基于单片机及 AD590 的恒温控制系统的设计

摘要

本设计是以单片机为核心，由控制部分、显示部分和温度测量部分组成。该系统的绝大部分功能都能通过硬件来实现，电路简单明了，系统的稳定性很高。

这套温度控制系统可以方便地实现温度测量、温度显示等功能，并通过与单片机连接的键盘可以实时设定测控温度的下限，还可以连接相应的外围电路，在收到单片机发出的指令后对环境进行检测。

本文首先描述系统硬件的工作原理，并附以系统结构图加以说明，着重介绍了本系统所应用的各硬件模块的功能和它的工作过程；其次，详细阐述了程序的各个模块及其实现过程。本系统的主要设计思想是以硬件为基础，软件和硬件相结合，最终实现各个模块的功能。

关键词：单片机，温度采集，硬件电路

Based microcontroller and AD590 temperature control system design

Abstract

This design is a microcontroller as the core, by the control portion, a display portion and temperature measurement components. Most of the functionality of the system can be realized by hardware circuit is simple and clear, high stability of the system.

This temperature control system can easily achieve temperature measurement, temperature display and other functions, and through the keyboard can be connected with the microcontroller real-time monitoring and control of temperature set limit, you can also connect the corresponding peripheral circuit, microcontroller received the directives issued after testing environment.

This paper first describes the system hardware works, accompanied by a system structure is described, highlighting the application of this system function of each hardware module and its working process; secondly, elaborated on the various modules of the program and its implementation process . The system's main design idea is based on the hardware, software and hardware combination, and ultimately the function of each module.

Keywords: microcontroller, temperature acquisition, the hardware circuit

目录

第一章.绪论

1.1 课题的背景

本文介绍了一个基于单片机的温度控制系统,该系统可以方便地实现温度采集、温度显示等功能。本系统的温度控制部分采用单片机完成。单片机有着体积小、功耗低、功能强、性能价格比高、使用电子元件较少、内部配线少、制造调试方便等显著优点，将其用于温度检测和控制系统中可大大地提高控制质量和自动化水平，具有良好的经济效益和推广价值。利用单片机对温度进行测控的技术，日益得到广泛应用。

在日常生活中我们用到此系统的实物要是变频空调。中国变频空调技术的应用分为三个阶段：引进变频技术、学习技术、自主创新达，即成长到快速发展。

起步阶段（1995 年—1999 年）

1995 年 4 月，海信成立空调研究小组，对国内外空调行业技术进行研究；次年，海信投资率先引进国外先进的变频技术和空调生产线，建立了国内第一家变频空调生产基地。中国变频空调历史由此开始；1999 年，我国第一台直流变频空调正式销向市场，变频空调迈入直流变频的新时代。

成长阶段（2000 年—2007 年）

2000 年，变频空调行业完成了初步的引进技术学习、技术的积累阶段，正式步入发展的征程；2007 年 4 月 19 日，在上海成立了中国首个变频空调推广联盟，提出了变频联盟未来 10 年目标：加速普及变频空调，并且慢慢淘汰定速空调。

快速发展阶段（2008 年—至今）

008 年 4 月，中国“变频空调推广联盟”成立 1 周年的会上，该联盟先后又得到了美的、海尔、格力等巨头的肯定和加入，变频联盟的组成成员由 2007 年的 5 家增至 12 家；2008 年 7 月，全国家用自动控制器标准化技术委员会正式成立，秘书处承担单位为海信集团，负责变频控制器领域国家标准制修订工作；2008 年 9 月，变频空调国家标准出台后，美的、格力、海尔等定速空调生

产厂家纷纷宣布大举进入变频空调的生产领域。中国变频空调发展进入快速增长阶段。

1.2 课题的意义

①变频空调通过内装变频器，不断调节空调机的核心——压缩机的转速，从而做到能源的合理使用；以为他的压缩机不会频繁启动，所以压缩机会一直处于稳定状态，这可以使空调整体节能的效果达到 30%以上。

②噪音的降低是由于变频空调的运转平衡，震动减小。

③空调机制冷（热）的量可以通过改变压缩机的转速来控制，其制冷（热）量是一个变化的值，通过外部温度的变化而变化，其精确度可达到 1%误差。

④当室温和设定温度相差较大时，压缩机会全速运转，温度会以极快的速度达到调节温度，你会发现制热或制冷效果非常明显。

⑤变频空调的电压适应性很强，甚至在 140— 240V 电压下有些变频空调就可以启动。

⑥环境温度对变频空调的影响不高，在-15t 的恶劣环境下有些变频空调也可以启动。

⑦变频空调的核心就是其变频压缩机，其变频压缩机可以根据房间的温度变化而改变其中电机的转速。变频空调达到设定温度后会以较低的速度运转，从而使得室温的变化不会那么剧烈。

1.3 确定设计方案

测量温度的方法多种多样，测温传感器是决定技术指标的关键元器件，本文使用 AD590 作为温度传感器，再通过模数转换把模拟信号转成数字信号，送入单片机进行处理并显示。我们还可以通过按键来改变单片机中的处理参数，以此来达到对温度的设定与控制。

1.4 课题研究的内容及要求

本课题主要阐述了变频空调的工作原理，包括了硬件电路的设计和对应程

序的编写。其硬件方面主要有：ADC0832（并口 A/D 转换），AD590（测温模块），运放，TLC5615（串口 D/A 转换），带字库12864液晶（并口）等。而程序的核心主要是 PID 调节。

要求：1.能够监控温度，能连续的测量当前的温度值并显示

2.能够设定预期到达的温度值

3.能够恒温，即将温度保持在设定温度

第 2 章.元器件选择的介绍

2.1.AT89C51 系列单片机介绍

AT89C51 是一种具有 4K 字节的 FLASH 存储器的 CMOS 8 位微处理器。单片机可以反复擦除只读存储器 1000 多次。单片机采用高密度非易失的存储器制造技术制造，并且与工业上标准的 MCS-51 的指令集和管脚相兼容。单片机将闪烁存储器以及多功能 8 位 CPU 糅合在一起，其集成芯片是一种高效的具有微控制功能的器件。ATM89C2051 是各种单片机版本的一种精简版。AT89C51 芯片能够作为很多嵌入式系统开发的一种灵活性高且价格便宜的方案。外形及引脚排列如图 2-1 所示。

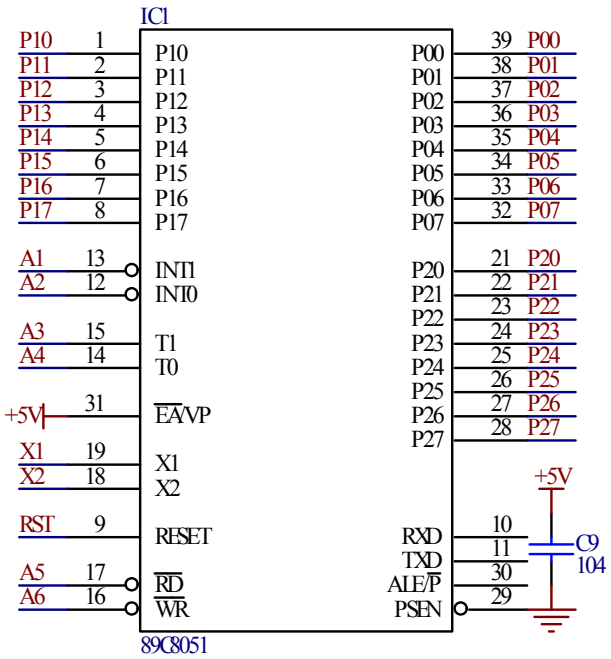


图 2-1 89C51 单片机引脚图

主要特性:

⑴与 MCS-51 兼容

⑵可编程的4K 字节 FLASH 存储器

⑶寿命：1000读写/擦循环

⑷可保留的数据时间：10年

⑸全静态的工作频率：0Hz-24MHz

⑹可锁定三级的程序存储器

⑺128×8位内部 RAM

⑻32条可编程 I/O 线

⑼16位的定时器两个/计数器

⑽5个中断源

⑾可编程串行通道

⑿低功耗的掉电模式

⒀单片机片内具有时钟电路和振荡器

管脚说明：

VCC：芯片供电电压源。

GND：接地。

P0口：P0口具有8位开路的双向 I/O 口，这个端口的每个脚可接收8个 TTL 门电流信号。当 P0口管脚第一次被定义为1时，它的管脚口状态为高组态。P0口能够定义和用作数据或地址的低8位，并且它还能储存这些数据和地址。在ATM89C51芯片编程时，P0能够被当作源码的输入口，当芯片进行烧录校检时，如果此时 P0输出原码，那么 P0外部一定要接上拉电阻，这里可以使用排阻。

P1口：P1口不用像 P0口一样外连上拉电阻，因为其内部已经有了上拉电阻；和 P1口一样，它也是一个具有8位开路的双向 I/O 口。当 P1口管脚应以为1时，内部上拉电阻为高电频，这时可以输入数据，P1口外部是下拉电阻时为低电平，会有电流输出，这是应为内部有上拉电阻的缘故。在进行 FLASH 校验和编程时，P1口作为低八位地址接收。

P2口：P2口缓冲器可接收，输出4个 TTL 门电流，当 P2口被写“1”时，其管脚被内部上拉电阻拉高，且作为输入。并因此作为输入时，P2口的管脚被外部拉低，将输出电流。这是由于内部上拉的缘故。P2口当用于外部程序存储器或16位地址外部数据存储器进行存取时，P2口输出地址的高八位。在给出地址

“1”时，它利用内部上拉优势，当对外部八位地址数据存储器进行读写时，P2口输出其特殊功能寄存器的内容。P2口在FLASH编程和校验时接收高八位地址信号和控制信号。

P3口：P3口管脚都是双向I/O口并且其内部带有8个上拉电阻，可以输入和输出4个TTL门电流信号。当P3口为“1”时，能输入数据。作为输入，由于外部有下拉电阻并且为低电平，P3口会输出电流是其内部上拉电阻引起的。

RST：复位。如果振荡器处于复位阶段，我们一定要使此端口保持有2个高电频的机器周期。

/PSEN：外部程序存储器的选通信号。在由外部程序存储器取指期间，每个机器周期两次/PSEN有效。但在访问外部数据存储器时，这两次有效的/PSEN信号将不出现。

/EA/VPP：当/EA保持低电平时，则在此期间外部程序存储器（0000H-FFFFH），不管是否有内部程序存储器。注意加密方式1时，/EA将内部锁定为RESET；当/EA端保持高电平时，此间内部程序存储器。在FLASH编程期间，此引脚也用于施加12V编程电源（VPP）。

XTAL1:反向振荡放大器的输入及内部时钟工作电路的输入。

XTAL2:来自反向振荡器的输出。

2.2 运算放大器

温度测量电路电路和输出电路共需要两路运算放大电路，所以选择双路运算放大器，一路作电压跟随器另一路作同向放大器。运算放大器是模拟集成电路中应用极为广泛的的一种器件，它不仅用于信号的运算、处理、变换、测量和信号产生电路，而且还可以用于开关电路中。运算放大器作为基本的电子原件，虽然本身具有非线性的特性，但在许多情况下，它作为线性电路的器件，很容易用来设计各种应用电路。本论文使用的是型号为LM358，其封装为8引线双列直插式DIP8。

型号为LM358运放的特性:

内部自动补偿频率；

有高增益的直流电压；

单位的增益频带宽约 1MHz；

单电源电压范围：3—30V；

双电源：±1.5 一±15V；

电流功耗低；

共模输入和接地的电压选择范围广；

差模输入和电源电压选择范围广；

输出电压:0 到-1.5V。

为了减少测量电压时对 AD590 的电流分流所以本设计先用一路放大器作为跟随器，电压跟随器的使用优点就是：输入的阻抗高，而输出阻的抗低。所以说，我们比较容易做到的是增大输入阻抗，可以使输入阻抗达到几兆欧姆。同时我们还要降低输出阻抗，最低能使输出阻抗小到几欧姆。

在本次温度采集及控制电路中，使用了两路电压跟随器。 电压跟随器在电路中起到了承上启下的作用，它可以把电压信号完整的从原电路中提取出来，然后供下级的放大器使用；这里它可以有效的防止电路中的干扰，隔离前级电路和下级电路的互相干扰，以及负载变化对输出电压的影响。

AD590 是恒流输出，其输出电流刚好是 1uA/K。在电路中用 10K 的电阻跟 AD59 串连，因此电阻两电压刚好就是 0.01V/K。在零摄氏度时电阻两端的电压为 2.73V 然而模数转换 ADC0832 的输入电压为 0-5V，分辨率为 0.19。精度比较低，如果电压跟随直接与模数转换模块直接相连就会有很大误差。所以还要经过差分放大电路把电压放大 10 倍。具体是把电压跟随器输出电压与一个标准的 2.73V 的相减然后再放大 10 倍。这样做之后温度每改变一摄氏度电压就改变 0.1V。这样就可以送入模数转换模块进行转换了。该电路是用两个输入信号的差值的绝对值作为有效输入信号，然后再把差值放大输出。如果原信号到干扰，那么这两个输入信号都是受过同一干扰，但我们需要的有效输入信号为两信号之差，所以干扰信号在输入信号处理时已经互消为 0，这就是所谓的抗共模干扰。一种单晶体管电流镜像与适当的负载相接合，其中结合了适当的开关集合，以实现比较器功能。具体地，差分电路包括单晶体管电流镜像，所述单晶体管电流镜像包括通过开关与晶体管相连的电容器以及通过各自独立的开关与电流镜像相连的两个电流源，与电容器开关一起操作电流源之一的开关，以便充电

电容器，并且操作另一个电流源的开关，以便所述电路作为具有电流源负载的源极跟随放大器进行操作。因此，晶体管特性的空间分布不会影响比较器功能。

2.3ADC0832 模/数转换器

型号为 ADC0832 芯片的特性:

⑴8 位分辨率

⑵逐次逼近式 A/D 转换器

⑶双通道 A/D 转换；

⑷输出输如的高低电平与 TT 和 CMOS 相兼容；

⑸电源的供电输入电压为 0~5V；

⑹其工作频率和转换时间为 250KH、32μS；

⑺功耗为 15mw；

芯片接口说明：

⑴CS_片选使能，芯片低电平使能信号。

⑵CH0 模拟量输入口 0。

⑶CH1 模拟量输入口 1。

⑷GND 接地参考电位。

⑸DI 输入数据信号，通道选择控制。

⑹DO 输出数据信号，转换后的数据输出。

⑺CLK 单片机同步时钟信号输入。

⑻Vcc/REF 输入参考及输入电源电压。

ADC0832 是 A/D 转换的芯片，其分辨率为 8 位，此芯片的分辨率最高为 256 级，可以满足大部分的模数转换要求。因为芯片内部参考输入与电源电压的复用，所以模拟量的输入电压信号只能在 0~5V 内变化。

ADC0832 与单片机共有 4 条公共接口，在 ADC0832 芯片上分别对应的是：CLK、DI、CS、DO。但又因为 DI 端与 DO 端在单片机通信传输上并没有冲突且都是双向传输，所以 DI 和 DO 端可以连接到同一单片机端口。ADC0832 没进入工作状态时，其 CS 输入应该呈高电平输入，此时 ADC0832 不可以使用，DO/DI 和

CLK 的端口可以处于任意状态。如果要进行 A/D 转，那么须先将 CS 端口一直置低电位，并且保持到 AD 转换结束。此时芯片开始转换工作，同时由处理器向芯片时钟输入端 CLK 输入时钟脉冲，DO/DI 端则使用 DI 端输入通道功能选择的数据信号。

2.4.TLC5615 数/模转换器

型号为 TLC5615 芯片的特性:

⑴DIN：输入串行数据接口；

⑵SCLK：输入串行时钟信号接口；

⑶CS　:片选信号口，低电平触发有效；

⑷DOUT：级联时串行数据输出口；

⑸AGND：模拟地；

⑹REFIN：提供芯片基准电压；

⑺OUT：DAC 转换后模拟电压的输出口；

⑻VDD：提供芯片电源电压。

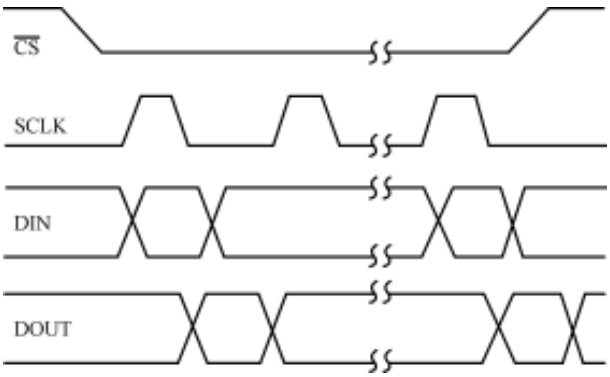


图 2-2 数/模转换时序图

由图 2-2 可以看出，CS 片选信号为持续低电平时，DIN 脚输入数据由 SCLK 脚输入的时钟信号进行同步的输入或输出，有效的高 4 位数据在前，低 4 位数据在后。输入数据时当 SCLK 脚的上升沿信号响应后 DIN 脚的串行输入数据会移入芯片内的 16 位移位寄存器，SCLK 脚的下降沿信号响应后 DOUT 脚会输出串行数据，CS 片选端口的上升沿信号响应后会把数据传送到芯片的 DAC 寄存器。当 CS 片选端口保持高电平时，DIN 端口的串行输入数据不能通过时钟的同步信号送入芯片的移位寄存器内。所以我们要使串行输入和输出有效必须满足以下两

个要求：第一点时钟信号 SCLK 端口的有效跳变触发；第二点 CS 片选信号保持为低电平。

TLC5615 是串行的数模转换芯片，它的工作方式有两种：非级联和级联。如果不用级联的工作方式，DIN 端口仅仅是要输入 12 位的数据。 而 DIN 端口输入的 12 的位数据中，TLC5615 芯片输入需要进行 D/A 转的换数据为前 10 位，而且输入数据时高 5 位在前、低 5 位在后，最后的两位要写入低于 LSB 位且这位数值为零，这是由于 数模转换芯片的 DAC（输入锁存器）只有 12 位宽。如果用 TL5615 的级联功工作方式，那么 DOUT 端口的数据要有 16 位下降沿触发时钟脉冲 ，因此要完成一次数据的传输并需提供 16 个时钟触发脉冲，而输入的数据也就应该为 16 位。

2.5AD590 温度传感器

AD590 是美国模拟器件公司生产的单片集成两端感温电流源。

它的主要特性如下：

（ 1 ）测温范围- 55℃~+150℃；

（ 2 ）线性电流输出 1μA / K；

（ 3 ）线性度好，满刻度范围为±0 .3℃；

（ 4 ）电源电压范围 4 ~ 30 V，当电源电压在 5 ~10V 之间，电压稳定度为 1 %时，所产生的误差只有±0.01℃；

（ 5 ）电阻采用激光修刻工艺：

（ 6 ）功率的损耗很低。

AD590 的工作原理:

AD590 能和温度成正比关系是因为采用了硅晶体管自身的基本性能，二极管器件的方程为：

$$I = I_s\left(e^{qV_{bc}/KT}-1\right) \approx I_s \cdot e^{qV/KT} \quad (1)$$

式中，I ——二极管的通过电流

I_s ——反向饱和电流

V_{bc} ——两端的电压

q ——一个电子的电荷量，等于 1.602×10^{-19} (库)

K——等于 1.38×10^{-23} (焦耳／K)

T——温度 (K)

由式 (1) 可知，I／I s = $e^{qV_{bc}/KT}$ ， 所以

V_{bc} = KT／q•l n I ／ I s = KT／q•l n J （ 2 ）

由式 （ 2 ） 可知 V_{bc} 与绝对温度有着正比关系。

本次设计使用的温度传感器 AD590 在日常生活中已经被广泛使用，他常常被使用在空调、冰箱、电热水壶等产品中，它具有稳定性高、抗干扰能力强、灵敏性好等特点，因此 AD590 可以跟好地帮助我们完成设计。

2.6 液晶显示器 LCD12864

这次的设计我们选用的是带中文字库的液晶显示器 LCD12864。它的数据写入是用了 8 位端口的并行输入，拥有 128X64 的显示分辨率，因此它可以清晰的显示当前我们系统的状况和我们需要的设定值。但我们每次写入数据之前必须先写入其指令码，而指令码可以帮助我们轻松的设定液晶当前的状态，例如液晶的光标位置、是否开启背景灯，显示数据的开始位置等等。因此我们必须严格按照时序图的标准，不断地往 LCD12864 中写入命令和数据。

表 1：并行接口

管脚号	管脚名称	电平	管脚功能描述
1	VSS	0V	电源地
2	VCC	3.0+5V	电源正
3	V0	-	对比度（亮度）调整
4	RS	H/L	RS=“H”,DB7——DB0 显示数据 RS=“L”,DB7——DB0 显示指令码
5	R/W(SID)	H/L	R/W=“H”,E=“H”,数据被送到 DB7——DB0 R/W=“L”,E=“H→L”,输入数据被写到 IR 或 DR

6	E(SCLK)	H/L	使能信号
7	DB0	H/L	输入端口
8	DB1	H/L	输入端口
9	DB2	H/L	输入端口
10	DB3	H/L	输入端口
11	DB4	H/L	输入端口
12	DB5	H/L	输入端口
13	DB6	H/L	输入端口
14	DB7	H/L	输入端口
15	PSB	H/L	H：8 位并口输入方式，L：串口输入方式
16	NC	-	空脚
17	/RESET	H/L	复位端，低电平触发
18	VOUT	-	液晶的驱动电压
19	A	VDD	背光源正极
20	K	VSS	背光源负极

表 2：指令说明

指	指 令 码										功 能
令	RS	R/W	D7	D6	D5	D4	D3	D2	D1	D0	
清除显示	0	0	0	0	0	0	0	0	0	1	DDRAM 设为"20H",并且把 DDRAM 的地址计数器AC 设为"00H"
地址归位	0	0	0	0	0	0	0	0	1	X	将 DDRAM 的地址计数器AC 设为"00H",并且将游标指向开头;这个指令对 DDRAM 的内容没有影响
显示状态开/关	0	0	0	0	0	0	1	D	C	B	D=1: 整体显示 ON C=1: 游标 ON B=1:游标的位置反白允许
进入点设定	0	0	0	0	0	0	0	1	I/D	S	数据的读写，会使光标跟着数据位数的变化而移动

游标或显示移位控制	0	0	0	0	0	1	S/C	R/L	X	X	设定光标的移动位置;DDRAM 的内容不会收到指令的影响
功能设定	0	0	0	0	1	DL	X	RE	X	X	DL=0/1：4/8 位数据 RE=1：扩充指令 RE=0：基本指令
设定 CGRAM	0	0	0	1	AC5	AC4	AC3	AC2	AC1	AC0	设定 CGRAM 地址
设定 DDRAM	0	0	1	0	AC5	AC4	AC3	AC2	AC1	AC0	设定 DDRAM 地址（显示位址） 第一行：80H－87H 第二行：90H－97H
读取忙标志和地址	0	1	BF	AC6	AC5	AC4	AC3	AC2	AC1	AC0	读取 BF 可以知道内部数据处理是否完成
写数据到 RAM	1	0	数据								D7——D0 的数据输入到内部的 RAM
读出 RAM 的值	1	1	数据								从 RAM 读出 D7——D0 端口的数据

写操作时序

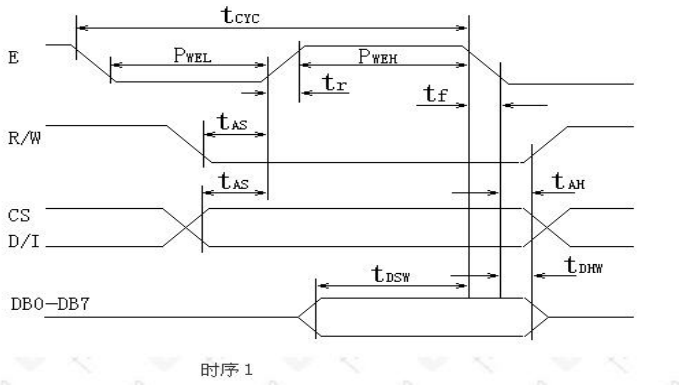


图 2-3 液晶写操作时序图

读操作时序

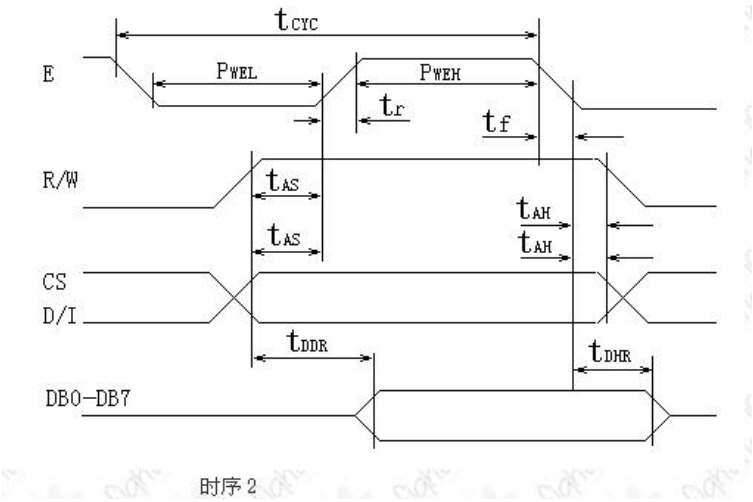


图 2-4 液晶读操作时序图

第三章.硬件电路设计

3.1 温度传感器及放大电路

传感器 J1(AD590)输出电流是以绝对温度零度（-273℃）为基准,每增加 1℃,它会增加 1μA 输出电流,因此在室温 25℃时,其输出电流 Iout=（273+25）=298μA。测量 Vo 时,不可分出任何电流，所以在应用时我们还要通过运算放大器来作相应处理才能达到测量 Vi0 时，不分出任何的电流,电路如图 3-1 所示。

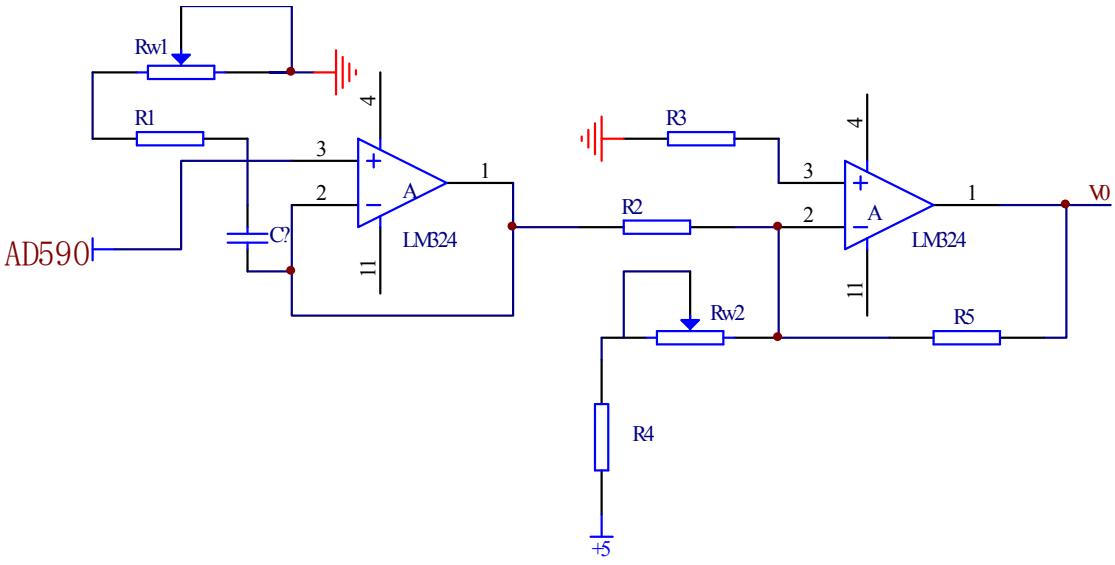


图 3-1 放大电路

其输出电流公式为 I=CT×(273. 15+ t)。式中，CT=1 uA / K，是标称温度系数；t 表示摄氏温度(℃)。AD590 的输出信号经第 1 级放大器 LM358 将电流信号变换为反相电压信号。信号经第 2 级放大器 LM358 放大为稳定的电压信号。

根据 AD590 特性，温度为绝对温度 xK 时，AD590 的输出电流为 xμA。令第 1 级放大器的输出电压为 V1 ，第 2 级放大器的输出电压为 V0，则：

V1=一(R1+Rw1)×x ×10-6；V0 / R5+Vtef / (R4 +Rw2)+V1 / R2=0

3.2 模/数转换电路

温度的电压信号经过电压跟随器的保持和差分放大电路的放大后，为了把放大后的电压信号输入到单片机中，我们还要使用模数转换器件 ADC0832 进行模数转换，这样才能把 0-5V 的电压转换为单片机可识别的数字量。因为模数转换模块的制造方式和工作方式不同，我们可供选择的这方面的模块会有许多。a/d 转换芯片的分辨率根据模块的不同分为以下几种：4 位的、6 位的、8 位的、10 位的、14 位、16 位、31/2 位的和 51/2 位的等等。如果根据行骗的转换速绿也可已分为以下几种：超高速、次超高速、高速、低速等等。A/D 转换也有两种转换方式，即为直接和间接转换。而直接转换，就是把输入的模拟量直接转换成数字量，其原理有逐次逼近法等方法。由于逐次逼近发比较易于转换且模块的工艺比较简便，因此大部分的芯片都选用的这种方式；间接转换就是添加一个中间量，先把输入的模拟量转换成中间的特殊量，再将中间的特殊量变为数字量，如电压/时间型、电压/频率型、电压/脉宽型等。

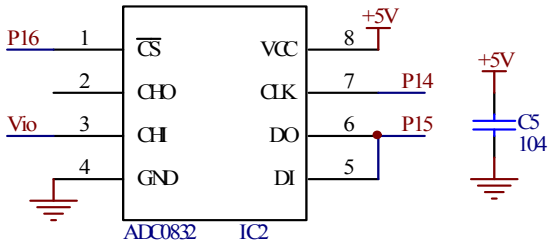


图 3-2 模/数芯片 ADC082 转换引脚图

引脚说明：

1 脚连接单片机 P16 口，用来控制芯片的片选信号，低电平有效。

3 脚连接模拟量输入 Vi0 口。

4 脚接地。

5、6 脚连接单片机 P15 口，作用为数字量输出端。

7 脚连接单片机 P14 口，作用接受串口时钟信号

8 脚接+5V 电源

3.3 数/模转换电路

单片机输出信号为数字量，如果我们要用来控制变频器等控制器，那么我们就要把这些数字量变为模拟电压信号，这样我们就要用到数/模转换芯片 TLC5615.其原理主要为逐次逼近法，该方法把原来的数字量进行量化，再逐步还原,.就像把模/数转换的步骤进行逆转一样。

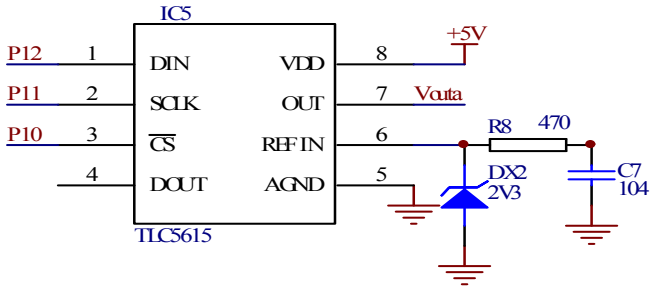


图 3-3 数/模转换芯片 TLC5615 芯片引脚图

引脚说明：

㈠1 脚连接单片机 P12 口，用来输入串口数字量。

㈡2 脚连接单片机 P11 口，用来输入时钟信号。

㈢3 脚连接单片机 P10 口，用来输入片选信号，低电频有效。

㈣4 脚用于级联串行输入，这里不使用。

㈤5 脚接地。

㈥6 脚基准电压输入，这里连接+5V 电源，这里 R8 用作限流和防止电流波动太大，DX2 稳压二极管稳压，C7 电容用来为芯片抗干扰。

㈦7 脚连接模拟电压输出。

㈧8 脚连接+5V 电源。

3.4 时钟电路的设计

时钟是单片机的心脏，单片机各功能部件的运行都是以时钟频率为基准。因此，时钟频率直接影响单片机的速度，时钟电路的质量也直接影响单片机系统稳定性。常用的时钟电路有两种方式，一种是内部时钟方式，另一种是外部时钟方式。本设计选用内部时钟方式如图 3-4：

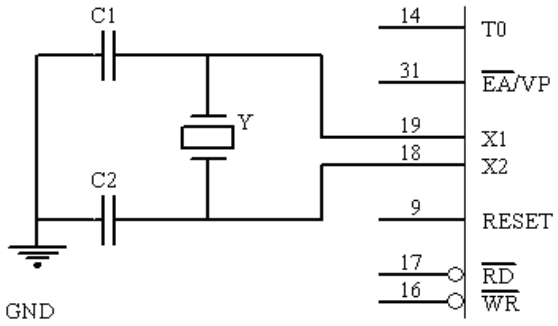


图 3-4 时钟电路部分原理图

电路中的电容 C1 和 C2 典型值通常选择 30pF 左右，该电容大小会影响振荡器频率的高低，振荡器的稳定性和起振的快速性。晶振的振荡器频率的范围通常在 1.2～12MHz 之间，晶体的频率越高，则系统得时钟频率也就变高，单片机的运行速度也就越快。但反过来运行速度快，对存储器的速度要求就高。对印刷电路板的工艺要求也高，即要求浅间的寄生电容要小；晶体和电容应尽可能安装得与单片机芯片靠近，以减少寄生生活，更好的保证振荡器稳定，可靠地工作。

3.5 复位电路

复位是单片机的初始化操作，只需给复位引脚 RST 加上大于 2 个机器周期（即 24 个时钟振荡周期）的高电平就复位，复位时，PC 初始化为 0000H，使 AT89C51 从 OUT 单元开始执行程序。本设计采用的电路如图 3-5 所示。

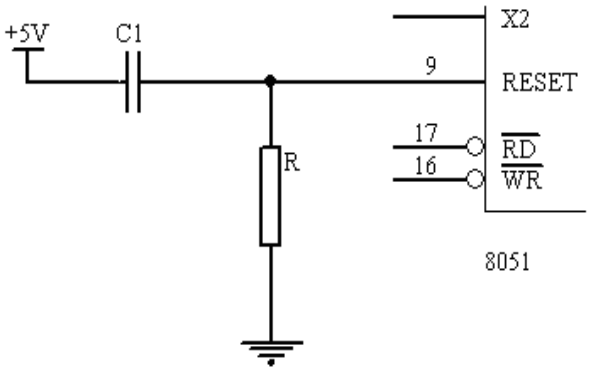


图 3-5 复位电路部分原理图

上电瞬间，RC 电路充电，RESET 引脚端出现正脉冲，只要 RESET 保持 10ms 以上高电平，就能使单片机有效的复位。当时钟频率选用 6MHz 时，C 取 22μF，R 取 1KΩ。上电自动复位电路由上电瞬间 C 与 R 构成充电电路，RESET 端的电位与 Vcc 相同，随着充电电流的减少，RESET 的电位逐渐下降。

3.6.按键电路设计

本次设计我们会用到许多按键，但由于按键的机械特点我们不能直接使用，我们必须通过一些方法消除按键的机械抖动。常见的消抖方法采用软件或硬件来实现：软件消抖主要是采用延时多次读取按键接口数据，通过比较前后两次读取按键端口的数据来判断是否有键按下；简单硬件消抖则简单硬件消抖则是采用电容来平掉信号的毛刺。但是对稳定性要求比较高的应用则需要采用相对复杂的集成电路来实现。

软件延时并不需要增加新的硬件，但采用这种方式来设计，一般通过软件指令或者定时器的方式来设定延时的时间，采用通用处理器，由于运行速度不一致，需要将软件做相应的修改。

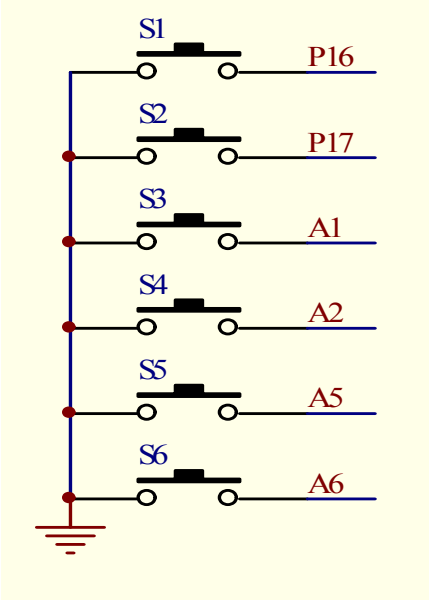


图 3-6 按键与单片机引脚连接图

本次按键设计中，我们用到了 6 个按键，分别对应本文中单片机的 P1.6、P1.7、A1、A2、A3、A4 等端口。按键 S1 是菜单键，用来设定温度；S2 是温度增加键，用来升温，每次加 1 度；S3 是温度减少键，用来降温，每次减 1 度；S4 是开启键，只有当 S4 打开后，那么才能进行温度调节；S5 是关闭键，S5 按下后，关闭温度调节系统；S6 为复位键，当 S6 按下时，所有值恢复初始值。

3.7. 显示电路

单片机把采集回来的信息和按键设定后的数据通过并行 LCD12864 液晶进行显示。

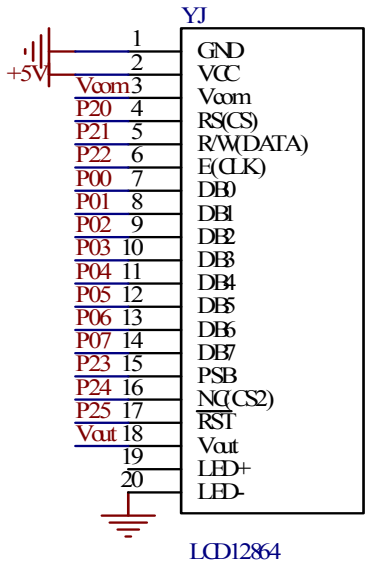


图 3-7 液晶与单片机引脚连接图

引脚说明：

1 脚接地。

2 脚接+5V 电源。

3 脚外接可调电阻，来调节液晶对比度。

4 脚接单片机 P20 口，高电频表示输入数据，低电频表示输入指令。

5 脚接单片机 P21 口，高电频且使能端 E 为高电频时表示读数据，低电频且使能端由高变低时表示写数据。

6 脚连接单片机 P22 口，使能端口。

7 脚~14 脚连接单片机 P0 口，并行数据输入。

15 脚连接单片机 P23 口，高电频并行输入，低电频串行输入。

17 脚连接单片机 P25 口，复位低电频有效。

18 脚为液晶驱动电压输出端口。

3.8. 电源电路

电源电路的输入端使用的是带中心抽头的变压器将 220V 电压降为 15V 再经过全波整流后输入的电压，输入的 15V 电压经过 7812 三端稳压器把电压稳定在 12V，让后再通过带散热片的三端稳压器 7805 把电压稳定在 5V。

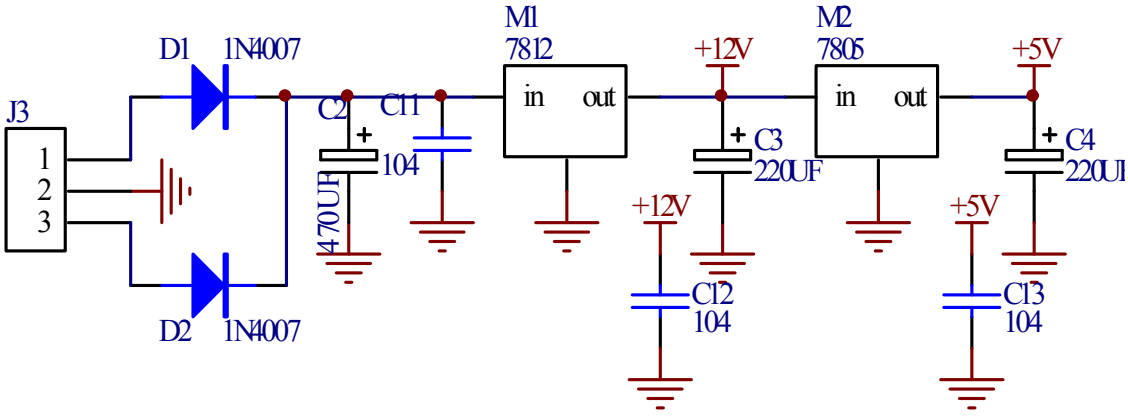


图 3-8 电源电路图

3.7. 输出控制原理

图 3-9 变频器 MM420 实物图

本次设计的输出端连接的是一款西门子公司生产的 MICROMASTER420 变频器。这款变频器适用于多种变速驱动应用，其主要特性有以下几方面：

⑴引导调试简单

⑵模块化结构允许组态的最大灵活性

⑶三个全可编程绝缘数字量输入

⑷可量测的模拟量输入(0 V 到 10 V， 0 mA 到 20 mA) 也可以被用作第 4 个数字量输入

⑸一个可编程模拟量输出（0 mA 到 20 mA）

⑹1 个可编程继电器输出

⑺30 V DC/5 A, 阻性负载

⑻250 V AC/2 A, 感性负载

(9)因高脉冲频率而获得低噪音电机运转,可调节（如果必要，可降额运行）。

这款变频器的外部接线一共有8根线。其中3根线连接滑动变阻器，是通过滑动变阻器的电压变化来控制频率的。从上文可知输出信号经过数/模转换后又通过放大电路放大得到一个0~10V 变化的电压信号；因为我们得到的已经是一个变化的电压信号，所以这里不需要通过滑动变阻器的调节来改变电压，我们可以直接连接一个固定电阻即可。

其中的2根线是控制电动机正反转的，一般是连接2个外接开关回路即可，这里我们用的是2个光电耦电路进行控制的，光耦的左端是由单片机进行控制的，右端连接变频器。而另外的3根线是连接的压缩机，这里就不做详细介绍了。

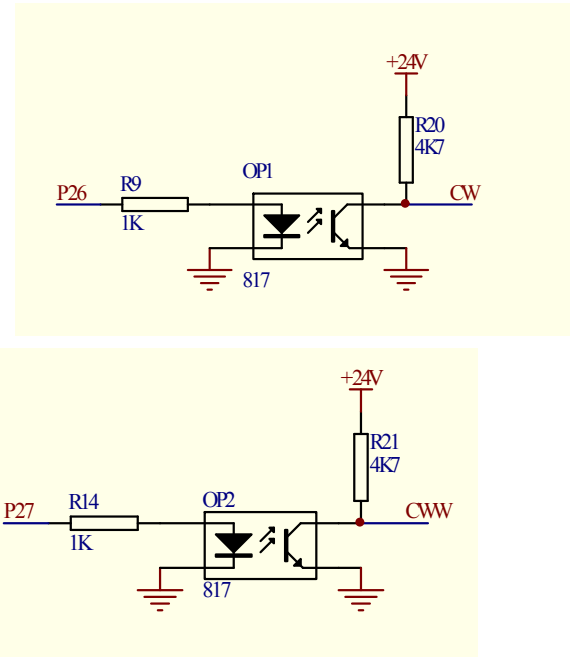


图3-10控制变频机正反转光耦连接图

第四章.软件设计

4.1 主程序设计

硬件制作完毕之后，接下来就是程序设计，本设计采用 C 语言,在主程序中，主要实现初始化，模数转换，按键处理，温度采用动态显示方式。当有键按下时,进入按键处理程序,程序流程图如图 4-1 所示:

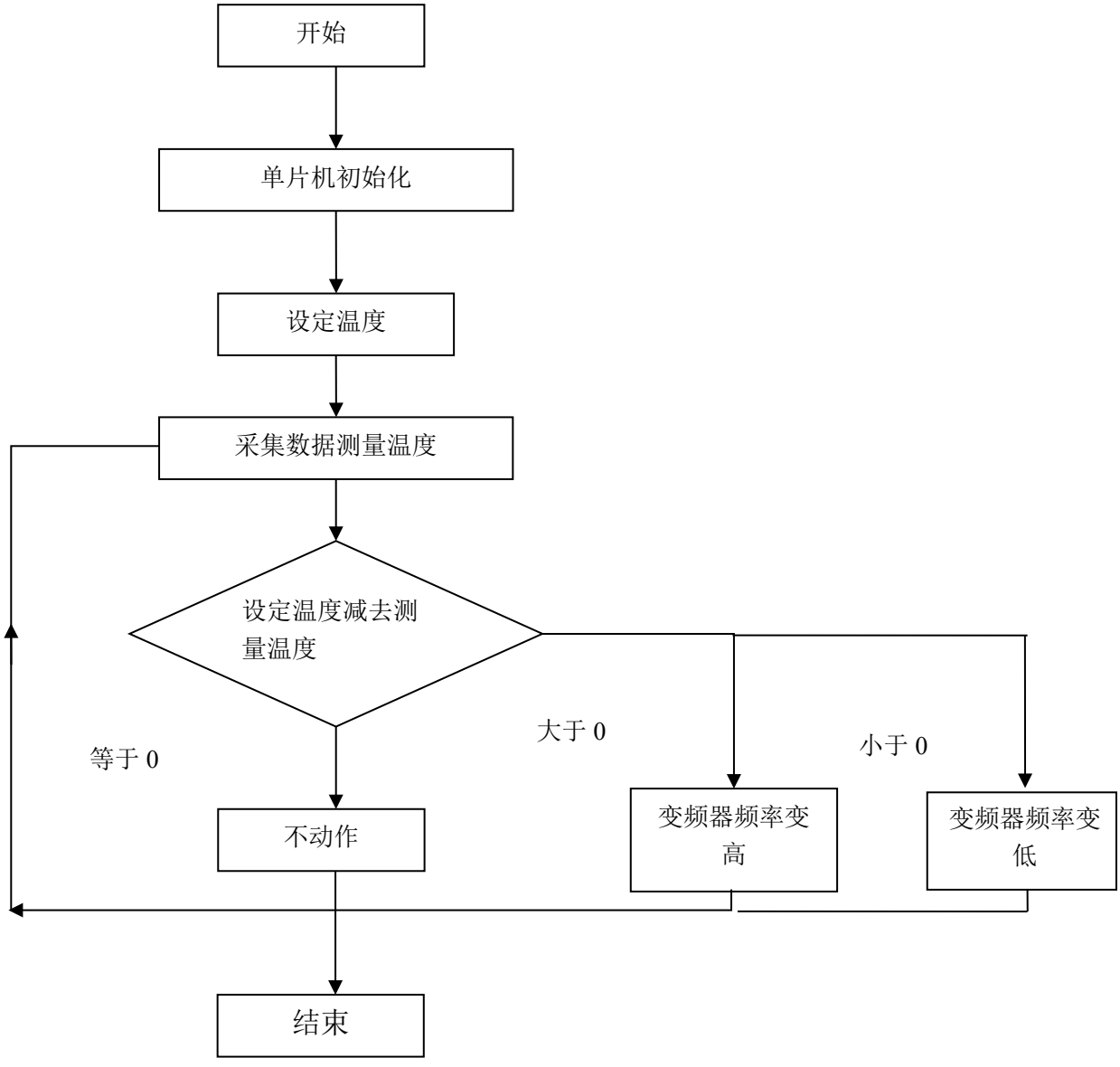


图 4-1 主程序流程图

4.2 监控系统设计

图 4-2 为系统的按键流程图。主要是通过人为的对外部按键的控制来调节

系统的温度，从而实现系统对温度的手动和自动控制。

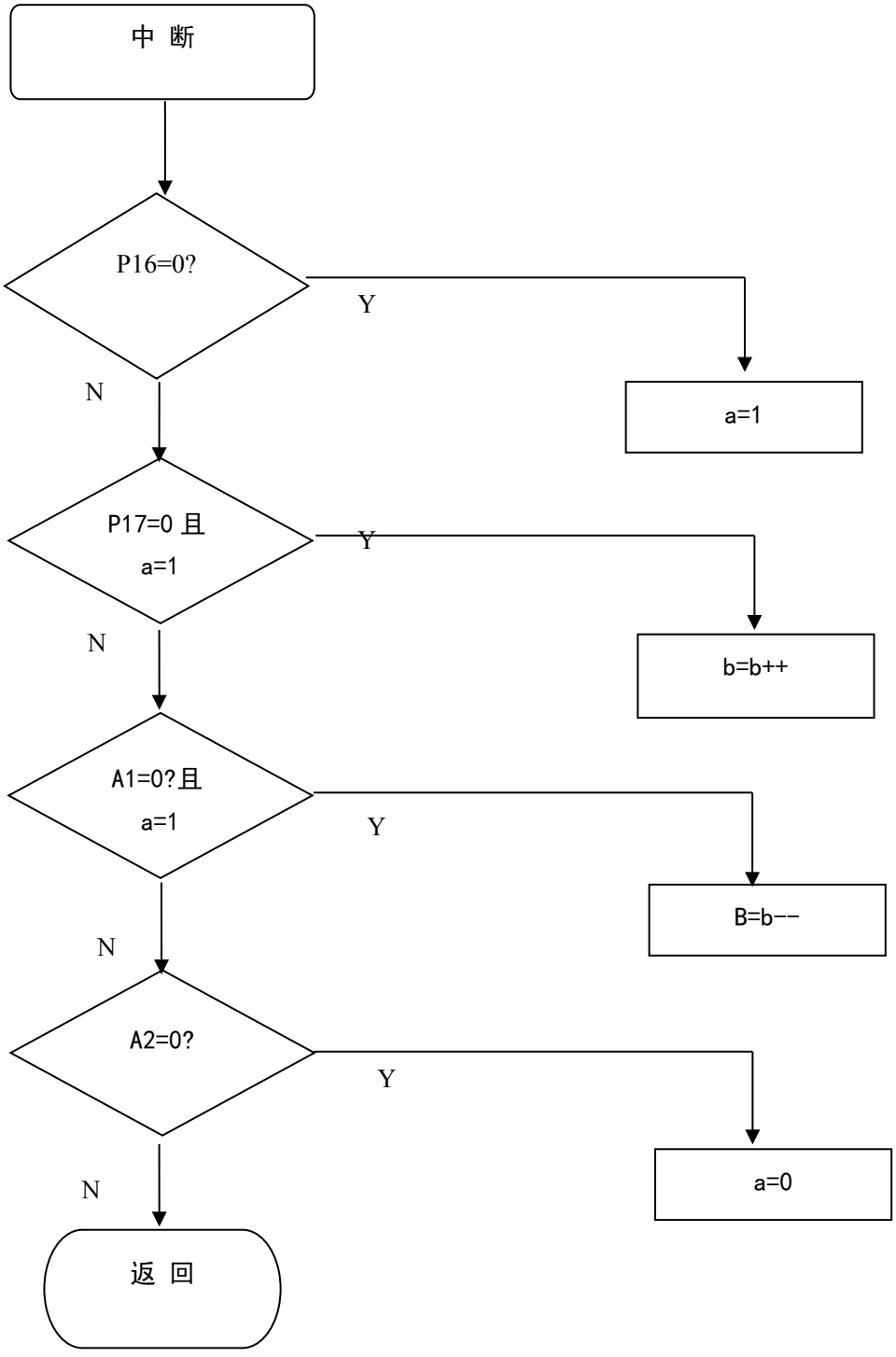


图4-2 按键程序流程图

为了更好地了解按键系统这里提供了一段按键去抖动的程序，里面的delay 子程序就是去抖动的延时程序。

```
#include<reg51.h>
sbit SI=P1^6;
```

```
sbit LED0=P3^0;
void delay()
{
   unsigned char i,j;
   for(i=0;i<100;i++)
      for(j=0;j<100;j++)         ;
}
void main()
{
   LED0=0;
   while(1)
   {
      if(SI==0)
      {
         delay();      //延时一段时间
         if(SI==0)     //再次确认是否有按键按下
            LED0=1;    //当有按键按下，点亮 LED
       }
      else
      {
            LED0=0;
     }
   }
}
```

4.3PID 控制系统设计

在工业上，偏差控制又称为 PID（Proportional Integral and Differential,比例积分与微分）。这是工业控制过程中应用广泛的一种控制形式。

PID 控制的理想微分方程为：

$$U(t)=k_p\left[e(t)+\frac{1}{T_i}\int_0^t e(t)dt+T_d\frac{de(t)}{dt}\right] \quad (4-1)$$

其中 e(t)=r(t)-y(t)称为偏差值，可作为温度调节器的输入信号，其中 r(t)为给定值，y(t)为被测变量的值；k_p为比例系数；T_i为积分时间常数；T_d为微分时间常数；u(t)为调节器的输出控制器输出控制信号。

但计算机只能处理数字信号，故上述数学方程必须加以变换。若设温度的采样周期为 T，第 n 次的采样得到的输入偏差为e_n，调节器输出为 u_n，则有：

$$\frac{de(t)}{dt}=\frac{e_n-e_{n-1}}{T} \quad （微分用差分代换）$$

$$\int_0^t e(t)dt=\sum_{k=0}^{n}e_k\times T \quad （积分用求和代替）$$

这样式 4-1 便可写为：

$$u_n=k_p\left[e_n+\frac{1}{T_i}\sum_{k=0}^{n}e_k\times T+T_d\frac{e_n-e_{n-1}}{T}\right] \quad (4-2)$$

写成递推式为:

$$u_n=k_p\left[e_n-e_{n-1}+\frac{T}{T_i}e_n+\frac{T_d}{T}(e_n-2e_{n-1}+e_{n-2})\right]+\left(e_{n-1}+\frac{1}{T_i}\sum_{k=0}^{n-1}e_k\times T+T_d\frac{e_{n-1}-e_{n-2}}{T}\right)k_p$$
$$=u_{n-1}+k_p\left[e_n-e_{n-1}+\frac{T}{T_i}e_n+\frac{T_d}{T}(e_n-2e_{n-1}+e_{n-2})\right]$$

把上式改写为：

$$U(n)=u(n-1)+k_p\{E(n)-E(n-1)+k_I E(n)+k_D[E(n)-2E(n-1)+E(N-2)]\}$$
$$=U(n-1)+p_P+p_I+p_D$$

(4-3)

式4-3可以改写成：$p_k=p(k-1)+p_p+p_I+p_D$

使用增量式 PID 算法，所谓增量式 PID 算法就是通过计算相邻两次控制量运算之差，得到的差值就是增量，如果为正则表示要在上一次控制量的基础上增加控制量，反之则在上一次控制量的基础上减少控制量。对于温度控制来说

就是增加或者减少加热比例，可以比较精确的控制。

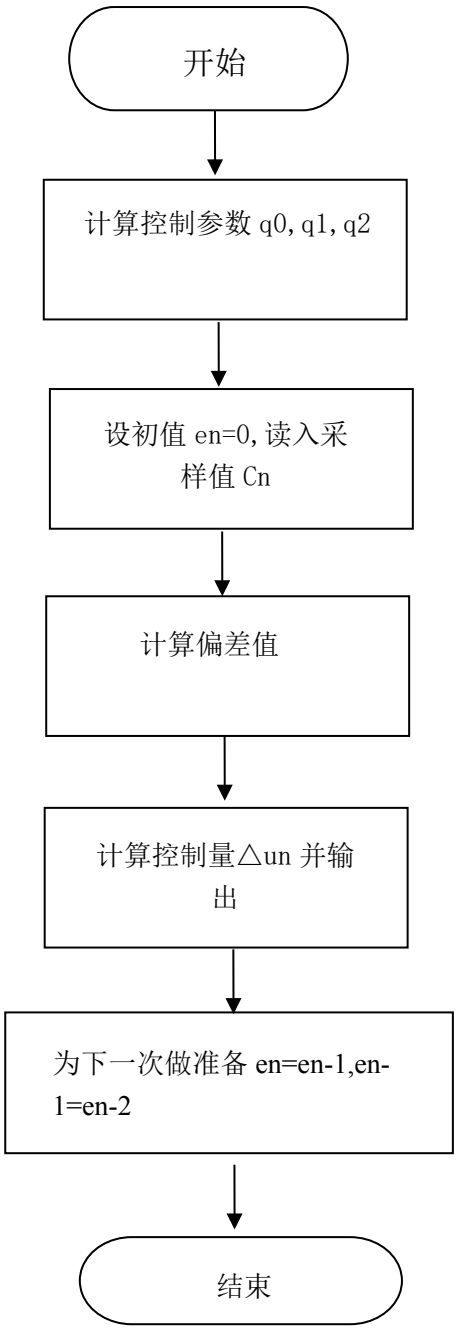


图 4-3 PID 控制流程图

致谢

本论文是在我的导师×××的亲切关怀和悉心指导下完成的。他严肃的科学态度，严谨的治学精神，精益求精的工作作风，深深地感染和激励着我。从课题的选择到项目的最终完成，×老师都始终给予我细心的指导和不懈的支持。在这几个月里，×老师不仅在学业上给我以精心指导，同时还在思想、生活上给我以无微不至的关怀，在此谨向×老师致以诚挚的谢意和崇高的敬意。

参考文献

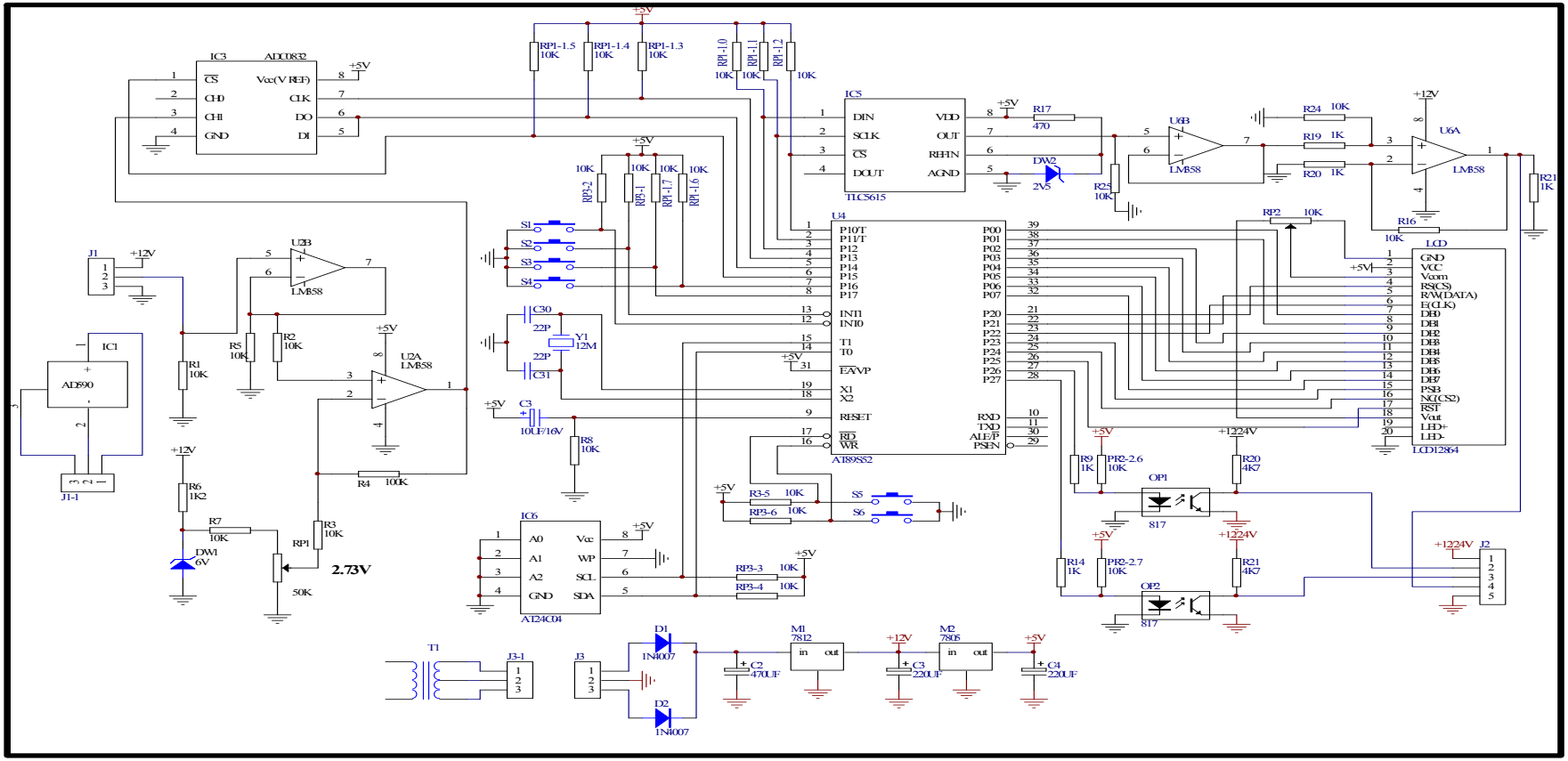
[1]张友德等，单片机原理应用与实验[M]，复旦大学出版社1992.
[2]张毅刚，彭喜源，谭晓钧，曲春波.MCS－51单片机应用设计[M].哈尔滨工业大学出版社2001.1.
[3]宋庆环，才卫国，高志，89C51单片机在直流电动机调速系统中的应用[M]。唐山学院，2008.4
[4]陈　锟　危立辉，基于单片机的直流电机调速器控制电路[J]，中南民族大学学报(自然科学版)，2003.9.
[5]李维军 韩小刚 李 晋，基于单片机用软件实现直流电机 PWM 调速系统[J]，维普资讯，2007.9
[6]曹巧媛.单片机原理及应用[M].北京，电子工业出版社，1997.
[7]刘大茂，严飞.单片机应用系统监控主程序的设计方法[J].福州大学学报(自然科学福建农林大学硕士论文版)，1998.2.
[8] 刘昌华，易逵编著.8051单片机的 C 语言应用程序设计与实践[M].国防工业出版社2007.9
[9]朱定华，戴汝平编著.单片机原理与应用[M].清华大学出版社北方交通大学出版社，2003.8.
[10] 金伟正，单线数字温度传感器的原理与应用 [J]，电子技术与应用，2000
[11]薛钧义 张彦斌编著. MCS—51/96系列单片微型计算[M].西安交通大学出版社，1997.8
[12]陈国呈 编著.PWM 逆变技术及应用[M].中国电力出版社.2007年7月
[13]马忠梅　等编著.单片机的 C 语言应用程序设计（第4版）[M]，北京航天航空大学出版社.2007. 4
[14]　李朝青，《单片机原理及接口技术》（简明修订版）[M]，北京航空航天大学出版社，1998
[15] 李广弟，《单片机基础》[M]，北京航空航天大学出版社，1994
[16] 阎　石，《数字电子技术基础》（第三版）[M]，高等教育出版社，1989
[17] 廖常初，现场总线概述[J]，电工技术，1999
[18]　于永学、葛建，1-WIRE 总线数字温度传感器 DS18B20及应用[J]，电子产品世界，2003
[19] 陈跃东，AD590集成温度传感器原理与应用[J]，安徽机电学院学报，2002
[20]胡振宇、刘鲁源、杜振辉，DS18B20接口的 C 语言程序设计 [J]，单片机与嵌入式系统应用，2002

结论

通过这次设计,我发现在现实设计中还需要注意很多的细节,包括程序设计和硬件设计都要我们小心仔细,一个地方出错就可能会整个系统失效。在硬件设计时，由于电路图转印不好使得腐蚀后的电路板出现断线，在调试过程中引来很大的麻烦，在写调整设定温度程序时，当按选择键择选要调整时还是十位或个位时，看不出是要调整那一位为了方便用户看出现在是在调整那一位，就必须使调整位闪烁，但显示的时候已经用动态显示，如果直接改显示时间，就会使所有的位都闪烁，这样根本就不能实现，后面经过和同学导论，我使不调的位显示 4 到 5 次再显示调整位 1 次，这样做最后就可以实现了，而且这样做还可以改变闪烁速度。

本次设计电路原理图时还有一个错误，这一错误导致电路不能正常工作，在看 AD590 的 PDF 文档后，发现 AD590 供电电压为 4-13V，所以我不假思索就把电源供电设为 5V。电路板制作完成后调试发现测量温度不准确，测到 30 多摄氏度之后就上不去了。后来才发现 AD590 是与 10K 的电阻串连，当温度上升到 30 摄氏度时电阻两端的电压为 3V 而供电电压为 5V。由此可知 AD590 两端的电压为 2V 这一电压远远低于正常工作电压，找出问题的原因后我立即把供电电压改为 12V 然后重新制作一块电路板，最后终于调试成功。

附录 1


IC3 ADC0832
CS
CH0
CH1
GND
Vcc(V REF)
CLK
DO
DI
+5V
RP1-1.5 10K
RP1-1.4 10K
RP1-1.3 10K
RP1-1.0 10K
RP1-1.1 10K
RP1-1.2 10K
IC5
DIN
SCLK
CS
DOUT
VDD
OUT
REFIN
AGND
TLC5615
R17 470
DW2 2V5
R25 10K
U6B LM358
U6A LM358
R24 10K
R19 1K
R20 1K
R21 1K
R16 10K
+12V
RP2-2 10K
RP2-1 10K
RP1-1.7 10K
RP1-1.6 10K
U4
P10/T
P11/T
P12
P13
P14
P15
P16
P17
INT1
INT0
T1
T0
EA/VP
X1
X2
RESET
RD
WR
P00
P01
P02
P03
P04
P05
P06
P07
P20
P21
P22
P23
P24
P25
P26
P27
RXD
TXD
ALE/P
PSEN
AT89S52
RP2 10K
LCD
GND
VCC
Vcom
RS(CS)
R/W(DATA)
E(CLK)
DB0
DB1
DB2
DB3
DB4
DB5
DB6
DB7
PSB
NC(/CS2)
RST
Vout
LED+
LED-
LCD12864
S1
S2
S3
S4
C30 22P
Y1 12M
C31 22P
C3 10UF/16V
R8 10K
S5
S6
R3-5 10K
RP3-6 10K
RP3-3 10K
RP3-4 10K
IC6
A0
A1
A2
GND
Vcc
WP
SCL
SDA
AT24C04
J1
J1-1
IC1
AD590
U2B LM358
U2A LM358
R5 10K
R2 10K
R1 10K
R4 100K
R6 1K2
R7 10K
DW1 6V
RP1
R3 10K
50K
2.73V
OP1 817
OP2 817
R9 1K
PR2-2.6 10K
R20 4K7
R14 1K
PR2-2.7 10K
R21 4K7
+1224V
J2
T1
J3-1
J3
D1 1N4007
D2 1N4007
M1 7812
M2 7805
C2 470UF
C3 220UF
C4 220UF

附录 2

```
#include<reg51.h>
#include<intrins.h>
#include<math.h>
#include<string.h>
struct PID {
unsigned int SetPoint; // 设定目标 Desired Value
unsigned int Proportion; // 比例常数 Proportional Const
unsigned int Integral; // 积分常数 Integral Const
unsigned int Derivative; // 微分常数 Derivative Const
unsigned int LastError; // Error[-1]

unsigned int PrevError; // Error[-2]

unsigned int SumError; // Sums of Errors
};
struct PID spid; // PID Control Structure
unsigned int rout; // PID Response (Output)
unsigned int rin; // PID Feedback (Input)
sbit data1=P1^0;
sbit clk=P1^1;
sbit plus=P2^0;
sbit subs=P2^1;
sbit stop=P2^2;
sbit output=P3^4;
sbit DQ=P3^3;
unsigned char flag,flag_1=0;
unsigned char high_time,low_time,count=0;//占空比调节参数
unsigned char set_temper=35;
unsigned char temper;
unsigned char i;
unsigned char j=0;
unsigned int s;
延时子程序,延时时间以 12M 晶振为准,延时时间为 30us×time
void delay(unsigned char time)
{
unsigned char m,n;
for(n=0;n<time;n++)
for(m=0;m<2;m++){}
}
写一位数据子程序
void write_bit(unsigned char bitval)

{
EA=0;
DQ=0; /*拉低 DQ 以开始一个写时序*/
if(bitval==1)
{
  _nop_();
  DQ=1; /*如要写 1，则将总线置高*/
}
delay(5); /*延时 90us 供 AD590 采样*/
DQ=1; /*释放 DQ 总线*/
_nop_();
_nop_();
EA=1;
}

写一字节数据子程序
void write_byte(unsigned char val)
{
unsigned char i;
unsigned char temp;
EA=0;
```

```
TR0=0;
for(i=0;i<8;i++)  /*写一字节数据，一次写一位*/
{
  temp=val>>i;  /*移位操作，将本次要写的位移到最低位*/
  temp=temp&1;
  write_bit(temp); /*向总线写该位*/
}
delay(7); /*延时 120us 后*/
EA=1;
}
```

读一位数据子程序

```
unsigned char read_bit()
{
unsigned char i,value_bit;
EA=0;
DQ=0; /*拉低 DQ，开始读时序*/
_nop_();
_nop_();
DQ=1; /*释放总线*/
for(i=0;i<2;i++){}
value_bit=DQ;
EA=1;
return(value_bit);
}
```

读一字节数据子程序

```
unsigned char read_byte()
{
unsigned char i,value=0;
EA=0;
for(i=0;i<8;i++)
{
  if(read_bit())  /*读一字节数据，一个时序中读一次，并作移位处理*/
  value|=0x01<<i;
  delay(4);  /*延时 80us 以完成此次都时序，之后再读下一数据*/
}
EA=1;
return(value);
}
```

复位子程序

```
unsigned char reset()
{
unsigned char presence;
EA=0;
DQ=0; /*拉低 DQ 总线开始复位*/
delay(30); /*保持低电平 480us*/
DQ=1; /*释放总线*/
delay(3);
presence=DQ; /*获取应答信号*/
delay(28); /*延时以完成整个时序*/
EA=1;
return(presence);   /*返回应答信号，有芯片应答返回 0,无芯片则返回 1*/
}
```

获取温度子程序

```
void get_temper()
{
unsigned char i,j;
do
{
  i=reset(); /*复位*/
}while(i!=0); /*1 为无反馈信号*/
i=0xcc; /*发送设备定位命令*/
write_byte(i);
i=0x44; /*发送开始转换命令*/
write_byte(i);
delay(180); /*延时*/
do
{
  i=reset(); /*复位*/
}while(i!=0);
i=0xcc; /*设备定位*/
write_byte(i);
i=0xbe; /*读出缓冲区内容*/
write_byte(i);
j=read_byte();
i=read_byte();
i=(i<<4)&0x7f;
s=(unsigned int)(j&0x0f);
s=(s*100)/16;
j=j>>4;
```

```
temper=i|j;  /*获取的温度放在 temper 中*/
}
void PIDInit (struct PID *pp)
{
memset ( pp,0,sizeof(struct PID));
}
```
PID 计算部分
```
unsigned int PIDCalc( struct PID *pp, unsigned int NextPoint )
{
unsigned int dError,Error;
Error = pp->SetPoint - NextPoint; // 偏差
pp->SumError += Error; // 积分
dError = pp->LastError - pp->PrevError; // 当前微分
pp->PrevError = pp->LastError;
pp->LastError = Error;
return (pp->Proportion * Error // 比例项
+ pp->Integral * pp->SumError // 积分项
+ pp->Derivative * dError); // 微分项
}
```
温度比较处理子程序
```
compare_temper()
{
unsigned char i;
if(set_temper>temper)
{
  if(set_temper-temper>1)
  {
   high_time=100;
   low_time=0;
  }
  else
  {
   for(i=0;i<10;i++)
   {
    get_temper();
    rin = s; // Read Input
    rout = PIDCalc ( &spid,rin );
// Perform PID Interation
   }
   if (high_time<=100)
   {
    high_time=(unsigned char)(rout/800);
   }
   else
   {
    high_time=100;
      }
   low_time= (100-high_time);
  }
}
else if(set_temper<=temper)
{
  if(temper-set_temper>0)
  {
   high_time=0;
   low_time=100;
  }
  else
  {
   for(i=0;i<10;i++)
   {
    get_temper();
    rin = s; // Read Input
    rout = PIDCalc ( &spid,rin );
// Perform PID Interation
   }
   if (high_time<100)
   {
    high_time=(unsigned char)(rout/10000);
   }
   else
   {
    high_time=0;
   }
   low_time= (100-high_time);
```

```
  }
}
}
```

T0 中断服务子程序，用于控制电平的翻转 ,40us*100=4ms 周期

```
void serve_T0() interrupt 1 using 1
{
if(++count<=(high_time))
output=1;
else if(count<=100)
{
  output=0;
}
else
count=0;
TH0=0x2f;
TL0=0xe0;
}
```

```
void      disp_1(unsigned      char
disp_num1[6])
{
unsigned char n,a,m;
for(n=0;n<6;n++)
{
// k=disp_num1[n];
  for(a=0;a＜8;a++)
  {
   clk=0;
   m=(disp_num1[n]&1);
   disp_num1[n]=disp_num1[n]>>1;
   if(m==1)
   data1=1;
   else
   data1=0;
   _nop_();
   clk=1;
   _nop_();
  }
}
}
```

显示子程序

功能：将占空比温度转化为单个字符，显示占空比和测得到的温度

```
void display()
{
unsigned          char          code
number[]={0xfc,0x60,0xda,0xf2,0x66,
0xb6,0xbe,0xe0,0xfe,0xf6};
unsigned char disp_num[6];
unsigned int k,k1;
k=high_time;
k=k%1000;
k1=k/100;
if(k1==0)
disp_num[0]=0;
else
disp_num[0]=0x60;
k=k%100;
disp_num[1]=number[k/10];
disp_num[2]=number[k%10];
k=temper;
k=k%100;
disp_num[3]=number[k/10];
disp_num[4]=number[k%10]+1;
disp_num[5]=number[s/10];
disp_1(disp_num);
}
```

主程序

```
main()
{
unsigned char z;
unsigned char a,b,flag_2=1,count1=0;

unsigned                              char
phi1[]={2,0xce,0x6e,0x60,0x1c,2};;
TMOD=0x21;
TH0=0x2f;
TL0=0x40;
SCON=0x50;
```

```
PCON=0x00;
TH1=0xfd;
TL1=0xfd;
PS=1;
EA=1;
EX1=0;
ET0=1;
ES=1;
TR0=1;
TR1=1;
high_time=50;
low_time=50;
PIDInit ( &spid ); // Initialize Structure
spid.Proportion = 10; // Set PID Coefficients
spid.Integral = 8;
spid.Derivative =6;
spid.SetPoint = 100; // Set PID Setpoint
while(1)
{
  if(plus==0)
  {
   EA=0;
   for(a=0;a<5;a++)
    for(b=0;b<102;b++){}
   if(plus==0)
   {
    set_temper++;
    flag=0;
   }
  }
  else if(subs==0)
  {
   for(a=0;a<5;a++)
    for(b=0;a<102;b++){}
   if(subs==0)
   {
    set_temper--;
    flag=0;
   }
  }
  else if(stop==0)
  {
   for(a=0;a<5;a++)
    for(b=0;b<102;b++){}
   if(stop==0)
   {
    flag=0;
    break;
   }
   EA=1;
  }
  get_temper();
  b=temper;
  if(flag_2==1)
  {
   a=b;
  }
  if((abs(a-b))>5)
  {
   temper=a;
  }
  else
  {
   temper=b;
  }
  a=temper;
  flag_2=0;
  if(++count1>30)
  {
   display();
   count1=0;
  }
  compare_temper();
}
TR0=0;
z=1;
while(1)
{
  EA=0;
  if(stop==0)
  {
```

```
  for(a=0;a<5;a++)
   for(b=0;b<102;b++){}
  if(stop==0)
  disp_1(phi1);

}
```

```
   EA=1;
}
}
```