内核：中断异常和任务切换

中断与异常

由CPU外部设备引起的外部事件如I/O中断、时钟中断、控制台中断等是异步产生的（即产生的时刻不确定），与CPU的执行无关，我们称之为异步中断(asynchronous interrupt)也称外部中断,简称中断(interrupt)。

而把在CPU执行指令期间检测到不正常的或非法的条件(如除零错、地址访问越界)所引起的内部事件称作同步中断(synchronous interrupt)，也称内部中断，简称异常(exception)。

把在程序中使用请求系统服务的系统调用而引发的事件，称作陷入中断(trap interrupt)，也称软中断(soft interrupt)，系统调用(system call)简称trap。

中断请求：IRQ

中断处理例程：ISR

NMI —— 不可屏蔽

INTR —— 可屏蔽

中断向量表

Table 6-1. Protected-Mode Exceptions and Interrupts

Vector	Mne-monic	Description	Type	Error Code	Source
0	#DE	Divide Error	Fault	No	DIV and IDIV instructions.
1	#DB	Debug Exception	Fault/ Trap	No	Instruction, data, and I/O breakpoints; single-step; and others.
2	—	NMI Interrupt	Interrupt	No	Nonmaskable external interrupt.
3	#BP	Breakpoint	Trap	No	INT3 instruction.
4	#OF	Overflow	Trap	No	INTO instruction.
5	#BR	BOUND Range Exceeded	Fault	No	BOUND instruction.
6	#UD	Invalid Opcode (Undefined Opcode)	Fault	No	UD instruction or reserved opcode.
7	#NM	Device Not Available (No Math Coprocessor)	Fault	No	Floating-point or WAIT/FWAIT instruction.
8	#DF	Double Fault	Abort	Yes (zero)	Any instruction that can generate an exception, an NMI, or an INTR.
9		Coprocessor Segment Overrun (reserved)	Fault	No	Floating-point instruction.[1]
10	#TS	Invalid TSS	Fault	Yes	Task switch or TSS access.
11	#NP	Segment Not Present	Fault	Yes	Loading segment registers or accessing system segments.
12	#SS	Stack-Segment Fault	Fault	Yes	Stack operations and SS register loads.
13	#GP	General Protection	Fault	Yes	Any memory reference and other protection checks.
14	#PF	Page Fault	Fault	Yes	Any memory reference.

Table 6-1. Protected-Mode Exceptions and Interrupts (Contd.)

15	—	(Intel reserved. Do not use.)		No	
16	#MF	x87 FPU Floating-Point Error (Math Fault)	Fault	No	x87 FPU floating-point or WAIT/FWAIT instruction.
17	#AC	Alignment Check	Fault	Yes (Zero)	Any data reference in memory.[2]
18	#MC	Machine Check	Abort	No	Error codes (if any) and source are model dependent.[3]
19	#XM	SIMD Floating-Point Exception	Fault	No	SSE/SSE2/SSE3 floating-point instructions[4]
20	#VE	Virtualization Exception	Fault	No	EPT violations[5]
21	#CP	Control Protection Exception	Fault	Yes	RET, IRET, RSTORSSP, and SETSSBSY instructions can generate this exception. When CET indirect branch tracking is enabled, this exception can be generated due to a missing ENDBRANCH instruction at target of an indirect call or jump.
22-31	—	Intel reserved. Do not use.			
32-255	—	User Defined (Non-reserved) Interrupts	Interrupt		External interrupt or INT *n* instruction.

32号之后由操作系统决定

IDTR中断描述符寄存器

IDTR寄存器（同GDTR一样）存储着IDT，IDT每项8字节，一般一共有256项

注意：每个核心上都有IDTR，也就都有IDT，hook IDT的时候要全部核心都要hook

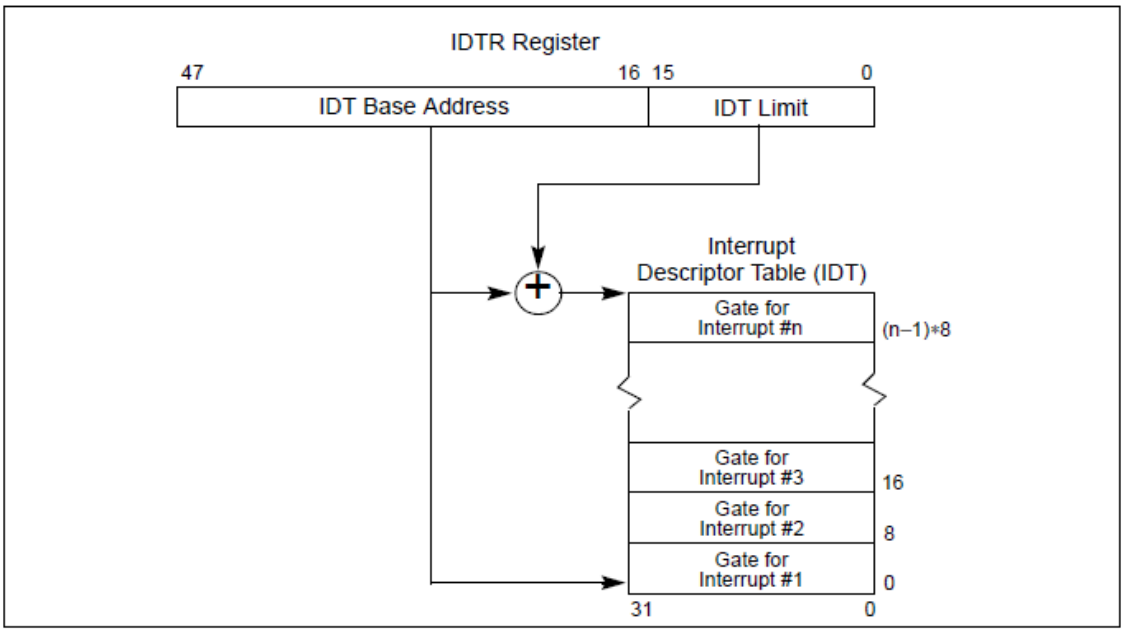

Figure 6-1. Relationship of the IDTR and IDT

中断描述符（门描述符，各种门具备R0权限）：

- 任务门（Task Gate）
- 中断门（Interrupt Gate）
- 陷阱门（Trap Gate）

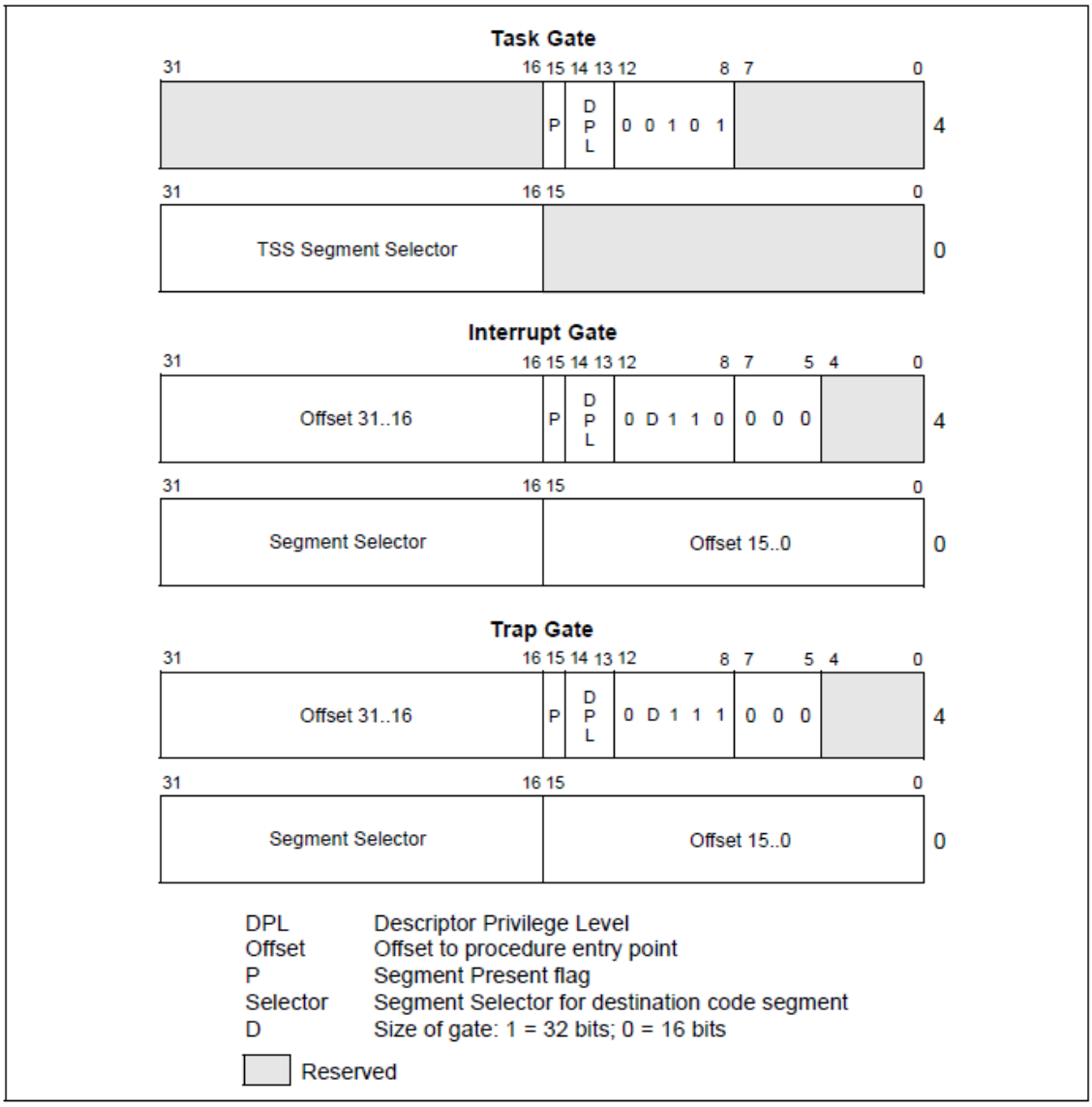

Figure 6-2. IDT Gate Descriptors

相关指令

- `sidt m`
  - 将中断描述符表寄存器 (IDTR) 中的内容存储到目标操作数
- `lidt m16&32`
  - 将源操作数中的值加载到中断描述符表寄存器 (IDTR)
  - 特权指令
- `iret`
  - 中断返回

中断请求级别

每个线程都运行在IRQL中断请求级别（0 ~ 31）上，高优先级可以打断低优先级

其中，0 ~ 2级给程序，3 ~ 31级给硬件

```
#define PASSIVE_LEVEL   0
// 最低级别，没有被屏蔽的中断，在这个级别上，线程执行用户模式，可以访问分页内存

#define APC_LEVEL       1
// 在这个级别上，只有APC级别的中断被屏蔽，可以访问分页内存

#define DISPATCH_LEVEL  2
// 这个级别，DPC和更低的中断被屏蔽，不能访问分页内存，所有的被访问的内存不能分页。
```

注意：

1. 有些内核API需要符合响应的级别，不然就蓝屏
2. 没有特殊情况不要提级别

`KeGetCurrentIrql`获取当前的中断请求级别，`KeRaiseIrql`提升级别，`KeLowerIrql`降低级别

关于裸函数

`__declspec(naked)`将函数定义为裸函数

当函数为裸函数时，编译器不会对函数添加额外的汇编代码，故在函数内需要自己保存和恢复环境，主要用在hook函数上

hook函数：

- 当hool函数为普通的函数时，如果在函数结束前`jmp`到了被hook的函数上，栈就会被破坏
- 当hook函数为裸函数时，需要手动内联汇编去保存恢复环境，就不存在`jmp`时，栈被破坏的问题

任务切换

任务状态段

TSS（Task-State Segment）任务状态段，存在于GDT和IDT中，用于保存任务切换时的运行环境

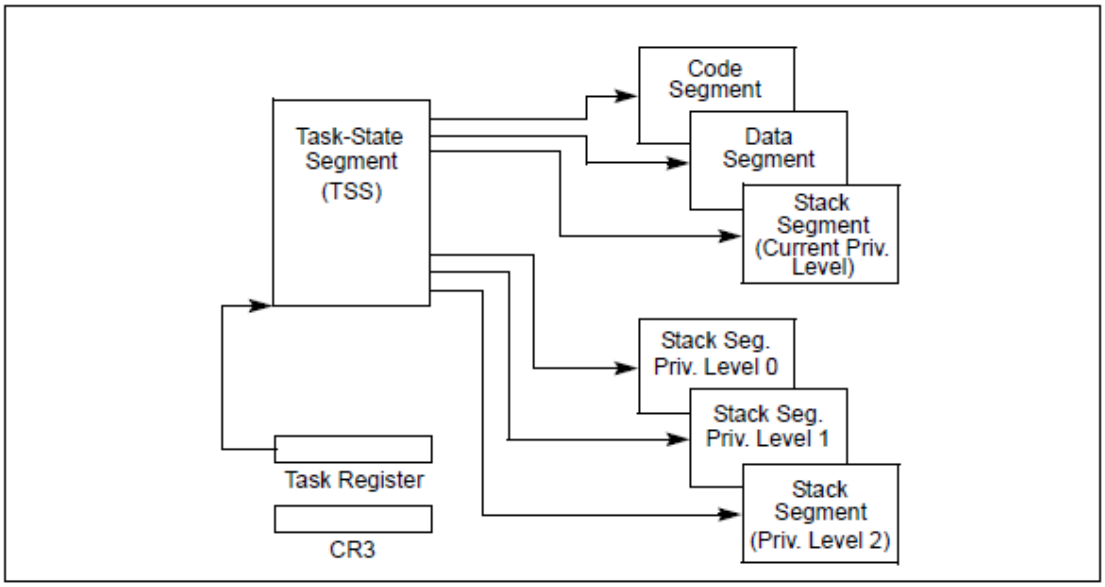

Figure 7-1. Structure of a Task

任务段描述符

任务段描述符同其他描述符结构一样，具体如下图

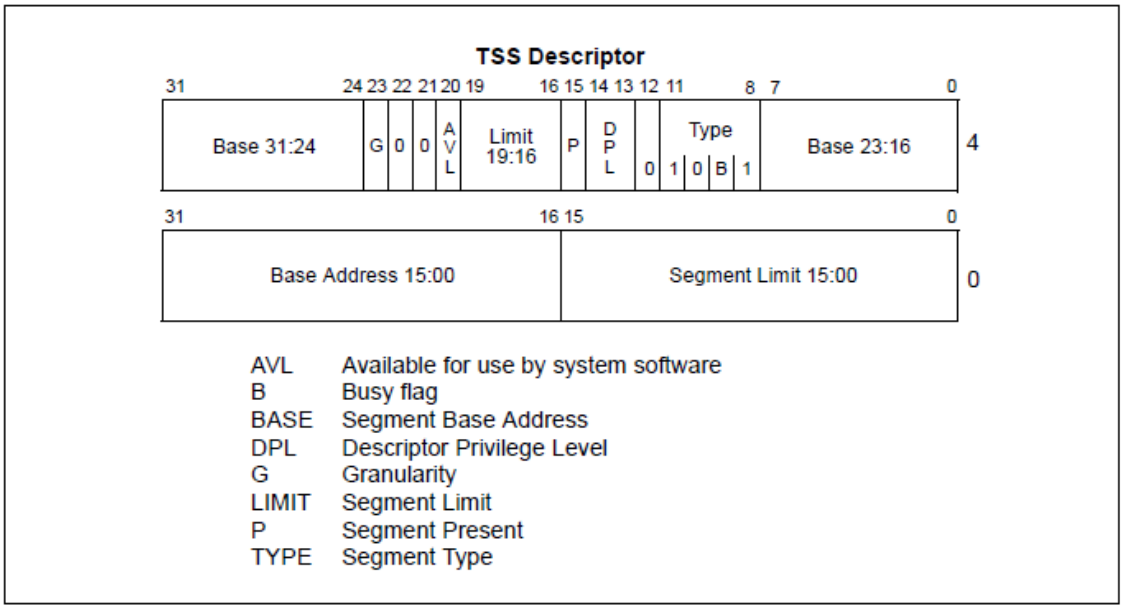

Figure 7-3. TSS Descriptor

任务寄存器

任务寄存器包含16位段选择子和整个段描述符（32位的base和16位的limit）

其中选择子为tr寄存器，发生任务切换：`mov tr, xxx`

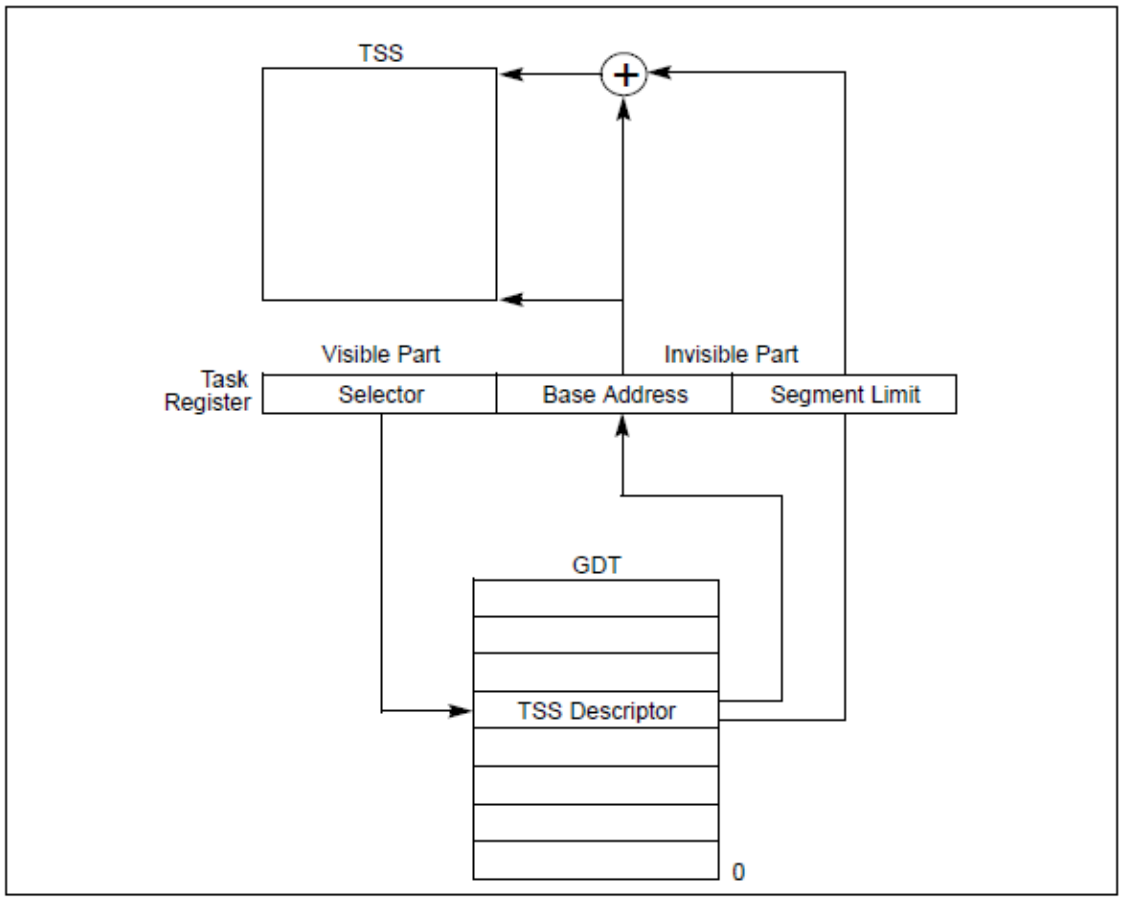

Figure 7-5. Task Register