数码管动态控制

该电路使用到数码管和时钟电路，有关电路原理参考“ 3.1.6 七段数码管（SMG）显示电路 ”和“ 3.1.4 时钟电路(CLK) ”

CPLD 静态扫描驱动数码管很简单，只要字码上送上字形码，再控制位码就可以了，但需要较多 IO 口，一位数据需要占用 9 个 IO 口，这样也造成了通过 CPLD 的总电流加大，使 CPLD 芯片发热厉害，不利于 CPLD 的工作

这一节我们使用动态扫描的方式驱动数码管

动态扫描也就是各个数码管上送相同的字码，但在不同时刻不同数码管的位码的控制信号不同，从而实现不同数码管显示不同字符，比如当字码为 S 那个时间段，位码 W4 有效，当字码为 O 那个时间段，位码 W3 有效，当这两个动作不断迅速重复时，由于视觉停留，你就看到 SO 了。

数码管 SMG4～SMG1 的 7 段笔画、小数点对应段在硬件上连接在一起。下面我们就用这 4 位动态数码管通过 CPLD 来做动态驱动实验

功能描述：通过 CPLD 用动态扫描驱动的方式来控制 4 个数码管，让他们显示 SOON

源程序：(GUIDE 光盘/samples/qt42/DYNAMICSMG/DYNAMICSMG.V)

```
//动态驱动数码管显示程序，dynamicsmg.v
//DOWNLOAD FROM WWW.HUSOON.COM
module DYNAMICSMG(SMG, W,CLK);            //定义模块结构
      output [8:1] SMG;                   //定义数码管段输出引脚,SMG=dp g f e d c b a
      output [4:1] W;                     //定义数码管选择输出引脚，W=W4 W3 W2 W1
      input CLK;                          //定义输入时钟引脚,CLK=11.0592MHz

      reg [8:1] SMG_REG;                  //定义数码管段输出寄存器
      reg [4:1] W_REG;                    //定义数码管选择输出寄存器
      reg [2:1] DISP_DAT;                 //定义显示数据代号寄存器
      reg [36:0] COUNT;                   //定义计数器寄存器

always @(posedge CLK)                     //定义 CLK 信号下降沿触发
begin
      COUNT=COUNT+1;                      //计数器值加 1
end

always @(COUNT[14:13])                                  //定义显示数据触发事件
begin
      case (COUNT[14:13])                               //选择扫描显示数据
            3'h0: DISP_DAT =2'h3;                       //显示 SMG4 数值
            3'h1: DISP_DAT =2'h2;                       //显示 SMG3 数值
            3'h2: DISP_DAT =2'h1;                       //显示 SMG2 数值
            3'h3: DISP_DAT =2'h0;                       //显示 SMG1 数值
      endcase

      case (COUNT[14:13])                               //选择哪个数码管
            3'h0: W_REG = 4'b0111;                      //选择数码管 SMG4
            3'h1: W_REG = 4'b1011;                      //选择数码管 SMG3
            3'h2: W_REG = 4'b1101;                      //选择数码管 SMG2
            3'h3: W_REG = 4'b1110;                      //选择数码管 SMG1
      endcase
end

always @(DISP_DAT)                        //显示译码输出
begin
      case (DISP_DAT)                     //选择输出数据
            2'h3: SMG_REG = 8'b01101101;  //显示 S
            2'h2: SMG_REG = 8'b00111111;  //显示 O
            2'h1: SMG_REG = 8'b00111111;  //显示 O
            2'h0: SMG_REG = 8'b00110111;  //显示 N
       default: SMG_REG = 8'b00000000;    //关掉所有数码管
```

```
        endcase
end


assign SMG=SMG_REG;                    //输出数码管译码结果
assign W=W_REG;                    //输出数码管选择

endmodule
```

操作：在 QUARTUS 中建立工程，并用上面的语句建立 verilog-HDL 文件，保存、编译，连后选定芯片 EPM7128SLC84-15,并按下表指定管脚:

	To	Location	General Function
1	CLK	PIN_83	Global Clock
2	SMG[1]	PIN_6	I/O
3	SMG[2]	PIN_79	I/O
4	SMG[3]	PIN_81	I/O
5	SMG[4]	PIN_8	I/O
6	SMG[5]	PIN_9	I/O
7	SMG[6]	PIN_4	I/O
8	SMG[7]	PIN_80	I/O
9	SMG[8]	PIN_5	I/O
10	W[1]	PIN_76	I/O
11	W[2]	PIN_77	I/O
12	W[3]	PIN_10	I/O
13	W[4]	PIN_11	I/O
14	<<new>>	<<new>>	

再编译、仿真、下载，并把排针 JP1 对应脚用跳冒插上，J2 上把跳冒插到 11.0592MHz 上，你将看到实验结果

注意：

1. 当 J2 选择 11.0592MHz 的频率时，你看到的是“ SOON ”，可能你对动态显示概念还不是很清楚，认为这不是静态的吗？你可以把 J2 上的时钟信号选择到 2048Hz 上，你就会看到 SOON 逐步动态显示，

连后你想象一下如果这个动态显示动作很快，快到你眼睛看不出是一个个显示，是不是就是上面 11.0592MHz 所显示的“ SOON ”呢？

2. 编译时出现警告：“ output pins is stuck at vcc or gnd ”，这个警告不影响电路，因为设计就是的思路就是这样，比如 DP 就是要它不显示，那么它一直为低电平，软件就认为它连到 GND 了。
3. 这里为了便于数组的使用 SMG=DP G F E D C B A
4. 这里 SOON 都采用近似的，具体建静态数码管控制部分