2014级物联网工程专业电子设计竞赛课程研究报告

填写时间：2016 年 06 月 11 日

姓名	张青	**学号**	14111206049
主要参与的项目（含课程竞赛，个人创新项目等）	单片机课程设计		
摘要（简要陈述所研究的内容，100字以内）： 研究单片机的硬件结构和芯片集成模块，利用CPU、寄存器、中断、定时技术，AD转换、传感器数据采集和处理、LCD显示原理、日期时间转换处理，实现了基于单片机和LCD模块的万年历功能。			
成绩评定			

目录

单片机万年历

一、 系统基本方案选择和论证

1.1.1 单片机芯片的选择方案和论证

方案一:

采用 89C51 芯片作为硬件核心，采用 Flash ROM，内部具有 4KB ROM 存储空间,能于 3V 的超低压工作,而且与 MCS-51 系列单片机完全兼容,但是运用于电路设计中时由于不具备 ISP 在线编程技术, 当在对电路进行调试时,由于程序的错误修改或对程序的新增功能需要烧入程序时，对芯片的多次拔插会对芯片造成一定的损坏。

方案二:

采用 AT89S52,片内 ROM 全都采用 Flash ROM；能以 3V 的超底压工作；同时也与 MCS-51 系列单片机完全该芯片内部存储器为 8KB ROM 存储空间，同样具有 89C51 的功能，且具有在线编程可擦除技术，当在对电路进行调试时，由于程序的错误修改或对程序的新增功能需要烧入程序时，不需要对芯片多次拔插，所以不会对芯片造成损坏。

所以选择采用 AT89S52 作为主控制系统.

1.1.2 显示模块选择方案和论证

方案一：

采用 LED 液晶显示屏,液晶显示屏的显示功能强大,可显示大量文字,图形,显示多样,清晰可见,选用 1602LCD 显示屏，可以显示较多信息，尽量满足万年历的显示需求。

方案二：

采用点阵式数码管显示，点阵式数码管是由八行八列的发光二极管组成，对于显示文字比较适合,如采用在显示数字显得太浪费,且价格也相对较高,所以也不用此种作为显示.

方案三：

采用 LED 数码管动态扫描,LED 数码管价格适中,对于显示数字最合适,但是数码管显示的信息有限，无法满足万年历的需求。

所以采用了 1602LCD 作为显示。

1.1.3 时钟芯片的选择方案和论证

方案一：

直接采用单片机定时计数器提供秒信号，使用程序实现年、月、日、星期、时、分、秒计数。综合利用硬件定时和外部脉冲计数，节约成本，一定程度上提高了精度，虽然实现的时间误差较大，但目前别无选择，只能选择此种时钟芯片。

1.1.4 温度传感器的选择方案与论证

方案一：

使用热敏电阻作为传感器，用热敏电阻与一个相应阻值电阻相串联分压，利用热敏电阻阻值随温度变化而变化的特性，采集这两个电阻变化的分压值，并进行 A/D 转换。。此设计方案需用 A/D 转换电路，增加硬件成本而且热敏电阻的感温特性曲线并不是严格线性的，会产生较大的测量误差。

方案二：

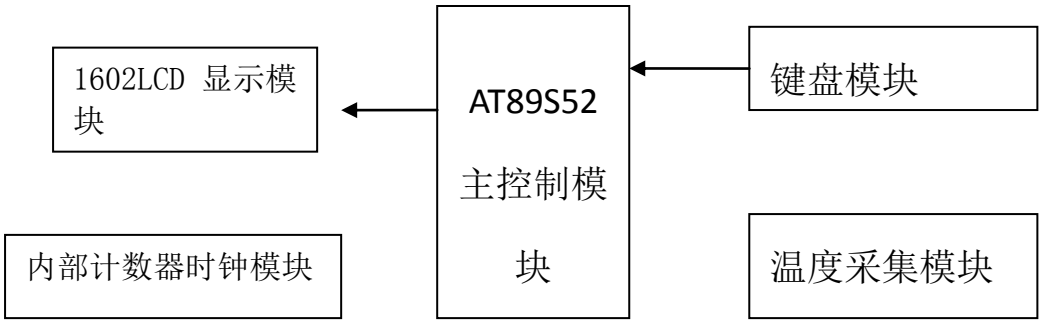
采用数字式温度传感器 DS18B20，此类传感器为数字式传感器而且仅需要一条数据线进行数据传输，易于与单片机连接，可以去除 A/D 模块，降低硬件成本，简化系统电路。另外，数字式温度传感器还具有测量精度高、测量范围广等优点。并且在单片机的集成开发板上本身带有 18B20 温度传感器模块。所以选择此种传感器。

1.2 电路设计最终方案决定

综上各方案所述,对此次作品的方案选定：采用 AT89S52 作为主控制系统; 内部时钟芯片提供时钟;数字式温度传感器;1602LCD 作为显示。

二、系统的硬件设计与实现

2.1 电路设计框图



2.2 系统硬件概述

本电路是由 AT89S52 单片机为控制核心，具有在线编程功能，低功耗，能在 3V 超低压工作；利用 cpu、寄存器、中断、定时技术，AD 转换、传感器数据采集和处理、LCD 显示原理、日期时间转换处理，实现了基于单片机和 LCD 模块的万年历功能。

时钟电路由内部计数器提供，它是一种高性能、低功耗、带 RAM 的实时时钟电路，它可以对年、月、日、周日、时、分、秒进行计时，闰年识别功能。工作电压为 2.5V～5.5V。

可产生年、月、日、周日、时、分、秒，具有使用寿命长，精度高和低功耗等特点，温度的采集由 DS18B20 构成；显示部份是 1602LCD 显示模块。

2.3 主要单元电路的设计

2.3.1 单片机主控制模块的设计

AT89S52 单片机为 40 引脚双列直插芯片,有四个 I/O 口 P0,P1,P2,P3, MCS-51 单片机共有 4 个 8 位的 I/O 口（P0、P1、P2、P3），每一条 I/O 线都能独立地作输出或输入。

单片机的最小系统如下图所示,18 引脚和 19 引脚接时钟电路,XTAL1 接外部晶振和微调电容的一端,在片内它是振荡器倒相放大器的输入,XTAL2 接外部晶振和微调电容的另一端,在片内它是振荡器倒相放大器的输出.第 9 引脚为复位输入端,接上电容,电阻及开关后够上电复位电路,20 引脚为接地端,40 引脚为电源端.　如图-1 所示

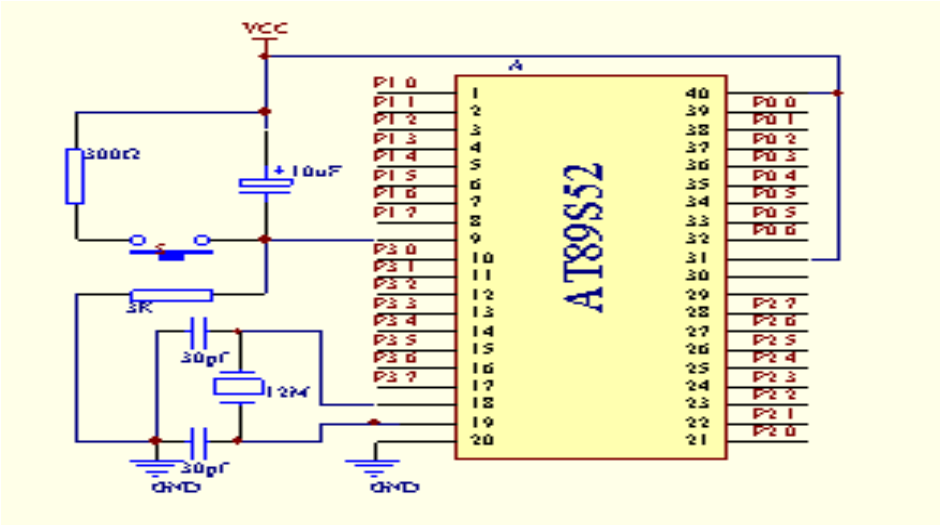


图-1 主控制系统

2.3.2 时钟电路模块的设计

图-2 示出单片机内部时钟的原理图，图中 TMOD 寄存器选择定时器工作模式，此系统中选择定时器内部定时，计数器外部计数模式。单片机晶振为 12MHz，对振荡器进行 12 分频，设置 TL0 和 TH0 初值，溢出后外部计数脉冲取反，满 10 次为 1s，从而将秒钟加 1，实现时间的获取。

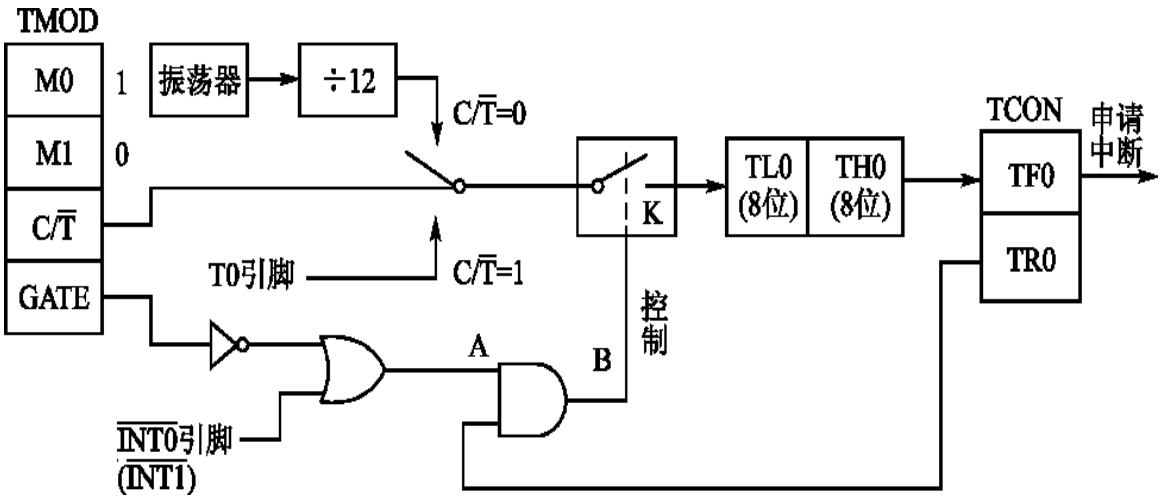


图-2 单片机内部定时器原理图

2.3.3 温度采集模块设计

如图-3 所示。采用数字式温度传感器 DS18B20，它是数字式温度传感器，具有测量精度高，电路连接简单特点，此类传感器仅需要一条数据线进行数据传输，使用 P 0.7 与 DS18B20 的 I/O 口连接加一个上拉电阻,Vcc 接电源,Vss 接地。

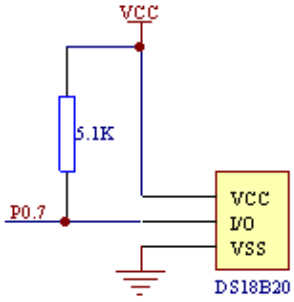


图-3 DS18B20 温度采集

2.3.4 显示模块的设计

如图－4 所示，1602LCD 作为显示模块，GND、VCC、V0 作为电源驱动，RS、RW、E 作为控制信号，DB0~DB7 作为数据输入信号，BL1、BL2 作为背光灯驱动，使用时，对相应引脚进行编程，将数据保存在数组中，按行对 LCD 显示扫描，并且做适当延时，以保证显示效果。

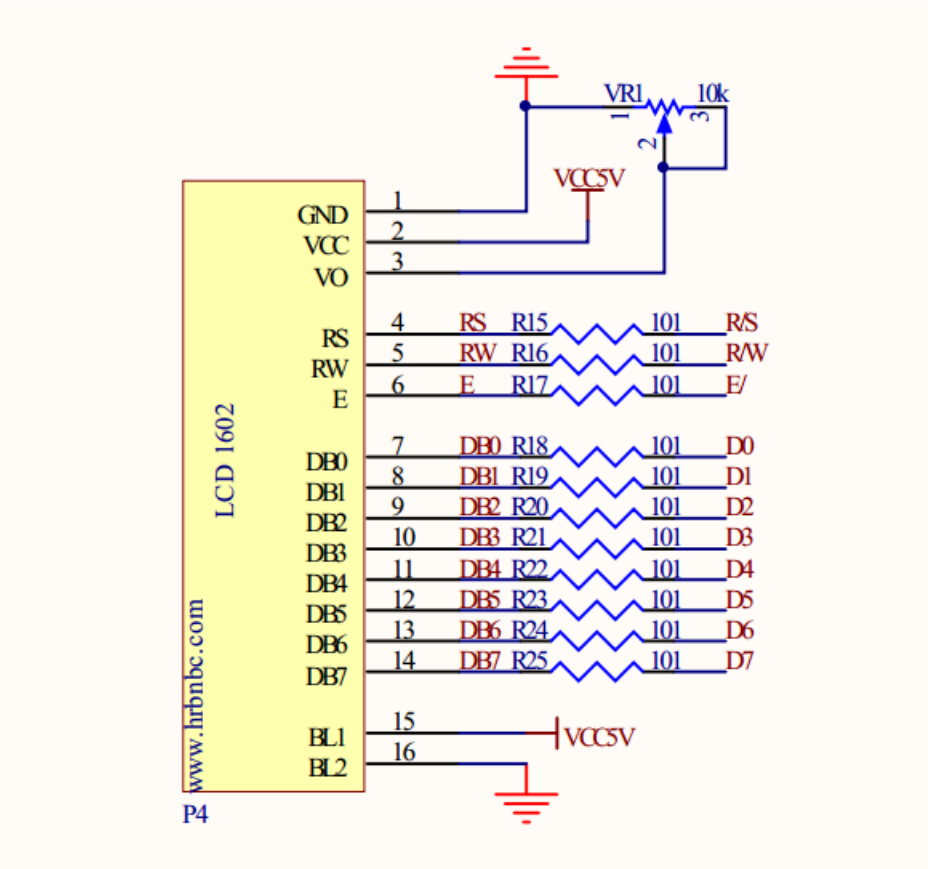


图 4 LCD1602 电路图

2.3.5 按键控制模块的设计

如图-5 所示，五个独立按键用来控制年、月、日、时、分的数值，使用时，利用软件对按键进行消抖，判断按键值为低电平时，即可认为相应按键被按下，对应的数值加 1，以此来控制时间和日期的调整

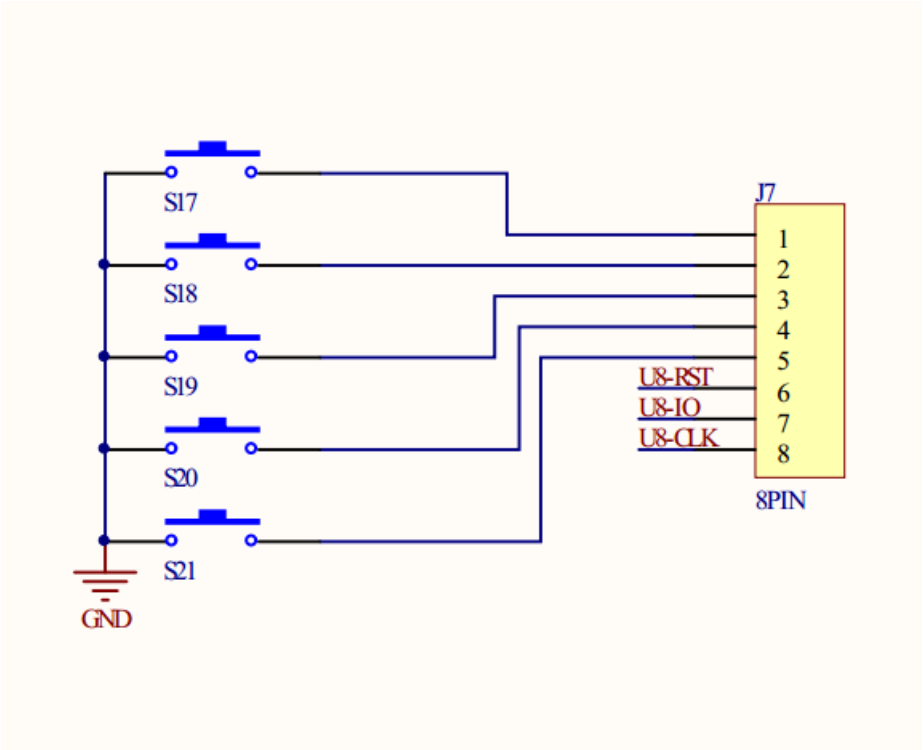


图 5 独立键盘电路图

三、系统的软件设计实现

3.1 程序流程框图

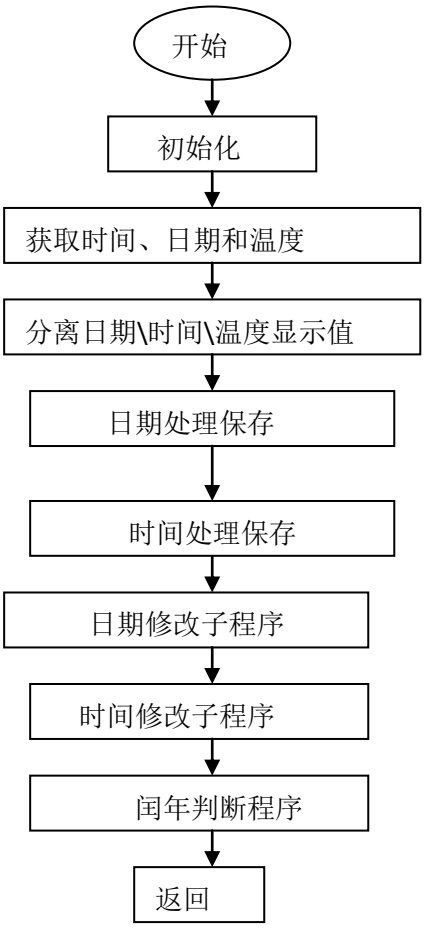


图-A 主程序流程图

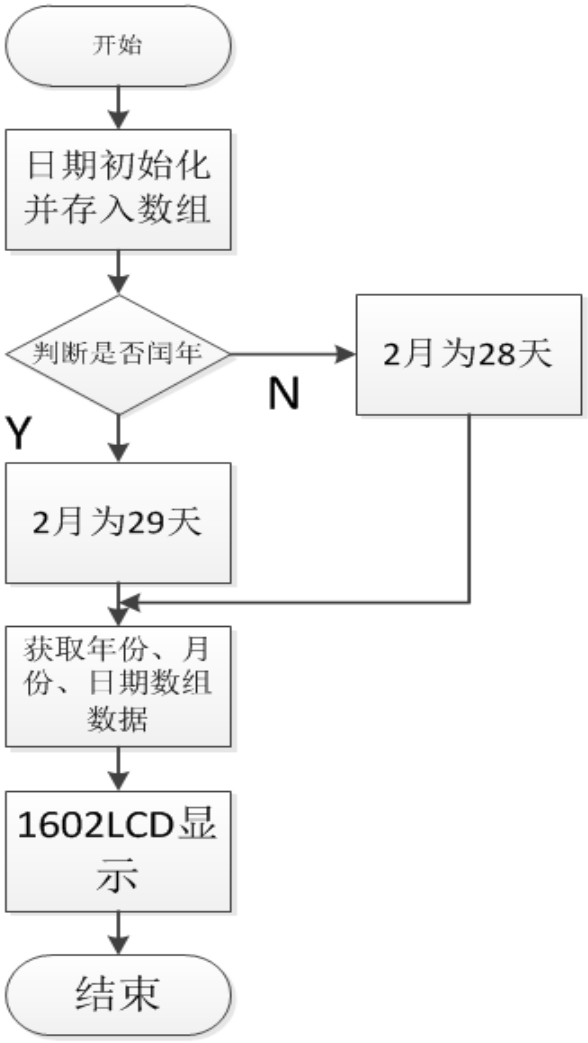


图-B 计算日期程序流程图

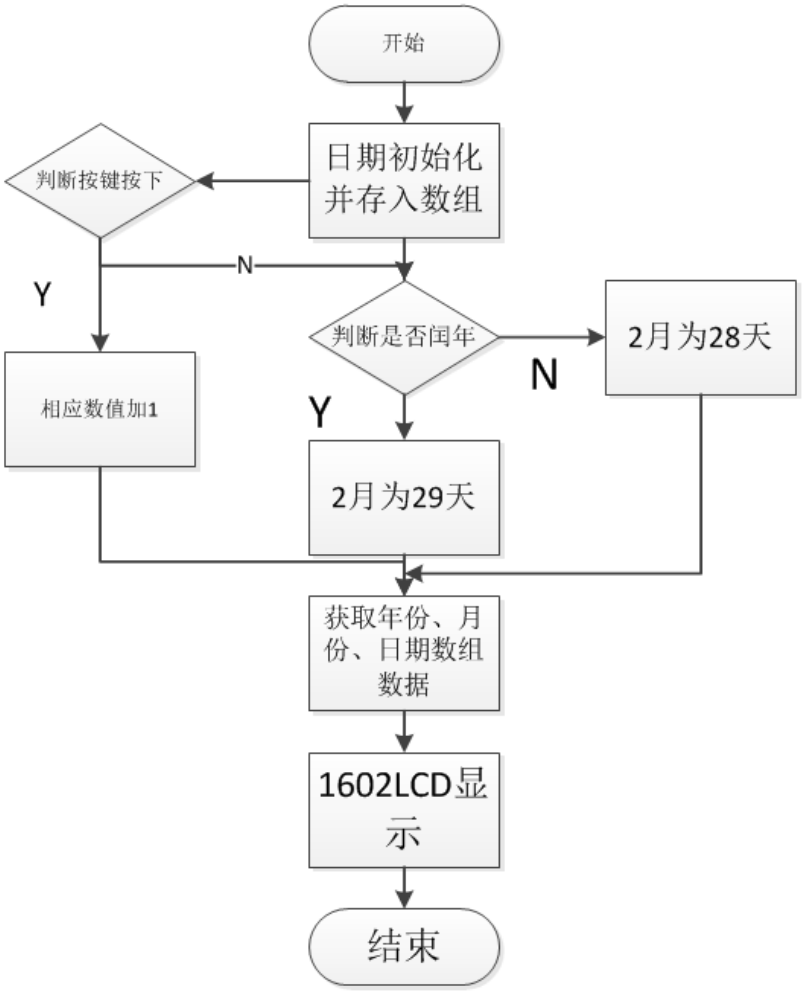


图-C 时间日期调整程序流程图

注：以上只列出了日期调整的程序流程，时间的获取和调整和时间类似，不再赘述。

3.2 子程序的设计

3.2.1 DS18B20 温度子程序代码

```
//*************************************************************************
**********************
//传感器初始化
//*************************************************************************
```

```
**********************
init_18b20()
{
        uchar flag;
        DQ=1;
        delay(10);   //延时
        DQ=0;
        delay(2000); //*延时，要求精度，要求大于 480us*
        DQ=1;
        delay(200); //*延时，要求精度，要求大于 15us*
        flag=DQ;      //DQ 管脚送出 60-240us 的 0 脉冲,以示初始化成功
        delay(10);   //延时
}
//*************************************************************************
**********************
//写一个字节函数
//*************************************************************************
**********************
write_byte(uchar t)
{
 uchar i;
 for(i=0;i<8;i++)     //循环 8 次写入 1 字节
  {
    DQ=0;               //数据线置低
    delay(20);         //延时
    DQ=t&0x01;          //发送 1 位数据,最低位开始
    delay(150);         //*延时，要求精度*
    DQ=1;               //数据线置高
    t=t>>1;             //右移 1 位
  }
}
//*************************************************************************
**********************
//读一个字节函数
//*************************************************************************
**********************
uchar read_byte()
{
 uchar i,value=0;;
 for(i=0;i<8;i++)     //循环 8 次读取 1 字节
  {
    value=value>>1; //右移 1 位
    DQ=0;               //数据线置低
    delay(10);         //延时
```

```
    DQ=1;                //数据线置高
    delay(10);          //延时
    if(DQ==1)value=value|0x80;//判断接收的 1 位数据是否为 1
    delay(50);          //*延时，要求精度*
  }
 return(value);
}
//*************************************************************************
*********************
//数据处理子函数
//*************************************************************************
*********************
chuli(uint temperature)
{
 float t;
 if(temperature&0x8000)              //判断是否为负数
   {
     temperature=~temperature+1;//取反加 1
     dis1[9]=0xb0;                    //显示负号
   }
 else
   {
     dis1[9]=0x2b;                    //显示正号
   }
 t=temperature*0.0625+0.05;        //计算出温度值，百分位四舍五入

 temperature=t*10;                   //本实验显示到小数点后 1 位,所以乘 10，以便分离得到
十分位

 dis1[13]=temperature%10+0x30;       //除 10 取余得温度十分位，1602 只识别 ASCII 码，+0x30
目的就是把 16 进制转 ASCII
 dis1[10]=temperature/100+0x30;     //除 100 取整得温度十位
 dis1[11]=temperature%100/10+0x30;//除 100 取余得十位和个位，然后除 10 取整得温度个位
}
//*************************************************************************
*********************
//温度采集函数
//*************************************************************************
*********************
uint get_temp()
{
    uint dat;
    uchar wenl,wenh;
    init_18b20();              //复位
```

```
    write_byte(0xcc);        //不进行编号匹配
    write_byte(0x44);        //进行温度转换
    init_18b20();            //复位
    write_byte(0xcc);        //不进行编号匹配
    write_byte(0xbe);        //发读命令
    wenl=read_byte();        //温度低八位
    wenh=read_byte();        //温度高八位
    dat=(wenh<<8)+wenl;     //数据高低 8 位合并
    return(dat);             //返回测量结果
}void timer0_init(){         //定时器 T0 初始化
    TMOD=0x61;
    TH0=0x03c;
    TL0=0x0b0;
    TH1=0x0f6;
    TL1=0x0f6;
    TR0=1;
    TR1=1;

    ET0=1;
    ET1=1;
    EA=1;

}
```

3.2.2 温度模块代码

```
//*************************************************************************************
*********************
//传感器初始化
//*************************************************************************************
*********************
init_18b20()
{
        uchar flag;
        DQ=1;
        delay(10);   //延时
        DQ=0;
        delay(2000); //*延时，要求精度，要求大于 480us*
```

```
        DQ=1;
        delay(200); //*延时，要求精度，要求大于 15us*
        flag=DQ;      //DQ 管脚送出 60-240us 的 0 脉冲,以示初始化成功
        delay(10);   //延时
}
//*****************************************************************************
*********************
//写一个字节函数
//*****************************************************************************
*********************
write_byte(uchar t)
{
 uchar i;
 for(i=0;i<8;i++)     //循环 8 次写入 1 字节
  {
     DQ=0;               //数据线置低
     delay(20);         //延时
     DQ=t&0x01;          //发送 1 位数据,最低位开始
     delay(150);         //*延时，要求精度*
     DQ=1;               //数据线置高
     t=t>>1;             //右移 1 位
  }
}
//*****************************************************************************
*********************
//读一个字节函数
//*****************************************************************************
*********************
uchar read_byte()
{
 uchar i,value=0;;
 for(i=0;i<8;i++)     //循环 8 次读取 1 字节
  {
     value=value>>1; //右移 1 位
     DQ=0;               //数据线置低
     delay(10);         //延时
     DQ=1;               //数据线置高
     delay(10);         //延时
     if(DQ==1)value=value|0x80;//判断接收的 1 位数据是否为 1
     delay(50);         //*延时，要求精度*
  }
 return(value);
}
//*****************************************************************************
```

```
**********
//数据处理子函数
//*****************************************************************************
**********
chuli(uint temperature)
{
 float t;
 if(temperature&0x8000)              //判断是否为负数
   {
     temperature=~temperature+1;//取反加 1
     dis1[9]=0xb0;                    //显示负号
   }
 else
   {
     dis1[9]=0x2b;                    //显示正号
   }
 t=temperature*0.0625+0.05;        //计算出温度值，百分位四舍五入

 temperature=t*10;                   //本实验显示到小数点后 1 位,所以乘 10，以便分离得到
十分位

 dis1[13]=temperature%10+0x30;       //除 10 取余得温度十分位，1602 只识别 ASCII 码，+0x30
目的就是把 16 进制转 ASCII
 dis1[10]=temperature/100+0x30;     //除 100 取整得温度十位
 dis1[11]=temperature%100/10+0x30;//除 100 取余得十位和个位，然后除 10 取整得温度个位
}
//*****************************************************************************
**********
//温度采集函数
//*****************************************************************************
**********
uint get_temp()
{
    uint dat;
    uchar wenl,wenh;
    init_18b20();              //复位
    write_byte(0xcc);         //不进行编号匹配
    write_byte(0x44);         //进行温度转换
    init_18b20();              //复位
    write_byte(0xcc);         //不进行编号匹配
    write_byte(0xbe);         //发读命令
    wenl=read_byte();         //温度低八位
    wenh=read_byte();          //温度高八位
    dat=(wenh<<8)+wenl;      //数据高低 8 位合并
```

```
    return(dat);                //返回测量结果
}void timer0_init(){        //定时器 T0 初始化
    TMOD=0x61;
    TH0=0x03c;
    TL0=0x0b0;
    TH1=0x0f6;
    TL1=0x0f6;
    TR0=1;
    TR1=1;

    ET0=1;
    ET1=1;
    EA=1;

}
```

3.2.3 时间和日期模块代码

```
void Handle() //时间处理、获取函数
{
{   if(s17==0)
    {       delay1(80);
            if(s17==0)
           {
           min++;
           if(min==60){min=0;hour++;}
           }
    }
    if(s18==0)
       {   delay1(80);
                if(s18==0)
               hour++;
               if(hour==24){hour=0;day++;}
       }
       if(sec==60)
       {
      sec=0;
      min++;
      if(min==60)
      {
        min=0;
```

```
            hour++;
          if(hour==24){hour=0;day++;}
          }
      }
}
dis1[7]=sec%10+0x30; //将得到的时间数值转换成 ascII 码让 1602 识别
dis1[6]=sec/10+0x30;

dis1[4]=min%10+0x30;
dis1[3]=min/10+0x30;

dis1[1]=hour%10+0x30;
dis1[0]=hour/10+0x30;
}
void Handlemonth()      //闰年判断和每月的天数判断
{
if(month==1||month==3||month==5||month==7||month==8||month==10||month==12)//31
天的月份
Dayx=31;
if(month==4||month==6||month==9||month==11)//30 天的月份
Dayx=30;
if((month==2&&year/4==0&&year/100!=0)||year==2000)//闰年 2 月天数
Dayx=29;
if(month==2&&(year/4!=0||year/100==0)&&year!=2000) //非闰年 2 月的天数
Dayx=28;
}
void HandleDate()//日期处理函数 1901-01-01 0 1 2 3 5 6 8 9
{
  if(s19==0)
      {         delay1(80);
                if(s19==0)
               {
               year++;
               if(year==2100){year=1901;}
               }
      }
      if(s20==0)
      {         delay1(80);
                if(s20==0)
               {
               month++;
               if(month==13){month=1;year++;}
               }
      }
```

```
    if(s21==0)
    {       delay1(80);
            if(s21==0)
           {
           day++;
           if(day==Dayx+1){day=1;month++;}
           }
    }//按键调整日期模块
           if(day==Dayx+1)
   {
           day=1;month++;
           if(month==13)
           {
           month=1;
           year++;
           if(year==2100)
           year=1901;
           }

  }//日期进位模块
  dis2[3]=year/1000+0x30;
  dis2[4]=(year/100)%10+0x30;
  dis2[5]=(year/10)%10+0x30;
  dis2[6]=((year%1000)%100)%10+0x30;
  dis2[8]=month/10+0x30;
  dis2[9]=month%10+0x30;
  dis2[11]=day/10+0x30;
  dis2[12]=day%10+0x30;//日期保存进数组
  }
```

void t0() interrupt 1//计数器 0 内部定时 65636 机器周期中断一次使 p36（50ms 变反一次,100ms 一个周期)

```
{
    TF0=0;
    TH0=0x03c;
    TL0=0x0b0;
    p36=~p36;
}
```

void t1() interrupt 3//定时器 1 外部计数，记满 10 个外部脉冲计数器加 1，并输出（1 秒加 1）（P3.6?P3.5）

```
{

    TF1=0;
    sec++;
```

```
}
```

3.2.4LCD 模块代码

```
//*************************************************************************************
*********************
//查忙
//*************************************************************************************
*********************
checkbusy()
{
    P0= 0xFF;                  //单片机 I/O 口设置为输入
    rs = 0;                   //命令/数据选择,为 0 时选择命令
    rw = 1;                    //读/写选择，为 1 时选择读
    e = 0;
    e = 1;                     //使能
    while (BusyFlag== 1); //查忙标志位，等待标志位为 0，即表示写入完毕
    e= 0;                      //关闭读写
}
//*************************************************************************************
*********************
//向 LCD 写一命令
//*************************************************************************************
*********************
wcode(uchar t)
{
  checkbusy();      //查忙
  rs=0;              //  写的是命令
  rw=0;               //  写状态
  e=1;                //使能
  P0=t;               //写入命令
  delay2(20);        //等待写入,如果时间太短，会导致液晶无法显示
  e=0;                //数据的锁定
}
//*************************************************************************************
*********************
//向 LCD 写一数据
//*************************************************************************************
*********************
wdata(uchar t)
{
```

```
   checkbusy();     //查忙
   rs=1;              // 写的是数据
   rw=0;               // 写状态
   e=1;                //使能
   P0=t;               //写入数据
   delay2(20);        //等待写入,如果时间太短，会导致液晶无法显示
   e=0;                //数据的锁定
}
//*****************************************************************************
**********************
//LCD 显示第一行
//*****************************************************************************
**********************
xian1()
{
   uchar i;
   wcode(0x80);               //设置第一行显示地址
   for(i=0;i<16;i++)        //循环 16 次，写完 1 行
     {
        wdata(dis1[i]);     //写入该行数据
//delay(10000);
     }
}
//*****************************************************************************
**********************
//LCD 显示第二行
//*****************************************************************************
**********************
xian2()
{
    uchar i;
    wcode(0xc0);             //设置第二行显示地址
    for(i=0;i<16;i++)     //循环 16 次，写完 1 行
     {
        wdata(dis2[i]);   //写入该行数据
     }
}
//*****************************************************************************
**********************
//LCD  初始化
//*****************************************************************************
**********************
InitLCD()
    {
```

```
    wcode(0x01);      //清屏
    wcode(0x06);     //输入方式控制,增量光标不移位
    wcode(0x0e);     //显示开关控制
    wcode(0x38);     //功能设定:设置 16x2 显示，5x7 显示,8 位数据接口    38
    }
```

四、成品调试

走时准确性的调试：对于每次跳到 60 秒时的数据变动问题，我将时间设置为从 0 开始重新计数，符合钟表的走时标准，一天走时完成之后，日期加 1.经过 24 小时的不间断走时，误差时间不到 1 秒，说明成品的走时准确性还是比较高的。
日期的调试：调试过程中，年份范围从 1901~2100，对该年段范围内的闰年进行判断，然后确定 2 月份的天数。在编程过程中，犯了一个错误，将日期从 0 开始计数，最后一天不算入天数，导致日期显示错误，经过改正修复了错误。测试了所有的年份和月份，结果完全符合日历上的日期标准。
温度模块调试：温度传感器可以实时动态显示当前温度，通过用手按住温度传感器，可以发现 LCD 显示温度不断升高，放开手后，温度缓慢下降，说明温度模块工作正常。
LCD 显示模块调试：LCD 显示时，需要选择适当的延时时间，使得屏幕显示正常，经过调试之后，屏幕显示稳定，没有重影，也没有闪烁，能够正确显示数值，说明 LCD 工作正常。

五、总结

在整个设计过程中，充分发挥人的主观能动性，自主学习，学到了许多没学到的知识。较好的完成了作品。达到了预期的目的，在最初的设计中，自主学习研究，请教他人。完成最初的设想。由于在开发板上完成，故电路已经焊接完成，不需要重复焊接，但对开发板电路需要比较熟悉，找到正确的连线方式。对电路的设计、布局要先有一个好的构思，才显得电路板美观、大方、逻辑清晰。程序编写中，由于思路不清晰，开始时遇到了很多的问题，经过静下心来思考，不断调试，理清了思路，反而得心应手。在此次设计中，知道了做凡事要有一颗平常的心，不要想着走捷径，一步一脚印。也练就了我们的耐心，做什么事都在有耐心。这是最重要的。

附录 1

主函数和延时函数模块：

```
/**********************************
      延时函数和主函数模块
**********************************/
```

```
//*****************************************************************************
**********************
//精确延时函数(使用 12M 晶振，延时 T=2*t+5 个指令周期,t 为 char 型,改为 int 型,会加大误
差）
//*****************************************************************************
**********************
void delay(uint t)
{
  while(--t);
}
//*****************************************************************************
**********************
//延时函数
//*****************************************************************************
**********************
delay1(uint time)                //int 型数据为 16 位,所以最大值为 65535
 {
   uint    i,j;                    //定义变量 i,j,用于循环语句
   for(i=0;i<time;i++)       //for 循环,循环 50*time 次
       for(j=0;j<120;j++); //for 循环,循环 50 次
 }
//*****************************************************************************
**********************
//延时函数
//*****************************************************************************
**********************
delay2(uint time)                //int 型数据为 16 位,所以最大值为 65535
 {
   uint    i,j;                    //定义变量 i,j,用于循环语句
   for(i=0;i<time;i++)       //for 循环,循环 50*time 次
       for(j=0;j<100;j++); //for 循环,循环 50 次
 }
//*****************************************************************************
**********************
//主函数
//*****************************************************************************
**********************
main()
{
    uint temp;          //临时变量保存温度值
    InitLCD();        //初始化 1602
    InitLCD();          //液晶初始化
    timer0_init();    //定时器初始化
       while(1)                      //死循环
```

```
     {
    Handle();           //  时间数据处理
    Handlemonth();//闰年判断和每月的天数判断
    HandleDate();              //日期处理函数 1901-01-01 0 1 2 3 5 6 8 9
    temp=get_temp();     //获取温度值
    chuli(temp);           //温度数据处理
    xian1();               //显示第一行
    xian2();               //显示第二行
    delay1(40);              //延时
    }
}
```

附录 2

头文件定义和声明：

```
#include "reg51.h"                 //包含头文件
#define uchar unsigned char
#define uint    unsigned int
uchar                  data                                  dis1[16]={0x00,0x00,':',0x00,0x00,':',0x00,0x00,'
',0x00,0x00,0x00,'.',0x00,0xeb,'C'};//LCD 第一行显示当前时间和温度
uchar data    dis2[16]={' ',' ',' ',0x00,0x00,0x00,0x00,'-',0x00,0x00,'-',0x00,0x00,' ',' ',' ',};//LCD 第 2
行显示日期
init_18b20();
write_byte(uchar t);
uchar read_byte();
chuli(uint temperature);
uint get_temp();
void timer0_init();
void Handle();
checkbusy();
wcode(uchar t);
wdata(uchar t);
xian1();
xian2();
InitLCD();
void delay(uint t);
delay1(uint time);
delay2(uint time);
void HandleDate();//日期处理函数 1901-01-01 0 1 2 3 5 6 8 9
void Handlemonth();     //闰年判断和每月的天数判断
sbit rs=P2^5;       //命令/数据选择
sbit rw=P2^4;       //读写口
sbit    e=P2^3;       //锁存控制
```

```
sbit BusyFlag=P0^7;//查忙标志位
sbit p36=P3^6;
uint hour=0,min=0,sec=0;
uint Dayx;
uint year=1901,month=01,day=01;
sbit DQ=P2^7;       //温度控制口
sbit s17=P3^0;//分调整按键
sbit s18=P3^1;//时调整按键
sbit s19=P3^2;//年调整按键
sbit s20=P3^3; //月调整按键
sbit s21=P3^4;    //日调整按键
```