单片机实验报告

姓名：冯子英

学号:533160829

实验三 步进电机原理及应用

一、 实验目的和要求

初步学习和掌握 MCS-51 的体系结构和汇编语言，了解 Keil 编程环境和程序下载工具的使用方法。

了解步进电机的工作原理，学习用单片机的步进电机控制系统的硬件设计方法，掌握定时器和中断系统的应用，熟悉单片机应用系统的设计与调试方法。

了解数码管输出的原理及编程方式。

二、 实验设备

单片机测控实验系统，步进电机控制实验模块，Keil 开发环境，STC-ISP 程序下载工具

三、 实验内容

编制 MCS-51 程序使步进电机按照规定的转速和方向进行旋转，并将已转动的步数显示在数码管上。

步进电机的转速分为两档，当按下 S1 开关时，进行快速旋转，速度为 60 转/分。当松开开关时，进行慢速旋转，速度为 10 转/分。当按下 S2 开关时，按照顺时针旋转；当松开时，按照逆时针旋转。

本程序要求使用定时器中断来实现，不准使用程序延时的方式。

四、 实验步骤

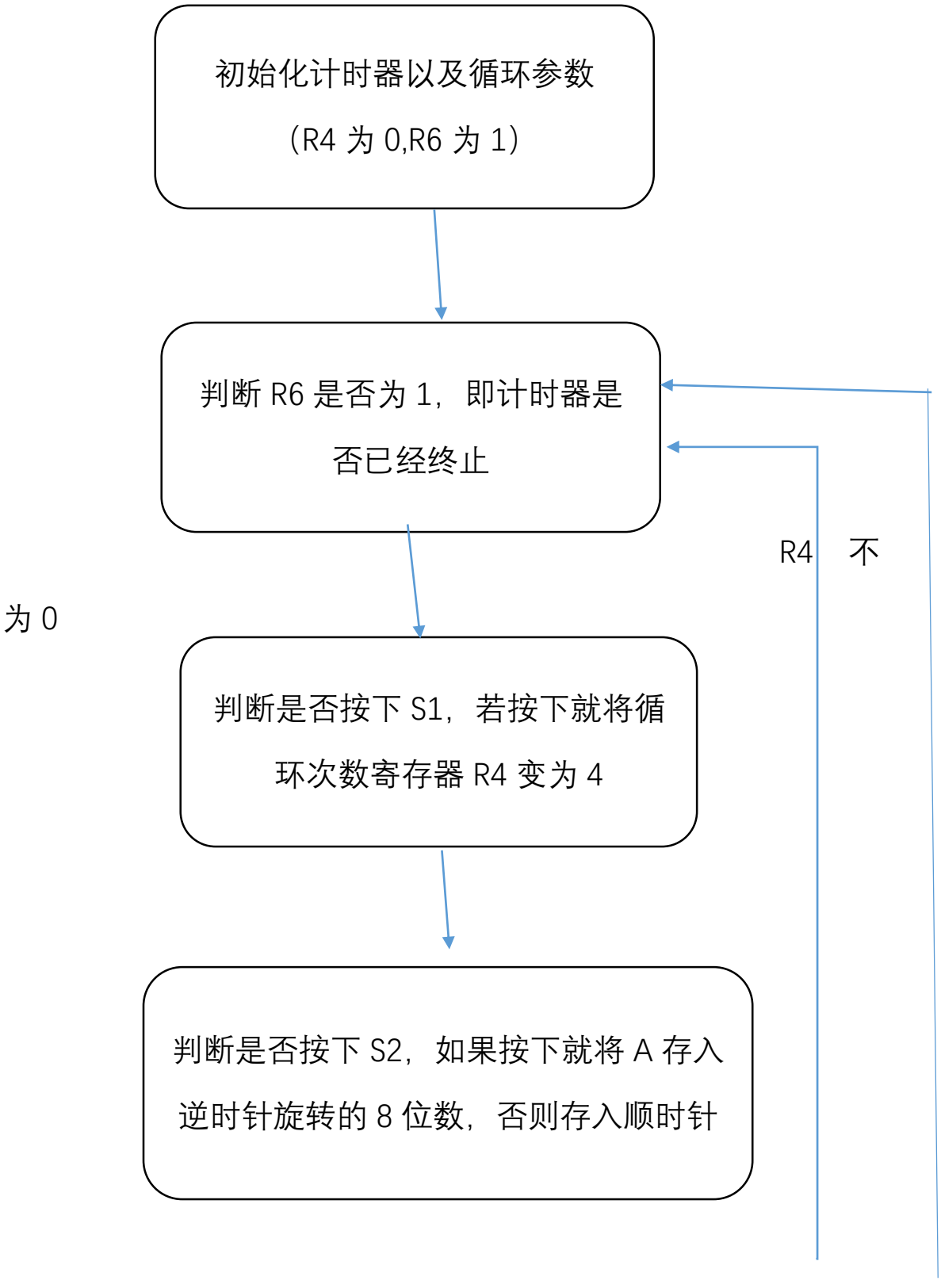
4.1 预习

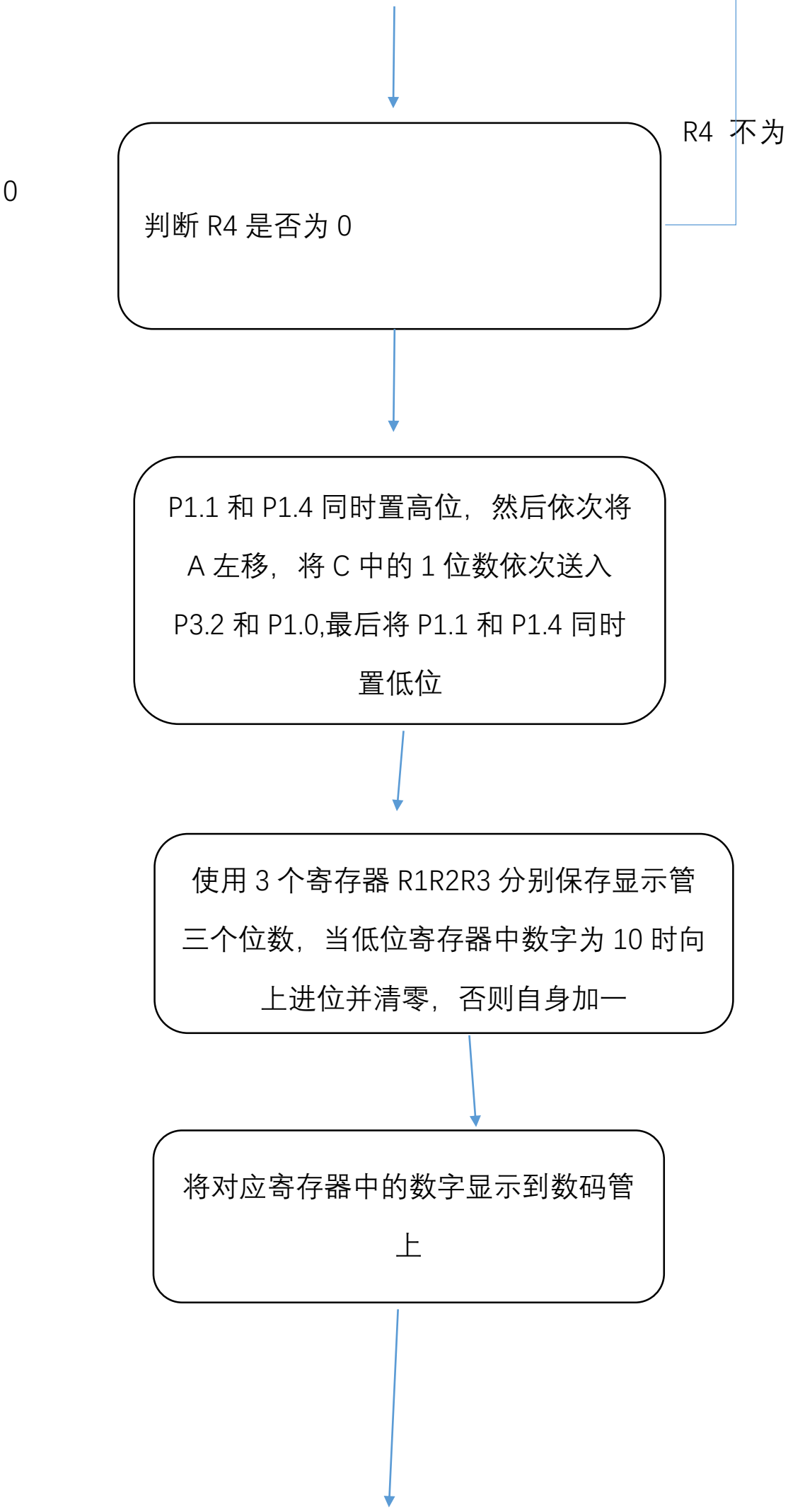
4.2 简单程序录入和调试

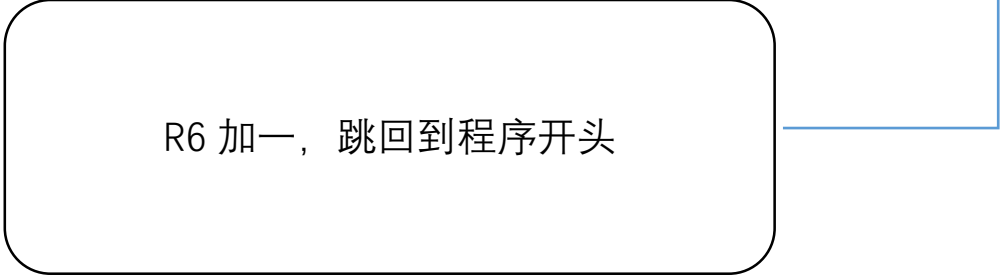
4.3 程序调试

4.4 编写程序，完成功能

五、 程序流程图




R4 不为
0
判断 R4 是否为 0
P1.1 和 P1.4 同时置高位，然后依次将 A 左移，将 C 中的 1 位数依次送入 P3.2 和 P1.0,最后将 P1.1 和 P1.4 同时置低位
使用 3 个寄存器 R1R2R3 分别保存显示管三个位数，当低位寄存器中数字为 10 时向上进位并清零，否则自身加一
将对应寄存器中的数字显示到数码管上



六、程序代码

```
        ORG 0000H
            LJMP START
        ORG 000BH
              LJMP    BREAK
        ORG 0040H
        START:
            P4 EQU 0C0H
            P4SW EQU 0BBH
            MOV IE,#82H
            MOV TMOD,#00000001B
            MOV TCON,#00010000B
            ;MOV R0,#5DH;快速旋转高 4 位
            ;MOV R1,#3DH;快速旋转低 4 位
            ;MOV R2,#0F4H;慢速旋转高四位
            ;MOV R3,#24H;慢速旋转低四位
            MOV TH0,#5DH
            MOV TL0,#3DH
            MOV R5,#00H;循环次数
```

```
    MOV A,#01111000B
    MOV R4,#06H
    MOV R1,#00H;数码管最右
    MOV R2,#00H;数码管次右
    MOV R3,#00H;数码管左
    MOV 25H,#5DH
    MOV 26H,#3DH
    MOV DPTR,#TABLE
    MOV P4SW,#70H
TEST:
MOV R6,#00H;R6 控制是否从死循环跳出
CJNE R5,#04H,RIOT;是否已经旋转了 4 次了，没有则跳转
MOV R5,#00H
TEST1:
    JB P3.6,SET1;是否按下 S1，按下则跳转，变成快速模式
    MOV R4,#06H
    ;已经进行的循环数加 1

TEST2:
    JB P3.7,SET2;;是否按下 S2，按下则跳转，变成逆时
```

针模式

```
        MOV A,#01111000B
        JMP RIOT

    SET2:
        MOV A,#01001011B
        JMP RIOT
    SET1:
        MOV R4,#00H
        JMP    TEST2
    RIOT:
        CJNE R4,#00H,DEAL
        SETB P1.1
        SETB P1.4
        RLC A
        MOV P3.2,C
        RLC A
        MOV P1.0,C
    SHOW:
        MOV 22H,A
        LCALL NUM
        MOV A,22H
```

```
        INC R5
        MOV R4,#06H
        JMP ENDING
    DEAL:
        DEC R4
        JMP ENDING
    ENDING:;死循环，等待计数器终止
          CJNE R6,#01H,ENDING;
        LJMP TEST
    BREAK:;计数器中断处理程序
        MOV TH0,25H
        MOV TL0,26H
        MOV R6,#01H
        RETI
    STOP:
    RET
    NUM:                                ;子程序显示已经转动的
步数
    S0:         MOV A,R1
                CALL EXP
                MOV A,R2
                CALL EXP
```

```
        MOV A,R3
        CALL EXP

        CJNE  R1,#9H,S1     ;R1  与代表的数相同则执
行 S1
        MOV    R1,#00H
        CJNE R2,#9H,S2
        MOV    R2,#00H
        CJNE R3,#9H,S3
        MOV    R3,#00H

S1:     INC R1              ;未满 10 则加一
        LJMP STOP
S2:     INC R2
        LJMP STOP
S3:     INC R3
        LJMP STOP
EXP:
        MOV    21H,R0
        MOVC  A,@A+DPTR     ;给 DPTR 赋表的首地
址值，给 A 数据的偏移量，得到的查询结果放在 A 中
        MOV    R0,#8;连续 8 位赋值
```

```
CLY:    CLR P4.4            ;时钟线低电平
        RLC A               ;累加器 A 左移
        MOV P4.5,C
        SETB P4.4           ;P4.4   ;时钟线高电平
        DJNZ R0,CLY
        MOV   R0,21H;复原 R0
    RET
    TABLE:
    DB
0C0H,0F9H,0A4H,0B0H,99H,92H,82H,0F8H,80H,90H
      END
```

七、问题解决

如果在步进电机旋转结束的时候，不将 CE1 和 CE2 复位，可能会导致步进电机在原地振动，不会旋转

实验四 LED 点阵显示器

一、 实验目的和要求

了解 LED 点阵显示的基本原理和实现方法。掌握点阵汉字库的编码和从标准字库中提取汉字编码的方法。

二、 实验设备

单片机测控实验系统

LED 点阵显示器实验模块

Keil 开发环境

STC-ISP 程序下载工具

三、 实验内容

了解 16*16 点阵电路的原理，编写汇编语言程序。

编写一行汉字字符（至少三个字）的显示程序。

能够从左到右（或从右到左）循环显示（要求显示过程中字的大小与屏幕尺寸相适应）。

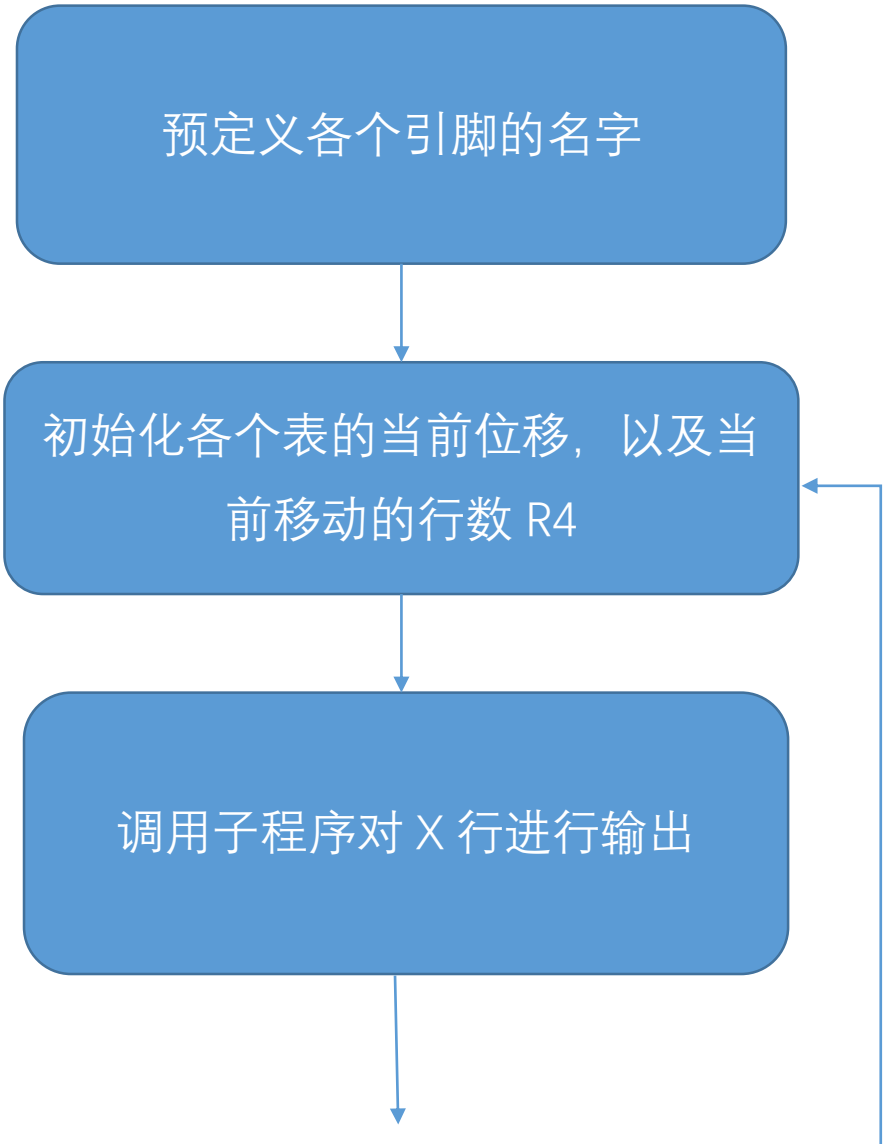
四、 实验步骤

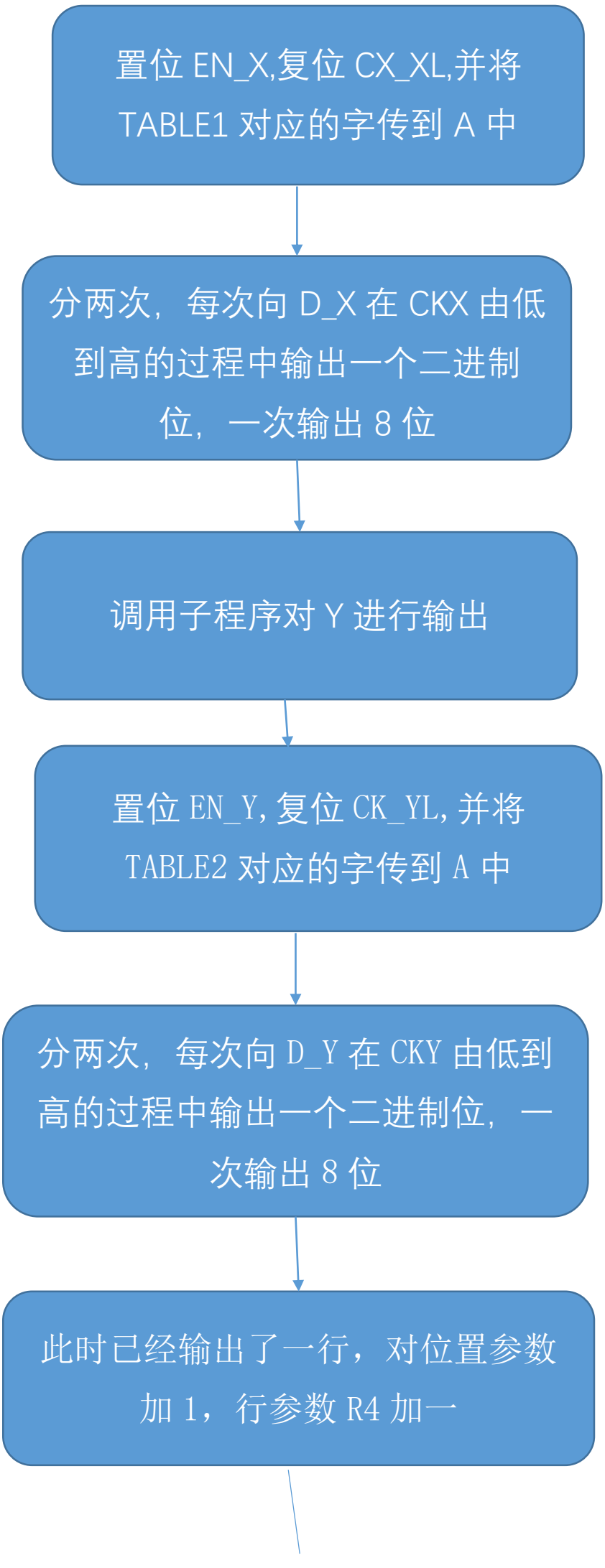
1. 掌握点阵式 LED 显示屏的控制方法;

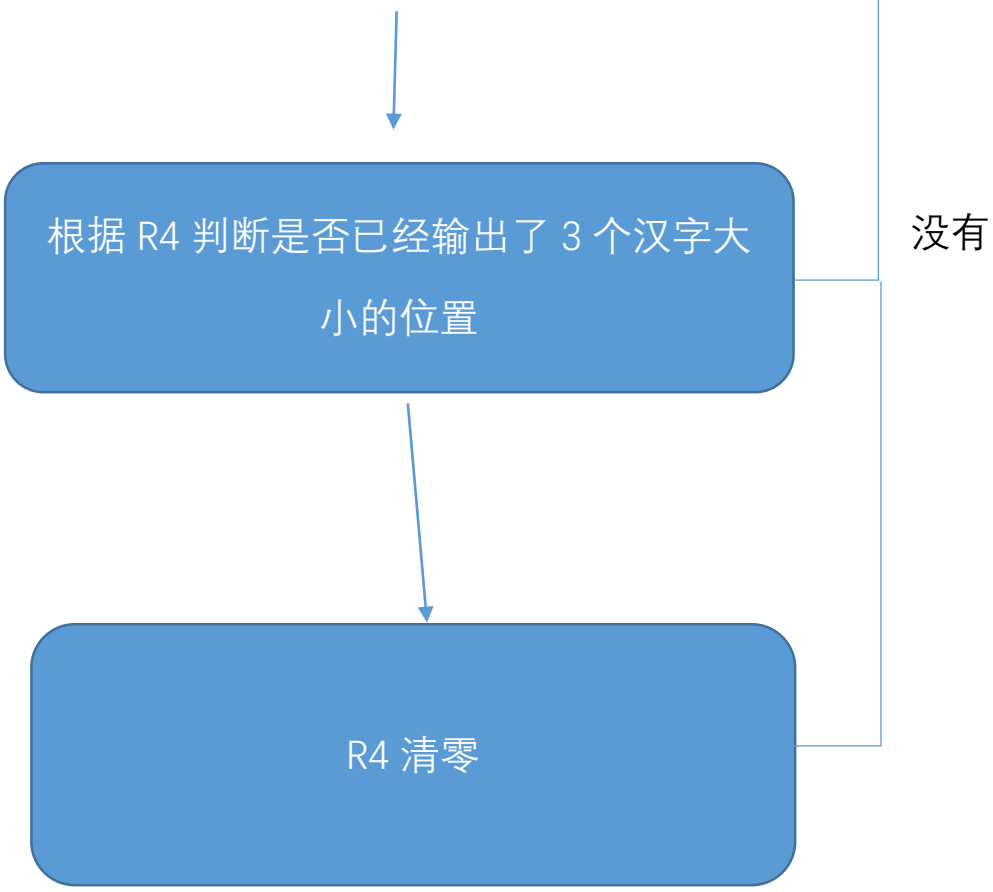
2. 使用 MCS-51 汇编语言，使用 LED 点阵显示器显示出正确的汉字字符及动态效果;

3. 将编译后的程序下载到 51 单片机，观察 LED 显示屏的显示结果。

五、程序流程图




置位 EN_X,复位 CX_XL,并将 TABLE1 对应的字传到 A 中
分两次，每次向 D_X 在 CKX 由低到高的过程中输出一个二进制位，一次输出 8 位
调用子程序对 Y 进行输出
置位 EN_Y,复位 CK_YL,并将 TABLE2 对应的字传到 A 中
分两次，每次向 D_Y 在 CKY 由低到高的过程中输出一个二进制位，一次输出 8 位
此时已经输出了一行，对位置参数加 1，行参数 R4 加一



六、程序代码

```
ORG 0000H
LJMP START
ORG 0030H
START:预定义接口
        D_X EQU P0.0
        CK_X EQU P0.1
        CK_XL   EQU P0.2
        D_Y  EQU P0.3
        EN_Y    EQU P0.4
        CK_Y EQU P0.5
        CK_YL   EQU P0.6
```

```
        EN_X EQU P0.7
REPETETION:;输出了一个 16*16
        MOV R4,#0
INIT:;初始化
        MOV R0,#0   ;TABLE1 的位移，x 的位移
        MOV R1, #0 ； TABLE2 的位移，对应的 y 的值
        MOV R2,#16;
SQUARE16：
        CALL DEALX
        CALL DEALY
        LCALL DELAY
        INC R0;每个存放偏移量的寄存器向后移两位，指向下一个
        INC R1
        DJNZ R2,SQUARE16
        LCALL DELAY
        LCALL DELAY
        LCALL DELAY
        LCALL DELAY
        LCALL DELAY
        LCALL DELAY
        INC R4
```

```
        INC R4
        MOV A,R4
        SUBB A,#96
        JZ   REPETETION
        LJMP INIT;输出了 3 个字了，重头开始
DEALX:
SETX:
                SETB EN_X ;X 的使能端
    CLR CK_XL
    MOV DPTR,#TABLE1
    MOV A,R0
    MOVC A,@A+DPTR;从 table 取数
    MOV R6,#8;连续输出 8 次
DEALX1:
    CLR CKX;在 ckx 由低到高的过程中输出
    RLC A
    MOV D_X,C;向 D_X 输出位
    SETB CKX
    DJNZ R6,DEALX1;未输出一个字就继续
                INC R0
    MOV A,R0
    MOVC A,@A+DPTR;将 A 指向下一个 8 位字
```

```
    MOV R6,#8
DEALX2:
    CLR CKX
    RLC A
    MOV D_X,C
    SETB CKX
    DJNZ R6,DEALX2
    SETB CK_XL
    CLR EN_X;关闭使能端
                    RLC
DEALY:
SETY:
                    SETB EN_Y :Y 的使能端
    CLR   CK_YL
    MOV DPTR,#TABLE
    MOV A,R1
    ADD A,R4
    MOVC A,@A+DPTR
    MOV R6,#8
DEALY1:
                    CLR CKY
    RLC A
```

```
    CPL C
    MOV D_Y,C
    SETB CKY
    DJNZ R6,DEALY1
    MOV A,R1
                    INC R1
    ADD A,R1
    MOVC A,@A+DPTR
    MOV R6,#8
DEALY2:
    CLR CKY
    RLC A
    CPL C
    MOV D_Y,C
    SETB CKY
    DJNZ R6,DEALY2
    SETB CK_YL
    CLR EN_Y
DELAY:
    MOV R6,#0FFH
LOOPD1:
    MOV R2,#0FH
```

```
LOOPD2:
    DJNZ R2,LOOPD2
    DJNZ R6,LOOPD1
RET
TABLE:

DB
04H,00H,24H,10H,24H,12H,24H,21H,7FH,0FEH,0C4H,40H,44H,82
H,04H,02H;
DB
04H,04H,0FFH,0C8H,04H,30H,44H,28H,34H,44H,05H,82H,04H,1F
H,00H,00H

DB
01H,00H,26H,82H,34H,84H,2CH,89H,24H,91H,24H,0E1H,37H,0B
2H,4CH,0AAH;
DB
44H,0A4H,44H,0A4H,4CH,0AAH,74H,0B2H,0C4H,81H,45H,01H,0
6H,01H,00H,00H
DB
00H,80H,01H,00H,06H,00H,1FH,0FFH,0E0H,00H,02H,08H,04H,30
H,18H,0C0H;
```

DB

F0H,02H,10H,01H,13H,0FEH,10H,00H,10H,80H,14H,60H,18H,18H,00H,00H

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
```