

# **Vivado Design Suite**

## **Tutorial:**

### ***Power Analysis and Optimization***

UG997 (v2013.2) June 19, 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 2012 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

The following table shows the revision history for this document.

Date	Version	Revision
06/19/2013	2013.2	Initial Xilinx Release.

# Table of Contents

Revision History.....	2
Vivado Power Analysis and Optimization.....	5
Overview .....	5
Software Requirements.....	5
Hardware Requirements.....	5
Locating Tutorial Design Files.....	5
Lab 1: Power Analysis using Vivado.....	7
Step 1: Creating a New Project .....	7
Step 2: Synthesizing the Design.....	13
Step 3: Report Power Settings.....	14
Step 4: Running Report Power .....	18
Step 5: Power Properties.....	20
Step 6: Editing Power Properties and Refining the Power Analysis .....	21
Step 7: Running Behavioral Simulation with SAIF Output .....	22
Step 8: Incorporating SAIF Data into Power Analysis.....	26
Step 9: Implementing the Design.....	28
Lab 2: Vivado Simulator Timing Simulation and Power Analysis.....	29
Step 1: Open the Implemented Design.....	29
Step 2: Running Report Power in Vectorless Mode .....	33
Step 3: Running Report Power with Vivado Simulator SAIF Data.....	34
Lab 3: ModelSim Timing Simulation and Power Analysis.....	39
Introduction.....	39
Step 1: Set Up to Run Timing Simulation in ModelSim .....	39
Step 2: Running Report Power in Vectorless Mode .....	44
Step 3: Running Report Power with ModelSim SAIF Data.....	45
Lab 4: Power Optimization in Vivado .....	48

Introduction.....	48
Step 1: Run report_power_opt to Examine User/Design Specific Power Optimizations .....	48
Step 2: Set Up Options to Run Power Optimization .....	50
Step 3: Run report_power_opt to Examine Tool Optimizations.....	52
Step 4: Turn Off Optimizations on Specific Signals and Rerun Implementation.....	53
Step 5: Run report_power_opt to Examine Tool Optimizations Again.....	53
Conclusion.....	54

# Vivado Power Analysis and Optimization

---

## Overview

This tutorial introduces the power analysis and optimization use model recommended for use with the Xilinx<sup>®</sup> Vivado<sup>®</sup> Integrated Design Environment (IDE). The tutorial describes the basic steps involved in taking a small example design from RTL to implementation, estimating power through the different stages, and using simulation data to enhance the accuracy of the power analysis. It also describes the steps involved in using the power optimization tools in the design.

---

## Software Requirements

This tutorial requires that the 2013.2 Vivado Design Suite software release or later is installed. For installation instructions and information, see the [Vivado Design Suite User Guide: Release Notes, Installation, and Licensing \(UG973\)](#).

---

## Hardware Requirements

The supported Operating Systems include Red Hat 5.6 Linux 64 and 32 bit, and Windows 7, 64 and 32 bit.

Xilinx recommends a minimum of 2 GB of RAM when using the Vivado tools.

---

## Locating Tutorial Design Files

1. Download the `ug997-vivado-power-analysis-optimization-tutorial.zip` file from the Xilinx website:

<http://www.xilinx.com/cgi-bin/docs/rdoc?v=2013.2;t=vivado+tutorials>

2. Extract the zip file contents into any write-accessible location.

This tutorial refers to the location of the extracted `ug997-vivado-power-analysis-optimization-tutorial.zip` file contents as `<Extract_Dir>`.

**Note:** You will modify the tutorial design data while working through this tutorial. You should use a new copy of the original data each time you start this tutorial.

The following table describes the contents of the zip file containing the tutorial design files, `ug997-vivado-power-analysis-optimization-tutorial.zip`.

Directories/Files	Description
<code>/src</code>	Contains the design HDL, testbench, and constraints file necessary for the functional simulation.
<code>/src/clk_wiz</code> <code>/src/fir_compiler</code> <code>/src/fifo_generator</code>	Contains files associated with IP in the design.
<code>top.v</code>	Top module for the design.
<code>constraints.xdc</code>	Contains clocking and timing constraints for the design.
<code>testbench.v</code>	Testbench for simulating the design.
<code>readme.txt</code>	A readme file about the tutorial design.

# Lab 1: Power Analysis in Vivado

In this lab, you will learn about the Power Analysis and Optimization features in the Vivado IDE. The lab will take you through the steps of project creation and power analysis at the synthesis stage, using the Vivado Report Power feature in vectorless mode. It will also demonstrate using the SAIF file generated from behavioral simulation for Vivado Report Power Analysis.

You will analyze power in the Vivado IDE. Then you will examine some of the major features in the Power Report window and closely examine some power specific Tcl commands. You will also learn to create a SAIF file by simulating the design in the timing simulation stage using both the Vivado simulator and ModelSim.

You will also learn how to invoke Power Optimization after `opt_design` in the Vivado IDE. You will examine the power optimization report and selectively turn power optimizations ON or OFF on specific signals, nets, modules, or hierarchy.

---

## Step 1: Creating a New Project

To create a project, use the New Project wizard to name the project, to add RTL source files and constraints, and to specify the target device.

- On Linux,
  1. Change to the directory where the lab materials are stored:  

```
cd <Extract_Dir>/Vivado_Power_Tutorial
```
  2. Launch the Vivado IDE: `vivado`



**Figure 1: Vivado IDE – Getting Started Page**

- On Windows,
  1. Launch the Vivado Design Suite IDE:
 

**Start > All Programs > Xilinx Design Tools > Vivado 2013.2 > Vivado 2013.2**
  2. As an alternative, click the **Vivado 2013.2** Desktop icon to start the Vivado IDE.  
 The Vivado IDE Getting Started page contains links to open or create projects and to view documentation.
  3. In the Getting Started page, click **Create New Project** to start the New Project wizard.
  4. Click **Next** to continue to the next screen.

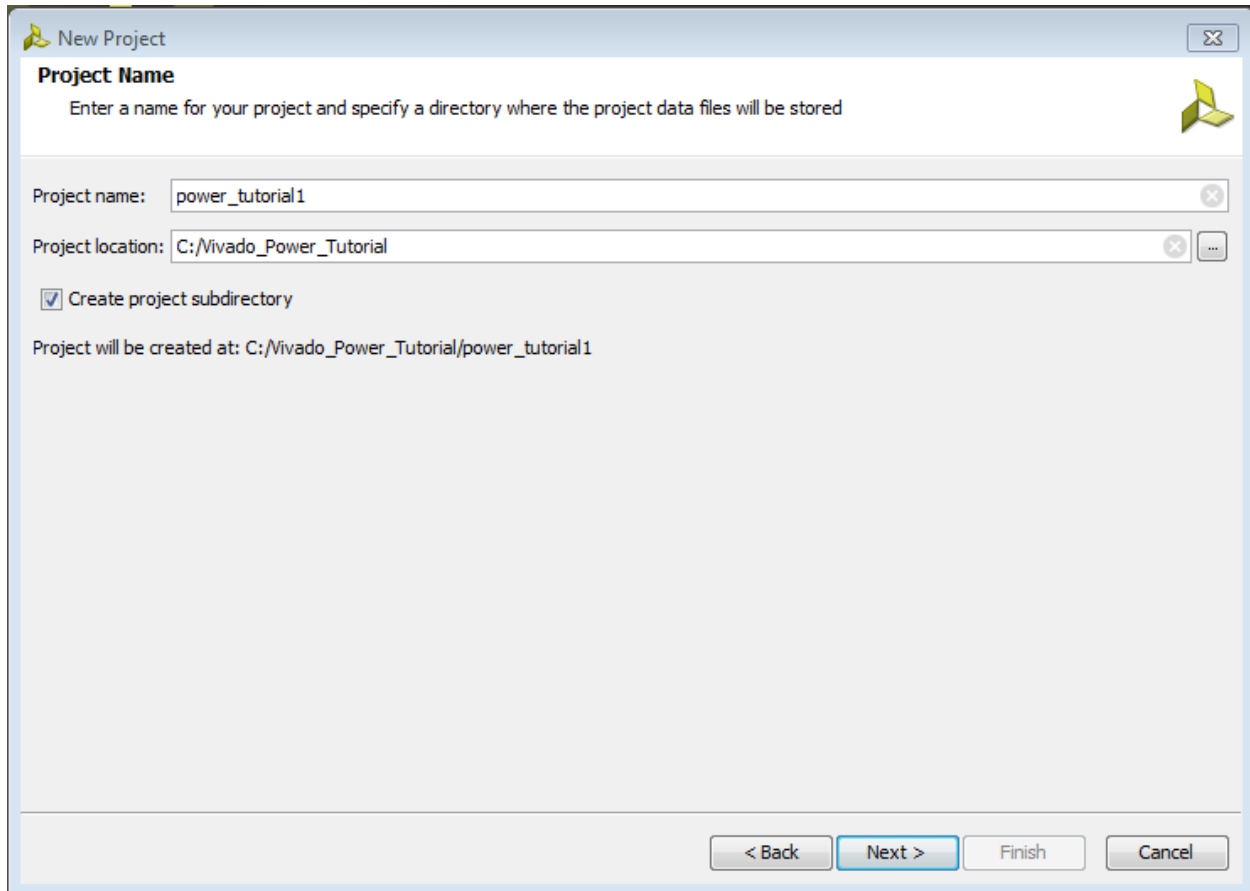


Figure 2: Creating a New Project

5. In the Project Name page, name the new project `power_tutorial1` and provide the project location (`C:\Vivado_Power_Tutorial`). Ensure that **Create Project Subdirectory** is checked and click **Next**.
6. In the Project Type page, specify the **type of project** to create as **RTL Project**, ensure that **Do not specify sources at this time** is unchecked, and click **Next**.
7. In the Add Sources page:
  - a. Set **Target Language** to Verilog.
  - b. Click the **Add Files** button.
  - c. In the Add Source Files dialog box, navigate to the `<Extract_Dir>/src` directory.
  - d. Select the two `.v` source files, and click **OK**.
  - e. In the Add Sources page, change the **HDL Source For** the `testbench.v` file to **Simulation Only**.

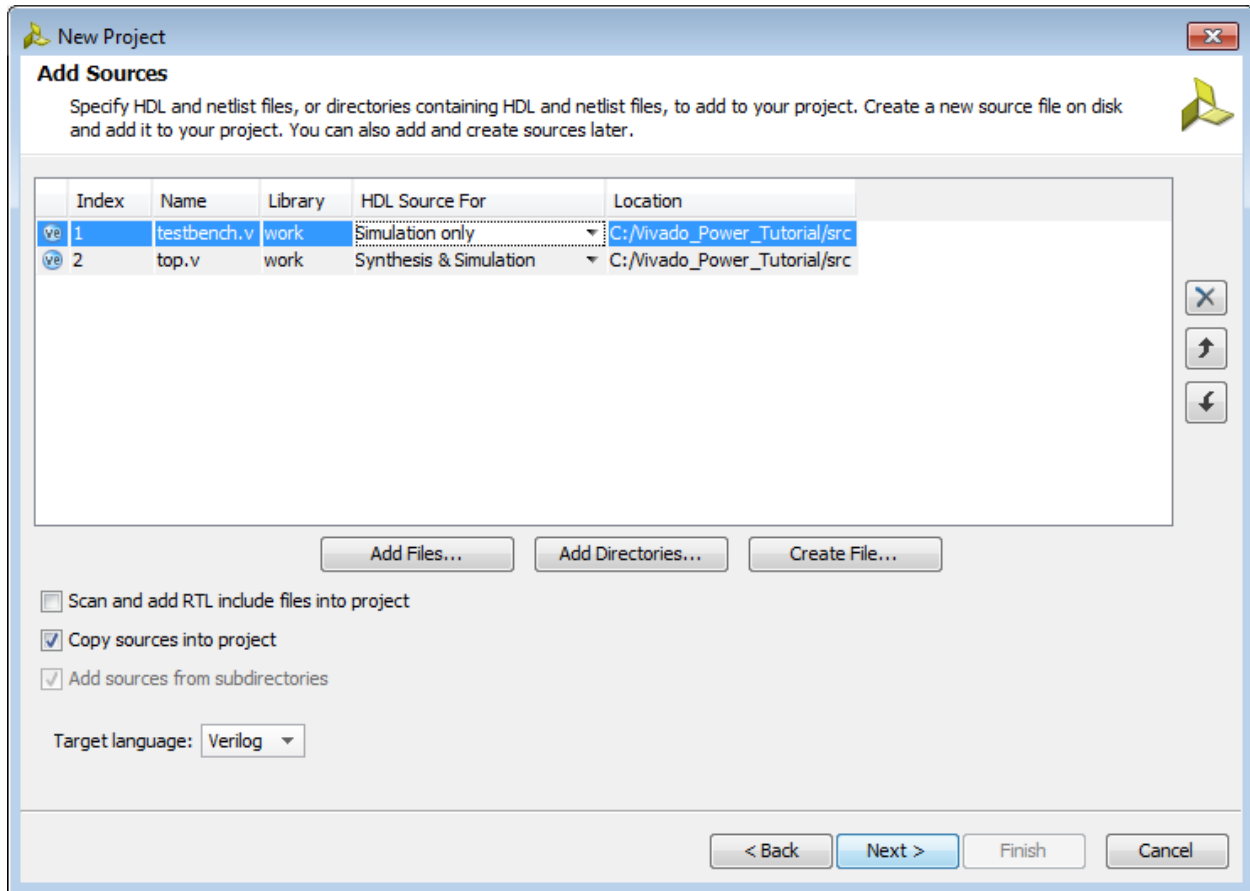


Figure 3: Setting HDL Source Type for testbench.v

- f. Verify that the files are added and **Copy sources into project** is checked. Click **Next**.
8. In the Add Existing IP (optional) dialog box, click **Next**.
9. In the Add Constraints (optional) dialog box, the provided XDC file **constraints.xdc** should automatically appear in the main window.
10. Click **Next** to continue.
11. In the Default Part dialog box, specify the **xc7k325tffg900-2** part. Click **Boards** to specify the board for the target device and select **Kintex-7 KC705 Evaluation Platform**. Then click **Next**.
12. Review the New Project Summary page. Verify that the data appears as expected, per the steps above, and click **Finish**.

**Note:** It might take a moment for the project to initialize in the Vivado IDE.

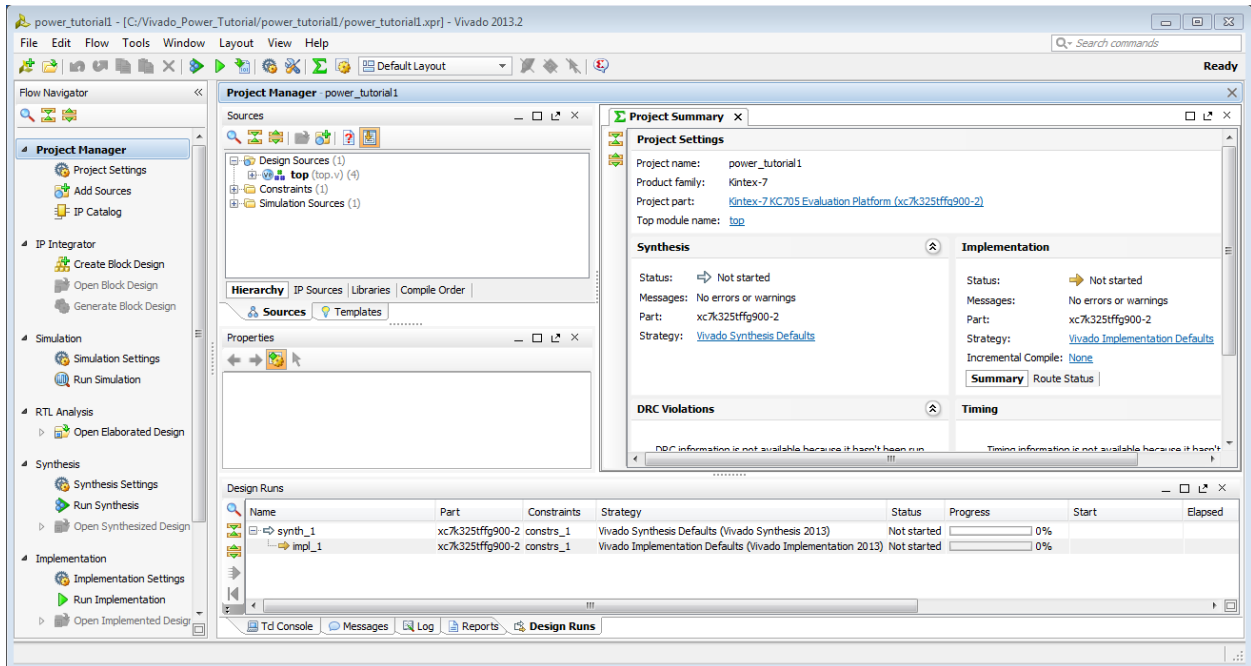


Figure 4: Project in Vivado IDE

13. In the Sources window in the Vivado IDE expand the **top** Design Source.

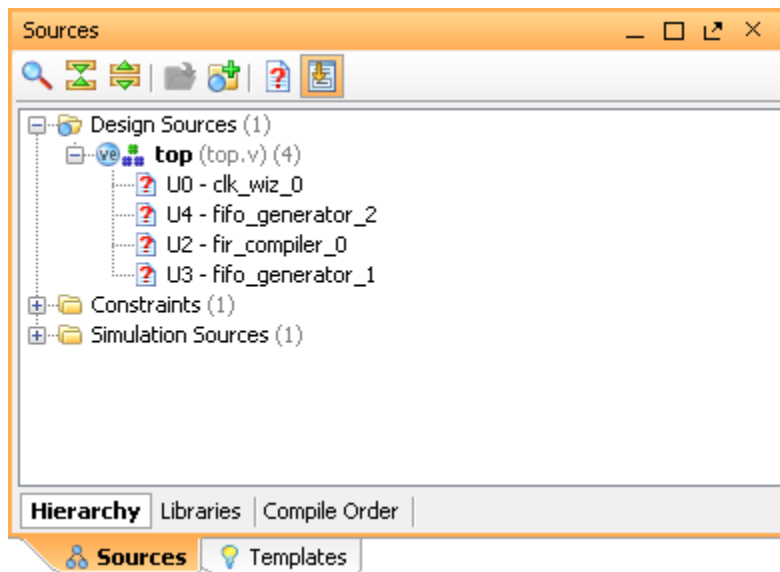


Figure 5: Expanding the top Module in Design Sources Window

Notice the **U2 - fir\_compiler\_0**, **U0 - clk\_wiz\_0**, **U2 - fifo\_generator\_2** and **U3 - fifo\_generator\_1** IP cores have a question mark next to them. This indicates the data sources associated with IP cores in the design have not been specified.

14. In the Sources window, right-click on **U0 - clk\_wiz\_0** and select **Add Sources**.

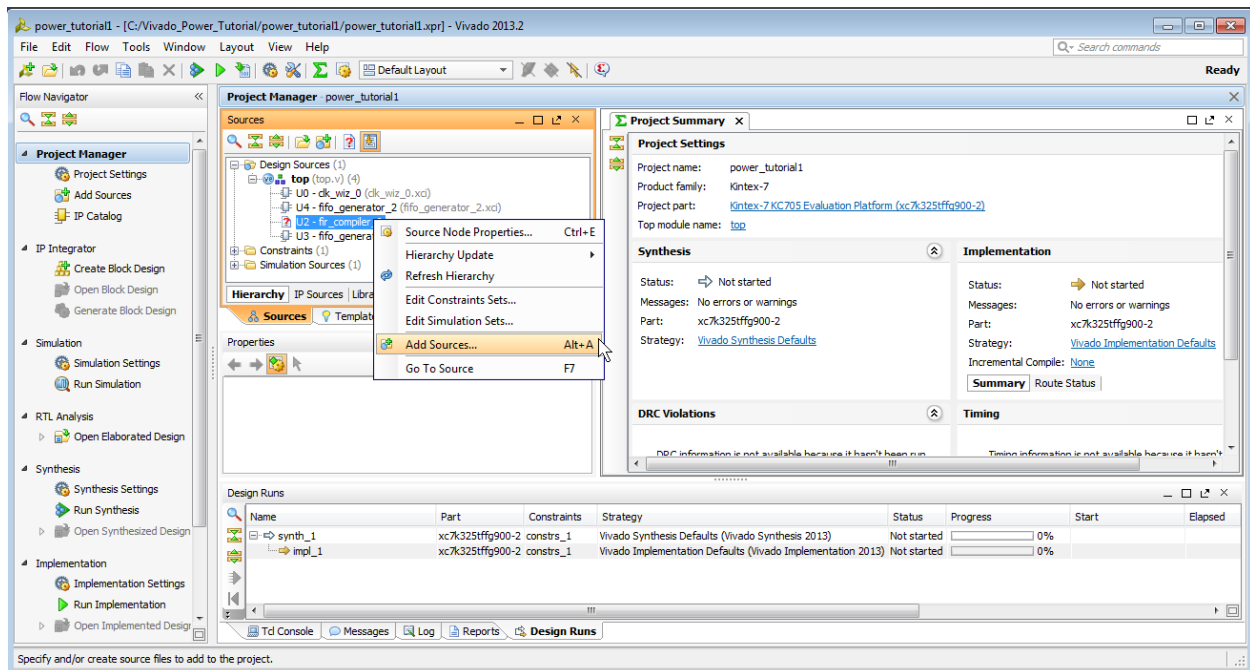


Figure 6: Vivado IDE – Add Sources

15. Choose **Add Existing IP** and click **Next**.
16. In the Add Existing IP page:
  - a. Click the **Add Files** button.
  - b. In the Add Configurable IP dialog box, navigate to the `/src/clk_wiz` directory.
  - c. Select the `clk_wiz_0.xci` source file, and click **OK**.
  - d. Click **Finish** in the Add Existing IP page.
17. Right-click on **U2 - fir\_compiler\_0** and select **Add Sources**.
18. In the Add Sources wizard, choose **Add Existing IP** and click **Next**.
19. In the Add Existing IP page:
  - a. Click the **Add Files** button.
  - b. In the Add Configurable IP dialog box, navigate to the `/src/fir_compiler` directory.
  - c. Select the `fir_compiler_0.xci` source file, and click **OK**.
  - d. Click **Finish** in the Add Existing IP page.
20. Right-click on **U4 - fifo\_generator\_2** and select **Add Sources**
21. Choose **Add Existing IP** and click **Next**.
22. In the Add Existing IP page:

- a. Click the **Add Files** button.
  - b. In the Add Configurable IP dialog box, navigate to the `/src/fifo_generator` directory.
  - c. Select the `fifo_generator_2.xci` source file, and click **OK**.
  - d. Click **Finish** in the Add Existing IP page.
23. Right-click on `U3 - fifo_generator_1` and select **Add Sources**.
24. Choose **Add Existing IP** and click **Next**.
25. In the Add Existing IP page:
- a. Click the **Add Files** button.
  - b. In the Add Configurable IP dialog box, navigate to the `/src/fifo_generator` directory.
  - c. Select the `fifo_generator_1.xci` source file, and click **OK**.
  - d. Click **Finish** in the Add Existing IP page.

Now the design is ready for Synthesis.

---

## Step 2: Synthesizing the Design

1. Click **Run Synthesis** in the Flow Navigator.  
The Synthesis Completed dialog box appears after synthesis has completed on the design.
2. Open the synthesized design by selecting **Open Synthesized Design** in the Synthesis Completed dialog box and clicking **OK**.

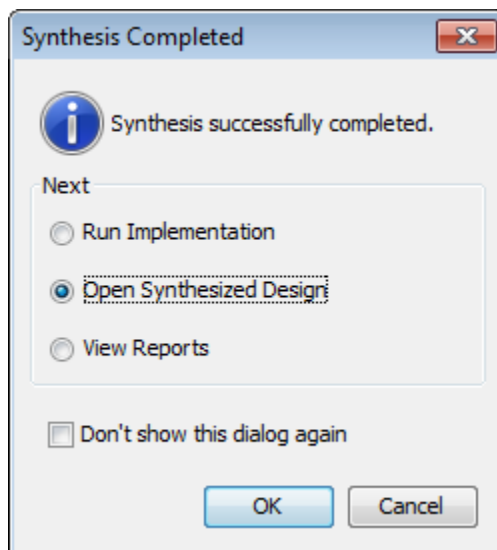


Figure 7: Synthesis Completed Dialog Box

---

## Step 3: Report Power Settings

The Vivado IDE allows you to specify input data to the Report Power tool to enhance the accuracy of the power analysis.

In the Vivado IDE, you can configure thermal, environmental, and power supply options to mimic the board level settings as closely as possible.

1. In the main menu bar, select **Tools > Report > Report Power**.
2. In the Report Power dialog box, specify the **Output text file** as `power_1.pwr`.
3. Specify the **Output XPE file** as `power_1.xpe`. After creating this file when Report Power runs, you can import the file and results into the Xilinx Power Estimator (XPE).

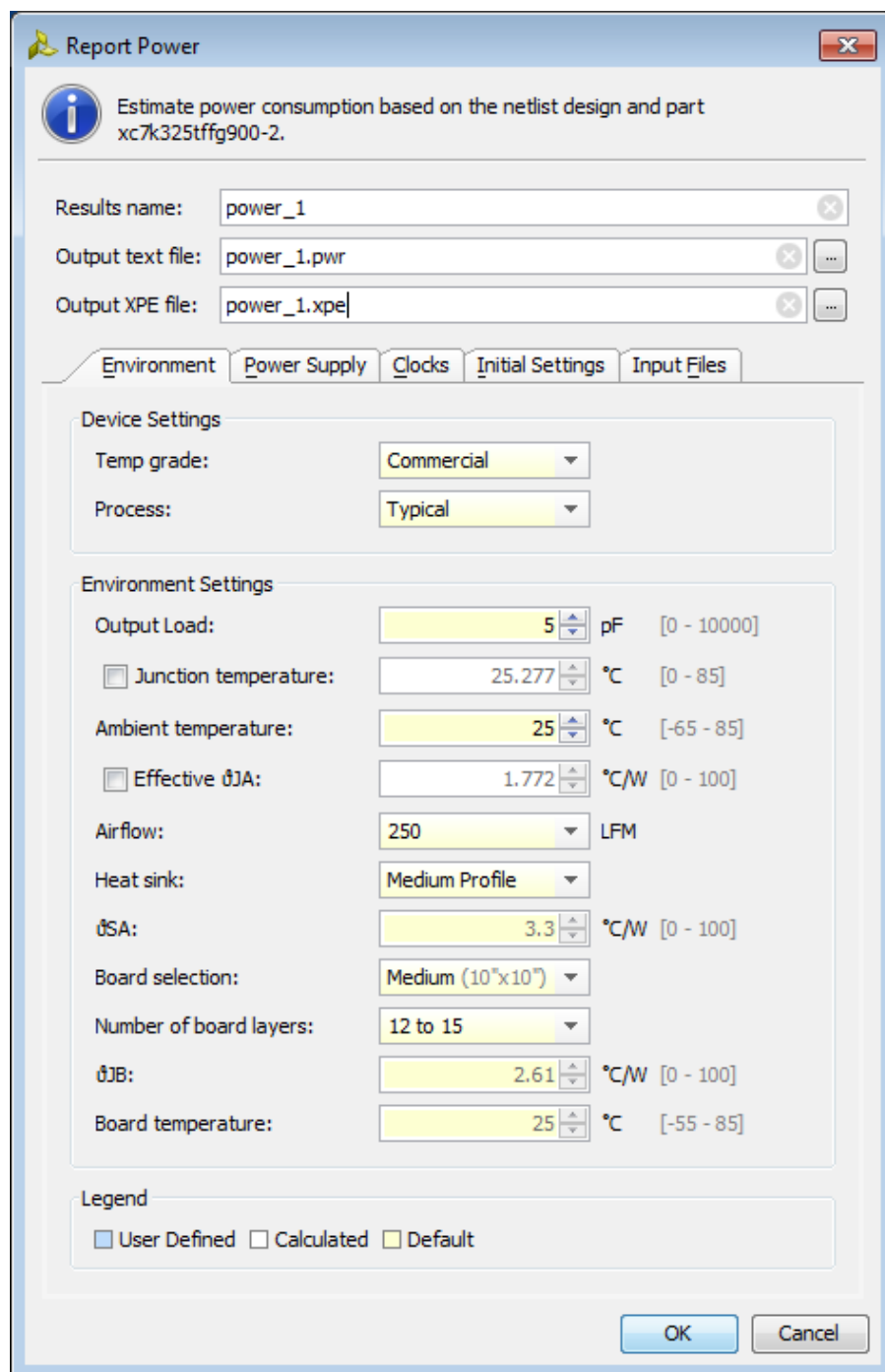


Figure 8: Report Power Dialog Box

4. Examine the **Environment** and **Power Supply** tabs in the Report Power dialog box (Figure 8 and Figure 9).



**IMPORTANT:** By default, Vivado Report Power uses nominal values for voltage supply sources. Voltage is a large factor contributing to both static and dynamic power. For the most accurate analysis, ensure that actual voltage values are entered for each supply. Similarly, ensure temperature and other environmental factors match actual operating conditions.

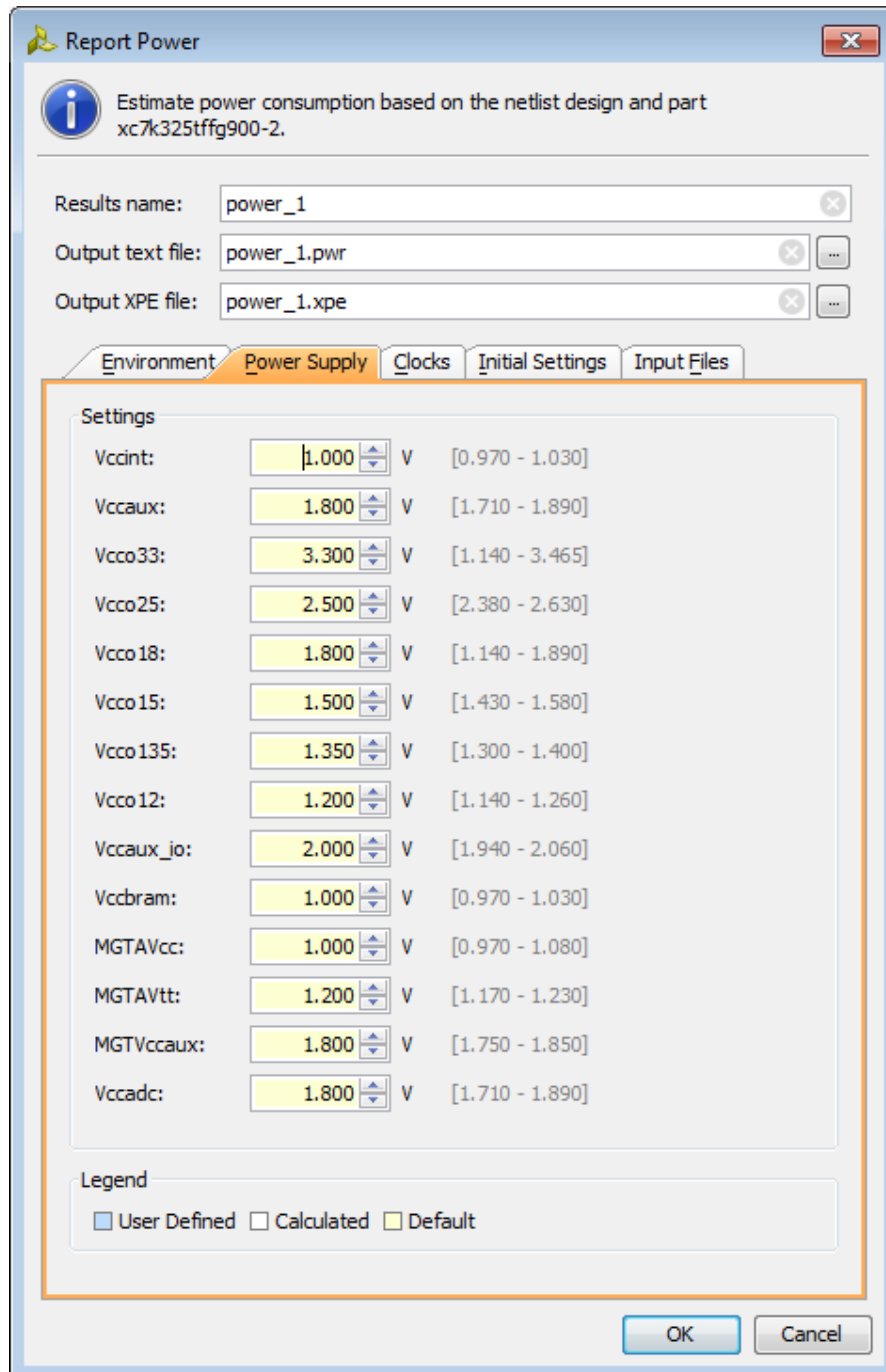


Figure 9: Report Power Dialog Box – Power Supply Tab

5. In the **Clocks** tab click on the **Show Constrained Clocks** link and examine the constrained clocks in the design.



**IMPORTANT:** Ensure that all the relevant clocks in the design are constrained. Vivado Report Power will assume a 0 MHz frequency on clocks that are not constrained, and this could have a significant impact on the power estimation results.

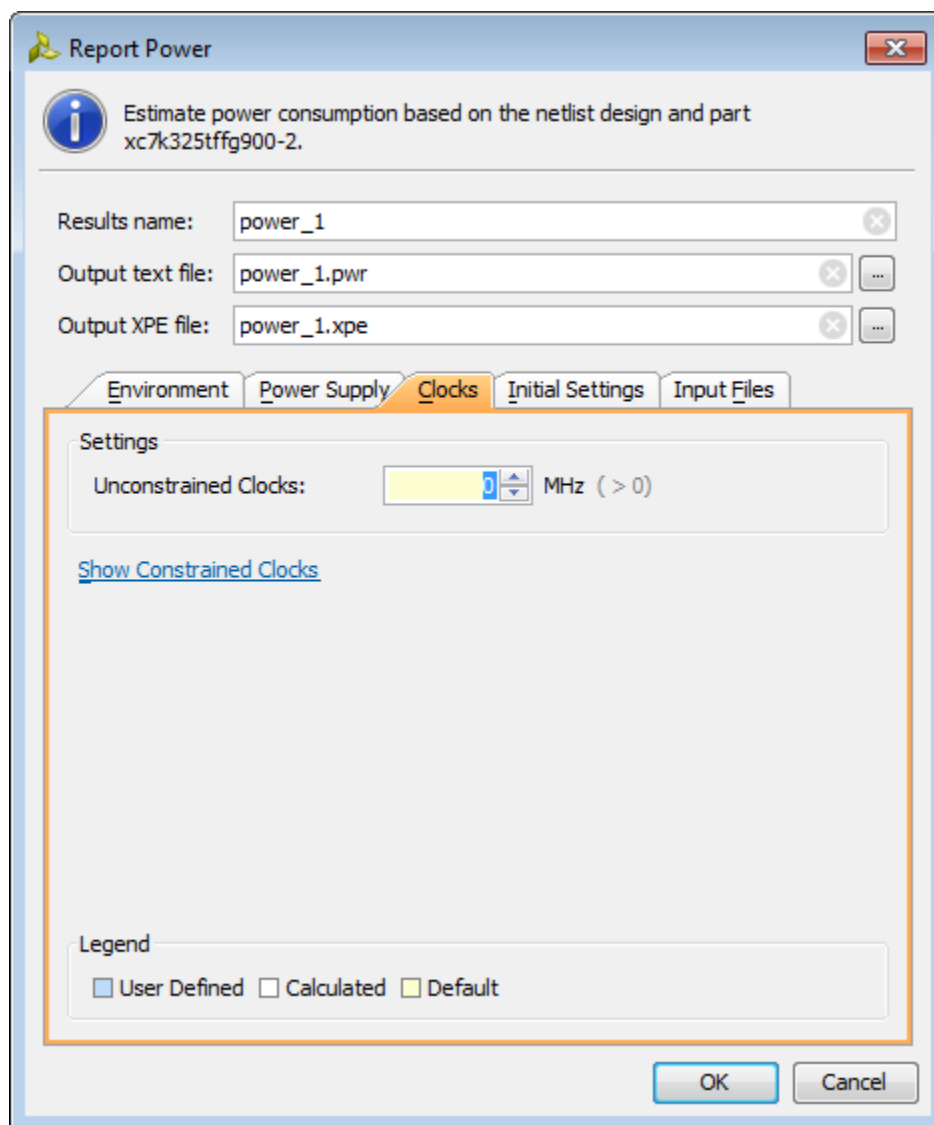


Figure 10: Report Power Dialog Box – Clacks Tab

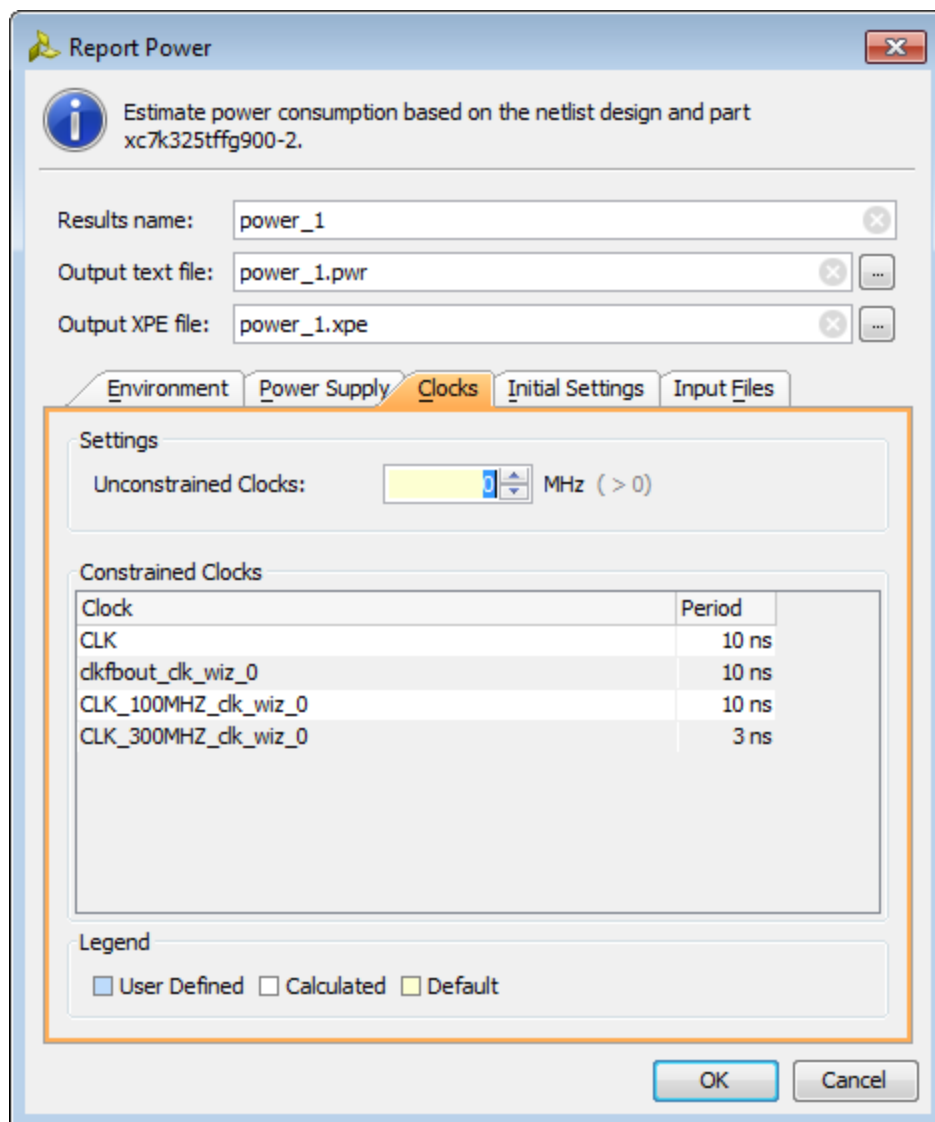


Figure 11: Report Power Dialog Box – Constrained Clocks

---

## Step 4: Running Report Power

1. Click **OK** on the Report Power dialog box.

This runs the `report_power` command.

2. Examine the Power Report **power\_1** generated in the Power tab of the results windows area in the Vivado IDE.

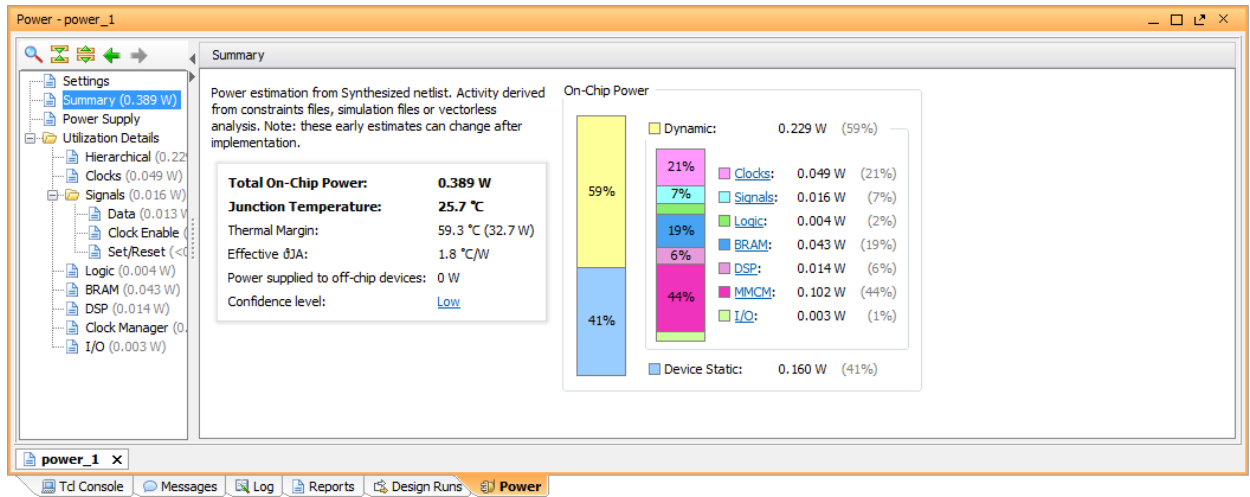


Figure 12: Power Report

3. Examine the power breakdown in the Power Report by block type (**Logic, BRAM, I/O**, etc.).
4. Examine the power supply breakdown in the **Power Supply** view.

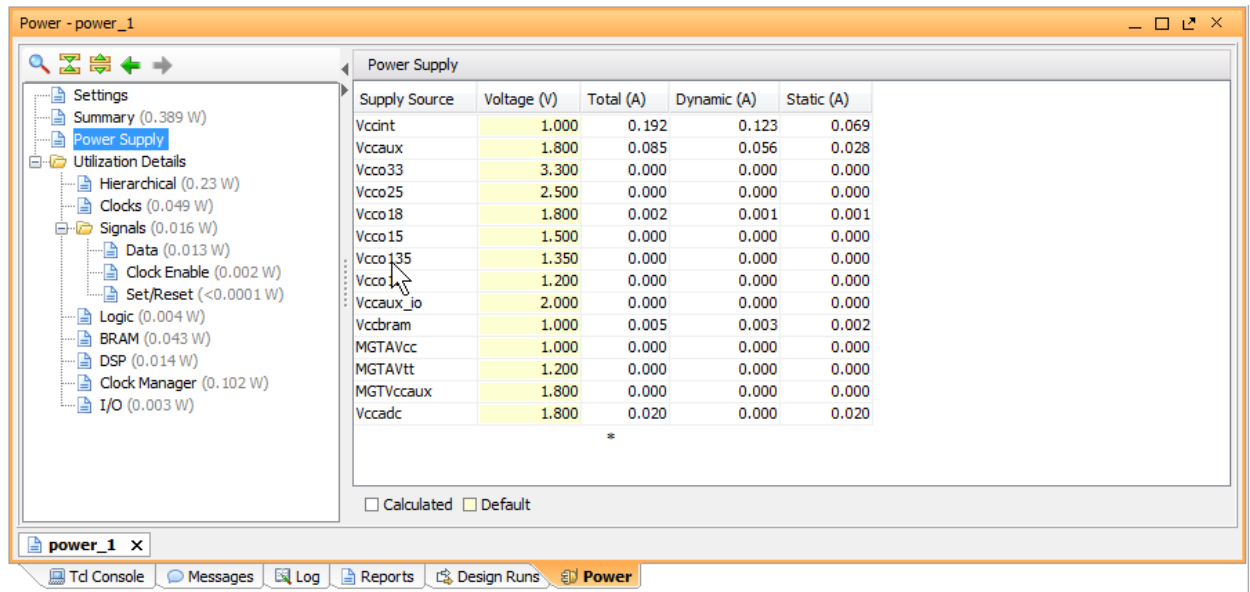


Figure 13: Power Report – Power Supply Display

5. Examine the Hierarchical breakdown of the power in the **Utilization Details > Hierarchical** view.

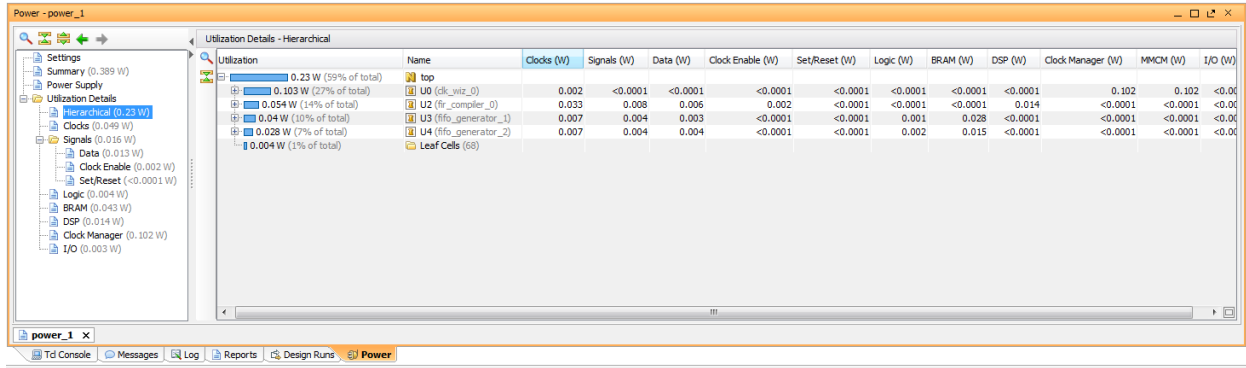


Figure 14: Power Report – Hierarchical Display

6. Examine the **Clocks** view and the various **Signals** views (**Data**, **Clock Enable**, and **Set/Reset**).

## Step 5: Power Properties

1. Note the total power (**Total On-Chip Power**) in the Power Report **Summary** view.
2. Click on the **I/O** tab in the Power Report.
3. Click on the **DATA\_IN** I/O port.

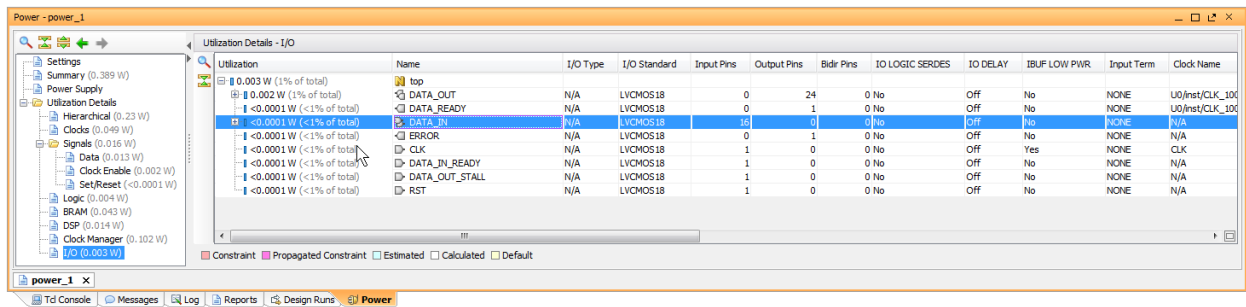


Figure 15: Power Report – I/O Display

4. Note that there is a **Power** tab in the Properties window, displaying I/O Port Bus Properties for the **DATA\_IN** I/O.

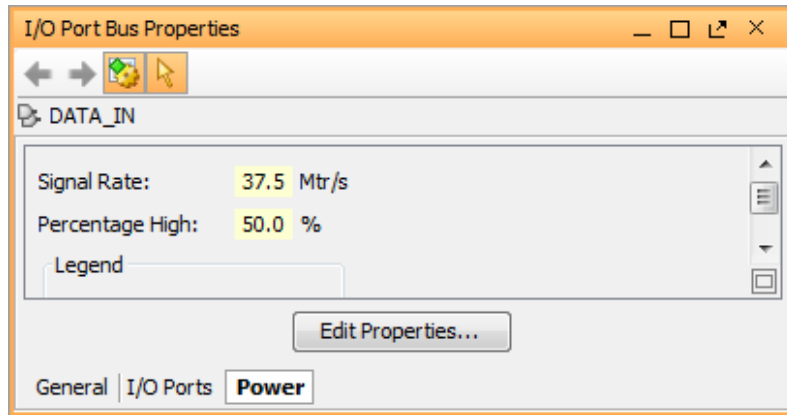


Figure 16: Properties Window – I/O Port Bus Properties

5. Note the **Signal Rate** is 37.5 Mtr/s and **Percentage High** is 50% for the **DATA\_IN** I/O.

## Step 6: Editing Power Properties and Refining the Power Analysis

1. In the I/O Port Bus Properties window, click the **Edit Properties** button.
2. In the Edit Power Properties dialog box, change the **Signal Rate** to 200 and leave the **Percentage High** at 50.0%.

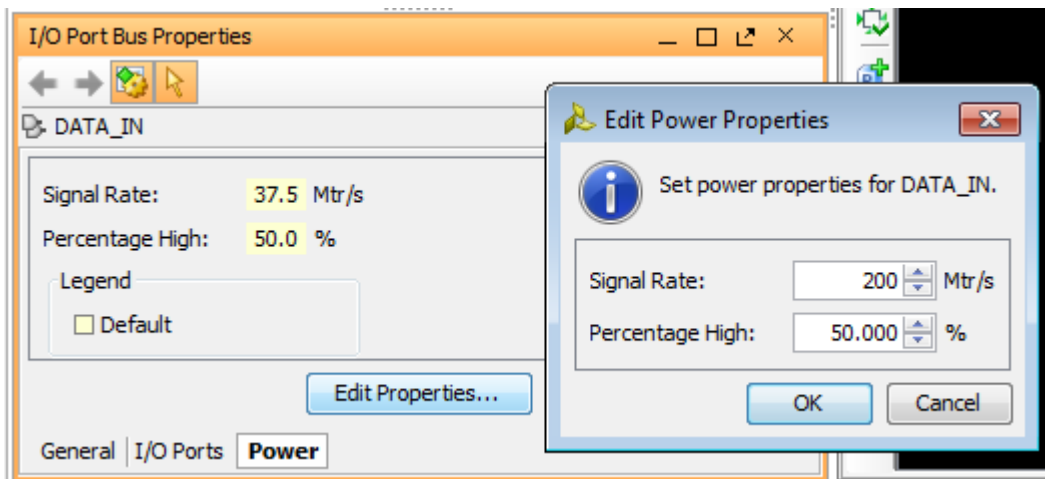


Figure 17: Edit Power Properties Dialog Box

3. Click **OK**.
4. Rerun Report Power (**Tools > Report > Report Power**).

5. Change the **Output text File** and **Output XPE File** to `power_2.pwr` and `power_2.xpe` respectively. Then click **OK**.
6. In the Power window, note the change in total power reported in the **power\_2** report compared to the **power\_1** report). The total power has *increased* due to the change in the **Signal Rate** for the **DATA\_IN** I/O. Note that the **Signal Rate** of the **DATA\_IN** I/O is now color coded as being **User Defined** both in the properties window and in the I/O view of the Power Report.

We recommend using this methodology to double check the signal rates and %high values of high impact I/O ports, control signals (such as resets and clock enables) and high fanout nets. This is an opportunity to guide the Report Power tool to the right estimation scenario.

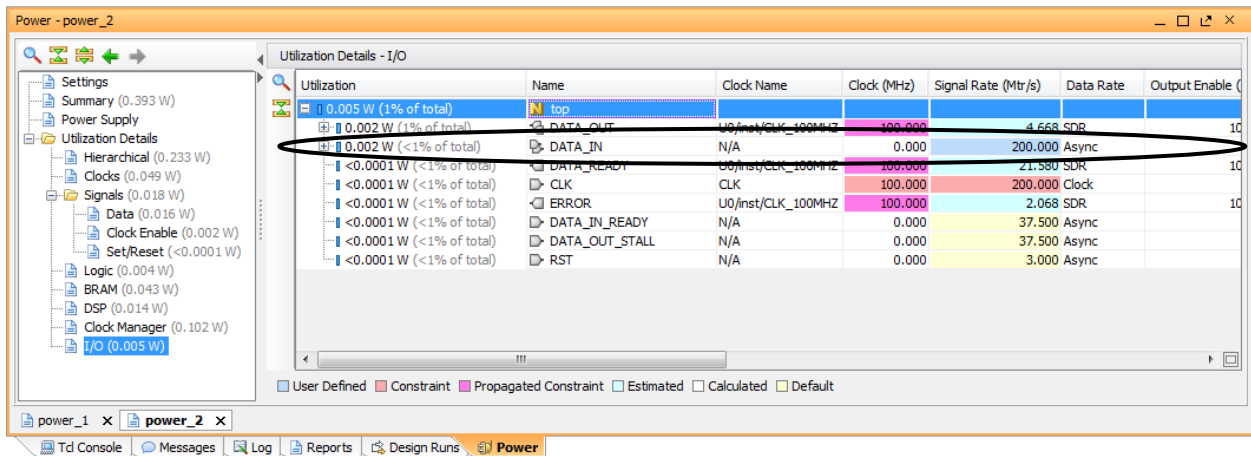


Figure 18: Power Report – User Defined Signal Rate



**TIP:** In Tcl, use the `set_switching_activity` command to change the signal rate and %high of signals and use `report_switching_activity` to query values of signal rate and %high of signals.

```
set_switching_activity -signal_rate 200.0 -static_probability 0.5 -objects [get_ports DATA_IN]

report_switching_activity -signal_rate -static_probability -objects [get_ports DATA_IN]
```

## Step 7: Running Behavioral Simulation with SAIF Output

Now that you have created a Vivado project for the tutorial design, you can set up and launch the Vivado simulator to run behavioral simulation. Simulation will generate a switching activity values file (SAIF) that will enable you to do more accurate power estimation on your design.

1. In the Flow Navigator, click **Simulation Settings** to set the behavioral simulation properties.
2. In the Project Settings dialog box, note that the following Simulation defaults are automatically set:

**Simulation set:** `sim_1`

**Simulation top-module name:** `testbench`

3. In the Compilation tab, ensure that **-debug** (the debug level) is set to **typical**, which is the default value.

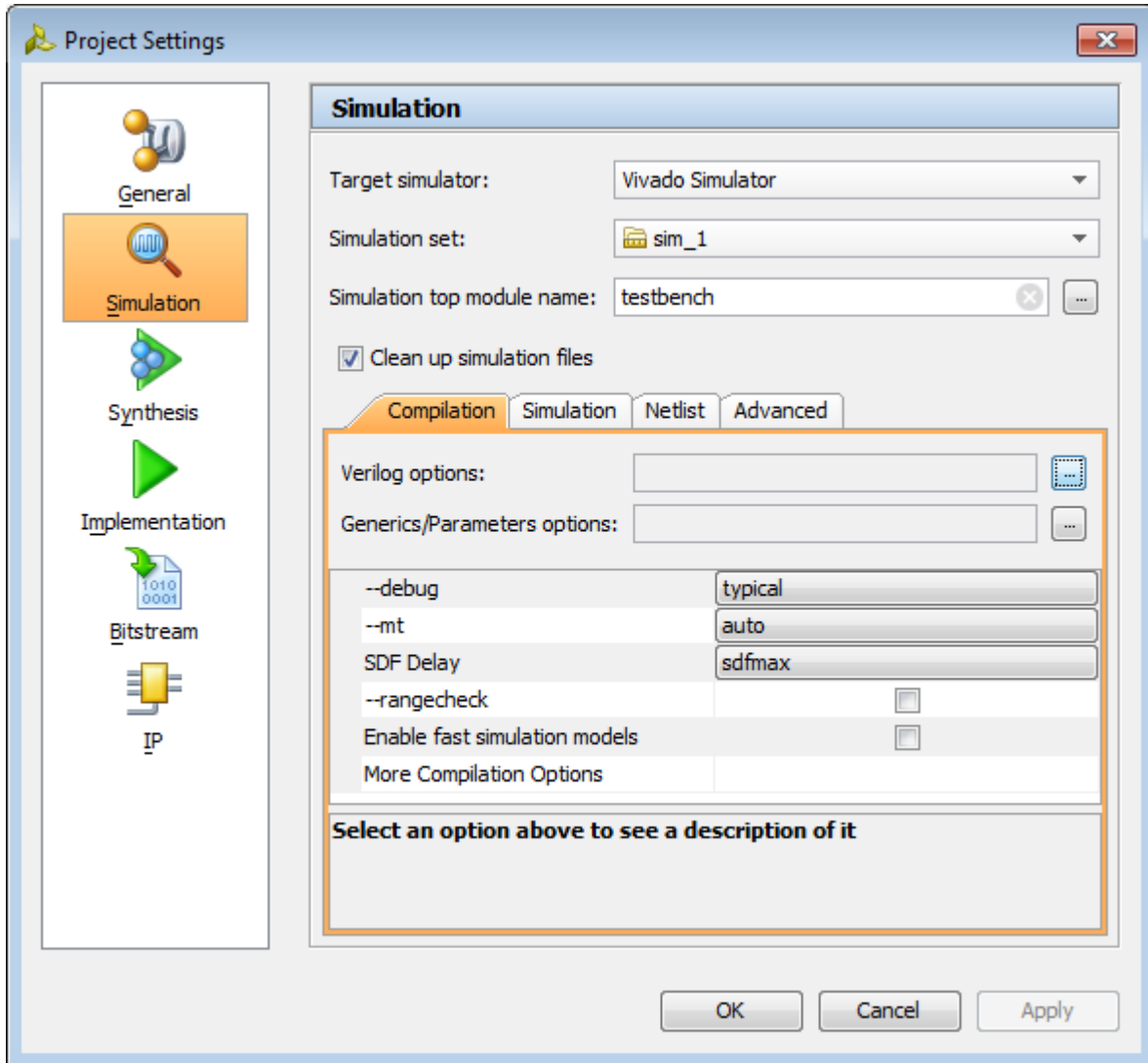


Figure 19: Project Settings – Simulation

4. In the Simulation tab set the **Design Under Test Instance** as `testbench/UUT`.
5. In the Simulation tab set the **SAIF Filename** to `power_tutorial_behav.saif`. Observe that the **Simulation Run Time** is `1000ns`.

6. Click **OK**.

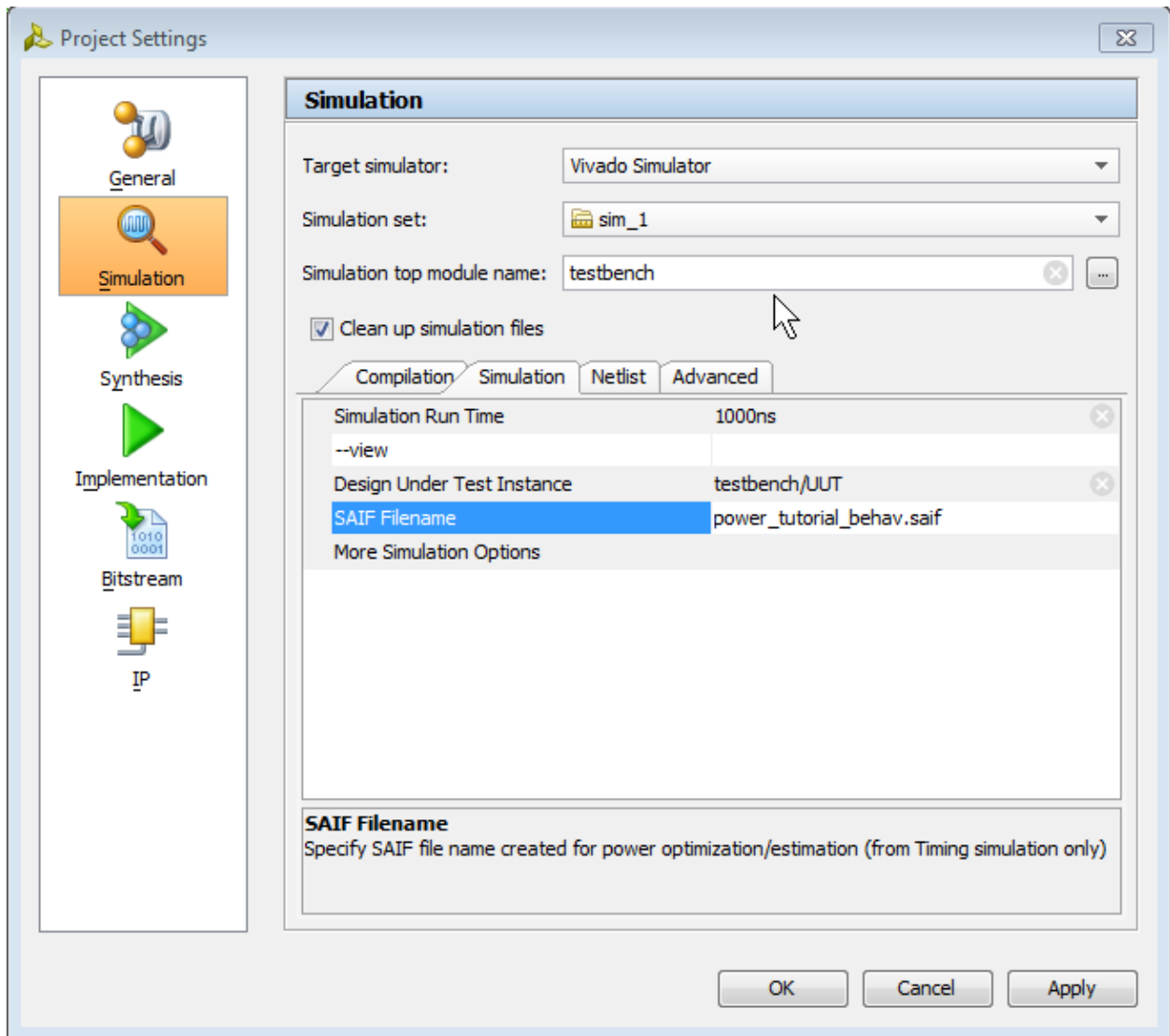
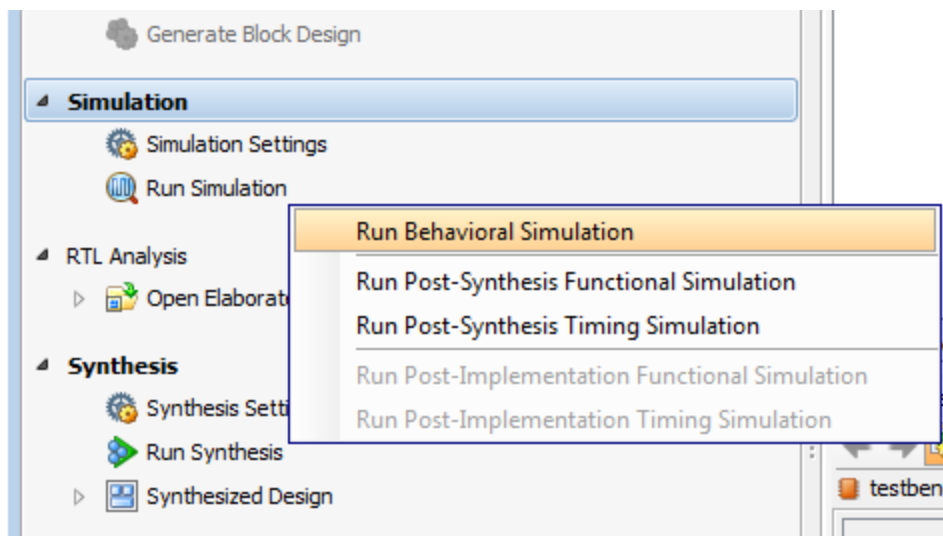


Figure 20: Project Settings – Entering Simulation Settings

With the simulation settings properly configured, you can launch the Vivado simulator to perform a behavioral simulation of the design.

7. In the Flow Navigator, click **Run Simulation > Run Behavioral Simulation**.



**Figure 21: Running Behavioral Simulation from Flow Navigator**

When you launch the Run Behavioral Simulation command, the Vivado tool runs `xelab` in the background to elaborate and compile the design into a simulation snapshot, which the Vivado simulator can run. When that process is complete, the Vivado tool launches `xsim` to run the simulation.

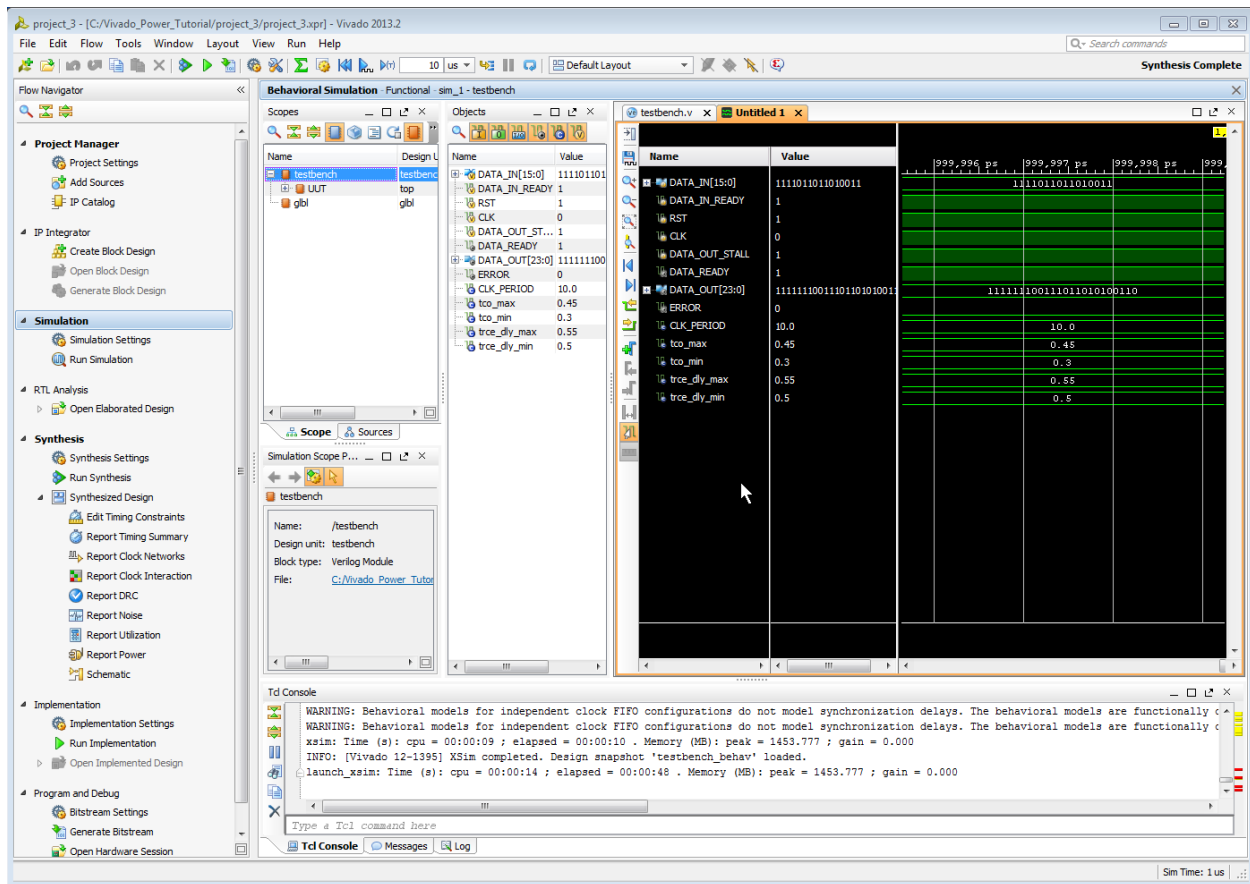


Figure 22: Behavioral Simulation Window in the Vivado IDE

## Step 8: Incorporating SAIF Data into Power Analysis

The SAIF output file requested in the simulation run is generated in the project directory. To further guide the power analysis algorithm we use this SAIF file – a Switching Activity Interchange Format file.

1. Ensure the SAIF file requested is generated. Check to see that the SAIF file requested in the simulation settings prior to running simulation appears in this directory:

```
/ <project_directory> / <project_name> / <project_name> .sim / sim_1 /
behav / power_tutorial_behav.saif
```

2. In the Flow Navigator window, click on **Synthesized Design**.
3. In the main menu bar, select **Tools > Report > Report Power**.
4. In the Report Power dialog box, specify the SAIF file location in the **Input Files** tab.

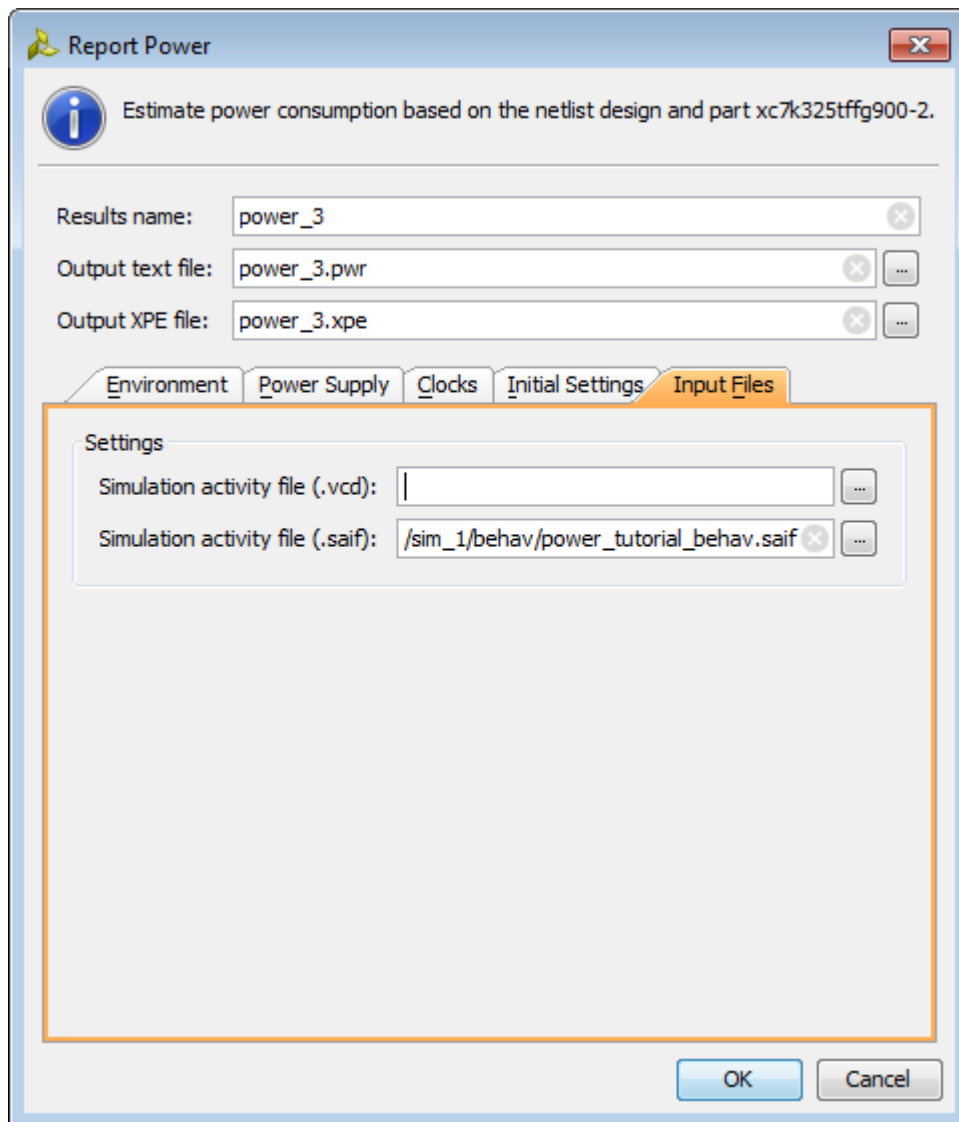


Figure 23: Specifying SAIF File Location

5. Click **OK** in the Report Power dialog box.

The report\_power command runs, and the Power Report **power\_3** is generated in the Power tab of the results windows area.

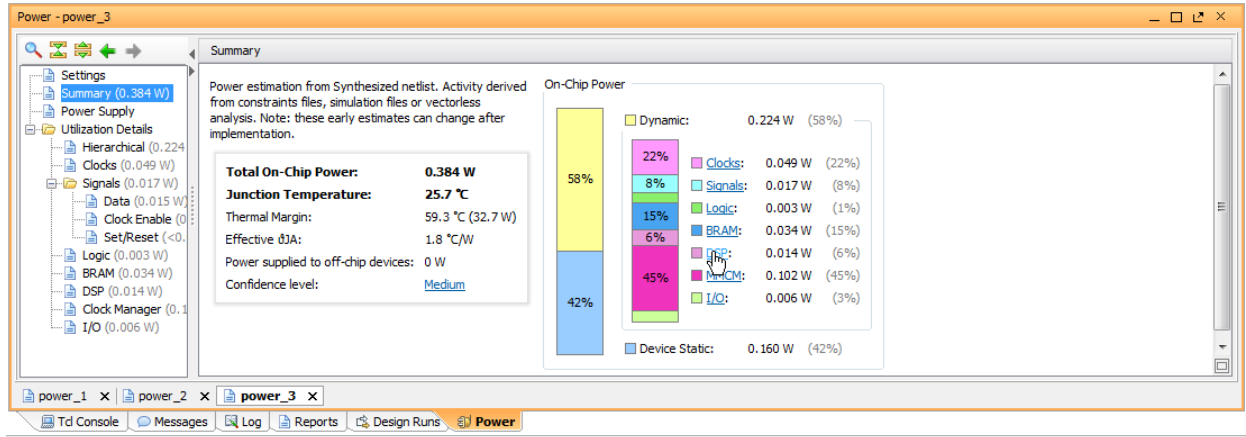


Figure 24: Power Report – Summary

- Go to the **I/O** view in the Power Report. Note that all the I/O port activity data has been set from simulation data we specified.

The data is color coded to indicate activity rates read from the **Simulation** output file.

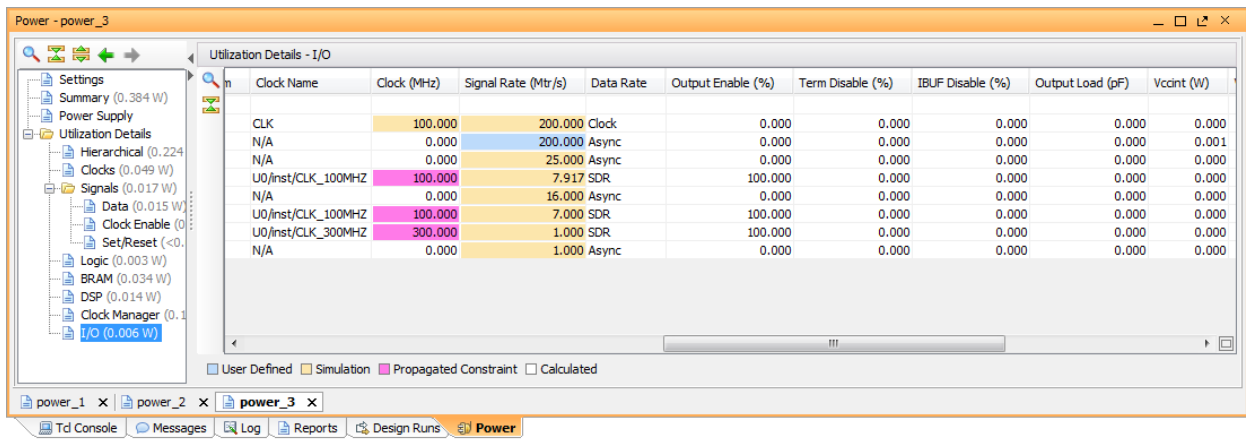


Figure 25: Power Report – I/O Display

- Note the difference in total power numbers (**Total On-Chip Power** in the **Summary** view) between a pure vectorless run in the **power\_1** results versus with the behavioral simulation data in the **power\_3** results.

## Step 9: Implementing the Design

- In the Flow Navigator, click **Run Implementation**.
- When prompted to **Save project before launching implementation** click **Don't Save**.

# Lab 2: Vivado Simulator Timing Simulation and Power Analysis

In this lab, you will learn about generating a SAIF file after running a timing level simulation using the Vivado Simulator. The lab will take you through the steps for SAIF file creation, running timing simulation, and estimating power using the SAIF data.

---

## Step 1: Open the Implemented Design

1. In the Implementation Complete dialog box, select **Open Implemented Design** and click **OK** to open the implemented design.

Now you are ready to set up and launch the Vivado simulator to run post implementation timing simulation. You will set the timing simulation properties in the Vivado IDE, then run the timing simulation.

2. In the Flow Navigator, click **Simulation Settings** to set the timing simulation properties.
3. In the Project Settings dialog box, note that the following defaults are automatically set:

**Simulation set:** `sim_1`

**Simulation top-module name:** `testbench`

4. In the Compilation tab, ensure that **-debug** (the debug level) is set to **typical**, which is the default value.

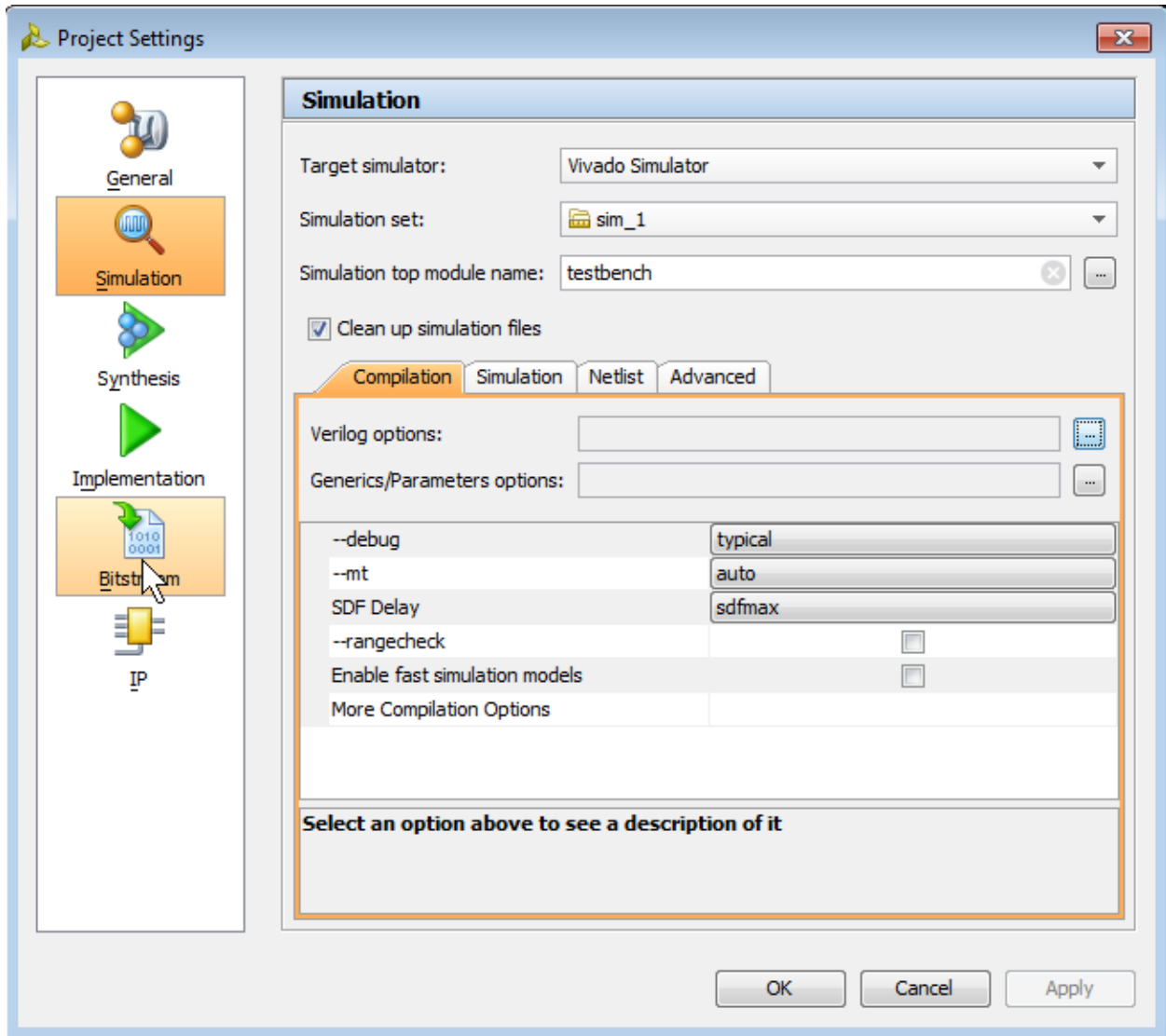


Figure 26: Simulation Settings – Compilation Tab

5. In the Simulation tab set the **Design Under Test Instance** to testbench/UUT and the **SAIF Filename** to power\_tutorial\_timing\_xsim.saif.
6. Observe that the **Simulation Run Time** is 1000ns.
7. Click **OK**.

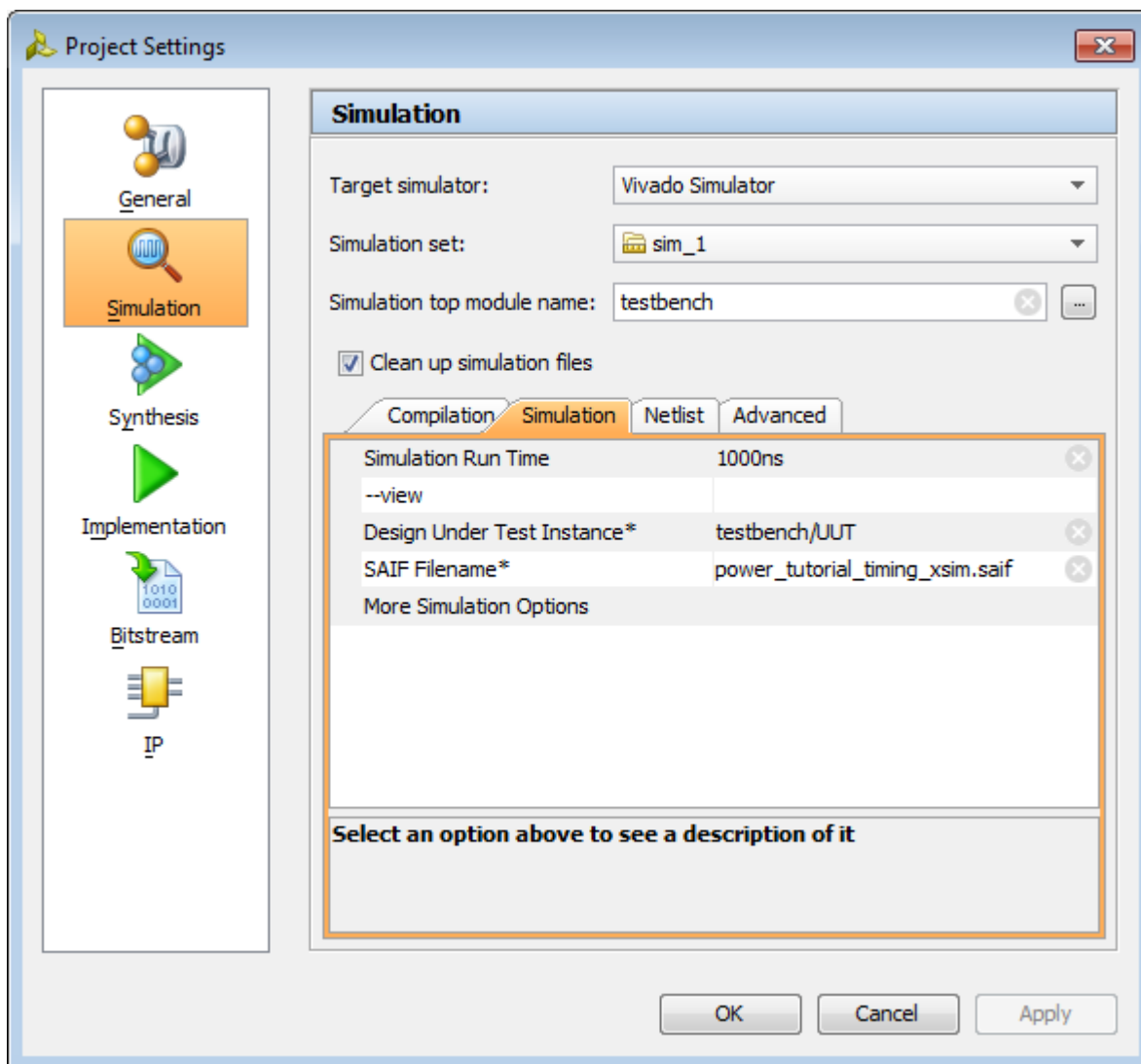
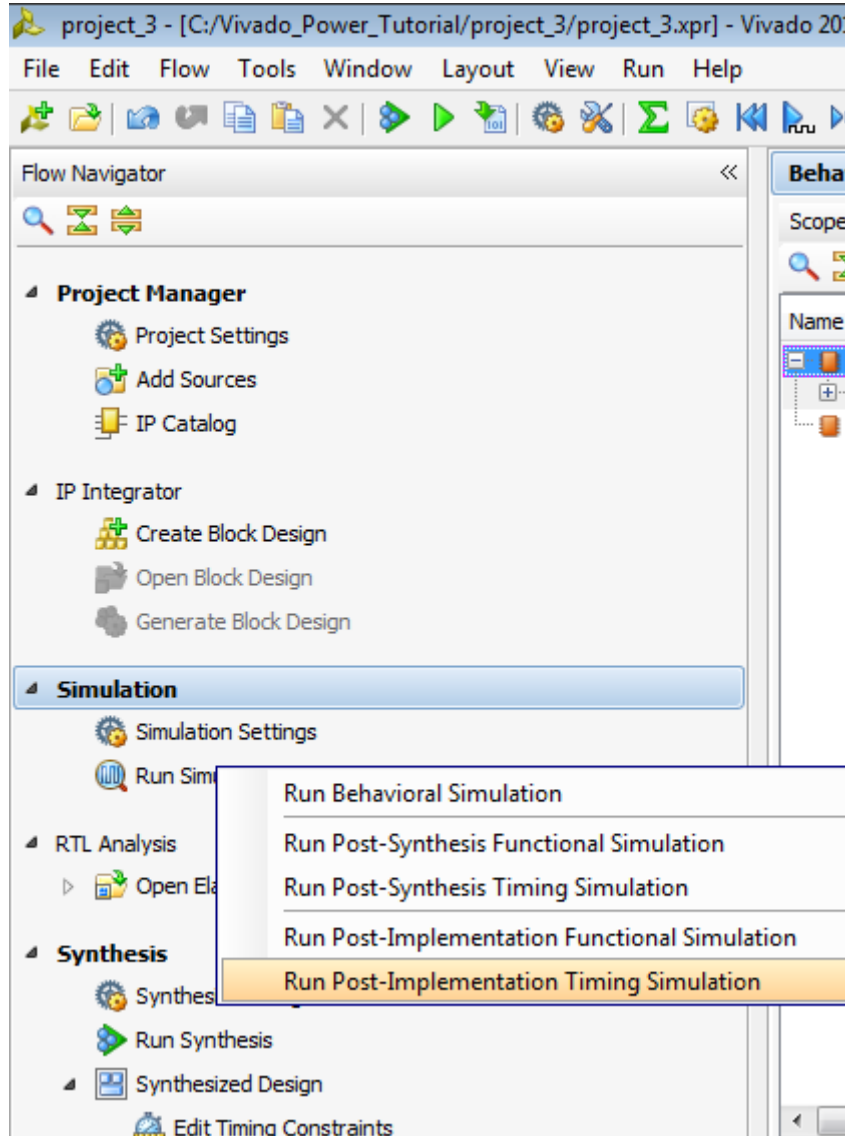


Figure 27: Timing Simulation Settings – Simulation Tab

With the simulation settings properly configured, you can launch the Vivado simulator to perform a timing simulation of the design.

8. In the Flow Navigator, click **Run Simulation > Run Post-Implementation Timing Simulation**.



**Figure 28: Running Post-Implementation Timing Simulation**

9. After the Vivado simulator has finished simulating the design, ensure that the SAIF file requested has been generated. Check to see that the SAIF file requested in the simulation settings prior to running simulation appears in this directory:

```
/<project_directory>/<project_name>/<project_name>.sim/
sim_1/impl/timing/power_tutorial_timing_xsim.saif
```

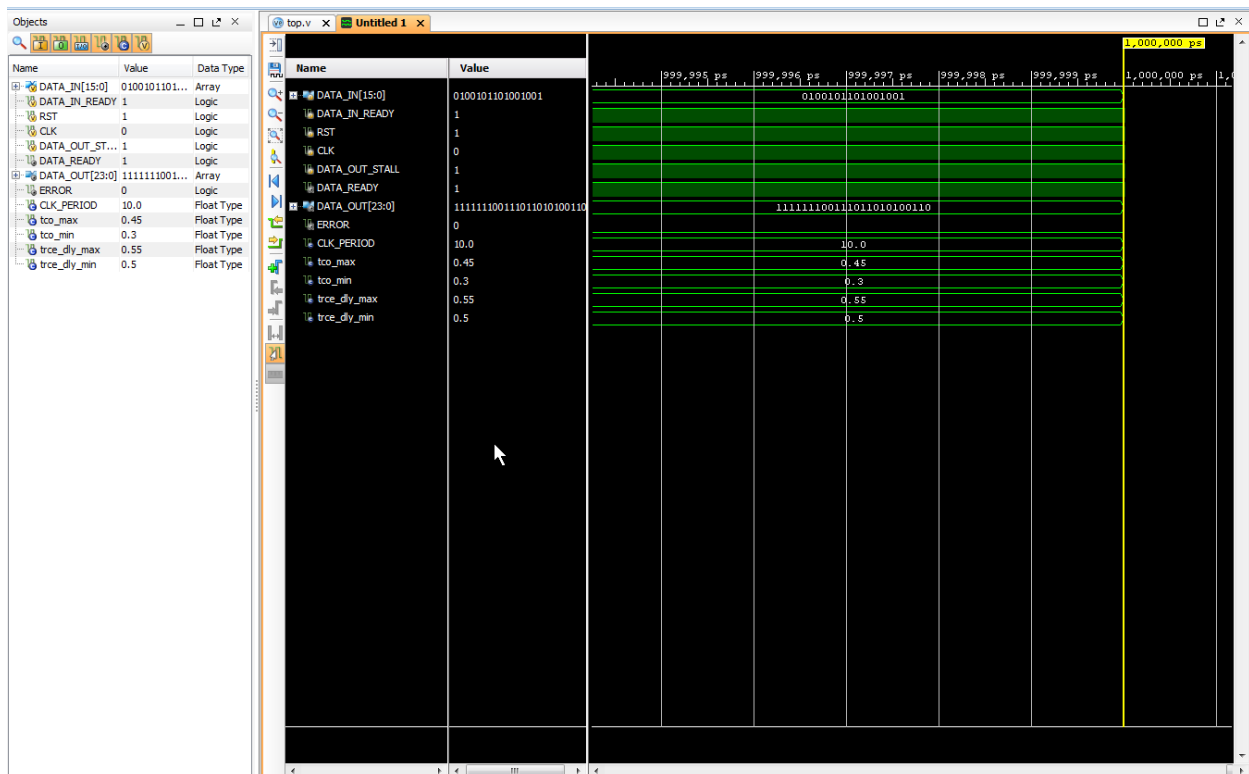


Figure 29: Post-Implementation Timing Simulation using the Vivado Simulator

## Step 2: Running Report Power in Vectorless Mode

1. In the Flow Navigator, select **Implemented Design > Report Power** to open the Report Power dialog box.

You can also select **Tools > Report > Report Power** from the main menu bar.

2. Verify that all the input settings look right and click **OK**.

The Report Power command creates a Power Report under the **power\_1** tab in the results windows area.

3. Note the total power (**Total On-Chip Power**) in the power report **Summary** page (**0.335W** in [Figure 30](#)).

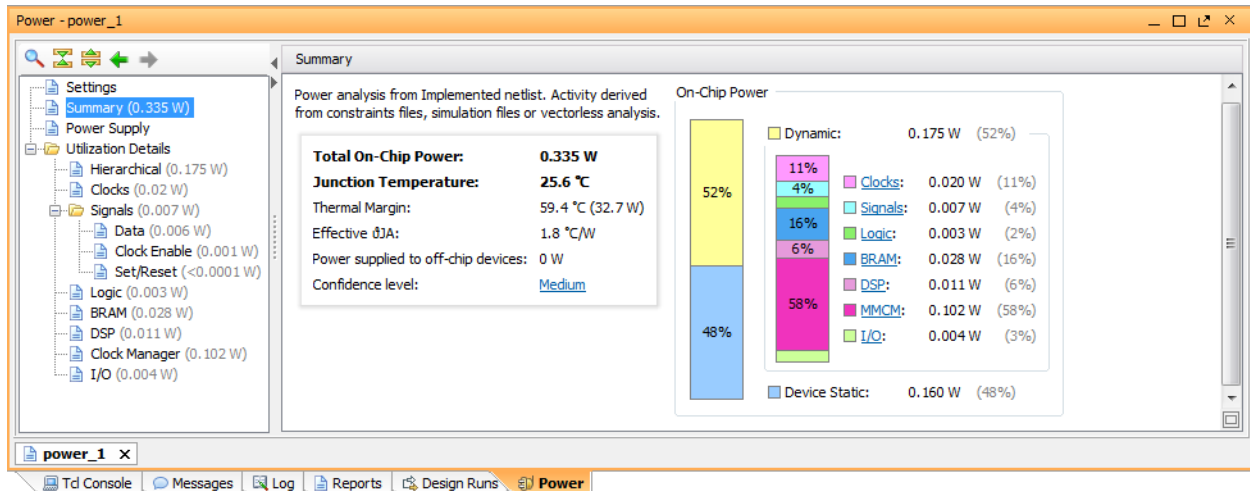


Figure 30: Running Post-Implementation Report Power in Vectorless Mode

## Step 3: Running Report Power with Vivado Simulator SAIF Data

The project directory contains the SAIF output file requested in the previous timing simulation run. We use this SAIF file – a “Switching Activity Interchange Format” file – to further guide the power analysis algorithm.

1. In the main menu bar, select **Tools > Report > Report Power**.
2. In the Report Power dialog box, specify the SAIF file location in the **Input Files** tab.

The SAIF file, which was requested in the simulation settings prior to running timing simulation, should appear here:

```
/<project_directory>/<project_name>/<project_name>.sim/  
sim_1/impl/timing/power_tutorial_timing_xsim.saif
```

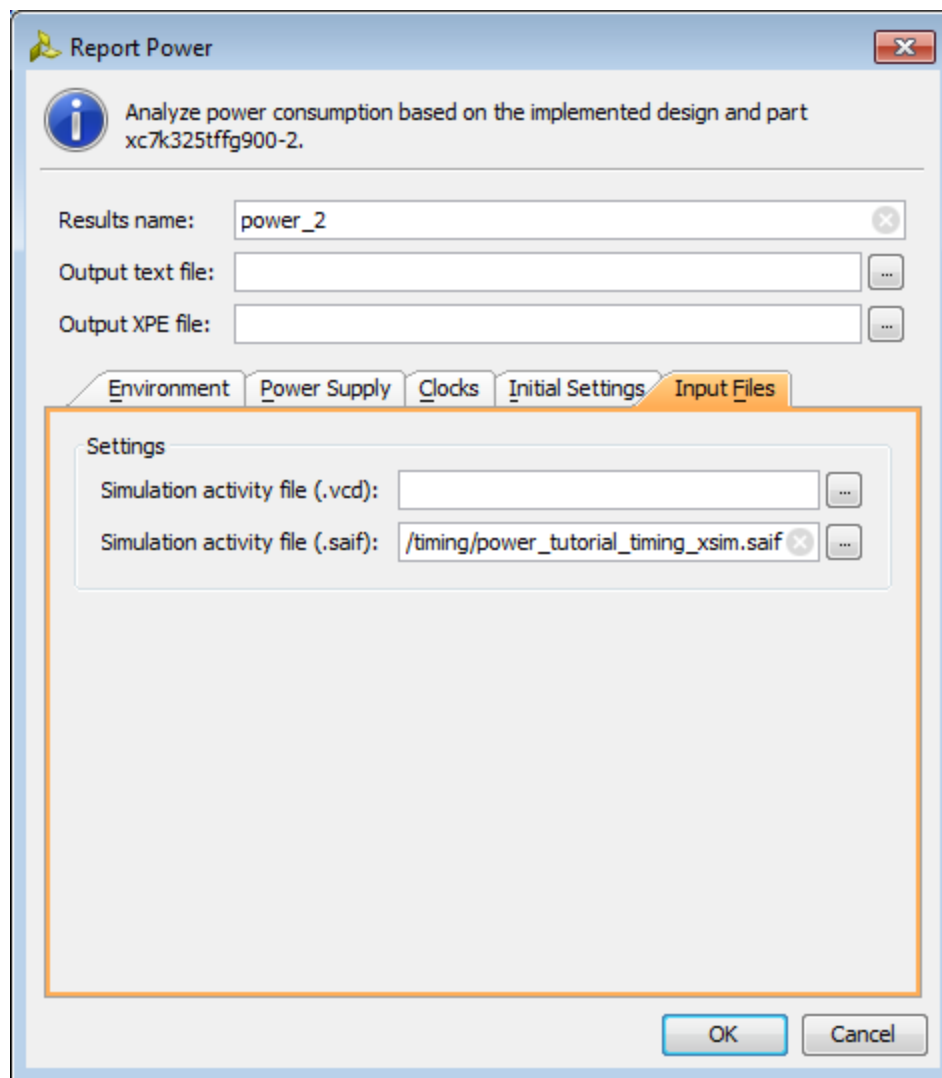


Figure 31: Specifying SAIF File Location

3. Click **OK** in the Report Power dialog box.

After the Report Power command completes, the **Power** tab in the results windows area displays Power Report **power\_2**.

4. Note the change in total power (**Total On-Chip Power** in the **Summary** view) in the **power\_2** report compared to the **power\_1** report. The total power estimated in the report generated with SAIF file data will be different than the total power estimated in the vectorless run (**power\_1** results).

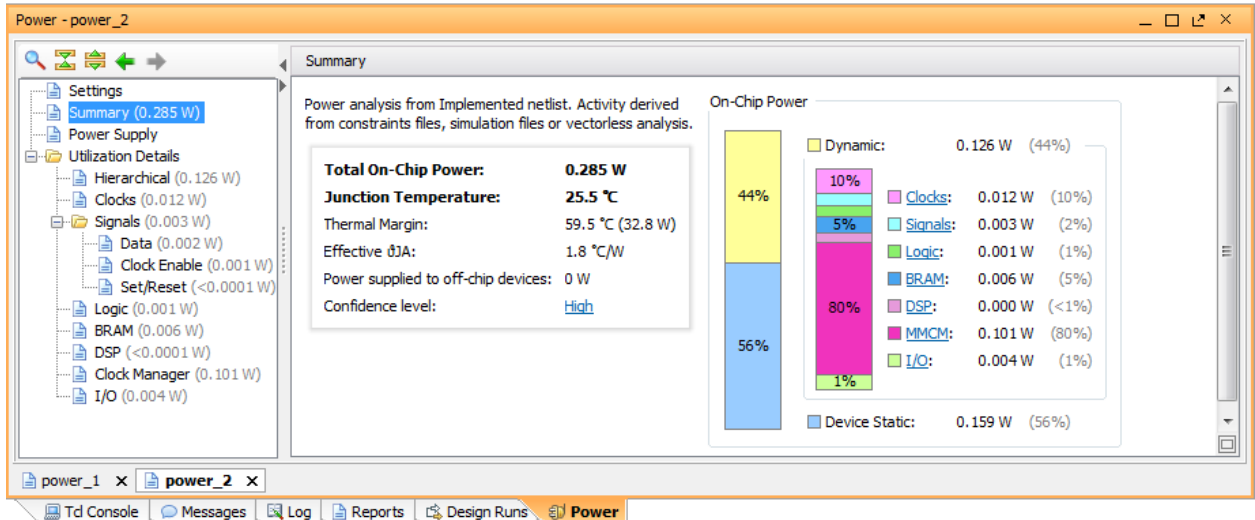


Figure 32: Power Report with SAIF Data

- Examine the summary and block level power distribution in the **Summary** view of the Power Report.
- Go to the **Signals > Data** view in the Power Report. Note that all the **Signal Rate** data has been set from simulation data the SAIF file provided.

The data is color coded to indicate activity rates read from the **Simulation** output file.

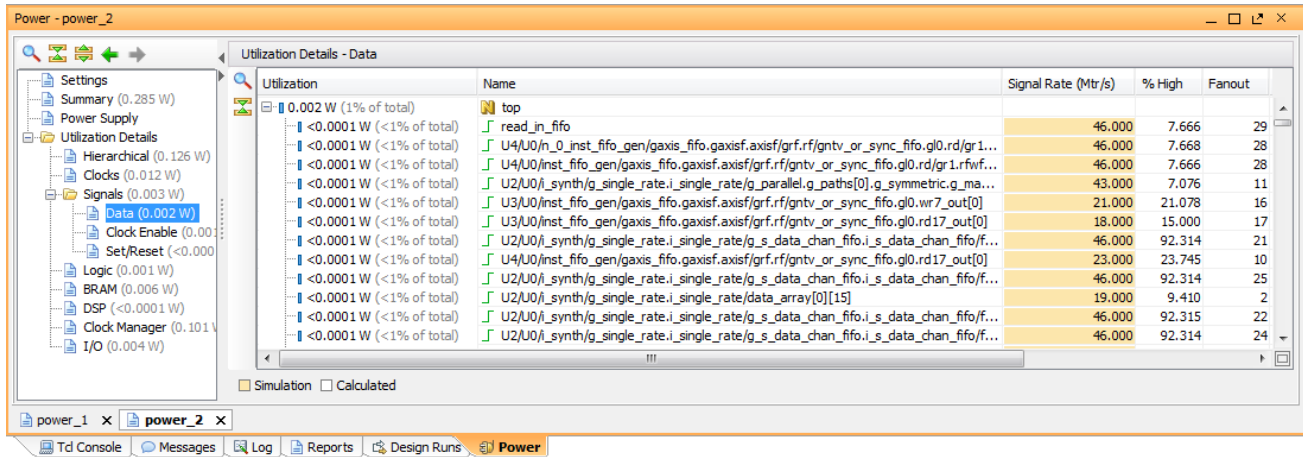


Figure 33: Power Report Signals > Data View - Simulation SAIF Data for Signal Rates

- In the **Summary** view of the **power\_1** report (the report generated by the vectorless analysis), click on the Confidence level (**Medium** in Figure 34).

The Confidence Level is a measurement of the accuracy and the completeness of the input data Report Power uses as it performs a power analysis.

Notice that for the vectorless analysis the **Confidence Level** is **Medium**, since for the **I/O Activity** more than 25% of inputs are missing user specification and for **Internal Activity** less than 25% of internal nodes are user specified.

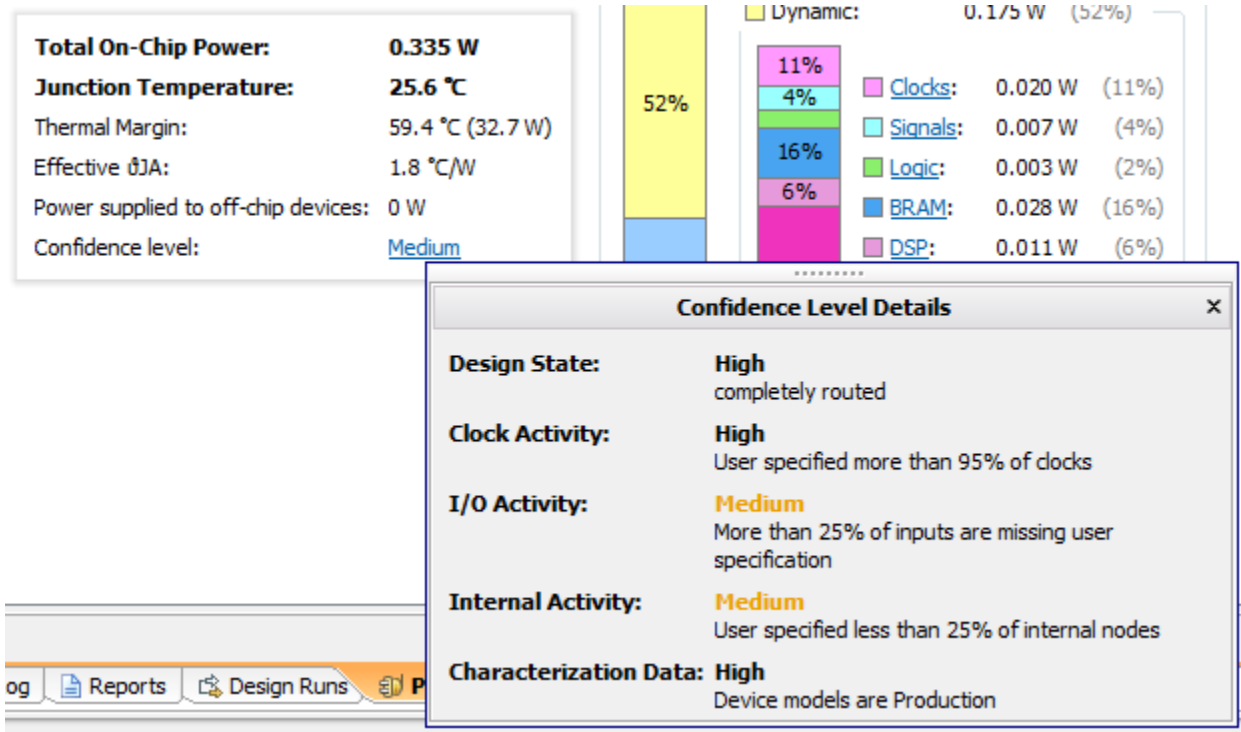


Figure 34: Confidence Level for Vectorless Power Analysis (power\_1)

- In the **Summary** view of the **power\_2** report (the report generated by the analysis for which you specified a SAIF file as input), click on the Confidence level (**High** in Figure 34).

Notice that the **Confidence Level** has increased to **High**, since for the **I/O Activity** more than 95% of inputs are user specified and for **Internal Activity** more than 25% of internal nodes are user specified.

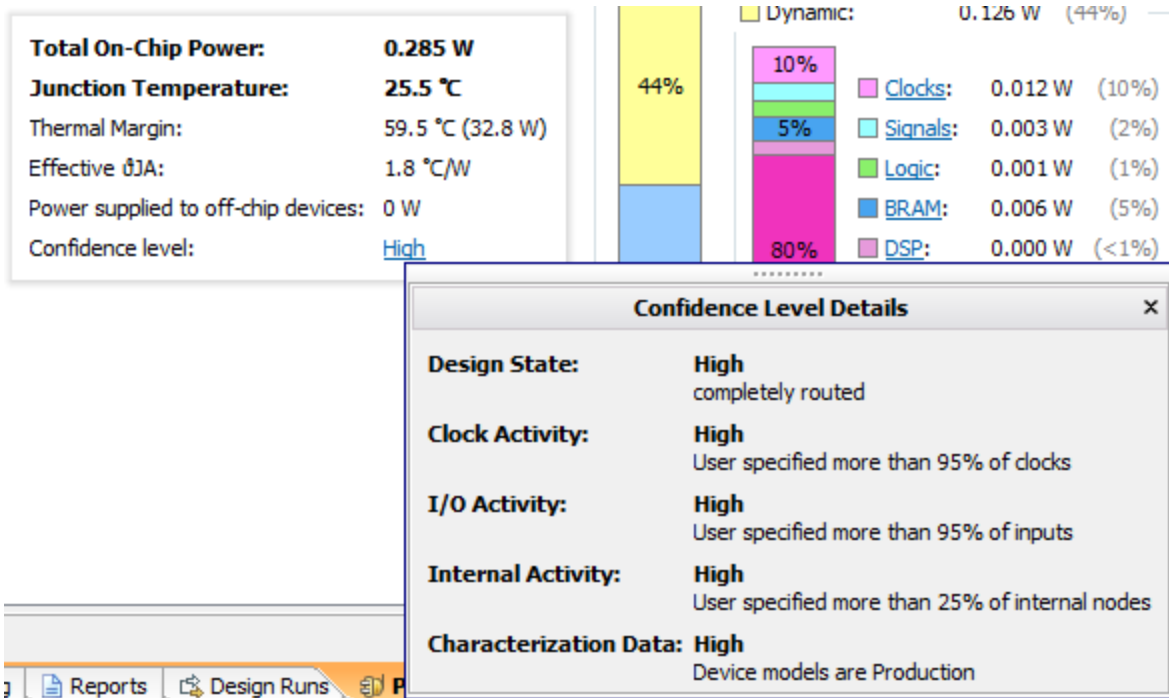


Figure 35: Confidence Level for Power Analysis with SAIF Input (power\_2)

# Lab 3: ModelSim Timing Simulation and Power Analysis

---

## Introduction

In this lab, you will learn about generating a SAIF file after running a timing level simulation using a ModelSim simulator. The lab will take you through the steps for SAIF file creation, running timing simulation, and estimating power using the SAIF data.



**IMPORTANT:** Ensure that the Vivado Design Suite knows where to pick up the ModelSim tool.

You can either: Manually set the path to ModelSim/QuartaSim using the \$PATH environment variable

OR

From the **Tools > Options > General** dialog box, define the path to the simulator in the Vivado IDE under **3<sup>rd</sup> Party Tools** section '**QuartaSim/modelSim install path:**'

---

---

## Step 1: Set Up to Run Timing Simulation in ModelSim

Now you are ready to set up and launch the Vivado simulator to run post-implementation timing simulation. You will set the timing simulation properties in the Vivado IDE, and run the timing simulation.

1. In the Flow Navigator, select **Implemented Design**.
2. In the Flow Navigator, click **Simulation Settings** to set the timing simulation properties.
3. In the Project Settings dialog box, set the **Target simulator** to **QuartaSim/ModelSim**.
4. Click **Yes** to **Ok to change your target simulator to 'QuartaSim/ModelSim'?**

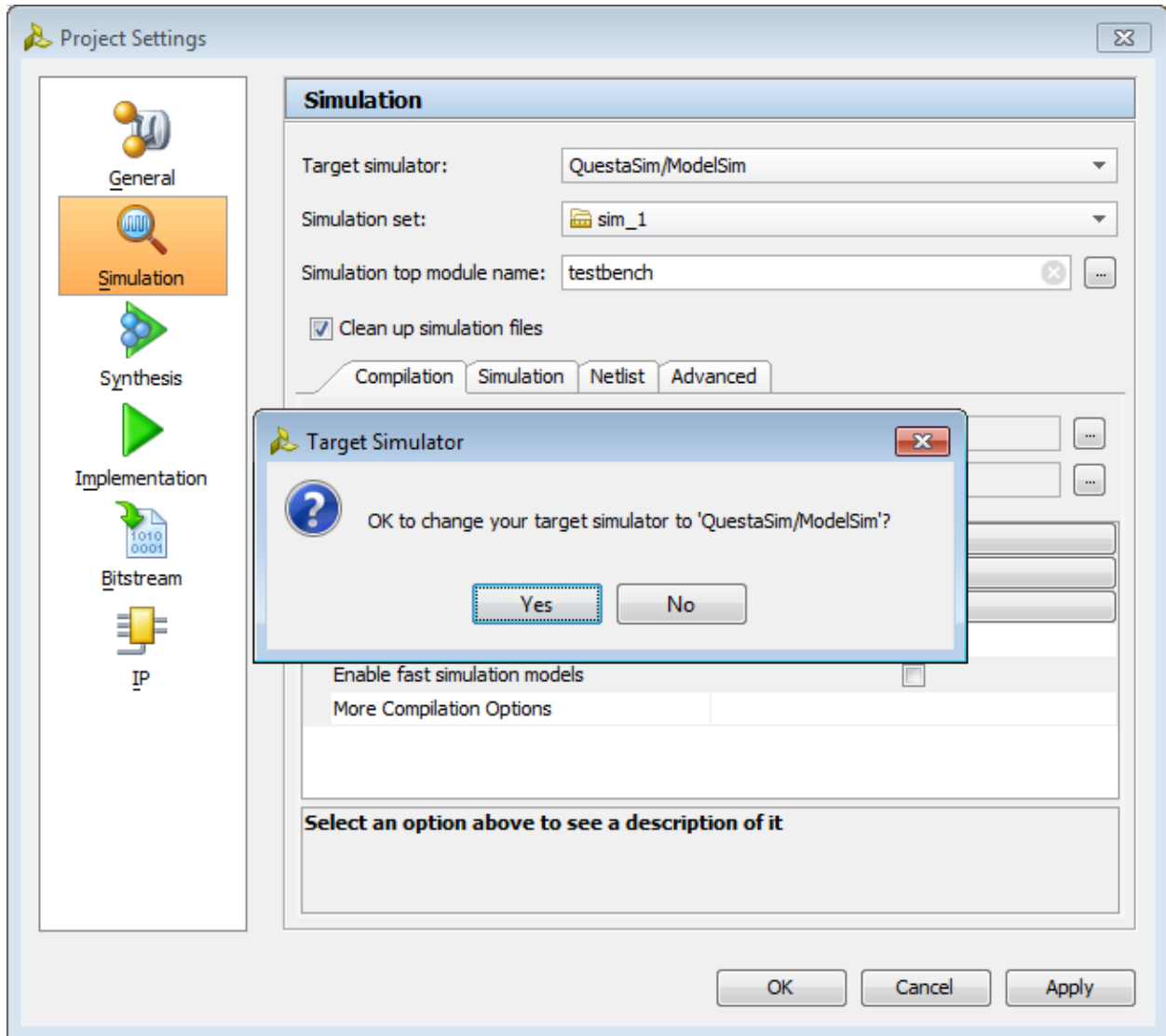


Figure 36: Simulation Settings – Changing the Target Simulator

5. Ensure that the **Compiled library location** points to a valid location for the compiled Xilinx simulation libraries.

To determine this location, type `compile_simlib` in the Vivado IDE Tcl Console, to compile simulation libraries for ModelSim and QuestaSim simulators in a directory with read/write permissions. Note the path to this directory and enter it in the **Compiled library location** field. It should point to the `compplib` directory.

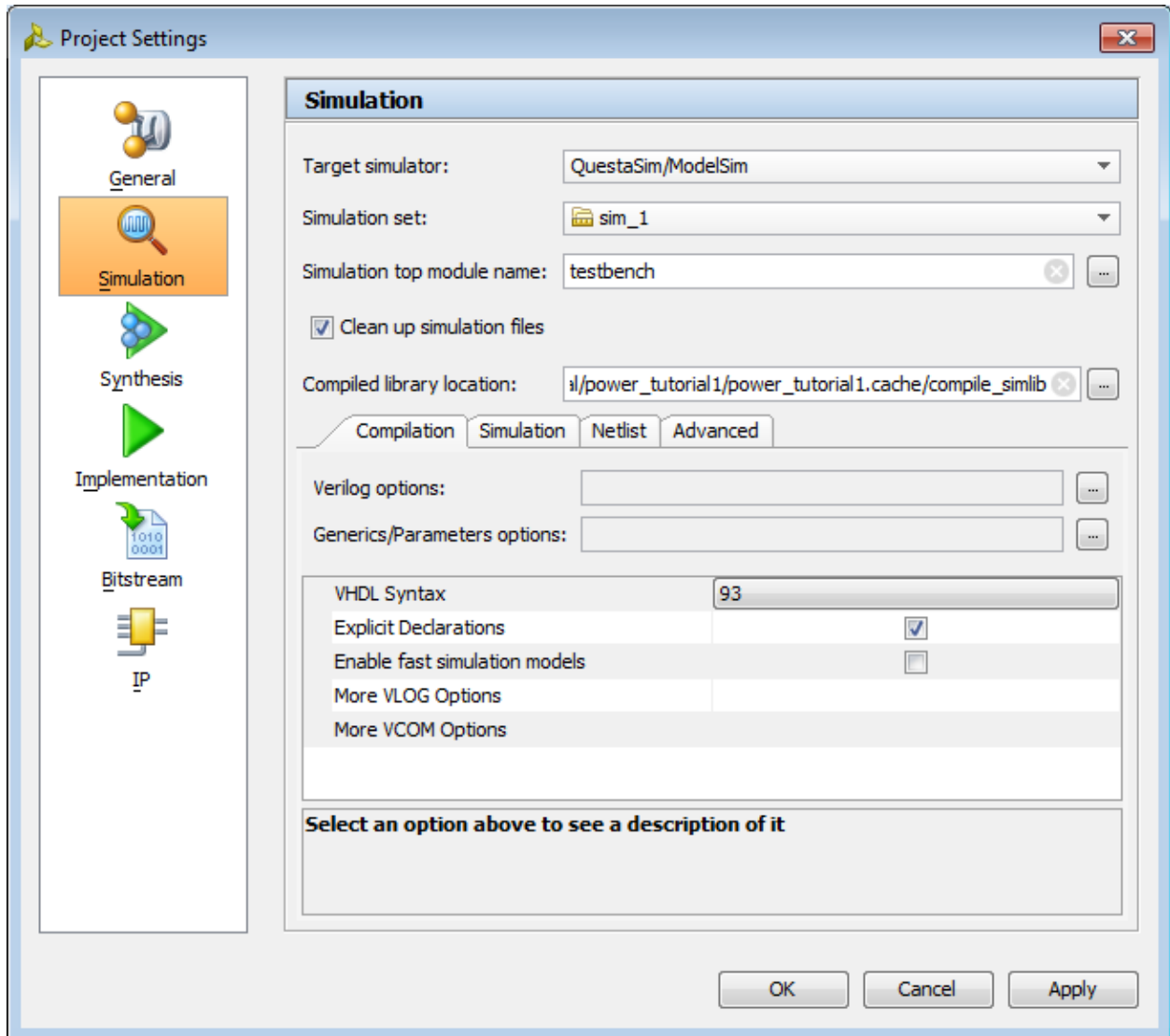


Figure 37: Simulation Settings – Entering the Compiled Library Location

6. In the **Simulation** tab, specify the **Design Under Test Instance** as testbench/UUT.
7. Set the **SAIF Filename** to power\_tutorial\_timing\_modelsim.saif
8. Note that the **Simulation Run Time** is 1000ns.

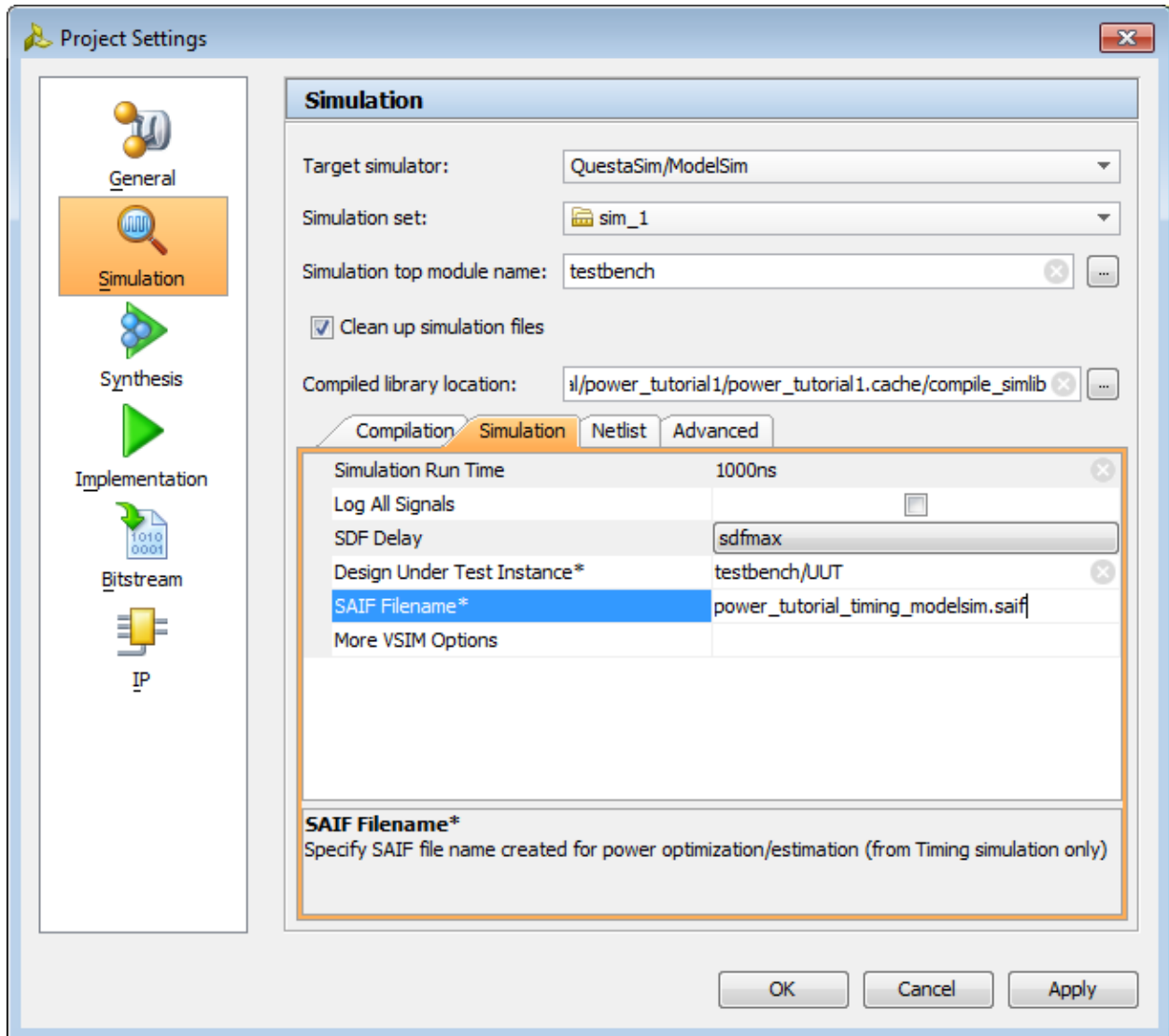
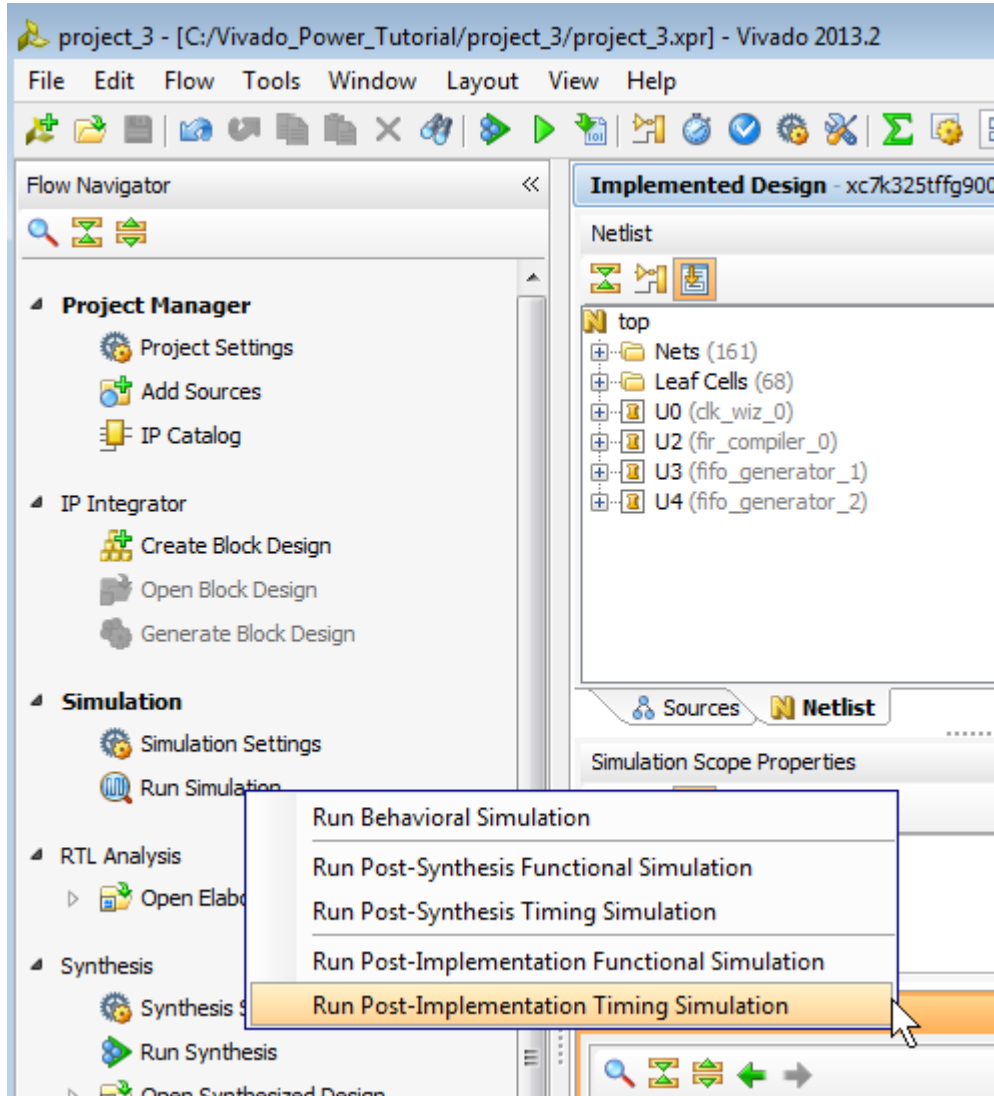


Figure 38: Simulation Settings – Simulation Tab

9. Click **OK**.

With the simulation settings properly configured, you can launch the ModelSim simulator to perform a timing simulation of the design.

10. In the Flow Navigator, click **Run Simulation > Run Post-Implementation Timing Simulation**.



**Figure 39: Running Post-Implementation Timing Simulation**

A separate QuestaSim/ModelSim GUI opens and starts simulating the design.

11. After the Modelsim simulator has finished simulating the design, ensure that the SAIF file requested has been generated. Check to see that the SAIF file requested in the simulation settings prior to running simulation appears in this directory:

```
/<project_directory>/<project_name>/<project_name>.sim/  
sim_1/impl/timing/power_tutorial_timing_modelsim.saif
```

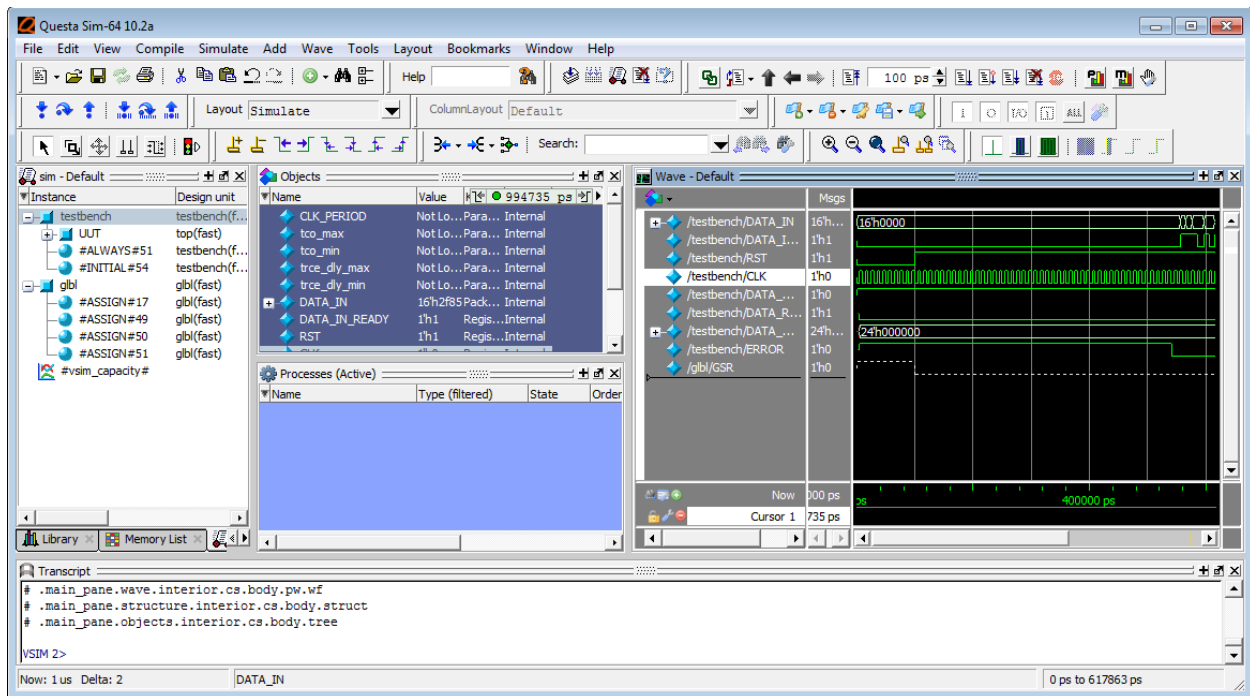


Figure 40: Running Post-Implementation Timing Simulation on ModelSim

## Step 2: Running Report Power in Vectorless Mode

If SAIF based report\_power has already been run in this session, please close the Implemented design and reopen it. This will clear the SAIF data in the power engine from the earlier runs.

1. In the Flow Navigator, select **Implemented Design > Report Power** to open the Report Power dialog box.  
Alternatively, select **Tools > Report > Report Power** in the main menu bar.
2. Verify that all the input settings are correct and click **OK**.

The Report Power command creates a Power Report under the **power\_1** tab in the results windows area. Note the total power (0.332W in the figure below).

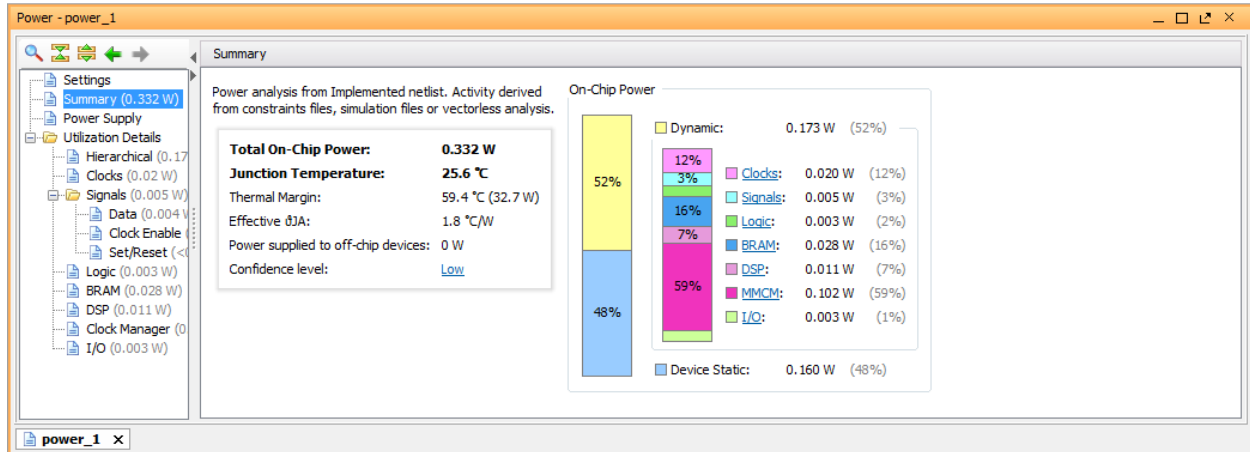


Figure 41: Power Report – Vectorless Mode Run

## Step 3: Running Report Power with ModelSim SAIF Data

The SAIF output file requested in the simulation run has been generated under the project directory. We use this SAIF file – a “Switching Activity Interchange Format” file – to further guide the power estimation algorithm.

1. In the main menu bar, select **Tools > Report > Report Power**.
2. In the Report Power dialog box, specify the SAIF file location in the **Input Files** tab.

The SAIF file, which was requested in the simulation settings prior to running simulation, should appear here:

```
/<project_directory>/<project_name>/<project_name>.sim/  
sim_1/impl/timing/power_tutorial_timing_modelsimsaif
```

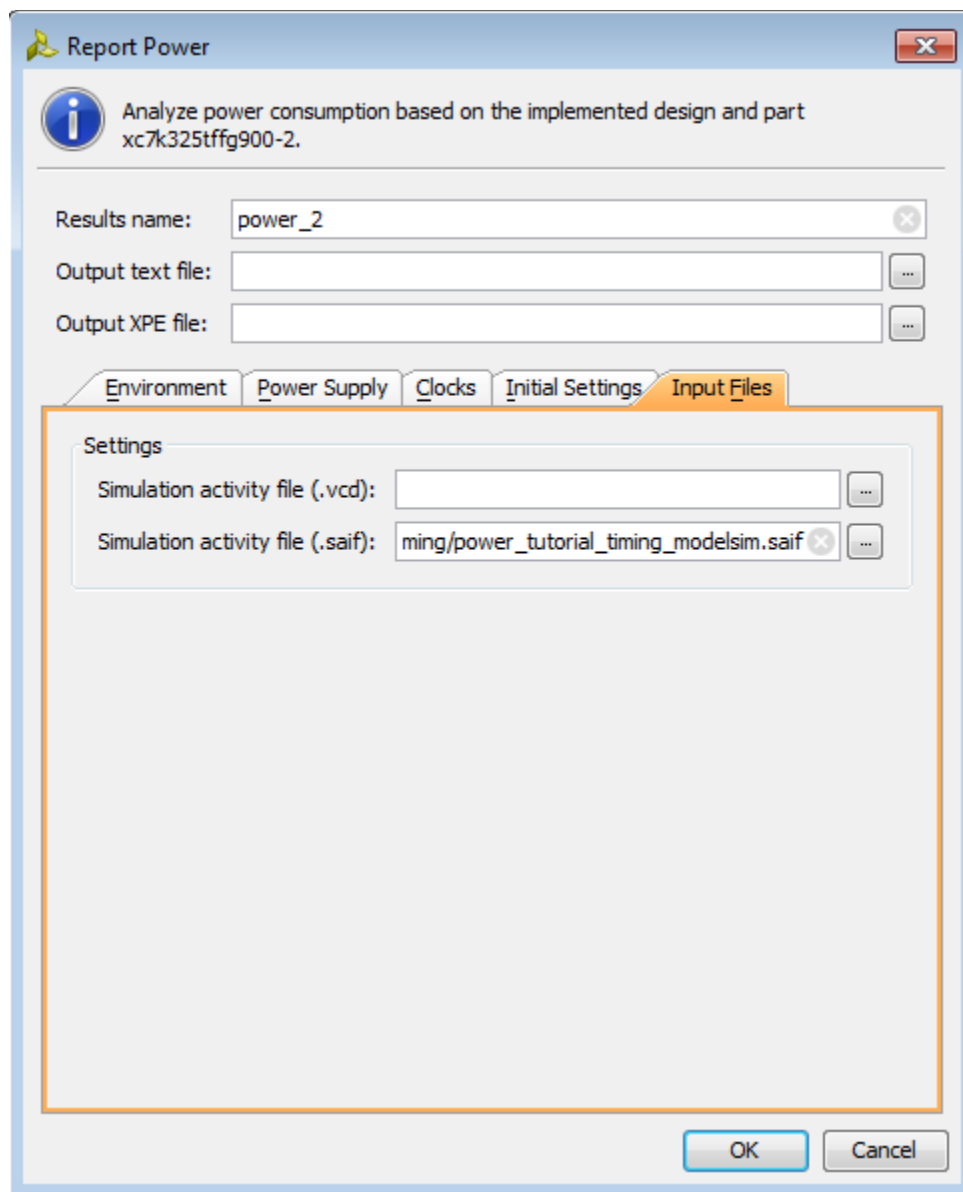


Figure 42: Report Power Dialog Box – Specifying SAIF File

3. Click **OK** in the Report Power dialog box.

The Report Power command runs, and the Power Report **power\_2** is generated in the **Power** tab of the results windows area.

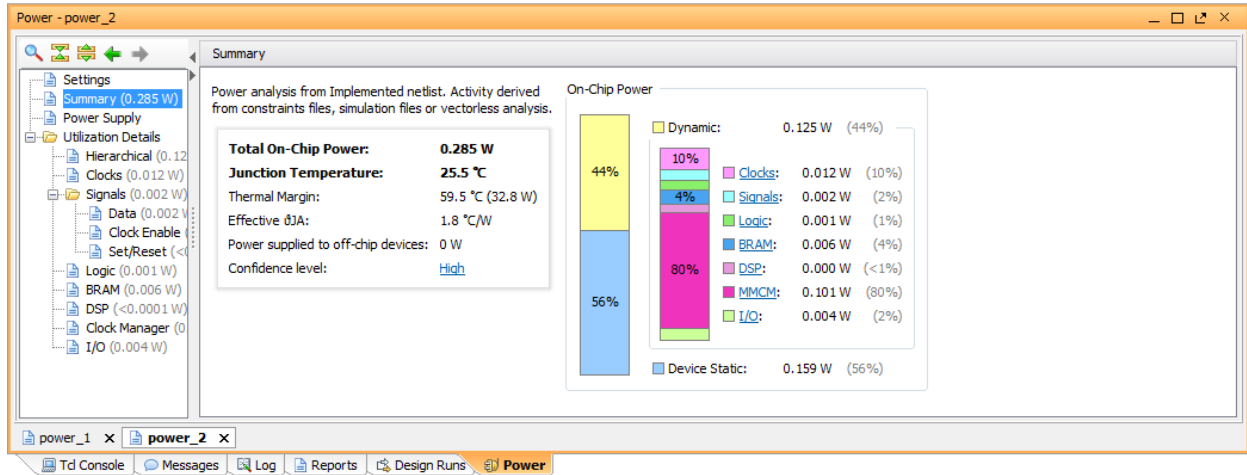


Figure 43: Power Report – With SAIF File Data

- Go to the **Signals > Data** view in the Power Report and scroll to the right. Note that all the Signal Rate data has been set from simulation SAIF data you provided.

The data is color coded to indicate activity rates read from the **Simulation** output file.

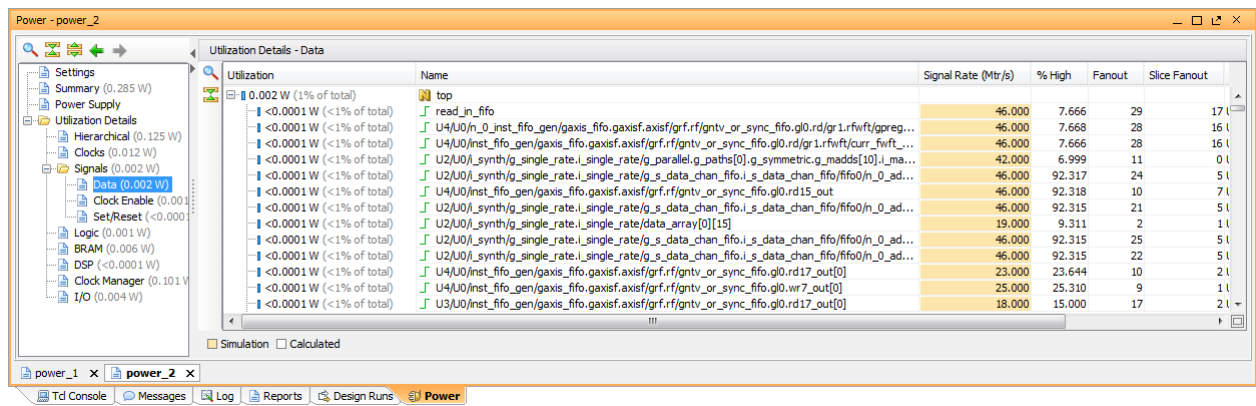


Figure 44: Signals > Data View with SAIF File Data

- Note the change in total power (**Total On-Chip Power** in the **Summary** view) in the **power\_2** report compared to the **power\_1** report. The total power estimated in the report generated with SAIF file data will be different than the total power estimated in the vectorless run (**power\_1** results).

# Lab 4: Power Optimization in Vivado

## Introduction

In this lab, you will learn about using the Power Optimization features in Vivado. The lab will take you through the steps for invoking Power Optimization after synthesizing the design. It will also guide on using the power optimization report, making decisions and selectively turning off power optimization on signals, blocks, and hierarchies.



**TIP:** When you run Implementation on your design, the Vivado tools may perform some power optimizations by default during `opt_design`. These optimizations will not affect performance, and will have little impact on area and runtime.

## Step 1: Run `report_power_opt` to Examine User/Design Specific Power Optimizations

1. In the Flow Navigator, select **Implemented Design**.
2. In the main menu bar, select **Tools > Report > Report Power Optimization**.

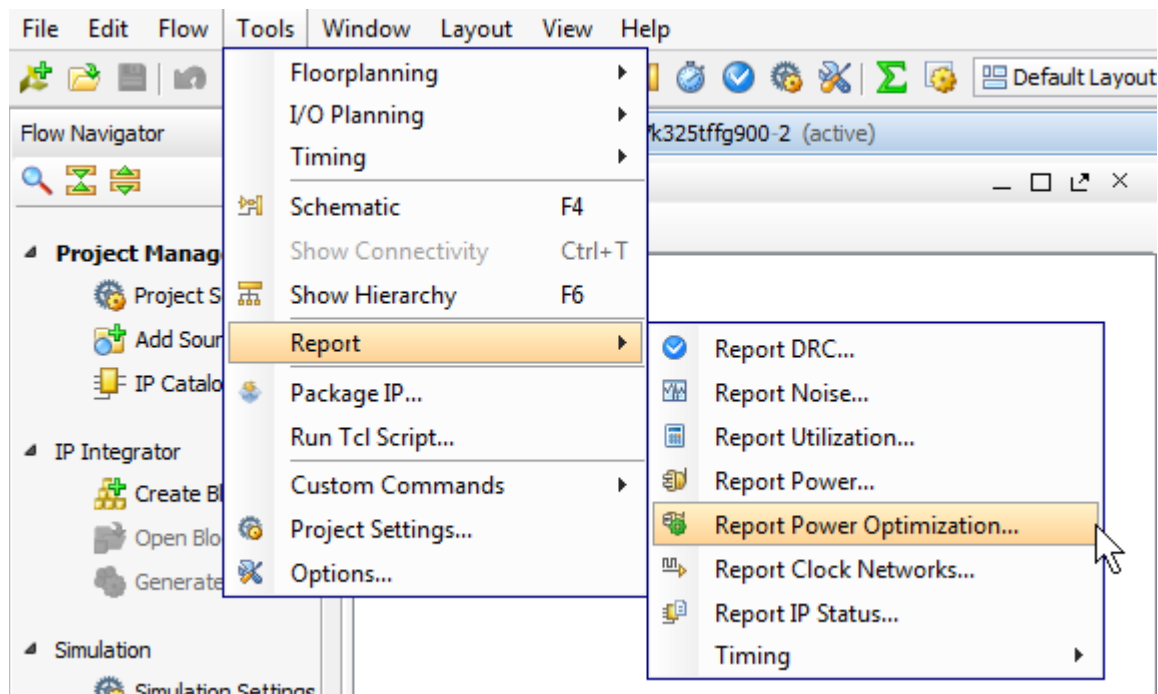


Figure 45: Generating a Power Optimization Report

- The Report Power Optimization dialog box appears, as shown below.

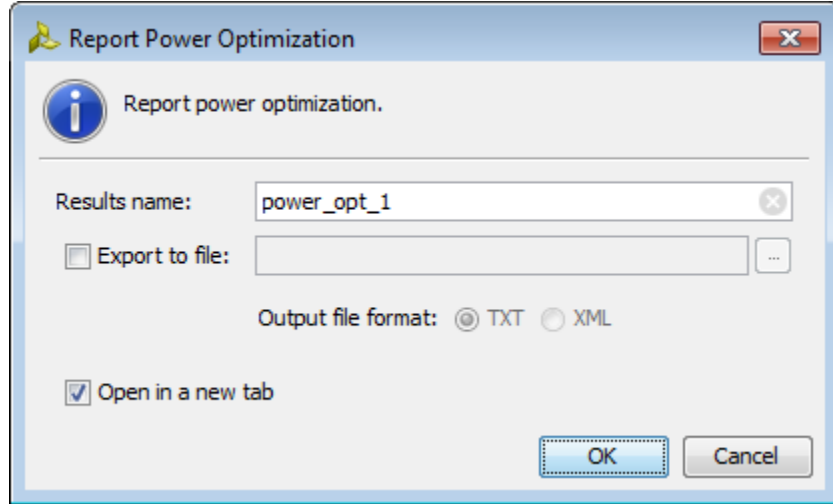


Figure 46: Report Power Optimization Dialog Box

- Type in `power_opt_1` for the **Results name**.
- Ensure that the **Open in a new tab** option is checked.
- Click **OK**.

Alternatively, in the Tcl Console execute this Tcl command:

```
report_power_opt -name power_opt_1
```

- Observe the report **power\_opt1** is generated in the **Power Opt** tab of the results windows area.

When the report opens, the Summary view is displayed in the report.

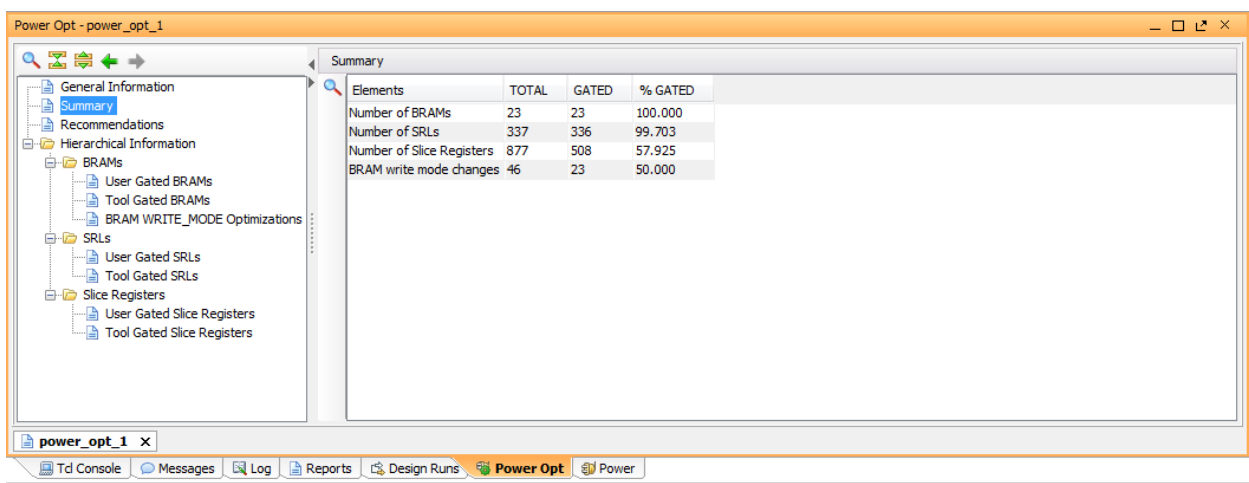


Figure 47: Power Optimization Report – Summary View

- In the power optimization report, select **Tool Gated BRAMs**. Note that the tool has gated the BRAMs.

Also select **BRAM\_WRITE\_MODE Optimization** and note that the tool has made power-efficient BRAM WRITE\_MODE optimizations by default.

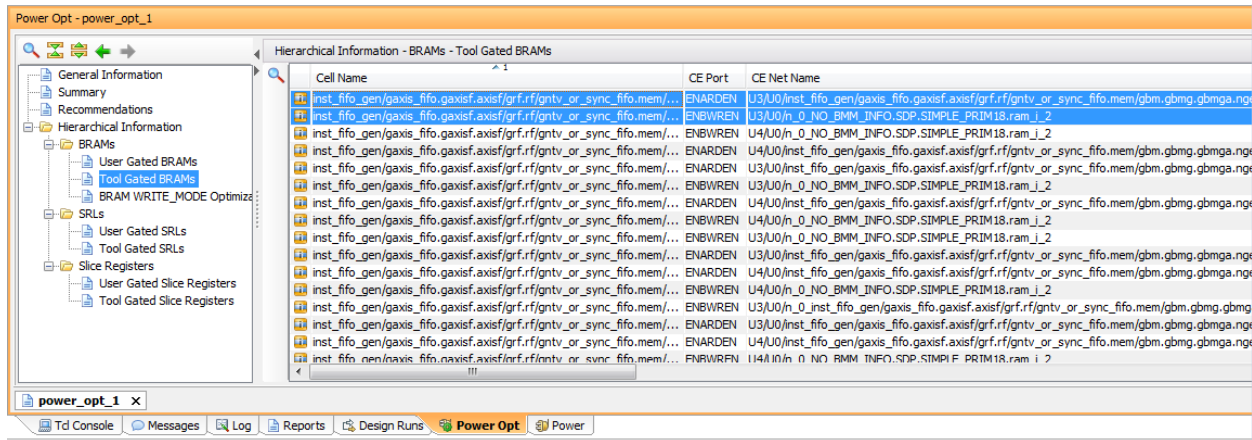


Figure 48: Tool Gated BRAMs

- Note that there are no **Tool Gated SRLs** or **Tool Gated Slice Registers**.

## Step 2: Set Up Options to Run Power Optimization

- In the Flow Navigator, click **Implementation Settings**.
- In the Project Settings dialog box, check the **is\_enabled** checkbox under **Power Opt Design**.

This ensures Power Optimization runs after opt\_design. Enabling the power\_opt\_design option prior to place\_design results in a complete power optimization to be performed. This option yields the best possible power saving from Vivado.

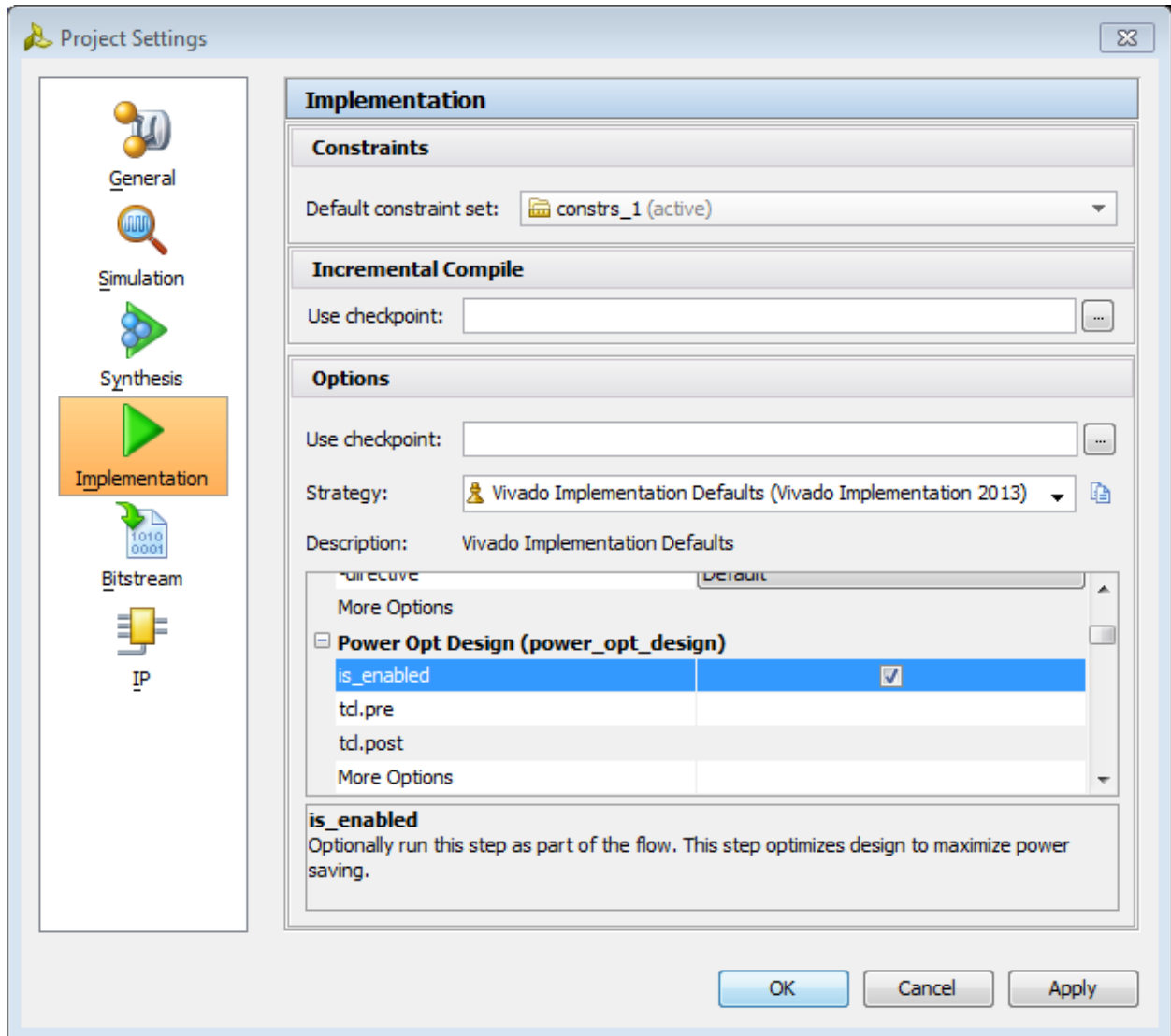


Figure 49: Implementation Settings

3. Click **OK**.
4. In the Create New Run dialog box, click **Yes** to “Properties for the completed run ‘impl\_1’ have been modified. Do you want to preserve the state of ‘impl\_1’ and apply these changes to a new run?”.

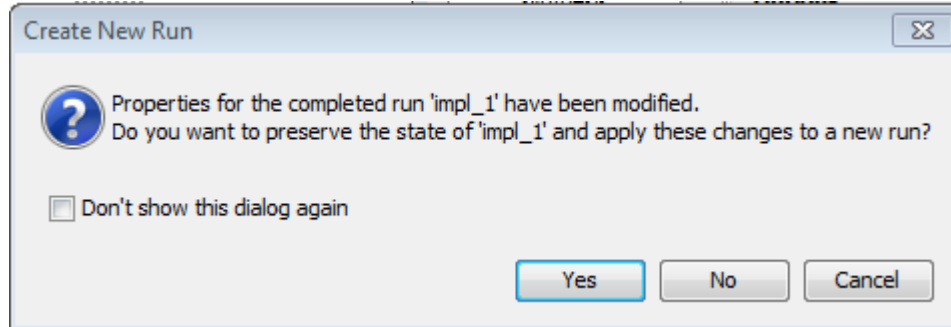


Figure 50: Create New Run Dialog Box

5. in the Create Run dialog box, set the **Run Name** to impl\_2.
6. Click **OK**.
7. In the Flow Navigator, select **Run Implementation**.  
You are running Implementation with Power Optimization turned on.
8. In the Implementation Completed dialog box, select **Open Implemented Design** and click **OK**.

## Step 3: Run report\_power\_opt to Examine Tool Optimizations

1. In the main menu bar, select **Tools > Report > Report Power Optimization**.
2. in the Report Power Optimization dialog box, enter the **Results name** power\_opt\_2.

Alternatively, in the Tcl Console execute this Tcl command:

```
report_power_opt -name power_opt_2
```

3. In the generated report in the Power Opt window, note that there are no Tool Gated SRLs but the tool has gated Slice Registers.

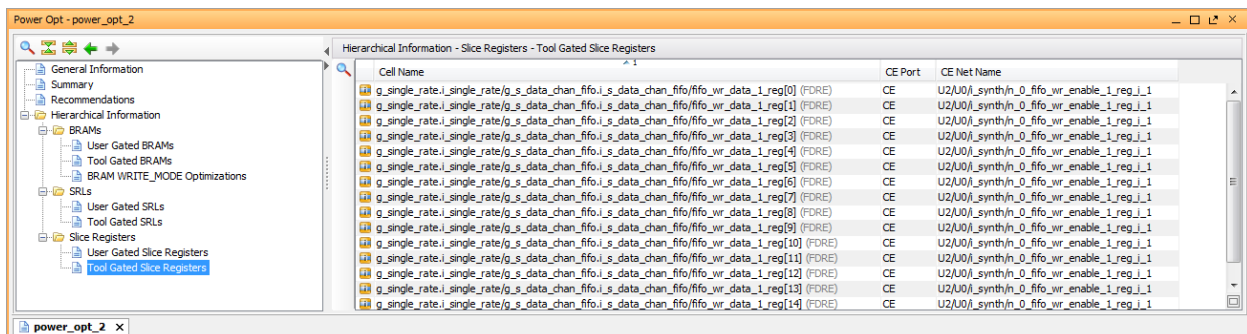


Figure 51: Tool Gated Slice Registers

---

## Step 4: Turn Off Optimizations on Specific Signals and Rerun Implementation

In this step you will turn off the power optimization on specific Slice Registers.



**IMPORTANT:** Power optimization works to minimize the impact on timing while maximizing power savings. However, in certain cases, if timing degrades after power optimization, you can identify and apply power optimizations only on non-timing critical clock domains or modules using the `set_power_opt` XDC command.

---

This step will ensure that the tool does not gate this Slice Register:

```
U2/U0/i_synth/g_single_rate.i_single_rate/  
g_s_data_chan_fifo.i_s_data_chan_fifo/fifo_wr_data_1_reg[0]
```

1. In the Tcl Console, type this command:

```
set_power_opt -exclude_cells [get_cells U2/U0/i_synth/  
g_single_rate.i_single_rate/g_s_data_chan_fifo.i_s_data_chan_fifo/  
fifo_wr_data_1_reg[0] ]
```

This will prevent the tool from gating this Slice Register.

2. From the **Flow Navigator** choose **Run Implementation**
3. Click **Save** in the Save Project dialog box before launching implementation.
4. In the Implementation Completed dialog box, select **Open Implemented Design** and click **OK**.

---

## Step 5: Run `report_power_opt` to Examine Tool Optimizations Again

1. In the main menu bar, select **Tools > Report > Report Power Optimization**.
2. In the Report Power Optimization dialog box, type in the **Results name** as `power_opt_3`.  
Alternatively, execute this Tcl command in the Tcl Console:

```
report_power_opt -name power_opt_3
```
3. In the generated report **power\_opt\_3** in the Power Opt window, display **Tool Gated Slice Registers**.

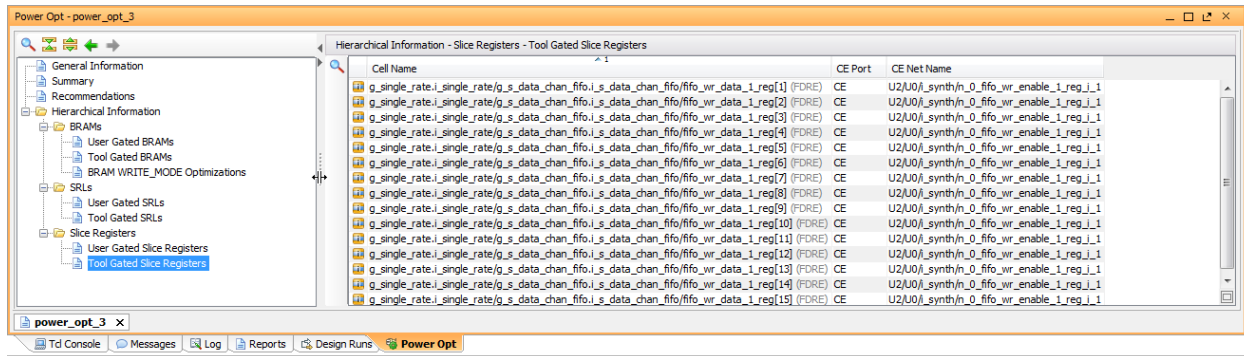


Figure 52: Displaying Tool Gated Slice Registers

4. Note that this slice register is no longer in the list of **Tool Gated Slice Registers**:

```
U2/U0/i_synth/g_single_rate.i_single_rate/g_s_data_chan_fifo.i_s_data_chan_fifo/fifo_wr_data_1_reg[0]
```

## Conclusion

In this tutorial, you accomplished the following:

- Used the Report Power dialog box to verify and set device, thermal, and environmental conditions that contribute to power estimation.
- Synthesized the design and estimated the power after synthesis.
- Set switching activities on an I/O port and reran Report Power.
- Ran behavioral simulation using the Vivado Simulator and generated a SAIF file that is input to Report Power for a more accurate power analysis.
- Implemented the design, ran post-implementation timing simulation using the Vivado Simulator, and generated a SAIF file that is input to report power for a more accurate power analysis.
- Ran Modelsim post-implementation timing simulation and generated a SAIF file that is input to report power for a more accurate power analysis.
- Learned how to invoke power optimization as part of an implementation run.
- Examined the power optimization report and selectively turned off power optimizations on a cell in the design