Incremental Design Flow User Guide

April 2013
## Type Conventions Used in This Document

<table>
<thead>
<tr>
<th>Convention</th>
<th>Meaning or Use</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Bold</strong></td>
<td>Items in the user interface that you select or click. Text that you type into the user interface.</td>
</tr>
<tr>
<td><code>&lt;Italic&gt;</code></td>
<td>Variables in commands, code syntax, and path names.</td>
</tr>
<tr>
<td><strong>Ctrl+L</strong></td>
<td>Press the two keys at the same time.</td>
</tr>
<tr>
<td><strong>Courier</strong></td>
<td>Code examples. Messages, reports, and prompts from the software.</td>
</tr>
<tr>
<td>...</td>
<td>Omitted material in a line of code.</td>
</tr>
<tr>
<td>.</td>
<td>Omitted lines in code and report examples.</td>
</tr>
<tr>
<td>[ ]</td>
<td>Optional items in syntax descriptions. In bus specifications, the brackets are required.</td>
</tr>
<tr>
<td>( )</td>
<td>Grouped items in syntax descriptions.</td>
</tr>
<tr>
<td>{ }</td>
<td>Repeatable items in syntax descriptions.</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Contents

Using Incremental Design Flow  1
The Incremental Design Data Flow  2
About This Guide  4

Running the Incremental Design Flow  5
Creating Partitions in Synplify Pro  6
Running Incremental Design in Diamond  8
Backing Up an Incremental Design Run  11
Restoring Golden Backup Files  11
Restoring Previous Incremental Design Results  12
Archiving Incremental Design Results  12
Viewing Reports for Incremental Design  13
Incremental Design Flow Tcl Commands  13
  prj_incr enable  13
  prj_incr restore  14
  prj_incr backup_golden  14
Incremental Design Flow Database Extended Tcl Commands  14
  icf_data reload  14
  icf_data save  14
Incremental Design Flow Partition Extended Tcl Commands  15
  icf_part set_level  15
  icf_part set_effort  15
  icf_part set_bbox  15
  icf_part set_anchor  15
  icf_part set_color  15

Using Partition Manager  17
Running the Partition Manager  21
Creating a New Partition with the Partition Manager  21
CONTENTS

Setting Preservation Data  21
Setting Reimplementation Effort  22
Setting Preservation Data or Reimplementation Effort for Some or All
  Partitions at the Same Time  22
Specifying Partition Anchor and Bounding Box  22
Changing Partition Colors  23
Cross-Probing Partitions from Partition Manager  23
Removing a Partition Using Partition Manager  24

Floorplan View and Physical View Partition Support  25
Displaying Partitions in Floorplan View  25
Editing Partitions in Floorplan View  26
Displaying Partitions in Physical View  27

Recommended Strategies for Incremental Design  29
Recommended Incremental Design Flow: Scenario 1  30
Recommended Incremental Design Flow: Scenario 2  30
Recommended Incremental Design Flow: Scenario 3  31
General Incremental Design Flow Limitations  32
General Synthesis Recommendations  33
  Avoiding Common Design Flow Issues  34

Troubleshooting Error and Warning Messages  39
Troubleshooting Missing Files and Folders  39
  Known Issues  40
Map Warnings & Errors  44
Placer Warnings & Errors  45
Router Warnings & Errors  47
Supplying Archives for Technical Support  47
Chapter 1

Using Incremental Design Flow

The Incremental Design flow feature is supported for LatticeECP2M and LatticeECP3 devices. Incremental Design is a design methodology that preserves certain process results and performance on portions of a design while reducing re-processing time by focusing on other parts of the design during multiple design processing iterations. It accomplishes this by using design partitions you define in synthesis. Partitions allow you to lock down and preserve timing stability by preventing specified modules from being changed in the next design iteration.

In addition to partitions, the Incremental Design flow uses a previous design file to guide the rest of your design implementation toward smaller, more incremental changes. Using this strategy, you no longer have to re-implement your design from the beginning which forces changes that effect the technology mapping and layout of your entire design. Using a reference design, your previous I/O placement also remains intact.

You might consider using the Incremental Design flow for the following reasons:

▸ It reduces re-processing runtime during multiple iterations.
▸ It preserves previous timing results on portions of a design during multiple iterations.
▸ It minimizes runtime variations for design changes during multiple iterations.
▸ It helps achieve timing closure by partition planning and coding improvement.

The Incremental Design flow in Diamond requires you to first specify design partitions during synthesis in the top-level Synplify FPGA Design Constraints (.fdc) file. Design partitions are defined by the “Compile Points” feature in Synplify Pro, which enables you to create constraints in the .fdc file that tell the synthesis compiler to preserve and timestamp modules. So, partitions
allow you to “lock” these portions of your design to preserve the logic through synthesis and mapping if there is no source code change to a partition. Place and Route (PAR) later will determine how components fall into the design layout with the aid of a partition planning utility within the PAR tool that is specific to this flow in Diamond.

Once you associate your .fdc file with your Diamond project and turn on the incremental flow mode, the PAR output of this mode is a reference physical design (NCD) file. Diamond incremental flow will use this placed-and-routed reference NCD to help guide your design through the next processing iteration and take advantage of improved runtime and timing stability.

Through appropriate partition planning and coding improvement, the incremental design methodology will help achieve timing closure through multiple iterations. With today’s ever-increasing FPGA densities to meet more complex applications, it can be challenging to meet your design goal. These designs are usually comprised of multiple function blocks and each block may have significant amount of logic or it may perform a specific task that requires being addressed separately using “divide and conquer” approach.

Incremental Design allows you to isolate and lock down satisfactory portions of such a design and focus on others. At a minimum, reducing your runtime on larger designs using this methodology will give you more time to work on portions of your designs that need more work.

The Incremental Design Data Flow

The section describes each design entry and implementation phase of the Incremental Design flow in terms of how data is processed in the Diamond environment through its various core tools. Figure 1, “Incremental Design Data Flow” below illustrates the flow to provide you with a good basic understanding of how the flow works within the system.

The following list provides a high-level outline on how the Diamond Incremental Design flow works and how data is processed through each step:

1. You begin by defining your design partitions using “compile points” in top-level .fdc file using Synplify Pro. The type of compile point that is required for Incremental Design flow in Synplify Pro is {locked, partition}.

2. Synplify Pro runs a partition-based synthesis using compile point constraints. The output EDIF file has “partition” properties labeled on modules instances. These include timestamps from Synplify Pro.

3. The Design Translation process (EDIF2NGD and NGDBUILD) recognizes and honors “partition” properties and passes them onto their output NGO and NGD files, respectively. Other than passing the properties, these two processes are exactly the same as in the normal flow.

4. The Map process can optionally use the previous PAR NCD result as the reference to guide the process for each partition, according to user settings set through the Partition Manager. Otherwise, new mapped data
for a partition (or the whole design, depending on user settings) will be generated.

5. The PAR process can optionally use the previous PAR NCD result as the reference to guide the process for each partition, according to user settings set through the Partition Manager. Otherwise, new PAR data for a partition (or the whole design, depending on user settings) will be generated.

6. View process reports in the Diamond Report window. If the PAR process finished successfully (err code 0, regardless of timing score), its result is ready to be used as the reference for the next iteration.
About This Guide

Running the Incremental Design Flow  This section focuses on running the actual Incremental Design flow in Synplify-Pro and Lattice Diamond.

Using Partition Manager  This section describes the Diamond Partition Manager tool in detail and its role in the flow.

Recommended Strategies for Incremental Design  This section provides advice to make the best strategic design decisions to help you avoid issues during the flow and get the best results.

Troubleshooting Error and Warning Messages  This section lists any common errors or warnings that you might encounter in the flow and provides you solutions for working around them.
Running the Incremental Design Flow

This section provides all description and procedural information you need to know to run a complete Incremental Design flow through Diamond software. A list of Tcl commands is provided in “Incremental Design Flow Tcl Commands” on page 13.

Along with the steps provided within this chapter, there are two additional requirements for running the Incremental Design flow in Diamond. These requirements are as follows:

- Create a Synopsys FPGA Design Constraints (.fdc) file with compile points in Synplify Pro so that the Incremental Design flow can function properly within Diamond.
- All work in your Diamond project directory should be kept within one Diamond project implementation so that your reference files do not become disassociated with your target design.

Take the following steps to implement an incremental design in Diamond:

1. Create a new Diamond project and implementation; add HDL files as necessary.
2. Turn on incremental flow by choosing Design > Enable Incremental Design Flow.
3. Launch the Lattice OEM version of Synplify Pro from within Diamond. The Synplify Pro project should be created automatically for you. In Synplify Pro, create a top-level Synopsys FPGA Design Constraints (.fdc) file. This file is necessary for defining design partitions for synthesis. See “Creating Partitions in Synplify Pro” on page 6.
4. In Diamond, add your .fdc file to incremental flow enabled implementation. See “Running Incremental Design in Diamond” on page 8.
5. Run synthesis, Map, and PAR in Diamond. You can run this from the Tcl Console if you prefer to do so. See “Running Incremental Design in Diamond” on page 8.

6. Analyze static timing using the Timing Analysis view (Timing Preference file) or the Place & Route Trace process (TRACE Report file) to determine how close you are to meeting your design’s timing constraints.

The TRACE report and Timing Analysis view analysis tools should provide indicators that tell you about resource utilization and alert you to any possible timing issues. You should also review Map and PAR reports. Review these reports to ensure that the data results make sense in regard to what you would expect to see in your design.

If you are not already aware of the module or modules that you need to re-implement without locking down its logic using synthesis directives for recompile and re-synthesis, the reports should make this clearer. See “Viewing Reports for Incremental Design” on page 13.

7. If PAR finishes successfully, the PAR NCD will be used as a reference automatically during the next processing iteration. If necessary, you can modify your code; change strategies, preferences or partition management settings; and run the next processing iteration.

To perform these steps follow the detailed procedures described in “Creating Partitions in Synplify Pro” on page 6 and “Running Incremental Design in Diamond” on page 8, respectively.

Creating Partitions in Synplify Pro

You create design partitions for your source files by creating a top-level Synopsys FPGA Design Constraints (.fdc) file in Synplify Pro. You need to generate an .fdc file and later associate it with your Diamond project for the Incremental Design flow to work properly and generate data later in the flow.

To create an .fdc file using the standalone Synplify Pro tool:

1. Open your incremental design flow project in Diamond. This procedure assumes that you have already created a project and an implementation, turned on the incremental flow for the implementation for your design (see “Running Incremental Design in Diamond” on page 8), and imported your source files into your Diamond project.

2. In Diamond, choose Tools > Synplify Pro for Lattice or click the toolbar button. Synplify Pro opens.

Notice that Synplify Pro automatically imports in all of your source HDL files and uses your project name in Diamond but appends “_syn” to the project name. In addition, it sets the same device family in both implementation and synthesis projects. It is important to open Synplify Pro from Diamond instead of launching from the Start menu or from some other installation so that the synthesis project inherits the project characteristics from Diamond.
Now you will see your source files in a folder in your Project view. For example, if they are Verilog files the folder will be named accordingly.

3. Choose Run > Compile Only or F7. This enables the Synopsys FPGA Design Constraints (.fdc) file to be initialized, which allows you to define compile points.

4. Click the New Constraint File toolbar icon.

5. In the New Constraint File dialog box, click Constraint File (Scope) and, in the Create a New Scope File dialog, ensure that the default options are checked or selected and click OK:
   - Initialize Constraints tab: Clocks, I/O Delays
   - Select File Type tab: Top Level

   An interactive SCOPE spreadsheet referred to as the Compile Points panel opens. This panel allows you to specify the compile points of your design, and to enable or disable them. Compile points specify design partitions. Note that this panel is only available when defining a top-level constraint file, and only if the device technology supports the compile points.

6. Click on the Maximize button on the top right of the window’s title bar to view the entire SCOPE spreadsheet and choose the Compile Points tab. You now will see a blank spreadsheet with Enabled, Module, Type, and Comment columns.

7. Double click on the top row’s Module column cell and choose a module that you want it to be a partition from the drop-down menu that appears to the right. In addition, in the Type cell, choose locked, partition for each chosen module and ensure that each module is enabled with the check box in the far left column. Repeat these steps for each module that you want it to be a partition in your design.

8. To save your FPGA Design Constraints (.fdc) file in the default synthesis project location, choose File > Save and click Yes in the prompt dialog.
Running Incremental Design in Diamond

After you have created a FPGA Design Constraints (.fdc) file using Synplify Pro as described in “Creating Partitions in Synplify Pro” on page 6, you can begin the procedure of implementing your incremental design in Diamond. This topic describes how to associate that .fdc file with your project, turn on the incremental run modes, and implement your design.

To run the Incremental Design flow in Diamond:

1. Open your incremental design project in Diamond. To enable the incremental design flow for the active implementation, choose Design > Enable Incremental Design Flow, or enter the following command in the Tcl Console window at the prompt:

   prj_incr set -enable
2. Right-click the Synthesis Constraint Files folder in the File List view, and choose Add.

3. Choose Existing File, and in the Add Existing File dialog box, browse to the .fdc file you generated in the procedure “Creating Partitions in Synplify Pro” on page 6, and click Add to target it for use during synthesis. Right-click the added .fdc file, and choose Set As Active.

4. In the Process view, double click the Place & Route Design process to run the incremental design flow through synthesis, mapping, and place & route.

Be aware that after this point the Map program generates partition database data that contains partition definitions, packing, and optimization strategies that Map uses based on the defined partition.

If this first processing iteration completes successfully, the software automatically creates a backup copy of the placed-and-routed data and stores them in a subfolder called “inc1” inside the Diamond implementation folder. In future design iterations. If PAR completes successfully, these files in “inc1” will be overwritten with the results of the new run and the previous results files in “inc1” are copied to “inc2”, so you have a backup reference design available to you.

Note
These folders (inc1, inc2, etc.) and files in them are not directly accessible in the Diamond GUI. You must navigate to them using Windows. These folders, and the files in them, are managed by Diamond. You should not manually manipulate them.

For more information on restoring or backing up previous incremental design runs, see the “Backing Up an Incremental Design Run” and “Restoring Previous Incremental Design Results” topics in this chapter.

It is highly recommended that you use the File > Archive Project command (archiving all files) to save the state of your project. See “Archiving Incremental Design Results” on page 12.

5. In the Process view, double click the Place & Route Trace process to run post PAR trace and get the timing results on your placed-and-routed NCD file.

6. Use the Report viewer tab in the center panel to open your Map and Place & Route Process Reports; and your Map Trace and Place & Route Trace Analysis Reports.
To ensure that the results of your design run are satisfactory, see “Viewing Reports for Incremental Design” on page 13, which describes what key elements of the reports you should be viewing.

7. After reviewing the result, you can make necessary HDL code changes, adjust preferences and strategies, set partition controls, and then repeat the process from step 4. above. For setting partition controls in order to fine control the next core processing iteration, use Partition Manager. Refer to “Using Partition Manager” on page 17.

When you rerun the design flow to start another core processing iteration, the NCD file in “inc1” will be used as the reference design for both Map and PAR. The flow will copy the reference files from “inc1” to the working directory before Map is run.

**Note**

So that Synplify Pro correctly handles partition information, you must stay in your original Diamond project implementation in which you began your incremental design. If you want to create a new implementation and copy source correctly, Synplify will still not reference its partition files. This flow is not ideal for using across multiple project implementations because it relies on a referencing files it uses to guide synthesis and implementation in a relative way.

**Note**

If you perform the Rerun process on a synthesized design, or the Rerun All process, your synthesis result will be erased and regenerated. Any runtime savings that were gained by preserving the result will be lost, because the preserved result will have been erased.
Back up an Incremental Design Run

You can preserve a “golden” set of PAR reference files from a successful Incremental Design run into a “ref” folder in your Diamond project.

To back up a set of design files in the Incremental Design flow:

- In the Run Manager, right-click on the selected Implementation <strategy> and choose Incremental > Backup as Golden Reference.

  or

- In the Tcl Command Console, enter the following command:

  ```
  prj_incr backup_golden -impl <implementation name>
  ```

Performing this task saves a backup of your current files in “inc1” to a newly generated “ref” folder and prevents them from being overwritten on future design runs. This allows you to start your design over again from an ideal starting point that you chose rather than from a possible undesirable point in a later iteration.

Restoring Golden Backup Files

If for some reason you want use the golden backup files as the reference to guide the next core processing iteration, you can restore them.

To restore your golden backup files in the Incremental Design flow:

- In the Run Manager, right-click on the selected Implementation <strategy> and choose Incremental > Restore Golden as Reference.

  or

- In the Tcl Command Console, enter the following command:

  ```
  prj_incr restore -impl <implementation name> golden
  ```

Performing this task copies your “golden” backup files in “ref” folder and overwrites your file set in the “inc1” folder, restoring them as the reference for future design runs.
Restoring Previous Incremental Design Results

When a core processing iteration completes successfully, this iteration's reference files are copied from /inc1 to /inc2, and this iteration's PAR results are copied to /inc1 and will be used as reference for the next iteration. If for some reason you still want to use the previous reference (in /inc2) to guide the next iteration, you can restore it.

**Note**

Whenever you run an iteration of your Incremental Design project, your previous results are overwritten in the “inc1” folder in your project implementation directory and moved to the “inc2” folder. However, the software will automatically use whatever is in “inc1” as reference input into its next iteration, not those files in “inc2”.

To restore the previous reference files for Incremental Design:

1. In the Run Manager, right-click on the selected Implementation <strategy> and choose Incremental > Restore Previous Reference.

   or

2. Enter the following command in the Diamond Tcl Console window at the prompt:

   ```
   prj_incr restore -impl <implementation name> previous
   ```

   This command restores the previous design run as your reference design in “inc1”, overwriting your last results with the file that was moved to the “inc2” folder. You should have matching file sets in both “inc1” and “inc2” folders.

**Note**

For the Incremental Design flow to work, the Map process must be rerun on the new reference design. So, restoring a previous result will force the Process List to reset its run status before the Map process.

Archiving Incremental Design Results

It is recommended that you use the File > Archive Project command to retain the original state of your project. There may be a case in which you might wish to return to your original input source file set or you might want to know what the original file set contained. After many iterations and result files overwrites, it could be difficult to keep track of this.

In addition, you may wish to archive incremental design runs that are close to satisfying your design goals so if future iterations take you further away from your goals you can go back to a point that does not force you to start over.
Viewing Reports for Incremental Design

This section introduces you to the reports and the specific subsections in them that you should view to evaluate your incremental design run. It also will give you tips and clues to better go about verifying what resources were generated and if they meet expectations.

To view reports for your incremental design:
1. Double click on the appropriate report in either the Process Report or Analysis Report folder. This assumes that you have run a design flow and that there are generated report files.
2. View the report in the panel to the right and use the scroll bars to navigate to portions of the report that you wish to view.

To use the Timing Analysis view to analyze your incremental design:
1. After you have run through Place & Route design, choose Tools > Timing Analysis view.
2. In Timing Analysis view, click on a defined timing preference in the preference window in the lower left corner of the view. The Path Table and the Detailed Path Table spreadsheet information is populated with various data on delay and timing related to that preference.
3. In the bottom right of the Timing Analysis view, click on the Reports tab. Notice that as you scroll through the timing report, the elements reported on are highlighted in blue.
4. Right click on some design element in the report and choose either Show in FP View or Show in Physical View from the popup menu.

Incremental Design Flow Tcl Commands

This section provides syntax and examples of the commands you might use during this flow with some explanation of what happens when the commands are run.

prj_incr enable

Get or set the incremental design flow mode.

Usage:
prj_incr set [-enable|-disable] [-impl <implementation name>]
**prj_incr restore**

Set the previous or golden backup as the reference for the next incremental run.

**Usage:**

```
pjr_incr restore [-impl <implementation name>] <previous|golden>
```

**prj_incr backup_golden**

Set the current successful results as the golden reference backup.

**Usage:**

```
pjr_incr backup_golden [-impl <implementation name>]
```

**Incremental Design Flow Database Extended Tcl Commands**

**icf_data reload**

Loads or reloads the settings of the partitions from the data on disk. The changes in memory will be discarded.

**Usage:**

```
icf_data reload
```

**icf_data save**

Saves the partition setting changes to the data on disk.

**Usage:**

```
icf_data save
```
Incremental Design Flow Partition Extended Tcl Commands

icf_part set_level

Sets the preservation level for a partition.

Usage:
icf_part set_level -part <partition name> [-value <SYNTHESIS|MAPPED|PLACED|ROUTED>]

icf_part set_effort

Sets the reimplementation effort for a partition.

Usage:
icf_part set_effort -part <partition name> [-value <GUIDED|UNGUIDED>]

icf_part set_bbox

Sets the bounding box for a partition.

Usage:
icf_part set_bbox -part <partition name> [-width <value>] [-height <value>]

icf_part set_anchor

Sets the anchor for a partition.

Usage:
icf_part set_anchor -part <partition name> [-anchor <RxCy>]

icf_part set_color

Sets the display color of the partition in the graphical user interface. The “xxx” represents a three-digit number ranging from 000 to 255 for the red, green, and blue value.
**Usage:**
```
icf_part set_color -part <partition name> [-color <RxxxGxxxBxxx>]
```

**Incremental Design Flow Tcl Command Examples**

The following are a few examples of Tcl command lines and a description of what each does for Incremental Design.

**Example 1**  The following example shows the Tcl command usage for loading the partition information.
```
icf_data reload
```

**Example 2**  The following example shows the Tcl command usage for saving the partition information.
```
icf_data save
```

**Example 3**  The following example illustrates the Tcl command usage for setting the preservation level for a partition.
```
icf_part set_level -part CGROUP_GRAY0 -value MAPPED
```

**Example 4**  The following example illustrates the Tcl command usage for setting the reimplementation effort for a partition.
```
icf_part set_effort -part CGROUP_GRAY0 -value GUIDED
```

**Example 5**  The following example illustrates the Tcl command usage for setting the bounding box for a partition.
```
icf_part set_bbox -part CGROUP_GRAY0 -width 4 -height 4
```

**Example 6**  The following example illustrates the Tcl command usage for setting the anchor for a partition.
```
icf_part set_anchor -part CGROUP_GRAY0 -anchor R10C10
```

**Example 7**  The following example illustrates the Tcl command usage for setting the color for a partition.
```
icf_part set_color -part CGROUP_GRAY0 -color R0G50B255
```
Using Partition Manager

Partition Manager is the Diamond graphical user interface (GUI) used to perform such tasks such as preservation data control, re-implementation effort control, region assignment, and acts as the central interface between the user and partition database.

Partition Manager is only available if Incremental Flow is enabled for an implementation in the project and can be run when the flow is at the pre-Map, post-Map, or post-PAR stage.

Partition Manager can be used after the post-Map stage to create new partitions, edit partition information or remove existing partitions. New partitions can be created for hierarchical modules. You can edit existing partition information such as partition's preservation data level, reimplementation effort, anchor and bounding box. Existing partitions can be deleted and the flow rerun without the deleted partitions.

When a partition's information has been updated in either Partition Manager or Floorplan View, the Diamond process is reset to the Translate process.

The GUI for the Partition Manager is in the form of a table. Each row corresponds to a partition in the project. The columns for each row are for the partition name, partition source, the preservation data, the reimplementation effort, the partition anchor and bounding box. All attributes except Partition Name and Partition Source may be edited.

Default values are shown in blue. Anchor and bounding box cells are blank if they are not set for the partition. The top partition module in the project is always in the first row. A color selection box allows you to change the color of the partition in as it appears in Diamond Floorplan View.

The Partition Manager view is shown in Figure 5.
An asterisk will appear in the Partition Manager title bar, as shown in Figure 5, if any partition parameters have been modified. Changes can be discarded before they are applied by clicking the Discard All Changes button. Changes must be applied by clicking the Apply button.

A partition can be added by right-clicking a hierarchical module in the Instances pane on the left side and choosing Create Partition from the pop-up menu.

A partition can be removed by highlighting the partition and clicking the Remove Partition button, or by highlighting a partitioned hierarchical module in the Instances pane on the left side and choosing Remove Partition from the pop-up menu.

**Preservation Data**

Preservation Data allows the user to define partition based implementation process data to be preserved during the next core processing iteration. When it is defined, and if there is no change to invalidate the defined preservation data for the partition, the defined data will be copied from the reference files during the next iteration. All the data after the defined data will be regenerated and the reference is ignored.

Levels of preservation data include:

- Synthesis
- Mapped
- Placed
- Routed

For example, if a partition's preservation data is set to "Mapped", during the next core processing iteration, if there is no change (such as source code change) to invalidate this partition's mapped result, the new mapped result for this partition will be copied from the reference files. Placed and routed results for this partition will be regenerated regardless of the reference files. If the reference doesn't exist, the new mapped result will be regenerated as well.
Reimplementation Effort

If the Preservation Data defined above is invalidated (due to the source code change, for example) and cannot be preserved, Reimplementation Effort allows the user to specify during the next core processing iteration whether the core processes (Map, Placer and Router) should use the reference to guide it (Guided) and generate a new result, or ignore the reference (Unguided) and regenerate a new result.

Preservation Effort has two choices
- Guided
- Unguided

Using the same example as above, if the preservation data was set to "Mapped" and the reimplementation effort was set to "Guided", during the next core processing iteration:

1. If there is no change to invalidate the defined "mapped" data for the partition, the mapped result will be copied from the reference. Placed and routed results will be regenerated regardless of the reference.

2. If there are changes (such as source code changes) that invalidates the defined "mapped" data for the partition, the reference will be used to "guide" Map to generate a new mapped data. Placed and routed results will be regenerated regardless of the reference.

Similarly, if the preservation data was set to "mapped" and the preservation effort was set to "Unguided", during the next core processing iteration:

1. If there is no change to invalidate the defined "mapped" data for the partition, the mapped result will be copied from the reference. Placed and routed results will be regenerated regardless of the reference.

2. If there are changes (such as source code changes) that invalidates the defined "mapped" data for the partition, the reference will be ignored. Map will generate new mapped data. Placed and routed results will be regenerated regardless of the reference.

The expected result of the Reimplementation Effort setting, combined with Preservation Data setting, for each partition, is described in Table 1.

<table>
<thead>
<tr>
<th>Preservation Data Setting</th>
<th>Reimplementation Effort Setting</th>
<th>Expected Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>Synthesis</td>
<td>Guided</td>
<td>Map, place, and route results will be regenerated without using the reference.</td>
</tr>
<tr>
<td>Mapped</td>
<td>Guided</td>
<td>If the mapped data is still valid, copy from the reference. Otherwise, use the reference to guide and regenerate mapped data. Placed and routed data will be regenerated regardless of the reference.</td>
</tr>
<tr>
<td>Placed</td>
<td>Guided</td>
<td>If the placement data is still valid, copy from the reference. Otherwise, use the reference to guide and regenerate placement data. Routing data will be regenerated regardless of the reference.</td>
</tr>
</tbody>
</table>
Preservation Data and Reimplementation Effort settings control only Map, Place and Route. The synthesis process is not controlled by these settings. Synthesis is controlled by Synplify Pro.

Changing only preservation data and/or reimplementation effort settings will not reset the flow. To force any changes to take effect without changing the source code, in the Diamond Process view, right-click on Place and Route Design and choose **Rerun All**.

The default preservation data setting is “Placed.” The default reimplementation effort setting is “Guided.”

### Anchor and Bounding Box

The partition's anchor and/or bounding box (BBox) are editable with the Partition Anchor and BBox dialog box. The dialog box specifies, based on the device, acceptable values for row, column, height, and width.

After making the desired changes to the partitions, the changes will be applied to the implementation. If the anchor or bounding box changes were made after a successful PAR, the process will be reset to before PAR.

When the user closes the Partition Manager, if changes were made but not applied or reverted, a dialog appears asking “Do you want to save your in-memory ICF changes into `<file_name>.icf`?” with the options of **Yes**, **No**, and **Cancel**.

When the user closes Diamond Project Navigator, if changes were made but not applied or reverted, a dialog appears listing changes that need to be saved. The user may select the file if he wants changes to be saved to it before closing.

<table>
<thead>
<tr>
<th>Preservation Data Setting</th>
<th>Reimplementation Effort Setting</th>
<th>Expected Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>Routed</td>
<td>Guided</td>
<td>If the routing data is still valid, copy from the reference. Otherwise, use the reference to guide and regenerate routing data.</td>
</tr>
<tr>
<td>Synthesis</td>
<td>Unguided</td>
<td>Map, place, and route results will be regenerated without using the reference.</td>
</tr>
<tr>
<td>Mapped</td>
<td>Unguided</td>
<td>If the mapped data is still valid, copy from the reference. Otherwise, ignore the reference and regenerate map data. Placed and routed data will be regenerated regardless of the reference.</td>
</tr>
<tr>
<td>Placed</td>
<td>Unguided</td>
<td>If the placement data is still valid, copy from the reference. Otherwise, ignore the reference and regenerate placement data. Routing data will be regenerated regardless of the reference.</td>
</tr>
<tr>
<td>Routed</td>
<td>Unguided</td>
<td>If the routing data is still valid, copy from the reference. Otherwise, ignore the reference and regenerate routing data.</td>
</tr>
</tbody>
</table>
Running the Partition Manager

If you have an active implementation with incremental design flow enabled, after synthesis and translation, you can run the Partition Manager.

To run the Partition Manager:
1. Diamond, choose Tools > Partition Manager or click the toolbar button.

Creating a New Partition with the Partition Manager

You can create a new partition to hierarchical modules in an incremental design after the MAP process has been run.

To create a new partition with Partition Manager:
1. Right-click a hierarchical module in the Instances pane on the left side and choose Create Partition from the pop-up menu.
2. In the Partition Anchor and BBox dialog box, you can either accept the default Anchor and Bounding box settings, or you can specify Anchor and Bounding box locations, and then click OK.

See Also “Specifying Partition Anchor and Bounding Box” on page 22.

Setting Preservation Data

You can set the partition data to be preserved and used for the next incremental iteration in Partition Manager. Preservation data includes partition processing results including synthesized data, mapped data, placement data and routed data.

To set Preservation Data in Partition Manager:
1. Double-click the Preservation Data cell of the Partition Name row you wish to set. You can also right-click on the cell to display a menu with the preservation data options. This menu has a “Clear” option to reset the preservation data level to the default value.
2. In the drop-down menu, select from Synthesis, Mapped, Placed, or Routed data you wish to preserve.
3. Click Apply when finished editing. To discard all changes and revert to the last save, click Discard All Changes.
Setting Reimplementation Effort

You can set the partition effort for changed partitions to determine how previous processing results are used during the current incremental iteration.

To set Reimplementation Effort in Partition Manager:
1. Double-click the Reimplementation Effort cell of the Partition Name row you wish to set. You can also right-click on the cell to display a menu with the reimplementation effort options. This menu has a “Clear” option to reset the reimplementation effort level to the default value.
2. In the drop-down menu, select from Guided or Unguided.
3. Click Apply when finished editing. To discard all changes and revert to the last save, click Discard All Changes.

Setting Preservation Data or Reimplementation Effort for Some or All Partitions at the Same Time

You can set the partition effort or Reimplementation Effort for some or all partitions at the same time.

To set some or all partitions to same Preservation Data or Reimplementation Effort in Partition Manager:
1. Select some items, or all items, from the Preservation Data column or Reimplementation column, and right-click.
2. In the pop-up menu, choose one of the following:
   - For Preservation Data, select from Synthesis, Mapped, Placed, or Routed data you wish to preserve.
   - For Reimplementation Effort, select from Guided or Unguided.
3. Click Apply when finished editing. To discard all changes and revert to the last save, click Discard All Changes.

Specifying Partition Anchor and Bounding Box

You can set the partition anchor and bounding box within the device architecture using Partition Manager. As values are changed, the number of LUTs, REGs, and EBRs encompassed by the bounding box will be shown in the dialog.

To set specify partition anchor and/or bounding box:
1. Double-click the Partition Anchor & BBox cell of the Partition Name row you wish to set, or right click the cell and choose Edit Anchor and BBox, to display the Position Anchor and BBox dialog box.
2. If modifying anchor, check the **Anchor** box and specify Row and/or Column.

3. If modifying bounding box, check the **BBox** box and specify Height and/or Width.

4. Click **OK**

When a partition has been edited and applied in Partition Manager, the Diamond process is re-set to the pre-Map process.

**Note**

Anchor and bounding box can also be edited in the Floorplan View. Refer to “Editing Partitions in Floorplan View” on page 26.

---

### Changing Partition Colors

You can change the partition colors that display in Diamond Floorplan View by using the Color selection box in Partition Manager.

**To change the color of a partition in Partition Manager:**

1. Double-click the item you want to change. The Select Color dialog box opens.

2. Select a color by doing one of the following:
   - Click one of the basic colors on the left.
   - Click in the color pallet on the right.
   - Enter values in the boxes at the bottom right. Work in one column or the other as they are two separate methods of coding color.

   The rectangle in the lower-middle changes to show the selected color. You can change the darkness with the vertical slider at the upper-left.

3. Click **OK**.

### Cross-Probing Partitions from Partition Manager

You can cross probe and view partitions in either Diamond Floorplan View or Physical View from Partition Manager. In order to perform cross probing, the partitions must have anchor and bounding box set.

**To cross probe a partition in Floorplan View from Partition Manager:**

1. In Partition Manager, highlight the Function Name, Preservation Data, Reimplementation, Partition Anchor & BBox, or Color box in a partition.

2. Right-click, and in the dropdown menu, choose either **Show In > Floorplan View** or **Show In > Physical View**.
Removing a Partition Using Partition Manager

You can remove a partition using Partition Manager.

To remove using Partition Manager:
1. In Partition Manager, highlight the partition you wish to remove.
2. Click Remove Partition, then click Apply.
   or:
   Right-click a partitioned hierarchical module in the Instances pane on the left side and choose Remove Partition from the pop-up menu.

The partition will be removed when the Translate process is re-run.
Floorplan View and Physical View Partition Support

Both Floorplan View and Physical View can display partition information to support the Incremental Design Flow. Partitions can be edited in Floorplan View, but not in Partition View.

- Floorplan View loads partition information that has not yet been loaded into memory. Data is only loaded after the Translate process. See “Displaying Partitions in Floorplan View” on page 25.
- Physical View can display partition information to support the Diamond Incremental Design Flow. Data is only loaded after the Translate process. See “Displaying Partitions in Physical View” on page 27.

Displaying Partitions in Floorplan View

Partitions are displayed nearly identically to UGROUPs and REGIONs in Floorplan View. If a partition has an anchor and bounding box, then it can be displayed in Floorplan View. However, if a partition does not have either an anchor or bounding box, the partition will not be displayed in Floorplan View. Refer to “Specifying Partition Anchor and Bounding Box” on page 22 for information on how to specify anchor and bounding box.

Partitions will have a solid border with a patterned background, similar to UGROUP bounding boxes. The patterned background is different than that of a UGROUP.

Note

REGIONs have a solid background.
To display partitions in Floorplan View:
1. Choose **Tools > Floorplan View**.
2. By default, partitions are displayed in Floorplan View. If they do not appear on the layout, click the button on the vertical toolbar, or click **View > Preference Placement Display > Display Partitions**.

See Also
- “Editing Partitions in Floorplan View” on page 26
- “Cross-Probing Partitions from Partition Manager” on page 23
- “Displaying Partitions in Physical View” on page 27

**Editing Partitions in Floorplan View**

If an anchor and bounding box have been specified for a partition in Partition Manager, the anchors and bounding boxes can be viewed and edited in Floorplan View.

To edit partitions in Floorplan View:
1. Choose **Tools > Floorplan View**.
2. By default, partitions are displayed in Floorplan View. If they do not appear on the layout, click the button on the vertical toolbar, or click **View > Preference Placement Display > Display Partitions**.
3. Right-click the partition you wish to edit and choose **Edit Partition** to display the Position Anchor and BBox dialog box.
4. If modifying anchor, check the **Anchor** box and specify Row and/or Column.
5. If modifying bounding box, check the **BBox** box and specify Height and/or Width.
6. Click the **Save Partitions** button on the vertical toolbar.

When a partition has been edited and applied in Floorplan View, the Diamond process is re-set to the pre-Map process.

**Note**

Anchor and bounding box can also be edited in the Partition Manager. Refer to “Specifying Partition Anchor and Bounding Box” on page 22.

See Also
- “Displaying Partitions in Floorplan View” on page 25
- “Cross-Probing Partitions from Partition Manager” on page 23
- “Displaying Partitions in Physical View” on page 27
Displaying Partitions in Physical View

Physical View displays partition information with the similar way as UGROUP or REGION. The color used for partitions are different than that of the UGROUP or REGION.

To display partitions in Physical View:
2. By default, partitions are displayed in Physical View. If they do not appear on the layout, click View > Show Layers > Partition.

See Also
- “Displaying Partitions in Floorplan View” on page 25
- “Editing Partitions in Floorplan View” on page 26
- “Cross-Probing Partitions from Partition Manager” on page 23
Chapter 5

Recommended Strategies for Incremental Design

This chapter provides recommended strategies for using Incremental Design. It outlines the circumstances that can dictate when using this flow is the best approach and when it is not. For the most part, what might lead you to use this flow will likely be a matter of trial and error, but it is useful to gain an understanding as to when Incremental Design should be your first approach to implementing your design.

Although incremental flow in general will provide shorter runtime and relatively consistent result, there will be situations it may take longer. For example, it might be possible that incremental flow can not meet the timing goals while normal flow can. Incremental flow is suitable for well-partitioned designs. Flat designs or designs with many critical paths crossing partitions may actually result in a drop-off in performance using the Incremental Design flow.

Definitions of Terms

The following are common terms used in incremental design flow:

**Unguided iteration**  A core processing (Map, Place and Route) iteration where the physical reference design (NCD) doesn’t exist, or it does exist but is ignored. To run an unguided iteration if the reference NCD is available but needs to be ignored, set all partitions’ preservation data to “Synthesis” in Partition Manager.

**Guided iteration**  A core processing (Map, place and route) iteration where the reference NCD is used for all partitions or certain partitions, based on the user settings through the Partition Manager.

**ICF file**  The partition data base/file that holds all of the partition’s related information and user settings set through the Partition Manager. Each reference NCD has its own ICF file associated with it.
**Recommended Incremental Design Flow: Scenario 1**

This scenario applies to designs in which timing closure can be easily achieved, or has been previously addressed using incremental flow by locking down the timing critical partitions. The majority of the design is locked down, meaning no further source code changes are allowed, and the number of partitions that need to be changed are very limited.

1. In Diamond, create a new project and implementation, and add HDL codes.
2. Add synthesis constraints and adjust synthesis strategies if necessary, and complete a synthesis iteration.
3. Open the Hierarchy window, examine resource usage information for each module in order to guide partition creation.
4. Launch Synplify Pro, compile the design, then define compile points based on the observations in step 3.
5. Adjust synthesis, Map and PAR strategies if necessary, and complete synthesis, Map and PAR.
6. Examine the process reports. For these types of designs, timing closure should be easily achieved.
   
   Optionally, use Partition Manager to change Preservation Data of all partitions to **Routed** or **Placed** (default) and leave Reimplementation Effort as **Guided** (default).
7. If required, modify HDL code to certain partitions, and repeat step 5.
8. If required, repeat step 6 and step 7.

For these types of designs, timing closure can be easily achieved and runtime reductions for each iteration should be obvious.

**Recommended Incremental Design Flow: Scenario 2**

This scenario applies to designs in which timing closure can be easily achieved for certain part of a design but not in other parts of the design. For example, timing critical paths fall into certain partitions.

1. In Diamond, create a new project and implementation, and add HDL codes.
2. Add synthesis constraints and adjust synthesis strategies if necessary, and complete a synthesis iteration.
3. Open the Hierarchy window and examine resource usage information for each module in order to guide partition creation.
4. Launch Synplify Pro, compile the design, then define compile points based on the observations in step 3.
5. Adjust synthesis, Map and PAR strategies. If necessary; it is recommended that:
a. Multi-seeds PAR is enabled.
b. Routing method option is set to NBR.

6. Complete synthesis, Map and PAR.

7. Examine the process reports. Find timing critical paths and identify partitions that have timing problems. For these types of designs, timing problems should fall into certain partitions but not the others.

8. Using Partition Manager:
   a. For partitions with identified timing problems, set Preservation Data to Synthesis.
   b. For partitions without timing issue, set Preservation Data to Routed or Placed (default) and leave Reimplementation Effort to Guided (default).

9. If required, modify HDL code to identified partitions in order to improve the timing

10. Repeat step 5 and step 6

11. If required, repeat step 7 through step 10

12. When all timing issues are fixed, the design will be in a condition as described in “Recommended Incremental Design Flow: Scenario 1” on page 30. At this point, Partition Manager can be used to set all partitions Preservation Effort to Routed or Placed (default), and Reimplementation Effort to Guided.

    If there is any need to do a minor change to certain partitions, we can follow the step 7 of “Recommended Incremental Design Flow: Scenario 1” on page 30.

---

**Recommended Incremental Design Flow: Scenario 3**

This scenario applies to designs in which timing closure is hard to achieve for the whole design and partitions need to be redefined in order to isolate critical paths

1. In Diamond, create a new project and implementation, and add HDL code.

2. Add synthesis constraints, adjust synthesis strategies if necessary, and complete a synthesis iteration

3. Open the Hierarchy window and examine resource usage information for each module in order to guide partition creation.

4. Launch Synplify Pro, compile the design, and define compile points based on the observations in step 3.

5. Adjust synthesis, Map and PAR strategies if necessary. It is recommended that:
   a. Multi-seeds PAR is enabled.
   b. Routing method option is set to NBR.
6. Complete synthesis, Map and PAR.
7. Examine the process reports and identify timing critical paths. For these types of designs, there may be more than one critical path across multiple partitions. Isolate each critical path, identify modules that are included in the path, and use the information as the guideline for partition modification.
8. Using Partition Manager, set all partition Preservation Data to Synthesis.
9. If required, modify HDL code to identified critical paths in order to improve timing.
10. Based on the observation in step 7, you might need to change the partition definitions. Using Synplify Pro, add or remove compile points, as needed. It may be necessary to group certain modules that cover a critical path to create a new module to be able to define a new compile point to cover a complete critical path. In some cases, you might also want to set the partitions without timing issue with Preservation Data to Routed or Placed (default) and leave Reimplementation Effort to Guided (default)
12. If required, repeat step 7 through step 11.
13. When all timing issues are fixed, the design will be in a condition as described in “Recommended Incremental Design Flow: Scenario 1” on page 30, we can use Partition Manager to set all partitions Preservation Data to Routed or Placed (default), and the Reimplementation Effort to Guided.

If there is any need to do a minor change to certain partitions, we can follow the step 7 of “Recommended Incremental Design Flow: Scenario 1” on page 30.

**General Incremental Design Flow Limitations**

The following is a list of general limitations that you should use as a guideline for determining whether or not your design is a good candidate for running through the Incremental Design flow.

- Although the Incremental Design flow in general will provide shorter runtimes and a relatively consistent result, there will be situations it may take longer. For example, in some cases the incremental flow might not meet timing goals whereas the normal flow can.
- The Incremental Design flow is suitable for well-partitioned designs. A flat design or a design with many critical paths crossing partitions may actually result in poorer performance using the incremental flow.
- Designs that have a very high resource utilization may not be a good candidates for running in the Incremental Design flow because typically this flow requires ample resource availability.
- Designs with congestion may also not be good candidates for the Incremental Design flow for the same reason, that is, incremental designs tend to require more resource availability to resolve congestion, such as loose placement and detour routing.
The partition floorplanning that occurs in this flow can automatically generate partition constraints. A completed design is required to generate a reasonable partition. Running incremental flow on partial design is not recommended. Oftentimes the auto-generated partition constraint is not optimal. It is strongly recommended that if you must adjust partition constraints, that you do so based on your knowledge of the design.

To achieve the best stability and quick timing loop closure, it is strongly recommended that you keep the changes from one iteration to the next iteration to a minimum. Any major change or a lot of minor changes will result in a completely different result which may undermine the overall effectiveness of the incremental design flow’s inherent strategy.

The Incremental Design flow may yield loose packing and lesser performance due to partition constraints. Also, if the incremental router is applied, longer runtime and worse performance may result, if, from iteration to iteration, the change is significant or some congestion occurs.

Reveal is not generally recommended for use with the Incremental Design Flow. Any module that Reveal needs to trigger or trace into will be a changed module, so its implementation will not be preserved from the reference design. Additionally, it is likely that these debug connections will cross multiple partitions.

The Timing Driven Mapping (-tdm), Timing Driven Packing (-td_pack), and Timing Driven Node Replication (-split_node) Map strategy options are not supported in the Incremental Design flow. These timing-driven options are not supported because they do not honor design partition region boundaries.

---

**General Synthesis Recommendations**

This section lists any general recommendations to ensure the Incremental Design flow can be run in the Diamond environment and to eliminate or minimize any error or warning messages you encounter.

**The synthesis recommendations for Incremental Design are as follows:**

- A minor RTL change may result in significant difference in your mapping and place-and-routing result. For this reason, it is recommended that you make any HDL source changes before using the incremental design flow, or certainly not when you are using the guided iteration run flow mode that uses an NCD as a reference file to guide implementation. To set the next iteration as unguided iteration, change all of the partitions preservation data to “Synthesis” using Partition Manager.

- To avoid issues during synthesis, you must use Synplify Pro version of the Synplify software since Compile Points feature is not supported in other versions. The Synplify Pro version in Diamond supports the Incremental Design flow.

- In the beginning of the Incremental Design flow you must create a top-level FPGA Design Constraints (.fdc) file in Synplify Pro and associate it with the Diamond project you are running. If you do not do this, the
Incremental Design will not work properly and the flow is actually running as in a normal flow.

- Create your .fdc file in the graphical user interface of the Synplify Pro tool, not the command line. If you do not do this, module names to apply your compile points types (locked, partition) may not be recognizable to you in the command line because the modules are parameterized and appear with names such as “ro_cnt_8s_0_6s” and the like. All of these extra prefixes or suffixes like this are the parameterization.

- To discover and analyze error and warning conditions for your partitioned modules, open Synplify Pro (synplify_pro) and open the SCOPE editor panel which allows you to edit your .fdc file. Click on the Compile Points tab and observe that all of the compile points are marked in yellow, which indicates an error or warning condition.

**Avoiding Common Design Flow Issues**

You should go about using the Incremental Design flow as prescribed in this guide in a step-by-step fashion to ensure that you have initiated the flow correctly and you get the expected outputs when you run it. It is probable in these cases you have not performed a step and this has prevented Diamond from recognizing your active implementation of your project as an incremental design.

See the section in “Troubleshooting Error and Warning Messages” on page 39 on diagnosing common usage issues.

**Defining Design Partitions**

How you define your partitions is design-dependent and will have an impact on how effective using the Incremental Design flow in Diamond will be. This section provides you with useful information you can employ when you are defining partitions with compile points when you are creating your .fdc file in Synplify Pro.

Below is a list of factors involved in defining partitions that can effect the quality of results you might achieve using the Incremental Design flow. The list also includes how partitions are treated in regard to resource utilization.

**You should understand the following about how partitions can affect the incremental design flow:**

**Number of Partitions** It is recommended that you limit the number of partitions in your design to a reasonable figure, for example, 30, with respect to the percent partitions in total utilization which is described later in this section. If your design has too many partitions it will lose the global placement picture. Also, it is best to contain the portions of the design that could change to the smallest partition possible, thus preserving other parts of the design. As a general rule, if more than about 10 percent of the design is changing, it is usually advisable to re-run an unguided incremental pass.
As a recommended practice, you should not change the number of partitions in your design when running guided iterations, or you may encounter unexpected results. Defining what gets partitioned in your design should be done prior to using an NCD reference design for more incremental changes.

Size of Partitions There is no size limit for a design partition. In general, partition size is defined by a given functional module where a compile point is applied. However, any changes to a large partition can result in significant changes in how Map and PAR pack, place and route components and hence your Fmax result as well.

Shape of Partition Regions A partition’s region is always in the shape of a rectangle. A non-rectangular shape is not allowed. However, overlapping among partitions is allowed in the Incremental Design flow to achieve optimal results.

Percent Partitions in Total Utilization It is suggested that at least 75 percent of your incremental design is covered by partition instances. To find out what percentage of your design is covered by partitions, look at the “PARTITION Utilization” section in the Map Report (.mrp) file. This section of the file is shown below to illustrate how this percentage is given along with the total number of design partitions:

Total comps 450 (92.78%) in 65 partitions.

Please understand that the limit on the number of design partitions mentioned previously in this section is a more critical factor in the successful implementation of your incremental design.

Components within Partitions Understand that partition BBOX definitions keep those components in a particular design partition confined to a region that is SLICE-based but the region will not include in its boundaries the placement of special elements such as EBRs, DSP, and DDR because they are row-based. However, EBR and DSP will be placed close to a relative partition.

So, the BBOX definition does not affect the placement of these special elements at all, only logic that is placed in SLICE components or PFU/PFF elements. If you want to place special elements within a specific physical region, use the UGROUP or LOCATE preferences as appropriate. In addition, due to architecture limitations, not all global resources will be able to be included in an individual partition. For example, global resources such as PLL, DLL, PCS, DCS, CLKDIV, and DLLDEL may not be used in the same partitions due to complicated rules and shape of ring. PIO/IOL (particularly clock PIO) are considered a global resource as well, which means PIO/IOL will not be included in a partition.

Note

Any conflict between a user preference and a partition definition must be resolved by the user. The placer will simply error out when encounters a conflict, such as a PGROUP or carry-chain crossing different partitions.
Placement Variability in Guided Iterations

It is important to understand how guided iterations use the reference design to guide placement. The major concept to understand is that when you are using the incremental flow, the default settings in the data file preserve placement of the reference design unless the design’s source HDL changes. This is implemented that way by design.

So, you need to be aware that a poor placement result will be preserved as readily as a good placement result. For this reason, we recommend that you get your design close to timing closure in the unguided iteration run flow mode before switching to the guided iteration run flow mode.

You should expect to see no variation in timing inside a partition with unchanged HDL in incremental flow when you run multiple cost tables with default data file settings. Also, if you attempt to change placement in the guided iteration by changing UGROUP or LOCATE preferences, you may see unexpected results. See the next chapter, "Troubleshooting Error and Warning Messages" on page 39 for related issues.

Handling Timing Issues in Guided Iterations

If you are already in guided iteration and are seeing timing errors reported in your TRACE report, you have the following three options:

- Go back to unguided iteration run flow mode. This would be the appropriate choice if you believe changes to the partition planning or major HDL changes throughout the design are required.
- If the timing errors can be addressed by making HDL changes (e.g., adding pipeline stages), stay in guided run flow mode and use default preservation data and re-implementation effort settings for all partitions.
- If you believe the timing errors could be addressed by allowing some variability into the placement of a partition where HDL is not changed, you can achieve this by changing settings through the Partition Manager.

To do this, open Partition Manager, and for all partitions that you want variability, change their preservation data to "mapped" and reimplementation effort to "guided." Then, run multiple cost tables and force a PAR rerun. After doing this you will see some variability in the result for that partition within the constraints of the partition planning that appear in the data file.

Adding Pipeline Registers to Incremental Designs

The Incremental Design flow is well suited to the case where you want to add pipeline registers and keep the rest of the design unchanged for placement. In guided iteration run flow mode, add the registers to your HDL and rerun the flow. No manual intervention should be necessary.

If for some reason you require that the new registers go into a specific area that is within their partition boundary and that has enough open sites to place the new registers, you can do this. In this particular case, add a UGROUP to
the new registers with bounding box (BBOX) and anchor point (SITE). Note that this is only an option with changed logic. For more information, see the troubleshooting section on UGROUPs in incremental mode.
Troubleshooting Error and Warning Messages

This chapter provides a listing of error and warning messages that are specific to running the Incremental Design flow in Diamond. The chapter also includes a list of current known issues that you might encounter during the flow and provides some information to help you work around them effectively.

Troubleshooting Missing Files and Folders

This section provides some examples of common issues that occur to users who are missing some file or folder inherent to the Incremental Design flow and they cannot understand the cause of it and cannot proceed.

An Incremental Design flow user could encounter cases where they expect that there is some file or folder in a certain location, but for some reason it does not exist. In most cases, this happens because they have skipped an important step in the design flow steps as they are described in “Running the Incremental Design Flow.”

The following is a list of the most common reasons why you are not seeing the files and folders you would expect to see in the Incremental Design flow:

- You did not create an FPGA Design Constraints (.fdc) file in Synplify Pro.
- You did not create any compile point constraints in your .fdc file in Synplify Pro.
- You did not associate your .fdc file with your Diamond project.
- You did not enable the Incremental Design flow.
Known Issues

The section lists known issues that you might encounter in the Incremental Design flow. If there is a workaround, that information is provided.

**Extra partition is defined by Incremental Design flow**  ▶ In some cases, after Synthesis, an extra partition is defined by the Incremental Design flow. For example, if you have 10 compile points defined in Synplify Pro, the Map or PAR may report more than 10 partitions. There are two possible reasons:

▶ The top level design is always treated as a partition.

▶ Synplify Pro compile points are module based. Therefore, if a module contains a compile point defined with “n” number of instances, then effectively there will be “n” number of partitions for that module.

**Missing SITE/BBOX definition for partition in ICF**  Sometimes after running, the PAF utility does not assign an anchor SITE and regional BBOX to the PARTITION setting in the ICF file. In these cases you will see a message in the .par file (i.e., if the PAF is run inside par as in Diamond) similar to the following:

INFO: PARTITION CGROUP_serdes_quad has no valid comp, this partition will be ignored.

And then later,

PARTITION 2 : “CGROUP_serdes_quad” (SITE and BBOX are not assigned)

There will be no usage information shown in this case because there are no slices in the partition.

**Issue**: One reason PAF does not define a SITE and BBOX for a given partition in the ICF is that your compile point constraint does not include any SLICEs. This could happen for the following reasons:

▶ They are optimized away. This may be expected behavior, but also may happen if inputs or outputs do not connect to ports. check tool logs)

▶ There are actually no contents of the partition that are slices. This could happen, for example, if you put a compile point on a module which only contains a SERDES block, or only I/O. This is not an error condition but a compile point like this does serve any purpose for Map and PAR.

**Solution**: Check your tool log files in Synplify and attempt to determine why your inputs and outputs are not connecting to ports.

**Using I/O buffers with tristate controls in modules with compile points**  If I/O buffers with tristate controls are present within modules with compile points defined in your design, you will encounter error messages during implementation in Diamond.

**Issue**: The synplify_pro synthesis tool cannot support I/O buffers with tristate controls inside of modules with compile points. If the buffers are instantiated, it will detect them, issue a warning message, and disable the compile point.
Check the Synplify log or the warning tab in Diamond. The message will appear similar to the following example shown below:

@W: BN106 |Cannot apply constraint syn_compile_point to v:work.immir

@W: BN106 |Cannot apply constraint syn_cptype to v:work.immir

If you miss this warning message, you will only see this after the Map process, when you are missing partitions in the Map Report (.mrp) PARTITION report and the ICF file.

**Solution**: Either do not use a compile point on the module or move the buffers up to the top level of the design's hierarchy in the HDL.

**Unused design elements may generate Map errors in modules with compile points**  Using compile points with designs that contain unused logic may result in the unintended functionality and Map error messages.

**Issue**: If you use compile points during synthesis and your design contains unused elements, for example, a multiplier exists in a test case that is driven by a driver in another partition, but the outputs are not intended to be used, it will likely result in Map errors with messages similar to the following.

ERROR - map: Mico32_u/LM32/cpu/multiplier/product_2_36_71 : Pin A13 has no driver. Possible causes are (1) redundant logic or (2) undriven input.

ERROR - map: Mico32_u/LM32/cpu/multiplier/product_2_36_71 : Input B Pin B0 is not connected but the Source is statically connected to GND(Parallel Input)

**Solution**: Either do not use a compile point on the module or remove any unused logic from your design before mapping.

**Changing I/O placement and function**  The Incremental Design flow tries to preserve the I/O placement of the reference design. In the incremental flow, you can change I/O placement via new LOCATE preferences, where the new placement is feasible – that is, where the pins are available, and support the function of the I/O.

The new preferences will be honored and you will see messages, as shown below, in the PAR Report (.par) file that inform you that the placement in the reference design is being overridden by the new LOCATE preferences.

WARNING - par: Comp 're' has been LOCATE'd and will not be placed as per guided file

**Issue**: Changing I/O function in the incremental flow can potentially cause an I/O placement failure. An example of a problem like this might be changing a non-DDR I/O to a DDR, or changing an I/O type from LVCMOS to LVDS. The original placement may no longer be feasible due to the resource availability at the reference design's pin locations.

**Solution**: If you see an I/O placement failure in the Incremental Design flow, change your LPF preference file to manually locate the changed I/O to the
appropriate pins or, instead of using the incremental flow, use the normal run flow mode.

**Parameterized modules in standalone Synplify Pro** Using the standalone Synplify Pro software for Incremental Design, parameterized modules in a design make it impossible to determine what their actual module names are and results in error messages later in the design flow that are difficult to interpret.

**Issue:** In this specific case, a user encountered a Map error that prevented the ICF file from being opened which occurred using an unguided iteration run flow mode they attempted to process through Diamond’s Place & Route Design process. The following Map error message was reported in the Output view:

```
WARNING - map: There is no partition, and the option "-inc" is ignored.
```

This warning implies that there were no PARTITION properties or constraints recognized by Map so they were not passed through the implementation flow. The .fdc file contains compile points that become PARTITION settings in the Incremental Design flow.

This condition occurred when the user attempted build their .fdc constraints file in a standalone Synplify Pro software and because of parameterized module naming not matching the actual module names, it appears to the software that there were no compile points. In the Synplify Report the following “Cannot apply constraint” error message occurred:

```
@W: BN106 |Cannot apply constraint syn_compile_point to v:work.ud_cnt
@W: BN106 |Cannot apply constraint syn_cptype to v:work.ud_cnt
@W: BN106 |Cannot apply constraint syn_compile_point to v:work.ro_cnt
@W: BN106 |Cannot apply constraint syn_cptype to v:work.ro_cnt
@W: BN106 |Cannot apply constraint syn_compile_point to v:work.ud_cnt_uniq_1
@W: BN106 |Cannot apply constraint syn_cptype to v:work.ud_cnt_uniq_1
@W: BN106 |Cannot apply constraint syn_compile_point to v:work.ro_cnt_uniq_2
@W: BN106 |Cannot apply constraint syn_cptype to v:work.ro_cnt_uniq_2
```

In addition, if you reopen the standalone version of Synplify Pro, choose **Run > Compile Only** and open the .fdc file using the SCOPE editor spreadsheet panel. Now, click on the Compile Points tab and observe that all of the compile points are marked in yellow. This indicates that there is an error condition you have to resolve before you proceed.

**Solution:** You should create your .fdc file inside of the Synplify Pro graphical user interface and it is highly recommended that you run Synplify Pro for Lattice from within the Diamond interface and associate the file in the Synthesis strategy settings as prescribed. This ensures that source and .fdc constraints files are recognized and associated with your Incremental Design flow project in Diamond.

**Preference/net naming issues** Preference and net names can vary from design to design in the Incremental Design flow for several possible reasons.
Troubleshooting Missing Files and Folders

**Issue:** In some cases preferences that are honored in the normal flow are not honored in the incremental flow. For example, a clock name has significantly changed which leads to a FREQUENCY constraint, related to that clock not honored; a net name has changed which leads to a MAXDELAY constraint, related to that net, not honored; or a Dynamic Clock Select (DCS) name has changed which leads to USE PRIMARY DCS NET "<net_name>" "<quadrant_location>" not honored. This may be due to synthesis optimization being performed differently with compile points present as opposed to when they are absent from a design.

**Solution:** When using incremental design, you should double check Map warnings about preferences. If you see warnings indicating that preferences are ignored which are accepted in normal flow, you may need to modify your preference file, or use synthesis attribute "syn_keep" to preserve net names. Another method, for checking changed clock names, is to open Spreadsheet View after running the Translate Design process in Diamond. In the Spreadsheet View, check the Clock Resource tab. This tab lists all of the clock names that are recognized by the tool, including those that have their names changed. Users should use the names listed in this tab to set their constraints. As always, it is better to constrain ports rather than nets if possible, as they will not be renamed.

**UGROUPs in guided iterations**  In certain cases UGROUPs you assign to unchanged SLICE components as attributes or preferences are not honored.

**Issue:** When you are running in the guided iteration run flow mode and you attempt to move unchanged SLICEs relative to the reference design by adding a UGROUP to HDL or preference file, the placer will not, by default, honor the new UGROUP location. There is no error or warning messages to alert you to this condition.

**Solution:** To avoid this situation, have your UGROUPs defined where you desire them to be located when you run the first implementation. While changed logic can be located via a UGROUP, this will only work where there are empty sites available. The placer will not rip up any unchanged placed logic from a reference design by default.

**Tcl Commands Can Only Be Run Through Tcl Shell**  The database extended Tcl commands and Partition Extended Tcl commands can only be run through a Tcl Shell, and cannot be run in the Diamond GUI.

**General Warnings & Errors**
You may encounter the following general warnings and errors when using the Incremental Design flow.

- If the user's command has a reference file specified, such as par -w ...
  <file_name>.ncd, but file.ncd does not exist, PAR will switch to Unguided iteration with the following warning message:

  Failed to open reference file %s, switch to unguided iteration

- If the user's .icf file has no partition, or all partitions are disabled, PAR will switch to normal flow with the following error message:
No partition is defined or all partitions are disabled, switch to normal flow.

- If the user's command + d has a reference file specified, such as par -w ... -<file_name>.ncd, and the level and effort settings in the Partition Manager meet one or both of the following conditions:
  - Preservation data = synthesis and any reimplementation effort.
  - Preservation data = Map, reimplementation effort = unguided.

PAR will switch to Unguided iteration with the following warning message:

Because of the level/effort setting in icf file, redo unguided iteration.

**Note**

If this warning occurs, the Floorplan view will be invoked, so the regions of partitions will be changed.

- If the user's command has a reference file specified, such as par -w ... -<file_name>.ncd, but one of partitions has changed the anchor or bbox through the Partition Manager, PAR will switch to unguided iteration with the following warning message:

Because some of the partitions changed partition plan, guide iteration is disabled; The regions in icf files will be honored but the locations of comp will be changed.

## Map Warnings & Errors

You can encounter the following warning and errors in the Incremental design flow when using the Map process.

**WARNING – map:** There is no partition, and the option "-inc" is ignored.

**This condition may be the result of the following:**

- No partition has been found so Map will run in normal mode.
- No ICF file will be created.
- The "par -inc -icf" command may fail due to missing ICF file.
- There is no partition, so you cannot use the incremental flow.

**WARNING – map:** The time stamp %s for partition %s in current design is older than the reference one %s.

**This condition may be the result of the following:**

- The time stamp for a given partition in current design is older than the reference one. This condition may have occurred when a modified
design is used as a reference for the previous design. This practice is not recommended in the Incremental Design flow.

- The partition will be marked as “CHANGED”.

**ERROR - map: Sorry, there is no license for %s.**

This condition may be the result of the following:

- The Incremental Design flow requires a special feature license. If you do not have it in your license file, you will see the message. Go to the Lattice web site and contact Sales support to obtain a license.

**ERROR - map: Sorry, device %s is not supported for %s.**

This condition may be the result of the following:

- Incremental flow may not be supported for all Lattice FPGA/CPLD families. You get this message because the selected device is not supported for incremental flow.

---

**Placer Warnings & Errors**

You can encounter the following warning and errors in the Incremental design flow as a result of the placer processing your design.

- For running in a multi-par (multiple seeds) format, after each seed, par may issue one of the following three kind of messages during the Incremental Design flow:

  **WARNING: “message …”**
  
  PAR continues after this warning message is issued.

  **ERROR: “message …”**
  
  PAR will exit after it issues this error message.

  **WARNING: placement not success in this seed and will continue next seed if exists. “message …”,**
  
  PAR will abort running for the current seed after this warning message and try the next seed. The message could be one of the follows:

  **ERROR :**
  
  “Internal database corrupted: NULL site in slice placement”

  **NOT SUCCESS :**
  
  “Fatal Error in preferplace(): cannot find initial placement for UPGROUPs. Please check the UPGROUP preferences to alleviate constraints.”
“Impossible to place SLICE_10 in UPGROUP GROUP1 in prefer location.”

“Placer could not relatively place comp SLICE_10 at R15C8D(x=61/y=8.”

“Failed to find the anchored location.”

▶ Anchoring error during PGROUP pre-placement:

pgroup=SLICE_10 anchor=R15C8D. Not enough available sites in bbox for all group comps. Please check for multiple LOCATE/REGION/QUADRANT design preferences leading to highly constrained PGROUP placement regions.

Error in pgroup initial placement!

Comp ‘SLICE_10’ remains unplaced after Phase 0 flattening.

set_initial_anchors failure on PGROUP placement"

WARNING:

switch to high-effort for PGROUP initial placement due to congestion...

The following are messages issued by the placer. For each PAR incremental design run, the PAR file will have the following info:

Starting incremental place and route ...

Incremental design will exit if you encounter any of the following PAR error messages:

ERROR: No *.icf file for incremental par flow.
ERROR: Failed to open file design.icf for incremental par flow.
ERROR: Incremental flow can’t support guide file with option “-g filename”.
ERROR: option -g and -ref can’t be appeared in command line simultaneously.
ERROR: file name missing after "-ref".
ERROR: Failed to open reference file design_ref.ncd.

ICF Parser Warnings & Errors

The following are messages issued by the ICF parser. If the ICF file was loaded successfully, it will issue the following message:

Completed loading 3 partition data.

▶ If the ICF parser detects an error, it may issue any of the following error messages listed below:

ERROR: comp%d %s belong to both partition %s and %s.
ERROR: Illegal keyword XXX
ERROR: Must assign value for keyword XXX for partition P0.
ERROR: Invalid icf string
ERROR: Fail to open ICF file XXX.
ERROR: No statement in ICF file XXX.
ERROR: Illegal icf key-value XXX.
ERROR: XXX YYY is an illegal pair, value must be in "ZZZ".

Then PAR will exit after printing out the following line:
There are %d errors in icf file.

Router Warnings & Errors

You can encounter the following warning and errors in the Incremental design flow as a result of the router processing your design.

▷ Usually, in incremental flow, the positions of primary clock signals are copied according to the reference design. However, for any reason, if the position of a primary clock signal has been preassigned, the copy will be skipped and you will encounter the following warning message:

WARNING – par: The pclk position for the signal %s has been already assigned, and the assignment from the guided/reference design is ignored.

▷ As mentioned in the prior case, in incremental flow, the clock assignment is usually copied from reference design. However, for any reason, if the target position has been used by other signals, the copy will be skipped and you will encounter the following message:

WARNING – par: The pclk position %d has been already occupied. The pclk assignment for the signal %s copied from guided/reference design is ignored.

▷ Usually in the incremental flow, the positions of secondary clock signals are copied according to the reference design. However, for any reason, if the position of a secondary clock signal has been preassigned, the copy will be skipped and you will encounter the following message:

WARNING – par: The sclk position for the signal %s has been already assigned, and the assignment from the guided/reference design is ignored.

▷ As mentioned above, in incremental flow, the clock assignment is usually copied from reference design. However, for any reason, if the target position has been used by other signals, the copy will be skipped and you will encounter the following message:

WARNING – par: The sclk position %d has been already occupied. The sclk assignment for the signal %s copied from guided/reference design is ignored.

Supplying Archives for Technical Support

As a general advisory, if you encounter issues that require debugging by Lattice technical support, it would be extremely helpful to provide project archives both for the initial reference design files and the guided iteration run flow mode version where the issue was encountered.
As described earlier in this guide in the section, “Archiving Incremental Design Results” on page 12, make sure that at the appropriate points in the flow that you archive your project.