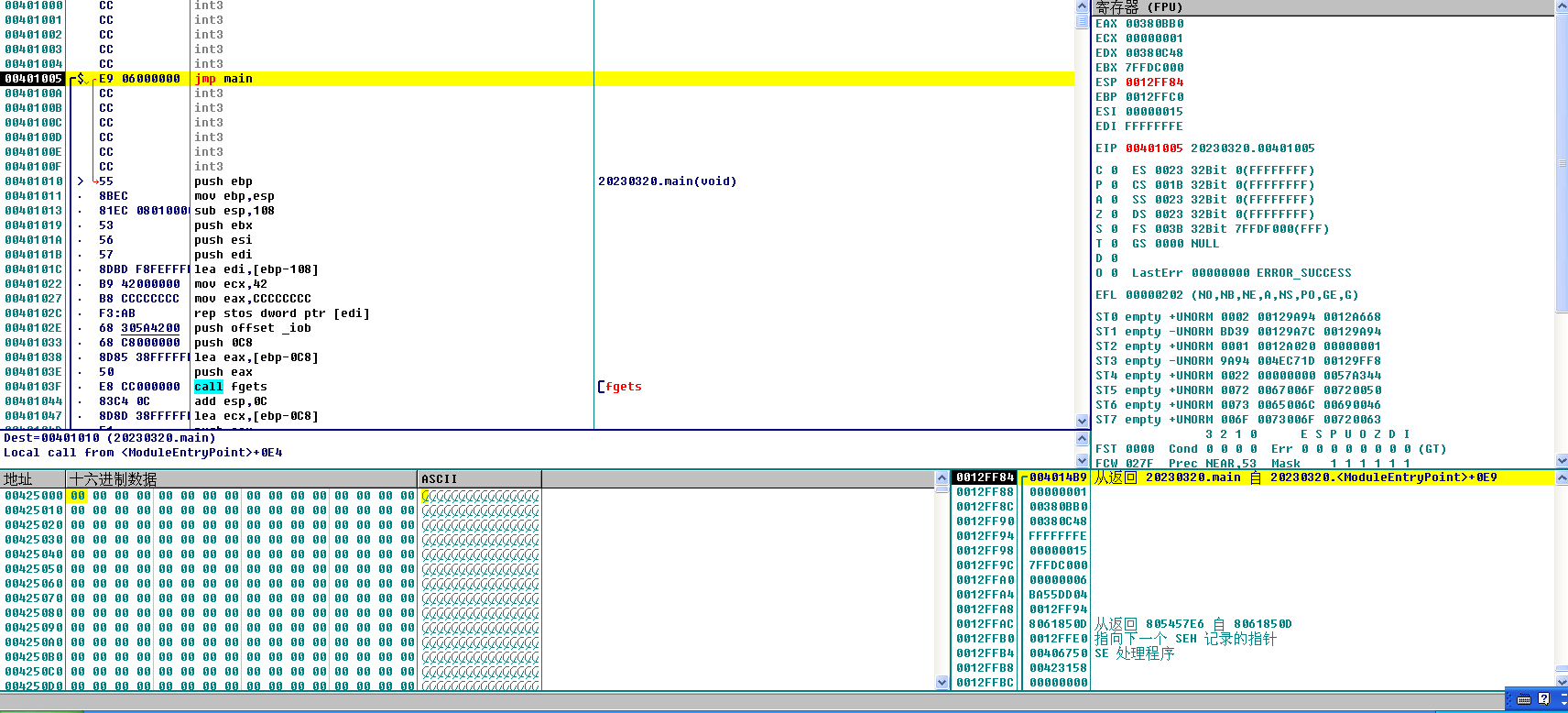
1. Ollydbg打开，找到主函数，然后进入



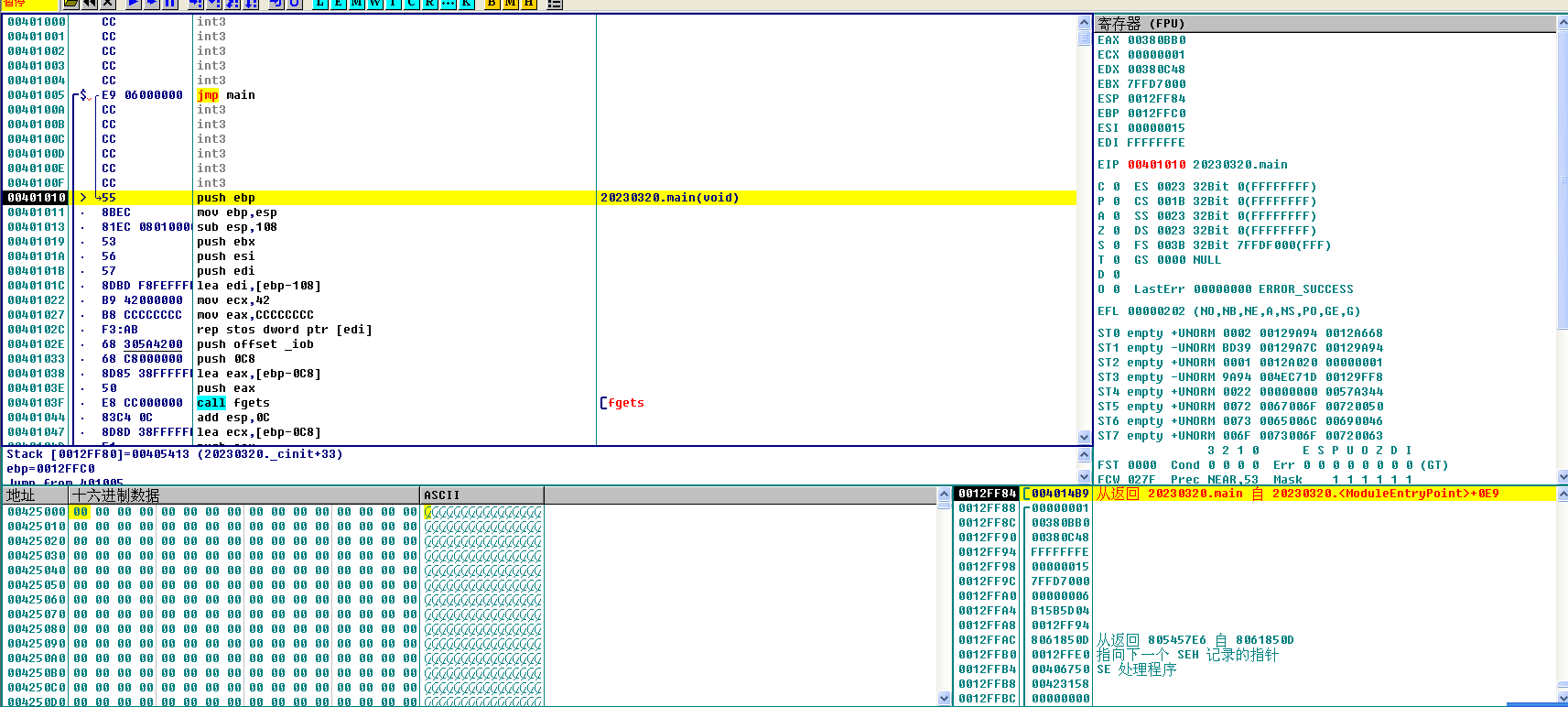
1. F9执行到下面的位置



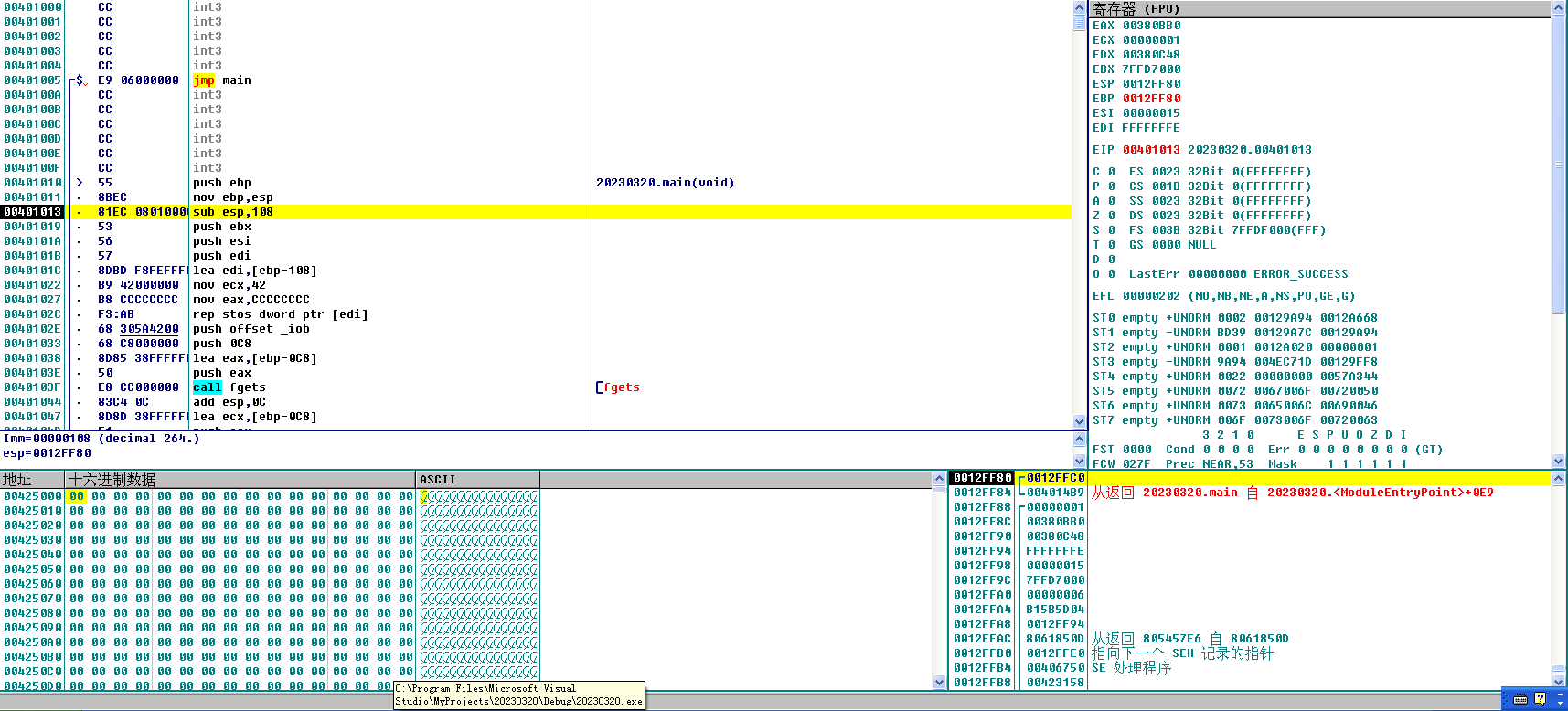
1. F7进入



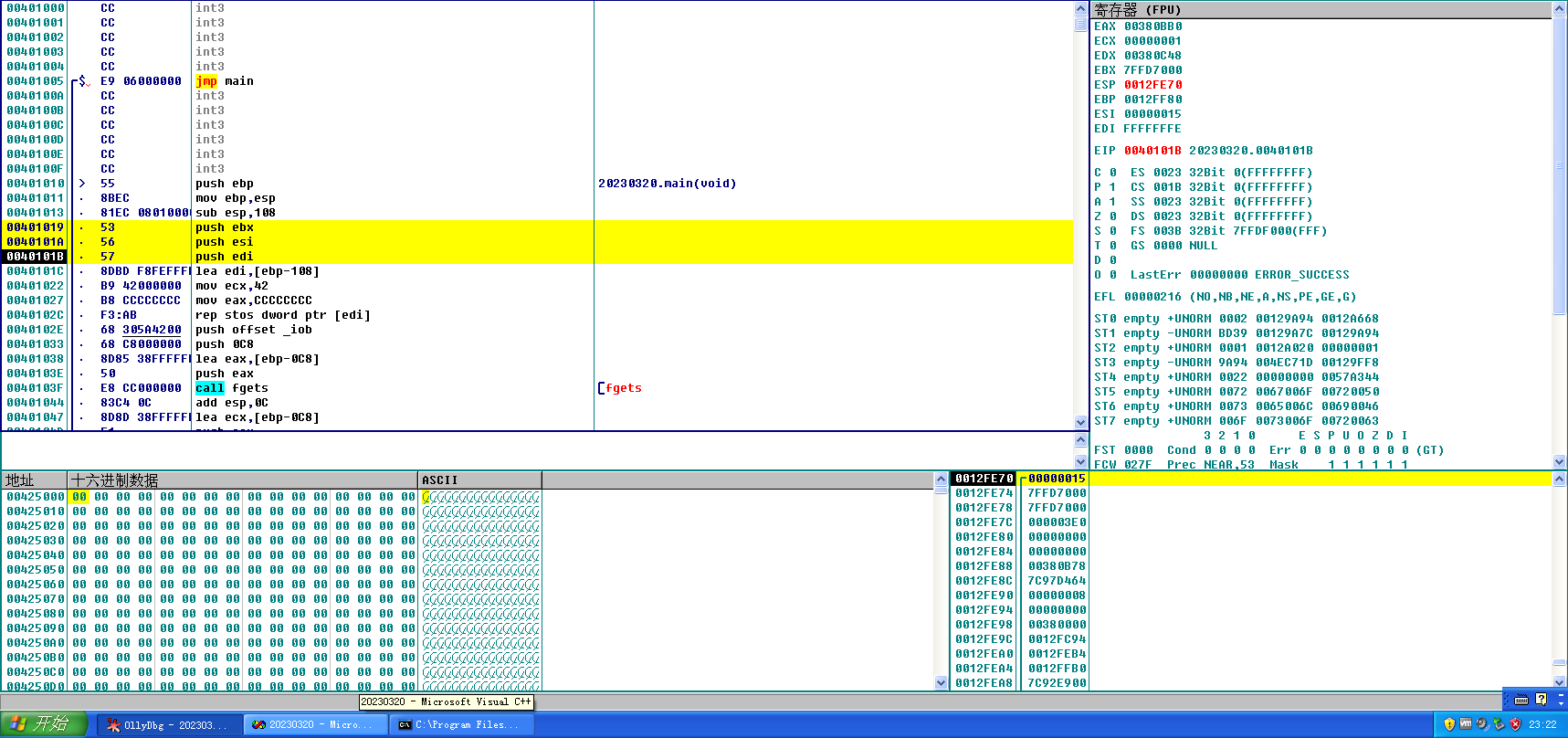
1. F7，debug模式里需要push ebp



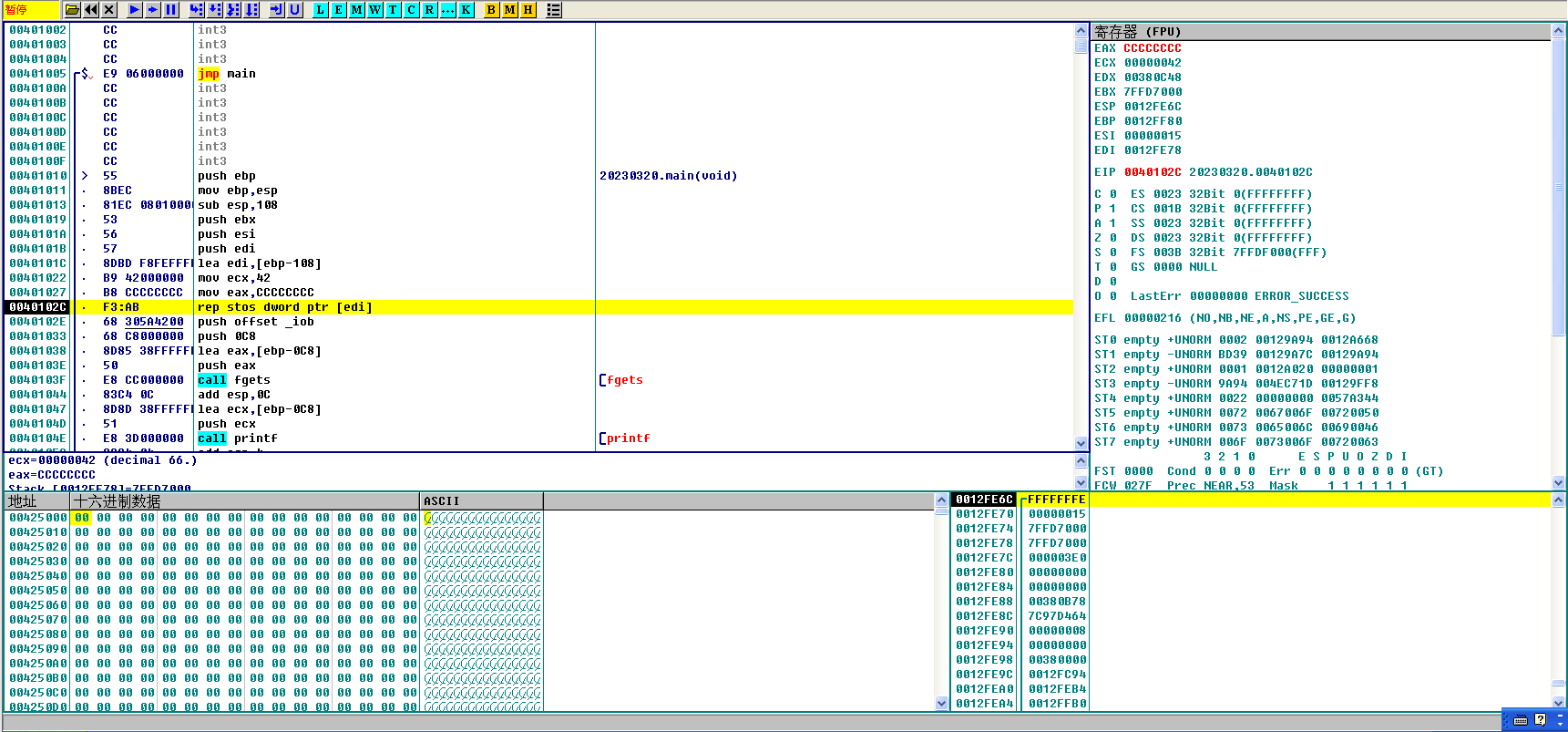
1. F7 mov ebp,esp；sub esp,108。十六进制的108远比0C8（十进制的200）大，用来抬高栈帧，给出很大的运行区域。



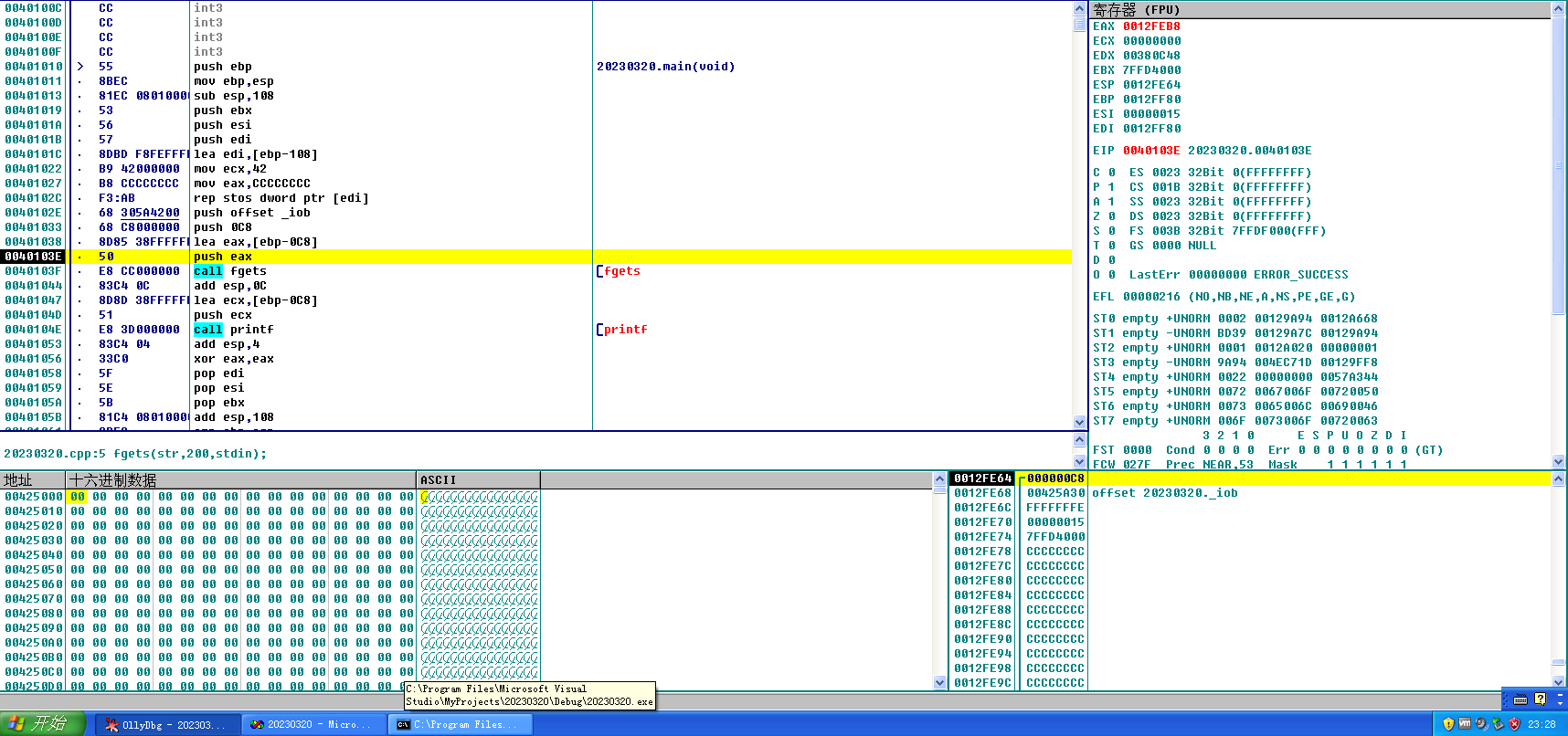
1. F7 3个push都在整个区域的顶部，整个栈帧的顶部，去保存调用函数前那些函数的寄存器的值



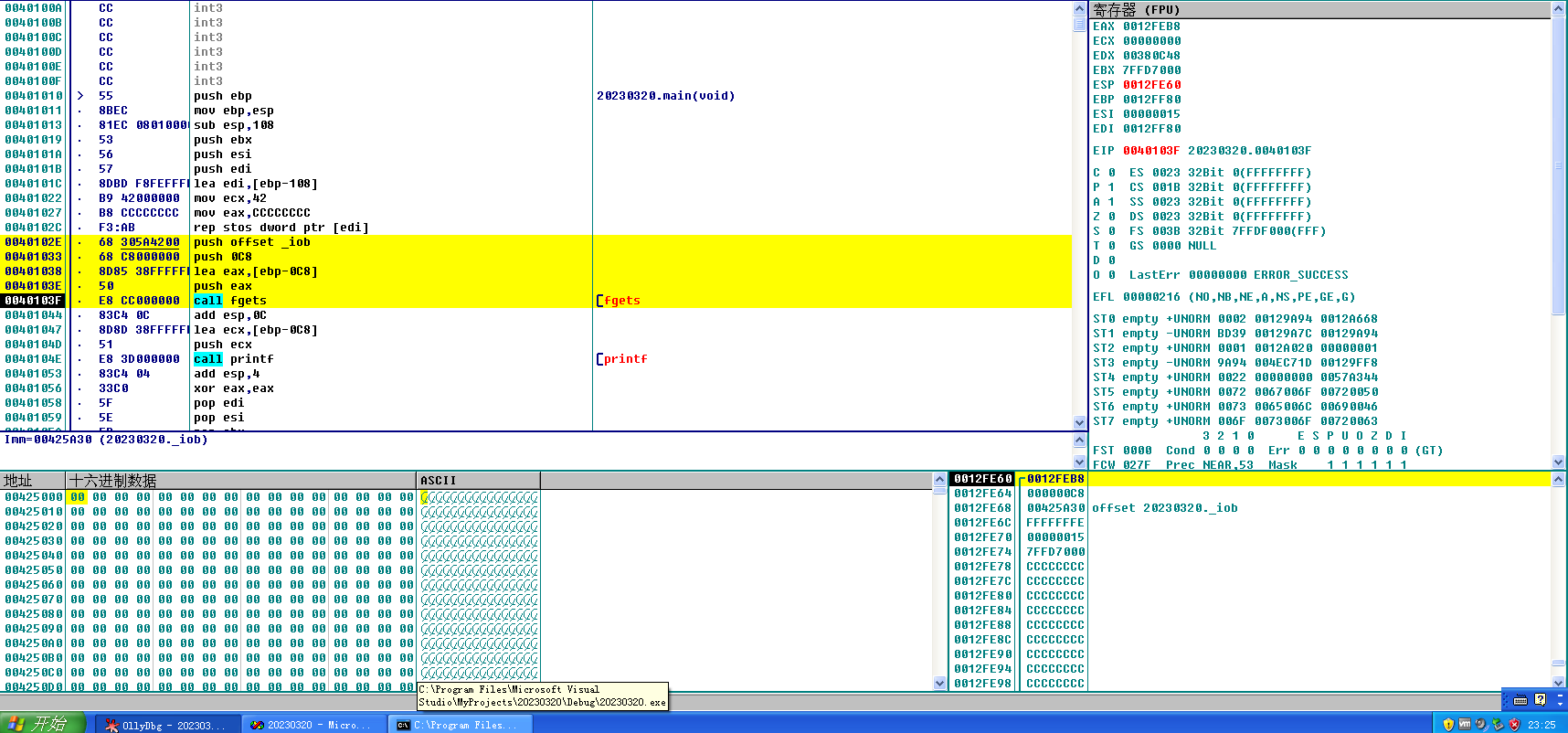
1. F7 ecx和rep两条指令，将16进制108的空间，全都初始化成CCCCCCCC



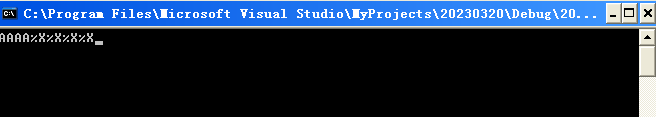
1. F7 eax=0012FEB8，就是局部变量的，存放输入的字符串的起始地址



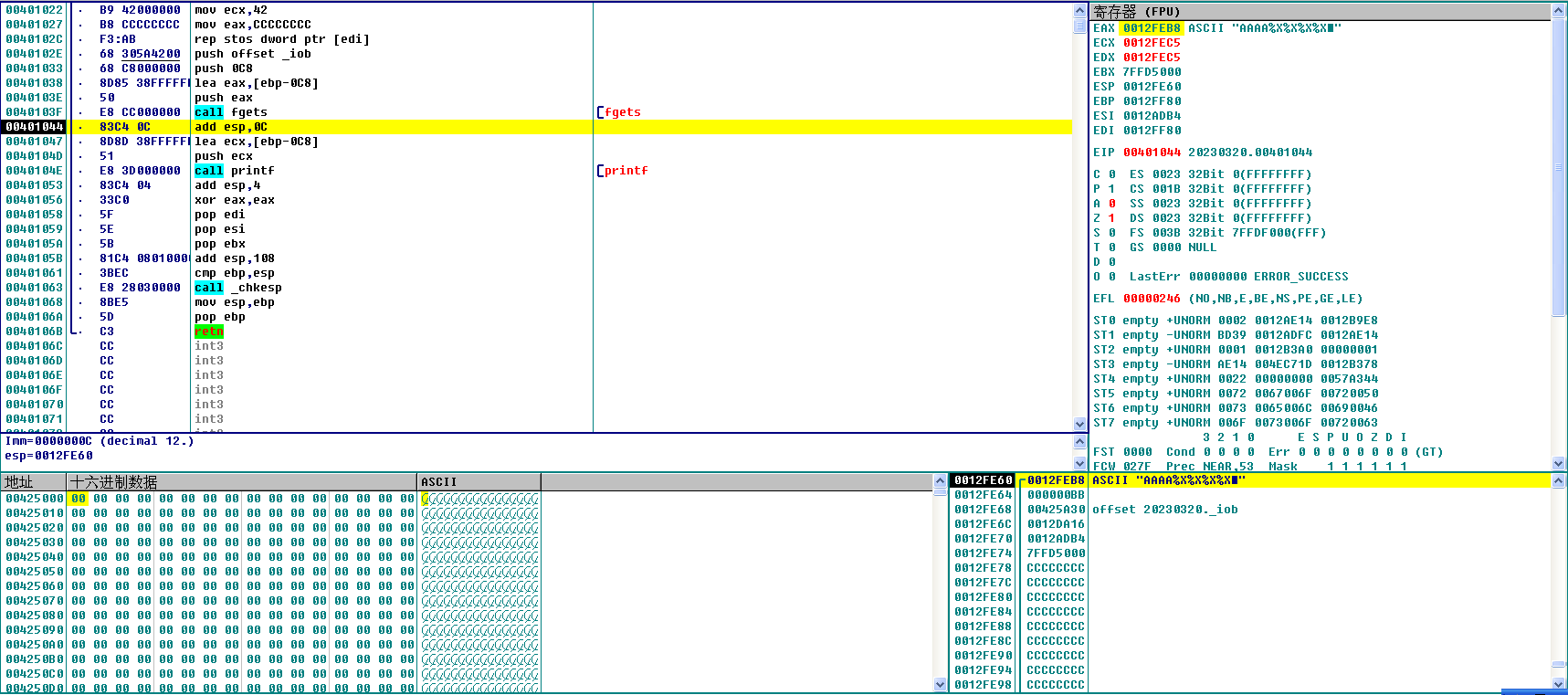
1. F8 调用fgets前3个push，lea用来取地址，[ebp-0C8]就是字符数组的地址



1. 输入AAAA%X%X%X%X

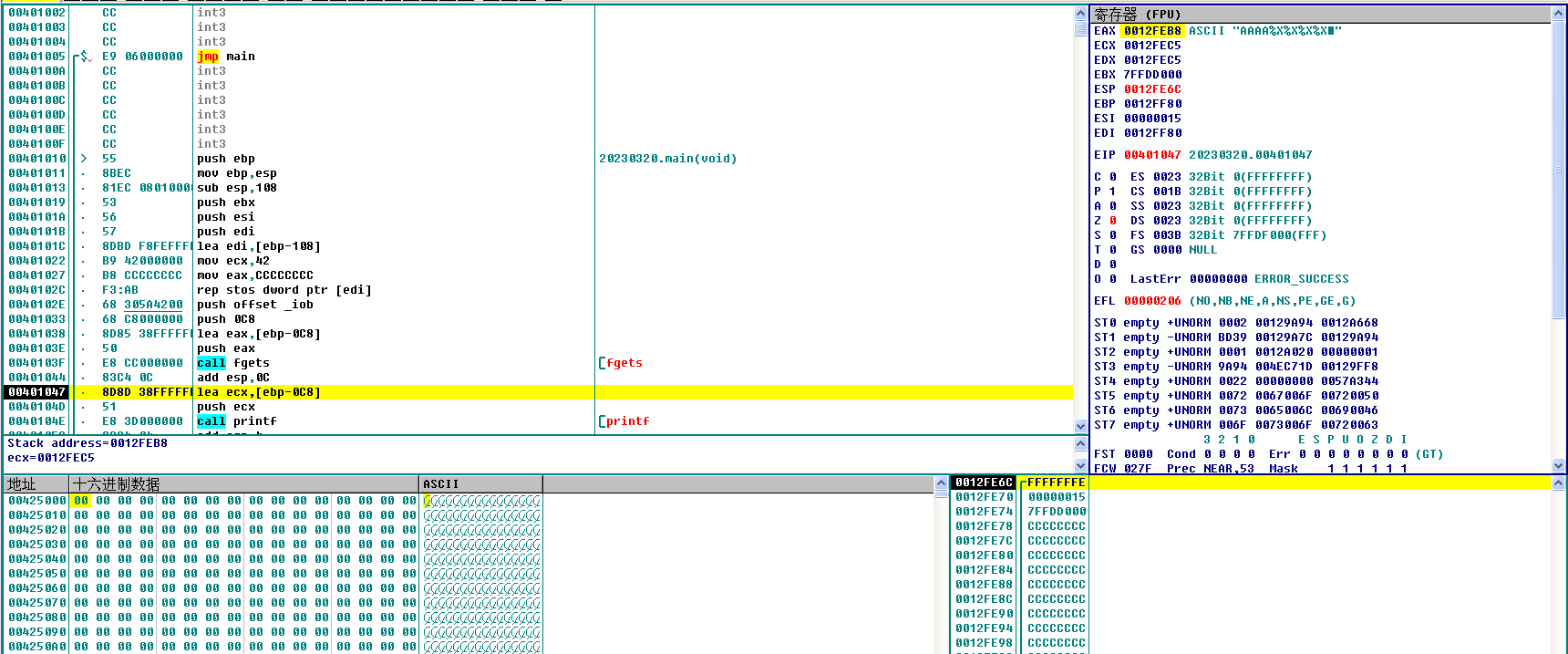


1. 0012FEB8里存着AAAA%X%X%X%X

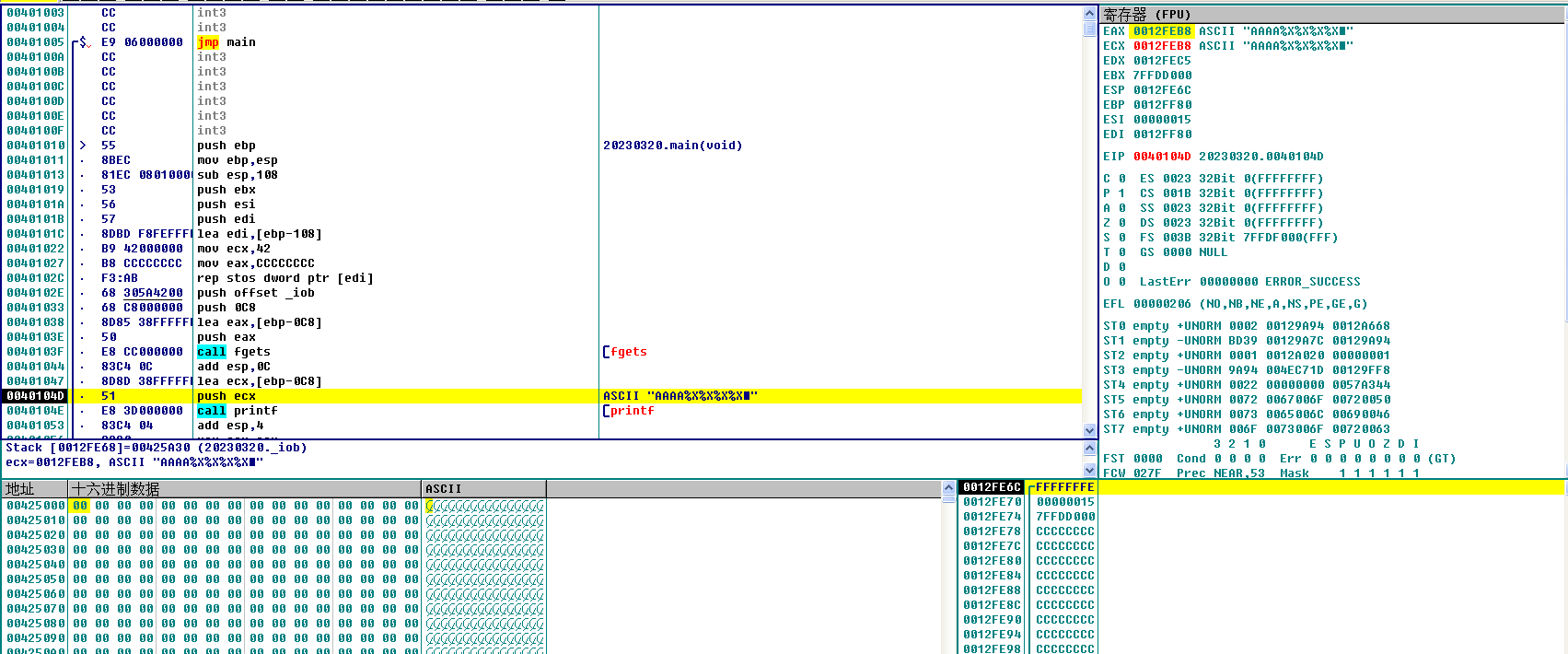


debug模式下，3个push对一个add，fgets调用之后恢复栈帧状态

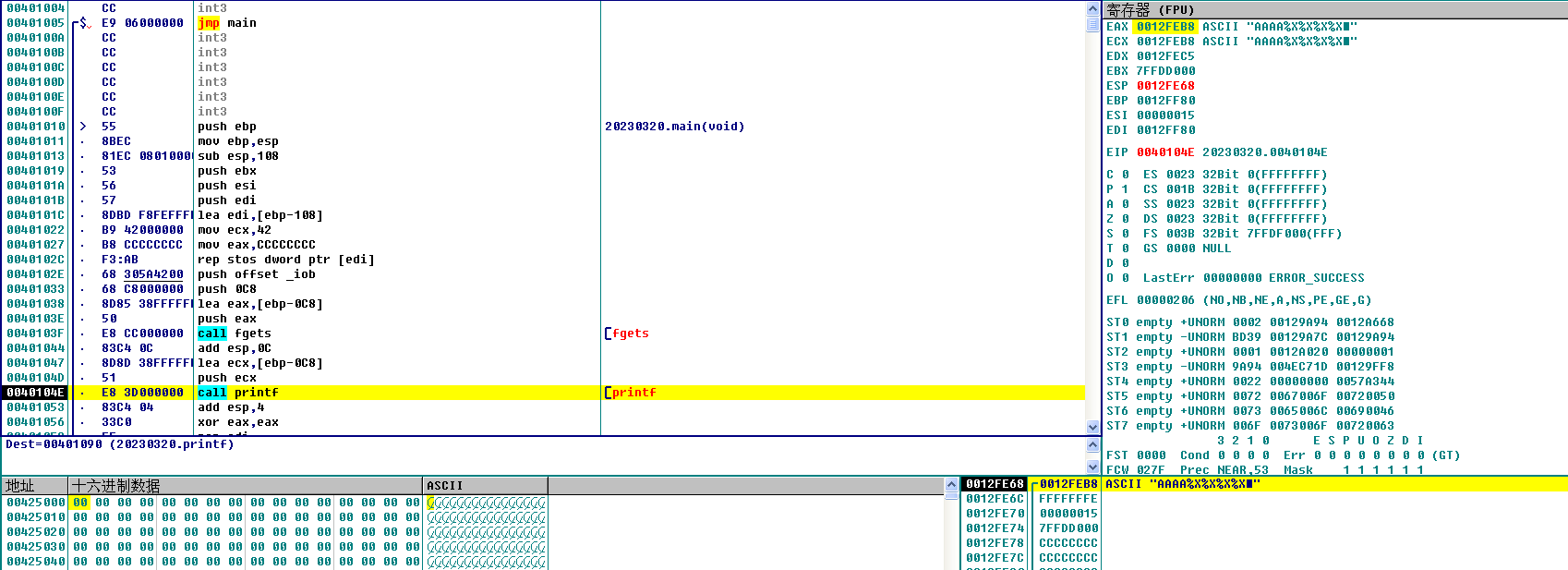
1. esp=0012FE6C,0012FE6C里存的是FFFFFFFE，esi=00000015，00000015地址是0012FE70，ebx=7FFDD000，7FFDD000地址是0012FE74



1. F8，ecx=0012FEB8,字符数组的起始地址

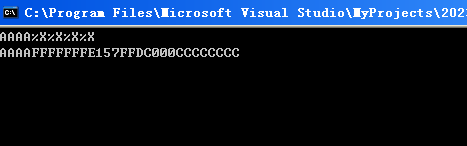


1. 作为参数入栈

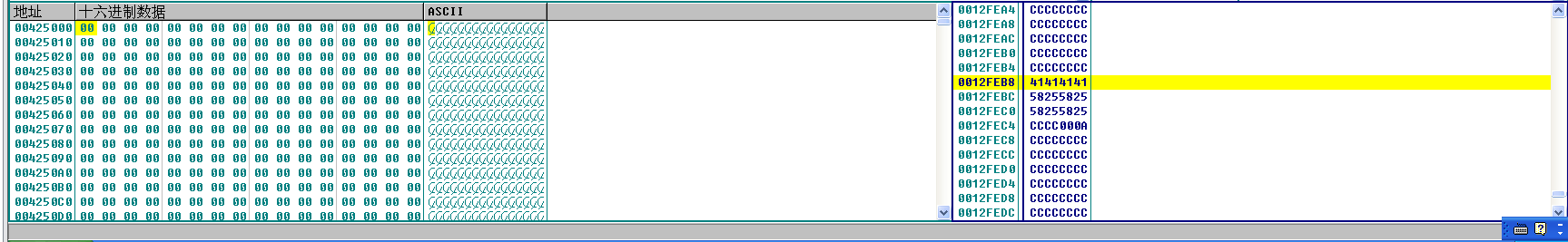


esp=0012FE68,地址0012FE68存的是0012FEB8，也就是字符串的起始地址，esi=00000015，00000015地址是0012FE70，ebx=7FFDD000，7FFDD000地址是0012FE74。

1. 字符串从地址0012FEB8开始打印输出AAAA，然后找下一个参数FFFFFFFE，找下一个参数00000015，找下一个参数7FFDD000，找下一个参数CCCCCCCC。

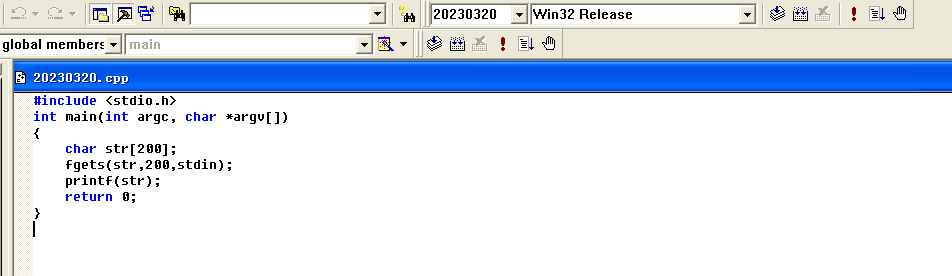


1. 地址0012FEB8的数据41414141，也就是AAAA

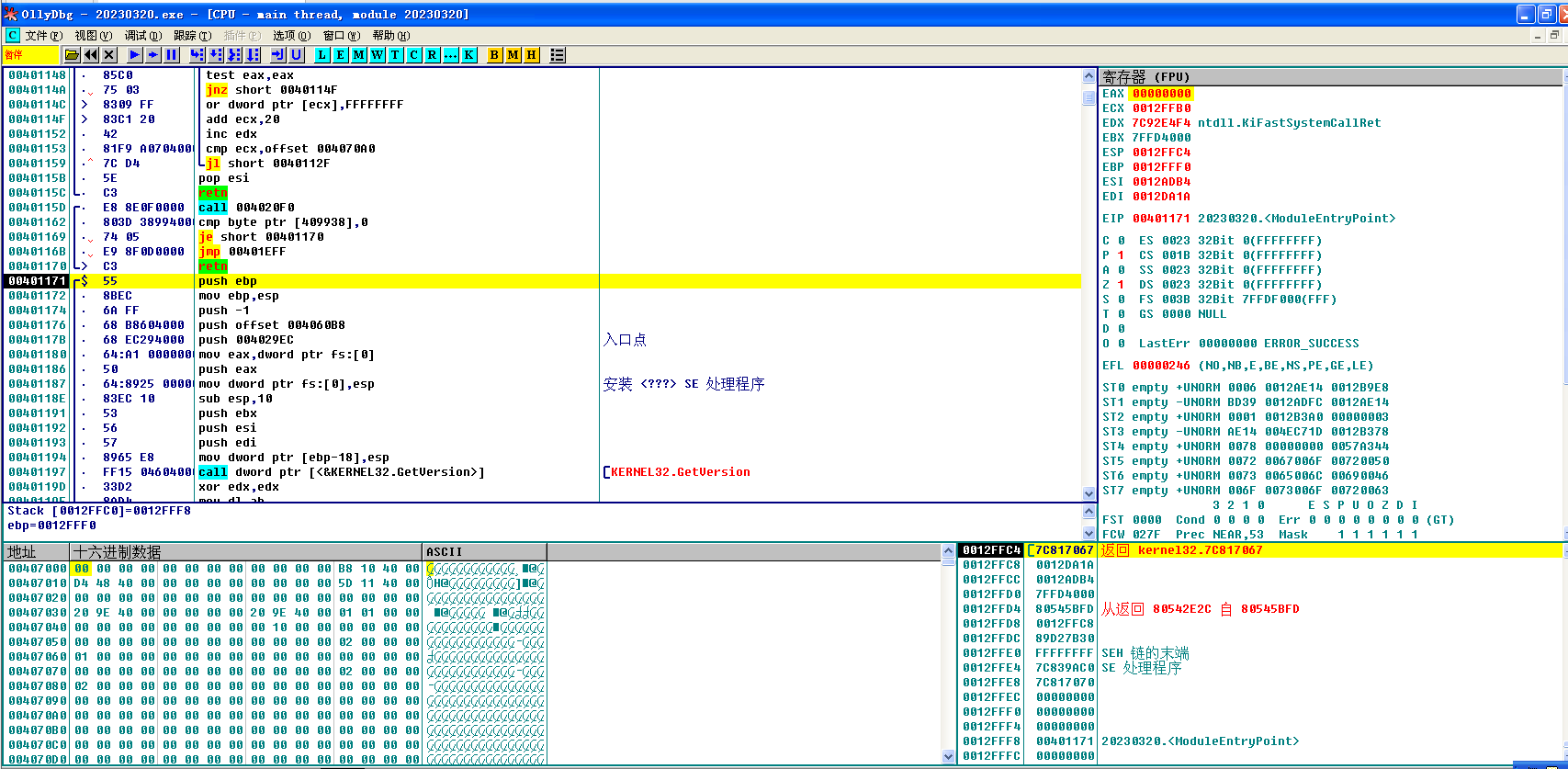


debug模式下给的空间，远比字符串要大，就全是CCCCCCCC

1. **Release模式**
2. 调成Release模式

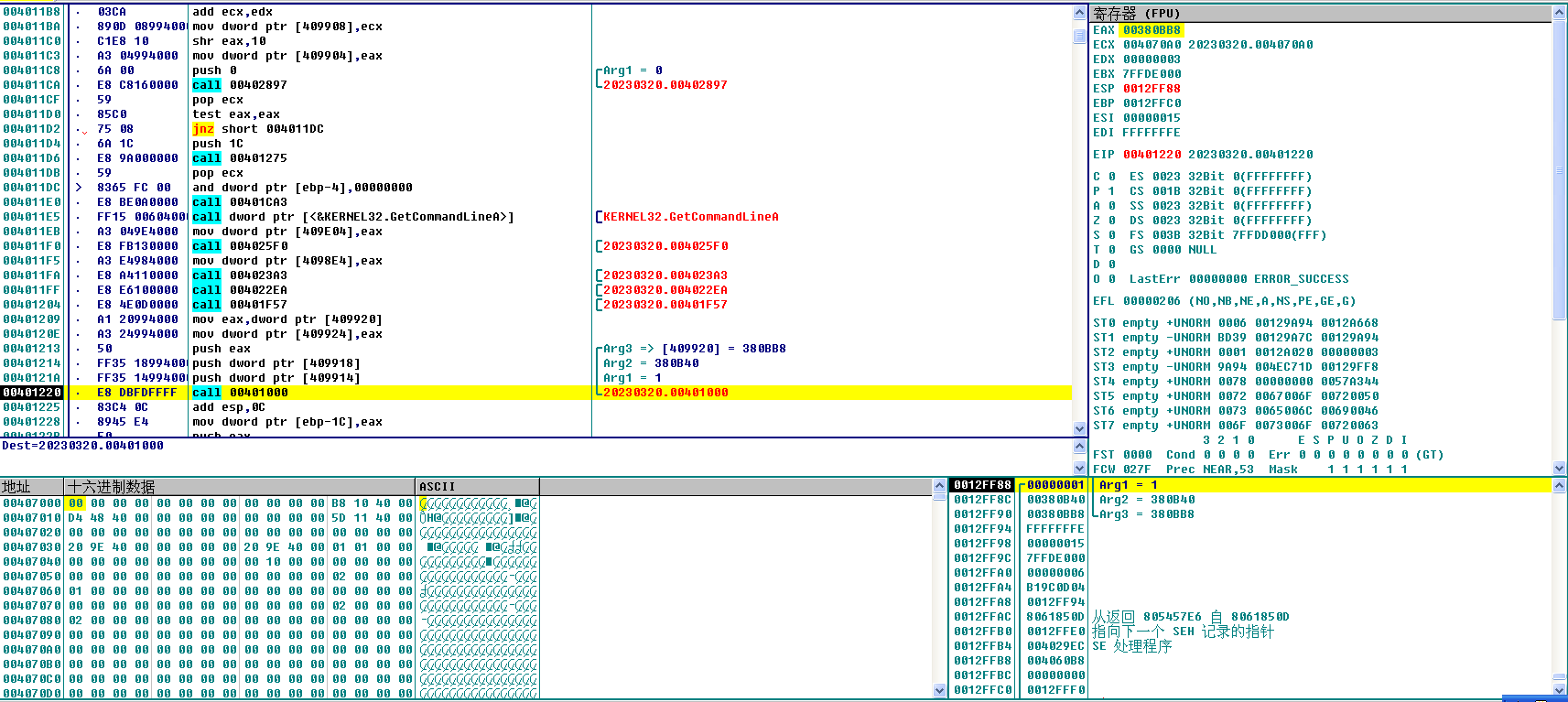
****

1. OLLYDBG查看

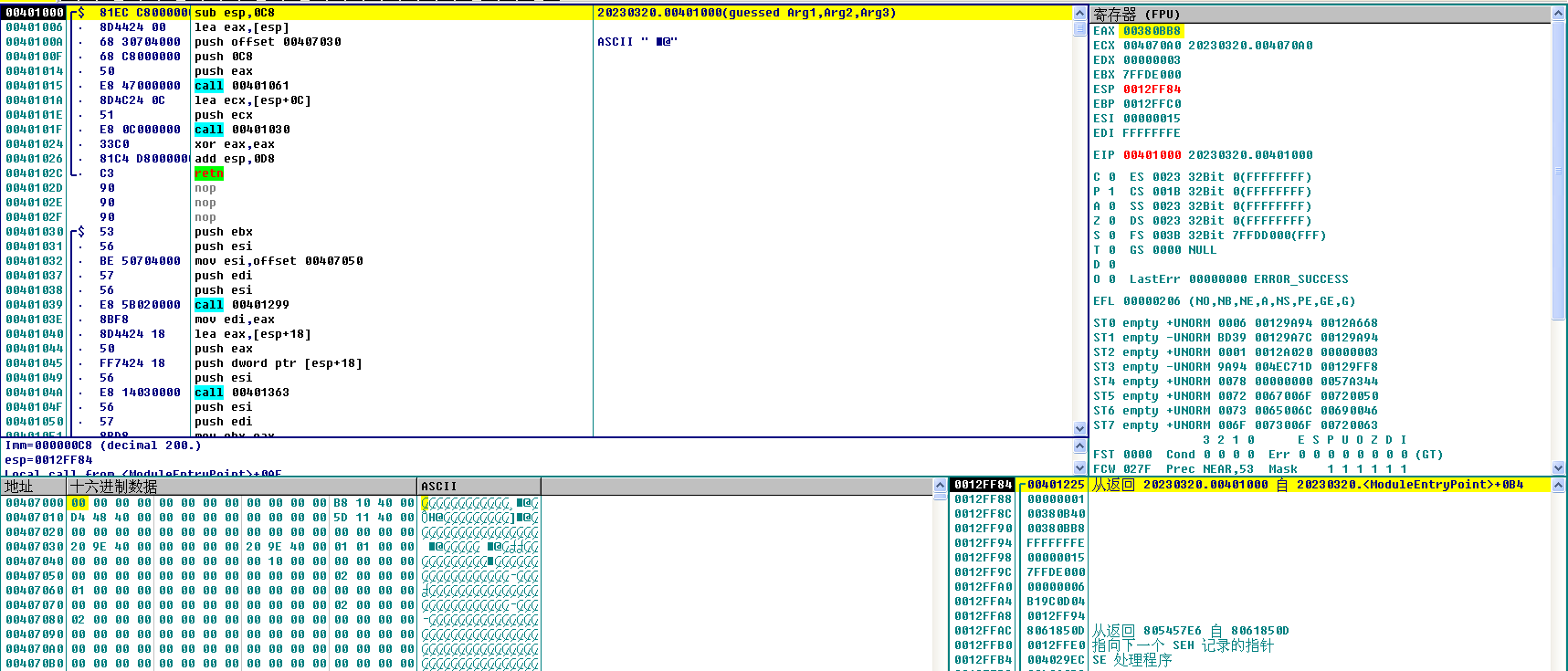


显然release模式代码更加紧凑，调试信息也少了一些。

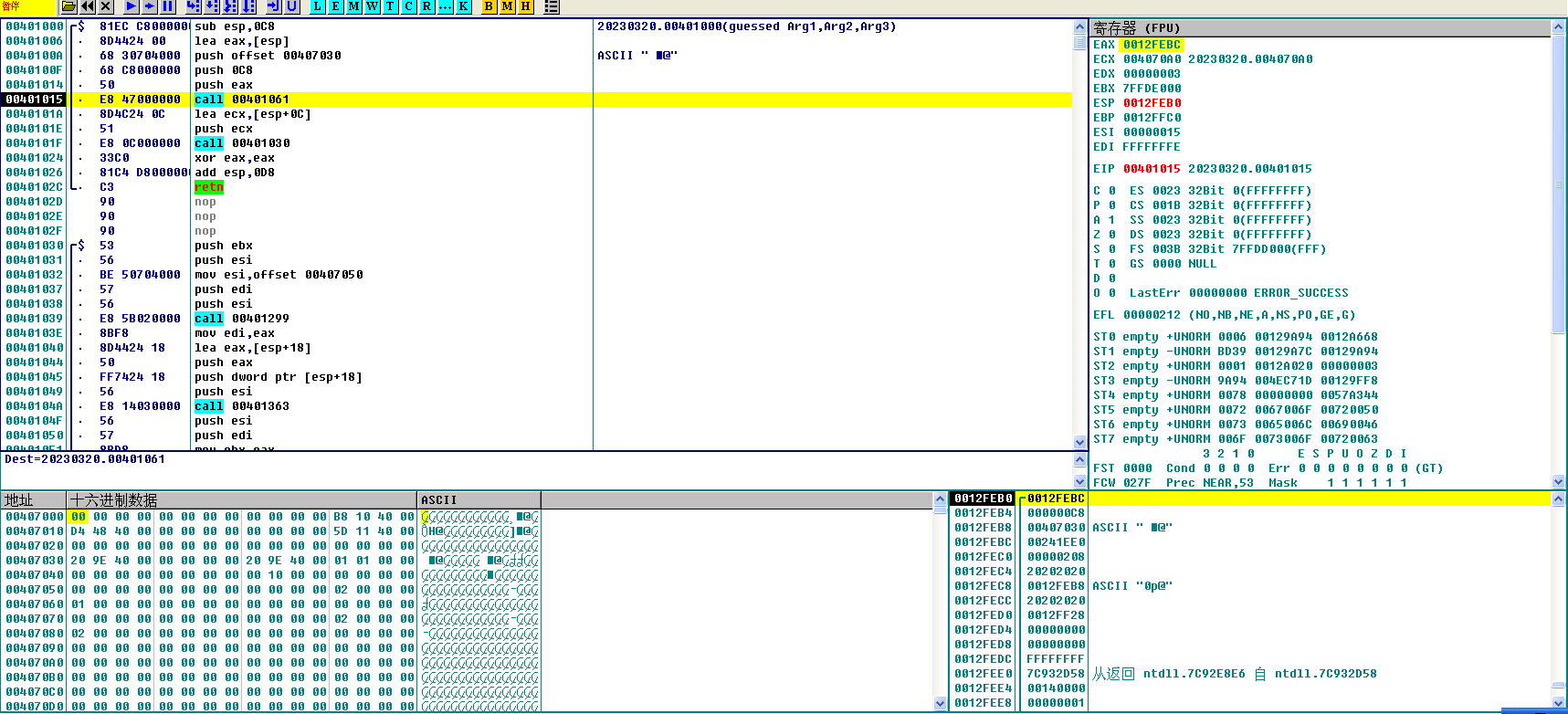
1. push，push，push和Arg1=1辅助找到主函数入口，00401220，call 00401000可直接定位到主函数的地址



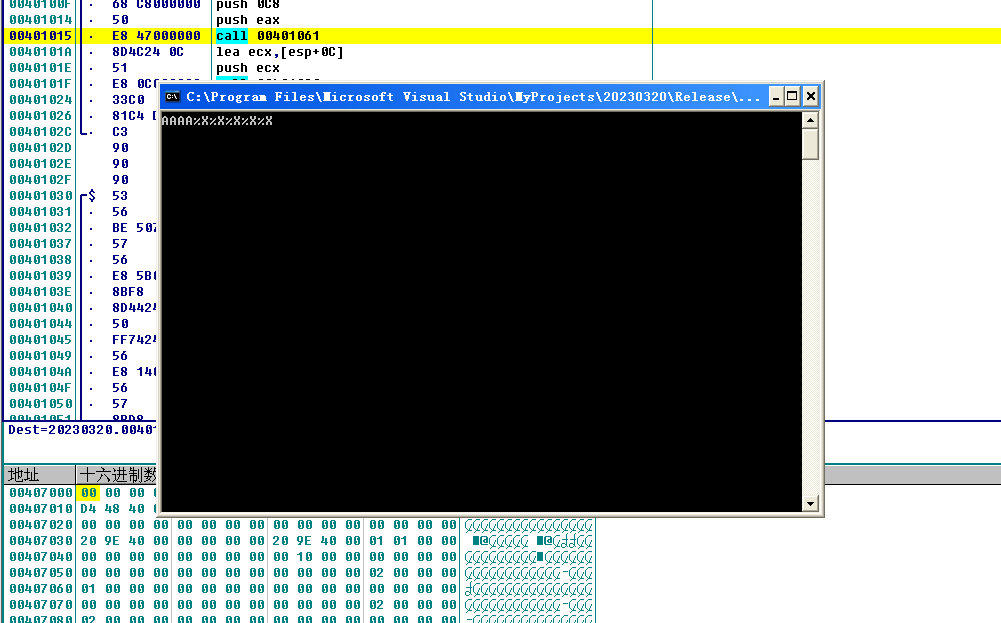
1. F7，直接定位到函数主函数的地址，sub esp,0C8，只抬高了200，也就是只给局部变量留了刚刚好的内存空间。Release模式下也没有类似debug模式下push寄存器的值的代码。



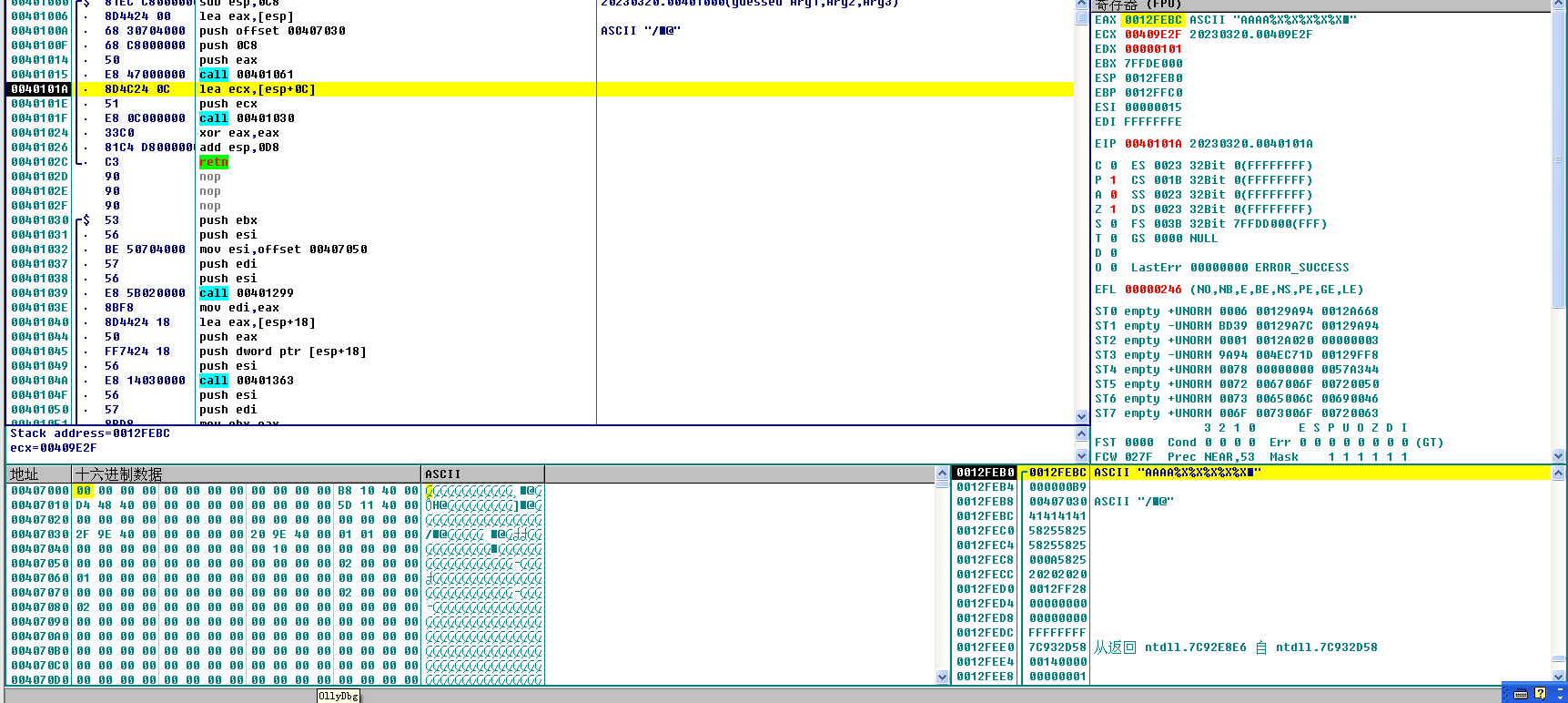
1. F7，call 00401061调用前有3个push，push eax作用是将eip的值赋值给eax。0012FEBC就是要输入的字符串数组的地址。



1. F8，输入AAAA%X%X%X%X

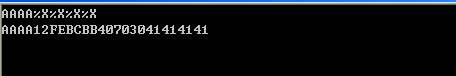


1. 回车，输入完毕



call调用后，并没有像debug模式中用add eip恢复栈帧。0012FEBC里的数据是41414141，也就是AAAA。

1. 输出结果



地址0012FEAC的参数数据是0012FEBC，也就是字符串的入口地址，输出的是AAAA，然后按照地址再往后调用4个地址的参数数据，即0012FEBC，000000BB，00407030,41414141。

最后retn之前的xor eax,eax和add esp，0C8用来恢复函数调用前原来的状态。

**心得体会：**

执行printf(str)语句的时候，Debug模式和Relase模式的栈帧结构：



Release模式采用了更高的优化级别，以尽可能地提高代码的执行效率，而Debug模式则不进行优化，以方便调试程序。Release模式不包含调试信息，以减小程序体积，而Debug模式包含完整的调试信息，方便程序员在调试时查看变量、函数等信息。Release模式采用的是编译器的默认设置，而Debug模式通常会开启额外的检查和调试代码，以方便调试程序。Release模式生成的代码更加紧凑，运行速度更快，而Debug模式生成的代码比较臃肿，运行速度较慢。