第8章 中断及中断系统

8.1 中断工作原理
8.2 中断指令与中断调用
8.3 中断系统应用
8.4 中断服务程序编写

- ✓中断是CPU与外设之间进行输入输出的一种有效方法，它是CPU中止正在执行的程序而转去处理特殊事件的过程。具有这种功能实现这种过程的软、硬件环境称中断系统。
- ✓中断技术的应用非常广泛。现代计算机系统中多道程序的分时运行、实时控制、人机通讯、计算机故障处理、对I/O设备的管理等均使用中断技术。中断技术能够充分发挥计算机的软、硬件功能，提高工作效率和实时处理能力。

8.1中断工作原理

8.1.1 中断

- ✓中断是指CPU在执行当前程序的过程中，遇到了某些随机出现的外设请求，暂停正在执行的程序而转去执行为外设服务的程序；服务完毕，CPU再返回到暂停处继续执行原来的程序。故中断首先是对外设而言的，称之为外中断或硬件中断。
- ✓在386和486中把许多指令执行过程中产生错误的情况处理和内部软件中断统称为异常中断，简称为异常；把外部中断称为中断。异常和中断构成了整个系列微机中的中断系统。

- ✓在中断传送方式下，CPU不再循环查询外设的状态，而是在外设“准备好”后，主动通知CPU。具体地说，外设通过接口电路向CPU发出中断请求信号，CPU暂停执行当前正在执行的程序，转入执行相应的中断服务程序。在中断服务程序中，执行I/O操作，再返回继续执行原来被中断的程序。这样CPU就避免了把大量时间耗费在等待、查询外设信息的操作上，故提高了CPU的工作效率。
- ✓中断原理如图8.1所示。

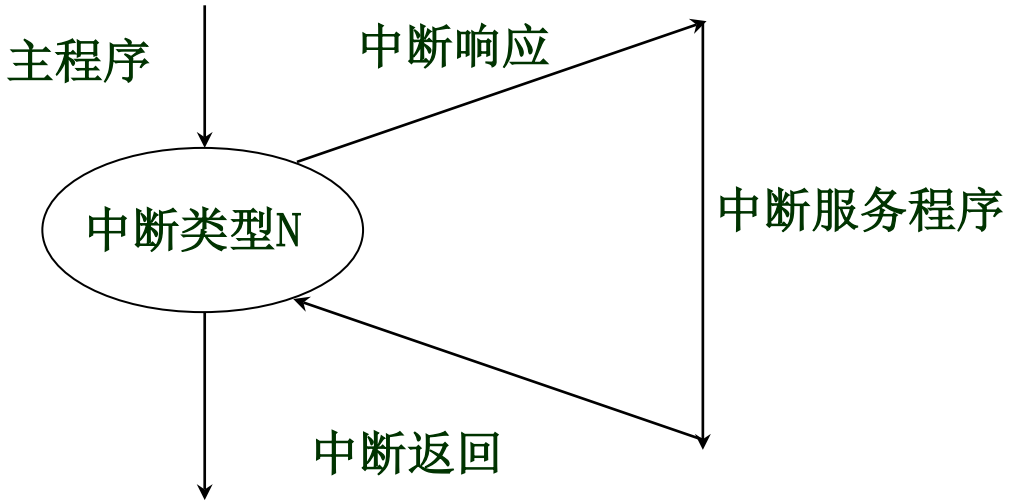

图8.1 中断工作原理示意图

8.1.2 中断类型

外设的中断是通过INTEL8259A可编程中断控制器（PIC）连接到主机上的，系统对中断源的管理也是通过8259A PIC实现的。8259A PIC共有8条外界中断请求信号线IR0－IR7，可以产生8位编码，每一种编码对应一种中断类型号，中断类型号的范围是00H－FFH，用来区分外部中断源类型。常用的中断类型号及其对应的中断名如表8-1所示。

表8-1 常用的中断类型号及其对应的中断名

中断类型号	中断类型
00-1F	BIOS中断
20-3F	DOS中断向量
40-5F	扩充BIOS中断向量
60-67	用户中断向量
68-6F	保留
70-77	I/O设备中断向量
78-7F	保留
80-FD	BASIC
F1-FF	保留

8.1.3 中断服务

每种类型的中断都由相对应的中断服务程序来处理，中断类型由中断类型号来标识。中断服务程序的功能是多种多样的，所有中断服务程序都具有相同的结构模式。中断服务程序的编写方法和标准子程序很类似，下面是编写中断服务程序的步骤，但它与子程序编写有一些不同之处。

（1）保存寄存器内容；

（2）如允许中断嵌套，则开中断（STI）；

（3）处理中断；

（4）关中断（CLI）；

（5）送中断结束命令（EOI）给中断命令寄存器；

（6）恢复寄存器内容；

（7）返回被中断的程序（IRET）。

中断处理程序除了编写结构特点外，在位置上也有特点。它们在80X86系统中，都是由操作系统提供的，都不是浮动装配的，而是固定装配的。装配的起始地址由中断向量表给出。并且，中断服务程序通常都是常驻内存的，即系统一启动，就完成中断服务程序的装配。

8.1.4 中断向量表

每种中断类型都安排一个中断类型号。80X86中断系统能处理256种类型的中断，类型号为000-0FFH。每种中断类型号都对应一个中断服务程序，中断向量是中断服务程序的入口地址，中断向量表就是各类型中断处理程序的入口地址表。

其中，存储器的低1.5KB，地址从0段0000～5FFH为系统占用，其中最低的1KB，地址从0000～3FFH存放中断向量。中断向量表中的256项中断向量对应256种中断类型，每项占用4个字节，其中2个字节存放中断服务程序的段地址（16位），另2个字节存放偏移地址（16位）。因为各服务程序的段地址和偏移地址在中断向量表中按中断类型号顺序存放（如图8.2所示），所以每类中断向量的地址可由中断类型号N×4计算出来。

中断向量表

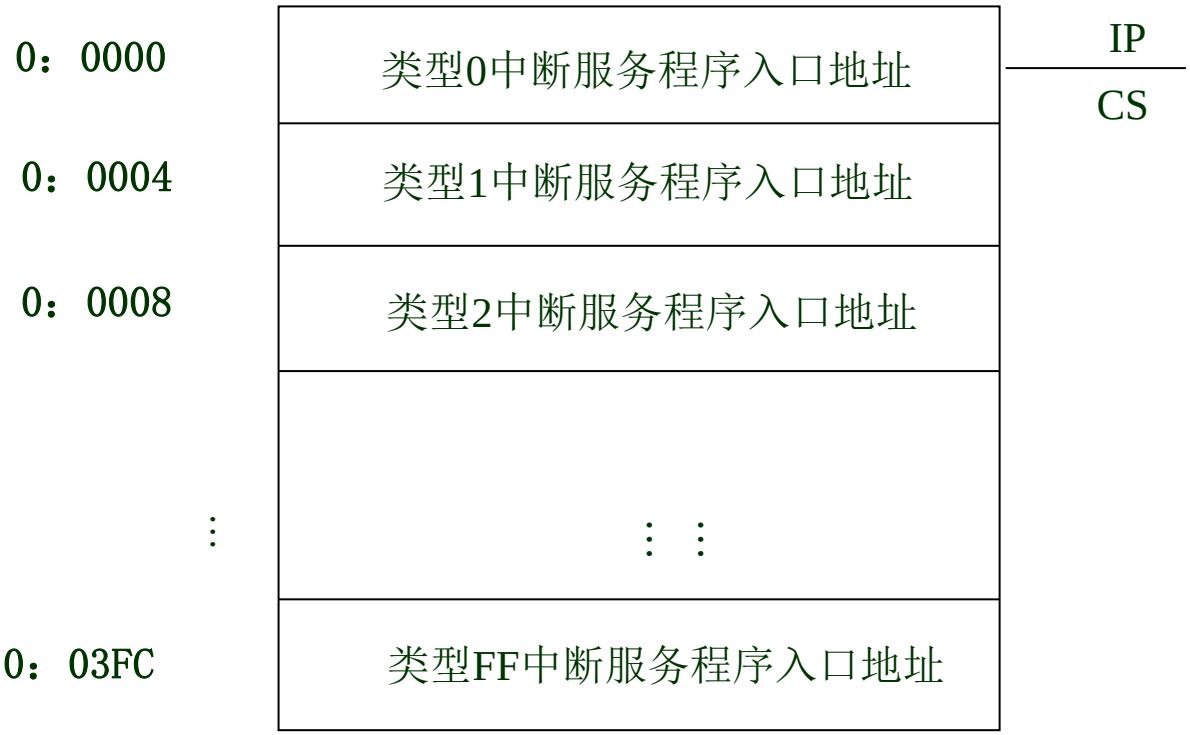

图8.2 中断向量表

8.2中断指令与中断调用

8.2.1 软中断指令

软中断又称为程序自中断，它是用中断指令实现的。

(1)中断指令

格式：INT N；其中，N为中断类型号。

功能：①（FLAGS）→↓（SP） 0→IF、TF

②（CS）→↓（SP） （4*N+2）→CS

③（IP）→↓（SP） （4*N）→IP

✓说明：首先将CS、IP寄存器的内容压入堆栈，然后IP、CS从中断向量表的4*N～4*N+3共4个字节单元中获取N号中断处理子程序的入口地址。最后根据IP和CS寄存器中地址值的内容，CPU便转入N号中断处理子程序执行中断服务。

✓为了保证在执行完中断处理子程序之后能正确返回到被中断程序处继续执行，通常在中断处理子程序的最后写上中断返回指令。

(2)中断返回指令

格式：IRET恢复断点地址

功能：①↑（SP）→ IP

②↑（SP）→CS

③↑（SP）→FLAGS ；恢复标志寄存器的内容

- ✓说明：执行IRET之后，恢复了被中断程序的断点地址和处理机状态，使处理机从中断处理子程序返回到被中断程序处继续执行。
- ✓注意：如果中断处理程序中用了堆栈，则必须在执行中断返回指令之前对栈中信息进行相应处理，保证断点地址处于栈顶，否则执行IRET后会出现无法预计的错误。

⑶CLI指令

格式：CLI

功能：将FLAGS中的IF＝0，则CPU不响应外部中断。

⑷STI指令

格式：STI

功能：将FLAGS中的IF＝1，则CPU响应外部中断。

“INT N”指令在程序中可用来调用某一特定功能的程序段（如从键盘输入一个字符，输出一个字符到显示器等）。从某种意义上说有些类似CALL指令，所不同的是“INT N”指令保护主程序标志寄存器中各标志位的状态。另外，由于类型码N可在00～0FF中任意指定，所以，用程序自中断指令不仅能测试各种中断处理子程序，还可根据需要，扩充系统功能，增加新的软中断指令。

增加一个新的软中断指令通常要做以下几件事：

(1)根据新增加的软中断指令的入口、出口参数及功能编制中断处理子程序。

(2)查看中断矢量表，找出一个空闲的中断类型号，假定为N。

(3)将新编制的中断处理子程序的入口地址送入中断矢量表4*N～4*N+3四个字节中。

此后，便可使用“INT N”实现N号中断调用，执行新增加的软中断处理子程序了。

8.2.2 中断调用

中断调用采用软中断指令“INT N”来实现。

例8.1 INT 21H（DOS中断调用）

该DOS中断调用语句执行以下5个步骤：

①取中断类型号21H；

②计算中断向量地址；

③取中断向量，偏移地址送IP，段地址送CS；

④转入中断处理程序；

⑤中断返回到INT指令的下一条指令。

✓具体中断调用过程如图8.3所示。

✓采用中断向量的方法，可以加快中断处理的速度。因为计算机可直接通过中断向量表转向相应的中断服务程序，而不需要CPU去逐个检测和确定中断原因。

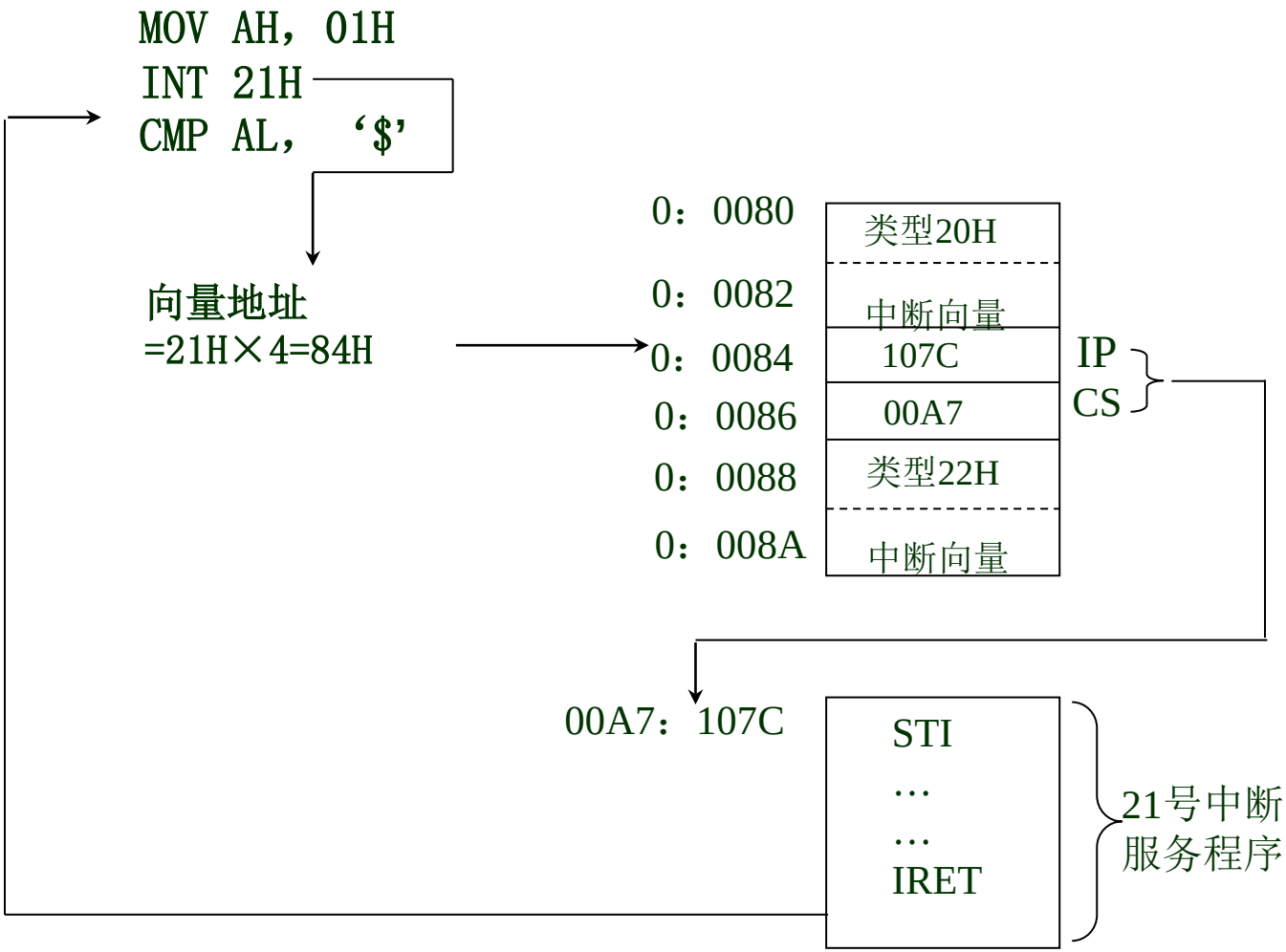

图8.3 INT 21H中断操作过程图

8.2.3 中断与子程序

子程序是程序设计的一种方法，是把具有固定功能、在编程中经常重复使用的程序段设计成子程序，在需要时调用。

中断是计算机系统支持的一种重要功能，当发生中断时，系统执行一段特定的服务程序，根据中断源的不同，需要把中断分为软件中断与硬件中断。

软件中断、硬件中断与子程序之间有一些共同之处。

（1）都需要相应程序段的支持。

（2）软件中断与子程序都由特定指令调用。

（3）发生调用时，系统均自动记载返回地址。

（4）软件中断和子程序都可以带有入口参数和出口参数。

（5）可以用子程序调用代替软件中断的调用指令。

软件中断、硬件中断与子程序三者之间也存在着本质的差别。

（1）调用方式不同。

（2）系统保护的值不同。

（3）返回方式不同。

（4）共享方式不同。

（5）在内存中存在的时间不同。

8.3中断系统应用

8.3.1 中断系统

中断系统是计算机系统提供给系统和用户的底层服务接口，分别存在于ROM（只读存储器）芯片和计算机操作系统中。驻留在ROM中的BIOS提供了系统加电自检、引导装入、主要I/O设备的处理程序、接口控制等功能模块来处理所有的系统中断。使用BIOS功能调用，给程序员编程带来了很大方便，程序员不必了解硬件I/O接口的特性，可直接用指令设置参数，然后中断调用BIOS中的程序，所以利用BIOS功能编写的程序简洁，可读性好，而且易于移植。

DOS（Disk Operating System）是IBM PC机的磁盘操作系统，它是由软盘或硬盘提供的。它的两个DOS模块IBMBIOS.COM、IBMDOS.COM提供了更多更必要的中断服务，使用DOS中断操作比使用相应功能的BIOS操作更简易，而且DOS对硬件依赖性更少些。

IBMBIOS.COM是一个输入输出设备处理程序，是ROM BIOS的功能扩展，它提供了从DOS到ROM BIOS的低级接口，可以将数据从外设读入内存，或从内存写到外设。

IBMDOS.COM包括一个文件管理程序和其它处理程序，在DOS环境下运行的程序可以调用这些服务。为了完成DOS功能调用，IBMDOS.COM把信息传送给IBMBIOS.COM，或者IBMBIOS.COM再把信息传送给ROM BIOS，形成一级或多级BIOS调用。

DOS和BIOS中断调用模块关系图

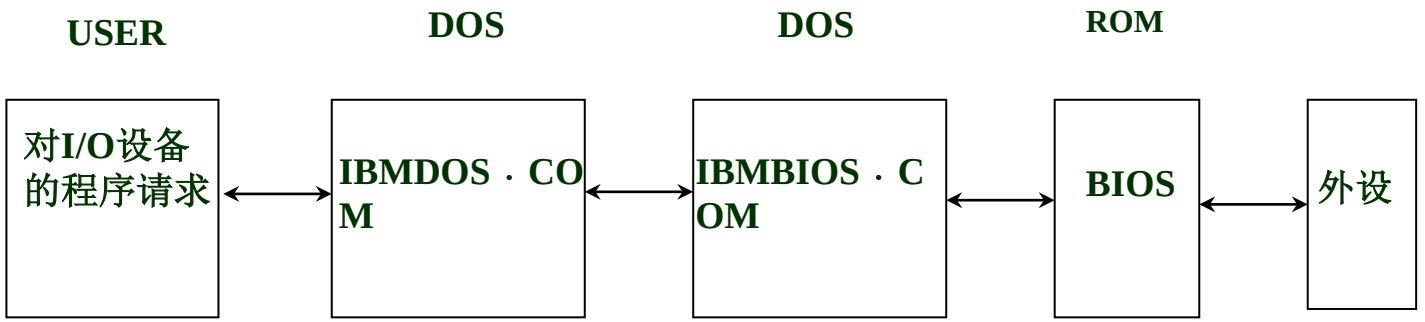

图8.4 DOS和BIOS中断调用模块关系图

在某些情况下，既能选择DOS中断也能选择BIOS中断来执行同样的功能。例如打印机输出一个字符，可用DOS中断21H的功能5，也可用BIOS中断17H的功能0。因为BIOS比DOS更靠近硬件，因此应尽可能地使用DOS功能，但在少数情况下必须使用BIOS功能，例如，BIOS中断17H的功能2为读打印机状态，它就没有等效的DOS功能。

DOS中断能处理大多数的I/O操作，但有一些功能没有提供，如声音控制等，这就要考虑用I/O指令在端口级上编程，或使用高级语言编程。

DOS中断和BIOS中断是两种最常用的中断，其它中断类型详见附录3。这些中断服务在当前流行的WINDOWS操作系统中依然存在，因为WINDOWS系统仍然加载了DOS内核。

8.3.2 中断服务程序

中断系统是操作系统提供的中断服务程序的集合，不同的中断类型对应着不同的中断服务程序，只不过中断服务程序的名字用的是00～FFH编号，即中断类型号而已。这样有利于快速查找中断服务程序的入口地址。

当计算机启动成功之后，这些中断服务程序将被调入内存。操作系统提供的以INT 21H形式调用的中断只是DOS中断（20H～FFH）服务程序的一部分，还有一部分是BIOS中断（10H～1AH）服务程序，以及其它中断类型的服务程序等。

同一种中断类型的服务程序又具有很多子功能，完成各个子功能的程序段都集中放在一起，并且有一个总控程序，构成了一个整体。例如DOS系统的21H号中断，整个中断服务子功能程序段的入口地址放在了21H号中断向量中。这是一个软件中断，调用方式是INT指令，并规定调用时AH中必须存放子功能号，不同的子功能还需要有不同的入口参数，在前面第5.6章节中讲述的1号、2号、9号、10号子功能，实际上是DOS提供给用户的21H号中断服务程序的子功能，同时还包括如何读写文件、如何申请和释放内存、如何修改中断向量、如何取得及修改系统当前的日期和时间等等

同理，BIOS中断服务程序也具有很多子功能，也构成了一个整体。BIOS中断服务程序即包括固化在计算机内ROM芯片中的服务程序，也包括DOS模块IBMBIOS.COM中的一部分。BIOS中断功能包括从计算机启动时的设备自检、系统初始化，到引导操作系统，在到对外部设备的直接控制，主要涉及键盘、显示器、打印机、串行通讯等。

BIOS中断类型主要包括10H、14H、16H、17H号等，每种中断类型服务程序都同样对应了许多子功能。

8.3.3 DOS中断调用

DOS系统中的21H号中断，又称为DOS系统功能调用。系统功能调用的常用模式如下：

（1）入口参数初始化；

（2）子功能号送入AH；

（3）INT 21H ；子程序请求中断指令

有的子程序不需要入口参数，但大部分需将参数送入指定寄存器。

程序员只需给出这三方面的信息，不必关心具体程序如何实现、在内存中的存放地址怎样等等，DOS会根据所给的信息，自动转入相应的子功能程序去执行。调用结束后若有出口参数，一般在寄存器中。有些子程序，如屏幕显示字符，调用结束后会在屏幕上马上看到结果。下面举一综合实例描述DOS系统功能调用的实现过程。

例8.2 中断调用显示系统时间和日期，在出现的提示信息中输入大写字母“D”,可显示系统当前日期；输入大写字母“T”,可显示系统当前时间；输入大写字母“Q”，可结束程序。

8.3.4 BIOS中断调用

BIOS中断调用与DOS中断调用类似，只是中断类型号不同而已，部分BIOS中断类型的功能见附录5，中断调用的方法是：首先给出入口参数，然后写明软中断指令。

例如，BIOS中断10H号类型调用方法如下：

（1）入口参数初始化；

（2）子功能号送入AH；

（3）INT 10H ；子程序请求中断指令

BIOS的10H号中断服务主要实现屏幕显示功能。

例8.3 调用BIOS中断，编程实现图形方式下的10×16行列方格的程序。

8.4中断服务程序编写

中断服务程序一般是长期保留在内存中的，在用户程序结束后还能够被其它应用程序调用，或者是CPU在响应硬件中断时调用。因此，编写一个中断服务子程序还需要掌握以下技术：如何让一段程序常驻内存，如何修改中断向量使其指向新的中断服务程序。

8.4.1 常驻内存技术

内存是由操作系统管理的，系统专门为驻留程序设计了一个中断功能调用。

驻留方法：

MOV DX，<驻留程序节长度>

MOV AH，31H

INT 21H

功能说明：

程序驻留前要告诉系统，驻留程序的长度是多少。方法是把驻留长度放在DX中，长度单位是“节”而不是字节，1节等于16个字节。如果需要驻留的程序长度是n字节，则DX的值可通过下式计算：

DX = (n÷10H) + 1 + 10H

其中(n÷10H)+1是计算出驻留程序需要多少“节”，加1是为了预防驻留程序以字节计算的长度不是16的整数倍。再加16节是因为每个程序在调入内存时，操作系统都为它安排了一个称为“程序段前缀（PSP）”的专用内存区，并且放在程序的前面，这个程序段前缀的长度是256字节，刚好16节，它必须与需要驻留的程序一起驻留内存。

如果一个应用程序中编写了一段程序需要常驻内存，总是把这段程序写在代码段的最前面，如果数据段也需要驻留，则应该数据段在前，代码段放后。计算驻留长度时，应该把数据段的长度加上代码段中驻留部分的长度一起计算。

例如，一个应用程序由代码段、数据段、堆栈段构成，数据段的各个变量共占200个字节，代码段中需要驻留的部分有500个字节，则段的编排次序应该是数据段、代码段、堆栈段，因为堆栈段是不需要驻留的，驻留节长度计算如下：

数据段：200B÷16+1=13节，代码段：500B÷16+1=32节，PSP：16节，总共66节。因此，调用21H号DOS中断的31H号子功能进行程序驻留前，必须把DX置为66，即42H。

8.4.2 修改中断向量

中断向量共有4个字节，系统专门为修改中断向量设计了一个中断功能调用。

调用方法：

```
MOV AL，<中断号>
MOV AH，25H
INT 21H
```

功能说明：

在调用之前首先初始化DS：DX寄存器的内容，其中DS：DX=新的中断服务程序的入口地址，即中断向量。

注意：在主程序的初始化部分，即修改中断向量之前，应首先保存原中断号所对应的中断向量值，再设置新的中断向量，在主程序结束部分应恢复保存过的中断向量，否则会引起系统功能混乱，以至不能正常启动。

8.4.3 中断编程实例

8086系统中有一个定时器，每隔约0．05秒向CPU发出一次8号中断请求，这是一个硬中断，不受程序的控制。8号中断服务程序中有一条调用1CH号中断的指令，而1CH号中断服务程序只有一条IRET指令，且1CH号中断是一个可供用户使用的中断，它只由8号中断调用，与硬中断有类似的效果。下面就利用1CH号中断，编写一个时钟程序。

例8.4 编写一个时钟程序，要求把时钟信号显示在屏幕的右上角，并在程序结束后常驻内存。

分析：由于系统每隔0．05秒就产生一次1CH号中断，因此在1CH号中断服务程序中应该有一个0到20的计数器，初值为20。每次中断调用该服务程序时，就把计数器的值减1，当计数器的值没减到0时，说明还没有中断20次，即不足1秒钟，因而不需要更新钟的读数。当计数值到0后，就需要让时钟向前走1秒，把秒数加1，若满60秒则向分钟数进1，分钟数满60后再向小时数进1，小时数满24就清0，并且要把已走了1秒的钟的当前读数显示在屏幕上。另外，为了计数下一秒，还要把0到20的计数器重新置初值为20。处理完这些工作后，中断结束并返回。

程序需要完成以下工作：把1CH号中断向量改为新的中断服务程序的入口地址，新的中断服务程序常驻内存

总之：

汇编语言的编程有很多技巧值得探讨，这需要在充分了解机器性能、熟练掌握指令系统的基础上，在实践中摸索、总结、提高。