

 **Rob Graessle** Update HDL Lab 3 to 2024.1170286f · last week 

149 lines (86 loc) · 8.79 KB

[Preview](#) [Code](#) [Blame](#)[Raw](#)   

## Lab 3: Timing and Resource Analysis

In this lab, you learn how to verify the functionality of your designs by simulating in Simulink® to ensure that your Vitis Model Composer design is correct when you implement the design in your target AMD device.

### Objectives

After completing this lab, you will be able to:

- Identify timing issues in the HDL files generated by Vitis Model Composer and discover the source of the timing violations in your design.
- Perform resource analysis and access the existing resource analysis results, along with recommendations to optimize.

### Procedure

This lab has two primary parts:

- In Step 1 you will learn how to do timing analysis in Vitis Model Composer.
- In Step 2 you will learn how to perform resource analysis in Vitis Model Composer.

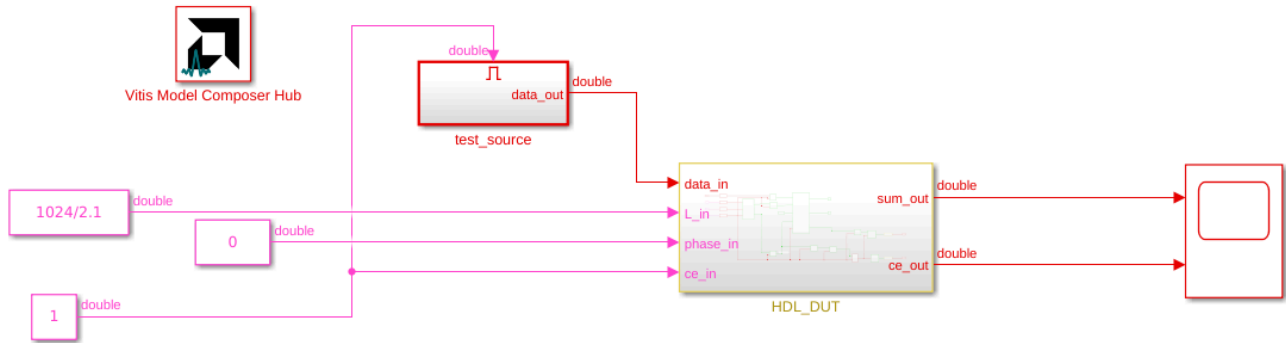
### Step 1: Timing Analysis in Vitis Model Composer

1. Invoke Vitis Model Composer.
  - On Windows systems select **Windows > AMD Design Tools > Vitis Model Composer 2024.1**.
  - On Linux systems, type `model_composer` at the command prompt.
2. Navigate to the Lab3 folder: `\HDL_Library\Lab3`.

You can view the directory contents in the MATLAB® Current Folder browser, or type `ls` at the command line prompt.

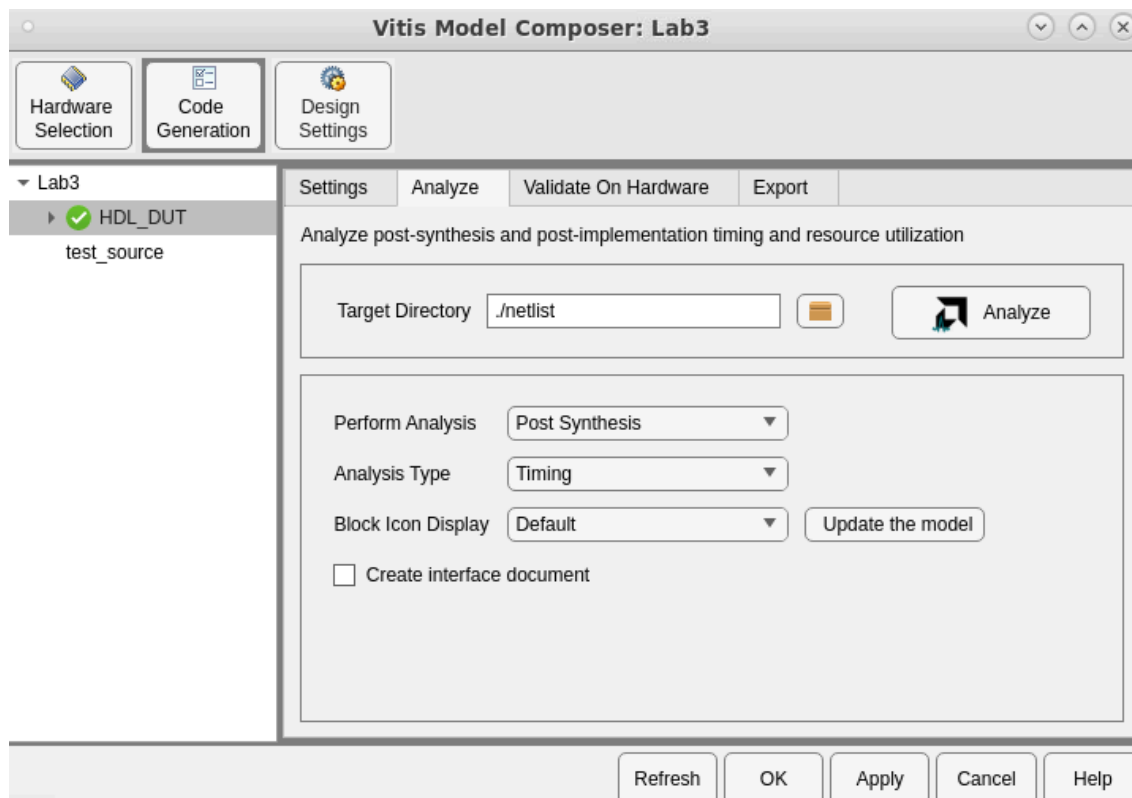
3. Open the Lab3 design using one of the following:
  - At the MATLAB command prompt, type `open Lab3.slx`
  - Double-click Lab3.slx in the Current Folder browser.

The Lab3 design opens, as shown in the following figure.



Note that this design differs from the previous labs in that all of the HDL blocks are placed inside a subsystem (**HDL\_DUT**) while the Vitis Model Composer Hub block remains at the top level of the model. Going forward, this is a recommended approach for architecting your model using Vitis Model Composer.

4. From the Simulink Toolstrip, click the Run button to simulate the design.
5. Double-click the **Vitis Model Composer Hub** block to open the Properties Editor.
6. Select the **HDL\_DUT** subsystem on the left, then the **Analyze** tab.
7. From the Perform Analysis menu, select **Post Synthesis** and from Analyzer type menu select **Timing** as shown in the following figure.



8. In the Model Composer Hub dialog box, click **Analyze**.

When you generate, the following occurs:

- Vitis Model Composer generates the required files for the selected compilation target. For timing analysis Vitis Model Composer invokes Vivado in the background for the design project, and passes design timing constraints to Vivado.

- Depending on your selection for Perform Analysis (Post Synthesis or Post Implementation), the design runs in Vivado through synthesis or through implementation.
- After the Vivado tools run is completed, timing paths information is collected and saved in a specific file format from the Vivado timing database.
- Vitis Model Composer processes the timing information and displays a Timing Analyzer table with timing paths information as shown in the following figure.

Timing Analyzer: Lab3

Post Synthesis Timing Paths: Clicking on an instance name highlights corresponding block/subsystem in the model

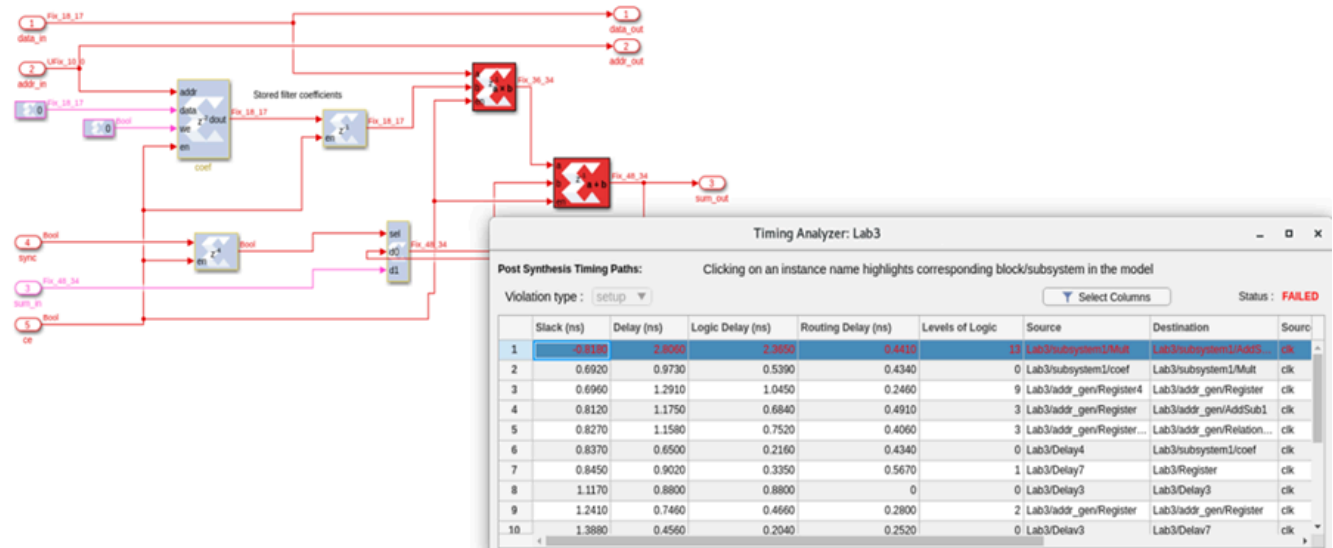
Violation type: **Setup** Select Columns Status: **FAILED**

	Slack (ns)	Delay (ns)	Logic Delay (ns)	Routing Delay (ns)	Levels of Logic	Source	Destination	Source Clock	Destination Clock	Path Constraints
1	-0.8180	2.8060	2.3650	0.4410	13	Lab3/subsystem1/Mult	Lab3/subsystem1/AddSub2	clk	clk	create_clock -name clk -period 2 [get_ports ...
2	0.6920	0.9730	0.5390	0.4340	0	Lab3/subsystem1/coef	Lab3/subsystem1/Mult	clk	clk	create_clock -name clk -period 2 [get_ports ...
3	0.6960	1.2910	1.0450	0.2460	9	Lab3/addr_gen/Register4	Lab3/addr_gen/Register	clk	clk	create_clock -name clk -period 2 [get_ports ...
4	0.8120	1.1750	0.6840	0.4910	3	Lab3/addr_gen/Register	Lab3/addr_gen/AddSub1	clk	clk	create_clock -name clk -period 2 [get_ports ...
5	0.8270	1.1580	0.7520	0.4060	3	Lab3/addr_gen/Register10	Lab3/addr_gen/Relation1	clk	clk	create_clock -name clk -period 2 [get_ports ...
6	0.8370	0.6500	0.2160	0.4340	0	Lab3/Delay4	Lab3/subsystem1/coef	clk	clk	create_clock -name clk -period 2 [get_ports ...
7	0.8450	0.9020	0.3350	0.5670	1	Lab3/Delay7	Lab3/Register	clk	clk	create_clock -name clk -period 2 [get_ports ...
8	1.1170	0.8800	0.8800	0	0	Lab3/Delay3	Lab3/Delay3	clk	clk	create_clock -name clk -period 2 [get_ports ...
9	1.2410	0.7460	0.4660	0.2800	2	Lab3/addr_gen/Register	Lab3/addr_gen/Register	clk	clk	create_clock -name clk -period 2 [get_ports ...
10	1.3880	0.4560	0.2040	0.2520	0	Lab3/Delay3	Lab3/Delay7	clk	clk	create_clock -name clk -period 2 [get_ports ...
11	1.3900	0.3860	0.2160	0.1700	0	Lab3/addr_gen/Relation1	Lab3/Delay3	clk	clk	create_clock -name clk -period 2 [get_ports ...
12	1.4000	0.5740	0.3350	0.2390	1	Lab3/Delay7	Lab3/Register1	clk	clk	create_clock -name clk -period 2 [get_ports ...
13	1.4340	0.4100	0.2160	0.1940	0	Lab3/addr_gen/AddSub1	Lab3/addr_gen/Register10	clk	clk	create_clock -name clk -period 2 [get_ports ...
14	1.4340	0.4100	0.2160	0.1940	0	Lab3/addr_gen/AddSub1	Lab3/addr_gen/Register3	clk	clk	create_clock -name clk -period 2 [get_ports ...
15	1.4470	0.3970	0.2160	0.1810	0	Lab3/subsystem1/AddSub2	Lab3/Delay2	clk	clk	create_clock -name clk -period 2 [get_ports ...
16	1.4580	0.3860	0.2160	0.1700	0	Lab3/addr_gen/Register3	Lab3/Delay4	clk	clk	create_clock -name clk -period 2 [get_ports ...
17	1.4580	0.3860	0.2160	0.1700	0	Lab3/Delay4	Lab3/Delay4	clk	clk	create_clock -name clk -period 2 [get_ports ...
18	1.4580	0.3860	0.2160	0.1700	0	Lab3/Delay2	Lab3/Register	clk	clk	create_clock -name clk -period 2 [get_ports ...

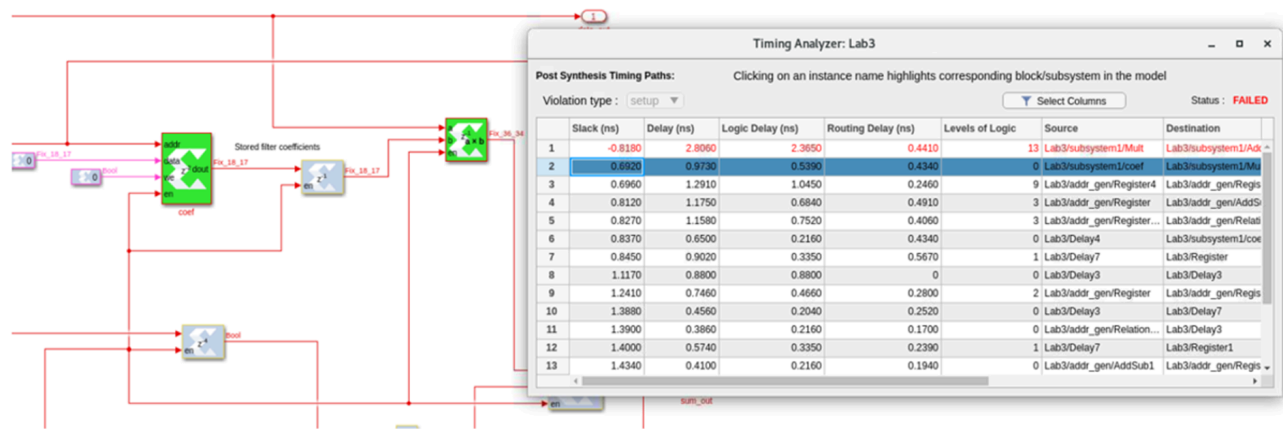
In the timing analyzer table, paths with lowest slack values display, with the worst slack showing at the top and increasing toward the bottom. Paths with timing violations have a negative slack and display in red.

- Cross probe from the Timing Analyzer table to the Simulink model by clicking any path in the Timing Analyzer table, which highlights the corresponding Vitis Model Composer HDL blocks in the model. This allows you to troubleshoot timing violations by analyzing the path on which they occur.

When you cross probe, you see the corresponding path as shown in the following figure. Blocks with timing violations are highlighted in red.



- Double-click the second path in the Timing Analyzer table and cross-probe, the corresponding highlighted path in green which indicates no timing violation.

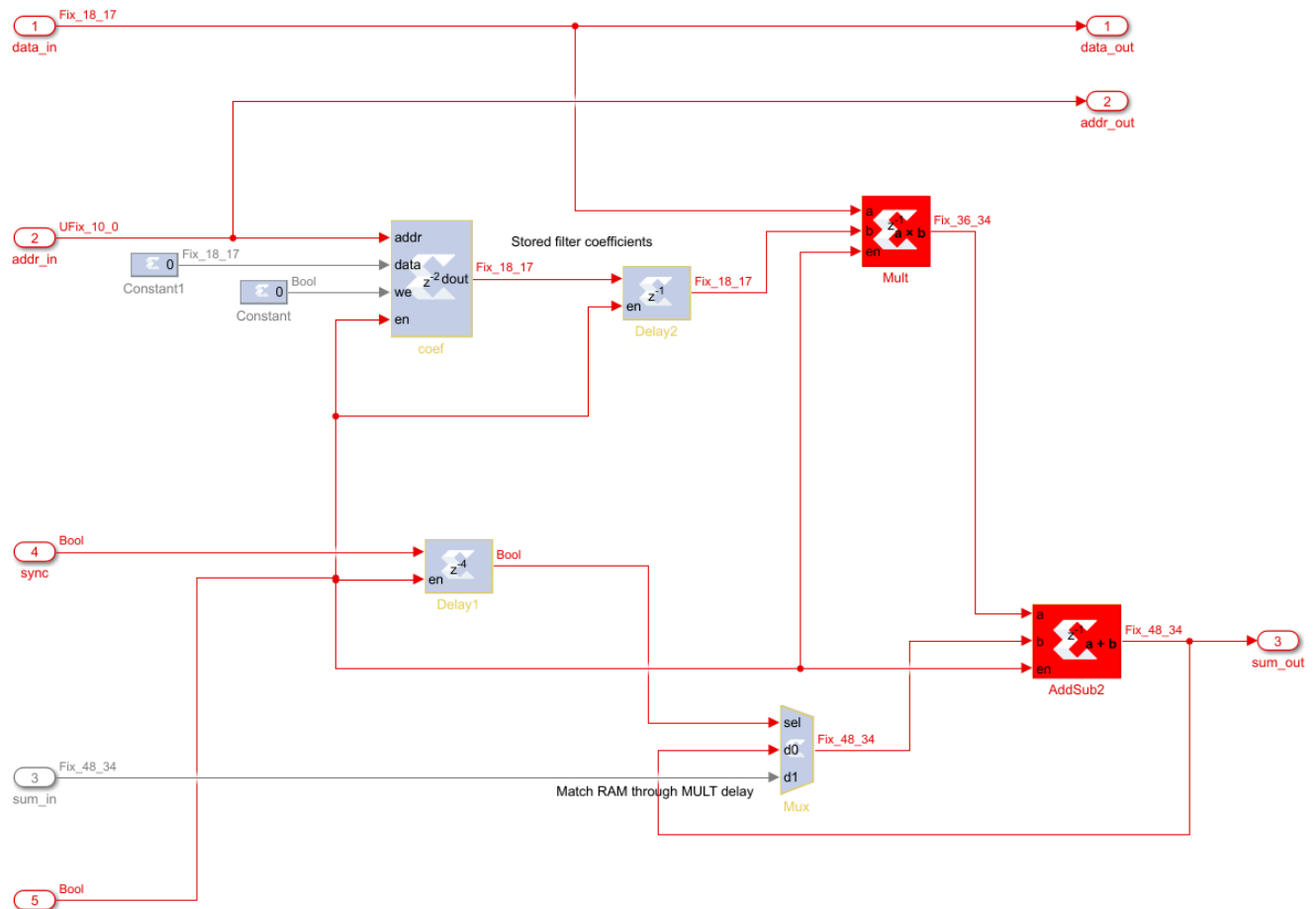


If you close the Timing Analyzer sometime later you might want to relaunch the Timing Analyzer table using the existing timing analyzer results for the model. To do this, simply click **Analyze** in the Model Composer Hub block again. The table that opens will display the results stored in the Target Directory specified in the Model Composer Hub dialog box, regardless of the option selected for Perform Analysis (Post Synthesis or Post Implementation).

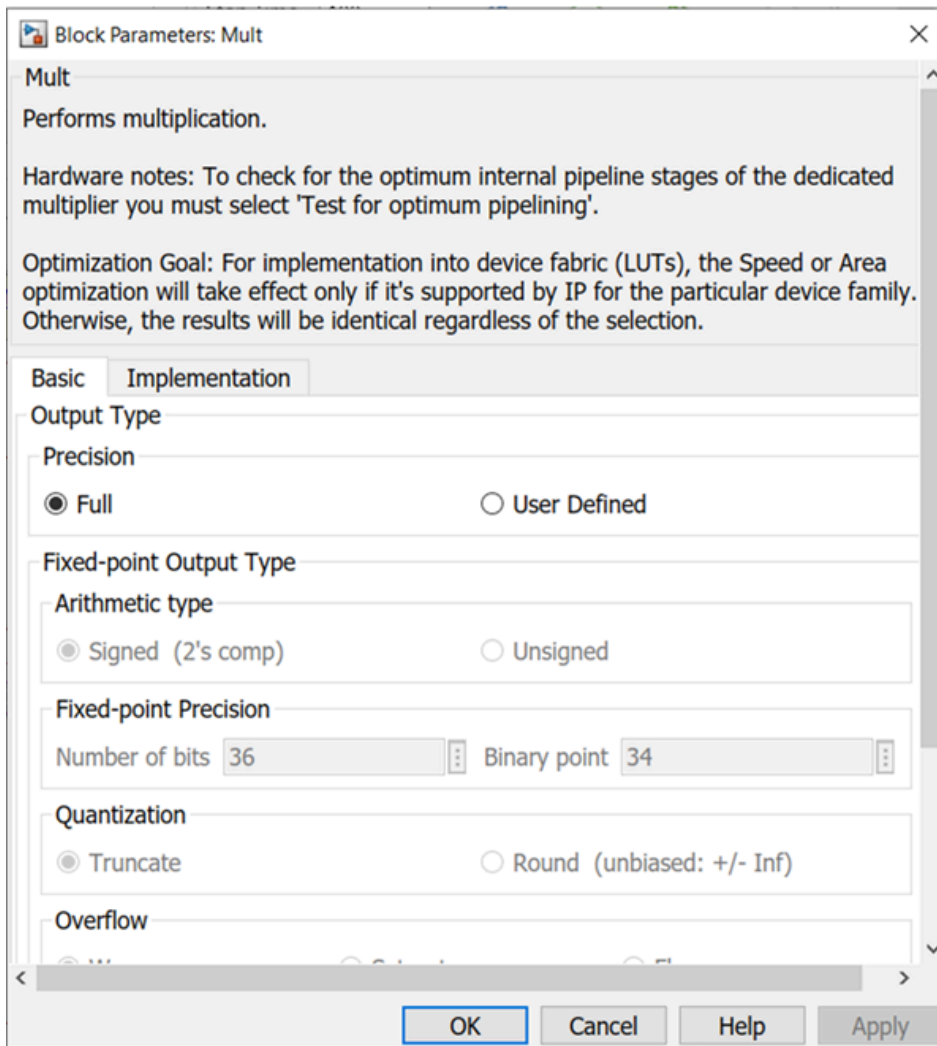
## Troubleshooting Timing Violations

Inserting some registers in the combinational path might give better timing results and might help overcome timing violations if any. This can be done by changing latency of the combinational blocks as explained in the following.

1. Click the violated path from the Timing Analyzer window which opens the violated path as shown in the following figure.



2. Double-click the Mult block to open the Multiplier block parameters window as shown in the following figure.

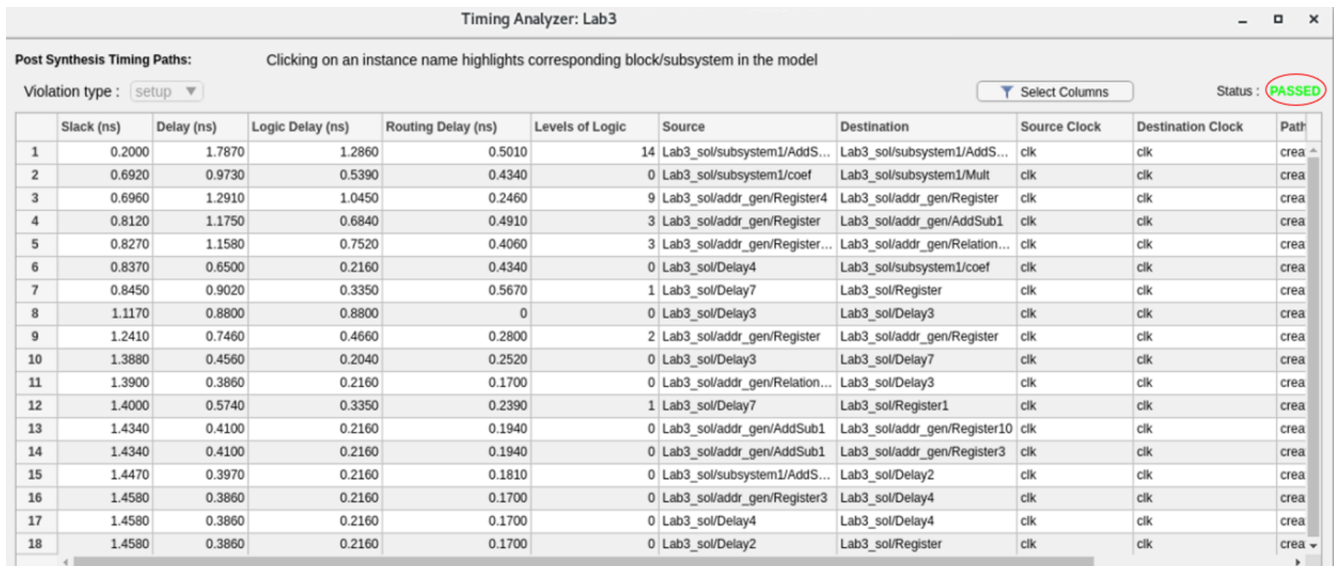


The image shows the 'Block Parameters: Mult' dialog box. It has a title bar with a close button. The main area is divided into two tabs: 'Basic' and 'Implementation'. The 'Basic' tab is selected. Under 'Output Type', there are two radio buttons: 'Full' (selected) and 'User Defined'. Under 'Fixed-point Output Type', there are two radio buttons: 'Signed (2's comp)' (selected) and 'Unsigned'. Under 'Fixed-point Precision', there are two input fields: 'Number of bits' (set to 36) and 'Binary point' (set to 34). Under 'Quantization', there are two radio buttons: 'Truncate' (selected) and 'Round (unbiased: +/- Inf)'. Under 'Overflow', there is a dropdown menu. At the bottom, there are four buttons: 'OK', 'Cancel', 'Help', and 'Apply'.

3. Under Basic tab, change the latency from 1 to 2 and click OK.


4. Double-click the **Model Composer Hub** block, ensure that the Analysis Type is Timing and click **Analyze**.

After the generation completes, it opens the timing Analyzer table as shown in the following figure. Observe the status pass at the top-right corner. It indicates there are no timing violated paths in the design.



The image shows the 'Timing Analyzer: Lab3' window. It has a title bar with standard window controls. Below the title bar, there is a section for 'Post Synthesis Timing Paths:' with a note: 'Clicking on an instance name highlights corresponding block/subsystem in the model'. There is a 'Violation type:' dropdown set to 'setup'. To the right, there is a 'Select Columns' button and a 'Status:' label with a red circle around the word 'PASSED'. Below this is a table with 11 columns: 'Slack (ns)', 'Delay (ns)', 'Logic Delay (ns)', 'Routing Delay (ns)', 'Levels of Logic', 'Source', 'Destination', 'Source Clock', 'Destination Clock', and 'Path'. The table contains 18 rows of data. The 'Path' column shows various paths like 'Lab3\_sol/subsystem1/AddS...', 'Lab3\_sol/subsystem1/Mult', 'Lab3\_sol/addr\_gen/Register4', etc. The 'Status' label 'PASSED' is highlighted with a red circle.

	Slack (ns)	Delay (ns)	Logic Delay (ns)	Routing Delay (ns)	Levels of Logic	Source	Destination	Source Clock	Destination Clock	Path
1	0.2000	1.7870	1.2860	0.5010	14	Lab3_sol/subsystem1/AddS...	Lab3_sol/subsystem1/AddS...	clk	clk	crea
2	0.6920	0.9730	0.5390	0.4340	0	Lab3_sol/subsystem1/coef	Lab3_sol/subsystem1/Mult	clk	clk	crea
3	0.6960	1.2910	1.0450	0.2460	9	Lab3_sol/addr_gen/Register4	Lab3_sol/addr_gen/Register	clk	clk	crea
4	0.8120	1.1750	0.6840	0.4910	3	Lab3_sol/addr_gen/Register	Lab3_sol/addr_gen/AddSub1	clk	clk	crea
5	0.8270	1.1580	0.7520	0.4060	3	Lab3_sol/addr_gen/Register...	Lab3_sol/addr_gen/Relation...	clk	clk	crea
6	0.8370	0.6500	0.2160	0.4340	0	Lab3_sol/Delay4	Lab3_sol/subsystem1/coef	clk	clk	crea
7	0.8450	0.9020	0.3350	0.5670	1	Lab3_sol/Delay7	Lab3_sol/Register	clk	clk	crea
8	1.1170	0.8800	0.8800	0	0	Lab3_sol/Delay3	Lab3_sol/Delay3	clk	clk	crea
9	1.2410	0.7460	0.4660	0.2800	2	Lab3_sol/addr_gen/Register	Lab3_sol/addr_gen/Register	clk	clk	crea
10	1.3880	0.4560	0.2040	0.2520	0	Lab3_sol/Delay3	Lab3_sol/Delay7	clk	clk	crea
11	1.3900	0.3860	0.2160	0.1700	0	Lab3_sol/addr_gen/Relation...	Lab3_sol/Delay3	clk	clk	crea
12	1.4000	0.5740	0.3350	0.2390	1	Lab3_sol/Delay7	Lab3_sol/Register1	clk	clk	crea
13	1.4340	0.4100	0.2160	0.1940	0	Lab3_sol/addr_gen/AddSub1	Lab3_sol/addr_gen/Register10	clk	clk	crea
14	1.4340	0.4100	0.2160	0.1940	0	Lab3_sol/addr_gen/AddSub1	Lab3_sol/addr_gen/Register3	clk	clk	crea
15	1.4470	0.3970	0.2160	0.1810	0	Lab3_sol/subsystem1/AddS...	Lab3_sol/Delay2	clk	clk	crea
16	1.4580	0.3860	0.2160	0.1700	0	Lab3_sol/addr_gen/Register3	Lab3_sol/Delay4	clk	clk	crea
17	1.4580	0.3860	0.2160	0.1700	0	Lab3_sol/Delay4	Lab3_sol/Delay4	clk	clk	crea
18	1.4580	0.3860	0.2160	0.1700	0	Lab3_sol/Delay2	Lab3_sol/Register	clk	clk	crea

 **Note:** a. For quicker timing analysis iterations, post-synthesis analysis is preferred over post-implementation analysis. b. Changing the latency of the block might increase the number of resources which can be seen using Step 2: Resource Analysis in Vitis Model Composer.

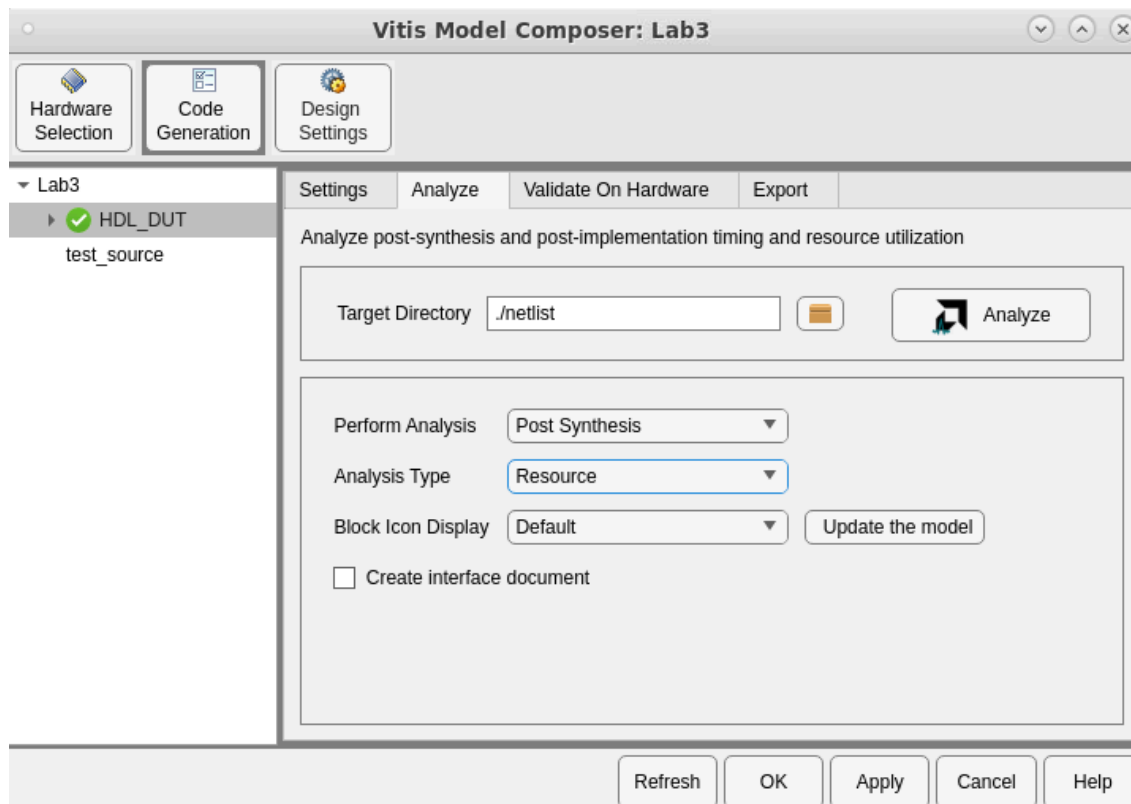
## Step 2: Resource Analysis in Vitis Model Composer


In this step we use same design, `Lab3.s1x`, used for Step 1 but we are going to perform Resource Analysis.

★ **Tip:** Resource Analysis can be performed whenever you generate any of the following compilation types:

- IP catalog
- Hardware Co-Simulation
- Synthesized Checkpoint
- HDL Netlist

1. Double-click the **Model Composer Hub** block in the Simulink model. On the HDL Settings tab, ensure that one of the Compilation Types listed above is selected.
2. In the Analyze tab, set the Perform Analysis field to **Post Synthesis** and Analysis Type type field to **Resource**.



 **Note:** In order to see accurate results from Resource Analyzer Window it is recommended to specify a new target directory rather than use the current working directory.

3. In the Model Composer Hub dialog box, click **Analyze**.

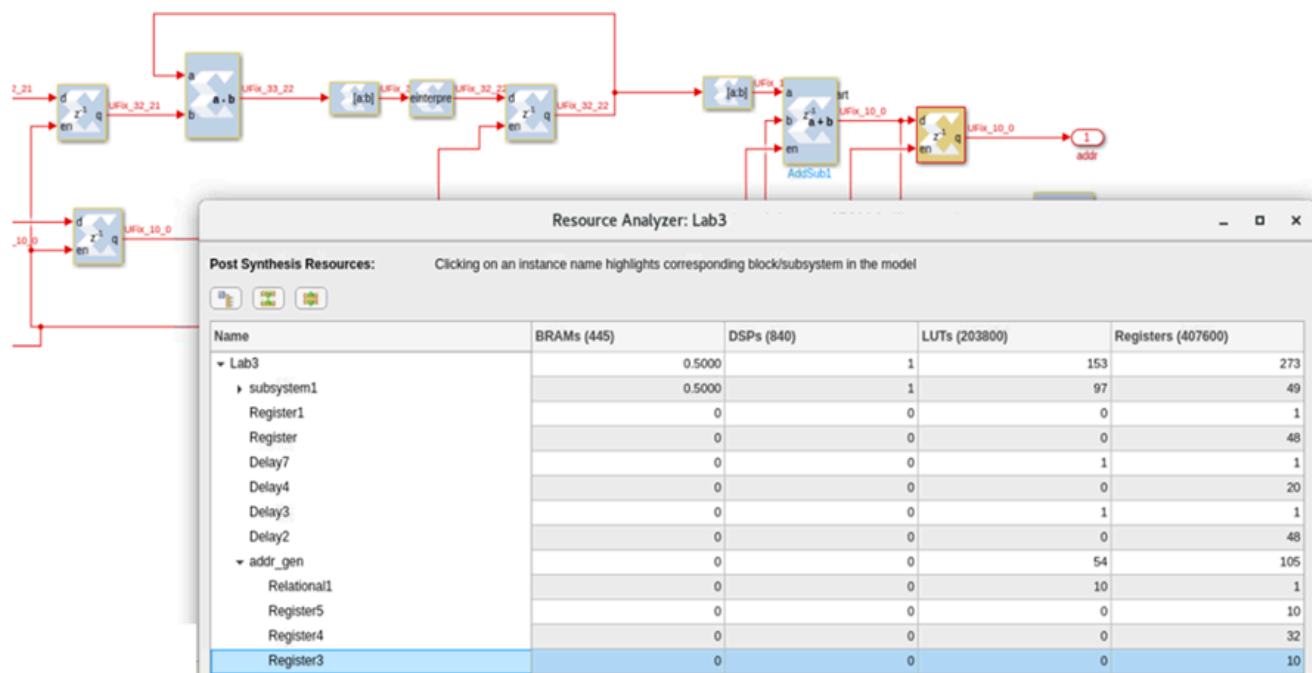
Model Composer processes the resource utilization data and displays a Resource Analyzer window with resource utilization information.

Post Synthesis Resources: Clicking on an instance name highlights corresponding block/subsystem in the model					
Name	BRAMs (445)	DSPs (840)	LUTs (203800)	Registers (407600)	
▼ Lab3	0.5000	1	153	273	
▶ subsystem1	0.5000	1	97	49	
Register1	0	0	0	1	
Register	0	0	0	48	
Delay7	0	0	1	1	
Delay4	0	0	0	20	
Delay3	0	0	1	1	
Delay2	0	0	0	48	
▶ addr_gen	0	0	54	105	

Each column heading (for example, BRAMs, DSPs, or LUTs) in the window shows the total number of each type of resources available in the AMD device for which you are targeting your design. The rest of the window displays a hierarchical listing of each subsystem and block in the design, with the count of these resource types.

You can cross probe from the Resource Analyzer window to the Simulink model by clicking a block or subsystem name in the Resource Analyzer window, which highlights the corresponding Vitis Model Composer HDL block or subsystem in the model.

Cross probing is useful to identify blocks and subsystems that are implemented using a particular type of resource. The block you have selected in the window will be highlighted yellow and outlined in red. If the block or subsystem you have selected in the window is within an upper-level subsystem, then the parent subsystem is highlighted in red in addition to the underlying block as shown in the following figure.



**!! Important:** If the Resource Analyzer window or the Timing Analyzer window opens and no information is displayed in the window (table cells are empty), double-click the Model Composer Hub block and set the Target Directory to a new directory, that is, a directory that has not been used before. Then run the analysis again.

## Summary

---

In this lab you learned how to use timing and resource analysis inside Model Composer which, in turn, invokes Vivado synthesis to collect the information for the analysis. You also learned how to identify timing violated paths and to troubleshoot them for simple designs.

Copyright 2024 Advanced Micro Devices, Inc.

Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at

<http://www.apache.org/licenses/LICENSE-2.0>



Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.