

实验四 七段数码管控制的设计

一、实验目的

掌握控制七段数码管显示的方法；掌握计数器的设计方法。

二、实验内容

完成 8 位十六进制数显示的数码管显示控制逻辑的设计。要求能同时稳定地显示各个位的数字，并能定义 8 位数码管哪些位要显示，哪些位不需要显示。

设计要求：

包含一个 8 位十六进制数显示模块 hexseg8

模块输入要求：

1) 2 个 4 位的输入 hex0[3:0]、hex1[3:0]，分别输入 2 个 4 位二进制数，使用 SW0-SW3 输入 hex0 所需信号，SW4-SW7 输入 hex1 所需信号。数字输入高位 (MSB, Most-Significant-Bit) 在左，低位 (LSB, Least-Significant-Bit) 在右。

2) 一个 8 位的输入 en[7:0] 表明哪一位数码管需要显示（**高电平有效**，比如只需要左边 4 位显示，右边 4 位不显示，则 en = 8'b11110000）。使用 SW8 的 8 个开关控制对应顺序的 8 个数码管的显示与否（如 SW8-3 控制左数第三个数码管 DK3 的显示与否）。

3) 输入时钟为 100MHz 的板载信号，在该模块内分频产生两个信号：一个信号频率为 1Hz，用于控制输出计数器的跳变频率；另一个信号频率为 1kHz，用于控制数码管的动态扫描。

4) 一个按键开关 S3，按下后以 1s 为周期的计数器清零，返回 8'h00 状态。

模块输出信号要求：

1) 两个 8 位的输出数码管段码 segs0[7:0]；segs1[7:0]，连接数码管，控制左四位数码管显示和右四位数码管显示。

2) 一个 8 位使能有效端 len[7:0]，连接数码管，控制数码管的亮灭。

显示要求:

若8位全部显示(en=8'b11111111), 要求左起1, 2两位实现班级(0X), 左起3, 4两位实现学号后两位(XX), 左起5, 6两位实现一个每1s增加一个数的、变化范围从00~ff的两位十六进制计数器, 在计满后自动回00。最后两位显示拨码开关输入的两个十六进制数, 前四位拨码开关对应数码管DK7, 后四位拨码开关对应数码管DK8。左数第四位和左数第六位右下角的小数点亮起。

例如: 9班37号同学在拨码开关上输入0xff时数码管应有如下显示:

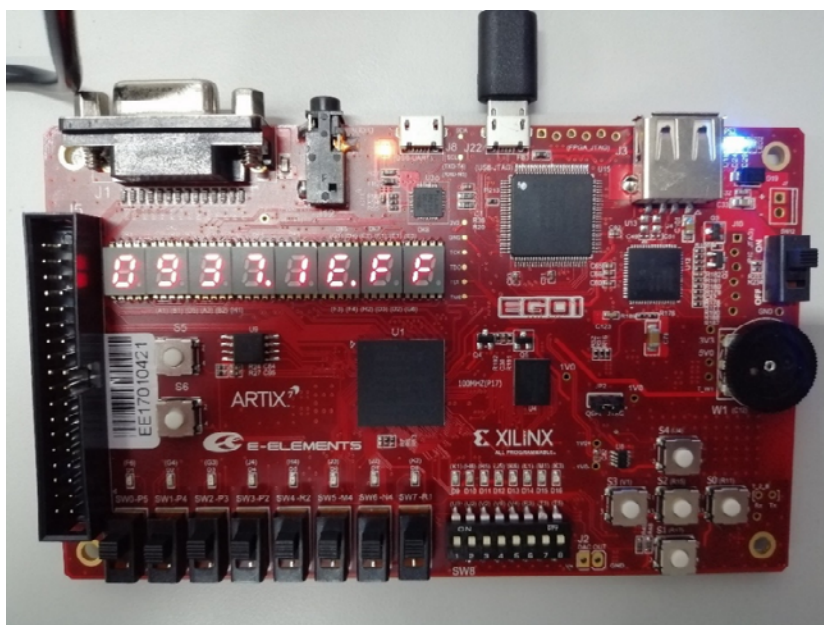





图4-1 七段数码管管脚原理图

如果将en的某些位置0, 则该位置的数码管不亮。(即使5、6两位不亮, 计数器仍然保持计数, 不需要计数器停止)

按照以上要求完成设计, 适当划分子模块, 如分频模块、译码器模块(将1位十六进制数转换成数码管显示信号)、按键消抖模块等, 并参照管脚分配表完成约束文件。

仿真波形要求:

请自行编写仿真文件, 用于模块功能测试。要求产生100MHz的输入信号, 实现以下波形的仿真和测量:

- 1) 分频器分频精度: 用仿真视图上方的   选中1kHz信号lclk的两个相邻的上升沿, 用  工具进行标示, 比较两个标尺时间之间的间隔是否为准确的1ms? 如果不

是，请考虑分频模块是否正确？

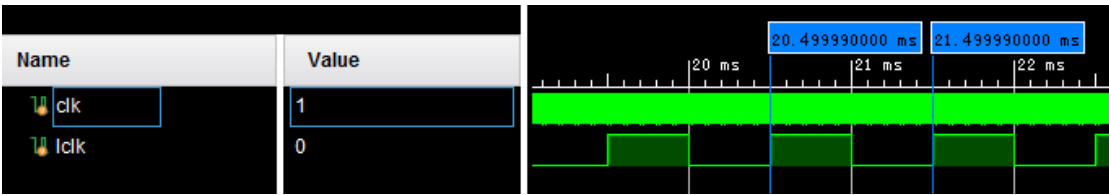


图4-2 七段数码管管脚原理图

2) 计数器的计数原理，按下图将用于计数产生1kHz的计数器变量divcount2的值显示出来(图中已抹去)，请思考在计数器电路中非阻塞赋值所导致的信号翻转位置（何时由0变1？或由1变0？）。

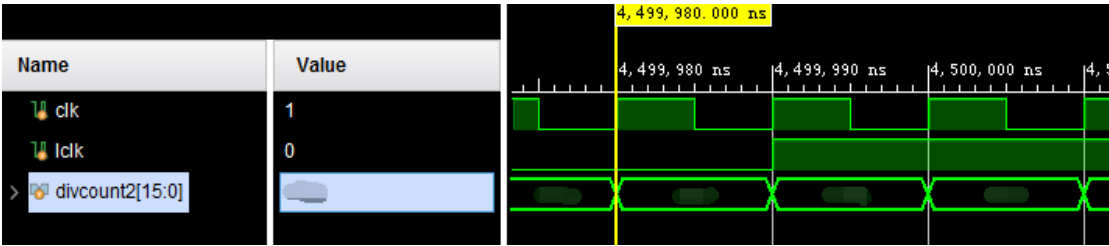


图4-3 七段数码管管脚原理图

工程文件要求：

工程文件名称：hexseg8； 主模块文件名：hexseg8.v；
仿真文件文件名：hexseg8_sim.v； 约束文件文件名：hexseg8.xdc。

管脚分配：

三、相关知识

1、分频电路的verilog描述

EG01搭载一个100MHz的时钟芯片，输出的时钟信号直接与FPGA全局时钟输入引脚（P17）相连。如需使用其他频率的时钟，可以通过计数器来设计简单的分频电路，下面展示的是一段分频器的代码。分频器一般用作将较高频率的时钟降频为低频时钟，可以通过记录接收到的高频时钟的上升沿个数计数,在计到一定上升沿数后对计数器清零，同时翻转某一1位信号，得到降频的时钟信号。

该always块中有两个电路并行。先描述了一个记满100000次上升沿后计数器清零的计数器电路；再描述了一个当计数器值为99999时，翻转reg型变量clk1(在声明该变量时应

该声明一个初值，如1'b0)的电平。

```
always @ (posedge clk)
begin
    if(divcount1==17'd99999)
    begin
        divcount1<=17'd0;
        clk1<=~clk1;
    end
    else divcount1<= divcount1+1;
end
```

该模块的仿真结果如下图所示，请思考，这个分频得到的时钟的时钟周期是多少？
时钟频率是多少？

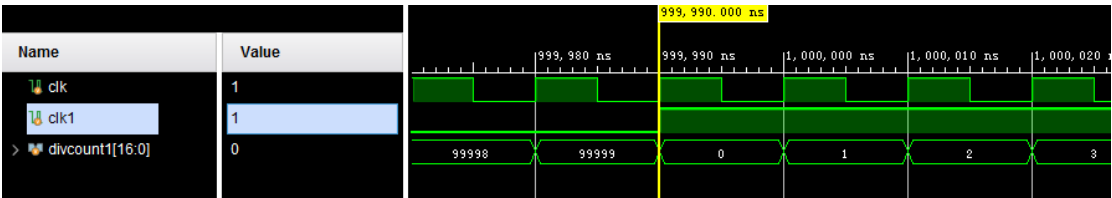


图4-4 七段数码管管脚原理图

2、数码管电路及其控制

实验板上的数码管是8个共阴极的七段数码管(外加小数点显示)。其原理图如下：

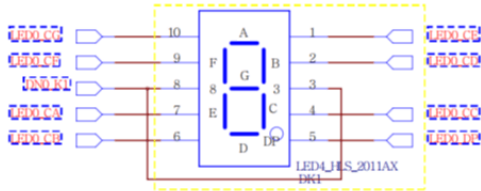


图4-5 七段数码管管脚原理图

一片数码管由7段显示LED，1个小数点和1对控制其整体亮灭的引脚组成。引脚LEDX-CA代表着这个引脚控制A段的亮灭，也就是8字型的最上端的横，其余依此类推。DN0-K1控制DK1数码管是否亮。

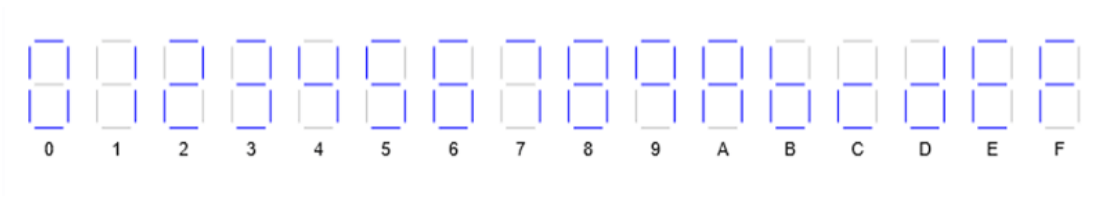


图4-6 七段数码管显示数字示意图

使得某一段LED亮，需要在对应管脚上提供高电平信号。例如若想显示7，则A，B，C

三段应为高电平，其他段为低电平，若不需要显示小数点，则提供给A[~]G，DP的八位高低电平为8'b11100000,即8'he0。使整个数码管在给非空的信号下能够发亮，需要给对应的DN0-KX端提供高电平（高电平使能）。

使用大量数码管时，通常不会将每一个数码管的每一个引脚单独连接在芯片的管脚上，而是将4个数码管连成一组，共用A[~]G和DP八个管脚，DN0-K1[~]DN0K4四个管脚可以独立控制。这时如果想让四个数码管“看上去”常亮，可以采用循环显示的方法，让DK0亮较短时间内后，切换到DK1亮，同时公共的8个引脚上提供DK1的数码管显示信号，以此类推，从而实现一个视觉难以分辨的“流水灯”，就可以观察到常亮的现象。

本实验建议切换频率为1kHz（1ms切换一次），过低的频率会有明显的流水灯现象。

数码管的管脚分配如下，其中名称后面的0或1代表共用端口的左面四个数码管组成的一组 and 右面四个数码管组成的一组：

表4-1 数码管管脚

名称	原理图标号	FPGA IO PIN
A0	LED0_CA	B4
B0	LED0_CB	A4
C0	LED0_CC	A3
D0	LED0_CD	B1
E0	LED0_CE	A1
F0	LED0_CF	B3
G0	LED0_CG	B2
DP0	LED0_DP	D5
A1	LED1_CA	D4
B1	LED1_CB	E3
C1	LED1_CC	D3
D1	LED1_CD	F4
E1	LED1_CE	F3
F1	LED1_CF	E2
G1	LED1_CG	D2
DP1	LED1_DP	H2
DN0_K1	LED_BIT1	G2
DN0_K2	LED_BIT2	C2
DN0_K3	LED_BIT3	C1
DN0_K4	LED_BIT4	H1
DN1_K1	LED_BIT5	G1
DN1_K2	LED_BIT6	F1
DN1_K3	LED_BIT7	E1
DN1_K4	LED_BIT8	G6

3、按键开关消抖实现

板上使用的是机械按键开关，如板子右下角的五个按键。按键的电路原理图如下所示：

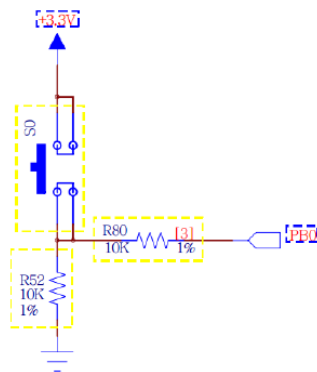


图4-7 按键开关原理图

可以看出，按下开关使开关闭合时，对应的端口PB0为高电平，断开时为低电平。

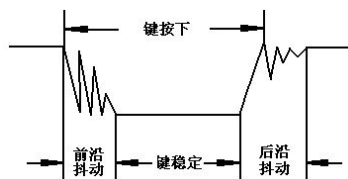


图4-8 按键抖动示意图（按下的电平与本实验不同）

机械按键由于按下时的机械振动，会导致输入信号是一个连续抖动的信号。为了避免信号的抖动，需要进行防抖动处理。使用verilog编写按键防抖的思路是用1kHz的时钟进行采样，如果出现高电平则让某一计数器加1，如果出现低电平则让计数器清零。当时钟上升沿处连续出现高电平，计数器总能记到一定的数值，当增大到某一数值时认为之后的信号是稳定的，此时让某一信号翻转成1，该信号即为消抖的信号。计数器在这里用作延时，这是一种延时消抖的方式。

本实验建议使用1kHz的信号采样。由于抖动时间大概在5~10ms范围，于是计数器的最大数值可以设置为10。

本实验使用的S3开关的引脚是V1。