

Electronic Clock Report

MG

2023.12.10

目录

1	要求	4
1.1	基本要求	4
1.2	发挥部分	4
2	摘要	5
3	功能实现	6
3.1	时间，学号，温度的交替显示	6
3.1.1	时间显示实现	6
3.1.2	学号，温度交替显示实现	6
3.2	6 种时间动画的实现	7
3.2.1	1,2 两种时间动画实现	7
3.2.2	3,4 两种时间动画实现	8
3.2.3	5,6 两种时间动画实现	12
3.3	7 种脱机动画的实现	14
3.3.1	1,2 两种脱机动画的实现	14
3.3.2	3,4 两种脱机动画的实现	15
3.3.3	5,6 两种脱机动画的实现	16
3.3.4	7 脱机动画的实现	17
3.4	光控调节 LED 以及数码管的亮度	19
3.4.1	光控开关设置	19
3.4.2	光控开关状态显示	19
3.4.3	光控判断与调节亮度显示	20
3.5	闹钟的开关与显示设置	21
3.5.1	闹钟的开关设置	21
3.5.2	闹钟分钟的调节	21
3.5.3	闹钟时钟的调节	22
3.5.4	闹钟分钟的显示	23
3.5.5	闹钟时钟的显示	24
3.5.6	闹钟的手动关闭	24

目录	3
4 参考文献	26

1 要求

1.1 基本要求

- 参照焊接指导，完成硬件电路的焊接与测试过程。
- 在输入电压下，能够精确显示时，分（24 小时制）。
- 完成温度采集与显示，实现时间与温度交替显示。
- 实现 LED 的点亮，完成时间秒的显示。1 秒对应一个 LED 灯点亮。
- 实现闹钟的功能，能够设置闹钟时间，和闹钟功能的打开与关闭。

1.2 发挥部分

- 保证焊接部分和作品外观尽量美观。
- 能够实现数码管显示自己学号后 4 位，并且能够实现时间，温度，学号的切换或者交替显示。
- 能够实现 6 种时间动画。
- 能够实现 7 种脱机动画。
- 实现光控功能，根据环境自动调节亮度。

2 摘要

基于比赛要求，首先对电子小闹钟零件进行焊接，然后应用 keil 软件以及小黑学长提供的基本框架进行程序设计，通过对 c 语言的学习与理解，最终能够读懂小黑的代码并且衍生拓展，能够实现

- (1) 时间，学号，温度的交替显示。
- (2) 6 种时间动画以及 7 种脱机动画。
- (3) 光控调节 LED 以及数码管的亮度。
- (4) 闹钟的开关与显示设置。
- (5) 在时间动画中添加了两种 1min 计时功能。

以上效果实现能够涵盖所有要求，并且 (5) 的实现超出题目要求。

3 功能实现

3.1 时间，学号，温度的交替显示

3.1.1 时间显示实现

```
1      display_array[0] = time_hour / 16;  
2      display_array[1] = time_hour % 16;  
3      display_array[2] = time_min / 16;  
4      display_array[3] = time_min % 16;
```

代码解释 display_array 是一个存储数码管显示数字的数组, 其中 display_array[0] 代表小时十位数, 其中 display_array[1] 代表小时个位数, 其中 display_array[2] 代表分钟十位数, 其中 display_array[3] 代表分钟个位数。

time_hour 和 time_min 分别是存储小时数据和存储分钟数据的两个变量。

3.1.2 学号，温度交替显示实现

```
1      if (bcd2hex(time_sec) <= 2)  
2      {  
3          display_array[0] = 1;  
4          display_array[1] = 8;  
5          display_array[2] = 7;  
6          display_array[3] = 8;  
7      }
```

代码解释 time_sec 变量存储了秒针数据, 当秒针指在 0 到 2 秒的时候, 显示学号 1878。

```
1     else if (bcd2hex(time_sec) >= 30 && bcd2hex(time_sec) <= 32)
2
3     {
4         temperature = getTemperature();
5
6         display_array[0] = _SPACE;
7         display_array[1] = temperature / 1000;
8         display_array[2] = temperature % 1000 / 100;
9
10        display_array[3] = _C;
11    }
```

代码解释 temperature 变量中存储了温度数据，在秒针指向 30 到 32 时显示温度，摄氏度单位用 C 表示。

3.2 6 种时间动画的实现

3.2.1 1,2 两种时间动画实现

动画 1

```
1     if (time_animate == 0)
2     {
3         ledAllOff();
4         setLed(bcd2hex(time_sec));
5     }
```

代码解释 time_animate 是时间动画的指针，当 time_animate=0 时，开机动画 1 启用，效果为背景灯全灭的秒钟点亮转动。

动画 2

```
1     else if (time_animate == 1)
2     {
3         ledAllOn();
4         resetLed(bcd2hex(time_sec));
5     }
```

代码解释 time_animate 是时间动画的指针，当 time_animate=1 时，开机动画 2 启用，效果为背景灯全亮的秒钟熄灭转动。

3.2.2 3,4 两种时间动画实现

动画 3

```
1  else if (time_animate == 2)
2  {
3      ledAllOff();
4      setLed(bcd2hex(time_sec));
5      if (bcd2hex(time_sec)%60>=5)
6      {
7          led_array[0]&=0xef;
8      }
9      if (bcd2hex(time_sec)%60>=10)
10     {
11         led_array[1]&=0xfd;
12     }
13     if (bcd2hex(time_sec)%60>=15)
14     {
15         led_array[1]&=0xbd;
16     }
17     if (bcd2hex(time_sec)%60>=20)
18     {
```



```
19         led_array[2]&=0xf7;
20     }
21     if (bcd2hex(time_sec)%60>=25)
22     {
23         led_array[3]&=0xfe;
24     }
25     if (bcd2hex(time_sec)%60>=30)
26     {
27         led_array[3]&=0xde;
28     }
29     if (bcd2hex(time_sec)%60>=35)
30     {
31         led_array[4]&=0xfb;
32     }
33     if (bcd2hex(time_sec)%60>=40)
34     {
35         led_array[4]&=0x7b;
36     }
37     if (bcd2hex(time_sec)%60>=45)
38     {
39         led_array[5]&=0xef;
40     }
41     if (bcd2hex(time_sec)%60>=50)
42     {
43         led_array[6]&=0xfd;
44     }
45     if (bcd2hex(time_sec)%60>=55)
46     {
47         led_array[6]&=0xbd;
48     }
49
```

```
50 }
```

代码解释 time_animate 是时间动画的指针，当 time_animate=2 时，开机动画 3 启用，效果为背景灯全灭，秒针点灯转动的过程中，将红灯逐个点亮。

动画 4

```
1  else if (time_animate == 3)
2  {
3      ledAllOn();
4      resetLed(bcd2hex(time_sec));
5      if (bcd2hex(time_sec)%60>=5)
6      {
7          led_array[0]|=~0xef;
8      }
9      if (bcd2hex(time_sec)%60>=10)
10     {
11         led_array[1]|=~0xfd;
12     }
13     if (bcd2hex(time_sec)%60>=15)
14     {
15         led_array[1]|=~0xbd;
16     }
17     if (bcd2hex(time_sec)%60>=20)
18     {
19         led_array[2]|=~0xf7;
20     }
21     if (bcd2hex(time_sec)%60>=25)
22     {
23         led_array[3]|=~0xfe;
24     }
```

```
25         if (bcd2hex(time_sec)%60>=30)
26         {
27             led_array[3]|=~0xde;
28         }
29         if (bcd2hex(time_sec)%60>=35)
30         {
31             led_array[4]|=~0xfb;
32         }
33         if (bcd2hex(time_sec)%60>=40)
34         {
35             led_array[4]|=~0x7b;
36         }
37         if (bcd2hex(time_sec)%60>=45)
38         {
39             led_array[5]|=~0xef;
40         }
41         if (bcd2hex(time_sec)%60>=50)
42         {
43             led_array[6]|=~0xfd;
44         }
45         if (bcd2hex(time_sec)%60>=55)
46         {
47             led_array[6]|=~0xbd;
48         }
49     }
```

代码解释 time_animate 是时间动画的指针，当 time_animate=3 时，开机动画 4 启用，效果为背景灯全亮，秒针灭灯转动的过程中，将红灯逐个熄灭。

3.2.3 5,6 两种时间动画实现

动画 5

```
1  else if (time_animate == 4)
2  {
3      ledAllOff();
4      setLed1(bcd2hex(time_sec));
5      for(k=0;k<7;k++)
6      {
7          if(bcd2hex(time_sec)>=8*(k+1))
8          {
9              led_array[k]=0;
10             }
11         }
12         if(bcd2hex(time_sec)==0)
13         {
14             led_array[7]=0xf;
15             bell=!bell;
16             delay(1000);
17             bell=1;
18         }
19
20
21 }
```

代码解释 time_animate 是时间动画的指针，当 time_animate=4 时，开机动画 5 启用，效果为背景灯全灭的秒钟点亮转动，转动过程中把经过的 LED 灯点亮，当经历 1min 后，即把所有灯点亮，bell 电平反置，蜂鸣器响起，实现 1min 计时。

动画 6

```
1  else
2  {
3      ledAllOn ();
4      resetLed1 (bcd2hex (time_sec));
5      for (k=0;k<7;k++)
6      {
7          if (bcd2hex (time_sec)>=8*(k+1))
8          {
9              led_array[k]=0xff;
10             }
11         }
12         if (bcd2hex (time_sec)==0)
13         {
14             led_array[7]=0;
15             bell=!bell;
16             delay (1000);
17             bell=1;
18         }
19     }
```

代码解释 time_animate 是时间动画的指针，当 time_animate=5 时，开机动画 6 启用，效果为背景灯全亮的秒钟灭灯转动，转动过程中把经过的 LED 灯熄灭，当经历 1min 后，即把所有灯熄灭，bell 电平反置，蜂鸣器响起，实现 1min 计时。

3.3 7 种脱机动画的实现

3.3.1 1,2 两种脱机动画的实现

动画 1

```
1      if (offline_animate == 0)
2      {
3          i%=15;
4          ledAllOff();
5          setLed(i);
6          setLed(i + 15);
7          setLed(i + 30);
8          setLed(i + 45);
9      }
```

代码解释 offline_animate 是脱机动画的指针，当 offline_animate=0 时，脱机动画 1 启用，其中 i 作为 led_array 数组的指针，对其进行 15 取模的运算，实现以两个 led_array 数组跨度，即间隔 16 个 LED 灯，四个 LED 灯进行顺时针亮灯旋转效果。

动画 2

```
1      else if (offline_animate == 1)
2      {
3          i%=15;
4          ledAllOn();
5          resetLed(i);
6          resetLed(i + 15);
7          resetLed(i + 30);
8          resetLed(i + 45);
9      }
```

代码解释 `offline_animate` 是脱机动画的指针，当 `offline_animate=1` 时，脱机动画 2 启用，其中 `i` 作为 `led_array` 数组的指针，对其进行 15 取模的运算，实现以两个 `led_array` 数组跨度，即间隔 16 个 LED 灯，四个 LED 灯进行顺时针灭灯旋转效果。

3.3.2 3,4 两种脱机动画的实现

动画 3

```
1  else if (offline_animate == 2)
2  {
3      i%=10;
4      ledAllOff();
5      setLed(i);
6      setLed(i + 10);
7      setLed(i + 20);
8      setLed(i + 30);
9      setLed(i + 40);
10     setLed(i + 50);
11
12 }
```

代码解释 `offline_animate` 是脱机动画的指针，当 `offline_animate=2` 时，脱机动画 3 启用，其中 `i` 作为 `led_array` 数组的指针，对其进行 10 取模的运算，实现间隔 10 个 LED 灯，六个 LED 灯进行顺时针亮灯旋转效果。

动画 4

```
1  else if (offline_animate == 3)
2  {
3      i%=10;
4      ledAllOn();
5      resetLed(i);
6      resetLed(i + 10);
7      resetLed(i + 20);
8      resetLed(i + 30);
9      resetLed(i + 40);
10     resetLed(i + 50);
11
12 }
```

代码解释 offline_animate 是脱机动画的指针，当 offline_animate=3 时，脱机动画 4 启用，其中 i 作为 led_array 数组的指针，对其进行 10 取模的运算，实现间隔 10 个 LED 灯，六个 LED 灯进行顺时针灭灯旋转效果。

3.3.3 5,6 两种脱机动画的实现

动画 5

```
1  else if (offline_animate == 4)
2  {
3      i%=30;
4      ledAllOff();
5      setLed(i);
6      setLed(i + 30);
7
8  }
```


代码解释 offline_animate 是脱机动画的指针，当 offline_animate=4 时，脱机动画 5 启用，其中 i 作为 led_array 数组的指针，对其进行 30 取模的运算，实现间隔 30 个 LED 灯，两个 LED 灯进行顺时针亮灯旋转效果。

动画 6

```
1  else if (offline_animate == 5)
2  {
3      i%=30;
4      ledAllOn();
5      resetLed(i);
6      resetLed(i + 30);
7
8  }
```

代码解释 offline_animate 是脱机动画的指针，当 offline_animate=5 时，脱机动画 6 启用，其中 i 作为 led_array 数组的指针，对其进行 30 取模的运算，实现间隔 30 个 LED 灯，两个 LED 灯进行顺时针灭灯旋转效果。

3.3.4 7 脱机动画的实现

动画 7

```
1  else
2  {
3      ledAllOff();
4      for (s=3;s<8;s++)
5      {
6          led_array[s]=0;
7      }
8      i%=60;
9      if (i<=24)
10     {
11         setLed(i);
```

```
12         }  
13         else  
14         {  
15             resetLed(i);  
16         }  
17     }
```

代码解释 offline_animate 是脱机动画的指针，当 offline_animate=6 时，脱机动画 7 启用，其中 i 作为 led_array 数组的指针，对其进行 60 取模的运算，实现其对 60 秒的长度表示，当 i 在 0 到 24 时，执行 setLed 操作，即点亮操作，当 i 在 25 到 59 的时，执行 resetLed 操作，即熄灭操作。for 循环语句表示将 led_array 数组的 3 到 7 号全部点亮。

3.4 光控调节 LED 以及数码管的亮度

3.4.1 光控开关设置

```
1  if (page == 8)
2  {
3      if (BTN2 == 0)
4      {
5          delay(20);
6
7          if (BTN2 == 0)
8          {
9              light_control = !light_control;
10             while (BTN2 == 0)
11                 ;
12             }
13             delay(10);
14         }
15     }
```

代码解释 在 page8 页面进行光控开关设置，light_control 代表光控开关，1 开 0 关，当 BTN 按键按下，light_control 反置，最终实现光控开关设置。

3.4.2 光控开关状态显示

```
1  if (page == 8)
2  {
3      ledAllOff();
4      display_array[0] = _C;
5      display_array[1] = _C;
6      display_array[2] = _SPACE;
7      display_array[3] = light_control;
```

```
8 }
```

代码解释 在 page8 页面进行光控开关显示，light_control 代表光控开关，1 开 0 关，display_array 数组为数码管显示数组，最终显示效果为 1 或者 0。

3.4.3 光控判断与调节亮度显示

```
1 // 光控判断
2 if (light_control == 1)
3 {
4     light = getLight();
5 }
6 else
7 {
8     light = 8;
9 }
10
11 // 显示
12 display();
13 delay(light);
```

代码解释 在 page8 页面进行光控判断，light_control 代表光控开关，如果为 1，light 开始接受环境光照数据，并且在显示中实现 LED 灯的亮度调节。

3.5 闹钟的开关与显示设置

3.5.1 闹钟的开关设置

```
1  if (page == 5)
2  {
3      if (BTN2 == 0)
4      {
5          delay(20);
6
7          if (BTN2 == 0)
8          {
9              alarm_control = !alarm_control;
10             while (BTN2 == 0)
11                 ;
12             }
13             delay(10);
14         }
15     }
```

代码解释 在 page5 页面进行闹钟开关设置, alarm_control 代表闹钟开关, 1 开 0 关, 按下 BTN2 按键进行反置操作。

3.5.2 闹钟分钟的调节

```
1  // 设置分钟
2  if (page == 3)
3  {
4      if (BTN2 == 0)
5      {
6          delay(20);
7          if (BTN2 == 0)
```

```
8      {
9          // 分钟超过60置0
10         if (alarm_min >= 0x60)
11             alarm_min = 0;
12         // 设置分钟加1
13         alarm_min = alarm_min + 0x01;
14         // 16进制转为BCD
15         if ((alarm_min & 0x0f) >= 0x0a)
16             alarm_min = (alarm_min & 0xf0) + 0x10;
17         while (BTN2 == 0)
18             ;
19     }
20     delay(10);
21 }
22 }
```

代码解释 在 page3 页面进行闹钟分钟设置, alarm_min 代表分钟数据, 通过对 alarm_min 的调整, 达到调整闹钟分钟的目的。

3.5.3 闹钟时钟的调节

```
1 // 设置时钟
2 if (page == 4)
3 {
4     if (BTN2 == 0)
5     {
6         delay(20);
7         if (BTN2 == 0)
8         {
9             alarm_hour += 0x01;
10            // 时钟超过24置0
```

```
11         if ((alarm_hour & 0x0f) >= 0x0a)
12             alarm_hour = (alarm_hour & 0xf0) + 0x10;
13         if (alarm_hour >= 0x24)
14             alarm_hour = 0;
15         while (BTN2 == 0)
16             ;
17     }
18     delay(10);
19 }
20 }
```

代码解释 在 page4 页面进行闹钟时钟设置，alarm_hour 代表时钟数据，通过对 alarm_hour 的调整，达到调整闹钟时钟的目的。

3.5.4 闹钟分钟的显示

```
1  if (page == 3)
2  {
3      ledAllOff();
4      if (blink > 0)
5      {
6          display_array[2] = alarm_min / 16;
7          display_array[3] = alarm_min % 16;
8      }
9      else
10     {
11         display_array[2] = _SPACE;
12         display_array[3] = _SPACE;
13     }
14     display_array[0] = alarm_hour / 16;
15     display_array[1] = alarm_hour % 16;
```

```
16 }
```

代码解释 在 page3 页面进行闹钟分钟的显示，display_array 数组的 2,3 位分别存放分钟十位和个位，由于 alarm_min 是 16 进制数，所以对其进行 16 整除和取余过程，就能得到十进制分钟数据。

3.5.5 闹钟时钟的显示

```
1  if (page == 4)
2  {
3      ledAllOff();
4      display_array[2] = alarm_min / 16;
5      display_array[3] = alarm_min % 16;
6      if (blink > 0)
7      {
8          display_array[0] = alarm_hour / 16;
9          display_array[1] = alarm_hour % 16;
10     }
11     else
12     {
13         display_array[0] = _SPACE;
14         display_array[1] = _SPACE;
15     }
16 }
```

代码解释 在 page4 页面进行闹钟时钟的显示，display_array 数组的 0,1 位分别存放时钟十位和个位，由于 alarm_hour 是 16 进制数，所以对其进行 16 整除和取余过程，就能得到十进制时钟数据。

3.5.6 闹钟的手动关闭


```
1  if (alarm_control == 1)
2  {
3      if (time_hour == alarm_hour && time_min ==
4          alarm_min)
5      {
6          if (BTN1 == 0)
7          {
8              delay(10);
9              if (BTN1 == 0)
10             {
11                 alarm_control = 0;
12                 while (BTN1 == 0)
13                     ;
14             }
15         }
16         if (BTN2 == 0)
17         {
18             delay(10);
19             if (BTN2 == 0)
20             {
21                 alarm_control = 0;
22                 while (BTN2 == 0)
23                     ;
24             }
25         }
26     }
27 }
```

代码解释 当闹钟响起时，用户可以手动关闭，按下 BTN1 或者 BTN2 按键，都可以实现闹钟的关闭，再次开启需要到闹钟设置页面再次设置。

4 参考文献

小黑学长 github 仓库中的代码框架:

<https://github.com/mobyw/DigitalTubeClock?>