

Arduino 旅程的开始

实验室要求开始做传感器方面的实验，我对电路不熟悉啊，怎么办，从头开始学 51 单片机很痛苦的，还是万能的淘宝帮忙了，找到了 Arduino 开发平台，Design in Italy，意大利的团队设计的开发平台及相应的软件和库，功能强大，操作还是很简便的，所以购买的 Arduino Leonardo 版本。

**本店率先先将Arduino套件
从Arduino UNO板子升级为Arduino Leonardo板子。
让您更接近Arduino流行前线。**

基本性能配置对比：

	Arduino Uno	Arduino Leonardo
外观		
主控型号	Atmega328p	Atmega32u4
供电电压	5v	5v
建议供电电压	DC7-12v	DC7-12v
极限供电电压	DC6-20V	DC6-20V
可用 IO	20	20
PWM	7	7
模拟输入 IO	6	12
Flash Memory	32KB	32KB
SRAM	2KB	2.5KB
EEPROM	1KB	1KB
串口	1个	1个硬件 com+1个 usb 虚拟 com
USB 控制器	无	1个

两个版本的对比，大多数的教材和视频都是基于 Arduino Uno 的，两者还是有些许区别的。

好东西不私藏，分享给大家，整理过的 Arduino Uno 和 Leonardo 的资料：

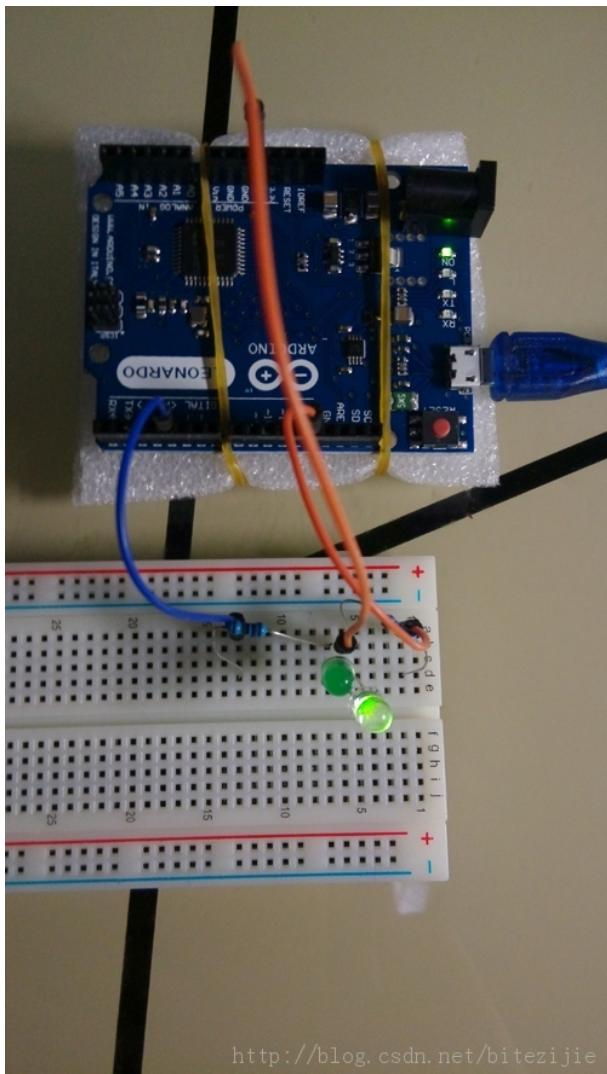
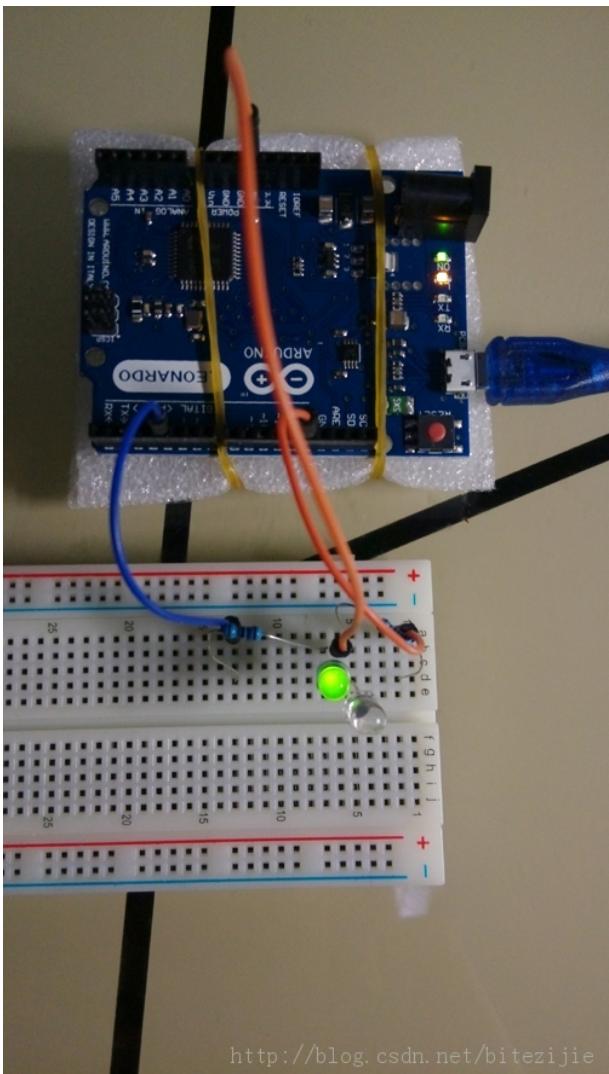
<http://yunpan.cn/QNqz5TzcvwgN3> 访问密码 c7c4

理解了 IO 的概念和定义的方法，更多的代码可以参考 http://kb.open.eefocus.com/index.php?title=Arduino_IDE，开源数据库。

Arduino IDE 是 Arduino 的开放源代码的集成开发环境，其界面友好，语法简单以及能方便的下载程序，使得 Arduino 的程序开发变得非常便捷。作为一款开放源代码的软件，Arduino IDE 也是由 Java、Processing、avr-gcc 等开放源码的软件写成，其另一个最大特点是跨平台的兼容性，适用于 Windows、Max OS X 以及 Linux。2011 年 11 月 30 号 Arduino 官方正式发布了 Arduino1.0 版本，可以下载不同系统下的压缩包，也可以在 github 上下载源码重新编译自己的 IDE。

按照教程，第一个例子就是点亮 LED，第二个例子就是 Hello World。

点亮 LED 的例子，我做了一点点改动，看看效果



两个 IO 口，让绿灯和透明灯一亮一灭。

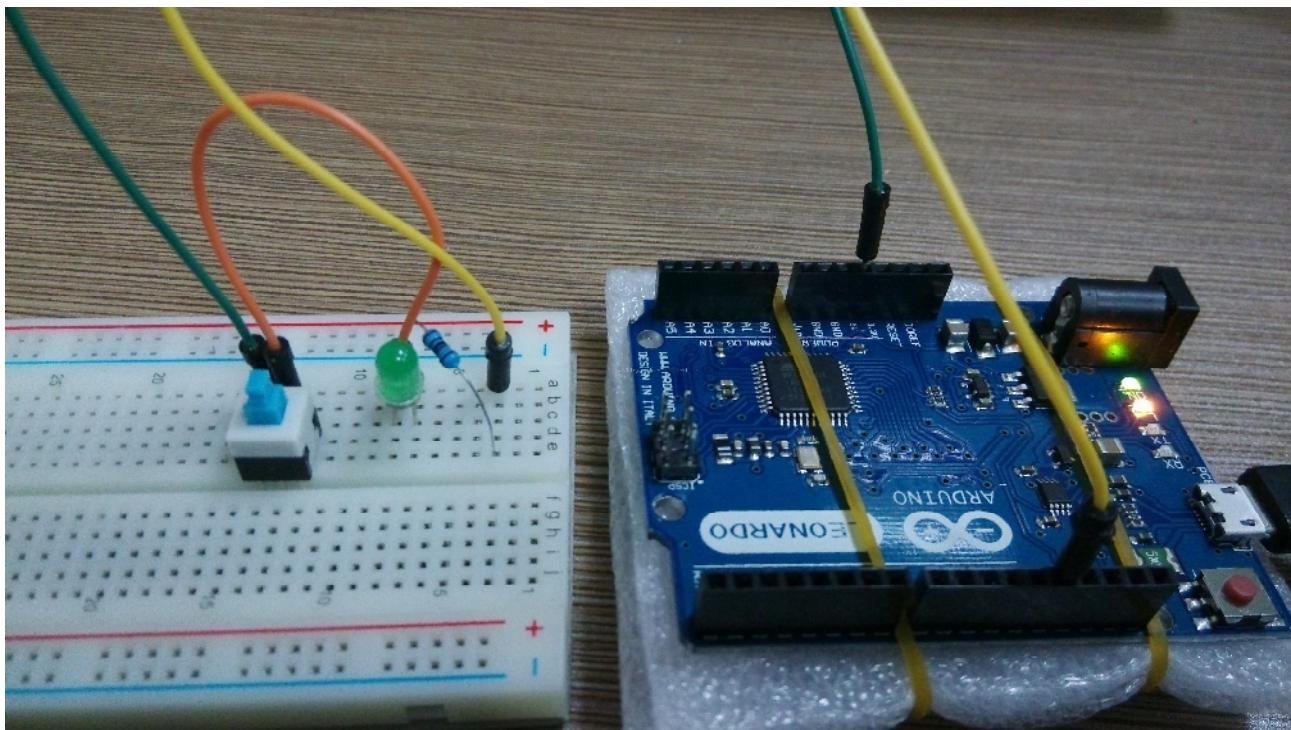
[plain] view plaincopy

```
1. #define leda 4
2. #define ledb 13
3.
4. void setup()
5. {
6.     pinMode(leda, OUTPUT);      // 设置 LED 引脚为输出引脚
7.     pinMode(ledb, OUTPUT);
8. }
9.
10. void loop()
11. {
12.     digitalWrite(leda, 1);    // 设置 LED 引脚输出高电平，点亮 LED
13.     digitalWrite(ledb, 0);
14.     delay(1000);           // 延时 1s
15.     digitalWrite(leda, 0);    // 设置 LED 引脚输出低电平，熄灭 LED
16.     digitalWrite(ledb, 1);
17.     delay(1000);           // 延时 1s
18. }
```

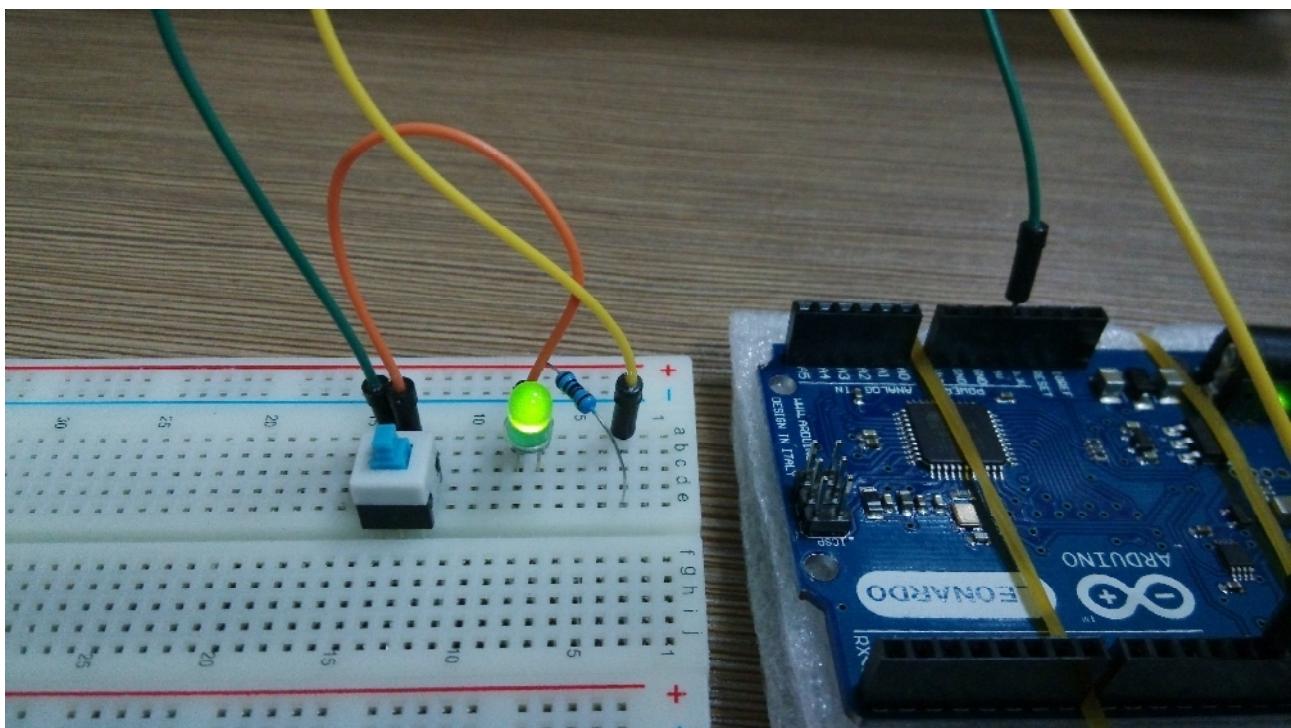
Arduino Leonardo 按键控制 LED 亮与灭

按键控制 LED 亮灭，试用两种方法，一种是最简单的开关，另一种就是用按键的 IO 来控制 LED。

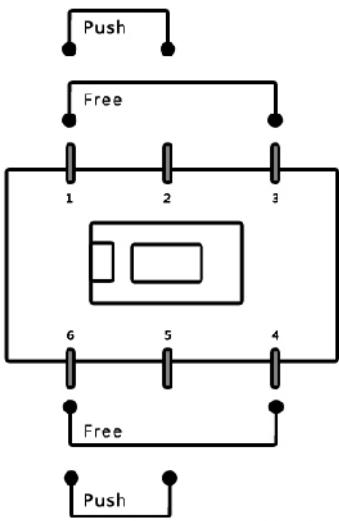
第一种方案：



按键为自锁按键，按键弹起是 LED 是灭的。



按键按下时 LED 点亮。

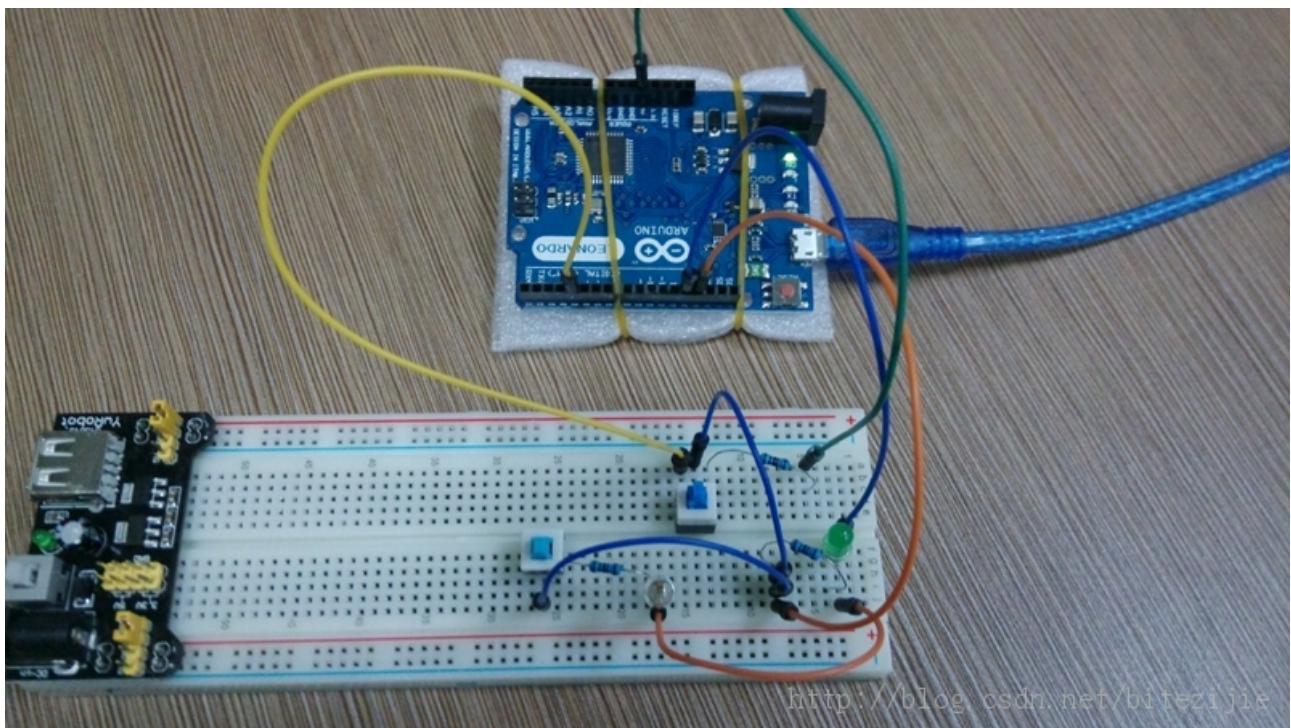


<http://blog.csdn.net/bitezijie>

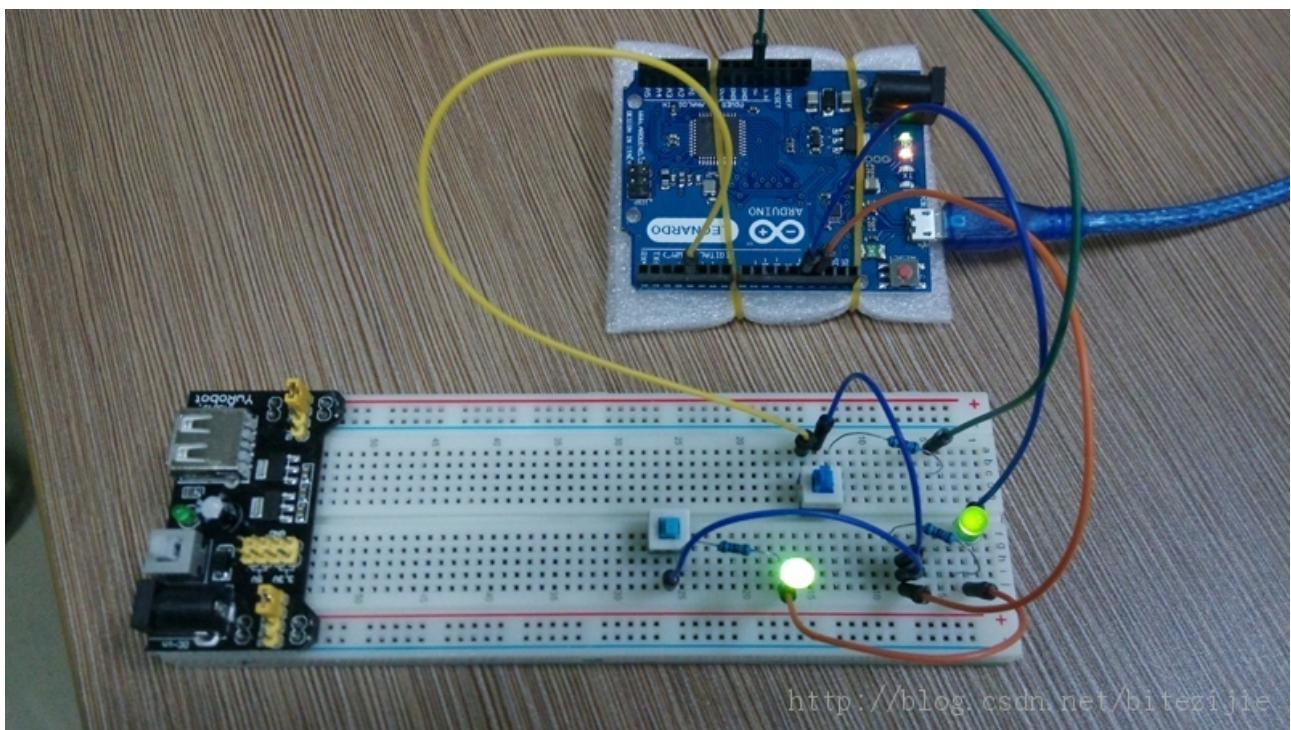
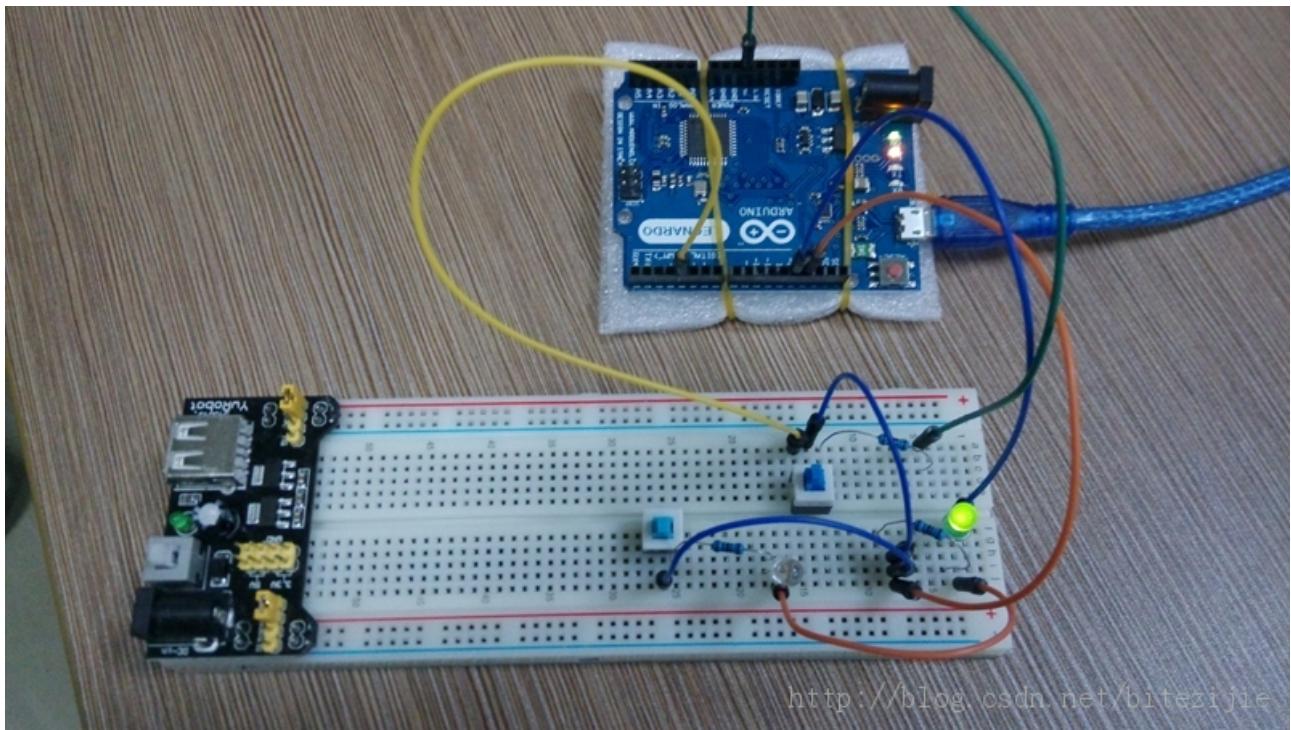
自锁按键的原理。

代码仍用点亮 LED 那段代码即可， I O 给出高电平让 LED 常亮，自锁按键实现开关。

第二种方案



深蓝色非自锁按钮 Free 时有上拉电阻 1k 欧，按下时快速地拉低电平，检测到低电平，LED 亮，再按一次，检测到低电平，LED 翻转电平，熄灭。



浅蓝色自锁按钮如同方案一的功能，这里只是结合起来。

看看代码，也有三种写法：

第一种：

[plain] view plaincopy C

```
1. #define LED 13
2. #define KEY 2
3. unsigned char KEY_NUM = 0;
4. bool Flag_LED = 0;
5.
```

```

6. void setup()
7. {
8.     pinMode(LED, OUTPUT);
9.     pinMode(KEY, INPUT);
10. }
11.
12. void loop()
13. {
14.     Scan_KEY();           //按键扫描
15.     if(KEY_NUM == 1)      //假如按键按下执行该程序
16.     {
17.         KEY_NUM = 0;       //清零标志位
18.         Flag_LED = !Flag_LED;
19.         digitalWrite(LED,Flag_LED); //LED 翻转
20.     }
21. }
22.
23. void Scan_KEY()           //按键扫描
24. {
25.     if( digitalRead(KEY) == 0 ) //查看按键是否按下
26.     {
27.         delay(20);          //延时 20ms, 去抖动
28.         if( digitalRead(KEY) == 0 ) //查看按键是否按下
29.         {
30.             KEY_NUM = 1;
31.             while(digitalRead(KEY) == 0); //松手检测
32.         }
33.     }
34. }
```

第二种：

[\[plain\]](#) [view](#) [plaincopy](#) 

```

1. #define LED 13
2. #define KEY 4
3. int KEY_NUM = 0;           //按键键值存放变量, 不等于 1 说明有按键按下
4.
5. void setup()
6. {
7.     pinMode(LED,OUTPUT);    //定义 LED 为输出引脚
8.     pinMode(KEY,INPUT_PULLUP); //定义 KEY 为带上拉输入引脚
9. }
10.
11. void loop()
12. {
13.     ScanKey();           //按键扫描程序, 当按键按下时候, 该子程序会修改 KEY_NUM 的值
```

```
14. if(KEY_NUM == 1) //是否按键按下
15. {
16.     digitalWrite(LED,!digitalRead(LED));//LED 的状态翻转
17. }
18. }
19.
20. void ScanKey() //按键扫描程序
21. {
22.     KEY_NUM = 0; //清空变量
23.     if(digitalRead(KEY) == LOW) //有按键按下
24.     {
25.         delay(20); //延时去抖动
26.         if(digitalRead(KEY) == LOW) //有按键按下
27.         {
28.             KEY_NUM = 1; //变量设置为 1
29.             while(digitalRead(KEY) == LOW); //等待按键松手
30.         }
31.     }
32. }
```

第三种

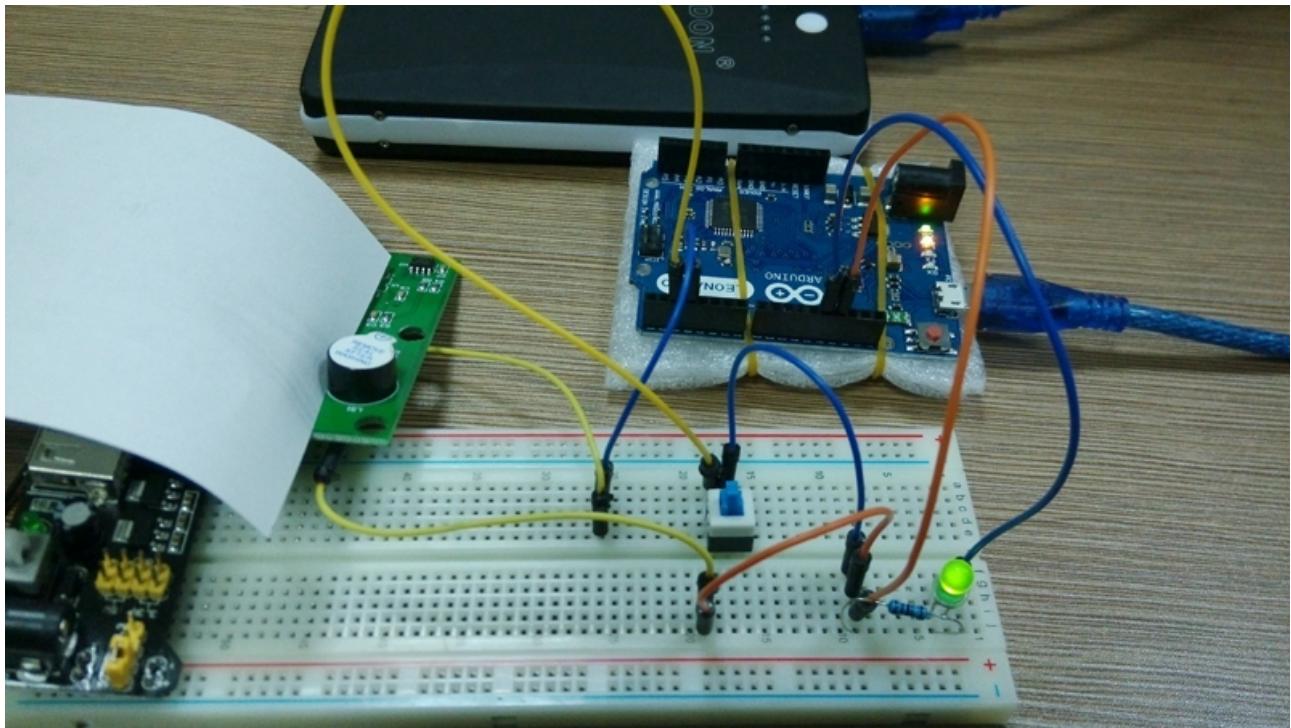
[plain] view plaincopy 

```
1. #define LED 13
2. #define KEY 4
3. int KEY_NUM = 0; //按键键值存放变量，不等于 1 说明有按键按下
4.
5. void setup()
6. {
7.     pinMode(LED,OUTPUT); //定义 LED 为输出引脚
8.     pinMode(KEY,INPUT_PULLUP); //定义 KEY 为带上拉输入引脚
9. }
10.
11. void loop()
12. {
13.     if(digitalRead(KEY) == LOW) //有按键按下
14.     {
15.         delay(20); //延时去抖动
16.         if(digitalRead(KEY) == LOW) //有按键按下
17.         {
18.
19.             digitalWrite(LED,!digitalRead(LED)); //LED 的状态翻转
20.             while(digitalRead(KEY) == LOW); //等待按键松手
21.         }
22.     }
23. }
```

Arduino 按键控制有源蜂鸣器

[plain] view plaincopy C

```
1. #define LED 13
2. #define KEY 2
3. #define Buzzer 3
4. int KEY_NUM = 0;           //按键键值变量
5.
6. void setup()
7. {
8.     pinMode(LED,OUTPUT);      //LED 为 IO 输出
9.     pinMode(KEY,INPUT_PULLUP); //按键为 IO 带上拉输入
10.    pinMode(Buzzer,OUTPUT);   //蜂鸣器为 IO 输出
11.    digitalWrite(Buzzer,LOW); //蜂鸣器初始为不鸣叫
12. }
13.
14. void loop()
15. {
16.     ScanKey();              //按键扫描
17.     if(KEY_NUM == 1)         //当有按键按下时
18.     {
19.         digitalWrite(LED,!digitalRead(LED)); //LED 状态翻转
20.     }
21. }
22.
23. void ScanKey()
24. {
25.     KEY_NUM = 0;
26.     if(digitalRead(KEY) == LOW)
27.     {
28.         delay(20);           //延时去抖动
29.         if(digitalRead(KEY) == LOW)
30.         {
31.             BuzzerDi();        //滴一声
32.             KEY_NUM = 1;       //设置键值
33.             while(digitalRead(KEY) == LOW); //松手检测
34.         }
35.     }
36. }
37.
38. void BuzzerDi()
39. {
40.     digitalWrite(Buzzer,HIGH); //蜂鸣器响
41.     delay(100);              //延时 20ms
42.     digitalWrite(Buzzer,LOW); //蜂鸣器关闭
43. }
```



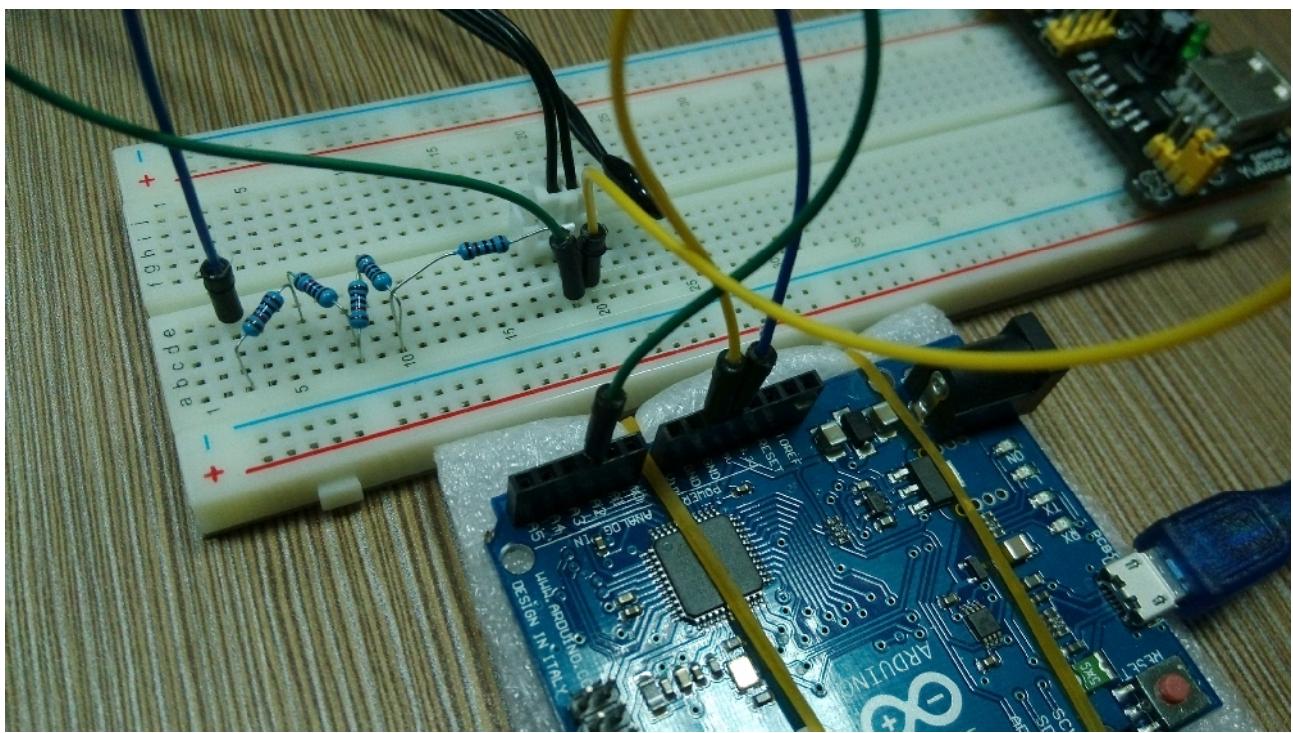
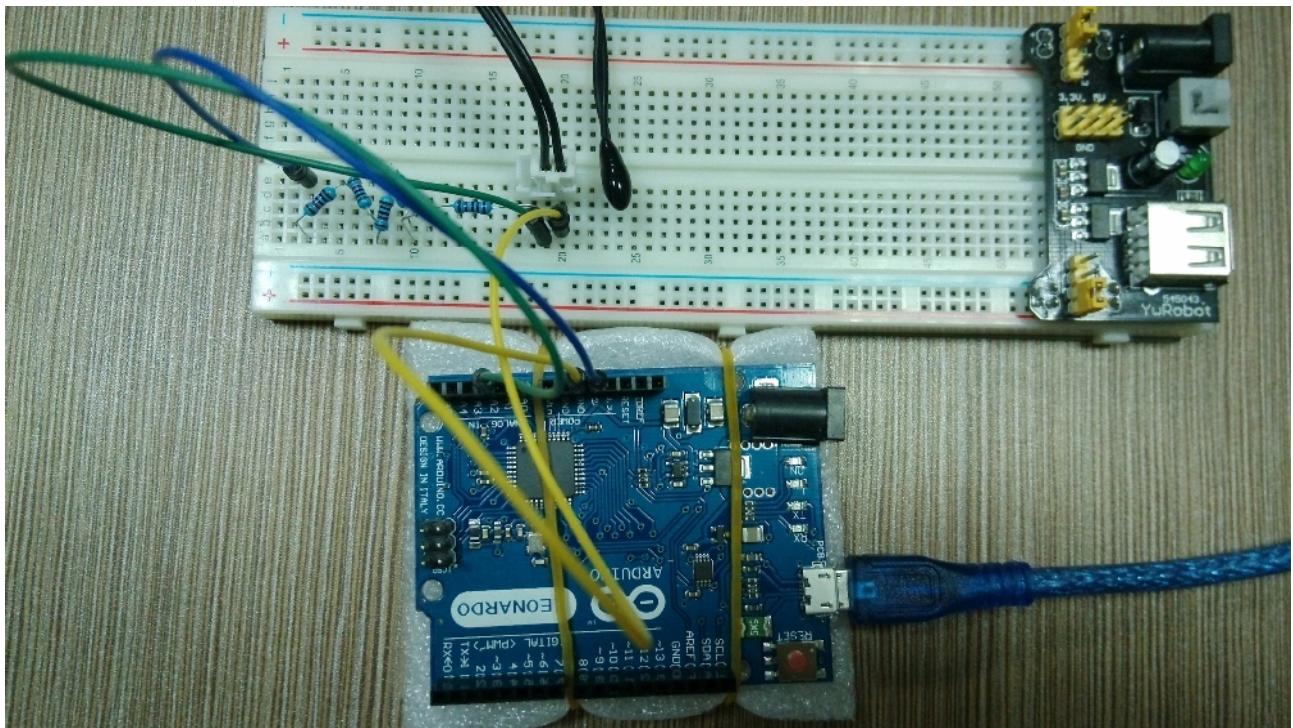
PS: 发现用移动电源供电很方便。只剩下焊在板上的蜂鸣器了，取下来的话针脚就太短了，焊了两条线，先凑合用用，下次买元件的时候再更新，并计划做一个蜂鸣器检验的治具。

Arduino 负温度系数热敏电阻 (NTC) 测温

一直都用 NTC 作为温度传感器来测温，采用 Arduino 没有现成的例子用 NTC 测温的，LM35D 温度传感器，这款传感器能够测量 0-100 摄氏度的温度，并以电压的数值输出。从 0 度开始温度每升高 1 度输出电压就会提高 10mv。而 NTC 则不然，NTC 根据温度变化产生电阻阻值变化，而且是非线性的变化，这就需要用上拉电阻或下拉电阻来选择分辨率较好的区间。

[plain] view plaincopy 

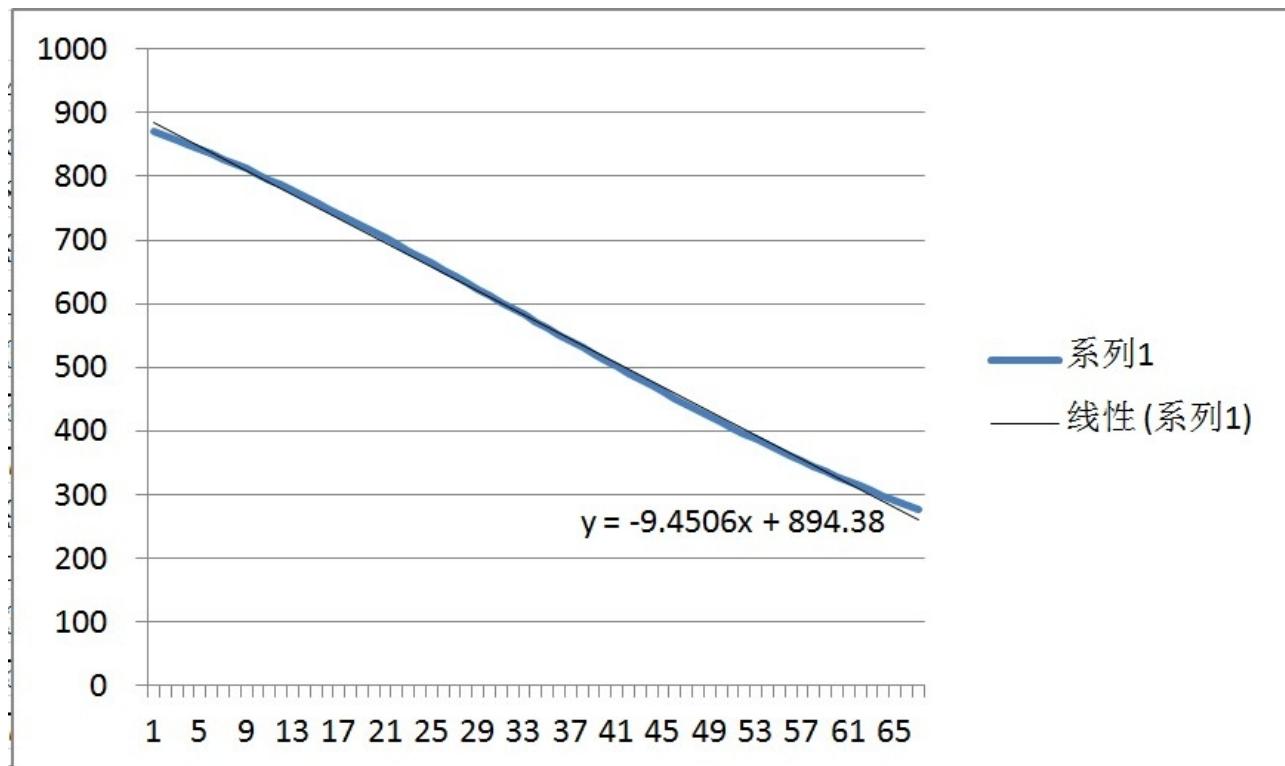
```
1. #define Pot A2          //电位器引脚命名
2.
3. int PotBuffer = 0;      //AD 读取数据缓存变量
4.
5. void setup()
6. {
7.     Serial.begin(9600);  //初始化串口波特率为 9600
8. }
9.
10. void loop()
11. {
12.     PotBuffer = analogRead(Pot);           //读取 AD 值
13.     float temp = -0.10581*PotBuffer+94.6374; //浮点运算，根据拟合的温度与电阻曲线的斜率换算反函数
14.     Serial.print("TEMP = ");              //串口输出“Pot = ”
15.     Serial.print(temp);                  //串口输出 temp 的值
16.     Serial.print("; AD = ");
17.     Serial.println(PotBuffer);
18.     delay(500);                      //延时 500ms
19. }
```



NTC 的 $R_{25}=100K$, $B=3950$, 选用 $50K$ 作为上拉电阻, 根据温度与阻值的真值表, 求出待测范围如 $0-60^{\circ}C$ 的 NTC 实际电压值, $0-5V$ 等分 1024 份, 求出电压值对应的 AD 值, 以温度为 X 轴, AD 值为

Y 轴做出曲线，并拟合成线性函数，求出此线性函数的反函数，最终得到【temp = -0.10581*PotBuffer+94.6374】。

3	51k 电阻， 100k NTC。 对应表			48	
4					
5	870	4. 25457	0	263.283	268.583
6	864	4. 221931	1	250.554	255.47
7	857	4. 188356	2	238.516	243.074
8	850	4. 153842	3	227.127	231.354
9	842	4. 118402	4	216.349	220.267
10	835	4. 082042	5	206.146	209.776
11	827	4. 044856	6	196.505	199.869
12	820	4. 006594	7	187.33	190.445
13	812	3. 967533	8	178.657	181.541
14	803	3. 9276	9	170.435	173.104
15	795	3. 886817	10	162.64	165.108
16	786	3. 845199	11	155.246	157.528
17	778	3. 802778	12	148.231	150.34
18	769	3. 759562	13	141.573	143.521
19	760	3. 715595	14	135.252	137.05
20	751	3. 670898	15	129.25	130.908
21	742	3. 6255	16	123.548	125.076
22	732	3. 579445	17	118.13	119.537



P	Q	R	S
反函数			
K	B	k	b
-9.4506	894.38	-0.10581	-94.6374
		-0.10581	94.6374
验证	AD	TE	
	664	24.37956	

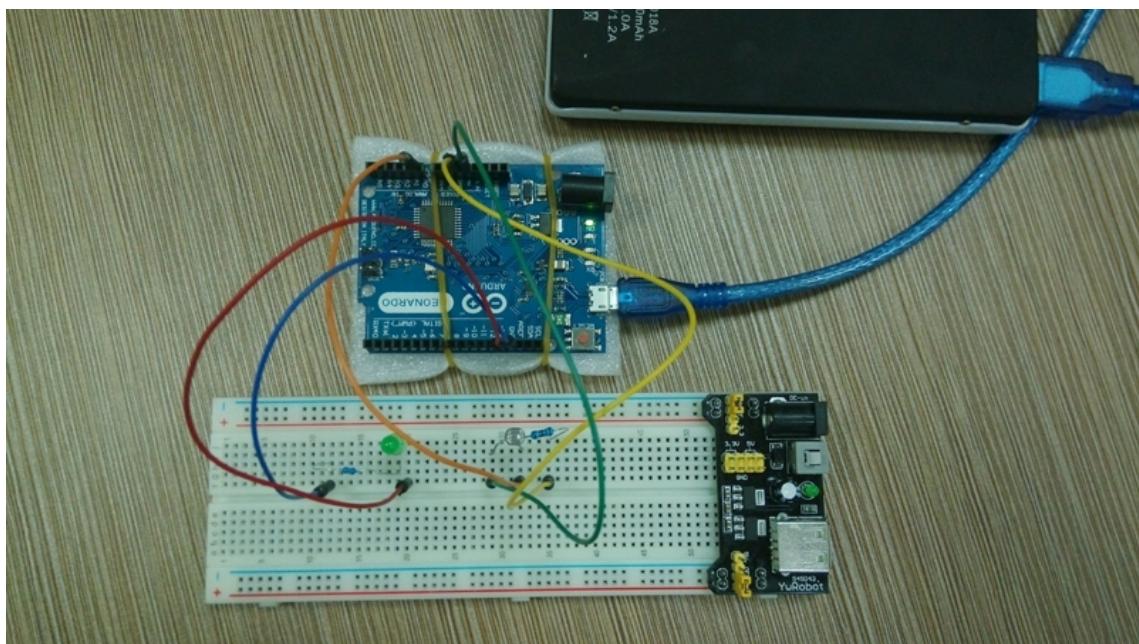
Arduino 光敏电阻调节呼吸灯

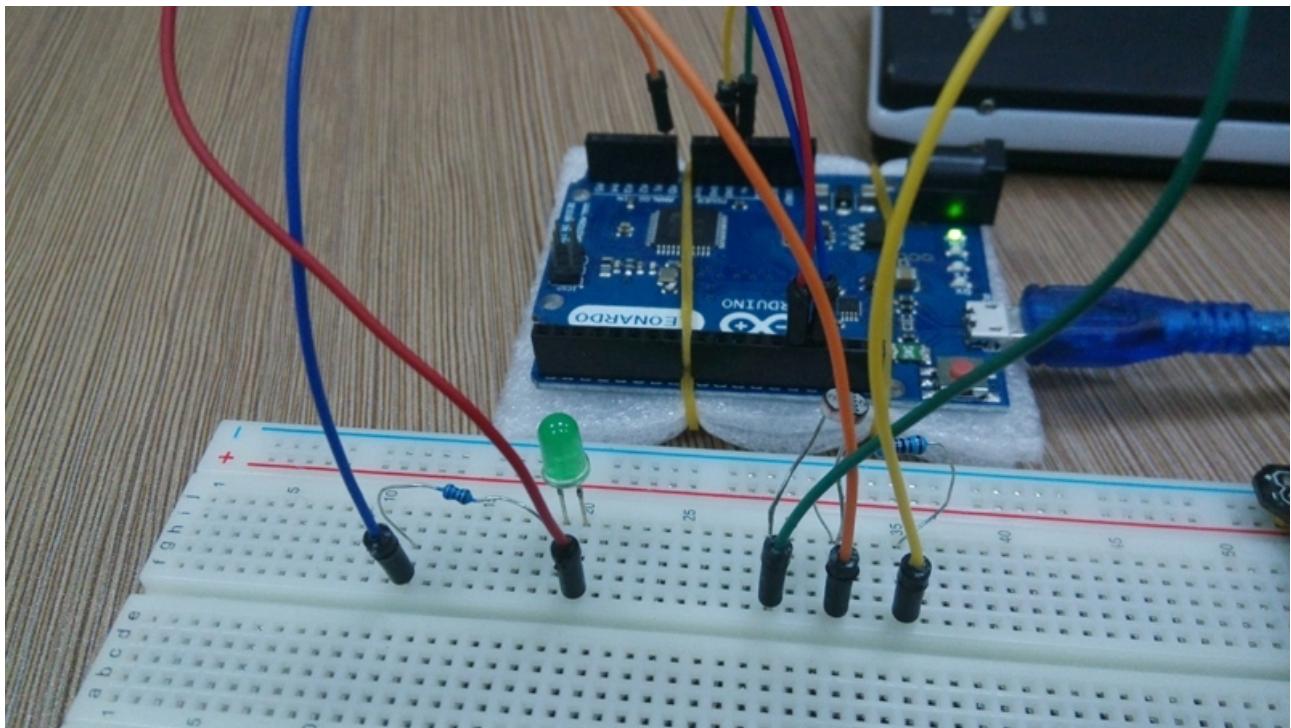
今天测试 PWM 做呼吸灯，以及用光敏电阻做达文西呼吸灯，并输出串口光敏 AD 值，可作为智能家居中，卧室慢慢亮起的地灯，或根据室内光线调节光亮等效果。

呼吸灯：

[plain] view plaincopy C

```
1. void setup ()  
2. {  
3.     pinMode(13,OUTPUT);  
4. }  
5.  
6. void loop()  
7. {  
8.     for (int a=1; a<=100;a++)          //循环语句，控制 PWM 亮度的增加  
9.     {  
10.        analogWrite(13,a);  
11.        delay(20);                  //当前亮度级别维持的时间,单位毫秒  
12.    }  
13.    for (int a=100; a>=1;a--)      //循环语句，控制 PWM 亮度减小  
14.    {  
15.        analogWrite(13,a);  
16.        delay(20);                  //当前亮度的维持的时间,单位毫秒  
17.    }  
18.    delay(500);                  //完成一个循环后等待的时间,单位毫秒  
19. }
```



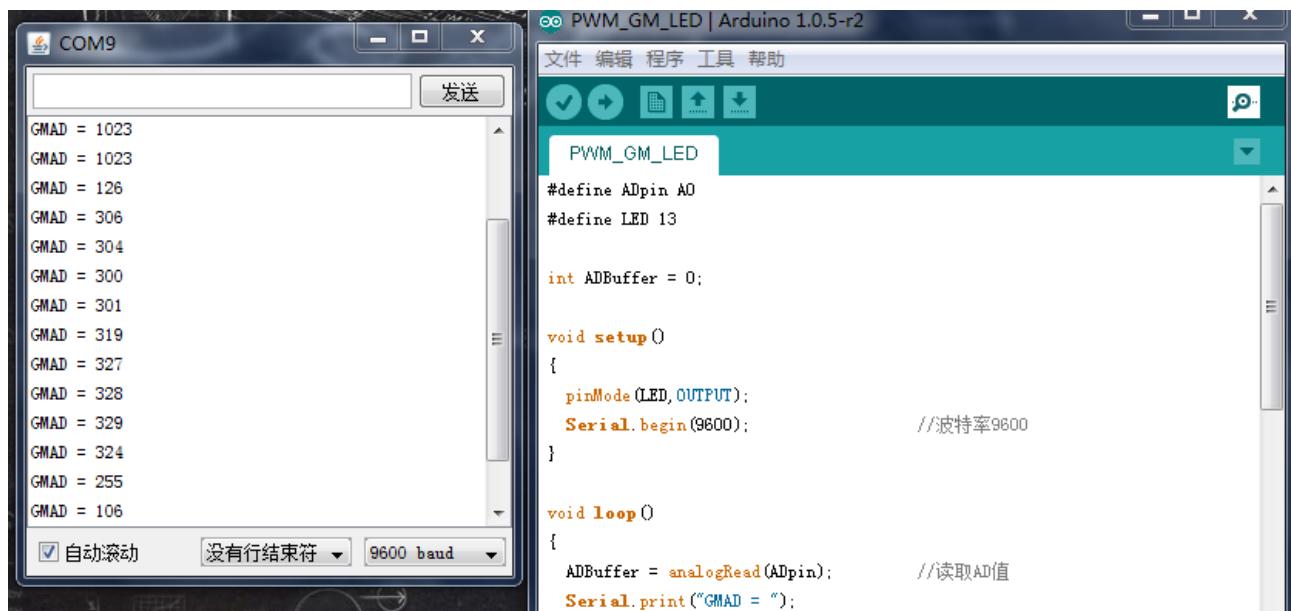


左边是 LED 呼吸灯，右边是光敏电阻。

[plain] view plaincopy

```
1. #define ADpin A0
2. #define LED 13
3.
4. int ADBuffer = 0;
5.
6. void setup()
7. {
8.     pinMode(LED,OUTPUT);
9.     Serial.begin(9600);          //波特率 9600
10. }
11.
12. void loop()
13. {
14.     ADBuffer = analogRead(ADpin);    //读取 AD 值
15.     Serial.print("GMAD = ");
16.     Serial.println(ADBuffer);
17.     if(ADBuffer < 180)            //ADBuffer 值大于设定值，相当于光照强度小于设定值
18.     {
19.         for (int a=1; a<=100;a++)   //循环语句，控制 PWM 亮度的增加
20.         {
21.             analogWrite(LED,a);
22.             delay(20);           //当前亮度级别维持的时间，单位毫秒
```

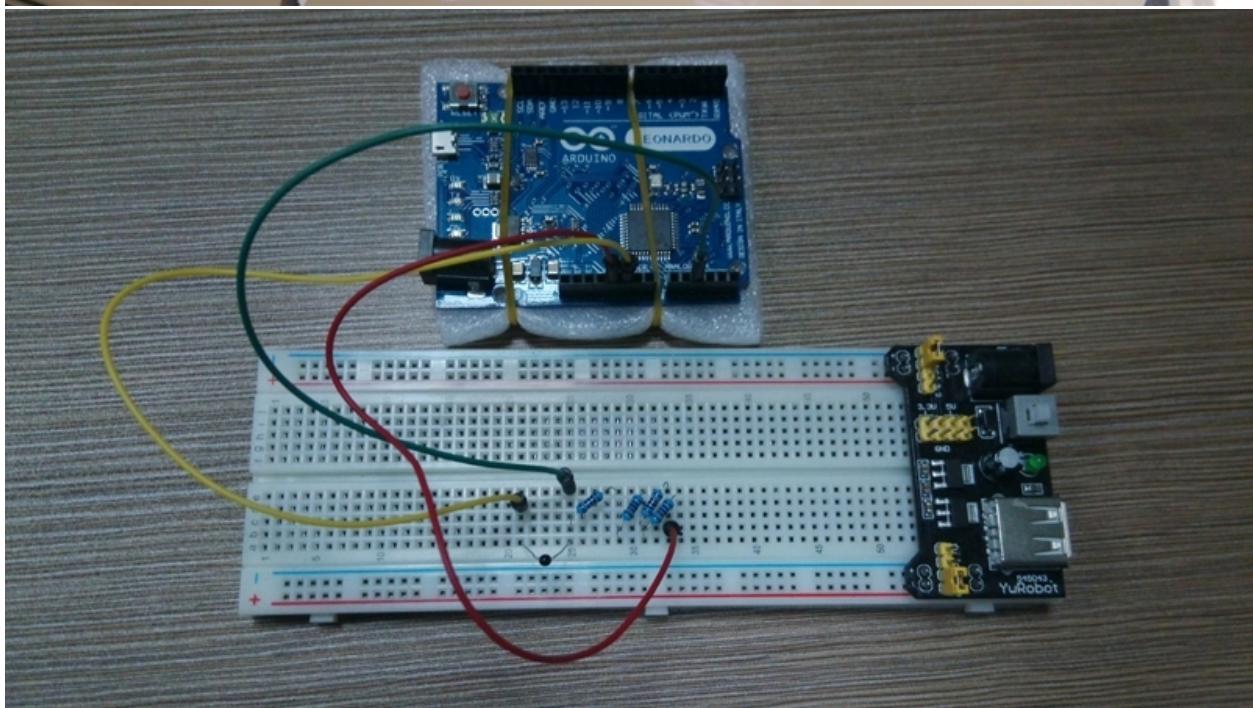
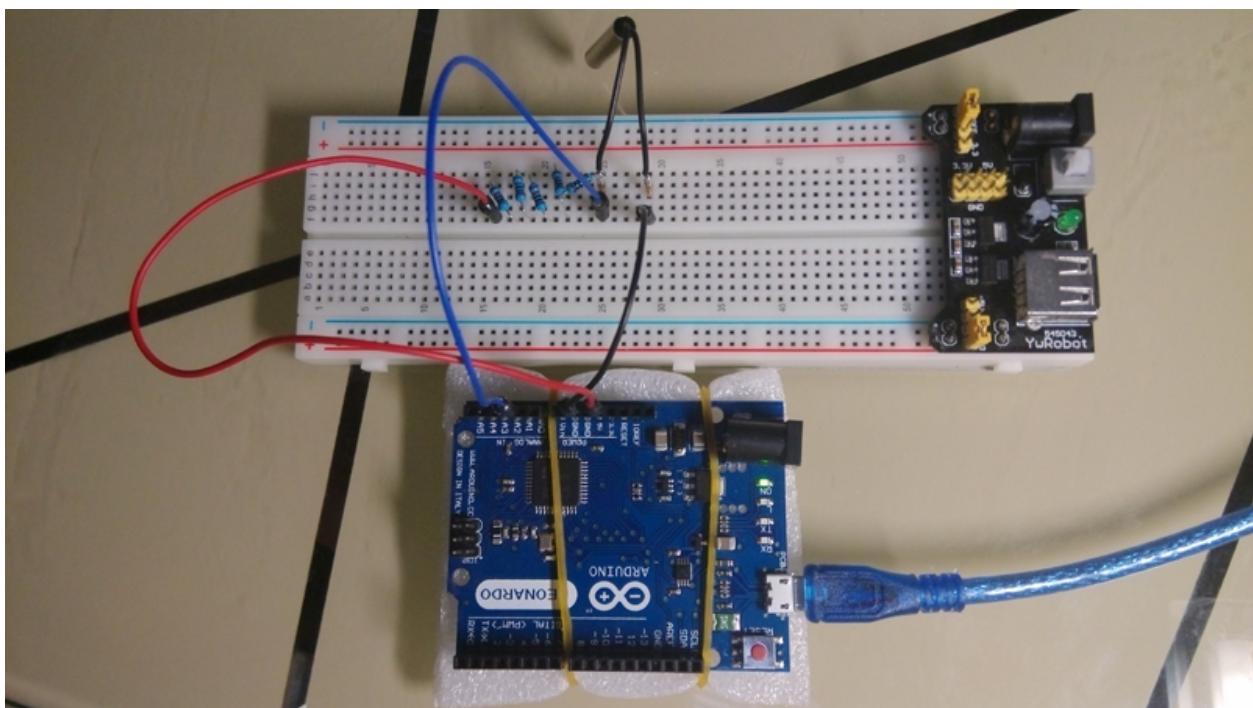
```
23.     }
24.     for (int a=100; a>=1;a--)
25.     {
26.         analogWrite(13,a);
27.         delay(20);           //当前亮度的维持的时间,单位毫秒
28.     }
29.     delay(500);
30. }
31. else
32. {
33.     digitalWrite(LED,LOW); //关闭 LED
34. }
35. delay(500);           //延时 500ms
36. }
```

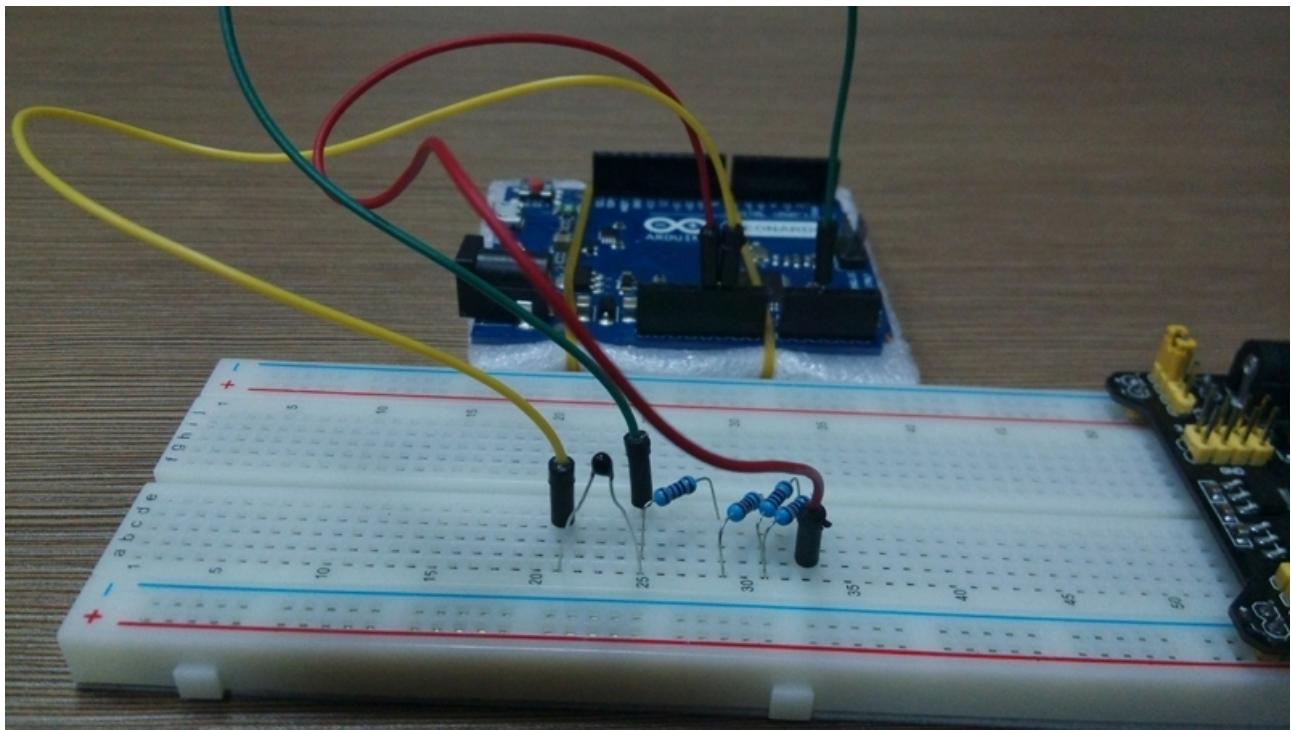


Arduino 温度传感器 NTC 温度 AD 对应值映射改进

上次做的 NTC 测温方案，功能是实现了，但是有些不足的地方，例如 AD 值飘得厉害，温度值也就飘，尝试调整线性函数和映射 AD 值，从 2^{10} 位降到 8 位、7 位测试其稳定性。

NTC 选型，除了上次用的环氧树脂型的，这次选用不锈钢或铜镀锌外壳，和小黑头插件这两种做测试。





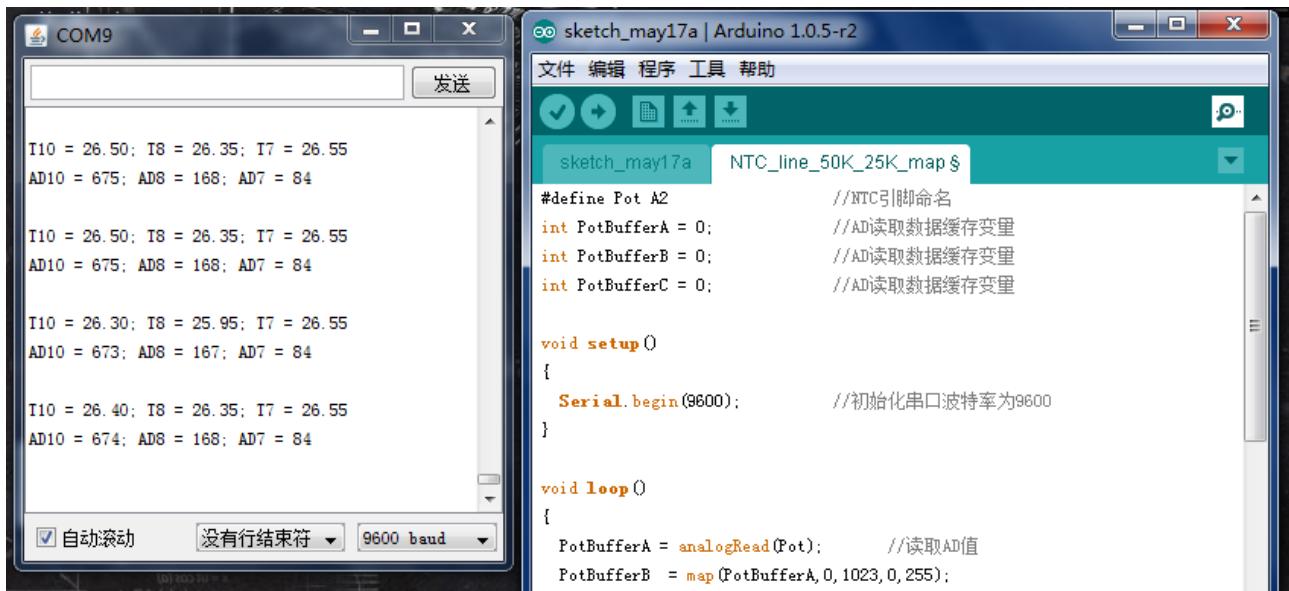
[plain] view plaincopy

```
1. #define Pot A2          //NTC 引脚命名
2. int PotBufferA = 0;    //AD 读取数据缓存变量
3. int PotBufferB = 0;    //AD 读取数据缓存变量
4. int PotBufferC = 0;    //AD 读取数据缓存变量
5.
6. void setup()
7. {
8.     Serial.begin(9600);      //初始化串口波特率为 9600
9. }
10.
11. void loop()
12. {
13.     PotBufferA = analogRead(Pot);      //读取 AD 值
14.     PotBufferB = map(PotBufferA,0,1023,0,255);
15.     PotBufferC = map(PotBufferA,0,1023,0,128);
16.     float ta = 0.1*PotBufferA-41;
17.     //浮点运算，根据拟合的温度与电阻曲线的斜率换算反函数
18.     float tb = 0.4*PotBufferB-40.85;
19.     float tc = 0.8*PotBufferC-40.65;
20.     Serial.print("T10 = ");           //串口输出“T10 = ”
21.     Serial.print(ta);               //串口输出 ta 的值
22.     Serial.print("; T8 = ");        //串口输出“T8 = ”
23.     Serial.print(tb);               //串口输出 tb 的值
24.     Serial.print("; T7 = ");        //串口输出“T7 = ”
25.     Serial.println(tc);             //串口输出 tc 的值
```

```

26. Serial.print("AD10 = ");           //串口输出“AD10 = ”
27. Serial.print(PotBufferA);          //串口输出 PotBufferA 的值
28. Serial.print("; AD8 = ");         //串口输出“AD8 = ”
29. Serial.print(PotBufferB);          //串口输出 PotBufferB 的值
30. Serial.print("; AD7 = ");         //串口输出“AD7 = ”
31. Serial.println(PotBufferC);        //串口输出 PotBufferC 的值
32. Serial.println();
33. delay(5000);                   //延时 500ms
34. }

```



map(value, fromLow, fromHigh, toLow, toHigh)

value:需要映射的值

fromLow:当前范围值的下限

fromHigh:当前范围值的上限

toLow:目标范围值的下限

toHigh:目标范围值的上限

将一个数从一个范围映射到另外一个范围。也就是说，会将 **fromLow** 到 **fromHigh** 之间的值映射到 **toLow** 在 **toHigh** 之间的值。

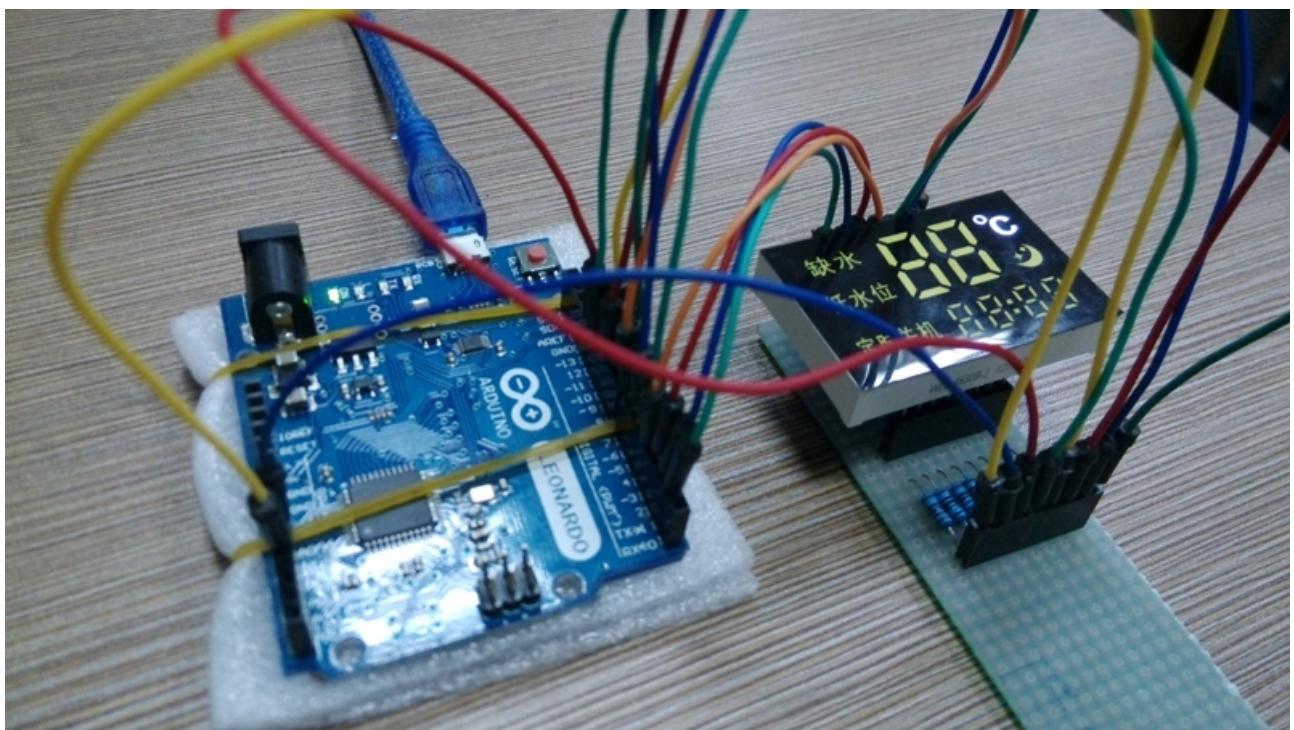
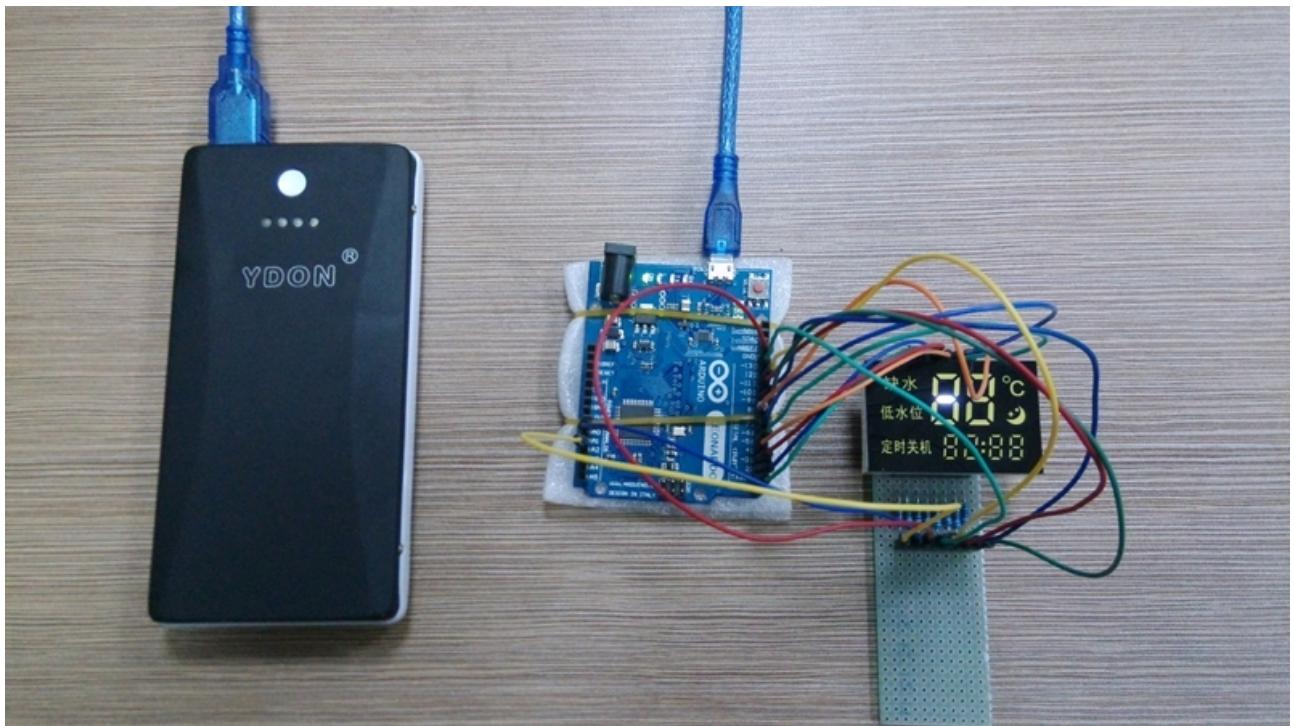
不限制值的范围，因为范围外的值有时是刻意的和有用的。如果需要限制的范围，**constrain()** 函数可以用于此函数之前或之后。

map() 函数使用整型数进行运算因此不会产生分数，这时运算应该表明它需要这样做。小数的余数部分会被舍去，不会四舍五入或者平均。

Arduino 数码管 LED 屏驱动

今天测试数码管 LED 屏驱动，用某产品的一个共阴极的 LED 屏，根据电路图做数码管 LED 屏的检测。

代码写得有些冗长，有好几种驱动的方法，这里只是其中一种最直接的方案，抽空要做个更有效率的调用和驱动的方案。



[plain] view plaincopy 

```
1. //设置阴极接口
2. int d1 = 1;
3. int d2 = 2;
4. int d3 = 3;
5. int d4 = 4;
6. int d5 = 5;
7. int d6 = 6;
8. int d7 = 7;
9. //设置阳极接口
10. int a = 8;
11. int b = 9;
12. int c = 10;
13. int d = 11;
14. int e = 12;
15. int f = 13;
16. int g = A0;
17. int h = A1;
18.
19. void setup()
20. {
21. pinMode(d1, OUTPUT);
22. pinMode(d2, OUTPUT);
23. pinMode(d3, OUTPUT);
24. pinMode(d4, OUTPUT);
25. pinMode(d5, OUTPUT);
26. pinMode(d6, OUTPUT);
27. pinMode(d7, OUTPUT);
28. pinMode(a, OUTPUT);
29. pinMode(b, OUTPUT);
30. pinMode(c, OUTPUT);
31. pinMode(d, OUTPUT);
32. pinMode(e, OUTPUT);
33. pinMode(f, OUTPUT);
34. pinMode(g, OUTPUT);
35. pinMode(h, OUTPUT);
36. digitalWrite(a, LOW);
37. digitalWrite(b, LOW);
38. digitalWrite(c, LOW);
39. digitalWrite(d, LOW);
40. digitalWrite(e, LOW);
41. digitalWrite(f, LOW);
42. digitalWrite(g, LOW);
43. digitalWrite(h, LOW);
44. digitalWrite(d1, HIGH);
45. digitalWrite(d2, HIGH);
```

```
46. digitalWrite(d3, HIGH);
47. digitalWrite(d4, HIGH);
48. digitalWrite(d5, HIGH);
49. digitalWrite(d6, HIGH);
50. digitalWrite(d7, HIGH);
51. }
52.
53. void loop()
54. {
55. //缺水
56. digitalWrite(d7, LOW);
57. digitalWrite(a, HIGH);
58. digitalWrite(b, HIGH);
59. delay(500);
60. //低水位
61. digitalWrite(a, LOW);
62. digitalWrite(b, LOW);
63. digitalWrite(c, HIGH);
64. digitalWrite(d, HIGH);
65. delay(500);
66. //定时关机
67. digitalWrite(c, LOW);
68. digitalWrite(d, LOW);
69. digitalWrite(e, HIGH);
70. digitalWrite(f, HIGH);
71. digitalWrite(g, HIGH);
72. delay(500);
73. //温度十位
74. digitalWrite(d7, HIGH);
75. digitalWrite(d1, LOW);
76. digitalWrite(e, LOW);
77. digitalWrite(f, LOW);
78. digitalWrite(g, LOW);
79. digitalWrite(a, HIGH);
80. delay(300);
81. digitalWrite(a, LOW);
82. digitalWrite(b, HIGH);
83. delay(300);
84. digitalWrite(b, LOW);
85. digitalWrite(c, HIGH);
86. delay(300);
87. digitalWrite(c, LOW);
88. digitalWrite(d, HIGH);
89. delay(300);
90. digitalWrite(d, LOW);
91. digitalWrite(e, HIGH);
92. delay(300);
93. digitalWrite(e, LOW);
94. digitalWrite(f, HIGH);
95. delay(300);
96. digitalWrite(f, LOW);
97. digitalWrite(g, HIGH);
98. delay(300);
99. digitalWrite(a, HIGH);
100. digitalWrite(b, HIGH);
101. digitalWrite(c, HIGH);
102. digitalWrite(d, HIGH);
103. digitalWrite(e, HIGH);
104. digitalWrite(f, HIGH);
105. delay(300);
106. //温度个位
107. digitalWrite(d1, HIGH);
108. digitalWrite(d2, LOW);
109. digitalWrite(b, LOW);
110. digitalWrite(c, LOW);
111. digitalWrite(d, LOW);
112. digitalWrite(e, LOW);
113. digitalWrite(f, LOW);
114. digitalWrite(g, LOW);
115. digitalWrite(a, HIGH);
116. delay(300);
117. digitalWrite(a, LOW);
118. digitalWrite(b, HIGH);
119. delay(300);
120. digitalWrite(b, LOW);
121. digitalWrite(c, HIGH);
122. delay(300);
123. digitalWrite(c, LOW);
124. digitalWrite(d, HIGH);
125. delay(300);
126. digitalWrite(d, LOW);
127. digitalWrite(e, HIGH);
128. delay(300);
129. digitalWrite(e, LOW);
130. digitalWrite(f, HIGH);
131. delay(300);
132. digitalWrite(f, LOW);
133. digitalWrite(g, HIGH);
134. delay(300);
135. digitalWrite(a, HIGH);
136. digitalWrite(b, HIGH);
137. digitalWrite(c, HIGH);
```

```
138.digitalWrite(d, HIGH);
139.digitalWrite(e, HIGH);
140.digitalWrite(f, HIGH);
141.delay(300);
142.//温符号度
143.digitalWrite(d2, HIGH);
144.digitalWrite(a, LOW);
145.digitalWrite(b, LOW);
146.digitalWrite(c, LOW);
147.digitalWrite(d, LOW);
148.digitalWrite(e, LOW);
149.digitalWrite(f, LOW);
150.digitalWrite(g, LOW);
151.digitalWrite(h, HIGH);
152.digitalWrite(d1, LOW);
153.delay(500);
154.//睡眠符号
155.digitalWrite(d1, HIGH);
156.digitalWrite(d2, LOW);
157.delay(500);
158.//小时十位 8
159.digitalWrite(d2, HIGH);
160.digitalWrite(d3, LOW);
161.digitalWrite(b, LOW);
162.digitalWrite(c, LOW);
163.digitalWrite(d, LOW);
164.digitalWrite(e, LOW);
165.digitalWrite(f, LOW);
166.digitalWrite(g, LOW);
167.digitalWrite(a, HIGH);
168.delay(300);
169.digitalWrite(a, LOW);
170.digitalWrite(b, HIGH);
171.delay(300);
172.digitalWrite(b, LOW);
173.digitalWrite(c, HIGH);
174.delay(300);
175.digitalWrite(c, LOW);
176.digitalWrite(d, HIGH);
177.delay(300);
178.digitalWrite(d, LOW);
179.digitalWrite(e, HIGH);
180.delay(300);
181.digitalWrite(e, LOW);
182.digitalWrite(f, HIGH);
183.delay(300);

184.digitalWrite(f, LOW);
185.digitalWrite(g, HIGH);
186.delay(300);
187.digitalWrite(a, HIGH);
188.digitalWrite(b, HIGH);
189.digitalWrite(c, HIGH);
190.digitalWrite(d, HIGH);
191.digitalWrite(e, HIGH);
192.digitalWrite(f, HIGH);
193.delay(300);
194.//小时个位 8
195.digitalWrite(d3, HIGH);
196.digitalWrite(d4, LOW);
197.digitalWrite(b, LOW);
198.digitalWrite(c, LOW);
199.digitalWrite(d, LOW);
200.digitalWrite(e, LOW);
201.digitalWrite(f, LOW);
202.digitalWrite(g, LOW);
203.digitalWrite(h, LOW);
204.digitalWrite(a, HIGH);
205.delay(300);
206.digitalWrite(a, LOW);
207.digitalWrite(b, HIGH);
208.delay(300);
209.digitalWrite(b, LOW);
210.digitalWrite(c, HIGH);
211.delay(300);
212.digitalWrite(c, LOW);
213.digitalWrite(d, HIGH);
214.delay(300);
215.digitalWrite(d, LOW);
216.digitalWrite(e, HIGH);
217.delay(300);
218.digitalWrite(e, LOW);
219.digitalWrite(f, HIGH);
220.delay(300);
221.digitalWrite(f, LOW);
222.digitalWrite(g, HIGH);
223.delay(300);
224.digitalWrite(a, HIGH);
225.digitalWrite(b, HIGH);
226.digitalWrite(c, HIGH);
227.digitalWrite(d, HIGH);
228.digitalWrite(e, HIGH);
229.digitalWrite(f, HIGH);

230.delay(300);
231.//冒号
232.digitalWrite(a, LOW);
233.digitalWrite(b, LOW);
234.digitalWrite(c, LOW);
235.digitalWrite(d, LOW);
236.digitalWrite(e, LOW);
237.digitalWrite(f, LOW);
238.digitalWrite(g, LOW);
239.digitalWrite(h, HIGH);
240.delay(500);
241.//分钟十位 8
242.digitalWrite(d4, HIGH);
243.digitalWrite(d5, LOW);
244.digitalWrite(b, LOW);
245.digitalWrite(c, LOW);
246.digitalWrite(d, LOW);
247.digitalWrite(e, LOW);
248.digitalWrite(f, LOW);
249.digitalWrite(g, LOW);
250.digitalWrite(a, HIGH);
251.delay(300);
252.digitalWrite(a, LOW);
253.digitalWrite(b, HIGH);
254.delay(300);
255.digitalWrite(b, LOW);
256.digitalWrite(c, HIGH);
257.delay(300);
258.digitalWrite(c, LOW);
259.digitalWrite(d, HIGH);
260.delay(300);
261.digitalWrite(d, LOW);
262.digitalWrite(e, HIGH);
263.delay(300);
264.digitalWrite(e, LOW);
265.digitalWrite(f, HIGH);
266.delay(300);
267.digitalWrite(f, LOW);
268.digitalWrite(g, HIGH);
269.delay(300);
270.digitalWrite(a, HIGH);
271.digitalWrite(b, HIGH);
272.digitalWrite(c, HIGH);
273.digitalWrite(d, HIGH);
274.digitalWrite(e, HIGH);
275.digitalWrite(f, HIGH);
```

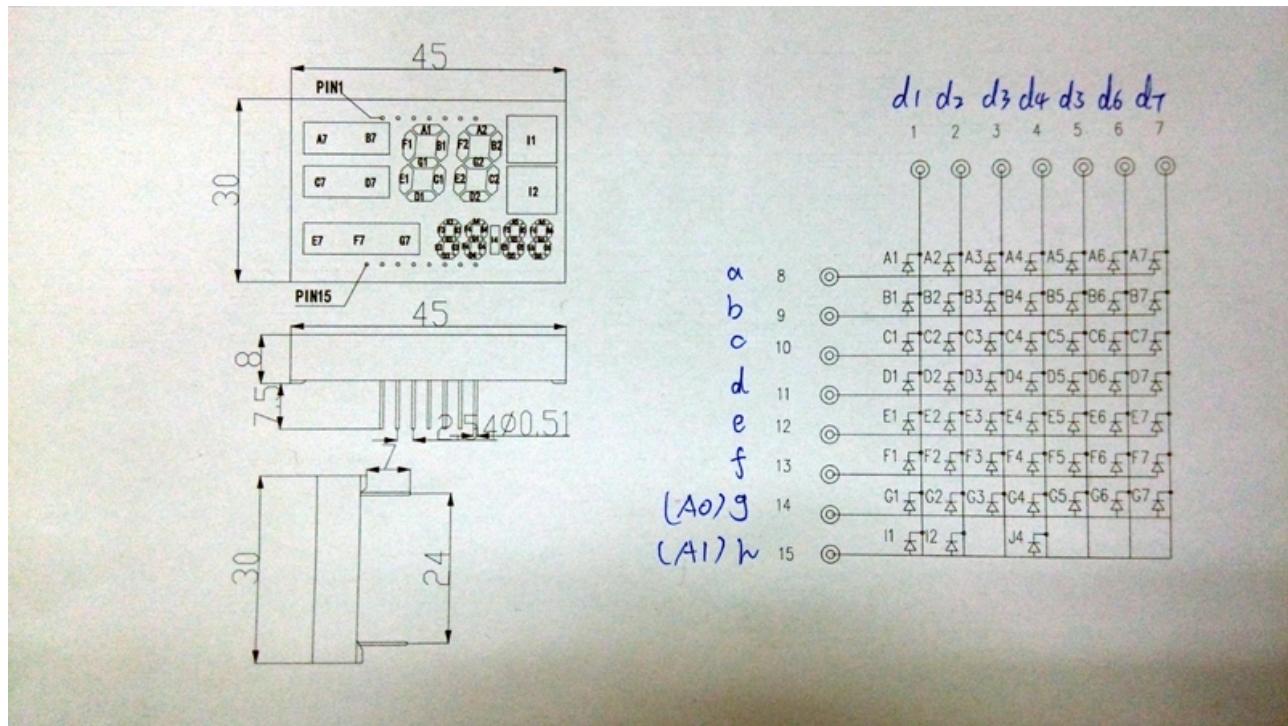
```

276.delay(300);
277.//分钟个位 8
278.digitalWrite(d5, HIGH);
279.digitalWrite(d6, LOW);
280.digitalWrite(b, LOW);
281.digitalWrite(c, LOW);
282.digitalWrite(d, LOW);
283.digitalWrite(e, LOW);
284.digitalWrite(f, LOW);
285.digitalWrite(g, LOW);
286.digitalWrite(a, HIGH);
287.delay(300);
288.digitalWrite(a, LOW);
289.digitalWrite(b, HIGH);
290.delay(300);
291.digitalWrite(b, LOW);
292.digitalWrite(c, HIGH);
293.delay(300);
294.digitalWrite(c, LOW);
295.digitalWrite(d, HIGH);
296.delay(300);
297.digitalWrite(d, LOW);
298.digitalWrite(e, HIGH);
299.delay(300);

300.digitalWrite(e, LOW);
301.digitalWrite(f, HIGH);
302.delay(300);
303.digitalWrite(f, LOW);
304.digitalWrite(g, HIGH);
305.delay(300);
306.digitalWrite(a, HIGH);
307.digitalWrite(b, HIGH);
308.digitalWrite(c, HIGH);
309.digitalWrite(d, HIGH);
310.digitalWrite(e, HIGH);
311.digitalWrite(f, HIGH);
312.delay(300);
313.//全亮
314.digitalWrite(a, HIGH);
315.digitalWrite(b, HIGH);
316.digitalWrite(c, HIGH);
317.digitalWrite(d, HIGH);
318.digitalWrite(e, HIGH);
319.digitalWrite(f, HIGH);
320.digitalWrite(g, HIGH);
321.digitalWrite(h, HIGH);
322.digitalWrite(d1, LOW);
323.digitalWrite(d2, LOW);

324.digitalWrite(d3, LOW);
325.digitalWrite(d4, LOW);
326.digitalWrite(d5, LOW);
327.digitalWrite(d6, LOW);
328.digitalWrite(d7, LOW);
329.delay(500);
330.//全灭
331.digitalWrite(a, LOW);
332.digitalWrite(b, LOW);
333.digitalWrite(c, LOW);
334.digitalWrite(d, LOW);
335.digitalWrite(e, LOW);
336.digitalWrite(f, LOW);
337.digitalWrite(g, LOW);
338.digitalWrite(h, LOW);
339.digitalWrite(d1, HIGH);
340.digitalWrite(d2, HIGH);
341.digitalWrite(d3, HIGH);
342.digitalWrite(d4, HIGH);
343.digitalWrite(d5, HIGH);
344.digitalWrite(d6, HIGH);
345.digitalWrite(d7, HIGH);
346.delay(500);
347.}

```



计划采用机器视觉来检测，自动鉴别出断笔，笔段亮度差异较大等质量检验问题。

Arduino 数码管 LED 驱动 数组法

上个例子讲到驱动 LED 数码管，采用一种最直接的方案，对每个 LED 进行高低电平的控制，这样的优点是每个 LED 都是受控可检的，避免了由于短路造成的假象，但对于数字变化来说，写起来就非常冗余，因此这次尝试用数组的方法实现。

```
[plain] view plaincopy
1. //设置阴极接口
2. int d1 = 1;
3. int d2 = 2;
4. int d3 = 3;
5. int d4 = 4;
6. int d5 = 5;
7. int d6 = 6;
8. int d7 = 7;
9. //设置阳极接口
10. int a = 8;
11. int b = 9;
12. int c = 10;
13. int d = 11;
14. int e = 12;
15. int f = 13;
16. int g = A0;
17. int h = A1;
18.
19. byte gyang[8] = { a, b, c, d, e, f, g, h };
20. byte gyin[7] = { d1, d2, d3, d4, d5, d6, d7 };
21.
22. byte gong_yang[14][8] = {
23. { 1,1,0,0,0,0,0,0 }, // 0 = 缺水
24. { 0,0,1,1,0,0,0,0 }, // 1 = 低水位
25. { 0,0,0,0,1,1,1,0 }, // 2 = 定时关机
26. { 1,1,1,1,1,1,0,0 }, // 3 = 0
27. { 0,1,1,0,0,0,0,0 }, // 4 = 1
28. { 1,1,0,1,1,0,1,0 }, // 5 = 2
29. { 1,1,1,1,0,0,1,0 }, // 6 = 3
30. { 0,1,1,0,0,1,1,0 }, // 7 = 4
31. { 1,0,1,1,0,1,1,0 }, // 8 = 5
32. { 1,0,1,1,1,1,1,0 }, // 9 = 6
33. { 1,1,1,0,0,0,0,0 }, // 10 = 7
34. { 1,1,1,1,1,1,1,0 }, // 11 = 8
35. { 1,1,1,1,0,1,1,0 }, // 12 = 9
36. { 0,0,0,0,0,0,0,1 }, // 13 = 摄氏度符号、睡眠符号
37. };
38. byte gong_yin[5][7] = {
39. { 1,1,1,1,1,1,0 }, // 0 = 缺水、低水位、定时关机
40. { 0,1,1,1,1,1,1 }, // 1 = 温度十位、摄氏度符号
41. { 1,0,1,1,1,1,1 }, // 2 = 温度个位、睡眠符号
42. { 0,0,1,1,1,1,1 }, // 3 = 温度个和十位
43. { 1,1,0,0,0,0,1 }, // 4 = 时钟个、十位和两点
44. };
45.
46. void setup()
47. {
48. pinMode(d1, OUTPUT);
49. pinMode(d2, OUTPUT);
50. pinMode(d3, OUTPUT);
51. pinMode(d4, OUTPUT);
52. pinMode(d5, OUTPUT);
53. pinMode(d6, OUTPUT);
54. pinMode(d7, OUTPUT);
55. pinMode(a, OUTPUT);
56. pinMode(b, OUTPUT);
57. pinMode(c, OUTPUT);
58. pinMode(d, OUTPUT);
59. pinMode(e, OUTPUT);
60. pinMode(f, OUTPUT);
61. pinMode(g, OUTPUT);
62. pinMode(h, OUTPUT);
63. digitalWrite(a, LOW);
64. digitalWrite(b, LOW);
65. digitalWrite(c, LOW);
66. digitalWrite(d, LOW);
67. digitalWrite(e, LOW);
68. digitalWrite(f, LOW);
69. digitalWrite(g, LOW);
70. digitalWrite(h, LOW);
71. digitalWrite(d1, HIGH);
72. digitalWrite(d2, HIGH);
73. digitalWrite(d3, HIGH);
74. digitalWrite(d4, HIGH);
75. digitalWrite(d5, HIGH);
```

```
76. digitalWrite(d6, HIGH);
77. digitalWrite(d7, HIGH);
78. }
79.
80. void loop()
81. {
82.     GongYang(0);
83.     GongYin(0);
84.     delay(300);
85.     GongYang(1);
86.     delay(300);
87.     GongYang(2);
88.     delay(300);
89.     GongYin(3);
90.     GongYang(3);
91.     delay(300);
92.     GongYang(4);
93.     delay(300);
94.     GongYang(5);
95.     delay(300);
96.     GongYang(6);
97.     delay(300);
98.     GongYang(7);
99.     delay(300);
100.    GongYang(8);
101.    delay(300);
102.    GongYang(9);
103.    delay(300);
104.    GongYang(10);
105.    delay(300);
106.    GongYang(11);
107.    delay(300);
108.    GongYang(12);
109.    delay(300);
110.    GongYang(13);
111.    GongYin(1);
112.    delay(300);
113.    GongYin(2);
114.    delay(300);

115.    GongYin(4);
116.    GongYang(3);
117.    delay(300);
118.    GongYang(4);
119.    delay(300);
120.    GongYang(5);
121.    delay(300);
122.    GongYang(6);
123.    delay(300);
124.    GongYang(7);
125.    delay(300);
126.    GongYang(8);
127.    delay(300);
128.    GongYang(9);
129.    delay(300);
130.    GongYang(10);
131.    delay(300);
132.    GongYang(11);
133.    delay(300);
134.    GongYang(12);
135.    delay(300);
136. }
137.
138. void GongYang(int x)
139. {
140.     for (int i = 0; i < 8; i++)
141.     {
142.         digitalWrite(gyang[i], gong_yang[x][i]);
143.     }
144. }
145.
146. void GongYin(int y)
147. {
148.     for (int i = 0; i < 7; i++)
149.     {
150.         digitalWrite(gyin[i], gong_yin[y][i]);
151.     }
152. }
```

SETUP 语句和 **LOOP** 语句还有很多冗余的代码，用 **for** 循环优化代码。

[plain] view plaincopy 

```
1. //设置阴极接口
2. int d1 = 1;
3. int d2 = 2;
4. int d3 = 3;
5. int d4 = 4;
6. int d5 = 5;
7. int d6 = 6;
8. int d7 = 7;
9. //设置阳极接口
10. int a = 8;
11. int b = 9;
12. int c = 10;
13. int d = 11;
14. int e = 12;
15. int f = 13;
16. int g = A0;
17. int h = A1;
18.
19. byte gyang[8] = { a, b, c, d, e, f, g, h };
20. byte gyin[7] = { d1, d2, d3, d4, d5, d6, d7 };
21.
22. byte gong_yang[14][8] = {
23. { 1,1,0,0,0,0,0,0 }, // 0 = 缺水
24. { 0,0,1,1,0,0,0,0 }, // 1 = 低水位
25. { 0,0,0,0,1,1,1,0 }, // 2 = 定时关机
26. { 1,1,1,1,1,1,0,0 }, // 3 = 0
27. { 0,1,1,0,0,0,0,0 }, // 4 = 1
28. { 1,1,0,1,1,0,1,0 }, // 5 = 2
29. { 1,1,1,1,0,0,1,0 }, // 6 = 3
30. { 0,1,1,0,0,1,1,0 }, // 7 = 4
31. { 1,0,1,1,0,1,1,0 }, // 8 = 5
32. { 1,0,1,1,1,1,1,0 }, // 9 = 6
33. { 1,1,1,0,0,0,0,0 }, // 10 = 7
34. { 1,1,1,1,1,1,1,0 }, // 11 = 8
35. { 1,1,1,1,0,1,1,0 }, // 12 = 9
36. { 0,0,0,0,0,0,0,1 }, // 13 = 摄氏度符号、睡眠符号
37. };
38.
39. byte gong_yin[5][7] = {
40. { 1,1,1,1,1,1,0 }, // 0 = 缺水、低水位、定时关机
41. { 0,1,1,1,1,1,1 }, // 1 = 温度十位、摄氏度符号
42. { 1,0,1,1,1,1,1 }, // 2 = 温度个位、睡眠符号
43. { 0,0,1,1,1,1,1 }, // 3 = 温度个和十位
44. { 1,1,0,0,0,0,1 }, // 4 = 时钟个、十位和两点
45. };
46.
47. void setup()
48. {
49. pinMode(A0, OUTPUT);
50. pinMode(A1, OUTPUT);
51. for(int i=1; i<14; i++)
52. {
53. pinMode(i, OUTPUT);
54. }
55. }
56.
57. void loop()
58. {
59. GongYin(0);
60. for(int i=0; i<3; i++)
61. {
62. GongYang(i);
63. delay(300);
64. }
65. GongYin(3);
66. for(int i=3; i<13; i++)
67. {
68. GongYang(i);
69. delay(300);
70. }
71. GongYang(13);
72. GongYin(1);
73. delay(300);
74. GongYin(2);
75. delay(300);
76. GongYin(4);
77. for(int i=3; i<13; i++)
78. {
79. GongYang(i);
80. delay(300);
81. }
82. }
83.
84. void GongYang(int x)
85. {
86. for (int i = 0; i < 8; i++)
87. {
```

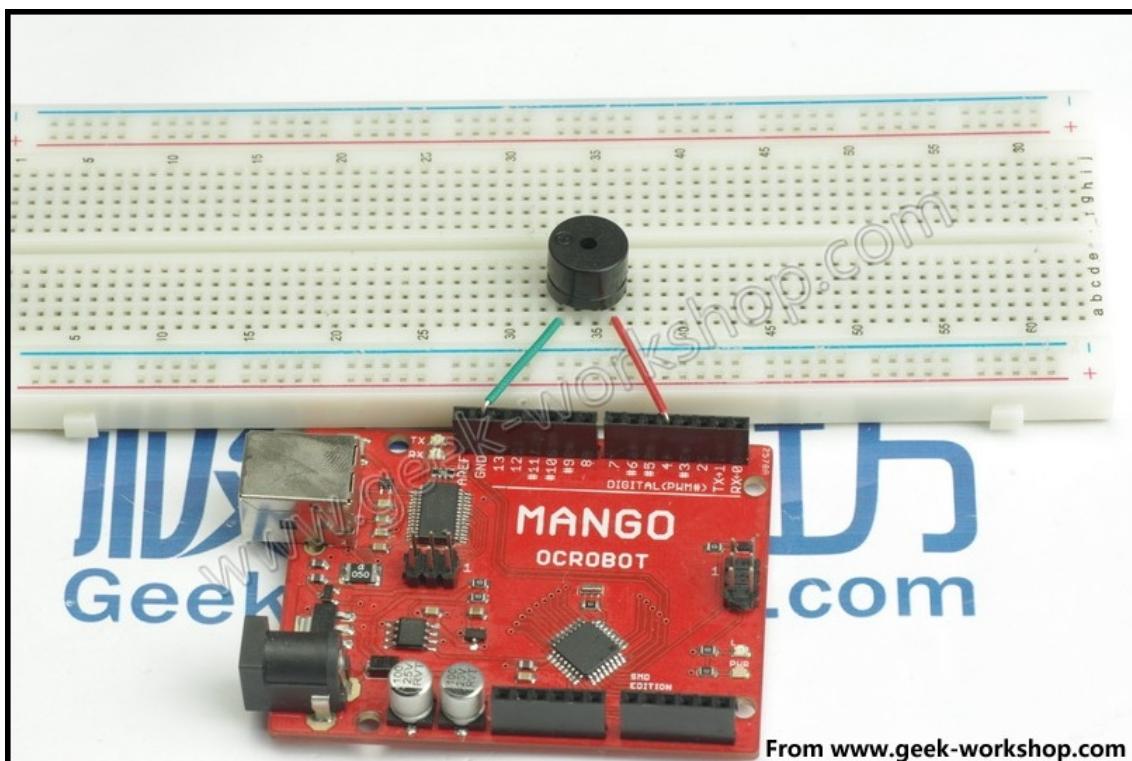
```
88.     digitalWrite(gyang[i], gong_yang[x][i]);  
89. }  
90.  
91.  
92. void GongYin(int y)  
93. {  
  
94.     for (int i = 0; i < 7; i++)  
95.     {  
96.         digitalWrite(gyin[i], gong_yin[y][i]);  
97.     }  
98. }
```

Arduino 无源蜂鸣器警报声

之前做了个有源蜂鸣器的提示音，这次无源蜂鸣器到货了，先来个警报声测试一下，下一步开始无源蜂鸣器的音乐播放之路，期待吧，骚年！

[plain] view plaincopy C

```
1. void setup()
2. {
3.   pinMode(5,OUTPUT);
4. }
5.
6. void loop()
7. {
8.   for(int i=200;i<=800;i++) //用循环的方式将频率从 200HZ 增加到 800HZ
9.   {
10.     tone(5,i); //在四号端口输出频率
11.     delay(5); //该频率维持 5 毫秒
12.   }
13.   delay(4000); //最高频率下维持 4 秒钟
14.   for(int i=800;i>=200;i--)
15.   {
16.     tone(5,i);
17.     delay(10);
18.   }
19. }
```



From www.geek-workshop.com

Arduino 数码管 LED 驱动 读位法 16 进制

区别与上两个驱动的方法，用 `bitRead()` 读取位数，代码简洁一点。

```
1. //设置阴极接口
2. int d1 = 1;
3. int d2 = 2;
4. int d3 = 3;
5. int d4 = 4;
6. int d5 = 5;
7. int d6 = 6;
8. int d7 = 7;
9. //设置阳极接口
10. int a = 8;
11. int b = 9;
12. int c = 10;
13. int d = 11;
14. int e = 12;
15. int f = 13;
16. int g = A0;
17. int h = A1;
18. const unsigned char Yang[] = { a, b, c, d, e, f, g, h };
19. const unsigned char YangCode[] ={
20.     0x03,    // 0 = 缺水
21.     0x0C,    // 1 = 低水位
22.     0x70,    // 2 = 定时关机
23.     0x3F,    // 3 = 0
24.     0x06,    // 4 = 1
25.     0x5B,    // 5 = 2
26.     0x4F,    // 6 = 3
27.     0x66,    // 7 = 4
28.     0x6D,    // 8 = 5
29.     0x7D,    // 9 = 6
30.     0x07,    // 10 = 7
31.     0x7F,    // 11 = 8
32.     0x6F,    // 12 = 9
33.     0x80,    // 13 = 摄氏度符号、睡眠符号
34. };
35. byte gyin[7] = { d1, d2, d3, d4, d5, d6, d7 };
36. byte gong_yin[5][7] = {
37.             { 1,1,1,1,1,1,0 }, // 0 = 缺水、低水位、定时关机
38.             { 0,1,1,1,1,1,1 }, // 1 = 温度十位、摄氏度符号
39.             { 1,0,1,1,1,1,1 }, // 2 = 温度个位、睡眠符号
40.             { 0,0,1,1,1,1,1 }, // 3 = 温度个和十位
41.             { 1,1,0,0,0,0,1 }, // 4 = 时钟个、十位和两点
42.         };
```

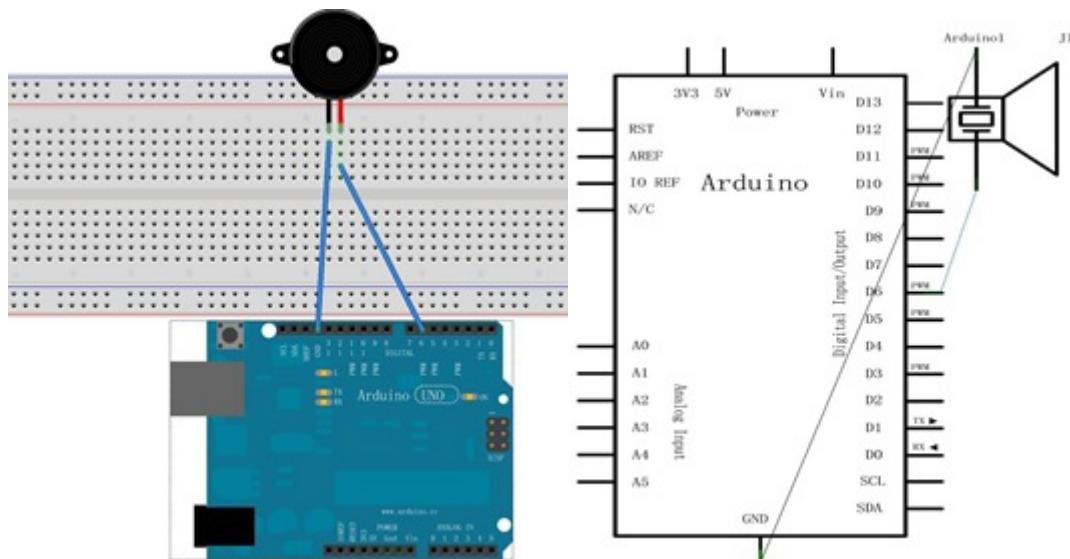
```
43. void setup()
44. {
45.     pinMode(A0, OUTPUT);
46.     pinMode(A1, OUTPUT);
47.     for(int i=1; i<14;i++)
48.     {
49.         pinMode(i,OUTPUT);
50.     }
51. }
52.
53. void loop()
54. {
55.     GongYin(3);
56.     GongYin(4);
57.     for (int a=3;a<13;a++)
58.     {
59.         digitalWrite(h, HIGH);//显示时钟的冒号
60.         delay(300);
61.         YangShow(a);
62.         delay(300);
63.     }
64. }
65.
66. void GongYin(int y)
67. {
68.     for (int i = 0; i < 7; i++)
69.     {
70.         digitalWrite(gyin[i], gong_yin[y][i]);
71.     }
72. }
73.
74. void YangShow(char data)
75. {
76.     char i;
77.     char j;
78.     char hc;
79.     if(0<=data<14)
80.     {
81.         hc = YangCode[data];
82.         for(i=0;i<8;i++)
83.         {
84.             j = bitRead(hc,i);
85.             digitalWrite(Yang[i], j);
86.         }
87.     }
88. }
```

Arduino 无源蜂鸣器 音乐播放实验

上两次实验做了有源蜂鸣器的按键响声控制和无源蜂鸣器的报警声控制，这次尝试做 Mid 音乐播放，还好有点乐理知识，吹拉弹类的乐器都会点，最要做好音符对应的频率，节拍对应的间隔时间，就能做 Mid 音乐了。

无源蜂鸣器：一种一体化结构的电子讯响器，分为有源蜂鸣器与无源蜂鸣器。这里的“源”不是指电源，而是指震荡源，有源蜂鸣器内部带震荡源，所以只要一通电就会响，而无源内部不带震荡源，所以如果仅用直流信号无法令其鸣叫，必须用 2K-5K 的方波去驱动它。从外观上看，两种蜂鸣器区别不大，但将两种蜂鸣器的引脚都朝上放置时，可以看出有绿色电路板的一种是无源蜂鸣器，没有电路板而用黑胶封闭的一种是有源蜂鸣器。

电路及原理图都非常简单，这里用《欢乐颂》做测试。



欢乐颂

1=D $\frac{4}{4}$

贝多芬 曲
邓映易 译配

3 3 4 5 | 5 4 3 2 | 1 1 2 3 | 3 . 2 2 - |

Joy-ful, joy- ful, we a- dore thee, God of glo- ry, lord of love.
欢乐女神，圣洁美丽，灿烂光芒照大地，

3 3 4 5 | 5 4 3 2 | 1 1 2 3 | 2 . 1 1 - |

Heart un-flo- like flowers be-fore thee, O-pening to the sun a-bove.
我们怀着火样的热情，来到你的圣殿里！

||: 2 2 3 1 | 2 3 4 3 1 | 2 3 4 3 2 | 1 2 5 3 |

Melt the clouds of sin and sad-ness. rive the dark of doubts a- way. Give
你的威力能把人类重新团结在一起，在

3 3 4 5 | 5 4 3 4 2 | 1 1 2 3 | 2 . 1 1 - :|

-ver of im- mor-tal glad-ness. full us with the light of day.
你温柔翅膀之下，一切人类成兄弟。

```
1. #define D0 -1
2. #define D1 262
3. #define D2 293
4. #define D3 329
5. #define D4 349
6. #define D5 392
7. #define D6 440
8. #define D7 494
9.
10. #define M1 523
11. #define M2 586
12. #define M3 658
13. #define M4 697
14. #define M5 783
15. #define M6 879
16. #define M7 987
17.
18. #define H1 1045
19. #define H2 1171
20. #define H3 1316
21. #define H4 1393
22. #define H5 1563
23. #define H6 1755
24. #define H7 1971
25. //列出全部D调的频率
26.
27. #define WHOLE 1
28. #define HALF 0.5
29. #define QUARTER 0.25
30. #define EIGHTH 0.25
31. #define SIXTEENTH 0.625
32. //列出所有节拍
33.
34. int tune[] =          //根据简谱列出各频率
35. {
36.     M3, M3, M4, M5,
37.     M5, M4, M3, M2,
38.     M1, M1, M2, M3,
39.     M3, M2, M2,
40.     M3, M3, M4, M5,
41.     M5, M4, M3, M2,
42.     M1, M1, M2, M3,
43.     M2, M1, M1,
44.     M2, M2, M3, M1,
45.     M2, M3, M4, M3, M1,
46.     M2, M3, M4, M3, M2,
```

```
47.     M1,M2,D5,D0,
48.     M3,M3,M4,M5,
49.     M5,M4,M3,M4,M2,
50.     M1,M1,M2,M3,
51.     M2,M1,M1
52. };
53. float durt[] =      //根据简谱列出各节拍
54. {
55.     1,1,1,1,
56.     1,1,1,1,
57.     1,1,1,1,
58.     1+0.5,0.5,1+1,
59.     1,1,1,1,
60.     1,1,1,1,
61.     1,1,1,1,
62.     1+0.5,0.5,1+1,
63.     1,1,1,1,
64.     1,0.5,0.5,1,1,
65.     1,0.5,0.5,1,1,
66.     1,1,1,1,
67.     1,1,1,1,
68.     1,1,1,0.5,0.5,
69.     1,1,1,1,
70.     1+0.5,0.5,1+1,
71. };
72. int length;
73. int tonepin=5;    //得用 5 号接口
74. void setup()
75. {
76.     pinMode(tonepin,OUTPUT);
77.     length=sizeof(tune)/sizeof(tune[0]);    //计算长度
78. }
79. void loop()
80. {
81.     for(int x=0;x<length;x++)
82.     {
83.         tone(tonepin,tune[x]);
84.         delay(500*durt[x]);
85.         //这里用来根据节拍调节延时，500 这个指数可以自己调整，在该音乐中，我发现用 500 比较合适。
86.         noTone(tonepin);
87.     }
88.     delay(2000);
89. }
```

下次再尝试用 **64** 和弦的喇叭，演奏更精彩的音乐。

Arduino 温湿度传感器 DHT11 模块实验

网上有很多 DHT11 的测试，试了 N 个程序，总是不得要领，各种报错，最后终于找到一套可用的库。

首先是 DHT11.h 文件

```
1. #ifndef __DHT11_H__
2. #define __DHT11_H__
3. #include <Arduino.h>
4. //DHT11 IO 设置
5. #define DHT11_DQ 2
6. #define DHT11_DQ_0 digitalWrite(DHT11_DQ,LOW)
7. #define DHT11_DQ_1 digitalWrite(DHT11_DQ,HIGH)
8.
9. //函数或者变量声明
10. extern void DHT11_Init();
11. extern unsigned char DHT11_Read_Byte();
12. extern void DHT11_Read();
13.
14. extern unsigned char HUMI_Buffer_Int;
15. extern unsigned char TEM_Buffer_Int;
16.
17. #endif
```

其次是 DHT11.cpp 文件

```
1. #include "DHT11.h"
2. //定义变量
3. unsigned char HUMI_Buffer_Int = 0;
4. unsigned char TEM_Buffer_Int = 0;
5. //*****
6. //初始化 DHT11
7. //*****
8. void DHT11_Init()
9. {
10.     pinMode(DHT11_DQ,OUTPUT);
11.     DHT11_DQ_0;                      //拉低总线，发开始信号;
12.     delay(30);                      //延时要大于 18ms，以便 DHT11 能检测到开始信号;
13.     DHT11_DQ_1;
14.     delayMicroseconds(40);           //等待 DHT11 响应;
15.     pinMode(DHT11_DQ,INPUT);
16.     while(digitalRead(DHT11_DQ) == HIGH);
17.     delayMicroseconds(80);           //DHT11 发出响应，拉低总线 80us;
```

```
18.     if(digitalRead(DHT11_DQ) == LOW);
19.     delayMicroseconds(80);           //DHT11 拉高总线 80us 后开始发送数据;
20. }
21.
22. //*****
23. //读一个字节 DHT11 数据
24. //*****
25. unsigned char DHT11_Read_Byte()
26. {
27.     unsigned char i,dat = 0;
28.     unsigned int j;
29.     pinMode(DHT11_DQ,INPUT);
30.     for( i=0; i<8; i++)
31.     {
32.         if(digitalRead(DHT11_DQ) == LOW)
33.         {
34.             while(digitalRead(DHT11_DQ) == LOW);    //等待 50us;
35.             delayMicroseconds(30);      //判断高电平的持续时间, 以判定数据是'0'还是'1';
36.             if(digitalRead(DHT11_DQ) == HIGH)
37.                 dat |= (1<<(7-i));        //高位在前, 低位在后;
38.             while(digitalRead(DHT11_DQ) == HIGH); //数据'1', 等待下一位的接收;
39.         }
40.     }
41.     return dat;
42. }
43.
44. //*****
45. //读取温湿度值, 存放在 TEM_Buffer 和 HUMI_Buffer
46. //*****
47. void DHT11_Read()
48. {
49.     DHT11_Init();
50.     HUMI_Buffer_Int = DHT11_Read_Byte(); //读取湿度的整数值
51.     DHT11_Read_Byte();                //读取湿度的小数值
52.     TEM_Buffer_Int = DHT11_Read_Byte(); //读取温度的整数值
53.     DHT11_Read_Byte();                //读取温度的小数值
54.     DHT11_Read_Byte();                //读取校验和
55.     delayMicroseconds(50);           //DHT11 拉低总线 50us
56.     pinMode(DHT11_DQ,OUTPUT);
57.     DHT11_DQ_1;                     //释放总线
58. }
```

最后是主程序文件

```
1. #include <Arduino.h>
2. #include "DHT11.h"
3.
4. void setup() //Arduino 程序初始化程序放在这里，只在开机时候运行一次
5. {
6.     Serial.begin(9600); //设置通讯的波特率为 9600
7.     DHT11_Read(); //读取温湿度值
8.     delay(200); //等待传感器稳定
9. }
10.
11. void loop() //Arduino 程序的主程序部分，循环运行内部程序
12. {
13.     DHT11_Read(); //读取温湿度值
14.     Serial.print("HUMI = ");
15.     Serial.print(HUMI_Buffer_Int);
16.     Serial.println(" %RH");
17.     Serial.print("TMEP = ");
18.     Serial.print(TEM_Buffer_Int);
19.     Serial.println(" C");
20.     delay(1000); //延时 1s
21. }
```

三个文件保存在同一个文件夹即可。

