单片机控制与应用实验

第二次实验报告

雷新丽 53160805

实验三　步进电机原理及应用

一、实验目的和要求

1.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。

2.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。

3.了解数码管输出的原理及编程方式。

二、实验设备

1.单片机测控实验系统

2.步进电机控制实验模块

3.Keil 开发环境

4.STC–ISP 程序下载工具

三、实验内容

1.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。

2.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。

3.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

1.预习

2.简单程序录入和调试

3.程序调试

4.编写程序，完成功能

五、实验原理

1.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。

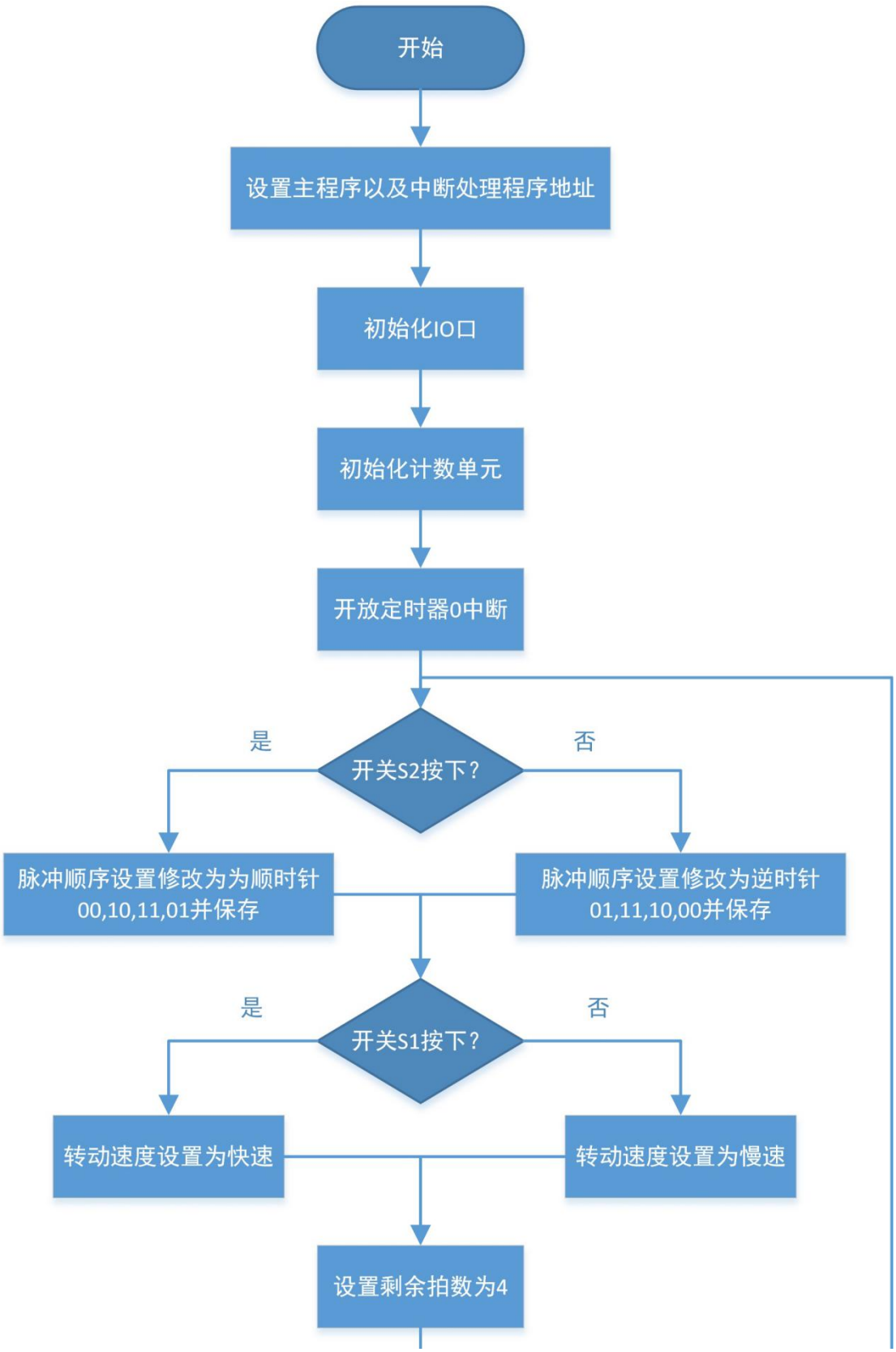
2.本开发平台有 3 个数码管，使用串行方式连接在一起，具体电路参见实验原理。要想输出一个字形码，就需要从高位到低位依次向移位寄存器输出 8 个比特。移位寄存器的数据线和时钟线分别接到单片机的 P4.5 和 P4.4 管脚，可以使用 MCS–51 里面的位操作指令进行输出。连续输出 3 个字形，24 个 bit 之后，欲显示的字形将稳定地显示在数码管上，程序可以转而执行其他工作。

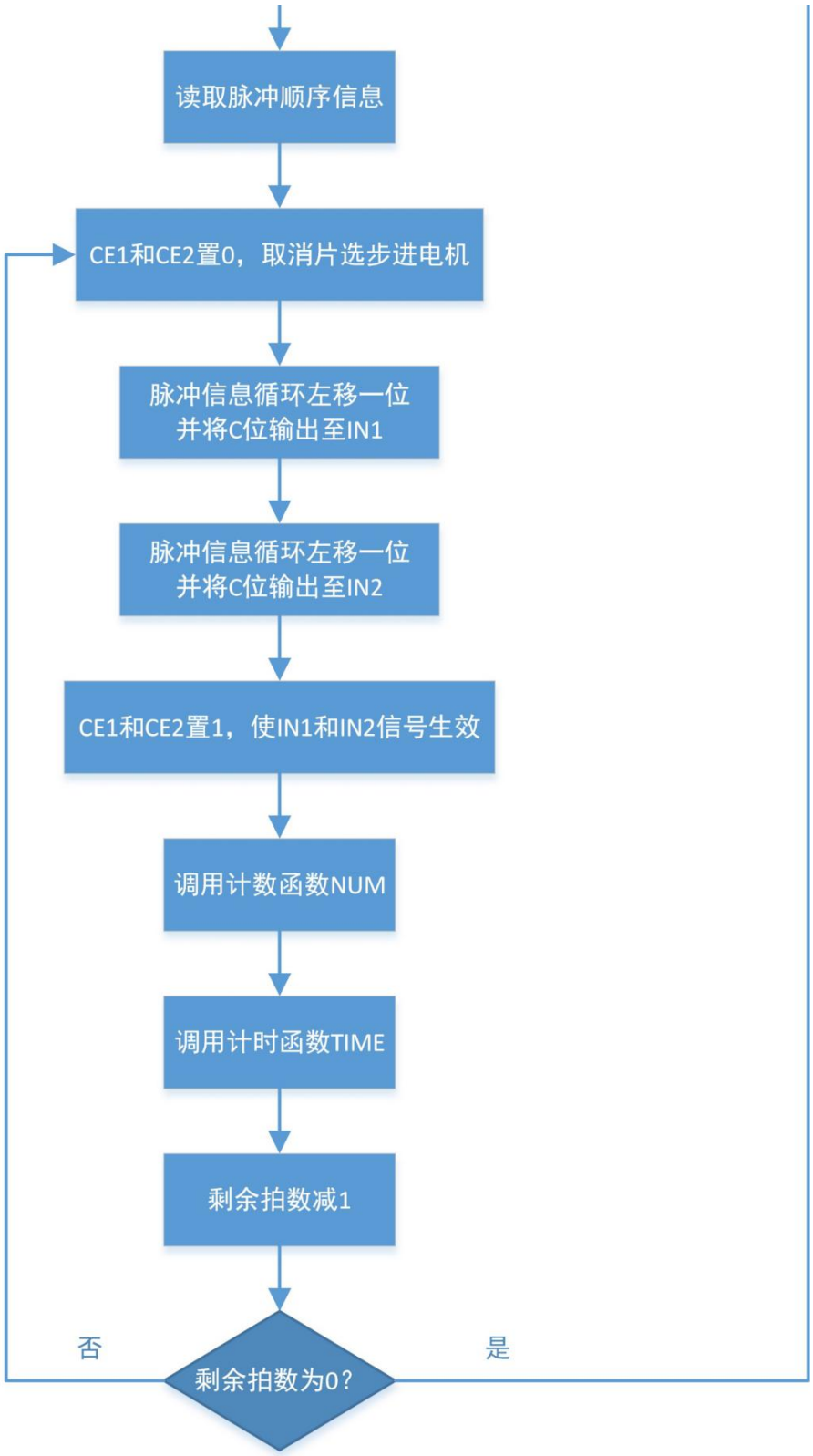
3. 采用 3 个 74HC164 级联控制三个数码管的显示，具体实验原理如下图所示。其中使用单片机 P4.5 作为模拟串口数据，使用 P4.4 模拟串口时钟，CLR 端接高电平。使用上一个 74HC164 的 Q7 作为下一个 74HC164 的输入端。

4. 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。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。

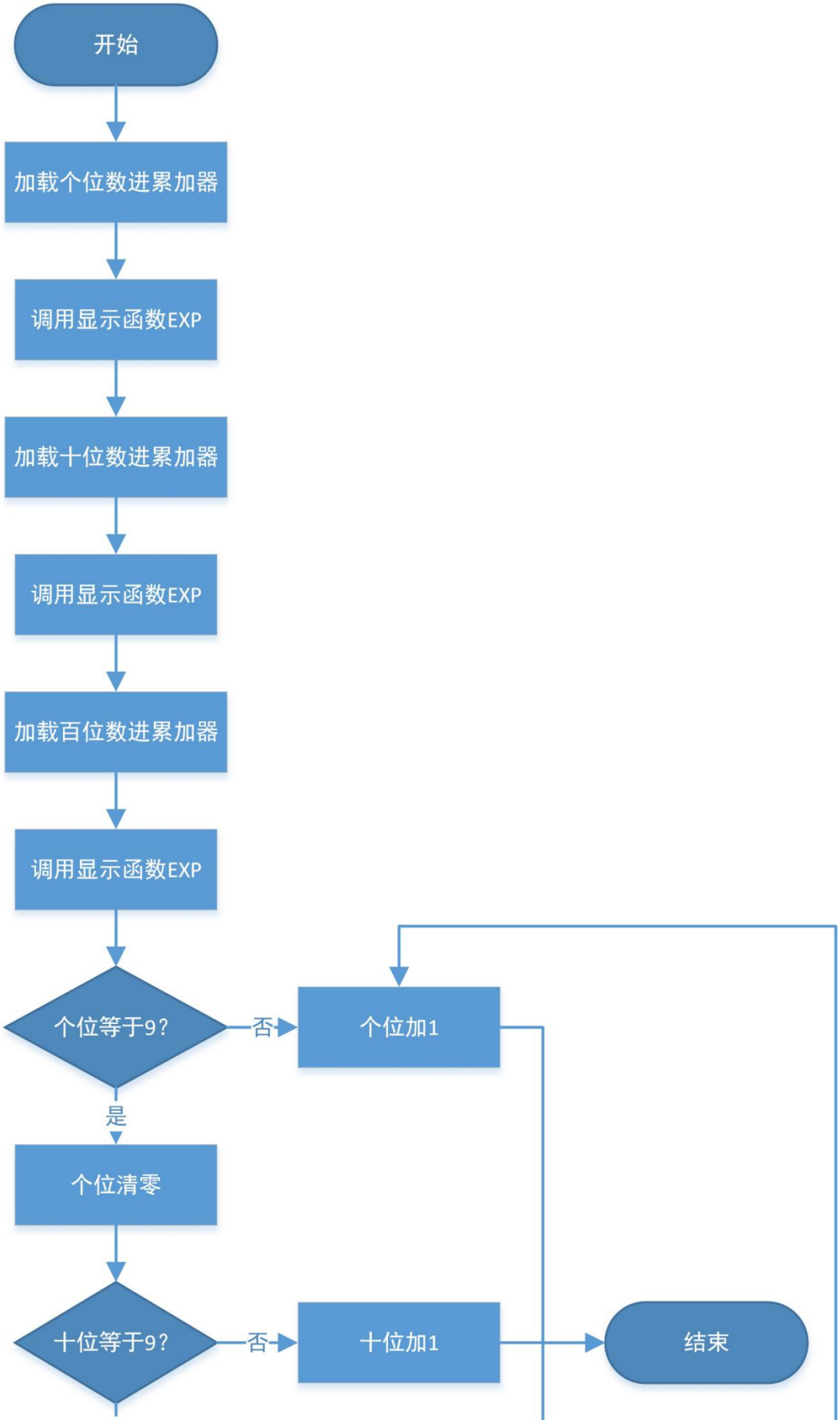
六、程序流程图

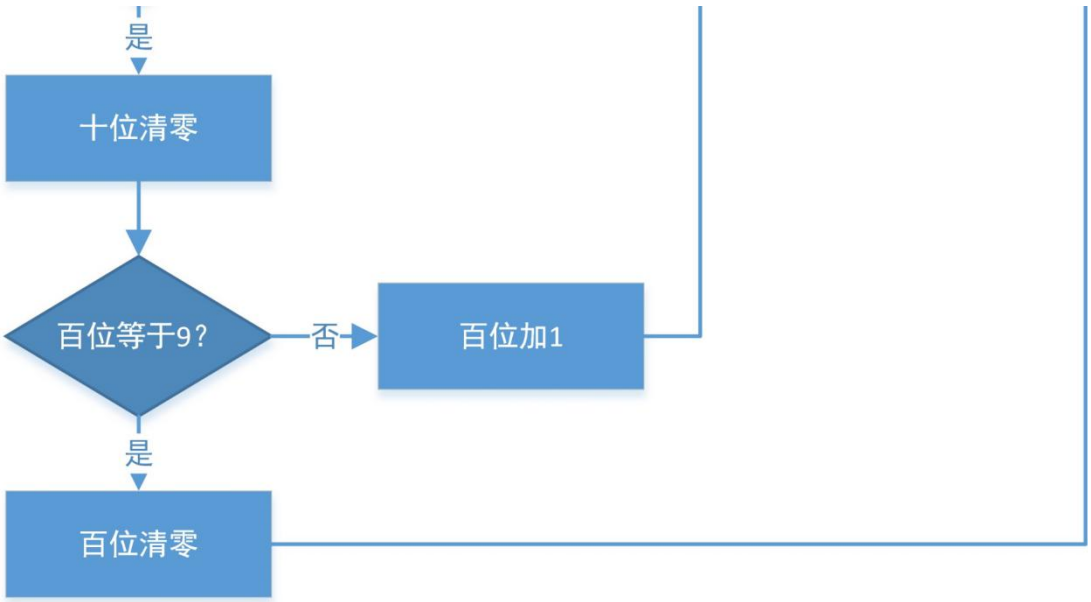
主程序：


开始
设置主程序以及中断处理程序地址
初始化IO口
初始化计数单元
开放定时器0中断
开关S2按下？
是
否
脉冲顺序设置修改为为顺时针
00,10,11,01并保存
脉冲顺序设置修改为逆时针
01,11,10,00并保存
开关S1按下？
是
否
转动速度设置为快速
转动速度设置为慢速
设置剩余拍数为4

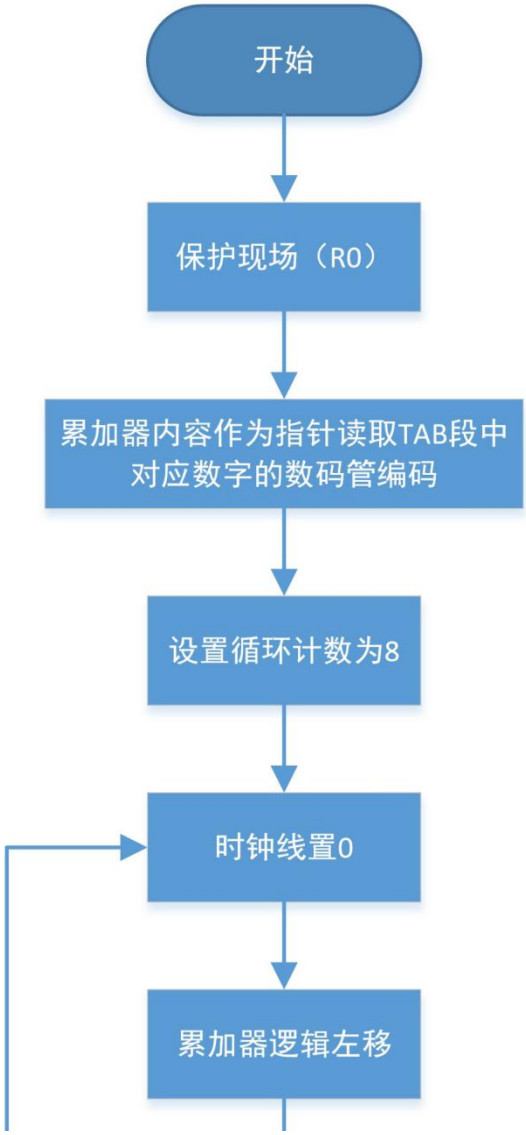

读取脉冲顺序信息
CE1和CE2置0，取消片选步进电机
脉冲信息循环左移一位
并将C位输出至IN1
脉冲信息循环左移一位
并将C位输出至IN2
CE1和CE2置1，使IN1和IN2信号生效
调用计数函数NUM
调用计时函数TIME
剩余拍数减1
剩余拍数为0?
否
是

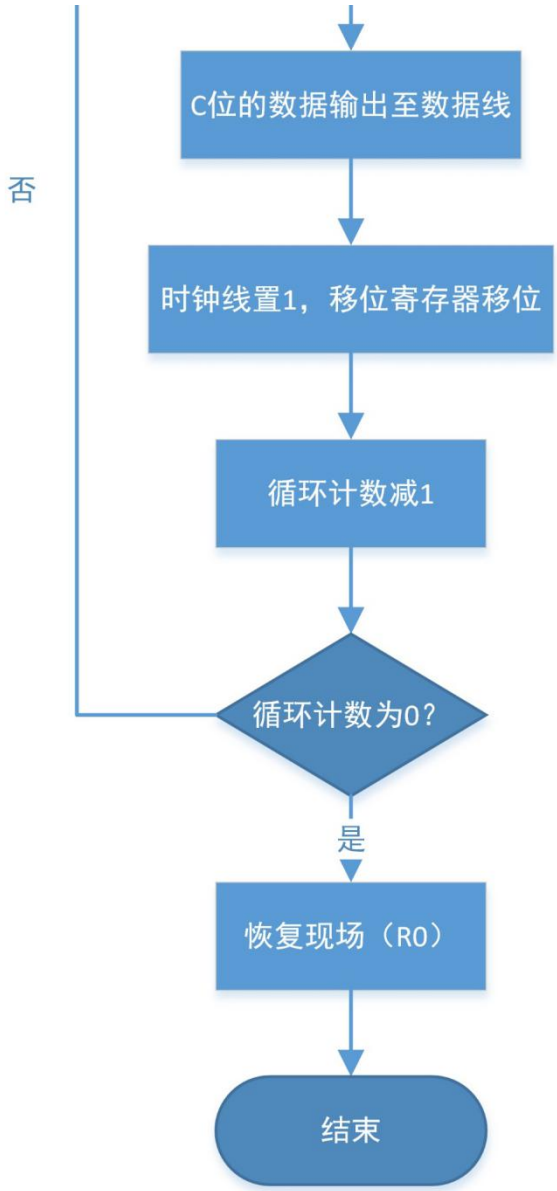
计数函数 NUM：


开始
加载个位数进累加器
调用显示函数EXP
加载十位数进累加器
调用显示函数EXP
加载百位数进累加器
调用显示函数EXP
个位等于9?
否
个位加1
是
个位清零
十位等于9?
否
十位加1
结束

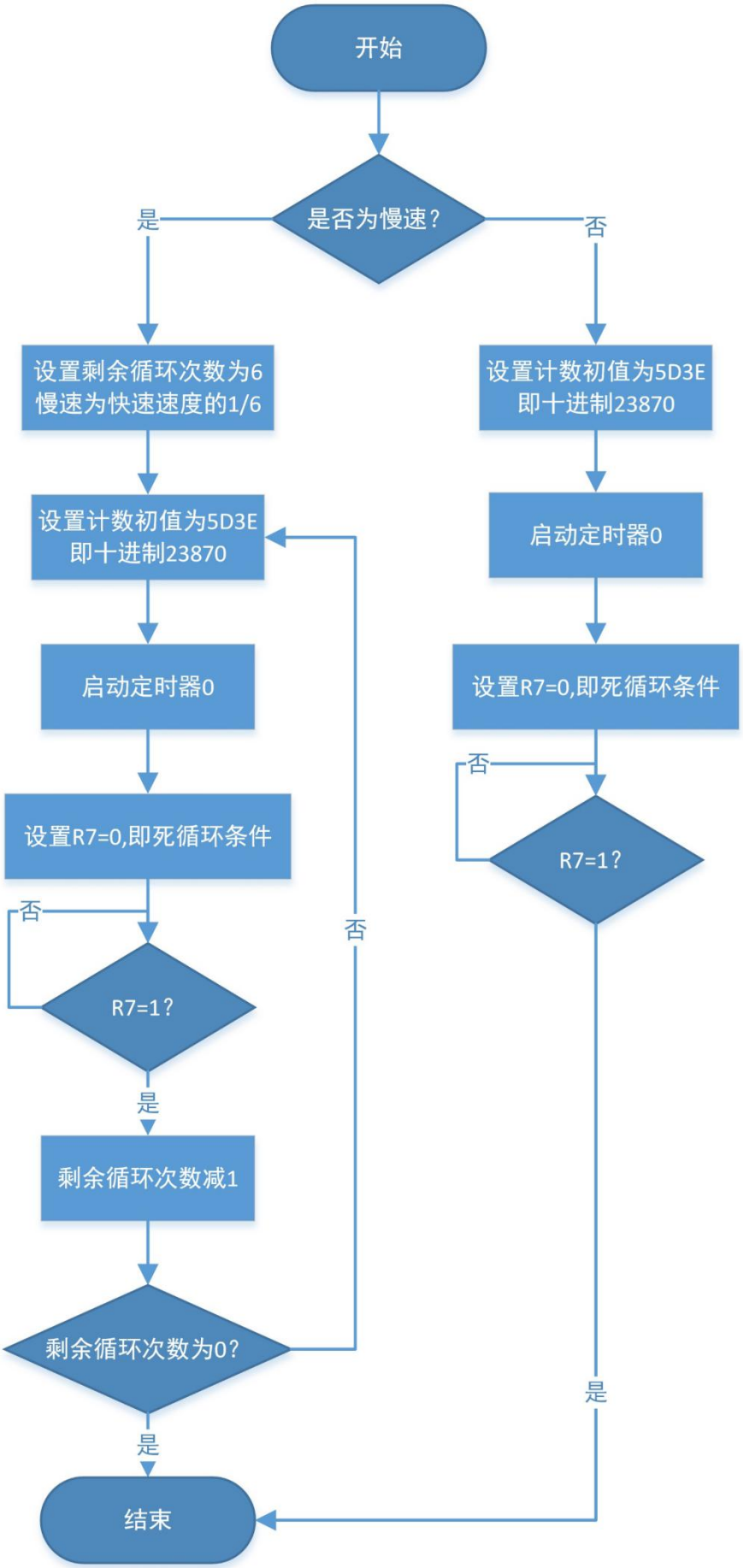

是
十位清零
百位等于9?
否
百位加1
是
百位清零

显示函数 EXP：

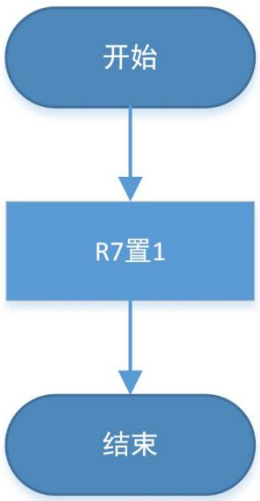

开始
保护现场（R0）
累加器内容作为指针读取TAB段中对应数字的数码管编码
设置循环计数为8
时钟线置0
累加器逻辑左移



计时函数 TIME：


开始
是否为慢速？
是
否
设置剩余循环次数为6
慢速为快速速度的1/6
设置计数初值为5D3E
即十进制23870
设置计数初值为5D3E
即十进制23870
启动定时器0
启动定时器0
设置R7=0,即死循环条件
设置R7=0,即死循环条件
否
否
R7=1？
R7=1？
是
剩余循环次数减1
剩余循环次数为0？
否
是
是
结束

定时器 0 中断处理程序 EINT0:



七、程序代码

```
ORG  0000H                ;复位起始地址
     LJMP  START
ORG  000BH                ;中间地址保留给中断向量表
     LJMP  EINT0          ;定时器 0 中断程序入口地址
ORG  0040H                ;程序实际起始地址
START:
     P4    EQU  0C0H      ;P4 口地址
     P4SW  EQU  0BBH      ;P4SW 寄存器地址
     CLK   EQU  P4.4      ;移位寄存器的时钟线
     DAT   EQU  P4.5      ;移位寄存器的数据线
     SW    EQU  P3.6      ;标准 I/O 口 PORT3[6]

     MOV   P4SW,#70H           ;01110000B , P4.4,P4.5,P5.6 作 I/O 口
PORT4[4],PORT4[5],PORT4[6]
     MOV  DPTR,#TAB       ;此为数据指针 DPTR0（16 位）,指向数据区域 TAB 的首址
LP:
     ;计数单元数字
     MOV  R3,#0           ;个位
     MOV  R4,#0           ;十位
     MOV  R5,#0           ;百位
I1:
     MOV  TMOD,#01H       ;TMOD(定时器模式寄存器),设置 T0 工作在方式 1(16 位的计数
器),GATE(控制定时器的两种启动方式)等于 0，不受外部控制
     MOV IE,#82H          ;IE(中断允许寄存器)T0 中断允许,EA 置 1 开放中断,ET0 置 1 允许
ET0(定时器 0)中断;
     ;ORL  IP,#2H          ;逻辑或，T0 中断优先级高


NEXT:
```

```
        JB    P3.7,OPP          ;如果 P3.7 等于 1 则转移(开关 S2,按下为 0)
        MOV   R0,#00101101B  ;00,10,11,01 按下，顺时针
        MOV       20H,R0
        LJMP  SS1
OPP:
        MOV   R0,#01111000B  ;01,11,10,00 松开，逆时针
        MOV       20H,R0
SS1:
        JB    P3.6,SPD          ;如果 P3.6 等于 1 则转移(开关 S1,按下为 0)
        MOV   R2,#0H              ;按下，快速
        LJMP  L0
SPD:
        MOV   R2,#1H              ;松开，慢速
L0:
        MOV   R1,#4               ;4 拍
        MOV       R0,20H
;让步进电机顺/逆时针转动一步
L1:   MOV   A,R0
        CLR   P1.1                ;CE1 置 0,CE1 和 CE2 二者选一即可
        CLR   P1.4                ;CE2 置 0
        RLC   A                   ;循环左移操作,A 的最高位进 C
        MOV   P3.2,C              ;IN1
        RLC   A
        MOV   P1.0,C              ;IN2
        SETB  P1.1                ;CE1 置 1
        SETB  P1.4                ;CE2 置 1
        MOV   R0,A
        LCALL   NUM               ;LCALL:长跳转
        LCALL   TIME
        DJNZ  R1,L1               ;DJNZ:减一不为零跳转指令
        LJMP  NEXT
;计时部分
TIME:
        CJNE  R2,#1,QUICK    ;R2!=1 时,即按下 S1 时跳转
        MOV   R6,#6
;慢速,计时 6 次,即快速的 1/6
TIM2:
        MOV   TH0,#5DH            ;S=23870=5D3EH
        MOV   TL0,#3EH
        SETB  TR0                 ;定时器 0 启动
        MOV   R7,#0H
;死循环,程序等待中断
TIM3:
        CJNE  R7,#1H,TIM3
```

```
        DJNZ  R6,TIM2
        LJMP  OUT
;快速  60 转/分
QUICK:
        MOV  TH0,#5DH          ;S=23870=5D3EH
        MOV  TL0,#3EH
        SETB  TR0              ;定时器 0 启动
        MOV  R7,#0H            ;R7 置 0
;死循环,程序等待中断
TIM1:
        CJNE  R7,#1H,TIM1
OUT:
        RET
;计时器 0 中断处理
EINT0:
        MOV  R7,#1             ;R7 置 1,跳出死循环
        RETI
;显示已转动的步数，每转动一次显示一个数
NUM:
S0:
        ;显示已转动的步数
        MOV  A,R3
        CALL  EXP
        MOV  A,R4
        CALL  EXP
        ;mov  r5,0ffh
        MOV  A,R5
        CALL  EXP
        ;已转动的步数加一,达到 999 时归零
        CJNE  R3,#9,S1         ;CJNE:不相等则转移,R3 每次增 1,到 9 之后归零
        MOV   R3,#0
        CJNE  R4,#9,S2
        MOV   R4,#0
        CJNE  R5,#9,S3
        MOV   R5,#0
S1:
        INC  R3
        LJMP  STOP
S2:
        INC  R4
        LJMP  STOP
S3:
        INC  R5
        LJMP  STOP
```

```
STOP:
RET
;输出数字至数码管
EXP:
    MOV     21H,R0          ;暂存 R0 内容至 21H 单元
    MOVC  A,@A+DPTR     ;MOVC 指令用于在代码区获取数据
    MOV  R0,#8          ;R0=8,作循环计数用
;将数字编码输出到 74HC164
;LOOP
CLY:
    CLR  CLK            ;P4.4 时钟线低电平
    RLC  A              ;累加器 A 的逻辑操作指令
    MOV  DAT,C
    SETB CLK            ;P4.4 时钟线高电平,CLK 每次由低变高时,数据右移一位,输入到
Q0
    DJNZ  R0,CLY
;LOOP
    MOV     R0,21H          ;从 21H 单元恢复数据到 R0
RET
;存储数码管每个数字对应编码
TAB:
DB  0C0H,0F9H,0A4H,0B0H,99H,92H,82H,0F8H,80H,90H
END
```

八、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?

答:

(1) 单四拍工作模式下，每次步进角度为 360/24=15 度

(2) 单四拍工作模式下，每次只有一相通电，因此 CE 端信号也需要按照一定的顺序改变。

采用单四拍工作模式，线圈的驱动顺序应为 A→B→$\overline{A}$→$\overline{B}$→A，IN1 和 IN2 脉冲顺序改为 01→10→00→00→01，CE1 和 CE2 脉冲顺序改为 10→01→10→01。

修改程序时将除了改变相位值之外，还应控制 CE 端信号。

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?

答:

(1) 单双八拍工作模式下，每次步进角度为 360/48=7.5 度

(2) 采用单双八拍工作模式，线圈的驱动顺序应为 A→AB→B→BA→$\overline{A}$→$\overline{A}$$\overline{B}$→$\overline{B}$→$\overline{B}$$\overline{A}$→A，IN1 和 IN2 脉冲顺序改为 01→11→10→10→00→00→00→01→01， CE1 和 CE2 脉冲顺序改为 10→11→01→11→10。修改程序时由于定时器定时周期变了，故需修改定时初始值为 44703；相位值和 CE 端信号也需改变，还有循环次数。

3.步进电机的转速取决于那些因素？有没有上、下限?

答:

(1) 取决于脉冲频率和工作模式

(2) 由于各种物理因素（如摩擦、机械惯性、响应时间等），步进电机的最高转速有限制，且根据电机的不同而不同，转速下限为 0

4.如何改变步进电机的转向?

答：调换脉冲顺序。

5.步进电机有那些规格参数，如何根据需要选择型号?

答;

(1) 功率、马力、电流、转速、效率、功率因数、额定转矩、额定电流、重量、空起频率等

(2) a. 选择需要的额定转矩：通常根据需要的转矩大小(即所要带动物体的扭力大小)，来选择哪种型号的电机。大致说来，扭力越大，所选择的电机的额定转矩也越大，同时电机尺寸也就越大。

b. 选择合适的转速：电机的输出转矩，与转速成反比。就是说，步进电机在低速时输出转矩较大，在高速旋转状态的转矩较小。如果高速的同时希望获得较大输出转矩，应该选择电感稍小一些的电机。要求低速大力矩时，就要选择电感和电阻大一些的电机。

c. “空起频率”的选择：步进电机空载起动频率的选择步进电机空载起动频率，通常称为“空起频率”。这是选择电机比较重要的一项指标。如果要求在瞬间频繁启动、停止、高转速，通常需要“加速启动”。如果需要直接启动达到高速运转，最好选择反应式或永磁电机。

d. 考虑使用环境：特种步进电机能够防水、防油，用于某些特殊场合。例如水下机器人，就需要放水电机。

实验四　LED 点阵显示屏

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

1.单片机测控实验系统

2.LED 点阵显示器实验模块

3.Keil 开发环境

4.STC-ISP 程序下载工具

三、实验内容

1.了解 16*16 点阵电路的原理，编写汇编语言程序。

2.编写一行汉字字符（至少三个字）的显示程序。

3.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、实验原理

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 ASC16、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.行和列分别使用两个移位寄存器作为输出。

3.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)。

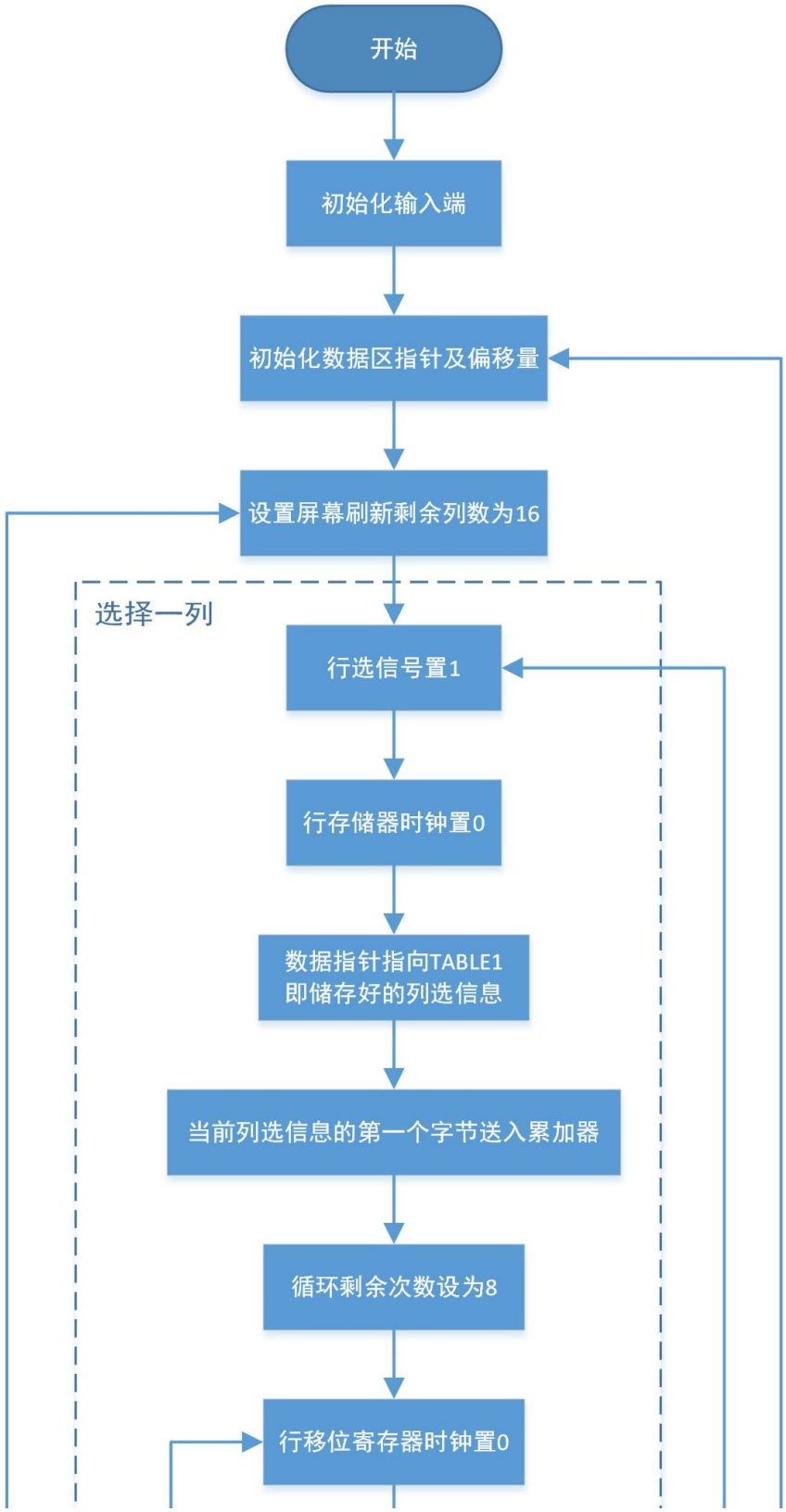
4.为了能够显示出一个点阵字型，需要进行循环扫描，也就是每一次只点亮一行，然后

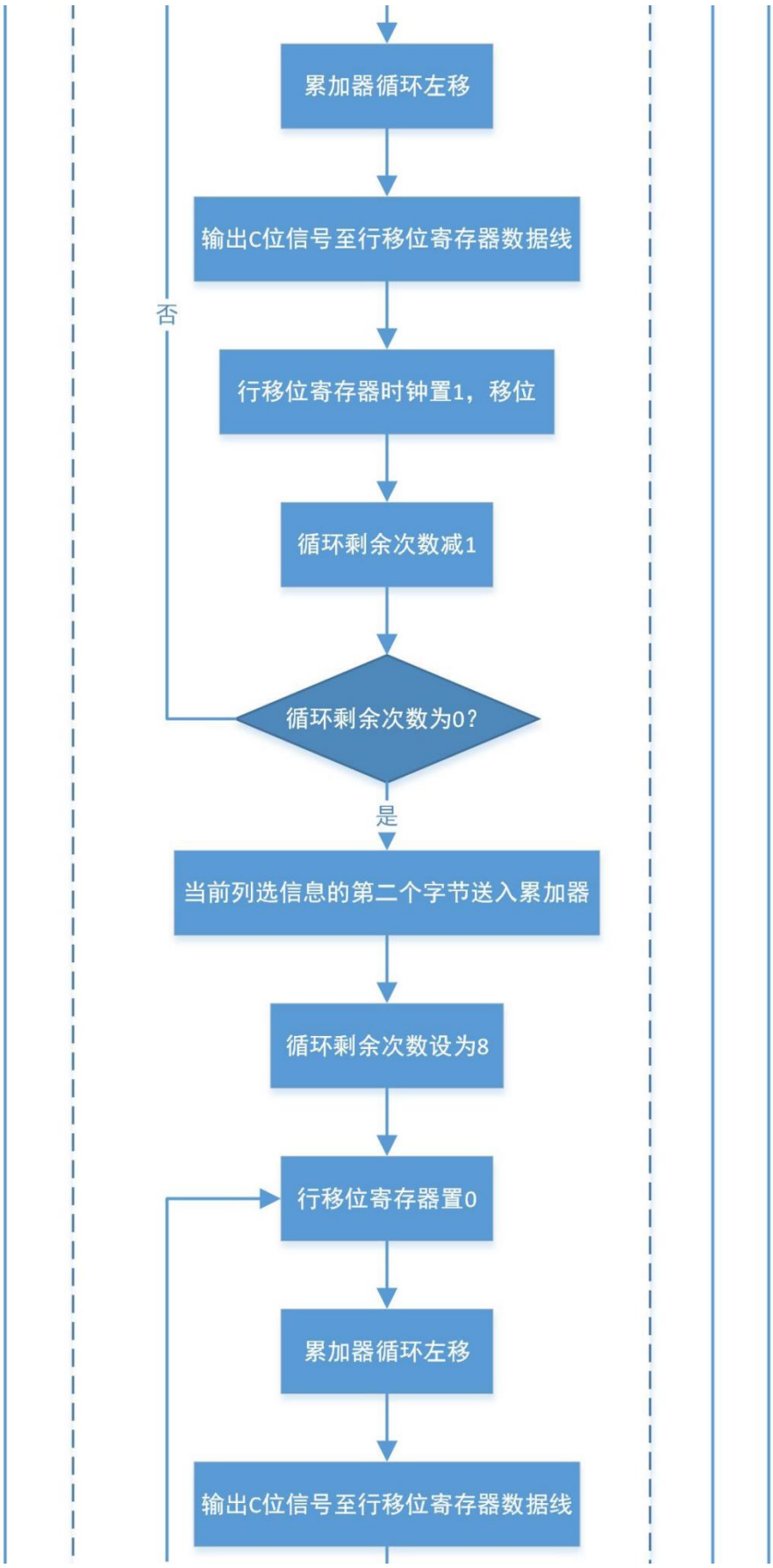
在列上输出该列对应的 16 个点阵值。

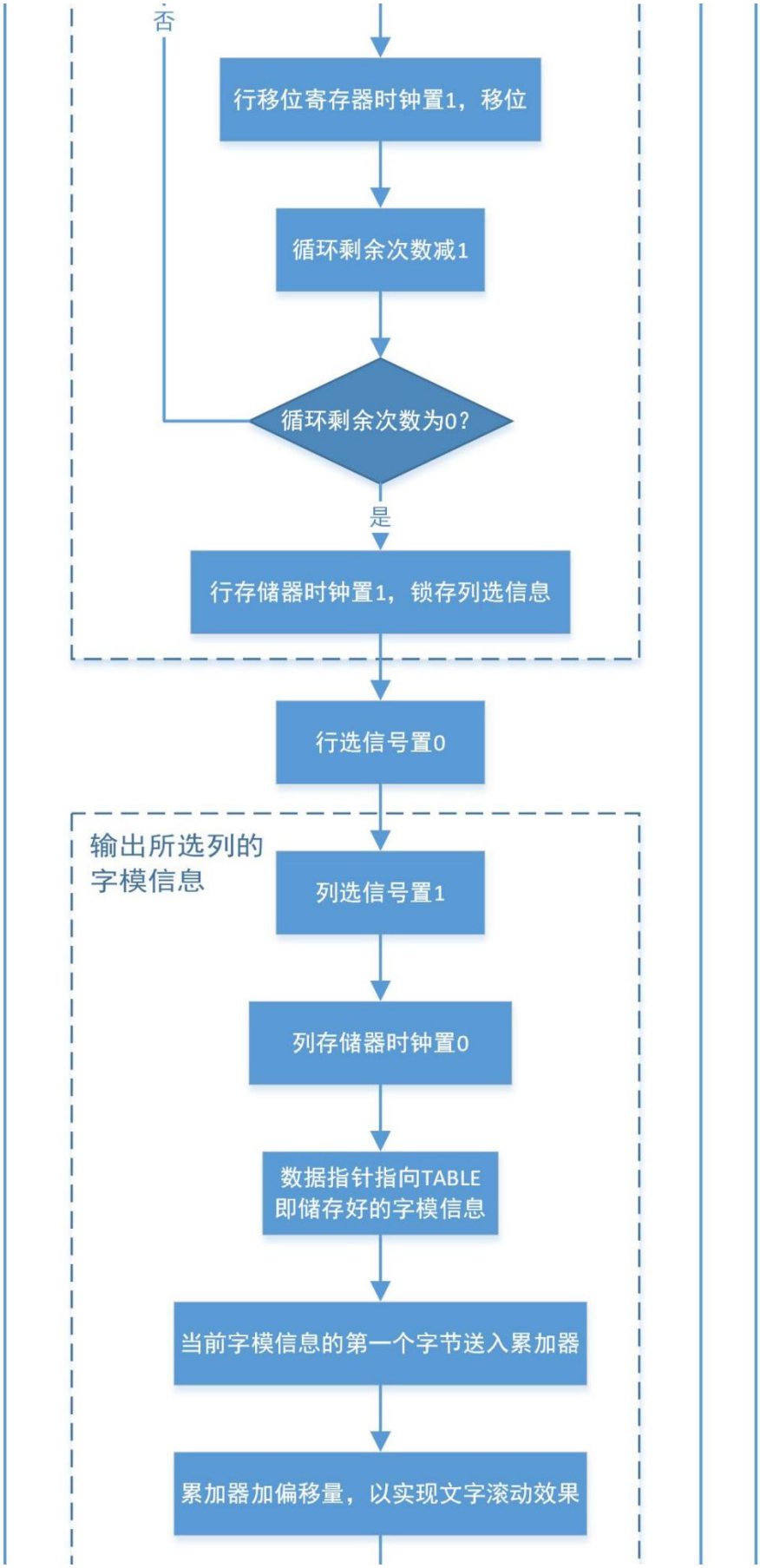
5.输出一行后暂停一段时间，输出下一行。为了达到较好的显示效果，整屏总的扫描时间不高于 40ms。

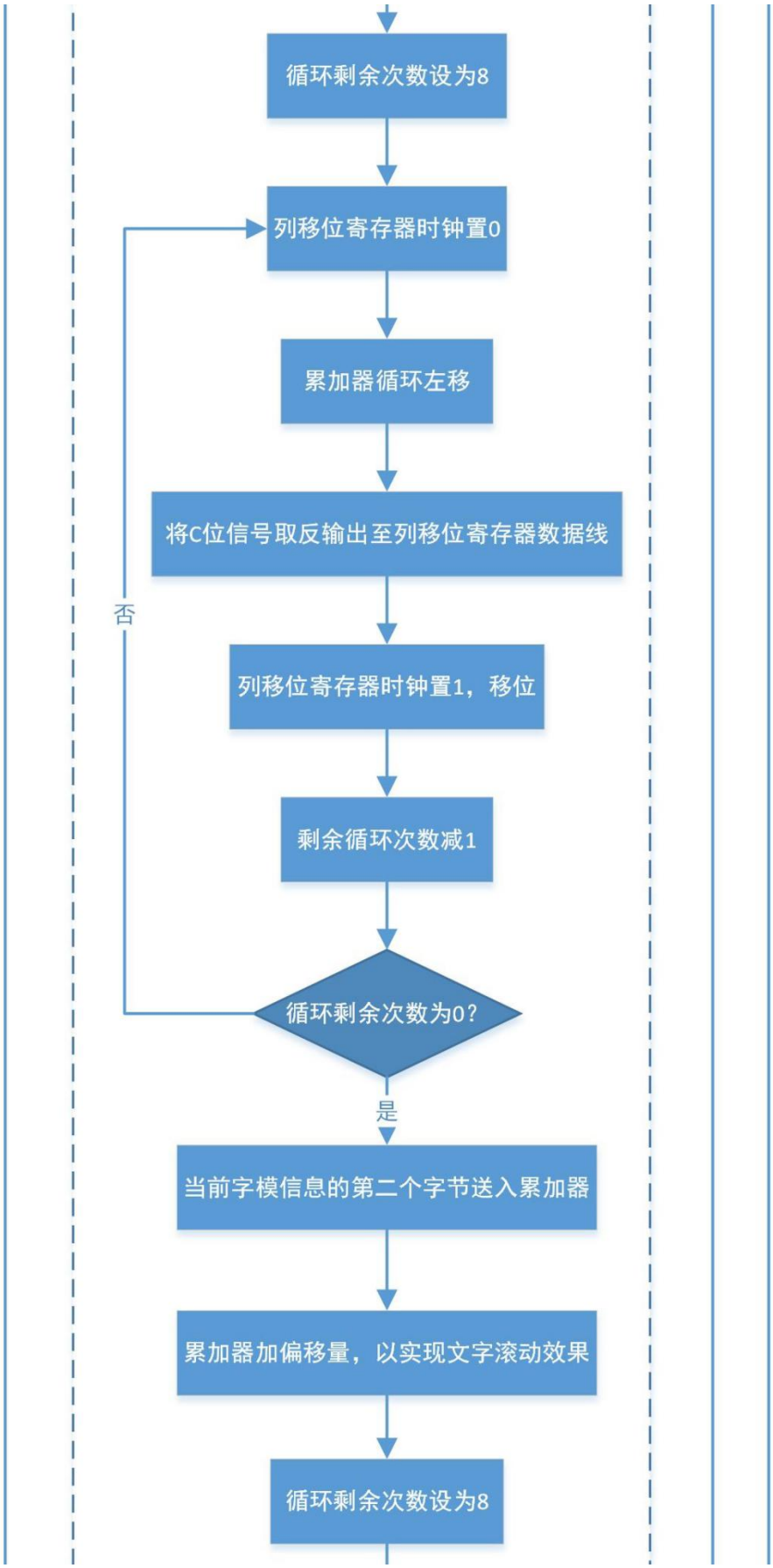
六、程序流程图

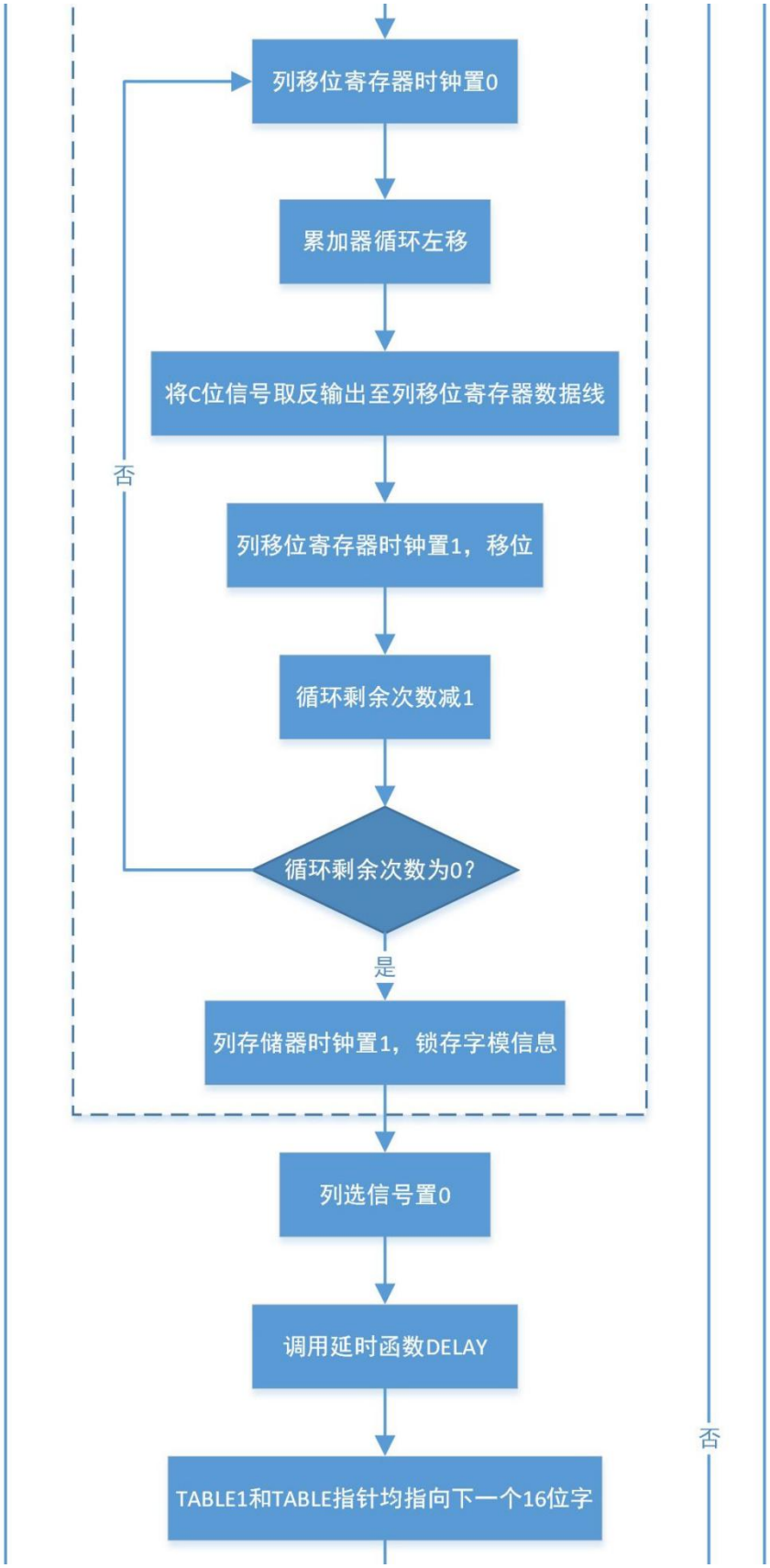
主程序：

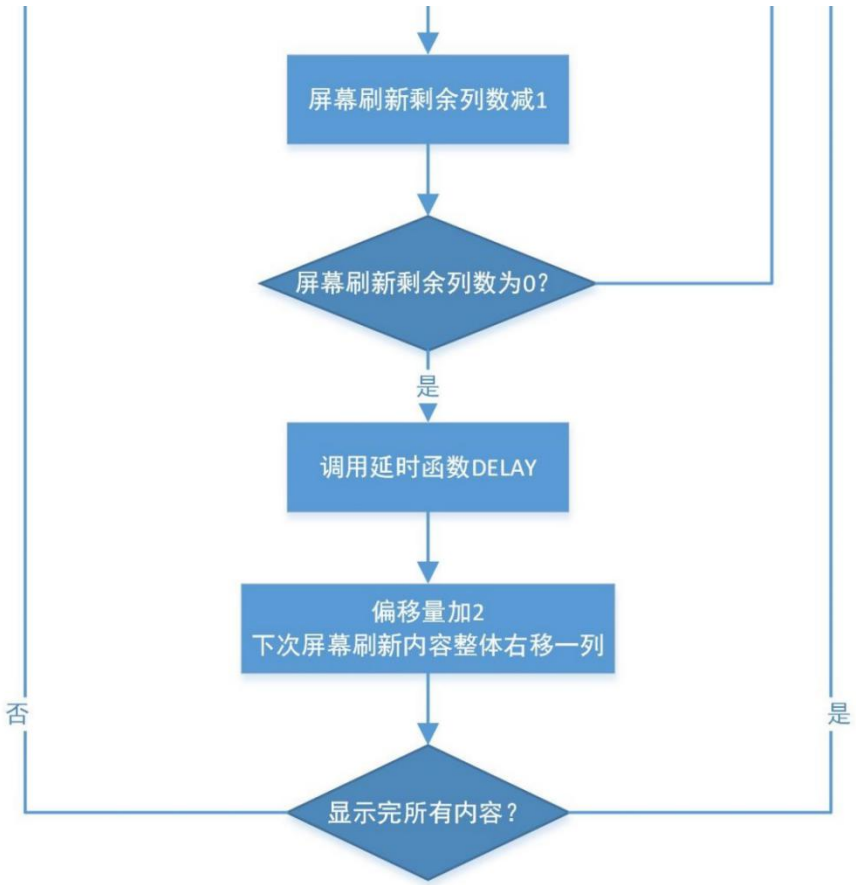



累加器循环左移
输出C位信号至行移位寄存器数据线
否
行移位寄存器时钟置1，移位
循环剩余次数减1
循环剩余次数为0?
是
当前列选信息的第二个字节送入累加器
循环剩余次数设为8
行移位寄存器置0
累加器循环左移
输出C位信号至行移位寄存器数据线

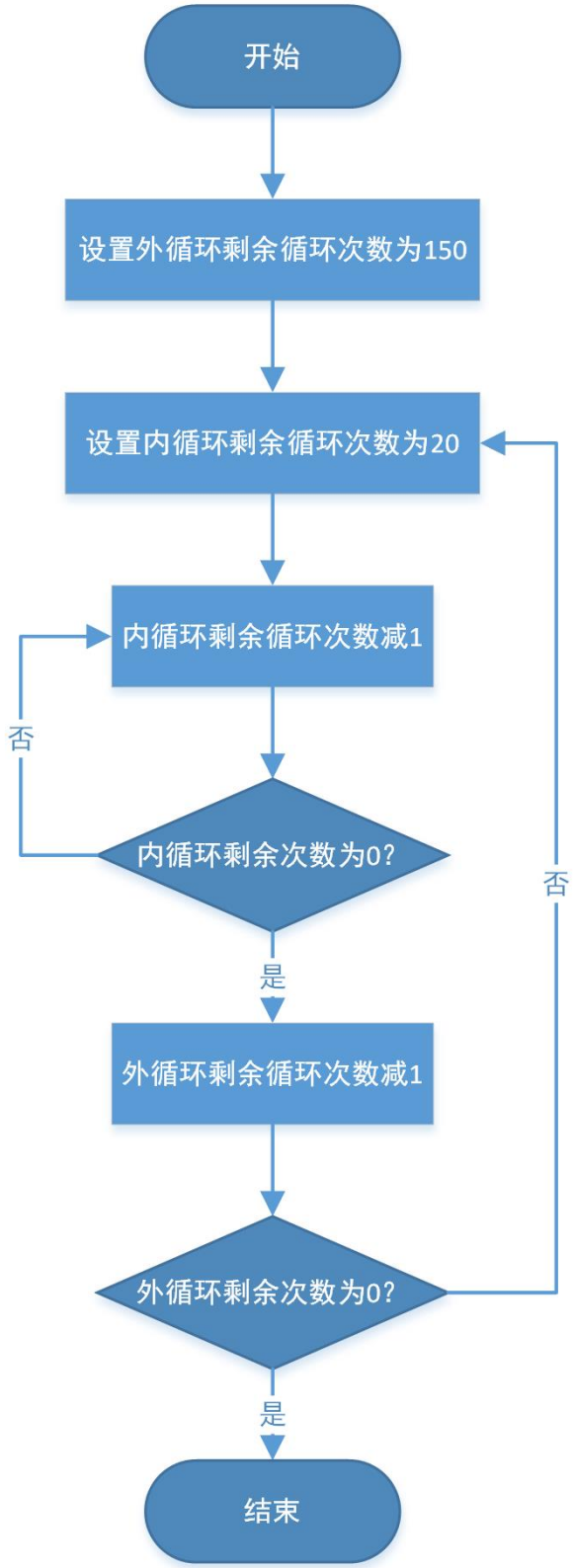

否
行移位寄存器时钟置1，移位
循环剩余次数减1
循环剩余次数为0?
是
行存储器时钟置1，锁存列选信息
行选信号置0
输出所选列的
字模信息
列选信号置1
列存储器时钟置0
数据指针指向TABLE
即储存好的字模信息
当前字模信息的第一个字节送入累加器
累加器加偏移量，以实现文字滚动效果


循环剩余次数设为8
列移位寄存器时钟置0
累加器循环左移
将C位信号取反输出至列移位寄存器数据线
列移位寄存器时钟置1，移位
剩余循环次数减1
循环剩余次数为0?
否
是
当前字模信息的第二个字节送入累加器
累加器加偏移量，以实现文字滚动效果
循环剩余次数设为8


列移位寄存器时钟置0
累加器循环左移
将C位信号取反输出至列移位寄存器数据线
列移位寄存器时钟置1，移位
循环剩余次数减1
循环剩余次数为0?
否
是
列存储器时钟置1，锁存字模信息
列选信号置0
调用延时函数DELAY
TABLE1和TABLE指针均指向下一个16位字
否


屏幕刷新剩余列数减1
屏幕刷新剩余列数为0?
是
调用延时函数DELAY
偏移量加2
下次屏幕刷新内容整体右移一列
否
显示完所有内容?
是

延时函数 DELAY：



七、程序代码

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

```
START:
    D_X    EQU  P0.0    ;串行位移行输入
    D_Y    EQU  P0.3    ;串行位移列输入
    CKX    EQU  P0.1    ;行移位寄存器时钟
    CKY    EQU  P0.5    ;列移位寄存器时钟
    CK_XL  EQU  P0.2    ;行存储器时钟输入
    CK_YL  EQU  P0.6    ;列存储器时钟输入
    EN_X   EQU  P0.7    ;行使能端
    EN_Y   EQU  P0.4    ;列使能端
    MOV  R7,#0          ;R7 为偏移量,每刷新完一次屏幕便会加 2
;显示
SM:
    ;因为每次要输出 2 个字节，所以每个表要用到两个寄存器作为指针
    MOV  R0,#0          ;table1 的第一个字节
    MOV  R1,#1          ;table1 的第二个字节
    MOV  R4,#0          ;table 的第一个字节
    MOV  R5,#1          ;table 的第二个字节
;刷新屏幕
;LLOOPB(多个 L 代表着里面还有循环)
    MOV  R3,#16         ;一个屏幕 16 行
;输出一行至 LED(74HC595),行的作用是用来选择一列
SM16:
    SETB  EN_X          ;选中行
    CLR  CK_XL          ;行存储器时钟输入置 0
    MOV  DPTR,#TABLE1  ;DPTR=TABLE1
    MOV  A,R0           ;A=0
    MOVC  A,@A+DPTR
;LOOPB
    MOV  R6,#8
YW1:
    CLR  CKX            ;行移位寄存器时钟置 0
    RLC  A
    MOV  D_X,C          ;输出一位
    SETB  CKX           ;上升沿,行移位寄存器移位
    DJNZ  R6,YW1
;LOOPE
    MOV  A,R1           ;A=1
    MOVC  A,@A+DPTR     ;DPTR=TABLE1
;LOOPB
    MOV  R6,#8
YW0:
    CLR  CKX
    RLC  A
    MOV  D_X,C          ;行移位寄存器时钟置 0
```

```
        SETB  CKX               ;上升沿,行移位寄存器移位
        DJNZ  R6,YW0
;LOOPE
        SETB  CK_XL             ;上升沿,行位移寄存器→行存储器
        CLR  EN_X               ;取消选中行
        ;LCALL  DELAY
;输出一列至 LED(74HC595),列的作用是显示字模信息
        SETB  EN_Y              ;选中列
        CLR   CK_YL             ;列存储器时钟输入置 0
        MOV  DPTR,#TABLE    ;DPTR=TABLE
        MOV  A,R4
        ADD  A,R7
        MOVC  A,@A+DPTR
;LOOPB
        MOV  R6,#8
YW3:
        CLR  CKY
        RLC  A
        CPL  C
        MOV  D_Y,C              ;输出一位
        SETB  CKY               ;上升沿,列移位寄存器移位
        DJNZ  R6,YW3
;LOOPE
        MOV  A,R5
        ADD  A,R7
        MOVC  A,@A+DPTR
;LOOPB
        MOV  R6,#8
YW2:
        CLR  CKY
        RLC  A
        CPL  C
        MOV  D_Y,C              ;输出一位
        SETB  CKY               ;上升沿,列移位寄存器移位
        DJNZ  R6,YW2
;LOOPE
        SETB  CK_YL             ;上升沿,行位移寄存器→行存储器
        CLR  EN_Y               ;取消选中列
        LCALL  DELAY            ;延迟
;数据区域指针调整
        INC  R0                 ;table1 指针指向下一个 16 位字
        INC  R0
        INC  R1
        INC  R1
```

```
    INC  R4              ;table 指针指向下一个 16 位字
    INC  R4
    INC  R5
    INC  R5
    DJNZ  R3,SM16    ;重复 16 次以刷新整个屏幕
;LLOOPE
    LCALL  DELAY         ;延时,控制文字移动速度
    LCALL  DELAY
    LCALL  DELAY
    INC  R7
    INC R7               ;偏移量加 2,下一次刷新屏幕时将从第二个 16 位字开始,以实现内容右移效果
    MOV  A,R7
    SUBB  A,#160         ;累加器带借位减立即数,当所有内容显示完毕时,R7 应该为 5*16*2=160,相当于后错 5 个汉字长,此时屏幕上显示的是最后那个空格
    JZ   START           ;当所有内容显示完毕时,重新显示内容
    LJMP  SM             ;下一次刷新屏幕
;延时,约为 3000*(12+24)/12M=9ms?
DELAY:
    MOV  R6,#150
DEL1:
    MOV  R2,#20
DEL2:
    DJNZ  R2,DEL2
    DJNZ  R6,DEL1
RET
;数据区域
;要输出的字,字模为逐列，顺向，阴码，水平镜像翻转
TABLE:
;"  "
DB
0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH
DB
0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH
;"学"
DB  00H,00H,0CH,20H,0AH,20H,C8H,20H,28H,20H,19H,20H,09H,A0H,09H,60H;
DB  69H,7EH,89H,21H,09H,22H,09H,20H,69H,20H,88H,20H,0CH,20H,02H,20H
;"单"
DB  00H,08H,00H,08H,1FH,0C8H,92H,48H,52H,48H,32H,48H,12H,48H,1FH,0FFH
DB  12H,48H,32H,48H,52H,48H,92H,48H,1FH,0C8H,00H,08H,00H,08H,00H,00H
;"片"
DB  00H,00H,00H,00H,04H,00H,04H,00H,04H,00H,04H,7FH,0FCH,40H,04H,40H
```

```
DB  04H,40H,04H,40H,04H,40H,04H,40H,7FH,0F8H,00H,06H,00H,01H,00H,00H
;"机"
DB  00H,00H,00H,1EH,00H,02H,00H,02H,7FH,0FCH,40H,00H,40H,00H,40H,00H
DB  7FH,0F8H,00H,06H,08H,0C1H,09H,00H,0FFH,0FFH,0BH,00H,08H,0C0H,08H,20H
;"  "
DB
0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FF
H,0FFH
DB
0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FFH,0FF
H,0FFH
;每一行分别对应着一列，共 16 列
TABLE1:
DB   80H,00H;1000  0000  0000  0000
DB  40H,00H;0100  0000  0000  0000
DB  20H,00H;0010  0000  0000  0000
DB  10H,00H;0001  0000  0000  0000
DB  08H,00H;0000  1000  0000  0000
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
```

八、思考题

1. 如何使用软件调整和控制 LED 点阵的亮度?

答：实验中过程发现 LED 点阵亮度和同一列同时点亮的 LED 个数有关，同时点亮的 LED 个数越多，则亮度越低，而一列只有一个 LED 点亮时亮度很高。这可能是因为多个 LED 间电流分流所致。故可以采取每次显示半列，反复刷新的方法来提高亮度。

2. 如何尽量避免显示过程中的闪烁?

答：将屏幕的刷新时间控制在 40ms 以内，并且尽量减少每次刷新过后的延迟时间，刷新的频率越高，闪烁越不明显。对于屏幕刷新，可以采用“重叠输出”的办法，即将列信息锁存的同时开始向移位寄存器输出下一个字模信息。另外在延时方面，可以采用定时器定时代替软件计时以使屏幕刷新更稳定。

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）？

答：将本实验的控制逻辑推广到更大的显示屏，因为屏幕工作原理相同，所以要更改的一个是行/列刷新时循环的次数，一个是列选信息存储的长度（如 64 列则一条列选信息 8 个字节），还需要加入一些必要的控制逻辑（如全局数据区域指针等）。因为每刷新一列都需

要完整地输出整个列的字模信息，所以可以针对每一行设一个数据指针，通过计算来设定每行数据指针的初始位置，可以采用对字模信息进行移位再输出来实现多行的效果，例如：如果是第二行，则将在这一列要输出的字模信息右移 16 位再进行输出，如果是第三行则右移 32 位……在输出字模信息时将每一行数据指针指向的信息合并为一整列的信号进行输出，这样可以做到同时显示多行文字。可能存在的问题是对内存区域的频繁读可能会影响刷新速度。