中国科学院大学
University of Chinese Academy of Sciences

讲堂沙龙系列活动第一期
操作系统答疑讲堂

陶坤

2020年11月15日

主办：中国科学院大学网络空间安全学院

中国科学院大学 University of Chinese Academy of Sciences | 网络空间安全学院 School of Cyber Security

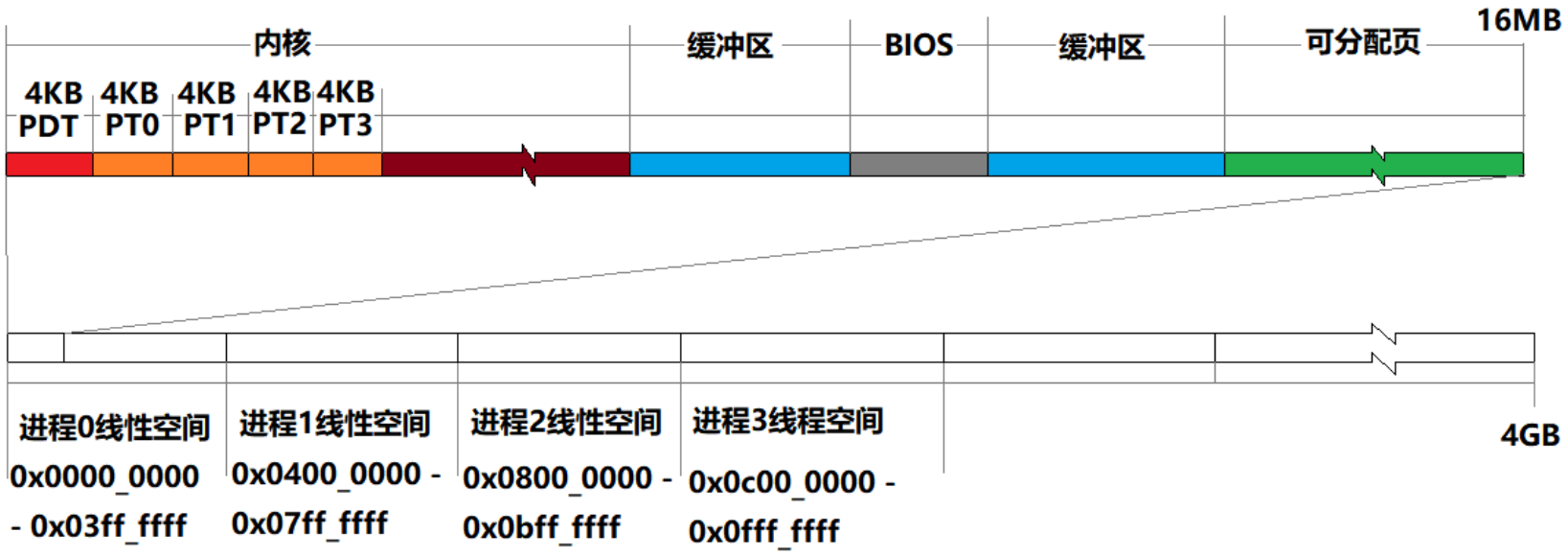
操作系统答疑讲堂

信息工程研究所 陶坤

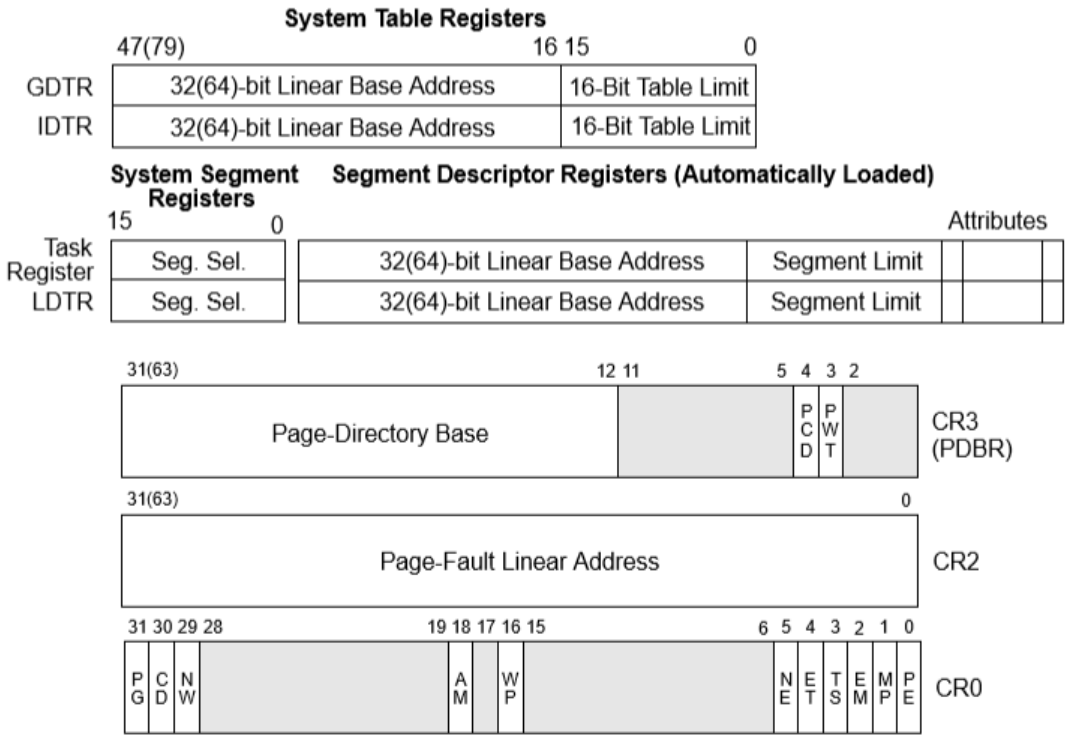
目录

- 内存布局
- 特殊寄存器
- 段机制
- 门机制
- 任务管理机制
- 页机制
- 系统初始化

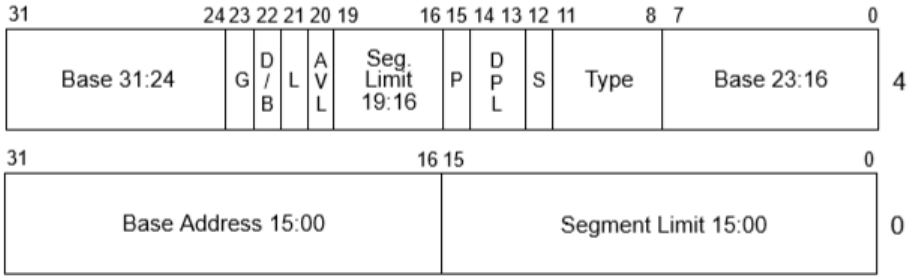
内存布局


内核
缓冲区
BIOS
缓冲区
可分配页
16MB
4KB PDT
4KB PT0
4KB PT1
4KB PT2
4KB PT3
进程0线性空间
0x0000_0000 - 0x03ff_ffff
进程1线性空间
0x0400_0000 - 0x07ff_ffff
进程2线性空间
0x0800_0000 - 0x0bff_ffff
进程3线程空间
0x0c00_0000 - 0x0fff_ffff
4GB

特殊寄存器

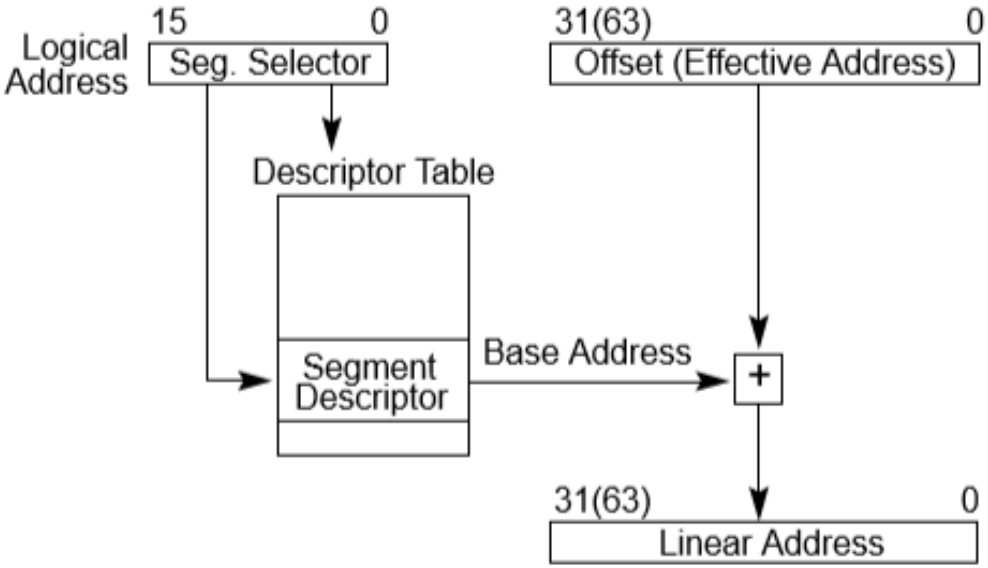

System Table Registers
47(79)
16 15
0
GDTR
32(64)-bit Linear Base Address
16-Bit Table Limit
IDTR
32(64)-bit Linear Base Address
16-Bit Table Limit
System Segment Registers
Segment Descriptor Registers (Automatically Loaded)
15
0
Attributes
Task Register
Seg. Sel.
32(64)-bit Linear Base Address
Segment Limit
LDTR
Seg. Sel.
32(64)-bit Linear Base Address
Segment Limit
31(63)
12 11
5 4 3 2
Page-Directory Base
PCD
PWT
CR3 (PDBR)
31(63)
0
Page-Fault Linear Address
CR2
31 30 29 28
19 18 17 16 15
6 5 4 3 2 1 0
PG
CD
NW
AM
WP
NE
ET
TS
EM
MP
PE
CR0

段机制



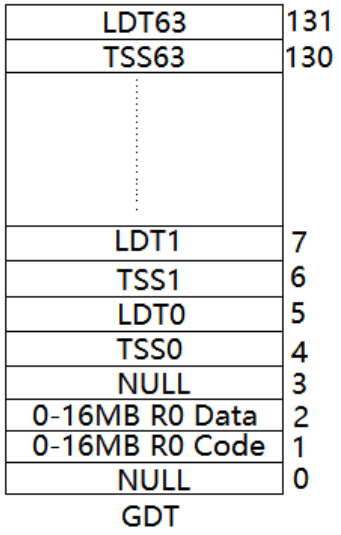
L — 64-bit code segment (IA-32e mode only)
AVL — Available for use by system software
BASE — Segment base address
D/B — Default operation size (0 = 16-bit segment; 1 = 32-bit segment)
DPL — Descriptor privilege level
G — Granularity
LIMIT — Segment Limit
P — Segment present
S — Descriptor type (0 = system; 1 = code or data)
TYPE — Segment type

段机制


Logical Address
15
0
Seg. Selector
31(63)
0
Offset (Effective Address)
Descriptor Table
Segment Descriptor
Base Address
+
31(63)
0
Linear Address

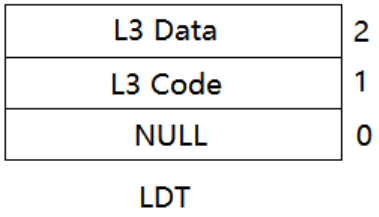
段机制

- GDT0为NULL，这是处理器的要求。
- GDT1为内核代码段
- GDT2为内核数据段
- GDT3为NULL
- GDT4为TSS0描述符
- GDT5为LDT0描述符

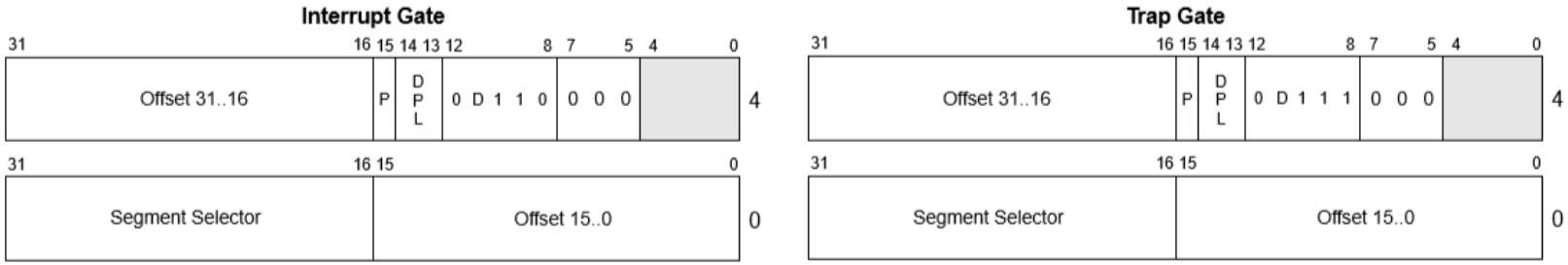


段机制

- 系统中的所有LDT都具有相同的形式
- LDT0为NULL
- LDT1为用户代码段
- LDT2为用户数据段

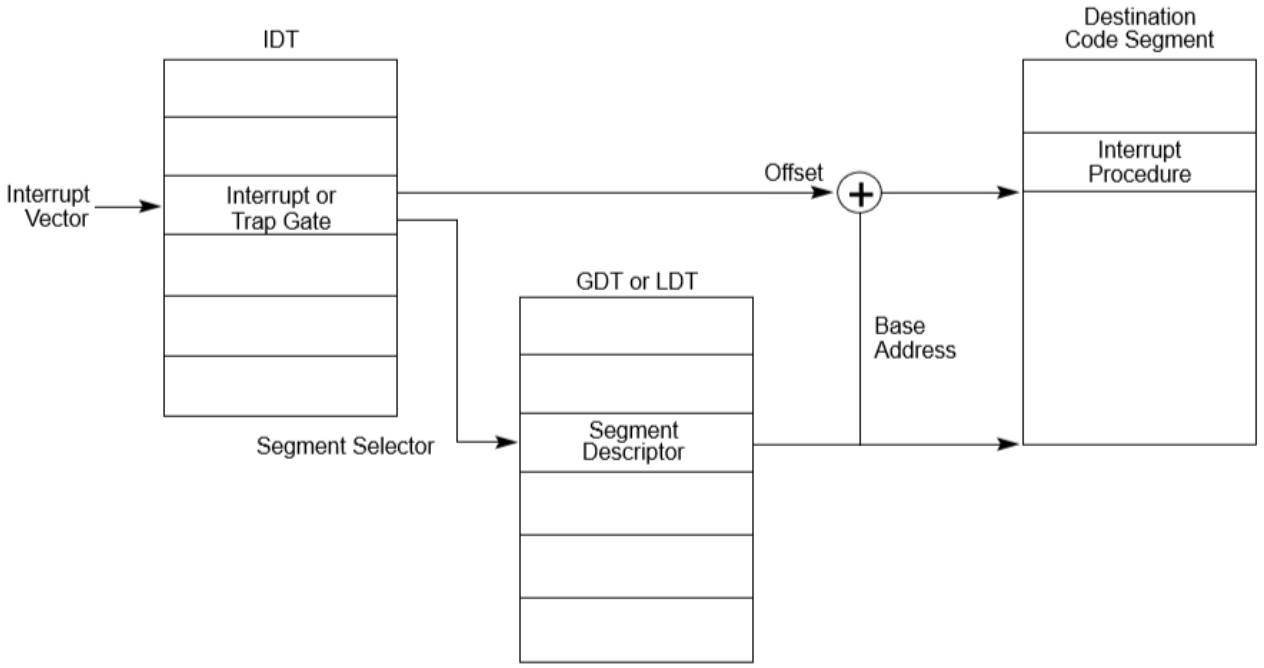


门机制



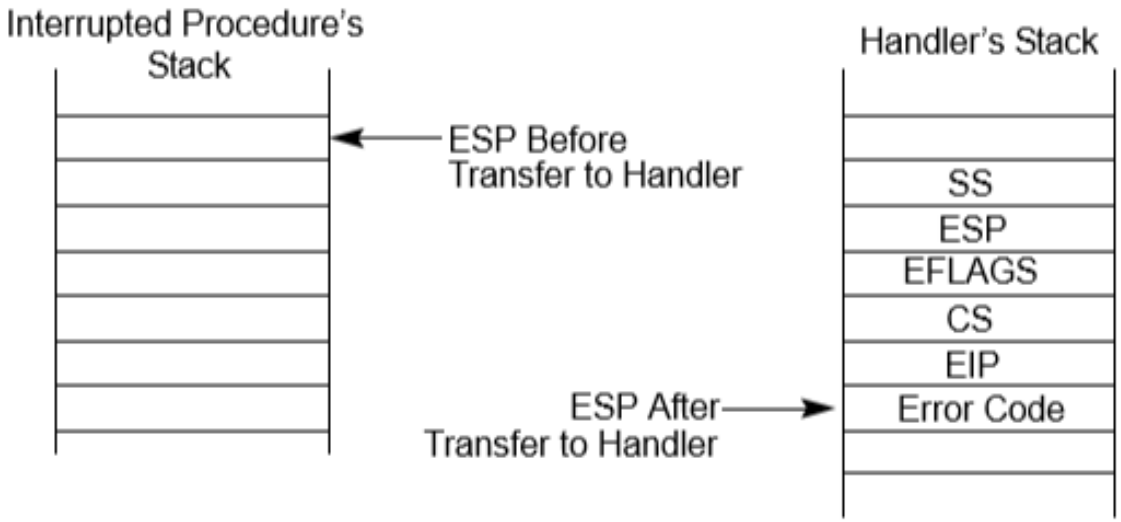
DPL	Descriptor Privilege Level
Offset	Offset to procedure entry point
P	Segment Present flag
Selector	Segment Selector for destination code segment
D	Size of gate: 1 = 32 bits; 0 = 16 bits

门机制


IDT
Interrupt
Vector
Interrupt or
Trap Gate
Segment Selector
GDT or LDT
Segment
Descriptor
Offset
Base
Address
Destination
Code Segment
Interrupt
Procedure

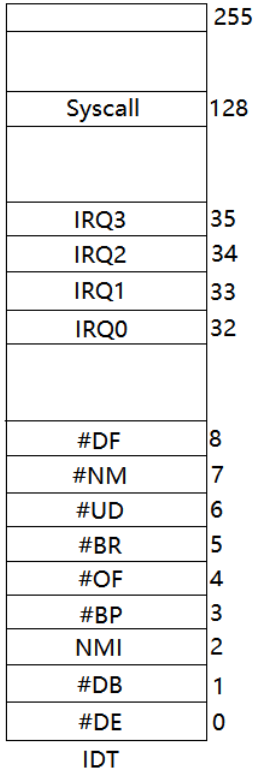
门机制

- 堆栈变化


Interrupted Procedure's Stack
Handler's Stack
ESP Before Transfer to Handler
SS
ESP
EFLAGS
CS
EIP
Error Code
ESP After Transfer to Handler

门机制

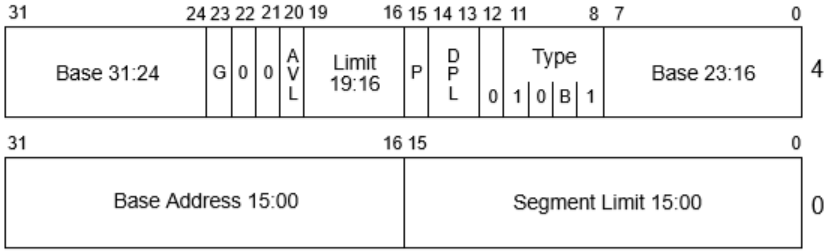
- IDT中所有的描述符必须时门描述符。
- IDT0 – IDT31是处理器保留的，不可以给软件使用，软件智能接收这个中断。
- IDT32 – IDT47用来接收PCI发送来的IRQ（setup.s中设置的）。
- IDT128也就是0x80用于系统调用。
- 其他的中断号不使用。



任务管理机制

31 — 15 — 0			
I/O Map Base Address	Reserved	T	100
Reserved	LDT Segment Selector		96
Reserved	GS		92
Reserved	FS		88
Reserved	DS		84
Reserved	SS		80
Reserved	CS		76
Reserved	ES		72
EDI			68
ESI			64
EBP			60
ESP			56
EBX			52
EDX			48
ECX			44
EAX			40
EFLAGS			36
EIP			32
CR3 (PDBR)			28
Reserved	SS2		24
ESP2			20
Reserved	SS1		16
ESP1			12
Reserved	SS0		8
ESP0			4
Reserved	Previous Task Link		0

TSS Descriptor



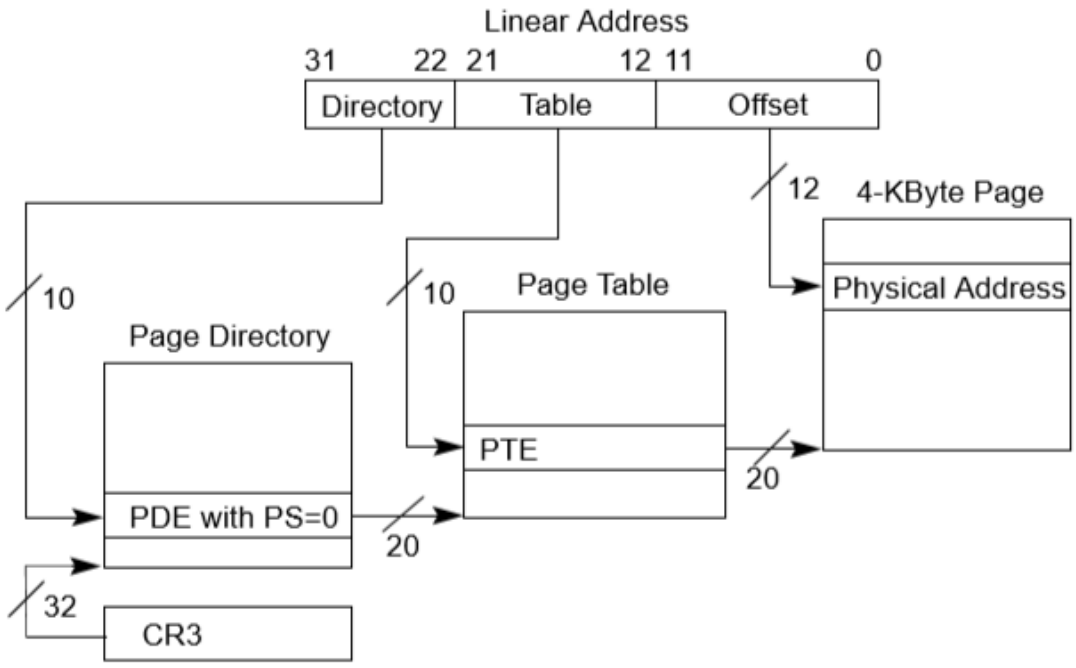
AVL	Available for use by system software
B	Busy flag
BASE	Segment Base Address
DPL	Descriptor Privilege Level
G	Granularity
LIMIT	Segment Limit
P	Segment Present
TYPE	Segment Type

任务管理机制

- 当跳转/调用/返回到一个任务段描述符的时候，将会发生任务切换。
- 任务切换时，根据方式不同，对目标的描述符中B位要求不同，对原任务描述符的B位影响也不一样，违反则发生#GP错误或#TS错误。

```
#define switch_to(n) {\
struct {long a,b;} __tmp; \
__asm__("cmpl %%ecx,_current\n\t" \
    "je 1f\n\t" \
    "movw %%dx,%1\n\t" \
    "xchgl %%ecx,_current\n\t" \
    "ljmp %0\n\t" \
    "cmpl %%ecx,_last_task_used_math\n\t" \
    "jne 1f\n\t" \
    "clts\n" \
    "1:" \
    ::"m" (*&__tmp.a),"m" (*&__tmp.b), \
    "d" (_TSS(n)),"c" ((long) task[n])); \
}
```

页机制


Linear Address
31
22
21
12
11
0
Directory
Table
Offset
12
4-KByte Page
Physical Address
10
10
Page Table
Page Directory
PTE
20
PDE with PS=0
20
32
CR3

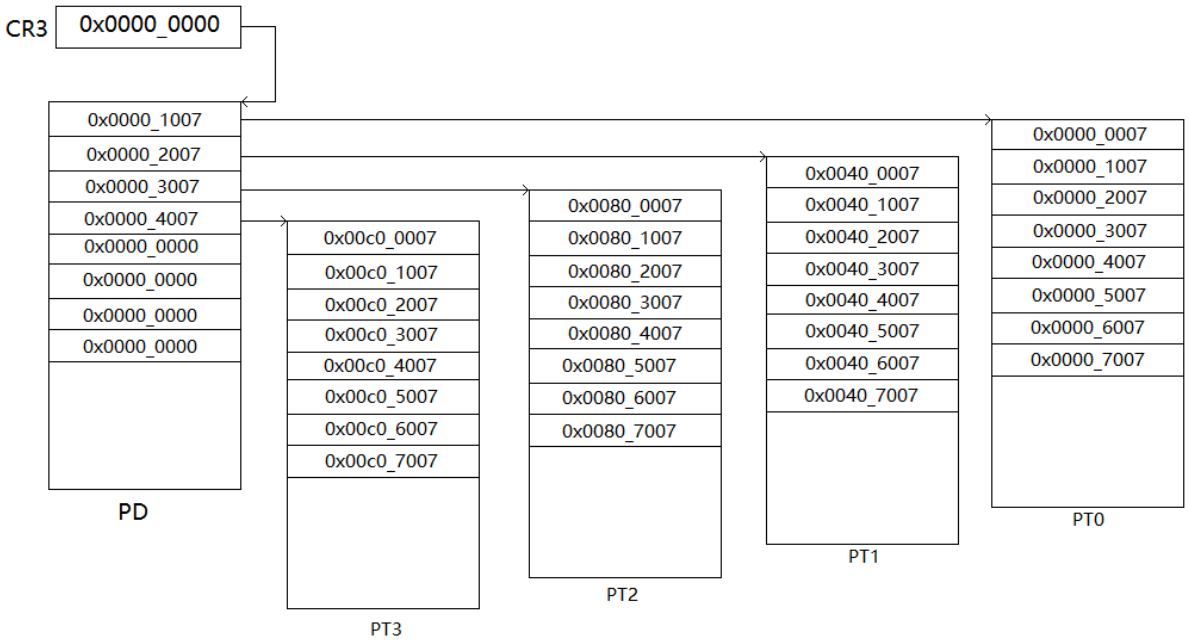
页机制

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Address of page directory[1]																				Ignored							PCD	PWT	Ignored			CR3
Address of page table																				Ignored				0	Ign	A	PCD	PWT	U/S	R/W	1	PDE: page table
Ignored																															0	PDE: not present
Address of 4KB page frame																				Ignored			G	PAT	D	A	PCD	PWT	U/S	R/W	1	PTE: 4KB page
Ignored																															0	PTE: not present

Figure 4-4. Formats of CR3 and Paging-Structure Entries with 32-Bit Paging

页机制

- 系统中仅使用一个页表，即所有的进程分布再一个4GB的线性空间内。



系统初始化

- 处理器初始化

寄存器	值	寄存器	值
EAX,ECX,EBX,ESP, EBP,ESI,EDI	0x00000000	DS,ES,FS,GS,SS	Selector=0x0000 Base=0x00000000 Limit=0xFFFF
EDX	0x00000Fxx	CS	Selector=0xF000 Base=0xFFFF0000 Limit=0xFFFF
EFLAGS	0x00000002	EIP	0x0000FFF0
CR0	0x60000010	CR2,CR3	0x0000000
GDTR,IDTR	Base=0x00000000 Limit=0xFFFF	LDTR,TR	Selector=0x0000 Base=0x00000000 Limit=0xFFFF

系统初始化

- POST
  - 清空内存
  - 设置中断向量表
  - 设置键盘
  - 设置显示器
  - 设置串口
  - 设置PIC
  - 设置硬盘
  - 设置并口
  - 设置软盘
  - 设置CMOS
  - 操作系统引导

系统初始化

- BOOT(bootsect.s)
  - 加载Setup
  - 加载内核映像

系统初始化

- setup.s
  - 获取内存大小
  - 获取显示器参数
  - 获取硬盘参数
  - 关中断
  - 展开内核映像
  - 加载IDTR
  - 加载GDTR
  - 打开A20
  - 设置PIC
  - 设置CR0.PE

系统初始化

- head.s
  - 设置段寄存器
  - 设置栈
  - 重新设置IDT
  - 重新设置GDT
  - 重新设置段寄存器
  - 重新设置栈
  - 测试A20
  - 检查x87协处理器
  - 设置页表
  - 打开分页

系统初始化

- main.c::main()
  - 初始化内存管理
  - 设置处理器保留中断
  - 初始化字符设备
  - 初始化块设备
  - 初始化TTY
  - 初始化系统时间
  - 初始化处理器调度
  - 初始化缓冲区
  - 初始化硬盘

系统初始化

- main.c::main()
  - 初始化软盘
  - 开中断
  - move_to_user_mode
  - 创建新进程
  - 进程0待机，进程1继续完成初始化

系统初始化

- main.c::init()
  - 系统设置
  - 打开STDIN
  - 打开STDOUT
  - 打开STDERR
  - 创建新进程
  - 进程2执行shell
  - 进程1等待进程2结束