

同济大学计算机系

数字逻辑课程实验报告



学 号 2350752

姓 名 田思宇

专 业 计算机科学与技术

授课老师 张冬冬

一、实验内容

- 1、数据选择器实验
- 2、数据分配器实验
- 3、八路数据传输实验

逻辑表达式：

由于图 6.2.1、6.2.2 给出了真值表
故直接列写表达式

1. 数据选择器：
$$F = C_0 \bar{S}_1 \bar{S}_2 + C_1 \bar{S}_1 S_0 + C_2 S_1 \bar{S}_0 + C_3 S_1 S_2$$
$$= \sum_{i=0}^3 C_i M_i$$

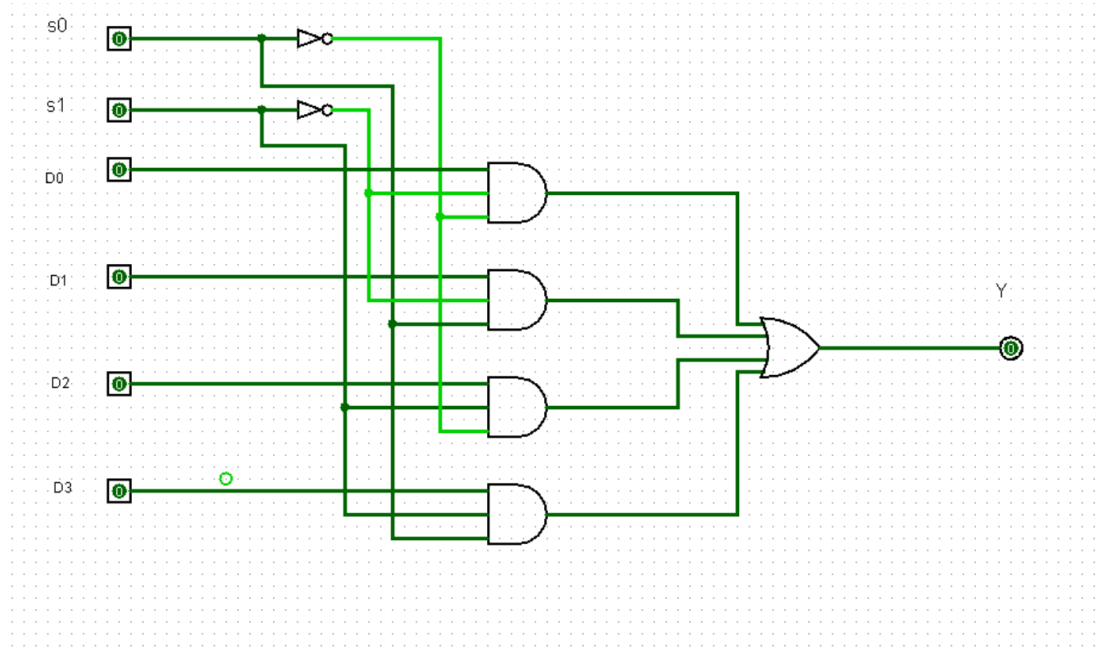
2. 数据分配器：
$$F = Z_0 \bar{S}_1 \bar{S}_2 + Z_1 \bar{S}_1 S_0 + Z_2 S_1 \bar{S}_0 + Z_3 S_1 S_2$$
$$= \sum_{i=0}^3 Z_i M_i$$

二、硬件逻辑图

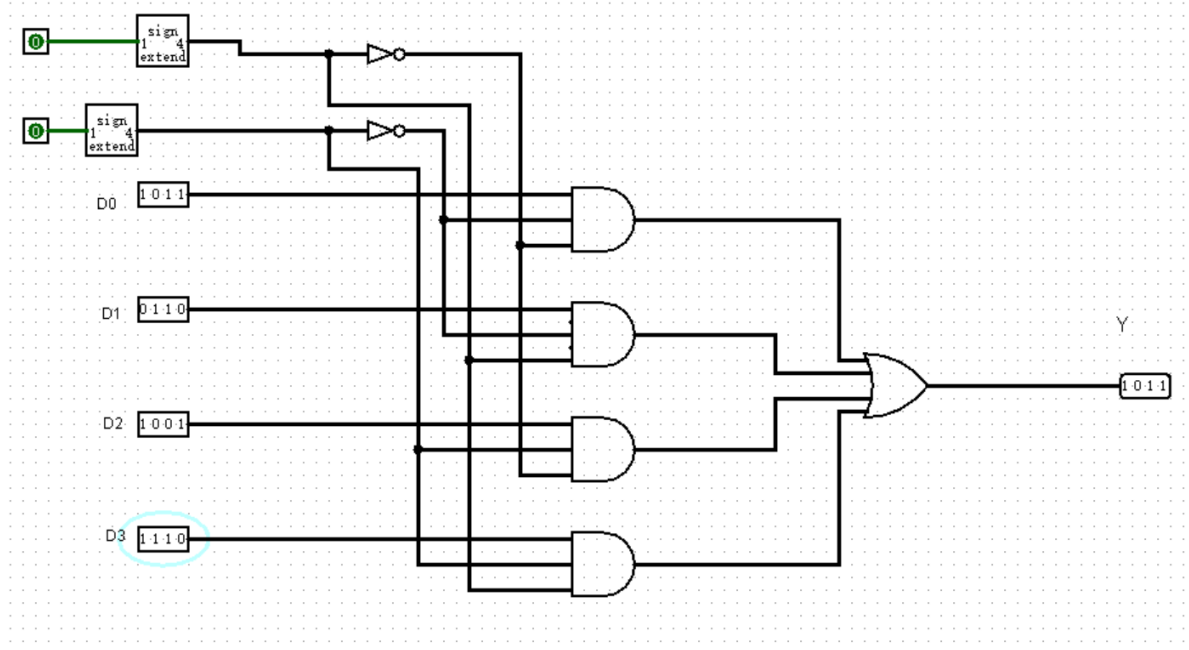
（实验步骤中要求用 logisim 画图的实验，在该部分给出 logisim 原理图，否则该部分在实验报告中不用写）

实验一：

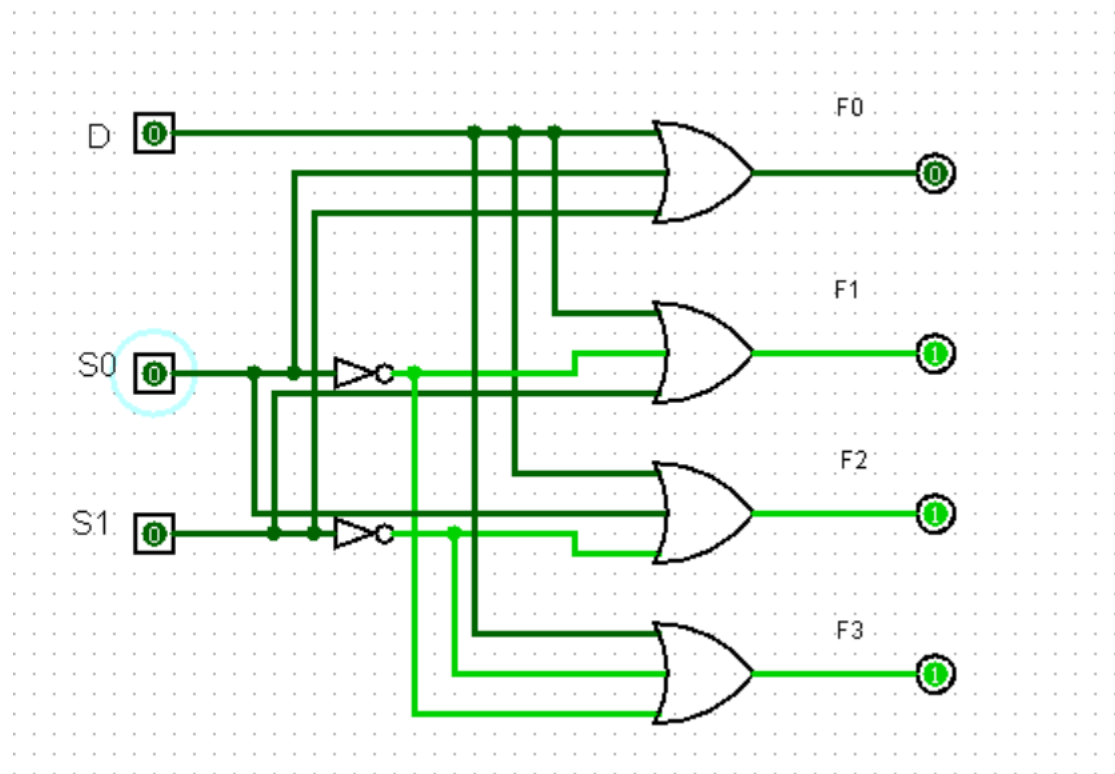
- 1、每个输入是 1 位的四选一选择器的原理图



2、每个输入是 4 位的四选一选择器的原理图



实验二：数据分配器原理图



三、模块建模

（该部分要求对实验中建模的所有模块进行功能描述，并列出各模块建模的verilog 代码）

实验一：

功能描述：这是一个四个输入且每个输入是 4bit 的选择器，定义了四个 4 位 wire 类型 input 代表实际的输入，以及两个 1 位的控制输入信号，然后再定义 4 位输出，两个一位的控制信号组合起来有四种取值可能，一共四个与门（四个输入），每一个与门（每一个输入）对应一个最小项，这样就实现了同一时刻只有一个输入的信号可以进行输出。

```
module selector41(
input [3:0] iC0,
input [3:0] iC1,
input [3:0] iC2,
input [3:0] iC3,
input iS1,
input iS0,
output reg [3:0] oZ
);
reg [1:0] hebing;
always @(iC0,iC1,iC2,iC3,iS1,iS0)
begin
hebing={iS1,iS0};
```

```

case(hebing)
2'b00: oZ=iC0;
2'b01: oZ=iC1;
2'b10: oZ=iC2;
2'b11: oZ=iC3;
default: oZ=4'bxxxx;
endcase
end
endmodule

```

实验二：

功能描述：这是一个数据分配器模块，将一路输入分配给四路输出中的一个。首先定义了输入 ic，以及四个输出 oz0-oz3，is1 和 is2 属于控制信号，代码没有采用行为及描述，而是根据 logisim 的原理图来进行实例化，更加贴合底层逻辑，两位的控制信号对应四个最小项，通过四个三输入的与门来实现分配器的功能（输出低电平有效）。

```

module de_selector14(
input iC,
input iS1,
input iS0,
output reg oZ0,
output reg oZ1,
output reg oZ2,
output reg oZ3
);
always @(*)
begin
oZ0=iC|iS1|iS0;
oZ1=iC|iS1|~iS0;
oZ2=iC|~iS1|iS0;
oZ3=iC|~iS1|~iS0;
end
endmodule

```

实验三：

功能描述：本模块实现的功能是八路数据传输，首先定义八位的输入，然后定义 abc 三个控制信号，以及八位的输出，采用行为级描述，abc 的最小项有八个，刚好可以控制八路输入中的一路送到八路输出中的一路，其余七路输出全为 1。

```

module transmission8(
    input [7:0] iData,
    input A, B, C,
    output reg [7:0] oData
);
always @(A, B, C, iData)
begin

```

```

oData = 8'b11111111;
case ({A, B, C})
    3'b000: oData = {7'b1111111,iData[0]};
    3'b001: oData = {6'b111111,iData[1],1'b1};
    3'b010: oData = {5'b11111,iData[2], 2'b11};
    3'b011: oData = {4'b1111,iData[3], 3'b111};
    3'b100: oData = {3'b111,iData[4], 4'b1111};
    3'b101: oData = {2'b11,iData[5], 5'b11111};
    3'b110: oData = {1'b1,iData[6], 6'b111111};
    3'b111: oData = {iData[7],7'b1111111};
    default: oData = 8'b11111111;
endcase
end
endmodule

```

四、测试模块建模

实验一：

```

`timescale 1ns/1ns
module selector41_tb;
reg [3:0] IC0,IC1,IC2,IC3;
wire [3:0] OZ;
reg IS0,IS1;
initial
begin
    IC0=4'b0011;
    IC1=4'b0111;
    IC2=4'b1001;
    IC3=4'b0110;
end
initial
begin
    IS0=0;
    IS1=0;
    #100;
    IS0=1;
    IS1=0;
    #100;
    IS0=0;
    IS1=1;
    #100;
    IS0=1;
    IS1=1;

```

```

end
selector41
selector41_tb(.iC0(IC0),.iC1(IC1),.iC2(IC2),.iC3(IC3),.oZ(OZ),.iS0(IS0),.iS1(IS1));
endmodule

```

实验二：

```

`timescale 1ns/1ns
module distribute_tb;
reg iC;
reg iS1;
reg iS0;
wire oZ0;
wire oZ1;
wire oZ2;
wire oZ3;
initial
begin
iC=0;
iS1=0;
iS0=0;
#50 iS0=1;
#50 iS0=0;iS1=1;
#50 iS0=1;
end
de_selector14
distribute_tb14(iC,iS1,iS0,oZ0,oZ1,oZ2,oZ3);
endmodule

```

实验三：

```

`timescale 1ns / 1ps
module transmission8_tb;
reg [7:0] iData;
reg A;
reg B;
reg C;
wire [7:0] oData;
transmission8 uut (
    .iData(iData),
    .A(A),
    .B(B),

```

```

        .C(C),
        .oData(oData)
    );
initial begin
    iData = 8'b000000000;
    A = 0; B = 0; C = 0;
    #10;
    A = 0; B = 0; C = 0;
    iData = 8'b000000000;
    #10;
    A = 0; B = 0; C = 1;
    iData = 8'b000000000;
    #10;
    A = 0; B = 1; C = 0;
    iData = 8'b000000000;
    #10;
    A = 0; B = 1; C = 1;
    iData = 8'b000000000;
    #10;
    A = 1; B = 0; C = 0;
    iData = 8'b000000000;
    #10;
    A = 1; B = 0; C = 1;
    iData = 8'b000000000;
    #10;
    A = 1; B = 1; C = 0;
    iData = 8'b000000000;
    #10;
    A = 1; B = 1; C = 1;
    iData = 8'b000000000;
    #10;
end
endmodule

```

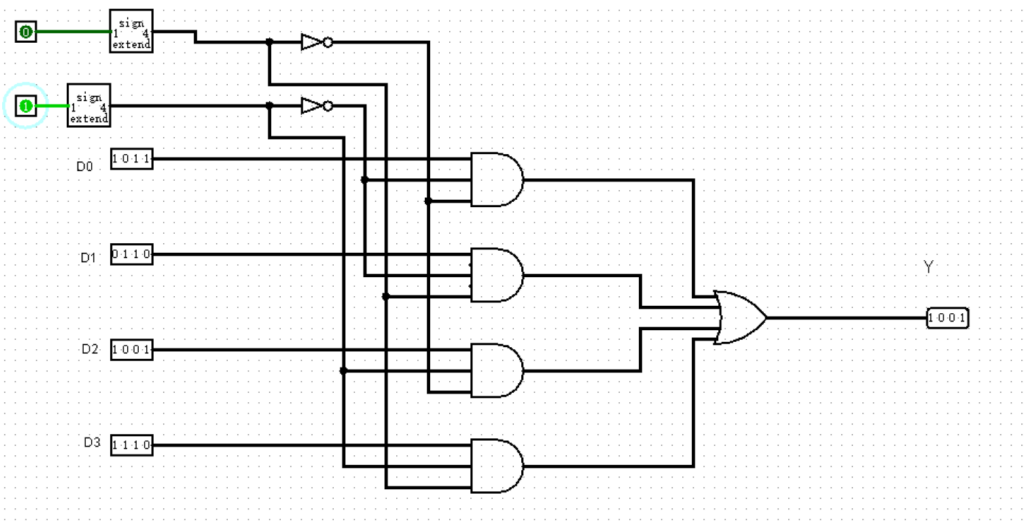
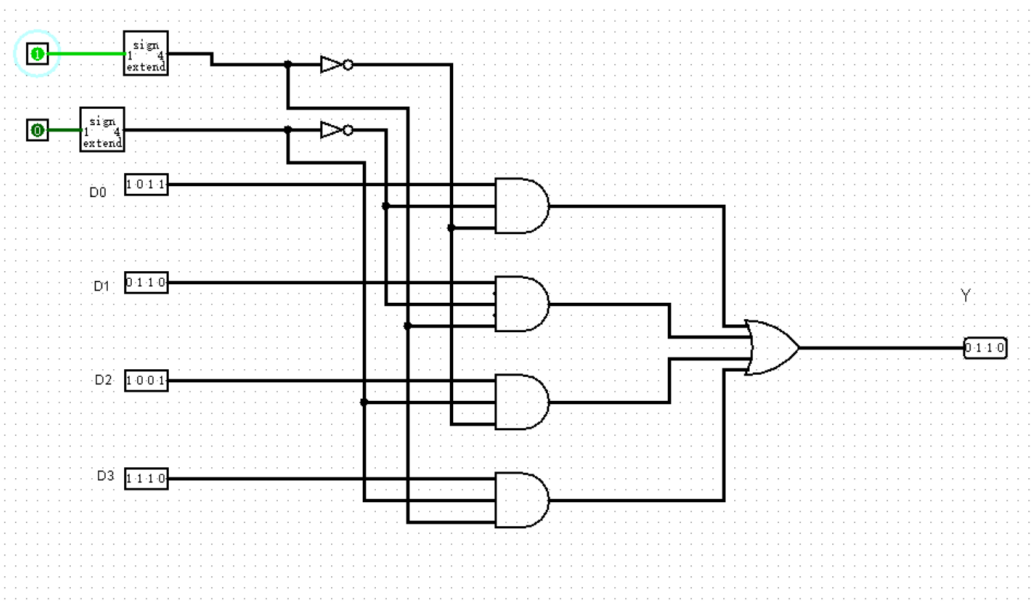
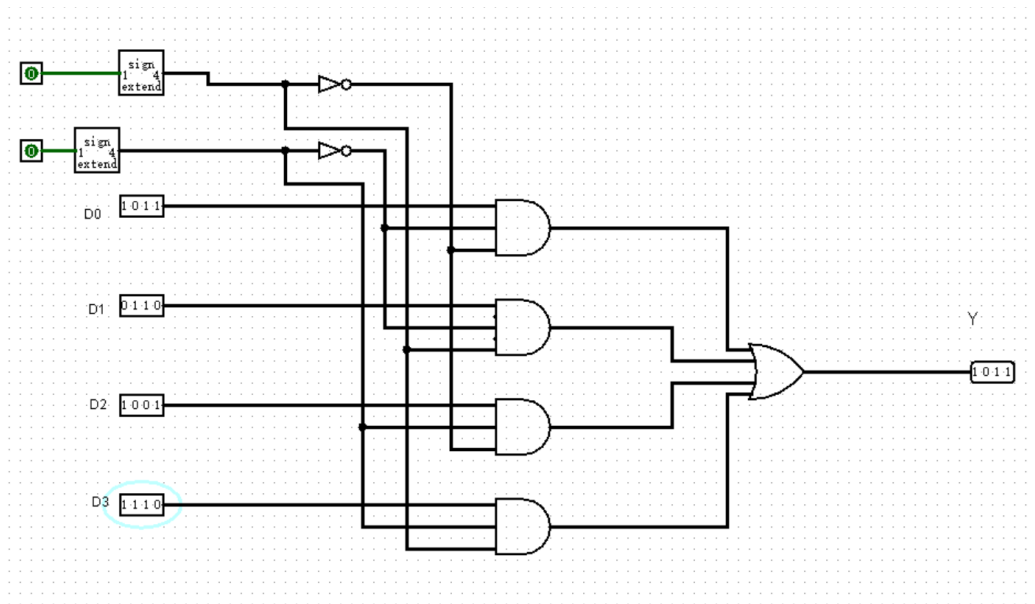
五、实验结果

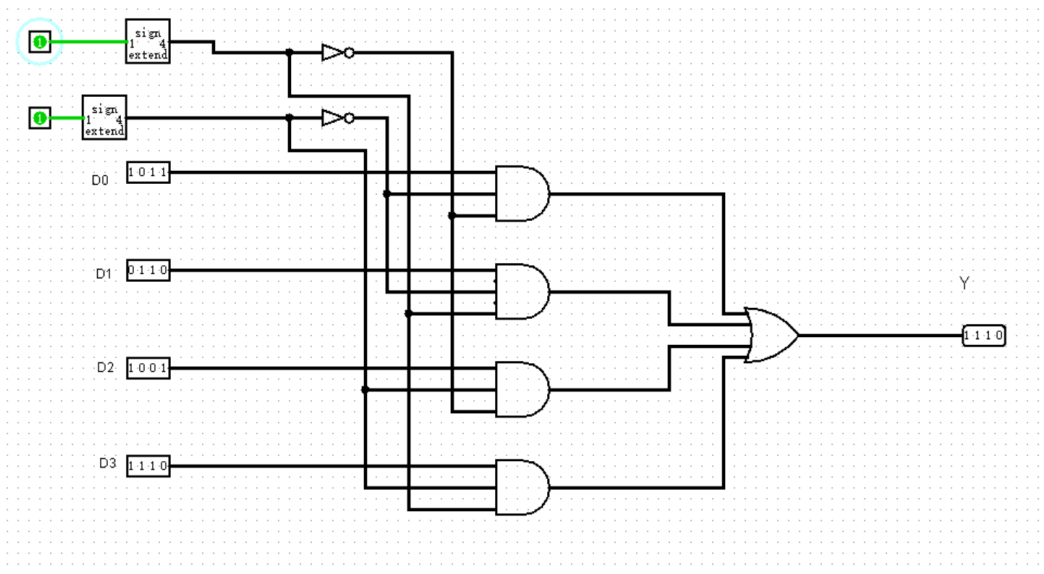
（该部分可截图说明，要求 logisim 逻辑验证图、modelsim 仿真波形图、以及下板后的实验结果贴图（实验步骤中没有下板要求的实验，不需要下板贴图））

实验一：

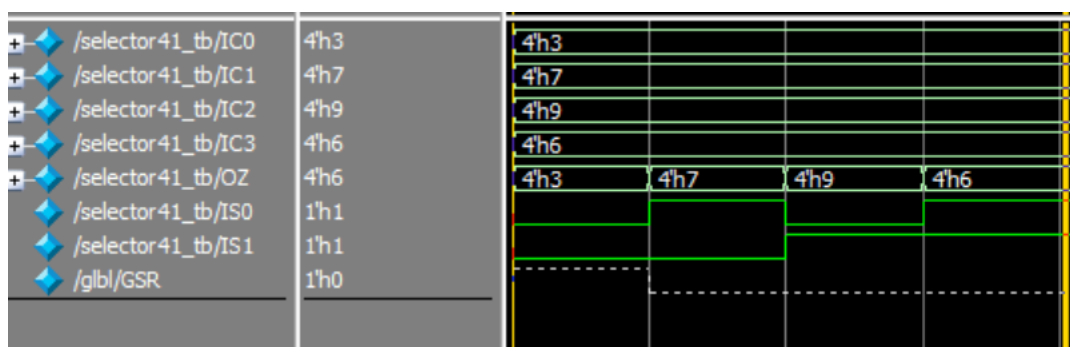
1、逻辑验证图：

这四张图片分别代表控制信号的四种不同取值情况，同时在验证图中设置了四个输入信号都不同，00 可以输出 d0，01 可以 d1，10 可以输出 d2，11 可以输出 d3。观察四个图片可以验证逻辑正确。



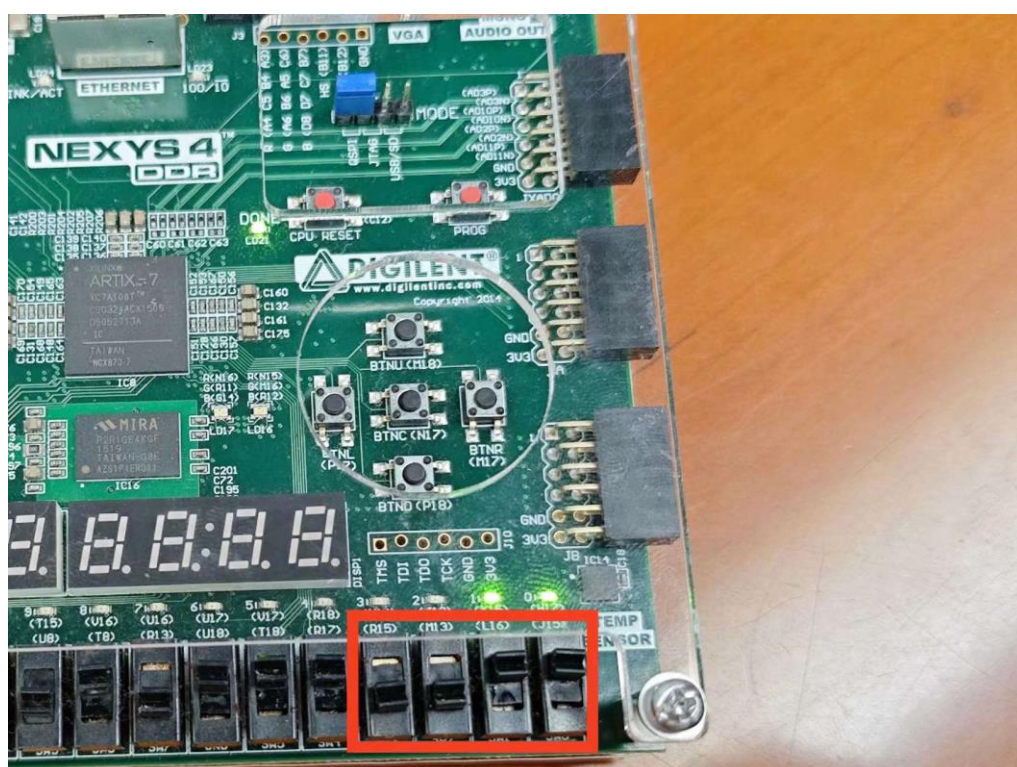


2、仿真波形图：

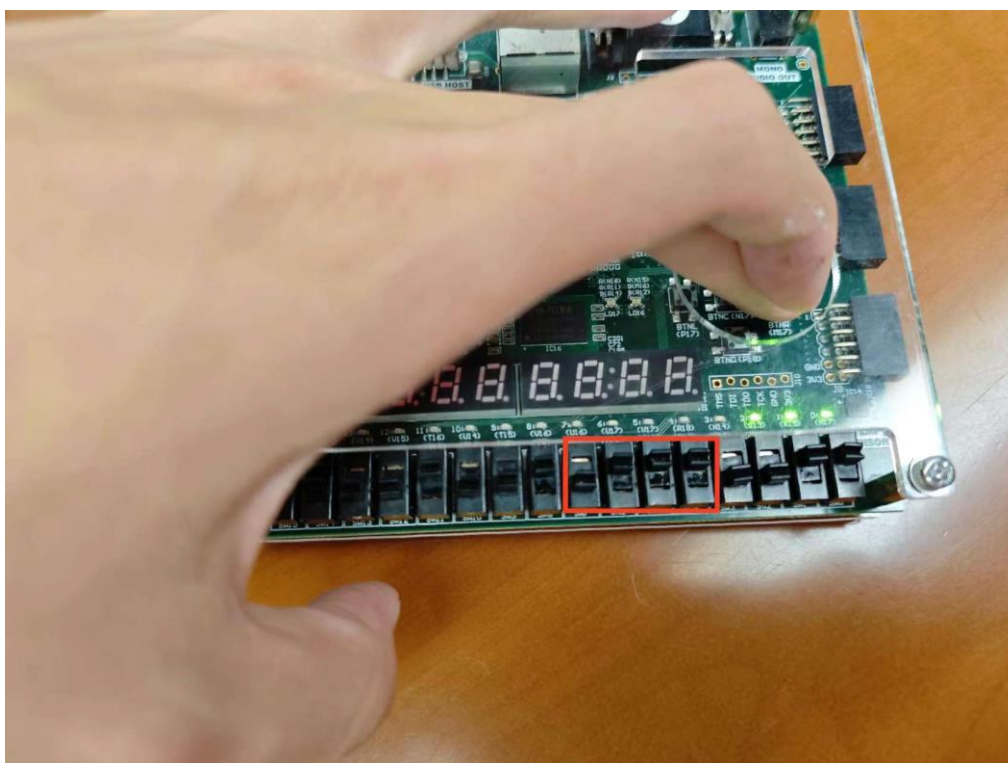


3、下板实验结果：

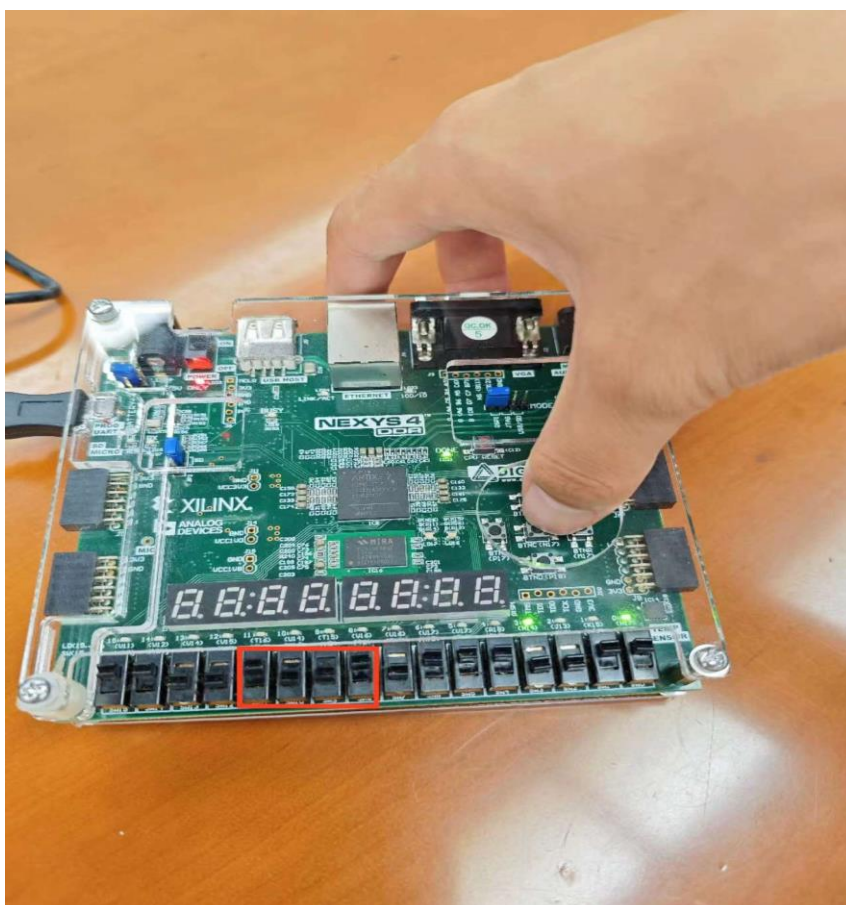
当控制信号为 00 时，第一个门的输入可以输出，如图所示



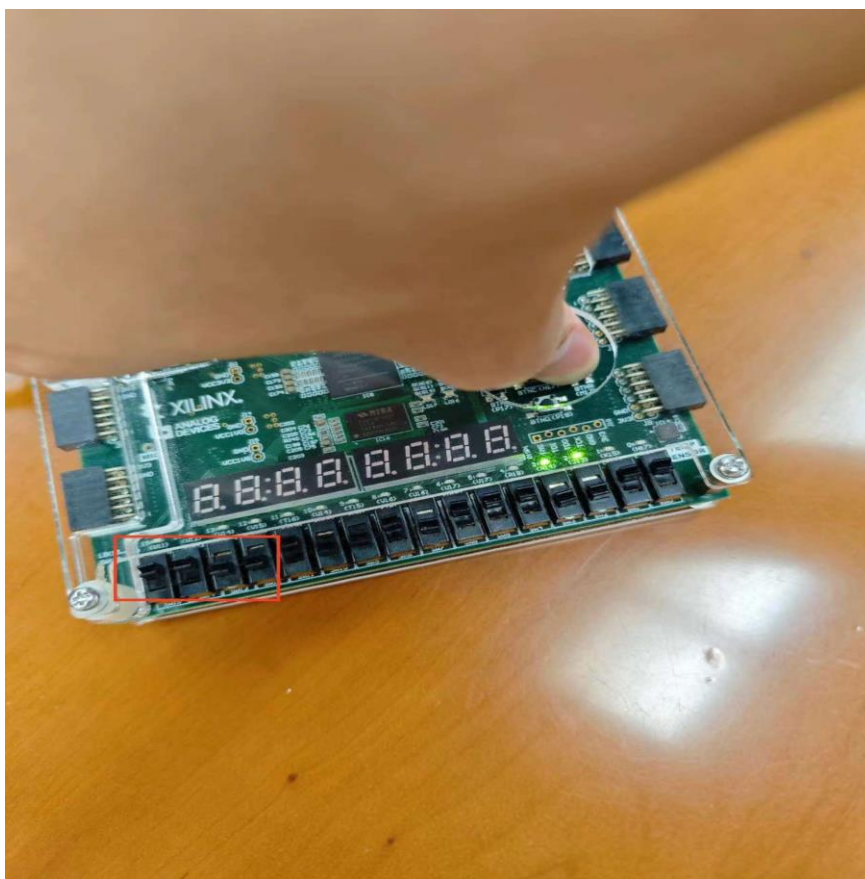
当控制信号为 01 时，第二个门的输入可以输出，如图所示



当控制信号为 10 时，第三个门的输入可以输出，如图所示



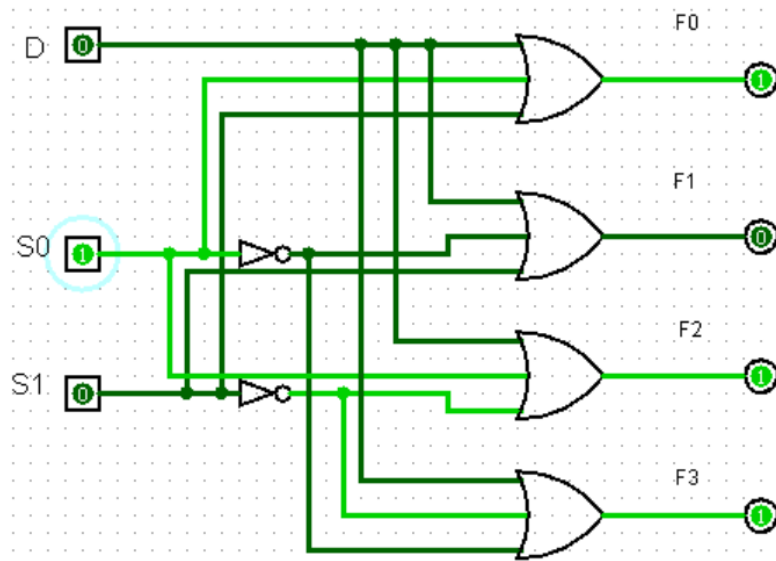
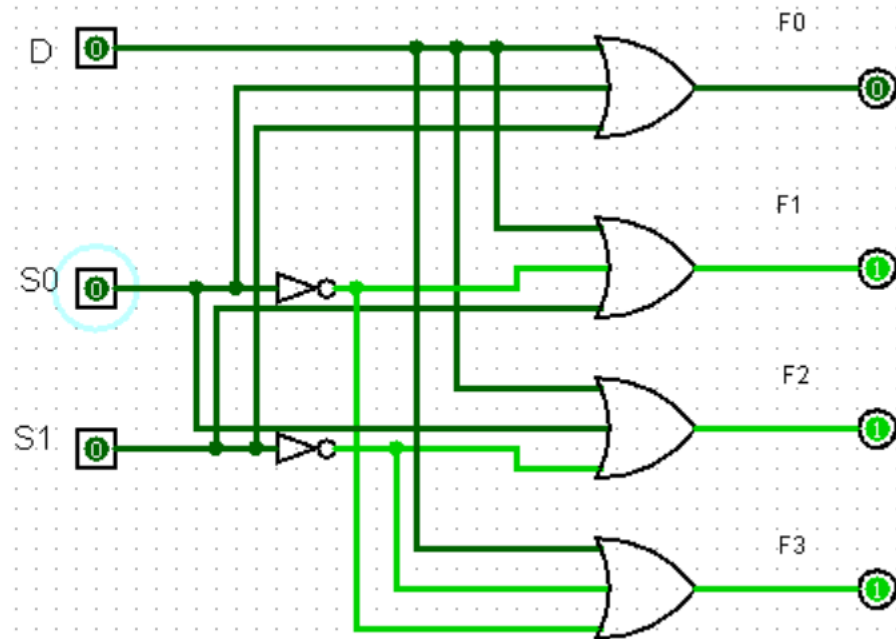
当控制信号为 11 时，第四个门的输入可以输出，如图所示

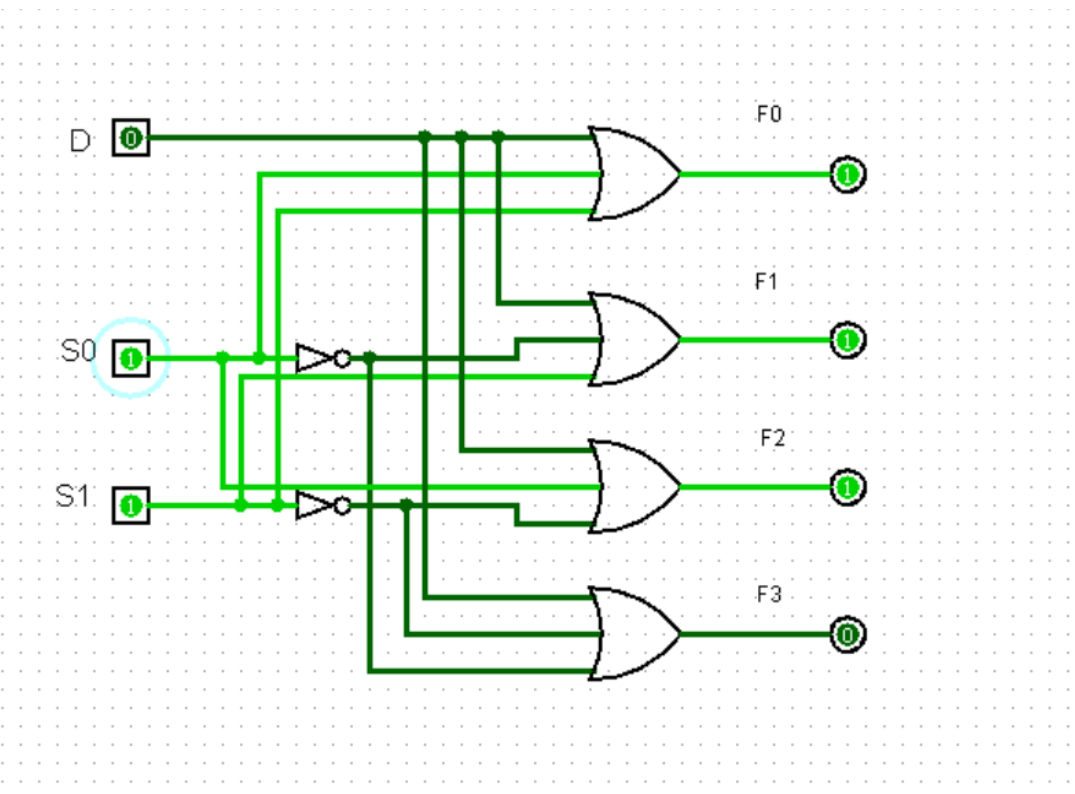
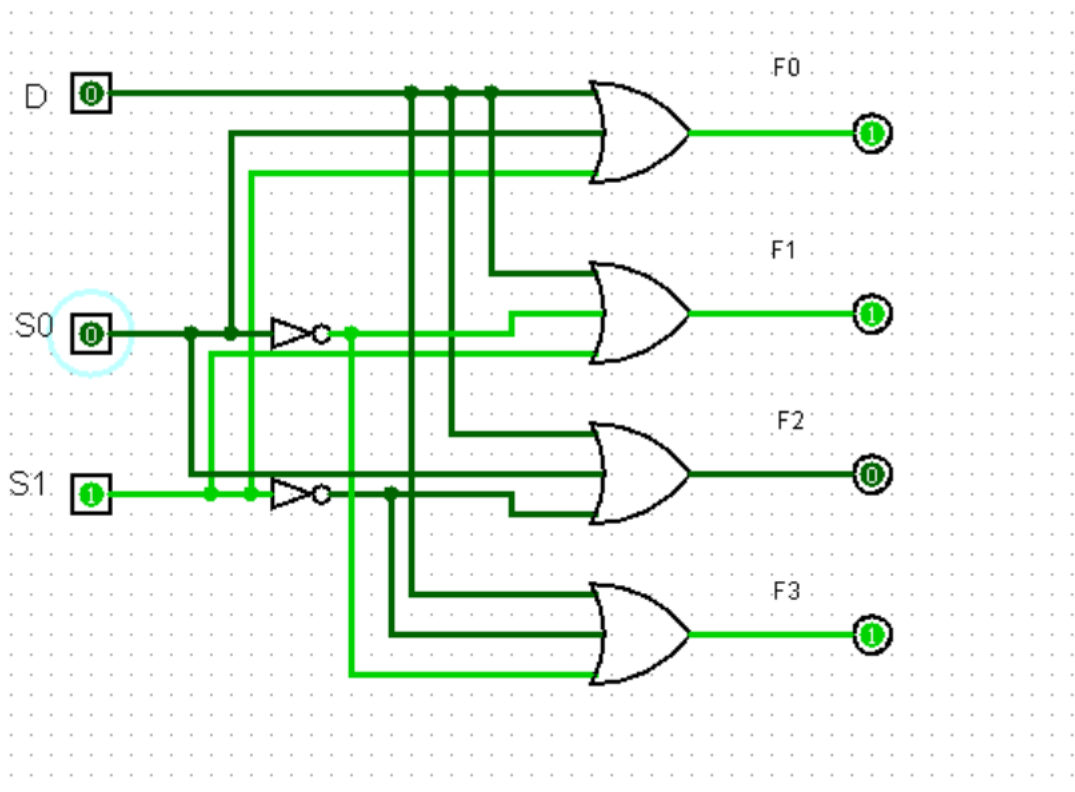


实验二：

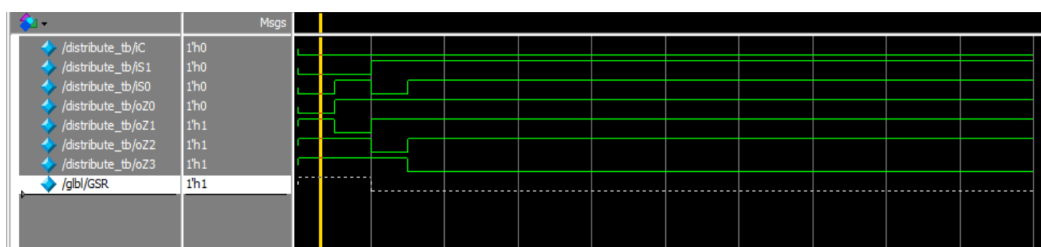
1、逻辑验证图：

这四张图片分别代表控制信号的四种不同取值情况，00 可以分配给 f0，01 可以分配给 f1，10 可以分配给 f2，11 可以分配给 f3。观察四个图片可以验证逻辑正确。



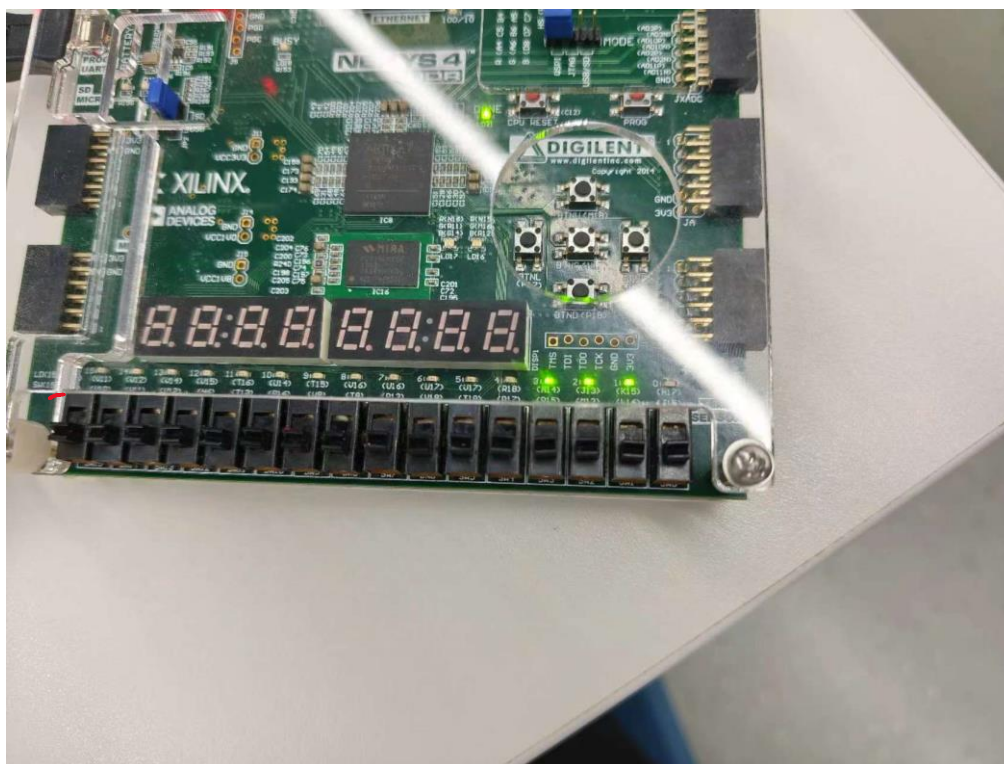


2、仿真波形图：

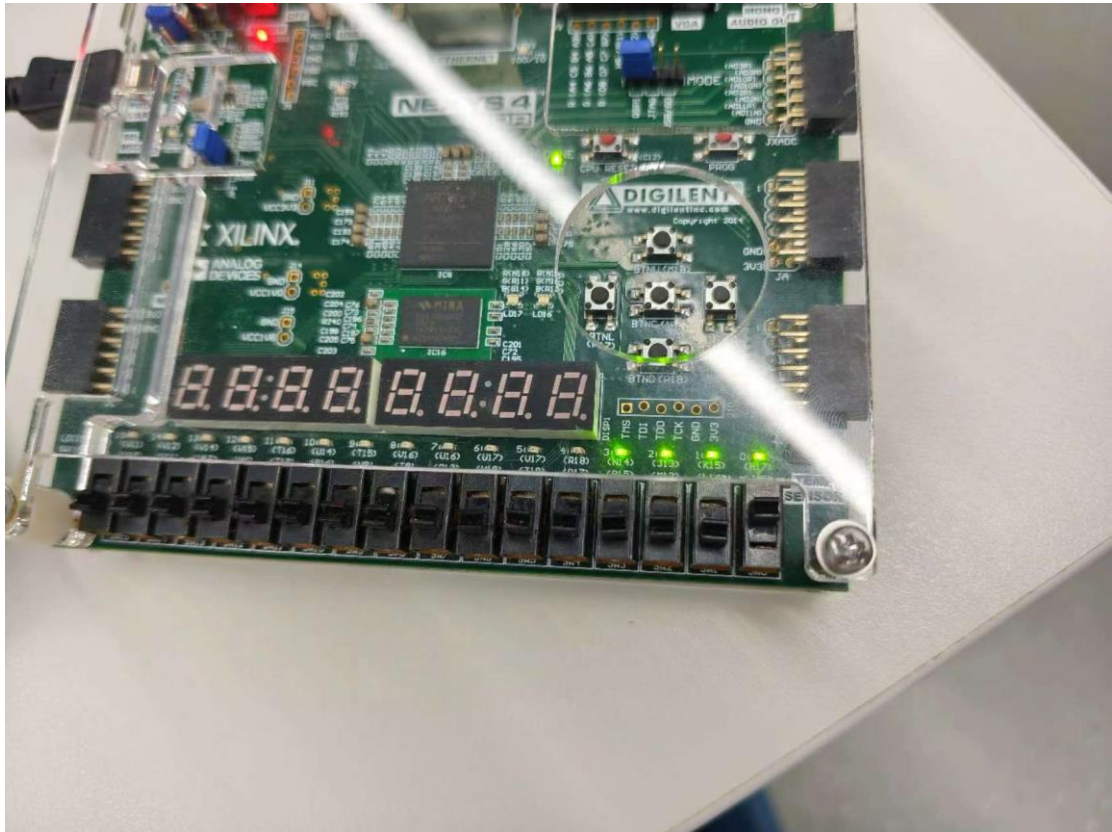


3、下板实验结果：

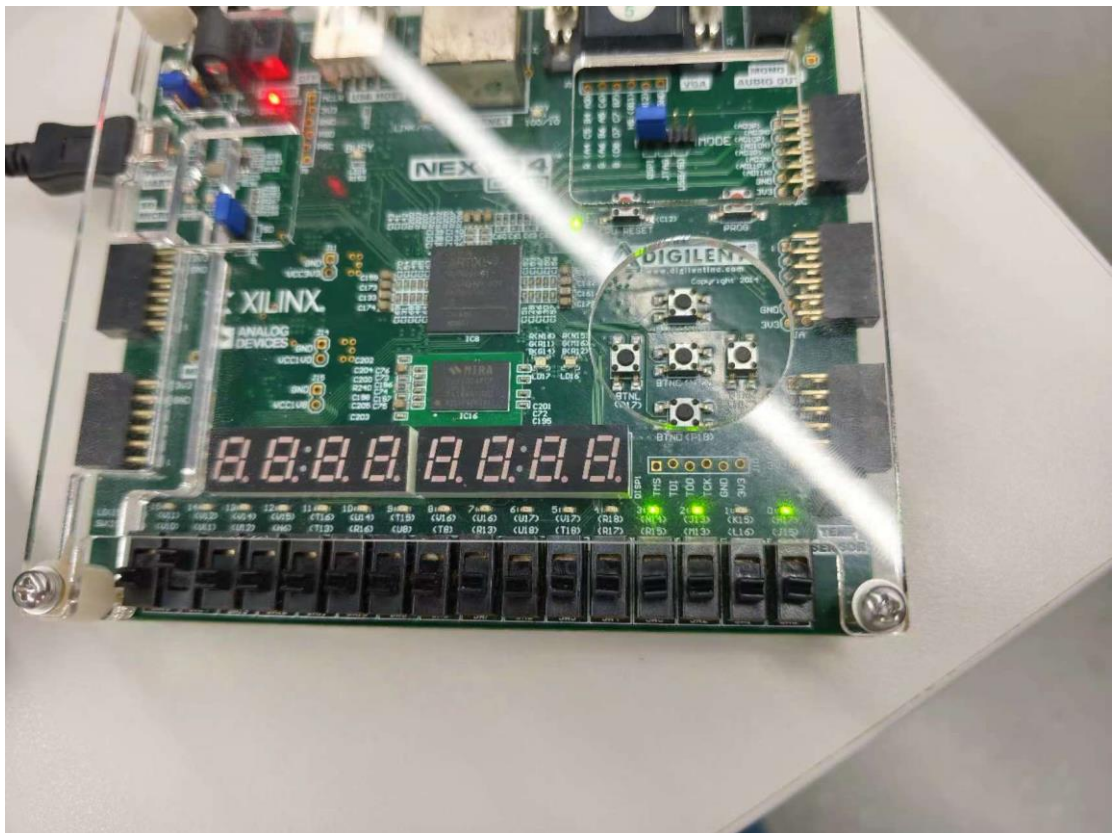
(1) 当控制信号是 00，输入是 0 的时候，可以看到最右边的灯是灭的，而左边三个都是亮的，说明输入这时候分配到了 z0



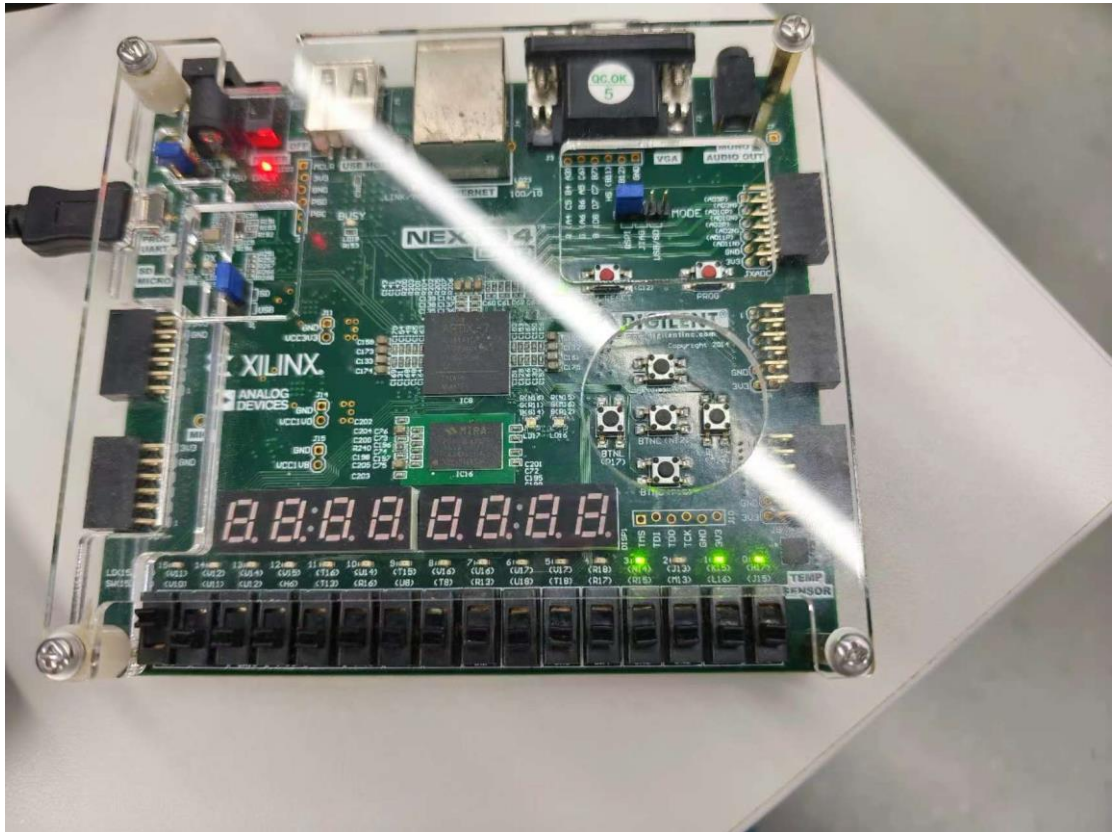
当输入信号是 1 的时候，最右边的灯变亮了，更加应证了输入分配给了 f0



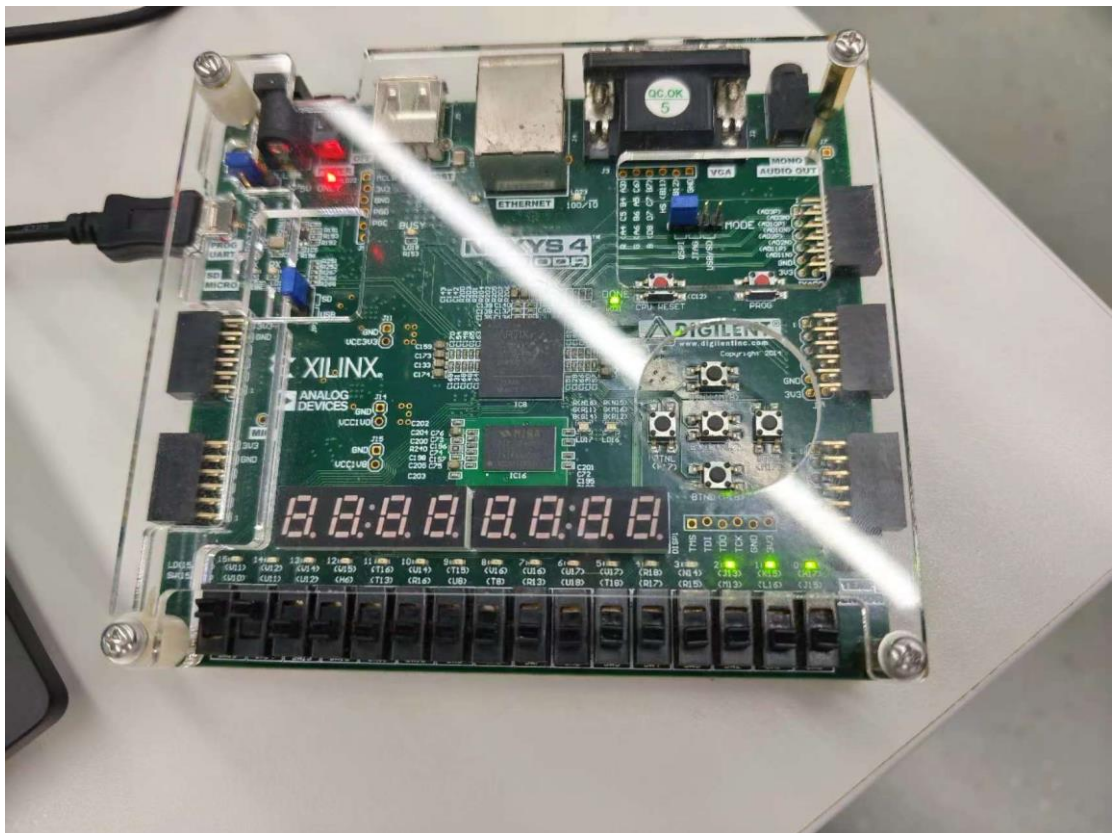
(2) 当控制信号是 01 时，输入分配给 z1（第二个灯）



(3) 当控制信号是 10，输入分配给 z2（第三个灯）

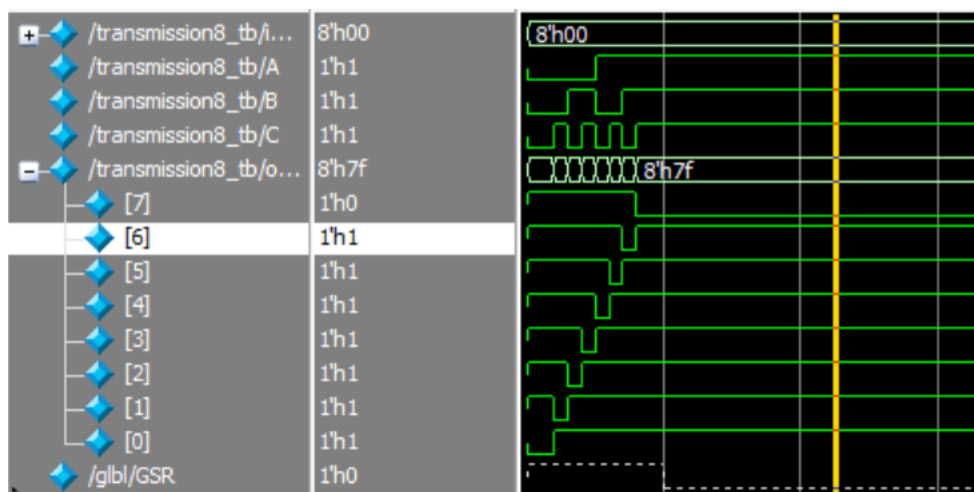


(4) 当控制信号是 11，输入分配给 z3（第四个灯）



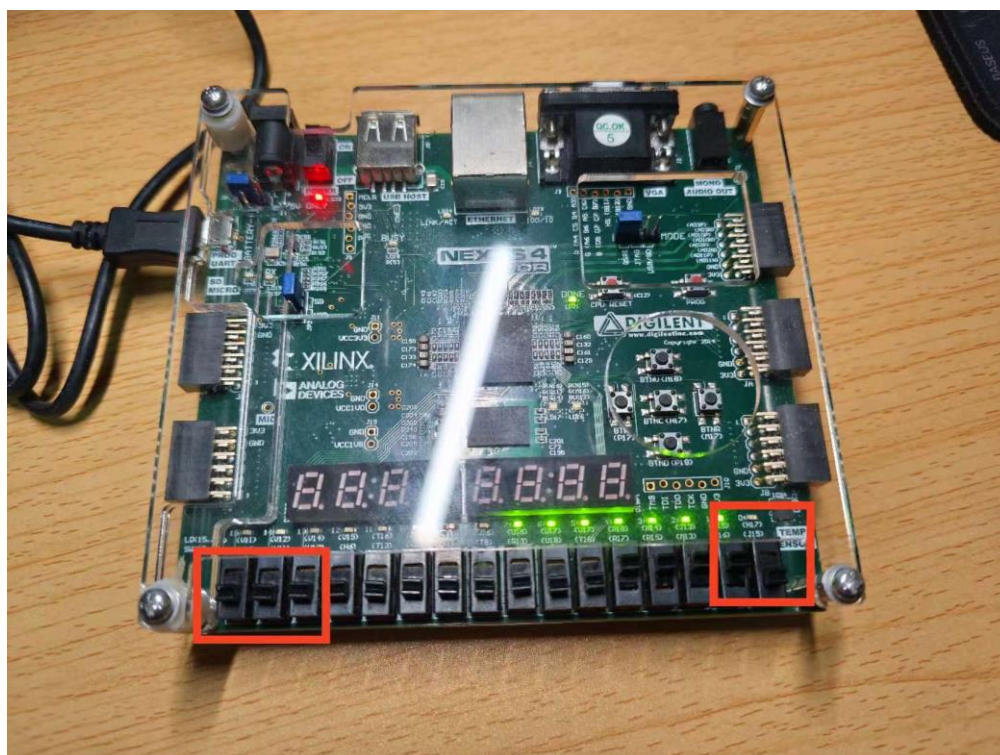
实验三：

1、仿真波形图

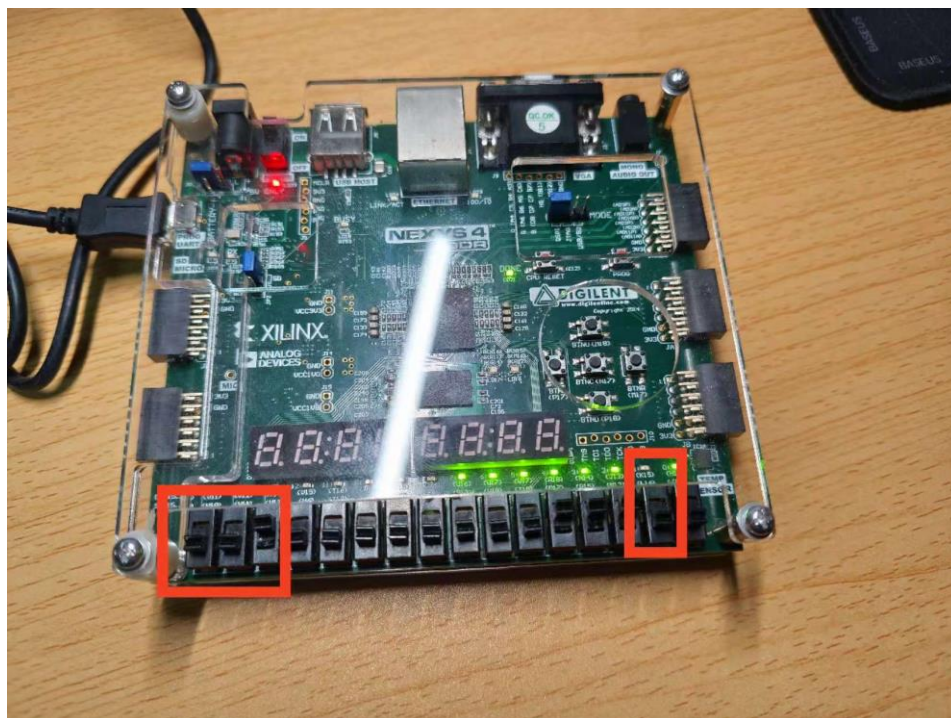


2、下板实验结果

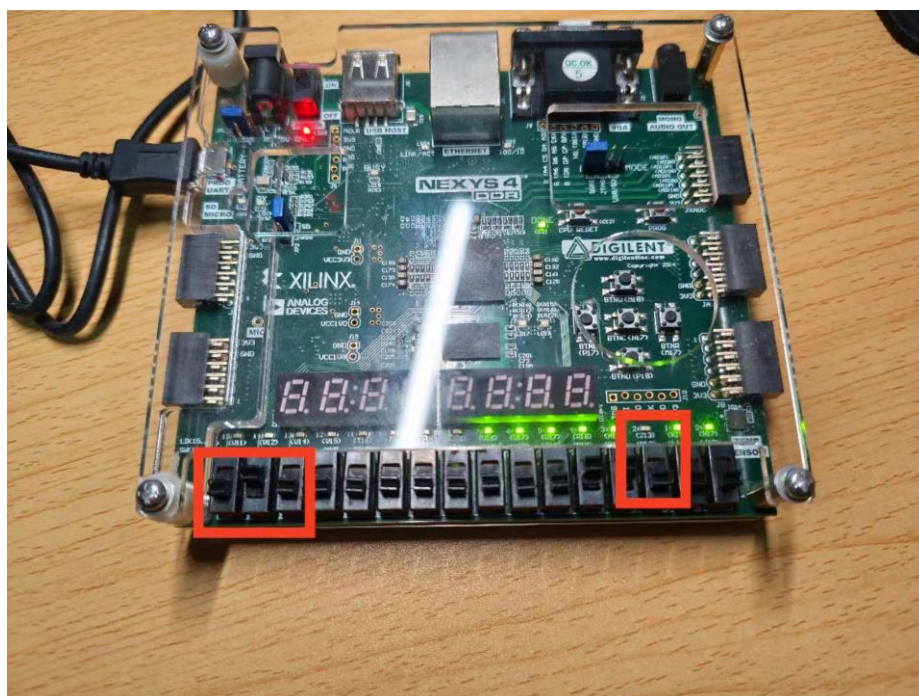
当控制信号为 000 时，d0 送到 f0，如图 d0 为 0，f0 是灭的 (0)，其余七路输出都是 1



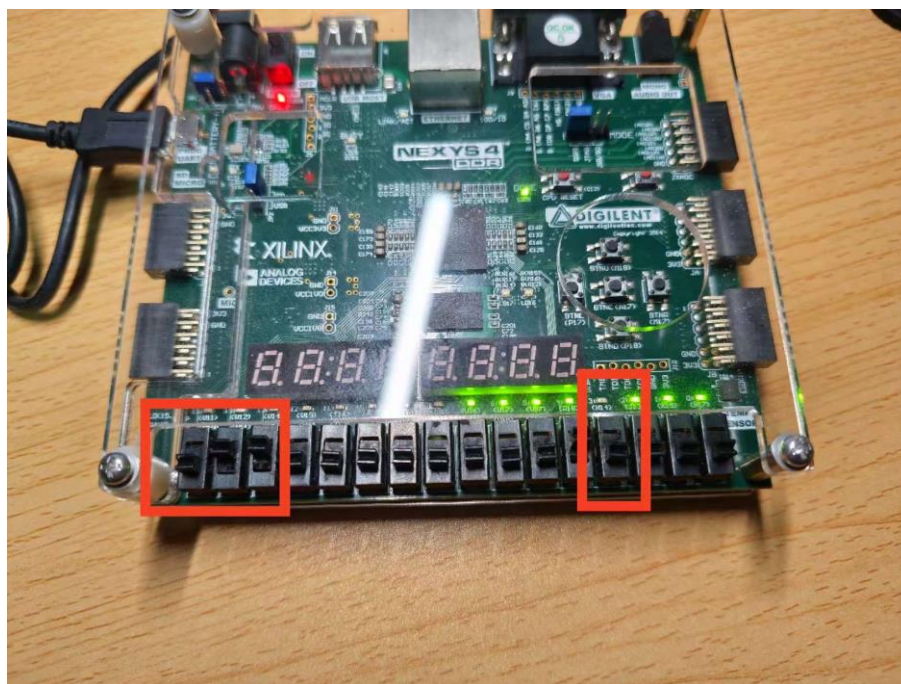
控制信号为 001 时，d1 送到 f1，如图所示



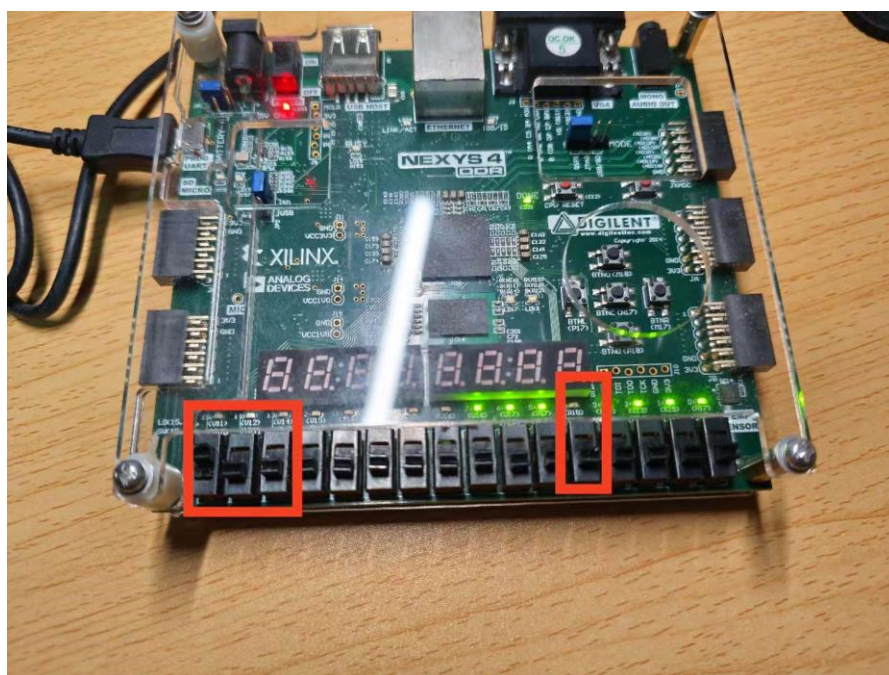
控制信号为 010, d2 送到 f2, 如图所示



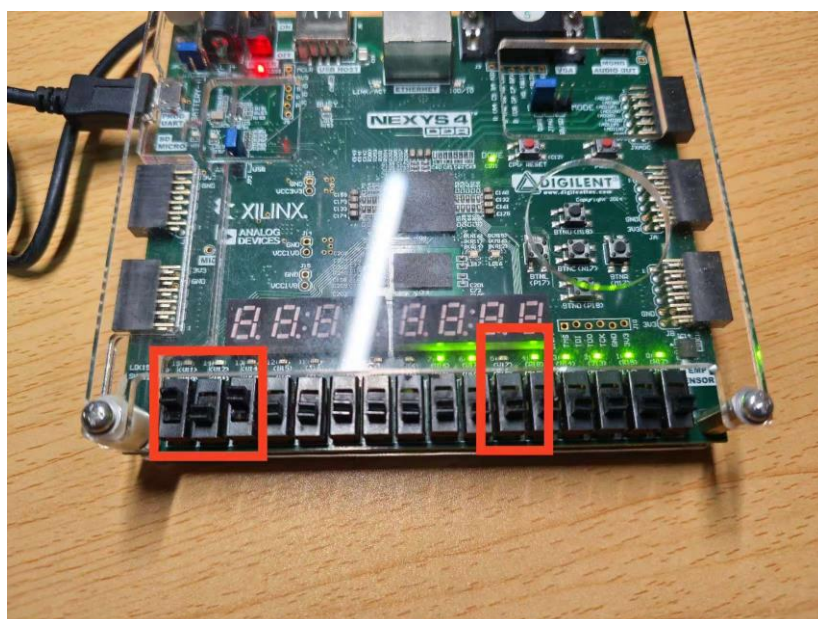
控制信号为 011, d3 送到 f3, 如图所示



控制信号为 100，d4 送到 f4，如图所示



控制信号为 101，d5 送到 f5，如图所示



控制信号为 111，d7 送到 f7，如图所示

