

硬件设计实验

实验报告

实验项目名称： 智慧闹钟

专 业 班 级：

姓 名：

学 号：

指 导 教 师：

完 成 时 间：

目录

[一、 设计目标](#_Toc7925_WPSOffice_Level1) [4](#_Toc7925_WPSOffice_Level1)

[二、 设计想法](#_Toc22582_WPSOffice_Level1) [4](#_Toc22582_WPSOffice_Level1)

[三、 硬件原理](#_Toc32259_WPSOffice_Level1) [5](#_Toc32259_WPSOffice_Level1)

[1. 无源蜂鸣器](#_Toc20303_WPSOffice_Level2) [5](#_Toc20303_WPSOffice_Level2)

[2. LED数码管电路](#_Toc11958_WPSOffice_Level2) [5](#_Toc11958_WPSOffice_Level2)

[3. 振动传感器](#_Toc22939_WPSOffice_Level2) [6](#_Toc22939_WPSOffice_Level2)

[4. 温度传感器和光照传感器](#_Toc30031_WPSOffice_Level2) [7](#_Toc30031_WPSOffice_Level2)

[5. CPU引脚分配](#_Toc9725_WPSOffice_Level2) [11](#_Toc9725_WPSOffice_Level2)

[四、 软件原理](#_Toc30434_WPSOffice_Level1) [11](#_Toc30434_WPSOffice_Level1)

[1. 数码管显示模块](#_Toc15809_WPSOffice_Level2) [11](#_Toc15809_WPSOffice_Level2)

[2. 温度光照传感器的AD转换模块](#_Toc6526_WPSOffice_Level2) [13](#_Toc6526_WPSOffice_Level2)

[3. 蜂鸣器判断模块](#_Toc31709_WPSOffice_Level2) [14](#_Toc31709_WPSOffice_Level2)

[4. 蜂鸣器发声模块](#_Toc6119_WPSOffice_Level2) [15](#_Toc6119_WPSOffice_Level2)

[五、 软件流程](#_Toc24669_WPSOffice_Level1) [16](#_Toc24669_WPSOffice_Level1)

[六、 设计结果](#_Toc11989_WPSOffice_Level1) [17](#_Toc11989_WPSOffice_Level1)

[七、 操作说明](#_Toc19749_WPSOffice_Level1) [17](#_Toc19749_WPSOffice_Level1)

[1. 按键说明](#_Toc20100_WPSOffice_Level2) [17](#_Toc20100_WPSOffice_Level2)

[2. 显示说明](#_Toc19952_WPSOffice_Level2) [17](#_Toc19952_WPSOffice_Level2)

[3. 操作说明](#_Toc23448_WPSOffice_Level2) [18](#_Toc23448_WPSOffice_Level2)

[八、 设计日志](#_Toc5565_WPSOffice_Level1) [18](#_Toc5565_WPSOffice_Level1)

[九、 后续学习](#_Toc20303_WPSOffice_Level1) [18](#_Toc20303_WPSOffice_Level1)

[附录：源代码](#_Toc11958_WPSOffice_Level1) [19](#_Toc11958_WPSOffice_Level1)

1. 设计目标

1. 通过无源蜂鸣器实现闹钟响铃功能；

2. 通过光照传感器实现闹钟开启功能；

3. 通过振动传感器实现闹钟关闭功能；

4. 通过温度传感器实现温度测量功能；

5. 通过按键KEY1实现蜂鸣器变调功能；

6. 通过按键KEY2实现闹钟开关的功能；

7. 通过数码管显示温度、闹钟开关和对应的音调；

1. 设计想法

由于社会的发展和科技的进步，人们对生活品质要求越来越高，而计算机时代的来临和物联网技术的发展也会给我们的生活带来巨大的变化，所以我希望通过物联网传感技术实现对闹钟的创意设计，从而改进人们的生活品质。

首先相对于传闹钟设置闹铃的方式，我想把这款创意闹钟设计成通过光照传感器来实现对闹钟的开启，更有效实现闹钟的自动化、智能化，不过为了防止误触带来闹钟的开启，需要对开启条件进行考量。其次我希望改变传统闹钟按键关闭的方式，实现通过运动或手部摆动帮助用户清醒，所以需要通过振动传感器关闭闹钟的功能。然后我希望闹钟可以实现温度显示的功能，帮助用户识别当前温度，辨别冷暖，适当的增减衣物。最后通过按键实现闹钟的总开关和闹铃音调的改变。

所以在本次实验中，我通过三个按键，光照传感器、温度传感器、振动传感器以及数码管对上述想法进行实现。

1. 硬件原理
2. 无源蜂鸣器

无源蜂鸣器利用电磁感应现象，为音圈接入交变电流后形成的电磁铁与永磁铁相吸或相斥而推动振膜发声，接入直流电只能持续推动振膜而无法产生声音，只能在接通或断开时产生声音。无源蜂鸣器的工作原理与扬声器相同。在使用方波信号源驱动的应反向并联一个二极管，防止突然断电时产生的高压反向电动势击穿其他元件以及使用寿命缩短。

有源蜂鸣器往往比无源的贵，就是因为里面多个震荡电路，只需接入额定电压的直流电即可发出指定频率的声音，频率由内部振荡电路决定，无法改变。而无源蜂鸣器内部不带震荡源，所以如果用直流信号无法令其鸣叫。必须用2K~5K的方波去驱动它。因此，无源蜂鸣器的优点是：制作成本低且声音频率范围宽，可高分贝的发出某些频率的超声波以及可以做出不同音调的发声效果。

根据如上特性，我们可以用蜂鸣器发声用做提醒需要。而针对不同情况，我们可以将不同的音调用于不同情况的提醒需要。用num记录当前音调数值，在最后一位数码管上显示音调。在操作过程中，通过改变重装值，从而改变定时器中断频率，对声音频率进行改变。当每次摁下KEY1时，num数值改变，当num到达9的时候，再次摁下KEY1，num变为0。

1. 无源蜂鸣器电路原理图

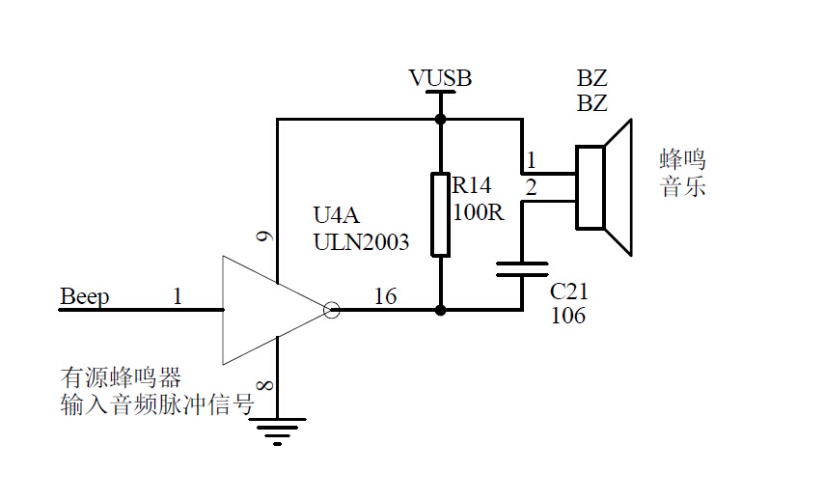


图1 无源蜂鸣器电路原理图

1. 如何修改输入蜂鸣器方波的频率

输入蜂鸣器的方波是由定时器产生的，定时器中断的频率决定了输入蜂鸣器方波的频率，而定时器中断的频率是通过重装值确定的。因此，在操作过程中，通过改变重装值，从而改变定时器中断频率，对声音频率进行改变。这样，就可以让蜂鸣器发出不同频率的蜂鸣声。

1. LED数码管电路

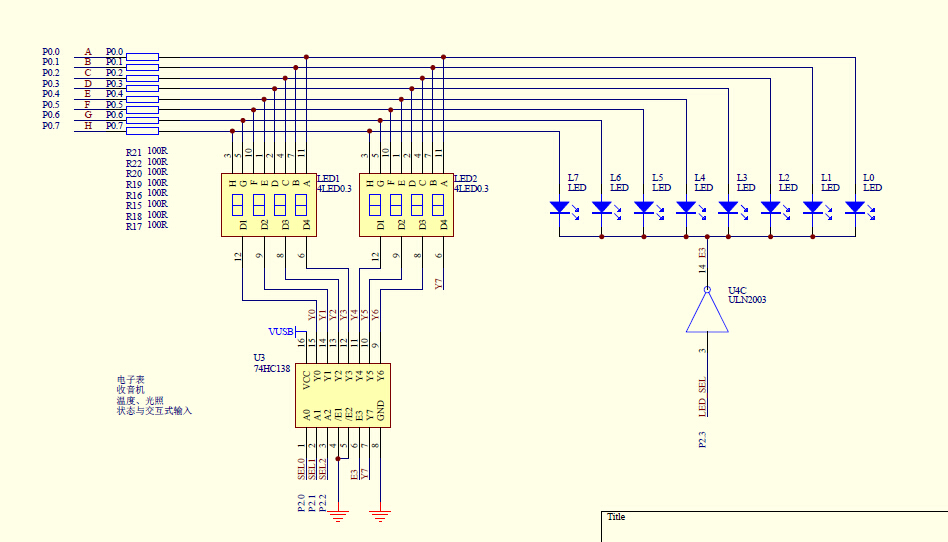


图2 数码管与发光二极管硬件电路图

1. LED数码管引脚

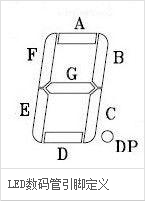


图3 LED数码管引脚定义

1. LED数码管工作原理

P0口的8位输出分别控制1个LED数码管的7段和一个小数点；而P2.3经反相器U4C控制74HC138的使能信号E3，结合P2.0、P2.1、P2.2这3个位选控制信号确定8个LED数码管中的哪个被点亮；电阻R15～R22为限流电阻。当段选为高、使能信号有效时，对应的LED管将会发光。通过以一定频率扫描位选信号，修改段选信号进行数码管点亮一段时间，从而给人视觉上几个数码管几乎同时显示的效果。

1. 振动传感器

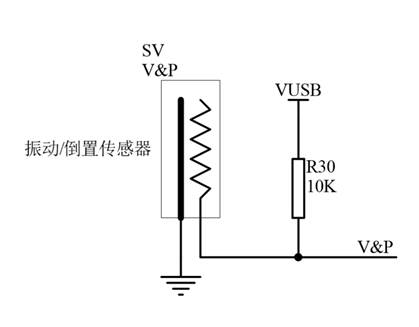


图4  振动传感器电路原理图

本实验板中使用的振动传感器是一种简单的器件，管内有一跟固定的导线，在这根导线的周围有另一根较细的导线以螺旋状环绕它。可以想象为一个弹簧旁边有一跟导线。在不振动时，两根导线不会相碰，一旦振动发生，两根导线就会短接。所以我们只需判断导线是否短接了，就可以知道振动是否发生。

1. 温度传感器和光照传感器
2. ADC编程设计流程图

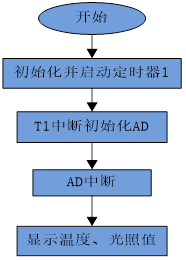


图5 ADC编程设计流程图

1. 相关寄存器设置

（1） P0（8位）和P2.3需要设置成推挽输出，以驱动电路正常发光。按键作为输入， 不需推挽，涉及寄存器及配置值如下：

P0M1=0x00;

P0M0=0xff;

P2M1=0x00;

P2M0=0x08;

（2） 采用定时器1，在定时器中断中进行AD的初始化，涉及寄存器（含可位寻址） 及配置如下：

IE=0xa8;

TH1=(65535-40000)/256;

TL1=(65535-40000)%256;

TR1=1; //启动定时器

（3） 对于温度通过AD采集，涉及寄存器及配置如下：

P1ASF=0xff;

ADC\_RES=0;

ADC\_RESL=0; //AD转换结果寄存器清0

ADC\_CONTR=0X8B; //10001011，后三位决定P1^3作为A/D输入来用;

ADC\_START(右4位)为1，开始转换；最高位ADC\_Power=1，打开A/D转换器电 源CLK\_DIV=0X20;

1. AD中断处理程序

（1）每中断AD一次，计数器time加1；

（2）屏蔽所有中断；

（3）如果中断次数达到2000次，则如果flag为1，则求温度AD平均值，并从tempdata数组中获取相应AD值对应的摄氏温度值，AD中断重新计数，调用date\_processtemp()方法完成数码管左边三个温度值显示。如果flag为-1，则求光AD平均值，AD中断重新计数，调用date\_processlight()方法用数码管右侧三个显示光信息。

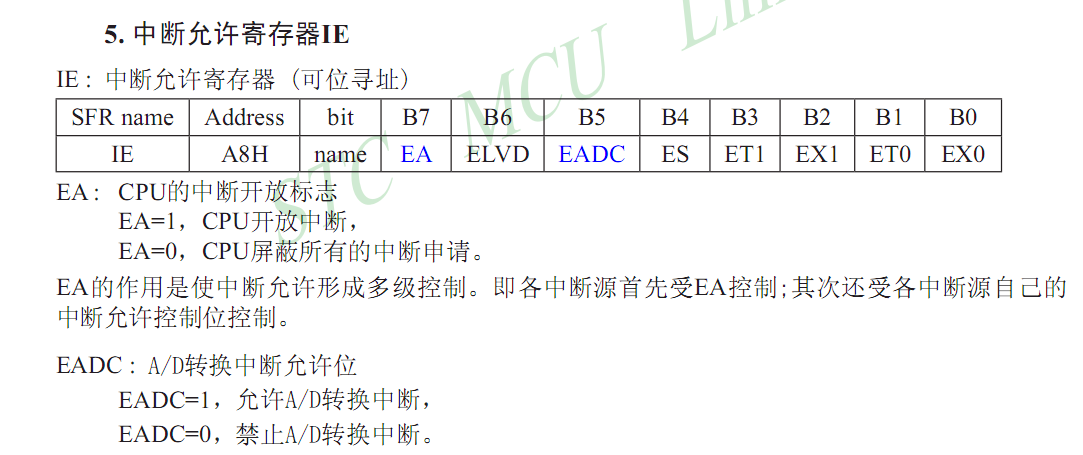
（4）如果flag为1，则将存储了温度AD值的ADC\_RES和ADC\_RESL寄存器数值经过处理后累加，温度中断次数加1；如果flag为-1，则将存储了光AD值的ADC\_RES和ADC\_RESL寄存器数值累加，光中断次数加1。

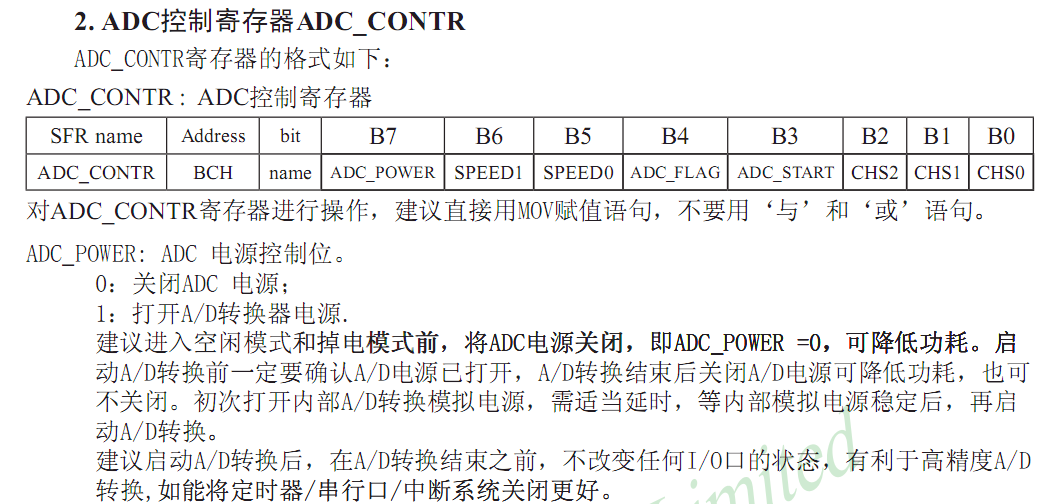
（5）标志本次数模转换结束，并重新开放AD中断和定时中断。

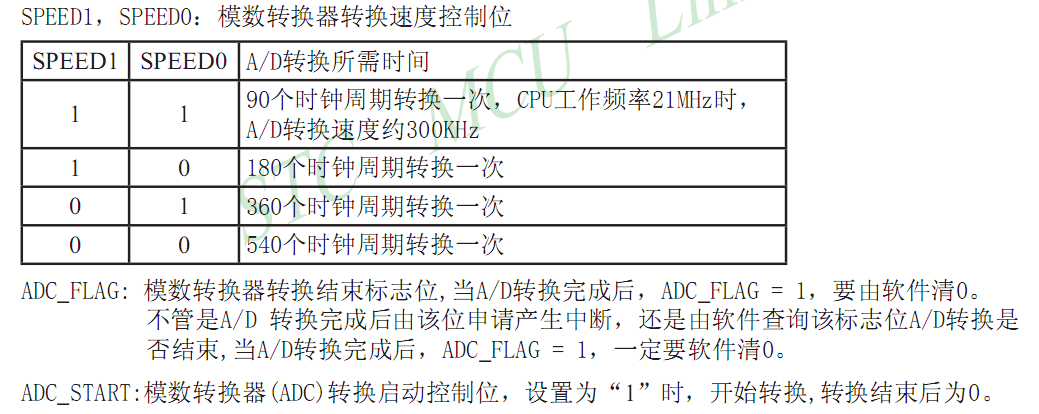
ADC\_CONTR&=~0X10; //ADC\_FLAG=0，数模转换结束标志被程序清为0

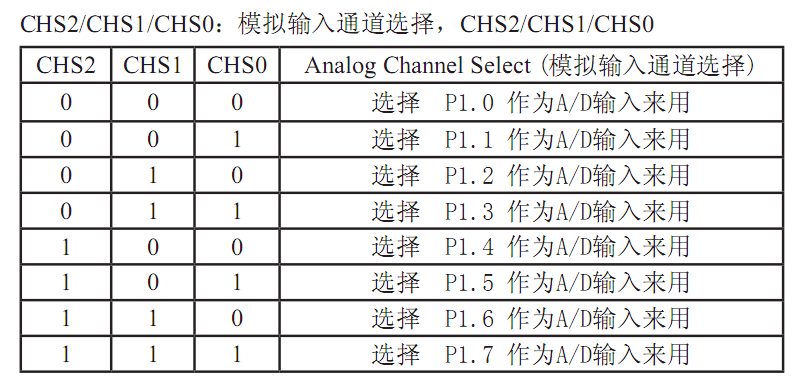
ADC\_CONTR|=0X08; //ADC\_START=1，模数转换启动，结束后自动为0

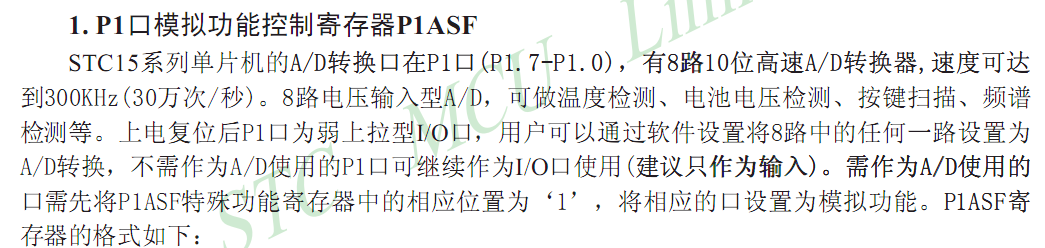
1. 相关寄存器知识

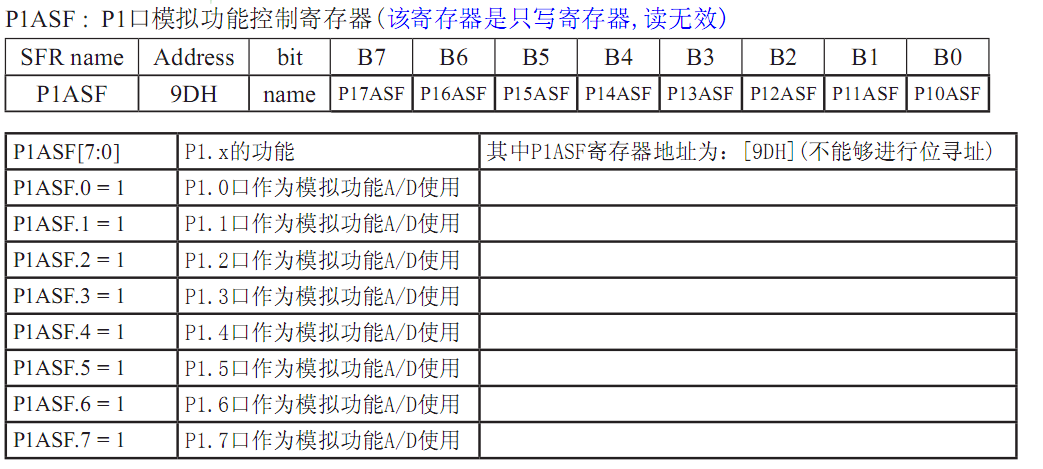


****

****

****





1. **interrupt 和 using**

interrupt 和 using 都是 C51 的关键字。C51 中断过程通过使用 interrupt 关键字和中断号(0 到 31)来实现。中断号指明编译器中断程序的入口地址中断序号对应着 8051中断使能寄存器IE 中的使能位，对应关系如下：

IE寄存器 C51中的 8051的的使能位 中断号 中断源

IE.0 0 外部中断0

IE.1 1 定时器0 溢出

IE.2 2 外部中断1

IE.3 3 定时器1 溢出

IE.4 4 串口中断

IE.5 5 定时器2 溢出

有了这一声明，编译器不需理会寄存器组参数的使用和对累加器A、状态寄存器、寄存器B、数据指针和默认的寄存器的保护。只要在中断程序中用到，编译器会把它 们压栈，在中断程序结束时将他们出栈。C51 支持所有 5 个 8051 标准中断从 0 到 4 和在 8051 系列（增强型）中多达 27 个中断源。

using 关键字用来指定中断服务程序使用的寄存器组。用法是：using 后跟一个0 到3 的数，对应着 4 组工作寄存器。一旦指定工作寄存器组，默认的工作寄存器组就不会被压栈，这将节省 32 个处理周期，因为入栈和出栈都需要 2 个处理周期。这一做法的缺点是所有调用中断的过程都必须使用指定的同一个寄存器组，否则参数传递会发生错误。因此对于using，在使用中需灵活取舍。

1. 温度电路图

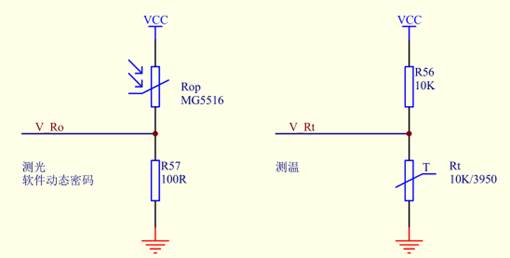


图6 温度电路图

1. CPU引脚分配

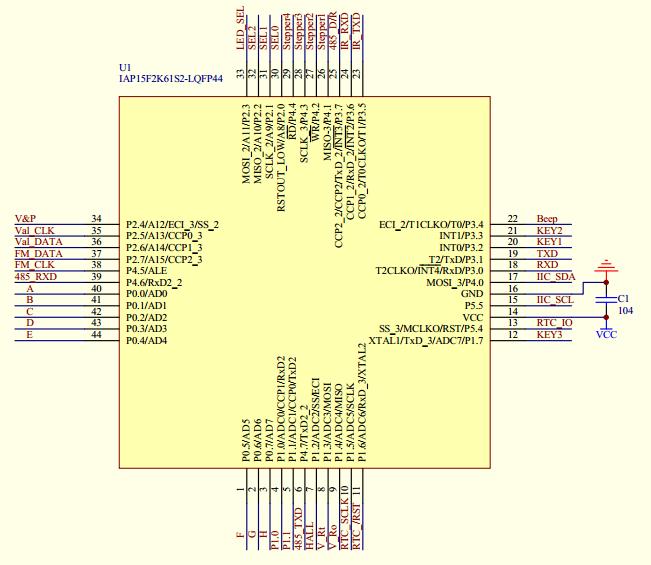


图7 芯片引脚电路图

1. 软件原理
2. 数码管显示模块
3. **数组段选用于存储相应的数码管显示的段选信号，函数weixuan(char i)用于选择对应的位置**

char segtable[]={0x3f,0x06,0x5b,0x4f,0x66,0x6d,0x7d,0x07,

0x7f,0x6f,0x77,0x7c,0x39,0x5e,0x79,0x71};

void weixuan(char i)

{

SEL2=i/4;

SEL1=i%4/2;

SEL0=i%2;

}

1. **数码管显示函数**

void SEG\_Display()

{

//用于设置温度的位数

if(temp<0)

{

P0=0;

weixuan(0); //若温度为负，则首位显示‘-‘

P0=0x40;

Delay(10);

}

else

{

P0=0;

weixuan(5);

P0=segtable[temp\_bai];

Delay(10);

}

P0=0;

weixuan(6);

P0=segtable[temp\_shi];

Delay(10);

P0=0;

weixuan(7);

P0=segtable[temp\_ge];

Delay(10);

P0=0;

weixuan(1); //显示闹钟开关

P0=segtable[ringopen];

Delay(10);

P0=0;

weixuan(3); //显示闹铃音调

P0=segtable[num];

Delay(10);

}

1. 温度光照传感器的AD转换模块
2. 使用定时器1的中断函数为温度ADC计时

IE=0xa8; //EA=1打开总中断，EADC=1允许A/D转化中断，ET1=1允许T1中断

TMOD=0x01; //使用定时器1，16位不可重装载模式，TH1、TL1全用

TH1=(65535-1000)/256;//高8位赋初值，定时40000周期

TL1=(65535-1000)%256;//低8位赋初值

TR1=1; //启动定时器1

1. 初始化温度光照AD转换

void InitADC\_temp()

{

P1ASF=0xff; //将P1口作为模拟功能A/D使用

ADC\_RES=0; //寄存器ADC\_RES和ADC\_RESL保存A/D转化结果

ADC\_RESL=0; //初始赋值0

ADC\_CONTR=0x8b; //ADC\_POWER=1打开A/D转换器电源;ADC\_START=1启动模拟转换器ADC;CHS=011选择P1^3作为A/D输入使用

CLK\_DIV=0x20; //ADRJ=1：ADC\_RES[1:0]存放高2位ADC结果，ADC\_RESL[7:0]存放低8位ADC结果

}

void InitADC\_light()

{

P1ASF=0xff;

ADC\_RES=0;

ADC\_RESL=0;

ADC\_CONTR=0x8c; //CHS=100选择P1^4作为A/D输入使用

CLK\_DIV=0x20;

}

1. AD中断，当flag为0时查找对应AD的温度值，处理温度部分数据，当flag为1时查找对应AD的光照值，处理光照部分数据，转换完成后，ADC\_FLAG清零，ADC\_START赋1，打开中断。

time++;

IE=0x00; //关闭中断

if(time>2000) //取多次值求平均值减小误差

{

if(flag==1) //此时flag=1执行温度部分

{

date\_temp=(sumt+t/2)/t; //四舍五入

temp=tempdata[date\_temp-1]; //查找对应的AD的温度值

sumt=0;

t=0;

time=0;

date\_processtemp();

}

if(flag==-1) //此时flag=-1执行光部分

{

light=(suml+l/2)/l; //四舍五入

suml=0;

l=0;

time=0;

}

}

if(flag==1)

{

//对应温度的数据处理

t++;

date\_temp=(ADC\_RES\*256+ADC\_RESL)/4;//由10位AD值转换为8位AD值

sumt+=date\_temp; //求t次AD值的和

}

if(flag==-1)

{

//处理光部分的数据

l++;

suml+=ADC\_RES\*256+ADC\_RESL;//求l次AD值的和

}

1. 蜂鸣器判断模块
2. 定时器中断处理，进行判断，满足光照条件开启蜂鸣器，满足振动条件关闭蜂鸣器

if(ringopen==1){ //判断按键开关

if(light<40){ //判断之前是否黑夜

if(dark<2000){

dark++;} //利用dark进行计数

state=0; //状态为黑夜

}else{

if(white<2000){ //判断现在是否白天

white++;} //利用white进行计数

state=1; //状态为白天

}

if(white==2000&&dark==2000&&state==1){//判断从黑夜进入白天且当前为白天

dark=0; //dark white清零且打开开关

white=0;

ring=1;

}

}

if(sbtVib == 0)

{

Delay(100); //延时消除抖动

if(sbtVib == 0)

{

ring=0; //通过振动关闭闹钟

}

}

1. 蜂鸣器发声模块
2. 定时器中断处理，进行判断，满足条件则蜂鸣器取反

if(ring==1&&ringopen==1){//光照开关且按键开关为开时蜂鸣器启动

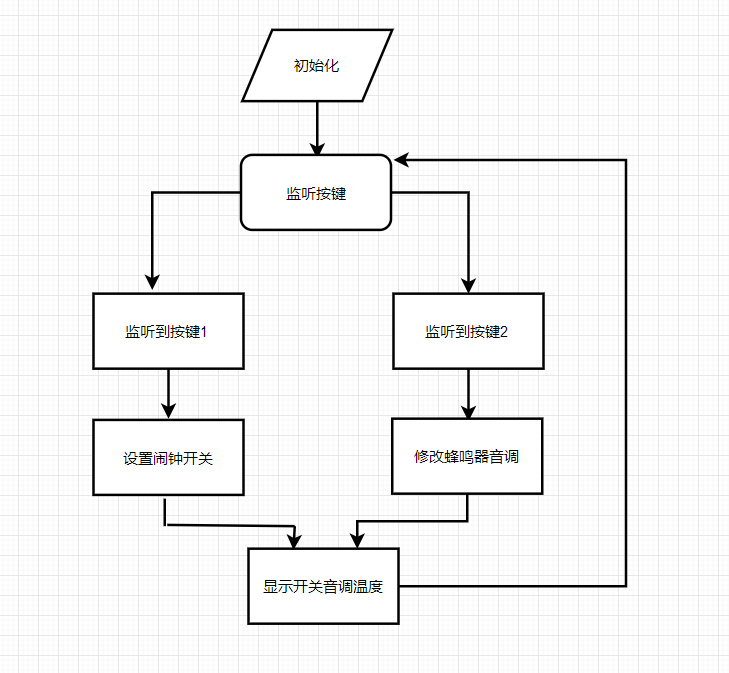
sbtbeep = ~sbtbeep;

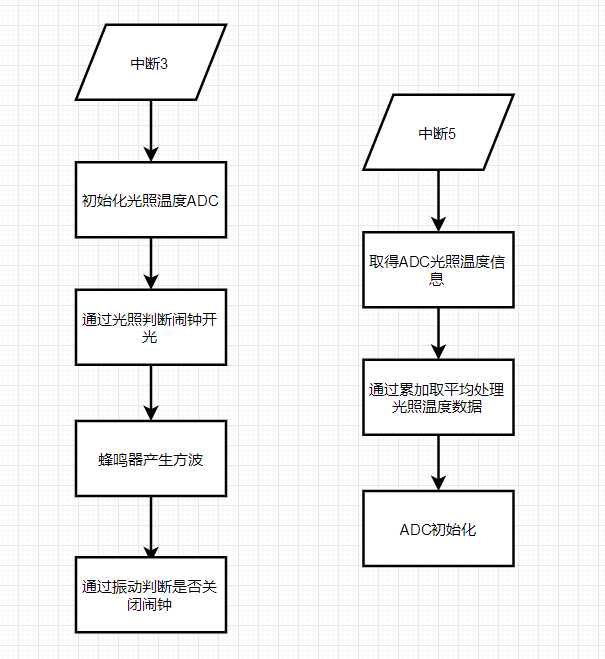
}else{

sbtbeep = 0;

}

1. 软件流程

****

****

1. 设计结果
2. 打开闹钟总开关，遮挡光照传感器十秒钟，取走遮挡物，闹钟开始响，实现光照控制闹钟开启；
3. 按按键2修改音调，蜂鸣器音调改变，实现按键2控制音调；
4. 温度实时显示在数码管，实现温度的实时显示；
5. 振动STC板多次，闹铃声停止，实现振动控制闹钟关闭；
6. 按按键1，关闭闹钟总开关，遮挡光照传感器十秒钟，取走遮挡物，闹钟没有反应，实现按键控制闹钟的总开关。
7. 操作说明
8. 按键说明：
9. 按键1为总开关按键，控制闹钟的开启和关闭，按键1只能在闹钟未开启的状态下进行设置（为了防止通过按键控制闹钟的关闭）；
10. 按键2为音调调节按键，按压按键可以控制闹铃的音调。
11. 显示说明：
12. 第一个数字为闹钟开关按键，0代表开的状态，1代表关的状态；
13. 第二个数字代表闹铃的音调，有0-9十个状态，通过按键2不断递增，数字越大，音调越低；
14. 第三个数字代表温度，当温度为零下时，数码管会显示负数；
15. 操作说明：

通过按键1开启闹钟总开关，从黑夜进入白天，光照传感器接受信息开启闹钟，闹钟开始响铃，通过振动关闭闹钟，数码管第三个数字实时显示当前温度，提醒您适时增减衣物，通过按键2可以改变闹铃的音调。

1. 设计日志
2. 刚开始出现手动干扰时或黑夜进入白天时闹铃都会开启

解决方案：设计white和dark两个计数变量，state状态变量，通过亮度低时dark递增计数state设为0，亮度高时white递增计数state设为1，只有满足dark和white达到一定数量即都需满足黑夜和白天都有一定的时间且当前状态为1即现在状态为白天时，才开启闹钟开关。

1. 闹钟关闭后对闹钟进行黑夜入白天测试没反应，再次开启开关时，闹钟直接开始响

解决方案：设置总的ringopen变量，并设置为最初的条件判断，根据按键2的状态进行改变，当为关闭的状态时，dark和white不发生计数，闹钟不能进入开启状态。

1. 多次测试过程中，发现一次按键会发生多次响应

解决方案：为按键加入防抖程序。

1. 闹铃过程中发现可以通过按键关闭闹钟

解决方案：在按键响应程序中加入条件判断，必须在闹铃没有响的情况下才能置ringopen为1。