

## 实验零 Quartus 使用初步

2020 年秋季学期

*The Hitchhiker's Guide to the Galaxy itself has outsold the Encyclopedia Galactica because it is slightly cheaper, and because it has the words 'DON'T PANIC' in large, friendly letters on the cover.*

— “The Hitchhiker's Guide to the Galaxy”, Douglas Adams

在本次实验中，我们以 Quartus17.1 为例，介绍如何利用 FPGA 开发平台 DE10-Standard 进行数字逻辑设计。课后可以参考相关资料，在自己的电脑上安装 Lite 版的开发软件套装。

## 0.1 设计流程

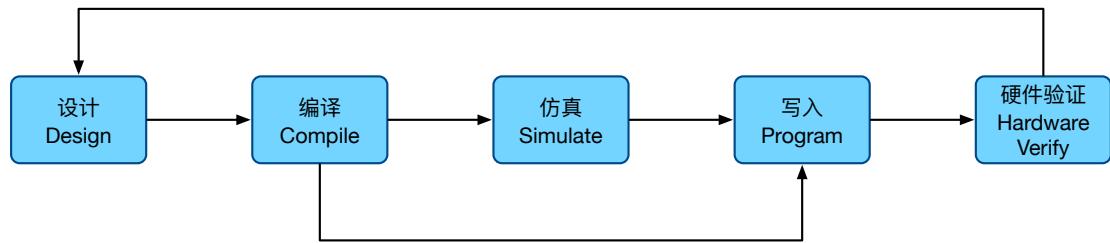


图 0-1: FPGA 设计流程

FPGA 开发流程如图 0-1 所示。其基本流程是从设计、编译、仿真、写入器件到硬件验证的一个过程。实现过程中如果出现问题，可能会反复修改设计或者代码。

FPGA 的编译一般包含多个步骤，具体的命名在不同教材和工具中略有不同。其中基本的编译完成语法分析和逻辑实现，在完成基本编译后，可以进行前端的功能仿真，对设计的逻辑功能进行验证。在仿真没有问题后，可以进行综合和布局布线。这时基本的逻辑功能将映射到实际物理器件上，生成网表和 FPGA 二进制烧写文件。随后可以进行后端仿真，针对物理器件的实际延时和物理性能，分析最终器件是否能满足设计要求。最后，可以将 FPGA 二进制文件写入 FPGA，在实际物理硬件上进行验证。

## 0.2 设计目标

在本实验中，我们将一步一步设计一个简单的 FPGA 应用：双控开关。该应用的目标是让两个拨动开关能够同时控制一盏灯。例如，在卧室中可以通过门口或床头的开关来开关卧室的灯，这样开关灯时就不用跑来跑去。

假设两个开关分别用 A 和 B 来表示，灯用 F 来表示。双控开关可用一个简单的表达式：

$$F = A \oplus B = \bar{A}B + A\bar{B} \quad (0-1)$$

来实现。下面我们就一步一步地来在 DE10-Standard 平台上实现这个功能。

## 0.3 建立工程

双击桌面上的 Quartus 图标（）打开 Quartus，出现如图 0-2 界面。

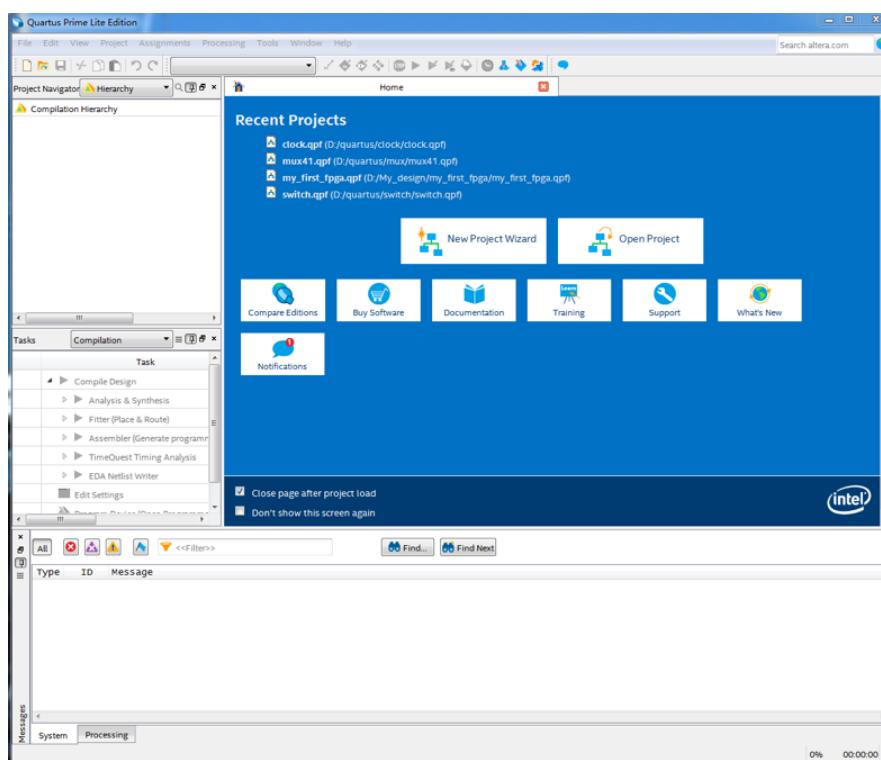


图 0-2: Quartus II 主界面

在 Quartus 的工作界面中点击 **New Project Wizard** 或者点击 **File→New Project Wizard**，弹出新建工程介绍页面（图 0-3）。

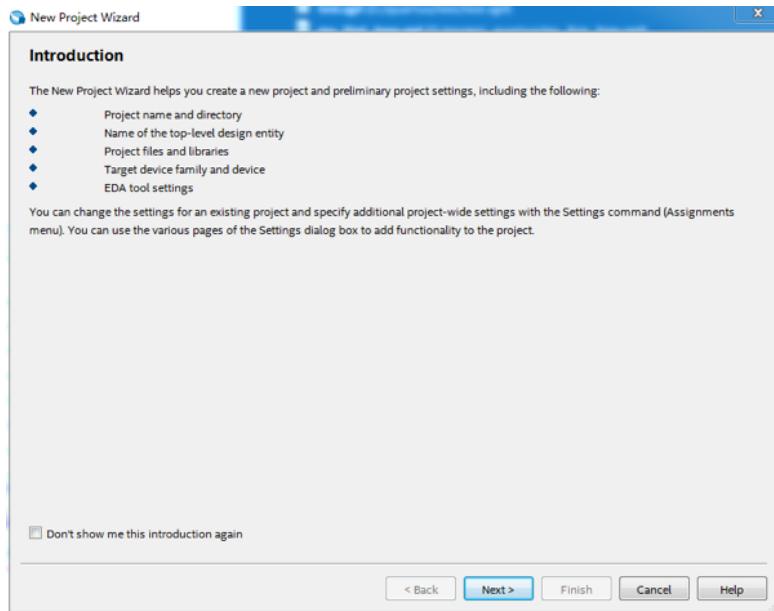


图 0-3: 新建工程界面

点击[Next]。在弹出的界面中填写以下信息：

- a. 这个工程的工作路径。请你自己新建一个路径专门用于保存本学期的所有工程，比如：D:\My\_design。再为本次工程新建一个文件夹，用于保存编译过程中的所有文件：D:\My\_design\my\_first\_fpga。
- b. 为本次工程取一个名字，比如：my\_first\_fpga。
- c. 为本次工程的顶层实体取一个名字，这个名字必须和将要输入的设计文件中的顶层实体的名字一样。比如：my\_first\_fpga。

 请注意文件名、工程名和路径中都不能含有空格，并建议不要包含中文字符。工程名和文件名请用字母开头，是字母、数字和下划线的组合。

如图 0-4 所示，点击 **Next**。如路径中的文件不存在，点击 **OK** 新建文件。

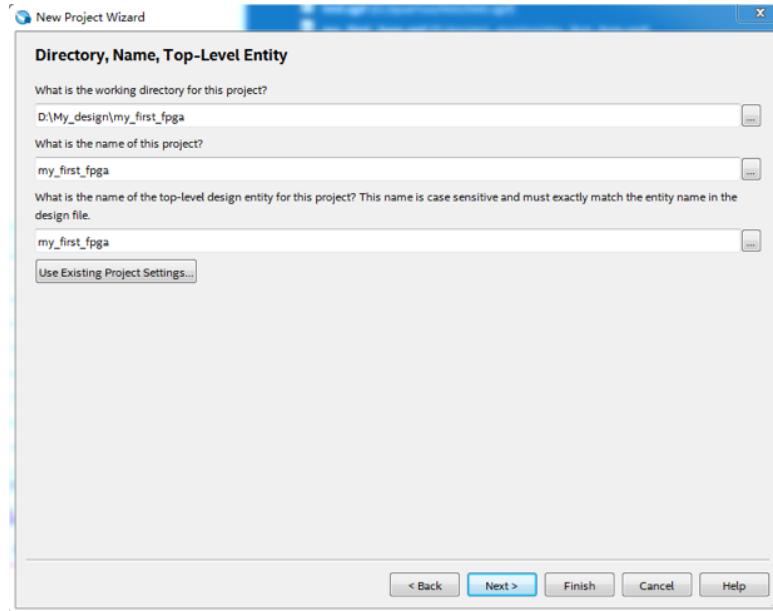


图 0-4: 输入文件名

如图 0-5 所示默认建立一个空工程，点击 **Next**。

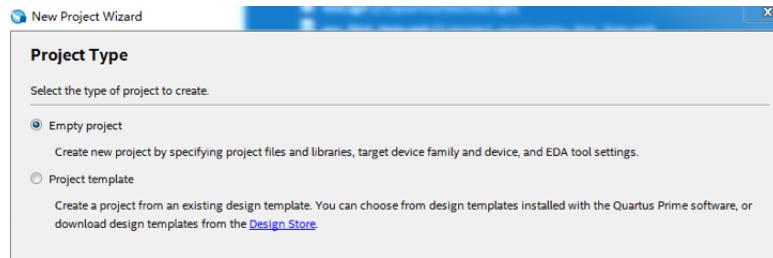


图 0-5: 建立空工程

图 0-6 中可以选择已经设计好的文件加入本工程，这里没有，点击 **Next** 继续。

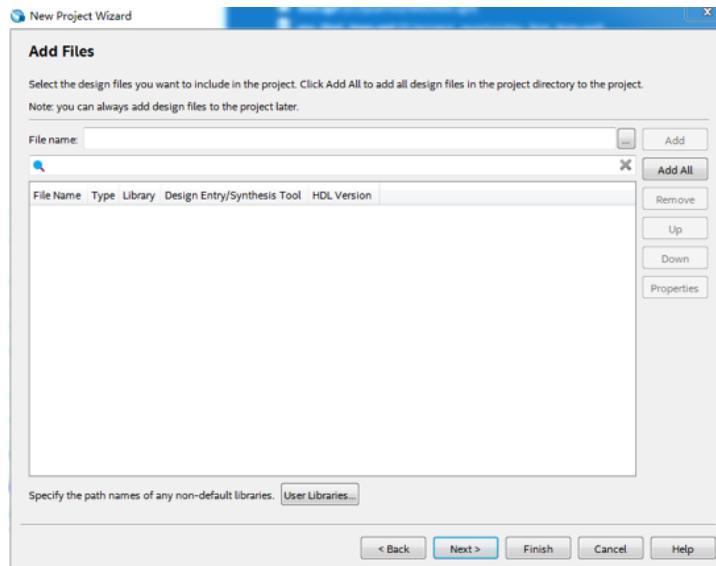


图 0-6: 添加源文件

进入选择器件窗口图 0-7。根据开发平台上的 FPGA 型号 **5CSXFC6D6F31C6** 选择。

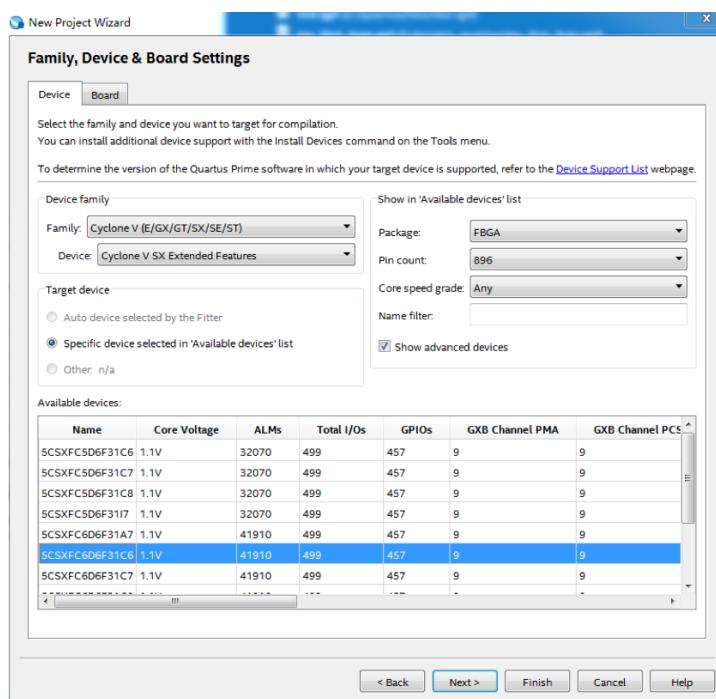


图 0-7: 选择器件

点击[Next]进入EDA工具设置，这里对仿真工具进行设置，如图在[Simulation]选项的下拉框中选择 ModelSim-Altera。选择 ModelSim-Altera 之后，语言就默认为：Verilog HDL，如图 0-8。

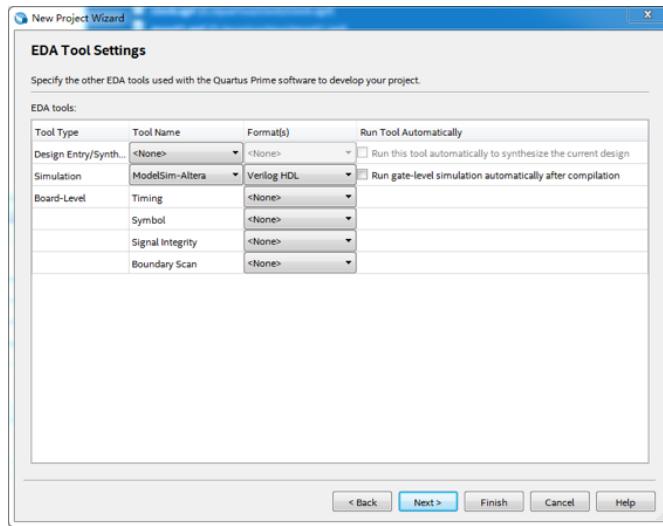


图 0-8: 选择仿真工具

点击[Next]，弹出新建工程总结框，点击[Finish]完成新建工程，等待 Quartus 打开工程，进入如图 0-9 界面。

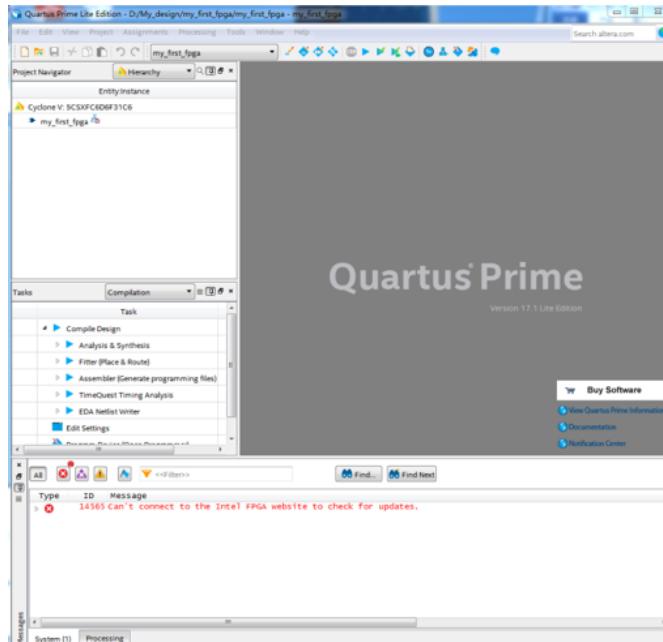


图 0-9: 工程主界面

## 0.4 设计并编译

在建立完一个空工程后，我们需要添加设计文件实现具体功能。首先要完成工程的顶层模块的设计，顶层模块的输入输出将直接对应 FPGA 的引脚。

### 0.4.1 添加设计文件

要添加设计文件，选择 **File→New** 在弹出的对话框（图 0-10）中选择 Verilog HDL File，点击 **OK**。

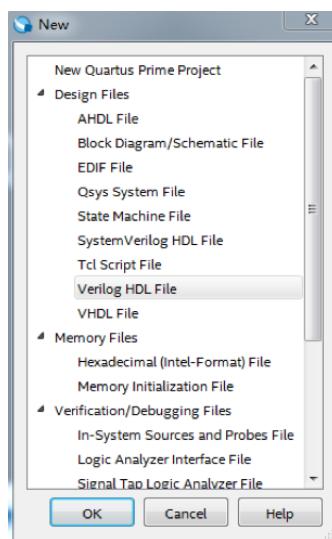


图 0-10：添加设计文件

### 0.4.2 设计

在弹出的工作区（图 0-11）输入表 0-1 中的顶层设计文件，请参考教科书上第五章的 Verilog 语法理解设计文件的含义。

这里顶层实体名应与新建工程时取的名称 my\_first\_fpga 完全一致。

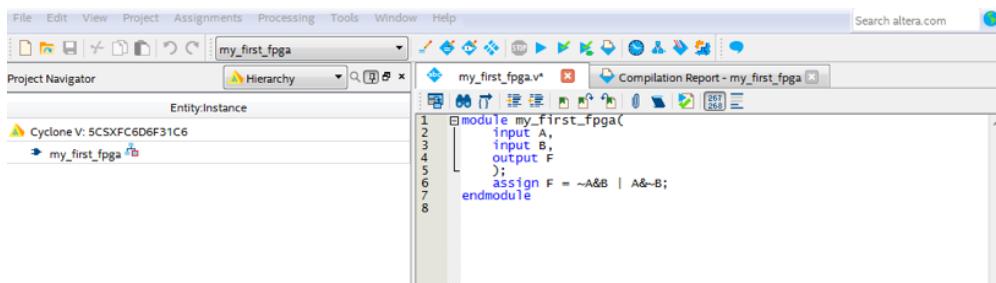


图 0-11: 输入设计文件

表 0-1: 双控开关设计文件

```

1 module my_first_fpga(
2     input A,
3     input B,
4     output F
5 );
6     assign F = ~A&B | A&~B;
7
8 endmodule

```

保存，文件名默认为 my\_first\_fpga。点击[Hierarchy]的下拉框，选择[Files]，如图 0-12。

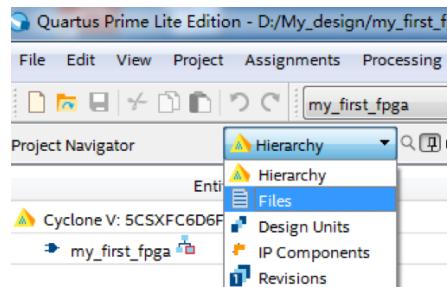


图 0-12: 选择文件视图

之后会在项目导航栏显示刚刚保存的 my\_first\_fpga.v 文件。

### 0.4.3 分析与综合

点击 Processing→Start→Start Analysis & Synthesis 或者点击图标 对工程进行分析和综合，检查设计中可能出现的语法错误。如果设计代码中有错误，会在 Message 框中给出提示，如图 0-13 所示。

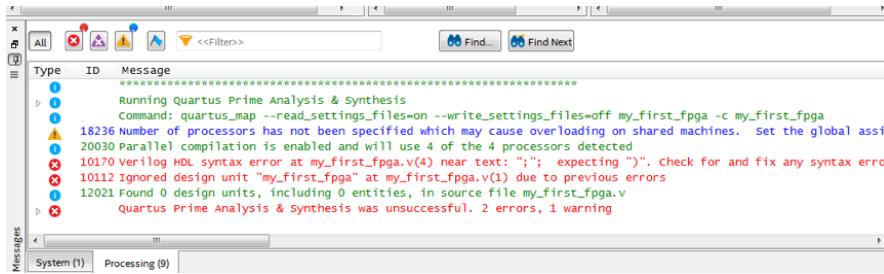


图 0-13: 错误提示

请根据提示内容修改错误，直至分析和综合通过，如图 0-14 所示。

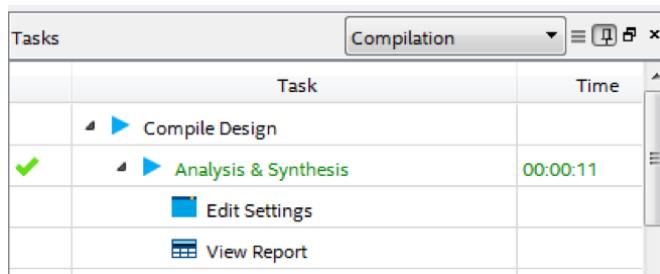


图 0-14: 分析与综合结果正确

## 0.5 功能仿真

功能仿真时验证代码功能实现的基本方式。Quartus 提供了基于 ModelSim 的功能仿真。开发者可以编写测试文件（test bench）来调用实现的模块，对输入输出进行分析，从而判断功能实现的正确性。

点击 **Assignments→Settings** 进行仿真设置，如图 0-15 所示。

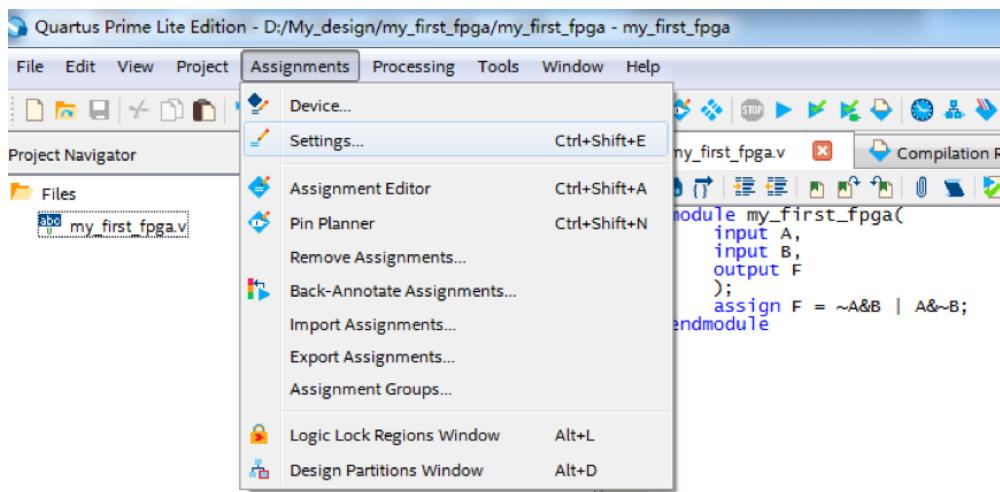


图 0-15：仿真设置入口

### 0.5.1 仿真设置

如图 0-16 所示，点击红框 1 处的 Simulation 出现右边的界面。红框 2 是设置所用的仿真工具名称，红框 3 是设置工程编写用的 HDL 和时间刻度。红框 4 是本地仿真链接设置，这里先选 none，在工程下生成仿真 simulation 文件夹。

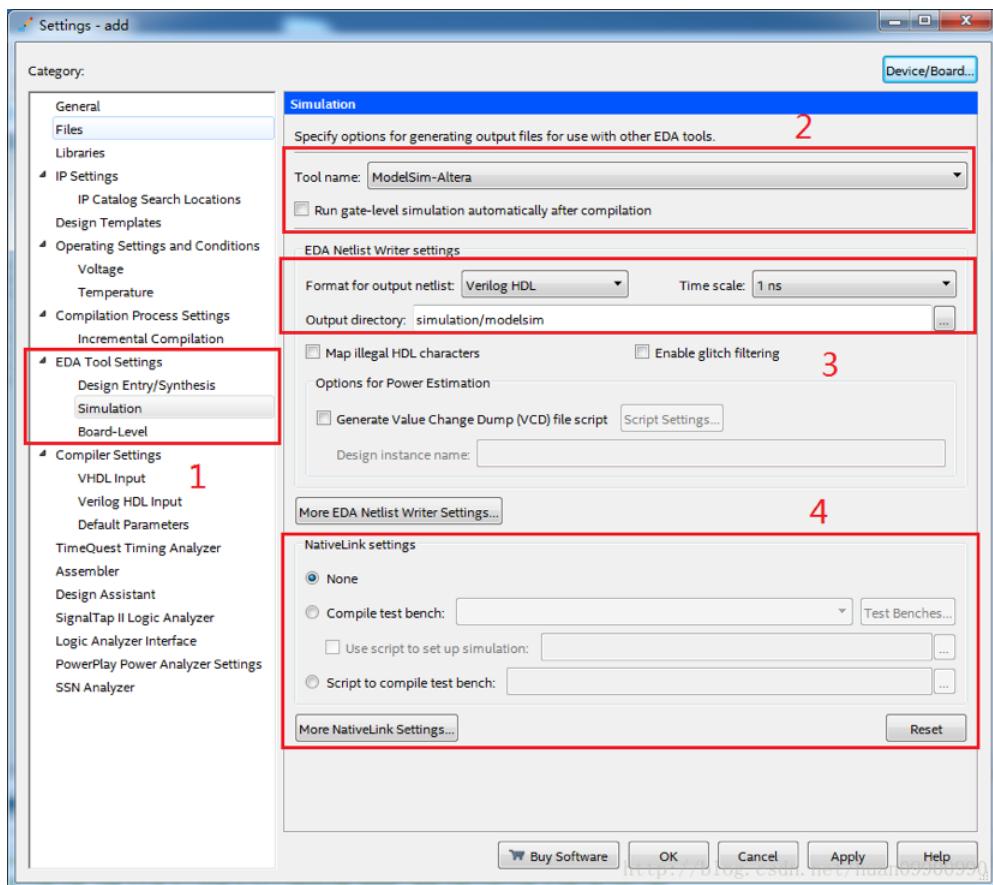


图 0-16: 仿真设置

当 ModelSim 的路径没有设置时，需要在 **Tools→Options** 中设置 ModelSim 的可执行文件路径。在弹出的 options 界面中选择 **General→EDA Tool Options**，出现如图 0-17 界面，设置 ModelSim-Altera 为本机 ModelSim 的安装路径。如 D:\intelFPGA\_lite\17.1\modelsim\_ase\win32aloem

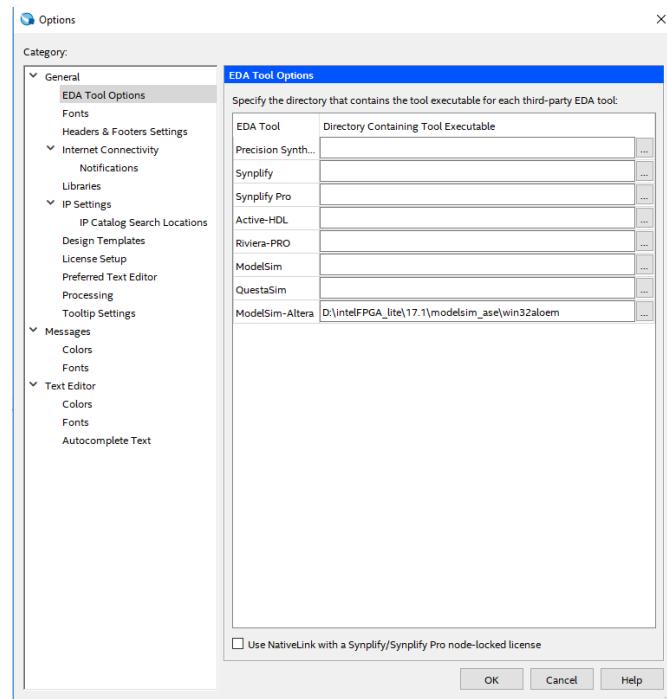


图 0-17: ModelSim 路径设置

### 0.5.2 编写测试文件

首先，可以点击 **Processing→Start→Start Test bench Template Writer** 自动生成仿真文件模板。

然后打开仿真文件模板。选择 **File→Open**，测试文件 Test bench 的默认路径是：项目目录 \simulation\modelsim\ 项目名.vt

比如本工程的测试文件是：

D:\My\_design\my\_first\_fpga\simulation\modelsim\my\_first\_fpga.vt

 请注意，Verilog HDL 测试文件的后缀名是“.vt”。

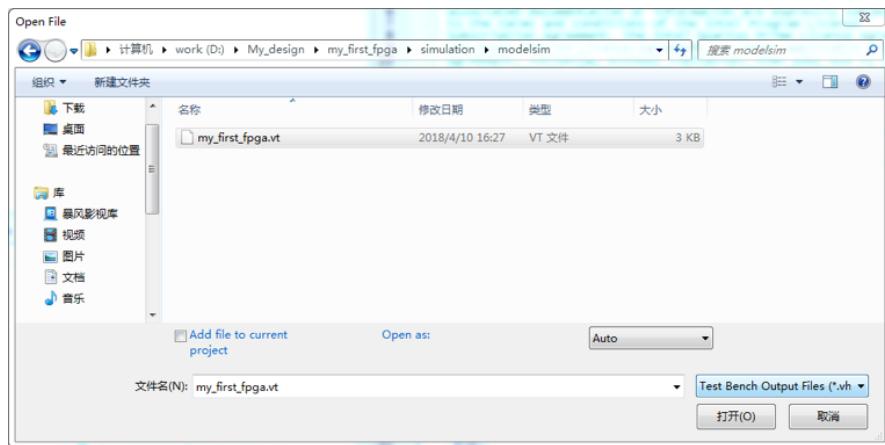


图 0-18: 打开测试文件

---

打开测试文件后，根据我们自己的要求，对其进行修改保存：

我们对测试代码的修改主要是按照我们自己的要求，在测试代码中增加激励。对输入 A 和 B 按测试需求进行设置，相关语法参考教科书中测试代码的内容。

表 0-2: 双控开关测试文件修改后结果

```

1 `timescale 10 ns/ 1 ps
2 module my_first_fgpa_vlg_tst();
3 // constants
4 // general purpose registers
5 //reg eachvec;
6 // test vector input registers
7 reg A;
8 reg B;
9 // wires
10 wire F;
11
12 // assign statements (if any)
13 my_fgpa il (
14 // port map - connection between master ports and signals/registers
15     .A(A),
16     .B(B),
17     .F(F)
18 );
19 initial
20 begin
21 // code that executes only once
22 // insert code here --> begin
23     A = 1'b0; B= 1'b0; #20;
24     A = 1'b0; B= 1'b1; #20;
25     A = 1'b1; B= 1'b0; #20;
26     A = 1'b1; B= 1'b1; #20;
27     A = 1'b0; B= 1'b0; #20;
28 // --> end
29 //$display("Running testbench");
30 end
31 //always
32 // optional sensitivity list
33 // @(event1 or event2 or .... eventn)
34 //begin
35 // code executes for every event on sensitivity list
36 // insert code here --> begin
37
38 //@eachvec;
39 // --> end
40 //end
41 endmodule

```

```
27 `timescale 10 ns/ 1 ps
28 module my_first_fpga_vlg_tst();
29 // constants
30 // general purpose registers
31 //reg eachvec;
32 // test vector input registers
33 reg A;
34 reg B;
35 //wires
36 wire F;
37
38 // assign statements (if any)
39 my_first_fpga i1 (
40 // port map - connection between master ports and signals/registers
41 .A(A),
42 .B(B),
43 .F(F)
44 );
45 initial
46 begin
47 // code that executes only once
48 // insert code here --> begin
49 A = 1'b0; B = 1'b0; #20;
50 A = 1'b0; B = 1'b1; #20;
51 A = 1'b1; B = 1'b0; #20;
52 A = 1'b1; B = 1'b1; #20;
53 A = 1'b0; B = 1'b0; #20;
54
55 // --> end
56 //$display("Running testbench");
57 end
58 //always
59 // optional sensitivity list
60 //@(event1 or event2 or .... eventn)
61 //begin
62 // code executes for every event on sensitivity list
63 // insert code here --> begin
64
65 //@eachvec;
66 // --> end
67 //end
68 endmodule
```

图 0-19: 测试代码

### 0.5.3 添加测试文件

打开 **Assignments→Settings→Simulation**，进入图 0-20 的界面，在红框里选择 **Compile test bench**。

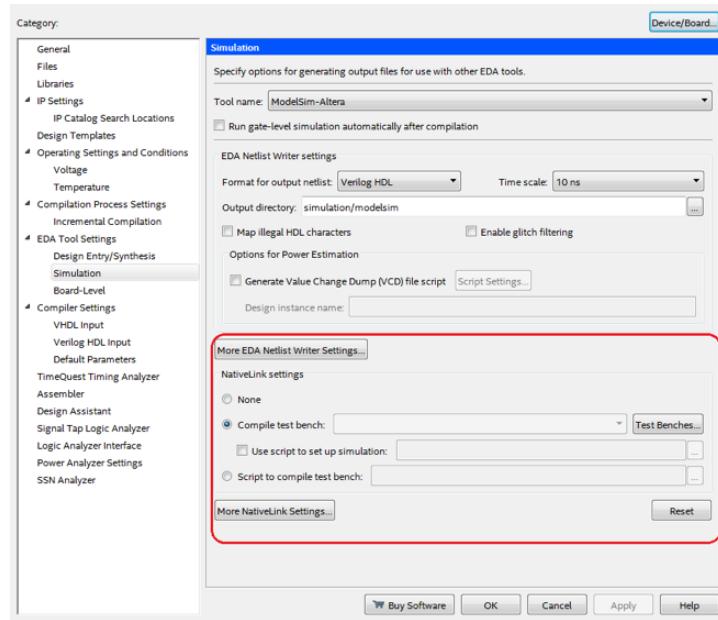


图 0-20: 添加测试代码 I

点击 **Test Benches...** 按钮，进入图 0-21 的界面

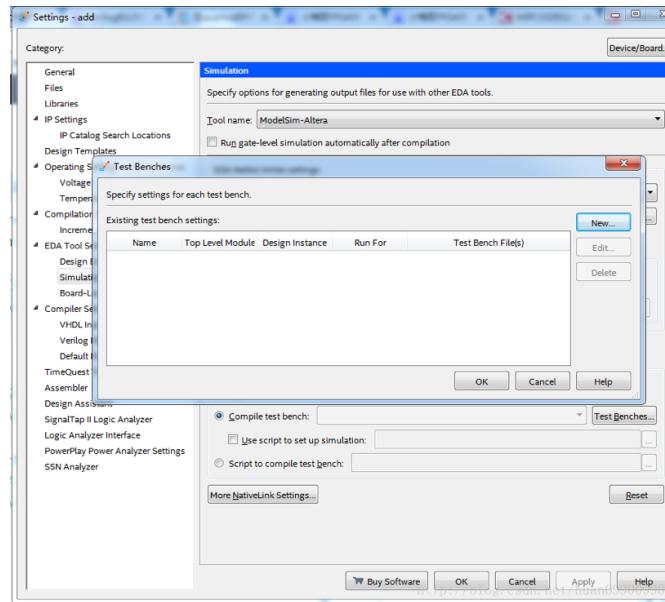


图 0-21: 添加测试代码 II

点击 **New** 按钮，进入下面的窗口，分别设置 **Test bench name**（测试文件的名称）：my\_first\_fpga，和 **Top level module in test bench**（测试文件顶层实体的名称）：my\_first\_fpga\_vlg\_tst，填写好如图 0-22 所示：

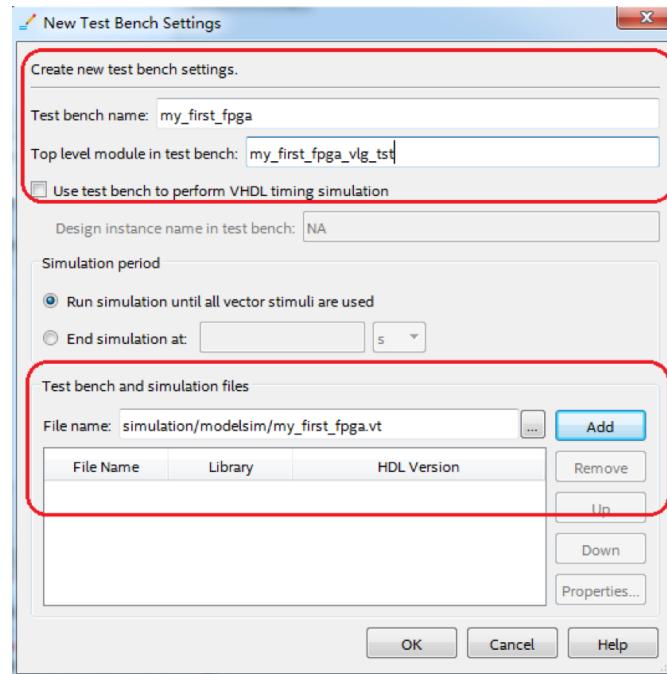


图 0-22: 添加测试代码 III

再点击 **Add**, 将先前保存的 test bench 加入 (图 0-23)

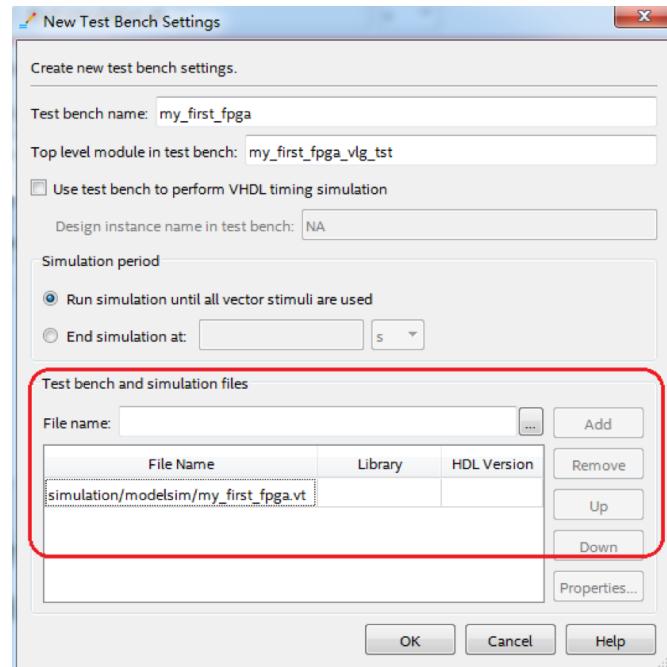


图 0-23: 添加测试代码 IV

点击**OK**。

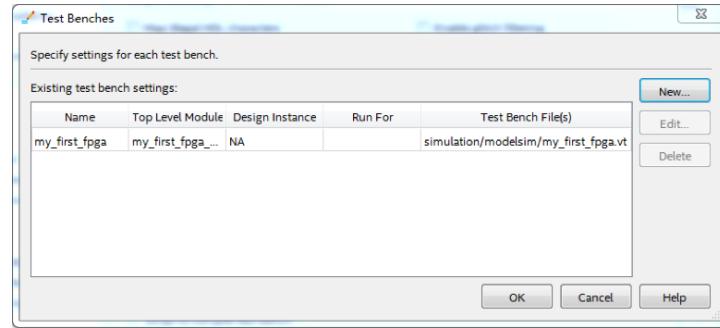


图 0-24: 添加测试代码 V

再点击**OK**, test bench 已经被加入了。

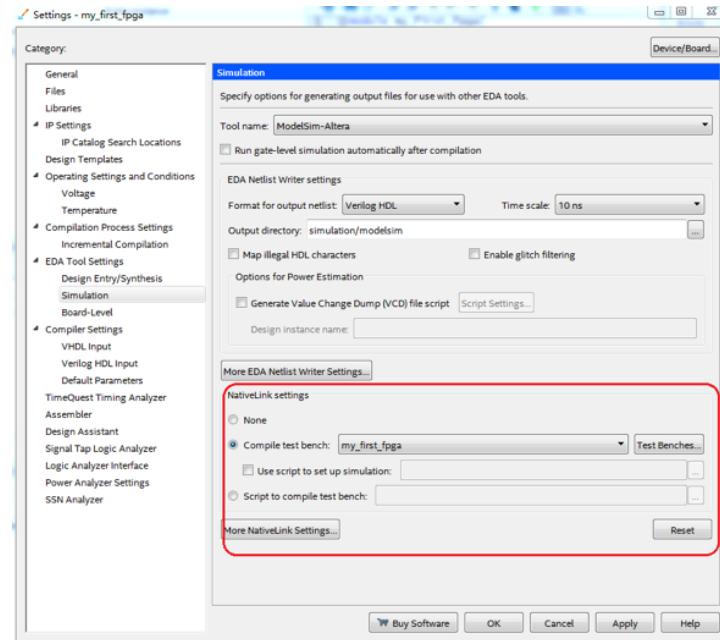


图 0-25: 添加测试代码 VI

点击**Apply**, 再点击**OK**, 返回 Quartus 主界面。

### 0.5.4 运行仿真

配置好仿真文件后，依次选择 **Tools→Run Simulation Tool→RTL Simulation** 进入 RTL 仿真

等待弹出 ModelSim 仿真窗口。如出现错误，请检查 ModelSim 是否安装，并设置 ModelSim 的路径。

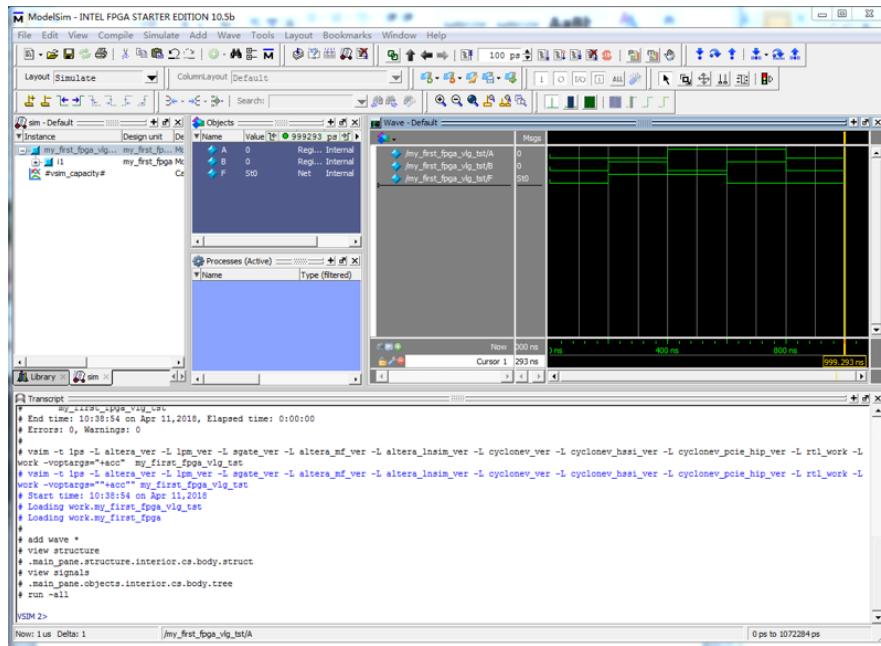


图 0-26: ModelSim 界面

利用缩放工具 ，对仿真结果窗口进行调整，在合适的窗口中查看仿真结果是否满足设计需要。

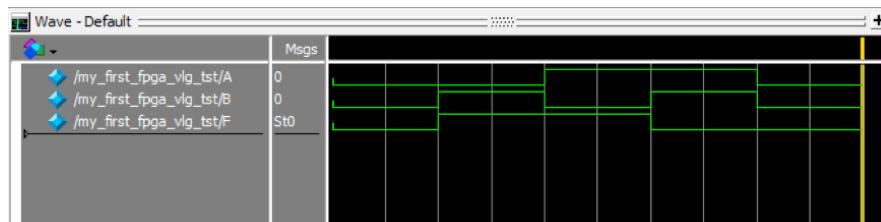


图 0-27: ModelSim 仿真波形

## 0.6 细化及生成二进制文件

分析与综合生成的只是电路的功能性描述，通过功能仿真后验证了电路在功能上的正确性。但是我们需要将电路映射到实际的物理器件上，并生成二进制文件，这样才能够将二进制写入 FPGA 并在实际物理器件上运行。

### 0.6.1 引脚分配

仿真正确后，则对工程进行分析细化并准备配置引脚。

点击 **Processing→Start→Start Analysis & Elaboration** 或者点击图标 ，分析细化后的结果如图 0-28。

	Task	Time
	▶ Compile Design	
	▶ Analysis & Synthesis	
	▶ Edit Settings	
	View Report	
✓	▶ Analysis & Elaboration	00:00:13
	▶ Partition Merge	
	▶ Netlist Viewers	
	▶ Design Assistant (FPGA)	

图 0-28: 分析细化正确结果

分析完成之后就可以进行引脚分配了。

点击 **Assignments→Pin Planner** 或者  图标，弹出引脚分配界面（图 0-29）。

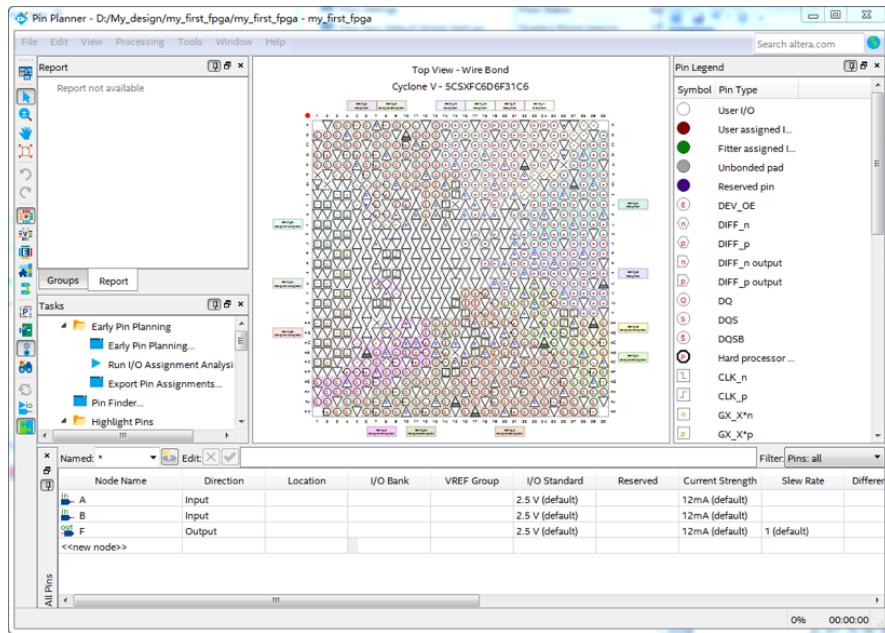


图 0-29: 引脚分配界面

此时顶层实体的输入输出引脚已经出现在引脚分配表中了，在这个工程中，一共需要用到两个输入端，我们使用开关 SW0 和 SW1；和一个输出端，我们选择发光二极管 LEDR0 作为输出，根据 DE10-Standard 的说明文件 DE10-Standard\_User\_manual.pdf：

Table 3-6 Pin Assignment of Slide Switches

Signal Name	FPGA Pin No.	Description	I/O Standard
SW[0]	PIN_AB30	Slide Switch[0]	Depend on JP3
SW[1]	PIN_Y27	Slide Switch[1]	Depend on JP3

(a) SW 引脚

Table 3-8 Pin Assignment of LEDs

Signal Name	FPGA Pin No.	Description	I/O Standard
LEDR[0]	PIN_AA24	LED [0]	3.3V

(b) LED 引脚

图 0-30: DE10-Standard 引脚分配表

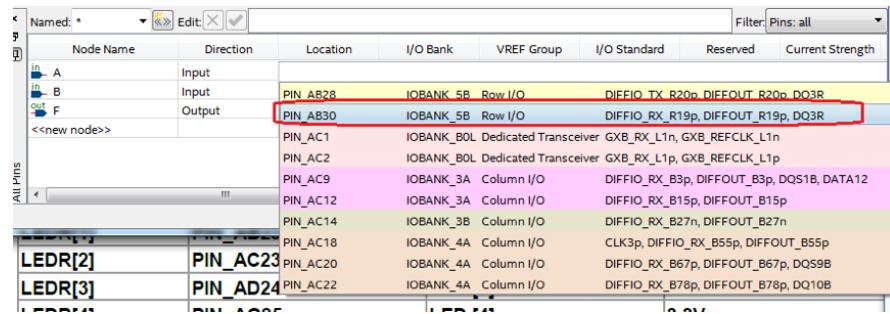
在引脚配置界面中，双击各输入输出引脚后的 Location 栏根据下拉单，



Node Name	Direction	Location	I/O Bank	VREF Group	I/O Standard	Reserved	Current Strength
In A	Input				2.5 V (default)		12mA (default)
In B	Input				2.5 V (default)		12mA (default)
out F	Output				2.5 V (default)		12mA (default)

图 0-31: 引脚分配位置

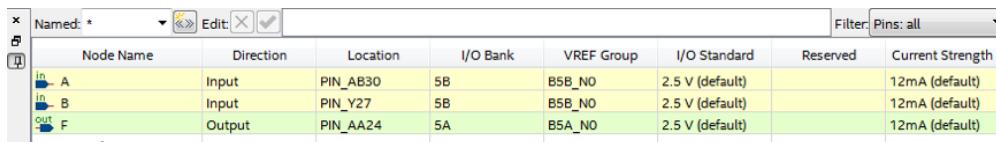
选择相应引脚，A 对应 SW0，选择 PIN\_AB30；也可以直接输入 AB30 指定该引脚。



Node Name	Direction	Location	I/O Bank	VREF Group	I/O Standard	Reserved	Current Strength
In A	Input	PIN_AB28	IOBANK_5B_Row I/O		DIFFIO_RX_R20p, DIFFOUT_R20p, DQ3R		
In B	Input	PIN_AB30	IOBANK_5B_Row I/O		DIFFIO_RX_R19p, DIFFOUT_R19p, DQ3R		
out F	Output	PIN_AC1	IOBANK_B0L Dedicated Transceiver	GXB_RX_L1n, GXB_REFCLK_L1n			
		PIN_AC2	IOBANK_B0L Dedicated Transceiver	GXB_RX_L1p, GXB_REFCLK_L1p			
		PIN_AC9	IOBANK_3A Column I/O		DIFFIO_RX_B3p, DIFFOUT_B3p, DQS1B, DATA12		
		PIN_AC12	IOBANK_3A Column I/O		DIFFIO_RX_B15p, DIFFOUT_B15p		
		PIN_AC14	IOBANK_3B Column I/O		DIFFIO_RX_B27n, DIFFOUT_B27n		
		PIN_AC18	IOBANK_4A Column I/O		CLK3p, DIFFIO_RX_B55p, DIFFOUT_B55p		
		PIN_AC20	IOBANK_4A Column I/O		DIFFIO_RX_B67p, DIFFOUT_B67p, DQS9B		
LEDR[2]	PIN_AC23	PIN_AC22	IOBANK_4A Column I/O		DIFFIO_RX_B78p, DIFFOUT_B78p, DQ10B		
LEDR[3]	PIN_AA24						

图 0-32: 引脚选择

B 对应 SW1，选择 PIN\_Y27；F 对应 LEDR0，选择 PIN\_AA24.



Node Name	Direction	Location	I/O Bank	VREF Group	I/O Standard	Reserved	Current Strength
In A	Input	PIN_AB30	5B	BSB_N0	2.5 V (default)		12mA (default)
In B	Input	PIN_Y27	5B	BSB_N0	2.5 V (default)		12mA (default)
out F	Output	PIN_AA24	5A	BSA_N0	2.5 V (default)		12mA (default)

图 0-33: 引脚分配完成

### 0.6.2 生成二进制文件

点击 Processing→Start Compilation 或者点击图标 对工程进行编译，编译过程中，在 Message 框中会显示编译过程的各种信息，如果发现错误，也会显示出来，请根据提示改正错误，直至编译完成。

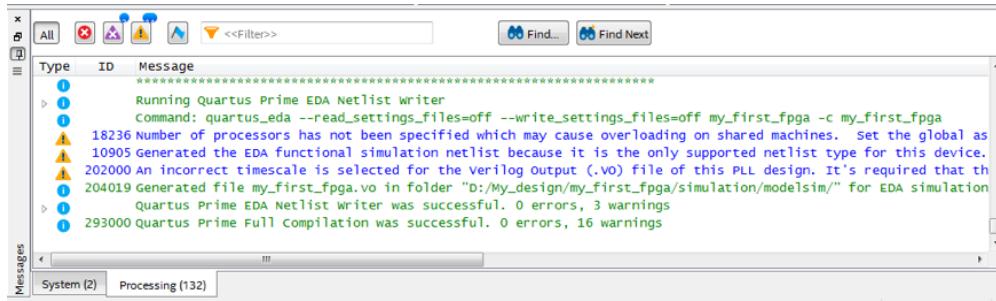


图 0-34: 编译正确结果

编译完成后会在工程目录下 output\_files 文件夹里生成二进制文件 my\_first\_fpga.sof。

## 0.7 写入硬件

编译完成之后就可以把电路下载到 FPGA 开发平台上验证其正确性了。在编程 FPGA 开发板前首先需要：

- 把 DE10-Standard 开发板接上电源，并通过 USB-Blaster II 接到电脑上。
- 打开开发板电源开关

### 0.7.1 连接硬件

点击 **Tools→Programmer**，或者点击  图标，打开 FPGA 编程界面

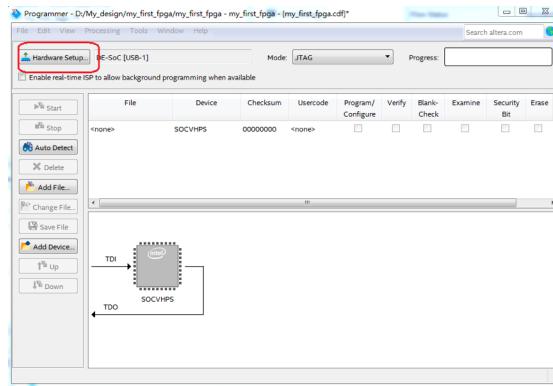


图 0-35: FPGA 编程界面

点击 **Hardware Setup...**，打开硬件设置界面，此时如果没有检测到 FPGA，点击 **Close**。

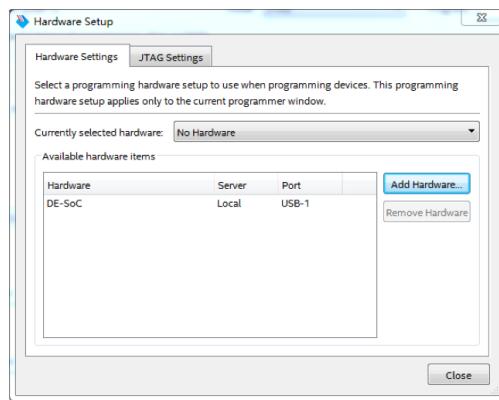


图 0-36: 硬件设置界面

 注意，有些时候看不到 FPGA 板是由于没有选择合适的硬件，可以如图 0-37 所示来选择 DE-SoC。特别是打开硬件设置后才开启开发板电源时，此情况经常出现。

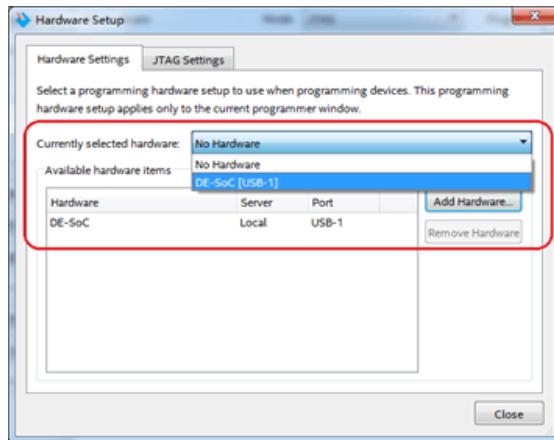


图 0-37: 部分情况下需要选择硬件

点击 **Auto Detect** 来查找所有连接在 JTAG 链上的器件，选择 [5CSXFC6D6](#)，点击 **OK**。

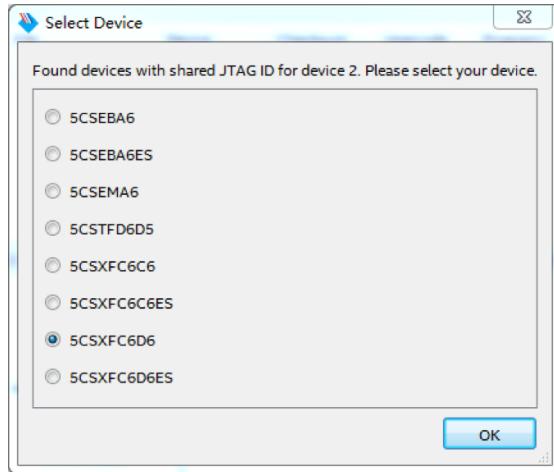


图 0-38: 器件选择界面

这时开发平台上的 HPS 和 FPGA 都会显示出来

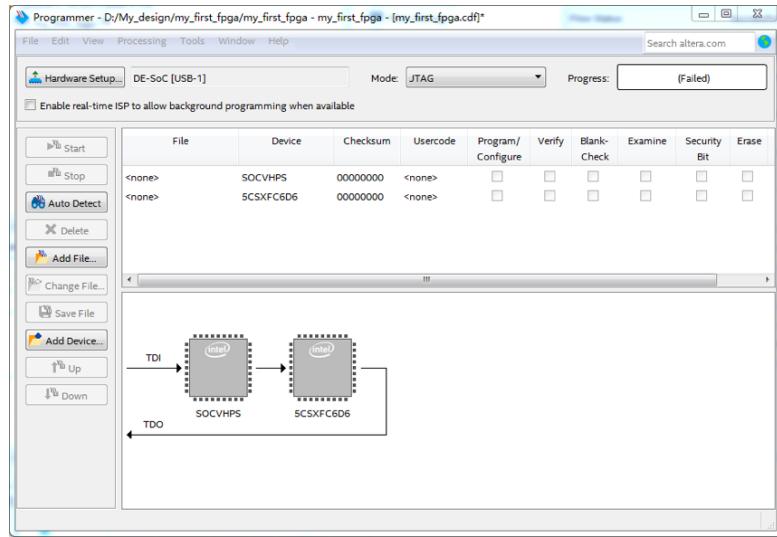


图 0-39: 器件选择界面

选择 FPGA 芯片 5CSXFC6D6，点击 **Change File**，选择编译产生的在工程目录下 output\_files 文件里的 my\_first\_fpga.sof 文件作为编程文件，点击 **Open**。

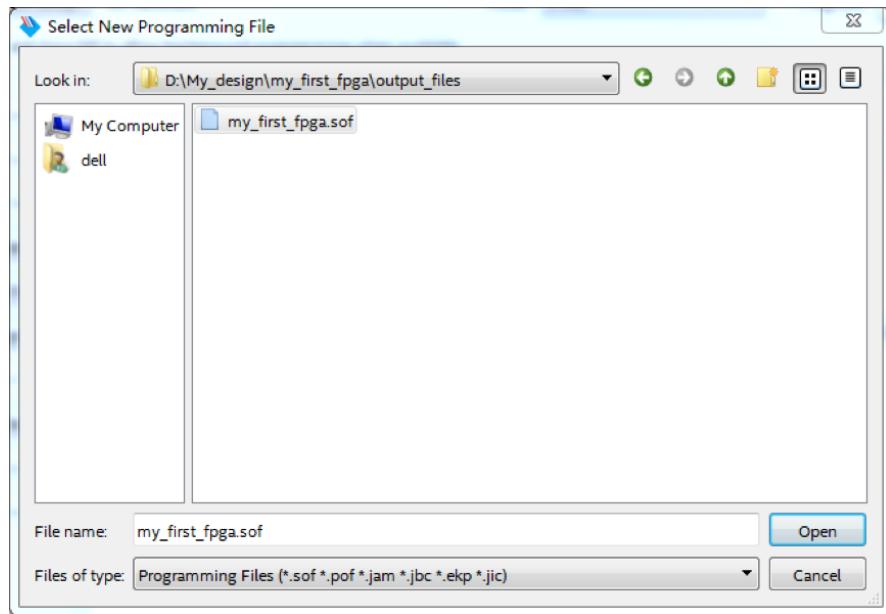


图 0-40: 选择二进制文件

勾选 Program/Configure 选项。

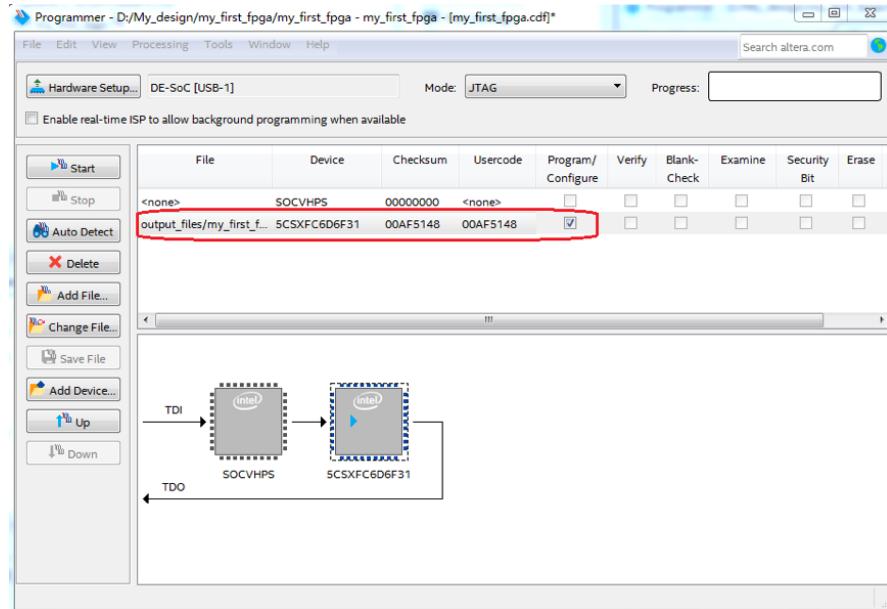


图 0-41: 勾选编程要求

点击 Start 开始把二进制.sof 文件编程到 FPGA 中。

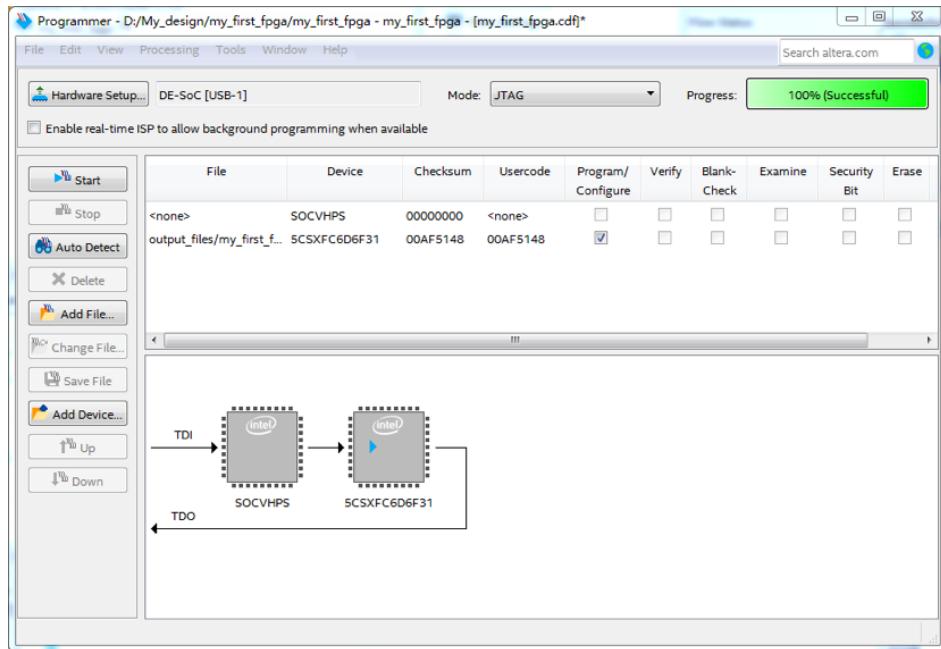


图 0-42: 编程成功界面

## 0.8 物理器件验证

现在可以拨动开关 SW0 和 SW1 看看发光二极管 LEDR0 的输出是否满足设计要求啦！

## 0.9 使用 System Builder

DE10-Standard 光盘中提供了 DE10-Standard 的 System Builder 工具，可以自动生成 DE10 开发板的工程及顶层文件，无需手动分配引脚。自己动手试试吧。

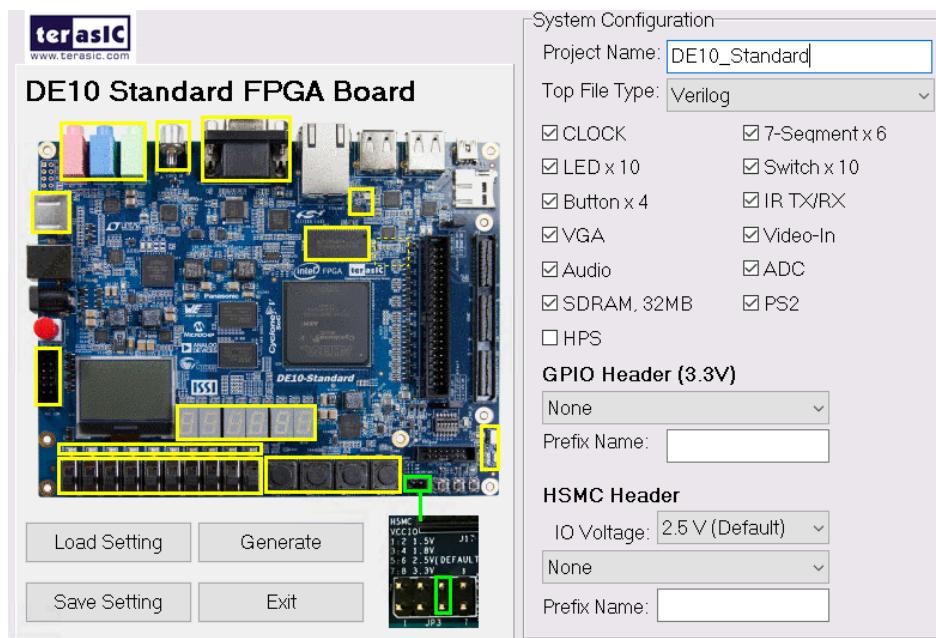


图 0-43: DE10-Standard System Builder 界面