

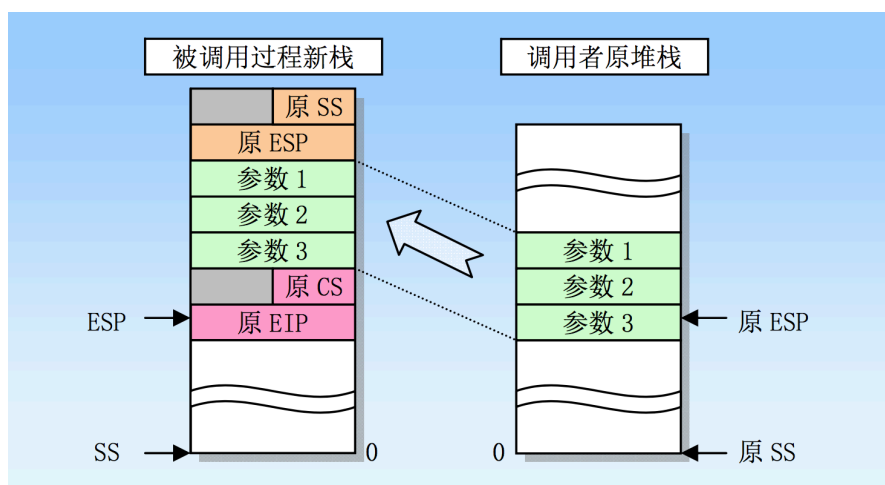
# 1 实现特权级转移

## 1.1 理论知识

特权级转移需要借助堆栈切换。当调用门用于把程序控制转移到一个更高级别的非一致性代码段时，处理器会自动切换到目的代码段特权级的堆栈。此时处理器会按照以下步骤切换堆栈：

- 当前任务的 TSS 段存放着特权级 0、1 和 2 的堆栈的初始指针值。处理器会将目的代码段的 DPL 作为新任务的 CPL，并从 TSS 中选择新栈的 SS 和 ESP。
- 将 SS 和 ESP 寄存器的当前值压入新栈，并将新栈的段选择符和栈指针加载到 SS 和 ESP。
- 将调用门描述符中指定的参数从当前栈压入新栈。参数数目由调用门描述符中的 PARAM COUNT 字段决定。
- 将 CS 和 EIP 寄存器的当前值压入新栈，并将目的代码段选择符加载到 CS，将调用门选择符中的偏移值加载到 EIP 中。

调用过程如下图所示：



当调用过程结束后，处理器使用 RET 执行远返回到一个调用过程。此时 CPU 会执行以下步骤：

- 检查保存的 CS 寄存器中的 RPL 字段值，以确定返回时特权级是否需要改变。
- 弹出新栈中的 CS 和 EIP 值，并且检查代码段描述符的 DPL 和代码段选择符的 RPL。

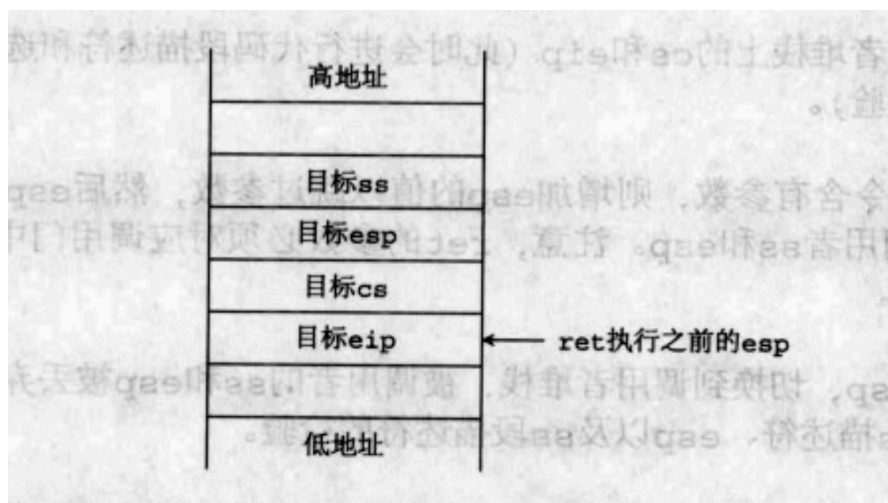
- 如果返回过程会改变特权级，而且此时 RET 指令包含一个参数个数操作数，那么就需要在弹出 CS 和 EIP 之后，将参数个数值加载到 ESP 寄存器中，用于丢弃新栈中的参数。
- 弹出 SS 和 ESP，从而切换回调用者的堆栈。
- 检查 DS、ES、FS 和 GS，如果其中的段选择符指向的段描述符的 DPL 小于新 CPL(仅适用于一致代码段)，处理器将用空选择符来加载这个段寄存器。

综上，使用调用门实现不同特权级之间的调用可以分为两个过程，一个是通过调用门和 call 指令实现从低特权级转移到高特权级，另一个是通过 ret 指令实现从高特权级到低特权级。

## 1.2 代码实现

### 1.2.1 高特权级到低特权级

根据上一节可知，处理器通过 ret 指令实现从高特权级到低特权级。在 ret 指令执行之前，堆栈中应该已经有了 ss、esp、cs 和 eip。如下图所示：



首先添加一个特权级为 3 的代码段，为了实现代码段转移，我们需要添加一个代码段和相应的堆栈段。先是在 GDT 表中添加该代码段和堆栈段的描述符，然后定义堆栈段 ring3 和代码段 ring3。随后，我们在 32 位代码段中通过执行 retf 指令跳转到 ring3 代码段中。

```
1 ; 在GDT中添加相应的代码段和堆栈段
2 [SECTION .gdt]
3 ; ...
4 ; 特权级为3
```

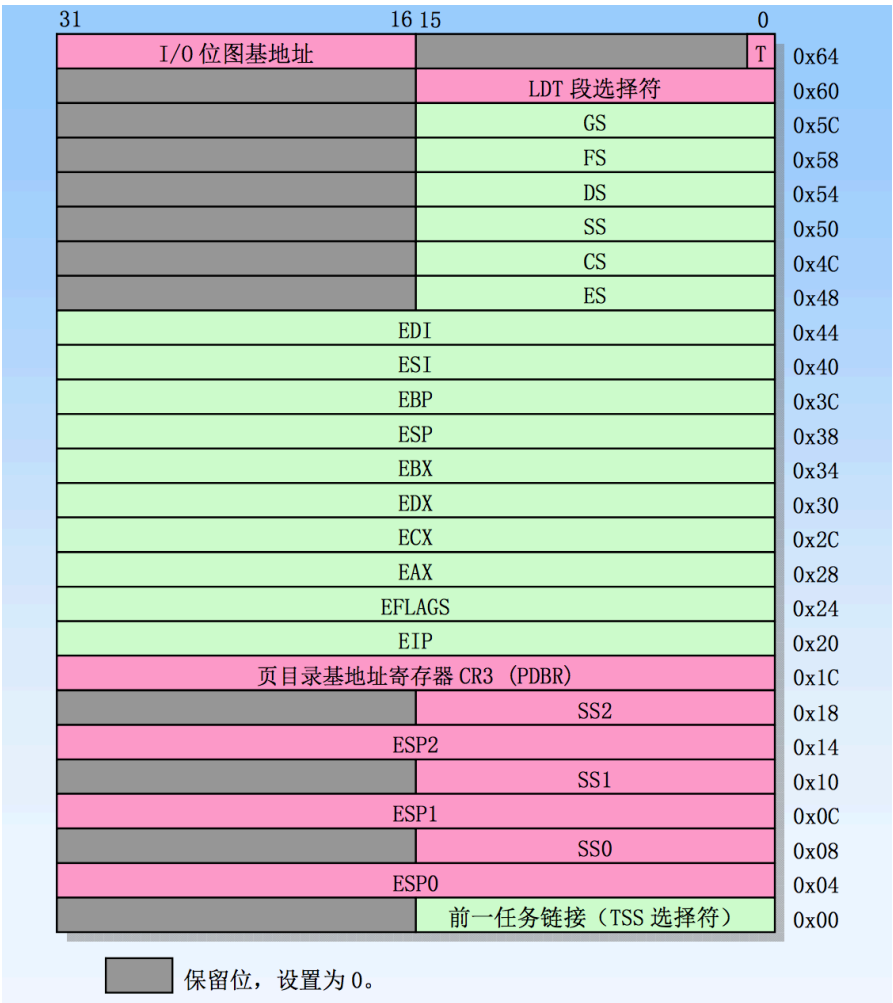
```

5 LABEL_DESC_CODE_RING3: Descriptor 0, SegCodeRing3Len-1, DA_C+DA_32+DA_DPL3
6 LABEL_DESC_STACK3: Descriptor 0, TopOfStack3, DA_DRWA+DA_32+DA_DPL3
7 ; ...
8 ; 请求特权级为3
9 SelectorCodeRing3 equ LABEL_DESC_CODE_RING3-LABEL_GDT+SA_RPL3
10 SelectorStack3 equ LABEL_DESC_STACK3-LABEL_GDT+SA_RPL3
11 ; ...
12 ; 定义堆栈段 ring3
13 [SECTION .s3]
14 ALIGN 32
15 [BITS 32]
16 LABEL_STACK3:
17     ; 该堆栈段有512个字节大小
18     times 512 db 0
19 TopOfStack3 equ $-LABEL_STACK3-1
20 ; ...
21 ; 定义代码段 ring3
22 [SECTION .ring3]
23 ALIGN 32
24 [BITS 32]
25 LABEL_CODE_RING3:
26     mov ax, SelectorVideo
27     mov gs, ax
28     mov edi, (80*14 + 0) * 2
29     mov ah, 0Ch
30     mov al, '3'
31     mov [gs:edi], ax
32     jmp $
33 SegCodeRing3Len equ $-LABEL_CODE_RING3
34 ; ...
35 [SECTION .s32]
36 [BITS 32]
37 LABEL_SEG_CODE32:
38     ; ...
39     ; 压入 ss
40     push SelectorStack3
41     ; 压入 esp
42     push TopOfStack3
43     ; 压入 cs
44     push SelectorCodeRing3
45     ; 压入 eip
46     push 0
47     ; 执行 ret 指令
48     retf

```

### 1.2.2 低特权级到高特权级

从低特权级到高特权级转移的时候，需要用到 TSS，所以要添加 TSS 段。需要根据 TSS 的结构定义 TSS，TSS 的结构如下图：



```
1 [SECTION .gdt]
2 LABEL_DESC_TSS: Descriptor 0, TSSLen-1, DA_386TSS
3 ; ...
4 SelectorTSS equ LABEL_DESC_TSS-LABEL_GDT
5 ; ...
6 [SECTION .tss]
7 ALIGN
8 [BITS 32]
9 LABEL_TSS:
10     DD 0 ; 前一任务链接
11     DD TopOfStack ; 0级堆栈段基址
12     DD SelectorStack ; 0级堆栈选择符
13     DD 0 ; 1级堆栈段基址
14     DD 0 ; 1级堆栈选择符
15     DD 0 ; 2级堆栈段基址
16     DD 0 ; 2级堆栈选择符
17     DD 0 ; CR3
18     DD 0 ; EIP
```

```

19      DD 0 ; EFLAGS
20      DD 0 ; EIP
21      DD 0 ; EAX
22      DD 0 ; ECX
23      DD 0 ; EDX
24      DD 0 ; EBX
25      DD 0 ; ESP
26      DD 0 ; EBP
27      DD 0 ; ESI
28      DD 0 ; EDI
29      DD 0 ; ES
30      DD 0 ; CS
31      DD 0 ; SS
32      DD 0 ; DS
33      DD 0 ; FS
34      DD 0 ; GS
35      DD 0 ; LDT段选择符
36      DW 0 ; 调试陷阱T标志位
37      DW $-LABEL_TSS+2 ; I/O位图基址
38      DB 0ffh ; I/O位图结束标志
39      TSSLen equ $-LABEL_TSS

```

接着添加调用门，用于不同特权级的转移。调用门的添加步骤在第四次中已经讲到了，在这里就不再详细论述。添加调用门成功后，就可以在 ring3 代码段中通过调用门转移到特权级为 0 的代码段中。

```

1      [SECTION .gdt]
2      ; ...
3      ; 定义一个特权级为0的代码段
4      LABEL_DESC_CODE_TEST: Descriptor 0, SegCodeDestLen-1, DA_C+DA_32
5      ; 定义一个能够跳转到0特权级代码段的门描述符
6      LABEL_CALL_GATE_TEST: Gate SelectorCodeDest, 0, 0, DA_386Gate+DA_DPL3
7      ; ...
8      SelectorCallGateTest equ LABEL_CALL_GATE_TEST - LABEL_GDT
9
10     ; 在32位代码段中加载TSS描述符
11     ; 需要在特权级变换之前加载TSS描述符
12     [SECTION .s32]
13     ; ...
14     mov ax, SelectorTSS
15     ltr ax
16     ; ...
17
18     ; 在ring3代码段中通过调用门跳转到特权级为0的代码段中
19     [SECTION .ring3]
20     ALIGN 32
21     [BITS 32]
22     LABEL_CODE_RING3:
23         mov ax, SelectorVideo
24         mov gs, ax
25         mov edi, (80 * 14 + 0) * 2
26         mov ah, 0Ch
27         mov al, '3'
28         mov [gs:edi], ax
29         call SelectorCallGateTest:0

```

```
30     jmp $  
31     SegCodeRing3Len equ $-LABEL_CODE_RING3
```