

旋转 LED 显示原理介绍

如何让一列灯旋转起来感觉像稳定的字显示在空中呢？首先我们来分析下人的眼睛，其实人的眼睛非常好骗的，只要让电机扫描的快一点就行了，实际上肉眼在 24 帧/秒以上就不会看到闪烁的，所以要保证电机的速度能在一秒转 24 圈以上，这样的话人眼就觉得旋转的字很稳定很清晰的显示在空中了。

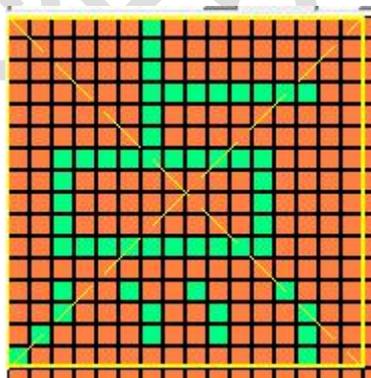
旋转 LED 旋转起来是一个圆，那么就需要有一个传感器来判断起点位置，有人用霍尔传感器，有人用红外对管，笔者觉得用红外对管来的便宜些，而且实现起来也容易。这个起点检测非常重要，单片机就是根据这个起点来判断是否要开始显示数据的。如果起点检测不到单片机就不开始显示。

如何让一列灯不断的送数据实现一个文字的显示呢？这个我们就要了解文字取模的原理了，这里以 PC2002 字幕软件为例，取一个 16*16 的中文字，见字幕选项设置：



从第一列开始向下每取 8 个点作为一个字节，如果最后不足 8 个点就补满 8 位。

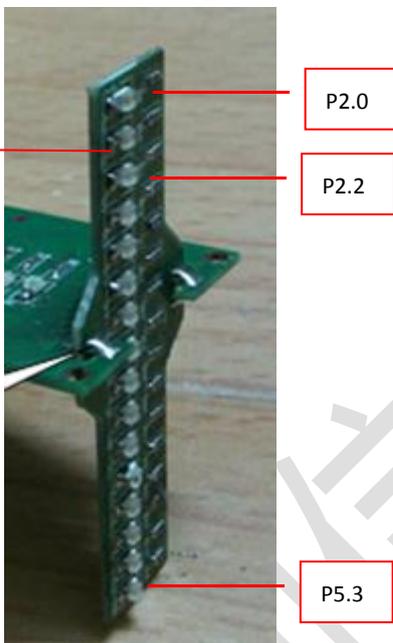
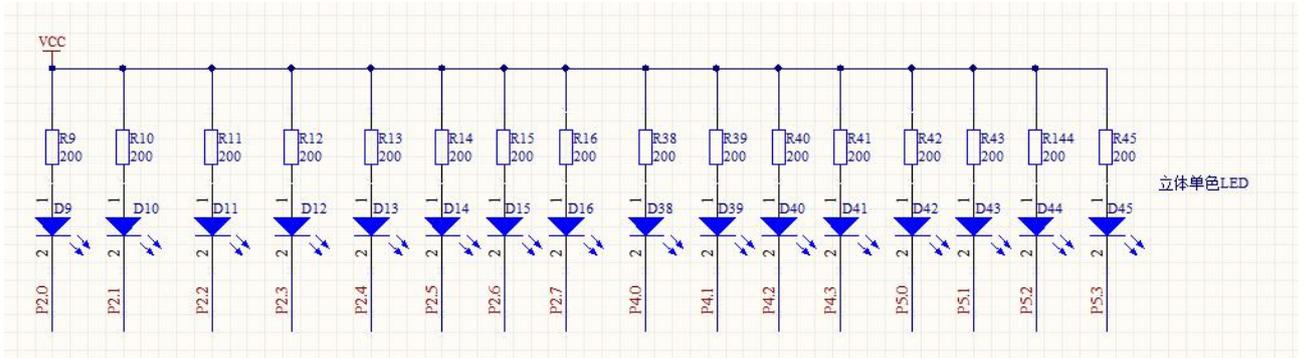
取模顺序是从低到高，即第一个点作为最低位。如*-----取为 00000001



取模后如下表：

0xFF,0x7F,0xFF,0xBF,0x3F,0xC8,0xBF,0xFB,0xBF,0xFB,0xBF,0xEB,0x80,0x9B,0xB7,0xFB,0xB7,0xEB,0xB7,0x9B,0xB7,0xFB,0x37,0xF8,0xF7,0xEF,0xF7,0x1F,0xFF,0xFF,0xFF,0xFF,/*"点",0*/

现在我们知道 16*16 取模是一列一列取的了，一列有 2 个字节，一共 16 列，所以一个 16*16 的汉字就有 32 个字节，需要占用单片机的 Code 空间 32 个字节。然后再结合硬件来分析，如下图：



一列灯 16 个 刚好对应 16*16 一个汉字的一列：2 个字节，所以把取模到的数据依次送到 P2 口 和 P4 P5 口，这里硬件中 P4 P5 组成一个字节，所以显示程序如下：

for(i=0;i<16;i++) //送 16 列 显示 这里只显示一个字。

```

{
    P2=zimo[i*2]; //送数据低位显示
    P4=(zimo[i*2+1]); //送数据高位显示,这里用了单片机 P4 和 P 口,是 LQFP48 脚才有的 IO 口
    P5=(zimo[i*2+1])>>4; //这里行和列 都是 IO 口独立驱动的 LED
    DelayUs(200); //延时让 LED 亮起来 每列延时的时间
    P2=0XFF;
    P4=P5=0XFF;
}
    
```

在什么时候送显示呢？单片机 IO 一判断到 红外接收管接收到起点信号，就开始显示，显示完 16 列后等待下一次的起点信号。这样只要电机的速度够快就会稳定的把字显示字空中了。平面的文字显示同理。

如何让一组文字不断的移动？这就需要一个字幕计数器，旋转 LED 每旋转一圈，这个字

幕计数器就加一，指向下一列，这样不断的刷新，感觉文字就在移动了，程序如下：

j 就是字幕计数器，每转一圈 j 就会加 1；

if(KEY==0) //红外接收管 判断起始位

```
{
  j++;
  if(j>672) //根据显示的字数定义改数值 672/16=42 个字 显示完 42 个字后 重新开始
  {
    j=0;
  }
  for(i=j;i<128+j;i++) //每转一圈 前进一列 这里定义一圈中同时显示 128/16=8 个字,
  {
    P2=zimo[i*2]; //送数据低位显示
    P4=(zimo[i*2+1]); //送数据高位显示,这里用了单片机 P4 和 P5 口 是 LQFP48 脚才有的
    P5=(zimo[i*2+1])>>4; //这里行和列 都是 IO 口独立驱动的 LED
    DelayUs(200); //延时让 LED 亮起来 每列延时的时间
    P2=0XFF;
    P4=P5=0XFF;
  }
}
```

最后一点是供电的问题，旋转 LED 供电问题是比较麻烦的，这里我采用了无线供电方式，经过实践论证，功率很有限，需要改进的地方还很多，电路原理是把直流转成交流，然后经过初级线圈，最后次级线圈感应得电，经过整流滤波后给旋转部分供电，直流转交流部分电路是一个自激振荡电路。动手能力强的朋友经过改造电机电刷方法实现，如果有条件制作的欢迎用此种方式。