

第 2 章 实验手册

本章共设计了 11 个实验，大部分实验由几个小实验组成。本实验系列注重前后知识的关联性，前面的实验，会为后面的试验打基础，甚至前序课程的实验会为后续课程实验做基础。

另外，为了方便学生实验，与本讲义配套的还有实验的相关文件，大部分实验提供初始设计文件、仿真文件和约束文件，放在在 X:\sysclassfileS\digit 的相关文件夹中，X 是你安装的盘符。

2.1 VIVADO工具与Verilog语言的使用

2.1.1 拨码开关与LED灯-熟悉vivado和实验台

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 Minisys 实验板的功能和使用方法。

二、实验内容

以一个简单的 24 位拨码开关的读和 24 位 LED 灯的输出电路为例，利用 Verilog HDL 语言，在 Vivado 中创建简单的 24 位拨码开关的输入和 24 位 LED 灯的输出电路会将设计下载的 Minisys 实验平台。

注意，由于 Minisys 实验板所用的 XC7A100T-1FGG484C 芯片比较新，因此，需要 64 位的 Vivado 2015.4 及以后的版本。

三、实验步骤

1. 创建一个项目

双击  打开 Vivado，然后单击  创建一个新项目（或者在菜单栏选择 File -> New Project...）。图 2-1 所示。



图 2-1 New Project

点击 **Next**。显示如图 2-2 所示的界面。按图 2-2 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 C:/sysclassfiles/digit/Ex_1，点击 **Next**。最后，整个项目将在 C:/sysclassfiles/digit/Ex_1/Ex_1 中。

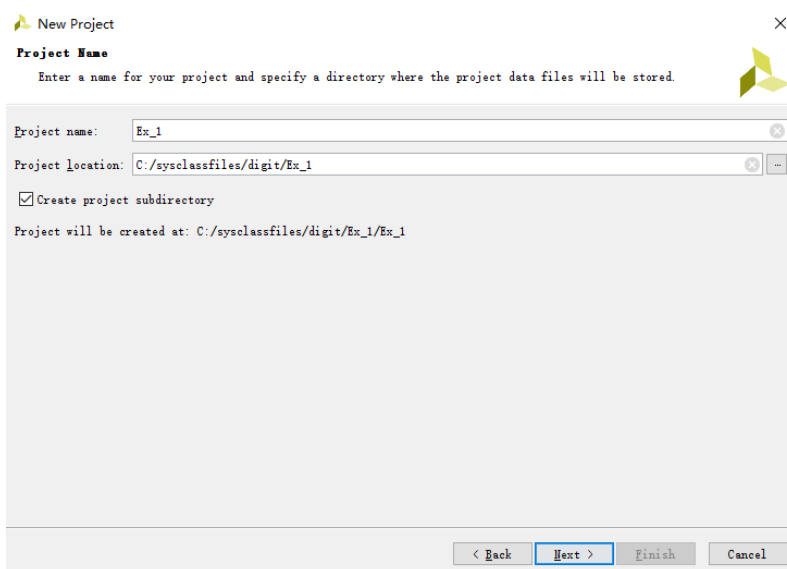


图 2-2 项目名称

如图 2-3 所示设置选择项目类型。点击 **Next**

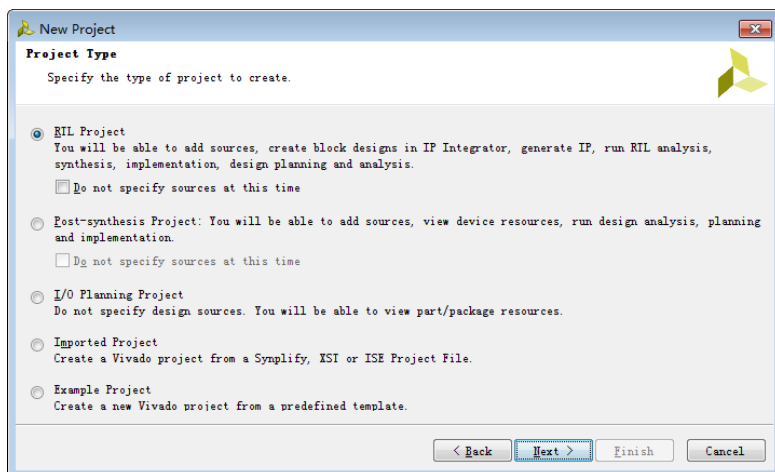


图 2-3 项目类型

因为不需要增加源文件，所以 Add source 窗口点击 **Next**（但 Target Language 选择 Verilog）。

不增加 IP 核，所以 Add ExistingIP 窗口点击 **Next**。

不增加约束文件，所以 Add Constraints 窗口点击 **Next**。

按图 2-4 选择器件为 **xc7a100tffgg484-1**。点击 **Next**。

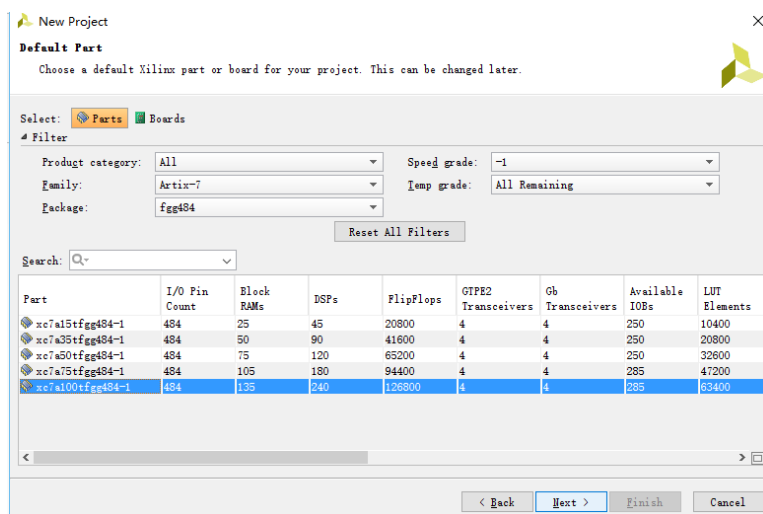


图 2-4 选择器件

可以看到如图 2-5 所示的新项目概览。点击 **Finish**。

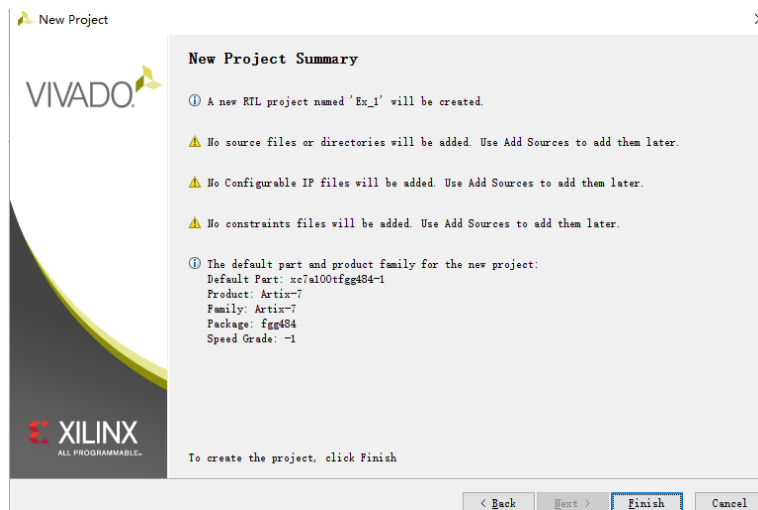


图 2-5 新项目概览

2. 添加源代码

在图 2-6 所示的窗口中右键点击 Design Sources（图中高亮处），在弹出的菜单中选择 Add Sources....，出现图 2-7 所示的对话框。

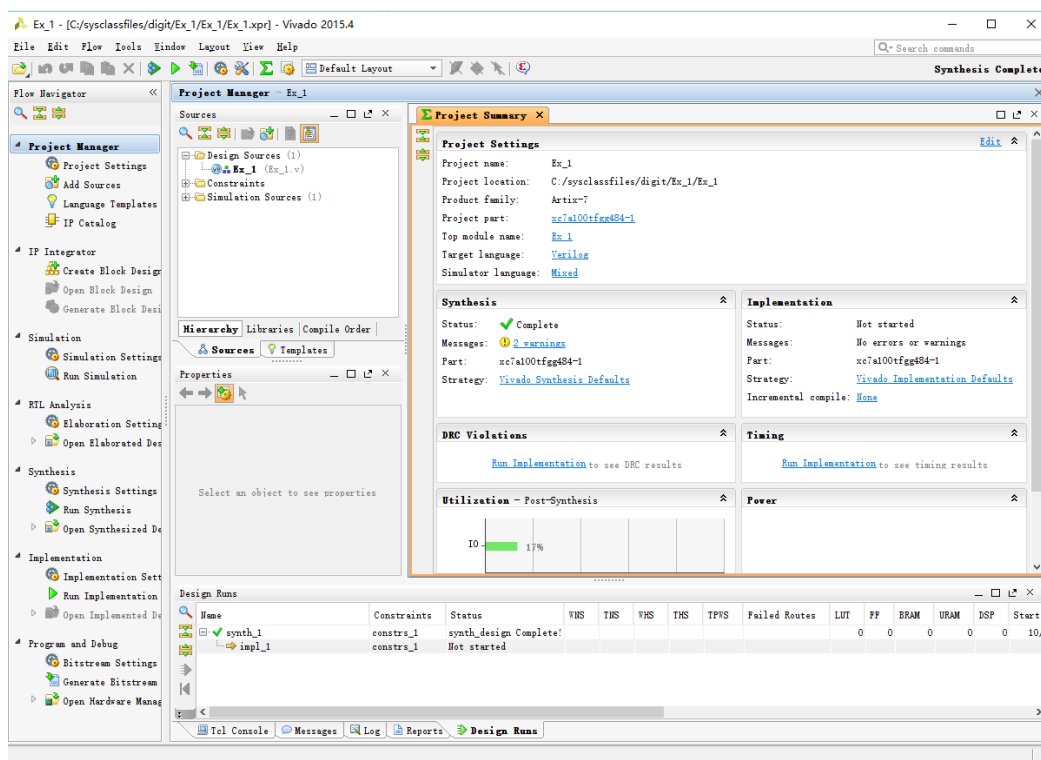


图 2-6 创建新项目后的界面

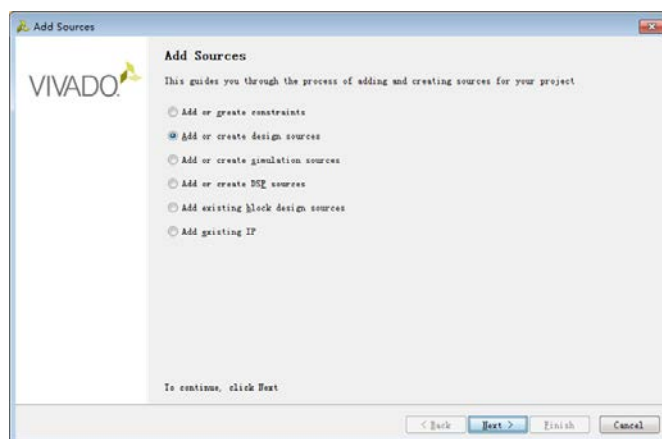



图 2-7 添加源程序对话框

按照图 2-7 所示选择后点击 Next。在接下来打开的对话框（如图 2-8 所示）中点击 , 并选择 Create File...。

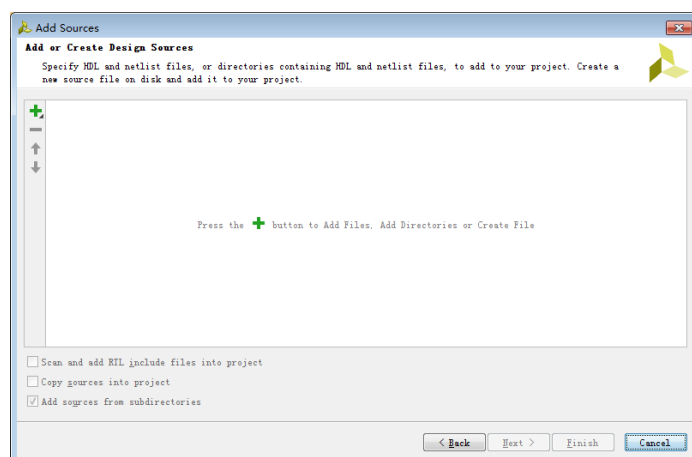


图 2-8 添加或创建设计文件对话框

此时会看到 Create Source File 对话框，按照图 2-9 所示填写后点击 OK。

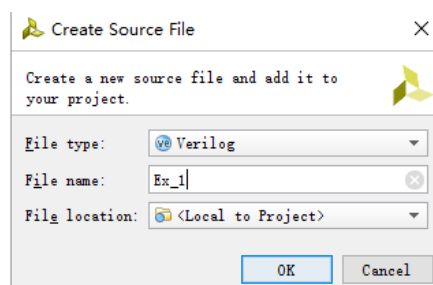


图 2-9 Create Source File 对话框

此时会看到图 2-10 所示的界面。

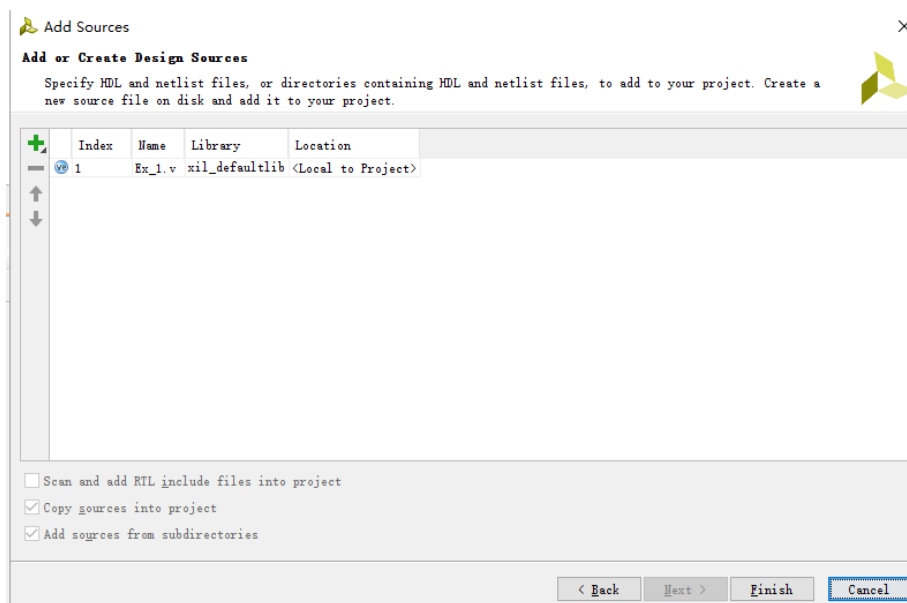


图 2-10 创建设计文件后的添加或创建设计文件对话框

点击 **Finish**。在接下来的 Define Module 对话框中如图 2-11 所示设置，然后点击 **OK**。

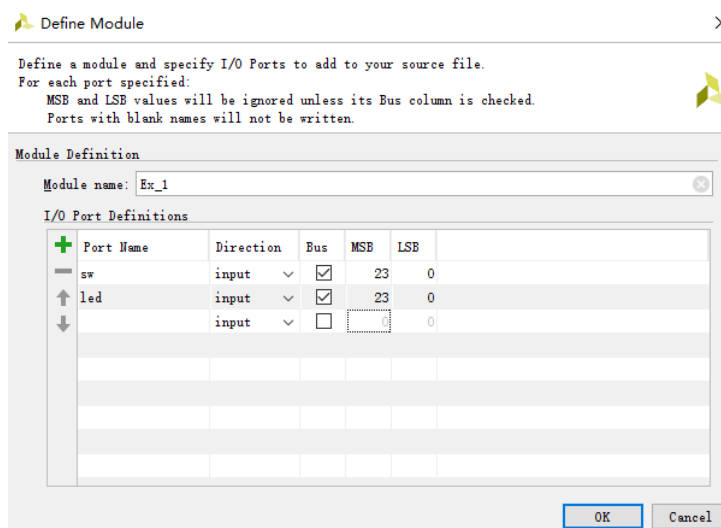


图 2-11 Define Module 对话框

接下来在图 2-12 所示界面上双击 Ex_1 文件（图中高亮部分）就会在右边显示初始的 Ex_1 文件内容。

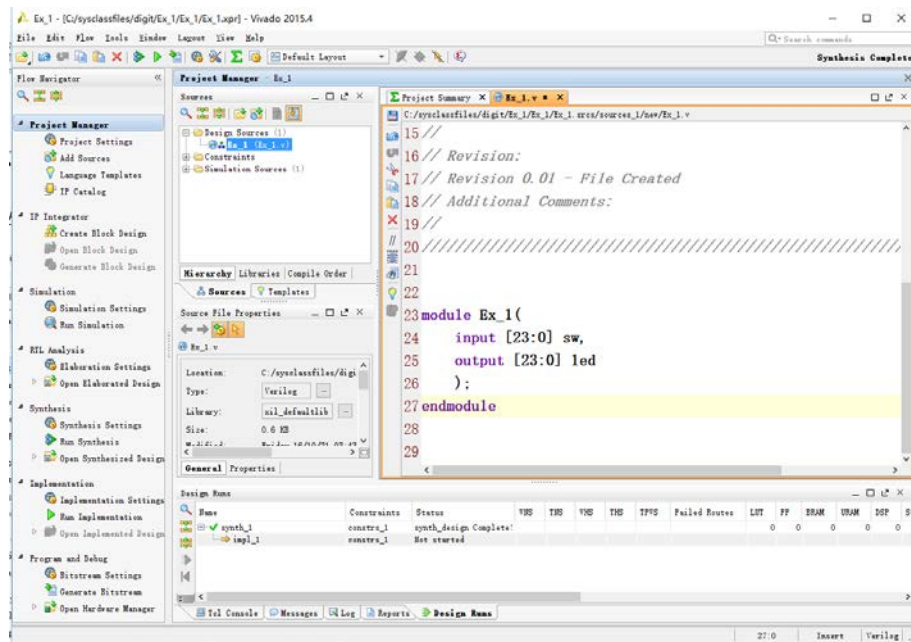


图 2-12 打开 Ex_1.v

可以看到文件中 Ex_1 的模块是空的。

```
module Ex_1(
    input [23:0] sw,
    output [23:0] led
);
endmodule
```

用下面的程序将这个空模块替代掉。

```
module Ex_1(
    input [23:0] sw,
    output [23:0] led
);
    assign led = sw;
endmodule
```

这个模块很简单，就是将拨码开关的内容赋值给 LED。

3. 仿真

以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 2-13 所示的设置点击 **Next**。

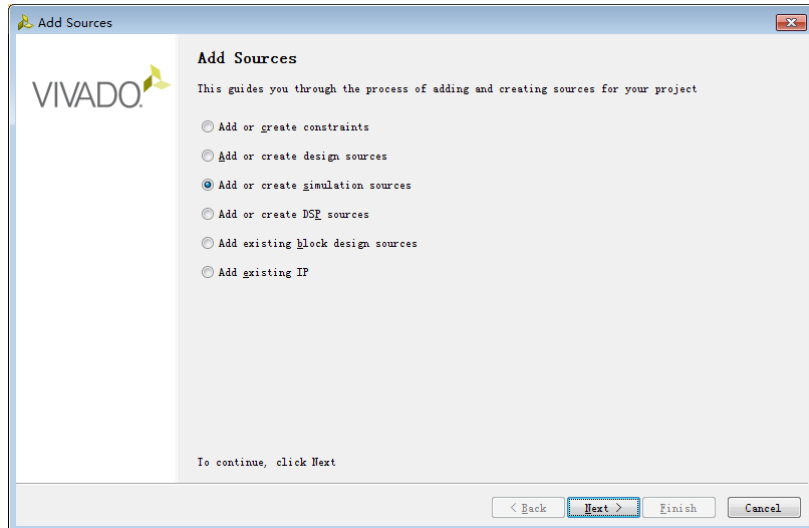


图 2-13 添加仿真源程序

在弹出的如图 2-14 的窗口中点击 , 并选择并选择 Create File...。

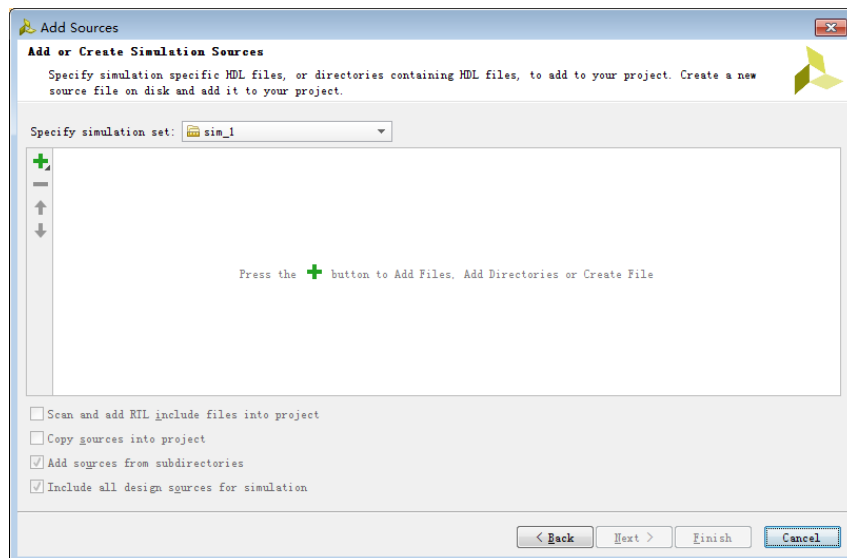


图 2-14 增加仿真源程序第二步-、Add Sources 对话框

在创建文件的窗口如图 2-15 那样设置仿真源文件的文件名为 Ex_1_sim, 并点击 **OK**。

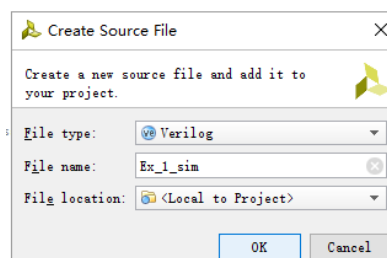


图 2-15 设置仿真源文件文件名

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 2-16) 点击 Yes。

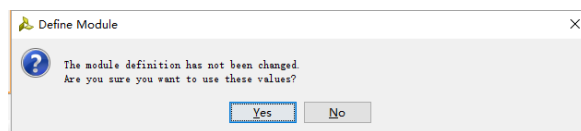


图 2-16 Define Module 窗口

如图 2-17 所示，现在在项目中看到了这个仿真源文件（图中高亮部分）。

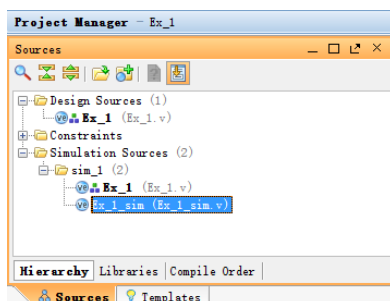


图 2-17 项目中的仿真源文件

双击图 2-17 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 [23:0] sw = 24'h000000;
    //output
    wire [23:0] 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+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。

保存好 Ex_1_sim.v 文件后，得到的项目文件层次如图 2-18 所示。

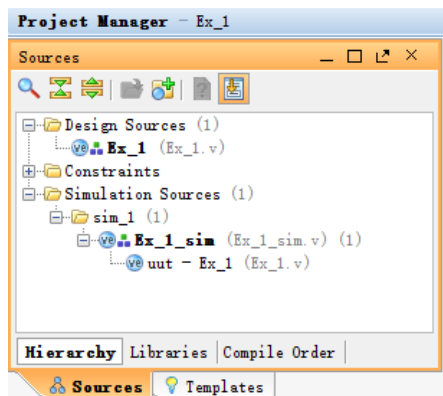


图 2-18 编辑完 Ex_1_sim.v 后的项目文件层次

在如图 2-19 所示的 Project Manager 中点击 Run Simulation，在弹出的菜单中选择 Run behavior Simulation。

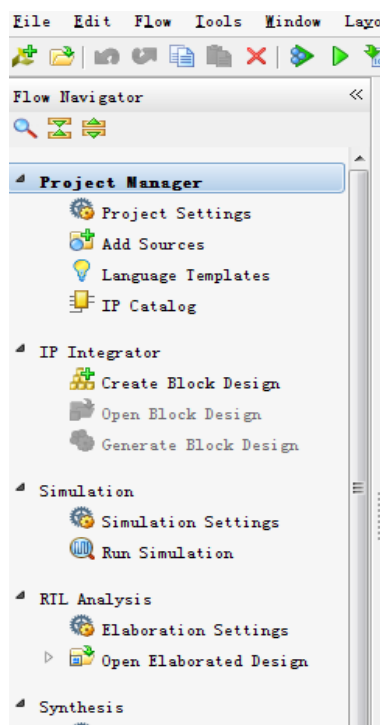


图 2-19 点击 Run Simulation

将工具条中的仿真时间调整为 10 us，点击 ，然后点击 开始仿真。

仿真完后，点击 得到如图 2-20 所示的波形图。

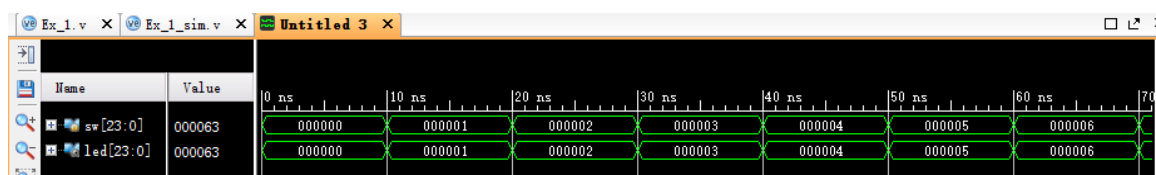



图 2-20 仿真波形图

从图上可以明显地看到，每隔 10n，输入数据 SW 加 1，输出数据 led 紧跟着变化

4. 综合

在 Project Manager 中点击 Run Synthesis 或者工具条上的按钮。综合如果没有问题，会弹出如图 2-21 所示的窗口。

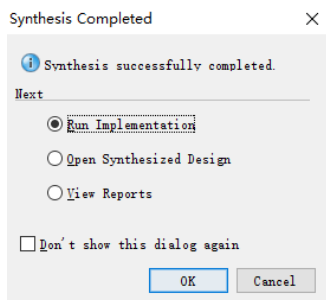


图 2-21 综合以后的窗口

由于还没有进行管脚分配，所以按照图 2-21 的选择，点击 OK，对设计进行管脚分配，如果已经管脚分配过了，那么图 2-21 中选择 Run Implementation 来运行实现。

5. 管脚分配

按照图 2-21 的选择，点击 OK 或在 Project Manager 中点击 Open Synthesized Design。有一个滚动条说明进度。滚动条结束后，选择菜单 Layout->I/O Planning。出现如图 2-22 所示的管脚分配表。

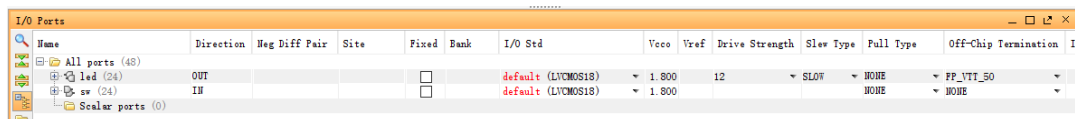


图 2-22 管脚分配表

先将 I/O Std 这一列全部改成 LVCMOS33。然后根据表 2-1，点击 site 对每个管脚进行分配。

表 2-1 Ex_1 管脚分配表

信号	部件	管脚	信号	部件	管脚
led[23]	RLD7	K17	sw[23]	SW23	Y9
led[22]	RLD6	L13	sw[22]	SW22	W9
led[21]	RLD5	M13	sw[21]	SW21	Y7
led[20]	RLD4	K14	sw[20]	SW20	Y8
led[19]	RLD3	K13	sw[19]	SW19	AB8
led[18]	RLD2	M20	sw[18]	SW18	AA8
led[17]	RLD1	N20	sw[17]	SW17	V8
led[16]	RLD0	N19	sw[16]	SW16	V9

led[15]	YLD7	M17	sw[15]	SW15	AB6
led[14]	YLD6	M16	sw [14]	SW14	AB7
led[13]	YLD5	M15	sw [13]	SW13	V7
led[12]	YLD4	K16	sw [12]	SW12	AA6
led[11]	YLD3	L16	sw [11]	SW11	Y6
led[10]	YLD2	L15	sw [10]	SW10	T6
led[9]	YLD1	L14	sw [9]	SW9	R6
led[8]	YLD0	J17	sw [8]	SW8	V5
led[7]	GLD7	F21	sw [7]	SW7	U6
led[6]	GLD6	G22	sw [6]	SW6	W5
led[5]	GLD5	G21	sw [5]	SW5	W6
led[4]	GLD4	D21	sw [4]	SW4	U5
led[3]	GLD3	E21	sw [3]	SW3	T5
led[2]	GLD2	D22	sw [2]	SW2	T4
led[1]	GLD1	E22	sw [1]	SW1	R4
led[0]	GLD0	A21	sw [0]	SW0	W4

管脚分配好后，如图 2-23 所示。

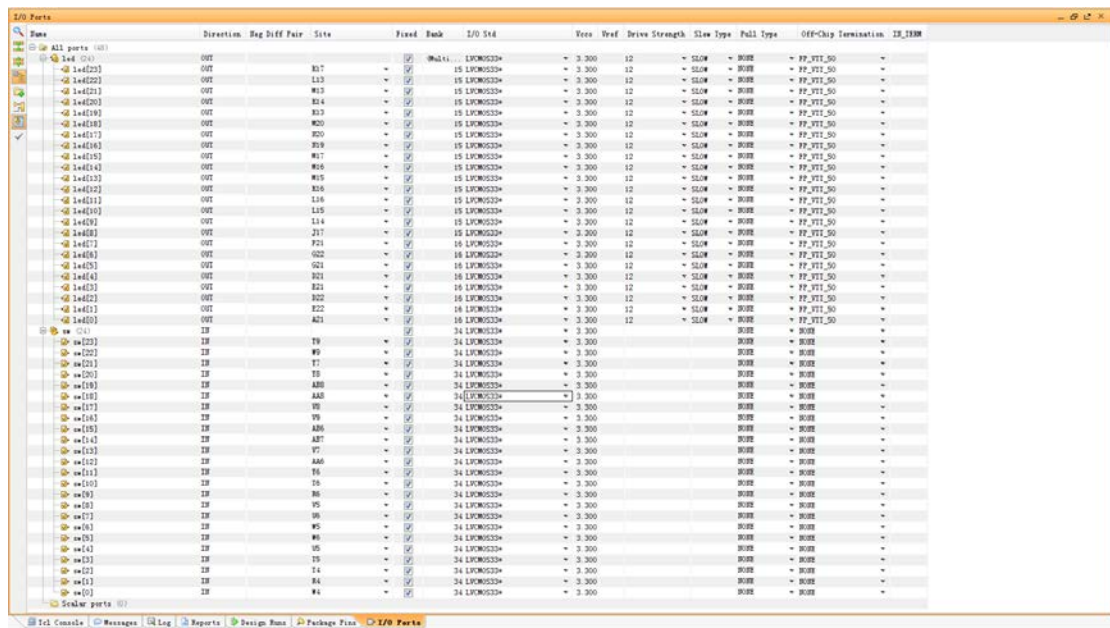


图 2-23 管脚分配后

按 Ctrl+S，出现如图 2-24 所示的窗口，点击 OK。



图 2-24 保存管脚分配设置

在出现的如图 2-25 所示的窗口中填写约束文件名为 **Ex_1**。

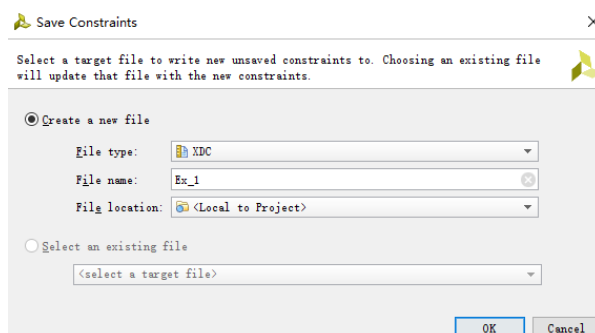


图 2-25 命名约束文件

点击 OK。然后选择 File->Close Synthesized Design 关闭综合设计窗口。

6. 实现

点击工具条中的  按钮或者在 Project Manager 中点击 Run Implementation 来对设计进行实现。

实现完后出现图 2-26 的窗口。

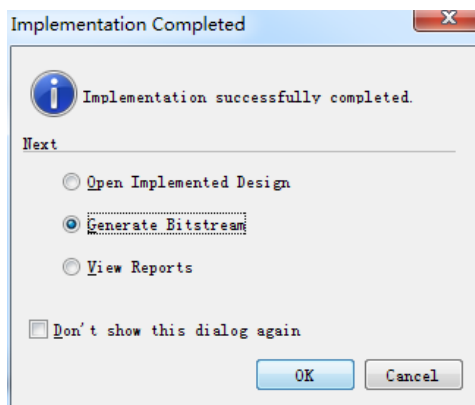



图 2-26 实现后的窗口

这样就完成了实现。

7. 产生比特流文件并下载

按照图 2-26 的设置点击 **OK**，或者在 Project Manager 中点击 Generate Bitstream 或者工具条上的  按钮。。比特流生成后会出现图 2-27 所示的对话框

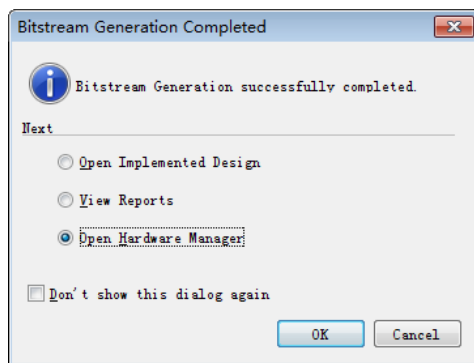


图 2-27 比特流生成完成

用 USB 下载线将 Minisys 板的 PROG UART 与 PC 机的 USB 相连，打开 Minisys 板的电源。按照图 2-27 所示选中 Open Hardware Manager。点击 **OK**。（在 Project Manager 中点击 Hardware Manager）。出现图 2-28 所示的界面。

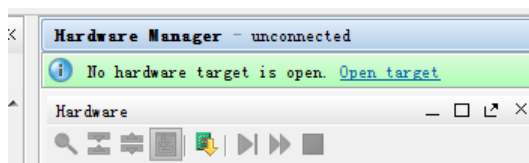


图 2-28 Hardware Manager

点击 Open target->Open new target。在弹出的窗口中点击 Next，再次点击 Next。此时出现图 2-29 所示的滚动条。

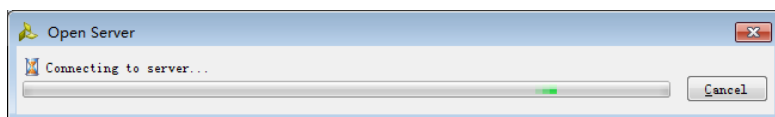


图 2-29 寻找硬件

硬件连接上后，会出现图2-30 所示界面。

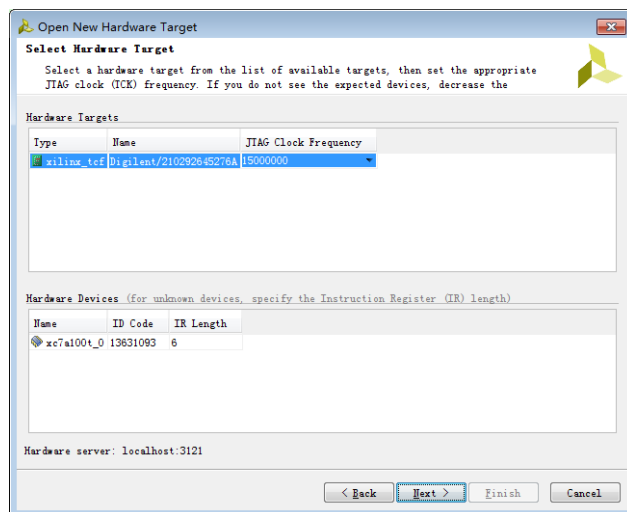


图 2-30 找到硬件

所列的正是接到 PC 上的实验板的主芯片。点击 Next，然后点击 Finish。

出现如图 2-31 所示界面。

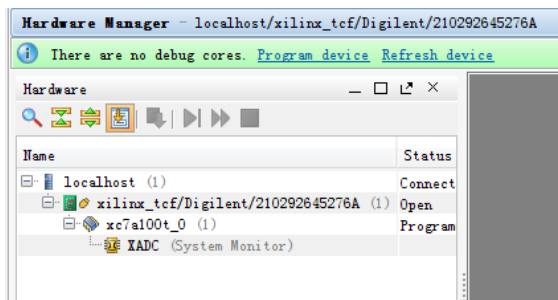


图 2-31 连接上硬件后

点击 Program devices->xc7a100t_0。出现的 Program Device 窗口（如图 2-32）中点击 Program。

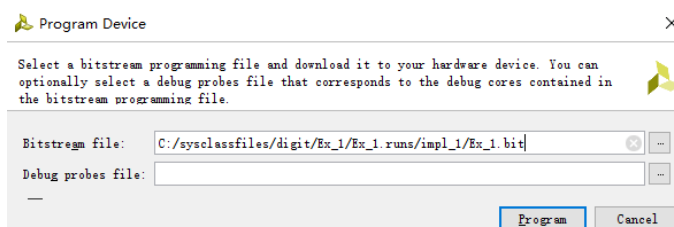


图 2-32 Program Device 窗口

出现如图 2-33 所示的正在下载的进度条。

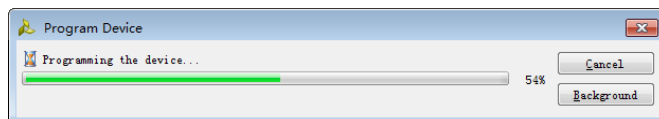


图 2-33 下载进度条

下载结束后，会看到 Minisys 板上的 LED 灯会随着拨码开关的变化而变化。

四、下板验证结果

使用 Class\solution\digit\Ex_1\Ex_1 文件夹中的工程文件下板验证。

将比特流文件下到板上后，随意拨动拨码开关，可以发现当相应的开关被拨上、拨下后，其对应的 LED 灯被点亮、熄灭。

2.1.2 可配置输入端口数和数据宽度的“与门”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 之间变化的与门 andgate，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。并将该与门封装成可配置输入端口数和数据宽度的“与门”IP 核。

三、实验步骤

1. 创建并仿真 andgate 项目

采用 2.1.1 节中的步骤创建 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;
  //outbut
  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
```

按照 2.1.1 中的仿真方法，可以得到图 2-34 所示的仿真波形。

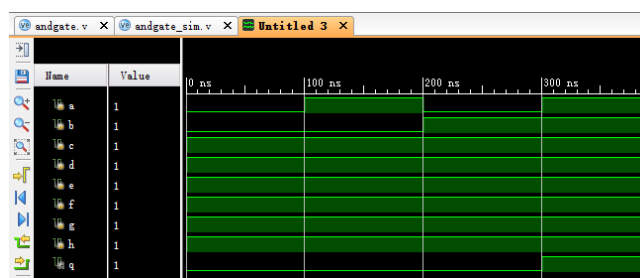


图 2-34 1 位 8 输入与门的仿真波形

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 2-34 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。

下面来仿真测试一下 32 位 2 输入与门的结果。建立如下的仿真文件。：

```
module andgate32_sim( );
// input
reg [31:0] a=32'h00000000;
reg [31:0] b=32'h00000000;
reg [31:0] c=32'hffffffff;
reg [31:0] d=32'hffffffff;
reg [31:0] e=32'hffffffff;
reg [31:0] f=32'hffffffff;
reg [31:0] g=32'hffffffff;
reg [31:0] h=32'hffffffff;
//outbut
wire [31:0] q;
// 实例化与门的时候，设定宽度为 32
andgate #(8,32) u(.a(a),.b(b),.c(c),.d(d),
                .e(e),.f(f),.g(g),.h(h),.q(q));
initial begin
#100 a=32'hffffffff;
#100 begin a=32'h00000000;b=32'hffffffff;end
#100 a = 32'h007fa509;
#100 a=32'hffffffff;
end
```

```
end
endmodule
```

按照 2.1.1 中的仿真方法，可以得到图 2-35 所示的仿真波形。

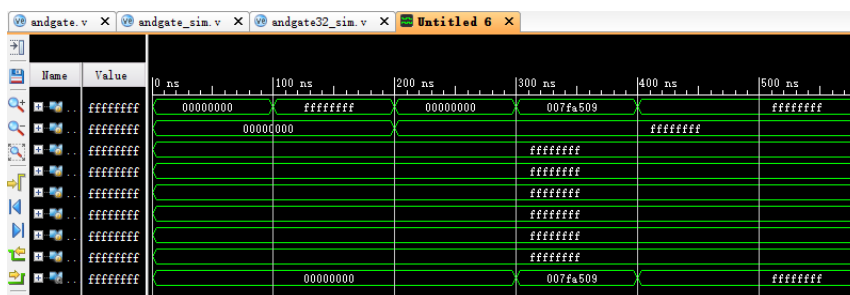


图 2-35 32 位 8 输入与门的仿真波形

从图 2-35 中可以看到结果是正确的。

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 2-36 所示的对话框，选择 Cancel。

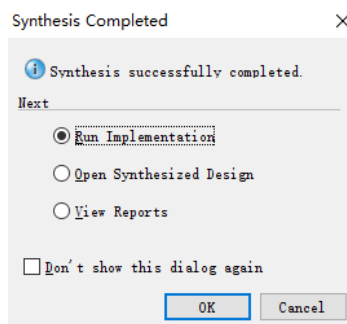


图 2-36 综合结束

在如图 2-37 所示的的界面中点击 **Project Settings**。

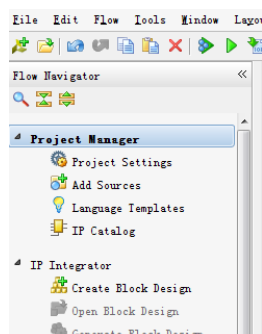


图 2-37 Project Manager

在 Project Settings 对话框中选择 IP，并进入 Packager 选项卡，如图 2-38 所示进行设置。设置好后，点击 Apply，然后点击 OK。记住这里设置的各个属性。

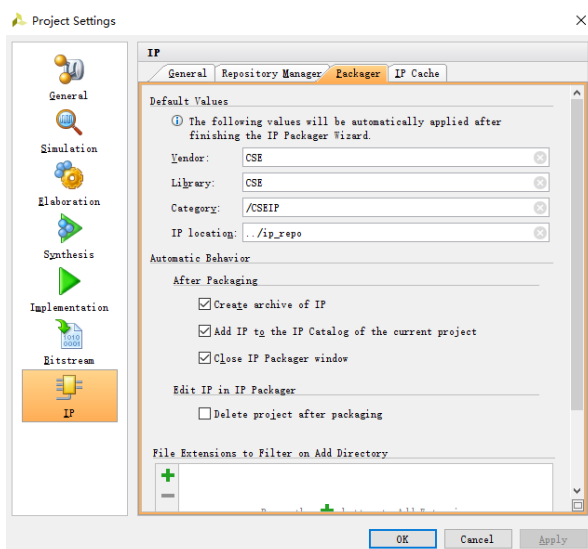


图 2-38 项目设置中设置 IP 核封装属性

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 2-39 所示设置封装选项。点击 Next。

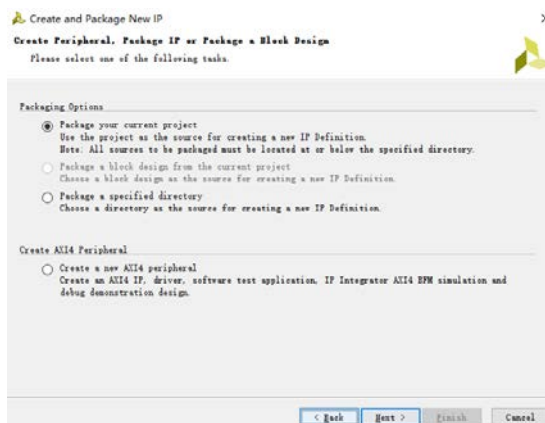


图 2-39 封装选项

在 IP Location（图 2-40）中不做修改，点击 Next。

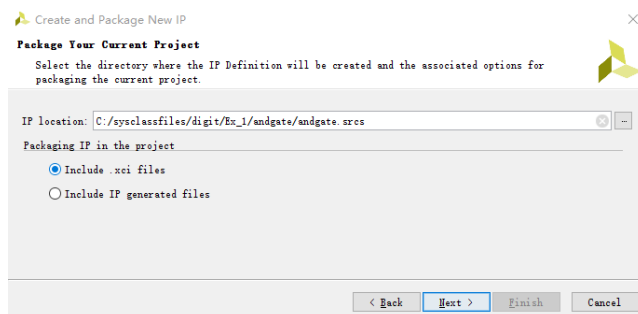


图 2-40 IP Location

封装后的 IP 放在了 C:/sysclassfiles/digit/Ex_1/andgate/andgate.srcs 这个文件夹中。

在图 2-41 所示的对话框中点击 **Finish**。

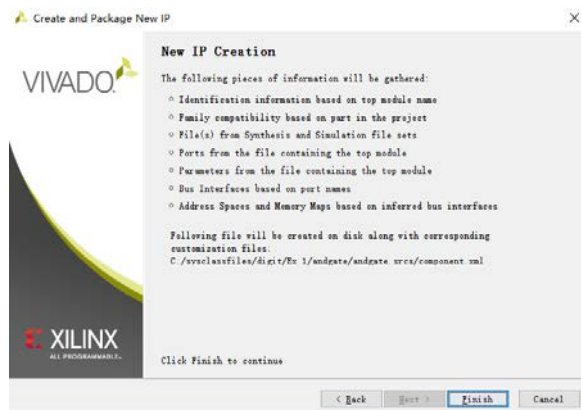


图 2-41 创建 IP 核结束

这时可以看到如图 2-42 所示的 IP 封装设置。按图 2-42 那样设置好各个项目。

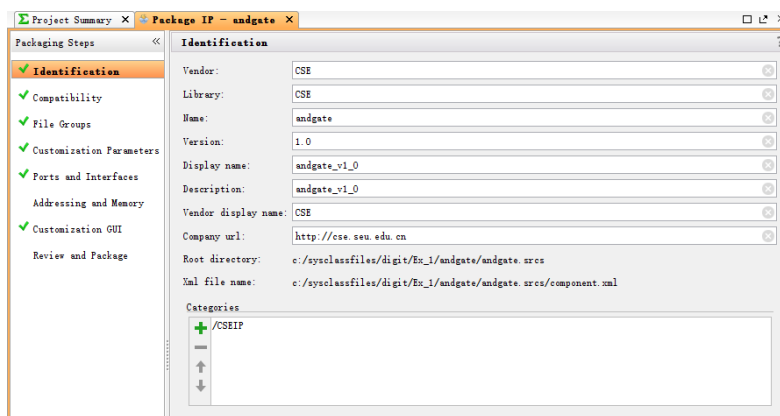



图 2-42 设置 Identification

在图 2-43 中可以添加 IP 核所支持的芯片家族。点击  并选择 Add Family Explicitly, 在如图 2-44 所示的 Add Family 对话框。

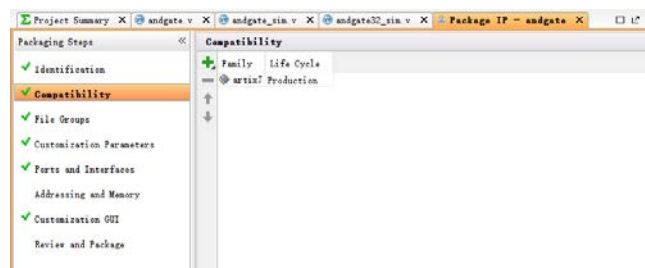


图 2-43 设置 compatibility



图 2-44 Add Family 对话框

在 Add Family 对话框中选中除 artix7 之外的所有芯片家族(因为 artix7 系列已经有了), Life-cycle 选择 Production, 然后点击 **OK**。这样就设置完了 compatibility。

接下来到 Customization Parameters (图 2-45 所示)。



图 2-45 Customization Parameters

双击图 2-45 中的 Port_Num 得到图 2-46 所示的 IP 和参数的对话框, 按照图 2-46 那样设置 Port_Num 参数后点 **OK**。

图 2-46 中可以看到, IP 核至少有两个输入端, 最多可以有 8 个输入端。

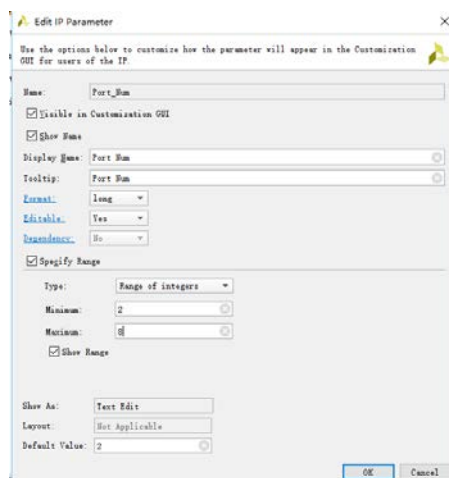


图 2-46 编辑 IP 核的 Port_Num 参数

双击图 2-45 中的 WIDTH，得到图 2-47 所示的 Edit IP Parameter 对话框，按照图 2-47 所示设置 WIDTH 参数后后点击 OK。

从图 2-47 图中可以看到，数据位宽 WIDTH 最小是 1 位，最大是 32 位。

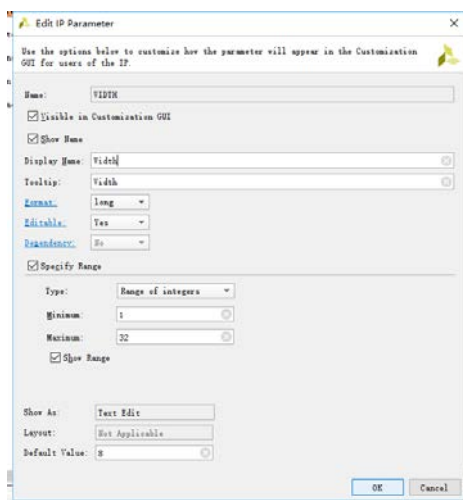


图 2-47 编辑 IP 核的 WIDTH 参数

接下来到 Ports and Interfaces，如图 2-48 所示。

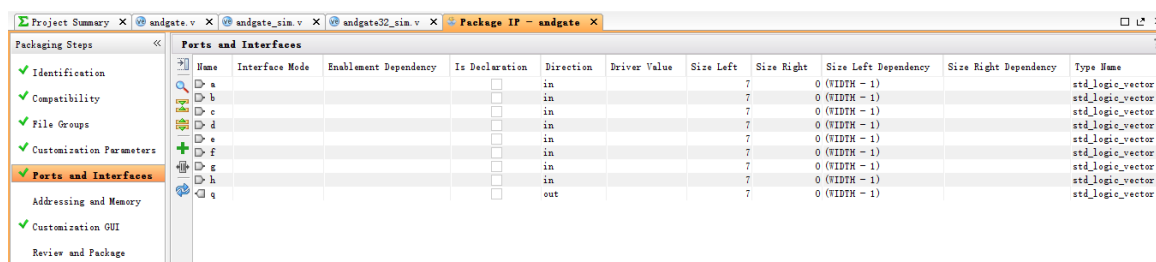


图 2-48 Ports and Interfaces 设置

由于设计的数据输入端最小是 2 个，因此，a,b 两个输入端时钟都是 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 2-49 所示设置。



图 2-49 编辑端口 c 的参数

图 2-49 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，需要给它赋一个驱动值。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 2-49 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 2-50 所示。

Name	Interface Mode	Enablement Dependency	Is Declaration	Direction	Driver Value	Size Left	Size Right	Size Left Dependency	Size Right Dependency	Type Name
a	in			in		7	0 (V125K - 1)			std_logic_vector
b	in			in	4294967295	7	0 (V125K - 1)			std_logic_vector
c	in	\$Port_Num > 2		in	4294967295	7	0 (V125K - 1)			std_logic_vector
d	in	\$Port_Num > 2		in	4294967295	7	0 (V125K - 1)			std_logic_vector
e	in	\$Port_Num > 4		in	4294967295	7	0 (V125K - 1)			std_logic_vector
f	in	\$Port_Num > 5		in	4294967295	7	0 (V125K - 1)			std_logic_vector
g	in	\$Port_Num > 6		in	4294967295	7	0 (V125K - 1)			std_logic_vector
h	in	\$Port_Num > 7		in	4294967295	7	0 (V125K - 1)			std_logic_vector
q	out			out	4294967295	7	0 (V125K - 1)			std_logic_vector

图 2-50 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 2-51 所示。



图 2-51 Review and Packaging 设置

可以看到 IP 生成到一个称为 CSE_CSE_andgate_1.0.zip 文件中，路径是 C:/sysclassfiles/digit/Ex_1/andgate/andgate.srcs。

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

为了方便今后的使用，请大家在 C:\sysclassfiles\digit\路径下新建一个文件夹 IPCore，并将 C:\sysclassfiles\digit\Ex_1\andgate\andgate.srcs\CSE_CSE_andgate_1.0.zip 文件拷贝到 C:\sysclassfiles\digit\IPCore 中，并将其解压缩。

2.1.3 多种基本门电路的IP核设计

一、实验目的

通过实验，进一步熟悉使用 vivado 进行可配置 IP 核设计的方法，对各种门电路的逻辑有更加感性的认识，同时也为今后的实验以及后续的课程积累基本的门电路 IP 核。

二、实验内容

仿照 2.1.2 节的设计方法，使用 Verilog HDL 语言的数据流描述法设计下列各种 1~32 位数据宽度可变的基本门电路，出了非门外，其他门电路还要求输入端口了在 2-8 之间变化，最后将它们分别封装成 IP 核。一定要注意不同门电路的各个端口在 disable 时候的取值 Driver value 的设定，要用一个合理的值。

需要完成的 IP 核及相关参数如表 2-2 所示。

表 2-2 基本门电路及参数表

门电路	模块名	参 数			
		Port_Num	WIDTH	输 入	输 出
或门	orgate	2~8	1~32	a,b,c,d,e,f,g,h	q
非门	notgate	-	1~32	a	c
与非门	nandgate	2~8	1~32	a,b,c,d,e,f,g,h	q
或非门	norgate	2~8	1~32	a,b,c,d,e,f,g,h	q
异或门	xorgate	2~8	1~32	a,b,c,d,e,f,g,h	q
异或非门	nxorgate	2~8	1~32	a,b,c,d,e,f,g,h	q

在 X:\sysclassfiles\digit\Ex_1 的相关目录下给出的配套实验文件中给出了初始设计文件和仿真文件，供大家完善和使用。

表 2-2 中的各个门电路的 IP 核封装好后，统一拷贝到 C:\sysclassfiles\digit\IPCore 中，并分别解压这些.zip 文件，如图 2-52 所示。

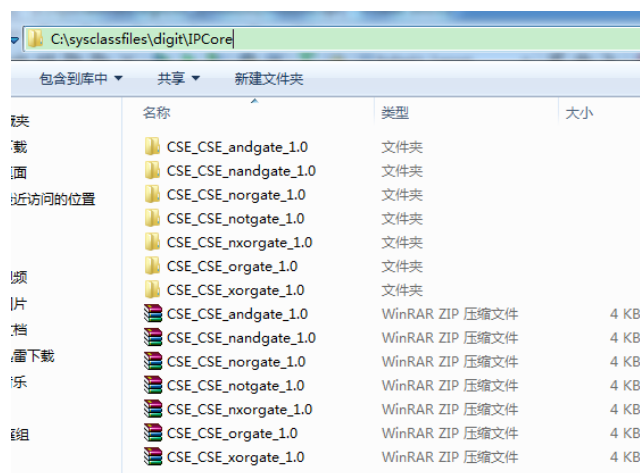


图 2-52 封装好的 IP 核

2.1.4 74 系列基本逻辑门电路芯片的设计

一、实验目的

1. 通过实验，让学生学会自定义 IP 核的使用；
2. 对 74 系列几个基本逻辑门电路芯片内部结构加深了解。
3. 进一步了解 Verilog HDL 的三种描述方法。

二、实验内容

通过对 2.1.2 和 2.1.3 节封装的 IP 核的使用，利用原理图（Block Design）法、Verilog 结构化描述方法、数据流描述法和行为描述法设计 7400 芯片，比较一下这几种方法设计的不同和优缺点。采用上述方法之一设计 7404，7420 和 7486 芯片，通过仿真验证自己设计的正确性。

三、实验步骤

1. 用 Block Design 设计 7400 芯片

7400 芯片是四 2 输入与非门，图 2-53 是 7400 的结构示意图，表 2-3 是 7400 真值表。

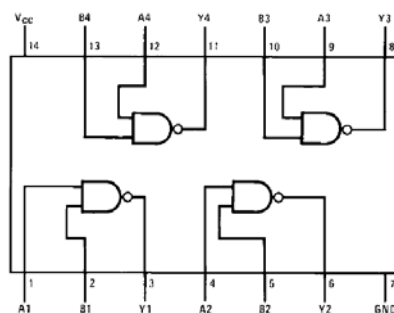


图 2-53 7400 芯片结构示意图

表 2-3 7400 真值表

输入		输出 •
A	B	Y
0	0	1
0	1	1
1	0	1
1	1	0

1) 导入 IP 核

在 C:/sysclassfiles/digit/Ex_1 文件夹中创建一个新的项目 S7400。在界面左侧的 Project Manager 中点击 Project Settings，打开 Project Settings 对话框，并转到 IP 项上。如图 2-54 所

示。

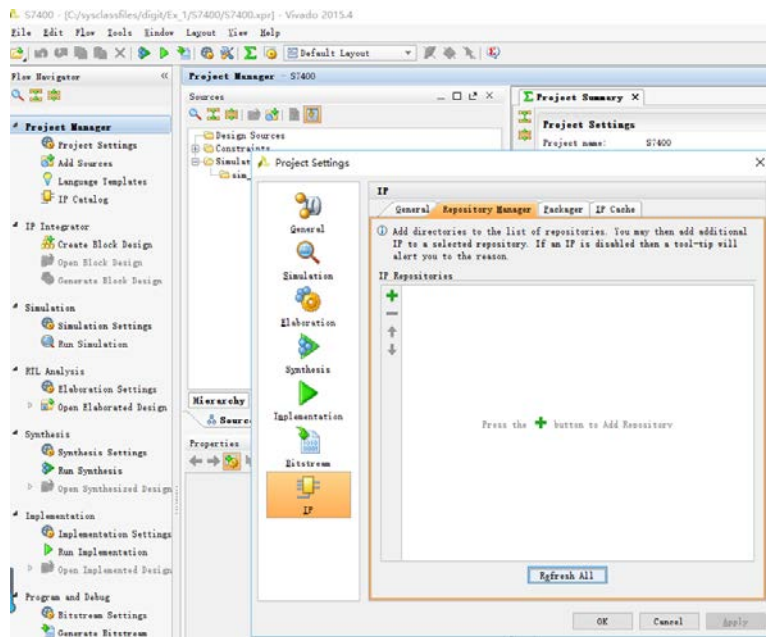


图 2-54 项目设置中添加 IP 核

点击 **+**，找到下面的目录：C:\sysclassfiles\digit\IPCore。如图 2-55 所示，选择 IPCore 的文件夹，点击 Select。

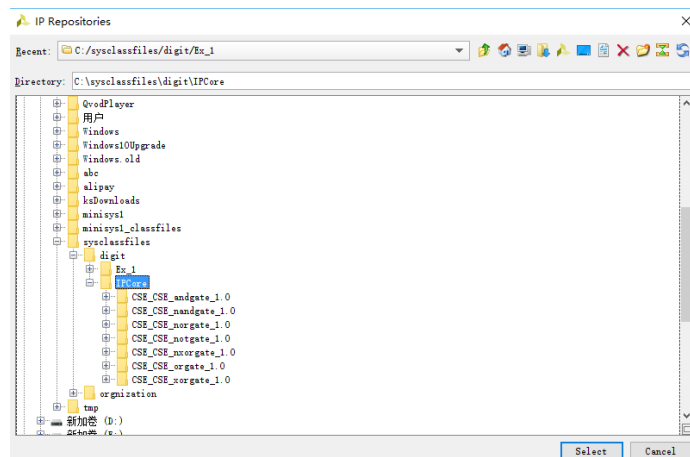


图 2-55 选择 IP 核的位置

系统弹出如图 2-56 所示的 Add Repository 对话框，检查一下，确实是 7 个 IP 核，点击 **OK**。

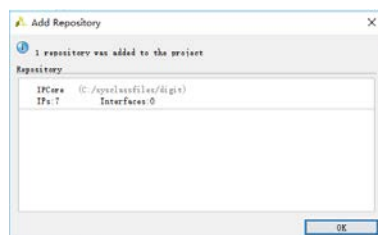


图 2-56 Add Repository 对话框

现在回到了如图 2-57 所示的 Project Settings 对话框。点击 Apply，然后点击 OK。

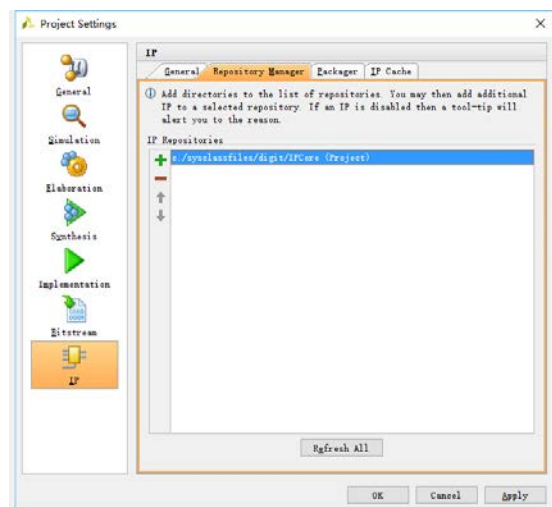


图 2-57 导入 IP 核后的 Project Settings 对话框

现在点击 Project Manager 下的 IP Catalog，就会看到如图 2-58 所示的界面中，IP Catalog 里已经有了自己设计的 7 个 IP 核。

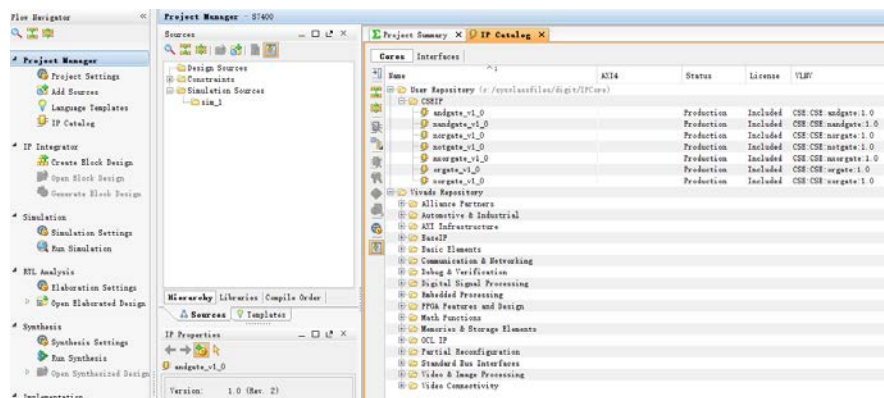


图 2-58 IP Catalog 中的 7 个基本部件

2) 创建 bd 设计文件

点击 Project Manager 下的 Create Block Design，打开图 2-59 所示的对话框，按照图 2-54 设置后点击 OK。

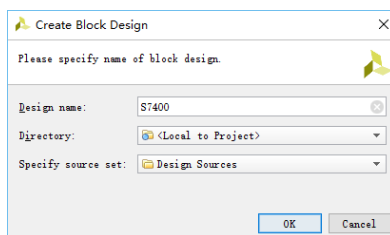


图 2-59 创建 bd 文件

现在得到了图 2-60 所示的界面

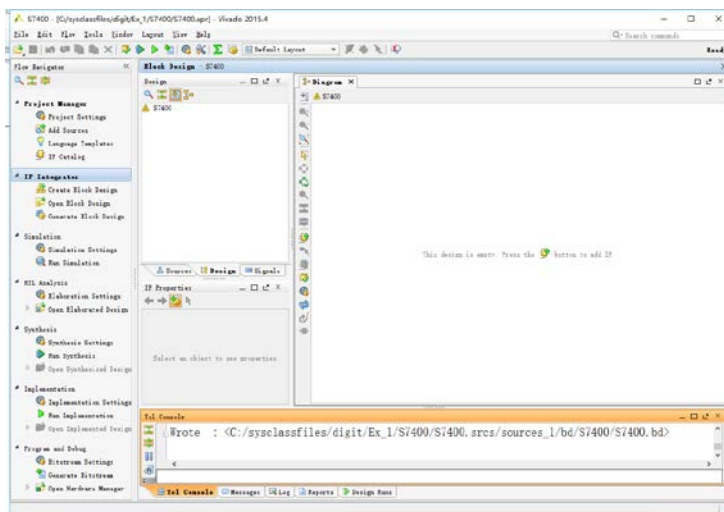


图 2-60 创建了 S7400.bd

3) 放置基本门电路

现在点击 Project Manager 下的 IP Catalog, 在如图 2-61 所示的窗口中双击 nandgate_v1_0 (图 2-61 中高亮部分)。

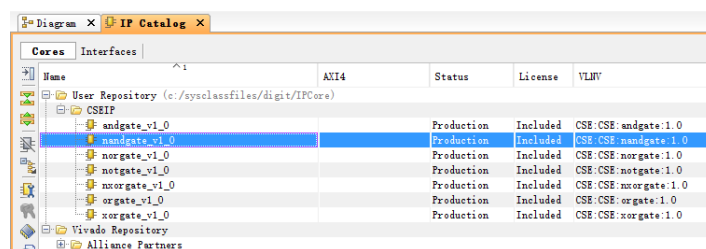


图 2-61 选择与非门 IP 核

在出现的如图 2-62 所示的对话框中选择 **Add IP to Block Design**。



图 2-62 添加 IP 核

此时界面上出现了所选择的 IP 核心, 如图 2-63 所示。

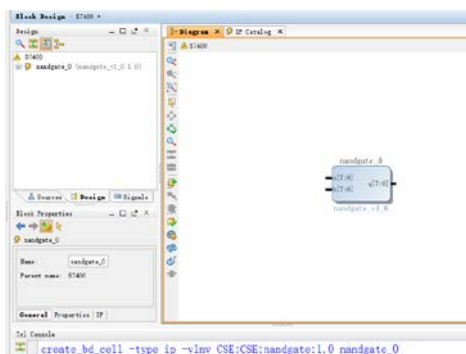


图 2-63 放置一个与非门后

左键点击图 2-63 中的 nandgate_0 器件，选中它，然后右键点击它，并在弹出的菜单中选择 **Customize Block...**（或者双击 nandgate_0 器件）就会打开图 2-64 所示的窗口。按照图中设置 Port_Num 为 2，Width 为 1，然后点击 **OK**。

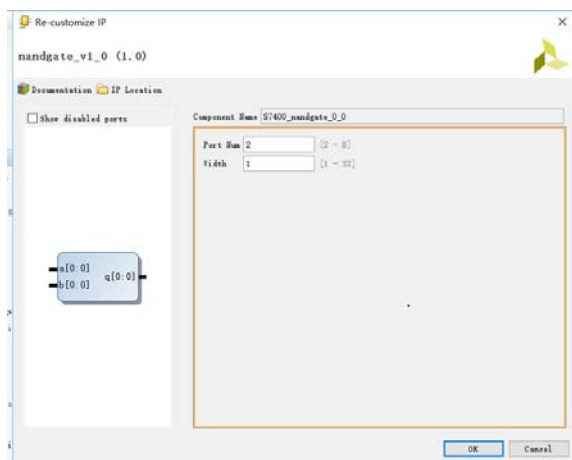


图 2-64 设置与非门数据宽度为 1

这样就放置好了一个与非门，按照这个方法再放置另三个与非门，并设置好他们的数据宽度都是 1，数据端口数都为 2。放置好后，将他们的位置移动到如图 2-65 所示。

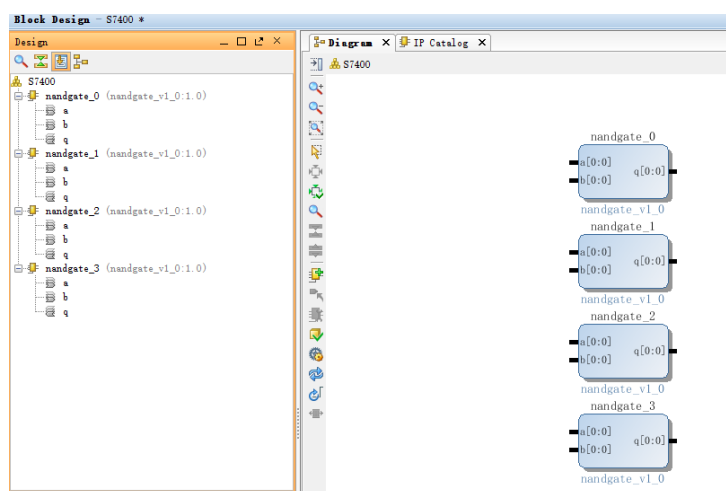


图 2-65 放好位置的四个基本门电路

4) 放置输入输出端口

在图 2-65 中空白处右键点击，在弹出的菜单中选择 **Create Port...**。打开如所示的对话框，按照图 2-66 所示设置，就添加立刻输入端 A1。

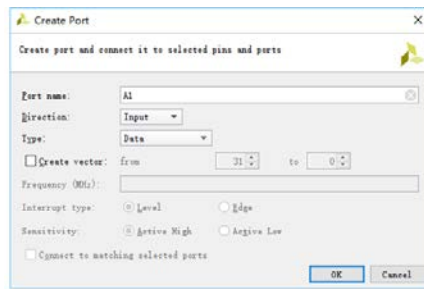


图 2-66 添加输入端 A1

按照上述方法在增加输入端 A2, A3, A4, B1, B2, B3, B4 和输出端 Y1, Y2, Y3, Y4。并按照图 2-67 所示连接好电路

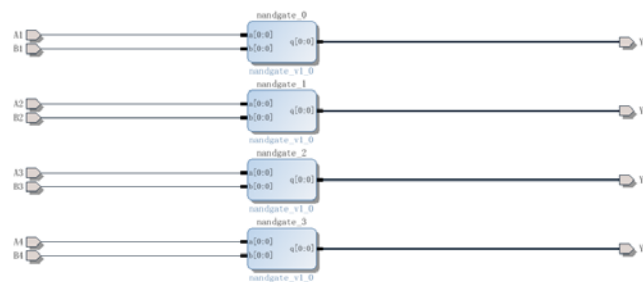


图 2-67 连接好的电路

5) 仿真实证

按照前面章节的办法，建立以下仿真文件。

```
`timescale 1ns / 1ps
module s7400_sim( );
    // INPUT
    reg A1=0,B1=0,A2=1,B2=0,A3=0,B3=1,A4=1,B4=1;
    //OUTPUT
    wire Y1,Y2,Y3,Y4;
    S7400 U1(.A1(A1),.A2(A2),.A3(A3),.A4(A4),
              .B1(B1),.B2(B2),.B3(B3),.B4(B4),
              .Y1(Y1),.Y2(Y2),.Y3(Y3),.Y4(Y4));
    initial begin
        #100 begin A1=1;B1=1;A2=0;B2=1;A3=1;B3=1;A4=0;B4=0; end
    end
endmodule
```

仿真后得到如图 2-68 所示的仿真结果，证明设计的逻辑是正确的,同时也说明与非门、或门和非门的 IP 核是可用的。

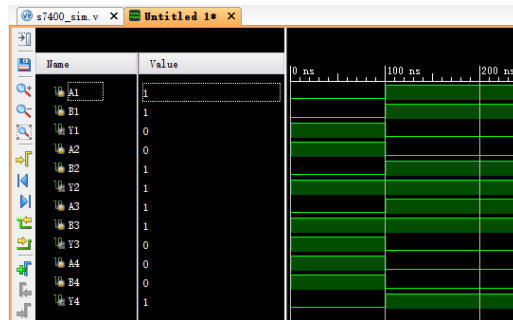


图 2-68 S7400 仿真结果

6) 生成输出文件

在 Project Manager 的 Source 界面中，右键点击 S7400，在弹出的菜单中选择 Generate Output Products...，之后在弹出的如图 2-69 所示的 Generate Output Products 对话框中选择 Global，点击 Generate。

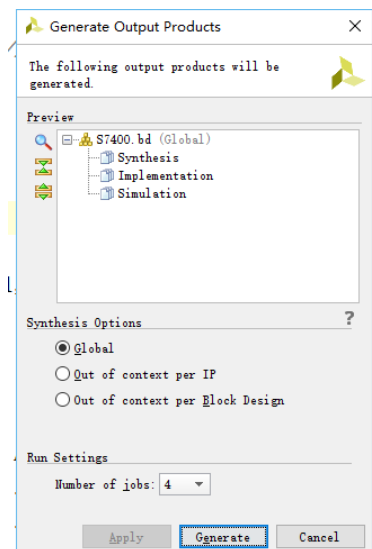


图 2-69 生成输出文件

7) 生成 HDL 文件

再次在 Project Manager 的 Source 界面中，右键点击 S7400，在弹出的菜单中选择 Create HDL Wrapper...，之后在弹出的如图 2-70 所示的 Create HDL Wrapper 对话框中选择 Let Vivado Manage wrapper and auto-update，点击 OK。

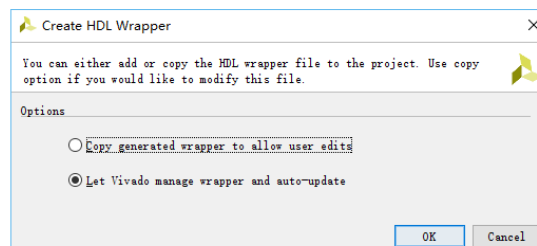


图 2-70 生成 HDL 文件

8) 进行管脚约束文件配置、综合、实现、生成比特流文件，下载到板子上进行验证。

7400 的管脚分配如表 2-4 所示

表 2-4 7400 管脚分配表

信号	部件	管脚	信号	部件	管脚
Y4	GLD3	E21	A4	SW3	T5
Y3	GLD2	D22	A3	SW2	T4
Y2	GLD1	E22	A2	SW1	R4
Y1	GLD0	A21	A1	SW0	W4
B4	SW7	U6			
B3	SW6	W5			
B2	SW5	W6			
B1	SW4	U5			

在 X:\sysclassfiles\digit\Ex_1\S7400 下给出的配套实验文件中给出了仿真文件和约束文件，供大家使用。

2. 用 Verilog 语言设计 7400 芯片

1) 用结构化描述方法

a) 在 C:\sysclassfiles\digit\Ex_1 文件夹中创建一个新的项目 S7400_v1。

b) 导入 IP 核。

c) 点击 Project Manager 下的 IP Catalog，在如图 2-61 所示的窗口中双击 nandgate_v1_0（图 2-61 中高亮部分），就会打开图 2-64 所示的窗口。按照图中设置 Port_Num 为 2，Width 为 1，然后点击 **OK**。弹出如图 2-71 所示的对话框，点击 **Generate**。在弹出的对话框中点击 **OK**。可以看到与非门 IP 核 nandgate_0 被加入到项目中，如图 2-72 所示。

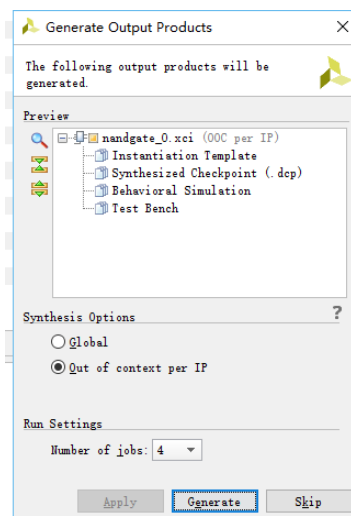


图 2-71 生成所需 IP 核心

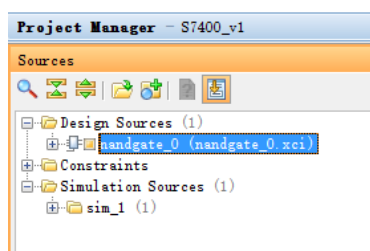


图 2-72 进到项目中的与非门 IP 核

d) 用 c)的方法,再向项目中添加三个与非门的 IP 核,分别称为 nandgate_1, nandgate_2, 和 nandgate_3。如图 2-73 所示。

e) 建立结构化描述的 S7400_v1.v 设计文件。

```
module S7400_v1(
    input  A1,
    input  B1,
    output Y1,
    input  A2,
    input  B2,
    output Y2,
    input  A3,
    input  B3,
    output Y3,
    input  A4,
    input  B4,
    output Y4
);
    nandgate_0 U1( .a(A1), .b(B1), .q(Y1));
    .....
endmodule
```

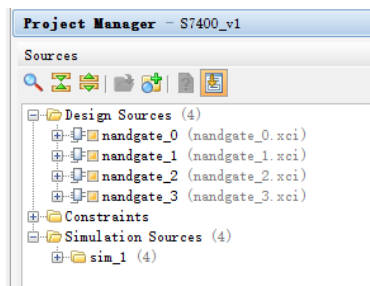


图 2-73 引入四个与非门 IP 核到项目中

省略号的部分请大家自行补充完整。

f) 使用仿真文件仿真, 仿真文件可参考 S7400_sim 的仿真文件。

g) 进行管脚约束文件配置、综合、实现、生成比特流文件, 下载到板子上进行验证。

管脚分配表如表 2-4 所示。

在 X:\sysclassfiles\digit\Ex_1\S7400_v1 下给出的配套实验文件中给出了初始设计文件、仿真文件和约束文件，供大家完善和使用。

2) 用数据流描述方法

用数据流描述方法不需要导入 IP 核，因为这种方法是直接用逻辑表达式产生输出，不需要元件例化 IP 核。

在 C:\sysclassfiles\digit\Ex_1 文件夹中创建一个新的项目 S7400_v2。设计文件如下：

```
module S7400_v2(
    input  A1,
    input  B1,
    output Y1,
    input  A2,
    input  B2,
    output Y2,
    input  A3,
    input  B3,
    output Y3,
    input  A4,
    input  B4,
    output Y4
);
    assign Y1 = ~(A1 & B1);
    .....
endmodule
```

省略号的部分大家自行补充完整，然后完成对设计的仿真和实现。

在 X:\sysclassfiles\digit\Ex_1\S7400_v2 下给出的配套实验文件中给出了初始设计文件、仿真文件和约束文件，供大家完善和使用。

3) 用行为描述方法

用行为描述方法也不需要导入 IP 核。

在 C:\sysclassfiles\digit\Ex_1 文件夹中创建一个新的项目 S7400_v3。设计文件如下：

```
module S7400_v3(
    input A1,
    input B1,
    output reg Y1,
    input A2,
    input B2,
    output reg Y2,
    input A3,
```

```

    input B3,
    output reg Y3,
    input A4,
    input B4,
    output reg Y4
);
always @(*)
begin
    Y1 <= ~(A1 & B1);
    .....
end
endmodule

```

省略号的部分大家自行补充完整，然后完成对设计的仿真和实现。

在 X:\sysclassfiles\digit\Ex_1\S7400_v3 下给出的配套实验文件中给出了初始设计文件、仿真文件和约束文件，供大家完善和使用。

3. 74 系列其他基本逻辑门电路芯片的设计

1) 7404 芯片的设计

7404 芯片是 6 反相器，图 2-74 给出了 7404 的结构示意图，表 2-5 给出了 7404 的真值表。

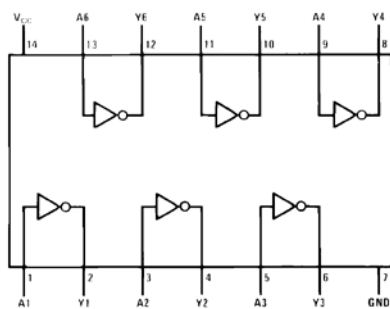


图 2-74 7404 结构示意图

表 2-5 7404 真值表

输入	输出
A	Y
0	1
1	0

请大家用 Verilog 语言或者 Block Design 的方法中选一种，设计和实现 7404 芯片。

2) 7420 芯片设计

7420 芯片是二 4 输入与非门，图 2-75 给出了 7420 的结构示意图。表 2-6 给出了 7420 的真值表。

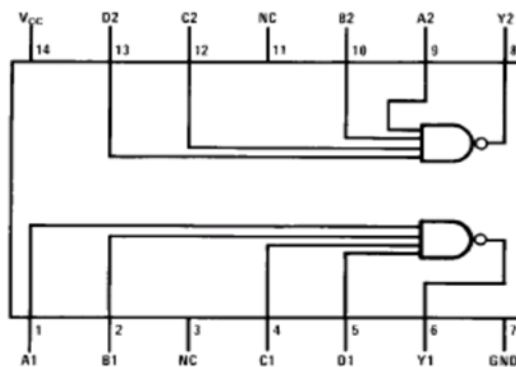


图 2-75 7420 结构示意图

表 2-6 7420 真值表

输入				输出
A	B	C	D	Y
X	X	X	0	1
X	X	0	X	1
X	0	X	X	1
0	X	X	X	1
1	1	1	1	0

请大家用 Verilog 语言或者 Block Design 的方法中选一种，设计和实现 7420 芯片。

2) 7486 芯片设计

7486 芯片是四 2 输入异或门，图 2-76 给出了 7486 的结构示意图。表 2-7 给出了 7486 的真值表。

请大家用 Verilog 语言或者 Block Design 的方法中选一种，设计和实现 7486 芯片。

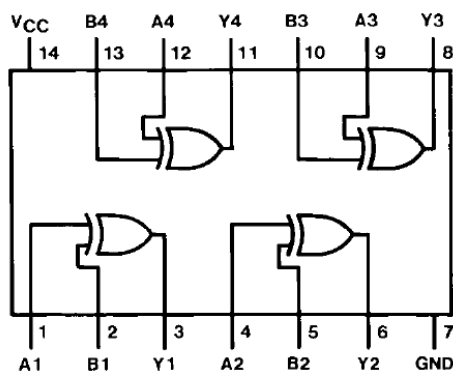


图 2-76 7486 结构示意图

表 2-7 7486 真值表

输入		输出
A	B	Y
0	0	0
0	1	1
1	0	1
1	1	0

三、下板验证结果

使用 class\solution\digit\Ex_1\S7400 中的工程文件下板验证。

- 1) 依次拨动 SW4、SW0（即 B1、A1）为 00、01、10 和 11，只有当 A1、B1 同时为 1 时，GLD0 熄灭（为 0），其他情况 GLD0 都点亮（为 1），这与与非运算的结果相一致；
- 2) 对开关 SW5、SW1，SW6、SW2，SW7、SW3 重复 1) 种操作，分别观察 GLD1、GLD2、GLD3 的变化情况。

S7400_v1、S7400_v2 和 S7400_v3 的验证操作和结果与 S7400 是相同的。