项目二　交通灯时间显示电路的设计与制作

☆ 项目描述

LED 数码管是单片机控制系统中最常见的显示器件之一，一般用来显示处理结果或输出信号的数字状态。数码管显示是单片机控制电路中比较常见的人机交互环节，因此关于数码管的显示驱动也是单片机应用系统中的重要研究内容。将数码管每个 LED 指示段对应单片机的一个 I/O 引脚，通过改变单片机的 I/O 引脚输出的高低电平来控制对应 LED 段码的亮和灭，由于 I/O 口之间互相独立，因此可以用 I/O 口直接控制 LED 数码管的显示内容。动态显示是将多位数码管的每个相同段码引脚连接在一起，然后接单片机的任意一个 8 位端口输出数据，而将各位数码管的公共端单独分别送至单片机的 I/O 口进行位选通，用单片机的引脚作为位选输出经三极管放大后驱动数码管，从而实现数码管的动态显示。

本项目所设计的交通灯时间显示电路是以单片机最小应用系统作为控制器的基础电路，用 LED 发光二极管作为红、黄、绿三种颜色指示信号灯，并由几个数码管分别显示红灯、黄灯与绿灯的倒计时间，达到模拟十字路口的红、黄、绿交通灯自动控制的状态转换，实现十字路口城乡交通管理智能化。


图 2.1　数码管显示交通灯示意图

☆ 项目分析

1. 工作任务

设计一个由 51 单片机控制的交通灯数码管时间显示电路，并制作硬件电路，进行软件编程，实现数码管倒计时显示和发光二极管构成的指示灯（红、黄、绿）按照规定时间要求进行配合显示。

2. 项目任务要求

（1）设计一个十字路口的交通灯数码管时间显示控制电路，由单片机最小系统电路扩展得到每个路口各 3 组红、黄、绿发光二极管和各 2 组 2 位七段数码管构成的倒计时显示器，模拟指挥十字路口南北方向和东西方向的直行、左转和右转车辆交替通行。

（2）每次绿灯变换为红灯前，要求先亮黄灯 3 s 才能变换，以免造成交通安全隐患。

（3）各方向的灯光变换和数码管时间显示按照项目一中表 1.1 的 10 种状态依次循环。

（4）合理选择电路元器件，了解所选用的电路元器件的主要性能特点及管脚排列。

（5）设计电路原理图，画出 PCB 版图，进行软件编程。

（6）按照给出的电路装配图进行电路安装。

（7）进行电路软硬件联合调试与检测，并分析测试现象。

（8）编写相关的技术文档及工艺文档。包括：产品的功能说明；方案选择报告；产品电路原理图及分析；工具、测试仪器仪表、元器件及材料清单；电路板上的电路布局图（参见附录一的实验开发板）；电路装配的工艺流程说明；调整测试记录；测试结果分析；现场介绍所需的幻灯演示文稿。

（9）将实验板上装配的数码管模拟城乡街道交通灯时间显示控制电路上电模拟演示，并现场介绍功能。

3. 本项目对能力的要求

（1）能进行工作场所安全生产排查和采取相应措施。

（2）会描述七段数码管的构成和特点。

（3）会描述单片机驱动数码管电路的构成与驱动芯片的特点。

（4）能采用 51 单片机对静态扫描与驱动电路进行软件编程。

（5）能采用 51 单片机对多位数码管的动态扫描与驱动电路进行软件编程。

（6）能熟练使用 Keil C51 和 Proteus 仿真软件进行电路设计与软硬件调试。

（7）能进行电路的安装、调试和测试，并进行正确的分析。

☆ 项目分解与实施

根据以上对项目的分析，依据循序渐进的原则，从实现对单片机控制数码管单个静态显示开始，到 2 位动态显示再到 4 位数码管的动态显示，分别能用 Keil 软件进行调试、用 Proteus 软件进行仿真以及实际实验板的调试，最后实现交通灯数码管时间显示电路的设计与制作。

因此，按照先简单后复杂的顺序对本项目进行分解，包括以下三个学习任务：

（1）单个七段数码管显示电路的设计与制作。

（2）2 位七段数码管显示电路的设计与制作。

（3）4 位七段数码管显示电路的设计与制作。

任务一　单个七段数码管显示电路的设计与制作

【任务要求】

根据 51 单片机的最小系统电路构成，结合七段数码管驱动电路的特点，设计与制作出 51 单片机对单个七段数码管的驱动电路，运用单片机的相关理论知识，对单片机控制单个七段数码管进行 9 s 倒计时的显示程序编写，使用 Keil C51 和 Proteus 仿真软件进行电路软件调试，并进行实际电路的调试与检测。

具体的任务要求如下：

（1）用 51 单片机控制 1 位七段数码管进行 9 s 倒计时显示。

（2）选出适合本学习任务的七段数码管驱动元件以及其他元器件。

（3）根据设计要求，设计出单片机对单个七段数码管的驱动电路。

（4）编写单片机对单个七段数码管的显示程序。

（5）焊接、制作单片机驱动单个七段数码管的电路板。

（6）使用 Keil C51 和 Proteus 仿真软件进行软硬件联合调试。

（7）协作解决设计与制作中遇到的问题。

【相关知识】

一、数码管的显示原理及驱动电路

（一）数码管的显示原理

先来看几种常见 LED 数码管的图片。图 2.2 分别为 1 位数码管、2 位数码管和 4 位数码管的实物图，图中这几种数码管右下方都有一个小数点。除此之外也有右下角不带点的数码管，还有“米”字数码管等。

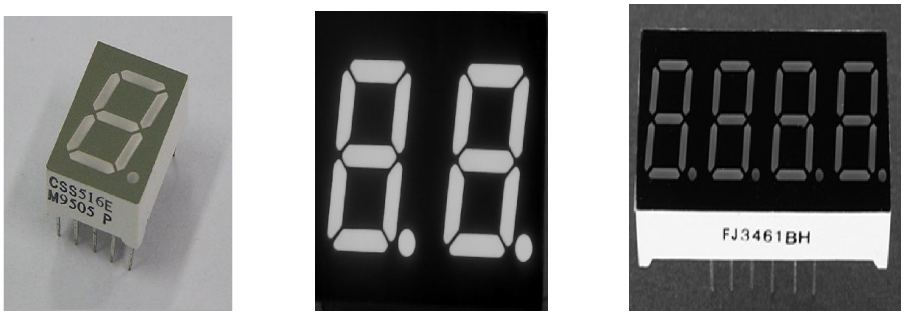


图 2.2　单位数码管、双位数码管、四位数码管实物图

不论哪种数码管，它们的显示原理是一样的，都是靠点亮内部的 LED 发光二极管来发光，因此常称为 LED 数码管。下面介绍 LED 数码管是如何亮起来并显示不同字符的。

LED 数码管实际上是由内部的七个发光二极管组成 8 字形构成的，如果加上小数点就是 8 个。每个发光二极管成为数码管显示的一个段，一般仍习惯把它们统称作七段数码管。当数码管特定的段加上合适的正向电压后，这些特定的段就会发光，就形成人们眼睛看到的数码管所显示的字符了。这些段分别由字母 a、b、c、d、e、f、g、dp (或 A、B、C、D、E、F、G、DP) 来表示，LED 数码管各个段的名称定义和封装尺寸如图 2.3 所示。

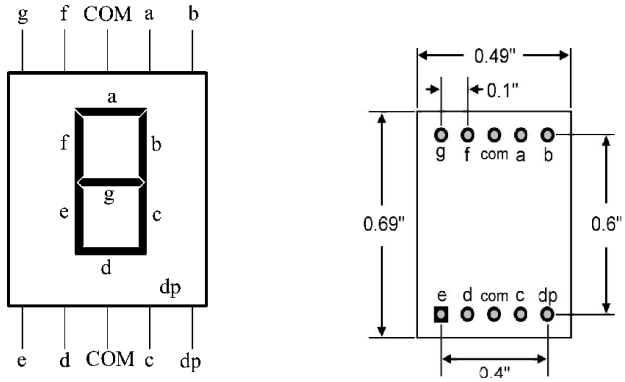


图 2.3　LED 数码管各个段的定义和封装尺寸

比如需要数码管显示一个字符“2”, 那么应当是 a、b、d、e、g 亮，而 c、f、dp 不亮。LED 数码管有普通亮度和高亮度之分，也有 0.5 寸、1 寸等不同的尺寸之分。小尺寸数码管的显示段常由 1 个发光二极管组成，而大尺寸的数码管显示段常由 2 个或多个发光二极管组成。一般情况下，单个发光二极管的管压降为 1.8 V 左右，电流不超过 30 mA。发光二极管的阳极一起连接到电源正极的数码管称为共阳数码管，发光二极管的阴极一起连接到电源负极的数码管称为共阴数码管。常用 LED 数码管显示的数字和字符是 0、1、2、3、4、5、6、7、8、9、A、B、C、D、E、F。

在常用的数码管模块中，一位数码管的封装引脚是 10 个。显示一个 8 字需要 7 个小段，另外还有一个小数点，所以其内部一共由 8 个发光二极管组成，最后还有一个公共端 com，生产商为了封装统一，1 位数码管都封装为 10 个引脚，其中第 3 和第 8 引脚是接在一起的。

我们知道，数码管的公共端可分为共阳极和共阴极。对共阴极数码管来说，它的 8 个发光二极管的阴极在数码管内部是全部连接在一起的，所以称“共阴”, 而它们的阳极是独立的。通常在设计电路时一般把阴极接地，因此，当给数码管的任何一个阳极提供一个合适的高电平时，对应的这个发光二极管段就被点亮了。

如果想要共阴极数码管显示出一个“8”字，并且把右下角的小数点也点亮的话，可以给 8 个段

码的阳极全部送高电平 ;如果想让它显示出一个“0”字 ,那么除了给“g”、“dp”这两个段码送低电平外，其余段码全部都送高电平，这样它就显示出“0”字了。想让它显示哪部分，就给相对应的发光二极管送高电平，因此，在显示数字的时候首先做的就是给 0～9 十个数字编码，需要数码管亮什么数字就直接把相应编码送到它的阳极就行了。数码管内部的发光二极管点亮时，至少需要 5 mA 以上的电流，而且电流不可过大，否则会烧毁发光二极管。由于单片机的 I/O 口作为输出口时送不出如此大的电流（只有约 10～50 μA），所以共阴极数码管与单片机连接时需要考虑驱动问题，可以用上拉电阻的方法或使用专门的数码管驱动电路。

共阳极数码管内部 8 个发光二极管的所有阳极是全部连接在一起的。电路连接时，公共端接高电平，因此要点亮的那个发光管就需要给阴极送低电平，此时显示数字的编码与共阴极编码刚好是相反的关系。

（二）数码管的驱动电路

1. 专用驱动芯片

在数字电路中，七段 LED 数码管一般是在译码驱动电路的驱动下工作的，七段数码管使用时应当配用相应的译码驱动器。常用的译码驱动器有 74LS247（配共阳极数码管）、74HC4511（配共阴极数码管）等。下面只介绍共阴极数码管对应的译码驱动芯片 74HC4511（或者 CD4511）。

集成芯片 74HC4511 是将锁存、译码、驱动三种功能集于一身的“三合一”器件。锁存器的作用是避免在计数过程中出现跳数现象，便于观察和记录。译码器的作用是将 4 位 BCD 码转换成数码管显示所需要的七段码，再经过大电流反相器，驱动 LED 数码管（共阴极）。译码器属于非时序电路，其输出状态与时钟无关，仅取决于输入的 BCD 码。显示时利用该器件的七段码输出端 a、b、c、d、e、f、g 可直接驱动数码管 LED（共阴极）。

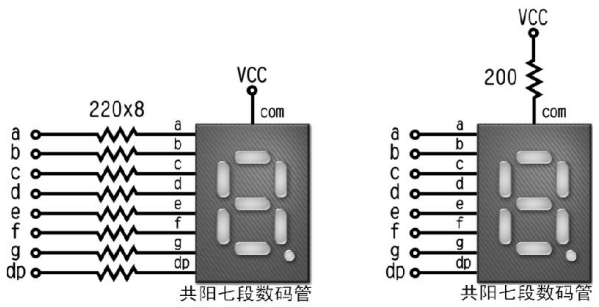
74HC4511 的功能表如表 2.1 所示。

表 2.1　74HC4511 译码驱动器的功能表

显示	输　入							输　出						
	$\overline{LT}$	$\overline{BI}$	$\overline{LE}$	D	C	B	A	a	b	c	d	e	f	g
8	0	×	×	×	×	×	×	1	1	1	1	1	1	1
空白	1	0	×	×	×	×	×	0	0	0	0	0	0	0
0	1	1	0	0	0	0	0	1	1	1	1	1	1	0
1	1	1	0	0	0	0	1	0	1	1	0	0	0	0
2	1	1	0	0	0	1	0	1	1	0	1	1	0	1
3	1	1	0	0	0	1	1	1	1	1	1	0	0	1
4	1	1	0	0	1	0	0	0	1	1	0	0	1	1
5	1	1	0	0	1	0	1	1	0	1	1	0	1	1
6	1	1	0	0	1	1	0	0	0	1	1	1	1	1
7	1	1	0	0	1	1	1	1	1	1	0	0	0	0
8	1	1	0	1	0	0	0	1	1	1	1	1	1	1
9	1	1	0	1	0	0	1	1	1	1	0	0	1	1
空白	1	1	0	1	0	1	0	0	0	0	0	0	0	0
空白	1	1	0	1	0	1	1	0	0	0	0	0	0	0
空白	1	1	0	1	1	0	0	0	0	0	0	0	0	0
空白	1	1	0	1	1	0	1	0	0	0	0	0	0	0
空白	1	1	0	1	1	1	0	0	0	0	0	0	0	0
空白	1	1	0	1	1	1	1	0	0	0	0	0	0	0
与输出相同	1	1	1	×	×	×	×	决定于 $\overline{LE}$ 变高电平前的输入状态						

2. 三极管驱动电路

如图 2.4(a)所示，对于单个的共阳极七段数码管，可以把它的共阳极公共端 com 接 VCC，然后将每一个阴极引脚各接一个限流电阻，限流电阻可选 100～330 Ω 之间的阻值。电阻值越大，亮度越弱；电阻值越小，亮度越亮。如果采用图 2.4 (b) 所示的接法，在 com 端只使用一个限流电阻，则显示不同的数字时，将会有不同的亮度，并不妥当。



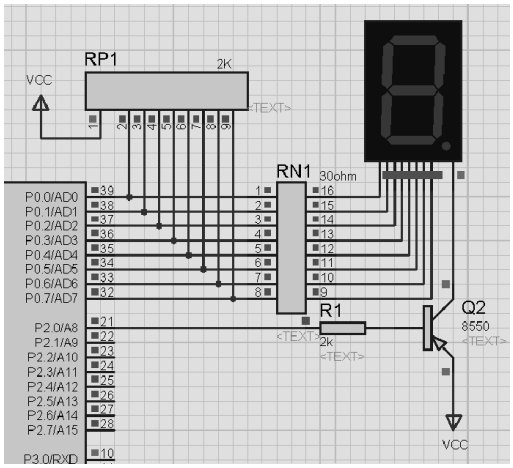
（a）恰当的连接　　（b）不恰当的连接

图 2.4　共阳极数码管的连接方法

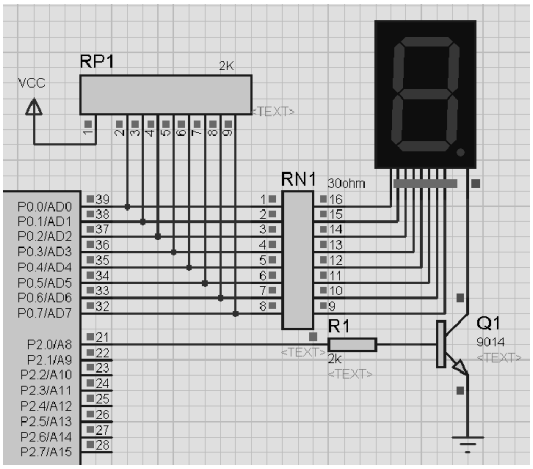
对于单个的共阴极七段数码管，如果把它的共阴极公共端 com 接 GND，然后将每一个阳极引脚各接一个限流电阻，再接到单片机的 8 个 I/O 引脚，这种接法将会使七段数码管显示字符的亮度不足。因为单片机 I/O 引脚的输出方式为强下拉/弱上拉，而高电平输出电流很小，所以数码管会很暗，因此需要通过驱动电路进行电流的放大。

数码管的驱动电路可以采用三极管驱动，也可以用带负载能力比较强的 TTL 型 OC 门电路及 74HC 系列或专用的 CD4000 系列的驱动电路直接驱动。在单片机控制数码管电路中，较常见的驱动方法是采用便宜又实用的三极管驱动。

图 2.5 给出了单片机控制 LED 数码管的两种常见驱动电路。图 2.5（a）为共阳数码管的驱动电路，图中单片机的 I/O 口（如果是用 P0 端口，还需要加上拉电阻，建议使用 P2 端口）接一个 1～2 kΩ 的限流电阻，然后接 PNP 型三极管的基极，三极管的发射极接 VCC，集电极接数码管的公共端 com。数码管的段码输出端（a、b、c、d、e、f、g、dp）每个都接一个 100～470 Ω 的电阻（每个段的电阻值要相同，以免产生亮度不同现象），然后再接单片机 I/O 口（比如 P0、P1 或 P3 端口，一般为加过上拉电阻后的 P0 端口）。PNP 型三极管一般采用中小功率三极管 8550（其他中小功率的 PNP 管也行）。



（a）共阳数码管



（b）共阴数码管

图 2.5　三极管驱动数码管电路

图 2.5（b）为共阴极数码管的驱动电路，图中单片机的 I/O 口接一个 1～2 kΩ 的限流电阻，然后接 NPN 型三极管的基极，三极管的发射极接 GND，集电极接数码管的公共端 com。数码管的段码输出端（a、b、c、d、e、f、g、dp）每个都接一个 100～470 Ω 的电阻，然后再接单片机 I/O 口。NPN 型三极管一般采用三极管 9014（其他中小功率的 NPN 管也行）。

二、数码管的显示编码原理

（一）数码管的静态显示

数码管的静态显示也称静态驱动显示。静态驱动是指每个数码管的每一个段码都由一个单片机的 I/O 端口进行驱动，或者使用如 BCD 码、二-十进制译码器译码进行驱动。静态驱动的优点是编程简单，显示亮度高；缺点是占用 I/O 端口较多。采用静态显示的方法要想驱动 5 个数码管的段码则需要 5×8 = 40 根 I/O 引脚来驱动，要知道一个 51 单片机可用的 I/O 引脚才 32 个。因此实际应用时必须增加译码驱动器进行驱动，增加了硬件电路的复杂性。

当多位数码管应用于某一系统时，它们的“位选”是可独立控制的，而“段选”是连接在一起的，

我们可以通过位选信号控制哪几个数码管亮，而在同一时刻，因为它们的段选是连接在一起的，所以送入所有数码管的段选信号都是相同的，那么它们显示的数字也必定一样，数码管的这种显示方法叫做静态显示。

下面用 C 语言写一个简单的程序，先让一个共阴极数码管显示一个“8”字。假定单片机的 P0 端口接共阴极数码管的 8 个段码，P2.0 引脚通过 NPN 三极管驱动数码管的公共端 com。在操作时，可以将数据从单片机的 P0 口直接送出到数码管的 8 个段码管脚。由于数码管为共阴极，所以 com 选通时为低电平，位选（公共端 com）关闭时为高电平。位选确定后，再确定段选，要显示“8”，那么只有 dp 段为 0，其余段都为 1，所以接着应该将 P0 的数据输出端再送一个 0x7F（二进制为 0111 1111）。程序代码如下：

```
#include <reg51.h>              //包含 51 系列单片机头文件
sbit wei=P2^0 ;                 //声明位选接 P2.0 引脚
void main (  )                  //主程序开始
{
    while ( 1 )                 //无穷循环
    {
    wei=1 ;                     //送位选信号到 NPN 三极管
    P0=0x7f ;                   //送段码显示 8
    }
}
```

（二）数码管的编码原理

下面介绍一种编码方法，在数码管显示数字时通常都要用到这种编码方法。刚才显示的数

字是“8”，给 P0 端口送的数据是 0x7f，这是根据实际电路图给出的编码。不同的电路，编码可能不同，共阳极数码管的编码与共阴极数码管的编码也是不同的，因此一定要掌握编码原理，也就是要明白数码管显示的原理。

对于共阴极数码管，如果 a 连接 8051 输出端口的最低位，dp 连接 8051 输出端口的最高位，并且希望小数点不亮，则可以将字符 0～F 进行如表 2.2 所示的编码。

表 2.2　共阴极数码管编码表

符　号	编　码	符　号	编　码	符　号	编　码	符　号	编　码
0	0x3f	4	0x66	8	0x7f	C	0x39
1	0x06	5	0x6d	9	0x6f	D	0x5e
2	0x5b	6	0x7d	A	0x77	E	0x79
3	0x4f	7	0x07	B	0x7c	F	0x71

在用 C 语言编程时，编码定义方法如下：

```
unsigned char code table[] = {
0x3f，0x06，0x5b，0x4f，                    //对应符号“0”、“1”、“2”、“3”
0x66，0x6d，0x7d，0x07，                    //对应符号“4”、“5”、“6”、“7”
0x7f，0x6f，0x77，0x7c，                    //对应符号“8”、“9”、“A”、“B”
0x39，0x5e，0x79，0x71}；                   //对应符号“C”、“D”、“E”、“F”
```

编码定义方法与 C 语言中的数组定义方法非常相似，不同的地方就是在数组类型后面多了一个 code 关键字，code 即表示编码的意思。需要注意的是，单片机 C 语言中定义数组时是占用内存 RAM 空间的，而定义编码时是直接分配到程序空间中，编译后编码占用的是程序存储空间，而非内存空间。

unsigned char 是数组类型，也就是数组中元素变量类型，table 是数组名，我们可以自由定

义它，但是不要和关键字重名；table 后面必须加中括号[]，中括号内部要注明当前数组内的元素个数，也可以不注明，C51 编译器在编译时能够自动计算出来，因此在使用时经常不注明。等号右边用一个大括号包含所有元素，大括号后面加一个分号，大括号内部元素与元素之间用逗号隔开，注意最后一个元素后面不要加逗号。调用数组方法如下：

P0 = table[4]；

即将 table 这个数组中的第 5 个元素直接赋给 P0 口，也就是：

P0 = 0x66；

需要注意的是，在调用数组时，table 后面中括号里的数字是从 0 开始的，对应后面大括号里的第 1 个元素。有了这种编码方法，在编写数码管显示程序时就会方便很多。

对于共阳极数码管，如果 a 连接 8051 输出端口的最低位，dp 连接 8051 输出端口的最高位，并且希望小数点不亮，则可以将字符 0～F 进行如表 2.3 所示的编码。

表 2.3　共阳极数码管编码表

符　号	编　码	符　号	编　码	符　号	编　码	符　号	编　码
0	0xc0	4	0x99	8	0x80	C	0xa7
1	0xf9	5	0x92	9	0x90	D	0xa1
2	0xa4	6	0x82	A	0xa0	E	0x84
3	0xb0	7	0xf8	B	0x83	F	0x8e

同样的，用 C 语言编程时，编码定义方法如下：

```
unsigned char code table[] = {
0xc0，0xf9，0xa4，0xb0，          //对应符号“0”、“1”、“2”、“3”
0x99，0x92，0x82，0xf8，          //对应符号“4”、“5”、“6”、“7”
0x80，0x90，0xa0，0x83，          //对应符号“8”、“9”、“A”、“B”
```

```
    0xa7，0xa1，0x84，0x8e}；                    //对应符号“C”、“D”、“E”、“F”
```

三、数码管的静态显示编程方法

下面结合项目一介绍过的延时程序，实现一位七段数码管静态显示功能。

【例 2.1】 让实验板上共阳极数码管点亮，依次显示 0～F，时间间隔为 0.5 s，循环下去。

实验板上单片机的 P0 端口接共阳极数码管的 8 个段码，P2.0 引脚通过 PNP 三极管 8550 驱动数码管的公共端 com。

根据本例静态控制七段数码管显示 0～F 的先后顺序和各个字符亮灭时间，可以画出对应的时序图，具体程序流程图与算法描述此处略。参考程序源代码如下：

```
/*********************************************************************
  静态显示程序：Tube1_Static.c 一个数码管静态显示的简单程序
*********************************************************************/

//==声明区=======================================

  #include <reg51.h>                          //包含 8051 寄存器的头文件
  #define uchar unsigned char
  #define uint unsigned int
  sbit wei=P2^0；                             //声明位选的位置

  uchar num；                                 //声明变量 num

  uchar code table[]={                        //声明段码数组变量

    0xc0，0xf9，0xa4，0xb0，                  //对应符号“0”、“1”、“2”、“3”

    0x99，0x92，0x82，0xf8，                  //对应符号“4”、“5”、“6”、“7”

    0x80，0x90，0xa0，0x83，                  //对应符号“8”、“9”、“A”、“B”
```

```
    0xa7，0xa1，0x84，0x8e}；                //对应符号“C”、“D”、“E”、“F”
void delayms（uint）;                        //声明 1ms 延时函数
//==主程序=========================================
void main（ ）                               //主函数
{
      wei=0；                                //关闭位选
      P0=0xff；                              //初始化 P0 端口
      while（1）                             //无穷循环
      {
        for（num=0；num<16；num++）          //循环 16 次
          {
              wei=1；                        //位选中
              P0=table[num]；                //P0 送段码
              delayms（500）;                //延时 0.5 s
              wei=0；                        //消影
          }
      }
}
//==子程序=========================================
void delayms（uint x）                       //延时函数
{
    uint i；                                 //声明变量 i
    while（x--）                             //循环 x 次
```

```
    for ( i=120 ; i>0 ; i-- ) ;                    //循环 120 次 ( 约 1 ms )
  }
```

将代码编译下载到实验板后，可以看到共阳极数码管上的数字依次从 0～F 变换显示。本程序有两个方面需要注意：

第一，在刚进入主函数后，执行了一次位选关闭命令，然后接着便进入了大循环。因为本例的要求是让数码管依次从 0～F 变换显示，所以位选命令关闭执行一次，也就是开始的时候先关闭数码管显示，然后再根据需要操作位选。

第二，在 while (1) 大循环中，使用了一个 for 循环语句，原来仅用 for 作延时，而在这里用 for 语句来实现一个规定数目的有限循环，大家要掌握它的用法，以后会经常用到。

【例 2.2】　让实验板上共阳极数码管点亮，倒计时显示 9～0，时间间隔为 1 s，循环下去。

实验板上单片机的 P0 端口接共阳极数码管的 8 个段码，P2.0 引脚通过 PNP 三极管 8550 驱动数码管的公共端 com。

根据本例静态控制七段数码管倒计时显示 9～0 的先后顺序和各个字符亮灭时间，可以画出对应的时序图，具体程序流程图与算法描述此处略。参考例 2.1，得到的程序源代码如下：

```
/*****************************************************************
  静态显示程序：Tube2_Static.c 数码管静态显示的倒计时程序
*****************************************************************/
//==声明区=======================================
  #include <reg51.h>                          //包含 8051 寄存器的头文件
  #define uchar unsigned char
  #define uint unsigned int
  sbit wei=P2^0 ;                             //声明位选的位置
  uchar num ;                                 //声明变量 num
  uchar code table[]={                        //声明段码数组变量
  0xc0 , 0xf9 , 0xa4 , 0xb0 , 0x99 ,          //对应符号“0”、“1”、“2”、“3”、“4”
```

```
 0x92，0x82，0xf8，0x80，0x90}；            //对应符号“5”、“6”、“7”、“8”、“9”
void delayms（uint）；                      //声明 1ms 延时函数
//==主程序=====================================
 void main（ ）                             //主函数
 {
        wei=0；                             //关闭位选
        P0=0xff；                           //初始化 P0 端口
        while（1）                          //无穷循环
        {
          for（num=9；num>=0；num--）       //循环 10 次
           {
                wei=1；                     //位选中
                 P0=table[num]；            //P0 送段码
                 delayms（1000）；          //延时 1 s
                wei=0；                     //消影
           }
       }
 }
//==子程序=====================================
 void delayms（uint x）                     //延时函数
 {
    uint i；                                //声明变量 i
    while（x--）
        for（i=120；i>0；i--）；            //延时约 x 个 1 ms
 }
```

【任务实施】

1. 任务目标

本任务要求设计一个 9 s 倒计时电路。它由以下几部分组成：51 单片机、时钟电路、复位电路、电源电路、共阳数码管显示及驱动电路组成。具体任务目标如下：

（1）进一步熟悉单片机各个端口的输入/输出功能和使用方法。

（2）掌握单片机控制 1 个七段数码管接口电路的硬件设计和静态显示的编程方法。

2. 实施步骤

（1）各部分芯片及元件的选择：

① 根据本任务需要，这里推荐选择的 51 单片机型号是 DIP40 封装的 STC89C52RC。

② 单片机最小系统电路的相关元器件可以参考项目一中的任务一进行选择。

③ 数码管采用共阳七段数码管，驱动电路元器件采用中小功率 PNP 型三极管 8550。

（2）各部分电路的设计：

① 数码管显示电路的设计：数码管的公共端 com 接驱动电路的输出，数码管的段码输出

端（a、b、c、d、e、f、g、dp）每个都接一个 100～470 Ω 的电阻后再接单片机 I/O 口（比如 P0、P1 或 P3 端口，一般为加过上拉电阻后的 P0 端口）。

② 驱动电路的设计：单片机的 I/O 口 P2.0（第 21 引脚）接一个 1～2 kΩ 的限流电阻，然后接 PNP 型三极管 8550 的基极，三极管 8550 的发射极接 VCC，集电极接数码管的公共端 com。

（3）画出相应的 PCB 图：使用相应的电路图绘制工具 Altium Designer 或者 Protel 99SE 等软件画出相应的 PCB 图。

（4）编写软件，并进行软件仿真：

① 运用 Keil 软件建立一个工程，用 C 语言编写一个程序，实现数码管从数字 9 到数字 0 每秒倒计时循环显示。编好的程序添加到工程中进行调试并产生 hex 文件。参考源程序如下：

```
#include <reg51.h>
#define uchar unsigned char
#define uint unsigned int
sbit wei=P2^0；
uchar num；
uchar code table[]={
0xc0，0xf9，0xa4，0xb0，0x99，
0x92，0x82，0xf8，0x80，0x90｝；
void delayms（uint）；
void main（ ）
{
        wei=0；
        P0=0xff；
        while（1）
        {
          for（num=9；num>=0；num--）
            {
                  wei=0；
                  P0=table[num]；
                  delayms（1000）；
                  wei=l；
            }
       }
 }
void delayms（uint x）
{
    uint i；
    while（x--）
        for（i=120；i>0；i--）；
}
```

② 运用 Proteus 软件进行仿真：打开 Proteus 应用程序，在其中找到元器件（AT89C51、

7SEG-MPX1-CA 显示数码管，RES 电阻等）按图 2.6 连接好，将 Keil 生成的 hex 文件装载入 51 单片机芯片中，进行仿真，仿真效果如图 2.6 所示。

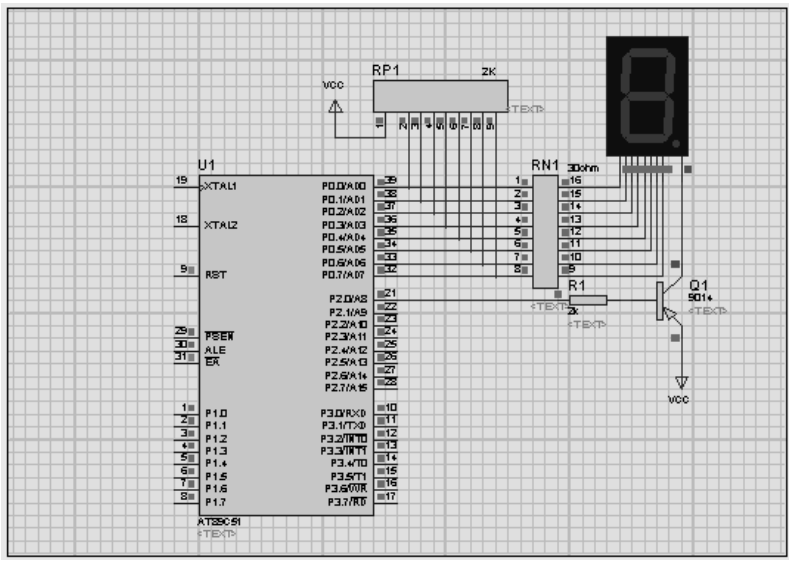


图 2.6　数码管 9 s 倒计时仿真电路

（5）装配单片机驱动 1 个七段数码管的电路板。

① 根据指导老师发放的焊接装配图，参考任务小组设计的电路图，领取相应的元器件并识别、测试元器件性能是否满足需要。

② 工具准备：电烙铁、焊锡丝、金属镊子、尖嘴钳、斜口钳、吸锡器等焊接工具，万用表、示波器、直流稳压电源等测试工具。

③ 按照工艺要求安装元器件并焊接。步骤如下：

- 在给出的实验板上安装共阳极数码管并焊接。
- 按焊接图所示插入三极管驱动电路的相关元器件电阻 R1～R9 和三极管 Q1 并焊接。
- 按焊接图所示插入输入/输出端口的连接插座 JP1、JP3 和上拉电阻 RP3 并焊接。

（6）把编译好的程序下载烧写到单片机芯片中，进行软、硬件联合调试。

（7）观察实验现象，如有故障，测试电路板并查找原因。

3. 技术报告及评测

将测试点、测试结果及故障原因分析记录下来。测试点有：电路板电源电压（JP10）、单片机电源管脚（第 20、40 脚）、单片机复位管脚（第 9 脚）、单片机晶体管脚（第 17、18 脚）、单片机 EA 管脚（第 31 脚）、七段数码管 DS2 的 8 个段码管脚、七段数码管 DS2 的公共端 com 管脚，驱动三极管 8550 的 E、B、C 极，电阻 R1～R9 的两端。

任务完成后撰写技术报告及效果评测。

任务二　2 位七段数码管显示电路的设计与制作

【任务要求】

根据 51 单片机的最小系统电路构成，结合两位七段数码管动态显示驱动电路的特点，设计与制作一个 51 单片机对 2 位七段数码管的驱动电路，运用单片机的相关理论知识，完成单片机控制 2 位七段数码管 60 秒倒计时的显示程序编写，并进行实际电路的调试与检测。

具体的任务要求如下：

（1）用 51 单片机控制 2 位七段数码管进行 60 秒倒计时显示。

（2）选出适合本学习任务的 2 位七段数码管元件以及其他元器件。

（3）根据设计要求，设计出单片机对 2 位七段数码管的驱动电路。

（4）焊接、制作单片机驱动 2 位七段数码管的电路板。

（5）编写单片机对 2 位七段数码管的显示程序，并进行软硬件联合调试。

（6）协作解决设计与制作中遇到的问题。

【相关知识】

一、数码管的动态显示原理

在实际的单片机系统中，往往需要显示多位数字信息。用数码管显示多位信息时，由于每个数码管至少需要 8 个 I/O 口，如果需要多个数码管，就需要太多 I/O 口，而单片机的 I/O 口资源是有限的。在实际应用中，一般采用动态显示的方式解决此问题。

动态显示是把所有数码管的段选全部连接在一起进行交替显示。将所有数码管的段选全部连接在一起，如何能显示不同的内容呢？原来，动态显示利用人的视觉暂留效应使人看到多个数码管同时显示各自的信息，它是一种最常见的多位显示方法，应用非常广泛。

2 个数码管在同一时间进行显示可用两种不同的方式获得：第一就是传统的方式，一个数码管连接一个译码及驱动电路，这种方式需要太多 I/O 口；第二种方式是利用人眼的视觉暂留效应，把 2 个数码管按一定顺序（从左至右或从右至左）循环进行点亮，当点亮的频率（即扫描频率）很低时，人们看到的是数码管一个个被点亮；然而，当点亮频率足够高时，看到的不再是一个一个被点亮，而是全部同时显示（点亮），与传统方式得到的结果看起来是一样的。因此只要给数码管驱动电路一个足够高的扫描工作频率，那么就可以实现 2 个（或更多）数码管同时被点亮。如果在 2 个（或更多）数码管点亮的同时，同步地切换 BCD 七段译码器的输入数据，就可以实现 2 个（或更多）数码管显示不同的数据。而产生这个扫描频率的驱动电路，可以通过硬件的计数器和译码器来实现，BCD 七段译码器的输入数据切换电路，可以通过计数器的输出来控制几个多路数据选择器电路实现，只要计数频率足够高，就可以实现显示的要求，当然，以上硬件电路也可以通过软件编程来代替实现。

在进行软件编程时，需要输出段选和位选信号。位选信号选中其中一位数码管，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，再选中另一位数码管，再输出对应的