目录

项目二　密码锁控制器的制作

项目简介

图 2- 1　密码锁实图

在日常生活和工作中，住宅与部门的安全防范、单位的文件档案、财务报表以及一些个人资料的保存多以加锁的办法来解决（如图 2- 1）。目前门锁主要用弹子锁，其钥匙容易丢失；保险箱主要用机械密码锁，其结构较为复杂，制造精度要求高，成本高，且易出现故障，人们常需携带多把钥匙，使用极不方便，且钥匙丢失后安全性即大打折扣。针对这些锁具给人们带来的不便是因为使用机械式钥匙开锁，为满足人们对锁的使用要求，增加其安全性，用密码代替钥匙的密码锁应运而生。它的出现为人们的生活带来了很大的方便，有很广阔的市场前景。

电子密码锁是一种通过密码输入来控制电路或是芯片工作，从而控制机械开关的闭合，完成开锁、闭锁任务的电子产品。它的种类很多，有简易的电路产品，也有基于芯片的性价比较高的产品。现在应用较广的电子密码锁是以芯片为核心，通过编程来实现，其性能和安全性已大大超过了机械锁。密码锁的特点如下：

1）保密性好，编码量多，远远大于弹子锁。随机开锁成功率几乎为零。

2）密码可变，用户可以随时更改密码，防止密码被盗，同时也可以避免因人员的更替而使锁的密级下降。

3）误码输入保护，当输入密码多次错误时，报警系统自动启动。

4）无活动零件，不会磨损，寿命长。

5）使用灵活性好，不像机械锁必须佩带钥匙才能开锁。

6）电子密码锁操作简单易行，一学即会。

根据项目简介，可以把本项目分为以下几部分：数码管显示、按键识别、计时功能、密码锁功能等。所以在下面的章节将这些模块作为子任务介绍。

任务一　密码锁 PCB 硬件制作

一、任务要求

搭建密码锁模块，首先得有硬件作为基础。在随书附送的 PCB 中，取出密码锁模块。其正面与背面效果如图 2- 2 所示：

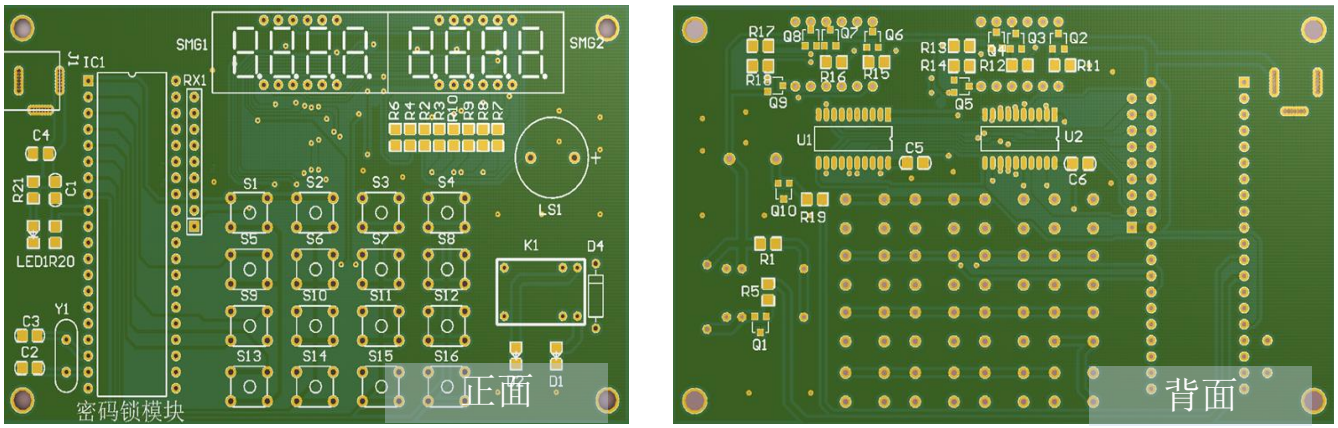

图 2- 2　PCB 正面与背面 PCB 图

本次任务要求为把密码锁模块电路板焊接完成。需要注意的是：由于我们使用最小系统板进行程序编写与调试，所以 PCB 正面左侧的单片机最小系统可以不焊，只需在 IC1 位置焊上底座就可以了如图 2- 3 所示。若不使用最小系统板，这些元器件需要焊接。

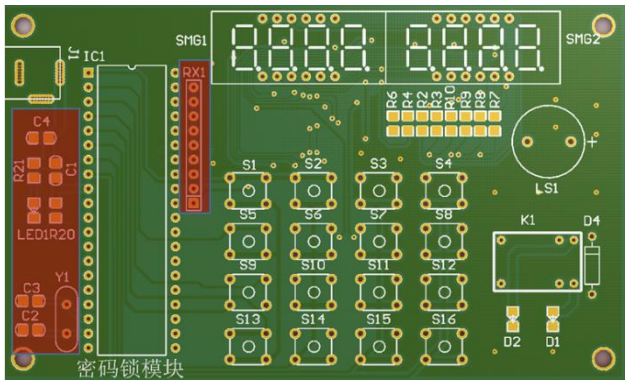

图 2- 3　使用最小系统板不需装配部分

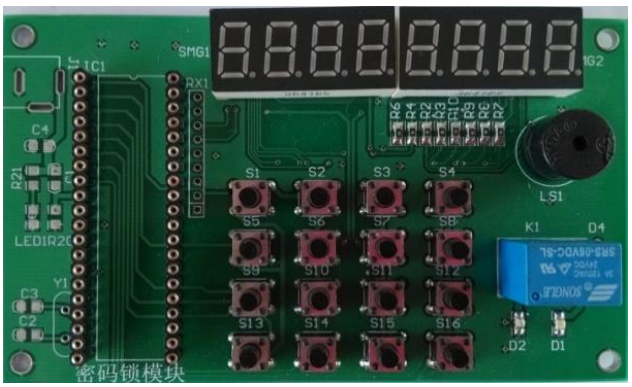

图 2- 4　密码锁模块实物演示图

图 2-4 为密码锁模块实物演示图，此图仅仅作为参考，以实际电路板为准。本任务需要同学们根据电路原理图及装配图装配要求完成实物 PCB 的焊接。

二、相关知识

下面就本次任务需要用到的元器件做一个简单介绍。

1、四位数码管 3641BS

数码管采用四位一体 3641BS 共阳极连接。其实物图与元器件管脚如下：

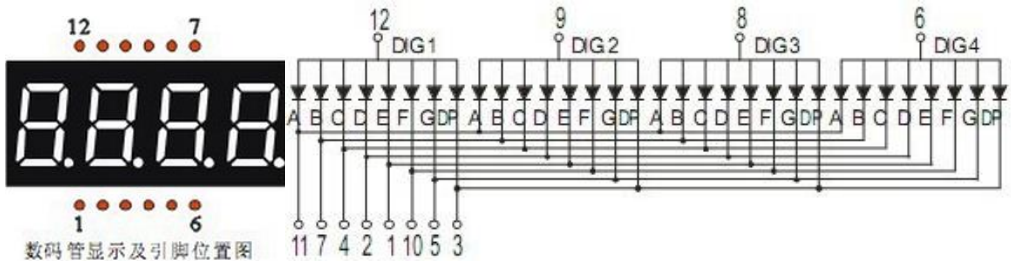

图 2-5　数码管 3641BS 实物图与管脚图

2、74HC377

74HC377 是 8D 触发器，实现了数据锁存功能。

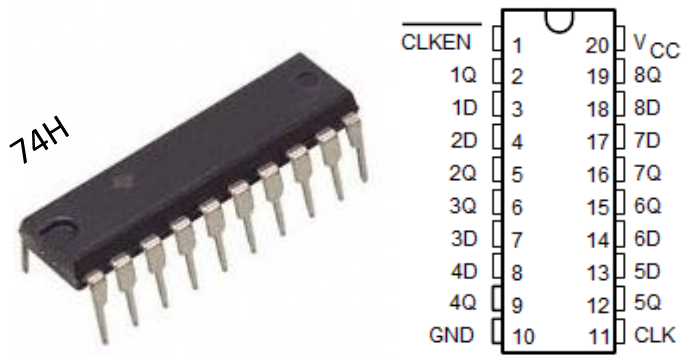

图 2-6　74HC377 集成电路实物图与管脚图

3、继电器 HFD41A

继电器 HFD41A 具有以下特性：5A 触电切换能力、具有一组转换触点形式、印制板式引出端、具有塑封型与防焊剂型两种封装形式、F 级绝缘等级、符合 RoHS 环保产品。其六个管脚分别为线圈、公共端、常开、常闭，实物图和管脚图如图 2-7 所示。

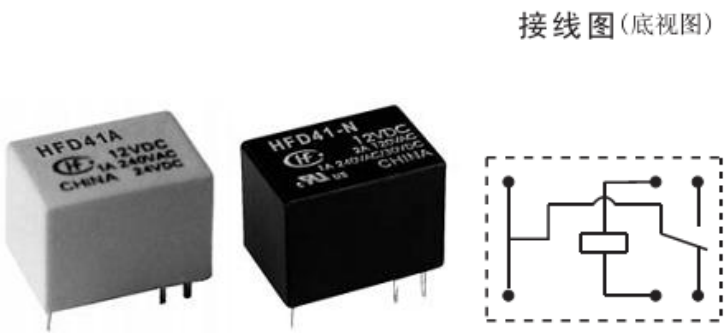

图 2- 7　继电器 HFD41A 实物图与管脚图

4、蜂鸣器 HYDZ

蜂鸣器分有源与无源两种，本项目中使用的蜂鸣器为有源蜂鸣器，也称直流蜂鸣器。有源蜂鸣器实物图与管脚图如图 2- 8 所示。

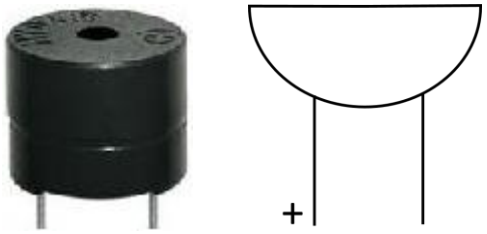

图 2- 8　有源蜂鸣器实物图与管脚图

三、操作训练

请按照电路原理图与 PCB 装配图焊接 PCB 板。其 PCB 板装接工艺要求如下：

1、装接工艺要求：

1.1　贴片元件的安装工艺

贴片电阻、贴片电容、贴片三极管水平安装，贴紧印制板。电阻标号方向应一致。

1.2　贴片集成电路安装工艺

贴片集成电路水平安装，贴紧印制板，安装时请注意集成电路方向，避免装反。

1.3　直插元件安装工艺

蜂鸣器、晶振、插针等直插式元件请直立安装，插到底。

1.4　直插集成电路安装工艺

直插式集成电路安装，请先安装好底座，再将集成电路插入。安装时请注意集成电路底座方向与装配图一致，避免装反。

1.5　其他工艺要求

所有插入焊盘孔的元件引脚及导线均采用直脚焊，剪脚留头在焊面以上0.5mm~1mm 未述之处均按常规工艺。

2、电路原理图

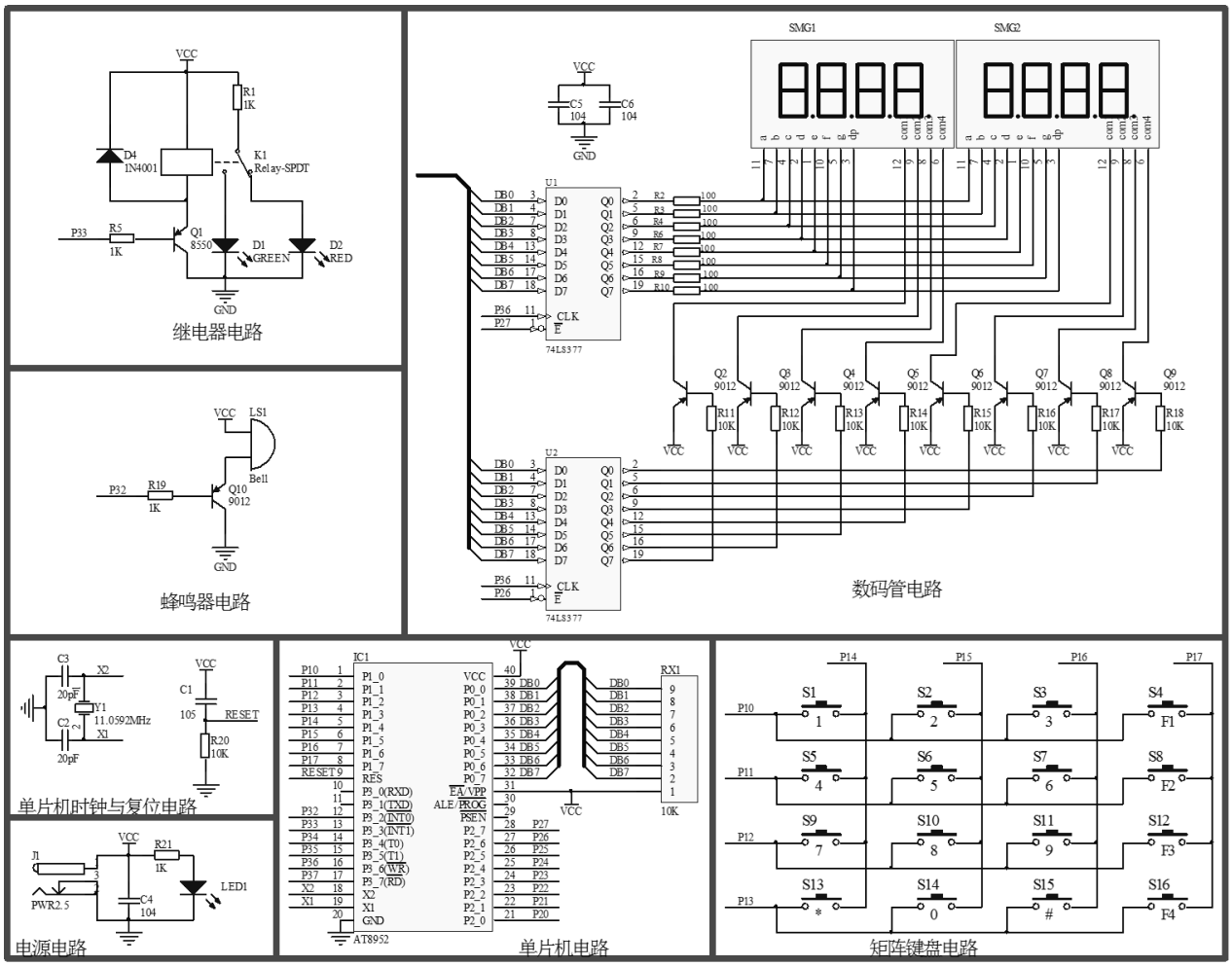

图 2- 9　密码锁原理图

3、PCB 板装配图

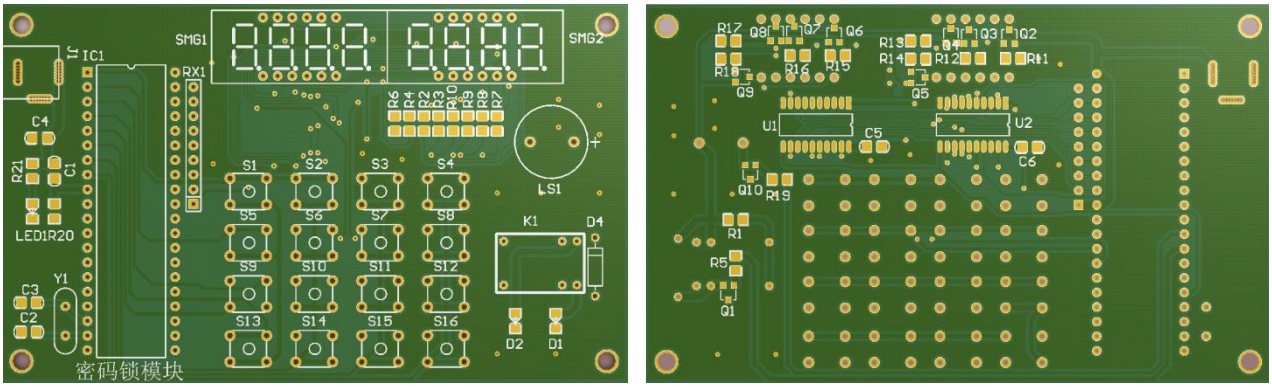

图 2- 10　PCB 板装配图

4、元器件清单

元器件清单如下表所示：

表 2- 1　密码锁需装配元器件清单

序号	元件名	参数	标号	封装	数量	备注
1	电容	104	C5, C6	0805C	2	
2	发光二级管	GREEN	D1	LED-SM	1	
3	发光二级管	RED	D2	LED-SM	1	
4	二极管	1N4001	D4	diode-0.4	1	
5	单排座		IC1	40 针	1	
6	电源接口	PWR2.5	J1	POWER 接口 1	1	
7	继电器	Relay-SPDT	K1	继电器 6 脚	1	
8	无源蜂鸣器	Bell	LS1	BELL	1	
9	三极管	8550	Q1	SOT-23	1	
10	三极管	9012	Q2,Q3,Q4,Q5,Q6, Q7,Q8,Q9,Q10	SOT-23	9	
11	电阻	1K	R1,R5, R19	0805R	3	
12	电阻	100	R2,R3,R4,R6,R7, R8,R9, R10	0805R	8	
13	电阻	10K	R11,R12,R13,14 R15,R16, R17,R18	0805R	8	
14	按键		S1~S16	AN66	16	
15	数码管	SMG	SMG1,SMG2	数码管 X4-0.36	2	
16	集成电路	74LS377	U1, U2	SOP20 窄体	2	

若使用单片机最小系统进行调试，下表中器件不需安装。若直接安装单片机

在 PCB 板上，则下表中元件必须安装。

表 2- 2　密码锁非必须装配元器件

序号	元件名	参数	标号	封装	数量	备注
17	电容	105	C1	0805C	1	不焊
18	电容	20pF	C2, C3	0805C	2	不焊
19	电容	104	C4	0805C	1	不焊
20	发光二级管	红色	LED1	LED-SM	1	不焊
21	电阻	1K	R21	0805R	1	不焊
22	电阻	10K	R20	0805R	1	不焊
23	排阻	10K	RX1	rx	1	不焊
24	晶振	11.0592MHz	Y1	X1	1	不焊

四、任务评价

表 2- 3　任务一完成情况汇总表

项目名称			评价时间		年　月　日		
小组名称		小组成员					
评价内容	评价要求	权重	评价标准	学生自评得分	小组评价得分	教师评价得分	合计
职业与安全意识	1. 工具摆放、操作符合安全操作规程; 2. 遵守纪律、爱惜设备和器材、工位整洁; 3. 具有团队协作精神。	10%	好（10） 较好（8） 一般（6） 差（<6）				
印制线路板插件	1、插件正确，电阻标志方向一致。 2、元器件高度符合工艺要求，元器件装插平整。	15%	好（15） 较好（12） 一般（9） 差（<9）				
印制线路板焊接	焊点光亮、焊料适量，无虚焊、漏焊、假焊、搭锡、溅锡、铜箔起翘等现象,无毛刺、孔隙留脚长度，焊面以上0.5mm~1mm。	60%	好（60） 较好（45） 一般（30） 差（<30）				
印制线路板的总装	装配正确，无烫伤、划伤工件和导线，紧固件装配牢固可靠，导线剥头尺寸符合工艺要求绝缘恢复良好。	15%	好（15） 较好（12） 一般（9） 差（<6）				
教师签名			学生签名			总分	
项目评价=学生自评（0.2）+小组评价（0.3）+教师评价（0.5）							

五、思考与练习

请写一份装配报告，注明安装中遇到的问题与思考。

任务二　数码管显示器的数字显示

一、任务要求

在任务一中制作完成的板子上实现本任务，找到数码管，并且编程实现在数码管上左起显示出自己的八位学号（不足位补零）。要求整个显示过程无闪烁无抖动，亮度足够。

二、相关知识

在单片机应用系统中，数码管系发光器件的一种，其内部由七个条形发光二极管和一个小圆点发光二极管组成，根据各管的亮暗组合成字符。

1、数码管的结构

数码管是由发光二极管组成的显示器件，一般由八个发光二极管构成字段，通过字段不同的亮灭组合来显示数字、字母等字符。由于去掉小数点剩余七个，俗称七段数码管。常见的数码管有 10 根管脚，管脚排列如图 2- 11 所示，其中 com 为公共端，a-g、dp 为输入端，根据内部发光二极管的接线形式可分为共阴极和共阳极两种。共阴极连接方式（图 2- 12）由于 com 端接低电平，因此输入管脚高电平时相应发光二极管会被点亮，共阳极连接方式（图 2- 13）由于 com 端接高电平，因此输入管脚低电平时相应发光二极管会被点亮。因此在使用时要注意，共阴极数码管公共端接地，共阳极数码管公共端接电源。每段发光二极管需 5~10mA 的驱动电流才能正常发光，一般需加限流电阻控制电流的大小以免发光二极管被烧毁。

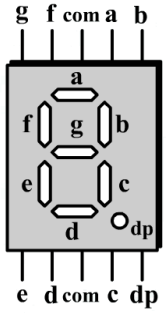

图 2- 11　数码管管脚图

数码管的引脚如图，有两种连接方式：共阴极和共阳极。

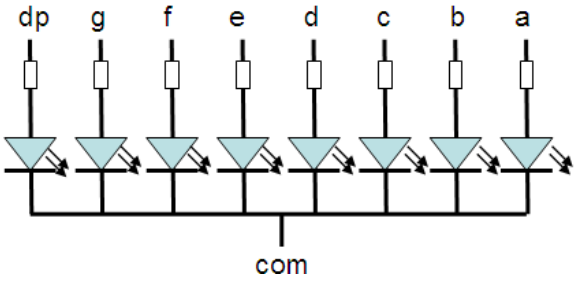

图 2- 12　共阴极连接方式

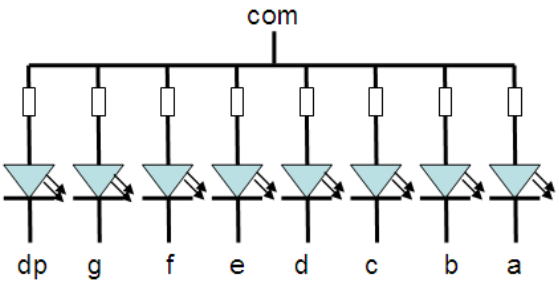

图 2- 13　共阳极连接方式

2、数码管显示原理

数码管的显示原理：要显示某字形，需将此字形对应的字段点亮，因而要送不同的电平组合数据至数码管引脚，该组合数据称为段码。显然共阴极和共阳极的段码是不同的。如图 2- 14 所示，需要一个数码管显示数字 2，由图可知，数码管 a、b、g、e、d 五个输入端应当点亮，由共阴极和共阳极显示原理（共阴极 1 点亮，共阳极 0 点亮）得到表 2- 4，将其转换为十六进制即可。

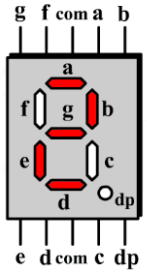

图 2- 14　数码管显示 2

表 2- 4　段码：以显示数字 2 为例

段码	dp	g	f	e	d	c	b	a
共阳	1	0	1	0	0	1	0	0
共阴	0	1	0	1	1	0	1	1

由表 2- 4 可知：数字 2 的段码，共阳极为 0xa4，共阴极为 0x5b。根据这个原理，可以得出共阳极、共阴极连接方式数字 0~9 显示的段码（表 2- 5、表 2-

6)。在程序中只需将该段码直接写入某一个口，就能显示相应的字符。

表 2- 5　共阳极数字段码表

字符 / 编码		0	1	2	3	4	5	6	7	8	9
dp	P0.7	1	1	1	1	1	1	1	1	1	1
g	P0.6	1	1	0	0	0	0	0	1	0	0
f	P0.5	0	1	1	1	0	0	0	1	0	0
e	P0.4	0	1	0	1	1	1	0	1	0	1
d	P0.3	0	1	0	0	1	0	0	1	0	0
c	P0.2	0	0	1	0	0	0	0	0	0	0
b	P0.1	0	0	0	0	0	1	1	0	0	0
a	P0.0	0	1	0	0	1	0	0	0	0	0
段码		0XC0	0XF9	0XA4	0XB0	0X99	0X92	0X82	0XF8	0X80	0X90

表 2- 6　共阴极数字段码表

字符 / 编码		0	1	2	3	4	5	6	7	8	9
dp	P0.7	0	0	0	0	0	0	0	0	0	0
g	P0.6	0	0	1	1	1	1	1	0	1	1
f	P0.5	1	0	0	0	1	1	1	0	1	1
e	P0.4	1	0	1	0	0	0	1	0	1	0
d	P0.3	1	0	1	1	0	1	1	0	1	1
c	P0.2	1	1	0	1	1	1	1	1	1	1
b	P0.1	1	1	1	1	1	0	0	1	1	1
a	P0.0	1	0	1	1	0	1	1	1	1	1
段码		0X3F	0X06	0X5B	0X4F	0X66	0X6D	0X7D	0X07	0X7F	0X6F

例：如果把一个共阳极的数码管接到单片机的 P0 口上，应当怎样写程序呢？

数码管显示程序　　display.c

```
#include"reg51.h"
main()
{
    while(1)
    {
        P0=0xa4;//直接赋值，将段码赋给所对应的 I/O 口
    }
}
```

3、数码管的片选

在任务中由于要同时显示多个数码管，这就涉及到数码管的片选信号，比如需要点亮某个数码管，就要把对应的 I/O 口选中。

4、数码管的显示方式

数码管的显示方式有两种：静态显示和动态显示。所谓静态显示，是指当显示器显示某个字符时，相应段的发光二极管处于恒定的导通或截止状态，直到需要显示另一个字符为止。

静态显示的接口电路采用一个并行口接一个数码管，数码管的公共端按共阴极或共阳极分别接地或接 Vcc。这种接法，每个数码管都要单独占用一个并行 I/O 口，以便单片机传送段码到数码管控制数码管的显示。显然其缺点就是当显示位数多时，占用 I/O 口过多。

静态显示方式的优点是显示的数据稳定，无闪烁，占用 CPU 时间少，其缺点是由于数码管时钟发光，功耗比较大。

动态显示的基本原理是利用人眼的“视觉暂留”效应和发光二极管的余晖现象来工作的。所谓动态就是利用循环扫描的方式，分时轮流选通各显示器的公共端，使各个显示器轮流导通。当扫描速度达到一定程度时，人眼就分辨不出来了，认为是各个显示器同时发光。

在编程时，需要输出段选和位选信号，位选信号选中其中一个数码管，然后输出段码，使该数码管显示所需要的内容，延时一段时间后，再选中另一个数码管，再输出对应的段码，高速交替。例如需要显示数字“12”时，先输出位选信号，选中第一个数码管，输出 1 的段码，延时一段时间后选中第二个数码管，输出 2 的段码。把上面的流程以一定的速度循环执行就可以显示出“12”，由于交替的速度非常快，人眼看到的就是连续的“12”在动态显示程序中。各个位的延时时间长短是非常重要的，如果延时时间长，则会出现闪烁现象；如果延时时间太短，则会出现显示暗且有重影。

动态扫描方法的连接方式是用一个接口电路把所有数码管的 8 个笔划段 a~g、dp 同名端连在一起，而每一个数码管的公共端 com 各自独立的受一条 I/O 口控制。CPU 向字段输出口送出段码时，所有数码管接收到相同的段码，但究竟是哪

个数码管亮，则取决于 com 端，com 端与单片机的 I/O 口相连接，由单片机输出位选信号到 I/O 口控制何时点亮某个数码管。动态显示的优点是当显示的位数比较多时，采用该方式比较节省 I/O 口，硬件电路也较静态显示简单，但是其缺点是稳定度不如静态显示方式，而且在显示位数较多的时候 CPU 要轮番扫描，占用 CPU 较多的时间。

三、操作训练

1、任务分析

根据前面章节数码管的介绍可知，要使用一个数码管显示数字（静态显示），需要用到单片机 8 个 I/O 口，而本任务要求 8 个数码管同时显示，总共需要 8*8=64 个 I/O 口，而 51 单片机只有 32 个 I/O 口，所以在本任务中需要用到数码管的动态显示。

2、补充知识——数组

本任务中，数码管的动态扫描程序需要多次用到数字 0-9 的段码，所以考虑先定义字形编码表，程序中直接调用即可，这就涉及到数组的概念。

数组是一个由若干同类型变量组成的集合，引用这些变量时可用同一名字。数组均由连续的存储单元组成，最低地址对应于数组的第一个元素，最高地址对应于最后一个元素，数组可以是一维的，也可以是多维的。

用一个下标来区分其元素的数组，称为“一维数组”；用两个或多个下标来区分其元素的数组，称为“二维数组”或“多维数组”。

在程序中说明一个数组后，系统就为它在内存分配一个连续的存储区，顺序存放该数组中的元素。这个存储区所需要的字节数，按如下公式计算：

总字节数=数组元素个数×数据类型长度

2.1 一维数组的定义

在 C 语言中，数组定义和数组成员的标志为方括号[ ]，数组名表示数组存储的首地址，访问数组成员可采用“数组名[下标]”的访问方式。使用 C 语言声明一个一维数组的格式如下：

类型说明符　数组名[常量表达式]；

为了定义一个名称为 M7G 的具有 10 个无符号字符型数据元素的数组，可以使用下面的程序实现：

unsigned char M7G[10]；

该数组共有 10 个元素，每个元素由不同的的下标表示，分别为 M7G[0]，M7G[1]，M7G[2]，M7G[3]，M7G[4]，M7G[5]，M7G[6]，M7G[7]，M7G[8]，M7G[9]。在 C 语言中，数组第一个元素的下标为 0 而非 1，最后一个元素的下标为“常量表达式”-1 的值。

数组名的命名规则和变量名是一样的。常量表达式表示元素的个数，即数组长度。声明一个数组后，51 编译器就为其分配相应的存储空间，一个一维数组占用的存储空间大小为：

数组数据类型占用字节数×数组长度（或数组的元素个数）

2.2 一维数组的初始化和引用

所谓一维数组的初始化，即指在说明数组的同时为其诸元素(变量)赋初值。完整的数组说明语句格式为：

<存储类型><数据类型><数组名>[长度]={<常量 1>，<常量 2>，…};

其中<常量 1>是数组第 1 个元素的值，<常量 2>是数组第 2 个元素的值，<常量 3>是数组第 3 个元素的值，如此等等。

比如，有如下数组说明语句：float f[4] = {0.1，1.1，2.1，3.1};表示名为 f 的数组有 4 个元素，存储类型是 auto，数据类型是 float，各元素的初值为：f[0]=0.1，f[1]=1.1，f[2]=2.1，f[3]=3.1。

关于数组元素初始化的几点注意：

若说明时是对数组的所有元素赋初值，那在数组说明中<长度>可省略(但是数组名后面的方括号不能没有，即显示空)。如 unsigned char M7G[]={0xc0，0xf9，0xa4，0xb0，0x99，0x92，0x82，0xf8，0x80，0x90}；这个语句表示定义了无符号型数组 M7G，一共有 10 个元素。

若数组说明时给出<长度>，但只依次为前几个元素赋了初值。那 C 语言将自动对余下元素赋初值：为数值型的赋 0(或 0.0)；为字符型的赋空字符。

若数组的存储类型是 static 的，那么该数组所有元素都是静态（static）型变量。

若数组说明时给出了<长度>，并对元素进行了初始化，那所列出的元素初始值的个数，不能多于数组元素的个数。否则 C 语言就会判定为语法错。

对数组元素引用的时候直接调用第 N 个元素即可，如 a=M7G[0]表示把 M7G 数组中第 0 个元素赋值给变量 a。

2.3 二维数组的定义和引用

定义格式：类型说明符　数组名[长度 1][长度 2]；

引用格式：数组名[下标 1][下标 2]　　下标都是从 0 开始；

二维数组可以理解为是一个特殊的一维数组，它包含的元素又是一个一维数组。比如，“int a[3][4];”说明了名为 a 的二维整型数组：该数组共有 3*4=12 个元素，每个都是 int 型变量。第 1 个下标从 0 变到 2，第 2 个下标从 0 变到 3。这 12 个元素的名称是：

a
- a[0] 包含 a[0][0]　a[0][1]　a[0][2]　a[0][3]
- a[1] 包含 a[1][0]　a[1][1]　a[1][2]　a[1][3]
- a[2] 包含 a[2][0]　a[2][1]　a[2][2]　a[2][3]

为处理二维数组，C 语言先把二维数组看成是有<长度 1>这么多个元素的一维数组，每个元素的名为：<数组名>[0]，<数组名>[1]，…，<数组名>[<长度 1>−1]。然后再把该一维数组的每个元素看作是有<长度 2>这么多个元素的一维数组。

这样，数组 a 先视为有 3 个元素的一个一维数组，其元素名分别是：a[0]，a[1]，a[2] (其实就是 3 行)。随之，a[0]是有 4 个元素的一维数组，分别是：a[0][0]，a[0][1]，a[0][2]，a[0][3]；a[1]是有 4 个元素的一维数组，分别是：a[1][0]，a[1][1]，a[1][2]，a[1][3]；a[2]是有 4 个元素的一维数组，分别是：a[2][0]，a[2][1]，a[2][2]，a[2][3]。

2.4 二维数组元素的初始化和引用

在说明二维数组的同时，对每个元素赋予初始取值，这就是所谓的“二维数组的初始化”。

二维数组的初始化，有如下几种方法：

1、分行对二维数组进行初始化。比如语句：

```
int a[3][4]={{1,2,3,4},{5,6,7,8},{9,10,11,12}};
```

说明了一个名为 a 的 int 型二维数组，共有 12 个元素。

2、不分行将所有数据依次列在一个花括号中。比如语句：

```
int a[3][4]={1,2,3,4,5,6,7,8,9,10,11,12};
```

3、分行进行初始化时，可只对部分元素赋初值，剩余元素的初值由系统自动补齐：若是数值型的，就赋予 0(或 0.0)；若是字符型的，就赋予空字符。比如语句：

```
int a[3][4]={{1},{4},{11}};
```

4、若是对二维数组的全部元素进行初始化，那在数组说明语句中，<长度 1>可以省略不写(方括号还是要的)。比如语句：

```
int a[][4]={1,2,3,4,5,6,7,8,9,10,11,12};
```

等同于语句：

```
int a[3][4]={{1,2,3,4},{5,6,7,8},{9,10,11,12}};
```

或：

```
int a[3][4]={1,2,3,4,5,6,7,8,9,10,11,12};
```

二维数组元素的引用与一维数组类似，这里不做详述。

3、流程图

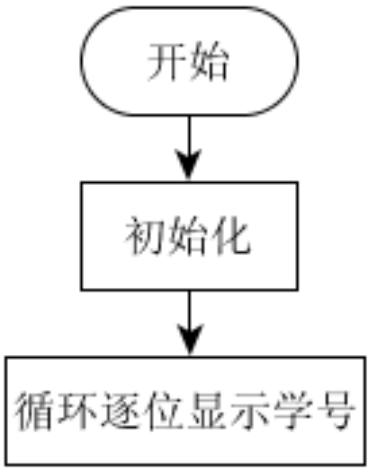

图 2- 15　流程图

4、参考程序

数码管显示程序 SHUMAGUANdsp.C

```
#include<reg51.h>
unsigned char code M7G[]={0xc0,0xf9,0xa4,0xb0,0x99, 0x92,0x82,0xf8,0x80,0x90};
//定义段码数组
unsigned char str[8]={0x01,0x02,0x04,0x08,0x10,0x20,0x40,0x80};
//定义片选信号数组
unsigned char xuehao[]={0,8,0,4,0,6,1,8};
unsigned int i;
unsigned char n；
unsigned char xdata          LED_DM    _at_ 0x7FFF;
unsigned char xdata          LED_PX    _at_ 0xBFFF;
void main()
{
    while(1)
    {
        DM=M7G[xuehao[n]]；//选择段码
        PX=str[n];//片选信息
        for(i=0;i<10000;i++);//延时
        n=(n+1)%8;
    }
}
```

5、程序说明

在上述程序中，主要是由

DM=M7G[xuehao[n]];

PX=str[n];

for(i=0;i<10000;i++);

这三句话循环，第一句是选择段码信息，第二句是选择片选信息，第三句是延时以利用人眼的“视觉暂留”效应使得看起来好像多个数码管都被点亮。

6、任务实施

6.1 建立工程

打开 keil 软件，通过菜单“Project”，新建立一个工程“new Project”项目 SHUMAGUANdsp，然后再建一个文件名为 SHUMAGUANdsp.C 的源程序文件，将上面的参考程序输入并保存。建立的 SHUMAGUANdsp.C 文件添加入本项目中。

6.2 编译并生成 HEX

单击“Build target”按钮，对源程序进行编译和链接，产生 HEX 文件。

6.3 烧录芯片

打开 STC-ISP 下载软件，选中单片机型号为“STC90C52RC”，选择最低波特率为 2400 最高波特率为 115200（为默认值一般不需修改）。点击打开程序文件按钮，在刚才建立的 SHUMAGUANdsp 文件夹中生成的 SHUMAGUANdsp.HEX 文件。然后点击“下载/编程”按钮。最后给最小系统通电，等待电脑端串口与单片机握手成功。就能把程序载入到单片机中去了。其详细操作图如图 2- 16 所示：

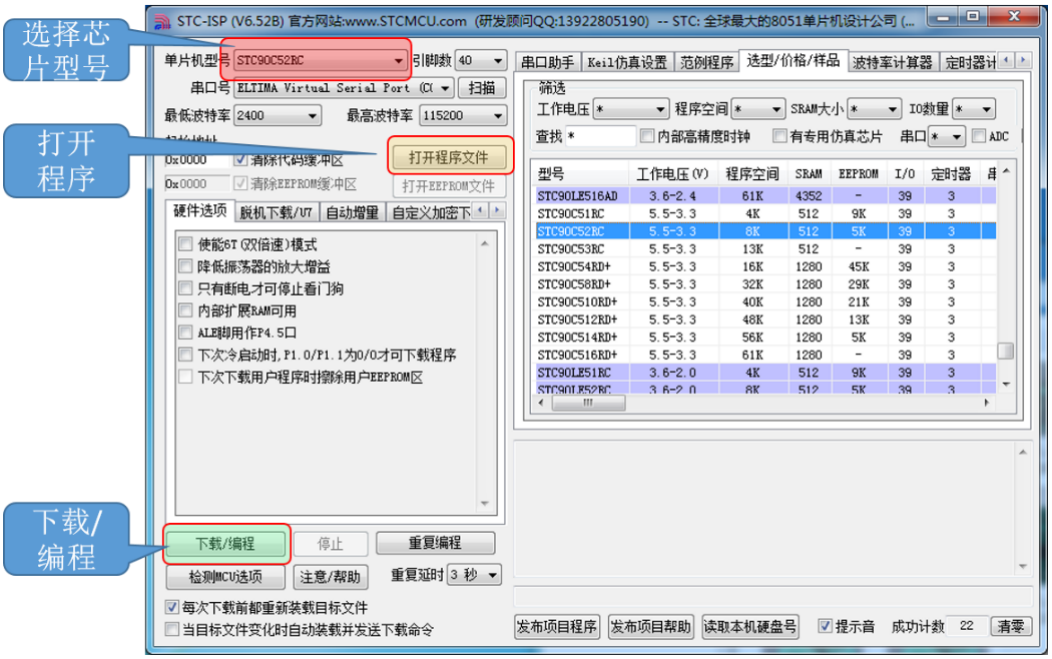

图 2- 16　STC 单片机程序烧写示意图

6.4 硬件调试

程序下载完成后，把单片机最小系统接入电路中，调试电路使其能正常工作，并在数码管上显示完整学号。

四、任务评价

表 2- 7　任务二完成情况评价表

项目名称			评价时间		年　月　日		
小组名称		小组成员					
评价内容	评价要求	权重	评价标准	学生自评得分	小组评价得分	教师评价得分	合计
职业与安全意识	1．工具摆放、操作符合安全操作规程；2．遵守纪律、爱惜设备和器材、工位整洁；3．具有团队协作精神。	10%	好（10） 较好（8） 一般（6） 差（<6）				
印制线路板调试	1．能够根据任务要求进行印制线路板的调试；2．根据任务要求，能够对对应电路部分进行硬件电路的检查与故障检修。	15%	好（15） 较好（12） 一般（9） 差（<9）				
项目功能测试	1．编写的程序能成功编译；2．程序能正确烧写到芯片中；3．系统启动后数码管能正常显示字符。	60%	好（60） 较好（45） 一般（30） 差（<30）				
问题与思考	1．动态显示的延时时间应该如何选择；2．如果需要数码管显示一些特殊字符怎么修改程序；	15%	好（15） 较好（12） 一般（9） 差（<6）				
教师签名			学生签名			总分	
项目评价=学生自评（0.2）+小组评价（0.3）+教师评价（0.5）							

五、知识拓展

以上我们完成了一个简单的数码管显示任务，0-9 的数字我们都能显示出来，但是在实际应用中经常会遇到显示一些特殊字符，如“A”、“F”、“P”等，应该怎么做呢？

前面已经介绍了数码管的段码概念，显示字符和显示数字的方法是一样的，以字母 F 为例，数码管 a、e、f、g 四个输入端应当点亮，如表 2-8　段码：以显示字母 F 为例所示：

表 2-8　段码：以显示字母 F 为例

段码	dp	g	f	e	d	c	b	a
共阳	1	0	0	0	1	1	1	0
共阴	0	1	1	1	0	0	0	1

由表 2-8　段码：以显示字母 F 为例可知：数字 2 的段码，共阳极为 0x8e，共阴极为 0x71。

其他特殊字符的显示方式以此类推。

六、思考与练习

1、如果把动态显示的延时时间缩短到一定程度会有什么现象？

2、本任务的显示方式是左起显示，如果需要靠右显示，程序应当如何更改？

任务三　数码管按键键值显示

一、任务要求

利用矩阵键盘和数码管完成一个按键键值显示任务。要求：

1、系统上电后，数码管不显示；

2、按下按键，数码管最右端显示该键键值；

3、按下另一个按键，数码管更新显示。

矩阵式键盘各按键对应键值如图 2- 17 所示。

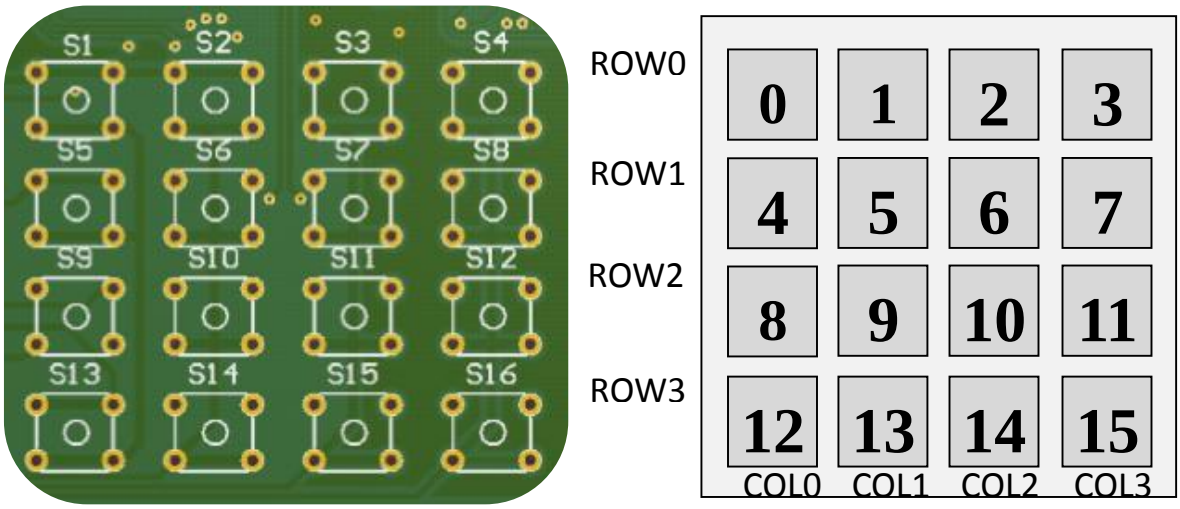

图 2- 17　按键与键值对应图

二、相关知识

2.1　矩阵键盘介绍

键盘有两类，一类是独立键盘，另一类是行列式键盘，即矩阵键盘。

2. 1. 1　什么是矩阵式键盘？

当键盘中按键数量较多时，为了减少 I/O 口线的占用，通常将按键排列成矩阵形式。在矩阵式键盘中，每条水平线和垂直线在交叉处不直接连通，而是通过一个按键加以连接。这样做有什么好处呢？一个并行口（8 位）可以构成 4*4=16 个按键，比直接将端口线连接到键盘多出了一倍，而且线数越多，区别就越明显。比如再多加一条线就可以构成 20 键的键盘，而直接用端口线则只能多出一个键

（9 键）。由此可见，在需要的按键数量比较多时，采用矩阵法来连接键盘是非常合理的。

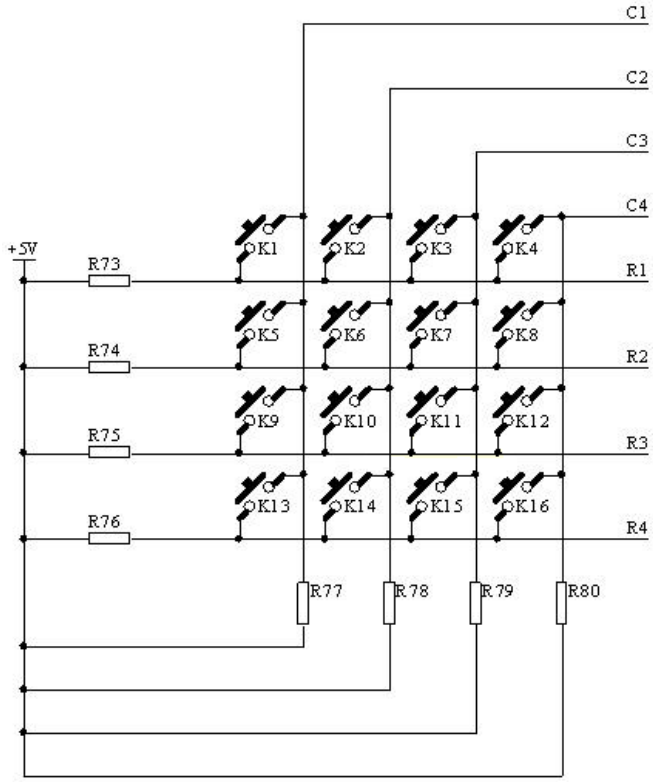

图 2- 18　矩阵键盘示意图

矩阵式结构的键盘显然比独立式键盘复杂一些，识别也要复杂一些，在上图中，列线通过电阻接电源，并将行线所接的单片机 4 个 I/O 口作为输出端（R1~R4），而列线所接的 I/O 口则作为输入端（C1~C4）。这样，当按键没有被按下时，所有的输出端都是高电平，代表无键按下，行线输出是低电平；一旦有键按下，则输入线就会被拉低，这样，通过读入输入线的状态就可得知是否有键按下了，具体的识别及编程方法如下所述：

2. 1. 2　矩阵式键盘的按键识别方法

确定矩阵式键盘上任何一个键被按下通常采用“行扫描法”或者“行反转法”。行扫描法又称为逐行（或列）扫描查询法，它是一种最常用的多按键识别方法。因此我们就以“行扫描法”为例介绍矩阵式键盘的工作原理：

1、判断键盘中有无键按下将全部行线 R1-R4 置低电平，然后检测列线的状态，只要有一列的电平为低，则表示键盘中有键被按下，而且闭合的键位于低电平线与 4 根行线相交叉的 4 个按键之中；若所有列线均为高电平，则表示键盘中无键按下。

2、判断闭合键所在的位置在确认有键按下后，即可进入确定具体闭合键的过程。其方法是：依次将行线置为低电平（即在置某根行线为低电平时，其它线为高电平），当确定某根行线为低电平后，再逐行检测各列线的电平状态，若某列为低，则该列线与置为低电平的行线交叉处的按键就是闭合的按键。

下面给出一个具体的例子：单片机的 P1 口用作键盘 I/O 口，键盘的列线接到 P1 口的低 4 位，键盘的行线接到 P1 口的高 4 位，也就是把列线 P1.0~P1.3 分别接 4 个上拉电阻到电源，把列线 P1.0~P1.3 设置为输入线，行线 P1.4~P1.7 设置为输出线，4 根行线和 4 根列线形成 16 个相交点，如图所示。检测当前是否有键被按下：检测的方法是 P1.4~P1.7 输出全“0”，读取 P1.0~P1.3 的状态，若 P1.0~P1.3 为全“1”，则说明无键闭合；否则有键闭合。去除键抖动：当检测到有键按下后，延时一段时间再做下一次的检测判断，若仍有键按下，应识别出是哪一个键闭合，方法是对键盘的行线进行扫描，P1.4~P1.7 按下述 4 种组合依次输出：P1.7　1110；P1.6　1101；P1.5　1011；P1.4 0111；在每组行输出时读取 P1.0-P1.3；若全为“1”，则表示为“0”这一行没有键闭合；否则就是有键闭合。由此得到闭合键的行值和列值，然后可采用计算法或查表法将闭合键的行值和列值转换成所定义的键值。为了保证按键每闭合一次 CPU 仅作一次处理，必须去除键释放时的抖动。

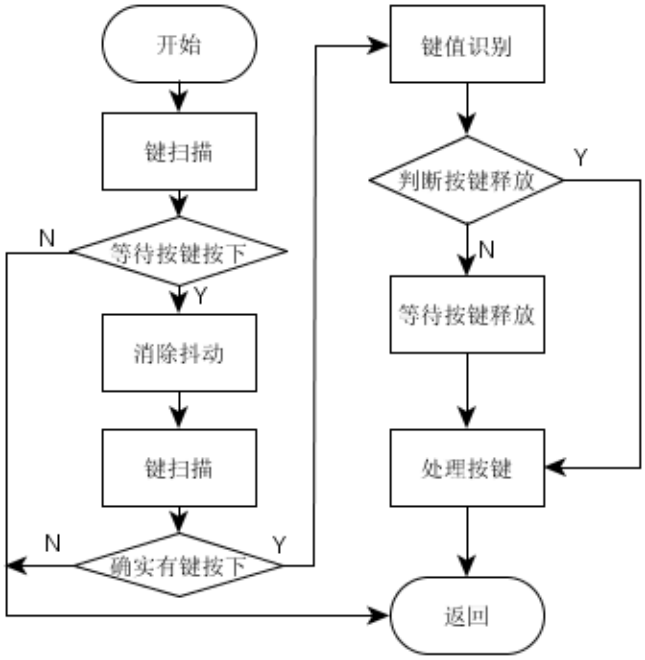

图 2- 19　矩阵键盘程序流程图

按键去抖动主要采用“软件消抖”，即检测出按键闭合后执行一个延时程序，

产生 5~10m 的延时，让前沿抖动消失后再一次检测按键的状态，如果仍保持闭合状态电平，则确认为真正有键按下。当检测到按键释放后，也要给 5~10ms 的延时，待后沿抖动消失后才能转入该键的处理程序。

三、操作训练

1、任务分析

矩阵式键盘子程序，此部分需要不停的调用以判断是否有键按下，所以应放到主程序中循环调用：

矩阵式按键子程序 key.c

```
unsigned char Key(void)
{
	unsigned char kt,temp;
	while(1)
	{
		KIO=0XF0;//高四位拉高
		if (KIO!=0xf0)//有键按下
		{
			if (++kt==2)//延时防抖
			{
				temp=KIO;
				KIO=temp|0x0f;//取键值
				switch (KIO)//判断
				{
					case 0x77:return 3; break;//3
					case 0x7b:return 7;    break;//7
					case 0x7d:return 11; break;//B
					case 0x7e:return 15; break;//F
					case 0xb7:return 2; break;//2
					case 0xbb:return 6; break;//6
					case 0xbd:return 10; break;//A
					case 0xbe:return 14; break;//E
					case 0xd7:return 1; break;//1
					case 0xdb:return 5; break;//5
					case 0xdd:return 9; break;//9
					case 0xde:return 13; break;//D
					case 0xe7:return 0; break;//0
```

```
                        case 0xeb:return 4; break;//4
                        case 0xed:return 8;    break;//8
                        case 0xee:return 12; break;//C
                        default:    break;
                    }
                }
            }
            else kt=0;//无键按下，kt 清零
        }
}
```

2、主程序流程图

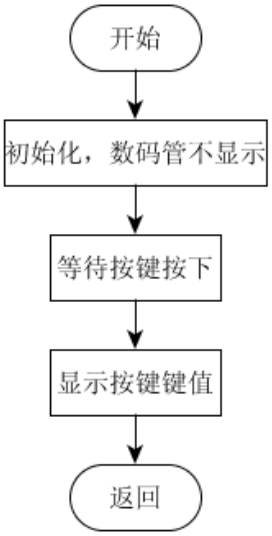

图 2- 20　按键键值显示流程图

3、参考程序

数码管按键键值显示程序　　SHUMAGUANkey.C

```
#include<reg51.h>
#define KIO P1
unsigned char xdata LED_DM _at_ 0x7FFF;
unsigned char xdata LED_PX _at_ 0xBFFF;
unsigned char code M7G[]=
{
      0xc0,//0
      0xf9,//1
      0xa4,//2
      0xb0,//3
      0x99,//4
```

```
    0x92,//5
    0x82,//6
    0xf8,//7
    0x80,//8
    0x90,//9
    0x88,//10   A
    0x83,//11 B
    0xc6,//12 C
    0xa1,//13 d
    0x86,//14 E
    0x8e,//15 F
};//定义段码数组
unsigned char str[8]={0x01,0x02,0x04,0x08,0x10,0x20,0x40,0x80};
//定义片选信号数组
unsigned char key(void);
void main()
{
    unsigned char temp;
    while(1)
    {
        temp=key();
        LED_DM=M7G[temp];//选择按键按下的键值对应的段码
        LED_PX=str[7];//片选信息，选择最右端数码管
    }
}
unsigned char Key(void)
{
    unsigned char kt,temp;
    while(1)
    {
        KIO=0XF0;
        if (KIO!=0xf0)//与后面 else 对应
        {
            if (++kt==2)
            {
                temp=KIO;
                KIO=temp|0x0f;
                switch (KIO)
                {
                    case 0x77:return 3; break;//3
                    case 0x7b:return 7; break;//7
```

```
                        case 0x7d:return 11;break;//B
                        case 0x7e:return 15;break;//F
                        case 0xb7:return 2; break;//2
                        case 0xbb:return 6; break;//6
                        case 0xbd:return 10;break;//A
                        case 0xbe:return 14;break;//E
                        case 0xd7:return 1; break;//1
                        case 0xdb:return 5; break;//5
                        case 0xdd:return 9; break;//9
                        case 0xde:return 13;break;//D
                        case 0xe7:return 0; break;//0
                        case 0xeb:return 4; break;//4
                        case 0xed:return 8;break;//8
                        case 0xee:return 12;break;//C
                        default:   break;
                    }
                }
            }
            else kt=0;
        }
}
```

4、程序说明

本程序在任务二的基础上加入了矩阵按键的功能，判断按键按下后还加入了软件延时去抖，确保按键按下不被误判。

5、任务实施

5.1 建立工程

打开 keil 软件，通过菜单“Project”，新建立一个工程“new Project”项目 SHUMAGUANkey，然后再建一个文件名为 SHUMAGUANkey.C 的源程序文件，将上面的参考程序输入并保存。建立的 SHUMAGUANkey.C 文件添加入本项目中。

5.2 编译并生成 HEX

单击  “Build target”按钮，对源程序进行编译和链接，产生 HEX 文件。

5.3 烧录芯片

打开 STC-ISP 下载软件，选中单片机型号为“STC90C52RC”, 选择最低波特率

为 2400 最高波特率为 115200（为默认值一般不需修改）。点击打开程序文件按钮，在刚才建立的 SHUMAGUANkey 文件夹中生成的 SHUMAGUANkey.HEX 文件。然后点击“下载/编程”按钮。最后给最小系统通电，等待电脑端串口与单片机握手成功。就能把程序载入到单片机中去了。其详细操作图如图 2- 21 所示：

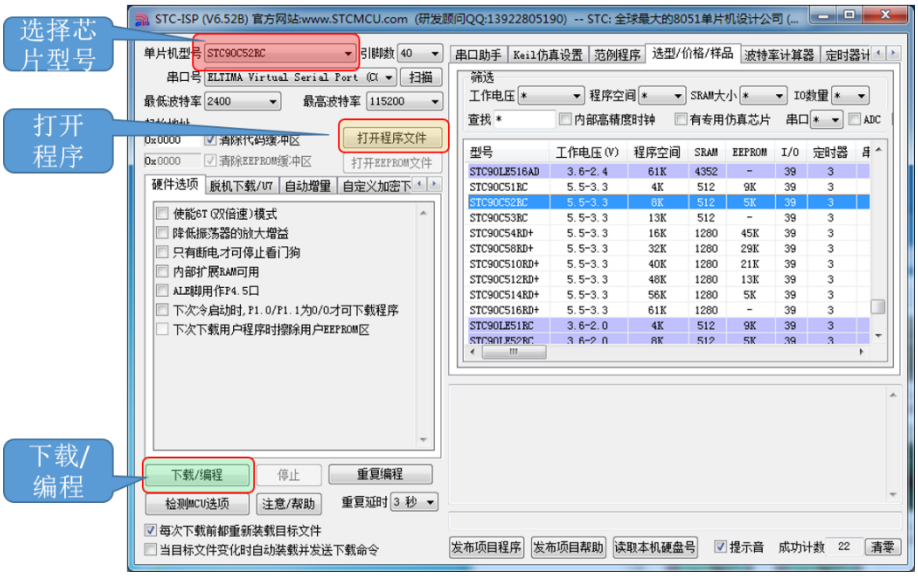

图 2- 21　STC 单片机程序烧写示意图

5.4　硬件调试

程序下载完成后，把单片机最小系统接入电路中，调试电路使其能正常工作，按键按下数码管显示递增。

四、任务评价

表 2- 9　任务三完成情况评价表

项目名称			评价时间		年　月　日		
小组名称		小组成员					
评价内容	评价要求	权重	评价标准	学生自评得分	小组评价得分	教师评价得分	合计
职业与安全意识	1. 工具摆放、操作符合安全操作规程；2. 遵守纪律、爱惜设备和器材、工位整洁；3. 具有团队协作精神。	10%	好（10） 较好（8） 一般（6） 差（<6）				
印制线路板调试	1. 能够根据任务要求进行印制线路板的调试；2. 根据任务要求，能够对对应电路部分进行硬件电路的检查与故障检修。	15%	好（15） 较好（12） 一般（9） 差（<9）				
项目功能测试	1. 编写的程序能成功编译；2. 程序能正确烧写到芯片中；3. 系统启动后数码管能正常显示字符。	60%	好（60） 较好（45） 一般（30） 差（<30）				
问题与思考	1. 如果需要数码管靠左显示应当如何修改程序；2. 如果计数器最大计数为8，第九次按下按键数码管显示0，应该如何更改程序。	15%	好（15） 较好（12） 一般（9） 差（<6）				
教师签名			学生签名			总分	
项目评价=学生自评（0.2）+小组评价（0.3）+教师评价（0.5）							

五、思考与练习

本任务只设置在最右端数码管显示，如果需要最左端数码管显示应当如何更改程序？

任务四　数码管倒计时秒表制作

一、任务要求

利用键盘完成数码管 10 秒倒计时秒表，要求如下：

1、系统上电，数码管显示“00-00-10”，不变化；

2、按下开始按键，数码管开始倒计时，每隔一秒变化一次，直至数码管上显示变为“00-00-00”，不再变化；

3、倒计时过程中任意时间按下复位按键，数码管显示“00-00-10”，并且重新开始倒计时显示。

该任务用的矩阵键盘对应按键如图 2- 22 所示。

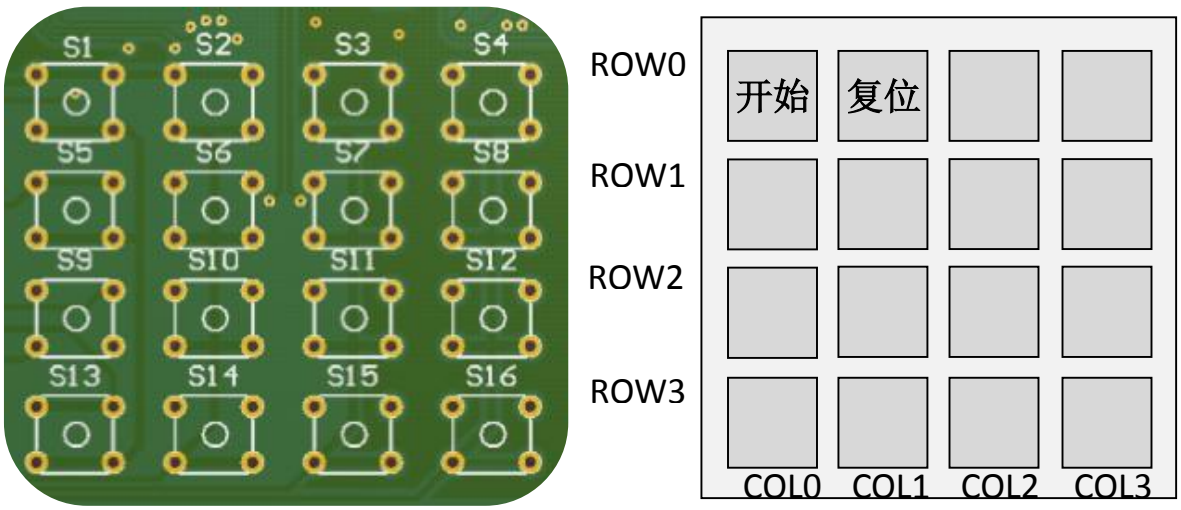

图 2- 22　矩阵式按键示意图

二、相关知识

本任务在任务三的基础上加入了计时功能，需要用到单片机自带的定时器，这部分内容已经在项目一中介绍，此处不再赘述。

三、操作训练

1、任务分析

计算是单片机最基本的功能， 单片机不仅具有位处理的能力，而且能够进行算术运算和逻辑运算，所以由单片机来完成本任务的计时功能是很容易的。

根据任务分析，单片机的主要任务是对按键的状态进行采集，然后控制计时器工作，通过显示电路显示计时器的数值。

本任务的硬件电路主要还是由矩阵按键和数码管显示两个模块组成，按键只需要开始键和复位键，数码管显示电路还是跟前面的任务一样。

软件编程主要由按键识别、计时、数码管进行动态扫描显示等部分组成。其中，计时是软件编程的核心。计时实现的方法有两种：一是通过软件延时来实现，二是通过单片机内部的定时器来实现。由于本项目对计时的精度要求不算太高，基本上都能满足要求，而单片机内部的定时器和 CPU 并行工作，具有软件执行效率高的特点，所以本任务的定时使用定时器中断方式来实现。具体实现的方法如下：

1.1　1s 计时精度的实现

设置 2ms 计时变量 ms，使用单片机定时器 T0 工作在定时方式 1 定时 2ms，每隔 2ms，ms 值加 1，直至 500 次，实现 1s 定时，此部分在中断服务子程序中完成。

1 秒计时子程序 sec.c

```
ms++;
if(ms>=500)//定时满 1 秒
{
      ms=0;//计时变量清零
      second--;//秒变量减一
      if(second==255)//减至 0
            second=0;      //保持为 0 不变
}
```

1.2　动态扫描实现：

使用显示子程序进行 8 位数码管的动态扫描。在定时中断服务子程序中调用该显示程序达到消隐作用，实现字符稳定显示。

数码管显示子程序 dsp.c

```
unsigned char xdata LED_DM _at_ 0x7FFF;
unsigned char xdata LED_PX _at_ 0xBFFF;
void    Display()
```

```
{
    static unsigned char dp_h;//定义静态变量 dp_h
    DM=0xff;//消隐
    PX=0xff;//不选择任何数码管
    DM=M7G[str[dp_h]];//选择当前要显示的段码
    PX (1<<dp_h);//选择当前要显示的片选信号
    dp_h++;
    dp_h&=0x07;//使得 dp_h 在 0-7 之间变化
}
```

1.3 主程序

循环查询按键状态并将计数变量通过数码管显示。

2、主程序流程图

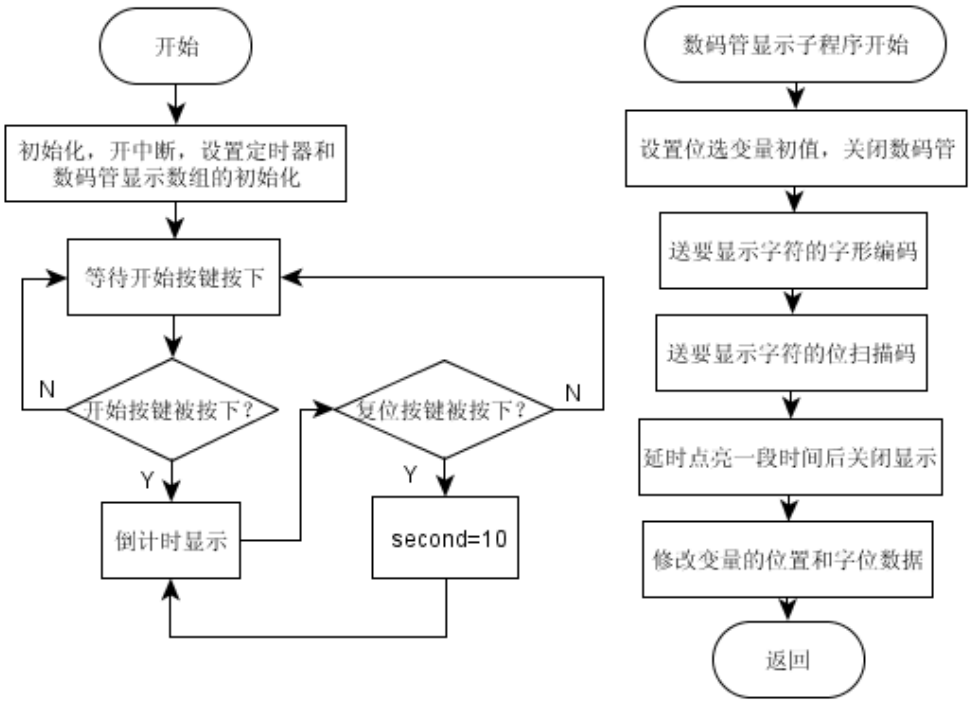

图 2- 23 数码管倒计时流程图

3、参考程序

数码管倒计时程序 SHUMAGUANtime.C

```
#include "reg51.h"
sbit fuwei=P3^0;//定义复位按键
sbit start=P3^1;//定义开始按键
```

```
unsigned char xdata LED_DM _at_ 0x7FFF;
unsigned char xdata LED_PX _at_ 0xBFFF;
unsigned char code M7G[]=
{
	0xc0,//0
	0xf9,//1
	0xa4,//2
	0xb0,//3
	0x99,//4
	0x92,//5
	0x82,//6
	0xf8,//7
	0x80,//8
	0x90,//9
	0xbf,//10-
};
unsigned char str[8]={0,0,10,0,0,10,1,0};//片选，选择某个数码管
unsigned char second;
unsigned int ms;
void intime()//实时显示子程序
{
	str[7]=second%10;//秒变化
	str[6]=second/10; //秒变化
	str[5]=10;//不变
	str[4]=0; //不变
	str[3]=0; //不变
	str[2]=10;//不变
	str[1]=0; //不变
	str[0]=0; //不变
}
void Display()//数码管显示子程序
{
	static unsigned char dp_h;
	DM=0xff;
	PX=0xff;
	DM=M7G[str[dp_h]];
	PX=(1<<dp_h);
```

```
    dp_h++;
    dp_h&=0x07;
}
unsigned char Key(void)
{
    unsigned char kt,temp;
    while(1)
    {
        KIO=0XF0;
        if (KIO!=0xf0)//与后面 else 对应
        {
            if (++kt==2)
            {
                temp=KIO;
                KIO=temp|0x0f;
                switch (KIO)
                {
                    case 0x77:return 3; break;//3
                    case 0x7b:return 7; break;//7
                    case 0x7d:return 11;break;//B
                    case 0x7e:return 15;break;//F
                    case 0xb7:return 2; break;//2
                    case 0xbb:return 6; break;//6
                    case 0xbd:return 10;break;//A
                    case 0xbe:return 14;break;//E
                    case 0xd7:return 1; break;//1
                    case 0xdb:return 5; break;//5
                    case 0xdd:return 9; break;//9
                    case 0xde:return 13;break;//D
                    case 0xe7:return 0; break;//0
                    case 0xeb:return 4; break;//4
                    case 0xed:return 8;break;//8
                    case 0xee:return 12;break;//C
                    default:    break;
                }
            }
        }
```

```
            else kt=0;
        }
}
void TIME0() interrupt 1//定时中断服务子程序
{
        TH0=0xf8;
        TL0=0xcc;
        ms++;
        if(ms>=500)//定时满 1s
        {
            ms=0;
            second--;
            if(second==255)
                second=0;
        }
        Display();
        intime();
}
int kt;
void main()
{
        TMOD=1;
        TH0=0xf8;//定时器 0，初值为 2ms
        TL0=0xcc;//定时器 0，初值为 2ms
        ET0=1;
        TR0=1;
      EA=1;//开中断
        while(1)
        {
            temp=key();
            if(temp==0)//开始按键被按下
                second=9;// 10 秒倒计时开始
            if(temp==11)//复位按键被按下
                second=1; //复位为 10 秒
        }
}
```

4、程序说明

本任务主要目的在熟练使用矩阵式按键和数码管显示的基础上，增加定时器的功能，与之前任务的不同在于把数码管显示放在定时中断服务程序中完成，按键功能放在主函数中完成。

5、任务实施

5.1　建立工程

打开 keil 软件，通过菜单“Project”，新建立一个工程“new Project”项目 SHUMAGUANtime，然后再建一个文件名为 SHUMAGUANtime.C 的源程序文件，将上面的参考程序输入并保存。建立的 SHUMAGUANtime.C 文件添加入本项目中。

5.2　编译并生成 HEX

单击“Build target”按钮，对源程序进行编译和链接，产生 HEX 文件。

5.3　烧录芯片

打开 STC-ISP 下载软件，选中单片机型号为“STC90C52RC”，选择最低波特率为 2400 最高波特率为 115200（为默认值一般不需修改）。点击打开程序文件按钮，在刚才建立的 SHUMAGUANtime 文件夹中生成的 SHUMAGUANtime.HEX 文件。然后点击“下载/编程”按钮。最后给最小系统通电，等待电脑端串口与单片机握手成功。就能把程序载入到单片机中去了。其详细操作图如图 2-24 所示：

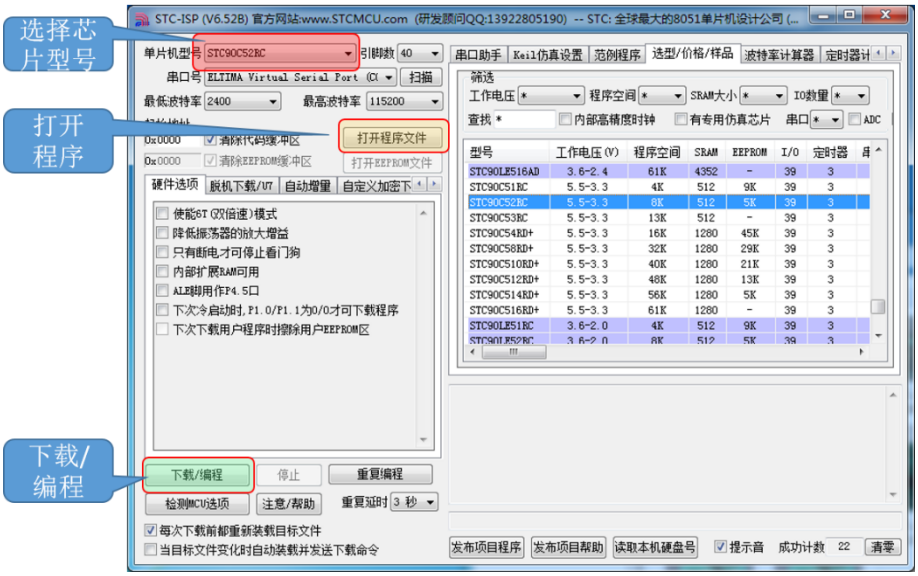

图 2-24　STC 单片机程序烧写示意图

5.4 硬件调试

程序下载完成后，把单片机最小系统接入电路中，调试电路使其能正常工作，并完成本任务。

四、任务评价

表 2- 10 任务四完成情况评价表

项目名称			评价时间		年 月 日		
小组名称		小组成员					
评价内容	评价要求	权重	评价标准	学生自评得分	小组评价得分	教师评价得分	合计
职业与安全意识	1. 工具摆放、操作符合安全操作规程；2. 遵守纪律、爱惜设备和器材、工位整洁；3. 具有团队协作精神。	10%	好（10） 较好（8） 一般（6） 差（<6）				
印制线路板调试	1. 能够根据任务要求进行印制线路板的调试；2. 根据任务要求，能够对对应电路部分进行硬件电路的检查与故障检修。	15%	好（15） 较好（12） 一般（9） 差（<9）				
项目功能测试	1. 编写的程序能成功编译；2. 程序能正确烧写到芯片中；3. 系统启动后数码管能正常显示字符。	60%	好（60） 较好（45） 一般（30） 差（<30）				
问题与思考	1．如果倒计时时间为20s，应当如何修改程序。	15%	好（15） 较好（12） 一般（9） 差（<6）				
教师签名			学生签名			总分	
项目评价=学生自评（0.2）+小组评价（0.3）+教师评价（0.5）							

五、知识拓展

数码管的动态扫描时间间隔并不是没有要求，扫描频率太低数码管会出现闪烁的现象，频率太高则亮度不够甚至无法看清，所以一般扫描间隔多为几毫秒。

一般间隔 1ms 就可以（本任务就是使用的 1ms），如果不够亮可以适当地增大间隔时间，不会有影响。但是，有一点需要注意，从点亮第一个数码管到最后一个数码管被点亮，整个过程最好不要超过 20ms，一旦超过则会出现闪烁的现象。

另外，点亮一个数码管后，在点亮第二个数码管之前需要先关断第一个数码管，否则会出现显示混乱的情况。

六、思考与练习

1．如果倒计时时间为 20s，应当如何修改程序。

任务五　数码管电子钟制作

一、任务要求

利用矩阵式键盘完成数码管电子钟的制作，要求如下：

1、系统上电，数码管显示“00-00-00”，不变化；

2、按下设置按键，从左向右利用数字键和 A 键（代表-）更改数字，按下一位显示一位，全部设置完成后判断是否超出时间范围，如有不合理时间会做自动调整存入设置好的时间，按下开始键从设置好的时间开始计时显示；

3、按下开始按键，数码管开始计时，每隔一秒变化一次，要求分、时的进位正常；

4、按下复位按键，数码管显示“00-00-00”，不变化，等待其他按键的按下；

5、每次时变化的时候（整点），蜂鸣器会发出“嘀”一声（频率时间自拟）。

键盘设置分布如图 2- 25 所示。

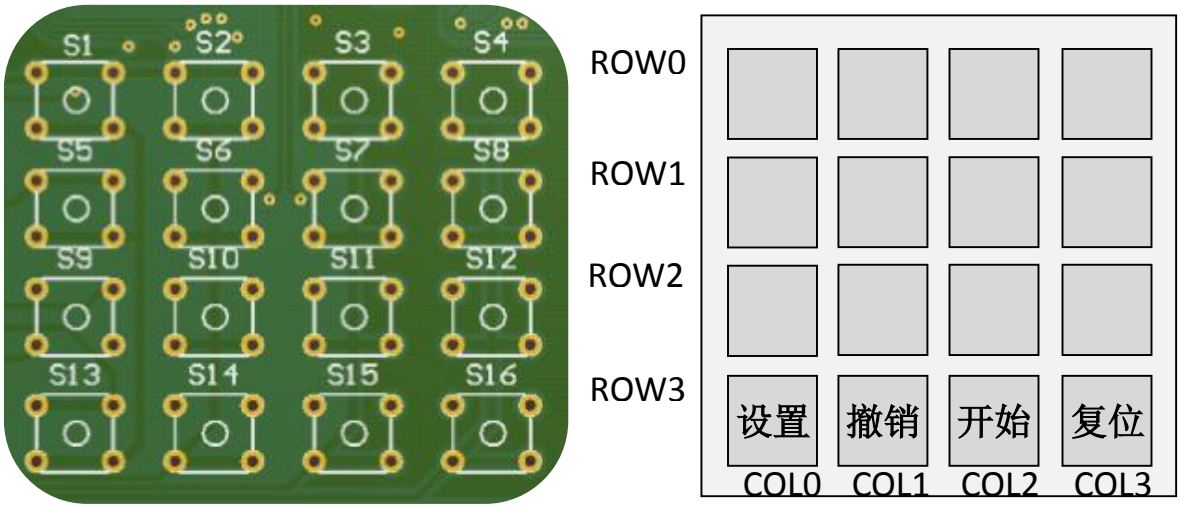

图 2- 25　矩阵式按键示意图

二、相关知识

2.1　蜂鸣器的使用介绍

2. 1. 1　蜂鸣器简介

1、蜂鸣器的作用

蜂鸣器是一种一体化结构的电子讯响器，采用直流电压供电，广泛应用于计

算机、打印机、复印机、报警器、电子玩具、汽车电子设备、电话机、定时器等电子产品中作发声器件。

2、蜂鸣器的分类

蜂鸣器主要分为有源蜂鸣器和无源蜂鸣器两种类型。本项目使用有源蜂鸣器，无源蜂鸣器将在项目三中进行详细介绍。

3、蜂鸣器电路图形符号

蜂鸣器在电路中用字母“H”或“HA”（旧标准用“FM”、“LB”、“JD”等）表示。

2.1.2 蜂鸣器的结构原理

1、压电式蜂鸣器

压电式蜂鸣器主要由多谐振荡器、压电蜂鸣片、阻抗匹配器及共鸣箱、外壳等组成。有的压电式蜂鸣器外壳上还装有发光二极管。

多谐振荡器由晶体管或集成电路构成。当接通电源后（1.5~15V 直流工作电压），多谐振荡器起振,输出 1.5~2.5kHZ 的音频信号，阻抗匹配器推动压电蜂鸣片发声。

压电蜂鸣片由锆钛酸铅或铌镁酸铅压电陶瓷材料制成。在陶瓷片的两面镀上银电极，经极化和老化处理后，再与黄铜片或不锈钢片粘在一起。

2、电磁式蜂鸣器

电磁式蜂鸣器由振荡器、电磁线圈、磁铁、振动膜片及外壳等组成。

接通电源后，振荡器产生的音频信号电流通过电磁线圈，使电磁线圈产生磁场。振动膜片在电磁线圈和磁铁的相互作用下，周期性地振动发声。

有源蜂鸣器直接接上额定电源(新的蜂鸣器在标签上都有注明)就可连续发声；而无源蜂鸣器则和电磁扬声器一样，需要接在音频输出电路中才能发声。

三、操作训练

1、任务分析

该任务从功能上可以分为三部分：设置或复位、计时、报时，均在矩阵式键盘上完成。

矩阵式键盘子程序，此部分需要不停的调用以判断是否有键按下，所以应放

到主程序中循环调用：

矩阵式按键子程序 key.c

```
unsigned char Key(void)
{
    unsigned char kt,temp;
    while(1)
    {
        KIO=0XF0;//高四位拉高
        if (KIO!=0xf0)//有键按下
        {
            if (++kt==2)//延时防抖
            {
                temp=KIO;
                KIO=temp|0x0f;//取键值
                switch (KIO)//判断
                {
                    case 0x77:return 3; break;//3
                    case 0x7b:return 7;    break;//7
                    case 0x7d:return 11; break;//B
                    case 0x7e:return 15; break;//F
                    case 0xb7:return 2; break;//2
                    case 0xbb:return 6; break;//6
                    case 0xbd:return 10; break;//A
                    case 0xbe:return 14; break;//E
                    case 0xd7:return 1; break;//1
                    case 0xdb:return 5; break;//5
                    case 0xdd:return 9; break;//9
                    case 0xde:return 13; break;//D
                    case 0xe7:return 0; break;//0
                    case 0xeb:return 4; break;//4
                    case 0xed:return 8;    break;//8
                    case 0xee:return 12; break;//C
                    default:    break;
                }
            }
        }
        else kt=0;//无键按下，kt 清零
    }
}
```

设置、复位部分程序，此部分在读入键值后进行：

设置、复位功能子程序 con.c

```
hour=LED_str[0]*10+LED_str[1];//记录设置的时数据
min=LED_str[3]*10+LED_str[4]; //记录设置的分数据
second=LED_str[6]*10+LED_str[7]; //记录设置的秒数据
temp=key();
if (temp<10)//按下数字按键
{
    if (Num<8)
    LED_str[Num]=temp;//显示按下的数字
    Num++;
    Num&=0x07;
}
else if (temp==12)//按下设置按键
{
    CLEAR;//清屏
    Num=0;//从最左边数码管显示
    start=0;//实时显示子程序开关标志
}
else if (temp==13)//按下撤销按键
{
    Num-=1;//退回一位数码管
    LED_str[Num]=17;//不显示当前按下的数字
    start=0;
}
else if(temp==15)//按下复位按键
{
    second=0;min=0;hour=0;//变量均清零并显示
    start=0;
}
```

开始、计时、报时部分程序：

剩余判断部分子程序 con.c

```
else if (temp==14)//开始
{
        if(second>=60) second=59;//判断秒是否超范围，取最大值
        if(min>=60)     min=59; //判断秒是否超范围，取最大值
        if(hour>=24)    hour=23;//判断秒是否超范围，取最大值
        start=1;
}
```

计时功能在定时中断服务主程序中实现：

计时功能子程序 time.c

```
ms++;
    if(ms==500)//计时满 1 秒
    {
        RED=1;//LED 熄灭
        ms=0;//ms 清零
        second++;//秒加一
        if(second==60)//判断秒是否超范围
        {

            second=0;//秒清零
            min++;//分加一
            if(min==60)//判断分是否超范围
            {
                min=0;//分清零
                hour++;//时加一
                RED=0;//LED 点亮（因为此时时变量进位，即整点）
                if(hour==24)//判断时是否超范围
                    hour=0;//时清零
            }
        }
    }
```

2、主程序流程图

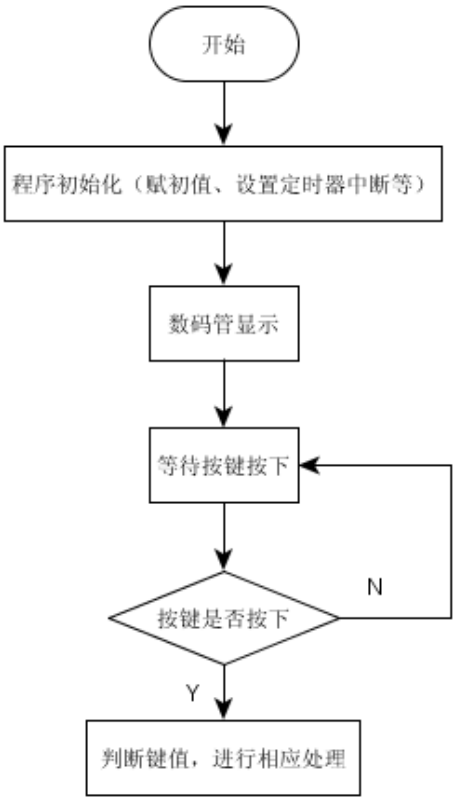

图 2- 26　电子钟主程序流程图

3、参考程序

电子钟程序　　SHUMAGUANclock.C

```
#include "reg51.h"
#define KIO P1
sbit RED=P3^0;
unsigned char xdata LED_DM _at_ 0x7FFF;
unsigned char xdata LED_PX _at_ 0xBFFF;
unsigned char code M7G[]=
{
    0xc0,//0
    0xf9,//1
    0xa4,//2
    0xb0,//3
    0x99,//4
    0x92,//5
    0x82,//6
```

```
    0xf8,//7
    0x80,//8
    0x90,//9
    0x88,//10    A
    0x83,//11 B
    0xc6,//12 C
    0xa1,//13 d
    0x86,//14 E
    0x8e,//15 F
    0xbf,//16 -
    0xff,//17  无
    0xaf,//18 r
    0xa3,//19 o
    0x8c,//20 P
};
unsigned char LED_str[8]={17,17,17,17,17,17,17,17};
unsigned char second,min,hour,Num;
unsigned int ms;
void DSP();
void Intime();
bit start;
unsigned char key(void);
void TIME0() interrupt 1
{
    TH0=0xf8;
    TL0=0xcc;
    ms++;
    if(ms==500)
    {
        RED=1;
        ms=0;
        second++;
        if(second==60)
        {

            second=0;
            min++;
```

```
                if(min==60)
                {
                        min=0;
                        RED=0;
                        hour++;
                        if(hour==24)
                                hour=0;
                }
        }
    }
    DSP();
    if(start)   Intime();
}
void main()
{
    unsigned char temp;
    TMOD=1;
    TH0=0xf8;
    TL0=0xcc;
    ET0=1;
    TR0=1;
    EA=1;
    while(1)
    {
        hour=LED_str[0]*10+LED_str[1];
        min=LED_str[3]*10+LED_str[4];
        second=LED_str[6]*10+LED_str[7];
        temp=key();
        if (temp<10)
        {
            if (Num<8)
            LED_str[Num]=temp;
            Num++;
            Num&=0x07;
        }
        else if (temp==12)//设置
        {
```

```
            CLEAR;
            Num=0;
            start=0;
        }
        else if (temp==13)//撤销
        {
            Num-=1;
            LED_str[Num]=17;
            start=0;
        }
        else if (temp==14)//开始
        {
            if(second>=60) second=59;
            if(min>=60)      min=59;
            if(hour>=24)    hour=23;
            LED_str[7]=second%10;
            LED_str[6]=second/10;
            LED_str[5]=16;
            LED_str[4]=min%10;
            LED_str[3]=min/10;
            LED_str[2]=16;
            LED_str[1]=hour%10;
            LED_str[0]=hour/10;
            start=1;
        }
        else if(temp==15)//复位
        {
            second=0;min=0;hour=0;
            LED_str[7]=0;
            LED_str[6]=0;
            LED_str[5]=16;
            LED_str[4]=0;
            LED_str[3]=0;
            LED_str[2]=16;
            LED_str[1]=0;
            LED_str[0]=0;
            start=0;
```

```
            }
        }
}
unsigned char Key(void)
{
    unsigned char kt,temp;
    while(1)
    {
        KIO=0XF0;
        if (KIO!=0xf0)//与后面 else 对应
        {
            if (++kt==2)
            {
                temp=KIO;
                KIO=temp|0x0f;
                switch (KIO)
                {
                    case 0x77:return 3; break;//3
                    case 0x7b:return 7; break;//7
                    case 0x7d:return 11;break;//B
                    case 0x7e:return 15;break;//F
                    case 0xb7:return 2; break;//2
                    case 0xbb:return 6; break;//6
                    case 0xbd:return 10;break;//A
                    case 0xbe:return 14;break;//E
                    case 0xd7:return 1; break;//1
                    case 0xdb:return 5; break;//5
                    case 0xdd:return 9; break;//9
                    case 0xde:return 13;break;//D
                    case 0xe7:return 0; break;//0
                    case 0xeb:return 4; break;//4
                    case 0xed:return 8;break;//8
                    case 0xee:return 12;break;//C
                    default:    break;
                }
            }
        }
```

```
            else kt=0;
        }
}
void DSP()
{
        static unsigned char dp_h;
        unsigned int j;
        DM=0xff;
        PX=0xff;
        DM=M7G[LED_str[dp_h]];
        PX=(1<<dp_h);
        for(j=0;j<550;j++);
        dp_h++;
        dp_h&=0x07;
}
void Intime()
{
        LED_str[7]=second%10;
        LED_str[6]=second/10;
        LED_str[5]=16;
        LED_str[4]=min%10;
        LED_str[3]=min/10;
        LED_str[2]=16;
        LED_str[1]=hour%10;
        LED_str[0]=hour/10;
}
```

4、程序说明

本任务在任务四的基础上加深了矩阵式键盘应用的难度，在主程序中调用按键子程序，并且对各个键值进行判断和使用，同时结合任务四的定时器应用，完成电子钟的设计。

5、任务实施

5.1 建立工程

打开 keil 软件，通过菜单“Project”，新建立一个工程“new Project”项目

SHUMAGUANclock，然后再建一个文件名为 SHUMAGUANclock.C 的源程序文件，将上面的参考程序输入并保存。建立的 SHUMAGUANclock.C 文件添加入本项目中。

5.2 编译并生成 HEX

单击“Build target”按钮，对源程序进行编译和链接，产生 HEX 文件。

5.3 烧录芯片

打开 STC-ISP 下载软件，选中单片机型号为“STC90C52RC”，选择最低波特率为 2400 最高波特率为 115200（为默认值一般不需修改）。点击打开程序文件按钮，在刚才建立的 SHUMAGUANclock 文件夹中生成的 SHUMAGUANclock.HEX 文件。然后点击“下载/编程”按钮。最后给最小系统通电，等待电脑端串口与单片机握手成功。就能把程序载入到单片机中去了。其详细操作图如图 2-27 所示：

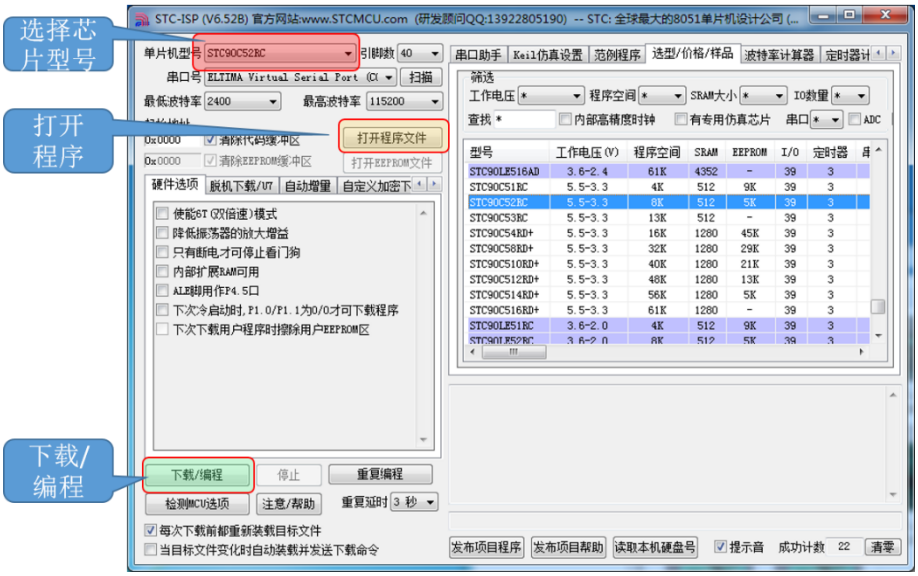

图 2-27 STC 单片机程序烧写示意图

5.4 硬件调试

程序下载完成后，把单片机最小系统接入电路中，调试电路使其能正常工作，并完成本任务。

四、任务评价

表 2- 11　任务五完成情况评价表

项目名称			评价时间		年　月　日		
小组名称		小组成员					
评价内容	评价要求	权重	评价标准	学生自评得分	小组评价得分	教师评价得分	合计
职业与安全意识	1. 工具摆放、操作符合安全操作规程；2. 遵守纪律、爱惜设备和器材、工位整洁；3. 具有团队协作精神。	10%	好（10） 较好（8） 一般（6） 差（<6）				
印制线路板调试	1. 能够根据任务要求进行印制线路板的调试；2. 根据任务要求，能够对对应电路部分进行硬件电路的检查与故障检修。	15%	好（15） 较好（12） 一般（9） 差（<9）				
项目功能测试	1. 编写的程序能成功编译；2. 程序能正确烧写到芯片中；3. 系统启动后数码管能正常显示字符。	60%	好（60） 较好（45） 一般（30） 差（<30）				
问题与思考	1. 如果在运行过程中出现设置的时间超出正常范围是什么原因？如何解决？	15%	好（15） 较好（12） 一般（9） 差（<6）				
教师签名			学生签名			总分	
项目评价=学生自评（0.2）+小组评价（0.3）+教师评价（0.5）							

五、思考与练习

1、如果在运行过程中出现设置的时间超出正常范围是什么原因？如何解决？

2、数码管初始显示时间需要显示别的时间应当如何更改程序？

任务六　密码锁控制器制作

一、任务要求

使用实验设备，设计以下电路。

1、用 4×4 矩阵键盘组成 0~9 数字键及确认键和删除键；

2、用 8 位数码管显示输入密码及当前状态信息；

3、用一个独立继电器，控制密码锁开关。继电器得电为开锁状态，继电器不得电为关闭状态。

编程实现以下功能：

1、上电后，数码管无显示；

2、当按下数字键 0~9 中任意按键时，数码管靠左显示按下数字，再次按下任意数字键，数码管靠左第二位显示当前数字，以此类推，但是当删除键按下时，当前显示的这个数码管熄灭，下次按下数字键还是在当前位置显示（即撤销功能），当数码管 8 个 LED 都有显示数字后，不响应继续输入的数字按键；

3、当按下确定键后，单片机对输入的密码与设定密码进行比较，若密码正确，则数码管显示 PASS 并且控制继电器开锁，若密码错误，则数码管显示 Error 并且继电器保持关闭状态；（默认密码为 12345678）

4、开锁后保持 5 秒，然后自动进入关闭状态。

按键设置分布如图 2- 28 所示。

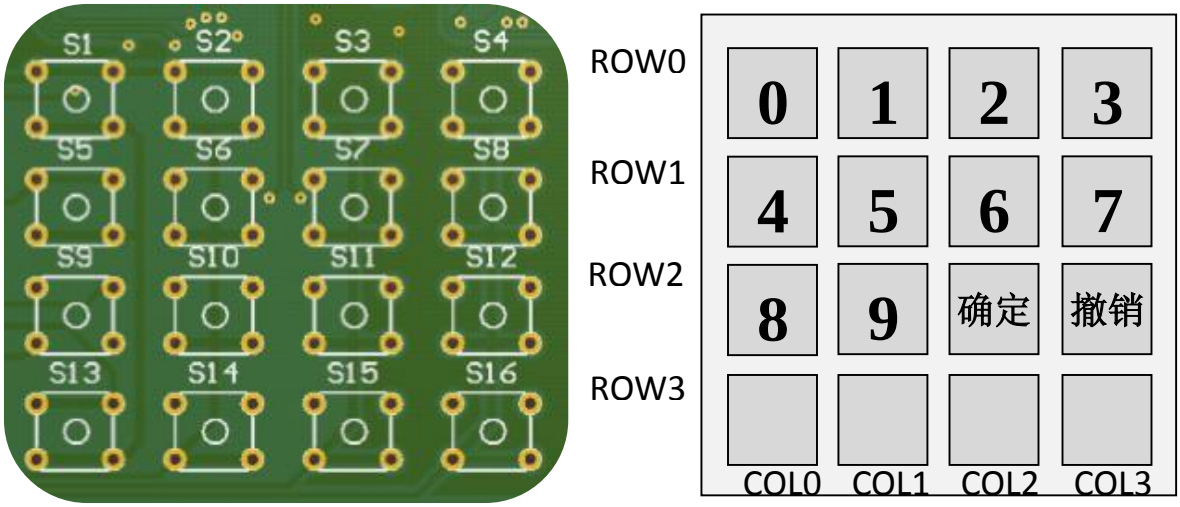

图 2- 28　按键与密码锁控制器键盘对应图

二、相关知识

2.1 继电器的使用介绍

继电器是一种电子控制器件，它具有控制系统（又称输入回路）和被控制系统（又称输出回路），通常应用于自动控制电路中，它实际上是用较小的电流去控制较大电流的一种“自动开关”。故在电路中起着自动调节、安全保护、转换电路等作用。当输入量(如电压、电流)达到规定值时，使被控制的输出电路导通或断开的电器。具有动作快、工作稳定、使用寿命长、体积小等优点。广泛应用于电力保护、自动化、运动、遥控、测量和通信等装置中。

电磁式继电器一般由铁芯、线圈、衔铁、触点簧片等组成的。只要在线圈两端加上一定的电压，线圈中就会流过一定的电流，从而产生电磁效应，衔铁就会在电磁力吸引的作用下克服返回弹簧的拉力吸向铁芯，由此带动衔铁的动触点与静触点（常开触点）吸合。当线圈断电后，电磁的吸力也随之消失，衔铁就会在弹簧的反作用力返回原来的位置，使得动触点与原来的静触点（常闭触点）吸合。这样吸合、释放，从而达到了在电路中的导通、切断的目的。对于继电器的“常开”、“常闭”触点，可以这样来区分：继电器线圈未通电时处于断开状态的静触点，称为“常开触点”；处于接通状态的静触点称为“常闭触点”。

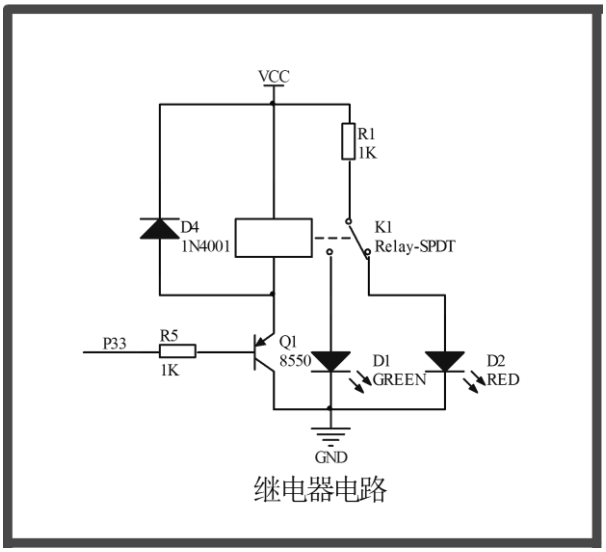

图 2- 29　继电器接线图

继电器模块内部结构如图所示：当 P33 接口输出低电平，使 PNP 三级管 Q1 饱和导通，使继电器 K1 线圈得电，绿色 LED 灯 D1 点亮，红色 LED 灯 D2 熄灭。

当 P33 接口输出高电平，则三极管 Q1 截止，继电器 K1 线圈不得电，红色 LED 灯 D2 点亮，绿色 LED 灯 D1 熄灭。由此得出：只要单片机的 IO 口向继电器控制三极管输出低电平，就能使继电器工作，反之输出高电平继电器不工作。

三、操作训练

1、任务分析

本任务用到的模块其实都是前面已经学习过的，比如：数码管显示，4X4 行列键盘，继电器，由于这个任务比较复杂，将其分化成各个简单的子任务来完成。密码锁任务，可以分化成以下几个子任务：

1.1 按键显示电路程序设计

显示电路采用 8 位数码管显示，需要做到以下功能：按下行列按键中 0~9 键，8LED 数码管靠左显示当前按下的键值。当数码管 8 个 LED 都有数字显示时，不响应当前数字按键。如何让输入的按键一个一个靠左显示在数码管上，其实很简单，我们可以设置一个变量，每次按下按键先把键值放入变量所对应的显示缓存中，把此变量加一。这样就完成了现实电路的设计。

在软件编程中，由于整个程序运行过程中，数码管显示总有一部分不变（如 PASS、Error 等），可以考虑把这部分显示的程序做一个宏定义。如：

```
#define PASS str[0]=20,str[1]=10,str[2]=str[3]=5,str[4]=str[5]=str[6]=str[7]=17
#define Error str[0]=14,str[1]=str[2]=str[4]=18,str[3]=19,str[5]=str[6]=str[7]=17
```

又如清屏，可以用宏定义为：

```
#define CLEAR str[0]=str[1]=str[2]=str[3]=str[4]=str[5]=str[6]=str[7]=17
```

数字按键识别及撤销功能（B 键）的实现：

按键识别程序 key.c

```
temp=key();
if (temp<16)//按下的为数字按键
{
    if (Num<8)//若当前位数不足 8 位
    LED_str[Num]=temp;//显示
    Con=Con*10+temp;//保存输入口令
```

```
        Num++;//位数加一
        Num&=0x07;//使得位数在 0-7 变化
}
else if (temp==50)//设置 B 键值为 50，表示撤销按键被按下
{
        Num-=1;//退回一位
        LED_str[Num]=17;//显示空，等待重新输入
        Con/=10;
//已保存的输入口令右移一位（相当于不保存当前错误输入）
}
```

1.2 密码检验电路程序设计

密码检验电路需要做到以下功能：当按下确定键后，程序检查输入的键值和固有密码是否相同。如果相同，则开锁，如果不同则清空显示，重新等待新的密码输入。

密码检测子程序 code.c

```
if (Con==Shu)
{
        Con=0;
        PASS;
        PM=0；
        for (Ms=0;Ms<550;Ms++);
}
else
{
        Num=0;Con=0;
        Error;
        PM=1；
        for (Ms=0;Ms<330;Ms++);
}
CLEAR;
PM=1；
```

1.3 继电器模块设计

在软件中，继电器的演示和 LED 类似，低电平表示得电动作，高电平不得电无动作。

2、主程序流程图

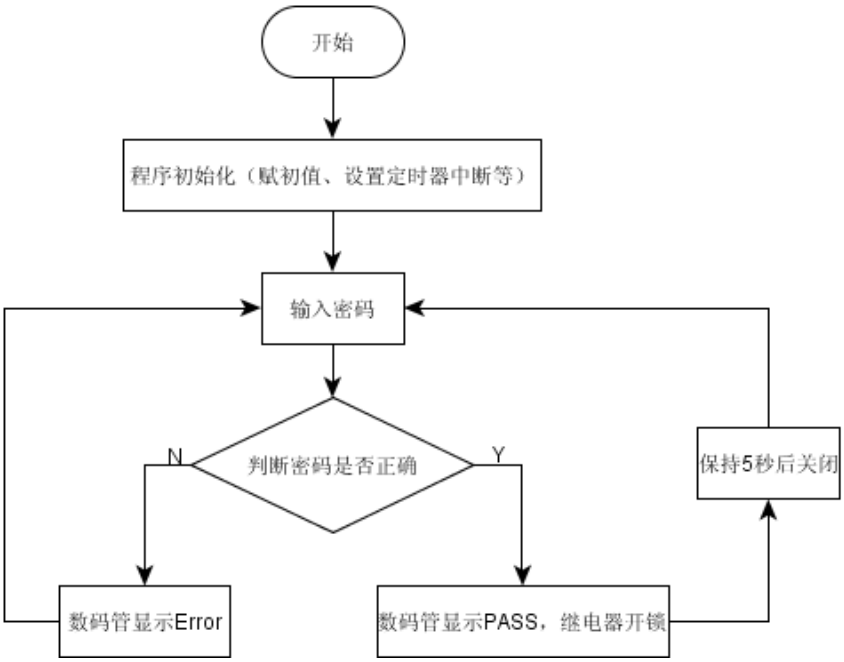

图 2- 30　密码锁主程序流程图

3、参考程序

密码锁程序　　SHUMAGUANlock.C

```
#include "reg51.h"
#define KIO P1
sbit RED=P3^0;
sbit GREEN=P3^1;
unsigned char xdata LED_DM _at_ 0x7FFF;
unsigned char xdata LED_PX _at_ 0xBFFF;
unsigned char code M7G[]=
{
    0xc0,//0
    0xf9,//1
    0xa4,//2
    0xb0,//3
    0x99,//4
    0x92,//5
    0x82,//6
    0xf8,//7
    0x80,//8
```

```
	0x90,//9
	0x88,//10 A
	0x83,//11 B
	0xc6,//12 C
	0xa1,//13 d
	0x86,//14 E
	0x8e,//15 F
	0xbf,//16 -
	0xff,//17 无
	0xaf,//18 r
	0xa3,//19 o
	0x8c,//20 P
};
unsigned char str[8]={17,17,17,17,17,17,17,17};//片选，选择某个数码管
#define PASS str[0]=20,str[1]=10,str[2]=str[3]=5,str[4]=str[5]=str[6] =str[7]=17
#define Error str[0]=14,str[1]=str[2]=str[4]=18,str[3]=19,str[5]= str[6] =str[7]=17
#define CLEAR str[0]=str[1]=str[2]=str[3]=str[4]=str[5]=str[6]= str[7]=17
unsigned char Num,Second;
unsigned int Ms;
unsigned long Con=0,Shu=12345678;
void DSP();
unsigned char key(void);
void TIME0() interrupt 1
{
	TH0=0xf8;
	TL0=0xcc;
	DSP();
}
void main()
{
	unsigned char temp;
	TMOD=1;
	TH0=0xf8;
	TL0=0xcc;
	ET0=1;
	TR0=1;
	EA=1;
```

```
    while(1)
    {
        temp=key();
        if (temp<16)
        {
            if (Num<8)
            LED_str[Num]=temp;
            Con=Con*10+temp;
            Num++;
            Num&=0x07;
        }
        else if (temp==50)
        {
            Num-=1;
            LED_str[Num]=17;
            Con/=10;
        }
    }
}
unsigned char Key(void)
{
    unsigned char kt,temp;
    while(1)
    {
        KIO=0XF0;
        if (KIO!=0xf0)//与后面 else 对应
        {
            if (++kt==2)
            {
                temp=KIO;
                KIO=temp|0x0f;
                switch (KIO)
                {
                    case 0x77:return 3; break;//3
                    case 0x7b:return 7; break;//7
                    case 0x7d:return 50;    break;//B
                    case 0x7e:return 15;    break;//F
```

```
                case 0xb7:return 2; break;//2
                case 0xbb:return 6; break;//6
                case 0xbd:
                                if (Con==Shu)
                                {
                                Con=0;
                                PASS;
                                GREEN=0;
                                for (Ms=0;Ms<330;Ms++);
                                }
                                else
                                {
                                Num=0;Con=0;
                                ERROR;
                                RED=0;
                                for (Ms=0;Ms<330;Ms++);
                                }
                                for (Ms=0;Ms<110;Ms++);
                                Ms=0;
                                CLEAR;
                                RED=1;
                                GREEN=1;
                                break;//A
                case 0xbe:return 14;      break;//E
                case 0xd7:return 1; break;//1
                case 0xdb:return 5; break;//5
                case 0xdd:return 9; break;//9
                case 0xde:return 13;      break;//D
                case 0xe7:return 0; break;//0
                case 0xeb:return 4; break;//4
                case 0xed:return 8;break;//8
                case 0xee:return 12;break;//C
                default:   break;
            }
        }
    }
```

```
            else kt=0;
        }
}
void DSP()
{
        static unsigned char dp_h;
        unsigned int j;
        DM=0xff;
        PX=0xff;
        DM=M7G[LED_str[dp_h]];
        PX=(1<<dp_h);
        dp_h++;
        dp_h&=0x07;
}
```

4、程序说明

本任务在任务五的基础上，继续加深矩阵式键盘的应用，并且加入了审核密码的功能和蜂鸣器功能，这两部分内容都放在主函数中实现。

5、任务实施

5.1 建立工程

打开 keil 软件，通过菜单“Project”，新建立一个工程“new Project”项目 SHUMAGUANlock，然后再建一个文件名为 SHUMAGUANlock.C 的源程序文件，将上面的参考程序输入并保存。建立的 SHUMAGUANlock.C 文件添加入本项目中。

5.2 编译并生成 HEX

单击“Build target”按钮，对源程序进行编译和链接，产生 HEX 文件。

5.3 烧录芯片

打开 STC-ISP 下载软件，选中单片机型号为“STC90C52RC”，选择最低波特率为 2400 最高波特率为 115200（为默认值一般不需修改）。点击打开程序文件按钮，在刚才建立的 SHUMAGUANlock 文件夹中生成的 SHUMAGUANlock.HEX 文件。然后点击“下载/编程”按钮。最后给最小系统通电，等待电脑端串口与单片机握手成功。就能把程序载入到单片机中去了。其详细操作图如图 2- 31 所示：

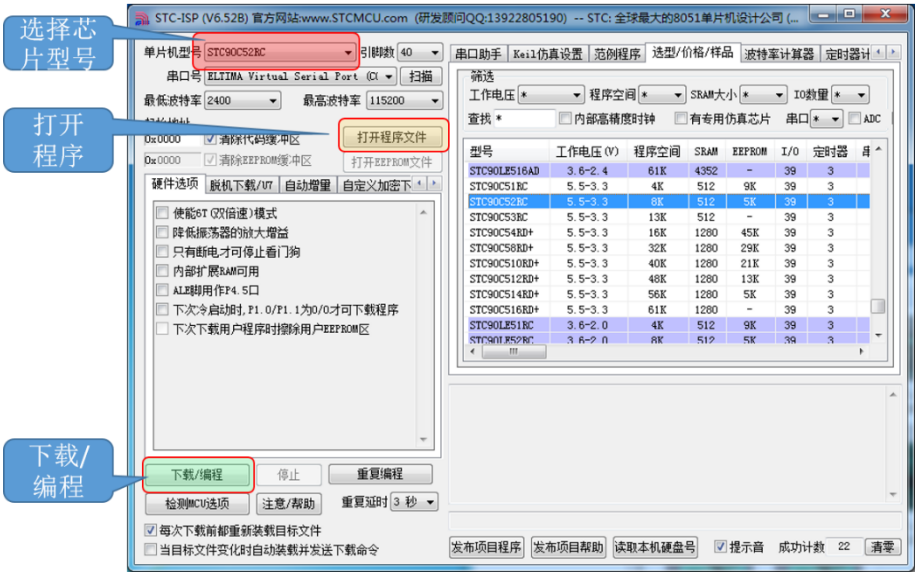

图 2- 31　STC 单片机程序烧写示意图

5.4　硬件调试

程序下载完成后，把单片机最小系统接入电路中，调试电路使其能正常工作，并完成本任务。

四、任务评价

表 2- 12　任务六完成情况评价表

项目名称			评价时间		年　月　日		
小组名称		小组成员					
评价内容	评价要求	权重	评价标准	学生自评得分	小组评价得分	教师评价得分	合计
职业与安全意识	1. 工具摆放、操作符合安全操作规程；2. 遵守纪律、爱惜设备和器材、工位整洁；3. 具有团队协作精神。	10%	好（10） 较好（8） 一般（6） 差（<6）				
印制线路板调试	1. 能够根据任务要求进行印制线路板的调试；2. 根据任务要求，能够对对应电路部分进行硬件电路的检查与故障检修。	15%	好（15） 较好（12） 一般（9） 差（<9）				
项目功能测试	1. 编写的程序能成功编译；2. 程序能正确烧写到芯片中；3. 系统启动后数码管能正常显示字符。	60%	好（60） 较好（45） 一般（30） 差（<30）				
问题与思考	1. 如果在运行过程中出现按下按键没有响应是什么原因？如何解决；2. 如何更改初始密码。	15%	好（15） 较好（12） 一般（9） 差（<6）				
教师签名			学生签名			总分	
项目评价=学生自评（0.2）+小组评价（0.3）+教师评价（0.5）							

五、思考与练习

1、如果在运行过程中出现按下按键没有响应是什么原因？如何解决？

2、在密码锁任务中应当如何更改初始密码？

★任务七　计算器制作

一、任务目标

计算器为生活中常用设备如图 2- 32，它把人们从繁杂的笔算劳动中解脱出来。本次任务的目的是利用矩阵式按键和数码管完成一个计算器。

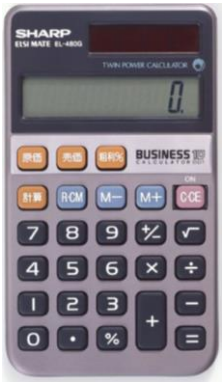

图 2- 32　日常生活中使用的实物计算机

1、系统上电后，数码管靠右显示 0；

2、按下数字键，数码管上依次显示；按下 S4、S8、S12、S16 键，对应表示进行“加、减、乘、除”操作，再按下 S15 做等号计算，数码管上显示计算结果；

3、在输入数字过程中按下 S13(ON/C 键)，表示复位，数码管靠右显示 0。

4、计算器具体运行流程请参考实物计算器计算效果。

本任务中按键与计算器功能键对应关系如图 2- 33 所示：

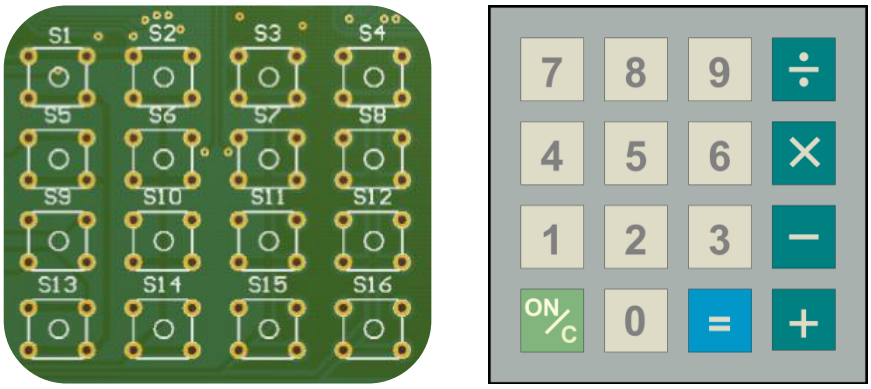

图 2- 33　按键与计算器键盘对应图

二、相关知识

该任务所用到的知识点前面已经都介绍过，本任务不再赘述。

三、操作训练

1、任务分析

该任务其他的功能在前面章节已经介绍，本任务中主要需要解决乘除法的实现。

1.1 加、减、乘的实现

当两个操作数输入完成后，软件实现上需将按键键入的数在逻辑上转换为自然数的形式，即两个四位的数字 A1，A2，A3，A4，B1，B2，B3，B4 应该转换自然数 C1=1000*A4+100*A3+10*A2+A1，C2=1000*B4+100*B3+10*B2+B1，CC 为 C1、C2 进行运算所得的结果，输出为 CC4=CC/10，CC3=CC%1000/10，CC2=CC%100/10，CC1=CC%10，然后将这四个数值通过查询段码表显示在数码管上。

但在这其中可能会出现高位有 0 的情况，需要进行消零处理，在数值显示之前需要进行零值判断，如果是 0 的话则数码管不显示。

1.2 除法的实现

除法与加减乘运算有区别在于除法有可能使得商中有小数点，以 5 除以 4 为例，整数部分是 CC=5/4，为 1，再对其取余数为 1，让 1*10 后再对 4 取除法的商为 2 即小数点后第一位，然后 1*10 对 4 取除法的余数为 2，让 2*10 再对 4 取除法的商为 5，即小数点后第二位为 5，完成小数部分输出。

2、主程序流程图

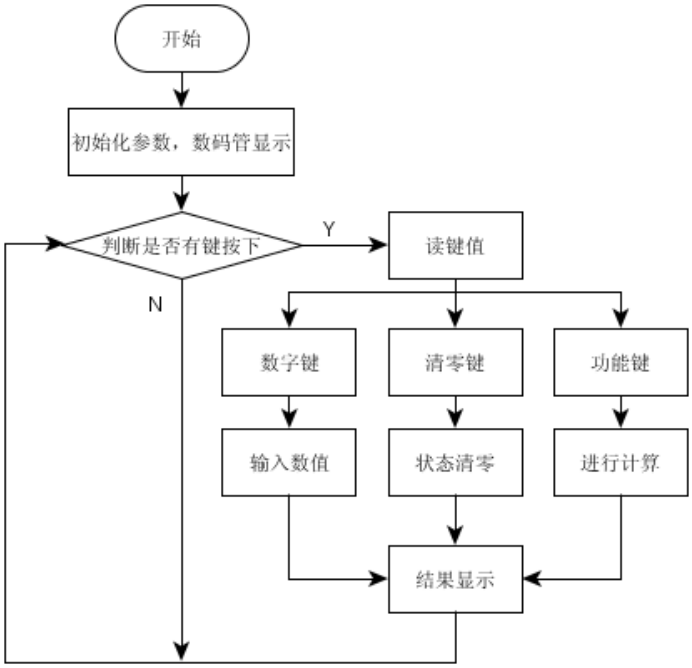

图 2- 34　计算器主程序流程图

3、参考程序

计算器程序　　SHUMAGUANcul.C

```
#include<reg51.h>
int jieguo();//求解
void delayms(unsigned char delaytime);//延时
void display(int a,unsigned char b);//显示第 b 位显示 a
void scankey();//扫描矩阵键盘
void scanLED();//刷新数码管
unsigned char testkey();//键盘触发
long *p;//输入数据指向指针(指向 dataA 或 dataB)
unsigned char key,key_ys;
//key 键值缓存,key_ys 按键类型(数字、运算符、等于三类)
long dataA,dataB,dataSum;//操作数 dataA,dataB；结果 dataSum
unsigned char jisuanfu;//运算符
void main()
{
    dataSum=-1;  //结果初始化
    dataA=0;//dataA 初始化
```

```
    dataB=0;//dataB 初始化
    p=&dataA;//输入数据指向 dataA
    jisuanfu=0;//运算符初始化
    key_ys=0;//按键类型初始化
    while(1)
    {
        if(testkey())
        {
            //键盘触发
            delayms(5);    //防抖
            if(testkey())
            {
                //键盘触发确认
                scankey();      //键值扫描
            }
        }
        scanLED();//刷新数码管
    }
}
unsigned char testkey()
{
    //键盘触发
    unsigned char a;//申明变量 a
    P1=0xf0;//行线置 0
    a=P1;      //缓存列线数据
    if(a==0xf0) return 0;//无按键返回 0
    else return 1;//有按键返回 1
}
void scankey()
{
    //键值扫描
    P1=0xf0; //行线置 0
    key=P1;  //缓存列线数据
    P1=key|0x0f;  //某列线置 0
    key=P1;  //读出键值
    while(testkey());//等待按键释放
    switch(key)
```

```
    {
        //按键动作
        case 0xe7:
            dataSum=0;
            dataA=0;
            dataB=0;
            p=&dataA;
            key_ys=1;
            jisuanfu=0;
            break;//CLR
        case 0xeb:
            if(key_ys)dataB=0;
            *p=(*p)*10+1;
            dataSum=*p;
            key_ys=0;
            break;//1
        case 0xed:
            if(key_ys)dataB=0;
            *p=(*p)*10+4;
            dataSum=*p;
            key_ys=0;
            break;//4
        case 0xee:
            if(key_ys)dataB=0;
            *p=(*p)*10+7;
            dataSum=*p;
            key_ys=0;
            break;//7
        case 0xd7:
            if(key_ys)dataB=0;
            *p=(*p)*10+0;
            dataSum=*p;
            key_ys=0;
            break;//0
        case 0xdb:
            if(key_ys)dataB=0;
            *p=(*p)*10+2;
```

```
            dataSum=*p;
            key_ys=0;
            break;//2
        case 0xdd:
            if(key_ys)dataB=0;
            *p=(*p)*10+5;
            dataSum=*p;
            key_ys=0;
            break;//5
        case 0xde:
            if(key_ys)dataB=0;
            *p=(*p)*10+8;
            dataSum=*p;
            key_ys=0;
            break;//8
        case 0xb7:
            if(key_ys==2)
            {
                dataB=0;
                jisuanfu=0;
            }
            dataB=dataA=jieguo();
            p=&dataB;
            key_ys=2;
            break;//=
        case 0xbb:
            if(key_ys)dataB=0;
            *p=(*p)*10+3;
            dataSum=*p;
            key_ys=0;
            break;//3
        case 0xbd:
            if(key_ys)dataB=0;
            *p=(*p)*10+6;
            dataSum=*p;
            key_ys=0;
        break;//6
```

```
            case 0xbe:
                if(key_ys)dataB=0;
                *p=(*p)*10+9;
                dataSum=*p;
                key_ys=0;
                break;//9
            case 0x77:
                if (!key_ys)
                dataB=dataA=jieguo();
                p=&dataB;
                jisuanfu=0;
                key_ys=1;
                break;//+加法
            case 0x7b:
                if (!key_ys)
                dataB=dataA=jieguo();
                p=&dataB;
                jisuanfu=1;
                key_ys=1;
                break;//-减法
            case 0x7d:
                if (!key_ys)
                dataB=dataA=jieguo();
                p=&dataB;
                jisuanfu=2;
                key_ys=1;
                break;//*乘法
            case 0x7e:
                if (!key_ys)
                dataB=dataA=jieguo();
                p=&dataB;
                jisuanfu=3;
                key_ys=1;
                break;//除法
        }
}
int jieguo()
```

```
{
    //求解
    switch(jisuanfu)
    {
        case 0:dataSum=dataA+dataB;
        dataB=0;
        return dataSum;
        break;//+

        case 1:dataSum=dataA-dataB;
        dataB=0;
        return dataSum;
        break;//-
        case 2:dataSum=dataA*dataB;
        dataB=0;
        return dataSum;
        break;//*
        case 3:
        if(dataB)dataSum=dataA/dataB;
        else dataSum=2650;
        dataB=0;
        return dataSum;
        break;//\
        case 0xff:dataSum=-1;
        return dataSum;
        break;
    }
}
void scanLED()
{
    //刷新数码管
    unsigned char DD7,DD6,DD5,DD4,DD3,DD2,DD1,DD0;//8 个参数
    if(dataSum>=0&&dataSum<=99999999)
    {
        //显示数据值范围限定
        DD7=dataSum/10000000;    //最高位
        if(DD7)display(DD7,7); //非 0 显示
```

```
        else    display(10,7);      //为 0 不显示
        delayms(2);    //延时
        DD6=dataSum%10000000/1000000;//次高位
        if(DD7+DD6)display(DD6,6);//非 0 或高位非 0 显示本位
        else    display(10,6);      //高位和本位皆 0 不显示
        delayms(2);    //延时
        DD5=dataSum%1000000/100000;//十万位
        if(DD7+DD6+DD5)display(DD5,5);//非 0 或高位非 0 显示本位
        else    display(10,5);//高位和本位皆 0 不显示
        delayms(2);    //延时
        DD4=dataSum%100000/10000;    //万位
        if(DD7+DD6+DD5+DD4)display(DD4,4);//非 0 或高位非 0 显示本位
        else    display(10,4);//高位和本位皆 0 不显示
        delayms(2);    //延时
        DD3=(dataSum%10000)/1000;     //千位
        if(DD7+DD6+DD5+DD4+DD3)display(DD3,3);
        //非 0 或高位非 0 显示本位
        else display(10,3);  //高位和本位皆 0 不显示
        DD2=(dataSum%1000)/100; //百位
        if(DD7+DD6+DD5+DD4+DD3+DD2)display(DD2,2);
        //非 0 或高位非 0 显示本位
        else display(10,2);  //高位和本位皆 0 不显示
        delayms(2);    //延时
        DD1=dataSum/10%10;  //十位
        if (DD7+DD6+DD5+DD3+DD2+DD1)display(DD1,1);
        //非 0 或高位非 0 显示本位
        else display(10,1);  //高位和本位皆 0 不显示
        delayms(2);    //延时
        DD0=dataSum%10;      //末位
        display(DD0,0);     //显示末位
        delayms(2);    //延时
    }
    else
    {
        //数值超限
        for(DD0=0;DD0<8;++DD0)
        {
```

```
                //循环 8 次
                display(10,DD0);//不 DD0 位显示
                delayms(2);          //延时
            }
        }
}
unsigned char code M7G[]=//7 段显示码
{
    0xc0,//0
    0xf9,//1
    0xa4,//2
    0xb0,//3
    0x99,//4
    0x92,//5
    0x82,//6
    0xf8,//7
    0x80,//8
    0x90,//9
    0xff,//无
};
/******************数码管*************************************/
//WR 接 P3^6
unsigned char xdata DM _at_ 0x7fff;          //断码（P2^7）
unsigned char xdata PX _at_ 0xbfff;          //片选（P2^6）
void display(int a,unsigned char b)
{
    //动态显示函数：b 位显示 a
    switch(a)
    {
        case 0:PX=0xff;DM=M7G[0];PX=~(1<<b);break;
        case 1:PX=0xff;DM=M7G[1];PX=~(1<<b);break;
        case 2:PX=0xff;DM=M7G[2];PX=~(1<<b);break;
        case 3:PX=0xff;DM=M7G[3];PX=~(1<<b);break;
        case 4:PX=0xff;DM=M7G[4];PX=~(1<<b);break;
        case 5:PX=0xff;DM=M7G[5];PX=~(1<<b);break;
        case 6:PX=0xff;DM=M7G[6];PX=~(1<<b);break;
        case 7:PX=0xff;DM=M7G[7];PX=~(1<<b);break;
```

```
            case 8:PX=0xff;DM=M7G[8];PX=~(1<<b);break;
            case 9:PX=0xff;DM=M7G[9];PX=~(1<<b);break;
            case 10:PX=0xff;DM=M7G[10];PX=~(1<<b);break;
        }
}
void delayms(unsigned char i)
{
        //延时
        unsigned char ss;
        for(i;i>0;--i)
            for(ss=112;ss>0;--ss);
}
```

4、程序说明

本任务在之前六个任务的基础上，继续加深矩阵式键盘的应用，并且注重程序的严密性，考虑到各种计算结果的可能性测试。

5、任务实施

5.1 建立工程

打开 keil 软件，通过菜单“Project”，新建立一个工程“new Project”项目 SHUMAGUANcul，然后再建一个文件名为 SHUMAGUANcul.C 的源程序文件，将上面的参考程序输入并保存。建立的 SHUMAGUANcul.C 文件添加入本项目中。

5.2 编译并生成 HEX

单击  “Build target”按钮，对源程序进行编译和链接，产生 HEX 文件。

5.3 烧录芯片

打开 STC-ISP 下载软件，选中单片机型号为“STC90C52RC”，选择最低波特率为 2400 最高波特率为 115200（为默认值一般不需修改）。点击打开程序文件按钮，在刚才建立的 SHUMAGUANcul 文件夹中生成的 SHUMAGUANcul.HEX 文件。然后点击“下载/编程”按钮。最后给最小系统通电，等待电脑端串口与单片机握手成功。就能把程序载入到单片机中去了。其详细操作图如图 2- 35 所示：

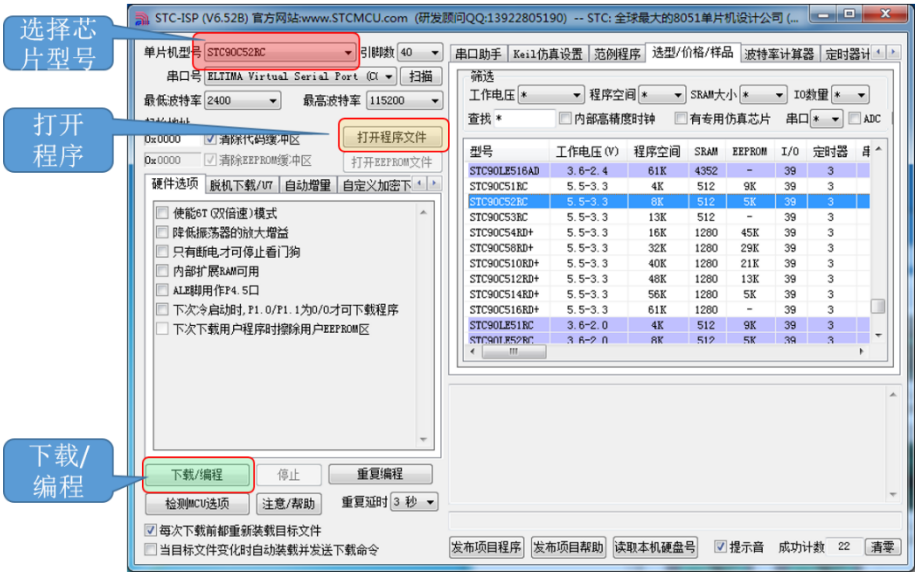

图 2- 35　STC 单片机程序烧写示意图

5.4　硬件调试

程序下载完成后，把单片机最小系统接入电路中，调试电路使其能正常工作，并完成本任务。

四、任务评价

表 2- 13　任务五完成情况评价表

项目名称			评价时间		年　月　日		
小组名称		小组成员					
评价内容	评价要求	权重	评价标准	学生自评得分	小组评价得分	教师评价得分	合计
职业与安全意识	1. 工具摆放、操作符合安全操作规程；2. 遵守纪律、爱惜设备和器材、工位整洁；3. 具有团队协作精神。	10%	好（10） 较好（8） 一般（6） 差（<6）				
印制线路板调试	1. 能够根据任务要求进行印制线路板的调试；2. 根据任务要求，能够对对应电路部分进行硬件电路的检查与故障检修。	15%	好（15） 较好（12） 一般（9） 差（<9）				
项目功能测试	1. 编写的程序能成功编译；2. 程序能正确烧写到芯片中；3. 系统启动后数码管能正常显示字符。	60%	好（60） 较好（45） 一般（30） 差（<30）				
问题与思考	1. 运行过程中除法运算出现误差是什么原因造成的。	15%	好（15） 较好（12） 一般（9） 差（<6）				
教师签名			学生签名			总分	
项目评价=学生自评（0.2）+小组评价（0.3）+教师评价（0.5）							

五、思考与练习

1．运行过程中除法运算出现误差是什么原因造成的？