深圳大学考试答题纸

(以论文、报告等形式考核专用)

二〇 二四 ～二〇 二五 学年度第 一 学期

课程编号 1602080001 课序号 01 课程名称 硬件描述语言与逻辑综合 主讲教师 刘春平 评分

学 号 2022280485 姓名 贾苏健 专业年级 电子信息工程（文华班） 22 级

教师评语：

题目： **基于 Verilog 的数字时钟**

设计思路与架构（15 分）		源程序（20 分）		仿真结果（20 分）		实验结果（20 分）		总结和体会（15 分）		排版和格式（10 分）
硬件方框图或软件流程图（10 分）	关于设计方案清晰的文字说明或表述（5 分）	正确性（无语法错误和逻辑错误）（15 分）	规范性（格式规范、必要的注释）（5 分）	正确、全面、清晰的仿真波形或数据（15 分）	关于仿真结果必要的分析和说明（5 分）	正确、全面、清晰的实验波形、数据或截图（15 分	关于实验结果必要的分析和说明（5 分）	作业过程中碰到的问题及解决方法（10 分）	作业过程中个人收获及不足、对本课程的建议等（5 分）	排版漂亮、格式规范（10 分）

目录

一、实验目的

1、 模块化设计：按照功能划分设计分频器、计数器、译码器等子模块，并在顶层模块中进行整合。

2、 时钟功能实现：实现秒、分、时的计时逻辑，支持进位、复位功能，并确保计时范围为秒（00-59）、分（00-59）、时（00-23）。

3、 数码管显示：设计数码管显示模块，正确显示时间信息。

4、 仿真与验证：通过仿真工具验证各模块的正确性，包括分频、计数、进位、复位及数码管显示功能，分析仿真波形和数据。

二、设计方案

1、 系统框架

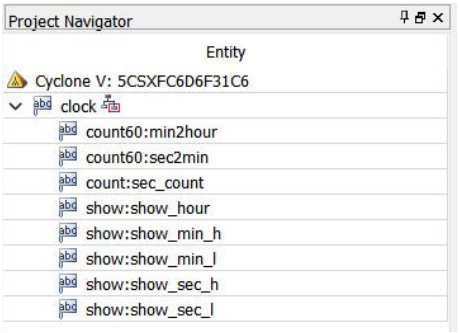
数字时钟系统的总体框架由以下几个模块组成：

分频模块（count）：将主时钟信号分频至 1 Hz，用于驱动秒计数。

计数模块（count60 和 hour 计数器）：实现秒到分钟、分钟到小时的进位计数逻辑。

译码模块（show）：将计数器输出的二进制时间数据转换为七段数码管显示信号。

顶层模块（clock）：整合所有子模块，完成系统功能。



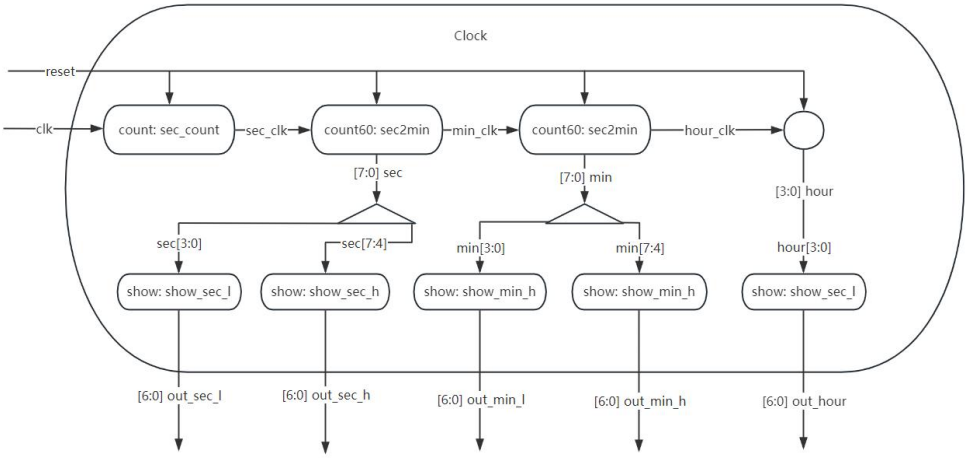
图（1）qpf 文件系统设计

2、 设计步骤

(1) 程序框图

以下是系统程序的逻辑流程：

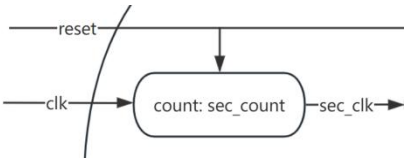
分频模块生成 1 Hz 的秒脉冲。秒计数模块通过秒脉冲进行计数，计数至 59 时发送分钟进位信号。分钟计数模块接收秒进位信号进行计数，计数至 59 时发送小时进位信号。小时计数模块接收分钟进位信号进行计数，计数至 23 时清零。译码模块将各计数模块的输出转换为七段数码管信号。系统实现复位功能，通过复位信号清零计时数据。



图（2）系统框图

(2) 模块说明

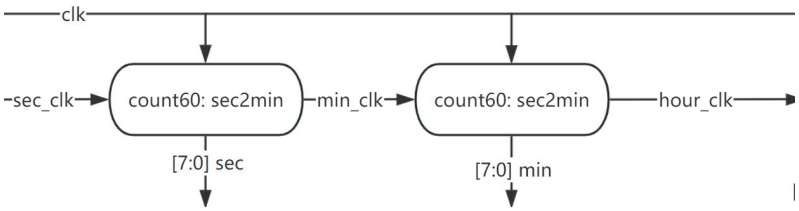
分频模块：将主时钟分频至 1 Hz。通过计数器记录时钟周期，在达到设定值时输出翻转信号。



图（3）分频模块

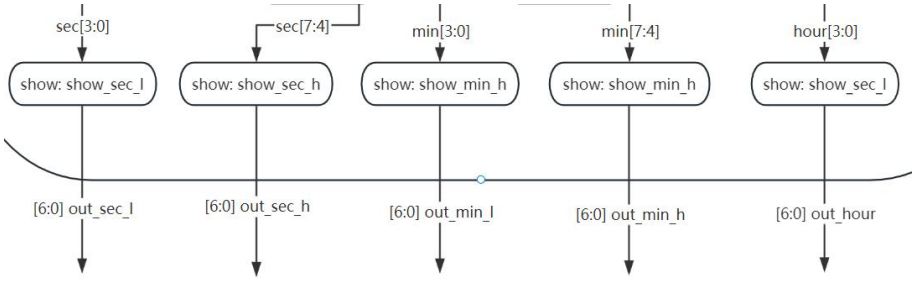
秒计数模块：计数范围为 0-59。当秒计满 59 后，输出分钟进位信号。

分钟计数模块：计数范围为 0-59。接收秒计数模块的进位信号，当分钟计满 59 后，输出小时进位信号。



图（4）时钟计数模块

译码模块：将计数器的二进制数据转换为七段数码管信号，驱动数码管显示时间。



图（5）数码管译码模块

3、 代码程序

(1) 顶层模块 clock.v

顶层模块（clock）：整合各功能模块，完成数字时钟设计。

```
1  module clock(reset, clk, out_sec_l, out_sec_h, out_min_l, out_min_h, out_hour);
2      input reset, clk;
3      output [6:0] out_sec_l, out_sec_h, out_min_l, out_min_h, out_hour;
4
5      wire [6:0] out_sec_l, out_sec_h, out_min_l, out_min_h, out_hour;
6      wire [7:0] sec, min;
7      reg [3:0] hour;
8      wire sec_clk, min_clk, hour_clk;
9
10     count sec_count(reset, clk, sec_clk);
11     count60 sec2min(reset, sec_clk, sec, min_clk);
12     count60 min2hour(reset, min_clk, min, hour_clk);
13
14     always @(negedge hour_clk)
15     begin
16         if (reset)
17             hour[3:0] <= 0;
18         else
19             hour[3:0] <= hour[3:0] + 1;
20     end
21
22     show show_sec_l(out_sec_l, sec[3:0]);
23     show show_sec_h(out_sec_h, sec[7:4]);
24     show show_min_l(out_min_l, min[3:0]);
25     show show_min_h(out_min_h, min[7:4]);
26     show show_hour(out_hour, hour[3:0]);
27 endmodule
28
```

图（6）顶层模块 clock.v 代码

(2) 分频模块 count.v

分频模块（count）：实现秒脉冲信号生成。

```
module count(reset, clk, clk_out);
    input reset, clk;
    output clk_out;

    reg [31:0] cnt;
    reg clk_out;

    always @(posedge clk)
    begin
        if (reset)
            cnt <= 0;
        else if (cnt == 24999999)
        begin
            cnt <= 0;
            clk_out <= ~clk_out;
        end
        else
            cnt <= cnt + 1;
    end
endmodule
```

图（7）分频模块 count.v 代码

(3) 计数模块 count60.v

计数模块（count60）：实现计数和进位逻辑。

```
module count60(reset, clk, qout, cout);
    input reset, clk;
    output cout;
    output [7:0] qout;

    reg [7:0] qout;
    reg cout;

    always @(posedge clk or posedge reset)
    begin
        if (reset)
            qout <= 0;
        else
        begin
            if (qout[3:0] == 9)
            begin
                qout[3:0] <= 0;
                if (qout[7:4] == 5)
                    qout[7:4] <= 0;
                else
                    qout[7:4] <= qout[7:4] + 1;
            end
            else
                qout[3:0] <= qout[3:0] + 1;

            cout = (qout == 8'h59) ? 1 : 0;
        end
    end
endmodule
```

图（8）计数模块 count60.v 代码

(4) 译码模块 show.v

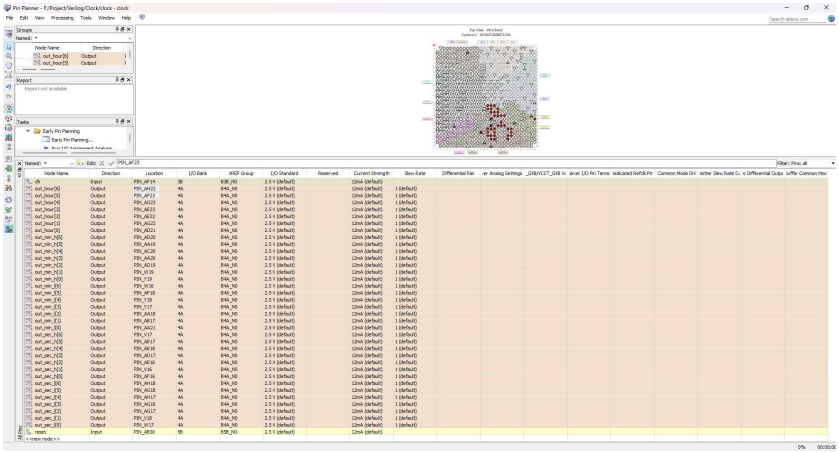
译码模块（show）：实现二进制到七段数码管信号的转换。

```
module show(decodeout, indec);
    input [3:0] indec;
    output [6:0] decodeout;

    reg [6:0] decodeout;

    always @(indec)
    begin
        case (indec)
            4'd0: decodeout = 7'b1000000;
            4'd1: decodeout = 7'b1111001;
            4'd2: decodeout = 7'b0100100;
            4'd3: decodeout = 7'b0110000;
            4'd4: decodeout = 7'b0011001;
            4'd5: decodeout = 7'b0010010;
            4'd6: decodeout = 7'b0000010;
            4'd7: decodeout = 7'b1111000;
            4'd8: decodeout = 7'b0000000;
            4'd9: decodeout = 7'b0010000;
        endcase
    end
endmodule
```

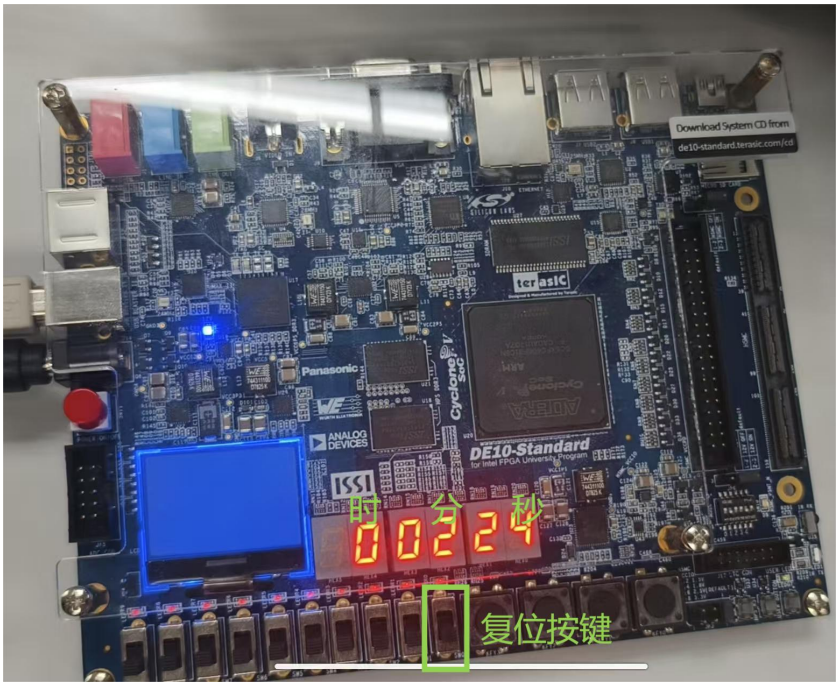
图（9）译码模块 show.v 代码

(5) 引脚绑定

数码管输出端绑定至显示端口。复位引脚为 DE10 板上 SW0 开关。



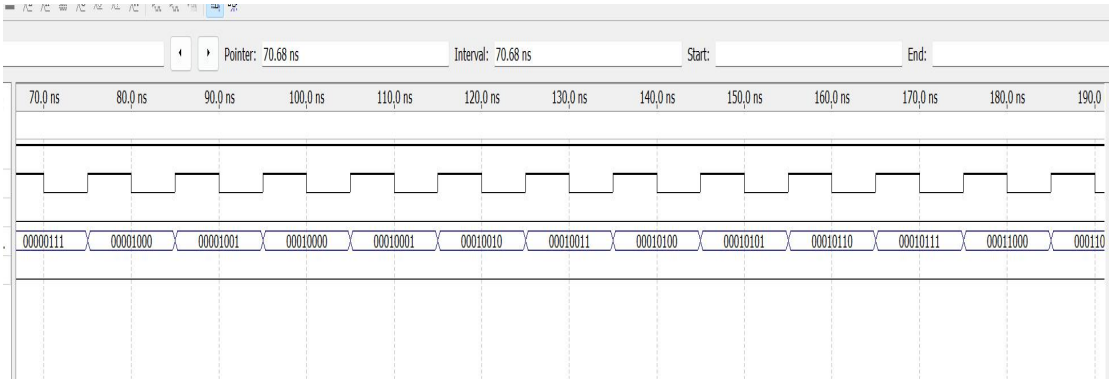
图（10）引脚绑定示意图



图（11）按键说明

三、波形仿真

使用仿真工具（如 Quartus II 中的 University Program VMF）对时钟模块进行仿真，验证其功能，包括秒、分钟、小时的计数逻辑及进位信号。**在仿真中，count60 的输出为 [7:0] 的八位数据，需使用十六进制显示 BCD 码进行观察。实际仿真过程中对部分参数进行了调整，但由于未保存截图，这里特此说明。**

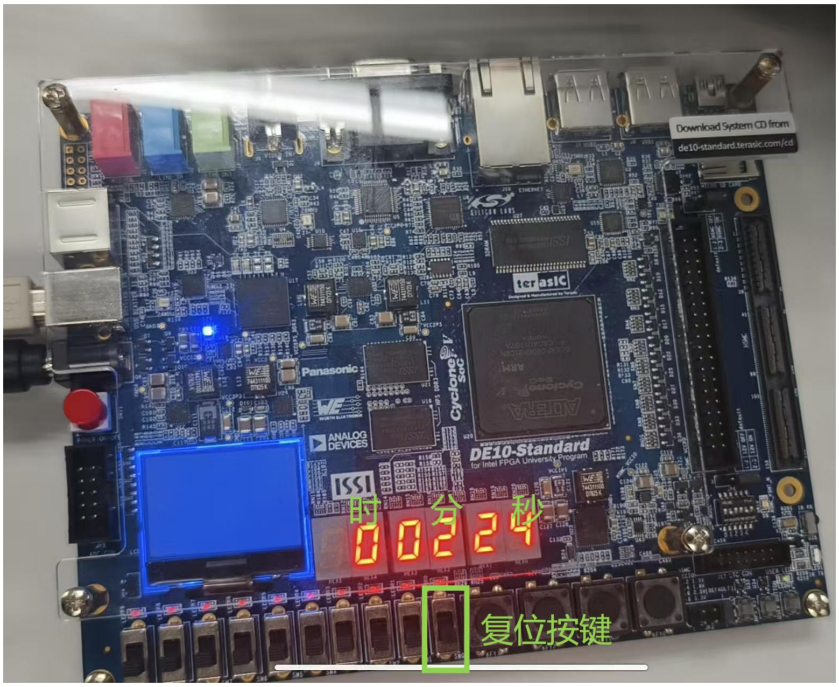


图（12）计数模块仿真波形

四、结果展示

1、 接线说明

通过 FPGA 开发板完成以下接线：时钟信号和复位信号接入开发板输入引脚。数码管的控制信号接入开发板输出引脚。

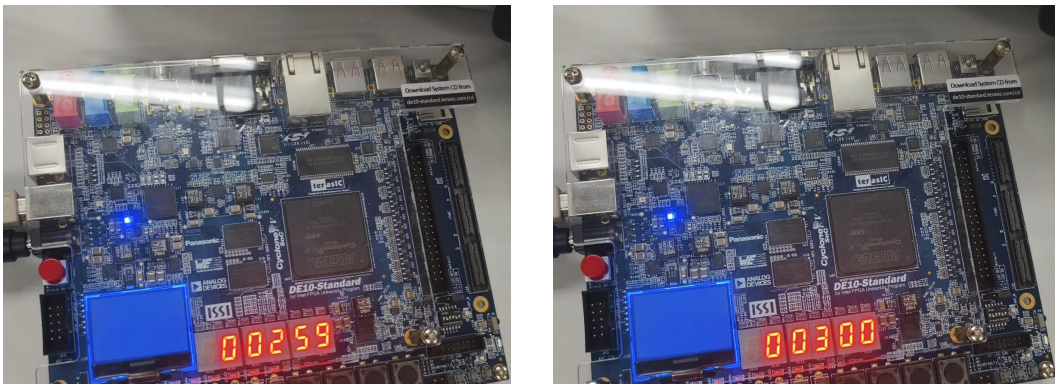


图（11）接线示意图

2、 效果显示

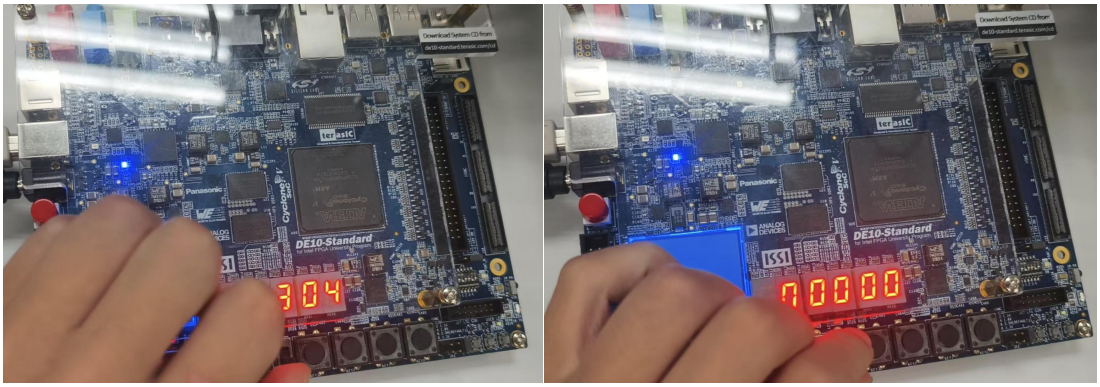
硬件系统运行后，数码管显示时间数据，能正确显示秒、分钟和小时的计时功能。按下复位键，时钟归零。秒、分钟和小时的进位逻辑正常。

以下是进位功能的延时演示。从图中可以看出，系统成功从 02:59 进位到 03:00。



图（12）进位功能演示

接下来是复位功能的演示。第一张图展示了复位开关拨动之前的状态，可以看到数码管显示 03:04。拨动复位开关后，数码管显示变为 00:00，说明复位功能正常。



图（12）复位功能演示

五、心得体会

通过本次实验，我加深了对 Verilog 硬件描述语言的理解，掌握了模块化设计的思路以及 FPGA 数字系统的开发流程。在实验过程中，通过分频、计数和译码模块的设计与整合，我体会到了模块化设计在硬件开发中的重要性，同时解决了数码管显示乱码、复位信号未全局生效等实际问题，进一步提升了硬件调试能力和问题解决能力。本次实验让我认识到仿真与实际硬件调试之间的差异，学会了在实践中结合工具进行高效验证和优化设计，同时增强了独立学习和动手能力。希望今后能接触到更多综合性实验项目，以进一步提升实践能力和电子设计水平。