奋斗的小孩之 altera 系列

第二十六篇 单独按键消抖

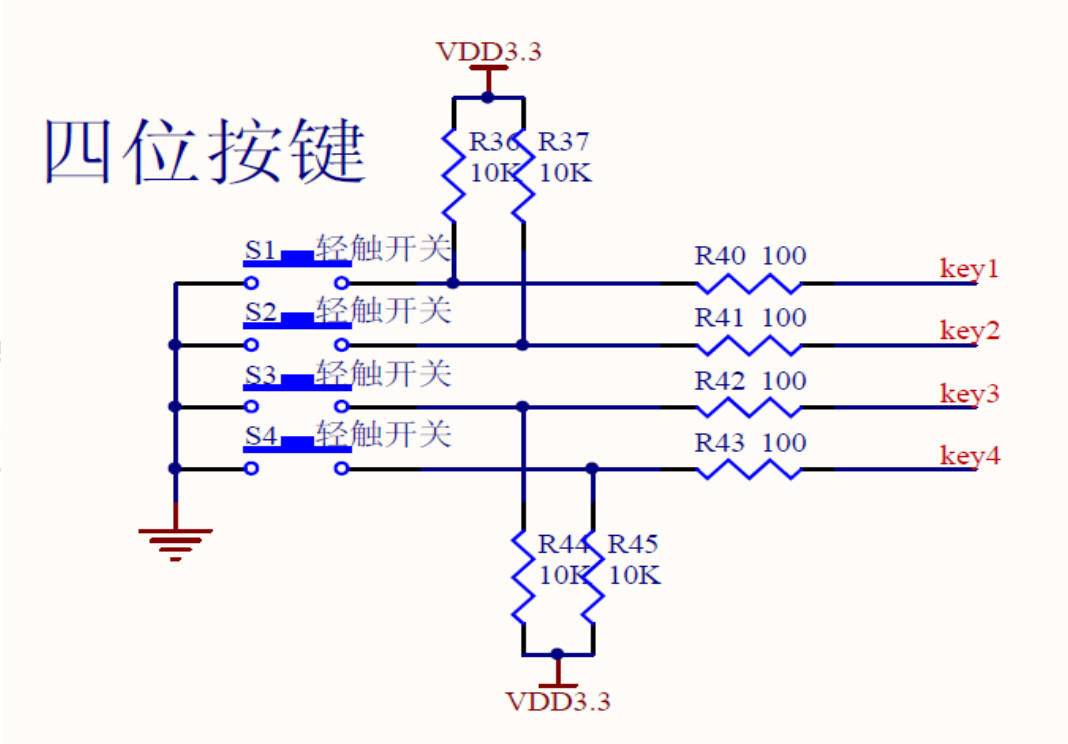
对于每一个的小实验，我们都可以把它看作是一个小项目，逐步的去分析，设计，调试，最后完成功能。下面我们就开始我们的“小项目”。

项目名称：单独按键消抖

具体要求：消除按键按下以及抬起时所带来的抖动。

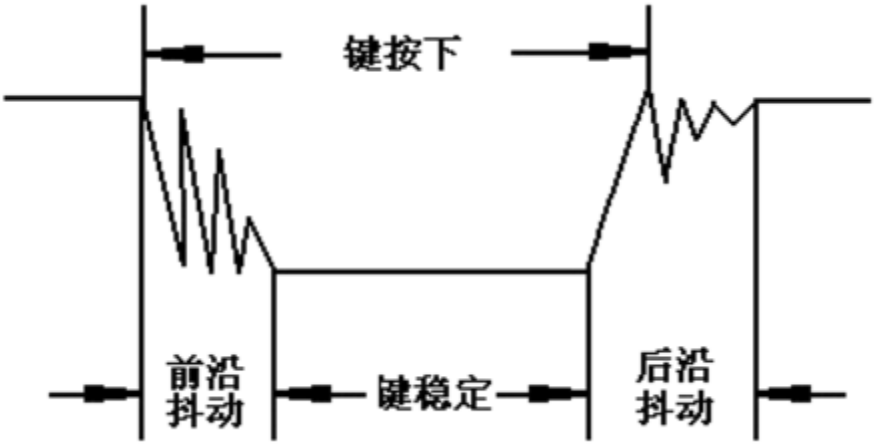
项目分析：

1. 按键电路



2. 抖动的产生

通常的按键所用开关为机械弹性开关，当机械触点断开、闭合时，由于机械触点的弹性作用，一个按键开关在闭合时不会马上稳定地接通，在断开时也不会一下子断开。因而在闭合及断开的瞬间均伴随有一连串的抖动。



3. 按键抖动带来的危害

键抖动会引起一次按键被误读多次。为确保 CPU 对键的一次闭合仅作一次处理，必须去除键抖动。在键闭合稳定时读取键的状态，并且必须判别到键释放稳定后再作处理。

4. 抖动的一些参数

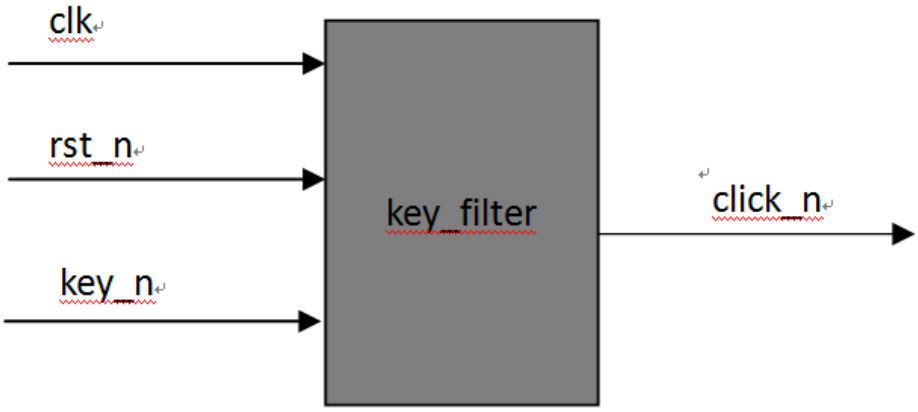
抖动时间的长短由按键的机械特性决定，一般为 5ms～10ms。这是一个很重要的时间参数，在很多场合都要用到。按键稳定闭合时间的长短则是由操作人员的按键动作决定的，一般为零点几秒至数秒。

5. 解决办法

一是延时重采样；二是持续采样。从理论上来说，延时（如

10ms）重采样的准确率肯定低于持续采样。笔者采用持续采样。

架构图如下：

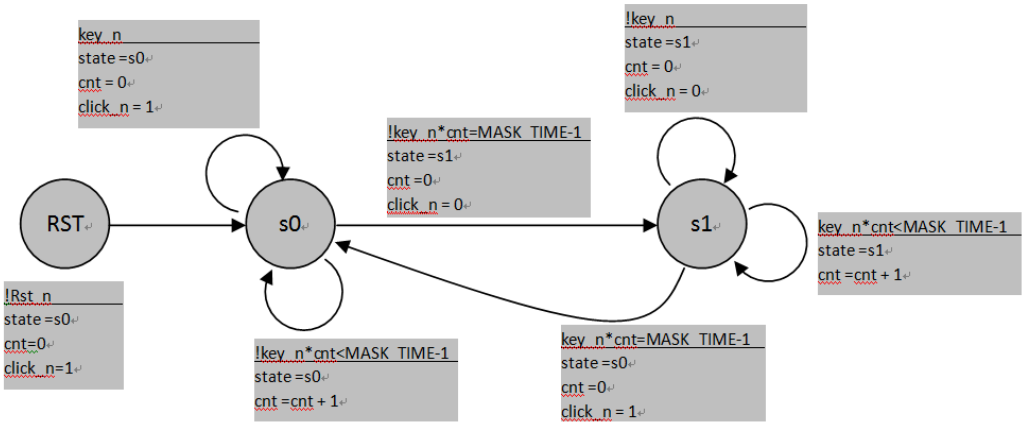


key_n：带有抖动的低电平有效的按键输入（按键按下为低电平）

click_n：滤除抖动之后的低电平有效的按键波形

系统设计：

1. 工程的名称：key_filter
2. 状态转移图如下：



MASK_TIME ： 持续采样的时间（笔者选择为 10ms）

设计代码如下：

```
/*
模块名称：key_filter
模块功能：消除按键按下以及抬起时所带来的抖动。
编写时间：2016-08-30
作者：至芯科技----奋斗的小孩
邮箱：zxopenhxs@126.com
*/
module key_filter (clk, rst_n, key_n, click_n);

    input clk;
    input rst_n;
    input key_n;

    output reg click_n;

    parameter MASK_TIME = 500_000;//驱动时钟为 50M，10ms
                                  //为 500_000 个周期
```

```
reg [18:0] cnt;
reg state;

localparam s0 = 1'b0,
           s1 = 1'b1;

always @ (posedge clk or negedge rst_n)
   begin
      if (!rst_n)
         begin
            click_n <= 1'b1;
            cnt <= 19'd0;
            state <= s0;
         end
      else
         begin
            case (state)
               s0 : begin
                     if (key_n == 1'b0)
                        begin
                           if (cnt < MASK_TIME - 1)
```

```
                    begin
                        cnt <= cnt + 1'b1;
                    end
                else
                    begin
                        state <= s1;
                        cnt <= 19'd0;
                        click_n <= 1'b0;
                    end
            end
        else
            begin
                click_n <= 1'b1;
                cnt <= 19'd0;
                state <= s0;
            end
    end

s1 : begin
        if (key_n == 1'b1)
            begin
```

```
                if (cnt < MASK_TIME - 1)
                    begin
                        cnt <= cnt + 1'b1;
                    end
                else
                    begin
                        click_n <= 1'b1;
                        state <= s0;
                        cnt <= 19'd0;
                    end
            end
        else
            begin
                click_n <= 1'b0;
                cnt <= 19'd0;
                state <= s1;
            end
    end

default : state <= s0;
```

```
                endcase

            end

        end


endmodule
```

激励代码如下：

```
/*
模块名称：key_filter_tb
模块功能：为 key_filter 模块提供激励信号
编写时间：2016-08-30
作者：至芯科技----奋斗的小孩
邮箱：zxopenhxs@126.com
*/
`timescale 1ns/1ps


module key_filter_tb;


    reg clk;
    reg rst_n;
    reg key_n;
```

```
wire click_n;


initial begin
   clk = 1;
   key_n = 1;
   rst_n = 0;
   #200.1
   rst_n = 1;


   #200
   key_n = 0;
   #10
   key_n = 1;//模仿按键按下时的抖动
   #20
   key_n = 0;
   #80
   key_n = 1;


   #200
   key_n = 0;
```

```
    #400//按下的时间超过滤除抖动的时间

    key_n = 1;
    #10
    key_n = 0;//模仿按键抬起时的抖动
    #20
    key_n = 1;
    #80
    key_n = 0;

    key_n = 1;

    #800 $stop;
end

always #10 clk = ~clk;//本地晶振为 50MHz

key_filter
    #(
   .MASK_TIME(5)//仿真时,将滤除抖动的时间改成 5 个时钟周期
```

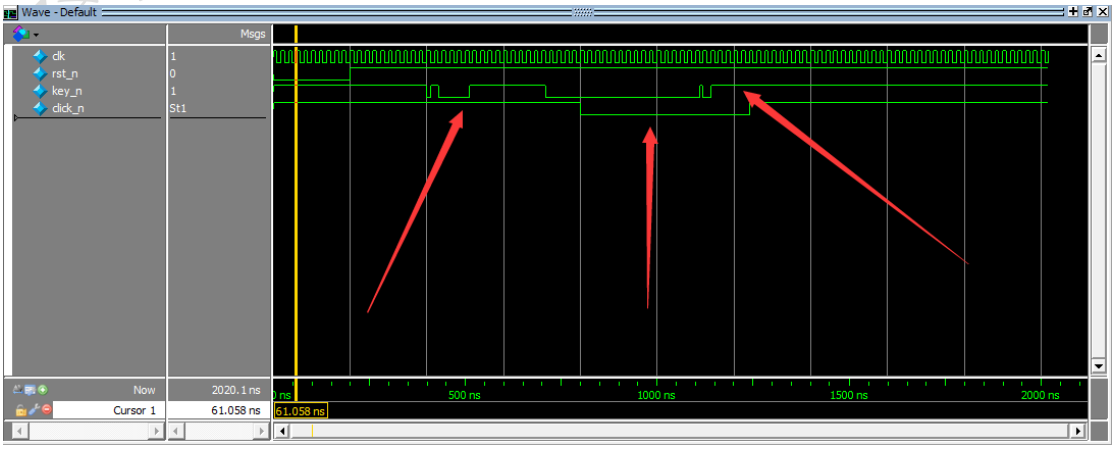
```
    )
    key_filter_dut(
        .clk(clk),
        .rst_n(rst_n),
        .key_n(key_n),
        .click_n(click_n)
    );

endmodule
```

解析：

模仿按键抖动时，抖动的时间（低电平或者高电平的持续时间）一定不要超过，否则将会被认为按键按下或者抬起。模仿按键真正按下或者真正抬起时，给予的时间一定要超过抖动（低电平或者高电平的持续时间）的时间。

仿真波形如下：



本次设计成功地将按键按下以及抬起时的抖动滤除。

如果本地晶振和笔者的设计不同，请自行更改设计，以保证设计的正确性。如果还是有不明白的读者可以发邮件到我邮箱或者加群询问。

制作人:奋斗的小孩　　fpga 交流群:282124839