同济大学计算机系

数字逻辑课程实验报告

学　　号 2252707

姓　　名 陈艺天

专　　业 计算机科学与技术

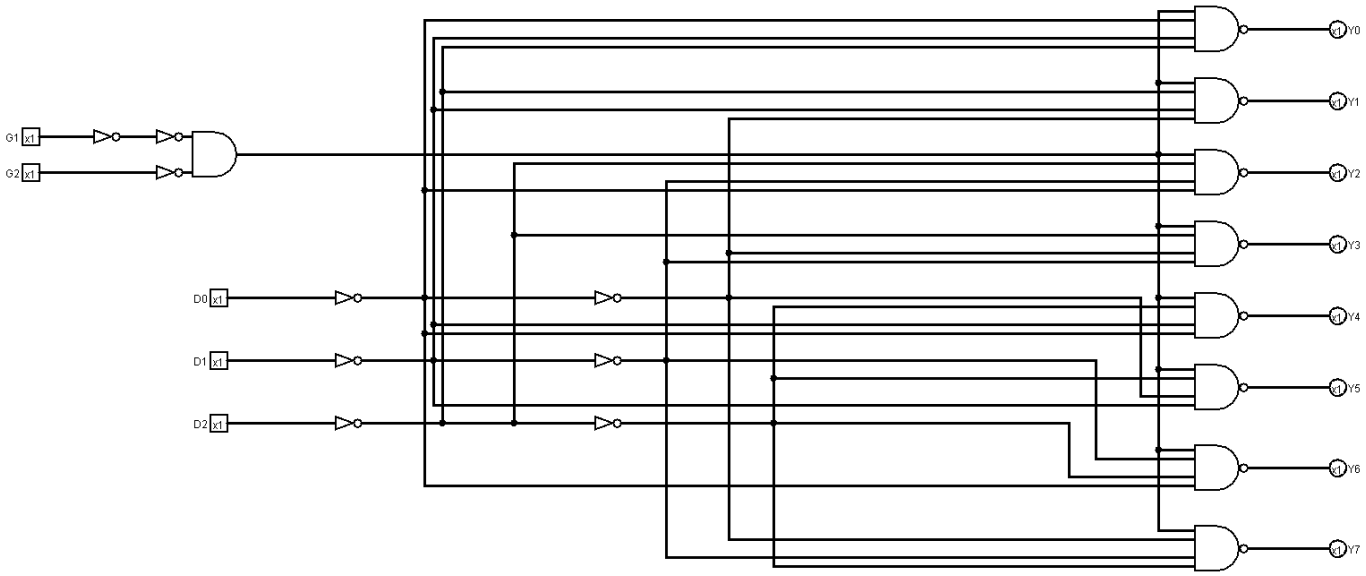
授课老师 张冬冬

1. 3-8 译码器

一、实验内容

（1） 在本次实验中，我们将使用 **Verilog HDL** 语言实现 3-8 译码器设计和仿真。
（2） 使用 **logisim** 画出译码器实验的逻辑图。
（3） 使用 **verliog** 语言实现译码器。

二、硬件逻辑图



三、模块建模

功能描述：该模块实现 3-8 译码器；iData 为 2 为输入，iEna 为控制信号，oData 位输出。

```
module decoder
    input[2:0] iData,
    input[1:0] iEna,
    output[7:0] oData
    );

    assign oData[0]=
    ~iEna[1]|iEna[0]|iData[0]|iData[1]|iData[2];

    assign oData[1]=
    ~iEna[1]|iEna[0]|~iData[0]|iData[1]|iData[2];
```

```
    assign oData[2]=
    ~iEna[1]|iEna[0]|iData[0]|~iData[1]|iData[2];

    assign oData[3]=
    ~iEna[1]|iEna[0]|~iData[0]|~iData[1]|iData[2];

    assign oData[4]=
    ~iEna[1]|iEna[0]|iData[0]|iData[1]|~iData[2];

    assign oData[5]=
    ~iEna[1]|iEna[0]|~iData[0]|iData[1]|~iData[2];

    assign oData[6]=
    ~iEna[1]|iEna[0]|iData[0]|~iData[1]|~iData[2];

    assign oData[7]=
    ~iEna[1]|iEna[0]|~iData[0]|~iData[1]|~iData[2];

endmodule
```

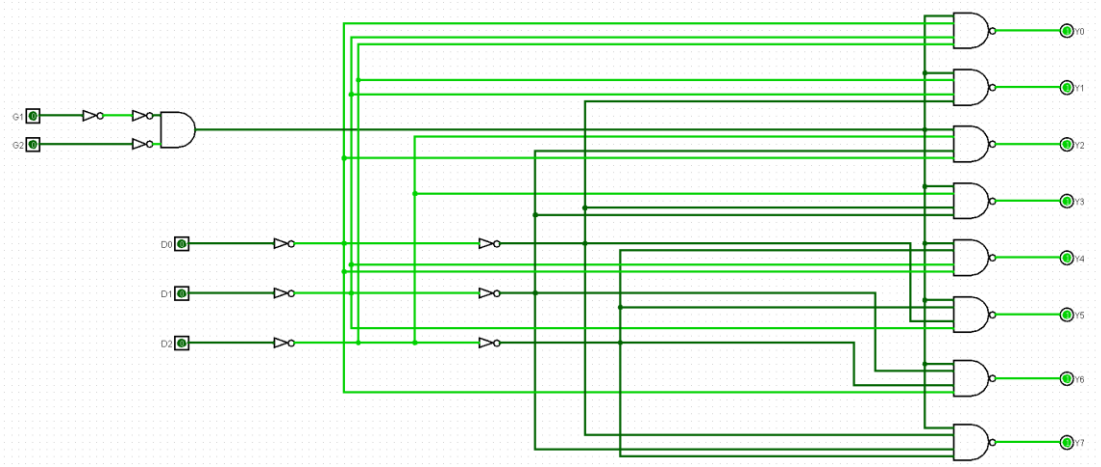
四、测试模块建模

```
module decoder_tb;
    reg[1:0] iEna;
    reg[2:0] iD;
    wire[7:0] out;

    decoder testdecoder(iD,iEna,out);

    initial begin
        iEna[1]=1;
        iEna[0]=0;

        #320 iEna[1]=0;
        iEna[0]=1;

        #320 iEna[1]=1;
        iEna[1]=1;

        #320 iEna[1]=0;
        iEna[1]=0;
    end
```

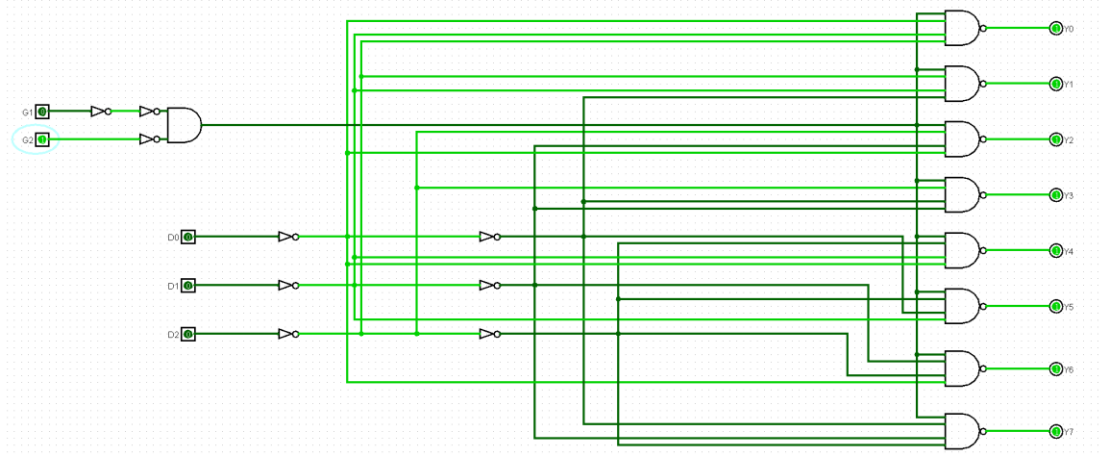
```
    initial begin
        iD=3'b000;
        #40 iD=3'b001;
        #40 iD=3'b010;
        #40 iD=3'b011;
        #40 iD=3'b100;
        #40 iD=3'b101;
        #40 iD=3'b110;
        #40 iD=3'b111;

        #40 iD=3'b000;
        #40 iD=3'b001;
        #40 iD=3'b010;
        #40 iD=3'b011;
        #40 iD=3'b100;
        #40 iD=3'b101;
        #40 iD=3'b110;
        #40 iD=3'b111;

        #40 iD=3'b000;
        #40 iD=3'b001;
        #40 iD=3'b010;
        #40 iD=3'b011;
        #40 iD=3'b100;
        #40 iD=3'b101;
        #40 iD=3'b110;
        #40 iD=3'b111;

        #40 iD=3'b000;
        #40 iD=3'b001;
        #40 iD=3'b010;
        #40 iD=3'b011;
        #40 iD=3'b100;
        #40 iD=3'b101;
        #40 iD=3'b110;
        #40 iD=3'b111;
    end

endmodule
```

五、实验结果

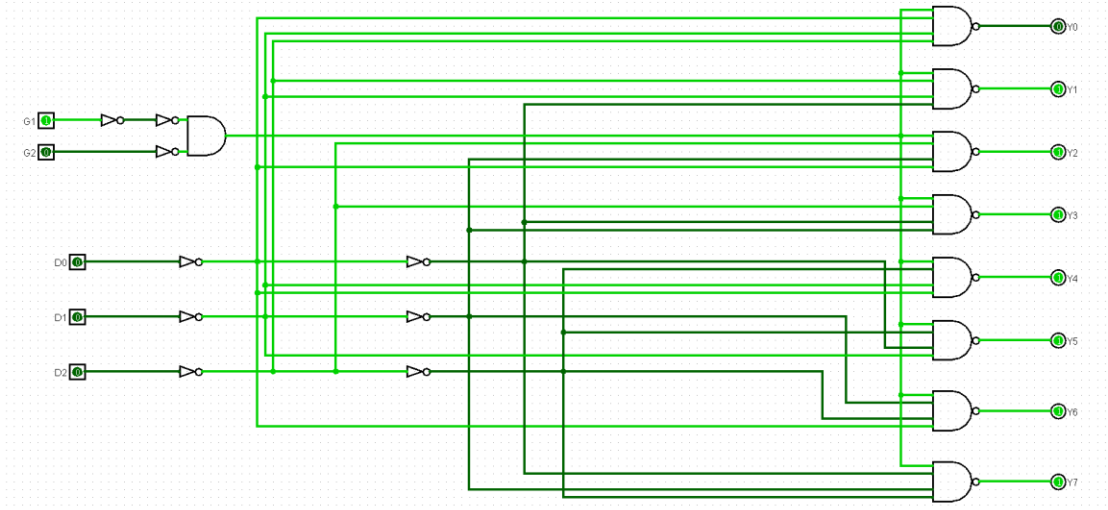
（1） logisim 逻辑验证图



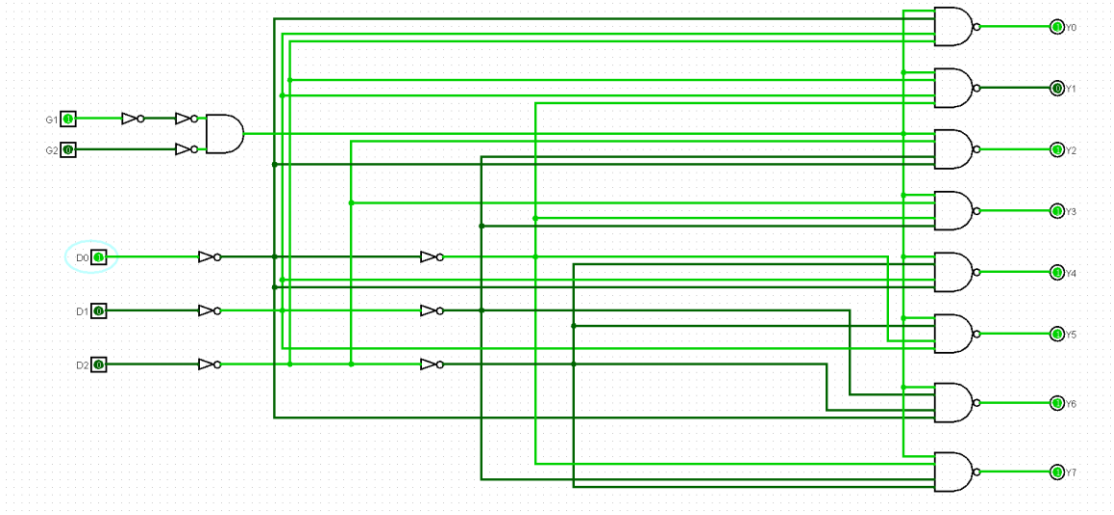
此时 G1 为 0，结果符合预期，全部为 1



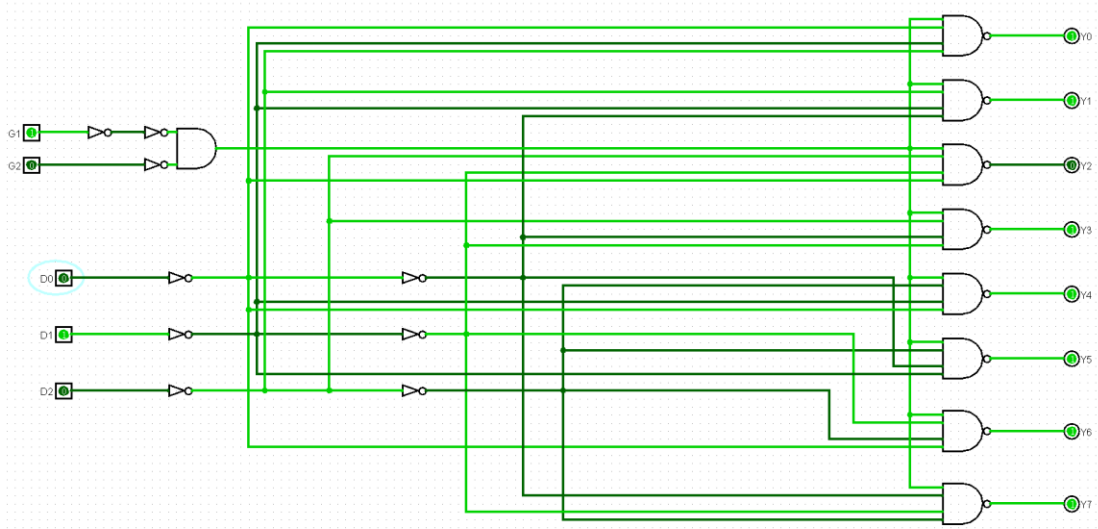
此时 G2 为 1，结果符合预期，全部为 1



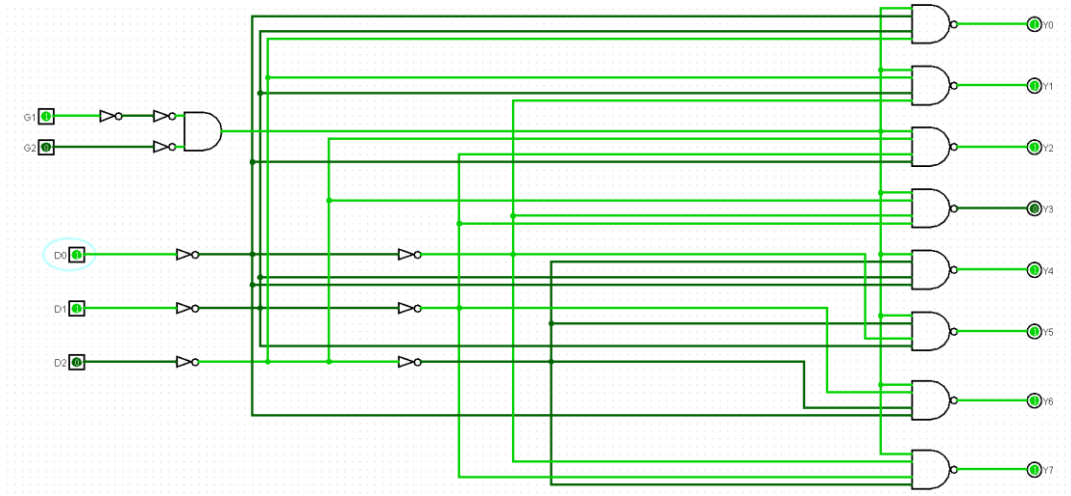
此时输入为（0，0，0）仅有 Y0 为低电平



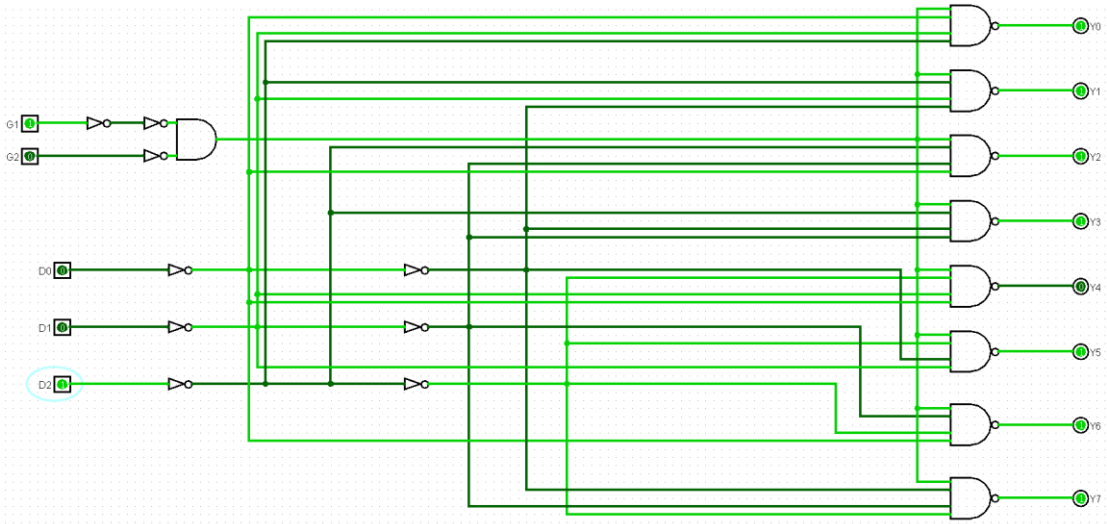
此时输入为（0，0，1）仅有 Y1 为低电平



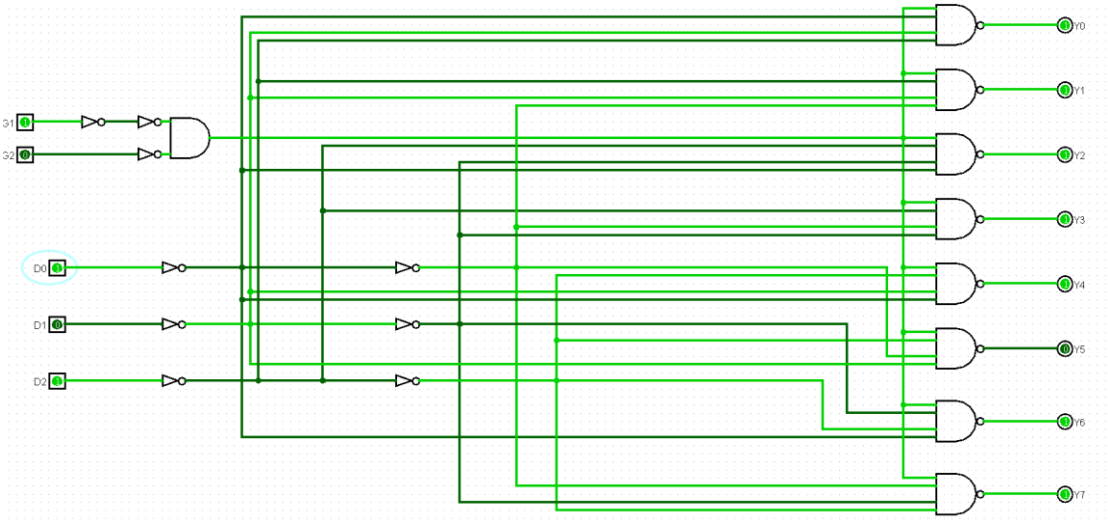
此时输入为（0，1，0）仅有 Y2 为低电平



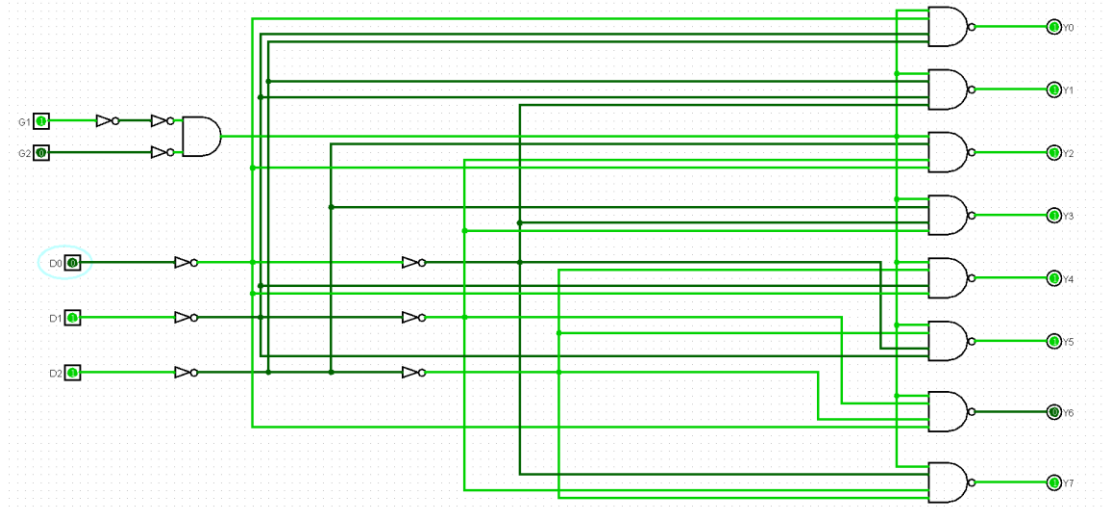
此时输入为（0，1，1）仅有 Y3 为低电平



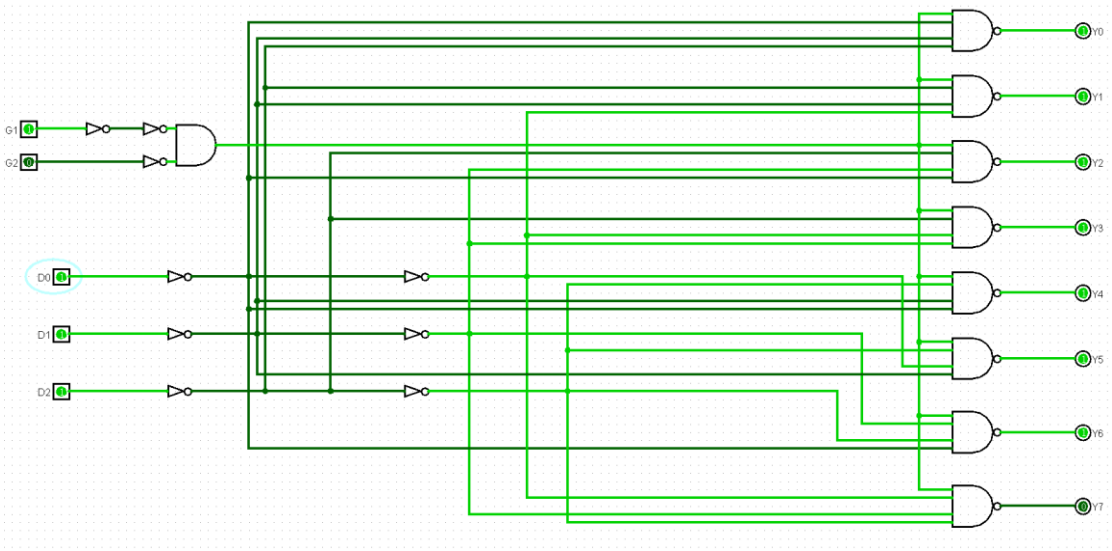
此时输入为（1，0，0）仅有 Y4 为低电平



此时输入为（1，0，1）仅有 Y5 为低电平

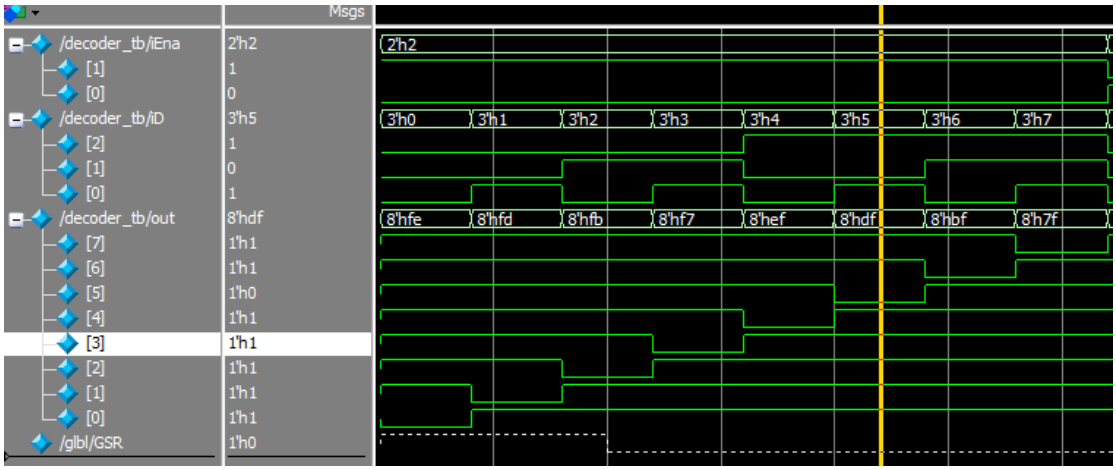


此时输入为（1，1，0）仅有 Y6 为低电平

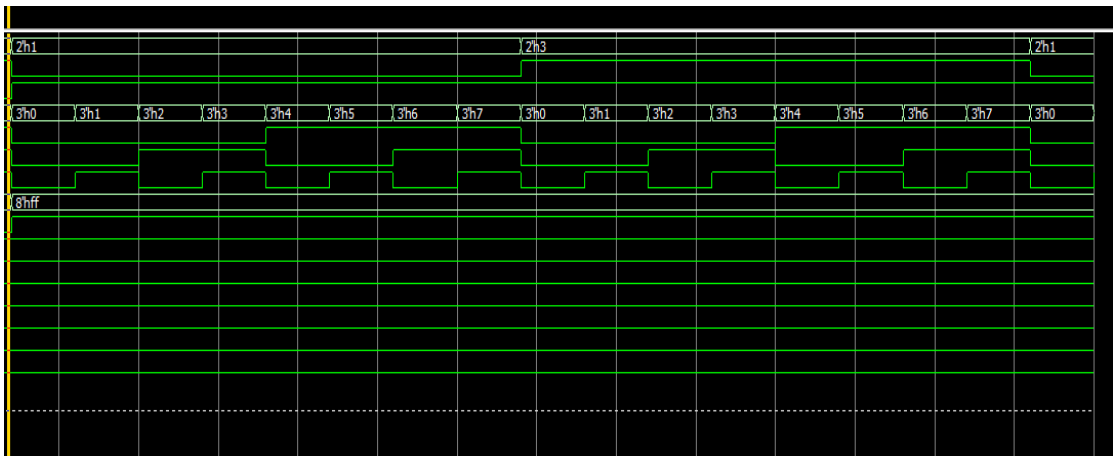


此时输入为（1，1，1）仅有 Y7 为低电平

（2） **modelsim** 仿真波形图



第一部分控制信号有效，结果符合预期，随输入值变化

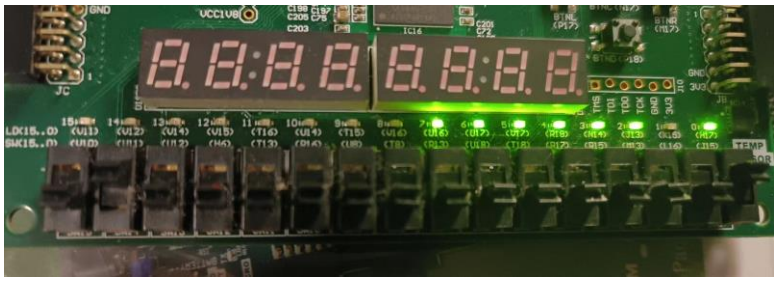


后续控制信号有效，结果符合预期，不随输入变化

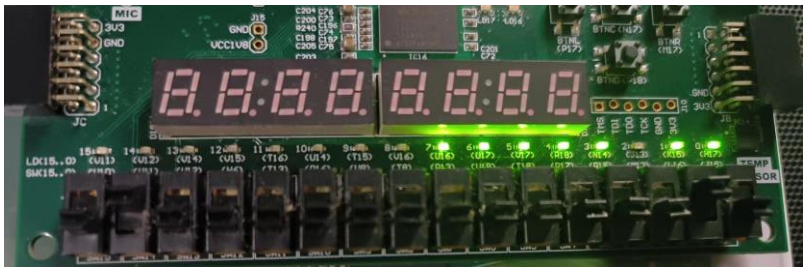
（3） 下板实验结果

控制信号无效，全为高电平

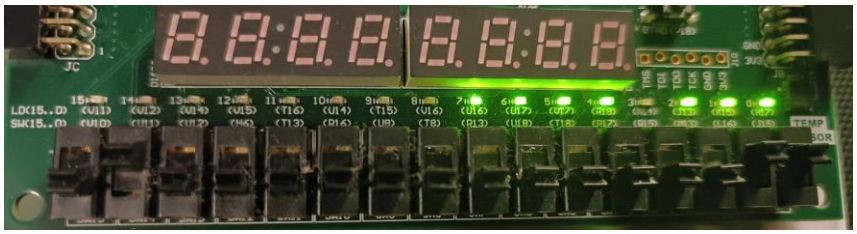
此时输入为（0，0，0）仅有 Y0 为低电平



此时输入为（0，0，1）仅有 Y1 为低电平



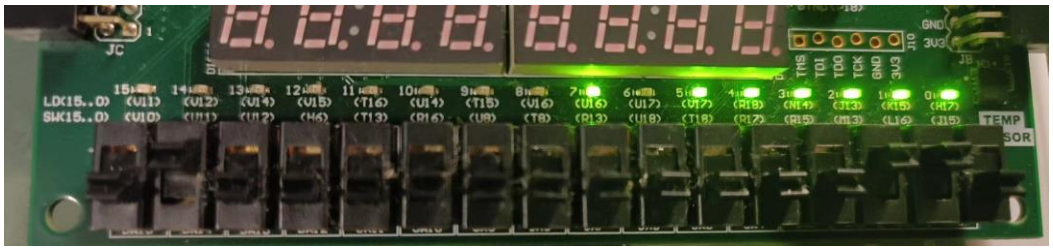
此时输入为（0，1，0）仅有 Y2 为低电平



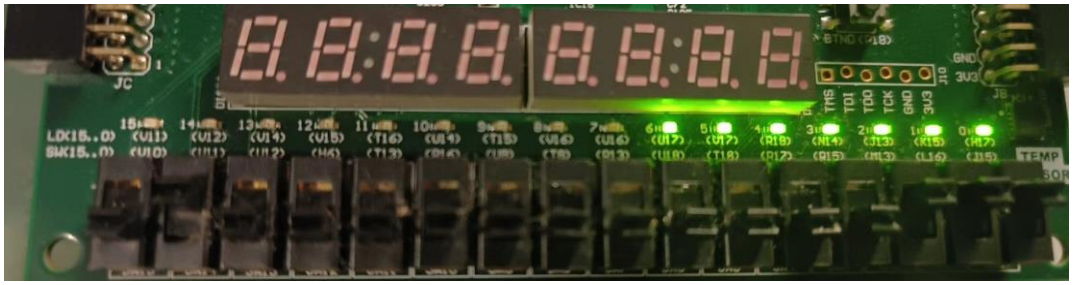
此时输入为（0，1，1）仅有 Y3 为低电平

此时输入为（1，0，0）仅有 Y4 为低电平

此时输入为（1，0，1）仅有 Y5 为低电平



此时输入为（1，1，0）仅有 Y6 为低电平



此时输入为（1，1，1）仅有 Y7 为低电平

2. 7 段数码管译码驱动器

一、 实验内容

（1） 在本次实验中，我们将使用 **Verilog HDL** 语言实现 7 段数码管译码驱动器的设计和仿真。

（2）使用 **verliog** 语言实现 7 段数码管译码驱动器。

二、 模块建模

```
功能描述：该模块实现 3-8 译码器；iData 为 4 位输入，oData 为 6 位输出。控制数字的显示

module display7(
    input[3:0] iData,
    output[6:0] oData
    );
    reg[6:0]oData_tmp;
    always @(*) begin
        case(iData)
            4'b0000:
                oData_tmp=7'b1000000;
            4'b0001:
                oData_tmp=7'b1111001;
            4'b0010:
                oData_tmp=7'b0100100;
            4'b0011:
                oData_tmp=7'b0110000;
            4'b0100:
                oData_tmp=7'b0011001;
            4'b0101:
                oData_tmp=7'b0010010;
            4'b0110:
                oData_tmp=7'b0000010;
            4'b0111:
                oData_tmp=7'b1111000;
            4'b1000:
                oData_tmp=7'b0000000;
            4'b1001:
                oData_tmp=7'b0010000;
            default:
                oData_tmp=7'b1111111;
```

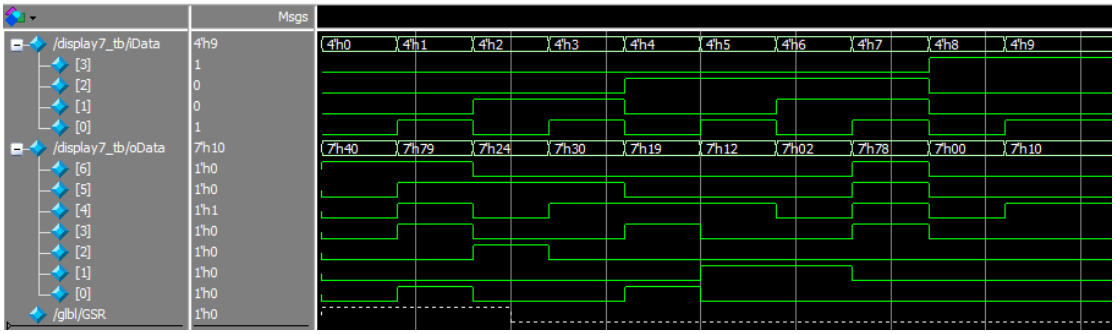
```
        endcase
    end
    assign oData=oData_tmp;
endmodule
```

三、 测试模块建模

```
module display7_tb;
    reg[3:0] iData;
    wire[6:0] oData;

    display7 my_display7(iData,oData);

    initial begin
        iData=4'b0000;
        #40;
        iData=4'b0001;
        #40;
        iData=4'b0010;
        #40;
        iData=4'b0011;
        #40;
        iData=4'b0100;
        #40;
        iData=4'b0101;
        #40;
        iData=4'b0110;
        #40;
        iData=4'b0111;
        #40;
        iData=4'b1000;
        #40;
        iData=4'b1001;
    end

endmodule
```

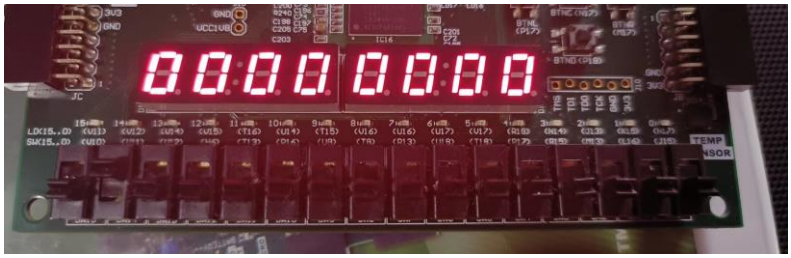
四、 实验结果

（1） modelsim 仿真波形图

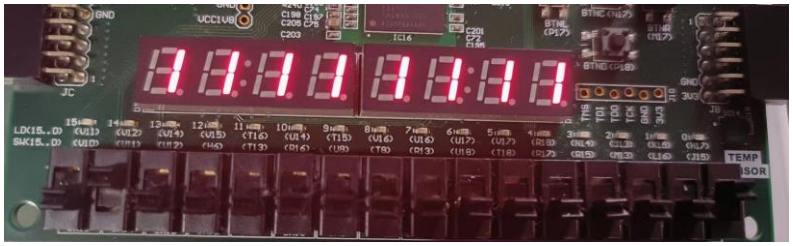


仿真结果如图，符合逻辑功能表中的输入输出

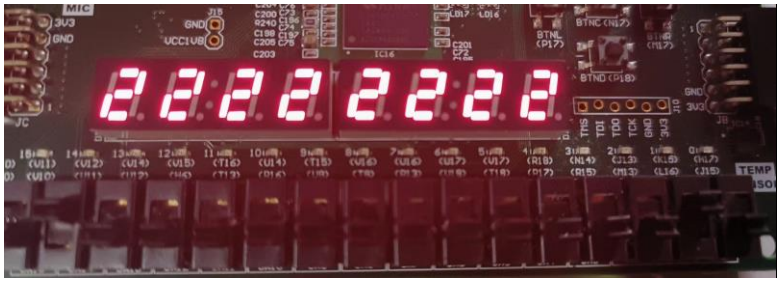
（**2**） 下板实验结果



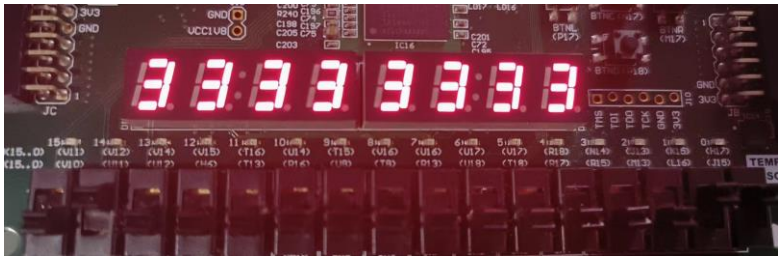
输入为 0b0000，显示数字 0



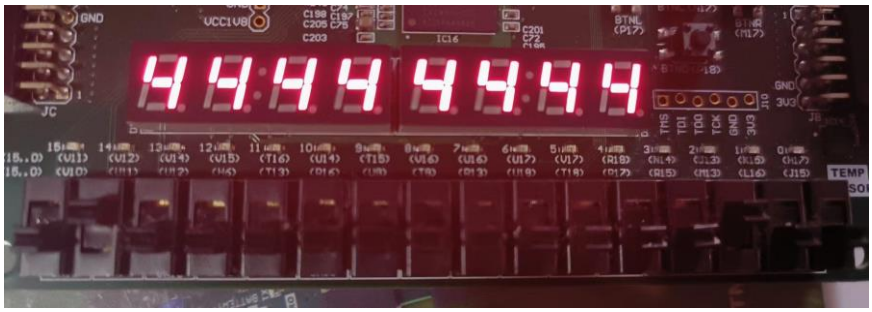
输入为 0b0001，显示数字 1



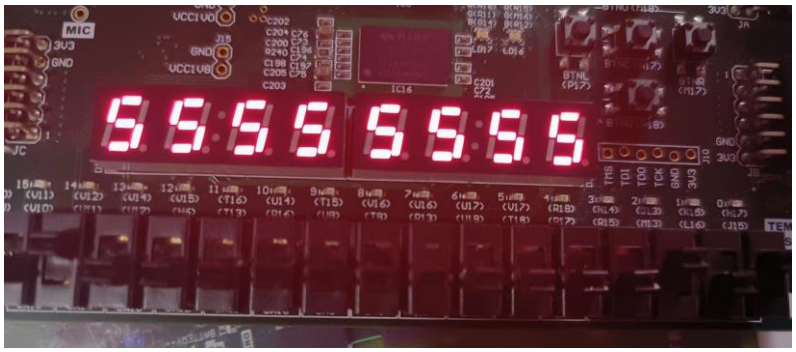
输入为 0b0010，显示数字 2



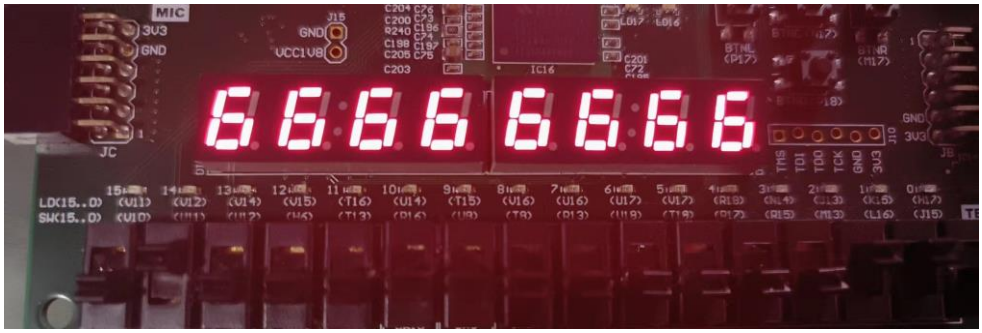
输入为 0b0011，显示数字 3



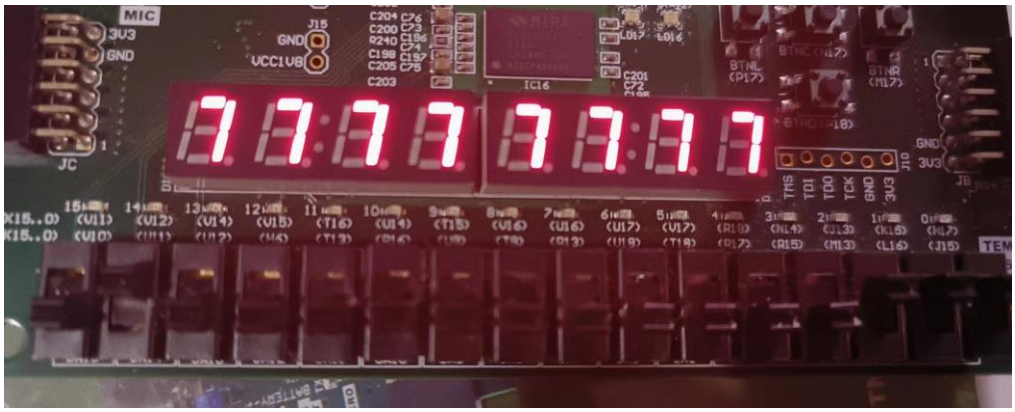
输入为 0b0100，显示数字 4



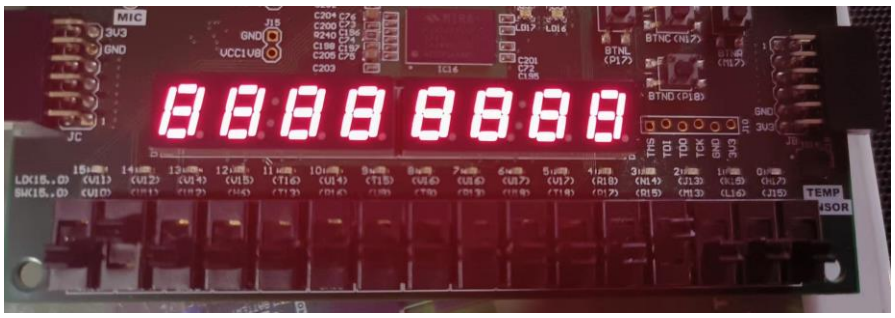
输入为 0b0101，显示数字 5



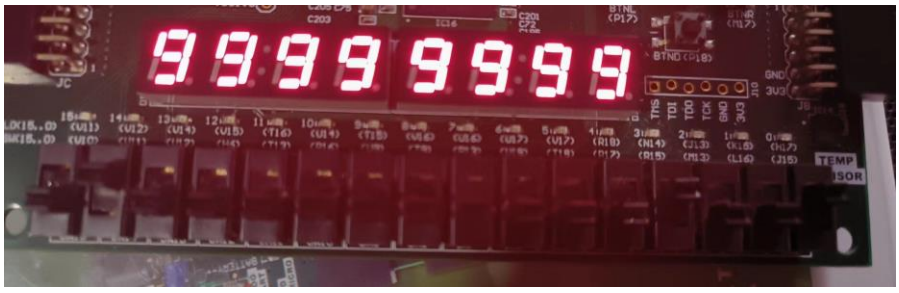
输入为 0b0110，显示数字 6



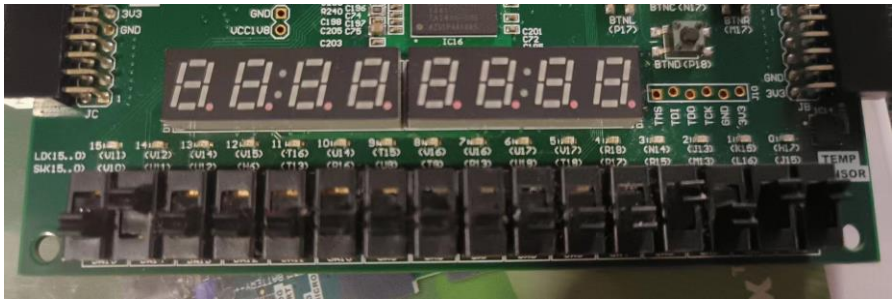
输入为 0b0111，显示数字 7



输入为 0b1000，显示数字 8



输入为 0b1001，显示数字 9



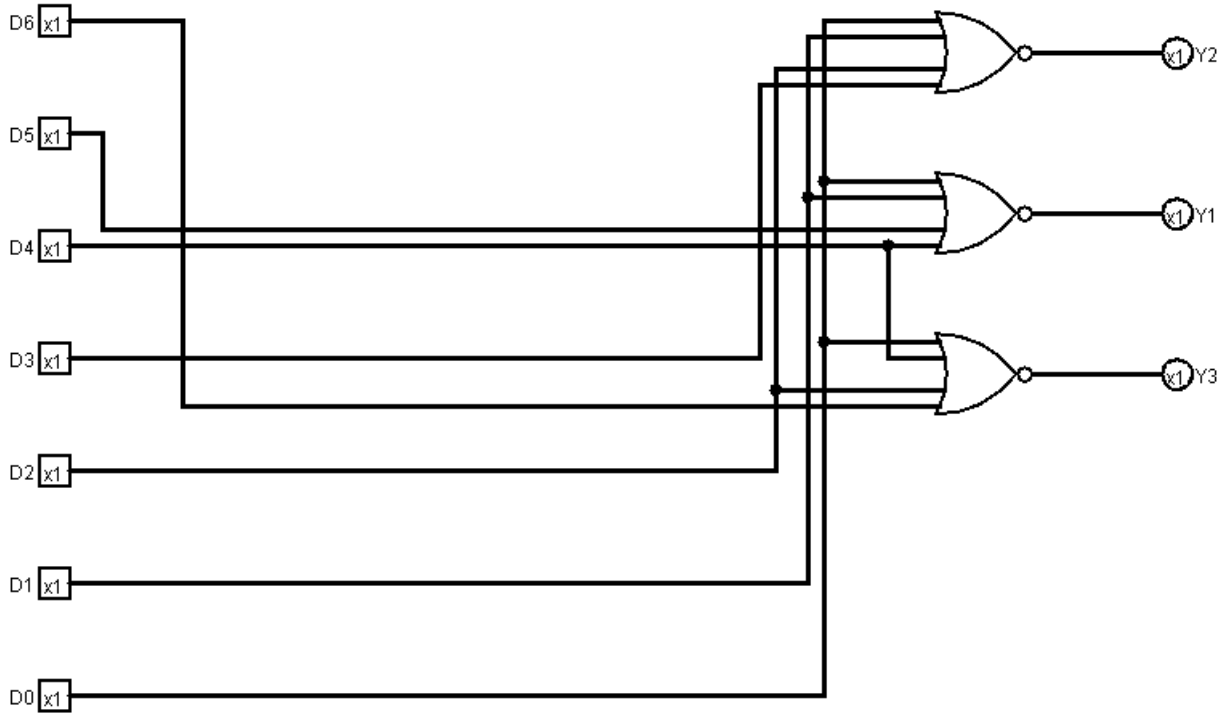
输入为 0b1011，非法，不显示

3. 普通 8-3 译码器

一、 实验内容

（1） 在本次实验中，我们将使用 **Verilog HDL** 语言实现普通 8-3 译码器的设计和仿真。
（2） 使用 **logisim** 画出译码器实验的逻辑图。
（3） 使用 **verliog** 语言实现普通 8-3 译码器。

二、 硬件逻辑图



三、 模块建模

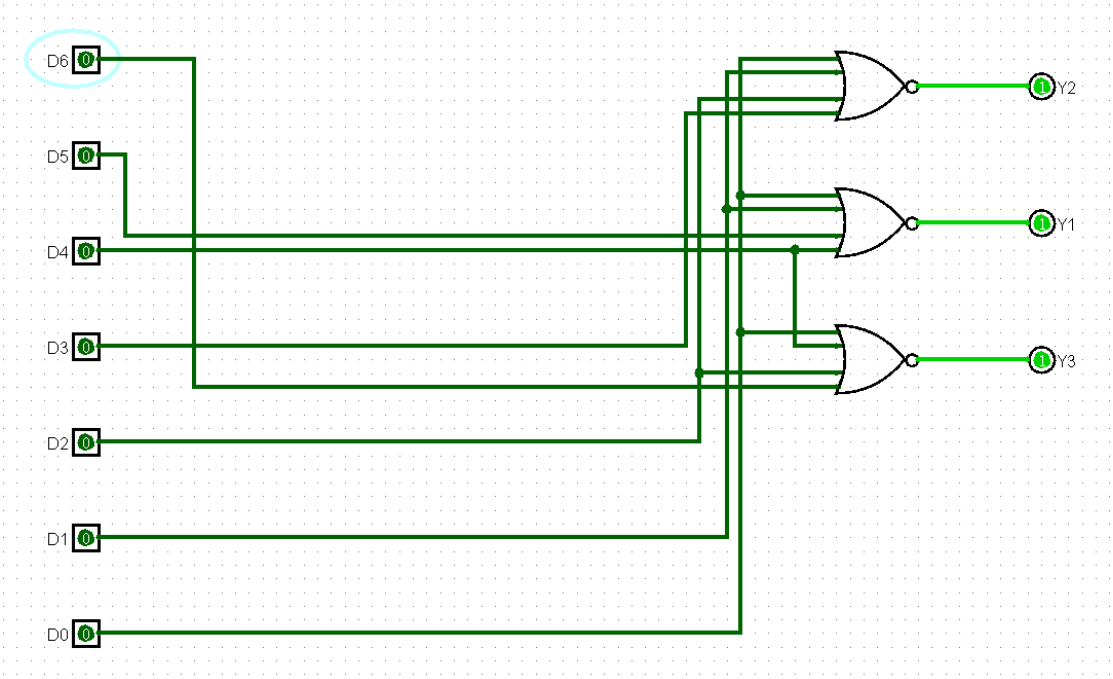
```
//该模块实现功能：普通 8-3 编码器，iData 为 7 位输入，oData 位 3 位输出

module encoder83(
    input[7:0] iData,
    output[2:0] oData
    );
    assign oData[2]=iData[7]|iData[6]|iData[5]|iData[4];

    assign oData[1]=iData[7]|iData[6]|iData[3]|iData[2];

    assign oData[0]=iData[7]|iData[5]|iData[3]|iData[1];
endmodule
```

四、 测试模块建模

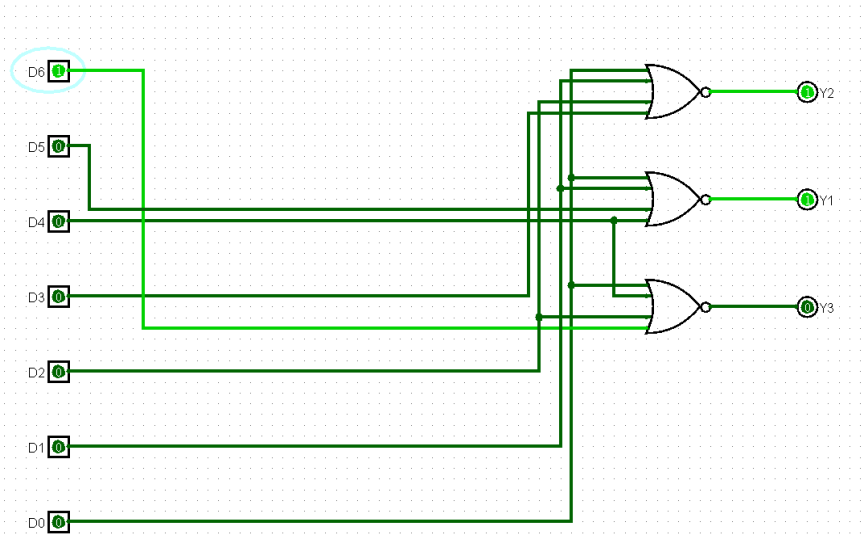
```
module encoder83_tb;

    reg [7:0]iData;
    wire [2:0]oData;

    encoder83 my_encoder(iData,oData);

    initial begin
        iData=8'b10000000;
        #40 iData=8'b01000000;
        #40 iData=8'b00100000;
        #40 iData=8'b00010000;
        #40 iData=8'b00001000;
        #40 iData=8'b00000100;
        #40 iData=8'b00000010;
        #40 iData=8'b00000001;
    end
endmodule
```

五、 实验结果

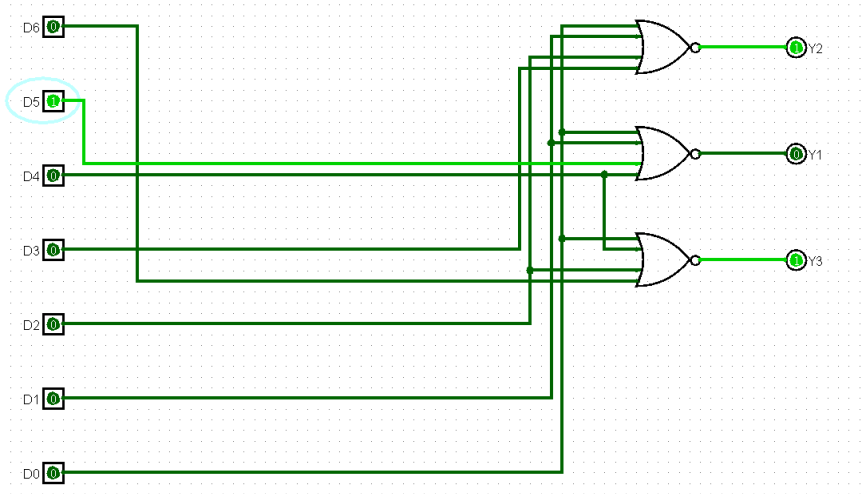
（1） logisim 逻辑验证图



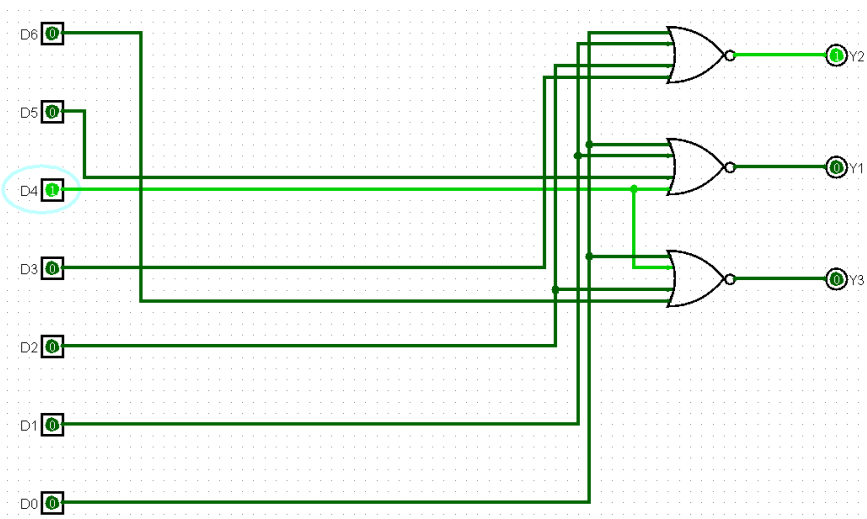
输入为（0，0，0，0，0，0，0，1），输出为 111



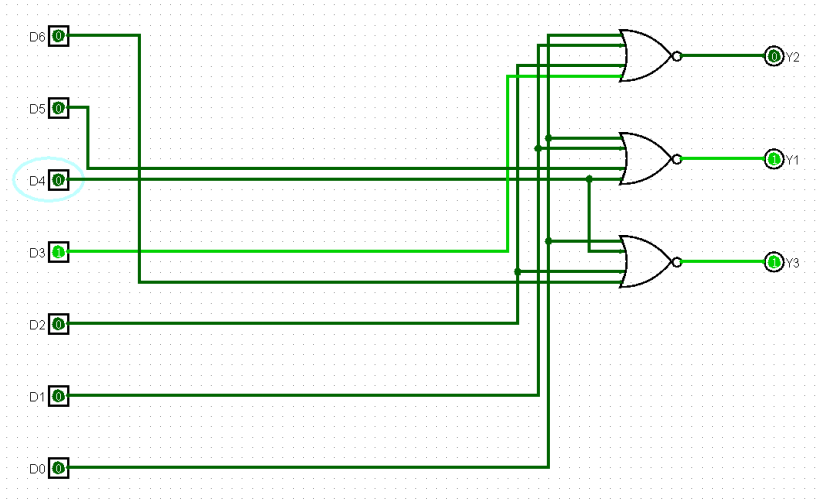
输入为（0，0，0，0，0，0，1，0），输出为 110



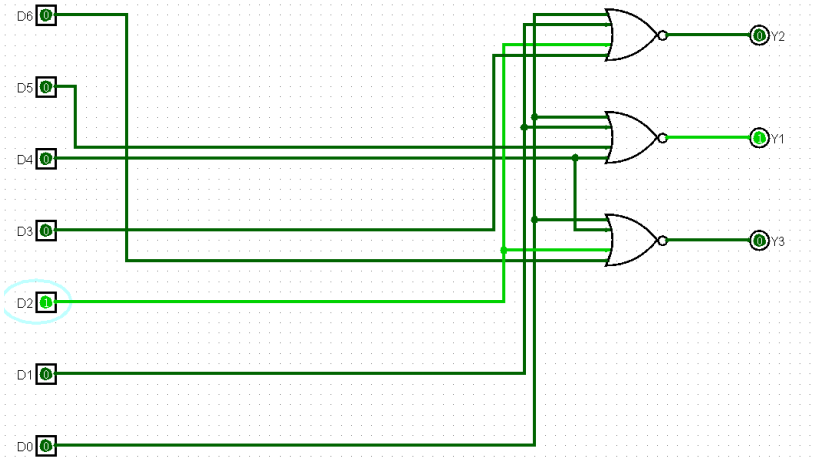
输入为（0，0，0，0，0，1，0，0），输出为 110



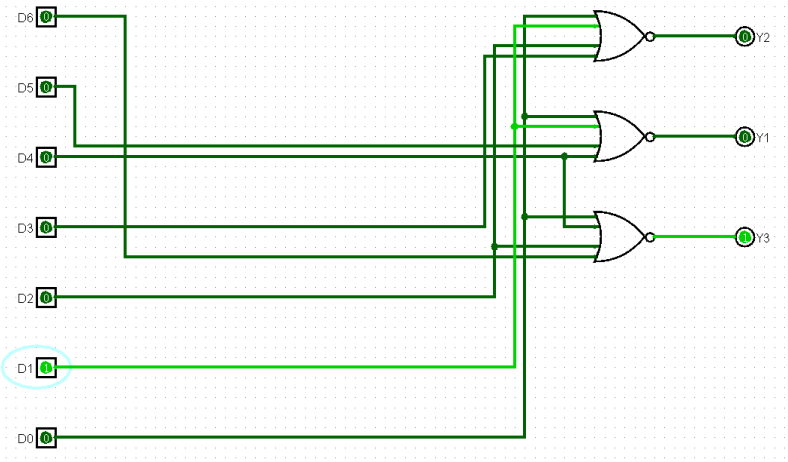
输入为（0，0，0，0，1，0，0，0），输出为 100



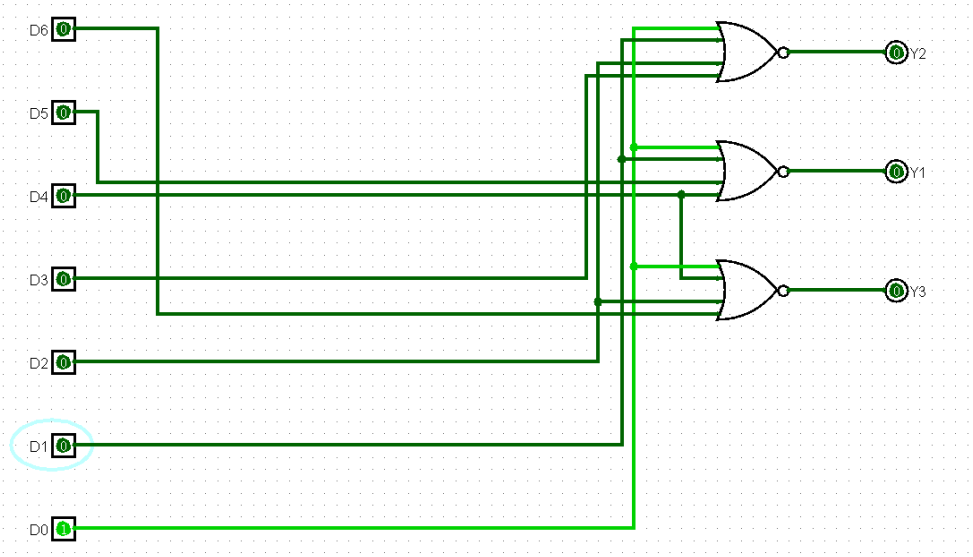
输入为（0，0，0，1，0，0，0，0），输出为 011



输入为（0，0，1，0，0，0，0，0），输出为 010

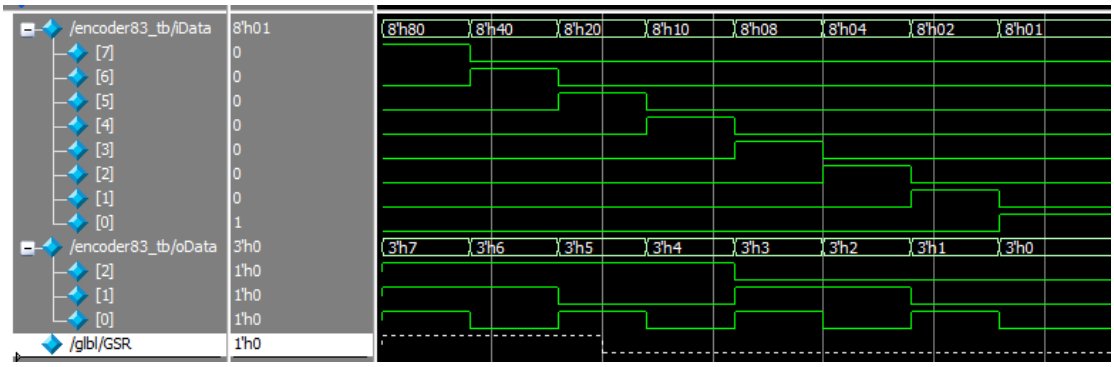


输入为（0，1，，0，0，0，0，0），输出为 001

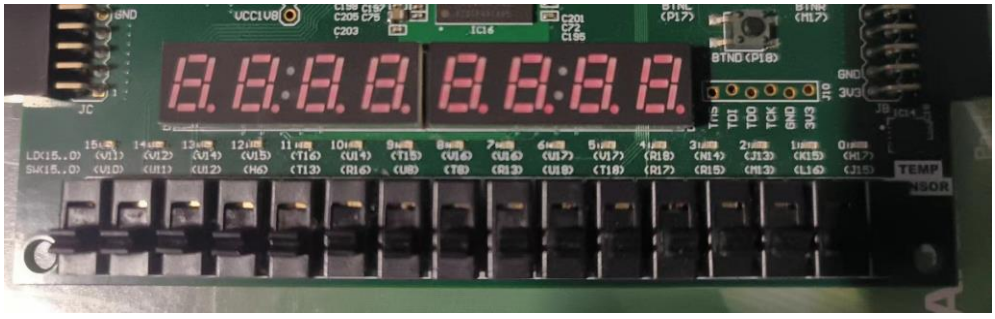


输入为（1，0，0，0，0，0，0，0），输出为 000

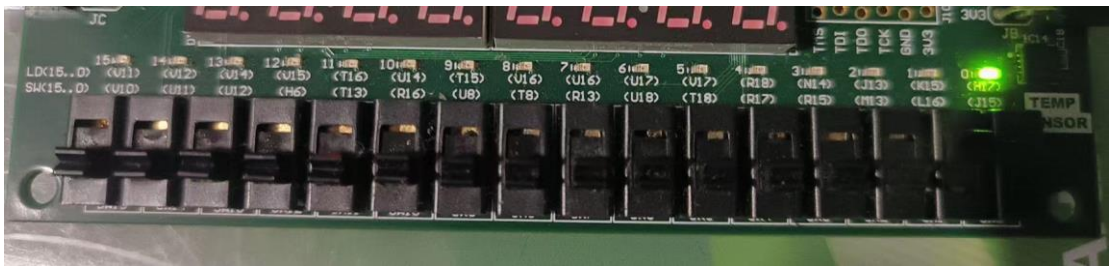
（2） **modelsim** 仿真波形图



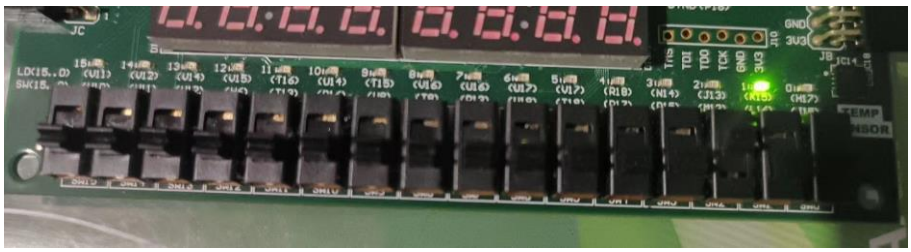
（3） 下板实验结果



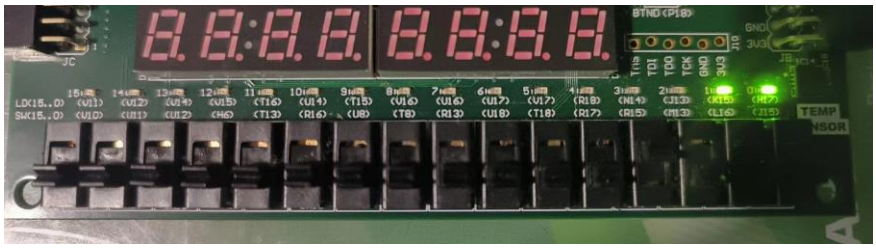
输入为（1，0，0，0，0，0，0，0），输出为 000



输入为（0，1，0，0，0，0，0，0），输出为 001



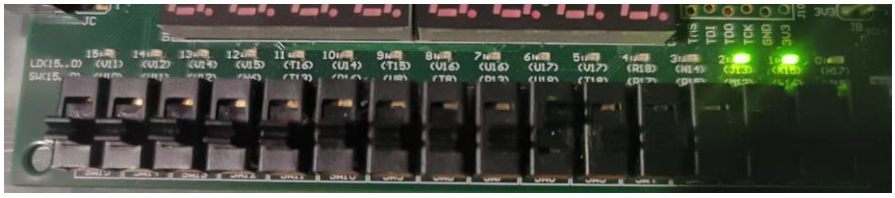
输入为（0，0，1，0，0，0，0，0），输出为 010



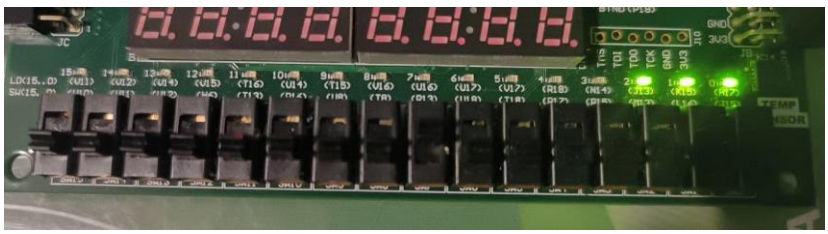
输入为（0，0，0，1，0，0，0，0），输出为 011

输入为（0，0，0，0，1，0，0，0），输出为 100

输入为（0，0，0，0，0，1，0，0），输出为 101



输入为（0，0，0，0，0，0，1，0），输出为 110



输入为（0，0，0，0，0，0，0，1），输出为 111

4. 3-8 优先译码器

一、 实验内容

（1） 在本次实验中，我们将使用 **Verilog HDL** 语言实现 3-8 优先译码器设计和仿真。
（2） 使用 **verliog** 语言实现优先译码器。

二、 模块建模

```
//模块功能：通过 if 语句实现优先译码器，iData 为 8 位输入，oData 为 3 位输出。
module encoder83_Pri(
    input [7:0] iData,
    input iEI,
    output [2:0] oData,
    output oEO
    );

    reg [2:0] oData_tmp;
    reg oEO_tmp;

    always @(*) begin
        oData_tmp = 3'b111;
        if (iEI)
            oEO_tmp = 1;
        else if (iData == 8'b11111111)
            oEO_tmp = 0;
        else begin
            oEO_tmp = 1;
            if (!iData[7])
                oData_tmp = 3'b000;
            else if (!iData[6])
                oData_tmp = 3'b001;
            else if (!iData[5])
                oData_tmp = 3'b010;
            else if (!iData[4])
                oData_tmp = 3'b011;
            else if (!iData[3])
                oData_tmp = 3'b100;
            else if (!iData[2])
                 oData_tmp = 3'b101;
            else if (!iData[1])
                 oData_tmp = 3'b110;
            else if (!iData[0])
```

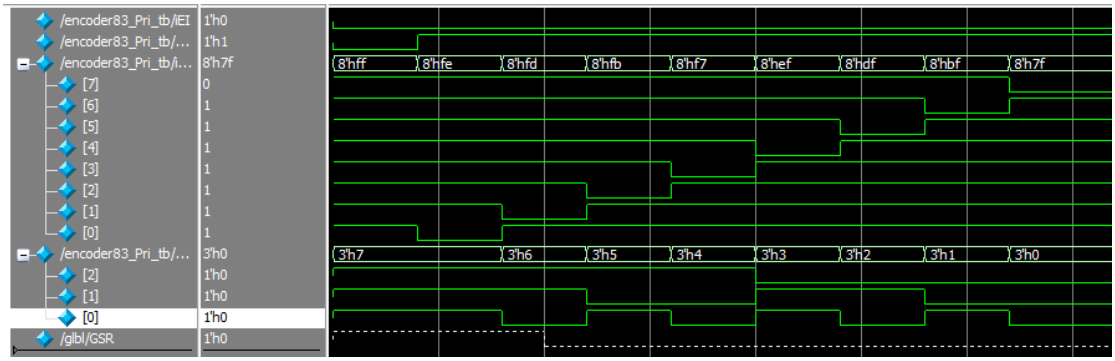
```
                oData_tmp = 3'b111;
            else
                ;
        end
    end
    assign oEO=oEO_tmp;
    assign oData=oData_tmp;
endmodule
```

三、 测试模块建模

```
module encoder83_Pri_tb;
    reg iEI;
    wire oEO;
    reg [7:0]iData;
    wire [2:0] oData;

    encoder83_Pri my_encoder_pri(iData,iEI,oData,oEO);

    initial begin
        iEI=0;
    end

    initial begin
        //iData=8'b00000000;
        iData=8'b11111111;
        #40 iData=8'b11111110;
        #40 iData=8'b11111101;
        #40 iData=8'b11111011;
        #40 iData=8'b11110111;
        #40 iData=8'b11101111;
        #40 iData=8'b11011111;
        #40 iData=8'b10111111;
        #40 iData=8'b01111111;
    end
endmodule
```

四、 实验结果

（1） modelsim 仿真波形图


/encoder83_Pri_tb/iEI
/encoder83_Pri_tb/...
/encoder83_Pri_tb/i...
[7]
[6]
[5]
[4]
[3]
[2]
[1]
[0]
/encoder83_Pri_tb/...
[2]
[1]
[0]
/glbl/GSR
1'h0
1'h1
8'h7f
0
1
1
1
1
1
1
1
3'h0
1'h0
1'h0
1'h0
8'hff
8'hfe
8'hfd
8'hfb
8'hf7
8'hef
8'hdf
8'hbf
8'h7f
3'h7
3'h6
3'h5
3'h4
3'h3
3'h2
3'h1
3'h0

（2） 下板实验结果