浙江工业大学

《可编程逻辑器件及应用》
课程实验报告

学生姓名 凌智城

指导教师 龚树凤

专业班级 通信工程1803班

培养类别 全日制本科

所在学院 信息工程学院

提交日期 2021年1月4日

（完成正文后，再根据实际页码对目录页码进行补正）

目　录

（目录内容可以根据个人实际情况，适当调整）

实验一：常用组合电路、时序逻辑电路设计

1.1 计数器

1.1.1 设计任务

使用 Modelsim 软件设计同步置数、同步复位功能的 6 位二进制计数器，并完成仿真验证。

1.1.2 设计思路与原理

输出 out，输入数据 data，置数信号 load，时钟 lck，清零 rst

采用同步置数同步复位的方法，若达到 clk 上升沿，此时 rst 为低电平则计数器清零，否则若 load 为高电平表示允许置数，将 data 赋值给 out，其他情况均为 out=out+1 即正常计数状态。

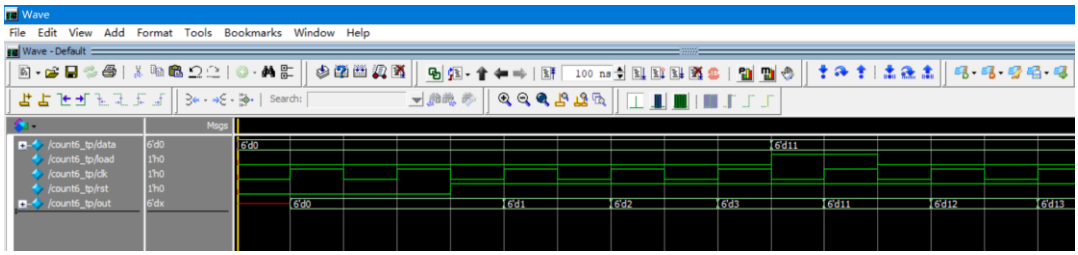
学号为 201806061211，故在 testbench 中设置置数信号来临之前 data=4’b1011 即十进制的十一，等 load 信号来临之后置给 out

1.1.3 Verilog 源代码与注释

```
module count6(out,data,load,clk,rst);
    output[5:0]out;
    input[5:0]data;
    input load,clk,rst;
    reg[5:0]out;
    always@(posedge clk )
        if(!rst)                        //同步复位
            out=6'b0000;                //如果有 rst 为 0 则清零
        else if(load)                   //同步置数
            out=data;                   //如果 load 为 1 则置数
        else                            //不是上述情况则+1
            out=out+1;
endmodule
```

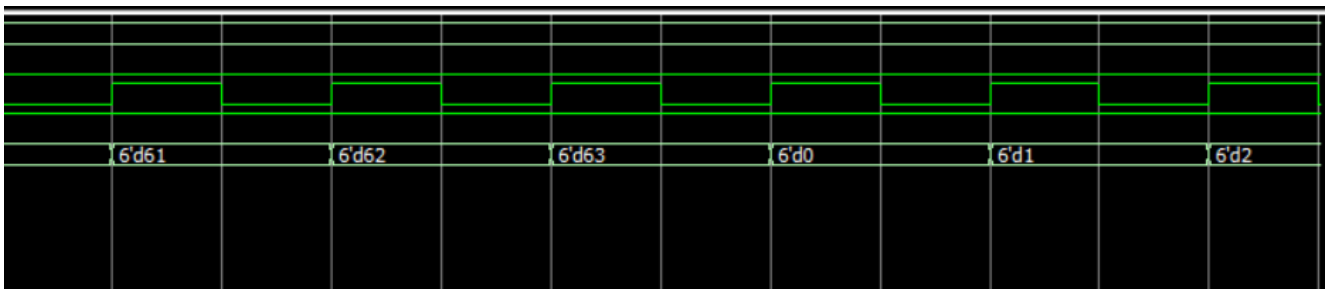
```
`timescale 10ns/1ns
module count6_tp;
    reg[5:0]data;                    //输入为 reg 型
    reg load,clk,rst;
    wire[5:0]out;                    //输出为 wire 型
    count6 mycount(.out(out),.data(data),.load(load),.clk(clk),.rst(rst));
    initial clk=0;
    always
        begin
        #5 clk=1'b1;            //每隔 5 进行一次翻转
        #5 clk=1'b0;
        end
    initial
        begin
            data=6'b000000;          //初始化
            load=0;
            rst=0;
            #20 rst=1;                    //复位
            #30 data=4'b0111;        //置数 data 为 11（学号尾号为 11）
    load=1;                     //置数
            #10 load=0;
            #800 $finish;
        end
endmodule
```

1.1.4 仿真结果与分析



在 destbench 中将 data 赋值为 11（学号后两位），load 信号来临之后，同步置数，故在下一个 load=1 且 clk 上升沿到达是将 data 置给 out

图 1-1 六进制计数器的仿真截图（初始部分）



二位六进制计数器最高计数至 6’b111111 即 6’d63，从 0~63

图 1-2 六进制计数器的仿真截图（计数溢出部分）

1.2 分频器

1.2.1 设计任务

设计输出占空比为 50%能实现 6 和 7 分频的分频器，并完成仿真验证工作

1.2.2 设计思路与原理

先分别设计一个 6 分频一个 7 分频 division_6 和 division_7 分频器，由顶层 fp 调用两个底层分频模块，用 fp_tp 进行测试。6 分频器设计一个模 6/2 计数器，计数器的计数范围是 0～6/2-1，当计数值为 6/2-1 时，输出信号进行翻转；七分频器两个计数器分别是 count1 和 count2，clk_A、clk_B 分别是 count1、count2 控制的模 7 计数器输出端，clk_even 是信号 clk_A 和 clk_B 的逻辑或输出端，clk_A 和 clk_B 进行按位或运算即可得到 clk_odd

1.2.3 Verilog 源代码与注释

```
module division_6(clk,rst,clk_odd);      //六分频器
    input clk;
    input rst;
```

```
    output clk_odd;
    reg clk_odd;
    reg[3:0] count;
    parameter N=6;
    always @(posedge clk)              //同步清零
   if(!rst)                            //如果 rst=0 则全部清零
        begin
       count<=1'b0;
       clk_odd<=1'b0;
        end
   else if(N%2==0)
        begin                          //如果小于 N/2-1 那么计数+1，clk_odd 保
持
       if(count<N/2-1)
            begin
           count<=count+1'b1;
            end
       else                            //其他情况则计数值清零，clk_odd 反转
            begin                      //达到 50%占空比的六分频器
           count<=1'b0;
           clk_odd<=~ clk_odd;
            end
       end
endmodule


module division_7(clk,rst,clk_even);       //七分频器
    input clk,rst;
    output clk_even;
    reg[3:0] count1,count2;                //定义两个计数信号
    reg clkA,clkB;                         //相应的，产生两个 clk 信号
    parameter N=7;

    assign clk_re=~clk;
    assign clk_even=clkA|clkB;             //两个 clk 信号做或运算，合成
clk_even
                                           //即达到七分频效果
    always @(posedge clk)                  //时钟上升沿同步清零
       if(!rst)                            //如果 rst=0 则全部清零
            begin
                count1<=1'b0;
                clkA<=1'b0;
            end
   else if(N%2==1)                         //7 为奇数，一半要以余一为界
```

```
          begin
if(count1<(N-1))
                        begin
                            count1<=count1+1'b1;          //在到达 7 之前
                                if(count1==(N-1)/2)       //count1＋1
                                    begin                 //如果刚好加到了
                                        clkA=~clkA;   //比一半小的最大的
                                    end                       //整数，则反转
                         end
              else                                            //其他情况则反转
                   begin
                       clkA=~clkA;
                       count1=1'b0;
                   end
          end
     else
          clkA=1'b0;

always @(posedge clk_re)              //clk_re 时钟上升沿同步清零
//即 clk 时钟下降沿同步清零
          if(!rst)                            //如果 rst=0 则全部清零
               begin
                    count2<=1'b0;
                    clkB<=1'b0;
               end
          else if(N%2==1)
                  begin
          if(count2<(N-1))
                        begin
                            count2<=count2+1'b1;
                                if(count2==(N-1)/2)
                                    begin
                                        clkB=~clkB;
                                    end
                     end
              else
                   begin
                       clkB=~clkB;
                       count2=1'b0;
                   end
           end
     else
         clkB=1'b0;
endmodule
```

```
module fp(clk,rst,clk_even,clk_odd);
        input clk;
        input rst;
        output clk_odd;
        output clk_even;
        division_6 fp1(clk,rst,clk_odd);
        division_7 fp2(clk,rst,clk_even);
endmodule


`timescale 10ns/1ns
module fp_tp;
        reg clk,rst;
        wire clk_odd,clk_even;
        fp myfp(clk,rst,clk_even,clk_odd);
        initial clk=0;
       always
             begin
                 #5 clk=1'b1;                    //每隔五 clk 翻转
                 #5 clk=1'b0;
             end
       initial
             begin
                 rst=0;
                 #20 rst=1;
                 #800 $finish;
             end
endmodule
```

1.2.4 仿真结果与分析

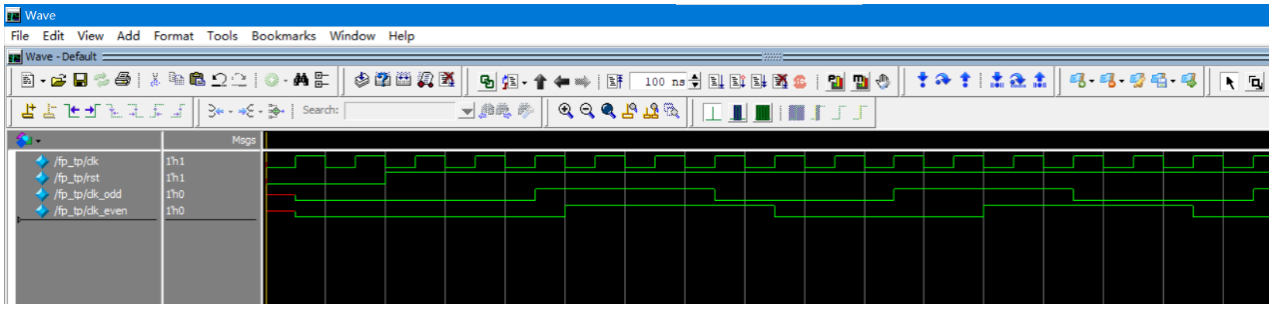


图 1-3　6 分频器和 7 分频器的仿真截图

fp_tp 测试模块，clk 时钟信号每过五个时间翻转一次，置数信号 rst 来临之后，clk_odd 为六分频即每隔六个 clk 翻转一次，clk_even 为七分频即每隔七个 clk 翻转一次

实验总结

1、 通过仿真和硬件实验的比较，你对哪些概念从不理解到理解了？

第一次进行 FPGA 的仿真和硬件实验,在 verilogHDL 文件中的 wire 和 reg 类型，以及测试模块中的类型，输入需要是 wire，输出可以是 wire/reg, 而测试模块输入必须是 reg，输出必须是 wire，initial 和 always 不能互相嵌套，assign 只能对 wire 进行赋值，并且对如何通过 Modelsim 对测试模块进行仿真测试都有了比较详细的了解，加深了代码规范的印象。

2、 相比其它方法（比如数字逻辑设计、单片机等），你认为 FPGA 对数字电路设计有哪些优势？

可以不用考虑具体的数字逻辑实现单元的构成,但传统逻辑电路设计必须考虑这些,并自己搭建;设计灵活,可以不悲标准器件在逻辑功能上限制,并且由于 FPGA 的可编程性，可以大大缩减设计周期

3、 你对设计一个比较复杂的工程项目有何一般性的方法？

模块化设计，分为顶层和底层模块分开设计，将一个复杂的工程项目拆解成多个小模块，可以同时由多个人进行开发，提升设计效率，最后子模块调试完成后由顶层模块调用。

实验改进建议

建议一：

建议二：