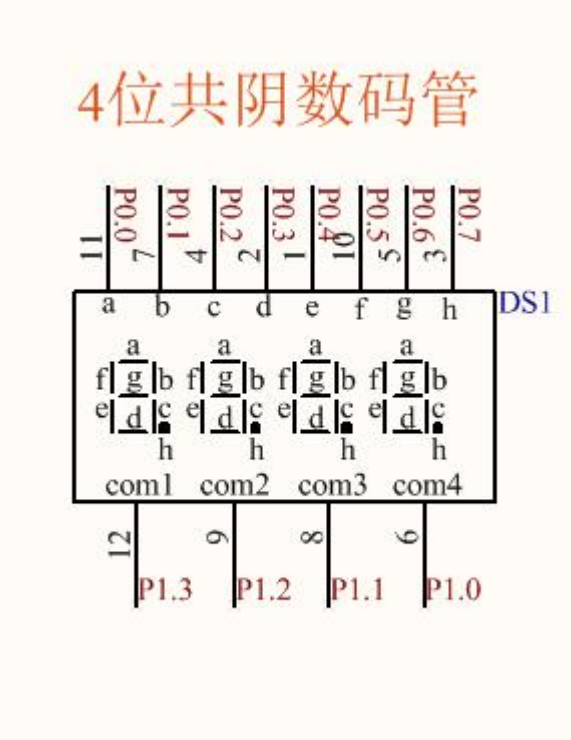
第一百一十三节： 动态扫描的数码管显示数字。

【113.1　动态扫描的数码管。】



上图 113.1.1　数码管

上一节，看到打开显示的数码管右起第 1 个（com4）和第 3 个(com2)在任意时刻显示的数字是一样的，为什么？因为四个数码管的 8 个段码 a,b,c,d,e,f,g,h 所连接的单片机 IO 口是共用的，如果把四个数码管全部打开（com1,com2,com3,com4 全部输出低电平），会发现四个数码管在任意时刻显示的四个数字也是一样的！实际应用中，要四个数码管能各自独立显示不同的数字，就需要用到“分时动态扫描”的方式。所谓分时，就是在任意时刻只能显示其中一个数码管（某个 com 输出低电平），其它三个数码管关闭（其它三个 com 输出高电平），每个数码管显示停留的时间固定一致并且非常短暂，四个数码管依次循环的切换显示，只要切换画面的速度足够快，人的视觉就分辨不出来，感觉八个数码管是同时亮的（实际不是同时亮），跟动画片“1 秒钟动态切换显示多少幅画面”的原理一样。现在编写一个程序例子，四个数码管要显示四个不同的数字“1234”，程序代码如下：

```
#include "REG52.H"

/* 注释一：
*  SCAN_TIME 是每个数码管停留显示的短暂时间。这里称为“扫描时间”。这个时间既不能太长也不能
*  太短，要调试到恰到好处。太长，则影响其它数码管的显示，会让人觉得画面不连贯不是同时亮；
*  太短，又会影响显示的亮度。具体的时间应该根据实际项目不断调试修正而得到最佳显示的数值。
*/

#define SCAN_TIME  1
```

```
void T0_time();
void SystemInitial(void) ;
void Delay(unsigned long u32DelayTime) ;
void PeripheralInitial(void) ;

void DisplayScan(void);    //数码管的动态扫描函数，放在定时中断里。

sbit P1_0=P1^0;  //右起第 1 位数码管的公共端 com4
sbit P1_1=P1^1;  //右起第 2 位数码管的公共端 com3
sbit P1_2=P1^2;  //右起第 3 位数码管的公共端 com2
sbit P1_3=P1^3;  //右起第 4 位数码管的公共端 com1

//根据原理图得出的共阴数码管编码转换表，类似于一个字库表
code unsigned char Cu8DigTable[]=
{
0x3f,  //0       序号 0
0x06,  //1       序号 1
0x5b,  //2       序号 2
0x4f,  //3       序号 3
0x66,  //4       序号 4
0x6d,  //5       序号 5
0x7d,  //6       序号 6
0x07,  //7       序号 7
0x7f,  //8       序号 8
0x6f,  //9       序号 9
0x00,  //不显示  序号 10
};

volatile unsigned char vGu8ScanTimerFlag=0;  //动态扫描的定时器
volatile unsigned int vGu16ScanTimerCnt=0;

/* 注释二：
*  vGu8Display_Righ_4，vGu8Display_Righ_3，vGu8Display_Righ_2，vGu8Display_Righ_1，这四个
*  全局变量用来传递每位数码管需要显示的数字，作为对上面应用层调用的接口变量。
*/

volatile unsigned char vGu8Display_Righ_4=1;  //右起第 4 位数码管显示的变量。这里显示“1”
volatile unsigned char vGu8Display_Righ_3=2;  //右起第 3 位数码管显示的变量。这里显示“2”
volatile unsigned char vGu8Display_Righ_2=3;  //右起第 2 位数码管显示的变量。这里显示“3”
volatile unsigned char vGu8Display_Righ_1=4;  //右起第 1 位数码管显示的变量。这里显示“4”

void main()
{
    SystemInitial();
```

```
    Delay(10000);
    PeripheralInitial();
    while(1)
    {
    }
}

/* 注释三：
*  DisplayScan 数码管的动态扫描函数，之所以放在定时中断里，是因为动态扫描数码管对时间均匀度
*  要求很高，如果放在 main 主函数中，期间稍微出现一些延时滞后或者超前执行的情况，都会导致
*  数码管出现“闪烁”或者“忽暗忽亮”的显示效果。
*/

void DisplayScan(void)
{
    static unsigned char Su8GetCode;  //从编码转换表中提取出来的编码。
    static unsigned char Su8ScanStep=1;  //扫描步骤

    if(0==vGu16ScanTimerCnt)  //定时的时间到，切换显示下一个数码管，依次动态快速循环切换显示
    {

/* 注释四：
*  在即将切换显示到下一个新的数码管之前，应该先关闭显示所有的数码管，避免因关闭不彻底而导致
*  数码管某些段位出现“漏光”，也就是数码管因程序处理不善而出现常见的“鬼影”显示情况。
*/

        P0=0x00; //输出显示先清零，先关闭显示所有的数码管
        //先关闭所有的 com 口，先关闭显示所有的数码管
        P1_0=1;  //右起第 1 位数码管的公共端 com4，“总开关”关闭，输出低电平 1
        P1_1=1;  //右起第 2 位数码管的公共端 com3，“总开关”关闭，输出高电平 1
        P1_2=1;  //右起第 3 位数码管的公共端 com2，“总开关”关闭，输出低电平 1
        P1_3=1;  //右起第 4 位数码管的公共端 com1，“总开关”关闭，输出高电平 1

        switch(Su8ScanStep)
        {
           case 1: //显示右起第 1 个数码管
                Su8GetCode=Cu8DigTable[vGu8Display_Righ_1]; //从编码转换表中提取出来的编码。
                P0=Su8GetCode; //段码端输出需要显示的编码
                P1_0=0;  //右起第 1 位数码管的公共端 com4，“总开关”打开，输出低电平 0
                P1_1=1;  //右起第 2 位数码管的公共端 com3，“总开关”关闭，输出高电平 1
                P1_2=1;  //右起第 3 位数码管的公共端 com2，“总开关”关闭，输出高电平 1
                P1_3=1;  //右起第 4 位数码管的公共端 com1，“总开关”关闭，输出高电平 1
                break;
```

```
            case 2: //显示右起第 2 个数码管
                 Su8GetCode=Cu8DigTable[vGu8Display_Righ_2]; //从编码转换表中提取出来的编码。
                 P0=Su8GetCode; //段码端输出需要显示的编码
                  P1_0=1;  //右起第 1 位数码管的公共端 com4，“总开关”关闭，输出高电平 1
                  P1_1=0;  //右起第 2 位数码管的公共端 com3，“总开关”打开，输出低电平 0
                  P1_2=1;  //右起第 3 位数码管的公共端 com2，“总开关”关闭，输出高电平 1
                  P1_3=1;  //右起第 4 位数码管的公共端 com1，“总开关”关闭，输出高电平 1
                 break;

            case 3: //显示右起第 3 个数码管
                 Su8GetCode=Cu8DigTable[vGu8Display_Righ_3]; //从编码转换表中提取出来的编码。
                 P0=Su8GetCode; //段码端输出需要显示的编码
                  P1_0=1;  //右起第 1 位数码管的公共端 com4，“总开关”关闭，输出高电平 1
                  P1_1=1;  //右起第 2 位数码管的公共端 com3，“总开关”关闭，输出高电平 1
                  P1_2=0;  //右起第 3 位数码管的公共端 com2，“总开关”打开，输出低电平 0
                  P1_3=1;  //右起第 4 位数码管的公共端 com1，“总开关”关闭，输出高电平 1
                 break;

            case 4: //显示右起第 4 个数码管
                 Su8GetCode=Cu8DigTable[vGu8Display_Righ_4]; //从编码转换表中提取出来的编码。
                 P0=Su8GetCode; //段码端输出需要显示的编码
                  P1_0=1;  //右起第 1 位数码管的公共端 com4，“总开关”关闭，输出高电平 1
                  P1_1=1;  //右起第 2 位数码管的公共端 com3，“总开关”关闭，输出高电平 1
                  P1_2=1;  //右起第 3 位数码管的公共端 com2，“总开关”关闭，输出高电平 1
                  P1_3=0;  //右起第 4 位数码管的公共端 com1，“总开关”打开，输出低电平 0
                 break;

        }

        Su8ScanStep++;
        if(Su8ScanStep>4) //如果扫描步骤大于 4，继续从第 1 步开始扫描
        {
            Su8ScanStep=1;

        }

        vGu8ScanTimerFlag=0;
        vGu16ScanTimerCnt=SCAN_TIME;
        vGu8ScanTimerFlag=1;  //启动新一轮的定时器
    }
}

void T0_time() interrupt 1
{
```

```
    DisplayScan(); //数码管的动态扫描函数

    if(1==vGu8ScanTimerFlag&&vGu16ScanTimerCnt>0)  //数码管显示切换时间的定时器
    {
        vGu16ScanTimerCnt--;
    }


    TH0=0xfc;
    TL0=0x66;
}


void SystemInitial(void)
{
    //初始化上电瞬间数码管的状态，关闭显示所有的数码管
    P0=0x00;
    P1_0=1;  //右起第 1 位数码管的公共端 com4，“总开关”关闭，输出低电平 1
    P1_1=1;  //右起第 2 位数码管的公共端 com3，“总开关”关闭，输出高电平 1
    P1_2=1;  //右起第 3 位数码管的公共端 com2，“总开关”关闭，输出低电平 1
    P1_3=1;  //右起第 4 位数码管的公共端 com1，“总开关”关闭，输出高电平 1

    TMOD=0x01;
    TH0=0xfc;
    TL0=0x66;
    EA=1;
    ET0=1;
    TR0=1;
}

void Delay(unsigned long u32DelayTime)
{
    for(;u32DelayTime>0;u32DelayTime--);
}

void PeripheralInitial(void)
{

}
```