

## 一． 基于 Vivado 的数字逻辑电路设计流程

下面以共阳极七段数码管显示译码器为例，讲解基于 Xilinx Vivado 的数字逻辑电路设计流程。

### 1. 创建工程



(1). 双击桌面 Vivado 图标，启动 Vivado 2018.2，如图 1-1 所示。点击 Create Project，或在菜单栏中选择 File - New Project，即可新建工程。

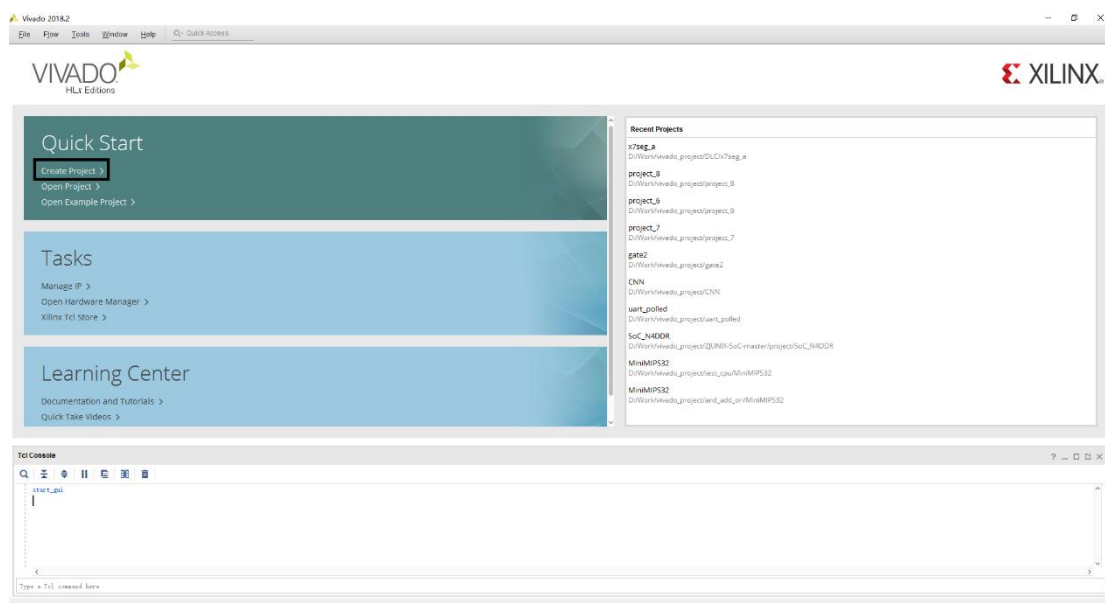


图 1-1 Vivado 开始窗口

(2). New Project 新建工程向导打开，如图 1-2 所示，直接点击 Next。

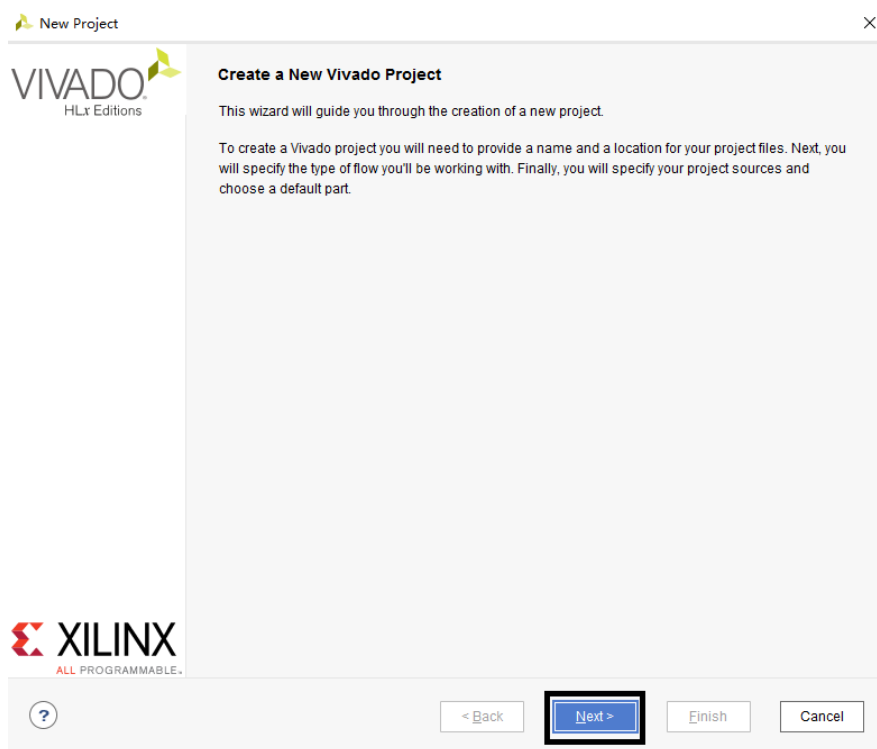


图 1-2 New Project 窗口

(3). 在 Project name 文本框中, 输入 x7seg\_a 作为工程名; 在 Project location 文本框中选择 D:/Work/vivado\_project/DLC 作为工程路径(工程路径可自行定义); 勾选 Create project subdirectory 在工程路径下生成一个名为 x7seg\_a 的文件夹, 用于保存该工程所有文件。如图 1-3 所示, 点击 Next 进入下一步。

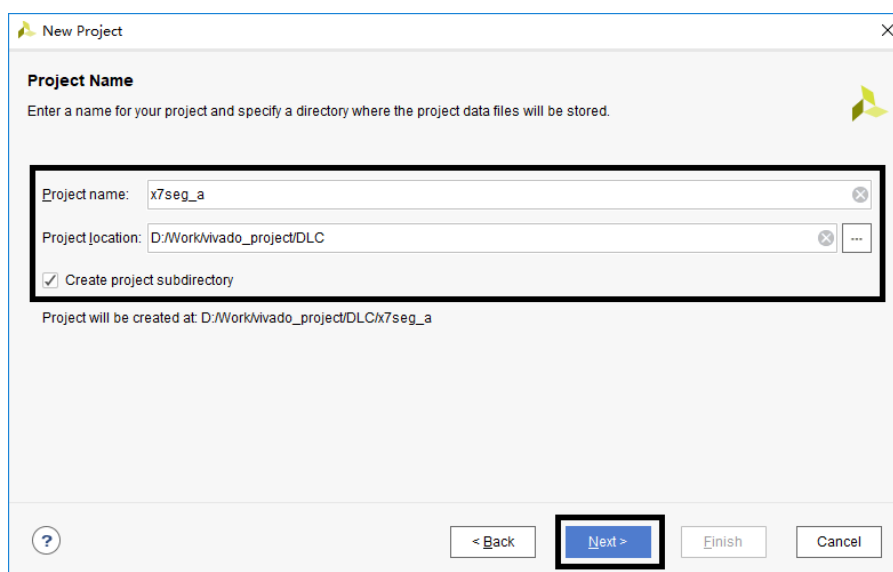


图 1-3 Project Name 窗口

(4). 在 Project Type 窗口中, 选择 RTL Project, 表示将创建基于 RTL 的工程。然后, 勾选 Do not specify sources at this time 选项 (这样将跳过添加源文件的步骤, 源文件可以在后面设计过程中再添加, 如果希望在工程创建的同时添加源文件, 则不勾选该选项) 如图 1-4 所示, 点击 Next 进入下一步。

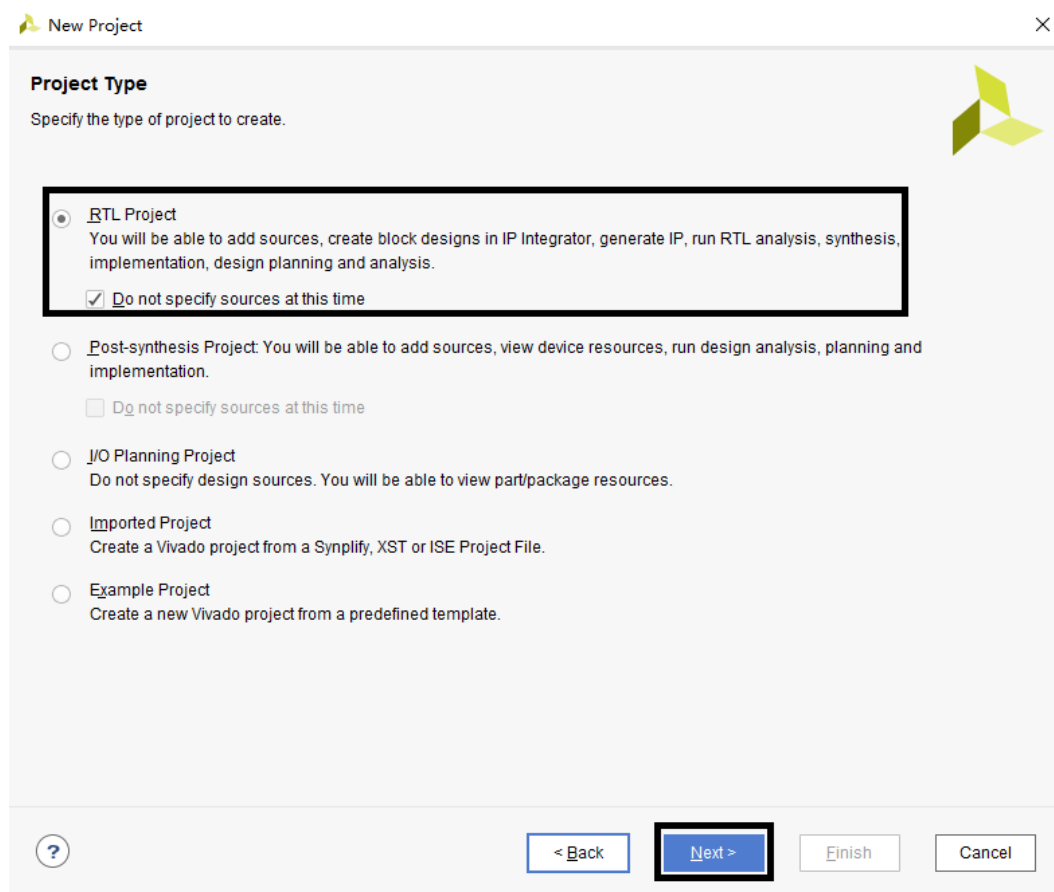


图 1-4 Project Type 窗口

(5). 在 Default Part 窗口中, 选择目标 FPGA 器件型号, 如图 1-5 所示。根据远程硬件平台上的 FPGA 型号 (xc7a35tftg256-1), 在 Family 下拉菜单中选择 Artix, Package 下拉菜单中选择 ftg256, Speed 下拉菜单选择 -1。然后, 在下面的列表中选择 xc7a35tftg256-1, 点击 Next, 进入下一步。

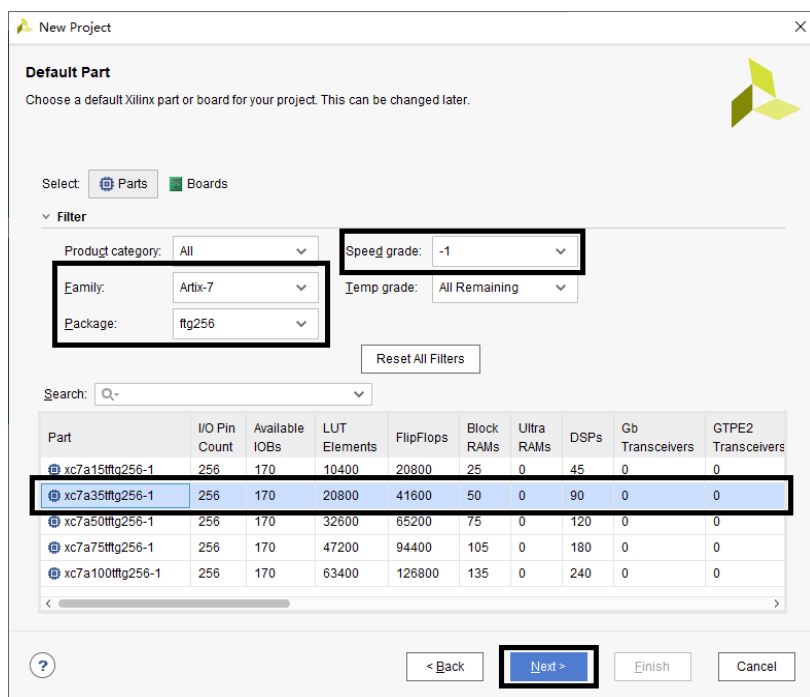


图 1-5 器件选型

(6). 在 New Project Summary 窗口中，如图 1-6 所示，查看工程创建内容是否正确。若无需修改，则点击 Finish，空白工程创建完成，进入 Vivado 主设计窗口，如图 1-7 所示。

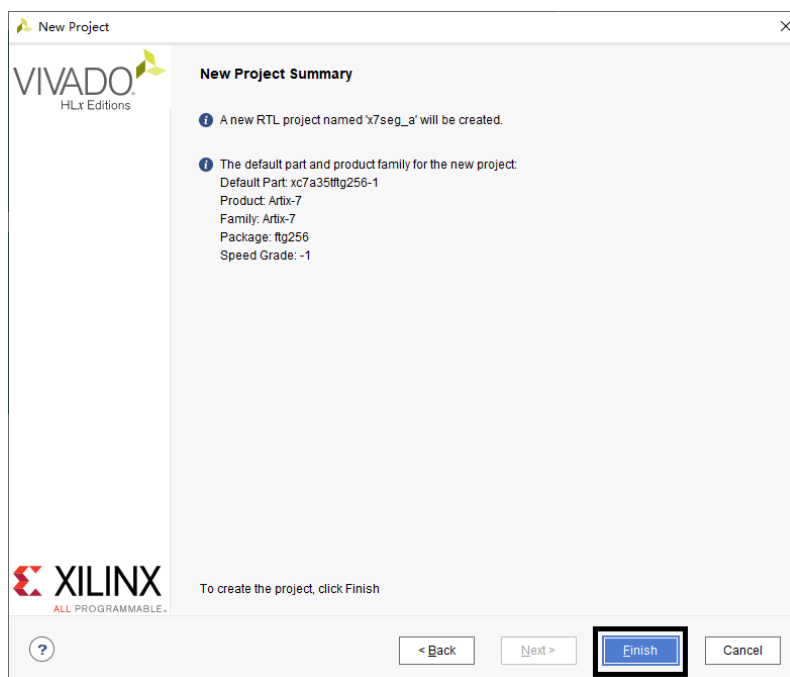


图 1-6 New Project Summary 界面

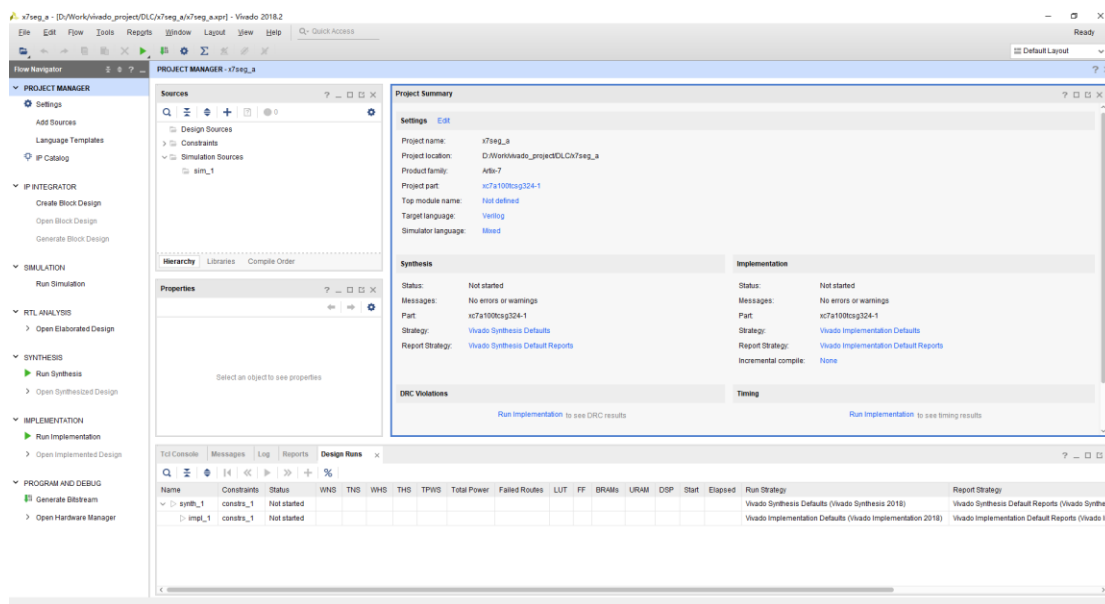


图 1-7 Vivado 主设计窗口

## 2. 添加设计源文件

(1). 在流程导航栏 (Flow Navigator) 中, 选择 PROJECT MANAGER 下的 Add Sources 选项, 如图 1-8 所示

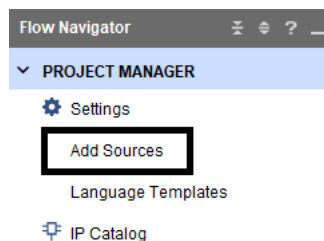


图 1-8 添加源文件

(2). 在弹出的 Add Sources 窗口中, 如图 1-9 所示, 选择 Add or create design sources 用于添加或创建 HDL 设计源文件, 单击 Next 进入下一步。

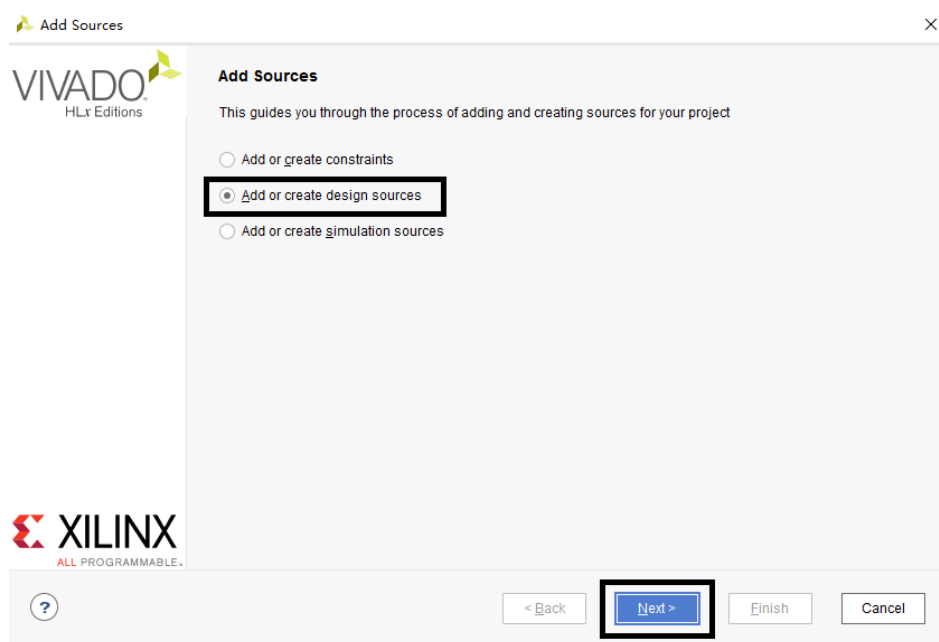


图 1-9 Add Sources 窗口

(3). 进入 Add or Create Design Sources 窗口，如图 1-10 所示。如果已经存在 HDL 设计文件，则单击 Add File 按钮进行添加；否则单击 Create File 按钮。现在单击 Create File 按钮创建新的设计源文件。

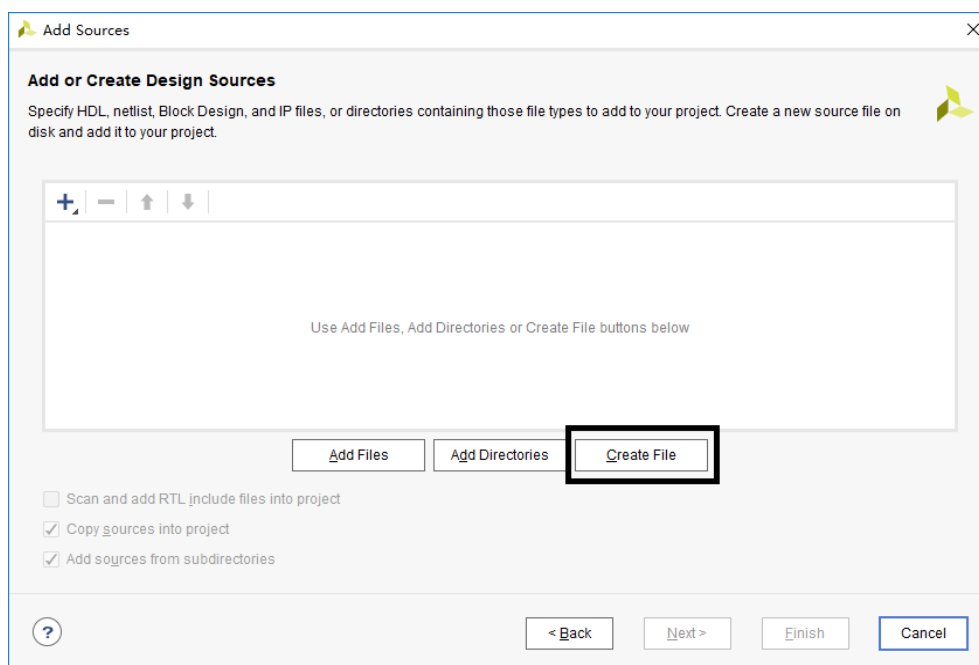


图 1-10 Add or Create Design Sources 窗口

(4). 在打开的 Create Sources File 窗口中，如图 1-11 所示，File type 下拉菜

单中选择 SystemVerilog，在文本框 File name 中输入“x7seg\_a”，其它选项默认，单击 OK 按钮退出，即创建了一个名为 x7seg\_a 的 SystemVerilog HDL 文件。如有需要，按同样方法可以一次性新建或添加多个文件。

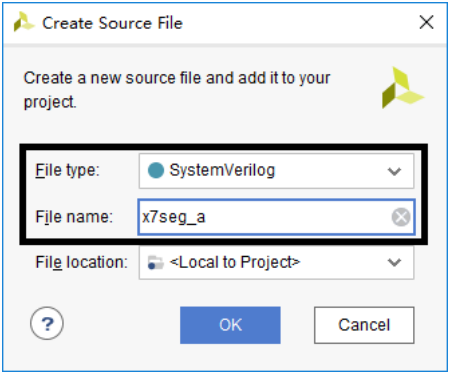


图 1-11 创建设计源文件

(5). 返回 Add or Create Design Sources 窗口，显示出刚刚创建的设计源文件，如图 1-12 所示。单击 Finish 按钮，完成设计源文件的创建。

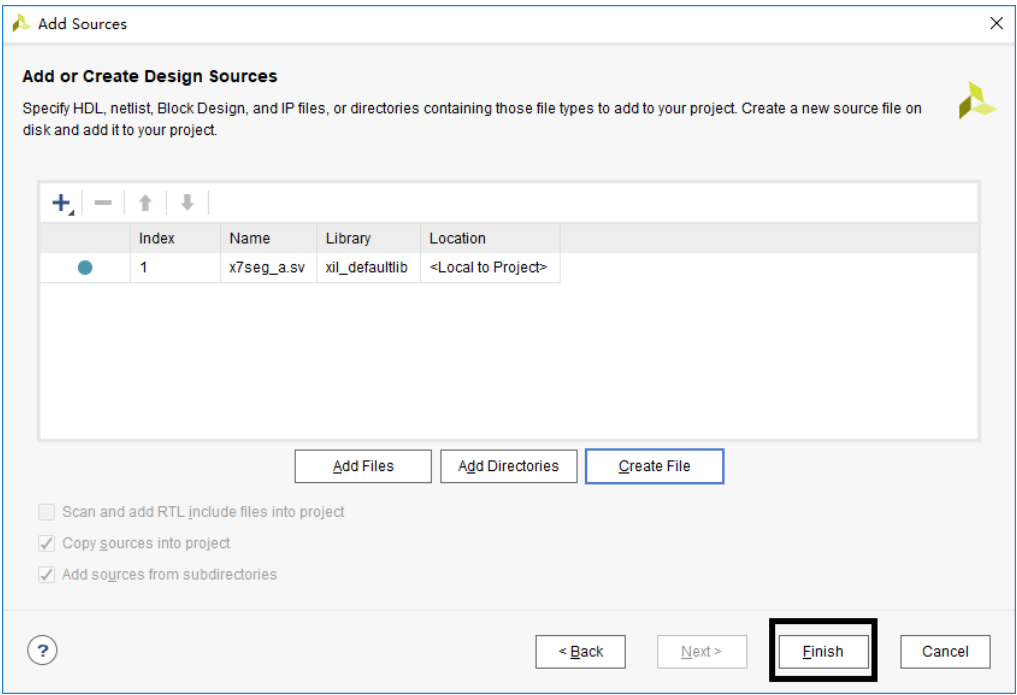


图 1-12 已创建的设计源文件

(6). 弹出 Define Module 窗口，用于定义模块和指定 I/O 端口，如图 1-13 所示。设计者既可以选择在该窗口设置端口，也可直接跳过该步骤，直接在源文件

中添加。选择后者，直接点击 OK 按钮，进入下一步。

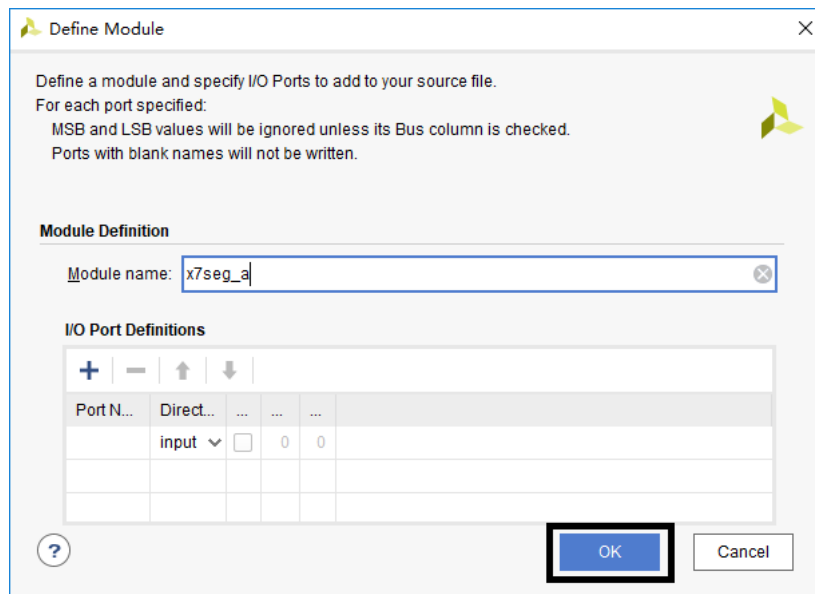


图 1-13 定义模块和指定 I/O 端口

(7). 弹出如图 1-14 所示对话框，提示用户该模块未做任何改动，单击 Yes 按钮。

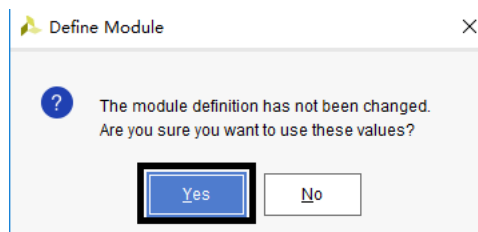


图 1-14 Define Module 提示对话框

(8). 返回 Vivado 主设计窗口，在 Sources 子窗口下的 Design Sources 选项下可看到刚刚添加的设计源文件 **x7seg\_a.sv**，双击该源文件，进入程序编辑窗口，如图 1-15 所示。



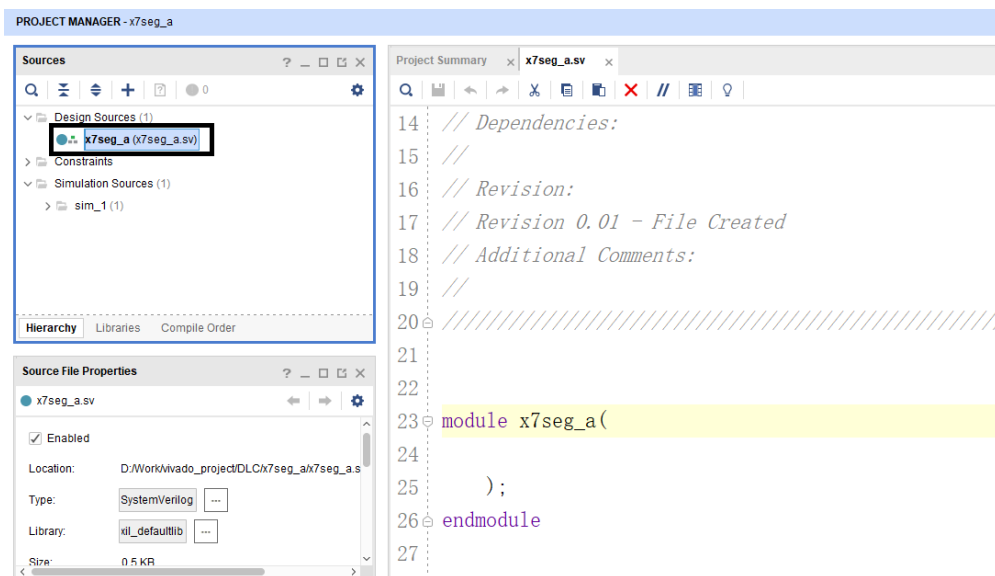


图 1-15 Sources 子窗口（设计源文件的输入）

(9). 在程序编辑窗口添加七段数码管显示译码电路的 SystemVerilog HDL 源代码，如下所示。代码编写完成后（a\_to\_g[6]对应 a 管，a\_to\_g[0]对应 g 管），按“Ctrl + S”进行保存。

```

module x7seg_a(

    input      [3 : 0] x,
    output logic [6 : 0] a_to_g

);

    always_comb begin

        case (x)

            0      : a_to_g = 7'b0000001;
            1      : a_to_g = 7'b1001111;
            2      : a_to_g = 7'b0010010;
            3      : a_to_g = 7'b0000110;
            4      : a_to_g = 7'b1001100;
            5      : a_to_g = 7'b0100100;
            6      : a_to_g = 7'b0100000;
            7      : a_to_g = 7'b0001111;
            8      : a_to_g = 7'b0000000;
            9      : a_to_g = 7'b0000100;
            'hA    : a_to_g = 7'b0001000;
            'hB    : a_to_g = 7'b1100000;
            'hC    : a_to_g = 7'b0110001;
            'hD    : a_to_g = 7'b1000010;
            'hE    : a_to_g = 7'b0110000;
            'hF    : a_to_g = 7'b0111000;
            default : a_to_g = 7'b0000001;

        endcase

    end

endmodule

```

### 3. 行为仿真

(1). 行为仿真需要添加测试程序 testbench。与添加设计源文件的过程相同，在 Flow Navigator 中，选择 PROJECT MANAGER 下的 Add Sources 选项。

(2). 在弹出的 Add Sources 窗口中，如图 1-16 所示，选择 Add or create simulation sources 用于添加或创建仿真文件，单击 Next 进入下一步。

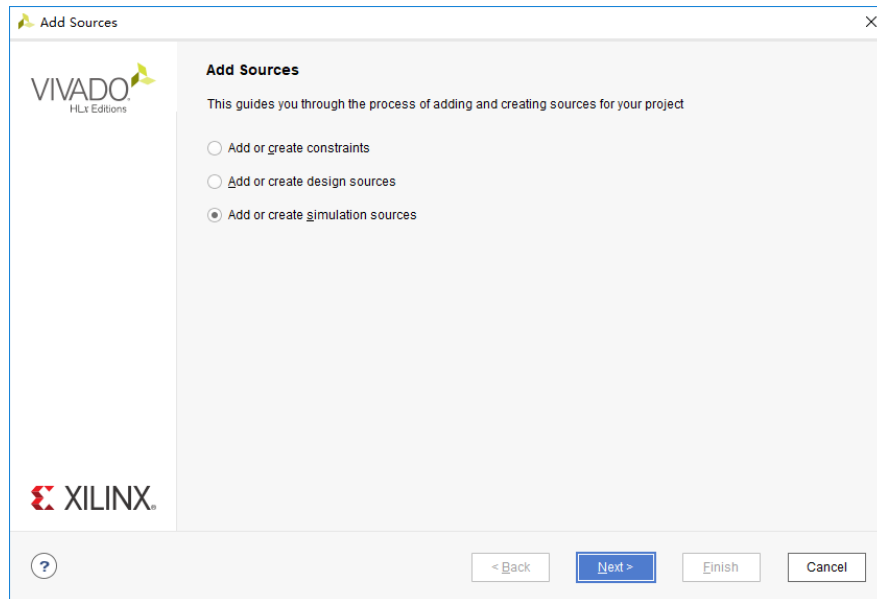


图 1-16 Add Sources 窗口

(3). 在弹出的 Add or Create Simulation Sources 窗口中，如图 1-17 所示，点击 Create File 按钮，弹出 Create Source File 界面，用于创建仿真文件。在 File type 下拉菜单中选择 SystemVerilog，在文本框 File name 中输入 x7seg\_a\_tb，其他选项默认，即创建了一个名为 x7seg\_a\_tb 的仿真测试文件，单击 OK，退出该界面。最后，点击 Finish，完成测试文件的添加。

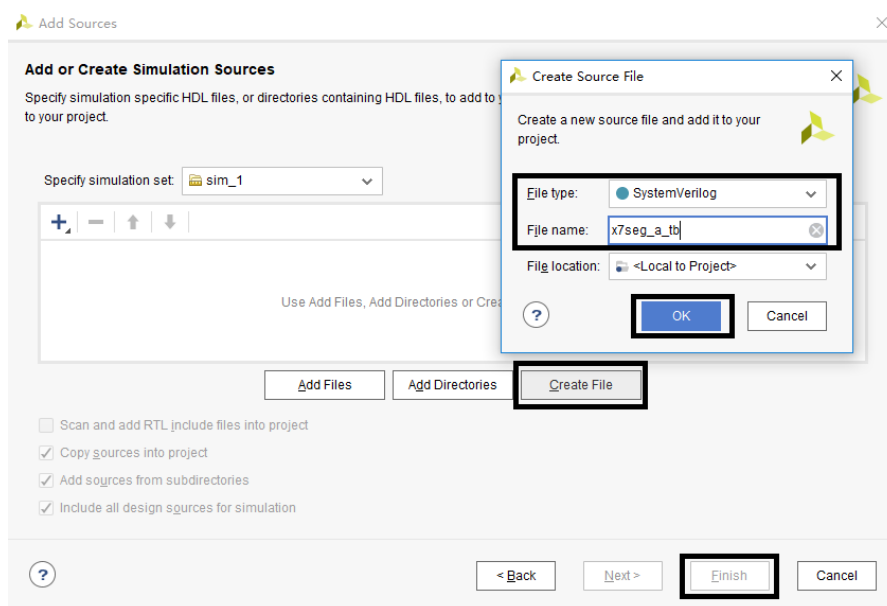


图 1-17 创建仿真文件

(4). 由于仿真文件无需指定任何 I/O 端口，故在弹出的 Define Module 窗口中，如图 1-18 所示，直接单击 OK 按钮。然后，在弹出的提示用户模块未作任何改动的对话框中，单击 Yes 按钮。

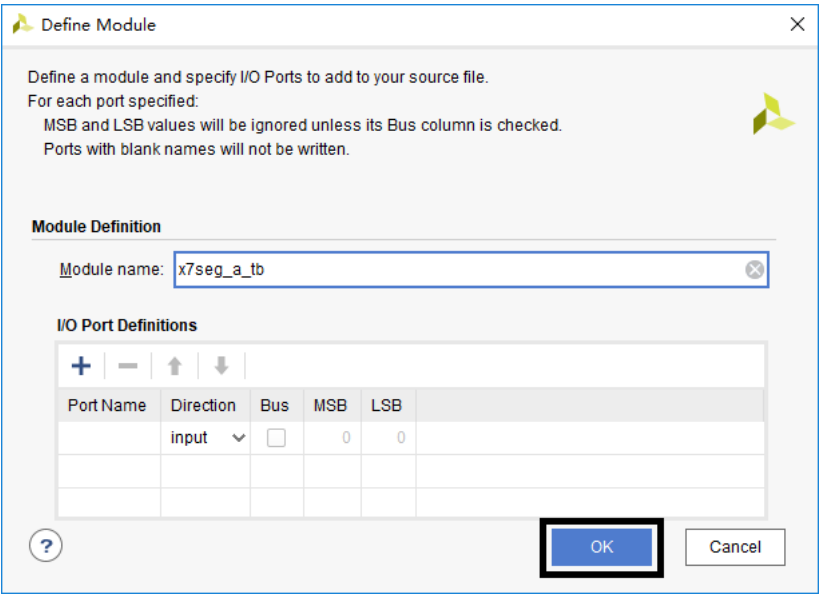


图 1-18 Define Module 窗口

(5). 返回 Vivado 主设计窗口，在 Sources 子窗口下的 Simulation Sources 选项下可看到刚刚添加的仿真测试文件 **x7seg\_a\_tb.sv**，如图 1-19 所示。双击该文件，进入程序编写窗口。

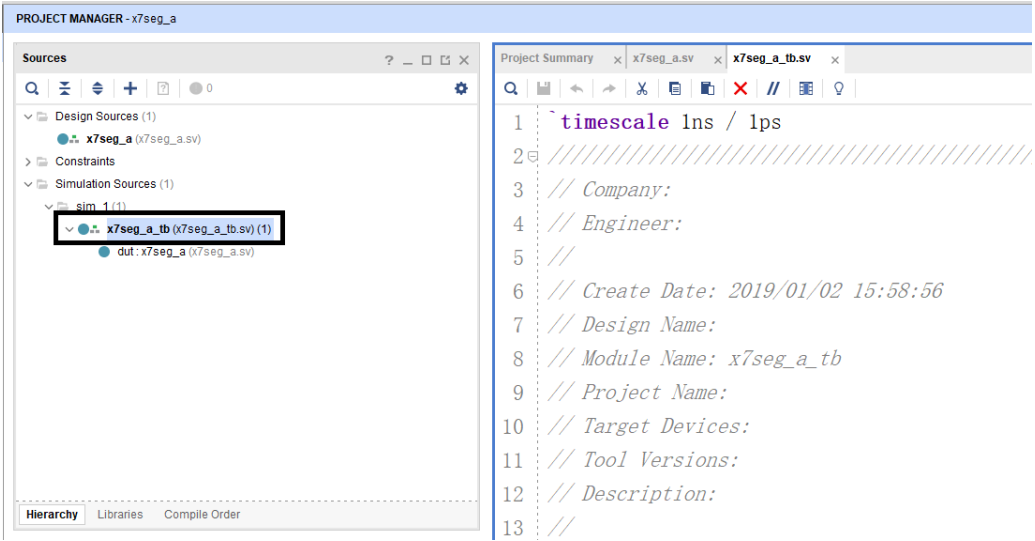


图 1-19 Sources 子窗口（仿真测试文件的输入）

(6). 按如下编写仿真测试文件后，按“Ctrl + S”进行保存。

```
`timescale 1ns/1ps

module x7seg_a_tb( );

    logic [3 : 0] x;
    logic [6 : 0] a_to_g;

    integer i;

    x7seg_a DUT(.x(x), .a_to_g(a_to_g));

    initial begin

        for(i = 0; i < 16; i = i + 1) begin
            x = i;
            #20;
        end

    end

    initial begin
        $timeformat(-9, 0, "ns", 5);
        $monitor("At time %t: x = %b, a_to_g = %b", $time,
x, a_to_g);
    end

endmodule
```

(7). 在 Flow Navigator 中点击 SIMULATION → Run Simulation → Run Behavioral Simulation，开始行为仿真，如图 1-20 所示。

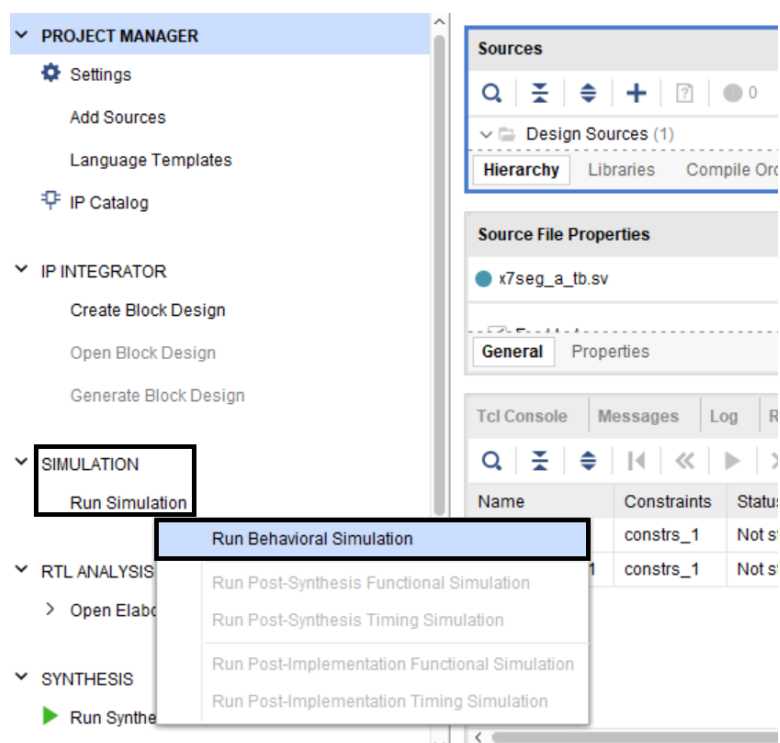
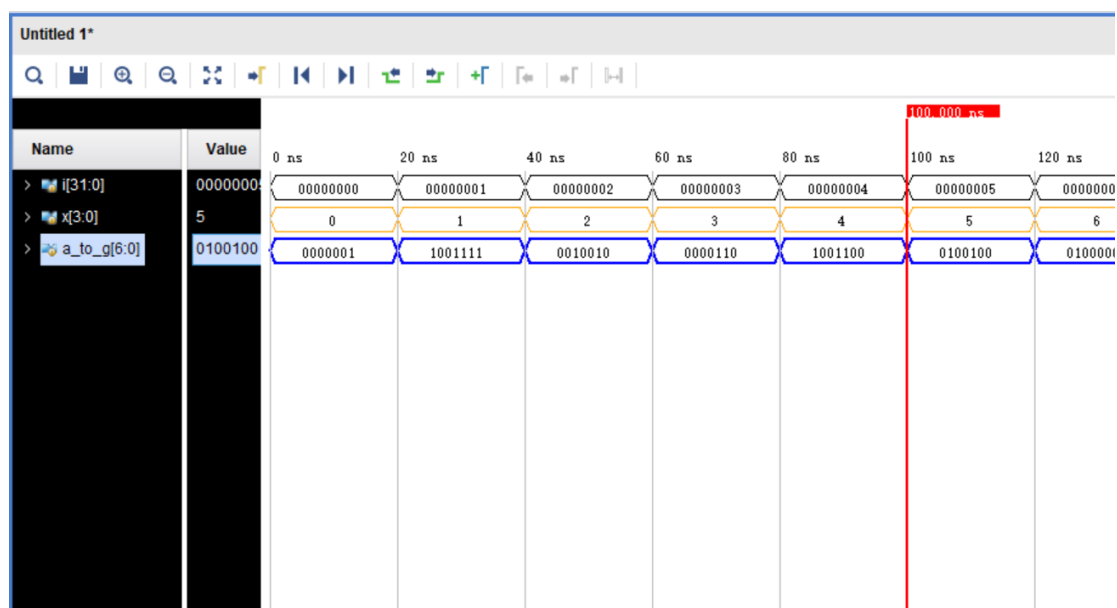
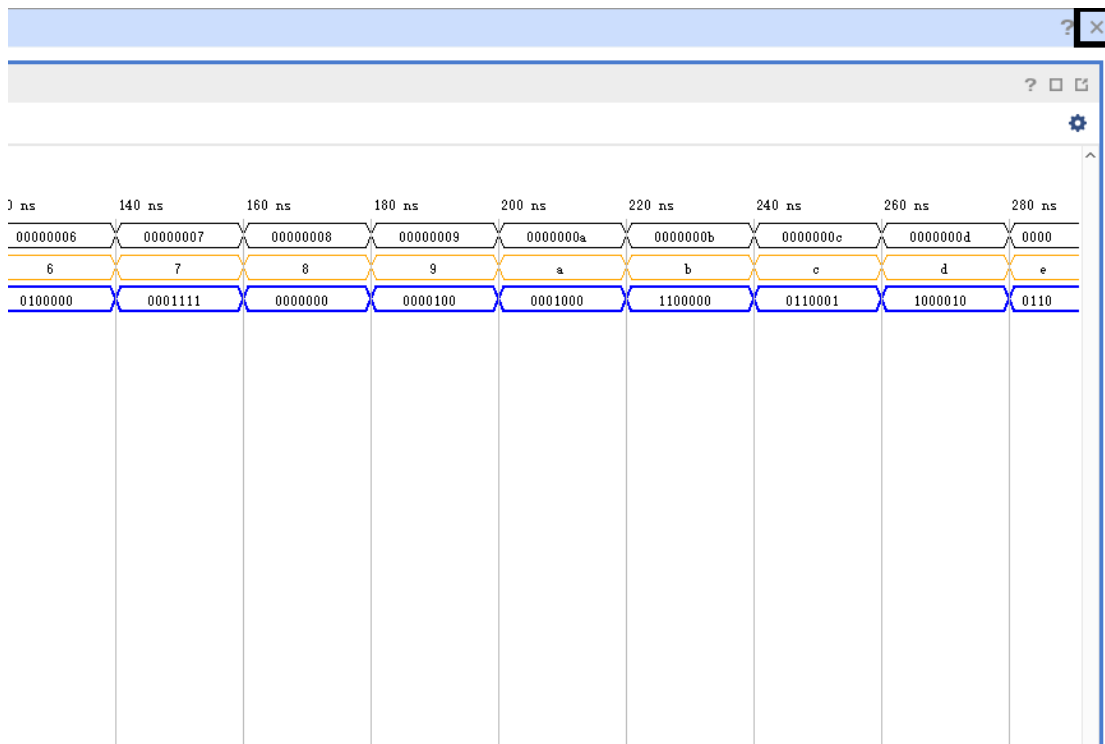


图1-20 行为仿真

(8). 如果没有出现报错信息，则弹出如图 1-21(a)所示的仿真波形图（测试向量不同，波形图就不同）。同学们可根据波形图验证所设计的七段数码管显示译码器是否正确。验证通过后点击右上角“X”，如图 1-21(b)所示，关闭仿真窗口。



(a)



(b)

图 1-21 仿真波形图

#### 4. 添加设计约束

在工程中添加了设计源文件，并完成行为仿真后，需要给设计添加约束，其中包含管脚约束和时序约束。**管脚约束**是将设计源文件中顶层模块的输入/输出端口和FPGA芯片的某个管脚进行绑定。**时序约束**是为设计设置期望的时钟属性，如频率，偏斜、抖动等，其目的是为了提高设计的工作频率和获得正确的时序分析报告，在综合和实现过程中，EDA 工具会以时序约束为标准，优化电路以尽量满足约束要求，同时产生实际时序和用户期望时序之间的差异，并形成报告。添加约束有两种方法，第一种是新建 XDC 约束文件，并输入约束命令；第二种方法是利用 Vivado 中的 I/O Planning 功能进行图形化的约束。本文采用第一种方法（也是推荐使用的办法，第二种可上网学习），具体步骤如下。

(1). 在 Flow Navigator 中选择 PROJECT MANAGER 下的 Add Sources 选项。

(2). 在弹出的 Add Sources 窗口中, 如图 1-22 所示, 选择 Add or create simulation constraints 用于添加或创建约束文件, 单击 Next 进入下一步。

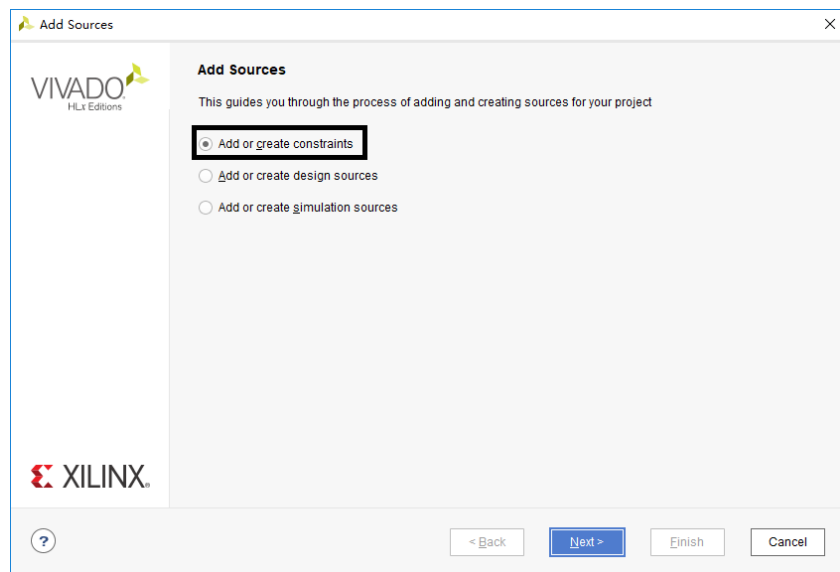


图 1-22 Add Sources 窗口

(3). 在弹出的 Add or Create Constraints 窗口中, 如图 1-23 所示, 点击 Create File 按钮, 弹出 Create Constraints File 窗口, 创建约束文件。在文本框 File name 中输入 x7seg\_a\_xdc, 其他选项默认, 即创建了一个名为 x7seg\_a\_xdc 的约束文件, 单击 OK, 退出该窗口。最后, 点击 Finish, 完成约束文件的添加。

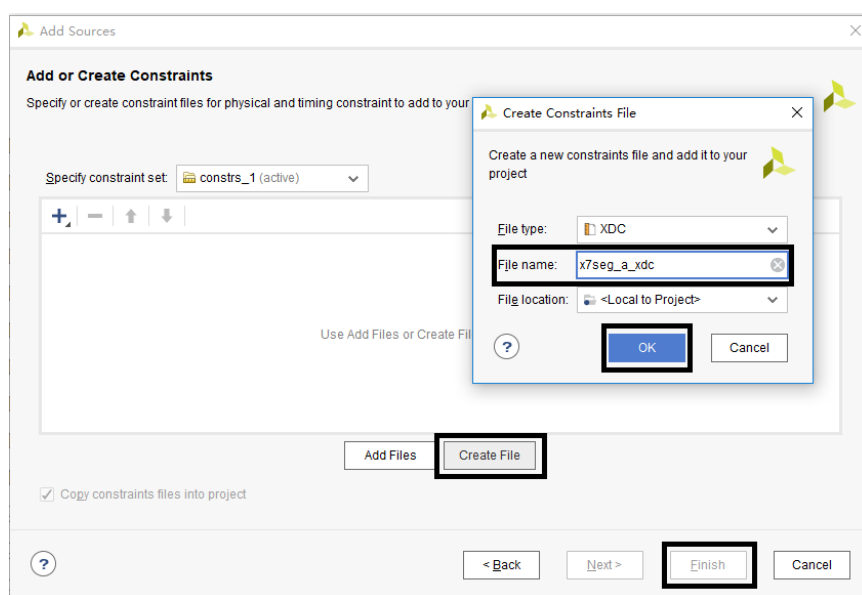


图 1-23 创建约束文件



(4). 返回 Vivado 主窗口，在 Sources 子窗口下的 Constraints 选项下可看到刚刚添加的约束文件 **x7seg\_a\_xdc.xdc**，如图 1-24 所示。双击该文件，进入编辑窗口，添加约束命令，本文只需要给出管脚约束。

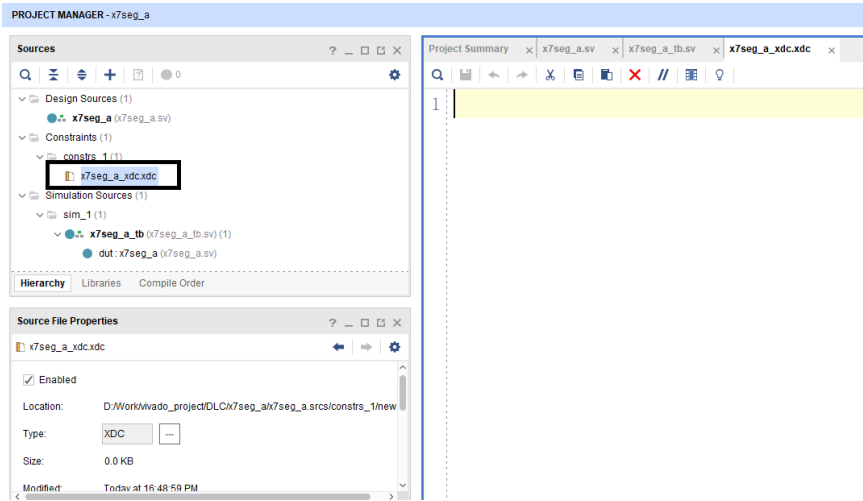


图 1-28 Sources 子窗口（约束文件的输入）

(5). 打开所提供的“远程硬件平台管脚对应关系.xlsx”文件，如图 1-29 所示。该文件提供了远程 FPGA 硬件平台所提供的可用管脚，分为输入管脚和输出管脚，是进行管脚约束的依据。其中 FPGA 一列给出了所有可用管脚的名称，注意，备注中有专门注释的管脚是与专门设备连接的管脚，不能分配给本设计使用。

远程硬件平台管脚对应关系.xlsx - Excel									
文件 开始 插入 页面布局 公式 数据 审阅 视图 帮助 Acrobat 百度网盘 操作说明搜索									
K18									
A B C D E F G H I J									
10	输入管脚				输出管脚（FPGA输出，STM32输入）				
11	编号	FPGA	备注	备注	编号	FPGA	备注	备注	
12	0	B9	GPIO22		0	L14	GPIO5		
13	1	D11	GPIO33		1	M14	GPIO6		
14	2	B11	GPIO25		2	L15	GPIO3	1602-en	
15	3	B12	GPIO26		3	M15	GPIO4	12864-E	
16	4	A10	GPIO23		4	K13	GPIO7		
17	5	B10	GPIO24		5	L13	GPIO8		
18	6	J13	GPIO0		6	L12	GPIO9		
19	7	K15	GPIO1		7	T9	GPIO10		
20	8	K16	GPIO2		8	C6	GPIO43		
21	9	P6	GPIO16		9	D6	GPIO44		
22	10	D10	GPIO17		10	D5	GPIO45		
23	11	C8	GPIO18		11	A2	GPIO49	1602-en	
24	12	C9	GPIO19	UART6_TX	12	D3	GPIO54	PWM输出	
25	13	A8	GPIO20	UART6_RX	13	C4	GPIO55		
26	14	A9	GPIO21		14	D16	GPIO39	PWM输出	
27	15	J1	GPIO74		15	C7	GPIO42		
28	16	L3	GPIO75		16	D8	GPIO28	UART7_TX	
29	17	L2	GPIO76		17	A12	GPIO27	UART7_RX	
30	18	K3	GPIO77		18	D9	GPIO29		
31	19	K2	GPIO78		19	E12	GPIO30		

图 1-29 远程硬件平台管脚对应关系.xlsx

(6). 在 x7seg\_a\_xdc.xdc 文件中填写如下所示的管脚约束（#后的是注释部分）。例如“set\_property -dict { PACKAGE\_PIN B9 IOSTANDARD LVCMOS33 } [get\_ports { x[0] }];”，表示将 FPGA 的 B9 管脚与设计中的输入信号 x[0]连接在一起。因此，对于每个管脚约束，只需要修改上句约束中的红色部分即可。

```
#Constraints of input ports x
set_property -dict { PACKAGE_PIN B9 IOSTANDARD LVCMOS33 } [get_ports { x[0] }];
set_property -dict { PACKAGE_PIN D11 IOSTANDARD LVCMOS33 } [get_ports { x[1] }];
set_property -dict { PACKAGE_PIN B11 IOSTANDARD LVCMOS33 } [get_ports { x[2] }];
set_property -dict { PACKAGE_PIN B12 IOSTANDARD LVCMOS33 } [get_ports { x[3] }];

#Constraints of output ports a_to_g
set_property -dict { PACKAGE_PIN K13 IOSTANDARD LVCMOS33 } [get_ports { a_to_g[0] }];
set_property -dict { PACKAGE_PIN L13 IOSTANDARD LVCMOS33 } [get_ports { a_to_g[1] }];
set_property -dict { PACKAGE_PIN L12 IOSTANDARD LVCMOS33 } [get_ports { a_to_g[2] }];
set_property -dict { PACKAGE_PIN T9 IOSTANDARD LVCMOS33 } [get_ports { a_to_g[3] }];
set_property -dict { PACKAGE_PIN C6 IOSTANDARD LVCMOS33 } [get_ports { a_to_g[4] }];
set_property -dict { PACKAGE_PIN D6 IOSTANDARD LVCMOS33 } [get_ports { a_to_g[5] }];
set_property -dict { PACKAGE_PIN D5 IOSTANDARD LVCMOS33 } [get_ports { a_to_g[6] }];
```

## 5. 设置烧写文件的格式

最终的设计通过编译后被转化为特定的配置文件，然后才能烧写进 FPGA 进行工作，配置文件的格式是 bin，设置步骤如下。

(1). 在 Flow Navigator 中选择 Settings 选项，进行工程设置。

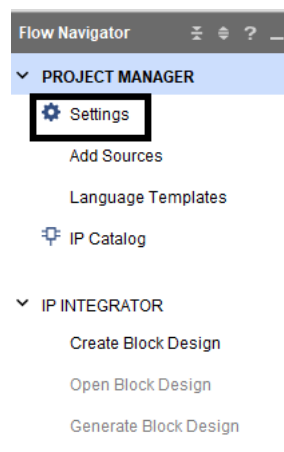


图 1-30 工程设置

(2). 在弹出的 Setting 窗口中，如图 1-31 所示，选择 Bitstream，勾选 -bin\_file 选项，再依次点击 Apply 和 OK 按钮关闭窗口。

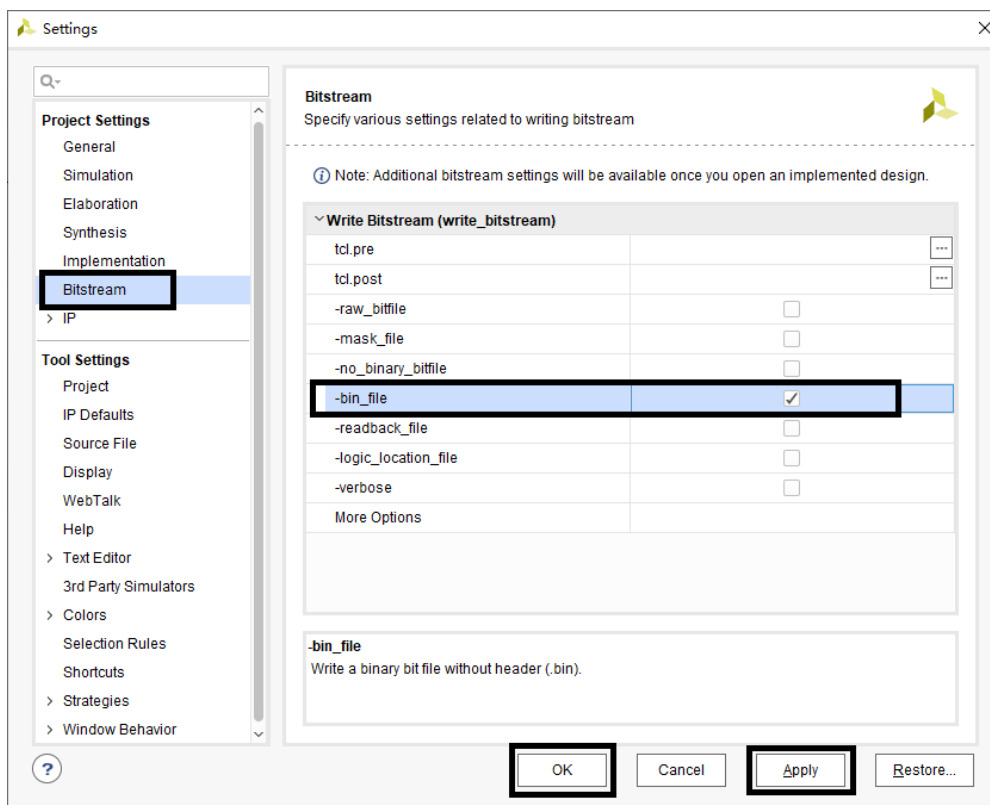
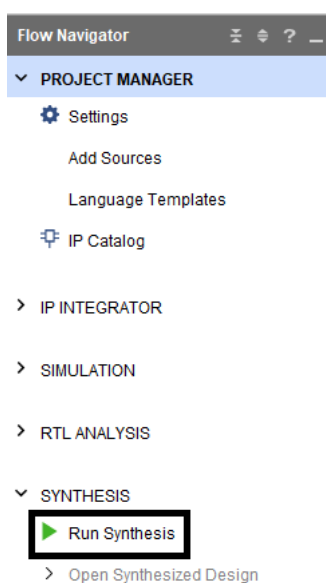


图 1-31 设置生成 bin 配置文件

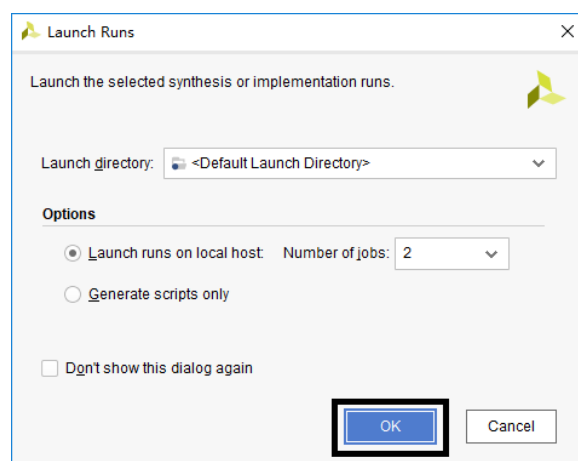
## 6. 综合

在 Flow Navigator 中, 点击 SYNTHESIS → Run Synthesis, 如图 1-32(a)所示。

然后, 在弹出的窗口中单击 OK 按钮, 如图 1-32(b)所示, 开始进行工程综合。



(a)



(b)

图 1-32 启动工程综合

## 7. 实现

综合完成后，如果没有报错，Vivado 会弹出一个窗口，提示综合已完成，等待设计者输入下一项任务。此时，可选择第 1 个选项 Run Implementation，点击 OK 按钮开始实现过程，如图 1-33 所示。（用户也可单击 Cancel 按钮关掉该窗口，然后在 Flow Navigator 中单击 IMPLEMENTATION → Run Implementation，执行实现过程。）

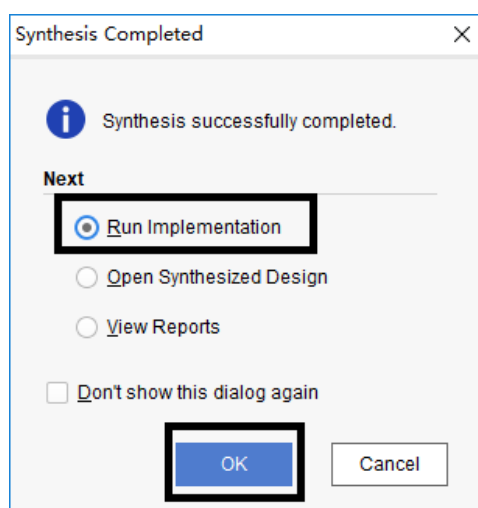


图 1-33 启动工程实现

## 8. 生成 bin 文件

实现完成后，如果没有报错，Vivado 会弹出一个窗口，提示实现已完成，等待设计者输入下一项任务。如果设计者需要继续生成 bin 文件，可选择第 2 个选项 Generate Bitstream，点击 OK 按钮，开始生成 bin 文件，如图 1-34 所示。（用户也可单击 Cancel 按钮关掉该窗口，然后在 Flow Navigator 中单击 PROGRAM AND DEBUG → Generate Bitstream，产生 bin 文件。）

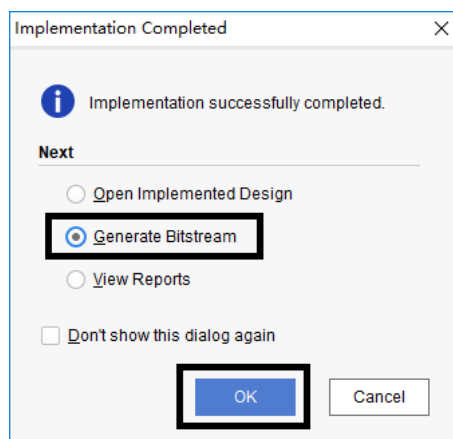


图 1-34 生成 bin 文件

至此，采用 Xilinx Vivado 进行数字逻辑电路开发的流程已经完毕，经过了创建工程、设计输入、行为仿真、添加约束文件、工程配置、综合、实现、生成 bin 文件八个步骤。这八个步骤同学们必须熟练掌握！当然，上述流程仅仅是 Vivado 开发的简略流程，除此之外，Vivado 还提供了十分丰富的设计、分析功能，包括原理图分析、资源使用情况报告、时序报告等等。大家可在使用过程中逐步体会。下面将利用远程 FPGA 硬件平台完成最终的上板验证。

## 二． 远程云端硬件平台的使用流程

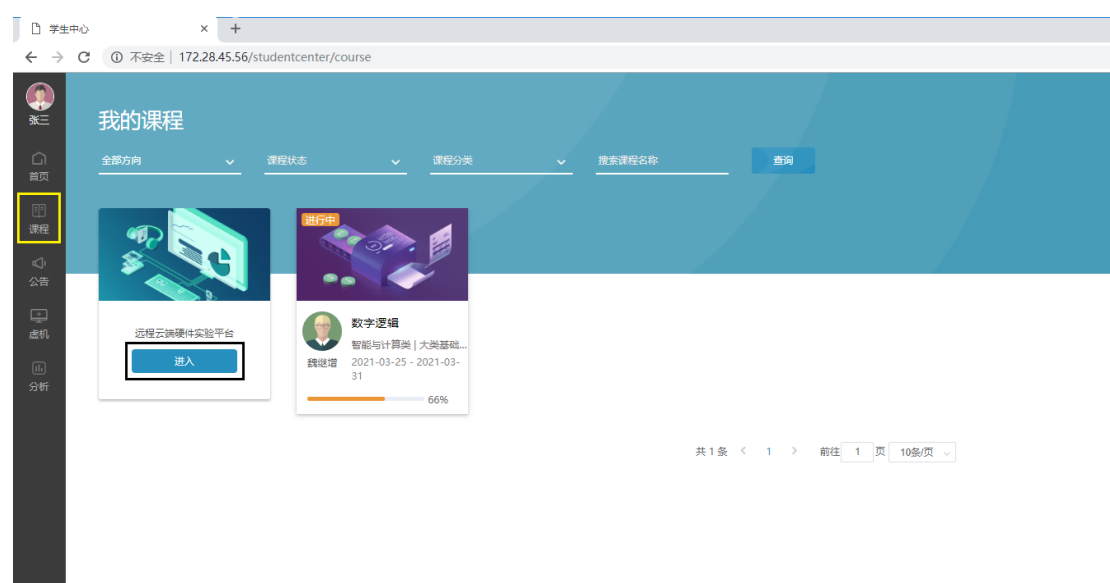
下面将利用远程 FPGA 平台对 7 段数码管显示译码器进行上板验证。

### 1. 登录远程 FPGA 硬件平台

(1). 输入网址 <http://172.28.45.56>，使用用户名和密码登入智算学部虚拟仿真实验平台，如图 1-35(a)所示。进入平台后，在左侧导航栏选择“课程”，然后找到“远程云端硬件实验平台”，点击进入，如图 1-35(b)所示。



(a)



(b)

图 1-35 远程 FPGA 硬件平台登录



(2). 进入远程 FPGA 硬件平台后, 如图 1-36 所示, 将鼠标放置在图标之上, 显示用户已与一块 FPGA 建立连接, 编号是 2004 (每位同学只能与一块 FPGA 连接, 故编号也不相同)。远程可用的 FPGA 编号为 2001~2100, 共计 100 块。如果显示为红色, 表示没有连接上设备, 可点击右上角姓名, 在弹出的菜单中选择连接设备, 通过手工输入设备编号建立连接, 如图 1-37 所示。



图 1-36 进入远程云端硬件实验平台



(a)



(b)

图 1-37 手工连接设备

## 2. 添加 FPGA 逻辑器件

(1). 点击左侧导航栏的“实验面板”打开实验图纸，如图 1-38 所示。

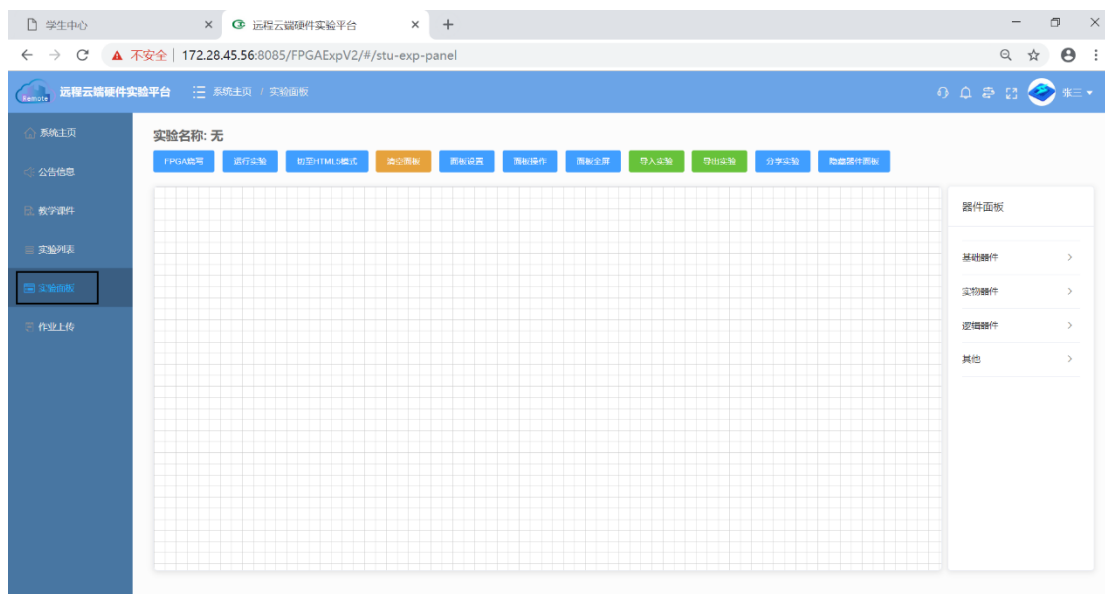


图 1-38 打开实验图纸

(2). 在右侧“器件面板”中选择“逻辑器件→自定义管脚 FPGA”，如图 1-39 所示，将其拖拽到实验图纸之上。



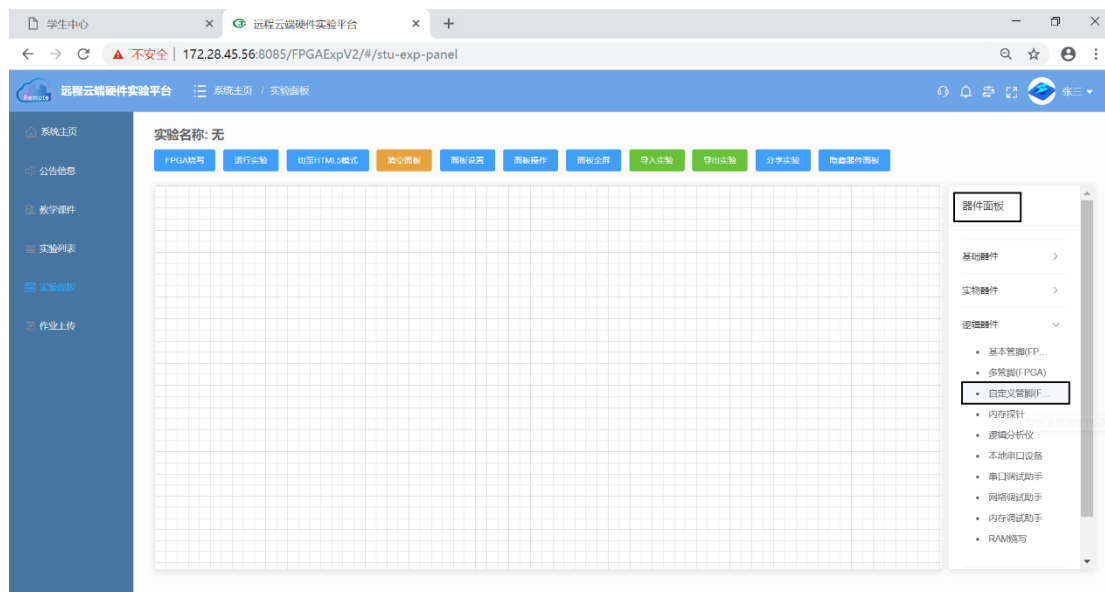
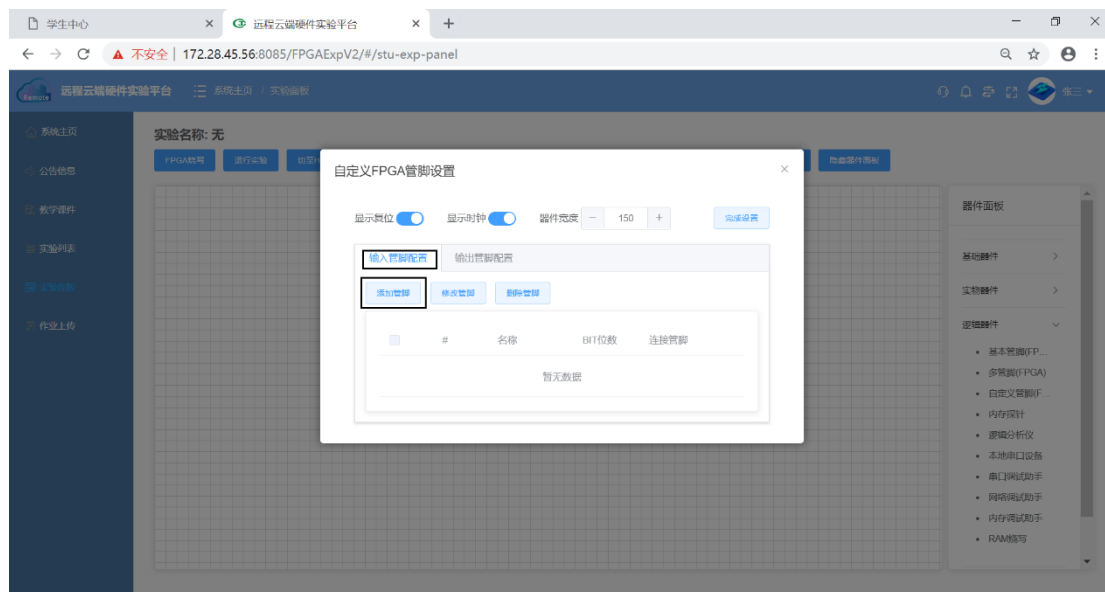
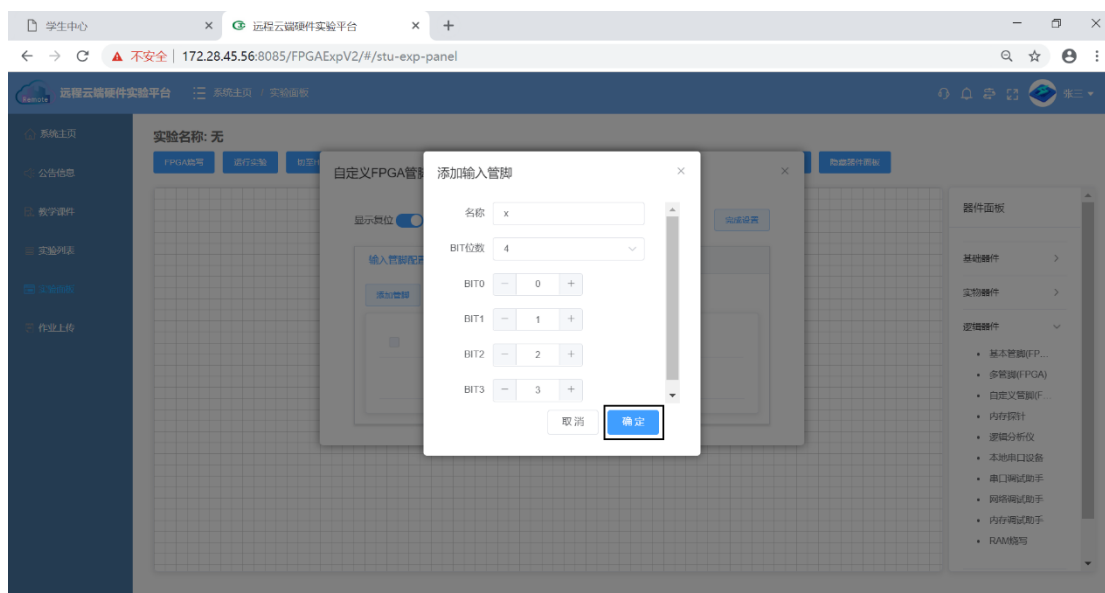


图 1-39 添加自定义管脚 FPGA

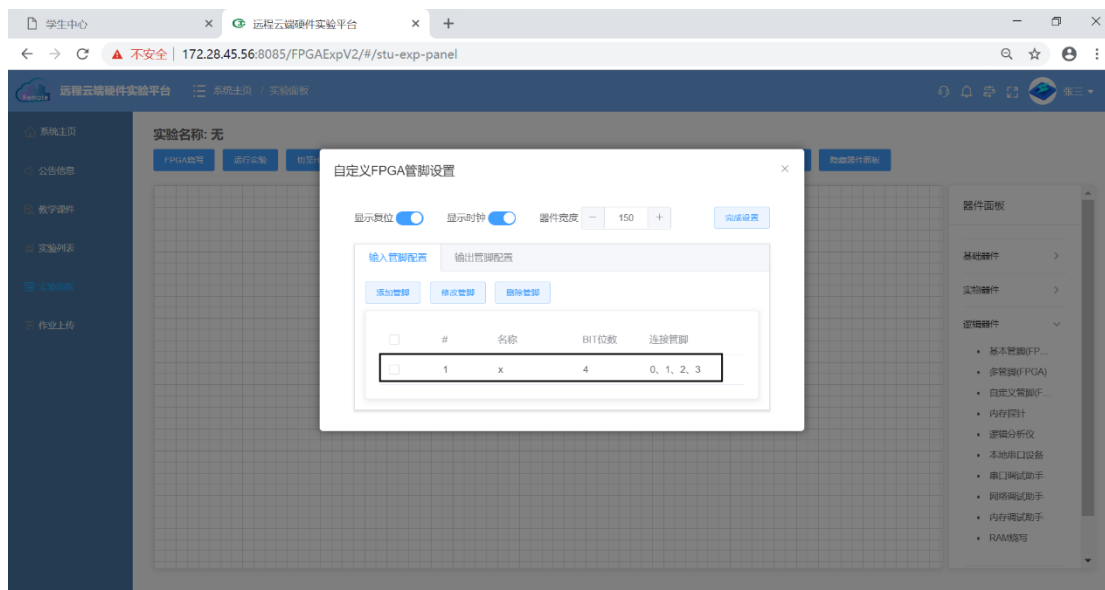
(3). 在弹出的“自定义 FPGA 管脚设置”窗口中为 FPGA 添加输入和输出管脚。如图 1-40(a)所示，在“输入管脚配置”标签中单击“添加管脚”按钮，在弹出的窗口中按图 1-40(b)所示进行输入管脚的配置。其中，“名称”中填入输入管脚的名称，即 x；“BIT 位数”表示端口宽度，填入 4；参照“远程硬件平台管脚对应关系.xlsx”文件，将 7 段数目管显示译码器的输入端口 x 所绑定的 4 个管脚（即 B9，D11，B11 和 B12）对应的编号（即 0~3），依次填入 BIT0~BIT3 中，点击“确定”按钮关闭窗口，完成设计中输入端口和 FPGA 输入管脚的绑定，如图 1-40(c)所示。



(a)



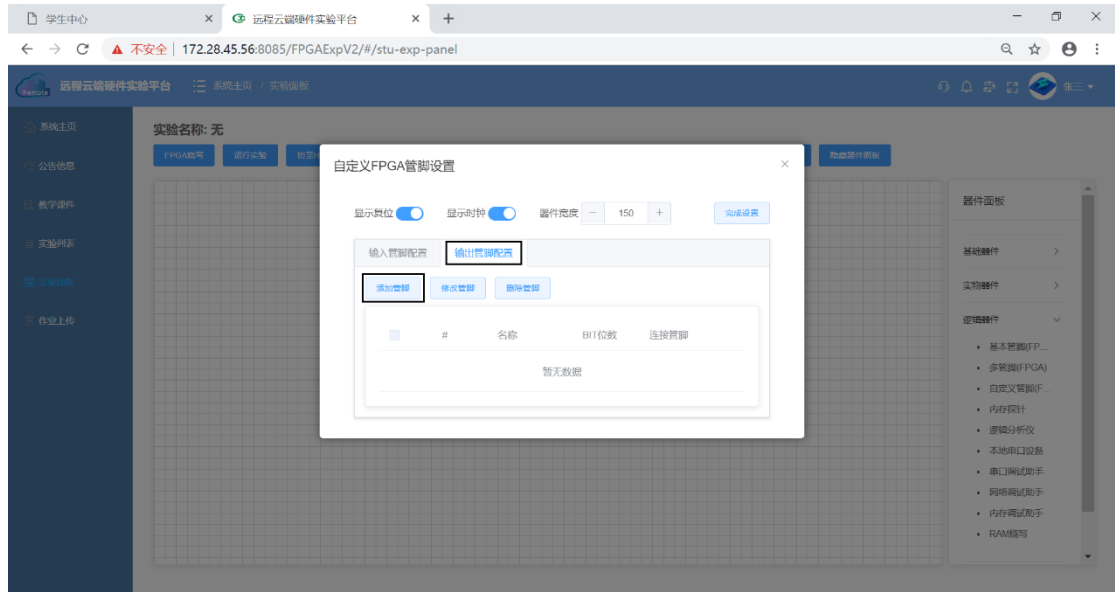
(b)



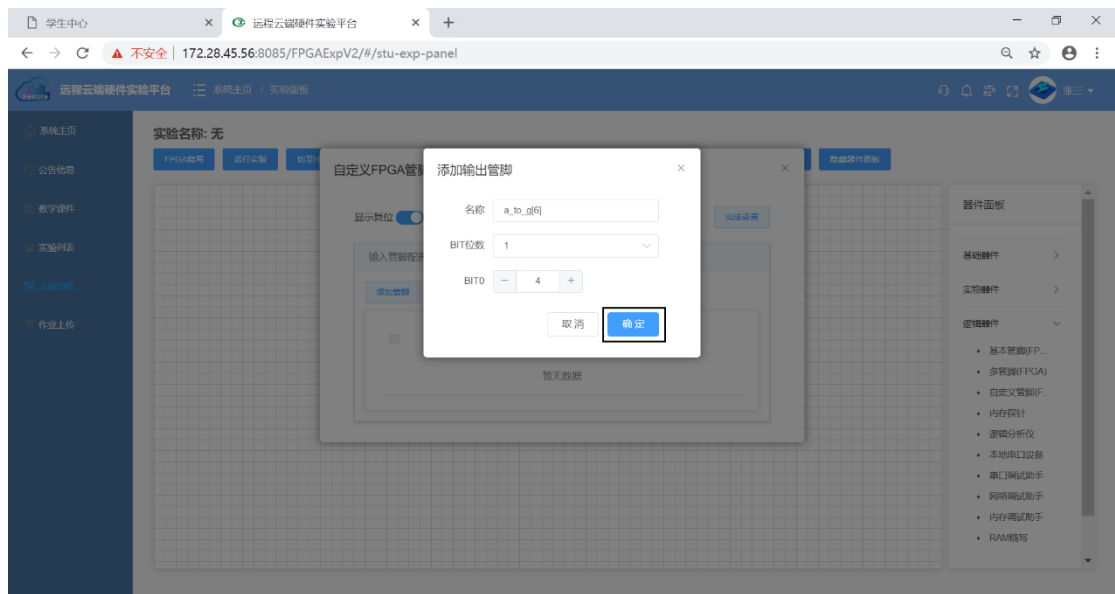
(c)

图 1-40 添加 FPGA 输入管脚并绑定

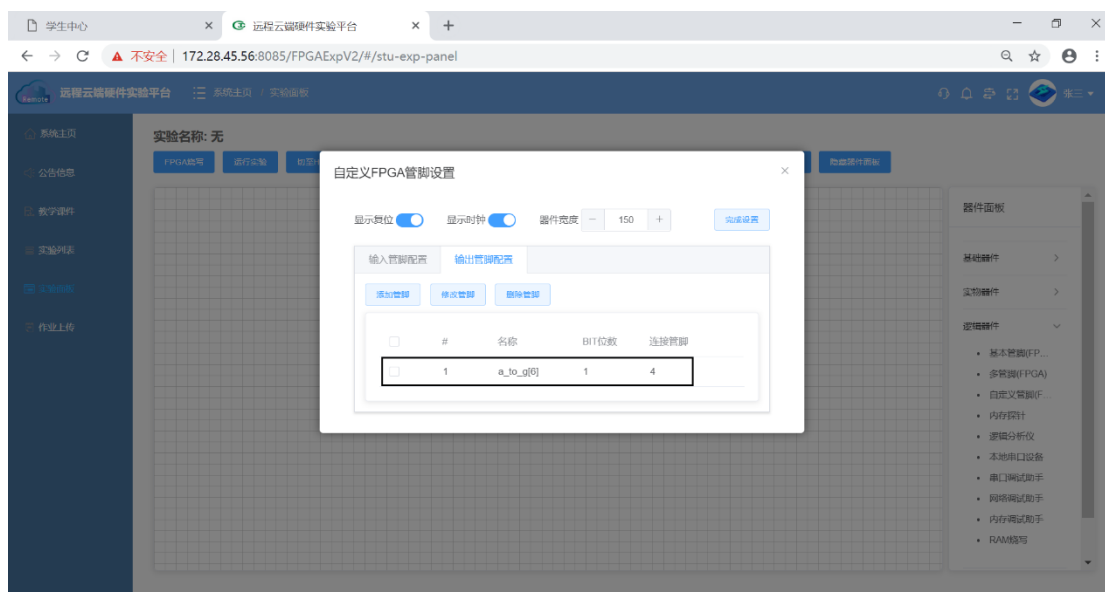
(4). 继续在“自定义 FPGA 管脚设置”窗口中选择“输出管脚配置”标签，单击“添加管脚”按钮，如图 1-41(a)所示。在弹出的窗口中按图 1-42(b)所示进行 FPGA 输出管脚的配置。其中，“名称”中填入 a\_to\_g[6]；“BIT 位数”填入 1；参照“远程硬件平台管脚对应关系.xlsx”文件，将 7 段数目管显示译码器输出端口的第 6 位（a\_to\_g[6]）所绑定的管脚（即 K13）对应的编号（即 4）填入 BIT0 中，点击“确定”按钮关闭窗口，完成将设计中输出管脚的第 6 位与 FPGA 管脚的绑定，如图 1-41(c)所示。依照相同步骤完成剩余 6 个管脚（即 a\_to\_g[5]~a\_to\_g[0]）的绑定，如图 1-41(d)所示。



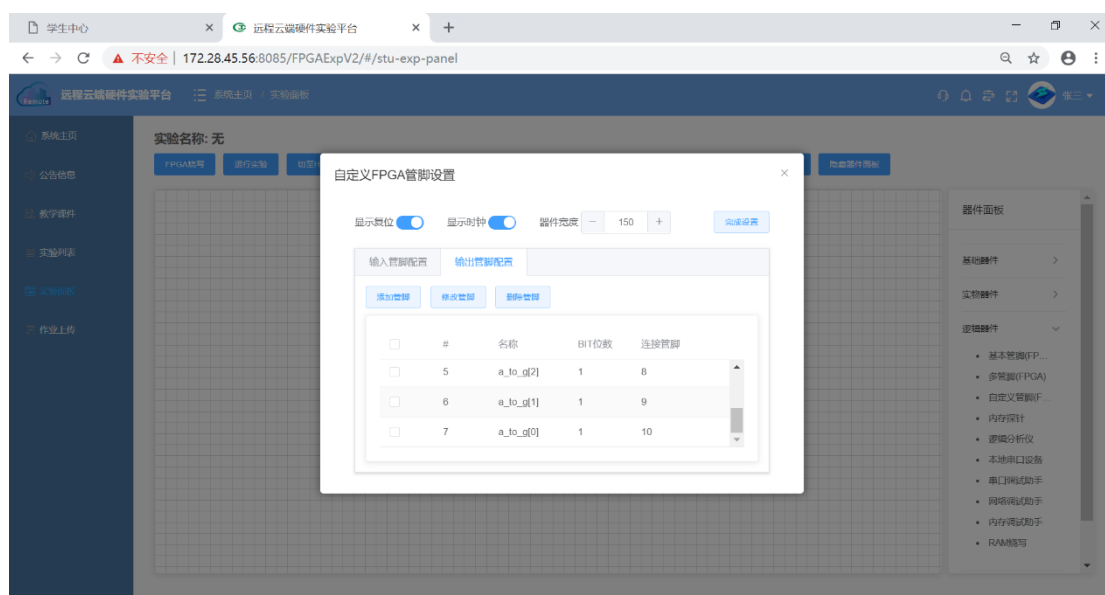
(a)



(b)



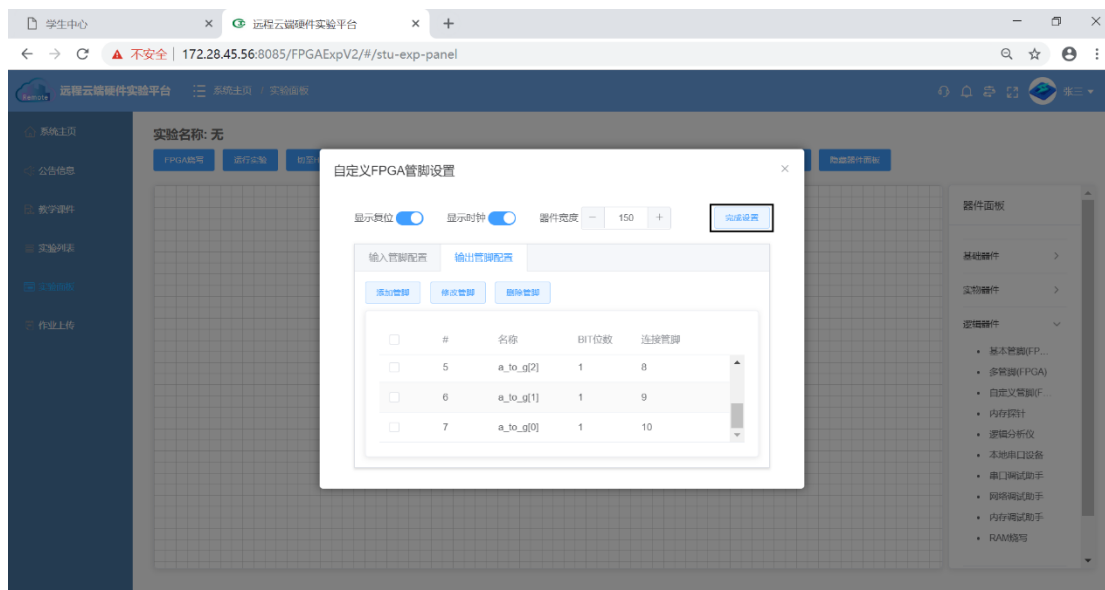
(c)



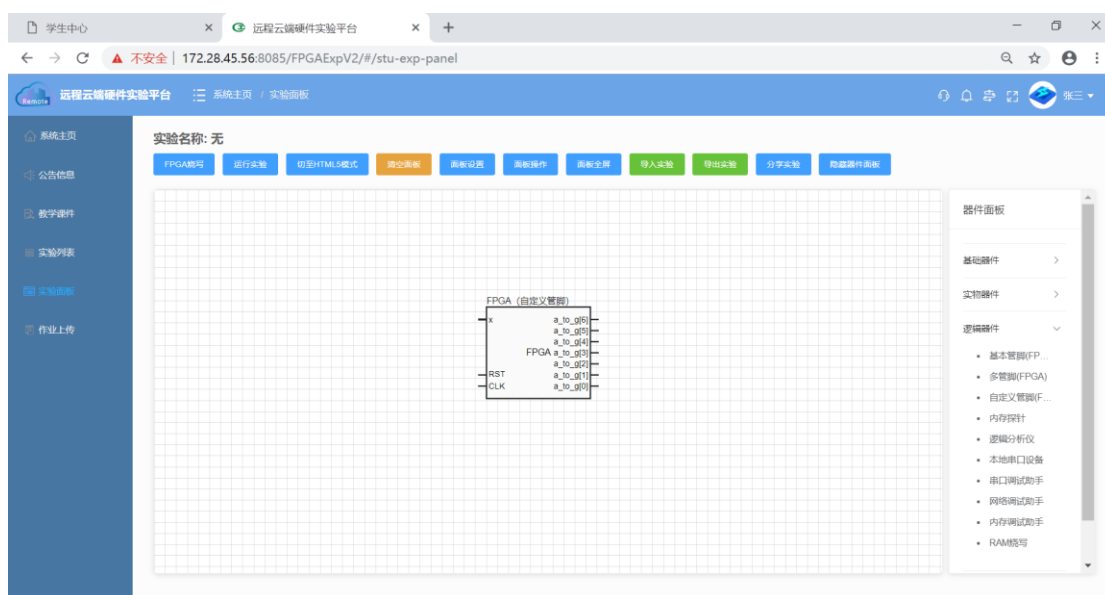
(d)

图 1-41 添加 FPGA 输出管脚并绑定

(5). 在“自定义 FPGA 管脚设置”窗口中单击“完成设置”按钮,如图 1-42(a)所示,完成逻辑器件 FPGA 的添加,如图 1-42(b)所示。



(a)



(b)

图 1-42 完成逻辑器件 FPGA 的添加

### 3. 添加 7 段数码管器件

(1). 从右侧“实物理器件”中，拖拽一个“数码管”器件进入实验图纸，如图 1-43 所示。

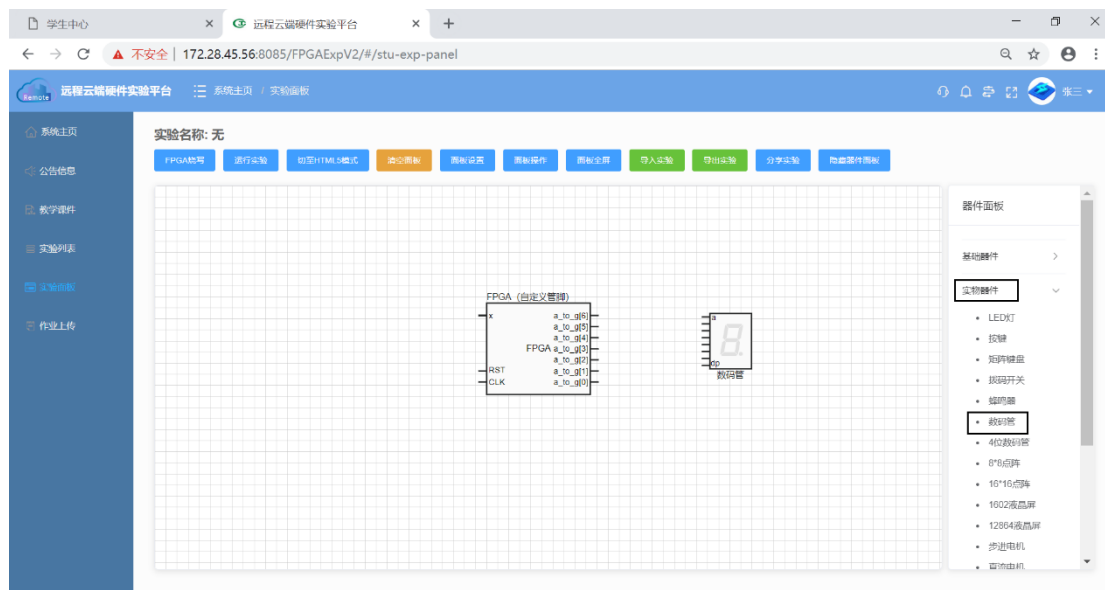


图 1-43 添加七段数码管

(2). 右键单击“数码管”元件，在弹出的菜单中选择“编辑参数”对数码管进行配置。其中，“公共极类型”选择共阳极，“管脚合并”不选通，单击确定按钮完成配置，如图 1-44 所示。

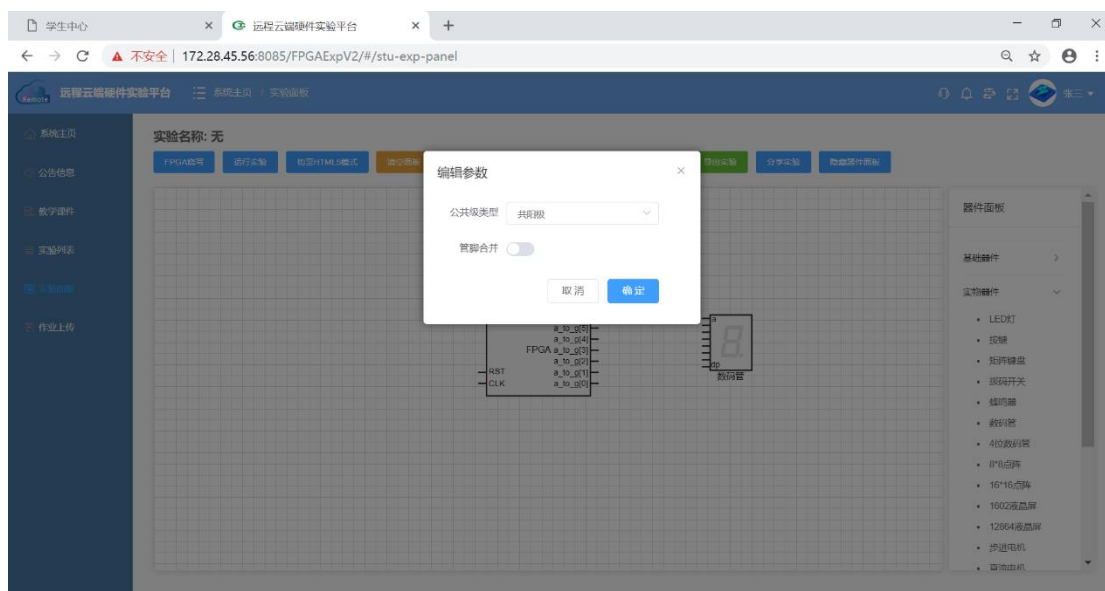


图 1-44 配置七段数码管

(3). 按照图 1-45 所示，将 FPGA 的输出管脚与七段数码管相连。连接的方法是鼠标单击器件的引出管脚一次即可引出一条导线，在单击另一个器件的管脚

即可实现连接。

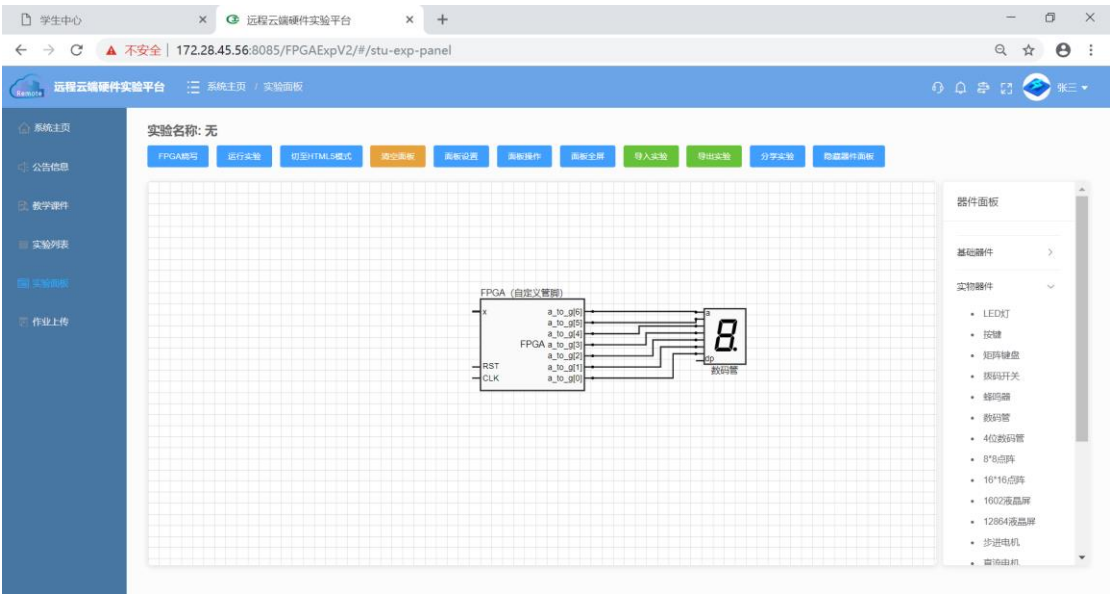


图 1-45 连接 FPGA 和七段数码管

#### 4. 添加输入器件

(1). 从右侧“基础器件”中，拖拽一个“多位输入”器件进入实验图纸，如图 1-46 所示。

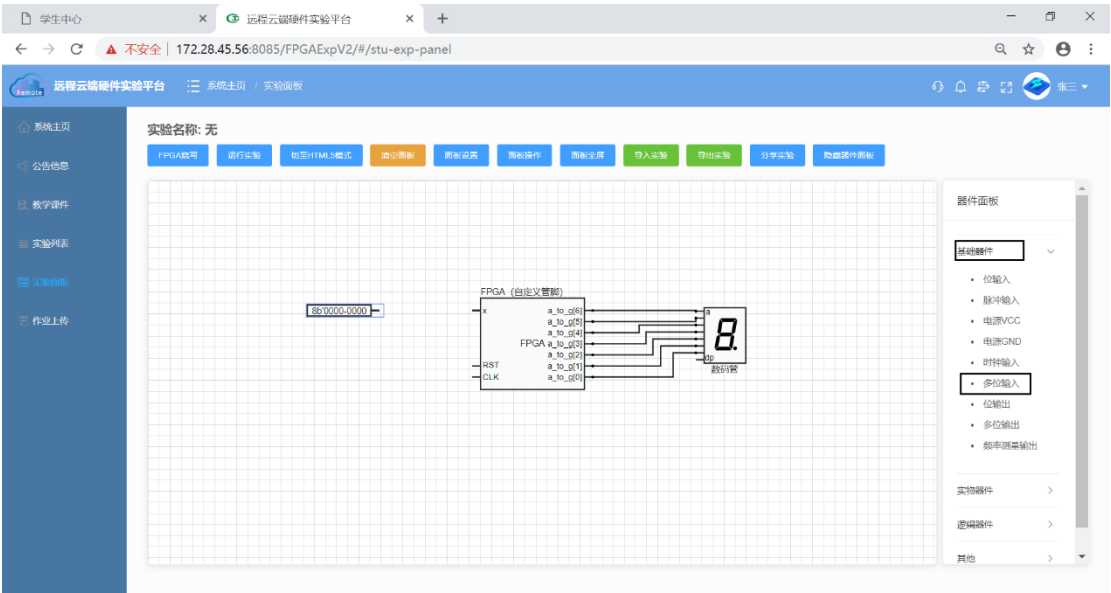


图 1-46 添加多位输入器件

(2). 右键单击“多位输入”器件，在弹出的菜单中选择“编辑参数”进行配置。其



中，“BIT 位数”设置为 4，“显示格式”设置为二进制，“数值设定”设置为 3，表示将 3 作为输入通过端口 x 送入 FPGA，单击确定按钮按成配置，如图 1-47 所示。

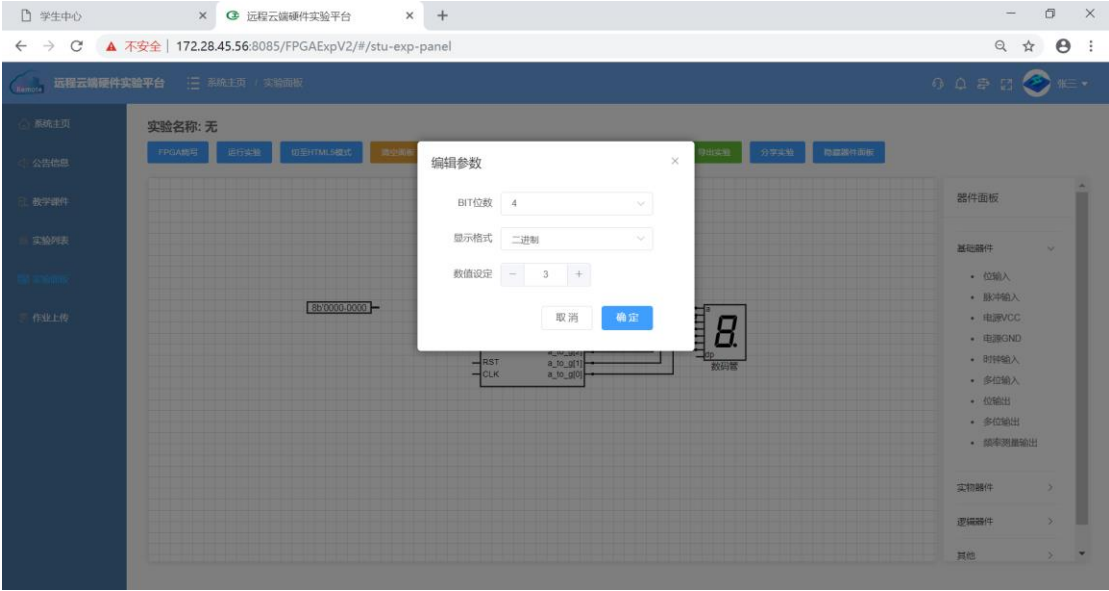


图 1-47 配置多位输入器件

(3). 按照图 1-48 所示，将 FPGA 的输入管脚与多位输入器件连接。至此，针对 7 段数码管显示译码器的验证平台已搭建完毕。

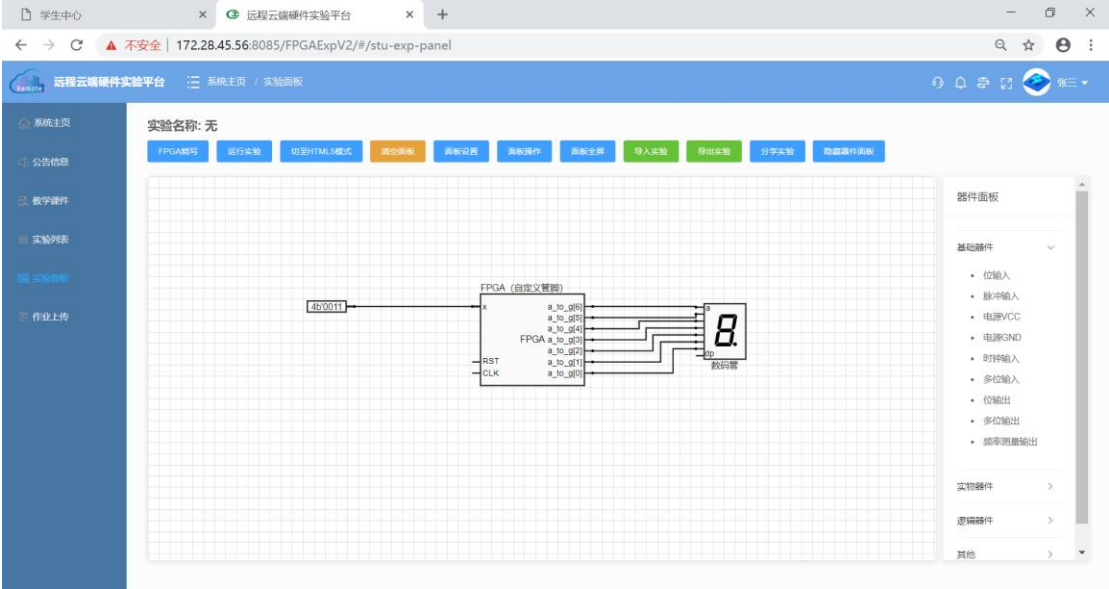


图 1-48 连接 FPGA 和多位输入器件

(4). 在实验图纸上方的功能栏中单击“导出实验”，可完成验证平台的保存，

如图 1-49 所示。后续可通过功能栏中的“导入实验”重新加载测试平台。

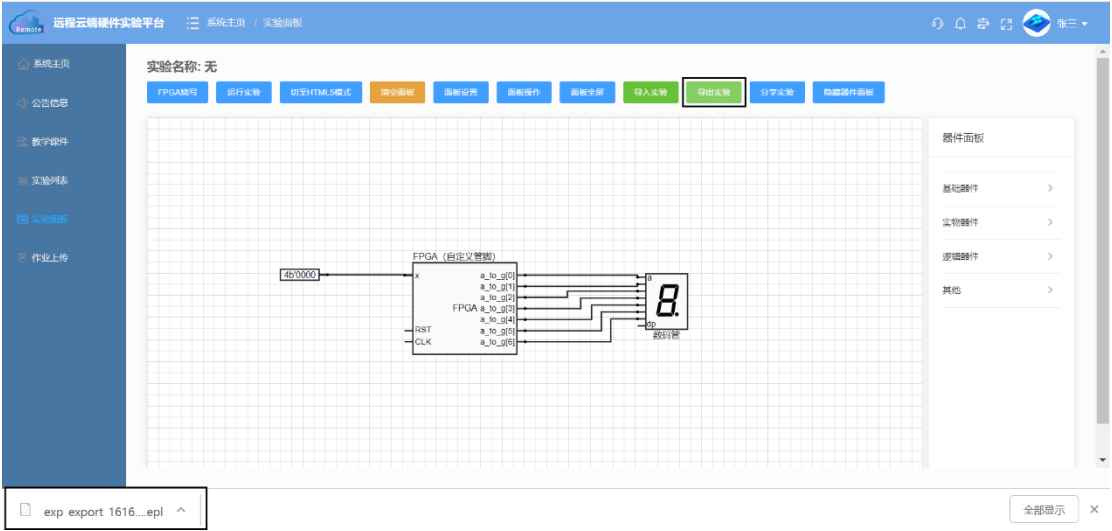
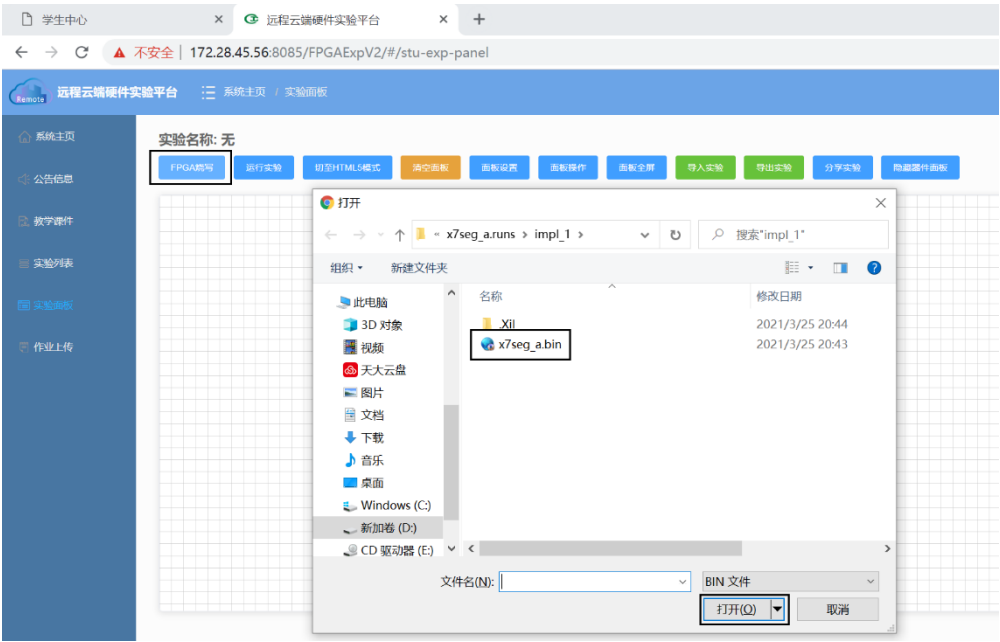


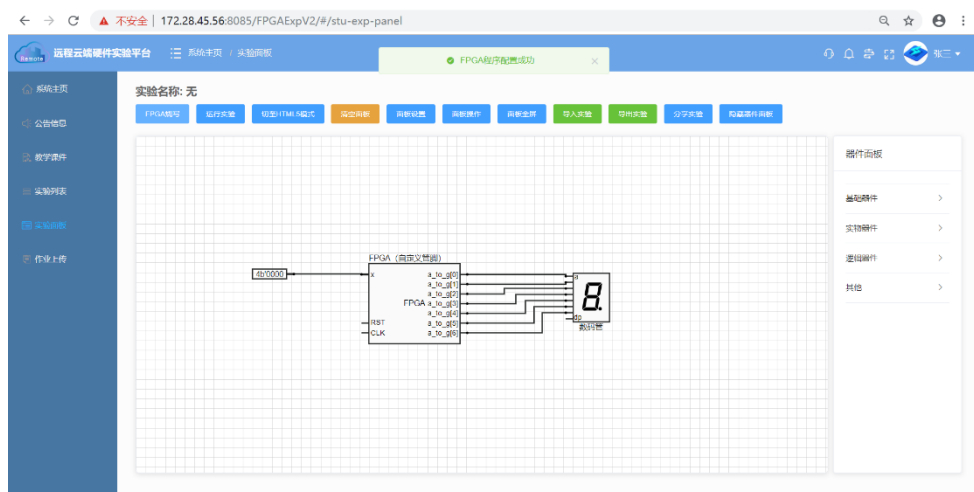
图 1-49 导出实验

## 5. 烧写 bin 文件进行功能验证

(1). 在实验图纸上方的功能栏中单击“FPGA 烧写”，在弹出的窗口中进入“工程路径\x7seg\_a\x7seg\_a.runs\impl\_1”目录，选中生成的“x7seg\_a.bin”文件，单击“打开”按钮，完成 FPGA 烧写，如图 1-50(a)所示。如果烧写成功，则显示“FPGA 程序配置成功”信息，如图 1-50(b)所示。



(a)



(b)

图 1-50 烧写 bin 文件

(2). 在实验图纸上方的功能栏中单击“运行实验”，进行上板验证，如图 1-51 所示。可以看出，当输入为 3 时，七段数码管上也显示出 3 字样，与预想一致。可以更改多位输入器件的值，对所设计的七段数码管显示译码器的功能进行进一步验证。

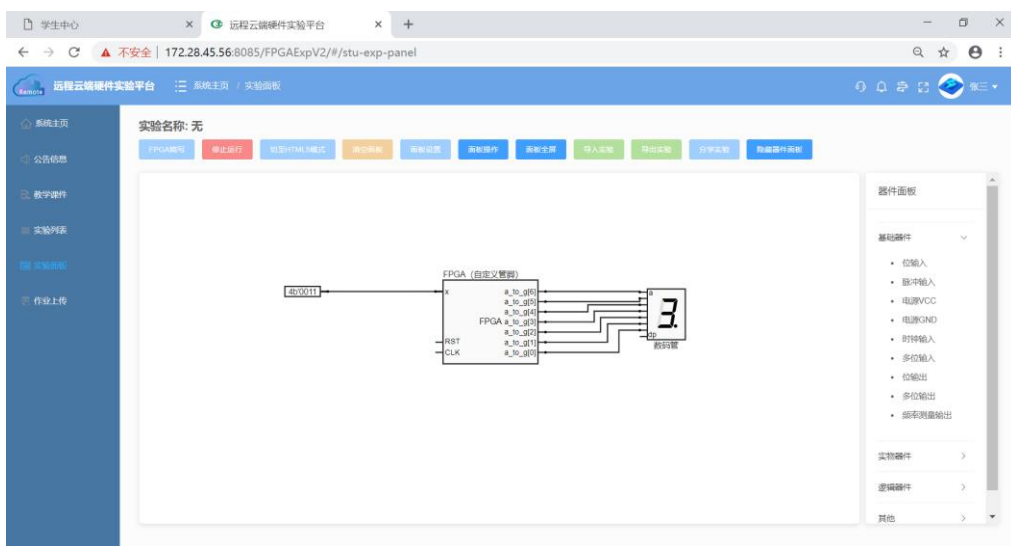


图 1-51 运行实验

注意：在实验图纸中为 FPGA 逻辑器件添加输入/输出管脚时，管脚名可以与 HDL 设计中的输入/输出端口名不同，因为绑定关系是通过 FPGA 管脚序号和约束文件中 FPGA 管脚名称确定的，而不是通过名字确定的。