实验二 3-8 译码器

实验介绍

本实验将使用 Verilog 语言设计方法实现 3/8 译码器的设计和仿真。

实验目标

1．学习设计一个 3/8 译码器；

2．学习设计仿真工具的使用方法。

3．学习如使用开发板

实验原理

接口定义，可以直接复制到文件中

（请务必按照接口定义编写代码，在将来的实验中也是如此，模块名也请按照给出的定义命名）

```
module decoder(
    input [2:0] data_in,    // 3 位输入，根据其输入，data_out 的相应位会被置为 0
    input ena,      // 1 位输入，当 ena 为高电平时译码器工作，否则 data_out 输出为全 1
    output [7:0] data_out // 8 位输出，相应位的值由 data_in 的值确定
    );
```

如下为 3-8 译码器的真值表：

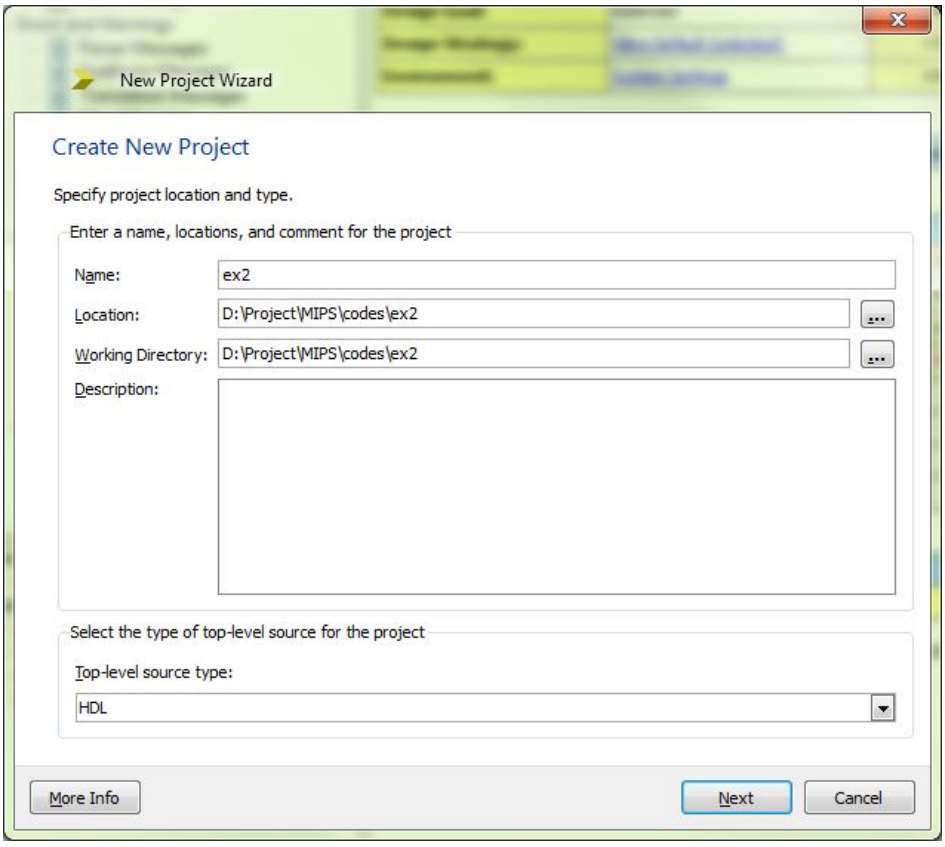
输入				输出							
ENA	A	B	C	Y7	Y6	Y5	Y4	Y3	Y2	Y1	Y0
1	0	0	0	1	1	1	1	1	1	1	0

1	0	0	1	1	1	1	1	1	1	0	1
1	0	1	0	1	1	1	1	1	0	1	1
1	0	1	1	1	1	1	1	0	1	1	1
1	1	0	0	1	1	1	0	1	1	1	1
1	1	0	1	1	1	0	1	1	1	1	1
1	1	1	0	1	0	1	1	1	1	1	1
1	1	1	1	0	1	1	1	1	1	1	1
0	X	X	X	1	1	1	1	1	1	1	1

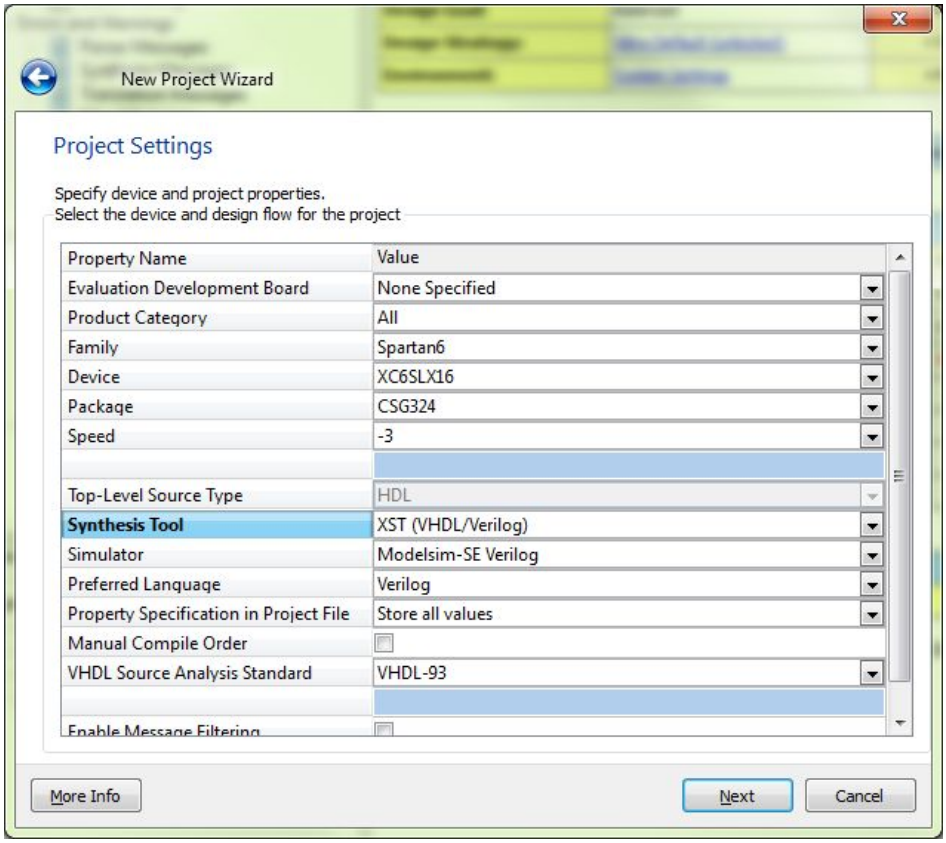
实验步骤

1. 新建 ISE 工程

a) 在 ISE 中点击 File -> New Project，按照弹出的对话框提示新建工程。

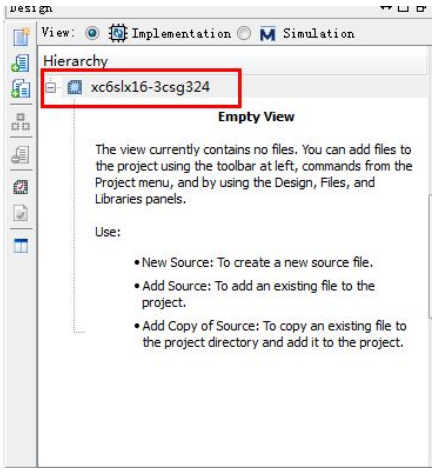


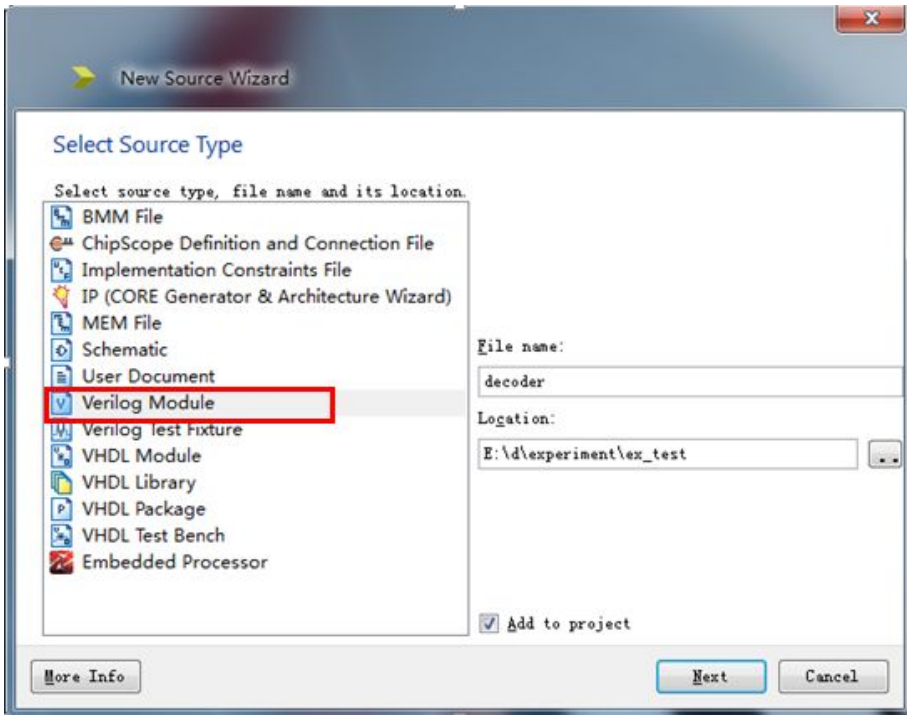
b) 在 Project Setting 页面，我们选用的开发版为 Spartan6 系列，XC6SLX16；Simulator 使用 Modelsim-SE Verilog；Preferred language 选择 Verilog。



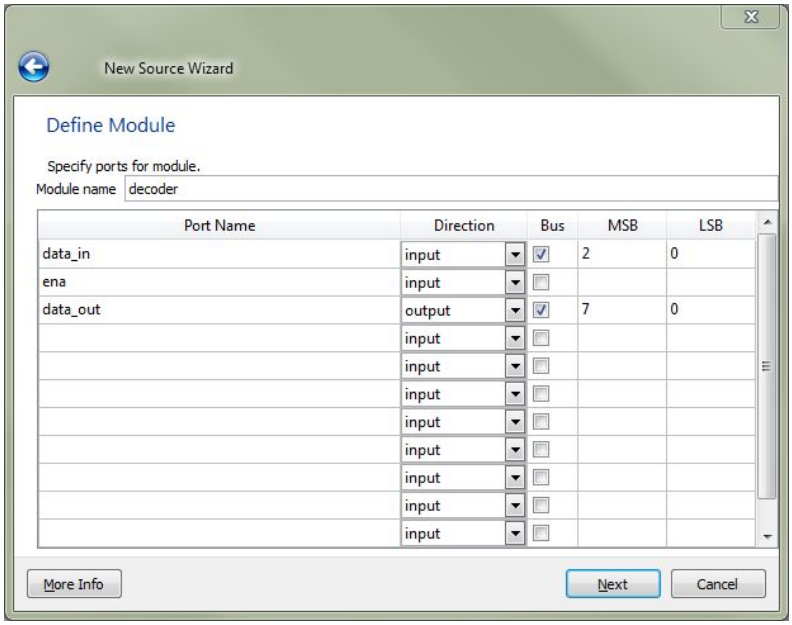
2. 编写模块

a) 右键点击红框出，点击 New Source，在弹出的对话框中选择 Verilog Module 添加模块，并且输入 Verilog 文件名。





b) 单击 Next 进入 Define Module。Port Name 表示端口名称，Direction 表示端口方向（可选择为 input、output 或 inout），MSB 表示信号最高位，LSB 表示信号最低位，对于单信号的 MSB 和 LSB 不用填写。端口定义这一步我们也可以略过，在源程序中再行添加。若进行端口定义，本实验中的设置如下：

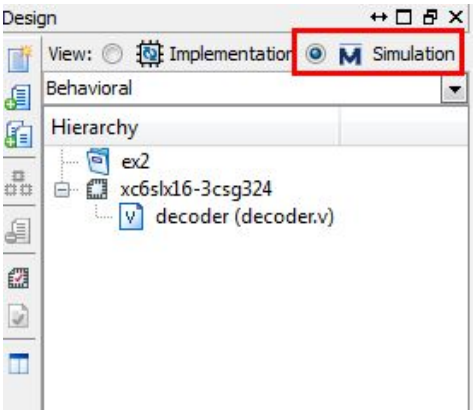


c) 编写代码，可参考附录中的代码

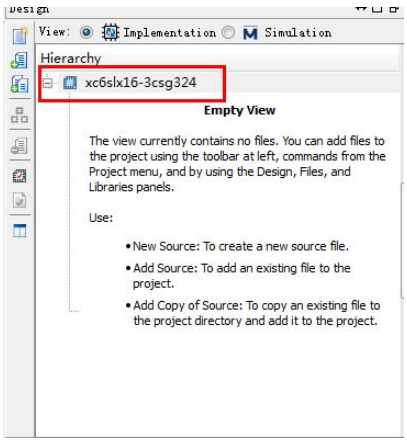
3. 行为仿真

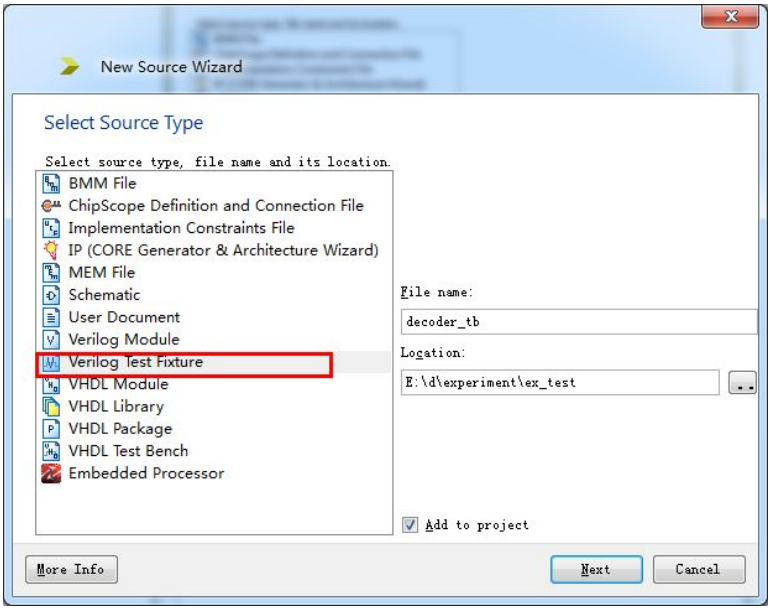
行为仿真用以检测设计的逻辑正确性

a) 完成代码编写后，对模块进行测试。在工程管理区勾选 Simulation。

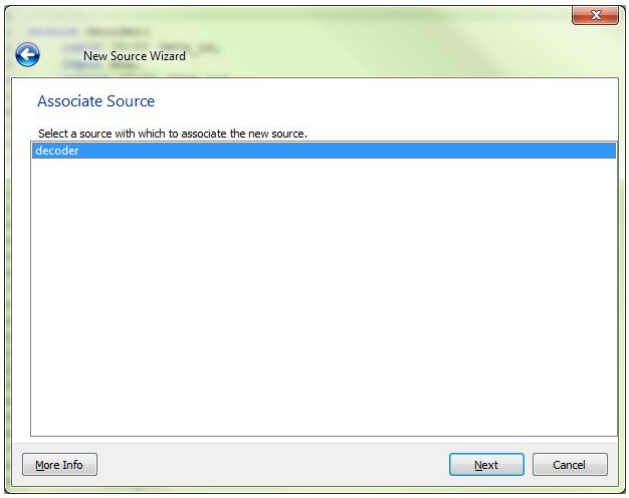


b) 右键点击红框处，点击 new source，选择 Verilog Test Fixture



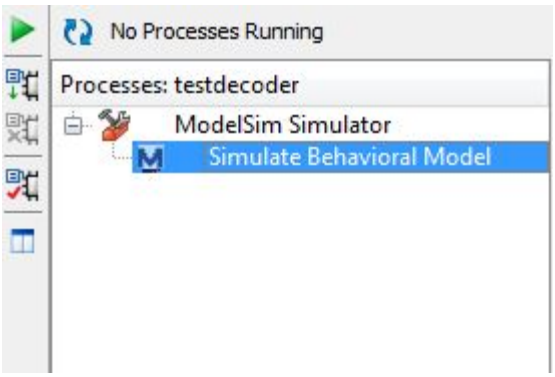


c) 选择要进行测试的模块名称，本实验中只有 decoder 模块。

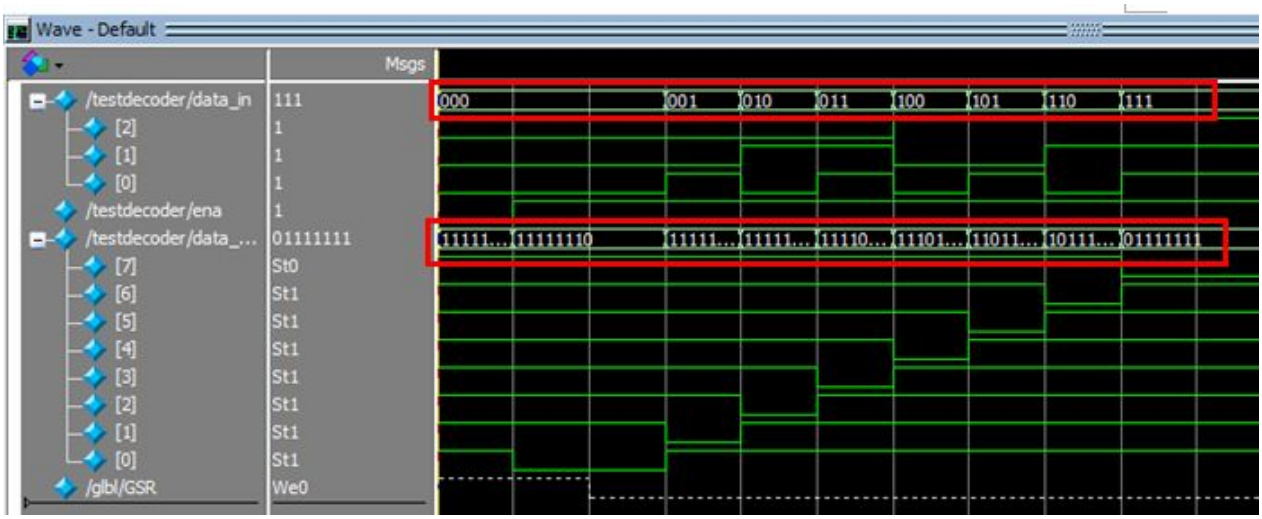


d) 编写测试代码，可参考附录

e) 完成测试代码的编写后，双击 Simulate Behavioral Model，进行仿真。



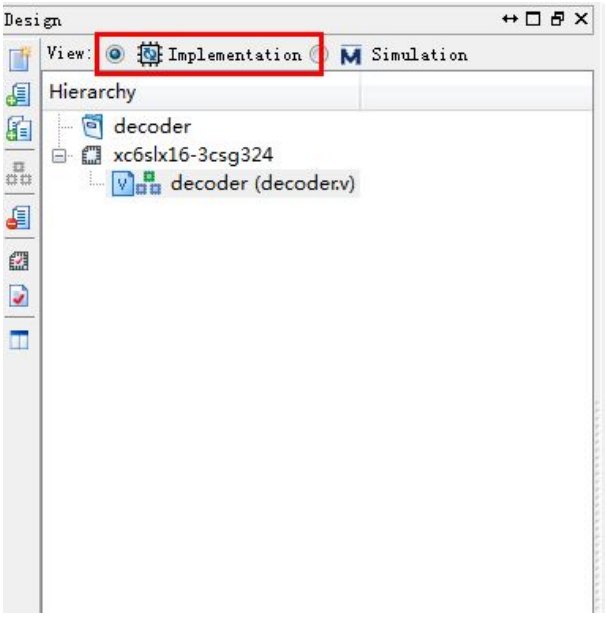
f) 观察波形，验证程序的正确性。本例中查看 data_out 的值是否和 data_in 的值对应，即图中红框处的值



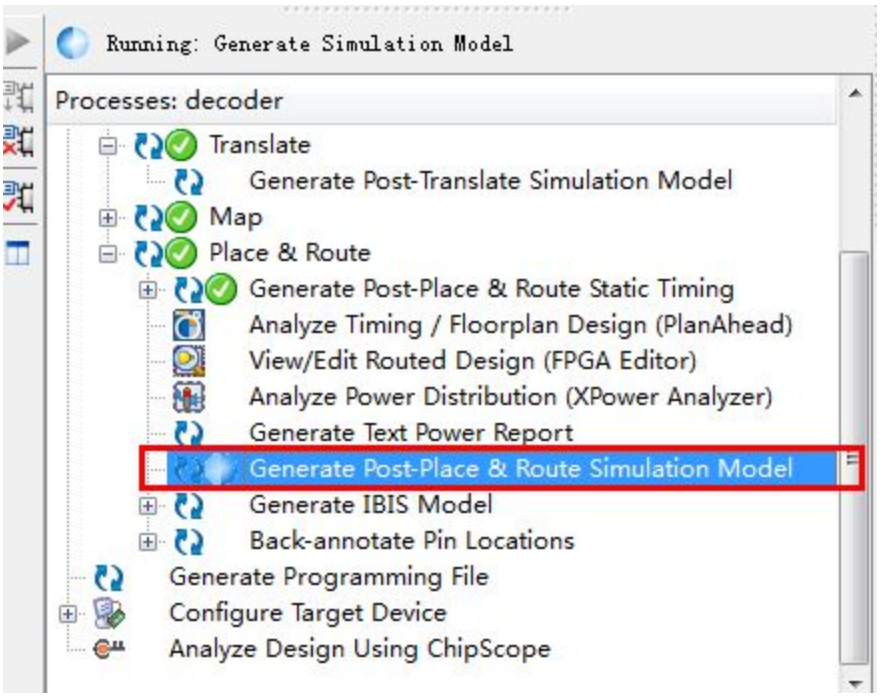
4.后仿真

后仿真会将模块的延时加入仿真模型，此模型将真实地反映出模块在开发板上运行的过程，根据延时后的模型确定 clk 频率等必要参数。所以在下板前，务必使用后仿真来验证设计的结果

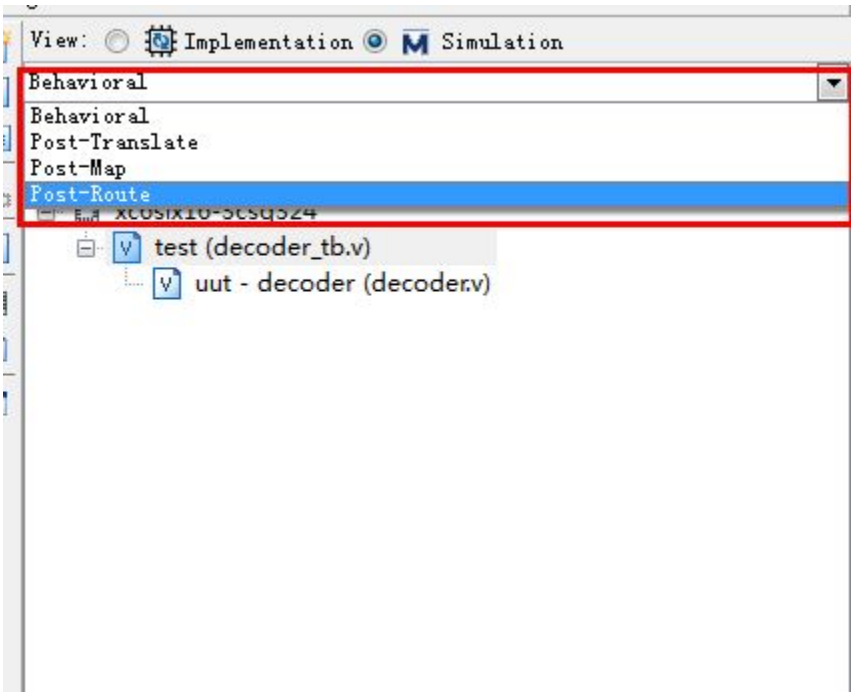
A)选择 implementation



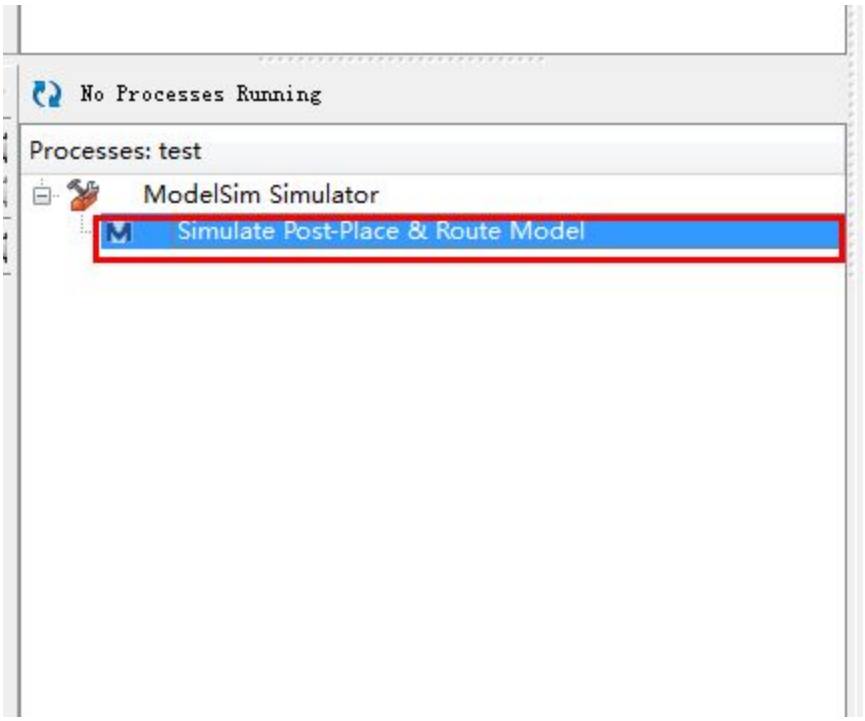
B)双击红框所示处，等待绿钩出现

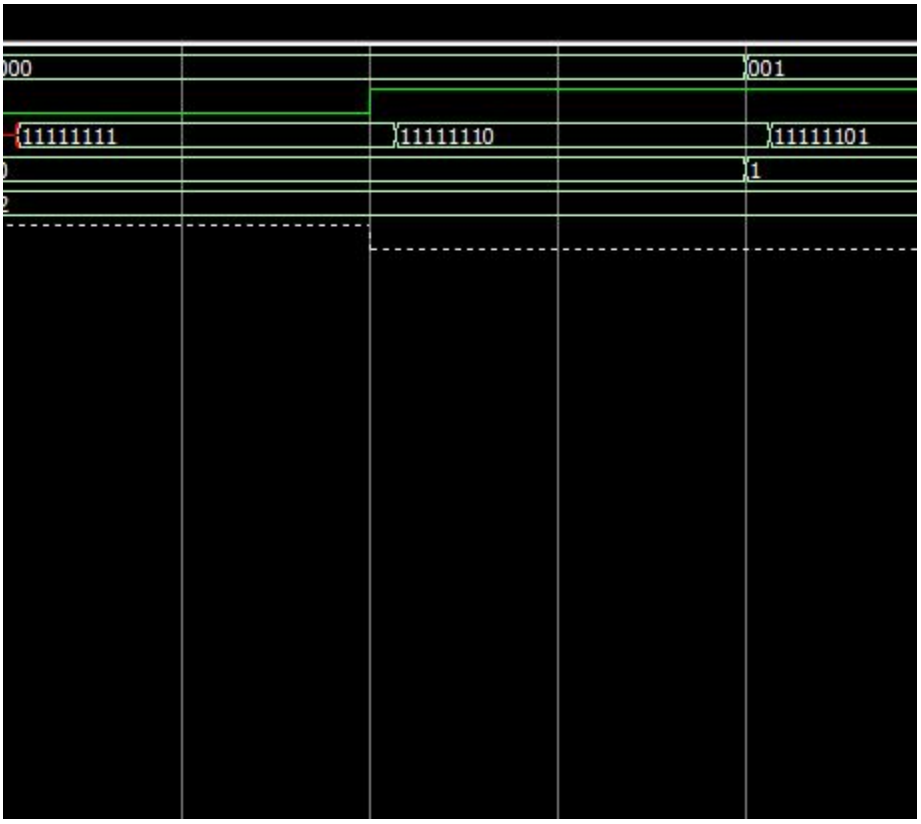


C)去到 simulator 下，选择 post-route



D)再次使用 modelsim 查看波形时就能够看到加入延时后的效果

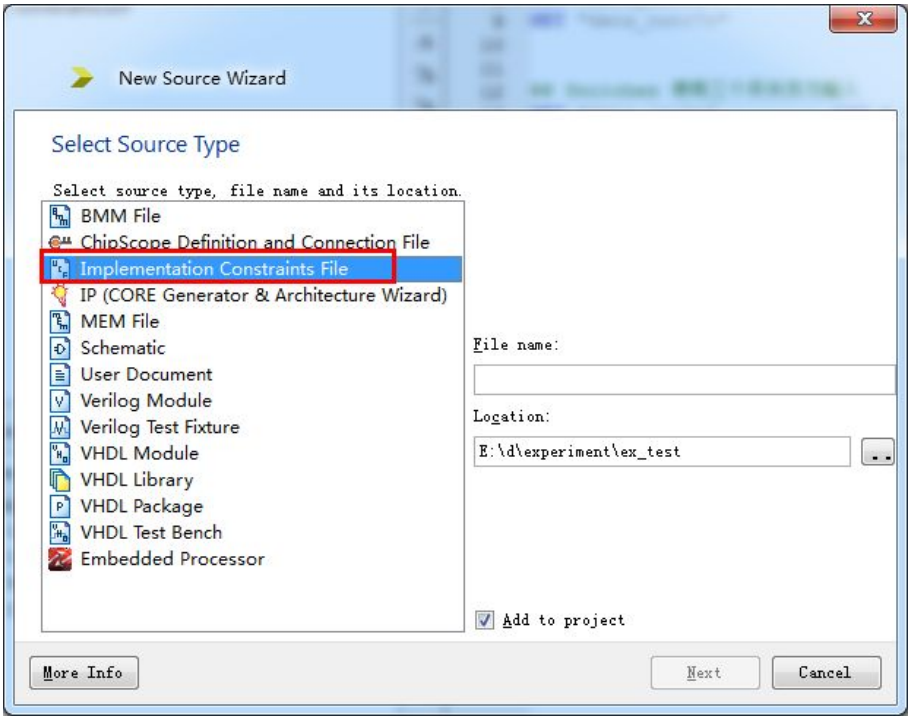

No Processes Running
Processes: test
ModelSim Simulator
Simulate Post-Place & Route Model


000
001
11111111
11111110
11111101
1

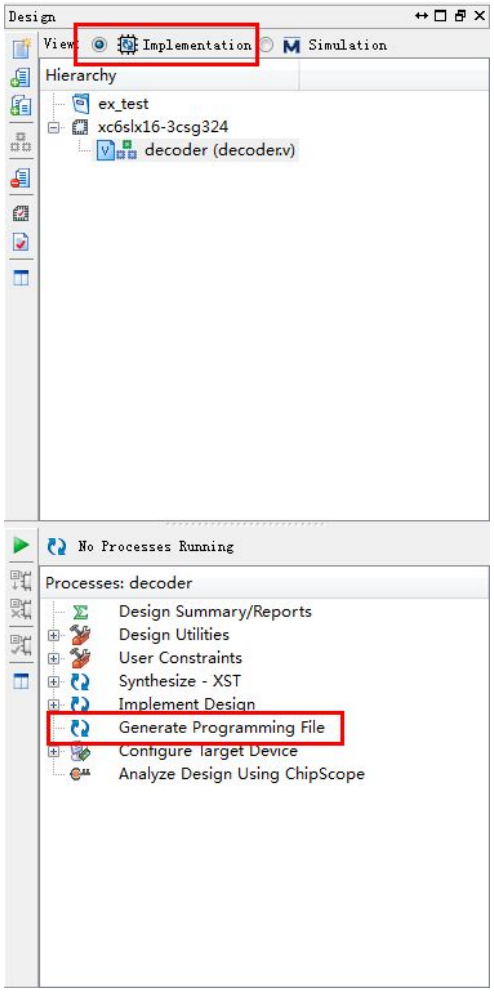
5.将模块下载到开发板

a)添加约束文件，右键工程选择 New Resource，选择 Implementation Constraints File，在工程中新建的 ucf 文件中写入如下几行（完整的 ucf 文件可以从课程网站上获得）

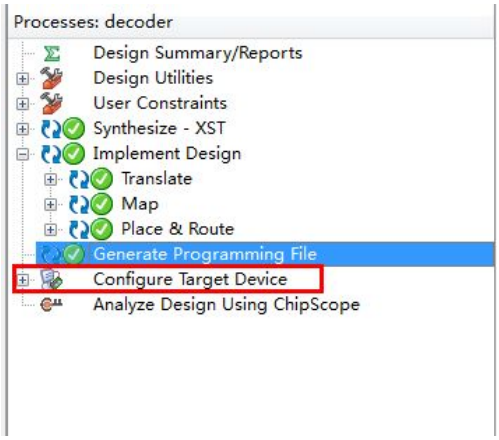
```
## Leds 将输出绑定到 LED 灯端口

NET "data_out<0>"       LOC = "U16" | IOSTANDARD = "LVCMOS33";   #Bank = 2, Pin name = IO_L2P_CMPCLK,          Sch name = LD0

NET "data_out<1>"       LOC = "V16" | IOSTANDARD = "LVCMOS33";   #Bank = 2, Pin name = IO_L2N_CMPMOSI,          Sch name = LD1

NET "data_out<2>"       LOC = "U15" | IOSTANDARD = "LVCMOS33";   #Bank = 2, Pin name = IO_L5P,                Sch name = LD2

NET "data_out<3>"       LOC = "V15" | IOSTANDARD = "LVCMOS33";   #Bank = 2, Pin name = IO_L5N,                Sch name = LD3

NET "data_out<4>"       LOC = "M11" | IOSTANDARD = "LVCMOS33";   #Bank = 2, Pin name = IO_L15P,               Sch name = LD4

NET "data_out<5>"       LOC = "N11" | IOSTANDARD = "LVCMOS33";   #Bank = 2, Pin name = IO_L15N,               Sch name = LD5

NET "data_out<6>"       LOC = "R11" | IOSTANDARD = "LVCMOS33";   #Bank = 2, Pin name = IO_L16P,               Sch name = LD6

NET "data_out<7>"       LOC = "T11" | IOSTANDARD = "LVCMOS33";   #Bank = 2, Pin name = IO_L16N_VREF,           Sch name = LD7

## Switches 使用三个开关作为输入

NET "data_in<0>"        LOC = "T10" | IOSTANDARD = "LVCMOS33";   #Bank = 2, Pin name = IO_L29N_GCLK2,          Sch name = SW0

NET "data_in<1>"        LOC = "T9"  | IOSTANDARD = "LVCMOS33";   #Bank = 2, Pin name = IO_L32P_GCLK29,          Sch name = SW1

NET "data_in<2>"        LOC = "V9"  | IOSTANDARD = "LVCMOS33";   #Bank = 2, Pin name = IO_L32N_GCLK28,          Sch name = SW2

## Buttons 将 ena 约束到按键上

NET "ena"       LOC = "B8"  | IOSTANDARD = "LVCMOS33";   #Bank = 0, Pin name = IO_L33P, Sch name = BTNS
```



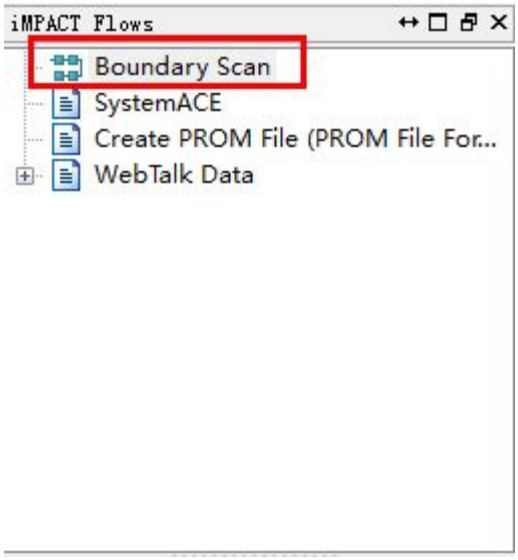
b)在 implementation 状态下，双击 generate programming file



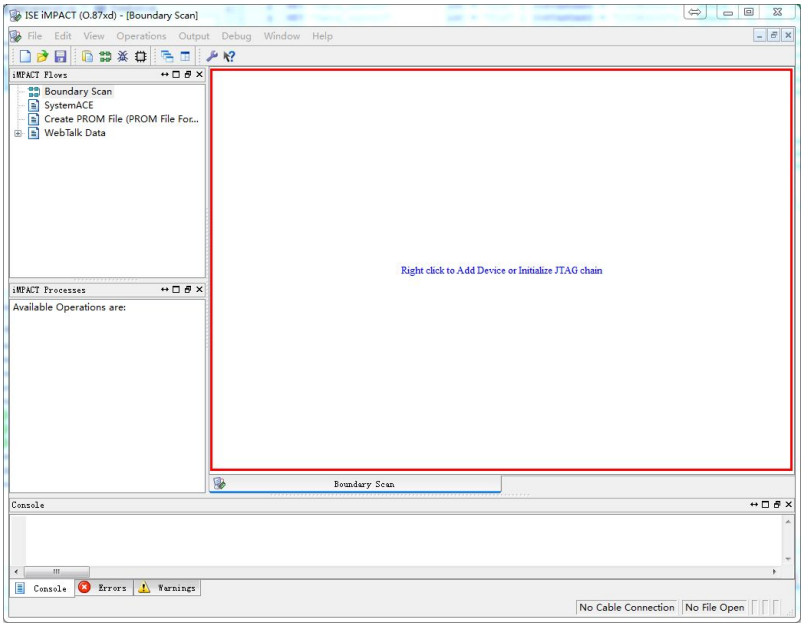
c) 双击 Configure Target Device



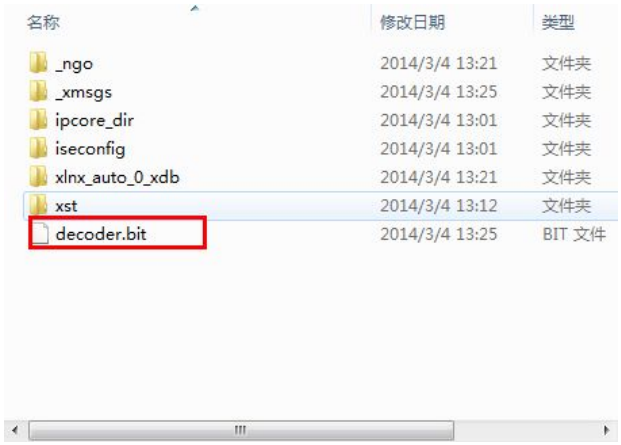
d)双击 boundary scan



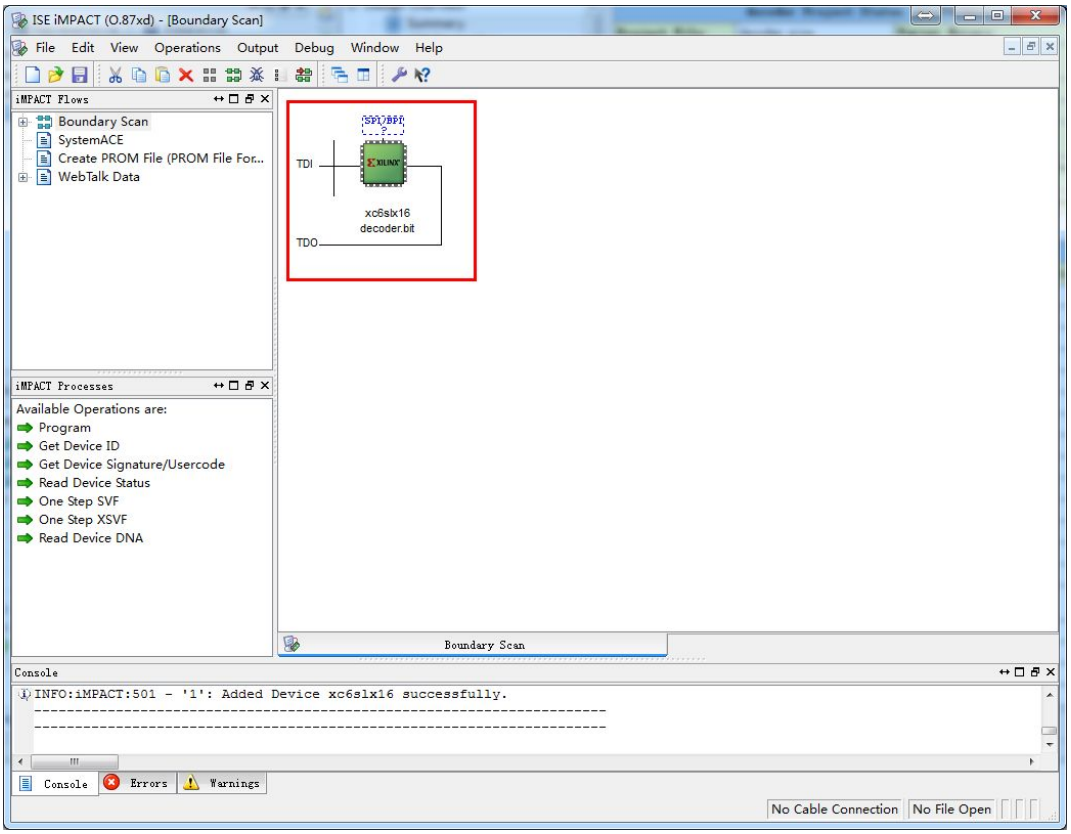
e)右键点击右边空白区域，选择 Add Xilinx Device



f) 到项目的文件夹下选择生成的 bit 文件



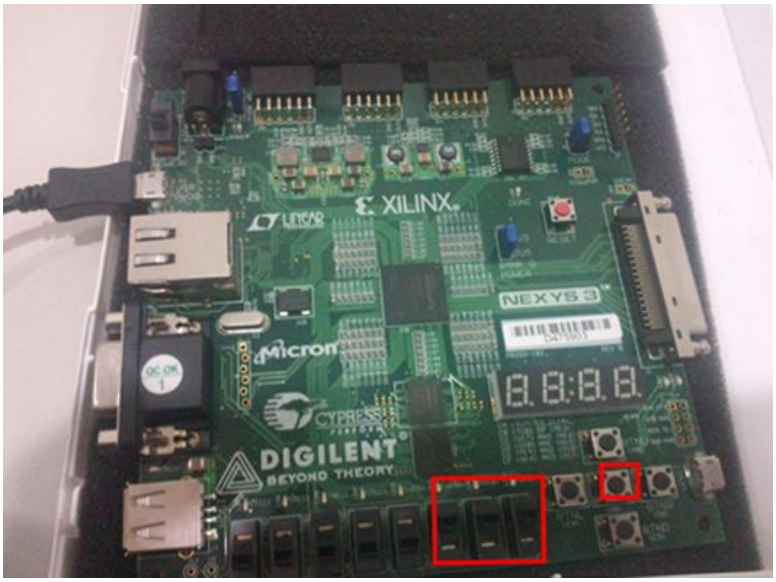
g)右键点击红色区域，选择 program



h)等待 Program Succeeded 出现

Program Succeeded

i) 拨动开发板上的开关的底三位，并按下开发板上的十字开关观察实验现象



附录：（直接拷贝代码可能会出现错误）

3-8 译码器代码：

```
module decoder(
   input [2:0] data_in,
   input ena,
   output [7:0] data_out
   );
            reg[7:0] data_temp;
           assign data_out = data_temp;
            always @ (ena or data_in)  begin
                     if (ena == 1)
                              case (data_in)
                                       3'b000:
                                                data_temp= 8'b11111110;
                                       3'b001:
                                                data_temp= 8'b11111101;
                                       3'b010:
                                                data_temp= 8'b11111011;
                                       3'b011:
                                                data_temp= 8'b11110111;
                                       3'b100:
                                                data_temp= 8'b11101111;
```

```
                3'b101:
                    data_temp= 8'b11011111;
                3'b110:
                    data_temp= 8'b10111111;
                3'b111:
                    data_temp= 8'b01111111;
            endcase
        else
            data_temp= 8'b11111111;

    end

endmodule
```

3-8 译码器 testbench 代码

```
module decoder_tb;
	// Inputs
	reg [2:0] data_in;
	reg ena;

	// Outputs
	wire [7:0] data_out;
	integer count;

	// Instantiate the Unit Under Test (UUT)
	decoder uut (
		.data_in(data_in),
		.ena(ena),
		.data_out(data_out)
	);
	initial begin
		// Initialize Inputs
		data_in = 0;
		ena = 0;
		count = 0;
```

```
        // Wait 100 ns for global reset to finish
        #100;
        ena = 1;                                    //enable 位置为有效
        while(count < 8)begin                                   //循环改变输入
            $display("%b %b %b\n", ena, data_in, data_out);
            #100                                    //延时
            data_in = data_in + 1;
            count = count + 1;
        end
        #100;
        count = 0;
        data_in = 0;
        ena = 0;                                //enable 位置为无效
        while(count < 8)begin                                   //循环改变输入
            $display("%b, %b, %b\n", ena, data_in, data_out);
            #100                                    //延时
            data_in = data_in + 1;
            count = count + 1;
        end
    end
endmodule
```