

**中国地质大学（武汉）自动化学院**

**电路综合实习报告**

二🌕二一年七月二十日

目录

[第一部分：实验箱系统实习——车辆状况监测系统 1](#_Toc5529)

[1. 内容概述 1](#_Toc23660)

[1.1任务要求 1](#_Toc18547)

[1.2设计理念 2](#_Toc5071)

[1.3功能实现 2](#_Toc20630)

[1.4系统特色 2](#_Toc5661)

[2.硬件模块 3](#_Toc2207)

[2.1系统设计 4](#_Toc968)

[2.2直流电机 4](#_Toc1652)

[2.3 继电器 5](#_Toc7527)

[2.4温度传感器 5](#_Toc18775)

[2.5压力传感器 6](#_Toc18974)

[2.6矩阵键盘及八段数码管 6](#_Toc30008)

[2.7LCD模块 7](#_Toc31239)

[2.8 16×16点阵 8](#_Toc28146)

[2.9蜂鸣器 9](#_Toc3354)

[2.10 A/D转换模块 10](#_Toc31001)

[2.11 D/A转换模块 10](#_Toc27177)

[2.12 8255端口扩展模块 11](#_Toc892)

[3. 软件设计 12](#_Toc3820)

[3.1系统框图 12](#_Toc20370)

[3.2 电机模块 12](#_Toc26926)

[3.3按键以及数码管显示 14](#_Toc15460)

[3.4 LCD显示 17](#_Toc2709)

[3.5点阵、蜂鸣器及LED 20](#_Toc4277)

[3.6速度档位及油箱余量模拟 23](#_Toc5035)

[4. 实习心得 24](#_Toc21621)

[4.1遇到问题及解决 24](#_Toc22532)

[4.2收获与感想 24](#_Toc21186)

[第二部分：硬件板系统实习——车辆状况监测系统 25](#_Toc3253)

[1.内容概述 25](#_Toc28935)

[1.1任务要求 25](#_Toc7039)

[1.2实现功能 26](#_Toc11256)

[1.3系统特色 26](#_Toc18524)

[2. 硬件模块 26](#_Toc20855)

[2.1矩阵键盘 27](#_Toc1093)

[2.2单色数码管 28](#_Toc1707)

[2.3 单色点阵 28](#_Toc24437)

[3. 软件设计 29](#_Toc25556)

[3.1系统框图 29](#_Toc24178)

[3.2按键扫描 29](#_Toc31639)

[3.3单色点阵显示 31](#_Toc20637)

[3.4数码管显示开车时间 33](#_Toc4571)

[3.5其他模块 35](#_Toc31778)

[4. 实习心得 36](#_Toc9844)

[4.1遇到问题及解决 36](#_Toc21114)

[4.2收获感想 36](#_Toc23078)

[实习意见建议 37](#_Toc14809)

# 第一部分：实验箱系统实习——车辆状况监测系统

## 内容概述

### 1.1任务要求

在本部分的实习内容中，主要是利用伟福9000实验箱上的资源尽量完成自己个人设想的内容，如有需要可以自行增加外设。

最终作品接线及本部分实习所用外设如下图：

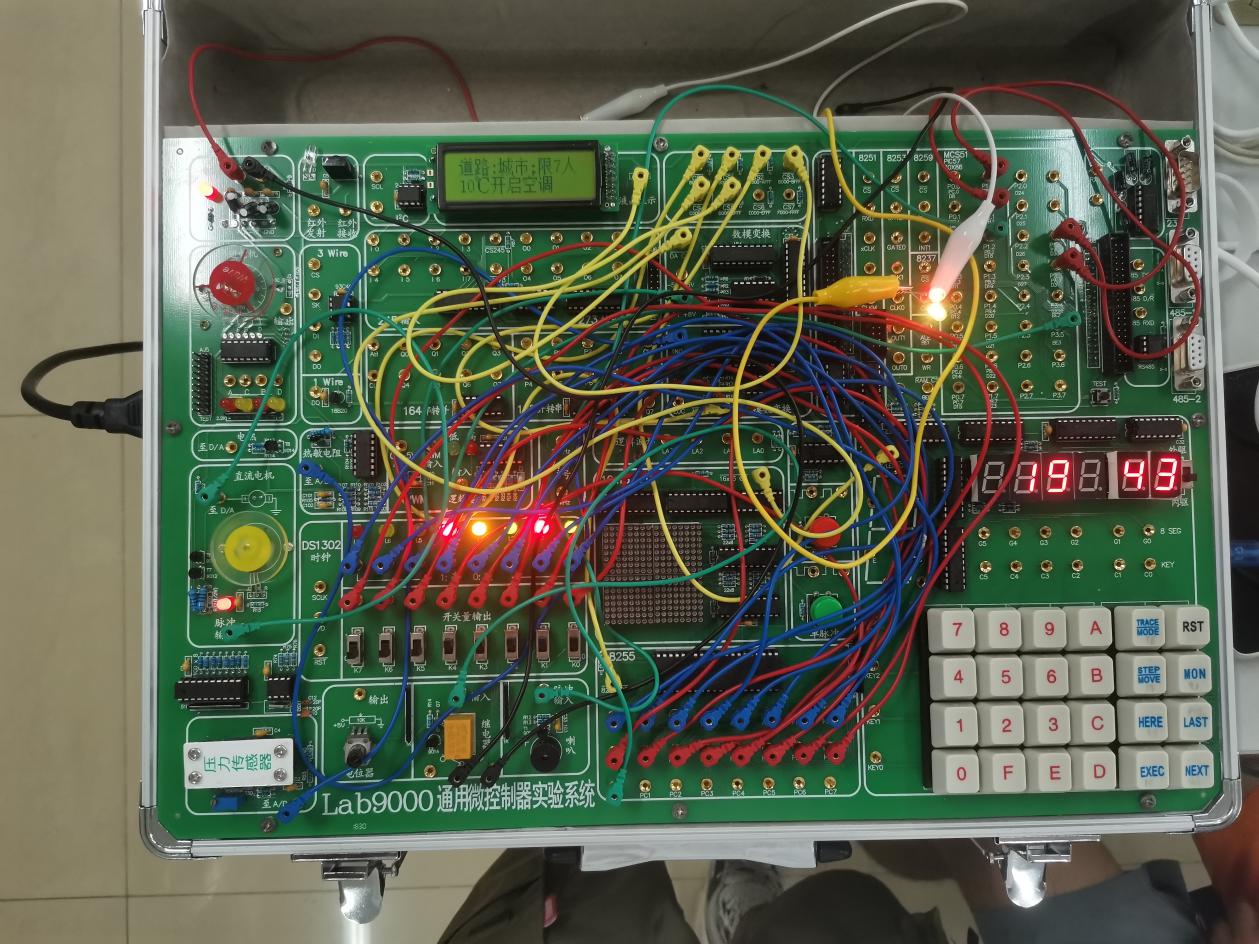


图 1实验箱接线图

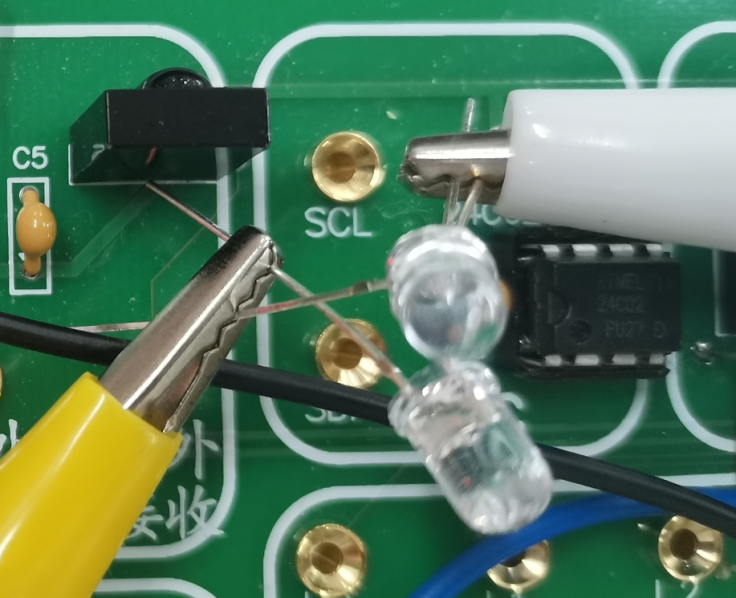


图 2外设LED灯

### 1.2设计理念

考虑到驾驶员在驾驶车辆时需要实时了解车辆的状况，故构思了一个车辆状况监测系统。本系统支持实时监测车辆载重、当前车速和车内温度，能实时显示车速和车内温度。当车内温度过高可自动开启空调；当出现超载、超速、缺油等异常情况可自动报警。本系统的宗旨是提升车辆的安全性与舒适度，为提高用户体验，道路类型、限重和空调开启温度均可根据不同情况自行选择，适用范围广，自动化程度高，操作简单，方便维护。

### 1.3功能实现

1)液晶屏（道路类型、载重人数、空调开启温度的显示）；

2)点阵（异常报警显示）；

3)矩阵键盘（用于改变道路类型、载重人数、空调开启温度）；

4)八段数码管（显示速度、温度）；

5)直流电机+霍尔传感器（模拟车辆行驶以及获取车辆行驶速度）；

6)压力传感器（监测是否超载）；

7)温度传感器（模拟天气温度）；

8)继电器+外设LED（模拟空调的开关）；

9)8255+拨码开关+LED灯（模拟油量剩余油量以及模拟当前速度档位）

10)蜂鸣器（异常报警）；

11)+5V电源模块（给外接LED供电）  
12)A/D转换模块（将压力传感器和温度传感器的模拟量转换为数字量）

13)D/A转换模块（将数字量转换为模拟量以实现控制电机的转速）

### 1.4系统特色

1)使用了霍尔传感器，实现速度的实时监测；

2)使用了8255+拨码开关+LED灯模拟油量剩余油量以及模拟当前速度档位，显示直观且便于操作；

3)用继电器控制空调的开关；

4)使用八段数码管实时显示当前速度和温度；

5)限重、道路模式和空调开启温度均有多种选择，系统开机后可随时更改，使本系统适用范围广，使用灵活；

6)LCD状态显示仅在按键时刷新；

7)采用了大量的宏定义，方便修改各类传感器的临界阈值。

## 2.硬件模块

74HC138是本次实验中必不可少的模块，对A、B、C口译码得到8个地址互异的接口，通过\*CS0~\*CS7分别接上各模块锁存器的锁存端，实现对各个模块总线方式控制。

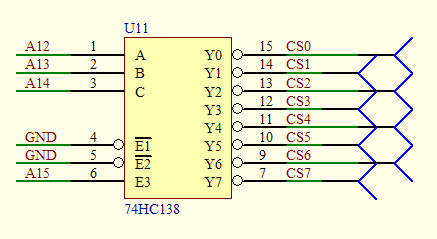


图 3 74HC138原理图

### 2.1系统设计

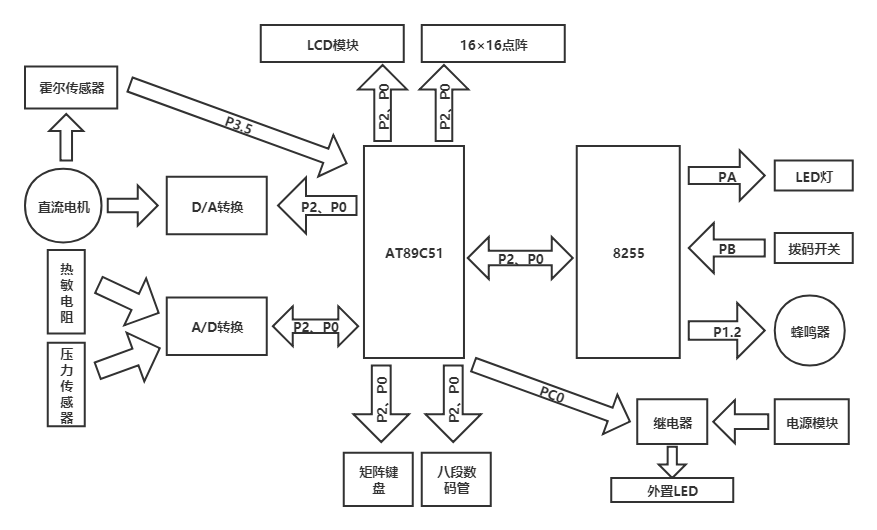


图 4 系统框图

本系统使用了伟福9000实验箱中的13个模块，系统框图如上图所示。以下为各类模块的原理以及实现。

### 2.2直流电机

D/A转换电路用于控制直流电机，将数字量转换为模拟量从而控制直流电机的转速。D/A转换电路可以输出-8V~+8V的电压，其中0x80输出0V，0xff输出+8V，0x00输出-8V。电机有启动最小电压，正转反转的最小电压对应的十六进制数分别为0xc0、0x40。

直流电机转盘上有一小磁体，磁体经过霍尔元件时会在测量电路中产生一个脉冲，当电机连续转动则会产生周期性的脉冲。使用T0定时，T1计数的方式在单位时间内对脉冲进行计数，从而实现电机测速。

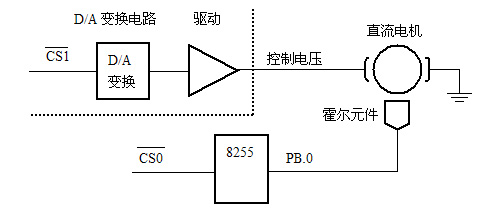


图 5直流电机工作原理图

### 2.3 继电器

8255的PC0口连接继电器的输入口，通过改变PC0口的值可以控制三极管的导通和截止，从而使继电器工作在常开或常闭状态，实现对电气电路执行元件的开关控制。

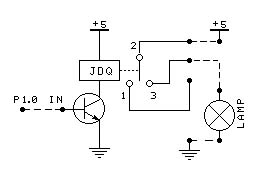


图 6继电器工作原理图

### 2.4温度传感器

热敏电阻由半导体陶瓷材料制成，随着当外界温度变化时，电阻值也会发生相应的线性变化。大多数热敏电阻为负温度系数，即当外界温度升高，电阻阻值会下降。通过测量电路和A/D转换模块可以测得电阻变化的阻值大小，从而实现测温。

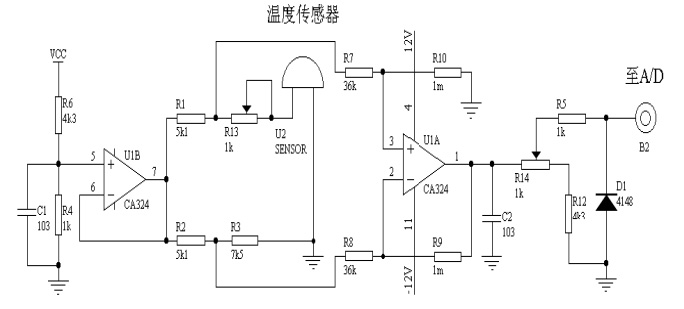


图 7温度传感器原理图

### 2.5压力传感器

压力传感器由基体材料和应变片组成，应变片黏贴在基体材料上，当基体材料发生形变，应变片随之发生形变，从而导致电阻值发生变化，俗称电阻应变效应。

通过A/D转换模块可将压力传感器测得的模拟信号转换为数字信号，从而可以获取当前压力的大小。

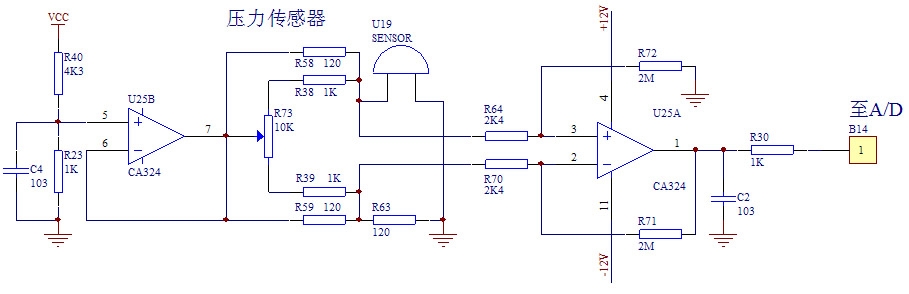


图 8压力传感器原理图

### 2.6矩阵键盘及八段数码管

**列扫描行读入原理：**

行线外接VCC，在没有按键的情况下，始终为高电平。扫描时列线逐列给低电平，当按键按下，行线与列线相交，对应行线的电平变为低电平；根据当前同时为低电平的行线值和列线值可以得到哪个按键被按下。

**六位八段数码管的显示原理：**

利用人体的视觉暂留原理，六个八段数码管分时选通快速刷新，同时74374给被选通的数码管送段码使数码管点亮。

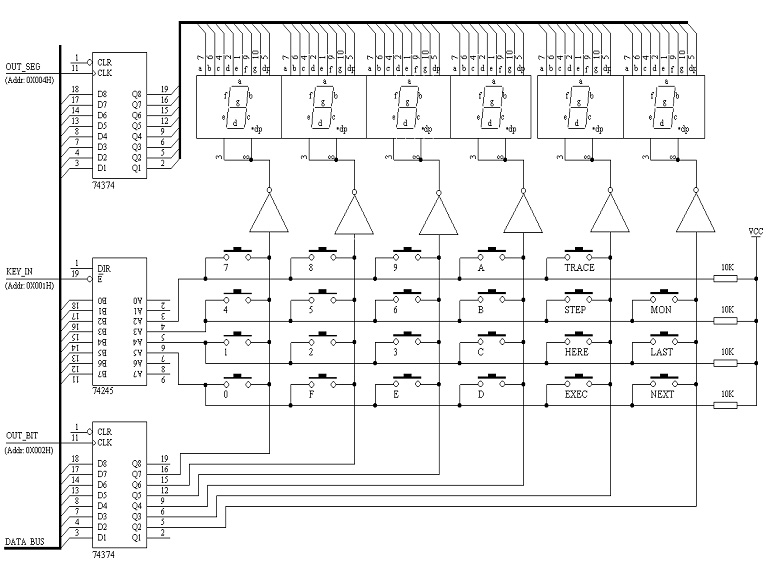


图 9矩阵键盘原理图

### 2.7LCD模块

本实验采用的液晶显示屏内置控制器为SED1520，点阵为122x32，需要两片SED1520控制，由E1、E2分别选通，以控制显示屏的左右两半屏。图形液晶显示模块有两种连接方式。一种为直接访问方式，一种为间接控制方式。本实验采用直接控制方式。

直接控制方式就是将液晶显示模块的接口作为存储器或 I／O 设备直接挂在计算机总线上。计算机通过地址译码控制E1和E2的选通；读／写操作信号\*RD／\*WR由地址线A1控制；命令/数据寄存器选择信号AO由地址线A0控制。实际电路如下图所示。

地址映射如下。

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0X000H | 0X001H | 0X002H | 0X003H | 0X004H | 0X005H | 0X006H | 0X007H |
| 写E1指令 | 写E1数据 | 读E1状态 | 读E1数据 | 写E2指令 | 写E2数据 | 读E2状态 | 读E2数据 |

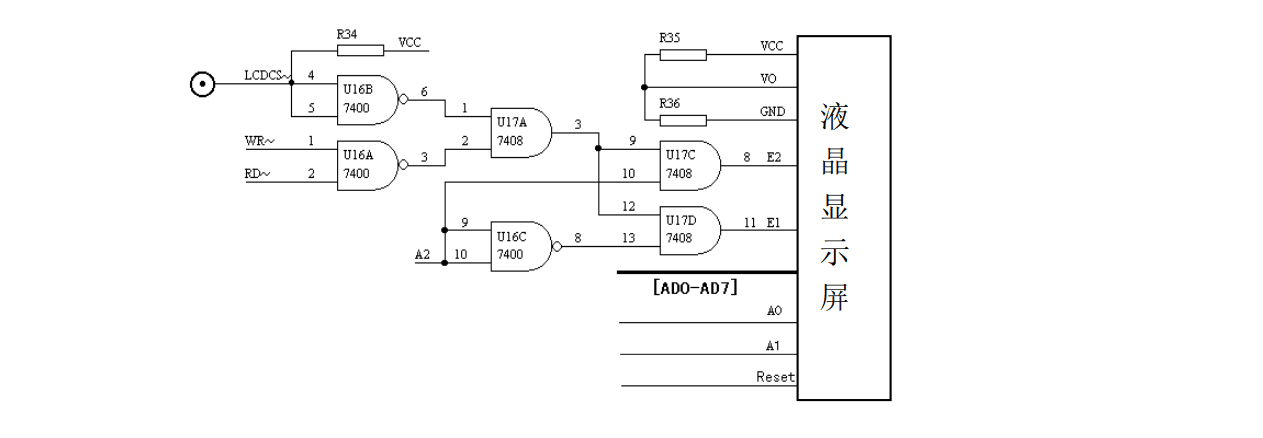


图 10 LCD原理图

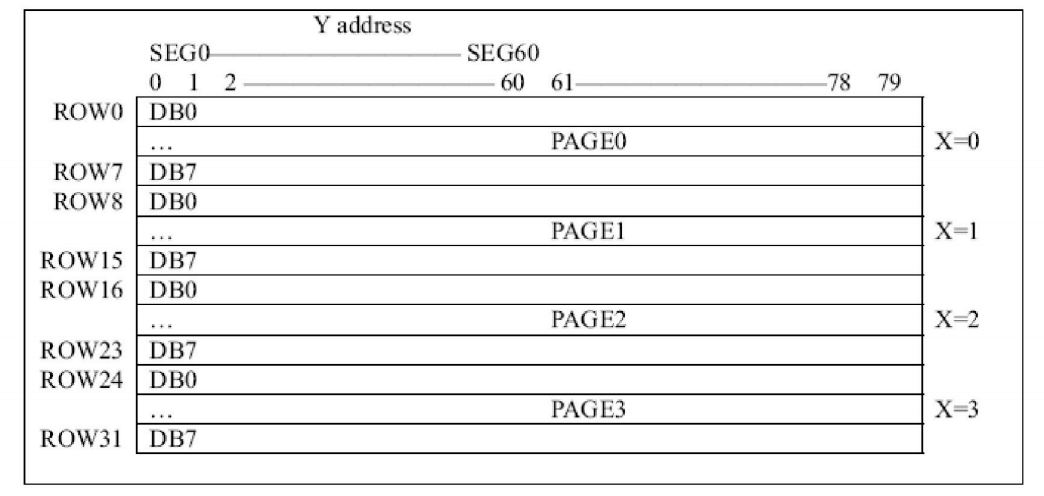


图 11 LCD工作原理图

### 2.8 16×16点阵

16×16点阵共需要16个列驱动及16个行驱动，用到了4片74HC574锁存列信号和行信号。每个行与每个列可以选中一个发光管，共有256个发光管，采用动态驱动方式，每次显示一行，10ms后再显示下一行。

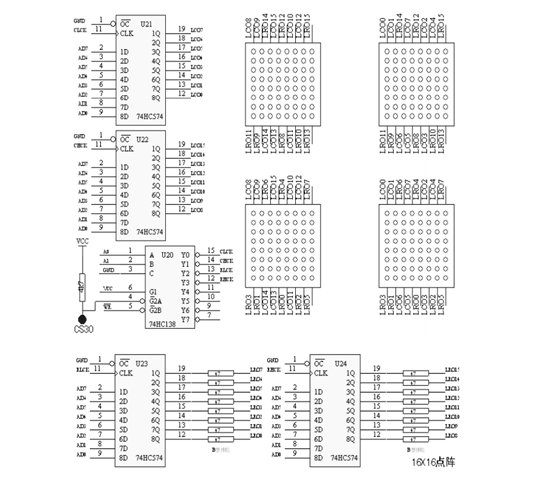


图 12点阵工作原理图

### 2.9蜂鸣器

蜂鸣器模块由一个NPN型BJT对输入端口的信号进行放大，再由电容器滤波后送给蜂鸣器。由于该蜂鸣器是无源蜂鸣器，故需要给脉冲才能鸣响，脉冲的频率不同，发出的音高也不同。运用此原理可以将蜂鸣器做成一个音乐播放器。

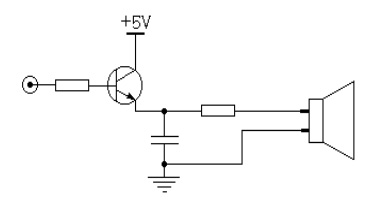


图 13蜂鸣器原理图

### 2.10 A/D转换模块

ADC0809是一款8位逐次逼近式A/D模数转换器。其内部有一个8通道多路开关，它可以根据地址码锁存译码后的信号，只选通8路模拟输入信号中的一个进行A/D转换。模拟输入电压范围0～+5V，转换时间为100μs(时钟为640KHz时)，130μs（时钟为500KHz时）。由于硬件原因，在实验箱上只能使用IN0、IN1两个信号输入。

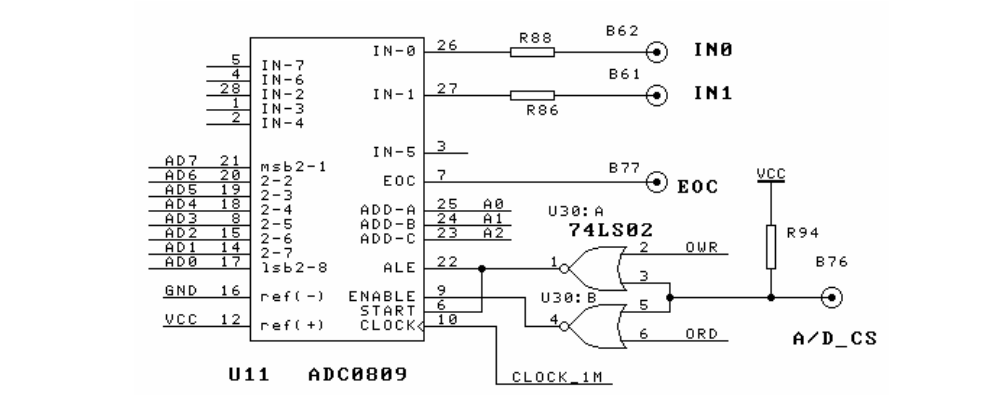


图 14ADC0809原理图

### 2.11 D/A转换模块

DAC0832是8位的D/A转换集成芯片。由8位输入锁存器、8位DAC寄存器、8位D/A转换电路及转换控制电路构成。具有双缓冲、单缓冲和直通三种输入方式。

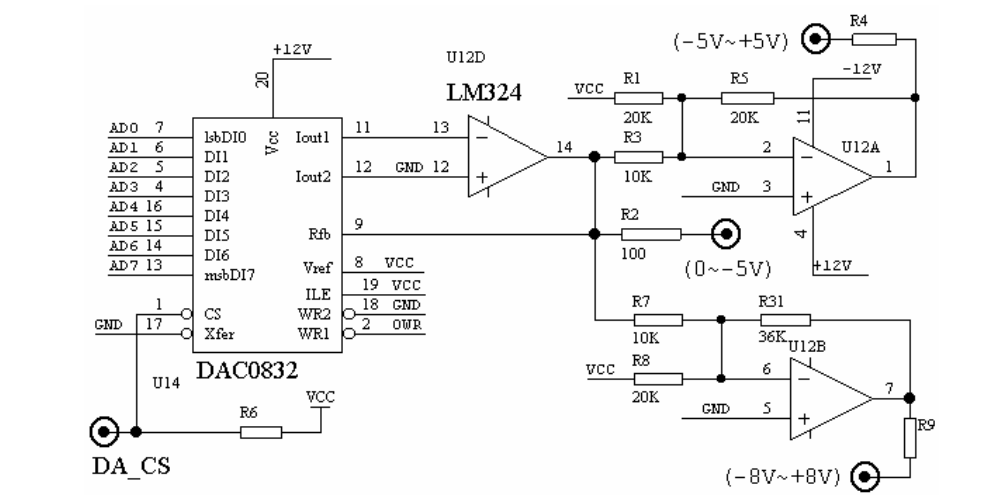


图 15DAC0832原理图

### 2.12 8255端口扩展模块

8255是一款可编程并行I/O接口芯片，有3个8位并行I/O口，共40个引脚。 通过控制字可以设置PA、PB和PC口的工作方式，使用灵活，通用性强。

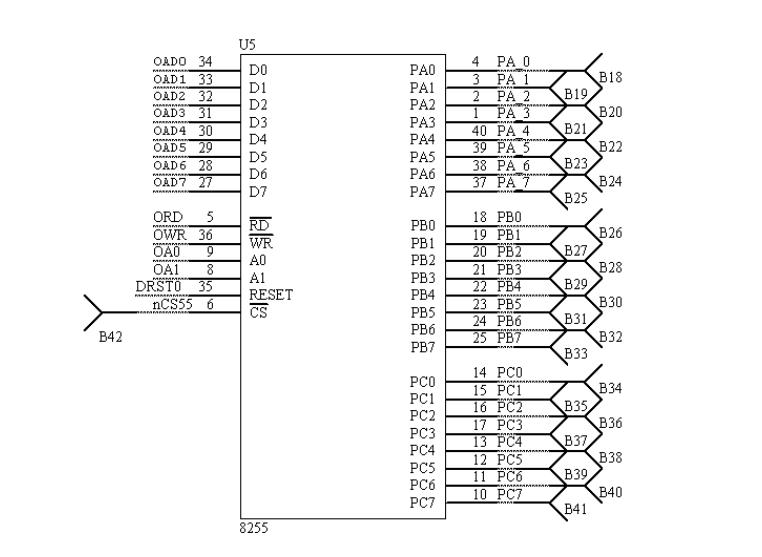


图 16 8255原理图

## 软件设计

### 3.1系统框图

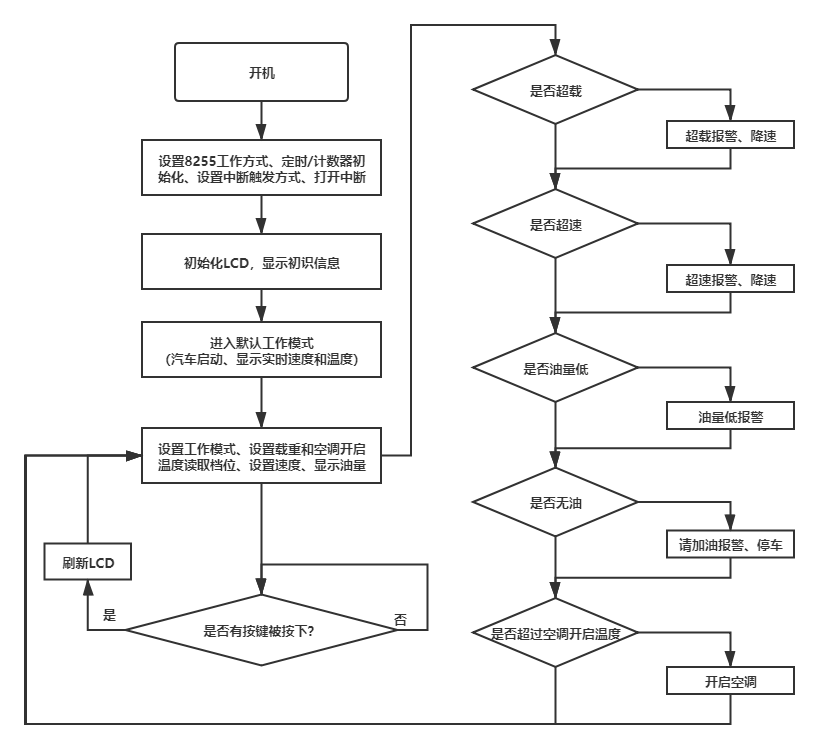


图 17 系统主流程图

### 3.2 电机模块

直流电机主要用于模拟车辆行驶状态。使用DAC0809将数字量转换为模拟量实现对电机转速的控制，DAC0809的IN0地址为0xd000；使用T0、T1定时/计数器实现电机测速，T0工作在定时方式，T1工作在计数方式；T0定时结束自动进入中断，每进入中断60次读取一次T1计数器的值，从而实现3秒的长定时。



图 18测速模块

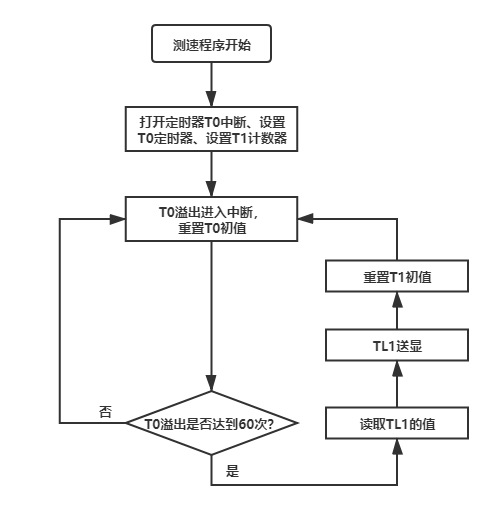


图 19测速程序流程图

void readSpeed()

{

int i=0;

TMOD = 0x51; // T0定时、T1计数

TR0 = 0;//关闭T0

TH0 = 0x3c;//设置初值，50000次溢出

TL0 = 0xb0;

TR1 = 0;//设置计数器T1

TH1 = 0;

TL1 = 0;

TR0 = 1;//打开定时器

TR1 = 1;//打开计数器

}

void INTT1() interrupt 2 //外部中断1

{

//需要关闭定时和计数、并将speed置0

TR0 = 0;//关闭T0

TH0 = 0x3c;//设置初值，50000次溢出

TL0 = 0xb0;

TR1 = 0;//设置计数器T1

TH1 = 0;

TL1 = 0;

speed=TL1;

TT=60;

//以上为关闭定时器和计数器

MOTOR=0x80;

if(oil()==2){//显示请加油

DZOO();

}

if(overspeed==1){//当速度超过设定值，进入中断0

DZS();

overspeed=0;

}

PortA=0x00;//PA口置0

MOTOR=SetSpeed();

P1\_1=1;//关闭中断

//重新打开定时器和计数器

TR0 = 1;//打开定时器

TR1 = 1;//打开计数器

}

每次进入中断后需要重新给计数初值，每次读取T1计数器的值后需要将计数器的值置为0，由此可以实现循环测速。

### 3.3按键以及数码管显示

按键主要用于道路模式的选择、空调开启温度的增减和载重人数的选择。0和F用于加减空调开启温度，A、B、C分别用于载重5、7、9人的选择，1、2、3分别用于城市、公路、高速三种道路模式的选择。八段数码管前三位实时显示当前时间，后两位实时显示当前速度。

unsigned char GetKey()

{

unsigned char Pos;

unsigned char i;

unsigned char k;

i = 6;

Pos = 0x20; // 找出键所在列

do {

OUTBIT = ~ Pos;

Pos >>= 1;

k = ~IN & 0x0f;

} while ((--i != 0) && (k == 0));

// 键值 = 列 x 4 + 行

if (k != 0) {

i \*= 4;

if (k & 2)

i += 1;

else if (k & 4)

i += 2;

else if (k & 8)

i += 3;

OUTBIT = 0;

do Delay(10); while (TestKey()); // 等键释放

return(KeyTable[i]); // 取出键码

} else return(0xff);

}

主要思路：行线外接VCC，在没有按键的情况下，始终为高电平。扫描时，列线逐列给低电平，当按键按下，行线与列线相交，对应行线的电平变为低电平；根据当前为低电平的行线值和列线值可以得到哪个按键被按下。

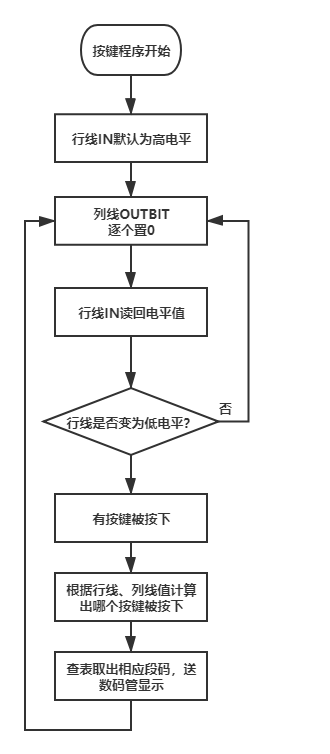


图 20按键扫描流程图

void DisplayS()//速度显示函数

{

unsigned int i=0,sum=0;

i=(speed>>4) & 0x0f;

if(i>9) sum=16\*(i-0x0a+10);

else sum=16\*i;

i=speed & 0x0f;

if(i>9) sum+=i-0x0a+10;

else sum+=i;

LEDBuf[4] = LEDMAP[sum/10];

LEDBuf[5] = LEDMAP[sum%10];

}

void DisplayT()//温度显示函数

{

signed char T;

T = CurTemp;

if (T < 0) {

//如果温度低于零度,显示'-'号

LEDBuf[0] = 0x40;

T = - T;

} else LEDBuf[0] = 0;

//显示温度十位数

LEDBuf[1] = LEDMAP[T / 10];

//显示温度个位数

LEDBuf[2] = LEDMAP[T % 10];

LEDBuf[3] = 0;

}

数码管的前3位显示当前温度，数码管的后2位显示当前速度。将速度和温度显示函数放入main()函数的while()循环中，便可实现温度和速度的实时显示。

### 3.4 LCD显示

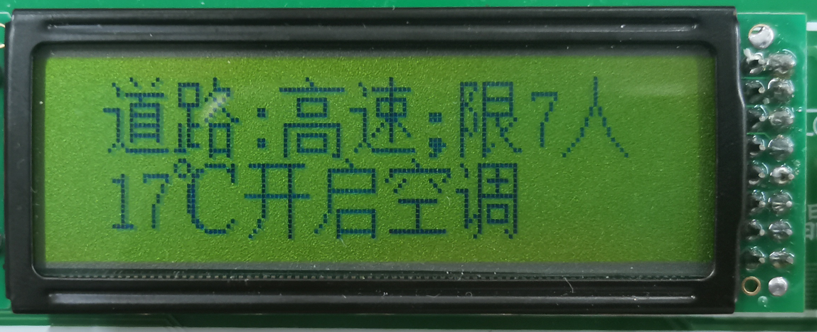


图 21 LCD显示效果图

LCD主要用于显示当前的道路类型、限载人数和空调开启温度。由于液晶的显示需要刷新，如果实时刷新会占用单片机大量的运行时间，导致按键响应慢、八段数码管闪烁等问题。另外，我们发现LCD刷新过快时可能会出现显示模糊的现象，故我们决定采用按键时刷新的方式。

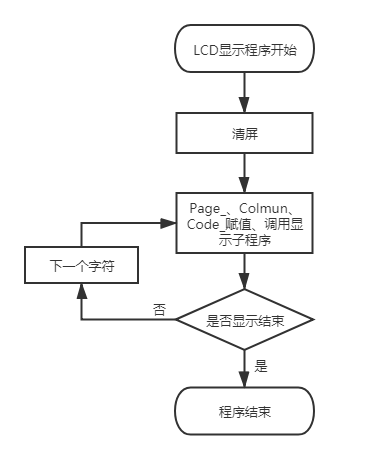


图 22LCD显示流程图

void Setmoshi()//设置模式：道路、标载、开启温度

{

if(TestKey())

{

switch (GetKey()& 0x0f) // (GetKey()& 0x0f)的值就等于按键的数值

{

case 1: {initMotor=0x45;DisplayLcd();break;} //可容许右起1档,按1

case 2: {initMotor=0x54;DisplayLcd();break;} //可容许右起2档,按2

case 3: {initMotor=0x59;DisplayLcd();break;} //可容许所有档,按3

case 0x0a: {initYALI=0x07;DisplayLcd();break;}//载重5人,按a

case 0x0b: {initYALI=0x15;DisplayLcd();break;}//载重7人,按b

case 0x0c: {initYALI=0x25;DisplayLcd();break;}//载重9人,按c

case 0x00: {--initTEMP;DisplayLcd();break;} //下调空调开启温度,按0

case 0x0f: {++initTEMP;DisplayLcd();break;} //上调空调开启温度,按F

} //1档40，2档55，3档75

}

}

以上为按键选择程序，每当有按键按下，LCD自动刷新一次。没有按键按下时LCD不刷新。

void DisplayLcd()

{

unsigned int A=0x03;

Init(); //初始化 (移到主函数)

Clear(); //清屏

//----------------------以下是道路部分-----------------------------------

Page\_ = 0x02; //（上排 ）

Column = 0x00;//列位置

Code\_ = 0x00;

WriteCHN16x16(); //“道”

Page\_ = 0x02;

Column = 0x10;

Code\_ = 0x01;

WriteCHN16x16(); //“路”

Page\_ = 0x02;

Column = 0x20;

Code\_ = 0x0a;

WriteNUM8x16(); //“：”

Page\_ = 0x02;

Column = 0x28;

switch (initMotor)

{

case 0x45: Code\_ = 0x02;break; //1档，“城”

case 0x54: Code\_ = 0x03;break; //2档，“公”

case 0x59: Code\_ = 0x04;break; //3档，“高”

}

WriteCHN16x16(); //“城”、“公”、“高”

Page\_ = 0x02;

Column = 0x38;

switch (initMotor)

{

case 0x45: Code\_ = 0x05;break; //1档，“市”

case 0x54: Code\_ = 0x06;break; //2档，“路”

case 0x59: Code\_ = 0x07;break; //3档，“速”

}

WriteCHN16x16(); //“市”、“路”、“速”

Page\_ = 0x02;

Column = 0x48;

Code\_ = 0x0b;

WriteNUM8x16(); //“;”}

以上为显示当前道路的程序，显示空调开启温度和载重人数的程序由该程序重载得来，故不再展示。

### 3.5点阵、蜂鸣器及LED

当系统进入超速、油量低、缺油、超载等状况时，蜂鸣器发出警报的同时点阵会显示当前车辆异常情况的提示语句。在超速时，蜂鸣器报警、LED闪烁、点阵显示“已超速”、电机自动降速；在超载时，蜂鸣器报警、LED闪烁、点阵显示“已超载”、电机自动降速；油箱剩余一格时，蜂鸣器报警、LED闪烁、点阵显示“油量低”、电机保持当前转速；当油箱耗尽，蜂鸣器报警、LED闪烁、点阵显示“请加油”、电机自动停转。

设置中断为低电平触发，当出现异常状况，通过I/O口置为低电平的方式进入中断函数。进入中断后，重新设置定时器和计数器，以便于出中断后重新开始测速；出中断前，将I/O口重新置为高电平。

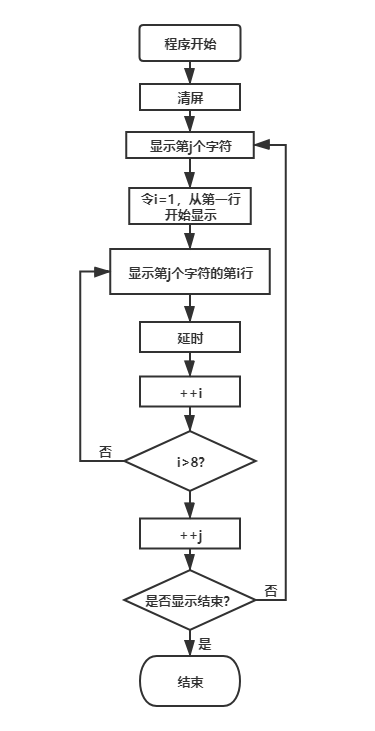


图 23点阵刷新流程图

void INTT0() interrupt 0 //外部中断0

{

//需要关闭定时和计数、并将speed置0

TR0 = 0;//关闭T0

TH0 = 0x3c;//设置初值，50000次溢出

TL0 = 0xb0;

TR1 = 0;//设置计数器T1

TH1 = 0;

TL1 = 0;

speed=TL1;

TT=60;

//以上为关闭定时器和计数器

MOTOR=0xc5;

PortA=0xff;

if(oil()==0){//显示油量低

DZO();

}

if(ADResult0>initYALI){//显示已超载

DZZ();

}

PortA=0x00;//PA口置0

P1\_0=1;//关闭中断

//重新打开定时器和计数器

TR0 = 1;//打开定时器

TR1 = 1;//打开计数器

}

以下为点阵和蜂鸣器结合的程序代码。将8255的PA口取反可实现LED灯闪烁；蜂鸣器需要输出持续的脉冲才能发出警报，点阵也需要逐行刷新，在此将点阵刷新的频率作为蜂鸣器脉冲的频率，以此实现蜂鸣器和点阵同时工作。

void BEE(){//蜂鸣器报警

int i;

PortA=~PortA;

for(i=10;i!=0;--i){//LED闪烁、蜂鸣器bee

BEEP=0;

Delay1();

BEEP=1;

Delay1();

}

}

void DZS()//显示已超速

{

uchar i,j;

uchar count;

uint bitmask;

//清屏

ColLow = 0xff; //行驱动低有效

ColHigh= 0xff;

RowLow = 0x00; //列驱动高有效

RowHigh= 0x00;

for(j=3; j<6; j++){ //j:第几个。逐个显示所有数字

for(count =0; count <50; count ++){ //每个字显示50次

bitmask = 0x01;

BEE();//加入蜂鸣器报警函数

for(i=0;i<16;i++){//把第j个数逐行扫描

RowLow = 0x00; //首先清屏

RowHigh = 0x00;

ColLow = ~ CHNFont[j][i\*2 ]; //写出一行数据（Col送数据）//

ColHigh = ~ CHNFont[j][i\*2+1];

RowLow = bitmask & 0xff; //点亮此行显示一半屏 （Row控制点亮）

RowHigh = bitmask >> 8; //\*\*显示另一半屏\*\*

bitmask <<= 1; //左移位,指向下一行

delayDZ(1);

}

}

ColLow = 0xff;

ColHigh = 0xff;

}

}

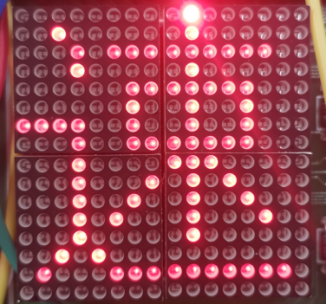
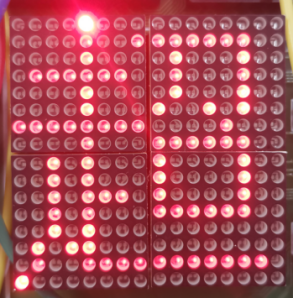
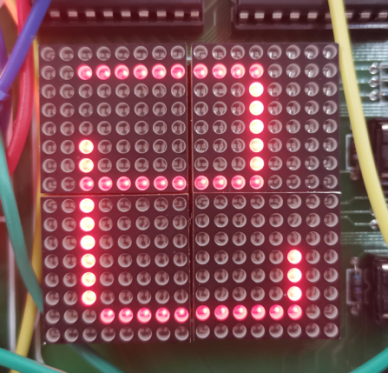


图 24点阵显示效果图

请加油、油量低、已超载的显示函数与已超速的显示函数结构完全相同，故不再展示。

### 3.6速度档位及油箱余量模拟

拨码开关和LED灯的高四位代表速度档位，不同档位对应不同速度，从左到右依次递减；拨码开关和LED灯的低四位代表油箱余量，从左到右依次递减。

unsigned char SetSpeed(){//档位选择函数

unsigned char S,R;

S=PortB;//开关的高4位设置速度

if(S>=0x00&&S<=0x0f)R=0xc5;

if(S>=0x10&&S<=0x1f)R=0xcf;

if(S>=0x20&&S<=0x3f)R=0xdf;

if(S>=0x40&&S<=0x7f)R=0xef;

if(S>=0x80&&S<=0xff)R=0xff;

return R;

}

int oil(){

unsigned char S,Z;

int R;

Z=PortB;

S = (Z & 0x0f);

if(S==0x01){

R=0;//油量低

}

else if(S==0x00){

R=2;//没油

}

else R=1;

return R;}

设计思路：通过8255的PB口读取拨码开关的状态并通过PA口实时显示；通过对PB口的读取获得一个数字量，对数字量的高4位和低4位分别进行判断可以获得当前的速度档位和当前的油箱余量。

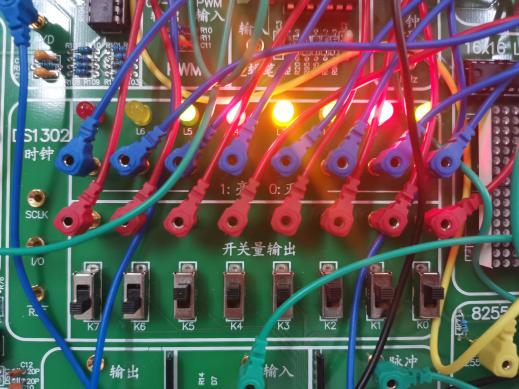


图 25拨码开关及LED效果图

## 实习心得

### 4.1遇到问题及解决

**1)直流电机速度的获取** 在例程中，直流电机通过定时器T0计时，需要等待低电平和高电平结束，这会占用单片机大量的运行时间，导致按键响应不灵敏，八段数码管频繁闪烁等问题。想要解决这个问题，我们需要改变测速的方式，通过询问老师，我们采用T0定时、T1计数的方式进行测速。

具体实现：设置T0工作在方式1初值为0x3cb0，T1工作在方式1初值为0x0000。T0每溢出60次读取一次T1计数器的值，T1计数器的值/T0定时的时间(3秒)即为电机转速。

**2)蜂鸣器和点阵不能同时工作**  
 16×16点阵需要不断的刷新，蜂鸣器需要输入脉冲才能发出警报，两者不能同时工作。

解决方法：将蜂鸣器报警函数加入点阵显示函数中，用点阵刷新的频率作为蜂鸣器脉冲的频率可以实现两者同时工作。

### 4.2收获与感想

在本部分的实习中，从最开始的设想到最后的系统实现，其实是一个从0到1的过程。在这个过程中，我的编程能力得到了提高，也锻炼了从网络中查找资源的能力。在这个过程中我掌握了模块化编程，学会了如何合理的分配单片机资源。

在实习刚开始时，可以说是无从下手，心里也比较着急，但是着急也不是办法，于是我和队友沉下心来研究实验箱上的各个模块并用例程测试。慢慢的我们发现，实现我们设想的功能好像也没有想象中的这么难。后来我们渐入佳境，最终基本实现了我们设想的功能。通过这次实习，我发现很多事情想想都是困难，但是做做都是答案，不要一开始就被困难吓倒，一定要积极的动手去做。

# 第二部分：硬件板系统实习——车辆状况监测系统

## 1.内容概述

### 1.1任务要求

在本部分实习中，主要任务是焊接硬件实验板并尽可能的将实验箱上的功能在硬件实验板上实现。

硬件实验板成品如下：

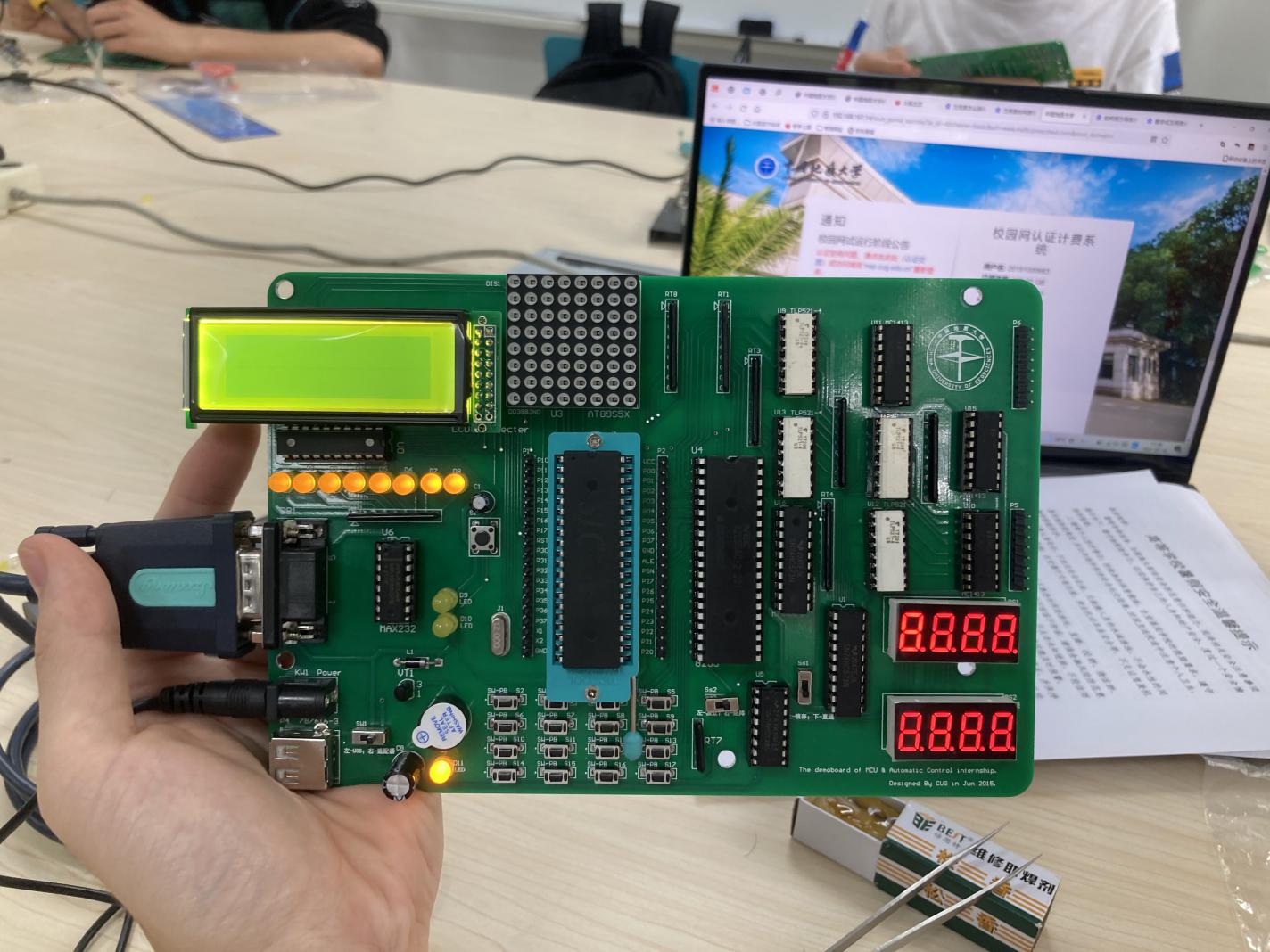


图 26硬件板正面

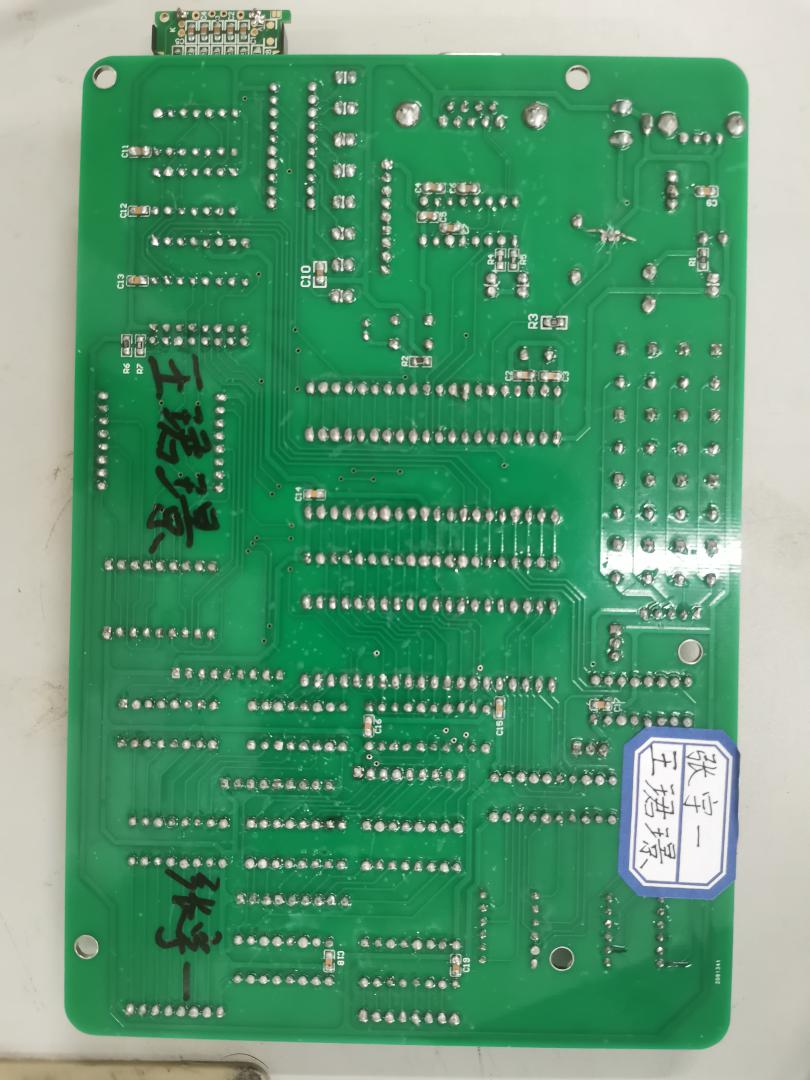


图 27硬件板背面

### 1.2实现功能

由于硬件板上没有直流电机、压力传感器、热敏电阻等模块，我们采用按键代替热敏电阻和直流电机，当按键按下则进入报警程序。最终实现的功能有：

1. LCD开机显示，显示作者信息、系统名称；
2. LCD显示当前道路模式、空调开启温度和载重人数；
3. 按键选择道路模式、增减空调开启温度、选择载重人数或进入报警状态；
4. 报警时LED闪烁、蜂鸣器发出警报、8×8点阵显示警报对应的图形；
5. 八段数码管记录行驶时间。

### 1.3系统特色

1. 使用八段数码管显示开车时间，相比实验箱更加人性化；
2. 点阵和键盘读取的程序由自己编写，实用性更强。其中点阵程序采用行控制点亮、列送字模数据的方式，支持“文字图形取模”软件取模，设置成“横向取模”和“字节倒序”即可，而该软件支持自行抠图后取模，适用范围更广，实用性更强；
3. 8×8点阵显示警报对应的图形，显示更加直观。

## 硬件模块

硬件板中各个模块的地址在设计之初已被确定，需要通过查询PCB原理图的方式来获取每个模块对应的地址。由于3-8译码器、8255、LCD和蜂鸣器的原理已经在实验箱实习部分进行了概述，在此不再重复叙述，而是对一些与实验箱有区别的资源进行简要说明。

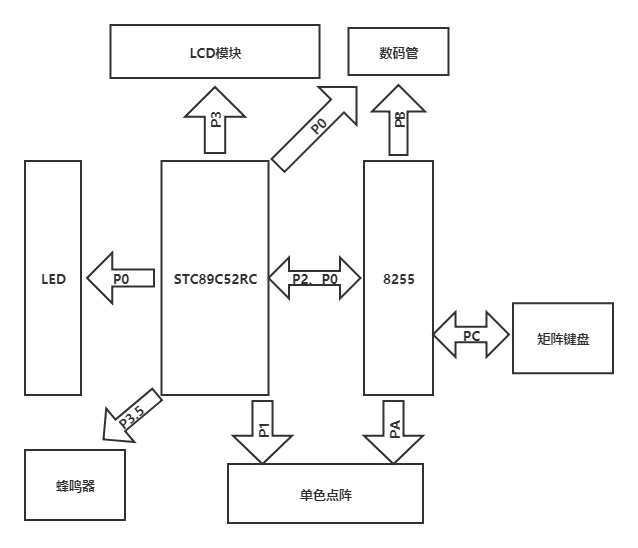


图 28硬件板系统框图

### 2.1矩阵键盘

硬件板上的矩阵键盘，输入和输出均由8255的PC端口完成，行线接VCC高电平，需要通过列扫描行读回的方式获取按键信息；扫描原理在实验箱部分已有详细叙述，故在此不再重复。

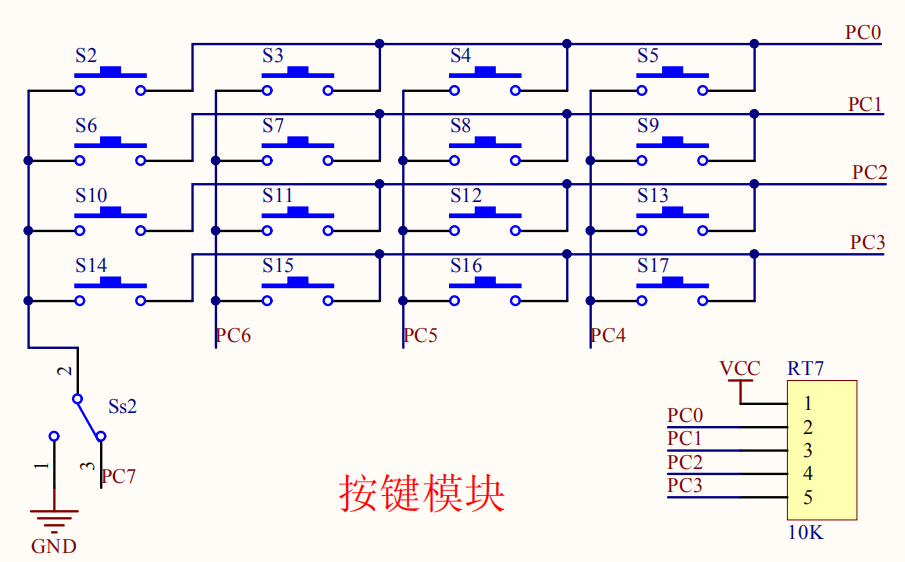


图 29 4×4矩阵键盘原理图

### 2.2单色数码管

单色数码管的工作原理在实验箱部分已有详细叙述，在此仅简要说明硬件板上单色数码管的控制方法。在硬件板中，使用P0口和8255的PB口控制两个4位的八段数码管；Y0~Y7作为位选信号，分时选通8位数码管。

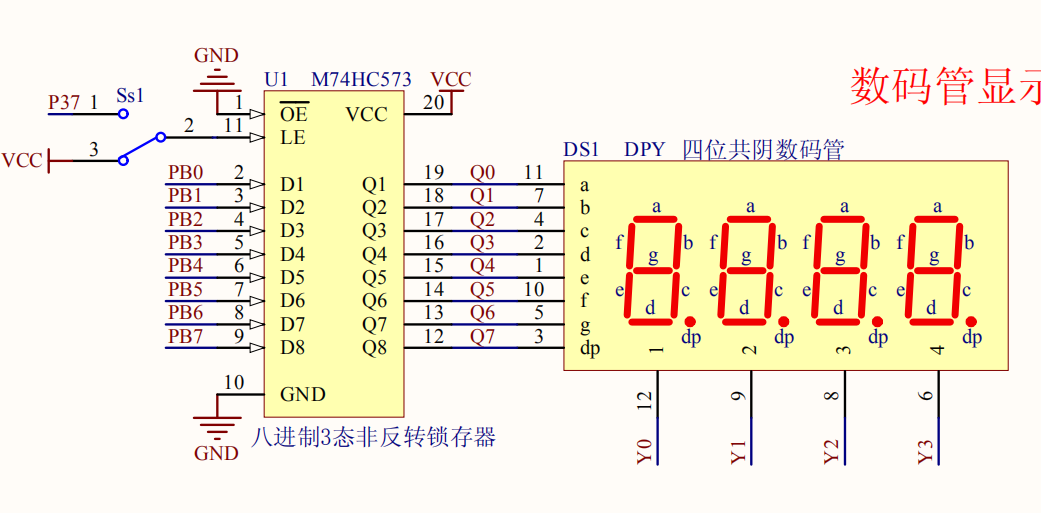


图 30硬件板数码管原理图

### 2.3 单色点阵

由原理图我们可以看出，点阵由P1口和8255的PA口共同控制，需要将PA设置为输出模式。

单色点阵低电平有效，只有某个LED所在的行和列同时给低电平时，它才会亮。故当需要点亮特定位置的点阵时，需要将对应的PA和P1置低。例如，行和列全给低时，64个LED全亮；行全给低，列部分高部分低，结果是给低的几列亮，其余几列灭。

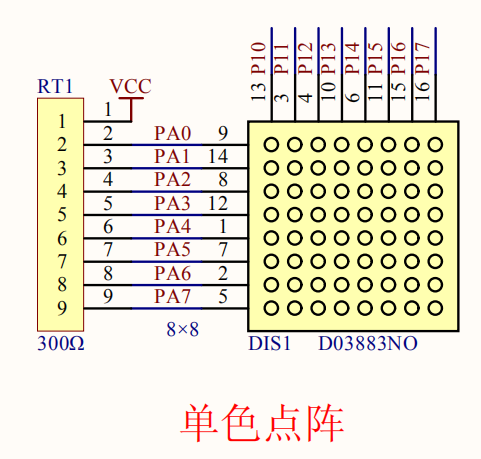


图 31 单色点阵原理图

## 软件设计

### 3.1系统框图

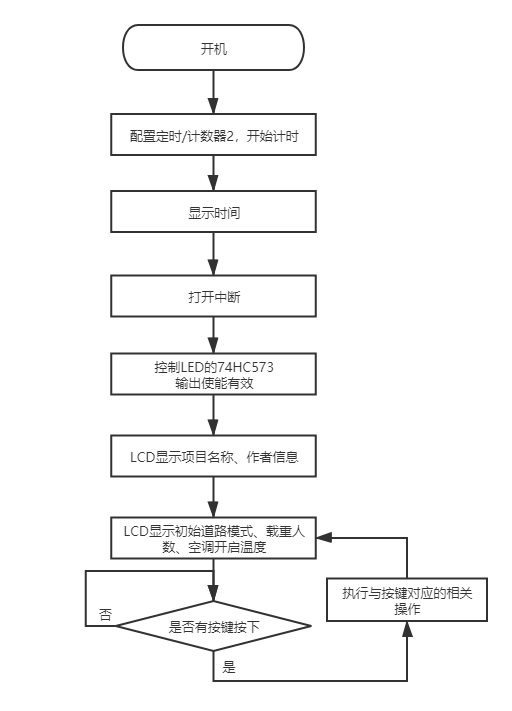


图 32硬件板系统主流程图

### 3.2按键扫描

按键扫描的原理与实验箱的原理相同，即列扫描行读回，但使用的端口不同。设置8255的工作方式为PC口高4位写状态、低4位读状态，PC口低4位外接VCC默认为高电平，PC口高4位依次循环置0。当有按键按下相应的行线电平会被置0，根据置0的行线和列线可以得出哪一个按键被按下。例如，当PC7=0且PC3=0时，代表左下角的按键被按下。监测到按键被按下后进入延时程序，若在延时程序结束后按键仍处于被按下的状态，则判断按键真正被按下，由此来消除按键抖动的影响。代码的具体实现见注释。

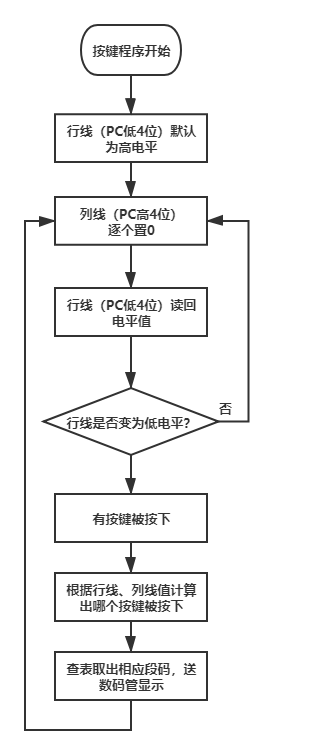


图 33硬件板按键扫描流程图

unsigned char KeyScan()

{

unsigned char Pos;

unsigned char i;

unsigned char k;

i = 4;//数码管有4位

Pos = 0x80; //从最高位开始置1

do {

PortC = ~ Pos;//从最高位开始置0

Pos >>= 1;//置0第2位

k = ~IN & 0x0f;//获取当前行线值

} while ((--i != 0) && (k == 0));//判断是否有按键按下

if (k != 0) {//以下为计算段码的公式

i \*= 4;

if (k & 2)

i += 1;

else if (k & 4)

i += 2;

else if (k & 8)

i += 3;

do Delay(10);

while (TestKey()); //延时消抖

return(KeyTable[i]); //通过查表的方式获取段码

} else return(0xff);

}

### 3.3单色点阵显示

单色点阵有PA口控制行，P1口控制列，当PA口和P0口同时置0，对应的点会被点亮。例如，行和列全给低时，64个LED全亮；行全给低，列部分高部分低，结果是给低的几列亮，其余几列灭；列全给高，行部分给高部分给低时可类比；通过逐行逐列依次刷新可以显示字符或图案。利用for循环切换所选字模，即可连续显示多个字符。具体软件实现过程见“单色点阵显示流程图”。

取模方法：使用文字图形取模软件，选用字节倒序和横向取模，输入对应的字可以自动生成字模；也可以自行绘制字模。

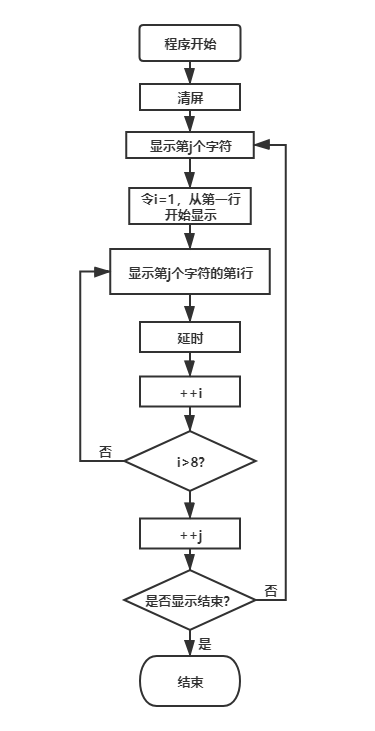


图 34单色点阵显示流程图

void DisplayDZtemp()

{

uchar j,i;

uchar count;

uchar bitmask;

//\*\*\*\*\*\*清屏\*\*\*\*\*\*\*\*\*

PortA=0xff;//低电平有效

P1=0xff;//低电平有效

for(j=3; j<4; j++){ //逐个显示所选字图。j:字模库中第j个字。

for(count =0; count <150; count ++){//每个字显示150次

BEE();

bitmask = 0x01;

for(i=0;i<8;i++){ //把第j个数逐行扫描

// PortA=0xff;//低电平有效//\*\*\*\*\*\*清屏\*\*\*\*\*\*\*\*\*

// P1=0xff;//低电平有效

P1= ~ Font[j][i];//给第i行送值（P1能锁存）

PortA=bitmask;//允许第一行（点亮第一行）

delayDZ (150);

bitmask <<= 1;//开启下一行

}

}

P1 = 0xff;

}

}

### 3.4数码管显示开车时间

硬件板使用的芯片是STC89C52RC，相比实验箱的AT89C51芯片增加了一个定时器。数码管显示开车时间使用了定时/计数器2，定时/计数器2与定时/计数器1和0不同，它有三种工作方式分别是捕获、自动重新装载（递增或递减计数）和波特率发生器。自动重装模式下当EXEN=0时，定时器2递增计数到0FFFFH，并在溢出后将TF2置位，然后将RCAP2L和RCAP2H中的16位值作为重新装载值装入定时器2中。RCAP2L和RCAP2H都是通过软件预设的。当EXEN=1时，16位重新装载可通过溢出或T2EX从1到0的负跳变实现，此负跳变同时将EXF2置位。如果定时器2中断被使能，则当TF2或者EXF2置1时产生中断。

令定时器T2工作在自动重新装载方式且EXEN=0，设置初值为(0xFFFF-917)/256，即12MHz晶振下定时1ms，溢出自动重装。溢出后进入中断程序，在中断程序中使用time和flag\_miao两个标志位实现1秒的计时，并使用count送显，结合数码管的刷新程序即可实现时间的显示；送显时，count对10求余代表“时”、count除以10后对10求余代表“分”、count除以100代表“秒”；具体实现见代码及流程图。

void timer2() interrupt 5

{

TF2=0; //定时器2必须由软件对溢出标志位清零

Time++;//用于计时

if(Time==100) {Time=0;flag\_miao++;}

if(!(Time%10)&&!flag\_key2) if(++flag\_key>3) flag\_key=0;//

if(!(Time%2)) flag\_tube=1;

if(flag\_miao==10)//整秒标志

{//显示播放时间

flag\_miao=0;//整秒标志置0

disp[3]=count%10;//显示小时

disp[2]=count/10%10;//显示分钟

disp[1]=count/100;//显示秒

count++;

//八段管刷新

P2=0x4f;//每秒都进行一次流水

if(flag\_i<8)

P0=~(1<<flag\_i),flag\_i++;

else if(flag\_i<14)

P0=~(0x80>>(flag\_i-7)),flag\_i++;

else flag\_i=0,P0=~(1<<flag\_i),flag\_i++;

P2=0x6f;

}

if(delay\_time)delay\_time--;

//数码管扫描

if(flag\_tube)

{

flag\_tube=0;

P2&=0xe3;

P2|=i<<2;

PortB=table[disp[i]];

i++;

if(i>3) i=0;

}

}

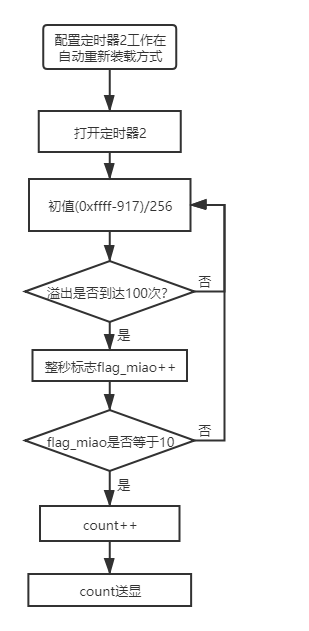


图 35时间显示程序流程图

### 3.5其他模块

硬件板还使用了蜂鸣器模块和LCD显示模块，由于蜂鸣器模块和LCD显示模块在实验箱部分已有详细叙述，故在此不再重复。

## 实习心得

### 4.1遇到问题及解决

**1)三极管的三个引脚距离过近，应该如何焊接？**  
 答：将三极管插入，并将引脚掰弯固定。先焊接中间的引脚，再焊接两侧的引脚。焊接时，使用适量的焊锡即可，切忌使用过多的锡。

**2)硬件板的烧录过程与实验箱的不同，应该如何解决？**

答：硬件板使用串口线烧录程序，通过在网络上收集资料，我们得知windows8以上的系统不需要手动安装驱动，可直接进行烧录。烧录的过程中需要将硬件板的电源关闭再打开。

**3)硬件板中点阵和键盘的工作模式与实验箱的有区别，应该如何解决？**

答：我们到解决方法是，按照伟福实验箱例程的思路，一步一步自行编写程序。

**4)无法检测到单片机。**

答：首先我们将STC-ISP软件升级到最高版本，但是问题依旧没有得到解决。我们将样品硬件板上的单片机安装到新的硬件板上，则成功烧录。我们猜测，无法检测到单片机的原因是单片机与硬件板不匹配。

**5)硬件板的键码与实验箱的不对应，应该如何解决？**

答：将伟福实验例程中的键码表先上下镜像，然后顺时针旋转90度，即可得到硬件板上对应的正确键码值。

### 4.2收获感想

在硬件板部分的实习中，我们遇到了许许多多的问题。在焊接过程中，由于经验不足，不慎将三极管的单个引脚焊接在一起；在软件调试前期，由于缺少硬件板的例程，各模块的代码需要自己动手编写。虽然说自行编写代码的过程非常痛苦，但是我们的编程能力也从中得到提升，对各个模块原理的理解也更加透彻。在王永涛老师的帮助下，我们小组还尝试使用了RPLIDAR传感器，通过阅读文件和实际操作，我们基本了解了RPLIDAR传感器的工作原理，开阔了眼界，但最后因为波特率不匹配的原因没能在硬件板上使用该传感器。在整个实习的过程中，我的编程能力得到了很大的提升，对整个系统的设计理念也有了较为清晰的认识，将自己在课程中学习到的理论转换为实际的操作。

# 实习意见建议

本次实习的指导老师们都非常的负责，老师授课的PPT生动有趣；在我们遇到问题的时候，老师总能出现在我们身边耐心的给我们答疑解惑，在老师的帮助下我们收获良多。美中不足是，实习群里的资料过多，有些资料没有说明具体的用途，有时候会导致我们感觉无从下手；另外，硬件板的设计也存在一些问题，比如说LED灯不能锁存等。虽然说这些问题对于老师来说可能是非常容易解决的，但是对于我们来说这些问题可能并不是那么好解决的，首先我们需要理解错误的原因， 其次是我们需要尝试几种不同的方法才能将问题解决，这都需要花费大量的时间。