浙江大学

本科实验报告

课程名称：计算机逻辑设计基础

姓　　名：仇国智

学　　院：竺可桢学院

专　　业：计算机科学与技术

学　　号：3220102181

指导教师：董亚波

2023 年 12 月 14 日

一、 实验目的和要求

1　实验目的

1.1　掌握支持并行输入的移位寄存器的工作原理

1.2　掌握支持并行输入的移位寄存器的设计方法

2　实验任务

2.1　任务 1：设计 8 位带并行输入的右移移位寄存器

2.2　任务 2：设计主板 16 位 LED 灯驱动模块

2.3　任务 3：设计主板 8 位数码管驱动模块

二、 实验内容和原理

1　实验内容

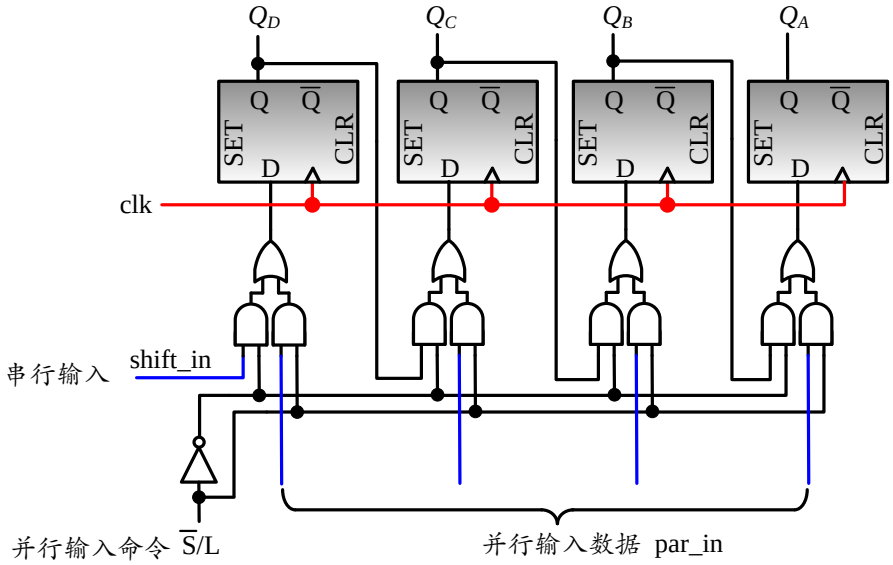
1.1　任务 1：设计 8 位带并行输入的右移移位寄存器

1.2　任务 2：设计主板 16 位 LED 驱动模块

1.3　任务 3：设计主板 8 位数码管驱动模块

2　实验原理

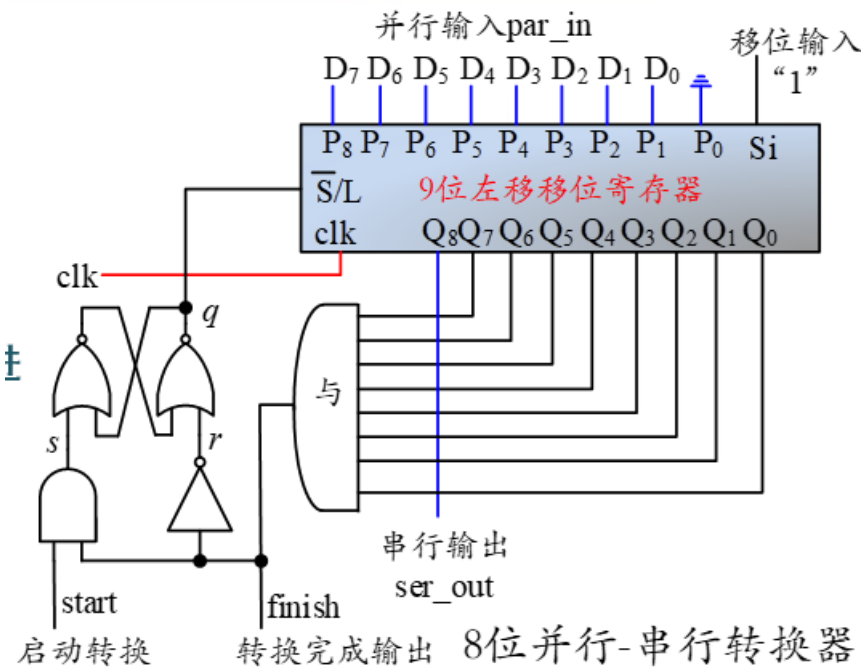
2.1　移位寄存器：每来一个时钟脉冲，寄存器中的数据按顺序向左或向右移动一位，必须采用主从触发器或边沿触发器，不能采用锁存器。数据移动方式：左移、右移、循环移位。数据输入输出方式：串行输入，串行输出，串行输入，并行输出，并行输入，串行输出。



2.2　实验板 16 位 LED 灯：实验板上，采用 2 个 8 位移位寄存器 74LV164A 构成 16 位串行左移移位寄存器，寄存器的并行输出控制 16 个 LED 灯 LED0-LED15。LED_CLK：16 位 LED 灯模

块的时钟，上升沿触发移位。LED_CLR：清零，使所有 LED 亮，低电平有效。LED_DO：串行移位数据输入，0 使 LED 亮。LED_EN：LED 模块总控开关，1 为使能。用 LED_CLK 和 LED_DO 按顺序 LED15，LED14，……，LED1，LED0 串行移入 16 位数据。

2.3 将需要显示的 16 位二进制数 num[15:0]从 LED_DT 引脚左移输出到 16 位 LED 灯模块，同时在 LED_CLK 引脚上提供 16 个周期的时钟。16 位数据移位完成后要停止时钟，避免把之前的数据移出 16 位 LED 灯模块，采用门控时钟方式给 LED_CLK 提供时钟：assign LED_CLK = clk | finish,finish 为转换结束标志，为 1 表示转换结束。



2.4 主板上 8 位数码管显示采用的是静态显示，不是动态扫描方式，实验板上用 8 个 74LV164A 构成 64 位串-并转换模块，并行输出控制 8 个 7 段数码管，通过 SEGCLR 和 SEGDT 串行接收 8 个数码管*8 段码，共计 64 位数据，移位先后顺序为 SEG7_DP，

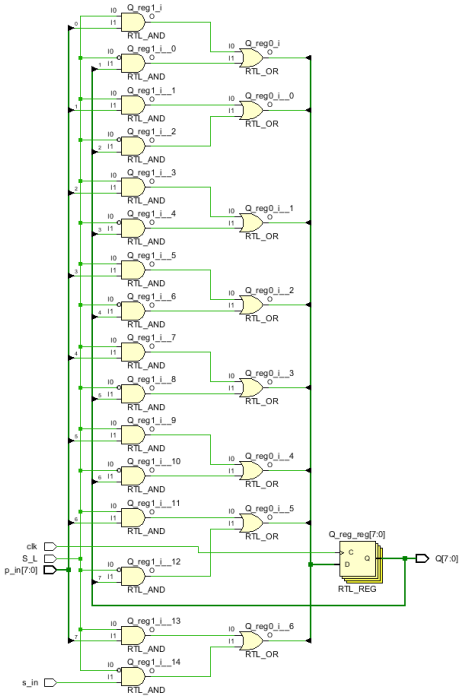
SEG7_g，SEG7_f，……，SEG0_b，SEG0_a。数码管共阳接法，段码为 0 时对应段亮，参考 16 位 LED 驱动模块，扩展设计 8 位数码管驱动模块。SEGCLK：8 位七段数码管模块的时钟，上升沿触发移位，SEGCLR：清零，所有段亮，低电平有效，SEGDT：串行移位数据输入，0 亮，SEGEN：8 位七段数码管模块总控开关，1 为使能，用 SEGCLR 和 SEGDT 按顺序 SEG7_DP，SEG7_g，SEG7_f，……，SEG0_b，SEG0_a 串行移入 64 位数据。

三、 实验过程和数据记录

1　任务 1：设计 8 位带并行输入的右移移位寄存器

1.1 新建工程，工程名称用 ShfitReg8b。Top Level Source Type 用 HDL，用结构化描述设计。

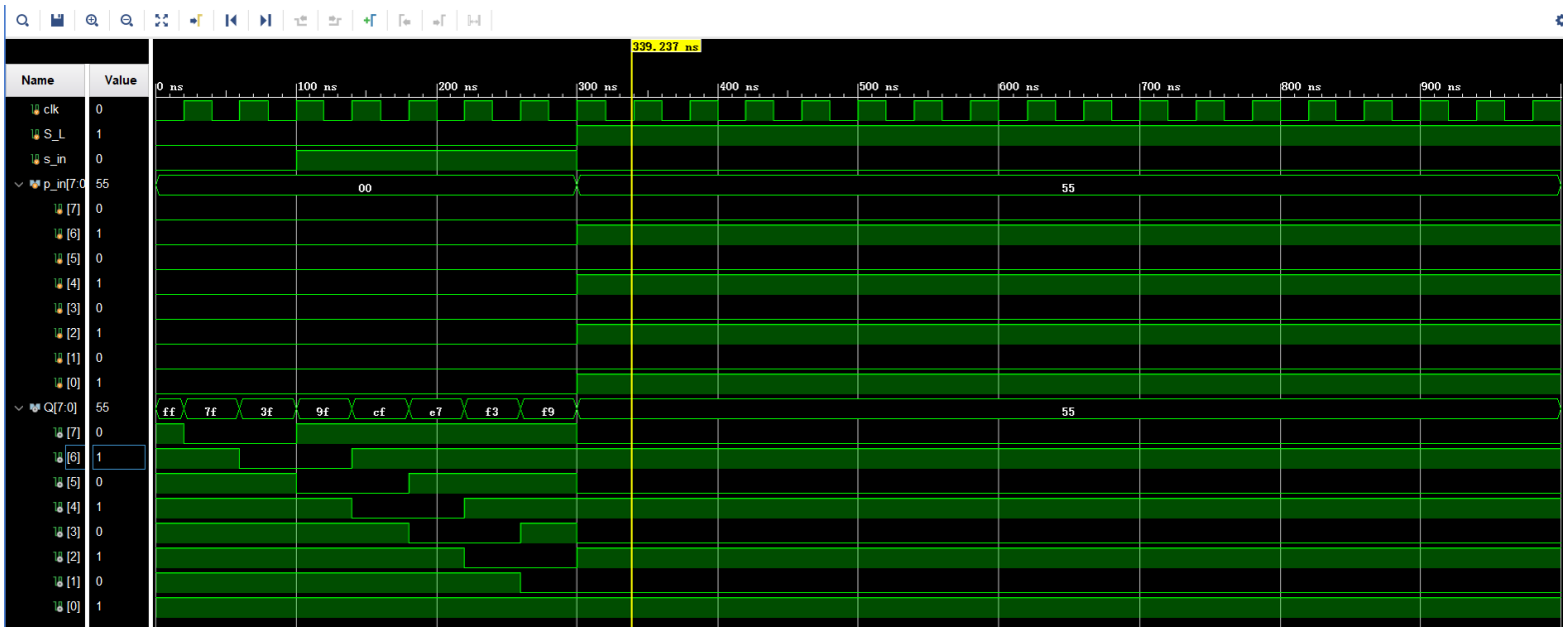
```
`timescale 1ns / 1ps
module shift_reg (
    input wire clk,
    S_L,
    s_in,
    input wire [7:0] p_in,
    output wire [7:0] Q
);
  reg [7:0] Q_reg=8'b11111111;
  assign Q[7:0] = Q_reg[7:0];
  always @(posedge clk) begin
    Q_reg[0] <= (S_L & p_in[0]) | (~S_L & Q_reg[1]);
    Q_reg[1] <= (S_L & p_in[1]) | (~S_L & Q_reg[2]);
    Q_reg[2] <= (S_L & p_in[2]) | (~S_L & Q_reg[3]);
    Q_reg[3] <= (S_L & p_in[3]) | (~S_L & Q_reg[4]);
    Q_reg[4] <= (S_L & p_in[4]) | (~S_L & Q_reg[5]);
    Q_reg[5] <= (S_L & p_in[5]) | (~S_L & Q_reg[6]);
    Q_reg[6] <= (S_L & p_in[6]) | (~S_L & Q_reg[7]);
    Q_reg[7] <= (S_L & p_in[7]) | (~S_L & s_in);
  end
endmodule
```



1.2 波形仿真

```
`timescale 1ns / 1ps
module shift_reg_test1(

    );
    reg clk, S_L, s_in;
    reg[7:0] p_in;
    wire [7:0] Q;
    shift_reg test1 (
        .clk(clk),
        .S_L(S_L),
        .s_in(s_in),
        .p_in(p_in),
        .Q(Q[7:0])
    );
    initial begin
        // Initialize Inputs
        S_L = 0;
        s_in = 0;
        p_in = 0;

        #100;

        // Add stimulus here
        S_L = 0;
```

```
        s_in = 1;
        p_in =0;
        #200;
        S_L = 1;
        s_in = 0;
        p_in = 8'b0101_0101;
        #500;
    end
    always begin
        clk = 0; #20;
        clk = 1; #20;
    end

endmodule
```



2　任务 2：设计主板 16 位 LED 驱动模块

2.1 编写 verilog 源文件设计

top1 模块：

```
module top1 (
    input wire [3:0] btn,
    input wire clk,
    SW,
    output wire LED_CLK,
    LED_CLR,
```

```
    LED_DO,
    LED_EN,
    BTNX4,
    output wire [3:0] AN,
    output wire [7:0] SEGMENT
);
wire temp;
  assign BTNX4   = 0;
  assign LED_CLR = 1;
  assign LED_EN  = 1;
  wire finish;
  wire [15:0] num;
  DispNum d_0 (
      .clk(clk),
      .HEXS(num),
      .LES(4'b0),
      .points(4'b0),
      .RST(1'b0),
      .AN(AN),
      .Segment(SEGMENT)
  );
  reg_4_4b d_1 (
      .clk(clk),
      .increase(btn),
      .num(num)
  );
  LED_DRV d_2 (
      .clk(clk),
      .sw(SW),
      .num(num),
      .finish(finish),
      .ser_out(temp)
  );
  assign LED_DO=~temp;
  assign LED_CLK = ~clk | finish;
endmodule
```

LED_DRV 模块:

```
module LED_DRV (
    input wire clk,
    sw,
    input wire [15:0] num,
    output wire finish,
    ser_out
```

```
);
  wire S_L, nS_L;
  wire [31:0] clk_1ms;
  wire start ;
  wire [16:0] O;
  clkdiv d_0 (
      .clk(clk),
      .rst(1'b0),
      .clkdiv(clk_1ms)
  );
  SLReg17b d_1 (
      .clk(clk),
      .S_L(S_L),
      .s_in(1'b1),
      .p_in({num, 1'b0}),
      .Q(O)
  );
  assign finish = &O[15:0];
  assign S_L = finish & start;
  assign ser_out = O[16];
  Load_Gen d_2 (
      .clk(clk),
      .clk_1ms(clk_1ms[17]),
      .btn_in(sw),
      .Load_out(start)
  );
endmodule
```

SLReg17b 模块：

```
`timescale 1ns / 1ps
module SLReg17b (
    input wire clk,
    S_L,
    s_in,
    input wire [16:0] p_in,
    output wire [16:0] Q
);
  reg [16:0] Q_reg;
  initial begin
    Q_reg = 17'b1111_1111_1111_1111_1;
  end
  assign Q = Q_reg;
  always @(posedge clk) begin
    Q_reg[0]  <= (S_L & p_in[0]) | (~S_L & s_in);
```

```
    Q_reg[1]  <= (S_L & p_in[1]) | (~S_L & Q_reg[0]);
    Q_reg[2]  <= (S_L & p_in[2]) | (~S_L & Q_reg[1]);
    Q_reg[3]  <= (S_L & p_in[3]) | (~S_L & Q_reg[2]);
    Q_reg[4]  <= (S_L & p_in[4]) | (~S_L & Q_reg[3]);
    Q_reg[5]  <= (S_L & p_in[5]) | (~S_L & Q_reg[4]);
    Q_reg[6]  <= (S_L & p_in[6]) | (~S_L & Q_reg[5]);
    Q_reg[7]  <= (S_L & p_in[7]) | (~S_L & Q_reg[6]);
    Q_reg[8]  <= (S_L & p_in[8]) | (~S_L & Q_reg[7]);
    Q_reg[9]  <= (S_L & p_in[9]) | (~S_L & Q_reg[8]);
    Q_reg[10] <= (S_L & p_in[10]) | (~S_L & Q_reg[9]);
    Q_reg[11] <= (S_L & p_in[11]) | (~S_L & Q_reg[10]);
    Q_reg[12] <= (S_L & p_in[12]) | (~S_L & Q_reg[11]);
    Q_reg[13] <= (S_L & p_in[13]) | (~S_L & Q_reg[12]);
    Q_reg[14] <= (S_L & p_in[14]) | (~S_L & Q_reg[13]);
    Q_reg[15] <= (S_L & p_in[15]) | (~S_L & Q_reg[14]);
    Q_reg[16] <= (S_L & p_in[16]) | (~S_L & Q_reg[15]);
  end
endmodule
```

reg_4_4b 模块：

```
module reg_4_4b (
    input wire clk,
    input wire[3:0]increase,
    output wire[15:0]num
);
    wire[31:0] clk_1ms;
    wire[3:0] c;
    reg[3:0] A=4'b0000,B=4'b0000,C=4'b0000,D=4'b0000;
    clkdiv d_0 (
        .clk(clk),
        .rst(1'b0),
        .clkdiv(clk_1ms)
    );
    pbdebounce d_1(
        .clk_1ms(clk_1ms[17]),
        .button(increase[0]),
        .pbreg(c[0])
    ),d_2(
        .clk_1ms(clk_1ms[17]),
        .button(increase[1]),
        .pbreg(c[1])
    ),d_3(
        .clk_1ms(clk_1ms[17]),
        .button(increase[2]),
```

```
        .pbreg(c[2])
    ),d_4(
        .clk_1ms(clk_1ms[17]),
        .button(increase[3]),
        .pbreg(c[3])
    );
    always @(posedge c[0]) begin
        A<=A+1'b1;
    end
    always @(posedge c[1]) begin
        B<=B+1'b1;
    end
    always @(posedge c[2]) begin
        C<=C+1'b1;
    end
    always @(posedge c[3]) begin
        D<=D+1'b1;
    end
    assign num={D,C,B,A};
endmodule
```

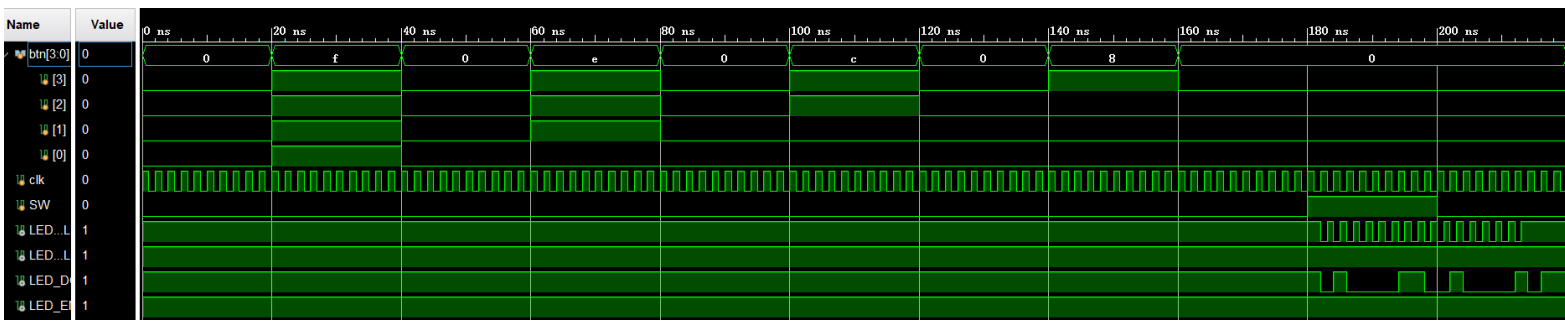
2.2 波形仿真

激励代码：

```
module top1_test2 ();
  reg [3:0] btn;
  reg clk=1'b0, SW;
//   reg[1:0] A=0,B=1,C=2,D=3;
//   reg[7:0] sum;
//   initial begin
    // sum<={A,B,C,D};
//   end
  wire LED_CLK, LED_CLR, LED_DO, LED_EN;
  always begin
     clk = ~clk;
     #1;
  end
  top1_test test2 (
    .btn(btn),
    .clk(clk),
    .SW(SW),
    .LED_CLK(LED_CLK),
```

```
    .LED_CLR(LED_CLR),
    .LED_DO(LED_DO),
    .LED_EN(LED_EN)
  );
  initial begin
    btn=4'b0000;
    SW=1'b0;
    #20;
    btn=4'b1111;
    #20;
    btn=4'b0000;
    #20;
    btn=4'b1110;
    #20;
    btn=4'b0000;
    #20;
    btn=4'b1100;
    #20;
    btn=4'b0000;
    #20;
    btn=4'b1000;
    #20;
    btn=4'b0000;
    #20;
    SW=1'b1;
    #20;
    SW=1'b0;
    #20;
    $finish;
  end
endmodule
```

仿真波形：

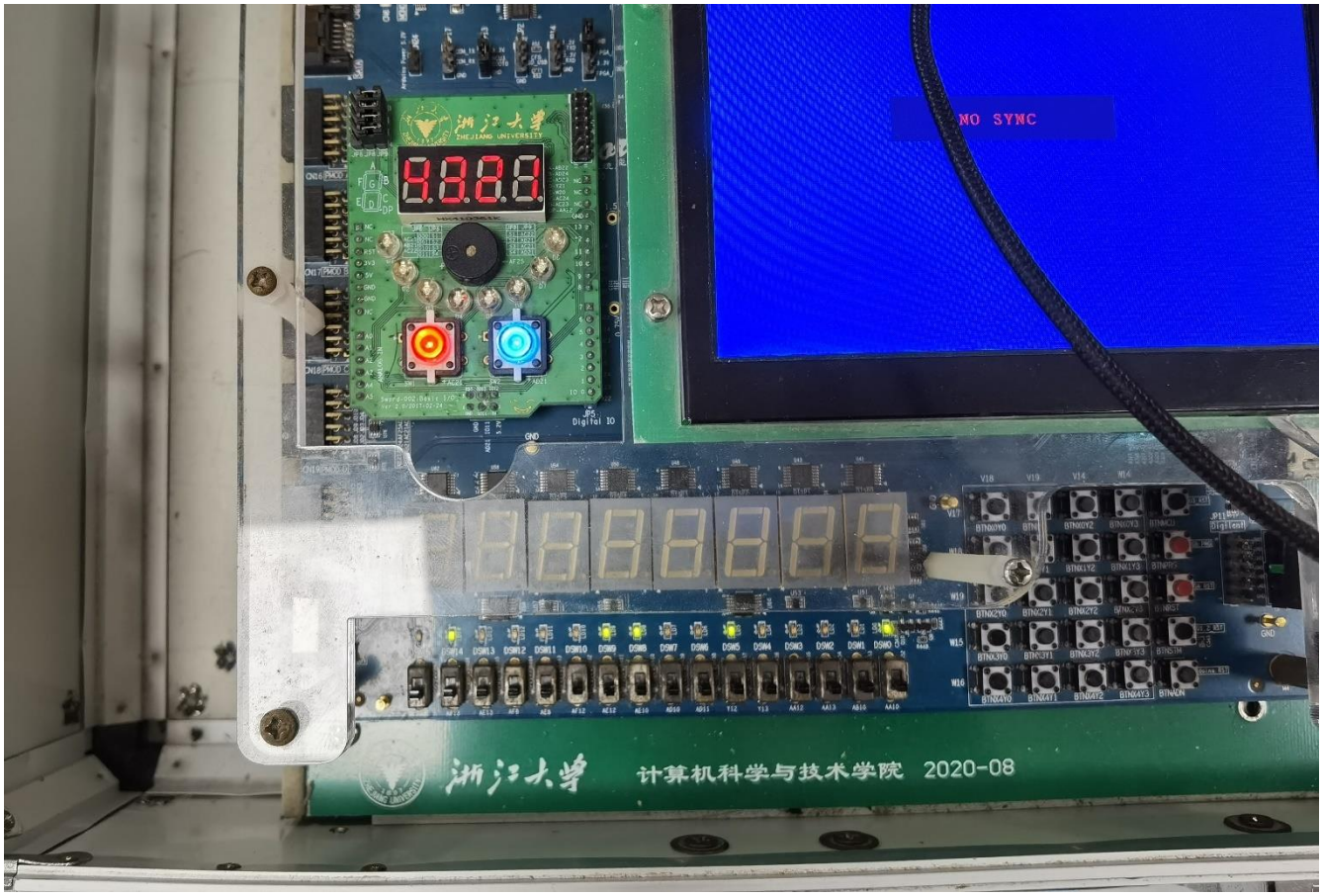


2.3 编写引脚约束文件

```
set_property IOSTANDARD LVCMOS33 [get_ports LED_CLK]
set_property IOSTANDARD LVCMOS33 [get_ports LED_CLR]
set_property IOSTANDARD LVCMOS33 [get_ports LED_DO]
set_property IOSTANDARD LVCMOS33 [get_ports LED_EN]
set_property PACKAGE_PIN N26 [get_ports LED_CLK]
set_property PACKAGE_PIN N24 [get_ports LED_CLR]
set_property PACKAGE_PIN M26 [get_ports LED_DO]
set_property PACKAGE_PIN P18 [get_ports LED_EN]
set_property IOSTANDARD LVCMOS18 [get_ports clk]
set_property PACKAGE_PIN AC18 [get_ports clk]
set_property PACKAGE_PIN AF10 [get_ports SW]
set_property IOSTANDARD LVCMOS15 [get_ports SW]
set_property IOSTANDARD LVCMOS18 [get_ports {btn[3]}]
set_property IOSTANDARD LVCMOS18 [get_ports {btn[2]}]
set_property IOSTANDARD LVCMOS18 [get_ports {btn[1]}]
set_property IOSTANDARD LVCMOS18 [get_ports {btn[0]}]
set_property PACKAGE_PIN W14 [get_ports {btn[0]}]
set_property PACKAGE_PIN V14 [get_ports {btn[1]}]
set_property PACKAGE_PIN V19 [get_ports {btn[2]}]
set_property PACKAGE_PIN V18 [get_ports {btn[3]}]
set_property CLOCK_DEDICATED_ROUTE FALSE [get_nets {btn[0]}]
set_property CLOCK_DEDICATED_ROUTE FALSE [get_nets {btn[1]}]
set_property CLOCK_DEDICATED_ROUTE FALSE [get_nets {btn[2]}]
set_property CLOCK_DEDICATED_ROUTE FALSE [get_nets {btn[3]}]
set_property PACKAGE_PIN W16 [get_ports BTNX4]
set_property IOSTANDARD LVCMOS18 [get_ports BTNX4]
set_property PACKAGE_PIN AB22 [get_ports {SEGMENT[0]}]
set_property PACKAGE_PIN AD24 [get_ports {SEGMENT[1]}]
set_property PACKAGE_PIN AD23 [get_ports {SEGMENT[2]}]
set_property PACKAGE_PIN Y21 [get_ports {SEGMENT[3]}]
set_property PACKAGE_PIN W20 [get_ports {SEGMENT[4]}]
set_property PACKAGE_PIN AC24 [get_ports {SEGMENT[5]}]
set_property PACKAGE_PIN AC23 [get_ports {SEGMENT[6]}]
set_property PACKAGE_PIN AA22 [get_ports {SEGMENT[7]}]
set_property IOSTANDARD LVCMOS33 [get_ports {SEGMENT[7]}]
set_property IOSTANDARD LVCMOS33 [get_ports {SEGMENT[6]}]
set_property IOSTANDARD LVCMOS33 [get_ports {SEGMENT[5]}]
set_property IOSTANDARD LVCMOS33 [get_ports {SEGMENT[4]}]
set_property IOSTANDARD LVCMOS33 [get_ports {SEGMENT[3]}]
set_property IOSTANDARD LVCMOS33 [get_ports {SEGMENT[2]}]
set_property IOSTANDARD LVCMOS33 [get_ports {SEGMENT[1]}]
set_property IOSTANDARD LVCMOS33 [get_ports {SEGMENT[0]}]
```

```
set_property IOSTANDARD LVCMOS33 [get_ports {AN[3]}]
set_property IOSTANDARD LVCMOS33 [get_ports {AN[2]}]
set_property IOSTANDARD LVCMOS33 [get_ports {AN[1]}]
set_property IOSTANDARD LVCMOS33 [get_ports {AN[0]}]
set_property PACKAGE_PIN AD21 [get_ports {AN[0]}]
set_property PACKAGE_PIN AC21 [get_ports {AN[1]}]
set_property PACKAGE_PIN AB21 [get_ports {AN[2]}]
set_property PACKAGE_PIN AC22 [get_ports {AN[3]}]
```

2.4 下载到实验板，进行实际验证。



3　任务 3：设计主板 8 位数码管驱动模块

3.1 设计计数器模块，并进行仿真

top2 模块：

```
`timescale 1ns / 1ps
```

```
module top2 (
    input wire [7:0] SW,
    input wire clk,
    output wire SEGCLK,
    output wire SEGCLR,
    output wire SEGDT,
    output wire SEGEN
);
  wire finish;
  assign SEGCLR = 1;
  assign SEGEN  = 1;
  wire [31:0] num;
  wire [63:0] code;
  reg_8_4b d_1 (
      .clk(clk),
      .increase(SW),
      .num(num)
  );
  MyMC14495 d_2 (
      .D0(num[0]),
      .D1(num[1]),
      .D2(num[2]),
      .D3(num[3]),
      .point(1'b0),
      .LE(1'b0),
      .a(code[0]),
      .b(code[1]),
      .c(code[2]),
      .d(code[3]),
      .e(code[4]),
      .f(code[5]),
      .g(code[6]),
      .p(code[7])
  );
  MyMC14495 d_3 (
      .D0(num[4]),
      .D1(num[5]),
      .D2(num[6]),
      .D3(num[7]),
      .point(1'b0),
      .LE(1'b0),
      .a(code[8]),
      .b(code[9]),
      .c(code[10]),
```

```
    .d(code[11]),
    .e(code[12]),
    .f(code[13]),
    .g(code[14]),
    .p(code[15])
);
MyMC14495 d_4 (
    .D0(num[8]),
    .D1(num[9]),
    .D2(num[10]),
    .D3(num[11]),
    .point(1'b0),
    .LE(1'b0),
    .a(code[16]),
    .b(code[17]),
    .c(code[18]),
    .d(code[19]),
    .e(code[20]),
    .f(code[21]),
    .g(code[22]),
    .p(code[23])
);
MyMC14495 d_5 (
    .D0(num[12]),
    .D1(num[13]),
    .D2(num[14]),
    .D3(num[15]),
    .point(1'b0),
    .LE(1'b0),
    .a(code[24]),
    .b(code[25]),
    .c(code[26]),
    .d(code[27]),
    .e(code[28]),
    .f(code[29]),
    .g(code[30]),
    .p(code[31])
);
MyMC14495 d_6 (
    .D0(num[16]),
    .D1(num[17]),
    .D2(num[18]),
    .D3(num[19]),
    .point(1'b0),
```

```
    .LE(1'b0),
    .a(code[32]),
    .b(code[33]),
    .c(code[34]),
    .d(code[35]),
    .e(code[36]),
    .f(code[37]),
    .g(code[38]),
    .p(code[39])
);
MyMC14495 d_7 (
    .D0(num[20]),
    .D1(num[21]),
    .D2(num[22]),
    .D3(num[23]),
    .point(1'b0),
    .LE(1'b0),
    .a(code[40]),
    .b(code[41]),
    .c(code[42]),
    .d(code[43]),
    .e(code[44]),
    .f(code[45]),
    .g(code[46]),
    .p(code[47])
);
MyMC14495 d_8 (
    .D0(num[24]),
    .D1(num[25]),
    .D2(num[26]),
    .D3(num[27]),
    .point(1'b0),
    .LE(1'b0),
    .a(code[48]),
    .b(code[49]),
    .c(code[50]),
    .d(code[51]),
    .e(code[52]),
    .f(code[53]),
    .g(code[54]),
    .p(code[55])
);
MyMC14495 d_9 (
    .D0(num[28]),
```

```
      .D1(num[29]),
      .D2(num[30]),
      .D3(num[31]),
      .point(1'b0),
      .LE(1'b0),
      .a(code[56]),
      .b(code[57]),
      .c(code[58]),
      .d(code[59]),
      .e(code[60]),
      .f(code[61]),
      .g(code[62]),
      .p(code[63])
  );
  SEG_DRV d_10 (
      .clk(clk),
      .num(code),
      .finish(finish),
      .ser_out(SEGDT)
  );
  assign SEGCLK = clk | finish;
endmodule
```

SLReg65b 模块：

```
`timescale 1ns / 1ps
module SLReg65b (
    input wire clk,
    input wire S_L,
    input wire s_in,
    input wire [64:0] p_in,
    output wire [64:0] Q
);
  reg [64:0] Q_reg;
  initial begin
    Q_reg =
65'b1111_1111_1111_1111_1111_1111_1111_1111_1111_1111_1111_1111_1111_1111_1111_1111_1
  end
  assign Q = Q_reg;
  always @(posedge clk) begin
    if (S_L) Q_reg <= p_in;
    else Q_reg<= {Q_reg[63:0], s_in};
  end
```

```
endmodule
```

SEG_DRV 模块：

```
module SEG_DRV (
    input wire clk,
    input wire [63:0] num,
    output wire finish,
    ser_out
);
  wire S_L;
  reg start;
  wire [64:0] O;
  reg [63:0] old_num;
  always @(posedge clk) begin
    old_num <= num;
  end
  initial begin
    old_num=64'b0;
    start   = 1'b0;
  end
  always @(posedge clk) begin
    if (num[63:0] != old_num[63:0]) start <= 1'b1;
    else start <= 1'b0;
  end
  SLReg65b d_1 (
      .clk(clk),
      .S_L(S_L),
      .s_in(1'b1),
      .p_in({num, 1'b0}),
      .Q(O)
  );
  assign finish = &O[63:0];
  assign S_L = finish & start;
  assign ser_out = O[64];
endmodule
```

reg_8_4b 模块：

```
module reg_8_4b (
    input wire clk,
    input wire[7:0]increase,
    output wire[31:0]num
);
    wire[31:0] clk_1ms;
    wire[7:0] c;
```

```
    reg[3:0]
A=4'b0000,B=4'b0000,C=4'b0000,D=4'b0000,E=4'b0000,F=4'b0000,G=4'b0000,H
=4'b0000;
    clkdiv d_0 (
        .clk(clk),
        .rst(1'b0),
        .clkdiv(clk_1ms)
    );
    pbdebounce d_1(
        .clk_1ms(clk_1ms[17]
        ),
        .button(increase[0]),
        .pbreg(c[0])
    ),d_2(
        .clk_1ms(clk_1ms[17]
        ),
        .button(increase[1]),
        .pbreg(c[1])
    ),d_3(
        .clk_1ms(clk_1ms[17]
        ),
        .button(increase[2]),
        .pbreg(c[2])
    ),d_4(
        .clk_1ms(clk_1ms[17]
        ),
        .button(increase[3]),
        .pbreg(c[3])
    ),d_5(
        .clk_1ms(clk_1ms[17]
        ),
        .button(increase[4]),
        .pbreg(c[4])
    ),d_6(
        .clk_1ms(clk_1ms[17]
        ),
        .button(increase[5]),
        .pbreg(c[5])
    ),d_7(
        .clk_1ms(clk_1ms[17]
        ),
        .button(increase[6]),
        .pbreg(c[6])
    ),d_8(
```

```
        .clk_1ms(clk_1ms[17]
        ),
        .button(increase[7]),
        .pbreg(c[7])
    );
    always @(posedge c[0]) begin
        A<=A+1'b1;
    end
    always @(posedge c[1]) begin
        B<=B+1'b1;
    end
    always @(posedge c[2]) begin
        C<=C+1'b1;
    end
    always @(posedge c[3]) begin
        D<=D+1'b1;
    end
    always @(posedge c[4]) begin
        E<=E+1'b1;
    end
    always @(posedge c[5]) begin
        F<=F+1'b1;
    end
    always @(posedge c[6]) begin
        G<=G+1'b1;
    end
    always @(posedge c[7]) begin
        H<=H+1'b1;
    end
    assign num={H,G,F,E,D,C,B,A};
endmodule
```

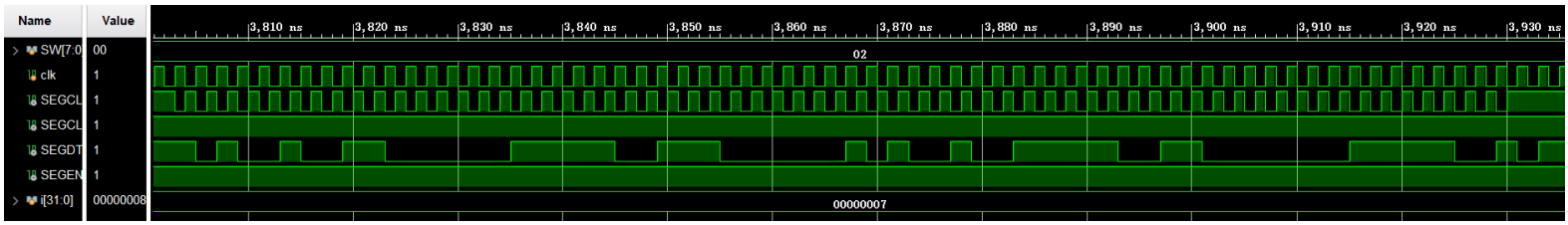
3.2 波形仿真：

激励代码：（显示为 20102181）

```
`timescale 0.01ps / 0.01ps
module top2_test ();
  reg [7:0] SW;
  reg clk;
  wire SEGCLK;
  wire SEGCLR;
  wire SEGDT;
```

```
  wire SEGEN;
  top2 test1 (
      .SW(SW),
      .clk(clk),
      .SEGCLK(SEGCLK),
      .SEGCLR(SEGCLR),
      .SEGDT(SEGDT),
      .SEGEN(SEGEN)
  );
always begin
    clk=0;
    #1;
    clk=1;
    #1;
end
integer  i=0;
initial begin
    SW=8'b0;
    #200000;
    SW[7]=1;
    SW[5]=1;
    SW[3]=1;
    SW[2]=1;
    SW[0]=1;
    #200000;
    SW=8'b0;
    #200000;
    SW[7]=1;
    SW[3]=1;
    #200000;
    SW=8'b0;
    #200000;
    for(i=0;i<8;i=i+1)
    begin
        SW[1]=1;
        #200000;
        SW[1]=0;
        #200000;
    end
    $finish;
end
endmodule
```

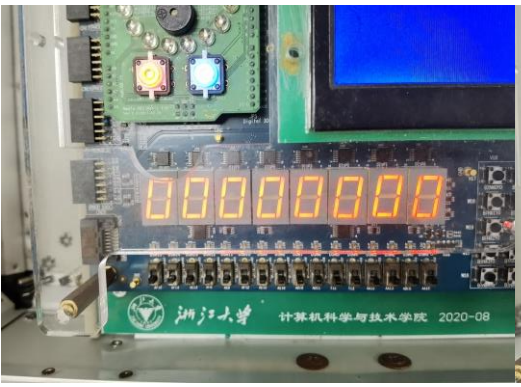
仿真波形：

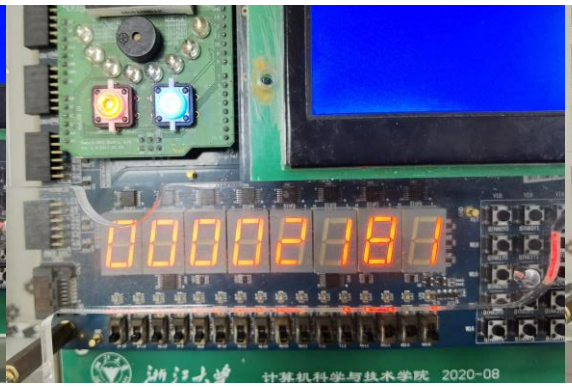


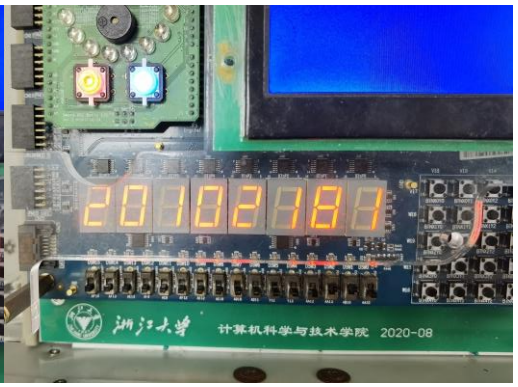
3.3 编写引脚约束文件

```
set_property IOSTANDARD LVCMOS18 [get_ports clk]
set_property PACKAGE_PIN AC18 [get_ports clk]
set_property PACKAGE_PIN M24 [get_ports SEGCLK]
set_property PACKAGE_PIN M20 [get_ports SEGCLR]
set_property PACKAGE_PIN L24 [get_ports SEGDT]
set_property PACKAGE_PIN R18 [get_ports SEGEN]
set_property IOSTANDARD LVCMOS15 [get_ports {SW[0]}]
set_property IOSTANDARD LVCMOS15 [get_ports {SW[1]}]
set_property IOSTANDARD LVCMOS15 [get_ports {SW[2]}]
set_property IOSTANDARD LVCMOS15 [get_ports {SW[3]}]
set_property IOSTANDARD LVCMOS15 [get_ports {SW[4]}]
set_property IOSTANDARD LVCMOS15 [get_ports {SW[5]}]
set_property IOSTANDARD LVCMOS15 [get_ports {SW[6]}]
set_property IOSTANDARD LVCMOS15 [get_ports {SW[7]}]
set_property PACKAGE_PIN AA10 [get_ports {SW[0]}]
set_property PACKAGE_PIN AB10 [get_ports {SW[1]}]
set_property PACKAGE_PIN AA13 [get_ports {SW[2]}]
set_property PACKAGE_PIN AA12 [get_ports {SW[3]}]
set_property PACKAGE_PIN Y13 [get_ports {SW[4]}]
set_property PACKAGE_PIN Y12 [get_ports {SW[5]}]
set_property PACKAGE_PIN AD11 [get_ports {SW[6]}]
set_property PACKAGE_PIN AD10 [get_ports {SW[7]}]

set_property IOSTANDARD LVCMOS33 [get_ports SEGCLK]
set_property IOSTANDARD LVCMOS33 [get_ports SEGCLR]
set_property IOSTANDARD LVCMOS33 [get_ports SEGDT]
set_property IOSTANDARD LVCMOS33 [get_ports SEGEN]
```

3.4 下载至实验板







四、 实验结果分析

仿真波形符合预期，实际移为显示视数清晰无波动。

果。

五、 实验心得体会

实验可以通过将时钟取反，为数据输入提供足够的准备时间。