电子科技大学

University of Electronic Science and Technology of China

本科生实验报告

实验课程：复杂数字集成电路设计（挑战性课程）

实验名称：实验 3：偶数分频和奇数分频

实验地点：无

学生姓名：周岳恒

学　　号：2021340105016

指导教师：廖永波

实验时间：2023 年 10 月 1 日

一、 实验目的

学会使用：1)触发器级联；2)计数器；两种方式实现偶数分频，从 2,4（2^n）分频到任意 2N 分频递进练习。

学会使用两种边沿触发的多个触发器实现奇数分频。最终把偶数分频和奇数分频放到一个模块中实现任意分频，使用 N 在模块外参数例化，方便代码设计。

二、 实验任务

1. 使用触发器级联的方式实现 2,4 分频。
2. 使用单一计数器方式实现 6 分频，并扩展到 2N 分频。
3. 使用两种边沿触发的多个触发器实现任意 N（N=2k-1）分频。
4. 在一个模块内实现任意 N（$N \in N^*$）分频。

三、 实验原理

1. 触发器级联使下一级时钟接到上一级输出状态，级联在末端的触发器时钟节拍变慢，如同 T 触发器级联形成二进制计数器。在第 n 个触发器的输出翻转表示完成 2^{n-1} 分频。
2. 计数器通过把输出数值通过组合逻辑判断是否到指定值，并把结果输出到此计数器的 clr 清零端输入实现任意偶数分频。
3. 奇数分频需要有一个上升沿计数器和一个下降沿计数器，通过组合逻辑处理两者的输出决定输出时钟信号在输入时钟的上升沿还是下降沿翻转。
4. 把奇数分频器和偶数分频器通过顶层模块连接，判断输入分频因子是奇数还是偶数后决定输出使用奇数分频器还是偶数分频器。

四、 实验过程

2,4 分频器：

对于 2^n 的分频器，只需要级联触发器，即下一级的触发器 clk 输入连接到上一级输出 q 即可实现分频功能。每个触发器的输入为自身输出取反。第 n 个触发器的输出就是 2^{n-1} 分频的输出时钟。

verilog 代码：

```
module div_2_4_d (
    input clk,
```

```
    input rst_n,
    output div_2,
    output div_4
);
    reg div_2_reg,div_4_reg;
    always @(posedge clk, negedge rst_n) begin
        if(~rst_n)
            div_2_reg <= 1'b0;
        else
            div_2_reg <= ~div_2_reg;
    end

    always @(posedge div_2_reg, negedge rst_n) begin
        if(~rst_n)
            div_4_reg <= 1'b0;
        else
            div_4_reg <= ~div_4_reg;
    end

    assign div_2 = div_2_reg;
    assign div_4 = div_4_reg;

endmodule
```

偶数分频器：

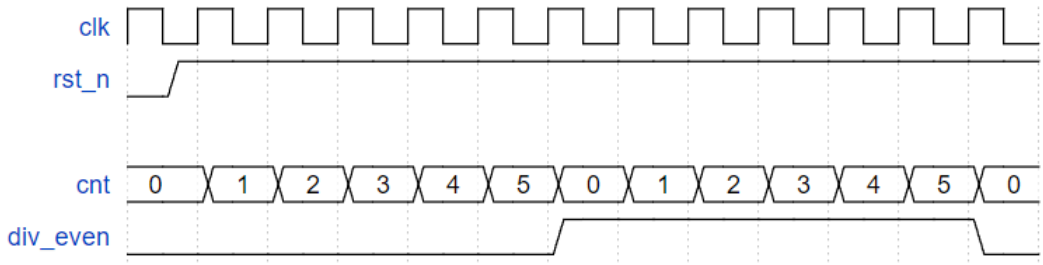
使用计数器来实现分频。对于 2N 分频，计数器的计数周期为 N-1，使用向上计数。到达计数周期后对控制时钟输出的触发器给一个时钟周期的使能有效信号。此触发器的输入为自身输出的取反即 T 触发器，实现时钟边沿的改变，且由使能信号控制只会在计数周期到达之后翻转。

verilog 代码：

```
module div_even #(
    parameter N = 6
)(
    input clk,
    input rst_n,
    output div_even
);
    reg [31:0] cnt;
    reg div_even_reg;
    always @(posedge clk, negedge rst_n) begin
        if(~rst_n) begin
```

```
            cnt <= 32'h0;
            div_even_reg <= 1'b0;
        end
        else begin
            if(cnt == (N/2-1)) begin
                cnt <= 32'h0;
                div_even_reg <= ~div_even_reg;
            end
            else
                cnt <= cnt + 32'h1;
        end
    end

    assign div_even = div_even_reg;
endmodule
```

波形图：



奇数分频器：

要使用上升沿触发器计数和下降沿触发器计数。对于 N 分频奇数计数器，这里把两个计数器的计数周期设置为 N-1，穷举出所有计数的可能性（以 N=5 为例）：

Table 1 先检测到上升沿

cnt1	cnt2	clk_out
0	0	0
1	0	0
1	1	0
2	1	0
2	2	1
3	2	1
3	3	1
4	3	1
4	4	1
0	4	0

Table 2 先检测到下降沿

cnt1	cnt2	clk_out
0	0	0
0	1	0
1	1	0
1	2	0
2	2	1
2	3	1
3	3	1
3	4	1
4	4	1
4	0	0

可以看到在复位后不管先检测到上升沿还是先检测到下降沿，在 cnt1 和 cnt2 都大于等于(N+1)/2 时的总个数正好位所有可能的个数/2，而一个计数器周期正好是 2N=10 分频，所以这样得到的 clk_out 时 5 分频，达到奇数分频的目的。

verilog 代码：

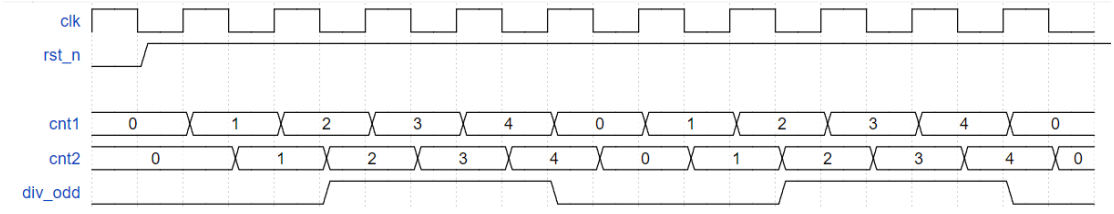
```
module div_odd #(
    parameter  N = 5
)
(
    input clk,
    input rst_n,
    output div_odd
);
    reg [31:0] cnt1,cnt2;
    wire trigger;
    assign trigger = (cnt1 >= (N-1)/2 && cnt2 >= (N-1)/2);

    always @(posedge clk, negedge rst_n) begin
        if(~rst_n)
            cnt1 <= 32'h0;
        else begin
            if(cnt1 == N-1)
```

```
                cnt1 <= 32'h0;
            else
                cnt1 <= cnt1 + 32'h1;
        end
    end
    always @(negedge clk, negedge rst_n) begin
        if(~rst_n)
            cnt2 <= 32'h0;
        else begin
            if(cnt2 == N-1)
                cnt2 <= 32'h0;
            else
                cnt2 <= cnt2 + 32'h1;
        end
    end

    assign div_odd = trigger;

endmodule
```

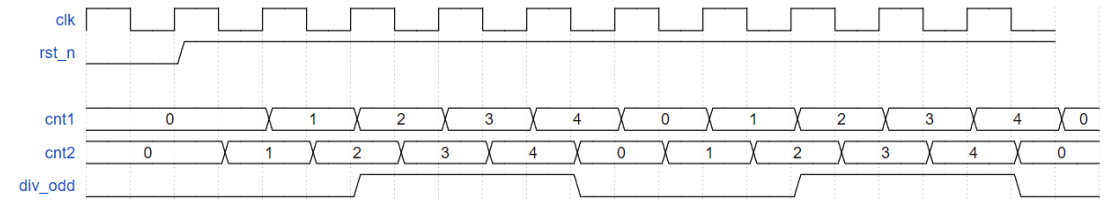
虽然计算(N-1)/2 用到了除法，且 N 位 parameter 不好用移位，但是 N 的设定本身就是在综合前就定下来的常量，所以不会综合出除法器或移位逻辑。

波形图：

先检测到上升沿 cnt1：



先检测到下降沿 cnt2：



任意分频计数器：

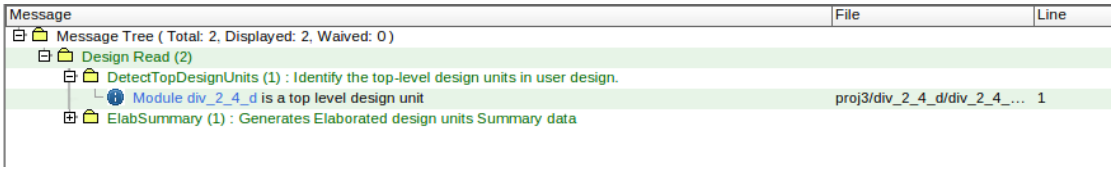
直接调用上述奇数分频计数器和偶数分频计数器即可。注意在给计数器的分频输入时：如果 N 为奇数，偶数分频计数器的 N 输入要把最低位置 0 保证是偶

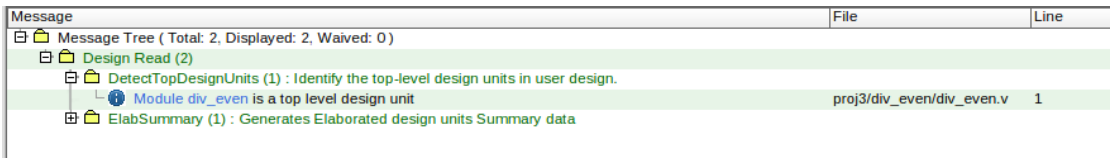
数；如果 N 是奇数，要保证奇数分频计数器的 N 输入要把最低位置 1 保证是奇数。最后根据 N 的最低位是否为 1，即是奇数还是偶数来控制多路选择输出。

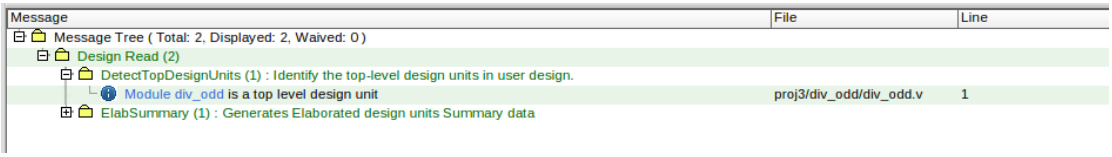
verilog 代码：

```
module division #(
    parameter N = 5
)(
    input clk,
    input rst_n,
    output div_out
);
    wire div_even,div_odd;
    div_even #(
        .N(N & 32'hFFFF_FFFE)
    ) u_even (
        .clk(clk),
        .rst_n(rst_n),
        .div_even(div_even)
    );
    div_odd #(
        .N(N | 32'h1)
    ) u_odd (
        .clk(clk),
        .rst_n(rst_n),
        .div_odd(div_odd)
    );

    assign div_out = (N & 32'h1 == 32'h1) ? div_odd : div_even;

endmodule
```

spyglass 报告：







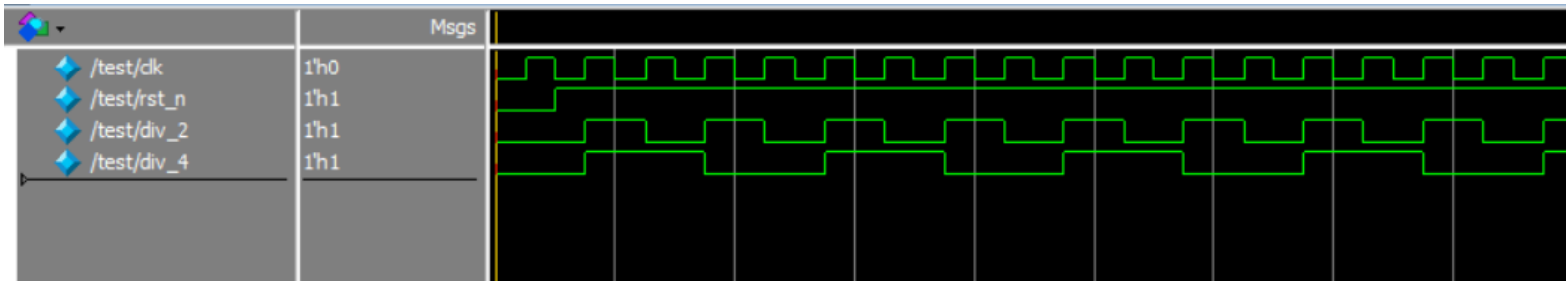
五、 仿真结果

2,4 分频器：

testbench：

```
`timescale 1ps/1ps
module test();
    reg clk,rst_n;
    wire div_2,div_4;

    div_2_4_d u1(
        .clk(clk),
        .rst_n(rst_n),
        .div_2(div_2),
        .div_4(div_4)
    );

    initial begin
        clk = 1'b0;
        forever #5 clk = ~clk;
    end

    initial begin
        rst_n = 1'b0;
        #10 rst_n = 1'b1;
    end

endmodule
```

仿真波形：



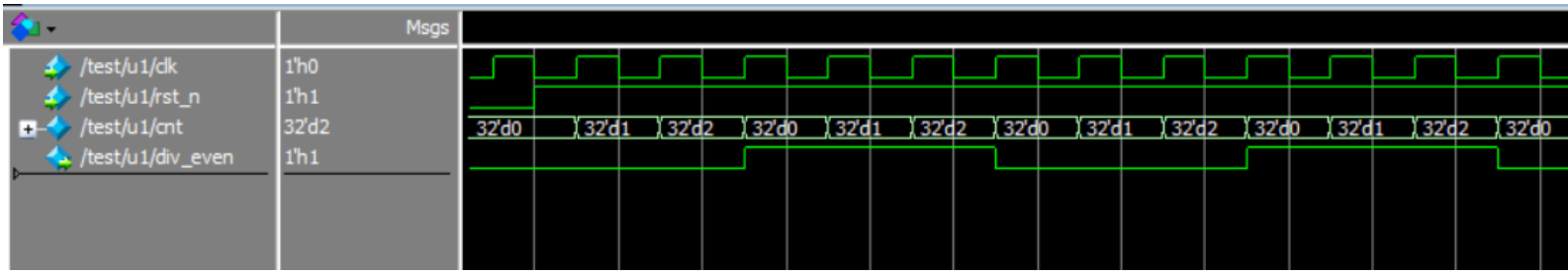
可以看到在复位结束后 div_2 为 clk 的 2 分频，div_4 为 clk 的 4 分频。

偶数分频器：

testbench：

```
`timescale 1ps/1ps
module test();
    reg clk,rst_n;
    wire div_even;

    div_even #(
        .N(6)
    ) u1(
        .clk(clk),
        .rst_n(rst_n),
        .div_even(div_even)
    );

    initial begin
        clk = 1'b0;
        forever #5 clk = ~clk;
    end

    initial begin
        rst_n = 1'b0;
        #10 rst_n = 1'b1;
    end

endmodule
```

仿真波形：



结果和预期波形图一致。

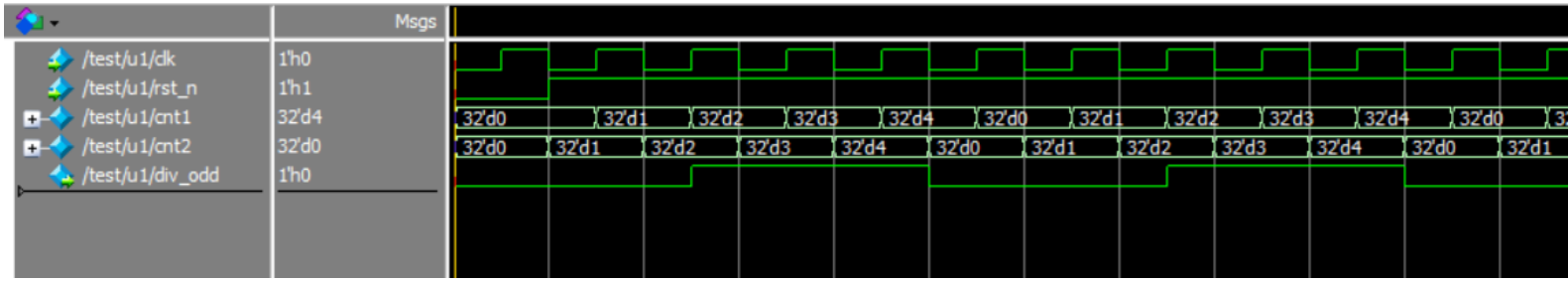
奇数分频器：

testbench：

```
`timescale 1ps/1ps
module test();
    reg clk,rst_n;
    wire div_odd;
```

```
    div_odd #(
        .N(5)
    ) u1(
        .clk(clk),
        .rst_n(rst_n),
        .div_odd(div_odd)
    );

    initial begin
        clk = 1'b0;
        forever #5 clk = ~clk;
    end

    initial begin
        rst_n = 1'b0;
        #10 rst_n = 1'b1;
    end

endmodule
```

仿真波形：



结果和预期波形图一致。

任意分频计数器：

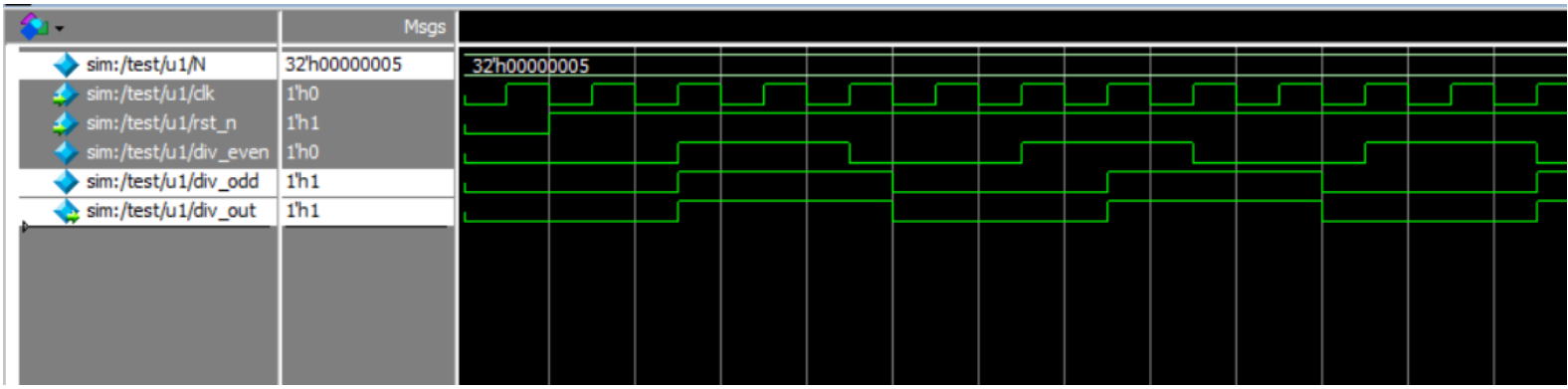
testbench：

```
`timescale 1ps/1ps
module test();
    reg clk,rst_n;
    wire div_out1,div_out2;

    division #(
        .N(5)
    ) u1(
        .clk(clk),
        .rst_n(rst_n),
        .div_out(div_out1)
    );
```

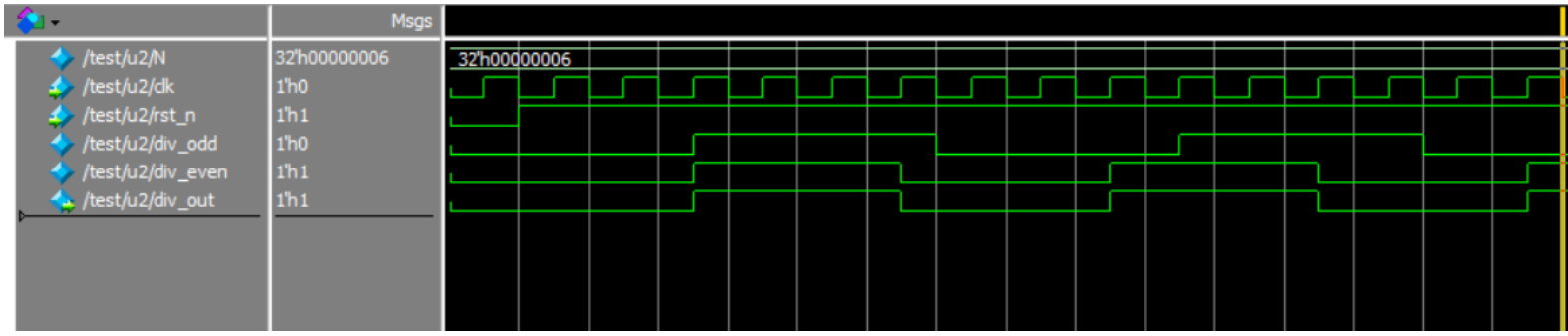
```
    division #(
        .N(6)
    ) u2(
        .clk(clk),
        .rst_n(rst_n),
        .div_out(div_out2)
    );

    initial begin
        clk = 1'b0;
        forever #5 clk = ~clk;
    end

    initial begin
        rst_n = 1'b0;
        #10 rst_n = 1'b1;
    end

endmodule
```

仿真波形：

N=5：



N=6：



六、 实验总结

本次实验从 2,4 分频器，偶数分频器，奇数分频器到任意数分频器的递进实验，使我对分频器有基本的掌握，能够设计出任意数分频的分频器。核心思想都是用计数器计数外加组合逻辑实现分频效果，但是奇数分频计数器的设计稍有复杂，但本质上都是一样的。

在实现指定数分频的设计后，我同样完成了任意数分频的设计，也就是说，可以通过，模块例化的方式指定分频因子 N 到别的主模块中实现方便例化，对今后的模块化设计大有裨益。

本次实验用到了 T 触发器级联的 2 进制计数器、有 clr 清零功能的计数器、上下边沿多触发器组合的计数器。在学习分频器之余，我对于计数器的理解程度进一步提升。