杭州电子科技大学
HANGZHOU DIANZI UNIVERSITY

电子信息虚拟仿真实验报告

实验 1: 动态扫描显示电路

学院 卓越学院

学号 23040447

姓名 陈文轩

专业 智能硬件与系统 (电子信息工程)

2025 年 3 月 16 日

目录

1 顶层电路设计 (verilog) 1

1.1 实验 1 1

1.2 实验 2 3

1.3 实验 3 3

2 远程平台操作截图 4

2.1 实验 1: 学号显示 4

2.2 实验 2: 计数器跳变截图 5

2.3 实验 3: 数字钟 6

3 实验设计过程简要介绍 7

3.1 实验 1: 学号显示 7

3.1.1 模块接口 7

3.1.2 实现细节 8

3.1.3 主要功能模块 8

3.2 实验 2: 计数器 8

3.3 实验 3: 数字钟 9

1 顶层电路设计 (verilog)

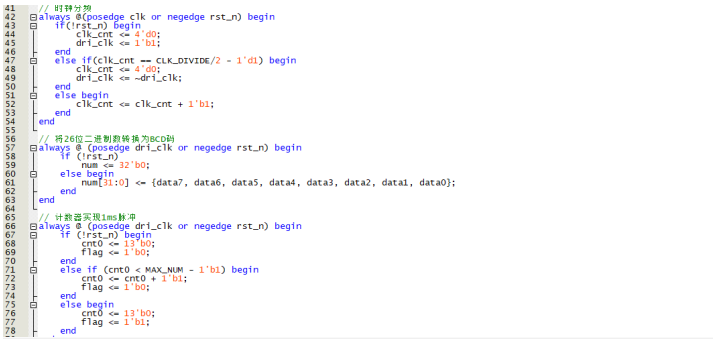
1.1 实验 1

```
module top_seg_led(
    //global clock
    input            sys_clk   ,        // 全局时钟信号
    input            sys_rst_n,        // 复位信号（低有效）

    //seg_led interface
    output    [7:0]  seg_sel   ,        // 数码管位选信号
    output    [7:0]  seg_led    ,       // 数码管段选信号
);
 //wire define
 wire    [32:0]  data;                  // 数码管显示的数值
 wire    [ 7:0]  point;                 // 数码管小数点的位置
 wire            en;                    // 数码管显示使能信号
 wire            sign;                  // 数码管显示数据的符号位


 assign data=23040447;
 assign en=1'b1;
 assign point=6'b000000;
 assign sign=1'b0;
 //数码管动态显示模块
seg_led u_seg_led(
    .clk          (sys_clk  ),        // 时钟信号
    .rst_n        (sys_rst_n),        // 复位信号

    .data         (data     ),        // 显示的数值
    .point        (point    ),        // 小数点具体显示的位置,高电平有效
    .en           (en       ),        // 数码管使能信号
    .sign         (sign     ),        // 符号位，高电平显示负号(-)

    .seg_sel      (seg_sel  ),        // 位选
    .seg_led      (seg_led  )         // 段选
);

 endmodule
```

图 1 实验 1：顶层电路设计

```
module seg_led(
    input                   clk     ,       // 时钟信号
    input                   rst_n   ,       // 复位信号

    input         [32:0]    data    ,       // 8位数码管要显示的数值
    input         [7:0]     point   ,       // 小数点具体显示的位置,从高到低,高电平有效
    input                   en      ,       // 数码管使能信号
    input                   sign    ,       // 符号位（高电平显示"-"号）

    output   reg  [7:0]     seg_sel,        // 数码管位选，最左侧数码管为最高位
    output   reg  [7:0]     seg_led         // 数码管段选
    );

//parameter define
localparam  CLK_DIVIDE = 4'd10      ;       // 时钟分频系数
localparam  MAX_NUM    = 13'd5000   ;       // 对数码管驱动时钟(5MHz)计数1ms所需的计数值

//reg define
reg    [ 3:0]           clk_cnt  ;       // 时钟分频计数器
reg                     dri_clk  ;       // 数码管的驱动时钟,5MHz
reg    [31:0]           num      ;       // 32位bcd码寄存器
reg    [12:0]           cnt0     ;       // 数码管驱动时钟计数器
reg                     flag     ;       // 标志信号（标志着cnt0计数达1ms）
reg    [2:0]            cnt_sel  ;       // 数码管位选计数器
reg    [3:0]            num_disp ;       // 当前数码管显示的数据
reg                     dot_disp ;       // 当前数码管显示的小数点

//wire define
wire   [3:0]            data0, data1, data2, data3, data4, data5, data6, data7;

// 提取各个位的BCD数值
assign  data0 = data % 10;
assign  data1 = (data / 10) % 10;
assign  data2 = (data / 100) % 10;
assign  data3 = (data / 1000) % 10;
assign  data4 = (data / 10000) % 10;
assign  data5 = (data / 100000) % 10;
assign  data6 = (data / 1000000) % 10;
assign  data7 = (data / 10000000) % 10;
```

图 2 实验 1：数码管显示模块



```
// 时钟分频
always @(posedge clk or negedge rst_n) begin
    if(!rst_n) begin
        clk_cnt <= 4'd0;
        dri_clk <= 1'b1;
    end
    else if(clk_cnt == CLK_DIVIDE/2 - 1'd1) begin
        clk_cnt <= 4'd0;
        dri_clk <= ~dri_clk;
    end
    else begin
        clk_cnt <= clk_cnt + 1'b1;
    end
end

// 将26位二进制数转换为BCD码
always @ (posedge dri_clk or negedge rst_n) begin
    if (!rst_n)
        num <= 32'b0;
    else begin
        num[31:0] <= {data7, data6, data5, data4, data3, data2, data1, data0};
    end
end

// 计数器实现1ms脉冲
always @ (posedge dri_clk or negedge rst_n) begin
    if (!rst_n) begin
        cnt0 <= 13'b0;
        flag <= 1'b0;
    end
    else if (cnt0 < MAX_NUM - 1'b1) begin
        cnt0 <= cnt0 + 1'b1;
        flag <= 1'b0;
    end
    else begin
        cnt0 <= 13'b0;
        flag <= 1'b1;
    end
```

图 3 实验 1：数码管显示模块

```
// 计数器循环选择8个数码管
always @ (posedge dri_clk or negedge rst_n) begin
    if (!rst_n)
        cnt_sel <= 3'b0;
    else if(flag) begin
        if(cnt_sel < 3'd7)
            cnt_sel <= cnt_sel + 1'b1;
        else
            cnt_sel <= 3'b0;
    end
end

// 控制数码管位选信号
always @ (posedge dri_clk or negedge rst_n) begin
    if (!rst_n) begin
        seg_sel  <= 8'b11111111;
        num_disp <= 4'b0;
        dot_disp <= 1'b1;
    end
    else if (en) begin
        seg_sel  <= ~(8'b00000001 << cnt_sel);
        num_disp <= num[(cnt_sel * 4) +: 4];
        dot_disp <= ~point[cnt_sel];
    end
    else begin
        seg_sel  <= 8'b11111111;
        num_disp <= 4'b0;
        dot_disp <= 1'b1;
    end
end

// 控制数码管段选信号
always @ (posedge dri_clk or negedge rst_n) begin
    if (!rst_n)
        seg_led <= 8'hc0;
    else begin
        case (num_disp)
```

图 4 实验 1：数码管显示模块

```
// 控制数码管段选信号
always @ (posedge dri_clk or negedge rst_n) begin
    if (!rst_n)
        seg_led <= 8'hc0;
    else begin
        case (num_disp)
            4'd0 : seg_led <= {dot_disp,7'b1000000};
            4'd1 : seg_led <= {dot_disp,7'b1111001};
            4'd2 : seg_led <= {dot_disp,7'b0100100};
            4'd3 : seg_led <= {dot_disp,7'b0110000};
            4'd4 : seg_led <= {dot_disp,7'b0011001};
            4'd5 : seg_led <= {dot_disp,7'b0010010};
            4'd6 : seg_led <= {dot_disp,7'b0000010};
            4'd7 : seg_led <= {dot_disp,7'b1111000};
            4'd8 : seg_led <= {dot_disp,7'b0000000};
            4'd9 : seg_led <= {dot_disp,7'b0010000};
            default: seg_led <= 8'b11111111;
        endcase
    end
end

endmodule
```

图 5 实验 1：数码管显示模块

1.2 实验 2

```
//
module top_seg_led(
    //global clock
    input           sys_clk  ,      // 全局时钟信号
    input           sys_rst_n,      // 复位信号（低有效）

    //seg_led interface
    output    [7:0] seg_sel  ,      // 数码管位选信号
    output    [7:0] seg_led         // 数码管段选信号
);

//wire define
wire    [32:0] data;                // 数码管显示的数值
wire    [ 7:0] point;               // 数码管小数点的位置
wire           en;                  // 数码管显示使能信号
wire           sign;                // 数码管显示数据的符号位

//*****************************************************
//**                    main code
//*****************************************************

//计数器模块，产生数码管需要显示的数据
reg [25:0] div_counter; // 分频计数器
reg [5:0] counter; // 6位计数器，范围0-63

always @(posedge sys_clk or negedge sys_rst_n) begin
    if (!sys_rst_n) begin
        div_counter <= 26'd0;
        counter <= 6'd0;
    end else if (div_counter == 26'd25_000_000) begin // 50MHz时钟，0.5秒分频
        div_counter <= 26'd0;
        if (counter == 6'd60)
            counter <= 6'd0;
        else
            counter <= counter + 1'b1;
    end else begin
        div_counter <= div_counter + 1'b1;
    end
end

assign data = {27'd0, counter}; // 将计数器值赋给data信号，高位补0


assign en=1'b1;
assign point=6'b000000;
assign sign=1'b0;
//数码管动态显示模块
seg_led u_seg_led(
    .clk            (sys_clk  ),        // 时钟信号
    .rst_n          (sys_rst_n),        // 复位信号

    .data           (data     ),        // 显示的数值
    .point          (point    ),        // 小数点具体显示的位置,高电平有效
    .en             (en       ),        // 数码管使能信号
    .sign           (sign     ),        // 符号位，高电平显示负号(-)

    .seg_sel        (seg_sel  ),        // 位选
    .seg_led        (seg_led  )         // 段选
);

//assign seg_led[7:0]=8'b10110000;
//assign seg_sel[5:0]=8'b101010;

endmodule
```

图 6 实验 2：顶层电路设计

1.3 实验 3

```
module top_seg_led(
    //global clock
    input           sys_clk  ,      // 全局时钟信号
    input           sys_rst_n,      // 复位信号（低有效）
    input     [1:0] key ,
    //seg_led interface
    output    [7:0] seg_sel  ,      // 数码管位选信号
    output    [7:0] seg_led         // 数码管段选信号
);

//wire define
wire    [32:0] data;                // 数码管显示的数值
wire    [ 7:0] point;               // 数码管小数点的位置
wire           en;                  // 数码管显示使能信号
wire           sign;                // 数码管显示数据的符号位

//*****************************************************
//**                    main code
//*****************************************************

//计数器模块，产生数码管需要显示的数据
reg [31:0] time_counter; // 用于计时的计数器
reg [31:0] total_seconds; // 统一的秒数计数器
// 时间调整逻辑
always @(posedge sys_clk or negedge sys_rst_n) begin
    if (!sys_rst_n) begin
        time_counter <= 0;
        total_seconds <= 12*3600; // 初始化为12:00:00
    end else begin
        time_counter <= time_counter + 1;
        if (time_counter == 50000000) begin // 每秒钟更新一次时间（假设时钟频率为25MHz）
            time_counter <= 0;
            case (key)
            2'b11: total_seconds <= (total_seconds >= 86400)?0:total_seconds + 3600; // 按下是低电平
            2'b10: total_seconds <= (total_seconds >= 86400)?0:total_seconds + 60;
            2'b01: total_seconds <= (total_seconds >= 86400)?0:total_seconds + 2;
            2'b00: total_seconds <= total_seconds + 1;
            default:total_seconds <= total_seconds + 1;
             endcase
//            if (total_seconds == 86400) begin // 一天86400秒
//                total_seconds <= 0;
//            end
        end

    end
end
// 通过组合逻辑将秒数分解为小时、分钟和秒
wire [4:0] hours = total_seconds / 3600;
wire [5:0] minutes = (total_seconds % 3600) / 60;
wire [5:0] seconds = total_seconds % 60;
```

图 7 实验 3：顶层电路设计

```
// 将 hours、minutes 和 seconds 拼接成一个 32 位的信号
assign data = hours*10000+minutes*100+seconds;

assign en=1'b1;
assign point=8'b00010100;
assign sign=1'b0;
//数码管动态显示模块
seg_led u_seg_led(
    .clk            (sys_clk  ),        // 时钟信号
    .rst_n          (sys_rst_n),        // 复位信号

    .data           (data     ),        // 显示的数值
    .point          (point    ),        // 小数点具体显示的位置,高电平有效
    .en             (en       ),        // 数码管使能信号
    .sign           (sign     ),        // 符号位，高电平显示负号(-)

    .seg_sel        (seg_sel  ),        // 位选
    .seg_led        (seg_led  )         // 段选
);

//assign seg_led[7:0]=8'b10110000;
//assign seg_sel[5:0]=8'b101010;

endmodule
```

图 8 实验 3：顶层电路设计

2　远程平台操作截图

2.1　实验 1: 学号显示

图 9 实验 1：远程平台操作截图

2.2 实验 2: 计数器跳变截图

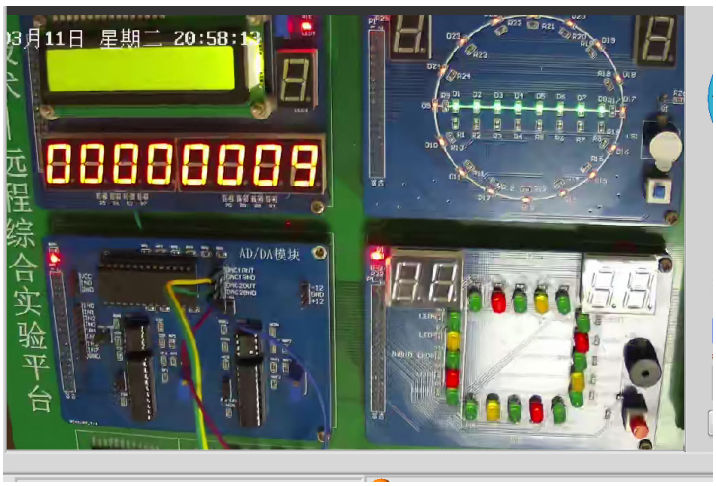


图 10 实验 2：远程平台操作截图 1

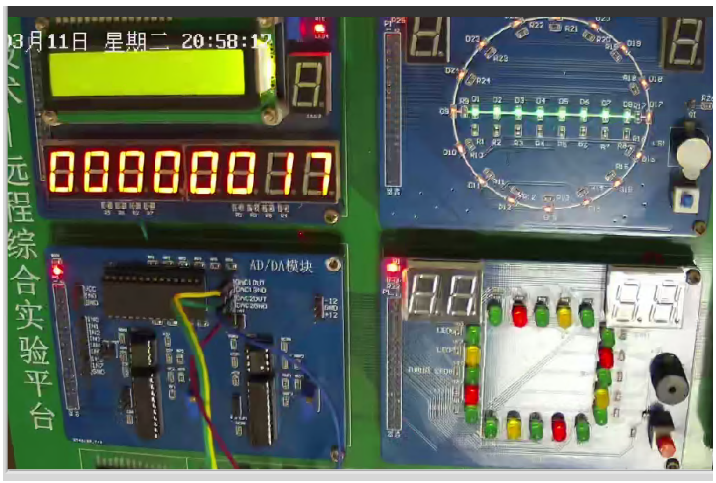


图 11 实验 2：远程平台操作截图 2

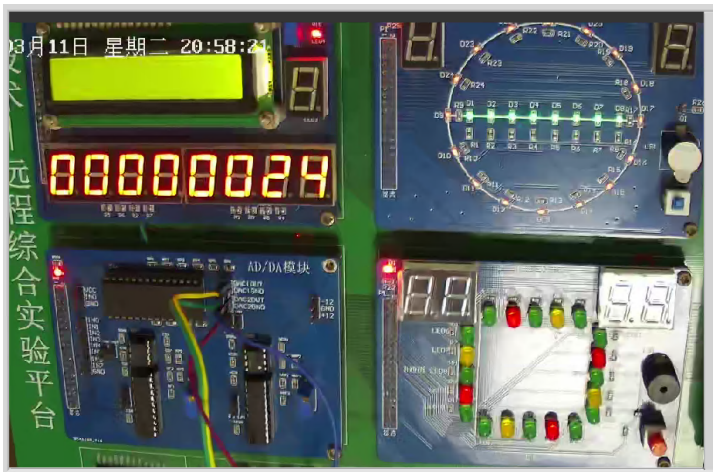


图 12 实验 2：远程平台操作截图 3

2.3 实验 3: 数字钟

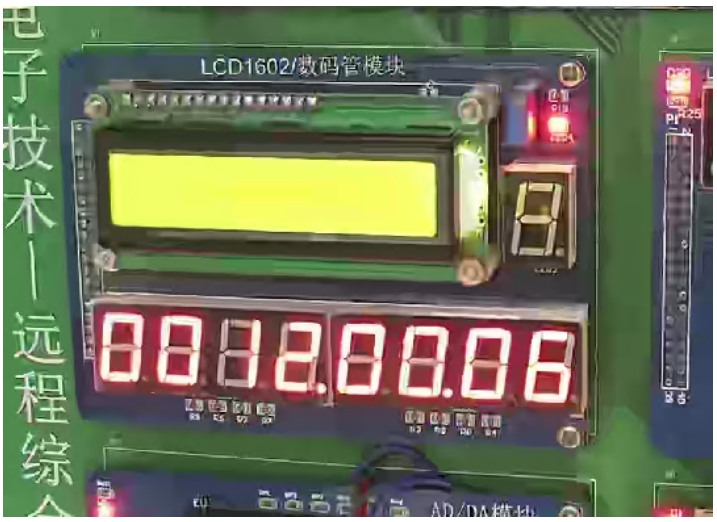


图 13 实验 3: 显示 12:00:06

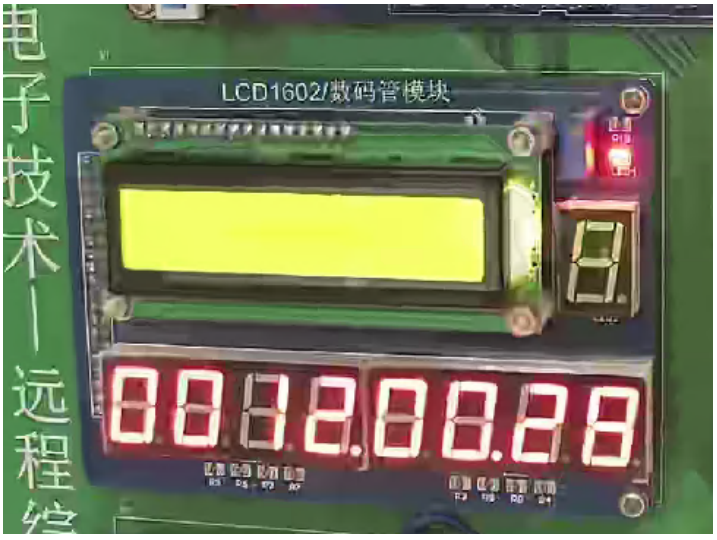


图 14 实验 3: 显示 12:00:28

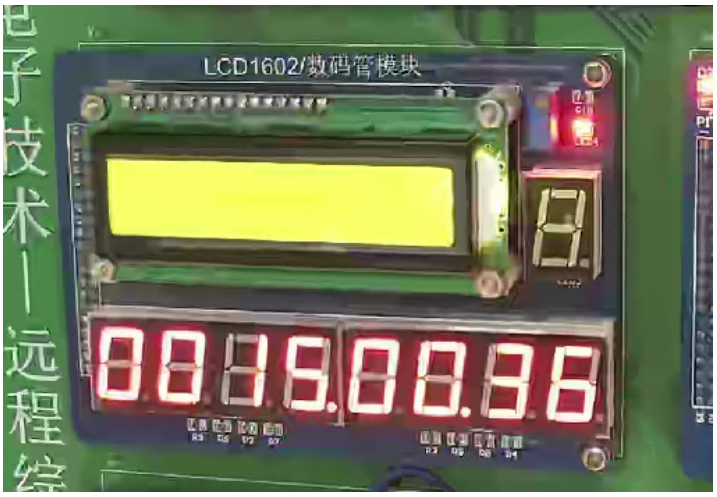


图 15 实验 3: 显示 15:00:36

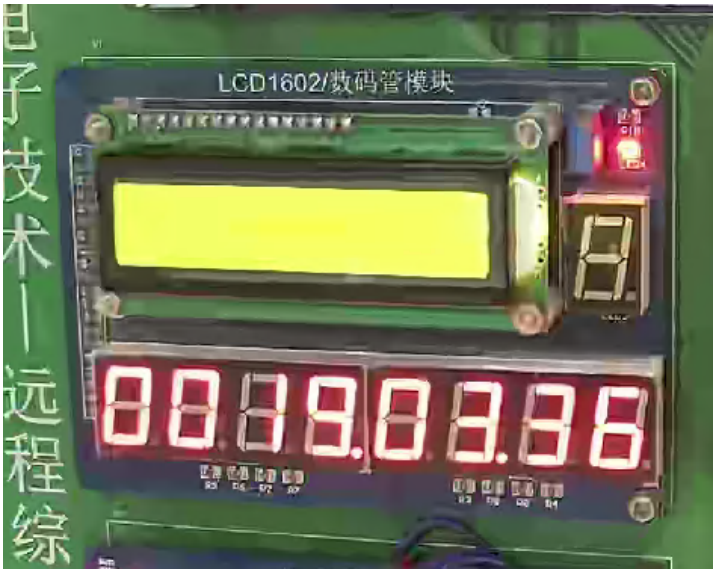


图 16 实验 3: 显示 19:03:06

3 实验设计过程简要介绍

3.1 实验 1: 学号显示

这个实验的目的是设计一个数码管显示模块 `seg_led`，该模块能够显示输入的数值，并且支持小数点显示和符号显示。模块的主要功能包括时钟分频、数值转换、数码管位选和段选控制。

3.1.1 模块接口

- 输入信号

  - clk：时钟信号
  - rst_n：复位信号（低电平有效）
  - data：要显示的数值（32 位）
  - point：小数点显示位置（8 位，高电平有效）
  - en：数码管使能信号
  - sign：符号位（高电平显示“-”号）
- 输出信号
  - seg_sel：数码管位选信号（8 位）
  - seg_led：数码管段选信号（8 位）

3.1.2 实现细节

- 时钟分频:通过时钟分频器将输入时钟信号分频为数码管驱动时钟 dri_clk。
- 数值转换：将输入的 32 位二进制数值转换为 BCD 码，以便数码管显示。
- 计数器：使用计数器生成 1ms 脉冲信号，并循环选择 8 个数码管进行显示。
- 位选控制：根据计数器的值控制数码管的位选信号 seg_sel。
- 段选控制:根据当前显示的数值和小数点位置控制数码管的段选信号 seg_led。

3.1.3 主要功能模块

- 时钟分频模块：将输入时钟信号分频为数码管驱动时钟。
- 数值转换模块：将输入的 32 位二进制数值转换为 BCD 码。
- 计数器模块：生成 1ms 脉冲信号，并循环选择数码管。
- 位选控制模块：控制数码管的位选信号。
- 段选控制模块：控制数码管的段选信号。

通过这个实验，可以掌握数码管显示的基本原理和实现方法，包括时钟分频、数值转换、位选和段选控制等。

3.2 实验 2: 计数器

数码管动态显示的原理同实验 1。

3.3 实验 3: 数字钟

这个实验的目的是设计一个数字钟，通过数码管显示当前的时间。实验的主要步骤包括时钟信号的分频、时间的计数和显示控制。具体实现细节与实验 1 类似，但需要增加时间计数和显示的逻辑。