第三次实验报告

53160818　　8 班　　石方哲

一、实验原理

1. 数码管显示方式

开发平台有 3 个数码管，使用串行方式连接在一起。想要输出 1 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 SN74HC164 的数据线和时钟线分别接到单片机的 P4.5 和 P4.4 管脚，A 和 B 端连接在一起，CLK 由低变高时读入数据并且已读入数据右移一位。连续输出 3 个字形，24 个比特之后，欲显示的字形将稳定的显示在数码管上。***在数码管控制字方面设置 P4SW 为 00110000B，P4SW 的地址为 0BBH。在这种情况下，P4SW.5=1，P4.5 设置成 I/O 口；P4SW.4=1，P4.4 设置成 I/O 口。***

2. 定时器中断

根据题目的要求，步进电机快转时为 60r/min，慢转时为 10r/min 同时单片机的频率为 12M，电机一转 24 步。并且可以由以下公式推导：

$$\begin{cases} t = (2^{\text{定时器最大位数}} - s) * \text{定时周期} \\ \text{定时周期} = \dfrac{12}{CPU\text{晶振频率}} \end{cases}$$

定时器T_0 和T_1 都具有计数方式和定时方式两种工作方式。对定时器T_0 和T_1 在特殊功能寄存器 TMOD 中都有一个控制位$C/\overline{T}$ 来选择T_0 或T_1 为定时或者计数，这个寄存器的地址为 89H。

$$\underbrace{\begin{matrix} 7 & 6 & 5 & 4 & 3 & 2 & 1 & 0 \\ GATA & C/\overline{T} & M_1 & M_0 & GATA & C/\overline{T} & M_1 & M_0 \end{matrix}}_{\text{高四位控制}T_1\text{，低四位控制}T_0}$$

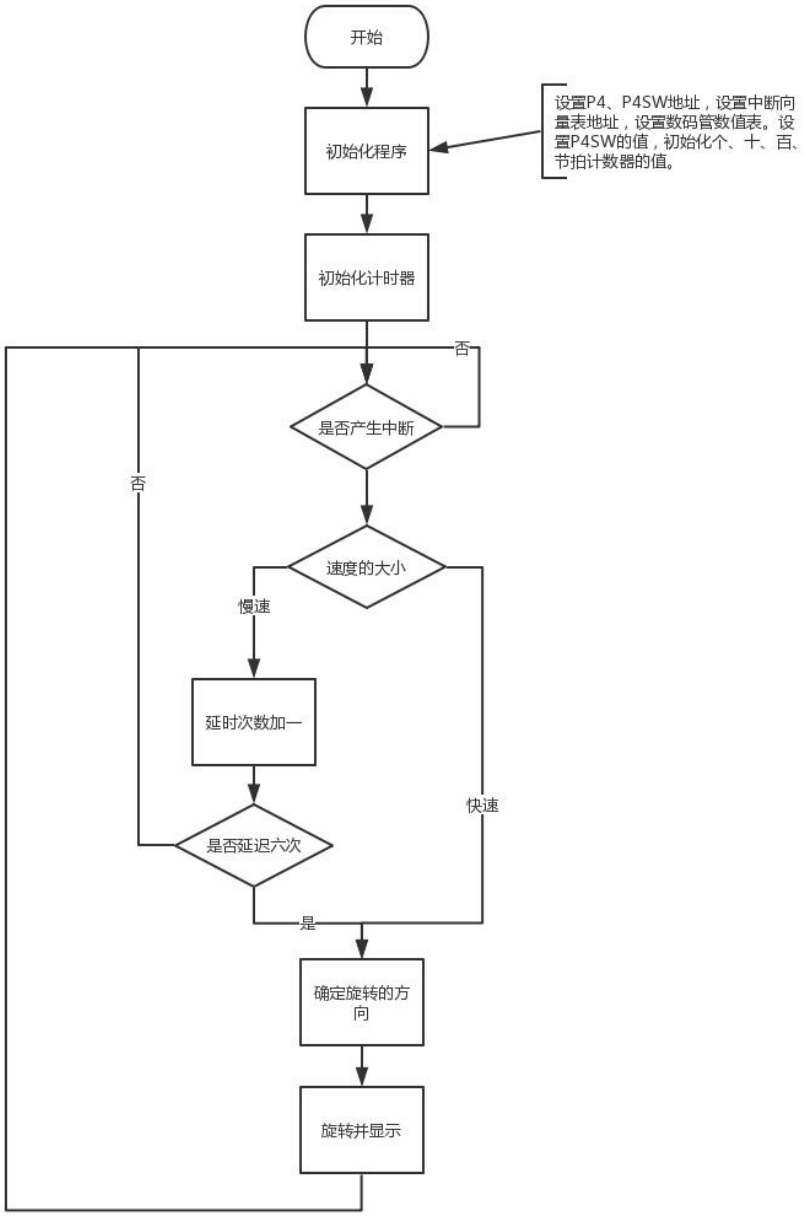
。TR_1 是定时器T_1 的运行控制位。该位由软件置位和清零。当 GATE（TMOD.7）=0，TR_1 =1 时就允许T_1 开始计数，TR_1 =0 时禁止T_1 计数。当 GATE（TMOD.7）=1，TR_1 =1 且$\overline{INT1}$ 输入高电平时，才允许T_1 计数。在

程序中使用的是定时器T_1，在设置完控制字的同时还需要打开中断允许控制位ET_1 / EA，具体设置见代码加粗斜体。

3. 步进电机工作原理

步进电机将数字脉冲转换为角位移。实验中的步进电机使用简单的双四拍工作模式。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。

二、实验流程图



三、思考题

1. 如采用单四拍工作模式，每次步进角度是多少，程序要如何修改？

步进电机中一共有 40 个小齿，所以每个小齿之间的角度为 9°，而采用单四拍工作方式，所以每次的步进角度为 9/4=2.25°。

具体的程序的通电方式修改为：A->B->C->D->A。

2. 如采用单双八拍工作模式，每次步进角度是多少，程序要如何修改？

步进电机中一共有 40 个小齿，所以每个小齿之间的角度为 9°，而采用单四拍工作方式，所以每次的步进角度为 9/8=1.125°。

在单八拍工作模式下，具体的程序的通电方式修改为：A->B->C->D->E->F->G->H->A。

在双八拍工作模式下，具体的程序的通电方式修改为：
AB->BC->CD->DE->EF->FG->GH->HA->AB。

3. 步进电机的转速取决于哪些因素？有没有上、下限？

转速主要取决于时钟周期。转速有上限同时也有下限。

4. 如何改变步进电机的转向？

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，即表示步进电机回转向。

5. 步进电机有哪些规格参数，如何根据需要选择型号？

步进电机的规格参数有相数、拍数、保持转矩、步距角、定位转矩、失步、失调角、运动矩频特性。按照实际需要，根据电机的最大速度、定位精度和力矩进行选择。

6. MCS51 中有哪些可存取的单元，存取方式如何？它们之间的区别和联系有哪些？

MCS－51 有五个独立的寻址空间。
64K 字节程序存储器空间（0－0FFFFH）
64K 外部数据存储器空间（0－0FFFFH）
256 字节内部 RAM 空间（0－0FFH）
256 位寻址空间（0－0FFH）
工作寄存器区

（具体内容见*附录三 MCS-51 系统结构与指令系统第三页*）

7. 说明 MOVC 指令的使用方法。

这种方式以 16 位的程序计数器 PC 或数据指针 DPTR 作为基寄存器，以 8 位的累加器 A 作为变址寄存器，基址寄存器和变址寄存器的内容相加作为 16 位的地址访问程序存储器。如：

MOVC A,@A+PC
MOVC A, @A+DPTR

8. MCS51 的指令时序是什么样的，哪类指令的执行时间较长？

MCS51 的时序单位有四个，它们分别是节拍、状态、机器周期和指令周期。MCS51 中的乘法和除法指令是最长的，都需要 4 个机器周期，其他指令都是 1~2 个机器周期。

9.在本实验环境下，能否控制显示数码的亮度？如何实现？

能，通过修改刷新频率。

四、实验过程中的问题和收获

Q1:如何使数码管工作。

A1:除了 **PPT** 课件中所提到的数码管以及移位寄存器 **SN74HC164** 的工作原理外，同时还要设置数码管控制字 **P4SW**。

Q2:如何使用定时器中断

A2:具体原理见报告的实验原理的定时器中断部分。

收获：这是第一次使用 **MCS51** 在单片机平台上编写程序。编写一个完整的程序，第一步要做的就是认真阅读相关文档。拿本次实验来说我草率的按照自己的想法在没有阅读文档的情况下动手编写，结果浪费了大量的时间在调试程序上。第二步，对自己需要编写的程序有一个清晰的认识，不用一上手就从整体编写，可以从小处逐步完善。当实在编不出来时，参考别人的代码也是一个选择。

五、代码

```
P4      DATA 0C0H
P4SW DATA 0BBH;状态寄存器的地址
ORG 0000H;设置程序的起始地址
LJMP START

ORG 001BH   ;定时器 1 中断入口地址  中断向量表的入口地址
LJMP TINT
```

```
ORG 0040H

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

START:
MOV P4SW,#30H;进行寄存器组的选择 一共分为 4 组 psw 的 3，4 位来确定
MOV DPTR,#TABLE ;DPTR 指向数码管首址相当以 ds
MOV R1,#00H ;个位
MOV R2,#00H ;十位
MOV R3, #00H ;百位

MOV R0, #00H ;节拍计数器
MOV R4, #00H ;慢速旋转，较快旋转六倍延时，R4 实现

MOV TMOD,#10H ;设置计数器 1 的计数方式，高四位用来设置计数器 1 Gate=0 不受外部时钟控制，C/T=0 定时模式 12 分频信号每个时钟周期加一，m1m0=01 为 16 位计数器
MOV TH1,#5DH;
MOV TL1,#47H;23884

SETB ET1 ;ET1 计数器 1 的中断允许控制位，允许 置一
SETB EA ;cpu 的总中断的允许位
SETB TR1 ;启动计数器 1 工作 控制计数器 1 的工作

LOOP : AJMP LOOP ;实现页内跳转 我猜测是进行延迟 来保证数据能够全部写入响应的位置

TINT:
MOV C,P3.6 ;读取开关 S1 状态
JNC TT0 ;JNC CY=0 转移，按下开关 1 跳到快速旋转模式
INC R4
CJNE R4,#6,RETURN ;慢速旋转在此有六倍延时 CJNE 比较不相等转移
MOV R4,#00H

TT0:
MOV C,P3.7 ;读取开关 S2 状态
JNC TT1;C 为 0 跳转，可替换为 JB P3.7,0,TT1
ACALL ROTATE1 ;逆时针 会进行为当前程序位置的存储
AJMP RETURN ;AJMP 为短跳转
TT1:
ACALL ROTATE2 ;顺时针

RETURN:
MOV TH1,#5DH
MOV TL1,#47H
```

```
RETI;中断返回指令

;逆时针旋转
ROTATE1:
CLR P1.3
CLR P1.4
CJNE R0,#0,A10
AJMP L10   ;若为 0，转到 L10，输出 01
A10:
CJNE R0,#1,A11
AJMP L11   ;若为 1，转到 LL1，输出 11
A11:
CJNE R0,#2,A12
AJMP L12   ;若为 2，转到 L12，输出 10
A12:
AJMP L13   ;若为 3，转到 L13，输出 00

L10:;the first 01
CLR P3.2 ;keep out the interupt int1
SETB P1.0;int2
INC R0
AJMP EXIT1
L11:
SETB P3.2 ;第二拍，11
SETB P1.0
INC R0
AJMP EXIT1
L12:
SETB P3.2   ;第三拍，10
CLR P1.0
INC R0
AJMP EXIT1
L13:
CLR P3.2      ;第四拍，00
CLR P1.0
MOV R0,#00H

EXIT1:
SETB P1.3
SETB P1.4
ACALL SHOWSTEPS ;显示步数
RET

;顺时针旋转
```

```
ROTATE2:
CLR P1.5
CLR P1.4
CJNE R0,#0,A20
AJMP L20   ;若为 0，转到 L20，输出 01
A20:
CJNE R0,#1,A21
AJMP L21   ;若为 1，转到 L21，输出 00
A21:
CJNE R0,#2,A22
AJMP L22   ;若为 2，转到 L22，输出 10
A22:
AJMP L23   ;若为 3，转到 L23，输出 11

L20:
CLR P3.2   ;第一拍 01
SETB P1.0
INC R0
AJMP EXIT2
L21:
CLR P3.2   ;第二拍 00
CLR P1.0
INC R0
AJMP EXIT2
L22:
SETB P3.2   ;第三拍 10
CLR P1.0
INC R0
AJMP EXIT2
L23:
SETB P3.2   ;第四拍 11
SETB P1.0
MOV R0,#00H

EXIT2:
SETB P1.3
SETB P1.4
ACALL SHOWSTEPS ;显示步数
RET

;显示步数子程序
SHOWSTEPS:
CJNE R1,#9H,LL1
CJNE R2,#9H,LL2
```

```
CJNE R3,#9H,LL3
MOV R1,#0    ;999 则清零
MOV R2,#0
MOV R3,#0
AJMP EXIT3
LL1:
INC R1
AJMP EXIT3
LL2:
MOV R1,#0
INC R2
AJMP EXIT3
LL3:
MOV R1,#0
MOV R2,#0
INC R3
EXIT3:
MOV A,R1 ;显示个位数
MOVC A,@A+DPTR
ACALL SHOW

MOV A,R2 ;显示十位数
MOVC A,@A+DPTR
ACALL SHOW

MOV A,R3 ;显示百位数
MOVC A,@A+DPTR
ACALL SHOW
RET

SHOW:
MOV R7,#8
LL4:
RLC A;move to the left
MOV P4.5,C
CLR P4.4;delay
SETB P4.4
DJNZ R7,LL4
RET

END
```

第四次实验报告

53160818 **8 班** 石方哲

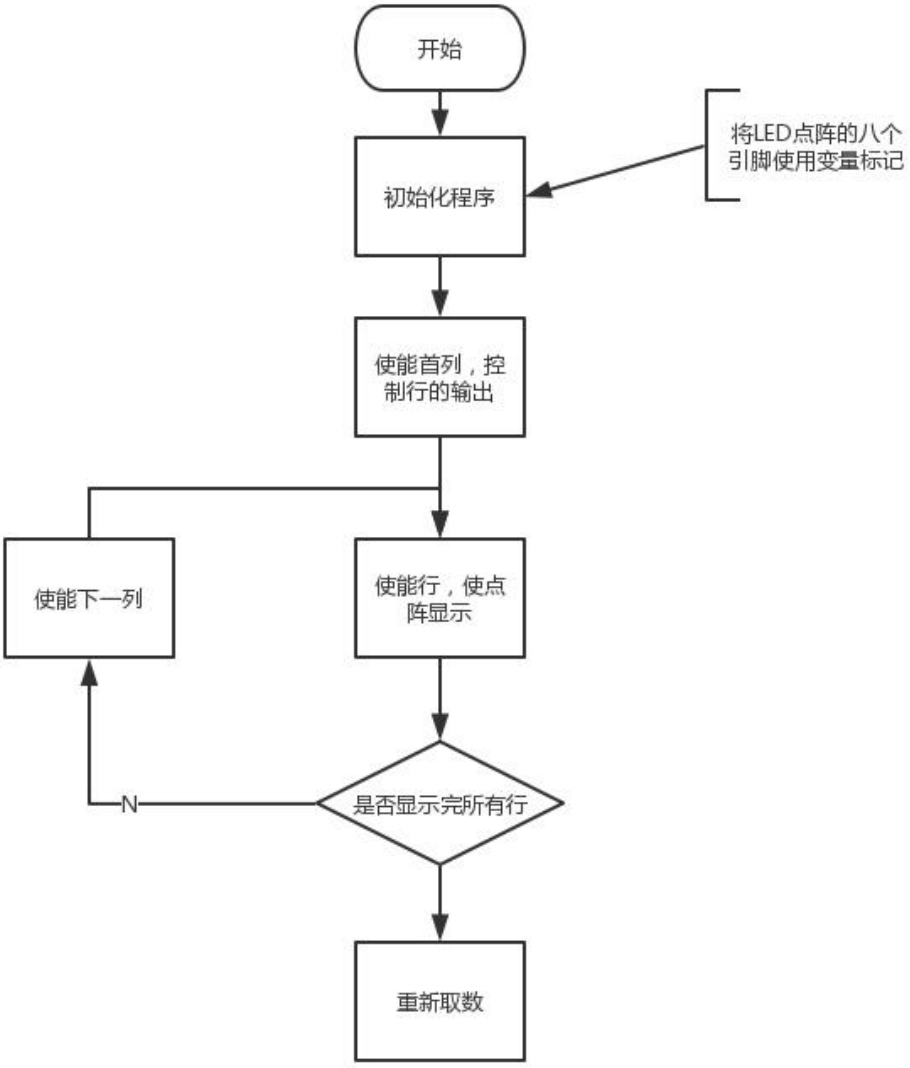
一、实验原理

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。

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。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。

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。其他信息见给定的参考资料。

二、实验流程图



三、思考题

1.如何使用软件调整和控制 LED 点阵的亮度

通过软件调整扫描频率来控制。

2.如何尽量避免显示过程中的闪烁

提高扫描频率。

3.如何将本实验的软硬件推广到多行多列的 LED 显示屏(如 64*1280)

硬件方面：添加新的 LED 以及 74HC59 来实现
软件方面：将控制行扫描的 16 位数字 0fffeH 改为 64 位的 0ffffffffffffffeH，将读入列值的 2 字节改为 160 字节，及重复输出 1280bit，结束后令行的输出移位一次。

四、实验过程中的问题和收获

收获：在本次实验中了解到了 LED 点阵的原理。

五、代码

```
ORG 0000H          ;复位起始地址
LJMP START

ORG 0050H          ;程序起始地址
START:
    DX EQU P0.0   ;P0.0 行串行移位输入
    CKX EQU P0.1  ;P0.1 移位存储器时钟输入
    CKXL EQU P0.2      ;P0.2 存储器时钟输入
    DY EQU P0.3        ;P0.3 列串行输入
    ENY EQU P0.4       ;P0.4 使能端
    CKY EQU P0.5       ;P0.5 移位存储器时钟输入
    CKYL EQU P0.6      ;P0.6 存储器时钟输入
    ENX EQU P0.7       ;P0.7 使能端
S0:
    MOV R3,#22H
    MOV DPTR,#TAB
S1:
；显示本屏
;列置一
    MOV R0,#32
    CLR ENX                ;行使能
    CLR ENY                ;列使能
    CLR CKYL           ;输出存储器锁存时钟清零，制造上升沿
    MOV R2,#15
    SETB C                 ;进位位置一
ROW1:
    CLR CKY                ;数据输入时钟清零，制造上升沿
    MOV DY,C               ;把进位位赋值给数据线
```

```
    SETB CKY            ;数据输入时钟置一
    DJNZ R2,ROW1        ;循环 16 次
    CLR C               ;进位位清零
    CLR CKY
    MOV DY,C
    SETB CKY
    SETB CKYL           ;输出存储器锁存时钟置一
;循环点亮每行
LOOP0:
;输出本行前八位
    MOV A,#32
    CLR C
    SUBB A,R0
    MOVC A,@A+DPTR              ;按规定字符表找到对应行的编码值
      MOV R2,#8
    CLR CKXL            ;输出存储器锁存时钟清零，制造上升沿
    SETB ENX            ;输出不使能
LOOP1:
    RRC A               ;带进位右移，最低位进 c 位
    CLR CKX             ;数据输入时钟线清零，制造上升沿
    MOV DX,C            ;把 C 赋值给数据线 DX
    SETB CKX            ;数据输入数据线置一
    DJNZ R2,LOOP1       ;循环 8 次
;输出后八位
    DEC R0
    MOV A,#32
    CLR C
    SUBB A,R0
    MOVC A,@A+DPTR      ;更新寄存器 A 中的值
    MOV R2,#8
LOOP2:
    RRC A
    CLR CKX
    MOV DX,C
    SETB CKX
    DJNZ R2,LOOP2
    SETB CKXL           ;输出存储器锁存时钟置一，保存到存储寄存器
    CLR ENX             ;输出使能

DEALY:                  ;延迟共循环 80*50
MOV R5,#80
TIME:
MOV R6,#50H
DJNZ R6,$
```

```
DJNZ R5,TIME
COPLAN:
    SETB C
    CLR CKYL
    CLR CKY
    MOV DY,C
    SETB CKY
    SETB CKYL
    DJNZ R0,LOOP0              ;显示本屏 32 行
    INC DPTR          ;加二指向下一行编码
    INC DPTR
    DJNZ R3,S1              ;循环 32 次
    JMP S0                  ;滚动结束后重新滚动
TAB:
;DB 41H,09H,62H,2AH,54H,4DH,0C8H,7FH,54H,8DH,62H,0AH,41H,09H,0C2H,1FH;
;DB 44H,12H,78H,32H,0FFH,5FH,41H,92H,49H,12H,0D9H,1FH,09H,00H,07H,00H;

;DB 00H,00H,02H,00H,44H,08H,88H,08H,10H,09H,0E0H,2AH,80H,04CH,0A2H,7FH;
;DB 0E1H,4CH,0A2H,4AH,24H,89H,0B8H,08H,40H,08H,40H,00H,00H,00H,00H,00H;

;DB 00H,00H,08H,04H,10H,04H,0E0H,7FH,40H,04H,80H,04H,00H,00H,0C0H,1FH;
;DB 40H,12H,40H,12H,0FFH,0FFH,40H,12H,40H,12H,0C0H,1FH,00H,00H,00H,00H;
END
```