浙江大学

本科实验报告

课程名称：	数字逻辑设计
姓　　名：	王浩雄
学　　院：	竺可桢学院
系：	混合班
专　　业：	计算机科学与技术
学　　号：	3230106032
指导教师：	马德

2025 年 5 月 22 日

浙江大学实验报告

课程名称：数字逻辑设计 实验类型：

实验项目名称：计数器、定时器设计与应用

学生姓名：王浩雄 专业：混合班 学号：3230106032

同组学生姓名：无 指导老师：马德

实验地点：紫金港东 4-509 实验日期：2025 年 5 月 22 日

一、 实验目的和要求

① 掌握同步四位二进制计数器 74LS161 的工作原理和设 计方法；

② 掌握时钟/定时器的工作原理与设计方法。

二、 实验内容

① 采用行为描述设计同步四位二进制计数器 74LS161

② 基于 74LS161 设计时钟应用

三、 主要仪器设备

① 装有 Vivado 2024.2 Enterprise 的计算机 1 台

② SWORD 开发板 1 套

四、 操作方法与实验步骤

1、采用行为描述设计同步四位二进制计数器 74LS161

① 根据设计要求，新建 Verilog 文件，以行为描述方式完成同步四位二进制计数器模块的设计。其中，输入、输出各信号含义已在下方代码中标出。

```
module My74LS161 (
    input wire clk,         // 时钟
    input wire clr_n,       // 异步清零，低电平有效
    input wire load_n,      // 异步加载，低电平有效
    input wire ctp,
    input wire ctt,
    input wire [3:0] d,     // 并行数据输入
    output reg [3:0] q,     // 四位计数输出
    output wire rco         // 进位输出
);

    // 进位输出
    assign rco = (q == 4'b1111) && ctp && ctt;

    always @(posedge clk or negedge clr_n or negedge load_n) begin
        if (!clr_n) begin
            // 异步清零
            q <= 4'b0000;
        end else if (!load_n) begin
            // 异步加载
            q <= d;
        end else if (ctp && ctt) begin
            // 同步计数
            q <= q + 1;
        end
    end

endmodule
```

② 新建仿真激励文件如下：

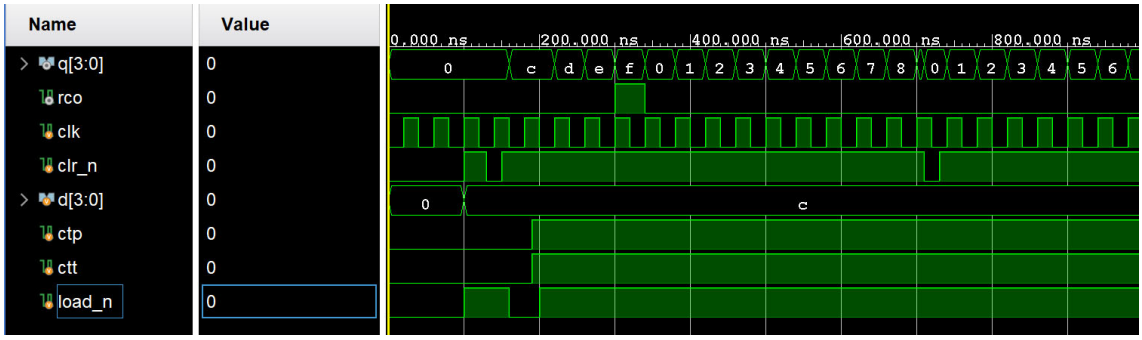
```
module My74LS161_tb;

    reg clk;
    reg clr_n;
    reg load_n;
    reg ctp;
```

```
    reg ctt;
    reg [3:0] d;
    wire [3:0] q;
    wire rco;
    My74LS161 uut (
        .clk(clk),
        .clr_n(clr_n),
        .load_n(load_n),
        .ctp(ctp),
        .ctt(ctt),
        .d(d),
        .q(q),
        .rco(rco)
    );
    // 生成时钟
    initial begin
        clk = 0;
        forever #20 clk = ~clk;
    end

    initial begin
        clr_n = 0;
        d = 0;
        ctp = 0;
        ctt = 0;
        load_n = 0;
        #100;
        clr_n = 1;
        load_n = 1;
        d = 4'b1100;
        ctt = 0;
        ctp = 0;
        #30 clr_n = 0;
        #20 clr_n = 1;
        #10 load_n = 0;
        #30 ctt = 1;
        ctp = 1;
        #10 load_n = 1;
        #510;
        clr_n = 0;
        #20 clr_n = 1;
        #500;
     end
endmodule
```

③ 运行仿真，得到如下的仿真波形。该波形与预期相符，表明计数器设计的正确性。



2、基于 74LS161 设计时钟应用

① 复用 74LS161 模块，编写 60 进制计数器，用于分、秒的计数；编写 24 进制计数器，用于小时的计数。

实现代码如下（仅展示 60 进制计数器代码，24 进制计数器代码与之类似）：

```
module Counter60 (
    input clk,
    input clr_n,
    input enable,
    output [3:0] ones,
    output [3:0] tens,
    output tick
);
    reg clr_n_ones = 1;
    reg clr_n_tens = 1;

    // 判断个位是否达到 9
    wire ones_max = (ones == 4'd9);
    // 判断十位是否达到 5
    wire tens_max = (tens == 4'd5);

    // 计数完成
    assign tick = (ones_max && tens_max && enable);


    My74LS161 cnt_ones (
        .clk(clk),
        .clr_n(clr_n & clr_n_ones),
        .load_n(1'b1),
        .ctp(enable),
```

```
        .ctt(1'b1),
        .d(4'b0000),
        .q(ones),
        .rco()
    );

    My74LS161 cnt_tens (
        .clk(clk),
        .clr_n(clr_n & clr_n_tens),
        .load_n(1'b1),
        .ctp(enable & ones_max),
        .ctt(1'b1),
        .d(4'b0000),
        .q(tens),
        .rco()
    );

    always @(posedge clk or negedge clr_n) begin
        if (!clr_n) begin
            clr_n_ones <= 1;
            clr_n_tens <= 1;
        end else begin
            // 个位到 9，下一拍清零个位
            if (ones_max && enable) begin
                clr_n_ones <= 0;
            end else begin
                clr_n_ones <= 1;
            end

            // 十位到 5 且个位刚清零时清零十位
            if (tens_max && ones_max && enable) begin
                clr_n_tens <= 0;
            end else begin
                clr_n_tens <= 1;
            end
        end
    end
endmodule
```

② 完成顶层模块 top.v

```
module Top (
    input wire clk,
    input wire [15:0] SW,
    output wire SEG_DT,
```

```
    output wire SEG_CLK,
    output wire SEG_CLR,
    output wire SEG_EN
);

    wire [31:0] clkdiv;
    wire tick_sec, tick_min, seg_driver_start;
    wire [31:0] num_SEG;
    wire [3:0] hour_tens;
    wire [3:0] hour_ones;
    wire [3:0] min_tens;
    wire [3:0] min_ones;
    wire [3:0] sec_tens;
    wire [3:0] sec_ones;


    // 秒计数器
    Counter60 u_sec (
        .clk(clkdiv[23]),
        .clr_n(~SW[0]),
        .enable(1'b1),
        .ones(sec_ones),
        .tens(sec_tens),
        .tick(tick_sec)
    );

    // 分计数器
    Counter60 u_min (
        .clk(clkdiv[23]),
        .clr_n(~SW[0]),
        .enable(tick_sec),
        .ones(min_ones),
        .tens(min_tens),
        .tick(tick_min)
    );

    // 时计数器
    Counter24 u_hour (
        .clk(clkdiv[23]),
        .clr_n(~SW[0]),
        .enable(tick_min),
        .ones(hour_ones),
        .tens(hour_tens),
        .tick()
```

```
    );

    assign num_SEG =
{8'b00000000,hour_tens,hour_ones,min_tens,min_ones,sec_tens,sec_ones};
    clkdiv u1(.clk(clk),.rst(SW[0]),.clkdiv(clkdiv));
    switch_edge_pulse
u2(.clk(clk),.sw(clkdiv[23]),.pulse(seg_driver_start));
    seg_driver
u3(.clk(clk),.reset(SW[0]),.start(seg_driver_start),.num(num_SEG),.SEG_
DT(SEG_DT),.SEG_CLR(SEG_CLR),.SEG_CLK(SEG_CLK),.SEG_EN(SEG_EN));

endmodule
```

③ 对顶层模块进行仿真，编写仿真激励文件如下：

```
module Top_tb;

    reg clk;
    reg [15:0] SW;
    wire SEG_DT;
    wire SEG_CLK;
    wire SEG_CLR;
    wire SEG_EN;

    Top uut (
        .clk(clk),
        .SW(SW),
        .SEG_DT(SEG_DT),
        .SEG_CLK(SEG_CLK),
        .SEG_CLR(SEG_CLR),
        .SEG_EN(SEG_EN)
    );

    always #5 clk = ~clk;

    initial begin
        clk = 0;
        SW = 16'b0;

        SW[0] = 1'b1;
        #100;

        SW[0] = 1'b0;
    end
endmodule
```

④ 运行仿真，得到如下的仿真波形。重点观察仿真波形的进位动作，该波形与预期相符，表明计数器设计的正确性。

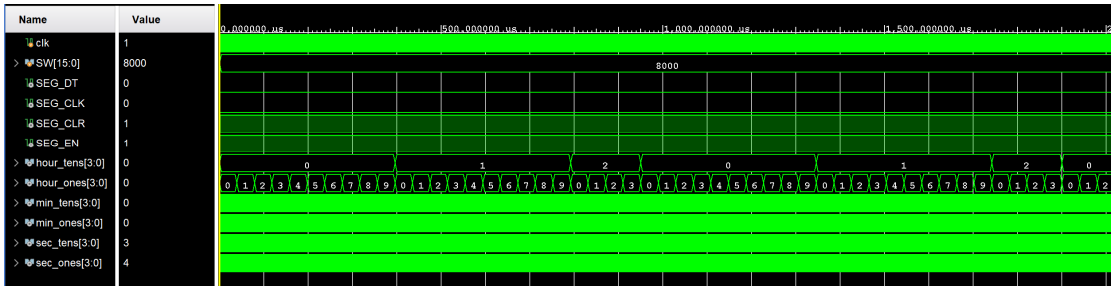


图 1：小时在到达 23 后重置为 0，设计正确

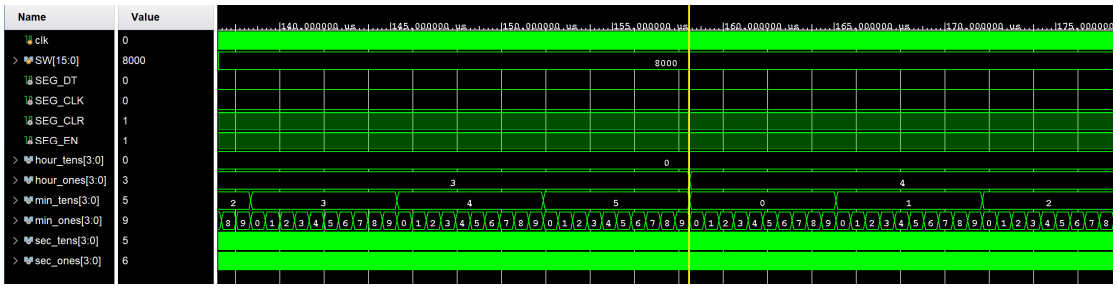


图 2：分钟和秒在到达 59 后重置为 0，设计正确

⑤ 下板，进行复位操作后，数码管显示稳定，从低位到高位分别显示秒、分钟、小时信息，表明计数器设计的正确性。

图 3:00 时 00 分 18 秒

图 4:00 时 01 分 29 秒

五、 讨论、心得

在本次实验中，我尝试实现了 4 位同步二进制计数器模块，在此基础上使用上一次实验设计的七段码驱动模块，实现了一个简易的 24 小时计时器，并成功通过下板验证其正确性。本次实验使我对计数器的设计原理及其实际应用的理解和认识大为加深。此外，通过自己编写顶层模块，我对 Verilog 语言的熟悉程度进一步增强，对分层次、模块化设计的方法更为熟练。