**1 实验目的和要求**

**1.1 实验目的**

（分点简要说明本次实验需要进行的工作和最终的目的）

1. Design a 2-1 Mux with the Xilinx software.
2. Design it with AND/OR/NOT gate level Verilog description.
3. Design it with Boolean function Verilog description.
4. Design it with Always statement and if-else statement.
5. Design it with always statement and case statement.
6. Design 4-1 Mux by using hierarchy structure of the module 2-1 Mux designed above.
7. Connect the input a, b, s to three switches. Connect they output to an LED. Implement the circuit onto the FPGA board to test and verify that it works correctly.
8. Goal：To learn how to use the Xilinx circuit design software to design and implement digital circuits.

**1.2 实验要求**

（说明本次实验的要求与任务）

Design a 2-1 Mux with the Xilinx software. Connect the input a, b, s to three switches. Connect they output to an LED. Implement the circuit onto the FPGA board to test and verify that it works correctly.

1)Design it with AND/OR/NOT gate level Verilog description.

2)Design it with Boolean function Verilog description.

3)Design it with Always statement and if-else statement.

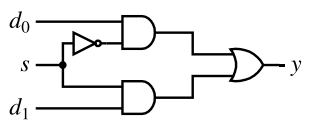
4)Design it with always statement and case statement.

Design 4-1 Mux by using hierarchy structure of the module 2-1 Mux designed above.

**2 实验原理**

（简要说明本次实验的理论，包括但不限于物理、数学或是算法方面的理论，电路原理图、算法框图等示意图也可以在此处给出）

HDL语言和算法原理

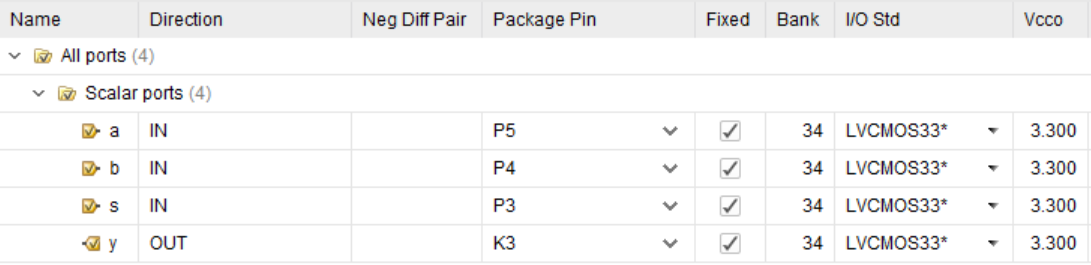
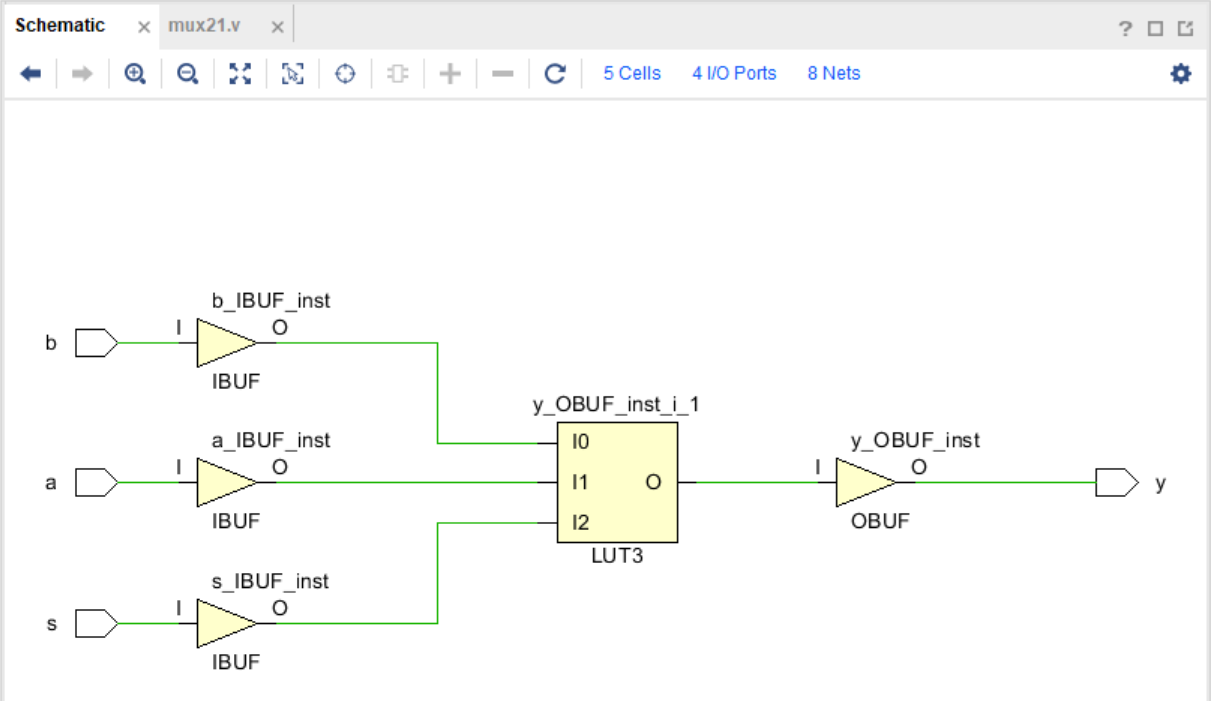
2选1选择器原理图****

**3 实验内容**

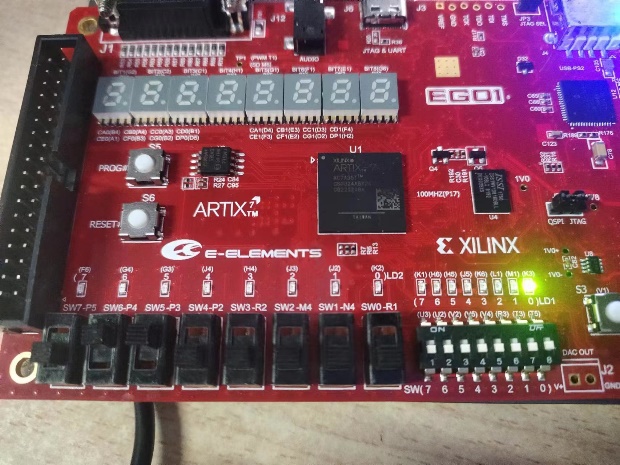
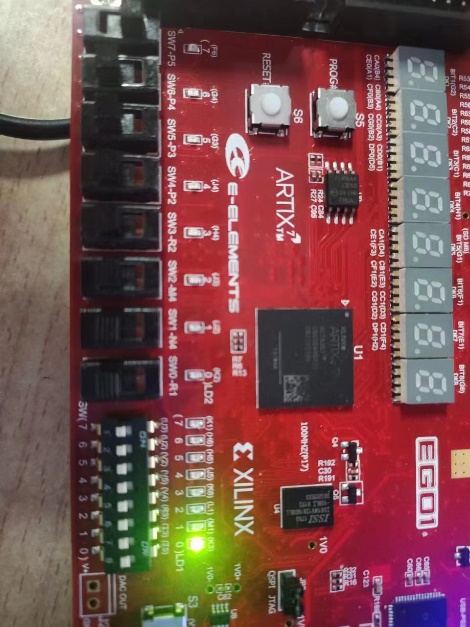
（分点阐述实验步骤）

1. 使用vivado软件，分别从门级，数据流级，两种行为级，编写2选1选择器的HDL语言代码。
2. 进行综合，管脚约束，执行，生成比特流，并烧写到电路板上。
3. 使用电路板，测试2选1选择器。
4. 新建工程，编写一个2选1选择器模块，用结构化的方式，使用编写的模块搭建一个4选1选择器。
5. 进行综合，管脚约束，执行，生成比特流，并烧写到电路板上。使用电路板，测试4选1选择器。

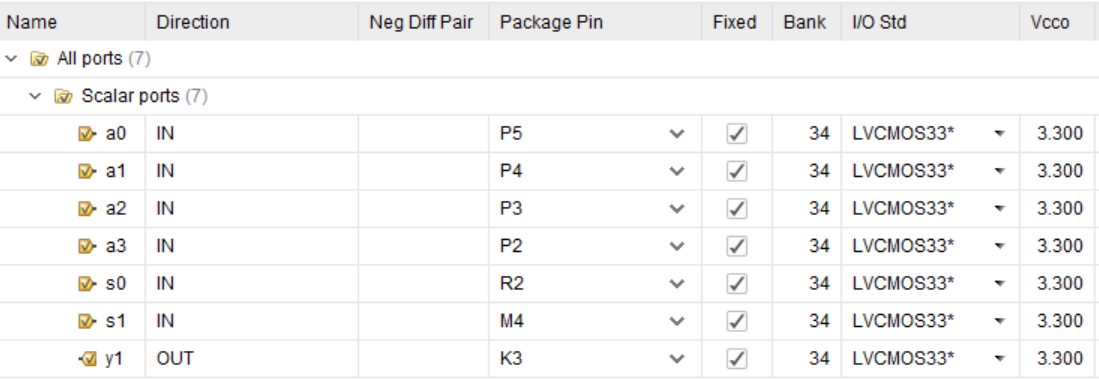
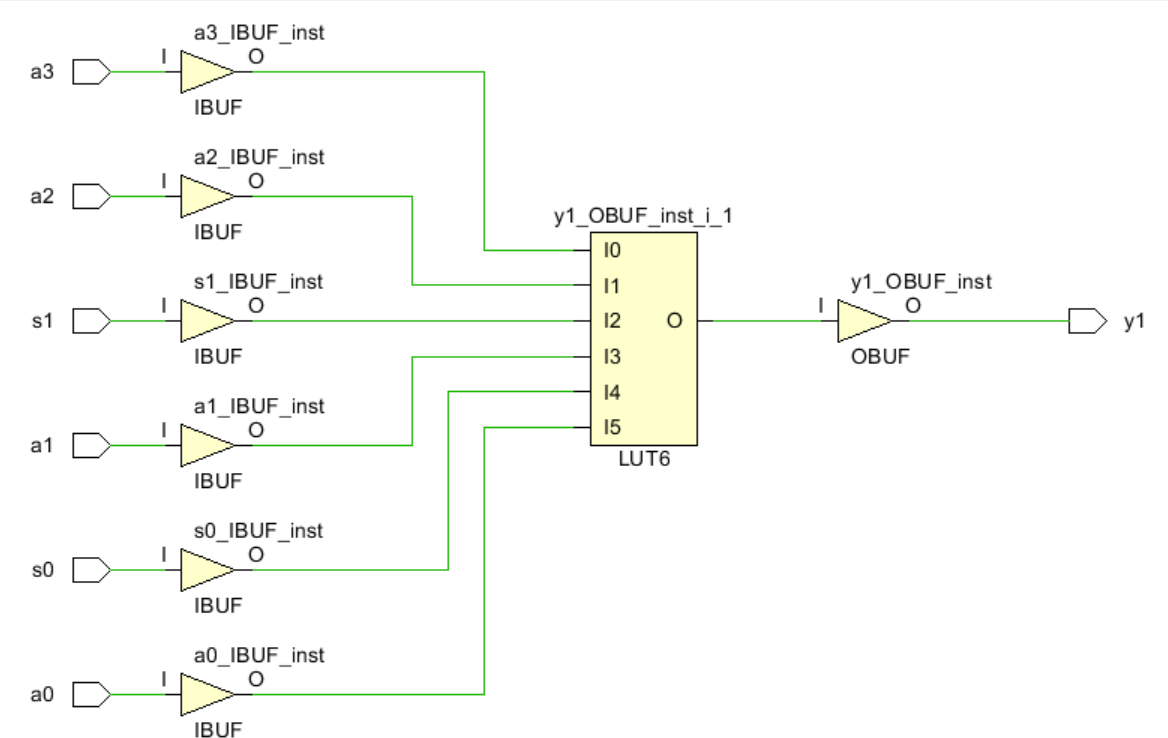
**4 实验结果和分析**



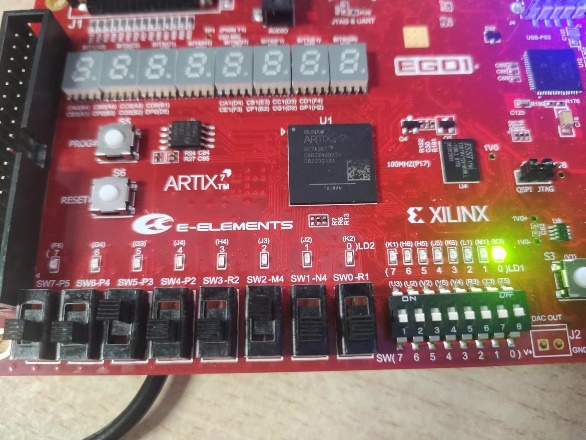
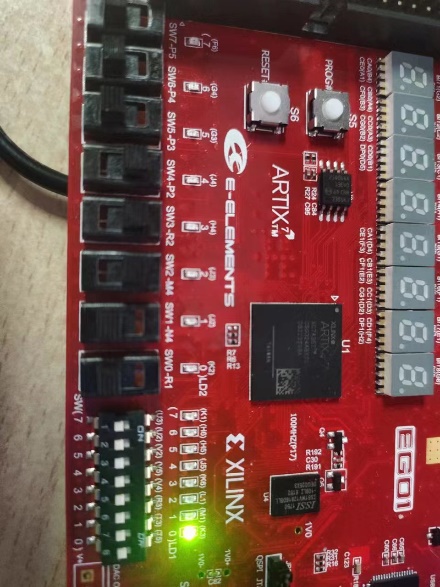
图表 1综合结果与管脚约束



图表 2 2选1选择器的电路板测试效果



图表 3综合效果和管脚约束



图表 4 4选1选择器电路板测试效果

**5 实验结论**

（基于实验结果和分析，得出结论）

成功实现了21选择器和41选择器的设计，运行和测试。

**6 源代码与分析**

（粘贴本次实验使用的源代码，并使用注释的方式进行适当的分析）

module mux21\_gatelevel( //门级设计代码

input a,b,s,

output y

);

wire sn,an,bn;

not U1(sn,s);

and U2(an,a,sn);

and U3(bn,b,s);

or U4(y,an,bn);

endmodule

module mux21\_dateflowlevel( //数据流级设计代码

input a,b,s,

output y

);

assign y=(~s&a)|(s&b);

endmodule

module mux21\_behaviorallevel\_ifelse( //行为级if-else语句设计代码

input a,b,s,

output reg y

);

always @(\*) begin

if( s == 1'b0) y=a;

else y=b;

end

endmodule

module mux21\_behaviorallevel\_case( //行为级case语句设计代码

input a,b,s,

output reg y

);

always @(\*) begin

case(s)

1'b0 :y=a;

1'b1 :y=b;

endcase

end

endmodule

module multiplexer ( //2选1选择器模块单元

input s, d0, d1,

output y

);

wire sn, snd0, sd1;

not U1(sn,s);

and U2(snd0,d0,sn);

and U3(sd1,d1,s);

or U4(y,snd0,sd1);

endmodule

module multiplexer4( //使用三个21选择器模块搭建41选择器

input a0,a1,a2,a3,s0,s1,

output y1

);

wire b1,b2;

multiplexer m0(s0,a0,a1,b1);

multiplexer m1(s0,a2,a3,b2);

multiplexer m2(.s(s1),.d0(b1),.d1(b2),.y(y1));

endmodule