实验六：EDA软件使用与组合逻辑设计

专业班级：**通信2101班**

姓名： **罗畅**

学号： U202113940

实验名称

EDA软件使用与组合逻辑设计

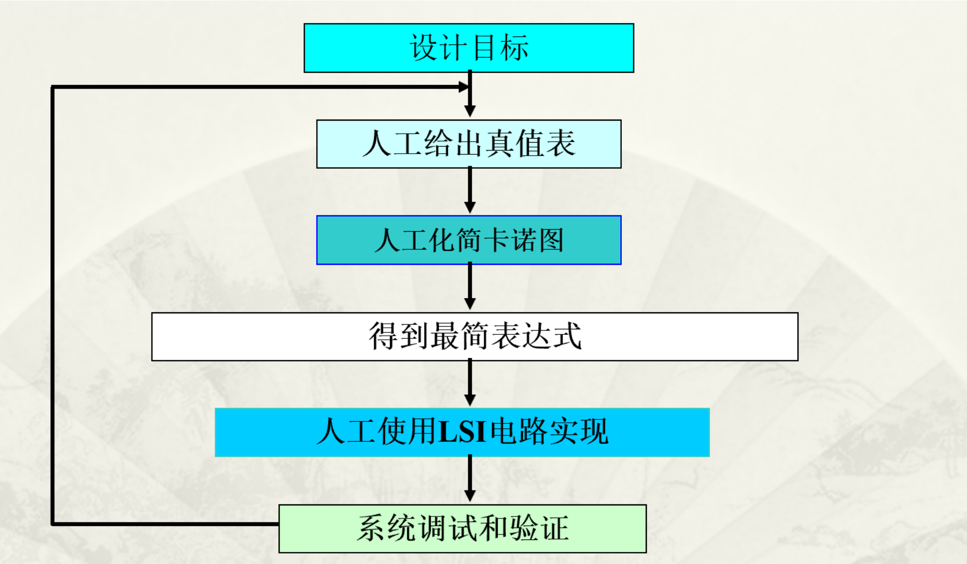
实验目的

1. 了解并掌握采用可编程逻辑器件实现数字电路与系统的方法
2. 学习并掌握采用Xilinx_ISE 软件开发可编程器件的过程
3. 学习使用verilog HDL描述数字逻辑电路与系统的方法
4. 掌握分层次、分模块的电路设计方法，熟悉使用可编程器件实现数字系统的一般步骤。

实验元器件

电脑软件*Vivado 2018.03*（*win11*没法装*ISE*呜呜呜）

实验原理



实验任务

按号分别实现下列项目之一（**三种编程方式形式实现**）

- 一位大小比较器（三灯指示，大于、等于、小于）
- 1位2选1数据选择器---p226实验任务二
- 求反加1
- 3线-8线译码器
- 8线-3线优先编码器

实验记录

结构型

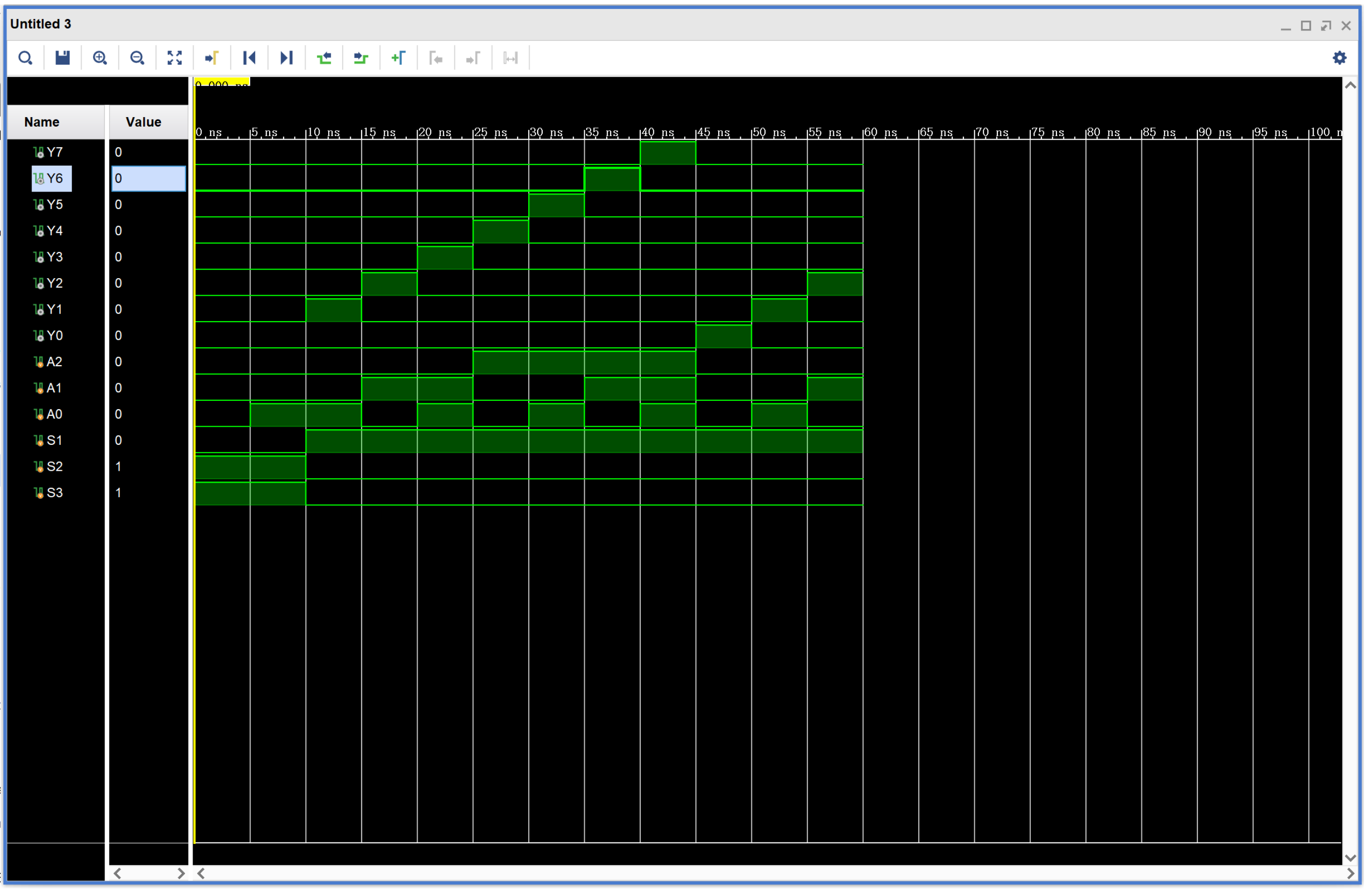
three_eight_decoder_structure.v

```verilog
`timescale 1ns / 1ps
//////////////////////////////////////////////////////////////////////////////////
// Company: HUST
// Engineer: Luo Chang
//
// Create Date: 2023/03/05 17:17:00
// Design Name:
// Module Name: three_eight_decoder_structure
// Project Name:
// Target Devices:
// Tool Versions:
// Description:
//
// Dependencies:
//
// Revision:
// Revision 0.01 - File Created
// Additional Comments:
//
//////////////////////////////////////////////////////////////////////////////////


module three_eight_decoder_structure(
    input A2,A1,A0,
    input S1,
    input S2,
    input S3,
    output Y7, Y6, Y5, Y4, Y3, Y2, Y1, Y0
    );

    wire En;

    and (En, S1, ~S2, ~S3);

    and (Y0, En, ~A2, ~A1, ~A0);
    and (Y1, En, ~A2, ~A1, A0);
    and (Y2, En, ~A2, A1, ~A0);
    and (Y3, En, ~A2, A1, A0);

    and (Y4, En, A2, ~A1, ~A0);
    and (Y5, En, A2, ~A1, A0);
    and (Y6, En, A2, A1, ~A0);
    and (Y7, En, A2, A1, A0);

endmodule
```

testbench.v

```verilog
`timescale 1ns / 1ps
//////////////////////////////////////////////////////////////////////////////////
// Company: HUST
// Engineer: Luo Chang
//
// Create Date: 2023/03/05 18:13:01
// Design Name:
// Module Name: testbench
// Project Name:
// Target Devices:
// Tool Versions:
// Description:
//
// Dependencies:
//
// Revision:
// Revision 0.01 - File Created
// Additional Comments:
//
//////////////////////////////////////////////////////////////////////////////////


module testbench;
    wire Y7,Y6,Y5,Y4,Y3,Y2,Y1,Y0;

    reg A2,A1,A0;
    reg S1, S2, S3;

    three_eight_decoder_structure U0(A2, A1, A0, S1, S2, S3,
    Y7, Y6, Y5, Y4, Y3, Y2, Y1, Y0);
    initial $monitor($time,"A = %b,S1 = %b,S2 = %b,S3 = %b,Y = %b", {A2,A1,A0}, S1, S2, S3, {Y7,Y6,Y5,Y4,Y3,Y2,Y1,Y0});

    always #5 {A2,A1,A0} = {A2,A1,A0}+1'b1;
    initial
        begin
            S1 = 0;
            S2 = 1;
            S3 = 1;
            {A2,A1,A0}=3'b000;
            #10
            S1 = 1;
            S2 = 0;
            S3 = 0;
            {A2,A1,A0}=3'b000;
            #50
            $stop;
        end

endmodule
```

wave.v



Name	Value
Y7	0
Y6	0
Y5	0
Y4	0
Y3	0
Y2	0
Y1	0
Y0	0
A2	0
A1	0
A0	0
S1	0
S2	1
S3	1

数据流型

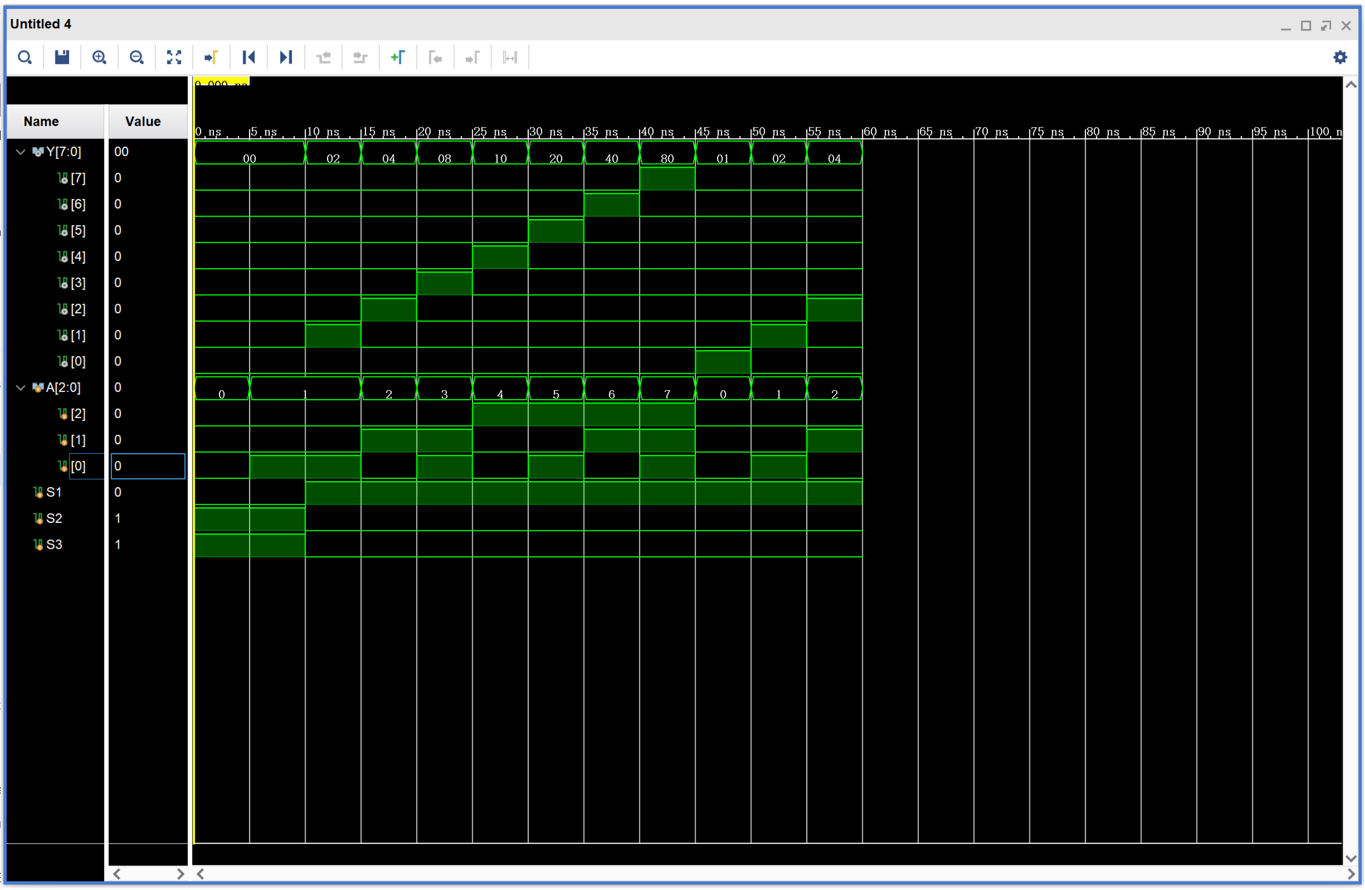
three_eight_decoder_data_flow.v

```
`timescale 1ns / 1ps
//////////////////////////////////////////////////////////////////////////////////
// Company: HUST
// Engineer: Luo Chang
//
// Create Date: 2023/03/06 10:08:00
// Design Name:
// Module Name: three_eight_decoder_data_flow
// Project Name:
// Target Devices:
// Tool Versions:
// Description:
//
// Dependencies:
//
// Revision:
// Revision 0.01 - File Created
// Additional Comments:
//
//////////////////////////////////////////////////////////////////////////////////


module three_eight_decoder_data_flow(
    input [2:0] A,
    input S1,
    input S2,
    input S3,
    output [7:0] Y
    );
    wire En;

    assign En = S1 && (~S2) && (~S3);

    assign Y[0] = En && ~A[2] && ~A[1] && ~A[0];
    assign Y[1] = En && ~A[2] && ~A[1] &&  A[0];
    assign Y[2] = En && ~A[2] &&  A[1] && ~A[0];
    assign Y[3] = En && ~A[2] &&  A[1] &&  A[0];

    assign Y[4] = En &&  A[2] && ~A[1] && ~A[0];
    assign Y[5] = En &&  A[2] && ~A[1] &&  A[0];
    assign Y[6] = En &&  A[2] &&  A[1] && ~A[0];
    assign Y[7] = En &&  A[2] &&  A[1] &&  A[0];

endmodule
```

testbench.v

```verilog
`timescale 1ns / 1ps
//////////////////////////////////////////////////////////////////////////////////
// Company: HUST
// Engineer: Luo chang
//
// Create Date: 2023/03/06 10:12:34
// Design Name:
// Module Name: testbench
// Project Name:
// Target Devices:
// Tool Versions:
// Description:
//
// Dependencies:
//
// Revision:
// Revision 0.01 - File Created
// Additional Comments:
//
//////////////////////////////////////////////////////////////////////////////////


module testbench;

    wire [7:0] Y;

    reg [2:0] A;
    reg S1, S2, S3;

    three_eight_decoder_data_flow U0(A, S1, S2, S3, Y);
    initial $monitor($time,"A = %b,S1 = %b,S2 = %b,S3 = %b,Y = %b", A, S1, S2, S3, Y);

    always #5 A = A + 1'b1;
    initial
        begin
            S1 = 0;
            S2 = 1;
            S3 = 1;
            A = 3'b000;
            #10
            S1 = 1;
            S2 = 0;
            S3 = 0;
            A = 3'b000;
            #50
            $stop;
        end

endmodule
```

wave


Untitled 4

Name	Value
Y[7:0]	00
[7]	0
[6]	0
[5]	0
[4]	0
[3]	0
[2]	0
[1]	0
[0]	0
A[2:0]	0
[2]	0
[1]	0
[0]	0
S1	0
S2	1
S3	1

行为型

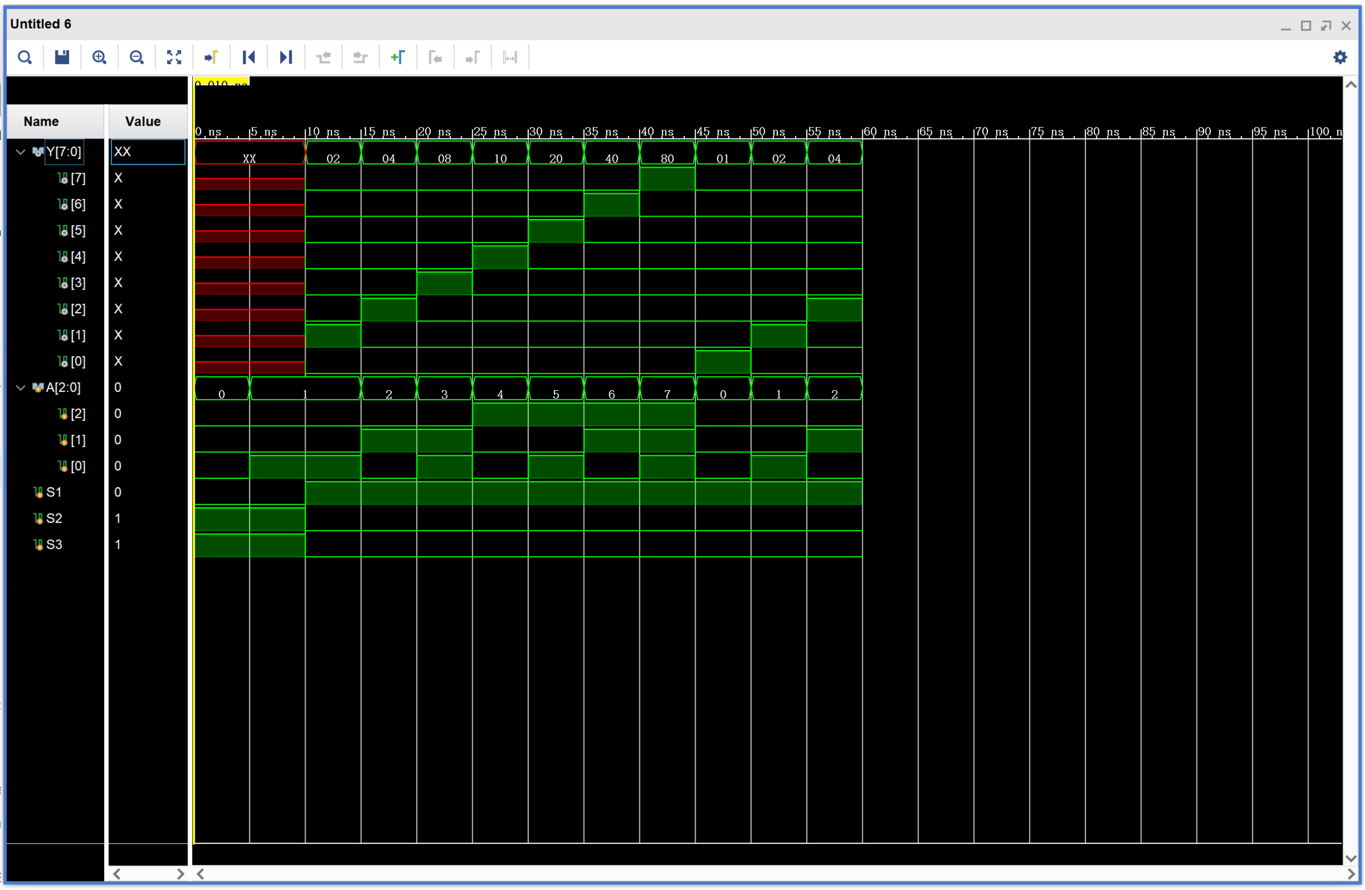
three_eight_behavior.v

```
`timescale 1ns / 1ps
//////////////////////////////////////////////////////////////////////////////////
// Company: HUST
// Engineer: Luo Chang
//
// Create Date: 2023/03/06 10:39:05
// Design Name:
// Module Name: three_eight_decoder_behavior
// Project Name:
// Target Devices:
// Tool Versions:
// Description:
//
// Dependencies:
//
// Revision:
// Revision 0.01 - File Created
// Additional Comments:
//
//////////////////////////////////////////////////////////////////////////////////


module three_eight_decoder_behavior(
    input [2:0] A,
    input S1,
    input S2,
    input S3,
    output reg [7:0] Y
    );

    always @(*)
        begin
            if (S1 == 1 && S2 == 0 && S3 == 0)
                begin
                    casex(A[2:0])
                       3'b000 : Y = 8'b0000_0001;
                       3'b001 : Y = 8'b0000_0010;
                       3'b010 : Y = 8'b0000_0100;
                       3'b011 : Y = 8'b0000_1000;
                       3'b100 : Y = 8'b0001_0000;
                       3'b101 : Y = 8'b0010_0000;
                       3'b110 : Y = 8'b0100_0000;
                       3'b111 : Y = 8'b1000_0000;
                    endcase
                end
       end
endmodule
```

testbench.v

```
`timescale 1ns / 1ps
//////////////////////////////////////////////////////////////////////////////////
// Company: HUST
// Engineer: Luo Chang
//
// Create Date: 2023/03/06 10:46:59
// Design Name:
// Module Name: testbench
// Project Name:
// Target Devices:
// Tool Versions:
// Description:
//
// Dependencies:
//
// Revision:
// Revision 0.01 - File Created
// Additional Comments:
//
//////////////////////////////////////////////////////////////////////////////////


module testbench;

    wire [7:0] Y;

    reg [2:0] A;
    reg S1, S2, S3;

    three_eight_decoder_behavior U0(A, S1, S2, S3, Y);
    initial $monitor($time,"A = %b,S1 = %b,S2 = %b,S3 = %b,Y = %b", A, S1, S2, S3, Y);

    always #5 A = A + 1'b1;
    initial
        begin
            S1 = 0;
            S2 = 1;
            S3 = 1;
            A = 3'b000;
            #10
            S1 = 1;
            S2 = 0;
            S3 = 0;
            A = 3'b000;
            #50
            $stop;
        end
endmodule
```

wave


Untitled 6
Name
Value
Y[7:0] XX
[7] X
[6] X
[5] X
[4] X
[3] X
[2] X
[1] X
[0] X
A[2:0] 0
[2] 0
[1] 0
[0] 0
S1 0
S2 1
S3 1
0 ns 5 ns 10 ns 15 ns 20 ns 25 ns 30 ns 35 ns 40 ns 45 ns 50 ns 55 ns 60 ns 65 ns 70 ns 75 ns 80 ns 85 ns 90 ns 95 ns 100 n
XX 02 04 08 10 20 40 80 01 02 04
0 1 2 3 4 5 6 7 0 1 2

实验结果分析

由以上对3-8线译码器的三种不同建模以及测试波形可得，三种方式的输出结果相同，均成功实现了3-8线译码器的功能

实验小结

通过本实验，我重新熟悉了verilog语言的三种编写方法，并掌握了新的编程软件vivado的使用方法。