东北大学秦皇岛分校控制工程学院

**《过程控制系统》课程设计**

**设计题目：智能化液位控制系统设计**

**学 生：陈若愚**

**专 业：测控技术与仪器**

**班级学号：20178210**

**指导教师：宋爱娟，齐世清**

**设计时间：2020.6. 22-2020.7.3**

东北大学秦皇岛分校控制工程学院

**《过程控制系统》课程设计任务书**

**专业 测控技术与仪器 班级 1701班 姓名 陈若愚**

**设计题目：** 智能化液位控制系统设计

**一、设计实验条件**

**过程控制系统实验室实验系统**

**二、设计任务**

1、系统构成：

**对象：**同学们根据需要自己选择控制的液位对象

**传感器：**根据所选对象选取合适的传感器，包括型号、灵敏度、测量范围、精度、原理等。

**控制器：**本组同学必须选取单片机作为控制器，单片机型号自定。

**执行器：**查找资料自行选择。必须和自己的控制对象相匹配。

2、写出液位测量与控制过程，绘制液位控制系统组成框图。

3、（1）系统硬件电路设计自选。

（2）编制液位测量控制程序：软件采用模块化程序结构设计，由液位采集程序、液位校准程序、液位控制程序等部分组成。

**三、设计说明书的内容**

1. **设计题目与设计任务（设计任务书）**
2. **前言（绪论）(设计的目的、意义等)**
3. **主体设计部分**
4. **参考文献**
5. **结束语**

**四、设计时间与设计时间安排**

**1、 设计时间： 2周**

**2、设计时间安排：**

熟悉实验设备、实验、收集资料： 4天

设计计算、绘制技术图纸： 4天

编写课程设计说明书： 5 天

答辩： 1天

# 前言

在现代工业生产控制过程中，主要的被控量通常为：液位、流量、温 度和压力４种。其中，液位作为这４种常见被控量中最容易测量和直接观察的被控量，理所应当受到的关注也是最多。而双容水箱作为最 常见的液位控制对象，工业上很多复杂的被控对象常常抽象为它的数学模型来研究。所以，对双容水箱的液位控制研究具有很广泛的应用背景和很重要的研究价值。可是双容水箱系统一般呈非线性，并且存在着容积延迟，液位变化缓慢，系统惯性比较大等问题。实践证明，传统PID系统对双容水箱的调节时间比较长且控制效果并不是很好[1]。

在人们生活以及工业生产等诸多领域常常会涉及到液位及流量的控制问题，需要设计出合适的控制器来自动调整液位及流量，使得液位满足要求。各种实际生产中的液位控制问题，我们将其简化为某种水箱的液位控制问题。这种问题具有非线性及滞后特性等等，常规PID控制效果不好，鉴于模糊控制特性，许多学者将模糊控制和PID控制结合应用在液位控制上[2]。

模糊控制也是一种发展比较早的控制算法，已经有着将近六十年的发展历史。任何控制方法的发展都必须先有控制方法的理论基础，才能被转化成生产力。模糊控制发展的最初阶段是美国的 L.A.Zadeh 教授于1965年提出了模糊集合这一概念[3]。在这一概念提出之后许多科学家对此概念进行了深入的研究。1974年，E.H.Mamdani用模糊控制器制造出了第一台锅炉控制的蒸汽机[4]，这标志着模糊控制这一概念从理论应用到了解决实际存在的问题，是理论转变为实际应用的巨大的进步。随着科技的发展，科学研究的工具也慢慢出现在科学研究中，1985年，Seiji Yasunobu和Soji Miyamoto教授利用先进的仿真系统对建立好的模糊控制器进行仿真分析，获得很好的控制效果，并将获得成功的模糊控制器用于仙台地铁的运行、加速及刹车上[5-7]。最终仙台地铁成功通车，这为以后模糊控制的应用奠定了基础。到 21 世纪初，模糊控制已经比较成熟，并被用在各行各业的各个领域，随之模糊产品被大量生产并被推往世界各地。

本文针对双容液位对象提出了串级模糊控制方法，与传统的PID算法相比，我们的方法减少了调节时间并增强了系统的稳定性。首先，对本文所使用的双容水箱液位控制系统进行了系统描述，建立了系统的原理图；然后，基于AT89C51单片机，通过Proteus软件搭建了系统的硬件连线图，并进行硬件功能方面的仿真；提出了一种串级-模糊控制算法，并进行了Simulink仿真分析，得到对应的控制效果图。通过效果图，我们的调节时间达到0.5s，具有良好的控制性能。

# 双容水箱液位控制系统建模

图1展示了一种双容液位控制系统。由两个一阶非周期惯性环节的水箱串联起来的液位控制系统，控制量是下水箱的水位。控制的目的既要使下水箱的液位高度等于给定值所期望的值，又要具有减少或消除来自系统内部或外部扰动的影响。输入量为上调节阀产生的阀门开度变化，而输出量为第二个水箱的液位增量。由水泵的正常工作提供水压。在正常工作状态下，双容水箱液位控制系统处于平衡状态。在整个实验过程中下调节阀开度保持不变，当调节阀开度发生变化时，液位随之发生变化。这会引起水箱中水压发生改变，导致流出量发生变化。双容水箱建模过程如下：

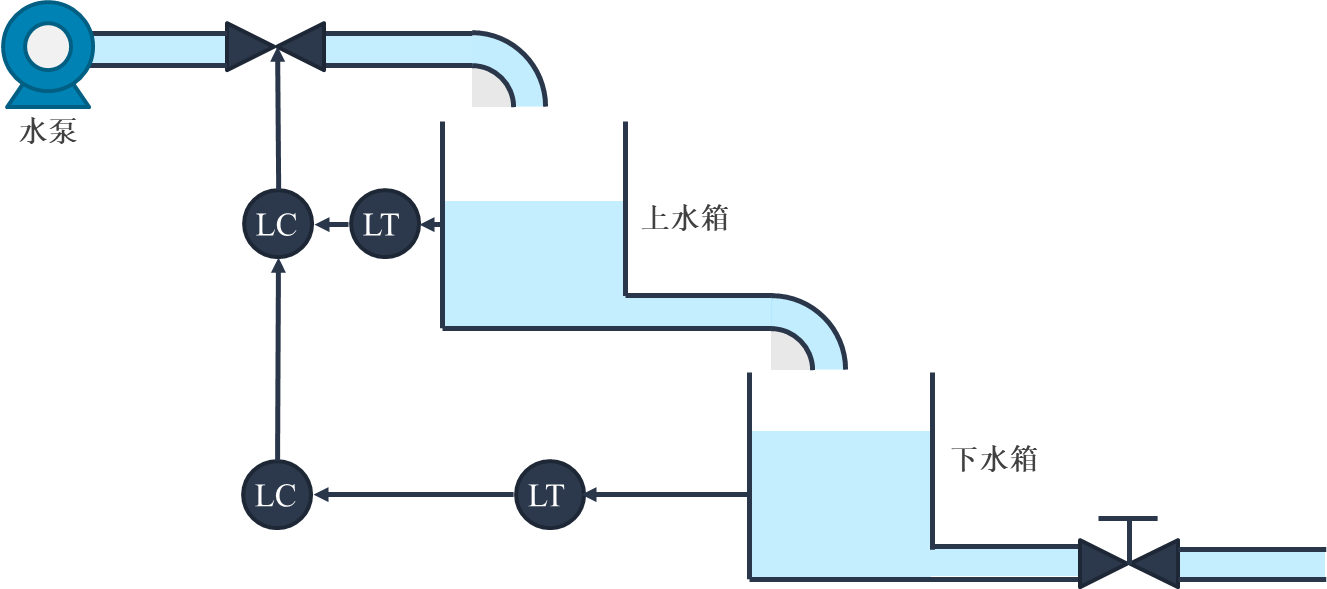


图1：本文设计的双容液位控制系统原理图。

式中，，为输入两个水箱水量的增量，表示输入水流量的增加。，表示两个水箱液位的增量。，表示两个液槽的液阻。，为流出端负载阀门的阻力即液阻。为调节阀开度的变化。为阀门流量系数。将(2)，(3)带入(1)中得到：

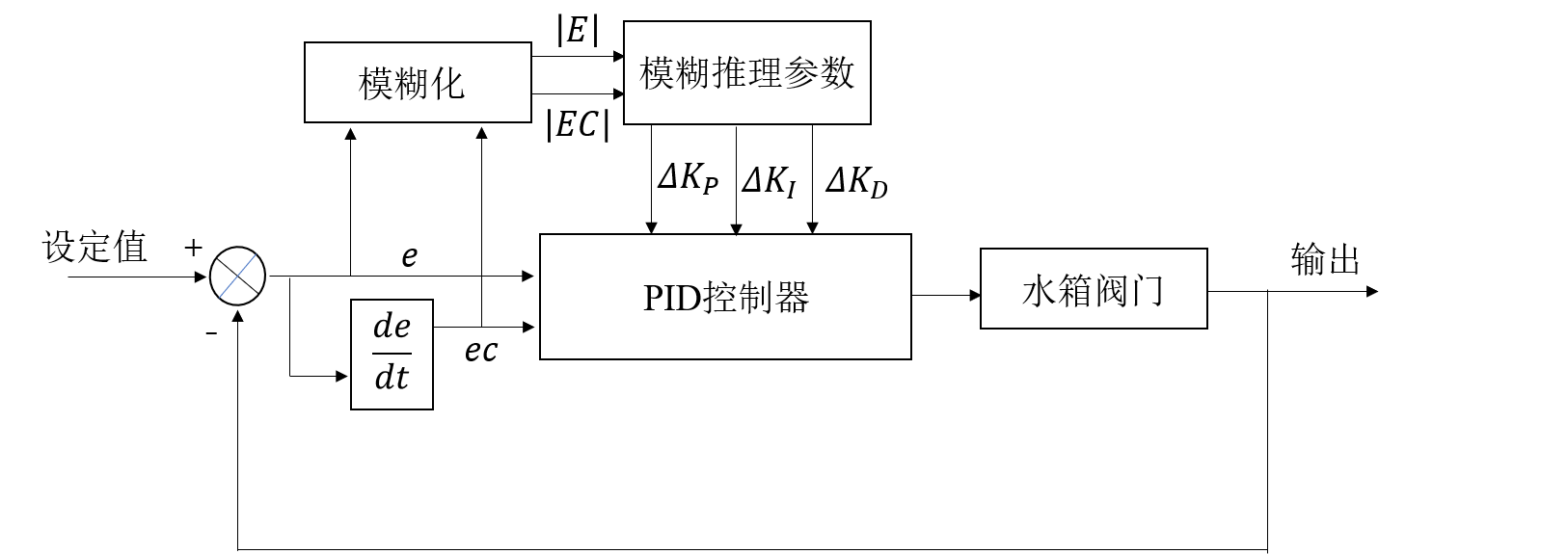


图2：模糊自适应PID的结构。

将式(2)，(3)，(5)代入(4)中，得到

分别将式(7)，(8)代入(9)得

式中，为第一个水槽的时间常数，为第二个水槽的时间常数，为双容水槽的传递系数。

在零初始条件下，对上式进行拉氏变换，把微分方程转变成代数方程，整理后得到双容水槽的传递函数:

模糊自适应 PID 控制就是应用模糊数学的基本理论和方法，把控制规则的条件和操作用模糊数学的模糊集来表示，并把这些模糊控制规则的有关信息（如评价指标、初始 PID 参数等）存入计算机知识库，然后计算机根据控制系统的实际响应情况，运用模糊推理不断地进行修正，实现对 PID 控制参数的自动调整，最终得到最佳调整参数。本文选用的模糊自适应 PID 控制器结构如图2所示。

表1：P参数的模糊规则表。

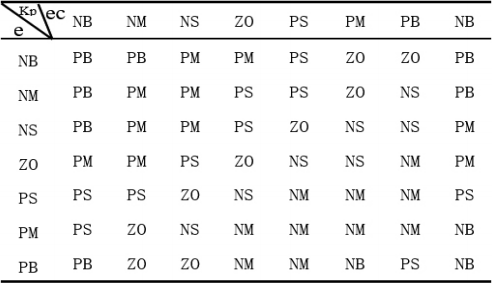


表2：K参数的模糊规则表。

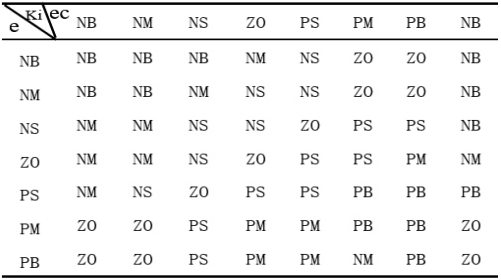
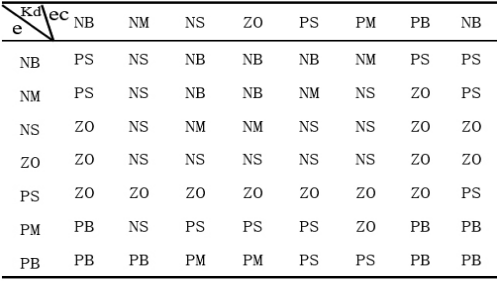


表3：D参数的模糊规则表。



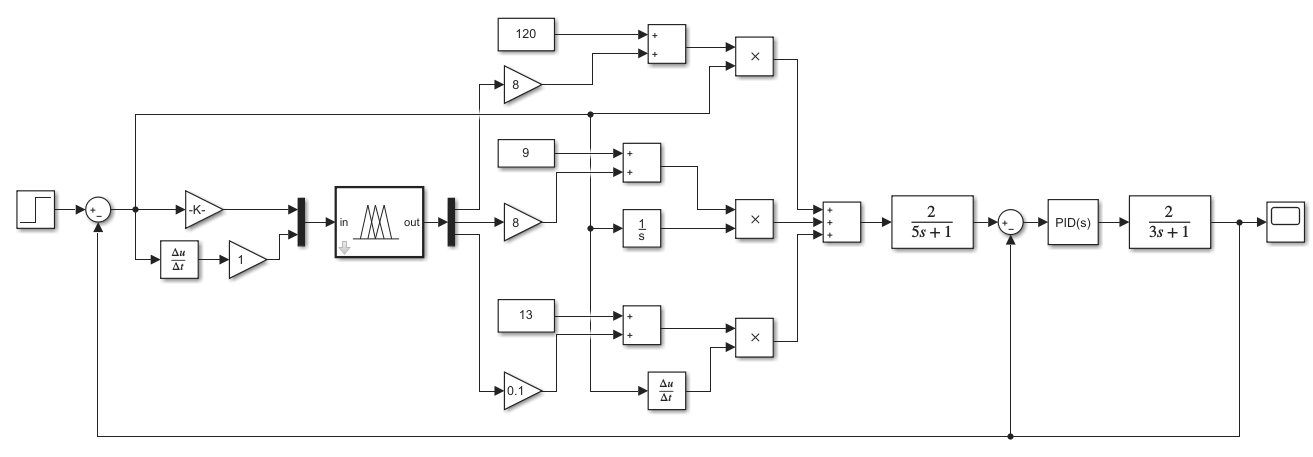


图3：模糊自适应PID的Simulink仿真框图。

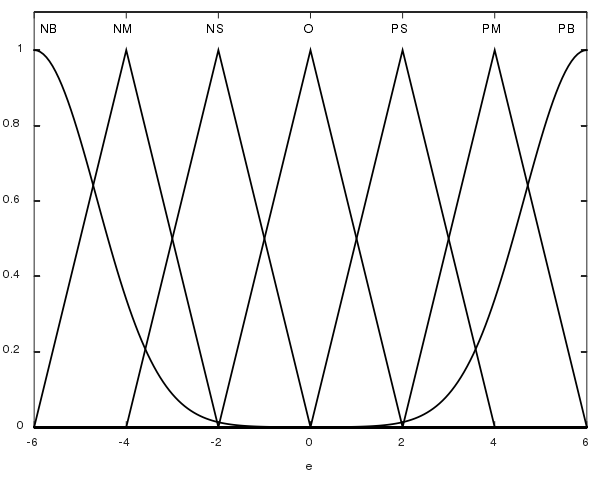


图4：误差e与ec的隶属度函数。

针对本文模型，选用二阶传递函数，采用模糊控制方式，模糊控制的Simulink仿真框图如图3所示：

对应的PID控制参数如上图所示，所选用的模糊规则表如表1-3所示，所设定误差e与ec的隶属度函数如图4所示。

我们给予一阶跃信号进行控制仿真，动态响应曲线如图5所示，调节时间较短，稳定性较强，控制系统较好。

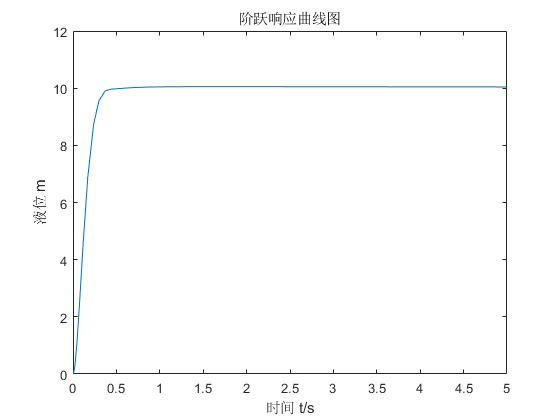


图5：Simulink仿真曲线。

# 硬件设计

本文所使用的控制器为AT89C51单片机，执行器采用电关阀，输入信号为4-20mA。所用的传感器采用GKY液位传感器，量程为0-1m，输出信号0-5V。所设计的硬件电路的整体架构如图6所示，电路包括51单片机最小系统，液位检测装置，参数显示电路，温度传感器，按键操作，水泵电机电路，状态显示电路和执行器控制电路。

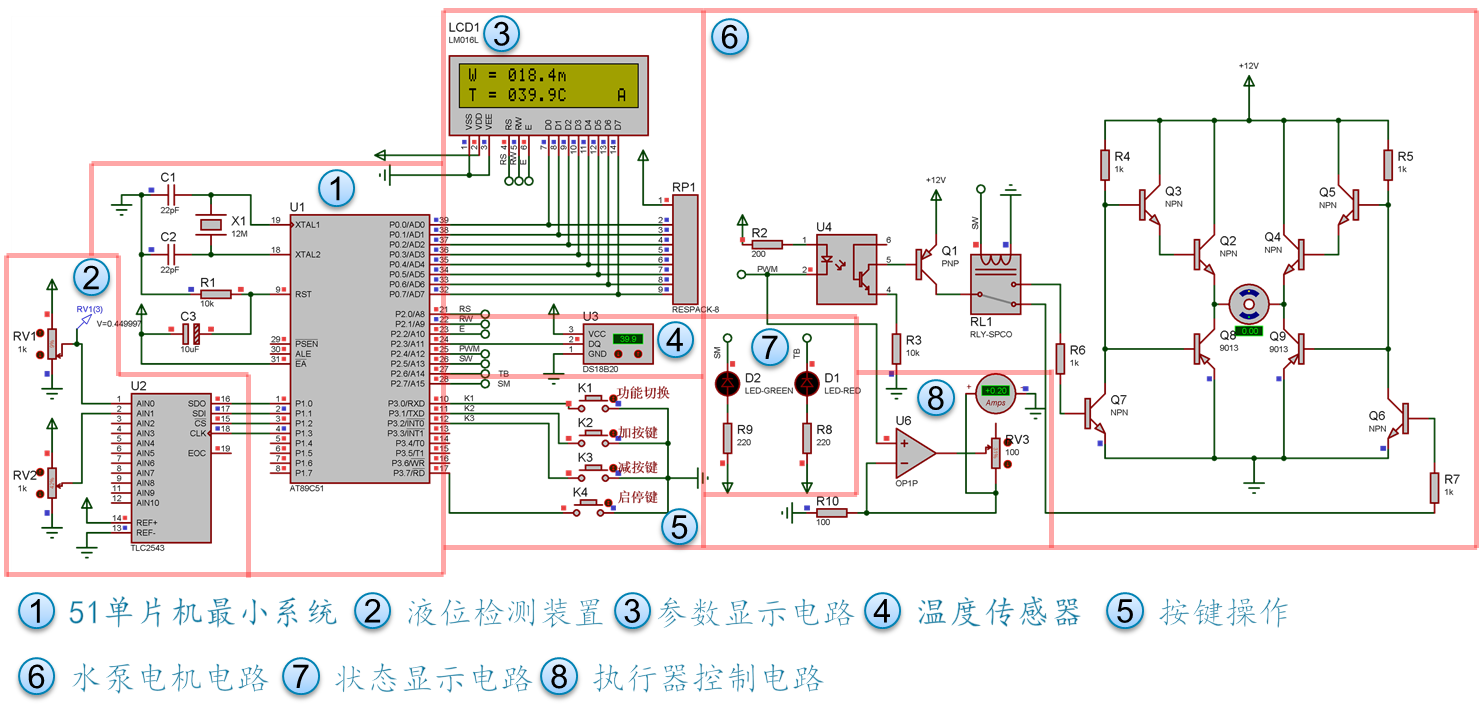


图6：硬件电路整体架构。

液位检测电路，如图6中的2所示，所使用的传感器为GKY液位传感器，其结构如图7所示，原理为因液位不同而漂浮于液位小球高度不同，使拥有不同的电阻值，通过TLC2543A/D转换器读取电压信号进行转换而获得液位高度。

参数显示电路，如图6中的3所示，通过不断地刷新而将单片机内部存储的数值显示出来。

温度传感器，如图6中的4所示，采用了DS18B20传感器，因为方便仿真。

按键操作电路，如图6中的5所示，包括4个按键，功能切换，加按键，减按键，启停键。

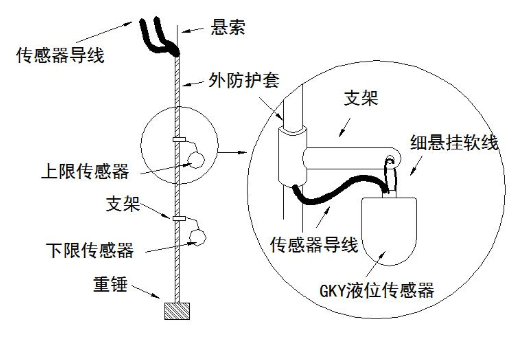


图7：GKY液位传感器结构。

水泵电机电路，如图6中的6所示，PWM口通过PWM调制可以控制电机的启停与速度，SW口的高低电平用于控制电机的正反转。

状态显示电路，如图6的7所示，包括2个LED灯，红灯是温度报警信号，如果温度超过了40℃，则红灯亮起；绿灯用于表示水泵是否工作的电路，启动电机时绿灯点亮，若因为人为关闭或者因异常而停止电机，则绿灯熄灭。

执行器控制电路，如图6的8所示，输入是由PWN调制以模拟输入的电压信号的大小，输出电流大小可表示为：，其中为接入的执行器的等效电阻值，通过调整滑动变阻器可以进行仪表调零等工作。

# 软件设计

软件方面主要模拟真实场景下控制系统的显示界面，参数设置，以及传感器信号收集和检测等等。整个控制系统的流程图如图8所示，整个系统的控制程序附录。

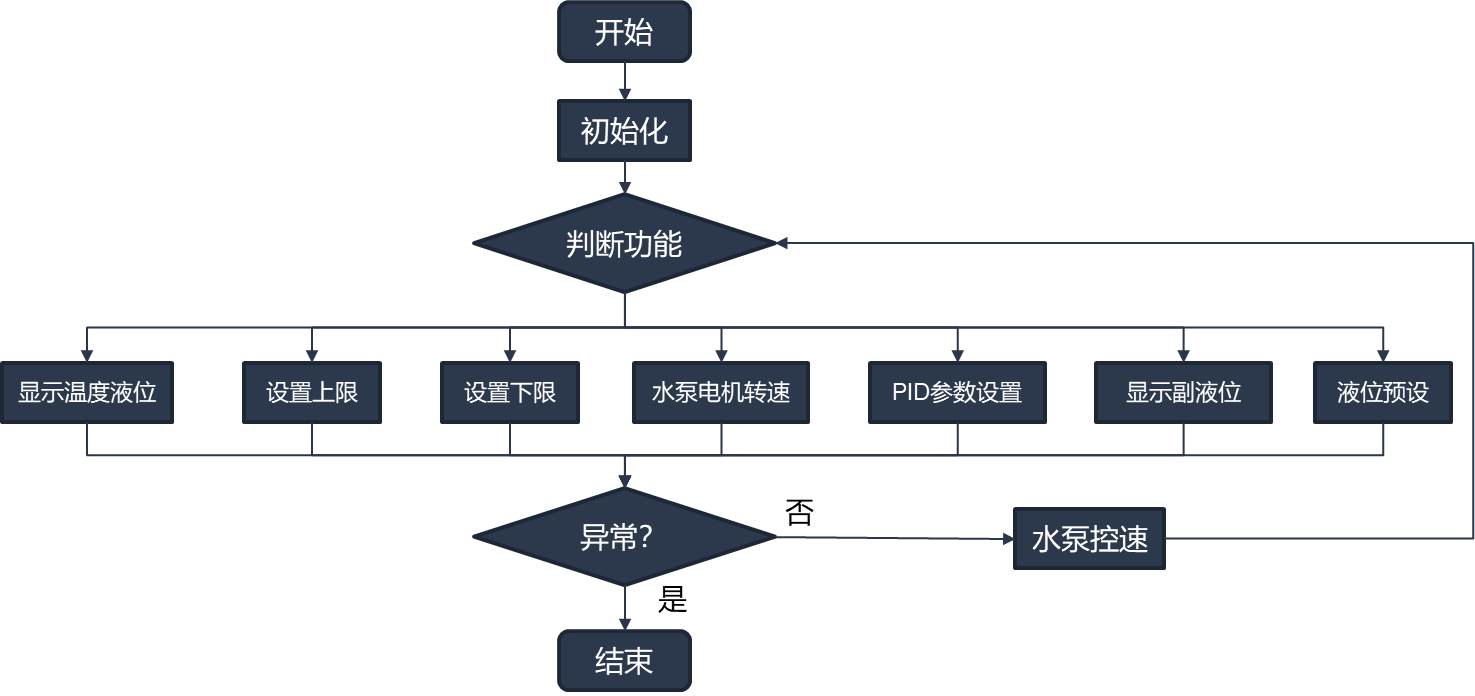


图8：控制系统的流程图。

该系统主要包括的功能，分别是显示主对象液位温度，设置水位的上下限，水泵工作状态显示，PID控制参数设置，显示副对象液位和控制液位高度的预设。其会根据按键中功能按钮的选择而选取不同的功能，而显示参数或者设置参数。在判断异常一栏，可能存在的异常包括人关闭系统和水泵电机温度过高，如果出现此情况电机将断电停止工作，否则根据情况而选择水泵的控制转速，温度报警的流程图如图9所示：

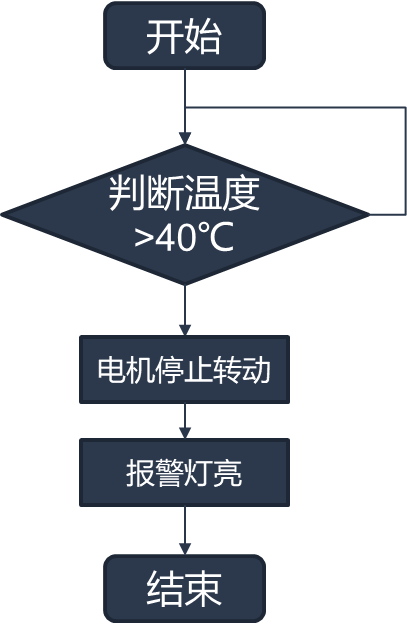


图9：温度报警功能的流程图。

正常情况下水泵只是提供一个水压，需要控制的对象是调节阀，但是可能存在水位超过上限或者低于下限的情况。这里假设水泵电机正转是放水，反转是吸水。当正常工作时，电机正常正转即可，当低于下限位时，水需要快速补充，因此电机加速正转，而当高于上限位时，水过多，因此电机反转。电机状态选择的控制流程图如图10所示：

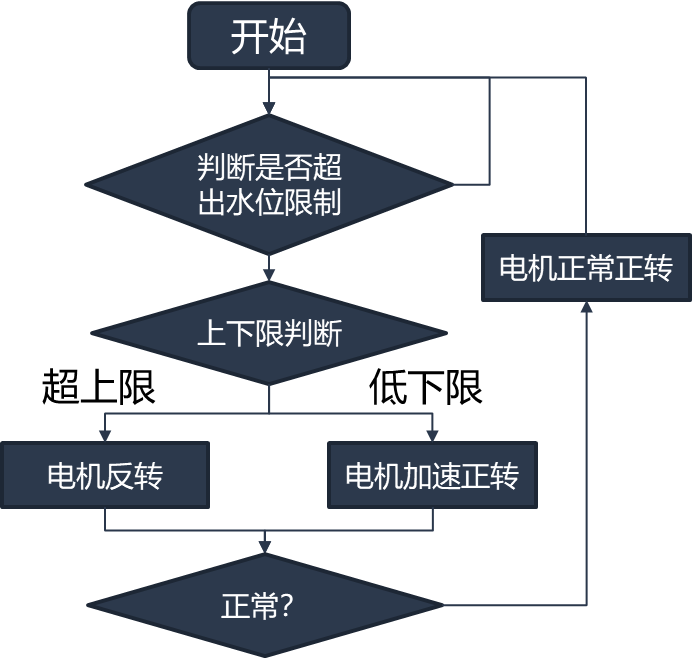


图10：水泵电机转速状态的流程图。

在预设液位时，预设液位不可低于下限也不可高于上限，因此设定时候也需要满足一定的条件，设置的流程图如图11所示：

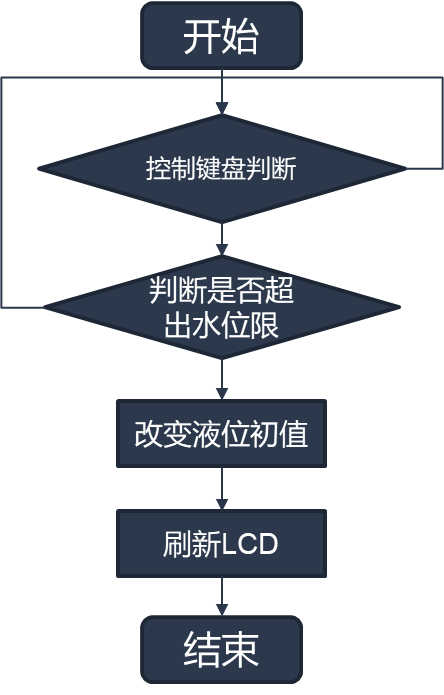


图11：预设液位的流程图。

# 参考文献

[1]侯君虹.基于迭代学习控制的双容水箱液位控制[J].仪器仪表用户,2020,27(06):4-7.

[2]张志军,曹秀爽.模糊PID在液位控制系统中的应用研究[J].电子世界,2020(07):130-132+136.

[3] 周孝信 , 刘向杰 , 柴天佑 . 模糊控制研究的现状与新发展 [M]. 信息与控制,2009,28:283-292.

[4] 李 人 厚 , 张 金 明 . 模 糊 控 制 系 统 化 设 计 和 稳 定 性 分 析 [J]. 自 动 化 学报,2003,315:493-497.

[5] Renotte Ch, Sehab R,Remy M．An approach to design fuzzy PI superisor for a nonlinear system. Annual Conference of the North American Fuzzy Information Processing Society-NAFIPS，2001:894-899．

[6] Pham D T, Li D. Fuzzy control of a three-tank system. Proceedings of the Institution of Mechanical Engineers ,Part C :Journal of Mechanical Engineering Science,2001,215:505-510.

[7] 唐涧涛,王涛,王艳平,佟绍成.模糊控制系统的设计及稳定性分析[M]．科学出版社，2004：24-32．

# 结束语

通过本次的课设，锻炼了我的领导能力和动手能力，我再一次作为组长带领我的团队解决一个大问题。不仅锻炼了我的领导思维能力，同时软件的编程也锻炼了我的逻辑思维能力。我很好的组织协调了整个团队快速解决一个大问题，同时我自己也负责了单片机软件编程方面的东西。编程一定不会一帆风顺，一定会因粗心而发现错误，而我也经常遇到一些问题，例如程序无法调试，与硬件不兼容等。我通过此次成功解决了全部的问题，并再次完成了一个较大的课题项目，这对我以后研究生的学习将会非常有帮助。最后还要感谢老师对我耐心的指导，如果没有老师的实验指导，我很难有较为清晰的思路以及快速解决问题的想法，同时也感谢我的组员对我的积极配合。

# 附录

主程序：

#include <reg51.h>

#include <DS18B20.h>

#include <LCD1602.h>

#include <TLC2543.h>

#define uchar   unsigned char

#define uint    unsigned int

sbit PWM=P2^4;      //电机PWM输出端口

sbit SW=P2^5;           //电机正反转控制端口

sbit K1=P3^0;           //功能设置按键

sbit K2=P3^1;           //加按键

sbit K3=P3^2;           //减按键

sbit K4=P3^7;           //启停键

sbit temperature\_buzzer=P2^6;   //温度警报,40度报警

sbit SM=P2^7;       //电机启停显示

uchar ratio=0;      //PWM占空比

uchar count=0;      //PWM定时计数

uchar func=0;           //系统功能，0当前状态，1-2上下限设置，3电机转速，4-6PID参数

uchar P\_=5;             //PID参数预设

uchar I=2;

uchar D=10;

uchar H\_set=100;        //????

uchar start\_motor=0;    //电机是否启动

uchar high=150,low=50,height=0;

uchar LCD\_Buf1[16]={"W = 000.0cm"};     //液位

uchar LCD\_Buf2[16]={"T = 000.0C"};      //温度

uchar LCD\_Buf3[16]={"H = 150.0cm"};     //液位上限

uchar LCD\_Buf4[16]={"L = 050.0cm"};     //液位下限

uchar LCD\_Buf5[16]={"Full = 384r/m"};   //转速

uchar LCD\_Buf6[16]={"Now  = 000r/m"};   //目前转速

code uchar PID\_display[16]={"PID set"};

code uchar fuBuf[16]={"Liquid level 2:"};

code uchar levelBuf[16]={"Level set:"};

int temperature\_buzzer\_judge=0; //判断是否报警，1为异常

//------------------------------------------------------

//显示温度及液位

//------------------------------------------------------

void Display\_HT(void)

{

    uint volt,t;

    //电压值显示数据处理

    volt=TLC2543(0)\*50.0/4095\*102.3/2.5;

    LCD\_Buf1[0]='W';

    LCD\_Buf1[1]=' ';

    LCD\_Buf1[2]='=';

    LCD\_Buf1[3]=' ';

    LCD\_Buf1[4]=volt/1000+'0';

    LCD\_Buf1[5]=volt/100%10+'0';

    LCD\_Buf1[6]=volt/10%10+'0';

    LCD\_Buf1[7]='.';

    LCD\_Buf1[8]=volt%10+'0';

    LCD\_Buf1[9]='c';

    LCD\_Buf1[10]='m';

    LCD\_Buf1[15]=' ';

    //温度值显示数据处理

    t=ReadTemperature();

    LCD\_Buf2[0]='T';

    LCD\_Buf2[1]=' ';

    LCD\_Buf2[2]='=';

    LCD\_Buf2[3]=' ';

    LCD\_Buf2[4]=t/1000+'0';

    LCD\_Buf2[5]=t/100%10+'0';

    LCD\_Buf2[6]=t/10%10+'0';

    LCD\_Buf2[8]=t%10+'0';

    if(t>400){

        temperature\_buzzer=0;   //温度超过40度报警

        temperature\_buzzer\_judge+=1;

    }

    else{

        temperature\_buzzer=1;   //正常

        temperature\_buzzer\_judge=0;

    }

    LCD\_Buf2[15]='A';

    //刷新LCD数据

    LCD\_Display\_String(0,0,LCD\_Buf1);

    LCD\_Display\_String(0,1,LCD\_Buf2);

}

//------------------------------------------------------

//显示副液位

//------------------------------------------------------

void Display\_FH(void)

{

    uint volt;

    //电压值显示数据处理

    volt=TLC2543(1)\*50.0/4095\*102.3/2.5;

    LCD\_Buf1[0]='W';

    LCD\_Buf1[1]=' ';

    LCD\_Buf1[2]='=';

    LCD\_Buf1[3]=' ';

    LCD\_Buf1[4]=volt/1000+'0';

    LCD\_Buf1[5]=volt/100%10+'0';

    LCD\_Buf1[6]=volt/10%10+'0';

    LCD\_Buf1[7]='.';

    LCD\_Buf1[8]=volt%10+'0';

    LCD\_Buf1[9]='c';

    LCD\_Buf1[10]='m';

    LCD\_Buf1[15]='H';

    //刷新LCD数据

    LCD\_Display\_String(0,0,fuBuf);

    LCD\_Display\_String(0,1,LCD\_Buf1);

}

//------------------------------------------------------

//上限设置程序

//------------------------------------------------------

void High\_Set()

{

    uint x;

    if(K2==0)               //加按键

    {   Delay(10);

        if((K2==0)&&(high<200))

            high++;

        while(K2==0);

    }

    if(K3==0)               //减按键

    {   Delay(10);

        if((K3==0)&&(high>low))

            high--;

        while(K3==0);

    }

    //上限显示数据处理

    x=high\*10;

    LCD\_Buf3[4]=x/1000+'0';

    LCD\_Buf3[5]=x/100%10+'0';

    LCD\_Buf3[6]=x/10%10+'0';

    LCD\_Buf3[8]=x%10+'0';

    LCD\_Buf3[9]='c';

    LCD\_Buf3[10]='m';

    LCD\_Buf4[15]='B';

    //刷新LCD数据

    LCD\_Display\_String(0,0,LCD\_Buf3);

    LCD\_Display\_String(0,1,LCD\_Buf4);

}

//------------------------------------------------------

//下限设置程序

//------------------------------------------------------

void Low\_Set()

{

    uint x;

    if(K2==0)               //加按键

    {   Delay(10);

        if((K2==0)&&(low<high))

            low++;

        while(K2==0);

    }

    if(K3==0)               //减按键

    {   Delay(10);

        if((K3==0)&&(low>1))

            low--;

        while(K3==0);

    }

    //下限显示数据处理

    x=low\*10;

    LCD\_Buf4[4]=x/1000+'0';

    LCD\_Buf4[5]=x/100%10+'0';

    LCD\_Buf4[6]=x/10%10+'0';

    LCD\_Buf4[8]=x%10+'0';

    LCD\_Buf4[9]='c';

    LCD\_Buf4[10]='m';

    LCD\_Buf4[15]='C';

    //刷新LCD数据

    LCD\_Display\_String(0,0,LCD\_Buf3);

    LCD\_Display\_String(0,1,LCD\_Buf4);

}

//------------------------------------------------------

//显示电机全速与当前转速

//------------------------------------------------------

void Display\_Motor()

{

    uint x;

    //电机当前转速显示数据处理

    switch(ratio)

    {

        case 100:   x=384;break;

        case 43:    x=192;break;

        case 26:    x=115;break;

        case 0:     x=0;

    }

    LCD\_Buf6[7]=x/100+'0';

    LCD\_Buf6[8]=x/10%10+'0';

    LCD\_Buf6[9]=x%10+'0';

    LCD\_Buf6[15]='D';

    //刷新LCD数据

    LCD\_Display\_String(0,0,LCD\_Buf5);

    LCD\_Display\_String(0,1,LCD\_Buf6);

}

//------------------------------------------------------

//PID参数设置

//------------------------------------------------------

void P\_set()

{

    if(K2==0)               //加按键

    {   Delay(10);

        if((K2==0)&&(P\_<999))

            P\_++;

        while(K2==0);

    }

    if(K3==0)               //减按键

    {   Delay(10);

        if((K3==0)&&(P\_>0))

            P\_--;

        while(K3==0);

    }

    LCD\_Buf1[15]='E';

    //刷新LCD数据

    LCD\_Buf1[0]='P';

    LCD\_Buf1[1]=' ';

    LCD\_Buf1[2]='=';

    LCD\_Buf1[3]=' ';

    LCD\_Buf1[4]=P\_/100+'0';

    LCD\_Buf1[5]=P\_/10%10+'0';

    LCD\_Buf1[6]=P\_%10+'0';

    LCD\_Buf1[7]=' ';

    LCD\_Buf1[8]=' ';

    LCD\_Buf1[9]=' ';

    LCD\_Buf1[10]=' ';

    LCD\_Display\_String(0,0,PID\_display);

    LCD\_Display\_String(0,1,LCD\_Buf1);

}

void I\_set()

{

    if(K2==0)               //加按键

    {   Delay(10);

        if((K2==0)&&(I<999))

            I++;

        while(K2==0);

    }

    if(K3==0)               //减按键

    {   Delay(10);

        if((K3==0)&&(I>0))

            I--;

        while(K3==0);

    }

    LCD\_Buf1[15]='F';

    //刷新LCD数据

    LCD\_Buf1[0]='I';

    LCD\_Buf1[1]=' ';

    LCD\_Buf1[2]='=';

    LCD\_Buf1[3]=' ';

    LCD\_Buf1[4]=I/100+'0';

    LCD\_Buf1[5]=I/10%10+'0';

    LCD\_Buf1[6]=I%10+'0';

    LCD\_Buf1[7]=' ';

    LCD\_Buf1[8]=' ';

    LCD\_Buf1[9]=' ';

    LCD\_Buf1[10]=' ';

    LCD\_Display\_String(0,0,PID\_display);

    LCD\_Display\_String(0,1,LCD\_Buf1);

}

void D\_set()

{

    if(K2==0)               //加按键

    {   Delay(10);

        if((K2==0)&&(D<999))

            D++;

        while(K2==0);

    }

    if(K3==0)               //减按键

    {   Delay(10);

        if((K3==0)&&(D>0))

            D--;

        while(K3==0);

    }

    LCD\_Buf1[15]='G';

    //刷新LCD数据

    LCD\_Buf1[0]='D';

    LCD\_Buf1[1]=' ';

    LCD\_Buf1[2]='=';

    LCD\_Buf1[3]=' ';

    LCD\_Buf1[4]=D/100+'0';

    LCD\_Buf1[5]=D/10%10+'0';

    LCD\_Buf1[6]=D%10+'0';

    LCD\_Buf1[7]=' ';

    LCD\_Buf1[8]=' ';

    LCD\_Buf1[9]=' ';

    LCD\_Buf1[10]=' ';

    LCD\_Display\_String(0,0,PID\_display);

    LCD\_Display\_String(0,1,LCD\_Buf1);

}

//------------------------------------------------------

//液位初设

//------------------------------------------------------

void Level\_setting()

{

    if(K2==0)               //加按键

    {   Delay(10);

        if((K2==0)&&(H\_set<high))

            H\_set++;

        while(K2==0);

    }

    if(K3==0)               //减按键

    {   Delay(10);

        if((K3==0)&&(H\_set>low))

            H\_set--;

        while(K3==0);

    }

    LCD\_Buf1[15]='I';

    //刷新LCD数据

    LCD\_Buf1[0]='H';

    LCD\_Buf1[1]=' ';

    LCD\_Buf1[2]='=';

    LCD\_Buf1[3]=' ';

    LCD\_Buf1[4]=H\_set/100+'0';

    LCD\_Buf1[5]=H\_set/10%10+'0';

    LCD\_Buf1[6]=H\_set%10+'0';

    LCD\_Buf1[7]=' ';

    LCD\_Buf1[8]='c';

    LCD\_Buf1[9]='m';

    LCD\_Buf1[10]=' ';

    LCD\_Display\_String(0,0,levelBuf);

    LCD\_Display\_String(0,1,LCD\_Buf1);

}

//------------------------------------------------------

//K1按键扫描及系统功能

//------------------------------------------------------

void Func\_Choice()

{

    if(K1==0)

    {   Delay(10);

        if(K1==0)

            func=(++func)%9;

        while(K1==0);

    }

    switch(func)

    {

        case 0: Display\_HT();break;

        case 1: High\_Set();break;

        case 2: Low\_Set();break;

        case 3: Display\_Motor();break;

        case 4: P\_set();break;

        case 5: I\_set();break;

        case 6: D\_set();break;

        case 7: Display\_FH();break;

        case 8: Level\_setting();break;

    }

}

//------------------------------------------------------

//电机转速和转向控制

//------------------------------------------------------

void Motor()

{

    uchar l;

    l=TLC2543(0)\*5.0/4095\*102.3/2.5;

    if(l<low)           //低于下限正转

    {

        SW=0;

        if(l<low\*0.1)   ratio=100;

        if((l>=low\*0.1)&&(l<low\*0.3))   ratio=42;

        if(l>=low\*0.3)  ratio=26;

    }

    else if(l>high)     //高于上限反转

    {

        SW=1;

        if(l<high+(204.7-high)\*0.7)     ratio=26;

        if((l>=high+(204.7-high)\*0.7)&&(l<high+(204.7-high)\*0.9))   ratio=43;

        if(l>=high+(204.7-high)\*0.9)    ratio=100;

    }

    else

        ratio=0;

}

//------------------------------------------------------

//K4启停按键扫描

//------------------------------------------------------

void motor\_key()

{

    uint t;

    t=ReadTemperature();

    if(t>400)

    {

        start\_motor=0;

        SM=1;

    }

    if(K4==0&&t<400)

    {   Delay(10);

        if(K4==0)

            start\_motor=~start\_motor;

            SM=~SM;

        while(K4==0);

    }

}

//------------------------------------------------------

//定时中断初始化程序

//------------------------------------------------------

void T0\_Init()

{

    TMOD=0x01;                  //设置T0工作方式2

    TH0=(65535-500)/256;        //定时1000us

    TL0=(65536-500)%256;

    EA=1;                       //开T0中断

    ET0=1;

    TR0=1;                      //启动定时

}

//------------------------------------------------------

//T0中断服务程序

//------------------------------------------------------

void Timer0()   interrupt 1

{

    TH0=(65535-500)/256;        //定时1000us

    TL0=(65536-500)%256;

    if(ratio==0)    PWM=1;

    else if(ratio==100) PWM=0;

    else

    {   count++;

        if(count==100)  count=0;

        if(count==1)        PWM=0;

        else if(count==(ratio+1))   PWM=1;

    }

}

//------------------------------------------------------

//基本数据初始化

//------------------------------------------------------

void base\_init()

{

    temperature\_buzzer=1;   //温度报警模块清零

}

void main()

{

    base\_init();    //基本数据初始化

    LCD\_Init();     //液晶显示器初始化

    T0\_Init();      //中断初始化

    while(1)

    {

        motor\_key();

        Func\_Choice();      //系统功能选择

        if(start\_motor==0){

            ratio=0;

            PWM=1;

        }

        else

            Motor();            //电机控制

    }

}

DS18B20头文件：

#include <REG51.h>

sbit  DQ=P2^3;    //定义DS18B20总线I/O

/\*\*\*\*\*延时子程序\*\*\*\*\*/

void Delay\_DS18B20(int num)

{

  while(num--) ;

}

/\*\*\*\*\*初始化DS18B20\*\*\*\*\*/

void Init\_DS18B20(void)

{

  unsigned char x=0;

  DQ=1;         //DQ复位

  Delay\_DS18B20(8);    //稍做延时

  DQ = 0;         //单片机将DQ拉低

  Delay\_DS18B20(80);   //精确延时，大于480us

  DQ = 1;         //拉高总线

  Delay\_DS18B20(14);

  x = DQ;           //稍做延时后，如果x=0则初始化成功，x=1则初始化失败

  Delay\_DS18B20(20);

}

/\*\*\*\*\*读一个字节\*\*\*\*\*/

unsigned char ReadOneChar(void)

{

  unsigned char i=0;

  unsigned char dat = 0;

  for (i=8;i>0;i--)

  {

    DQ = 0;     // 给脉冲信号

    dat>>=1;

    DQ = 1;     // 给脉冲信号

    if(DQ)

    dat|=0x80;

    Delay\_DS18B20(4);

  }

  return(dat);

}

/\*\*\*\*\*写一个字节\*\*\*\*\*/

void WriteOneChar(unsigned char dat)

{

  unsigned char i=0;

  for (i=8; i>0; i--)

  {

    DQ = 0;

    DQ = dat&0x01;

    Delay\_DS18B20(5);

    DQ = 1;

    dat>>=1;

  }

}

/\*\*\*\*\*读取温度\*\*\*\*\*/

unsigned int ReadTemperature(void)

{

  unsigned char a=0;

  unsigned char b=0;

  unsigned int t=0;

  float tt=0,ttt;

  TR0=0;

  Init\_DS18B20();

  WriteOneChar(0xCC);  //跳过读序号列号的操作

  WriteOneChar(0x44);  //启动温度转换

  Init\_DS18B20();

  WriteOneChar(0xCC);  //跳过读序号列号的操作

  WriteOneChar(0xBE);  //读取温度寄存器

  a=ReadOneChar();     //读低8位

  b=ReadOneChar();    //读高8位

  t=b;

  t<<=8;

  t=t|a;

  tt=t\*0.0625;

  ttt=tt\*10+0.5;     //放大10倍输出并四舍五入

  TR0=1;

  return(ttt);

}

/\*\*\*\*\*END\*\*\*\*\*/

LCD1602头文件：

#include <reg51.h>

#define uchar   unsigned char

#define uint    unsigned int

#define DP  P0              //LCD数据I/O口

sbit RS=P2^0;       //LCD寄存器选择

sbit RW=P2^1;       //LCD读/写控制

sbit EN=P2^2;       //LCD使能端

//------------------------------------------------------

//延时函数（n ms）

//------------------------------------------------------

void Delay(uint n)

{

    uchar i;

    uint j;

    for(j=0;j<n;j++)

        for(i=0;i<120;i++);

}

//------------------------------------------------------

//LCD1602忙检查

//------------------------------------------------------

uchar LCD\_Busy\_Check()

{

    uchar LCD\_Status;

    RS=0;

    RW=1;

    EN=1;

    Delay(1);

    LCD\_Status=DP;

    EN=0;

    return LCD\_Status;

}

//------------------------------------------------------

//LCD1602写命令函数

//------------------------------------------------------

void LCD\_Write\_Cmd(uchar cmd)

{

    while((LCD\_Busy\_Check()&0x80)==0x80);   //忙等待

    RS=0;   //选择命令寄存器

    RW=0;   //写

    EN=0;

    DP=cmd;

    EN=1;

    Delay(1);

    EN=0;

}

//------------------------------------------------------

//LCD1602写数据函数

//------------------------------------------------------

void LCD\_Write\_Data(uchar dat)

{

    while((LCD\_Busy\_Check()&0x80)==0x80);   //忙等待

    RS=1;

    EN=0;

    RW=0;

    DP=dat;

    EN=1;

    Delay(1);

    EN=0;

}

//------------------------------------------------------

//LCD1602初始化函数

//------------------------------------------------------

void LCD\_Init()

{

    LCD\_Write\_Cmd(0x38);        //三次显示模式设置，不检测忙信号

    Delay(5);

    LCD\_Write\_Cmd(0x38);

    Delay(5);

    LCD\_Write\_Cmd(0x38);

    Delay(5);

    LCD\_Write\_Cmd(0x38);        //显示模式设置，开始要求检测忙信号：8位、2行、5X7点阵

    Delay(5);

    LCD\_Write\_Cmd(0x01);        //清屏

    Delay(5);

    LCD\_Write\_Cmd(0x06);        //字符进入模式:屏幕不动，字符后移

    Delay(5);

    LCD\_Write\_Cmd(0x0C);        //显示开，光标关

    Delay(5);

}

//------------------------------------------------------

//LCD1602显示一行字符串函数

//x:显示起始列号,0~15

//y:显示起始行号,0:第一行,1:第二行

//\*str：字符串指针

//------------------------------------------------------

void LCD\_Display\_String(uchar x,uchar y,uchar \*str)

{

    uchar i=0;

    //设置显示起始位置

    if(y==0)    LCD\_Write\_Cmd(0x80|x);

    if(y==1)    LCD\_Write\_Cmd(0xC0|x);

    //显示字符串

    for(i=0;i<16;i++)

    {

        LCD\_Write\_Data(str[i]);

        Delay(1);

    }

}

TLC2543头文件：

#include <reg51.h>

sbit SDO=P1^0;      //TLC2543数据输入端

sbit SDI=P1^1;      //TLC2543数据输出端

sbit CS=P1^2;       //TLC2543片选端

sbit CLK=P1^3;      //TLC2543时钟

#define uchar   unsigned char

#define uint    unsigned int

//------------------------------------------------------

//延时函数（n ms）

//------------------------------------------------------

void Delay\_AD(uint n)

{

    uchar i;

    uint j;

    for(j=0;j<n;j++)

        for(i=0;i<120;i++);

}

//------------------------------------------------------

//TLC2543 AD转换程序

//port:转换通道号，0~10

//------------------------------------------------------

uint TLC2543(uchar port)

{

    uint i;

    uint ad\_value=0;

    CLK=0;

    CS=0;

    port<<=4;

    //读取12位串行转换结果

    for(i=0;i<12;i++)

    {

        if(SDO) ad\_value|=0x01;

        SDI=(bit)(port&0x80);

        CLK=1;

        Delay\_AD(1);

        CLK=0;

        Delay\_AD(1);

        port=port<<1;

        ad\_value=ad\_value<<1;

    }

    CS=1;

    ad\_value=ad\_value>>1;

    return ad\_value;

}