COD LAB0

> 实验题 1. 在 Verilog oj47 中，我们实现了一个 15 位的计数器。事实上在此基础上，通过 一些简单的修改，我们就可以实现一个任意位数的计数器。现在，我们希望完成一个时 分秒时钟，该时钟在每次 clk 上升沿时秒位加 1（clk 信号不必以秒为周期），满 20 后清 零，分位加 1；分位满 10 后清零，时位加 1；时位满 5 后三个位全部清零，如此循环。 也就是说，我们设计的时钟一分钟只有 20 秒，一小时只有 10 分钟，一天只有五小时。 请尝试使用模块化的设计方法完成该时钟，并给出所有模块输入输出的仿真波形。

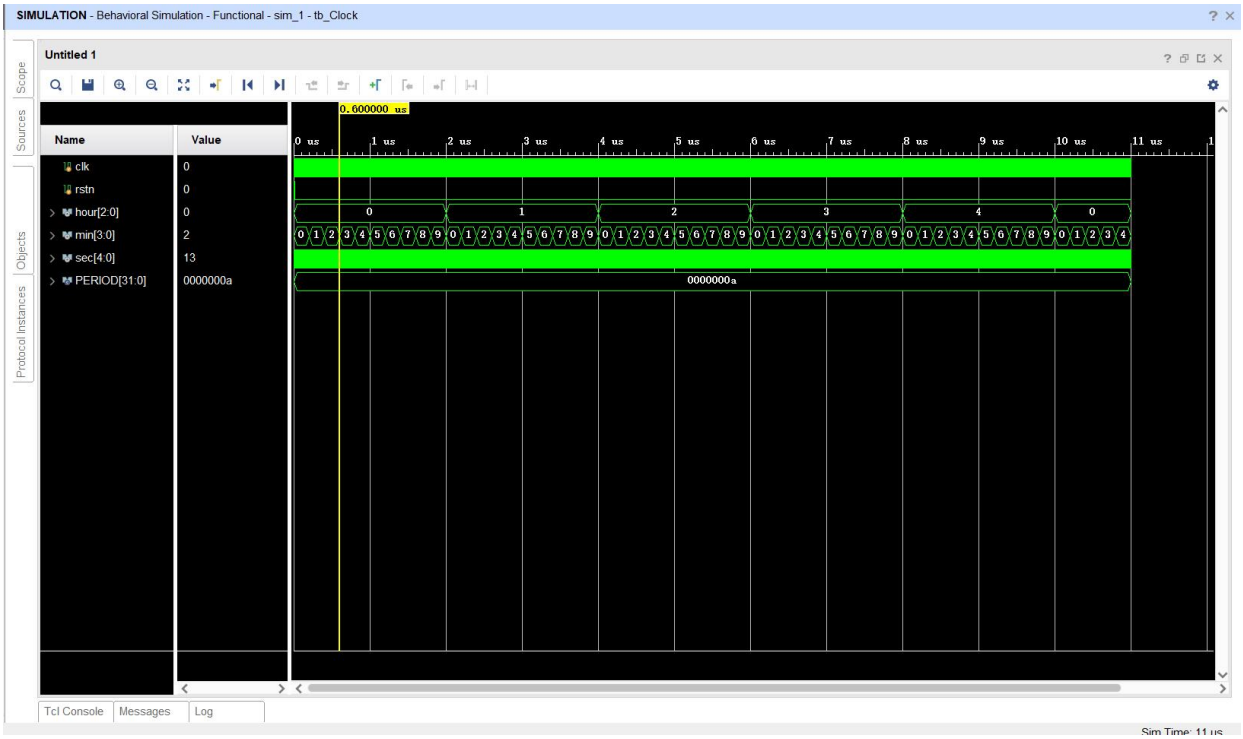
代码

```
`timescale 1ns / 1ps

module Clock (
    input clk,
    input rstn,
    output [2:0] hour,//3位，5进，101
    output [3:0] min,//4位，10进，1010
    output [4:0] sec//5位，20进，10100
);

    // 实例化子模块
    Sec sec1 (.clk(clk), .sec_rst(rstn), .sec_out(sec));
    Min min1 (.clk(clk), .min_rst(rstn), .min_out(min), .sec_in(sec));
    Hour hour1 (.clk(clk), .hour_rst(rstn), .hour_out(hour), .min_in(min), .sec_in(sec));

endmodule

// 秒模块
module Sec (
    input clk,
    input sec_rst,
    output reg [4:0] sec_out
);

    always @(posedge clk or posedge sec_rst) begin
        if (sec_rst) begin
            sec_out <= 5'b0;
        end
        else if (sec_out == 5'b10011) begin
              sec_out <= 5'b0;
            end
        else begin
              sec_out <= sec_out + 1;
            end
```

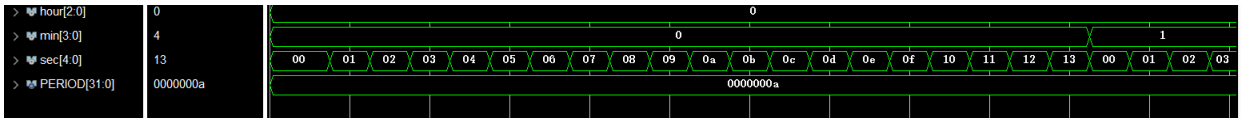
```
        end


endmodule

// 分模块
module Min (
   input clk,
   input min_rst,
   output reg [3:0] min_out,
   input [4:0] sec_in
);

   always @(posedge clk or posedge min_rst) begin
      if (min_rst) begin
         min_out <= 4'b0;
      end
      else if (sec_in == 5'b10011 && min_out == 4'b1001) begin
            min_out <= 0;
         end
      else if (sec_in == 5'b10011) begin
            min_out <= min_out + 1;
         end
      end


endmodule

// 时模块
module Hour (
   input clk,
   input hour_rst,
   output reg [2:0] hour_out,
   //output reg [3:0] min_out,
   //output reg [4:0] sec_out,
   input [3:0] min_in,
   input [4:0] sec_in
);

   always @(posedge clk or posedge hour_rst) begin
      if (hour_rst) begin
         hour_out <= 3'b0;
      end
      else if (min_in == 4'b1001 && sec_in == 5'b10011&& hour_out == 3'b100) begin
            hour_out <= 3'b0;
            //min_out <= 4'b0;
            //sec_out <= 5'b0;
         end
      else if (min_in == 4'b1001 && sec_in ==5'b10011) begin
            hour_out <= hour_out + 1;
         end
      end
```

```
87
88
89      endmodule
90
```

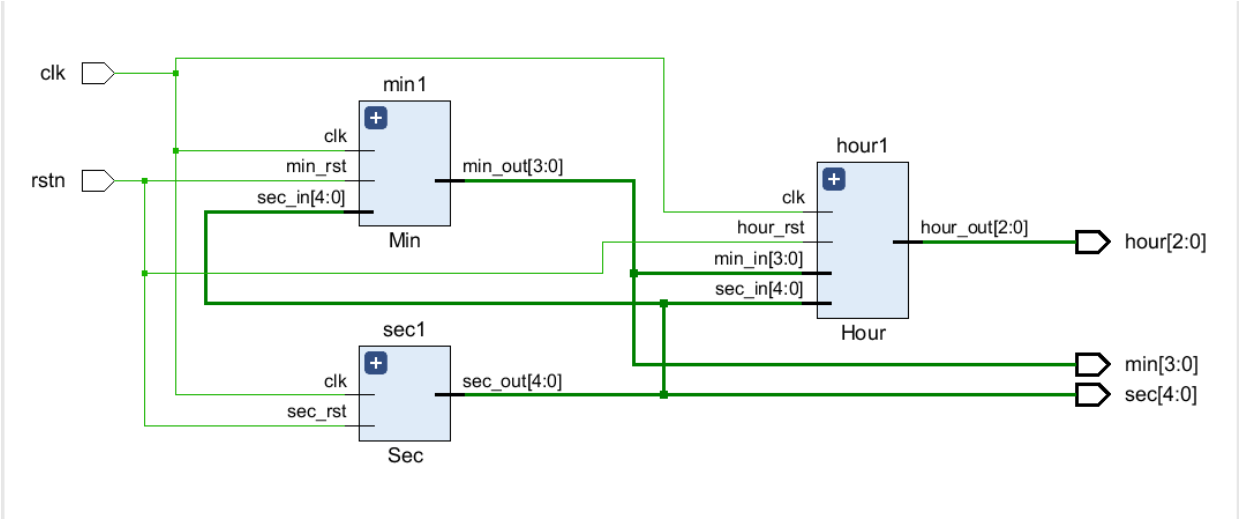
输入输出仿真波形



下面展示秒的进位：



VIVADO生成的电路图



反馈

vscode配置比较难受，系统环境变量配置完成并且手动添加Path后xvlog仍然只能显示代码高亮，没有报错功能

希望实验文档能够多多包含可能出现的问题的解决方法

本次实验不难，但是能很好帮助回忆数电学习的内容