4.4 倒计时定时器的设计、制作与调试

一、实验目的

1、学习掌握用 Arduino UNO 设计倒计时定时器；

2、学习掌握 PCB 电路板的设计和制作；

3、学习掌握 Arduino UNO 扩展板的设计与制作；

4、学习掌握旋转编码器的使用。

二、实验设计方案

在日常生活中常常需要定时器，通过设定时间来通知我们一项任务完成或者下一项任务应该开始。本项目使用 Arduino UNO 与其扩展板来实现倒计时定时器。硬件框图如图 4.13 所示，下面对各部分模块进行说明。

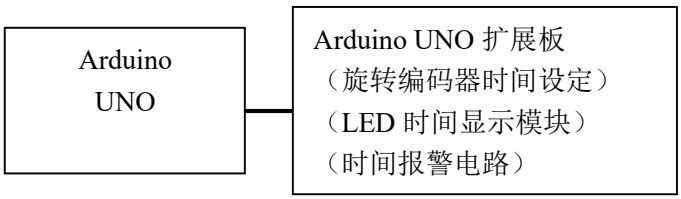


图 4.13 倒计时定时器框图

1、旋转编码器

旋转编码器是一种数字器件，这种器件有两个输出（A 和 B）。当旋转旋钮时，输出会有变化，这个变化就告诉你旋钮是顺时针旋转还是逆时针旋转。有关旋转编码器的内容，详见附录 2。

2、LED 时间显示模块

（1）7 段数码管结构

7 段数码管一般由 8 个发光二极管组成，其中由 7 个细长的发光二极管组成数字显示，另外一个圆形的发光二极管显示小数点。

当发光二极管导通时，相应的一个点或一个笔画发光。控制相应的二极管导通，就能显示出各种字符，尽管显示的字符形状有些失真，能显示的字符数量也有限，但其控制简单，使有也方便。发光二极管的阳极连在一起的称为共阳极数码管，阴极连在一起的称为共阴极数码管，如图 4.14 所示。

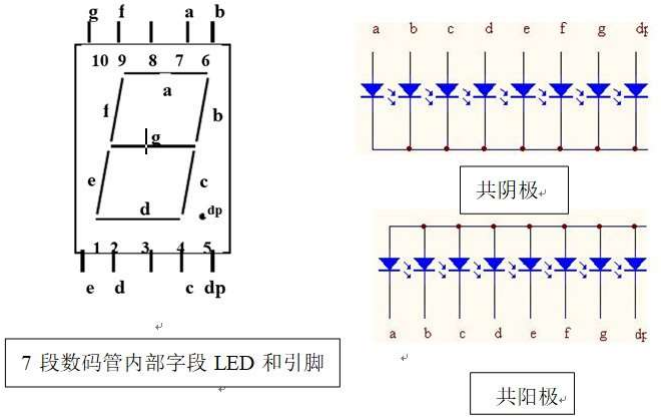


图 4.14　7 段数码管结构图

（2） 7 段数码管驱动方法

发光二极管（LED 是一种由磷化镓（GaP）等半导体材料制成的，能直接将电能转变成光能的发光显示器件。当其内部有一一电流通过时，它就会发光。

7 段数码管每段的驱动电流和其他单个 LED 发光二极管一样，一般为 5～10mA；正向电压随发光材料不同表现为 1.8~2.5V 不等。

7 段数码管的显示方法可分为静态显示与动态显示，下面分别介绍。

1）静态显示驱动：

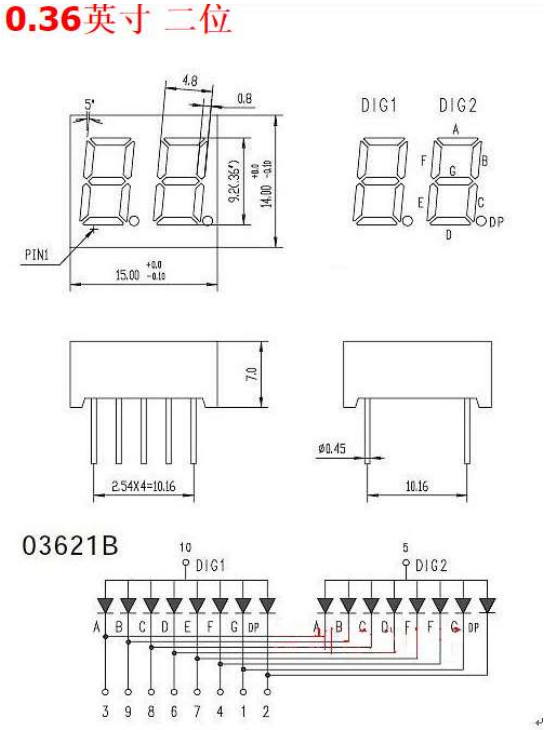
静态驱动也称直流驱动。静态驱动是指每个数码管的每一个段码都由一个单片机的 I/O 脚进行驱动，或者使用如 BCD 码二-十进位计数器进行驱动。静态驱动的优点是编程简单，显示亮度高，缺点是占用 I/O 脚多，如驱动 5 个数码管静态显示则需要 5×8=40 根 I/O 脚来驱动。

2）动态显示驱动：

动态驱动是将所有数码管的 8 个显示笔划"a,b,c,d,e,f,g,dp "的同名端连在一起，另外为每个数码管的公共极 COM 增加位元选通控制电路，位元选通由各自独立的 I/O 线控制，当单片机输出字形码时，所有数码管都接收到相同的字形码，但究竟是那个数码管会显示出字形，取决于单片机对位元选通 COM 端电路的控制，所以我们只要将需要显示的数码管的选通控制打开，该位元就显示出字形，没有选通的数码管就不会亮。

透过分时轮流控制各个 LED 数码管的 COM 端，就使各个数码管轮流受控显示，这就是动态驱动。在轮流显示过程中，每位元数码管的点亮时间为 1～2ms，由于人的视觉暂留现象及发光二极管的余辉效应，尽管实际上各位数码管并非同时点亮，但只要扫描的速度足够快，给人的印象就是一组稳定的显示资料，不会有闪烁感，动态显示的效果和静态显示是一样的，能够节省大量的 I/O 埠，而且功耗更低。

由于时间显示分和秒，只需 4 个数码管，可考虑两个 2 位 7 段发光 LED,封装和引脚图如下图所示。



3、时间报警电路

报警可采用蜂鸣器。蜂鸣器按工作原理可分为压电式及电磁式的二大类：压电式蜂鸣器主要由多谐振荡器、压电蜂鸣片、阻抗匹配器及共鸣箱、外壳等组成。它是以压电陶瓷的压电效应，来带动金属片的振动而发声；电磁式的蜂鸣器，则是用电磁的原理，通电时将金属振动膜吸下，不通电时依振动膜的弹力弹回。

蜂鸣器按是否含有驱动，又分为有源蜂鸣器和无源蜂鸣器两种。 “有源”是指蜂鸣器本身内含驱动了，直接给它一定的电压就可以响;“无源”是需要靠外部的驱动才可以响的。注意，这里的“源”不是指电源。而是指振荡源。 也就是说，有源蜂鸣器内部带振荡源，所以只要一通电就会叫。 而无源内部不带振荡源，所以如果用直流信号无法令其鸣叫，要用 2k~5k 的方波去驱动它。 无源蜂鸣器的优点是便宜、声音频率可控（可以做出“多来米发索拉西”的效果）以及在一些特例中，可以和 LED 复用一个控制口。而有源蜂鸣器的优点在于程序控制方便。

4、设计思路

定时器总是处于两种状态之一：停止定时与定时运行。在停止定时状态，转动旋转编码器就可以改变定时时间；在定时运行状态，定时器倒计时。按下旋转编码器上的按钮可以对这两种状态进行切换。

转动旋转编码器时定时时间并不是以秒为步进长度来改变时间，我们有一个标准时间数组来实现。

EEPROM 库用于保存最后使用的时间，所以每当这个装置上电时，它将记住最后一次使用的时间。

5、硬件

设计参考电路图如图 4.15 所示。

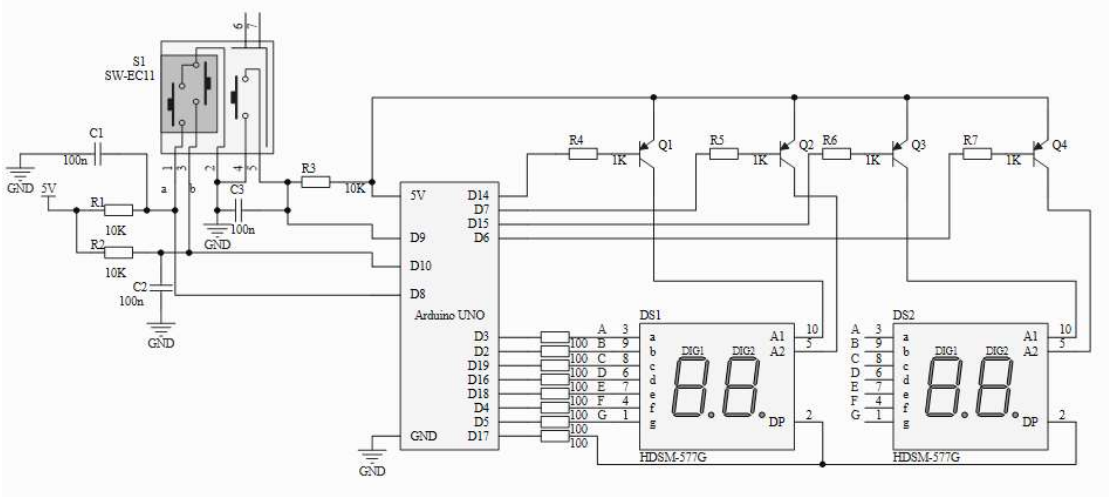


图 4.15 电路原理图

6、软件设计

参考程序为：

```
#include<EEPROM.h>
int segmentPins[] ={3,2,19,16,18,4,5,17};
int displayPins[] ={6,15,7,14};
int  times[]={5,10,15,20,30,45,100,130,200,230,300,400,500,600,700,800,900,1000,  1500,  2000,
3000};
int numTimes=19;
```

```
byte selectedTimeIndex;
int timerMinute;
int timerSecond;
int buzzerPin=11;
int aPin=8;
int bPin=10;
int buttonPin=9;

boolean stopped = true;
byte digits[10][8] ={
   //a   b   c   d   e   f   g .
   {1,1,1,1,1,1,0,0},//0
   {0,1,1,0,0,0,0,0},//1
   {1,1,0,1,1,0,1,0},//2
   {1,1,1,1,0,0,1,0},//3
   {0,1,1,0,0,1,1,0},//4
   {1,0,1,1,0,1,1,0},//5
   {1,0,1,1,1,1,1,0},//6
   {1,1,1,0,0,0,0,0},//7
   {1,1,1,1,1,1,1,0},//8
   {1,1,1,1,0,1,1,0}//9
  };

void setup()
{
for(int i=0;i<8;i++)
{
            pinMode(segmentPins[i],OUTPUT);
}
        for(int i=0;i<4;i++)
{
pinMode(displayPins[i],OUTPUT);
}
pinMode(buzzerPin,OUTPUT);
pinMode(buttonPin,INPUT);
pinMode(aPin,INPUT);
pinMode(bPin,INPUT);
selectedTimeIndex=EEPROM.read(0);
timerMinute=times[selectedTimeIndex]/100;
timerSecond=times[selectedTimeIndex]%100;
}

void loop()
{
```

```
    if(!digitalRead(buttonPin))
  {
        stopped = !stopped;
        digitalWrite(buzzerPin,LOW);
        while(!digitalRead(buttonPin)){};
        EEPROM.write(0,selectedTimeIndex);
  }
  updateDisplay();
}
void updateDisplay() //mmss
{
  int minsecs = timerMinute*100+timerSecond;
  int v = minsecs;
  for(int i=0;i<4;i++)
  {
      int digit=v%10;
      setDigit(i);
      setSegments(digit);
      v=v/10;
      process();
  }
  setDigit(5);// all digits off to prevent uneven illumination
}

void process()
{
  for(int i=0;i<100; i++)//tweak this number between flicker and blur
{
int change=getEncoderTurn();
if(stopped)
{
  changeSetTime(change);
}
  else
{
      updateCountingTime();
}
}
if(timerMinute == 0 && timerSecond == 0)
{
digitalWrite(buzzerPin,HIGH);
}
}
```

```
void changeSetTime(int change)
{
	selectedTimeIndex+=change;
	if(selectedTimeIndex<0)
	{
		selectedTimeIndex=numTimes;
	}
	else if(selectedTimeIndex>numTimes)
	{
		selectedTimeIndex = 0;
	}
	timerMinute=times[selectedTimeIndex]/100;
	timerSecond=times[selectedTimeIndex]%100;
}

void updateCountingTime()
{
	static unsigned long lastMillis;
	unsigned long m=millis();
	if(m>(lastMillis+1000) && (timerSecond>0 || timerMinute>0))
	{
	digitalWrite(buzzerPin,HIGH);
	delay(10);
digitalWrite(buzzerPin,LOW);
if(timerSecond==0)
{
		timerSecond=59;
		timerMinute--;
}
else
{
		timerSecond--;
}
lastMillis = m;
}
}

void setDigit(int digit)
{
  for(int i=0;i<4;i++)
  {
    digitalWrite(displayPins[i],(digit!=i));
  }
}
```

```
void setSegments(int n)
{
   for(int i=0;i<8; i++)
   {
      digitalWrite (segmentPins[i], !digits[n][i]);
   }
}

int getEncoderTurn()
{
   // return -1，0，or +1
   static int oldA=LOW;
   static int oldB=LOW;
   int result = 0;
   int newA=digitalRead(aPin);
   int newB=digitalRead(bPin);
   if (newA != oldA || newB != oldB)
{
   // something has changed
     if(oldA == LOW && newA==HIGH)
   {
      result = -(oldB*2-1);
   }
}
oldA = newA;
oldB = newB;
return result;
}
```

三、实验任务与要求

1、用 Arduino UNO 设计倒计时定时器，要求如下：设定倒计时时间若干（设定标准时间数组），通过旋转编码器选择，时间到时报警（上面例子中是数码管是共阳的，**要求按数码管是共阴的情况设计**）。

2、设计电路，完成相应器件的选择，制作 Arduino UNO 扩展板。

3、编制与调试倒计时定时器程序。

4、将制作的扩展板与 Arduino UNO 板组装后，进行系统联调。

四、思考题

1、考虑倒计时定时器的应用，例如如何控制定时加热，简要介绍实现的电路。

2、如何通过修改程序，消除秒钟滴答声。