Buidling Basic Elements for IPI

Introduction

This lab guides you through creating basic elements which can be used in basic digital design course. The steps can be used to extend it to creating IP of any complexity.

Objectives

After completing this lab, you will be able to:

- Use Create and Package IP feature of Vivado to create IP
- Simulate and verify IP functionality
- Generate the bitstream and verify the functionality in hardware

Procedure

This lab is separated into steps that consist of general overview statements that provide information on the detailed instructions that follow. Follow these detailed instructions to progress through the lab.

This lab comprises 4 primary steps: You will create two custom IPs in Vivado, create another project to use the created IPs, simulate the design, and verify the functionality in the hardware.

Create a Project for Creating IP in Vivado Step 1

1-1. Create a Vivado project calling it as xup_and2 in the c:\xup\digital directory using the provided xup_and2.v source file targeting the default V7 family device*.* Synthesize the design.

1-1-1. Open Vivado by selecting **Start** > **All Programs** > **Xilinx Design Tools** > **Vivado 2014.2** > **Vivado 2014.2**

1-1-2. Click the *Create New Project* link.

1-1-3. Click **Next**, and name the project **xup_and2** in the *c:\xup\digital* directory.

1-1-4. Click **Next** and make sure that *RTL Project* is selected.

1-1-5. Click **Next** and make sure that **Verilog** is selected as the *Target language* and *Simulation language*. Click *Add Files*, browse to *c:\xup\digital\sources* and select **xup_and2.v**, and click **OK**.

1-1-6. Click **Next** two times until *Add Constraints* form is displayed.

1-1-7. Remove any constraint files listed, if any, and click **Next** to see the *Default Part* form.

1-1-8. Click **Next** with the default V7 family part selected and then **Finish**.

1-1-9. Click *Run Synthesis* in the *Flow Navigator* pane. (This step is optional but recommended to make sure that the design is synthesizable).

1-1-10. Click **Cancel** when the process is completed and the dialog box is presented.

Make sure no errors are reported.

1-1-11. In the *Design Runs* tab, right click on the *synth_1* and select **Reset Runs**.

1-1-12. Make sure that the *Delete the generated files in the working directory* option is selected and click on **Reset**.

This makes sure that you are not providing the synthesis results.

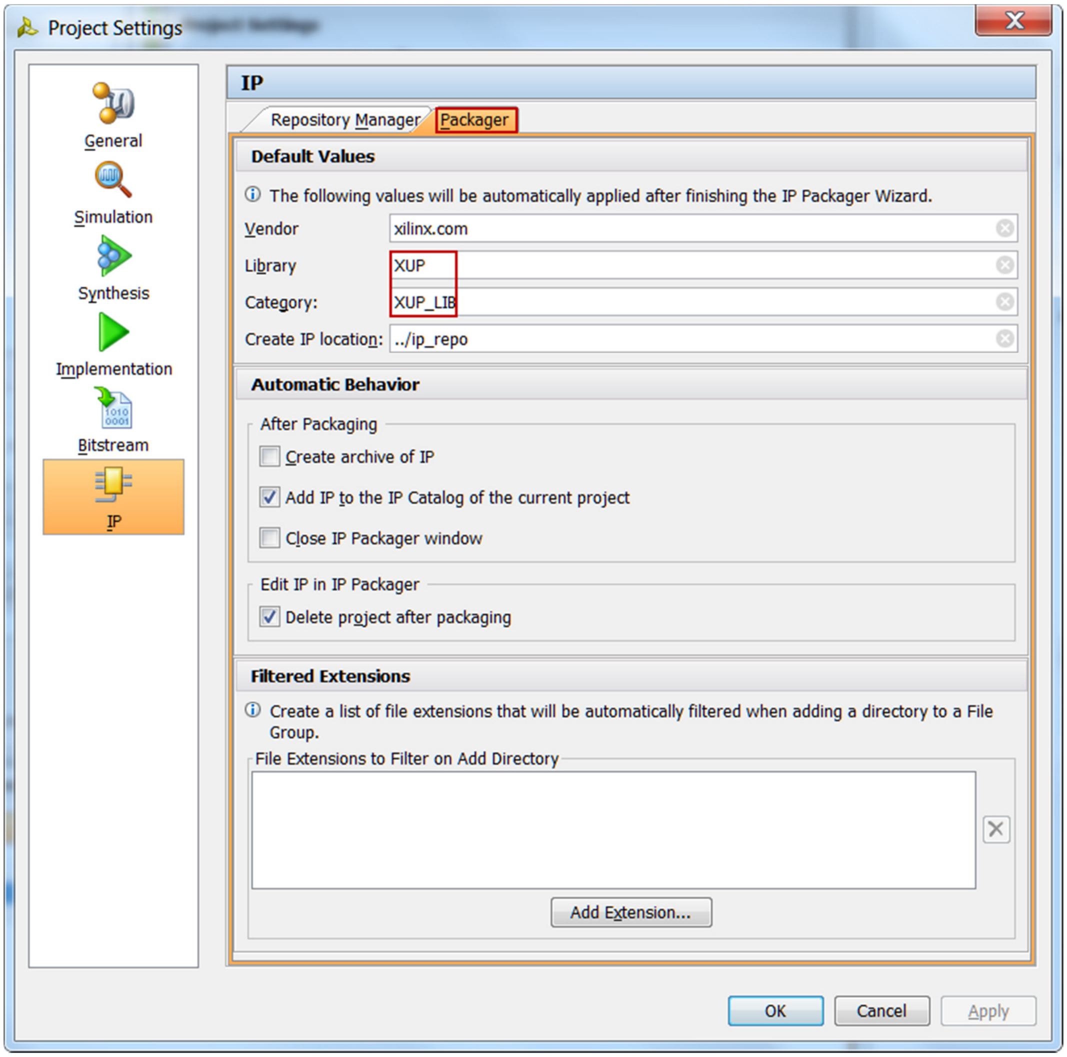
1-2. Set the library name and category if desired. Here you will use XUP as the library name and XUP_LIB as the category. You can change Vendor name if necessary.

1-2-1. Click *Project Settings* in the *Flow Navigator* pane.

1-2-2. Select *IP* in the left pane.

1-2-3. Click on the *Packager* tab.

1-2-4. Change the necessary fields as shown below and click **OK**.



Project Settings

IP

Repository Manager | Packager

Default Values

ⓘ The following values will be automatically applied after finishing the IP Packager Wizard.

Vendor: xilinx.com

Library: XUP

Category: XUP_LIB

Create IP location: ../ip_repo

Automatic Behavior

After Packaging

□ Create archive of IP

☑ Add IP to the IP Catalog of the current project

□ Close IP Packager window

Edit IP in IP Packager

☑ Delete project after packaging

Filtered Extensions

ⓘ Create a list of file extensions that will be automatically filtered when adding a directory to a File Group.

File Extensions to Filter on Add Directory

Figure 1. Setting up the Library and Category fields.

1-3. Package the IP.

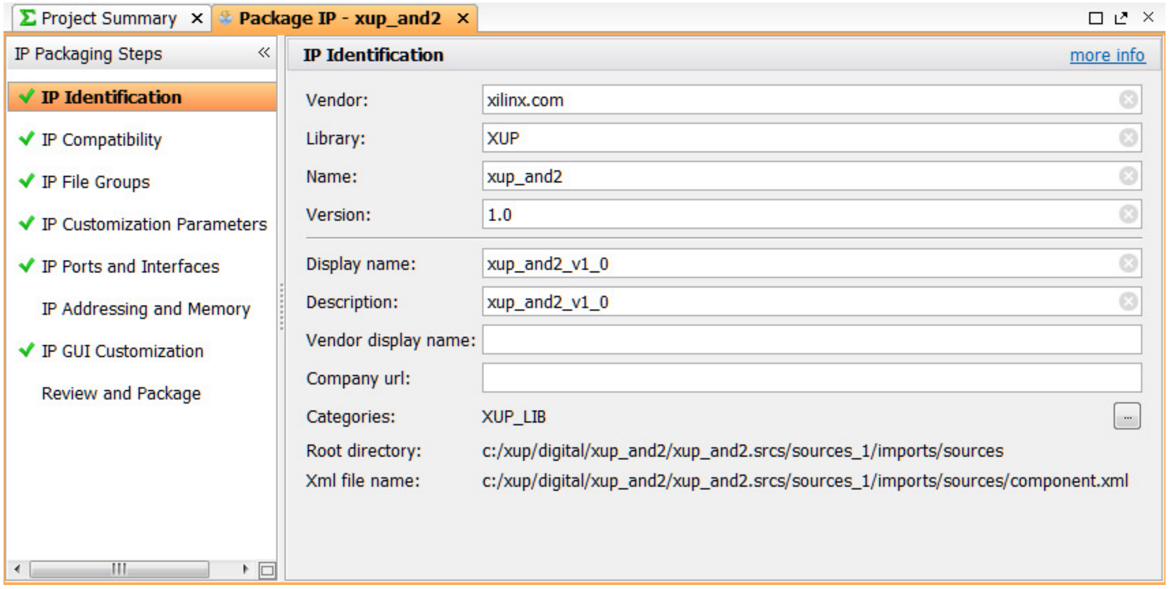
1-3-1. Select **Tools** > **Create and IP Package…**

1-3-2. Click **Next**.

1-3-3. With the *Package your project* option selected, click **Next** twice**,** and then click **Finish**.

The summary form will be displayed showing various components and files used in creating the IP as it stands at the moment. We will customize some of the components. Click **OK**.

The **Package IP – xup_and2** tab will be opened showing the default values and the available options.

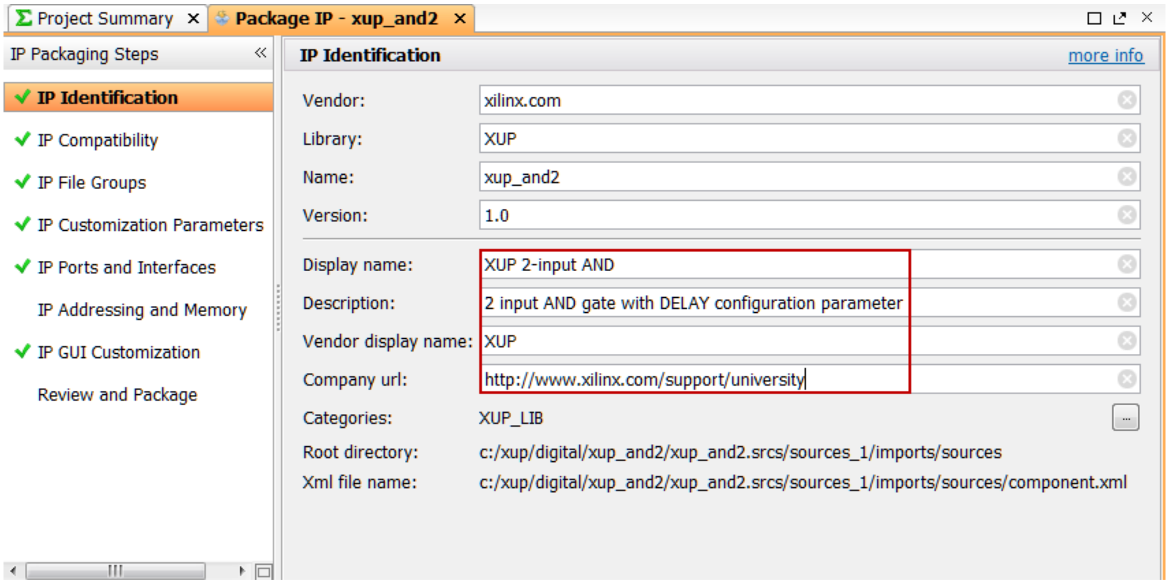


IP Identification

Vendor: xilinx.com
Library: XUP
Name: xup_and2
Version: 1.0
Display name: xup_and2_v1_0
Description: xup_and2_v1_0
Vendor display name:
Company url:
Categories: XUP_LIB
Root directory: c:/xup/digital/xup_and2/xup_and2.srcs/sources_1/imports/sources
Xml file name: c:/xup/digital/xup_and2/xup_and2.srcs/sources_1/imports/sources/component.xml

Figure 2. The IP Identification default values

1-3-4. Make necessary changes to the IP *Identification* fields as shown.



IP Identification

Vendor: xilinx.com
Library: XUP
Name: xup_and2
Version: 1.0
Display name: XUP 2-input AND
Description: 2 input AND gate with DELAY configuration parameter
Vendor display name: XUP
Company url: http://www.xilinx.com/support/university
Categories: XUP_LIB
Root directory: c:/xup/digital/xup_and2/xup_and2.srcs/sources_1/imports/sources
Xml file name: c:/xup/digital/xup_and2/xup_and2.srcs/sources_1/imports/sources/component.xml

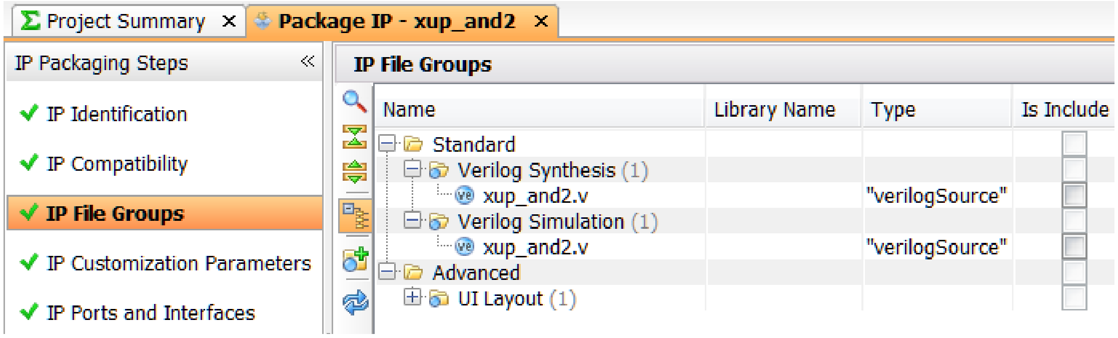
Figure 3. The IP Identification customized fields

1-3-5. Select **IP Compatibility**. This shows the different Xilinx FPGA Families that the IP supports. The value is inherited from the device selected for the project.

1-3-6. Right click in the *Family Support table* and select **Add Family…** from the menu.

1-3-7. Select the **zynq** and **artix7** families as we will be using this IP on the boards with these devices, and click **OK**.

1-3-8. Click on **IP File Groups** and expand the sub-folders to see its content. You can add additional files, like testbench, but we won't do that here.



IP Packaging Steps

- ✓ IP Identification
- ✓ IP Compatibility
- ✓ **IP File Groups**
- ✓ IP Customization Parameters
- ✓ IP Ports and Interfaces

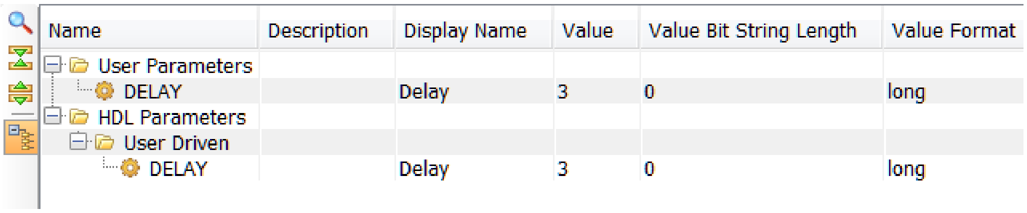
IP File Groups

Name	Library Name	Type	Is Include
Standard			□
Verilog Synthesis (1)			□
xup_and2.v		"verilogSource"	□
Verilog Simulation (1)			□
xup_and2.v		"verilogSource"	□
Advanced			□
UI Layout (1)			□

Figure 4. Updating IP File Groups

1-4. Edit IP Customization Parameter with the desired default value and type of values allowed

1-4-1. Click on the *IP Customization Parameters* and verify that delay parameter is included.

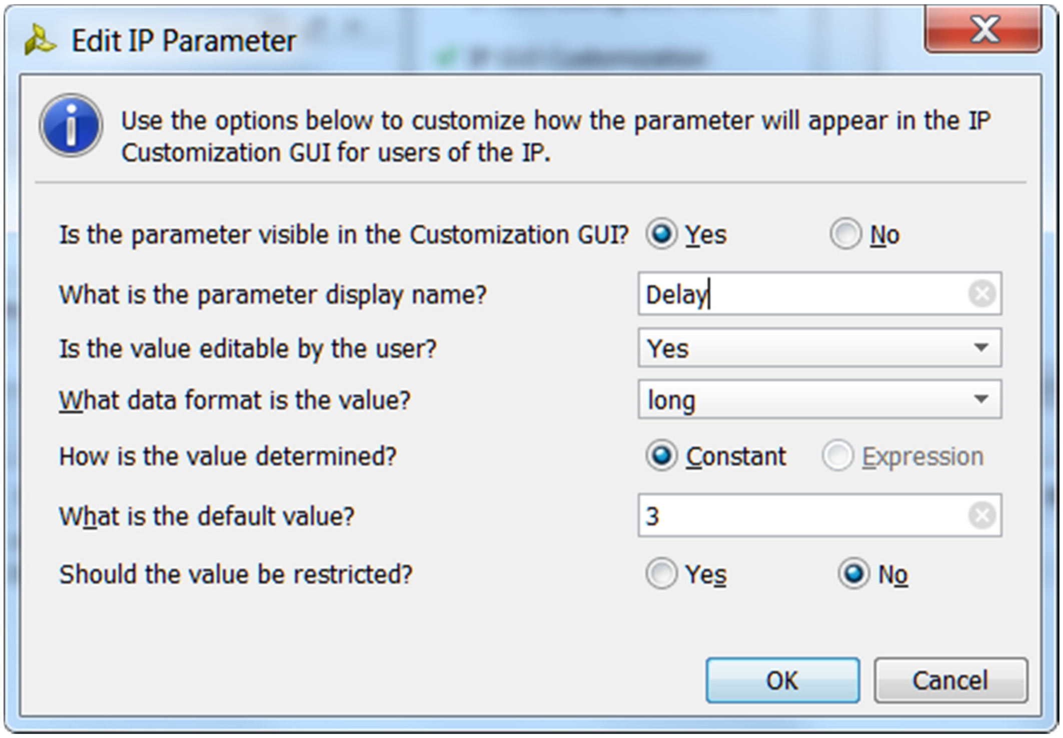


Name	Description	Display Name	Value	Value Bit String Length	Value Format
User Parameters					
DELAY		Delay	3	0	long
HDL Parameters					
User Driven					
DELAY		Delay	3	0	long

Figure 5. The Configuration Parameters

1-4-2. Right-click on the *DELAY* entry under the *User Parameters*, and select **Edit Parameters…**

The form will be displayed



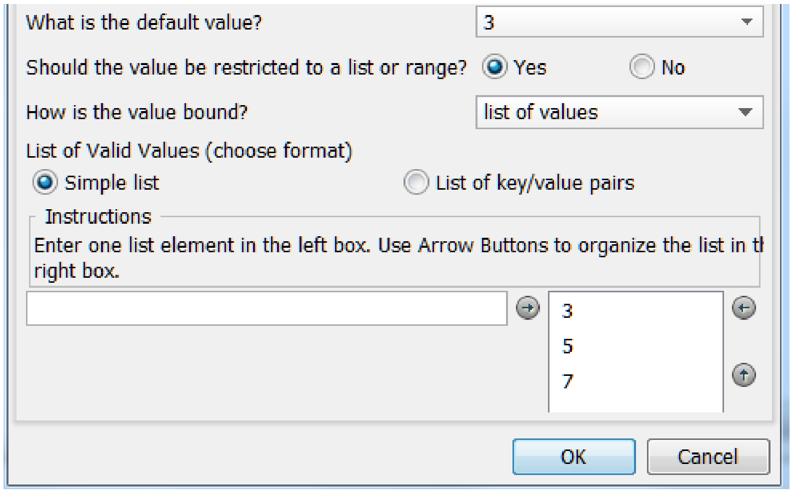
Edit IP Parameter

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

Is the parameter visible in the Customization GUI?	● Yes ○ No
What is the parameter display name?	Delay
Is the value editable by the user?	Yes
What data format is the value?	long
How is the value determined?	● Constant ○ Expression
What is the default value?	3
Should the value be restricted?	○ Yes ● No

Figure 6. Edit Parameter form

You can make the parameter fixed by selecting **No** in the *Is the value editable by the User* field. The data format can be long, float, bitstring, and string. You can change the default value (which is picked up from the model) to a different value. The value can be restricted to a list of values (with check-boxes) or even a range. Here is an example of the list of values option:



What is the default value? 3

Should the value be restricted to a list or range? ● Yes ○ No

How is the value bound? list of values

List of Valid Values (choose format)

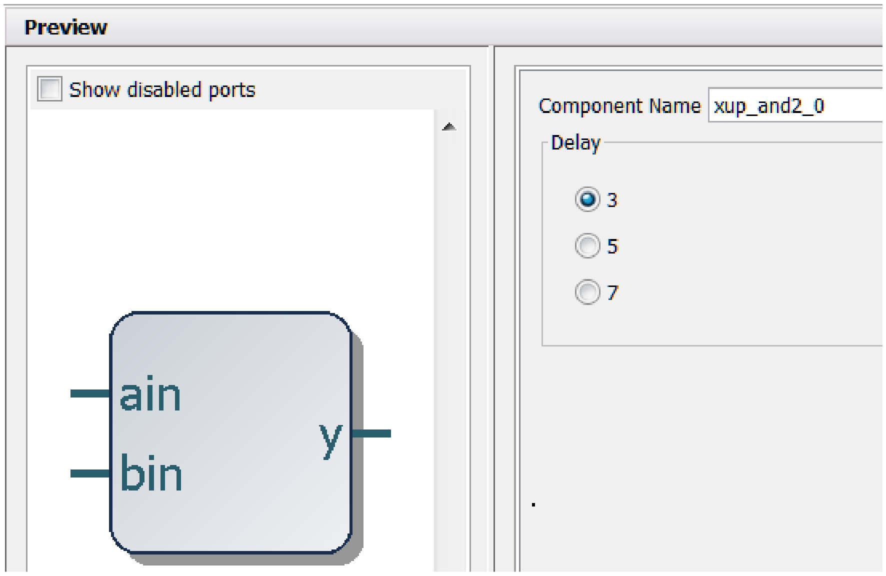
● Simple list ○ List of key/value pairs

Instructions

Enter one list element in the left box. Use Arrow Buttons to organize the list in th right box.

3
5
7

which will result in following GUI when a user tries to configure in a design



Preview

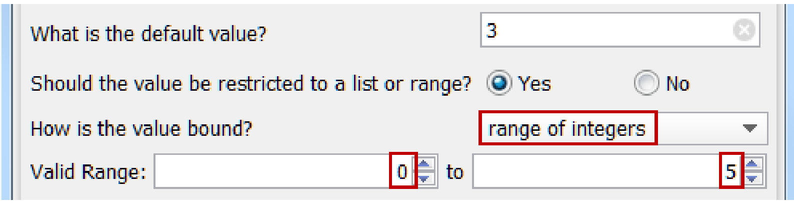
□ Show disabled ports

Component Name xup_and2_0

Delay

● 3
○ 5
○ 7

Here is an example of the range



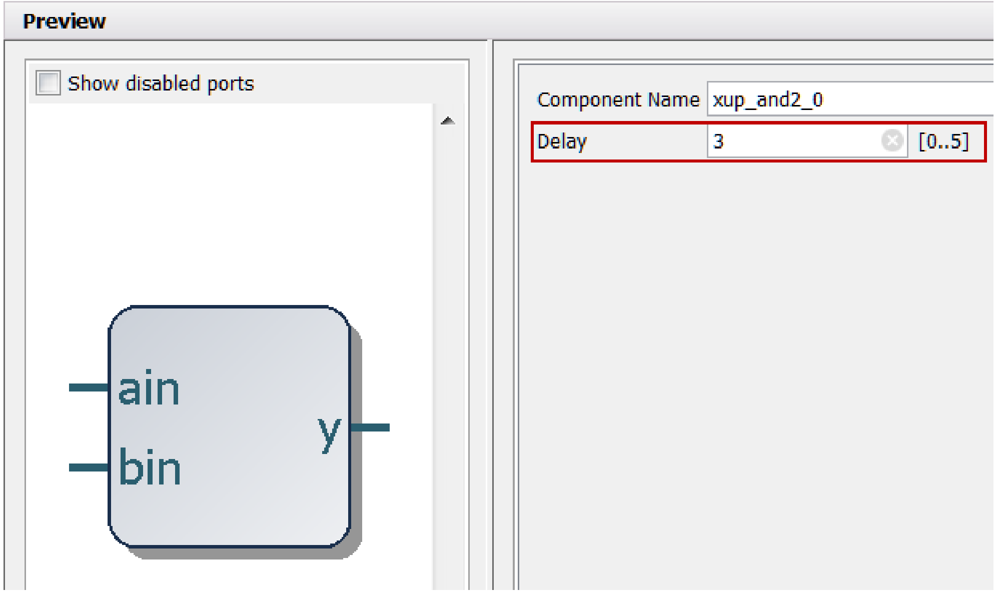
What is the default value? 3

Should the value be restricted to a list or range? ● Yes ○ No

How is the value bound? range of integers

Valid Range: 0 to 5

which will show up in the design as



Component Name: xup_and2_0

Delay: 3 [0..5]

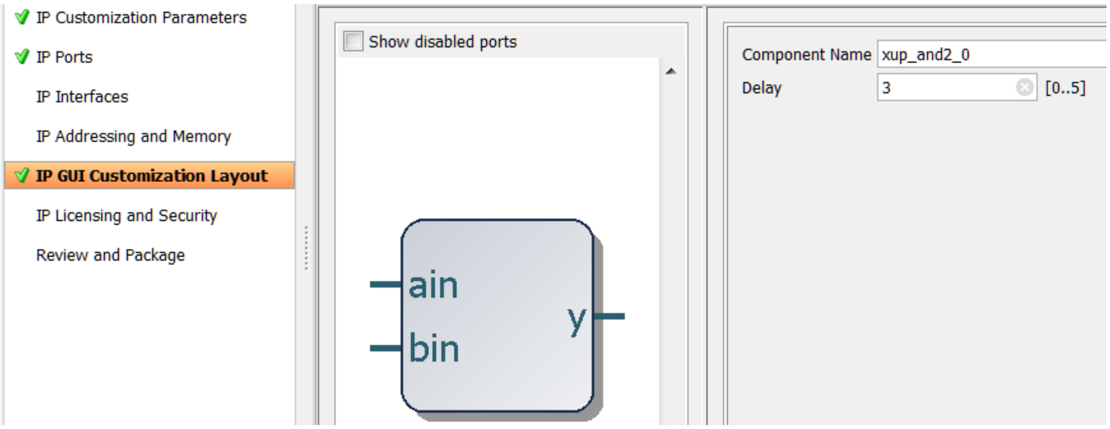
Figure 7. Various ways of configuring parameters

1-4-3. Set the parameters editing to *range of integers* and set the range as **0** to **5**. Click **OK**.

1-4-4. Click on the *IP Ports* and verify that the top-level ports are included.

1-4-5. Click on the *IP GUI Customization Layout* and then click on the *Refresh* button to see the GUI.

Notice that the input/output ports as well as the parameter with the default value are displayed.



Component Name: xup_and2_0

Delay: 3 [0..5]

Figure 8. The IP GUI Customization

1-4-6. Select *Review and Package*, and click on the **Package IP** button.

1-4-7. Click **OK**.

1-4-8. In the Vivado window click **File > Close Project.**

1-5. Create a Vivado project calling it as xup_and_vector in the c:\xup\digital directory using the provided xup_and_vector.v source file targeting the default V7 family device*.*

1-5-1. Open Vivado if it was closed.

1-5-2. Click the *Create New Project* link.

1-5-3. Click **Next**, and name the project **xup_and_vector** in the *c:\xup\digital* directory.

1-5-4. Click **Next** and make sure that *RTL Project* is selected.

1-5-5. Click **Next** and make sure that **Verilog** is selected as the *Target language* and *Simulation language*. Click *Add Files*, browse to *c:\xup\digital\source* and select **xup_and_vector.v**, and click **OK**.

1-5-6. Click **Next** two times until *Add Constraints* form is displayed.

1-5-7. Remove any constraint files listed, if any, and click **Next** to see the *Default Part* form.

1-5-8. Click **Next** with the default V7 family part selected and then **Finish**.

1-6. Set the library name and category if desired. Here you will use XUP as the library name and XUP_LIB as the category. You can change Vendor name if necessary.

1-6-1. Click *Project Settings* in the *Flow Navigator* pane.

1-6-2. Select *IP* in the left pane.

1-6-3. Click on the *Packager* tab.

1-6-4. Change the necessary fields and click **OK**.

1-7. Package the IP.

1-7-1. Select **Tools > Create and IP Package**.

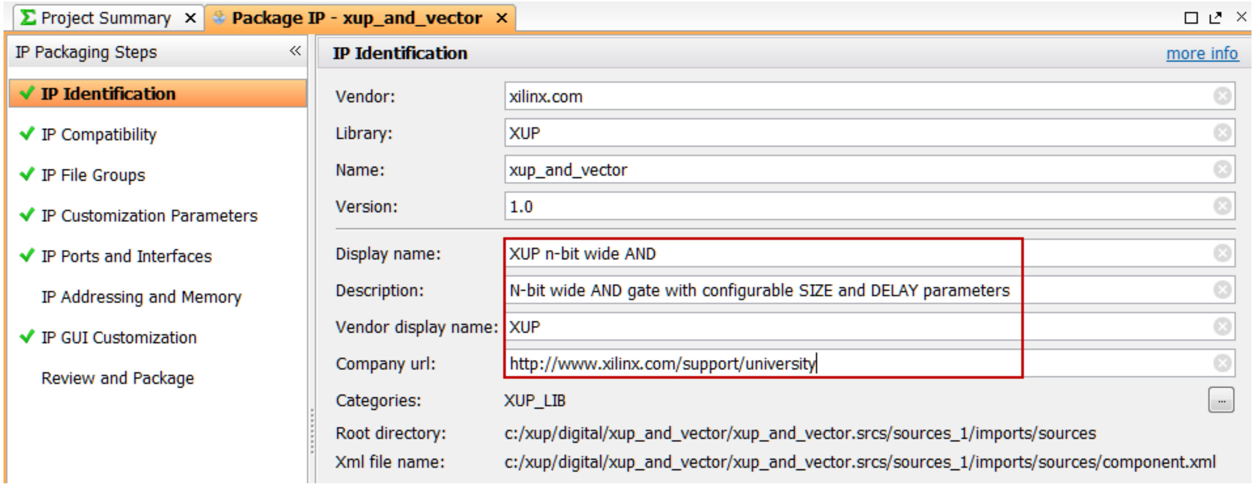
1-7-2. Click **Next**.

1-7-3. With the *Package your project* option selected, click **Next,** and then click **Finish**.

The summary form will be displayed showing various components and files used in creating the IP as it stands at the moment. We will customize some of the components. Click **OK**.

The **Package IP – xup_and_vector** tab will be opened showing the default values and the available options.

1-7-4. Make necessary changes to the IP *Identification* fields as shown.



Vendor: xilinx.com
Library: XUP
Name: xup_and_vector
Version: 1.0
Display name: XUP n-bit wide AND
Description: N-bit wide AND gate with configurable SIZE and DELAY parameters
Vendor display name: XUP
Company url: http://www.xilinx.com/support/university
Categories: XUP_LIB
Root directory: c:/xup/digital/xup_and_vector/xup_and_vector.srcs/sources_1/imports/sources
Xml file name: c:/xup/digital/xup_and_vector/xup_and_vector.srcs/sources_1/imports/sources/component.xml

Figure 9. The IP Identification customized fields

1-7-5. Select **IP Compatibility**. This shows the different Xilinx FPGA Families that the IP supports. The value is inherited from the device selected for the project.

1-7-6. Right click in the *Family Support table* and select **Add Family…** from the menu.

1-7-7. Select the **zynq** and **artix7** families as we will be using this IP on the boards with these devices, and click **OK**.

1-7-8. Click on **IP File Groups** and expand the sub-folders to see its content. You can add additional files, like testbench, but we won't do that here.

1-8. Edit IP Customization Parameter with the desired default value and type of values allowed

1-8-1. Click on the *IP Customization Parameters* and verify that DELAY and SIZE parameters are included.

1-8-2. Right-click on the *DELAY* entry under the *User Parameters*, and select **Edit Parameters…**

The form will be displayed

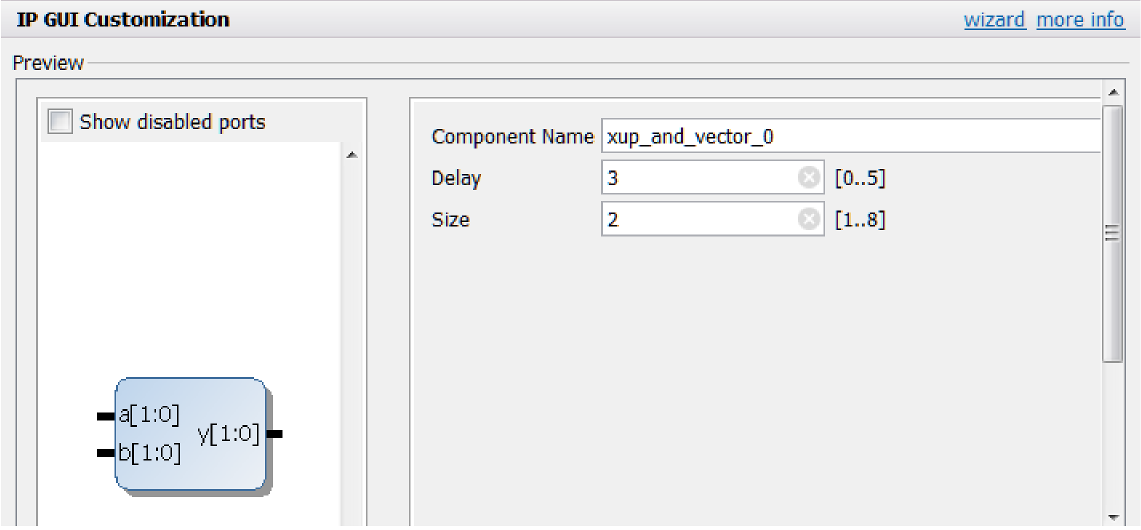
1-8-3. Set the parameters editing to *range of integers* and set the range as **0** to **5**. Click **OK**.

1-8-4. Similarly, set the SIZE entry parameters to *range of integers* and set the range as **1** to **8**. Click **OK**

1-8-5. Click on the *IP Ports* and verify that the top-level ports are included.

1-8-6. Click on the *IP GUI Customization Layout* and then click on the *Refresh* button to see the GUI.

Notice that the input/output ports as well as the parameter with the default value are displayed.



IP GUI Customization

Component Name: xup_and_vector_0

Delay: 3 [0..5]

Size: 2 [1..8]

Figure 10. The IP GUI Customization

1-8-7. Select *Review and Package*, and click on **Package IP.**

1-8-8. In the Vivado window click **File** > **Close Project.**

1-8-9. Click **OK**.

1-8-10. Using the Windows Explorer, copy the generated xup_and2 and xup_and_vector folders into the xup_lib folder under c:\xup\digital (create the folder if does not exist).

Create a Project for Testing the Created IPs Step 2

2-1. Create an empty Vivado project calling it as xup_and_test in the c:\xup\digital directory targeting the xc7a100tcsg324-1 device (for Nexys4) or xc7a35tcpg236-1 device (for Basys3). Setup the IP Repository to point to c:\xup\digital\xup_lib directory. Add the xup_and2 to the IP catalog.

2-1-1. Click the *Create New Project* link.

2-1-2. Set the directory path to *c:\xup\digital\* and the project name as **xup_and_test**.

2-1-3. Click **Next** and make sure that the *RTL Project* type is selected.

2-1-4. Click **Next** and make sure that **Verilog** is selected as the *Target language* and *Simulation language*.

2-1-5. Click **Next** three times until *Default Part* form is displayed*.*

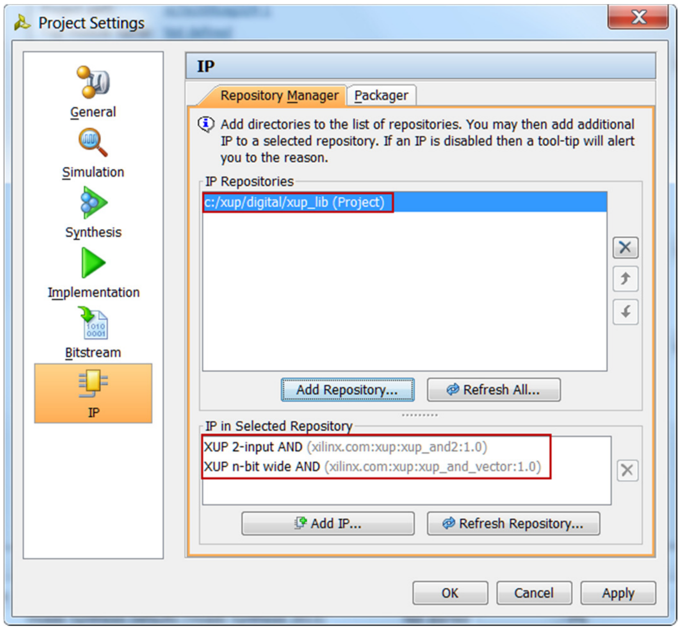
2-1-6. Using the appropriate filters, select *xc7a100tcsg324-1* part (for Nexys4) or *xc7a35tcpg236-1* part (for Basys3), then click **Next**, and then **Finish**.

2-1-7. Click **Project Settings** in the *Flow Navigator* pane.

2-1-8. Select **IP** in the left pane of the *Project Settings* form.

2-1-9. Click on the **Add Repository…** button, browse to **c:\xup\digita\xup_lib** and click **Select**.

The directory will be scanned and IP entries will appear in the **Selected Repository** window.



Add directories to the list of repositories. You may then add additional IP to a selected repository. If an IP is disabled then a tool-tip will alert you to the reason.

c:/xup/digital/xup_lib (Project)

XUP 2-input AND (xilinx.com:xup:xup_and2:1.0)
XUP n-bit wide AND (xilinx.com:xup:xup_and_vector:1.0)

Figure 11. Specify IP Repository

2-1-10. Click **OK**.

2-2. Create the block design, called system, instantiating the xup_and2 and setting the delay parameter to 5. Make the ports external.

2-2-1. Click on the **Create Block Design** in the *Flow Navigator* pane.

2-2-2. Set the design name to **system** and click **OK**.

2-2-3. IP from the catalog can be added in different ways. Click on *Add IP* in the message at the top of the *Diagram* panel.

2-2-4. Once the IP Catalog is open, type “XUP” into the Search bar, find and double click on **XUP 2-input AND** entry, or click on the entry and hit the *Enter* key to add it to the design.

2-2-5. Double-click on the *xup_and2_0* instance to open the configuration form.

2-2-6. Change the *Delay* value to **5** and click **OK**.

2-2-7. Right-click on the *y* port and select **Make External**.

2-2-8. Similarly, select the *ain* and *bin* ports and make them external.

2-2-9. The block diagram should look like below.

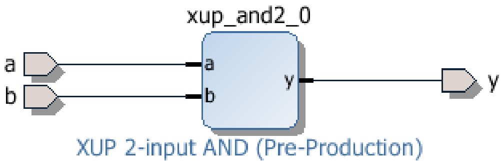


Figure 12. The block design

2-3. Instantiate the xup_and_vector. Set the Delay parameter to 2 and the Size to 3. Make the ports external.

2-3-1. Similarly, add an instance of an **XUP n-bit wide AND** to the design.

2-3-2. Double-click on the *xup_and_vector_0* instance to open the configuration form.

2-3-3. Change the *Delay* value to **2** and Size to 3, and click **OK**.

2-3-4. Right-click on the *y* port and select **Make External**.

2-3-5. Similarly, select the *ain* and *bin* ports and make them external.

2-3-6. The block diagram should look like below.

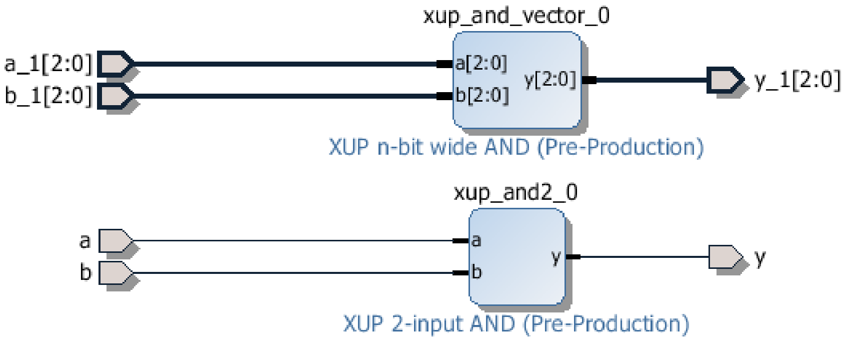


Figure 13. The block design

2-4. Validate the design. Generate the output products and create HDL wrapper.

2-4-1. Click **Tools > Validate design** and correct any errors if necessary

2-4-2. Right Click on *system.bd* in the **Sources** tab and select **Generate Output Products**.

This will generate the source files of the IP(s) in the design.

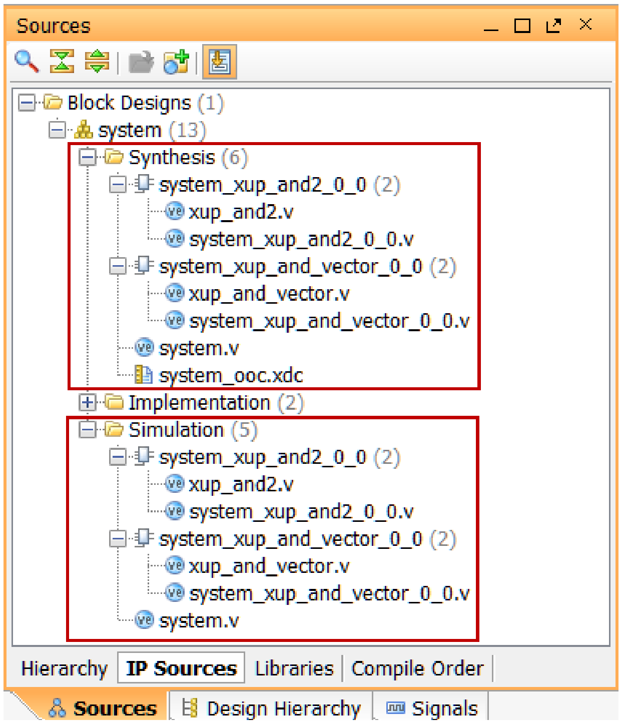
2-4-3. Right Click on *system.bd* in the **Sources** tab and select **Create HDL Wrapper**.

2-4-4. Select **Let Vivado Manage wrapper and auto-update** option and click **OK** when prompted.

2-5. Look at the source file (model of the IP).

2-5-1. Select **IP Sources** tab and expand the hierarchy.

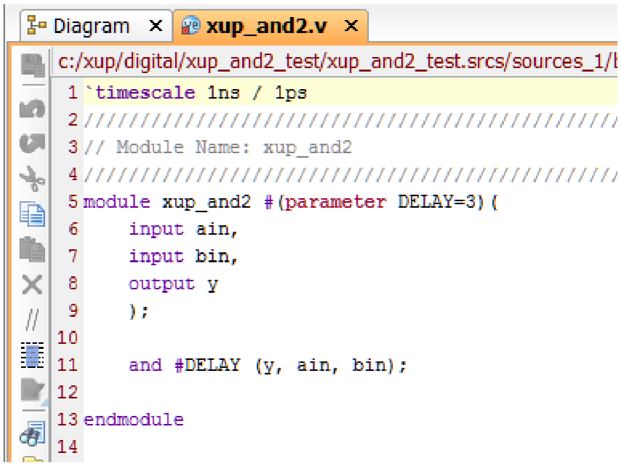
Observe the xup_and2.v entry under the synthesis and simulation categories. In this simple example, it is the same. In complex design, they may be different depending on how the IP was created.



```
Block Designs (1)
  system (13)
    Synthesis (6)
      system_xup_and2_0_0 (2)
        xup_and2.v
        system_xup_and2_0_0.v
      system_xup_and_vector_0_0 (2)
        xup_and_vector.v
        system_xup_and_vector_0_0.v
      system.v
      system_ooc.xdc
    Implementation (2)
    Simulation (5)
      system_xup_and2_0_0 (2)
        xup_and2.v
        system_xup_and2_0_0.v
      system_xup_and_vector_0_0 (2)
        xup_and_vector.v
        system_xup_and_vector_0_0.v
      system.v
```

Figure 14. The IP hierarchy

2-5-2. Double-click the *xup_and2.v* entry and look at the model used to create the IP and the functionality it is providing.



c:/xup/digital/xup_and2_test/xup_and2_test.srcs/sources_1/

```
`timescale 1ns / 1ps
//////////////////////////////////////////////////
// Module Name: xup_and2
//////////////////////////////////////////////////
module xup_and2 #(parameter DELAY=3)(
    input ain,
    input bin,
    output y
    );

    and #DELAY (y, ain, bin);

endmodule

```

Figure 15. The IP model

Simulate the Design Step 3

3-1. Set simulation time to 100 ns.

3-1-1. Select **Simulation > Simulation Settings** to open the form.

3-1-2. In the **Simulation** tab, set the simulation time to **100** ns and click **OK** to close the window.

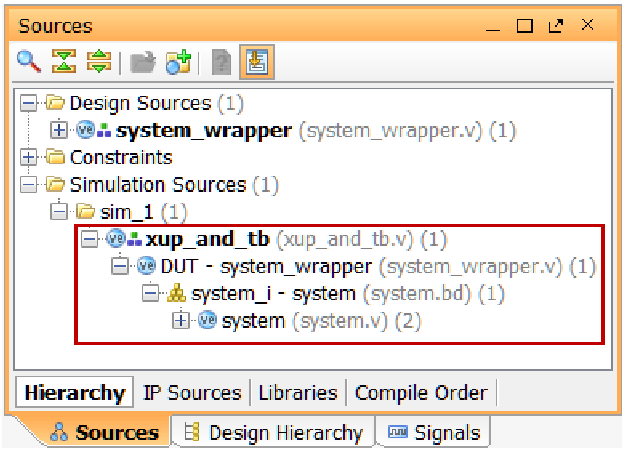
3-2. Add the provided xup_and_tb.v testbench, simulate and examine the output

3-2-1. Click *Add Sources.*

3-2-2. Select *Add or Create Simulation Sources* and click **Next.**

3-2-3. Click *Add Files*, and browse to the *c:\xup\digital* directory and select **xup_and_tb.v** and click **Finish.**

3-2-4. Expand the hierarchy and notice how the design is setup for the simulation.



```
Design Sources (1)
    system_wrapper (system_wrapper.v) (1)
Constraints
Simulation Sources (1)
    sim_1 (1)
        xup_and_tb (xup_and_tb.v) (1)
            DUT - system_wrapper (system_wrapper.v) (1)
                system_i - system (system.bd) (1)
                    system (system.v) (2)
```

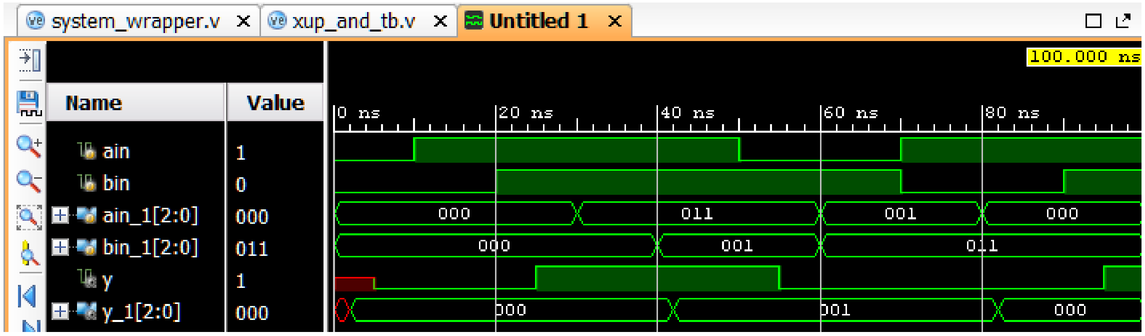
Figure 16. Hierarchy for simulation

3-3. Launch the simulator which will automatically elaborate the model source file, load the simulation model, and run the simulation.

3-3-1. In Vivado, select **Simulation > Run Simulation > Run Behavioral Simulation** to launch the simulator.

When done, the waveform window will show up.

3-3-2. Click on the zoom full button ( ) to see the entire simulation waveform. It should look similar to the one shown below.



Name	Value
ain	1
bin	0
ain_1[2:0]	000
bin_1[2:0]	011
y	1
y_1[2:0]	000

Figure 18. Full Simulation Output

3-3-3. Click at 20 ns to show a marker.

3-3-4. Add another marker by clicking ( ).

A blue ribbon will appear.

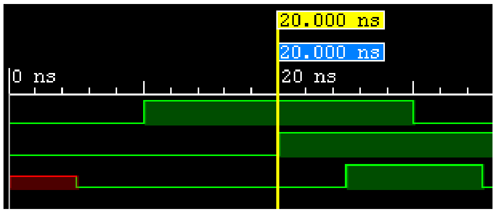


Figure 19. Another marker added

3-3-5. Drag the blue ribbon to the where y makes transition from 0 to 1. Verify that the time difference is 5 ns (you may have to zoom in for accurate measurement).

3-3-6. Close the simulation by selecting **File > Close Simulation** without saving the changes to the waveform.

3-4. Change the delay in the block diagram to 3, create the hdl wrapper, simulate the design and verify that the delay has changed to 3.

3-4-1. Select *Open Block Design* in the *Flow Navigator* pane.

3-4-2. In the *Diagram*, tab, double-click the *xup_and2_0* instance to open the configuration form.

3-4-3. Change the *delay* value to **3** and click **OK**.

3-4-4. Right Click on *system.bd* (expand the hierarchy if necessary) in the **Sources** tab and select **Create HDL Wrapper**.

3-4-5. Select **Let Vivado Manage wrapper and auto-update** option and click OK when prompted

3-4-6. In Vivado, select **Run Simulation** > **Run Behavioral Simulation** to launch the simulator.

3-4-7. Verify that the delay has changed to 3.

Verify the Design in Hardware Step 4

4-1. Add the provided design constraint file.

4-1-1. Click *Add Sources.*

4-1-2. Select *Add or Create Constraints* and click **Next.**

4-1-3. Click *Add Files*, and browse to the *c:\xup\digital* directory and select **xup_and2.xdc** and click **Finish.**

4-2. Connect the board and power it ON. Generate the bitstream, open a hardware session, program the FPGA and verify the functionality.

4-2-1. Make sure that the power supply source is jumper to *USB* and the provided Micro-USB cable is connected between the board and the PC. Note that you do not need to connect the power jack and the board can be powered and configured via USB alone

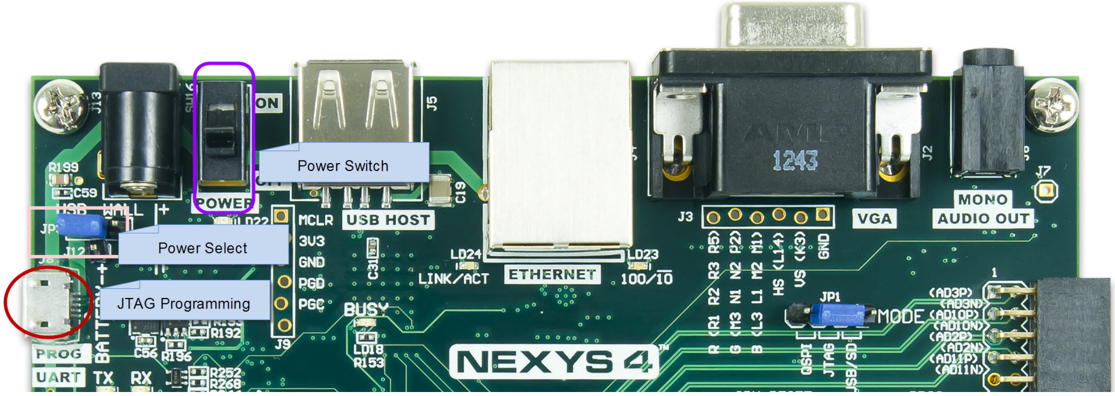


Figure 20. Board settings for Nexys4

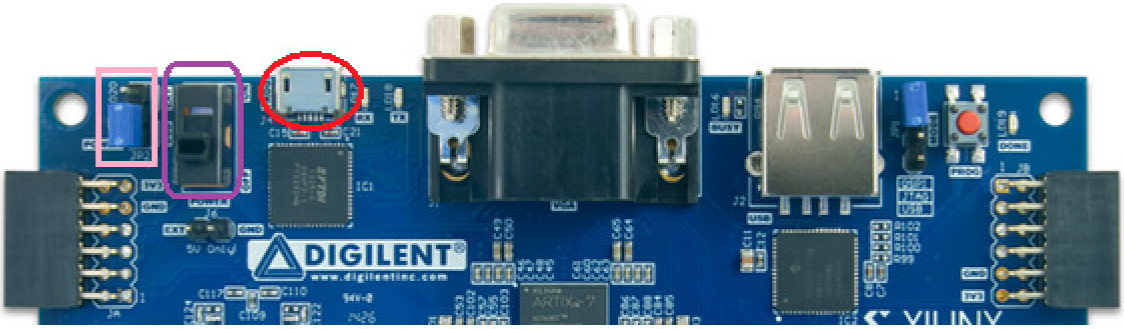


Figure 20. Board settings for Basys3

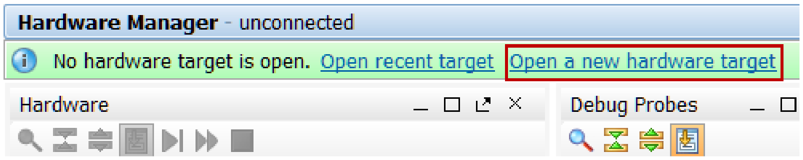
4-2-2. Power **ON** the switch on the board.

4-2-3. Click the *Generate Bitstream* in the *Flow Navigator* pane.

4-2-4. Click **Yes** to run the necessary processes and wait for the *Bitstream Generation Completed* form to appear.

4-2-5. Click on the **Open Hardware Manager** option and select **OK.**

You can also click on the Open Recent Hardware Target link if the board was already targeted before.



Hardware Manager - unconnected

No hardware target is open. Open recent target Open a new hardware target

Figure 21. Opening new hardware target

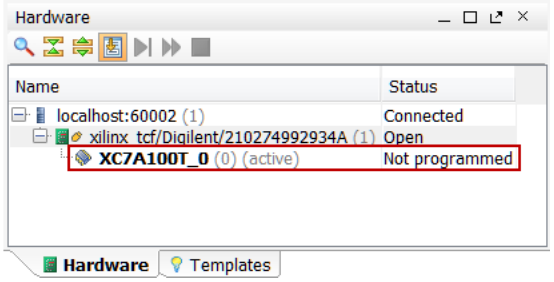
4-2-6. Click **Next** to see the Vivado CSE Server Name form.

4-2-7. Click **Next** with the localhost port selected.

The JTAG cable will be searched and the Xilinx_tcf should be detected and identified as e hardware target. It will also show the hardware devices detected in the chain.

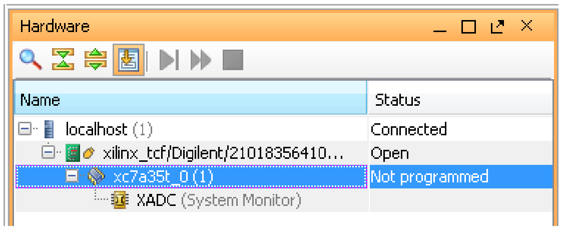
4-2-8. Click **Next** twice and **Finish**.

The Hardware Session status changes from Unconnected to the server name and the device is highlighted. Also notice that the Status indicates that it is not programmed.



Name	Status
localhost:60002 (1)	Connected
xilinx_tcf/Digilent/210274992934A (1)	Open
XC7A100T_0 (0) (active)	Not programmed

Figure 22. Opened hardware session for Nexys4



Name	Status
localhost (1)	Connected
xilinx_tcf/Digilent/21018356410...	Open
xc7a35t_0 (1)	Not programmed
XADC (System Monitor)	

Figure 22. Opened hardware session for Basys3

4-2-9. **Right-click** on the device and select *Program Device…* to program the target FPGA device. Click OK to program the board with the system_wrapper.bit file.

The DONE light will lit when the device is programmed. You may see LED[0] lit depending on the switches position.

4-2-10. Verify the functionality by flipping switches and observing the output on the LEDs.

4-2-11. Close the hardware session by selecting **File > Close Hardware Manager.**

4-2-12. Click **OK** to close the session.

4-2-13. Power **OFF** the board.

4-2-14. Close the **Vivado** program by selecting **File > Exit** and click **OK**.

Conclusion

This lab guided you through creating two custom IPs; one with scalar ports and another with vector ports, then using one of the custom IPs in the IP Integrator, simulating it, and verifying the functionality in the hardware.