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 10 月 24 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 5：变量译码器设计与运用

学生姓名：蔡佳伟 学号：3220104519 同组学生姓名：

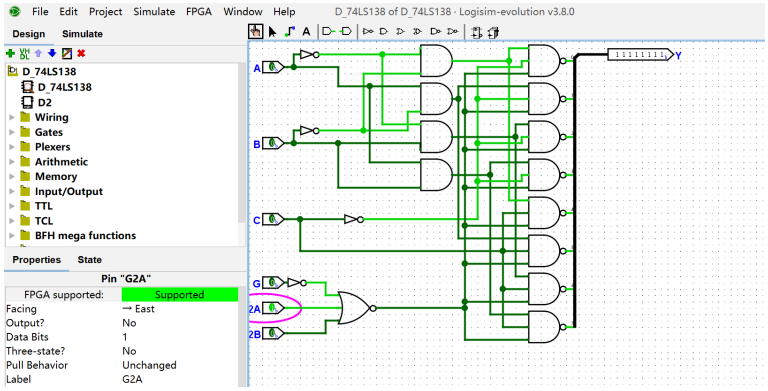
实验地点：紫金港东四 509 室 实验日期：2023 年 10 月 24 日

一、操作方法与实验步骤

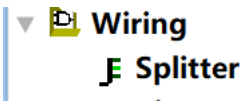
（一）原理图设计实现 74LS138 译码器模块

1、使用 Logisim 绘制译码器

绘制结果如下：

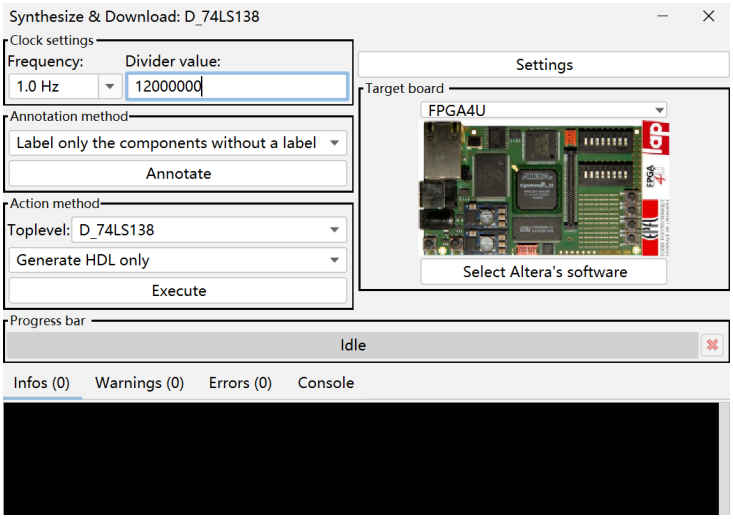


其中选择如下方式绘制 Y

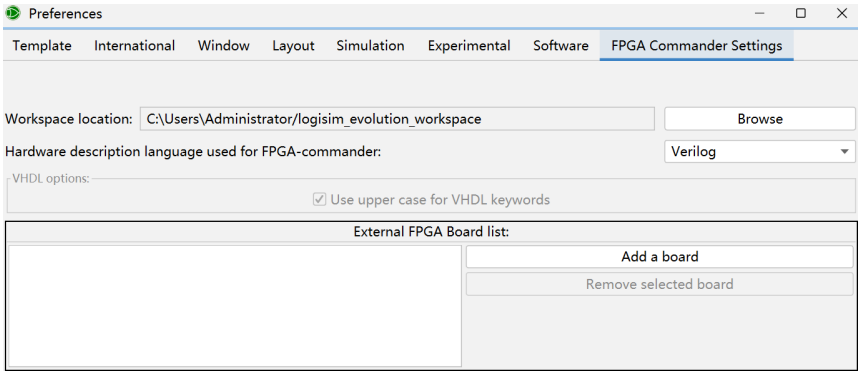


需要更改入口和出口名称，本次按照图片更改为 A、B、C、G、G2A、G2B 和 Y

之后导出电路图为 Verilog 代码



注意 Target board 选择 FPGA4U



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                :                D_74LS138
**
   **                                                                      **

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

  module D_74LS138( A,
                    B,
                    C,
                    G,
                    G2A,
                    G2B,
                    Y );


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

```
    **        The        inputs        are        defined        here
**

*************************************************************************
**********/
    input A;
    input B;
    input C;
    input G;
    input G2A;
    input G2B;


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

*************************************************************************
**********/
    output [7:0] Y;


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

*************************************************************************
**********/
    wire [7:0] s_logisimBus11;
    wire      s_logisimNet0;
    wire      s_logisimNet1;
    wire      s_logisimNet10;
    wire      s_logisimNet12;
    wire      s_logisimNet13;
    wire      s_logisimNet14;
    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_logisimNet23;
    wire      s_logisimNet3;
    wire      s_logisimNet4;
    wire      s_logisimNet5;
    wire      s_logisimNet6;
    wire      s_logisimNet7;
    wire      s_logisimNet8;
    wire      s_logisimNet9;


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

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


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

*****************************************************************
**********/
    assign s_logisimNet0  = C;
    assign s_logisimNet14 = G2A;
    assign s_logisimNet23 = G;
    assign s_logisimNet3  = G2B;
    assign s_logisimNet7  = B;
    assign s_logisimNet9  = A;


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

*****************************************************************
**********/
    assign Y = s_logisimBus11[7:0];


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

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

    // NOT Gate
    assign s_logisimNet4 = ~s_logisimNet7;

    // NOT Gate
    assign s_logisimNet8 = ~s_logisimNet9;

    // NOT Gate
    assign s_logisimNet1 = ~s_logisimNet0;

    // NOT Gate
    assign s_logisimNet12 = ~s_logisimNet23;


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

*****************************************************************
**********/
    NOR_GATE_3_INPUTS #(.BubblesMask(3'b000))
       GATES_1 (.input1(s_logisimNet12),
                .input2(s_logisimNet14),
                .input3(s_logisimNet3),
```

```
            .result(s_logisimNet6));

    AND_GATE #(.BubblesMask(2'b00))
       GATES_2 (.input1(s_logisimNet8),
                .input2(s_logisimNet7),
                .result(s_logisimNet5));

    AND_GATE #(.BubblesMask(2'b00))
       GATES_3 (.input1(s_logisimNet9),
                .input2(s_logisimNet7),
                .result(s_logisimNet10));

    AND_GATE #(.BubblesMask(2'b00))
       GATES_4 (.input1(s_logisimNet8),
                .input2(s_logisimNet4),
                .result(s_logisimNet2));

    AND_GATE #(.BubblesMask(2'b00))
       GATES_5 (.input1(s_logisimNet9),
                .input2(s_logisimNet4),
                .result(s_logisimNet13));

    NAND_GATE_3_INPUTS #(.BubblesMask(3'b000))
       GATES_6 (.input1(s_logisimNet5),
                .input2(s_logisimNet1),
                .input3(s_logisimNet6),
                .result(s_logisimBus11[2]));

    NAND_GATE_3_INPUTS #(.BubblesMask(3'b000))
       GATES_7 (.input1(s_logisimNet10),
                .input2(s_logisimNet1),
                .input3(s_logisimNet6),
                .result(s_logisimBus11[3]));

    NAND_GATE_3_INPUTS #(.BubblesMask(3'b000))
       GATES_8 (.input1(s_logisimNet2),
                .input2(s_logisimNet0),
                .input3(s_logisimNet6),
                .result(s_logisimBus11[4]));

    NAND_GATE_3_INPUTS #(.BubblesMask(3'b000))
       GATES_9 (.input1(s_logisimNet2),
                .input2(s_logisimNet1),
                .input3(s_logisimNet6),
                .result(s_logisimBus11[0]));

    NAND_GATE_3_INPUTS #(.BubblesMask(3'b000))
       GATES_10 (.input1(s_logisimNet13),
                 .input2(s_logisimNet0),
                 .input3(s_logisimNet6),
                 .result(s_logisimBus11[5]));

    NAND_GATE_3_INPUTS #(.BubblesMask(3'b000))
       GATES_11 (.input1(s_logisimNet5),
                 .input2(s_logisimNet0),
                 .input3(s_logisimNet6),
                 .result(s_logisimBus11[6]));

    NAND_GATE_3_INPUTS #(.BubblesMask(3'b000))
       GATES_12 (.input1(s_logisimNet10),
                 .input2(s_logisimNet0),
```

```
                .input3(s_logisimNet6),
                .result(s_logisimBus11[7]));

    NAND_GATE_3_INPUTS #(.BubblesMask(3'b000))
       GATES_13 (.input1(s_logisimNet13),
                .input2(s_logisimNet1),
                .input3(s_logisimNet6),
                .result(s_logisimBus11[1]));


  endmodule
```

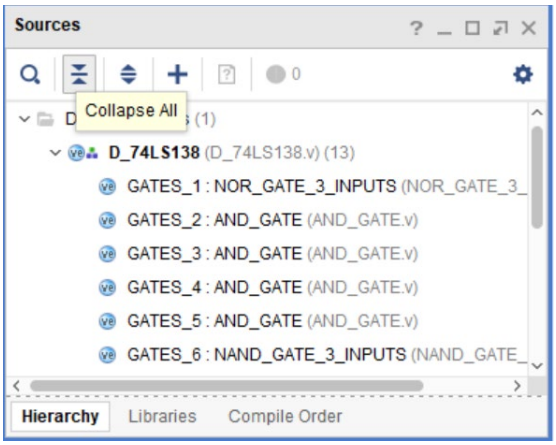
2、使用 Vivado 对电路生成的 Verilog 代码进行仿真

（1）新建工程，此处命名为 project_2

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。



导入后状态如下

（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。

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 D_74LS138_tb();

// Inputs
   reg G;
   reg G2A;
   reg G2B;
   reg C;
   reg A;
   reg B;

// Output
   wire [7:0] Y;

   D_74LS138 m0 (
    .Y(Y),
    .G(G),
    .G2A(G2A),
    .G2B(G2B),
    .A(A),
    .B(B),
    .C(C)
   );
// Initialize Inputs
  initial begin
    C = 0; B = 0; A = 0;
    G = 1; G2A = 0; G2B = 0; #50;
    {C, B, A} = 3'b000;

    repeat(8) begin
      {C, B, A} = {C, B, A} + 3'b1; #50;
    end

    G=1'b0; G2A=1'b0; G2B=1'b0; #50;
    G=1'b1; G2A=1'b1; G2B=1'b0; #50;
    G=1'b1; G2A=1'b0; G2B=1'b1; #50;
  end
endmodule // D_74LS138_tb
```

3、使用 Vivado 综合并上板验证

（1）添加约束文件并修改

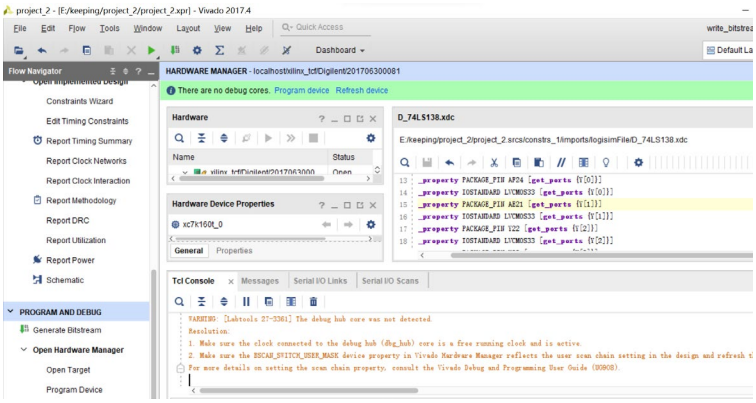
选择 Add or Create Constraints 将约束文件添加进入工程，并进行修改，设置引脚约束。修改后代码如下：

```
        s et_property PACKAGE_PIN AA10 [get_ports {A}]
        set_property IOSTANDARD LVCMOS15 [get_ports {A}]
        set_property PACKAGE_PIN AB10 [get_ports {B}]
        set_property IOSTANDARD LVCMOS15 [get_ports {B}]
        set_property PACKAGE_PIN AA13 [get_ports {C}]
        set_property IOSTANDARD LVCMOS15 [get_ports {C}]
        set_property PACKAGE_PIN AA12 [get_ports {G}]
        set_property IOSTANDARD LVCMOS15 [get_ports {G}]
        set_property PACKAGE_PIN Y13 [get_ports {G2A}]
        set_property IOSTANDARD LVCMOS15 [get_ports {G2A}]
        set_property PACKAGE_PIN Y12 [get_ports {G2B}]
        set_property IOSTANDARD LVCMOS15 [get_ports {G2B}]
        set_property PACKAGE_PIN AF24 [get_ports {Y[0]}]
        set_property IOSTANDARD LVCMOS33 [get_ports {Y[0]}]
        set_property PACKAGE_PIN AE21 [get_ports {Y[1]}]
        set_property IOSTANDARD LVCMOS33 [get_ports {Y[1]}]
        set_property PACKAGE_PIN Y22 [get_ports {Y[2]}]
```

```
set_property IOSTANDARD LVCMOS33 [get_ports {Y[2]}]
set_property PACKAGE_PIN Y23 [get_ports {Y[3]}]
set_property IOSTANDARD LVCMOS33 [get_ports {Y[3]}]
set_property PACKAGE_PIN AA23 [get_ports {Y[4]}]
set_property IOSTANDARD LVCMOS33 [get_ports {Y[4]}]
set_property PACKAGE_PIN Y25 [get_ports {Y[5]}]
set_property IOSTANDARD LVCMOS33 [get_ports {Y[5]}]
set_property PACKAGE_PIN AB26 [get_ports {Y[6]}]
set_property IOSTANDARD LVCMOS33 [get_ports {Y[6]}]
set_property PACKAGE_PIN W23 [get_ports {Y[7]}]
set_property IOSTANDARD LVCMOS33 [get_ports {Y[7]}]
```

（2）生成 bitstream 并烧录

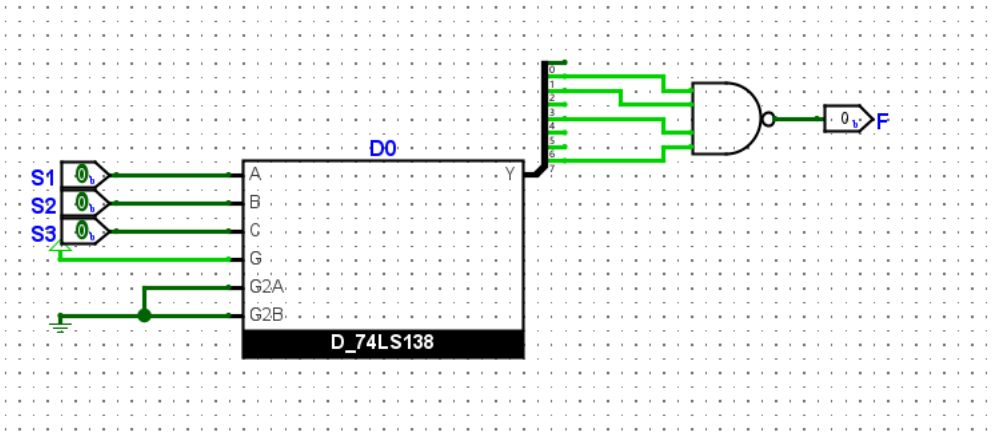
配置完成后，得到 bitstream。



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）用 74LS138 译码器实现楼道灯的控制

1．使用 Logisim 绘制原理图



绘制如上，其中，中间部分芯片如上图所示。

当一个工程中有多个电路时，我们可以在其中一个电路中使用另一个电路。**双击**进入我们希望编辑的电路，**单击**选择另一个电路，即可像使用库中的逻辑门一样使用这个电路组成新的逻辑。需要注意的是，如果你希望导出 Verilog，也需要为引入的模块进行命名。比如在完成了 D_74LS138 模块后，将其引入到新的电路中，并为其命名 D0

2．导出为 Verilog 代码

生成的 Verilog 代码如下：

```
    /*********************************************************************
*****************
    ** Logisim-evolution goes FPGA automatic generated Verilog code
**
    **                          https://github.com/logisim-evolution/
**
    **
**
    **                 Component                 :                 D2
**
    **
**

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

   module D2( F,
              S1,
              S2,
              S3 );


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

*********************************************************************
**************/
      input S1;
      input S2;
      input S3;


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

*********************************************************************
**************/
      output F;


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

*********************************************************************
**************/
      wire [7:0] s_logisimBus10;
      wire      s_logisimNet0;
      wire      s_logisimNet1;
      wire      s_logisimNet2;
      wire      s_logisimNet3;
      wire      s_logisimNet4;
```

```
      wire      s_logisimNet5;
      wire      s_logisimNet6;
      wire      s_logisimNet7;
      wire      s_logisimNet8;
      wire      s_logisimNet9;


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

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


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

*****************************************************************
**************/
      assign s_logisimNet7 = S3;
      assign s_logisimNet8 = S2;
      assign s_logisimNet9 = S1;


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

*****************************************************************
**************/
      assign F = s_logisimNet3;


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

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

      // Ground
      assign  s_logisimNet0  =  1'b0;


      // Power
      assign  s_logisimNet1  =  1'b1;



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

*****************************************************************
```

```
    **************/
          NAND_GATE_4_INPUTS #(.BubblesMask(4'h0))
             GATES_1 (.input1(s_logisimBus10[1]),
                      .input2(s_logisimBus10[2]),
                      .input3(s_logisimBus10[4]),
                      .input4(s_logisimBus10[7]),
                      .result(s_logisimNet3));



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

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

          D_74LS138   D0 (.A(s_logisimNet9),
                        .B(s_logisimNet8),
                        .C(s_logisimNet7),
                        .G(s_logisimNet1),
                        .G2A(s_logisimNet0),
                        .G2B(s_logisimNet0),
                        .Y(s_logisimBus10[7:0]));

       endmodule
```

3．导入约束文件

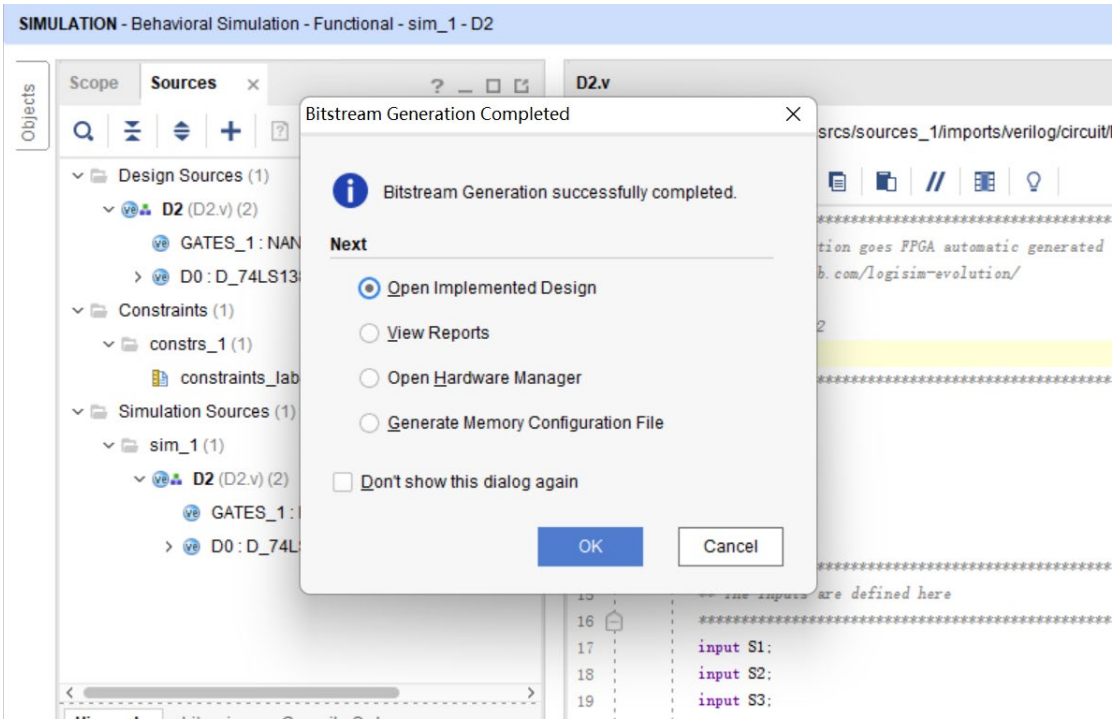
此次导入的约束文件和 lab4 相同

代码如下：

```
# Filename: constraints_lab4.xdc
## Constraints file for Lab4

# Input from switches
set_property PACKAGE_PIN AA10 [get_ports {S1}]
set_property IOSTANDARD LVCMOS15 [get_ports {S1}]
set_property PACKAGE_PIN AB10 [get_ports {S2}]
set_property IOSTANDARD LVCMOS15 [get_ports {S2}]
set_property PACKAGE_PIN AA13 [get_ports {S3}]
set_property IOSTANDARD LVCMOS15 [get_ports {S3}]

# Present output on LED of arduino
set_property PACKAGE_PIN AF24 [get_ports {F}]
set_property IOSTANDARD LVCMOS33 [get_ports {F}]
```

4．生成 bitstream 并烧录

操作同上，点击 PROGRAM AND DEBUG > Generate Bitstream **生成比特流，**生成比特流的结果将通过弹窗方式提示，如果生成失败请查看日志文件确定失败的原因。如生成成功，点击 cancel 即可。

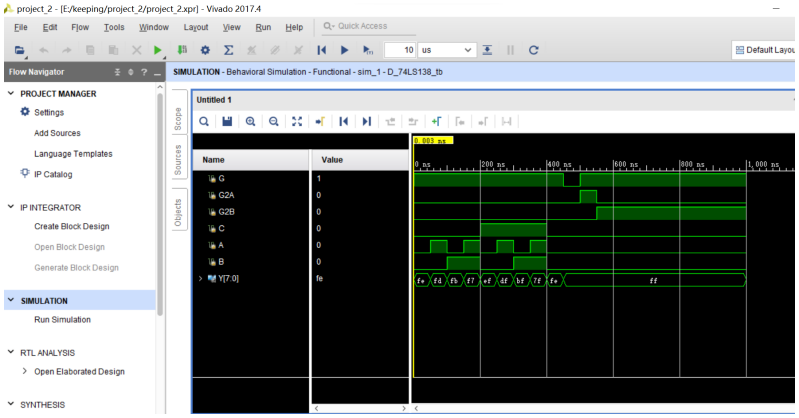


得到 bitstream 后，我们需要将下载器连接到电脑上，点击 PROGRAM AND DEBUG > Open Hardware Manager > Open Target > Auto Connect 进行识别和连接，成功连接后，点击 Program Device 选择 xc7k160t 设备，在下载程序界面选择我们刚刚生成的比特流文件，将其下载到板上。

二、实验结果与分析

（一）74LS138 译码器的设计

1、译码器的仿真结果

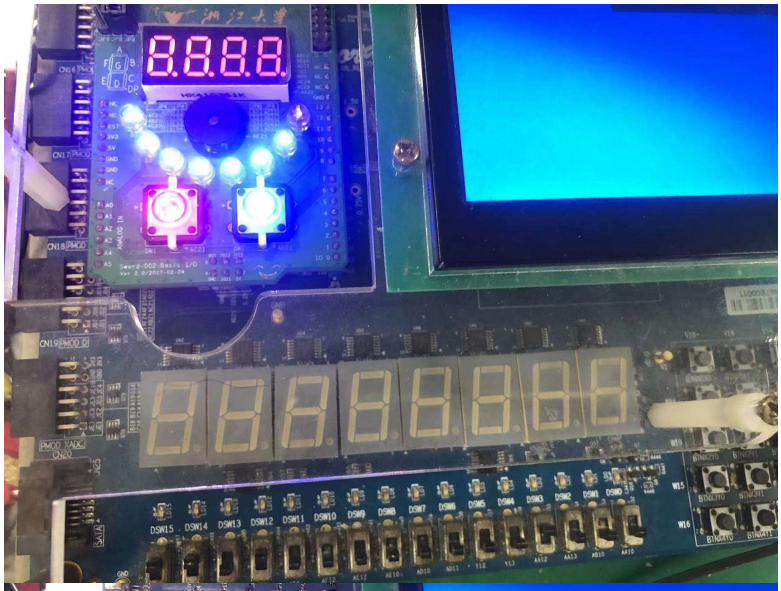


如图，A、B、C 及 G2A、G2B、G 开关的状态对 LED 输出的状态控制正确，说明译码功能成功实现。

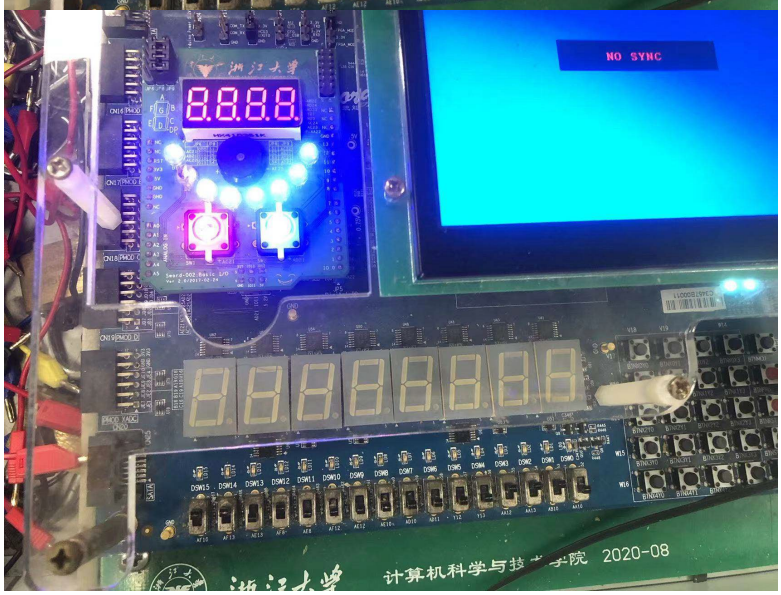
（二）74LS138 译码器上板验证结果

把 G 置为 1，G 置为 0 时无变化

S1、S2、S3	结果
000	0 灭，其他灯亮
001	1 灭，其他灯亮
010	2 灭，其他灯亮
011	3 灭，其他灯亮
100	4 灭，其他灯亮
101	5 灭，其他灯亮
110	6 灭，其他灯亮
111	7 灭，其他灯亮



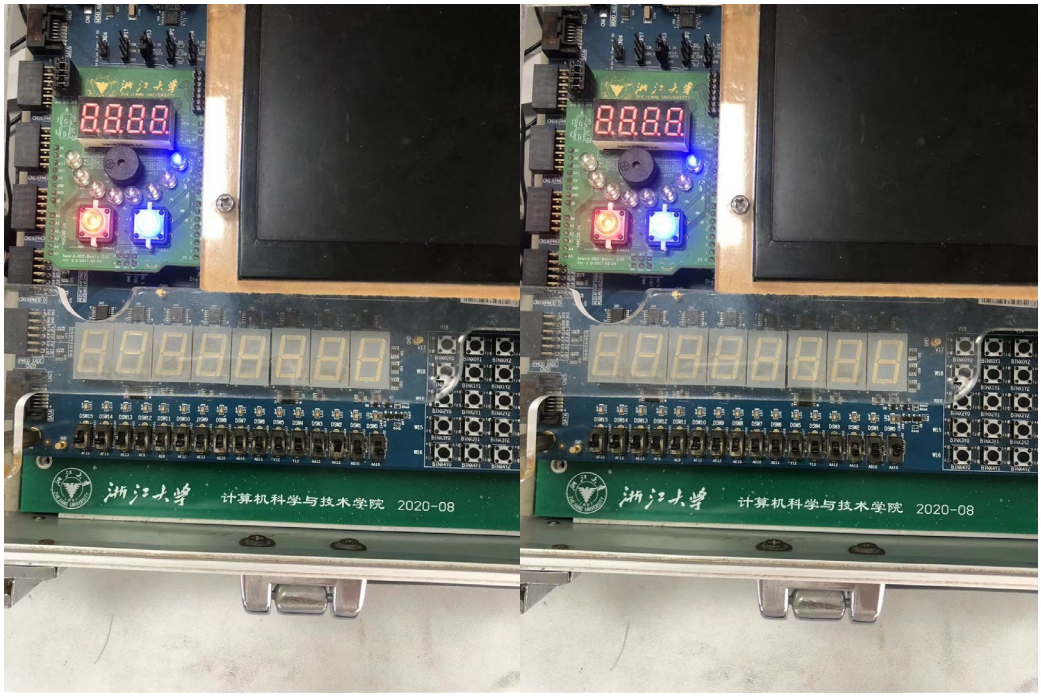
其中 000 和 110 两种情况如图所示，说明实验成功



（三）楼道灯功能的实现

S3	S2	S1	F
0	0	0	0
0	0	1	1
0	1	1	0
0	1	0	1
1	1	0	0
1	0	0	1
1	0	1	0
1	1	1	1

以上是楼道灯的真值表（以格雷码顺序排列）。可见，按动任何开关一次，灯泡亮暗改变。说明成功达成功能。

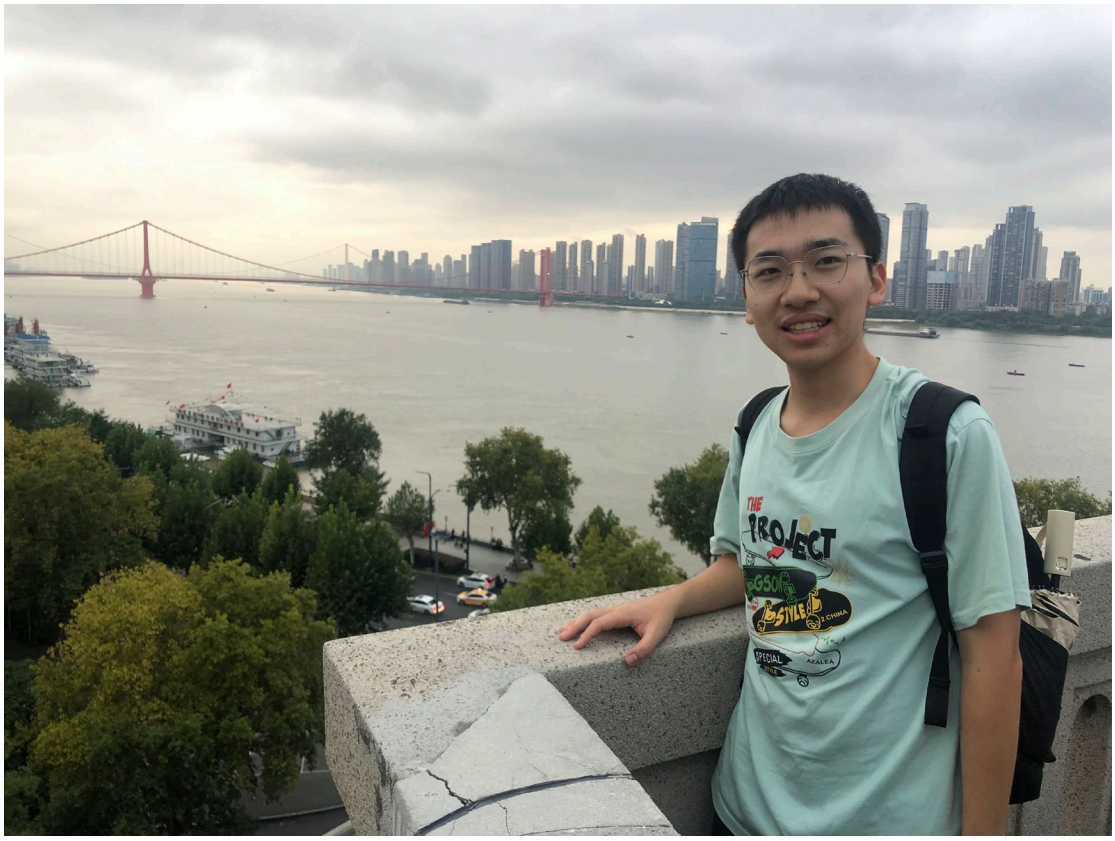


以上开关楼道灯均亮，说明控制楼道灯功能成功实现。

三、讨论、心得

在本次实验中，我继续学习了利用 Logisim 绘图和利用 Vivado 进行硬件电路的设计。画图中，我学会了导出电路为芯片，使用分流器等等。在 Vivado 中，我更加熟练了导入仿真文件，约束文件，进行仿真等功能，了解了数字电路的神奇。在过程中虽然有所波折，但总体来看我学会了很多知识。

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：	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 10 月 31 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 6：七段数码管显示译码器设计与应用

学生姓名：蔡佳伟　　学号：3220104519　　同组学生姓名：

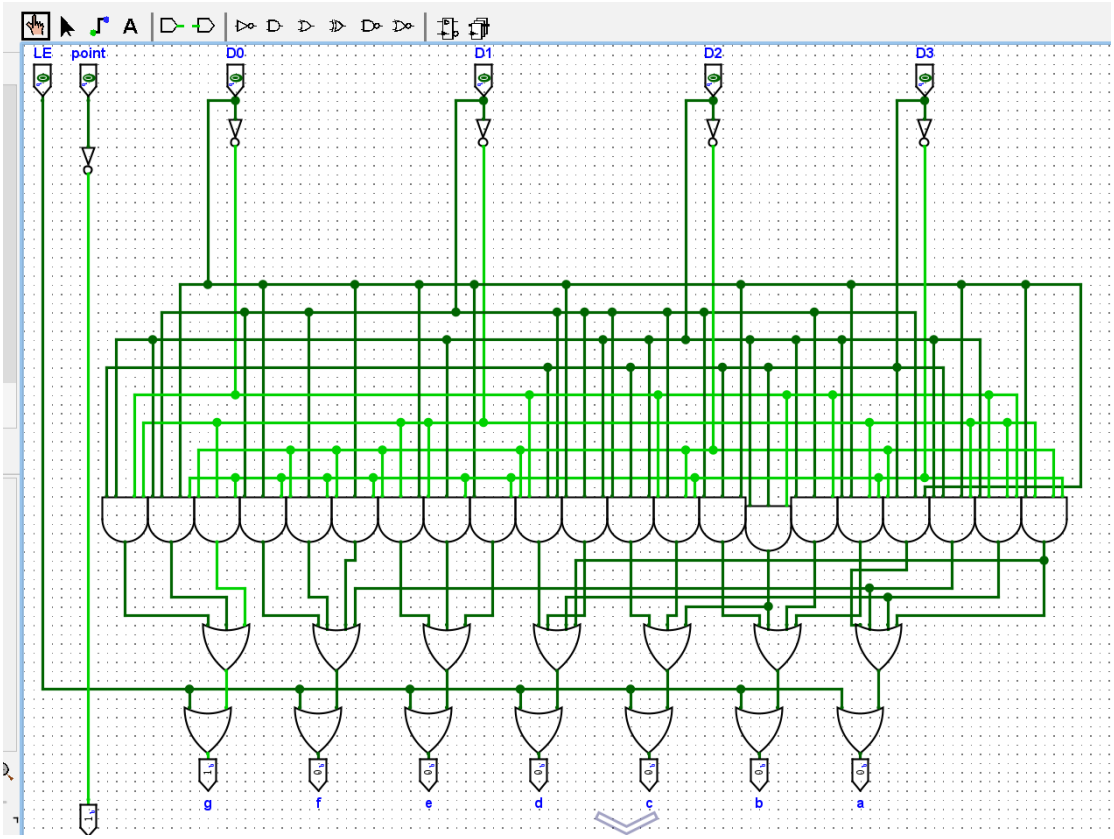
实验地点：紫金港东四 509 室　　实验日期：2023 年 10 月 31 日

一、操作方法与实验步骤

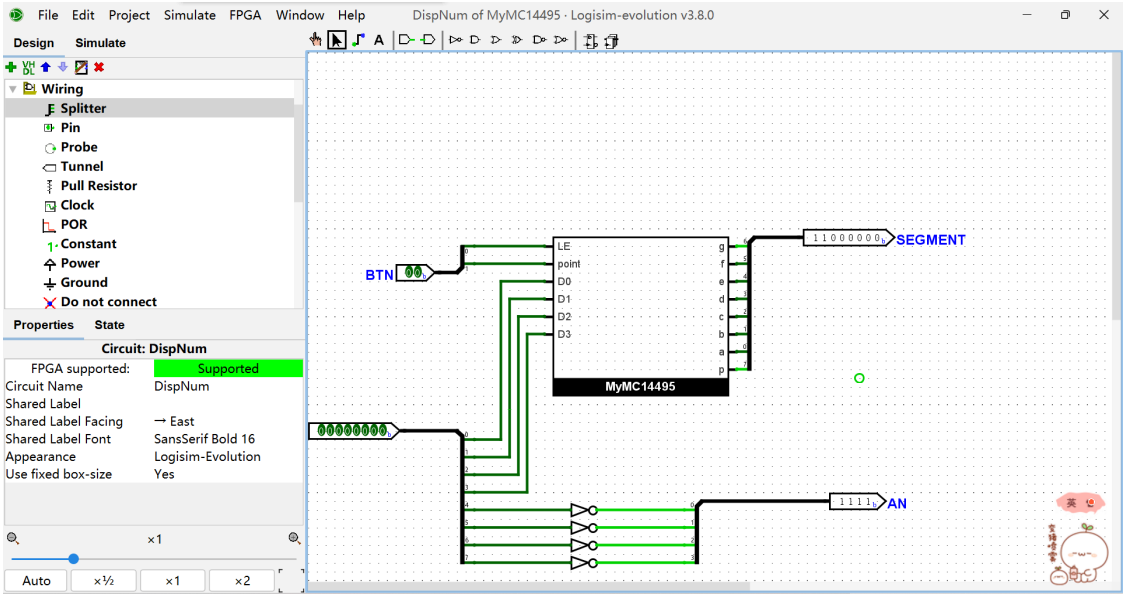
（一）原理图设计实现译码 MyMC14495 模块

1、使用 Logisim 绘制芯片逻辑电路图

绘制结果如下：

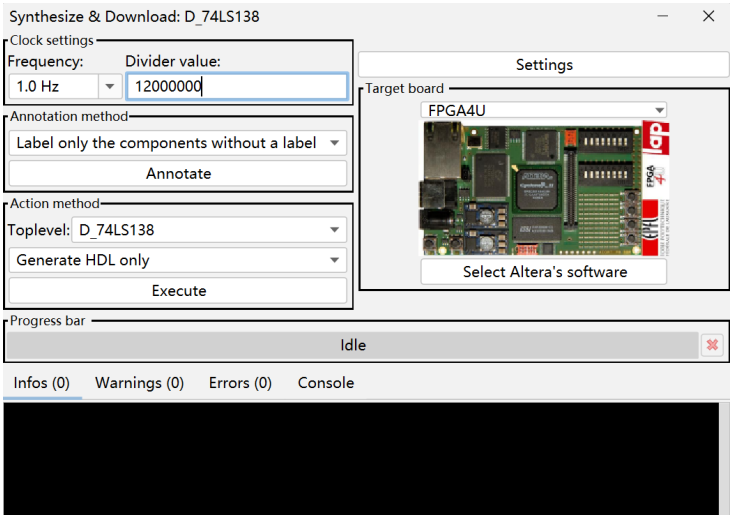


绘制完成后，绘制本届电路 DispNum，绘制如下：

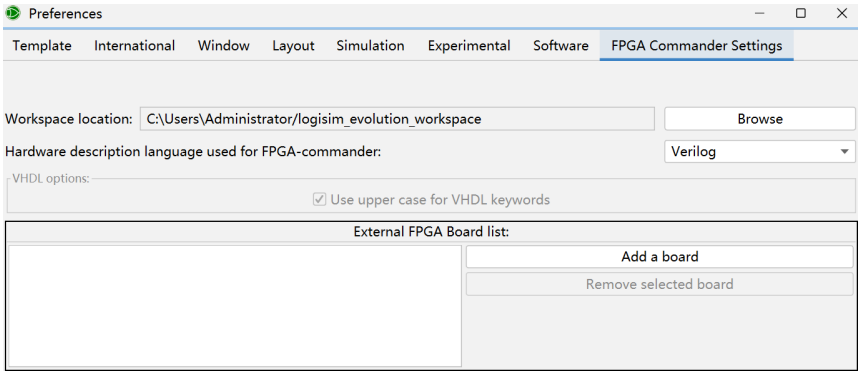


需要注意的是，两次绘制完成后，需要分别更改出口、入口的名称。同时注意分线器下标与连接端口的对应关系。

之后导出电路图为 Verilog 代码



注意 Target board 选择 FPGA4U



注意在 preferences 中把代码格式选为 Verilog

之后 Execute 生成代码，保存

MyMC14485 的 Verilog 代码如下：

```
  /*************************************************************************
************
   **  Logisim-evolution  goes  FPGA  automatic  generated  Verilog  code
**
   **                              https://github.com/logisim-evolution/
**
   **                                                                      **
   **                 Component                :                MyMC14495
**
   **                                                                      **

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

  module MyMC14495( D0,
                    D1,
                    D2,
                    D3,
                    LE,
                    a,
                    b,
                    c,
                    d,
                    e,
                    f,
```

```
                g,
                p,
                point );


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

*************************************************************************
**********/
    input D0;
    input D1;
    input D2;
    input D3;
    input LE;
    input point;


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

*************************************************************************
**********/
    output a;
    output b;
    output c;
    output d;
    output e;
    output f;
    output g;
    output p;


/*************************************************************************
***********
    **        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_logisimNet19;
    wire s_logisimNet2;
    wire s_logisimNet20;
    wire s_logisimNet21;
    wire s_logisimNet22;
    wire s_logisimNet23;
```

```
    wire s_logisimNet24;
    wire s_logisimNet25;
    wire s_logisimNet26;
    wire s_logisimNet27;
    wire s_logisimNet28;
    wire s_logisimNet29;
    wire s_logisimNet3;
    wire s_logisimNet30;
    wire s_logisimNet31;
    wire s_logisimNet32;
    wire s_logisimNet33;
    wire s_logisimNet34;
    wire s logisimNet35;
    wire s_logisimNet36;
    wire s_logisimNet37;
    wire s_logisimNet38;
    wire s_logisimNet39;
    wire s_logisimNet4;
    wire s_logisimNet40;
    wire s_logisimNet41;
    wire s_logisimNet42;
    wire s_logisimNet43;
    wire s_logisimNet44;
    wire s_logisimNet45;
    wire s_logisimNet5;
    wire s_logisimNet6;
    wire s_logisimNet7;
    wire s_logisimNet8;
    wire s_logisimNet9;


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

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


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

*********************************************************************
**********/
    assign s_logisimNet11 = D1;
    assign s_logisimNet12 = D2;
    assign s_logisimNet2  = D0;
    assign s_logisimNet21 = D3;
    assign s_logisimNet32 = point;
    assign s_logisimNet7  = LE;


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

*********************************************************************
```

```
**********/
    assign a = s_logisimNet45;
    assign b = s_logisimNet44;
    assign c = s_logisimNet43;
    assign d = s_logisimNet42;
    assign e = s_logisimNet41;
    assign f = s_logisimNet40;
    assign g = s_logisimNet39;
    assign p = s_logisimNet34;


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

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

    // NOT Gate
    assign s_logisimNet5 = ~s_logisimNet2;

    // NOT Gate
    assign s_logisimNet0 = ~s_logisimNet11;

    // NOT Gate
    assign s_logisimNet34 = ~s_logisimNet32;

    // NOT Gate
    assign s_logisimNet18 = ~s_logisimNet12;

    // NOT Gate
    assign s_logisimNet6 = ~s_logisimNet21;


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

*********************************************************************
**********/
    AND_GATE_4_INPUTS #(.BubblesMask(4'h0))
      GATES_1 (.input1(s_logisimNet12),
               .input2(s_logisimNet21),
               .input3(s_logisimNet2),
               .input4(s_logisimNet0),
               .result(s_logisimNet25));

    AND_GATE_4_INPUTS #(.BubblesMask(4'h0))
      GATES_2 (.input1(s_logisimNet12),
               .input2(s_logisimNet5),
               .input3(s_logisimNet0),
               .input4(s_logisimNet5),
               .result(s_logisimNet3));

    AND_GATE_4_INPUTS #(.BubblesMask(4'h0))
      GATES_3 (.input1(s_logisimNet21),
               .input2(s_logisimNet12),
               .input3(s_logisimNet5),
               .input4(s_logisimNet0),
```

```
            .result(s_logisimNet8));

    AND_GATE_4_INPUTS #(.BubblesMask(4'h0))
       GATES_4 (.input1(s_logisimNet2),
               .input2(s_logisimNet0),
               .input3(s_logisimNet18),
               .input4(s_logisimNet6),
               .result(s_logisimNet14));

    AND_GATE_4_INPUTS #(.BubblesMask(4'h0))
       GATES_5 (.input1(s_logisimNet12),
               .input2(s_logisimNet11),
               .input3(s_logisimNet2),
               .input4(s_logisimNet6),
               .result(s_logisimNet17));

    OR_GATE #(.BubblesMask(2'b00))
       GATES_6 (.input1(s_logisimNet7),
               .input2(s_logisimNet36),
               .result(s_logisimNet39));

    AND_GATE_3_INPUTS #(.BubblesMask(3'b000))
       GATES_7 (.input1(s_logisimNet18),
               .input2(s_logisimNet0),
               .input3(s_logisimNet6),
               .result(s_logisimNet33));

    OR_GATE_3_INPUTS #(.BubblesMask(3'b000))
       GATES_8 (.input1(s_logisimNet8),
               .input2(s_logisimNet17),
               .input3(s_logisimNet33),
               .result(s_logisimNet36));

    AND_GATE_3_INPUTS #(.BubblesMask(3'b000))
       GATES_9 (.input1(s_logisimNet11),
               .input2(s_logisimNet2),
               .input3(s_logisimNet6),
               .result(s_logisimNet15));

    AND_GATE_3_INPUTS #(.BubblesMask(3'b000))
       GATES_10 (.input1(s_logisimNet18),
                .input2(s_logisimNet11),
                .input3(s_logisimNet6),
                .result(s_logisimNet28));

    OR_GATE #(.BubblesMask(2'b00))
       GATES_11 (.input1(s_logisimNet7),
                .input2(s_logisimNet29),
                .result(s_logisimNet40));

    OR_GATE_4_INPUTS #(.BubblesMask(4'h0))
       GATES_12 (.input1(s_logisimNet15),
                .input2(s_logisimNet28),
                .input3(s_logisimNet23),
                .input4(s_logisimNet25),
                .result(s_logisimNet29));

    AND_GATE_3_INPUTS #(.BubblesMask(3'b000))
       GATES_13 (.input1(s_logisimNet18),
                .input2(s_logisimNet2),
                .input3(s_logisimNet6),
```

```
            .result(s_logisimNet23));

    AND_GATE_3_INPUTS #(.BubblesMask(3'b000))
       GATES_14 (.input1(s_logisimNet18),
                 .input2(s_logisimNet0),
                 .input3(s_logisimNet2),
                 .result(s_logisimNet24));

    OR_GATE #(.BubblesMask(2'b00))
       GATES_15 (.input1(s_logisimNet7),
                 .input2(s_logisimNet37),
                 .result(s_logisimNet41));

    AND_GATE_3_INPUTS #(.BubblesMask(3'b000))
       GATES_16 (.input1(s_logisimNet0),
                 .input2(s_logisimNet12),
                 .input3(s_logisimNet6),
                 .result(s_logisimNet19));

    OR_GATE_3_INPUTS #(.BubblesMask(3'b000))
       GATES_17 (.input1(s_logisimNet24),
                 .input2(s_logisimNet19),
                 .input3(s_logisimNet9),
                 .result(s_logisimNet37));

    AND_GATE #(.BubblesMask(2'b00))
       GATES_18 (.input1(s_logisimNet2),
                 .input2(s_logisimNet6),
                 .result(s_logisimNet9));

    AND_GATE_4_INPUTS #(.BubblesMask(4'h0))
       GATES_19 (.input1(s_logisimNet18),
                 .input2(s_logisimNet5),
                 .input3(s_logisimNet21),
                 .input4(s_logisimNet11),
                 .result(s_logisimNet38));

    OR_GATE #(.BubblesMask(2'b00))
       GATES_20 (.input1(s_logisimNet7),
                 .input2(s_logisimNet26),
                 .result(s_logisimNet42));

    OR_GATE_4_INPUTS #(.BubblesMask(4'h0))
       GATES_21 (.input1(s_logisimNet38),
                 .input2(s_logisimNet16),
                 .input3(s_logisimNet3),
                 .input4(s_logisimNet14),
                 .result(s_logisimNet26));

    AND_GATE_3_INPUTS #(.BubblesMask(3'b000))
       GATES_22 (.input1(s_logisimNet2),
                 .input2(s_logisimNet11),
                 .input3(s_logisimNet12),
                 .result(s_logisimNet16));

    AND_GATE_3_INPUTS #(.BubblesMask(3'b000))
       GATES_23 (.input1(s_logisimNet11),
                 .input2(s_logisimNet21),
                 .input3(s_logisimNet12),
                 .result(s_logisimNet22));
```

```
    OR_GATE #(.BubblesMask(2'b00))
       GATES_24 (.input1(s_logisimNet7),
                 .input2(s_logisimNet35),
                 .result(s_logisimNet43));

    OR_GATE_3_INPUTS #(.BubblesMask(3'b000))
       GATES_25 (.input1(s_logisimNet22),
                 .input2(s_logisimNet1),
                 .input3(s_logisimNet20),
                 .result(s_logisimNet35));

    AND_GATE_4_INPUTS #(.BubblesMask(4'h0))
       GATES 26 (.input1(s logisimNet5),
                 .input2(s_logisimNet11),
                 .input3(s_logisimNet18),
                 .input4(s_logisimNet6),
                 .result(s_logisimNet1));

    OR_GATE #(.BubblesMask(2'b00))
       GATES_27 (.input1(s_logisimNet7),
                 .input2(s_logisimNet27),
                 .result(s_logisimNet44));

    OR_GATE_4_INPUTS #(.BubblesMask(4'h0))
       GATES_28 (.input1(s_logisimNet30),
                 .input2(s_logisimNet20),
                 .input3(s_logisimNet13),
                 .input4(s_logisimNet10),
                 .result(s_logisimNet27));

    AND_GATE_3_INPUTS #(.BubblesMask(3'b000))
       GATES_29 (.input1(s_logisimNet12),
                 .input2(s_logisimNet11),
                 .input3(s_logisimNet5),
                 .result(s_logisimNet13));

    AND_GATE_4_INPUTS #(.BubblesMask(4'h0))
       GATES_30 (.input1(s_logisimNet12),
                 .input2(s_logisimNet2),
                 .input3(s_logisimNet0),
                 .input4(s_logisimNet6),
                 .result(s_logisimNet10));

    OR_GATE #(.BubblesMask(2'b00))
       GATES_31 (.input1(s_logisimNet7),
                 .input2(s_logisimNet31),
                 .result(s_logisimNet45));

    OR_GATE_4_INPUTS #(.BubblesMask(4'h0))
       GATES_32 (.input1(s_logisimNet4),
                 .input2(s_logisimNet25),
                 .input3(s_logisimNet3),
                 .input4(s_logisimNet14),
                 .result(s_logisimNet31));

    AND_GATE_4_INPUTS #(.BubblesMask(4'h0))
       GATES_33 (.input1(s_logisimNet18),
                 .input2(s_logisimNet21),
                 .input3(s_logisimNet11),
                 .input4(s_logisimNet2),
                 .result(s_logisimNet4));
```

```
    AND_GATE_3_INPUTS #(.BubblesMask(3'b000))
       GATES_34 (.input1(s_logisimNet12),
                 .input2(s_logisimNet21),
                 .input3(s_logisimNet5),
                 .result(s_logisimNet20));

    AND_GATE_3_INPUTS #(.BubblesMask(3'b000))
       GATES_35 (.input1(s_logisimNet11),
                 .input2(s_logisimNet21),
                 .input3(s_logisimNet2),
                 .result(s_logisimNet30));


  endmodule
```

DispNum 的 Verilog 代码如下

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

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

  module DispNum( AN,
                  BTN,
                  SEGMENT,
                  SW );


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

*****************************************************************
**********/
    input [1:0] BTN;
    input [7:0] SW;


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

*****************************************************************
**********/
    output [3:0] AN;
    output [7:0] SEGMENT;


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

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

*************************************************************************
**********/
    wire [7:0] s_logisimBus0;
    wire [7:0] s_logisimBus1;
    wire [1:0] s_logisimBus14;
    wire [3:0] s_logisimBus15;
    wire      s_logisimNet10;
    wire      s_logisimNet11;
    wire      s_logisimNet12;
    wire      s logisimNet13;
    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_logisimNet23;
    wire      s_logisimNet24;
    wire      s_logisimNet25;
    wire      s_logisimNet3;
    wire      s_logisimNet4;
    wire      s_logisimNet5;
    wire      s_logisimNet6;
    wire      s_logisimNet7;
    wire      s_logisimNet8;
    wire      s_logisimNet9;


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

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


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

*************************************************************************
**********/
    assign s_logisimBus0[7:0]  = SW;
    assign s_logisimBus14[1:0] = BTN;


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

*************************************************************************
**********/
    assign AN     = s_logisimBus15[3:0];
```

```
    assign SEGMENT = s_logisimBus1[7:0];


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

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

    // NOT Gate
    assign s logisimBus15[3] = ~s logisimBus0[7];

    // NOT Gate
    assign s_logisimBus15[2] = ~s_logisimBus0[6];

    // NOT Gate
    assign s_logisimBus15[1] = ~s_logisimBus0[5];

    // NOT Gate
    assign s_logisimBus15[0] = ~s_logisimBus0[4];


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

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

    MyMC14495   M1 (.D0(s_logisimBus0[0]),
                    .D1(s_logisimBus0[1]),
                    .D2(s_logisimBus0[2]),
                    .D3(s_logisimBus0[3]),
                    .LE(s_logisimBus14[0]),
                    .a(s_logisimBus1[0]),
                    .b(s_logisimBus1[1]),
                    .c(s_logisimBus1[2]),
                    .d(s_logisimBus1[3]),
                    .e(s_logisimBus1[4]),
                    .f(s_logisimBus1[5]),
                    .g(s_logisimBus1[6]),
                    .p(s_logisimBus1[7]),
                    .point(s_logisimBus14[1]));

  endmodule
```

2、使用 Vivado 对电路生成的 Verilog 代码进行仿真（注意，此部分仿真内容是 MyMC14485 的电路图）

（1）新建工程，此处命名为 project_4

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。

导入后状态如下

（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

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：

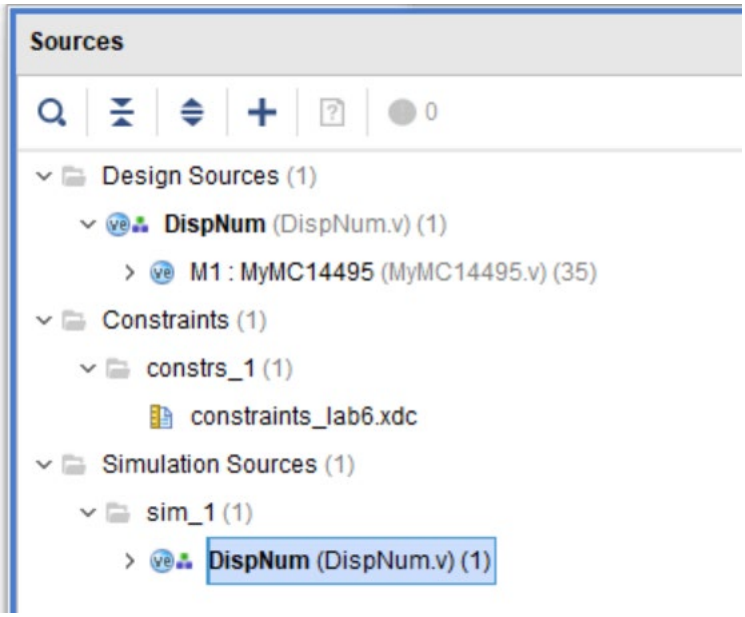
```
`timescale 1ns / 1ps

module MyMC14495_tb();

// Inputs
reg D0;
reg D1;
reg D2;
reg D3;
reg LE;
reg point;

// Output
wire p;
wire a;
wire b;
wire c;
wire d;
wire e;
wire f;
wire g;

// Instantiate the UUT
MyMC14495 MC14495_inst (
.D0(D0),
.D1(D1),
.D2(D2),
.D3(D3),
.LE(LE),
.point(point),
.p(p),
.a(a),
.b(b),
.c(c),
.d(d),
.e(e),
.f(f),
.g(g)
);

initial begin
    D3 = 0;
    D2 = 0;
    D1 = 0;
    D0 = 0;
    LE = 0;
    point = 0;
    #50;
    repeat(16) begin
```

```
      {D3, D2, D1, D0} = {D3, D2, D1, D0} + 4'b1; #50;
    end
    LE = 1;
end
endmodule
```

3、使用 Vivado 综合并上板验证（此处上板验证的是 DispNum）

（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。设置后结果如下：



修改后代码如下：

```
# Filename: constraints_lab6.xdc
## Constraints file for Lab6

# 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 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]}]

# Key as inputs
set_property PACKAGE_PIN AF13 [get_ports {BTN[0]}]
set_property IOSTANDARD LVCMOS15 [get_ports {BTN[0]}]
set_property PACKAGE_PIN AF10 [get_ports {BTN[1]}]
set_property IOSTANDARD LVCMOS15 [get_ports {BTN[1]}]
```

```
# 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]}]


# # 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
```

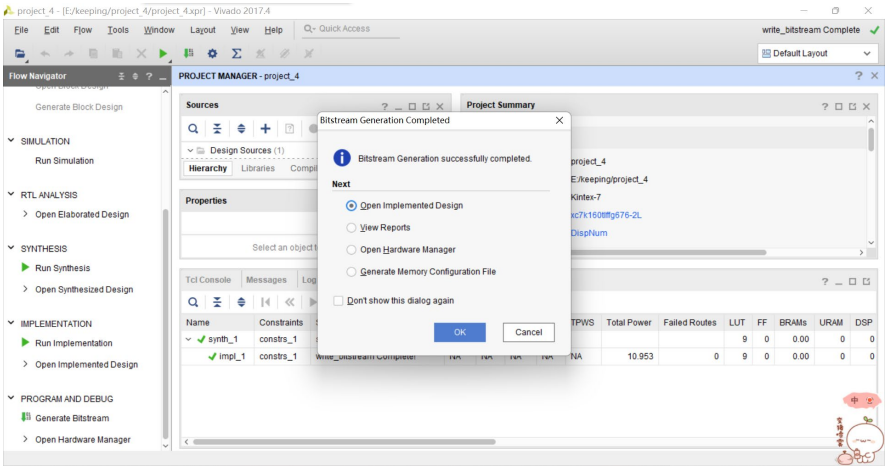
```
# set_property PACKAGE_PIN N26 [get_ports LEDCLK]
# set_property PACKAGE_PIN N24 [get_ports LEDCLR]
# set_property PACKAGE_PIN M26 [get_ports LEDDT]
# set_property IOSTANDARD LVCMOS33 [get_ports LEDCLK]
# set_property IOSTANDARD LVCMOS33 [get_ports LEDCLR]
# set_property IOSTANDARD LVCMOS33 [get_ports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

MOS33 对应 sword 板上数码管的区域，MOS15 对应按钮区域。应注意四位七段数码管共用一套阴极，而不是每位数码管都有自己的阴极，这为下一个实验我们实现每位数码管独立显示数字做出铺垫。

（2）生成 bitstream 并烧录

配置完成后，得到 bitstream。

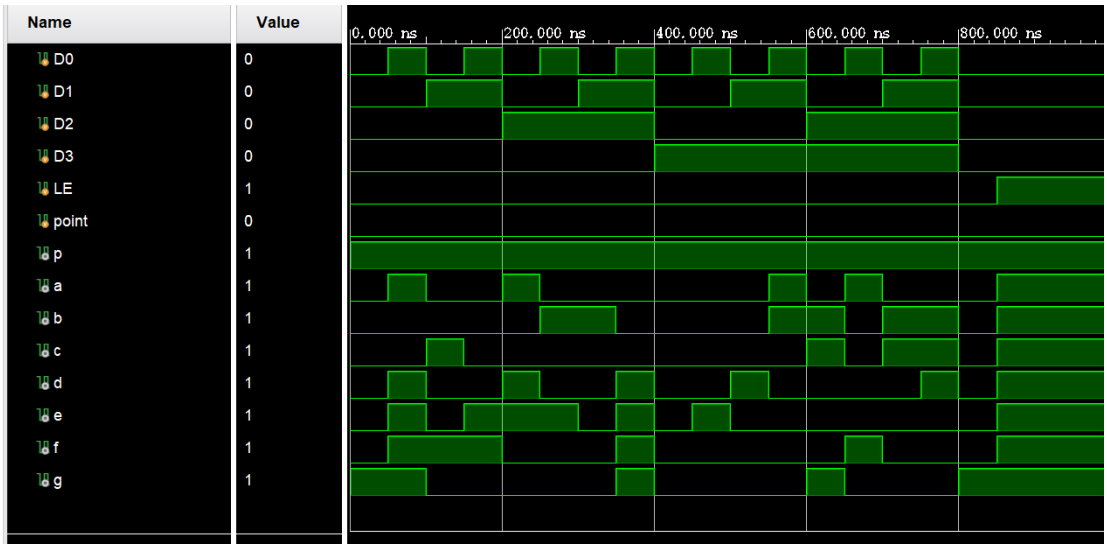


之后将下载器连接到电脑上。点击 PROGRAM AND DEBUG > Open Hardware Manager > Open Target > Auto Connect 进行识别和连接，成功连接后，点击 Program Device 选择 xc7k160t 设备，在下载程序界面选择我们刚刚生成的比特流文件，将其下载到板上。之后在板上实现相关操作。

二、实验结果与分析

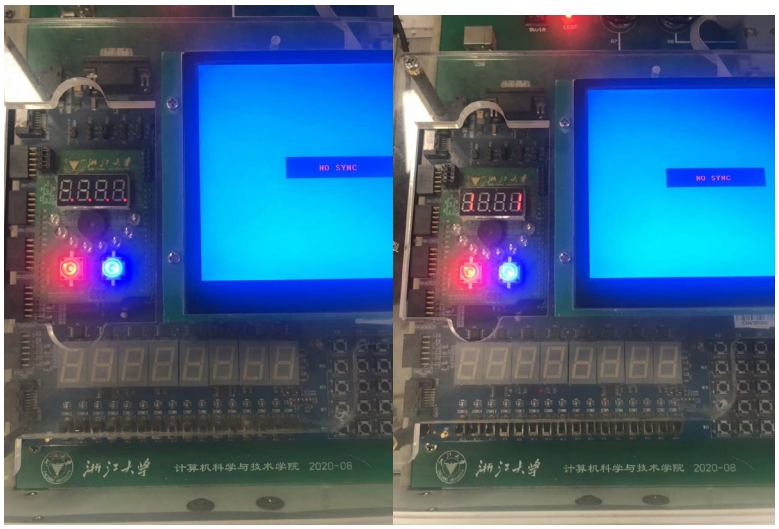
（一）MyMC14485 的仿真结果

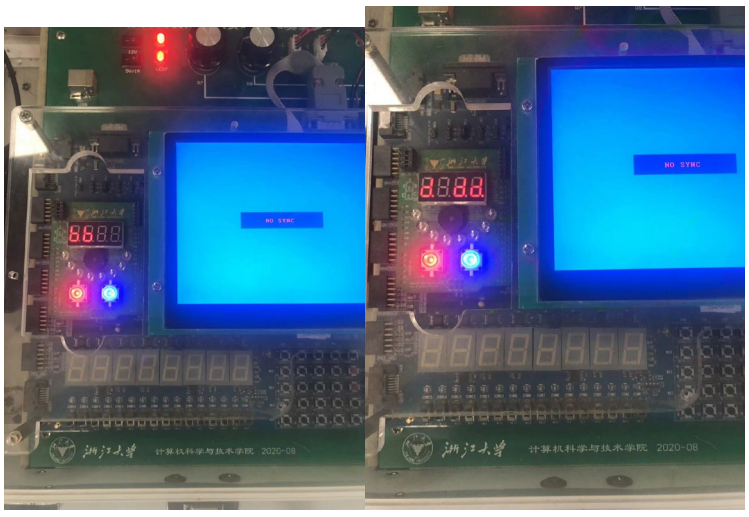
我们可以对照着七段数码管的分布图来检查仿真结果是否正确。发现结果确实如此，证明我们在卡诺图的化简过程以及原理图的绘制过程中没有因粗心出错。



如图所示，开关对显示屏控制状态正确，说明仿真过程成功实现。

（二）DispNum 上板结果





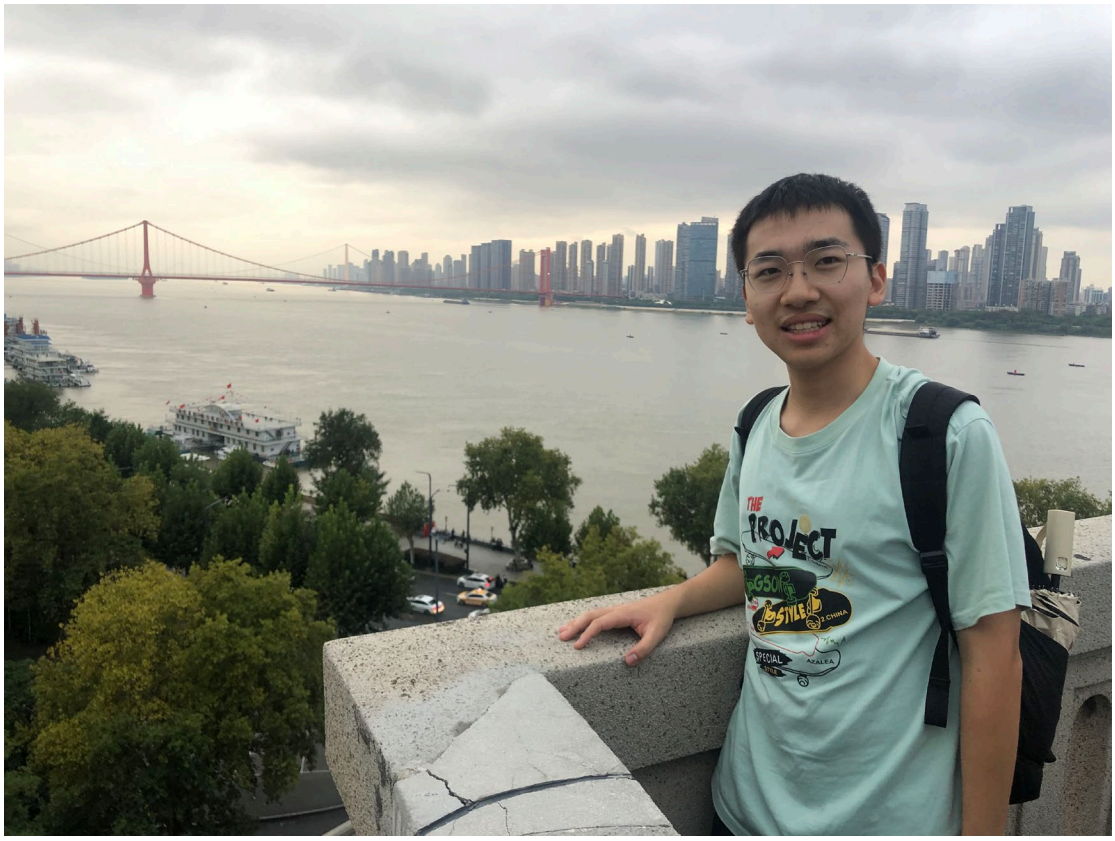
如图，可以通过开关控制四个七段数码管的亮起与熄灭，可以控制小数点是否显现，可以通

过开关控制显示 0~F16 个 16 进制数，行为与预期完全相符。

三、讨论、心得

本次实验在 MyMC14495 模块绘制时，使用了很多门电路。从中我学会了如何绘制一个较难的电路图。在线路较多时，选择分层、分点绘制。本次实验也是十分有趣的，显示十六个数字成功的时候我的内心获得了极大的成就感。
这次的画图确实非常复杂，这对我们的耐心和细心都是极大的考验，我周围有同学就因为不小心少连了一根线，导致最终出现错误的显示信息。我们应该尽量规避这样的错误。

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：	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 7 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 7：多路选择器设计与应用

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 7 日

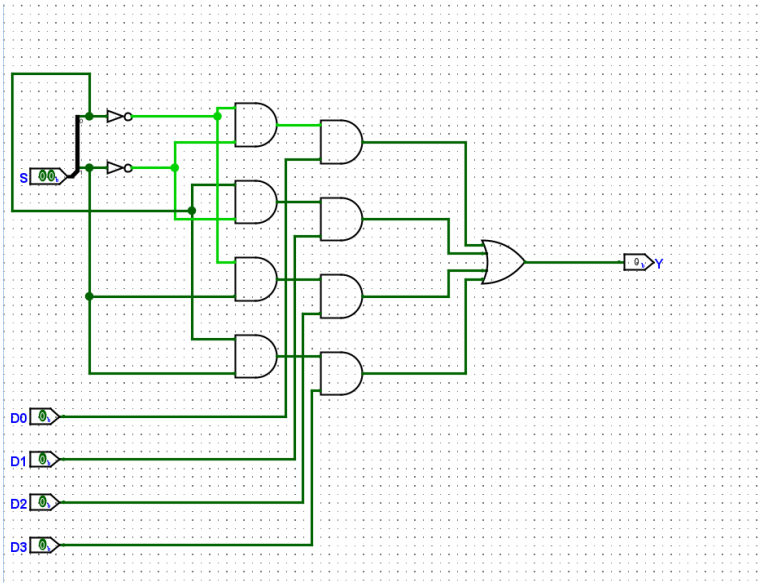
一、操作方法与实验步骤

（一）原理图设计实现 Mux4to1b4 模块

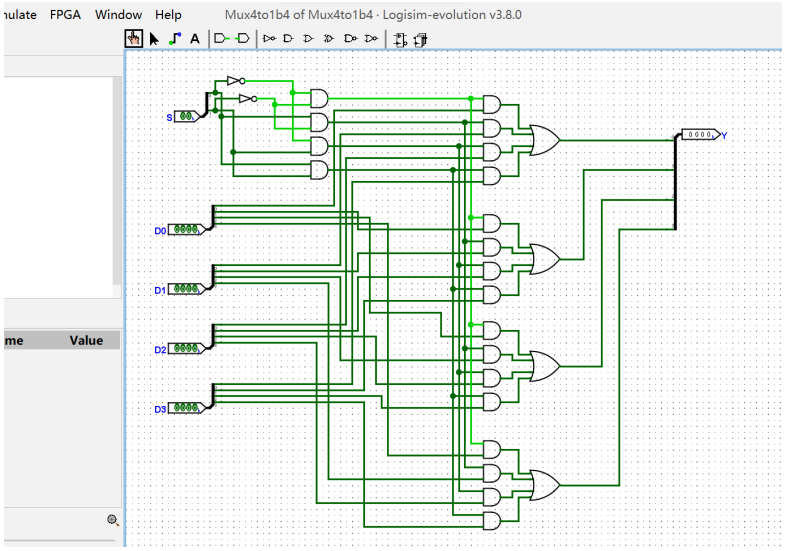
1、使用 Logisim 绘制芯片逻辑电路图

先绘制 Mux4to1

绘制结果如下：



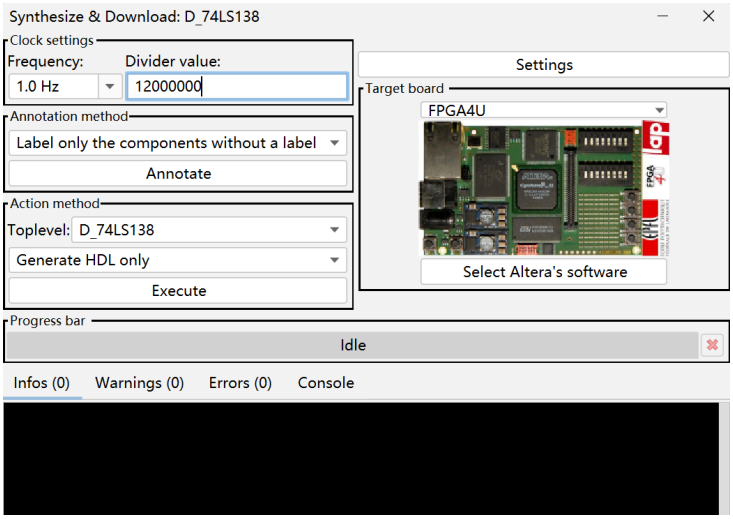
绘制完成后，绘制 Mux4to1b4，绘制如下：



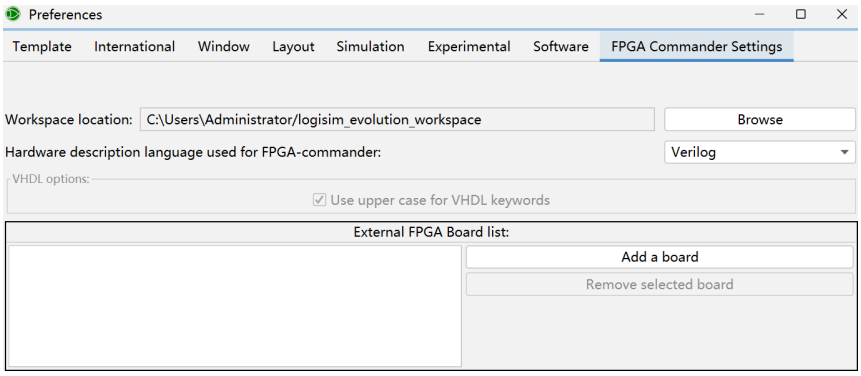
需要注意的是，两次绘制完成后，需要分别更改出口、入口的名称。同时注意分线器下标与连接端口的对应关系。可以通过改变输入值几次判断输出值是否符合预期的方法判断是

否绘制正确。

之后导出电路图为 Verilog 代码



注意 Target board 选择 FPGA4U



注意在 preferences 中把代码格式选为 Verilog

之后 Execute 生成代码，保存

Mux4to1b4 的 Verilog 代码如下：

```
  /*****************************************************************************
************
   **  Logisim-evolution  goes  FPGA  automatic  generated  Verilog  code
**
   **                              https://github.com/logisim-evolution/
**
   **                                                                        **
   **                 Component                :                Mux4to1b4
**
   **                                                                        **

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

  module Mux4to1b4( D0,
                    D1,
                    D2,
                    D3,
                    S,
                    Y );


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

```
***********
    **        The        inputs        are        defined        here
**

*************************************************************************
**********/
    input [3:0] D0;
    input [3:0] D1;
    input [3:0] D2;
    input [3:0] D3;
    input [1:0] S;


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

*************************************************************************
**********/
    output [3:0] Y;


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

*************************************************************************
**********/
    wire [1:0] s_logisimBus44;
    wire [3:0] s_logisimBus45;
    wire [3:0] s_logisimBus46;
    wire [3:0] s_logisimBus47;
    wire [3:0] s_logisimBus48;
    wire [3:0] s_logisimBus49;
    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_logisimNet19;
    wire      s_logisimNet2;
    wire      s_logisimNet20;
    wire      s_logisimNet21;
    wire      s_logisimNet22;
    wire      s_logisimNet23;
    wire      s_logisimNet24;
    wire      s_logisimNet25;
    wire      s_logisimNet26;
    wire      s_logisimNet27;
    wire      s_logisimNet28;
    wire      s_logisimNet29;
    wire      s_logisimNet3;
    wire      s_logisimNet30;
```

```
    wire      s_logisimNet31;
    wire      s_logisimNet32;
    wire      s_logisimNet33;
    wire      s_logisimNet34;
    wire      s_logisimNet35;
    wire      s_logisimNet36;
    wire      s_logisimNet37;
    wire      s_logisimNet38;
    wire      s_logisimNet39;
    wire      s_logisimNet4;
    wire      s_logisimNet40;
    wire      s_logisimNet41;
    wire      s logisimNet42;
    wire      s_logisimNet43;
    wire      s_logisimNet5;
    wire      s_logisimNet6;
    wire      s_logisimNet7;
    wire      s_logisimNet8;
    wire      s_logisimNet9;


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

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


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

*********************************************************************
**********/
    assign s_logisimBus44[1:0] = S;
    assign s_logisimBus45[3:0] = D0;
    assign s_logisimBus46[3:0] = D1;
    assign s_logisimBus47[3:0] = D2;
    assign s_logisimBus48[3:0] = D3;


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

*********************************************************************
**********/
    assign Y = s_logisimBus49[3:0];


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

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

```
    // NOT Gate
    assign s_logisimNet8 = ~s_logisimBus44[0];

    // NOT Gate
    assign s_logisimNet15 = ~s_logisimBus44[1];


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

**********************************************************************
**********/
    AND_GATE #(.BubblesMask(2'b00))
       GATES_1 (.input1(s_logisimBus44[0]),
              .input2(s_logisimNet15),
              .result(s_logisimNet1));

    AND_GATE #(.BubblesMask(2'b00))
       GATES_2 (.input1(s_logisimNet8),
              .input2(s_logisimBus44[1]),
              .result(s_logisimNet16));

    AND_GATE #(.BubblesMask(2'b00))
       GATES_3 (.input1(s_logisimBus44[0]),
              .input2(s_logisimBus44[1]),
              .result(s_logisimNet0));

    AND_GATE #(.BubblesMask(2'b00))
       GATES_4 (.input1(s_logisimNet8),
              .input2(s_logisimNet15),
              .result(s_logisimNet25));

    AND_GATE #(.BubblesMask(2'b00))
       GATES_5 (.input1(s_logisimNet1),
              .input2(s_logisimBus46[0]),
              .result(s_logisimNet4));

    AND_GATE #(.BubblesMask(2'b00))
       GATES_6 (.input1(s_logisimNet16),
              .input2(s_logisimBus47[0]),
              .result(s_logisimNet42));

    AND_GATE #(.BubblesMask(2'b00))
       GATES_7 (.input1(s_logisimNet0),
              .input2(s_logisimBus48[0]),
              .result(s_logisimNet18));

    AND_GATE #(.BubblesMask(2'b00))
       GATES_8 (.input1(s_logisimNet25),
              .input2(s_logisimBus45[1]),
              .result(s_logisimNet19));

    AND_GATE #(.BubblesMask(2'b00))
       GATES_9 (.input1(s_logisimNet1),
              .input2(s_logisimBus46[1]),
              .result(s_logisimNet2));

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

```
   GATES_10 (.input1(s_logisimNet16),
            .input2(s_logisimBus47[1]),
            .result(s_logisimNet41));

AND_GATE #(.BubblesMask(2'b00))
   GATES_11 (.input1(s_logisimNet0),
            .input2(s_logisimBus48[1]),
            .result(s_logisimNet20));

AND_GATE #(.BubblesMask(2'b00))
   GATES_12 (.input1(s_logisimNet25),
            .input2(s_logisimBus45[2]),
            .result(s_logisimNet35));

AND_GATE #(.BubblesMask(2'b00))
   GATES_13 (.input1(s_logisimNet1),
            .input2(s_logisimBus46[2]),
            .result(s_logisimNet39));

AND_GATE #(.BubblesMask(2'b00))
   GATES_14 (.input1(s_logisimNet16),
            .input2(s_logisimBus47[2]),
            .result(s_logisimNet27));

AND_GATE #(.BubblesMask(2'b00))
   GATES_15 (.input1(s_logisimNet0),
            .input2(s_logisimBus48[2]),
            .result(s_logisimNet36));

AND_GATE #(.BubblesMask(2'b00))
   GATES_16 (.input1(s_logisimNet25),
            .input2(s_logisimBus45[3]),
            .result(s_logisimNet37));

AND_GATE #(.BubblesMask(2'b00))
   GATES_17 (.input1(s_logisimNet1),
            .input2(s_logisimBus46[3]),
            .result(s_logisimNet40));

AND_GATE #(.BubblesMask(2'b00))
   GATES_18 (.input1(s_logisimNet16),
            .input2(s_logisimBus47[3]),
            .result(s_logisimNet28));

AND_GATE #(.BubblesMask(2'b00))
   GATES_19 (.input1(s_logisimNet0),
            .input2(s_logisimBus48[3]),
            .result(s_logisimNet38));

AND_GATE #(.BubblesMask(2'b00))
   GATES_20 (.input1(s_logisimNet25),
            .input2(s_logisimBus45[0]),
            .result(s_logisimNet17));

OR_GATE_4_INPUTS #(.BubblesMask(4'h0))
   GATES_21 (.input1(s_logisimNet17),
            .input2(s_logisimNet4),
            .input3(s_logisimNet42),
            .input4(s_logisimNet18),
            .result(s_logisimBus49[0]));
```

```
    OR_GATE_4_INPUTS #(.BubblesMask(4'h0))
       GATES_22 (.input1(s_logisimNet19),
                 .input2(s_logisimNet2),
                 .input3(s_logisimNet41),
                 .input4(s_logisimNet20),
                 .result(s_logisimBus49[1]));

    OR_GATE_4_INPUTS #(.BubblesMask(4'h0))
       GATES_23 (.input1(s_logisimNet35),
                 .input2(s_logisimNet39),
                 .input3(s_logisimNet27),
                 .input4(s_logisimNet36),
                 .result(s_logisimBus49[2]));

    OR_GATE_4_INPUTS #(.BubblesMask(4'h0))
       GATES_24 (.input1(s_logisimNet37),
                 .input2(s_logisimNet40),
                 .input3(s_logisimNet28),
                 .input4(s_logisimNet38),
                 .result(s_logisimBus49[3]));


  endmodule
```

2、使用 Vivado 对电路生成的 Verilog 代码进行仿真（注意，此部分仿真内容是 Mux4to1b4 的电路图）

（1）新建工程，此处命名为 project_5

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。

导入后状态如下

（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

之后进行仿真，仿真结果见“二、实验结果与分析”，仿真成功。

本次实验我自己撰写了仿真代码如下：

```
 `timescale 1ns/1ns

// Filename: lab7_tb.v
module lab7_tb ();

    reg [3:0] D0;
    reg [3:0] D1;
    reg [3:0] D2;
    reg [3:0] D3;
    reg [1:0] S;

    wire [3:0] Y;

    Mux4to1b4 Mux4to1b4_inst (
```

```
        .Y(Y),
        .D0(D0),
        .D1(D1),
        .D2(D2),
        .D3(D3),
        .S(S)
    );
integer i,j;
    initial begin
      S=0;
      D0=0;
      D1=0;
      D2=0;
      D3=0;
      for(i=0;i<4;i=i+1)begin
        S=i;
        for(j=0;j<4;j=j+1)begin
            D0=j;
            D1=j+4;
            D2=j+8;
            D3=j+12;
            #50;
        end
      end
    end

endmodule
```

（二）计分模块设计

1、设计 clkdiv 模块

计时器模块。在上一个实验中我们已经知道，4 个七段数码管公用一套 abcdefgp 输入接 口，因此我们需要用计时器，在极短时间内快速切换，每切换到一位输出其对应的信号。由 于人眼有视觉停留效应，看起来就成了 4 位不同的数字。

新建 Verilog 代码文件 clkdiv.v

输入 Verilog 代码

```
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
```

2、设计 Create Number 模块

由于我们需要实现按钮按一下，数字加一，因此需要这样的模块来进行运算

新建 Verilog 代码文件 CreateNumber.v

输入 Verilog 代码

```
module CreateNumber(
    input [3:0]              btn,
    output reg [15:0]     num
);

    wire [3:0] A, B, C, D;

    initial num <= 16'b1010_1011_1100_1101;

    assign A = num[15:12] + 4'b1;
    assign B = num[11: 8] + 4'b1;
    assign C = num[ 7: 4] + 4'b1;
    assign D = num[ 3: 0] + 4'b1;

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

endmodule
```

3、设计 Display Number 模块

```
# F
```

输入 4 路的 4 位信号，并根据选择信号输出，控制七段数码管的显像

导入上次实验的元器件 MyMC14495 和设计好的 DisplaySync.v clkdiv.v

DisplaySync.v 的 Verilog 代码如下

```
/*****************************************************************************
**
** 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 [3:0]    s_logisimBus10;
        wire [15:0] s_logisimBus11;
        wire [3:0]    s_logisimBus2;
        wire [3:0]    s_logisimBus3;
        wire [3:0]    s_logisimBus4;
        wire [3:0]    s_logisimBus5;
        wire [3:0]    s_logisimBus6;
        wire [3:0]    s_logisimBus9;
        wire            s_logisimNet16;
        wire            s_logisimNet17;
        wire            s_logisimNet18;
        wire            s_logisimNet19;
        wire            s_logisimNet20;
        wire            s_logisimNet21;
        wire            s_logisimNet22;
        wire            s_logisimNet23;
        wire            s  logisimNet7;
        wire            s_logisimNet8;


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

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


/********************************************************************************
*********
        **      Here    all     input   connections     are     defined
**
********************************************************************************
********/
        assign s_logisimBus10[3:0]    = points;
        assign s_logisimBus11[15:0] = hexs;
        assign s_logisimBus1[1:0]     = scan;
        assign s_logisimBus9[3:0]     = LEs;


/********************************************************************************
*********
        **      Here    all     output  connections     are     defined
**
********************************************************************************
********/
        assign AN       = s_logisimBus4[3:0];
        assign HEX      = s_logisimBus0[3:0];
        assign LE       = s_logisimNet7;
        assign point = s_logisimNet8;


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

        // Constant
        assign    s_logisimBus5[3:0]    =    4'h7;


        // Constant
        assign    s_logisimBus6[3:0]    =    4'hD;


        // Constant
        assign    s_logisimBus2[3:0]    =    4'hB;


        // Constant
        assign    s_logisimBus3[3:0]    =    4'hE;
```

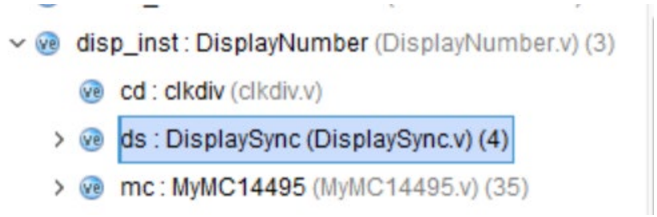
```
/*************************************************************************
*********
        **          Here          all          sub-circuits          are          defined
**

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

        Mux4to1b4     mux_hexs (.D0(s_logisimBus11[3:0]),
                                .D1(s_logisimBus11[7:4]),
                                .D2(s_logisimBus11[11:8]),
                                .D3(s_logisimBus11[15:12]),
                                .S(s  logisimBus1[1:0]),
                                .Y(s_logisimBus0[3:0]));

        Mux4to1     mux_points (.D0(s_logisimBus10[0]),
                                .D1(s_logisimBus10[1]),
                                .D2(s  logisimBus10[2]),
                                .D3(s_logisimBus10[3]),
                                .S(s_logisimBus1[1:0]),
                                .Y(s_logisimNet8));

        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_logisimNet7));

        Mux4to1b4     mux_AN (.D0(s_logisimBus3[3:0]),
                              .D1(s_logisimBus6[3:0]),
                              .D2(s_logisimBus2[3:0]),
                              .D3(s_logisimBus5[3:0]),
                              .S(s_logisimBus1[1:0]),
                              .Y(s_logisimBus4[3:0]));

    endmodule
```

DisplayNumber 的 Verilog 代码如下：

```
    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
```

导入后如图所示



4、各模块的汇总

新建 Verilog 代码文件，并命名为 top

右键将该文件设置为 Top，只有最顶层 module 才能进行引脚约束等操作

编写 Verilog 代码，连接 CreateNumber 和 DisplayNumber 模块

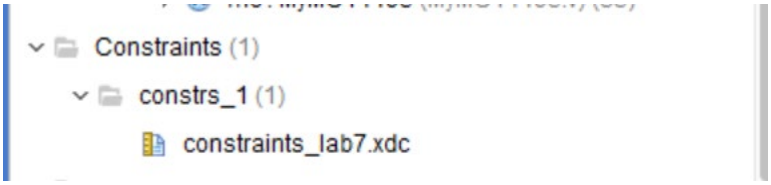
```
module top(
    input clk,
    input [7:0] SW,
```

```
        input [3:0] btn,
        output[3:0] AN,
        output[7:0] SEGMENT,
        output btnx
    );

        wire [15:0] num;
        assign btnx = 0;
        CreateNumber create_inst(
            .btn(btn),
            .num(num)
        );

        DisplayNumber disp_inst(
            .clk(clk),
            .rst(1'b0),
            .hexs(num),
            .points(SW[7:4]),
            .LEs(SW[3:0]),
            .AN(AN),
            .SEGMENT(SEGMENT)
        );

    endmodule
```

5、使用 Vivado 综合并上板验证

（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。设置后结果如下：



修改后代码如下：

```
# Filename: constraints_lab7.xdc
## Constraints file for Lab7

# Main clock
set_property CLOCK_DEDICATED_ROUTE FALSE [get_nets btn_IBUF[0]]
set_property CLOCK_DEDICATED_ROUTE FALSE [get_nets btn_IBUF[1]]
set_property CLOCK_DEDICATED_ROUTE FALSE [get_nets btn_IBUF[2]]
set_property CLOCK_DEDICATED_ROUTE FALSE [get_nets btn_IBUF[3]]

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 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]}]

# Key as inputs
set_property PACKAGE_PIN V18 [get_ports {btn[3]}]
set_property IOSTANDARD LVCMOS18 [get_ports {btn[3]}]
set_property PACKAGE_PIN V19 [get_ports {btn[2]}]
set  property IOSTANDARD LVCMOS18 [get  ports {btn[2]}]
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 PACKAGE_PIN W16 [get_ports {btnx}]
set_property IOSTANDARD LVCMOS18 [get_ports {btnx}]
# 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]}]


# # 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]
```

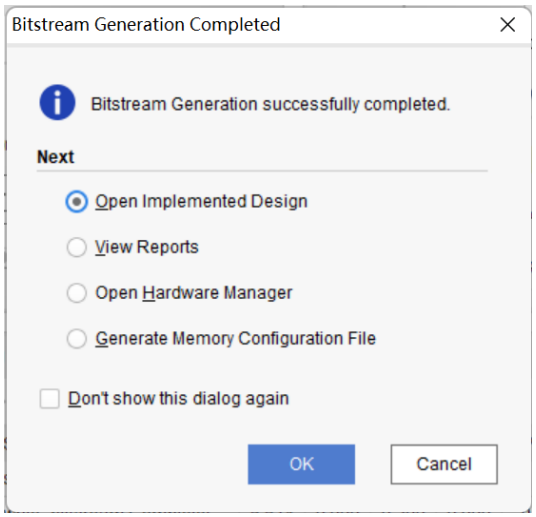
```
# # create_clock -period 10.000 -name clk [get_ports "clk_p"]

# # 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。

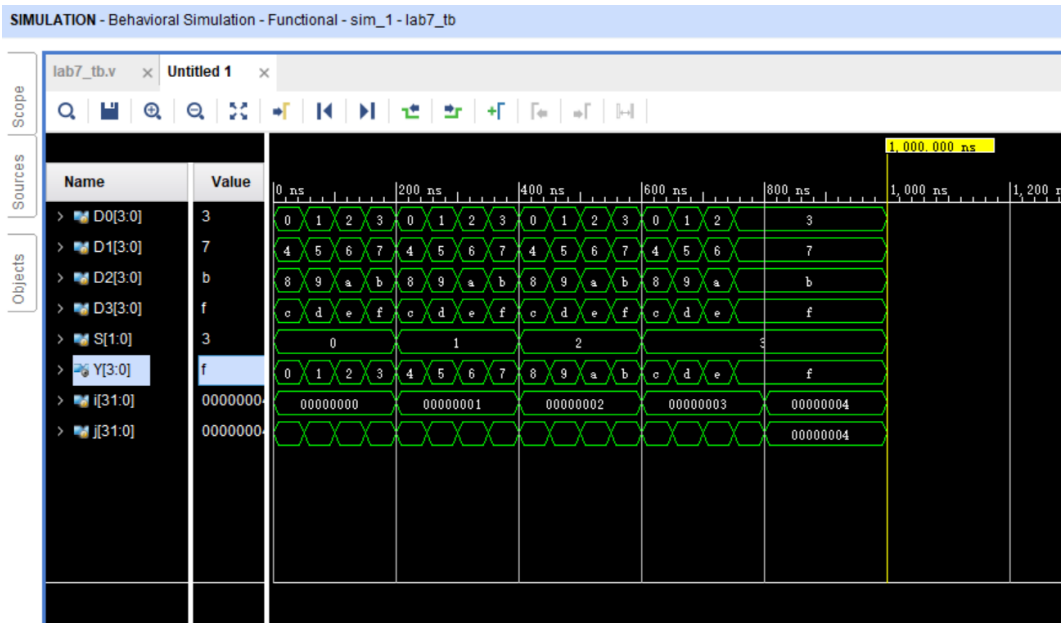


之后将下载器连接到电脑上。点击 PROGRAM AND DEBUG > Open Hardware Manager > Open Target > Auto Connect 进行识别和连接，成功连接后，点击 Program Device 选择 xc7k160t 设备，在下载程序界面选择我们刚刚生成的比特流文件，将其下载到板上。之后在板上实现相关操作。

二、实验结果与分析

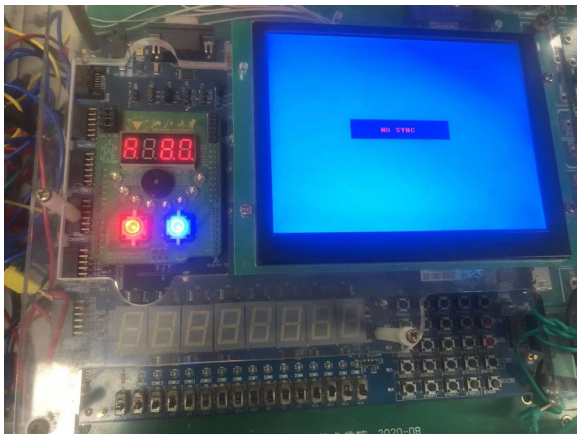
（一）Mux4to1b4 的仿真结果

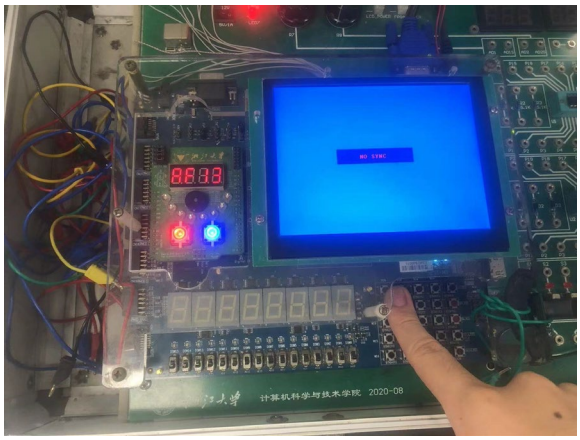
分析原理，仿真图中遍历了输出的每一种情况，发现完全符合预期。证明了我对仿真代码的撰写、导入以及原理图的绘制过程中没有因粗心出错。



如图所示，开关对显示屏控制状态正确，说明仿真过程成功实现。

（二）计分板的使用上板结果





如图。成功实现所需要功能。由于按键时手会抖动，故每次按按钮数字可能会增加好几位，是正常现象。如果想要只增加 1，需要设计加入防抖动模块。

三、讨论、心得

本次实验比较困难，也收获到了很多。
（1）文件目录结构
在绘制多路选择器时，注意到，被其他器件调用的元件不再在文件目录中独自显示，而是置于调用它的元件的目录下，且每个被调用元件都有自己的命名。我们在 Verilog 代码中可以自行命名，而用原理图时默认为 XLXI 命名，双击该期间后，在弹出的对话框可以重命名。
（2）对 Verilog 代码的理解学习本次实验我们较多使用了 Verilog 代码，加深了我对它的理解。 以 top 代码为例： 我们每一个文件都相当于是一个元器件，括号里面的内容是此元器件的输入输出接口。 对于 top module 来说，接口用于在引脚约束以后，对应于开发板上一个个按钮、LED 灯等器件，供我们在开发板上手动操作并观察效果。对于其他元器件来说，接口就是供被调用使用的。接下来的 wire [15:0] num 就相当于函数中定义的变量，它先经过 Create Number 函数（元器件）被赋值，然后作为 disp_num 的参数，进行输出显示。
以上就是我本次实验的心得

四、个人生活照片