## 操作系统

## 实验报告

# 班号： 1603102

# 学号：1163200511

# 姓名： 李扬名

1. 请简述head.s 的工作原理。

源码段head.s分为初始化和执行两个部分。初始化阶段建立中断描述附表IDT、全局描述符表GDT­、设定定时芯片等。而在执行阶段则通过硬件的定时中断（0x08）切换两个任务打印字符A、B。

1. 请记录head.s 的内存分布状况，写明每个数据段，代码段，栈段的起始与终止的内存地址。

将head.s切分为初始化设置代码、IDT、GDT、Init Stack（内核堆栈空间）、LDT-0、TSS-0、Stack-0（任务0的堆栈段）、IDT-1、TSS-1、Stack-1（任务1的堆栈段）以及任务0和1的代码，如下表：

|  |  |  |
| --- | --- | --- |
| **名称** | **起始** | **终止** |
| 初始代码 | 0x00000000 | 0x000001A3 |
| IDT | 0x00000198 | 0x00000997 |
| GDT | 0x00000998 | 0x000009D7 |
| Init Stack | 0x000009D8 | 0x00000BD7 |
| IDT-0 | 0x00000BE0 | 0x00000BF7 |
| TSS-0 | 0x00000BF8 | 0x00000C5F |
| Stack-0 | 0x00000C60 | 0x00000E5F |
| IDT-1 | 0x00000E60 | 0x00000E77 |
| TSS-1 | 0x00000E77 | 0x00000EDF |
| Stack-1 | 0x00000EE0 | 0x000010DF |
| Task-0 | 0x000010E0 | 0x000010F3 |
| Task-1 | 0x000010F4 | 0x00001107 |

注意这里Init Stack是从高地址指向低地址的，因此起始比终止大。

1. 简述Head.s 57~62行在做什么？简述iret执行后，pc如何找到下一条指令？并记录iret执行前后，栈是如何变化的。
2. pushl $0x17 # 入栈任务0的堆栈段选择子

pushl $init\_stack # 入栈内核堆栈段的基址地址

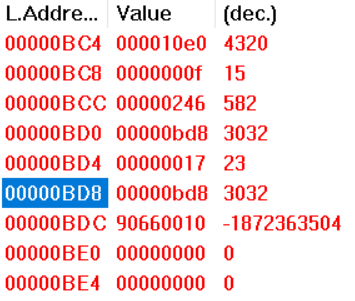
pushfl # 入栈标志寄存器EFLAGS

pushl $0x0f # 入栈任务0的代码段选择子

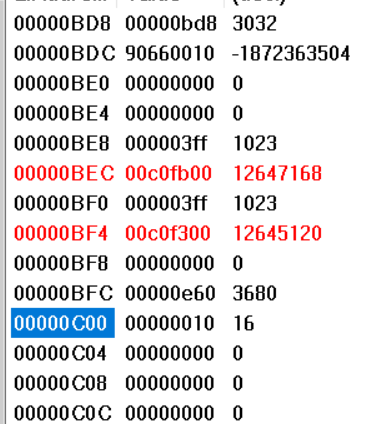
pushl $task0 # 入栈任务0的代码段基址

iret # 使用弾栈中断返回，特权降低

1. iret指令通过之前压入栈中task-0的代码段基地址修改代码段寄存器ESP的值，并设置EIP=0从而跳转到了task-0代码段。
2. 执行iret 指令前：

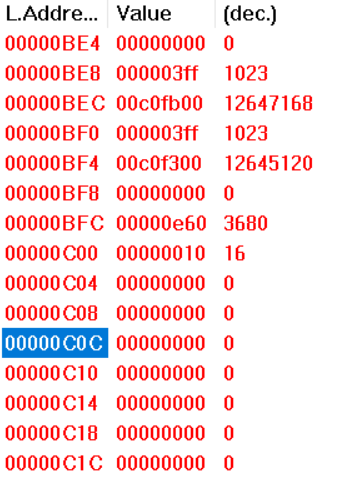


执行 iret 指令后：

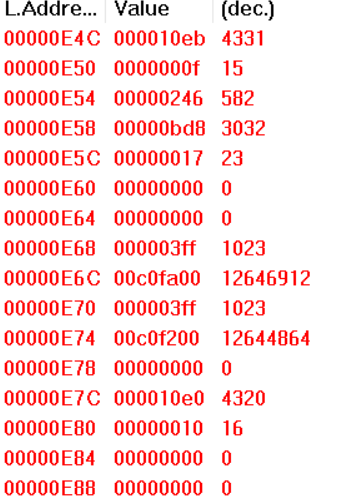


不难发现，经过iret指令后，堆栈由内核堆栈（其中被push了许多信息）变化到了task-0的堆栈空间，其中还未有内容。

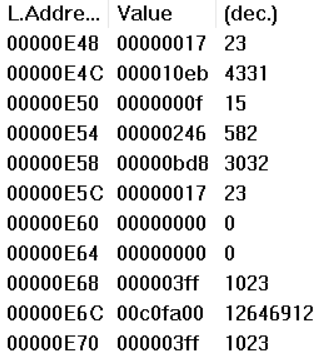
1. 当任务进行系统调用时，即int 0x80时，记录栈的变化情况。当执行完system\_interrupt函数，执行153行iret时，记录栈的变化情况。此外，当进入和退出system\_interrupt时，都发生了模式切换，请总结模式切换时，特权级是如何改变的？栈切换吗？如何进行切换的？
2. 执行int 0x80之前的栈（task-0堆栈）：



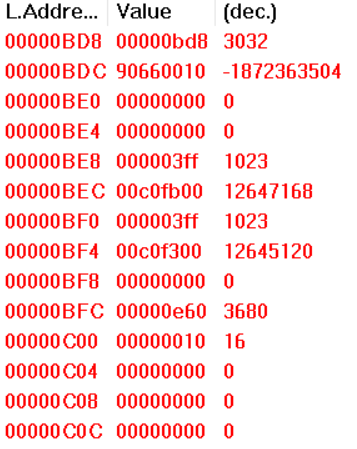
执行int 0x80后的堆栈（内核堆栈）：



1. 执行iret前的堆栈（内核堆栈）：



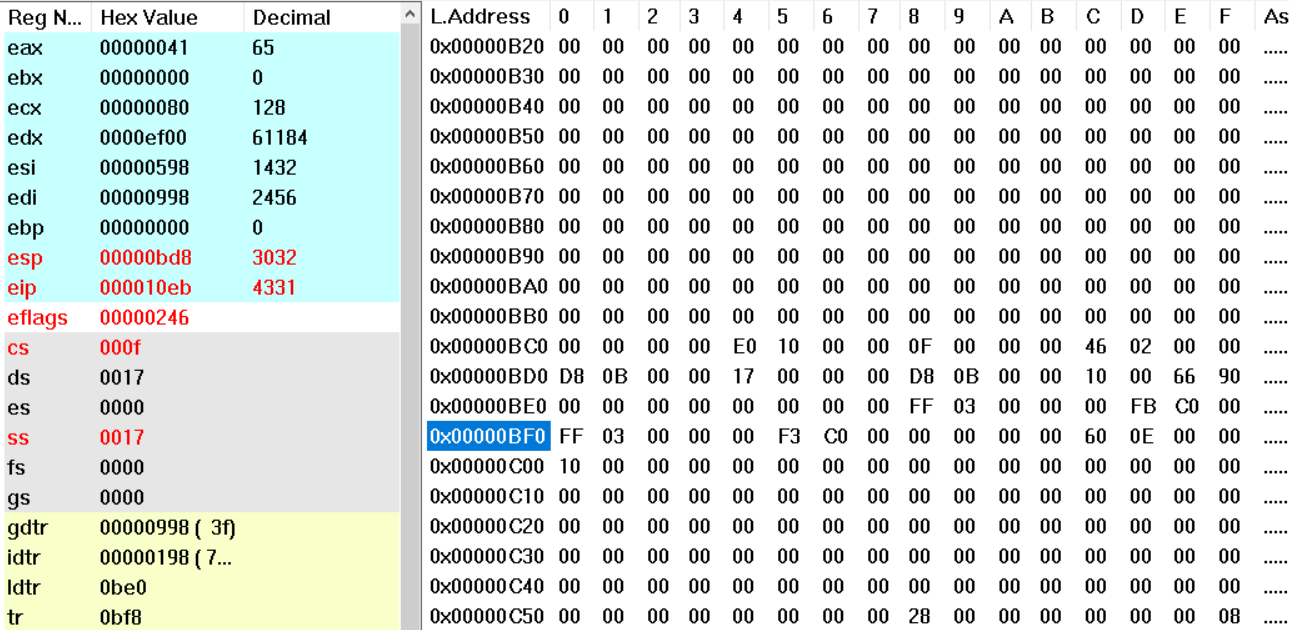
执行iret后的堆栈（task-0堆栈）：



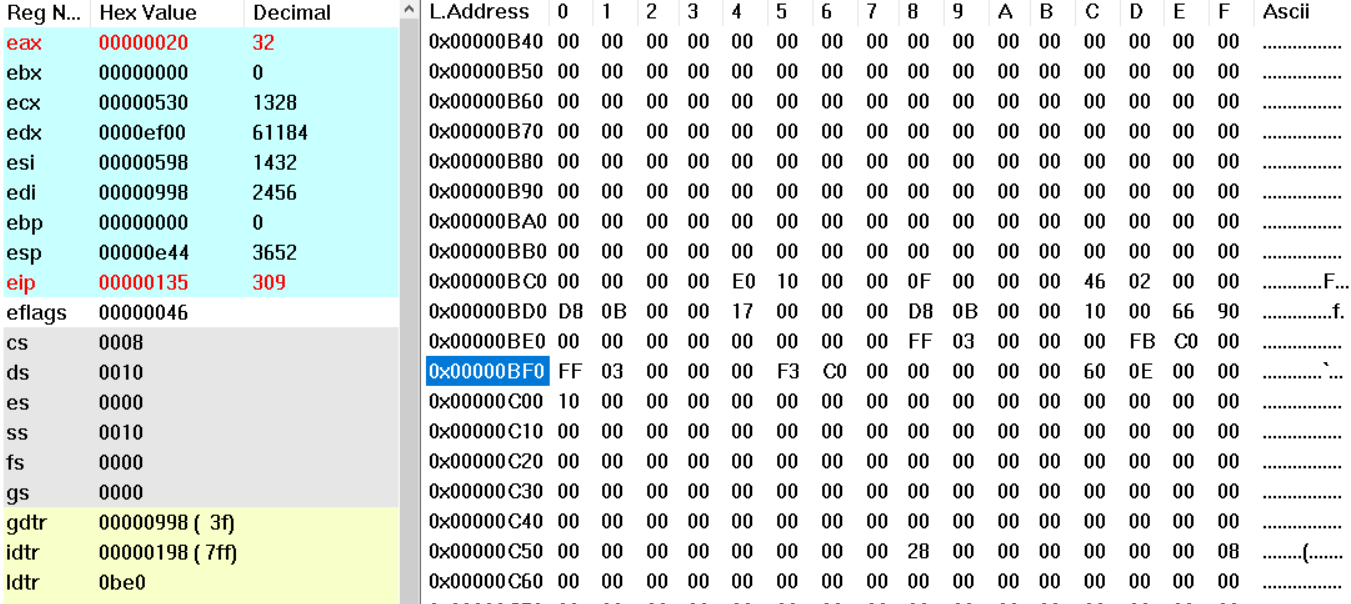
1. 在以上过程中，栈和特权级经历了：用户堆栈（特权级3）->内核堆栈（特权级0）->用户堆栈（特权级3）。类似操作系统的系统调用，特权级低的应用程序通过中断调用特权级高的内核代码。
2. 当时钟中断发生，进入到timer\_interrupt程序，请**详细记录**从任务0切换到任务1，以及又过了10ms，从任务1切换回到任务0，整个流程是怎样的？TSS是如何变化的？各个寄存器的值是如何变化的？请详细总结任务切换的过程。

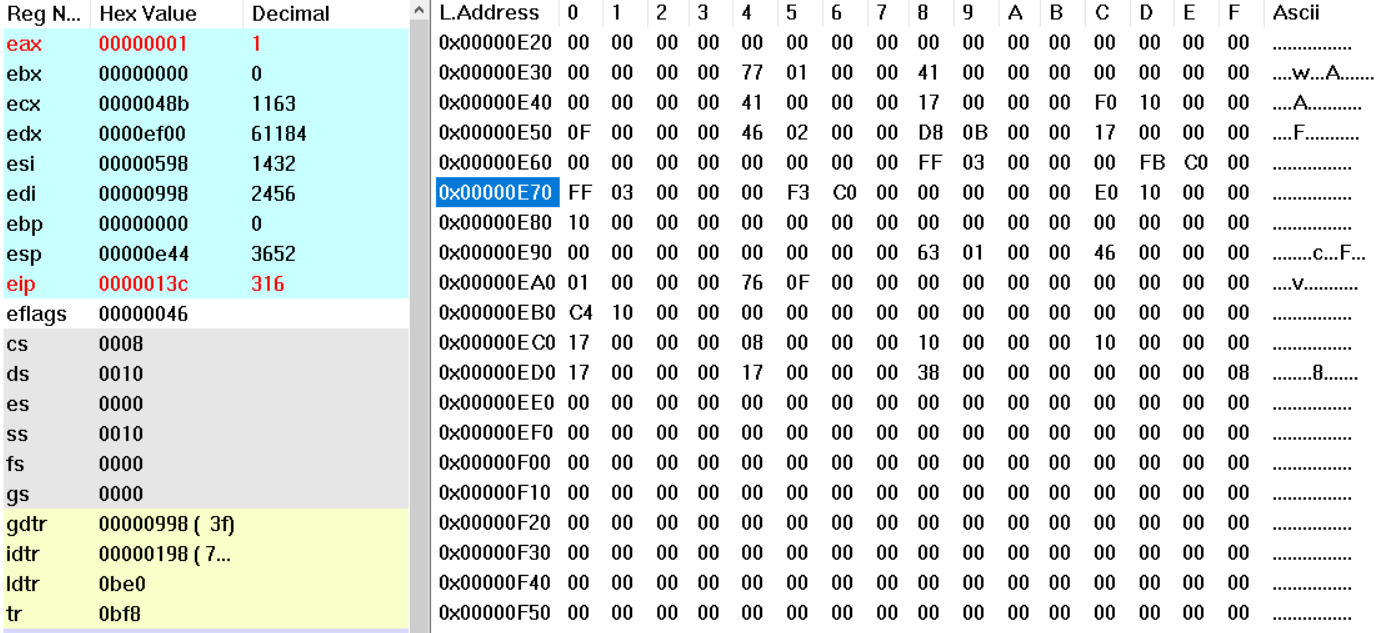
注意：这里的断点（0x12c）需要在iret到task-0开始打印之前。上述过程一共有以下5个阶段，其中的堆栈+寄存器变化如下：

1. Task-0到0x08中断之前（TSS-0）：

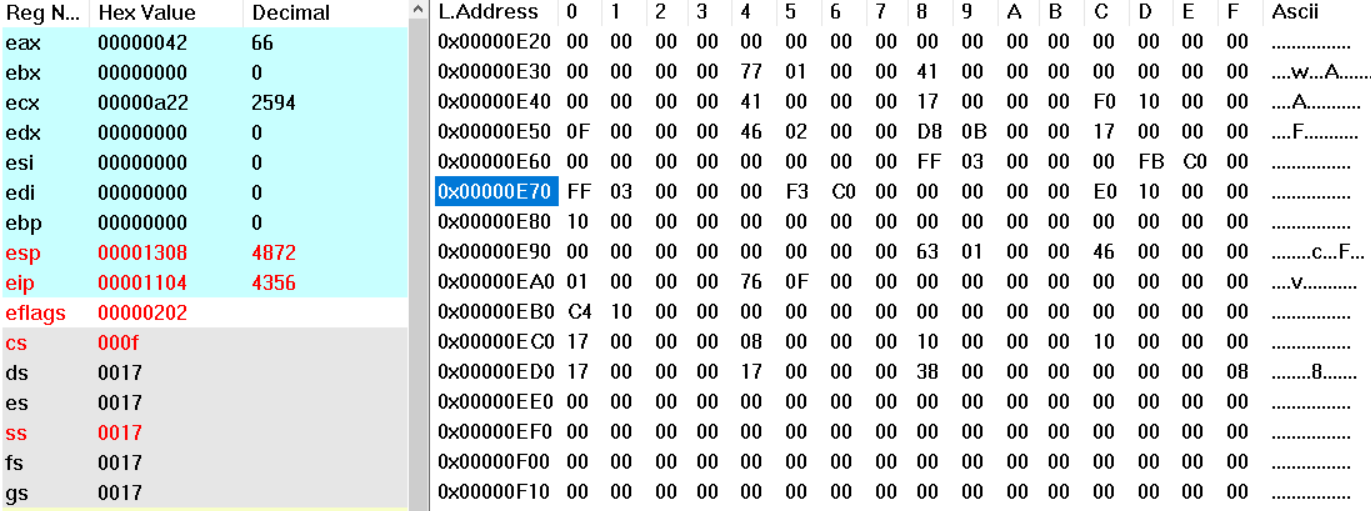


1. Task-0被切换到0x08后之间（TSS-0，1）：

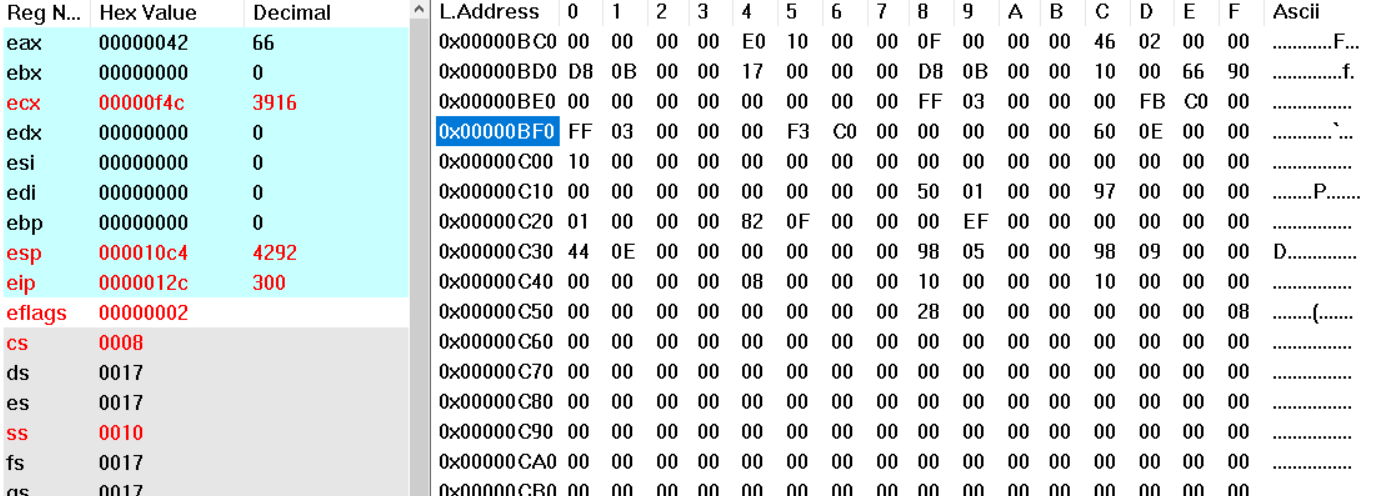


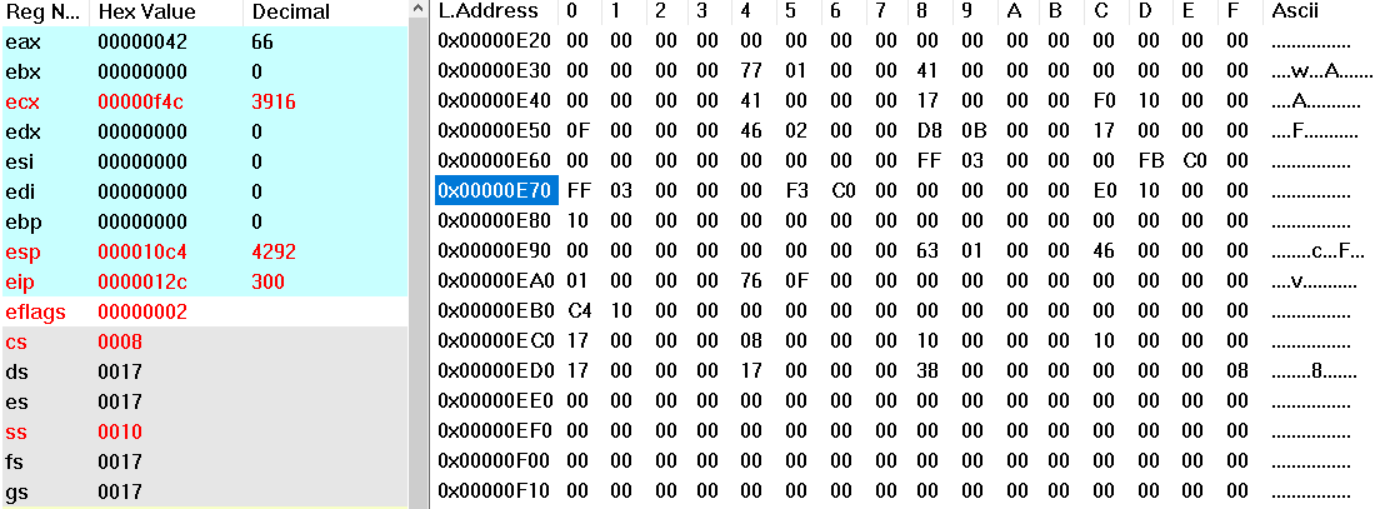


1. 0x08切换到Task-1之后（TSS-1）：

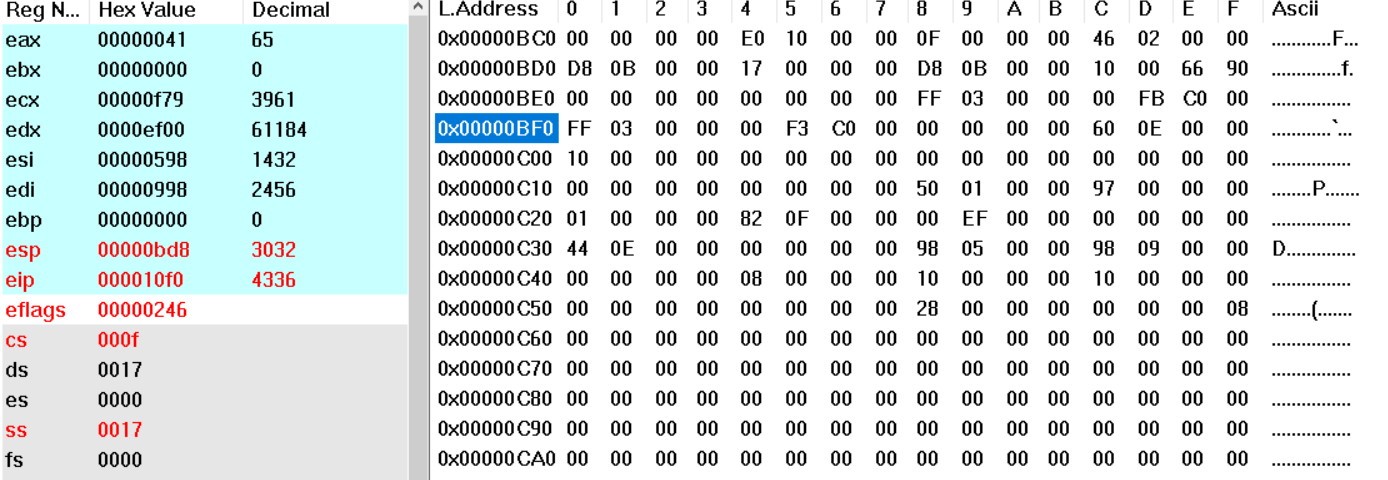


1. Task1切换到0x08之后（TSS-0，1）:





E，0x08切换到Task-0后（TSS-0）:



在任务切换的过程中，观察cs可知CPL（当前特权级）在用户程序和内核代码之间切换（3->0->3->…）。而切换任务时TSS保存了所有通用寄存器、状态寄存器等的取值，直至调用时被恢复。