第二讲 IA32 寄存器简介

IA32 作为 80x86 系列中的一员，必须保证向后兼容，也就是说，既要支持 16 位的处理器，也要支持 32 位的处理器。在 8086 中，所有的寄存器都是 16 位的，下面我们来看一下 IA32 中寄存器有何变化:

- 把 16 位的通用寄存器、标志寄存器以及指令指针寄存器扩充为 32 位的寄存器
- 段寄存器仍然为 16 位。
- 增加 4 个 32 位的控制寄存器
- 增加 4 个系统地址寄存器
- 增加 8 个调式寄存器
- 增加 2 个测试寄存器

一、通用寄存器

8 个通用寄存器是 8086 寄存器的超集，它们的名称和用途分别为:

EAX　一般用作累加器

EBX　一般用作基址寄存器（Base）

ECX　一般用来计数（Count）

EDX　一般用来存放数据（Data）

EBP　一般用作堆栈指针（Stack Pointer）

EBP　一般用作基址指针（Base Pointer）

ESI　 一般用作源变址（Source Index）

EDI　 一般用作目标变址（Destinatin Index）

8 个通用寄存器中通常保存 32 位数据，但为了进行 16 位的操作并与 16 为机保持兼容，它们的低位部分被当成 8 个 16 位的寄存器，即 AX、BX…DI。为了支持 8 位的操作，还进一步把 EAX、EBX、ECX、EDX 这四个寄存器低位部分的 16 位，再分为 8 位一组的高位字节和低位字节两部分，作为 8 个 8 位寄存器。这 8 个寄存器分别被命名为 AH、BH、CH、DH 和 AL、BL、CL、DL。对 8 位或 16 位寄存器的操作只影响相应的寄存器。例如，在做 8 位加法运算时，位 7 的进位并不传给目的寄存器的位 9，而是把标志寄存器中的进位标志（CF）置位。因此，这 8 个通用寄存器既可以支持 1 位、8 位、16 位和 32 位数据运算，也支持 16 位和 32 位存储器寻址。

二、段寄存器

8086 中有 4 个 16 位的段寄存器：CS、DS、SS、ES，分别用于存放可执行代码的代码段、数据段、堆栈段和其他段的基地址。在 IA32 中，有 6 个 16 位的段寄存器，但是，这些段寄存器中存放的不再是某个段的基地址，而是某个段的选择符（Selector）。因为 16 位的寄存器无法存放 32 位的段基地址，段基地址只好存放在一个叫做描述符表（Descriptor）的表中。因此，在 IA32 中，我们把段寄存器叫做选择符。下面给出 6 个段寄存器的名称和用途：

CS　代码段寄存器

DS　数据段寄存器

SS　堆栈段寄存器

ES、FS 及 GS　附加数据段寄存器

有关段选择符、描述符表及系统表地址寄存器将在段机制一节进行详细描述。

三、状态和控制寄存器

状态和控制寄存器是由标志寄存器 EFLAGS、指令指针 EIP 和 4 个控制寄存器组成，如图 2.1 所示：

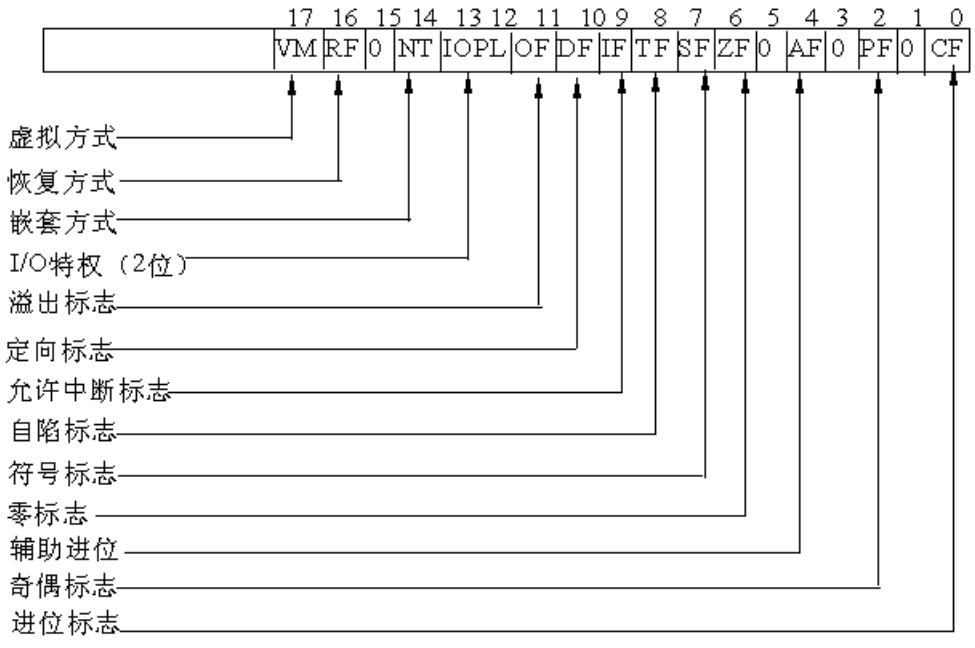
EFLAGS
EIP
CR0
CR1
CR2
CR3

图 2.1 状态和控制寄存器

1.指令指针寄存器和标志寄存器

指令指针寄存器 EIP 中存放下一条将要执行指令的偏移量（offset），这个偏移量是相对于目前正在运行的代码段寄存器 CS 而言的。偏移量加上当前代码段的基地址，就形成了下一条指令的地址。EIP 中的低 16 位可以分开来进行访问，给它起名叫指令指针 IP 寄存器，用于 16 位寻址。

标志寄存器 EFLAGS 存放有关处理器的控制标志，如图 2.2 所示。标志寄存器中的第



1、3、5、15 位及 18～31 位都没有定义。

图 2.2 i386 标志寄存器 EFLAGS

在这些标志位中，我们只介绍在 Linux 内核代码中常用且重要的几个标志位：

第 8 位 TF（Trap Flag）是自陷标志，当将其置 1 时则可以进行单步执行。当指令执行完后，就可能产生异常 1 的自陷（参看第四章）。也就是说，在程序的执行过程中，每执行完一条指令，都要由异常 1 处理程序 （在 Linux 内核中叫做 debug()）进行检验。当将第 8 位清 0 后，且将断点地址装入调试寄存器 DR0～DR3 时，才会产生异常 1 的自陷。

第 12、13 位 IOPL 是输入输出特权级位，这是保护模式下要使用的两个标志位。由于输入输出特权级标志共两位，它的取值范围只可能是 0、1、2 和 3 共 4 个值，恰好与输入输出特权级 0～3 级相对应。但 Linux 内核只使用了两个级别，即 0 和 3 级，0 表示内核级，3 表示用户级。在当前任务的特权级 CPL（Current Privilege Level）高于或等于输入输出特权级时，就可以执行像 IN、OUT、INS、OUTS、STI、CLI 和 LOCK 等指令而不会产生异常 13（即保护异常）。在当前任务特权级 CPL 为 0 时，POPF（从栈中弹出至标志位）指令和中断返回指令 IRET 可以改变 IOPL 字段的值。

第 9 位 IF（Interrupt Flag）是中断标志位，是用来表示允许或者禁止外部中断（参看第四章）。若第 9 位 IF 被置为 1，则允许 CPU 接收外部中断请求信号；若将 IF 位清 0，则表示禁止外部中断。在保护模式下，只有当第 12、13 位指出当前 CPL 为最高特权级时，才允许将新值置入标志寄存器 EFLAGS 以改变 IF 位的值。

第 10 位 DF（Direction Flag）是定向标志。DF 位规定了在执行串操作的过程中，对源变址寄存器 ESI 或目标变址寄存器 EDI 是增值还是减值。如果 DF 为 1，则寄存器减值；若 DF 为 0，则寄存器值增加。

第14位NT是嵌套任务标志位。在保护模式下常使用这个标志。当IA32在发生中断和执行CALL指令时就有可能引起任务切换。若是由于中断或由于执行CALL指令而出现了任务切换，则将NT置为1。若没有任务切换，则将NT位清0。

第17位VM （Virtual 8086 Mode Flag）是虚拟8086方式标志，是IA32新设置的一个标志位。表示IA32CPU是在虚拟8086环境中运行。如果IA32CPU是在保护模式下运行，而VM为又被置成1，这时IA32就转换成虚拟8086操作方式，使全部段操作就像是在8086CPU上运行一样。VM位只能由两种方式中的一种方式给予设置，即或者是在保护模式下，由最高特权级（0）级代码段的中断返回指令IRET设置，或者是由任务转换进行设置。Linux内核实现了虚拟8086方式，但在本书中我们不准备对此进行详细讨论。

从上面的介绍可以看出，要正确理解标志寄存器EFLAGS的各个标志需要很多相关的知识，有些内容在本章的后续部分还会涉及到。在后面的章节中，你会体会如何灵活应用这些标志。

2．控制寄存器

状态和控制寄存器组除了EFLAGS、EIP ，还有四个32位的控制寄存器，它们是CR0，CR1，CR2和CR3。现在我们详细看看它们的结构，如图 2－3所示。

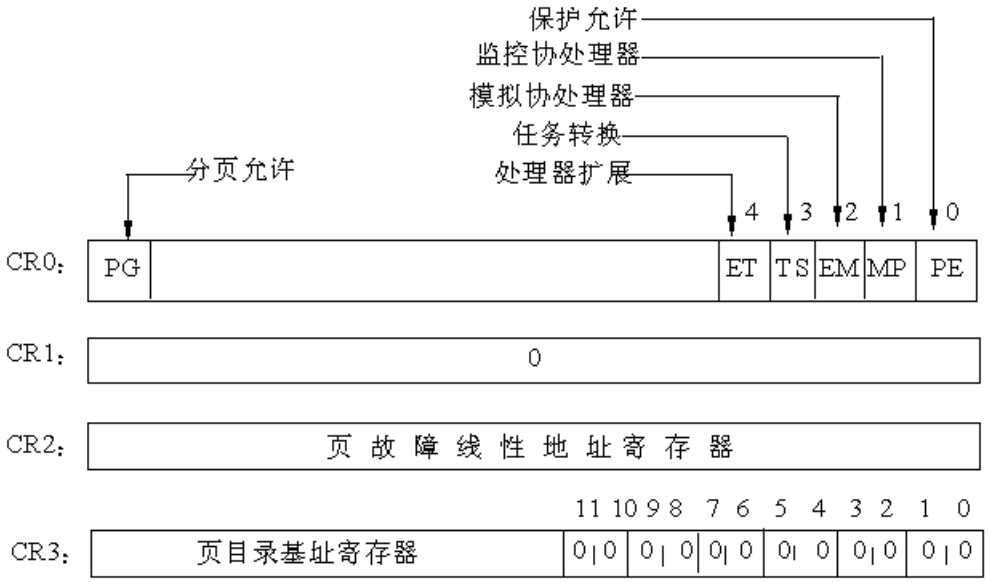


图 2－3 386中的控制寄存器组

这几个寄存器中保存全局性和任务无关的机器状态。

CR0中包含了6个预定义标志，0位是保护允许位PE(Protedted Enable)，用于启动保护模式，如果PE位置1，则保护模式启动，如果PE=0，则在实模式下运行。1位是监控协处理位MP(Moniter coprocessor)，它与第3位一起决定：当TS=1时操作码WAIT是否产生一个“协处理器不能使用”的出错信号。第3位是任务转换位(Task Switch)，当一个任务转换完成之后，自动将它置1。随着TS=1，就不能使用协处理器。CR0的第2位是模拟协处理器位EM (Emulate coprocessor)，如果EM=1，则不能使用协处理器，如果EM=0，则允许使用协处理器。第4位是微处理器的扩展类型位ET(Processor Extension Type)，其内保存着处理器扩展类型的信息，如果ET=0，则标识系统使用的是287协处理器，如果 ET=1，则表示系统使用的是387浮点协处理器。CR0的第31位是分页允许位(Paging Enable)，它表示芯片上的分页部件是否允许工作，下一节就会讲到。由PG位和PE位定义的操作方式如图2.4 所示。

PG	PE	方式
0	0	实模式，8080操作
0	1	保护模式，但不允许分页
1	0	出错
1	1	允许分页的保护模式

图 2－4　PG位和PE位定义的操作方式

CR1是未定义的控制寄存器，供将来的处理器使用。

CR2是页故障线性地址寄存器，保存最后一次出现页故障的全32位线性地址。

CR3是页目录基址寄存器，保存页目录表的物理地址，页目录表总是放在以4K字节为单位的存储器边界上，因此，它的地址的低12位总为0，不起作用，即使写上内容，也不会被理会。

这几个寄存器是与分页机制密切相关的，因此，在进程管理及虚拟内存管理中会涉及到这几个寄存器，读者要记住 CR0、CR2 及 CR3 这三个寄存器的内容。

四、系统地址寄存器

IA32 有 4 个系统地址寄存器，如图 2.5 所示，它保存操作系统要保护的信息和地址转换表信息:

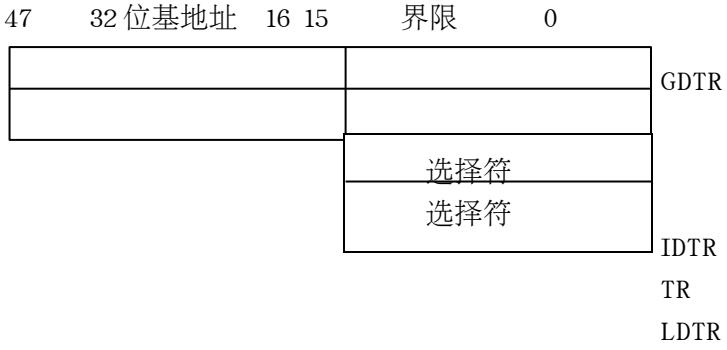


图 2.5　IA32 系统地址寄存器

这4个专用寄存器用于引用在保护模式下所需要的表和段，它们的名称和作用如下：

- 全局描述符表寄存器GDTR（Global Descriptor Table Register），是48位寄存器，用来保存全局描述符表（GDT）的32位基地址和16为GDT的界限。
- 中断描述符表寄存器IDTR（Interrupt Descriptor Table Register），是48位寄存器，用来保存中断描述符表（IDT）的32位基地址和16为IDT的界限。
- 局部描述符表寄存器LDTR（Global Descriptor Table Register），是16位寄存器，保存局部描述符表LDT段的选择符。
- 任务状态寄存器TR（Task State Register）是16位寄存器，用于保存任务状态段TSS段的16位选择符。

用以上4个寄存器给目前正在执行的任务（或进程）定义任务环境、地址空间和中断向量空间。有关全局描述符表GST、中断描述符表IDT、局部描述符表LDT及任务状态段TSS的具体内容将在稍后进行详细描述。

五、 调试寄存器和测试寄存器

1．调试寄存器

IA32为调试提供了硬件支撑。在IA32芯片内有8个32位的调试寄存器DR0~DR7，如图2.6所示。

这些寄存器可以使系统程序设计人员定义4个断点，用它们可以规定指令执行和数据读写的任何组合。DR0~DR3是线性断点地址寄存器，其中保存着4个断点地址。DR5、DR6是两个备用的调试寄存器，目前尚未定义。DR6是断点状态寄存器，其低序位是指示符位，当允许故障调试并检查出故障而进入异常调试处理程序（debug()）时，由硬件把指示符位置1，调试异常处理程序在退出之前必须把这几位清0。DR7是断点控制寄存器，它的高序半个字又被分为4个字段，用来规定断点字段的长度是1个字节、2个字节、4个字节及

规定将引起断点的访问类型。低序半个字的位字段用于“允许”断点和“允许”所选择的调试条件。

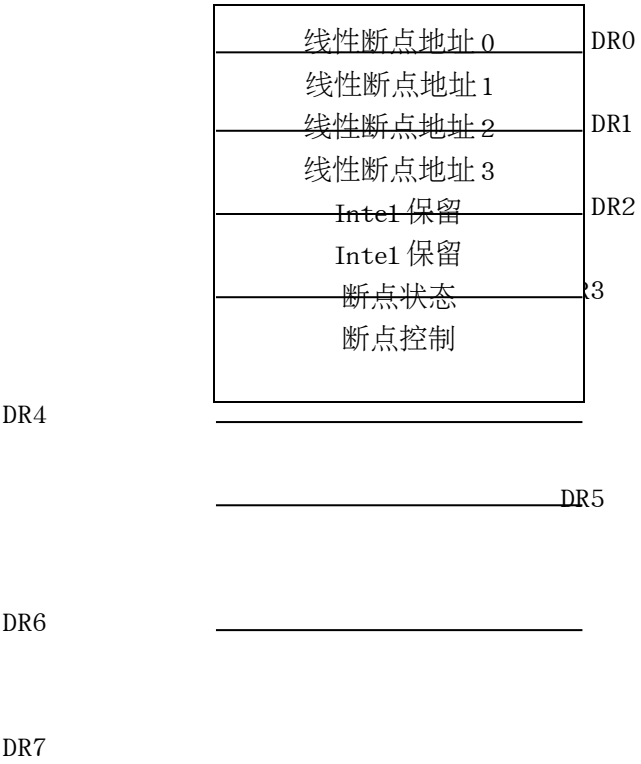


图 2.6 IA32 的调试寄存器

2. 测试寄存器

IA32 有两个 32 位的测试寄存器 TR6 和 TR7。这两个寄存器用于在转换旁视缓冲器（Translation Lookaside Buffer）中测试随机存储器（RAM）和相联存储器（CAM）。TR6 是测试命令寄存器，其内存放测试控制命令。TR7 是数据寄存器，其内保存转换旁路缓冲器测试的数据。