Creating a custom IP in Vivado

Introduction

This tutorial shows how to package a RTL project (VHDL) to create a custom IP in Vivado 2017.2.

Create the module

Open Vivado 2017.2 and create a new project. Create a new VHDL file called logic_function.vhd.

We will first create a 1-bit Logical AND. In the "*Define Module*" define 2 1-bit inputs (A_inp and B_inp) and 1 1-bit output (P_outp).

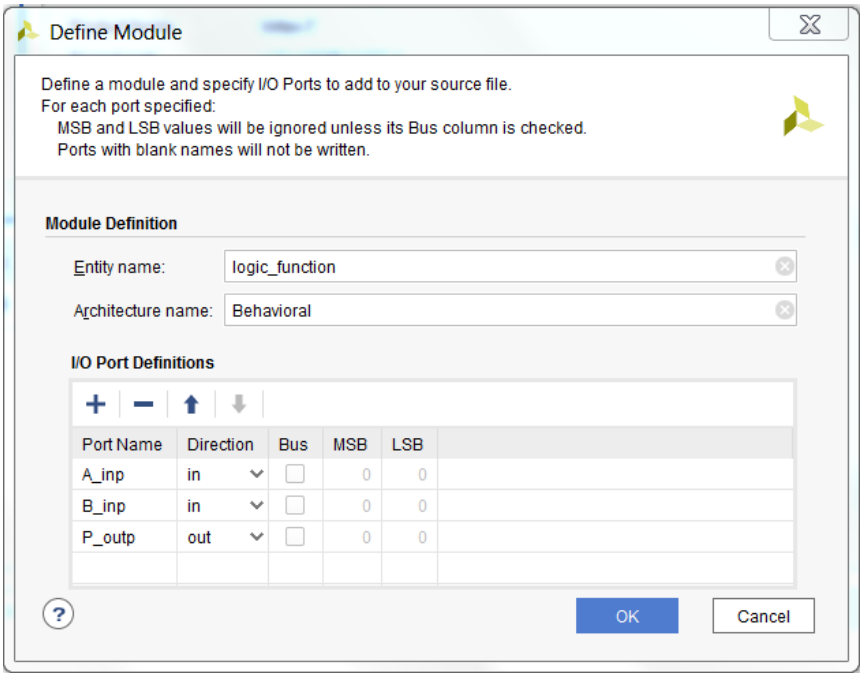


Figure 1 - Define the module

In the architecture of the module, just add the following line:

```
P_outp <= A_inp AND B_inp;
```

I. Package the module in an IP

Click on "*Tools > Create and Package New IP…*" to create a new project to package an IP. In the "Create Peripherals, Package IP or Package a Block Design" page, select "*Package your current project*".

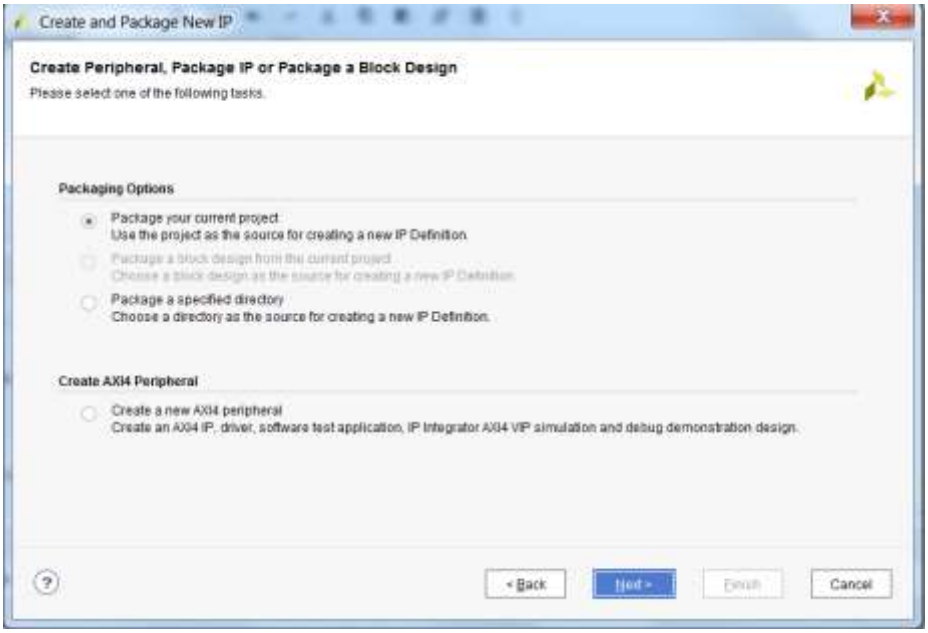


Figure 2 - Package your current project

In the "*Package your current project*" page, choose where you want the IP to be created and select "*Include .xci files*". If you are using revision control with your custom IP definition, the UG1118 recommends to use an external repository location (outside the initial project) for your custom IP and to place the entire custom IP directory into the revision control system to preserve all the necessary outputs from the IP packager. This create a temporary project where you can edit the IP configuration. You should have a window called "*Package IP – logic_function*" opened with various sections. The section "Identification" for example allow you to change how the IP will be identified in the Vivado IP catalog.

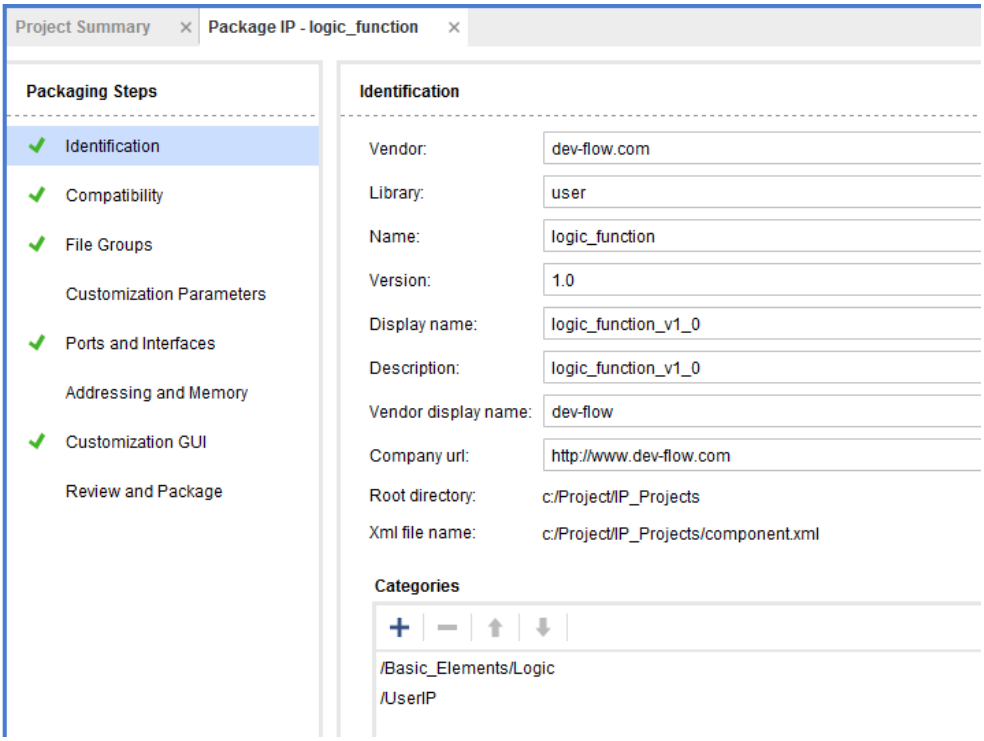
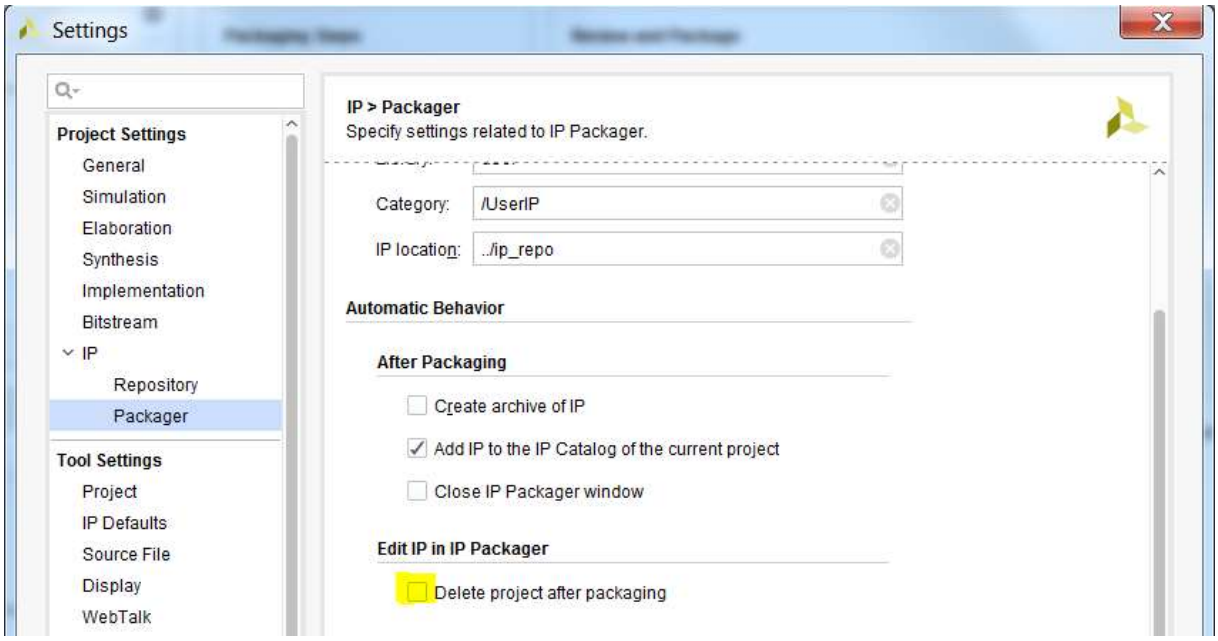


Figure 3 - Package IP – Identification

We won't configure the IP for the moment. Go in the section "*Review and Package*". If you want to keep the IP project (when using a revision control system for example), click on Edit Packaging settings (or go to project settings > IP > Packager) and disable "Delete project after packaging".



Click on "*Package IP*". Vivado will ask you if you want to close the project, click "*Yes*".

Then, in your initial project, if you go in the IP catalog ("*Window > IP catalog*") you should see the IP under UserIP.

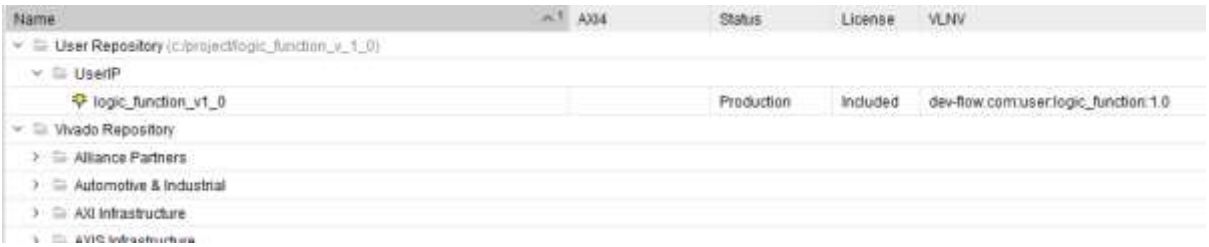


Figure 4 - User IP

> *Note: If you create a new Vivado project, you won't see the IP in the user catalog. This is because the folder where the IP is located is not scanned by Vivado. You need to add this folder to the "IP repositories" in the Project Settings (IP > Repositoryx).*

II. Modifying the IP

Now, we will modify the IP to do a logical AND on buses (not only on 1-bit signals). To modify the IP:

- If you have kept the IP packager project, open this project
- If you haven't kept the IP project, open the IP catalog and search for the IP. Right click on the IP and click on "Edit in IP Packager".

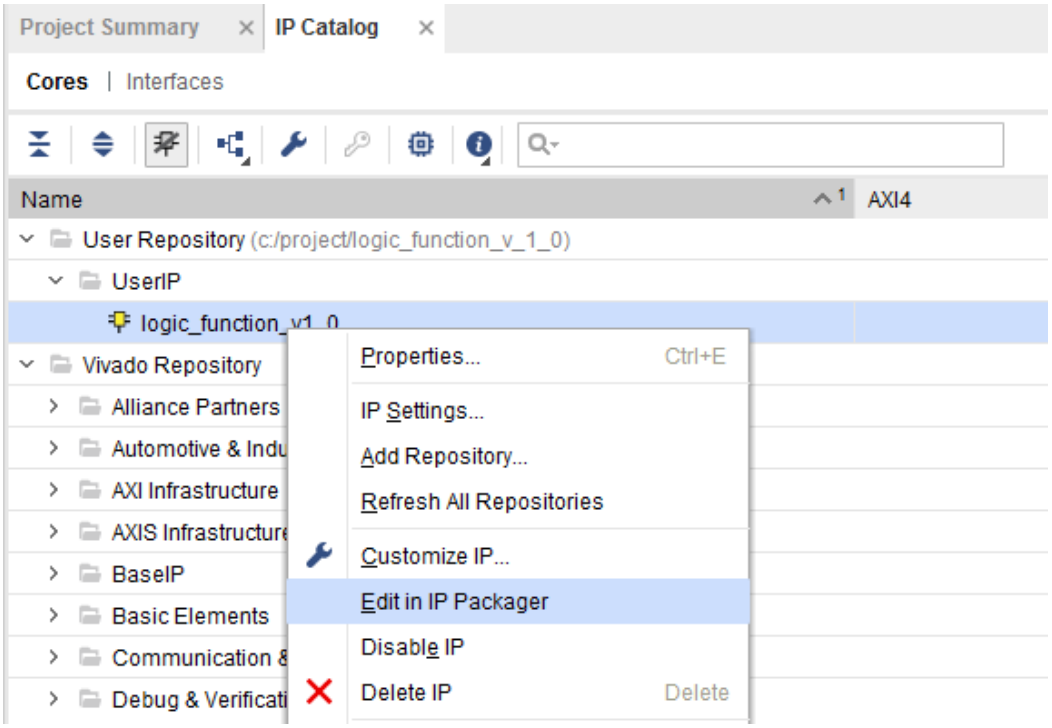


Figure 5 - Edit in IP Packager

In the design sources, doubled click on the source file (logic_function.vhd) to modify it.

Change the entity adding generics values to allow different size of inputs.

```
entity logic_function
   Generic (
      DATA_SIZE : INTEGER := 32
   );
   Port (
      A_inp : in STD_LOGIC_VECTOR(DATA_SIZE-1 downto 0);
      B_inp : in STD_LOGIC_VECTOR(DATA_SIZE-1 downto 0);
      P_outp : out STD_LOGIC_VECTOR(DATA_SIZE-1 downto 0)
   );
end logic_function;
```

In the "*Package IP*" window, you can see that some sections don't have a green mark. This is because we made some changes to the RTL which are not taken in account in the package IP project. To update the project, go in the section "*Review and Package*". Click on one of the "*Merge Changes*" links under *Edit IP Project Changes* to have the changes taken in account in the IP.

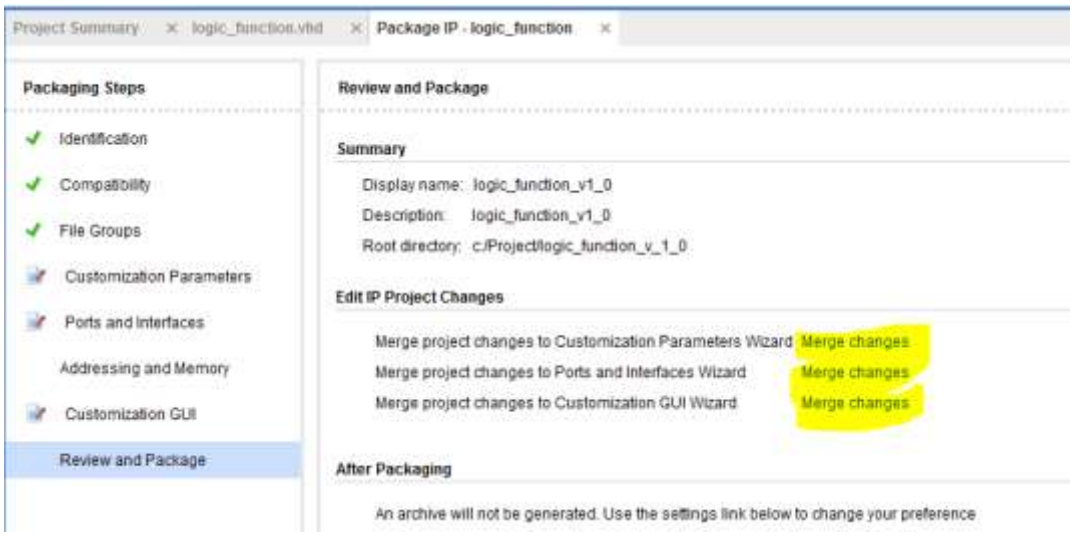


Figure 6 - Merge Changes

Then in *Customization Parameters*, under "*Hidden Parameters*", double click on "*DATA_SIZE*". In the window "*Edit IP Parameter*", enable "*Visible in customization GUI*" to be able to change the value when adding the IP to a project from the IP catalog.

Then enable "*Specify Range*" and select "*Range of Integer*" for type. Enter "*32*" as maximum value and click OK.

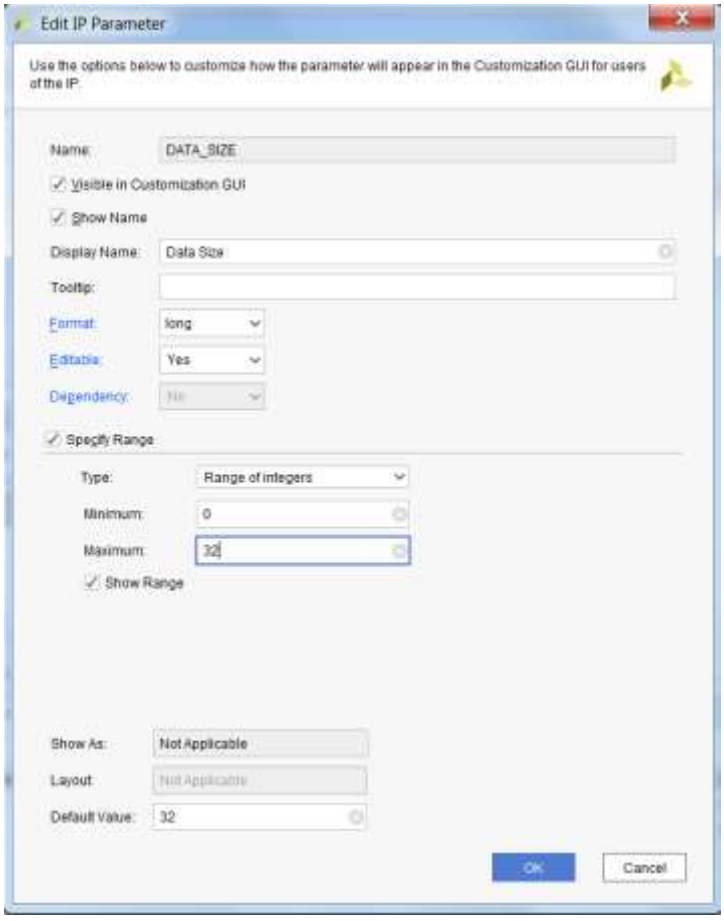


Figure 7 - Edit IP Parameter

In the “Customization GUI”, move the parameter “Data Size” under Page 0

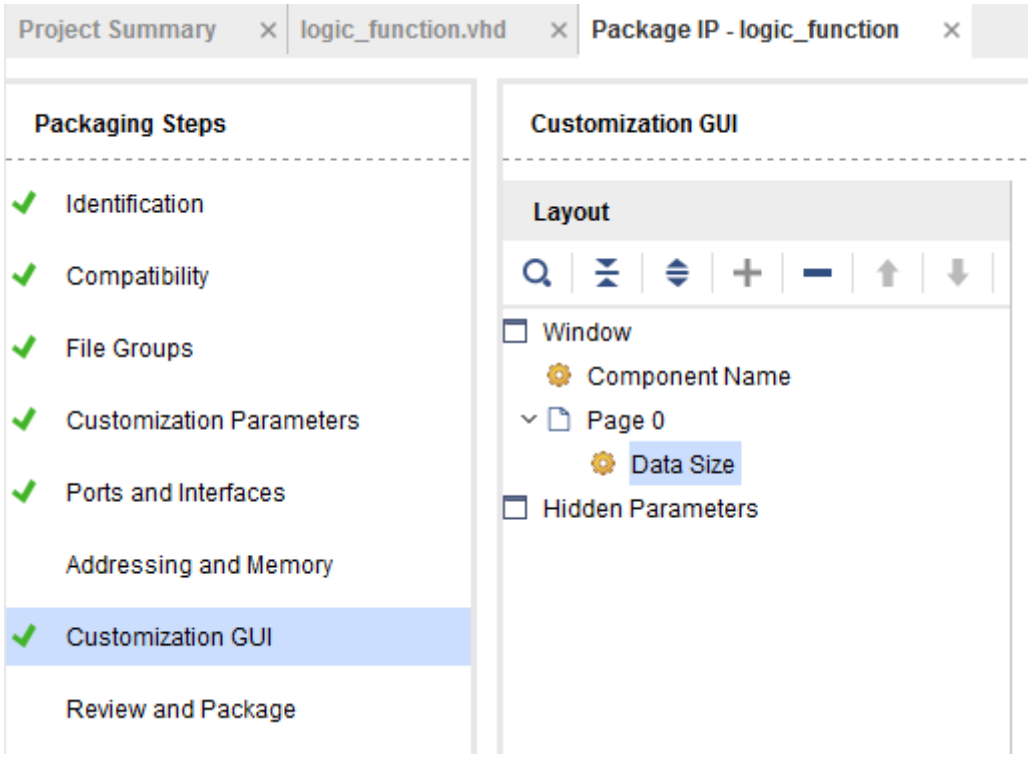


Figure 8 - Customization GUI

Then re-package the IP in the “*Review and Package*” section.

Now, if we add double click on the IP in the IP catalog, we can see that we can select the size of the data in the GUI.

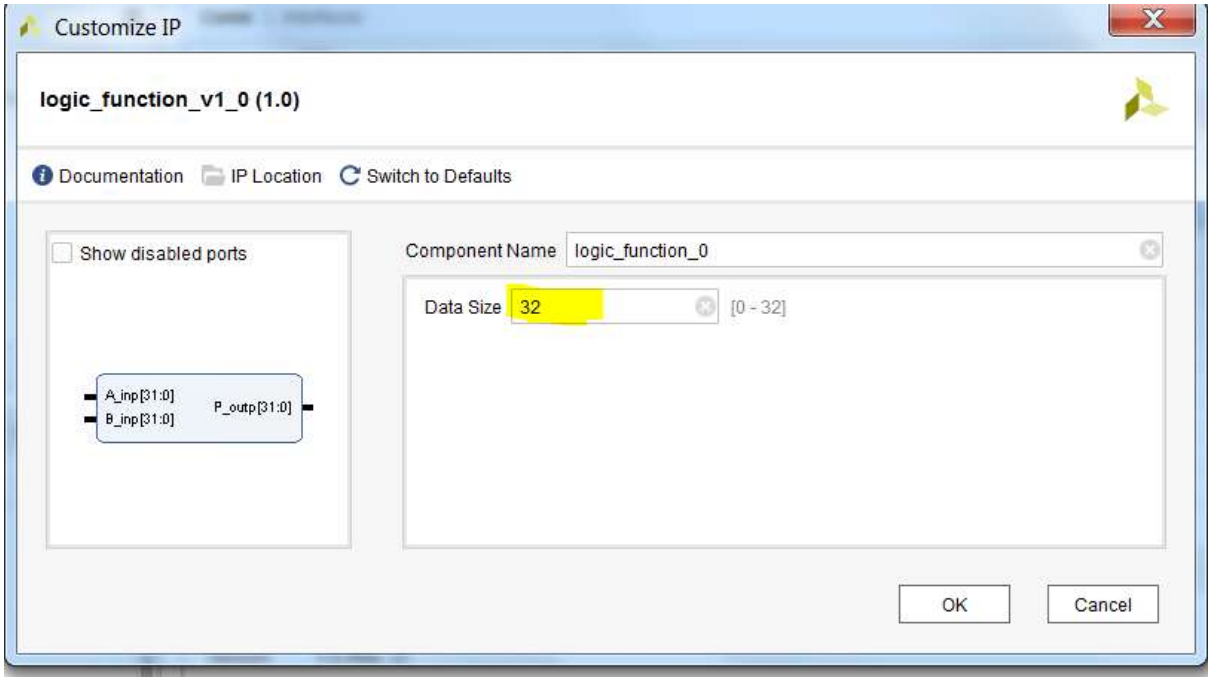


Figure 9 - IP GUI

III. Adding a different functionality

Now, we will add other functions to our IP: logical OR, logical NOR and logical NAND. And the user will be able to select the functionality he wants to use when adding the IP to a project.

Open the IP in "*IP Packager*" and open the source code and modify it with the following code.

```
entity logic_function is
   Generic (
      DATA_SIZE : INTEGER := 32;
      FUNC_SEL : INTEGER := 0
   );
   Port (
      A_inp : in STD_LOGIC_VECTOR(DATA_SIZE-1 downto 0);
      B_inp : in STD_LOGIC_VECTOR(DATA_SIZE-1 downto 0);
      P_outp : out STD_LOGIC_VECTOR(DATA_SIZE-1 downto 0)
   );
end logic_function;

architecture Behavioral of logic_function is

begin

   IF_AND: IF(FUNC_SEL = 0) GENERATE
      P_outp <= A_inp AND B_inp;
   END GENERATE;

   IF_OR: IF(FUNC_SEL = 1) GENERATE
      P_outp <= A_inp OR B_inp;
   END GENERATE;

   IF_NOR: IF(FUNC_SEL = 2) GENERATE
      P_outp <= NOT(A_inp OR B_inp);
   END GENERATE;

   IF_NAND: IF(FUNC_SEL = 3) GENERATE
      P_outp <= NOT(A_inp OR B_inp);
   END GENERATE;

end Behavioral;
```

Save the source file and merge the changes in the "*Package IP*" window. Edit the parameter FUNC_SEL in *Customization parameters*:

- enable "Visible in customization GUI"
- Specify range:
    - Pairs

- Key: And, Value: 0
- Key: Or, Value: 1
- Key: Nor, Value: 2
- Key: Nand, Value: 3

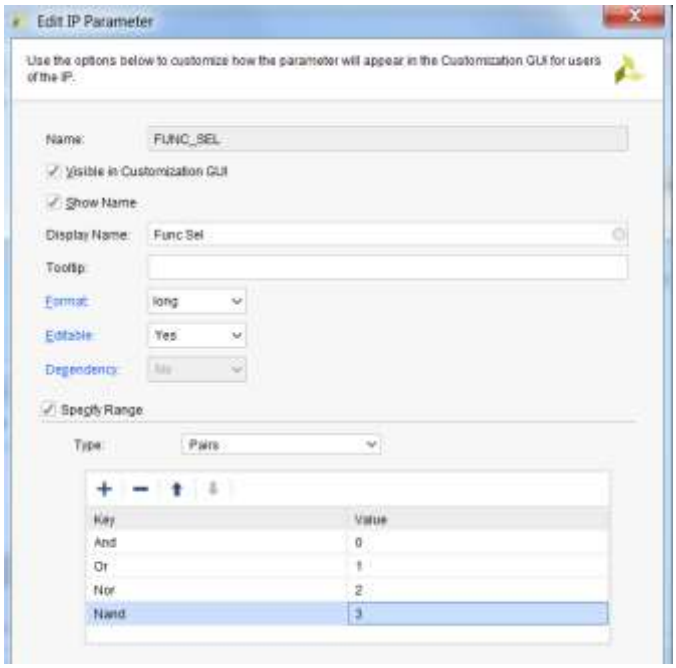


Figure 10 - Customization of parameter FUNC_SEL

In the "Customization GUI", move the parameter "Data Size" under Page 0. Validate and re-package the IP.

Now, from the IP GUI, you can select which function you want.

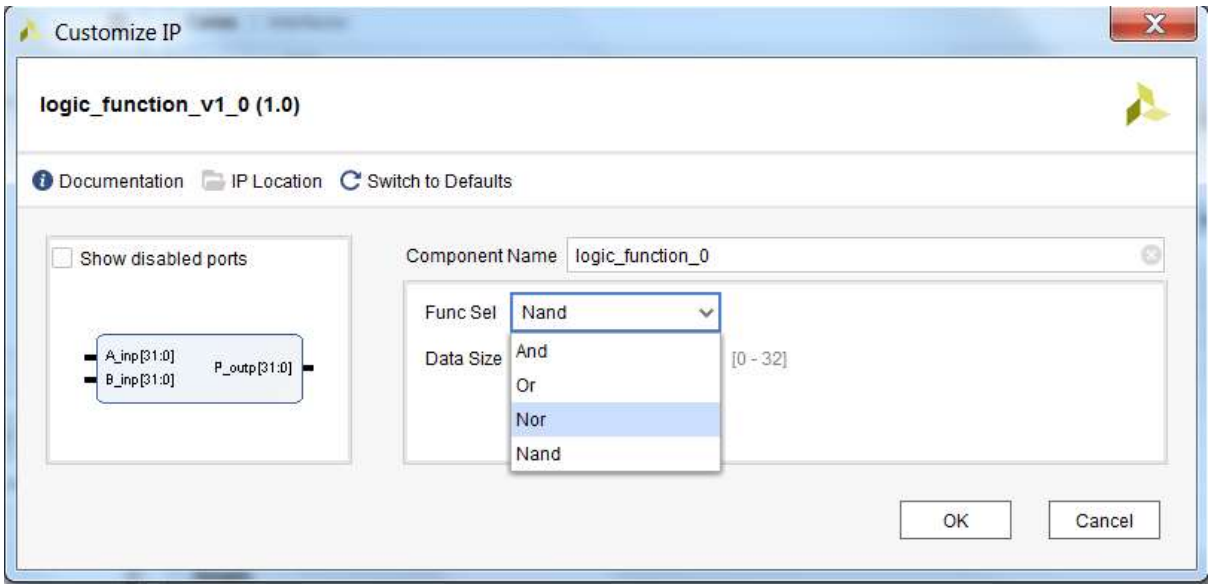


Figure 11 - Function selection in the IP GUI

IV. Unused port

We will now add a new function to our IP: a logic NOT. For this we only need one input port. The other one will be unused so we will disable it from the IP when the function is selected in the customization GUI.

Open the IP in "*IP Packager*" and open the source code and modify by adding the following code.

```
IF_NOT: IF(FUNC_SEL = 4) GENERATE
   P_outp <= NOT(A_inp);
END GENERATE;
```

Save the source file and merge the changes in the "*Package IP*" window. Edit the parameter FUNC_SEL by adding a new pair (key: Not, Value: 4).

In the "Ports and Interfaces" section, double click on B_inp to open the "*Edit Port*" window. If the user want to use the logical NOT function, the input port B_inp will be useless. Thus we need to change *Port Presence* to "*Optional*", set *Driver Value* to "*0*" and write "$FUNC_SEL != 4" in the box below as shown in the Figure 12.

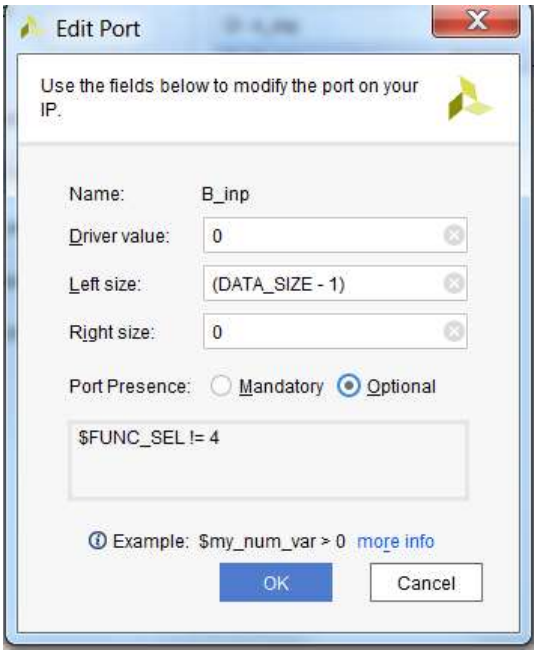


Figure 12 - Edit port B_inp

Re-package the IP.

Now in our original project, we can see that if we select the function "Not", the input port B_inp will be disabled for the IP.