**[linux源码阅读笔记 move\_to\_user\_mode()解析](https://www.cnblogs.com/elnino/p/4342477.html)**

在linux 0.11版本源代码中，在文件linux/include/asm/system.h中有一个宏定义  move\_to\_user\_mode()

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14 | 1 #define move\_to\_user\_mode() \   2 \_\_asm\_\_ ("movl %%esp,%%eax\n\t" \   3         "pushl $0x17\n\t" \   4         "pushl %%eax\n\t" \   5         "pushfl\n\t" \   6         "pushl $0x0f\n\t" \   7         "pushl $1f\n\t" \   8         "iret\n" \   9         "1:\tmovl $0x17,%%eax\n\t" \  10         "movw %%ax,%%ds\n\t" \  11         "movw %%ax,%%es\n\t" \  12         "movw %%ax,%%fs\n\t" \  13         "movw %%ax,%%gs" \  14         :::"ax") |

　　这个宏定义用于在内核初始化结束时“切换到”初始进程（任务0）。

　　首先解释指令iret.

　　iret 指令（interrupt return）中断返回，终端服务程序的最后一条指令。iret指令将推入堆栈的段地址和偏移地址弹出，使程序返回到原来中断发生的地方。它将产生以下三点效应：

1.恢复[IP](http://baike.baidu.com/view/8370.htm)(instruction pointer)：（IP）←（（SP）+1:（SP）），（SP）←（SP）+2

2.恢复[CS](http://baike.baidu.com/view/456.htm)(code segment)：（CS）←（（SP）+1:（SP）），（SP）←（SP）+2

3.恢复中断前的[PSW](http://baike.baidu.com/view/809499.htm)(program status word),即恢复中断前的[标志寄存器](http://baike.baidu.com/view/1845107.htm)的状态。

（FR）←（（SP）+1:（SP）），（SP）←（SP）+2

以上操作按顺序进行。

　　当使用IRET指令返回到相同保护级别的任务时，IRET会从堆栈弹出代码段选择子及指令指针分别到CS与IP寄存器，并弹出标志寄存器内容到EFLAGS寄存器。

*当使用IRET指令返回到一个不同的保护级别时，IRET不仅会从堆栈弹出以上内容，还会弹出堆栈段选择子及堆栈指针分别到SS与SP寄存器。*

　　因此，这段程序的大概意思是先将任务0所需要的各个寄存器的值压栈，压栈后执行IRET指令，利用该中断返回指令将各个寄存器设置为我们所理想的值。

　　但是，程序中压入了几个常数，0x17,0x0f,1f是什么意思呢。

　　分两类，0x17,0x0f是段选择子。段选择子用于保护模式下的寻址。0-1位表示请求的特权级，0表示系统级，3表示用户级。2位用于选择全局描述符表还是局部描述符表。

　　3-15位是描述符表项索引。

　　1f表示将下面标号1的程序段的偏移地址入栈。