

Listo para trabajar

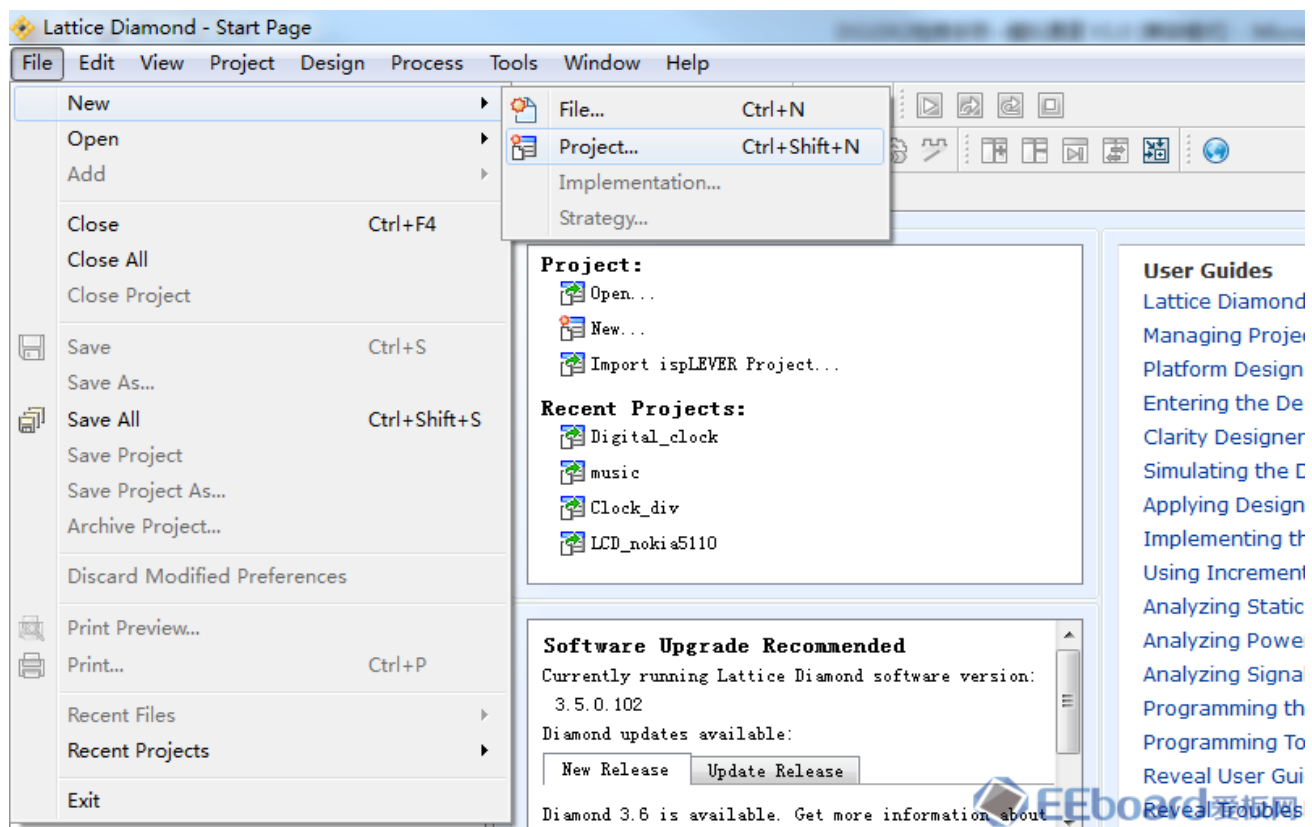
- Instalar el software Diamond
 - Consulte Instalación y configuración de Diamond para instalar Diamond. Si encuentra algún problema, primero puede consultar las Preguntas frecuentes sobre la instalación de Diamond .
- Prepare la placa de desarrollo FPGA STEP-MXO2-C y el cable de datos micro USB
 - Haga clic para obtener más descripción detallada de la placa de desarrollo STEP-MXO2-C

Comencemos nuestro primer proyecto FPGA.

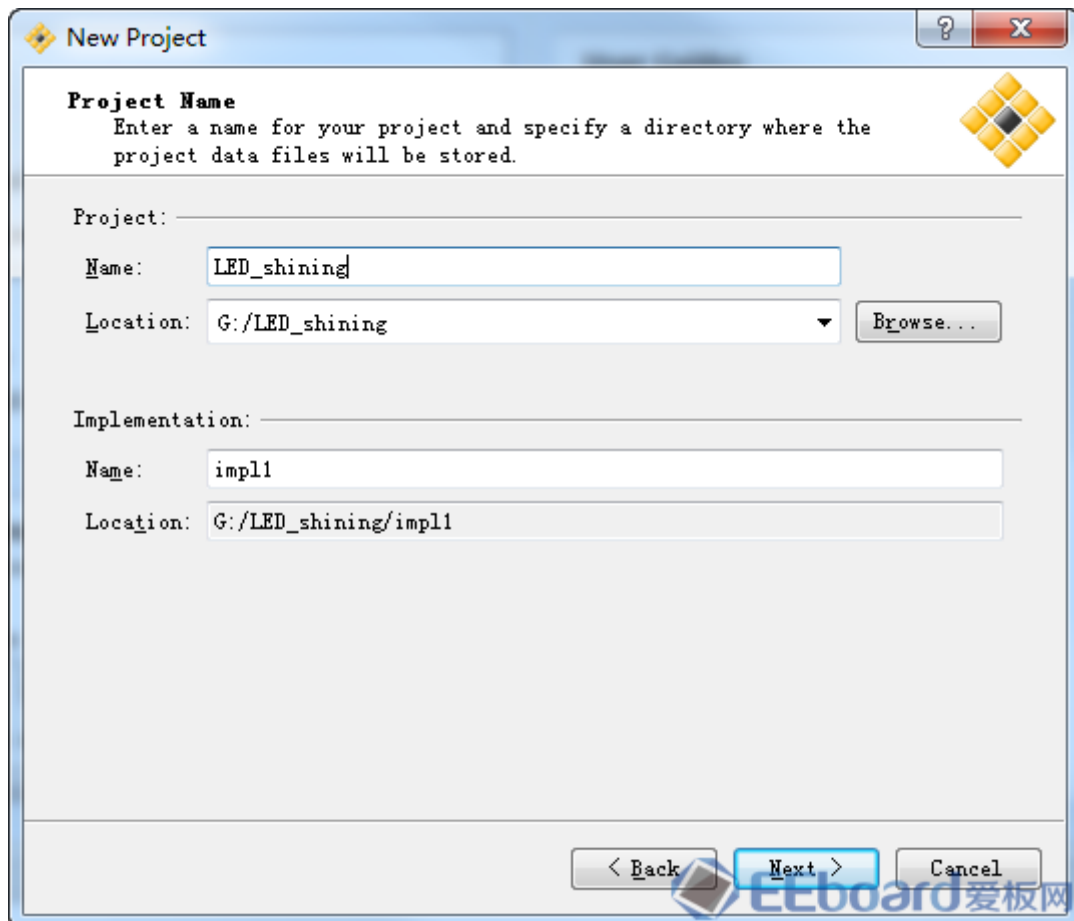
1 Ejecute la primera rutina

A continuación podemos iniciar el desarrollo de la lógica programable, tomamos el control del LED parpadeando alternativamente como ejemplo para completar nuestro primer programa:

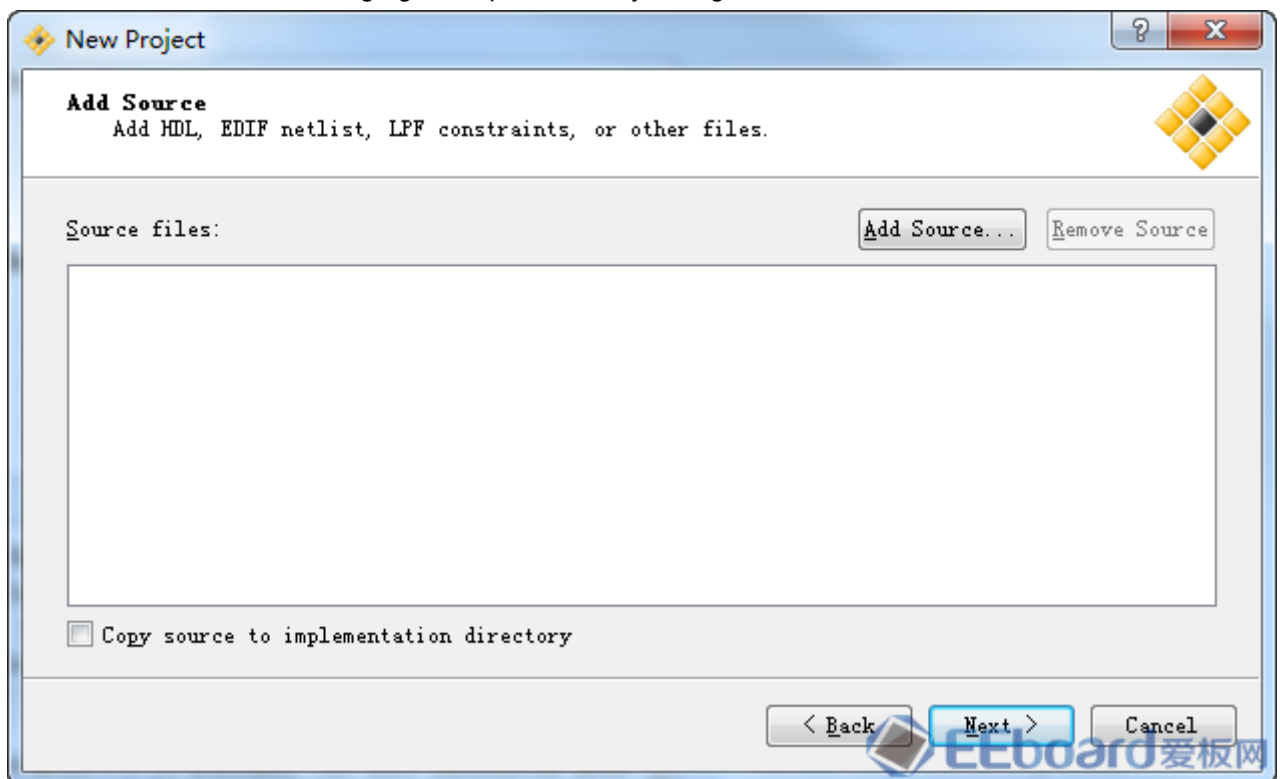
1. Haga doble clic para ejecutar el software Diamond, primero cree un nuevo proyecto: seleccione Archivo → Nuevo → Proyecto → Siguiente



2. Denominación del proyecto: nombraremos el nuevo LED del proyecto *brillando, el directorio del proyecto G: / LED* brillando, y luego haremos clic en Siguiente



3. Agregue archivos de diseño relacionados o archivos de restricción (si ya hay archivos de diseño y archivos de restricción, podemos optar por agregarlos al proyecto): aquí creamos un nuevo proyecto, no hay archivos relacionados, no es necesario agregar, simplemente vaya a Siguiente



4. Selección de dispositivo: Configure de acuerdo con el dispositivo de placa de desarrollo de paso FPGA LCMXO2-4000HC-4MG132C, Siguiente (El modelo del dispositivo debe confirmarse para ser correcto; de lo contrario, se informará un error durante la configuración del pin)

New Project ? X

Select Device
Specify a target device for the project.

Select Device:

Family: LatticeECP LatticeECP2 LatticeXP LatticeXP2 MachXO MachXO2 MachXO3L MachXO3LF Platform Manager Platform Manager 2

Device: LCMXO2-2000HC LCMXO2-2000HE LCMXO2-2000UHC LCMXO2-2000UHE LCMXO2-2000ZE LCMXO2-4000HC LCMXO2-4000HE LCMXO2-4000ZE LCMXO2-7000HC

Performance grade: () 4

Package type: CSBGA132

Operating conditions: Commercial

Part Names: LCMXO2-4000HC-4MG132C

Select ASC Device:

Part Names Numbers

L-ASC10-1SG48I 0

[Online Data Sheet for Device](#)

Device Information:

Voltage: 2.5V/3.3V

LUT: 4320

Registers: 4320

EBR Bits: 92.2K

EBR Blocks: 10

Dist RAM: 34560

DSP: -

PILL: 2

DLL: 0

PCS: -

APIO: -

PIO Cells: 280

PIO Pins: 105

Max Programmable IOs: 104

VMON Pins: 0

IMON Pins: 0

TMON Pins: 0

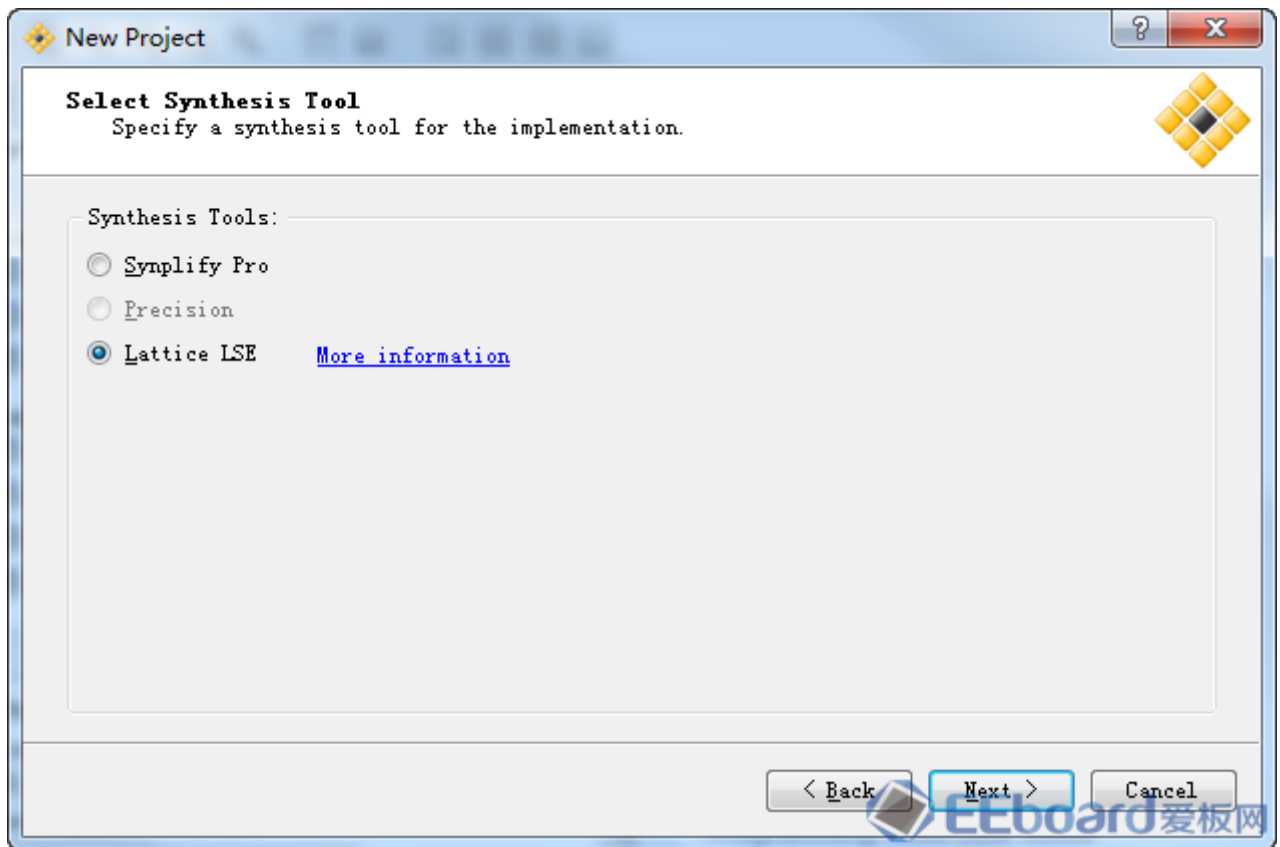
Trim/Mar Pins: 0

HVOUT Pins: 0

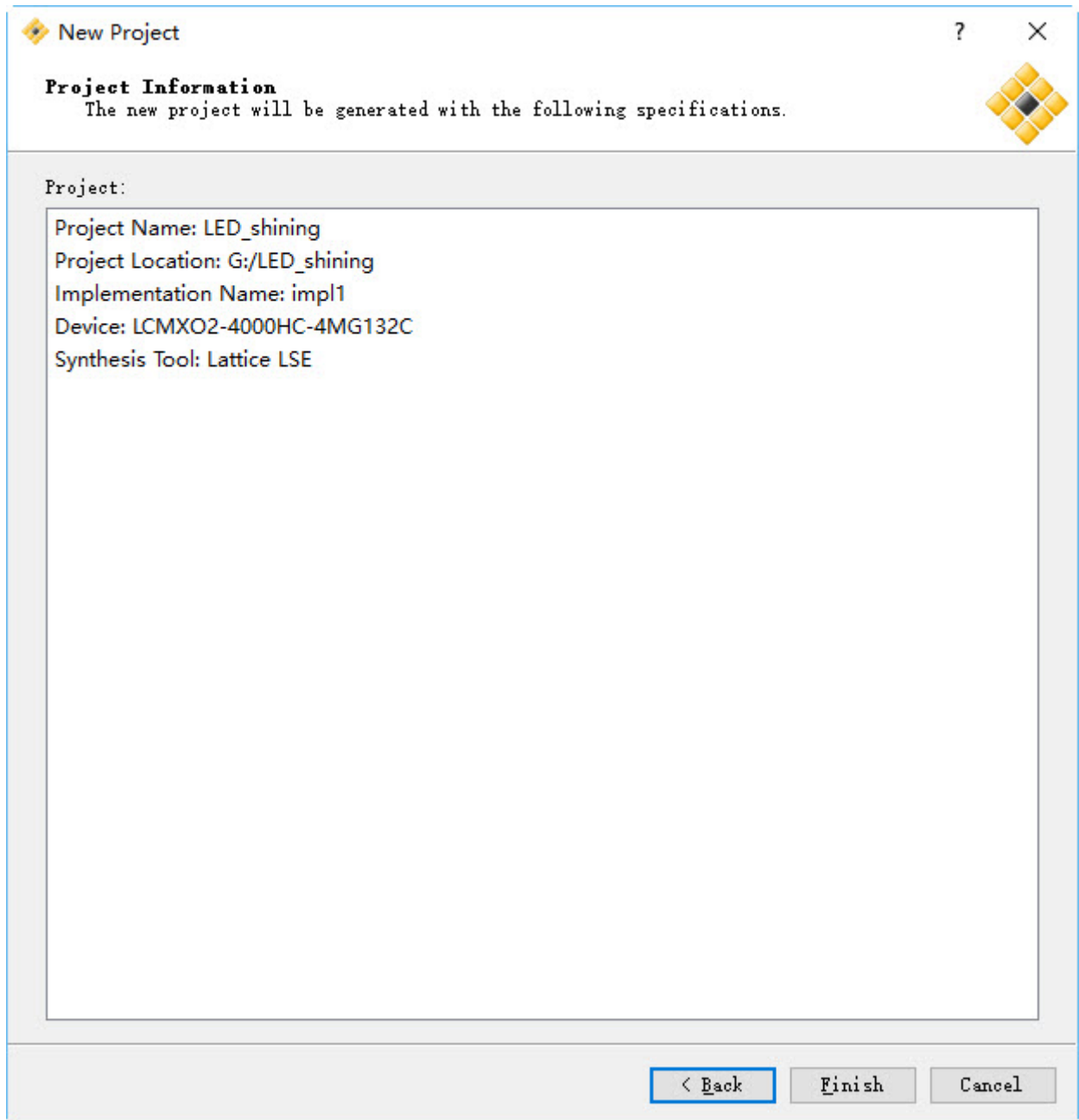
GPIO: 0

< Back Next > Cancel

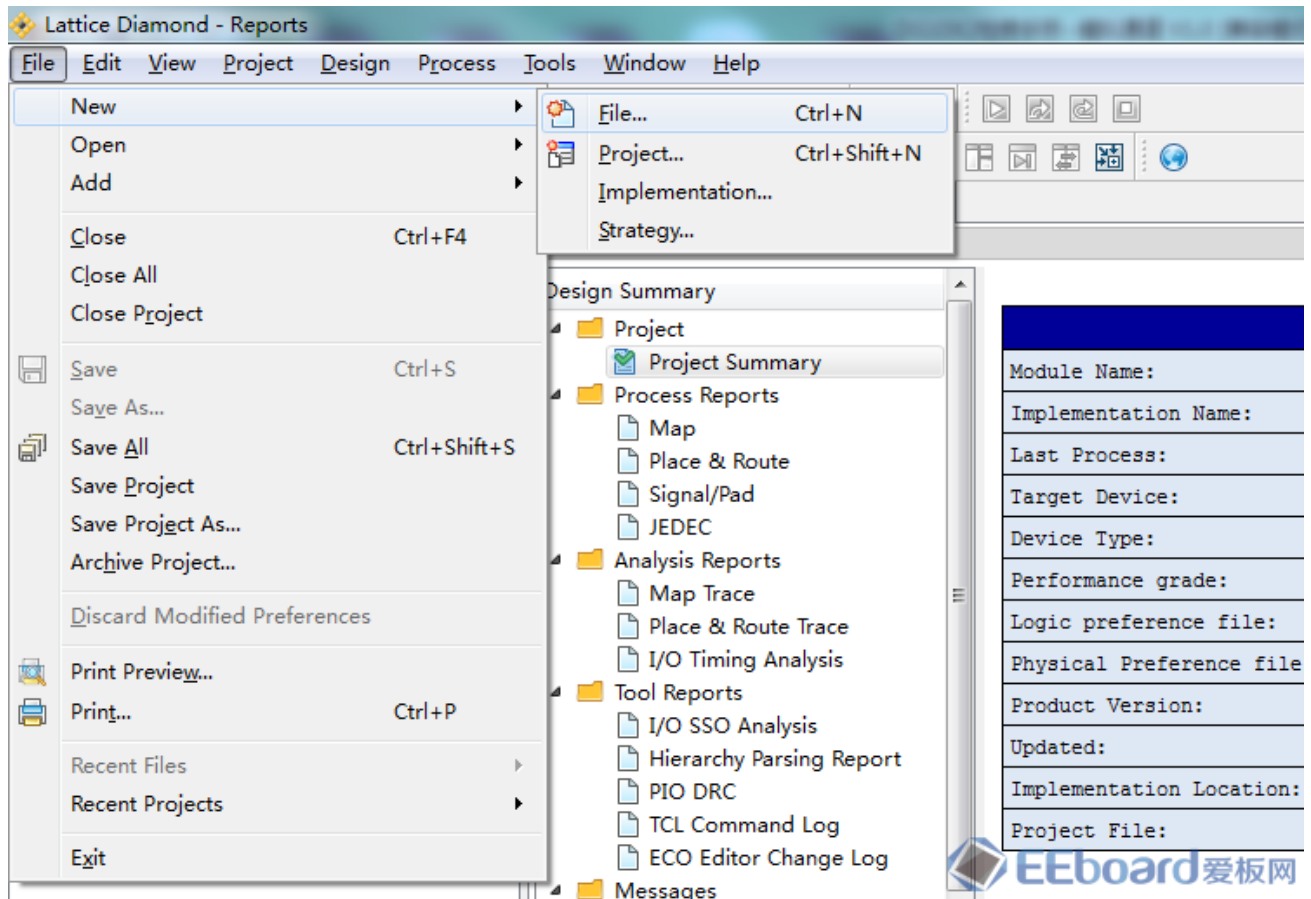
5. Elija una herramienta completa: Synplify Pro (tercero) y Lattice LSE (fábrica original) están disponibles, usaremos Lattice LSE y directamente Siguiente



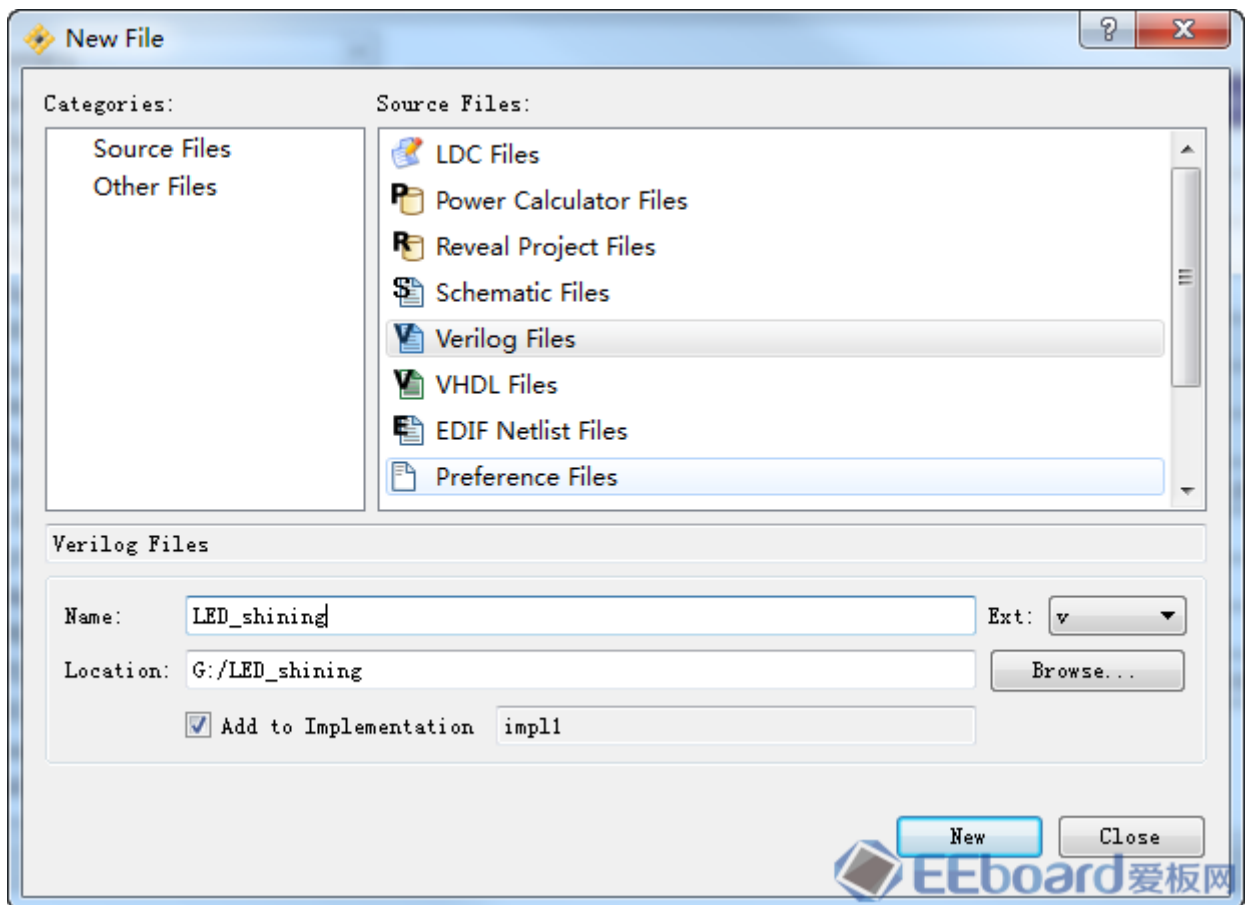
6. Confirmación de la información del proyecto: toda la información seleccionada arriba está aquí, confirme que no hay problema, directamente Finalizar



7. El proyecto ha sido construido, agregamos el archivo de diseño a continuación, seleccionamos Archivo → Nuevo → Archivo



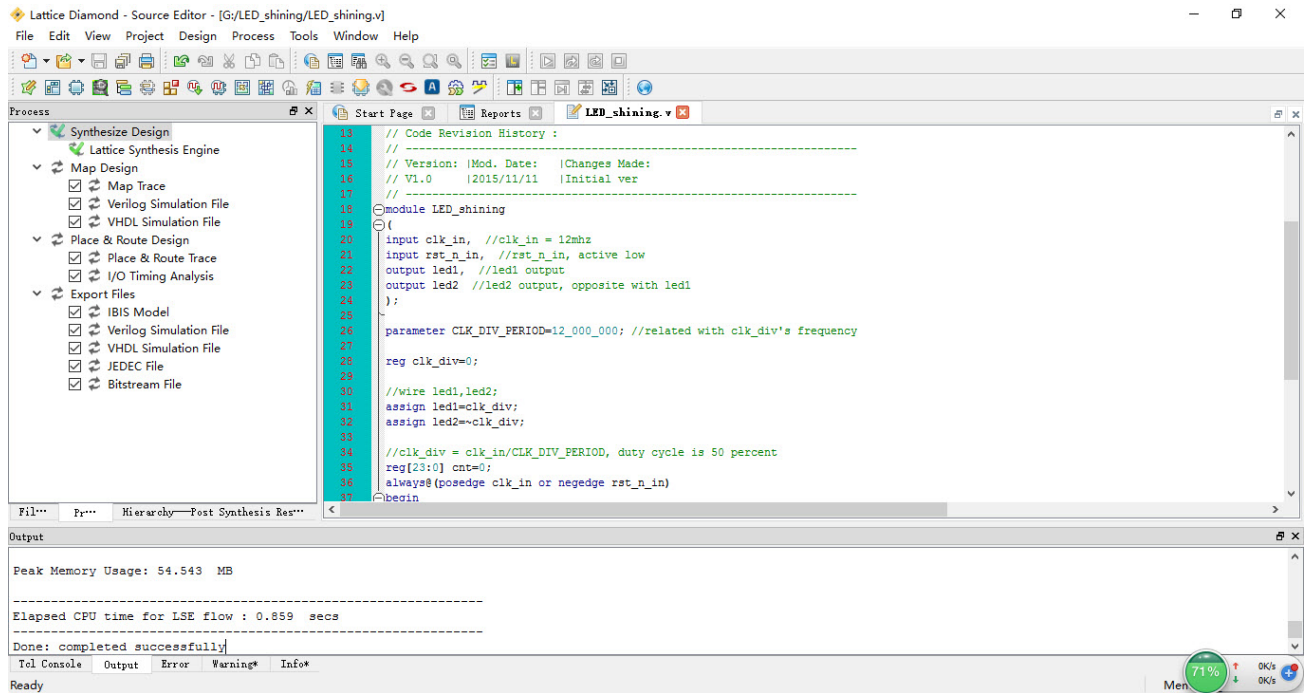
8. 选择Verilog Files (选择自己使用的硬件描述语言), Name填写LEDshining, 然后点击New, 这样我们就创建了一个新的设计文件LEDshining.v, 然后我们就可以在设计文件中进行编程了



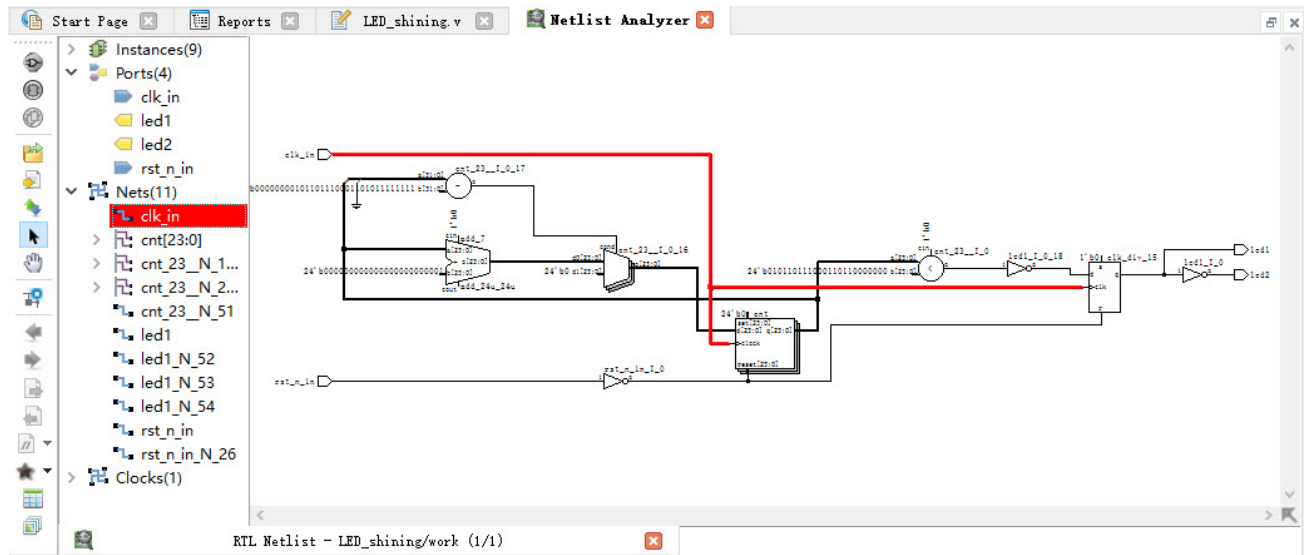
9. 程序源码已经准备好, 如下, 将代码复制到设计文件LED_shining.v中, 并保存。

```
// -----  
// >>>>>>>>>>>>>>>>>>>>> COPYRIGHT NOTICE <<<<<<<<<<<<<<<<<<<<<<<<  
// -----  
// Module: LED_shining  
//  
// Author: Step  
//  
// Description: LED_shining  
//  
// Web: www.stepfpga.com  
//  
// -----  
// Code Revision History :  
// -----  
// Version: |Mod. Date:    |Changes Made:  
// V1.0      |2015/11/11   |Initial ver  
// -----  
module LED_shining  
(  
input clk_in,           //输入系统12MHz时钟  
input rst_n_in,        //输入复位信号  
output led1,           //输出led1  
output led2            //输出led2, 与led1取反  
);  
parameter CLK_DIV_PERIOD=12_000_000; //分频常数定义  
reg clk_div=0;          //定义reg型变量, 用作分频后时钟输出  
//wire led1,led2;       //wire型变量定义, 可以省略, verilog里默认是wire型  
assign led1=clk_div;    //持续赋值语句, 将分频后时钟赋给led1, 产生闪烁效果  
assign led2=~clk_div;   //取反赋值给led2, 与led1形成交替闪烁  
//偶数分频电路 clk_div = clk_in/CLK_DIV_PERIOD, 占空比50%, CLK_DIV_PERIOD必须为偶数  
reg[23:0] cnt=0;        //分频用的计数器, 2**cnt-1>CLK_DIV_PERIOD, 计数器最大值要大于分  
频常数  
always@(posedge clk_in or negedge rst_n_in)  
begin  
    if(!rst_n_in)  
        begin  
            cnt<=0;  
            clk_div<=0;  
        end  
    else begin  
        if(cnt==(CLK_DIV_PERIOD-1)) cnt<=0;  
        else cnt<=cnt+1'b1;  
        if(cnt<(CLK_DIV_PERIOD>>1)) clk_div<=0;  
        else clk_div<=1;  
    end  
end  
endmodule
```

1. 程序编写完成，需要综合，在软件左侧Process栏，选择Process，双击Synthesis Design，对设计进行综合，综合完成后Synthesis Design显示绿色对勾（如果显示红色叉号，说明代码有问题，根据提示修改代码），如图



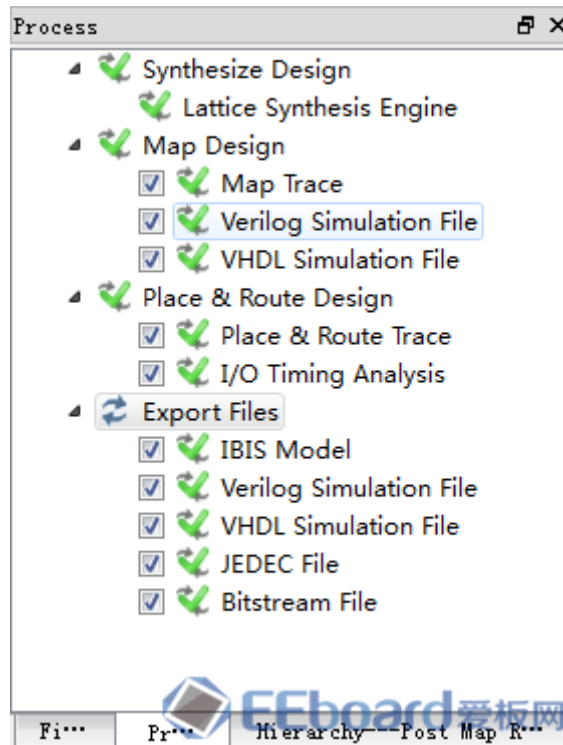
2. 通过综合工具，我们的代码就被综合成了电路，生成的具体电路，我们可以通过选择Tools → Netlist Analyzer查看（仅限Lattice的综合工具，第三方综合工具无法查看），如图



3. 综合生成电路后，分配管脚，选择Tools → Spreadsheet View,按照下图分配FPGA管脚，然后设置IO_TYPE为LVCMOS33，保存，界面如下

	Name	Group By	Pin	BANK	BANK VCC	VREF	IO TYPE	PULLMODE	DRIVE	SLEWRATE	CLAMP
1	All Ports	N/A	N/A	N/A	N/A	N/A	LVCMOS33		N/A	N/A	
1.1	Input	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
1.1.1	Clock	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
1.1.1.1	clk_in	N/A	C1	5	Auto	N/A	LVCMOS33	DOWN	NA	NA	ON
1.1.2	rst_n_in	N/A	L14	1	Auto	N/A	LVCMOS33	DOWN	NA	NA	ON
1.2	Output	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
1.2.1	led1	N/A	N13	1	Auto	N/A	LVCMOS33	DOWN	8	SLOW	OFF
1.2.2	led2	N/A	M12	1	Auto	N/A	LVCMOS33	DOWN	8	SLOW	OFF

4. 在软件左侧Process栏，选择Process，勾选所有选项，直接双击Export Files，所有布局布线输出依次完成，结束后，所有选项显示绿色对勾。

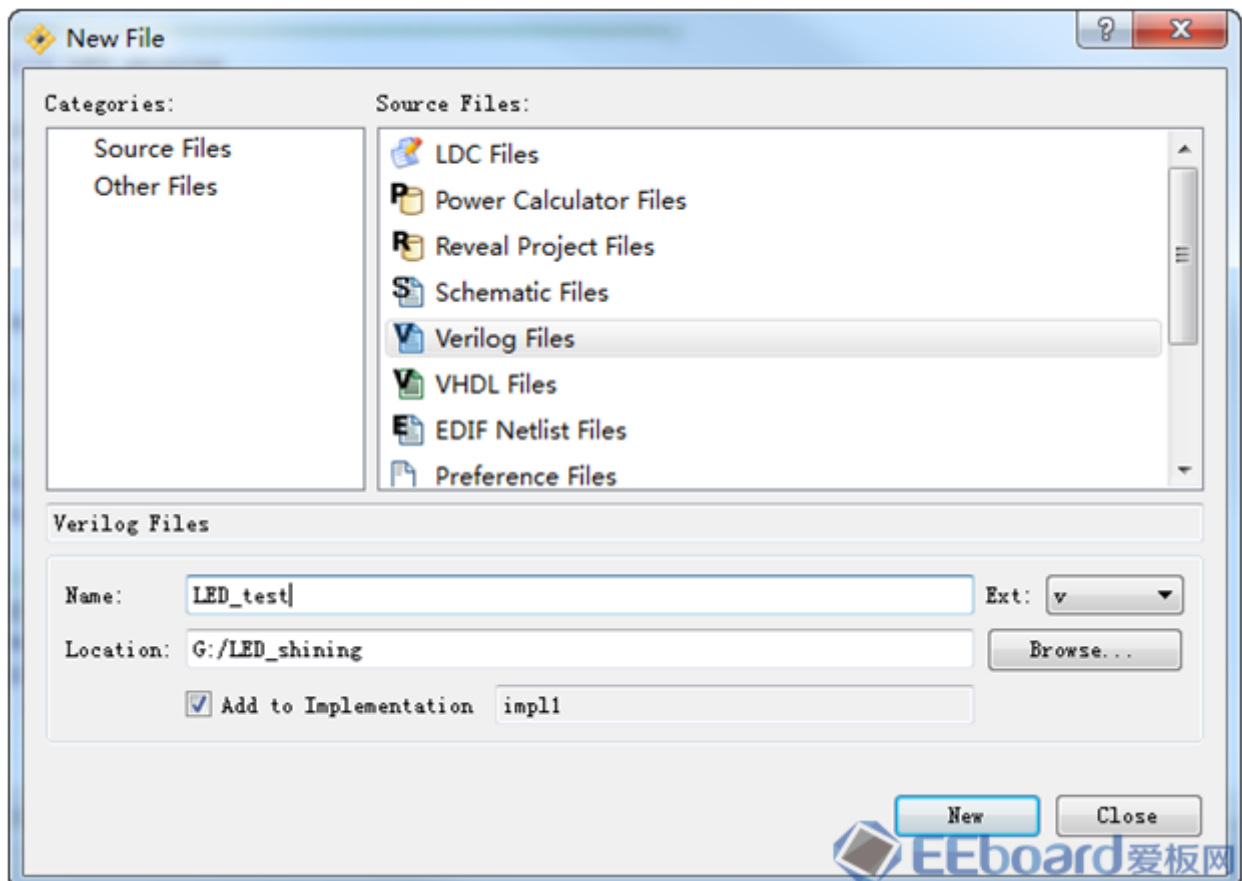


到这里完成了第一个程序流文件的生成，下面可以下载到FPGA中。

2 工程仿真

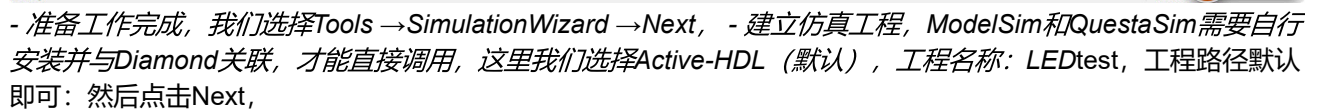
上面我们走了整个工程开发的过程，例程较为简单，对于复杂的工程开发需要预仿真和后仿真等，保证最终的程序设计逻辑和时序符合我们的设计要求。仿真软件很多，这里我们使用软件自带的Active-HDL软件进行功能仿真：

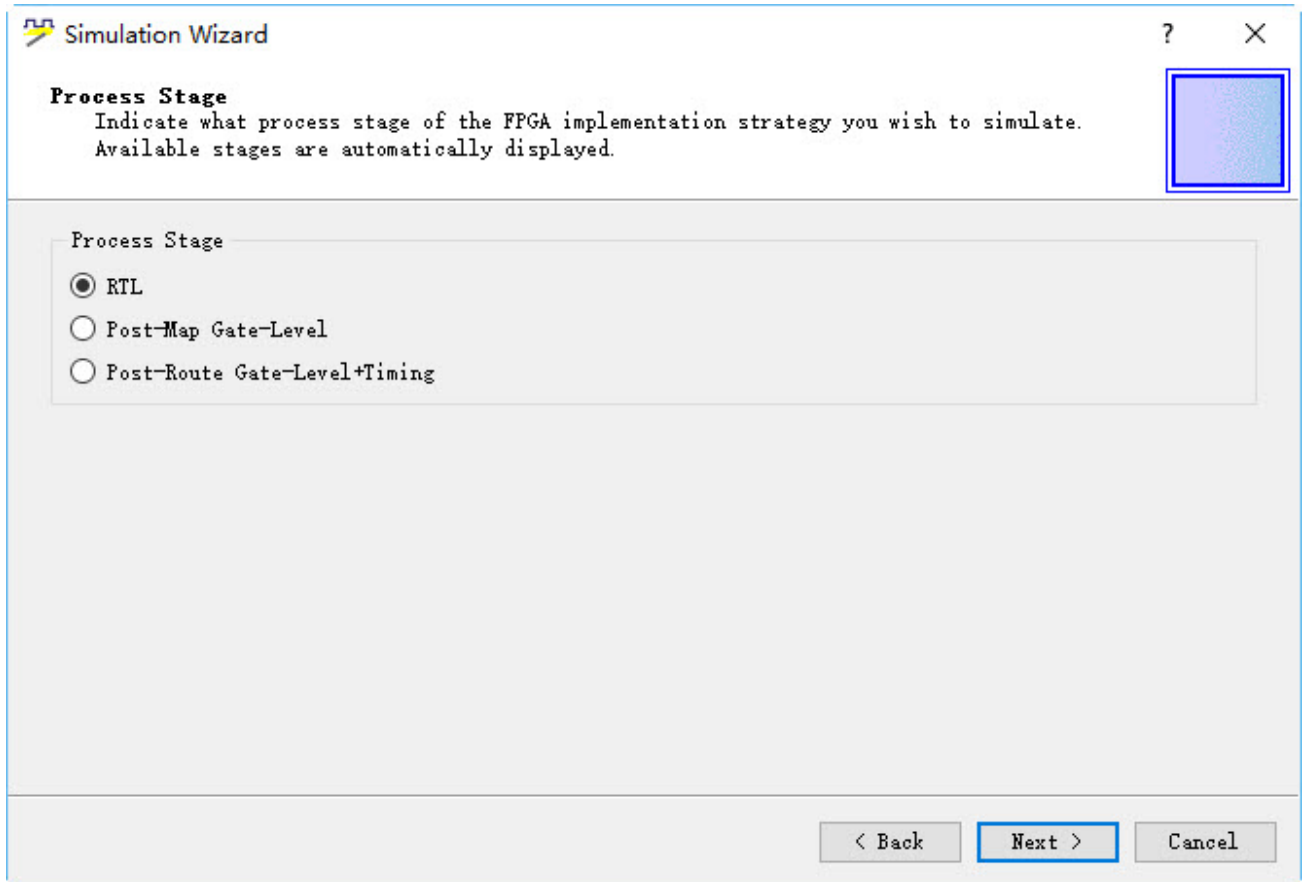
1. 首先我们添加testbench文件，和前面添加设计文件一样，File → New → File → Verilog Files, Name填写，然后 New,



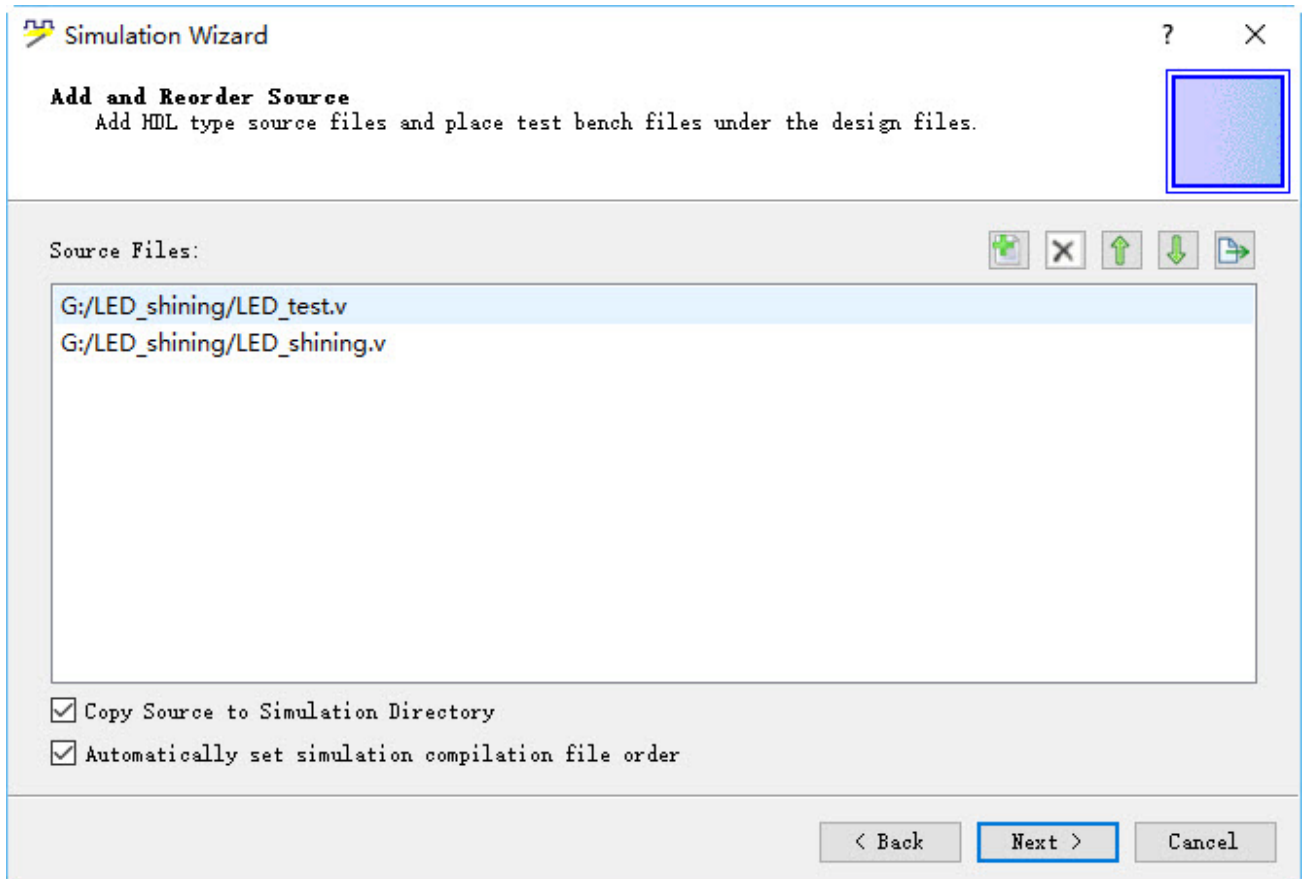
```
// -----  
// >>>>>>>>>>>>>>>>>>>>>> COPYRIGHT NOTICE <<<<<<<<<<<<<<<<<<<<<<<<  
// -----  
// Module: LED_test  
//  
// Author: Step  
//  
// Description: Testbench for LED_shining  
//  
// Web: www.stepfpga.com  
//  
// -----  
// Code Revision History :  
// -----  
// Version: |Mod. Date:   |Changes Made:  
// V1.0      |2015/11/11  |Initial ver  
// -----  
  
`timescale 1ns / 100ps  
  
module LED_test;  
  
parameter CLK_PERIOD = 40;  
parameter CLK_DIV_PERIOD=20;  
  
reg sys_clk;  
initial  
    sys_clk = 1'b0;  
always  
    sys_clk = #(CLK_PERIOD/2) ~sys_clk;           //产生周期为40ns的时钟激励，频率25MHz  
  
reg sys_rst_n;  
//产生一个初始100ns低电平然后变高电平的复位信号激励  
initial  
begin  
    sys_rst_n = 1'b0;  
    #100;  
    sys_rst_n = 1'b1;  
end  
  
wire led1,led2;  
//module例化  
LED_shining #  
(.CLK_DIV_PERIOD(CLK_DIV_PERIOD))  
LED_shining_uut  
(  
.clk_in(sys_clk),          //传递时钟  
.rst_n_in(sys_rst_n),     //传递复位信号  
.led1(led1),              //传递输出led1  
.led2(led2)                //传递输出led2  
);  
endmodule
```

10/16

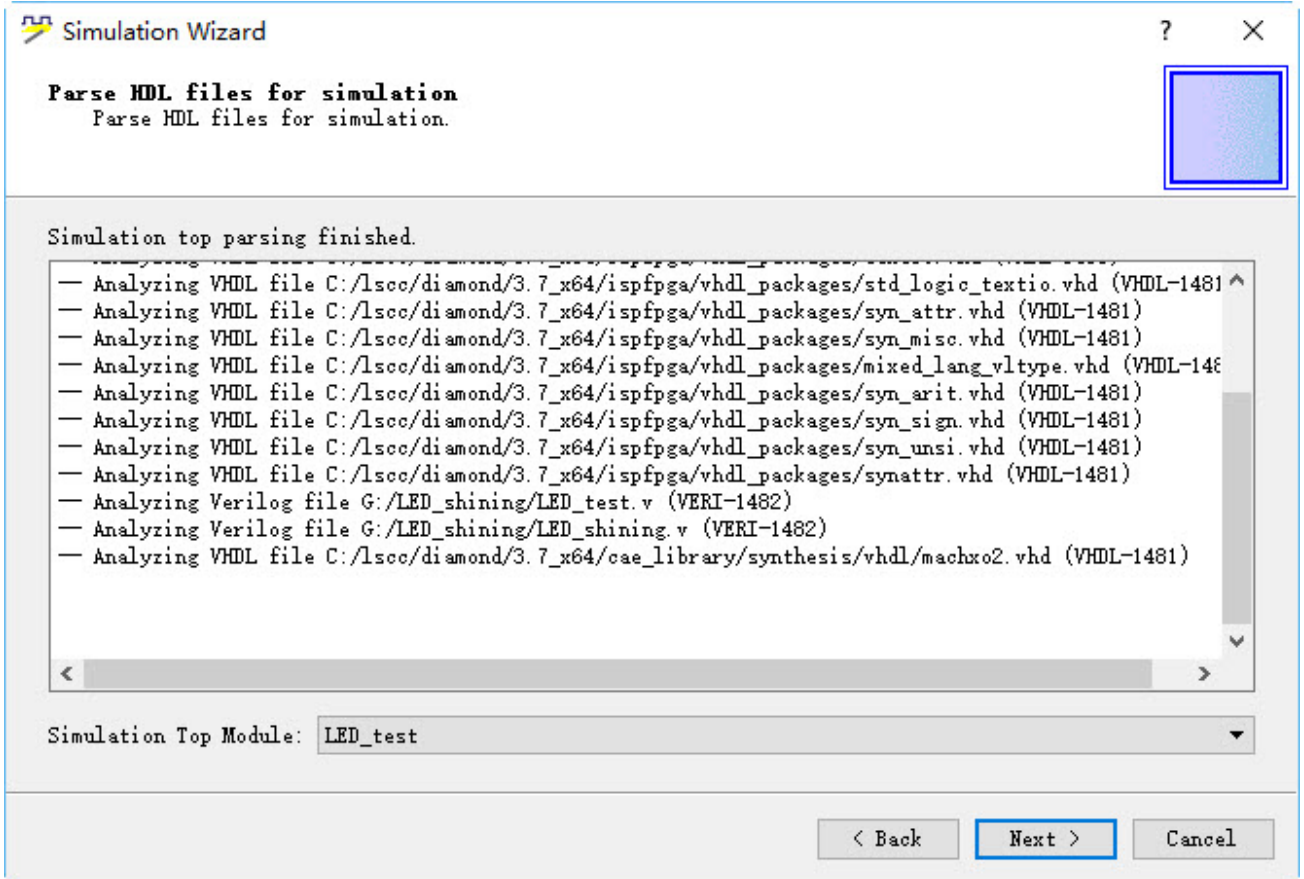




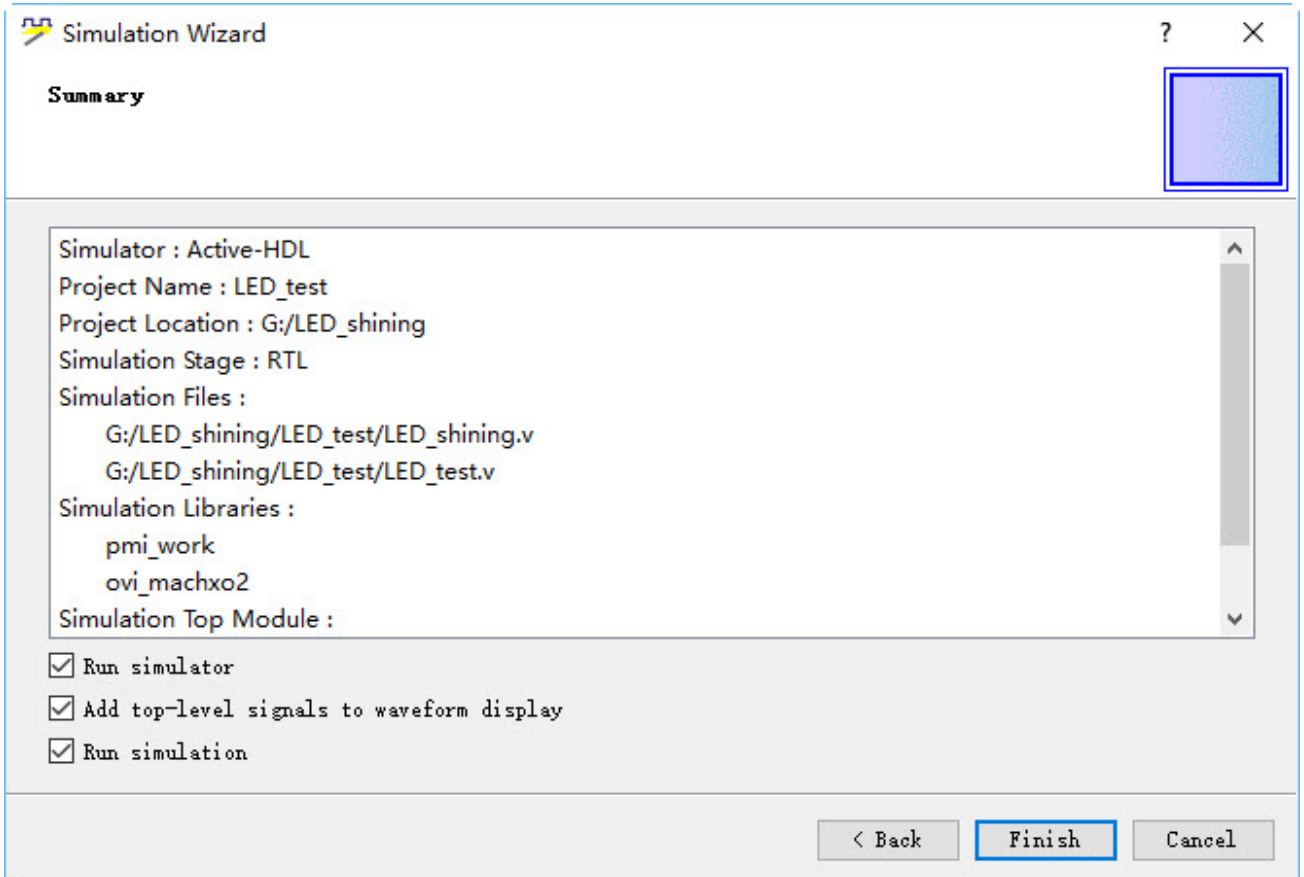
3. 勾选Copy Source toSimulation Directory, 然后Next



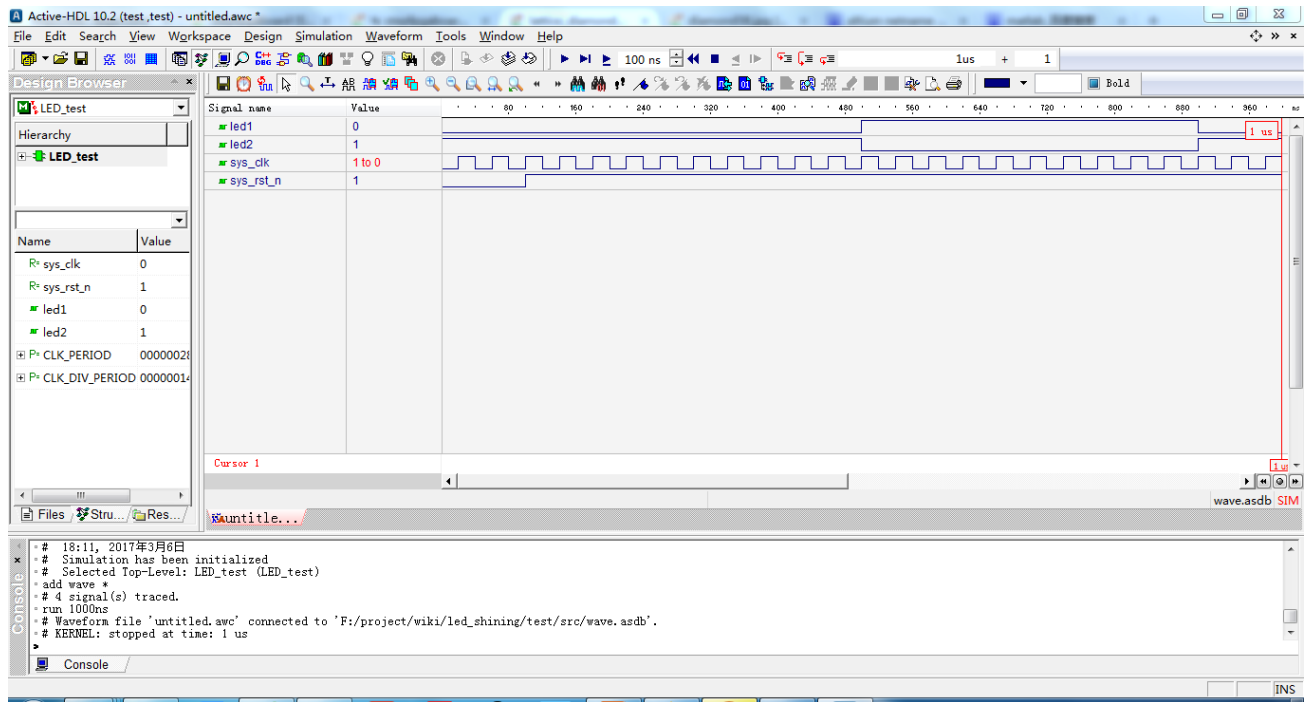
4. 点击Next



5. 点击Finish，等待仿真软件的自动运行并显示仿真时序



6. 查看仿真结果



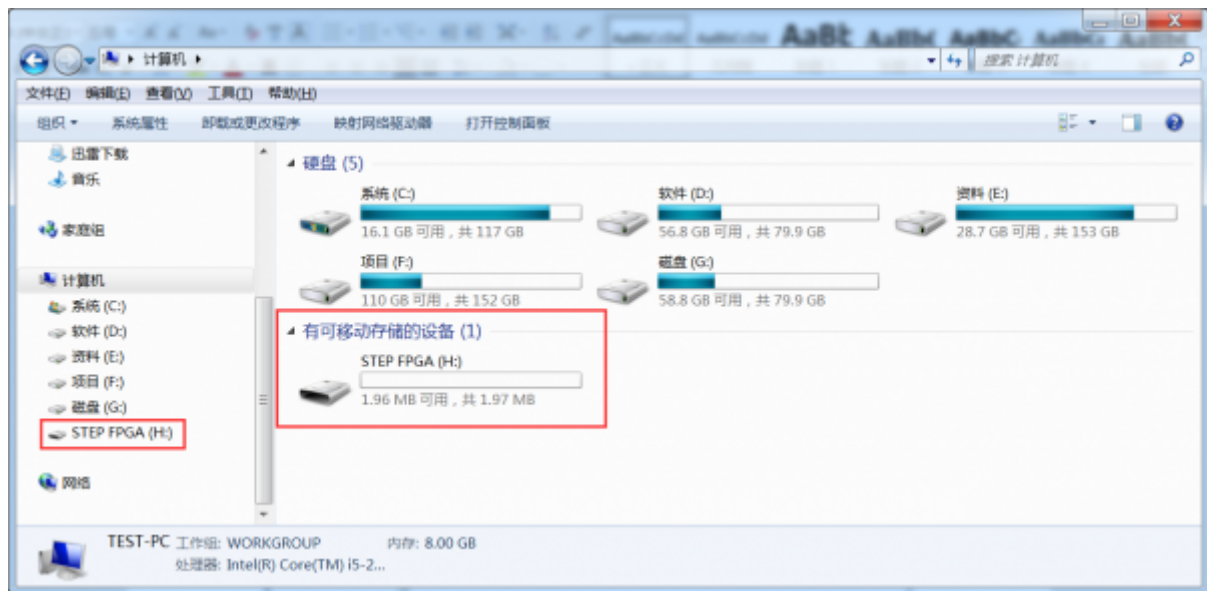
3 下载程序到FPGA

El chip de programación dedicado se ha integrado en la placa de desarrollo de pie pequeño, por lo que solo se necesita un cable Micro USB para conectarse a la computadora para completar la fuente de alimentación y las funciones de programación. La versión MXO2-C es diferente de otras versiones de Lattice MXO2 durante la descarga del programa. Después de vincular la placa a la PC, será reconocida como un "dispositivo de almacenamiento masivo", solo necesita copiar el archivo .JED generado por el programa en la placa para completar la descarga. Descargue el programa compilado en la placa de desarrollo:

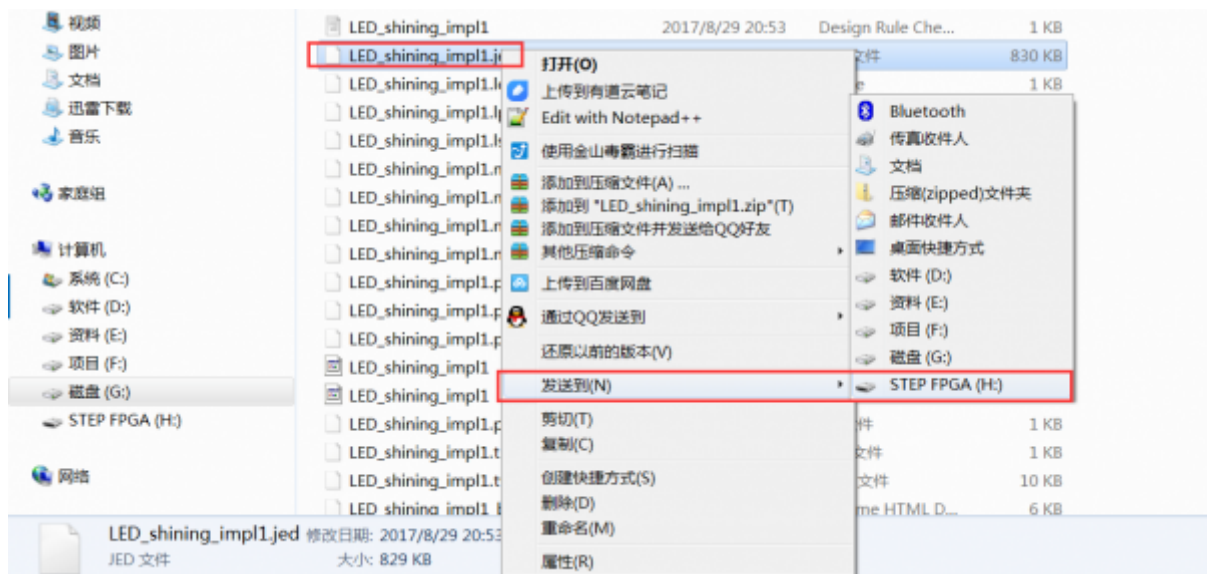
1. Utilice el cable USB Mico para conectar la placa de desarrollo a la computadora, como se muestra en la figura



1. Se mostrará una unidad flash USB llamada STEP FPGA en el Explorador



1. Busque el archivo JED generado en el directorio del proyecto, el valor predeterminado está en la carpeta impl, copie el archivo JED al disco U de STEP FPGA



1. Espere a que se complete la copia del archivo



1. Observe que el LED de la placa de desarrollo FPGA parpadea alternativamente y funciona.

4 Introducción a STEP MXO2

Aquí entendemos el proceso completo de desarrollo con el software Diamond. A continuación, comenzamos el tutorial introductorio STEP-MXO2 paso a paso en el diseño lógico programable.