**湖 北 大 学**

计算机与信息工程学院

2023 —— 2024学年度

第 一 学期

**学 生 实 验 报 告 册**

学生姓名：

班 级： 通信工程2101班

学 号： 202131116030035

课程名称： EDA应用设计

任课老师： 周艳玲

学生实验守则

1、学生在规定的时间内进行实验，不得无故缺席或迟到。

2、学生在每次实验前对排定要做的实验应进行预习，并按要求作好预习报告。

3、每次实验前，必须交上次实验报告和本次实验预习报告，并经指导教师提问、检查同意后，才可进行本次实验。

4、学生进入实验室指定位置后，首先根据仪器清单核对自己使用的仪器是否有缺少或损坏，发现问题及时向指导教师报告，严禁擅自动用别组仪器。

5、实验时必须有实事求是、严肃认真的科学态度，严格遵守仪器操作规程和注意事项。

6、实验完毕应将实验数据交给指导教师检查，合格后，整理复原好仪器设备，方可离开实验室。

7、保持实验室肃静和整洁，不得大声喧哗，乱丢垃圾和吃东西。

8、学生在实验过程中，由于不遵守操作规程或未经许可，擅自进行实验而造成事故、损坏仪器设备，应及时报告，并填写损坏清单，按院有关规定进行赔偿。

实 验 报 告 单

**实验名称： 按键消抖**

同组人 无 实验室 实训大楼618-619 时间2023

**一、实验目的：**

1.了解Verilog 语法：规约运算符（|和&）、位拼接运算符（{}）。。

2.学习典型时序电路逻辑：计数器。

**二、实验内容：**

建立一个按键消抖及验证电路，对一位按键输入信号进行消抖操

作，并分别对经过消抖的信号和原始按键信号进行计数，对比验证按

键消抖效果。

按键消抖电路为简单时序电路，包含顶层模块、按键消抖模块和

计数器模块，以及2.3 节中介绍的分频器模块和移位寄存器模块。按键消抖模块例化一个分频器模块和一个移位寄存器实现消抖。其原理是以10ms 周期对原始按键信号进行采样，若连续3 次（按键抖动时间通常小于10ms）采样值相同，则相应改变输出值。

计数器模块在每个时钟上升沿对输入脉冲信号进行计数，并输出

计数值和计数溢出信号。计数位宽、最大计数值（模值）可通过参数

配置。

按键消抖验证电路顶层模块分别对原始按键输入信号和经过消

抖的按键信号进行计数，并分别用4 位LED 显示计数结果。对比按一

次按键对应计数结果的变化，可验证按键消抖效果。

**三、实验代码：**

key\_db.v 按键消抖

------------------------------------------------------------------------------------------------------

`timescale 1ns/1ns

module key\_db #(

parameter RST\_VALUE =1'b0, //表示复位时的值，默认为0（低电平）

parameter CLK\_FREQ\_Hz =27000000 //时钟频率，默认值

)(

input clk, //时钟信号

input rst, //复位信号

input key\_i, //按键输入信号

output key\_db\_o //消抖后的按键信号

);

wire clk\_100Hz;

freq\_div #( //实例化一个频率分频器模块，将输入信号分频得到100Hz的时钟信号

.DIV\_RATE\_2N (CLK\_FREQ\_Hz/100) //分频系数

) freq\_div\_i(

.clk\_in\_i (clk),

.rst\_i (rst),

.clk\_out\_o (clk\_100Hz)

);

//分频到100HZ

wire [2:0] samples; //存储按键信号的采样值

shift\_reg #( //实例化移位寄存器模块

.WIDTH (3),

.INIT\_VALUE ({3{RST\_VALUE[0]}})

) shift\_reg\_i(

.clk (clk\_100Hz), //将分频后的时钟信号作为时钟输入

.rst (rst),

.load\_i (1'b0), //低电平

.load\_data\_i (3'h0),

.reg\_in\_i (key\_i), //将按键信号作为数据输入

.reg\_out\_o (samples[2:0])

);

assign key\_db\_o =RST\_VALUE ? |samples[2:0] :&samples[2:0]; //根据复位值RST\_VALUE的状态，设置输出端口key\_db\_o的值，复位值为高电平则输出按键信号samples[2:0]的或操作结果,否则输出与操作结果

endmodule

------------------------------------------------------------------------------------------------------

counter.v 测试模块

------------------------------------------------------------------------------------------------------

`timescale 1ns/1ns //计数器

module counter #( //定义两个参数

parameter WIDTH =4, //计数器的宽度 四个灯led5-8

parameter DATA\_MAX =15 //计数器可计数的最大值 led5-8: 1111

)(

//定义输入端口

input clk, //时钟信号

input rst, //复位信号

input pulse\_i, //脉冲输入信号

output reg cout\_o, //定义一个寄存器类型的输出端口，表示计数器溢出信号

output reg [WIDTH-1:0] count\_o //表示计数器当前计数的值

);

localparam SIM\_DELAY =1; //定义一个本地常量，表示仿真运行的延时时间

always @(posedge clk or posedge rst) begin //边沿触发符合D触发器的发生条件（posedge表示上升沿触发，negedge表示下降沿触发）

if(rst) begin

cout\_o <=#SIM\_DELAY 0;

count\_o <=#SIM\_DELAY 0;

end else if(pulse\_i) begin //脉冲输入信号为高电平时执行

if(count\_o <DATA\_MAX) begin //计数值小于最大值时增加计数值

cout\_o <=#SIM\_DELAY 1'b0;

count\_o <=#SIM\_DELAY count\_o+1;

end else begin //计数值大于等于最大值时计数器清零

cout\_o <=#SIM\_DELAY 1'b1;

count\_o <=#SIM\_DELAY 0;

end

end

end

endmodule

------------------------------------------------------------------------------------------------------

//代码 key\_db\_demo.v

------------------------------------------------------------------------------------------------------`timescale 1ns/1ns

module key\_db\_demo(

input clk, //时钟信号

input rst\_n,

input key\_i, //按键输入信号

output [7:0] led\_o //输出端口：8个LED灯

);

wire rst =~rst\_n; //定义一个连线RST,通过取反操作将rst\_n转化为正性的复位信号

wire key\_db;

key\_db key\_db\_i( //按键消抖控制器

.clk (clk),

.rst (rst),

.key\_i (key\_i), //按键输入的值

.key\_db\_o (key\_db)

);

wire[7:0] led; //存储LED灯的状态

counter #( //16进制计数 下面四个消抖

.WIDTH (4),

.DATA\_MAX (15)

)counter\_i0(

.clk (key\_i),//对按键输入信号计数，消抖前

.rst (rst),

.pulse\_i (1'b1),

.cout\_o (),

.count\_o (led[3:0]) //计数器的输出结果作为LED灯的前四位

);

counter #(

.WIDTH (4), //位宽四位

.DATA\_MAX (15) //最大计数值15

)counter\_i1(

.clk (key\_db),//对消抖后的按键信号计数，将该信号作为时钟脉冲

.rst (rst),

.pulse\_i (1'b1), //初始化脉冲信号为高电平

.cout\_o (),

.count\_o (led[7:4]) //将输出结果作为LED灯的后四位，消抖后LED灯点亮的个数依次加一

);

assign led\_o[7:0]=~led[7:0]; //将led[7:0]取反后赋值给LED输出端口led\_o[7:0]，实现LED灯的控制。

endmodule

**四、实验结果：**

按动按键，发现上面四个led灯无规律变化，下面四个根据按键的次数一次累加变化。因为上面的输入信号是未经过按键消抖的信号，而下面是消抖过的。

成绩：

批阅教师：————————

日 期：————————

实 验 报 告 单

**实验名称： 脉宽调制**

同组人 无 实验室 实训大楼618-619 时间2023

**一、实验目的：**

1.了解Verilog 语法：关系运算符（>）。

**二、实验内容：**

一个脉宽调制PWM 控制电路，用于控制一个LED 灯的亮度该电路为简单时序电路，包含顶层模块、脉宽调制控制器模块，以及2.5 节中介绍的按键消抖模块等。

脉宽调制控制器内部实现一个16 位计数器，在输入时钟驱动下进行周期性计数，产生一个锯齿波。将输入阈值信号与锯齿波进行比较，产生脉冲宽度受控于输入阈值的脉冲输出。阈值越大，脉冲宽度越大，用于驱动LED 灯也越亮。

**三、实验代码：**

pwm\_ctrl.v pwm波控制模块

初始状态计数器小于阈值，输出高电平；

通过按键提高阈值，改变输出，最后表现为led熄灭

------------------------------------------------------------------------------------------------------

`timescale 1ns/1ns

module pwm\_ctrl(

input clk,

input rst,

input enable\_i,

input [15:0] thres\_i, //输入阈值

output pwm\_o

);

wire [15:0] count;

counter #(

.WIDTH (16),

.DATA\_MAX (16'hffff)

) counter\_i (

.clk (clk),

.rst (rst),

.pulse\_i (1'b1),

.cout\_o (),

.count\_o (count[15:0])

);

assign pwm\_o = (count[15:0] < thres\_i[15:0]) ? 1'b1 : 1'b0;

//输出的判断条件 如果小于阈值则输出高电平，否则输出低电平

endmodule

------------------------------------------------------------------------------------------------------

pwm\_demo.v

------------------------------------------------------------------------------------------------------

`timescale 1ns/1ns

module pwm\_demo(

input clk,

input rst\_n,

input key\_i,

output led\_o

);

wire rst = ~rst\_n;

wire key\_db;//按键消抖

key\_db key\_db\_i(

.clk (clk),

.rst (rst),

.key\_i (key\_i),

.key\_db\_o (key\_db)

);

wire [3:0] thres;

counter #(

.WIDTH (4),

.DATA\_MAX (15)

) counter\_i (

.clk (key\_db),

.rst (rst),

.pulse\_i (1'b1),

.cout\_o (),

.count\_o (thres[3:0]) //阈值的高四位可变部分

);

pwm\_ctrl pwm\_ctrl\_i(

.clk (clk),

.rst (rst),

.enable\_i (1'b1),//使能信号

.thres\_i ({thres[3:0], 12'hfff}),

.pwm\_o (led\_o)

);

endmodule

**四、实验结果：**

通过按键改变阈值，十五次为一个循环，每按一次阈值提高一次，灯亮的时间越来越短，灯的亮度不断变暗直到熄灭。

成绩：

批阅教师：————————

日 期：————————