PDS 下按键消抖实验

1 文档简介

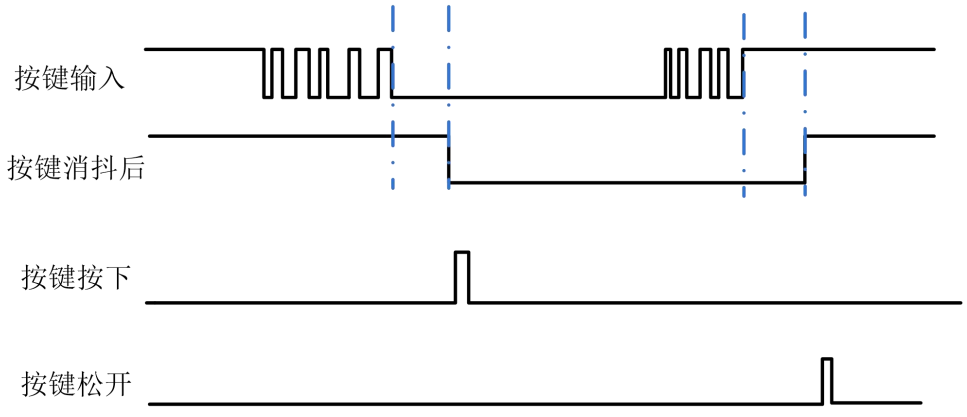
本文主要讲解按键消抖原理及程序编写，程序实现按键按下后数字加 1，并在 led 出来，通过 PDS 软件编译调试。

2 实验环境

- Windows 10 64 位
- PDS
- 开发板

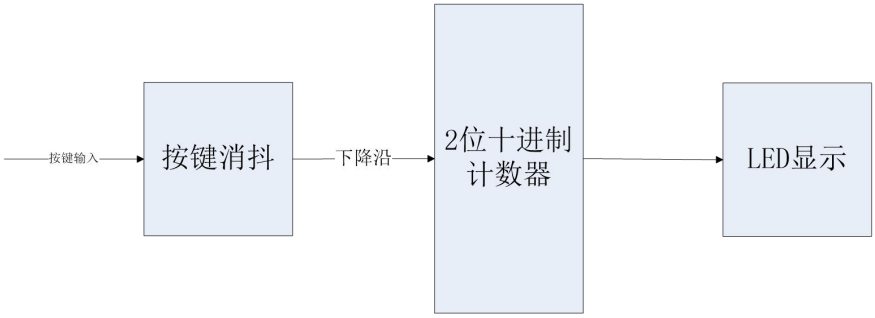
3 实验原理

按键做为基本的人机输入接口，在很多电子设计中都能见到，由于机械特性，在按键按下或松开的时候，按键输入值是有抖动的，无论按下去是多平稳，都难以消除抖动，按键消抖方式有很多，本实验主要是通过 FPGA 计时来消抖。实验中设计了一个计数器，当按键输入有变化时，计时器清零，否则就累加，直到加到一个预定值（例如 10ms），就认为按键稳定，输出按键值，这样就得到以后没有抖动的按键值。由于在很多地方需要用到按键下降沿或上升沿的检测，按键消抖模块直接集成了上升沿和下降沿检测的功能。



4 程序设计

如下图所示，通过按键消抖后，在按键按下时，十进制计数器加 1，通过数码管译码扫描后显示出来。



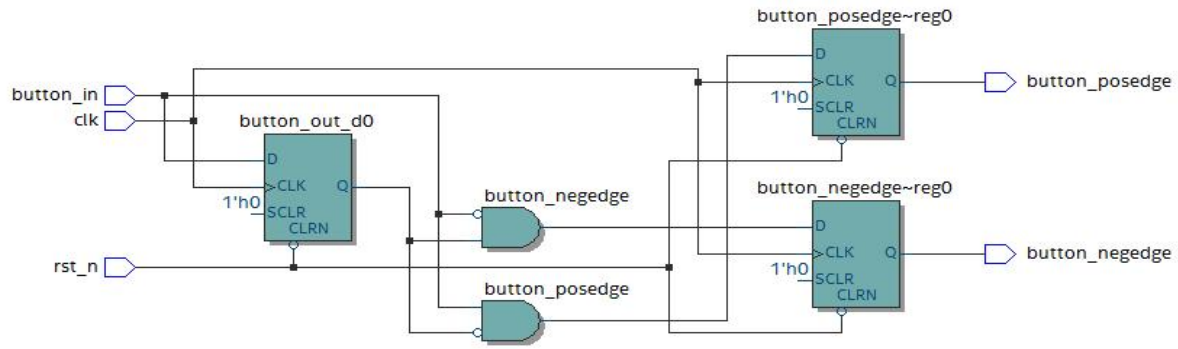
按键消抖部分的原理在上节已经讲过，按键消抖部分代码写的非常精炼，阅读起来稍显费解，建议结合仿真波形去读代码。在提供的例程文件下的 src 文件夹中同时提供了仿真文件 key_debounce_tb.v，可以通过添加仿真文件来进行仿真观察代码中信号的变化

信号名称	方向	说明
clk	in	时钟输入
rst_n	in	异步复位输入，低复位
button_in	in	按键输入
button_posedge	out	消抖后按键上升沿，高有效，1 个时钟周期
button_negedge	out	消抖后按键下升沿，高有效，1 个时钟周期
button_out	out	消抖后按键输出

按键消抖模块（ax_debounce）端口

LED 显示部分在本章不做说明，就例程中按键消抖模块“ax_debounce”模块做一些讲解，模块中通过了两级 D 触发器来寄存键值，只有当键值稳定时才将键值输出。我们可以看到在 assign 赋值语句中有一条“assign a_reset=（DFF1 ^ DFF2）”，学过数字电路的应该都知道“^”是异或运算符，运算符两边相同运算结果为 0，不同运算结果为 1。在程序中 DFF1 和 DFF2 比较运算后的值通过“assign”赋给“a_reset”表示比较锁存键值的前后两级寄存器的值是否一致，只有前后两级寄存器的值一致，也就是 a_reset 的值为 0 时才表示当前锁存的键值没有变化。当计数器累加到“TIMER_MAX_VAL”，表示锁存的键值已经稳定可以输出。另外在模块中我们可以看到“{.....}”符号，要注意这可不是大括号，这表示位拼接运算符，其作用是将运算符内的两位，或是多位信号拼接在一起，具体用法请参考例程。

最后，程序中需要说明的是“button_posedge”和“button_negedge”两个输出信号，这是一种常用的上升沿和下降沿的采集方法，其描述的 RTL 视图如下：

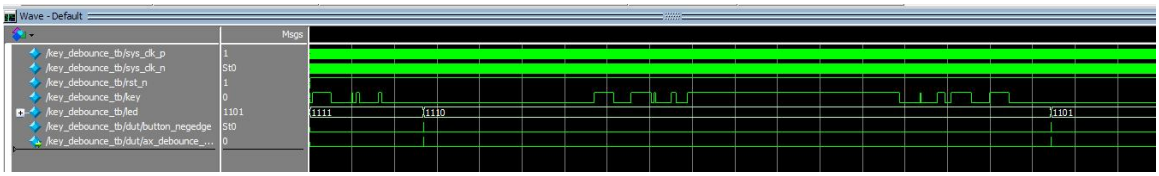


当然还有其他的边沿检测电路的描述方法，但是其基本原理都是在逻辑时序电路里先将需要检测的信号作为输入非阻塞赋值给一个自定义寄存器，通过判断前后两级寄存器的值来判断是上升沿或是下降沿，由 0 ->1 变化是上升沿，由 1 -> 0 变化是下降沿；

5 Modelsim 仿真

$random 生成随机数模拟按键抖动，按下按键对应输出二进制数据加一。按键的按下和释放经消抖后会得到一个稳定的下降沿和上升沿。仿真结果和部分仿真文件如下图所示：

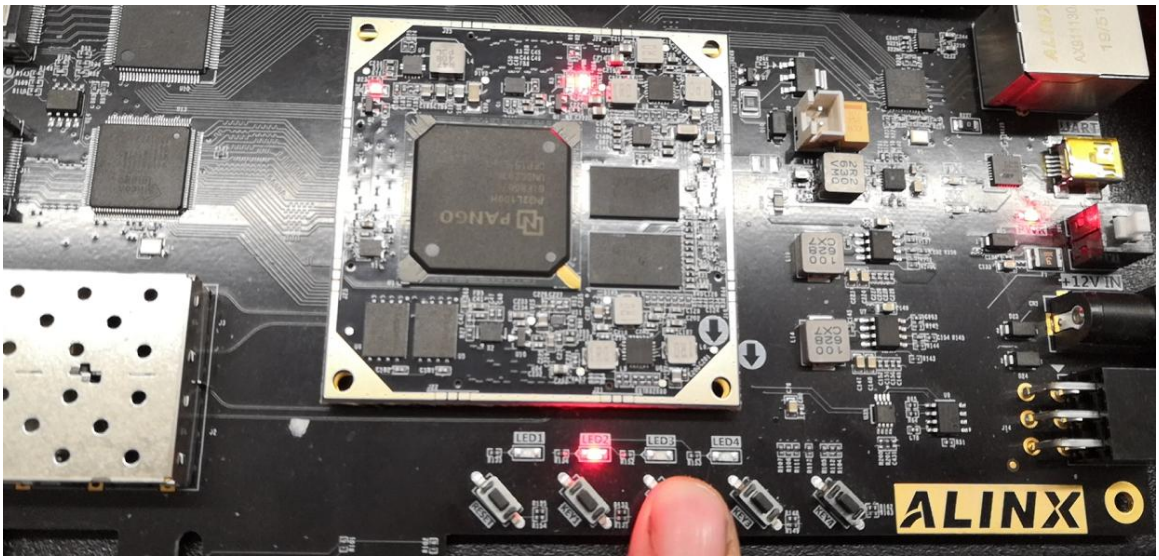
```
`timescale 1ns/1ns
module key_debounce_tb();
reg sys_clk;
reg rst_n;
reg key;
wire  button_negedge;
wire  button_posedge;
wire[3:0] led;
initial
begin
    sys_clk = 1'b0;
    rst_n = 1'b0;
    key = 1'b1;
    #100; rst_n = 1'b1;
    #2000; key = 1'b0;
    #({$random} %1000);
    key = ~key;
    #({$random} %1000);
    key = ~key;
    #({$random} %1000);
    key = ~key;
    #({$random} %1000);
    key = ~key;
    #({$random} %10000000);
    key = ~key;
    #({$random} %10000000);
    key = ~key;
    #({$random} %10000000);
    key = ~key;
    #({$random} %10000000);
    key = ~key;
    #({$random} %10000000);
    key = ~key;
    #({$random} %10000000);
    key = ~key;
    #({$random} %10000000);
    key = ~key;
    #({$random} %10000000);
    key = ~key;
    #({$random} %10000000);
    key = 1'b0;
    #1000000000;
    key = 1'b1;
```



button_negedge 为按键经过消抖后按键的下降沿，button_posedge 为按键经过消抖后按键的上升沿。

6 实验现象

开发板上电后下载程序，按下“KEY2”按键，可以看到 4 个 LED 会变化，对应二进制数据，按一次加一，如果不经过消抖，是无法实现按一次加一的。



AXP100 开发板操作