

同济大学计算机系

数字逻辑课程实验报告



学 号 2154312

姓 名 郑博远

专 业 计算机科学与技术

授课老师 郭玉臣

一、实验内容

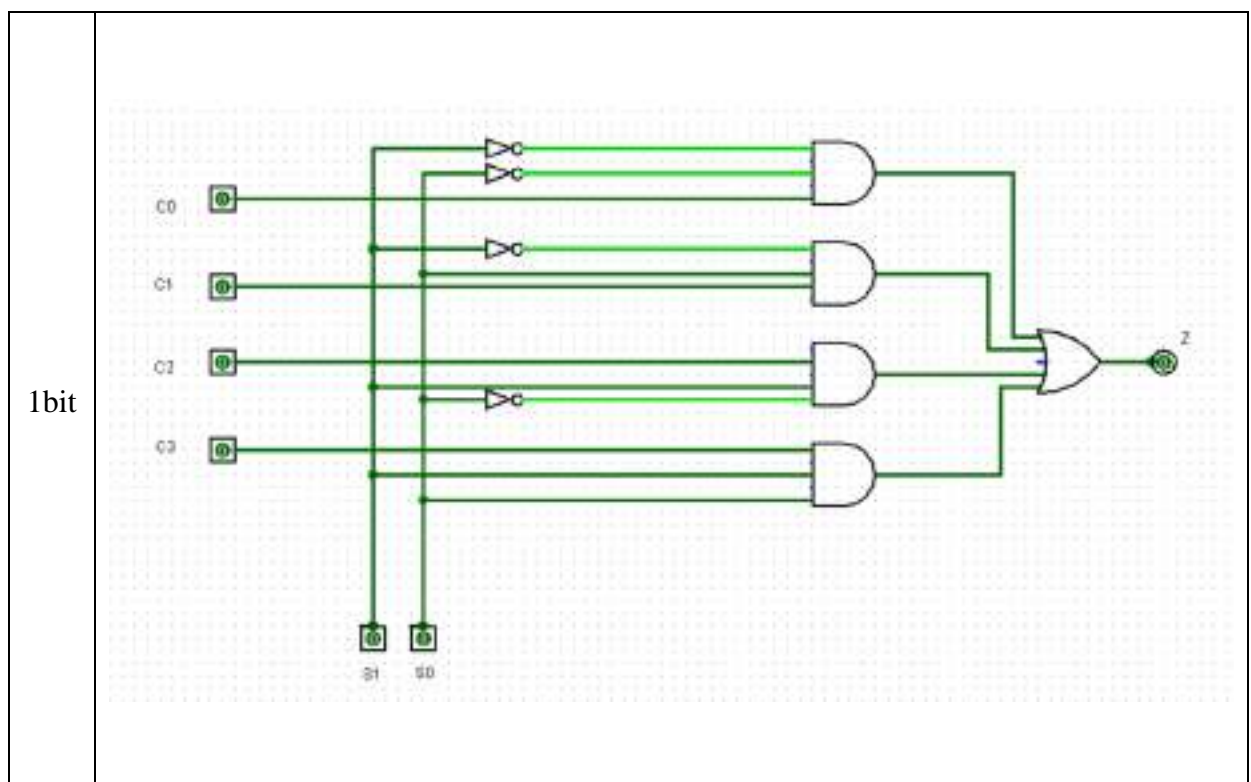
(1) 数据选择器实验：数据选择器（MUX）是一种多路输入、单路输出的标准化逻辑构件。选择器的开关由两根控制线 s_0 和 s_1 的编码控制，选择 4 路输入中的一路作为输出。输出 z 的逻辑值将和被选中的输入逻辑值相同。

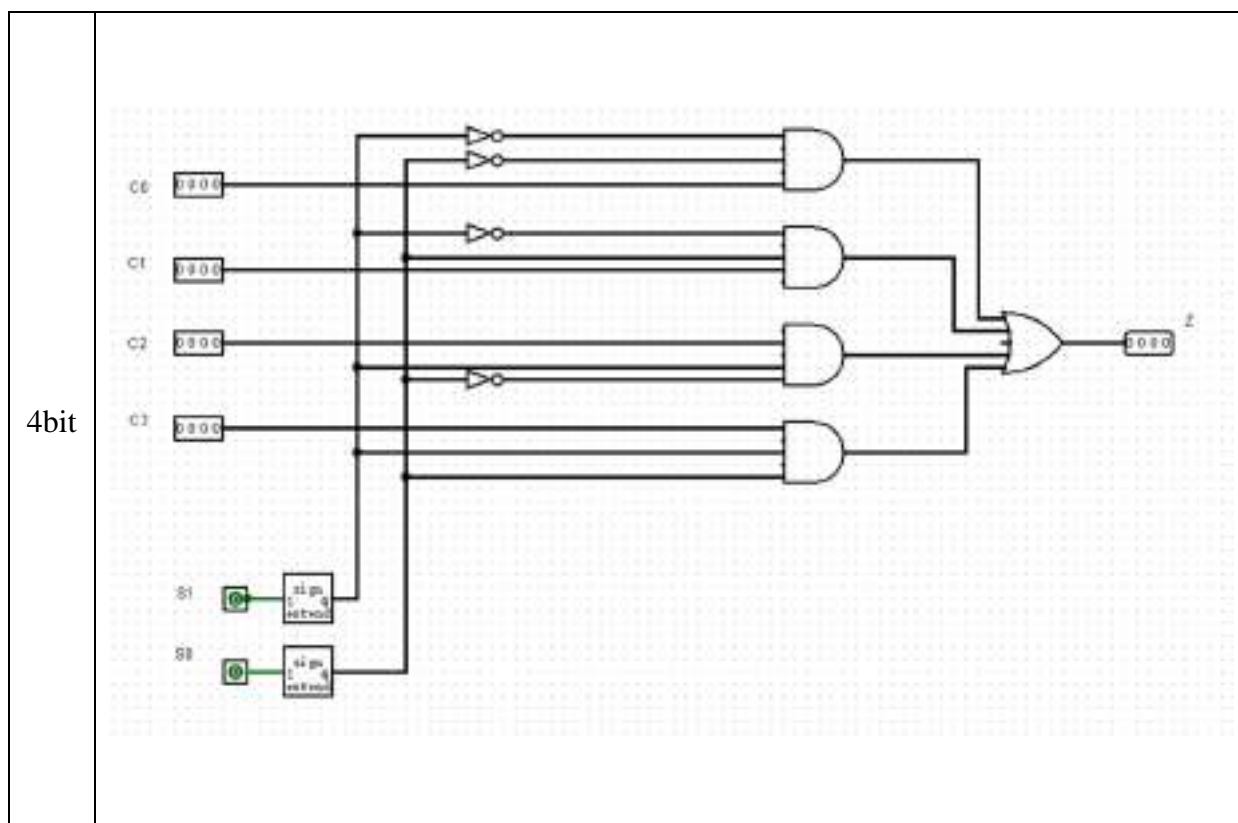
(2) 数据分配器实验：数据分配器（DMUX）的功能与多路选择器相反，它是一种单路输入、多路输出的逻辑构件。

(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 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 $D_5 \rightarrow f_5$ 的数据传输。

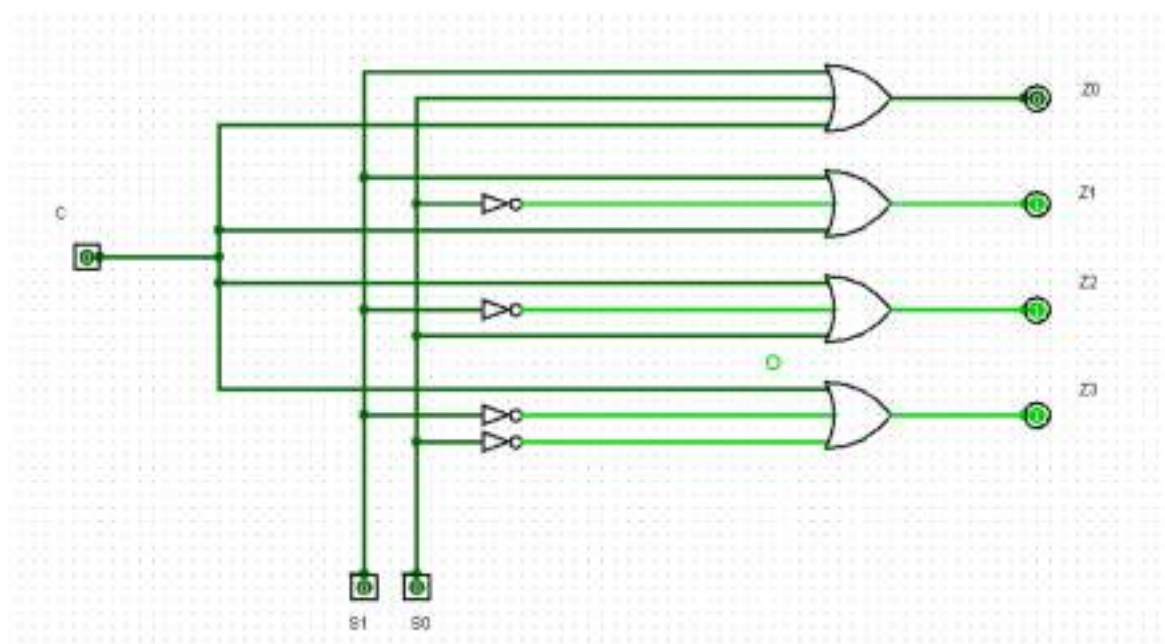
二、硬件逻辑图

(1) 数据选择器实验：

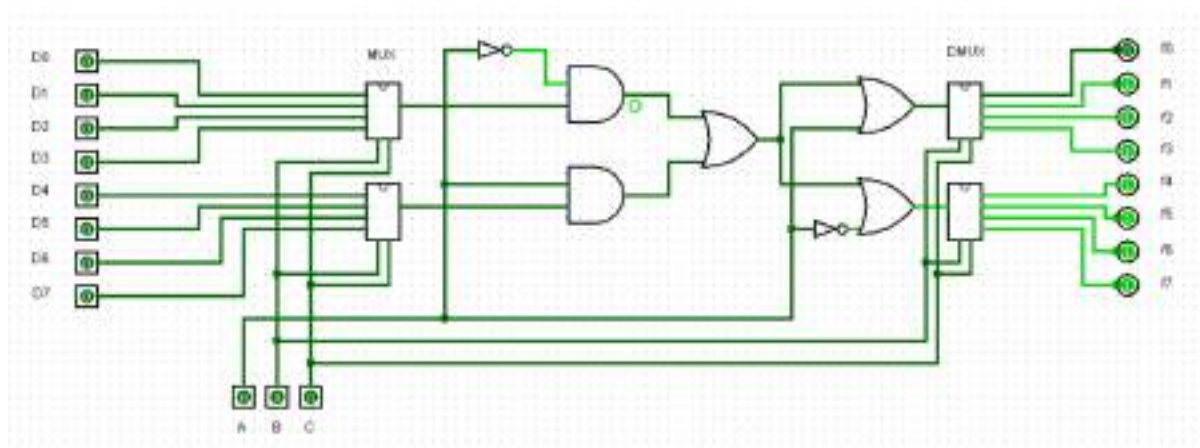




(2) 数据分配器实验:



(3) 8 路数据传输实验:



三、模块建模

(1) 数据选择器实验：

iC0、iC1、iC2、iC3 为模块的四个输入，由 iS1、iS0 代表两根控制线，共同选择 iC0~iC3 中的一路在 oZ 中输出。以下是 Verilog 代码：

```
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;
    endcase
end
endmodule

```

(2) 数据分配器实验:

iC 是模块的一路输入, iS1、iS0 是两个控制信号, oZ0、oZ1、oZ2、oZ3 是模块的输出; 通过 iS1、iS0 的选择控制 iC 的这一路信号在 oZ0~oZ3 的哪一路中输出。Verilog 代码如下:

```

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
);
    reg [3:0] oZ;
    assign {oZ0, oZ1, oZ2, oZ3} = oZ;

    always @(*)
        begin
            case({iS1, iS0})
                2'b00:

```

```

        oZ = {iC, 3'b111};
    2'b01:
        oZ = {1'b1, iC, 2'b11};
    2'b10:
        oZ = {2'b11, iC, 1'b1};
    2'b11:
        oZ = {3'b111, iC};
        default;
    endcase
end

endmodule

```

(3) 8 路数据传输实验:

iData 与 oData 分别为模块的 8 路输入与输出。通过 A、B、C 三路控制信号来决定将哪一路的输入信号传输到对应路的输出信号。例如当 ABC=101 时, 实现 D5->f5 的数据传输。以下是本小题的 Verilog 代码:

```

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

    assign oData = iData | ~(1 << {A, B, C});

endmodule

```

四、测试模块建模

(1) 数据选择器实验:

```

`timescale 1ns / 1ps

module selector41_tb;
    reg [3:0] iC0, iC1, iC2, iC3;
    reg iS0, iS1;
    wire [3:0] oZ;
    initial begin
        iC0 = 4'b0001;
        iC1 = 4'b0010;
        iC2 = 4'b0100;
        iC3 = 4'b1000;
    end

    initial begin
        {iS1, iS0} = 2'b00;
        #10;
        {iS1, iS0} = 2'b01;
        #10;
        {iS1, iS0} = 2'b10;
        #10;
        {iS1, iS0} = 2'b11;
    end

    selector41 selector41_inst(iC0, iC1, iC2, iC3, iS1,
iS0, oZ);
endmodule

```

(2) 数据分配器实验:

```

`timescale 1ns / 1ps

module de_selector14_tb;
    reg iC;
    reg iS0, iS1;
    wire oZ0, oZ1, oZ2, oZ3;
    initial begin
        iC = 0;
        #10;
        iC = 1;
        #10;
        iC = 0;
        #10;
        iC = 1;
        #10;
    end

```

```

        iC = 0;
        #10;
        iC = 1;
        #10;
        iC = 0;
        #10;
        iC = 1;
    end

    initial begin
        {iS1, iS0} = 2'b00;
        #20;
        {iS1, iS0} = 2'b01;
        #20;
        {iS1, iS0} = 2'b10;
        #20;
        {iS1, iS0} = 2'b11;
    end

    de_selector14 de_selector14_inst(iC, iS1, iS0, oZ0,
oZ1, oZ2, oZ3);
endmodule

```

(3) 8 路数据传输实验:

```

`timescale 1ns / 1ps

module transmission8_tb();
    wire [7:0] iData;
    wire A, B, C;
    wire [7:0] oData;
    reg [7:0] iDatatot;
    reg [2:0] ctrl;
    assign iData = iDatatot;
    assign {A, B, C} = ctrl;

    initial
    begin
        iDatatot = 8'b00000000;
        #10;
        for(iDatatot = 8'b00000001; iDatatot <=
8'b10000000; iDatatot = iDatatot << 1)
            begin

```



```

        #20;
    end
end

initial
begin
    for(ctrl = 3'b000; ctrl <= 3'b111; ctrl = ctrl +
3'b001)
        begin
            #20;
        end
    end

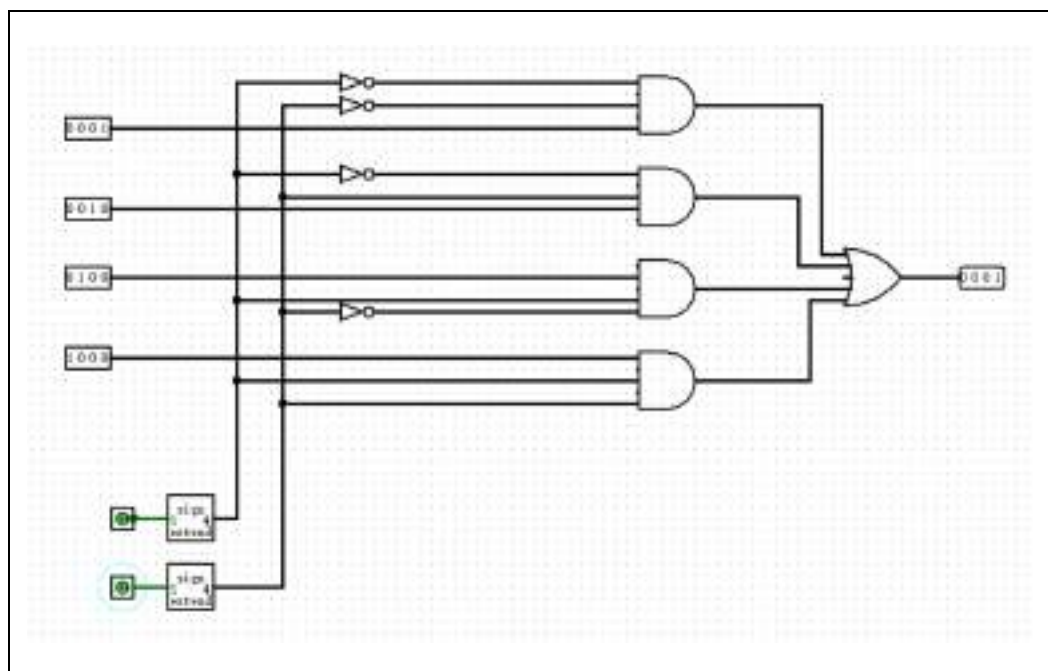
    transmission8
    transmission8_instc(iData, A, B, C, oData);
endmodule

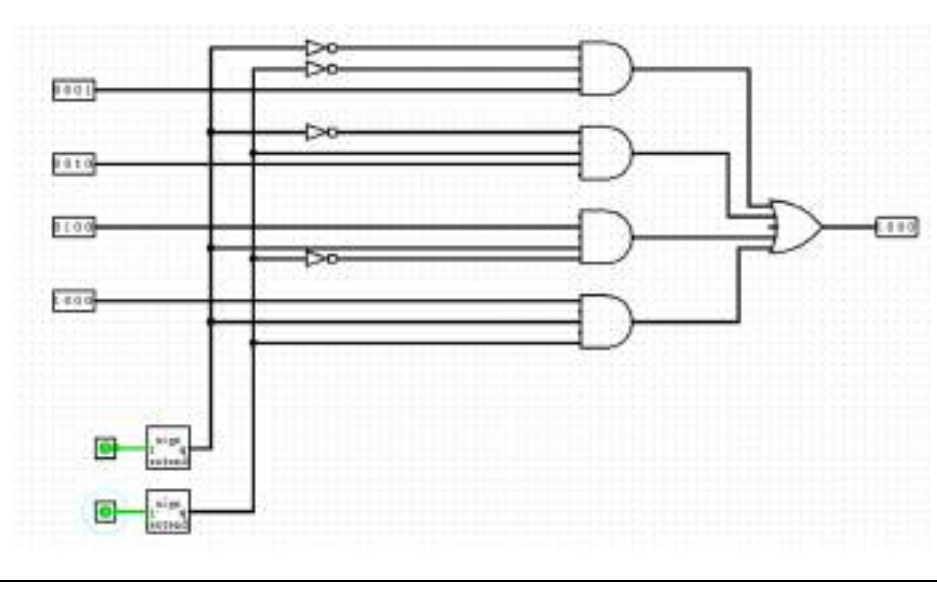
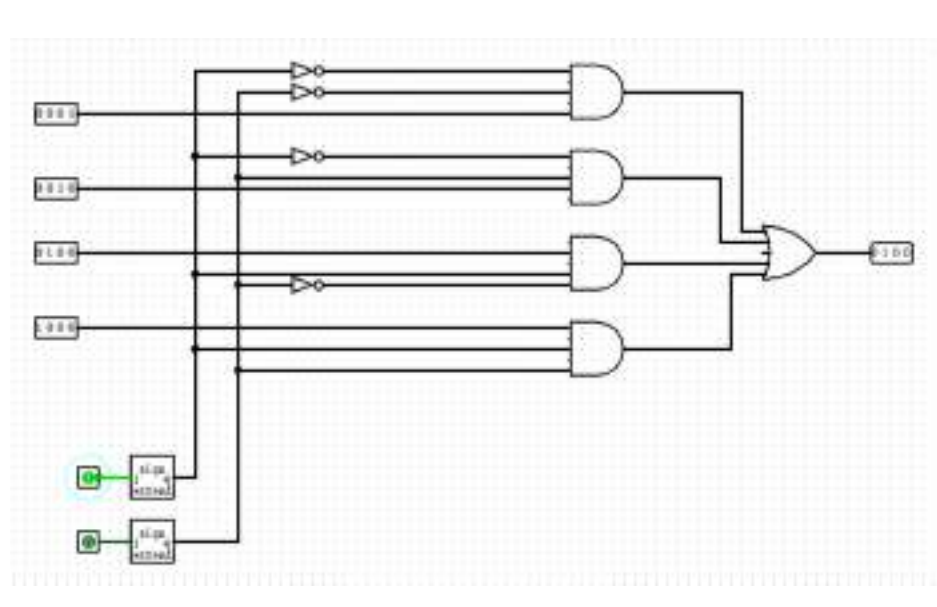
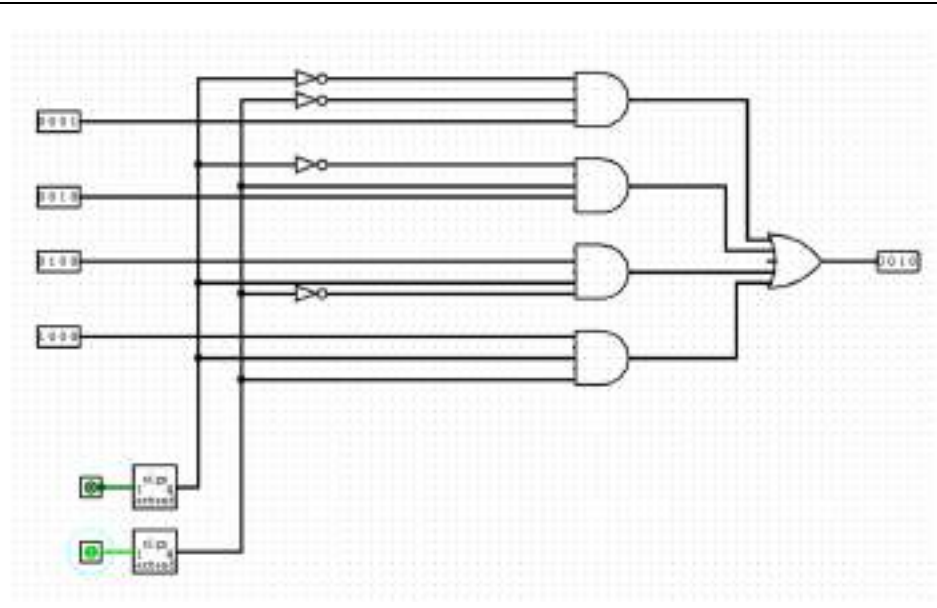
```

五、实验结果

(1) 基本门电路实验：

a) logisim 逻辑验证图





由图可知，设 C0~C4 分别为 0001、0010、0100 和 1000，在 S1、S0 分别为 00、01、10、11 的情况下，Z 的输出结果依次与 C0~C4 相同。



b) modelsim 仿真波形图



c) 下板实验结果图

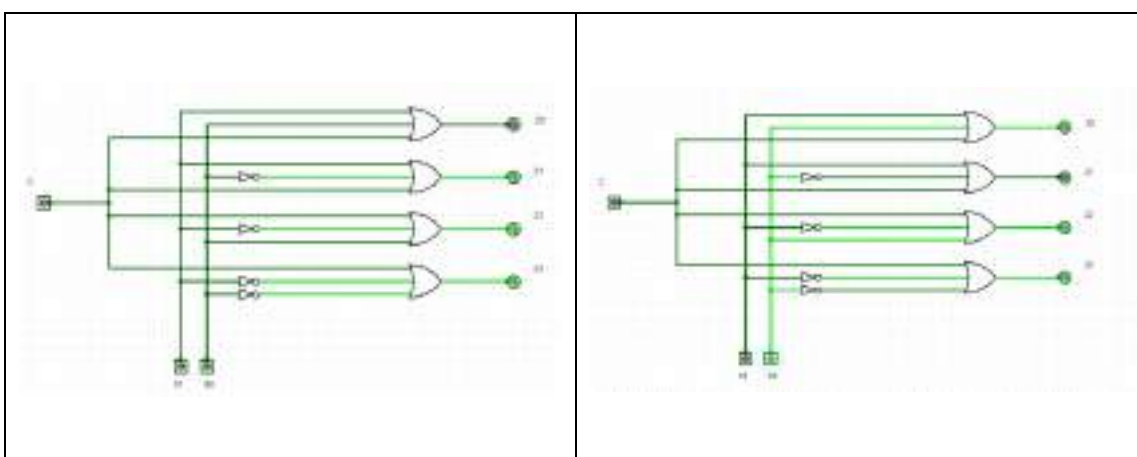
将 C0、C2、C2、C3 的输入分别设置为 0001、0010、0100、1000（即 J15、T18、R16、V10 打开，其他开关均关闭），分别改变 S0、S1 观察现象。

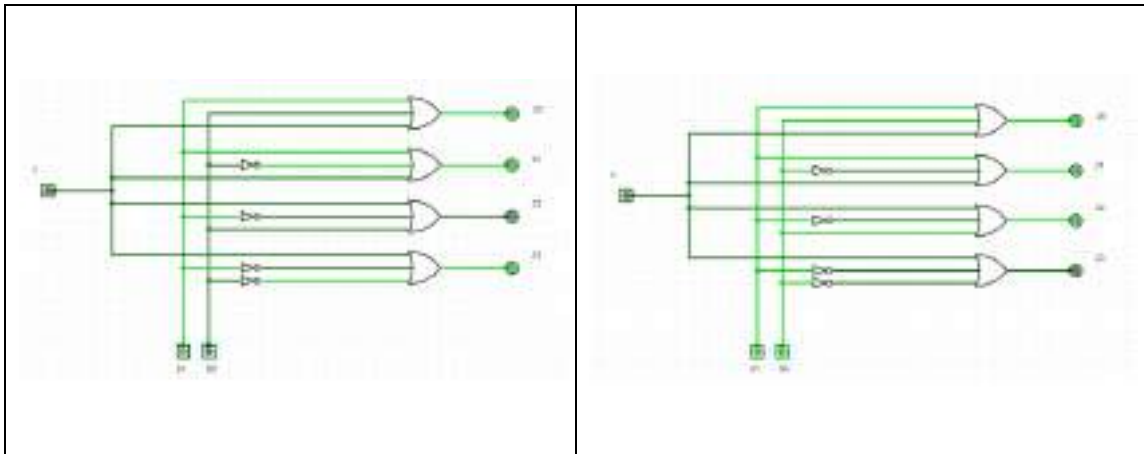
<p>不按下 N17、M17， 仅 H17 亮起</p>	A photograph of the Nexys4 board. A white circle highlights a group of LEDs. LED H17 is illuminated, while others in the circle are not. The 7-segment display shows '00:00:00'.
<p>按下 N17、不按 M17， 仅 K15 亮起</p>	A photograph of the Nexys4 board with a finger pressing switch N17. A white circle highlights the same group of LEDs as in the previous image. LED K15 is now illuminated, and H17 is not. The 7-segment display shows '00:00:00'.

<p>按下 M17、不按 N17， 仅 J13 亮起</p>	
<p>按下 N17、M17， 仅 N14 亮起</p>	

(2) 三态门实验：

a) logisim 逻辑验证图





b) modelsim 仿真波形图



c) 下板实验结果图

当 iC 为 1 时, J15、L16、M13、R15 均亮起; 当 iC 为 0 时, 观察如下:

V10、U11 均关闭,
H17 不亮, 其余亮起



U11 打开、V10 关闭，
K15 不亮，其余亮起



V10 打开、U11 关闭，
J13 不亮，其余亮起

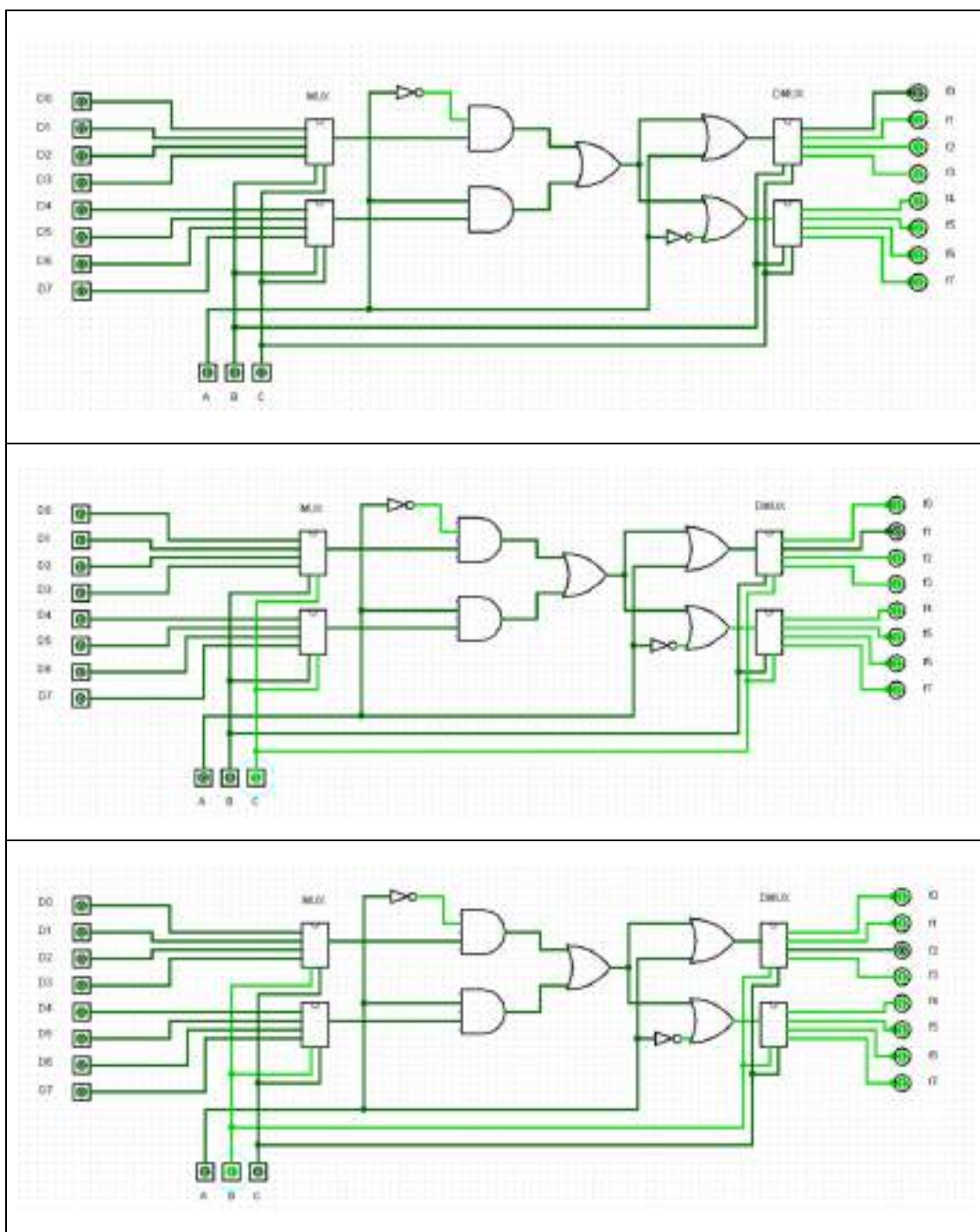


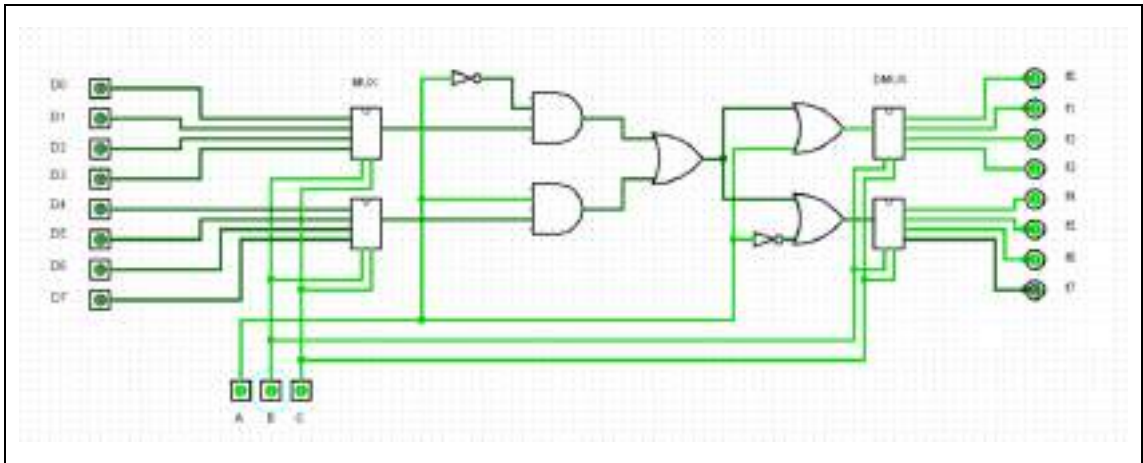
V10、U11 均打开，
N14 不亮，其余亮起



(3) 8 路数据传输实验:

a) logisim 逻辑验证图





b) modelsim 波形仿真图



c) 下板实验结果图（前三张图 D0~D7 均为 0，即 J15~R13 关闭）

V10、U11、U12 均打开，
除 U16 不亮外，其余 7 灯亮



U11 打开，V10、U12 关闭，
除 J13 不亮外，其余 7 灯亮



V10、U11 打开，U12 关闭，
除 U17 不亮外，其余 7 灯亮



在上图基础上打开开关 U18，
U17 亮起，即 8 个灯全亮

