

# 单片机第三四次实验

学号 21160806

姓名 邓 艳 红

# 实验三 步进电机原理及应用

## 原理总结

- (该实验涉及的基本原理及其在实验中的使用方法)
- 1、本实验采用定时器中断实现，使用定时器时，首先应由外部条件得到要定时的时间长度  $t$ ，如本实验中，就是根据要求的速度计算出的每一步之间的间隔。然后选择适当的定时器工作方式，去计算想要设定的计数器初值  $s$ ，使用如下方程。
- $(2 \text{ 定时器最大位数} - s) \times \text{定时周期} = t$
- $\text{定时周期} = 12/\text{CPU 晶振频率}$
- $(2 \text{ 定时器最大位数} - s) \times \text{定时周期} = t$
- 得到的  $s$  需要分成高 8 位和低 8 位，分别放入计数器 THx 和 TLx 中 (x 为 0 或 1)。如果  $s$  为负数，说明需要的定时时间太长，即使定时器的最大时间也无法满足要求。这种情况下，需要加入软件循环才能实现。我们可以将需要的定时时间分成  $n$  份，利用定时器达到  $t/n$  的时间长度，然后在定时器处理程序中，累计某一变量，如果到达  $n$ ，说明总的时间  $t$  已经达到。
- 要想使用定时器中断，除了上面的定时器初值设定外，还需要将其他相关的特殊功能寄存器也都设置好。如果使用方式 0 和方式 1，不要忘记在计数结束后重新恢复计数器初值。
- 2、我们使用的单片机系统的频率是 12M；步进电机转动一周需要 24 步。
- 本步进电机实验板，使用 FAN8200 作为驱动芯片。CPU 通过如下 4 个引脚与 FAN8200 相连，即：
- | CPU  | FAN8200 |
|------|---------|
| P1.1 | CE1     |
| P1.4 | CE2     |
| P3.2 | IN1     |
| P1.0 | IN2     |
- 3、本实验使用简单的双四拍工作模式即可，这也是 FAN8200 比较方便的工作方式。只要将 CE1 和 CE2 分别置为高，然后 IN1 和 IN2 按照预定的脉冲输出，即 01→11→10→00→01 这个循环构成一个方向旋转的输出脉冲，将此序列翻转，就是相反方向的输出脉冲。
- 4、数码管显示：
- 本开发平台有 3 个数码管，使用串行方式连接在一起，具体电路参见实验原理。要想输出一个字形码，就需要从高位到低位依次向移位寄存器输出 8 个比特。移位寄存器的数据线和时钟线分别接到单片机的 P4.5 和 P4.4 管脚，可以使用 MCS-51 里面的位操作指令进行输出。连续输出 3 个字形，24 个 bit 之后，欲显示的字形将稳定地显示在数码管上，程序可以转而执行其他工作。
- 七段字形的编码方式需要通过实验获得。这些编码作为程序中的常数，使用 DB 命令存放。

在程序中，需要将数值转换为相应的字形编码，可以使用 MOVC 指令来完成。

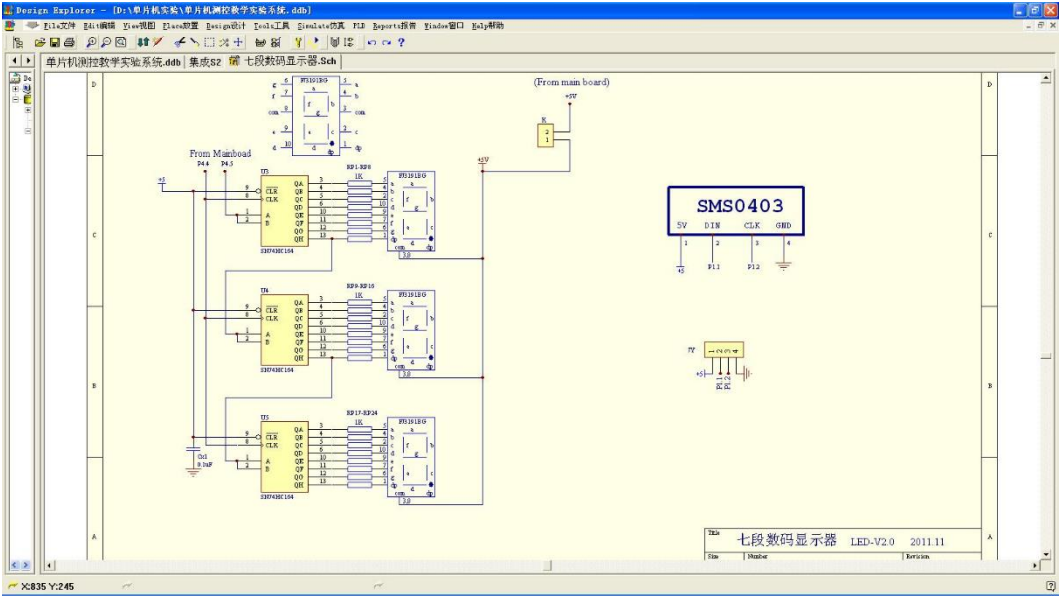
5、74HC164 是高速 CMOS 器件。74HC164 是 8 位边沿触发式移位寄存器，串行输入数据，然后并行输出。数据通过两个输入端（A 或 B）之一串行输入；任一输入端可以用作高电平使能端，控制另一输入端的数据输入。两个输入端或者连接在一起，或者把不用的输入端接高电平，一定不要悬空。

6、时钟（CLK）每次由低变高时，数据右移一位，输入到 Q0，Q0 是两个数据输入端（A 和 B）的逻辑与，它将上升时钟沿之前保持一个建立时间的长度。

7、主复位（CLR）输入端上的一个低电平将使其它所有输入端都无效，同时非同步地清除寄存器，强制所有的输出为低电平。

8、采用 3 个 74HC164 级联控制三个数码管的显示，具体实验原理如下图所示。其中使用单片机 P4.5 作为模拟串口数据，使用 P4.4 模拟串口时钟，CLR 端接高电平。使用上一个 74HC164 的 Q7 作为下一个 74HC164 的输入端。

实验涉及到原理图：



程序分析

（程序设计的思路、程序代码+注释）

```
ORG    0000H
LJMP   START
ORG    000BH ;T0 中断服务程序
LJMP   T0_INT
ORG    0040H
```

START:

```
P4 EQU    0C0H    ;P4 地址
P4SW EQU    0BBH    ;P4 方式控制字地址
;MOV    P4,#0FFH
CLK EQU    P4.4
DATEQU    P4.5
MOV    P4SW,#30H
```

```
SWH1 EQU    P3.6 ;S1
SWH2 EQU    P3.7 ;S2
IN1 EQU    P3.2
IN2 EQU    P1.0
CE1 EQU    P1.3
CE2 EQU    P1.4
```

```
;MOV    SP,#60H
MOV    DPTR,#TABLE
```

```
MOV    R0,#0
MOV    R1,#0
MOV    R2,#0
MOV    R3,#50
MOV    R5,#1
MOV    R6,#1;从 11 开始
```

```
SETB    CE1 ;双四拍工作模式,只要将 CE1 和 CE2 分别置为高
SETB    CE2
SETB    EA ;EA 是整个 CPU 的中断允许标志。当 EA=1 时, CPU 可以响应中断;
SETB    ET0 ;ET1 和 ET0 是 T1 和 T0 的中断允许位
```

```
MOV    TMOD,#01H;T0 计数器, 方式 1
MOV    TL0,#3EH
MOV    TH0,#5DH;计数初值
SETB    TR0 ;运行控制位 TR0 和 TR1 分别控制两个定时器是否允许计数
```

LL1:LJMP LL1

;.....中断服务程序.....

T0\_INT:

```
PUSH    ACC
;PUSH    PSW
;PUSH    DPL
;PUSH    DPH
```

CLRTR0

MOV TL0,#3EH

MOV TH0,#5DH;计数初值

SETB TR0

DJNZ R3,IEND

JNB SWH1,V1;为 0 跳转 (SWH1 按下)

MOV R3,#5;慢速

JMP V2

V1: MOV R3,#1;快速

V2: LCALL DISPLAY;显示步数

LCALL STEP;电机转动

IEND:

;POP DPH

;POP DLH

;POP PSW

POP ACC

RETI

;.....取段码 显示数字.....

DISPLAY:

MOV A,R0

MOVC A,@A+DPTR

LCALL SENDNUM

MOV A,R1

MOVC A,@A+DPTR

LCALL SENDNUM

MOV A,R2

MOVC A,@A+DPTR

LCALL SENDNUM

RET

;.....按位送数.....

SENDNUM:

MOV R4,#8

SE1:CLRCLK

RLCA

```
MOV    DAT,C
SETB   CLK
DJNZ   R4,SE1

RET
```

STEP:

JB SWH2,SHUN;按下，跳转，顺时针

;.....逆时针.....

```
CJNE   R5,#1,N1;R5 不为 1 转移(R5==0)
CJNE   R6,#1,N3;R6 不为 1 转移(R6==0)
CLRIN1;(R5==1,R6==1)
SETB   IN2;送 01
MOV    R5,#0
MOV    R6,#1
LJMP   ST0
```

N1: CJNE R6,#1,N2;R6 不为 1 转移(R6==0)  
CLRIN1;(R5==0,R6==1)  
CLRIN2;送 00  
MOV R5,#0  
MOV R6,#0  
LJMP ST0

N2: SETB IN1;(R5==0,R6==0)  
CLRIN2;送 10  
MOV R5,#1  
MOV R6,#0  
LJMP ST0

N3: SETB IN1;(R5==1,R6==0)  
SETB IN2;送 11  
MOV R5,#1  
MOV R6,#1  
LJMP ST0

;.....顺时针.....

SHUN:

```
CJNE   R5,#1,SH1;R5 不为 1 转移(R5==0)
CJNE   R6,#1,SH3;R6 不为 1 转移(R6==0)
```

```

SETB  IN1;(R5==1,R6==1)
CLRIN2;送 10
MOV    R5,#1
MOV    R6,#0
LJMP   ST0

```

```

SH1:   CJNE   R6,#1,SH2;R6 不为 1 转移(R6==0)
        SETB   IN1;(R5==0,R6==1)
        SETB   IN2;送 11
        MOV    R5,#1
        MOV    R6,#1
        LJMP   ST0

```

```

SH2:   CLRIN1;(R5==0,R6==0)
        SETB   IN2;送 01
        MOV    R5,#0
        MOV    R6,#1
        LJMP   ST0

```

```

SH3:   CLRIN1;(R5==1,R6==0)
        CLRIN2;送 00
        MOV    R5,#0
        MOV    R6,#0
        LJMP   ST0

```

;.....增加步数.....

```

ST0:INC R0
        CJNE   R0,#10,ST1
        MOV    R0,#0
        INC R1
ST1:CJNE   R1,#10,ST2
        MOV    R1,#0
        INC R2
ST2:CJNE   R2,#10,ST3
        MOV    R2,#0

```

```

ST3:RET

```

;.....段码表.....

```

TABLE:
        DB  0C0H,0F9H,0A4H,0B0H,99H,92H,82H,0F8H,80H,90H

```

END

## 问题分析

（实验过程中遇到的问题及解决方法）

1、开始时程序无法正常运行

解决办法：发现 P4 口没有正常定义为接口。加上对 P4SW 接口的正确定义。



## 实验四 LED 点阵显示屏

### 原理总结

（该实验涉及的基本原理及其在实验中的使用方法）

1、高亮度 LED 发光管构成点阵，通过编程控制可以显示中英文字符、图形及视频动态图形。所显示字符的点阵数据可以自行编写（即直接点阵画图），也可从标准字库（如 ASCII6、HZ16）中提取。后者需要正确掌握字库的编码方法和字符定位的计算。

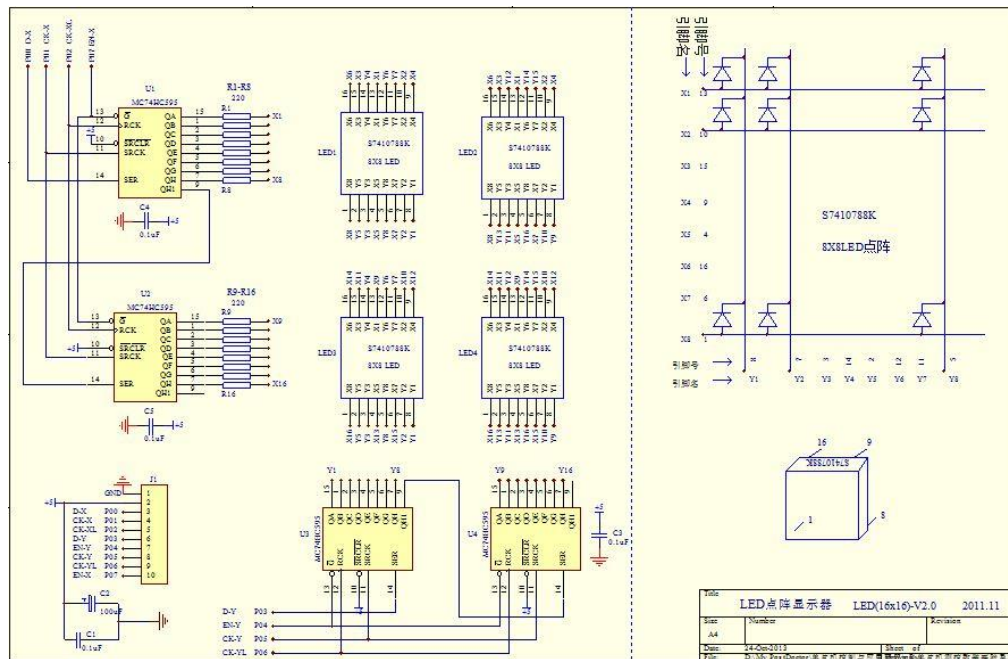
2、实验用的 LED 点阵显示屏为 16\*16 点阵。行和列分别使用两个移位寄存器作为输出。当移位寄存器输出的第  $i$  行为 0，第  $j$  列为 1 时点亮点( $i,j$ )。为了能够显示出一个点阵字型，需要进行循环扫描，也就是每一次只点亮一行，然后在列上输出该列对应的 16 个点阵值。输出一行后暂停一段时间，输出下一行。为了达到较好的显示效果，整屏总的扫描时间不高于 40ms。上述过程中行列可以互换。

3、实验中使用的移位寄存器是 74HC595，它是一个同时具有串行移位和输出锁存驱动功能的器件。74HC595 是具有 8 位移位寄存器和一个存储器，三态输出功能。移位寄存器和存储器是分别的时钟。

4、数据在 SRCK（移位寄存器时钟输入）的上升沿输入到移位寄存器中，在 RCK（存储器时钟输入）的上升沿输入到存储寄存器中去。移位寄存器有一个串行移位输入（行  $Dx$  (P00)、列  $Dy$ (P03)），和一个串行输出（QH），和一个异步的低电平复位，存储寄存器有一个并行 8 位的，具备三态的总线输出，当使能（P02 和 P07 为低电平）时，存储寄存器的数据输出到总线。

5、在控制 74HC595 时，首先将数据放到串行输入的 SI 端，然后在串行时钟 SRCK 上产生一个脉冲，即可输出一个 bit，重复以上步骤 16 次，输出所有列值。然后给存储器时钟 RCK 一个脉冲，将串行数据锁存起来。将使能端 输出低电平，驱动到 LED 点阵上。行的输出每次只移位一次，并重新锁存即可。

**本实验涉及到的电路原理图：**



## 程序分析

(程序设计的思路、程序代码+注释)

```
ORG 000H
LJMP START
ORG 0040H
```

START:

```
DX EQU P0.0 ;行数据口
DY EQU P0.3 ;列数据口
CLKYWX EQU P0.1 ;行移位寄存器时钟
CLKYWY EQU P0.5 ;列移位寄存器时钟
CLKCCX EQU P0.2 ;行存储器时钟
CLKCCY EQU P0.6 ;列存储器时钟
```

```
OUTX EQU P0.7 ;行输出使能
OUTY EQU P0.4 ;列输出使能
```

;.....

SM: ;无限循环

```
MOV R0,#0
MOV R1,#1
```

MOV R4, #1 ;table 高 8 位指针

MOV R5, #0 ;table 低 8 位指针

;.....逐行扫描.....

MOV R3, #16 ;扫描 16 次

SM16: SETB OUTX ;行输出使能置高电平

SETB OUTY ;列输出使能置高电平

;.....送行扫描码.....

CLR CLKCCX ;列存储器时钟置低电平

MOV DPTR, #TABLE1

MOV A, R0

MOVC A, @A+DPTR

MOV R6, #8

YW1: CLR CLKYWX

RLC A

MOV DX, C

SETB CLKYWX ;将高 8 位列选码按位送入到移位寄存器中

DJNZ R6, YW1

MOV A, R1

MOVC A, @A+DPTR

MOV R6, #8

YW0: CLR CLKYWX

RLC A

MOV DX, C

SETB CLKYWX ;将低 8 位列选码按位送入到移位寄存器中

DJNZ R6, YW0

SETB CLKCCX ;将移位寄存器中的数据送到存储器中

CLR OUTX ;将行输出使能置低电平

;.....送列扫描码.....

CLR CLKCCY ;列存储器时钟置低电平

MOV DPTR, #TABLE

```

MOV A,R4
MOVC A,@A+DPTR
MOV R6,#8
YW3:  CLR CLKYWY
      RRC A
      MOV DY,C
      SETB CLKYWY ;将高 8 位列选码按位送入到移位寄存器中
      DJNZ R6,YW3

      MOV A,R5
      MOVC A,@A+DPTR
      MOV R6,#8
YW2:  CLR CLKYWY
      RRC A
      MOV DY,C
      SETB CLKYWY ;将低 8 位列选码按位送入到移位寄存器中
      DJNZ R6,YW2

      SETB CLKCCY ;将移位寄存器中的数据送到存储器中

      CLR OUTY ;将列输出使能置低电平

      LCALL DELAY1

;.....更新扫描码.....

      INC R0
      INC R0
      INC R1
      INC R1

      INC R4 ;TABLE 指针 R4, R5 分别加 2
      INC R4
      INC R5
      INC R5

      DJNZ R3,SM16 ;进行下一次扫描

      LJMP SM ;重新扫描

```

;.....延时函数.....

DELAY1:

MOV

R6,#20

DEL1: MOV R2,#20

DEL2: DJNZ R2,DEL2

DJNZ R6,DEL1

RET

;.....扫描码表.....

TABLE:

邓(0) 艳(1) 红(2)

DB 40H,00H,47H,C2H,44H,41H,44H,42H,7CH,7CH,01H,00H,01H,00H,FFH,FFH;

DB 01H,02H,05H,84H,09H,60H,11H,10H,61H,08H,01H,04H,01H,02H,00H,00H;"邓",0

DB 20H,00H,21H,21H,26H,21H,24H,A2H,F4H,A4H,24H,B8H,34H,A0H,2CH,A0H;

DB 24H,A0H,24H,BEH,F4H,A1H,24H,21H,25H,21H,26H,27H,20H,00H,00H,00H;"艳",1

DB 00H,80H,01H,00H,06H,00H,1FH,FFH,E2H,02H,02H,02H,22H,22H,22H,22H;

DB 22H,22H,FEH,FEH,22H,22H,22H,22H,22H,22H,22H,22H,02H,02H,00H,00H;"红",2

DB 40H,00H,47H,C2H,44H,41H,44H,42H,7CH,7CH,01H,00H,01H,00H,FFH,FFH;

DB 01H,02H,05H,84H,09H,60H,11H,10H,61H,08H,01H,04H,01H,02H,00H,00H;"邓",0

TABLE1:

DB 80H,00H

DB 40H,00H

DB 20H,00H

DB 10H,00H

DB 08H,00H

DB 04H,00H

DB 02H,00H

DB 01H,00H

```
DB 00H,80H
DB 00H,40H
DB 00H,20H
DB 00H,10H
DB 00H,08H
DB 00H,04H
DB 00H,02H
DB 00H,01H
```

```
END
```

## 问题分析

（实验过程中遇到的问题及解决方法）

- 1、在所有的字显示结束后，后面出现一串乱码（一条斜杠或者是）。  
调整“在所有字显示完毕后”控制归零的那一条语句，计算显示三个字码需要的正确长度。
- 2、字码飘过速度太快。  
解决方法：调整延时程序。
- 3、字码显示不清楚  
解决办法：修改试验程序使其能清晰显示。