

# RTL Design and IP Generation Tutorial

*PlanAhead Design Tool*

UG675(v14.5) April 10, 2013





#### Notice of Disclaimer

The information disclosed to you hereunder (the "Materials") is provided solely for the selection and use of Xilinx products. To the maximum extent permitted by applicable law: (1) Materials are made available "AS IS" and with all faults, Xilinx hereby DISCLAIMS ALL WARRANTIES AND CONDITIONS, EXPRESS, IMPLIED, OR STATUTORY, INCLUDING BUT NOT LIMITED TO WARRANTIES OF MERCHANTABILITY, NON-INFRINGEMENT, OR FITNESS FOR ANY PARTICULAR PURPOSE; and (2) Xilinx shall not be liable (whether in contract or tort, including negligence, or under any other theory of liability) for any loss or damage of any kind or nature related to, arising under, or in connection with, the Materials (including your use of the Materials), including for any direct, indirect, special, incidental, or consequential loss or damage (including loss of data, profits, goodwill, or any type of loss or damage suffered as a result of any action brought by a third party) even if such damage or loss was reasonably foreseeable or Xilinx had been advised of the possibility of the same. Xilinx assumes no obligation to correct any errors contained in the Materials or to notify you of updates to the Materials or to product specifications. You may not reproduce, modify, distribute, or publicly display the Materials without prior written consent. Certain products are subject to the terms and conditions of the Limited Warranties which can be viewed at <http://www.xilinx.com/warranty.htm>; IP cores may be subject to warranty and support terms contained in a license issued to you by Xilinx. Xilinx products are not designed or intended to be fail-safe or for use in any application requiring fail-safe performance; you assume sole risk and liability for use of Xilinx products in Critical Applications: <http://www.xilinx.com/warranty.htm#critapps>.

©Copyright 2010-2013 Xilinx, Inc. Xilinx, the Xilinx logo, Artix, ISE, Kintex, Spartan, Virtex, Vivado, Zynq, and other designated brands included herein are trademarks of Xilinx in the United States and other countries. All other trademarks are the property of their respective owners.

---

## Revision History

Date	Version	Revision
04/10/2013	14.5	Validated with Release.

# Table of Contents

Revision History.....	2
Table of Contents .....	3
RTL Design and IP Generation Tutorial.....	5
Introduction .....	5
Tutorial Design Description .....	5
Software Requirements.....	6
Hardware Requirements .....	6
Preparing the Tutorial Design Files.....	6
Lab 1: RTL Design .....	7
Step 1: Creating a New RTL Project.....	7
Opening the PlanAhead Tool.....	7
Creating a New RTL Project .....	8
Adding Source Files .....	9
Adding a Constraints File.....	10
Selecting a Default Part .....	11
Step 2: Managing Source Files .....	12
Exploring the Sources View and Project Summary .....	12
Setting VHDL Library Names.....	13
Configuring Simulation Source Files.....	14
Defining Global Include Files.....	15
Setting Compilation Order .....	16
Creating a New RTL Source File and Importing a Template.....	18
Step 3: Running Behavioral Simulation.....	21
Step 4: Elaborating and Analyzing the RTL Design .....	23
Elaborating and Opening the RTL Design.....	23
Examining the RTL Netlist and Hierarchy.....	24
RTL Design and IP Generation Tutorial .....	3

Traversing the Schematic.....	25
Using the Find Command.....	27
Step 5: Estimating Resource Utilization .....	29
Step 6: Running RTL Design Rule Checks.....	31
Step 7: Selecting IP from the Xilinx IP Catalog.....	32
Opening and Searching the IP Catalog.....	32
Customizing an IP Core.....	34
Instantiating the IP Core.....	35
Conclusion.....	38

# RTL Design and IP Generation Tutorial

---

## Introduction

This tutorial provides an overview of the Register Transfer Level (RTL) development and analysis environment, in which you will:

- Import RTL sources and review them using the Text Editor
- Run Behavioral Simulation on the bft module
- Run elaboration to compile the RTL
- Use a variety of RTL analysis features to explore your compiled RTL design. These include:
  - Analyze the RTL logic hierarchy using the RTL schematic
  - Estimate RTL resources
  - Run RTL Design Rules Checks (DRCs)
- Browse the Xilinx® IP Catalog, and customize and implement an Intellectual Property (IP) core in the design

Many of the PlanAhead™ software analysis features are covered in more detail in other tutorials. Not every command or command option is covered.

The objective of this tutorial is to familiarize you with the RTL development and analysis process using the PlanAhead tool.

---

## Tutorial Design Description

The small sample design used in this tutorial includes:

- A RISC processor CPU core
- A pseudo FFT
- Four gigabit transceivers (GTs)
- Two USB interfaces

The design targets an xc7k70t device. A small design is used to:

- Allow the tutorial to be run with minimal hardware requirements
- Enable timely completion of the tutorials
- Minimize data size

---

## Software Requirements

The PlanAhead tool is installed with ISE Design Suite software. Before starting the tutorial, be sure that the PlanAhead tool is operational, and that the tutorial design data is installed.

For installation instructions and information, see the *ISE Design Suite 14: Release Notes, Installation, and Licensing* ([UG631](#)).

---

## Hardware Requirements

Xilinx recommends a minimum of 2 GB of RAM when using the PlanAhead tool on larger devices. For this tutorial, a smaller xc7k70t design is used, and the number of designs open at one time is limited. Although 1 GB is sufficient, it can impact performance.

---

## Preparing the Tutorial Design Files

Copy the files from the ISE software installation area:

**<ISE\_install\_area>/ISE\_DS/PlanAhead/examples/PlanAhead\_Tutorial.zip**

Extract the zip file contents into any write-accessible location which will be referred to in this tutorial as the extraction directory, or *<Extract\_dir>*.



**RECOMMENDED:** *The tutorial sample design data is modified while performing this tutorial. A new copy of the original `PlanAhead_Tutorial` data should be extracted each time you run this tutorial.*

---

# Lab 1: RTL Design

## Step 1: Creating a New RTL Project

The PlanAhead tool lets you create several project types depending on where in the design flow the tool is being used. RTL sources can be used to create a project for development and analysis, synthesis, implementation, and bit file creation.

### Opening the PlanAhead Tool

Open the PlanAhead tool:

- On Windows, select the **Xilinx PlanAhead 14.5** desktop icon or:  
**Start > All Programs > Xilinx Design Tools > ISE Design Suite 14.5 > PlanAhead > PlanAhead**
- On Linux, change the directory to  
`<Extract_Dir>/PlanAhead_Tutorial/Tutorial_Created_Data`, and type **planAhead**.

The PlanAhead Getting Started page opens, providing links to open or create projects, and view the documentation.

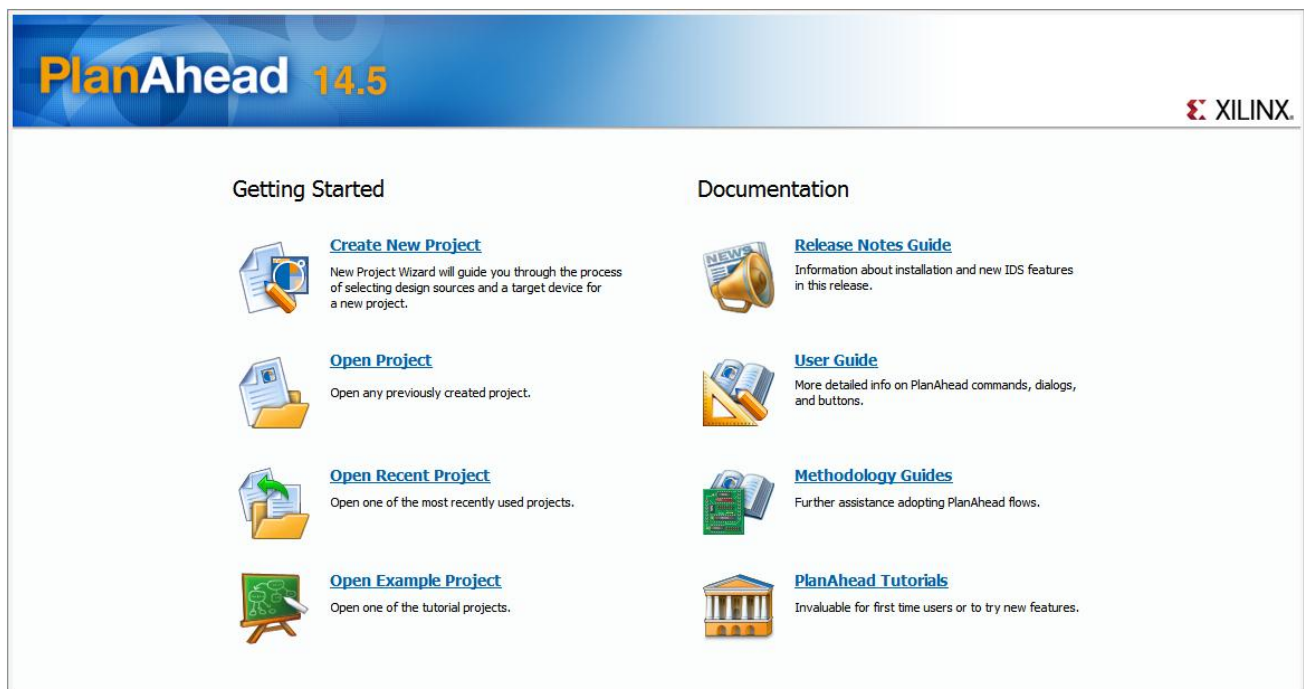
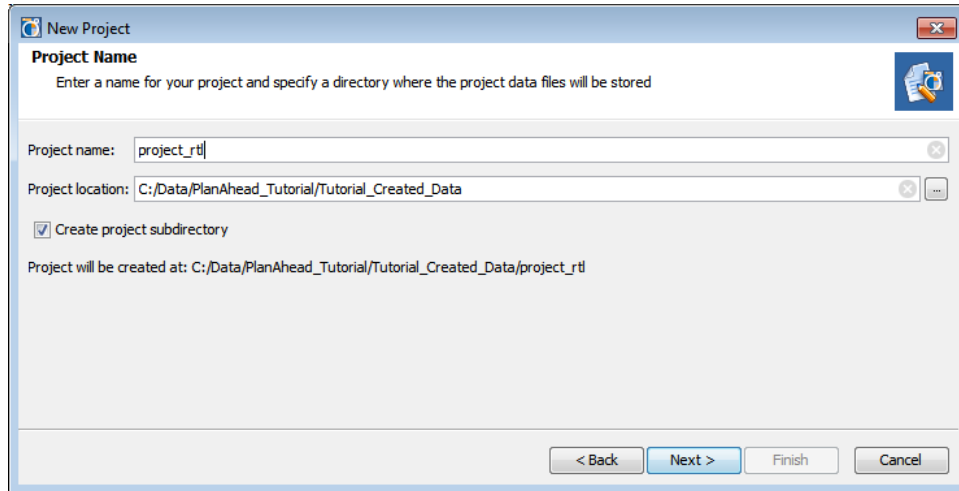


Figure 1: Getting Started Page

## Creating a New RTL Project

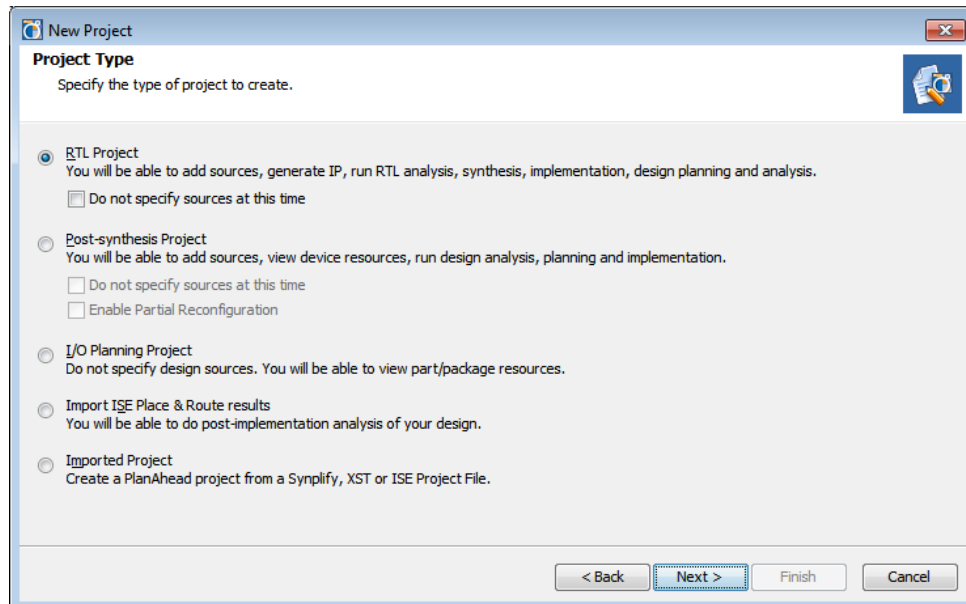
1. On the Getting Started page, select the **Create New Project** link.
2. In the **Create a New PlanAhead Project** confirmation dialog box, click **Next**.

The Project Name page opens:



**Figure 2: Entering the New Project Name**

3. Enter the Project name: **project\_rtl**.
4. For **Project Location**, browse to, and select:  
<Extract\_Dir>\PlanAhead\_Tutorial\Tutorial\_Created\_Data.
5. Click **Next** to open the Project Type page, as shown in [Figure 3](#).



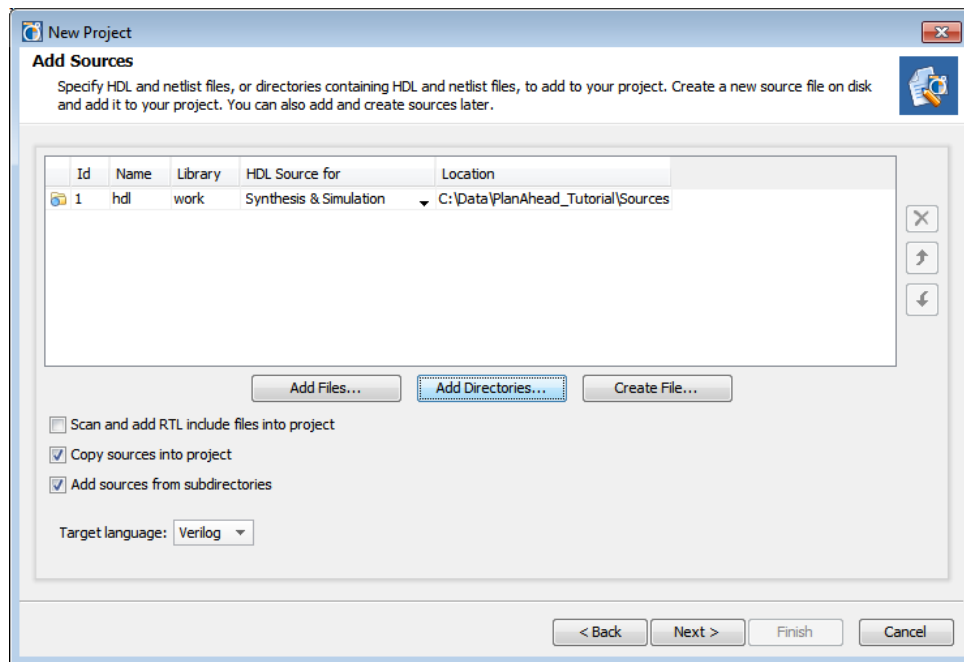
**Figure 3: Project Type**



6. Select Specify RTL Project, and click **Next**.

The Add Sources page opens:

## Adding Source Files



**Figure 4: New Project: Add Sources Dialog Box**

1. Click **Add Directories**, browse to:  
`<Extract_Dir>/PlanAhead_Tutorial/Sources/hdl`
2. Verify that the following checkboxes are selected:
  - Copy sources into project.
  - Add sources from subdirectories.
3. Check that the Add Source dialog box matches what is shown in [Figure 4](#), and click **Next**.  
The Add Existing IP page opens.
4. Click **Next** to skip past this since you will not be adding any IP at this time.  
The Add Constraints Files page opens:

## Adding a Constraints File

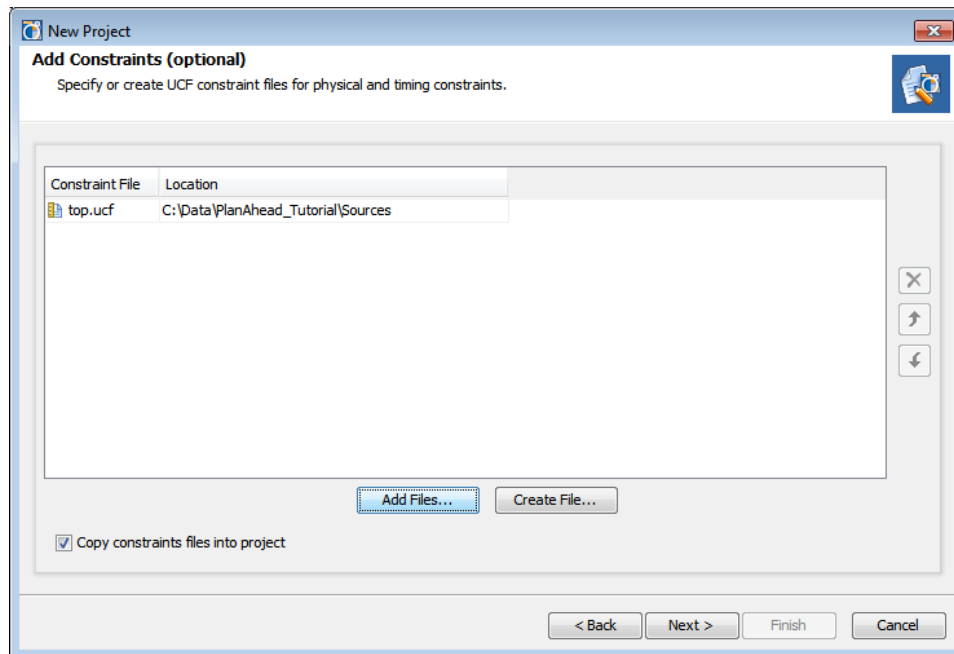


Figure 5: Selected Constraint File to Add to the Project

1. Click **Add Files** and browse to select the following file:  
`<Extract_Dir>/PlanAhead_Tutorial/Sources/top.ucf`
2. Click **OK**.
3. Ensure the **Copy constraints into project** option is set to on, and click **Next**.  
The Default Part page opens.

## Selecting a Default Part

1. In the Filter section, click the **Family** pull-down menu, and select **Kintex-7**. The list is filtered to only show Kintex®-7 devices
2. In the Search field type **70t**. The 70t devices are listed.

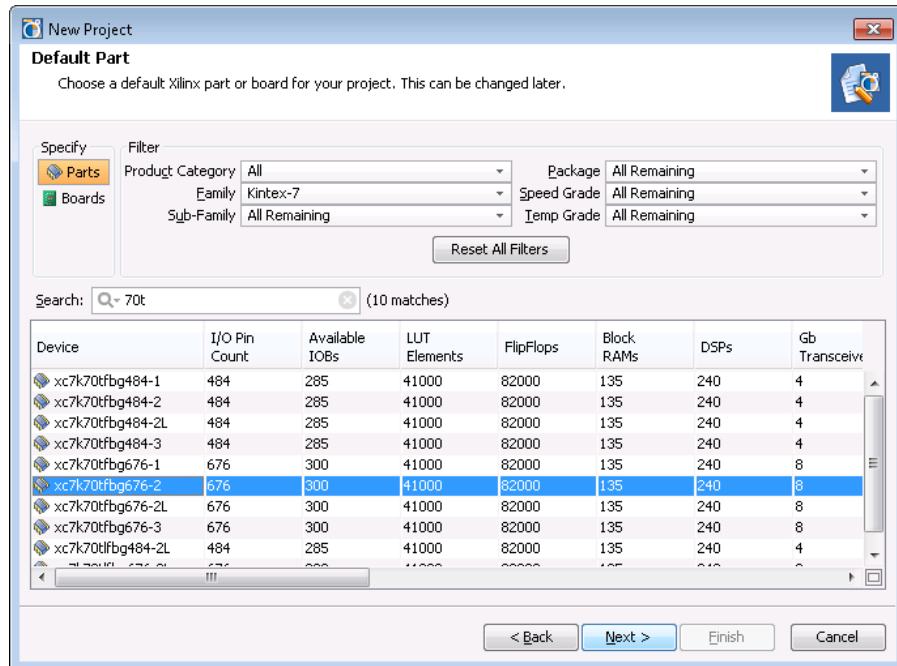


Figure 6: Selecting a Family and Default Part

3. Select **xc7k70tfbg676-2** and click **Next**.
4. Review the New Project Summary page, and click **Finish**.

The PlanAhead environment opens.

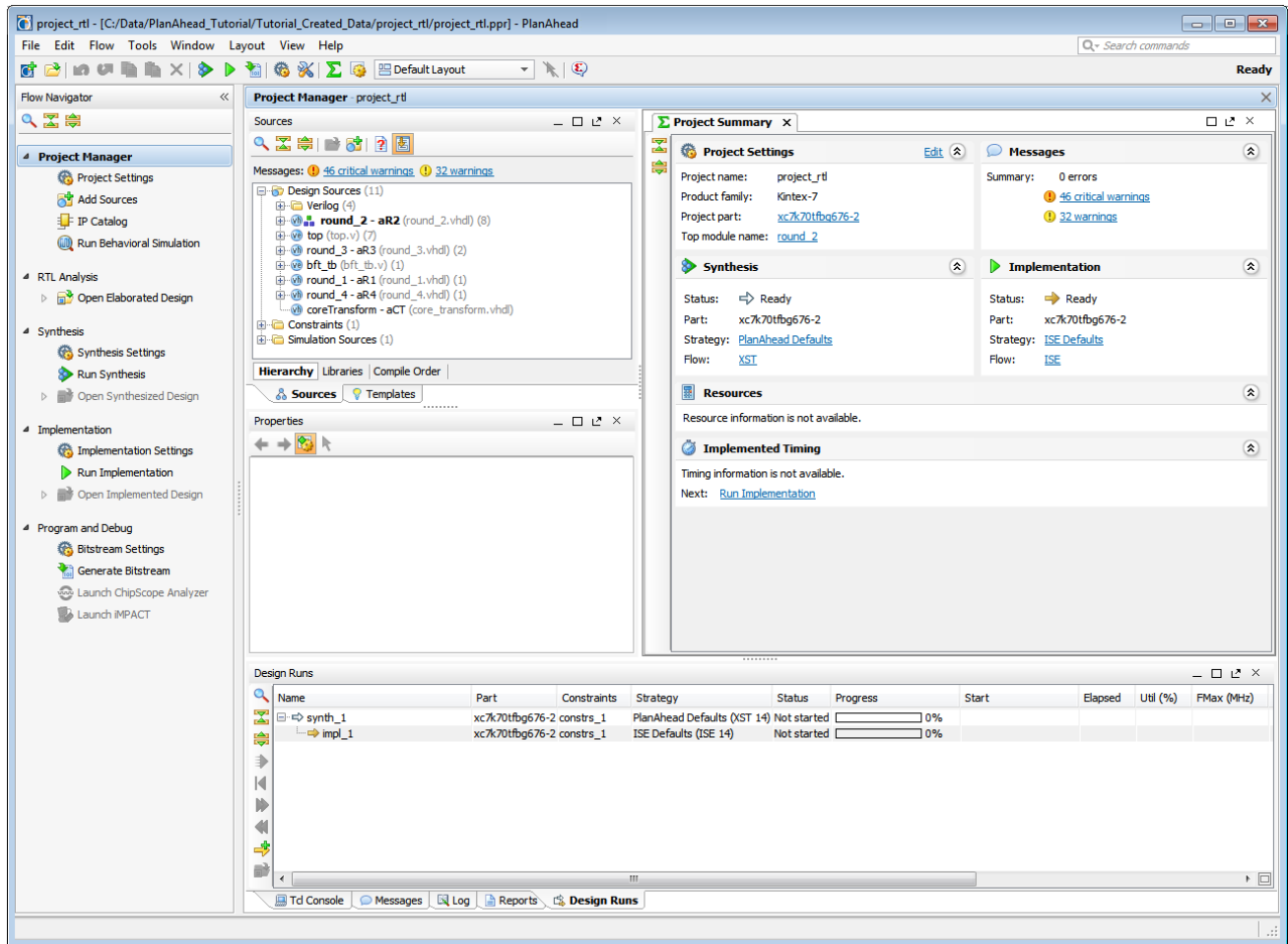


Figure 7: Default View Layout

## Step 2: Managing Source Files

The PlanAhead tool allows different file types to be added as design sources, including Verilog, VHDL, and NGC format cores. The files are listed by category in the Sources view.

### Exploring the Sources View and Project Summary

1. Examine the information currently displayed in the Project Summary of the new project. More information is displayed as the design progresses through the design flow.
2. Examine the Sources view as shown in Figure 8.

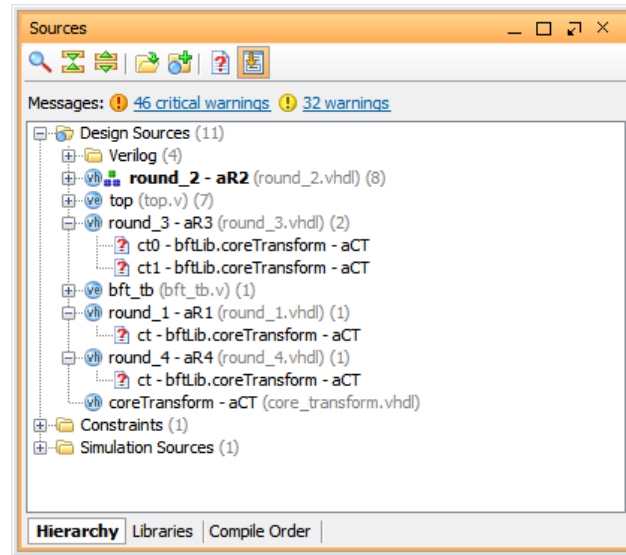


Figure 8: Viewing the Hierarchy

The Hierarchy tab of the Sources view is displayed by default. The Design Sources folder currently lists `round_2` as the top module of the design, as indicated by the hierarchy icon. The PlanAhead tool will automatically determine the top module; however, at this time many of the source files in the project are not properly defined, preventing this feature from being successful. The `bft.vhdl` file references entities from the `bftLib`, however several VHDL files have incorrect library names which prevent the entities from being found as shown by the missing file icon in Figure 8. You need to correct the Library names in order for the design to compile and display the correct hierarchy.

## Setting VHDL Library Names

VHDL Library names can be set for individual files or for directories when using the Add Sources dialog box. When you created this project, you added the `hdl` directory while adding source files. This included the `bftLib` subdirectory, but you did not specify the Library Name for the files in that directory at that time. You will do that now.

1. Click the **Libraries** tab.
2. **Expand** the **Work** and **Unreferenced** folders in the **VHDL** folder.
3. Hold the **Ctrl** key and select **bft\_package.vhdl**, **round\_1-4.vhdl**, and **core\_transform.vhdl**
4. Right-click to open the popup menu, and select the **Set Library** command.
5. Enter the Library Name: **bftLib**.

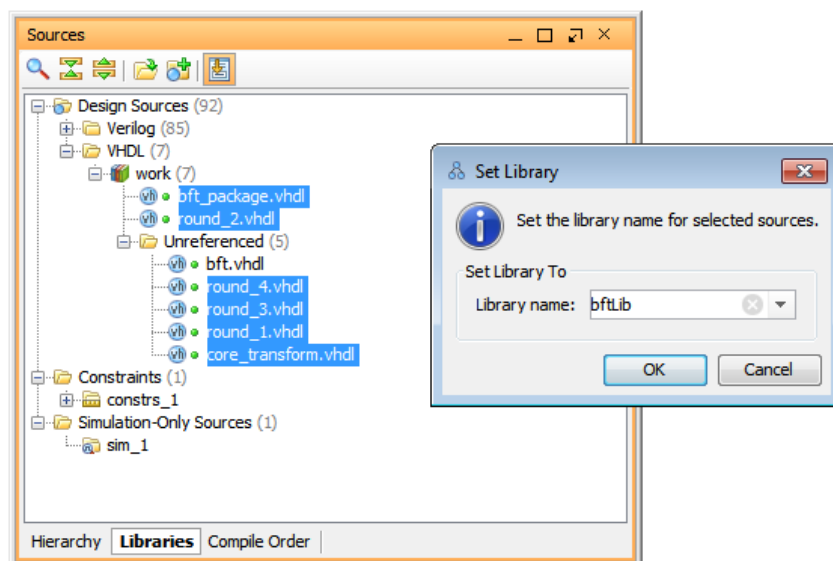


Figure 9: Setting VHDL Library Names

6. Click **OK**.
7. **Verify** that the **VHDL** folder now looks as shown in [Figure 10](#).

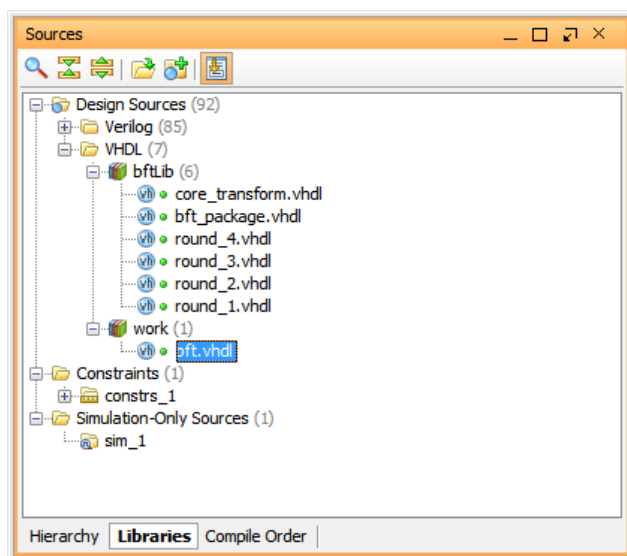



Figure 10: VHDL Library

## Configuring Simulation Source Files

1. In the Libraries tab of the Sources view, **expand** the **Verilog** folder and the **work** subfolder.
2. Scroll to the bottom of the work folder and **expand** the **Unreferenced** folder.

You should see the **bft\_tb.v** source file. It is a simulation test bench and needs to be set to Simulation Only.

3. Select the `bft_tb.v` file and right-click to select **Move to Simulation Sources**.
4. Click the **Collapse All** button  in the Sources view header.
5. Expand the **sim\_1** folder under Simulation-Only Sources.

The test bench file is now shown as unreferenced under the **Simulation-Only Sources** folder. The file will become a referenced file when it is set as the top file for simulation in a later step.

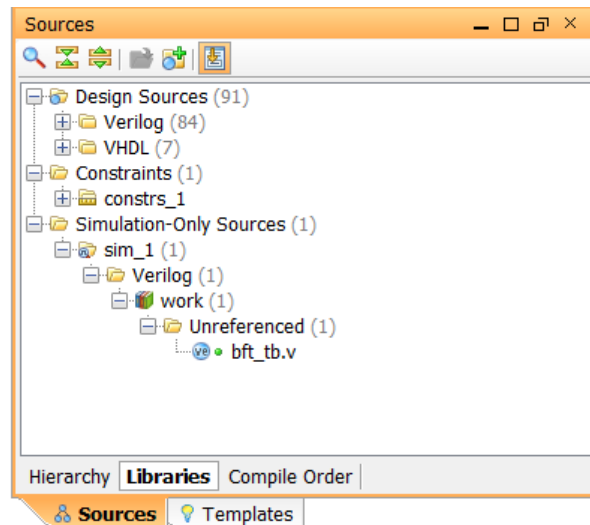



Figure 11: Viewing the Simulation-Only Sources

## Defining Global Include Files

1. In the Sources view header, click the **Search** button. 
2. In the Search field, enter **timescale**.  
Notice that only files with that name appear.
3. Select the Verilog `timescale.v` file, and right-click it to select **Set Global Include**.  
Notice that the file displays under the new Global Include folder. This insures that the same ``timescale` directive is included for all Verilog source files.

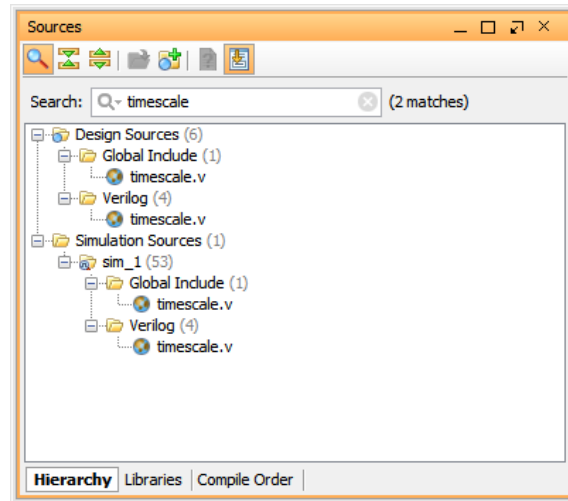


Figure 12: Set Global Include

4. In the Sources view header, click the **Search** button  to close the Search field and to display all files.

## Setting Compilation Order

The PlanAhead tool will automatically select a top module, and order and display source files based on compilation order required by the top module. However, you can also manually specify a top module.

1. In the Sources view, click the **Hierarchy** tab.

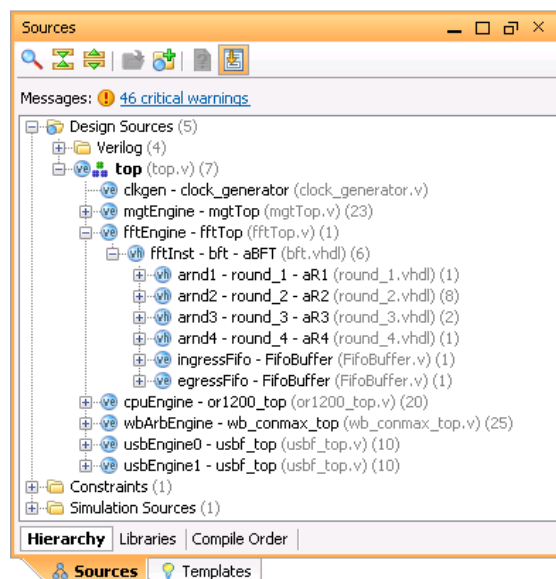
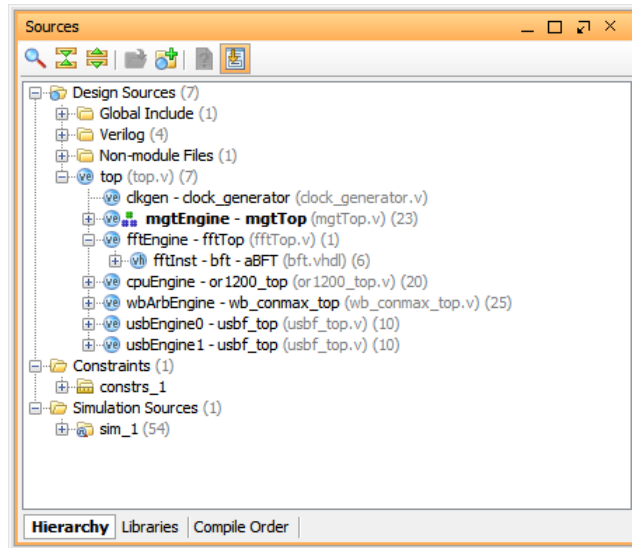


Figure 13: Hierarchy with Top Module




Notice that the design hierarchy now displays as shown in [Figure 13](#). When the VHDL library, `bftLib`, was properly defined, and the various entities and modules referenced by the design could be located, the PlanAhead tool identified the top module of the design, and sorted the source files for compilation according to the requirements of that top module.

2. In the Hierarchy tab of the Sources view, expand the top module, and select the `mgtEngine`.



**Figure 14: Specifying the Top Module**

3. Right-click to open the popup menu, and select **Set as Top**

The `mgtEngine` block is defined as the top-module in the Sources view, and is marked by the hierarchy icon  as shown in [Figure 14](#), indicating it is now the top-level of the hierarchy.

4. In the Sources view, **select** the **Compile Order** tab.

You will notice that the required source files and the compile order of those files are updated based on the new top module.

**Note:** The Compile Order tab displays source files in the order in which they will be compiled, first to last. The top module is usually the last.

To manually enable or disable a file, select the file, right-click to open the popup menu, and select **Enable File** or **Disable File**. To manually change the Compile Order, select the file, right click, and select **Move to Top**, **Move Up**, **Move Down**, or **Move to Bottom**.

If you move or reorder files in the Compile Order tab, the PlanAhead tool may prompt you to enable manual compile order as shown in [Figure 15](#). You can also turn on manual compile order by setting the Hierarchy Update mode from the popup menu of the Hierarchy tab of the Sources view.

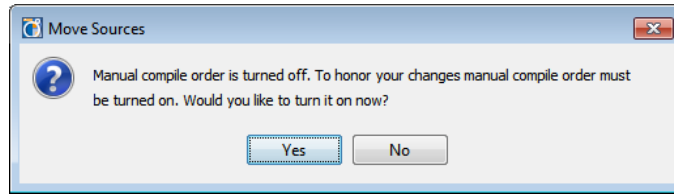


Figure 15: Enable Manual Compile Order

5. **Select** the **Hierarchy tab** in the Sources view.
6. Right-click in the Sources view, select **Hierarchy Update**.  
Examine the three different update modes that are available, ranging from fully automatic to completely manual file control.
7. Hit the **Esc** key to cancel the context menu.  
For this exercise though, you will now restore the top module.
8. **Select** the **top** module, right-click and select **Set as Top**.  
The source file and compile order should be updated to reflect the restored top module.
9. In the Sources view, **Select** one of the **VHDL** or **Verilog** files, right-click to examine the available popup menu commands for source files. To dismiss it, press the **Esc** key.
10. In the Sources view, **double-click** a VHDL or Verilog **source file** to open it in the Text Editor.
11. Right-click in the Text Editor to view the available popup commands.
12. Close the Text Editor by clicking on the **X** in the view tab.

## Creating a New RTL Source File and Importing a Template

The PlanAhead tool enables you to create new Verilog or VHDL source files. Xilinx provides standard language templates for you to use as a starting point for a variety of logic and code constructs.

1. In the Flow Navigator under Project Manager, select Add Sources.
2. In the Add Sources dialog box, select **Add or Create Design Sources**, and click **Next**.
3. Click **Create File...** button in the Add or Create Design Sources dialog box.

The Create Source File dialog box opens:

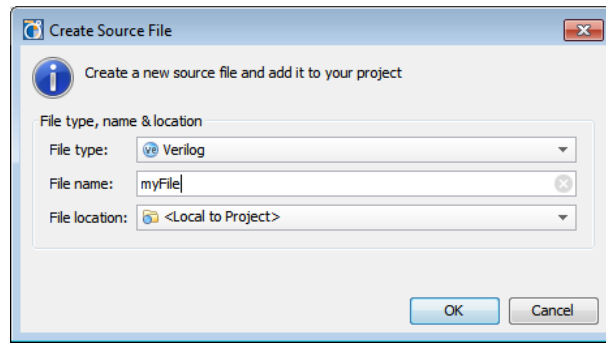


Figure 16: Create Source File

4. In the File name field, type **myFile**, and click **OK**.
5. In the **Add Sources** dialog box, click **Finish**.

The Define Modules dialog opens to help you to define the ports of a new Verilog module or VHDL entity. In this case however, you will use a template from the Xilinx Language Templates to define this module.

6. Click the **Cancel** button and **Yes** to confirm.

The new empty `myFile.v` is added to the Sources view, and listed under the Non-module Files folder list in the Sources view.

7. In the Sources view, expand the **Non-module Files** folder and double-click **myfile.v** to open the file in the Text Editor.
8. Click the **Templates** view tab next to the Sources view, or use the **Window > Language Templates** command from the main menu.
9. **Expand** the **Verilog** subfolders to examine the types of templates available.

[Figure 17](#) shows the Language Templates view. The Xilinx Language Templates provide many standard logic elements for the different Xilinx FPGAs in both Verilog and VHDL, as well as UCF constraint templates for defining design constraints. The top half of the Language Templates view lists the available language and constraint templates.

For this exercise you can select any Kintex-7 template. In [Figure 17](#) the Global Clock Buffer with Clock Enable (BUFGCE) has been selected from the Kintex-7 folder of the Device Primitive Instantiation folder. The bottom half of the Language Templates view displays a preview of the currently selected template.

10. In the Text Editor, right-click and select **Insert Template**.

The template text is now inserted in the `myFile.v` file. The template text is inserted at the current location of the cursor in the text Editor window.

**Note:** You can also copy text from the preview pane of the Language Template view by selecting the desired text and using the Copy command. You can then paste it into the Text Editor window as needed.

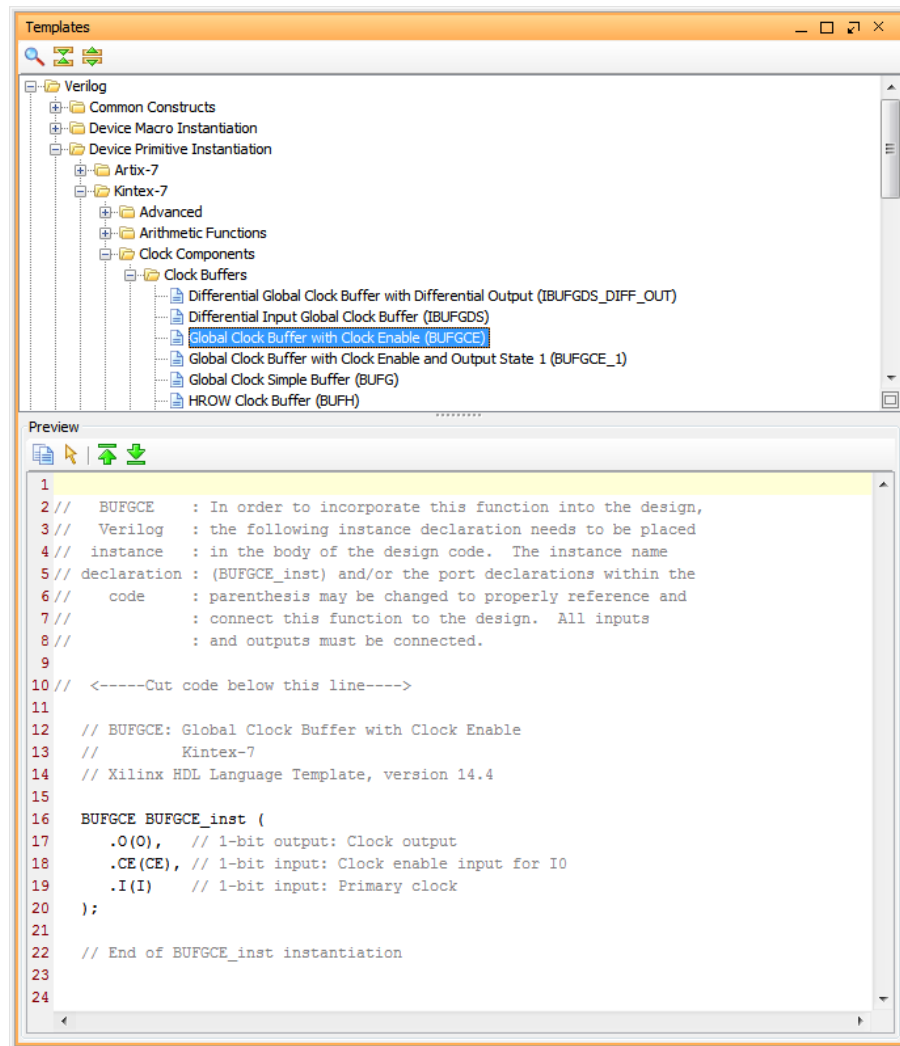


Figure 17: Xilinx Language Templates

11. Close `myFile.v` in the Text Editor by clicking the **X** button.
12. In the Save Text Editor Changes dialog box, **Click No** to close the file without saving.
13. In the Sources view, click the **Collapse All** button.

## Step 3: Running Behavioral Simulation

The Xilinx ISE Simulator (ISim) logic simulation environment is integrated with the PlanAhead tool. ISim can be used for behavioral or timing simulation. You can run behavioral logic simulation on the entire design, or on an individual module.

1. In the Flow Navigator, select **Run Behavioral Simulation**.

The Launch Behavioral Simulation dialog box opens.

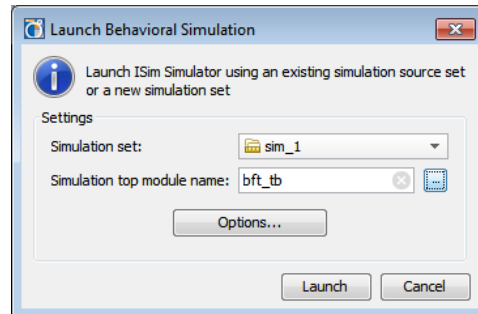


Figure 18: Launch Behavioral Simulation

2. **Click** the file browser icon for the **Simulation Top Module Name** field.

This field lets you define the top-level module to launch simulation from. In this exercise, you will specify the test bench for the bft module which can be found in the fftEngine block under the Sources view.

3. **Select bft\_tb**, and click **OK**.
4. **Click** the **Options** button on the Launch Behavioral Simulation dialog box.

This opens the Simulation Options form as shown below. This form lets you define various options and directives for the ISim fuse compiler and simulator. Refer to the *PlanAhead User Guide* ([UG632](#)), for more information on these various options.

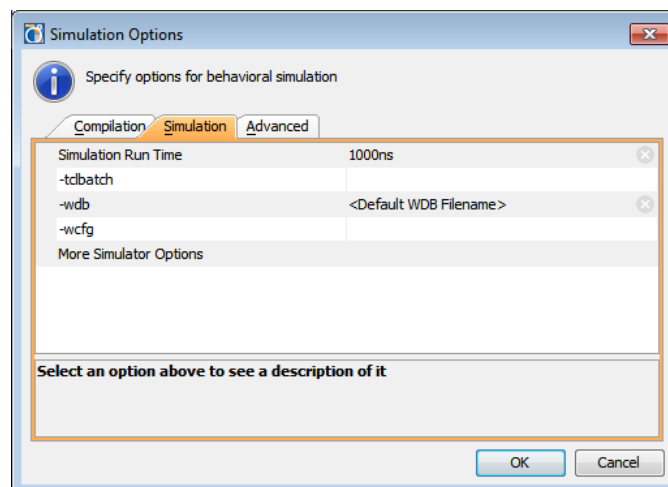
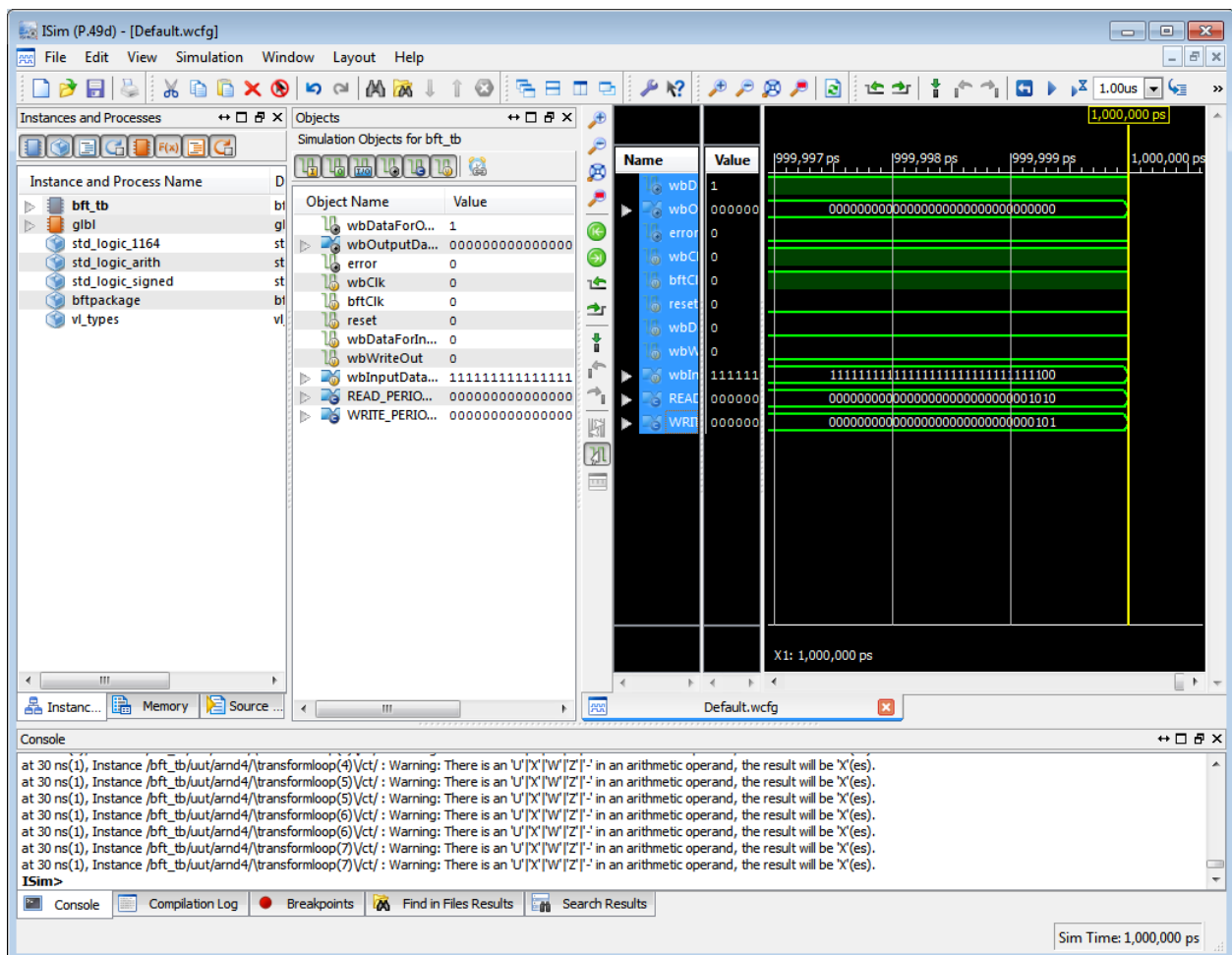


Figure 19: Simulation Launch Options

5. Click **Cancel** to close the Simulation Options dialog box.
6. Click **Launch**, and wait for ISim to open.



**Figure 20: Running ISim for Behavioral Simulation**

You can explore the ISim environment at this time. Refer to the *ISim In-Depth Tutorial* ([UG682](#)), for information about using ISim.

7. Close the ISim window by using the **File > Exit** command from the main menu.
8. Click **Yes** to confirm closing the tool if necessary.

## Step 4: Elaborating and Analyzing the RTL Design

The PlanAhead tool provides RTL elaboration capabilities to compile RTL source files in the project. Once elaborated, the RTL views enable cross-selection of logic objects. Displayed compilation errors and warnings are cross-selectable to module definition or instantiation lines in the RTL code. The RTL logic hierarchy is expanded and available for analysis. Opening the RTL Design from the Flow Navigator automatically elaborates the RTL design and displays the Design Analysis view layout.

- The RTL Netlist and Hierarchy views display the logic hierarchy of the design.
- The RTL Schematic enables interactive logic exploration.
- The Find command enables searching of RTL logic objects.
- The Instance Properties view displays information about the selected logic instantiation including resource estimation.
- The RTL DRCs highlight potential areas of the design to improve power or performance.

### Elaborating and Opening the RTL Design

1. Under RTL Analysis in the Flow Navigator, select **Open Elaborated Design**.
2. If prompted, click **OK** in the Warnings dialog.  
The RTL Design is elaborated and opened.
3. Click the **Messages** view tab and click the **Collapse All** toolbar button
4. Expand the **Elaborated Design** folder, and scroll through the various messages associated with the `open_rtl_design` command.

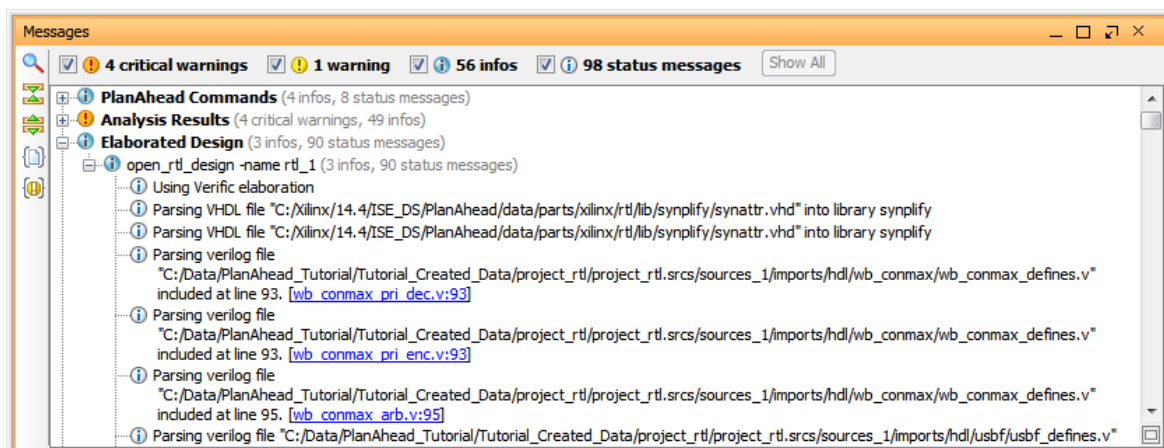


Figure 21: Viewing RTL Elaboration Messages

## Examining the RTL Netlist and Hierarchy

1. In the RTL Netlist view, expand the usbEngine0 instance by clicking the plus sign (+).
2. Select the **usbEngine0/u0** instance.
3. In the RTL Netlist view, right-click to open the popup menu, and select the **Go to Definition** popup command.

The RTL file `usbf_utmi_if.v` opens in the Text Editor. This is the RTL code that defines the UTMI Interface module. The file opens to the line containing the `usbf_utmi_if` module definition.

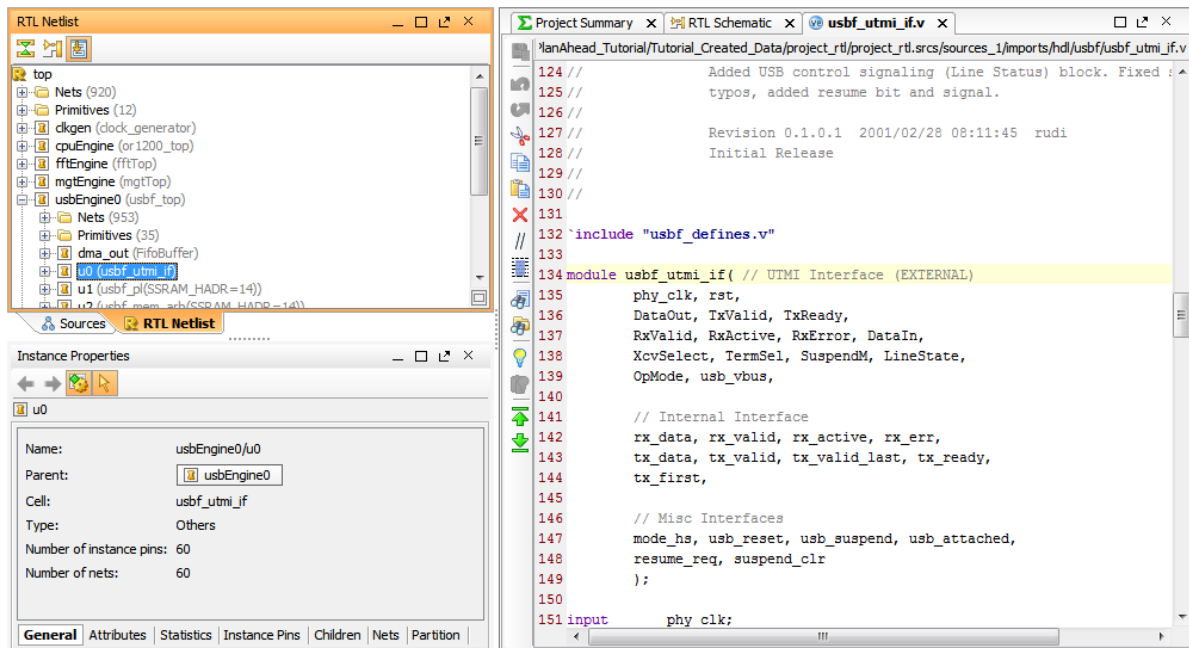


Figure 22: RTL Netlist - Go to Definition

4. In the RTL Netlist view, right-click and select the **Go to Instantiation** popup command.

The RTL file `usbf_top.v` opens to the line containing the `usbf_utmi_if` instance.

5. In the RTL Netlist view, right-click and select the **Show Hierarchy** popup command.

The RTL Hierarchy view opens with the `usbEngine0/u0` module selected. The modules display with blocks sized relative to the amount of logic contained in them, making it easy to locate large modules.



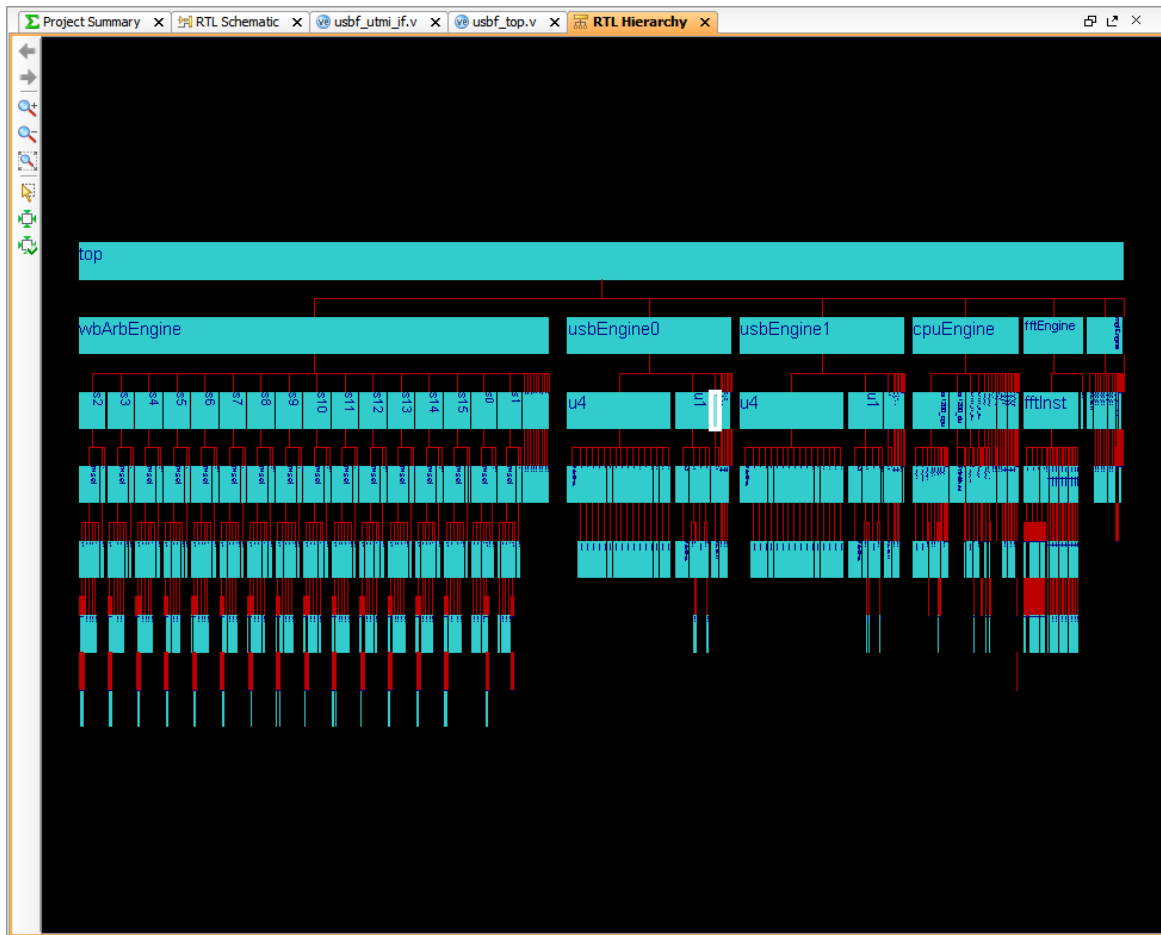
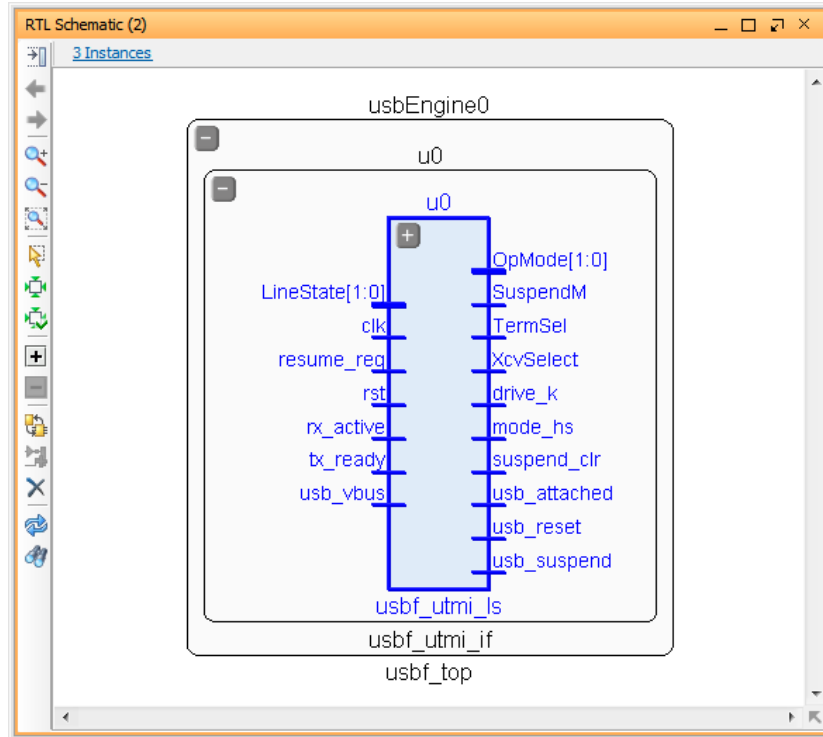


Figure 23: Hierarchy View

6. To close the RTL Hierarchy, click **X** on the view tab.
7. To close the Text Editor, click **X** on all open RTL files.

## Traversing the Schematic

1. In the RTL Netlist view, expand and select the usbEngine0/u0/u0 instance (the level below the previous selection).
2. In the RTL Netlist view, click the **Schematic** button to open the schematic window.  
You can also right-click and select the **Schematic** command from the popup menu.



**Figure 24: RTL Schematic**

- Double-click the **LineState[1:0]** pin on the outside of the u0 module to expand the schematic outward from the pin as shown in Figure 20.

For additional schematic exploration capabilities, see the *Design Analysis and Floorplanning Tutorial: PlanAhead Design Tool* ([UG676](#)).



**TIP:** Click and drag using the left mouse button in the RTL Schematic view from the lower right to the upper left to use the Zoom Fit command.

- Select the **LineState\_r** instance in the RTL Schematic view.
- In the Schematic view, right-click to open the popup menu and select the **Go to Instantiation** to go to the selected logic instance in the `usb_f_top.v` file.

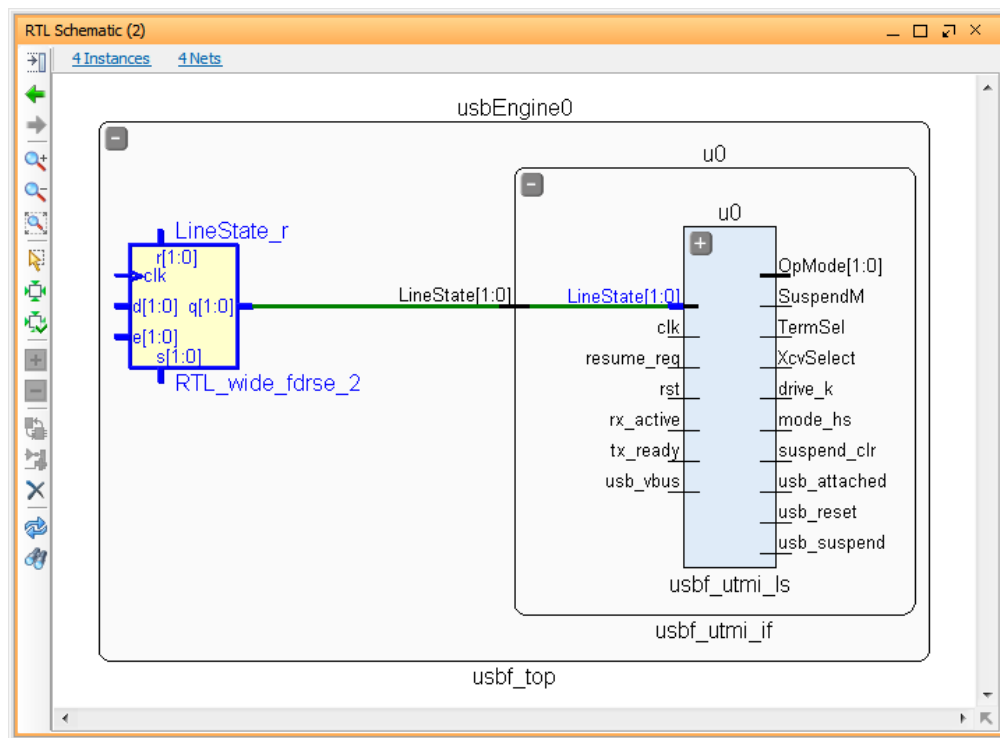




Figure 25: Expanding Logic

6. Close the various open Text Editor, Hierarchy, and Schematic windows.
7. In the RTL Netlist view, click the **Collapse All** button .

## Using the Find Command

1. Click the Find button in the main toolbar , or select Edit > Find to open the Find dialog box.

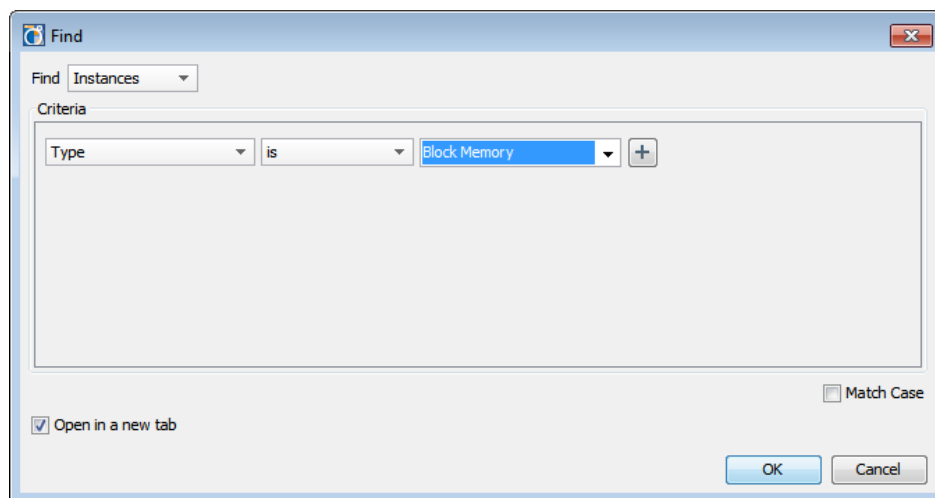
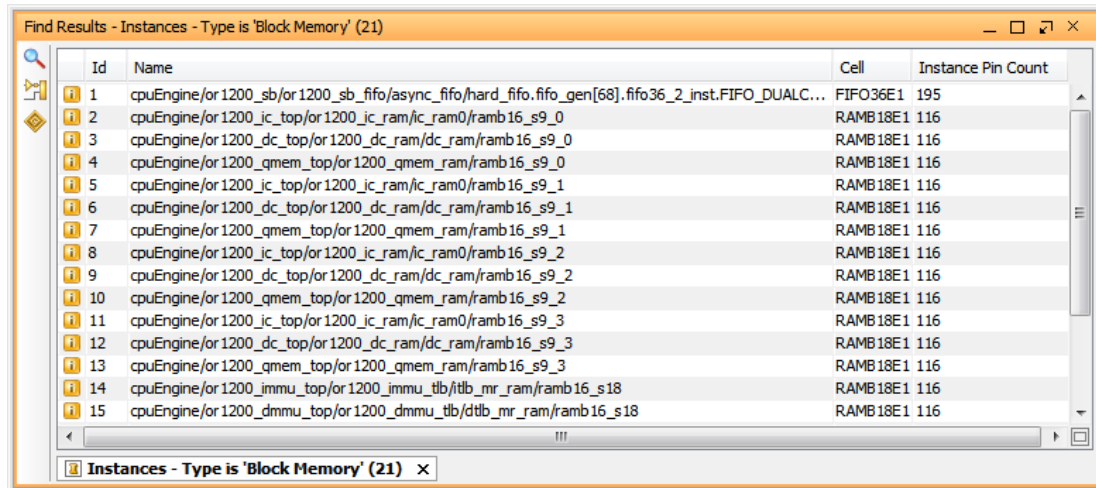


Figure 26: Using Find

2. Examine the Find filter options.
3. Set the Criteria to **Type > is > Block Memory**, and click **OK**.

The Find Results view opens to display the results of the find operation.



**Figure 27: Find Results**

4. Select one of the Block RAMs in the Find Results; right-click to open the popup menu and select **Go to Instantiation**.

The instance is selected in the RTL Netlist view and displayed in the Text Editor.

5. Close the Find Results view and the Text Editor.

## Step 5: Estimating Resource Utilization

With the elaborated RTL Design opened, you can begin to examine the resource requirements of the design. The Project Summary provides a view of the Resources required by the design at each step in the design flow. Obviously the estimation of required resources in the RTL design is going to be less accurate than the resources utilized by the synthesized or implemented design, but it does begin to offer some insight into the requirements of the design.

1. In the Project Summary, select the **Run Report RTL Utilization** command.

The RTL Resource Estimation opens in the Project Summary.

2. In the RTL Netlist view, **select top**.
3. In the Netlist Properties view, **select** the **Statistics** tab.

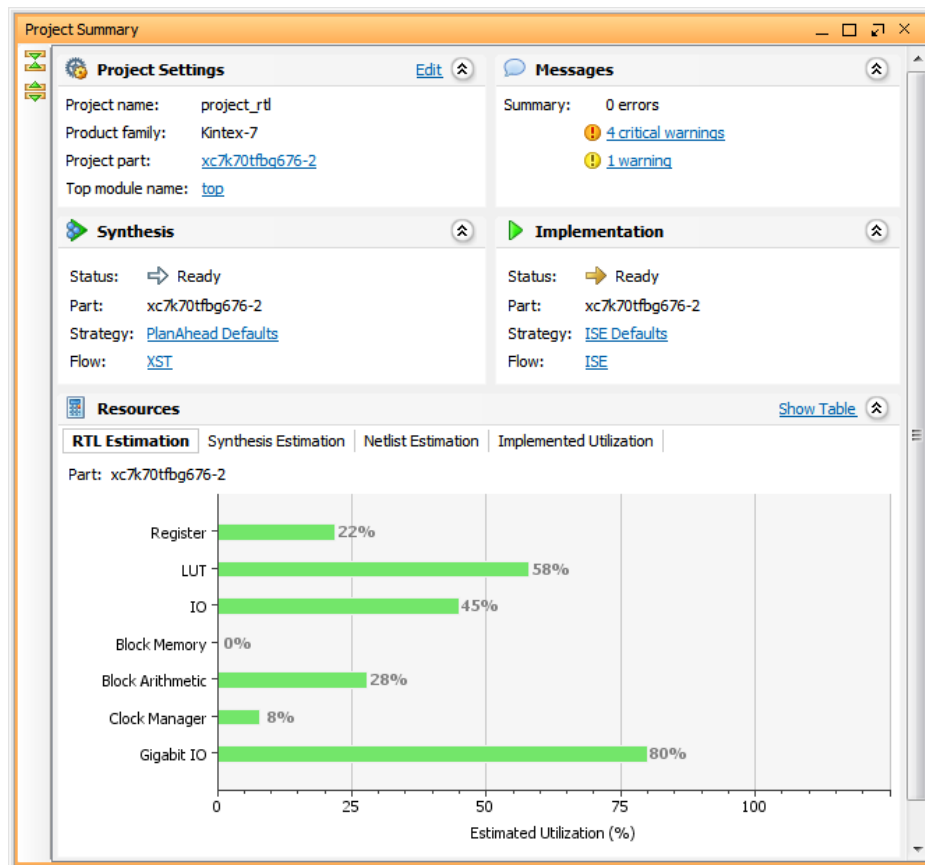
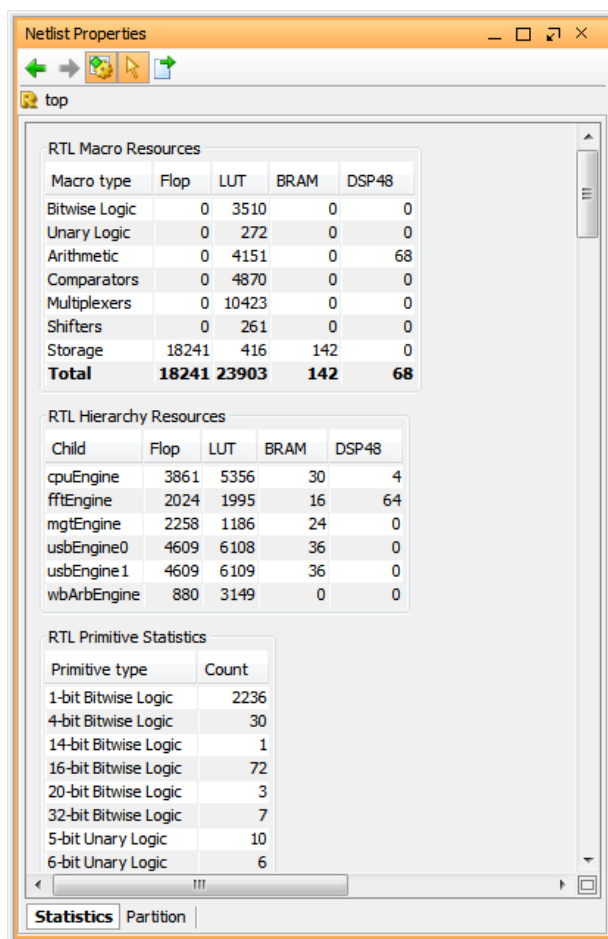


Figure 28: RTL Resource Utilization

The RTL Macro Resources displays in the Netlist Properties view, as shown in Figure 20. If the Netlist Properties view is not displayed, select **Window > Properties** from the main menu.



**Figure 29: RTL Netlist Property Statistics**

4. Scroll down the Netlist Properties view to examine the various categories of statistics.
  - RTL Macro Resources
  - RTL Hierarchy Resources
  - RTL Primitive Statistics
  - RTL Memory Resources
  - Net Boundary Statistics
  - Clock Report
5. In the RTL Netlist view, select other modules and examine the statistics in the Properties view for those selected modules.

Notice that the Netlist Properties view has changed to the Instance Properties view for the selected modules. This is because you are no longer viewing the top-level of the design. The statistics presented in the Properties view are now for the resources required by the module, rather than for the whole design.

## Step 6: Running RTL Design Rule Checks

PlanAhead provides Design Rule Checks (DRC) that can be run on the RTL Design. These include LINT-style RTL checks for power or performance improvement suggestions. There are basic I/O bank and voltage rules for the RTL Design also. After the design is synthesized, a more comprehensive set of logic design, I/O, and clock DRCs is available for the Synthesized Design.

1. From the Flow Navigator, select Report DRC.
2. In the Run DRC dialog box, expand and examine the RTL rules, and click OK.

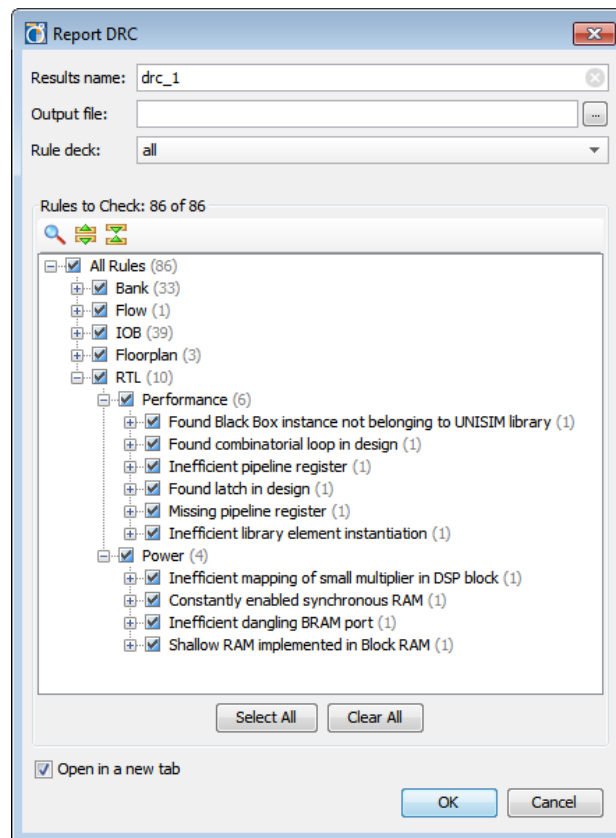
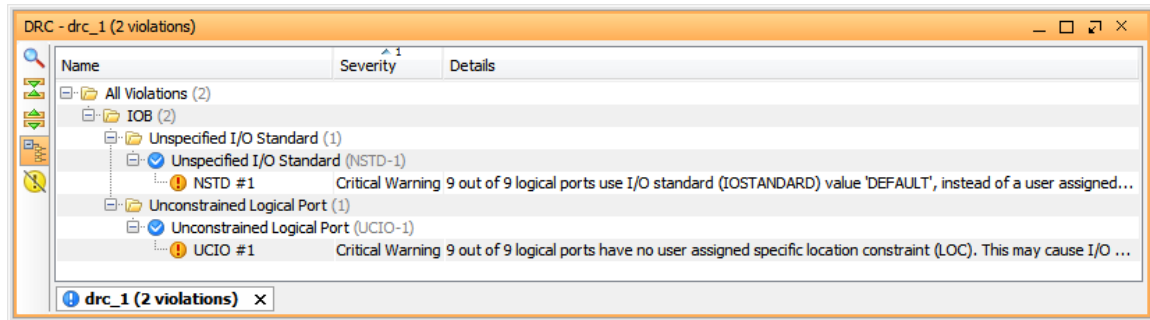


Figure 30: Running DRC

The DRC Results view opens as shown in [Figure 31](#).



**Figure 31: DRC Report**

3. The DRC Results viewer color codes messages as follows:
  - Errors with a red icon
  - Critical Warnings with an orange icon
  - Warnings with a yellow icon
  - Informational messages with a blue icon
4. Select the UCIO #1 latch warning in the violations list.  
 The Violation Properties view displays with information about the violation and links to select the offending logic objects.
5. Close the DRC Results view.
6. Close the RTL Design by clicking the Close **X** icon in the Elaborated Design banner. Click **OK** in the confirmation dialog box if needed.

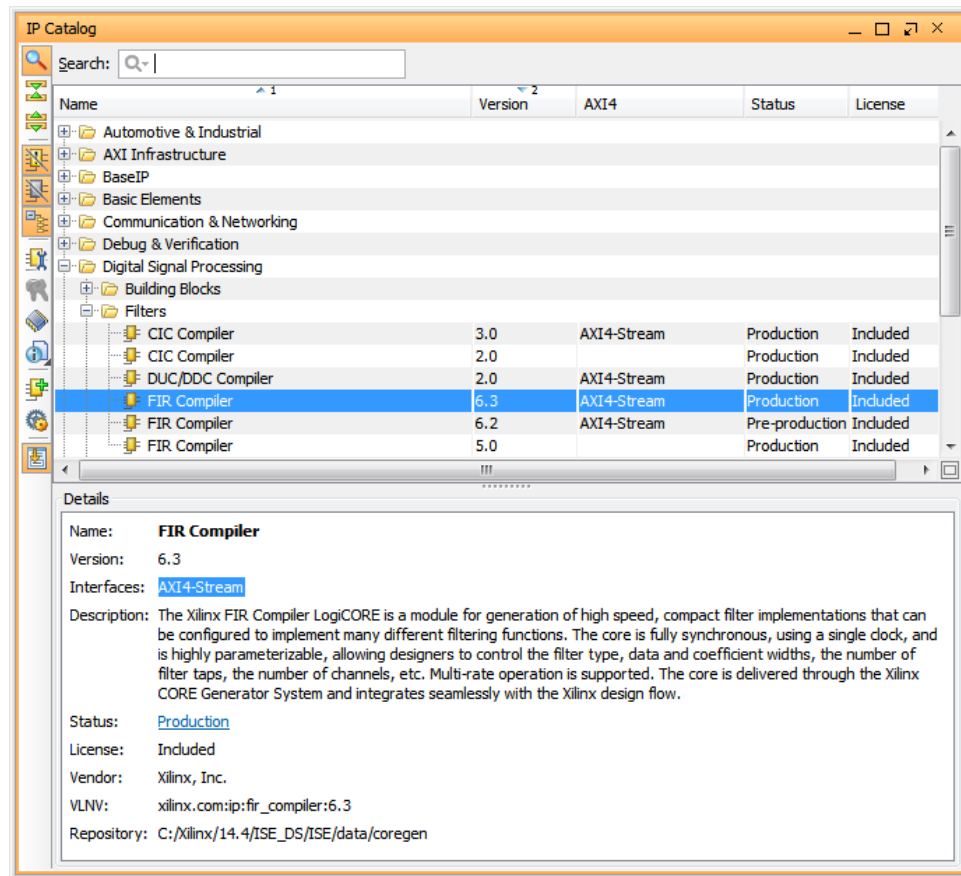
## Step 7: Selecting IP from the Xilinx IP Catalog

The PlanAhead tool is integrated with the CORE Generator™ tool to provide an IP Catalog with search and filtering capabilities. This allows you to easily find the desired IP. You can customize, instantiate and implement the core into your FPGA design from within the PlanAhead tool. You can access the IP Catalog from the Project Manager and Elaborated Design environments.

### Opening and Searching the IP Catalog

1. Select IP Catalog from the Flow Navigator.
2. Expand some of the IP categories.
3. Select any IP and explore the available toolbar buttons and popup menu commands.





**Figure 32: Browsing the IP Catalog**

The Details for a selected IP display at the bottom of the IP Catalog view. By default, only the most recent version of the IP supported for the target device family is displayed.

You can change the list of IP cores listed in the catalog by enabling and disabling commands on the toolbar menu of the IP Catalog view.

- To view all IP in the catalog, toggle the Hide Superseded and Discontinued IPs button and the Hide Incompatible IPs button .
- To view a flattened list of IP, disable the **Group by Category** toolbar button .
- Type **fir** in the Search field at the top of the view.
- Select a FIR Compiler IP.



The details of the IP core are displayed in the Details panel at the bottom of the IP Catalog view.

- From the popup menu, select the **Data Sheet** command.

This will open a PDF view of the datasheet for the selected IP core. Examine the datasheet, and close the PDF viewer when finished.

- Clear the Search field to expand the Catalog list.

## Customizing an IP Core

1. Enable the Group by Category button. 
2. Click the **Collapse All** button  in the IP Catalog toolbar menu.
3. Expand the **Math Functions > Adders & Subtracters** folder.
4. Double-click the **Adder Subtractor** to run the Customize IP command.

This opens the Customize IP dialog box for the selected core, as shown in [Figure 33](#). This is the interface to the CORE Generator™ software in the PlanAhead tool. Different IP will display different dialog boxes for customization.

Examine the different fields and information available in the Customize IP dialog box.

5. In the **B Input Width** field, type **18**.
6. Click **Generate**.

Clicking Generate has a different effect when launched from PlanAhead than when run from CORE Generator standalone.

- In standalone mode, the CORE Generator™ software automatically launches XST to synthesize the IP core.
- When launched from the PlanAhead tool, the synthesis step is not run automatically. This lets you instantiate and configure multiple IP cores into your RTL design before launching synthesis. You can synthesize the IP as part of the overall design, or synthesize it standalone as a block of the design.

When the Adder Subtractor core has been generated, it is added to the design, and listed under the IP Sources tab of the Sources view. In addition, since a new source has been added to the design, the Elaborated Design is shown as being out-of-date and must be reloaded, but first the new IP core must be instantiated into the design.

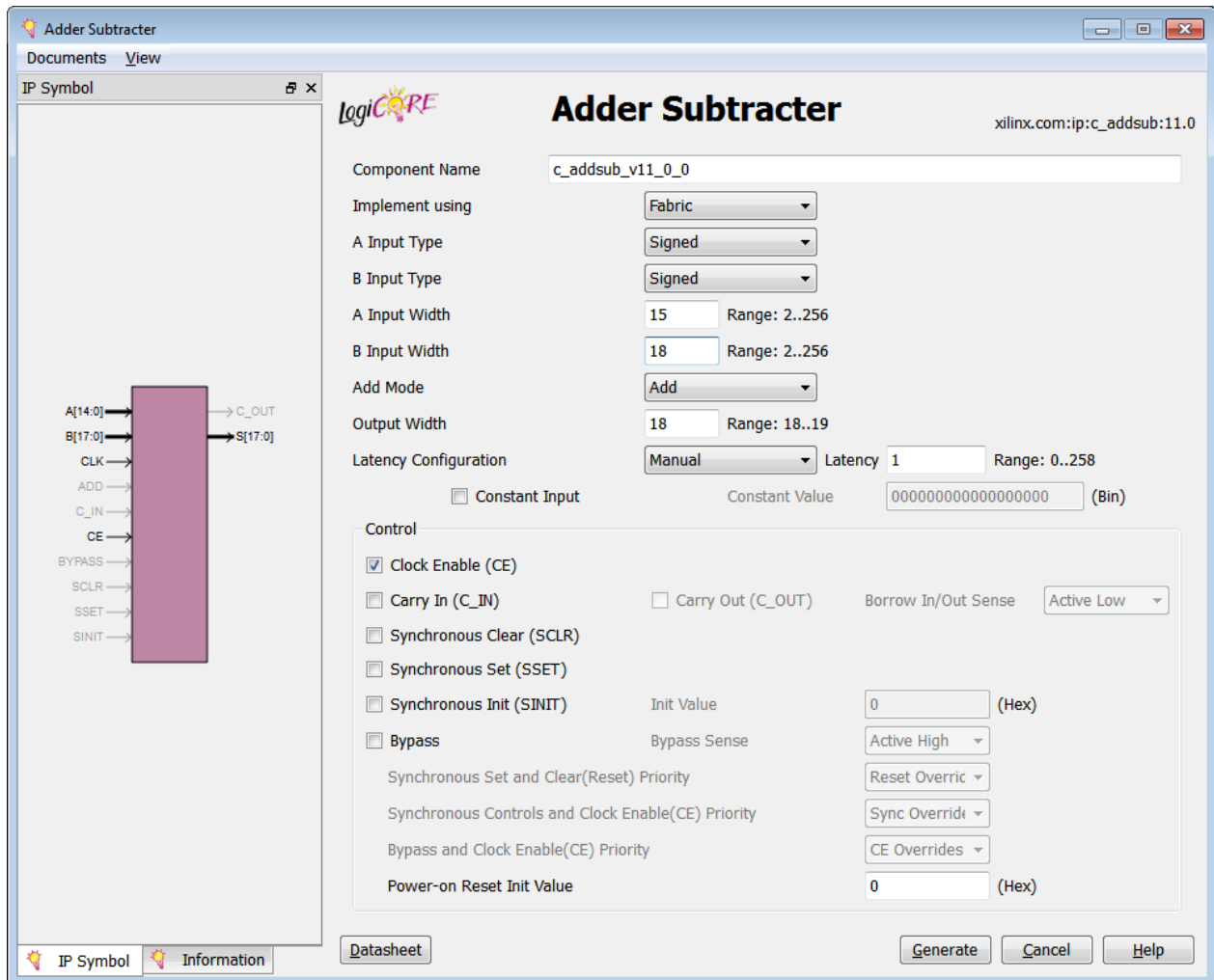


Figure 33: Customizing IP

## Instantiating the IP Core

At this time, the Adder Subtractor core definition is added to the project files, and is available for use in the design. However, the core has not yet been integrated into the design. At this time you will create an instance of the IP in the RTL design.

1. In the upper right of the Elaborated Design banner, **click** the 'X' button to close the design.
2. In the Sources view, select the **IP Sources** tab and click the **Expand All** button.
3. Double-click the **c\_addsub\_v11\_0\_0.vao** file to open the instantiation template in the Text Editor.

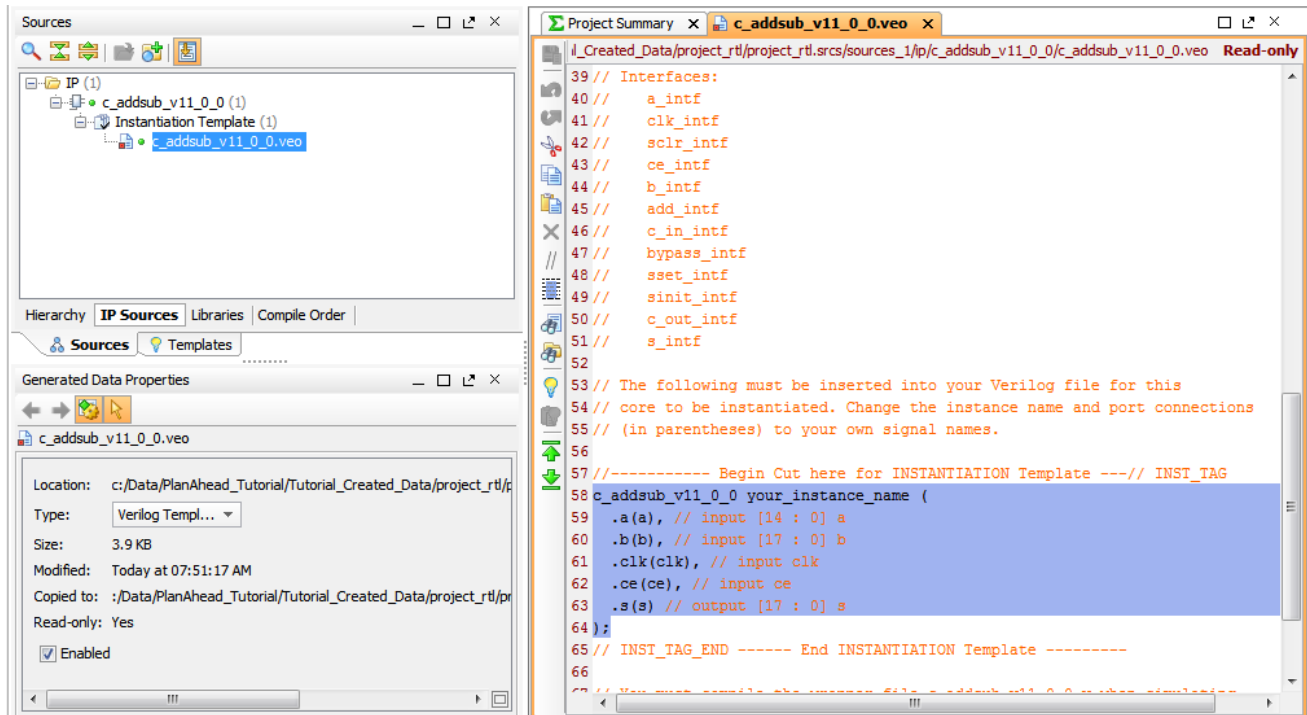


Figure 34: Viewing the Instantiation Template

4. Select the text in the Text Editor, starting just below the line labeled "Begin Cut here..." and ending at the line with "//INST\_TAG\_END".
5. Click the **Copy Text** button.

The copied text can easily be pasted into any RTL source file to instantiate the IP module into your design.

With the IP instantiated into your design, the IP can be synthesized along with the rest of the design, or as a standalone module. You can also generate the simulation files required to simulate the IP core as part of the design.

6. Select the **c\_addsub\_v11\_0\_0** IP core from any tab in the Source view.
7. Right-click to open the popup menu and **select** the **Generate Output Products** command.

The Manage Output Products dialog box opens as is shown in Figure 35. This lets you generate the specific files needed to support the IP core in the overall design.

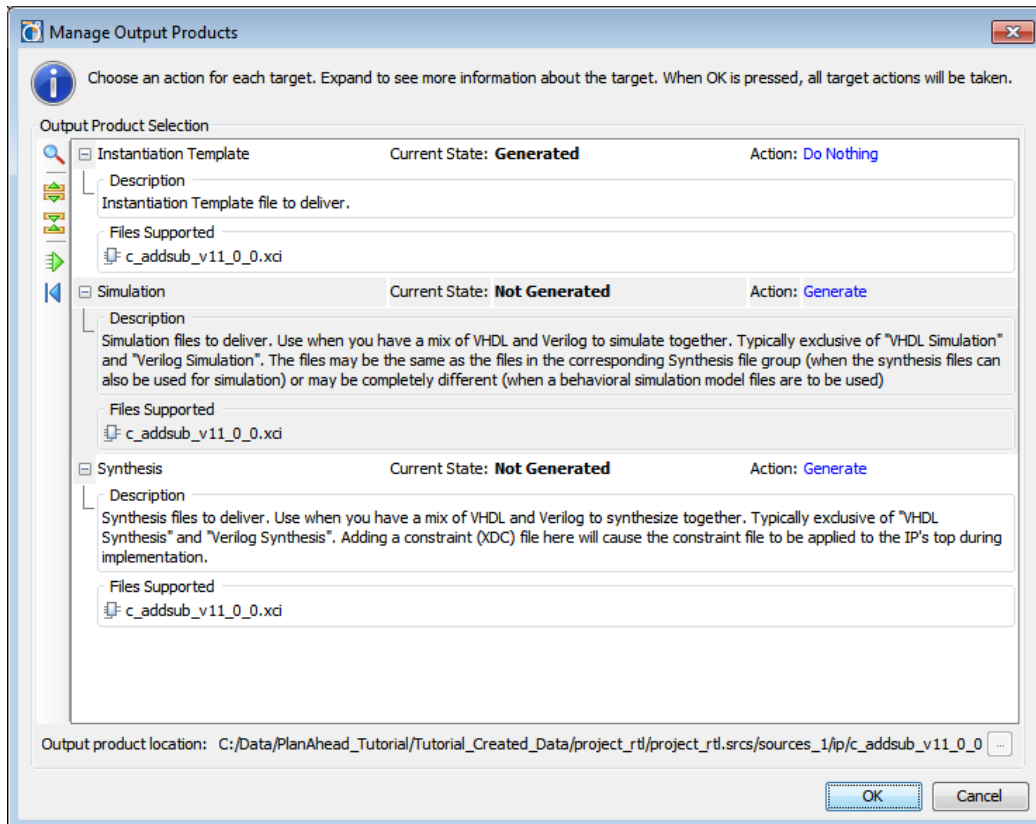


Figure 35: Manage IP Output Files

8. **Examine** the dialog box, and **then close** it.

You can also change the parameters of an IP core by selecting it in the Sources view and using the Re-customize IP command from the popup menu. This will open the Customize IP dialog box as was seen previously in Figure 33.

9. Close the `.v eo` instantiation template file in the Text Editor.
10. Close the IP Catalog.
11. Select **File > Exit**. If prompted, click **No** to save and **OK** to close PlanAhead.

---

## Conclusion

In this tutorial, you:

- Used a small RTL project to examine the PlanAhead RTL development and analysis environment.
- Started by creating an RTL project, explored RTL sources and the RTL editor.
- Ran behavioral simulation, elaborated the RTL design, and explored the analysis capabilities, which included examining the RTL logic hierarchy, RTL schematic exploration, searching for logic types, reviewing RTL resource and running RTL DRCs.
- Examined the Xilinx IP Catalog, and customized, and instantiated a small adder IP core.