同济大学计算机系

数字逻辑课程实验报告

学　　号 2353900

姓　　名 汪嘉晨

专　　业 计算机科学与技术（精英班）

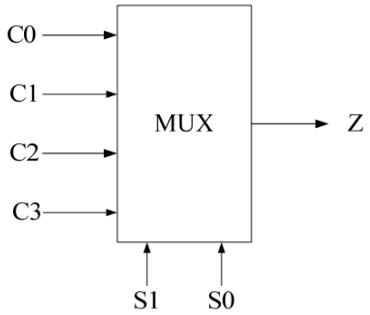
授课老师 郭玉臣

一、实验内容

在本次实验中，我们将使用 Verilog HDL 语言实现数据选择器和数据分配器的设计和仿真。实验分为三个部分：

1.数据选择器实验

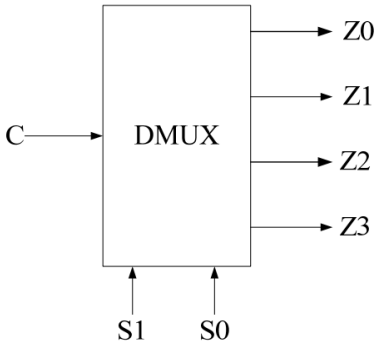
数据选择器（MUX）是一种多路输入、单路输出的标准化逻辑构件。所要建模的 4 选 1 数据选择器及其真值表如图 6.2.1 所示。选择器的开关由两根控制线 s0 和 s1 的编码控制，选择 4 路输入中的一路作为输出。输出 z 的逻辑值将和被选中的输入逻辑值相同。



选择输入		数据输入				输出
S1	S0	c0	c1	c2	c3	Z
0	0	c0	×	×	×	c0
0	1	×	c1	×	×	c1
1	0	×	×	c2	×	c2
1	1	×	×	×	c3	c3

2.数据分配器实验

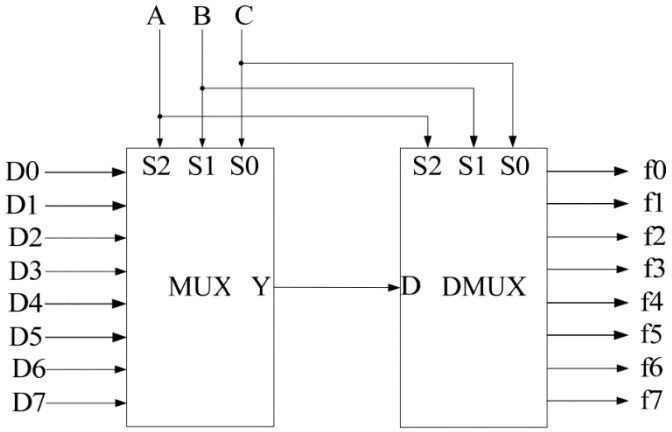
数据分配器（DMUX）的功能与多路选择器相反，它是一种单路输入、多路输出的逻辑构件。图 6.2.2 为 1 线-4 线数据分配器的功能框图，表 3.2 为其真值表。图 3.2 中 c 为数据输入端，s1，s0 为选择控制输入端，z0~z3 为数据输出端。



选择输入		输出			
S1	S0	Z0	Z1	Z2	Z3
0	0	C	1	1	1
0	1	1	C	1	1
1	0	1	1	C	1
1	1	1	1	1	C

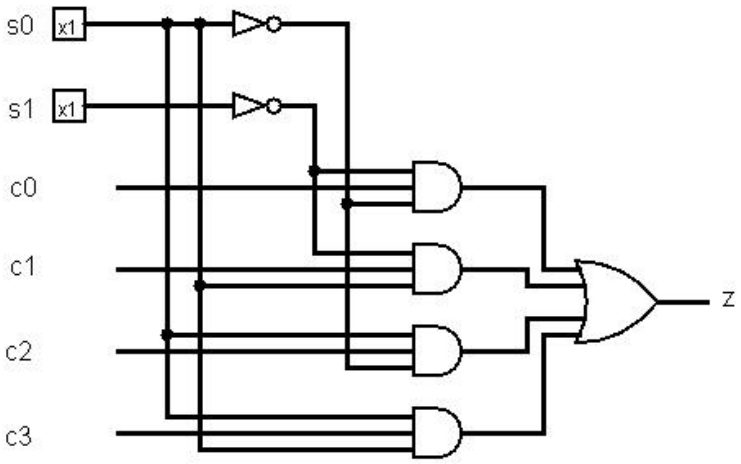
3.8 路数据传输实验

在数据选择器和数据分配器实验基础上实现图 6.2.3 中 8 路数据传输模块的建模。数据的传输由输入控制端 ABC 的编码决定，例如当 ABC=101 时，实现 D5->f5 的数据传输。

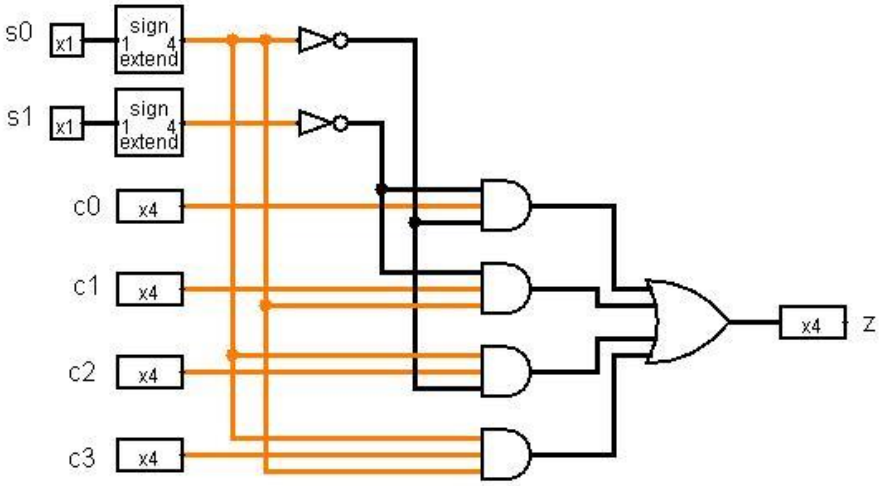


二、硬件逻辑图

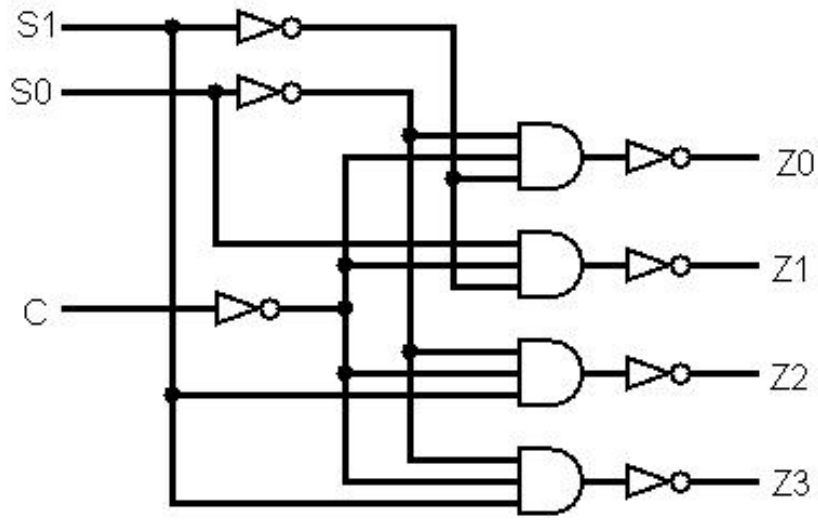
1. 一位四选一数据选择器



2.四位四选一数字选择器



3.1 线-4 线数据分配器



三、模块建模

1.数据选择器实验

```
module selector41(
    input [3:0] iC0, // 四位输入信号 c0
    input [3:0] iC1, // 四位输入信号 c1
    input [3:0] iC2, // 四位输入信号 c2
    input [3:0] iC3, // 四位输入信号 c3
    input iS1, // 选择信号 s1
    input iS0, // 选择信号 s0
    output reg [3:0] oZ // 四位输出信号 z
);

always @(*) begin
    case ({iS1, iS0})
        2'b00: oZ = iC0;
        2'b01: oZ = iC1;
        2'b10: oZ = iC2;
        2'b11: oZ = iC3;
        default: oZ = 4'b0000; // 默认输出
    endcase
end

endmodule
```

2.数据分配器实验

```
module de_selector14(
    input iC,    // 输入信号 C
    input iS1,   // 选择信号 S1
    input iS0,   // 选择信号 S0
    output oZ0,  // 输出信号 Z0
    output oZ1,  // 输出信号 Z1
    output oZ2,  // 输出信号 Z2
    output oZ3   // 输出信号 Z3
);

assign oZ0 = (~iS1 & ~iS0 & iC) | (iS1 | iS0);
assign oZ1 = (~iS1 & iS0 & iC) | (iS1 | ~iS0);
assign oZ2 = (iS1 & ~iS0 & iC) | (~iS1 | iS0);
assign oZ3 = (iS1 & iS0 & iC) | (~iS1 | ~iS0);

endmodule
```

3.8 路数据传输实验

```
module transmission8(
    input [7:0] iData, // 输入信号 D7~D0
    input A,           // 选择信号 S2
    input B,           // 选择信号 S1
    input C,           // 选择信号 S0
    output [7:0] oData // 输出信号 f0~f7
);

// 根据选择信号 A, B, C 的不同组合来设置输出信号 oData 的每一位
assign oData[0] = (~A & ~B & ~C & iData[0]) | (A | B | C);
   // 当 A, B, C 都为 0 时，传输 iData[0]
assign oData[1] = (~A & ~B & C & iData[1]) | (A | B | ~C);
   // 当 A, B 为 0，C 为 1 时，传输 iData[1]
assign oData[2] = (~A & B & ~C & iData[2]) | (A | ~B | C);
   // 当 A 为 0，B 为 1，C 为 0 时，传输 iData[2]
assign oData[2] = (~A & B & ~C & iData[2]) | (A | ~B | C);
   // 当 A 为 0，B 为 1，C 为 0 时，传输 iData[2]
assign oData[3] = (~A & B & C & iData[3]) | (A | ~B | ~C);
   // 当 A 为 0，B 为 1，C 为 1 时，传输 iData[3]
assign oData[4] = (A & ~B & ~C & iData[4]) | (~A | B | C);
    // 当 A 为 1，B 为 0，C 为 0 时，传输 iData[4]
assign oData[5] = (A & ~B & C & iData[5]) | (~A | B | ~C);
    // 当 A 为 1，B 为 0，C 为 1 时，传输 iData[5]
assign oData[6] = (A & B & ~C & iData[6]) | (~A | ~B | C);
    // 当 A 为 1，B 为 1，C 为 0 时，传输 iData[6]
assign oData[7] = (A & B & C & iData[7]) | (~A | ~B | ~C);
   // 当 A, B, C 都为 1 时，传输 iData[7]

endmodule
```

四、测试模块建模

1.数据选择器实验

```
module tb_selector41;

    // 测试平台中的信号
    reg [3:0] iC0;
    reg [3:0] iC1;
    reg [3:0] iC2;
    reg [3:0] iC3;
    reg iS1;
```

```
    reg iS0;
    wire [3:0] oZ;

    // 实例化被测试模块
    selector41 uut (
        .iC0(iC0),
        .iC1(iC1),
        .iC2(iC2),
        .iC3(iC3),
        .iS1(iS1),
        .iS0(iS0),
        .oZ(oZ)
    );

    // 初始化信号并生成测试向量
    initial begin
        // 初始化输入信号
        iC0 = 4'b0001;
        iC1 = 4'b0010;
        iC2 = 4'b0100;
        iC3 = 4'b1000;
        iS1 = 0;
        iS0 = 0;

        // 监视输出信号
        $monitor("Time = %0t, iS1 = %b, iS0 = %b, oZ = %b", $time, iS1, iS0, oZ);

        // 生成测试向量
        #10 iS0 = 1; // 选择 iC1
        #10 iS1 = 1; iS0 = 0; // 选择 iC2
        #10 iS0 = 1; // 选择 iC3
        #10 $finish; // 结束仿真
    end

endmodule
```

2.数据分配器实验

```
module tb_de_selector14;
    // 测试平台中的信号
    reg iC;
    reg iS1;
    reg iS0;
    wire oZ0;
```

```
    wire oZ1;
    wire oZ2;
    wire oZ3;

    // 实例化被测试模块
    de_selector14 uut (
        .iC(iC),
        .iS1(iS1),
        .iS0(iS0),
        .oZ0(oZ0),
        .oZ1(oZ1),
        .oZ2(oZ2),
        .oZ3(oZ3)
    );

    // 初始化信号并生成测试向量
    initial begin
        // 初始化输入信号
        iC = 0;
        iS1 = 0;
        iS0 = 0;

        // 监视输出信号
        $monitor("Time = %0t, iC = %b, iS1 = %b, iS0 = %b, oZ0 = %b, oZ1 = %b, oZ2 = %b, oZ3 = %b",
                 $time, iC, iS1, iS0, oZ0, oZ1, oZ2, oZ3);

        // 生成测试向量
        #10 iC = 1; iS1 = 0; iS0 = 0; // 期望 oZ0 = 1, 其余为 1
        #10 iC = 0; iS1 = 0; iS0 = 1; // 期望 oZ1 = 0, 其余为 1
        #10 iC = 1; iS1 = 1; iS0 = 0; // 期望 oZ2 = 1, 其余为 1
        #10 iC = 0; iS1 = 1; iS0 = 1; // 期望 oZ3 = 0, 其余为 1
        #10 $finish; // 结束仿真
    end

endmodule
```

3.8 路数据传输实验

```
module transmission8_tb;

```

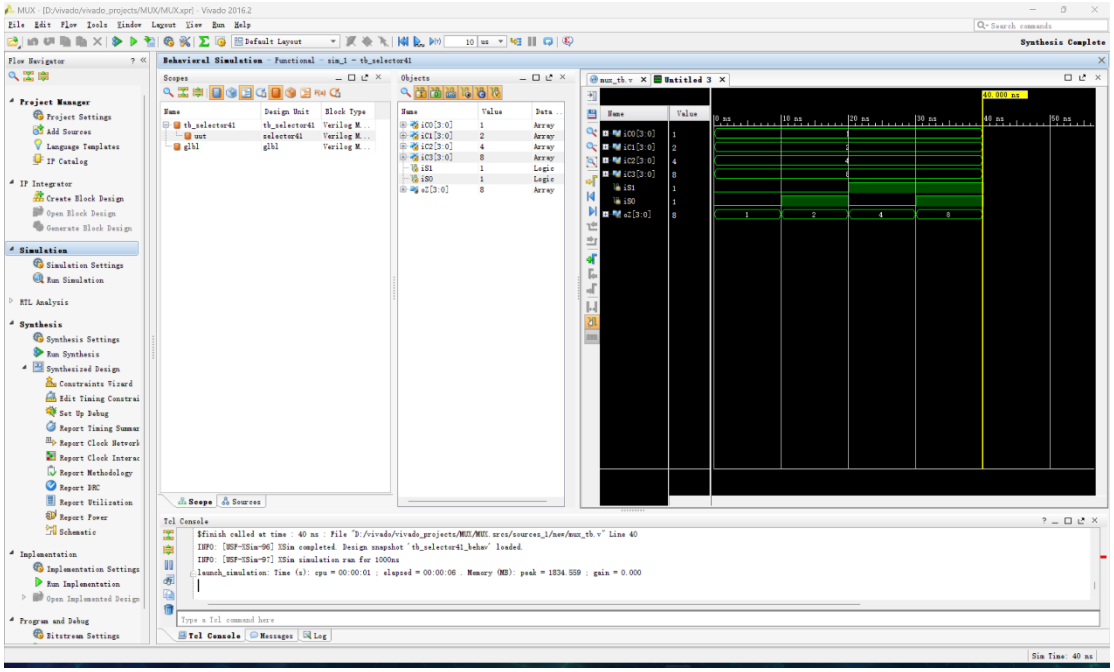
```
    // 输入信号
    reg [7:0] iData;
    reg A, B, C;

    // 输出信号
    wire [7:0] oData;

    // 实例化被测试模块
    transmission8 uut (
        .iData(iData),
        .A(A),
        .B(B),
        .C(C),
        .oData(oData)
    );

    // 初始化输入信号
    initial begin
        // 打印标题
        $display("A B C | iData     | oData");
        $display("-------------------------");

        // 测试不同的选择信号组合
        iData = 8'b10101010; // 示例输入数据

        // 测试所有选择信号组合
        {A, B, C} = 3'b000; #10;
        $display("%b %b %b | %b | %b", A, B, C, iData, oData);
        {A, B, C} = 3'b001; #10;
        $display("%b %b %b | %b | %b", A, B, C, iData, oData);
        {A, B, C} = 3'b010; #10;
        $display("%b %b %b | %b | %b", A, B, C, iData, oData)
        {A, B, C} = 3'b011; #10;
        $display("%b %b %b | %b | %b", A, B, C, iData, oData)
        {A, B, C} = 3'b100; #10;
        $display("%b %b %b | %b | %b", A, B, C, iData, oData)
        {A, B, C} = 3'b101; #10;
        $display("%b %b %b | %b | %b", A, B, C, iData, oData);
        {A, B, C} = 3'b110; #10;
        $display("%b %b %b | %b | %b", A, B, C, iData, oData)
        {A, B, C} = 3'b111; #10;
        $display("%b %b %b | %b | %b", A, B, C, iData, oData);
        // 结束仿真
        $finish;
```

```
47.     end
48.
49. endmodule
```

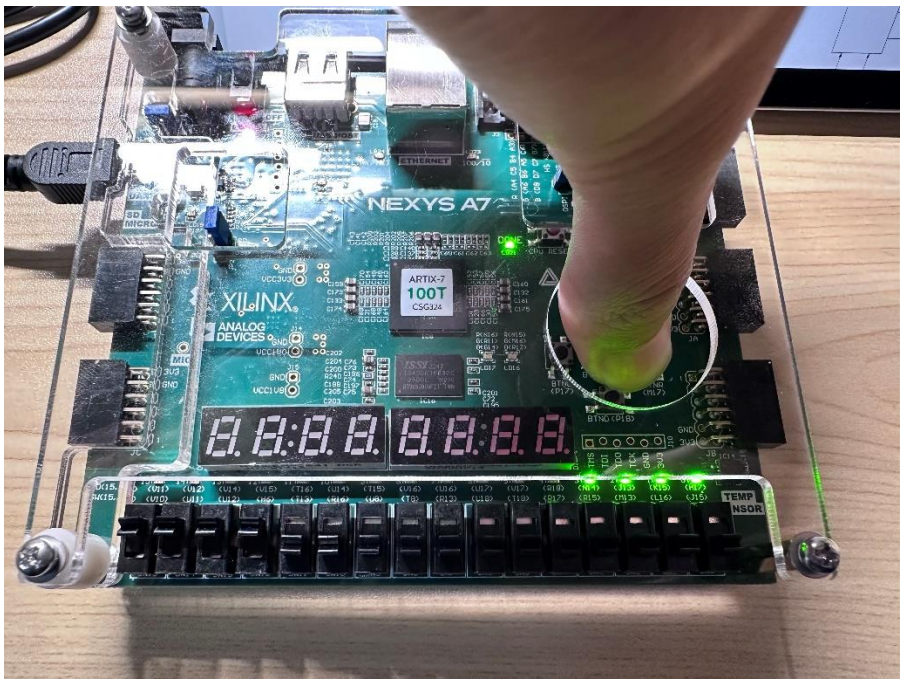
五、实验结果

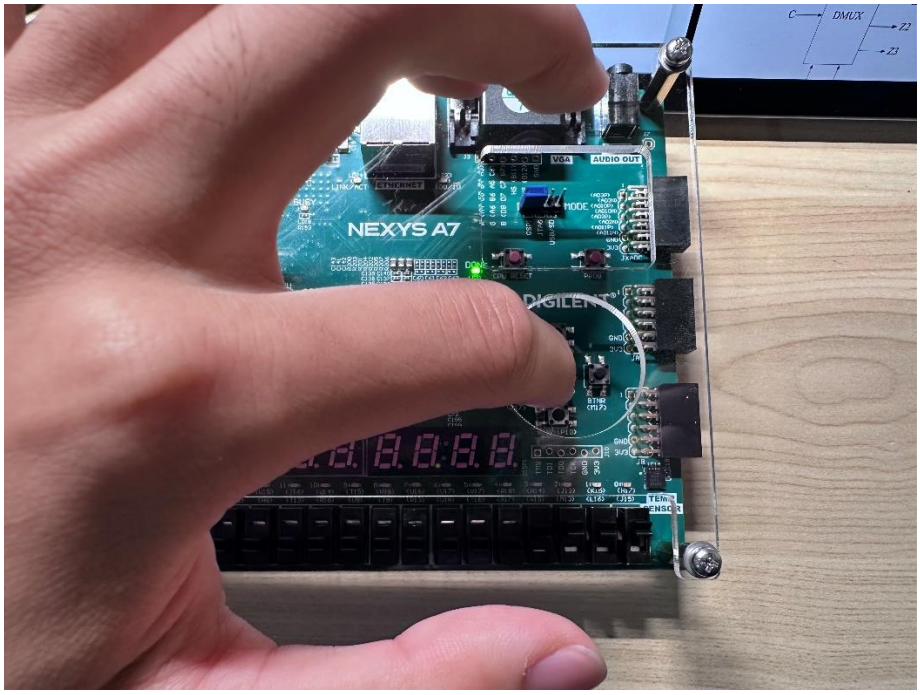
1.数据选择器实验

modelsim 仿真如图所示：在 S1,S0 取值不同情况时，输出端输出不同数据



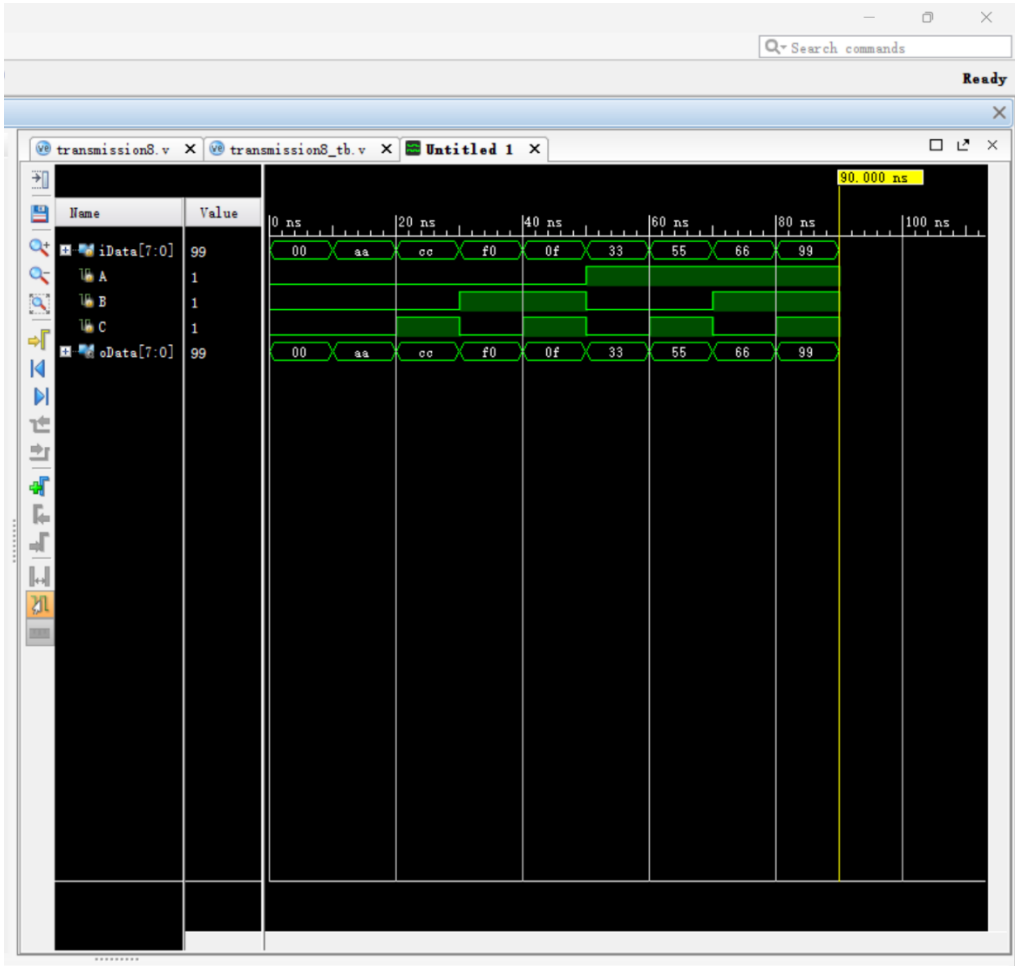
综合下板，通过按压 S1,S0 代表的按钮，可以发现：



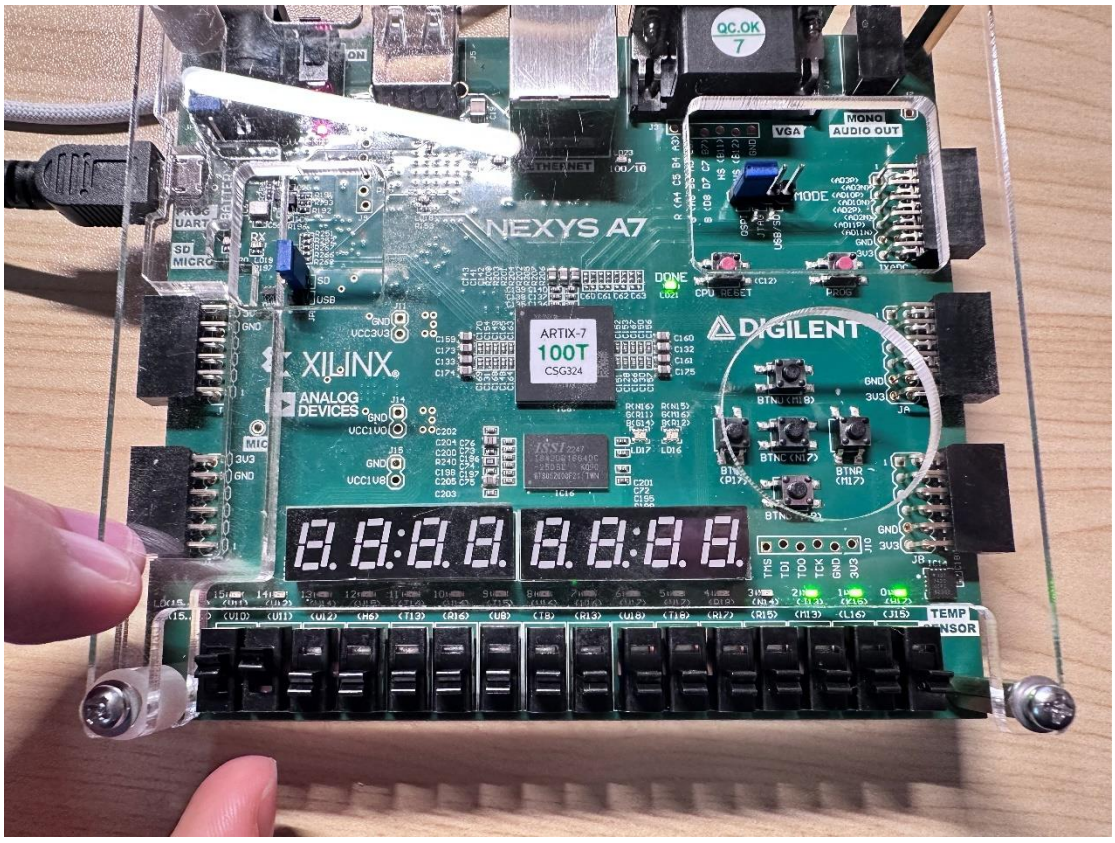


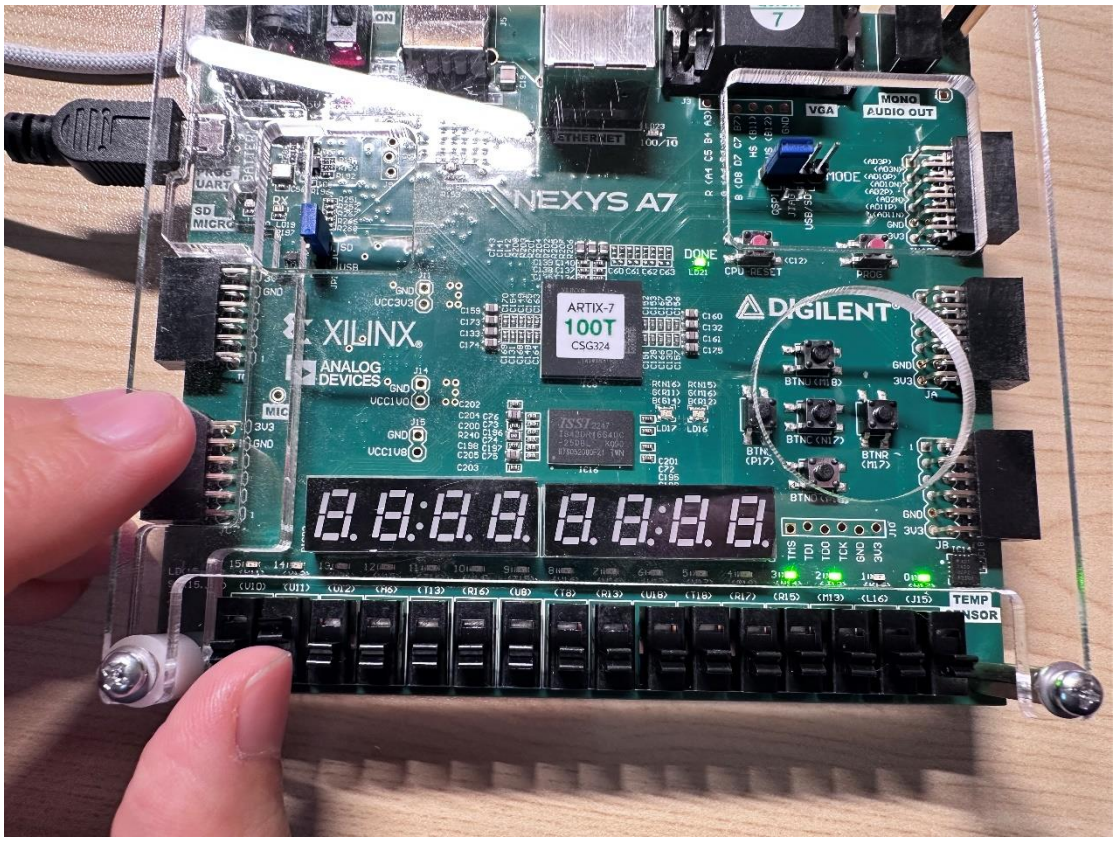
2.数据分配器实验

modelsim 仿真如图所示：



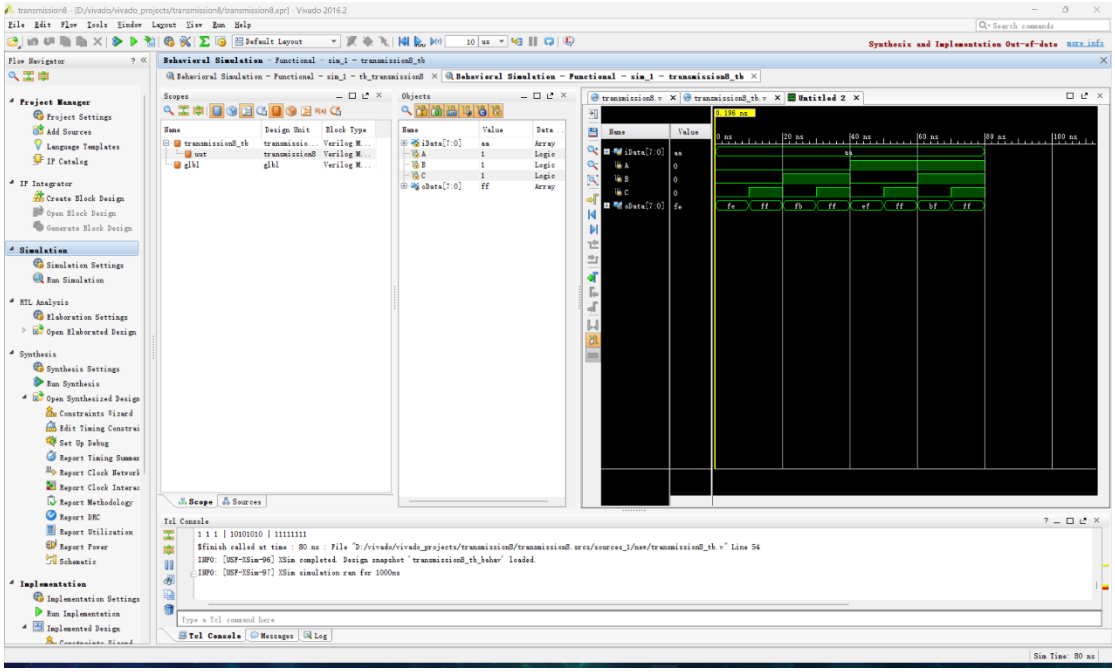
综合下板，可以发现与真值表相同：

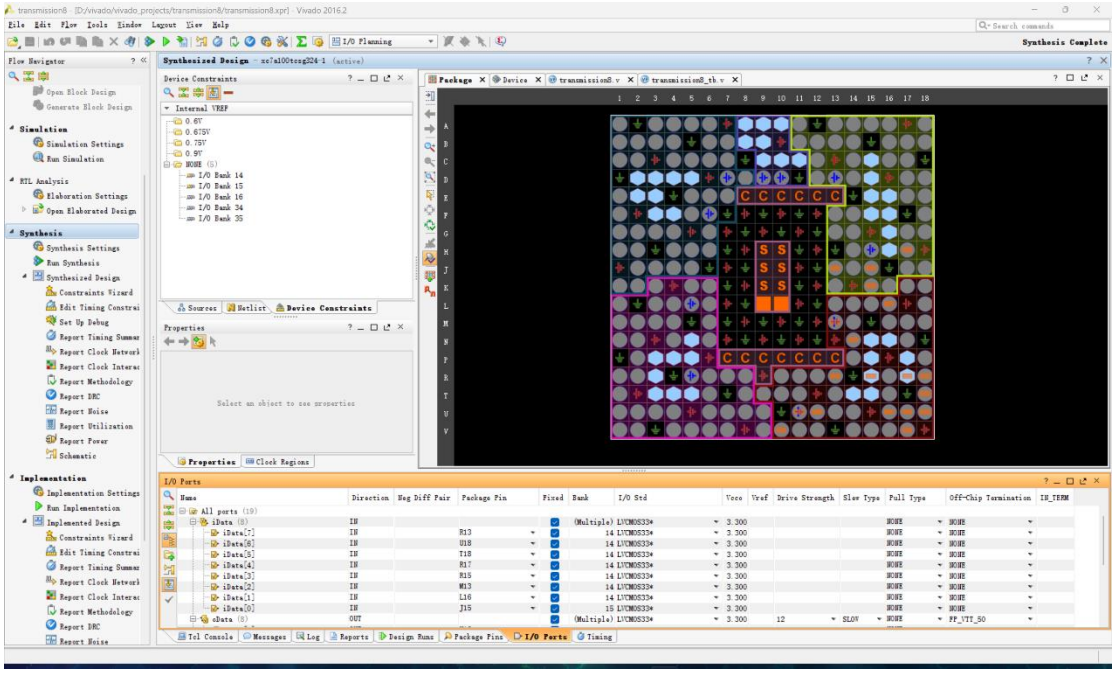




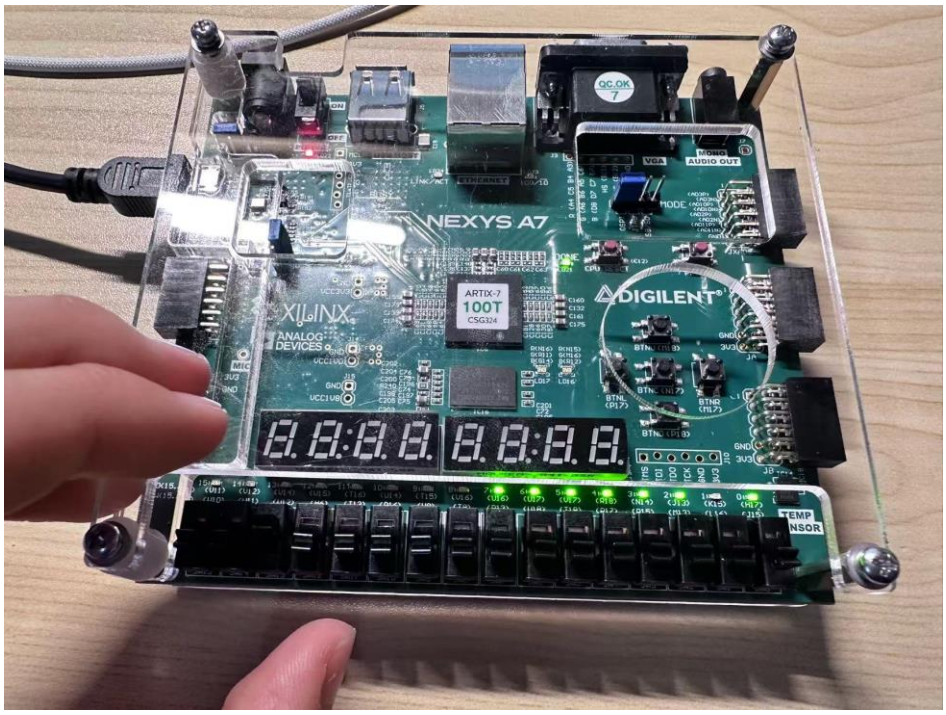
3.8 路数据传输实验

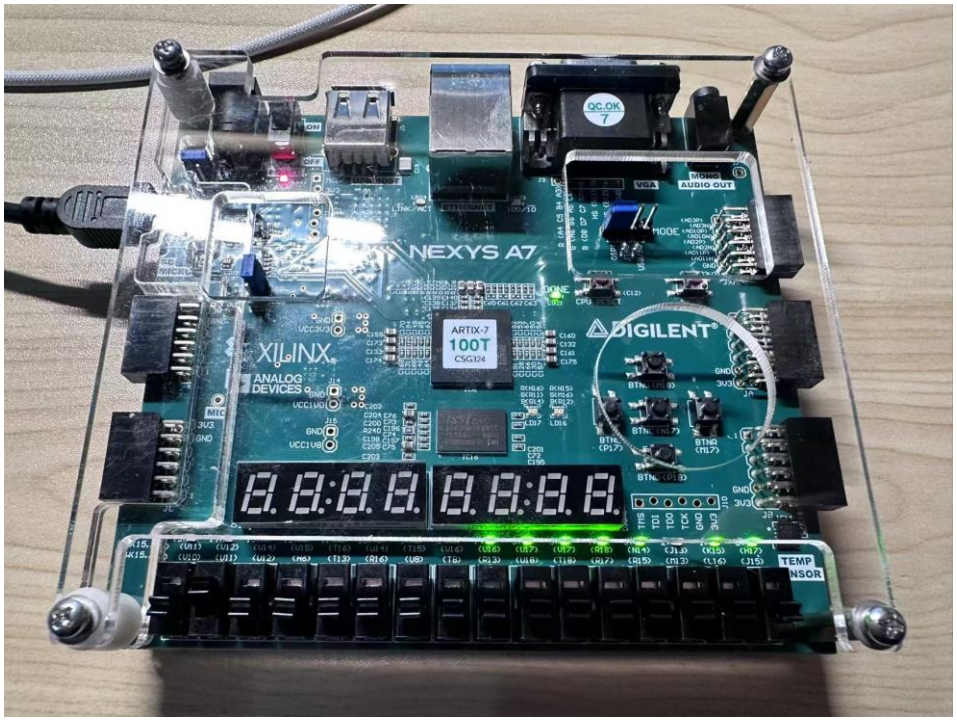
modelsim 仿真如图所示：在 S1,S0 取值不同情况时，输出端输出不同数据





综合下板，通过改变 S0,S1,S2 按钮，可以发现显示不同的内容，与真值表相同：


NEXYS A7
XILINX
ANALOG DEVICES
DIGILENT
ARTIX-7
100T
CSG324
VGA
MONO AUDIO OUT
ETHERNET
QC.OK
TEMP SENSOR


NEXYS A7
XILINX
ANALOG DEVICES
DIGILENT
ARTIX-7
100T
CSG324
VGA
MONO AUDIO OUT
QC.OK
TEMP SENSOR