浙江大学

本科实验报告

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

2025 年 3 月 29 日

浙江大学实验报告

课程名称：数字逻辑设计 实验类型：

实验项目名称：多路选择器设计与应用

学生姓名：王浩雄 专业：混合班 学号：3230106032

同组学生姓名：无 指导老师：马德

实验地点：紫金港东 4-509 实验日期：2025 年 3 月 29 日

一、 实验目的和要求

① 掌握多路选择器的的逻辑构成和逻辑功能；

② 掌握多路选择器的使用方法；

③ 掌握 4 位数码管扫描显示方法；

④ 4 位数码管显示应用——记分板设计。

二、 实验内容

① 4 位四选一选择器设计

② 记分板设计

三、 主要仪器设备

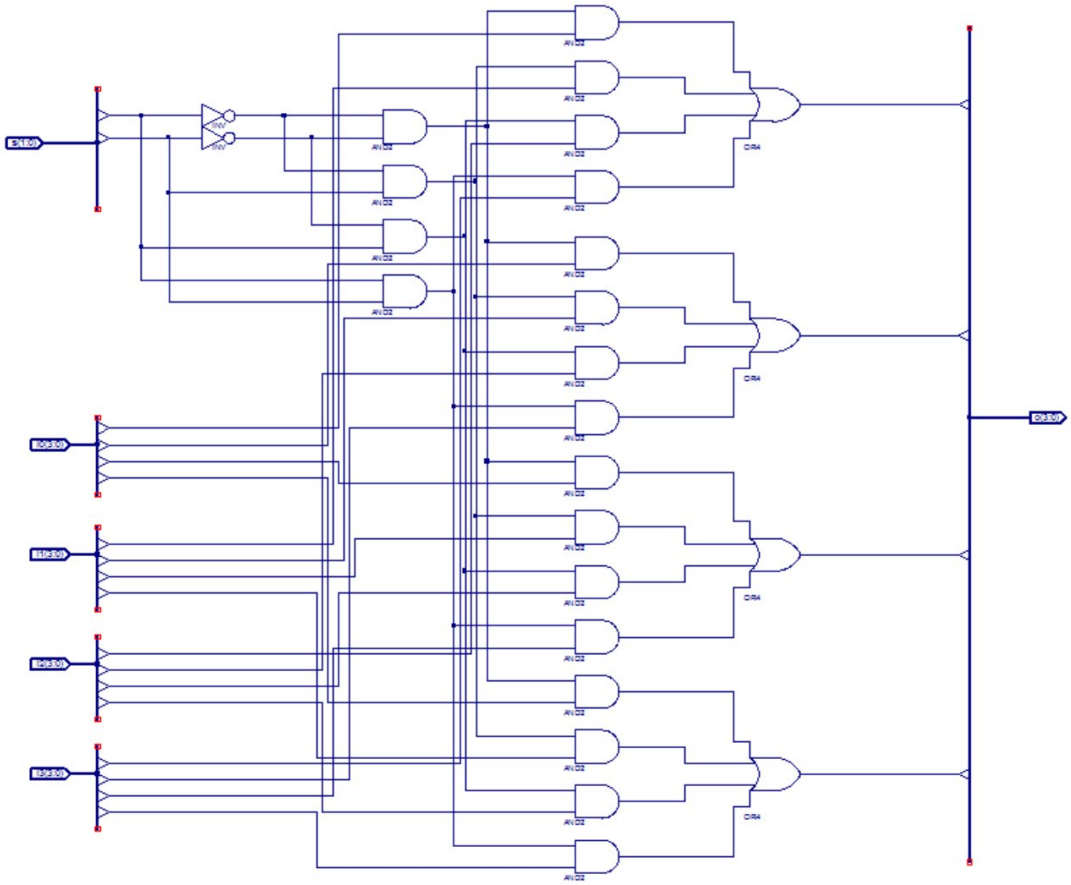
① 装有 ISE 14.7 和 Vivado 2024.2 Enterprise 的计算机 1 台

② SWORD 开发板 1 套

四、 操作方法与实验步骤

1、4 位四选一多路选择器设计

① 在 Xilinx ISE 中新建 Schematic 文件，结合设计要求和真值表，绘制电路连接图如下：



② 由电路连接图生成的硬件描述代码如下：

```
module Mux4to1b4(I0, I1, I2, I3, s, o);

    input [3:0] I0, I1, I2, I3;
    input [1:0] s;
   output [3:0] o;

   wire XLXN_1, XLXN_2, XLXN_3, ...;

   OR4  XLXI_1
(.I0(XLXN_4), .I1(XLXN_3), .I2(XLXN_2), .I3(XLXN_1), .O(o[0]));
   AND2  XLXI_2 (.I0(I0[0]), .I1(XLXN_35), .O(XLXN_1));
   AND2  XLXI_3 (.I0(I1[0]), .I1(XLXN_39), .O(XLXN_2));
```

```
   AND2  XLXI_4 (.I0(I2[0]), .I1(XLXN_43), .O(XLXN_3));
   AND2  XLXI_5 (.I0(I3[0]), .I1(XLXN_47), .O(XLXN_4));
   AND2  XLXI_6 (.I0(XLXN_26), .I1(XLXN_24), .O(XLXN_35));
   AND2  XLXI_7 (.I0(s[0]), .I1(XLXN_24), .O(XLXN_39));
   AND2  XLXI_8 (.I0(s[1]), .I1(XLXN_26), .O(XLXN_43));
   AND2  XLXI_9 (.I0(s[0]), .I1(s[1]), .O(XLXN_47));
   AND2  XLXI_24 (.I0(I0[1]), .I1(XLXN_35), .O(XLXN_5));
   AND2  XLXI_25 (.I0(I1[1]), .I1(XLXN_39), .O(XLXN_6));
   AND2  XLXI_26 (.I0(I2[1]), .I1(XLXN_43), .O(XLXN_7));
   AND2  XLXI_27 (.I0(I3[1]), .I1(XLXN_47), .O(XLXN_8));
   OR4  XLXI_28
(.I0(XLXN_8), .I1(XLXN_7), .I2(XLXN_6), .I3(XLXN_5), .O(o[1]));
   AND2  XLXI_39 (.I0(I0[2]), .I1(XLXN_35), .O(XLXN_9));
   AND2  XLXI_40 (.I0(I1[2]), .I1(XLXN_39), .O(XLXN_10));
   AND2  XLXI_41 (.I0(I2[2]), .I1(XLXN_43), .O(XLXN_11));
   AND2  XLXI_42 (.I0(I3[2]), .I1(XLXN_47), .O(XLXN_12));
   OR4  XLXI_43
(.I0(XLXN_12), .I1(XLXN_11), .I2(XLXN_10), .I3(XLXN_9), .O(o[2]));
   AND2  XLXI_44 (.I0(I0[3]), .I1(XLXN_35), .O(XLXN_13));
   AND2  XLXI_45 (.I0(I1[3]), .I1(XLXN_39), .O(XLXN_14));
   AND2  XLXI_46 (.I0(I2[3]), .I1(XLXN_43), .O(XLXN_15));
   AND2  XLXI_47 (.I0(I3[3]), .I1(XLXN_47), .O(XLXN_16));
   OR4  XLXI_48
(.I0(XLXN_16), .I1(XLXN_15), .I2(XLXN_14), .I3(XLXN_13), .O(o[3]));
   INV  XLXI_49 (.I(s[1]), .O(XLXN_24));
   INV  XLXI_50 (.I(s[0]), .O(XLXN_26));
endmodule
```

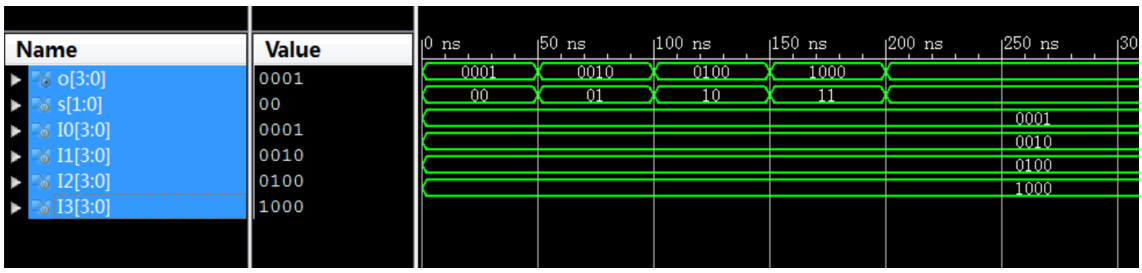
③ 新建仿真激励文件 sim.v 如下：

```
module Mux4to1b4_Mux4to1b4_sch_tb();

// Inputs
   reg [1:0] s;
   reg [3:0] I0, I1, I2, I3;
// Output
   wire [3:0] o;
// Instantiate the UUT
   Mux4to1b4 UUT (.o(o),.s(s),.I0(I0),.I1(I1),.I2(I2),.I3(I3));
// Initialize Inputs
   initial begin
        s = 0;
        I0 = 4'b0001;
        I1 = 4'b0010;
        I2 = 4'b0100;
```

```
        I3 = 4'b1000;
        #50
        s = 1;
        #50
        s = 2;
        #50
        s = 3;
        #50
        s = 0;
    end
endmodule
```

④ 运行仿真，得到如下的仿真波形。该波形与预期相符，表明原理图绘制的正确性。

（注：由于显示问题，下图的变量名与波形没有对齐）



2、记分板设计

① 在 Vivado 中新建项目。实现动态扫描模块的子模块 clkdiv。它是一个简单的时钟分频器，其输出在每个时钟信号上升沿自增。复位信号为同步复位，当时钟信号的正边沿到来且复位信号为有效时（本实验中复位信号为高电平有效）进行复位。使用 Verilog 代码实现 clkdiv 模块如下：

```
module clkdiv(
    input clk, input rst, output reg [31:0] clkdiv
);

    always @(posedge clk or posedge rst) begin
        if (rst) begin
            clkdiv <= 0;
        end else begin
            clkdiv <= clkdiv + 1;
        end
    end
endmodule
```

② 使用 Verilog 代码实现 CreateNumber 模块如下：

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

    wire [3:0] A,B,C,D;
    initial num <= 16'b1010101111001101;

    assign A = num[3:0] + 4'd1;
    assign B = num[7:4] + 4'd1;
    assign C = num[11:8] + 4'd1;
    assign D = num[15:12] + 4'd1;

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

③ 实现动态扫描模块的子模块 DisplaySync，实现数字与使能信号的选择逻辑。使用 Verilog 代码实现 dispsync 模块如下：

```
module dispsync(
    input [15:0] Hexs,
    input   [1:0] Scan,
    input   [3:0] Point,
    input   [3:0] Les,
    output reg[3:0] Hex,
    output reg p,LE,
    output reg[3:0] AN);

    always @* begin
        case(Scan)
            2'b00: begin
                Hex = Hexs[3:0], p = Point[0];
                LE = Les[0] ,AN = 4'b1110;
            end
            2'b01: begin
                Hex = Hexs[7:4], p = Point[1];
                LE = Les[1], AN = 4'b1101;
            end
            2'b10: begin
```

```
                Hex = Hexs[11:8], p = Point[2];
                LE = Les[2], AN = 4'b1011;
            end
            2'b11: begin
                Hex = Hexs[15:12], p = Point[3];
                LE = Les[3], AN = 4'b0111;
            end
        endcase
    end
endmodule
```

④ 设计动态扫描模块 DisplayNumber，使用时钟分频器获得合适的扫描信号，选用 clkdiv[18:17]两位信号。使用 DisplaySync 模块选择合适的数字和使能信号，并将四位数字和小数点控制信号输出到 MyMC14495 模块实例中得到七段信息。使用 Verilog 代码实现 DispNum 模块如下：

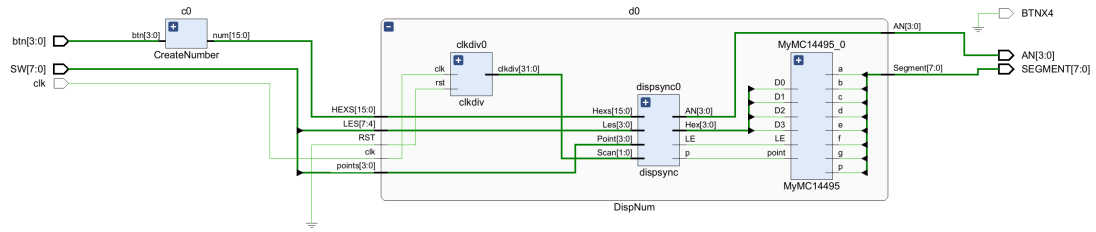
```
module DispNum(
    input clk,
    input [15:0] HEXS,
    input [7:4] LES,
    input [3:0] points,
    input RST,
    output [3:0] AN,
    output [7:0] Segment

);

    wire [31:0] clkdiv;
    wire [3:0] HEX;
    wire p, LE;

    clkdiv clkdiv0(.clk(clk), .rst(RST), .clkdiv(clkdiv));

    dispsync dispsync0(
        .Hexs(HEXS),
        .Scan(clkdiv[18:17]),
        .Point(points),
        .Les(LES),
        .Hex(HEX),
        .p(p),
        .LE(LE),
        .AN(AN[3:0])
    );
```

```
    MyMC14495 MyMC14495_0(
        .LE(LE),
        .point(p),
        .D0(HEX[0]), .D1(HEX[1]), .D2(HEX[2]), .D3(HEX[3]),
        .p(Segment[7]), .a(Segment[0]), .b(Segment[1]),
        .c(Segment[2]), .d(Segment[3]), .e(Segment[4]),
        .f(Segment[5]), .g(Segment[6])
    );
endmodule
```

⑤使用 Verilog 代码实现顶层连接如下：

```
module top(
    input wire clk,
    input wire [7:0] SW,
    input wire [3:0] btn,
    output wire [3:0] AN,
    output wire [7:0] SEGMENT,
    output wire BTNX4
);

    wire [15:0] num;
    CreateNumber c0(btn,num);
    DispNum d0(clk, num, SW[7:4], SW[3:0], 1'b0, AN, SEGMENT);
    assign BTNX4 = 1'b0;

endmodule
```

⑥ 在 Vivado 中运行 Linter Design，查看由 Verilog 代码生成的连接图。经检查，该图与设计要求中提供的连接图逻辑相同。



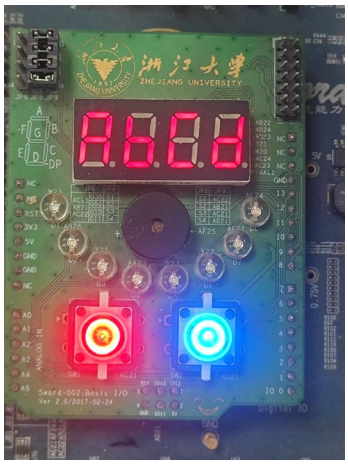
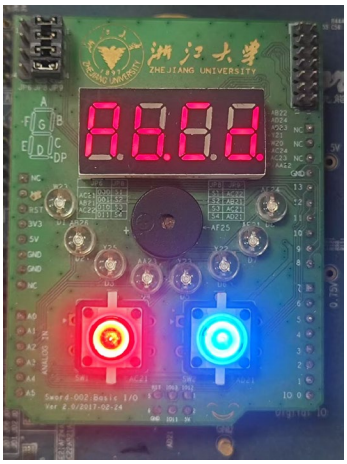
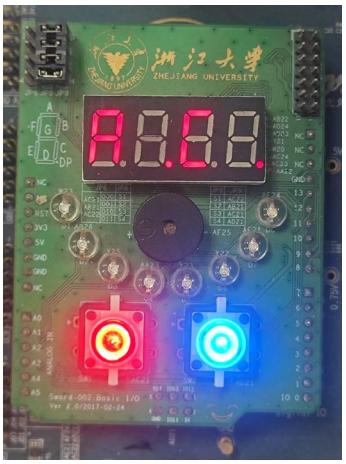
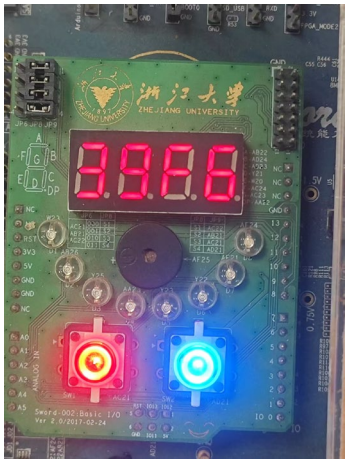
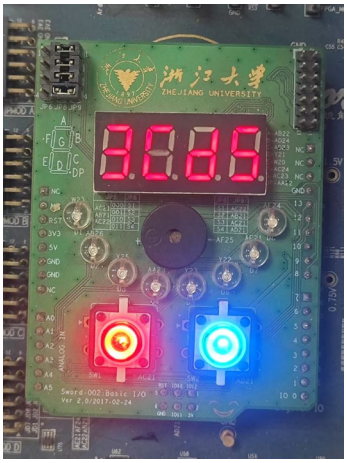
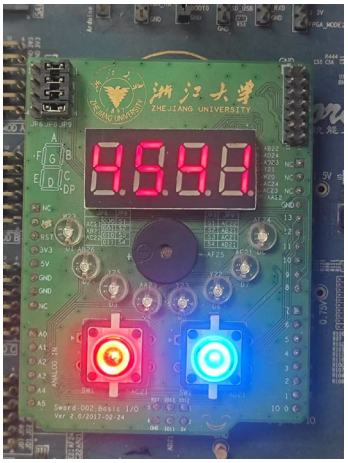
⑦ 将 Xilinx ISE 中使用的.ucf 引脚约束文件修改为 Vivado 的.xdr 引脚约束文件，并补充完全。

```
set_property PACKAGE_PIN AC18 [get_ports clk]
set_property IOSTANDARD LVCMOS18 [get_ports clk]
```

```
set_property PACKAGE_PIN W14       [get_ports {btn[0]}]
set_property IOSTANDARD  LVCMOS18  [get_ports {btn[0]}]

set_property PACKAGE_PIN V14       [get_ports {btn[1]}]
set_property IOSTANDARD  LVCMOS18  [get_ports {btn[1]}]

set_property PACKAGE_PIN V19       [get_ports {btn[2]}]
set_property IOSTANDARD  LVCMOS18  [get_ports {btn[2]}]

set_property PACKAGE_PIN V18       [get_ports {btn[3]}]
set_property IOSTANDARD  LVCMOS18  [get_ports {btn[3]}]

set_property CLOCK_DEDICATED_ROUTE FALSE [get_nets btn*]

set_property PACKAGE_PIN W16       [get_ports BTNX4]
set_property IOSTANDARD  LVCMOS18  [get_ports BTNX4]

set_property PACKAGE_PIN AB22      [get_ports {SEGMENT[0]}]
set_property IOSTANDARD  LVCMOS33  [get_ports {SEGMENT[0]}]

set_property PACKAGE_PIN AD24      [get_ports {SEGMENT[1]}]
set_property IOSTANDARD  LVCMOS33  [get_ports {SEGMENT[1]}]

set_property PACKAGE_PIN AD23      [get_ports {SEGMENT[2]}]
set_property IOSTANDARD  LVCMOS33  [get_ports {SEGMENT[2]}]

set_property PACKAGE_PIN Y21       [get_ports {SEGMENT[3]}]
set_property IOSTANDARD  LVCMOS33  [get_ports {SEGMENT[3]}]

set_property PACKAGE_PIN W20       [get_ports {SEGMENT[4]}]
set_property IOSTANDARD  LVCMOS33  [get_ports {SEGMENT[4]}]

set_property PACKAGE_PIN AC24      [get_ports {SEGMENT[5]}]
set_property IOSTANDARD  LVCMOS33  [get_ports {SEGMENT[5]}]

set_property PACKAGE_PIN AC23      [get_ports {SEGMENT[6]}]
set_property IOSTANDARD  LVCMOS33  [get_ports {SEGMENT[6]}]

set_property PACKAGE_PIN AA22      [get_ports {SEGMENT[7]}]
set_property IOSTANDARD  LVCMOS33  [get_ports {SEGMENT[7]}]

set_property PACKAGE_PIN AD21      [get_ports {AN[0]}]
set_property IOSTANDARD  LVCMOS33  [get_ports {AN[0]}]
```

```
set_property PACKAGE_PIN AC21      [get_ports {AN[1]}]
set_property IOSTANDARD  LVCMOS33  [get_ports {AN[1]}]

set_property PACKAGE_PIN AB21      [get_ports {AN[2]}]
set_property IOSTANDARD  LVCMOS33  [get_ports {AN[2]}]

set_property PACKAGE_PIN AC22      [get_ports {AN[3]}]
set_property IOSTANDARD  LVCMOS33  [get_ports {AN[3]}]

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

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

set_property PACKAGE_PIN AA13      [get_ports {SW[2]}]
set_property IOSTANDARD  LVCMOS15  [get_ports {SW[2]}]

set_property PACKAGE_PIN AA12      [get_ports {SW[3]}]
set_property IOSTANDARD  LVCMOS15  [get_ports {SW[3]}]

set_property PACKAGE_PIN Y13       [get_ports {SW[4]}]
set_property IOSTANDARD  LVCMOS15  [get_ports {SW[4]}]

set_property PACKAGE_PIN Y12       [get_ports {SW[5]}]
set_property IOSTANDARD  LVCMOS15  [get_ports {SW[5]}]

set_property PACKAGE_PIN AD11      [get_ports {SW[6]}]
set_property IOSTANDARD  LVCMOS15  [get_ports {SW[6]}]

set_property PACKAGE_PIN AD10      [get_ports {SW[7]}]
set_property IOSTANDARD  LVCMOS15  [get_ports {SW[7]}]
```

⑧ 下板，按其功能设计要求进行验证。得到如下的结果。该结果与设计要求和预期相符。由于未设置防抖动，每次按按钮时对应数字可能会增加好几位，这是正常现象。

开关控制数码管和小数点的消隐		
		
按下按钮，对应数码管示数增加		
		

五、 实验结果与分析

（请见上方分析）

六、 讨论、心得

在本次实验中，我使用画图实现了 4 位四输入多路选择器的设计，并成功通过自己设计的仿真代码运行仿真，验证其正确性。在进行记分板设计的过程中，由于我有使用 Vivado 的基础和经验，我全程使用 Vivado+Verilog 代码编写的形式完成了这一部分。我对动态扫描同步输出模块和通用计数分频模块的原理和功能有了更为直观的了解，进一步掌握了分层次、模块化设计的方法。