# 同济大学计算机系

# 数字逻辑课程综合实验报告


学　　号 2352595

姓　　名 张嘉麟

专　　业 计算机科学与技术

授课老师 郭玉臣

# 一、实验内容

在本次实验中，使用 Verilog HDL 语言实现 3-8 译码器、8-3 编码器 以及 七段数码管 的设计和仿真。这些电路模块在数字电路和微电子系统中有着广泛的应用。

3-8 译码器：用于将 3 位二进制输入信号转换为 8 条输出信号的译码电路，可用于地址译码等场景。分为普通译码器和七段数码管译码器。

七段数码管：是一种广泛用于数码显示的元件，通过控制七段 LED 灯的点亮组合来显示 0-9 等数字。

8-3 编码器：将 8 条输入信号压缩为 3 位二进制输出，实现信息的压缩传输。分为普通编码器和具有优先级的 8-3 编码器。

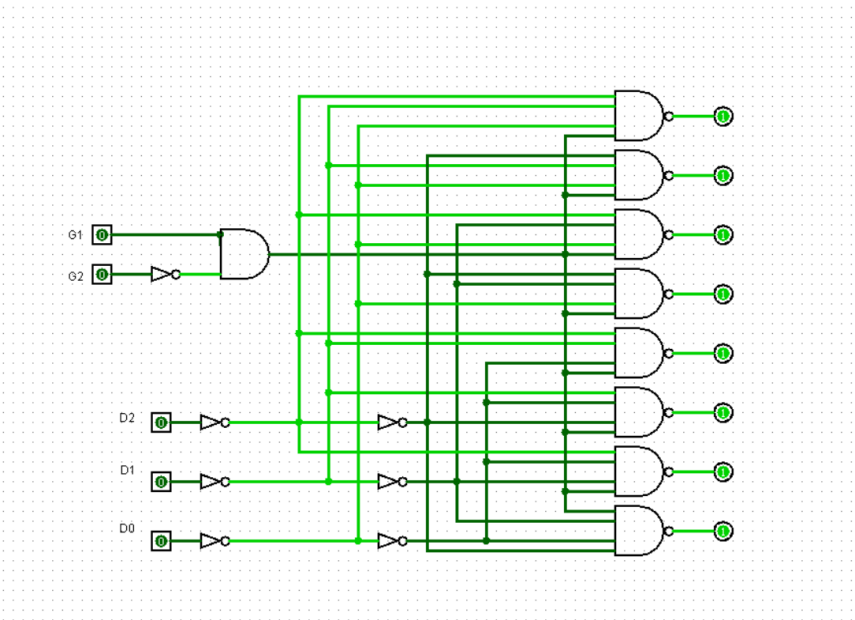
具有优先级的 8-3 编码器是一种数字编码器，用于将 8 条输入信号转换成 3 位二进制输出信号，并根据输入信号的优先级进行编码。当多个输入同时有效时，它会优先编码具有较高优先级的输入信号。通常在数字电路中应用于需要处理多个信号但仅输出一个信号的场景，如中断处理系统等。

# 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）

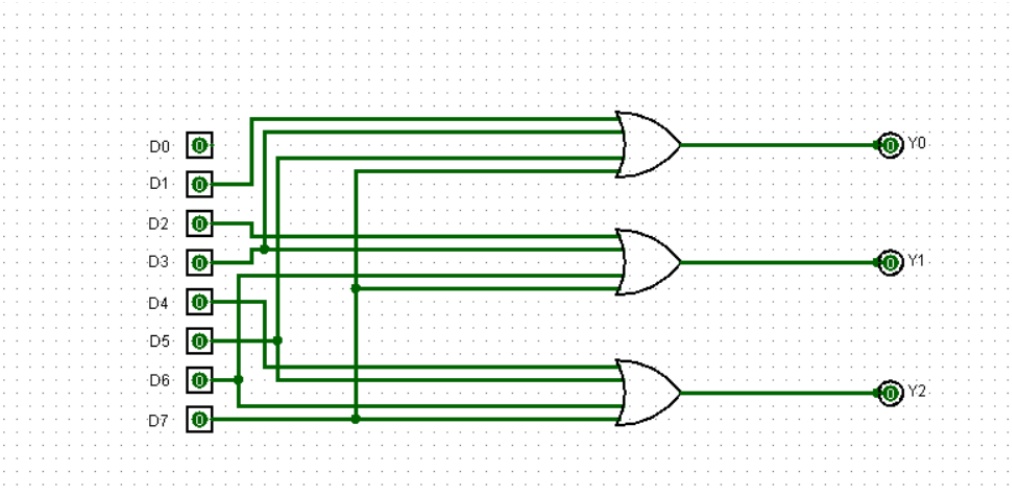
## 2.1 译码器

### 2.1.1 decoder



## 2.2 编码器

### 2.2.1 encoder83




# 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）

## 2.1 译码器

**2.1.1 decoder**

```
module decoder(
	input [2:0] iData,
	input [1:0] iEna,
	output [7:0] oData
);
	assign oData = (iEna == 2'b10 ? ~(8'b00000001 << iData) : 8'b11111111);
endmodule
```

**2.1.2 display7**

```
module display7(
	input [3:0] iData,
	output reg [6:0] oData
);
	always @(*)
	begin
		case(iData)
			4'b0000 : oData = 7'b1000000;
```

```
                4'b0001 : oData = 7'b1111001;
                4'b0010 : oData = 7'b0100100;
                4'b0011 : oData = 7'b0110000;
                4'b0100 : oData = 7'b0011001;
                4'b0101 : oData = 7'b0010010;
                4'b0110 : oData = 7'b0000010;
                4'b0111 : oData = 7'b1111000;
                4'b1000 : oData = 7'b0000000;
                4'b1001 : oData = 7'b0010000;
            endcase
        end

endmodule
```

## 2.2 编码器

### 2.2.1 encoder83

```
module encoder83(
        input [7:0] iData,
        output reg [2:0] oData
);
        always@ (*)
            case(iData)
                8'b10000000 : oData = 3'b111;
                8'b01000000 : oData = 3'b110;
                8'b00100000 : oData = 3'b101;
                8'b00010000 : oData = 3'b100;
                8'b00001000 : oData = 3'b011;
                8'b00000100 : oData = 3'b010;
                8'b00000010 : oData = 3'b001;
                8'b00000001 : oData = 3'b000;
            endcase
endmodule
```

### 2.2.2 encoder83_pri

```
module encoder83_Pri(
        input [7:0] iData,
        input iEI,
        output reg [2:0] oData,
        output reg oEO
);
        always@ (*)
        begin
```

```
        if(iEI == 1)
            {oData, oEO} =    4'b1110;
        else
        begin
            casex(iData)
                8'b11111111 : {oData, oEO} = 4'b1110;
                8'b11111110 : {oData, oEO} = 4'b1111;
                8'b1111110x : {oData, oEO} = 4'b1101;
                8'b111110xx : {oData, oEO} = 4'b1011;
                8'b11110xxx : {oData, oEO} = 4'b1001;
                8'b1110xxxx : {oData, oEO} = 4'b0111;
                8'b110xxxxx : {oData, oEO} = 4'b0101;
                8'b10xxxxxx : {oData, oEO} = 4'b0011;
                8'b0xxxxxxx : {oData, oEO} = 4'b0001;
            endcase
        end
    end

endmodule
```

# 四、测试模块建模

（要求列写各建模模块的 test bench 模块代码）

## 2.1 译码器

### 2.1.1 decoder

```
module decoder_tb;
    reg [3 : 0] iData;
    reg [2 : 0] iEna;
    wire [7 : 0] oData;
    initial
    begin
        for(iEna = 0; iEna <= 2'b11; iEna = iEna + 2'b01)
            for(iData = 0; iData <= 3'b111; iData = iData + 3'b001)
                begin
                    #20;
                end
    end
    decoder
    decoder_instc(iData[2 : 0], iEna[1 : 0], oData);
endmodule
```

### 2.1.2 display7

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

    initial
        begin
        for(iData = 0; iData <= 4'b1001; iData = iData + 4'b0001)
            #20;
        end

    display7 display7_inst(iData, oData);
endmodule
```

## 2.2 编码器

### 2.2.1 encoder83

```
module encoder83_tb;
    reg [8:0] iData;
    wire [2:0] oData;

    initial
    begin
        for(iData = 8'b00000001; iData <= 8'b10000000; iData = iData << 1)
        begin
            #20;
        end
    end
    encoder83 encoder83_inst(iData[7 : 0], oData);
endmodule
```

### 2.2.2 encoder83_pri

```
module encoder83_Pri_tb;
    reg [8:0] iData;
    reg iEI;
    wire [2:0] oData;
    wire oEO;
    initial
    begin
        iEI = 0;
        for(iData = 8'b01111111; iData <= 8'b11111111; iData = iData +
8'b00000001)
            #5;
```

```
    end
    encoder83_Pri encoder83_Pri_inst(iData[7:0], iEI, oData, oEO);
endmodule
```

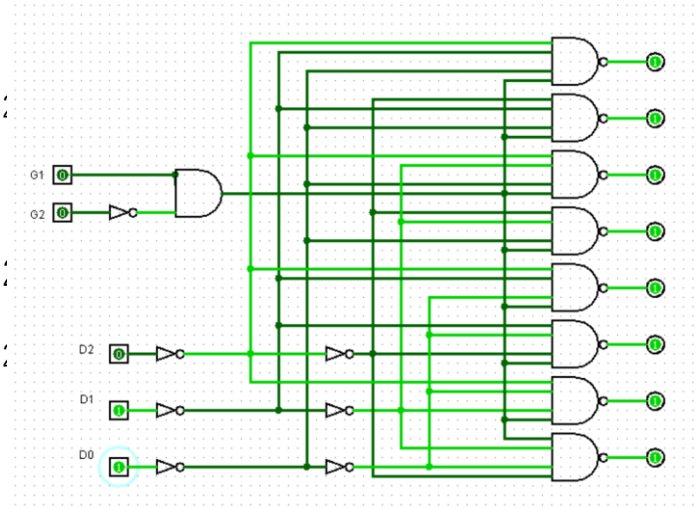
# 五、实验结果

（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））

## 2.1 译码器

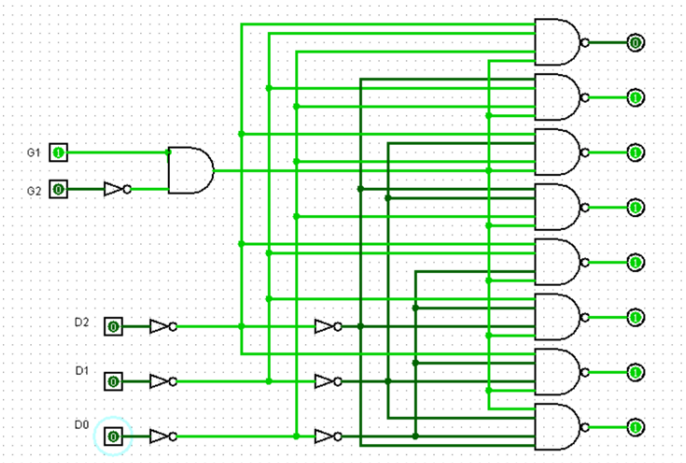
### 2.1.1 decoder

1.logisim 逻辑验证图

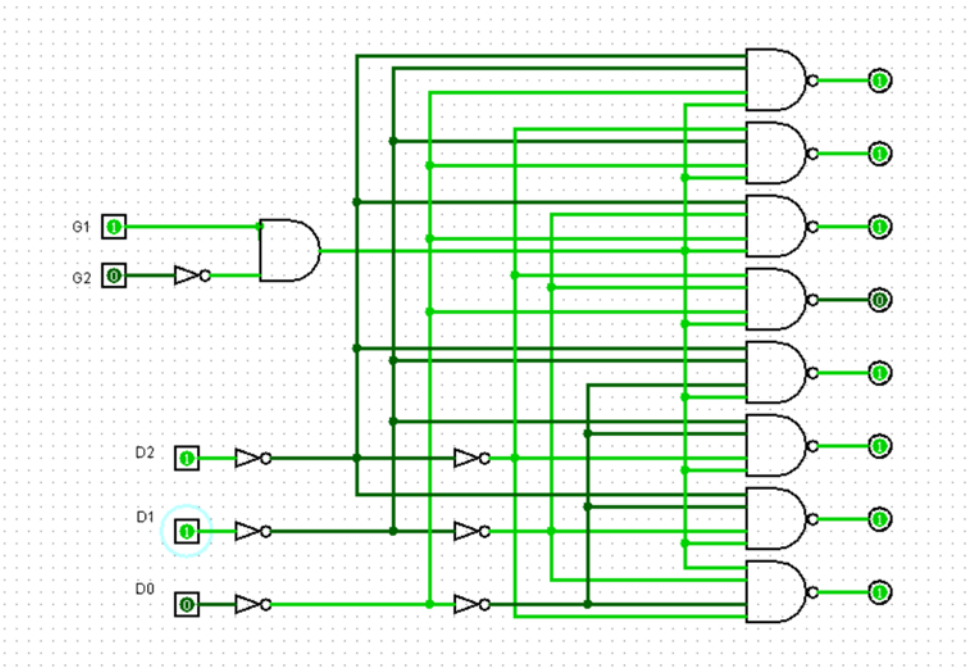



2.2.2 encoder83_pri

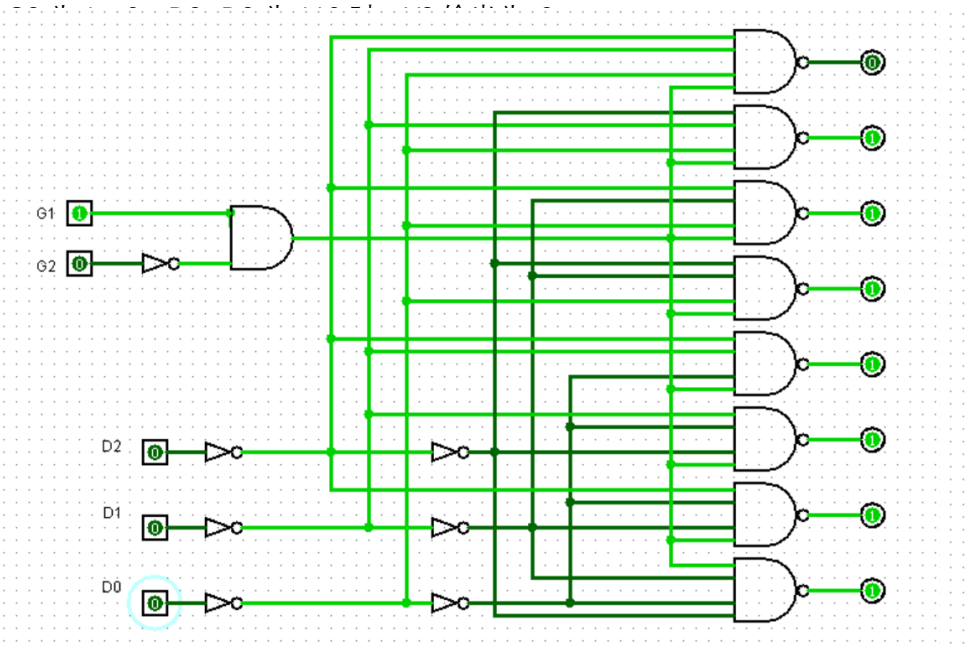
当 G1、G2 不为 1、0 时，输出结果均为 1

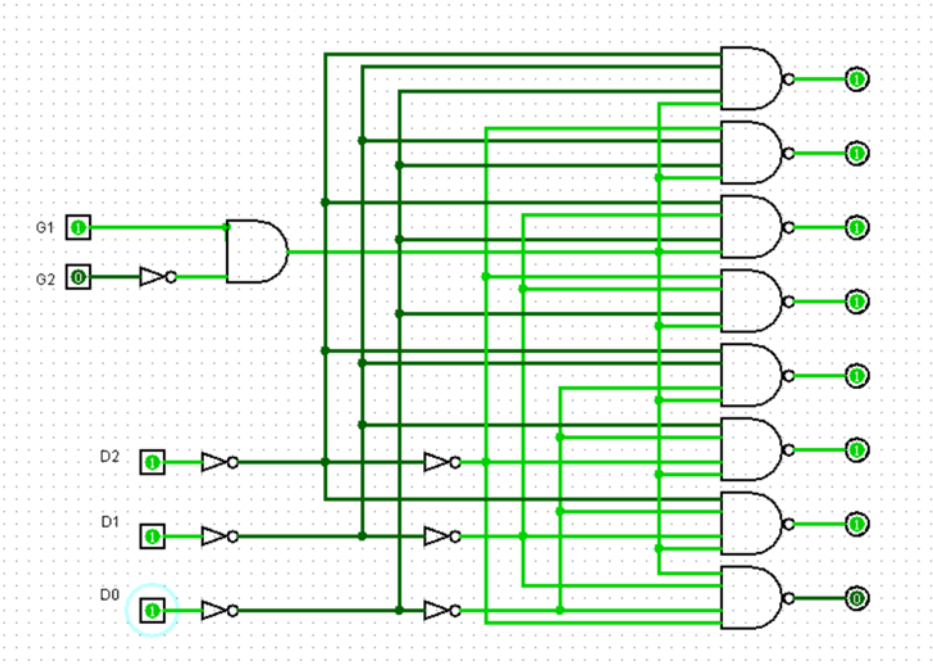



当 G1、G2 为 1、0，D2~D0 为 000 时，Y0 输出为 0



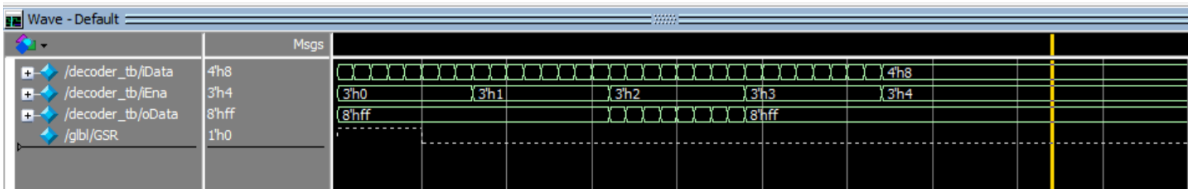

当 G1、



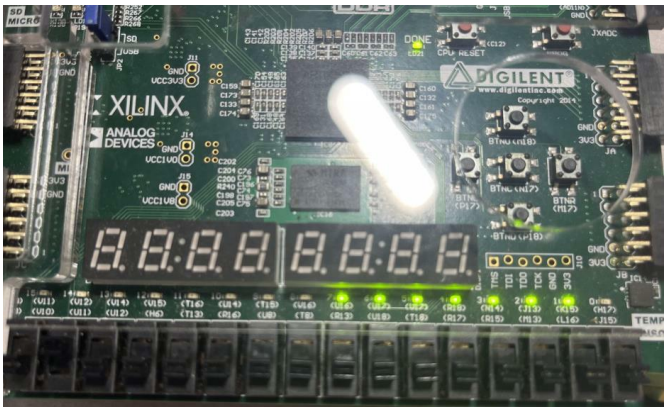


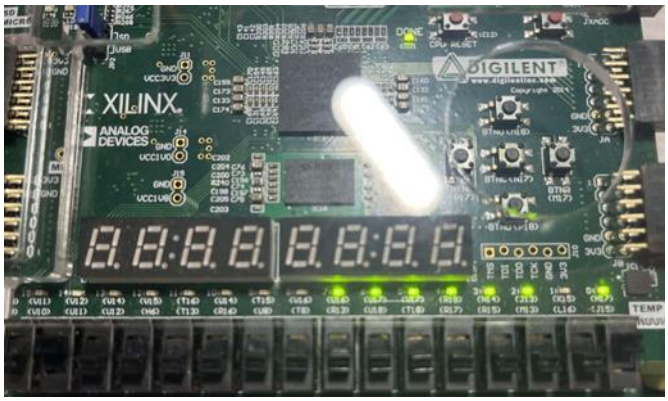

当 G1、G2 为 1、0，D2~D0 为 111 时，Y7 输出为 0。

2.modelsim 仿真波形图




3.下板结果





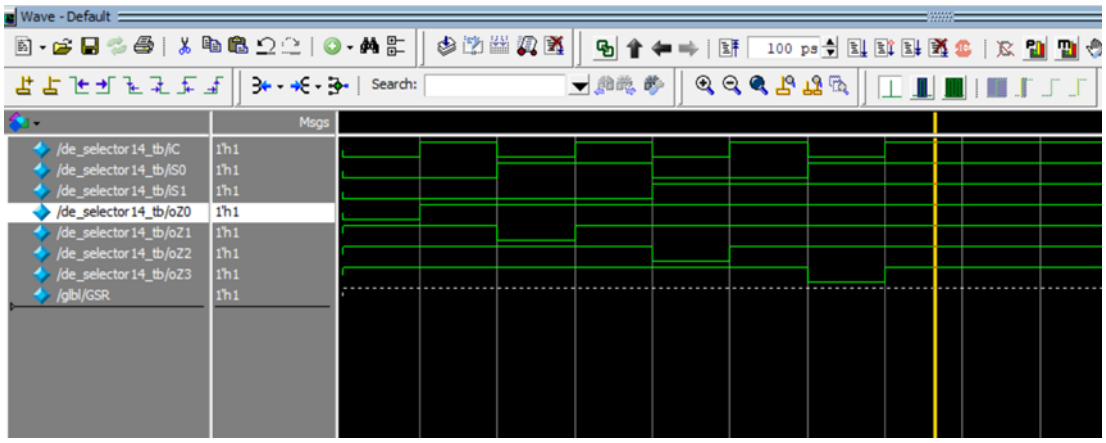
分别对应上述四种情况。

### 2.1.2 display7

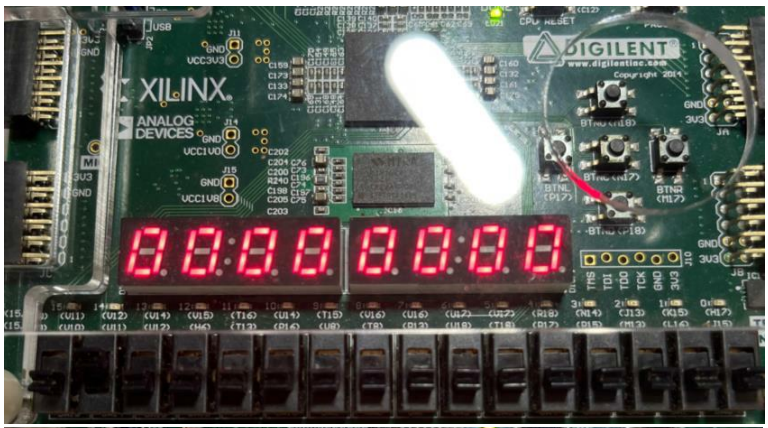
1.logisim 逻辑验证图

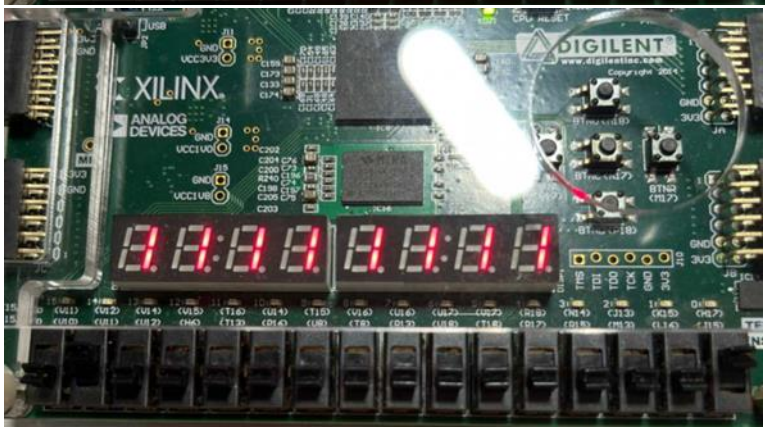
无

2.modelsim 仿真波形图


Wave - Default
Search:
Msgs
/de_selector14_tb/iC
/de_selector14_tb/iS0
/de_selector14_tb/iS1
/de_selector14_tb/oZ0
/de_selector14_tb/oZ1
/de_selector14_tb/oZ2
/de_selector14_tb/oZ3
/glbl/GSR

3.下板结果


XILINX
DIGILENT
ANALOG DEVICES


XILINX
DIGILENT
ANALOG DEVICES

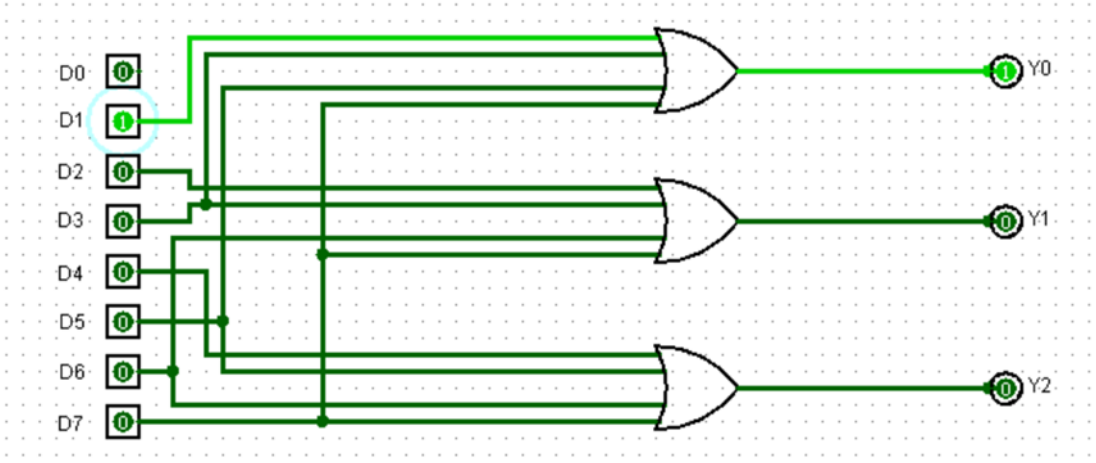

XILINX
DIGILENT
ANALOG DEVICES

省略其他情况。

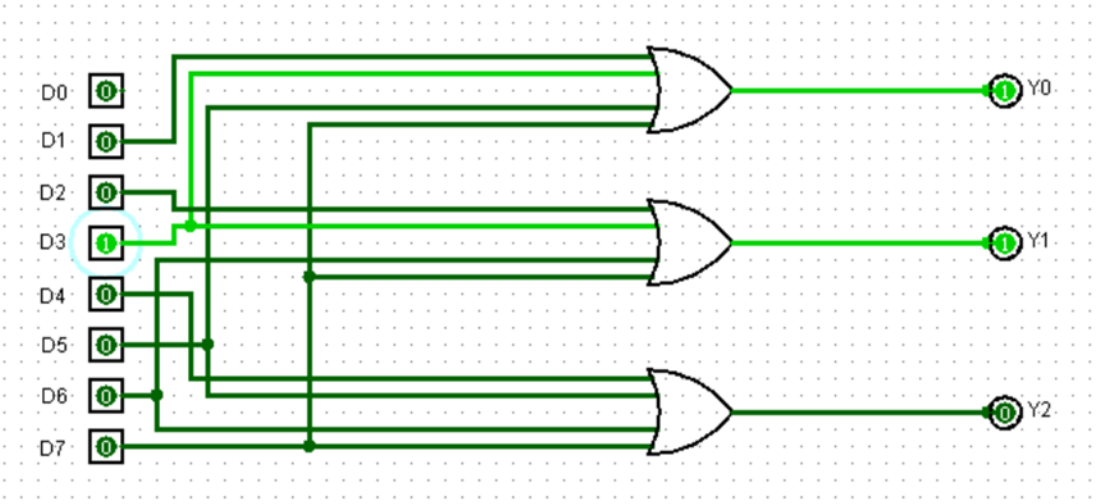
## 2.2 编码器

### 2.2.1 encoder83

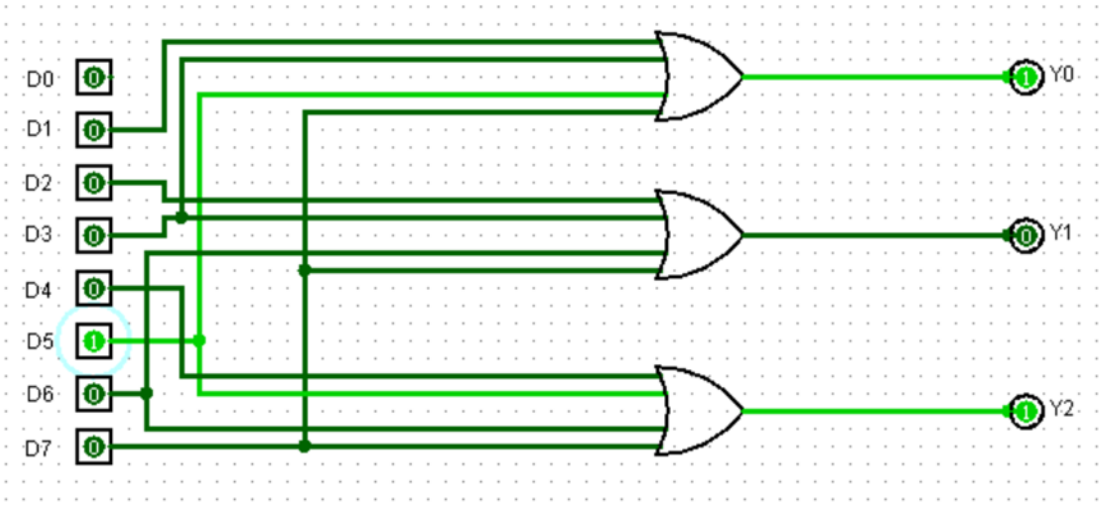
1.logisim 逻辑验证图



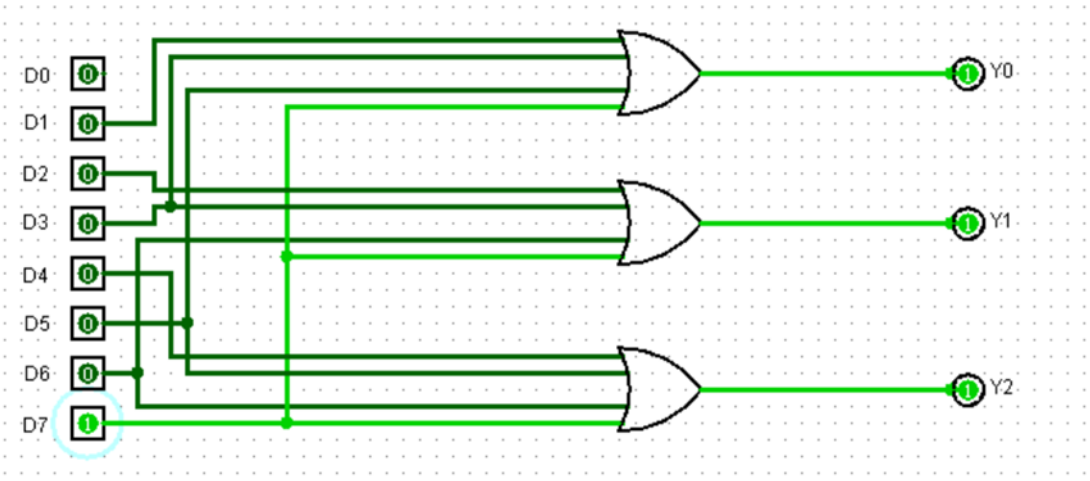

D1 输入为 1，其他为 0 时；Y1、Y2 输出为 0，Y0 为 0




D3 输入为 1，其他为 0 时；Y0、Y1 输出为 1，Y2 为 0

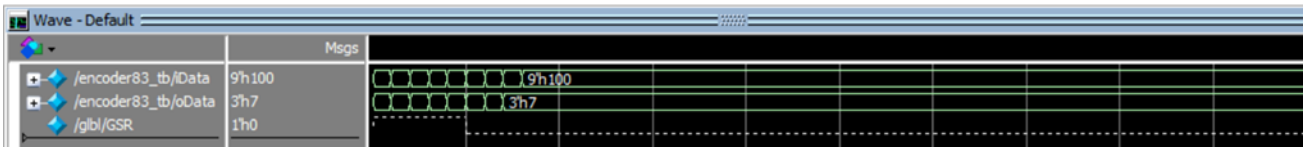



D5 输入为 1，其他为 0 时；Y1、Y2 输出为 1，Y1 为 0

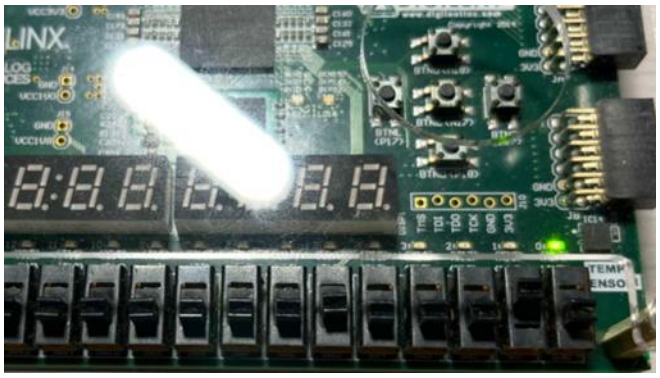



D7 输入为 1，其他为 0 时；Y0、Y1、Y2 输出为 1

2.modelsim 仿真波形图




3.下板结果

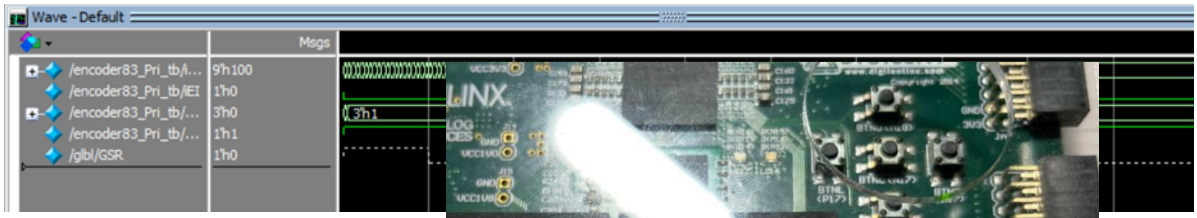


## 2.2.2 encoder83_pri

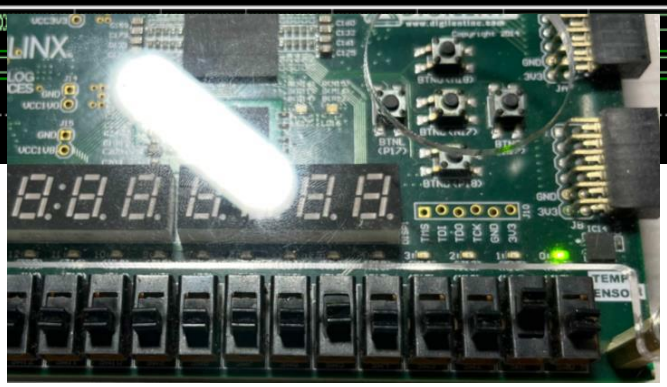
1.logisim 逻辑验证图

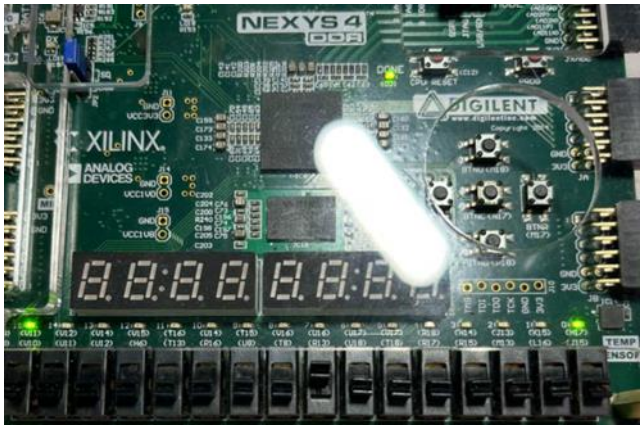
无

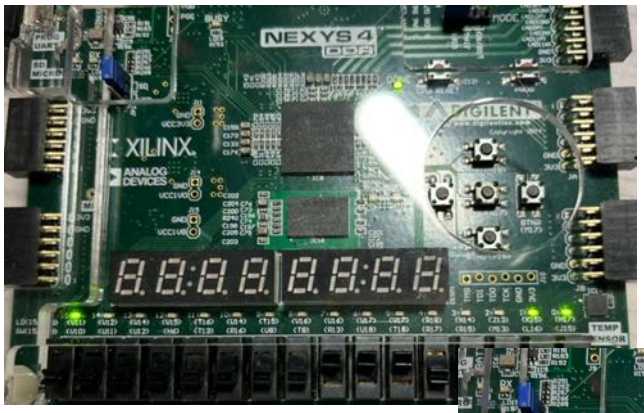
2.modelsim 仿真波形图

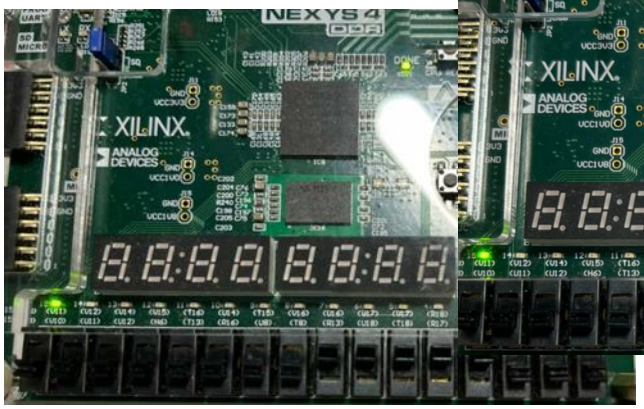


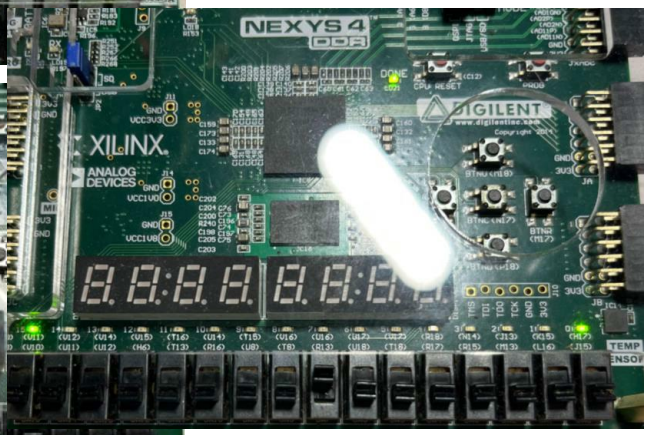

3.下板结果




NEXYS 4
DDR
XILINX
ANALOG DEVICES
DIGILENT


NEXYS 4
DDR
XILINX
ANALOG DEVICES
DIGILENT


NEXYS 4
DDR
XILINX
ANALOG DEVICES


NEXYS 4
DDR
XILINX
ANALOG DEVICES
DIGILENT