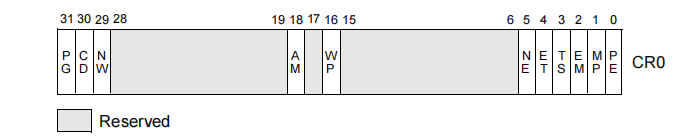
### PTE HOOK

复制自己进程所有PDE,PTE,物理页。在自己的进程里hook R0函数，其他进程不受影响，可以用来做驱动通信，过PG。

### CR0



PE=1 开启段

PG=1 开启页

有段模式，段页模式，实模式。不能只开页不开段。

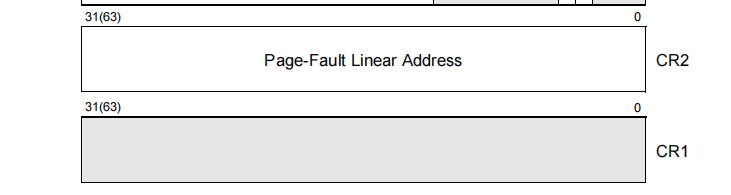
TS(task switched)位，任务切换时置位。

CD位，缓存机制总开关。

WP位，写保护，WP置0，可以写R/W=0的页。不建议商用，和VT有冲突。

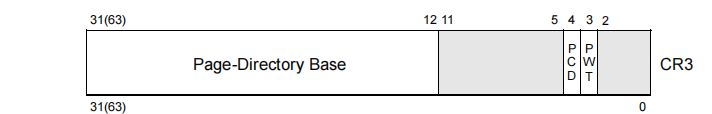
AM位，X86下是强制栈对齐开关；X64下还有其他用途，待会说。

### CR2



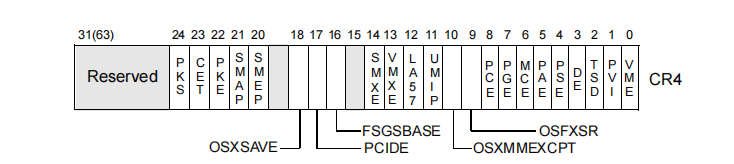
CR2在页异常发生时，保存异常的内存地址。

### CR3



101012下指向第一个页目录表项，29912下指向第一个页目录指针表项，PCD,PWT是缓存相关的位。

### CR4



CR4.VME 开启虚拟8086

CR4.PVI 虚拟8086模式下的模拟中断

TSD置1 R3可以调用rdtsc

DE位

DR0-DR7硬件调试

DR4-DR5无法访问

DE=1，DR4==DR6, DR5==DR7

DE=0，访问这俩会异常

PSE=1，大页有效；否则无效

PAE=1，29912； 否则101012

MCE 开启机器检查中断0x12

PGE=1 页的G位有效，否则无效

置0再置1就可以刷TLB的G页

PCE 监控事件开关

PCIDE x64 用，每个进程的CR3是否缓存

FSGSBASE = 1时：

fs.base可以不通过段解析去拆分获得

rdmsr(IA32E\_FS\_BASE)直接读base

获取FS是为了拿KPCR，有其他方式

SMEP SMAP

super mode execute/access

置位，R0无法访问R3

直接改会PG

CR0.AM=1，SME才生效

所以CR0.AM=1，就可以忽略SME

R0访问R3了

CR4.CET 火哥也没用过

CR4.PKS 没卵用，给页表加密用

置1后 CR3的值是加密值，只在X64生效