

## 实验一 VIVADO工具与Verilog语言的使用

### 1. 拨码开关控制LED灯-熟悉 Vivado 和 EGO-1 实验板的使用

#### 一、实验目的

熟悉 Vivado 的开发环境及开发流程，掌握 Vivado 中 Verilog HDL 文本输入设计方法，包括仿真、综合、实现与下载。熟悉 EGO-1 实验板的功能和使用方法。

#### 二、实验内容

运用 Verilog HDL 语言，在 Vivado 中实现用 8 位拨码开关控制 8 位 LED 灯的电路，并将设计下载到 EGO-1 开发板，验证结果。

**注意：**EGO-1 实验板的主芯片为 XC7A35T-CSG324-1，需要 64 位的 Vivado（2015.4 及以后的版本）。本指导书中的配图以 Vivado 2018.3 Webpack 版展示 Vivado 环境下的基本操作。

#### 三、实验步骤

##### 1. 创建一个项目

双击  打开 Vivado，然后点击  创建一个新项目（或者在菜单栏选择 File->Project->New）。图 1-1 所示。

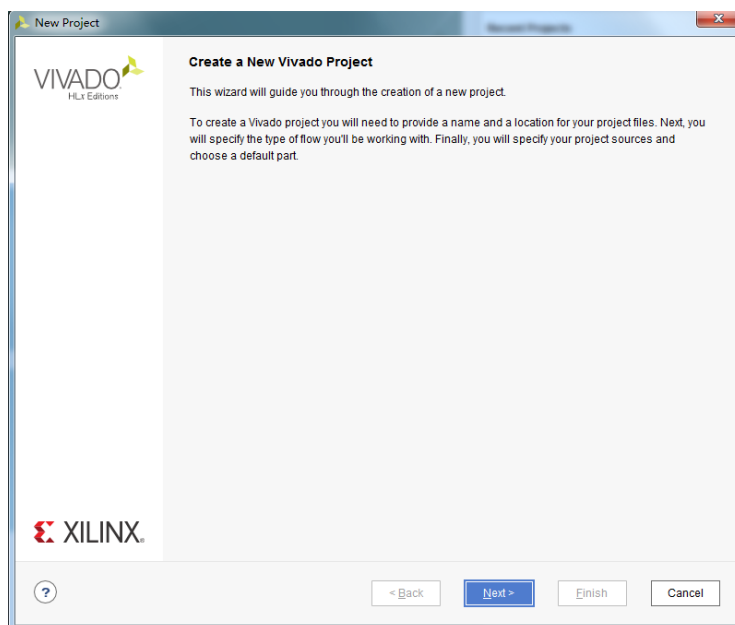


图 1-1 New Project

点击 **Next**。显示如图 1-2 所示的界面。按图 1-2 中所示命名项目名称和路径。这里项目名

称为 Ex\_1，项目的位置是 D:/digit\_FPGA，点击**Next**。最后，整个项目将在 D:/digit\_FPGA/Ex\_1中。

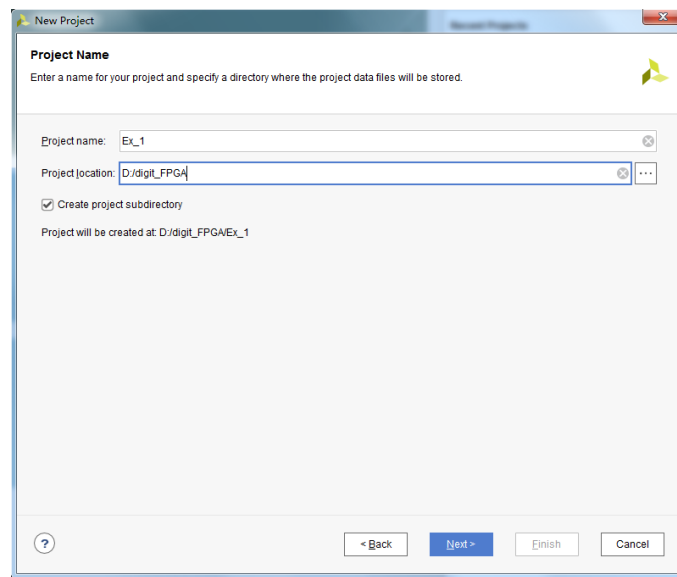


图 1-2 项目名称

如图 1-3 所示设置选择项目类型。不需要增加源文件，勾选“Do not specify sources at this time”，点击 **Next**。

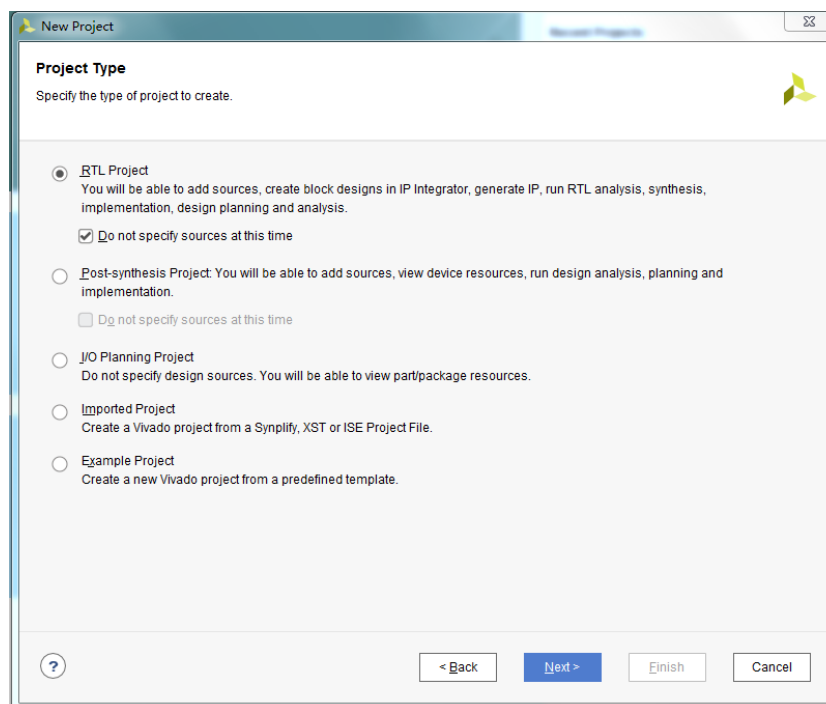


图 1-3 项目类型

按图 1-4 改变下拉列表中的选项，选择器件为 xc7a35tcsg324-1。点击 **Next**。

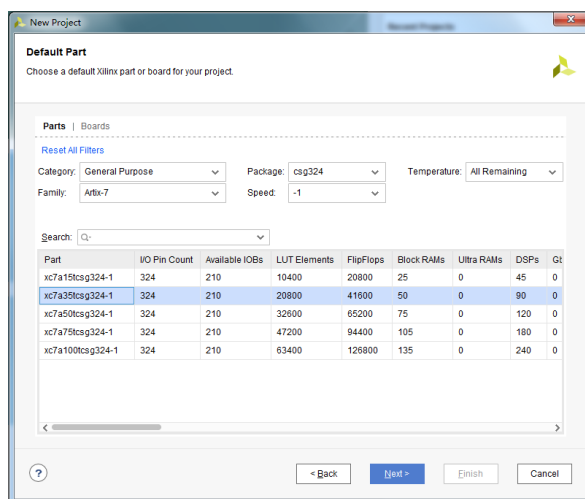


图 1-4 选择器件

可以看到如图 1-5 所示的新项目概览。点击 **Finish**。

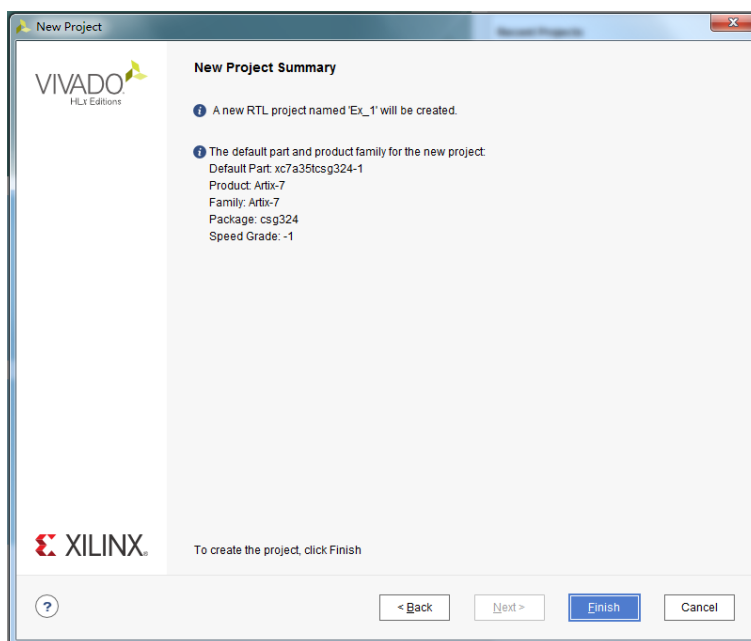


图 1-5 新项目概览

## 2. 添加源代码

在图 1-6 所示的窗口中右键点击 **Design Sources**，在弹出的菜单中选择 **Add Sources...**，出现图1-7 所示的对话框。

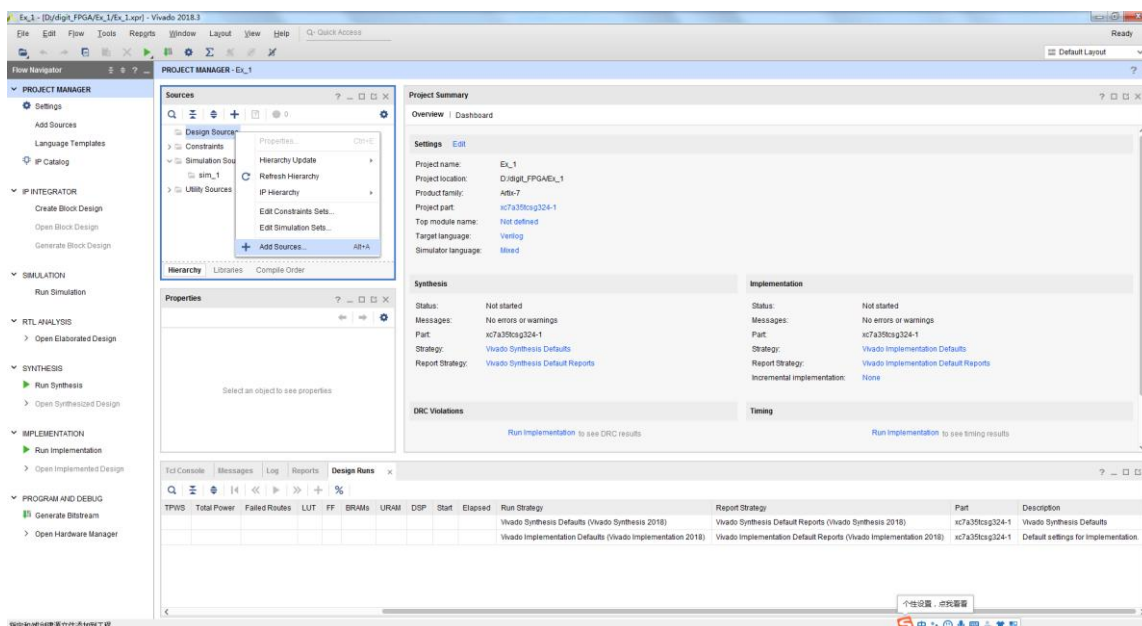


图 1-6 创建新项目后的界面

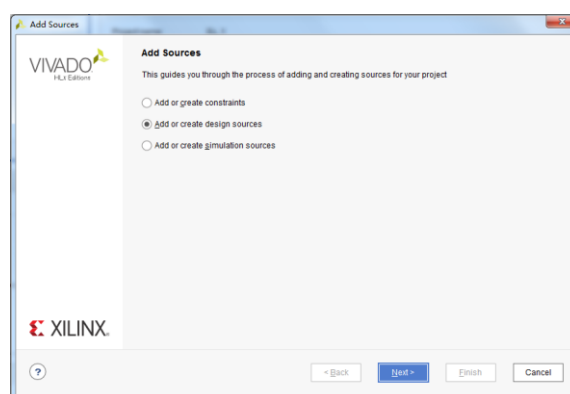



图 1-7 添加源程序对话框

按照图1-7 所示选择后点击 Next。在接下来打开的对话框（如图1-8 所示）中点 ，并选择 Create File...。

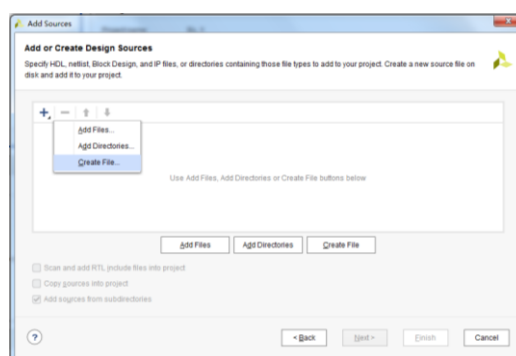


图 1-8 添加或创建设计文件对话框

此时会看到 Create Source File 对话框，按照图 1-9 所示填写后点击 OK。

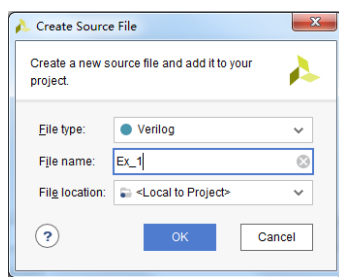


图 1-9 Create Souese File 对话框

此时会看到图 1-10 所示的界面。

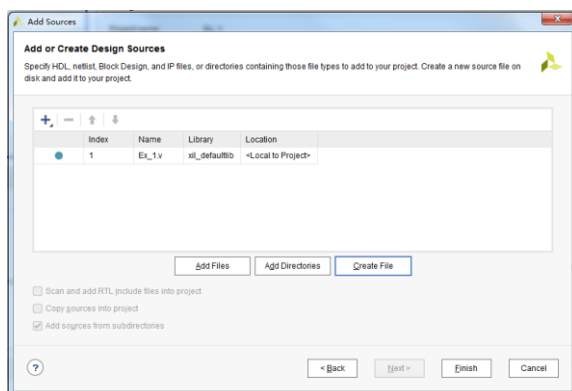


图 1-10 创建设计文件后的添加或创建设计文件对话框

点击 **Finish**。在接下来的 Define Module 对话框中如图 1-11 所示设置，然后点击 **OK**。  
(特别要注意Direction的设置)

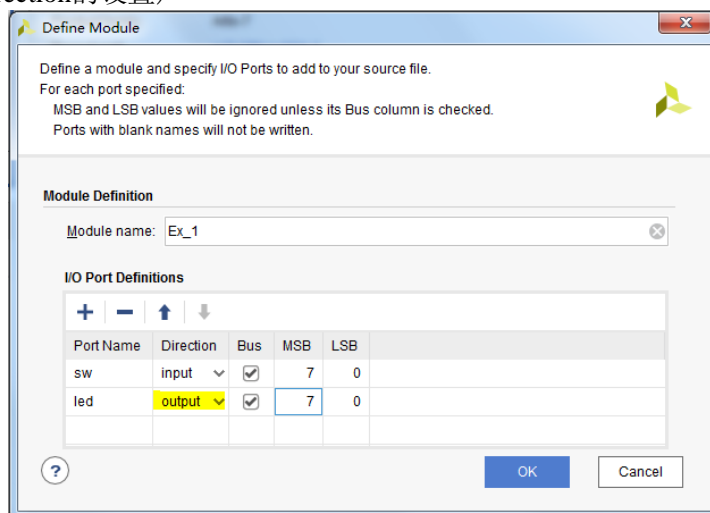


图 1-11 Define Module 对话框

接下来在图 1-12 所示界面上双击 Ex\_1 文件就会在右边显示初始的 Ex\_1 文件内容。

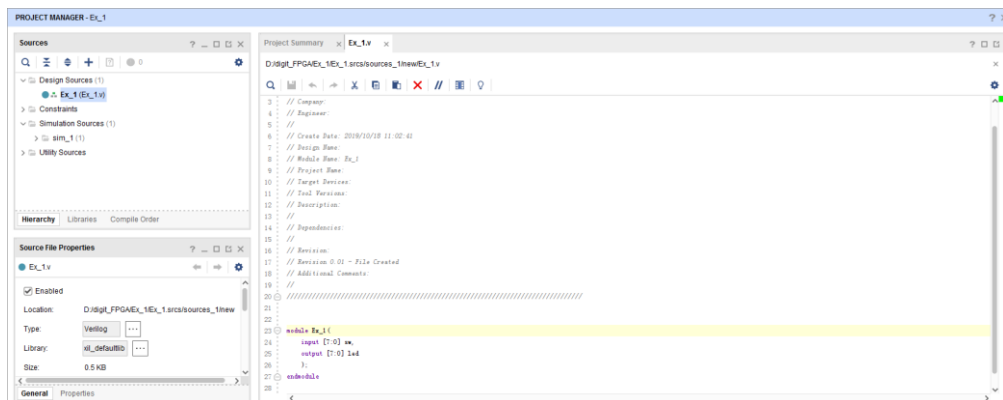


图 1-12 打开Ex\_1.v

可以看到文件中 Ex\_1 的模块是空的。

```
module Ex_1(
    input [7:0] sw,
    output [7:0] led
);
endmodule
```

用下面的程序将这个空模块替代掉。

```
module Ex_1(
    input [7:0] sw,
    output [7:0] led
);
    assign led = sw;
endmodule
```

这个模块很简单，就是将拨码开关上的高低电平赋值给 LED。

### 3. 仿真

检查电路设计是否正确。右键点击 Project Manager 下面 Sources 中的 Simulation Sources，在弹出的菜单中选择Add Source…。按照图1-13 所示的设置点击Next。

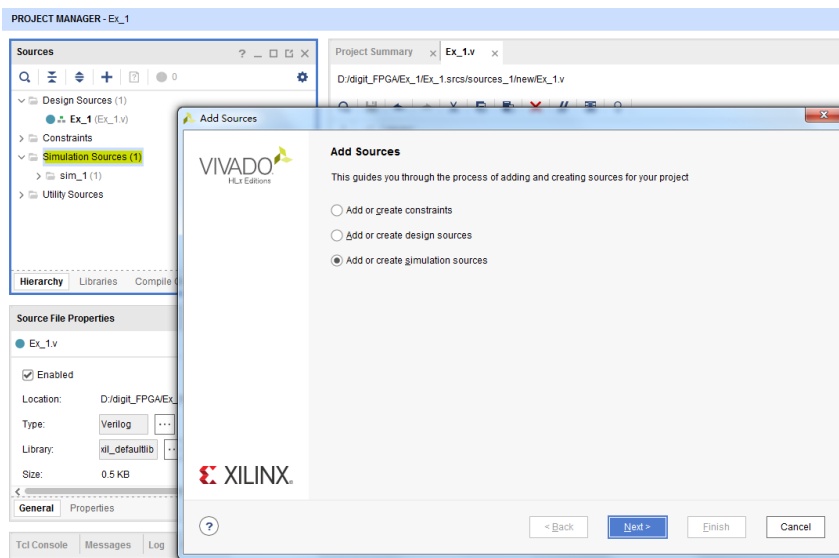


图 1-13 添加仿真源程序

在弹出的如图1-14 的窗口中点 **+**，并选择Create File...

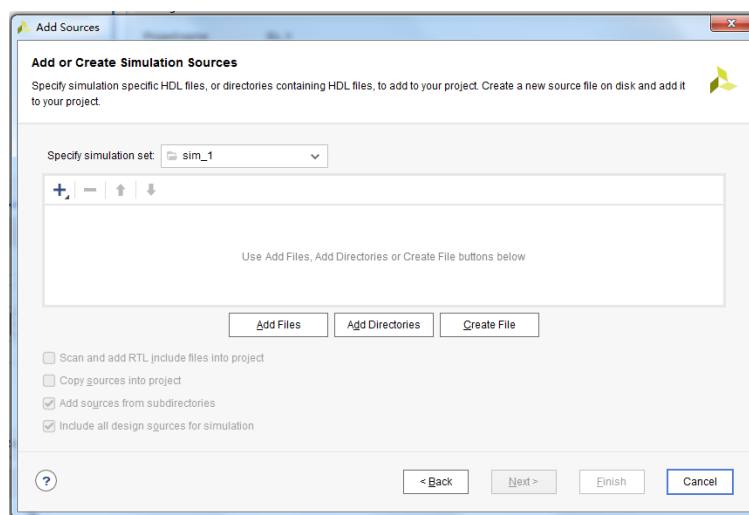


图 1-14 增加仿真源程序第二步——Add Sources 对话框

在创建文件的窗口如图 1-15 那样设置仿真源文件的文件名为 Ex\_1\_sim，并点击**OK**。

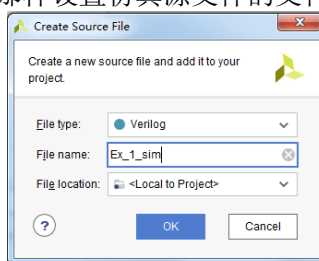


图 1-15 设置仿真源文件文件名

现在回到了 Add Sources 对话框，点击 **Finish**。在接下来的 Define Module 窗口中点击 **OK**，接下来弹出的窗口（如图 1-16）点击 Yes

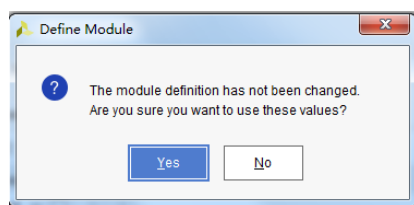


图 2-16 Define Module 窗口

如图 1-17 所示，现在在项目中看到了这个仿真源文件（图中高亮部分）

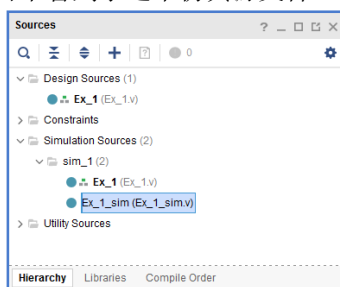


图 1-17 项目中的仿真源文件

双击图 1-17 中的高亮部分，打开该文件。可以看到目前的模块定义为：

```
module Ex_1_sim(
```

```
    );
```

```
endmodule
```

用下面的代码替代。

```
module Ex_1_sim( );
    //input
    reg [7:0] sw = 8'h00;
    //output
    wire [7:0] led;
    //instantiate the Unit under test

    Ex_1 uut(
        .sw(sw),
        .led(led)
    );
    always #10 sw = sw+1;
endmodule
```

上面的代码首先例化了 Ex\_1 模块，对 sw 初始化为 0。

always #10 sw = sw+1; 这句每隔 10 个单位时间将 sw 加 1。注意 Ex\_1\_sim.v 文件的第一行是 `timescale 1ns / 1ps 这表明一个单位时间是 1ns，10 个单位时间就是 10ns，因此 sw 每隔 10ns 会加 1。

设计文件新建完成后，在 Design Sources 和 Simulation Sources 中都有，而仿真文件只会出现在 Simulation Sources 文件夹中。设计文件可以用于仿真，也可以用于产生最终烧写进开发板的比特流，而仿真文件仅用于仿真。

保存好 Ex\_1\_sim.v 文件后，得到的项目文件层次如图 1-18 所示。



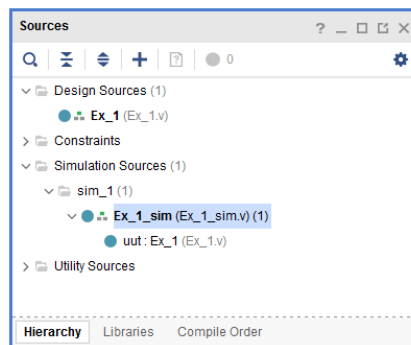


图 1-18 编辑完 Ex\_1\_sim.v 后的项目文件层次

在如图 1-19 所示的 Flow Navigator->Simulation 中点击 Run Simulation，在弹出的菜单中选择 Run behavior Simulation。

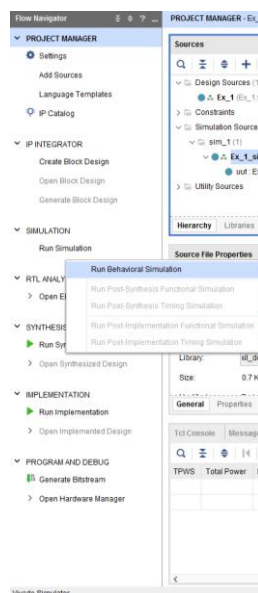



图 1-19 点击 Run Simulation

仿真完后，点击 ，使Cursor回到0时刻，再按住Ctrl键向下滚动滚轮缩放时间轴，得到如图 1-20 所示的波形图。

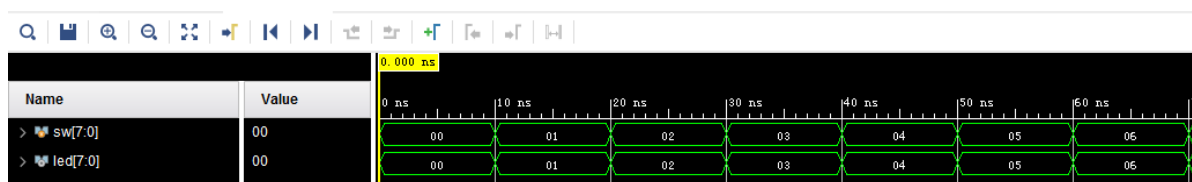


图 1-20 仿真波形图

从图上可以明显地看到，每隔 10ns，输入数据 sw 加 1，输出数据 led 与 sw 同步变化。

#### 4. 添加约束文件

单击右上角，退出Simulation界面，在弹出的Confirm Close中点选OK，回到Project Manager界面。右键单击 Project Manager 下面 Sources 中的 Constraints，在弹出的菜单中选择 Add Source…。按照图 1-21 所示的设置点击 Next。

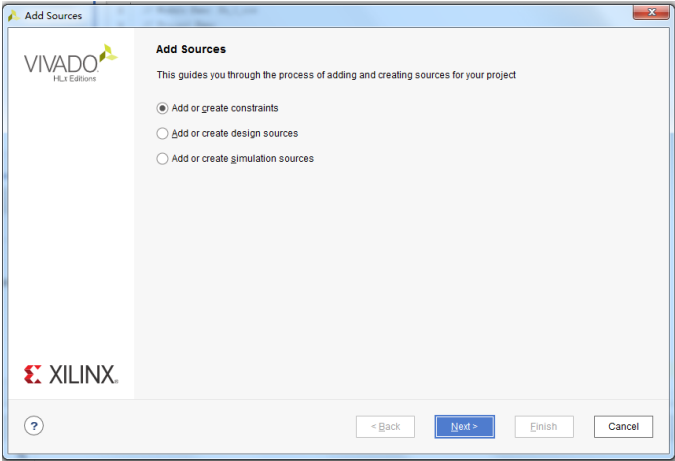


图 1-21 添加约束文件


约束文件初始为空，拷贝Ex\_1.xdc的内容，添加。

约束文件中，将 IOSTANDARD全部设置成 LVCMOS33。然后根据表 1-1，对每个管脚进行分配。（管脚分配既可以参考实验板的说明手册，也可以直接在实验板丝印上读出，如SW0-R1说明开关SW0对应管脚R1）

表 1-1 Ex\_1 管脚分配表

信号	部件	管脚	信号	部件	管脚
led[0]	D1	F6	sw[0]	SW0	P5
led[1]	D2	G4	sw[1]	SW1	P4
led[2]	D3	G3	sw[2]	SW2	P3
led[3]	D4	J4	sw[3]	SW3	P2
led[4]	D5	H4	sw[4]	SW4	R2
led[5]	D6	J3	sw[5]	SW5	M4
led[6]	D7	J2	sw[6]	SW6	N4
led[7]	D8	K2	sw[7]	SW7	R1

#### 5. 综合 (Synthesis)

在Flow Navigator中点击Synthesis ->Run Synthesis 或者单击工具条上按钮并在弹出的列表中点选Run Synthesis

综合如果没有问题， 会弹出如图1-23 所示的窗口。

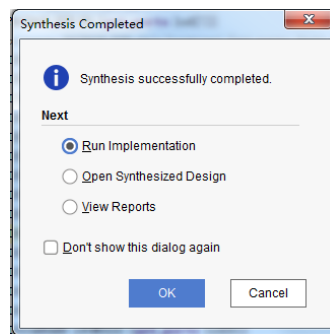


图 1-23 综合以后的窗口

按下面步骤查看管脚分配结果：在左侧Flow Navigator中展开RTL ANALYSIS,点击Open Elaborated Design，如果弹出Elaborated Design对话框，直接无视，点选OK，由此进入Elaborated Design界面，调出该界面后再点击菜单栏Layout->I/O Planning，在界面下部可以看到管脚分配情况如下图所示：

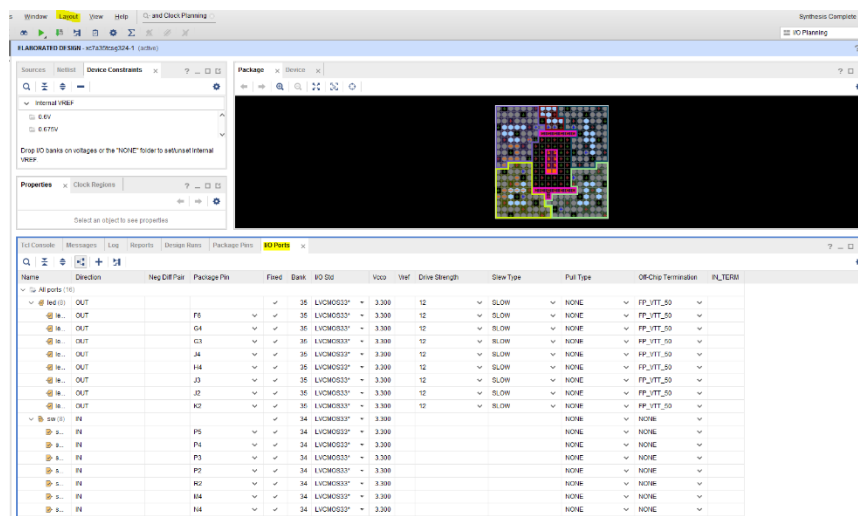



图 1-22 管脚分配后查看结果图

## 6. 实现(Implementation)

关闭Elaborated Design界面，单击工具条上按钮在弹出的列表中点选Run Implementation 或者在 Flow Navigator 中点击 Implementation ->Run Implementation 来对设计进行实现。

实现完后出现图 1-24 的窗口。

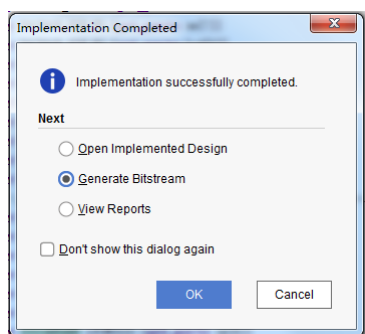


图 1-24 实现后的窗口

这样就完成了实现。

## 7. 产生比特流文件并下载

按照图 1-24 的设置点击 **OK**，或者在 Project Manager 中点击 Program and Debug→

Generate Bitstream 或者工具条上  按钮。比特流生成后会出现图 1-25 所示的对话框

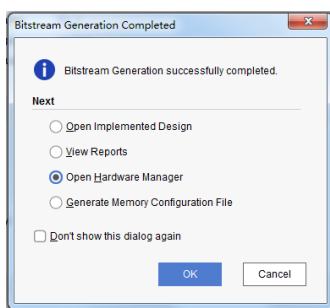


图 1-25 比特流生成完成

用USB\_JTAG线将EG01实验板的JTAG（J22）与PC机的USB相连，**打开EGO-1 板的电源**（如果不打开电源则无法连接！！）。按照图 1-25 所示选中 Open Hardware Manager。点击OK，出现Hardware Manager界面。

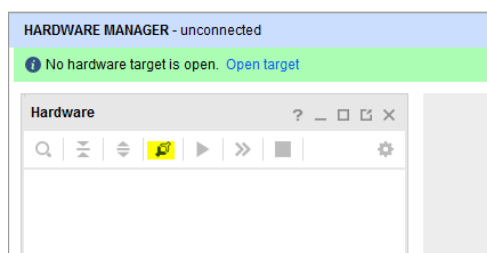



图 1-26 Hardware Manager

点击 ，或在左侧 Flow Navigator 中点击 Program and Debug->Open Hardware Manager->Open Target->Auto Connect,进行自动连接。

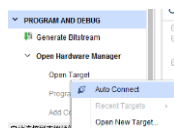


图 1-27 连接硬件

硬件连接上后，出现如图 1-28 所示界面。

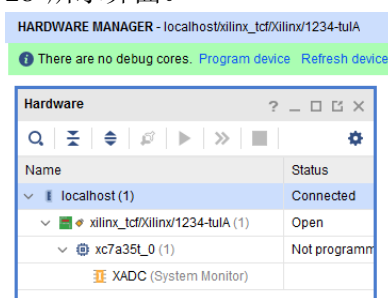


图 1-28 连接上硬件后

点击 Program devices->xc7a100t\_0。出现的 Program Device 窗口（如图 1-30）中点击 Program。

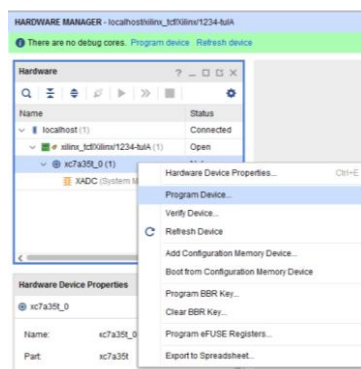


图 1-29 调出Program Device 窗口

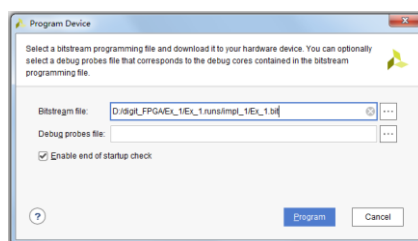


图 1-30 Program Device 窗口

出现如图 1-31 所示的正在下载的进度条。

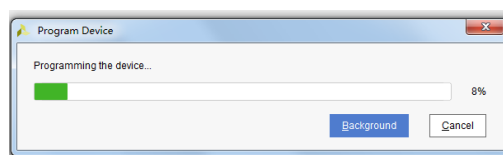


图 1-31 下载进度条

下载结束后，会看到 EGO-1 板上的 LED 灯会随着拨码开关的变化而变化，正确的现象为：开关拨上，其上面对应的贴片LED亮起；开关拨下，其上面对应的贴片LED熄灭。

实验结束，关闭 Hardware Manager。

## 2. 可配置输入端口数和数据宽度的“与非门”IP核设计

### 一、实验目的

通过实验，让学生进一步熟悉 Vivado 的使用，学会可配置 IP 核的设计与封装方法，同时也能对与非门逻辑有更直观的认识。

### 二、实验内容

使用 Verilog HDL 语言的数据流描述方法设计一个数据宽度可在 1~32 之间变化，输入端口数可在 2~8 之间变化的与非门 `nandgate`，输入端 8 个，分别是 a,b,c,d,e,f,g,h，输出端为 q。利用仿真来验证你的设计。并将该与非门封装成可配置输入端口数和数据宽度的“与非门”IP 核。

### 三、实验步骤（以与门为例）

#### 1. 创建并仿真 `andgate` 项目

创建 `andgate` 项目。其中，`andgate.v` 文件中的 `andgate` 模块如下：

```
module andgate
#(parameter Port_Num = 2, // 指定缺省的输入是 2个输入端口
parameter WIDTH=8)      // 指定数据宽度参数，缺省值是 8
(
    input [(WIDTH-1):0] a,
    input [(WIDTH-1):0] b,
    input [(WIDTH-1):0] c,
    input [(WIDTH-1):0] d,
    input [(WIDTH-1):0] e,
    input [(WIDTH-1):0] f,
    input [(WIDTH-1):0] g,
    input [(WIDTH-1):0] h,
    output [(WIDTH-1):0] q
);
assign q = (a & b & c & d & e & f & g & h);
```

```
endmodule
```

建立如下的仿真文件仿真 1 位 8 输入的情况。:

```
`timescale 1ns / 1ps module
andgate_sim( );
    // input
    reg a=0;
    reg b=0;
    reg c=1;
    reg d=1;
    reg e=1;
    reg f=1;
    reg g=1;
    reg h=1;
    //output
    wire q;
    // 实例化与门的时候，设定宽度为 1
    andgate #(8,1) u(.a(a),.b(b),.c(c),.d(d),
                    .e(e),.f(f),.g(g),.h(h),.q(q));

    initial begin
        #100 a=1;
        #100 begin a=0;b=1;end
        #100 a=1;
    end

endmodule
```

仿真可以得到图 2-1 所示的仿真波形。

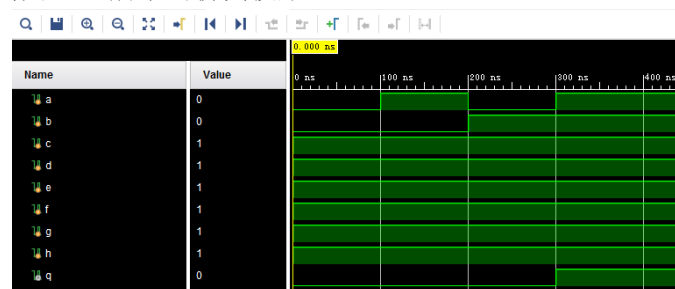


图 2-1 1 位 8 输入与门的仿真波形

将c,d,e,f,g,h 这6 个输入均设置为 1，因此，q 会随着 a,b 输入的变化而变化，从图 2-1 的仿真结果可以看到当a,b 任意一个为0，q 都输出0，是满足“与门”逻辑的。

## 2. 综合并封装 IP 核

仿真正确的 andgate 模块进行综合（参考前面章节的步骤），综合结束后会出现图 2-2 所示的对话框，选择 **Cancel**。

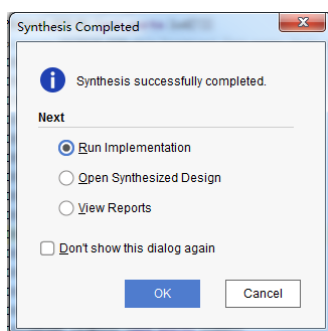


图 2-2 综合结束

在如图2-3 所示的的界面中点击**Project Manager->Settings**。

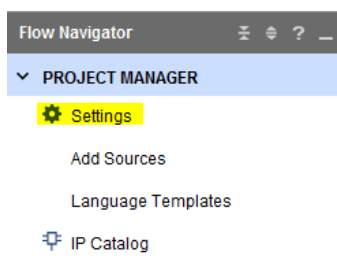


图 2-3 Project Manager-Settings

在Project Settings 对话框中选择 **IP**，并进入 **Packager** 选项卡，如图2-4 所示进行设置。

设置好后，点击 **Apply**(若不单击Apply，则可能出现设置失败的情况！！)，然后点击 **OK**。

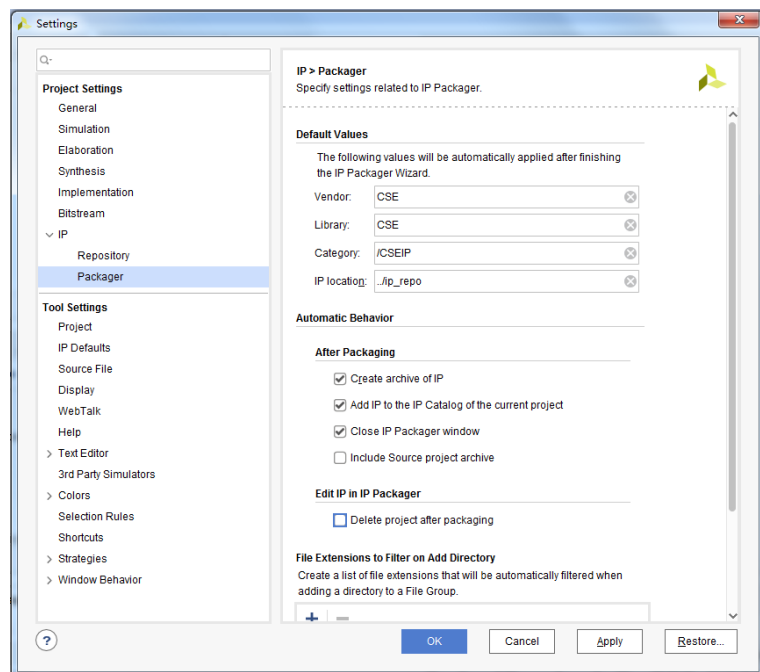


图 2-4 项目设置中设置 IP 核封装属性

在Vivado 的菜单栏中选择 **Tools->Create and Package IP...**。在弹出的窗口中点击



**Next**。在之后弹出的窗口中如图 2-5 所示设置封装选项。点击 **Next**。

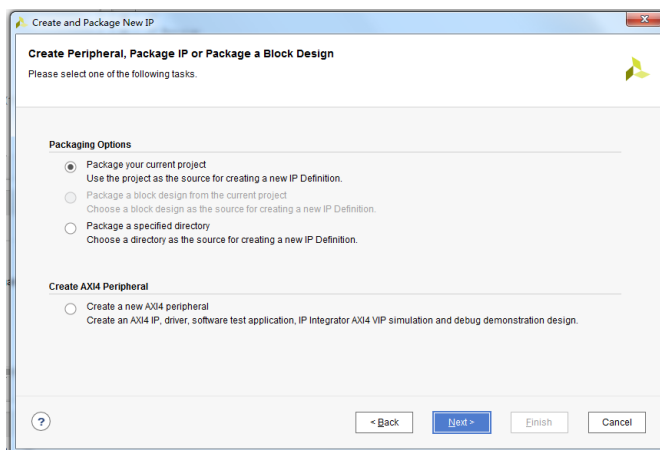


图 2-5 封装选项

在 IP Location（图 2-6）中不做修改，点击**Next**。

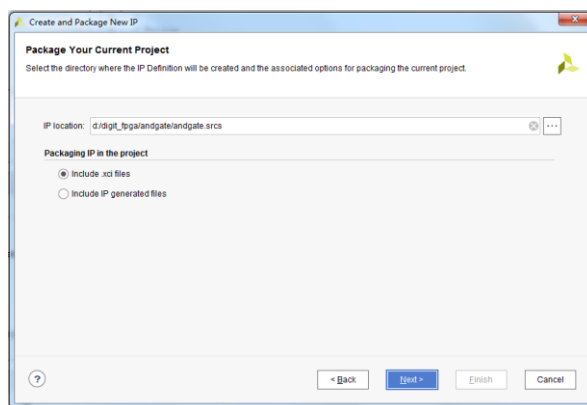


图 2-6 IP Location

封装后的 IP 放在了d:/digit\_fpga/andgate/andgate.srcs 这个文件夹中。

在图 2-7 所示的对话框中点击 **Finish**。

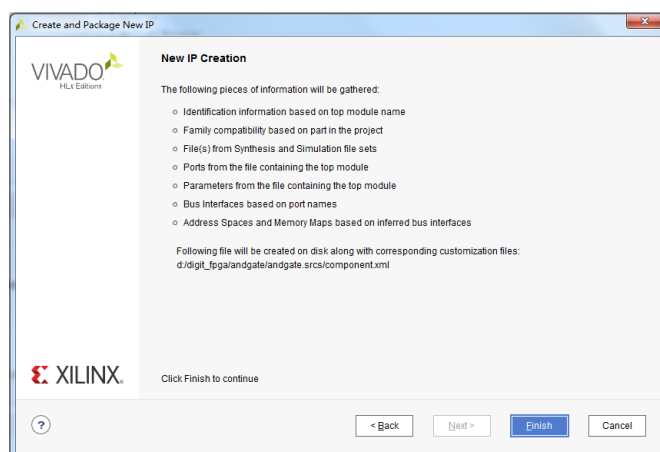


图 2-7 创建IP 核结束

这时可以看到如图 2-8 所示的 IP 封装设置。按图 2-8 那样设置好各个项目。

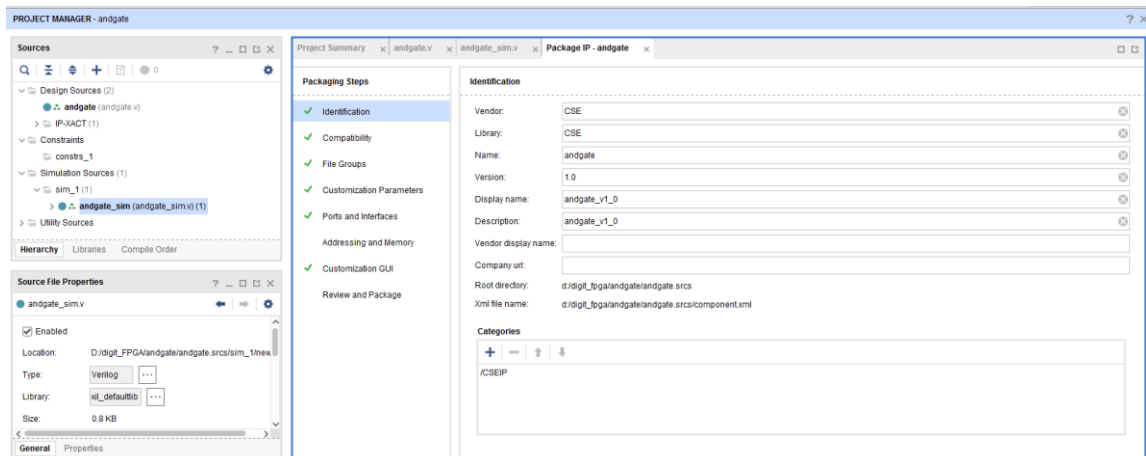


图 2-8 设置Identification

在图2-9 中可以添加IP 核所支持的芯片系列。本图已经默认添加了全部芯片系列，若希望添加新的芯片系列，可以点 **+** 并选择Add Family Explicitly，在如图2-10 所示的 Add Family 对话框。在Add Family 对话框中选中需要添加的芯片系列，底部Life-cycle 选择 Production，然后点击 **OK**。这样就设置完了 compatibility。

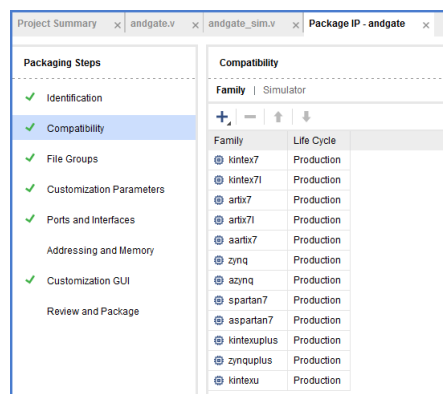


图 2-9 设置compatibility

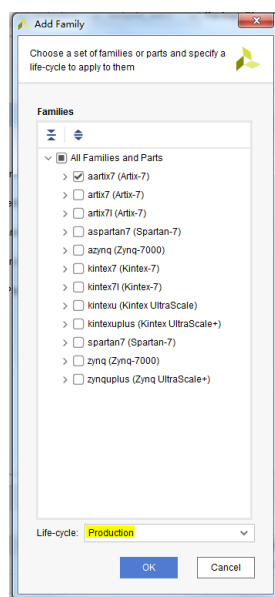


图 2-10 Add Family 对话框

接下来到 Customization Parameters（图2-11 所示）

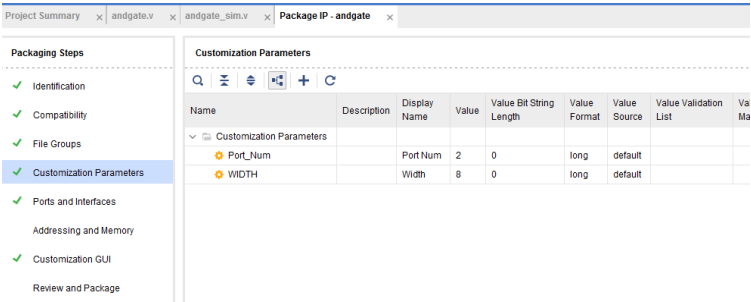


图 2-11 Customization Parameters

双击图 2-11中的 Port\_Num 得到图2-12 所示的 IP 和参数的对话框，按照图2-12 那样设置 Port\_Num 参数后点 **OK**。

图 2-12 中可以看到，IP 核至少有 2 个输入端，最多可以有 8 个输入端。

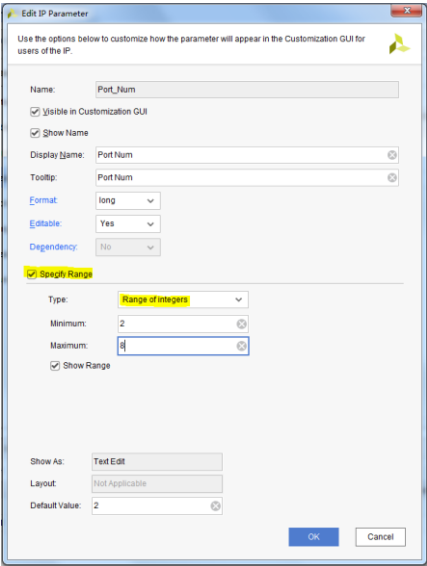


图 2-12 编辑IP 核的Port\_Num 参数

双击图 2-12 中的 WIDTH，得到图 2-13 所示的 Edit IPParameter 对话框，按照图 2-13所示设置WIDTH 参数后后点击 **OK**。

从图 2-13 图中可以看到，数据位宽WIDTH 最小是 1 位，最大是 32 位。

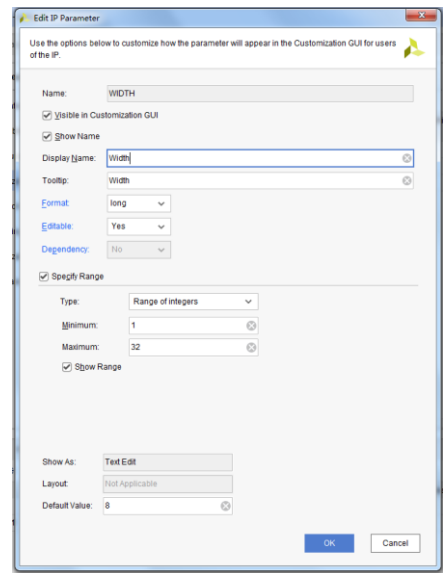
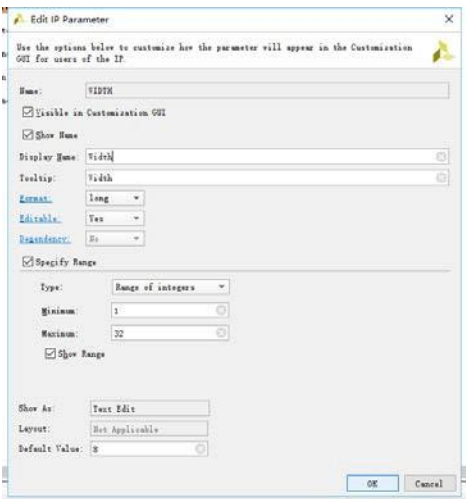


图 2-13 编辑IP 核的WIDTH 参数

接下来到 Ports and Interfaces，如图 2-14 所示。

Project Summary

andgate.v

andgate\_sim.v

Package IP - andgate

Packaging Steps

✓ Identification

✓ Compatibility

✓ File Groups

✓ Customization Parameters

✓ Ports and Interfaces

Addressing and Memory

✓ Customization GUI

Review and Package

Ports and Interfaces

🔍

🔧

⚙️

+

🔗

↺

Name	Interface Mode	Enablement Dependency	Is Declaration	Access Handle	Access Type	Direction	Driver Value	Size Left	Size Right	Size Left Dependency	Size Right Dependency	Type Name
a			<input type="checkbox"/>		ref	in		7	0	(WIDTH - 1)		std_logic_vector
b			<input type="checkbox"/>		ref	in		7	0	(WIDTH - 1)		std_logic_vector
c			<input type="checkbox"/>		ref	in		7	0	(WIDTH - 1)		std_logic_vector
d			<input type="checkbox"/>		ref	in		7	0	(WIDTH - 1)		std_logic_vector
e			<input type="checkbox"/>		ref	in		7	0	(WIDTH - 1)		std_logic_vector
f			<input type="checkbox"/>		ref	in		7	0	(WIDTH - 1)		std_logic_vector
g			<input type="checkbox"/>		ref	in		7	0	(WIDTH - 1)		std_logic_vector
h			<input type="checkbox"/>		ref	in		7	0	(WIDTH - 1)		std_logic_vector
q			<input type="checkbox"/>		ref	out		7	0	(WIDTH - 1)		std_logic_vector

图 2-14 Ports and Interfaces 设置

由于设计的数据输入端最小是 2 个，因此，a,b 两个输入端始终都是 Enable 的。而 c~h 输入端则要根据 Port\_Num 的值决定是否 Enable。请双击端口 c，在打开的对话框中，按照

图2-15 所示设置。

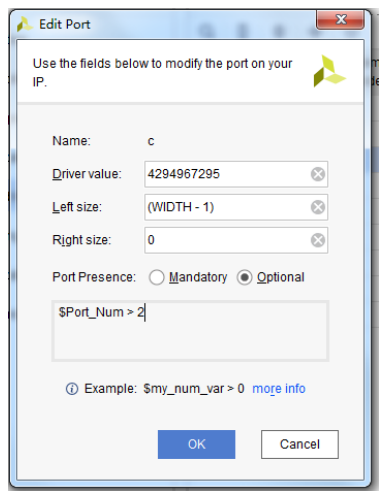


图 2-15 编辑端口c 的参数

图2-15 中Driver value 是指端口 c 在disable 时候的取值。在 andgate 模块中，共定义了 8 个输入端口，但实际应用中允许只用 2~8 个中的任意多个输入端口，不用的端口需要设置成 disable 状态。在 IP 核中，所谓 disable 的端口实际上只是不提供给外部接口输入，但在模块中该端口依然存在，所以如果必要，需要对它赋一个驱动值。由于制作的是“与门”，因此所有不接外部接口的 disable 输入端应该赋值为全 1，也就是 16 进制的 FFFFFFFF，十进制的 4294967295，因此在这里给出的赋值是 4294967295。

在看图2-15 中的Port Presence（端口存在），选择的是Optional 说明该端口的存在是有条件的，在下面的启动表达式编辑框中，输入\$Port\_Num > 2，表明当输入端口大于 2 的时候，c 端口被启用（Enable）

设置好后点击 **OK**。

请大家按照上述步骤设置好端口 d,e,f,g,h 的参数。设置好以后的 Ports and Interfaces 如图 2-16 所示。










Packing Steps		Ports and Interfaces												
		<div><div><div><div></div><div></div><div></div><div></div><div></div><div></div></div></div><div><div></div><div></div><div></div><div></div><div></div><div></div></div></div>												
✓ Identification		Name	Interface Mode	Enabment Dependency	Is Declaration	Access Handle	Access Type	Direction	Driver Value	Size Left	Size Right	Size Left Dependency	Size Right Dependence...	Type Name
✓ Compatibility		 a			<input type="checkbox"/>		ref	in		7	0	(WIDTH - 1)		std_logic_vector
✓ File Groups		 b			<input type="checkbox"/>		ref	in		7	0	(WIDTH - 1)		std_logic_vector
✓ Customization Parameters		 c		\$Port_Num > 2	<input type="checkbox"/>		ref	in	4294967295	7	0	(WIDTH - 1)		std_logic_vector
✓ Ports and Interfaces		 d		\$Port_Num > 3	<input type="checkbox"/>		ref	in	4294967295	7	0	(WIDTH - 1)		std_logic_vector
		 e		\$Port_Num > 4	<input type="checkbox"/>		ref	in	4294967295	7	0	(WIDTH - 1)		std_logic_vector
Addressing and Memory		 f		\$Port_Num > 5	<input type="checkbox"/>		ref	in	4294967295	7	0	(WIDTH - 1)		std_logic_vector
✓ Customization GUI		 g		\$Port_Num > 6	<input type="checkbox"/>		ref	in	4294967295	7	0	(WIDTH - 1)		std_logic_vector
		 h		\$Port_Num > 7	<input type="checkbox"/>		ref	in	4294967295	7	0	(WIDTH - 1)		std_logic_vector
Review and Package		 q			<input type="checkbox"/>		ref	out		7	0	(WIDTH - 1)		std_logic_vector

图 2-16 设置后的Ports and Interfaces

大家可以到 Customization GUI 验证一下参数的变化对 IP 封装的影响。接下来到 Review

and Packaging, 如图 2-17 所示。

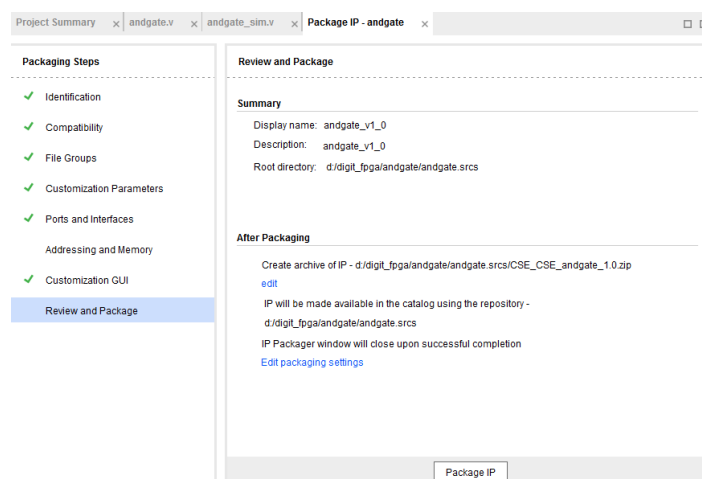


图 2-17 Review and Packaging 设置

点击 Package IP 或 Re-Package IP。这样，andgate 的 IP 核就生成了。

可以看到 IP生成到一个称为 CSE\_CSE\_andgate\_1.0.zip 文件中，路径是 D:\digit\_FPGA\andgate\andgate.srds。

为了方便今后的使用，请大家新建一个文件夹 IPcore，并将CSE\_CSE\_andgate\_1.0.zip 文件拷贝到 IPcore 文件夹中，并将其解压缩。