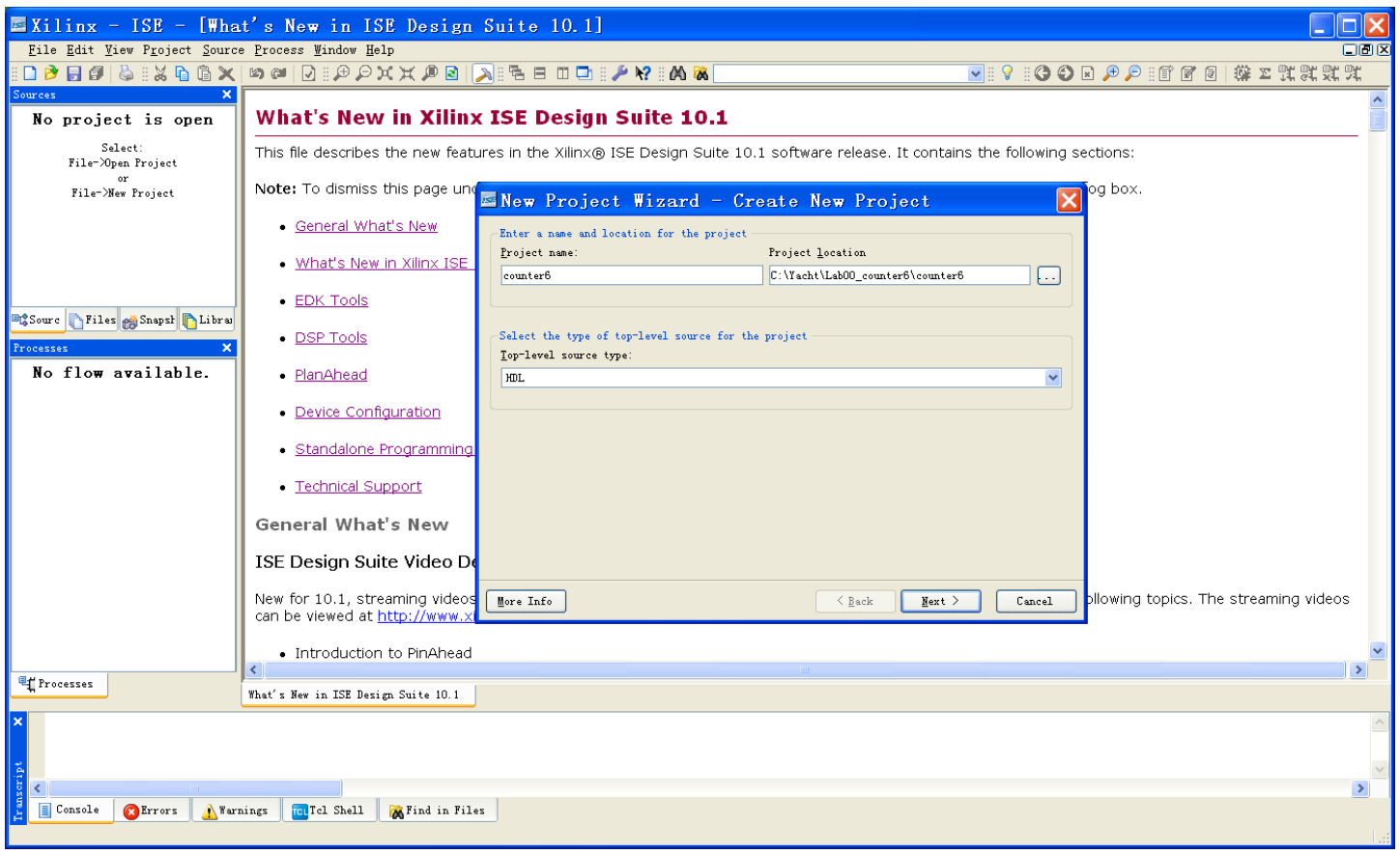
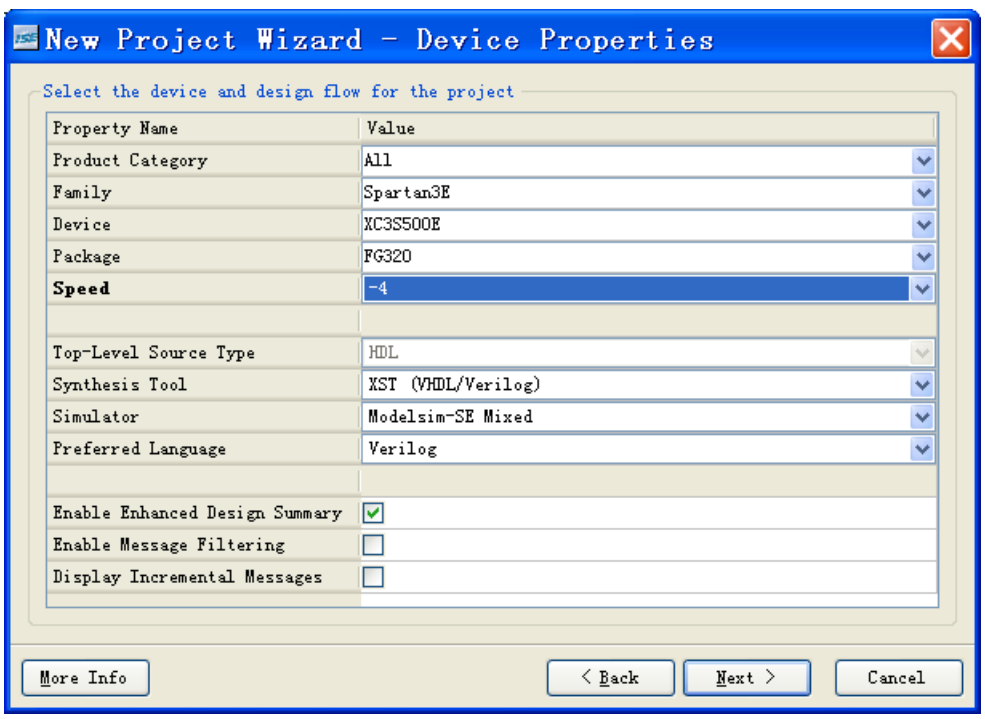
在 Xilinx ISE 10.1 中使用 ModelSim 进行功能仿真和综合实现之后的后仿真

一、新建 Xilinx ISE 工程

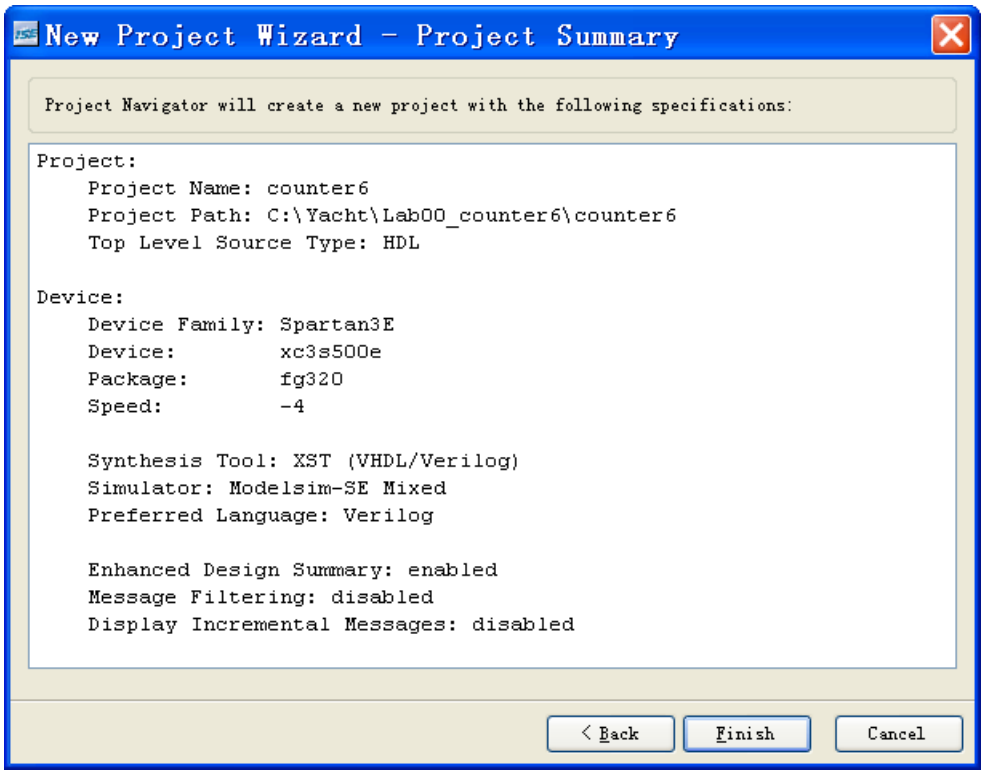
（1）新建一个 ISE 工程，ISE 工程名为：counter6



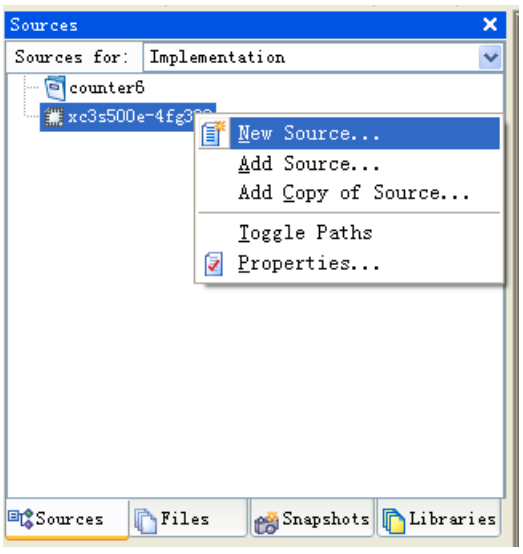
设置 ISE 工程的 FPGA 器件属性：



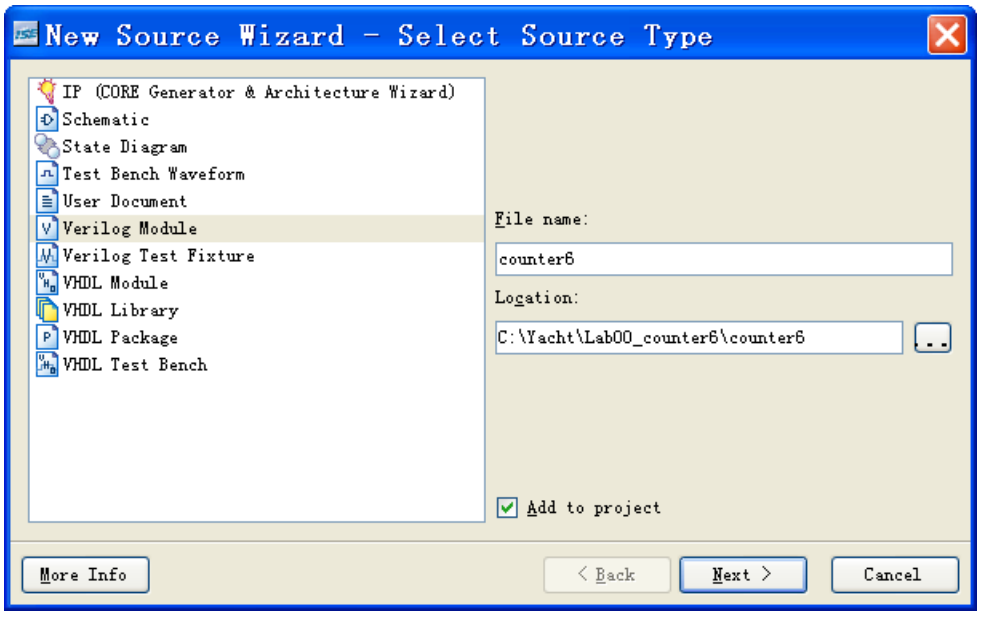
新建工程 FPGA 器件，以及综合工具和仿真工具的设置信息：



（2）新建一个 Verilog 文件：



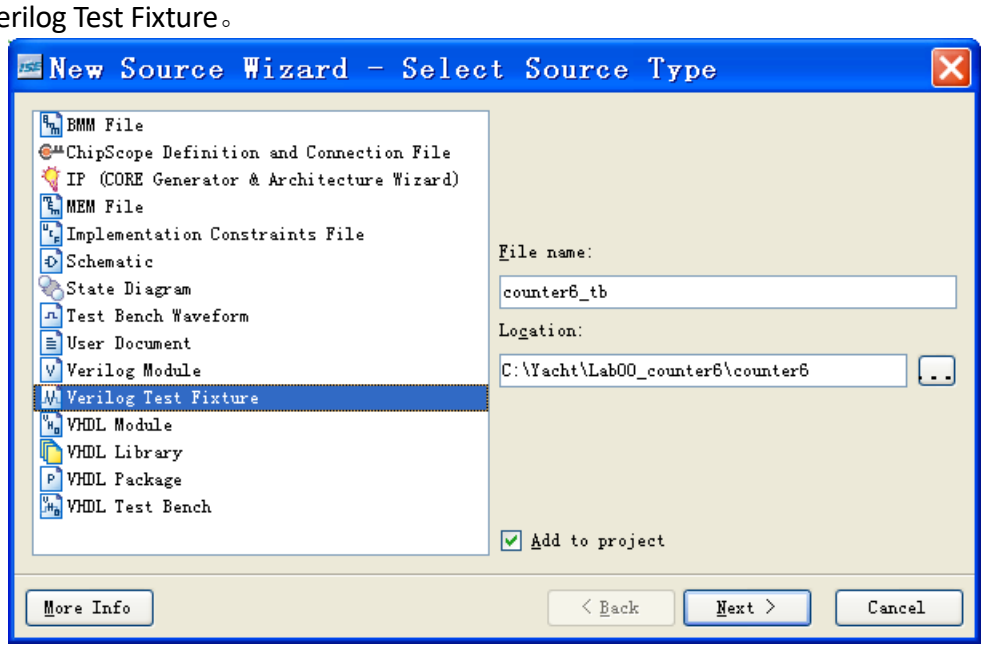
（3）选择 Verilog Module 输入 File name 为 counter6，单击 next 默认，直到 Finish。



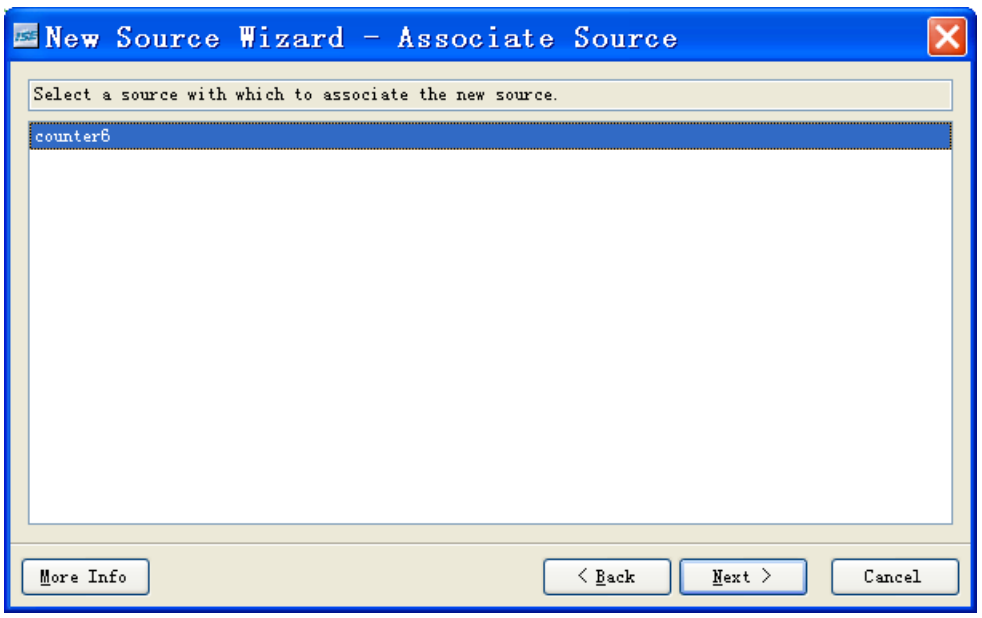
（4）在 counter6.v 文件中输入以下代码：

```
module counter6(output reg [5:0] cnt, input clk, reset );
    always@(posedge clk)
    begin
        if(reset)
            cnt <= 6'b0;
        else
            cnt <= cnt + 1'b1;
    end
endmodule
```

（5）新建一个 Verilog Testbench 文件（用于仿真，不对其进行综合），操作步骤与第（2）步相同，这里，文件类型选择 Verilog Test Fixture。



选择新建仿真测试平台关联的被测试模块的源文件：



（7）测试平台模块文件创建完成后，ISE 自动生成一个测试平台编码模板：

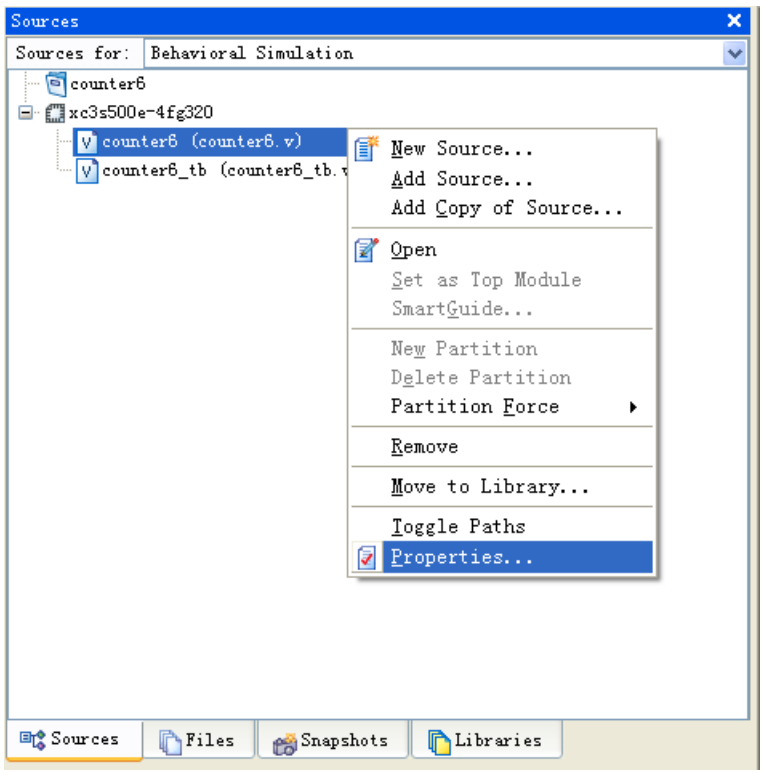
```
module counter6_tb;
    // Outputs
    wire ;
    // Instantiate the Unit Under Test (UUT)
    counter6 uut (
        .()
    );

    initial begin
        // Initialize Inputs

        // Wait 100 ns for global reset to finish
        #100;

        // Add stimulus here
    end

endmodule
```

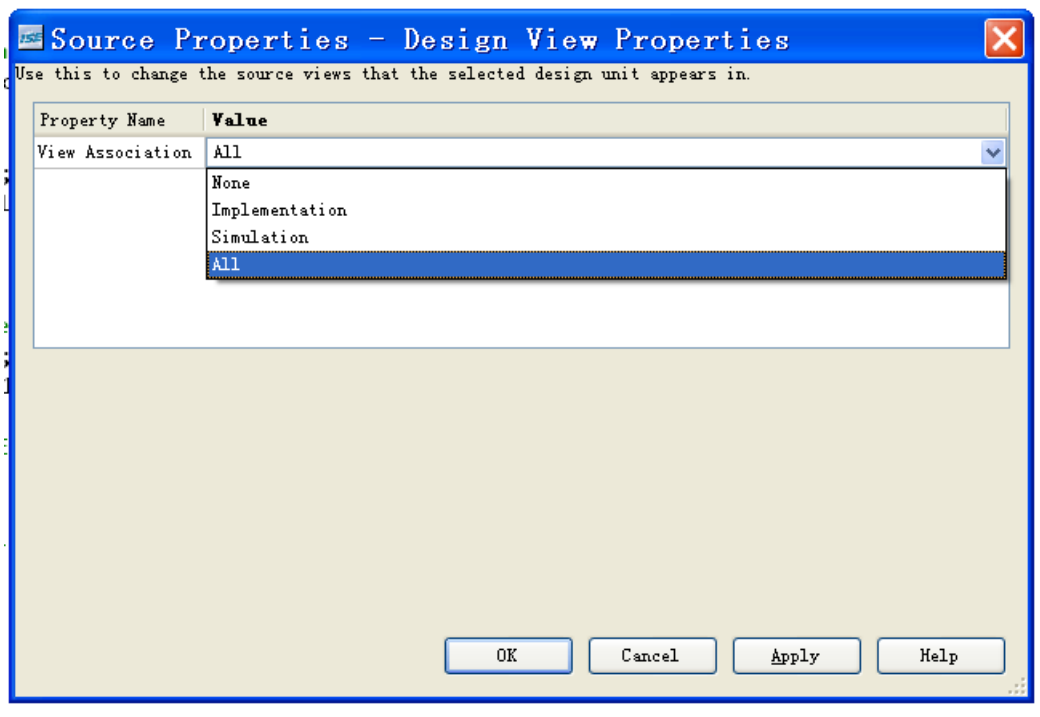
按照仿真测试的内容，对 counter6_tb 模块进行修改：

```
module counter6_tb;
    parameter DELAY = 100;
    // Outputs
    wire [5:0] p_cnt;
    // Inputs
    reg p_clk, p_rst;
    // Instantiate the Unit Under Test (UUT)
    counter6 uut ( .cnt(p_cnt), .clk(p_clk) , .reset(p_rst) );

    initial begin
        p_clk = 1'b0;
        forever #(DELAY/2) p_clk = ~p_clk;
    end

    initial begin
        // Initialize Inputs
        p_rst = 1'b1;
        #180 p_rst = 1'b0;
        // Wait 40*DELAY ns for global reset to finish
        #(40*DELAY)  $finish;

        // Add stimulus here
    end

    initial
        $monitor("At time t=%t, clk=%d, reset=%d, cnt=%d",
                 $time, p_clk, p_rst, p_cnt);

endmodule
```

注：可以修改工程中 Verilog 源文件的属性，例如，某个 Verilog 文件是实现（Implementation）电路，只用于综合成电路的 Verilog 模块，还是仿真（Simulation），只用于仿真的 Verilog 模块，还是 All，既用于实现也用于仿真。

要修改右键点击要修改的文件，选择 Properties，



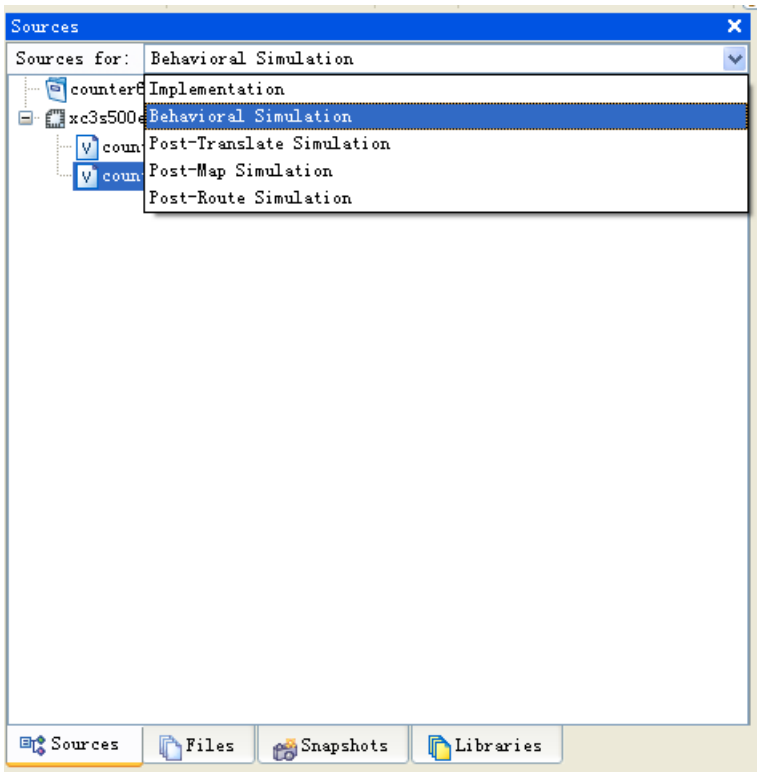


这里，显然，counter6.v 的属性为 All，既用于实现也用于仿真。而 counter6_tb.v 的文件属性为 Simulation）。

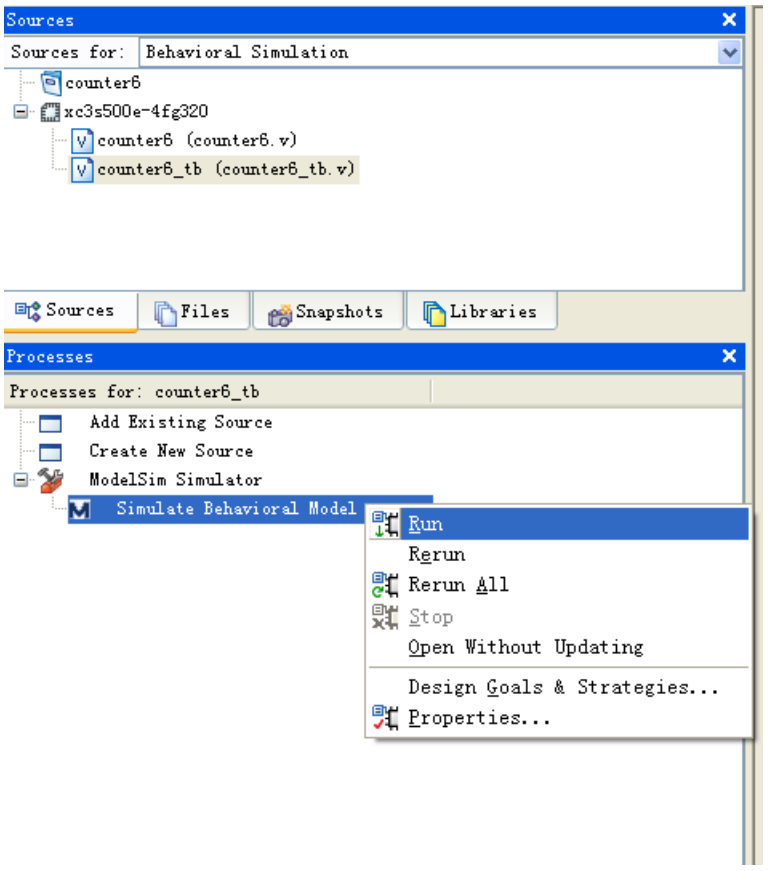
二、行为仿真

行为仿真也就是对 Verilog 设计模块进行功能仿真。在综合之前，可以先进行功能仿真，确保模块设计代码的正确性。由于大型程序的综合需要很长的时间，功能仿真正确后，再对模块进行综合，有利于提高开发效率。

（1）在 Sources 窗口，切换到 Behavioral Simulation

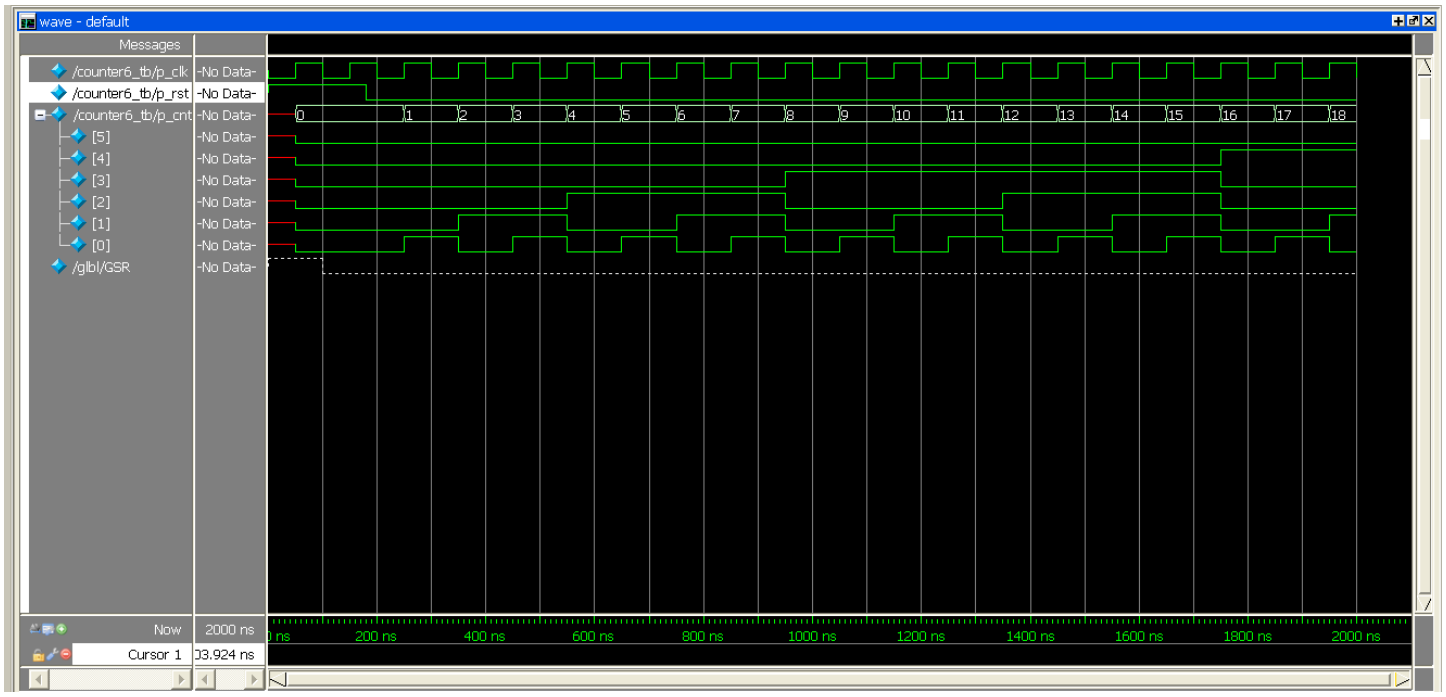


（2）、单击选中 counter6_tb 文件，然后，右键点击 Simulate Behavioral Model，出现弹出菜单：

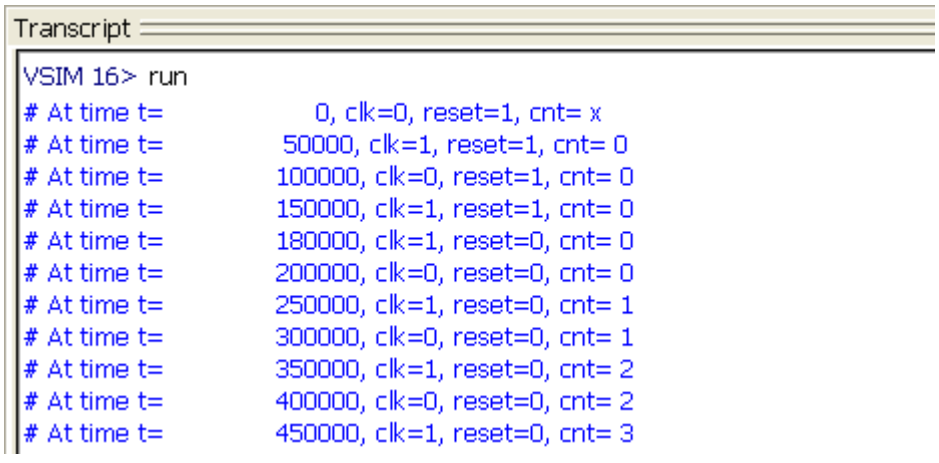


选择“Run”等，运行 ModelSim 进行行为（功能）仿真：

（3）、如果 Verilog 设计模块和仿真测试平台模块代码没有语法/语义错误的话，行为仿真完成后，可以在 ModelSim 中查看波形：



监控或显示输出信息：

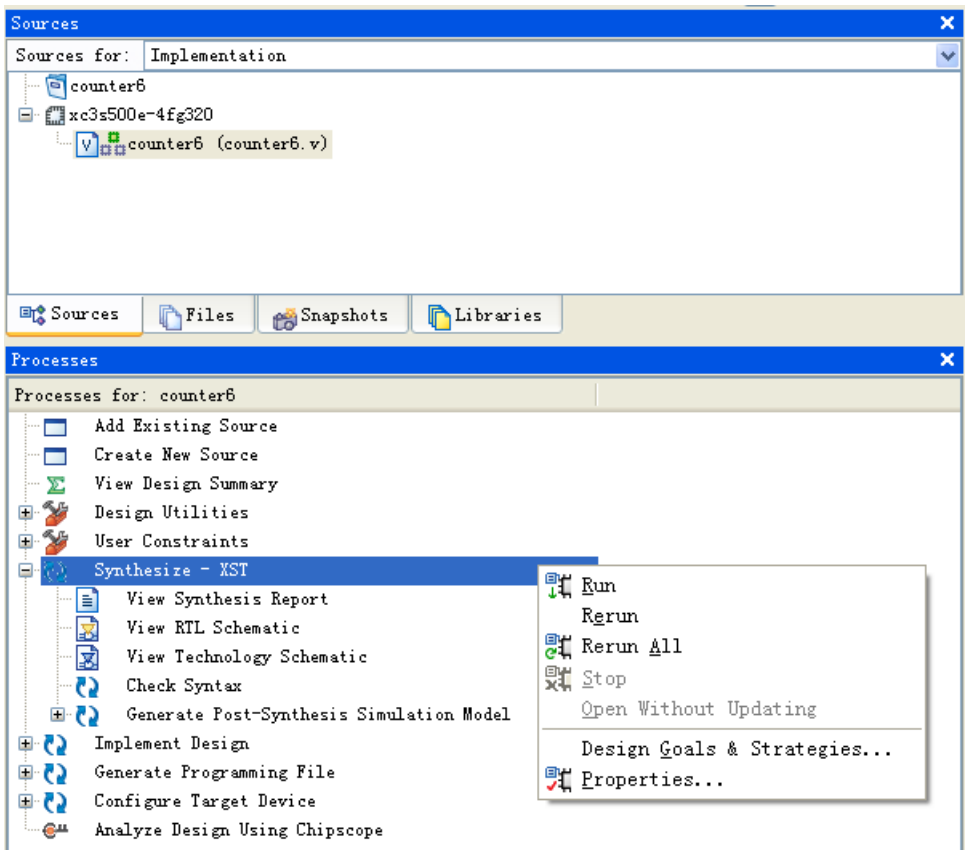


三、时序（综合后）仿真

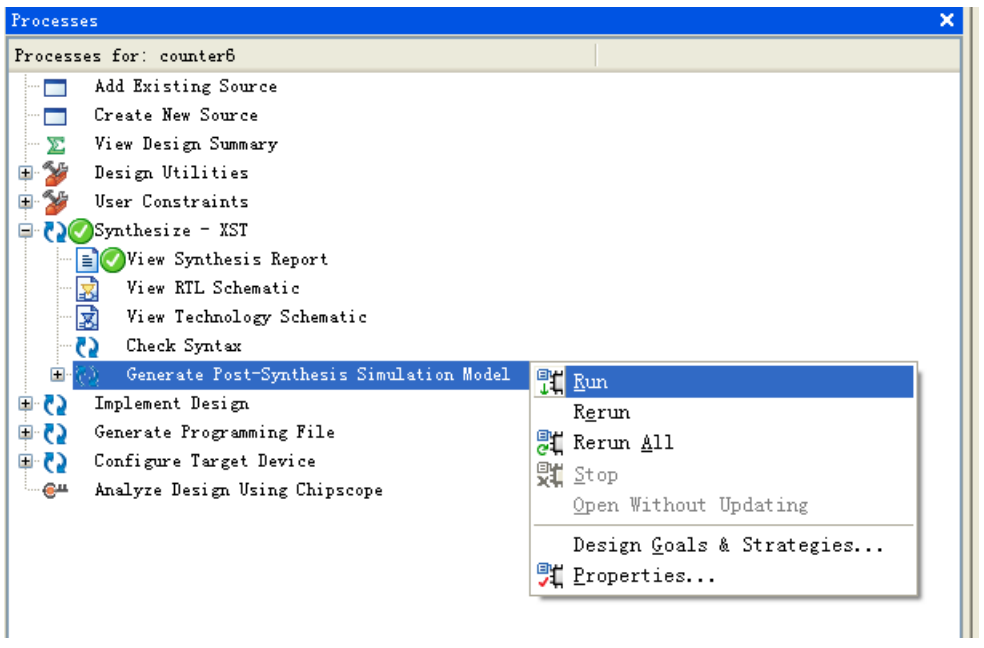
综合（Synthesize）是将 HDL 语言描述的设计输入，翻译成由与、或、非门、触发器等，以及 RTL 级逻辑单元组成和连接的网表。时序仿真将考虑综合后产生的基本门、触发器和其它基本逻辑器件的时延，还有布局布线产生的时延。

在 Sources 窗口，点击选中设计文件：counter.v，在“Sources for”选项中，选择：Implementation（实现），相应的 Processes（处理）窗口中的工具也随之改变。

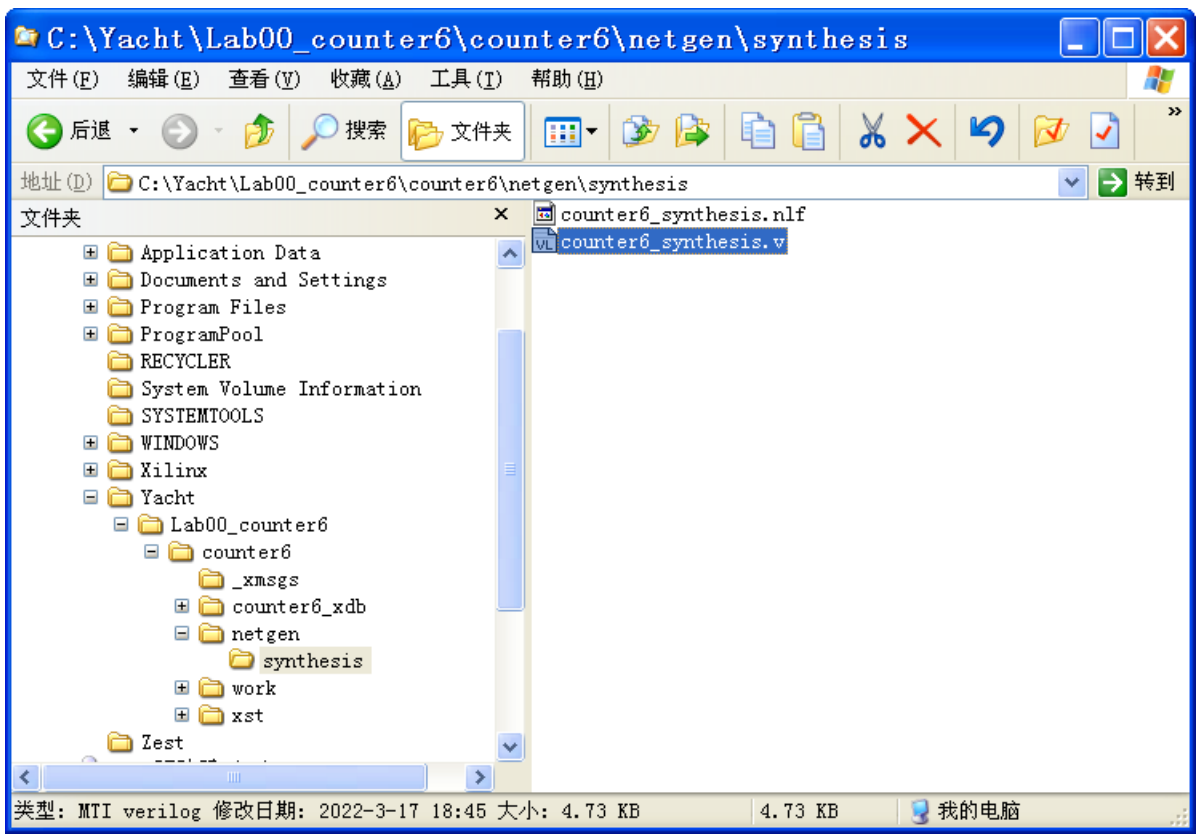
（1）进行综合，右键点击 Synthesize – XST，对设计模块进行综合。



综合完成后，运行 Generate Post-Synthesis Simulation Model，可生成综合后仿真模型（Generate Post-Synthesis Simulation Model）。



产生的综合后仿真模型在工程文件夹中的子文件夹：netgen\synthesis\中，生成 counter6_synthesis.v 等文件。



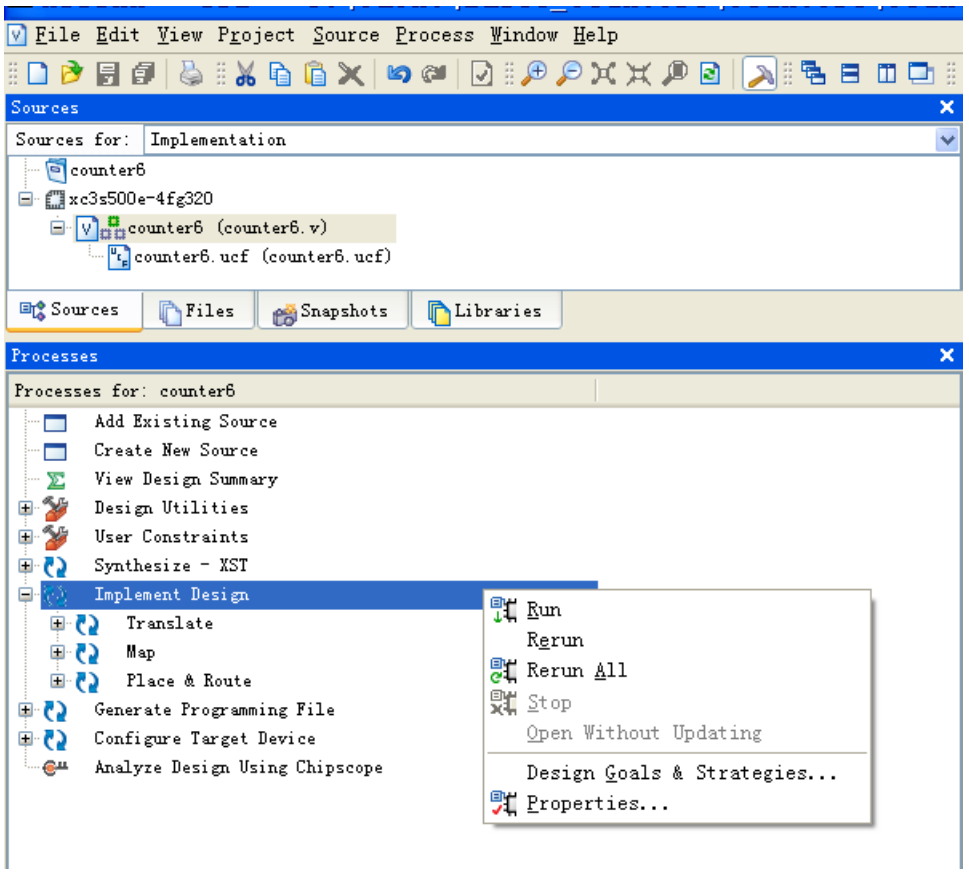
综合后，进行 ISE 的实现（Implement Design），包括翻译、映射、布局布线。在这三个过程中都可以生成一个仿真模型（翻译和映射不会产生延时，因此，常用布局布线后产生的仿真模型进行时序仿真）。

在进行设计实现（Implement Design）之前，首先需要添加用户约束文件：

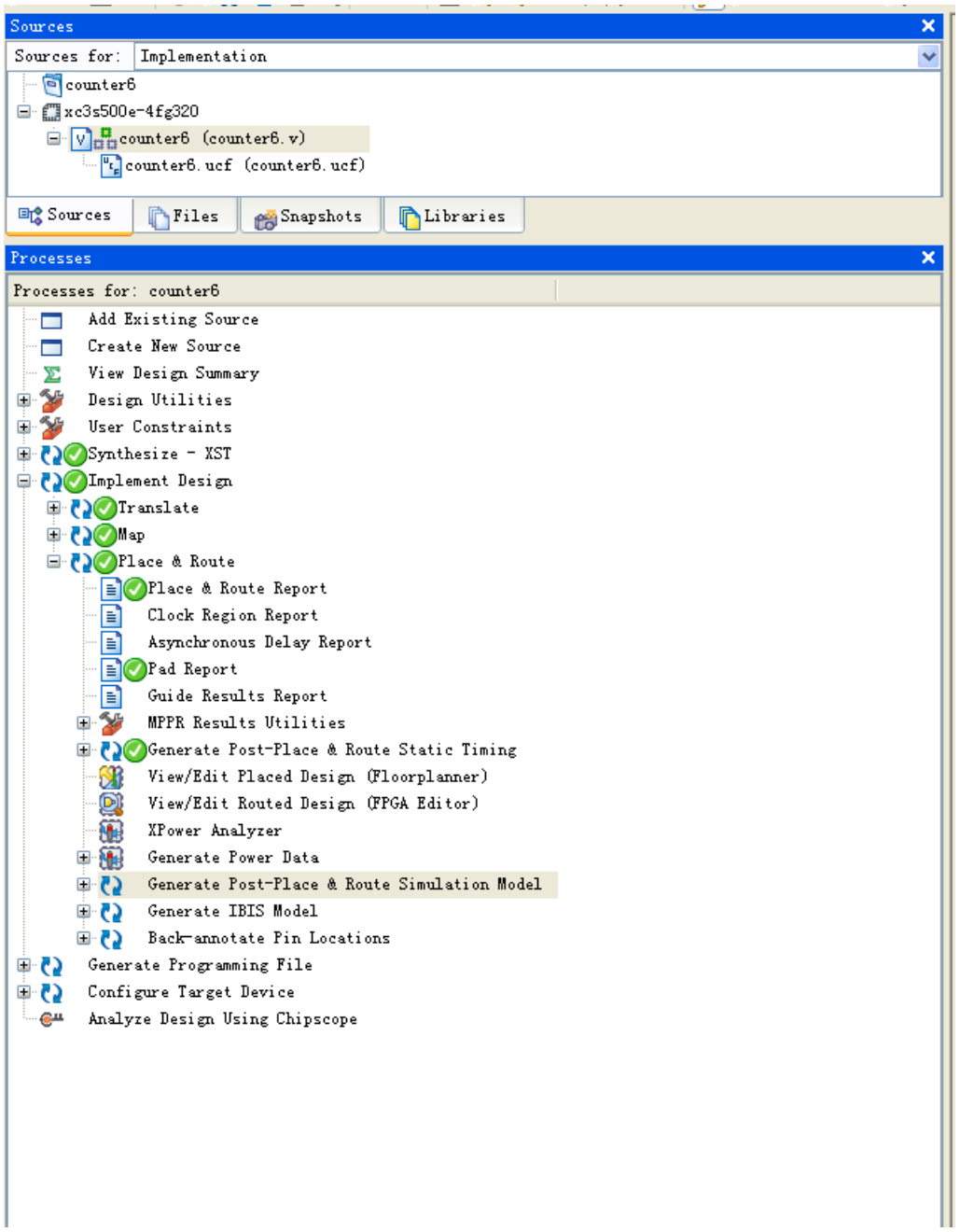
```
NET "cnt<5>" LOC = "D11" | IOSTANDARD = LVTTL | SLEW = SLOW | DRIVE = 8 ;
NET "cnt<4>" LOC = "C11" | IOSTANDARD = LVTTL | SLEW = SLOW | DRIVE = 8 ;
NET "cnt<3>" LOC = "F11" | IOSTANDARD = LVTTL | SLEW = SLOW | DRIVE = 8 ;
NET "cnt<2>" LOC = "E11" | IOSTANDARD = LVTTL | SLEW = SLOW | DRIVE = 8 ;
NET "cnt<1>" LOC = "E12" | IOSTANDARD = LVTTL | SLEW = SLOW | DRIVE = 8 ;
NET "cnt<0>" LOC = "F12" | IOSTANDARD = LVTTL | SLEW = SLOW | DRIVE = 8 ;

NET "clk" LOC = "C9" | IOSTANDARD = LVCMOS33 ;

NET "reset" LOC = "N17" | IOSTANDARD = LVTTL | PULLUP ;
```

```
1
2  NET "cnt<5>" LOC = "D11" | IOSTANDARD = LVTTL | SLEW = SLOW | DRIVE = 8 ;
3  NET "cnt<4>" LOC = "C11" | IOSTANDARD = LVTTL | SLEW = SLOW | DRIVE = 8 ;
4  NET "cnt<3>" LOC = "F11" | IOSTANDARD = LVTTL | SLEW = SLOW | DRIVE = 8 ;
5  NET "cnt<2>" LOC = "E11" | IOSTANDARD = LVTTL | SLEW = SLOW | DRIVE = 8 ;
6  NET "cnt<1>" LOC = "E12" | IOSTANDARD = LVTTL | SLEW = SLOW | DRIVE = 8 ;
7  NET "cnt<0>" LOC = "F12" | IOSTANDARD = LVTTL | SLEW = SLOW | DRIVE = 8 ;
8
9  NET "clk" LOC = "C9" | IOSTANDARD = LVCMOS33 ;
10
11 NET "reset" LOC = "N17" | IOSTANDARD = LVTTL | PULLUP ;
```

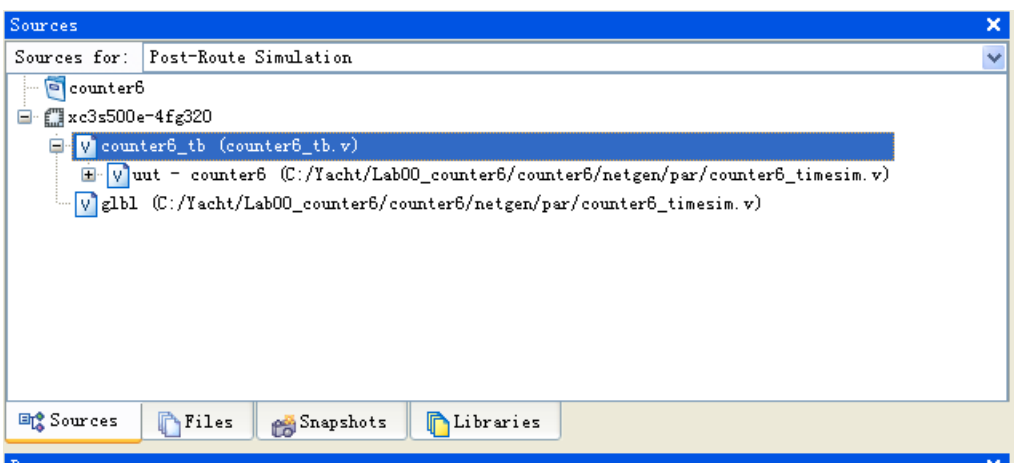
（2）进行实现，执行 Implement Design



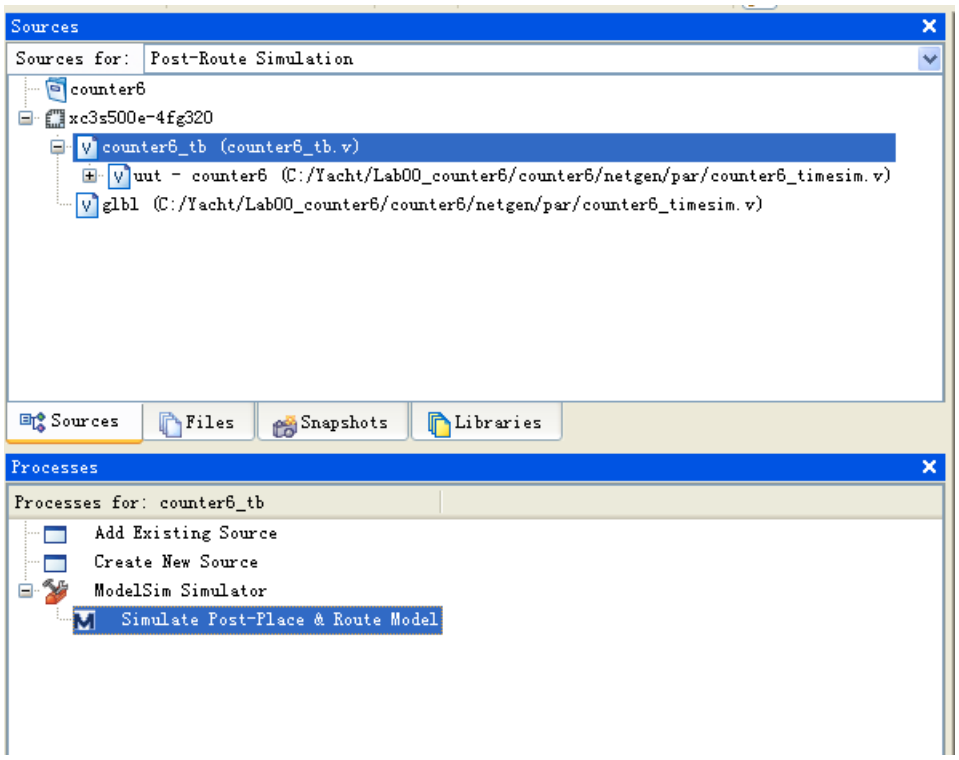
设计实现（Implement）完成后，执行 Generate Post-Place & Route Simulation Model，生成布局布线后仿真模型。



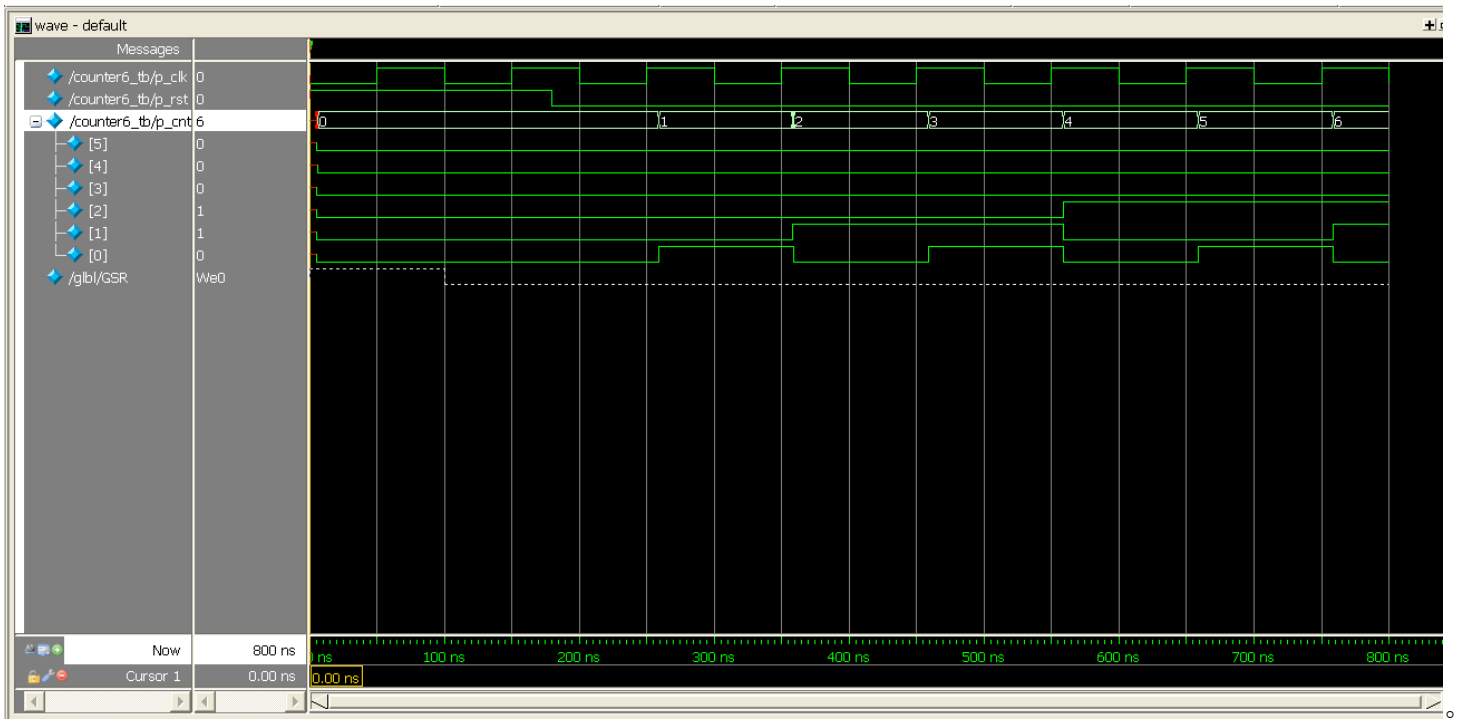
（3）在“Sources for”窗口，切换到 Post-Route Simulation：



（4）然后，点击选中 counter6_tb.v 文件，运行“Processes”窗口中的 Simulate Post-Place&Route Model，启动 ModelSim：



（5）在 ModelSim 中观察仿真波形，可以看到计数器的输出 cnt 有明显的延时：



至此，一个完整的基于 Verilog 的 FPGA 模块设计、行为仿真和实现之后的（布局布线）后仿真过程全部完成。