PlanAhead Tutorial

Release 10.1

1/15/2008 9:47 AM
Included in the PlanAhead™ source code is source code for the following programs:

**Centerpoint XML**

**NLView Schematic Engine**
Copyright © Concept Engineering.

**Static Timing Engine by Parallax Software Inc.**
Copyright © Parallax Software Inc.

**Java Two Standard Edition**

**Powered By JIDE** – [http://www.jidesoft.com](http://www.jidesoft.com)

**The BSD License for the JGoodies Looks**
Copyright© 2001-2008 JGoodies Karsten Lentzsch. All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
- Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
- Neither the name of JGoodies Karsten Lentzsch nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
# Table of Contents

**PlanAhead Tutorial** ................................................................. 2

**Introduction** ........................................................................ 8
- Tutorial Description........................................................................... 8
- Related PlanAhead Documentation and Information.......................... 9
- Interoperability .................................................................................... 9
- Dependencies.......................................................................................... 9
  - PlanAhead Lite Software ................................................................. 9
  - Corresponding Sample Design Data .................................................. 10
  - Adequate Hardware Resources .......................................................... 10
  - Xilinx ISE Placement and Routing Software ....................................... 10

**I/O Pin Assignment with PinAhead Lab** ................................. 12
- Introduction...................................................................................... 12
- Objectives......................................................................................... 12
- Procedure.......................................................................................... 12
  - Creating a PinAhead Project Step 1 ................................................. 12
  - Importing and Analyzing an I/O Port List Step 2 ......................... 17
  - Creating and Configuring I/O Ports Step 3 ..................................... 19
  - Creating I/O Port Interfaces Step 4 ............................................... 20
  - Placing I/O Ports Step 5 ................................................................. 22
  - Running DRC and WASSO Analysis Step 6 .................................... 31
  - Exporting the I/O Pinout Configuration Step 7 ............................ 37
- Conclusion......................................................................................... 38

**Getting Started with PlanAhead Lab** ...................................... 39
- Introduction...................................................................................... 39
- Description of Challenges in the Sample Design ............................... 39
- Objectives......................................................................................... 39
- Procedure.......................................................................................... 39
  - Creating a New Project Step 1 ......................................................... 40
  - Viewing the Device Resources and Clock Regions Step 2 .............. 48
  - Exploring the Logical Netlist Hierarchy Step 3 .............................. 50
  - Updating the Project with a New Netlist and New Constraints Step 4 52
  - Saving the Floorplan and Closing the Project Step 5 .................... 59
- Conclusion......................................................................................... 59

**Design Analysis and Exploration Lab** .................................... 60
- Introduction...................................................................................... 60
- Objectives......................................................................................... 60
- Procedure.......................................................................................... 60
  - Displaying Design Statistics Step 1 ................................................. 60
  - Running Design Rule Checks (DRC) Step 2 .................................... 62
  - Running WASSO Analysis Step 3 .................................................... 64
  - Graphically Viewing Logical Hierarchy Step 4 ............................. 66
Introduction

The purpose of this Tutorial is to provide a quick introduction to some of the capabilities and benefits of using the PlanAhead™ software for designing high-end FPGAs. The document consists of a series of labs with step-by-step exercises highlighting methods to achieve and maintain better performing designs in much less time. Not every command or command option is covered in the Tutorial. The Tutorial uses the features contained in the PlanAhead Lite software product, which is bundled as a part of ISE™ Design Suite™ version 10.1 and beyond. The full feature PlanAhead product can also be used to perform this Tutorial. Both versions are referred to as PlanAhead in this Tutorial.

Tutorial Description

The small sample design consisting of an 802.11 receiver core with a viterbi decoder and uart is used throughout this Tutorial. It utilizes a 2v1000 device. We’ve used a small design intentionally to allow the tutorial to be run with minimal hardware requirements and enable timely completion of the labs, as well as to minimize the shipped data size. It may not be the best candidate for performance improvement through floorplanning, as typically the best results are seen with the larger devices.

These exercises are intended to provide a walk through of the various analysis, floorplanning and implementation features of PlanAhead. Do not put too much emphasis on the specific aspects of this design. Focus on the processes and functionality of PlanAhead and how they might relate to your specific designs.

Do not let the size of this document scare you off. It contains many screenshots, and the lab actually goes by fairly quickly. Pre-run ISE results are used to reduce the Tutorial duration. There is a Tutorial stop/entry point before each Lab. Allow about 4-6 hours to complete this Tutorial.

It is also recommended that users close PlanAhead after each lab as instructed. Each lab relies on pre-run ISE implementation data that must match the floorplan it was generated with exactly. Users may get out of sync by failing to exit PlanAhead properly after each lab.

The Lab design data is modified during the process of performing this Tutorial. A new copy of the original PlanAhead _Tutorial lab data is required each time the Tutorial is performed. For more information on the tutorial data, refer to the Corresponding Sample Design Data section of this chapter.
If you have any questions or issues with the Tutorial, please do not hesitate to contact the Xilinx Customer Support hotline.

---

**Related PlanAhead Documentation and Information**

The following documentation is included in the release and available through a PDF viewer. The documents are available either under the PlanAhead Tools | Help menu, the Getting Started jump page or the Windows Start menu.

PDF format versions of the documents are available in the installation directory. The default location is:

```
<install_dir>/Xilinx/10.1/PlanAhead/doc
```

For a printable or PDF viewable version of this Tutorial, refer to the `PlanAhead_Tutorial.pdf`.

For more detailed information about the functionality of PlanAhead, refer to the `PlanAhead User Guide`.

For information about helpful floorplanning hints, refer to the `PlanAhead Methodology Guide`.

For more information about PlanAhead, including video demonstrations of the benefits of using PlanAhead, visit the following web site:

```
http://www.xilinx.com/planahead
```

---

**Interoperability**

PlanAhead runs on the following Operating Systems:

* Windows 2000 and XP
* Red Hat Linux 7.3, 8.0 and 9.0
* Red Hat Enterprise Linux 3.0, 4.0 & 5.0
* Sun Solaris 5.8 and 5.9

---

**Dependencies**

**PlanAhead Lite Software**

By default, the PlanAhead Lite software is installed with ISE 10.1 software and is accessed by entering a **valid ISE registration ID** during
installation. Please ensure that PlanAhead is operational and the sample design data has been installed prior to beginning the Tutorial. For installation instructions and information, please refer to *ISE 10.1 Release Notes*.

**Corresponding Sample Design Data**

This Tutorial uses sample design data that is included with the PlanAhead software shipment. The Tutorial design data is located in the following directory:

```plaintext
<planAhead_install_dir>/testcases/PlanAhead_Tutorial.zip
```

The PlanAhead installation process will automatically extract the zip file in the directory above. Extract the zip file in any desired write accessible location to perform the tutorial. The location of the unzipped PlanAhead_Tutorial data is referred to as the `<InstallDir>` throughout this document.

**NOTE:** If you are using the default PlanAhead_Tutorial location at `<PlanAhead InstallDir>/testcases/PlanAhead_Tutorial`, the Open an Example Project link on the Getting Started page can be used throughout the Tutorial to open the Tutorial design.

The lab sample design data is modified during the process of performing this Tutorial. A new copy of the original PlanAhead_Tutorial lab data is required each time the Tutorial is performed.

In order to reduce the data size, some implementation files were removed, leaving only the required results data in the run directories.

Refer to the Introduction section of this Tutorial for more information about the example design.

**Adequate Hardware Resources**

Xilinx highly recommends 1GB or more of RAM for use with PlanAhead on larger devices. For this Tutorial we’ve used a smaller 2v1000 design and have limited the number of Floorplans open at any one time. Having 512Mb should be sufficient, but it may impact performance.

**Xilinx ISE Placement and Routing Software**

Access to the Xilinx ISE™ Placement and Routing software (“ISE”) is suggested to perform this Tutorial, but is not required. The Xilinx commands used in this Tutorial include ngdbuild, map, par, trce and xdl. PlanAhead 10.1 supports use with Xilinx ISE software, versions 6.1i through 10.1. Previously run ISE 9.2.4 results are included and used in the Tutorial.
I/O Pin Assignment with PinAhead Lab

Introduction

This lab describes the procedure to create and assign I/O Ports to physical package pins. The PinAhead environment provides means to import or create the initial I/O Port list, group the related ports into separate folders called “Interfaces” and assign them to package pins. The capabilities include fully automatic pin placement or semi-automated interactive modes to allow controlled I/O Port assignment. PinAhead shows the relationship of the physical I/O banks and pins with their die I/O pads. Intelligent decisions can be made to optimize the connectivity between the PCB and FGPA. A CSV format pinout can be exported for use in PCB schematic symbol or HDL header generation.

The instructions are consistent for each Lab in this Tutorial. Below each general instruction for a given procedure, you will find accompanying step-by-step directions and screenshots that provide more detail for performing the general instruction. If you feel confident about a specific instruction, feel free to skip the step-by-step directions and move on to the next general instruction in the procedure.

Objectives

After completing this lab, you will be able to:

- Examine device I/O resources and the I/O bank die to package relationships
- Create, import, examine, configure and group I/O Ports
- Assign I/O Ports to physical pins using the various placement modes
- Run DRC and WASSO analysis to check for legal placement
- Export the pinout configuration

Procedure

This lab comprises eight primary steps: you will create a PinAhead project, examine device I/O resources, import and examine initial I/O port list, create I/O ports, create I/O interfaces, place I/O ports using various placement modes, run DRCs and WASSO analysis and finally export the pinout configuration.

Creating a PinAhead Project

PlanAhead is invoked to access the PinAhead environment. PinAhead is an environment completely contained within the PlanAhead Lite product. Opening the PinAhead environment simply loads a PlanAhead view layout with views
more applicable to placing I/O Ports. PinAhead can be invoked without a design for device resource analysis purposes.

In this lab, you will create a new project, open the PinAhead environment and explore the views.

Open PlanAhead and create the \textit{project\_pinout} Project and open PinAhead.

1) On Windows, select the \textbf{Xilinx PlanAhead 10.1 Desktop} icon or \textbf{Start | Programs | Xilinx ISE 10.1 | PlanAhead | PlanAhead}

   On Linux or UNIX, change directory in the \\
   \texttt{<InstallDir>/PlanAhead\_Tutorial/labs} directory and enter the \texttt{planAhead}

2) Select the \textbf{Create a New Project} link on the \textbf{Getting Started} help page

3) Click \textbf{Next} to confirm Project creation and to display the \textit{Project Name} dialog (Figure 3b-1)

   \begin{figure}[h]
   \centering
   \includegraphics[width=\textwidth]{Figure_3b-1.png}
   \caption{Enter PinAhead Project Name and Location}
   \end{figure}

4) Enter the \textit{Project name} \texttt{project\_pinout} and \textit{Project location} \\
   \texttt{<InstallDir>/PlanAhead\_Tutorial/labs/projects} (Figure 3b-1)

5) Click \textbf{Next} to display the \textit{Design Input} dialog (Figure 3b-2)
Figure 3b-2. Choose to Create Project without a Netlist

6) Select **No, do not import a netlist at this time**

7) Click **Next** to display the **Product Family** dialog

8) Click **Next** to accept the **virtex5** default and display the Floorplan Name dialog (Figure 3b-3)
Figure 3b-3. Select the Target Device

9) Use the Choose Part browser to select **virtex-5lx > 5vlx110 > ff1153 > xc5vlx110ff1153-1** (Figure 3b-3)

Click Next and Finish in the New Project Summary dialog to create the Project (Figure 3b-4).

Figure 3b-4. PinAhead Environment

Note that since no netlist was imported, the PinAhead view layout comes up by default.

Select several I/O Banks to show the package to die relationship and view I/O Bank Properties.

Select and expand the **I/O Bank 0** to view package pin specifications.

1) In the Package Pins view, select the **I/O Bank 0**

2) In the Properties view, scroll to notice the information displayed for each of the pins in the I/O Bank

3) In the Package Pins view, expand the **I/O Bank 0** (Figure 3b-5)
Figure 3b-5. Cross Highlighting I/O Banks

Notice the package pin information displayed for each pin in the I/O Bank. This I/O bank contains two VCC pins and two System Monitor pins. Notice the internal package trace min and max delays are also displayed.

4) Select one of the Package Pins and notice it highlighted in both the Device and Package views.

5) Scroll down the list of I/O Banks in the Package Pins view to select and expand several of the I/O Banks listed.

Notice the I/O Banks highlight in both the Package and Device views allowing designers to view the internal and external locations for optimal I/O Port assignment.

Notice the Package Pin view displays Global clocks (GC) and regional clock capable (CC) pins. More information is displayed as I/O Ports get assigned.

Set prohibits on all VREF pins.

Sort the Package Pins view by Voltage to select all VREF I/O pin assignments. Use the Set Prohibits popup command to prohibit placement on them and then remove the pin assignments.

1) Maximize the Package Pins view (not the Package view) by clicking the Maximize/restore banner icon.

2) Select the Group by I/O Bank button in the Package Pins view to expand and flatten the list.

3) Reverse sort the list by clicking the Voltage column header twice and scroll to find the VREF values.

4) Use the Shift key to select all of the VREF Voltage Pins and then use the right mouse button to invoke the “popup menu” to select the Set Prohibit command.

5) Minimize the Package Pins view by selecting the Maximize/restore icon keeping the I/O Ports selected.
6) Zoom into an area of the Package view to view the Prohibited pins marked with red Xs (Figure 3b-6)

Figure 3b-6. Examine Prohibited VREF Package Pins

7) **Zoom Fit** the Package view

8) In the Package Pins view, click the **Group by I/O Bank** icon

9) Click the **Collapse All** icon

---

**Importing and Analyzing an I/O Port List**

**Step 2**

- Import and examine the CSV format I/O Port list from the following file:
  
  `<InstallDir>/PlanAhead_Tutorial/labs/design_files/io_port_list.csv`

1) In a Windows Explorer view, browse and open the following I/O Ports list spreadsheet file:
  
  `<InstallDir>/PlanAhead_Tutorial/labs/design_files/io_port_list.csv`
2) Examine the I/O Ports spreadsheet format and content

3) Exit **without** saving

4) In PlanAhead, select **File | Import I/O Ports | From CSV**

5) Use the browser to select 
   `<InstallDir>/PlanAhead_Tutorial/labs/design_files/io_port_list.csv`

6) Notice the **I/O Ports** view is now populated with the imported I/O Ports (Figure 3b-7)

   ![I/O Ports View](image)

   **Figure 3b-7. I/O Bus Ports are Grouped**

   Notice that the buses are grouped together and are expandable.

7) Most of the remaining steps for I/O pin assignment involve interaction with the I/O Ports view. Unless otherwise specified, all commands in this lab should be run with the cursor in the I/O Ports view. Maximize the I/O Ports view clicking the **Maximize/restore** view banner icon.

8) Click the **Expand All** toolbar button

9) Scroll down the list of buses and signals

   Notice the **Neg Diff Pair** fields are populated for the FRM busses indicating they are Diff pairs

10) Click the **Group by Interface and Bus** toolbar button

    Notice the I/O Ports are now displayed as a list rather than grouped by bus.
11) Click the I/O Std column to sort by I/O Standard

12) Scroll down the list to display the 6 different I/O Standards values applied

13) Notice that some of the FRM0 and FRM1 signals are tied to a different LVTTL I/O Standard (Figure 3b-8)

14) Click the Group by Interface and Bus toolbar button

15) Click the Collapse All toolbar button

Figure 3b-8. Examining I/O Standard and Diff Pair Requirements

Notice the QDR memory interfaces and the PCI have unique I/O Standards applied.

Create a new I/O bus port called TCAM.

1) Select the Create I/O Ports popup menu command.

The Create I/O Ports dialog will appear (Figure 3b-9)
Figure 3b-9. Create I/O Ports

2) Enter TCAM in the Name field

3) Select the Create Bus: option and click OK

The new I/O Ports are displayed in the I/O Ports view. (Figure 3b-10)

Figure 3b-10. Display Newly Added I/O Ports

Creating I/O Port Interfaces

It is often beneficial to group the I/O Ports associated with the various I/O interfaces. PinAhead enables the definition of groups of pins, busses or other interfaces to be grouped together as an “Interface”.
This helps with I/O Port management and also when generating interface specific PCB schematic symbols. It also forces the I/O Port autoplace command to try and group the entire interface together on the device.

Create Interfaces for all of the similar I/O Port groups.

1) Click the Name column header in the I/O Ports view to sort the Ports by name

2) Select the top QDR0… bus

3) Use the Shift key to select the other 2 buses that have names starting with QDR0

4) Select the Create I/O Port Interface popup menu command

5) The Create I/O Port Interface dialog will appear (Figure 3b-11)

![Create I/O Port Interface](image)

**Figure 3b-11. Create I/O Port Interface**

6) Enter QDR0 in the Name field and click OK

7) Create I/O Port Interfaces for each of the following bus name groups: QDR1, FRM1, FRM0 and PCI

   **Hint:** Signals and Buses can be added into existing Interfaces by dragging them onto the Interface in the I/O Ports view.

8) Expand the Scalar ports folder at the bottom of the I/O Ports list

9) Use the Shift key to select and drag the corresponding signals into their I/O Port Interfaces (Figure 3b-12)
Placing I/O Ports Step 5

PinAhead provides several ways to place the I/O Ports onto either package pins or die I/O pads. The automatic placement command will attempt to place all or the selected group of I/O Ports, obey all I/O Banking rules while grouping buses and Interfaces together. For more control over I/O Port placement, the three semi-automatic placement modes allow dragging of selected I/O Ports into the Package or Device views. The three placement modes include **Place I/O Ports in an I/O Bank**, **Place I/O Ports in an Area** and **Place I/O Ports Sequentially**. The command remains active until the entire group of selected I/O Ports is placed or until the **ESC** key is pressed.

PinAhead enables interactive DRCs to be toggled on and off during I/O placement.

Turn on the interactive I/O placement DRCs.

1) Select **Tools | Options | General**

2) Ensure the **Automatically ensure legal I/O placement** switch is set to **on**

3) Click **OK**
Place the QDR0 I/O Port Interface using the **Place I/O Ports in an I/O Bank** placement mode.

Place all I/O Ports using the three placement modes and the automatic placer.

1) Expand the QDR0 Interface and QDR0/Scalar ports folder

2) Select the QDR0 Interface

3) Use the Ctrl key to **deselect** the QDR0_CQP I/O Port with the DIFF_HSTL_I I/O Std and the QDR0_KN I/O Port with the conflicting I/O Std LVCMOS25 (Figure 3b-13)

![Figure 3b-13. Place I/O Ports in an I/O Bank](image)

4) In the Package view, select the **Place I/O Ports in an I/O Bank** button

5) Notice that as the cursor is dragged over the Package Pins, the assignment pattern is displayed. The number of pins to be placed is shown in the Tooltip (Figure 3b-14)

6) Select the upper package pin in the left I/O Bank to place the Interface (Figure 3b-14)
7) Select the upper pin in the adjacent I/O Bank to place the remaining I/O Ports (Figure 3b-15)
Figure 3b-15. Continue to Place I/O Ports in I/O Banks

8) Select the top pin of the adjacent I/O Bank to place the remainder of the I/O Ports (Figure 3b-16)
Figure 3b-16. Finish Placing QDR0 I/O Ports in I/O Banks

Note that the I/O Ports are assigned in the order that they appear in the I/O Ports view. Assignment order is vectored out from the initial pin selected.

9) Select the **Collapse All** button.

Place the **QDR1** I/O Port Interface using the **Place I/O Ports in an Area** placement mode.

1) **Zoom In** to the upper right quadrant of the device in the Device view (**Figure 3b-17**)

2) Select the **QDR1** Interface

3) In the Device view, select the **Place I/O Ports in an Area** button.

4) The cursor will display a cross indicating draw rectangle mode

5) Draw a rectangle around the upper left I/O banks as shown in (**Figure 3b-17**)
Figure 3b-17. Place QDR1 I/O Ports in an Area

Place the FRM0_TDat_P differential pair bus using the Place I/O Ports Sequentially placement mode.

1) In the Package view, toggle the Show Differential I/O pairs button.

2) Zoom in to an area in the bottom center of the device where you can see the diff pair pins.

3) Expand the FRM0 Interface in the I/O Ports view and select the FRM0_RDat_P bus.

4) In the Package view, select the Place I/O Ports Sequentially button.

5) Drag and click to place the first Diff pair I/O Port into one of the lower I/O Banks (Figure 3b-18).
Notice that both of the diff pair I/O Ports were placed on legal sites.

Notice the Site field in the I/O Port Properties view to manually enter a pin location.

6) Click to place the next Diff pair I/O bus port (Figure 3b-19)
Figure 3b-19. Place Remaining Diff Pair I/O Bus Ports Sequentially

Notice the different icon in the I/O Ports view indicating that the Port is assigned.

7) Continue and place the entire bus in the same I/O Bank.

8) In the Package view, toggle the **Show Differential I/O pairs** button.

9) **Zoom fit** the Package view.

   Automatically place the remaining I/O Ports using the **Tools | Autoplace I/O Ports** command.

1) If active, click the **Unselect All** Toolbar icon.

2) Select the **Tools | Autoplace I/O Ports** command.

3) Click **Next** to confirm I/O port placement.
4) The Placed I/O Ports dialog is displayed and enables the keeping or removing of placed I/O Ports (Figure 3b-20)

![Autoplace I/O Ports dialog](image)

**Figure 3b-20. Automatically Place Remaining I/O Ports**

5) Click **Next** to accept the default selection to keep placed ports

6) Click **Finish** in the Summary dialog to place the ports

7) Click **OK** to confirm placement results

8) Click the **Collapse All** button in the I/O Ports view

9) Select the various Interfaces and notice how they are grouped nicely during placement (Figure 3b-21)
PlanAhead has an extensive set of I/O related DRC checks to ensure that I/O Ports were assigned accordingly.

Run the I/O related DRC checks.

1) Select **Tools | Run DRC**

2) Deselect the **Clock, Floorplan, DSP4** and **RAM16** rule categories as shown in **Figure 3b-22**
Figure 3b-22. Run I/O Related DRCs

3) Expand the **IOB** and **Bank** rules to examine the rule types

4) Click **OK**

Notice there were no violations found.

PinAhead enables the configuration of I/O ports for I/O standard, drive strength and slew type. Change an I/O Standard to create a DRC violation and then use the Violation view to identify the offending objects.

Change the I/O Standard for one of the I/O Ports to create a conflict.

1) Expand the **TCAM** bus in the I/O Ports view and select one of the Ports

2) Select the **Configure I/O Ports** popup menu command

The **Configure I/O Ports** dialog appears. (Figure 3b-23)
3) Change the I/O Standard to **LVCMOS12** and click **OK**

Rerun DRC and identify the offending objects.

Run DRC. Select the violation and follow the links in the Violations Properties to find the problem.

1) Select **Tools | Run DRC** and click **OK** to run the same rules again

2) Notice the violation reported in the **DRC Results view** (Figure 3b-24)

3) Select the **BIVC #1** violation (Figure 3b-24)

4) In the Violation Properties view, notice the violation has to do with I/O Ports with conflicting VCC Voltages. (Figure 3b-25)
5) In the Violation Properties view, select each one of the site links and view the I/O Port information in the I/O Ports view. Notice the violation has to do with the I/O Port assigned to site we changed. Note that your pin assignment location may vary.

6) Select the offending I/O Port with the LVCMOS12 I/O Standard and use the popup menu **Configure I/O Ports** command to set it back to the default LVCMOS25

7) Select the **Tools | Run DRC**

8) Notice no violations were found. If your design still reports violations, please proceed regardless.

Run **WASSO Analysis** to check for potential signal integrity issues

Run the **Run WASSO Analysis** command to check for potential signal integrity issues.

1) Select **Tools | Run WASSO Analysis**

2) The Run WASSO Analysis dialog is displayed (**Figure 3b-26**)
Figure 3b-26. Run WASSO Analysis

Notice the **Output File** field in the Run WASSO Analysis dialog to allow creation of a report file.

Notice the Tooltips for the entry fields indicating the values to enter.

3) Click **OK** to accept defaults

4) The WASSO results will be displayed in a Workspace view (Figure 3b-27)
Figure 3b-27. Examine WASSO Results

5) Scroll down the list of I/O Banks and their neighbors. Notice the Status is OK for the entire device.

6) Close the WASSO – results_1 view by clicking on the Close X icon.
Exporting the I/O Pinout Configuration      Step 7

The I/O Ports configuration can be exported in UCF, CSV, VHDL or Verilog format. This is useful in HDL header and PCB schematic symbol generation. The file also contains package information for all pins and can be used as a seed to begin I/O Port configuration via the spreadsheet.

Export the I/O Ports list using the **File | Export I/O Ports** command.

1) Select the **File | Export I/O Ports** command

2) Use the default CSV format and browse to the following folder:
   &lt;Installdir&gt;/PlanAhead_Tutorial/labs/design_files

3) Click **OK** to accept the default file name, *floorplan_1.csv* (Figure 3b-28)

4) Open a Windows Explorer window and browse to find and open the *floorplan_1.csv* file (Figure 3b-29)

5) Close the *floorplan_1.csv* file

   Close PlanAhead.

   1) Select **File | Exit** and click **Yes** to save and **OK** to close PlanAhead
Conclusion

In this Lab you used the PinAhead I/O pin planning environment to import, create and configure I/O Ports. You created Interfaces by grouping the related I/O Ports together.

You then used the semi automatic placement modes to assign critical I/O Ports to package pins. Placement of the remaining I/O Ports was done using automatic placement.

You then ran DRCs to validate legal I/O placement. You then created a violation to identify some I/O pin assignment issues. We fixed the I/O pins in question using the Violations view and got no DRC or WASSO analysis errors.

You then exported and examined the I/O Ports list usable for HDL header or PCB schematic symbol generation.
Getting Started with PlanAhead Lab

Introduction

This lab illustrates the steps you take to create a Project and initial floorplan by importing a synthesized design netlist into the PlanAhead software. An initial Floorplan is created using a constraints file and by selecting a target device. It also introduces the PlanAhead environment and views. An updated Netlist is then imported to demonstrate the process to update the design with a new netlist and constraints file.

Description of Challenges in the Sample Design

The sample design used in the tutorial is an 802.11 receiver module with a viterbi decoder. The main challenge in this design is the placement of the BRAMs and Multipliers. The design targets a 2v1000 device and these types of devices have restrictions on BRAM usage. BRAMs using data paths wider than 18 bits require routing channels from the adjacent Multiplier sites making those sites unusable. This is currently not reflected in the PlanAhead resource utilization statistics or in the ISE map report. It’s a Virtex-II restriction that makes this design a lot more difficult to implement that it looks, as you’ll see.

Objectives

After completing this lab, you will be able to:
- Import a design EDIF netlist and create a new PlanAhead Project
- Import UCF constraints and create a new Floorplan
- View the device resources
- View the device clock regions
- Explore the logical netlist hierarchy
- Import a new top-level netlist and constraints
- Save the Floorplan

Procedure

This lab comprises five primary steps: you will create a new project and a new Floorplan, view the device resources and clock regions, explore the logical netlist hierarchy, import an updated netlist and constraints from synthesis, and finally save the Floorplan.

Below each general instruction for a given procedure, you will find accompanying step-by-step directions and screenshots that provide more detail for performing
the general instruction. If you feel confident about a specific instruction, feel free
to skip the step-by-step directions and move on to the next general instruction in
the procedure.

**Note:** Before beginning the Tutorial, you should copy the tutorial lab data to a
folder for which you have write privileges. The lab data is only good for one pass
through the Tutorial, so it helps preserve the original Lab data for future Tutorial
users. You should extract the Tutorial sample design data to a writable location.
You can find the sample Tutorial design zip file in the PlanAhead installation
area:

```
<planAhead_install_dir>\testcases\PlanAhead_Lite_Tutorial.zip
```

You can also download the lab files for this module from the Xilinx FTP site at

### Creating a New Project

<table>
<thead>
<tr>
<th>Step 1</th>
</tr>
</thead>
</table>

Launch the PlanAhead software.

1. On Windows, select the **Xilinx PlanAhead 10.1 Desktop icon or Start | Programs | Xilinx| 10.1 PlanAhead | PlanAhead**

On Linux or UNIX, change directory in the

```
<InstallDir>/PlanAhead_Tutorial/labs directory and enter the planAhead
```

The PlanAhead Environment appears *(Figure 5b-1)*.
Figure 5b-1: The PlanAhead Environment

Notice the PlanAhead Getting Started jump page with links to open or create Projects and to view documentation.

Create a new project called `project_1` using the EDIF files in the `<Install_Dir>\PlanAhead_Tutorial\labs\design_files` directory.

1) Select the **Create a New Project** link on the Getting Started jump page.

The New Project overview dialog appears (Figure 5b-2).
2) Click **Next**

The Project Name dialog appears (Figure 5b-3).

![Figure 5b-2: New Project Overview](image)

![Figure 5b-3: Project Name Dialog](image)

3) Use the file browser on the **Project location** field to select the following folder: `<Install_Dir>\PlanAhead_Tutorial\labs\projects` (Figure 5b-3)

4) Use the default Project name: **project_1**
5) Click **Next**

6) Click the option, **Yes to import a synthesized (EDIF or NGC) netlist** (Figure 5b-4)

![Figure 5b-4: Electing to Import a Netlist](image)

The **Input Netlist** dialog is displayed (Figure 5b-5)

![Figure 5b-5: Selecting a Top-level Netlist to Import](image)
7) Click the browser icon next to Netlist file: field and navigate to the `<Install_DIR>\PlanAhead_Tutorial\labs\design_files` folder.

8) Select `802RCV_cos13.edf` and click Open (Figure 5b-5)

**NOTE:** The PlanAhead software must be told where to find module-level or core EDIF netlists that comprise the design. By default, the PlanAhead software will search for module netlists in the PlanAhead invocation directory and in the directory containing the top-level netlist. Additional search path directories may be included by using the Add button in the Netlist directories in this dialog.

9) Click Next

Wait while the netlists are imported. After the netlists have been imported, you will be prompted to select a target device for the initial Floorplan.

Select the Virtex2 product device family and target device. Create a new Floorplan called `my_fp` using the UCF file in the `<Install_DIR>\PlanAhead_Tutorial\labs\design_files` directory.

1) Click Next to accept the default Product Family: `virtex2` (Figure 5b-6)

![Figure 5b-6: Selecting the Product Family](image)

2) Enter `my_fp` in the Floorplan name: field and click Next to use the target device that is specified in the netlist (Figure 5b-7)
Note that an alternate device can be selected for each Floorplan.

**Figure 5b-7: Defining Floorplan Name and Selecting a Target Device**

3) Click the **Add** button and use the browser to select the `802RCV_cos13.ucf` file and click **Open** (Figure 5b-8)

**Figure 5b-8: Selecting a UCF File**

4) Click **Next**
5) Review the Summary information for the Floorplan and click Finish (Figure 5b-9)

**Figure 5b-9: New Project Summary**

Wait a few seconds for the Floorplan to initialize. The PlanAhead Desktop will appear (Figure 5b-10).

**TIP:** Various Floorplans can be created in the Project to experiment with different devices, timing constraints or placement constraints. They all reference the same imported Netlist.
Figure 5b-10: The PlanAhead Desktop Environment

**NOTE:** The tile rectangles may or may not appear in the Device view depending on the window size and zoom level.

Notice that the Floorplan - my fp is now displayed.

6) Scroll up in the Console view to review design import messages. These messages are also written to the planAhead.log file

**TIP:** Review the messages carefully when importing a new Netlist or constraints to ensure successful parsing and to highlight any errors in the input files. The messages are similar to what you would see in the ISE ngdbuild report.

**NOTE:** PlanAhead enables the user to configure the Environment views and create custom view layouts. This capability is not covered in this Tutorial. Please refer to *Chapter 3: Using the Viewing Environment* of the PlanAhead User Guide for more information.
**Viewing the Device Resources and Clock Regions**

**Step 2**

In the Device view, zoom in and examine various device resources.

1) In the Device view, click in the upper left-hand corner of the device, and drag down and to the right to draw a zoom area rectangle in an area small enough to see the devices resources, as shown in [Figure 5b-11](#).

![Figure 5b-11: Zooming into the Device to view CLBs and Logic Gate Sites](image)

2) Slowly move the cursor over the various device resources to view the Tooltips.

   Notice the SLICE coordinates are displayed in the very bottom banner on the PlanAhead Desktop.

   Notice the I/O port locations and IO buffer assignments along the outside edges.

   Notice the tall red tiles indicating sites for a single RAM and MULT.

   Notice the square blue tiles containing 4 CLB SLICEs and 2 center TBUFS.
3) Select the **Zoom Fit** Toolbar button to fit the whole device in the Device view.

Open the Clock Regions view and display a clock region in the Device view.

The clock regions in the device are displayed in the Device view and can be used to help guide floorplanning. The Clock Region view lists all of 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 *Design Partitioning* lab.

1) Select the **Window | Clock Regions** command

2) Select one of the clock regions listed in the table

3) Click the **Properties** view tab to display the Clock Region Properties view

4) Click the **Statistics** tab in the Clock Region Properties view to examine the logic contents

5) Click the **Resources** tab in the Clock Region Properties view to examine the contents

The clock region is highlighted in the Device view, as shown in Figure 5b-12.

![Figure 5b-12: Viewing a Clock Region Properties](image)

6) Select one of the **BUFGMUXs** in the Resources list and notice it is highlighted in the Device view
7) Click the **I/O Banks** tab in the Clock Region Properties view to examine the I/O Banks related to the clock region.

## Exploring the Logical Netlist Hierarchy

**Step 3**

Use the Netlist view to explore the design hierarchy. Collapse the netlist tree when you are finished.

1) Click **Collapse All** Toolbar button in the Netlist view.

2) In the Netlist view, click the **receiver** expand widget to expand the module.

3) Expand the **RChainTopInst** module.

The Netlist view should look like Figure 5b-13.

![Figure 5b-13: Expanding the Netlist View](image-url)
Notice the Primitives folders in the Netlist view. They contain the top-level instances of each module.

Notice the Nets folders in the Netlist view. They contain the module net names that are relevant to the module.

Maximize the Netlist view full screen.

Any view can be displayed using the full PlanAhead screen by either double-clicking on the desired view tab or selecting the **Maximize/restore** icon in the view header.

1) Double-click the **Netlist view** tab in the lower left corner of the Netlist view.

Notice the Netlist view is now displayed full screen (Figure 5b-14).

![Figure 5b-14: Displaying Views using the Full Screen](image)

2) Experiment expanding and traversing the hierarchy

3) When finished, click **Collapse All** Toolbar button in the Netlist view

4) Click the **Maximize/restore** icon to restore the Netlist view
The Workspace graphical views do not have the same view header as the other views. Double click the Workspace view tabs to maximize the view or use the popup menu under the view tab. To restore Workspace views back to the original size, double-click the view tab again.

Use this capability at your discretion throughout this Tutorial. If your views get messed up, use the Layout | Load Layout | PlanAhead Default command to reset them back to default positions.

---

**Updating the Project with a New Netlist and New Constraints**

**Step 4**

PlanAhead Projects can be updated with a complete top-level or module-level netlist received from synthesis. The process basically consists of merging the new netlist with the existing Project. All Floorplans in a Project are affected by the netlist update.

Occasionally, the netlist changes can be significant and may include new UCF constraints files. This example design update involves significant design changes that include removing the *uartInst* logic module, adding new I/O port definitions and revising timing constraints.

Clear all of the timing and placement constraints from the design to prepare for a new UCF file.

1) Select the **Constraints** view tab (next to the Netlist view tab)

2) Select the **Group by type** icon in the Constraints view to display the list of all timing constraints (**Figure 5b-15**).
3) Select the top constraint in the Constraints view list and press the Ctrl-A key to select all of the constraints in the list

4) Select the popup menu Delete command and click OK to confirm delete

5) Select the Tools | Clear Placement Constraints command

6) Select Both in the Clear Placement Constraints dialog and click Next

7) Click Next to accept the defaults in the Instance Types to Unplace dialog

8) If prompted, select the following options: (Figure 5b-16)

**Figure 5b-15: Deleting All Timing Constraints**
Unplace all xxx fixed instances

Unplace all xxx fixed ports

Figure 5b-16: Clear All Fixed Placement Constraints

9) Click Next

10) Click Finish to remove all of the placement constraints in the Floorplan

Examine the ROOT design statistics, then perform the top-level netlist and constraints update.

1) In the Physical Hierarchy view, select the ROOT design partition

2) View the utilization statistics in the Pblock Properties view prior to updating the netlist and note the SLICE utilization (Figure 5b-17)
3) Note the SLICE and RAM16 statistics

4) Select File | Update Netlist

5) Select the Replace the entire design option and click Next

6) Select the browser icon and navigate to the `<Install_Dir>/PlanAhead_Tutorial/labs/design_files/rev1_top_update` directory

7) Select the `802RCV_c13_SPT.edf` netlist file and click Open
8) Use the **Add** button browser in the *Netlist directories* field to select the `<Install_Dir>\PlanAhead_Tutorial\labs\design_files` directory which contains some of the original core netlist files used in the design (*Figure 5b-18*)

![Update Netlist](image)

**Figure 5b-18: Import New Netlist to Update Project**

Note the lower-level netlists that will be imported as needed by the top-level netlist.

9) Click **Next**, wait until the design imports and then click **Finish** on the Summary page and wait while the Project updates

10) Click **Yes** to save changes to the Floorplan

11) In the Physical Hierarchy view, select the **ROOT** design partition

12) View the updated utilization statistics prior to updating the netlist and note the increased SLICE and RAM utilization (*Figure 5b-19*)
Figure 5b-19: Import New Netlist to Update Project

Notice the significant logic change. There is much more device logic now being utilized.

1) Import the revised UCF constraints and update the floorplan accordingly.

1) Select the File | Import Constraints command
2) Use the Add button browser in the Constraints files field to select 
<Install_Dir>\PlanAhead_Tutorial\labs\design_files\rev1_top_update\802RCV_c13_SPT.ucf and click OK (Figure 5b-20)

![Import Constraints Dialog Box](image)

**Figure 5b-20: Import New Constraints to Update Floorplan**

3) Select the Console view tab and examine the netlist and UCF update reports (Figure 5b-21)

![Console Window](image)

**Figure 5b-21: Examine the Update Command Logs**

Notice the messages indicating successful UCF import.
**Saving the Floorplan and Closing the Project**

**Step 5**

1. Select **File | Exit**

2. If prompted, click **Yes** to Save the Floorplan and click **OK** in the PlanAhead close confirm dialog.

**NOTE:** It is recommended that users exit PlanAhead after each lab, as instructed. Each lab relies on the pre-run ISE implementation data matching the floorplan for which it was generated. Users may get out of sync by failing to exit PlanAhead after each lab.

**Conclusion**

In this lab, you launched the PlanAhead software and created a project and a new Floorplan. You also explored some of the features of the Device view and Netlist view.

The next step is to perform further design analysis in preparation for design partitioning and area floorplanning.
Design Analysis and Exploration Lab

Introduction

This lab introduces the analysis features of the PlanAhead software that enable early detection of potential design issues, alternate device exploration, initial floorplanning direction and post implementation analysis. By running Design Rule Checks (DRC), you can quickly resolve constraint conflicts that would otherwise cause implementation errors. A quick Weighted Average Simultaneous Switching Output “WASSO” analysis can be performed to highlight potential IO pin signal switching issues. You can target alternate devices and choose the optimal device by analyzing device utilization statistics. Logic can be explored in the Netlist, Logic Hierarchy and Schematic views. You can view, modify, or create constraints assigned in the design. You can analyze placement and timing results from ISE implementation to uncover bottlenecks and to drive initial floorplanning decisions.

Objectives

After completing this lab, you will be able to:

- Display the design resource estimates
- Run the Design Rule Checks
- Run WASSO analysis
- Graphically view the logical hierarchy
- Explore logic in the Schematic
- Modify the design timing constraints
- Import ISE placement and highlight module placement locations
- Import Tree timing results and view critical paths

Procedure

This lab comprises 8 primary steps: you will display design resource estimates, run Design Rule Checks (DRC), run WASSO analysis, view the logical hierarchy, explore logic in the schematic, change the design timing constraints, import ISE placement results and highlight module placement, and finally import and analyze ISE Tree timing results.

Displaying Design Statistics  Step 1

Launch the PlanAhead software and open the Project_2
<Install_Dir>\PlanAhead_Tutorial\labs\projects\Project_2\Project_2.ppr
1) On Windows, select the Xilinx PlanAhead 10.1 Desktop icon or Start | Programs | Xilinx | 10.1 | PlanAhead | PlanAhead

On Linux or UNIX, change directory in the <InstallDir>/PlanAhead_Tutorial/labs directory and enter the planAhead

**NOTE:** If you are using the default PlanAhead Tutorial location at <PlanAhead InstallDir>/testcases/PlanAhead_Tutorial, the Open an Example Project link on the Getting Started page can be used throughout the tutorial to open the Tutorial design.

2) Select File | Open Project

3) Browse to the <Install_Dir>|PlanAhead_Tutorial|labs|projects|Project_2 folder

4) Select Project_2.ppr and click Open

View the resource estimates of the entire design.

The PlanAhead utilization analysis provides design statistics to help determine the optimal device for the design. Multiple device types can be explored to determine the best overall utilization and performance estimations.

1) Select the ROOT design in the Physical Hierarchy view (**Figure 7b-1**)

![Physical Hierarchy View](image)

**Figure 7b-1: Physical Hierarchy View**

2) View the Physical Resource Estimates displayed in the Pblock Properties view

3) If needed, select the Statistics tab in the Pblock Properties view (**Figure 7b-2**
4) Scroll through the design statistics displayed in the Pblock Properties view.

Notice the device resource utilization for each type of logic element, carry chain count and longest length, Tristate Buffer Statistics, the Clock Report and Clock Region Statistics, IO utilization and the net and instance counts.

If this design contained any RPMs, the RPM count and maximum size information would also be displayed.

**Running Design Rule Checks (DRC)**

**Step 2**

Run the Design Rule Checker (DRC).
It is a good idea to run the Design Rule Checker after importing a design and before implementation.

1) Select **Tools | Run DRC**

The Run DRC dialog appears (**Figure 7b-3**). You will use the default options to run all rule checks.

![Run DRC dialog](image)

**Figure 7b-3: Running DRC**

2) Click **OK** to run the checks

Errors, Warnings and Information messages appear in the DRC Results window. Errors display a red icon, Warnings display an orange icon and Information messages display a yellow dialog.

Notice that the **DRC Results** view is displayed and indicates one IOB Information message (**Figure 7b-4**).
3) Click the **IOSR #1** Information message in the in the DRC Results view. The Violation Properties view is displayed describing the violation. Links are provided to identify the problem objects.

4) Click the **receiveADCValid** link and notice the Netlist and Device views highlight that object (Figure 7b-5)

5) Close the DRC Results view by clicking on the view tab **Close X** icon

---

**Running WASSO Analysis**

Run a WASSO analysis using default values.

WASSO analysis is an excellent way to identify potential signal integrity issues related to too many simultaneously switching signals. The analysis is performed across the entire device, on each IO bank and each IO pin. Results are presented in a table viewer.
1) Select the Tools | Run WASSO Analysis command

The Run WASSO Analysis dialog appears (Figure 7b-6).

![Run WASSO Analysis dialog](image)

**Figure 7b-6: Running WASSO Analysis**

2) Click OK

Notice the WASSO results table displayed in the Workspace view (Figure 7b-7).
### Device Parameters
- Maximum ground bounce: 600.0 mV
- Capacitance per output driver: 15.0 pF

### Board Parameters
- Board Thickness: 62.0 mils
- Finished via diameter: 12.0 mils
- Pad to via breakout length: 33.0 mils
- Breakout width: 12.0 mils
- Other PCB parasitic inductance: 0.0 nH
- Socket inductance: 0.0 nH

### Table: WASSO Analysis

<table>
<thead>
<tr>
<th>Package</th>
<th>WASSO Allowance</th>
<th>WASSO Utilization</th>
<th>WASSO Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>Banks</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Bank 0</td>
<td>100%</td>
<td>42.7%</td>
<td>OK</td>
</tr>
<tr>
<td>Bank 1</td>
<td>100%</td>
<td>20%</td>
<td>OK</td>
</tr>
<tr>
<td>Bank 2</td>
<td>100%</td>
<td>1.3%</td>
<td>OK</td>
</tr>
<tr>
<td>Bank 3</td>
<td>100%</td>
<td>9.3%</td>
<td>OK</td>
</tr>
<tr>
<td>Bank 4</td>
<td>100%</td>
<td>14.7%</td>
<td>OK</td>
</tr>
<tr>
<td>Bank 5</td>
<td>100%</td>
<td>42.7%</td>
<td>OK</td>
</tr>
<tr>
<td>Bank 6</td>
<td>100%</td>
<td>24%</td>
<td>OK</td>
</tr>
<tr>
<td>Bank 7</td>
<td>100%</td>
<td>24%</td>
<td>OK</td>
</tr>
<tr>
<td>Neighbors</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Bank 0,1</td>
<td>100%</td>
<td>31.3%</td>
<td>OK</td>
</tr>
<tr>
<td>Bank 1,2</td>
<td>100%</td>
<td>10.7%</td>
<td>OK</td>
</tr>
<tr>
<td>Bank 2,3</td>
<td>100%</td>
<td>5.3%</td>
<td>OK</td>
</tr>
<tr>
<td>Bank 3,4</td>
<td>100%</td>
<td>12%</td>
<td>OK</td>
</tr>
<tr>
<td>Bank 4,5</td>
<td>100%</td>
<td>28.7%</td>
<td>OK</td>
</tr>
<tr>
<td>Bank 5,6</td>
<td>100%</td>
<td>33.3%</td>
<td>OK</td>
</tr>
<tr>
<td>Bank 6,7</td>
<td>100%</td>
<td>24%</td>
<td>OK</td>
</tr>
<tr>
<td>Bank 7,0</td>
<td>100%</td>
<td>33.3%</td>
<td>OK</td>
</tr>
</tbody>
</table>

**Figure 7b-7: Running WASSO Analysis**

3) Close the WASSO results view by clicking on the Close X icon in the view tab (Figure 7b-7)
Use the Instance Hierarchy view to display the logical hierarchy tree graphically. The trees show relative module sizes and location. Select timing paths and view where the logic resides in the design hierarchy.

1) Expand the **receiver/RChainTopInst** module in the Netlist view
2) Select the **CONVDECOD (viterbi)** module in the Netlist view
3) Select **Show Hierarchy** popup menu command
4) The Hierarchy view is invoked in the Workspace, as shown below (Figure 7b-8)

![Hierarchy View](image)

**Figure 7b-8: Instance Hierarchy View**

Notice the hierarchical relationship of the modules as well as the relative sizes. The selected logic module is displayed in the Hierarchy view. It can be useful to visualize the module location and relative size prior to floorplanning. Users can select modules directly from this view for floorplanning. Logic selected in other views is highlighted in the Hierarchy view.

5) Select any module in the Hierarchy view and notice it is also selected in the Netlist view
6) Click the **Close Window** icon to close the Hierarchy view
Exploring Logic in the Schematic

Step 5

Experiment with the schematic traversal and expansion commands to explore logic in the Schematic view. Expand the clk pin and all of the logic inside of the viterbi module. Selectively remove logic from the schematic and traverse the logic modules.

1) With the CONVEDECOD module selected, select the Schematic popup menu command

2) Notice the module displayed in the Schematic view (Figure 7b-9)

Figure 7b-9: Viewing Selected Logic in the Schematic

Notice that there are not too many interface signals associated with the viterbi which makes it a good candidate to isolate with floorplanning.

3) Double-click the clk pin on the viterbi schematic symbol

4) **Zoom Fit** the Schematic view and notice the net automatically expands to the clock buffer (Figure 7b-10)
5) Double-click the `recsymbo` bus pin on the viterbi schematic symbol.

6) **Zoom Fit** the Schematic view and notice the bus automatically expands to show all bits (Figure 7b-11).

7) Double-click the viterbi schematic symbol and notice the schematic now displays the logic contents of the viterbi module (Figure 7b-12).
8) Select the Select Area Toolbar icon and draw a rectangle around some of the logic displayed.

9) Click the Remove selected logic from schematic icon in the Schematic view and notice the logic is no longer displayed.

10) Select the Previous Schematic icon in the Schematic view several times to return to previous logic viewed in this Schematic viewing session.

11) Click the Close Window icon to close the Schematic view.

12) Click the Collapse All icon in the Netlist view.

Figure 7b-12: Expand Selected Modules in the Schematic
Note: There are many Schematic traversal commands and options that enable a variety of exploration options which are not all covered in this lab.

Modifying the Design Timing Constraints  

Step 6

Modify the properties of the TS_GCLK_F constraints to change the 16.5ns target to 16ns. View the list of all design timing constraints.

1) Click on the Constraints tab next to the Netlist tab to display the Constraints view (Figure 7b-13)

2) Click to expand the Clk period..., and Timespec period... constraints

3) Select the TIMESPEC TS_GCLK_F constraint

4) Observe the Constraint Properties view (Figure 7b-13)

Figure 7b-13: Timing Constraint Properties

Note that you can modify the constraint value, duty cycle, and priority. As changes are made, the Apply Toolbar button will appear in the Properties dialog and must be clicked for any changes to take effect.

5) Change the value of the Width: field to 16 and click the Apply button at the bottom of the Constraint Properties view when it appears

6) Toggle the Group by type Constraints view Toolbar button off to view all the design timing constraints in list fashion as opposed to categorically (Figure 7b-14)
7) Change the two TIMEGRP ... OFFSET constraints with 16.5ns values to 16 by selecting them in the Constraints view, changing the appropriate fields in the Constraints Properties view and clicking **Apply**

Examine the ability to create a new constraint.

1) In the Constraints view Toolbar, click the **Create new constraint** Toolbar button.

Examine the options in the dialog (Figure 7b-15).

---

**Figure 7b-14: Viewing a List of all Timing Constraints Defined**

**Figure 7b-15: Creating a New Constraint**
2) Click **Cancel** to close the New Timing Constraint dialog

3) Click the **Netlist** view tab to display the Netlist
Importing and Analyzing ISE Implementation Results  Step 7

Import the implemented result

<Install_Dir>\PlanAhead_Tutorial\labs\projects\Project_2\Project_2.runs\orig_fp_16ns\run_3\orig_fp_16ns_routed.xdl

A floorplanning strategy can be determined by examining previous implementation results. Placement and timing results can be imported and analyzed using the PlanAhead interface. Failing path groups can be identified and floorplanned.

1) Select File | Import Placement

2) Ensure the Netlist button is selected

3) Using the File name browser, navigate to the following directory

<Install_Dir>\PlanAhead_Tutorial\labs\projects\Project_2\Project_2.runs\orig_fp_16ns\run_3

4) Select orig_fp_16ns_routed.xdl and click Open (Figure 7b-16)

Figure 7b-16: Importing an Implemented Result
5) Click **OK**

The Device view displays the results from ISE placement.

6) If the green I/O nets are displayed, toggle off the **Show/Hide I/O Nets** Toolbar button.

Expand the `receiver/RChainTopInst` and `receiver/synchroTOPInst` modules, then highlight the entire design with cycling colors to easily view placement.

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

1) Click **Collapse All** Toolbar button in the **Netlist view**

2) In the Netlist view, expand the `receiver` module, then expand the `receiver/RChainTopInst` and `receiver/synchroTOPInst` modules

3) Use the **Shift** key to select all of the modules as shown in Figure 7b-17
Figure 7b-17: Selecting many Netlist Modules

4) Use the popup menu and select **Highlight Primitives | Cycle colors** option

Note that there may be a slight delay for highlighting to take effect.

The primitives in each module are highlighted in a different color as shown in Figure 7b-18.
Notice the Netlist view has colored icons to match the primitive logic highlight colors.

Figure 7b-18: Highlighting Module Placement

5) Click the **Unhighlight All** Toolbar button

**Importing and Analyzing Trce Timing Results**

**Step 8**

Import the following Trce timing result file:

```
<Install_Dir>/PlanAhead_Tutorial/labs/projects/Project_2/Project_2.runs/orig_fp_16ns/run_3/orig_fp_16ns.twr
```

Timing results after implementation (TWR) can be analyzed to drive the floorplanning effort. The path sorting and selection techniques available in the TimeAhead environment can be used with imported TWR report data.

1) If available, click the **Unselect All** Toolbar button
2) Select **File | Import TRCE Results**

Ensure the **Netlist** button is selected.

3) Navigate to select the File Name:

```<Install_Dir>\PlanAhead_Tutorial\labs\projects\Project_2\Project_2.runs\orig_fq_16ns\run_3\orig_fq_16ns.twr (Figure7b-19)```

![Import TRCE Results](image)

**Figure 7b-19: Importing Timing Results**

4) Click **OK**

Examine the top path in the Timing view. Select all paths that begin at the same source and display them in the Schematic view. Examine the Pblock creation options, to create a Pblock from the Schematic.

1) Select the first path in the list of paths presented in the Timing Results view (**Figure 7b-20**)

As paths are selected in the Timing Results view, the Path properties view displays the details of the timing path.

Because you have imported implementation placement results, the path is also highlighted in the Device view. This cross-selecting quickly shows the entire...
path and allows you to take appropriate floorplanning steps to improve the timing.

Figure 7b-20: Examining Timing Paths

2) Click the **Maximize/restore** icon in the Timing Results view

3) Use the scroll bar and **Shift** key and to all of the **failing** paths that begin at `receiver/synchroTOPInst/autoCORRi/inputMem/B5`

4) Use the popup menu to select **Schematic**

5) Click the **Maximize/restore** icon in the Timing Results view to return the view

All of the instances on the selected paths are displayed in the Schematic view.

6) Use the **Collapse All** Toolbar button in the Netlist view

7) Use the popup menu in the Schematic view to select the **Select Primitive Parents** command which will select the smallest parent modules that contain all of the instances on the selected paths (**Figure 7b-21**)
Figure 7b-21: Selecting Path Object Parent Modules for Floorplanning

Notice the corresponding logic modules selected in the Netlist view.

Clear the ISE imported placement constraints from the Floorplan. Exit PlanAhead.

1) Select the **Device** view tab in the Workspace to display the Device view

2) Select the **Tools | Clear Placement Constraints** command

   The Clear Placement Constraints dialog will appear (**Figure 7b-22**)
3) Click **Next** to elect to clear Instances only

The next dialog in the Clear Placement Constraints wizard allows you to filter which logic types to clear (**Figure 7b-23**)

4) Click **Next** to clear all non-I/O related placement constraints
5) Click **Finish** in the final Clear Placement Constraints confirmation dialog to clear the Instance placement constraints.

6) Select **File | Exit**

7) When prompted, click **Yes** to save changes and **OK** to confirm close.

**Conclusion**

In this lab, you examined the utilization statistics of the entire design. You used Design Rule Checks to flag floorplanning problems and constraint conflicts before you ran the implementation. You ran WASSO analysis to determine if the assigned pin configuration is subject to signal noise related issues.

You explored logic modules and signals in the Schematic and Hierarchy views.

You modified the design constraints based on initial timing estimations and examined how to create new constraints.

You imported implementation results and highlighted module placement. You imported Trace timing results and analyzed critical path objects in the schematic. You then selected the parent modules of those path objects for possible floorplanning activity. You cleared the ISE assigned placement constraints.
Design Partitioning Lab

Introduction

This lab introduces the concept of floorplanning. You can selectively partition a design into physical blocks (Pblocks). This lab also focuses on using the interactive tools of the PlanAhead software to quickly create a top-level floorplan. This floorplan allows the design’s interconnect flow and logic module sizes and to be visualized. This type of floorplan is not usually a very effective floorplan to maximize performance. It should be used as a starting point only to provide you with information about how the logic will fit into the device and the overall data flow through the device. Logic requiring resources such as BRAMs, DSP48s, PPCs, etc. can be easily identified to help guide floorplanning. Large modules can be further partitioned to break up the lower level modules and show the connectivity at a finer granularity. If it is determined that lower level modules need to be floorplanned, the top-level floorplan can help you determine where the logic should be constrained in the device.

Objectives

After completing this lab, you will be able to:
- View the top-level data flow of the design
- Set PlanAhead viewing options
- View a clock domain
- View Bundle Net content
- Use Connectivity analysis to place Pblocks
- Use Resource Utilization Statistics to examine and shape Pblocks

Procedure

This lab comprises seven primary steps: you will partition and place the top-level design; adjust the viewing options; partition and place lower-level Pblocks; delete Pblocks; view the clock domains in the device; display the Bundle Net properties and net content, and finally adjust the Pblock placement and size.

Partitioning and Placing the Top-Level Design  Step 1

Launch the PlanAhead software and open the Project:
<Install_Dir>\PlanAhead_Tutorial\labs\projects\Project_2.

1) On Windows, select the Xilinx PlanAhead 10.1 Desktop icon or Start | Programs | Xilinx | 10.1 | PlanAhead | PlanAhead
On Linux or UNIX, change directory in the
<InstallDir>/PlanAhead_Tutorial/labs directory and enter the planAhead

2) On the Getting Started jump page, select the Project_2 link under the Open a Previously Created Project area

If the link does not exist, Select File | Open Project and browse to select the
<Install.Dir>/PlanAhead_Tutorial/labs/projects/Project_2/Project_2.ppr file
and click Open

Use the New Pblocks command to partition the ROOT and receiver levels of the
design.

1) Select the channel module in the Netlist view,
2) Use the Shift key to select all three top-level modules
3) Select the Tools | New Pblocks command
4) Click Next to accept the three selected modules
5) Click Next again to accept the default naming strategy
6) Click Finish on the Summary page to view the top-level Pblocks in the
Physical Hierarchy view (Figure 9b-1)

Figure 9b-1: Partitioning the Top-level Design

Use the Set Pblock Size command to place the Pblocks you just created.

1) Select the pblock_LED and click the Set Pblock Size Toolbar icon
(Make sure you select this icon and NOT the Draw Pblock icon)
2) Draw a small Pblock rectangle anywhere in the Device view
3) Repeat the Set Pblock Size steps for the other two Pblocks (Figure 9b-2)

![Figure 9b-2: Place Top-Level Pblocks](image)

4) If connectivity is not displayed, make sure that the **Show/Hide I/O nets** and **Show/Hide Bundle Nets** Toolbar buttons are toggled **on** (Figure 9b-3)

![Figure 9b-3: Displaying Connectivity](image)

Use these display toggle icon buttons at your discretion throughout this Tutorial to turn on an off connectivity display.

Adjust the placement and size the Pblocks based on connectivity and resources.

1) Adjust the placement of the Pblocks to untangle the connectivity (Figure 9b-3)
2) Select the `pblock_channel` Pblock and view the Pblock Property Statistics

3) Reshape the Pblock in the lower right corner to satisfy the resources required (Figure 9b-4)

4) Select the `pblock_LED` and reshape the Pblock to satisfy the resources required (Figure 9b-4)

5) Select the `pblock_receiver` and reshape the Pblock to satisfy the resources required (Figure 9b-4)

---

**Adjusting the Viewing Options  
Step 2**

Set the View Options to make some objects not selectable.

While creating floorplans and manipulating Pblocks, is often easier when certain objects are made non-selectable, such as IO nets, bundle nets and instances.

1) Select **Tools | Options**

2) Click the **Themes** button on the left to invoke the Color themes dialog

3) Select the **Device** tab of the Themes dialog

4) Click the **Select** column boxes for **Assigned Instance**, **IO Net** and **Core Net** to turn them **off** as shown in **Figure 9b-5**
Figure 9b-5: Setting View Options

5) Click the Save As button, enter any name and click OK

Note that you can save your own custom view option configurations for use later.

6) Click OK

Partitioning and Placing Lower-Level Pblocks  Step 3

Use the New Pblocks command to further partition the pblock_receiver Pblock. Place Pblock rectangles for the four Pblocks you created in the previous step.

The New Pblocks command can be used to iteratively walk down the design hierarchy, creating Pblocks at each level or to partition modules selectively. This helps to breakup larger modules and shows a finer level of connectivity granularity.
1) In the Netlist view, expand the receiver module

2) Select the busMuxWrapInst module and use the Shift key to select all four modules under the receiver

3) Select the Tools | New Pblocks command and click Next twice to accept the defaults values

4) Click Finish to create the four lower level Pblocks

The new lower-level Pblocks appear in the Physical Hierarchy view (Figure 9b-6).

![Physical Hierarchy](image)

**Figure 9b-6: Creating Lower-Level Pblocks**

Note the different Pblock icons indicating that the pblock_LED, pblock_channel and pblock_receiver Pblocks have rectangles defined, while the other Pblocks do not have rectangles defined yet.

Place the lower-level Pblocks

Sometimes, you can more easily see the connectivity between Pblocks if you create small rectangles in the Device view for each Pblock. At this point in the design, visualizing the data flow is more of a concern than Pblock size.

1) Select each of the lower-level Pblocks and use the Set Pblock Size command to draw small rectangles within the larger receiver Pblock

The Device view should look similar to **Figure 9b-7**.
Figure 9b-7: Placing Lower-Level Pblocks

Note the thick colored lines that represent bundles of nets that communicate between the Pblocks.

Notice that the Physical Hierarchy view displays the relationship of the lower-level Pblocks under the pblock_receiver (Figure 9b-8)
Deleting Pblocks  

Step 4

Delete the *pblock_receiver* Pblock.

Now that the *pblock_receiver* Pblock has been subdivided into child Pblocks, you can delete the original Pblock and promote the lower-level Pblocks to the top level.

1) In the Physical Hierarchy view, select the *pblock_receiver* Pblock

2) Use the popup menu to select **Delete**

3) Click **OK** in the confirmation dialog (do not select Remove Pblock children)

The *pblock_receiver* Pblock is removed, and the lower-level Pblocks are promoted to the top level as shown in **Figure 9b-9**.
Figure 9b-9: Lower-Level Pblocks Promoted to Top-level

Note the size and location of each Pblock, the bundle net connections, and the I/O connections. This is a good way to view the top level connectivity and data flow of the design as well as estimate the resources required for each module.

Viewing Design Clock Domains

Effective floorplanning often revolves around proper placement of the synchronous elements in different clock domains.

Clock domains can be easily highlighted to visualize connectivity, enabling proper Pblock placement relative to clock regions. Unfortunately, this example design only has two global system clocks with one being very small. Otherwise clock domains can be highlighted to help drive the Pblock placement.

Select and mark the global clock net $CK_{MIC\_0\_c}$.

1) Select Edit | Find

2) Set the following options in the Find dialog:
   - Find: Nets
   - First field: Type
   - Second field: is
   - Third field: Global Clock
Ensure the *Unique Nets Only* button is switched **on**

Click **OK**

3) Select **CK_MIC_0_i** from the **Find Results** view list

Note the highlighted net leading to the *pblock_LED* Pblock (Figure 9b-10).

This indicates the location of the loads on this clock net are all inside the *pblock_LED*.

**TIP:** You could reshape this Pblock inside a specific Clock Region to isolate that clock.

![Figure 9b-10: Viewing a Clock Domain](image)

Use the Schematic View to see how the global clock net fans out to the primitives in the design. When you are finished, close the Schematic View.

1) With the global clock net still selected in the Find Results view, use the popup menu to select **Schematic**
The Schematic view displays the CK_MIC_O net connected to the LED module.

2) In the Schematic, select the \textit{CK_MIC_O} \textit{ibuf} \textit{BUFGP} instance (Figure 9b-11)

\begin{figure}
\centering
\includegraphics[width=\textwidth]{schematic}
\caption{Viewing the Clock Signal in the Schematic}
\end{figure}

3) Use the popup menu to select \textit{Append Fanout} | \textit{To Primitives}

4) Click the \textit{Zoom Fit} Toolbar button to fit the Schematic view.

Note the logic inside of the pbblock\_LED Pblock is expanded, showing all of the logic that is driven by the clock (Figure 9b-12).
Figure 9b-12: Expanding the Clock Logic Primitives

5) **Close** the Schematic view
Select the global clock net $GCLK_F_c$ to view which Pblocks are in the clock domain.

1) In the Find Results view, select $GCLK_F_c$ from the list

Notice that this clock drives all of the other Pblocks in the design.

**TIP:** For designs that contain several clocks, you may want to draw Pblock rectangles that match up with the clock regions in the target device to isolate clocks logic and to prevent clock contention.

2) Close the **Find Results** view by clicking on the Close X icon

3) Click the **Hide Core Nets** Toolbar button

4) If activated, click the **Unselect All** Toolbar button

---

### Displaying the Bundle Net Content  Step 6

Examine the properties of one of the bundle nets.

The PlanAhead software displays the connections between the Pblocks with bundle nets. The color and line width of a bundle net indicate the number of signals within the bundle. The settings for color, net count, and line width is configurable.

1) Select one of the colored bundle nets in the Device view

2) View the **Bundle Net Properties** view

3) Click the **Nets** tab in the Bundle Net Properties view (Figure 9b-13) to see a list of all the nets contained within the bundle between the two modules
Adjusting the Pblock Placement and Size  Step 7

The bundle nets and I/O flightlines help to visualize the connectivity in the design. Pblocks can quickly be arranged to untangle the connectivity. This provides early indications of data flow through the design and highlights where potential routing congestion may occur. To aid in Pblock placement, the Resource Statistics can be examined to determine which types of device resources are required.

Move and adjust the Pblock placement to untangle the Bundle nets and to optimize the data flow. A sample placement is shown in Figure 9b-14.

The Pblocks with the heaviest Bundle nets should be placed close together.

1) Select one of the **Pblock rectangles** in the Device view

2) Drag or reshape the **Pblocks** into more appropriate locations to try and untangle the bundles nets

Remember that most of these Pblocks were sized at 150%, so they will required some space between them.
The final Pblock placement should look something like Figure 9b-14.

Figure 9b-14: Adjusted Pblock Placement

You can size Pblocks based on resource utilization statistics. These statistics provide insight into logic resources required to help guide Pblock shaping.

Resize the \textit{pblock\_busMuxWrapInst} Pblock.

1) In the Device view or Physical Hierarchy view, select the \textit{pblock\_busMuxWrapInst} Pblock

This is the Pblock with the most I/O connections, which should be located at the top of the device.
2) Click the **Statistics** tab the **Pblock Properties** view and view the Physical Resources Statistics chart

Notice this module requires 8 block RAM16s.

3) Click the **Set Pblock Size** Toolbar button and redraw the Pblock to contain the top eight RAM sites (**Figure 9b-15**)  

**Note:** Select the **Set Pblock Size** Toolbar button and **NOT Draw Pblock mode** Toolbar button.

**TIP:** Where possible, draw Pblock rectangles so the edges align with the edges of the BRAM/DSP/MULT sites (**Figure 9b-15**).

4) Move other Pblocks that may be overlapping or nested out of the area (**Figure 9b-15**)  

Resize the remaining Pblocks based on SLICE logic.

1) In the **Device** view or **Physical Hierarchy** view, select one of the remaining Pblocks

2) Click the **Statistics** tab in the **Pblock Properties** view and display the Physical Resources Statistics chart
3) Using the chart, draw approximate sizes for the Pblock using the **Set Pblock Size** Toolbar button.

4) Repeat steps 1 through 4 for the remaining Pblocks

Do not spend time fine tuning these rectangles. Approximate sizes will be sufficient for now. Don’t worry if rectangles don’t have enough device resources.

The final floorplan should look like **Figure 9b-16**.

**Figure 9b-16: Final Top-level Floorplan**

Notice the data flow and relative modules sizes are clearly visible using this top-level floorplanning analysis approach.

Save the Floorplan as a new name. Exit PlanAhead.
1) Select **File | Save Floorplan As…**

   Make sure you select the *Save Floorplan As* and **NOT** the *Save Project As* command.

2) Enter the name **top-level_fp** and click **OK**

3) Select **File | Exit**

4) If prompted, select **Yes** to save changes and click **OK** to confirm close

**Conclusion**

The top-level floorplan created by the Automatic Partitioner and Placer tools is useful as seed floorplan. It can help you determine where to floorplan lower-level modules, if needed. It can also help you to visualize the data flow through the design as well as the size and content of the various design modules. Determining which modules require the most BRAM/DSP/MULT resources can help drive the floorplanning efforts.

However, this initial floorplan was not design with timing in mind. It will probably hurt performance if run through P+R. To improve timing start with only floorplanning the modules that are timing critical.
Implementation Lab

Introduction

This lab introduces the PlanAhead software’s integration with the ISE implementation tools. Initially, the design will be implemented without floorplanning and the results used to drive the floorplanning effort. The initial floorplan is exported and run through the implementation tools. Results are then imported back into the PlanAhead software for further analysis and modification. ExploreAhead will also be used to queue and monitor multiple runs using various implementation strategies.

Objectives

After completing this lab, you will be able to:
- Run ExploreAhead to configure and launch multiple ISE implementation runs
- Monitor and review ExploreAhead results
- Simultaneously view the runs for multiple Floorplans in the Project
- Export the floorplan for using ISE outside of PlanAhead

Procedure

This lab comprises four primary steps: you will configure ExploreAhead runs and view the Launch Run features; review ExploreAhead results, simultaneously manage the runs for multiple Floorplans in the Project and finally export the floorplanned design for ISE use outside of PlanAhead.

Running ExploreAhead to Configure & Launch Runs Step 1

Launch the PlanAhead software and open Project_2
<Install_Dir>\PlanAhead_Tutorial\labs\projects\Project_2\Project_2.ppr. Run ExploreAhead to create 6 runs with the target 16ns global clock timing constraint. Review Run Properties to setup, monitor and launch runs. For time and disk cpu reasons, we will not launch any ExploreAhead runs in this Lab. Pre-run ISE results are used to complete the labs.

For training class situations, please do not launch the tools as it will impede others system performance significantly.

1) On Windows, select the Xilinx PlanAhead 10.1 Desktop icon or Start | Programs | Xilinx | 10.1 | PlanAhead | PlanAhead
On Linux or UNIX, change directory in the 
<InstallDir>/PlanAhead_Tutorial/labs directory and enter the planAhead

2) Select File | Open Project and browse to the 
<Install_Dir>/PlanAhead_Tutorial/labs/projects/Project_2 directory

3) Select Project_2.ppr and click Open

The Project will open with the previously created top-level fp Floorplan active.

4) Select File | Close Floorplan and click OK to confirm

5) Select File | Reopen Floorplan and select orig_fp

6) Select Tools | Run ISE Place & Route with ExploreAhead

7) Make sure the Flow field is set to ISE 9

8) Select the Select All button

9) Deselect the middle two strategies that involve resynthesis (Figure 11b-1) and click OK

![Create ExploreAhead Runs](image)

**Figure 11b-1: Creating and Configuring ExploreAhead Runs**
Note the runs displayed in the ExploreAhead view with different ISE strategies assigned (Figure 11b-2).

**Figure 11b-2: Viewing ExploreAhead Runs**

10) Select the `run_2` line in the ExploreAhead Runs list (Figure 11b-3)

Note the Run Properties **General** tab allows modification of the Run name or description and will indicate the disk location of the run data once the Run is launched. (Figure 11b-3)

**Figure 11b-3: General Run Properties**
11) Select the **Options** tab in the Run Properties view

Note the Run Properties Options tab allows customization of ISE command line options. Various strategy saving options exist in the popup menu. (Figure 11b-4)

12) Select the `map –pr` option line and using the popup arrow on the right side of the option line, select the `<none>` option (Figure 11b-4)

![Run Properties Options](image)

**Figure 11b-4: Setting Command Options for Runs**

Note that as command options are selected, a description of the option intent and available arguments are displayed below.

13) Click **Apply** to accept and save the new option for this Run only

Notice the **Strategy** name in the ExploreAhead Runs view is now displayed with an * (EA #1*) indicating it has been modified.

Invoke the Launch Runs dialog. Select Cancel to not invoke ISE.
1) Use the Shift key to select **run_1** through **run_6**

2) Use the popup menu to select **Launch Runs** (Figure 11b-5)

![Launch Runs](image)

**Figure 11b-5: Launching Runs**

Notice the options to generate scripts only and to export fixed and unfixed placement constraints.

Notice the ability to run simultaneously using the multiple processors available on the host machine.

3) Press **Cancel** to NOT launch runs.

**Reviewing ExploreAhead Results**  
**Step 2**

Open the **orig_fp_16ns** Floorplan. Review ExploreAhead results. Review how Runs can be managed for the entire Project. Close the **orig_fp** Floorplan.

1) Select **File | Reopen Floorplan | orig_fp_16ns**
Note the ExploreAhead Results view displays the results for each run. The status is updated live as ISE is running or when the Project is closed and reopened in PlanAhead. The Results list has a Progress bar that shows overall progress from ngdbuild through xdl as well as the Timing Score. The Monitor tab of the Run Properties view displays the standard command output status as commands are running (Figure 11b-6).

![Run Properties](image)

![ExploreAhead Runs](image)

**Figure 11b-6: Monitoring Runs**

2) Select the *run_1* line and click the Monitor tab in the Run Properties view

Note the ISE “standard out” command log is displayed.
3) Scroll the Run Properties Monitor view to examine the ISE command messages

4) Click the Reports tab in the Run Properties view

5) Double-click the Par Report to view the ISE par command .par log file in the Workspace (Figure 11b-7)

6) Scroll down the report to the timing section

Figure 11b-7: Viewing ISE Report Files

Note the design is missing timing by 1.065ns.

7) Close the PAR Report by clicking on the Close X icon

Import run_3 placement and timing results.

1) Select the run_3 line in the ExploreAhead Runs view and use the popup menu to select Import Run

2) Review the import options and click OK (Figure 11b-8)
3) Note the placement and timing results were imported for run_2 (Figure 11b-9)

![Figure 11b-9: View Imported Results](image)

### Launching Runs from Multiple Floorplans Simultaneously - Step 3

- View the Run status for all open Floorplans from the Project view.
It is advantageous to be able to queue multiple runs using varying timing or physical constraints. This allows experimentation to understand which options work best.

1) Select the **Project_2** Project tab

2) Select the **ExploreAhead Runs** view tab

3) In the ExploreAhead Runs view, select the **Maximize/restore** view icon to maximize the view (Figure 11b-10)

![Figure 11b-10: Viewing Runs from Multiple Floorplans](image)

Note that Runs from multiple floorplans can be launched and managed from within the Project view.

4) In the ExploreAhead Runs view, select the **Maximize/restore** view icon again to restore the view

5) Select the **Floorplan - orig_fp** Floorplan tab to activate the Floorplan

---

**Exporting the Floorplan for Implementation**  
**Step 4**

LExport the floorplan into a directory called

<Install_Dir>/PlanAhead_Tutorial/labs/projects/Project_2/orig_fp_export

PlanAhead facilitates using an independent ISE implementation flow outside of the integrated ExploreAhead environment by exporting an EDIF format netlist and UCF file containing the PlanAhead assigned physical constraints.

1) Select **File | Export Floorplan**
2) Browse to select the following **Directory Name**: 
<Install_Dir>\PlanAhead_Tutorial\labs\projects\Project_2\orig_fp_export\ (Figure 11b-11)

The Export Floorplan dialog should look like **Figure 11b-11**.

![Figure 11b-11: Export the Floorplan for Implementation](image)

3) **Click Next**

Review the Export Floorplan Summary (**Figure 11b-12**). The directory and file names are clearly listed.
4) Click **Finish** and wait while a moment the files are created

Note that these files can be used directly for ISE implementation. Any NGC format core files used in the design will be copied to this export directory for implementation.

5) Select the **Floorplan - orig_fp** Floorplan tab then click the **Close** X icon

6) Select **Yes** to Save floorplan, if prompted and **Yes** to confirm closing the Floorplan

7) Select **File | Exit** and click **Yes** to Save if prompted and to **OK** confirm close

**Conclusion**

In this lab, you saw how to use ExploreAhead to configure multiple ISE runs using different implementation strategies. The status of the runs is monitored and
is managed from within the Project. Implementation results are easily imported for further analysis.

Implementation can be setup and run without exiting the PlanAhead Desktop. You have access to all of the implementation options through the Run ISE Place & Route dialog.

You also saw that exporting a floorplan for implementation outside of PlanAhead is a simple process. ISE results for runs done outside of PlanAhead can be imported.
Floorplanning Lab

Introduction

This lab expands upon the concept of floorplanning to achieve timing closure by using initial timing analysis and previous implementation results to create a floorplan aimed at avoiding timing, utilization, and routing congestion issues.

The key to a good floorplan involves creative partitioning of the design, often driven by design-specific issues. Various factors can contribute to the partitioning strategy, such as clock regions, I/O interfaces, expected design iterations, the use of IP cores, incremental design, etc.

Objectives

After completing this lab, you will be able to:

- Analyze timing results from multiple ExploreAhead implementation Runs
- Assign critical path logic to a new Pblock
- Create Pblocks for other critical modules identified
- Assign placement constraints manually
- Analyze implementation results of critical logic Floorplan
- Remove placement constraints

Procedure

This lab comprises six primary steps: you will import and analyze implementation results; remove the placement constraints; assign critical path logic to a new Pblock; identify and floorplan other potential timing critical logic by exploring timing results from other ISE runs; assign placement constraints manually; and finally view implementation results for the critical logic Floorplan. Below each general instruction for a given procedure, you will find accompanying step-by-step directions and illustrated figures that provide more detail for performing the general instruction. If you feel confident about a specific instruction, feel free to skip the step-by-step directions and move on to the next general instruction in the procedure.

Analyzing Implementation Results

A floorplanning strategy can be determined by examining previous implementation results. Placement and timing results can be imported and analyzed using the PlanAhead interface. Failing path groups can be identified and floorplanned.
Launch the PlanAhead software and open the Project_2

1) On Windows, select the Xilinx PlanAhead 10.1 Desktop icon or Start | Programs | Xilinx | 10.1 | PlanAhead | PlanAhead

   On Linux or UNIX, change directory in the
   <InstallDir>/PlanAhead_Tutorial/labs directory and enter the planAhead

2) Select File | Open Project

3) Browse to the <Install_Dir>/PlanAhead_Tutorial/labs/projects/Project_2 directory

4) Select Project_2.ppr and click Open

   Ensure the Floorplan - orig_fp_16ns tab is available and selected at the top of
   the PlanAhead view. If not, select File | Reopen Floorplan | orig_fp_16ns

   Import and analyze the run_3 Trace timing results to understand which modules
   require floorplanning.

   1) Select ExploreAhead Results view tab

   2) Select the run_3 line in the ExploreAhead Results view

   3) Use the popup menu to select the Import Run command and click OK
      (Figure 13b-1)

   Figure 13b-1: Import Timing Results

   4) Click the Maximize/restore icon in the new Timing Results view

   5) Note that all of the failing paths are within the
      /receiver/synchroTOPInst/autoCORRi/….
6) Use the **Shift** key and scroll bar to select all of the failing red paths that begin at `receiver/synchroTOPInst/autoCORRi/…`

7) Click the **Maximize/restore** icon in the Timing Results view to restore the view.

Launch the Schematic view for all selected path instances.

1) Use the popup menu to select the **Schematic** command

   All of the instances on the selected paths are displayed in the Schematic view.

2) Click the **Collapse All** Netlist view Toolbar button.

3) In the Schematic view, use the popup menu to select the **Select Primitive Parents** command as shown in **Figure 13b-2**

![Figure 13b-2: Selecting Modules to Floorplan](image)

Note the modules selected in the Netlist view.

If the logic modules should somehow become unselected during the next few steps, simply select them again in the Schematic view.

View relative logic location in the Instance Hierarchy view

1) Use the popup menu to select the **Show Hierarchy** command

2) Zoom in to view the relative location of the selected logic in the context of the entire design logic hierarchy (**Figure 13b-3**)
Figure 13b-3: Highlighting Logic in Context of Entire Design

3) Notice from the relative sizes that the selected logic comprises a little more than half of the parent autoCORRi module logic

4) Close the Hierarchy view by clicking on the view tab Close icon

Highlight placement for the selected modules to floorplan.

1) Use the popup menu to select Highlight Primitives | Cycle Colors

2) Click on the Device view tab to display the Device view (Figure 13b-4)
Note the placement locations for those two logic modules are dispersed at the bottom of the device.

**TIP:** The **Highlight** commands can be effectively used to identify related logic.

3) In the Netlist view, select **corrMem** module under the **autoCORRi**

4) Select the popup menu **Highlight Primitives** command and select a color to highlight the logic

Notice the BRAM is now highlighted.

5) In the Netlist view, select the **/receiver/synchroTOPInst/autoCORRi** module

6) Select the popup menu **Select Primitives** command to select all of the parent module the logic (**Figure 13b-5**)
Figure 13b-5: Selecting Module Logic

Notice the Primitive logic appears to be related to the 2 BRAMs

7) Select the **Show Connectivity** popup menu command (Figure 13b-6)
Notice the connectivity for the module is contained in the lower center of the device.

### Clearing Placement Constraints

**Step 2**

Clear all placement constraints. It is easier to manipulate Pblocks without placement constraints assigned. Clear all placement constraints using the *Tools | Clear Placement Constraints* command.

The Clear Placement Constraints dialog enables selective filtering of the placement constraints to remove. The I/O and clock related resources are separated from the fabric logic, since they usually don’t change. This allows quick removal of the placement constraints imported from ISE. Logic types filters are also provided to selectively remove placement constraints by type.

Manually placed logic is assigned as fixed. Any selected logic can be “fixed” with the *Fix Instances* popup menu command. By default, all fixed logic receives LOC constraints in the UCF files passed to ISE. Unfixed logic does not.

1) Click the **Unselect All** Toolbar icon

2) Click the **Hide All Timing Paths** Toolbar icon

3) Click the **Hide Core Nets** Toolbar icon

4) Select the **Unhighlight All** Toolbar icon

5) Select **Tools | Clear Placement Constraints**

6) Ensure that the **Instance placement** box is on (Figure 13b-7)
Since the other logic in the autoCORRi module appeared to be related when we highlighted it, we’ll create Pblock containing the entire autoCORRi module. This will ensure that the implementation tools kept the logic tightly grouped rather than spread out over the device.

1) Select the **Collapse All** button in the Netlist view

2) Expand and select the following module: 
   `/receiver/synchroTOPInst/autoCORRi`

3) Use the popup menu to select the **Draw Pblock** command
4) Draw a rectangle in the lower middle area of the device as shown below (Figure 13b-8)

Note that we created it in the middle of the device as that is where the BRAMs and MULT sites are close together.

**TIP:** The rectangle will snap to the outside tile boundary, so start and end your Pblock rectangle just inside of the red BRAM sites, in this example.

5) Click **OK** to accept the default name `pblock_autoCORRi` in the New Pblock dialog (Figure 13b-8)

![Figure 13b-8: Drawing Pblock](image)

6) View the Pblock Properties to examine the Resource Utilization Estimates and reshape the Pblock, if need be (Figure 13b-9)
Scroll down to the Pblock Statistics

Notice that the two BRAMs in this Pblock are 36 bits wide (you can tell by the name). Therefore, six BRAM/MULT sites will be required in this Pblock. Remember the Virtex-II 18+ bit rule mentioned in the Design Challenges section of Lab 1.

Adjust the size of the Pblock to mimic the size shown above. The SLICE logic utilization should be ~67%.

**Examining Multiple Runs for Logic to Floorplan**

In order to identify other potential timing bottlenecks it is sometimes helpful to examine multiple ISE runs. This can help identify other logic that has potential to fail. Analyzing Trce results from each ExploreAhead run result can help uncover other target modules for floorplanning.

**NOTE:** In order to minimize the size of the Tutorial data, ISE data has been removed from most of the run directories. Only the command logs, timing results and status files remain.
Import an analyze the Trce timing results for run_2.

1) Click the **ExploreAhead Runs** view tab and select the run_2 line using in the ExploreAhead Results view

2) Use the popup menu to select the **Import Run** command

3) Click the **Import Placement** button off and click OK (Figure 13b-10)

![Figure 13b-10: Import Timing Results](image)

Note: We’ve elected not to import the placement Locs in the above dialog. If you inadvertently imported the placement, please use **Clear Placement Constraints** to remove them.

4) Click the **Maximize/restore** icon in the new Timing Results view

5) Note the modules that are failing timing are all in 
/receiver/RChainTopInst/bitREV_EQUAL/…

Create a Pblock for the other failing logic:
/receiver/RChainTopInst/bitREV_EQUAL,

1) Use the **Shift** key to select all of the failing paths that have a **From** location of /receiver/RChainTOPInst/bitREV_EQUAL

2) Click the **Maximize/restore** icon in the Timing Results view to restore the view

3) Use the popup menu to select **Schematic**

4) Click the **Collapse All** Netlist view Toolbar button to collapse the netlist tree
5) In the Schematic view, use the popup menu to select the Select Primitive Parents command (Figure 13b-11)

![Figure 13b-11: Identifying Critical Modules](image)

Notice that the parent modules that contained the path primitive logic objects are now selected.

6) Use the popup menu again to select Draw Pblock

7) Draw a rectangle somewhere in the center of the device

8) Enter the Pblock Name `pblock_bitrev` and click OK

9) View the Pblock Property Statistics and notice that 4 MULTs and 1 BRAM are required, but there is fairly low SLICE utilization for the Pblock

10) In the Netlist view, select the following parent module:
    `/receiver/RChainTopInst/bitREV_EQUAL`

11) Click and drag it into the newly created Pblock

12) Click OK to unassign the lower level logic modules

Notice that the selected parent module is now assigned to the Pblock.
Shape the Pblock using Pblock resource utilization statistics

1) Select the `pblock_bitREV_EQUAL`

2) View the Pblock Property Statistics to adjust the Pblock shape to look exactly as it does below (Figure 13b-12)

![Figure 13b-12: Create Pblock for bitREV_EQUAL Module](image)

Note the Pblock edges are aligned with the BRAM/MULT site location edges to maximize RAM/MULT sites available for Pblocks without overlap (Figure 13b-13).
Assigning Placement Constraints Manually  Step 5

Place a LOC constraint manually for one of the BRAMs in the `bitREV_EQUAL` module. Use the Find command to locate the BRAM and drag it into the Device view.

1) In the Netlist view, click the **Collapse** icon

2) Select the **Edit** | **Find** command

3) Ensure the **Find type** is set to **Instances** and **Criteria** set to **Type**

4) Select **Block RAM** in the third field

5) Click the **More** button to add another search criteria

6) Set the first field to **AND**, the second field to **Parent Pblock**, the third field to **is** and use the browser to select the **pblock_bitrev** (Figure 13b-14)
Figure 13b-14: Searching for Specific Logic Objects

7) Click **OK**

Note the Find Results view displays the single Block RAM in the pblock_bitrev (Figure 13b-15).
8) Select the **Create Site Constraint Mode** Toolbar button.

9) Select the BRAM entry in the Finds Results view and drag it into the lower left site on the pblock_bitrev (Figure 13b-16)

Note the context sensitive cursor will only allow placement onto a legal BRAM site.
Figure 13b-16: Manually Assign Placement Constraints

10) Select **Assign Instance Mode** Toolbar button

11) Select **Edit | Undo** to remove the placement constraint

**NOTE:** Most logic objects in the design including BUFGs, DCMs, Ports, PPCs, MGTs, etc. can be assigned to specific sites using the methods described in this step. PlanAhead has placement locations for all object types. Graphic locations are displayed on the I/O pads for imported logic objects that ISE placed on I/O pads.

### Viewing Implementation Results for Floorplan

#### Step 6

Open *fp1_16ns* Floorplan and view implementation results.

1) Select **File | Close Floorplan** and click **Yes** to save and **OK** to confirm close

2) Select **File | Reopen Floorplan | fp1_16ns**

3) Select the **ExploreAhead Results** view tab

Note the ExploreAhead implementation results (Figure 13b-17)
Figure 13b-17: Viewing Floorplanned Implementation Results

Note the results improve, and two do meet timing. Notice the run times decreased dramatically.

4) Select File | Exit and

5) If prompted, click Yes to save all changes and OK to confirm close

Conclusion

In this lab, you examined multiple implementation results to visualize and identify the critical logic in the design. You then selectively assigned that logic to Pblocks and shaped them using resource utilization statistics.

You then looked at the implementation results to see the effect of floorplanning the critical logic.
Floorplan Tuning Lab

Introduction

This lab teaches you to analyze implementation results and use that information to improve the floorplan toward meeting or retaining performance targets. Timing results can be imported to identify failing paths and to determine which logic should be further constrained to increase performance. Selective placement constraints can be assigned to lock logic placement for subsequent implementation attempts. Occasionally, creating additional Pblocks can provide consistency by alleviating congestion within critical modules.

The revised Floorplan can quickly be re-implemented using the same ExploreAhead Run configuration and strategies that were previously successful.

Objectives

After completing this lab, you will be able to:
- Analyze implementation results to adjust the floorplan
- Lock logic from a successful run
- Create additional Pblocks to alleviate congestion
- Examine the successful implementation results

Procedure

This lab comprises five primary steps: you will analyze the previous implementation results, improve the floorplan based on those results, use the connectivity display functions of PlanAhead to adjust Pblock placement; lock the critical logic placement by selectively clearing the remaining Locs; create additional Pblocks to add consistency and finally view the successful implementation results for the revised Floorplan. Below each general instruction for a given procedure, you will find accompanying step-by-step directions and illustrated figures that provide more detail for performing the general instruction. If you feel confident about a specific instruction, feel free to skip the step-by-step directions and move on to the next general instruction in the procedure.

Analyzing ISE Results to Adjust Floorplan Step 1

Launch the PlanAhead software and open the Project_2
<Install_Dir>\PlanAhead_Tutorial\labs\projects\Project_2.

1) On Windows, select the Xilinx PlanAhead 10.1 Desktop icon or Start | Programs | Xilinx | 10.1 | PlanAhead | PlanAhead
On Linux or UNIX, change directory in the 
<InstallDir>/PlanAhead_Tutorial/labs directory and enter the planAhead

2) Select File | Open Project

3) Browse to the <Install_Dir>|PlanAhead_Tutorial|labs|projects|Project_2 directory

4) Select Project_2.ppr and click Open

Ensure the Floorplan – fp1_16ns Floorplan tab is available and selected at the top of the PlanAhead view. If not, select File | Reopen Floorplan | fp1_16ns

Often times, several floorplan iterations are required to identify and constrain all of the critical logic. For the sake of this exercise, we’ll ignore the runs that are meeting timing already and focus on improving the worse run results.

Import the timing results from run_3 to identify additional failing logic.

1) Click the ExploreAhead Results view tab and select the run_3 line

2) Use the popup menu to select Import Run and click OK

3) Click the Maximize/restore icon in the new Timing Results view

Note there are paths failing inside of the receiver/RChainTopInst/RCCInst module.

4) Select the Slack column to sort by slack

5) Select the From column twice to sort the list

6) Use the Shift key to select all paths that start in the following module: receiver/RChainTopInst/RCCInst/… (Figure 15b-1)
7) Click the **Maximize/restore** icon in the Timing Results view to restore the view.

8) Select the **Schematic** popup menu command.

   Notice the logic modules involved is all inside the `receiver/RchainTopInst/RCCInst` module.

   **Draw Pblock for RCCInst module.**

   1) In the Schematic view, select the **Select Primitive Parents** popup menu command.

   2) Use the **Highlight Primitives** with popup menu command to highlight the logic placement with any chosen color.

   3) Select the **Device** view tab to view the highlighted placement locations.

   4) Select the **Unhighlight All** Toolbar icon.
5) Click the Statistics tab in the Instance Properties view

6) Scroll down to see that the module includes 2 RAMs and 1 MULT

7) In the Netlist view, re-select the following module: receiver/RChainTopInst/RCCInst

8) Select the Draw Pblock popup menu command

9) Draw a rectangle for the RCCInst module as shown below (Figure 15b-2)

![Figure 15b-2: Draw Pblock for RCCInst Module](image)

10) Click Yes to Also clear location constraints that fall outside of the pblock?

11) Use the Set Pblock Size popup menu command to adjust shape, if need be

**Using Connectivity Display to Identify Logic to Floorplan**  
Step 2

PlanAhead has extensive logic expansion, selection and highlighting capabilities. You can use these capabilities to validate that modules are suitable to floorplan. Logic modules that connect to logic all over the device may not be, while tightly grouped self contained modules are.

Pblocks with a high number of interface signals may benefit from having the interfacing logic placed next to or merged into the Pblock.
Routing congestion and timing inconsistency can be alleviated by floorplanning additional logic outside of the critical logic areas preventing logic from migrating into these critical areas.

Engineering knowledge of the design can often times identify such modules. In this case there is a large viterbi module that is fairly self contained. You can prevent the logic from migrating into the critical areas by creating a Pblock for it.

Use the Show Connectivity command to view the interface connectivity in and out of the viterbi module.

1) If I/O Nets and Bundles Nets are displayed, click the Show/Hide I/O Nets and Show/Hide Bundle Nets Toolbar buttons to turn them off.

2) Zoom Fit the Device view.

3) In the Netlist view, expand the receiver/RChainTopInst.

4) Select the CONVDECOD(viterbi) module and use the popup menu to select the Show Connectivity command (Figure 15b-3).

Figure 15b-3: Using the Show Connectivity command
Note that only the interface nets that connect the \textit{viterbi} module to the rest of the design are now displayed. There are only a few, making this a good candidate for floorplanning.

Notice also that the logic is heavily interspersed with our critical logic. We probably don't want that.

5) Select the \textbf{Unselect All} Toolbar icon

6) Select the \textbf{Hide Core Nets} Toolbar icon

7) In the Netlist view, select the \textbf{Collapse All} icon

Use the \textit{Show Connectivity} command to view the connectivity inside of the \textit{viterbi} module.

1) In the Netlist view, expand and select the \texttt{receiver/RChainTopInst/CONVDECOD} module

2) Select the \textbf{Select Primitives} popup menu command

3) Select the \textbf{Show Connectivity} popup menu command (Figure 15b-4)
Figure 15b-6: Displaying Routing Congestion

Notice all of the Connectivity through our critical logic.

Notice also that the module is quite large, but it relatively self contained as we saw displayed in the last step.

4) Use the Show Connectivity popup menu command or press the Ctrl T key again

Notice that all of the logic objects that the highlighted nets connected to are now selected.

5) Use the Show Connectivity popup menu command or press the Ctrl T key again (Figure 15b-7)
Figure 15b-7: Expanding and Selecting a Cone of Logic

Note the large module is not connected to logic all over the device.

**NOTE:** The Show Connectivity command can be used to select a cone of logic expanded from any selected design object or group or objects. Sequentially running the command will expand and select the logic cone.

6) Select the **Unselect All** Toolbar button

7) Select the **Hide Core Nets** Toolbar button
NOTE: The Show Connectivity command will leave previously displayed core nets visible in grey. The Hide Core Nets command must be run to turn them off in the display.

**Locking Critical Logic**

Step 3

Module performance can sometimes be inconsistent and only meet performance targets sporadically. To help ensure module performance, placement constraints from previous successful implementation runs can be used to lock logic in place.

**NOTE:** Placement constraints are applied to individual logic objects by name. Re-synthesizing the design and importing the new netlist may result in some of those names changing. This will cause mismatched placement constraints to be removed and logic with new names to be unconstrained. Use placement constraints with care.

Select placement Locs from the successful *autoCORRi* module and selectively remove all Locs except the ones associated with this critical logic.

1) In the Physical Hierarchy view, select the *pblock_autoCORRi*

2) Use the popup menu to select the **Select Primitives** command *(Figure 15b-8)*
Figure 15b-8: Selecting Critical Logic Primitives

3) Select the **Tools | Clear Placement Constraints** command

4) Click **Next** to clear Instance placement only

5) Select the **Unplace all except for XXX selected instances** option and click **Next** (Figure 15b-9)
Figure 15b-9: Clearing all Non-critical Placement Constraints

6) Continue to click **Next** to select defaults and select **Finish** to clear unfixed Instance placement only

Note that only the Locs for the *autoCORRi* module remain. Exporting the Floorplan for implementation with the placement constraints assigned will force the ISE tools to leave the logic placed as is. (**Figure 15b-10**)
7) Select the **Fix instances** popup menu command

![NOTE: When logic is Fixed, it is considered user assigned and is exported to ISE by default.]

![NOTE: Placement constraints only lock the placement of the logic. Routing is not locked and differences can cause discrepancies in performance results.]

**Figure 15b-10: Locking Critical Logic**
Creating Additional Pblocks to Alleviate Congestion  Step 4

Creating additional Pblocks can help make design performance more repeatable by constraining more of the logic. It can also keep logic from migrating into critical logic areas. It’s also very useful if taking advantage of incremental design techniques. However, creating too many Pblocks can also have a negative effect on performance. It’s sometimes a fine line.

Create additional Pblocks for other logic. Often, the I/O interface logic is the fastest and most performance-critical in the design. Constraining the interface logic into Pblocks located next to the I/O pads can improve performance dramatically. Remember from our top-level floorplan that the busMuxWrap module uses 8 BRAMs and connects to all of the I/Os at the top of the device. Let’s isolate and constrain those BRAMs to prevent them from migrating into critical areas.

1) Use the **Collapse All** Netlist view and expand the `receiver` module

2) Select the following module: `/receiver/busMuxWrapInst`

3) Click the **Draw Pblock** Toolbar button and draw a rectangle across the top of the device

4) Accept the default name in the New Pblock dialog and click **OK**

5) If needed, toggle the **Show/Hide I/O nets** and **Show/Hide Bundle Nets** Toolbar buttons **on**

   Note the green I/O connections and colored Bundle Nets.

6) Size the Pblock exactly as shown below ([Figure15b-11]).
Create separate Pblocks all at once for each of the following modules using the default naming convention: receiver/uartInst, receiver/RChainTopInst/CONVDECOD, channel.

Separate Pblocks can be created simultaneously. Use your own design knowledge about logic functions or scan the timing results to identify a list of modules to begin floorplanning and create them all at once.

1) In the Netlist view, expand the receiver/RChainTopInst module.

2) Hold down the Ctrl key and select the following three modules (Figure 15b-12):
   o channel
   o receiver/RChainTopInst/CONVDECOD
   o receiver/uartInst
Figure 15b-12: Selecting Multiple Modules for Pblocks

3) Select **Tools** | **New Pblocks**

4) Click **Next** to accept the three selected instances

You can choose a convention to be used to name each Pblock (**Figure 15b-13**).
5) Click **Next**

Review the Pblocks that will be created.

6) Click **Finish**

The newly created Pblocks will be listed in the Physical Hierarchy view (Figure 15b-14).

![Figure 15b-14: Newly Created Pblocks](image)

**Figure 15b-14. Newly Created Pblocks**

- Draw Pblock rectangles for each of the new Pblocks.
1) Select the *pblock_CONVDECOD* in the Physical Hierarchy view and click the Set Pblock Size Toolbar button. (Make sure you select the Set Pblock size icon and NOT the Draw Pblock icon.)

2) Draw a rectangle in the Device view.

3) Shape the *pblock_CONVDECOD* as shown below (Figure 15b-15).

![Figure 15b-15: Drawing Pblock Rectangles](image)

Note the Pblock icons in the Physical hierarchy view turn three dimensional as rectangles are added.

4) Repeat steps 1 through 3 for the *pblock_uartInst* on the right as shown below (Figure 15b-16).
5) Repeat steps 1 through 3 for the `pblock_channel` on the right as shown below (Figure 15b-17).
Make the `pblock_uartInst` Pblock into a child of the `pblock_busMuxWrapInst` Pblock.

Child and parent (nested) Pblocks can be used to hierarchically constrain logic. This can be used to constrain lower-level modules or to isolate logic modules for utilization purposes. Sometimes rectangles are too restrictive when floorplanning highly utilized designs.

1) Select the `pblock_busMuxWrapInst` Pblock

Notice the heavy Bundle net between the `pblock_busMuxWrapInst` and `pblock_uartInst` Pblocks.

2) View the **Physical Resource Statistics** in the Pblock Properties view

Notice the relatively low SLICE utilization (Figure 15b-18). The rectangle for this Pblock is large because the Pblock requires eight RAM sites. There are only ~29% of the Slices being required within this Pblock rectangle.

![Figure 15b-18: View Low Slice Utilization in pblock_busMuxWrapInst](image)

3) Select the `pblock_uartInst` Pblock and look at the utilization statistics

4) Confirm that the `pblock_uartInst` does not require any RAM or MULT sites

Since it is heavily connected to the `pblock_busMuxWrapInst`, merging them into the same area is likely to improve the design performance and density.

5) Drag the `pblock_uartInst` Pblock completely inside of the `pblock_busMuxWrapInst` Pblock (Figure 15b-19)
6) In the Physical Hierarchy view, expand and select the `pblock_busMuxWrapInst` Pblock

Notice the parent and child relationship.

The Pblock Properties view now shows the combined resource estimates for both Pblocks (Figure 15b-19)

![Figure 15b-19: Nested Pblocks](image)

Merge the `pblock_uartInst` and `pblock_busMuxWrapInst` Pblocks.

You could leave the Pblocks nested to constrain the `pblock_uartInst` logic to a solid area within the `pblock_busMuxWrapInst` rectangle. In this lab, you merge the Pblocks to allow all of the logic to float inside the entire rectangle.

1) In the Physical Hierarchy view, select the `pblock_uartInst` Pblock
2) Use the popup menu to select Delete
3) Click OK in the confirm dialog

The `pblock_uartInst` Pblock disappears from the Physical Hierarchy view (Figure 15b-20).

4) Select the `pblock_busMuxWrapInst` Pblock
Notice that both the `receiver/busMuxWrapInst` and `receiver/uartInst` instances are selected in the Netlist view (Figure 15b-20).

5) Examine the Pblock Properties and notice the combined utilization percentage (Figure 15b-20)

![Figure 15b-20: Examine the Merged Pblocks](image)

**Examining Results for Improved Floorplan**

Open the `fp2_16ns` Floorplan and notice consistency in the implementation results after several attempts.

1) Select File | Close Floorplan and Yes to save and click OK to confirm close

2) Select File | Reopen Floorplan | fp2_16ns

3) Click the ExploreAhead Results view tab

Note that the EA #6 strategy from `run_2` seems to work the best on this design (Figure 15b-21).
Figure 15b-21: Examine Consistent Implementation Runs

Notice the run time reduction from 19 1/2 minutes in the best original run to 6 minutes as more Pblocks were added.

Close the PlanAhead software.

1) Select **File | Exit** and select **Yes** to save all changes and click **OK** to confirm close

**Conclusion**

In this lab, you used an implementation result as a basis for improving the Floorplan.

You improved your chances of meeting performance targets consistently by isolating the minimum amount of logic needed to be constrained. You viewed the connectivity of the design to see if Pblock adjustment is needed.

You locked critical logic by selectively clearing the non-critical placement constraints in the design.

You used some of the other PlanAhead Pblock creation methods to create additional Pblocks for various logic modules. You shaped and sized Pblocks based on resource utilization statistics and you merged heavily connected Pblocks.

You viewed consistently successful results from the improved Floorplan.