一种用于温度控制的微型模块

软件设计说明书

# 1一种用于温度控制的微型模块软件设计背景

随着Internet的快速发展，光纤通信整个领域呈现良好的增长势头。国内外公司都相继研制成功了各种新型激光光源、光调制器以及光电探测器。作为光电子系统的核心器件，半导体激光器具有极大的研究、开发意义和市场潜力。对于半导体激光器的温度控制系统的研究与开发也就有了很大的应用。

由于半导体激光器的精度很高，要求对不同环境下同样要能快速、精确、实时的测量和记录，对于半导体器件影响较大的温度这一可控因素就有了很大要求。温度控制微型模块要求模块整体要小，能够实时测量半导体激光器的温度并且进行显示，而且要在一定的温度范围内进行温度控制。

温度控制的微型模块的设计方案，一般选择性能高，处理速度快，可编程，价格相对低的STM32系列单片机作为主控芯片；选择STM32F103C8T6作为主控芯片，配合测温精度高的DS18B20模块，温控效率高的TEC制冷片等模块，就可以实现本温控的设计。

STM32F103C8T6是一款性价比比较高，使用起来功能消耗也是比较低的芯片。此芯片属于中等容量芯片，闪存容量可达64K；它的功耗极低，可达36mA；它还具有丰富的IO接口，可满足外接多个硬件设备，满足本设计的需求。

温控模块半导体制冷片有多种型号供选择，它可根据不同的接线方式从而达到制冷制热的切换，其正常工作温度范围也符合要求，对于所需要的应用场景完全符合要求。

用于温度控制的微型模块软件是通过测温模块与被测物体进行连接，测温模块实时返回测量数据给MCU，通过对比提前设置好的目标温度启动温控模块进行温控，当达到目标温度之后自动停止温度控制，这样就完成了温度控制。

对于温度控制，当启动制冷的时候，需要做好散热处理，否则可能产生制冷效果不理想，甚至烧毁制冷片。装置的实物示意图如图1所示。

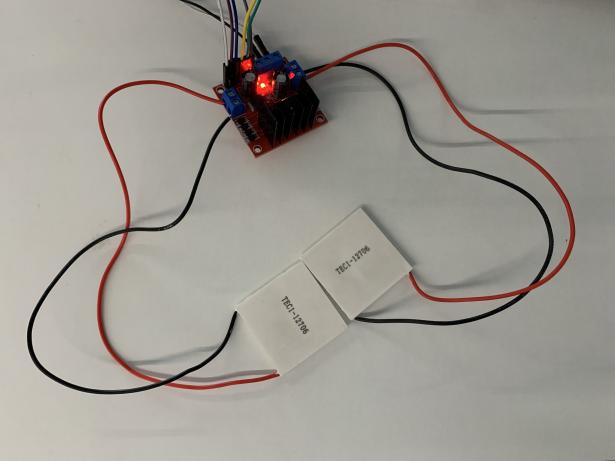


图1 一种用于温度控制的微型模块测试实物图

# 2用于温度控制的微型模块简介

## 2.1硬件整体方案设计

用于温度控制的微型模块主要由STM32F103C8T6（中心控制模块）、DS18B20(温度检测模块)、L298N(H桥电路模块)、TEC（温控模块）和OLED显示屏组成。STM32F103C8T6用于对各个模块的控制，对于输入模块检测到的数据经过计算处理，传输给各个输出模块进行驱动。整体系统结构图如图2所示。

图示

描述已自动生成

图2 一种用于温度控制的微型模块整体系统结构图

## 2.2硬件电路原理图设计

主控芯片使用STM32F103系列芯片，该系统板有40个管脚，各个管脚都有自己的名称定义和功能，主要可以分为五类，其中通过对不同管脚的配置可以实现不同的作用。大部分管脚都是通用的。值得注意的是此芯片大部分GPIO都具有复用功能，所以在配置的时候要特别注意。带模拟输入的接口不允许过大电流流过，其它的都可以。接上电源后，管脚不会复位，其状态信息会被保留。

主芯片采用了STM32F103系列，可以在不同的端口上设置 GPIO管脚，只需进行软件配置一下即可，比如输出、输入等端口。大多数引脚都是共用的。具有模拟输入功能的端口不可以通过过大的电流，其他的都可以。如有必要，还可以锁定 I/O管脚的外置功能，防止写入I/O寄存器时出现意外。上电过后的引脚不会复位，它的状态信息会保存下来。

这个芯片的一般工作状态如表1所示，供电电源一般要是一样的，在上电和正常运行期间，VDD和VDDA的最大区别是300 mV。当TJ不超过TJmax时，允许使用较低的TA，以提高PD值；当处于低功耗散度状态时，只要TJ不超过TJmax，TA就能扩展到这个范围；晶片所消耗的电流是多种参数和因素的综合指标，其中包括工作电压、环境温度、I/O管脚的负荷、产品的软件配置、工作频率等等。在输入模式下，所有I/O引脚都连接在一个静态水平上，外部设备都是在关闭的状态之下的。Flash存储器的存取时间调整到fHCLK的频率(0~24MHz为0等待周期，24~48MHz为1等待周期，超过48 MHz为2等待周期)。指示预取函数打开；当外部设备打开时：fPCLK1= fHCLK/2，fPCLK2=fHCLK；满足这些条件时，芯片的消耗电流最大。

表1 工作状态

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 符号 | 参数 | 条件 | 最小值 | 最大值 | 单位 |
| fHCLK | 内部AHB时钟频率 |  | 0 | 72 | MHz |
| fPCLK1 | 内部APB1时钟频率 |  | 0 | 36 |
| fPCLK2 | 内部APB2时钟频率 |  | 0 | 72 |
| VDD | 标准工作电压 |  | 2 | 3.6 | V |
| VDDA | 模拟部分工作电压（未使用ADC） | 必须与VDD相同 | 2 | 3.6 | V |
| 模拟部分工作电压（使用ADC） | 2.4 | 3.6 |
| VBAT | 备份部分工作电压 |  | 1.8 | 3.6 | V |
| PD | 功率耗散  温度标号6：TA=85℃  温度标号7：TA=105℃ | LFBGA100 |  | 454 | mW |
| LQFP100 | 434 |
| TFBGA64 | 308 |
| LQFP64 | 444 |
| LQFP48 | 363 |
| VFQFPN36 | 1110 |
| TA | 环境温度（温度标号6） | 最大功率耗散 | -40 | 85 | ℃ |
| 低功率耗散 | -40 | 105 |
| 环境温度（温度标号7） | 最大功率耗散 | -40 | 105 | ℃ |
| 低功率耗散 | -40 | 125 |
| TJ | 结温度范围 | 温度标号6 | -40 | 105 | ℃ |
| 温度标号7 | -40 | 125 |

主控芯片的电路原理图如图3所示，其标明了各个IO口的连线方式。

图片包含 图表

描述已自动生成

图3 主控芯片电路原理图

OLED在各种高端电子设备中得到了广泛的应用，而低端的 LCD则主要用于广告牌、计算器、电子手表等。

该方案使用了0.96寸的4管脚OLED屏幕，能够显示数字、符号、图像，液晶屏幕的内部为点阵 LED。VCC的正常工作电压为3~5.5 V, 在进行IIC通讯时，SCL是时钟管脚，SDA是数据管脚。

OLED为I2C通讯，采用8080并口通信方式。

下图4和图5分别为8080并口通信的写时序和读时序。

因为要写数据,由WR控制,所以RD设置为高电平

DC: 1,写数据 0 写命令

CS:要操作OLED前,先进行片选,拉低CS引脚,操作完成后拉高

WR:写数据,在WR的上升沿,使数据写入到SSD1306控制器内

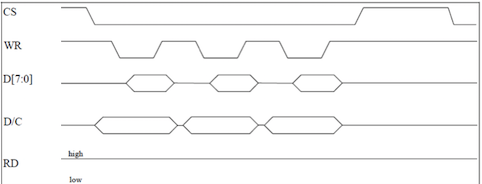


图4 并口写时序

因为要读数据,由RD控制,所以WR设置为高电平

DC: 1,读数据 0 读命令

CS:要操作OLED前,先进行片选,拉低CS引脚,操作完成后拉高

RD:读数据,在RD的上升沿,使数据锁存到数据线D[7:0]上

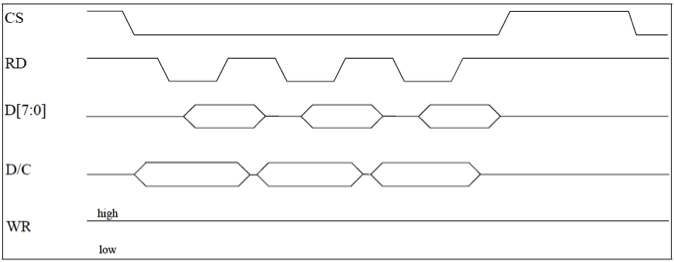


图5 并口读时序

无论读操作还是写操作:

DC=1 数据操作 , DC=0 命令操作

操作OLED前拉低片选CS引脚

都是在上升沿进行数据采集

写操作:

WR的上升沿,数据/命令写入SSD1306

读操作:

RD的上升沿,数据/命令锁存到数据线D[7:0]

DS18B20是常用的数字温度传感器，具有体积小，硬件开销低，抗干扰能力强，精度高的特点。DS18B20数字温度传感器接线方便，封装成后可应用于多种场合。测温范围 －55℃～+125℃，固有测温误差1℃。独特的单线接口方式，DS18B20在与微处理器连接时仅需要一条口线即可实现微处理器与DS18B20的双向通讯。

在初始化序列期间，总线控制器拉低总线并保持至少480us以发送一个复位脉冲，返回释放总线，进入接收状态（等待DS18B20应答）。总线释放后，单总线由上拉电阻拉到高电平。当DS18B20探测到I/O引脚上的上升沿后，等待15-60us，然后其以拉低总线60-240us的方式发出存在脉冲。至此，初始化时序完毕。所以，初始化成功的标志就是能否读到DS18B20这个先低后高的脉冲时序，并且拉低的时间要满足60-240us。

DS18B20初始化时序图如图6所示。

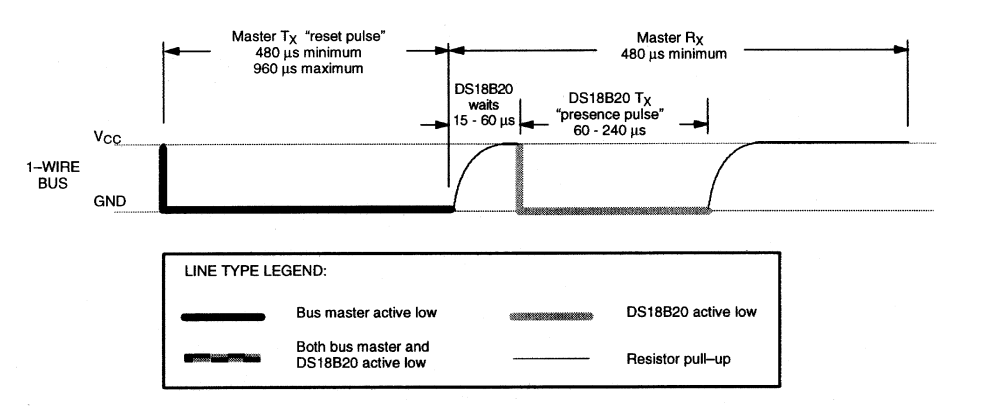


图6 DS18B20初始化时序图

主机在写时隙向DS18B20写入数据，其中分为写”0”时隙，和写”1”时隙。总线主机使用写“1”时间隙向DS18B20写入逻辑1，使用写“0”时间隙向DS18B20写入逻辑0。所有的写时隙必须有最少60us的持续时间，相邻两个写时隙必须要有最少1us的恢复时间。两种写时隙都通过主机拉低总线产生。为了产生写1时隙，在拉低总线后主机必须在15μs内释放总线。在总线被释放后，由于上拉电阻将总线恢复为高电平。为了产生写”0”时隙，在拉低总线后主机必须继续拉低总线以满足时隙持续时间的要求(至少60μs)。在主机产生写时隙后，DS18B20会在其后的15~60us的一个时间段内采样单总线（DQ）。在采样的时间窗口内，如果总线为高电平，主机会向DS18B20写入1；如果总线为低电平，主机会向DS18B20写入0。

DS18B20读时序图如图7所示。

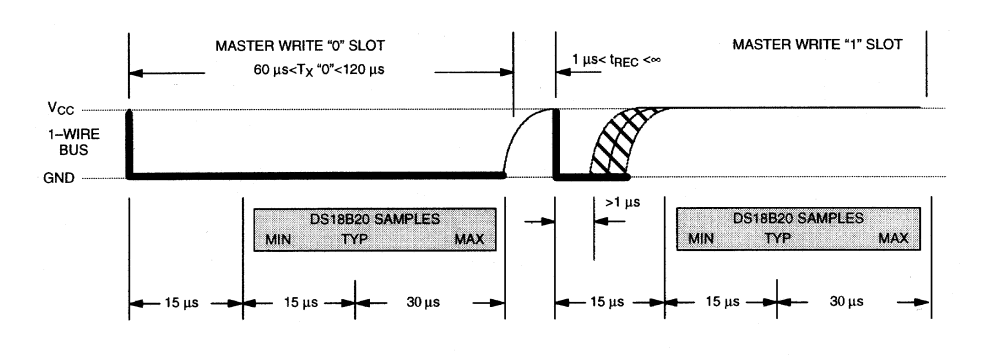


图7 DS18B20读时序图

主机发起读时序时，DS18B20仅被用来传输数据给控制器。因此，总线控制器在发出读指令后必须立刻开始读时序。所有读时序必须最少60us，包括两个读周期间至少1us的恢复时间。当总线控制器把数据线从高电平拉到低电平时，读时序开始，数据线必须至少保持1us，然后总线被释放。DS18B20 通过拉高或拉低总线来传输”1”或”0”。当传输逻辑”0”结束后，总线将被释放，通过上拉电阻回到上升沿状态。

从DS18B20输出的数据在读时序的下降沿出现后15us 内有效。因此，总线控制器在读时序开始后必须把I/O口设置为输入模式，以读取I/O口状态。

DS18B20写时序图如图8所示。

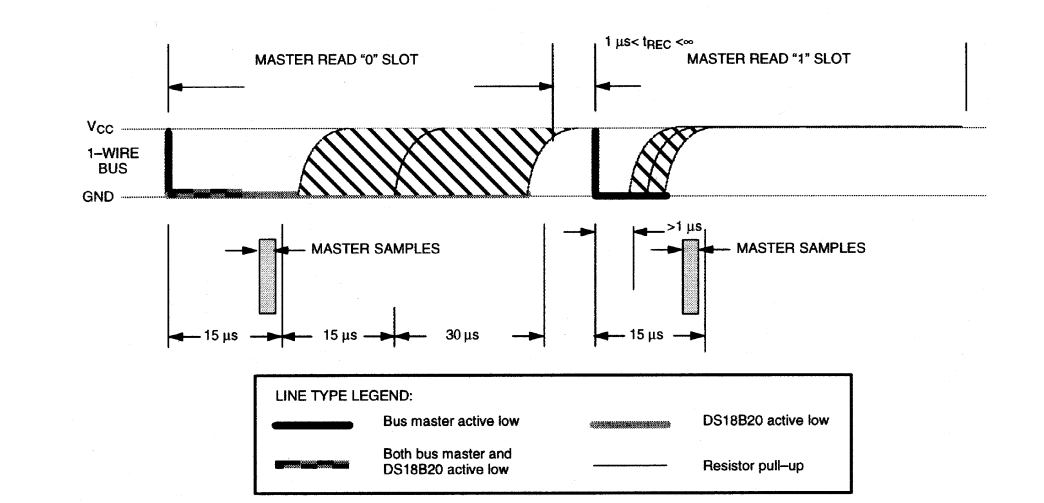


图8 DS18B20写时序图

L298N就是L298的立式封装，源自意法半导体集团旗下品牌产品，是一款可接受高电压、大电流双路全桥式电机驱动芯片，工作电压可达46V，输出电流最高可至4A，采用Multiwatt 15脚封装，接受标准TTL逻辑电平信号，具有两个使能控制端，在不受输入信号影响的情况下通过板载跳帽插拔的方式，动态调整电路运作方式，有一个逻辑电源输入端，通过内置的稳压芯片78MO5，使内部逻辑电路部分在低电压下工作，也可以对外输出逻辑电压5V，为了避免稳压芯片损坏，当使用大于12V驱动电压时，务必使用外置的5V接口独立供电。

L298N通过控制主控芯片上的I/O输入端，直接通过电源来调节输出电压，即可实现电机的正转、反转、停止，由于电路简单，使用方便，通常情况下L298N可直接驱动继电器（四路）、螺线管、电磁阀、直流电机（两台）以及步进电机（一台两相或四相）。

TEC是利用半导体材料珀尔帖制成的，是通过改变输入电流的方向来实现对目标物体进行加热或制冷的。TEC系列产品有密封和不密封两种封装，密封的TEC拥有防水防尘等特点，而不密封的TEC则散热效果更好，效率更高。此模块广泛应用在诸多领域，如固体激光器，冰箱制冷，空调，光学组件，IC，CPU温度调节，相机温度控制等。

[轻触开关](https://link.zhihu.com/?target=http://www.best-dz.com/html/list_1221.html" \t "https://zhuanlan.zhihu.com/p/_blank)是一种电子开关，使用时轻轻点按开关按钮就可使开关接通，当松开手时开关即断开，其内部结构是靠金属片受力弹动来实现通断的。轻触开关由于接触电阻小、按动有清脆的手感手感明显、高度规格齐全等方面的原因，在多方面得到广泛的应用。

下图9-12为各模块的电路图。

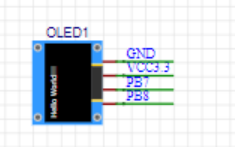
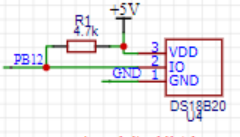


图9 OLED显示屏电路图 图10 DS18B20电路图

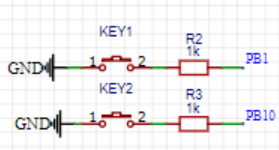
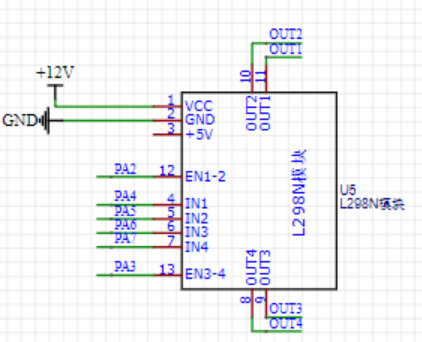


图11 （TEC温度控制驱动）L298N电路图 图12（调温驱动）按键电路图

晶振电路都是在一个反相放大器的两端接入晶振，再有两个电容分别接入到晶振的两端，另一个电容则接地，这两个电容串联的电容量就等于负载电容。采用32768HZ的陶瓷晶振启动电路，实现单片机的启动。

复位电路是一种用来使电路恢复到起始状态的电路设备，它的操作原理与计算器有着异曲同工之妙，只是启动原理和手段有所不同。复位电路，就是利用它把电路恢复到起始状态。就像计算器的清零按钮的作用一样，以便回到原始状态，重新进行计算。采用了一个按键，实现对于整个系统的硬件复位功能。

电源电路是指[车载功放](https://baike.sogou.com/lemma/ShowInnerLink.htm?lemmaId=545364&ss_c=ssc.citiao.link" \t "https://baike.sogou.com/_blank)的电源部分的设计，使用的电路形式和特点。对于一个功放来说，其电源部分非常重要，专业功放的电源电路的容量往往是根据放大器的实际消耗，再加足够的富裕量，因此比同样标称功率的普通功放的容量大得多，因此电源电路可以从一个侧面反映出整个功放的好坏。采用了两个电容，实现了电路的隔支流和滤波的作用，使电源电路工作更高效。

不论是STM32还是其他单片机，都会有BOOT引脚，这个BOOT引脚可以是1个也可以是2个，这里是一个BOOT0引脚，由于启动方式的不同配置，还可能有BOOT1引脚，通过这一组的BOOT引脚的上下拉电平，就可以通过硬件配置启动方式。采用了BOOT0=0,BOOT1=0/1,实现启动模式为主闪存存储器。

图13-16为晶振电路、复位电路、电源电路和BOOT模式选择电路图。

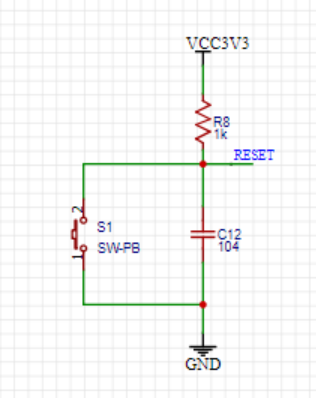
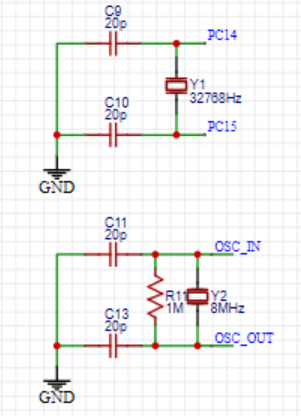


图13 晶振电路图 图14 复位电路图

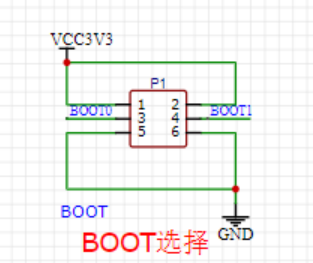
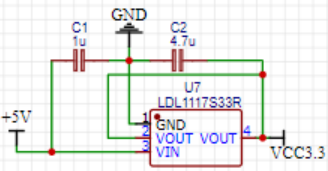


图15 电源电路图 图16 BOOT选择电路图

# 3软件设计

## 3.1主程序设计

主要是对 MCU进行初始化设定，包含了时钟晶振等，并进行了串口中断，数据的接收、处理、传输，并调用各个子程序，使得整个系统可以正常工作，达到预定的功能。

该系统的主程序主要包括OLED显示屏驱动、步进电机驱动和各个模块程序的头文件，其中主程序的流程图见图17。

图示

描述已自动生成

图17 主程序流程图

## 3.2 软件系统开发平台

Keil是 ARM公司于2009年推出的一款适合于新手编程的软件，随着各种嵌入式芯片的不断增加，它的性能也在不断地升级，与其他语言相比，C语言也有很多的优点和用处，主要包括编写、编译、调试代码，还内置ST-Link烧录功能，输出二进制的信号；具有模拟功能、查询语句出错；可以进行基础 MCU的开发

由于现有的 C语言和汇编语言都能在 Keil中实现，所以本设计使用 keil来实现软件的开发。使用方法也非常简单，首先创建一个新的文件，创建一个新的项目，然后键入一个文件名，再选用所需要的单片机，然后就成功地创建了工程文件。最后可以按照自己需要的功能，添加不同的内容，进行软件的设计。

在 keil中编写程序，调试，编译，然后按照下面的窗口提示，反复的对程序进行检查，发现 bug并进行修正，直至调试成功，达到了目的。

keil调试窗口如下图18-21所示。

图形用户界面, 文本, 应用程序

描述已自动生成

图形用户界面, 文本, 应用程序, 电子邮件

描述已自动生成

图形用户界面, 文本, 应用程序

描述已自动生成

图形用户界面, 应用程序

描述已自动生成

图18-21 调试图

# 4软件系统实现的主要功能

1. 温度数据的测量及显示
2. 目标温度的设置（加或减）
3. 温度的控制

# 5用于温度控制的微型模块软件设计

## **5.1用于温度控制的微型模块控制程序**

int main(void)

{

int8\_t Speed;

int8\_t Speed2;

uint16\_t Temp;

SETTEMP=25;

Motor\_Init();

Cooling\_Init();

Key\_Init();

OLED\_Init();

DS18B20\_UserConfig();

PID\_Init();

PID2\_Init();

TIM3\_Int\_Init(10-1,7200);

PWM\_Init();

OLED\_ShowString(1, 1, "Speed");

OLED\_ShowString(1, 7,"Temp");

OLED\_ShowString(1, 13,"Mode");

OLED\_ShowString(3, 1,"SetTp");

while(1)

{

KeyNum=Key\_GetNum();

if(KeyNum==2)

{

SETTEMP+=5;

if(SETTEMP>45)

SETTEMP=25;

Hot\_flag=1;

}

if(KeyNum==1)

{

SETTEMP-=1;

if(SETTEMP<15)

SETTEMP=25;

Cool\_flag=1;

}

Temp=DS18B20\_Read\_Temp();

if(flag100ms==1) //每100ms更新一次实时温度

{

OLED\_ShowString(4, 8, "Yes");

flag100ms=0;

flag500ms++;

if(flag500ms==5)

{

flag500ms=0;

TEMP=Temp/10;

OLED\_ShowTemp(2,7,Temp,4); //500ms测量一次温度

OLED\_ShowSignedNum(4,1,SETTEMP,3);

if(SETTEMP<25)

{

//Cool\_flag=1;

Speed2=PID2\_realize();

Cooling\_SetSpeed(Speed2);

OLED\_ShowString(2, 13, "Cool");

OLED\_ShowNum(2,1,Speed2,4);

Motor\_SetSpeed(0);

}

if(SETTEMP>=25&&SETTEMP<=30)

{

Cooling\_SetSpeed(0);

Motor\_SetSpeed(0);

OLED\_ShowString(2, 13, "Wait");

OLED\_ShowString(2, 1, "Wait");

}

if(SETTEMP>30)

{

Speed=PID\_realize();

Motor\_SetSpeed(Speed);

Cooling\_SetSpeed(0);

OLED\_ShowString(2, 13, "Warm");

OLED\_ShowNum(2,1,Speed,4);

}

}

}

if(flag100ms==0)

{

flag100ms=0;

flag500ms++;

if(flag500ms==5)

{

flag500ms=0;

OLED\_ShowTemp(2,7,Temp,4); //500ms测量一次温度

}

OLED\_ShowString(2, 1, "Stop");

OLED\_ShowString(2, 13, "Stop");

OLED\_ShowString(4, 1, "Stop");

OLED\_ShowString(4, 8, "Not");

}

}

}

## 5.2用于温度控制的微型模块的中断代码

void TIM3\_IRQHandler(void)

{

static u8 tmr100ms;

if(TIM\_GetITStatus(TIM3,TIM\_IT\_Update) == SET)

{

//KeyScan(); //按键扫描

tmr100ms++;

if(tmr100ms>=100)

{

tmr100ms=0;

flag100ms=1;

}

}

TIM\_ClearITPendingBit(TIM3,TIM\_IT\_Update);

}

## 5.3用于温度控制的微型模块的温度设置代码

uint8\_t Key\_GetNum(void)

{

uint8\_t KeyNum = 0;

if (GPIO\_ReadInputDataBit(GPIOB,GPIO\_Pin\_1)==0)

{

Delay\_ms(20);

while(GPIO\_ReadInputDataBit(GPIOB,GPIO\_Pin\_1)==0);

Delay\_ms(20);

KeyNum=1;

}

if (GPIO\_ReadInputDataBit(GPIOB,GPIO\_Pin\_10)==0)

{

Delay\_ms(20);

while(GPIO\_ReadInputDataBit(GPIOB,GPIO\_Pin\_10)==0);

Delay\_ms(20);

KeyNum=2;

}

return KeyNum;

}