浙江大学

本科实验报告

课程名称：	数字逻辑电路设计
姓　　名：	蔡佳伟
学　　院：	计算机科学与技术学院
专　　业：	软件工程
邮　　箱：	3220104519@zju.edu.cn
QQ　号：	3348536459
电　　话：	19550230334
指导教师：	洪奇军
报告日期：	2023 年 11 月 14 日

浙江大学实验报告

课程名称：　数字逻辑设计　实验类型：　综合

实验项目名称：　实验 8：加法器、加减法器与 ALU 基本原理与设计

学生姓名：　蔡佳伟　学号：　3220104519　同组学生姓名：

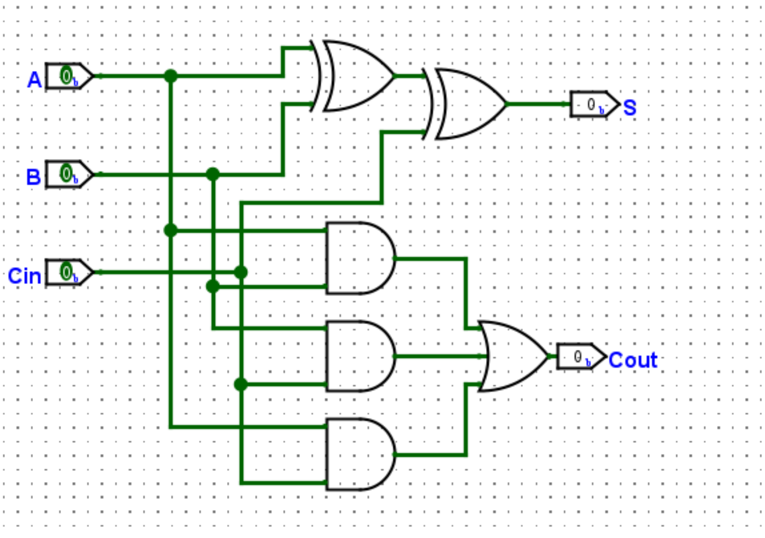
实验地点：　紫金港东四 509 室　实验日期：　2023　年　11　月　14　日

一、操作方法与实验步骤

（一）原理图设计实现一位全加器和四位加减法器

1、使用 Logisim 绘制一位全加器

绘制结果如下：

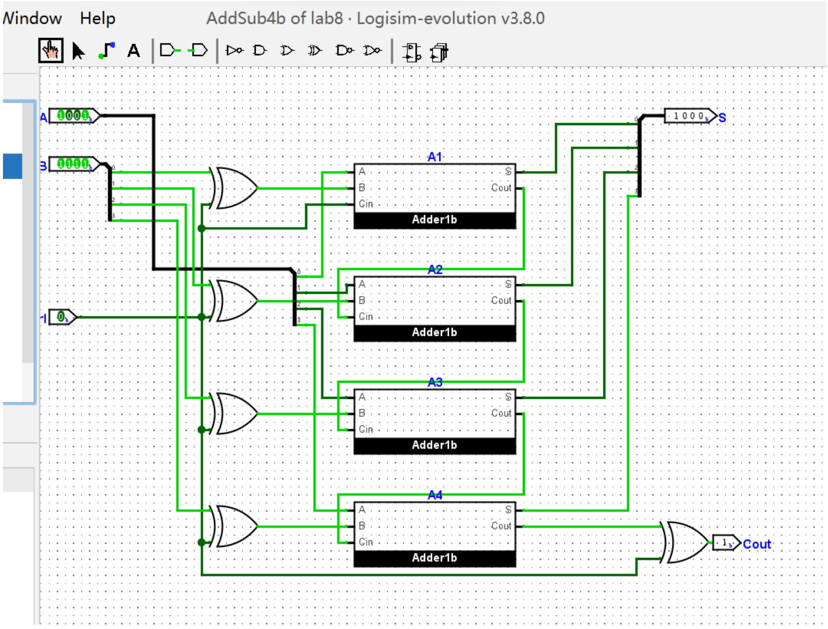


需要更改入口和出口名称，本次按照图片更改为 A、B、Cin、S、Cout

之后封装该模块为 Add1b

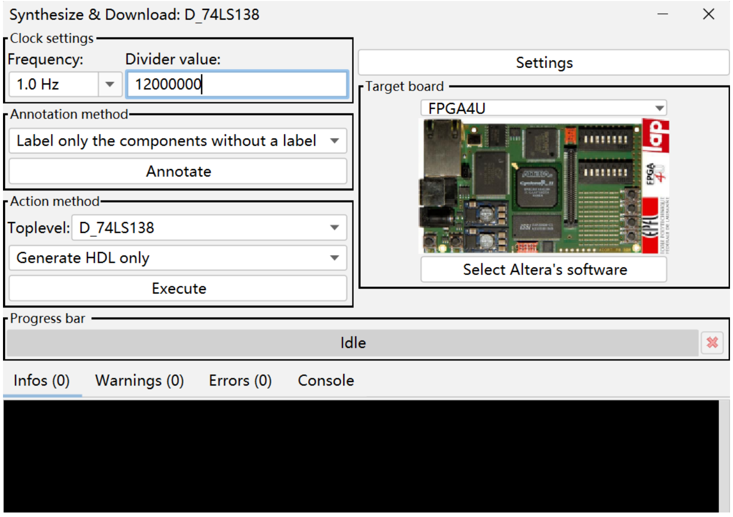
2、使用 Logisim 绘制四位加减法器

绘制结果如下：



注意需要使用上面绘制的 Adder1b 作为芯片，需要给芯片命名，否则无法导出为 Verilog 代码，同时需要给电路命名为 AddSub4b，否则影响之后操作。

之后导出为 Verilog 代码


Synthesize & Download: D_74LS138

Clock settings

Frequency: 1.0 Hz

Divider value: 12000000

Annotation method: Label only the components without a label

Action method

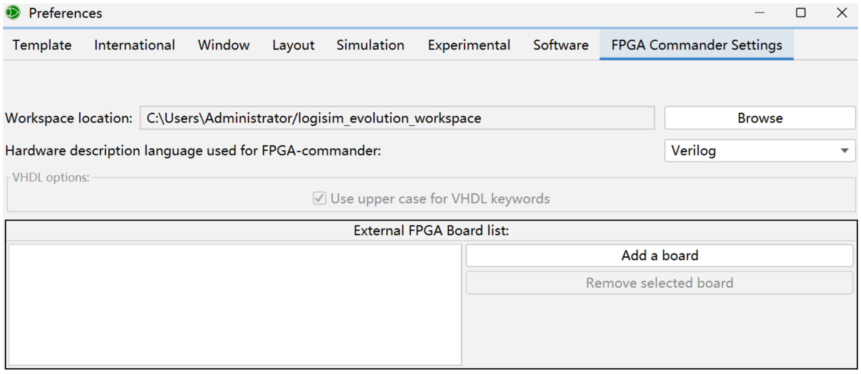
Toplevel: D_74LS138

Generate HDL only

Progress bar: Idle

Target board: FPGA4U

注意 Target board 选择 FPGA4U


Preferences

Workspace location: C:\Users\Administrator/logisim_evolution_workspace

Hardware description language used for FPGA-commander: Verilog

VHDL options:

☑ Use upper case for VHDL keywords

External FPGA Board list:

注意在 preferences 中把代码格式选为 Verilog

之后 Execute 生成代码，保存

Verilog 代码如下：

```
module AddSub4b( A,
            B,
            Cout,
            Ctrl,
            S );

input [3:0] A;
input [3:0] B;
input      Ctrl;

output      Cout;
output [3:0] S;

wire [3:0] s_logisimBus14;
wire [3:0] s_logisimBus23;
wire [3:0] s_logisimBus7;
wire      s_logisimNet0;
wire      s_logisimNet1;
wire      s_logisimNet10;
wire      s_logisimNet11;
wire      s_logisimNet12;
wire      s_logisimNet13;
wire      s_logisimNet15;
wire      s_logisimNet16;
wire      s_logisimNet17;
wire      s_logisimNet18;
wire      s_logisimNet19;
wire      s_logisimNet2;
wire      s_logisimNet20;
wire      s_logisimNet21;
wire      s_logisimNet22;
wire      s_logisimNet24;
wire      s_logisimNet3;
wire      s_logisimNet4;
wire      s_logisimNet5;
wire      s_logisimNet6;
wire      s_logisimNet8;
wire      s_logisimNet9;

assign s_logisimBus14[3:0] = A;
assign s_logisimBus23[3:0] = B;
assign s_logisimNet1      = Ctrl;

assign Cout = s_logisimNet24;
assign S    = s_logisimBus7[3:0];

XOR_GATE_ONEHOT #(.BubblesMask(2'b00))
   GATES_1 (.input1(s_logisimBus23[0]),
           .input2(s_logisimNet1),
           .result(s_logisimNet18));

XOR_GATE_ONEHOT #(.BubblesMask(2'b00))
   GATES_2 (.input1(s_logisimBus23[1]),
           .input2(s_logisimNet1),
           .result(s_logisimNet19));

XOR_GATE_ONEHOT #(.BubblesMask(2'b00))
   GATES_3 (.input1(s_logisimBus23[2]),
           .input2(s_logisimNet1),
           .result(s_logisimNet20));

XOR_GATE_ONEHOT #(.BubblesMask(2'b00))
```

```
      GATES_4 (.input1(s_logisimNet22),
              .input2(s_logisimNet1),
              .result(s_logisimNet24));

   XOR_GATE_ONEHOT #(.BubblesMask(2'b00))
      GATES_5 (.input1(s_logisimBus23[3]),
              .input2(s_logisimNet1),
              .result(s_logisimNet21));

   Adder1b   A1 (.A(s_logisimBus14[0]),
                .B(s_logisimNet18),
                .Cin(s_logisimNet1),
                .Cout(s_logisimNet9),
                .S(s_logisimBus7[0]));

   Adder1b   A2 (.A(s_logisimBus14[1]),
                .B(s_logisimNet19),
                .Cin(s_logisimNet9),
                .Cout(s_logisimNet10),
                .S(s_logisimBus7[1]));

   Adder1b   A3 (.A(s_logisimBus14[2]),
                .B(s_logisimNet20),
                .Cin(s_logisimNet10),
                .Cout(s_logisimNet11),
                .S(s_logisimBus7[2]));

   Adder1b   A4 (.A(s_logisimBus14[3]),
                .B(s_logisimNet21),
                .Cin(s_logisimNet11),
                .Cout(s_logisimNet22),
                .S(s_logisimBus7[3]));

endmodule
```

（二）ALU 的实现和简单应用

1、使用 Verilog 代码实现 ALU

先撰写 And2b4 与 Or2b4 模块，这里直接使用 Verilog 代码。

And2b4 模块代码如下，Or2b4 类似，这里由于篇幅原因省略。

```
module And2b4 ( A,
             B,
             res
            );


/**********************************************************************
***********
   **         The         inputs         are         defined         here
**

**********************************************************************
**********/
   input [3:0] A;
   input [3:0] B;

   output [3:0] res;

```

```
/*******************************************************************
**********
   **        The        wires        are        defined        here
**

*******************************************************************
*********/
   wire [3:0] s_logisimBus0;
   wire [3:0] s_logisimBus1;
   wire [3:0] s_logisimBus2;


/*******************************************************************
**********
   **     The     module     functionality     is     described     here
**

*******************************************************************
*********/


/*******************************************************************
**********
   **     Here     all     input     connections     are     defined
**

*******************************************************************
*********/
   assign s_logisimBus0[3:0] = A;
   assign s_logisimBus1[3:0] = B;


/*******************************************************************
**********
   **     Here     all     output     connections     are     defined
**

*******************************************************************
*********/
   assign res = s_logisimBus2[3:0];


/*******************************************************************
**********
   **     Here     all     normal     components     are     defined
**

*******************************************************************
*********/

   AND_GATE #(.BubblesMask(2'b00))
      GATES_1 (.input1(s_logisimBus0[0]),
               .input2(s_logisimBus1[0]),
               .result(s_logisimBus2[0]));

   AND_GATE #(.BubblesMask(2'b00))
      GATES_2 (.input1(s_logisimBus0[1]),
               .input2(s_logisimBus1[1]),
               .result(s_logisimBus2[1]));

   AND_GATE #(.BubblesMask(2'b00))
```

```
      GATES_3 (.input1(s_logisimBus0[2]),
              .input2(s_logisimBus1[2]),
              .result(s_logisimBus2[2]));

   AND_GATE #(.BubblesMask(2'b00))
      GATES_4 (.input1(s_logisimBus0[3]),
              .input2(s_logisimBus1[3]),
              .result(s_logisimBus2[3]));


 endmodule
```

2、撰写 ALU 模块 Verilog 代码如下：

```


module ALU( A,
            B,
            Cout,
            op,
            res );


/**********************************************************************
**********
   **         The          inputs          are          defined          here
**

***********************************************************************
*********/
   input [3:0] A;
   input [3:0] B;
   input [1:0] op;


/**********************************************************************
**********
   **         The          outputs          are          defined          here
**

***********************************************************************
*********/
   output      Cout;
   output [3:0] res;


/**********************************************************************
**********
   **         The          wires          are          defined          here
**

***********************************************************************
*********/
   wire [3:0] s_logisimBus0;
   wire [3:0] s_logisimBus10;
   wire [3:0] s_logisimBus2;
   wire [1:0] s_logisimBus3;
   wire [3:0] s_logisimBus5;
   wire [3:0] s_logisimBus7;
   wire [3:0] s_logisimBus8;
   wire       s_logisimNet1;
   wire       s_logisimNet4;
   wire       s_logisimNet6;
   wire      s_logisimNet9;


/**********************************************************************
**********
   **     The     module     functionality     is     described     here
**

***********************************************************************
*********/


/**********************************************************************
**********
   **     Here     all     input     connections     are     defined
**

***********************************************************************
*********/
   assign s_logisimBus0[3:0] = B;
   assign s_logisimBus2[3:0] = A;
   assign s_logisimBus3[1:0] = op;


/**********************************************************************
**********
   **     Here     all     output     connections     are     defined
**

***********************************************************************
*********/
   assign Cout = s_logisimNet6;
   assign res  = s_logisimBus8[3:0];


/**********************************************************************
**********
   **     Here     all     in-lined     components     are     defined
**

***********************************************************************
*********/

   // Ground
   assign  s_logisimNet4  =  1'b0;



/**********************************************************************
**********
   **       Here       all       sub-circuits       are       defined
**

***********************************************************************
*********/

   AddSub4b   AddSub4b1 (.A(s_logisimBus2[3:0]),
                      .B(s_logisimBus0[3:0]),
                      .Cout(s_logisimNet1),
                      .Ctrl(s_logisimBus3[0]),
                      .S(s_logisimBus5[3:0]));
   And2b4   And2b41 (.A(s_logisimBus2[3:0]),
                  .B(s_logisimBus0[3:0]),
                  .res(s_logisimBus10[3:0]));

   Or2b4   Or2b41 (.A(s_logisimBus2[3:0]),
                 .B(s_logisimBus0[3:0]),
                 .res(s_logisimBus7[3:0]));

   Mux4to1b4   Mux1 (.D0(s_logisimBus5[3:0]),
                  .D1(s_logisimBus5[3:0]),
                  .D2(s_logisimBus10[3:0]),
                  .D3(s_logisimBus7[3:0]),
                  .S(s_logisimBus3[1:0]),
                  .Y(s_logisimBus8[3:0]));

   Mux4to1   Mux2 (.D0(s_logisimNet1),
                 .D1(s_logisimNet1),
                 .D2(s_logisimNet4),
                 .D3(s_logisimNet4),
                 .S(s_logisimBus3[1:0]),
                 .Y(s_logisimNet6));

endmodule
```

需要注意各端口的命名不能重复，连线的正确性可以在之后通过仿真验证

3、使用 Vivado 对电路生成的 Verilog 代码进行仿真

（1）新建工程，此处命名为 project_7

注意：Project Name 界面不能有中文，Project Type 界面选择 RTL Project，Default Part 界面搜索并选择 xs7k160tffg676-2L，点击 Finish 完成工程创建。

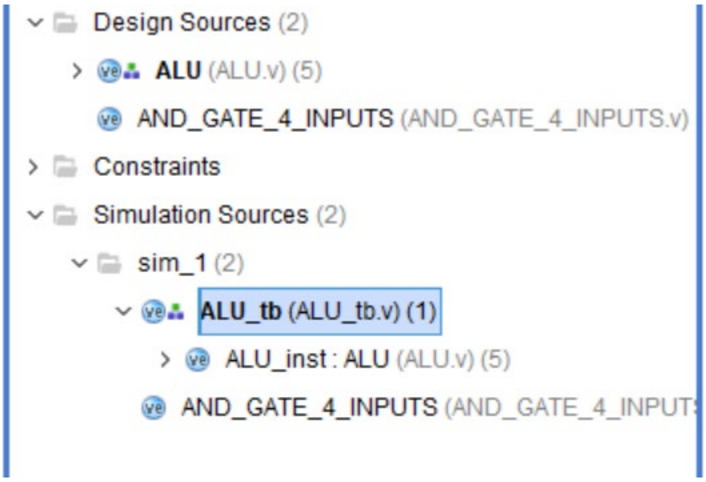
（2）添加综合文件

在 Project Manager 中选择 Add Sources，选择 Add or Create Design Source， 选择 verilog 子目录下的 circuit 和 gates 子目录的 Verilog 文件全部拷贝到工程中，随后点击 Finish 完成。 因为我们需要的是两个目录下的所有文件，因此可以通过 Add Directories 将两个目录下的全部文件添加进来。

（3）添加仿真文件并进行仿真

选择 Add or Create Simulation Sources 将仿真文件添加进入工程

导入后状态如下。



Design Sources (2)
ALU (ALU.v) (5)
AND_GATE_4_INPUTS (AND_GATE_4_INPUTS.v)
Constraints
Simulation Sources (2)
sim_1 (2)
ALU_tb (ALU_tb.v) (1)
ALU_inst : ALU (ALU.v) (5)
AND_GATE_4_INPUTS (AND_GATE_4_INPUT

之后进行仿真，仿真结果见“二、实验结果与分析”，仿真成功。

仿真代码如下：

```
`timescale 1ns / 1ps

module ALU_tb();

// Inputs
    reg [3:0] A;
    reg [3:0] B;
    reg [1:0] op;

// Outputs
    wire [3:0] res;
    wire Cout;

ALU ALU_inst(
    .A(A),
    .B(B),
    .op(op),
    .res(res),
    .Cout(Cout)
);

    initial begin
        op = 2'b00; A = 4'b1100; B = 4'b0011; #50;
        A = 4'b1111; B = 4'b1111; #50;
        op = 2'b01; A = 4'b1100; B = 4'b11; #50;
        A = 4'b0011; B = 4'b1100; #50;
        op = 2'b10; A = 4'b0011; B = 4'b0110; #50;
        A = 4'b1110; B = 4'b0011; #50;
        op = 2'b11; A = 4'b0011; B = 4'b0110; #50;
        A = 4'b1110; B = 4'b0011; #50;
    end

endmodule // ALU_tb
```

4、使用 Vivado 综合并上板验证

（1）撰写以下 clkdiv、pbdebounce、CreateNumber、ALU、DisplayNumber、Sseg_Dev 模块及 top 模块

各模块代码如下：

clkdiv.v:

```
module clkdiv(
    input               clk,
    input               rst, // Active-high
    output reg [31:0]   div_res
);

    always @(posedge clk) begin     // When postive edge of `clk` comes
        if(rst == 1'b1) begin
            div_res <= 32'b0;
        end else begin
            div_res <= div_res + 32'b1;  // Increase `div_res` by 1
        end
    end

endmodule
```

pbdebounce.v

```
module pbdebounce(
    input wire clk,
    input wire button,
    output reg pbreg
    );

    reg [7:0] pbshift;

    always@(posedge clk) begin
       pbshift = pbshift<<1;
       pbshift[0] = button;
       if (pbshift==8'b0)
          pbreg=0;
       if (pbshift==8'hFF)
          pbreg=1;
    end

endmodule
```

CreateNumber.v

```
/*
Module Name: CreateNumber
Description:
To change the value printed on Arduino using btns.
You will get a initial value printed as the para. INIT_HEXES defined.
After each pression on btn, a number will increase by 1.

This new module can handle i-1 when signal sw is 1
*/

module CreateNumber#(
parameter INIT_HEXES = 16'b1010_1011_1100_1101 // Init with "AbCd"
) (
input[3:0] btn,
input[3:0] sw,
output reg[15:0] num
);
wire[3:0] A, B, C, D;

initial num <= INIT_HEXES;

// D(the next num[3:0]) is always greater than current num[3:0] by 1
AddSub4b a0(.A(num[15:12]), .B(4'b0001), .Ctrl(sw[0]), .S(A));
AddSub4b a1(.A(num[11: 8]), .B(4'b0001), .Ctrl(sw[1]), .S(B));
AddSub4b a2(.A(num[ 7: 4]), .B(4'b0001), .Ctrl(sw[2]), .S(C));
AddSub4b a3(.A(num[ 3: 0]), .B(4'b0001), .Ctrl(sw[3]), .S(D));

// When pressing btn[0] num[3:0]++
always @(posedge btn[3]) num[15:12] <= A;
always @(posedge btn[2]) num[11: 8] <= B;
always @(posedge btn[1]) num[ 7: 4] <= C;
always @(posedge btn[0]) num[ 3: 0] <= D;

endmodule
```

DisplayNumber.v

```
module DisplayNumber(
    input       clk,
    input       rst,
    input [15:0] hexs,
    input [ 3:0] points,
    input [ 3:0] LEs,
    output[ 3:0] AN,
    output[ 7:0] SEGMENT
);
```

```
    wire [31:0]scan;
    wire [3:0]HEX;
    wire point;
    wire LE;

    clkdiv cd(
        .clk(clk),
        .rst(rst),
        .div_res(scan)
    );

    DisplaySync ds(
        .scan(scan[18:17]),
        .hexs(hexs),
        .points(points),
        .LEs(LEs),
        .HEX(HEX),
        .point(point),
        .LE(LE),
        .AN(AN)
    );

    MyMC14495 mc(
        .D0(HEX[0]),
        .D1(HEX[1]),
        .D2(HEX[2]),
        .D3(HEX[3]),
        .point(point),
        .LE(LE),
        .a(SEGMENT[0]),
        .b(SEGMENT[1]),
        .c(SEGMENT[2]),
        .d(SEGMENT[3]),
        .e(SEGMENT[4]),
        .f(SEGMENT[5]),
        .g(SEGMENT[6]),
        .p(SEGMENT[7])
    );
endmodule
```

Sseg_Dev.v

```
module SSeg_Dev(
    input clk,
    input flash,
    input [31:0] Hexs,
    input [7:0] LES,
    input [7:0] point,
    input rst,
    input Start,
    input seg_clk,
    input seg_clrn,
    input SEG_PEN,
    input seg_sout
);

   wire [63:0] SEGMENT;

   P2S  M2 (.clk(clk),
         .P_Data(SEGMENT[63:0]),
         .rst(rst),
         .Serial(Start),
         .EN(SEG_PEN),
         .sout(seg_sout),
         .s_clk(seg_clk),
         .s_clrn(seg_clrn));

   HexTo8SEG  SM1 (.flash(flash),
                .Hexs(Hexs[31:0]),
                .LES(LES[7:0]),
                .points(point[7:0]),
                .SEG_TXT(SEGMENT[63:0]));

endmodule
```

Top.v

```
module Top(
       input wire clk,
       input wire [1:0] BTN,
       input wire [1:0] SW1,
       input wire [1:0] SW2,
       input wire [11:0] SW,
       output wire [3:0] AN,
       output wire [7:0] SEGMENT,
       output wire BTNX4,
       output wire seg_clk,
       output wire seg_clrn,
       output wire seg_sout,
       output wire SEG_PEN
);
   wire [15:0] num;
   wire [1:0] btn_out;
   wire [3:0] res;
   wire Co;
   wire [31:0] clk_div;
   wire [15:0] disp_hexs;
   wire [15:0] disp_hexs_my;

   assign disp_hexs[15:12] = num[7:4];           // B
   assign disp_hexs[11:8]  = num[3:0];           // A
   assign disp_hexs[7:4]   = {3'b000, Co};
   assign disp_hexs[3:0]   = res[3:0];              // C

   /* Code here */
   assign disp_hexs_my = (4519); // Fill the last four digits of your
student id in ()

   assign BTNX4 = 1'b0;

   clkdiv m2(.clk(clk), .rst(0), .div_res(clk_div));
   pbdebounce
m0(.clk(clk_div[17]), .button(BTN[0]), .pbreg(btn_out[0]));
   pbdebounce
m1(.clk(clk_div[17]), .button(BTN[1]), .pbreg(btn_out[1]));

   CreateNumber m3(.btn(btn_out), .sw({2'b0, SW1}), .num(num)); //
Attachment

   // The ALU module you wrote
   ALU m5(    .A(num[3:0]),
                 .B(num[7:4]),                        // fill sth. in
()
                 .op(SW2[1:0]),                       // fill sth. in
()
                 .res(res[3:0]),                        // fill sth. in
()
                 .Cout(Co));

    // Module you design in Lab7
    DisplayNumber m6(     .clk(clk), .hexs(disp_hexs), .LEs(4'b1111),
// fill sth. in ()
                     .points(SW[3:0]),                     .rst(1'b0),
// fill sth. in ()
                     .AN(AN), .SEGMENT(SEGMENT));

    // Attachment
    SSeg_Dev  m7(.clk(clk),  .flash(clk_div[25]),  .Hexs({disp_hexs_my,
disp_hexs}), .LES(SW[11:4]),
                 .point({4'b0000,
SW[3:0]}), .rst(1'b0), .Start(clk_div[20]), .seg_clk(seg_clk),
                 .seg_clrn(seg_clrn), .SEG_PEN(SEG_PEN), .seg_sout(se
g_sout));

endmodule
```

注意：Top 模块的补充内容需要自己填写，填写结果如上所示。

整体来看，代码部分需要注意端口的命名和 module 的命名，不要出现大小写错误和命名错误，命名重复等问题

（2）导入到 Vivado：

需要注意：由于 DisplayNumber 是在 Lab7 中运用原理图完成，导入时需要导入 circuits 和 gates 两个文件夹，内含该代码所需的芯片，否则生成 bitstream 会报错。

（1）添加约束文件并修改

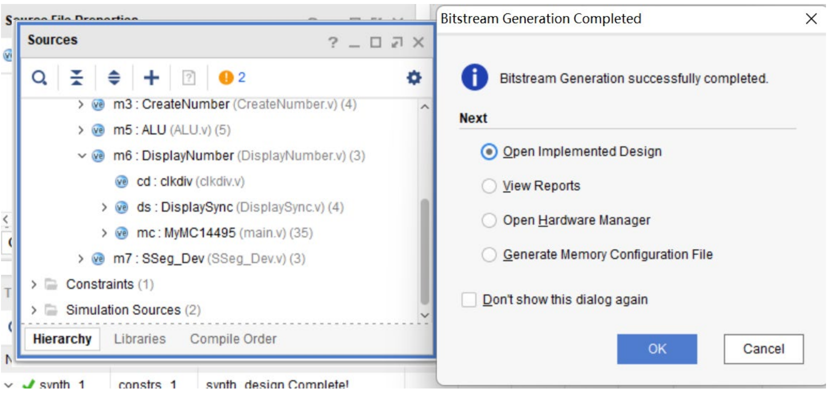
选择 Add or Create Constraints 将约束文件添加进入工程，并进行修改，设置引脚约束。修改后代码如下：

```
# Filename: constraints_lab8.xdc
## Constraints file for Lab8

# Main clock
set_property PACKAGE_PIN AC18 [get_ports clk]
set_property IOSTANDARD LVCMOS18 [get_ports clk]

create_clock -period 10.000 -name clk [get_ports "clk"]

# Switches as inputs
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 PACKAGE_PIN AE10 [get_ports {SW[8]}]
set_property PACKAGE_PIN AE12 [get_ports {SW[9]}]
set_property PACKAGE_PIN AF12 [get_ports {SW[10]}]
set_property PACKAGE_PIN AE8 [get_ports {SW[11]}]
set_property PACKAGE_PIN AF8 [get_ports {SW1[0]}]
set_property PACKAGE_PIN AE13 [get_ports {SW1[1]}]
set_property PACKAGE_PIN AF13 [get_ports {SW2[0]}]
set_property PACKAGE_PIN AF10 [get_ports {SW2[1]}]
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 IOSTANDARD LVCMOS15 [get_ports {SW[8]}]
set_property IOSTANDARD LVCMOS15 [get_ports {SW[9]}]
set_property IOSTANDARD LVCMOS15 [get_ports {SW[10]}]
set_property IOSTANDARD LVCMOS15 [get_ports {SW[11]}]
set_property IOSTANDARD LVCMOS15 [get_ports {SW1[0]}]
set_property IOSTANDARD LVCMOS15 [get_ports {SW1[1]}]
set_property IOSTANDARD LVCMOS15 [get_ports {SW2[0]}]
set_property IOSTANDARD LVCMOS15 [get_ports {SW2[1]}]

# Key as inputs
set_property PACKAGE_PIN W16 [get_ports BTNX4]
set_property IOSTANDARD LVCMOS18 [get_ports BTNX4]
set_property PACKAGE_PIN V14 [get_ports {BTN[1]}]
set_property IOSTANDARD LVCMOS18 [get_ports {BTN[1]}]
set_property PACKAGE_PIN W14 [get_ports {BTN[0]}]
set_property IOSTANDARD LVCMOS18 [get_ports {BTN[0]}]

set_property CLOCK_DEDICATED_ROUTE FALSE [get_nets BTN*]

# Arduino-Segment & AN
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]}]
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 {AN[0]}]
set_property IOSTANDARD LVCMOS33 [get_ports {AN[1]}]
set_property IOSTANDARD LVCMOS33 [get_ports {AN[2]}]
set_property IOSTANDARD LVCMOS33 [get_ports {AN[3]}]
set_property IOSTANDARD LVCMOS33 [get_ports {SEGMENT[0]}]
set_property IOSTANDARD LVCMOS33 [get_ports {SEGMENT[1]}]
set_property IOSTANDARD LVCMOS33 [get_ports {SEGMENT[2]}]
set_property IOSTANDARD LVCMOS33 [get_ports {SEGMENT[3]}]
set_property IOSTANDARD LVCMOS33 [get_ports {SEGMENT[4]}]
set_property IOSTANDARD LVCMOS33 [get_ports {SEGMENT[5]}]
set_property IOSTANDARD LVCMOS33 [get_ports {SEGMENT[6]}]
set_property IOSTANDARD LVCMOS33 [get_ports {SEGMENT[7]}]

set_property PACKAGE_PIN M24 [get_ports {seg_clk}]
set_property IOSTANDARD LVCMOS33 [get_ports {seg_clk}]
set_property PACKAGE_PIN M20 [get_ports {seg_clrn}]
set_property IOSTANDARD LVCMOS33 [get_ports {seg_clrn}]
set_property PACKAGE_PIN L24 [get_ports {seg_sout}]
set_property IOSTANDARD LVCMOS33 [get_ports {seg_sout}]
set_property PACKAGE_PIN R18 [get_ports {SEG_PEN}]
set_property IOSTANDARD LVCMOS33 [get_ports {SEG_PEN}]

# # Main clock
# set_property PACKAGE_PIN AC18 [get_ports clk_p]
# set_property PACKAGE_PIN AD18 [get_ports clk_n]
# set_property IOSTANDARD LVCMOS18 [get_ports clk_p]
# set_property IOSTANDARD LVCMOS18 [get_ports clk_n]

# # create_clock -period 10.000 -name clk [get_ports "clk_p"]

# # FPGA RST
# set_property PACKAGE_PIN W13 [get_ports RSTN]
# set_property IOSTANDARD LVCMOS18 [get_ports RSTN]

# # 7SEG
# set_property PACKAGE_PIN M24 [get_ports seg_clk]
# set_property PACKAGE_PIN L24 [get_ports set_sout]
# set_property PACKAGE_PIN R18 [get_ports seg_pen]
# set_property PACKAGE_PIN M20 [get_ports seg_clrn]
# set_property IOSTANDARD LVCMOS33 [get_ports seg_clk]
# set_property IOSTANDARD LVCMOS33 [get_ports set_sout]
# set_property IOSTANDARD LVCMOS33 [get_ports seg_pen]
# set_property IOSTANDARD LVCMOS33 [get_ports seg_clrn]

# # Audio out
# set_property PACKAGE_PIN P26 [get_ports AUD_PWM]
# set_property PACKAGE_PIN M25 [get_ports AUD_SD]
# set_property IOSTANDARD LVCMOS33 [get_ports AUD_PWM]
# set_property IOSTANDARD LVCMOS33 [get_ports AUD_SD]

# # Key Array
# set_property PACKAGE_PIN V17 [get_ports BTN_X0]
# set_property IOSTANDARD LVCMOS18 [get_ports BTN_X0]
# set_property PACKAGE_PIN W18 [get_ports BTN_X1]
# set_property IOSTANDARD LVCMOS18 [get_ports BTN_X1]
# set_property PACKAGE_PIN W19 [get_ports BTN_X2]
# set_property IOSTANDARD LVCMOS18 [get_ports BTN_X2]
# set_property PACKAGE_PIN W15 [get_ports BTN_X3]
# set_property IOSTANDARD LVCMOS18 [get_ports BTN_X3]
# set_property PACKAGE_PIN W16 [get_ports BTN_X4]
# set_property IOSTANDARD LVCMOS18 [get_ports BTN_X4]
# set_property PACKAGE_PIN V18 [get_ports BTN_Y0]
# set_property IOSTANDARD LVCMOS18 [get_ports BTN_Y0]
# set_property PACKAGE_PIN V19 [get_ports BTN_Y1]
# set_property IOSTANDARD LVCMOS18 [get_ports BTN_Y1]
# set_property PACKAGE_PIN V14 [get_ports BTN_Y2]
# set_property IOSTANDARD LVCMOS18 [get_ports BTN_Y2]
# set_property PACKAGE_PIN W14 [get_ports BTN_Y3]
# set_property IOSTANDARD LVCMOS18 [get_ports BTN_Y3]

# # Arduino
# set_property PACKAGE_PIN AF25 [get_ports ard_rst]
# set_property IOSTANDARD LVCMOS33 [get_ports ard_rst]
# set_property PACKAGE_PIN AF24 [get_ports {ard_led[0]}]
# set_property PACKAGE_PIN AF21 [get_ports {ard_led[1]}]
# set_property PACKAGE_PIN Y22 [get_ports {ard_led[2]}]
# set_property PACKAGE_PIN Y23 [get_ports {ard_led[3]}]
# set_property PACKAGE_PIN AA23 [get_ports {ard_led[4]}]
# set_property PACKAGE_PIN Y25 [get_ports {ard_led[5]}]
# set_property PACKAGE_PIN AB26 [get_ports {ard_led[6]}]
# set_property PACKAGE_PIN W23 [get_ports {ard_led[7]}]
# set_property IOSTANDARD LVCMOS33 [get_ports {ard_led[0]}]
# set_property IOSTANDARD LVCMOS33 [get_ports {ard_led[1]}]
# set_property IOSTANDARD LVCMOS33 [get_ports {ard_led[2]}]
# set_property IOSTANDARD LVCMOS33 [get_ports {ard_led[3]}]
# set_property IOSTANDARD LVCMOS33 [get_ports {ard_led[4]}]
# set_property IOSTANDARD LVCMOS33 [get_ports {ard_led[5]}]
# set_property IOSTANDARD LVCMOS33 [get_ports {ard_led[6]}]
# set_property IOSTANDARD LVCMOS33 [get_ports {ard_led[7]}]
# set_property PACKAGE_PIN AD21 [get_ports {ard_an[0]}]
# set_property PACKAGE_PIN AC21 [get_ports {ard_an[1]}]
# set_property PACKAGE_PIN AB21 [get_ports {ard_an[2]}]
# set_property PACKAGE_PIN AC22 [get_ports {ard_an[3]}]
# set_property PACKAGE_PIN AB22 [get_ports {ard_seg[0]}]
# set_property PACKAGE_PIN AD24 [get_ports {ard_seg[1]}]
# set_property PACKAGE_PIN AD23 [get_ports {ard_seg[2]}]
# set_property PACKAGE_PIN Y21 [get_ports {ard_seg[3]}]
# set_property PACKAGE_PIN W20 [get_ports {ard_seg[4]}]
# set_property PACKAGE_PIN AC24 [get_ports {ard_seg[5]}]
# set_property PACKAGE_PIN AC23 [get_ports {ard_seg[6]}]
# set_property PACKAGE_PIN AA22 [get_ports {ard_seg[7]}]
# # set_property IOSTANDARD LVCMOS33 [get_ports {ard_dio[13]}]
# # set_property IOSTANDARD LVCMOS33 [get_ports {ard_dio[12]}]
# set_property IOSTANDARD LVCMOS33 [get_ports {ard_an[0]}]
# set_property IOSTANDARD LVCMOS33 [get_ports {ard_an[1]}]
# set_property IOSTANDARD LVCMOS33 [get_ports {ard_an[2]}]
# set_property IOSTANDARD LVCMOS33 [get_ports {ard_an[3]}]
# set_property IOSTANDARD LVCMOS33 [get_ports {ard_seg[0]}]
# set_property IOSTANDARD LVCMOS33 [get_ports {ard_seg[1]}]
# set_property IOSTANDARD LVCMOS33 [get_ports {ard_seg[2]}]
# set_property IOSTANDARD LVCMOS33 [get_ports {ard_seg[3]}]
# set_property IOSTANDARD LVCMOS33 [get_ports {ard_seg[4]}]
# set_property IOSTANDARD LVCMOS33 [get_ports {ard_seg[5]}]
# set_property IOSTANDARD LVCMOS33 [get_ports {ard_seg[6]}]
# set_property IOSTANDARD LVCMOS33 [get_ports {ard_seg[7]}]

# #16leds
# set_property PACKAGE_PIN N26 [get_ports LEDCLK]
# set_property PACKAGE_PIN N24 [get_ports LEDCLR]
# set_property PACKAGE_PIN M26 [get_ports LEDDT]
# set_property IOSTANDARD LVCMOS33 [get_ports LEDCLK]
# set_property IOSTANDARD LVCMOS33 [get_ports LEDCLR]
# set_property IOSTANDARD LVCMOS33 [get_ports LEDDT]

# #16dips
# set_property PACKAGE_PIN AA10 [get_ports {switch[0]}]
# set_property PACKAGE_PIN AB10 [get_ports {switch[1]}]
# set_property PACKAGE_PIN AA13 [get_ports {switch[2]}]
# set_property PACKAGE_PIN AA12 [get_ports {switch[3]}]
# set_property PACKAGE_PIN Y13 [get_ports {switch[4]}]
# set_property PACKAGE_PIN Y12 [get_ports {switch[5]}]
# set_property PACKAGE_PIN AD11 [get_ports {switch[6]}]
# set_property PACKAGE_PIN AD10 [get_ports {switch[7]}]
# set_property PACKAGE_PIN AE10 [get_ports {switch[8]}]
# set_property PACKAGE_PIN AE12 [get_ports {switch[9]}]
# set_property PACKAGE_PIN AF12 [get_ports {switch[10]}]
# set_property PACKAGE_PIN AE8 [get_ports {switch[11]}]
# set_property PACKAGE_PIN AF8 [get_ports {switch[12]}]
# set_property PACKAGE_PIN AE13 [get_ports {switch[13]}]
# set_property PACKAGE_PIN AF13 [get_ports {switch[14]}]
# set_property PACKAGE_PIN AF10 [get_ports {switch[15]}]
# set_property IOSTANDARD LVCMOS15 [get_ports {switch[0]}]
# set_property IOSTANDARD LVCMOS15 [get_ports {switch[1]}]
# set_property IOSTANDARD LVCMOS15 [get_ports {switch[2]}]
# set_property IOSTANDARD LVCMOS15 [get_ports {switch[3]}]
# set_property IOSTANDARD LVCMOS15 [get_ports {switch[4]}]
# set_property IOSTANDARD LVCMOS15 [get_ports {switch[5]}]
```

```
        # set_property IOSTANDARD LVCMOS15 [get_ports {switch[6]}]
        # set_property IOSTANDARD LVCMOS15 [get_ports {switch[7]}]
        # set_property IOSTANDARD LVCMOS15 [get_ports {switch[8]}]
        # set_property IOSTANDARD LVCMOS15 [get_ports {switch[9]}]
        # set_property IOSTANDARD LVCMOS15 [get_ports {switch[10]}]
        # set_property IOSTANDARD LVCMOS15 [get_ports {switch[11]}]
        # set_property IOSTANDARD LVCMOS15 [get_ports {switch[12]}]
        # set_property IOSTANDARD LVCMOS15 [get_ports {switch[13]}]
        # set_property IOSTANDARD LVCMOS15 [get_ports {switch[14]}]
        # set_property IOSTANDARD LVCMOS15 [get_ports {switch[15]}]

        # # VGA
        # set_property PACKAGE_PIN N21 [get_ports {vga_red[0]}]
        # set_property PACKAGE_PIN N22 [get_ports {vga_red[1]}]
        # set_property PACKAGE_PIN R21 [get_ports {vga_red[2]}]
        # set_property PACKAGE_PIN P21 [get_ports {vga_red[3]}]
        # set_property IOSTANDARD LVCMOS33 [get_ports {vga_red[0]}]
        # set_property IOSTANDARD LVCMOS33 [get_ports {vga_red[1]}]
        # set_property IOSTANDARD LVCMOS33 [get_ports {vga_red[2]}]
        # set_property IOSTANDARD LVCMOS33 [get_ports {vga_red[3]}]
        # set_property PACKAGE_PIN R22 [get_ports {vga_green[0]}]
        # set_property PACKAGE_PIN R23 [get_ports {vga_green[1]}]
        # set_property PACKAGE_PIN T24 [get_ports {vga_green[2]}]
        # set_property PACKAGE_PIN T25 [get_ports {vga_green[3]}]
        # set_property IOSTANDARD LVCMOS33 [get_ports {vga_green[0]}]
        # set_property IOSTANDARD LVCMOS33 [get_ports {vga_green[1]}]
        # set_property IOSTANDARD LVCMOS33 [get_ports {vga_green[2]}]
        # set_property IOSTANDARD LVCMOS33 [get_ports {vga_green[3]}]
        # set_property PACKAGE_PIN T20 [get_ports {vga_blue[0]}]
        # set_property PACKAGE_PIN R20 [get_ports {vga_blue[1]}]
        # set_property PACKAGE_PIN T22 [get_ports {vga_blue[2]}]
        # set_property PACKAGE_PIN T23 [get_ports {vga_blue[3]}]
        # set_property IOSTANDARD LVCMOS33 [get_ports {vga_blue[0]}]
        # set_property IOSTANDARD LVCMOS33 [get_ports {vga_blue[1]}]
        # set_property IOSTANDARD LVCMOS33 [get_ports {vga_blue[2]}]
        # set_property IOSTANDARD LVCMOS33 [get_ports {vga_blue[3]}]
        # set_property PACKAGE_PIN M22 [get_ports vga_hs]
        # set_property PACKAGE_PIN [get_ports vga_vs]
        # set_property IOSTANDARD LVCMOS33 [get_ports vga_hs]
        # set_property IOSTANDARD LVCMOS33 [get_ports vga_vs]
```

（2）生成 bitstream 并烧录

配置完成后，得到 bitstream。



Bitstream Generation Completed

Bitstream Generation successfully completed.

Next

● Open Implemented Design

○ View Reports

○ Open Hardware Manager

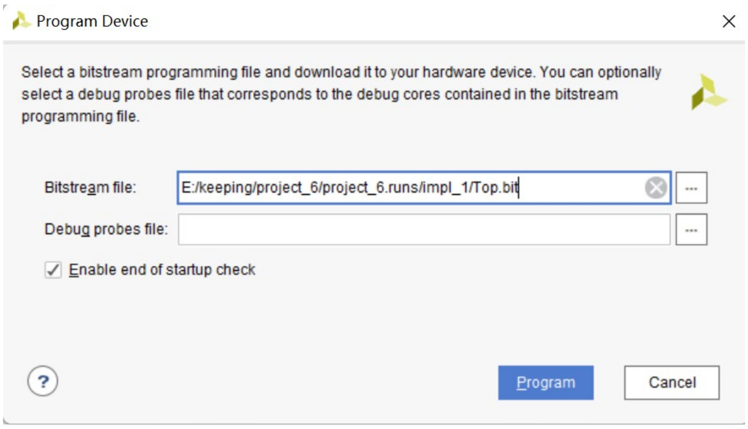
○ Generate Memory Configuration File

□ Don't show this dialog again

Sources
m3 : CreateNumber (CreateNumber.v) (4)
m5 : ALU (ALU.v) (5)
m6 : DisplayNumber (DisplayNumber.v) (3)
cd : clkdiv (clkdiv.v)
ds : DisplaySync (DisplaySync.v) (4)
mc : MyMC14495 (main.v) (35)
m7 : SSeg_Dev (SSeg_Dev.v) (3)
Constraints (1)
Simulation Sources (2)
Hierarchy
Libraries
Compile Order
OK
Cancel
synth_1
constrs_1
synth_design Complete!

之后将下载器连接到电脑上。点击 PROGRAM AND DEBUG > Open Hardware Manager > Open Target > Auto Connect 进行识别和连接，成功连接后，点击 Program Device 选择 xc7k160t 设备，在下载程序界面选择我们刚刚生成的比特流文件，将其下

载到板上。之后在板上实现相关操作。



Program Device

Select a bitstream programming file and download it to your hardware device. You can optionally select a debug probes file that corresponds to the debug cores contained in the bitstream programming file.

Bitstream file: E:/keeping/project_6/project_6.runs/impl_1/Top.bit

Debug probes file:

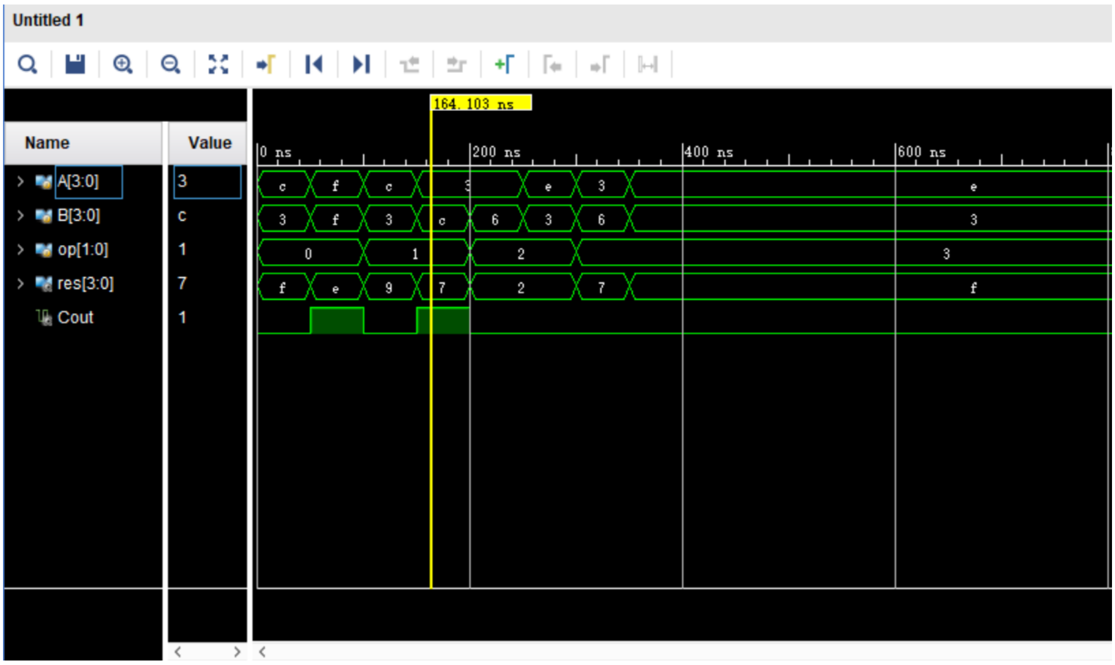
☑ Enable end of startup check

注意此处需要正确标出路径才可以正确上板。

二、实验结果与分析

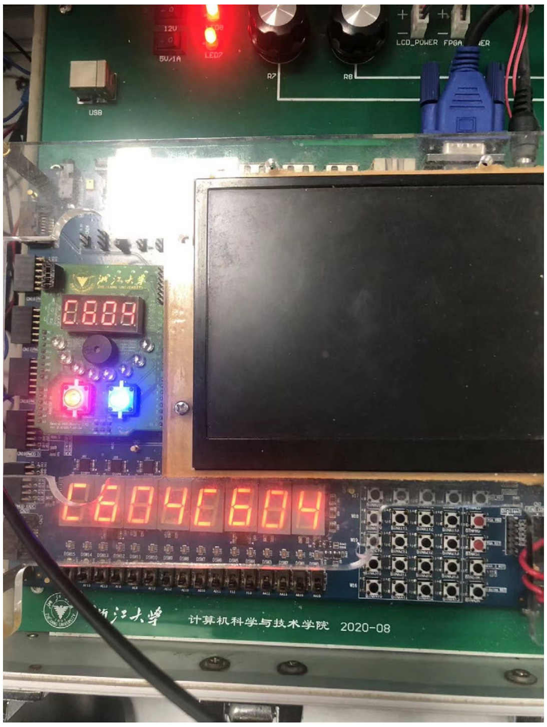
（一）ALU 加法器的设计

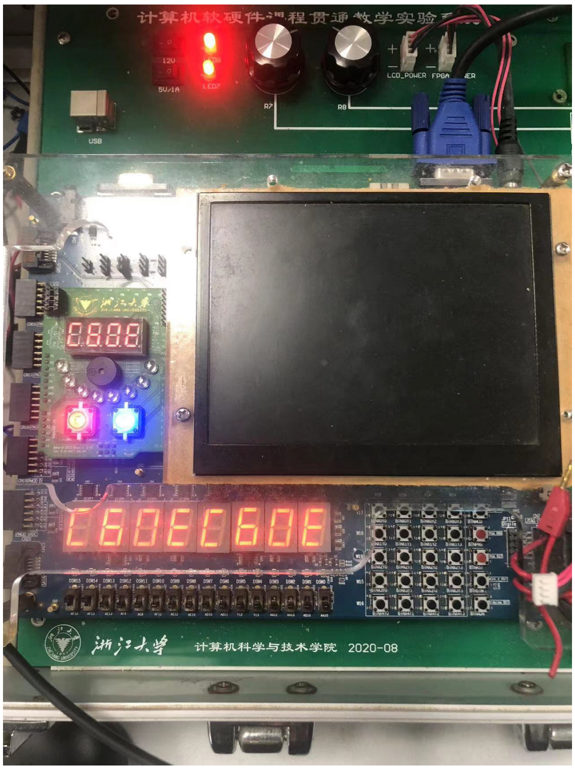
1、ALU 的仿真结果

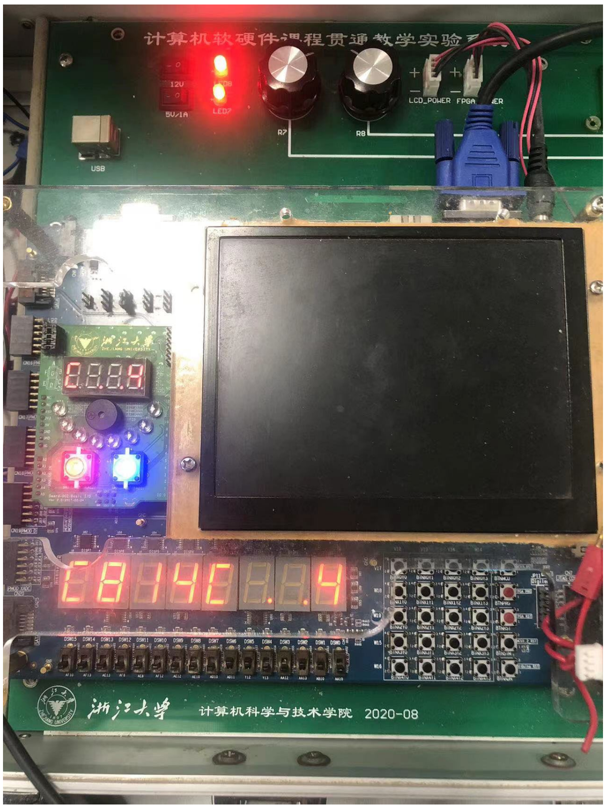


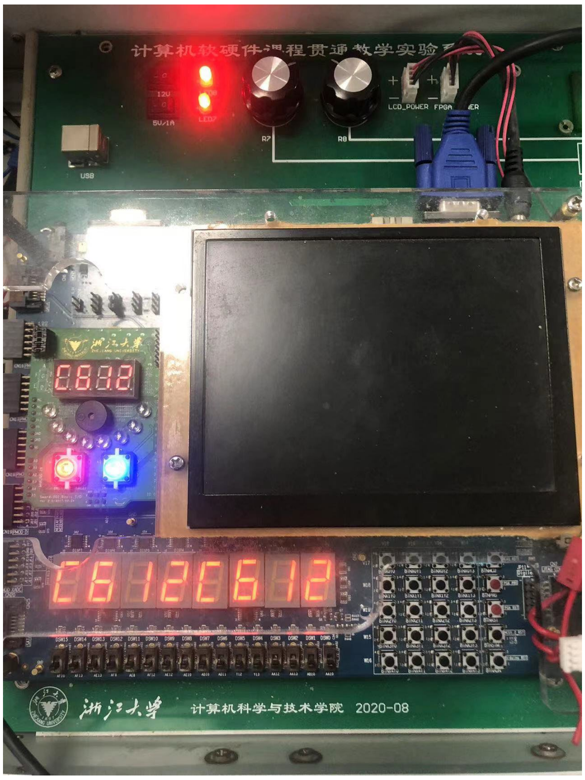
如图，A、B、op 对 res 和 Cout 的状态控制正确，说明成功实现 ALU 功能。

（二）ALU 简单应用模块实验结果









可以看到，显示数字功能正确，同时也可以实现加法、减法等功能。说明各部分代码撰写正确，成功实现需求的功能。

三、讨论、心得

在本次实验中，我增强了自己的 Verilog 代码撰写能力。因为之后我们需要学习很多硬件代码，Verilog 代码的撰写无疑是十分重要的。这次实验大部分代码需要自己修改和撰写，这提升了我的能力。同时，我对加法器的结构和构造了解更加清晰，这对我的数逻学习很有帮助。

四、个人生活照片

浙江大学

本科实验报告

课程名称：	数字逻辑电路设计
姓名：	蔡佳伟
学院：	计算机科学与技术学院
专业：	计算机科学与技术
邮箱：	3220104519@zju.edu.cn
QQ 号：	3348536459
电话：	19550230334
指导教师：	洪奇军
报告日期：	2023 年 10 月 30 日

浙江大学实验报告

课程名称：数字逻辑设计 实验类型：综合

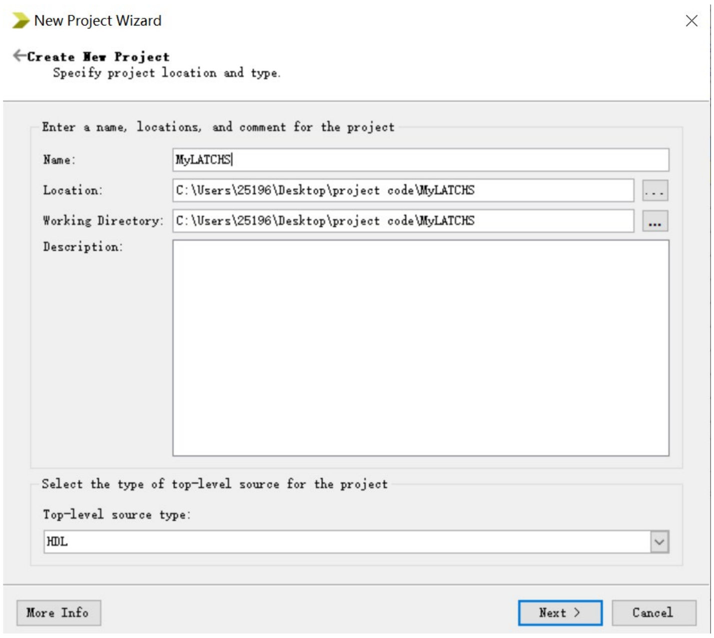
实验项目名称：实验 9：锁存器与触发器基本原理

学生姓名：蔡佳伟 学号：3220104519 同组学生姓名：无

实验地点：紫金港东四 509 室 实验日期：2023 年 10 月 24 日

一、操作方法与实验步骤

1. 新建工程 MyLATCHS



New Project Wizard

Create New Project
Specify project location and type.

Enter a name, locations, and comment for the project

Name: MyLATCHS

Location: C:\Users\25196\Desktop\project code\MyLATCHS

Working Directory: C:\Users\25196\Desktop\project code\MyLATCHS

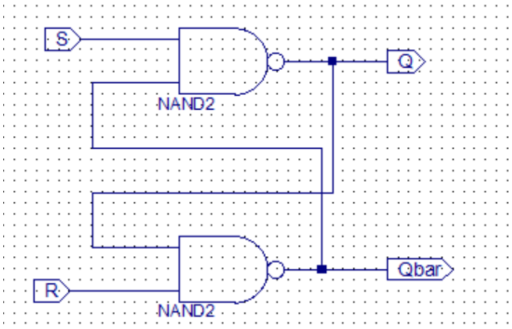
Description:

Select the type of top-level source for the project

Top-level source type: HDL

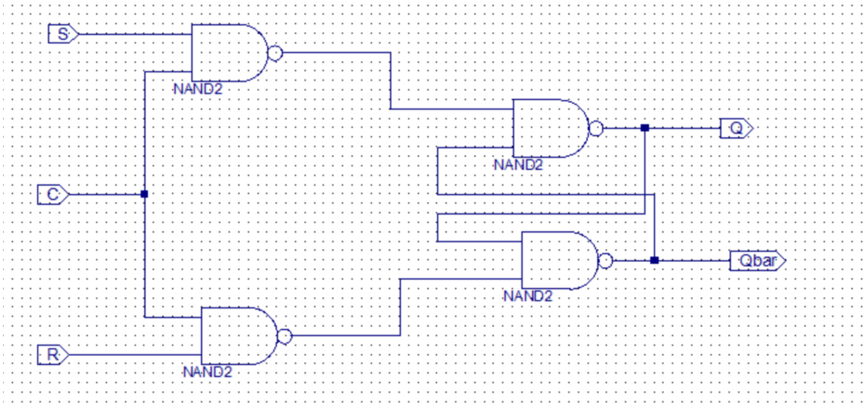
2. 新建源文件 SR_LATCH.sch

用原理图方式设计。



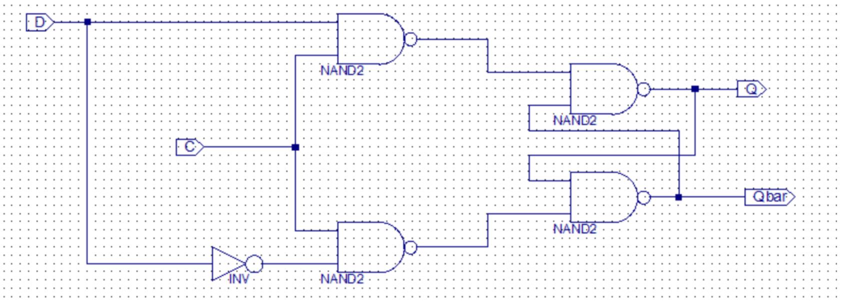
3. 新建源文件 CSR_LATCH.sch

用原理图方式设计。



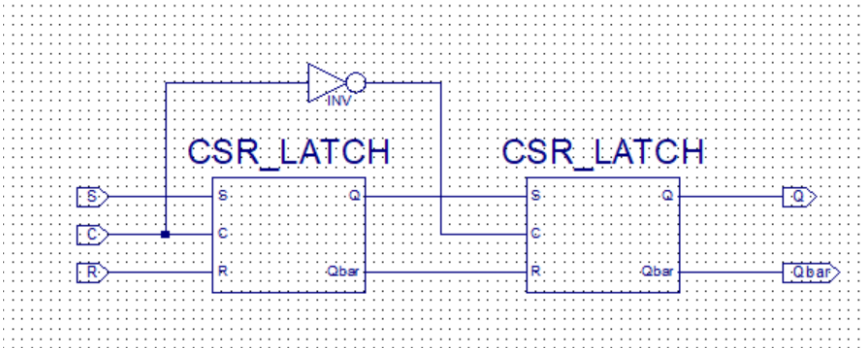
4. 新建源文件 D_LATCH.sch

用原理图方式设计。



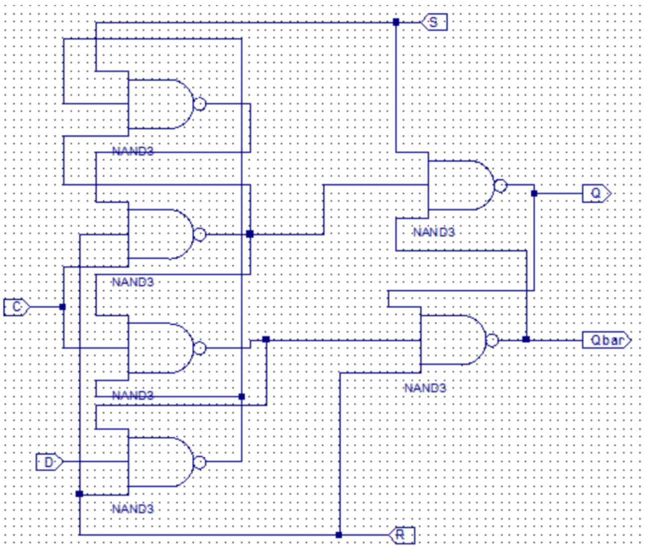
5. 新建源文件 MS_FLIPFLOP.sch

用原理图方式设计，调用 CSR_LATCH 实现。



6. 新建源文件 D_FLIPFLOP.sch

用原理图方式设计。



7. 新建源文件

类型是 Verilog，文件名 Top。

右键设为“Set as Top Module”。

```
module TOP(
       input clk,
       input [15:0] SW,
       input [3:0] BTN,
       output [7:0] LED,
       output ledclk,
       output ledsout,
       output ledclrn,
       output LEDEN,
       output BTNX3
   );
   wire [31:0] div;
   wire [3:0] BTN_OUT;
   wire CK;
   wire [15:0] num;
   wire [8:0] NLED;
   assign BTNX3=0;
   assign NLED={~LED8,~LED};

   pbdebounce p1(div[17],BTN[0],BTN_OUT[0]);
   pbdebounce p2(div[17],BTN[1],BTN_OUT[1]);
   pbdebounce p3(div[17],BTN[2],BTN_OUT[2]);
   pbdebounce p4(div[17],BTN[3],BTN_OUT[3]);

   clkdiv_pulse
m0(.clk(clk),.rst(1'b0),.Sel_CLK(SW[15]),.pulse(BTN_OUT[0]),.CK(CK
),.clkdiv(div[31:0]));
```

```
    CSR_LATCH
Mm2(.C(CK),.R(SW[0]),.S(SW[1]),.Q(LED[1]),.Qn(LED[0]));
    D_LATCH Mm3(.C(CK),.D(SW[2]),.Q(LED[3]),.Qn(LED[2]));
    MS_FLIPFLOP
Mm4(.C(CK),.R(SW[3]),.S(SW[4]),.Y(LED[6]),.Q(LED[5]),.Qn(LED[4]));
    D_FLIPFLOP
Mm5(.C(CK),.D(SW[5]),.Q(LED8),.Qn(LED[7]),.Sn(1'b1),.Rn(1'b1));

    LEDP2S #(.DATA_BITS(16),.DATA_COUNT_BITS(4),.DIR(0))

    U7(.clk(clk), .rst(1'b0), .Start(div[20]), .PData({7'h3F,NLED[8
:0]}), .sclk(ledclk), .sclrn(ledclrn), .sout(ledsout), .EN(LEDEN));
```

pbdebounce 模块：

```
module pbdebounce(
    input wire clk_1ms,
    input wire button,
    output reg pbreg
    );

    reg [7:0] pbshift;

    always@(posedge clk_1ms) begin
        pbshift=pbshift<<1;
        pbshift[0]=button;
        if (pbshift==8'b0)
            pbreg=0;
        if (pbshift==8'hFF)
            pbreg=1;
    end
endmodule
```

clkdiv_pulse 模块：

```
module clkdiv_pulse(input clk,
                    input rst,
                    input Sel_CLK,
                    input pulse,
                    output CK,
output reg [31:0] clkdiv
                    );
    always@(posedge clk or posedge rst) begin
if(rst) clkdiv<=0;
        else clkdiv<=clkdiv + 1'b1;
```

```
   end
   assign CK=(Sel_CLK) ? ~pulse : clkdiv[26];

endmodule
```

8. 下载验证：

```
#clk,          [15:0]          SW,          BTN[3:0],          [7:0]
LED,ledclk,ledsout,ledclrn,LEDEN,output BTNX3,
NET "ledclk"    LOC = N26 | IOSTANDARD = LVCMOS33;
NET "ledclrn"   LOC = N24 | IOSTANDARD = LVCMOS33;
NET "ledsout"   LOC = M26 | IOSTANDARD = LVCMOS33;
NET "LEDEN"    LOC = P18 | IOSTANDARD = LVCMOS33;

NET "clk"  LOC = AC18 | IOSTANDARD = LVCMOS18;
NET "clk"  TNM_NET = TM_CLK;
TIMESPEC TS_CLK_100M = PERIOD "TM_CLK" 10ns HIGH 50%;
#NET "RSTN" LOC =W13 | IOSTANDARD =  LVCMOS18;
NET "BTN[0]" LOC = V18 | IOSTANDARD = LVCMOS18;
NET "BTN[0]" CLOCK_DEDICATED_ROUTE = FALSE;
NET "BTN[1]" LOC = V19 | IOSTANDARD = LVCMOS18;
NET "BTN[1]" CLOCK_DEDICATED_ROUTE = FALSE;
NET "BTN[2]" LOC = V14 | IOSTANDARD = LVCMOS18;
NET "BTN[2]" CLOCK_DEDICATED_ROUTE = FALSE;
NET "BTN[3]" LOC = W14 | IOSTANDARD = LVCMOS18;
NET "BTN[3]" CLOCK_DEDICATED_ROUTE = FALSE;
NET "SW[0]"LOC = AA10 | IOSTANDARD = LVCMOS15;
NET "SW[1]"LOC = AB10 | IOSTANDARD = LVCMOS15;
NET "SW[2]"LOC = AA13 | IOSTANDARD = LVCMOS15;
NET "SW[3]"LOC = AA12 | IOSTANDARD = LVCMOS15;
NET "SW[4]"LOC =Y13  | IOSTANDARD = LVCMOS15;
NET "SW[5]"LOC =Y12  | IOSTANDARD = LVCMOS15;
NET "SW[6]"LOC =AD11  | IOSTANDARD = LVCMOS15;
NET "SW[7]"LOC =AD10  | IOSTANDARD = LVCMOS15;
NET "SW[8]"LOC =AE10 | IOSTANDARD = LVCMOS15;
NET "SW[9]"LOC =AE12  | IOSTANDARD = LVCMOS15;
NET "SW[10]"LOC =AF12  | IOSTANDARD = LVCMOS15;
NET "SW[11]"LOC =AE8  | IOSTANDARD = LVCMOS15;
NET "SW[12]"LOC =AF8  | IOSTANDARD = LVCMOS15;
NET "SW[13]"LOC =AE13  | IOSTANDARD = LVCMOS15;
NET "SW[14]"LOC =AF13  | IOSTANDARD = LVCMOS15;
NET "SW[15]"LOC =AF10  | IOSTANDARD = LVCMOS15;
NET"LED[0]"LOC=W23 | IOSTANDARD=LVCMOS33;
NET"LED[1]"LOC=AB26 | IOSTANDARD=LVCMOS33;
```

```
NET"LED[2]"LOC=Y25 | IOSTANDARD=LVCMOS33;
NET"LED[3]"LOC=AA23 | IOSTANDARD=LVCMOS33;
NET"LED[4]"LOC=Y23 | IOSTANDARD=LVCMOS33;
NET"LED[5]"LOC=Y22 | IOSTANDARD=LVCMOS33;
NET"LED[6]"LOC=AE21 | IOSTANDARD=LVCMOS33;
NET"LED[7]"LOC=AF24 | IOSTANDARD=LVCMOS33;
NET "BTNX3" LOC = W15 | IOSTANDARD = LVCMOS18;
```

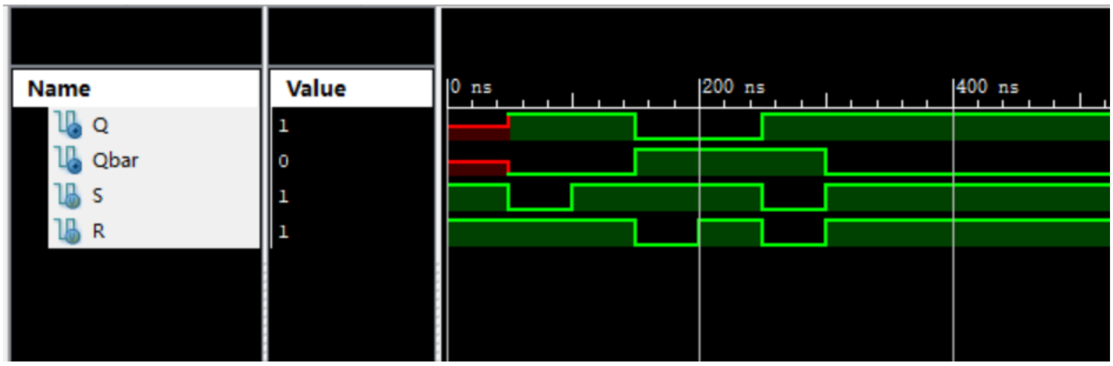
二、实验结果与分析

1.SR_LATCH 仿真

仿真文件：

```
`timescale 1ns / 1ps
module SR_LATCH_SR_LATCH_sch_tb();
// Inputs
  reg S;
  reg R;

// Output
  wire Q;
  wire Qbar;

// Bidirs

// Instantiate the UUT
  SR_LATCH UUT (
    .S(S),
    .R(R),
    .Q(Q),
    .Qbar(Qbar)
  );
// Initialize Inputs
    initial begin
      R=1;S=1; #50;
      R=1;S=0; #50;
      R=1;S=1; #50;
      R=0;S=1; #50;
      R=1;S=1; #50;
      R=0;S=0; #50;
      R=1;S=1; #50;
    end
endmodule
```

仿真结果：



符合理论上的结果预期，该元件设计正确。

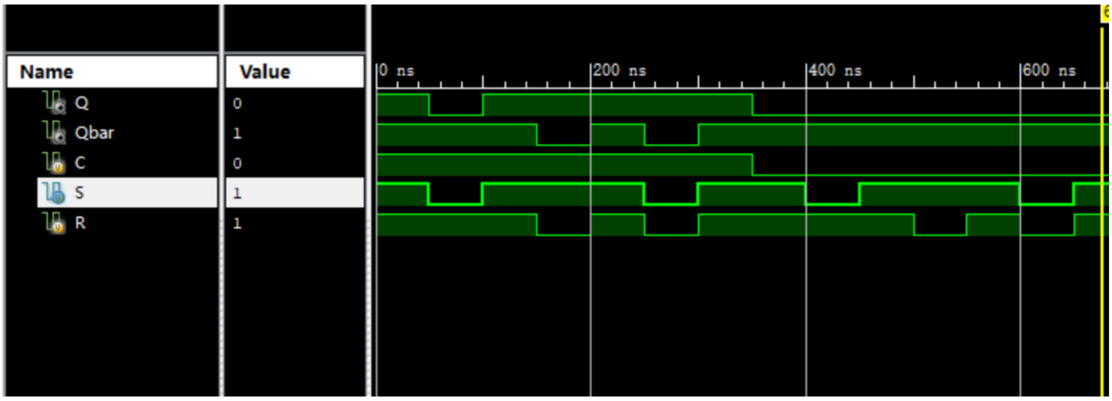
2. CSR_LATCH 仿真

仿真文件：

```
`timescale 1ns / 1ps
module CSR_LATCH_CSR_LATCH_sch_tb();
// Inputs
  reg C;
  reg S;
  reg R;
// Output
  wire Q;
  wire Qbar;
// Bidirs

// Instantiate the UUT
  CSR_LATCH UUT (
    .C(C),
    .S(S),
    .R(R),
    .Q(Q),
    .Qbar(Qbar)
  );
// Initialize Inputs
  initial begin
    C=1;R=1;S=1; #50;
    R=1;S=0; #50;
    R=1;S=1; #50;
    R=0;S=1; #50;
    R=1;S=1; #50;
R=0;S=0; #50;
R=1;S=1; #50;
C=0;R=1;S=1; #50;
```

```
R=1;S=0; #50;
    R=1;S=1; #50;
    R=0;S=1; #50;
    R=1;S=1; #50;
    R=0;S=0; #50;
    R=1;S=1; #50;
  end
endmodule
```

仿真结果：



符合理论上的结果预期，该元件设计正确。

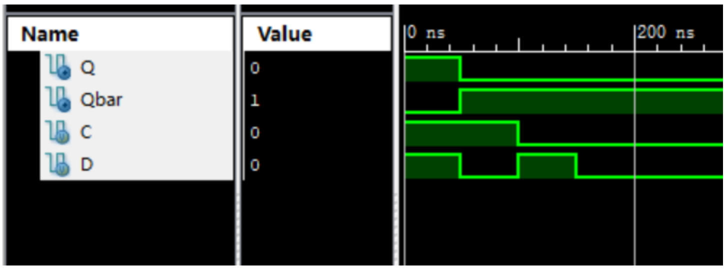
3. D_LATCH 仿真

仿真文件：

```
`timescale 1ns / 1ps

module D_LATCH_D_LATCH_sch_tb();
// Inputs
  reg C;
  reg D;

// Output
  wire Q;
  wire Qbar;

// Bidirs
// Instantiate the UUT
  D_LATCH UUT (
    .C(C),
    .Q(Q),
    .Qbar(Qbar),
    .D(D)
  );
```

```
// Initialize Inputs
  initial begin
    C=1;D=1; #50;
    D=0; #50;
    C=0;D=1; #50;
    D=0;
  end
endmodule
```

仿真结果：



符合理论上的结果预期，该元件设计正确。

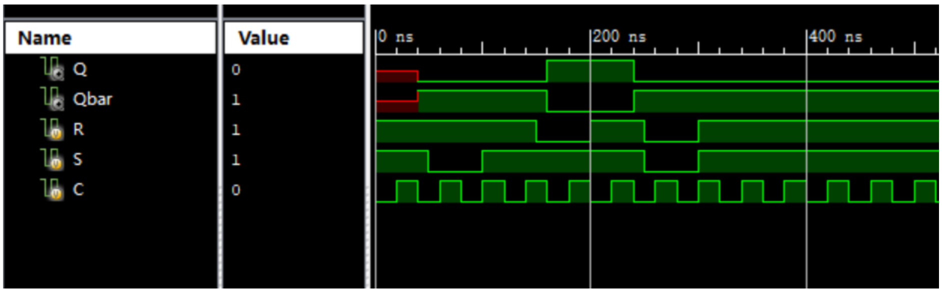
4. MS_FLIPFLOP 仿真

仿真文件：

```
`timescale 1ns / 1ps

module MS_FLIPFLOP_MS_FLIPFLOP_sch_tb();

// Inputs
  reg R;
  reg S;
  reg C;

// Output
  wire Q;
  wire Qbar;

// Bidirs

// Instantiate the UUT
  MS_FLIPFLOP UUT (
    .Q(Q),
    .Qbar(Qbar),
    .R(R),
    .S(S),
    .C(C)
  );
```

```
// Initialize Inputs
  initial begin
    R=1;S=1; #50;
    R=1;S=0; #50;
    R=1;S=1; #50;
    R=0;S=1; #50;
    R=1;S=1; #50;
    R=0;S=0; #50;
    R=1;S=1; #50;
  end
  always begin
    C=0;#20;
    C=1;#20;
  end
endmodule
```

仿真结果：

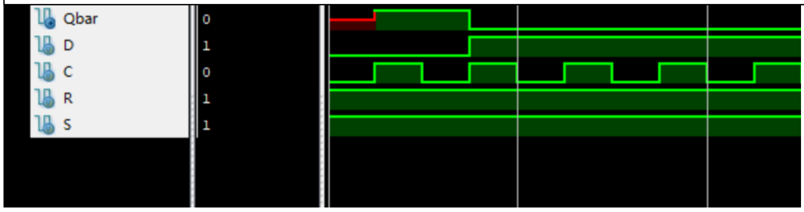


符合理论上的结果预期，该元件设计正确。

5. D_FLIPFLOP 仿真

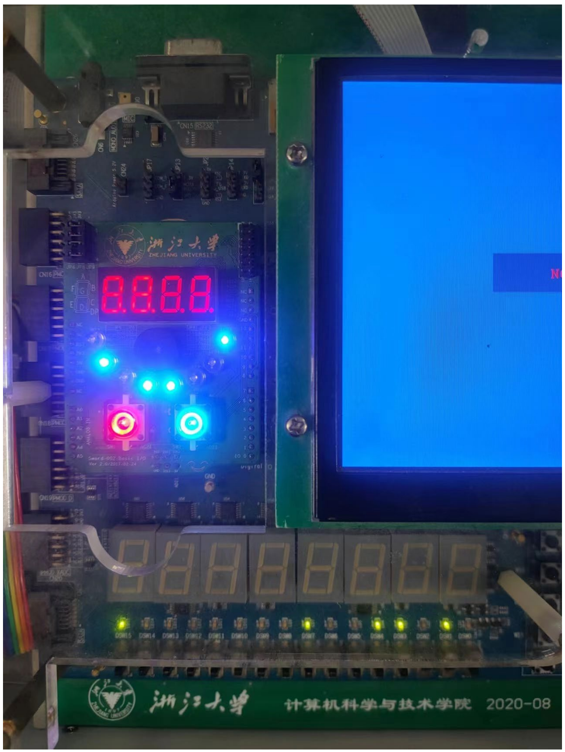
仿真文件：

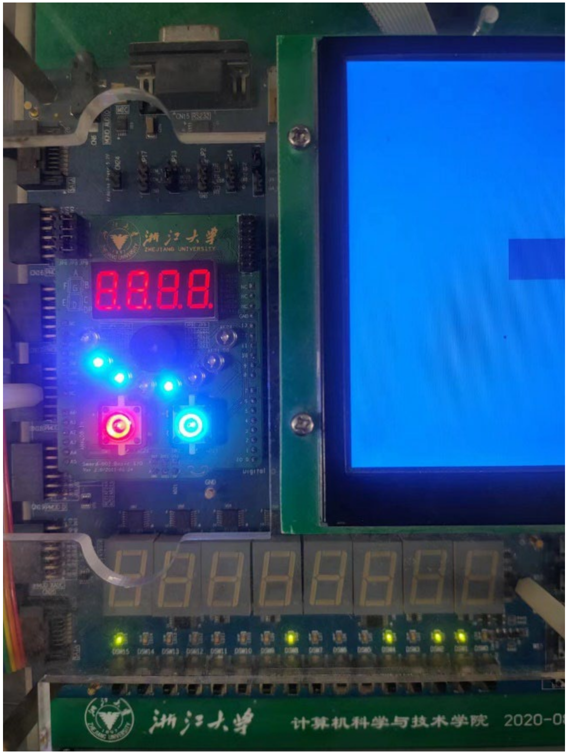
```
`timescale 1ns / 1ps

module D_FLIPFLOP_D_FLIPFLOP_sch_tb();

// Inputs
  reg C;
  reg R;
  reg S;
  reg D;

// Output
  wire Qbar;
  wire Q;
// Bidirs
// Instantiate the UUT
  D_FLIPFLOP UUT (
    .C(C),

.R(R),
    .Qbar(Qbar),
    .Q(Q),
    .S(S),
    .D(D)
  );
// Initialize Inputs
  initial begin
    S = 1;
    R = 1;
    D = 0; #150;
    D = 1; #150;
  end
  always begin
    C=0; #50;
    C=1; #50;
  end
endmodule
```

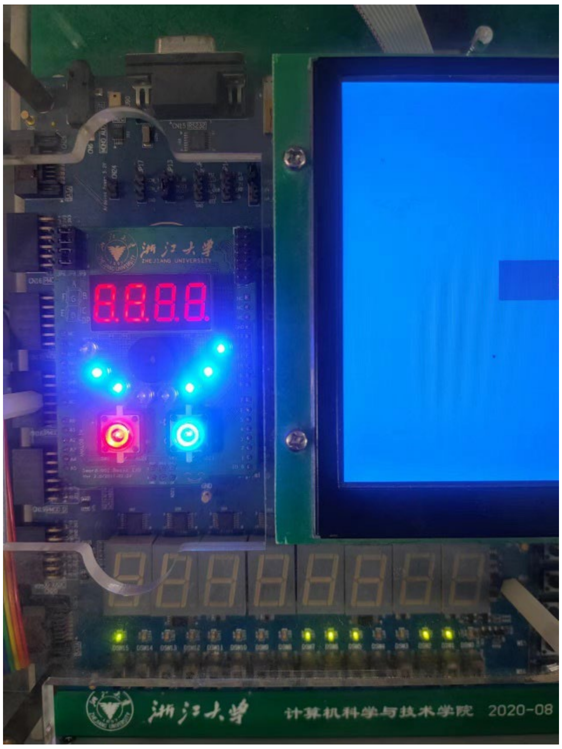


符合理论上的结果预期，该元件设计正确。

6. 上板后的实验结果


浙江大学
ZHEJIANG UNIVERSITY
浙江大学
计算机科学与技术学院 2020-08


浙江大学
ZHEJIANG UNIVERSITY
浙江大学
计算机科学与技术学院 2020-0

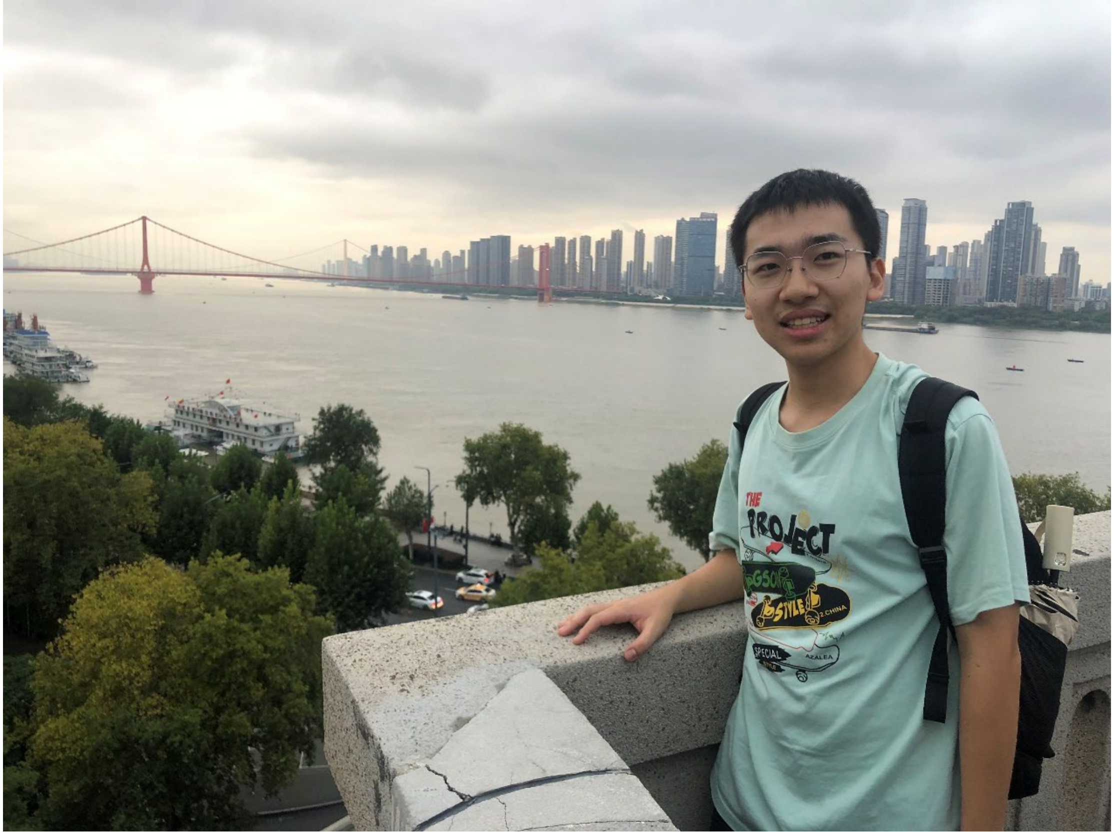


在点击按钮后，一盏灯会等一盏灯熄灭，一段时间后，另一盏灯会亮起。部分按钮会再亮一盏灯。说明实验结果正确。

三、讨论、心得

在本次实验中，由于 vivado 的仿真会出现问题，我使用 ise 完成了本次实验。在同学的帮助下，我了解了 ise 操作的基本步骤，也学会了本次锁存器与触发器的基本原理。通过绘图和仿真，我对实现机理和实际应用也有了更多的了解。

四、个人生活照片


THE
PROJECT
STYLE
CHINA
SPECIAL
AZALEA

浙江大學

本科实验报告

课程名称：数字逻辑电路设计

姓名：蔡佳伟

学院：计算机科学与技术学院

专业：软件工程

邮箱：3220104519@zju.edu.cn

QQ 号：3348536459

电话：19550230334

指导教师：洪奇军

报告日期：2023 年 11 月 28 日

浙江大学实验报告

课程名称：　　数字逻辑设计　　实验类型：　　综合

实验项目名称：　　实验 10：同步时序电路设计

学生姓名：　蔡佳伟　　学号：　3220104519　同组学生姓名：

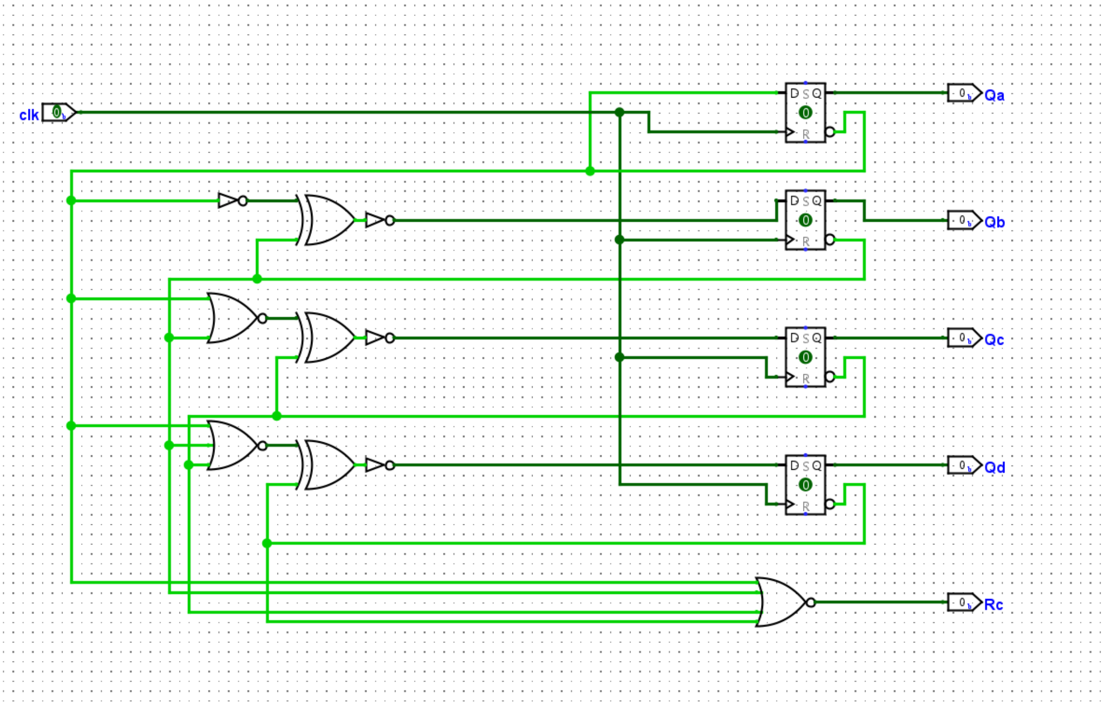
实验地点：　紫金港东四 509 室　实验日期：　2023　年　11　月　28　日

一、操作方法与实验步骤

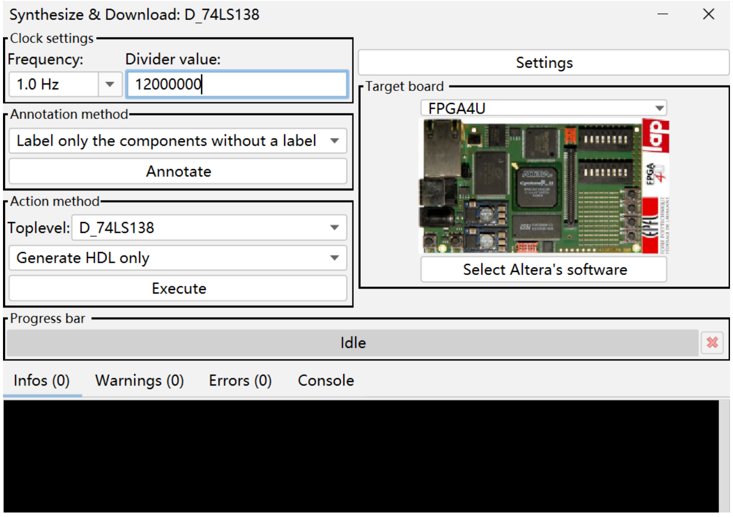
（一）原理图设计实现四位同步二进制计数器

1、使用 Logisim 绘制四位同步二进制计数器

绘制结果如下：



需要更改入口和出口名称，本次按照图片更改为 clk、Qa、Qb、Qc、Qd、Rc 命名为 Counter4b，之后导出为 Verilog 代码



Synthesize & Download: D_74LS138

Clock settings
Frequency: 1.0 Hz
Divider value: 12000000

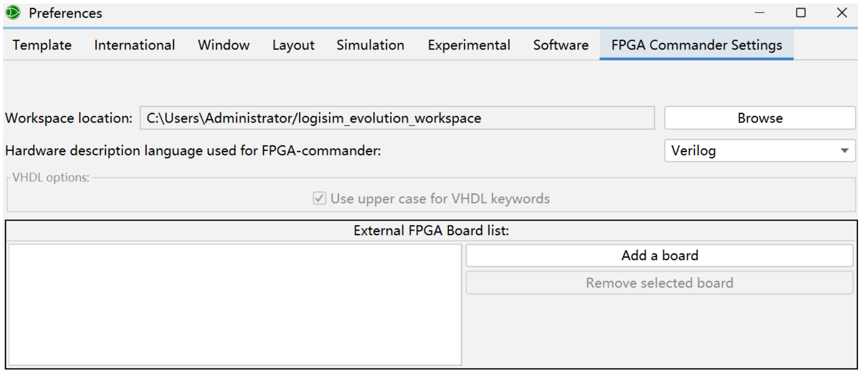
Annotation method: Label only the components without a label

Action method
Toplevel: D_74LS138
Generate HDL only

Target board: FPGA4U

Progress bar: Idle

注意 Target board 选择 FPGA4U



Preferences

Workspace location: C:\Users\Administrator/logisim_evolution_workspace

Hardware description language used for FPGA-commander: Verilog

VHDL options:
☑ Use upper case for VHDL keywords

External FPGA Board list:

注意在 preferences 中把代码格式选为 Verilog

之后 Execute 生成代码，保存

Verilog 代码如下：

```
/**********************************************************************
**********
 **   Logisim-evolution   goes   FPGA   automatic   generated   Verilog
code                **
 **
https://github.com/logisim-evolution/
 **
 **
  **
 **                         Component                              :
Counter4b                                                    **
 **
  **
 **********************************************************************
*********/

module Counter4b( Qa,
                  Qb,
                  Qc,
                  Qd,
                  Rc,
                  clk );

    /******************************************************************
**************
```

```
           **         The         inputs         are         defined
here                                           **
   ******************************************************************
*************/
   input clk;

   /*****************************************************************
**************
           **         The         outputs         are        defined
here                                          **
   ******************************************************************
*************/
   output Qa;
   output Qb;
   output Qc;
   output Qd;
   output Rc;

   /*****************************************************************
**************
           **         The         wires          are         defined
here                                            **
   ******************************************************************
*************/
   wire s_logisimNet0;
   wire s_logisimNet1;
   wire s_logisimNet10;
   wire s_logisimNet11;
   wire s_logisimNet12;
   wire s_logisimNet13;
   wire s_logisimNet14;
   wire s_logisimNet15;
   wire s_logisimNet16;
   wire s_logisimNet17;
   wire s_logisimNet18;
   wire s_logisimNet2;
   wire s_logisimNet3;
   wire s_logisimNet4;
   wire s_logisimNet5;
   wire s_logisimNet6;
   wire s_logisimNet7;
   wire s_logisimNet8;
   wire s_logisimNet9;

   /*********************************************************
```

2、对 Counter4b 进行仿真

先自行书写仿真代码，代码如下：

```
`timescale 1ns / 1ps

module Counter4b_tb();

// Inputs
reg clk;

// Output
wire Rc;
wire Qa;
wire Qb;
wire Qc;
wire Qd;

// Instantiate the UUT
Counter4b Counter4b_inst (
.clk(clk),
.Rc(Rc),
.Qa(Qa),
.Qb(Qb),
.Qc(Qc),
.Qd(Qd)
);

integer i;
initial begin
    for(i=0;i<16;i=i+1)begin
        clk=0;#20;
        clk=1;#20;
    end
end
endmodule
```

之后导入仿真代码和设计代码，进行仿真，仿真结果在“实验结果与分析”中呈现

3、上板验证

导入顶层 Top 模块，代码如下：

```
module Top(
    input wire clk,
    output wire LED,
    output wire [7:0] SEGMENT,
    output wire [3:0] AN
);

    wire Qa;
    wire Qb;
    wire Qc;
    wire Qd;
    wire [3:0] Hex;

    /* module clk_1s at submodules/clk_1s.v */
    clk_1s m0(.clk(clk), .clk_1s(clk_1s));

    /* You need to implement module Counter4b */
    Counter4b
m1(.clk(clk_1s), .Qa(Qa), .Qb(Qb), .Qc(Qc), .Qd(Qd), .Rc(LED));

    assign Hex = {Qd, Qc, Qb, Qa};

    // Please replace module below with your module completed in Lab 6
    // Pay attention to the correctness of the module name and port name
    // NOTE: SEGMENT and Segement are different port names

    // BTN[0]: LE, valid with value 0
    // BTN[1]: point, light with value 1
    // SW[7:4]: AN, light with value 1(AN[i] = ~SW[i+4])
    // SW[3:0]: number to display
    DispNum              display(.BTN(2'b00),              .SW({4'b0001,
Hex}), .SEGMENT(SEGMENT), .AN(AN));

endmodule
```

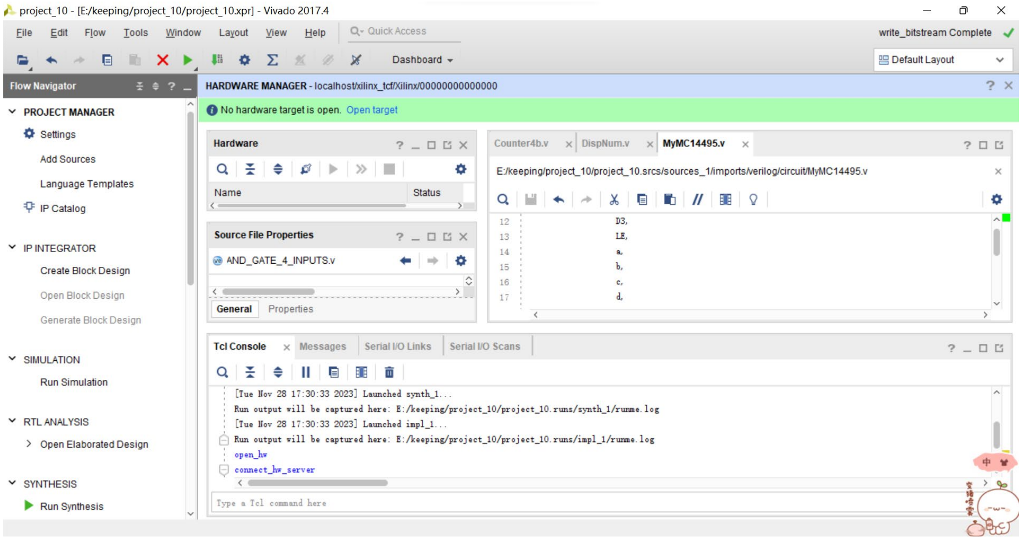
导入 clk1s_v 模块，代码如下：

```
`timescale 1ns / 1ps

module clk_1s(
    input clk,
    output reg clk_1s
);

    reg [31:0] cnt;

    initial begin
        cnt = 32'b0;
    end

    wire[31:0] cnt_next;
    assign cnt_next = cnt + 1'b1;

    always @(posedge clk) begin
        if(cnt<50_000_000)begin
            cnt <= cnt_next;
        end
        else begin
            cnt <= 0;
            clk_1s <= ~clk_1s;
        end
    end

endmodule
```

导入约束文件，代码如下：

```
# Filename: constraints_labA_part1.xdc
## Constraints file for LabA-part1

# Main clock
set_property PACKAGE_PIN AC18 [get_ports clk]
set_property IOSTANDARD LVCMOS18 [get_ports clk]

create_clock -period 10.000 -name clk [get_ports "clk"]

# LED
set_property PACKAGE_PIN AF24 [get_ports {LED}]
set_property IOSTANDARD LVCMOS33 [get_ports {LED}]

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]}]
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 {AN[0]}]
set_property IOSTANDARD LVCMOS33 [get_ports {AN[1]}]
set_property IOSTANDARD LVCMOS33 [get_ports {AN[2]}]
set_property IOSTANDARD LVCMOS33 [get_ports {AN[3]}]
set_property IOSTANDARD LVCMOS33 [get_ports {SEGMENT[0]}]
set_property IOSTANDARD LVCMOS33 [get_ports {SEGMENT[1]}]
set_property IOSTANDARD LVCMOS33 [get_ports {SEGMENT[2]}]
set_property IOSTANDARD LVCMOS33 [get_ports {SEGMENT[3]}]
set_property IOSTANDARD LVCMOS33 [get_ports {SEGMENT[4]}]
set_property IOSTANDARD LVCMOS33 [get_ports {SEGMENT[5]}]
set_property IOSTANDARD LVCMOS33 [get_ports {SEGMENT[6]}]
set_property IOSTANDARD LVCMOS33 [get_ports {SEGMENT[7]}]


# # Main clock
# set_property PACKAGE_PIN AC18 [get_ports clk_p]
# set_property PACKAGE_PIN AD18 [get_ports clk_n]
# set_property IOSTANDARD LVCMOS18 [get_ports clk_p]
# set_property IOSTANDARD LVCMOS18 [get_ports clk_n]

# # create_clock -period 10.000 -name clk [get_ports "clk_p"]

# # FPGA RST
# set_property PACKAGE_PIN W13 [get_ports RSTN]
# set_property IOSTANDARD LVCMOS18 [get_ports RSTN]

# # 7SEG
# set_property PACKAGE_PIN M24 [get_ports seg_clk]
# set_property PACKAGE_PIN L24 [get_ports set_sout]
# set_property PACKAGE_PIN R18 [get_ports seg_pen]
# set_property PACKAGE_PIN M20 [get_ports seg_clrn]
# set_property IOSTANDARD LVCMOS33 [get_ports seg_clk]
# set_property IOSTANDARD LVCMOS33 [get_ports set_sout]
# set_property IOSTANDARD LVCMOS33 [get_ports seg_pen]
# set_property IOSTANDARD LVCMOS33 [get_ports seg_clrn]

# # Audio out
# set_property PACKAGE_PIN P26 [get_ports AUD_PWM]
# set_property PACKAGE_PIN M25 [get_ports AUD_SD]
# set_property IOSTANDARD LVCMOS33 [get_ports AUD_PWM]
# set_property IOSTANDARD LVCMOS33 [get_ports AUD_SD]

# # Key Array
# set_property PACKAGE_PIN V17 [get_ports BTN_X0]
# set_property IOSTANDARD LVCMOS18 [get_ports BTN_X0]
# set_property PACKAGE_PIN W18 [get_ports BTN_X1]
# set_property IOSTANDARD LVCMOS18 [get_ports BTN_X1]
# set_property PACKAGE_PIN W19 [get_ports BTN_X2]
# set_property IOSTANDARD LVCMOS18 [get_ports BTN_X2]
# set_property PACKAGE_PIN W15 [get_ports BTN_X3]
# set_property IOSTANDARD LVCMOS18 [get_ports BTN_X3]
# set_property PACKAGE_PIN W16 [get_ports BTN_X4]
# set_property IOSTANDARD LVCMOS18 [get_ports BTN_X4]
# set_property PACKAGE_PIN V18 [get_ports BTN_Y0]
# set_property IOSTANDARD LVCMOS18 [get_ports BTN_Y0]
# set_property PACKAGE_PIN V19 [get_ports BTN_Y1]
# set_property IOSTANDARD LVCMOS18 [get_ports BTN_Y1]
# set_property PACKAGE_PIN V14 [get_ports BTN_Y2]
# set_property IOSTANDARD LVCMOS18 [get_ports BTN_Y2]
# set_property PACKAGE_PIN W14 [get_ports BTN_Y3]
# set_property IOSTANDARD LVCMOS18 [get_ports BTN_Y3]

# # Arduino
# set_property PACKAGE_PIN AF25 [get_ports ard_rst]
# set_property IOSTANDARD LVCMOS33 [get_ports ard_rst]
# set_property PACKAGE_PIN AF24 [get_ports {ard_led[0]}]
# set_property PACKAGE_PIN AE21 [get_ports {ard_led[1]}]
# set_property PACKAGE_PIN Y22 [get_ports {ard_led[2]}]
# set_property PACKAGE_PIN Y23 [get_ports {ard_led[3]}]
# set_property PACKAGE_PIN AA23 [get_ports {ard_led[4]}]
# set_property PACKAGE_PIN Y25 [get_ports {ard_led[5]}]
# set_property PACKAGE_PIN AB26 [get_ports {ard_led[6]}]
# set_property PACKAGE_PIN W23 [get_ports {ard_led[7]}]
# set_property IOSTANDARD LVCMOS33 [get_ports {ard_led[0]}]
# set_property IOSTANDARD LVCMOS33 [get_ports {ard_led[1]}]
# set_property IOSTANDARD LVCMOS33 [get_ports {ard_led[2]}]
# set_property IOSTANDARD LVCMOS33 [get_ports {ard_led[3]}]
# set_property IOSTANDARD LVCMOS33 [get_ports {ard_led[4]}]
# set_property IOSTANDARD LVCMOS33 [get_ports {ard_led[5]}]
# set_property IOSTANDARD LVCMOS33 [get_ports {ard_led[6]}]
# set_property IOSTANDARD LVCMOS33 [get_ports {ard_led[7]}]
# set_property PACKAGE_PIN AD21 [get_ports {ard_an[0]}]
# set_property PACKAGE_PIN AC21 [get_ports {ard_an[1]}]
# set_property PACKAGE_PIN AB21 [get_ports {ard_an[2]}]
# set_property PACKAGE_PIN AC22 [get_ports {ard_an[3]}]
# set_property PACKAGE_PIN AB22 [get_ports {ard_seg[0]}]
# set_property PACKAGE_PIN AD24 [get_ports {ard_seg[1]}]
# set_property PACKAGE_PIN AD23 [get_ports {ard_seg[2]}]
# set_property PACKAGE_PIN Y21 [get_ports {ard_seg[3]}]
# set_property PACKAGE_PIN W20 [get_ports {ard_seg[4]}]
# set_property PACKAGE_PIN AC24 [get_ports {ard_seg[5]}]
# set_property PACKAGE_PIN AC23 [get_ports {ard_seg[6]}]
# set_property PACKAGE_PIN AA22 [get_ports {ard_seg[7]}]
# # set_property IOSTANDARD LVCMOS33 [get_ports {ard_dio[13]}]
# # set_property IOSTANDARD LVCMOS33 [get_ports {ard_dio[12]}]
# set_property IOSTANDARD LVCMOS33 [get_ports {ard_an[0]}]
# set_property IOSTANDARD LVCMOS33 [get_ports {ard_an[1]}]
# set_property IOSTANDARD LVCMOS33 [get_ports {ard_an[2]}]
# set_property IOSTANDARD LVCMOS33 [get_ports {ard_an[3]}]
# set_property IOSTANDARD LVCMOS33 [get_ports {ard_seg[0]}]
# set_property IOSTANDARD LVCMOS33 [get_ports {ard_seg[1]}]
# set_property IOSTANDARD LVCMOS33 [get_ports {ard_seg[2]}]
# set_property IOSTANDARD LVCMOS33 [get_ports {ard_seg[3]}]
# set_property IOSTANDARD LVCMOS33 [get_ports {ard_seg[4]}]
# set_property IOSTANDARD LVCMOS33 [get_ports {ard_seg[5]}]
# set_property IOSTANDARD LVCMOS33 [get_ports {ard_seg[6]}]
# set_property IOSTANDARD LVCMOS33 [get_ports {ard_seg[7]}]

# #16leds
# set_property PACKAGE_PIN N26 [get_ports LEDCLK]
# set_property PACKAGE_PIN N24 [get_ports LEDCLR]
# set_property PACKAGE_PIN M26 [get_ports LEDDT]
# set_property IOSTANDARD LVCMOS33 [get_ports LEDCLK]
# set_property IOSTANDARD LVCMOS33 [get_ports LEDCLR]
# set_property IOSTANDARD LVCMOS33 [get_ports LEDDT]

# #16dips
# set_property PACKAGE_PIN AA10 [get_ports {switch[0]}]
# set_property PACKAGE_PIN AB10 [get_ports {switch[1]}]
# set_property PACKAGE_PIN AA13 [get_ports {switch[2]}]
# set_property PACKAGE_PIN AA12 [get_ports {switch[3]}]
# set_property PACKAGE_PIN Y13 [get_ports {switch[4]}]
# set_property PACKAGE_PIN Y12 [get_ports {switch[5]}]
# set_property PACKAGE_PIN AD11 [get_ports {switch[6]}]
# set_property PACKAGE_PIN AD10 [get_ports {switch[7]}]
# set_property PACKAGE_PIN AE10 [get_ports {switch[8]}]
# set_property PACKAGE_PIN AE12 [get_ports {switch[9]}]
# set_property PACKAGE_PIN AF12 [get_ports {switch[10]}]
# set_property PACKAGE_PIN AE8 [get_ports {switch[11]}]
# set_property PACKAGE_PIN AF8 [get_ports {switch[12]}]
# set_property PACKAGE_PIN AE13 [get_ports {switch[13]}]
# set_property PACKAGE_PIN AF13 [get_ports {switch[14]}]
# set_property PACKAGE_PIN AF10 [get_ports {switch[15]}]
# set_property IOSTANDARD LVCMOS15 [get_ports {switch[0]}]
# set_property IOSTANDARD LVCMOS15 [get_ports {switch[1]}]
# set_property IOSTANDARD LVCMOS15 [get_ports {switch[2]}]
# set_property IOSTANDARD LVCMOS15 [get_ports {switch[3]}]
# set_property IOSTANDARD LVCMOS15 [get_ports {switch[4]}]
# set_property IOSTANDARD LVCMOS15 [get_ports {switch[5]}]
# set_property IOSTANDARD LVCMOS15 [get_ports {switch[6]}]
# set_property IOSTANDARD LVCMOS15 [get_ports {switch[7]}]
# set_property IOSTANDARD LVCMOS15 [get_ports {switch[8]}]
# set_property IOSTANDARD LVCMOS15 [get_ports {switch[9]}]
# set_property IOSTANDARD LVCMOS15 [get_ports {switch[10]}]
# set_property IOSTANDARD LVCMOS15 [get_ports {switch[11]}]
# set_property IOSTANDARD LVCMOS15 [get_ports {switch[12]}]
# set_property IOSTANDARD LVCMOS15 [get_ports {switch[13]}]
# set_property IOSTANDARD LVCMOS15 [get_ports {switch[14]}]
# set_property IOSTANDARD LVCMOS15 [get_ports {switch[15]}]

# # VGA
# set_property PACKAGE_PIN N21 [get_ports {vga_red[0]}]
# set_property PACKAGE_PIN N22 [get_ports {vga_red[1]}]
# set_property PACKAGE_PIN R21 [get_ports {vga_red[2]}]
# set_property PACKAGE_PIN P21 [get_ports {vga_red[3]}]
# set_property IOSTANDARD LVCMOS33 [get_ports {vga_red[0]}]
# set_property IOSTANDARD LVCMOS33 [get_ports {vga_red[1]}]
# set_property IOSTANDARD LVCMOS33 [get_ports {vga_red[2]}]
# set_property IOSTANDARD LVCMOS33 [get_ports {vga_red[3]}]
# set_property PACKAGE_PIN R22 [get_ports {vga_green[0]}]
# set_property PACKAGE_PIN R23 [get_ports {vga_green[1]}]
# set_property PACKAGE_PIN T24 [get_ports {vga_green[2]}]
# set_property PACKAGE_PIN T25 [get_ports {vga_green[3]}]
# set_property IOSTANDARD LVCMOS33 [get_ports {vga_green[0]}]
# set_property IOSTANDARD LVCMOS33 [get_ports {vga_green[1]}]
# set_property IOSTANDARD LVCMOS33 [get_ports {vga_green[2]}]
# set_property IOSTANDARD LVCMOS33 [get_ports {vga_green[3]}]
# set_property PACKAGE_PIN T20 [get_ports {vga_blue[0]}]
# set_property PACKAGE_PIN R20 [get_ports {vga_blue[1]}]
# set_property PACKAGE_PIN T22 [get_ports {vga_blue[2]}]
# set_property PACKAGE_PIN T23 [get_ports {vga_blue[3]}]
# set_property IOSTANDARD LVCMOS33 [get_ports {vga_blue[0]}]
# set_property IOSTANDARD LVCMOS33 [get_ports {vga_blue[1]}]
# set_property IOSTANDARD LVCMOS33 [get_ports {vga_blue[2]}]
# set_property IOSTANDARD LVCMOS33 [get_ports {vga_blue[3]}]
# set_property PACKAGE_PIN M22 [get_ports vga_hs]
# set_property PACKAGE_PIN [get_ports vga_vs]
# set_property IOSTANDARD LVCMOS33 [get_ports vga_hs]
# set_property IOSTANDARD LVCMOS33 [get_ports vga_vs]
```

在导入以后，总体文件结构如下：



HARDWARE MANAGER - localhost/xilinx_tcf/Xilinx/00000000000000

No hardware target is open. Open target

E:/keeping/project_10/project_10.srcs/sources_1/imports/verilog/circuit/MyMC14495.v

```
D3,
LE,
a,
b,
c,
d,
```

```
[Tue Nov 28 17:30:33 2023] Launched synth_1...
Run output will be captured here: E:/keeping/project_10/project_10.runs/synth_1/runme.log
[Tue Nov 28 17:30:33 2023] Launched impl_1...
Run output will be captured here: E:/keeping/project_10/project_10.runs/impl_1/runme.log
open_hw
connect_hw_server
```

可以进行上板验证，操作如下：

先生成 bitstream 并烧录，等待烧录完成后，将下载器连接到电脑上。点击 PROGRAM AND DEBUG > Open Hardware Manager > Open Target > Auto Connect 进行识别和连接，成功连接后，点击 Program Device 选择 xc7k160t 设备，在下载程序界面选择我们刚刚生成的比特流文件，将其下载到板上。之后在板上实现相关操作。

板上验证结果之后呈现。

（三）十六位可逆同步二进制计数器

1、使用 Verilog 代码实现 RevCounter 计数器，要求是：添加一个 s 控制器，并确保计数器功能：

在时钟上升沿对输出 cnt 进行修改；当 s = 0 时进行自增，s = 1 时进行自减。

同步重置信号 rst 高位有效，即在时钟上升沿时若 rst = 1 才进行重置，重置时将 cnt 修改为 0。

非饱和计数，当前计数若为 16'hFFFF 则自增后为 16'h0；当前计数若为 16'h0 则自减后为 16'hFFFF。

在这里撰写 RevCounter.v 代码如下：

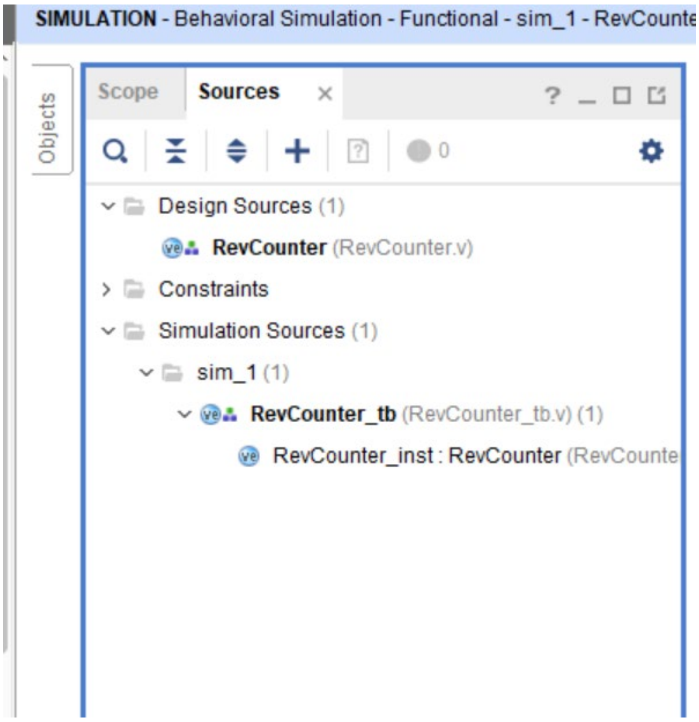
```
/** module RevCounter
  * input
    *   clk: A clock signal driven by module clk_1s.
    *   s: 1 for increment, 0 for decrement
    * output
    *   cnt: a 16-bits register
    *   Rc: rise when the counter reset(i.e. carry will be set), that is,
Rc becomes 1 when
    *           increment(s=1 & cnt=F) or decrement(s=0, cnt=0)
 */

//! NOTE: DO NOT CHANGE THE MODULE NAME & PORT NAMES
module RevCounter(
    input wire clk,
    input wire rst,
    input wire s,
    output reg [15:0] cnt = 0,
    output wire Rc
);
```

```
    always @(posedge clk) begin
        if (rst == 1'b1) begin
            cnt <= 16'h0; // 同步重置信号，将计数器重置为 0
        end
        else begin
            if (s == 1'b0) begin
                cnt <= (cnt == 16'hFFFF) ? 16'h0 : cnt + 1; // 自增操作，非饱和计数
            end
            else begin
                cnt <= (cnt == 16'h0) ? 16'hFFFF : cnt - 1; // 自减操作，非饱和计数
            end
        end
    end

    assign Rc = (s == 1'b0) ? (cnt == 16'hFFFF) : (cnt == 16'h0); // 当计数器的值为 0 时，反转信号为高电平

endmodule
```

2、对 RevCounter.v 模块进行仿真，仿真代码如下：

```
 `timescale 1ns / 1ps

module RevCounter_tb();

// Inputs
reg clk;
reg rst;
reg s;

// Output
wire Rc;
wire [15:0] cnt;

// Instantiate the UUT
RevCounter RevCounter_inst (
.clk(clk),
.rst(rst),
.s(s),
.Rc(Rc),
.cnt(cnt)
);

integer i;
initial begin
    rst = 0;
    s = 0;
    for(i=0;i<4;i=i+1)begin
        clk=1;#10;
        clk=0;#10;
    end
    rst = 1;
    for(i=0;i<2;i=i+1)begin
        clk=1;#10;
        clk=0;#10;
    end
    rst = 0;
    s = 1;
    for(i=0;i<4;i=i+1)begin
        clk=1;#10;
        clk=0;#10;
    end
    rst = 1;
    for(i=0;i<2;i=i+1)begin
        clk=1;#10;
        clk=0;#10;
    end
    rst = 0;

end
endmodule
```

之后，导入工程文件，进行仿真验证，工程文件模块如下：


SIMULATION - Behavioral Simulation - Functional - sim_1 - RevCounte

Design Sources (1)
RevCounter (RevCounter.v)
Constraints
Simulation Sources (1)
sim_1 (1)
RevCounter_tb (RevCounter_tb.v) (1)
RevCounter_inst : RevCounter (RevCounte

仿真结果在之后进行呈现。

3、导入顶层模块等文件下板进行验证

需要导入的文件除 RevCounter.v 外，文件及其代码如下：

Top.v

```
 module Top(
    input wire clk,
    input wire [1:0] SW,
    output wire LED,
    output wire [7:0] SEGMENT,
    output wire [3:0] AN
);

    wire[15:0] cnt;
    wire [3:0] Hex;
    wire clk_1s;

    /* module clk_100ms at submodules/clk_1s.v */
    clk_1s clk_div_1s (.clk(clk), .clk_1s(clk_1s));

    /* You need to implement module RevCounter */
    RevCounter
counter(.clk(clk_1s), .rst(SW[1]), .s(SW[0]), .cnt(cnt), .Rc(LED));
```

```
    // Please replace module below with your module completed in Lab **7**
    // imoprt submodules for module DisplayNumber from your prev. project
    DisplayNumber
display(.clk(clk), .rst(1'b0), .hexs(cnt), .LEs(4'b0000), .points(4'b0
000), .AN(AN), .SEGMENT(SEGMENT));

endmodule
```

DispNumber.v

```
 module DisplayNumber(
    input        clk,
    input        rst,
    input [15:0] hexs,
    input [ 3:0] points,
    input [ 3:0] LEs,
    output[ 3:0] AN,
    output[ 7:0] SEGMENT
);
    wire [31:0]scan;
    wire [3:0]HEX;
    wire point;
    wire LE;

    clkdiv cd(
        .clk(clk),
        .rst(rst),
        .div_res(scan)
    );

    DisplaySync ds(
        .scan(scan[18:17]),
        .hexs(hexs),
        .points(points),
        .LEs(LEs),
        .HEX(HEX),
        .point(point),
        .LE(LE),
        .AN(AN)
    );

    MyMC14495 mc(
        .D0(HEX[0]),
        .D1(HEX[1]),
        .D2(HEX[2]),
        .D3(HEX[3]),
        .point(point),
        .LE(LE),
        .a(SEGMENT[0]),
        .b(SEGMENT[1]),
        .c(SEGMENT[2]),
        .d(SEGMENT[3]),
        .e(SEGMENT[4]),
        .f(SEGMENT[5]),
        .g(SEGMENT[6]),
        .p(SEGMENT[7])
    );
endmodule
```

DisplaySyn.v

```

/*********************************************************************
```

```
*********
 **  Logisim-evolution  goes  FPGA  automatic  generated  Verilog  code
**
 **                              https://github.com/logisim-evolution/
**
 **                                                               **
 ** Component : DisplaySync                                           **
 **                                                               **

**********************************************************************
*******/

module DisplaySync( AN,
                 HEX,
                 LE,
                 LEs,
                 hexs,
                 point,
                 points,
                 scan );


/*********************************************************************
**********
  **        The        inputs        are        defined        here
**

**********************************************************************
*********/
  input [3:0]  LEs;
  input [15:0] hexs;
  input [3:0]  points;
  input [1:0]  scan;


/*********************************************************************
**********
  **        The        outputs        are        defined        here
**

**********************************************************************
*********/
  output [3:0] AN;
  output [3:0] HEX;
  output      LE;
  output      point;


/*********************************************************************
**********
  **        The        wires        are        defined        here
**

**********************************************************************
*********/
  wire [3:0]  s_logisimBus0;
  wire [1:0]  s_logisimBus1;
  wire [15:0] s_logisimBus19;
  wire [3:0]  s_logisimBus20;
  wire [3:0]  s_logisimBus21;
  wire [3:0]  s_logisimBus22;
```

```
   wire [3:0]  s_logisimBus23;
   wire [3:0]  s_logisimBus3;
   wire [3:0]  s_logisimBus4;
   wire [3:0]  s_logisimBus9;
   wire        s_logisimNet10;
   wire        s_logisimNet11;
   wire        s_logisimNet12;
   wire        s_logisimNet13;
   wire        s_logisimNet14;
   wire        s_logisimNet2;
   wire        s_logisimNet5;
   wire        s_logisimNet6;
   wire        s_logisimNet7;
   wire        s_logisimNet8;


/**********************************************************************
**********
   **     The     module     functionality     is     described     here
**

***********************************************************************
*********/


/**********************************************************************
**********
   **      Here      all      input      connections      are      defined
**

***********************************************************************
*********/
   assign s_logisimBus0[3:0]   = points;
   assign s_logisimBus19[15:0] = hexs;
   assign s_logisimBus1[1:0]   = scan;
   assign s_logisimBus9[3:0]   = LEs;


/**********************************************************************
**********
   **      Here      all      output      connections      are      defined
**

***********************************************************************
*********/
   assign AN    = s_logisimBus3[3:0];
   assign HEX   = s_logisimBus4[3:0];
   assign LE    = s_logisimNet10;
   assign point = s_logisimNet2;


/**********************************************************************
**********
   **     Here     all     in-lined     components     are     defined
**

***********************************************************************
*********/

   // Constant
   assign  s_logisimBus20[3:0]  =  4'hE;

   // Constant
   assign  s_logisimBus21[3:0]  =  4'hD;


   // Constant
   assign  s_logisimBus22[3:0]  =  4'hB;


   // Constant
   assign  s_logisimBus23[3:0]  =  4'h7;



/*********************************************************************
**********
   **       Here       all       sub-circuits       are       defined
**

*********************************************************************
*********/

   Mux4to1   mux_points (.D0(s_logisimBus0[0]),
                     .D1(s_logisimBus0[1]),
                     .D2(s_logisimBus0[2]),
                     .D3(s_logisimBus0[3]),
                     .S(s_logisimBus1[1:0]),
                     .Y(s_logisimNet2));

   Mux4to1   mux_LE (.D0(s_logisimBus9[0]),
                 .D1(s_logisimBus9[1]),
                 .D2(s_logisimBus9[2]),
                 .D3(s_logisimBus9[3]),
                 .S(s_logisimBus1[1:0]),
                 .Y(s_logisimNet10));

   Mux4to1b4   mux_hexs (.D0(s_logisimBus19[3:0]),
                     .D1(s_logisimBus19[7:4]),
                     .D2(s_logisimBus19[11:8]),
                     .D3(s_logisimBus19[15:12]),
                     .S(s_logisimBus1[1:0]),
                     .Y(s_logisimBus4[3:0]));

   Mux4to1b4   mux_AN (.D0(s_logisimBus20[3:0]),
                   .D1(s_logisimBus21[3:0]),
                   .D2(s_logisimBus22[3:0]),
                   .D3(s_logisimBus23[3:0]),
                   .S(s_logisimBus1[1:0]),
                   .Y(s_logisimBus3[3:0]));

endmodule
```

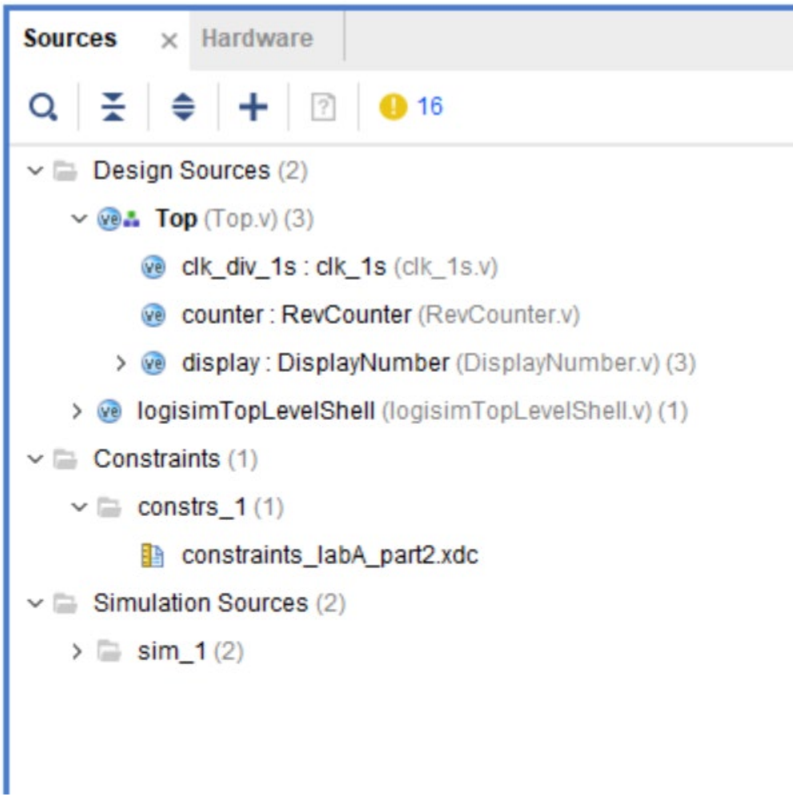
导入的约束文件如下：

```
 # Filename: constraints_labA_part1.xdc
## Constraints file for LabA-part1

# Main clock
set_property PACKAGE_PIN AC18 [get_ports clk]
set_property IOSTANDARD LVCMOS18 [get_ports clk]

create_clock -period 10.000 -name clk [get_ports "clk"]
```

```
set_property PACKAGE_PIN AA10 [get_ports {SW[0]}]
set_property PACKAGE_PIN AB10 [get_ports {SW[1]}]
set_property IOSTANDARD LVCMOS15 [get_ports {SW[0]}]
set_property IOSTANDARD LVCMOS15 [get_ports {SW[1]}]


# LED
set_property PACKAGE_PIN AF24 [get_ports {LED}]
set_property IOSTANDARD LVCMOS33 [get_ports {LED}]

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]}]
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 {AN[0]}]
set_property IOSTANDARD LVCMOS33 [get_ports {AN[1]}]
set_property IOSTANDARD LVCMOS33 [get_ports {AN[2]}]
set_property IOSTANDARD LVCMOS33 [get_ports {AN[3]}]
set_property IOSTANDARD LVCMOS33 [get_ports {SEGMENT[0]}]
set_property IOSTANDARD LVCMOS33 [get_ports {SEGMENT[1]}]
set_property IOSTANDARD LVCMOS33 [get_ports {SEGMENT[2]}]
set_property IOSTANDARD LVCMOS33 [get_ports {SEGMENT[3]}]
set_property IOSTANDARD LVCMOS33 [get_ports {SEGMENT[4]}]
set_property IOSTANDARD LVCMOS33 [get_ports {SEGMENT[5]}]
set_property IOSTANDARD LVCMOS33 [get_ports {SEGMENT[6]}]
set_property IOSTANDARD LVCMOS33 [get_ports {SEGMENT[7]}]


# # Main clock
# set_property PACKAGE_PIN AC18 [get_ports clk_p]
# set_property PACKAGE_PIN AD18 [get_ports clk_n]
# set_property IOSTANDARD LVCMOS18 [get_ports clk_p]
# set_property IOSTANDARD LVCMOS18 [get_ports clk_n]

# # create_clock -period 10.000 -name clk [get_ports "clk_p"]

# # FPGA RST
# set_property PACKAGE_PIN W13 [get_ports RSTN]
# set_property IOSTANDARD LVCMOS18 [get_ports RSTN]

# # 7SEG
# set_property PACKAGE_PIN M24 [get_ports seg_clk]
# set_property PACKAGE_PIN L24 [get_ports set_sout]
# set_property PACKAGE_PIN R18 [get_ports seg_pen]
# set_property PACKAGE_PIN M20 [get_ports seg_clrn]
# set_property IOSTANDARD LVCMOS33 [get_ports seg_clk]
# set_property IOSTANDARD LVCMOS33 [get_ports set_sout]
# set_property IOSTANDARD LVCMOS33 [get_ports seg_pen]
# set_property IOSTANDARD LVCMOS33 [get_ports seg_clrn]

# # Audio out
# set_property PACKAGE_PIN P26 [get_ports AUD_PWM]
# set_property PACKAGE_PIN M25 [get_ports AUD_SD]
# set_property IOSTANDARD LVCMOS33 [get_ports AUD_PWM]
# set_property IOSTANDARD LVCMOS33 [get_ports AUD_SD]

# # Key Array
# set_property PACKAGE_PIN V17 [get_ports BTN_X0]
# set_property IOSTANDARD LVCMOS18 [get_ports BTN_X0]
# set_property PACKAGE_PIN W18 [get_ports BTN_X1]
# set_property IOSTANDARD LVCMOS18 [get_ports BTN_X1]
# set_property PACKAGE_PIN W19 [get_ports BTN_X2]
# set_property IOSTANDARD LVCMOS18 [get_ports BTN_X2]
# set_property PACKAGE_PIN W15 [get_ports BTN_X3]
# set_property IOSTANDARD LVCMOS18 [get_ports BTN_X3]
# set_property PACKAGE_PIN W16 [get_ports BTN_X4]
# set_property IOSTANDARD LVCMOS18 [get_ports BTN_X4]
# set_property PACKAGE_PIN V18 [get_ports BTN_Y0]
# set_property IOSTANDARD LVCMOS18 [get_ports BTN_Y0]
# set_property PACKAGE_PIN V19 [get_ports BTN_Y1]
# set_property IOSTANDARD LVCMOS18 [get_ports BTN_Y1]
# set_property PACKAGE_PIN V14 [get_ports BTN_Y2]
# set_property IOSTANDARD LVCMOS18 [get_ports BTN_Y2]
# set_property PACKAGE_PIN W14 [get_ports BTN_Y3]
# set_property IOSTANDARD LVCMOS18 [get_ports BTN_Y3]

# # Arduino
# set_property PACKAGE_PIN AF25 [get_ports ard_rst]
# set_property IOSTANDARD LVCMOS33 [get_ports ard_rst]
# set_property PACKAGE_PIN AF24 [get_ports {ard_led[0]}]
# set_property PACKAGE_PIN AE21 [get_ports {ard_led[1]}]
# set_property PACKAGE_PIN Y22 [get_ports {ard_led[2]}]
# set_property PACKAGE_PIN Y23 [get_ports {ard_led[3]}]
# set_property PACKAGE_PIN AA23 [get_ports {ard_led[4]}]
# set_property PACKAGE_PIN Y25 [get_ports {ard_led[5]}]
# set_property PACKAGE_PIN AB26 [get_ports {ard_led[6]}]
# set_property PACKAGE_PIN W23 [get_ports {ard_led[7]}]
# set_property IOSTANDARD LVCMOS33 [get_ports {ard_led[0]}]
# set_property IOSTANDARD LVCMOS33 [get_ports {ard_led[1]}]
# set_property IOSTANDARD LVCMOS33 [get_ports {ard_led[2]}]
# set_property IOSTANDARD LVCMOS33 [get_ports {ard_led[3]}]
# set_property IOSTANDARD LVCMOS33 [get_ports {ard_led[4]}]
# set_property IOSTANDARD LVCMOS33 [get_ports {ard_led[5]}]
# set_property IOSTANDARD LVCMOS33 [get_ports {ard_led[6]}]
# set_property IOSTANDARD LVCMOS33 [get_ports {ard_led[7]}]
# set_property PACKAGE_PIN AD21 [get_ports {ard_an[0]}]
# set_property PACKAGE_PIN AC21 [get_ports {ard_an[1]}]
# set_property PACKAGE_PIN AB21 [get_ports {ard_an[2]}]
# set_property PACKAGE_PIN AC22 [get_ports {ard_an[3]}]
# set_property PACKAGE_PIN AB22 [get_ports {ard_seg[0]}]
# set_property PACKAGE_PIN AD24 [get_ports {ard_seg[1]}]
# set_property PACKAGE_PIN AD23 [get_ports {ard_seg[2]}]
# set_property PACKAGE_PIN Y21 [get_ports {ard_seg[3]}]
# set_property PACKAGE_PIN W20 [get_ports {ard_seg[4]}]
# set_property PACKAGE_PIN AC24 [get_ports {ard_seg[5]}]
# set_property PACKAGE_PIN AC23 [get_ports {ard_seg[6]}]
# set_property PACKAGE_PIN AA22 [get_ports {ard_seg[7]}]
# # set_property IOSTANDARD LVCMOS33 [get_ports {ard_dio[13]}]
# # set_property IOSTANDARD LVCMOS33 [get_ports {ard_dio[12]}]
# set_property IOSTANDARD LVCMOS33 [get_ports {ard_an[0]}]
# set_property IOSTANDARD LVCMOS33 [get_ports {ard_an[1]}]
# set_property IOSTANDARD LVCMOS33 [get_ports {ard_an[2]}]
# set_property IOSTANDARD LVCMOS33 [get_ports {ard_an[3]}]
# set_property IOSTANDARD LVCMOS33 [get_ports {ard_seg[0]}]
# set_property IOSTANDARD LVCMOS33 [get_ports {ard_seg[1]}]
# set_property IOSTANDARD LVCMOS33 [get_ports {ard_seg[2]}]
# set_property IOSTANDARD LVCMOS33 [get_ports {ard_seg[3]}]
# set_property IOSTANDARD LVCMOS33 [get_ports {ard_seg[4]}]
# set_property IOSTANDARD LVCMOS33 [get_ports {ard_seg[5]}]
# set_property IOSTANDARD LVCMOS33 [get_ports {ard_seg[6]}]
# set_property IOSTANDARD LVCMOS33 [get_ports {ard_seg[7]}]

# #16leds
# set_property PACKAGE_PIN N26 [get_ports LEDCLK]
# set_property PACKAGE_PIN N24 [get_ports LEDCLR]
# set_property PACKAGE_PIN M26 [get_ports LEDDT]
# set_property IOSTANDARD LVCMOS33 [get_ports LEDCLK]
# set_property IOSTANDARD LVCMOS33 [get_ports LEDCLR]
# set_property IOSTANDARD LVCMOS33 [get_ports LEDDT]

# #16dips
# set_property PACKAGE_PIN AA10 [get_ports {switch[0]}]
# set_property PACKAGE_PIN AB10 [get_ports {switch[1]}]
# set_property PACKAGE_PIN AA13 [get_ports {switch[2]}]
# set_property PACKAGE_PIN AA12 [get_ports {switch[3]}]
# set_property PACKAGE_PIN Y13 [get_ports {switch[4]}]
# set_property PACKAGE_PIN Y12 [get_ports {switch[5]}]
# set_property PACKAGE_PIN AD11 [get_ports {switch[6]}]
# set_property PACKAGE_PIN AD10 [get_ports {switch[7]}]
# set_property PACKAGE_PIN AE10 [get_ports {switch[8]}]
# set_property PACKAGE_PIN AE12 [get_ports {switch[9]}]
# set_property PACKAGE_PIN AF12 [get_ports {switch[10]}]
# set_property PACKAGE_PIN AE8 [get_ports {switch[11]}]
# set_property PACKAGE_PIN AF8 [get_ports {switch[12]}]
# set_property PACKAGE_PIN AE13 [get_ports {switch[13]}]
# set_property PACKAGE_PIN AF13 [get_ports {switch[14]}]
# set_property PACKAGE_PIN AF10 [get_ports {switch[15]}]
# set_property IOSTANDARD LVCMOS15 [get_ports {switch[0]}]
# set_property IOSTANDARD LVCMOS15 [get_ports {switch[1]}]
# set_property IOSTANDARD LVCMOS15 [get_ports {switch[2]}]
# set_property IOSTANDARD LVCMOS15 [get_ports {switch[3]}]
# set_property IOSTANDARD LVCMOS15 [get_ports {switch[4]}]
# set_property IOSTANDARD LVCMOS15 [get_ports {switch[5]}]
# set_property IOSTANDARD LVCMOS15 [get_ports {switch[6]}]
# set_property IOSTANDARD LVCMOS15 [get_ports {switch[7]}]
# set_property IOSTANDARD LVCMOS15 [get_ports {switch[8]}]
# set_property IOSTANDARD LVCMOS15 [get_ports {switch[9]}]
# set_property IOSTANDARD LVCMOS15 [get_ports {switch[10]}]
# set_property IOSTANDARD LVCMOS15 [get_ports {switch[11]}]
# set_property IOSTANDARD LVCMOS15 [get_ports {switch[12]}]
# set_property IOSTANDARD LVCMOS15 [get_ports {switch[13]}]
# set_property IOSTANDARD LVCMOS15 [get_ports {switch[14]}]
# set_property IOSTANDARD LVCMOS15 [get_ports {switch[15]}]

# # VGA
# set_property PACKAGE_PIN N21 [get_ports {vga_red[0]}]
# set_property PACKAGE_PIN N22 [get_ports {vga_red[1]}]
# set_property PACKAGE_PIN R21 [get_ports {vga_red[2]}]
# set_property PACKAGE_PIN P21 [get_ports {vga_red[3]}]
# set_property IOSTANDARD LVCMOS33 [get_ports {vga_red[0]}]
# set_property IOSTANDARD LVCMOS33 [get_ports {vga_red[1]}]
# set_property IOSTANDARD LVCMOS33 [get_ports {vga_red[2]}]
# set_property IOSTANDARD LVCMOS33 [get_ports {vga_red[3]}]
# set_property PACKAGE_PIN R22 [get_ports {vga_green[0]}]
# set_property PACKAGE_PIN R23 [get_ports {vga_green[1]}]
# set_property PACKAGE_PIN T24 [get_ports {vga_green[2]}]
# set_property PACKAGE_PIN T25 [get_ports {vga_green[3]}]
# set_property IOSTANDARD LVCMOS33 [get_ports {vga_green[0]}]
# set_property IOSTANDARD LVCMOS33 [get_ports {vga_green[1]}]
# set_property IOSTANDARD LVCMOS33 [get_ports {vga_green[2]}]
# set_property IOSTANDARD LVCMOS33 [get_ports {vga_green[3]}]
# set_property PACKAGE_PIN T20 [get_ports {vga_blue[0]}]
# set_property PACKAGE_PIN R20 [get_ports {vga_blue[1]}]
# set_property PACKAGE_PIN T22 [get_ports {vga_blue[2]}]
# set_property PACKAGE_PIN T23 [get_ports {vga_blue[3]}]
# set_property IOSTANDARD LVCMOS33 [get_ports {vga_blue[0]}]
# set_property IOSTANDARD LVCMOS33 [get_ports {vga_blue[1]}]
# set_property IOSTANDARD LVCMOS33 [get_ports {vga_blue[2]}]
# set_property IOSTANDARD LVCMOS33 [get_ports {vga_blue[3]}]
# set_property PACKAGE_PIN M22 [get_ports vga_hs]
# set_property PACKAGE_PIN [get_ports vga_vs]
# set_property IOSTANDARD LVCMOS33 [get_ports vga_hs]
# set_property IOSTANDARD LVCMOS33 [get_ports vga_vs]
```

需要注意的是，本次导入了前面实验的文件，需要确保前几次实验的正确性

导入后文件属性如下：



Design Sources (2)
  Top (Top.v) (3)
    clk_div_1s : clk_1s (clk_1s.v)
    counter : RevCounter (RevCounter.v)
    display : DisplayNumber (DisplayNumber.v) (3)
  logisimTopLevelShell (logisimTopLevelShell.v) (1)
Constraints (1)
  constrs_1 (1)
    constraints_labA_part2.xdc
Simulation Sources (2)
  sim_1 (2)

生成 bitstream 并烧录

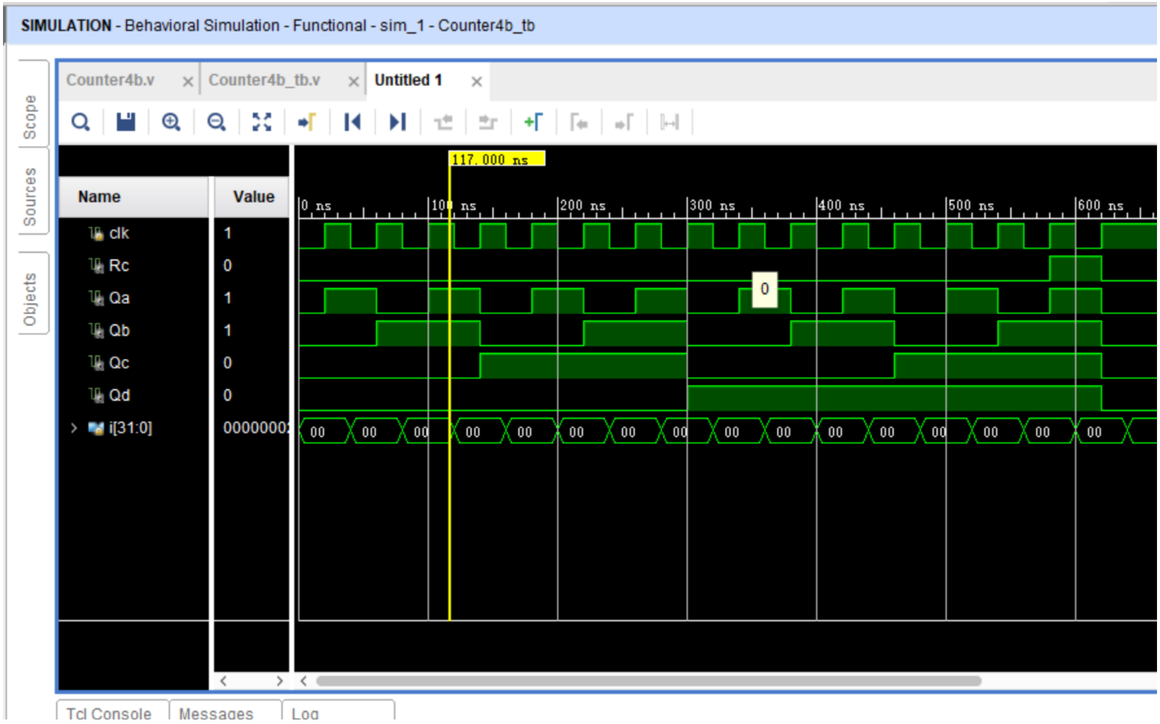
配置完成后，得到 bitstream。

之后将下载器连接到电脑上。点击 PROGRAM AND DEBUG > Open Hardware Manager > Open Target > Auto Connect 进行识别和连接，成功连接后，点击 Program Device 选择 xc7k160t 设备，在下载程序界面选择我们刚刚生成的比特流文件，将其下载到板上。之后在板上实现相关操作。

二、实验结果与分析

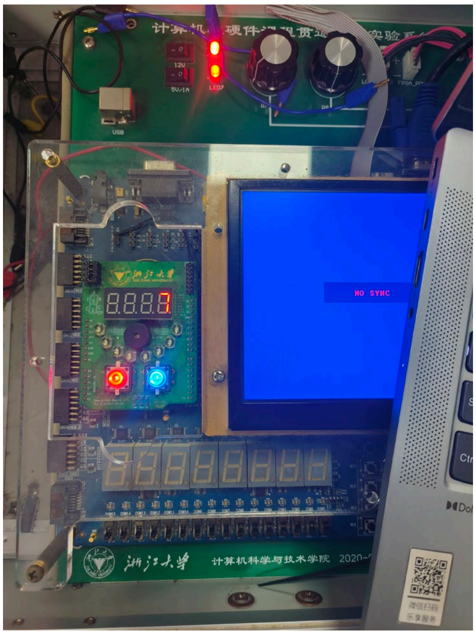
（一）四位同步二进制计数器实验结果

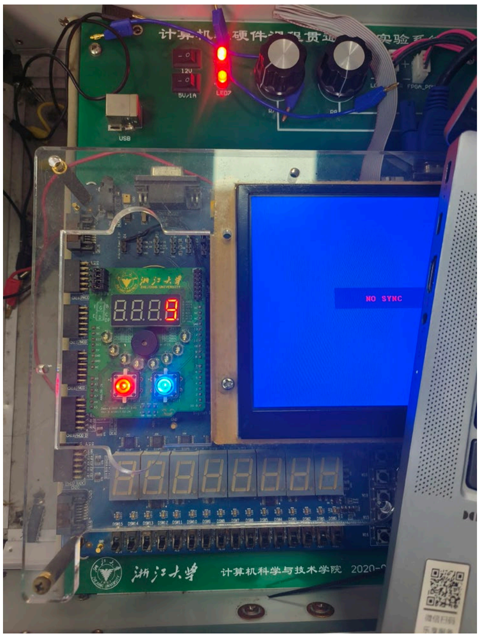
1、仿真结果

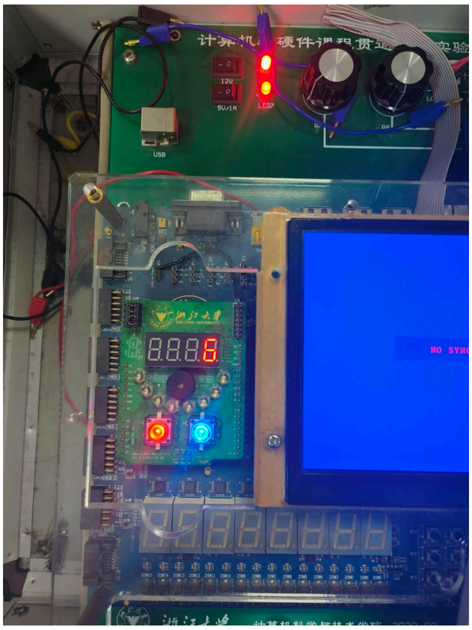


如图，clk 在 0/1 之间反复转换过程中，四位计数器和 Rc 反应正确，说明仿真文件及 design source 均正确

2、上板结果



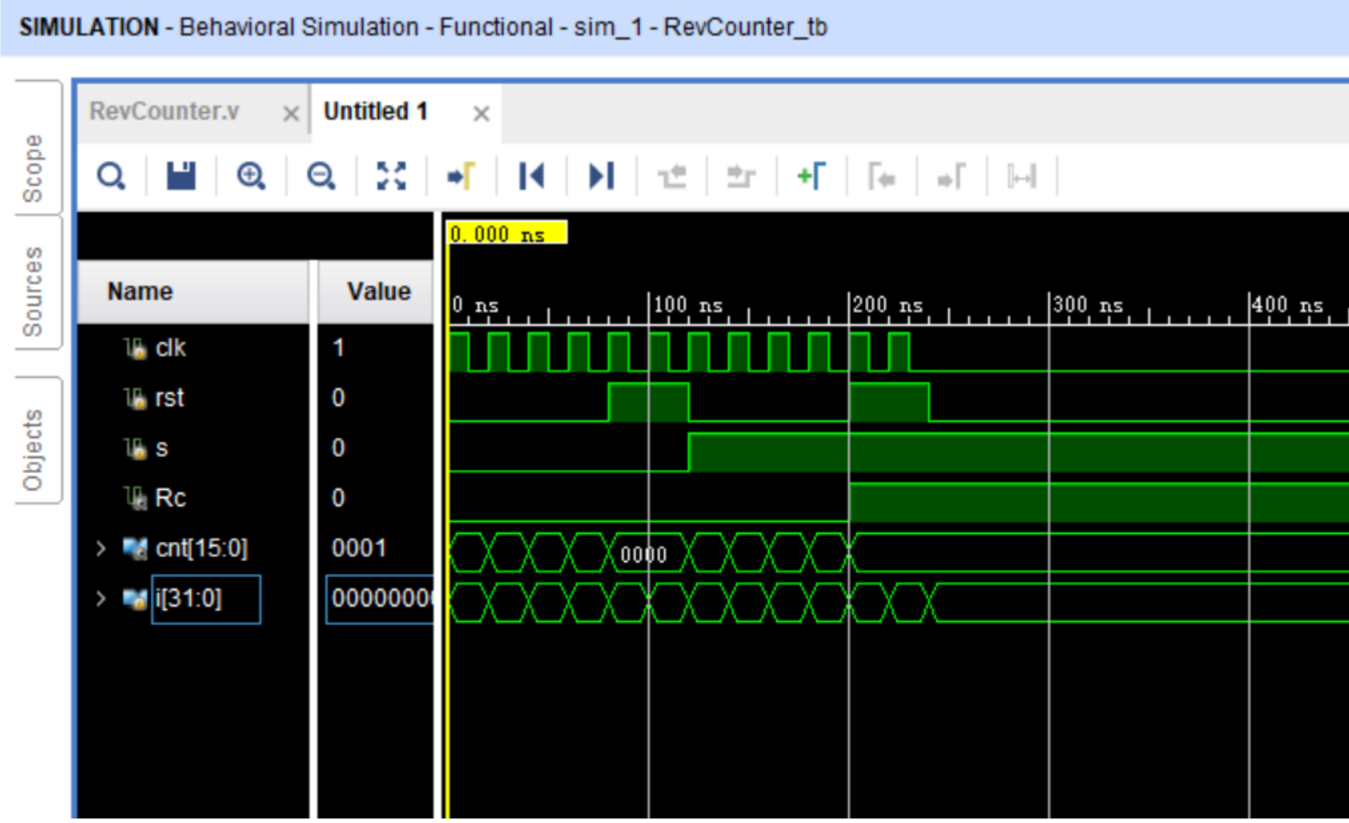




如图所示，正确实现四位同步二进制计数器。

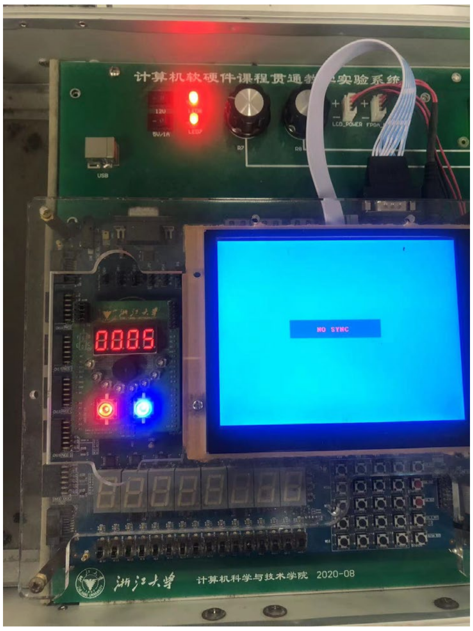
（二）十六位可逆同步二进制计数器实验结果

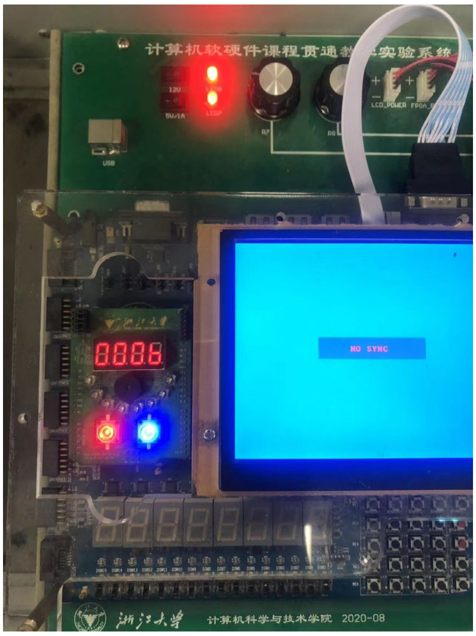
1、仿真结果

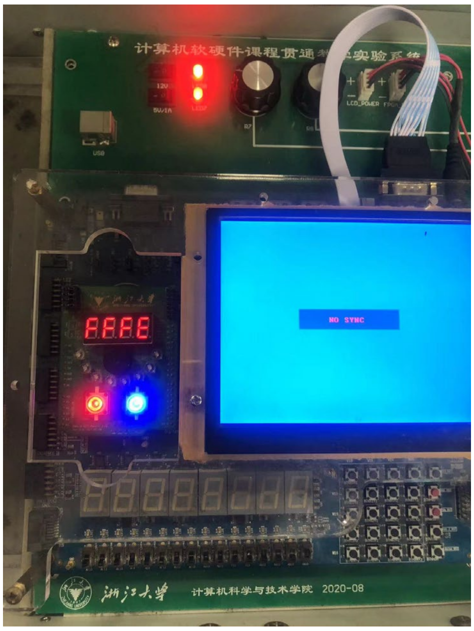


如图所示，正确实现 RevCounter 功能

2、上板结果


计算机软硬件课程贯通教学实验系统
0005
NO SYNC
浙江大学
计算机科学与技术学院 2020-08


计算机软硬件课程贯通教学实验系统
0006
NO SYNC
浙江大学
计算机科学与技术学院 2020-08


计算机软硬件课程贯通教学实验系统
FFFE
NO SYNC
浙江大学
计算机科学与技术学院 2020-08

如图所示，拨动开关，实现自加、自减的控制，四位均能正常显示，同时如果小于 0，会自动变成 FFFE，说明达成了要求的功能。

三、讨论、心得

在本次实验中，我增加了对时序电路、时序逻辑的理解，通过仿真代码，Verilog 代码的撰写，我更加了解如何学习硬件电路相关的知识。同时，我在完成更加复杂的功能之后，更有成就感，这个自加器的功能是很实用的，也许还会对大作业的完成有帮助。

四、个人生活照片