



```
        /* 通过按位或操作，在不改变低 7 位的前提下 */
        /* 将位 7 置为高电平 */
        Segment = SegCode[value[i]] | 0x80;
    }
    else
    {
        Segment = SegCode[value[i]];
    }
    delay_ms(1);
}

/* 这里扫描的时候 location 对应位不显示，只显示其他位 */
for(j = displayCount; j > 0; j--)
{
    for(i = 0; i < 6; i++)
    {
        /* 判断当期扫描的位是不是 location 对应位 */
        if(location == i)
        {
            /* 如果是的话，不选中任何位选，也就是全灭 */
            Digital = 0xff;
        }
        else
        {
            /* 对于其他为就正常显示就好 */
            Digital = DigCode[i];
        }

        /* 这里和上面是一样的，加小数点的操作 */
        if(i == 1 || i == 3)
        {
            Segment = SegCode[value[i]] | 0x80;
        }
        else
        {
            Segment = SegCode[value[i]];
        }
        delay_ms(1);
    }
}
```

## 【代码分析】

### MVC 第一步，确定 Model 模型量

我们要同时显示 6 位数管，每个数码必须要有自己的值。我们考虑用一个数组保存每一位数码管的值。

```
u8 value[6] = {1, 2, 3, 4, 5, 6};
```

本实验要求一定有一位闪烁，我们还需要一个模型量来记录具体是哪一位在闪烁。

```
u8 location = 0;
```

### MVC 第二步，View 层代码设计

动态扫描的时候，displayCount 次正常动态扫描，displayCount 次 location 指定的位不显示。举个例子，第一位在闪烁，displayCount 的值为 5。

```
亮亮亮亮亮亮
亮亮亮亮亮亮
亮亮亮亮亮亮
亮亮亮亮亮亮
亮亮亮亮亮亮
灭亮亮亮亮亮
灭亮亮亮亮亮
灭亮亮亮亮亮
灭亮亮亮亮亮
灭亮亮亮亮亮
```

然后我们看起就是闪烁的样子了。

控制的代码最简单了，没什么好分析的。

## 【实验任务】

看懂源代码，熟悉一下 MVC 的设计思想。