同济大学计算机系

数字逻辑课程实验报告

学　　号 **2253156**

姓　　名 闫浩扬

专　　业 计算机科学与技术（精英班）

授课老师 郭玉臣

一、 实验内容

- **实验介绍**

  在本次实验中，我们将使用 Verilog HDL 语言实现 3-8 译码器、8-3 普通编码器，8-3 优先编码器以及七段数码管的设计和仿真。

- **实验目标**

  深入了解译码器、编码器、优先编码器原理。

  使用 logisim 画出译码器以及编码器实验的逻辑图。

  学习使用 Verilog HDL 语言设计实现译码器、编码器。

二、 硬件逻辑图

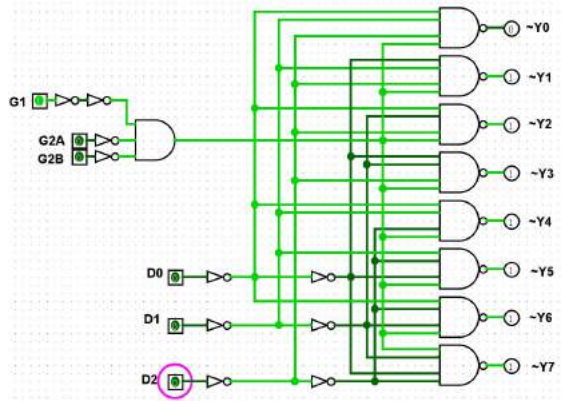
1) 3-8 译码器

实现译码功能的组合逻辑电路称为译码器，它的输入是一组二进制代码，输出是一组高低电平信号。所要建模的 3-8 译码器及真值表如图所示，它有三个编码输入、八个输出和二个使能输入（G1，G2）。作为译码器使用时，使能端必须满足 G1=1，G2=0。

真值表

G_1	G_2	D_2	D_1	D_0	$\overline{Y_0}$	$\overline{Y_1}$	$\overline{Y_2}$	$\overline{Y_3}$	$\overline{Y_4}$	$\overline{Y_5}$	$\overline{Y_6}$	$\overline{Y_7}$
×	1	×	×	×	1	1	1	1	1	1	1	1
0	×	×	×	×	1	1	1	1	1	1	1	1
1	0	0	0	0	0	1	1	1	1	1	1	1
1	0	0	0	1	1	0	1	1	1	1	1	1
1	0	0	1	0	1	1	0	1	1	1	1	1
1	0	0	1	1	1	1	1	0	1	1	1	1
1	0	1	0	0	1	1	1	1	0	1	1	1
1	0	1	0	1	1	1	1	1	1	0	1	1
1	0	1	1	0	1	1	1	1	1	1	0	1
1	0	1	1	1	1	1	1	1	1	1	1	0

logisim 逻辑图



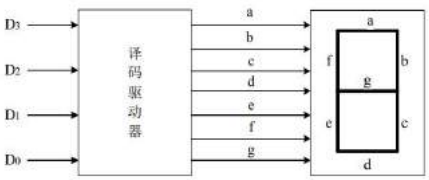
逻辑表达式

$Y_0 = \sim A_2 \& \sim A_1 \& \sim A_0$　$Y_1 = \sim A_2 \& \sim A_1 \& A_0$　$Y_2 = \sim A_2 \& A_1 \& \sim A_0$　$Y_3 = \sim A_2 \& A_1 \& A_0$

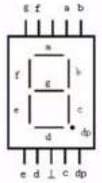
$Y_4 = A_2 \& \sim A_1 \& \sim A_0$　$Y_5 = A_2 \& \sim A_1 \& A_0$　$Y_6 = A_2 \& A_1 \& \sim A_0$　$Y_7 = A_2 \& A_1 \& A_0$

2) 七段数码管译码驱动器

下图为所要建模的七段数码管译码驱动原理图，它由译码驱动器和荧光数码管组成。荧光数码管是分段式半导体显示器件，7 个发光二极管组成 7 个发光段，发光二极管可以将电能转换成光能，从而发出清晰悦目的光线。本实验采用的是共阳极电路，故译码器的输出 a~g 分别加到 7 个阴极上。只有在阴极上呈低电平的二极管导通发光，显示 0~9 中相应的十进制数字。

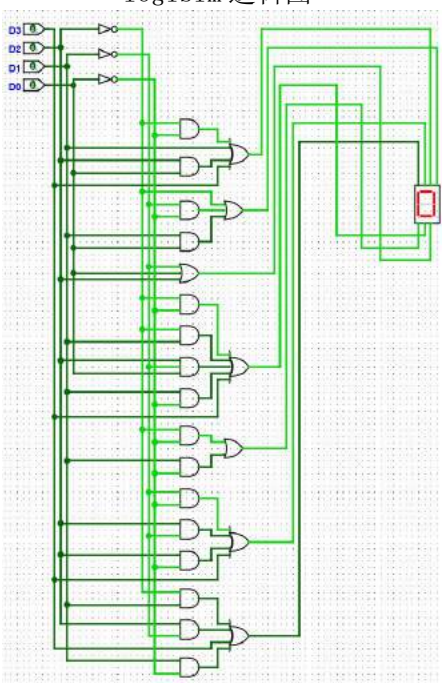


图一 七段数码管译码器对应控制部分



图二 数码管与引脚对应关系

logisim 逻辑图



真值表

输入				输出							显示
D_3	D_2	D_1	D_0	g	f	e	d	c	b	a	字符
0	0	0	0	1	0	0	0	0	0	0	0
0	0	0	1	1	1	1	1	0	0	1	1
0	0	1	0	0	1	0	0	1	0	0	2
0	0	1	1	0	1	1	0	0	0	0	3
0	1	0	0	0	0	1	1	0	0	1	4
0	1	0	1	0	0	1	0	0	1	0	5
0	1	1	0	0	0	0	0	0	1	0	6
0	1	1	1	1	1	1	1	0	0	0	7
1	0	0	0	0	0	0	0	0	0	0	8
1	0	0	1	0	0	1	0	0	0	0	9

$a = \overline{D2}\cdot\overline{D0}+D1+D2\cdot D0+D3$

$b = \overline{D2}+\overline{D1}\cdot\overline{D0}+D1\cdot D0$

$c = \overline{D1}+D0+D2$

$d = \overline{D2}\cdot\overline{D0}+\overline{D2}\cdot D1+D2\cdot\overline{D1}\cdot D0+D1\cdot\overline{D0}+D3$

$e = \overline{D2}\cdot\overline{D0}+D1\cdot\overline{D0}$

$f = \overline{D1}\cdot\overline{D0}+D2\cdot\overline{D1}+D2\cdot\overline{D0}+D3$

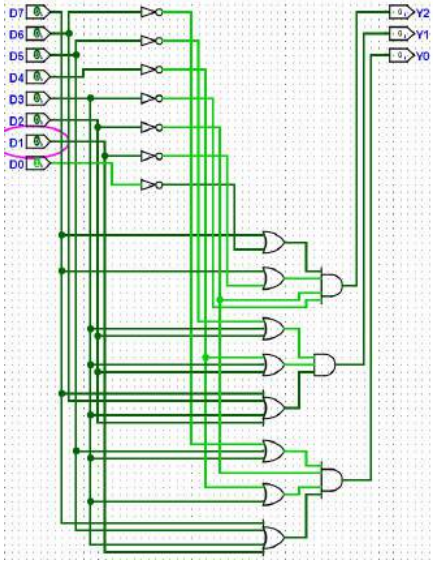
$g = \overline{D2}\cdot D1+D2\cdot\overline{D1}+D3+D1\cdot\overline{D0}$

真值表能看出，七段数码管译码器通过控制不同数码管发光，组合出相应字符。其本质是使用类 4-8 译码器。

3) 普通 8-3 编码器

用来完成编码工作的电路称为编码器。它可以实现对一组输入信号的二进制编码。下图为所要建模的普通 8-3 编码器及其真值表。它有 8 个输入以及 3 个输出，真值表中每行只有一个输入电平有效（高电平为 1，低电平为 0）。

logisim 逻辑图



真值表

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0	Y_2	Y_1	Y_0
1	0	0	0	0	0	0	0	1	1	1
0	1	0	0	0	0	0	0	1	1	0
0	0	1	0	0	0	0	0	1	0	1
0	0	0	1	0	0	0	0	1	0	0
0	0	0	0	1	0	0	0	0	1	1
0	0	0	0	0	1	0	0	0	1	0
0	0	0	0	0	0	1	0	0	0	1
0	0	0	0	0	0	0	1	0	0	0

逻辑表达式

$Y_2=I_4+I_5+I_6+I_7=\sim(\sim I_4\&\sim I_5\&\sim I_6\&\sim I_7)$

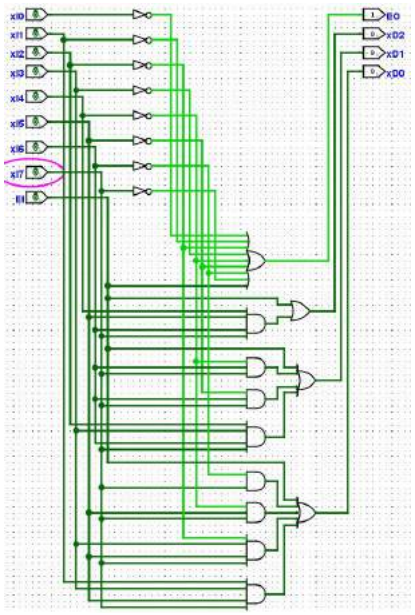
$Y_1=I_2+I_3+I_6+I_7=\sim(\sim I_2\&\sim I_3\&\sim I_6\&\sim I_7)$

$Y_0=I_1+I_3+I_5+I_7=\sim(\sim I_1\&\sim I_3\&\sim I_5\&\sim I_7)$

4) 具有优先级的 8-3 编码器

普通编码器对输入线是有限制的，即在任意一时刻所有输入线中只允许一个输入线信号有效，否则 编码器将发生混乱。为解决这一问题可以采用具有优先级的编码器。下图为所要建模的具有优先 级的 8-3 编码器及其真值表，它有八个输入端、三个输出端、一个选通输入端 EI 以及一个扩展输出端 EO。从真值表可以看出，输入输出的有效信号是低电平，在输入中，角标越大，优先级越高。

logisim 逻辑图



真值表

$\overline{D}_0$	$\overline{D}_1$	$\overline{D}_2$	$\overline{D}_3$	$\overline{D}_4$	$\overline{D}_5$	$\overline{D}_6$	$\overline{D}_7$	$\overline{Y}_2\overline{Y}_1\overline{Y}_0$			EI	EO
×	×	×	×	×	×	×	×	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	0	0
×	×	×	×	×	×	×	0	0	0	0	0	1
×	×	×	×	×	×	0	1	0	0	1	0	1
×	×	×	×	×	0	1	1	0	1	0	0	1
×	×	×	×	0	1	1	1	0	1	1	0	1
×	×	×	0	1	1	1	1	1	0	0	0	1
×	×	0	1	1	1	1	1	1	0	1	0	1
×	0	1	1	1	1	1	1	1	1	0	0	1
0	1	1	1	1	1	1	1	1	1	1	0	1

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）

（一）3-8 译码器实验 decoder38

接口定义

```
module decoder(
  input [2:0] iData, // 三位输入 D2, D1, D0
  input [1:0] iEna,  // 使能信号 G1, G2
  output [7:0] oData
  //八位译码输出 Y7 ~Y0, 低电平有效
);
```

- 功能描述

decoder 是一个 3 线-8 线译码器，这个模块采用了行为型描述，将三位输入映射到八位输出，根据输入的不同组合，选择哪个输出位激活。同时，含有两个使能位 G1 G2，可以根据他们的数值来切换译码器的工作状态。这种功能通常在数字电路中用于将特定输入模式映射到某些输出操作，例如在数字显示、存储器选择等应用中使用。

Verilog 代码描述

```
****接口定义 同上****
 reg [7:0] oData_reg;
always @(iData, iEna) begin
    if (~iEna[0] & iEna[1]) begin
        case (iData)
            3'b000: oData_reg = 8'b11111110;
            3'b001: oData_reg = 8'b11111101;
            3'b010: oData_reg = 8'b11111011;
            3'b011: oData_reg = 8'b11110111;
            3'b100: oData_reg = 8'b11101111;
            3'b101: oData_reg = 8'b11011111;
            3'b110: oData_reg = 8'b10111111;
            3'b111: oData_reg = 8'b01111111;
            default: oData_reg = 8'b11111111;
        endcase
    end else begin
        oData_reg = 8'b11111111;
    end
end
assign oData = oData_reg;
endmodule
```

XDC 约束

变量	iData[0]~[2]	iEna[0]~[1]	oData[0]~[7]
N4 板上的管脚	SW0~2 (J15、L16、M13)	SW14~15 (V10、U11)	LD0~7 (H17、K15、J13、N14、R18、V17、U17、U16)

（二）七段数码管译码驱动器 display7

接口定义

```
module display7(
   input  [3:0] iData;//四位输入 D3~D0
   output [6:0] oData;//七位译码输出 g~a
);
```

Verilog 代码描述

```
module display7(iData, oData);
    input [3:0] iData;
    output [6:0] oData;
    assign oData = (iData == 4'b0000) ? 7'b1000000 :
                   (iData == 4'b0001) ? 7'b1111001 :
                   (iData == 4'b0010) ? 7'b0100100 :
                   (iData == 4'b0011) ? 7'b0110000 :
                   (iData == 4'b0100) ? 7'b0011001 :
                   (iData == 4'b0101) ? 7'b0010010 :
                   (iData == 4'b0110) ? 7'b0000010 :
                   (iData == 4'b0111) ? 7'b1111000 :
                   (iData == 4'b1000) ? 7'b0000000 :
                   (iData == 4'b1001) ? 7'b0010000 :
                   7'b1111111;
endmodule
```

XDC 约束

变量	iData[0]~[3]	oData[0]~[6]
N4 板上的管脚	SW0~3(J15、L16、M13、R15)	CA(T10)、CB(R10)、CC(K16)、CD(K13)、CE(P15)、CF(T11)、CG(L18)

- 功能描述

display7 模块实现了一个数字显示器，其功能是将一个 4 位二进制输入 iData 映射到一个 7 段 LED 显示器的输出 oData，通过数码管的组合，显示不同的数字。使用了行为型描述方式，其原理类似于译码器原理，但是每一个输入值都对应一个 7 位的输出。

（三）普通 8-3 编码器　encoder83

接口定义

```
module encoder83(
    input [7:0] iData, //八位输入 D7~D0,高电平有效
    output [2:0] oData //三位编码输出 Y2~Y0
);
```

Verilog 代码描述

```
`timescale 1ns / 1ps
module encoder83(
    input [7:0] iData, // 八位输入 D7~D0，高电平有效
    output reg [2:0] oData // 三位编码输出 Y2~Y0
);
always @(*) begin
    case(iData)
        8'b00000001: oData = 3'b000; // Input D0
        8'b00000010: oData = 3'b001; // Input D1
        8'b00000100: oData = 3'b010; // Input D2
        8'b00001000: oData = 3'b011; // Input D3
        8'b00010000: oData = 3'b100; // Input D4
        8'b00100000: oData = 3'b101; // Input D5
        8'b01000000: oData = 3'b110; // Input D6
        8'b10000000: oData = 3'b111; // Input D7
        default: oData = 3'b000; // Default case
    endcase
end
endmodule
```

XDC 约束

变量	iData[0]~[7]	oData[0]~[2]
N4 板上的管脚	SW0~7 (J15、L16、M13、R15、R17、T18、U18、R13)	LD0~2 (H17、K15、J13)

- 功能描述

encoder83 模块实现了 8 线-3 线编码器，将 8 位输入编码成 3 位输出，使用了行为型描述。这种编码器属于普通的 8 线 3 线编码器，不具有优先级。

（四）具有优先级的 8-3 编码器　encoder83_Pri

接口定义

```
module encoder83_Pri(
    input [7:0] iData, //八位输入¯7~¯0,低电平有效
    input iEI, //选通输入信号 EI,低电平有效
    output [2:0] oData, //三位编码输出 3~ 0¯
    output oEO //扩展输出信号 EO,高电平有效
);
```

Verilog 代码

```
module encoder83_Pri (
    input [7:0] iData,
    input iEI,
    output [2:0] oData,
    output oEO
);
reg [7:0] invertedData;  // 存储 iData 的非
reg [2:0] priorityEncodedData;
reg internal_EO; // 模块内部使用的 EO
integer con; // 使用 integer 类型定义 con
always @(*) begin
    // 默认值
    priorityEncodedData = 3'b111;
    internal_EO = 1'b1;
    con = 0;
```

描述

```
下续
invertedData = ~iData;
    if  (iEI == 1'b1)begin
        priorityEncodedData = 3'b000;
    end
    else if (iEI == 1'b0) begin
        if (invertedData[7]==1'b1 && con == 0) begin
            priorityEncodedData = 3'b111;
            con = 1;
        end else if (invertedData[6]==1'b1 && con == 0) begin
            priorityEncodedData = 3'b110;
            con = 1;
        end else if (invertedData[5]==1'b1 && con == 0) begin
            priorityEncodedData = 3'b101;
            con = 1;
        end else if (invertedData[4]==1'b1 && con == 0) begin
            priorityEncodedData = 3'b100;
            con = 1;
        end else if (invertedData[3]==1'b1 && con == 0) begin
            priorityEncodedData = 3'b011;
            con = 1;
        end else if (invertedData[2]==1'b1 && con == 0) begin
            priorityEncodedData = 3'b010;
            con = 1;
        end else if (invertedData[1]==1'b1 && con == 0) begin
            priorityEncodedData = 3'b001;
            con = 1;
        end else if (invertedData[0]==1'b1 && con == 0) begin
            priorityEncodedData = 3'b000;
            con = 1;
        end else begin
         if (con == 0)begin
             priorityEncodedData = 3'b111;
          internal_EO = 1'b0;
          con = 1;
         end
        end
    end
end
assign oData = ~priorityEncodedData;
assign oEO = internal_EO;
endmodule
```

XDC 约束

变量	iData[0]~[7]	oData[0]~[2]	iEI	oEO
N4 板上的管脚	SW0~7 (J15、L16、M13、R15、R17、T18、U18、R13)	LD0~2 (H17、K15、J13)	SW15 (V10)	LD15 (V11)

- 功能描述

Encder83_Pri 实现了一个具有优先级的 83 编码器，可以将 8 位输入编码为 3 位输出，同时，含有使能端口 EI，低电平有效，在低电平时激活编码器，在高电平时使编码器无效，EO 用于指示编码是否完成。这个模块使用了行为型描述，可以应用于同时处理多个输入信号，根据这些信号的优先级来确定要同时处理的信号。

四、 测试模块建模

(一) 3-8 译码器实验 decoder

```
`timescale 1ns / 1ps
module decoder_tb;
    reg [2:0] iData;
    reg [1:0] iEna;
    wire [7:0] oData;
    decoder dut(
        .iData(iData),
        .iEna(iEna),
        .oData(oData)
    );
    initial begin
        iEna = 2'b11;
        iData = 3'b101;
        #10
        iEna = 2'b00;
        iData = 3'b111;
        #10
        iEna = 2'b10;
        iData = 3'b000;
        #10
        iEna = 2'b10;
        iData = 3'b001;
        #10
iEna = 2'b10;
        iData = 3'b010;
        #10
        iEna = 2'b10;
        iData = 3'b011;
        #10
        iEna = 2'b10;
        iData = 3'b100;
        #10
        iEna = 2'b10;
        iData = 3'b101;
        #10
        iEna = 2'b10;
        iData = 3'b110;
        #10
        iEna = 2'b10;
        iData = 3'b111;
        #10
        $finish;
    end
endmodule
```

(二) 七段数码管译码驱动器实验 display7

```
`timescale 1ns / 1ps
module display7_tb;
    reg [3:0] iData;
    wire [6:0] oData;

    display7 uut (
        .iData(iData),
        .oData(oData)
    );

    initial begin
        iData = 4'b0000;

        #10 iData = 4'b0000;
        #10 iData = 4'b0001;
        #10 iData = 4'b0010;
        #10 iData = 4'b0011;
#10 iData = 4'b0100;
        #10 iData = 4'b0101;
        #10 iData = 4'b0110;
        #10 iData = 4'b0111;
        #10 iData = 4'b1000;
        #10 iData = 4'b1001;
        $finish;
    end
    always @(oData) begin
        $display("iData = %b, oData = %b", iData, oData);
    end
endmodule
```

(三) 普通 8-3 编码器实验 encoder83

```
`timescale 1ns / 1ps
module encoder83_tb;
    reg [7:0] iData;
    wire [2:0] oData;

    encoder83 uut (
        .iData(iData),
        .oData(oData)
    );

    initial begin
        iData = 8'b00000001;
        #10 iData = 8'b00000010;
        #10 iData = 8'b00000100;
        #10 iData = 8'b00001000;
        #10 iData = 8'b00010000;
        #10 iData = 8'b00100000;
        #10 iData = 8'b01000000;
        #10 iData = 8'b10000000;
        #10
        $finish;
    end
    always @(oData) begin
        $display("iData = %b, oData = %b", iData, oData);
    end
endmodule
```

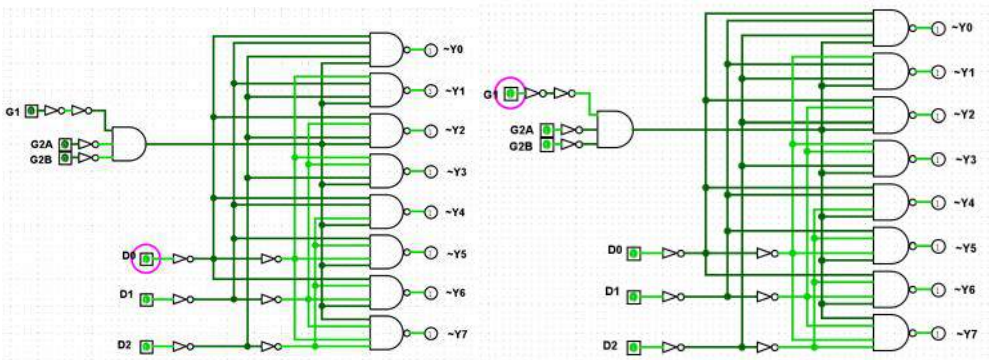
(四) 具有优先级的 8-3 编码器实验 encoder83_Pri

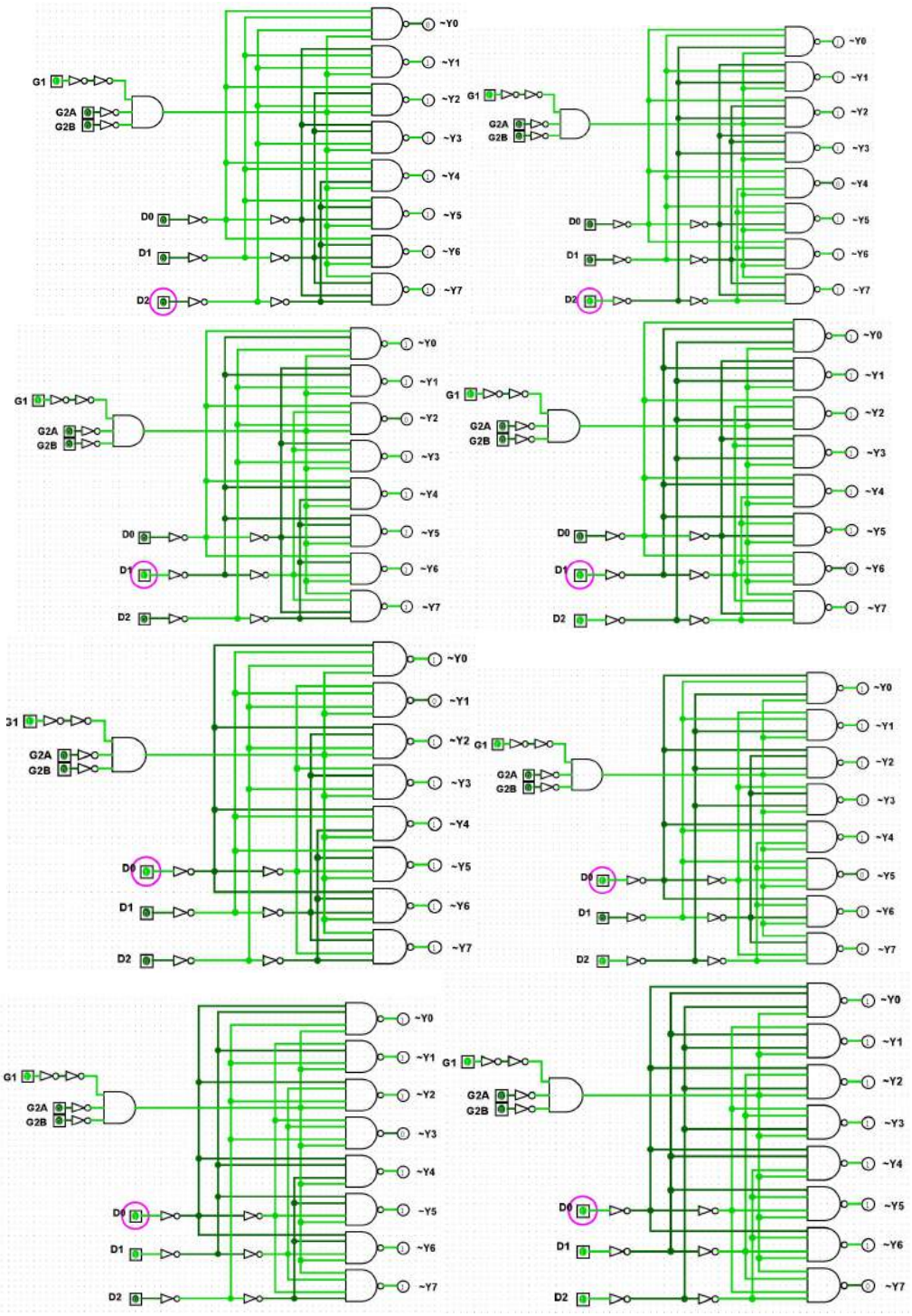
```
`timescale 1ns / 1ps
module encoder83_Pri_tb;
  reg [7:0] iData;
  reg iEI;
  wire [2:0] oData;
  wire oEO;
  encoder83_Pri my_encoder (
    .iData(iData),
    .iEI(iEI),
    .oData(oData),
    .oEO(oEO)
  );
  reg clk = 0;
  always begin
    #5 clk = ~clk;
  end
  initial begin
    iData = 8'b00000000;
    iEI = 1'b0;
    $display("iData = %h, iEI = %b",
iData, iEI);
    $monitor("oData = %b, oEO = %b",
oData, oEO);
    iData = 8'b10101010;
    iEI = 1'b1;
    #10;
    iData = 8'b00000000;
    iEI = 1'b0;
    #10;
    iData = 8'b11010101;
    iEI = 1'b0;
    #10;
    iData = 8'b01101010;
    iEI = 1'b0;
    #10;
    iData = 8'b00101010;
    iEI = 1'b0;
    #10;
    iData = 8'b0001110;
    iEI = 1'b0;
    #10;
    iData = 8'b0000111;
    iEI = 1'b0;
    #10;
    iData = 8'b00000100;
    iEI = 1'b0;
    #10;
    iData = 8'b11101100;
    iEI = 1'b0;
    #10;
    iData = 8'b10100010;
    iEI = 1'b0;
    #10;
    iData = 8'b00000001;
    iEI = 1'b1;
    #10;
    iData = 8'b00000000;
    iEI = 1'b1;
    #10;
    iData = 8'b00000000;
    iEI = 1'b0;
    #10;
    $finish;
  end
endmodule
```

五、 实验结果

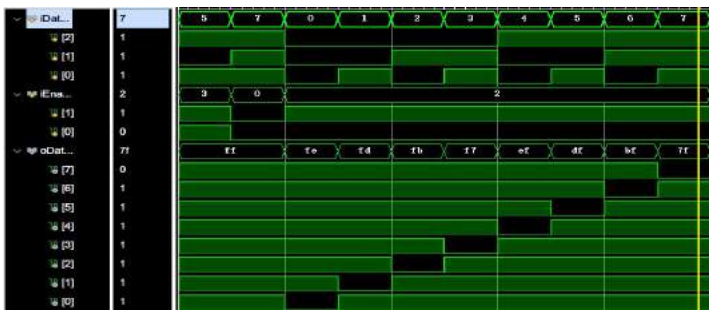
(一) 3-8 译码器实验

- logisim 验证图

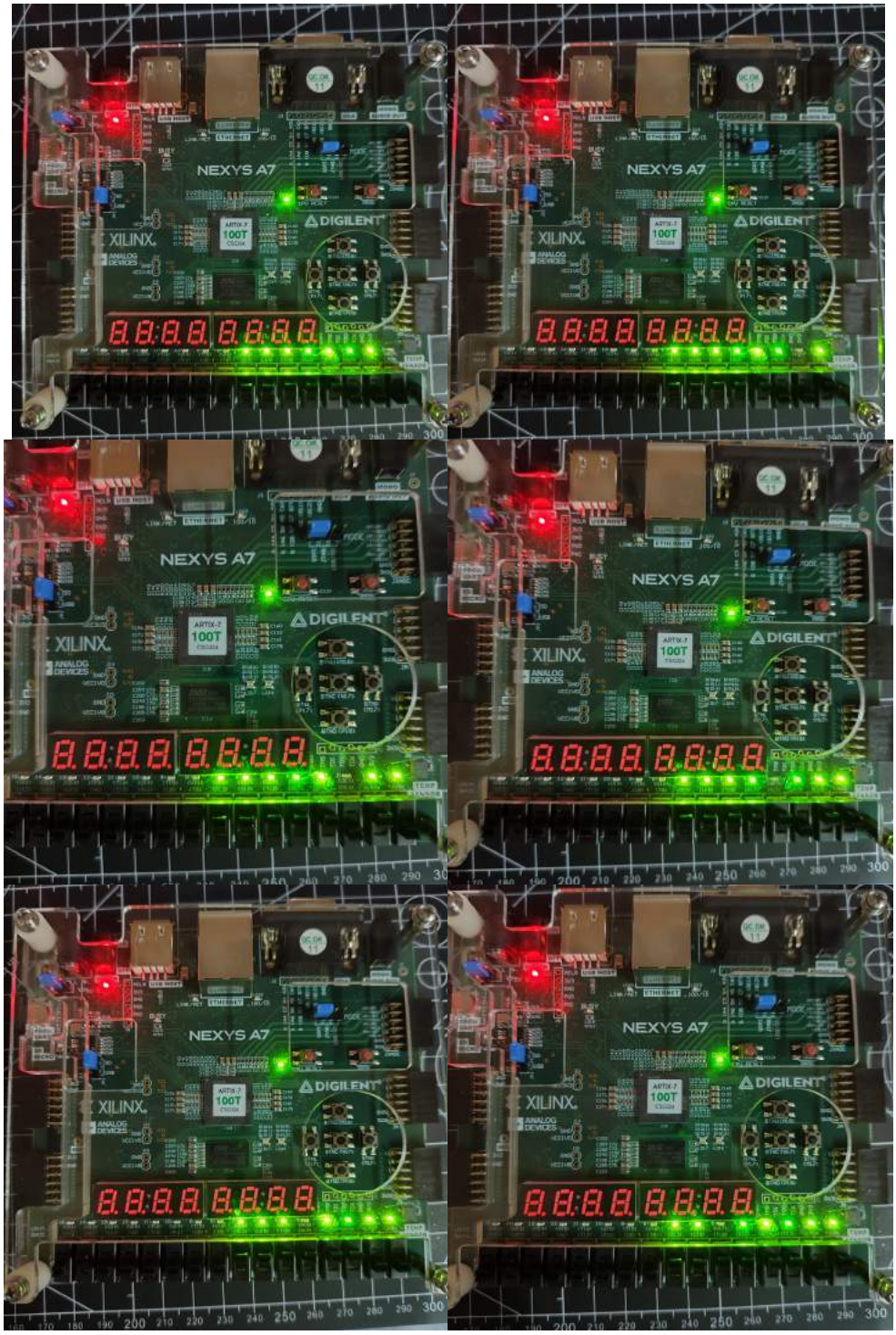


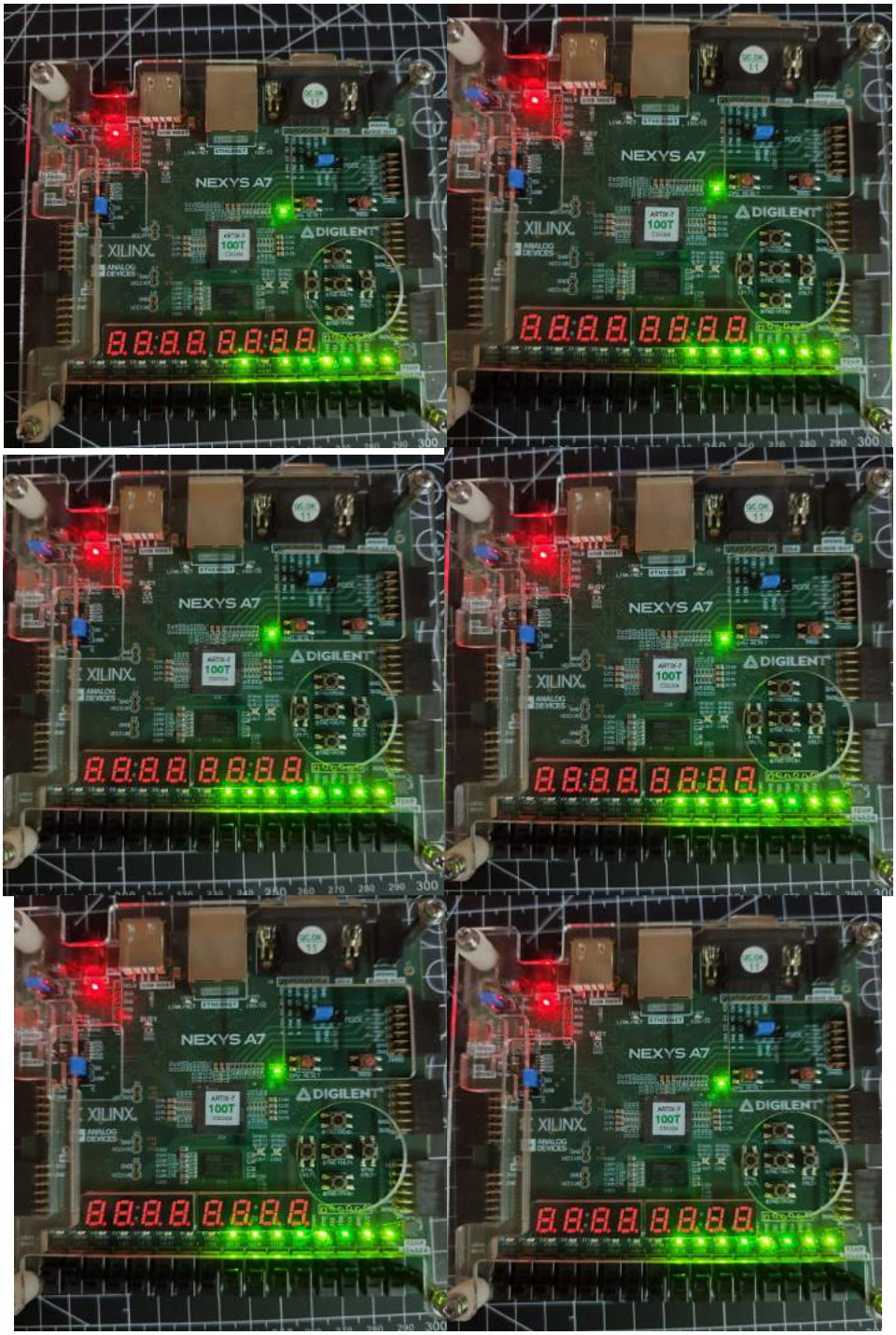


• modelsim 仿真



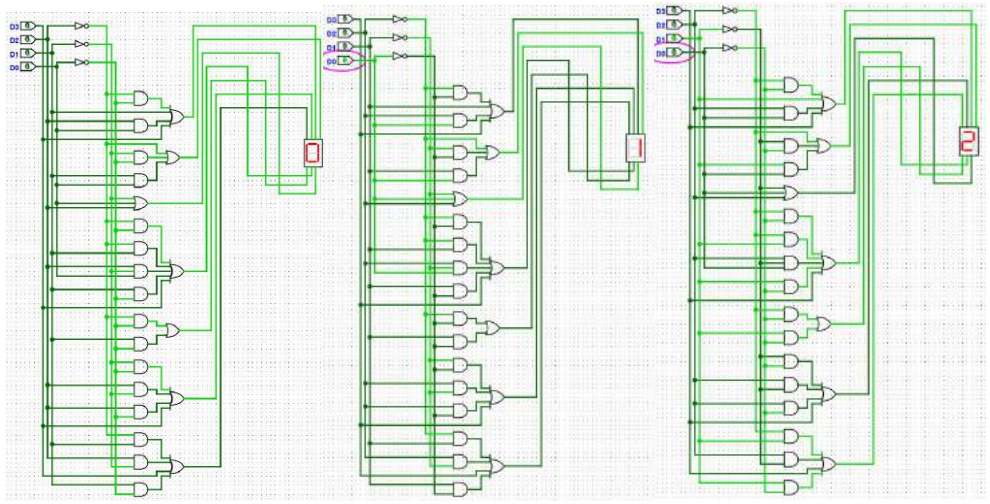
- 实验结果

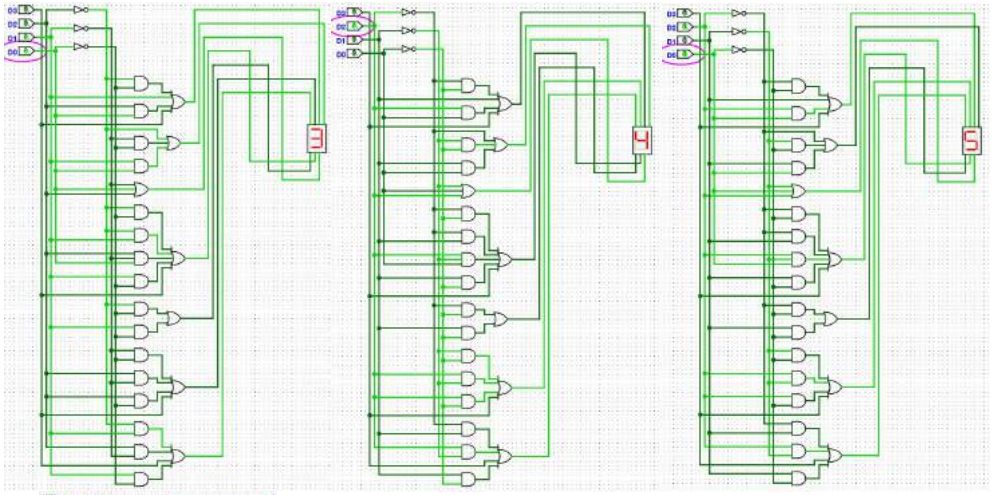


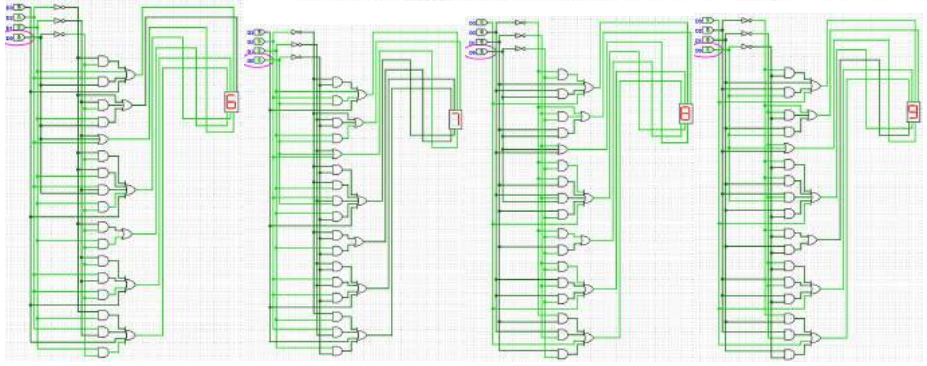


(二) 七段数码管译码驱动器实验

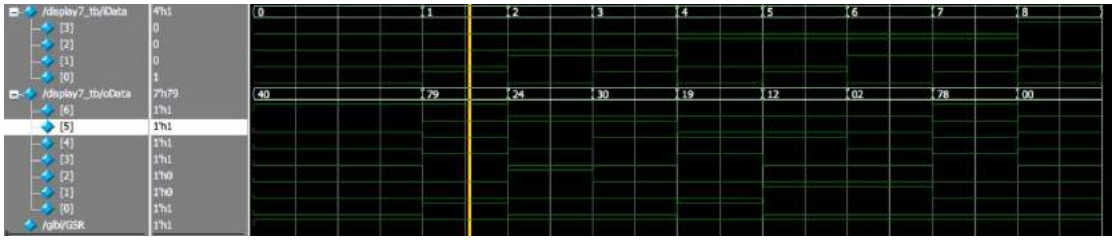
- logisim 原理图



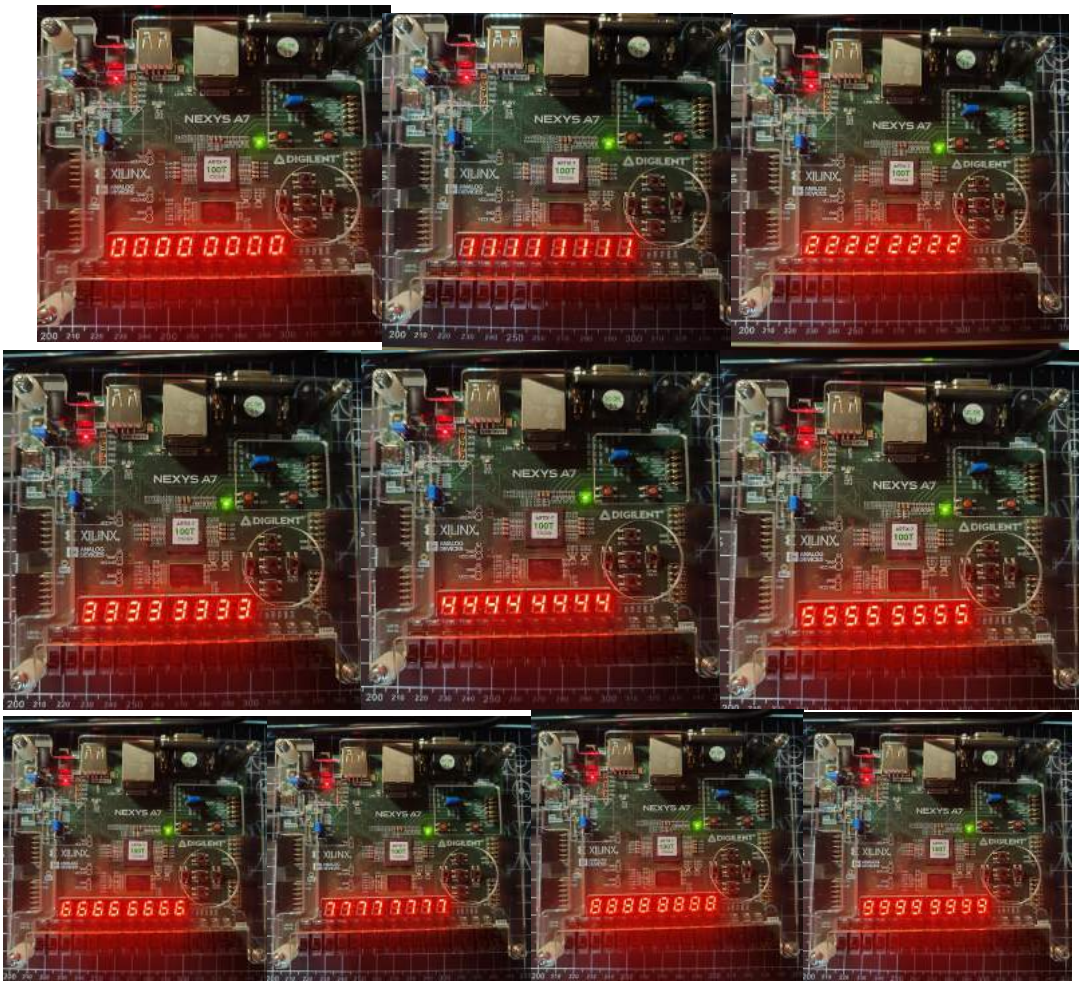




- modelsim 仿真

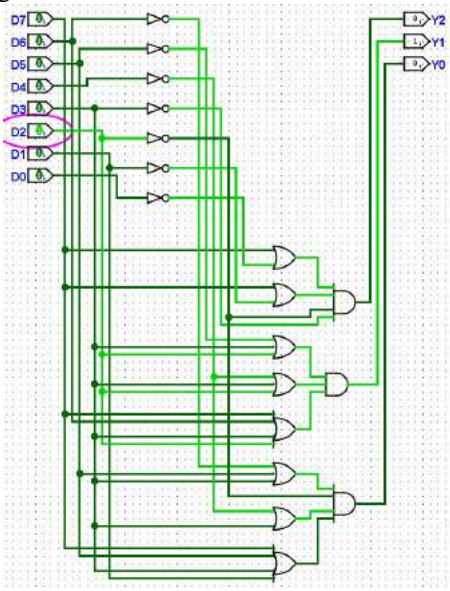


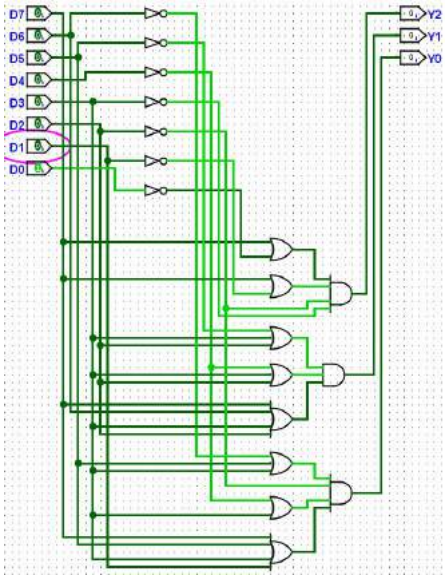
- 实验结果

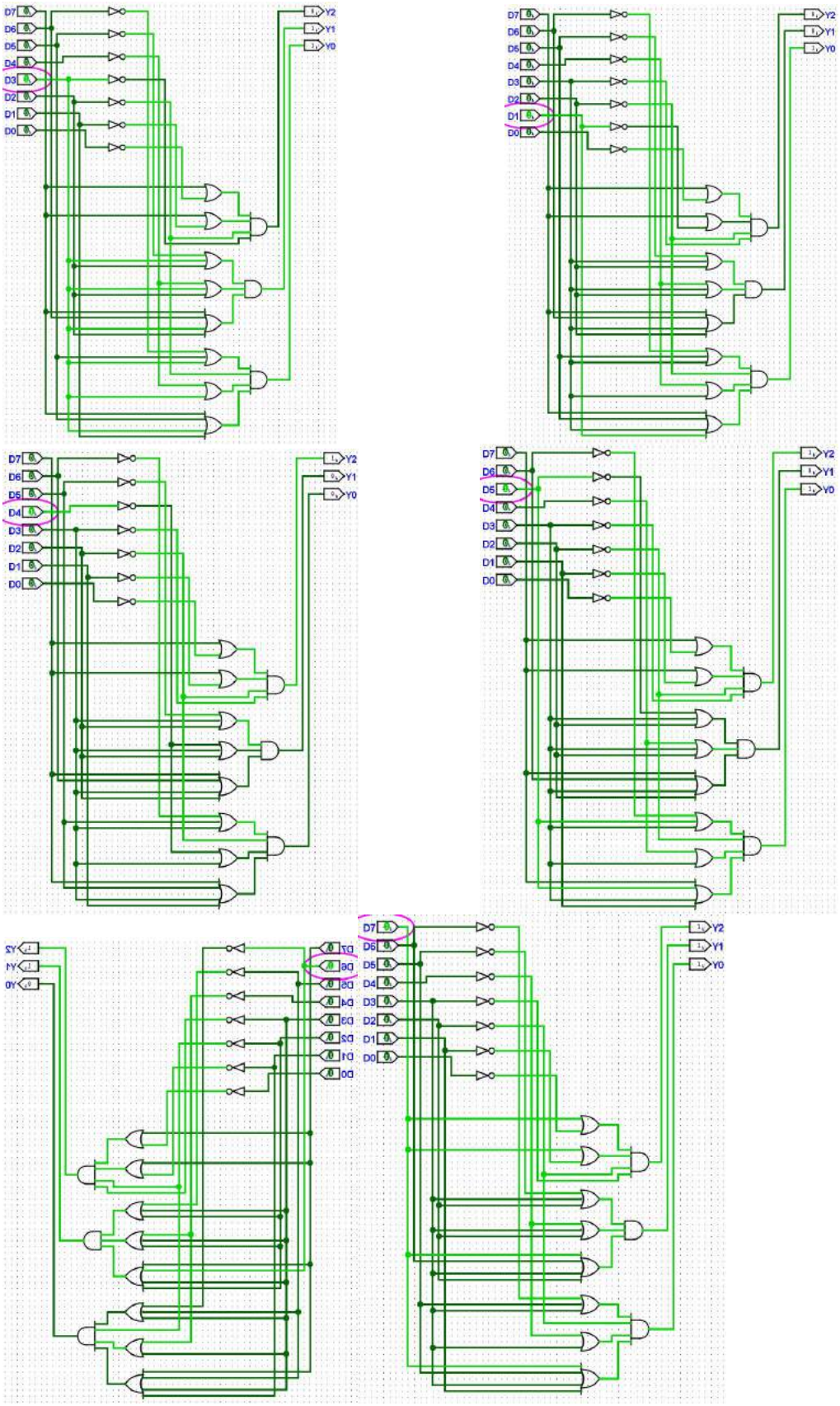


(三) 普通 83 编码器

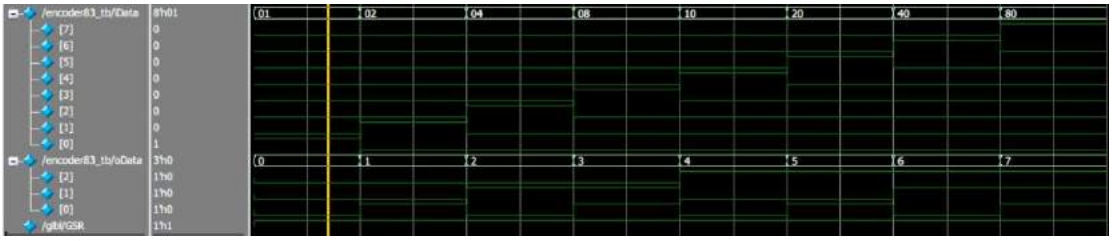
- logisim 原理图



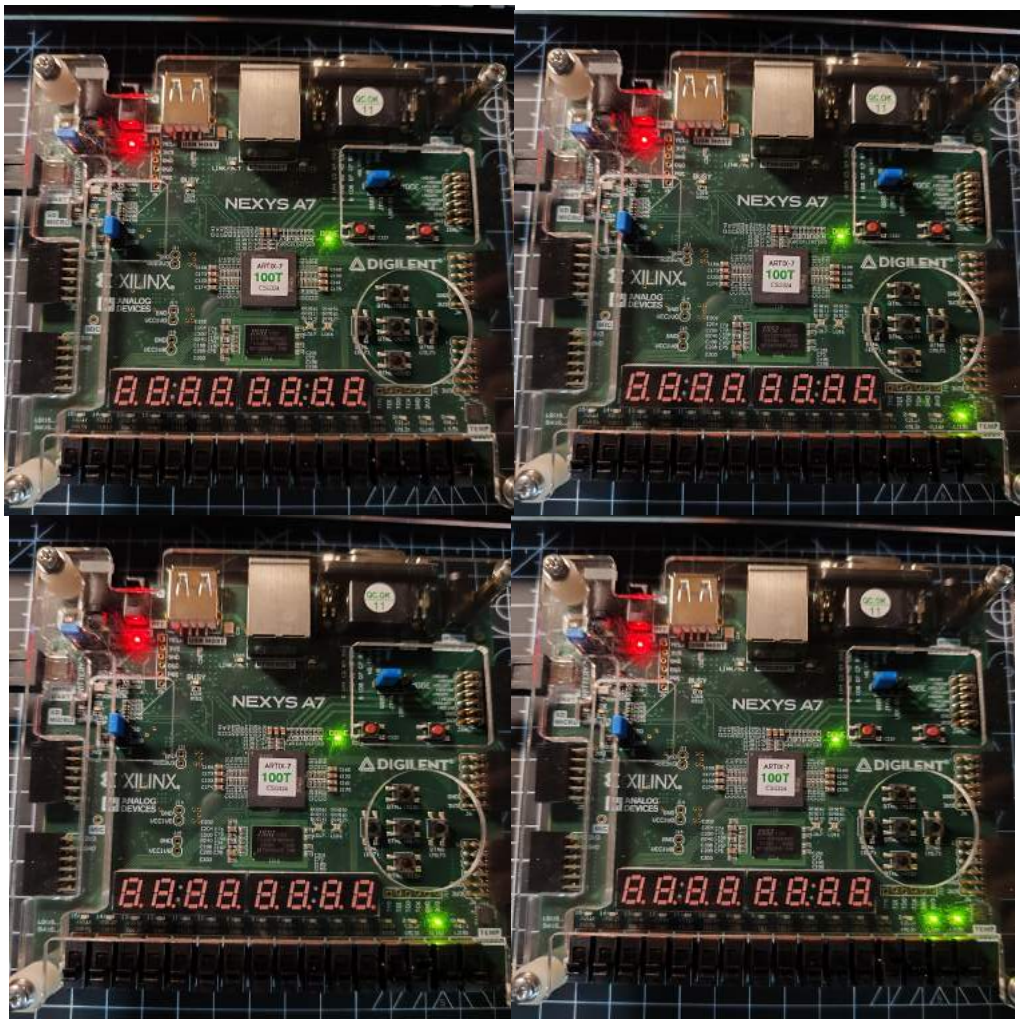


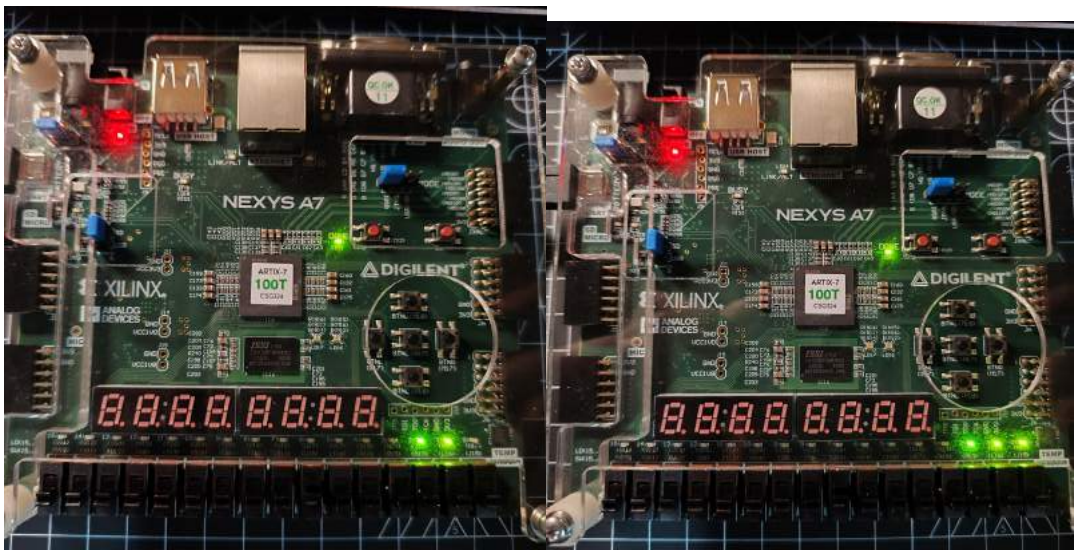


• modelsim 仿真波形图


/encoder83_tb/iData
[7]
[6]
[5]
[4]
[3]
[2]
[1]
[0]
/glbl/GSR
/encoder83_tb/oData
[2]
[1]
[0]

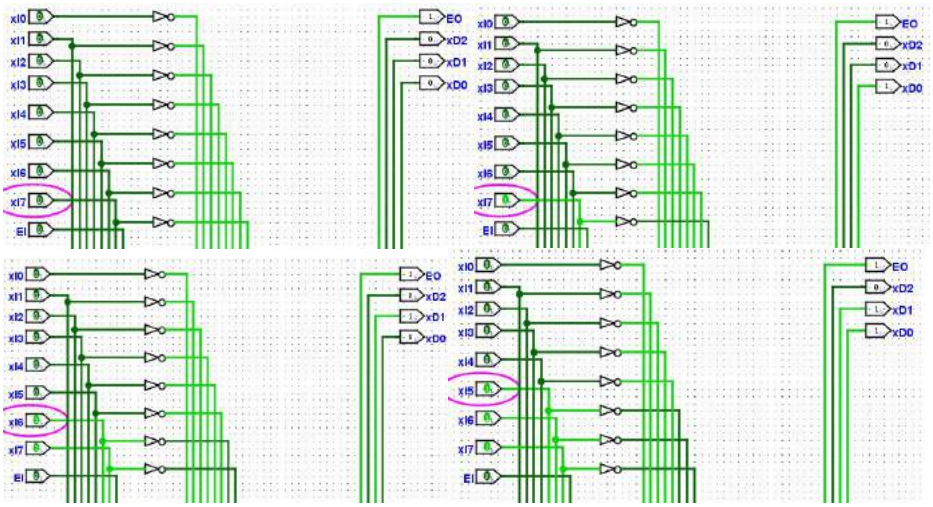
- 实验结果

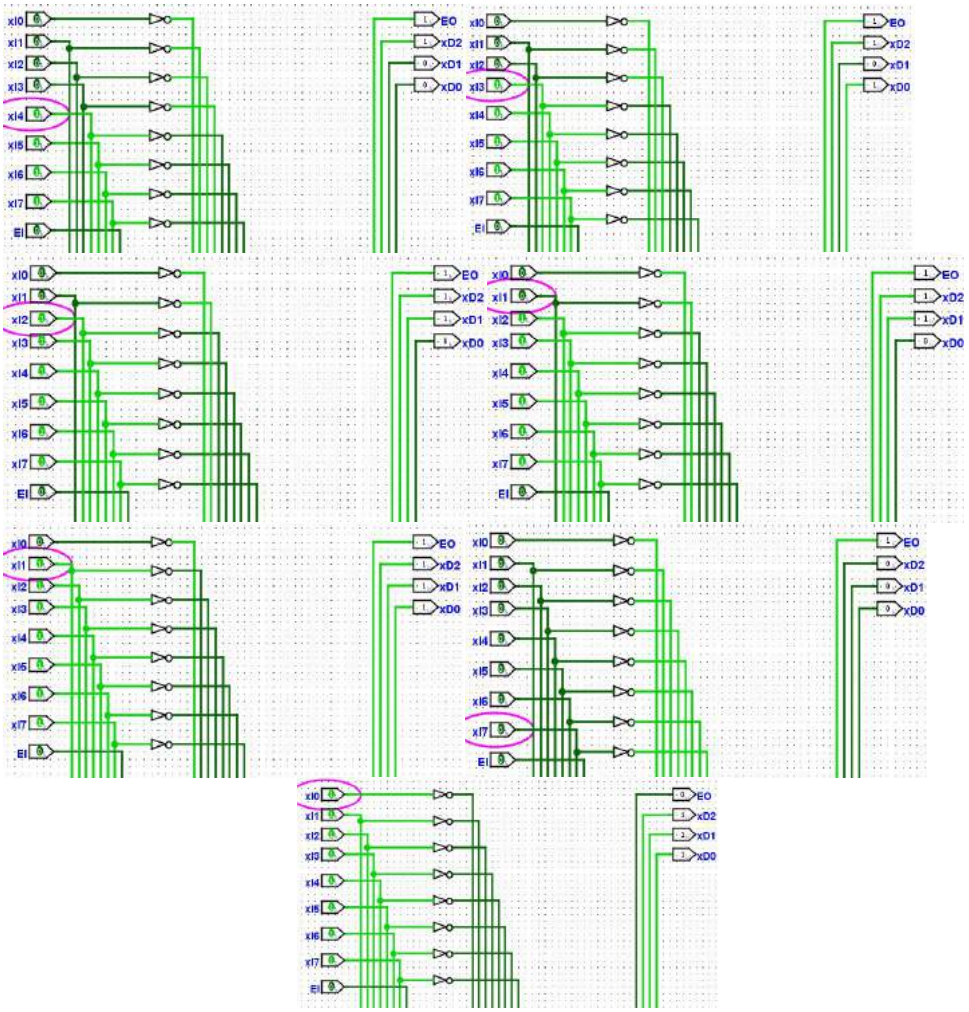

NEXYS A7
XILINX
DIGILENT
ANALOG DEVICES
100T



(四) 具有优先级的 8-3 编码器实验

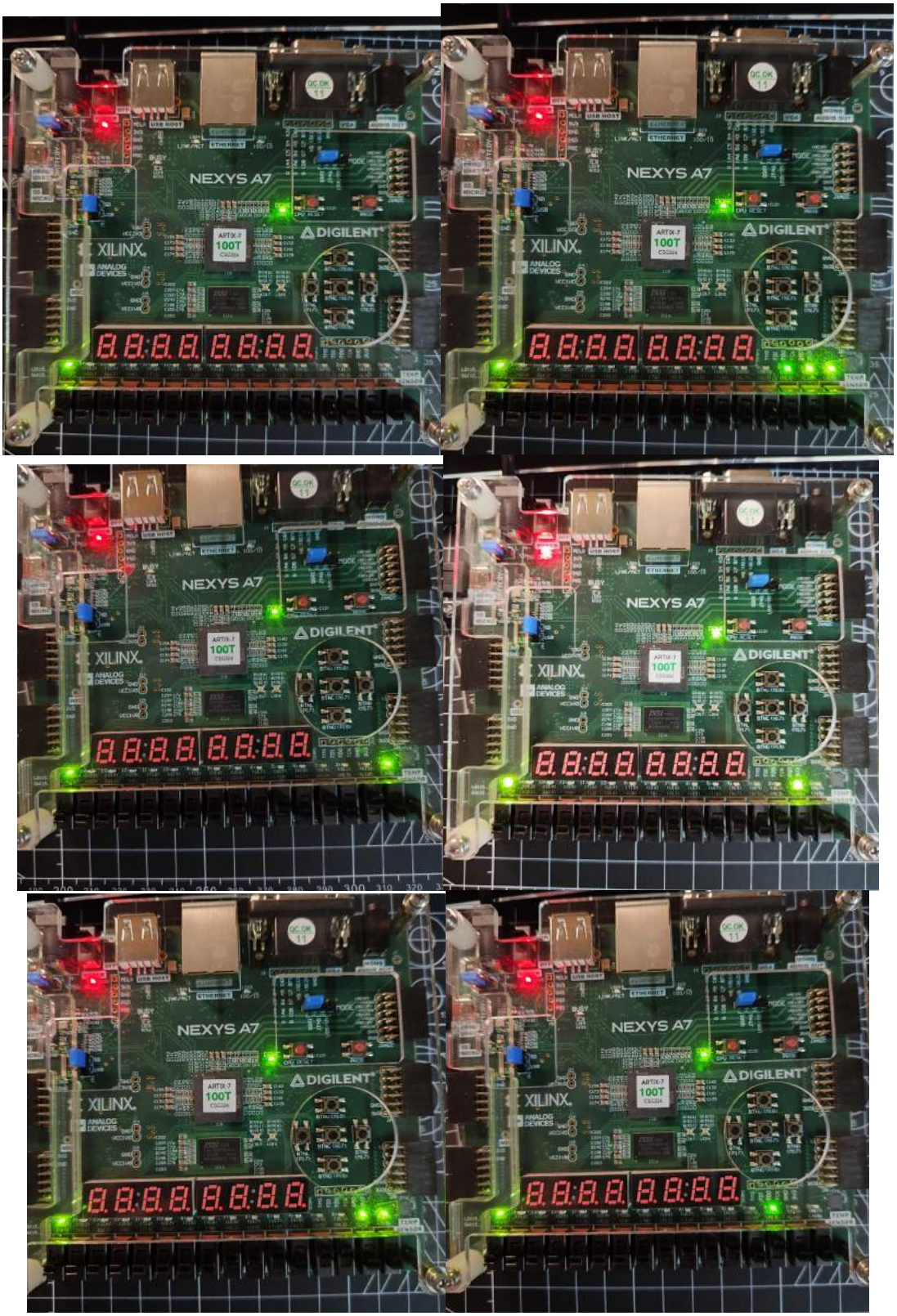
- logisim 原理图

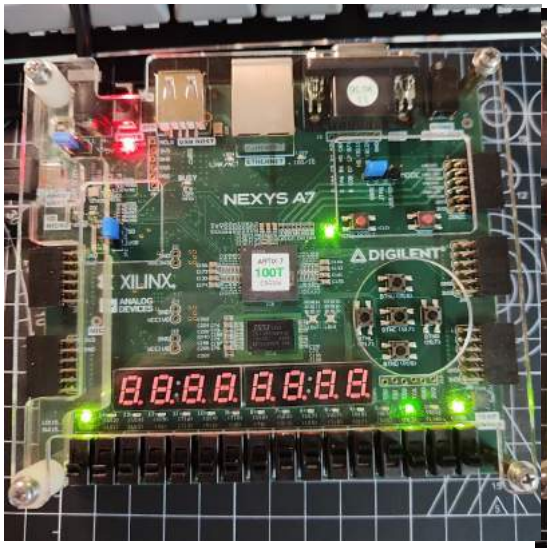


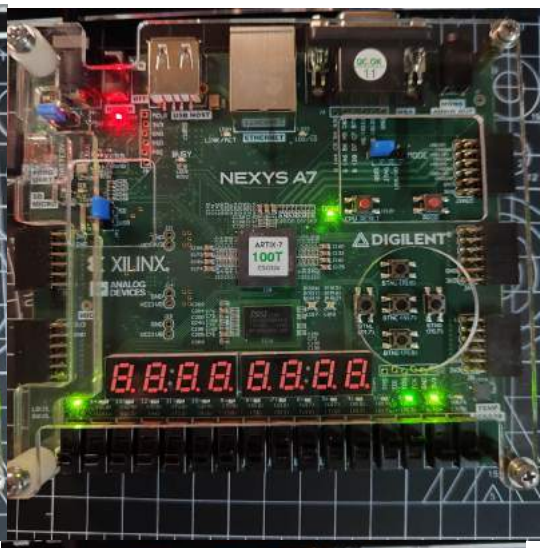


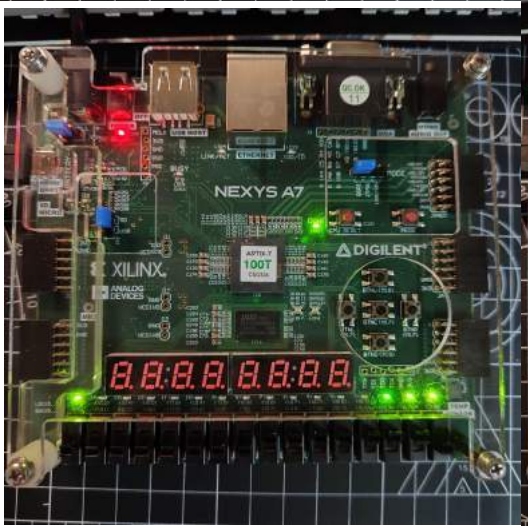
- modelsim 仿真

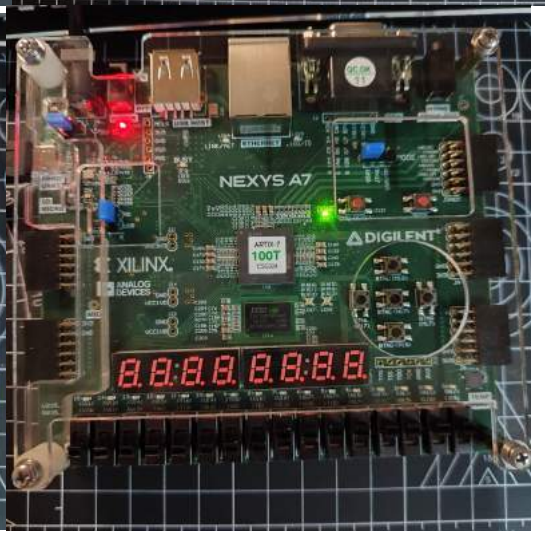
- 实验结果


NEXYS A7
DIGILENT
XILINX
ANALOG DEVICES
ARTIX-7
100T


NEXYS A7
DIGILENT
XILINX
ANALOG DEVICES
ARTIX-7
100T


NEXYS A7
DIGILENT
XILINX
ANALOG DEVICES
ARTIX-7
100T


NEXYS A7
DIGILENT
XILINX
ANALOG DEVICES
ARTIX-7
100T


NEXYS A7
DIGILENT
XILINX
ANALOG DEVICES
ARTIX-7
100T