



# DIGITAL DESIGN

LAB4 PARAMETER OF VERILOG & PACKAGE CUSTOMIZABLE IP CORE & USE IT

2020 FALL TERM

# LAB4

- Parameter in Verilog
- Package Customizable IP CORE
- Use The Customizable IP CORE to Do The Design

# PARAMETER

- **Parameter** is used to improve the readability and maintainability of code.
  - Used to define constants, such as delay and width variables, which is equivalent to defining an identifier representing a constant, also known as a symbolic constant.
  - They're run-time constants, and can be overridden when instantiate the module.
  - Syntax for parameter definitions :

```
parameter param_name1 = value/expression,  
           param_name2 = value/expression, ...
```

    - Use **comma** to separate different parameter definitions.
- The definition of the parameter is partial and **valid only in the current module**.
- The parameter definition can use the previously defined integer and real parameter.
- *All the following illustration base on the IP wrapper of NOR gate*

# PARAMETER DEFINITION(1)

```
module norgate
```

```
  #(parameter Port_Num = 2,WIDTH = 1) (
```

```
    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
```

## PARAMETER DEFINITION(2)

No #

```
module norgate(a, b, c, d, e, f, g, h, q);
```

```
    parameter Port_Num = 2, WIDTH = 1;
```

```
    input[WIDTH - 1 : 0] a, b, c, d, e, f, g, h;
```

```
    output[WIDTH - 1 : 0]q;
```

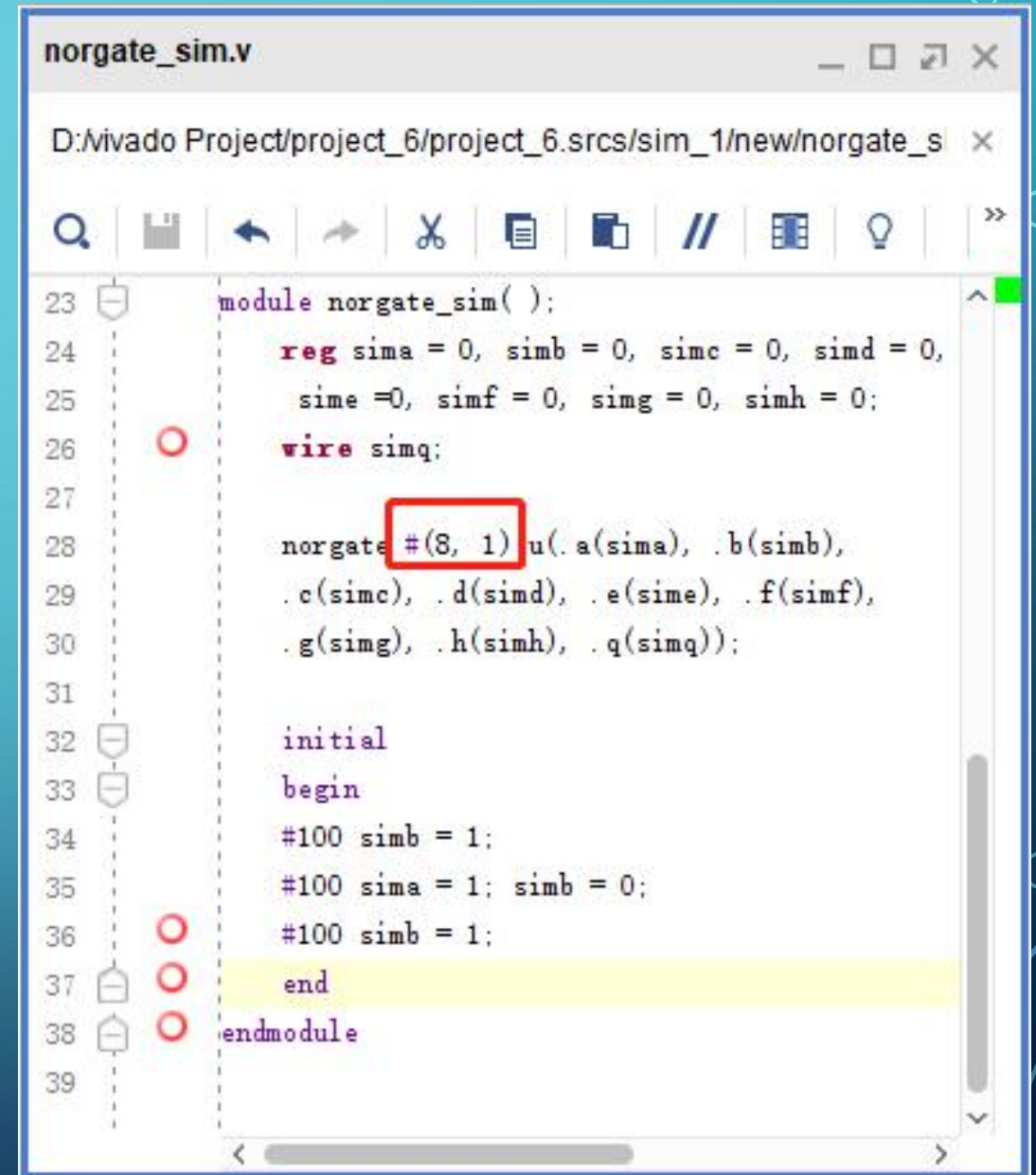
```
    assign q = ~ (a | b | c | d | e | f | g | h);
```

```
endmodule
```

# PARAMETER OVERRIDE

- The value of the parameter can be overridden when instantiate the module designed in the design source.

when instantiate the module andgate, we customize the Port\_Num as 8 and WIDTH as 1.



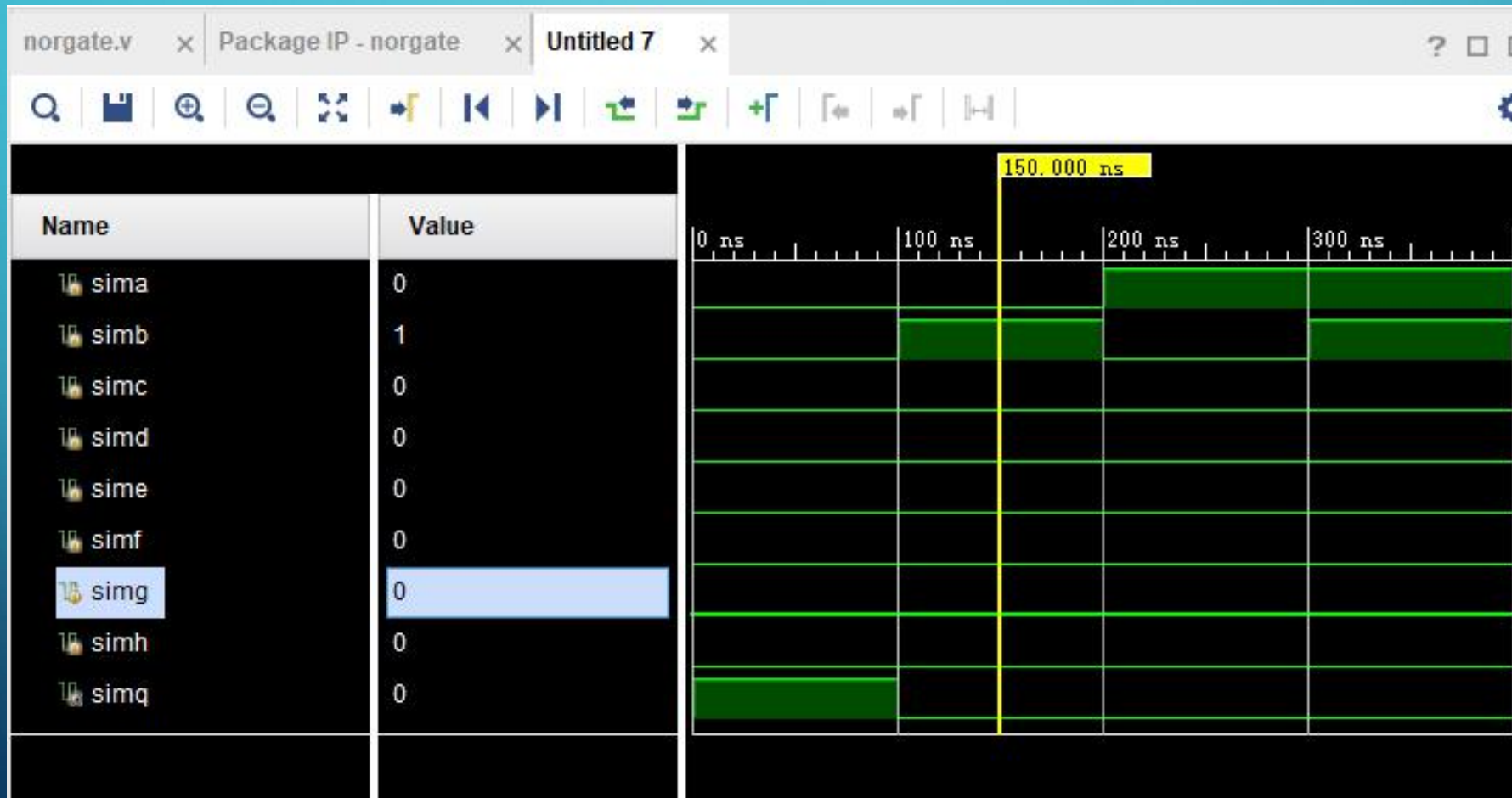
The screenshot shows a Verilog code editor window titled "norgate\_sim.v". The code defines a module "norgate\_sim" with several registers and a wire. A module instantiation "norgate #(8, 1) u" is shown, where the parameters "(8, 1)" are highlighted with a red box. The code also includes an initial block with timing delays and assignments.

```
23 module norgate_sim( );
24     reg sima = 0, simb = 0, simc = 0, simd = 0,
25     sime = 0, simf = 0, simg = 0, simh = 0;
26     wire simq;
27
28     norgate #(8, 1) u(.a(sima), .b(simb),
29     .c(simc), .d(simd), .e(sime), .f(simf),
30     .g(simg), .h(simh), .q(simq));
31
32     initial
33     begin
34         #100 simb = 1;
35         #100 sima = 1; simb = 0;
36         #100 simb = 1;
37     end
38 endmodule
39
```



- Using simulation to verify the function of the module in the design file.

Once the simulation result is correct we can start IP packaging.



# IP CORE

- **IP core**, or **IP block** is a reusable unit of logic, cell, or integrated circuit (commonly called a "chip") layout design that is the intellectual property of one party. IP cores may be licensed to another party or can be owned and used by a single party alone. The term is derived from the licensing of the patent and/or source code copyright that exist in the design. IP cores can be used as building blocks within application-specific integrated circuit (ASIC) designs or field-programmable gate array (FPGA) logic designs.
- IP cores (software) are typically offered as synthesizable RTL. Synthesizable cores are delivered in a hardware description language such as Verilog or VHSIC hardware description language (VHDL).  
----- from wiki

- A IP created by using vivado

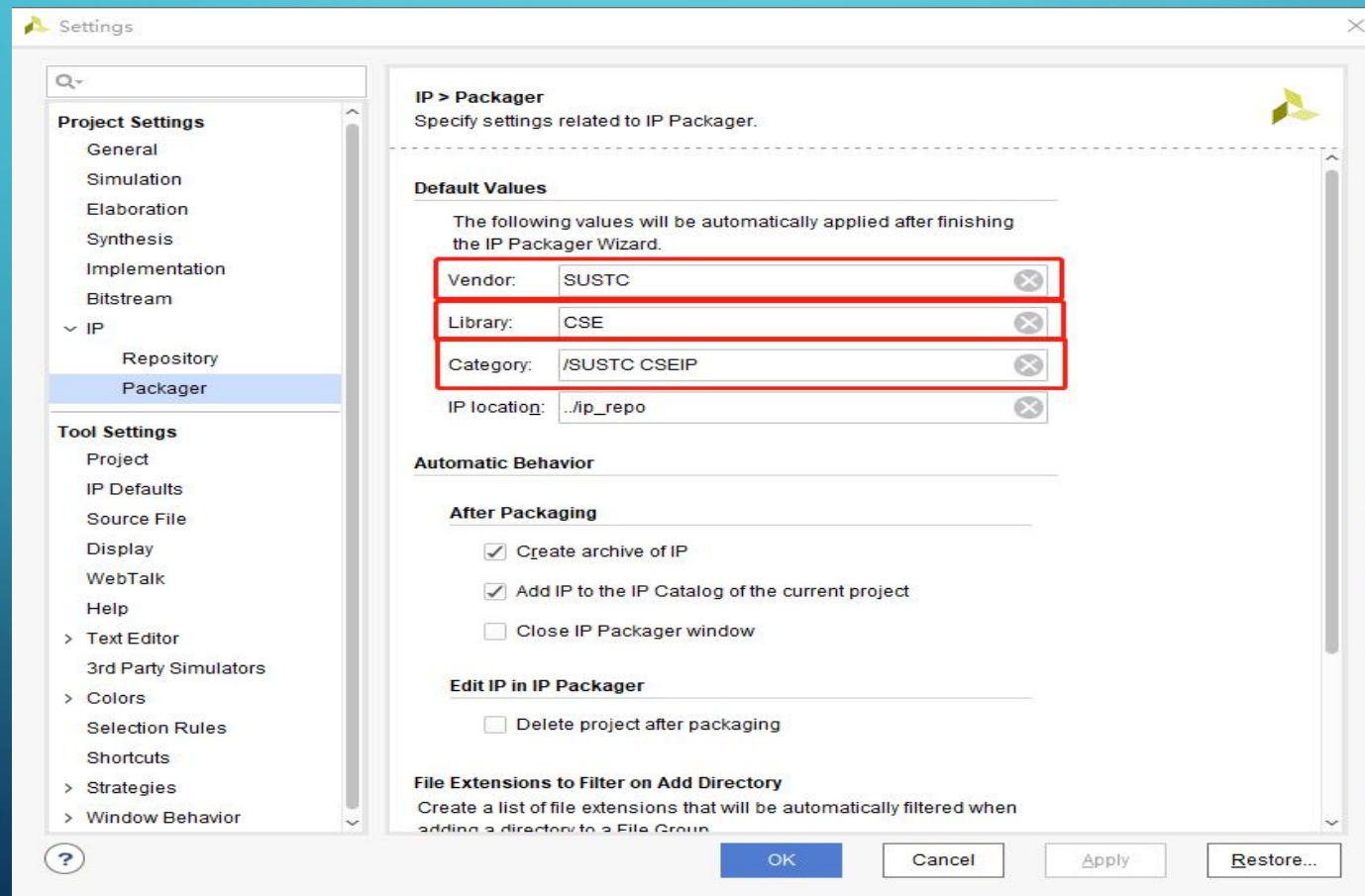


名称	类型
xgui	文件夹
component	XML 文档
orgate.v	V 文件



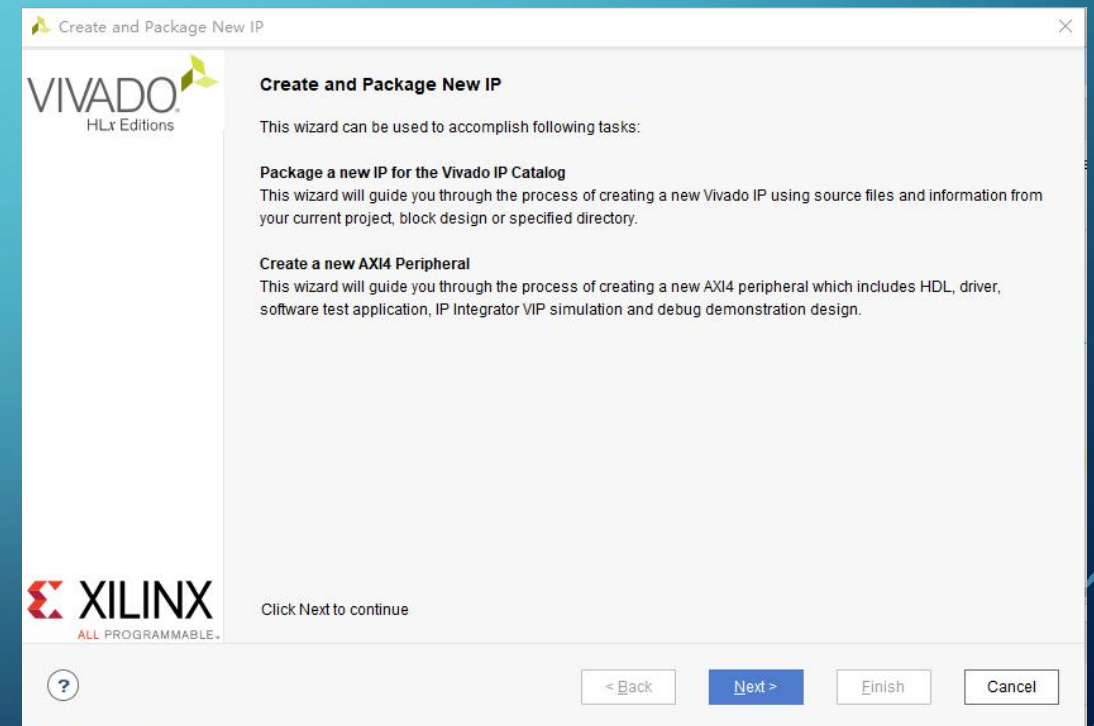
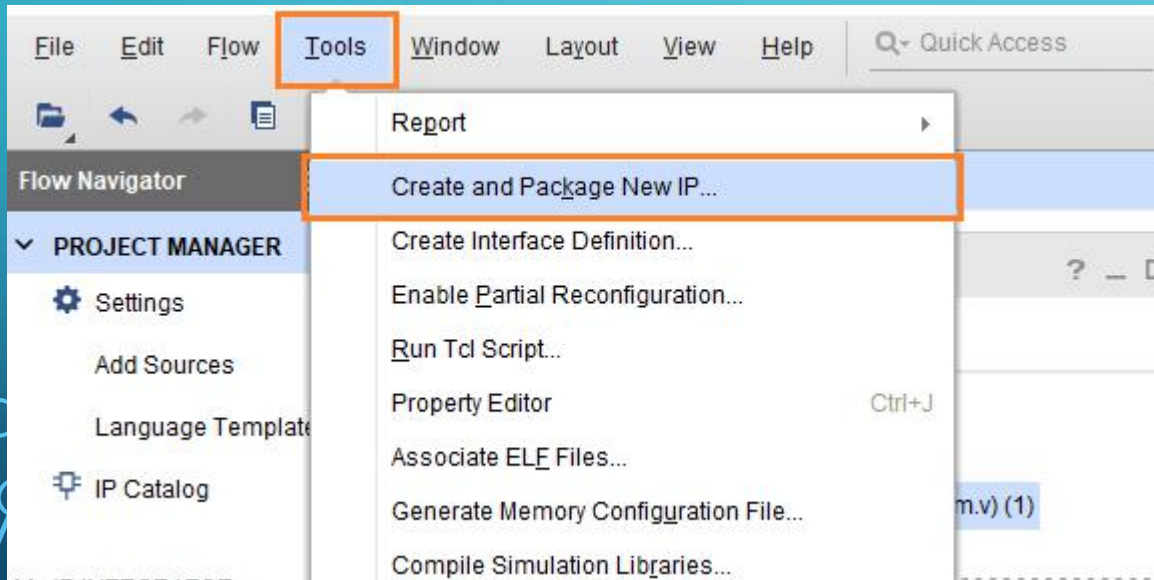
# PACKAGE CUSTOMIZABLE IP CORE(1)

- Click the **Flow Navigator**—> **Project Manager**—> **Setting**—> **IP**—> **Packager**
  - Modify the **vendor** to SUSTC, and **Library** to CSE. These two properties will be part of the IP file name.
  - Modify the **category** to SUSTC CSEIP. This property will be displayed as category name.



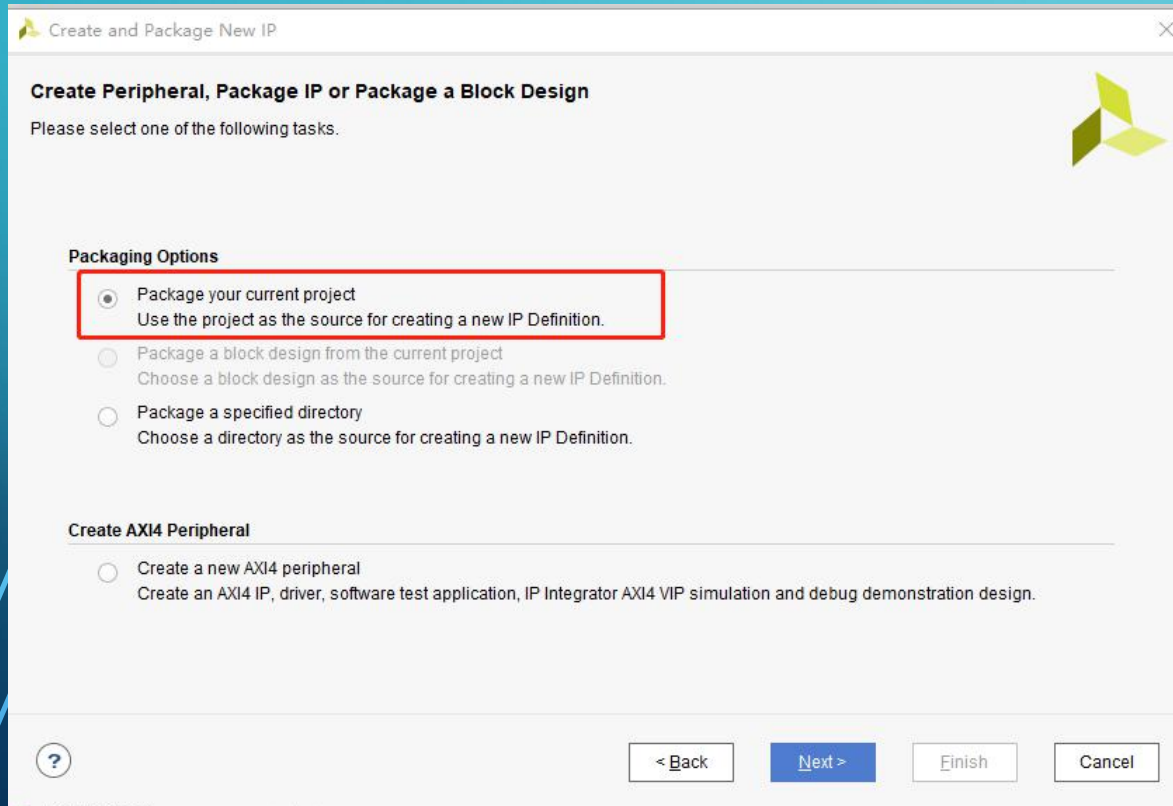
# PACKAGE CUSTOMIZABLE IP CORE(2)

- Click the **Tools** —> **Create and Package New IP...**
- Follow the wizard



# PACKAGE CUSTOMIZABLE IP CORE(3)

- Choose *package your current project*
  - choose ‘*include .xci file*’



The screenshot shows the first step of the 'Create and Package New IP' wizard. The title bar reads 'Create and Package New IP'. The main heading is 'Create Peripheral, Package IP or Package a Block Design' with a subtext 'Please select one of the following tasks.' Below this, there are three radio button options under the heading 'Packaging Options'. The first option, 'Package your current project', is selected and highlighted with a red rectangle; its description is 'Use the project as the source for creating a new IP Definition.' The other two options are 'Package a block design from the current project' and 'Package a specified directory'. At the bottom, there is a section for 'Create AXI4 Peripheral' with one option: 'Create a new AXI4 peripheral'. Navigation buttons at the bottom include '< Back', 'Next >', 'Finish', and 'Cancel'.

Create and Package New IP

Create Peripheral, Package IP or Package a Block Design

Please select one of the following tasks.

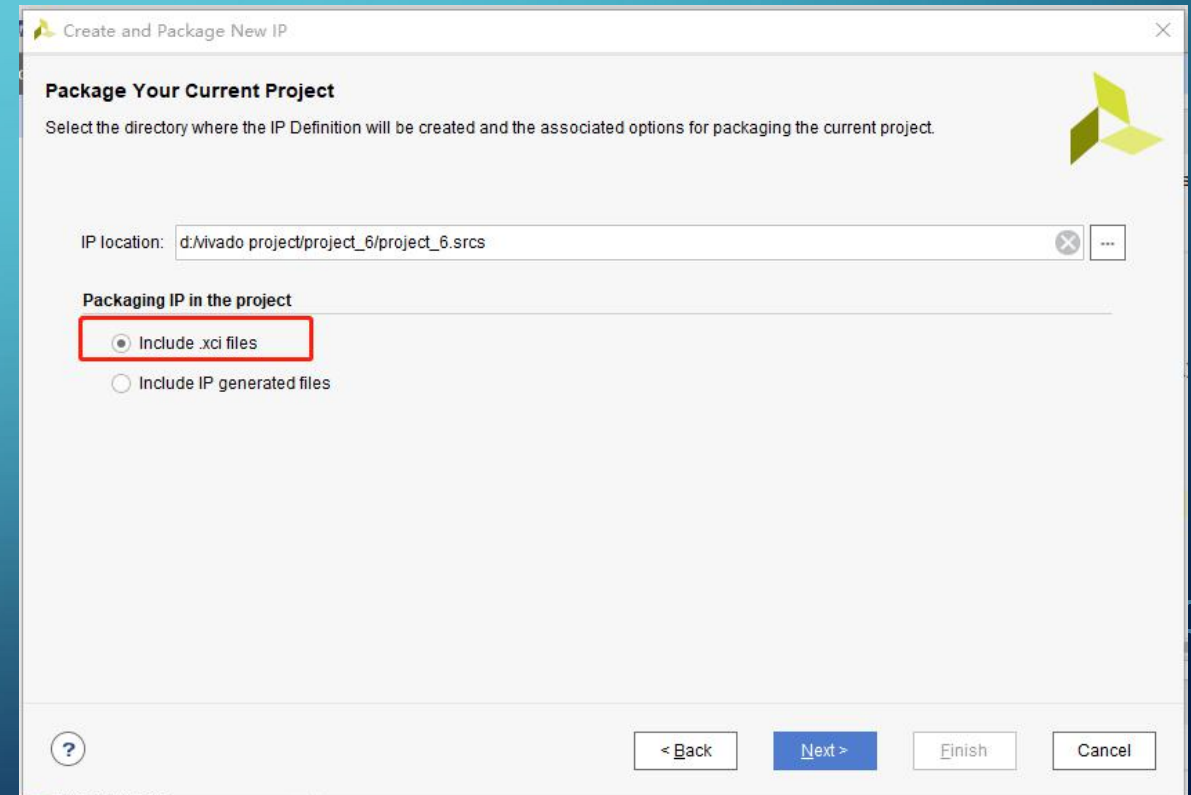
**Packaging Options**

- ☒ Package your current project  
Use the project as the source for creating a new IP Definition.
- ☐ Package a block design from the current project  
Choose a block design as the source for creating a new IP Definition.
- ☐ Package a specified directory  
Choose a directory as the source for creating a new IP Definition.

**Create AXI4 Peripheral**

- ☐ Create a new AXI4 peripheral  
Create an AXI4 IP, driver, software test application, IP Integrator AXI4 VIP simulation and debug demonstration design.

< Back Next > Finish Cancel



The screenshot shows the second step of the 'Create and Package New IP' wizard. The title bar reads 'Create and Package New IP'. The main heading is 'Package Your Current Project' with a subtext 'Select the directory where the IP Definition will be created and the associated options for packaging the current project.' Below this, there is a text field for 'IP location:' containing the path 'd:\vivado project\project\_6\project\_6.srcs'. Under the heading 'Packaging IP in the project', there are two radio button options. The first option, 'Include .xci files', is selected and highlighted with a red rectangle. The second option is 'Include IP generated files'. Navigation buttons at the bottom include '?', '< Back', 'Next >', 'Finish', and 'Cancel'.

Create and Package New IP

Package Your Current Project

Select the directory where the IP Definition will be created and the associated options for packaging the current project.

IP location: d:\vivado project\project\_6\project\_6.srcs

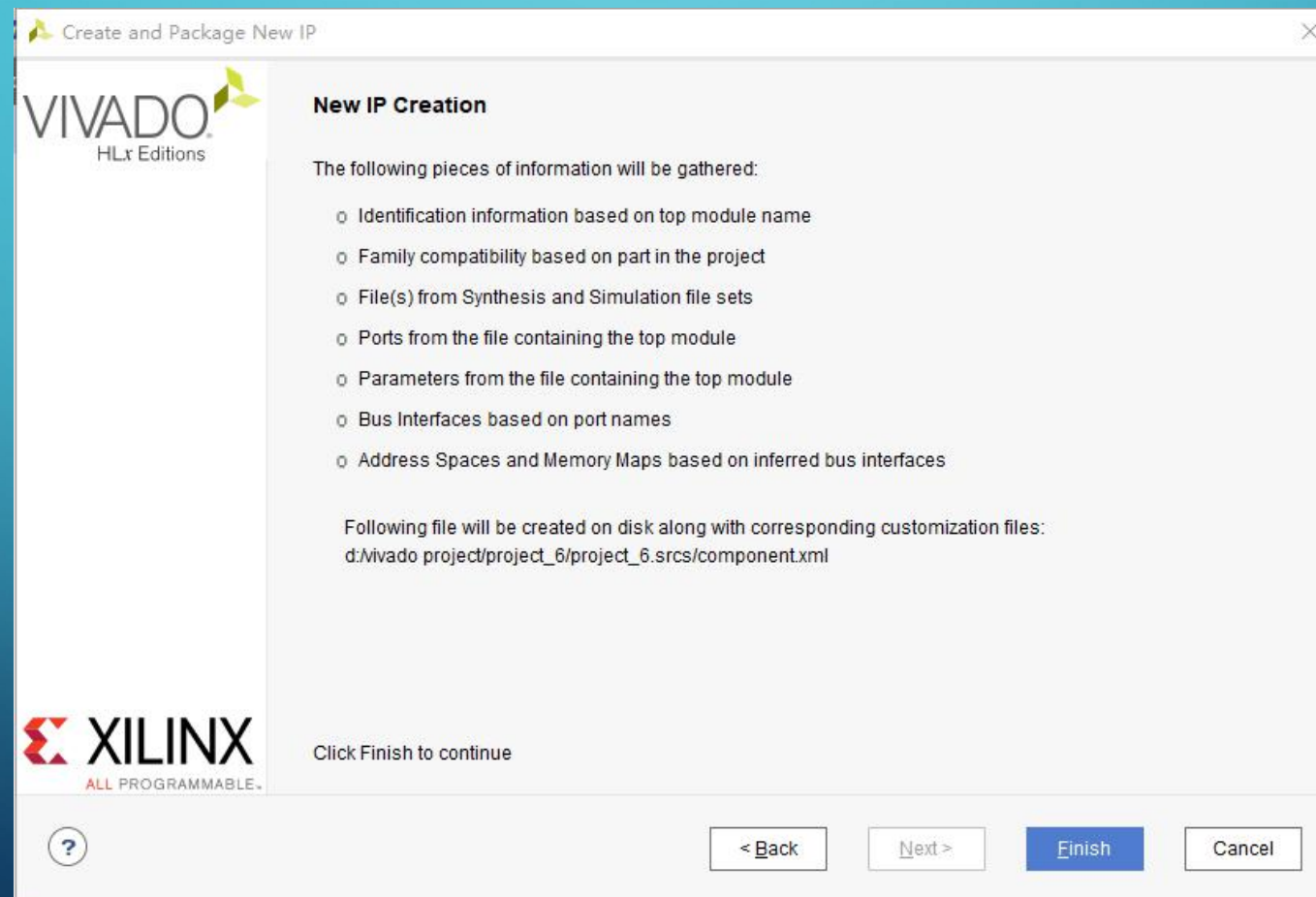
**Packaging IP in the project**

- ☒ Include .xci files
- ☐ Include IP generated files

? < Back Next > Finish Cancel

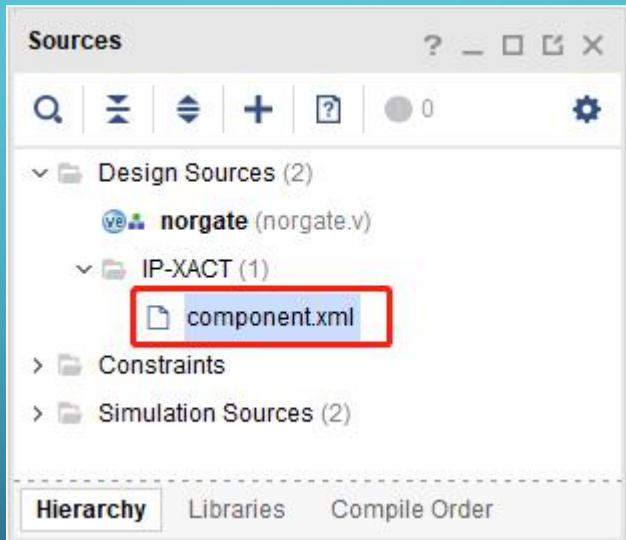
# PACKAGE CUSTOMIZABLE IP CORE(4)

- Click finish here



# PACKAGE CUSTOMIZABLE IP CORE(5)

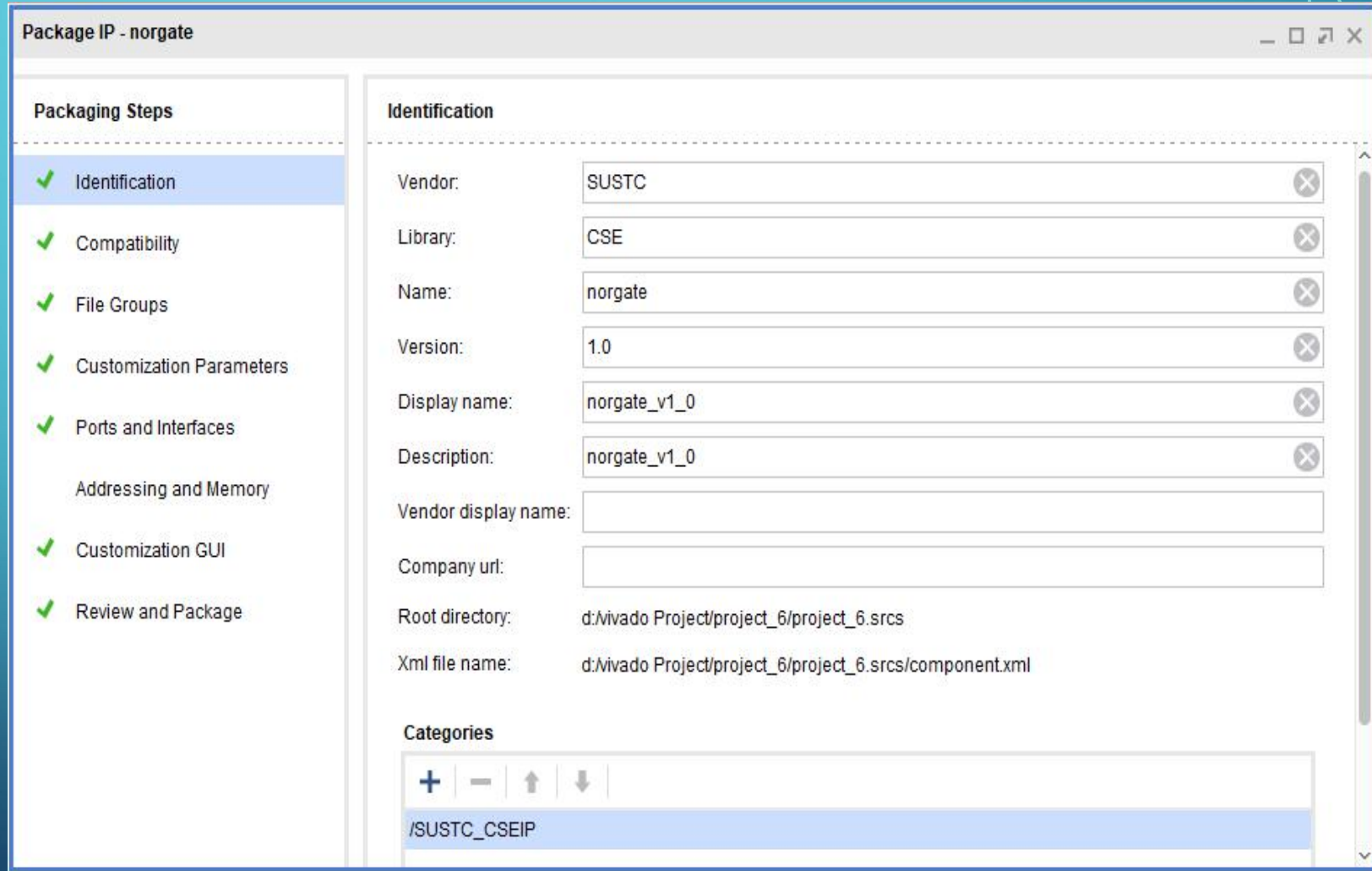
- A file named as **component.xml** will be added to your design sources automatically





# CUSTOMIZATION IP CORE(1)-IDENTIFICATION

- If you want to modify those properties, you can do it. Here we just keep it as it is.



The screenshot shows a software window titled "Package IP - norgate". On the left is a "Packaging Steps" sidebar with a list of steps, each preceded by a green checkmark: Identification, Compatibility, File Groups, Customization Parameters, Ports and Interfaces, Addressing and Memory, Customization GUI, and Review and Package. The "Identification" step is currently selected and highlighted. The main area of the window is titled "Identification" and contains several input fields with their corresponding values: Vendor (SUSTC), Library (CSE), Name (norgate), Version (1.0), Display name (norgate\_v1\_0), Description (norgate\_v1\_0), Vendor display name (empty), Company url (empty), Root directory (d:\vivado Project\project\_6\project\_6.srscs), and Xml file name (d:\vivado Project\project\_6\project\_6.srscs\component.xml). At the bottom of the main area is a "Categories" section with a list box containing the entry "/SUSTC\_CSEIP", which is currently selected. Above the list box are four small icons: a plus sign, a minus sign, an up arrow, and a down arrow.

Field	Value
Vendor	SUSTC
Library	CSE
Name	norgate
Version	1.0
Display name	norgate_v1_0
Description	norgate_v1_0
Vendor display name	
Company url	
Root directory	d:\vivado Project\project_6\project_6.srscs
Xml file name	d:\vivado Project\project_6\project_6.srscs\component.xml

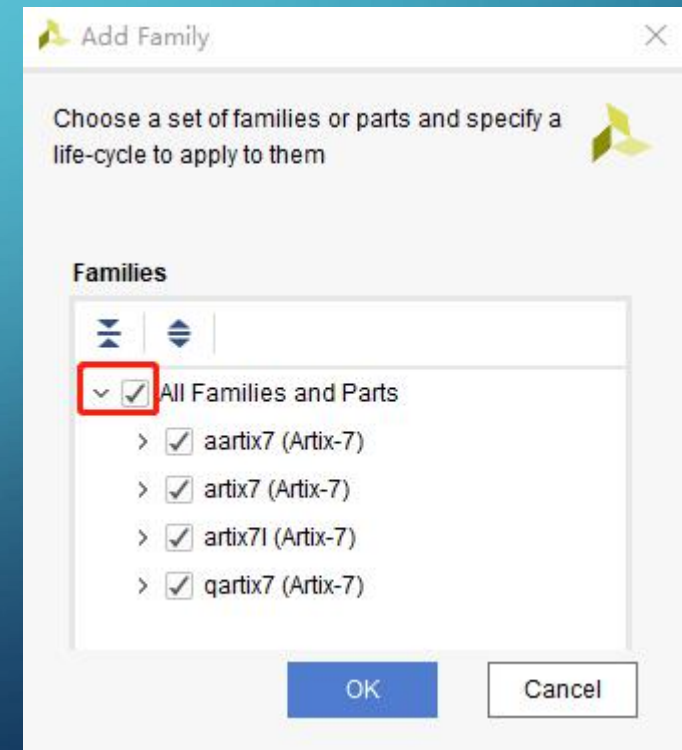
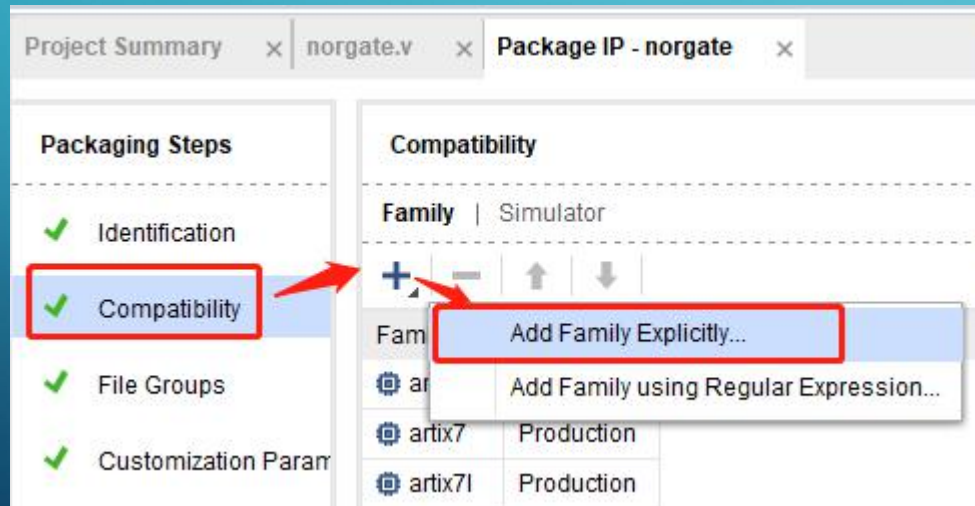
**Categories**

- /SUSTC\_CSEIP



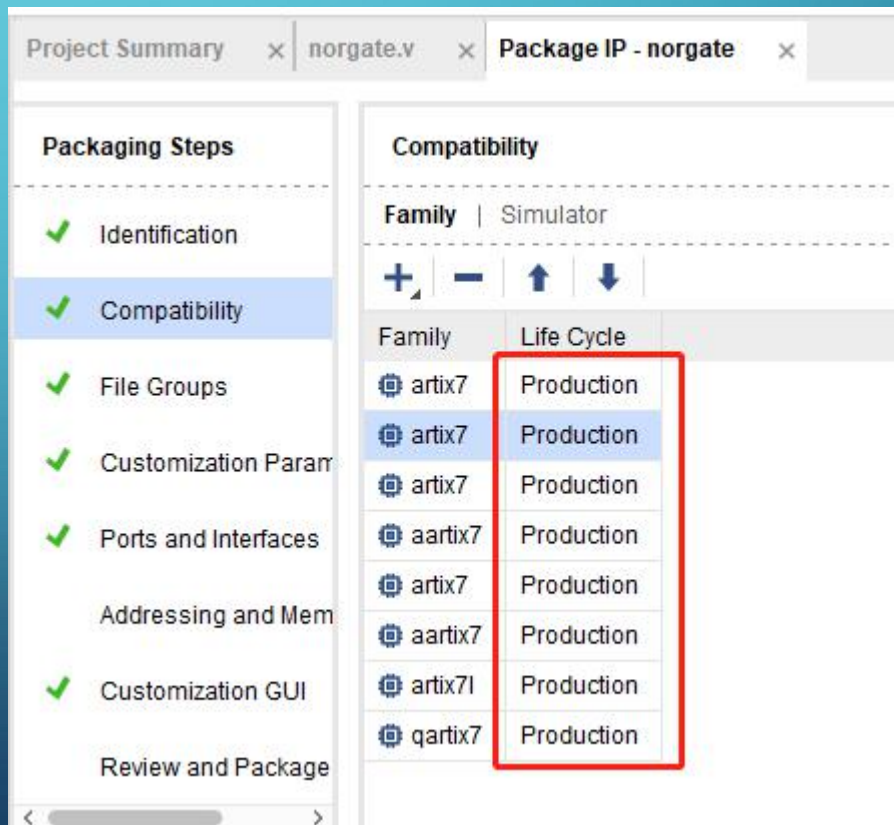
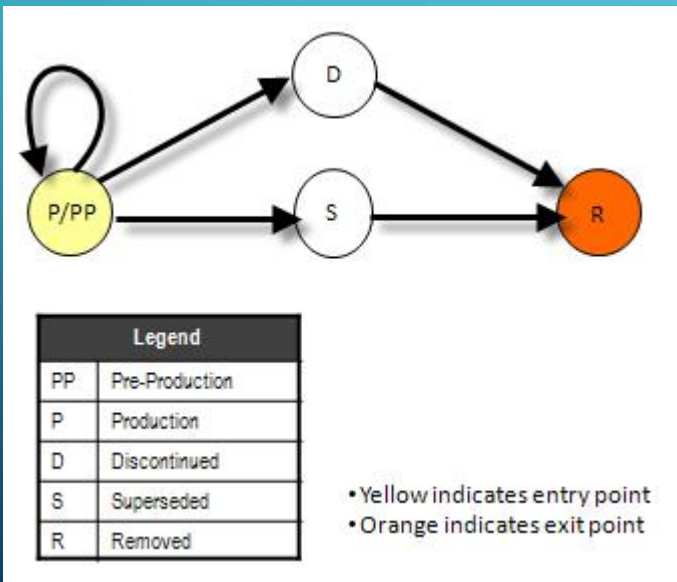
# CUSTOMIZATION IP CORE(2)-COMPATIBILITY(1)

- If you want the IP be valid with other FPGA Family, we can add them all.



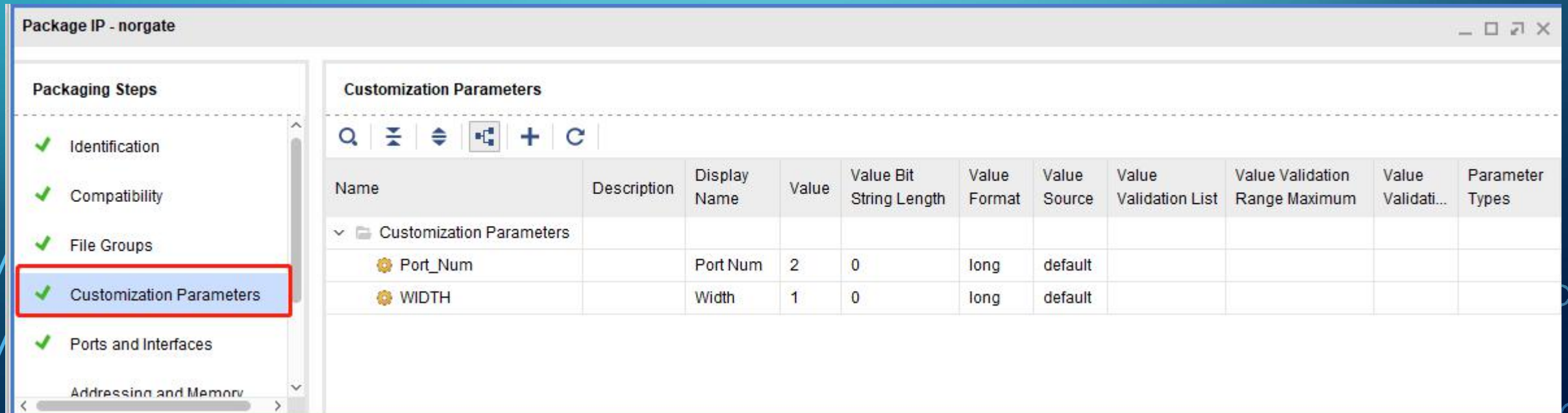
# CUSTOMIZATION IP CORE(2)-COMPATIBILITY(2)

- As for the *life circle* , we choose 'Production'.
- A Production IP core is one which is provided for general public release, and has been verified using production speed files.



# CUSTOMIZATION IP CORE(3)- CUSTOMIZATION PARAMETERS(1)

- Here we can customize the defined parameters, double-click the parameter name in the table, you can edit the relevant information.



The screenshot displays the 'Package IP - norgate' application window. On the left, a 'Packaging Steps' sidebar lists several steps: Identification, Compatibility, File Groups, Customization Parameters (highlighted with a red box), Ports and Interfaces, and Addressing and Memory. The main area is titled 'Customization Parameters' and contains a table with the following columns: Name, Description, Display Name, Value, Value Bit String Length, Value Format, Value Source, Value Validation List, Value Validation Range Maximum, Value Validation, and Parameter Types. The table lists two parameters: 'Port\_Num' and 'WIDTH'. The 'Port\_Num' parameter has a value of 2 and a bit string length of 0. The 'WIDTH' parameter has a value of 1 and a bit string length of 0. Both parameters have a 'long' format and a 'default' source.

Name	Description	Display Name	Value	Value Bit String Length	Value Format	Value Source	Value Validation List	Value Validation Range Maximum	Value Validation	Parameter Types
Customization Parameters										
Port_Num		Port Num	2	0	long	default				
WIDTH		Width	1	0	long	default				

# CUSTOMIZATION IP CORE(3)- CUSTOMIZATION PARAMETERS(2)

- For a NOR gate, there are at least two inputs; here eight inputs has been defined.
  - for the value of parameter 'Port\_Num', the minimum is 2, and maximum is 8.
- As for the parameter WIDTH, the minimum is 1, and maximum is 8.

The screenshot shows the 'Edit IP Parameter' dialog for the parameter 'Port\_Num'. The dialog has a title bar with a yellow icon and a close button. The main text says 'Use the options below to customize how the parameter will appear in the Customization GUI for users of the IP.' The fields are: Name: Port\_Num, Visible in Customization GUI: checked, Show Name: checked, Display Name: Port Num, Tooltip: Port Num, Format: long, Editable: Yes, Dependency: No, and Specify Range: checked. The 'Specify Range' section is highlighted with a red box, showing Type: Range of integ..., Minimum: 2, and Maximum: 8. The 'Show Range' checkbox is also checked. The dialog has OK and Cancel buttons at the bottom.

Use the options below to customize how the parameter will appear in the Customization GUI for users of the IP.

Name: Port\_Num

☒ Visible in Customization GUI

☒ Show Name

Display Name: Port Num

Tooltip: Port Num

Format: long

Editable: Yes

Dependency: No

☒ Specify Range

Type: Range of integ...

Minimum: 2

Maximum: 8

☒ Show Range

OK Cancel

The screenshot shows the 'Edit IP Parameter' dialog for the parameter 'WIDTH'. The dialog has a title bar with a yellow icon and a close button. The main text says 'Use the options below to customize how the parameter will appear in the Customization GUI for users of the IP.' The fields are: Name: WIDTH, Visible in Customization GUI: checked, Show Name: checked, Display Name: Width, Tooltip: Width, Format: long, Editable: Yes, Dependency: No, and Specify Range: checked. The 'Specify Range' section is highlighted with a red box, showing Type: Range of integ..., Minimum: 1, and Maximum: 8. The 'Show Range' checkbox is also checked. The dialog has OK and Cancel buttons at the bottom.

Use the options below to customize how the parameter will appear in the Customization GUI for users of the IP.

Name: WIDTH

☒ Visible in Customization GUI

☒ Show Name

Display Name: Width

Tooltip: Width

Format: long

Editable: Yes

Dependency: No

☒ Specify Range

Type: Range of integ...

Minimum: 1

Maximum: 8

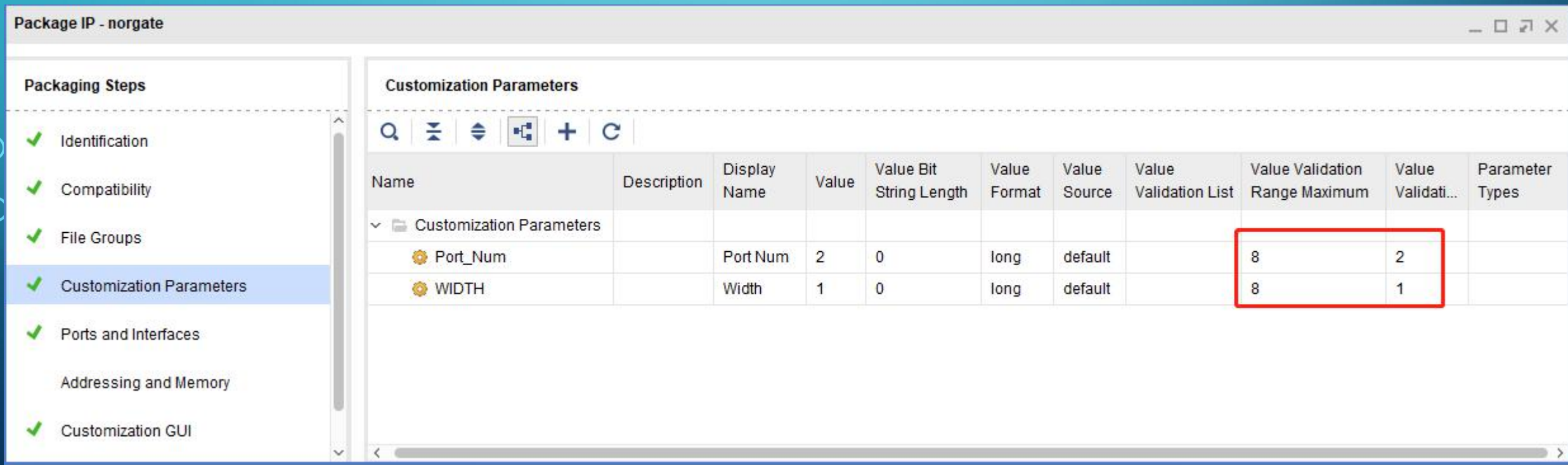
☒ Show Range

OK Cancel



# CUSTOMIZATION IP CORE(3)- CUSTOMIZATION PARAMETERS(3)

- After the setting , we can see the following changes which are marked by red box.



The screenshot shows the 'Package IP - norgate' application window. On the left, a 'Packaging Steps' sidebar lists various steps, with 'Customization Parameters' currently selected. The main area displays a table of 'Customization Parameters'.

Name	Description	Display Name	Value	Value Bit String Length	Value Format	Value Source	Value Validation List	Value Validation Range Maximum	Value Validation List	Parameter Types
Customization Parameters										
Port_Num		Port Num	2	0	long	default		8	2	
WIDTH		Width	1	0	long	default		8	1	

A red box highlights the 'Value Validation Range Maximum' and 'Value Validation List' columns for the 'Port\_Num' and 'WIDTH' parameters, showing values 8 and 2 for Port\_Num, and 8 and 1 for WIDTH.

# CUSTOMIZATION IP CORE(4)- PORTS AND INTERFACES(1)

- Here we set properties of ports. Double-click the port name in the table, you can edit the relevant information.

The screenshot displays the 'Package IP - norgate' application window. On the left, a 'Packaging Steps' sidebar lists several steps, with 'Ports and Interfaces' highlighted by a red rectangle. The main area on the right is titled 'Ports and Interfaces' and contains a table with the following columns: Name, Interface Mode, Enablement Dependency, Is Declaration, Direction, Driver Value, Size Left, Size Right, Size Left Dependency, Size Right Dependency, and T. The table lists nine ports, labeled 'a' through 'q'. Ports 'a' through 'h' are configured as 'in' (input) ports, while port 'q' is an 'out' (output) port. All ports have a 'Size Left' and 'Size Right' of 0 and a 'Size Left Dependency' of '(WIDTH - 1)'. The 'Is Declaration' column for all ports contains an unchecked checkbox.

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



# CUSTOMIZATION IP CORE(4)- PORTS AND INTERFACES(2)

- There are at least two input ports, so the port named 'a' and 'b' are **Mandatory** while others are **Optional**.
  - In this case, whether the ports named as c ~h is enabled is determined by the value of parameter Port\_Num.
- In a Nor gate , if the input port is disabled, the ' **Driver value** ' of that port should be 0.

The image displays two side-by-side screenshots of the 'Edit Port' dialog box, which is used to configure ports on an IP core. Both windows have a title bar with a close button and a small logo. The main instruction reads: 'Use the fields below to modify the port on your IP.'

**Left Screenshot (Port 'a'):**

- Name: a
- Driver value: (empty text box)
- Left size: (WIDTH - 1) (with a clear button 'X')
- Right size: 0 (with a clear button 'X')
- Port Presence: ☒ **Mandatory** ☐ Optional
- Text area: (empty)
- Example: \$my\_num\_var > 0 [more info](#)
- Buttons: OK, Cancel

**Right Screenshot (Port 'c'):**

- Name: c
- Driver value: 0 (with a clear button 'X')
- Left size: (WIDTH - 1) (with a clear button 'X')
- Right size: 0 (with a clear button 'X')
- Port Presence: ☐ Mandatory ☒ **Optional**
- Text area: \$Port\_Num > 2
- Example: \$my\_num\_var > 0 [more info](#)
- Buttons: OK, Cancel

# CUSTOMIZATION IP CORE(4)- PORTS AND INTERFACES(3)

Package IP - norgate

**Packaging Steps**

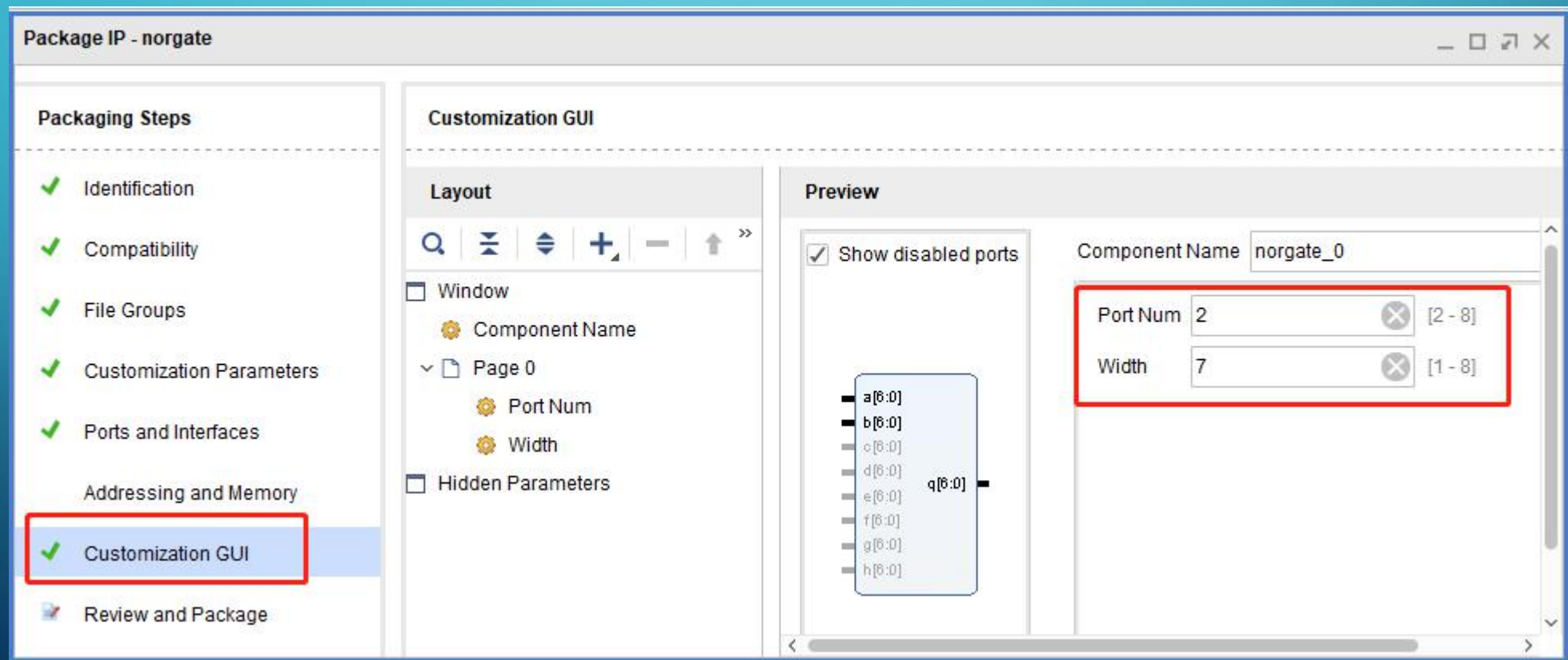
- ✓ Identification
- ✓ Compatibility
- ✓ File Groups
- ✓ Customization Parameters
- ✓ Ports and Interfaces**
- Addressing and Memory
- ✓ Customization GUI
- Review and Package

**Ports and Interfaces**

Name	Interface Mode	Enablement Dependency	Is Declaration	Direction	Driver Value	Size Left	Size Right	Size Left Dependency	Size Right Dependency
a			<input type="checkbox"/>	in		0	0	(WIDTH - 1)	
b			<input type="checkbox"/>	in		0	0	(WIDTH - 1)	
c		$\$Port\_Num > 2$	<input type="checkbox"/>	in	0	0	0	(WIDTH - 1)	
d		$\$Port\_Num > 3$	<input type="checkbox"/>	in	0	0	0	(WIDTH - 1)	
e		$\$Port\_Num > 4$	<input type="checkbox"/>	in	0	0	0	(WIDTH - 1)	
f		$\$Port\_Num > 5$	<input type="checkbox"/>	in	0	0	0	(WIDTH - 1)	
g		$\$Port\_Num > 6$	<input type="checkbox"/>	in	0	0	0	(WIDTH - 1)	
h		$\$Port\_Num > 7$	<input type="checkbox"/>	in	0	0	0	(WIDTH - 1)	
q			<input type="checkbox"/>	out		0	0	(WIDTH - 1)	

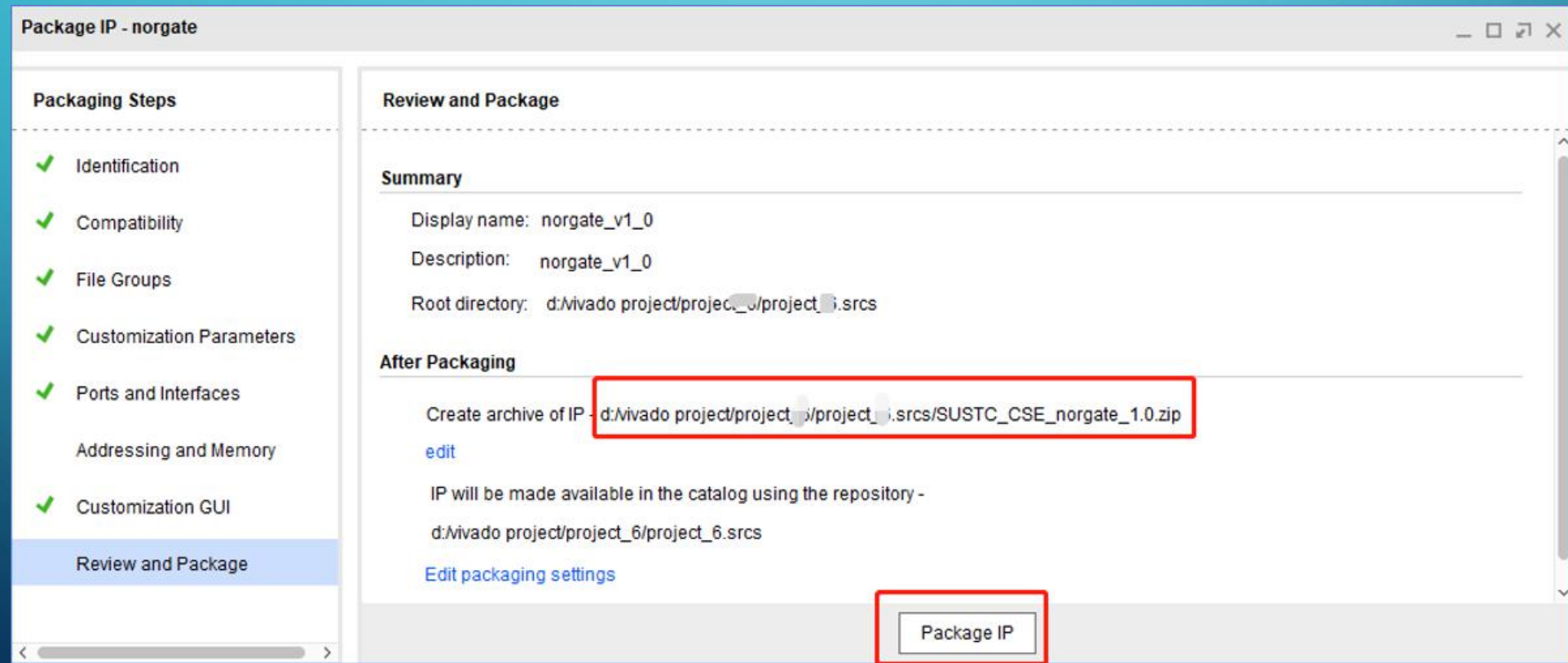
# CUSTOMIZATION IP CORE(5)- CUSTOMIZATION GUI

- Here we can verify the effect of parameter changes on IP encapsulation.



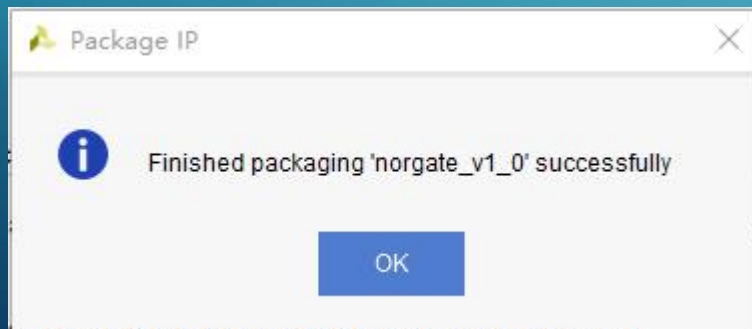
# CUSTOMIZATION IP CORE(6)- REVIEW AND PACKAGE

- Remember the file path where the IP core will be created. Click *Package IP*.



# PACKAGE CUSTOMIZABLE IP CORE

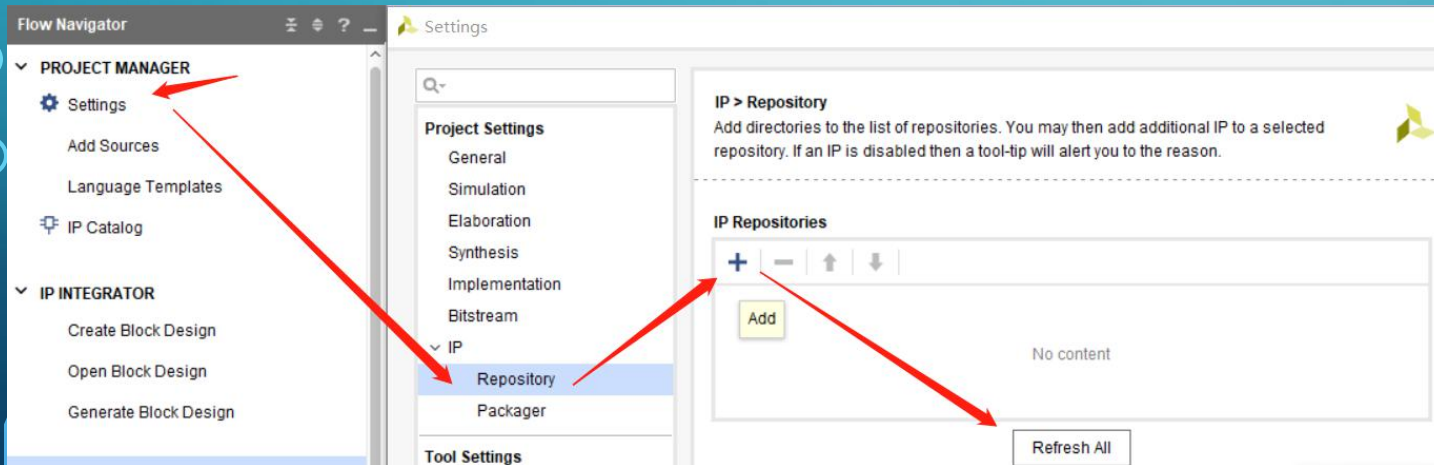
- If the packaging is success, You can find your IP core exactly under the file path showed before.
- Then you can put the IP core to your IPCore file, and decompress it, so you can used it in your following design.





# USING IP(1) - ADDING IP INTO PROJECT(1)

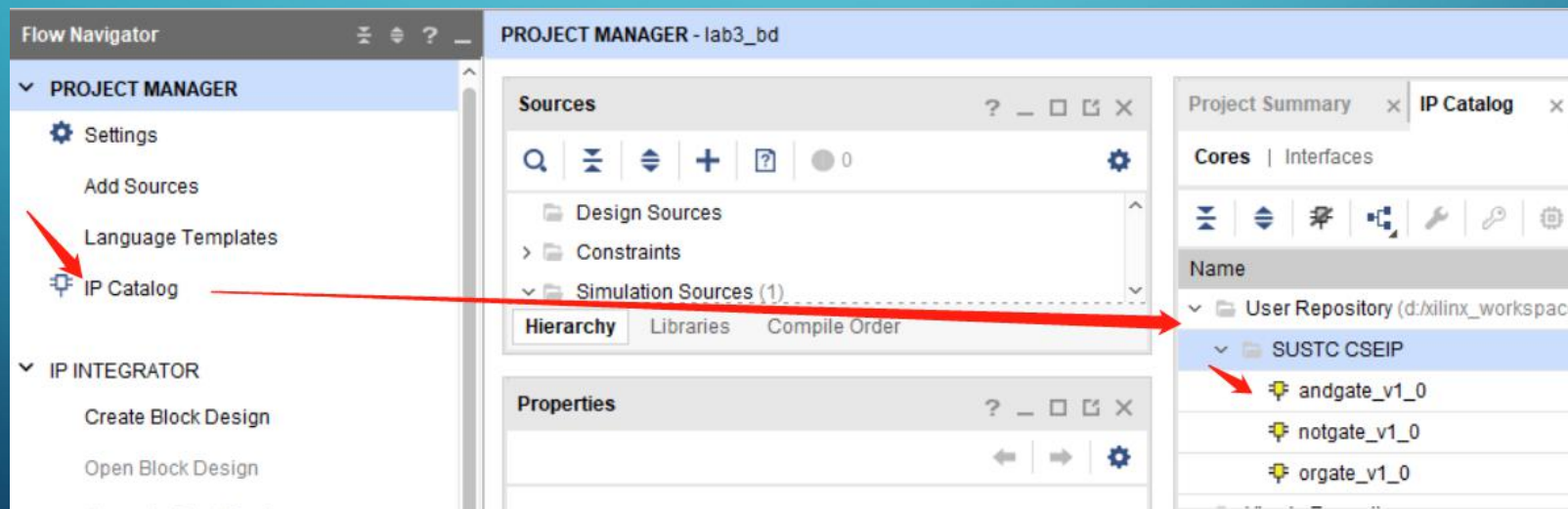
- Prepare for adding IP into current project:
  - 1. make you project to find the IP repositories
    - Flow navigator -> settings -> IP -> Repository (add directories to the list of repositories)
    - IP repositories is a place where stored some unzipped IPs





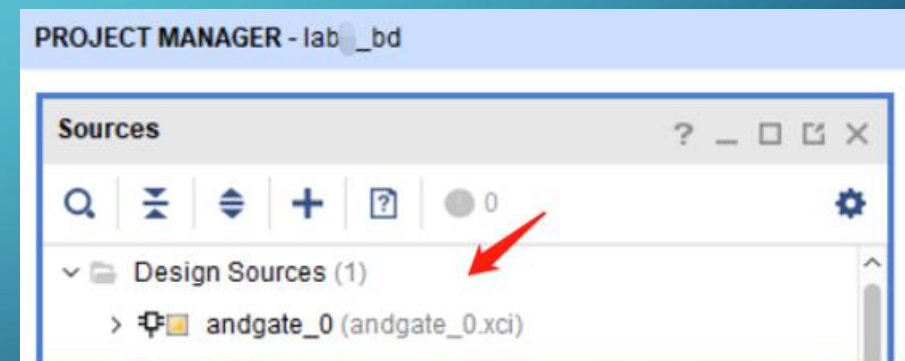
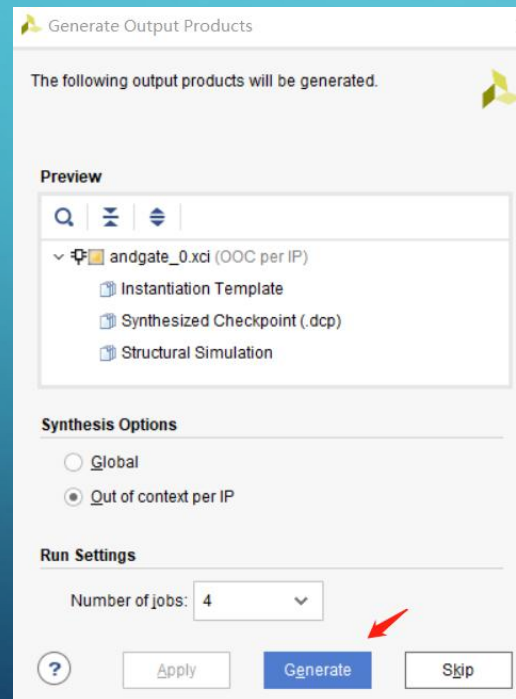
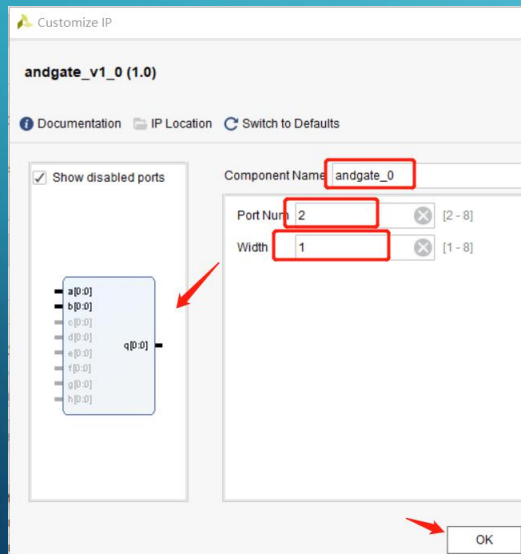
# USING IP(1) - ADDING IP INTO PROJECT(2)

- Prepare for adding IP into current project :
  - 2.find ip and add it to your project
    - Flow navigator ->PROJECT MANGER ->IP Catalog , find the IP and double click



# USING IP(1)- ADDING IP INTO PROJECT(3)

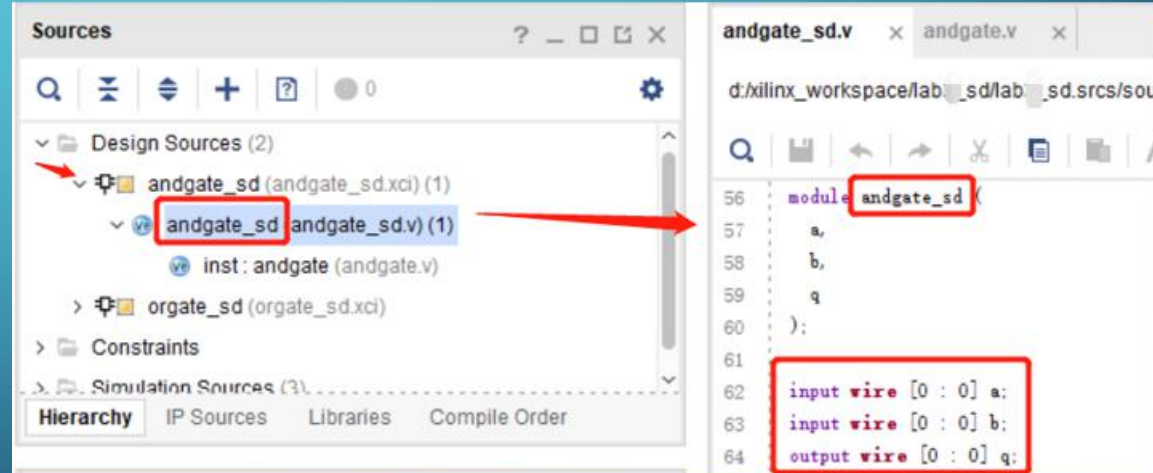
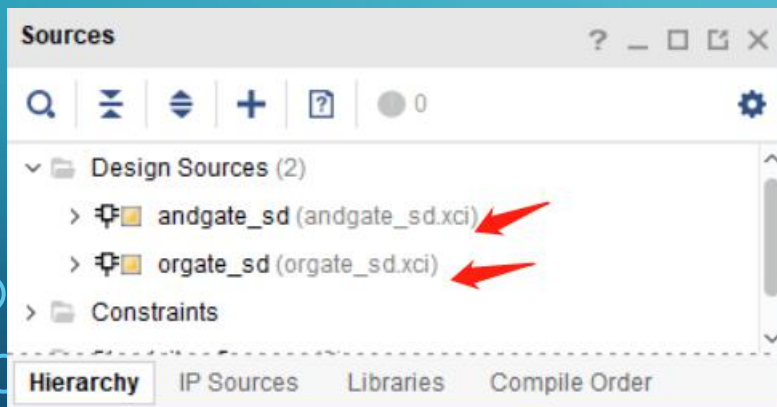
- Prepare for adding IP into current project :
  - 3. customize the IP , generate it so that it will be add into the source of the project ,which means the IP can be used in the project.



## USING IP(2)

## STRUCTURED DESIGN BASED ON IP(1)

- If you have added the IPs into the Design Sources of the project, you can find them, remember the module's name and ports' name:

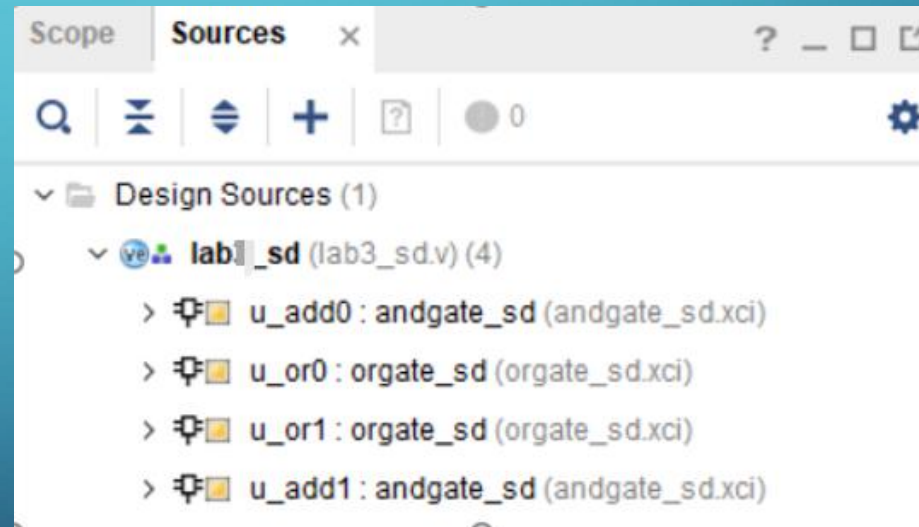


# USING IP(2)

## STRUCTURED DESIGN BASED ON IP(2)

1. Create a Verilog design file (Add sources -> choose Design source type)
2. Edit : instance IP , build module with IP instance.

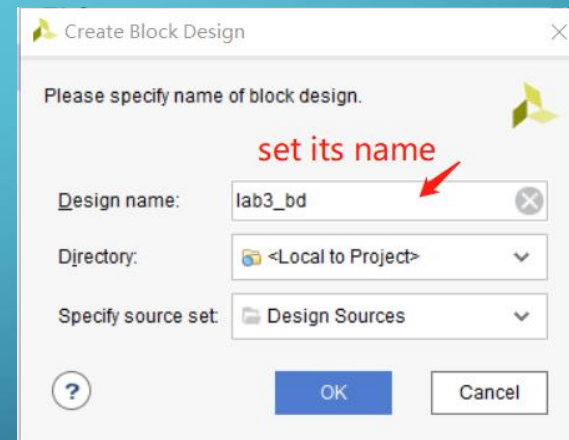
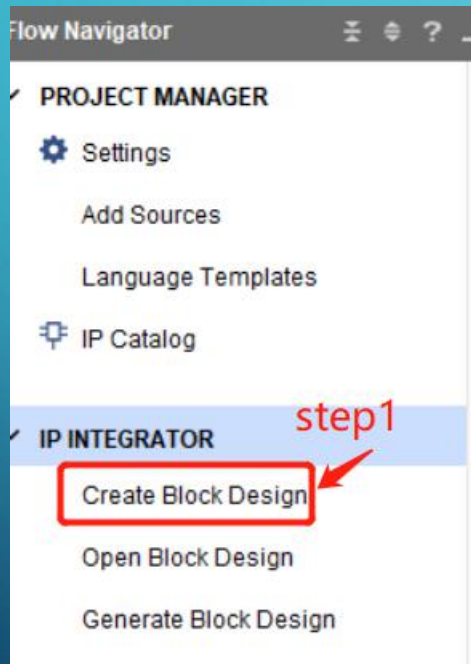
```
module lab3_sd(  
    input x,  
    input y,  
    output q1,  
    output q2,  
    output q3  
);  
  
// q1 = x  
assign q1 = x;  
  
// q2 = x | (x & y)  
wire temp0;  
andgate_sd u_add0( .a(x), .b(y), .q(temp0) );  
orgate_sd u_or0( .a(temp0), .b(x), .q(q2) );  
  
// q3 = x & (x | y)  
wire temp1;  
orgate_sd u_or1( .a(x), .b(y), .q(temp1) );  
andgate_sd u_add1( .a(temp1), .b(x), .q(q3) );  
endmodule
```



# USING IP(3) - BLOCK DESIGN(1)

step 1: create a block design , then set its name

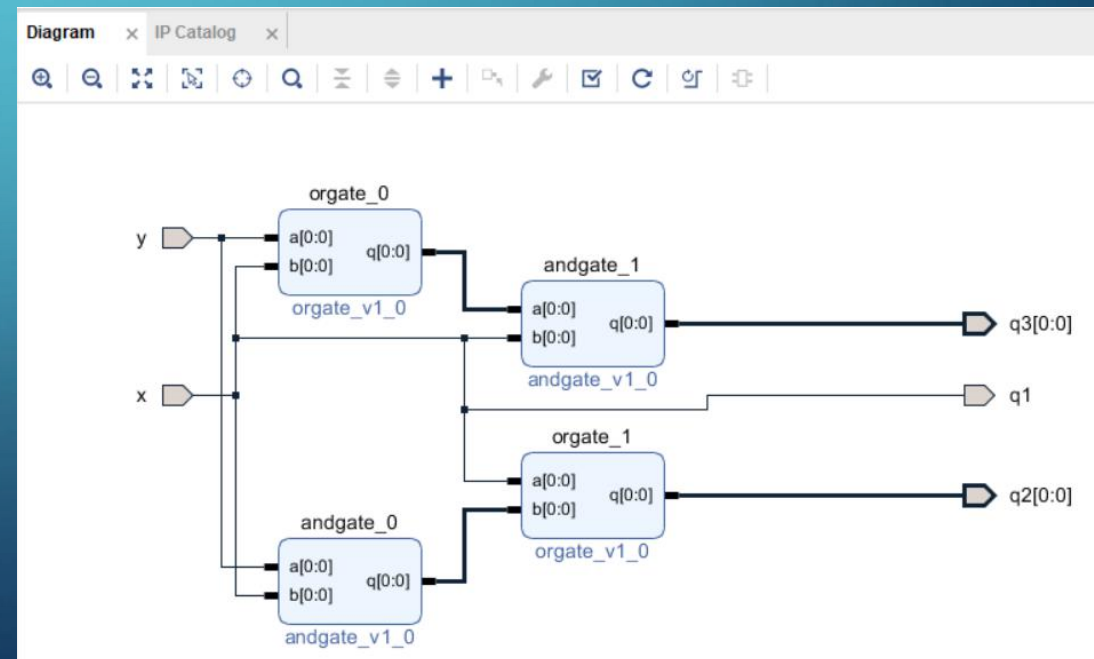
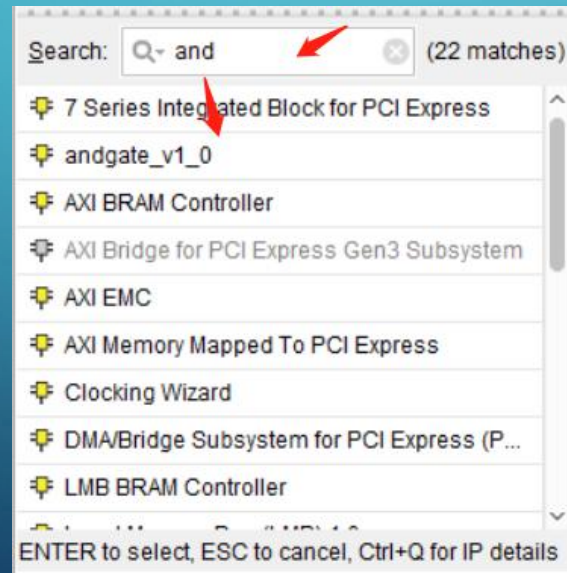
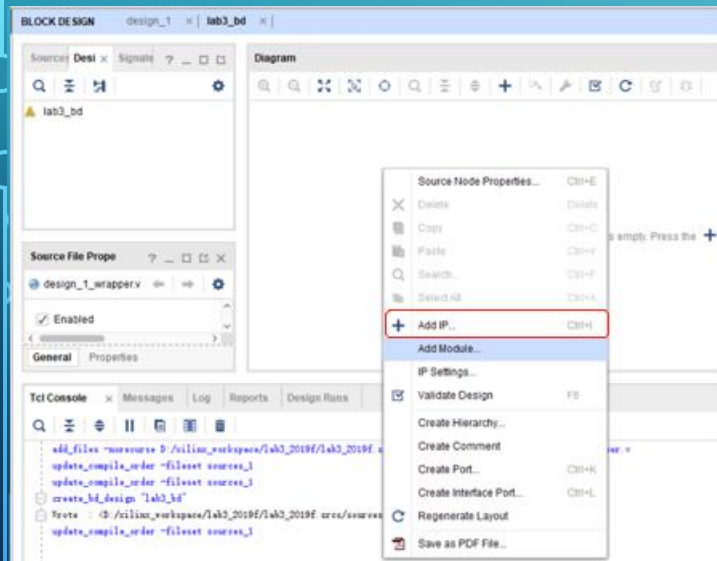
**NOTIC:** BLOCK DESIGN could be based on Module or IP core or both.





# USING IP(3) - BLOCK DESIGN(2)

1. create a block design, set its name. //same with step 1 of block design on module.
2. Instance the IP, place it : Find the IP , re-customize it if needed . Then place it in the Diagram.
3. Instance the port, place it:  
Right click blank place ,choose create port (to set the name , direction and width of the port)
4. Connect the ports and IP instances
5. Check if it is ok or not





# PRACTICES

- 1. Design a NAND gate, and package it to an IP Core named as SUSTC\_CSE\_nandgate.

Requirments: the maximum port is 8;

the minimum port is 2;

the width of the ports should range from 1 to 8.

- 2. Use the SUSTC\_CSE\_nandgate, SUSTC\_CSE\_norgate, SUSTC\_CSE\_notgate and SUSTC\_CSE\_andgate to verify the following equation. **it should be realized both by structured design and block design.**

$$(x+y)'(xy)' = x'y'$$

create testbench, do simulation to verify function of the design, generate bitstream file, test on the board.

# TIPS : BLOCK DESIGN BASED ON MODULE

Build block design based on module is almost same as using IP core, such as create block design, creat ports, connect ports with module. The only differece is 'add module'.  
NOTIC : **right click** blank in the 'Diagram' window, choose '**Add Module**' to add a module into the block

