大连理工大学本科设计报告

（计算机原理实验综合设计）

题目：基于 386EX 实验系统的步进电机调速系统设计

课程名称：计算机原理实验

学院（系）：电子信息与电气工程学部

专　　业：自动化

班　　级：电自 1704

学　　号：201795114

学生姓名：常海颖

成　　绩：

2019　年　12　月　31　日

题目：基于 386EX 实验系统的步进电机调速系统设计

1 设计要求

以 80386 为控制核心，采用八拍驱动方式驱动步进电机，系统可实现通过电位器控制步进电机转速、开关控制步进电机转向的功能，并能通过数码管和 LCD12864 对转速和转向等信息进行实时显示，并在最大速度或最小速度时进行报警，具体要求如下：

(1)步进电机采用八拍驱动方式，由 8255 的 PA 端口输出相序信号，控制步进电机旋转；

(2) 使用 ADC0809 模块读取电位器抽头电压，并转换为数字量,步进电机的转速可通过电位器人工调节,有八个速度档位, 速度档位可通过 LCD12864 和一位数码管进行显示；

(3)步进电机的转向可通过一位开关进行控制，拨动开关实现转向改变, 转向可通过 LCD12864 进行显示

(4)步进电机的转速达到最快 8 档或最低 1 档时，蜂鸣器会发出报警声，同时 LED 点阵显示报警信号。

2 设计分析及系统方案设计

系统方案设计如下：

(1)步进电机的驱动：步进电机采用八拍驱动方式，通过 8255 的 PA 端口的低四位输出相序信号，改变相序时间间隔即可实现调速，改变输入相序的次序即可改变转向。将 8255PA 端口的低四位与步进电机的输入端连接，同时将步进电机的四路输入与四路 LED 相连，监控控制步进电机的相序。

(2)步进电机转速的调节：步进电机的转速采用软件延时法进行调节。通过 ADC0809 读取电位器上的电压，将其转化为数字量，将该数字量作为两拍时序之间的延时时间，调节电位器改变延时时间，从而实现步进电机调速。ADC0809 选用通道零，使用中断法进行数据采集，将 AD_IRQ 连接到集成的主片 8259 的 MIR5 上。

(3)步进电机转向的调节：将 8255 的 PB 端口设为输入端口，将 PB7 接一位开关，读取 PB7 的高低电平状态来确定电机转向,通过改变输入相序的次序来实现改变电机的转向。

(4)显示部分：采用 LCD12864 和数码管对电机状态信息进行实时显示。将 PA 端口的高三位连接到 12864 的三个控制脚上，用于控制 LCD 使能、读写数据、以及控制输入信号为数据还是命令；将 8255 的 PC 端口设为输出端，将八个输出端接到 12864 的数据线上，用于写入数据和命令。

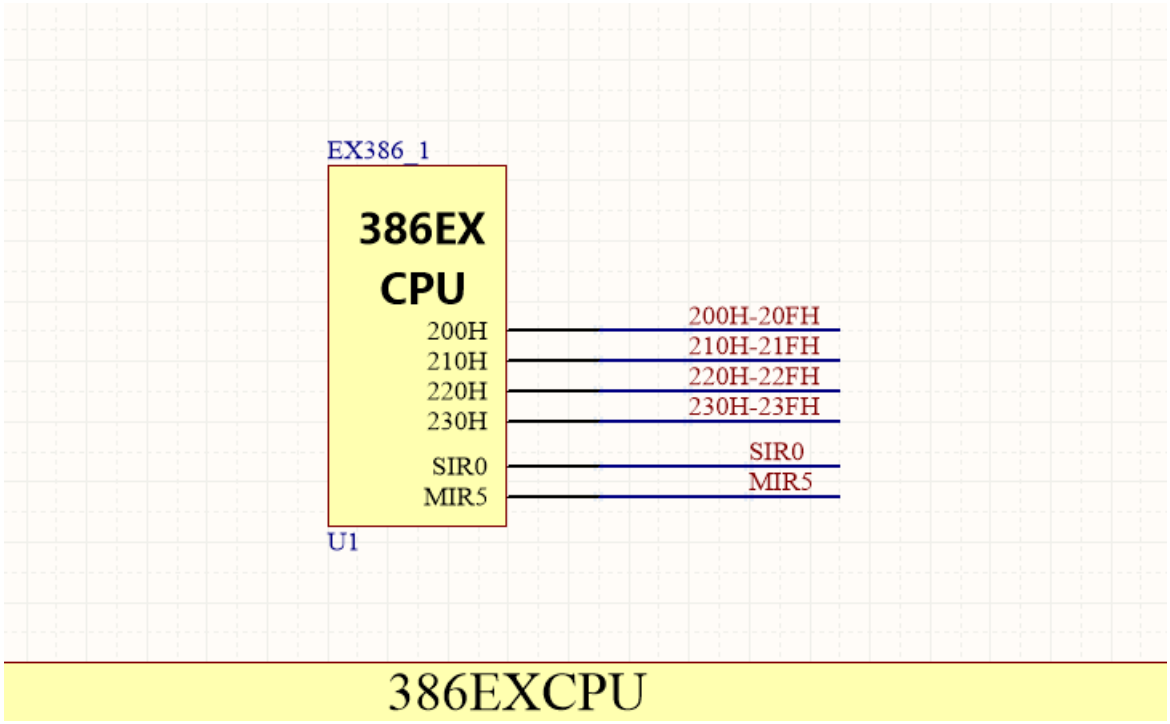
采用定时中断的方式对 12864 显示的速度和方向信息进行更新，通过 8254 芯片对 1MHz 脉冲进行两次分频，第一次使用 CNT1 并将其设置为方式 3，进行 10000 分频得到 100Hz 脉冲，再送入 CNT2，设置为方式 3，进行 20 分频，得到 5Hz 脉冲，将该脉冲连接到集成的从片 8259 的 SIR0 上，产生定时中断，对数据进行更新。

对于 LED 数码管，采用 CPU 模式，只需连接片选 CS 信号至地址开放译码端口，之后对指定的地址输出位选和段码，即可点亮数码管，通过一位数码管来显示电机转速档位。

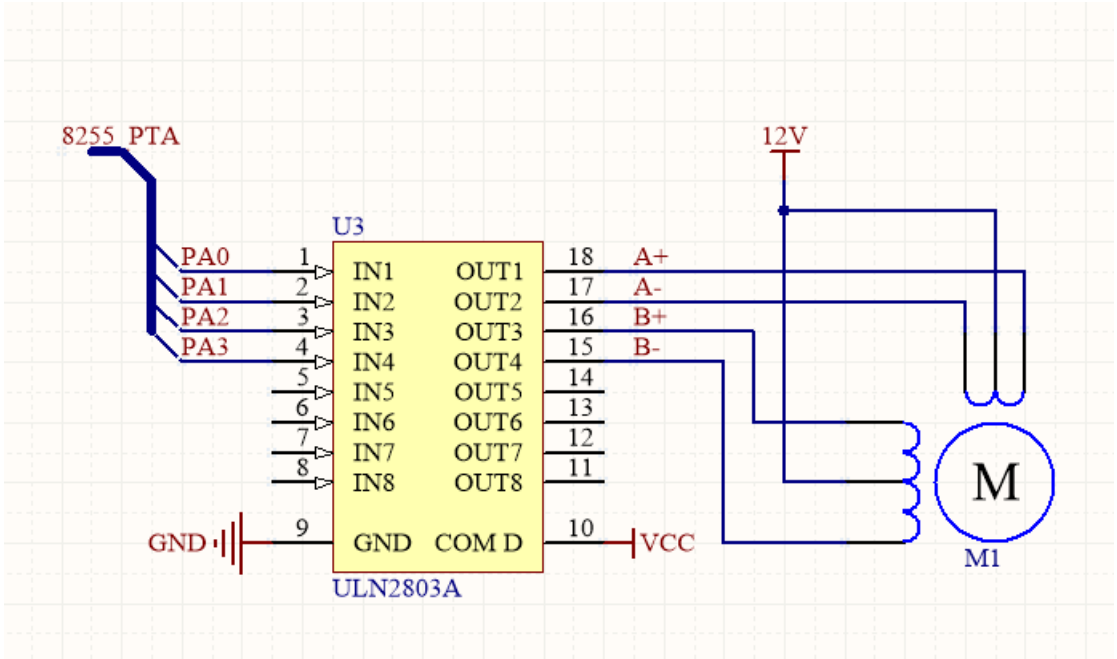
(5)报警部分：监控电机转速，当电机转速达到最高(8 档)或最低(1 档)时，通过 8255 的 PA4 端口输出高电平，使蜂鸣器报警，同时将该信号接到 LED 点阵上，使得 LED 点阵点亮报警。

3　系统电路图

系统总体电路图如下(电路图用 Altium Designer 绘制)



386EXCPU



步进电机驱动电路

4　外围接口模块硬件电路功能描述

(1) ADC0809

将 ADC0809 的选择控制输入线 ADDA、ADDB、ADDC 接地，选中通道零(IN0)作为输入，与电位器相连，将电位器上的电压信号转化为数字量。调节电位器，改变输入电压，输出相应的数字量，利用该数字量改变两拍时序之间的时间间隔，控制步进电机转速。ADC0809 引脚图如下：

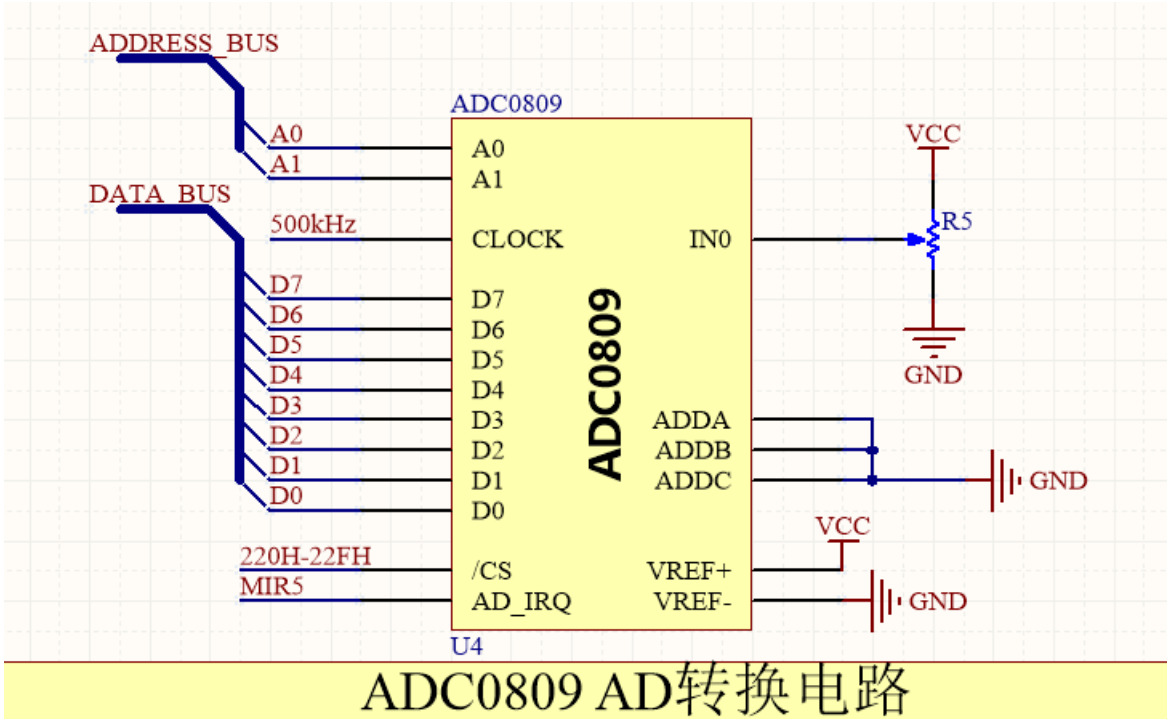


图 4.1 ADC0809 引脚图

(2) 8253

可编程定时器芯片，将门控输入信号线 GATE1 和 GATE2 接高电平，选中 CNT1 和 CNT2，使 CNT1 工作于方式 3、BCD 码计数，CLK1 接 1MHz 脉冲，对 1MHz 脉冲进行 10000 分频，得到 100Hz 脉冲；将 OUT1 接到 CLK2 中，CNT2 工作于方式 3、BCD 码计数，对 CLK2 进行 20 分频，得到 5Hz 脉冲，该脉冲连接到 SIR0 上，产生定时中断。

8253 引脚图如下：

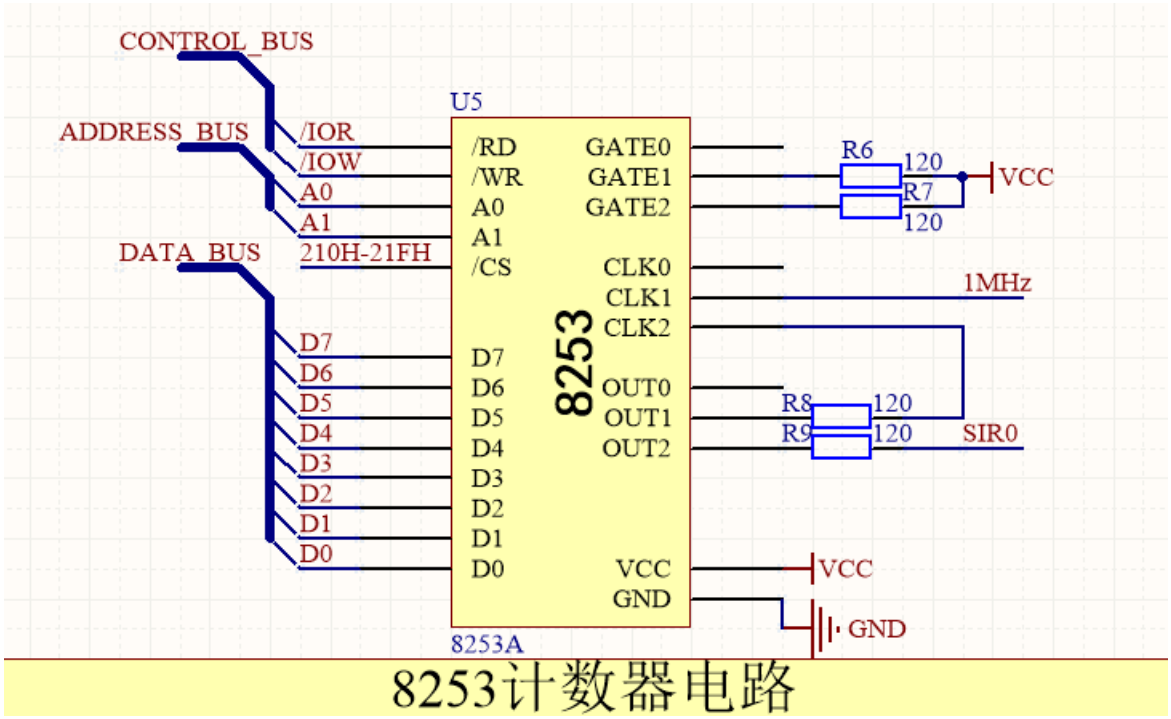


图 4.2 8253 引脚图

(3) 8255

可编程并行接口芯片，将 8255 的 PA 和 PC 端口作为输出端口，其中 PA 端口的低四位输出步进电机时序信号、高三位接到 LCD12864 的三个控制引脚、PA4 接到蜂鸣器上，PC 端口接 12864 的八路数据线；8255 的 PB 端口作为输入端口，PB7 接一位开关，检测开关状态。8255 引脚图如下：

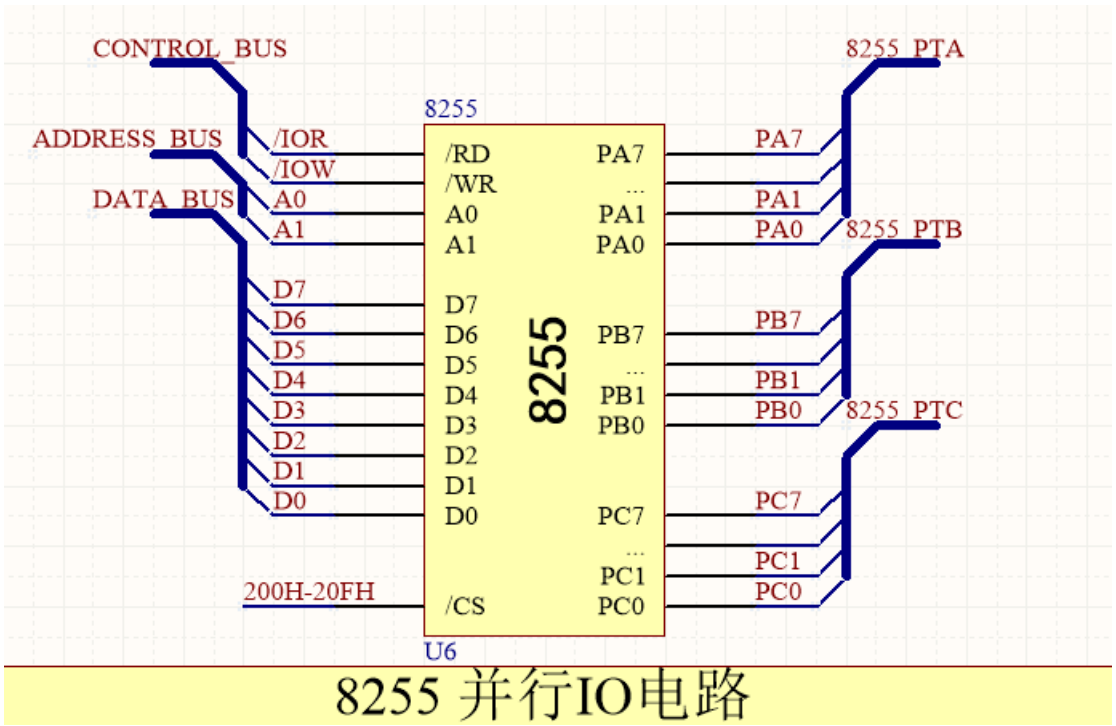


图 4.3 8255 引脚图

(4) LCD12864

显示模块电路，RS(D/I)为数据/命令控制引脚，高电平时为数据，低电平时为命令；R/W，读写控制引脚，高电平时为读操作，低电平时为写操作；E 为使能端，高电平有效；D0-D7 为并行数据线，RS 为高电平时，D0-D7 为数据，RS 为低电平时，D0-D7 为命令。系统通过 LCD12864 显示电机转速和转向等信息，采用中文字符显示，中文字符参考 ST7920 中文字库。LCD12864 引脚图如下：

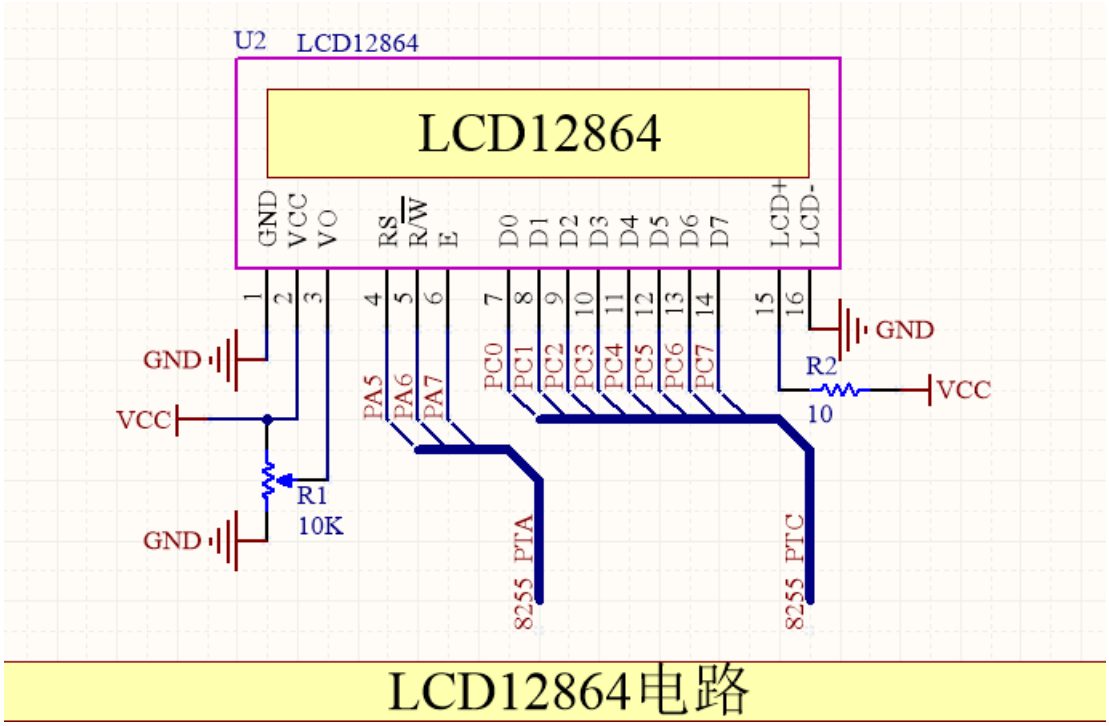


图 4.4 LCD12864 引脚图

5 主程序中主要变量说明

(1)芯片使用地址说明

变量名称	内存单元	功能
IO_8255	200h	8255 端口地址
IO_8254	210h	8254 端口地址
AD_IO	220h	ADC0809 地址
LED_IO	230h	数码管地址

(2)变量使用说明

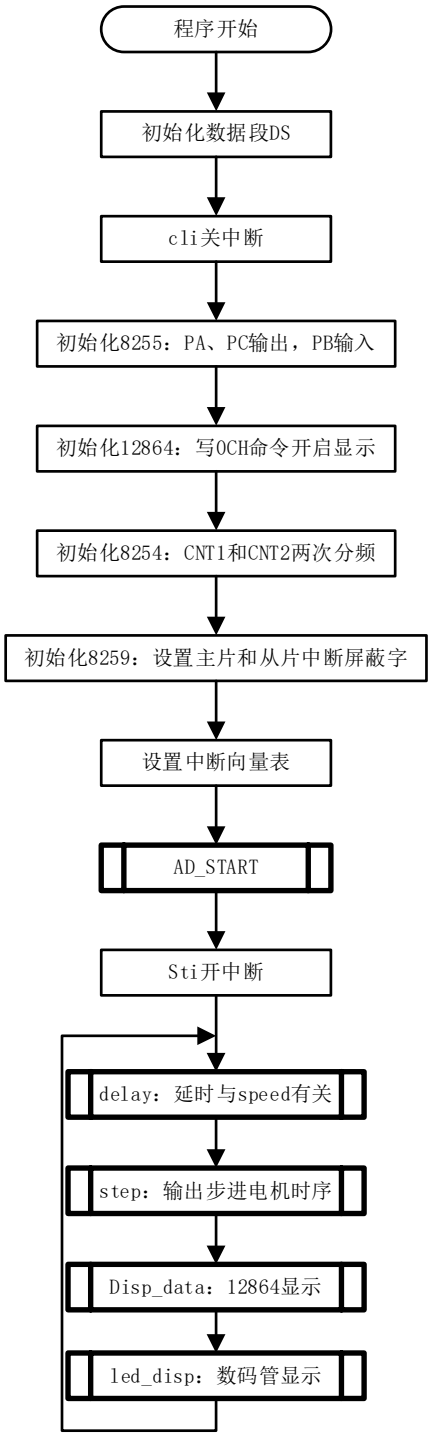
变量名称	内存单元	功能
stepper	db	共 8 字节，对应步进电机的相序代码
speed	db	步进电机的速度，两拍之间的延时时间
dir	db	步进电机的方向

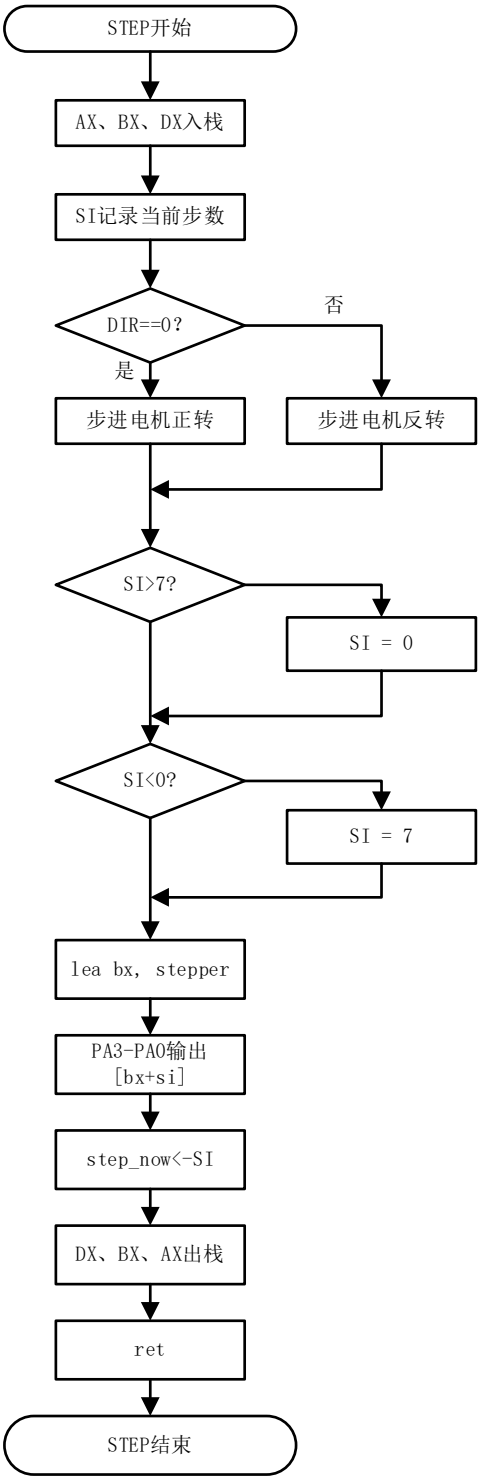
step_now	db	指示当前的一拍时序，范围 0-7
AD_result	db	AD 转换结果
led_hex	db	数码管 0-8 的字型码
led_dat	db	数码管要显示的数值
delayms	db	延时基准量，用于两次操作 LCD 之间
LCD_CMD	db	向 LCD 写入的命令字节
LCD_DAT	db	向 LCD 写入的数据字节
LCD_WORD	dw	向 LCD 写入的中文编码
PTA_temp	db	暂存当前 8255PA 端口的输出
MSG1	db	LCD 第一行：“步进电机-常海颖”
MSG2	db	LCD 第二行：“电自 1704 班”
MSG3	db	LCD 第三行：“方向->速度”
MSG4	db	LCD 第四行：“指导教师-谢老师”

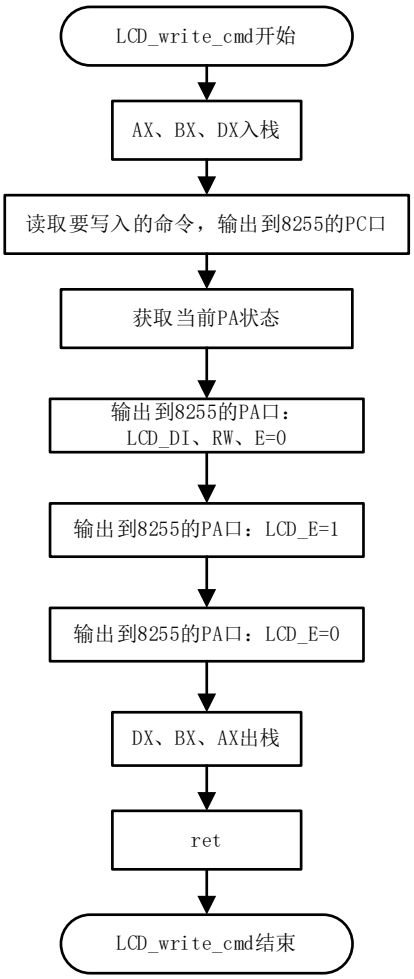
6 系统软件中各个子程序的功能描述

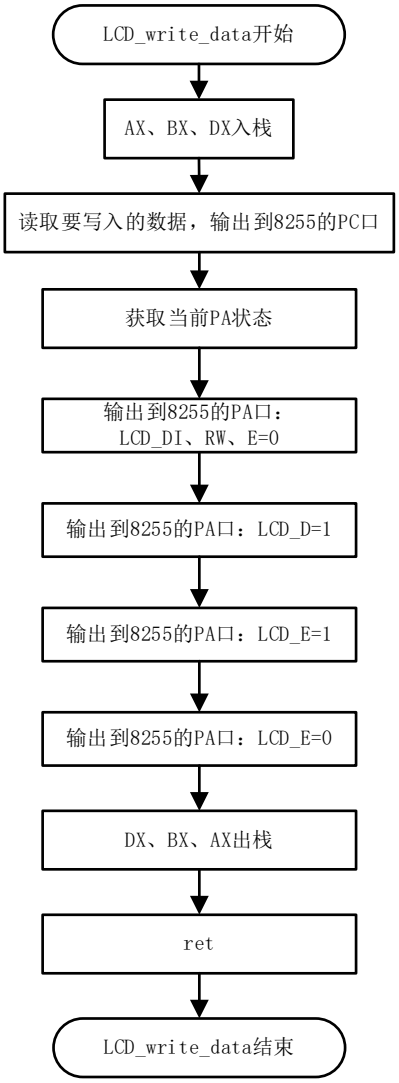
子程序名称	入口参数	出口参数	功能 描述
AD_IRQ_HANDLER	无	AD_result	AD 转换完成中断，读取 AD 转换结果
PIT_IRQ_HANDLER	无	speed,dir	定时中断，对转速和转向更新，并启动 AD 转换
read_switch	无	speed,dir	读取开关状态，确定转向；对 AD 采集的数字量进行转化，转化为两步时序之间的延时常数
AD_START	无	无	启动 AD 转换
delay	无	无	两步时序之间的延时间隔
step	无	stepper	用于输出步进电机的相序
Disp_Line	ax	无	向 LCD 写入一行内容
LCD_write_cmd	LCD_CMD	无	向 LCD 写入 1 个字节数据
LCD_write_data	LCD_DAT	无	向 LCD 写入 1 个字节命令
LCD_write_word	LCD_DAT	无	向 LCD 写入 2 个字节数据
LCD_delay	无	无	LCD 两次写入数据之间
Disp_data	LCD_CMD	无	刷新 LCD 第 3 行的速度和转向值
led_speed	无	无	刷新数码管显示

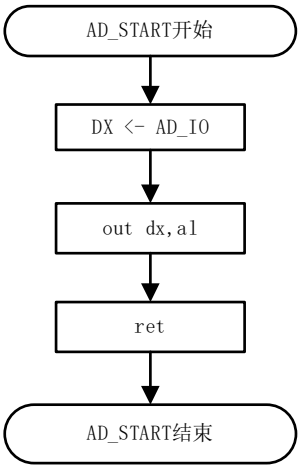
7 主程序程序流程图

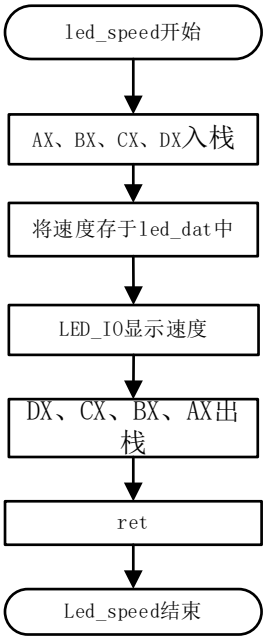

程序开始
初始化数据段DS
cli关中断
初始化8255：PA、PC输出，PB输入
初始化12864：写0CH命令开启显示
初始化8254：CNT1和CNT2两次分频
初始化8259：设置主片和从片中断屏蔽字
设置中断向量表
AD_START
Sti开中断
delay：延时与speed有关
step：输出步进电机时序
Disp_data：12864显示
led_disp：数码管显示

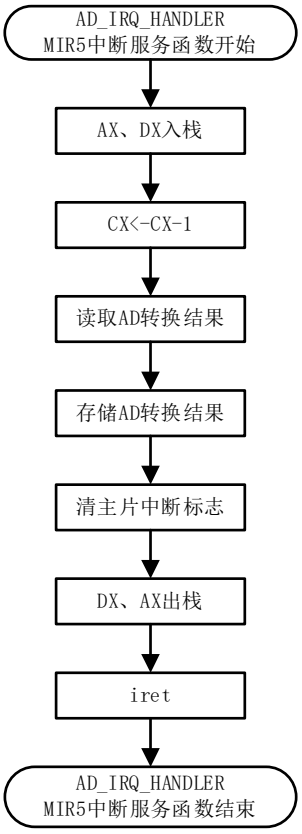

STEP开始
AX、BX、DX入栈
SI记录当前步数
DIR==0?
否
是
步进电机正转
步进电机反转
SI>7?
SI = 0
SI<0?
SI = 7
lea bx, stepper
PA3-PA0输出
[bx+si]
step_now<-SI
DX、BX、AX出栈
ret
STEP结束

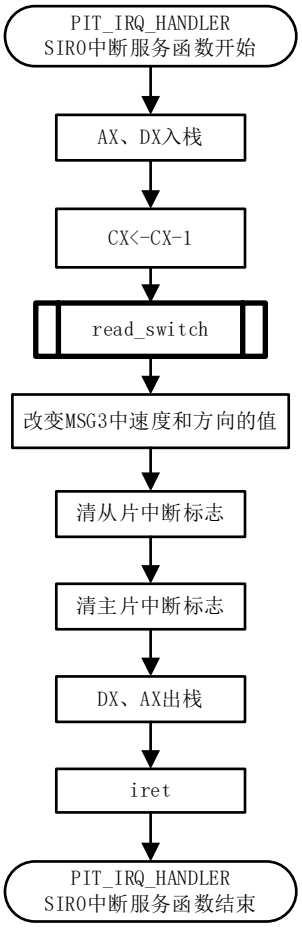

LCD_write_cmd开始
AX、BX、DX入栈
读取要写入的命令，输出到8255的PC口
获取当前PA状态
输出到8255的PA口：
LCD_DI、RW、E=0
输出到8255的PA口：LCD_E=1
输出到8255的PA口：LCD_E=0
DX、BX、AX出栈
ret
LCD_write_cmd结束


LCD_write_data开始
AX、BX、DX入栈
读取要写入的数据，输出到8255的PC口
获取当前PA状态
输出到8255的PA口：
LCD_DI、RW、E=0
输出到8255的PA口：LCD_D=1
输出到8255的PA口：LCD_E=1
输出到8255的PA口：LCD_E=0
DX、BX、AX出栈
ret
LCD_write_data结束


AD_START开始
DX <- AD_IO
out dx,al
ret
AD_START结束


led_speed开始
AX、BX、CX、DX入栈
将速度存于led_dat中
LED_IO显示速度
DX、CX、BX、AX出栈
ret
Led_speed结束





8 程序清单

```
.model small
.386
IO_8255 equ 200h
IO_8254 equ 210h
AD_IO  equ 220h
LED_IO equ 230h

data segment
    stepper    db  01H,03H,02H,06H,04H,0CH,08H,09H ;八拍时序
    speed      db  50                              ;步进电机速度
    dir        db  0                               ;步进电机方向
    step_now   db  0                               ;指示当前的一拍时序
    AD_result  db  0                               ;AD 转换的数字量
    led_hex        db  3fh,06h,5bh,4fh,66h,6dh,7dh,07h,7fh,6Fh;数码管 0-9 的数
字码
    led_dat        db  0                               ;数码管将要显示的数值
```

```
    delayms    db  0                                    ;延时基准量
    LCD_CMD        db  0                                    ;向 12864 写入的命令
    LCD_DAT        db  0                                    ;向 12864 写入的数据
    LCD_WORD   dw  0                                    ;向 LCD 写入的中文编码
    PTA_temp   db  0                                    ;暂存 8255PA 口当前的输出
    MSG1       dw  0B2BDh,0BDF8h,0B5E7h,0BBFAh,0A1AAh,0B3A3h,0BAA3h,0D3B1h ;
步进电机-常海颖
    MSG2       dw  0B0A0h,0B5E7h,0D7D4h,0A3B1h,0A3B7h,0A3B0h,0A3B4h,0B0E0h ;
电自 1704 班
    MSG3       dw  0B0A0h,0B7BDh,0CFF2h,0A1FAh,0CBD9h,0B6C8h,0A3B0h,0B0A0h
    ;MSG3+6 方向 0A1FAh MSG3+12 速度
    MSG4       dw  0D6B8h,0B5BCh,0BDCCh,0CAA6h,0A1AAh,0D0BBh,0C0CFh,0CAA6h ;
指导教师-谢老师
data ends

ss_ssg segment
    dw  50 dup(0)
ss_ssg ends

code segment
    assume cs:code,ds:data,ss:ss_ssg
START:
    cli                    ;关中断
    mov ax,data
    mov ds,ax
;8255 初始化
    mov dx,IO_8255+3       ;8255 控制口
    mov al,82H             ;10000010bA 输出 B 输入
    out dx,al

;lcd 初始化
    lea bx,LCD_CMD
    mov byte ptr [bx],0Ch    ;开显示
    call LCD_write_cmd
    mov ax,1
    call Disp_line           ;显示第一行
```

```
    mov ax,2
    call Disp_line          ;显示第二行
    mov ax,3
    call Disp_line          ;显示第三行
    mov ax,4
    call Disp_line          ;显示第四行

;step_now init
    lea bx,step_now
    mov byte ptr [bx],0

;8254 初始化
    mov dx,IO_8254+3        ;8254 control 213h
    mov al,01110111b        ;CNT1 BCD 计数 16bit 方式 3(方波)
    out dx,al
    mov dx,IO_8254+1        ;CNT1 ,211h
    mov al,0                ;10000 分频,得到 100Hz
    out dx,al
    out dx,al               ;BCD0
    mov dx,IO_8254+3        ;213h
    mov al,10110110b        ;CNT2 二进制计数 16bit 方式 3
    out dx,al
    mov dx,IO_8254+2        ;CNT2,212h
    mov ax,20;200           ;20 分频,得到 5Hz
    out dx,al
    mov al,ah
    out dx,al
;CLK1-1MHz,OUT1-100Hz-CLK2,OUT2-0.5Hz

;设置中断屏蔽字
;主片
    in  al,21h
    and al,11011011b        ;MIR5(AD) MIR2(SIRO)
    out 21h,al
;从片
    in  al,0a1h
```

```
    and al,11111110b      ;SIR0
    out 0a1h,al
;设置中断向量表
    push   ds
    mov ax,0
    mov ds,ax
    lea ax,cs:PIT_IRQ_HANDLER    ;AX 指向中断程序入口地址，进入定时中断
    mov si,70h                ;中断类型码,SIR0
    add si,si
    add si,si
    mov ds:[si],ax            ;中断向量表 IP
    push   cs
    pop    ax
    mov ds:[si+2],ax          ;中断向量表 CS
    ;AD
    mov ax,0
    mov ds,ax
    lea ax,cs:AD_IRQ_HANDLER ;AX 指向中断程序入口地址
    mov si,35h                ;IRQ5(MIR5)
    add si,si
    add si,si
    mov ds:[si],ax            ;中断向量表 IP
    push   cs
    pop    ax
    mov ds:[si+2],ax          ;中断向量表 CS
    pop    ds
    call   AD_START
    sti                       ;开中断

RUN:
    call   delay              ;延时 50ms
    call   step               ;输出步进电机的相序
    call   Disp_data          ;刷新 LCD 第 3 行的速度和转向值
    call   led_speed          ;刷新数码管显示
    jmp    RUN
```

```
;AD 中断
AD_IRQ_HANDLER proc  far
    push   ax
    push   dx             ;保护现场
    mov dx,AD_IO          ;读 AD
    inc dx
    in  al,dx             ;读入 AD 值
    lea bx,ad_result
    mov [bx],al
    mov al,20h            ;发送中断结束命令
    out 20h,al
    pop    dx             ;恢复现场
    pop    ax
    sti
    iret                  ;中断返回
AD_IRQ_HANDLER endp

;定时中断服务子程序
PIT_IRQ_HANDLER proc far
    push   ax
    push   dx             ;保护现场
    call   read_switch       ;读取速度方向
    lea bx,dir            ;更新方向信息
    mov al,[bx]
    lea bx,MSG3              ;指向 MSG3
    cmp al,0              ;判断步进电机转向
    jz  LEFT
    mov word ptr [bx+6],0A1FBh   ;右
    jmp RIGHT
LEFT:
    mov word ptr [bx+6],0A1FAh   ;左
RIGHT:
    mov al,20h            ;发送中断结束命令 SIR0
    out 0a0h,al
    mov al,20h            ;发送中断结束命令 MIR2(SIR0)
    out 20h,al
```

```
    call    AD_START        ;启动 AD 转换
    pop     dx              ;恢复现场
    pop     ax
    sti
    iret                    ;中断返回
PIT_IRQ_HANDLER endp

;速度读取处理
read_switch proc
    push    ax
    push    bx
    push    cx
    push    dx              ;保护现场
    mov dx,IO_8255+1        ;读取 8255 的 B 口
    in  al,dx               ;B7 送 dir
    and al,80h              ;10000000b 取最高位作为方向

    lea bx,dir
    mov [bx],al

    lea bx,ad_result
    mov al,[bx]                 ;取 ad_result
    shr  al,1               ;右移一位 除以 2
    shr  al,1               ;右移一位 除以 2
    shr  al,1               ;右移一位 除以 2 得 0-31
    add al,5                ;取值 5-36
    lea bx,speed            ;电位器调速
    mov [bx],al
    dec al
    shr al,1                ;右移一位 除以 2
    shr al,1                ;右移一位 除以 2
    mov ah,9
    sub ah,al
    mov al,ah               ;al 取值 1-8
    mov ah,0
    lea  bx,led_dat             ;数码管数值
```

```
    mov [bx],al
    lea bx,MSG3                 ;指向 MSG3
    add ax,0A3B0h
    mov [bx+12],ax          ;修改 12864 的显示速度
    pop     dx              ;恢复现场
    pop     cx
    pop     bx
    pop     ax
    ret
read_switch endp

;启动 AD 转换
AD_START   proc
    mov  dx,AD_IO           ;AD_IO 赋给 dx
    out  dx,al
    ret
AD_START   endp

;延时,通过 speed 值改变两个节拍之间延时时间,调节步进电机速度
delay  proc
    push    ax
    push    cx
    push    dx              ;保护现场
    mov dh,speed            ;通过 speed 值改变两个节拍之间延时时间
x1:
    mov cx,0180h
x2:
    loop    x2
    dec dh
    JNZ x1                  ;!=0
    pop     dx              ;恢复现场
    pop     cx
    pop     ax
    ret
delay  endp
```

```
;步进电机相序输出
step   proc
    push   ax
    push   bx
    push   dx            ;保护现场
    mov ax,0
    lea bx,step_now          ;步进电机当前的一拍时序
    mov al,byte ptr [bx]
    mov si,ax
    cmp dir,0            ;方向
    jz  FORWARD              ;=0
    dec si
    JMP OUTPUT_STEP
FORWARD:
    inc si
OUTPUT_STEP:
    cmp si,0
    JNL NOT_MIN              ;>=,反转循环
    mov si,7
NOT_MIN:
    cmp si,7
    JLE NOT_MAX              ;<=,正转循环
    mov si,0
NOT_MAX:
    mov dx,IO_8255
    lea bx,stepper
    mov al,[bx+si]           ;输出一个节拍
    out dx,al
    lea bx,PTA_temp
    mov [bx],al
    lea bx,step_now
    mov ax,si
    mov byte ptr [bx],al

    lea bx,led_dat
    mov ah,[bx]
```

```
    cmp ah,7                ;判断是否为最快速度
    JNGE   NON_ALARM_HIGH
    JNLE   ALARM_HIGH
NON_ALARM_HIGH:
    JMP OUTPUT1
ALARM_HIGH:
    or  PTA_temp,00010000b
    JMP OUTPUT1
OUTPUT1:
    mov dx,IO_8255          ;速度过快蜂鸣器报警
    mov al,PTA_temp
    out  dx,al

    cmp ah,2                ;判断是否为最低速度
    JNLE   NON_ALARM_LOW
    JNGE   ALARM_LOW
NON_ALARM_LOW:
    JMP OUTPUT2
ALARM_LOW:
    or  PTA_temp,00010000b
    JMP OUTPUT2
OUTPUT2:
    mov dx,IO_8255          ;速度过慢蜂鸣器报警
    mov al,PTA_temp
    out  dx,al
    pop    dx               ;恢复现场
    pop    bx
    pop    ax
    ret
step   endp

;LCD 显示第 N 行
Disp_line  proc
    push   bx
    push   cx
    push   dx               ;保护现场
```

```
    cmp ax,4
    jz  LINE4
    cmp ax,3
    jz  LINE3
    cmp ax,2
    jz  LINE2
LINE1:
    mov byte ptr [bx],80h    ;指向第一行
    call LCD_write_cmd
    lea si,MSG1
    jmp DISP
LINE2:
    mov byte ptr [bx],90h    ;指向第二行
    call LCD_write_cmd
    lea si,MSG2
    jmp DISP
LINE3:
    mov byte ptr [bx],88h    ;指向第三行
    call LCD_write_cmd
    lea si,MSG3
    jmp DISP
LINE4:
    mov byte ptr [bx],98h    ;指向第四行
    call LCD_write_cmd
    lea si,MSG4
DISP:
    lea bx,LCD_WORD
    mov cx,8
PUT:
    mov ax,[si]
    add si,2
    mov word ptr [bx],ax
    call LCD_write_word
    loop PUT
    pop    dx              ;恢复现场
    pop    cx
```

```
    pop    bx
    ret
Disp_line  endp

;LCD 写入相关子程序
;写入 LCD_CMD 的指令
LCD_write_cmd proc
    push   ax
    push   bx
    push   dx                ;保护现场
    lea bx,LCD_CMD
    mov al,[bx]
    mov dx,IO_8255+2
    out dx,al                ;PTC 输出
    mov dx,IO_8255
    lea bx,PTA_temp
    mov al,[bx]
    and al,00011111b         ;A6A7 置 0,写命令 I,W
    out dx,al
    or  al,10000000b         ;E=1 开使能
    out dx,al
    lea bx,delayms
    mov byte ptr [bx],1      ;延时 1ms
    call LCD_delay
    and al,01111111b         ;E=0
    out dx,al
    pop    dx                ;恢复现场
    pop    bx
    pop    ax
    ret
LCD_write_cmd    endp

;写入 LCD_DAT 数据
LCD_write_data proc
    push   ax
    push   bx
```

```
    push   dx                ;保护现场
    lea bx,LCD_DAT
    mov al,[bx]
    mov dx,IO_8255+2
    out  dx,al               ;PTC 输出
    mov dx,IO_8255
    lea  bx,PTA_temp
    mov al,[bx]
    and al,01011111b         ;W=0,E=0 关使能
    or  al,01000000b         ;D=1
    out dx,al
    or  al,10000000b         ;E=1
    out dx,al
    lea bx,delayms
    mov byte ptr [bx],1
    call LCD_delay
    and al,01111111b         ;E=0
    out dx,al
    pop    dx                ;恢复现场
    pop    bx
    pop    ax
    ret
LCD_write_data    endp

;写入 LCD_DAT 数据
LCD_write_word proc
    push   ax
    push   bx
    push   cx
    push   dx                ;保护现场
    lea bx,LCD_WORD
    mov ax,[bx]
    lea bx,LCD_DAT
    mov [bx],ah
    call   LCD_write_data
    mov [bx],al
```

```
    call   LCD_write_data
    pop    dx                ;恢复现场
    pop    cx
    pop    bx
    pop    ax
    ret
LCD_write_word    endp

;LCD 延时
LCD_delay proc
    push   ax
    push   cx
    push   dx
    mov dh,delayms           ;延时，用于写入数据之间
L1:
    mov cx,0090h
L2:
    loop   L2
    dec dh
    jnz L1
    pop    dx
    pop    cx
    pop    ax
    ret
LCD_delay  endp

Disp_data proc
    push   ax
    push   cx
    push   dx                ;保护现场
    lea bx,LCD_CMD
    mov byte ptr [bx],88h        ;指向 MSG3
    call LCD_write_cmd
    lea bx,LCD_WORD
    lea si,MSG3
    mov cx,8
```

```
PUTCH:
    mov ax,[si]
    add si,2
    mov word ptr [bx],ax
    call   LCD_write_word
    loop   PUTCH
    pop    dx                ;恢复现场
    pop    cx
    pop    ax
    ret
Disp_data endp

;数码管显示速度
led_speed proc
    push   ax
    push   bx
    push   cx
    push   dx                ;保护现场
    lea bx,led_dat           ;将速度存于 led_dat 中
    mov al,[bx]
    mov ah,0
    mov si,ax
    lea bx,led_hex
    mov dx,LED_IO+1
    mov al,01h               ;个位
    out dx,al
    mov dx,LED_IO
    mov al,[si+bx]
    out dx,al
    pop    dx                ;恢复现场
    pop    cx
    pop    bx
    pop    ax
    ret
led_speed  endp
```

```
code ends
end start
```

9　系统调试运行结果说明、分析所出现得问题，设计体会与建议

硬件环境：386EX 实验箱。

软件环境：Windows 8 386EX 集成开发环境。

设计语言：8086 汇编语言。

实验结果：调节电位器，步进电机的转速发生明显变化；改变开关 SW7 的状态，电机的转向发生变化；将步进电机的信号接到 LED 上，可以看到相序的切换过程；通过 LCD12864 和数码管可以读出当前步进电机的速度档位和转向等信息；当电机速度达到最大(8 档)或最小(1 档)时，蜂鸣器会发出报警声，同时 LED 点阵点亮报警。实验结果与预期相近，基本完成了预期的要求。

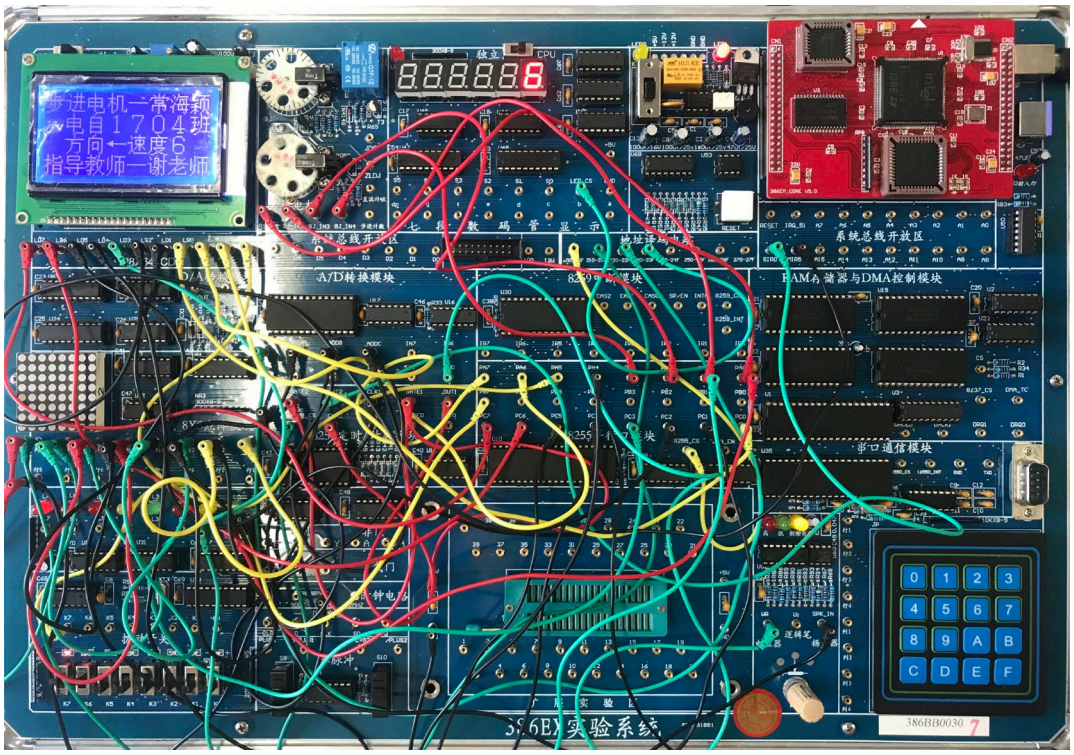
实验遇到的问题：

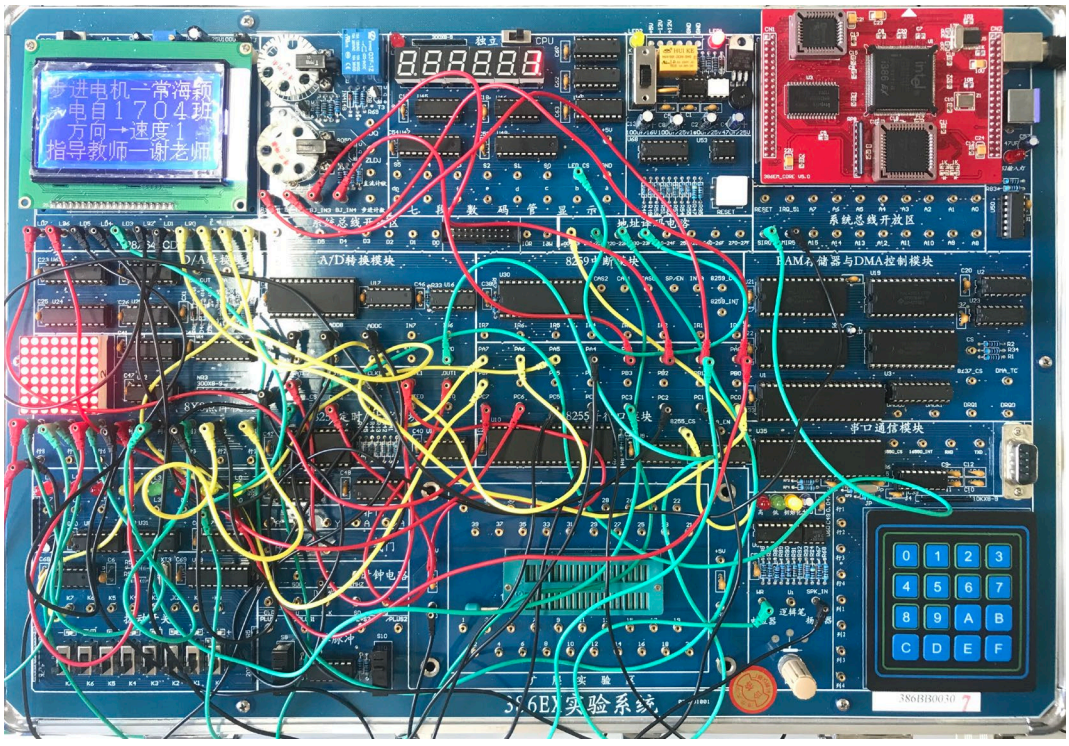
(1)刚开始遇到了很棘手的问题是中断问题，在单步调试的过程中，我发现中断只执行了一次，通过逻辑笔测量 MIR5 和 SIR0 引脚的中断触发信号，发现中断触发信号一直存在，但是 8259 却不能响应中断。通过查找实验讲义的例程，我发现在 8259 级联的时候，要特别注意退出中断的方式，尤其是中断在从片（SIR0）上时，当从片响应中断后会触发主片产生中断，所以在中断结束发送 EOI 命令的时候，要分别向主片和从片发送 EOI 命令，来通知两片 8259 清空中断标志位。此前我只是向从片发送了 EOI 命令，所以在此之后，主片不再响应同级别或者低优先级的中断。

(2)使用中断法进行 AD 转换，调速偶尔会停止响应，分析认为由于中断法 AD 采集只有在每次进入转换完成中断，读取转换结果后才开启下一次 AD 转换，所以一旦丢失一次 AD 采集，就会导致 AD 转换无法启动，使调速停止响应。解决办法是将开启 AD 转换的函数放到定时中断中，AD 转换完成中断只读取转换结果，这样使得 AD 能够在每次定时中断里正常开启，从而保证能够获得转换结果，使调速正常响应。

(3)由于将 LCD 显示和步进电机延时和运动函数都放在主循环中，而 LCD 显示占用了一定时间，导致步进电机转速无法达到实际最快转速。改进方案：应将步进电机运动函数放在定时器中断中，中断时间应稍短，约 1ms 周期，通过计数确定两步之间的间隔，这样能保证步进电机运动频率准确恒定，不受 LCD 影响。

(4)在复杂程序开发中，必然要用到模块化设计思想，此时画流程图是必不可缺的关键步骤，它能帮助开发者记录关键信息，以避免低级错误。





10 结论及设计体会

本次综合设计完成了步进电机调速系统的设计，实现了对步进电机转速和转向的调节，同时添加了数码管和 LCD12864 的显示以及报警功能。实验结果与预期相近，基本完成了实验要求。

本次设计实验不同于之前的实验，需要综合考虑的内容较多，同时需要对系统的整体做出统筹规划，在保证系统能够实现基本功能的同时，还要考虑系统的稳健性，保证

系统的各个部分能够相互协调、互不干扰。由于系统涉及的模块较多，因此需要对各芯片的通道和引脚做出合理的分配，充分利用芯片资源。

虽然汇编语言在今后的工作或者项目训练中不是很常用，但是汇编语言在电类学习中还是非常重要的，因为它可以直接操作 CPU，可以直接访问硬件，这是单片机等做不到的地方，虽然学习起来需要记忆的内容比较多，但是总体上来讲还是非常有意思又有意义的。

通过本次课程设计，我对 8255、8254、ADC0809 等芯片有了更深的了解，同时将理论知识应用于实践当中，对理论知识的理解也更加深刻，收获颇丰。

参考文献

[1]秦晓梅,巢明.计算机原理综合实验教程[M].大连:大连理工大学 电工电子实验中心,2017.