微控制器原理课程结业设计

题　　目　基于 GSM 的温度 CO 检测远程报警系统

班　　级　07031401 07031402

完 成 人　2014301805 宋　涵
2014301797 方泓堃
2014301817 邱美晶

指导老师　苏三买

2017 年 1 月 12 日

一、 设计题目

基于 GSM 的温度 CO 检测远程报警系统

二、 设计要求

1、能够采集环境温度和和 CO 浓度，并且通过液晶屏幕实时显示环境温度和 CO 浓度

2、能够设定温度和 CO 浓度伐值，实现远程超伐报警功能

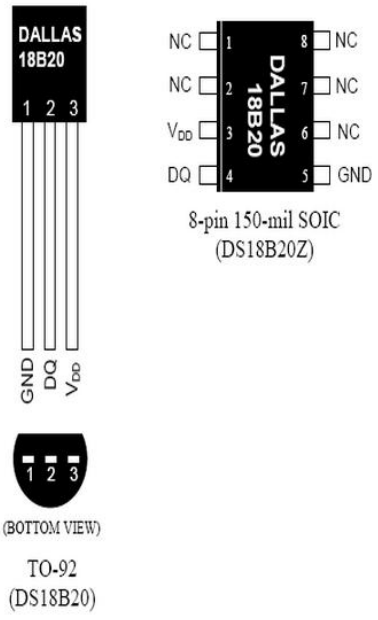
三、 系统硬件设计

1、 硬件设计原理

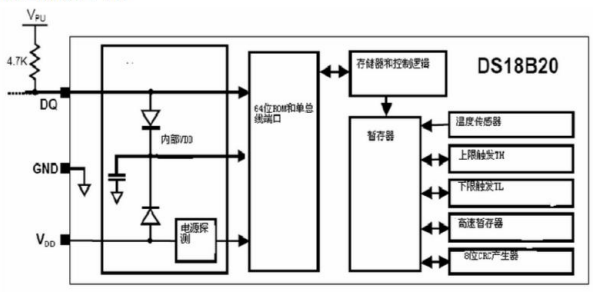
使用的设备有：

单片机 1 台	51 最小系统板一个
温度传感器模块 1 个	安信可 GPS/GSM/GPRS 三合一 A7 模块
CO 传感器模块 1 个	
DC5v 供电电源 1 个	DC5-9V 供电线
下载设备 1 个	SWD 调试下载线

温度采集模块，选用的 DS18B20 温度传感器，DS18B20 通过一个单线接口发送或接受信息，测温范围为-55～＋125℃，并且在-10～＋85℃精度为±5℃。

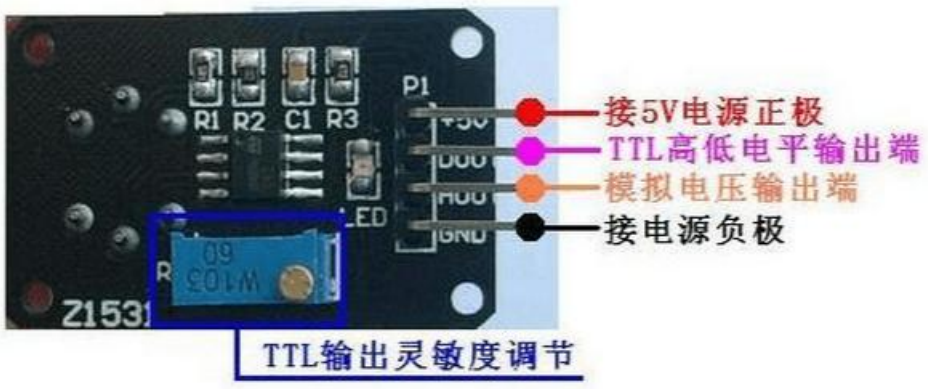


DS18B20方框图 图1

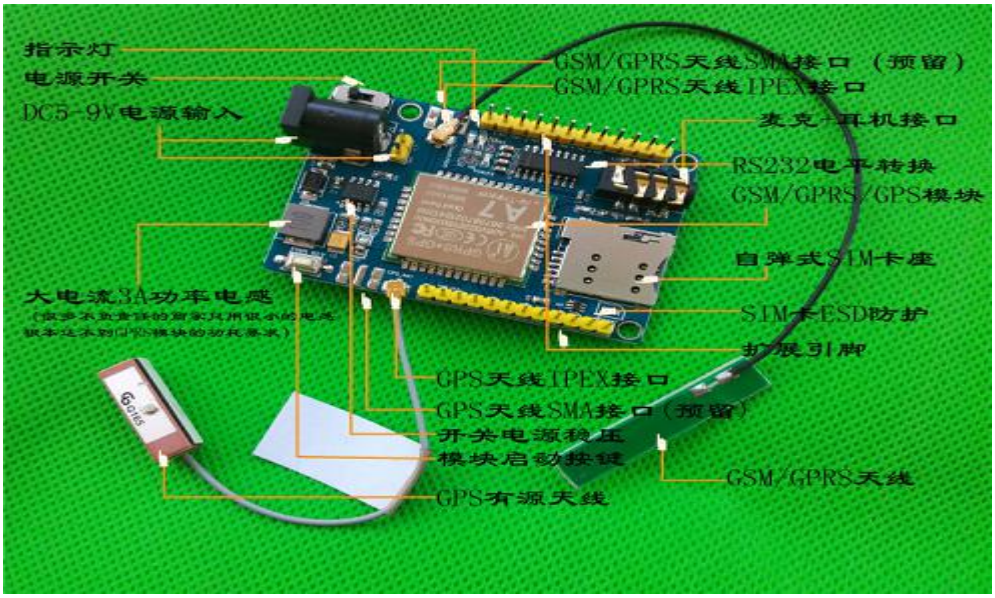


Co 浓度采集模块选用的是，MQ-9 可燃性气体传感器。该型传感器采

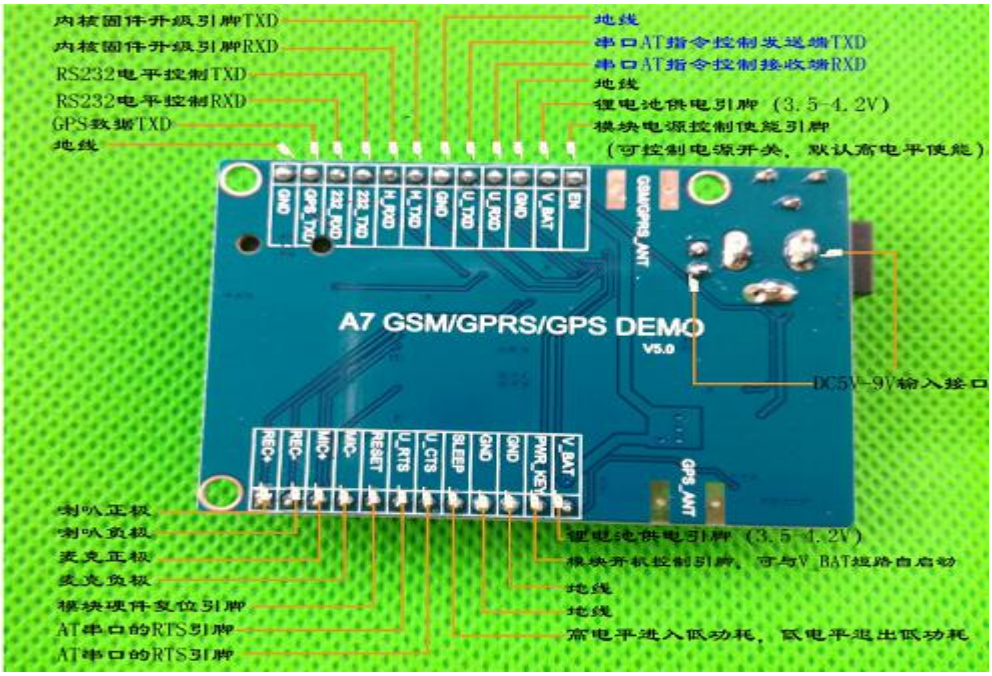
用高低温循环检测方式，传感器电导率岁空气中 CO 浓度增大而增大，Ao 信号端输出一个 0-5v 的模拟信号。其可同时可以检测多种可燃性气体，对于空气中的 CO 灵敏度较高，检测范围可达 10—1000ppm。



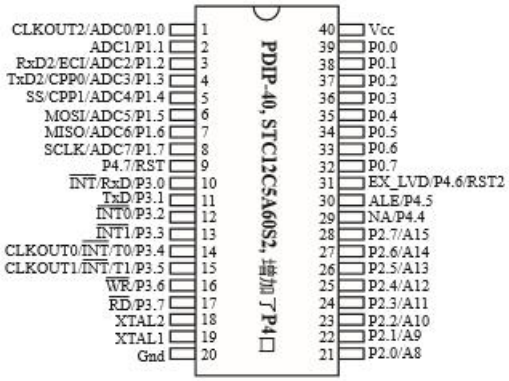
GSM 选用的是，同时集成有 GPS 功能的 A7 模块。



这里我们只用其 GSM 模块功能，将采集到的温度报警信息发送到指定手机号上。



考虑到与 A7 进行通讯需要 115200 的波特率，并且需要 adc 来检测接收到的模拟信号，我们选用的中控芯片型号是 STC12C5A60S2，其自带的辅助分频寄存器 AUXR 和 10 位 ADC 功能刚好满足我们的需要。



3. 辅助寄存器AUXR

STC12C5A60S2系列单片机是 1T 的8051单片机，为兼容传统8051，定时器0和定时器1复位后是传统8051的速度，即12分频，这是为了兼容传统8051。但也可不进行12分频，通过设置新增加的特殊功能寄存器AUXR, 将T0, T1设置为1T。普通111条机器指令是固定的，快3到24倍，无法改变。

AUXR格式如下：

AUXR：辅助寄存器

SFR name	Address	bit	B7	B6	B5	B4	B3	B2	B1	B0
AUXR	8EH	name	T0x12	T1x12	UART_M0x6	BRTR	S2SMOD	BRTx12	EXTRAM	S1BRS

T0x12：定时器0速度控制位。

0：定时器0速度是8051单片机定时器的速度，即12分频；

1：定时器0速度是8051单片机定时器速度的12倍，即不分频。

T1x12：定时器1速度控制位。

0：定时器1速度是8051单片机定时器的速度，即12分频；

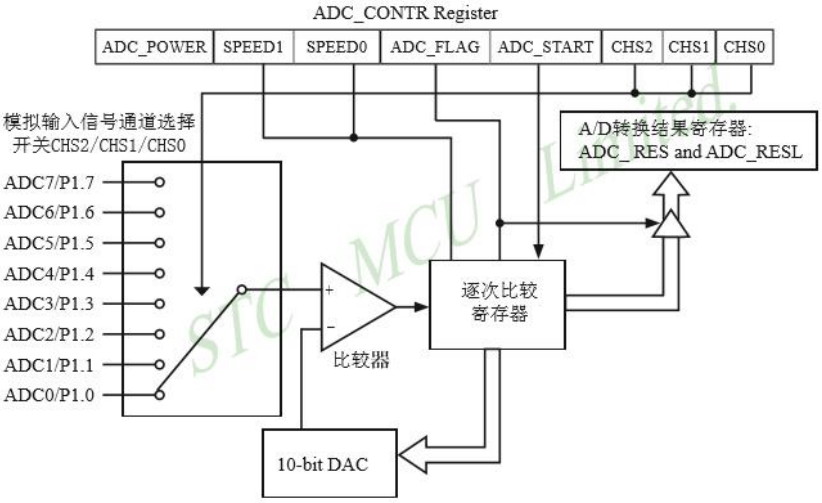
1：定时器1速度是8051单片机定时器速度的12倍，即不分频。

如果UART串口用T1作为波特率发生器，则由T1x12位决定UART串口是12T还是1T。

9.1 A/D转换器的结构

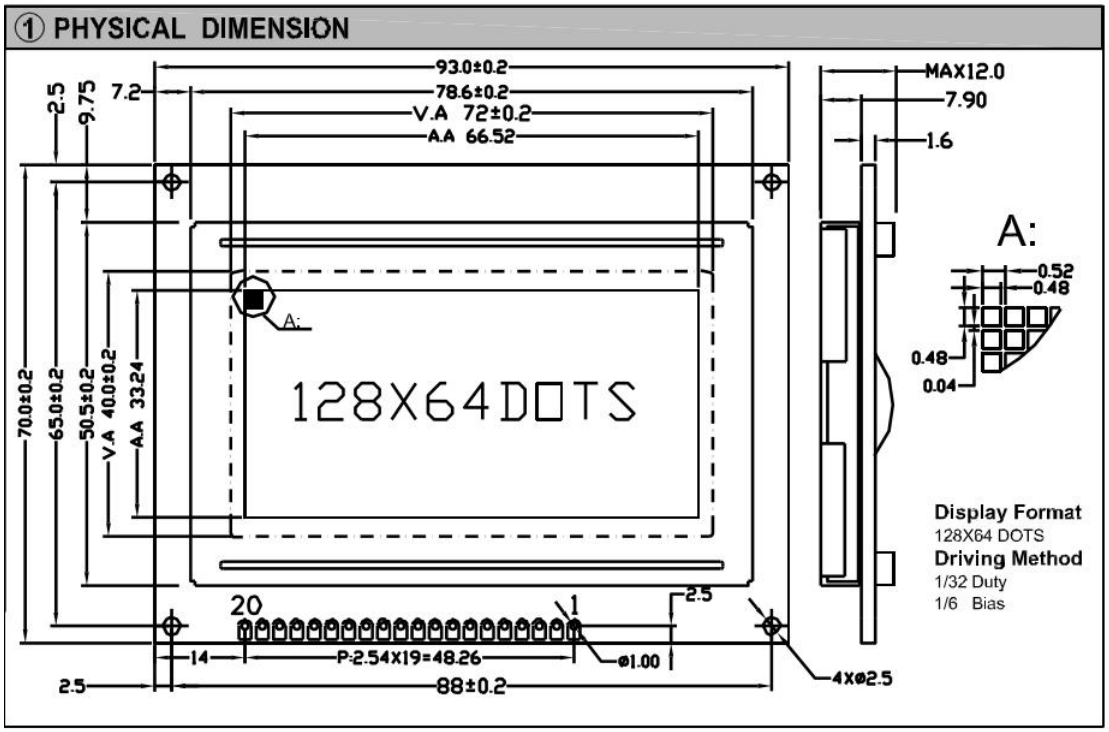
STC12C5A60AD/S2系列带A/D转换的单片机的A/D转换口在P1口(P1.7-P1.0)，有8路10位高速A/D转换器，速度可达到250KHz(25万次/秒)。8路电压输入型A/D，可做温度检测、电池电压检测、按键扫描、频谱检测等。上电复位后P1口为弱上拉型I/O口，用户可以通过软件设置将8路中的任何一路设置为A/D转换，不需作为A/D使用的口可继续作为I/O口使用。

STC12C5A60S2系列单片机ADC(A/D转换器)的结构如下图所示。



LCD 液晶模块选用的 LCD12864B 模块，其可 4 行显示，自带中文字库，完美满足我们要求。

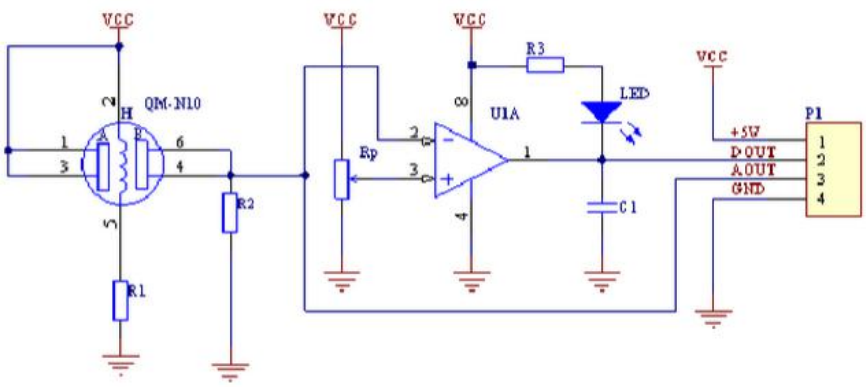
QC12864B



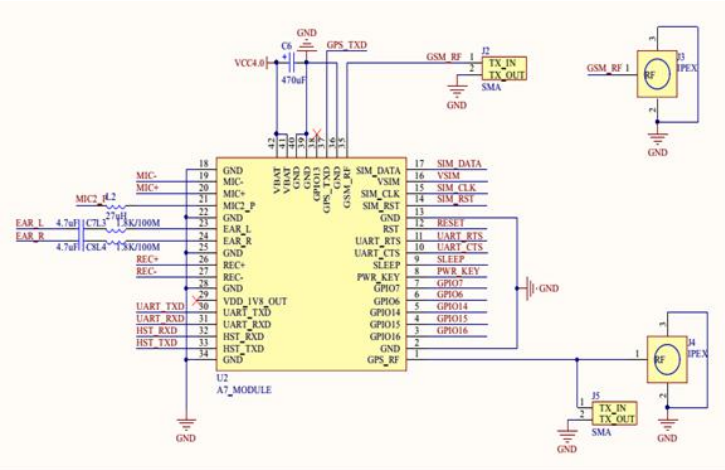
这里我们选用串行通讯，只需要接 9 个对应管脚即可。

2、 硬件电路图

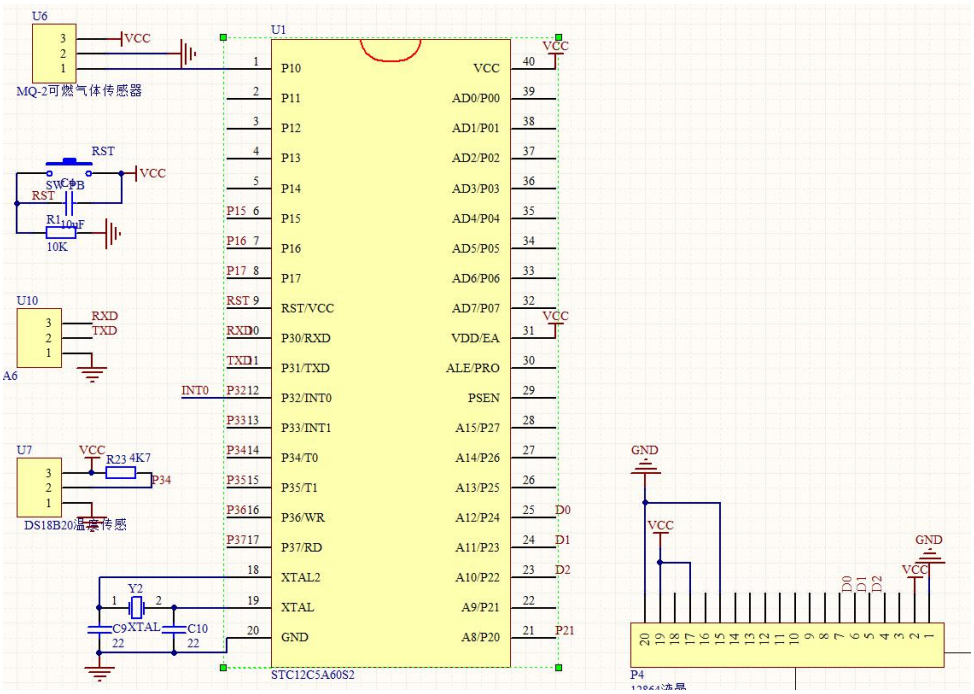
MQ-9 电路原理图：



A7 模块电路原理图：



系统电路原理图：

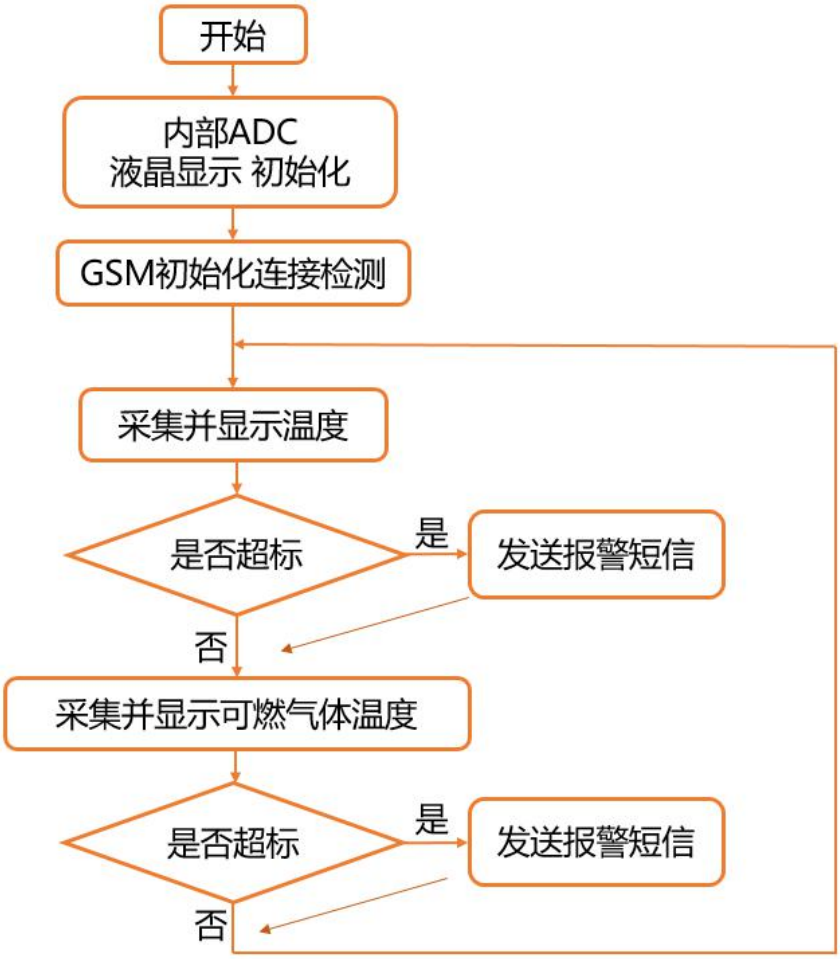


四、 系统软件设计

1、 设计思想

首先通过 DS18 B 20.c 和 LCD12864_s.c 和 USART.c 文件编写分别与 DS18B20、LCD12864 和 GSM 的基本通信,写基本指令函数,和初始化函数。main.c 里实现发短信的实现，和内部 AD 转换的实现，最终在 main()函数中调用这些函数来实现系统功能需求。

2、 软件流程图



3、 源程序（含注释，作为附录提供），核心程序应在报告中解释说明

//LCD12864 的函数

```
extern void WriteCommand( unsigned char Cbyte );写指令
extern void WriteData( unsigned char Dbyte );//显示字节
extern void LcmInit( void );//LCD 初始化
extern void LcmClearTXT( void );//清除文本信息
extern void PutStr(unsigned char row,unsigned char col,unsigned char *puts);//在指定位置写入一段字符串
```

//uart 的函数

```
void Uart_Init();//串口初始化，使波特率定为 115200
void SendData(unsigned char ch);//送数据
```

```
void SendString(char *s);//送入字符串
bit Hand(unsigned char *a);//握手判断
void CLR_Buf(void); //串口缓存清理

//DS18B20 的指令函数
extern void DS18B20_Delay( unsigned int n );延时
extern void DS18B20_Write_Byte( unsigned char dat);
extern unsigned char DS18B20_Read_Byte( );//读字节
extern bit DS18B20_Init();//初始化
extern unsigned int Get_temp(void);//读取一次温度

//main.c 的函数
extern void delay_ms(unsigned int n);
extern unsigned int sendCommand(char *Command, char *Response,
unsigned long Timeout, unsigned char Retry);//给 GSM 送入 AT
指令
extern void errorLog();//重新发指令，无应答的话
extern void soft_reset(void);//GSM 软件重启
extern void phone(char *number);//打电话
extern void sendMessage(char *number,char *msg);//发短信
extern unsigned int adc10_start(unsigned char channel);
//做一次 AD 转换

//main()
void main()
{
   ADC_CONTR = ADC_360T | ADC_ON;
   AUXR1 |= ADRJ;                                  //ADRJ = 1;
         //10bitAD 右对齐
```

```
  Uart_Init();

   LcmInit();                              //液晶初始化
   LcmClearTXT();                          //清除显示
   LcmClearBMP();
  delay_ms(100);
   PutStr(0,0,"。。初始化中。。");


   if (sendCommand("AT\r\n", "OK\r\n", 3000, 10) == Success);
   else errorLog();
   delay_ms(10);

   if (sendCommand("AT+CPIN?\r\n", "READY", 1000, 10) ==
Success);
   else errorLog();
   delay_ms(10);

   if (sendCommand("AT+CREG?\r\n", "CREG: 1", 1000, 10) ==
Success);
   else errorLog();
   delay_ms(10);

   if (sendCommand("AT+CMGF=1\r\n", "OK\r\n", 1000, 10) ==
Success);
   else errorLog();
   delay_ms(10);
```

```
    if (sendCommand("AT+CSCS=\"GSM\"\r\n", "OK\r\n", 1000, 10)
== Success);
    else errorLog();
    delay_ms(10);

    Temp_Buffer = Get_temp();
    delay_ms(1000);
    LcmClearTXT();                          //清除显示

    while(1)
    {
        Temp_Buffer = Get_temp();
     PutStr(0,0,"温度：");

        if( Temp_Buffer/1000 != 0 )          //如果第
一位为0，忽略显示
        {
            WriteData(Temp_Buffer/1000+0X30);     //显示温度
百位值
        }
        else
        {
            if(flag_temper == 1)             //根据温
度标志位显示温度正负
            {
                WriteData('-');
            }
            else
            {
```

```
            WriteData(' ');
        }
    }
    WriteData(Temp_Buffer%1000/100+0X30);      //显示温度十位值
    WriteData(Temp_Buffer%100/10+0X30);        //显示温度个位值
    WriteData('.');                            //显示小数点
    WriteData(Temp_Buffer%10+0X30);            //显示温度十分位值
    WriteData(' ');
    PutStr(0,7,"℃");

    if(flag_temper == 0)
    {
        if(Temp_Buffer/10 >= TEMP_MAX)
        {
            state = 1;
            PutStr(1,0,"温度超标");
        }
        else
        {
            state = 0;
            PutStr(1,0,"温度合适");
        }

        if(Temp_Buffer/10 <= TEMP_MAX - 1)     //低于报警值 1 度后发送短信功能才复位，防止不停的发短信
        {
```

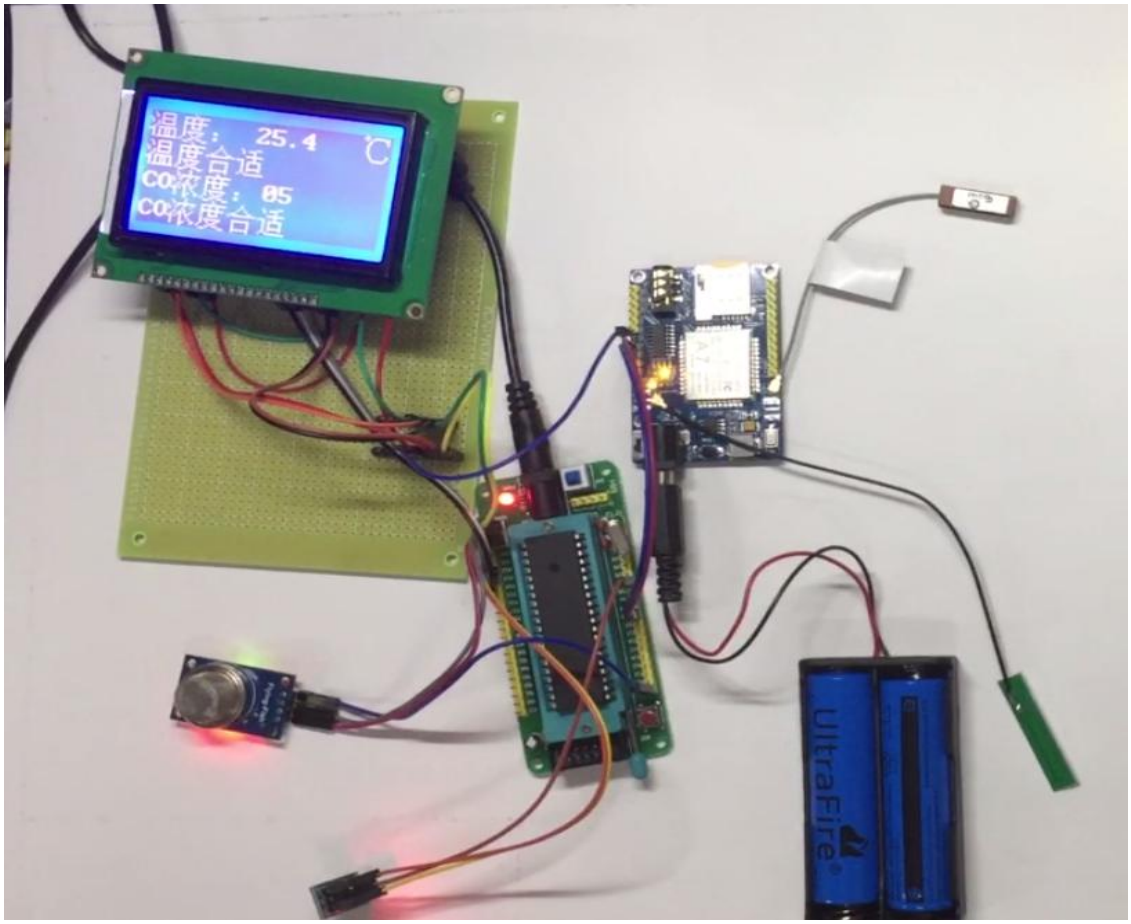
```
            state_pre = 0;
        }
    }

    if(state == 1&& state_pre == 0)
    {
        memset(msg,0,20);          //短信语句
    sprintf(msg,"HIGH TEMP:  %d",Temp_Buffer/10);//温度
        sendMessage(phoneNumber,msg);        //发送短信
        state_pre = state;
    }
    ADC_Buffer = adc10_start(0);         // P1.0 ADC
  PutStr(2,0,"CO 浓度：");
    ADC_Buffer = ADC_Buffer*100/1024;
    WriteData(ADC_Buffer%100/10+0X30);
    WriteData(ADC_Buffer%10+0X30);
        if(ADC_Buffer >= CO_MAX)
        {
            state = 1;
            PutStr(3,0,"CO 浓度超标");
        }
        else
        {
            state = 0;
            PutStr(3,0,"CO 浓度合适");
        }

        if(Temp_Buffer/10 <= TEMP_MAX - 1)      //低于报
```

警值 1 度后发送短信功能才复位，防止不停的发短信

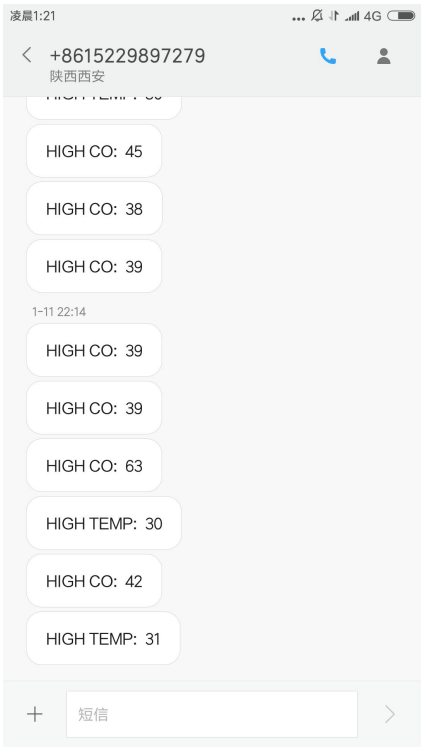
```
            {
                state_pre = 0;
            }

            if(state == 1&& state_pre == 0)
        {
            memset(msg,0,20);          //短信语句
       sprintf(msg,"HIGH CO:  %d",ADC_Buffer);//写入浓度
            sendMessage(phoneNumber,msg
            );      //发送短信
            state_pre = state;
        }
        delay_ms(500);

    }
}
```

五、 功能验证

1．实物验证



2. 手机端收到的报警信息



六、 附录：

Main.c

```
#include "main.h"
#include "uart.h"
#include "LCD12864_S.h"
#include "DS18B20.h"
//常量
#define Success 1U
#define Failure 0U
code char phoneNumber[] = "15934827892";     //替换成需要被拨打电话的号码
char msg[20] = "HIGH TEMP:00";       //短信内容
#define TEMP_MAX 30
#define CO_MAX 30
//定义变量
unsigned long  Time_Cont = 0;        //定时器计数器
unsigned char state = 0;
unsigned char state_pre = 0;
unsigned int Temp_Buffer = 0;
unsigned char key_num = 0;
bit flag_sw = 1;
unsigned int ADC_Buffer = 0;
void main()
{
ADC_CONTR = ADC_360T | ADC_ON;
AUXR1 |= ADRJ;                                    //ADRJ = 1;
//10bitAD 右对齐
  Uart_Init();
LcmInit();                                  //液晶初始化
LcmClearTXT();                              //清除显示
```

```
    LcmClearBMP();
      delay_ms(100);
    PutStr(0,0,"。。初始化中。。");
    if (sendCommand("AT\r\n", "OK\r\n", 3000, 10) == Success);
    else errorLog();
    delay_ms(10);
    if (sendCommand("AT+CPIN?\r\n", "READY", 1000, 10) == Success);
    else errorLog();
    delay_ms(10);
    if (sendCommand("AT+CREG?\r\n", "CREG: 1", 1000, 10) == Success);
    else errorLog();
    delay_ms(10);
    if (sendCommand("AT+CMGF=1\r\n", "OK\r\n", 1000, 10) == Success);
    else errorLog();
    delay_ms(10);
    if (sendCommand("AT+CSCS=\"GSM\"\r\n", "OK\r\n", 1000, 10) == Success);
    else errorLog();
    delay_ms(10);
    Temp_Buffer = Get_temp();
    delay_ms(1000);
    LcmClearTXT();                          //清除显示
    while(1)
    {
        Temp_Buffer = Get_temp();
        PutStr(0,0,"温度：");
            if( Temp_Buffer/1000 != 0 )                  //如果第一位为 0，忽略显
示
            {
                WriteData(Temp_Buffer/1000+0X30);       //显示温度百位值
            }
```

```
            else
            {
                if(flag_temper == 1)                    //根据温度标志位显
示温度正负
                {
                    WriteData('-');
                }
                else
                {
                    WriteData(' ');
                }
            }
            WriteData(Temp_Buffer%1000/100+0X30);       //显示温度十位值
            WriteData(Temp_Buffer%100/10+0X30);     //显示温度个位值
            WriteData('.');                         //显示小数点
            WriteData(Temp_Buffer%10+0X30);         //显示温度十分位值
            WriteData(' ');
        PutStr(0,7,"℃");
        if(flag_temper == 0)
        {
            if(Temp_Buffer/10 >= TEMP_MAX)
            {
                state = 1;
                PutStr(1,0,"温度超标");
            }
            else
            {
                state = 0;
                PutStr(1,0,"温度合适");
            }
```

```
        if(Temp_Buffer/10 <= TEMP_MAX - 1)  //低于报警值 1 度后发送短信功能
才复位，防止不停的发短信
            {
                state_pre = 0;
            }
        }
        if(state == 1&& state_pre == 0)
        {
            memset(msg,0,20);          //短信语句
          sprintf(msg,"HIGH TEMP:  %d",Temp_Buffer/10);
            sendMessage(phoneNumber,msg);          //发送短信
            state_pre = state;
        }
        ADC_Buffer = adc10_start(0);          // P1.0 ADC
        PutStr(2,0,"CO 浓度：");
        ADC_Buffer = ADC_Buffer*100/1024;
        WriteData(ADC_Buffer%100/10+0X30);
        WriteData(ADC_Buffer%10+0X30);
            if(ADC_Buffer >= CO_MAX)
            {
                state = 1;
                PutStr(3,0,"CO 浓度超标");
            }
            else
            {
                state = 0;
                PutStr(3,0,"CO 浓度合适");
            }
            if(Temp_Buffer/10 <= TEMP_MAX - 1)  //低于报警值 1 度后发送短信功能
才复位，防止不停的发短信
```

```
        {
            state_pre = 0;
        }
        if(state == 1&& state_pre == 0)
    {
        memset(msg,0,20);        //短信语句
      sprintf(msg,"HIGH CO:  %d",ADC_Buffer);
        sendMessage(phoneNumber,msg
        );       //发送短信
        state_pre = state;
    }
    delay_ms(500);
}
}
//*******************************************************
//做一次 ADC 转换
//*******************************************************
unsigned int adc10_start(unsigned char channel) //channel = 0~7
{
unsigned int    adc;
unsigned char   i;
P1ASF = (1 << channel);          //12C5A60AD/S2 系列模拟输入(AD)选择
ADC_RES = 0;
ADC_RESL = 0;
ADC_CONTR = (ADC_CONTR & 0xe0) | ADC_START | channel;
_nop_();
_nop_();
_nop_();
_nop_();
//  for(i=0; i<250; i++)         //13T/loop, 40*13=520T=23.5us @ 22.1184M
```

```
i = 250;
do{
    if(ADC_CONTR & ADC_FLAG)
    {
        ADC_CONTR &= ~ADC_FLAG;              //软件清零 ADC_FLAG
        _nop_();
        _nop_();
        _nop_();
        _nop_();
        adc = 0;
        adc = (ADC_RES << 8) | ADC_RESL;    //ADRJ_enable()
        return  adc;
    }
}while(--i);
return  1024;
}
void sendMessage(char *number,char *msg)
{
xdata char send_buf[20] = {0};
memset(send_buf, 0, 20);    //清空
strcpy(send_buf, "AT+CMGS=\"");
strcat(send_buf, number);
strcat(send_buf, "\"\r\n");
if (sendCommand(send_buf, ">", 3000, 10) == Success);
else errorLog();
SendString(msg);
SendData(0x1A);
}
void phone(char *number)
{
```

```
    char send_buf[20] = {0};
    memset(send_buf, 0, 20);    //清空
    strcpy(send_buf, "ATD");
    strcat(send_buf, number);
    strcat(send_buf, ";\r\n");
    if (sendCommand(send_buf, "SOUNDER", 10000, 10) == Success);
    else errorLog();
    }
    void errorLog()
    {
    while (1)
    {
        if (sendCommand("AT\r\n", "OK", 100, 10) == Success)
        {
            soft_reset();
        }
        delay_ms(200);
    }
    }
    void soft_reset(void)    //制造重启命令
    {
       ((void (code *) (void)) 0x0000) ();
    }
    unsigned int sendCommand(char *Command, char *Response, unsigned long Timeout,
unsigned char Retry)
    {
    unsigned char n;
    CLR_Buf();
    for (n = 0; n < Retry; n++)
    {
```

```
    SendString(Command);          //发送 GPRS 指令
    Time_Cont = 0;
    while (Time_Cont < Timeout)
    {
        delay_ms(100);
        Time_Cont += 100;
        if (strstr(Rec_Buf, Response) != NULL)
        {
            CLR_Buf();
            return Success;
        }
    }
    Time_Cont = 0;
}
CLR_Buf();
return Failure;
}
//******************************************************
//MS 延时函数(12M 晶振下测试)
//******************************************************
void delay_ms(unsigned int n)
{
unsigned int  i,j,m;
for(m = 0 ; m < 10 ; m++)
    for(i=0;i<n;i++)
        for(j=0;j<123;j++);
}
```

c

```
#include "DS18B20.h"
#include<intrins.h>
```

```
//定义变量
unsigned char flag_temper = 0;
void Delay800us()        //@11.0592MHz
{
unsigned char i, j;
_nop_();
i = 9;
j = 151;
do
{
    while (--j);
} while (--i);
}
void Delay40us()    //@11.0592MHz
{
unsigned char i;
_nop_();
_nop_();
i = 107;
while (--i);
}
void Delay200us()        //@11.0592MHz
{
unsigned char i, j;
_nop_();
_nop_();
i = 3;
j = 34;
do
{
```

```
    while (--j);
} while (--i);
}
//*********************************************************
//DS18B20 延时函数
//*********************************************************
void DS18B20_Delay( unsigned int n )
{
unsigned int i,j;
for(j = 0 ; j < 12 ; j++ );
    for(i = 0 ; i < n ; i++ );
}
//*********************************************************
//DS18B20 写 1 字节
//*********************************************************
void DS18B20_Write_Byte( unsigned char dat)
{
unsigned char i;
for( i = 0 ; i < 8 ; i++ )
{
    DS18B20_DQ = 0;
    _nop_();_nop_();_nop_();_nop_();_nop_();            //延时>1us
    _nop_();_nop_();_nop_();_nop_();_nop_();
    DS18B20_DQ = dat&0x01;      //先写低位
    dat >>= 1;
    DS18B20_Delay(6);   //延时 60~120us
    DS18B20_DQ = 1;     //释放总线
    _nop_();_nop_();_nop_();_nop_();_nop_();            //延时>1us
    _nop_();_nop_();_nop_();_nop_();_nop_();
}
```

```
}
//*****************************************************
//DS18B20 读 1 字节
//*****************************************************
unsigned char DS18B20_Read_Byte( )
{
unsigned char dat,i;
for( i = 0 ; i < 8 ; i++ )
{
    DS18B20_DQ = 0;
    _nop_();_nop_();_nop_();_nop_();_nop_();            //延时>1us
    _nop_();_nop_();_nop_();_nop_();_nop_();
    DS18B20_DQ = 1;     //释放总线
    _nop_();_nop_();_nop_();_nop_();_nop_();            //延时>1us
    _nop_();_nop_();_nop_();_nop_();_nop_();
    dat >>= 1;
    if( DS18B20_DQ == 1)
    {
        dat |= 0X80;
    }
    else
    {
        dat &= 0x7f;
    }
    DS18B20_Delay(6);   //延时 60~120us
}
return dat;
}
//*****************************************************
//DS18B20 初始化
```

```
//*********************************************************
bit DS18B20_Init()
{
bit Flag_exist = 1;
DS18B20_DQ = 1;          //释放总线
_nop_();_nop_();_nop_();_nop_();_nop_();                    //延时>1us
_nop_();_nop_();_nop_();_nop_();_nop_();
DS18B20_DQ = 0;
Delay800us();        //延时 480~960us
DS18B20_DQ = 1;          //释放总线
Delay40us();         //延时 15~60us
Flag_exist = DS18B20_DQ;
Delay200us();        //延时 60~240us
DS18B20_DQ = 1;          //释放总线
return Flag_exist;
}
//*************************************************************
//读取温度函数，返回温度的绝对值，并标注 flag_temper，flag_temper=1 表示负，
flag_temper=0 表示正
//*************************************************************
unsigned int Get_temp(void)          //读取温度值
{
float tt;
unsigned char a,b;
unsigned int temp;
if( DS18B20_Init() == 0 )                  //初始化
{
    DS18B20_Write_Byte(0xcc);           //忽略 ROM 指令
    DS18B20_Write_Byte(0x44);           //温度转换指令
//  _delay_ms(750);              //PROTEUS 仿真需要加
```

```
        if( DS18B20_Init() == 0 )                    //初始化
        {
            DS18B20_Write_Byte(0xcc);           //忽略 ROM 指令
            DS18B20_Write_Byte(0xbe);           //读暂存器指令
            a = DS18B20_Read_Byte();            //读取到的第一个字节为温度 LSB
            b = DS18B20_Read_Byte();            //读取到的第一个字节为温度 MSB
            temp = b;                          //先把高八位有效数据赋于 temp
            temp <<= 8;                        //把以上 8 位数据从 temp 低八位移到高八
位
            temp = temp|a;                     //两字节合成一个整型变量
            if(temp>0xfff)
            {
                flag_temper=1;                  //温度为负数
                temp=(~temp)+1;
            }
            else
            {
                flag_temper=0;                  //温度为正或者 0
            }
            tt = temp*0.0625;                  //得到真实十进制温度值
                                               //因为 DS18B20 可以精确到 0.0625 度
                                                   //所以读回数据的最低位代表的是
0.0625 度
            temp = tt*10+0.5;                  //放大十倍
                                                //这样做的目的将小数点后第一位也转
换为可显示数字
                                               //同时进行一个四舍五入操作。
        }
    }
    return temp;
```

```
}
```

DS18B20.h

```
#ifndef __DS18B20_H__
#define __DS18B20_H__
#include "STC12C5A.h"
//DS18B20 IO 设置
sbit DS18B20_DQ = P3^4;
//函数或者变量声明
extern void DS18B20_Delay( unsigned int n );
extern void DS18B20_Write_Byte( unsigned char dat);
extern unsigned char DS18B20_Read_Byte( );
extern bit DS18B20_Init();
extern unsigned int Get_temp(void);
extern unsigned char flag_temper;
extern void Delay800us();
extern void Delay40us();
extern void Delay200us();
#endif
```

LCD12864_S.c

```
#include "LCD12864_S.h"
/*****************************************************************
                    i0 口宏定义区
*****************************************************************/
sbit CS =P2^2;
sbit SID=P2^3;//r/w
sbit SCK=P2^4;//e
/*****************************************************************************
                    常量声明区
*****************************************************************************/
```

```
unsigned char code AC_TABLE[]={                         //坐标编码
0x80,0x81,0x82,0x83,0x84,0x85,0x86,0x87,
0x90,0x91,0x92,0x93,0x94,0x95,0x96,0x97,
0x88,0x89,0x8a,0x8b,0x8c,0x8d,0x8e,0x8f,
0x98,0x99,0x9a,0x9b,0x9c,0x9d,0x9e,0x9f,
};
/*********************************************************************
 声明：建议读者先查阅我们提供的 12864word 文档资料，理解 12864 定坐标的
 方式。
                发送一个字节
*********************************************************************/
void SendByte(unsigned char Dbyte)
{
unsigned char i;
for(i=0;i<8;i++)
{
    SCK = 0;
    Dbyte=Dbyte<<1;
    SID = CY;
    SCK = 1;
    SCK = 0;
}
}
/*************************************************************
                接收一个字节
*************************************************************/
unsigned char ReceiveByte(void)
{
unsigned char i,temp1,temp2;
temp1=temp2=0;
```

```
for(i=0;i<8;i++)
{
    temp1=temp1<<1;
    SCK = 0;
    SCK = 1;
    SCK = 0;
    if(SID) temp1++;
}
for(i=0;i<8;i++)
{
    temp2=temp2<<1;
    SCK = 0;
    SCK = 1;
    SCK = 0;
    if(SID) temp2++;
}
return ((0xf0&temp1)+(0x0f&temp2));
}
/**************************************************************************
                        检查忙状态
**************************************************************************/
void CheckBusy( void )
{
do SendByte(0xfc);     //11111,RW(1),RS(0),0
while(0x80&ReceiveByte());
}
/**************************************************************************
            写一个字节的指令
**************************************************************************/
void WriteCommand( unsigned char Cbyte )
```

```
{
CS = 1;
CheckBusy();
SendByte(0xf8);           //11111,RW(0),RS(0),0
SendByte(0xf0&Cbyte);
SendByte(0xf0&Cbyte<<4);
CS = 0;
}
/*********************************************************************
                    写一个字节的数据
*********************************************************************/
void WriteData( unsigned char Dbyte )
{
CS = 1;
CheckBusy();
SendByte(0xfa);           //11111,RW(0),RS(1),0
SendByte(0xf0&Dbyte);
SendByte(0xf0&Dbyte<<4);
CS = 0;
}
/*********************************************************************
                          lcd 初始化函数
*********************************************************************/
void LcmInit( void )
{
     WriteCommand(0x30);
     WriteCommand(0x03);
     WriteCommand(0x0c);
     WriteCommand(0x01);
     WriteCommand(0x06);
```

```
    }
    /*********************************************************************************
*****************************
```

设定光标函数

```
    *********************************************************************************
*****************************/
    void Location_xy_12864(unsigned char x,unsigned char y)
    {
    switch(x)
    {
        case 0:
            x=0x80;break;
        case 1:
            x=0x90;break;
        case 2:
            x=0x88;break;
        case 3:
            x=0x98;break;
        default:
            x=0x80;
    }
    y=y&0x07;
    WriteCommand(0x30);
    WriteCommand(y+x);
    WriteCommand(y+x);

    }
    /*********************************************************************************
*********
```

清除文本

```
    ********************************************************************************
*********/
    void LcmClearTXT( void )
    {
    unsigned char i;
    WriteCommand(0x30);
    WriteCommand(0x80);
    for(i=0;i<64;i++)
    WriteData(0x20);
    Location_xy_12864(0,0);
    }
    /********************************************************************************
************
                          清除图片
    ********************************************************************************
**************/
    void LcmClearBMP( void )
    {
    unsigned char i,j;
    WriteCommand(0x34);
    WriteCommand(0x36);
    for(i=0;i<32;i++)
    {
        WriteCommand(0x80|i);
        WriteCommand(0x80);
        for(j=0;j<32;j++)
        WriteData(0);
    }
    }
    /********************************************************************************
```

```
**************
                显示字符串
    ********************************************************************************
**************/
    void PutStr(unsigned char row,unsigned char col,unsigned char *puts)
    {
    WriteCommand(0x30);
    WriteCommand(AC_TABLE[8*row+col]);
    while(*puts != '\0')
    {
        if(col==8)
        {
            col=0;
            row++;
        }
        if(row==4) row=0;
        WriteCommand(AC_TABLE[8*row+col]);
        WriteData(*puts);
        puts++;
        if(*puts != '\0')
        {
            WriteData(*puts);
            puts++;
            col++;
        }
    }
    }
```

LCD12864_S.h

```
#ifndef __LCD12864_S_H__
```

```
#define __LCD12864_S_H__
#include "STC12C5A.h"
#include <intrins.h>
//LCD12864 的函数
extern void WriteCommand( unsigned char Cbyte );
extern void WriteData( unsigned char Dbyte );
extern void LcmInit( void );
extern void LcmClearTXT( void );
extern void LcmClearBMP( void );
//extern void PutTemp(unsigned char row,unsigned char col);
extern void PutStr(unsigned char row,unsigned char col,unsigned char *puts);
//extern void PutNum(unsigned char row,unsigned char col,unsigned int num);
#endif
```

MAIN.h

```
#ifndef __MAIN_H__
#define __MAIN_H__
#include "STC12C5A.h"
#include <intrins.h>
#define ADC_OFF()   ADC_CONTR = 0
#define ADC_ON      (1 << 7)
#define ADC_90T     (3 << 5)
#define ADC_180T(2 << 5)
#define ADC_360T(1 << 5)
#define ADC_540T0
#define ADC_CH0     0
#define ADC_CH1     1
#define ADC_CH2     2
#define ADC_CH3     3
#define ADC_CH4     4
```

```
#define ADC_CH5     5
#define ADC_CH6     6
#define ADC_CH7     7
```

MAIN.h

```
//函数或者变量声明
extern void delay_ms(unsigned int n);
extern unsigned int sendCommand(char *Command, char *Response, unsigned long Timeout, unsigned char Retry);
extern void errorLog();
extern void soft_reset(void);
extern void phone(char *number);
extern void sendMessage(char *number,char *msg);
extern unsigned int adc10_start(unsigned char channel);//channel = 0~7
#endif
```

uart.c

```
#include "uart.h"
#define Uart1_Buf_Max 100                    //串口数据缓存长度
u8 xdata  Rec_Buf[Uart1_Buf_Max];  //串口数据缓存
u8 point1 = 0;                 //绶存指针
void Uart_Init() //115200@11.0592M
{
    PCON &= 0x7F;     //波特率不倍速
    SCON = 0x50;      //8 位数据,可变波特率
    AUXR |= 0x40;     //定时器 1 时钟为 Fosc,即 1T
    AUXR &= 0xFE;     //串口 1 选择定时器 1 为波特率发生器
    TMOD &= 0x0F;     //清除定时器 1 模式位
    TMOD |= 0x20;     //设定定时器 1 为 8 位自动重装方式
    TL1 = 0xFD;       //设定定时初值
```

```
    TH1 = 0xFD;        //设定定时器重装值
    ET1 = 0;       //禁止定时器 1 中断
    TR1 = 1;       //启动定时器 1
    EA   = 1;      //打开全局中断控制
    ES   = 1;      //串行口中断
}
/*----------------------------
发送串口数据
----------------------------*/
void SendData(unsigned char ch)
{
    SBUF = ch;                  //写数据到 UART 数据寄存器
       while(TI == 0);
       TI = 0;
}
/*----------------------------
发送字符串
----------------------------*/
void SendString(char *s)
{
    while (*s)                  //检测字符串结束标志
    {
        SendData(*s++);         //发送当前字符
    }
}
bit Hand(unsigned char *a)                  // 串口命令识别函数
{
    if(strstr(Rec_Buf,a)!=NULL)
       return 1;
```

```
    else
        return 0;
}
void CLR_Buf(void)                              // 串口缓存清理
{
    memset(Rec_Buf, 0, Uart1_Buf_Max);      //清空

     point1 = 0;
}
void RECEIVE_DATA(void) interrupt 4 using 1          //串口中断
{
    ES = 0;
    if (RI)
     {
        RI = 0;                                  //清除 RI 位
        Rec_Buf[point1] = SBUF;
//      if (Rec_Buf[0] == 0xd9)
//          {
//              IAP_CONTR = 0x60;
//          }
        point1++;
        if(point1>=Uart1_Buf_Max)
        {
            point1 = 0;
        }
     }
    ES =  1;
}
```

UART.h

```
#ifndef __UART_H__
#define __UART_H__
#include "STC12C5A.h"
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
typedef   unsigned char u8;
typedef   unsigned int  u16;
typedef   unsigned long u32;
//函数或者变量声明
void Uart_Init();
void SendData(unsigned char ch);
void SendString(char *s);
bit Hand(unsigned char *a);
void CLR_Buf(void);
extern u8 xdata  Rec_Buf[];  //串口数据缓存
extern u8 point1;              //绶存指针
#endif
```

七、 设计心得

经过将进一个月的构思设计，到实物制作代码调试，经历了许多艰难困苦。有的时候为了找到程序里的 bug,熬到夜里两点；因为自己的稳压电源不合格，直接烧掉一块 A6 板子，被迫借同学的 A7 替代来用。但是收获也是成正比的。最直接的是我拿到了这个成品。同时在这过程学习到了大量的软硬件相关知识和相关代码调试经验，这种经历是无价的。