微机原理与接口技术

课程设计报告

姓　　名：　王恒立

学　　号：　**3140104567**

任课教师：　唐建中

　　林勇钢

　　李伟

提交日期：　**2017 年 7 月 3 日**

目　　录

一、　设计任务说明 3

二、　硬件系统设计 3

2.1　器件选型说明 3

2.2　原理设计 4

2.3　布局和布线设计 8

三、　软件系统设计 10

3.1　软件系统结构 10

3.2　数据排列功能的实现 10

四、　系统调试 11

4.1　调试过程 11

4.2　调试过程中的问题 11

五、　系统测试 13

5.1　测试工具和软件 13

5.2　测试方法 13

六、　课程设计总结 14

七、　附件 15

附件 1　硬件系统原理图 15

附件 2　程序及注释 18

附件 3 PCB 图 26

一、 设计任务说明

1. 设计由 10 个数字键、1 个设置键，5 个数码管组成的电路板；
2. 在设置键盘按下后，操作者连续输入任意 10 个 3 位数字（3 个数码管显示录入数据，另外两个数码管显示已录入的数据的个数）；
3. 输入完成后，单片机自动把所有录入的数按照从小到大的顺序以 2 秒为周期依次显示出来。

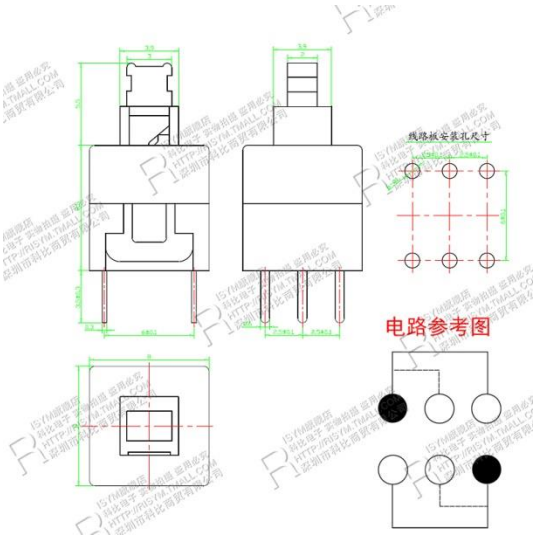
二、 硬件系统设计

2.1 器件选型说明

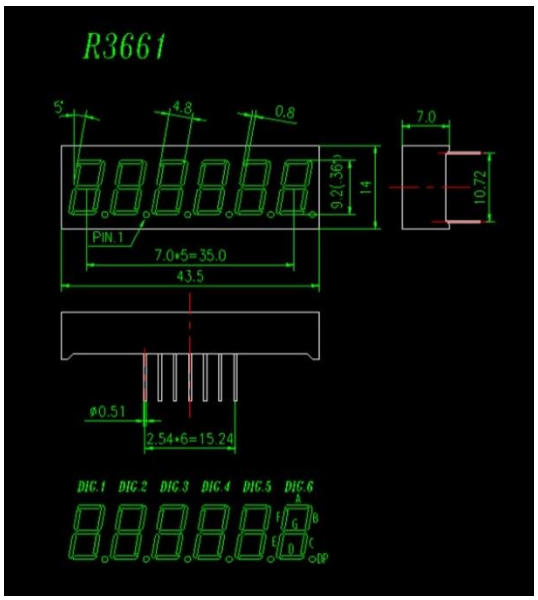
在本次课程设计过程中，我主要参考了短学期的单片机开发板上的电路原理进行设计，因此，本次选取的器件与开发板上的器件大致相同，下表为具体说明。

设计编号	物料描述	数量	说明
USB1	普通插座/4PIN/250V/1.5A/双排/弯角/母座/2.5mm/USB-B	1	
U1	自锁开关/8*8mm	1	
U2	GH340G/USB 转串口 IC	1	用于单片机烧录程序
LED	六位数码管/3661BS/共阳/0.36 英寸/14 个管脚（上 7 下 7）	1	用于输入输出过程的显示
U3	IC 测试座/40PIN	1	
ResT	10K/排阻	1	用于 P0 口的上拉电阻
LED1	LED 发光二极管/红色/3/80mcd/20m	1	开关指示灯
EC1	电解电容/105℃/100uF/16V/5*7mm/直角/P=2.0mm	1	
Q1-Q6	小信号三极管/PNP/-30V/500mA/TO-92/9012	6	用于放大电流
Q7	小信号三极管/NPN/25V/500mA/TO-92/9014	1	
R1-R6	碳膜电阻/1/8W/ 4.7KΩ±5%	6	
R7,R8,R10	碳膜电阻/1/8W/510Ω±5%	3	
R11	碳膜电阻/1/8W/1KΩ±5%	1	
R9,R12	碳膜电阻/1/8W/10KΩ±5%	2	
C1,C2,C5,C6	30pf/50V/瓷片电容	4	用于晶振的起振
C3,C4,C7,C8	0.1uf/50V/瓷片电容	4	电源正负极之间的滤波电容
S1-S17	微动开关/12VDC/30mA/6*6*4.3mm/插件	17	
D1	快恢复二极管/1N18/-65~+150℃/DO-41	1	起到续流保护的作用
P2-P7	连接器/单排单塑排针/直角/2.54mm/8pin/L1=3.2/L2=5.9/11.6	6	
P8	连接器/单排单塑排针/直角/2.54mm/6pin/L1=3.2/L2=5.9/11.6	1	
P10	连接器/单排单塑排针/直角/2.54mm/3pin/L1=3.2/L2=5.9/11.6	1	
	单片机/STC89C52	1	
	12M 晶振/DIP	1	
	11.0592M 晶振/DIP	1	
	USB 下载线	1	
	跳线帽	1	
	杜邦线	30	

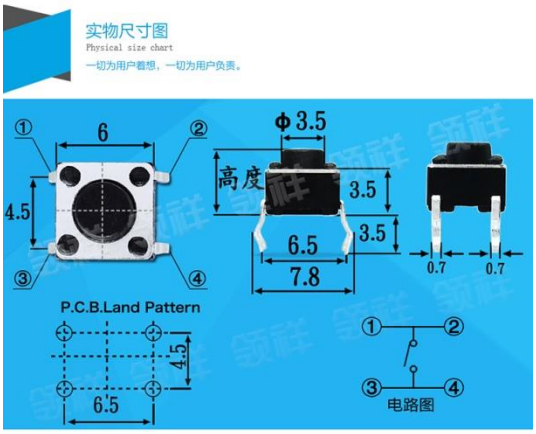
此外，由于本次课程设计我们未拿到关于单片机开发板的器件封装，因此实际设计中所有元器件的封装都是通过自己寻找尺寸绘制的，以下是部分重要元件的封装和尺寸。



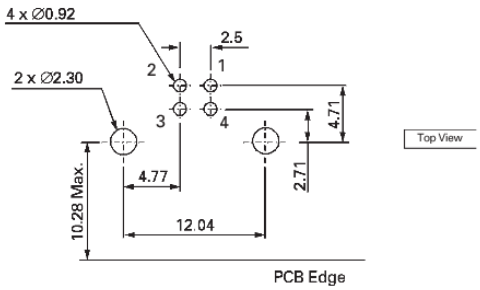
自锁按键开关



六位共阳数码管



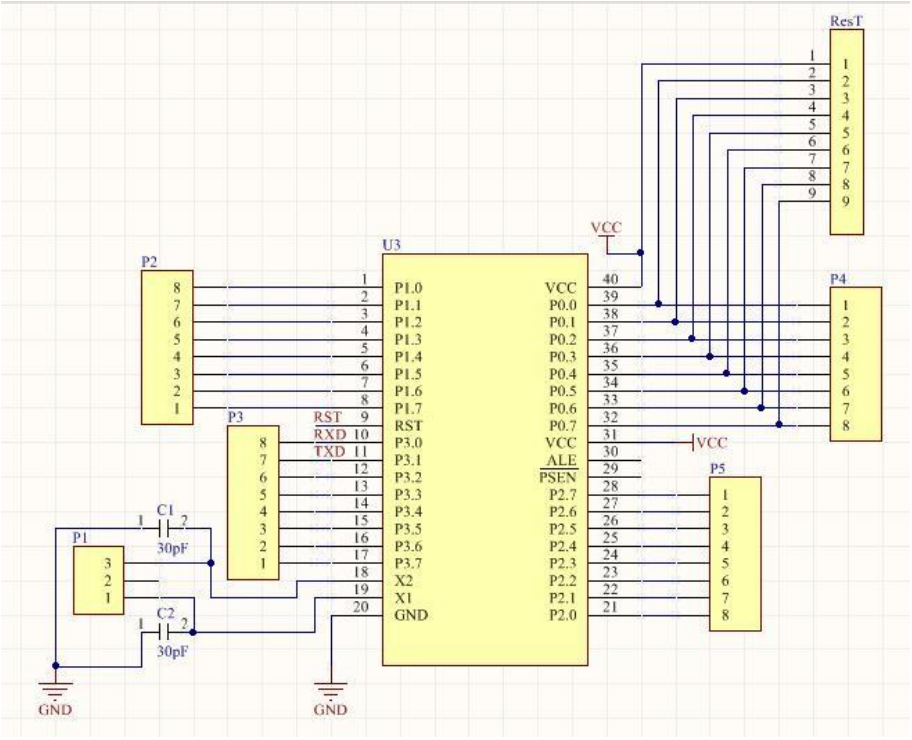
矩阵开关



USB 插座

2.2 原理设计

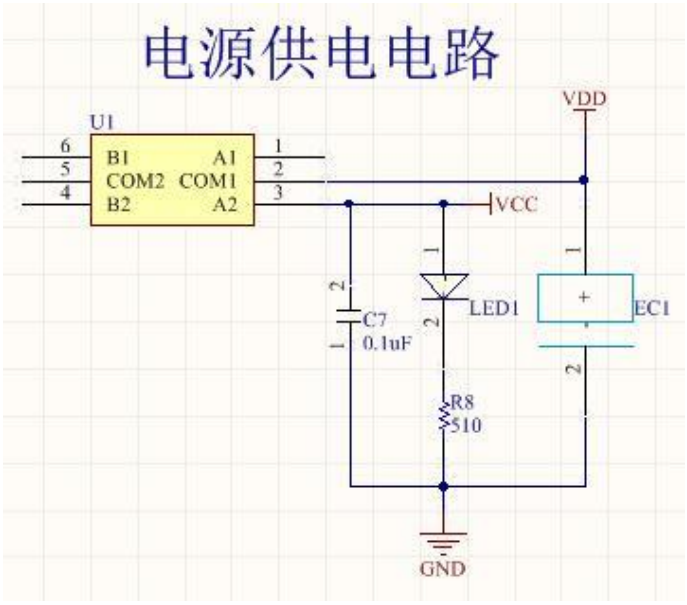
本次课程设计以 STC89C52 单片机为核心，基本的外围电路包括单片机模块电路、电源供电电路、USB 转串口电路、MCU 复位电路、数码管电路以及按键模块电路。其结构与短学期的开发板基本相同，主要是利用了单片机向数码管输出显示结果以及键盘扫描读取两方面的功能，下面将一一进行介绍。



单片机模块电路原理图

单片机模块电路原理图如上图所示。

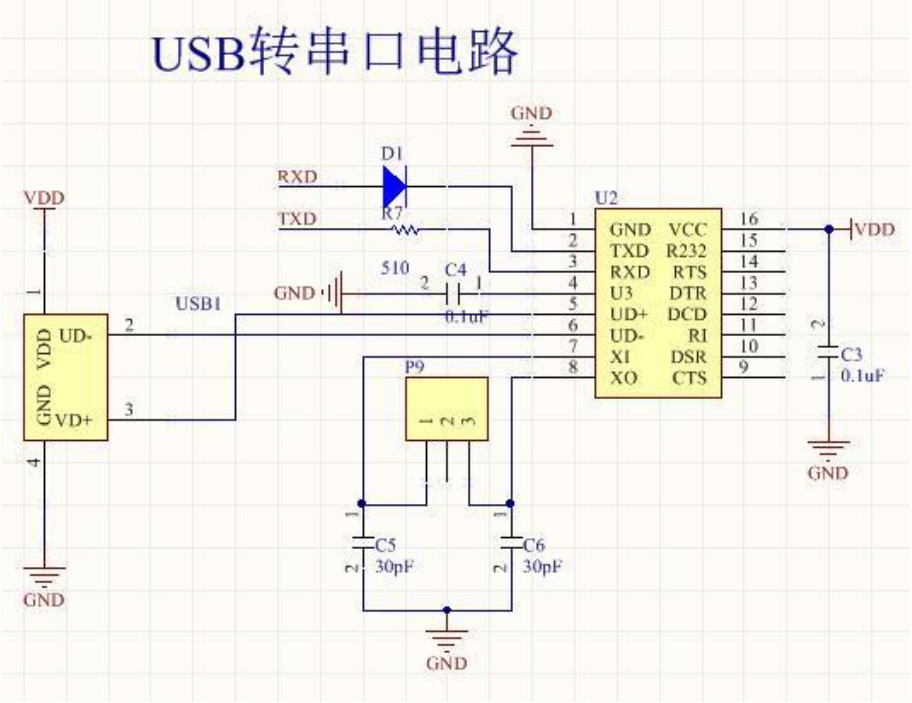
U3 为 IC 测试座 DIP40，测试座上放置芯片 STC89C52。P2、P3、P4、P5 为 8pin 连接器，功能是作为单片机的 IO 口，实际使用过程中使用杜邦线进行连接。ResT 为 10K 排阻，连接在 P0 口处，用于增加单片机承受负载的能力。P1、C1、C2 构成晶振回路，其中 P1 为 3pin 连接器，实际使用过程中把 11.0592M 晶振插在 P1 连接器上。



电源供电电路原理图

电源供电电路原理图如上图所示。

VDD 为单片机通过 USB 接口从计算机获得的+5V 电源电压，VCC 为单片机模块中工作所需的电源电压。U1 为电源自锁开关，当开关断开时，COM1 和 A2 不导通，即使单片机通过 USB 接口与计算机相连，VCC 也无电压，单片机不工作；当开关闭合后，COM1 和 A2 导通，如果单片机通过 USB 接口与计算机相连，VCC 电压会和 VDD 相等，均为+5V，单片机工作，此时 LED1 导通，电源指示灯亮。此外，EC1 为电解电容，其与 C7 的作用是滤除外界辐射的高频干扰以及单片机工作时自身产生的脉冲干扰。



USB 转串口电路原理图

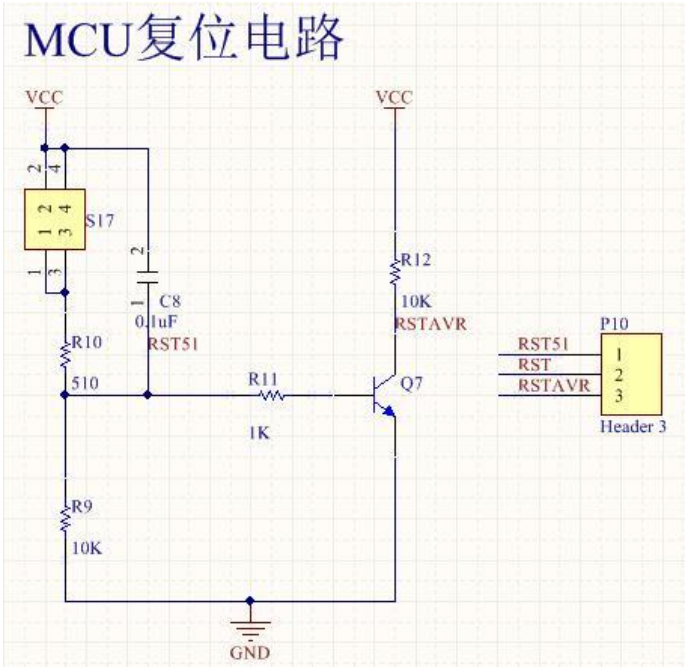
USB 转串口电路原理图如上图所示。

USB1 为 USB-B 普通插座，用于连接 USB 下载线。U2 为 CH340G，是 USB 转串口 IC。P9、C5、C6 构成晶振回路，其中 P9 为 3pin 连接器，实际使用过程中把 12M 晶振插在 P9 连接器上。D1 为快恢复二极管 1N4148，其作用是截止反相电流，防止下载程序时 CH340G 给单片机上电，使程序下载失败。此外，C3、C4 的作用是滤除外界辐射的高频干扰以及单片机工作时自身产生的脉冲干扰。

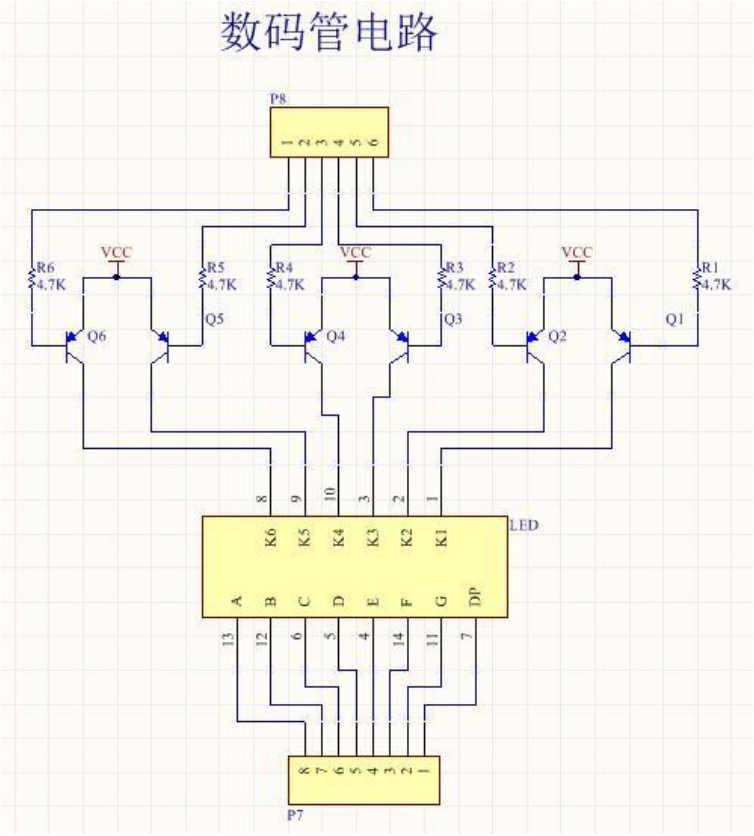
MCU 复位电路原理图如下图所示。

P10 为 3pin 连接器，实际使用过程中使用短路帽将 1、2 或 2、3 连接起来，可形成两种不同的复位模式。由于 RSTAVR 一直保持高电平，当短路帽采用 2、3 连接方式时，RST 也一直保持高电平，单片机复位功能始终有效。当短路帽采用 1、2 连接方式时，如果微动开关 S17 没有按下，那么电容 C8 两端电压近似 5V，RST51 和 RST 均为低电平，单片机复位功能无效；当微动开关 S17 按下时，电容 C8 迅速放电，RST51 和 RST 均为高电平，单片机复位功能有

效。



MCU 复位电路原理图

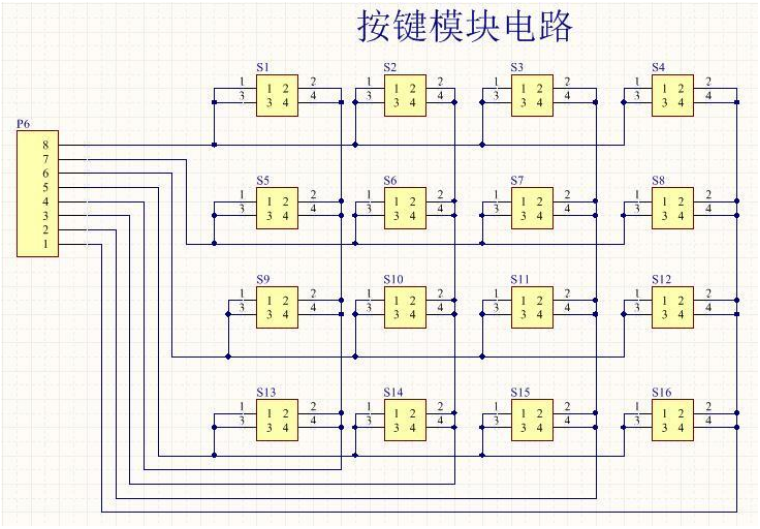


数码管电路原理图

数码管电路原理图如上图所示。

LED 为六位共阳数码管 3661BS，八位数码控制端与 P7 相连，P7 为 8pin 连接器，实际使

用过程中使用杜邦线进行连接。六位使能端通过三极管放大电路与 P8 相连，三极管放大电路的作用是对电流进行放大，P7 为 6pin 连接器，实际使用过程中使用杜邦线进行连接。



按键模块电路原理图

按键模块电路原理图如上图所示。

S1—S16 为 16 个微动开关，组成矩阵键盘，将其按照一定方式与 P6 相连，之后可通过软件方法识别是否有按键按下以及如果有按键按下，按下按键的编号是多少。P6 为 8pin 连接器，实际使用过程中使用杜邦线进行连接。

2.3 布局和布线设计

针对布局：

1. 考虑到右上方有写有备注，故将数码管放置在备注下方，从而能够便于观察。

2. 将 4*4 矩阵键盘等间距排列在数码管下方，能够方便进行按键，而不会被其余元件所遮挡。

3. 电容引线不能太长，尤其是高频旁路电容不能有引线，故把电容放在靠近连接的地方，使引线最短。

4. 其余元器件放置在合适的位置，使得电路板紧凑、整齐。

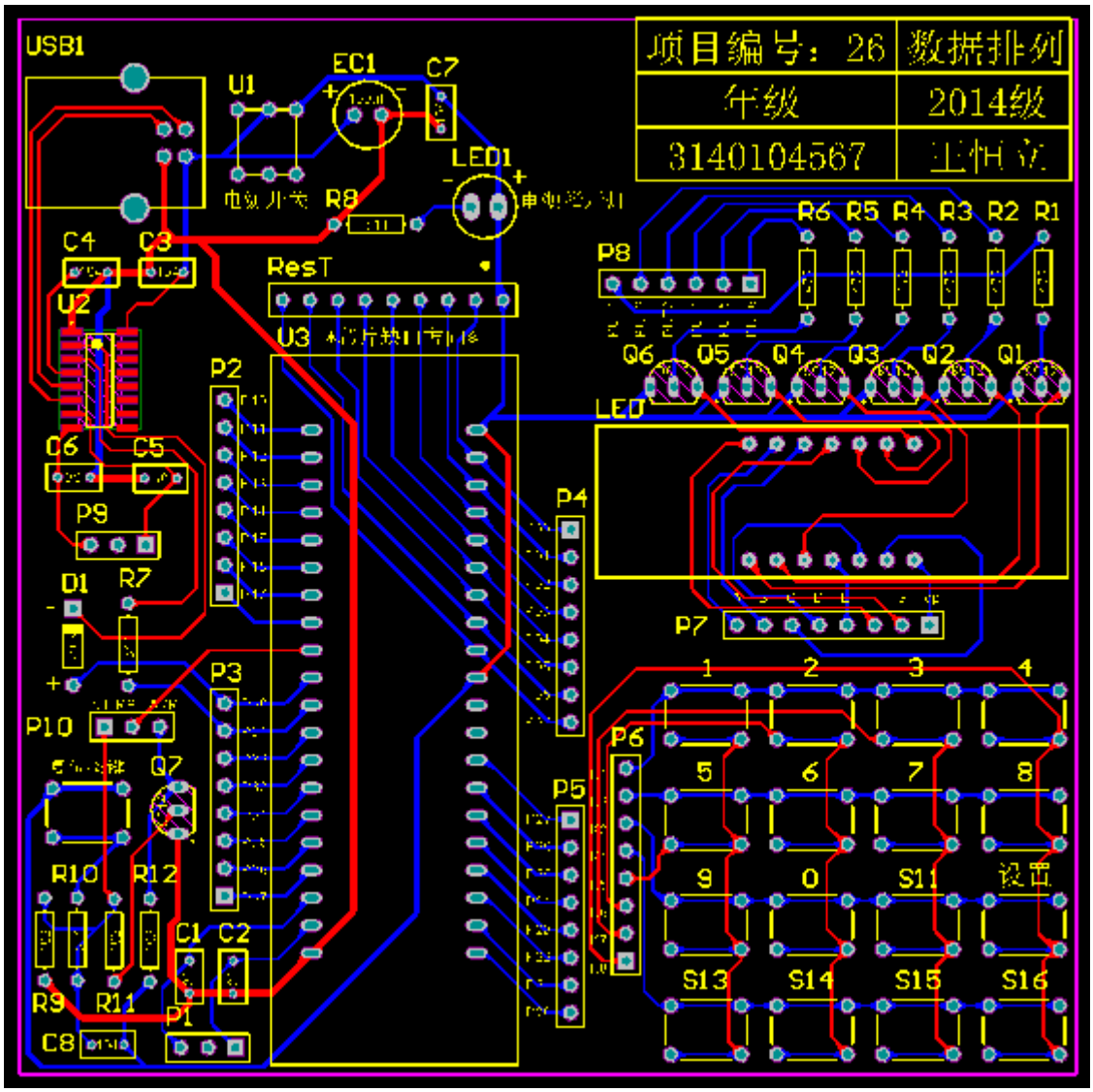
针对布线：

1. 由于单片机与 I/O 之间的连线较为简单，故先布置这之间的连线，保证其整齐。

2. 由单片机逐步向外扩展进行连线，对于芯片的线要留出足够的余量，因为芯片上的线只能够在顶层布线，而无法在底层布线。

3. 最后将所有的电源和地连接到一起，确保电路中没有遗漏的线。

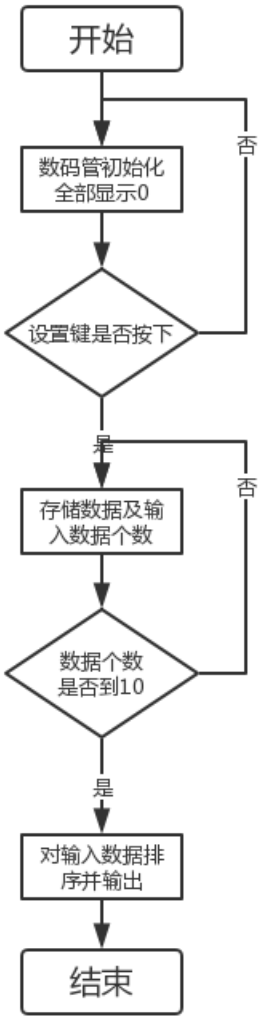
由此可以得到最后的 PCB 布线图如下所示：



PCB 布线图

三、 软件系统设计

3.1 软件系统结构



3.2 数据排列功能的实现

整个程序由以下 main、delayms、display、displaynum、KeyScan、sort、cleardata 几个部分构成，下面将依次进行介绍。

1. main 函数

其主要完成的工作是初始化数码管，使数码管全部显示 0。之后不断检测矩阵键盘，一旦按下设置键后，开始读取输入数据并存储，数据个数到 10 个后对数据进行排序并输出。

2. delayms 函数

其主要实现 ms 级的延时。

3. display 函数

其主要实现数码管动态扫描显示。

4. displaynum 函数

其主要实现把输入的数据按照从小到大的顺序以 2 秒为周期依次显示出来。

5. KeyScan 函数

其主要实现扫描 4*4 的矩阵键盘，并读取了按下的键值。

6. sort 函数

其主要实现对输入的 10 个数据进行从小到大的排序。

7. cleardata 函数

其主要实现清空全部数据的功能。

通过以上几个函数的组合，共同完成了数据排列功能的实现。

四、 系统调试

4.1 调试过程

1. 在板子未到期间，我进行了程序的编写。由于原来的开发板上基本能够满足我的要求，故我首先利用开发板来检验程序的正确性，从而成功编写程序，并完成了程序的调试。

2. 板子焊接完成后，我首先将其与电脑相连，观察是否有串口反应，从而验证 USB 转串口电路是否能够正常工作。

3. 之后我将程序烧录到单片机中，观察数码管是否能够正常的显示数字。

4. 检测按键，观察是否只有按下设置键才能进入数据输入环节，并观察是否能够通过按键进行数据输入，同时检测是否能正常地进入数据输出环节以及数据输出是否按照从小到大的顺序。

经过以上调试，若电路板能够达到要求并没有出现其他问题，即调试成功。

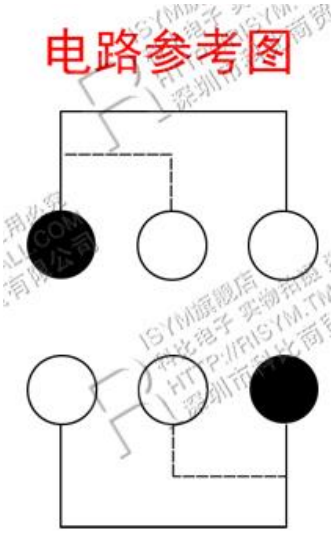
4.2 调试过程中的问题

在调试过程中，由于代码的编写难度并不是很大，所以我较快地编写了所需要的程序并成功调试。但是由于本次是我们自己设计电路，器件选型，所以在硬件调试上还是出现了很多的问题，具体问题如下所示。

1. 自锁开关连线错误

由于本次设计很大一部分电路参考了单片机开发板上的内容，所以对于自锁开关部分的电路我直接使用了开发板上的电路。在第一次上电时，发现电脑是有串口反应的，但是自锁开关无论是否按下，电源指示灯始终保持熄灭状态。

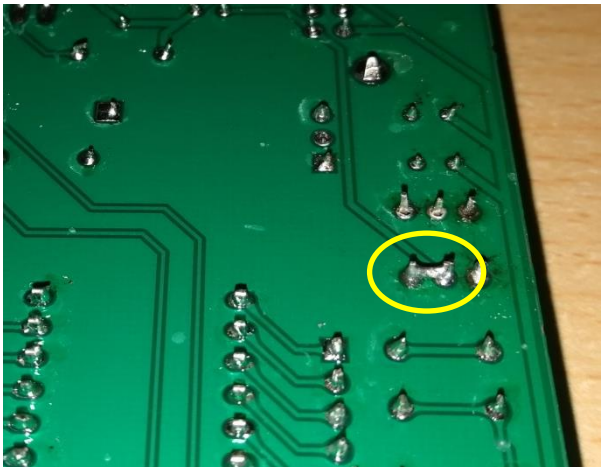
由此我把问题首先锁定到开关上，通过检查后我发现开发板的原理图其实本身就是错误的，自锁开关的内部结构原理图如下图所示。



自锁开关结构原理图

从电路原理图可以看出，VDD 和 VCC 两条线至少有一根联接到黑色端点，才能保证开关的某一状态下两者是相连的，而另一状态则断开。然后，我的 PCB 板上却是 VDD 和 VCC 分别接到了两个白色端点，从而导致两者之间始终不会相连，即无论开关什么状态发光二极管应该都不会发光。

为了解决这个问题，我在焊接的时候将其中一个引脚与黑色端点相连，从而保证开关使能够正常工作的，具体效果如下图所示。



2. 数码管型号错误

我们当初针对共阴共阳问题并没有深入的研究，认为如果型号不对仅需改变相应的数字位的代码即可。但当我将共阴数码管焊接到电路板上后，发现无论是改代码还是改连接方式都无法使数码管发亮。通过查阅资料我发现共阳数码管是采用 PNP 三极管进行电流放大的，而共阴数码管则是通过 NPN 三极管进行电流放大，因此在三极管选定的情况下更改数码管是行不通的。因此，我又重新购买了共阳数码管进行焊接，焊接完成后发现数码管能够正常工作。

解决了上述的问题，电路板终于能够成功实现数据排列的功能。

五、 系统测试

5.1 测试工具和软件

由于本次课程设计的功能是数据排列，所以只需要观察是否能实现数据输入、数据排序以及数据输出功能即可。

5.2 测试方法

1. 首先上电后观察是否只有摁下设置键后才能进入数据输入阶段，发现只有摁下设置键后才能进入数据输入阶段。

2. 通过按下设置键进入数据输入阶段，观察是否能进行数据输入，发现可以正常进行数据输入。

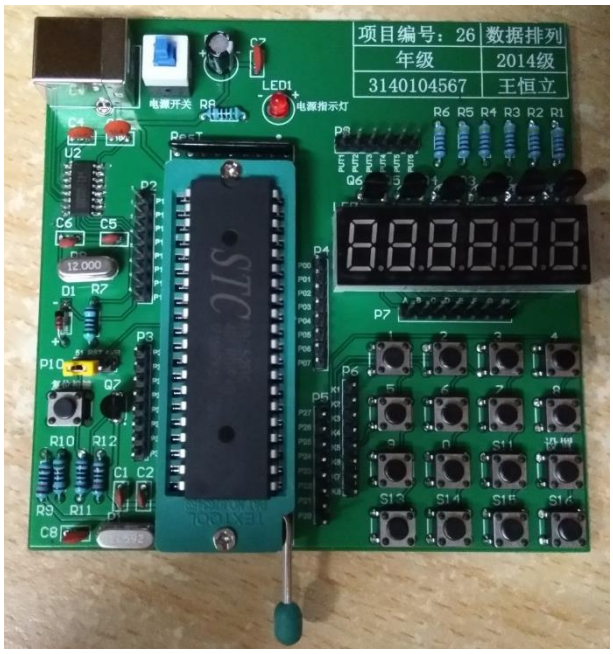
3. 当输入 10 个数据后，观察是否能正常进入数据输出阶段以及数据输出阶段是否符合要求，发现可以正常进入数据输出阶段，把输入的数据按照从小到大的顺序以 2 秒为周期依次显示出来。

根据以上测试，可以知道最终结果满足了课程设计题目的所有要求，故成功。

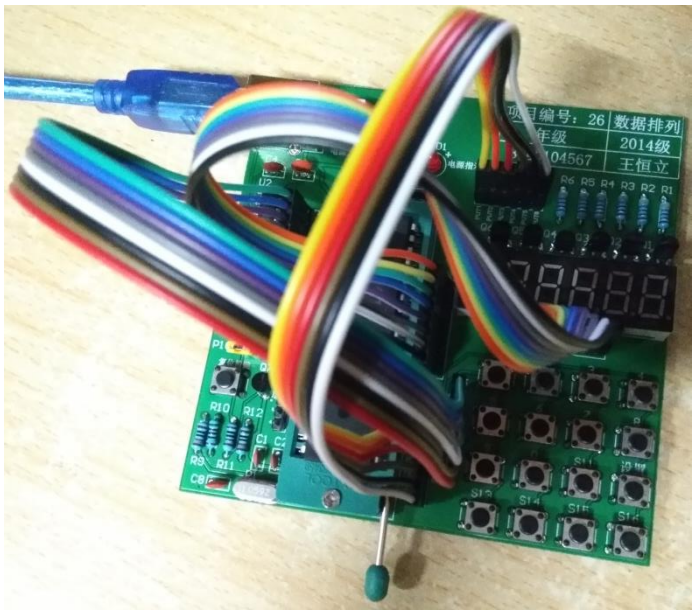
5.3 测试结果和数据记录

通过测试能够发现系统的功能正常，并能够满足题目的所有要求。

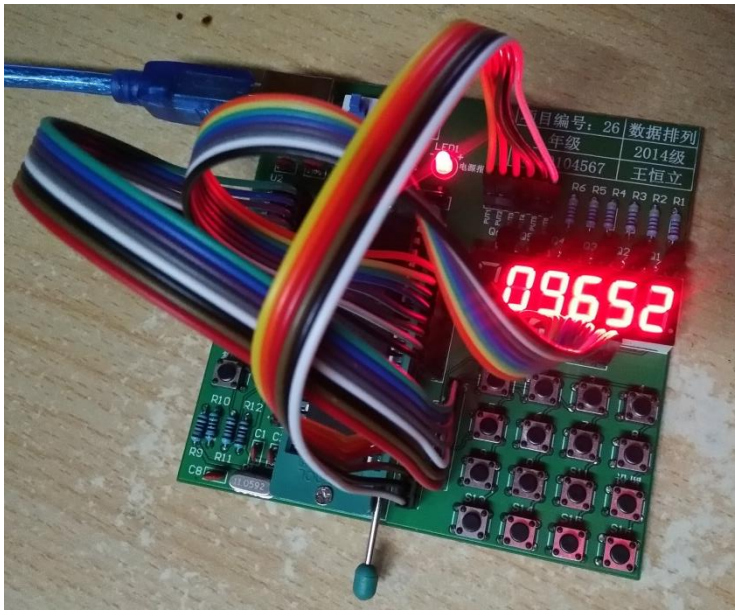
下图为电路板焊接完成后的图片。



下图为电路板按照程序连接好杜邦线后的图片。



下图为数据输出阶段的照片，该状态表示按照从小到大的顺序输出的第 9 个数据为 652。



六、 课程设计总结

在本次课程设计中，我学会了利用 Altium Designer 软件来进行电路板原理图的设计以及 PCB 板的绘制，我相信这在今后的实习和工作中是有很大作用的。此外，通过自己绘制电路，我对单片机的外围电路有了更深刻的了解，并学习了电路中各个元件所起到的作用。由于实际的电子元器件种类繁多，AD 中的元件库并没有我所需要的元件及其封装，我还通过查阅资料学习了如何根据元器件的封装尺寸自己绘制其原理图和封装图，从而完成了课程设计，这对我

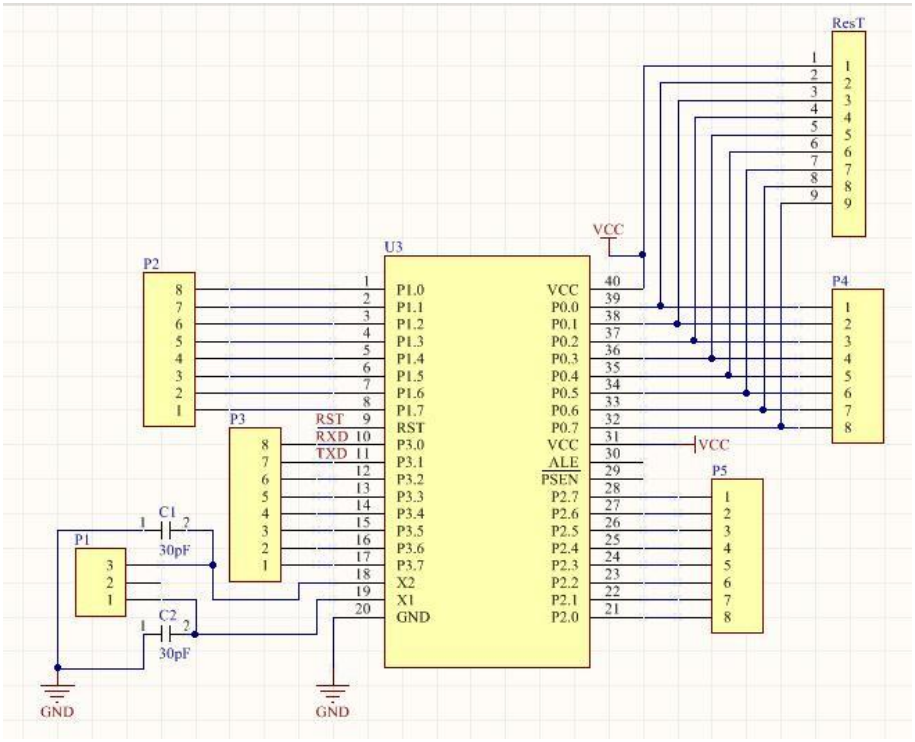
今后自己使用 AD 软件有了很大的提升。另外，通过本次设计也让我明白了设计时要十分的细心，不懂的地方要查资料，不能想当然的就做决定，这次数码管选择错误就是一个很大的教训。相信我在之后的设计中会更加的细心，并且也学会了如何处理这一类错误。

此外，本次课程设计也锻炼了我焊接的能力，使我对于焊接能够更加的熟练，而不是刚开始那样的手足无措。我遵从了从低到高的焊接原则，依次将贴片、电阻等元件进行焊接。其中最难焊接的还是贴片，很容易将两个管脚焊接在一起，或者由于芯片过热而导致芯片损坏。不过在课程设计中，我发现自己的焊接能力还是比较过硬的，几次焊接的结果都没有什么问题。

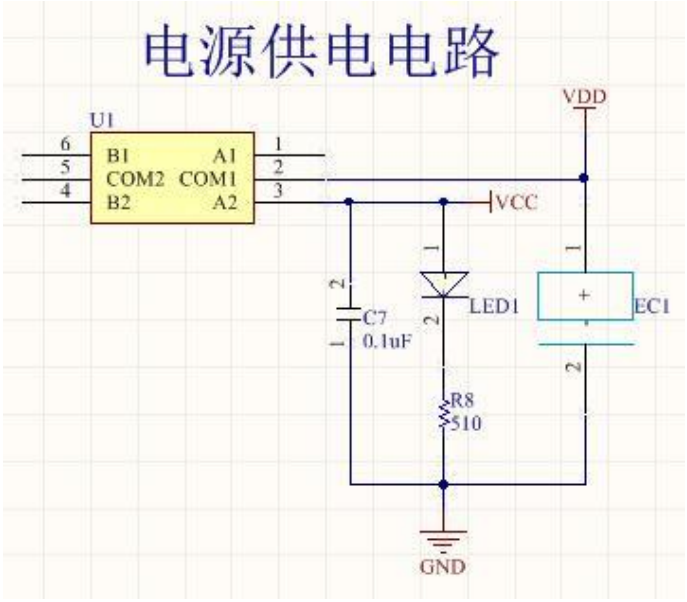
总而言之，这次课程设计使我在 PCB 设计、焊接、软件编程等方面都得到了很大的锻炼，这也是今后我们会经常使用到的技能。此外，我在这里提一点对课程的建议，希望老师能够提前安排时间，尽量不要将焊接、调试过程安排到考试周。

七、 附件

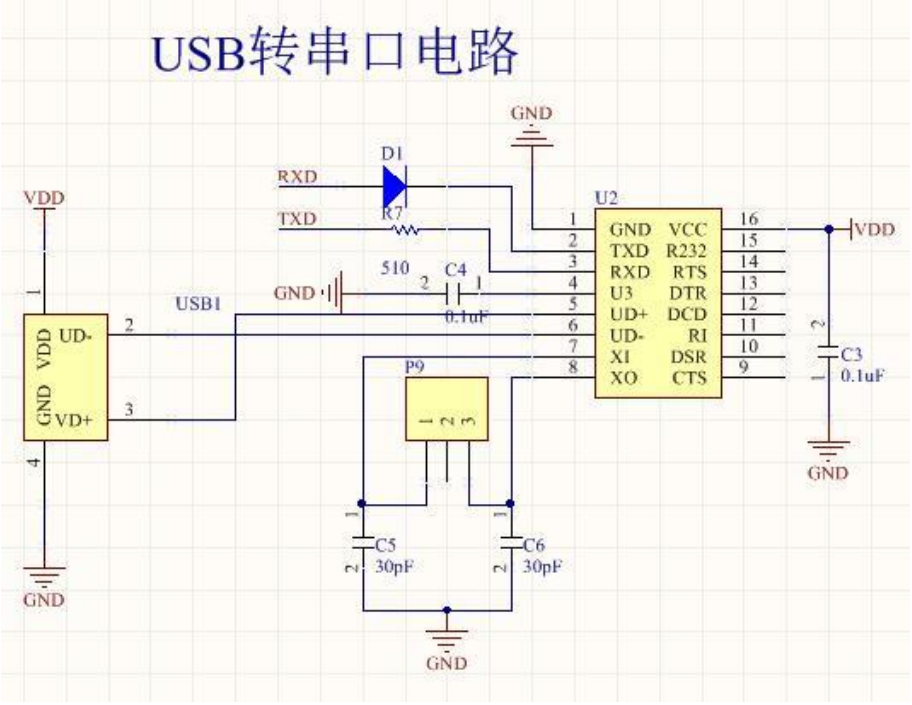
附件 1 硬件系统原理图



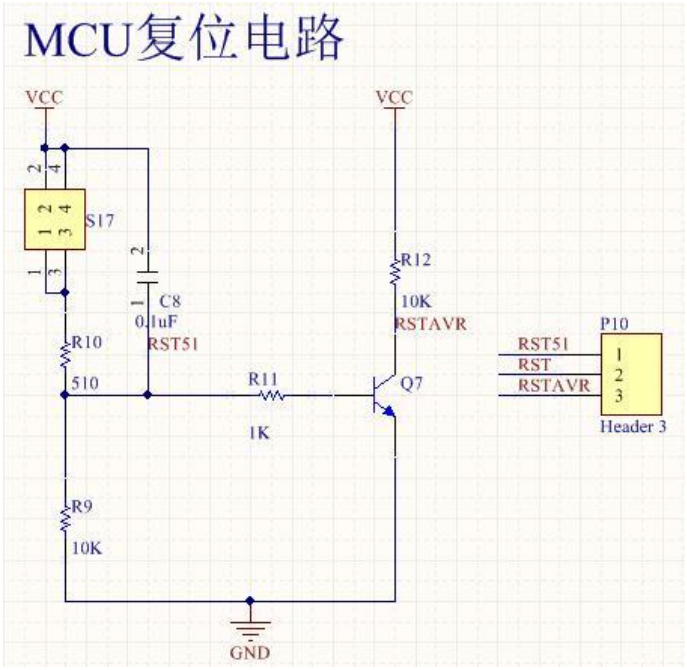
单片机模块电路原理图



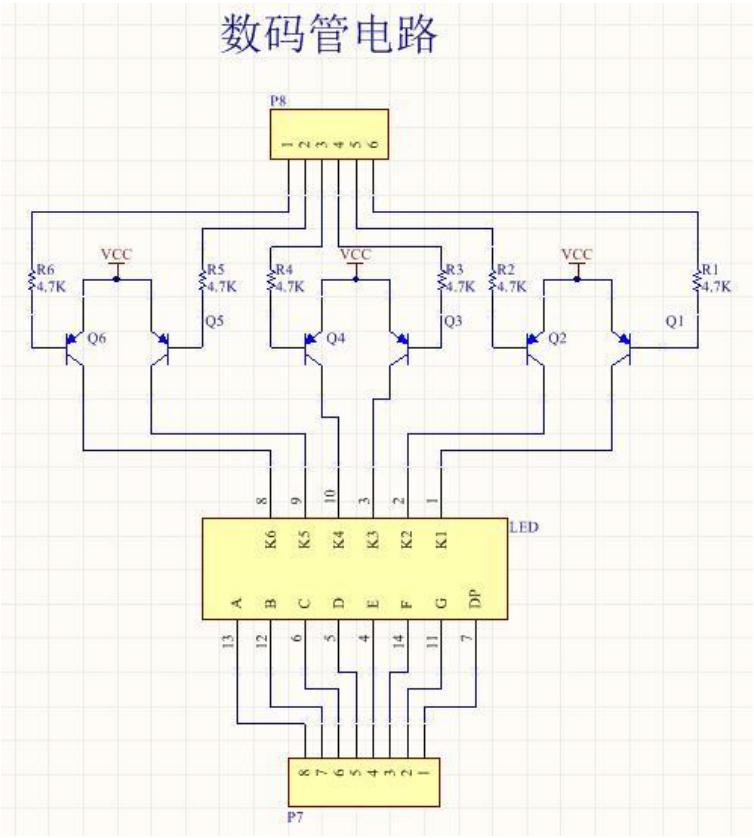
电源供电电路原理图



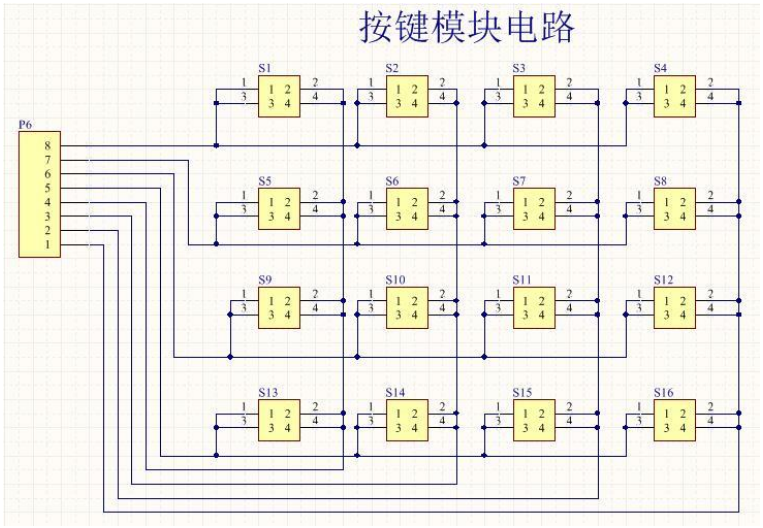
USB 转串口电路原理图



MCU 复位电路原理图



数码管电路原理图



按键模块电路原理图

附件 2 程序及注释

```
/*********************************微机原理与接口课程设计 ******************************
* 项目名称 ： 数据排列
* 项目编号 ： 26
* 班　　级 ： 机电 1401 班
* 姓　　名 ： 王恒立
* 学　　号 ： 3140104567
*********************************************************************************************/
#include<reg51.h>

//IO 接口定义
#define LED_PORT P0        //数码管接口
#define KEY_PORT P1        //矩阵键盘接口
#define LEDC_PORT P2       //数码管使能端接口

unsigned char code decode[10] = {0xc0,0xf9,0xa4,0xb0,0x99,0x92,0x82,0xf8,0x80,0x90};    //数码管显示编码
unsigned char code control[5] = {0xef,0xf7,0xfb,0xfd,0xfe};                  //数码管使能端编码
unsigned char tempdata[5] = {0xc0,0xc0,0xc0,0xc0,0xc0};                      //存储数码管显示数据
unsigned int numdata[10];                 //存储输入的 10 个数据

unsigned char KeyScan(void);              //实现矩阵键盘扫描
void delayms(unsigned int t);             //实现 ms 级的延时
void display(void);                       //实现数码管动态扫描显示
void displaynum(void);                    //实现把输入的数据按照从小到大的顺序以 2 秒为周期依次显示出来
void sort(void);                          //实现输入 10 个数据的排序
```

```
void cleardata(void);                    //实现清空数据

//全局变量定义
unsigned int count1,count2,num,tnum1,tnum2,tnum3,tnum4,tnum5,flag,flags;

int main (void)
{
    flag = 0;                //标志是否进入输入数据阶段
    flags = 0;               //标志是否进入输出数据阶段
    count1 = 0;              //记录输入数据个数
    count2 = 0;              //记录输入单个数据位数
    while(1)
    {
        if(flags == 0)
            display();
        if((num = KeyScan()) == 12) //当按下设置键后，进入循环
        {
            flag = 1;        //标志进入输入数据阶段
            cleardata();
        }
        while(flag == 1)     //进入输入数据阶段
        {
            display();
            while((num = KeyScan()) != 0xff) //矩阵键盘有按键按下时进入循环
            {
                //输入数据存储
                if(count2 == 0)
                {
                    tnum3 = num;
                    tnum4 = 0;
                    tnum5 = 0;
                }
                else if(count2 == 1)
                    tnum4 = num;
                else
                    tnum5 = num;

                if(tnum3 == 10)
                    tnum3 = 0;
                if(tnum4 == 10)
                    tnum4 = 0;
                if(tnum5 == 10)
                    tnum5 = 0;
```

```
                count2++;
                if(count2 == 3)
                {
                    numdata[count1] = 100*tnum3+10*tnum4+tnum5;
                    count2 = 0;
                    count1++;}

                tnum1 = count1/10;
                tnum2 = count1%10;

                tempdata[0] = decode[tnum1];
                tempdata[1] = decode[tnum2];
                tempdata[2] = decode[tnum3];
                tempdata[3] = decode[tnum4];
                tempdata[4] = decode[tnum5];
                display();

                if(count1 == 10)    //输入数据结束后，跳出循环
                {
                    unsigned int i;
                    for(i = 0;i < 80;i++)
                        display();
                    flags = 1;
                    flag = 0;
                    break;
                }
            }
        }

        while(flags == 1)          //进入输出数据阶段
        {
            sort();             //对数据进行排序
            displaynum();//把输入的数据按从小到大顺序以 2 秒为周期依次显示出来
            flags = 0;          //标志输出数据阶段结束
            cleardata();        //清空数据
        }
    }
}

void cleardata(void)            //实现清空数据
{
    //全局变量置零
    num = 0;
    count1 = 0;
```

```
    count2 = 0;
    tnum1 = 0;
    tnum2 = 0;
    tnum3 = 0;
    tnum4 = 0;
    tnum5 = 0;
    tempdata[0] = decode[0];
    tempdata[1] = decode[0];
    tempdata[2] = decode[0];
    tempdata[3] = decode[0];
    tempdata[4] = decode[0];
}

void delayms(unsigned int t)        //实现 ms 级的延时
{
    unsigned int i,j;
    for(i = t;i > 0;i--)
        for(j = 110;j > 0;j--);
}

void display(void)                  //实现数码管动态扫描显示
{
    //每调用一次 display()函数，耗时 25ms
    unsigned int i;
    for(i = 0;i < 5;i++)
    {
        LED_PORT = tempdata[i];
        LEDC_PORT = control[i];
        delayms(5);
    }
}

void displaynum(void)   //实现把输入数据按从小到大顺序以 2 秒为周期依次显示出来
{
    unsigned int i,n,t,count;
    t = 2000;           //每个数据显示 2 秒
    n = t/25;
    count = 1;          //记录显示数据个数
    while(count<=10)
    {
        //对数码管显示数据数组赋值
        tempdata[0] = decode[count/10];
        tempdata[1] = decode[count%10];
        tempdata[2] = decode[numdata[count-1]/100];
```

```
            tempdata[3] = decode[numdata[count-1]%100/10];
            tempdata[4] = decode[numdata[count-1]%100%10];

            //数码管动态扫描显示数据
            for(i = 0;i < n;i++)
                display();
            count++;
        }
}

void sort(void)             //实现输入 10 个数据的排序
{
    //采用选择排序的方法
    unsigned i,k,temp,index;
    for(k = 0;k < 9;k++)
    {
        index = k;
        for(i = k+1;i < 10;i++)
            if(numdata[i]<numdata[index])
                index = i;
        temp = numdata[index];
        numdata[index] = numdata[k];
        numdata[k] = temp;
    }
}

unsigned char KeyScan(void)     //实现矩阵键盘扫描
{
    unsigned char temp,num;
    num = 0xff;                 //若没有按键按下，则返回 0xff
    KEY_PORT = 0xfe;
    temp = KEY_PORT;
    temp = temp&0xf0;
    while(temp != 0xf0)         //若有按键 1、2、3、4 按下，则进入循环
    {
        delayms(5);             //延时消抖
        temp = KEY_PORT;
        temp = temp&0xf0;
        while(temp != 0xf0)     //若有按键 1、2、3、4 按下，则进入循环
        {
            temp = KEY_PORT;
            switch(temp)        //判断键值
            {
                case 0xee:num = 1;
```

```
                    break;
                case 0xde:num = 2;
                    break;
                case 0xbe:num = 3;
                    break;
                case 0x7e:num = 4;
                    break;
            }
            while(temp != 0xf0)      //判断按键是否松开
            {
                temp = KEY_PORT;
                temp = temp&0xf0;
            }
        }
    }

    KEY_PORT = 0xfd;
    temp = KEY_PORT;
    temp = temp&0xf0;
    while(temp != 0xf0)              //若有按键 5、6、7、8 按下，则进入循环
    {
        delayms(5);                  //延时消抖
        temp = KEY_PORT;
        temp = temp&0xf0;
        while(temp != 0xf0)          //若有按键 5、6、7、8 按下，则进入循环
        {
            temp = KEY_PORT;
            switch(temp)             //判断键值
            {
                case 0xed:num = 5;
                    break;
                case 0xdd:num = 6;
                    break;
                case 0xbd:num = 7;
                    break;
                case 0x7d:num = 8;
                    break;
            }
            while(temp != 0xf0)      //判断按键是否松开
            {
                temp = KEY_PORT;
                temp = temp&0xf0;
            }
        }
```

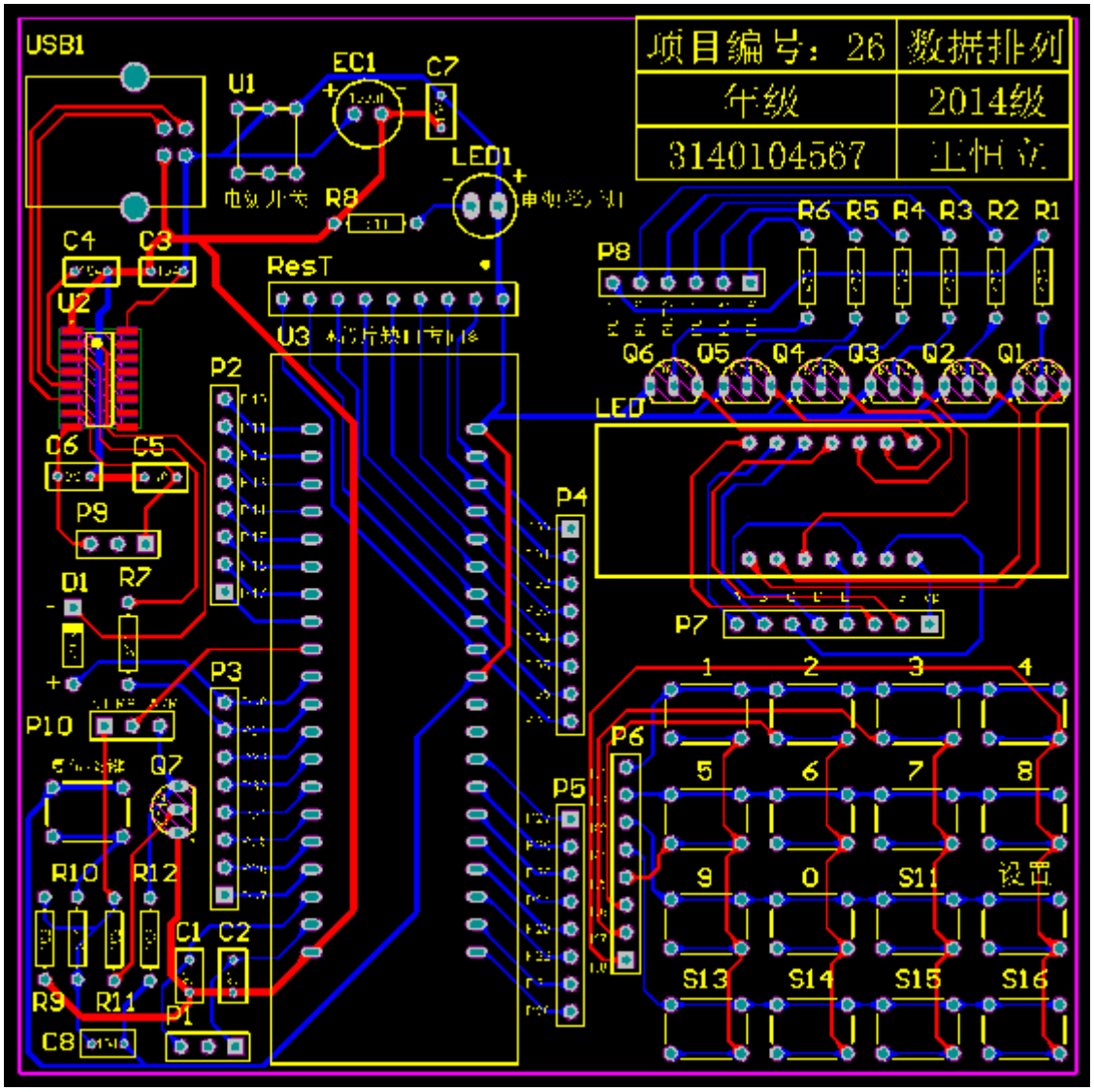
```
    }

    KEY_PORT = 0xfb;
    temp = KEY_PORT;
    temp = temp&0xf0;
    while(temp != 0xf0)              //若有按键 9、10、11、12 按下，则进入循环
    {
        delayms(5);                  //延时消抖
        temp = KEY_PORT;
        temp = temp&0xf0;
        while(temp != 0xf0)          //若有按键 9、10、11、12 按下，则进入循环
        {
            temp = KEY_PORT;
        switch(temp)                 //判断键值
            {
                case 0xeb:num = 9;
                    break;
                case 0xdb:num = 10;
                    break;
                case 0xbb:num = 11;
                    break;
                case 0x7b:num = 12;
                    break;
            }
        while(temp != 0xf0)          //判断按键是否松开
            {
                temp = KEY_PORT;
                temp = temp&0xf0;
            }
        }
    }

    KEY_PORT = 0xf7;
    temp = KEY_PORT;
    temp = temp&0xf0;
    while(temp != 0xf0)              //若有按键 13、14、15、16 按下，则进入循环
    {
        delayms(5);                  //延时消抖
        temp = KEY_PORT;
        temp = temp&0xf0;
        while(temp != 0xf0)          //若有按键 13、14、15、16 按下，则进入循环
        {
            temp = KEY_PORT;
        switch(temp)                 //判断键值
```

```
            {
                case 0xe7:num = 13;
                    break;
                case 0xd7:num = 14;
                    break;
                case 0xb7:num = 15;
                    break;
                case 0x77:num = 16;
                    break;
            }
        while(temp != 0xf0)             //判断按键是否松开
            {
                temp = KEY_PORT;
                temp = temp&0xf0;
            }
        }
    }
    return num;                         //返回键值
}
```

附件 3 PCB 图


项目编号：26
数据排列
年级
2014级
3140104567
USB1
U1
EC1
C7
LED1
R8
C4
C3
ResT
U2
U3
P2
P8
R6 R5 R4 R3 R2 R1
Q6 Q5 Q4 Q3 Q2 Q1
LED
C6
C5
P9
P4
D1
R7
P7
P3
P10
P6
Q7
P5
R10
R12
C1
C2
R9
R11
P1
C8
1 2 3 4
5 6 7 8
9 0 S11 设置
S13 S14 S15 S16