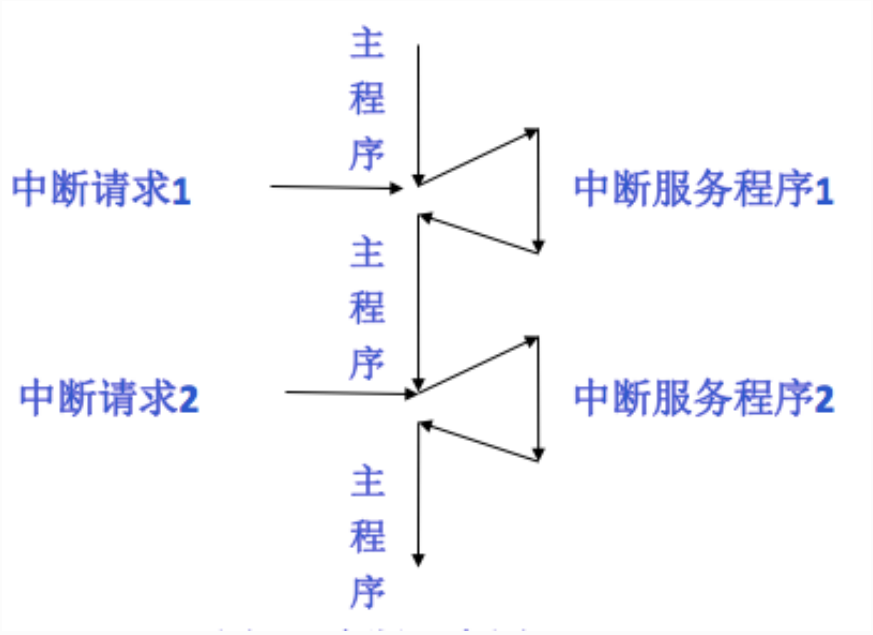
中断系统

中断概念

中断定义& 原理

- 什么是中断?

  就是指CPU在执行程序的过程中， 由于某种外部或内部事件的作用，使CPU停止当前正在执行的程序而去为该事件服务，待时间服务结束后，又能自己返回到被终止的程序中继续执行过程。



- 中断源分类
  - 外部中断源
    - I/O设备（键盘、打印机，显示器）
    - 数据通道（磁带、磁盘）
    - 时钟（8254）
    - 故障源（掉电、存贮器奇偶校验）
  - 内部中断源
    - 执行INT软件中断指令
    - CPU指令执行产生的异常
- 中断的分类
  - 外部中断

    CPU意外的设备，通过中断信号引脚（INTR、NMI）引发的中断
    - INTR引发可屏蔽硬件中断
    - NMI引发非屏蔽硬件中断
  - 内部中断
    - 软件中断

      由INT N 引发的终端，分为DOS中断、BIOS中断
    - 异常

      由于CPU本身故障、程序故障引发的终端

中断类型&中断向量

- 中断类型码

  为了区别这些不同的中断，微机系统分配了0~255的中断类型码，前32个为系统所专用，后224个可由用户设定，部分中断号如下：

```
0型中断      除法错中断
1型中断      单步或缺陷中断
2型中断      非屏蔽硬件中断
3型中断      断点中断
4型中断      溢出中断
5型中断      屏幕打印
08H～0FH 可屏蔽硬件中断
10H～1FH BIOS中断
20H～3FH DOS中断
```

- 中断向量

  定义：实模式下，中断服务子程序的入口地址，由服务程序段基址以及偏移地址共4个字节。

- 中断向量表

  通过一个地址指针表与终端服务的入口地址相联系，实模式下成为中断向量表；保护模式下，该表成为中断描述符表

  - 实模式下，中断向量表设置在系统的RAM最低端的1K单元（00000H ～ 003FFH）
- 中断向量表的初始化
  - 方法1：手动编写中断向量

```
* n型向量存放在4* n ～ 4* n+3
* 向量地址转换技巧：将向量或者地址转换成2进制，然后通过左移或右移快速转换
* 手动修改中断向量地址
    CLI
    PUSH     DS
    MOV     AX, 0000H
    MOV     DS, AX
    MOV     BX, 4*n
    MOV     AX, OFFSET SERVICE
    MOV     [BX], AX
    MOV     AX, SEG SERVICE
    MOV     [BX+2], AX
    POP         DS
    STI
```

  - 方法2：利用DOS功能调用

```
    CLI
    PUSH DS
    PUSHA
    MOV AX,SEG SERVICE
    MOV DS,AX
    MOV DX,OFFSET SERVICE
    MOV AH,25H
    MOV AL,n
    INT 21H
    POPA
    POP DS
    STI
```

(DOS功能调用)：专门用于中断向量的读出和写入

- int21H 35H
  - 功能：读出n型中断向量
  - 入口：AL=中断类型码
  - 出口：ES:BX=n型中断向量
- int21H 25H
  - 功能：写入N型中断
  - 入口：DS=中断程序段基址；DX=中断程序偏移地址；AL=中断类型码

中断响应过程

1. 标志寄存器处理

   将F寄存器的内容压入栈

   并且使I标志为0——禁止单步操作

   使得I标志为0——CPU关中断状态

2. 断口地址压入栈

   先压入CS段基址

   再压入IP地址从

3. CPU对应中断向量取出n型程序入口地址：IP：CS
4. CPU恢复断点IP，CS，恢复标志寄存器
5. 中断返回，继续执行

中断指令、管理

多级中断管理

- 8086 中断优先级

中断类型	优先级
除法错中断	最高
软件中断INT n	↓
断点中断	↓
溢出中断INTO	↓
NMI中断	↓
INTR中断	↓
单步中断	最低

- 中断系统应该具备的功能
  - 对于硬件中断，具备屏蔽和开放的功能
  - 能够响应优先级高的中断
  - 能够实现中断嵌套

- 处理中断完成返回断点

中断指令

- 开中断指令 STI 使I标志置为1，CPU处于开中断状态（开可屏蔽硬件中断）
- 关中断指令 CLI 使I标志置为0，CPU处于关中断状态 `注：ST～均为置一，CL～均为置零`
- 软件中断指令 INT n 无条件转向n型中断服务程序
- 中断返回 IRET
  - 一次从栈顶弹出6个元素，IP，CS，F
  - RET指令弹出IP，CS
- INTO 先判别F寄存器中O标志位是否为1，如果是则调用类型4中断程序

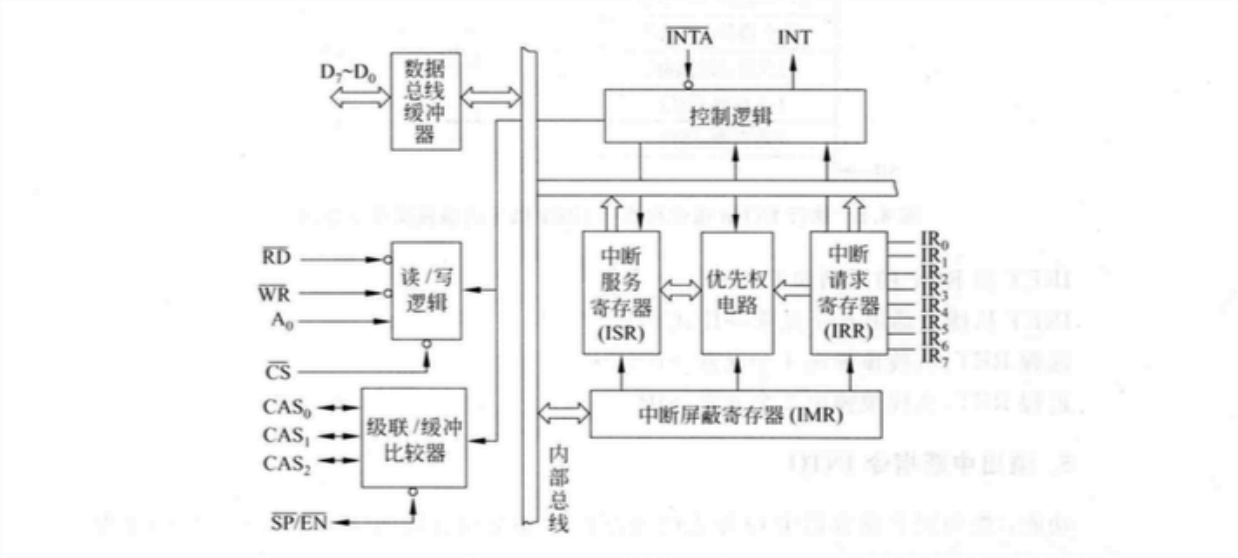
中断控制器— — 8259A

8259的结构

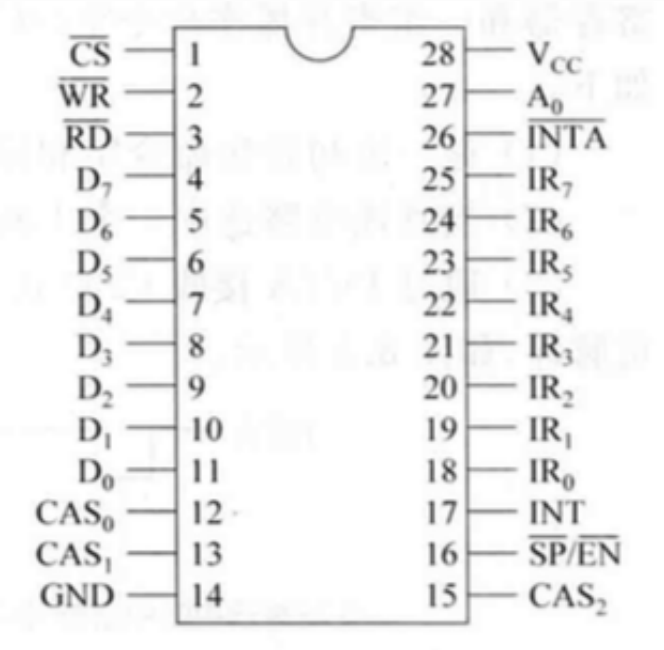
- 中断请求寄存器（8位 IRR） 寄存引脚IR0～IR7的中断请求，中断响应时，IRR的相应位复位
- 中断屏蔽寄存器（8位IMR） 保存程序员写入的中断屏蔽字，屏蔽该位为1，通过向屏蔽寄存器地址传入开放8位地址数据
- 优先权电路 比较同时送达请求的优先级，选择优先级高的请求
- 中断控制电路
  - 存一组初始化命令字和操作命令字，通过译码产生内部控制信号
  - 当判优电路选中一个中断源时向CPU提出中断请求
  - 通过INT接受CPU传来的中断响应，该信号为两个连续的负脉冲
- 中断服务寄存器（8位ISR） 记录CPU正在服务的中断源
- 数据总线缓冲器
  - 完成CPU数据线配接
  - 接受初始化命令字、操作字
  - 选中第二个中断脉冲的中断类型码
- 读/写控制模块

  接受片选信号CS、端口选择信号A0、读写控制信号RD、WR
- 级连/缓冲比较器

  1位8259可以管理8级中断，二片8259可管理15级中断，缓冲器是未完成多片级连的模块



- 8259的外部引脚
    - A0为地址线，当A0=0是偶地址，A0=1是奇地址
    - 通常IR0的优先级最高，IR7优先级最低



- 8259——CPU响应可屏蔽硬件中断

① 首先由中断请求寄存器寄存加到引脚IR0～IR7上的中断请求。
② 在中断屏蔽寄存器的管理下,没有被屏蔽的中断请求被送到优先权电路判优。
③ 经过优先权电路的判别,选中当前级别最高的中断源,然后从引脚INT向CPU发出中断请求信号。
④ CPU满足一定条件后,向8259A发出2个中断响应信号(负脉冲)。
⑤ 8259A从引脚INTA收到第1个中断响应信号之后,立即使中断服务寄存器中与被选中的中断源对应的那一位
置1,同时把中断请求寄存器中的相应位清0。
⑥ 从引脚INTA收到第2个中断响应信号后,8259A把选中的中断源类型码n,通过数据线送往CPU。
⑦ 在实模式下,CPU从4×n～4×n＋3单元取出该中断源的中断向量→IP、CS,从而引导CPU执行该中断源的
中断服务程序。

8258中断管理方式

PC机中的中断系统

PC中断管理

- 采用边沿触发，上升沿为中断请求
- 采用常规屏蔽触发，通过写入屏蔽字触发
- 完全嵌套优先级管理，IR0中断请求级别最高
- 采用常规结束，结束前送入中断结束命令

非屏蔽中断（NMI引脚触发）

- 响应条件
    - NMI有中断请求，系统没有DMA请求
    - CPU当前指令执行完毕
- 响应过程

  CPU在每一条指令的最后一个时钟周期检测NMI引脚，响应NMi引脚时不响应其他外部接受的中断向量。8086中非屏蔽对应的向量号固定为2.非屏蔽中断过程中屏蔽再次响应NMI，以IRET结束。

可屏蔽中断（INTR引脚触发）

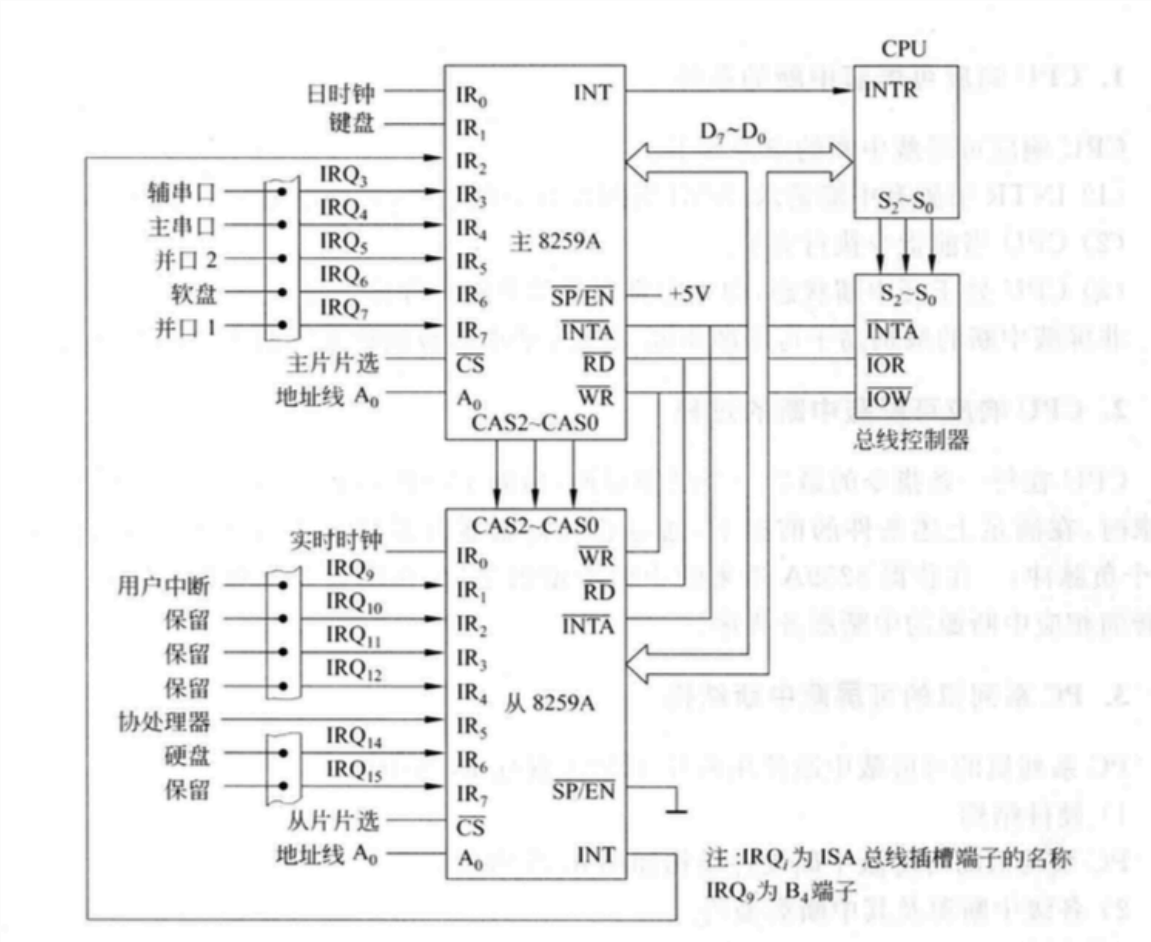
- 响应条件
    - INTR引脚有中断求，NMI引脚没有中断请求，系统没有DMA请求

- CPU当前指令执行完毕
- CPU处于开中断状态，I标志为1

- 响应过程

  CPU在每一条指令的最后一个时钟周检测INTR引脚，满足上述条件的情况下，向8259A发出中断响应信号（两个负脉冲），获得类型码后在实模式下查询向量地址，转向服务程序，中间可响应其他中断。

- 硬件结构



- 必记的中断类型码
  - 日时钟中断（主8259—IR0）：08H
  - 从8259连接（主8259—IR2）
  - 实时时钟（从 8259—IR0）：70H
  - 用户中断（从 8259 —IR1）：71H ——> 0AH
- 系统分配的口地址
  - 主8259
    - 奇地址：21H
    - 偶地址：20H
  - 从8259
    - 奇地址：A1H
    - 偶地址：A0H
  - 奇地址为中断屏蔽寄存器口地址
  - 偶地址为中断结束命令寄存器口地址

- 中断优先级

  从图中可以看出级别由高到低分别为：主IRO、IR1、从IR0~IR7、主IR3~IR7

- 中断结束字
  - 接入主片中断源，结束时向主8259中写结束字`

```
...
MOV AL,20H
OUT 20H,AL
恢复现场
IRET
```

  - 接入从片中断源，结束时需要向主、从8259中写结束字

```
...
MOV AL,20H
OUT 20H,AL
OUT 0A0H,AL
恢复现场
IRET
```

中断应用实例

日时钟中断

- 中断源：主—IR0
- 中断类型：08H
- 计算机系统时间

  BIOS系统规定：40H:6CH～40H:6FH这4个单元（共32位）为日时钟计数器，每55ms加1次，计数 到:001800B0H，为24小时,其计数值供系统软件使用。

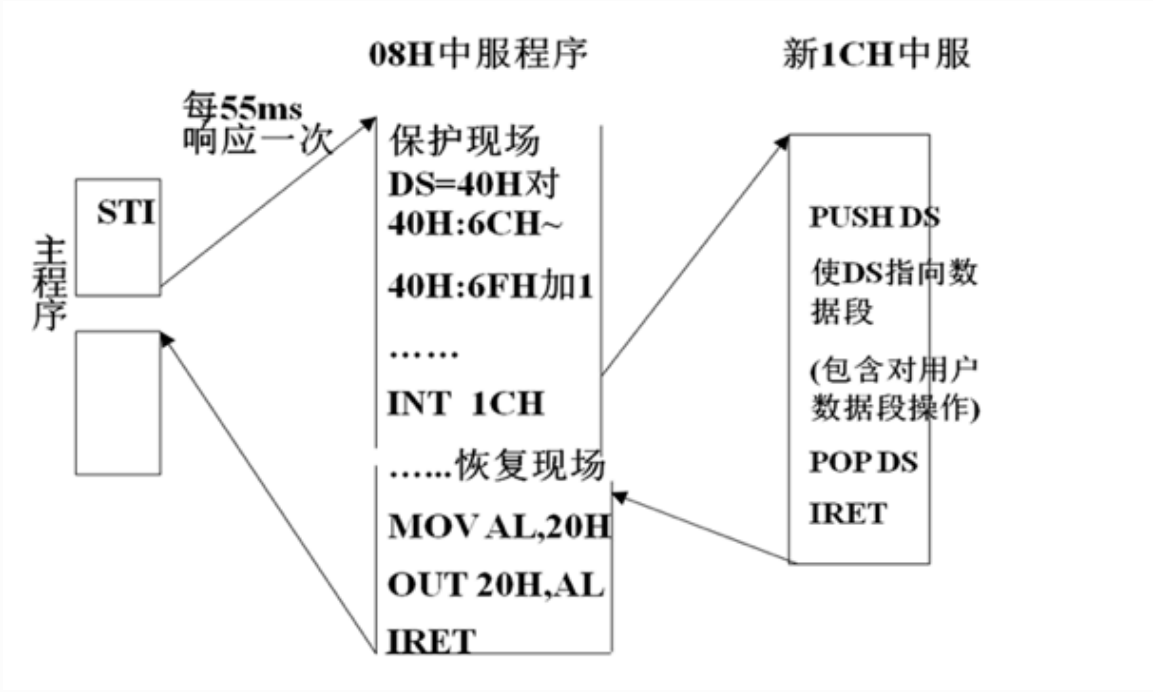
- 设计程序1：在08服务程序中会调用一次1CH服务程序，然后返回8，1CH中断也称为日时钟的外扩中断，用户可开发新的1CH取代从而设计新的55ms倍数的中断服务子程序
- 设计程序2：我们也可以直接设计新的08H服务程序，修改对应的入口地址即可，结束时需要向主8259写结束字
- 设计程序时需要注意不要嵌套调用21H功能,避免DOS重入

- 程序要求：

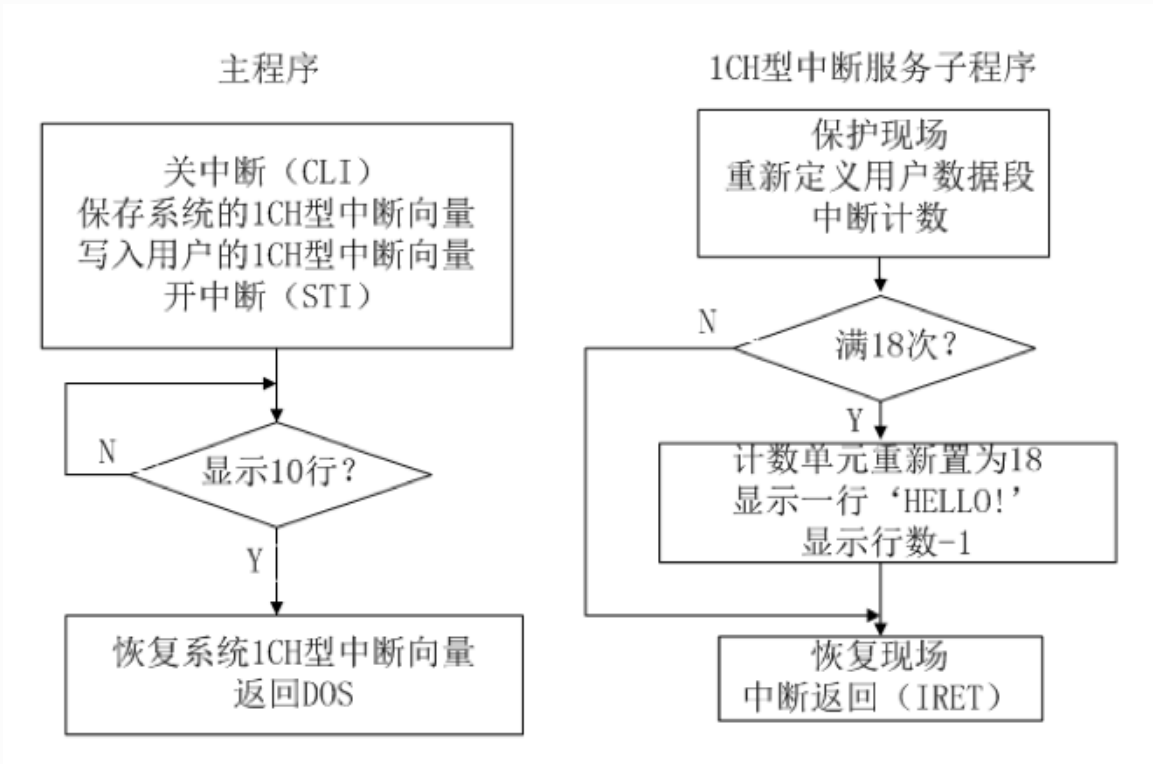
  要求利用PC系统机上的8254的0号定时计数器引发的日时钟中断，设计程序：每间隔1秒在PC终端屏幕上显示1行字符串“ HELLO！ ”，显示10行后结束。

方法1：置换1CH服务程序

- 程序执行过程：



- 程序思路：



- 注意事项
  - 中断服务执行前后需要保存DS以及恢复DS
  - `MOV DX,WORD PTR OLD1C` 和 `MOV DS, WORD PTR OLD1C+2` 两条指令先后顺序不可颠倒,以免改变当前执行位置。
  - 中断服务程序中使用了DOS功能调用来显示字符串。这是一种不推荐的做法。要避免发生DOS重入，可以选择有BIOS替换功能

- 代码实现

```
 [程序清单]
.486
DATA      SEGMENT USE16
MESG          DB      'HELLO!',0DH,0AH,'$'
OLD1C     DD      ?
ICOUNT    DB      18                    ;中断计数初值
COUNT     DB      10                    ;显示行数控制
DATA      ENDS
CODE      SEGMENT USE16
          ASSUME  CS:CODE,DS:DATA
BEG:      MOV     AX,DATA
          MOV     DS,AX
          CLI                           ;关中断
          CALL    READ1C
          CALL    WRITE1C
          STI                           ;开中断
SCAN:     CMP     COUNT,0
          JNZ     SCAN          ;是否已经显示10行，否转
          CALL    RESET
          MOV     AH,4CH
          INT     21H
;---------------------------------
SERVICE   PROC
          PUSHA                         ;保护现场
          PUSH    DS                    ;DS=40H
          MOV     AX,DATA
          MOV     DS,AX                 ;重新给DS赋值
          DEC     ICOUNT                ;中断计数
          JNZ     EXIT                  ;不满18次转
        MOV   ICOUNT,18
          DEC     COUNT               ; 显示行数减1
        MOV   AH,9            ; 显示字符串
      LEA     DX,MESG
      INT     21H
EXIT:     POP     DS                  ;恢复现场
          POPA
          IRET                ;返回系统8型中断服务程序
SERVICE   ENDP
;---------------------------------
READ1C    PROC                  ;转移系统1CH型中断向量
          MOV     AX,351CH
          INT      21H
          MOV     WORD PTR OLD1C,BX
          MOV     WORD PTR OLD1C+2,ES
          RET
READ1C    ENDP
;---------------------------------
WRITE1C   PROC                  ;写入用户1CH型中断向量
          PUSH    DS
          MOV     AX,CODE
```

```
          MOV     DS,AX
          MOV     DX,OFFSET SERVICE
          MOV     AX,251CH
          INT      21H
          POP      DS
          RET
WRITE1C   ENDP
;---------------------------------

RESET     PROC                  ;恢复系统1CH型中断向量
          MOV     DX,WORD PTR OLD1C
          MOV     DS,WORD PTR OLD1C+2
          MOV     AX,251CH
          INT      21H
          RET
RESET     ENDP
CODE      ENDS
             END     BEG
```

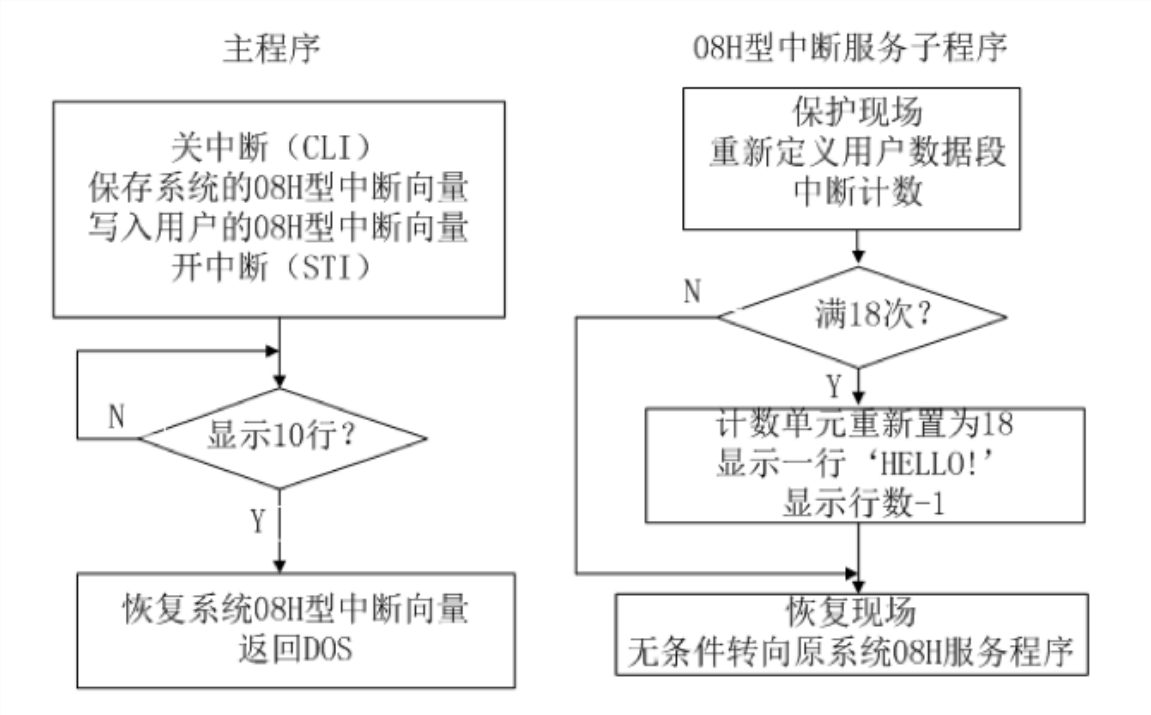
方法2：设计新08H服务程序

置换系统08H中断向量，将其指向自定义的中断服务程序。设定1个计数变量，每当系统提请18次日时钟中断时在自定义的中断服务程序中完成1次字符串显示。

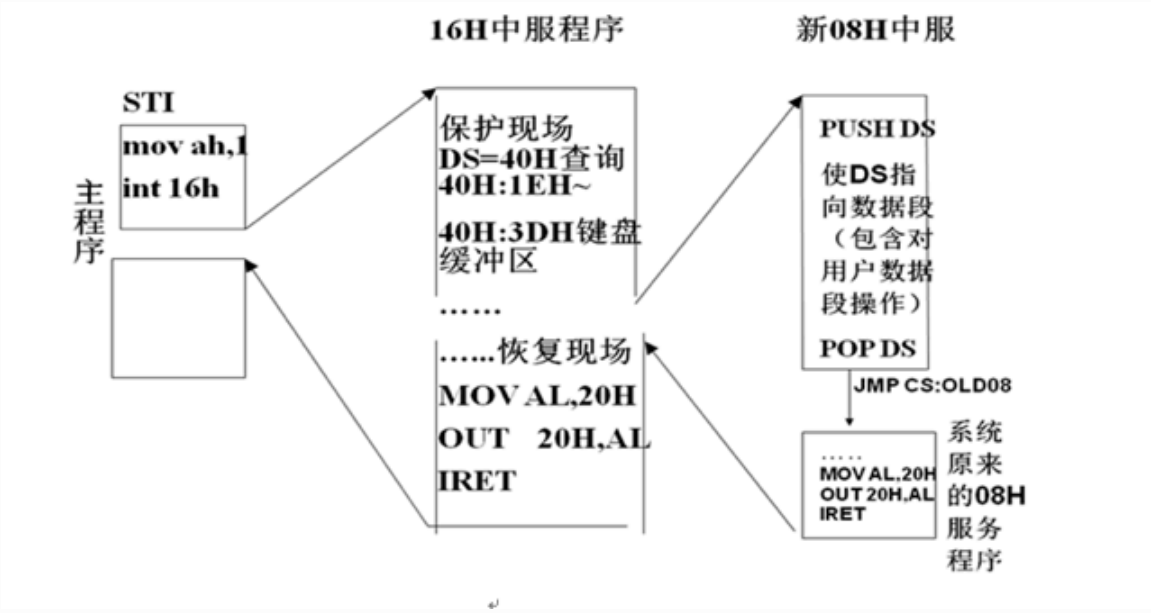
- 程序编写思路

```
①保存原来系统的08H中断向量到代码段OLD08双字单元 *
CODE   SEGMENT USE16
OLD08  DD ? ; 保存系统08H中断向量的变量
ASSUME  CS:CODE,DS:DATA
②置换08H中断向量使其指向我们自己的中断服务程序
③在每次中断服务程序的末尾调用1次日时钟中断服务程序 *
          JMP  CS:OLD08
④原来系统08H中断需要完成的任务，包括往主8259A写中断结束命令字,返回断点等均在原来的中断服务程序中完成。
```

- 程序框图



- 程序执行过程



- 注意事项
  - OLD08双字单元必须定义在代码段,不能设置在数据段。这样做能保证每次中断服务程序的末尾，能顺利地转到系统日时钟中断服务程序。
  - 程序结束需要写入结束字
- 程序代码

```
.486
DATA        SEGMENT  USE16
MESG        DB       'HELLO! ',0DH,0AH,'$'
ICOUNT      DB       18
COUNT       DB       10
DATA        ENDS
CODE        SEGMENT USE16
OLD08       DD ? ;代码段中保存系统08H中断向量的变量
            ASSUME  CS:CODE,DS:DATA
```

```
BEG:      MOV      AX,DATA
MOV      DS,AX
          CLI                  ;关中断
          CALL    READ08       ;保存原来的08中断向量
          CALL    WRITE08    ;置换08H型中断向量指向              自定义
中断服务程序
         STI                   ;开中断
SCAN:     CMP     COUNT,0
          JNZ      SCAN      ;是否已经显示10行，否转
          CLI
          CALL     RESET        ;恢复系统08中断向量
          STI
          MOV     AH,4CH
          INT      21H           ;返回 DOS
;-------------------------------------------
SERVICE   PROC              ;中断服务程序
          PUSHA
          PUSH     DS
          MOV     AX,DATA
          MOV     DS,AX
          DEC      ICOUNT     ;计18次55msx18=990ms
          JNZ       EXIT

MOV      ICOUNT,18
          DEC       COUNT         ; 显示行数减1
          MOV      AH,9            ;显示字符串
          MOV      DX,OFFSET MESG
          INT       21H
EXIT:   POP      DS
          POPA
          JMP      CS:OLD08     ;转向原来的08H服务程序
SERVICE   ENDP
;-------------------------------------------
READ08    PROC      ;保存原来系统的08H 中断向量
          MOV      AX,3508H
          INT       21H
          MOV      WORD PTR OLD08,BX
          MOV      WORD PTR OLD08+2,ES
          RET
READ08    ENDP
;-------------------------------------------
WRITE08  PROC      ;置换08H型中断向量指向自定义中断服务程序
          PUSH      DS
          MOV      AX,CODE
          MOV      DS,AX
          MOV      DX,OFFSET SERVICE
          MOV      AX,2508H
          INT       21H
          POP       DS
          RET
WRITE08   ENDP
;-------------------------------------------
```

```
RESET     PROC       ;恢复系统08中断向量
      MOV   DX,WORD PTR OLD08   ;注意和后一条指令的顺序
          MOV    DS,WORD PTR OLD08+2
          MOV       AX,2508H
          INT        21H
          RET
RESET     ENDP
CODE      ENDS
          END     BEG
```

用户中断

- 中断类型 71H
- 设计流程
  - ①开中断、保护现场；
  - ②向从8259A发出中断结束命令

    ```
    ① 若用户中断定义为0AH型，服务程序结束前只向主8259A送结束命令。
              MOV  AL，20H
              OUT  20H，AL
    ②若用户中断定义为71H型， 服务程序结束前，向主从8259A各送一中断结束命令
               MOV  AL，20H
              OUT  20H，AL
               OUT  0A0H，AL
    ```

  - ③执行INT 0AH,转向0AH服务程序。
  - 注意：0AH服务程序是用户预先设计好的,其中断向量已经存放在系统RAM 4×0AH～4×0AH＋3单元。
- 只有从8259IMR1置0，主8259IMR2置0，其中断请求方能送到CPU。 为实现用户终端，（不同机器用户中断可能是打开或者关闭状态)，保证中断申请由8259中断控制器提向CPU

  ```
  IN  AL,0A1H
  AND AL，11111101B
  OUT 0A1H,AL
  IN  AL,21H
  AND AL,111q11011B
  OUT 21H,AL
  ```

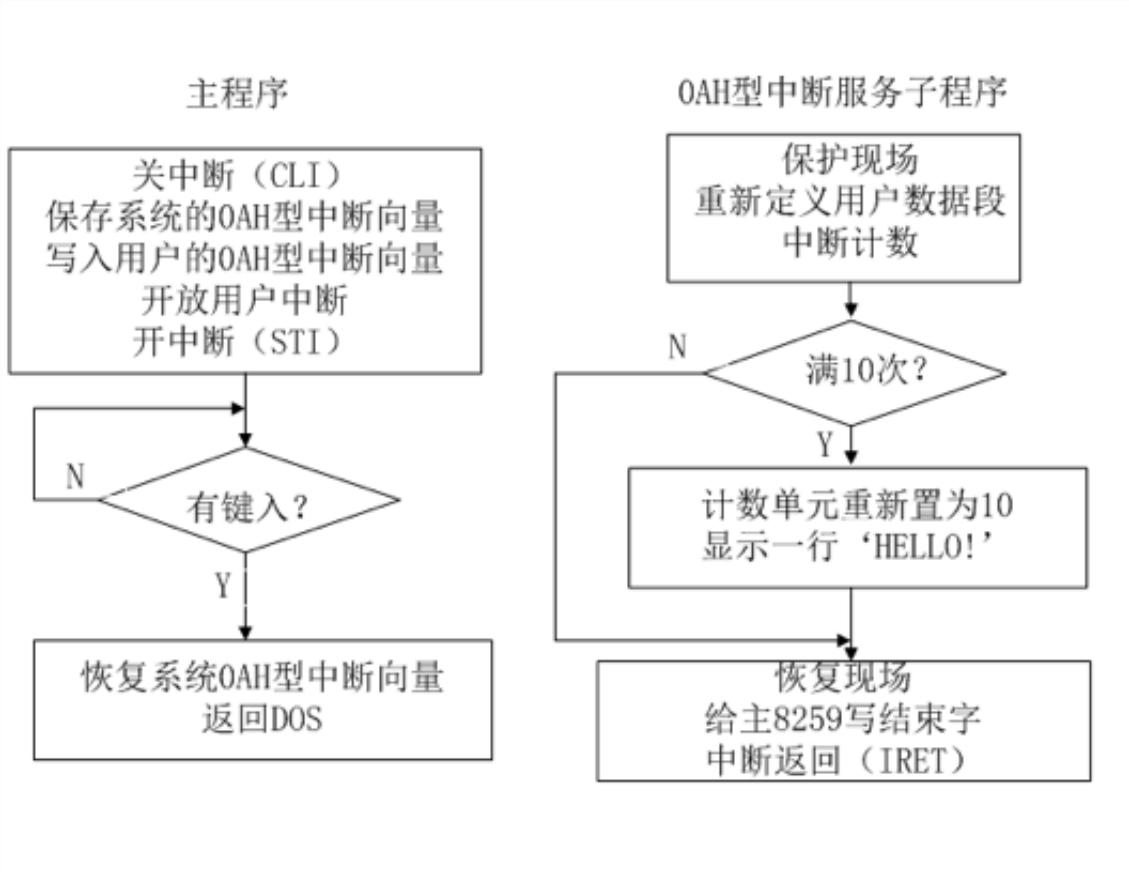
- 程序要求

  一外扩8254 0号计数器输出是周期为100ms的方波，将该8254的OUT0接至系统总线插槽B4端子，如下图所示。利用该8254的OUT0输出作为定时中断源，编程实现每隔1秒在屏幕上显示字符串'HELLO',主机有按键时显示结束。

  由B4端子接入，CPU响应接入71H型终端服务程序，在其中执行软件终端0AH，所以我们的修改服务程序即为0AH

1. 保存原来系统的0AH终端向量到数据段单元
2. 置换中断向量
3. 打开用户终端
4. 中断服务程序每执行10次显示一次字符串
5. 程序结束前送结束命令
6. 返回原来保存的0A向量

- 程序原理图




```
.486
DATA        SEGMENT USE16
MESG            DB      'HELLO!',ODH,0AH,'$'
OLD0A       DD      ?
ICOUNT      DB      10                      ;中断计数初值
DATA        ENDS
CODE        SEGMENT USE16
            ASSUME  CS:CODE,DS:DATA
BEG:        MOV     AX,DATA
            MOV     DS,AX
            CLI                             ;关中断
            CALL    READ0A
            CALL    WRITE0A
            MOV     ICOUNT,10
            MOV     AH,9        ; 显示字符串
            LEA     DX,MESG
            INT     21H
```

```
EXIT:     MOV     AL,20H
          OUT     20H,AL    ;给主8259A写结束字
          POP     DS                ;恢复现场
          POPA
          IRET          ;返回系统71型中断服务程序
SERVICE   ENDP
;---------------------------------
READ0A    PROC            ;转移系统0AH型中断向量
          MOV     AX,350AH
          INT     21H
          MOV     WORD PTR OLD0A,BX
          MOV     WORD PTR OLD0A+2,ES
          RET
READ0A    ENDP
;---------------------------------

WRITE0A   PROC            ;写入用户0AH型中断向量
          PUSH    DS
          MOV     AX,CODE
          MOV     DS,AX
          MOV     DX,OFFSET SERVICE
          MOV     AX,250AH
          INT      21H
          POP     DS
          RET
WRITE0A   ENDP
;---------------------------------
I8259A   PROC
          IN      AL, 0A1H
          AND     AL, 11111101B
          OUT     0A1H, AL      ; 从8259A IMR1置0
          IN      AL, 21H
          AND     AL, 11111011B
          OUT     21H,AL          ; 主8259AIMR2置0
      RET
I8259A   ENDP
;---------------------------------
RESET     PROC    ;恢复系统0AH型中断向量
          MOV     DX,WORD PTR OLD0A
          MOV     DS,WORD PTR OLD0A+2
          MOV     AX,250AH
          INT     21H
          RET
RESET     ENDP
CODE      ENDS
          END     BEG
```

- 软硬件中断相同点
    - 都会引起程序终止

    - CPU都会获得终端类型码，通过对应的入口地址获得该中断源的终端服务程序
- 软硬件中断的不同点
    - 引发方法不同：硬件终端是外设通过INTR和NIM引脚发起的，软件中断通过INT n指令执行
    - 获取方式不同，可屏蔽和非屏蔽分别由中断控制器8259A和自动产生；软件中断由指令提供
    - 响应条件不同，硬件终端存在可屏蔽，也存在非屏蔽。软件中断是不可屏蔽的
    - 结束方式不同。可屏蔽中断需要处理结束后发送中断结束命令+IRET，软件中断只需要IRET