

Design Analysis and Floorplanning Tutorial

PlanAhead Design Tool

This tutorial document was last validated using the following software version: ISE Design Suite 14.1

If using a later software version, there may be minor differences between the images and results shown in this document with what you will see in the Design Suite.

UG676 (v14.1) May 8, 2012





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, 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..

Table of Contents

Tutorial Objectives	4
Software Requirements	5
Hardware Requirements	5
Tutorial Design Description	5
Locating Tutorial Design Files	5
Step 1: Viewing the Device Resources and Clock Regions	6
Step 2: Exploring the Logical Netlist Hierarchy	12
Step 3: Displaying Design Resources Statistics	14
Step 4: Running Design Rule Checks (DRC)	16
Step 5: Performing Timing Analysis	19
Step 6: Implementing the Design	24
Step 7: Analyzing the Timing	28
Step 8: Highlighting Module-Level Placement	35
Step 9: Exploring Connectivity	37
Step 10: Using Placement Constraints	40
Step 11: Viewing Hierarchical Connectivity	43
Step 12: Using the Search Capability to View Clock Domains	48
Step 13: Floorplanning Timing-Critical Hierarchy	51
Conclusion	57

Design Analysis and Floorplanning Tutorial

This tutorial introduces some of the capabilities and benefits of using the Xilinx® PlanAhead™ software for designing high-end FPGAs. The document contains a series of step-by-step exercises that highlight methods to achieve and maintain better performing designs in less time. The steps detail the following:

- Pre-implementation design and analysis capabilities
- Implementation exploration features
- Implementation results floorplanning

Note: This tutorial covers a subset of the features of the PlanAhead software product bundled with the ISE® Design Suite. Additional features are covered in detail in other tutorials.

Tutorial Objectives

This tutorial takes you through the various analysis, floorplanning, and implementation features of the PlanAhead software. The exercises cover the following topics:

- Analyze device utilization statistics to target alternate devices and choose the optimal device.
- Run Design Rule Checks (DRC) to quickly resolve constraint conflicts that would otherwise cause implementation errors.
- Use the Netlist, Logic Hierarchy, and Schematic views to explore logic.
- Perform a quick estimation of timing performance to assess design feasibility and identify potential problem areas.
- View, modify, or create constraints in the design.
- Analyze the design hierarchical connectivity and data flow, as well as identify critical logic connectivity and clock domains.
- Floorplan timing-critical logic to improve timing.

When using this tutorial, focus on the processes and functionality of the PlanAhead software to determine how you can take advantage of these features in your designs.

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: *Installation and Licensing Guide* (UG798) at http://www.xilinx.com/support/documentation/sw_manuals/xilinx14_1/iil.pdf.

Hardware Requirements

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

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

Locating Tutorial Design Files

Copy the files from the ISE software installation area:

`<ISE_install_area>/ISE_DS/PlanAhead/testcases/PlanAhead_Tutorial.zip`

Extract the zip file contents into any write-accessible location. The unzipped PlanAhead_Tutorial data directory is referred to in this tutorial as `<Extract_Dir>`.

The tutorial sample design data is modified while performing this tutorial. A new copy of the original PlanAhead_Tutorial data is required each time you run the tutorial.

Step 1: Viewing the Device Resources and Clock Regions

1. Launch the PlanAhead Software.

- On Windows, select the Xilinx PlanAhead 14 Desktop icon or click **Start > All Programs > Xilinx Design Tools > ISE Design Suite 14.1 > PlanAhead > PlanAhead**.
- On Linux, go to the following directory
<Extract_Dir>/PlanAhead_Tutorial/Projects, and type **planAhead**.

The PlanAhead environment opens.

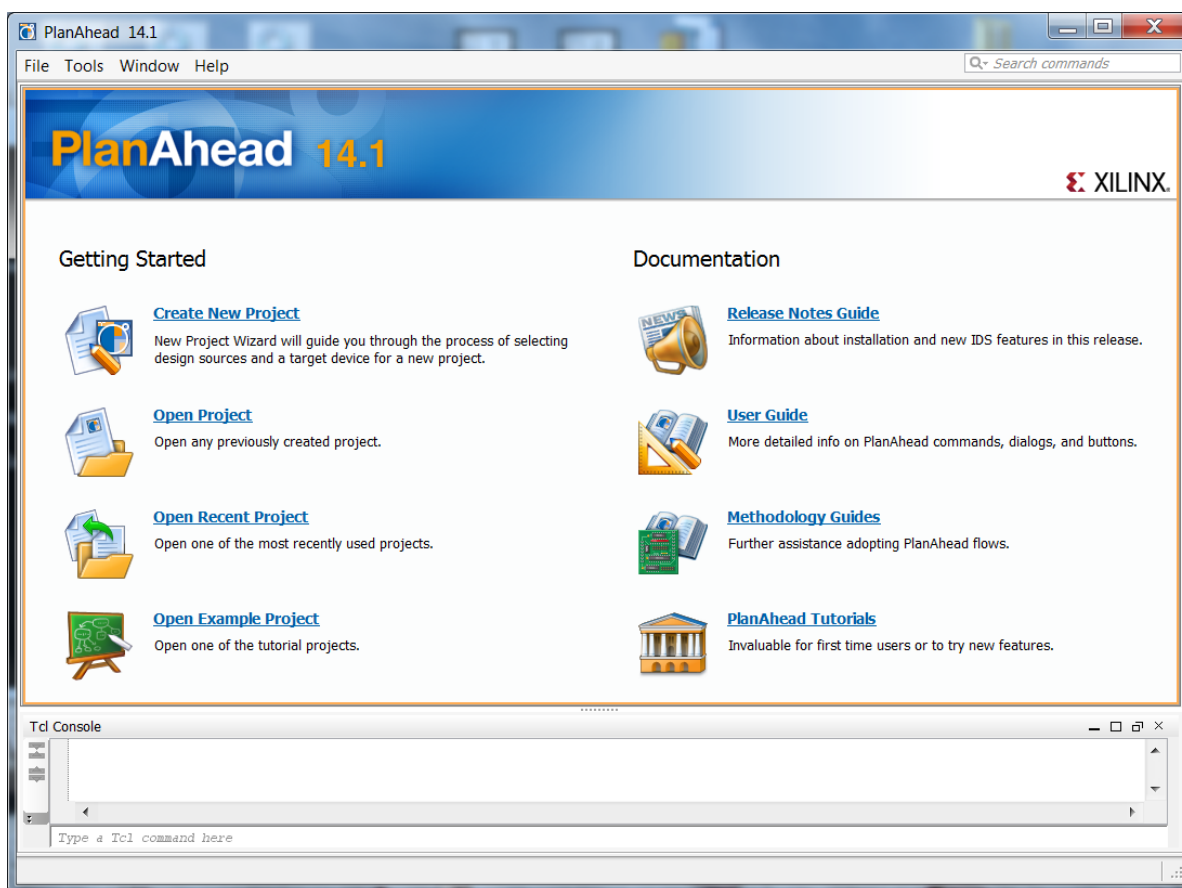


Figure 1: PlanAhead Getting Started Page

The PlanAhead Getting Started page contains links to open or create projects and to view documentation.

2. Select **File > Open Project**, browse to
<Extract_Dir>/PlanAhead_Tutorial/projects/project_cpu_hdl/ and
open **project_cpu_hdl.ppr**.

The PlanAhead environment opens with the project open.

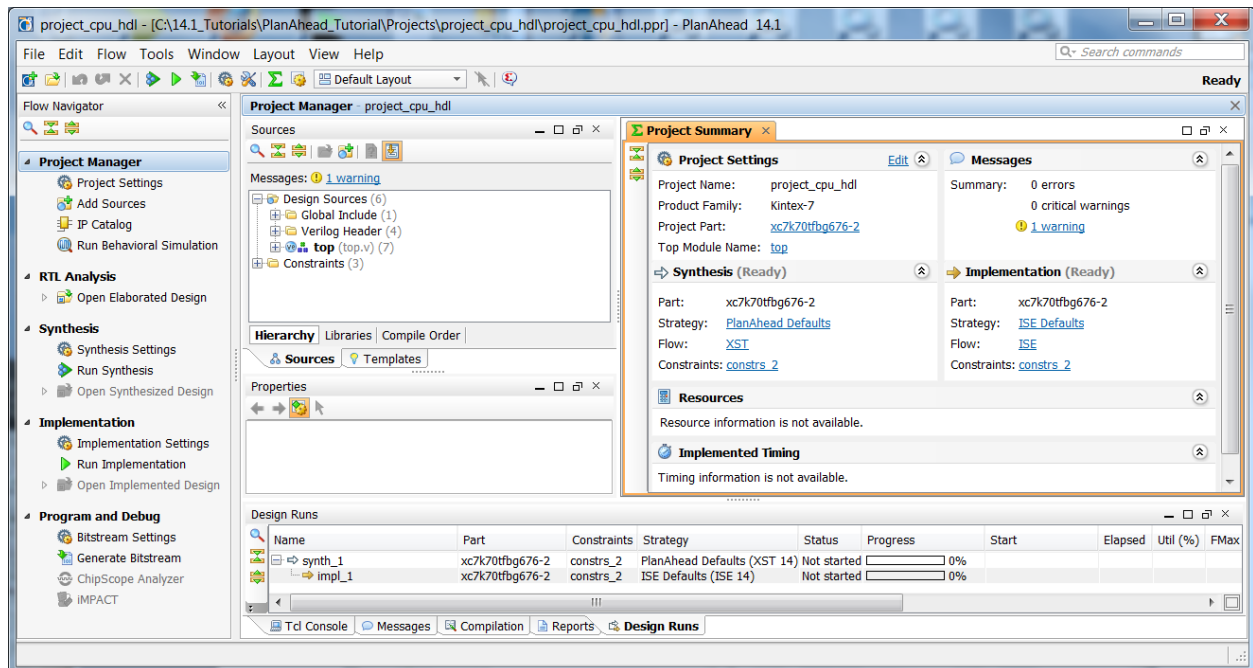



Figure 2: PlanAhead Environment

- In the Sources View, expand the Constraints folder to ensure `constrs_2` is active, as shown in the following figure. If needed, select **constrs_2**, right-click and select **Make active**. Use the **Collapse All** button , if needed.

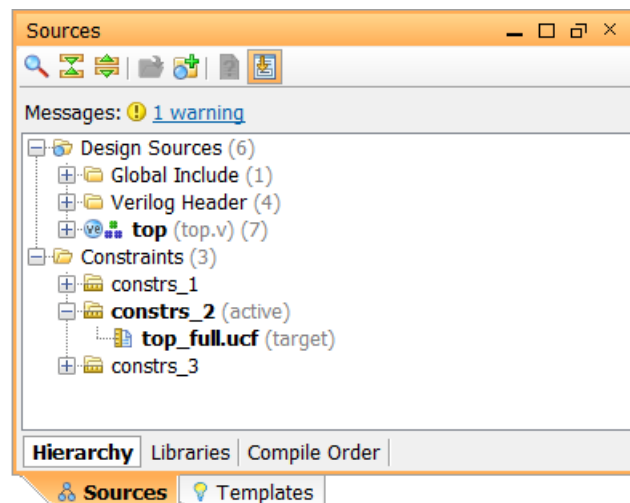


Figure 3: Sources View with `constrs_2` Active

- Click the Design Runs view and examine the runs information. Multiple runs are managed in the Design Runs view.

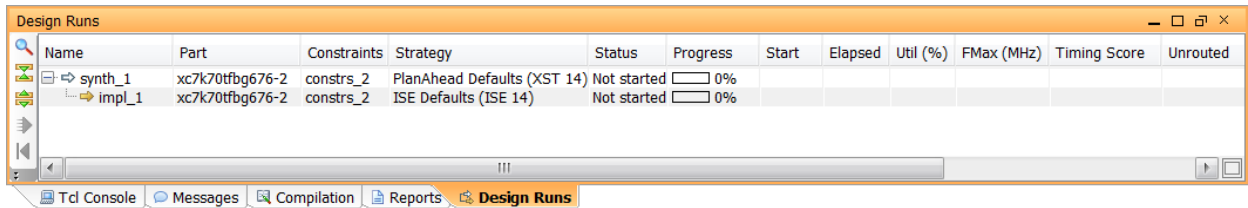


Figure 4: Design Runs View

5. In the Flow Navigator, select **Run Synthesis** to synthesize the design.
6. Watch the Compilation view for run status. It takes several minutes for the run to complete.
7. After synthesis completes, select **Open Synthesized Design** from the Synthesis Completed dialog and click **OK**.

The Synthesized Design opens with the netlist and `constrs_2` constraints.

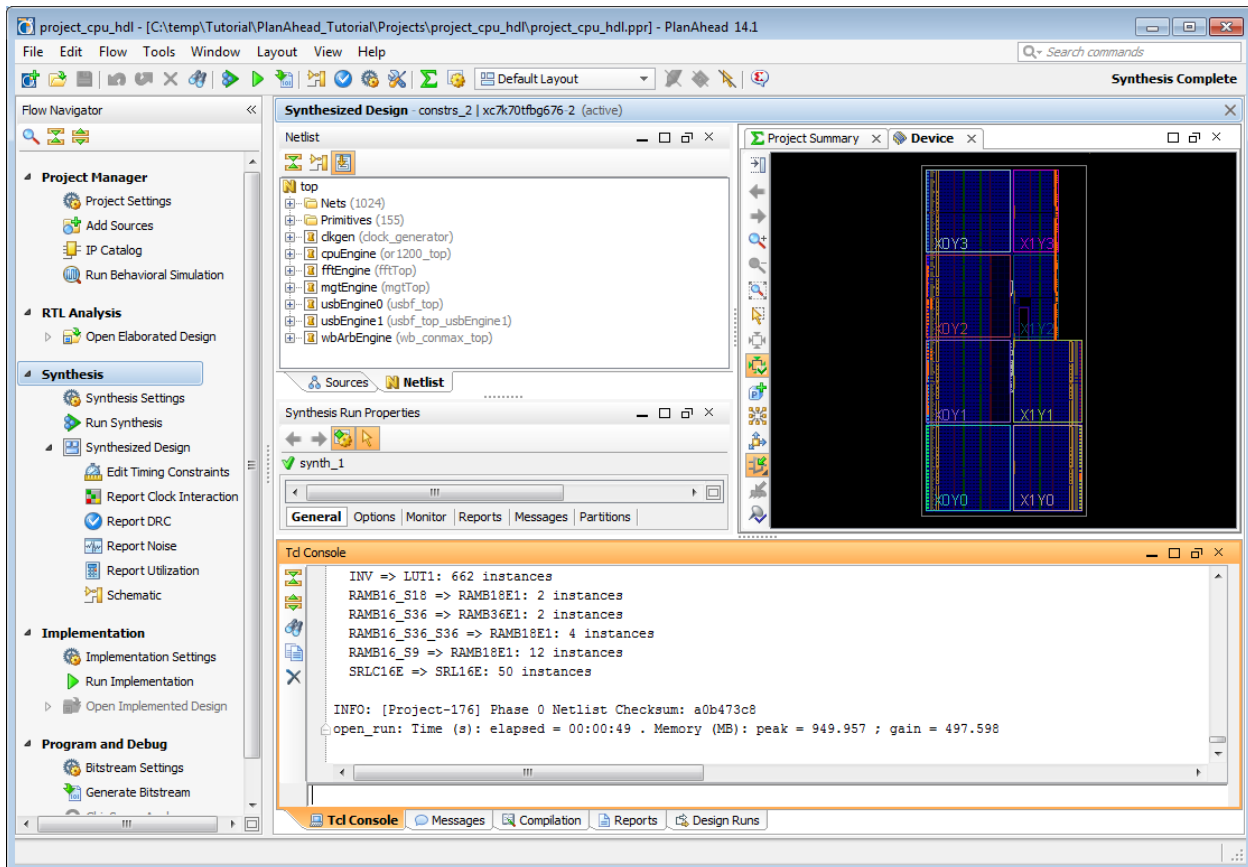


Figure 5: CPU Netlist in Netlist Design

8. In the Device view, use **Zoom Area** to examine various device resources as follows:

- Click in the upper left corner of the device, and drag down and to the right. This will draw a Zoom Area rectangle.
- Repeat to generate an area small enough to see the device resources.

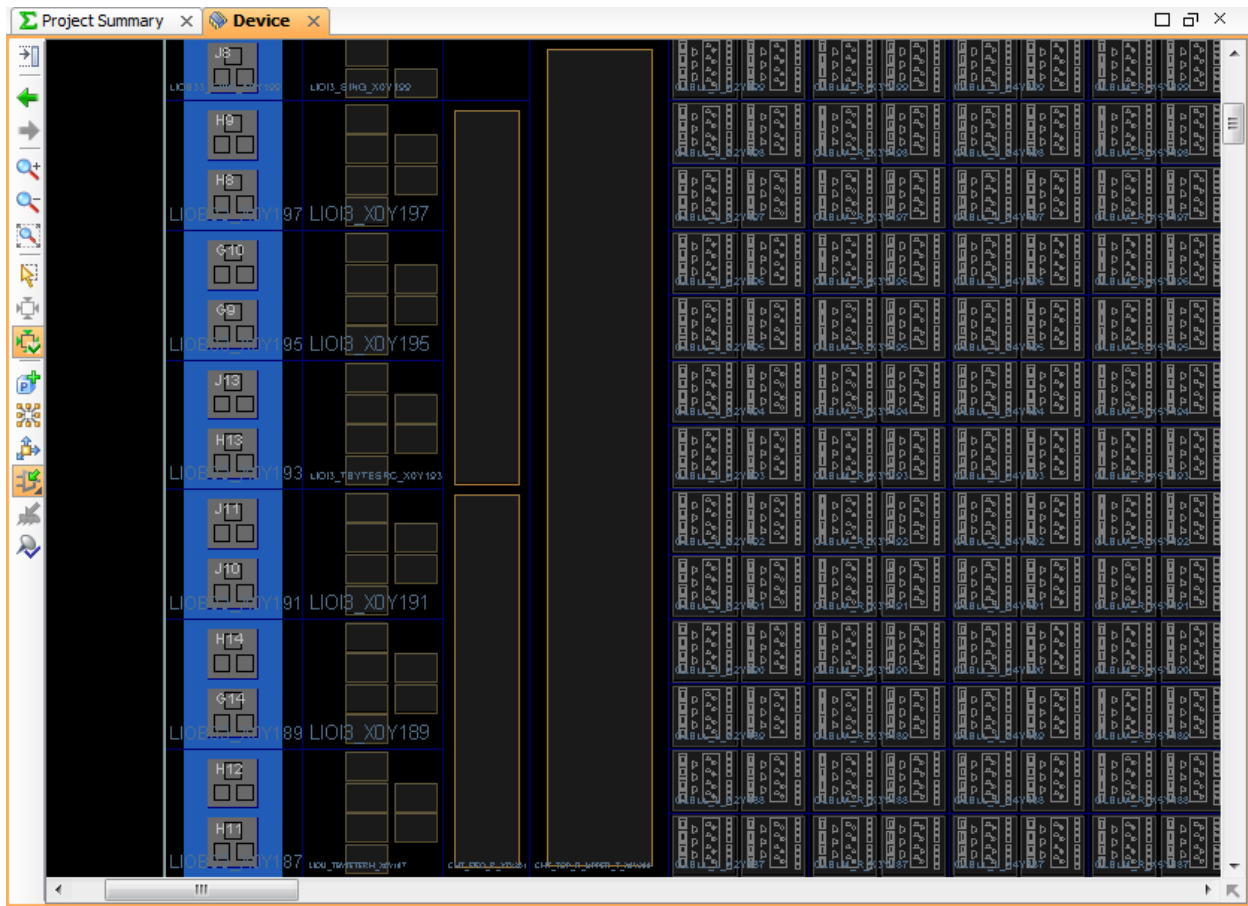


Figure 6: Zooming into the Device to View CLBs and Logic Gate Sites

9. Move the cursor slowly over the various device resources to view the tooltips, and take note of the following:
 - SLICE coordinates appear in the status bar at the bottom right of the PlanAhead software main window.
 - Hover the mouse over a site to show a tooltip.
 - I/O port locations and I/O buffer assignments appear inside I/O banks.
 - Tall dark red tiles indicate a RAMB36 site that can also hold two RAMB18s or a FIFO.
 - Tall green tiles indicate sites for DSP48s.
 - Square blue tiles are a CLB containing SLICES.

Displaying Site Information in the Site Properties View

10. Select several different types of Sites and view the Site and BEL Property information.
11. Click in the Device view and drag the cursor up and to the left, or press **Ctrl+O** to Zoom Fit the Device view to see the whole device.

Opening the Clock Regions View

The Device view shows the clock regions, which you can use to help guide floorplanning. The Clock Region view lists all the clock regions in the device. The Clock Region Properties view displays clock region information to indicate potential clock contentions prior to implementation. Viewing clock domains is covered later in the tutorial.

12. Select **Window > Clock Regions** to display the Clock Regions view.
13. Select one of the clock regions listed in the table.

The clock region is highlighted in the Device view.

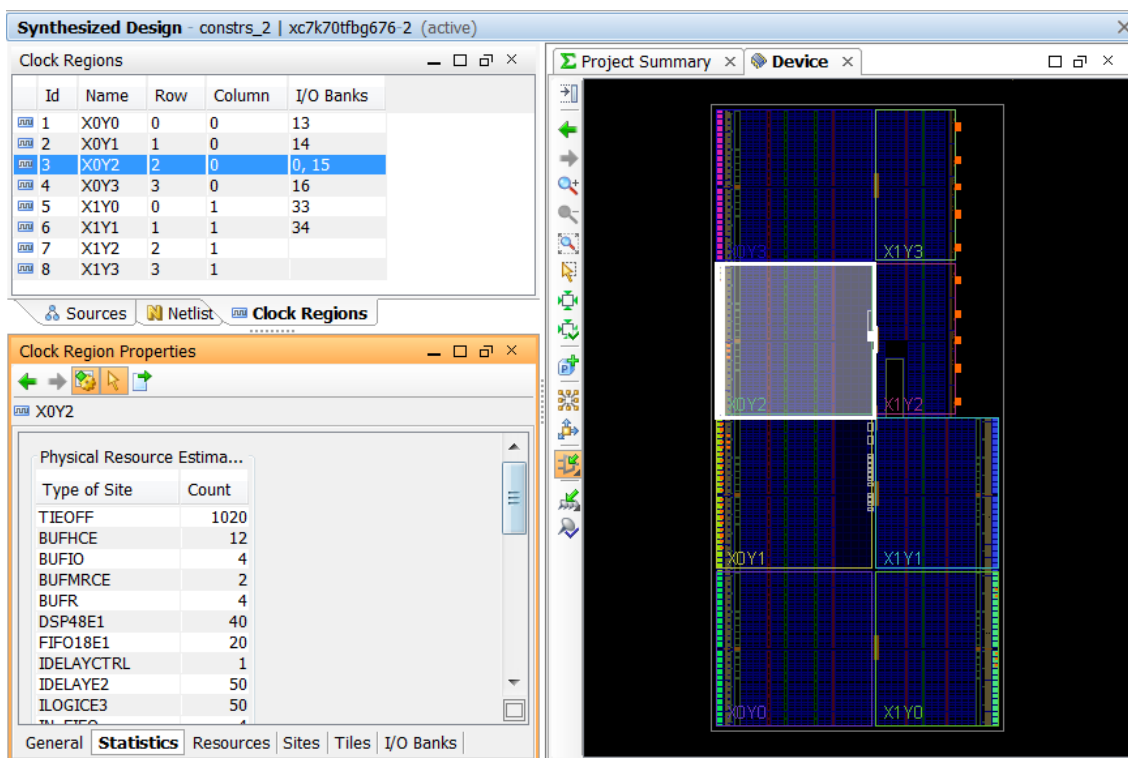




Figure 7: Viewing Clock Region Properties

14. Scroll to examine the **Clock Region Properties** view. If the properties window is not visible, select **Window > Properties**.
15. Click the **Statistics** tab, and scroll to examine the logic contents.

Exploring Clock Region Resources

16. In the Properties view, click the **Resources** tab to see the locations of the BUFR and IDELAYCTRL sites.
17. In the Resources list, select one of the BUFRs, and notice it is highlighted in the Device view. (It might be helpful to maximize the view.)
18. In either the Device view or the Properties view, right-click and select **Mark** to mark the site.
19. Select **Fit Selection** from the Device view side toolbar  to view the selected objects.
Note: If Auto Fit Selection is enabled, the Device view is fit to the BUFR on selection.
20. Select the **Unmark All** main toolbar button  to remove the mark.
21. In the Device view, use **Zoom Fit** to fit the whole device.
22. In the Clock Region Properties view, click the **I/O Banks** tab to examine the I/O Banks related to the clock region.
23. In the Clock Region Properties view, select one of the I/O banks. It is highlighted in the Device view as well.
24. Switch to the I/O Planning view layout if this is not the current view, shown in the following figure. To do so, select **Layout > I/O Planning**, or use the layout selector on the toolbar menu.

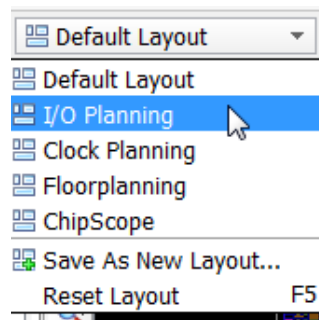


Figure 8: Selecting the I/O Planning Layout


The view layout has changed to make it easier for pin planning. The Package view and Package Pins view are now open at the bottom on the PlanAhead environment.

The I/O bank is highlighted in both the Device and Package Pins views.

25. In the Layout pulldown, change the tool layout back to **Default Layout**.

Step 2: Exploring the Logical Netlist Hierarchy

Using the Netlist View to Explore the Design Hierarchy

1. Click the Netlist view tab, and if needed, click the **Collapse All** button .
2. Expand the `cpuEngine` module.

The Netlist should now look similar to the one shown in the following figure.

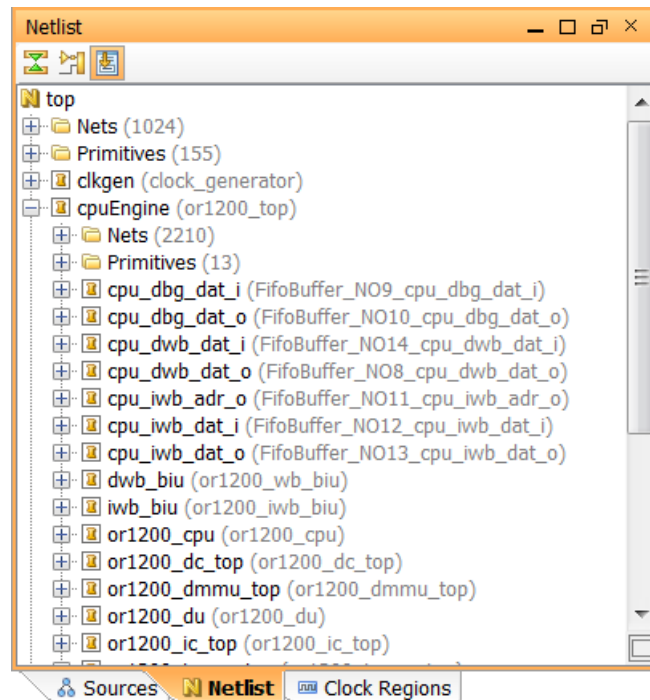



Figure 9: Expanding the Netlist View

Note: There is a `Primitives` folder in the Netlist view that contains the top-level instances of each module.

3. Expand the `Primitives` folder. These are the instances immediately under the `cpuEngine` level, not in any sub-module.
4. Expand the `Nets` folder. These are the nets immediately under `cpuEngine`.
5. In the Netlist view, click the **Collapse All** button  to simplify the netlist view.

Selecting Netlist Modules and Viewing Where the Logic Resides in the Design Hierarchy

6. In the Netlist view, expand the `usbEngine0` module.

7. Select the **u4** module.
8. Right-click and select **Show Hierarchy**, or press **F6**.

The Hierarchy view opens in the Workspace.

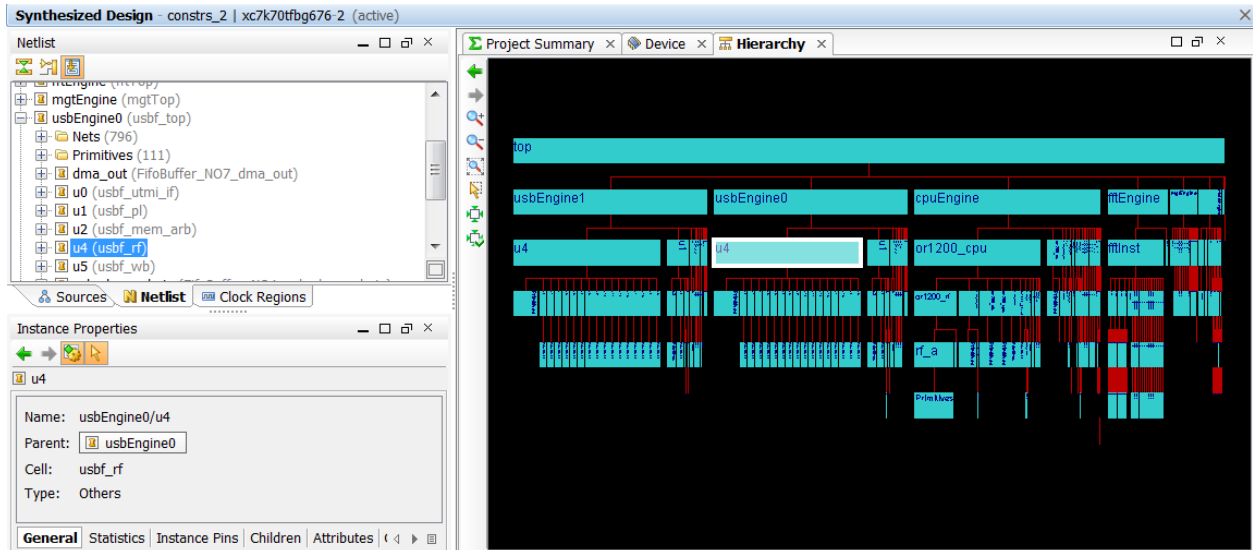


Figure 10: Displaying the Instance Hierarchy View

The Hierarchy view displays the hierarchical relationship of the modules and the relative sizes. The selected logic module displays in the Hierarchy view, which is useful to visualize the module location and relative size prior to floorplanning. You can select modules directly from this view for floorplanning. Logic selected in other views is highlighted in the Hierarchy view.

9. Select any module in the Hierarchy view, and notice that the module is selected in the Netlist and other views, as well. Selection works across views.
10. Click the **Unselect All** main toolbar button or press **F12**.
11. Close the Hierarchy view by clicking the **X** button in the view tab.

Step 3: Displaying Design Resources Statistics

The PlanAhead software utilization analysis provides design statistics to help determine the optimal device for the design. The tool helps you see how the logic resources are split between modules. It is easy to explore multiple device types to determine the best overall utilization and performance estimations.

Viewing the Resource Estimates of the Entire Design

1. Select **Report Utilization** from the Flow Navigator.
2. Scroll down the **Report Utilization** view.

The Report Utilization view shows resource utilization by logic type and per level of design hierarchy.

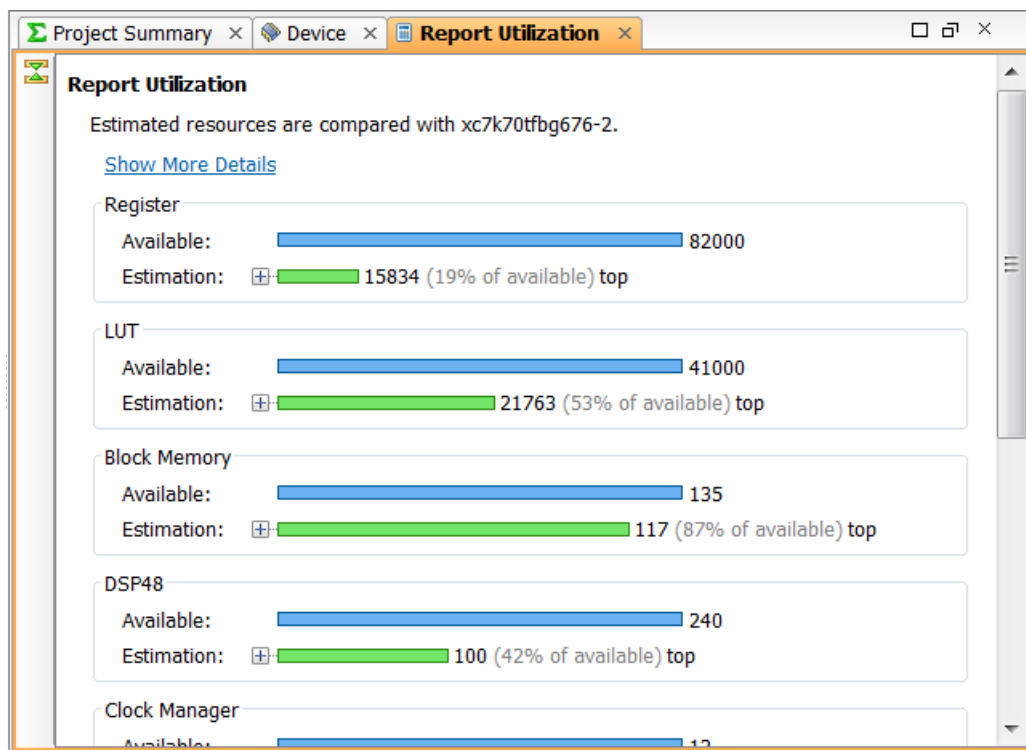



Figure 11: Resource Estimation View

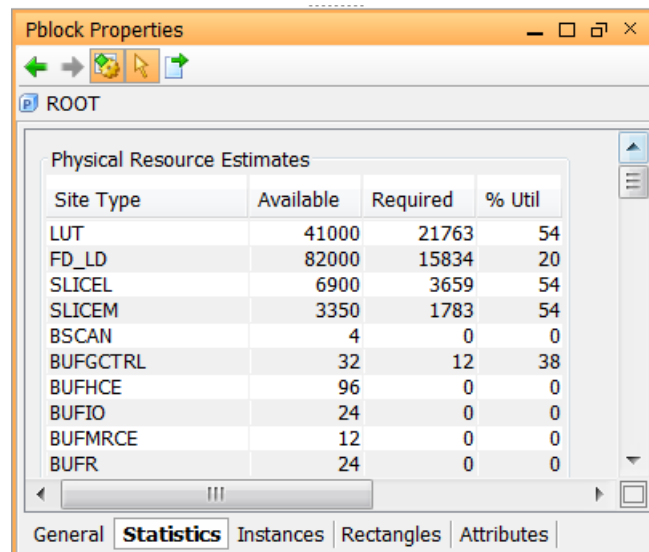
3. In the Report Utilization view, click the expand widget  next to Estimation under Block Memory (block RAM). You will see that `usbEngine0` and `usbEngine1` are the two largest consumers of block RAM.
4. Expand some of the other resources.
5. Select **usbEngine1** from the block RAM Estimation section.

6. Notice that `usbEngine1` is selected in the Netlist view as well.
7. Close the Report Utilization view.
8. Select **Layout > Reset Layout** to restore the default window sizes.

Exploring the Resources in a More Detailed View

This step shows you another way to see the resources in the part. The following view gives more detailed information.

9. Select **Window > Physical Constraints** to open the Physical Constraints view.
10. In the Physical Constraints view, select the **ROOT** design.
11. In the **Pblock Properties** view, select the **Statistics** tab, if necessary.
12. View the **Physical Resource Estimates** displayed in the Pblock Properties view, as shown in the following figure.



Site Type	Available	Required	% Util
LUT	41000	21763	54
FD_LD	82000	15834	20
SLICEL	6900	3659	54
SLICEM	3350	1783	54
BSCAN	4	0	0
BUFGCTRL	32	12	38
BUFHCE	96	0	0
BUFIO	24	0	0
BUFMRCE	12	0	0
BUFR	24	0	0

Figure 12: Viewing Design Resource Statistics

13. Scroll through the design statistics displayed in the Pblock Properties view.

Note: The Statistics tab displays the device resource utilization for each type of logic element, carry chain count, longest length, Clock Report, I/O utilization, and Primitive instance and interface net counts.

If this design had any RPMs, the RPM count and maximum size information would be shown as well

Step 4: Running Design Rule Checks (DRC)

Xilinx recommends running the Design Rule Checker (DRC) before implementation to check for some common design issues. The ISE implementation tool DRCs are used for design sign-off, and take precedence over the DRCs in the PlanAhead software.

Running the Design Rule Checker (DRC)

1. Select **Report DRC** in the Flow Navigator.

The Run DRC dialog box opens.

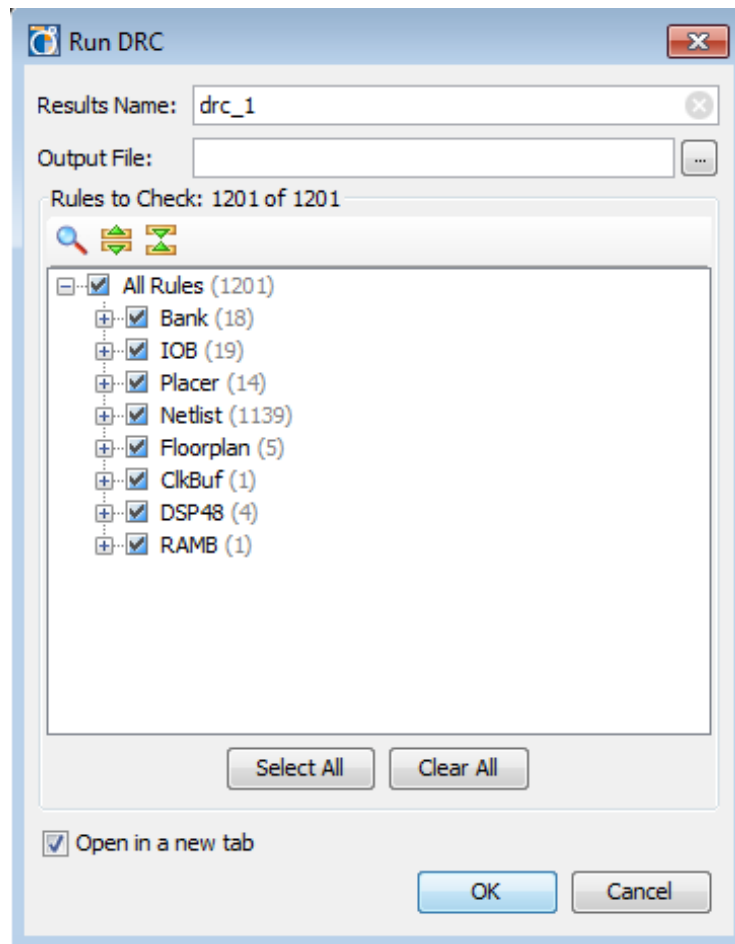


Figure 13: Running DRC

2. Click **OK** to run all rule checks.

The DRC Results view opens and indicates that there are several DSP48 and block RAM warnings.

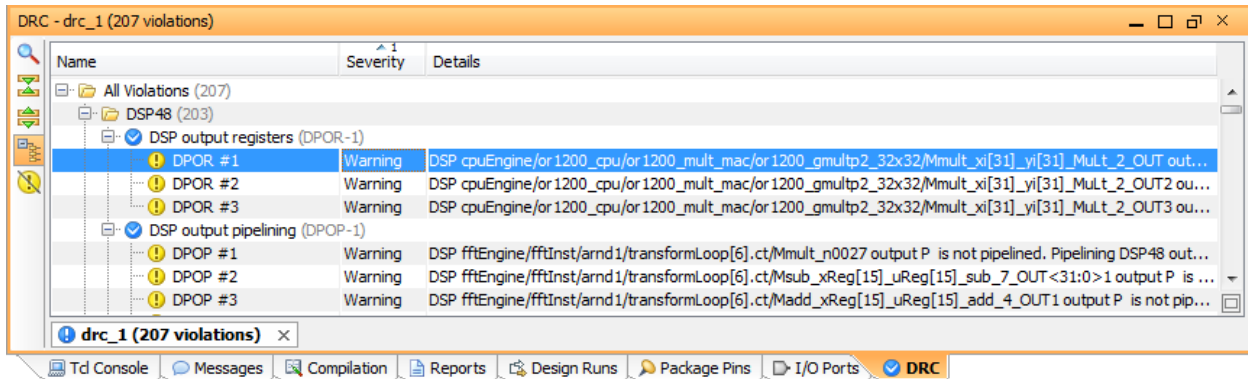


Figure 14: Design Rule Violations

Errors, warnings and information messages display in the DRC Results view.

- Information messages are indicated by a blue icon.
- Warnings are indicated by a yellow icon.
- Critical Warnings are indicated by an orange icon.
- Errors are indicated by a red icon.

DRC errors will not prevent Implementation from being run.

3. In the DRC Results view, click the **DPOR #1** Warning.

The Violation Properties view opens and describes the violation.

4. Right click in the Violations Properties view. Note: by default the tool will automatically select the instances in the DRC message.

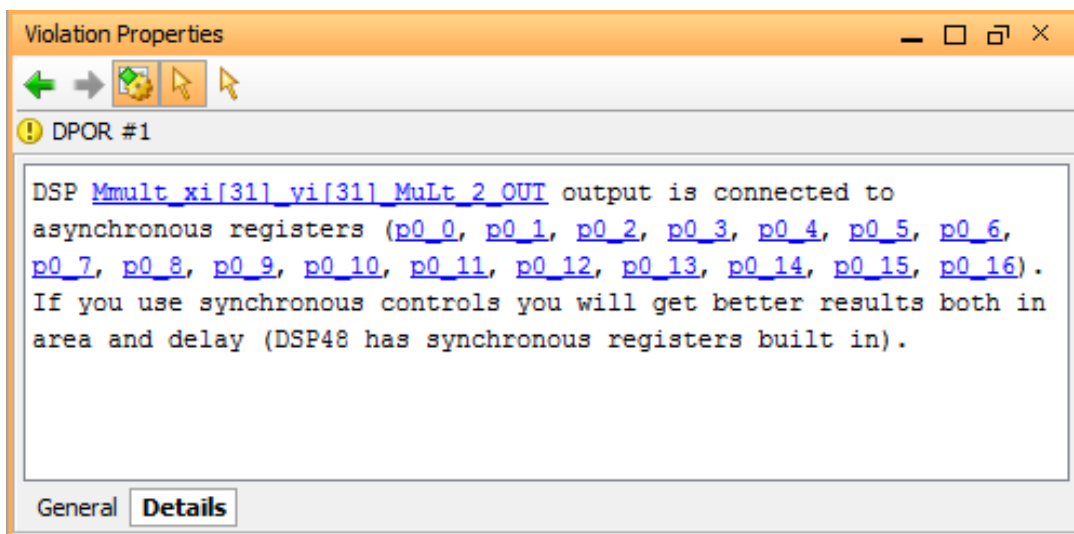


Figure 15: Violations Properties Window

5. Select the **Netlist** view.

Notice that `Mmult_xi[31]_yi[31]_MuLt_2_OUT` and the related flip-flops are selected.

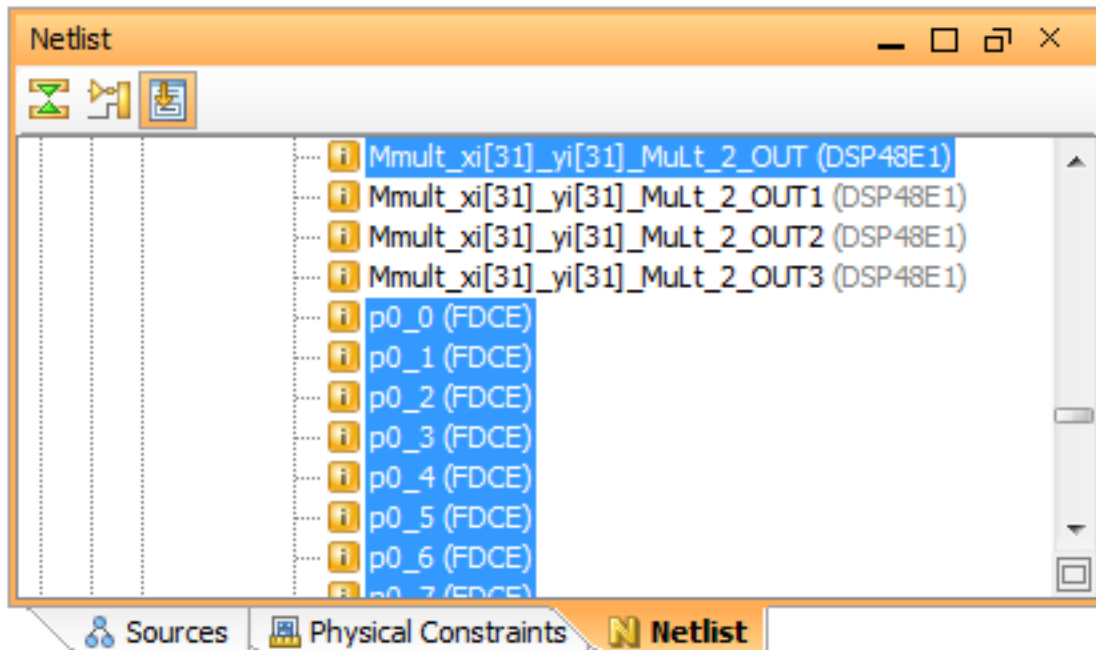


Figure 16: Instance Selected in Netlist View

6. In the DRC Results view, scroll down the list of violations.
7. Close the DRC Results view by clicking the **X** button in the tab.

Step 5: Performing Timing Analysis

Before running implementation it can be helpful to perform an early static timing estimation, including estimated route delays, to determine the feasibility of the design timing constraints. The PlanAhead timing engine provides an estimate of what the route delays will be and does not report on whether the design has met timing (post-implementation). In this step, you will investigate timing before implementing the design.

Note: Even with a fully placed design, only the implementation TRCE tool indicates whether a design has met timing when an implementation run is imported. The PlanAhead Report Timing and the Slack Histogram commands do not show whether or not a design has met timing.

Analyzing Timing End Points in the Slack Histogram

The Slack Histogram groups the end points into bins based on slack and presents a histogram with the timing information. The histogram makes it possible to visualize how many end points have tight timing versus those that have a margin.

1. Select **Tools > Timing > View Slack Histogram**.
2. Change the Number of bins to **20**.

The Generate Slack Histogram for Endpoints dialog box should look like the following:

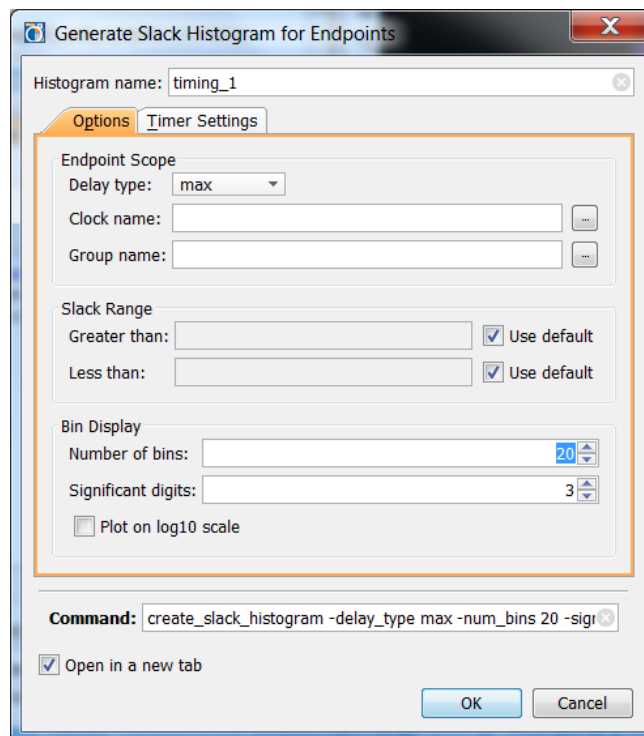


Figure 17: The Generate Slack Histogram for Endpoints Dialog Box

3. Click the **Timer Settings** tab.

4. Inspect the settings.

The Timer Settings tab lets you change how the tool accounts for route delay. The timer can estimate route delay or assume there is no interconnect routing delay.

5. Click **OK**.

A histogram shows the end points in bins.

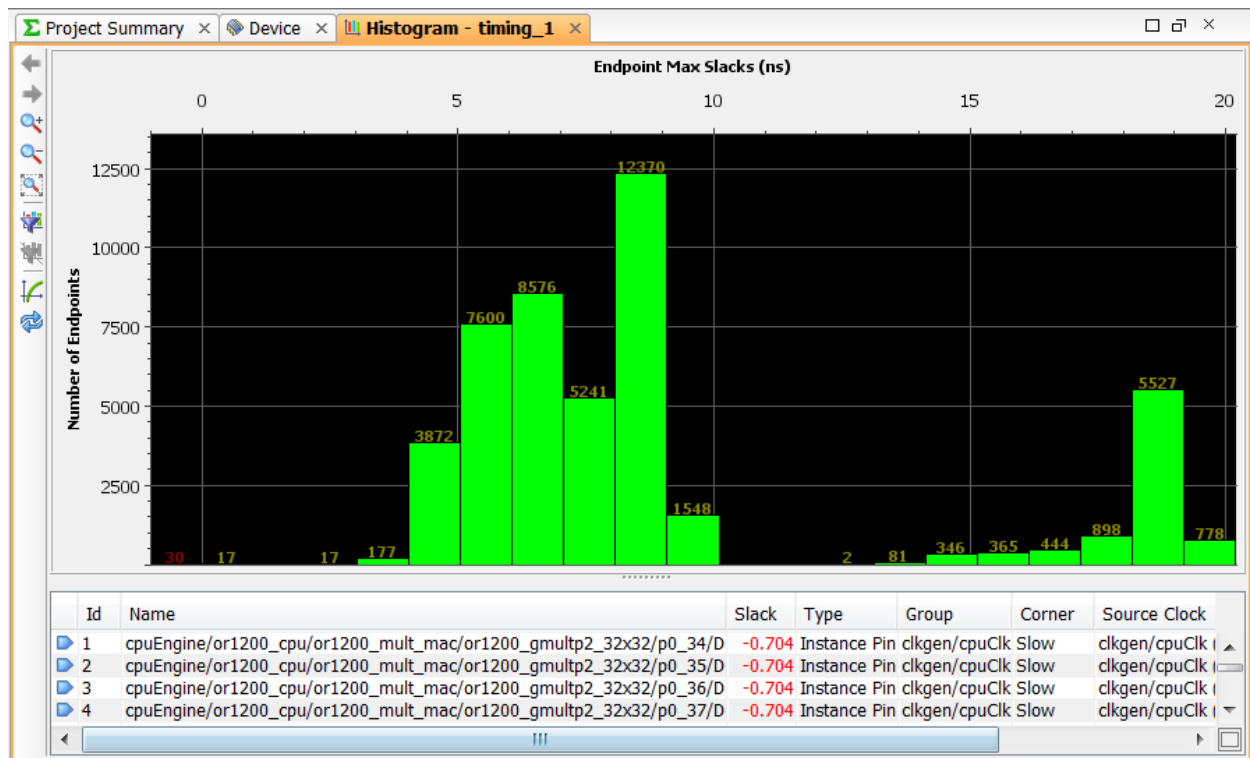


Figure 18: Slack Histogram

Notice that the failing end points display in the red bin on the left side of the slack 0 column separator. These end points are expected to have timing problems.

6. Click the various bins.

The end points in the lower view filter as the bins are selected. The slack range is also updated.

7. Select the leftmost bin, which is the bin with endpoints expected to fail timing.

8. Scroll down to inspect the end points.

The look for the failing timing end points. The tool will show red slack for the end points that are estimated to not meet timing.

9. Close the Histogram view.

Running Timing Analysis and Analyzing the Initial Results and Information in the Timing View

10. Select **Tools > Timing > Report Timing**.

The Report Timing dialog box opens to the Targets tab, in which you can specify timing start and end points.

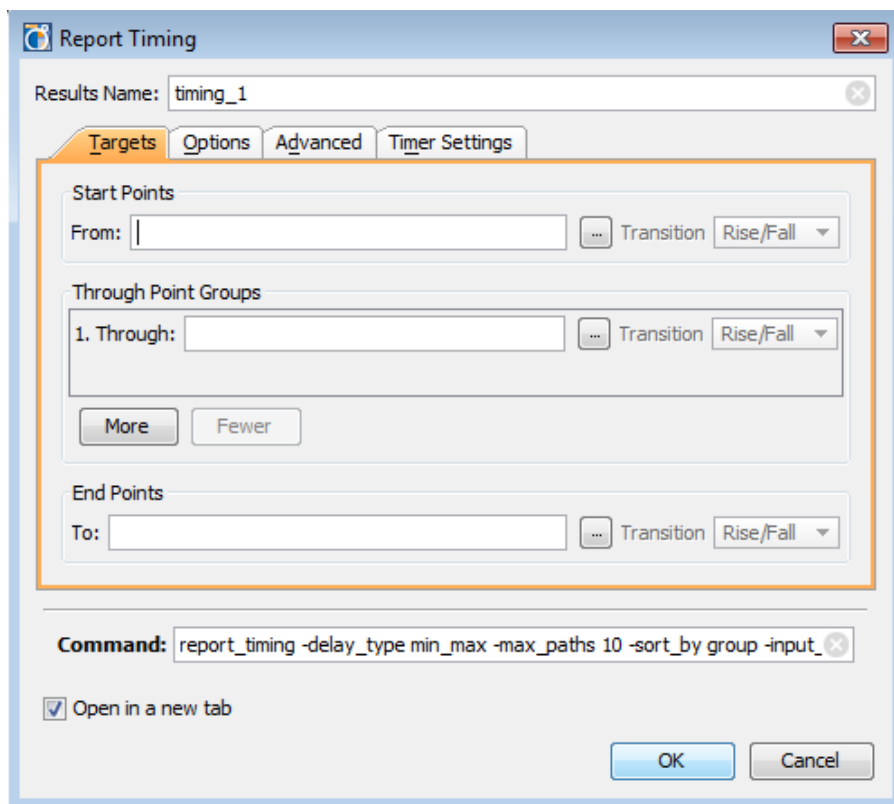


Figure 19: Running Timing Analysis

11. Select the **Options** tab.

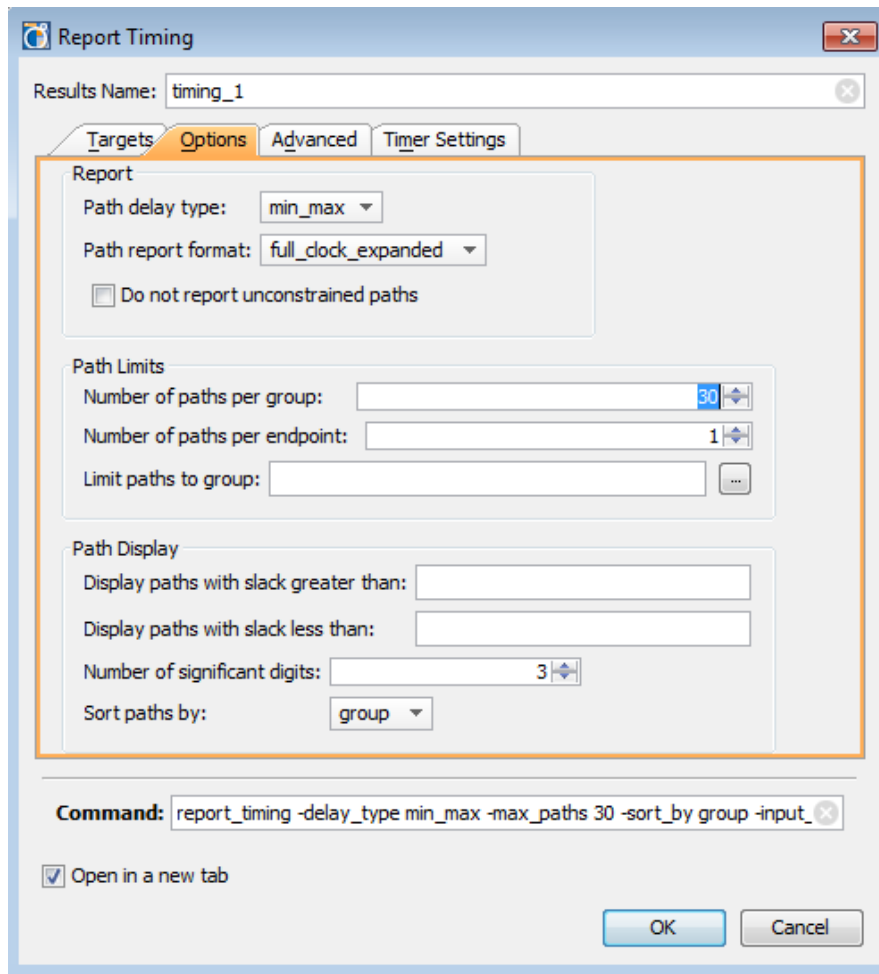


Figure 20: Report Timing Options Tab

12. In the **Number of paths per group** field, type 30.
13. Click **Advanced** and **Timer Settings**, and inspect the tabs. Do not change anything.
The Interconnect pulldown in Timer Settings has two values: Estimated, and None. The timing engine differs from the ISE timing engine and it cannot give you exact route delays. The routing numbers are estimated based on a routing delay model.
14. Click **OK** to run analysis.
The Timing view opens.

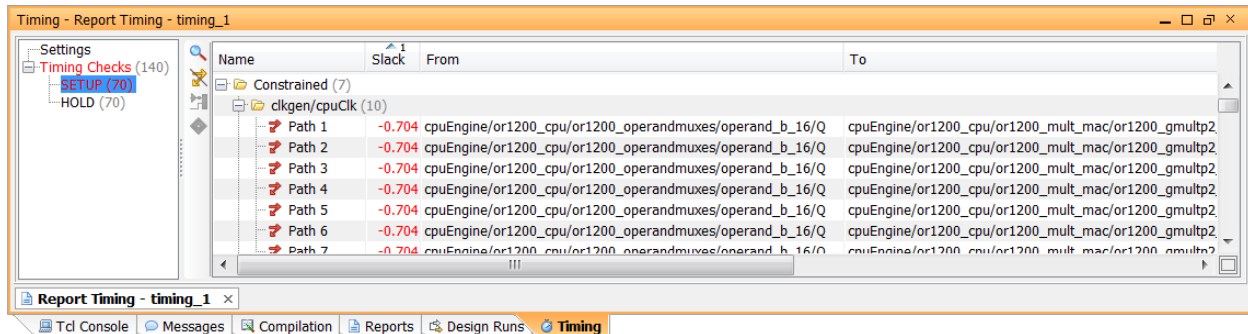


Figure 21: Analyzing the Timing Analysis Results

The list of paths display categorically in the Timing view. Paths are shown under folders indicating the relative clock constraint. The report shows Slack, Source, Destination, Total Delay, Logic Delay, Net Delay Percentage, Stages of logic, start clock, and destination clock. Failing paths appear in red.

The tree on the left allows you to select either Setup or Hold results.

15. Maximize the Timing view.

16. Scroll down the list of paths.

Note: Most of the timing paths have start and end points in the `cpuEngine` module.

17. Click the **To** column header twice to sort the list by Source.

The report is now reverse sorted by **To** column values.

Note: You can sort all of the table style views in the PlanAhead software in the same way.

- To perform a reverse sort, click the column header again.
- To perform a secondary sort, press **Ctrl** and select another column header.

18. Click **HOLD** from the tree on the left side of the Timing view and scroll down the list of paths.

19. Click **Settings** from the tree on the left side of the Timing view and examine the timer settings.


20. Click **SETUP** from the tree on the left side of the Timing view.

21. Restore the Timing view.

22. Close the Timing view.

Step 6: Implementing the Design

Reviewing the Implementation Tools Actions on the Design

1. If the Sources view is hidden from view, display it by selecting **Window > Sources**.
2. In the Sources view, click the **Collapse All** button , then expand `constrs_2`.

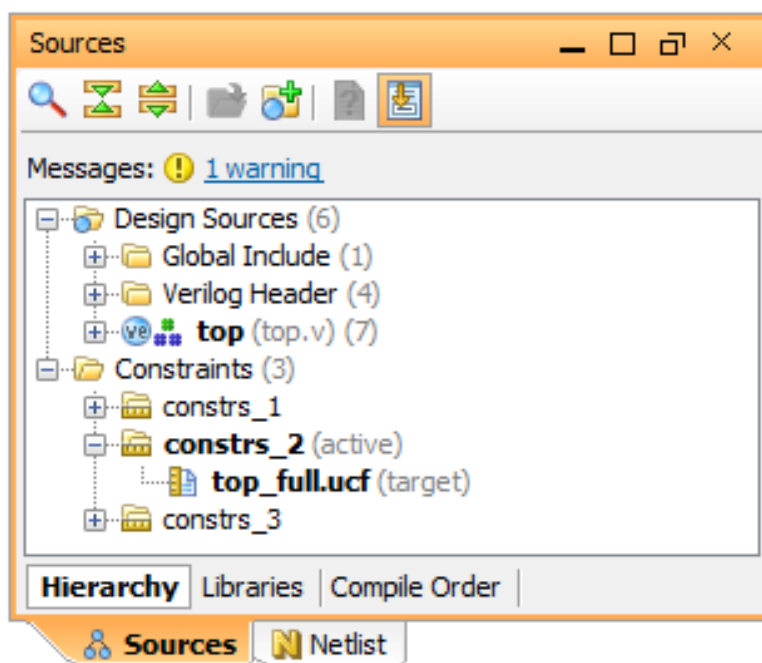


Figure 22: Sources View with Expanded `constrs_2`

When running the implementation tools, the PlanAhead software uses the active constraint file by default. Make sure that `constrs_2` using `top_full.ucf` is active. If Netlist Design or Implemented Design is open with changes, the changes will be used instead of the UCF on disk.

3. Double-click **`top_full.ucf`** to open the file. This enables you to examine the contents of the constraint file.
4. Close the UCF. Do not save any changes.

Implementing the Design

PlanAhead allows you to configure and run multiple Synthesis and Implementation runs in parallel or simultaneously. This tutorial will demonstrate how to do that, but only run one run to conserve time.

Configure Multiple Implementation Runs

5. In the Flow Navigator, right-click on **Implementation**.
6. Review the menu options, and click **Create Implementation Runs**.
7. Click **Next** in the Create New Runs confirmation dialog.
8. Click **Next** in the Set-up Implementation Runs dialog to accept the default part, synthesis run and constraints set.
9. In the Choose Implementation Strategies dialog, click **More** several times.
Notice the runs being created using the various supplied strategies. You can click on them to select the desired strategies or run and the order to run them.
10. Click **Next** to review the Launch Options.
11. Click **Cancel**.

Launch Implementation

The next step in the flow is implementing the design through the ISE Place and Route tools; specifically, NGDDBuild, Map, PAR, TRACE, and XDL.

12. Select **Run Implementation** in the Flow Navigator.
Implementation takes several minutes to complete. While waiting, you can experiment with some of the analysis features available in the Synthesized Design. Start with the ones shown in the Flow Navigator under Synthesized Design.
13. After implementation completes, select **Open Implemented Design** in the Implementation Completed dialog and click **OK**.
14. If prompted, select **Yes** to close the Synthesized Design.
Note: This prompt might have been turned off if you elected to never show this dialog again. To restore this prompt, go to **Tools > Options > Window Behavior** and check the **Show dialog before switching to a different design** checkbox.
The results from the implemented design are imported into the PlanAhead software. The Design view shows the placement of each instance and the post-place and route timing information, as shown in the following figure.

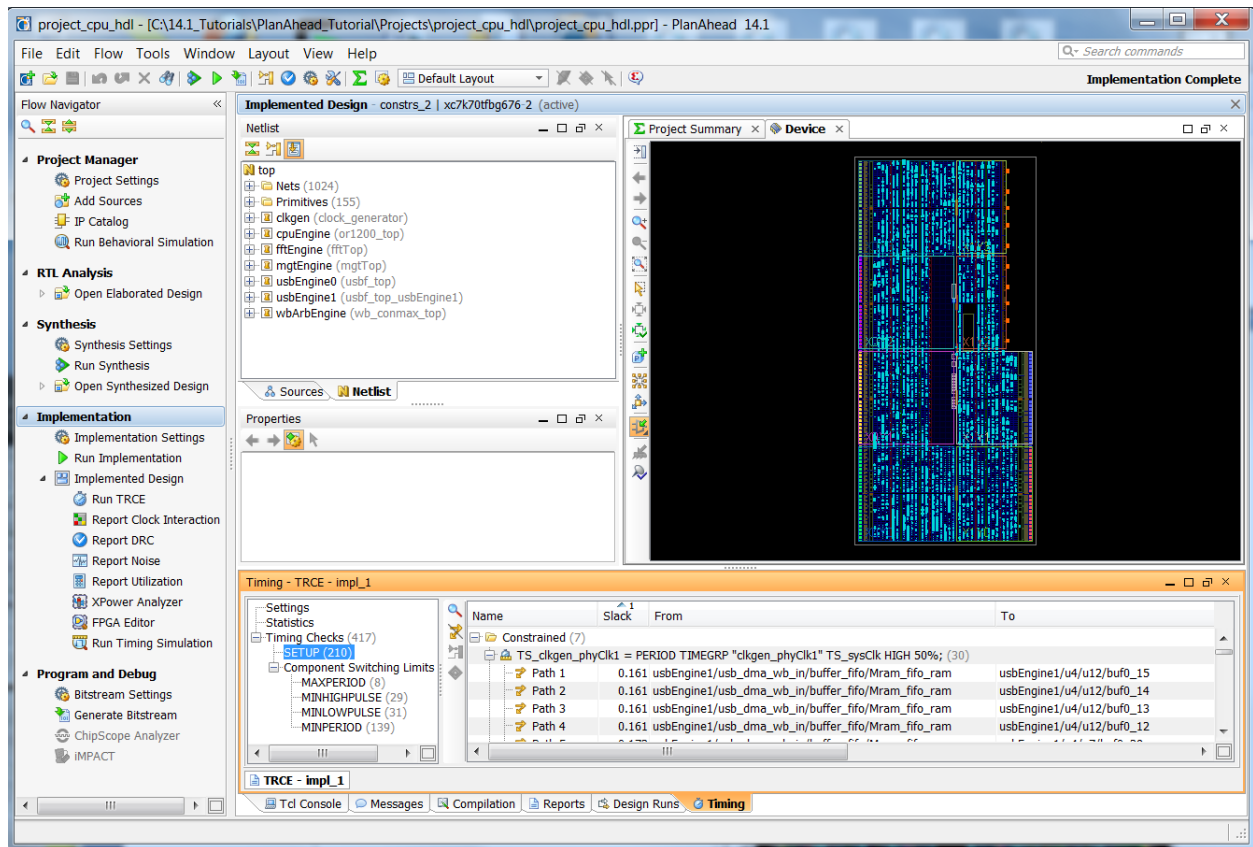



Figure 23: The Implemented Design

15. Click the **Messages** view tab to display the Messages view.
16. Click the **Collapse All** toolbar button .
17. Expand the **Implementation** messages.

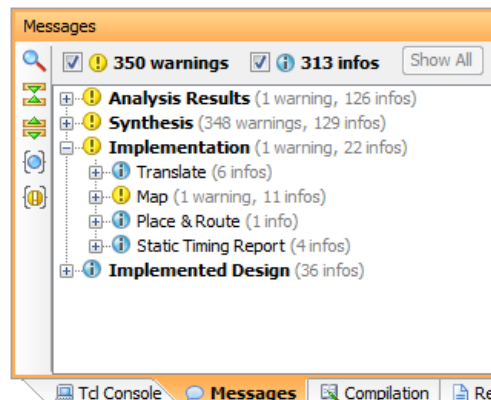


Figure 24: Messages

Note: The Messages view collects all information, warning, and error messages.

18. Click the checkbox next to the blue exclamation circle button on the top toolbar to hide information messages.
19. Expand **Map** folder to see the messages generated at this step.
20. Click the Reports view tab.
21. If the Reports view tab is not displayed, select **Window > Reports** to display the Reports view.

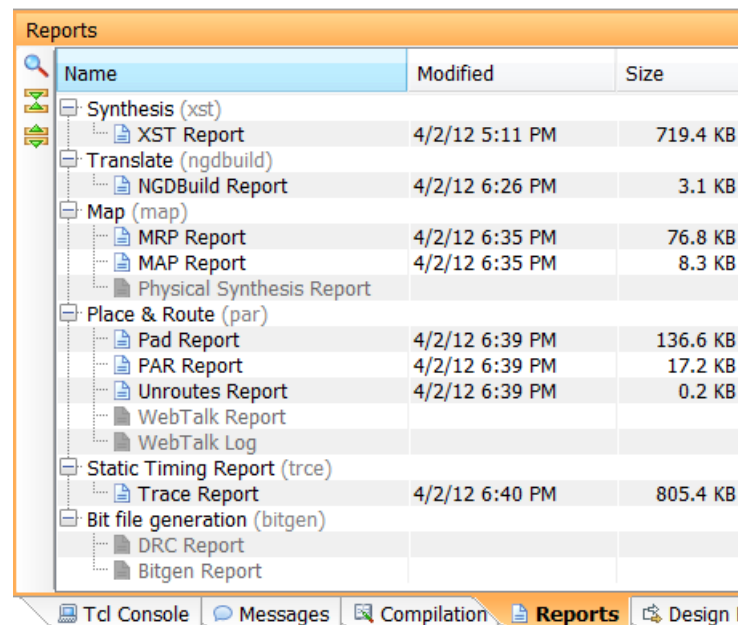


Figure 25: Viewing ISE Report Files

22. Double-click any reports to view those reports.
23. Close the report files.

Step 7: Analyzing the Timing

You can analyze timing results from the implementation process to drive the floorplanning effort. You can also use the path sorting and selection techniques available in the Timing view with the imported TRACE report data.

Exploring the Implementation Timing

1. Click the **Timing** tab.
2. In the Timing view, select **Path 1**.
3. Right-click and select **Mark** in the popup menu.

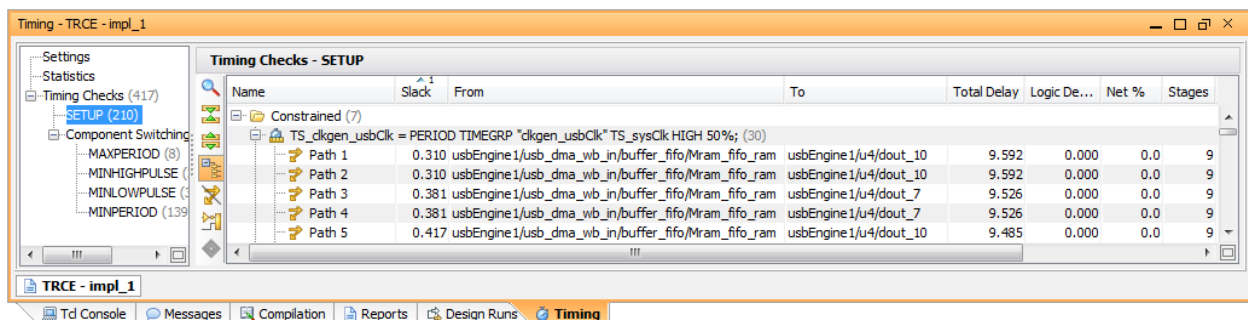


Figure 26: Selecting the Most Critical Timing Path

The imported timing paths are shown on a constraint-by-constraint basis. When you select paths in the Timing view, the Path Properties view shows the details of the timing path.

Because the placement was imported, the path is highlighted in the Device view. If needed, bring the Device view forward. The Device view makes it easier to understand how to take appropriate floorplanning steps to improve the timing.

After running the Mark command, the green diamond shows the timing path *start* point, and the red diamond shows the timing path *end* point.

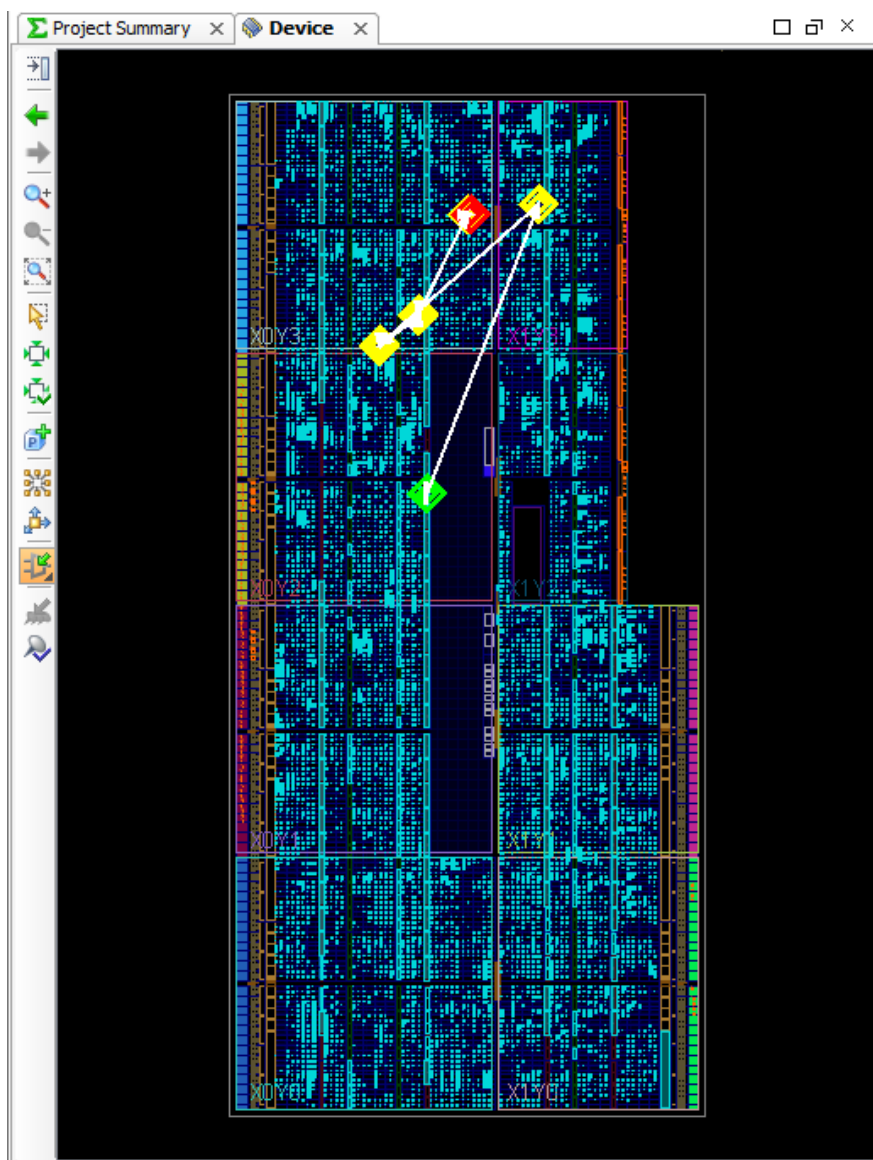


Figure 27: Placement of the Timing Path

4. Select the **Unmark All** button in the main toolbar or **View > Unmark All** to clear the marks.
 5. Timing paths are grouped by constraints. The worst paths are in TS_clkgen_usbClk.
 6. Press **Shift** and scroll to select all the paths in TS_clkgen_usbClk with usbEngine1/* as a start point. Do not grab other paths.
 7. Right-click and select **Schematic** from the popup menu.
- The Schematic view shows all of the instances on the selected paths.

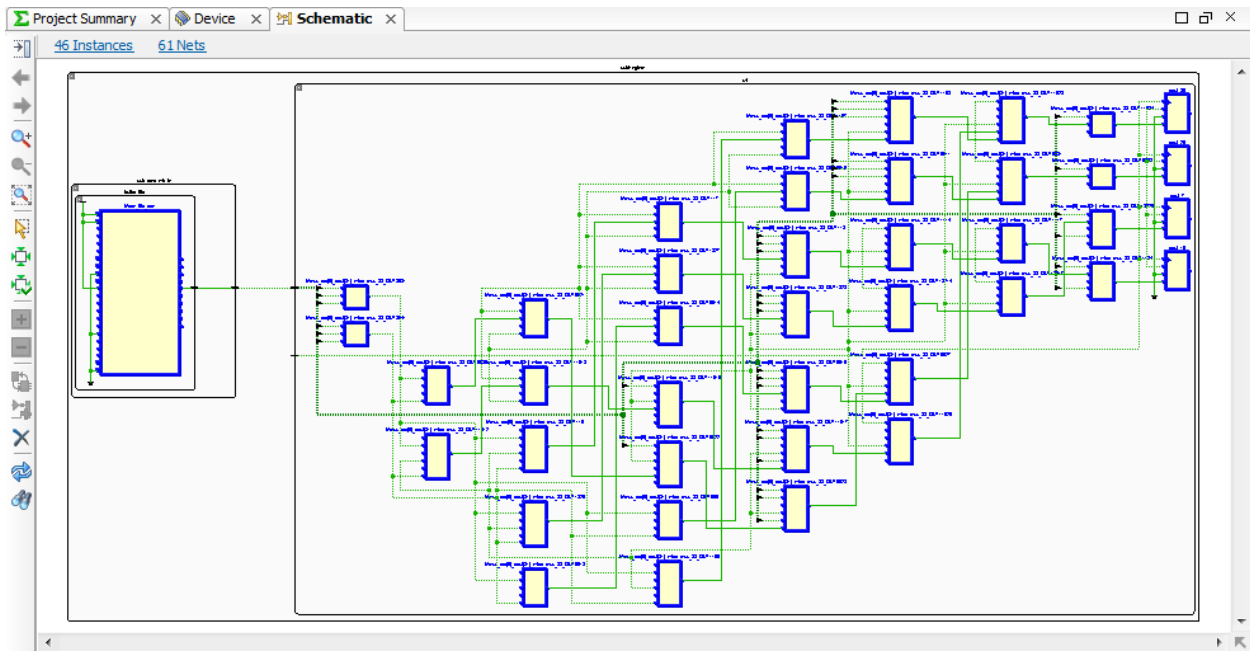



Figure 28: Examining Failing Timing Paths in the Schematic

8. In the Netlist view, click the **Collapse All** button .
9. In the Schematic view, right-click and select **Select Primitive Parents** to select the smallest parent modules that contain all of the instances in the selected paths.

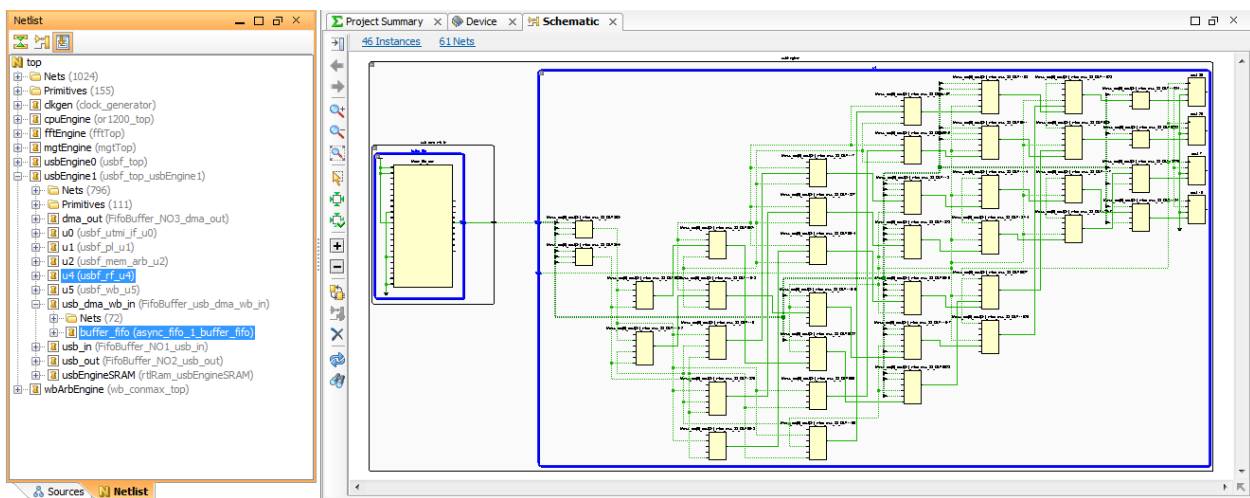


Figure 29: Selecting Path Object Parent Modules for Floorplanning

The corresponding logic modules are selected in the Netlist view.

10. Right-click and select the **Show Hierarchy** popup command, or press **F6**.
11. Use Zoom Area on the left side around `usbEngine1` to see the hierarchies.

Hint: To activate the Zoom Area mouse stroke, move the cursor to be just outside the top-left corner of the module. Hold down the left mouse button and drag the mouse to the bottom-right corner of the module. Release the mouse. Repeat as needed to see the three modules with the path logic highlighted in the Hierarchy view.

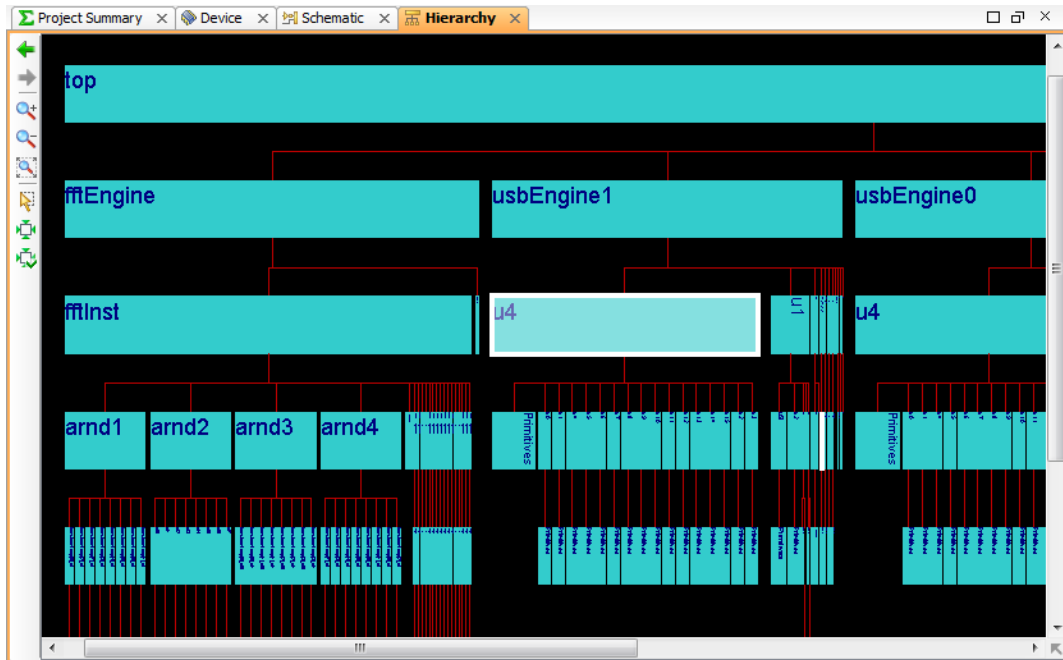



Figure 30: Viewing Selected Logic in the Hierarchy View

The logic is all contained in `usbEngine` and is a large amount of that block. Also notice an identical `usbEngine`. These two hierarchies need to be investigated for suitability for floorplanning.

12. In the **Hierarchy** view tab, click the **X** button to close the view.
13. In the **Schematic** view tab, click the **X** button to close the Schematic view.
14. Select **Path 1** in the Timing Results view.
15. Right-click and select **Schematic**.
16. Click **Unselect All**  or press **F12**.

The path logic is shown in the Schematic view.



17. Select the **RAMB36E1** on the left.
18. Right-click the select **Expand/Collapse > Collapse Outside**.



- Design Analysis and Floorplanning Tutorial
UG676 (v14.1) May 8, 2012

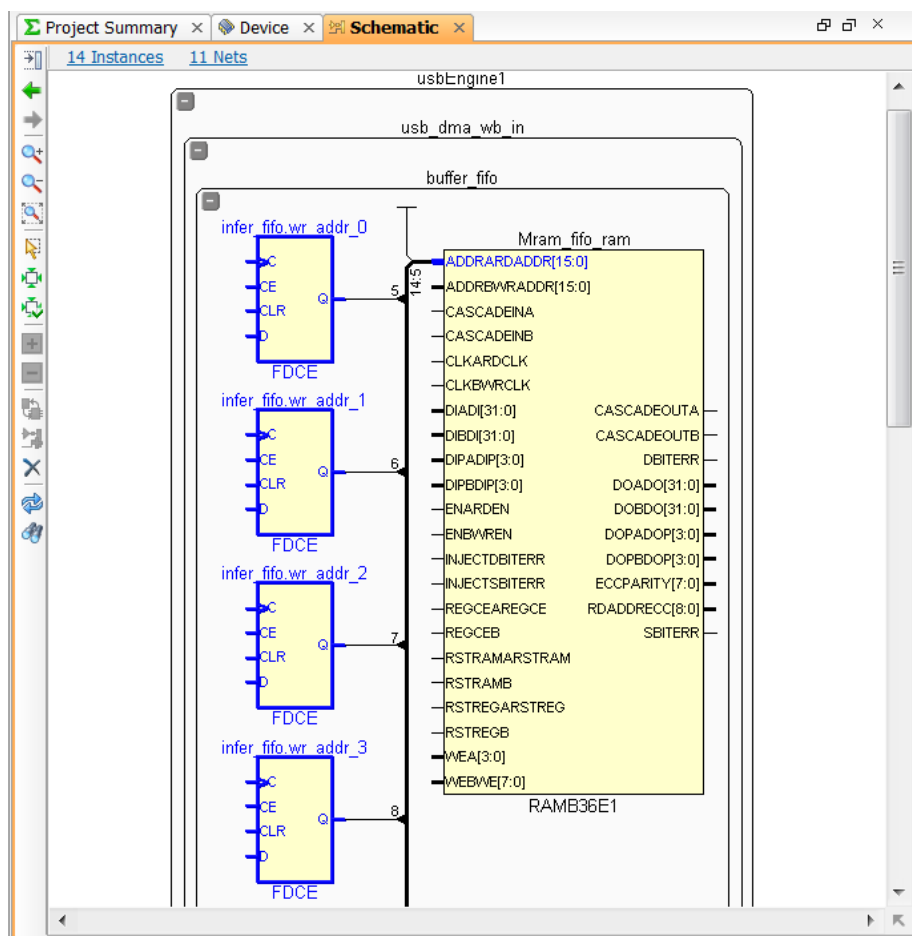





Figure 33: Interactively Expanding Logic in the Schematic View

20. Double-click the newly expanded top instance **infer_fifo.wr_addr_0**. This adds the logic to which it connects to the schematic.
21. In the Schematic view, select the **Previous schematic** button .
You can use the **Previous schematic** and **Next schematic** toolbar buttons to browse the various views displayed in the Schematic view. This lets you toggle the various levels of schematic expansion.
Note: The Edit > Undo command is *not* applicable to the Schematic view.
22. Select the rectangle that represents the **buffer_fifo** module. This contains the RAMB36E1 instance.
23. In the Schematic view tab, click the **X** button to close the Schematic view.
The input to this block RAM is contained in the `usbEngine` block. You do not have to consider other hierarchy for floorplanning.
24. Switch to the Device View.

25. Click the **Unselect All** button  or press **F12**.
26. In the Netlist view, click the **Collapse All** button .

Step 8: Highlighting Module-Level Placement

You can determine a floorplanning strategy by examining previous implementation results. You can analyze module placement and guide Pblock locations by understanding how the logic was implemented without floorplanning.

Highlighting Modules with Cycling Colors to Easily View Placement

1. In the Netlist view, select **usbEngine0** and **usbEngine1**.
2. Right-click and select **Highlight Primitives > Cycle Colors**.
3. Click the **Device** tab to view the highlighting.

The primitives in each module are highlighted in a different color.

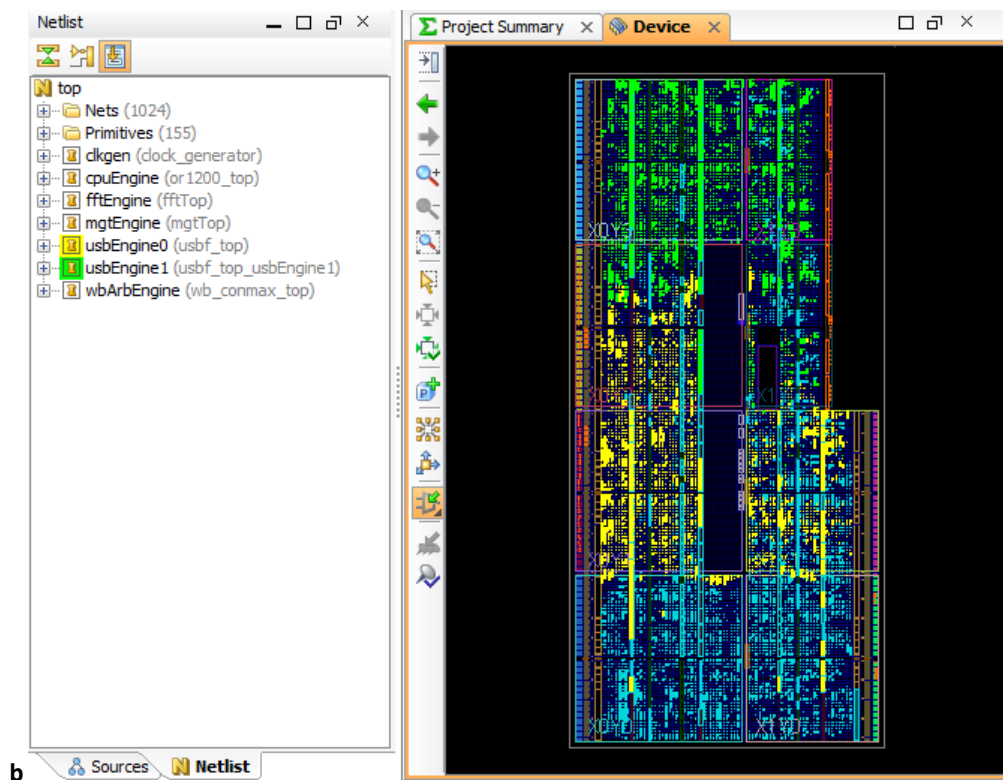



Figure 34: Highlighting Module Placement

Notice the grouped placement of the primitives within each module. Scroll around and change the Zoom level. Notice the logic is spread out. The block RAMs from the two blocks are intermingled. These might benefit from floorplanning to improve timing.

4. In the Device view, click the **Device View Layers** button .
5. Uncheck the box next to **Instances**.

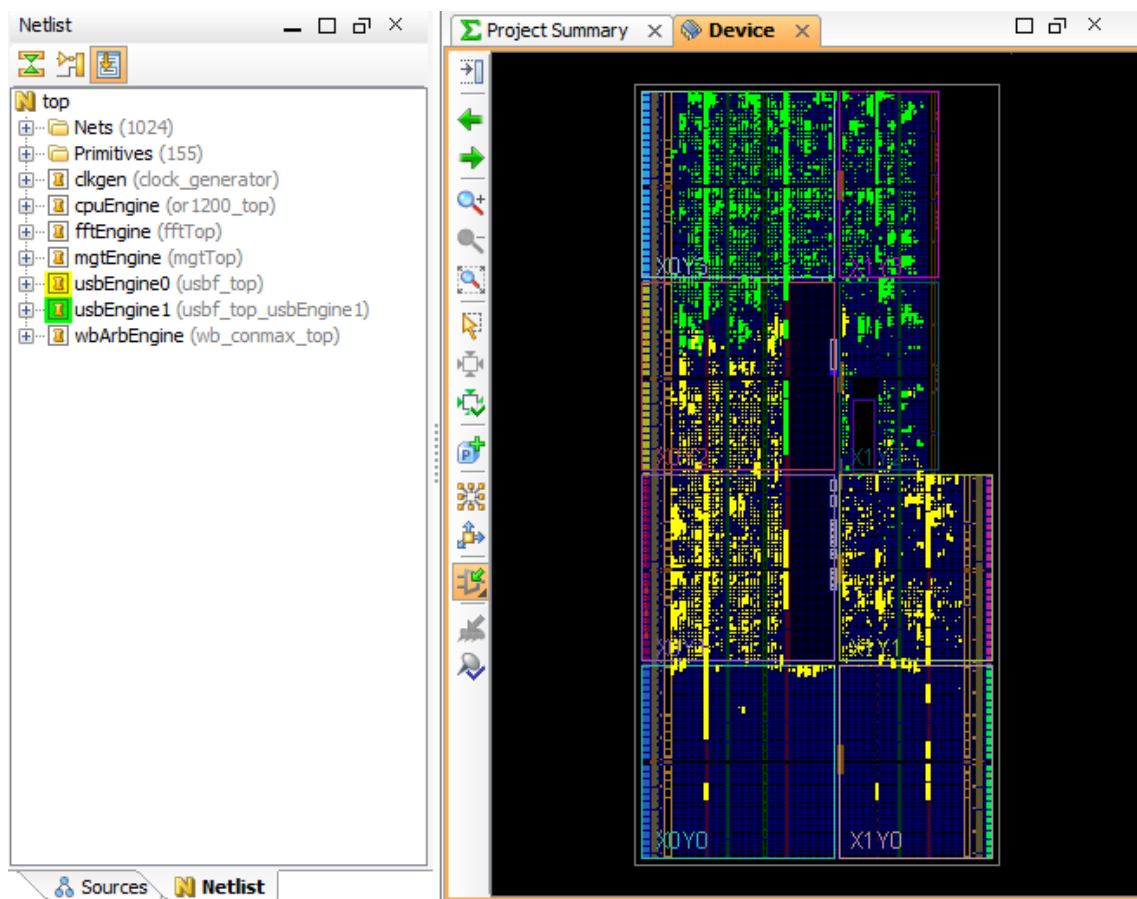




Figure 35: Simplifying the Device View

Various elements of the Netlist or elements of the Device can be hidden to make the Device view easier to interpret.

6. Check the box next to **Instances** to get the placement back.
7. Click the **Device View Layers** button  to hide the menu.
8. From the main toolbar, click the **Unhighlight All** button .


Step 9: Exploring Connectivity

The PlanAhead software has extensive logic expansion, selection, and highlighting capabilities. These capabilities can be used to validate that modules are suitable to floorplan. For example, logic modules that connect to logic throughout the device may not be suitable for floorplanning, while tightly grouped and self-contained modules are suitable.

You can alleviate routing congestion and timing inconsistency by floorplanning logic outside of the critical logic areas, thus preventing logic from migrating into the critical areas.

Visualizing the I/O Connectivity

The green lines in the Device view in the following figure show I/O Connectivity from placed instances to the I/O Pins.

1. In the Device view, select the **Show I/O Nets** toolbar button .

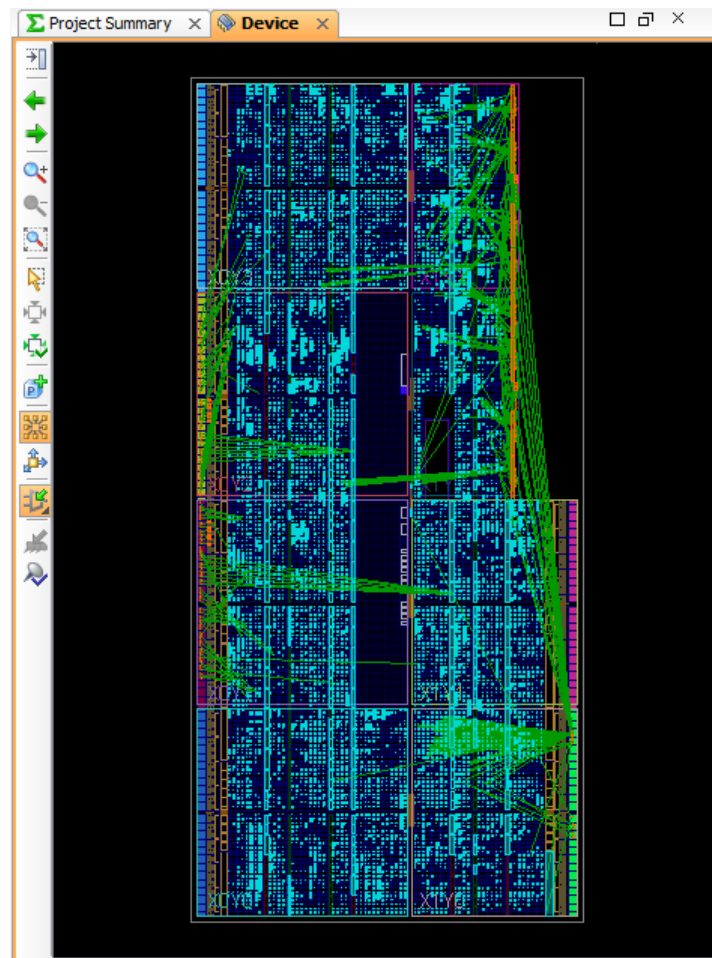


Figure 36: Device View with I/O Connectivity

Notice the I/O lines on the right side of the chip cross a long distance.


2. Click one of the I/O Nets.

Note: If you have difficulty selecting a net, check to see that the **Select** checkbox is enabled for I/O Nets in the PlanAhead Options dialog box. To locate the checkbox:

- a) Select **Tools > Options**.
- b) In the Options dialog box, click **Themes** in the left-hand menu, select the **Device** tab, and verify that the **Select** checkbox is enabled.

For more information on setting PlanAhead software options, refer to the *PlanAhead User Guide (UG632)*, available at

http://www.xilinx.com/support/documentation/sw_manuals/xilinx14_1/PlanAhead_UserGuide.pdf.

3. Inspect the net name in the Netlist view.
4. In the Netlist view, click the **Collapse All** button .
5. In the Netlist view, select **usbEngine0** and **usbEngine1**.
6. Right-click and select the **Show Connectivity** popup menu command.

The interface nets that connect `usbEngine0` and `usbEngine1` to the rest of the design are shown in yellow, as shown in the following figure.

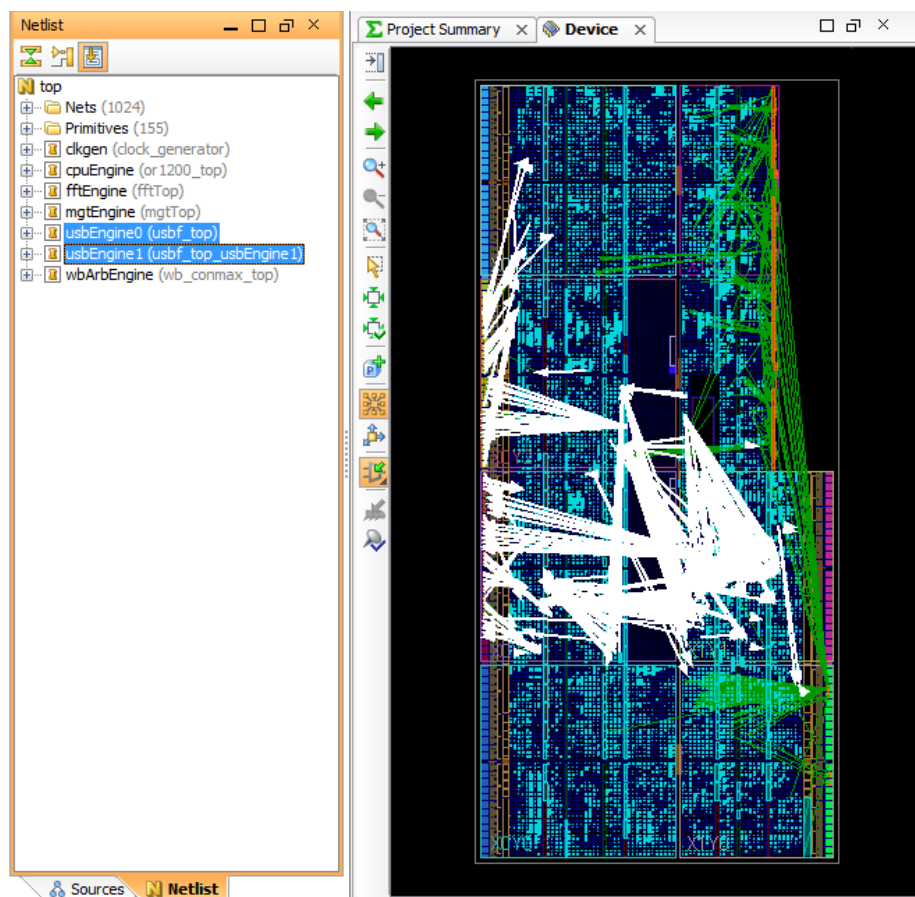


Figure 37: Using the Show Connectivity Command

7. Right-click and select **Show Connectivity** again to select all of the logic objects to which the interface nets connect.
8. Right-click and select **Show Connectivity** once again to highlight all of the nets that fanout from those selected logic objects.

You can use the Show Connectivity command to highlight or select a cone of logic from any source net or logic object.

9. Click the **Unselect All** button  or press **F12**.

Step 10: Using Placement Constraints

This step will show how to look for placement based on primitive type and clear all the placement created by place and route during implementation.

Viewing Placed Instances

1. Select **Edit > Find**.
2. Change the Instances search Criteria to be **Type > is > Block Memory**.

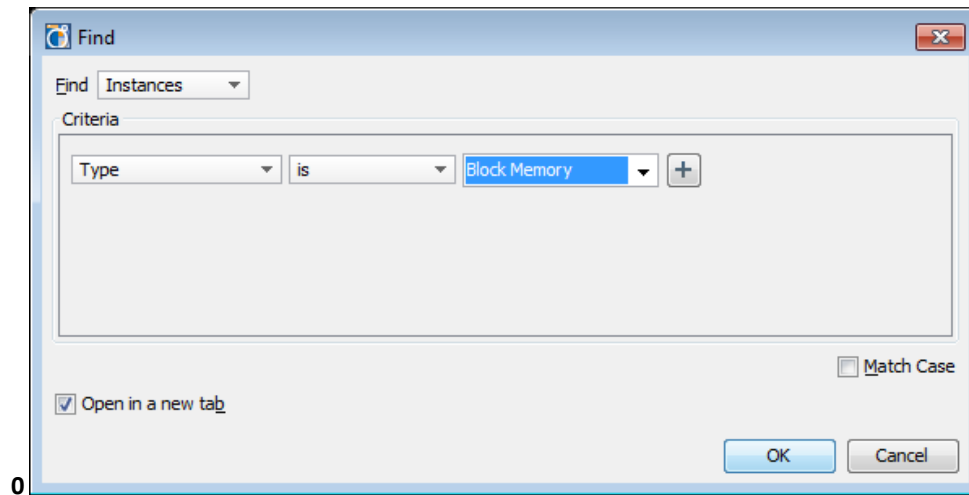


Figure 38: Find Dialog Box

3. Select **OK**.
4. Select any block RAM in the Find Results view.
The block Ram is selected in the Device view, but it might be too small to see.
5. Press the **Shift** key and select the first and last block RAM, or press **Ctrl+A** to select all block RAMs.
The Device view now has all the block RAMs selected.

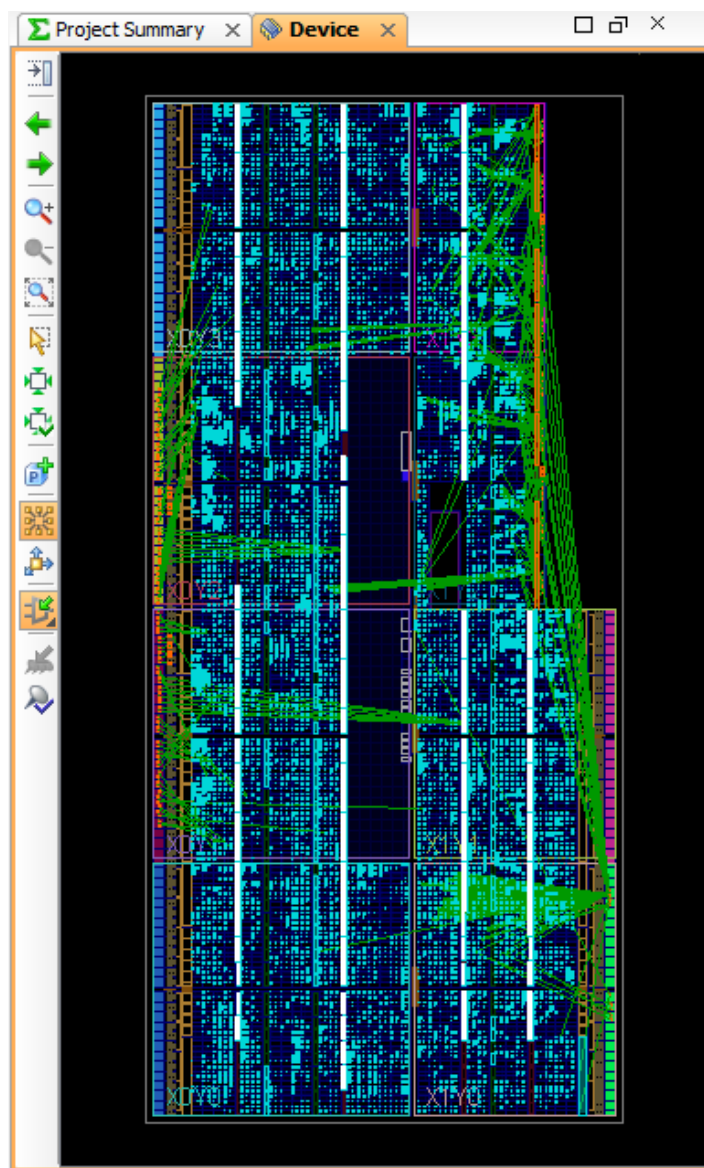


Figure 39: Device View with Block RAMs

Locking Placed Block Ram Instances


6. Right-click and select **Fix Instances**.
7. Select the **Unselect All** button in the main toolbar.
8. Zoom in and notice that the color of the "Fixed" instances is different.

If you were to save these changes, the placement LOC constraints will be written to the project UCF constraints files causing the placement to be locked during subsequent implementation attempts. The File > Save Design command will write the changes back to the active constraint set UCF files. The Save Design As command can be used to

create a new UCF file, leaving the original intact. You are prompted to Save upon closing the Implemented Design.

Clearing All Placement Constraints Using the Clear Placement Constraints Command

This step shows how to clear out placement to make it easier to understand data flow in a later step. The Clear Placement Constraints dialog box can selectively remove the placement constraints. The I/O and clock related resources are separated from the fabric logic because they usually do not change. This lets you quickly remove the placement constraints imported from the ISE software runs. Logic type filters are provided to selectively remove placement constraints by type.

9. Select **Tools > Floorplanning > Clear Placement**.
10. In the Clear Placement Constraints dialog box, select the **Instance placement** radio button.
11. Click **Next** to display the Instance Types to Unplace page.
12. Click **Default** to use the Default selection.
The configurable filters are available to selectively clear or keep logic LOC constraints by type.
13. Click **Next** to display the Fixed Placement page.
14. Select the radio button next to **Unplace all 43284 placed instances**.
Your number of Instances may vary.
15. Click **Next** to display the Clear Placement Summary page.
16. Click **Finish** after reviewing the Summary page.
17. In the Netlist view, click the **Collapse All** button .

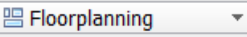
Step 11: Viewing Hierarchical Connectivity

It can be useful to look at the connectivity between the modules of the netlist. In this step, you use various tools to show the design connectivity. You will break apart the netlist hierarchy into a physical hierarchy – called a Pblock. Then you will place the Pblock to analyze design data flow. Finally you save the Pblock into the constraints file as an AREA_GROUP constraint.



Using the Auto-Creat Pblocks Command to Split Up the Top Level of the Design

1. Select **Tools > Floorplanning > Auto-create Pblocks**.
2. In the dialog box, review the options with which you can define the maximum number of Pblocks to create and specify the minimum Pblock size.

If more modules exist than the total number of Pblocks specified for creation, the PlanAhead software creates Pblocks with the largest modules.

3. Click **OK** to accept the seven selected modules.
4. Select the **Floorplanning** layout  from the View Layout drop down menu in the main toolbar to expose the Physical Constraints view.

Notice the top-level Pblocks in the Physical Constraints view.

In the Netlist view, the icon next to the six modules changed from  to , which shows that the instance was placed in a Pblock.

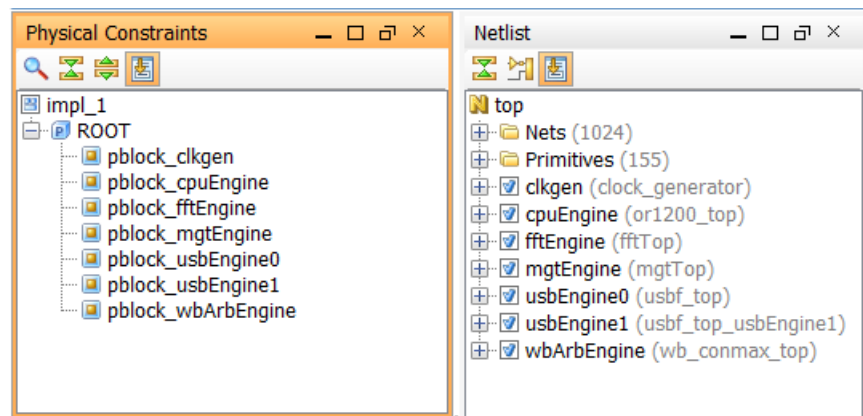


Figure 40: Partitioning the Top-Level Design

Using the Place Pblocks Command to Place the Newly Created Pblocks

5. Select **Tools > Floorplanning > Place Pblocks**.

The Place Pblocks dialog box contains options with which you can select Pblocks to place and adjust the target SLICE utilization sizes on Pblocks.

Note: The Place Pblocks command is intended to quickly create selected Pblocks. Pblocks are sized based upon SLICE logic only. Other non-SLICE ranges are not considered. Therefore, the Pblocks created using the Place Pblocks command might need modification to pass through the implementation tools successfully.

6. Click **OK** to place the Pblocks.

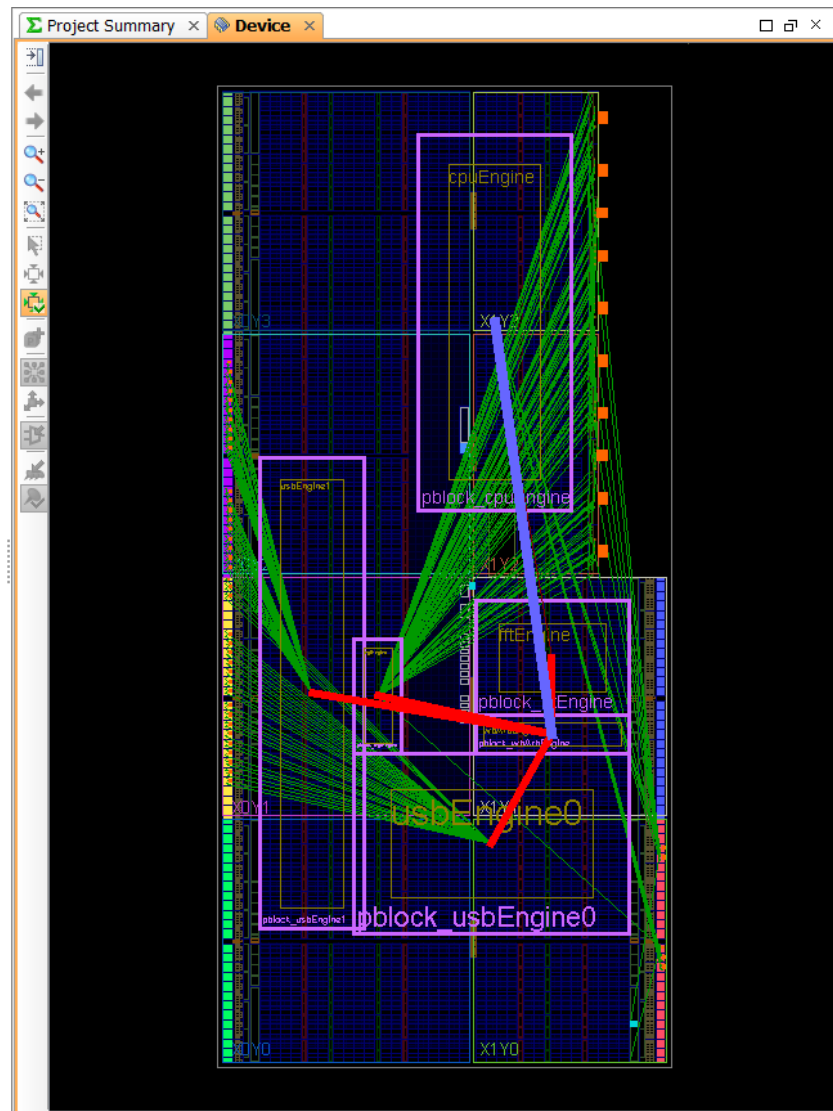




Figure 41: Place Top-Level Pblocks

Note: The Place Pblocks command could place the Pblocks differently from one run to the next. The placement might not be identical to the placement shown.

7. If connectivity is not displayed, make sure that the **Show/Hide I/O Nets** toolbar button  and **Device View Options**  > **Design** > **Bundle Nets** are on.

Hint: You can use these buttons throughout this tutorial to turn connectivity display on and off.

The bundle nets and I/O flight lines help to visualize the connectivity in the design. You can quickly arrange the Pblocks to untangle the connectivity.

This view provides early indications of data flow through the design and highlights where potential routing congestion could occur. Bundles show the number of nets shared between two Pblocks. To see the definitions, go to **Tools** > **Options** > **Colors** > **Bundle Nets**.

Adjusting the Placement and Size of the Pblocks Based on Connectivity and Resources

8. If needed, adjust the placement and size of the Pblocks to untangle the connectivity.

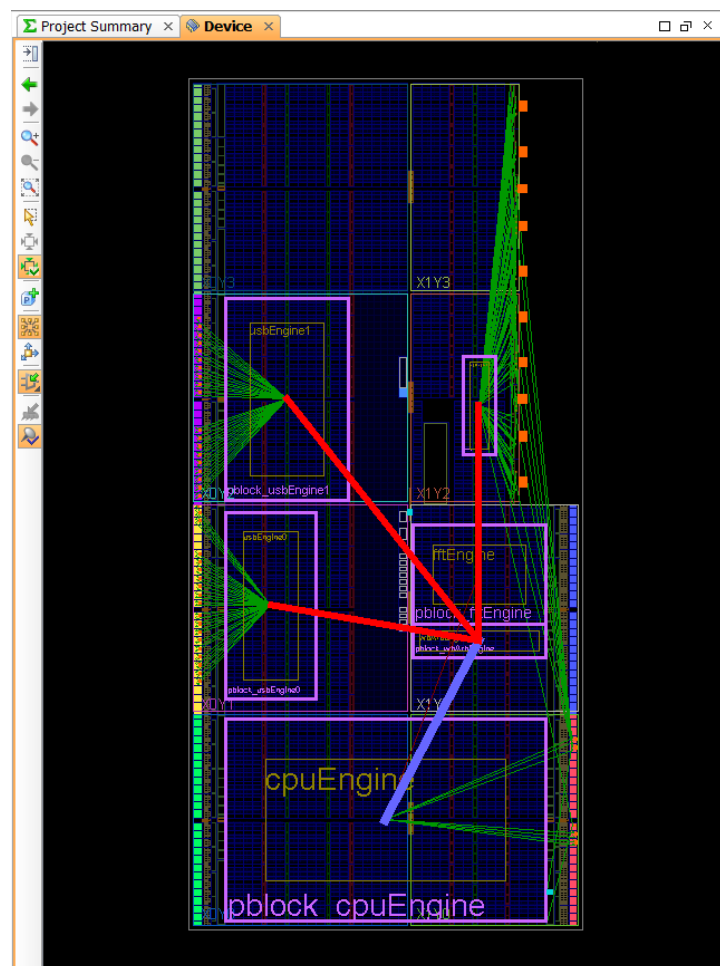


Figure 42: Reshaping Pblocks to Display Hierarchical Connectivity

Hint: In the Device view, right-click to use the **Set Pblock Size** command to redraw rectangles anywhere in the device. Move or redraw the rectangles to show a clear diagram of the logic connectivity. At this point, you do not need to size the Pblock to satisfy the resource requirements.

9. Click **OK** in the **Set Pblock** dialog box to accept all Grid Types within the Pblock rectangle.
10. If a **Choose LOC mode** dialog box pops up, accept the default and click **OK**.

For this design some registers are embedded in the I/O Pads. Their placement was not cleared during **Clear LOCs** because I/O placement was not cleared. If a Pblock initially covers these I/Os but then does not cover them on a Pblock move, the tool prompts you to determine which placement constraint has priority.

11. For this tutorial, leave the LOCs in their original positions.

Note the connectivity to `usbEngine1` and `usbEngine0`. Look at the I/O connectivity to see where the two blocks might be placed.

12. Select a Pblock, and click the **Statistics** tab in the Pblock Properties view.

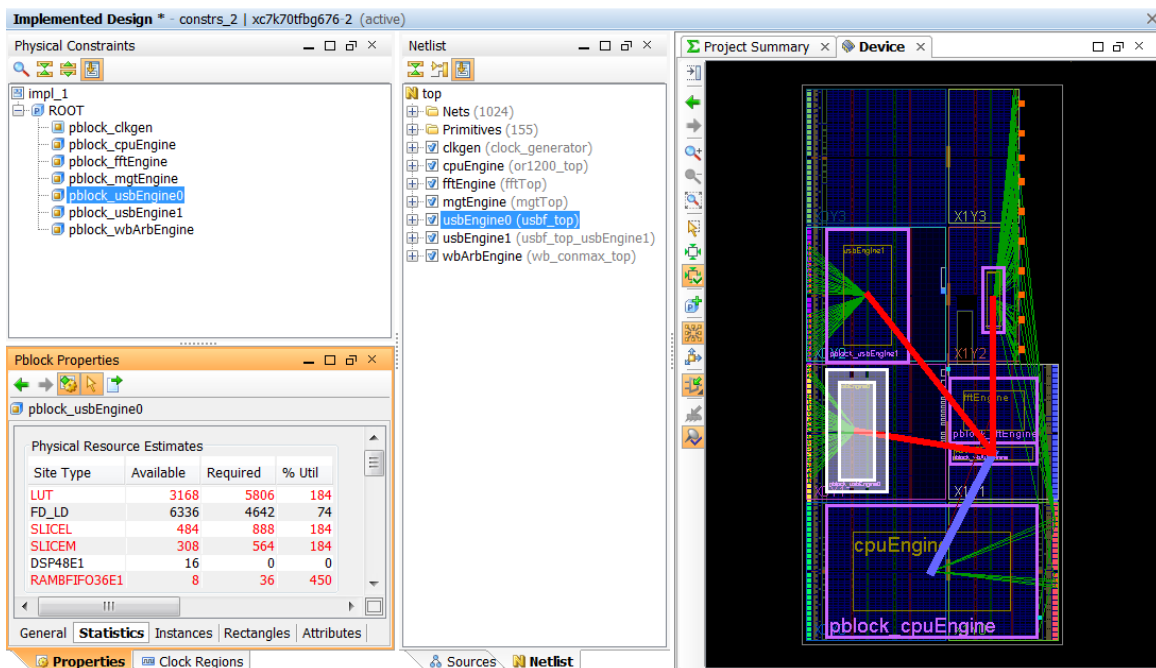


Figure 43: Sizing Pblocks Based on Resource Estimation

Notice the Physical Resource Estimates for the Pblock. You can use these resources when sizing Pblocks to ensure adequate resources exist within the Pblock rectangle to accommodate the logic to which it is assigned. The tool shows over-utilized or disabled resources in red.

- If a resource is over-utilized, the floorplan does not go through implementation.

- If a resource is disabled, the site is unconstrained to implementation.

The implementation tools can place the logic anywhere in the chip.

13. Scroll down the list of Pblock Property Statistics.

It is often easier to create floorplans and manipulate Pblocks when certain objects are made non-selectable, such as I/O nets, bundle nets, and instances, which you will do in the next sub-section.

Examining the Bundle Net Properties

14. In the Device view, select a colored bundle net.
15. Examine the Bundle Nets Properties view.
16. In the Bundle Net Properties view, click the **Properties** tab to see a list of all the nets contained within the bundle between the two modules.

The screenshot shows the 'Bundle Net Properties' window with the 'Bundle Net (118)' selected. The table lists 14 nets with their IDs, names, instance pins, flat pins, and driver status. The 'Driver' column contains checkmarks for all entries. The 'Instance Pins' and 'Flat Pins' columns show the number of pins for each net. The 'Id' column shows the net's identifier. The 'Name' column shows the net's name. The 'Driver' column shows a checkmark for each net. The 'Instance Pins' and 'Flat Pins' columns show the number of pins for each net. The 'Id' column shows the net's identifier. The 'Name' column shows the net's name. The 'Driver' column shows a checkmark for each net. The 'Instance Pins' and 'Flat Pins' columns show the number of pins for each net. The 'Id' column shows the net's identifier. The 'Name' column shows the net's name. The 'Driver' column shows a checkmark for each net. The 'Instance Pins' and 'Flat Pins' columns show the number of pins for each net. The 'Id' column shows the net's identifier. The 'Name' column shows the net's name. The 'Driver' column shows a checkmark for each net. The 'Instance Pins' and 'Flat Pins' columns show the number of pins for each net. The 'Id' column shows the net's identifier. The 'Name' column shows the net's name. The 'Driver' column shows a checkmark for each net. The 'Instance Pins' and 'Flat Pins' columns show the number of pins for each net. The 'Id' column shows the net's identifier. The 'Name' column shows the net's name. The 'Driver' column shows a checkmark for each net. The 'Instance Pins' and 'Flat Pins' columns show the number of pins for each net. The 'Id' column shows the net's identifier. The 'Name' column shows the net's name. The 'Driver' column shows a checkmark for each net. The 'Instance Pins' and 'Flat Pins' columns show the number of pins for each net. The 'Id' column shows the net's identifier. The 'Name' column shows the net's name. The 'Driver' column shows a checkmark for each net. The 'Instance Pins' and 'Flat Pins' columns show the number of pins for each net. The 'Id' column shows the net's identifier. The 'Name' column shows the net's name. The 'Driver' column shows a checkmark for each net. The 'Instance Pins' and 'Flat Pins' columns show the number of pins for each net. The 'Id' column shows the net's identifier. The 'Name' column shows the net's name. The 'Driver' column shows a checkmark for each net. The 'Instance Pins' and 'Flat Pins' columns show the number of pins for each net. The 'Id' column shows the net's identifier. The 'Name' column shows the net's name. The 'Driver' column shows a checkmark for each net. The 'Instance Pins' and 'Flat Pins' columns show the number of pins for each net. The 'Id' column shows the net's identifier. The 'Name' column shows the net's name. The 'Driver' column shows a checkmark for each net. The 'Instance Pins' and 'Flat Pins' columns show the number of pins for each net. The 'Id' column shows the net's identifier. The 'Name' column shows the net's name. The 'Driver' column shows a checkmark for each net. The 'Instance Pins' and 'Flat Pins' columns show the number of pins for each net. The 'Id' column shows the net's identifier. The 'Name' column shows the net's name. The 'Driver' column shows a checkmark for each net. The 'Instance Pins' and 'Flat Pins' columns show the number of pins for each net. The 'Id' column shows the net's identifier. The 'Name' column shows the net's name. The 'Driver' column shows a checkmark for each net. The 'Instance Pins' and 'Flat Pins' columns show the number of pins for each net. The 'Id' column shows the net's identifier. The 'Name' column shows the net's name. The 'Driver' column shows a checkmark for each net. The 'Instance Pins' and 'Flat Pins' columns show the number of pins for each net. The 'Id' column shows the net's identifier. The 'Name' column shows the net's name. The 'Driver' column shows a checkmark for each net. The 'Instance Pins' and 'Flat Pins' columns show the number of pins for each net. The 'Id' column shows the net's identifier. The 'Name' column shows the net's name. The 'Driver' column shows a checkmark for each net. The 'Instance Pins' and 'Flat Pins' columns show the number of pins for each net. The 'Id' column shows the net's identifier. The 'Name' column shows the net's name. The 'Driver' column shows a checkmark for each net. The 'Instance Pins' and 'Flat Pins' columns show the number of pins for each net. The 'Id' column shows the net's identifier. The 'Name' column shows the net's name. The 'Driver' column shows a checkmark for each net. The 'Instance Pins' and 'Flat Pins' columns show the number of pins for each net. The 'Id' column shows the net's identifier. The 'Name' column shows the net's name. The 'Driver' column shows a checkmark for each net. The 'Instance Pins' and 'Flat Pins' columns show the number of pins for each net. The 'Id' column shows the net's identifier. The 'Name' column shows the net's name. The 'Driver' column shows a checkmark for each net. The 'Instance Pins' and 'Flat Pins' columns show the number of pins for each net. The 'Id' column shows the net's identifier. The 'Name' column shows the net's name. The 'Driver' column shows a checkmark for each net. The 'Instance Pins' and 'Flat Pins' columns show the number of pins for each net. The 'Id' column shows the net's identifier. The 'Name' column shows the net's name. The 'Driver' column shows a checkmark for each net. The 'Instance Pins' and 'Flat Pins' columns show the number of pins for each net. The 'Id' column shows the net's identifier. The 'Name' column shows the net's name. The 'Driver' column shows a checkmark for each net. The 'Instance Pins' and 'Flat Pins' columns show the number of pins for each net. The 'Id' column shows the net's identifier. The 'Name' column shows the net's name. The 'Driver' column shows a checkmark for each net. The 'Instance Pins' and 'Flat Pins' columns show the number of pins for each net. The 'Id' column shows the net's identifier. The 'Name' column shows the net's name. The 'Driver' column shows a checkmark for each net. The 'Instance Pins' and 'Flat Pins' columns show the number of pins for each net. The 'Id' column shows the net's identifier. The 'Name' column shows the net's name. The 'Driver' column shows a checkmark for each net. The 'Instance Pins' and 'Flat Pins' columns show the number of pins for each net. The 'Id' column shows the net's identifier. The 'Name' column shows the net's name. The 'Driver' column shows a checkmark for each net. The 'Instance Pins' and 'Flat Pins' columns show the number of pins for each net. The 'Id' column shows the net's identifier. The 'Name' column shows the net's name. The 'Driver' column shows a checkmark for each net. The 'Instance Pins' and 'Flat Pins' columns show the number of pins for each net. The 'Id' column shows the net's identifier. The 'Name' column shows the net's name. The 'Driver' column shows a checkmark for each net. The 'Instance Pins' and 'Flat Pins' columns show the number of pins for each net. The 'Id' column shows the net's identifier. The 'Name' column shows the net's name. The 'Driver' column shows a checkmark for each net. The 'Instance Pins' and 'Flat Pins' columns show the number of pins for each net. The 'Id' column shows the net's identifier. The 'Name' column shows the net's name. The 'Driver' column shows a checkmark for each net. The 'Instance Pins' and 'Flat Pins' columns show the number of pins for each net. The 'Id' column shows the net's identifier. The 'Name' column shows the net's name. The 'Driver' column shows a checkmark for each net. The 'Instance Pins' and 'Flat Pins' columns show the number of pins for each net. The 'Id' column shows the net's identifier. The 'Name' column shows the net's name. The 'Driver' column shows a checkmark for each net. The 'Instance Pins' and 'Flat Pins' columns show the number of pins for each net. The 'Id' column shows the net's identifier. The 'Name' column shows the net's name. The 'Driver' column shows a checkmark for each net. The 'Instance Pins' and 'Flat Pins' columns show the number of pins for each net. The 'Id' column shows the net's identifier. The 'Name' column shows the net's name. The 'Driver' column shows a checkmark for each net. The 'Instance Pins' and 'Flat Pins' columns show the number of pins for each net. The 'Id' column shows the net's identifier. The 'Name' column shows the net's name. The 'Driver' column shows a checkmark for each net. The 'Instance Pins' and 'Flat Pins' columns show the number of pins for each net. The 'Id' column shows the net's identifier. The 'Name' column shows the net's name. The 'Driver' column shows a checkmark for each net. The 'Instance Pins' and 'Flat Pins' columns show the number of pins for each net. The 'Id' column shows the net's identifier. The 'Name' column shows the net's name. The 'Driver' column shows a checkmark for each net. The 'Instance Pins' and 'Flat Pins' columns show the number of pins for each net. The 'Id' column shows the net's identifier. The 'Name' column shows the net's name. The 'Driver' column shows a checkmark for each net. The 'Instance Pins' and 'Flat Pins' columns show the number of pins for each net. The 'Id' column shows the net's identifier. The 'Name' column shows the net's name. The 'Driver' column shows a checkmark for each net. The 'Instance Pins' and 'Flat Pins' columns show the number of pins for each net. The 'Id' column shows the net's identifier. The 'Name' column shows the net's name. The 'Driver' column shows a checkmark for each net. The 'Instance Pins' and 'Flat Pins' columns show the number of pins for each net. The 'Id' column shows the net's identifier. The 'Name' column shows the net's name. The 'Driver' column shows a checkmark for each net. The 'Instance Pins' and 'Flat Pins' columns show the number of pins for each net. The 'Id' column shows the net's identifier. The 'Name' column shows the net's name. The 'Driver' column shows a checkmark for each net. The 'Instance Pins' and 'Flat Pins' columns show the number of pins for each net. The 'Id' column shows the net's identifier. The 'Name' column shows the net's name. The 'Driver' column shows a checkmark for each net. The 'Instance Pins' and 'Flat Pins' columns show the number of pins for each net. The 'Id' column shows the net's identifier. The 'Name' column shows the net's name. The 'Driver' column shows a checkmark for each net. The 'Instance Pins' and 'Flat Pins' columns show the number of pins for each net. The 'Id' column shows the net's identifier. The 'Name' column shows the net's name. The 'Driver' column shows a checkmark for each net. The 'Instance Pins' and 'Flat Pins' columns show the number of pins for each net. The 'Id' column shows the net's identifier. The 'Name' column shows the net's name. The 'Driver' column shows a checkmark for each net. The 'Instance Pins' and 'Flat Pins' columns show the number of pins for each net. The 'Id' column shows the net's identifier. The 'Name' column shows the net's name. The 'Driver' column shows a checkmark for each net. The 'Instance Pins' and 'Flat Pins' columns show the number of pins for each net. The 'Id' column shows the net's identifier. The 'Name' column shows the net's name. The 'Driver' column shows a checkmark for each net. The 'Instance Pins' and 'Flat Pins' columns show the number of pins for each net. The 'Id' column shows the net's identifier. The 'Name' column shows the net's name. The 'Driver' column shows a checkmark for each net. The 'Instance Pins' and 'Flat Pins' columns show the number of pins for each net. The 'Id' column shows the net's identifier. The 'Name' column shows the net's name. The 'Driver' column shows a checkmark for each net. The 'Instance Pins' and 'Flat Pins' columns show the number of pins for each net. The 'Id' column shows the net's identifier. The 'Name' column shows the net's name. The 'Driver' column shows a checkmark for each net. The 'Instance Pins' and 'Flat Pins' columns show the number of pins for each net. The 'Id' column shows the net's identifier. The 'Name' column shows the net's name. The 'Driver' column shows a checkmark for each net. The 'Instance Pins' and 'Flat Pins' columns show the number of pins for each net. The 'Id' column shows the net's identifier. The 'Name' column shows the net's name. The 'Driver' column shows a checkmark for each net. The 'Instance Pins' and 'Flat Pins' columns show the number of pins for each net. The 'Id' column shows the net's identifier. The 'Name' column shows the net's name. The 'Driver' column shows a checkmark for each net. The 'Instance Pins' and 'Flat Pins' columns show the number of pins for each net. The 'Id' column shows the net's identifier. The 'Name' column shows the net's name. The 'Driver' column shows a checkmark for each net. The 'Instance Pins' and 'Flat Pins' columns show the number of pins for each net. The 'Id' column shows the net's identifier. The 'Name' column shows the net's name. The 'Driver' column shows a checkmark for each net. The 'Instance Pins' and 'Flat Pins' columns show the number of pins for each net. The 'Id' column shows the net's identifier. The 'Name' column shows the net's name. The 'Driver' column shows a checkmark for each net. The 'Instance Pins' and 'Flat Pins' columns show the number of pins for each net. The 'Id' column shows the net's identifier. The 'Name' column shows the net's name. The 'Driver' column shows a checkmark for each net. The 'Instance Pins' and 'Flat Pins' columns show the number of pins for each net. The 'Id' column shows the net's identifier. The 'Name' column shows the net's name. The 'Driver' column shows a checkmark for each net. The 'Instance Pins' and 'Flat Pins' columns show the number of pins for each net. The 'Id' column shows the net's identifier. The 'Name' column shows the net's name. The 'Driver' column shows a checkmark for each net. The 'Instance Pins' and 'Flat Pins' columns show the number of pins for each net. The 'Id' column shows the net's identifier. The 'Name' column shows the net's name. The 'Driver' column shows a checkmark for each net. The 'Instance Pins' and 'Flat Pins' columns show the number of pins for each net. The 'Id' column shows the net's identifier. The 'Name' column shows the net's name. The 'Driver' column shows a checkmark for each net. The 'Instance Pins' and 'Flat Pins' columns show the number of pins for each net. The 'Id' column shows the net's identifier. The 'Name' column shows the net's name. The 'Driver' column shows a checkmark for each net. The 'Instance Pins' and 'Flat Pins' columns show the number of pins for each net. The 'Id' column shows the net

Figure 44: Displaying Bundle Net Properties

Step 12: Using the Search Capability to View Clock Domains

Effective floorplanning is often dependent on proper placement of the synchronous elements in different clock domains. You can highlight clock domains to visualize connectivity and to ensure Pblocks are placed properly relative to clock regions. In this step, you highlight the USB Global Clock and view it in the schematic.

Selecting and Marking the Global Clock Nets

1. Select **Edit > Find**.
2. In the Find dialog box, set the following options:
 - Find: **Nets**
 - First field: **Type**
 - Second field: **is**
 - Third field: **Global Clock**
3. Ensure the **Unique Nets Only** checkbox is checked, and click **OK**.
4. Click the **Flat Pins** column header *twice* to sort the Find Results view.
5. In the Find Results view list, select **clkgen/usbClk_o**.

Notice the highlighted nets leading to I/Os in the two `usbEngine` Pblocks, as shown in the following figure. Since we left the block Memory instances fixed, you'll also see connectivity to the related placed BRAMs.

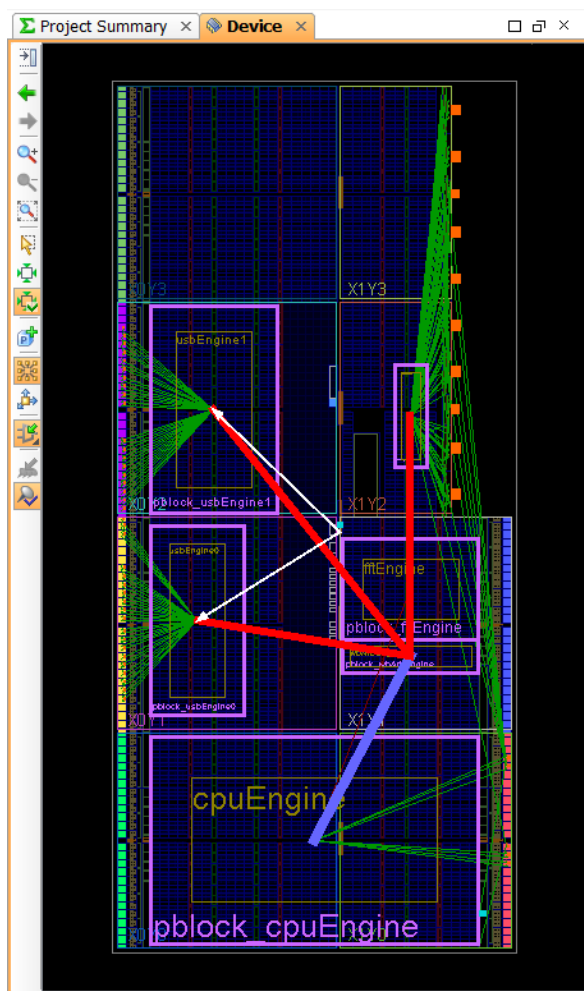


Figure 45: Viewing Clock Net Destinations

Observing the Global Clock Net Fan Out to Design Primitives

6. With the global clock net still selected in the Find Results view, right-click and select **Schematic**, or press **F4**.
The Schematic view shows the `usbClk_BUF` net connected to a group of registers.
7. Double-click on the port underneath O.
8. At the top of the schematic, zoom in on `usbClk_BUF`.

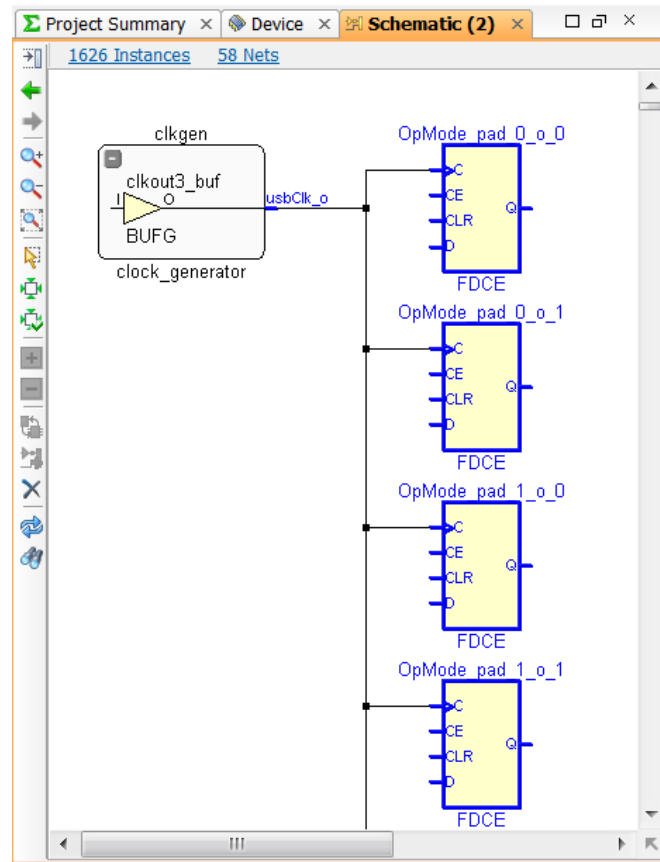




Figure 46: Viewing the Clock Signal in the Schematic

9. Close the Schematic view.
10. In the Physical Constraints view, press the **Shift** key, and select all six Pblocks.
11. Select the **Delete** from the popup menu.

Step 13: Floorplanning Timing-Critical Hierarchy

Floorplanning timing critical hierarchy or hierarchy that talks to I/Os with limited internal connectivity can improve timing performance. The previous steps show that `usbEngine1` and `usbEngine0` connect to I/Os on the left side of the chip. The `usbEngine1` and `usbEngine0` can go in the Clock Regions on the left-hand side next to their I/Os. The goal is to improve the placement of the timing critical gates by floorplanning a couple hierarchies.

Placing Pblocks for Timing-Critical Hierarchy

1. Click the **Device** tab.
2. In the Device view, ensure that I/O Nets are displayed .
3. In the Netlist view, click **Collapse All** and then select **usbEngine0**.
4. In the Device view select the **Draw Pblock** button .
The cursor turns to a cross.
5. Draw a rectangle in the clock region on the left side the device view next to the USB0 I/Os.

If you size the rectangle correctly along the clock region borders, you'll be prompted to confirm the Clock Region as the Pblock range. If not, you'll be prompted to accept the ranges for the logic types contained in the Pblock rectangle.
6. Click **OK** to accept either.

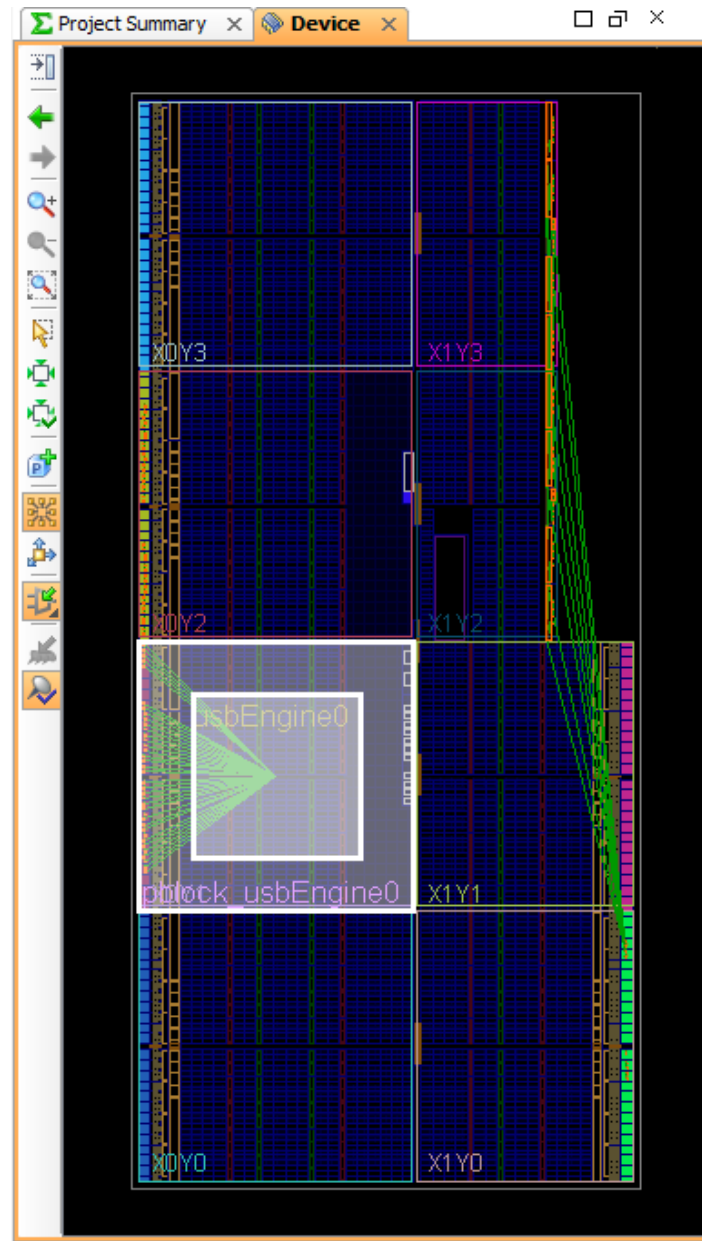


Figure 47: usbEngine0 Placed

7. Look at the **Pblock Properties**.
8. Click the **Statistics** tab.
Notice the red utilization statistics indicating the Pblock needs to be resized to go through implementation.
9. Scroll down the Pblock Properties view to notice the BRAM utilization is also red.
10. Scroll down to the bottom of the Pblock Properties view and examine the information provided.

11. Stretch the right side of Pblock all way to the bottom edge of the device and click **OK** to accept the Clock Region as the Pblock range.

Adjust the Pblock rectangle until prompted to accept the Clock Region and not the individual logic site types.

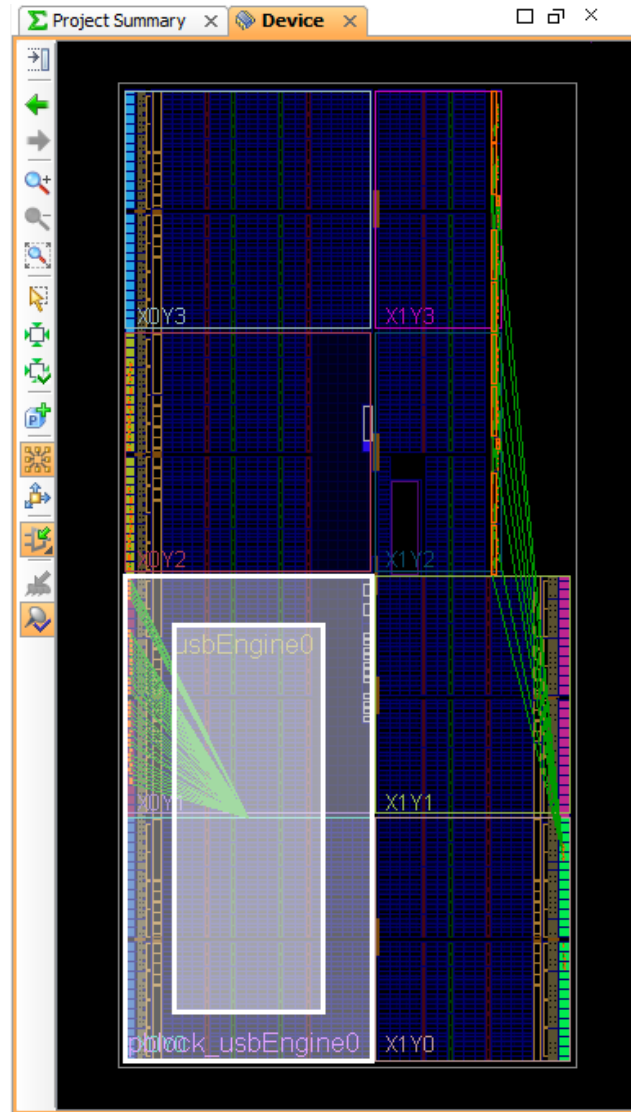


Figure 48: Floorplan both USB Timing Critical Blocks

12. Select **usbEngine1** from the netlist view and create a similar Pblock in the Clock Regions above the USB0 rectangle.

The end result is illustrated in the following figure.

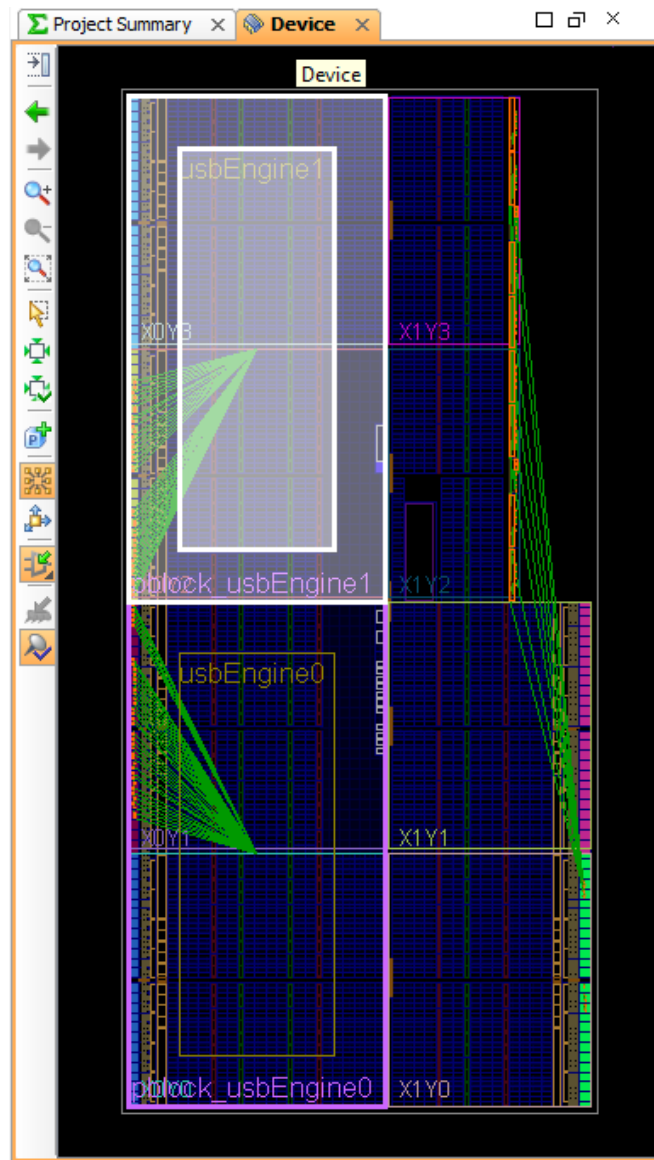


Figure 49: Floorplan both USB Timing Critical Blocks

13. Select **File > Save Design As**.
14. Notice the dialog default settings will create a new constraint set and make it active for the next run. This method will preserve your original constraints files.
15. Click **OK**.
16. In the top right of the Implemented Design view banner, click **X** to close it.
17. Click **OK** to exit and **No** if prompted to save.
The tool returns to the Project Manager.
18. In the Sources view, expand the Constraints folder.

The changes from the previous steps are saved in `constrs_4 > top_full.ucf`. If you didn't complete the previous steps, the lab ships with a completed floorplan in `constrs_3 > top_fpln.ucf`. It is easy to switch between constraint files. In the Sources view, select **constrs_3**, right-click and select **Make active**.

19. Double-click **constrs_4 > top_full.ucf** to open the file in the Text Editor.

Notice the format of the original UCF constraints file was maintained.

20. Scroll to the bottom of the UCF and notice the new `AREA_GROUP` lines.

These lines constrain levels of hierarchy and the gates in the hierarchy to regions of the chip, and store the floorplanning information from the previous steps.

21. Close the `top_full.ucf` file.

If you were to re-run Implementation, the new Floorplan constraints would be applied.

Creating Multiple Implementation Runs with the Floorplan

Another common approach to timing closure is to try various tool options in multiple runs. PlanAhead allows you to create, configure, launch and monitor results for multiple runs.

22. In the Flow Navigator, right-click on **Implementation** and select **Create Implementation Runs**.
23. Click **Next**.
24. Ensure that the tool is using:
 - Synthesized Netlist: **synth_1**
 - Constraints Set: **constrs_4**

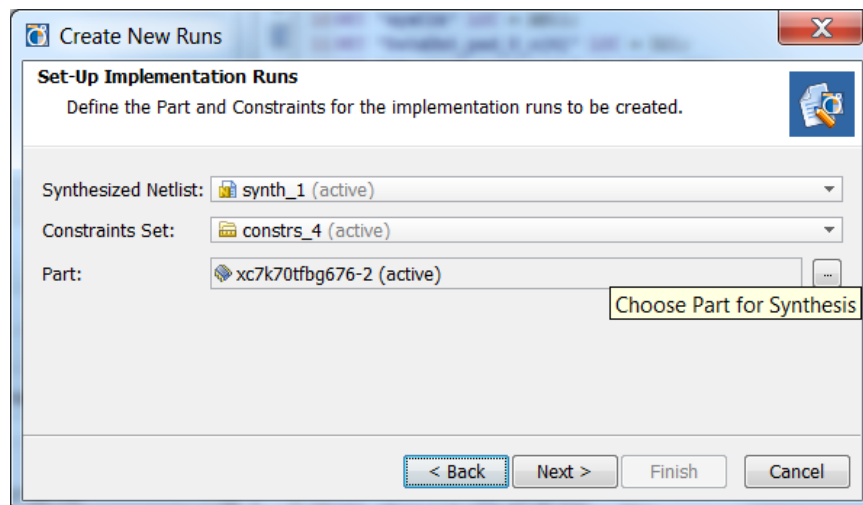


Figure 50: Selecting Netlist for Implementation

25. Click **Next**.

26. Click the **More** button three times to create additional runs with various Strategies.
The tool defines new runs and cycles through the list of strategies.

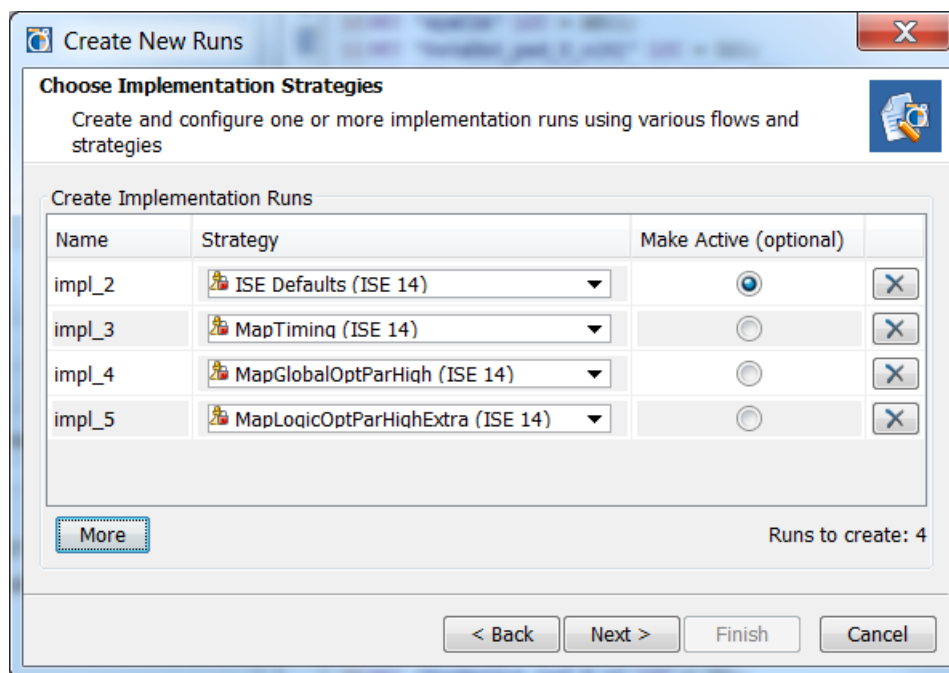


Figure 51: Multiple Runs with Multiple Strategies

27. Click **Next**.
The Launch Options pane is displayed. If the host machine has multiple CPUs, more than one run can be done in parallel.
28. Click **Next**.
The tool displays a summary page. The tutorial does not require you to continue through implementation. Don't click Finish.
29. Click **Cancel**.
30. Select **File > Exit** and click **OK** to confirm.

Conclusion

In this tutorial, you used the PlanAhead software to explore and analyze the synthesized design and targeted device prior to running the implementation tools. This enabled you to find potential design issues and errors early in the design cycle, rather than discovering issues during implementation. In addition, you used the graphical presentation of design resource estimates, design rule violations, timing estimation, constraints, and connectivity to help you understand your design and any potential areas with issues.

After running the design through the synthesis and implementation tools, you:

- Viewed implementation results and examined timing results.
- Analyzed critical path objects in the schematic, and selected the parent modules of those path objects.
- Highlighted module placement and displayed the connectivity of the modules using the Show Connectivity command.

After analyzing the design, you created a floorplan and used different command line switches to improve timing.