# 实时数据处理中软件设计的几项关键技术

许涛　（四川师范大学计算机系　成都 610068）

**摘要**：本文介绍 DOS 环境下，以 8086 汇编语言为工具，开发实时数据处理软件的一些关键技术。

## 1. 硬中断号与 I/O 地址的选择

外部数字脉冲通过哪个中断口进入计算机是必须要考虑的一个问题。如果选择不当，将封锁某些已连接外设的中断申请，使其功能失效。最典型的是封锁键盘。要完善地解决这个问题，应从两方面考虑。首先要明确该实时处理系统涉及到哪些硬中断设备，合理排出各硬中断设备的优先级；其次要弄清目前 8259A 中断控制器还有哪些中断口空闲。然后，按外加设备的中断优先级，选择一个合适的空闲口连接上。例如 IRQ2（XT 机）或 IRQ10、IRQ11、IRQ12、IRQ15（286 以上微机）。如果某些预留的设备中断口未用，即可占用。例如 IRQ3、IRQ4（XT 机）或 IRQ3、IRQ4、IRQ5（286 以上微机）。286 机有 16 位 I/O 地址线，但仅使用了其中的 10 位，因此实际可寻址范围是 000－3FFH。其中系统板上使用 000－0FFH，I/O 插槽使用 100－3FFH，100－3FFH 中有空余地址未用，建议选择其中部分使用，例如 280H、281H、282H 等等。

## 2. 主控程序与中断服务程序的联系

通常在实时处理软件系统中，主控程序的作用是基于 OS 之上的 SHELL，是操作员与硬件系统之间的接口。它将随时接收若干外中断信号（含键盘中断），然后按中断类型和中断优先级的高低逐个加以处理。中断服务程序是为某一外中断而编写的软件处理模块。二者之间的通信是由硬中断信号实现的。这里着重谈主控程序在同时监控多个功能按键的条件下与中断服务程序之间的联系。主控程序如何协调键盘功能的系统调用与外部数据脉冲中断之间的关系至关重要。如果软件处理不当，主程序将不能正常地实时响应外中断。下面对一事例进行分析。

具体情况是：主程序同时监控十个功能键 F1－－F10 和一个外部数据脉冲。当用户不按键时，仅处理外部数字脉冲，外部数据以 BCD 码形式提供，五个 BCD 码代表一个数据，在机内组合存储以后，立即在屏幕上动态显示出来，一旦有按键信号时，优先处理键盘中断。通常易将主程序构架写成如图 1 的形式，这种处理将不能正确接收外部数据脉冲，仅实现按键功能。

失败原因分析及改进方法如下：

```
        .......
main:
        mov ah,0bh
        int 21h
        cmp al,0
        jz main
        mov ah,08h
        int 21h
        cmp al,3bh
        jz temptfc1
        cmp al,3ch
        jz tempfuc2
        ...
        cmp al,44h
        jz tempfuc10
        jmp main
        ...
inta proc near
        ...
        iret
inta endp
        .......
```

图 1 错误形式

```
        .......
main:sti
        inc ord ptr loopword
        cmp word ptr loopword,20000
        jnz main
qq: mov ord ptr loopword,0
        mov ah,0bh
        int 21h
        cmp al,0
        jz main
        mov ah,08h
        int 21h
        cmp al,3bh
        jz temptfc1
        cmp al,3ch
        jz tempfuc2
        ...
        cmp al,44h
        jz tempfuc10
        jmp main
        ...
inta proc near
        ...
        iret
inta endp
        .......
```

图 2 正确形式

当 CPU 执行系统软中断时将禁止外中断（IF＝0），

图1中语句"MAIN:"和语句"JZ MAIN"之间有一系统调用但无开中断指令"STI ",当执行一次系统调用后,外中断到来时,CPU将永远不再处理它。很显然,应该在"MIAN:"和"JZ MAIN"之间加入一条开中断指令"STI"。例如,在第一个空语句处加入"STI ",变为"MAIN:STI"。特别在"STI "指令之后增加一个延时空循环,以使中断处理程序能连续处理五个以上的BCD码,如图2所示。实验证明,完全达到所需要求。

3. PUSH与POP的使用

众所周知,PUSH与POP的使用会带来很大的方便,但用法不当也会给程序的运行、调试及维护带来灾难。PUSH与POP必须"配对"使用,才能保证程序的正确性。要做到"配对"使用,非常不易。原因是一个软件系统通常有几千行以上的程序,其中将会有许多的模块,许多的分支,程序中的数据区也将有多个,从而形成多个进程多工作区交叉操作的复杂情形。PUSH与POP散列于整个程序之中,程序的进程是随机的,可能形成一个网状轨迹模型,不易确定先前压入堆栈的数据应该在何时及时弹出。因此必须认真对待和处理。这里提供一点经验,建议在下列情形下采用PUSH和POP:

如果程序中模块之间出现有嵌套,必须保存某些数据,此时应使用PUSH和POP。由于堆栈操作为字操作,在有大量嵌套产生和数据保存时,要开辟较大的堆栈区,以免溢出。在同一模块内或模块间为并列关系时,最好采用在数据区定义暂存单元来解决,其优点是存入次数与取出次数不需配对。

4. 原码及补码的使用

在数据采集系统中,一般都涉及到大量的数值计算。例如BCD码转换成十进制数,数字滤波,分辨率与分度值的计算,以及数据的统计与分析,数据的转换与显示等。究竟采用哪种机器码,要视情况而定。一般原则是"做加减法或判别两数大小时,选补码较方便。做乘除运算、逻辑运算、地址运算以及数据转换和显示时,选原码较方便"。在整个系统处理过程中,可能会大量涉及到两种机器码的交替使用,这时一定要注意符号数的符号处理。为正确处理数的符号,必须弄清各种有关汇编指令的含义和用法。例如"ADDAX,1"和"INC AX"的效果是不完全一样的。ADD可以对有符号数或无符号数进行处理,其运算结果将影响进位CF和符号位SF;而INC仅对无符号进行运算,其结果只影响CF位(SF位无意义)。总之除算法外,能恰当地使用机器码,也是提高数值处理效率和可靠性的一种有效手段。

5. 多字节数的处理

在8086指令系统中,为加减运算提供有多字节运算指令ADD和ADC、SUB和SBB,但乘除运算仅提供单字节与双字节、双字节和四字节的运算指令DIV和MUL、IDIV和IMUL。要直接做超出范围的乘除运算极为不便。遇到这种情况,可以利用加减运算指令编一个子程序来完成相应的乘除功能。

另外在8086指令系统中,要以十进制方式输出三个字节以上的数据也不太方便,需要一定技巧。

设被处理数据用四个字节表示,它的最大值为$2^{30}$(无符号数),约等于1000,000,000。可见有十位十进制数码。为分离出每一位数码,特设定十个常值单元,然后按顺序用这些常值单元去除四字节数,可以逐步分离出各位数字。

6. 缓冲区数据与磁盘文件数据的同步

实时数据处理,特别强调"实时性"。从计算机外部接收的数字脉冲,是在机内进行处理的,处理完后一般都需要保存在磁盘上。具体作法有两种,一是在内存开辟一个较大的数据工作区,每当从外部接收数字脉冲生成一个数据后就存入该工作区,一直等到数据处理结束时,才将内存数据区的数据以文件形式存放在磁盘上。二是定时或实时将接收的数据写往磁盘文件。定时一般是利用定时器中断,按固定的时间间隔自动将内存数据写往磁盘文件。实时是将接收的数据立即写入磁盘文件。显然在突然断电或掉电的情况下,第一种处理方式将会丢失大量有效数据,从而造成不可挽回的损失。因此,笔者建议采用第二种方法。如果接收外部数据的间隔时间大于刷新磁盘文件的时间,则可采用实时刷新方式,否则采用定时器中断刷新。

7. 抗干扰问题

在微机控制系统中,干扰问题是一个常见的、恼人的问题。其解决方法已有不少的文章发表,笔者仅提供一个实际例子的解决方法。

主程序同时监控十个功能键和一个外部数字脉冲,当按下某些功能键后,系统死机。经过多方分析,其原因是这些功能键对应的模块中存在系统调用,而进入系统调用前未屏蔽外部数据脉冲引起的中断产生干扰,造成所有外中断被禁止,导致死机。解决的办法是:

每个模块的入口处加指令 IN AX,21H

AND AX,11101111B ;

采集数据中断号为 IRQ4

OUT 21H, AX

每个模块的出口处加指令 IN AX, 21H

OR AX, 00010000B

OUT 21H, AX

8. 多工作区操作

DOS 操作系统对内存的管理是分段进行的，每段最大 64KB，这对于编制大型应用程序是不利的。当程序的指令代码数量超过 64KB 时，应采用覆盖技术和交换技术，分为两个以上的代码段分别汇编，然后通过 LINK 形成可装入模块。当一个程序的数据区超过 64KB 时，可增加若干个附加数据段来解决。特别是在进行多数据库操作时，可采用此法。要访问一个附加数据段，只需给 ES 或 DS 装入相应的段基值即可。该方法比仅用一个数据区辅之以频繁的磁盘 I/O 操作来实现相同的功能效率要高得多。处理多工作区时，必须特别小心段寄存器 DS、ES 和指令 PUSH、POP 的应用。

9. 标志的使用

程序运行中，往往要涉及到基本相同的工作，但在不同的地方和不同的时刻，其具体处理稍有差异。如何区别这些时刻和地方？使用标志来解决这类问题很方便。标志是一个逻辑量，其定义方法与普通变量一样，为便于记忆，可取名为 FLAG * 。若为二值逻辑，可装入 0 和 1 表示；若为多值逻辑，可用阿拉伯数字表示各个逻辑值。在使用标志后，要注意它带来的负面影响，如果有多个模块用到两个以上的标志，则标志间的组合情形有四种以上，每种情形的结果都应考虑到并加以实际测试以免出错。

10. 防止花屏与功能键的封闭

一般微机控制系统的工作程序都是一个封闭单元，具有自己统一的用户界面和操作方式。多采用键控及菜单操作方式，这就要求程序的坚固性和可靠性要高。否则，随意按键将引起花屏，甚至失控或死机。若进行系统调用"MOV AH, 0AH, INT 21H"时，不按规定输入可打印字符，而按 ESC 键，屏幕光标将乱跑，引起花屏。解决办法是用"接收字符中断"来替换"接收字符串中断"，或利用 BIOS 中断来解决。计算机专业人员所开发的软件大多数是面向基层普通用户，各种各样的误操作都可能出现，为防止误操作带来的损失，还应封闭在当时不起作用的各种按键。

11. 子程序及函数的应用

用汇编语言编程有一个最大的缺点是代码长，为使程序短小精悍，不超过 64KB 限制，应充分利用子程序和函数的功能。

12. 系统调试

实时处理系统的调试比普通系统的调试更加困难。它不仅与计算机有关，而且与外部设备有关。它的调试可分为两部分，一部分是机内设备与机外设备的单独调试(静态与模块调试)，另一部分是联调(动态与整体调试)。一般分块、静态调试问题较易解决，难的是联调。即使前者顺利通过，也不能保证联调成功，常常出现意料之外的问题。在联调期间，合作人员必须在一起进行工作，相互商讨，因为大多数情况是由不同工种或专业的人员共同设计制造一个系统，最好能连续作战，直到问题的解决。在系统投入运行前，应严格测试各项指标，将所有分支都走遍，争取一次成功。

13. 其它

在系统开发中，还应注意字地址的偶数性，若字地址为奇地址，在操作时可能出现意外的情形。因字地址为奇数时，CPU 将两次访问内存，其间可能有硬中断产生。为防止该情况，可以通过填充无用的单元达到此目的。

如果系统要涉及汉字处理，可以按下述方法简单处理。中文的存储可按普通字符串存储。中文输入可采用系统调用的 0AH 号功能，或采用菜单方式用代码输入(汉字菜单以字符串方式存于数据区)。中文输出可采用系统调用的 09H 号功能，或 BIOS 的 17H 调用(输出到打印机)。需注意的是在进入系统控制程序之前，应先进入中文操作环境。

14. 结束语

本文所阐述的若干问题，是微机实时控制系统开发过程中常见的典型问题，如果能很好地处理这些问题，将有助于提高开发效率，增加系统的可靠性及完整性。

(编者注：由于原文篇幅较长，本刊对部分内容做了删略)

参考文献

[1]沈美明，温冬蝉，"IBM－PC 汇编语言程序设计"，清华大学出版社，1991 年 6 月第一版

[2]钮爱英，"计算机组成与结构"，清华大学出版社，1990 年 10 月第一版

[3]李振格，"DOS 系统调用详解"，北京航空航天大学出版社，1994 年 8 月第一版