Hollow Man

一、 80x86 中的寄存器及作用

8086 CPU 中总共有 14 个 16 位寄存器，即 AX，BX，CX，DX，SP，BP，SI，DI，IP，FLAG，CS，DS，SS，ES 。这 14 个寄存器又分为了通用寄存器，控制寄存器和段寄存器。

通用寄存器：

AX：累加寄存器；

BX：基地址寄存器；

CX：计数器寄存器；

DX：数据寄存器；

以上 4 个 16 位寄存器可以当做 2 个独立的 8 位寄存器来使用，如 AX 的高 8 位为 AH，低八位为 AL，以此类推。

SP ：堆栈指针寄存器；

BP：基指针寄存器；

SI：源变址寄存器；

DI：目的变址寄存器；

控制寄存器：

IP：指令指针寄存器；

FLAG：标志寄存器；

段寄存器：

CS：代码段寄存器；

DS：数据段寄存器；

SS：堆栈段寄存器；

ES：附加段寄存器；

除了以上字面含义代表的功能，一下寄存器还有如下功能：

1. AX 和 DX:

在做乘除法时，有两种情况，即除数可以是 8 位或者是 16 位的，当除数是 8 位时，被除数一定是 16 位的，并且默认放在 AX 寄存器中，而当除数是 16 位时，被除数一定是 32 位的，因为 AX 是 16 位寄存器，所以，在这里还需要使用另一个 16 位寄存器

DX，其中 DX 存放 32 位的被除数的高 16 位，而 AX 则存放 32 位的被除数的低 16 位， 同时，当除法指令执行完成以后，如果除数是 8 位的，则在 AL 中会保存此次除法操作的商，而在 AH 中则会保存此次除法操作的余数，当然，如果除数是 16 位的话，则 AX 中会保存本次除法操作的商，而 DX 则保存本次除法操作的余数。

在做乘法运算时，两个相乘的数要么都是 8 位，要么都是 16 位，如果两个相乘的数都是 8 位的话，则一个默认是放在 AL 中，而另一个 8 位的乘数则位于其他的寄存器或者说是内存字节单元中，而如果两个相乘的数都是 16 位的话，则一个默认存放在 AX 中，另一个 16 位的则是位于 16 的寄存器中或者是某个内存字单元中。同时，当 MUL 指令执行完毕后，如果是 8 位的乘法运算，则默认乘法运算的结果是保存在 AX 中，而如果是 16 位的乘法运算的话，则默认乘法运算的结果有 32 位，其中，高位默认保存在 DX 中，而低位则默认保存在 AX 中。

同时，AH 的值也决定 21 号中断执行时体现的功能

2. BX, SI, DI, BP, DS, ES:

BX 可以用于寻址，BX 寄存器中存放的数据一般是用来作为偏移地址，搭配 SI，DI，BP，与段地址寄存器 DS, ES, SS 搭配使用.

3. CX：

当在汇编指令中使用循环 LOOP 指令时，可以通过 CX 来指定需要循环的次数，

4. FLAG:

Flag 寄存器中依次有 OF，DF，IF，TF，SF，ZF，AF，PF，CF 标志位的值有意义，

CF - 进位标志（第 0 位）：是用来反映计算时是否产生了由低位向高位的进位，或者产生了从高位到低位的借位。

PF - 奇偶标志（第 2 位）：用来记录相关指令执行后，其结果的所有的 Bit 位中 1 的个数是否为偶数 。

AF - 辅助进位标志（第 4 位）：用来辅助进位标志 。

ZF –零标志（第 6 位）：记录的是相关的指令执行完毕后，其执行的结果是否为 0 。

SF - 符号标志（第 7 位）：其记录相关指令执行完以后，其结果是否为负数 。

TF - 追踪标志（第 8 位）：主要是用于调试时使用 。

IF - 中断允许标志（第 9 位）：决定 CPU 是否能够响应外部可屏蔽中断请求（以后会做详细介绍） 。

DF - 方向标志（第 10 位）：其用于在串处理指令中，用来控制每次操作后 SI 和 DI 是自增还是自减 。

OF - 溢出标志（第 11 位）：其通常记录了有符号数运算的结果是否发生了溢出 。

二、 80x86 中的中断机制

中断有两种，一种是 CPU 外部产生的，称为外部中断，一种是 CPU 内部执行程序时产生的，称为内部中断。

外部的中断通常是由外部设备产生的，并且这类中断的产生是异步的。

CPU 内部产生的中断也分为两种，一种是软件主动产生的中断，通常称为陷阱，例如执行 int 0x80 指令，还有一种是 CPU 检测到异常，通常称为异常，例如除数为 0。

当通过一条 INT 指令进入一个中断服务程序时，在指令中给出一个中断向量。CPU 先根据该向量在中断向量表中找到一扇门（描述项），在这种情况下一般总是中断门。然后将这个门的 DPL（描述符优先级）与 CPU 的 CPL（当前运行优先级）相比，CPL 必须小于或等于 DPL，也就是优先级别不低于 DPL，才能穿过这扇门。注意，这里的检查只是检查当前的程序是否有访问该中断门的权限，只有通过权限检查才能去获取段选择码等信息，从而找到相应的段描述符。不过，如果中断是由外部产生或是因 CPU 异常而产生的话，那就免去了这一层检验。穿过了中断门之后，还要进一步讲目标代码段描述符中的 DPL 与 CPL 比较，目标段的 DPL 必须小于或等于 CPL。也就是说，通过中断门时只允许保持或提升 CPU 的运行级别；而不允许降低其运行级别。这两个环节中的任何一个失败都会产生一次全面保护异常。

进入中断服务程序时，CPU 要将当前 EFLAGS 寄存器的内容以及返回地址压入堆栈，返回地址是由段寄存器 CS 的内容和取指令指针 EIP 的内容共同组成的。如果中断是由异常引起的，则还要讲一个表示异常的出错代码也压入堆栈。进一步，如果中断服务程序的运行级别，也就是目标代码段的 DPL，与中断发生时的 CPL 不同，那就要引起更换堆栈。CPU 会根据寄存器 TR 的内容找到当前 TSS 结构，并根据目标代码的 DPL，从这 TSS 结构中取出新的堆栈指针（SS 加 ESP），并装入其堆栈段寄存器 SS 和堆栈指针寄存器 ESP，达到更换堆栈的目的。在这种情况下，CPU 不但将 EFLAGS、返回地址以及出错代码压入堆栈，还要先将原来的堆栈指针也压入新的堆栈。