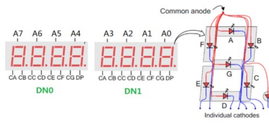
实验原理

1、数码管显示原理

下图是Minisys开发板上7段数码管单个数码管的组成，8个数码管分成2组，每组4个。以最右侧的为例，单个数码管有7个段及小数点，每个段是一个LED灯，LED灯的阳极接到一起形成共阳极即`Common anode`，LED灯的阴极接到段控制信号CA、CB、CC、…、CG、DP，其中，CA是A段的阴极，CB是B段的阴极，以此类推。8个数码管相同段的段选信号接在一起，比如8个数码管的A段都接到了CA信号。



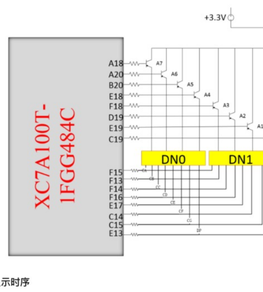
每一个数码管上方都有一个使能信号也叫位选信号，控制整个数码管的开关，比如A0信号控制最右侧的数码管。A0通过一个反相器接到数码管共阳极上，当A0位0的时候，整个数码管打开，显示才会受到CA…CG这几个信号的驱动，如果CA为0，A段LED管导通点亮。

下图列出了数码管显示0到F时点亮的段。比如显示数字0的时候，除了中间的G段外其他的段都被点亮了，而数字1只点亮了B段和C段。

每个待显示字符的编码计算方法如下：表格第一列是待显示的字符，CA-DP是段信号，最后一列是段信号组合(CA,CB,...,DP)得到的十六进制编码。以字符 `0` 和 `1` 为例：字符 `0` 只有CG和DP段不亮，对应是段信号是1，其他段信号是0；字符 `1` 只有CB和CC段亮，对应段信号是0，其他段是1。其他字符的段信号取值需自行补充完成。

```
// 七段数码管显示编码表（共阴极）
// 字符 | CA | CB | CC | CD | CE | CF | CG | DP | 十六进制
// -----|----|----|----|----|----|----|----|----|--------
//   0  |  0 |  0 |  0 |  0 |  0 |  0 |  1 |  1 | 8'h03
//   1  |  1 |  0 |  0 |  1 |  1 |  1 |  1 |  1 | 8'h9F
```

下图是数码管与主芯片的连接方式，A7~A0 是8个驱动三极管，DN0和DN1是2个数码管器件，位选信号接在上半部分的A18-C19引脚，A18连接的最左边的数码管，依次类推。段选信号CA~CG、DP接在下半部分的F15-E13引脚，F15连接的是CA段，依次类推。位选信号和段选信号都是低电平有效。在编写约束文件的时候需要参照此电路连接图。

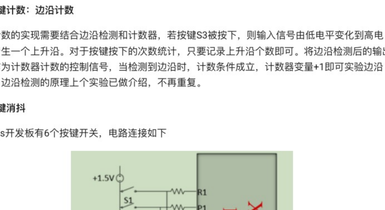


2、数码管动态显示时序

要想让8个数码管显示不同的数字，使能信号（A0-A7）和段信号（CA…CG）需依次被驱动。举个例子，如果想在数码管A0上显示数字3而数码管A1上显示数字9，先把 CA…CG 设置为显示数字3，并拉低A0信号，再把CA…CG设置为显示数字9并拉高A0拉低A1。每次只能打开一个数码管，如果同时打开，数码管显示的内容完全一样，等同于单个数码管。

从现象看8个数码管同时显示，实际数码管是轮询工作，每一时刻只有一个数码管在亮，类似于流水灯的实验，当流动的频率足够快，由于人眼的视觉暂留效应，就看不出闪烁了。数码管刷新速度建议设置为2ms刷新一次。

故需要很好地控制位使能信号A7-A0，并在对应位选信号有效时，将CA…CG信号切换为待显示的值。下图为数码管动态显示"10201508"的位控制信号led_en和段控制信号led_cx的时序，其中，led_en[7:0]分别接数码管A7-A0信号，led_cx[7:0]分别接数码管{CA,CB,CD...DP}。led_en和led_cx的变化周期建议是1-2ms，即动态扫描频率500-1000Hz。Verilog实现的电路能够实现图示的时序，就能正常控制数码管显示。



本实验功能稍多，同学们在实现的时候，应先做一下功能划分，从基础的功能开始实现，而不要一开始就把所有的功能一股脑写到一个文件，功能越多越容易出错也越难debug。

建议大家把数码管控制器的核心控制逻辑作为一个单独的模块，按照以下的接口实现，先能稳定地显示 `0x12345678`。以下模块本身不处理显示的数据，只管接收数据。后续实验还会用到数码管，对模块做一个好的封装方便后期的复用。

```
module led_ctrl_unit (
    input wire rst,
    input wire clk,
    input wire [31:0] display,  // 待显示的8个十六进制字符
    output reg [7:0]  led_en,   // 位选信号
    output reg [7:0]  led_cx    // 段选信号
);

endmodule
```

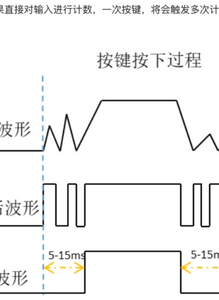
根据以上接口完成数码管核心控制器的逻辑后，再在顶层模块将 `0x12345678` 接到 `display`，其他信号做对应的处理。先稳定地显示静态数据，再添加使能信号控制8个数码管的亮灭，再添加其他模块实现按键计数、消抖、十进制计数器，然后在顶层模块例化这些子模块，把实验要求显示的数据接到 `display` 即可。

3、按键计数：边沿计数

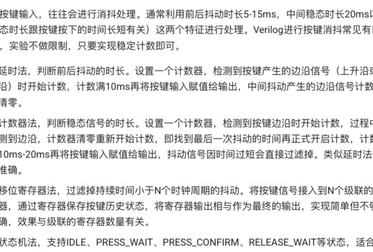
按键计数的实现需要结合边沿检测和计数器，若按键S3被按下，则输入信号由低电平变化到高电平，产生一个上升沿，对于按键按下的次数统计，只要记录上升沿个数即可。将边沿检测后的输出信号作为计数器计数的控制信号，当检测到边沿时，计数条件成立，计数器变量+1即可实现输边沿计数。边沿检测的原理上个实验已做介绍，不再重复。

4、按键消抖

Minisys开发板有6个按键开关，电路连接如下



对于实验中用的按键开关和拨码开关，属于机械部件，往往存在抖动，导致输入的信号有很多干扰，如下图所示。如果直接对输入进行计数，一次按键，将会触发多次计数，大家可以先尝试无消抖的计数。



对于按键输入，往往会进行消抖处理。通常利用前后抖动时长5-15ms，中间稳态时长20ms以上（稳态时长跟按键按下的时间长短有关）这两个特征进行处理。Verilog进行按键消抖常见有以下方法，实验不做限制，只要实现稳定计数即可。

- 延时法，判断前后抖动的时长。设置一个计数器，检测到按键产生的边沿信号（上升沿或下降沿）时开始计数，计数满10ms再将按键输入赋值给输出，中间抖动产生的边沿信号计数器不清零。
- 计数器法，判断稳态信号的时长。设置一个计数器，检测到按键边沿时开始计数，过程中每检测到边沿，计数器清零重新开始计数，即找到最后一次抖动的时间再正式开启计数，计数满10ms-20ms再将按键输入赋值给输出，抖动信号因时间过短会直接过滤掉。类似延时法但更准确。
- 移位寄存器法，过滤掉持续时间小于N个时钟周期的抖动。将按键信号接入到N个级联的寄存器，通过寄存器保存按键历史状态，将寄存器输出相与作为最终的输出，实现简单但不够准确，效果与级联的寄存器数量有关。
- 状态机法，支持IDLE、PRESS_WAIT、PRESS_CONFIRM、RELEASE_WAIT等状态，适合做短按/长按识别。