**[中断与异常 代码 总结](https://www.cnblogs.com/wanghj-dz/archive/2011/05/10/2042440.html)**

ICW (Initialization Command Word)初始化命令字。

主8259A对应的端口地址是20A和21A

从8259A对应的端口地址是A0h和A1h。

初始化过程：

1、往端口20h（主片）或A0h（从片）写入ICW1

2、往端口21h（主片）或A1h（从片）写入ICW2

3、往端口21h（主片）或A1h（从片）写入ICW3

4、往端口21h（主片）或A1h（从片）写入ICW4

这4步的顺序是不能颠倒的。

ICW1负责启动8259A和进行初始化工作

ICW2中断类型号的设置

ICW3主从片初始化设置

ICW4方式控制设置

--------------------------------------------------------------------------------

Init8259A:

     mov al,011h

     out  020h,al   ;主8259,ICW1

     call   io\_delay

     out  0A0h，al；从8259，ICW1

     call   io\_delay

 --------------------------------------------------------------------------------

   011h转换为2进制为0001，0001

     对应的ICW1   000 PC系统必须为0，1对ICW必须为1，0 edge triggered模式，0 8字节中断向量 0 联级8259 1 需要ICW4

 --------------------------------------------------------------------------------

    mov al,020h   ;IRQ0对应中断向量0X20

     out   021h,al   ;主8259，ICW2

     call io\_delay

     mov   al,028h  ;IRQ8对应中断向量0X28

     out    0A1h,al  ;从8259，ICW2

     call    io\_delay

   --------------------------------------------------------------------------------

  在往主8259A写入ICW2时，我们看到IRQ0（IRQ0在主8259A上）对应的了中断向量号20h，于是IRQ0---IRQ7就对应中断向量20h--27h。

     类似地IRQ8---IRQ15对应的中断向量28h---2Fh。对照3-11的表，我们知道20h--2Fh处于用户定义中断的范围内。

  --------------------------------------------------------------------------------

    mov   al,004h   ;IR2对应 从8259A

      out   021h，al  ;主8259，ICW3

      call  io\_delay

      mov   al,002h  ;对应主8259A的IR2

      out   0A1h ,al  ;从8259，ICW3

      call  io\_delay

--------------------------------------------------------------------------------

      004h转换为2进制为0000，0100对应的主片ICW3，是IR2级联从片为1，其余为0，表示IR2连着从片

      002h转换为2进制为0000，0010对应的从片ICW3，可以发现 从片连的主片的IR号为010，是2，所以从片连的是主片的IR2

--------------------------------------------------------------------------------

      mov   al,001h

      out    021h,al  ;主8259，ICW4

      call     io\_delay

      out  0A1h，al；从8259，ICW4

      call  io\_delay

--------------------------------------------------------------------------------

      001h转换为2进制为0000，0001  最后的1表示是80X86模式

      ICW1-4分别放入主从8259A完毕。

      下面写的就是OCW1，OCW1是控制主从8259A中断屏蔽的。

      ICW，OCW是根据写的顺序系统自动填充的。

--------------------------------------------------------------------------------

      mov   al，11111111b ；屏蔽主8259所有中断

      out    021h，al          ；主8259，OCW1

      call    io\_delay

      mov   al,11111111b;  屏蔽从8259所有中断

      out    0A1h，al       ；从8259，OCW1

      call    io\_delay

      ret

 ---------------------------------------------------------------------------------

      如果再往下写，系统就是填充OCW2，OCW2是控制EOI发送的，以通知8259A中断处理结束。

      mov   al，20h

      out    20h或A0h，al

      20h 转换为2进制为0010，0000，其对应的OCW2，正好EOI是1。

      延迟函数的功能是等待OUT操作的完成。

     在相应的位置添加调用Init8259A的指令后，对8259A的操作就结束了。

 下面我们来建立一个IDT

      为了操作方便，我们把IDT放进一个单独的段中：

--------------------------------------------------------------------------------

      [SECTION .idt]

      LABEL\_IDT:

      %rep 255

               Gate SelectorCode32,SpuriousHandler,0,DA\_386IGate

      %endrep

      IdtLen equ $-LABEL\_IDT

      IdtPtr  dw  IdtLen

                 dd 0

       ;END of[SECTION .idt]

--------------------------------------------------------------------------------

rep指令的作用是批量产生n个数据结构。本例是255个，从0开始算应为0~~ffh

自定义中断示例：定义了20h和80h中断

--------------------------------------------------------------------------------

 LABEL\_IDT:

      %rep 32

               Gate SelectorCode32,SpuriousHandler,0,DA\_386IGate

      %endrep

      .020h: Gate SelectorCode32,ClockHandler,0,DA\_386IGate

       %rep 95

               Gate SelectorCode32,SpuriousHandler,0,DA\_386IGate

       %endrep

      .080h: Gate SelectorCode32,UserIntHandler,0,DA\_386IGate

--------------------------------------------------------------------------------

首先批量产生32个，16进制表示为00h~~1fh

后面接着是20h中断

后面批量产生95个，加上前面的33个，一共是128个中断，16进制表示为00h~~7fh。

所以后面接着是80h中断。

下面看一个时钟中断的示例：

打开时钟中断：

--------------------------------------------------------------------------------

mov al, 11111110b ; 仅仅开启定时器中断 ;

out 021h, al ; 主8259, OCW1.  
call io\_delay

--------------------------------------------------------------------------------

IR0控制时钟中断，所以要把最后一位设置为0，打开IR0时钟中断。

在ICW2中IR0设置的中断向量为20h，所以只要时钟中断打开，系统就会在IDT中寻找20h的中断向量，从而跳转到处理函数去执行。

--------------------------------------------------------------------------------

int 80h

sti

jmp $

--------------------------------------------------------------------------------

首先调用80h中断向量的函数

sti 打开中断，这里只打开时钟中断，所以时钟在运行时就产生中断，该中断是由IR0控制的，IR0的中断向量是20h，所以跳到20h所指向的函数去执行。