实验四十四　　红外测温仪实验

一．实验目的

1．通过本实验掌握 MLX90614 的原理。

2．通过实验掌握数码管基本原理

3．通过实验掌握矩阵键盘的使用

4．通过实验掌握 1602 的使用

二．实验内容

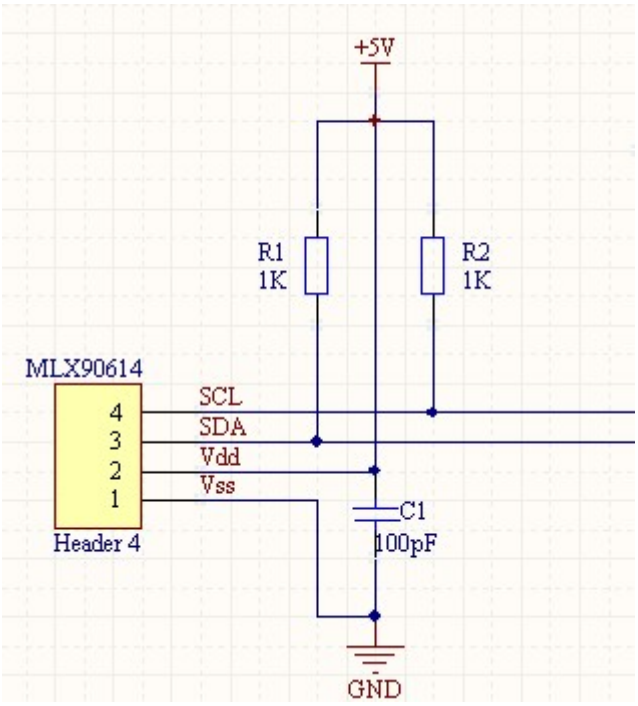
通过 MLX90614 非接触式测得物体温度

三．实验器材

1.主控屏+5V 电源

2. DCP-PRJ09 红外测温仪

四．实验原理图



五．实验步骤

1.接上电源

2.运行 Keil uVision 软件，新建一个工程，新建一个工程文件。将文件添加到工程中并编译,如有错，请更改直到编译成功,如有错，请更改直到编译成功

4.用编程器将生成的 HEX 文件烧写到单片机中，或用 STC 单片机专用的串口烧写软件，通过 MAX232 串口烧写到单片机中（只能用于 STC 单片机）。或用仿真器来执行程序，将程序下载到仿真器中，具体方法请参考仿真器的使用一节。用编程器将生成的 HEX 文件烧写到单片机中。

5. 程序下完后可以看到液晶屏有温度显示，用手掌覆盖红外传感器上方 2cm 处，看到此时液晶屏温度即为手掌的表面温度。

6. 按下 1 键可以看到数码管显示，用手掌覆盖红外传感器上方 2cm 处，看到此时数码管温度即为手掌的表面温度。此时液晶屏温度保持不变。

7. 按下除 1 键其他键，此时液晶屏又重新显示温度。

实验代码如下：

```
//--------------------------------------------------------------------------
//说明：显示分为 2 部分：数码管显示，液晶屏显示
//按 1 键数码管显示，此时液晶屏保持静止
//复位时或按除 1 键外其他键时，液晶屏显示，数码管熄灭
//--------------------------------------------------------------------------
#include<reg52.h>
#include<intrins.h>
#define uint unsigned int
#define uchar unsigned char
#define Nack_number 10
//**************端口定义*****************************************************
uchar flag;                                    //LCD 控制线接口
sbit RS=P0^6;                                  //RS 端
sbit RW=P0^7;                                  //读写端
sbit LCDE=P3^5;                                //使能端
                                               //mlx90614 端口定义
```

```
sbit SCK=P3^7;                                  //时钟线
sbit SDA=P3^6;                                  //数据线


sbit DPY1 = P3^2;                               //温度显示第 1 个数码管段选
sbit DPY2 = P3^3;                               //温度显示第 2 个数码管段选
sbit DPY3 = P3^4;                               //温度显示第 3 个数码管段选

sbit row1 = P0^3;                               //矩阵键盘第 1 列
sbit row2 = P0^4;                               //矩阵键盘第 2 列
sbit row3 = P0^5;                               //矩阵键盘第 3 列
sbit cow1 = P0^0;                               //矩阵键盘第 1 行
sbit cow2 = P0^1;                               //矩阵键盘第 2 行
sbit cow3 = P0^2;                               //矩阵键盘第 3 行
//**********数据定义**************************************************
bdata uchar flag1;                              //可位寻址数据
sbit bit_out=flag1^7;
sbit bit_in=flag1^0;
uchar tempH,tempL,err;


void CALTEMP(uint TEMP);
void ReadKey(void);
void initInt();
void delay1(uint z);
void show();
```

```
uchar key_num;
uchar mah[5];

/********数码管码值定义**********************************************************/
uchar code LED01[]={                    //LED 显示代码，0-9 共阳  不带小数点的
0xC0,0xF9,0xA4,0xB0,
0x99,0x92,0x82,0xF8,
0x80,0x90};

uchar code LED02[]={                    //LED 显示代码，0-9 共阳  带小数点的
0x40,0x79,0x24,0x30,
0x19,0x12,0x02,0x78,
0x00,0x10};
/**********全局变量定义**********************************************************/
bit b20ms,b100ms;    //定时标志位
uchar c20ms,c100ms;       //定时毫秒数

//*************************  LCD1602  ***********************************
//向 LCD 写入命令或数据*************************************************************
#define LCD_COMMAND           0                 //命令
#define LCD_DATA              1                 // 数据
#define LCD_CLEAR_SCREEN      0x01              // 清屏
#define LCD_HOMING            0x02              // 光标返回原点
//设置显示模式******* 0x08+    ****************************************************
#define LCD_SHOW              0x04          //显示开
```

```
#define LCD_HIDE              0x00          //显示关
#define LCD_CURSOR            0x02          //显示光标
#define LCD_NO_CURSOR         0x00          //无光标
#define LCD_FLASH             0x01          //光标闪动
#define LCD_NO_FLASH          0x00          //光标不闪动
//设置输入模式********** 0x04+ ****************************************
#define LCD_AC_UP             0x02          //光标右移 AC+
#define LCD_AC_DOWN           0x00          //默认 光标左移 AC-
#define LCD_MOVE              0x01          //画面可平移
#define LCD_NO_MOVE           0x00          //默认 画面不移动

//************************* mlx90614 **********************************
//command mode  命令模式
#define RamAccess     0x00                  //对 RAM 操作
#define EepomAccess   0x20                  //对 EEPRAM 操作
#define  Mode         0x60                  //进入命令模式
#define ExitMode  0x61                     //退出命令模式
#define  ReadFlag 0xf0                     //读标志
#define EnterSleep    0xff                  //进入睡眠模式
//ram address     read only RAM 地址（只读）
#define AbmientTempAddr       0x03          //周围温度
#define  IR1Addr              0x04
#define  IR2Addr              0x05
#define LineAbmientTempAddr   0x06          //环境温度
/*0x0000 0x4074    16500   0.01/单元
```

```
        -40      125*/
#define LineObj1TempAddr 0x07            //目标温度,红外温度
/*0x27ad-0x7fff      0x3559 22610      0.02/单元
   -70.01-382.19    0.01     452.2*/
#define   LineObj2TempAddr      0x08
//eepom address   EEPROM 地址
#define TObjMaxAddr         0x00              //测量范围上限设定
#define TObjMinAddr         0x01              //测量范围下限设定
#define PWMCtrlAddr         0x02              //PWM 设定
#define TaRangeAddr         0x03              //环境温度设定
#define KeAddr              0x04              //频率修正系数
#define ConfigAddr          0x05              //配置寄存器
#define SMbusAddr           0x0e              //器件地址设定
#define Reserverd1Addr      0x0f              //保留
#define Reserverd2Addr      0x19              //保留
#define ID1Addr             0x1c              //ID 地址 1
#define ID2Addr             0x1d                    //ID 地址 2
#define ID3Addr             0x1e              //ID 地址 3
#define ID4Addr             0x1f              //ID 地址 4

//************函数声明*****************************************************
void start();                                 //MLX90614 发起始位子程序
void stop();                                  //MLX90614 发结束位子程序
uchar ReadByte(void);                         //MLX90614 接收字节子程序
void send_bit(void);                          //MLX90614 发送位子程序
```

```
void SendByte(uchar number);                    //MLX90614 接收字节子程序
void read_bit(void);                            //MLX90614 接收位子程序
void delay(uint N);                             //延时程序
uint readtemp(void);                            //读温度数据
void init1602(void);                            //LCD 初始化子程序
void busy(void);                                //LCD 判断忙子程序
void cmd_wrt(uchar cmd);                        //LCD 写命令子程序
void dat_wrt(uchar dat);                        //LCD 写数据子程序
void display(uint Tem);                         //显示子程序
void Print(uchar *str);                       //字符串显示程序


//*************主函数*******************************************
void main()
{
    uint Tem;                                       //温度变量
    initInt();
    SCK=1;
    SDA=1;
    delay(4);
    SCK=0;
    delay(1000);
    SCK=1;
    init1602();                                     //初始化 LCD
    while(1)
    {
```

```
            while(b100ms)                       //每 100ms 扫描一次键盘
            {
             b100ms=0;
              ReadKey();
            }

            if(key_num==1)                      //按下 1 键时，进行数码管显示
            {
             Tem=readtemp();
             CALTEMP(Tem);
             show();
            }

            if(key_num!=1)          //液晶屏显示
            {
             Tem=readtemp();                            //读取温度
             cmd_wrt(0x01);                             //清屏
             Print("   Temperature:     ");          //显示字符串   Temperature: 且换行
             display(Tem);                              //显示温度
             Print(" ^C");                          //显示摄氏度
             delay(100000);                             //延时再读取温度显示
            }
        }
}
//------------------字符串显示程序-------------------------
```

```
void Print(uchar *str)                          //字符串显示程序
{
    while(*str!='\0')                           //直到字符串结束
    {
        dat_wrt(*str);                          //转成 ASCII 码
        str++;                                  //指向下一个字符
    }
}

//--------------输入转换并显示(用于 LCD1602)--------------------
void display(uint Tem)
{
    uint T,a,b;
    T=Tem*2;
    if(T>=27315)                                //温度为正
    {
        T=T-27315;                              //
        a=T/100;                                //温度整数
        b=T-a*100;                              //温度小数
        if(a>=100)                              //温度超过 100 度
        {
            dat_wrt(0x30+a/100);                //显示温度百位
            dat_wrt(0x30+a%100/10);             //显示温度十位
            dat_wrt(0x30+a%10);                 //显示温度个位
        }
```

```
        else if(a>=10)                          //温度超过 10 度
        {
            dat_wrt(0x30+a%100/10);             //显示温度十位
            dat_wrt(0x30+a%10);                 //显示温度个位
        }
        else                                    //温度不超过 10 度
        {
            dat_wrt(0x30+a);                    //显示温度个位
        }
        dat_wrt(0x2e);                          //显示小数点
        if(b>=10)                               //温度小数点后第 1 位数不等于 0
        {
        dat_wrt(0x30+b/10);                     //显示温度小数点后第 1 位数
        dat_wrt(0x30+b%10);                     //显示温度小数点后第 2 位数
        }
        else                                    //温度小数点后第 1 位数等于 0
        {
            dat_wrt(0x30);                      //显示温度小数点后第 1 位数 0
            dat_wrt(0x30+b);                    //显示温度小数点后第 2 位数
        }
    }
    else                                        //温度为负
    {
        T=27315-T;
        a=T/100;
```

```
            b=T-a*100;
            dat_wrt(0x2d);                          //显示负号
            if(a>=10)                               //温度低于负 10 度
            {
                dat_wrt(0x30+a/10);                 //显示温度十位
                dat_wrt(0x30+a%10);                 //显示温度个位
            }
            else                                    //温度高于负 10 度
            {
                dat_wrt(0x30+a);                    //显示温度个位
            }
            dat_wrt(0x2e);                          //显示小数点
            if(b>=10)                               //温度小数点后第 1 位数不等于 0
            {
                dat_wrt(0x30+b/10);                 //显示温度小数点后第 1 位数
                dat_wrt(0x30+b%10);                 //显示温度小数点后第 2 位数
            }
            else                                    //温度小数点后第 1 位数等于 0
            {
                dat_wrt(0x30);                      //显示温度小数点后第 1 位数 0
                dat_wrt(0x30+b);                    //显示温度小数点后第 2 位数
            }
        }
}
//--------------------根据十六进制计算温度-----------------------------
```

```
void CALTEMP(uint TEMP)
{
        uint T;
        uint a,b;
        uchar A4,A5,A6,A7,A8;
        T=TEMP*2;
        if(T>=27315)
                {
                    T=T-27315;
                    a=T/100;
                    b=T-a*100;
                    if(a>=100)
                         {
                            A4=a/100;
                            a=a%100;
                            A5=a/10;
                            a=a%10;
                            A6=a;
                         }
                    else if(a>=10)
                          {
                            A4=0;
                            A5=a/10;
                            a=a%10;
                            A6=a;
```

```
            }
        else
            {
               A4=0;
               A5=0;
               A6=a;
            }
        if(b>=10)
            {
               A7=b/10;
               b=b%10;
               A8=b;
            }
         else
            {
               A7=0;
               A8=b;
            }
    }
else
        {
            T=27315-T;
            a=T/100;
            b=T-a*100;
            A4=9;
```

```
    if(a>=10)
        {
            A5=a/10;
            a=a%10;
            A6=a;
        }
    else
        {
            A5=0;
            A6=a;
        }
    if(b>=10)
        {
            A7=b/10;
            b=b%10;
            A8=b;
        }
    else
        {
            A7=0;
            A8=b;
        }
}
mah[4]=A4;
 mah[3]=A5;
```

```
                    mah[2]=A6;
                    mah[1]=A7;
                    mah[0]=A8;
}
//------------------------------
void start(void)                                //停止条件是 SCK=1 时，SDA 由 1 到 0
{
    SDA=1;
    delay(4);
    SCK=1;
    delay(4);
    SDA=0;
    delay(4);
    SCK=0;
    delay(4);
}
//------------------------------
void stop(void)                                 //停止条件是 SCK=1 时，SDA 由 0 到 1
{
    SCK=0;
    delay(4);
    SDA=0;
    delay(4);
    SCK=1;
    delay(4);
```

```
    SDA=1;
}
//---------发送一个字节---------
void SendByte(uchar number)
{
    uchar i,n,dat;
    n=Nack_number;                          //可以重发次数
    Send_again:
    dat=number;
    for(i=0;i<8;i++)                        //8 位依次发送
    {
        if(dat&0x80)                        //取最高位
        {
            bit_out=1;                      //发 1
        }
        else
        {
            bit_out=0;                      //发 0
        }
        send_bit();                         //发送一个位
        dat=dat<<1;                         //左移一位
    }
    read_bit();                             //接收 1 位  应答信号
    if(bit_in==1)                           //无应答时重发
    {
```

```
            stop();
            if(n!=0)
            {
                n--;                                //可以重发 Nack_number=10 次
                goto Repeat;                        //重发
            }
            else
            {
                goto exit;                              //退出
            }
        }
        else
        {
            goto exit;
        }
Repeat:
        start();                                    //重新开始
        goto Send_again;                            //重发
        exit: ;                                         //退出
}
//-----------发送一个位---------
void send_bit(void)
{
        if(bit_out==1)
        {
```

```
            SDA=1;                                  //发 1
        }
        else
        {
            SDA=0;                                  //发 0
        }
        _nop_();
        SCK=1;                                      //上升沿
        delay(4);delay(4);
        SCK=0;
        delay(4);delay(4);
}
//----------接收一个字节--------
uchar ReadByte(void)
{
        uchar i,dat;
        dat=0;                                      //初值为 0
        for(i=0;i<8;i++)
        {
            dat=dat<<1;                             //左移
            read_bit();                             //接收一位
            if(bit_in==1)
            {
                dat=dat+1;                          //为 1 时对应位加 1
            }
```

```
    }
    SDA=0;                                  //发送应答信号 0
    send_bit();
    return dat;                             //带回接收数据
}
//----------接收一个位----------
void read_bit(void)
{
    SDA=1;                                  //数据端先置 1
    bit_in=1;
    SCK=1;                                  //上升沿
    delay(4);delay(4);
    bit_in=SDA;                             //读数据
    _nop_();
    SCK=0;
    delay(4);delay(4);
}



//------------------------------
uint readtemp(void)
{
    SCK=0;
    start();                                //开始条件
    SendByte(0x00);                         //发送从地址 00
```

```
    SendByte(0x07);                    //发送命令
    start();                           //开始条件
    SendByte(0x01);                    //读从地址 00
    bit_out=0;
    tempL=ReadByte();                  //读数据低字节
    bit_out=0;
    tempH=ReadByte();                  //读数据高字节
    bit_out=1;
    err=ReadByte();                    //读错误信息码
    stop();                            //停止条件
    return(tempH*256+tempL);
}
//******************LCD 显示子函数**********************
void init1602(void)                    //初始化 LCD
{
    cmd_wrt(0x01);                     //清屏
    cmd_wrt(0x0c);                     //开显示，不显示光标，不闪烁
    cmd_wrt(0x06);                     //完成一个字符码传送后，光标左移，显
示不发生移位
    cmd_wrt(0x38);                     //16×2 显示，5×7 点阵，8 位数据接口
}
void busy(void)                        //LCD 忙标志判断
{
    flag=0x80;                         //赋初值 高位为 1 禁止
    while(flag&0x80)                   //读写操作使能位禁止时等待 继续检测
```

```
    {
        P1=0xff;
        RS=0;                                   //指向地址计数器
        RW=1;                                   //读
        LCDE=1;                                 //信号下降沿有效
        flag=P1;                                //读状态位 高位为状态
        LCDE=0;
    }
}
void cmd_wrt(uchar cmd)                         //写命令子函数
{
    LCDE=0;
    busy();                                     //检测 读写操作使能吗
    P1=cmd;                                     //命令
    RS=0;                                       //指向命令计数器
    RW=0;                                       //写
    LCDE=1;                                     //高电平有效
    LCDE=0;
}
void dat_wrt(uchar dat)                         //写数据子函数
{
    busy();                                     //检测 读写操作使能吗
    LCDE=0;
    if(flag==16)
    {
```

```
            RS=0;                                   //指向指令寄存器
            RW=0;                                   //写
            P1=0XC0;                                //指向第二行
            LCDE=1;                                 //高电平有效
            LCDE=0;
        }
        RS=1;                                       //指向数据寄存器
        RW=0;                                       //写
        P1=dat;                                     //写数据
        LCDE=1;                                     //高电平有效
        LCDE=0;
}
//------------延时--------------
void delay(uint n)
{
        uint j;
        for(j=0;j<n;j++)
        {
            _nop_();
        }
}

//------------定时器初始化函数----------------
void initInt()
{
```

```
    TMOD = 0x10;                          //定时器 1 方式 1
    TH1=(65536-1000)/256;                 //定时器 1 设置 1ms 定时
    TL1=(65536-1000)%256;
    EA=1;                                 //开总中断
    ET1 = 1;                              //开定时器 T1 中断
    TR1 = 1;                              //启动定时器 T1
}

//------------定时器中断处理函数--------------------
void timer1handle() interrupt 3       //定时器 3 1ms 中断
{
    TH1=(65536-1000)/256;
    TL1=(65536-1000)%256;
    c20ms++;
    c100ms++;
    if(c20ms >= 20)       //20ms 计时器
    {
        c20ms = 0;
        b20ms = 1;
    }
    if(c100ms >= 50)      //100ms 计时器
    {
        c100ms = 0;
        b100ms = 1;
    }
```

```
}

//--------------------------温度显示函数-----------------------
void show()
{
    DPY1=0;
    P2=LED01[mah[3]];                //转换 8 位数显示，不带小数点的
    delay1(2);
    P2=0xFF;
    DPY1=1;

    DPY2=0;
    P2=LED02[mah[2]];                //转换 8 位数显示，带小数点的
    delay1(2);
    P2=0xFF;
    DPY2=1;

    DPY3=0;
    P2=LED01[mah[1]];                //转换 8 位数显示，不带小数点的
    delay1(2);
    P2=0xFF;
    DPY3=1;
}
```

```
void ReadKey(void)
{
    row1=0;              //矩阵键盘第 1 列,将第一列拉低,扫描是否有按键按下,第一列按键包括:1,4,7
    row2=1;              //矩阵键盘第 2 列
    row3=1;              //矩阵键盘第 3 列
    cow1=1;              //矩阵键盘第 1 行
    cow2=1;              //矩阵键盘第 2 行
    cow3=1;              //矩阵键盘第 3 行
    _nop_();             //延时函数
    if(!(cow1&cow2&cow3))     //如果有键按下,就返回,且判断是那个键值,否则继续扫描下一列
    {
    if(cow1==0)
    key_num=1;
    if(cow2==0)
    key_num=4;
    if(cow3==0)
    key_num=7;
    return;
    }

    row1=1;              //矩阵键盘第 1 列,将第一列拉低,扫描是否有按键按下,第一列按键包括:2,5,8
    row2=0;              //矩阵键盘第 2 列
    row3=1;              //矩阵键盘第 3 列
```

```
    cow1=1;                //矩阵键盘第 1 行
    cow2=1;                //矩阵键盘第 2 行
    cow3=1;                //矩阵键盘第 3 行
    _nop_();               //延时函数
    if(!(cow1&cow2&cow3))     //如果有键按下,就返回,且判断是那个键值,否则继续扫描下一列
    {
    if(cow1==0)
    key_num=2;
    if(cow2==0)
    key_num=5;
    if(cow3==0)
    key_num=8;
    return;
    }

    row1=1;                //矩阵键盘第 1 列,将第一列拉低,扫描是否有按键按下,第一列按键包括:3,6
    row2=1;                //矩阵键盘第 2 列
    row3=0;                //矩阵键盘第 3 列
    cow1=1;                //矩阵键盘第 1 行
    cow2=1;                //矩阵键盘第 2 行
    _nop_();               //延时函数
    if(!(cow1&cow2))  //如果有键按下,就返回,且判断是那个键值,否则继续扫描下一列
    {
    if(cow1==0)
```

```
        key_num=3;
        if(cow2==0)
        key_num=6;
        return;
        }
}
//--------------数码管显示延时函数----------------------------------------------------
void delay1(uint z)
{
   uint x,y;
   for(x=z;x>0;x--)
   for(y=110;y>0;y--);
}
```