浙江大学

本科实验报告

课程名称： 电子电路设计实验

姓　　名： 王跃民

学　　院： 信息与电子工程学院

　　系： 微电子科学与工程

专　　业： 微电子科学与工程

学　　号： 3210101552

指导教师： 施红军、叶险峰

2023 年 6 月 13 日

专业：微电子科学与工程
姓名：王跃民
学号：3210101552
日期：2023/6/13
地点：东四 216

浙江大学实验报告

课程名称：电子电路设计实验Ⅱ 指导老师：叶险峰、施红军 成绩：
实验名称：多功能数字时钟的设计与制作 实验类型：探索实验 同组学生姓名：孙雨豪

一、实验目的

1、学习掌握用 Arduino UNO 设计数字时钟；
2、学习掌握 PCB 电路板的设计和制作；
3、学习掌握 Arduino UNO 扩展板的设计与制作；
4、学习掌握 DS1302 时钟芯片和 LCD1602 液晶显示屏的使用。

二、实验任务与要求

1、用 Arduino UNO 设计多功能数字时钟，要求实现以下基本功能：
（1）显示时间、日期和星期
（2）断电保存时间
（3）通过按钮设置时间、日期
（4）整点响铃
（5）自定义闹钟
（6）显示温度
（7）自定义报警温度
（8）按键功能：按选择键进入设置时间功能；同时按 +- 键进入闹钟和报警温度设置功能；
（9）再按选择键光标跳动，光标跳到哪，当前的参数即可通过加减键修改。
2、自行添加创新功能；
3、设计的电路，完成相应器件的选择和参数计算，制作 Arduino UNO 扩展板；
4、编制与调试多功能数字时钟程序；
5、将制作的 Arduino UNO 扩展板与 Arduino UNO 板组装后，进行系统联调。

三、主要仪器设备

面包板、跳线电阻等基本元件若干、Arduino UNO、DS1302、LCD1602、LM35、蜂鸣器、按钮若干、装有 AD9 以及 Arduino IDE 的电脑。

四、实验原理

1、利用 Arduino UNO 板调用 DS1302、LCD1602、LM35 和蜂鸣器等模块实现上述功能。

五、实验步骤

1、用 AD 软件设计出电路原理图并仿真验证；
2、编写对应代码实现要求的功能；
3、在面包板上搭建设计出的电路进行实验，并通过实验结果修改原理图；
4、利用 AD 软件绘制对应的 PCB 板图并印刷。
5、在 PCB 板上焊接各类电子元件；
6、将代码拷贝到 Arduino UNO 板进行测试，并修改代码；

装 订 线

实验名称：__多功能数字时钟的设计与制作__ 姓名：__王跃民___ 学号：__3210101552_____

六、实验结果

1、绘制电路原理图

依据参考资料提供的电路原理图（Fig.1），在 AD 上画出如图 Fig.2 所示的电路原理图。

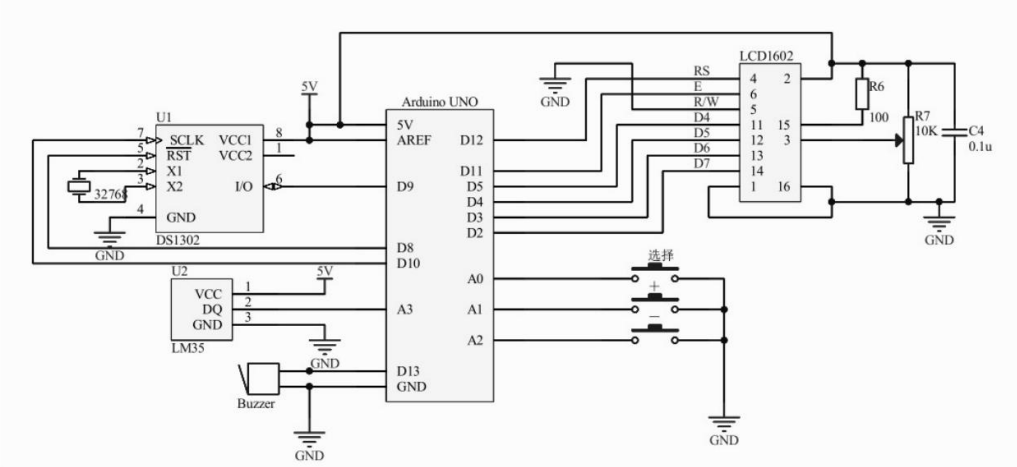


Fig.1

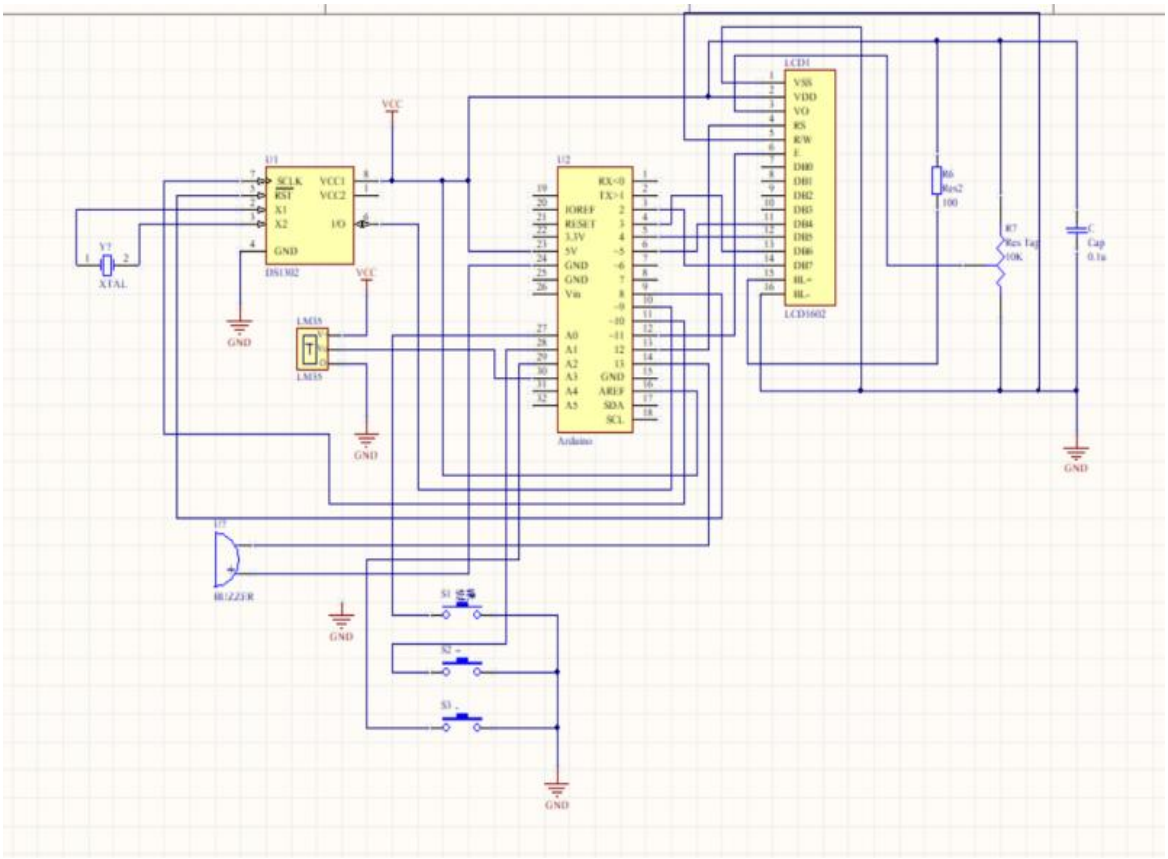


Fig.2

2、绘制 PCB 板图

利用 AD 软件对上图（Fig.2）电路原理图转变为双面板的 PCB 板图（Fig.3）。

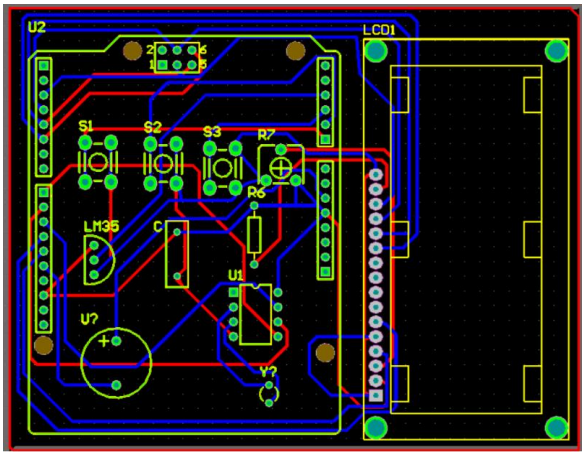


Fig.3

3、基本功能调试

（1）显示时间、日期、星期以及温度

实现原理：利用 time 函数计算时间并在 LCD 上显示小时、分钟以及秒数；利用 Day、Month、Year 函数分别计算日、月、年并在 LCD 上显示；利用 week 函数计算星期并在 LCD 上显示；利用 GetTemperatures 函数读取 LM35 的温度信息并在 LCD 上显示当前温度。

简要实现代码：

```
/**计算小时、分钟、秒数以及日期进位*/
void time() {
}
/** 根据年月计算当月天数 */
int Days(int year, int month){
}
/** 计算当月天数 */
void Day(){
}
/** 计算月份 */
void Month(){
}
/** 计算年份 */
void Year(){
}
/** 根据年月日计算星期几 */
void Week(int y,int m, int d){
}
/** 获取 LM35 温度 */
void GetTemperatures(){
}
/** 显示时间、日期、星期 、温度*/
void Display() {
lcd.setCursor(0, 1) ;
lcd.print("              ");
lcd.setCursor(0,0) ;
```

```
lcd.print("                ");//将 LCD 屏幕上显示信息清空
GetTemperatures();
time();
Day();
Month();
Year();
Week(year,month,day);
}
```

实现效果：

（2）断电保存时间

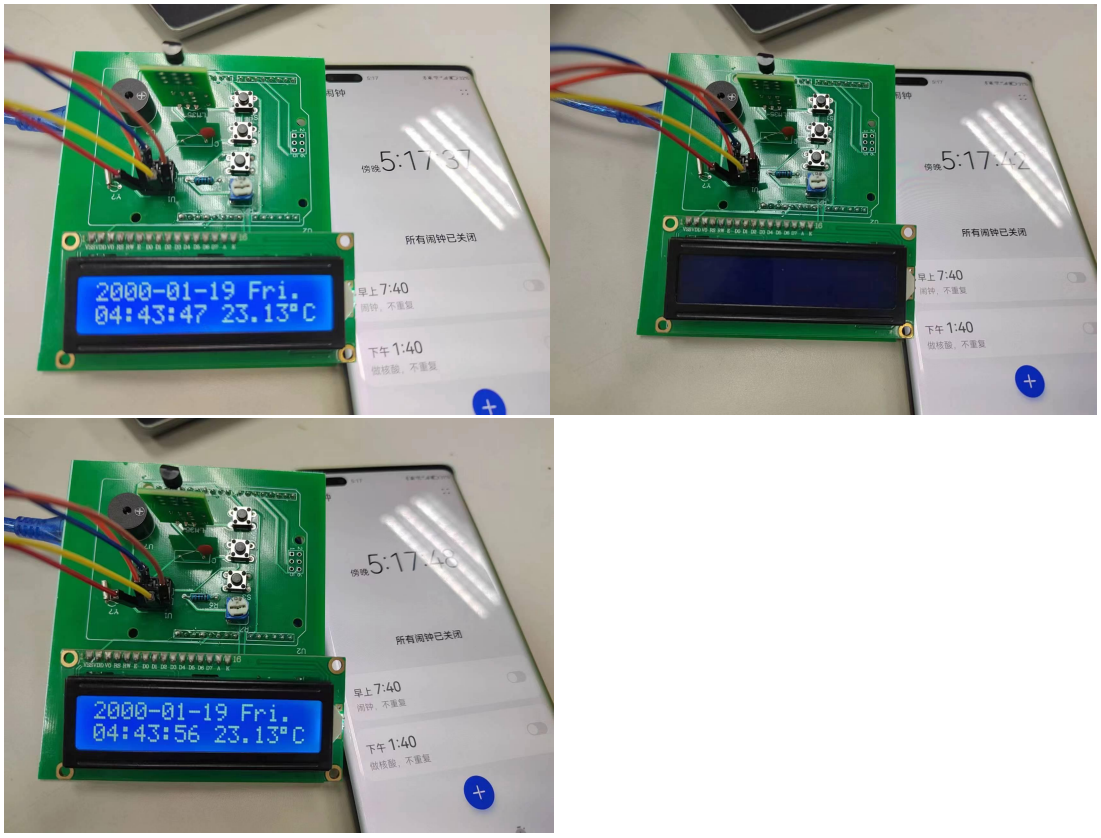
实现原理：setup 函数中实现读取 DS1302 中保存的时间，loop 函数中将现在显示的时间存入 DS1302 中，同时给 DS1302 配备一个独立电源，这样当我们拔出 Arduino 电源后，DS1302 还能自行计时，通电后又能将实时时间显示出来。（注意：最开始第一次由于 DS1302 中并没有保存时间（数据与实际需要时间相差太远，利用按键调节太过麻烦），所以可以第一次在 setup 函数中先自己设置一个时间并存入 DS1302，然后以后就可以将这一段注释掉）

简要实现代码：

```
int second = 0, minute = 0, hour = 0, day = 0, month = 0, year = 0; //当前时间
int SECOND = 0, MINUTE = 0, HOUR = 0, DAY = 0, MONTH = 0, YEAR = 0; //初始时间
void setup(){
Time t;
//t.hour=12;t.min=59;t.sec=59;          注释部分仅在第一次使用
//t.year=2023;t.mon=6;t.date=13;
//rtc.set_time(t.year,t.mon,t.date,t.hour,t.min,t.sec); //设置 DS1302 芯片初始时间
t=rtc.getTime();
SECOND=t.sec;MINUTE=t.min;HOUR=t.hour;
YEAR=t.year;MONTH=t.mon;DAY=t.date;
}
void loop() {
rtc.setTime(hour,minute,second); //向 DS1302 中保存时间
rtc.setDate(year,month,day);
```

}

实现效果：



（3）按键设置时间

实现原理：首先利用 6 个小函数（分别对应修改小时、分钟、秒、年、月、日）实现显示指定位置的光标并显示对应的数据，并能对按下 add 和 minus 键做出相应的变化，并刷新显示的数值。然后一个大函数通过按下 choose 键进入，然后按下 choose 键会改变 chose 的值，根据 chose 的值不同执行不同的小函数，并在 chose 超过指定值之后归零并退出函数。

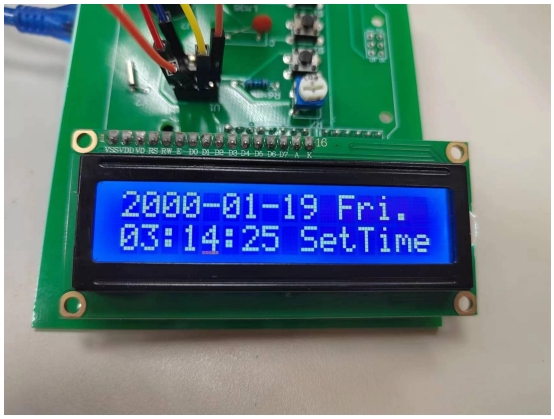
简要实现代码：

```
/** 通过按键设置时间 */
void Set_Time(int rol, int row, int &Time){
DisplayCursor(rol, row);
if(digitalRead(add) == LOW){
delay(ButtonDelay);
if(digitalRead(add) == LOW){
Time ++;
}
Display();
}
if(digitalRead(minus) == LOW){
……}
}
/** 按键选择 */
void Set_Clock(){
```

```
if(digitalRead(choose)==LOW){
……
while(1){
if(digitalRead(choose) == LOW){
delay(ButtonDelay);
if(digitalRead(choose) ==LOW){
chose++;
}
}
seconds = millis()/1000;
Display();
if(chose == 1){
Set_Time(1, 1, HOUR); //SetHour
}else if(chose == 2){
Set_Time(4, 1, MINUTE); //SetMinute
}else if(chose == 3){
Set_Time(7, 1, SECOND); //SetSecond
}else if(chose == 4){
Set_Time(9, 0, DAY); //SetDay
}else if(chose == 5){
Set_Time(6, 0, MONTH); // SetMonth
}else if(chose == 6){
Set_Time(3, 0, YEAR); //SetYear
}else if(chose >= 7) {
chose = 0;
delay(200);
break;
} } } }
```

在参考函数的基础上主要是在最后 chose>=7 的情况中加了一个 delay(200)，因为在实际操作时经常由于反应过于灵敏而在 chose=0 这一步执行之后 choose 又算按压了一次就又进入了 settime 模式，所以加一个 delay 来给使用者一个舒适的时间松开按键。加入 delay 之后就再也没有退出失败的情况了，并且不影响修改之前的效果。

实现效果：



图示效果是设置分钟时的情况，光标在分钟的个位（4 的位置）

（4）按键调整闹钟时间和报警温度

实现原理：与设置时间类似，不同是同时按下 add 和 minus 键进入该模式，有 3 个小函数对应修改闹钟小时、分钟和报警温度。然后按下 choose 键会改变 alarm_chose 的值，根据 alarm_chose 的值不同执行不同的小函数，并在 alarm_chose 超过指定值之后归零并退出函数。

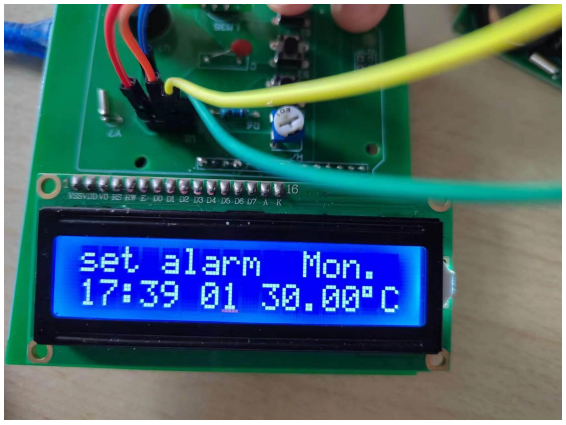
简要实现代码：

```
/** 设置闹钟小时 */
void Set_Alarm_Hour(){
DisplayCursor(1, 1);
if(digitalRead(add) == LOW){
delay(ButtonDelay);
if(digitalRead(add) == LOW){
alarm_hour ++;
if(alarm_hour == 24){
alarm_hour = 0;
}
FormatDisplay(0,1,alarm_hour);
} }
if(digitalRead(minus) == LOW){
…… } }
/** 设置闹钟分钟 */
void Set_Alarm_Minute(){
DisplayCursor(4, 1);
if(digitalRead(add) == LOW) {
delay(ButtonDelay);
if(digitalRead(add) == LOW){
alarm_minute ++;
if(alarm_minute == 60){
alarm_minute = 0;
}
FormatDisplay(3,1,alarm_minute);
} }
if(digitalRead(minus) == LOW){
……} }
/** 设置报警温度 */
void Set_Alarm_Temp(){
DisplayCursor(10, 1);
if(digitalRead(add) == LOW) {
delay(ButtonDelay);
if(digitalRead(add) == LOW){
Temp_Alarm ++;
} }
if(digitalRead(minus) == LOW){
……} }
/** 进入报警设置 */
void Set_Alarm(){
```

```
if(digitalRead(add) == LOW && digitalRead(minus) == LOW){
lcd.setCursor(0, 0);
lcd.print("set alarm ");
while(1){
if(digitalRead(choose) == LOW){
delay(ButtonDelay);
if(digitalRead(choose) == LOW){
alarm_choose++;
} }
FormatDisplay(0,1,alarm_hour);
lcd.setCursor(2, 1);
lcd.print(":");
……
lcd.setCursor(15, 1);
lcd.print("C"); //显示字母 C
if(alarm_choose == 1){
Set_Alarm_Hour();
}else if(alarm_choose == 2){
Set_Alarm_Minute();
}……else if(alarm_choose == 4){
Set_Alarm_Temp();
}……else if(alarm_choose >= 6){
alarm_choose=0;break;
}}}}
```

上述六行代码实现显示当前闹钟时间报警温度

实现效果：



（5）整点响铃、温度报警、闹钟响铃

实现原理：三个函数分别对应上述三个功能，验证满足响铃条件（整点、温度超标、到达闹钟时间）后利用 tone 函数实现不同频率的声音，然后 delay 函数控制响铃时间，noTone 函数结束响铃。

简要实现代码：

```
/** 正点蜂鸣 */
void Point_Time_Alarm(){
if(minute == 0 && second == 0){
tone(Tone,frequence);
```

```
    delay(600);
    noTone(Tone);
    }
    }
    /** 闹钟 指定时间蜂鸣 */
    void Clock_Alarm(){
    if(hour == alarm_hour && minute == alarm_minute && second ==
alarm_second){
    ……
    switch(model)
    {
     case 1:tone(Tone,1500);break;
     case 0:tone(Tone,2500);break;
     default:break;
    }
    delay(5000);
    }
    noTone(Tone);
    }
    /** 超过指定温度报警 */
    void Temperatures_Alarm(){
    if(Temperatures >= Temp_Alarm){
    tone(Tone,2000);
    delay(600);
    noTone(Tone);
    } }
```

swtich 部分代码与后续创新功能有关

由于响铃并不能体现出来所以报告中不展示实现效果。

4、创新功能测试

（1）选择闹钟铃声

创新效果：可以调节不同的闹钟铃声。

实现原理：创建 model 这一个全局变量记录所选铃声。编写一个类似设置闹钟时间的小函数实现设置 model，并在设置闹钟的主函数中 alarm_choose 的判定中加一个对进入设置闹钟铃声函数的判定。在 Clock_Alarm 函数中添加 switch(model)语句块，对不同的 model，tone 函数设置不同的铃声频率。

简要实现代码：

```
    /**设置闹钟铃声**/
    void Set_Alarm_Model(){
    DisplayCursor(7, 1);
    if(digitalRead(add) == LOW) {
    delay(ButtonDelay);
    if(digitalRead(add) == LOW){
    model ++;
    if(model == 2){
    model = 0;
```

```
}
FormatDisplay(6,1,model);
} }
if(digitalRead(minus) == LOW){
……} }
/** 进入报警设置 */
void Set_Alarm(){
if(digitalRead(add) == LOW && digitalRead(minus) == LOW){
……
while(1){
……
else if(alarm_choose == 3){
Set_Alarm_Model();
}……} } }
/** 闹钟 指定时间蜂鸣 */
void Clock_Alarm(){
……
switch(model)
{
 case 1:tone(Tone,1500);break;
 case 0:tone(Tone,2500);break;
 default:break;
}
……
}
```

由于响铃并不能体现出来所以报告中不展示不同铃声实现效果，设置铃声界面在基础功能设置闹钟时间时已经展示。

（2）设置闹钟提醒文本

创新效果：响铃时，LCD 上显示设置好的提示文本，比如到了这个时间要做什么。

实现原理：创建一个字符串数组记录设置的文本。编写一个类似设置闹钟时间的小函数实现设置闹钟文本，并在设置闹钟的主函数中 alarm_choose 的判定中加一个对进入设置闹钟文本函数的判定。在 Clock_Alarm 函数中判定成功后先输出设置好的文本之后再进行 model 判定及响铃。

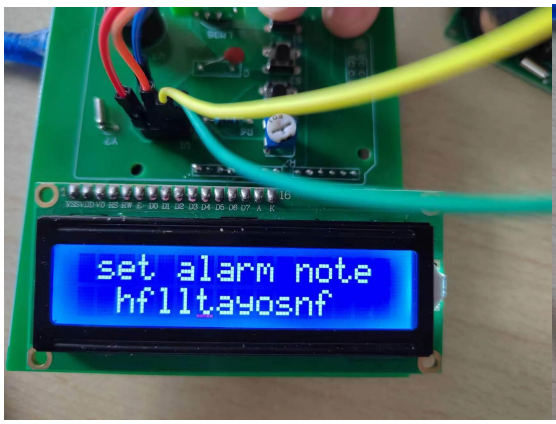
简要实现代码：

```
/**设置闹钟文本**/
void Set_Alarm_Note()
{
if(alarm_choose==5){
while(1){
……
if(digitalRead(choose) == LOW){
delay(ButtonDelay);
if(digitalRead(choose) == LOW){
note_choose++;
```

note_choose 相当于嵌套了一层类似设置

```
    }                                   时间的函数，因为字符串就相当于是十几
    }                                   个可以修改的数据
    lcd.setCursor(0, 1);
    lcd.print("  ");
    lcd.setCursor(2, 1);
    lcd.print(alarm_note);
    lcd.setCursor(13, 1);
    lcd.print("   ");                   上述 6 行实现 LCD 显示当前闹钟文本
    setnote(note_choose-1);
    if(note_choose >= 12){
    note_choose = 0; alarm_choose+=1;
    break;
    }}}}
    void setnote(int n)                 实现字符从' '到 'a'到'z'的变换
    {
    DisplayCursor(n+2, 1);
    if(digitalRead(add) == LOW){
    delay(ButtonDelay);
    if(digitalRead(add) == LOW){
    if(alarm_note[n]==' ')
    alarm_note[n]='a';
    else if(alarm_note[n]=='z'){
    alarm_note[n]=' ';
    } else alarm_note[n]++;
    lcd.setCursor(n+2, 1);
    lcd.print(alarm_note[n]);
    } }
    if(digitalRead(minus) == LOW){
    …… } }
    /** 闹钟 指定时间蜂鸣 */
    void Clock_Alarm(){
    if(hour == alarm_hour && minute == alarm_minute && second ==
alarm_second){
    lcd.setCursor(0, 0) ;
    lcd.print(" time for alarm ");
    lcd.setCursor(0, 1);
    lcd.print("  ");
    lcd.setCursor(2, 1);
    lcd.print(alarm_note);              输出设定的文本
    lcd.setCursor(13, 1);
    lcd.print("   ");
……}
```

实现效果：



（左图为设置闹钟文本界面，右图为响铃时 LCD 屏画面）

七、讨论、心得

在功能调试阶段我们遇到了一个“灵异事件”，在前两周测试时蜂鸣器正常发声，而从第三周开始蜂鸣器就一直不叫了，我们就开始寻找问题的缘由，由于前两周正常所以我们第一反应是，实现新功能新添加的代码导致了问题，于是我们换取了之前蜂鸣器正常发声的代码测试，蜂鸣器依然没有发声，于是我们排除了第一个可能性；然后我们又猜测是否是蜂鸣器坏了，为什么我们会这样想呢，因为我们检查时发现 PCB 板上蜂鸣器的正负是接反了的，于是我们怀疑可能由于反接导致蜂鸣器烧坏了，于是我们找了老师帮忙直接给蜂鸣器加电压发现会正常振动，由于其他可能性没有进展，所以后续我们甚至更换了蜂鸣器，依然是没有解决问题；于是就是第三个可能性了----虚焊，真实的问题确实也是这个，但是我们首先是将蜂鸣器重新焊了一下，然后用电压表分别接在 GND、D13 端口以及蜂鸣器两端测试，在第一次测试时两端电压正常，第二次电压不正常，由于 GND、D13 测试结果正常于是就忽略了不一定是蜂鸣器虚焊而是 arduino 排针有虚焊的可能性，在这个过程中测试结果有时 0 有时 2 点几 V，我们认为是温度报警函数 tone 函数之后会 noTone 于是电压就会下降是正常情况，但是我们并没有注意到波动频率不是恒定而是随机的，所以导致我们一开始没有发现排针虚焊的问题，后面几周在前面几种可能中反复检查依旧没有找出问题；最后求助老师，在老师的帮助下找出问题，老师一开始也和我们一样不理解缘由，但是在利用万用表的二极管档位后一下就测试出来 GND 和蜂鸣器负极不导通，于是就发现了虚焊问题。

上述调试过程，提高了我对于这一类问题的几种常见缘由的查找能力，以及通过网络查询资料的能力。这一次经历也是提醒了我，在电子实验时一定要细致，测试时注意数值的变化，不能自以为或者是觉得没有问题，应该要测试之后才能排除。

在整个实验过程中，我体会到了团队合作的重要性，以及一个良好的团队氛围的重要性，因为焊接过程全程是我操作的，但是最后蜂鸣器出现故障，队友并没有指责我，而是鼓励我积极地寻找解决方法。基本所有代码的修改与更新都是我在做，所以这一个过程中提高了我的代码能力以及阅读代码能力。如果不是团队作业，那么在连续几周都修改不出蜂鸣器故障的时候，我想我就会崩溃了，因为这实在是太“灵异”了，各种地方都修改了依旧没有奏效，但是因为我们有一个团队，一方面有一个伴所以心理压力没有那么大，另一方面也可以互相鼓励。所以团队协作真的很重要，这对于我以后其他课程大作业的协作有相当大的启示。

附：（完整代码）

```
#include <DS1302.h>
#include <DS1302.h>
/* *
* LCD RS pin to digital pin 12
* LCD Enable pin to digital pin 11
* LCD D4 pin to digital pin 5
* LCD D5 pin to digital pin 4
* LCD D6 pin to digital pin 3
* LCD D7 pin to digital pin 2
* LCD R/W pin to ground
* LCD VSS pin to ground
* LCD VCC pin to 5V
* */
#include <DS1302.h>
#include <LiquidCrystal.h> //LCD1602 显示头文件
LiquidCrystal lcd(12, 11, 5, 4, 3, 2);
#define choose A0 //选择端口
#define add A1 //加
#define minus A2 //减
#define Tone 13 //蜂鸣器端口
uint8_t CE_PIN = 8; //DS1302 RST 端口
uint8_t IO_PIN = 9; //DS1302 DAT 端口
uint8_t SCLK_PIN = 10; //DS1302 CLK 端口
DS1302 rtc(CE_PIN, IO_PIN, SCLK_PIN); //创建 DS1302 对象
unsigned long seconds;
int s = 0, m = 0, h = 0, d = 0, mon = 0, y = 0; //时间进位
int second = 0, minute = 0, hour = 0, day = 0, month = 0, year = 0; //当
前时间
int SECOND = 0, MINUTE = 0, HOUR = 0, DAY = 0, MONTH = 0, YEAR = 0; //初
始时间
int chose = 0, alarm_choose = 0 ,ButtonDelay = 10, frequence =
2093,note_choose=0;
int alarm_hour = 7, alarm_minute = 30, alarm_second = 0; //闹钟时间
float Temperatures, Temp_Alarm = 30 ;
int model=0;//闹钟铃声
char alarm_note[]="hello world";//闹钟文本
/** 格式化输出 */
void FormatDisplay(int col, int row,int num){
lcd.setCursor(col, row);
if(num < 10) lcd.print("0");
lcd.print(num);
}
/** 计算时间 */
```

```
void time() {
second = (SECOND + seconds) % 60; //计算秒
m = (SECOND + seconds) / 60; //分钟进位
FormatDisplay(6,1,second);
minute = (MINUTE + m) % 60; //计算分钟
h = (MINUTE + m) / 60; //小时进位
FormatDisplay(3,1,minute);
hour = (HOUR + h) % 24; //计算小时
d = (HOUR + h) / 24; //天数进位
FormatDisplay(0,1,hour);
lcd.setCursor(2, 1);
lcd.print(":");
lcd.setCursor(5, 1);
lcd.print(":");
}
/** 根据年月计算当月天数 */
int Days(int year, int month){
int days = 0;
if (month != 2){
switch(month){
case 1: case 3: case 5: case 7: case 8: case 10: case 12: days = 31; break;
case 4: case 6: case 9: case 11: days = 30; break;
}
}else{ //闰年
if(year % 4 == 0 && year % 100 != 0 || year % 400 == 0){
days = 29;
}
else{
days = 28;
}
}
return days;
}
/** 计算当月天数 */
void Day(){
int days = Days(year,month);
int days_up;
if(month == 1){
days_up = Days(year - 1, 12);
}
else{
days_up = Days(year, month - 1);
}
day = (DAY + d) % days;
```

```
if(day == 0){
day = days;
}
if((DAY + d) == days + 1 ){
DAY -= days;
mon++;
}
if((DAY + d) == 0){
DAY += days_up;
mon--;
}
FormatDisplay(8,0,day);
}
/** 计算月份 */
void Month(){
month = (MONTH + mon) % 12;
if(month == 0){
month = 12;
}
y = (MONTH + mon - 1) / 12;
FormatDisplay(5,0,month);
lcd.setCursor(7, 0);
lcd.print('-');
}
/** 计算年份 */
void Year(){
year = ( YEAR + y ) % 9999;
if(year == 0){
year = 9999;
}
lcd.setCursor(0, 0);
if(year < 1000){
lcd.print("0");
}
if(year < 100){
lcd.print("0");
}
if(year < 10){
lcd.print("0");
}
lcd.print(year);
lcd.setCursor(4, 0);
lcd.print('-');
}
```

```
/** 根据年月日计算星期几 */
void Week(int y,int m, int d){
if(m == 1){
m = 13;
}
if(m == 2){
m = 14;
}
int week = (d+2*m+3*(m+1)/5+y+y/4-y/100+y/400)%7+1;
String weekstr = "";
switch(week){
case 1: weekstr = "Mon. "; break;
case 2: weekstr = "Tues. "; break;
case 3: weekstr = "Wed. "; break;
case 4: weekstr = "Thur. "; break;
case 5: weekstr = "Fri. "; break;
case 6: weekstr = "Sat. "; break;
case 7: weekstr = "Sun. "; break;
}
lcd.setCursor(11, 0);
lcd.print(weekstr);
}
/** 显示时间、日期、星期 */
void Display() {
lcd.setCursor(0, 1) ;
lcd.print("              ");
lcd.setCursor(0,0) ;
lcd.print("               ");
GetTemperatures();
time();
Day();
Month();
Year();
Week(year,month,day);

}
/** 显示光标 */
void DisplayCursor(int rol, int row) {
lcd.setCursor(rol, row);
lcd.cursor();
delay(100);
lcd.noCursor();
delay(100);
}
```

```
/** 设置初始时间 */
void set(int y, int mon, int d, int h, int m, int s){
YEAR = y;
MONTH = mon;
DAY = d;
HOUR = h;
MINUTE = m;
SECOND = s;
}
/** 通过按键设置时间 */
void Set_Time(int rol, int row, int &Time){
DisplayCursor(rol, row);
if(digitalRead(add) == LOW){
delay(ButtonDelay);
if(digitalRead(add) == LOW){
Time ++;
}
Display();
}
if(digitalRead(minus) == LOW){
delay(ButtonDelay);
if(digitalRead(minus) == LOW){
Time --;
}
Display();
}
 }
/** 按键选择 */
void Set_Clock(){
if(digitalRead(choose)==LOW){
lcd.setCursor(9, 1);
lcd.print("SetTime");
while(1){
if(digitalRead(choose) == LOW){
delay(ButtonDelay);
if(digitalRead(choose) ==LOW){
chose++;
}
}
seconds = millis()/1000;
Display();
if(chose == 1){
Set_Time(1, 1, HOUR); //SetHour
}else if(chose == 2){
```

```
Set_Time(4, 1, MINUTE); //SetMinute
}else if(chose == 3){
Set_Time(7, 1, SECOND); //SetSecond
}else if(chose == 4){
Set_Time(9, 0, DAY); //SetDay
}else if(chose == 5){
Set_Time(6, 0, MONTH); // SetMonth
}else if(chose == 6){
Set_Time(3, 0, YEAR); //SetYear
}else if(chose >= 7) {
chose = 0;
delay(200);
break;
}
}
}
}
/**设置闹钟文本**/
void Set_Alarm_Note()
{
  if(alarm_choose==5){
while(1){
lcd.setCursor(0, 0);
lcd.print(" set alarm note ");
if(digitalRead(choose) == LOW){
delay(ButtonDelay);
if(digitalRead(choose) == LOW){
note_choose++;
}
}
lcd.setCursor(0, 1);
lcd.print("  ");
lcd.setCursor(2, 1);
lcd.print(alarm_note);
lcd.setCursor(13, 1);
lcd.print("   ");
setnote(note_choose-1);
if(note_choose >= 12){
note_choose = 0; alarm_choose+=1;
break;
}
}
}
}
```

```
void setnote(int n)
{
DisplayCursor(n+2, 1);

if(digitalRead(add) == LOW){
delay(ButtonDelay);
if(digitalRead(add) == LOW){
if(alarm_note[n]==' ')
alarm_note[n]='a';
else if(alarm_note[n]=='z'){
alarm_note[n]=' ';
} else alarm_note[n]++;
lcd.setCursor(n+2, 1);
lcd.print(alarm_note[n]);
}
}
if(digitalRead(minus) == LOW){
delay(ButtonDelay);
if(digitalRead(minus) == LOW){
if(alarm_note[n]==' ')
alarm_note[n]='z';
else if(alarm_note[n]=='a'){
alarm_note[n]=' ';
} else alarm_note[n]--;
lcd.setCursor(n+2, 1);
lcd.print(alarm_note[n]);
}
}
}
/** 设置闹钟小时 */
void Set_Alarm_Hour(){
DisplayCursor(1, 1);
if(digitalRead(add) == LOW){
delay(ButtonDelay);
if(digitalRead(add) == LOW){
alarm_hour ++;
if(alarm_hour == 24){
alarm_hour = 0;
}
FormatDisplay(0,1,alarm_hour);
}
}
if(digitalRead(minus) == LOW){
delay(ButtonDelay);
```

```
if(digitalRead(minus) == LOW){
alarm_hour --;
if(alarm_hour == -1){
alarm_hour = 23;
}
FormatDisplay(0,1,alarm_hour);
}
}
}
/** 设置闹钟分钟 */
void Set_Alarm_Minute(){
DisplayCursor(4, 1);
if(digitalRead(add) == LOW) {
delay(ButtonDelay);
if(digitalRead(add) == LOW){
alarm_minute ++;
if(alarm_minute == 60){
alarm_minute = 0;
}
FormatDisplay(3,1,alarm_minute);
}
}
if(digitalRead(minus) == LOW){
delay(ButtonDelay);
if(digitalRead(minus) == LOW){
alarm_minute --;
if(alarm_minute == -1){
alarm_minute = 59;
}
FormatDisplay(3,1,alarm_minute);
}
}
}
/**设置闹钟铃声**/
void Set_Alarm_Model(){
DisplayCursor(7, 1);
if(digitalRead(add) == LOW) {
delay(ButtonDelay);
if(digitalRead(add) == LOW){
model ++;
if(model == 2){
model = 0;
}
```

```
FormatDisplay(6,1,model);
}
}
if(digitalRead(minus) == LOW){
delay(ButtonDelay);
if(digitalRead(minus) == LOW){
model --;
if(model == -1){
model = 1;
}

FormatDisplay(6,1,model);
}
}
}
/** 设置报警温度 */
void Set_Alarm_Temp(){
DisplayCursor(10, 1);
if(digitalRead(add) == LOW) {
delay(ButtonDelay);
if(digitalRead(add) == LOW){
Temp_Alarm ++;
}
}
if(digitalRead(minus) == LOW){
delay(ButtonDelay);
if(digitalRead(minus) == LOW){
Temp_Alarm --;
}
}
}
/** 进入报警设置 */
void Set_Alarm(){
if(digitalRead(add) == LOW && digitalRead(minus) == LOW){
lcd.setCursor(0, 0);
lcd.print("set alarm ");
while(1){
if(digitalRead(choose) == LOW){
delay(ButtonDelay);
if(digitalRead(choose) == LOW){
alarm_choose++;
}
}
FormatDisplay(0,1,alarm_hour);
```

```
lcd.setCursor(2, 1);
lcd.print(":");
FormatDisplay(3,1,alarm_minute);
lcd.setCursor(5, 1);
lcd.print(" ");
FormatDisplay(6,1,model);
lcd.setCursor(9, 1);
lcd.print(Temp_Alarm);
lcd.setCursor(14, 1);
lcd.print((char)223); //显示 o 符号
lcd.setCursor(15, 1);
lcd.print("C"); //显示字母 C
if(alarm_choose == 1){
Set_Alarm_Hour();
}else if(alarm_choose == 2){
Set_Alarm_Minute();
}else if(alarm_choose == 3){
Set_Alarm_Model();
}else if(alarm_choose == 4){
Set_Alarm_Temp();
}else if(alarm_choose == 5){
Set_Alarm_Note();
}else if(alarm_choose >= 6){
alarm_choose=0;break;
}
}
}
}
/** 正点蜂鸣 */
void Point_Time_Alarm(){
if(minute == 0 && second == 0){
tone(Tone,frequence);
delay(600);
noTone(Tone);
}
}
/** 闹钟 指定时间蜂鸣 */
void Clock_Alarm(){
if(hour == alarm_hour && minute == alarm_minute && second == alarm_second){

lcd.setCursor(0, 0) ;
lcd.print(" time for alarm ");
lcd.setCursor(0, 1);
lcd.print("  ");
```

```
lcd.setCursor(2, 1);
lcd.print(alarm_note);
lcd.setCursor(13, 1);
lcd.print("   ");
switch(model)
{
  case 1:tone(Tone,1500);break;
  case 0:tone(Tone,2500);break;
  default:break;
}
delay(5000);
}

noTone(Tone);
}
/** 获取 LM35 温度 */
void GetTemperatures(){
long a = analogRead(A3); // Get temperatures
Temperatures = 125.0*a/1023;
lcd.setCursor(9, 1) ;
lcd.print(Temperatures); //获取温度
lcd.setCursor(14, 1);
lcd.print((char)223); //显示 o 符号
lcd.setCursor(15, 1);
lcd.print("C"); //显示字母 C
}
/** 超过指定温度报警 */
void Temperatures_Alarm(){
if(Temperatures >= Temp_Alarm){
tone(Tone,2000);
delay(600);
noTone(Tone);
}
}
void setup(){
for(int i = 2;i <= 13; i++){
pinMode(i,OUTPUT);
}
digitalWrite(add, HIGH);
digitalWrite(minus, HIGH);
digitalWrite(choose, HIGH);
lcd.begin(16, 2); //初始化 LCD1602
Time t;
//t.hour=12;t.min=59;t.sec=59;
```

```
//t.year=2023;t.mon=6;t.date=13;
//rtc.set_time(t.year,t.mon,t.date,t.hour,t.min,t.sec); //设置 DS1302芯片初始时间
t=rtc.getTime();
SECOND=t.sec;MINUTE=t.min;HOUR=t.hour;
YEAR=t.year;MONTH=t.mon;DAY=t.date;
rtc.writeProtect(false); // 关闭 DS1302 芯片写保护
rtc.halt(false); //为 true 时 DS1302 暂停
}
void loop() {
seconds = millis()/1000; //获取单片机当前运行时间
Display(); //显示时间
Set_Clock(); //设置时间
Set_Alarm(); //设置闹钟
Set_Alarm_Note();
Point_Time_Alarm(); //正点蜂鸣
Clock_Alarm(); //闹钟时间蜂鸣
GetTemperatures(); //获取 LM35 温度
Temperatures_Alarm(); //超过指定温度报警
//断电将单片机的当前时间写到 DS1302 芯片中
rtc.setTime(hour,minute,second); //设置 DS1302 芯片初始时间
rtc.setDate(year,month,day);
}
```