Virtuoso Design Planning and Analysis User Guide

Product Version ICADVM20.1
October 2020

Printed in the United States of America.

Cadence Design Systems, Inc. (Cadence), 2655 Seely Ave., San Jose, CA 95134, USA.

Disclaimer: Information in this publication is subject to change without notice and does not represent a commitment on the part of Cadence. Except as may be explicitly set forth in such agreement, Cadence does not make, and expressly disclaims, any representations or warranties as to the completeness, accuracy or usefulness of the information contained in this document. Cadence does not warrant that use of such information will not infringe any third party rights, nor does Cadence assume any liability for damages or costs of any kind that may result from use of such information.

Contents

B

Design Planning Forms

Preface

This document describes how to use the Virtuoso® Design Planning and Analysis tool to provide hierarchical generation capabilities in connectivity-driven and constraint-driven flows, including electrically-aware flows. Design Planning and Analysis provides full flexibility in planning and analyzing analog and mixed-signal chips, and blocks using hard blocks, soft blocks, and virtual hierarchies in a top-down or bottom-up approach, implementing the best of both design methodologies. In addition, the tool provides seamless placement planning, global route planning, and analysis capabilities, allowing optimization of layout plans at the top level, block level, and cell level.

This document is aimed at developers and designers of integrated circuits and assumes that you are familiar with:

- The Virtuoso design environment and application infrastructure mechanisms supporting consistent operations between all Cadence tools.
- The applications for designing and developing integrated circuits in the Virtuoso design environment, notably the Virtuoso Layout Suite XL layout editor and the Virtuoso Floorplanner.
- Virtuoso technology data.
- Component description format (CDF), which lets you create and describe your own components for use with Layout XL.

This preface contains the following topics:

- Scope
- Licensing Requirements
- Related Documentation
- Additional Learning Resources
- Customer Support
- Feedback about Documentation
- Typographic and Syntax Conventions

Scope

Unless otherwise noted, the functionality described in this guide can be used in the ICADVM20.1 release.

Licensing Requirements

For information about licensing in the Virtuoso design environment, see *Virtuoso Software Licensing and Configuration Guide*.

Related Documentation

This document does not contain information on all the functions and commands enabled in Layout EXL. Many of the features available in Virtuoso Layout Suite EXL, Virtuoso Layout Suite XL, and Virtuoso Floorplanner are described in the dedicated user guides. Where this is the case, you will find specific references to the documents that contain the most detailed information.

For information on any of the basic layout creation and editing commands supported in Layout XL, see:

- *Virtuoso Layout Suite XL: Basic Editing User Guide*
- *Virtuoso Layout Suite XL: Connectivity Driven Editing User Guide*

For information on any commands supported in Layout EXL, see *Virtuoso Layout Suite EXL Reference*.

For information on any commands supported in Virtuoso Floorplanner, see *Virtuoso Floorplanner User Guide*.

What’s New and KPNS

- Virtuoso Design Planning and Analysis What's New
- Virtuoso Design Planningand Analysis Known Problems and Solutions

Installation, Environment, and Infrastructure

- *Cadence Installation Guide*

- *Virtuoso Design Environment User Guide*
- *Virtuoso Design Environment SKILL Reference*
- *Cadence Application Infrastructure User Guide*

Technology Information

- *Virtuoso Technology Data User Guide* and the *Virtuoso Technology Data ASCII Files Reference*
- *Virtuoso Technology Data SKILL Reference*
- *Virtuoso Technology Data Constraints Reference*

Virtuoso Tools

- *Virtuoso Layout Viewer User Guide*
- *Virtuoso Layout Suite XL: Basic Editing User Guide*
- *Virtuoso Layout Suite XL: Connectivity Driven Editing*
- *Virtuoso Layout Suite EXL Reference*
- *Virtuoso Concurrent Layout User Guide*
- *Virtuoso Multi-Patterning Technology User Guide*
- *Virtuoso Placer User Guide*
- *Virtuoso Simulation Driven Interactive Routing User Guide*
- *Virtuoso Width Spacing Patterns User Guide*
- *Virtuoso RF Solution Guide*
- *Virtuoso Electromagnetic Solver Assistant User Guide*

Relative Object Design and Inherited Connections

- *Virtuoso Relative Object Design User Guide*
- *Virtuoso Schematic Editor User Guide*

Additional Learning Resources

Video Library

The Video Library on the Cadence Online Support website provides a comprehensive list of videos on various Cadence products.

To view a list of videos related to a specific product, you can use the *Filter Results* feature available in the pane on the left. For example, click the *Virtuoso Layout Suite* product link to view a list of videos available for the product.

You can also save your product preferences in the Product Selection form, which opens when you click the *Edit* icon located next to *My Products*.

Virtuoso Videos Book

You can access certain videos directly from Cadence Help. To learn more about this feature and to access the list of available videos, see Virtuoso Videos.

Rapid Adoption Kits

Cadence provides a number of Rapid Adoption Kits that demonstrate how to use Virtuoso applications in your design flows. These kits contain design databases and instructions on how to run the design flow.

In addition, Cadence offers the following training courses on Virtuoso Layout Suite XL:

- Virtuoso Connectivity-Driven Layout Transition
- Virtuoso Layout Pro: T3 Basic Commands (XL)
- Virtuoso Layout Pro: T4 Advanced Commands (XL)
- Virtuoso Layout for Advanced Nodes

To explore the full range of training courses provided by Cadence in your region, visit Cadence Training or write to training_enroll@cadence.com.

Note: The links in this section open in a separate web browser window when clicked in Cadence Help.

Help and Support Facilities

Virtuoso offers several built-in features to let you access help and support directly from the software.

- The Virtuoso *Help* menu provides consistent help system access across Virtuoso tools and applications. The standard Virtuoso *Help* menu lets you access the most useful help and support resources from the Cadence support and corporate websites directly from the CIW or any Virtuoso application.
- The Virtuoso Welcome Page is a self-help launch pad offering access to a host of useful knowledge resources, including quick links to content available within the Virtuoso installation as well as to other popular online content.

  The Welcome Page is displayed by default when you open Cadence Help in standalone mode from a Virtuoso installation. You can also access it at any time by selecting *Help – Virtuoso Documentation Library* from any application window, or by clicking the *Home* button on the Cadence Help toolbar (provided you have not set a custom home page).

For more information, see Getting Help in *Virtuoso Design Environment User Guide.*

Customer Support

For assistance with Cadence products:

- Contact Cadence Customer Support

  Cadence is committed to keeping your design teams productive by providing answers to technical questions and to any queries about the latest software updates and training needs. For more information, visit https://www.cadence.com/support.
- Log on to Cadence Online Support

  Customers with a maintenance contract with Cadence can obtain the latest information about various tools at https://support.cadence.com.

Feedback about Documentation

You can contact Cadence Customer Support to open a service request if you:

- Find erroneous information in a product manual
- Cannot find in a product manual the information you are looking for

- Face an issue while accessing documentation by using Cadence Help

You can also submit feedback by using the following methods:

- In the Cadence Help window, click the *Feedback* button and follow instructions.
- On the Cadence Support portal Product Manuals page, select the required product and submit your feedback by using the *Provide Feedback* box.

Typographic and Syntax Conventions

The following typographic and syntax conventions are used in this manual.

text — Indicates names of manuals, menu commands, buttons, and fields.

`text` — Indicates text that you must type exactly as presented. Typically used to denote command, function, routine, or argument names that must be typed literally.

`z_argument` — Indicates text that you must replace with an appropriate argument value. The prefix (in this example, `z_`) indicates the data type the argument can accept and must not be typed.

| — Separates a choice of options.

`{ }` — Encloses a list of choices, separated by vertical bars, from which you **must** choose one.

`[ ]` — Encloses an optional argument or a list of choices separated by vertical bars, from which you **may** choose one.

`[ ?argName t_arg ]`

Denotes a key argument. The question mark and argument name must be typed as they appear in the syntax and must be followed by the required value for that argument.

`...` — Indicates that you can repeat the previous argument.

Used with brackets to indicate that you can specify zero or more arguments.

Used without brackets to indicate that you must specify at least one argument.

`,...` — Indicates that multiple arguments must be separated by commas.

`=>` — Indicates the values returned by a Cadence® SKILL® language function.

`/` — Separates the values that can be returned by a Cadence SKILL language function.

If a command-line or SKILL expression is too long to fit within the paragraph margins of this document, the remainder of the expression is moved to the next line and indented. In code excerpts, a backslash (\ ) indicates that the current line continues on to the next line.

1

Getting Started with Design Planning and Analysis

Introduction to Design Planning and Analysis

This chapter describes how to start the Virtuoso® Design Planning and Analysis flow in Layout EXL. In addition, this section introduces the layout *Design Planning* workspace, the buttons available on the *Design Planning* toolbar, the commands available in the *Plan* menu, and the syntax to be used for setting the associated environment variables.

You can use environment variables to change the value of many aspects of your environment either for an individual design session or permanently until you change the value of the environment variable again.

This chapter covers the following topics.

- Prerequisites for using the Design Planning and Analysis Flow
- Plan Menu
- Design Planning Toolbar
- Design Planning Workspace
- Setting Environment Variables

Prerequisites for using the Design Planning and Analysis Flow

To use the Virtuoso Design Planning and Analysis flow, you must have access to the following Virtuoso design environment capabilities:

- **Virtuoso Release:** *ICADVM20.1*

  For information about licensing in the Virtuoso design environment, see *Virtuoso Software Licensing and Configuration Guide*.

- **Virtuoso Layout Suite Application:** *Virtuoso Layout Suite EXL*

  For information about the Layout Suite EXL editor, see *Virtuoso Layout Suite EXL Reference*.

- **Virtuoso Floorplanning Application:** *Virtuoso Layout Suite XL*

  For information about the floorplanning application, see *Virtuoso Floorplanner User Guide*.

Accessing the Design Planning Commands

To access the Design Planning commands, do one of the following:

- In the Layout EXL workspace drop-down, choose *Design_Planning*.

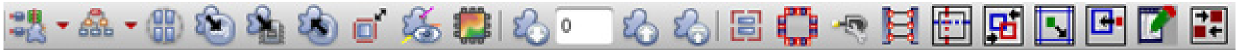
  The Design Planning Workspace displays.

- In the Layout EXL menu bar, choose the *Plan* menu.

  The Plan Menu displays.

- Right-click the menu bar in the Layout EXL window and choose *Toolbars – Design Planning*.

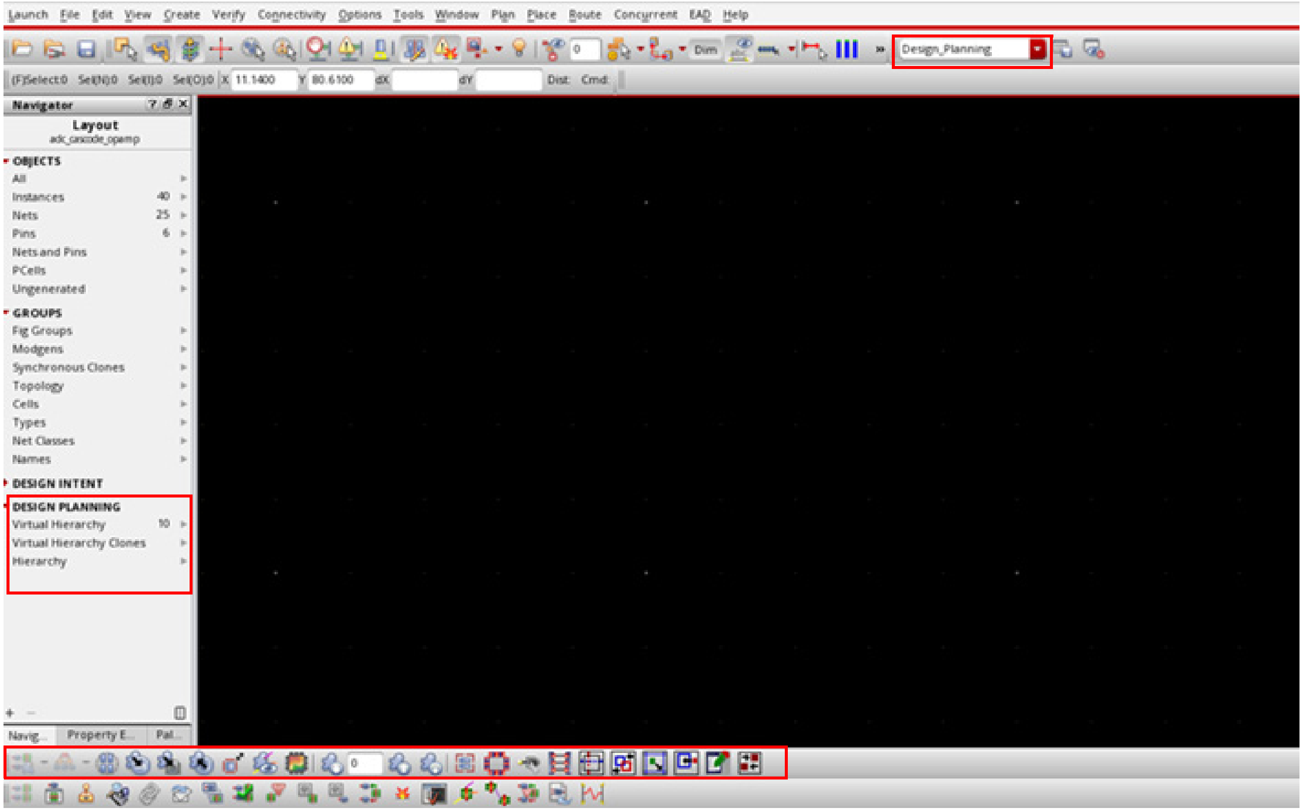
  The Design Planning Toolbar displays.



Design Planning Workspace

The default workspace for performing design planning and analysis tasks is called *Design Planning*.

The *Design Planning* workspace shows the layout canvas, the Design Planning Toolbar, and the Navigator assistant. When opened in the Design Planning workspace, the Navigator assistant displays the DESIGN PLANNING category at the bottom, which lists related datasets such as *Virtual Hierarchy* and *Virtual Hierarchy Clones*.



Plan Menu

The *Plan* menu gives you access to the various commands and related forms that are needed to support the design planning and analysis tasks.

Note: The most frequently used commands from the *Plan* menu are also available on the Design Planning Toolbar.

For more information on the individual commands in the Plan menu, see the table below.

Command	Submenu Command	Use to...	Form
Generate			

Command	Submenu Command	Use to...	Form
	Generate All From Source	Generate a virtual hierarchy for the schematic instances that have no layout counterparts generated and create soft blocks for schematic symbols with no schematic or virtual hierarchy.	Generate Layout
	Generate Selected From Source	Generate a virtual hierarchy for the selected schematic instances and create soft blocks for schematic symbols with no schematic or virtual hierarchy.	Generate Selected Components
	Reinitialize	Make collections of design objects based on their cell type and place them around the design boundary.	Reinitialize Floorplan
	Load Physical View	Import cellview information from an existing layout cellview.	Load Physical View
	Auto Generate Hierarchy	Generate a physical hierarchy by applying common parameters for boundary and pins. In addition, specify the area estimator function to use for all the blocks.	Auto-Generate Hierarchy
	Generate Physical Hierarchy	Generate the components in a design while maintaining the hierarchy levels defined in CPH Soft Block mode.	Generate Physical Hierarchy
Manage Hierarchy			

Command	Submenu Command	Use to...	Form
	Create Virtual Group	Create a group around the selected top-level instances or instances inside a virtual hierarchy. The created group can be opaque or transparent and can be automatically placed when resized.	Create Virtual Group
	Make Cell	Creates new cellviews using virtual hierarchies selected from the top cellview.	Make Cell
	Make Virtual Hierarchy	Integrate layout hierarchies that were realized outside the design.	Make Virtual Hierarchy
	Remaster	Replace the selected virtual hierarchy block with the selected layout master that exists on disk.	Remaster
I/O Planning			
	I/O Row Create	Create IO rows for the PAD type `siteDefs` available in the technology file.	Create IO Row
	I/O Pad Placer	Place instances of cell type `PAD` in IO rows.	IO Pad Placement
	Corner Pad Placer	Insert corner cells between IO rows to maintain signal continuity between IO rings.	Corner Pad Placement
	Insert I/O Filler	Insert filler cells in the empty spaces between pad instances to maintain signal continuity in the IO row.	Insert Filler Cells
Block Planning			

Command	Submenu Command	Use to...	Form
	Adjust Boundary	Resize or create the area boundary for the selected virtual hierarchy, the PR boundary for the selected soft block, or the top-level PR boundary.	Adjust Boundary
	Place Blocks	Automatically place all hard and soft blocks in a design and minimize the wire length and chip area.	Block Placer
	Load Solutions	Load other solutions that block placer generates.	Load Solutions
	Report Placement Statistics	Display the placement statistics report after block placement.	None
	Adjust Blocks	Adjust the existing floorplan by abutting and pushing the soft blocks in the design.	None See Adjust Blocks
	Remove Block Overlaps	Remove overlaps and place overlapping hard blocks and soft blocks adjacent to each other.	None See Remove Block Overlap
	Snap Soft Blocks to Grid	Snap soft blocks to the placement grid or to the manufacturing grid.	None See Snap Soft Blocks to Grid
	Pull Soft Blocks inside PR Boundary	Pull the soft blocks overlapping the PR boundary into the core area.	None See Pull Soft Blocks Inside PR Boundary
	Edit Soft Blocks	Modify soft block attributes.	Edit Soft Blocks
	Push Pre-Routes	Push the top-level implementation of power structures and signal routes into the block level, either as route or as a blockage.	Push into Blocks

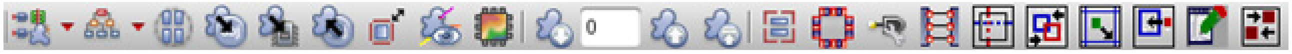
Command	Submenu Command	Use to...	Form
Pin Planning			
	Pin Tool	Create, resize, plan, and optimize pins, edit pin attributes, and set pin location constraints.	None See Using the Pin Tool
	Pin Planner	Assign or refine pin constraints and pin attributes.	Pin Placement
	Pin Optimization	Position block pins to minimize the net length honoring process rules and design constraints such as `min-spacing`, `min-width`, `wireType`, `order`, and `edge` at a particular level in the design.	Pin Placement
	Pin Checker	Run checks on pins and report the results.	Pin Checker
	Create/Edit Pin Group Guide	Create or edit pin groups and guides interactively.	Pin Group Guide
	Compare Pin Group Guides	Compare the pin group guides in two cellviews.	Compare Pin Group Guides
	Place Pin Group Guides as Schematic or Symbol	Generate pin group guides and place pins based on their relative positions in the schematic or symbol view.	Pin Group Guide – Place As Schematic/Symbol
	Create Soft Pins	Create additional soft pins on soft blocks.	Add Soft Pin
	Snap Pins	Snap top-level and level-1 pins to the grid appropriate to the block type.	Snap Pins

Command	Submenu Command	Use to...	Form
	Label Update	Update pin labels and text displays for clearer visualization.	Label Update
Placement Planning			
	Placement Planning	Define placement locations for transistors and other design elements.	Placement Planning
	Auto Place	Access the automatic placement tools to perform standard cell placement or to refine the existing analog device placement.	Automatic Placement
Route Planning			
	Congestion Analysis	Analyze the routing capacity of a design.	Congestion Analysis
	Track Pattern	Select a width spacing pattern for a region or global area that is displayed as tracks on the canvas.	Track Pattern Assistant
	Global Route All Nets	Run global route for all the nets in the layout.	Global Route
View			
	Analyze Connectivity	Draw flight lines between each connected pair of virtual hierarchy groups, soft blocks, or hard blocks. Select pairs of virtual hierarchy, soft block, or hard block to see the connections between them.	Analyze Connectivity

Command	Submenu Command	Use to...	Form
	Block Annotations	Display information about blocks.	Block Annotations Options
Options		Control the display of virtual hierarchies and blocks in a design.	Design Planning and Analysis Options

Design Planning Toolbar

The Design Planning toolbar gives you quick access to some commands that are useful for performing design planning and analysis. As with all Virtuoso toolbars, you can move the Design Planning toolbar anywhere within the layout window and dock it at an appropriate position. By default, the toolbar displays at the bottom of the layout window.



Note: Most commands available on the Design Planning toolbar can also be accessed using the Plan Menu.

For more information on the individual buttons on the Design Planning toolbar, see the table below.

Command	Use to...	Form
Generate All From Source **Note:** The *Generate All From Source* command is available under a drop-down menu. The command can be toggled with the *Generate Selected From Source* command.	Generate a virtual hierarchy for the schematic instances that have no layout counterparts generated and create soft blocks for schematic symbols with no schematic or virtual hierarchy.	Generate Layout

Command	Use to...	Form
Generate Selected From Source **Note:** The *Generate Selected From Source* command is available under a drop-down menu. The command can be toggled with the *Generate All From Source* command.	Generate a virtual hierarchy for the selected schematic instances and create soft blocks for schematic symbols with no schematic or virtual hierarchy.	Generate Selected Components
Generate Physical Hierarchy **Note:** The Generate *Generate Physical Hierarchy* command is available under a drop-down menu. The command can be toggled with the *Auto Generate Hierarchy* command.	Generates the physical hierarchy based on the configuration defined in Configure Physical Hierarchy (CPH).	Generate Physical Hierarchy
Auto Generate Hierarchy **Note:** The *Auto Generate Hierarchy* command is available under a drop-down menu. The command can be toggled with the Generate Physical Hierarchy command.	Generate a physical hierarchy by applying common parameters for boundary and pins. In addition, specify the area estimator function to use for all the blocks.	Auto-Generate Hierarchy

Command	**Use to...**	**Form**
Create Virtual Group	Create a group around the selected top-level instances or instances inside a virtual hierarchy. The group can be made opaque or transparent and can be automatically placed when resized.	Create Virtual Group
Make Cell	Creates new cellviews using virtual hierarchies selected from the top cellview.	Make Cell
Remaster	Replace the selected virtual hierarchy block with the selected layout master that exists on disk.	Remaster
Make Virtual Hierarchy	Integrate layout hierarchies that were realized outside the design.	Make Virtual Hierarchy
Adjust Boundary	Resize or create the area boundary for the selected virtual hierarchy, the PR boundary for the selected soft block, or the top-level PR boundary.	Adjust Boundary
Analyze Connectivity	Draw flight lines between each connected pair of virtual hierarchy groups, soft blocks, or hard blocks. Select pairs of virtual hierarchy, soft block, or hard block to see the connections between.	Analyze Connectivity
Congestion Analysis	Analyze the routing capacity of a design.	Congestion Analysis

Command	**Use to...**	**Form**
Increase Display Depth	Increase the display depth to display the objects deeper in the hierarchy. See Setting the Display Controls.	N/A
Decrease Display Depth	Decrease the display depth to display the objects at the higher levels in the hierarchy. See Setting the Display Controls.	N/A
Set Default Display Depth	Remove all stop level overrides and set the default display depth to `0`. See Setting the Display Controls.	N/A
IO Row Create	Create IO rows for the PAD type siteDefs available in technology.	Create IO Row
IO PAD Placer	Place instances of cell type `PAD` in IO rows.	IO Pad Placement
Pin Tool	Create, resize, plan, and optimize pins, edit pin attributes, and set pin location constraints.	None See Using the Pin Tool
Pin Optimizer for All Pins	Position block pins to minimize the net length honoring process rules and design constraints such as `min-spacing`, `min-width`, `wireType`, `order`, and `edge` at a particular level in the design.	Pin Placement

Command	Use to...	Form
Push Pre-Routes	Push the top-level implementation of power structures and signal routes into the block level, either as route or as a blockage.	Push into Blocks
Remove Block overlaps	Remove overlaps and place overlapping hard blocks and soft blocks adjacent to each other.	None See Remove Block Overlap
Snap Soft Blocks To Grid	Snap soft blocks to placement grid or the manufacturing grid.	None See Snap Soft Blocks to Grid
Pull Soft Blocks inside PR Boundary	Pull the soft blocks overlapping the PR boundary to the core area.	None See Pull Soft Blocks Inside PR Boundary
Edit Soft Blocks	Modify soft block attributes.	Edit Soft Blocks
Adjust Soft Blocks	Adjust the existing floorplan by abutting and pushing the soft blocks in the design.	None See Adjust Blocks

Setting Environment Variables

Environment variables control the values of the Design Planning options. For a list of all the supported environment variables and their values, see List of Design Planning Environment Variables.

There are two ways in which you can set environment variables:

- To set an environment variable that is applied every time you open the Design Planning environment, add the setting to your `.cdsenv` or `.cdsinit` file. For more information, see Setting Environment Variables in a .cdsenv or .cdsinit File.
- To set an environment variable that is applied for the duration of the current session, use the `envSetVal()` command in the CIW. For more information, see Setting Environment Variables in the CIW.

Setting Environment Variables in a .cdsenv or .cdsinit File

To have your environment variables set automatically when you start the Design Planning environment, do one of the following:

- Include the environment variables in the `.cdsenv` file in your home directory; for example,

  ```
  layoutDP autoPlaceOnAreaBoundaryEdit boolean t
  ```

- Include an `envSetVal( )` command in your `.cdsinit` file

  ```
  envSetVal("layoutDP" "autoPlaceOnAreaBoundaryEdit" 'boolean t)
  ```

For more information on the `.cdsenv` and `.cdsinit` files, see Environment Variables in the *Virtuoso Layout Suite XL: Basic Editing User Guide*.

Setting Environment Variables in the CIW

If you use any environment variable values consistently and do not want to set these values each time you use a command, you can set the variables to the value you normally use in the CIW and it will remain valid for the duration of the current session.

To set environment variables for a single session, do one of the following.

- Include `envSetVal( )` in any other Cadence SKILL file you load.
- Type `envSetVal( )` in the CIW.

For example, to set the Design Planning environment variable, `autoPlaceOnAreaBoundaryEdit`, type the following in the CIW or include it in a setup file.

```
envSetVal("layoutDP" "autoPlaceOnAreaBoundaryEdit" 'boolean t)
```

To determine the current value of any Design Planning environment variable, type the following in the CIW.

```
envGetVal("layoutDP" "autoPlaceOnAreaBoundaryEdit"))
```

Some Useful Acronyms and Terms

The table below provides the expansions and descriptions of some acronyms and terms that have been used in this user guide.

Term/Acronym	Expansion	Use/Function

CAS	Check Against Source	Compares the layout with the schematic and reports mismatches
CPH	Configure Physical Hierarchy	Defines how the layout is generated based on the schematic
GFS	Generate All From Source	Generates the layout from the schematic
UCN	Update Components And Nets	Updates the layout based on the schematic modifications

Related Resources

- Design Planning Functions in the *Virtuoso Layout Suite SKILL Reference*.
- Virtuoso Layout Suite XL: Basic Editing User Guide
- Virtuoso Layout Suite XL: Constraint Aware Editing User Guide
- Virtuoso Floorplanner User Guide

2

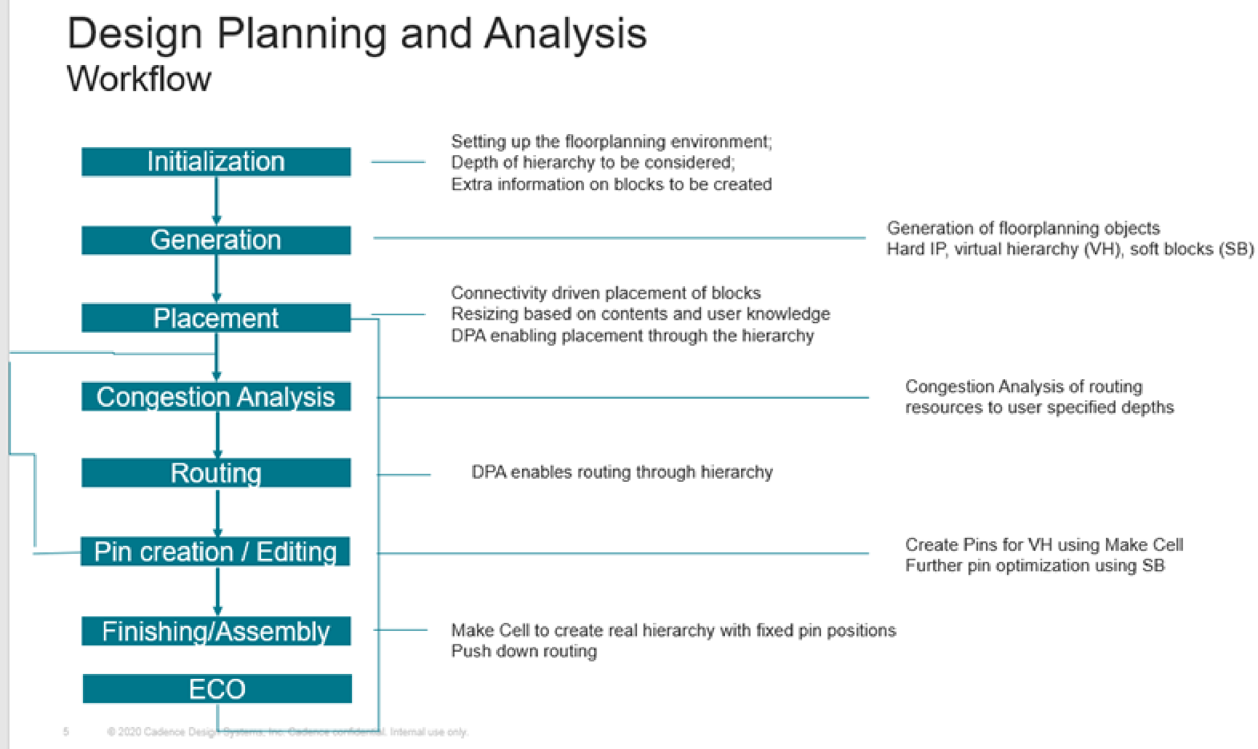
Design Planning and Analysis Flow

Introduction to the Design Planning and Analysis Flow

The Virtuoso® Design Planning and Analysis flow is a schematic-driven flow that offers an innovative solution for designs that have unimplemented layout views existing at the top level and the levels below and where the schematic has a hierarchy and some or all levels of the schematic hierarchy have unimplemented layout views. Usually, for such designs, the placement at the top level is oblivious to the actual lower-level instances, so the top-level placement is often based on estimation. After the design is routed, there might be changes needed to the top level that are often expensive to accommodate.

The Design Planning and Analysis Flow

The basic Design Planing and Analysis flow is illustrated below.



Design Planning and Analysis
Workflow

Initialization
Setting up the floorplanning environment;
Depth of hierarchy to be considered;
Extra information on blocks to be created

Generation
Generation of floorplanning objects
Hard IP, virtual hierarchy (VH), soft blocks (SB)

Placement
Connectivity driven placement of blocks
Resizing based on contents and user knowledge
DPA enabling placement through the hierarchy

Congestion Analysis
Congestion Analysis of routing
resources to user specified depths

Routing
DPA enables routing through hierarchy

Pin creation / Editing
Create Pins for VH using Make Cell
Further pin optimization using SB

Finishing/Assembly
Make Cell to create real hierarchy with fixed pin positions
Push down routing

ECO

5 © 2020 Cadence Design Systems, Inc. Cadence confidential. Internal use only.

In the Design Planning and Analysis flow, you can seamlessly layout, place, and route , backed by real routing data. The Design Planning and Analysis flow uses the existing schematic hierarchy, where available, to create a virtual hierarchy that can be realized, when appropriate. Blocks that have an existing IP in place and soft blocks from an existing Virtuoso Floorplanner flow are both recognized by the Design Planning and Analysis flow as valid inputs for design planning.

Generating a Layout

Use the Generate All From Source toolbar button to generate layout representations of the schematic design components. Alternatively, you can choose the *Plan — Generate — Generate All From Source* menu command.

For the schematic instances that have layout representations available, the *Generate All From Source* command creates corresponding layout views in the canvas.

Depending on the available source components, the Design Planning *Generate All From Source* command appropriately leverages the source to create the corresponding layout representations.

- If a hard block exists, the hard block is recognized and uses as a source for generating the layout view.
- If a soft block exists, the soft block pin information is used to plan the top-cell layout.
- If no hard or soft blocks exist, the schematic hierarchyis used to create a virtual hierarchy that entirely matches the schematic. See Generating a Virtual Hierarchy.
- If the schematic does not exist for a symbol in the schematic hierarchy, the default area estimates and pin information from the cell are used to automatically create a soft block. See Generating Soft Blocks for a Virtual Hierarchy.
- If multiple instances of the virtual hierarchy are needed, the virtual hierarchy is cloned.

For schematic instances that have constraints defined, the constraints are transferred to the generated virtual hierarchy. However, any constraints that exist within a virtual hierarchy are restricted to the virtual hierarchy.

Important

> If there is a missing schematic representation, soft blocks cannot be generated unless the pin information is available in the schematic symbols.

For more information on generating all the layout components in the Virtuoso platform using the source schematic, see Generating All Components From Source in the *Virtuoso Layout Suite XL: Connectivity Driven Editing User Guide*.

Generating Selected Components

Use the Generate Selected From Source toolbar button to generate layout representations of the selected schematic design components. Alternatively, you can choose the *Plan — Generate — Generate Selected From Source* menu command.

When selecting schematic instances to generate a virtual hierarchy, the Design Planning and Analysis tool adds the layout representations corresponding to the selected schematic instance to the drag set and encloses them in an area boundary. If multiple instances of the selected schematic instance need to be created, the Design Planning and Analysis tool clones the instances and adds them to the drag set. The generated virtual hierarchy can be accessed using the *DESIGN PLANNING–Virtual Hierarchy Clones* data set in the Navigator assistant.

The option to generate a virtual hierarchy using the selected schematic instances is available only when using the *As In Schematic* placement mode. For more information on generating layout instances as in the schematic, see Generating Components As In Schematic in the *Virtuoso Layout Suite XL: Connectivity Driven Editing User Guide*.

Generating a Virtual Hierarchy

Virtual layout hierarchy could be defined as the “yet-to-be-realized” layout hierarchy, which is entirely based on the schematic hierarchy. The virtual layout hierarchy is a *virtual* representation of not only the top-cell objects, but also the objects at the lower levels, generating a completely flat hierarchy at the top-cell level.

For the parts of the schematic hierarchy that are complete, you can use the Generate All From Source command to generate a virtual hierarchy. If a Modgen constraint is set on the top-level schematic being used for virtual hierarchy generation, the Modgen is created at the top level. If the constraint is specified at a lower level, the Modgen is created inside the virtual hierarchy.

The generated virtual hierarchy can easily be accessed using the new Virtual Hierarchy data set in the Navigator assistant. If multiple instances of a virtual hierarchy need to be generated, the DPA tool creates clones of the hierarchy. The DPA tool attempts to create the maximum number of clones, as possible, for the design situation. The generated clones can be accessed using the Virtual Hierarchy Clones data subset, which is available in the Navigator assistant for the designs that have virtual hierarchy clones generated.

Note: After a virtual hierarchy is generated and the layout design saved, the design can only be opened using Virtuoso® Layout Suite EXL.

For more information on generating a virtual hierarchy, see Generating a Virtual Hierarchy.

Generating Soft Blocks

When the schematic components in a design have missing layout representations, the Design Planning and Analysis tool generates a virtual hierarchy at the top level that represents the hierarchy in the layout. However, when the schematic counterparts themselves are missing, the DPA tool can create soft blocks for such schematic representations during virtual hierarchy generation. For more information see, Generating Soft Blocks for a Virtual Hierarchy.

Planning a Design

Physical design planning can be a time-consuming task when:

- **Using a flat flow** because the number of components that a layout designer may end up working with can leave the method impractical.
- **Using a hierarchical top-down or bottom-up flow** because
  - ❑ the layout cells are created very early in the design life cycle and are often placed based on estimates rather than real congestion data, causing rework.
  - ❑ the design planning often involves several place and route iterations until a feasible layout design is achieved.

The Design Planning and Analysis tool offers a more reliable paradigm for efficient layout planning at the top, block, and cell level, which involves creation of a virtual layout hierarchy that combines the advantages of both a flat and a hierarchical layout design. Working with virtual hierarchies allows you to edit a flat layout as you would edit a hierarchical layout. You can *realize* a virtual hierarchy or *virtualize* a real cellview at any time and any number of times to efficiently plan your layout design.

For optimal planning of virtual hierarchies in a layout design, the DPA tool supports the following Virtuoso Layout Suite commands:

- *Generate All From Source*

  Generates layout counterparts for soft and hard blocks. Generates a virtual hierarchy with a selectable area boundary for schematic instances that have no layout representations available. Generates soft blocks for schematic components that are missing.

  For more information, see Generating Layout, Generating a Virtual Hierarchy, and Generating Soft Blocks for a Virtual Hierarchy.

- *Generate Selected From Source*

  Generates a virtual hierarchy with a selectable area boundary for the selected schematic instances that have no layout representations available.

  For more information, see Generating Selected Components and Generating Components As In Schematic in the *Virtuoso Layout Suite XL: Connectivity Driven Editing User Guide*.

- *Edit In Place*

  Allows editing in place the generated virtual hierarchy clones. To edit virtual hierarchies that are not clone instances, you can edit the virtual hierarchies by adjusting the display depth to make them transparent.

  If you are in the *Design Planning* workspace, and Edit In Place into a virtual hierarchy or descend edit into the schematic, the layout workspace automatically switches to the default Virtuoso Layout EXL workspace. Upon return from edit mode, the workspace automatically switches back to the *Design Planning* workspace.

For more information on the command, see Editing In Place and Generating Transparent Layout Hierarchy.

- *Check Against Source*

  Checks and reports mismatches in connectivity, instance terminal mismatches, and so on, between the schematic and layout views for the virtual hierarchies selected at the top level or for the ones available on *Edit In Place*.

  For more information on the command, see Check Against Source.

- *Update Components And Nets*

  Updates the components from the source and if the selected set that has missing instances contains a virtual hierarchy group, the *Update Components And Nets* command generates the missing components inside the virtual hierarchy group. In addition, the command preserves the area boundary of the instances inside the virtual hierarchy group and generates soft blocks for schematic components that are missing.

  For more information, see Update Components And Nets.

- *Generate Chained Devices*

  Generates device chains for instances inside a virtual hierarchy when the selection is made at the top level.

  For more information on automatic device chaining, see Generate Chained Devices.

- *Interactive Chaining*

  Supports chaining of devices inside a virtual hierarchy when the device selection is made from the top level.

  For more information, see Chaining Devices Interactively.

- *Interactive Abutment*

  Supports abutment of devices inside a virtual hierarchy when the devices are selected from the top level.

  For more information, see Device Abutment.

- *Interactive Folding*

  Supports folding of devices inside a virtual hierarchy when the device selection is made from the top level.

For more information, see Transistor Folding.

In addition, you can use the following commands that are supported by the DPA tool specifically for working with virtual hierarchies.

- *Create Virtual Group*

  Creates a virtual hierarchy group that does not match the schematic hierarchy. The virtual hierarchy group is manually created and has an area boundary around the selected instances at the top level. If the selected instances are ungrouped, the area boundary enclosing the instances is deleted.

  The benefit of creating a virtual group is that it allows you to get an opaque or a transparent visualization of the design hierarchy.

  If you set the placement status of the generated virtual group to *none*, stretching the boundary of the virtual group will automatically place inside the stretched boundary. You can create a virtual group inside a virtual hierarchy to retain existing placements of certain design parts, such as a resistor bank, if those original placements are good,

  For more information on the command, see Create Virtual Group.

- *Adjust Boundary*

  Creates or resizes the area boundary of the selected virtual hierarchy or the PR boundary of the selected soft block. The command uses an enclosure or utilization area estimate to resize the boundary. Alternatively, you can stretch the boundary of a virtual hierarchy or a soft block to resize it.

  For more information on the command, see the Adjust Boundary form, `lxHiAdjustBoundary` SKILL function, and Adjusting the Area Boundary.

  To automatically place virtual hierarchy clones during area boundary adjustment, you must be in *Edit In Place* mode. For more information, see autoPlaceOnAreaBoundaryEdit.

  The area boundary of a virtual hierarchy can also resize *automatically* to accommodate any instances that are moved outside the virtual hierarchy. See Moving Instances Outside a Virtual Hierarchy.

  Adding new instances to an existing virtual group that was created using the *Create Virtual Group* command in the Design Planning toolbar also causes the area boundary of the updated virtual group to be automatically adjusted. See Adding Instances to a Virtual Group.

- *Make Virtual Hierarchy*

Supports concurrent layout designing by creating virtual hierarchies from real cellviews that were realized outside the top-cell view. The *Make* Virtual Hierarchy command automatically sets a *Force Descend* in the physConfig view for the integrated cellviews so that the existing cellviews are no longer picked up.

During Make Virtual Hierarchy, virtual pins are created in the integrated virtual hierarchy in place of pins in the real cellview. The shape of virtual hierarchy pins can be preserved so that the pins can be recreated, if the virtual hierarchy is replaced by a made cell again. If you choose not to preserve the pin shapes, the virtual hierarchy pins are deleted during the Make Virtual Hierarchy operation.

Note:

- ❑ The Make Virtual Hierarchy command canbe used to integrate multipleselected cells at a time, also supporting m-factored devices. Black box soft blocks are not supported but softMacro soft blocks are supported. For more information on the command, see Make Virtual Hierarchy.
- ❑ The *Make Virtual Hierarchy* command always generates new clones instead of adding members to existing clone families.
- ❑ *Make Cell* creates real cellviews from the selected virtual hierarchies. When you run the command hierarchically, it uses the bottom-up approach to also create a layout view for each level of the virtual hierarchy inside the selected virtual hierarchy.

If the made cellview has no area boundary, the command also generates a PR Boundary for the made cells. The made cellview has pins created for the interface nets that connect outside the virtual hierarchy being replaced by a new cellview. You can choose to invoke the congestion-aware global router to create pins on the boundary, create pins on the boundary of the virtual hierarchy ensuring the shortest possible net length, or create pins below the boundary and manually place them, as appropriate.

If routing exists, pins are created where the routes cross the boundary. If no routing exists or pin shapes are not preserved, pins are created below or on the boundary, depending on the pin creation option selected. Pins are also created below the boundary for lower-level cells when the Make Cell command is run hierarchically. See makeCellPinsBelow.

If the made virtual hierarchy was previously integrated with pins set to be preserved, the preserved pin shapes are reused to create the pins. The *Make Cell* command automatically updates the physConfig view to set the newly made cellview as new and available for use. You can choose to make cell using only the single selected clone or all virtual hierarchy clones in one go.

If a virtual hierarchy contains a Modgen, the constraint is copied to the new cellview and the Modgen created. For more information on Modgens, see Virtuoso Module Generator User Guide.

Note:

Although some differences between the schematic and layout hierarchies are ignored during a *Make Cell* run, the command cannot be supported for the following cases:

- ❑ The selected virtual hierarchy does not contain any bound instances.
- ❑ The bound schematic instance is not in the same hierarchy as the schematic instance corresponding to the selected virtual hierarchy.
- ❑ No schematic instance exists that matches the selected virtual hierarchy name, or the schematic master does not match the master of the bound schematic instance.

To overcome these limitations, run the Update Components And Nets command on the selected virtual hierarchy with the *Generate Virtual Hierarchy* option selected. If no bound instances are available, running the Update Components And Nets command may not help.

For more information on the supported *Make Cell* options, see Make Cell.

- *Remaster*

  Replaces the selected virtual hierarchy with the selected layout master that exists on disk. You can choose to replace all the clones of the selected virtual hierarchy with the selected layout variant.

  Although some differences between the schematic and layout hierarchies are acceptable, the Remaster command cannot be supported for the following cases:

  - ❑ The selected virtual hierarchy does not contain any bound instances.
  - ❑ The bound schematic instance is not in the same hierarchy as the schematic instance corresponding to the selected virtual hierarchy.
  - ❑ No schematic instance exists that matches the selected virtual hierarchy name, or the schematic master does not match the master of the bound schematic instance.

  To overcome these limitations, run the Update Components And Nets command on the selected virtual hierarchy with the *Generate Virtual Hierarchy* option selected. If no bound instances are available, running the Update Components And Nets command may not help.

  For more information on the supported *Remaster* options, see Remaster.

Placing a Design

The DPA tool supports both assisted and manual placement of virtual hierarchies.

When a virtual hierarchy is generated, the DPA tool creates an area boundary around each top level virtual hierarchy, which is driven by the size of the layout views present inside the hierarchy. Because the area boundary size is calculated based on the actual size of the objects instead of being an estimated size, it allows for more realistic placement of the blocks at the top level.

You can stretch, resize, or chop the area boundary, if needed, to change the aspect ratio of the virtual hierarchy and manually place the virtual hierarchy at the top level. The manual placement is supported to enable optimization of the floorplan across the hierarchical levels.

For more information on any of the editing commands that can be used to modify the area boundary of a virtual hierarchy, see Editing Objects in the *Virtuoso Layout Suite XL: Basic Editing User Guide*.

Alternatively, you can choose to run the in-built automatic placer to perform a “like-schematic” or an analog or custom placement, as required. For more information on the automatic placement support in the DPA tool, see Auto Placement in the *Virtuoso Placer User Guide.*

The DPA tool also enables you to use the in-built *Analyze Connectivity* tool to get a visual display of the connectivity profile of the virtual hierarchies in a design. When you select a virtual hierarchy at any level, the *Analyze Connectivity* command visualizes the connectivity from the selected hierarchical level and displays flight lines on the layout canvas to represent the cross-hierarchical connections.

With the visual connectivity aid to support you in your placement of the virtual hierarchies at the top level, you can make more informed decisions when placing the top-level objects manually. For more information on the visual connectivity analysis support in the DPA tool, see Analyze Connectivity in the *Virtuoso Floorplanner User Guide*.

Routing a Design

Creation of routes is required for the interface nets of a virtual hierarchy to allow pins to be created on the boundary during Make Cell. The DPA tool supports an in-built global router to automatically route all or selected nets. You can also route selected critical nets manually. For more information on the in-built global routing support in the DPA tool, see Global Route in the *Virtuoso Space-based Router User Guide*.

Note: You can change the virtual hierarchy display depth to allow the global router to see inside and route to internal components. For components that are opaque to the router, the router stops at the boundary.

To perform a detailed analysis of the generated routes to determine the routing capacity of the design, you can use the in-built congestion analysis tool. Having direct access to the congestion analysis tool makes it possible to have genuine and real routing congestion analysis information available upfront, which allows for better optimization of the top-level placement. The DPA tool supports seamless transitions across the generate-place-route phases, until an optimum layout plan is created.

For more information on the in-built congestion analysis support in the DPA tool, see Running Congestion Analysis in the *Virtuoso Space-based Router User Guide*.

Creating and Optimizing Pins

Finalizing the Design

Related Topics

Generating All Components From Source

Editing In Place

Check Against Source

Update Components And Nets

Analyze Connectivity

Auto Placement

Global Route

Running Congestion Analysis

3

Working with a Virtual Hierarchy

This chapter explains how you can generate, view, and edit a virtual layout hierarchy.

This chapter covers the following topics:

- Generating a Virtual Hierarchy
- Generating Soft Blocks for a Virtual Hierarchy
- Viewing a Virtual Hierarchy
- Setting the Display Controls
- Viewing Virtual Hierarchy Overrides
- Adjusting the Area Boundary
- Viewing Dynamic Display Measurements for a Virtual Hierarchy
- Level-1 Editing of a Virtual Hierarchy
- Editing a Virtual Hierarchy Using the Context-Sensitive Menu
- Editing a Virtual Hierarchy Clone

Generating a Virtual Hierarchy

To generate a virtual hierarchy:

1. In Layout EXL, invoke the Design Planner toolbar.

   Alternatively, launch the Design Planning Workspace.

   The Design Planning Toolbar displays.

2. Click the *Generate All From Source* ()toolbar button.

   The Generate Layout form displays with the *Virtual Hierarchy* option in the Design Planning group box selected.

3. (*Optional*) Choose the other layout generation options.
4. Click *OK*.

   Design Planner generates the layout components from scratch for the selected schematic source.

   For the schematic instances that have no soft or hard block, the Design Planner generates a virtual hierarchy, which can be accessed using the Navigator assistant. For more information, see Viewing a Virtual Hierarchy.

   For schematics that only have pins, no instances or physical binding, or for symbols in the schematic hierarchical design that have a missing schematic, the Design Planner generates soft blocks when the virtual hierarchy is generated. For more information, see Generating Soft Blocks for a Virtual Hierarchy.

Important

Any top-level virtual hierarchy blocks that have an *ignore for generation* attribute set on the instances do not have their soft blocks created in the virtual hierarchy.

Related Topics

Generating a Virtual Hierarchy

Generating Soft Blocks for a Virtual Hierarchy

Generate Layout

Generating All Components From Source

Generating Soft Blocks for a Virtual Hierarchy

To generate a soft block for a symbol block that has a missing schematic or a schematic that has pins but no instances or physical binding:

1. In Layout EXL, invoke the Design Planner toolbar.

   Alternatively, launch the Design Planning Workspace.

   The Design Planning Toolbar displays.
2. Click the *Generate All From Source* ( ) toolbar button.

The Generate Layout form displays, with the *Virtual Hierarchy* and *Auto Generate Soft Blocks* check boxes already selected. You can use the generateVirtualHierarchy and generateSoftBlocks environment variables to control the default values of the fields.

In the PR Boundary tab, the *Virtual Hierarchy Area Boundary* and the *Soft Block* group boxes are now enabled.

3. In the *Virtual Hierarchy Area Boundary* group box, choose from *Enclose by* or *Utilization (%)* to specify how the area boundary for the virtual hierarchy is created.

   If choosing **Enclose by**, type a value in the adjacent field to specify the distance from the objects inside the virtual hierarchy at which the area boundary is created. Alternatively, set the areaBoundaryEnclosure environment variable.

   If choosing **Utilization (%)**, type a value in the adjacent field to specify the acceptable area utilization percentage for deriving the size of the area boundary for the virtual hierarchy. This is automatically set to the same utilization percentage as the PR Boundary. Alternatively, set the Layout XL initUtilization environment variable.

4. *Choose* the appropriate hierarchy level for which the area boundary settings must be applied.

5. In the *Area* field, type a value to specify the area of the soft blocks to be created.

6. Click *OK*.

   The Design Planner generates the layout components from scratch for the selected schematic source.

   For the top-level virtual hierarchy blocks that have missing schematic or a schematic with only pins but no instances or physical binding, the Design Planner first searches for soft block definitions through the Configure Physical Hierarchy (CPH) window. If relevant definitions are found, matching soft blocks are generated in the layout canvas.

   If relevant definitions are not found in CPH, the Design Planner generates soft blocks and PR boundary based on the corresponding symbol view using the following environment variables:

   - `initIOPinLPP`, `initIOPinWidth`, and `initIOPinHeight` for soft blocks
   - `initAreaUitlization`, `aspectRatio`, and `lxGenerateArea` for PR boundary

   The placement of pins in soft blocks is performed based on the symbol view.

   Note: Placement of pins based on the symbol view results in creation of identical pins in terms of height, width and layer purpose-pair. Placement based on the physical configuration defined in CPH lets you specify a unique value for pins and the PR boundary.

Important

Any top-level virtual hierarchy blocks that have an *ignore for generation* attribute set on the instances do not have their soft blocks created in the virtual hierarchy.

The generated virtual hierarchies and soft blocks can be accessed using the Navigator assistant under the Virtual Hierarchy data set. For more information, see Viewing a Virtual Hierarchy and Accessing Hierarchical Objects.

Related Topics

Generating a Virtual Hierarchy

Generate Layout

Generating All Components From Source

Viewing a Virtual Hierarchy

In Layout EXL, the Navigator assistant supports three additional data sets for displaying the generated virtual hierarchies and related components:

- *Virtual Hierarchy*

  See Accessing Virtual Hierarchies.

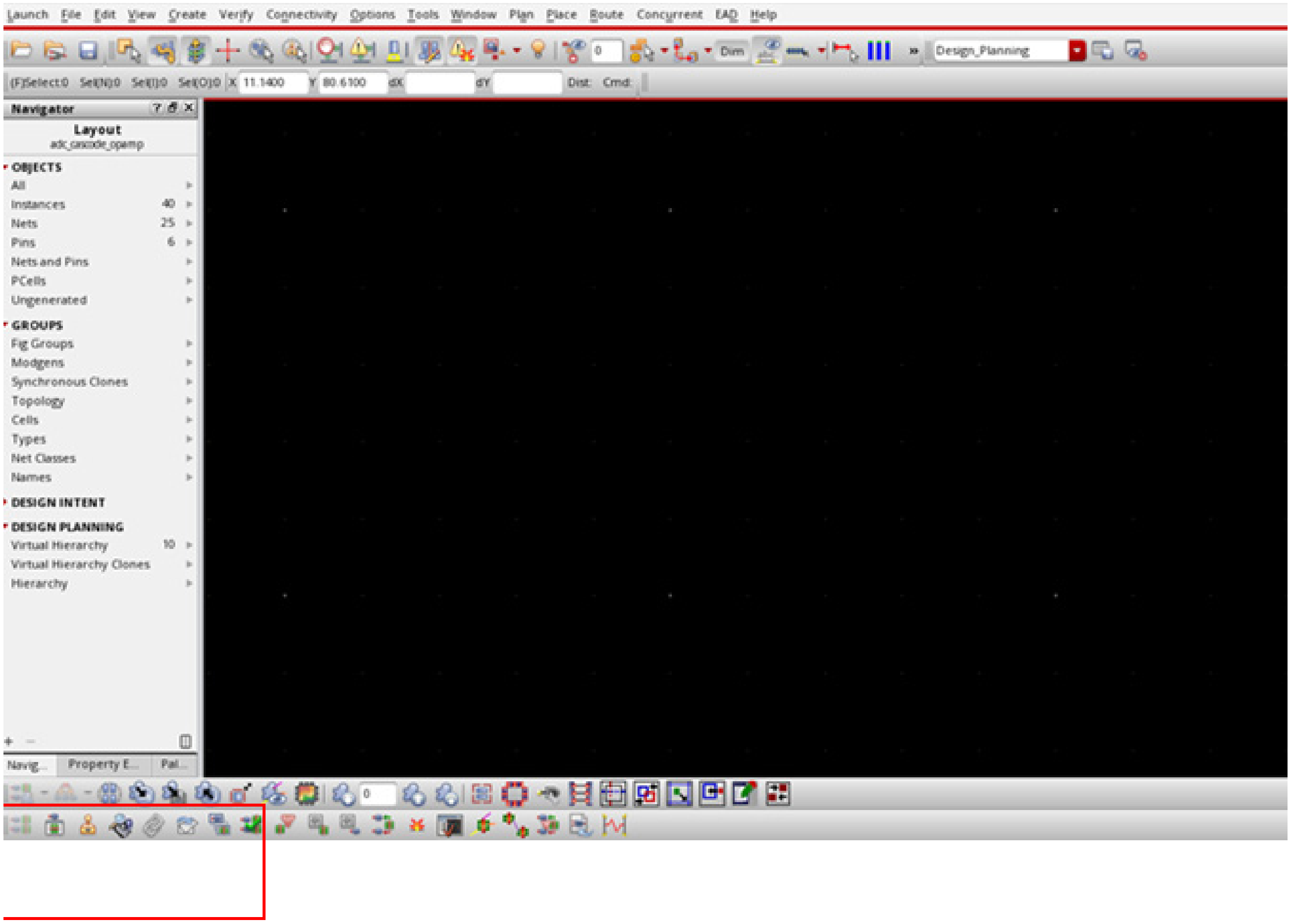
- *Virtual Hierarchy Clones*

  See Accessing Virtual Hierarchy Clones.

- *Hierarchy*

  See Accessing Hierarchical Objects.

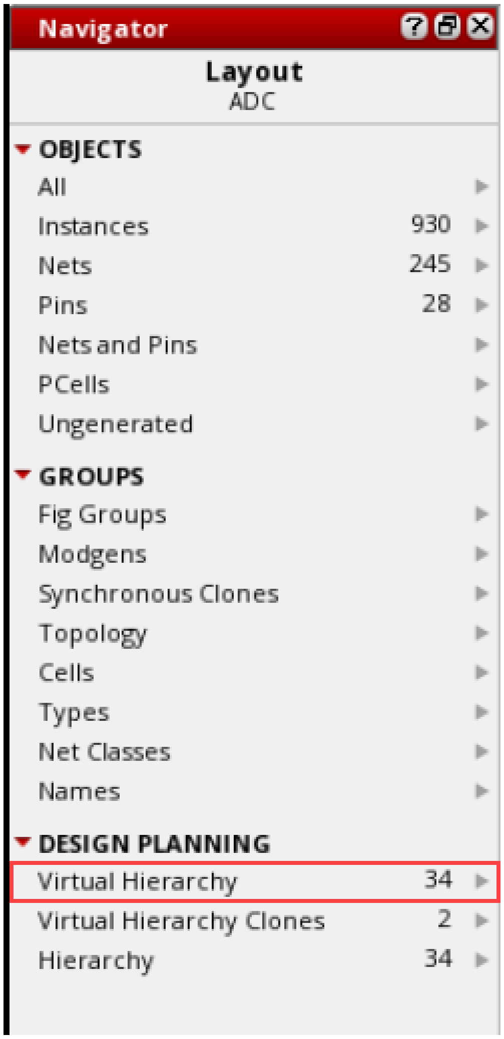
Each of these predefined data sets, as displayed in the figure below, is available under the *DESIGN PLANNING* category.



Accessing Virtual Hierarchies

To access the generated virtual hierarchies and interface nets in the layout canvas:

1. In the Navigator assistant, select the *Virtual Hierarchy* data set.



Navigator

Layout
ADC

▾ **OBJECTS**

All		▸
Instances	930	▸
Nets	245	▸
Pins	28	▸
Nets and Pins		▸
PCells		▸
Ungenerated		▸

▾ **GROUPS**

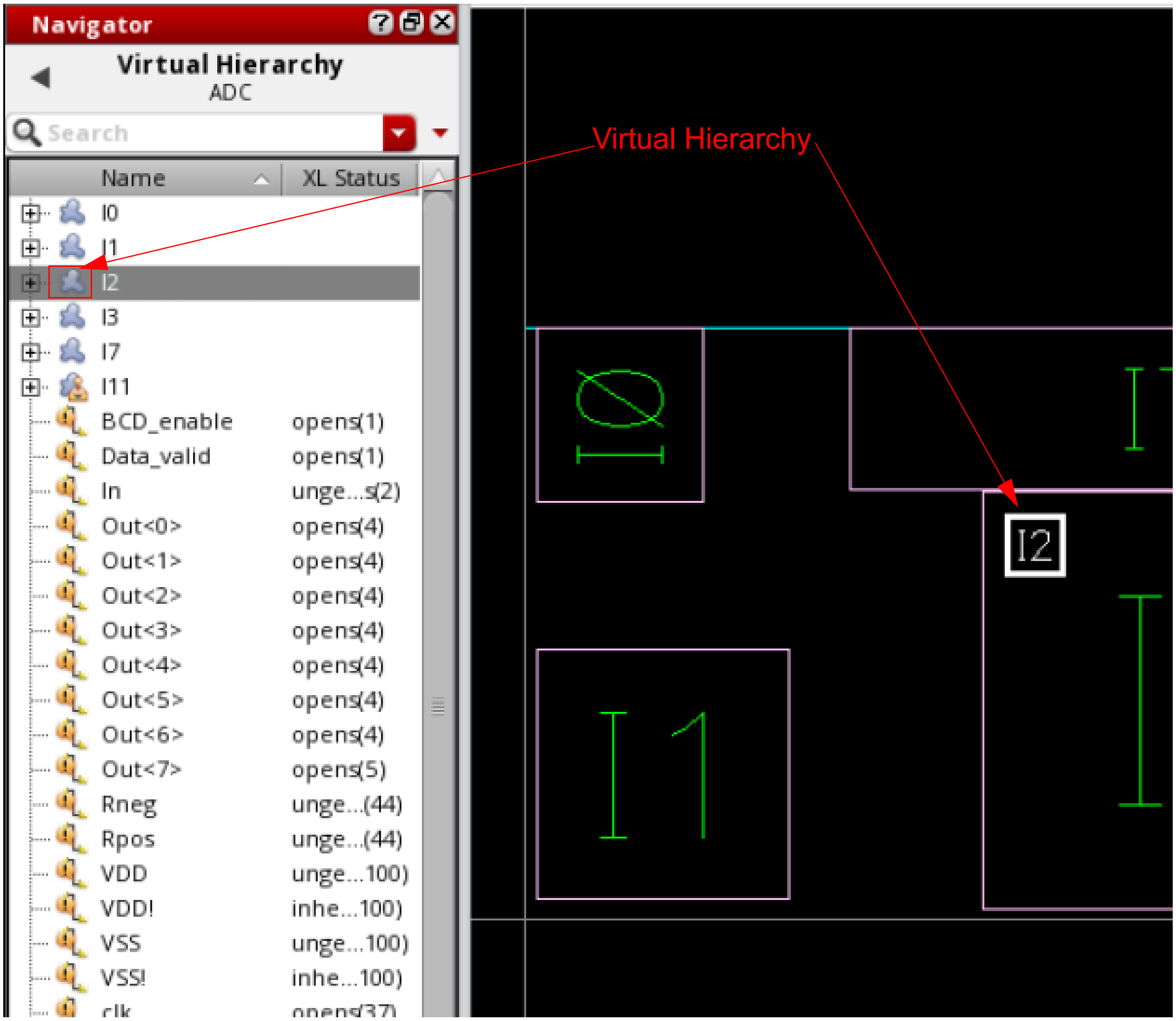
Fig Groups	▸
Modgens	▸
Synchronous Clones	▸
Topology	▸
Cells	▸
Types	▸
Net Classes	▸
Names	▸

▾ **DESIGN PLANNING**

Virtual Hierarchy	34	▸
Virtual Hierarchy Clones	2	▸
Hierarchy	34	▸

The Navigator tree updates to display the available virtual hierarchies and the associated instances and nets. The virtual hierarchy is represented using an amoeba-shaped icon

( ), as displayed in the figure below. If the virtual hierarchy is created as a group, it is represented in the Navigator using the group ( )icon.



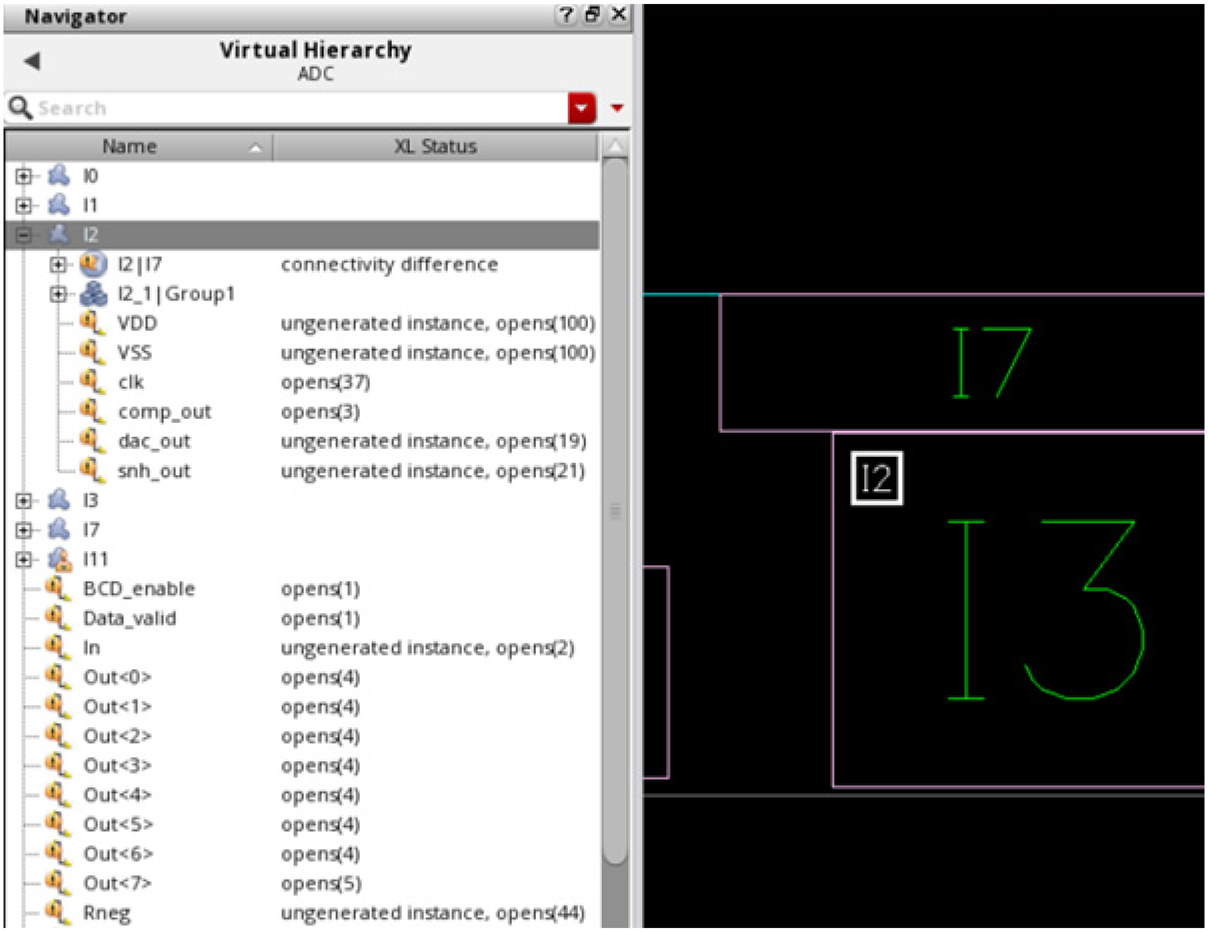
Name	XL Status
I0	
I1	
I2	
I3	
I7	
I11	
BCD_enable	opens(1)
Data_valid	opens(1)
In	unge...s(2)
Out<0>	opens(4)
Out<1>	opens(4)
Out<2>	opens(4)
Out<3>	opens(4)
Out<4>	opens(4)
Out<5>	opens(4)
Out<6>	opens(4)
Out<7>	opens(5)
Rneg	unge...(44)
Rpos	unge...(44)
VDD	unge...100)
VDD!	inhe...100)
VSS	unge...100)
VSS!	inhe...100)
clk	opens(37)

2. (*Optional*) Click a virtual hierarchy in the Navigator tree.

   The corresponding object is selected in the layout canvas. The Show Selection Info toolbar updates to display the following information related to the selected virtual group: display name, type of virtual group—*generated*, *clone*, or *created*, placement status, and the display stop level value. If a virtual pin is selected, the Show Selection Info toolbar updates to display the layer and net name on which the pin is created and the pin width and height.

3. (*Optional*) Click the (+) button adjacent to the virtual hierarchy icon in the Navigator tree to view the objects inside the virtual hierarchy.

The selected virtual hierarchy is expanded and the objects inside the hierarchy are listed in the Navigator tree, as displayed in the figure below. In addition, the Navigator assistant displays the XL Status of the virtual hierarchy components.

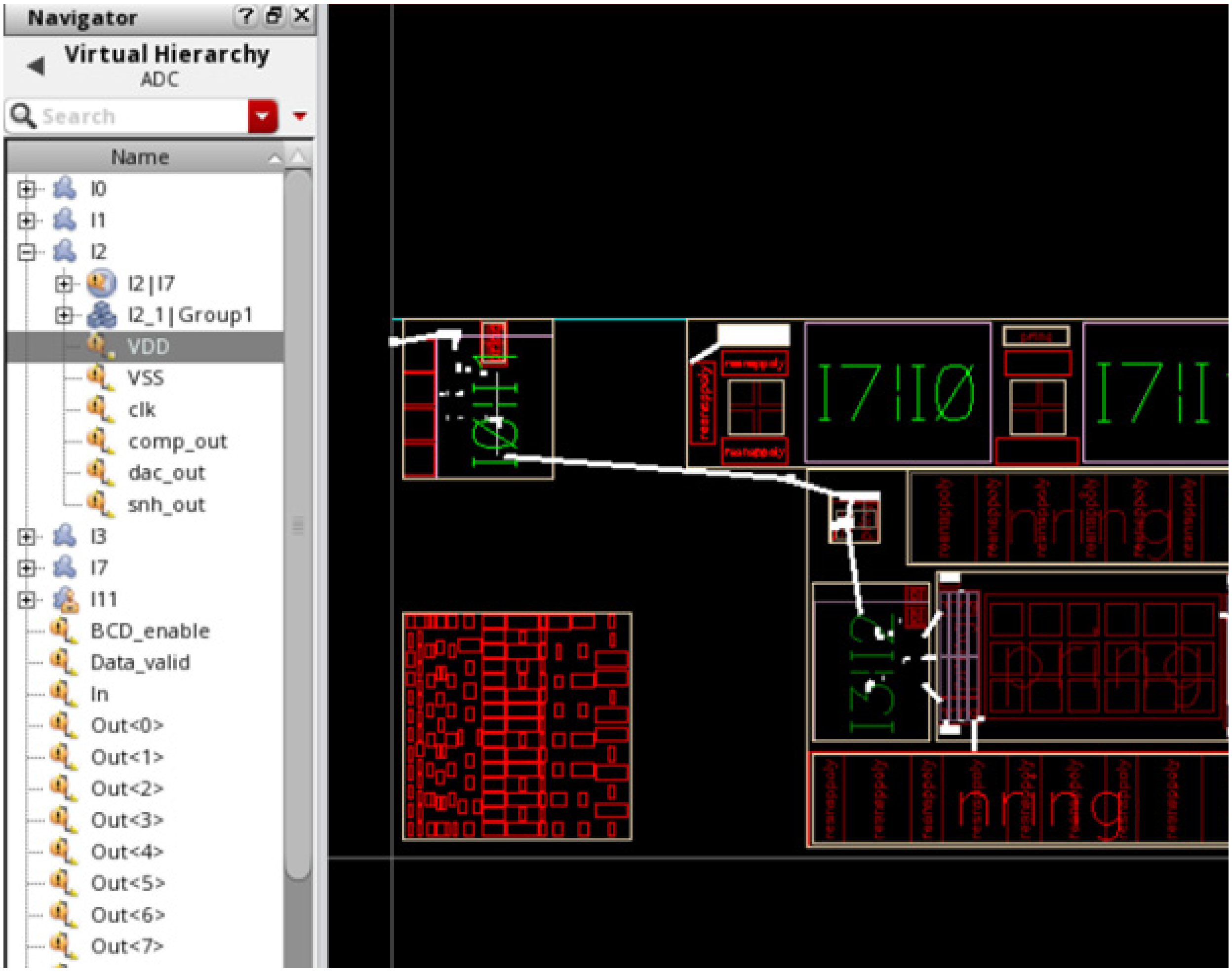


Name	XL Status
I0	
I1	
I2	
I2|I7	connectivity difference
I2_1|Group1	
VDD	ungenerated instance, opens(100)
VSS	ungenerated instance, opens(100)
clk	opens(37)
comp_out	opens(3)
dac_out	ungenerated instance, opens(19)
snh_out	ungenerated instance, opens(21)
I3	
I7	
I11	
BCD_enable	opens(1)
Data_valid	opens(1)
In	ungenerated instance, opens(2)
Out<0>	opens(4)
Out<1>	opens(4)
Out<2>	opens(4)
Out<3>	opens(4)
Out<4>	opens(4)
Out<5>	opens(4)
Out<6>	opens(4)
Out<7>	opens(5)
Rneg	ungenerated instance, opens(44)

Note: A virtual hierarchy can also contain an instance (or more) of another virtual hierarchy as one of the components. Moving such a virtual hierarchy around the layout canvas will display flight lines on the canvas. However, if the virtual hierarchy you move is opaque, which means it has no contents, no flight line will be generated to the virtual hierarchy.

4. (*Optional*) Click a virtual hierarchy component to view the corresponding layout representation on the canvas. If you select an interface net associated with a virtual

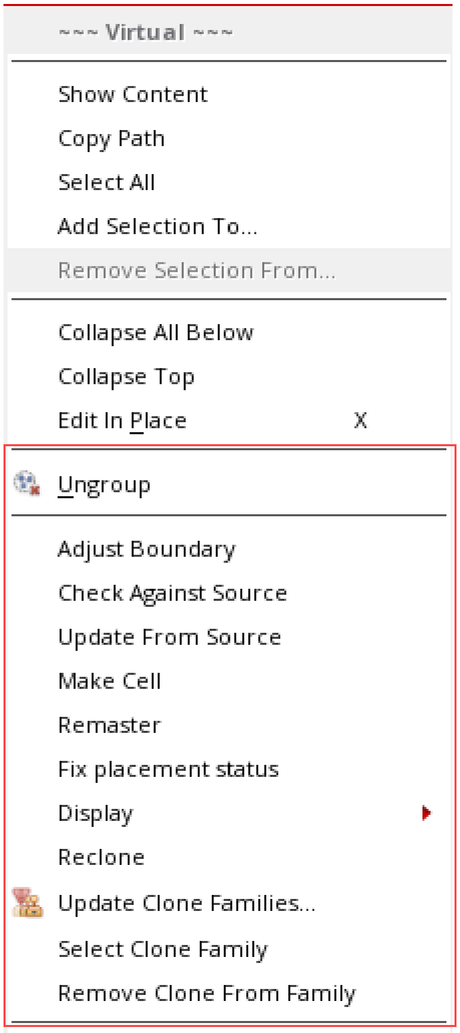
hierarchy in the Navigator tree, Layout EXL creates a probe highlighting all the associated instances and nets in the layout canvas, as displayed in the figure below.



If you cross-select the schematic representation of a virtual hierarchy, the boundary of the virtual hierarchy is highlighted in the layout canvas. If the cross-selected virtual hierarchy is opaque, which means it has no contents, no highlights are created in the layout canvas.

5. (*Optional*) Right-click a virtual hierarchy in the Navigator tree to perform any of the operations supported through the context menu.

For more information on the context menus, see Editing a Virtual Hierarchy Using the Context-Sensitive Menu.



Related Topics

Generating a Virtual Hierarchy

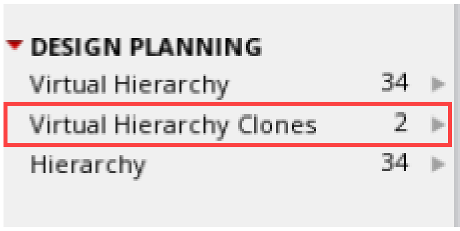
Accessing Virtual Hierarchy Clones

Accessing Hierarchical Objects

Accessing Virtual Hierarchy Clones

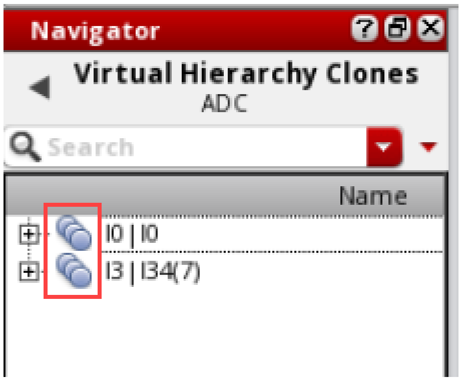
The virtual hierarchy clones generated in the layout are categorized under the *Virtual Hierarchy Clones* data set in the Navigator assistant.

1. In the Navigator assistant, select the *Virtual Hierarchy Clones* data set.



DESIGN PLANNING
Virtual Hierarchy 34
Virtual Hierarchy Clones 2
Hierarchy 34

The Navigator tree updates to display the available virtual hierarchy clone families. The virtual hierarchy clone family is represented using the ( ) icon in the Navigator assistant, as displayed in the figure below. The corresponding virtual hierarchy clones are represented using the ( ) icon.



2. (*Optional*) Click the (+) button adjacent to the virtual hierarchy clone family icon in the Navigator tree to view the corresponding clone family members.
3. (*Optional*) Click the (+) button adjacent to the virtual hierarchy clone icon in the Navigator tree to view the objects inside the virtual hierarchy clone.
4. (*Optional*) Right-click a virtual hierarchy clone in the Navigator tree to perform any of the operations supported through the context menu.

   A virtual hierarchy clone supports three additional commands in the context-sensitive menu:

   - *Update Clone Families*
   - *Select Clone Family*
   - *Remove Clone From Family*

For more information on the context menus, see Editing a Virtual Hierarchy Using the Context-Sensitive Menu.

Related Topics

Generating a Virtual Hierarchy

Viewing a Virtual Hierarchy

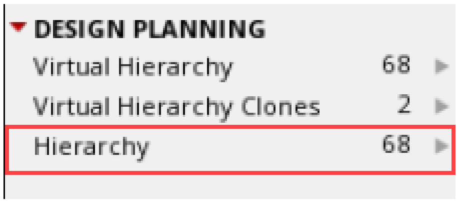
Accessing Hierarchical Objects

Accessing Hierarchical Objects

The hierarchical objects associated with the generated virtual hierarchy are categorized under the *Hierarchy* data set in the Navigator assistant.

Selecting the data set updates the Navigator tree to display the hierarchical instances such as instances, interface nets, soft blocks, sub-Modgens, clones, and so on, associated with the generated virtual hierarchies. In addition, the data set displays virtual hierarchies, if any, existing in the hierarchy of a virtual hierarchy. The *XL Status* of the hierarchical objects is also displayed. For soft blocks, the information balloon of the instance displays the message that the selected instance is a soft block.

Note: By design, the hierarchical display of a virtual hierarchy in the Navigator assistant is restricted to avoid complexity.



Related Topics

Generating a Virtual Hierarchy

Viewing a Virtual Hierarchy

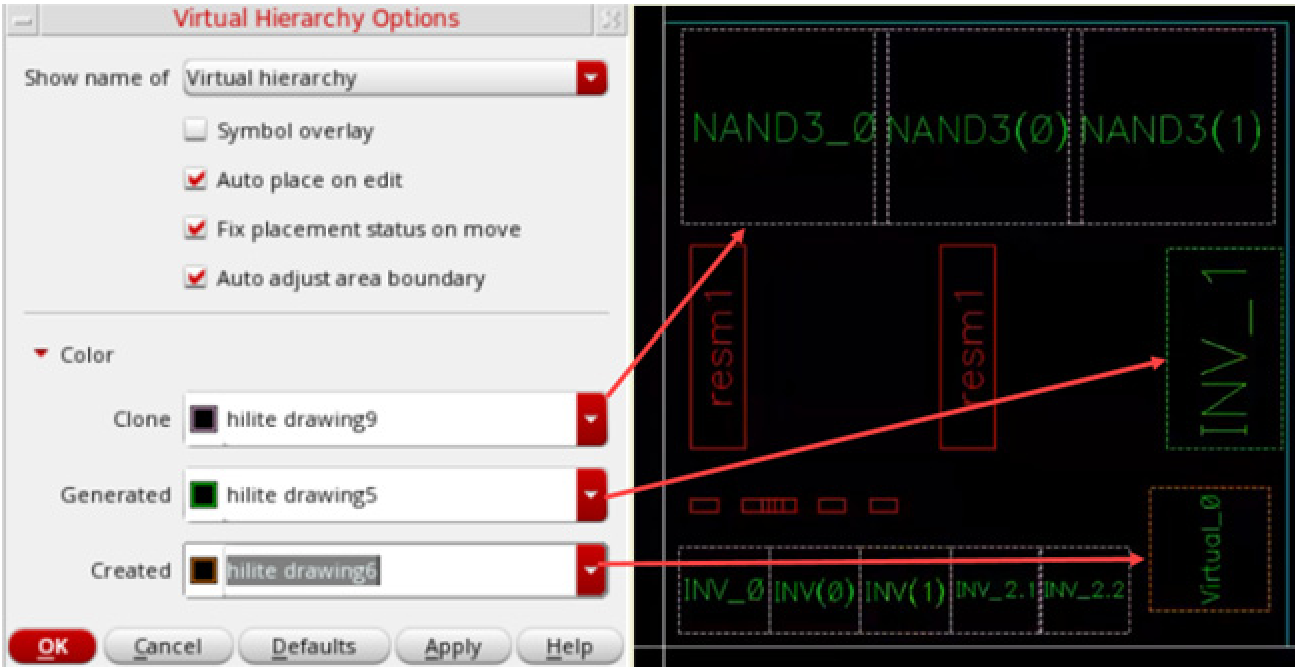
Accessing Virtual Hierarchy Clones

Navigator

Viewing the Virtual Hierarchy Types

Virtual hierarchies can be *created*, *cloned*, or *generated*. This means a layout in EXL can have three different types of virtual hierarchies—depending on how the virtual hierarchies were formed.

To support ease of identification, the bounding box of the virtual hierarchies in the layout can be color-coded differently using the *Color* options in the Design Planning and Analysis Options form. You can display the virtual hierarchy bounding box using the default color for each type or select a color of your choice. The text labels on each virtual hierarchy use the same color, irrespective of the virtual hierarchy type they belong to.



Virtual Hierarchy Options

Show name of: Virtual hierarchy

☐ Symbol overlay

☑ Auto place on edit

☑ Fix placement status on move

☑ Auto adjust area boundary

▾ Color

Clone: hilite drawing9

Generated: hilite drawing5

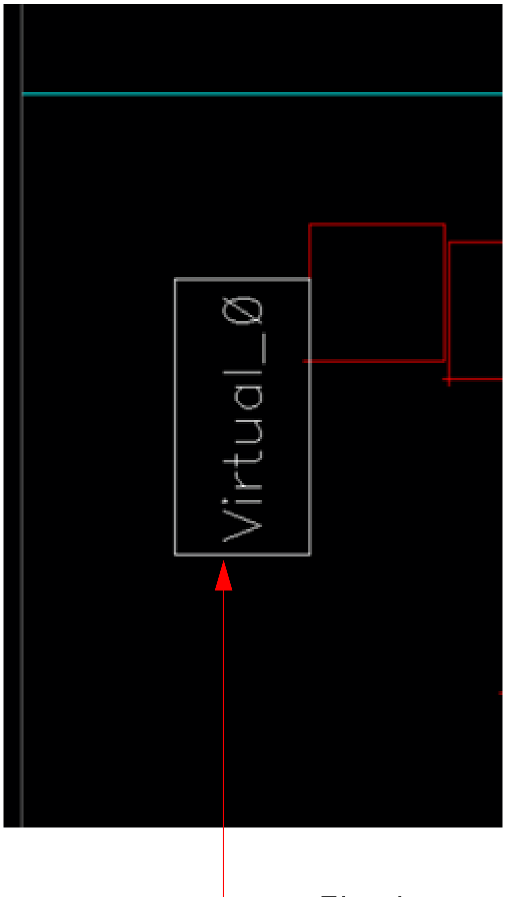
Created: hilite drawing6

Viewing the Virtual Hierarchy Placement Status

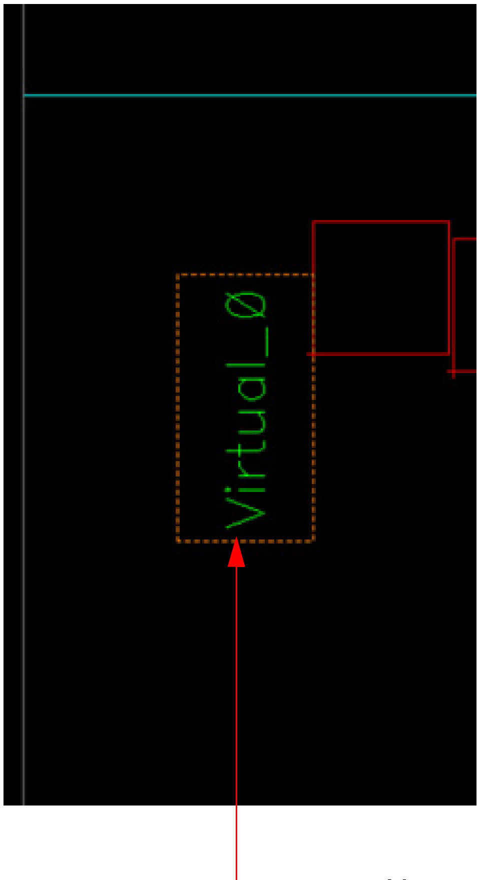
To know whether a virtual hierarchy has its placement fixed or not, you can check its bounding box.

- If the bounding box of a virtual hierarchy is a solid line, as shown in the figure below on the left, the placement status of the virtual hierarchy is *Fixed*.

- If the bounding box of a virtual hierarchy is a dashed line, as shown in the figure below on the right, the placement status of the virtual hierarchy is *None*.



VH Placement status: *Fixed*



VH Placement status: *None*

Setting the Display Controls

By default, when a virtual hierarchy is generated, the Design Planner sets its display depth to `0` to hide the contents within. This helps reduce the clutter at the top level, which is otherwise possible, given that a virtual hierarchy is flat at the top level.

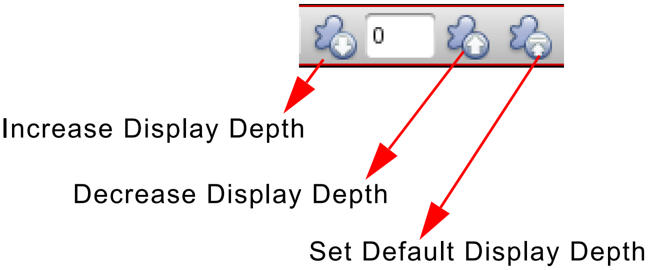
Because the top-cell layout of a virtual hierarchy is flat, setting the display control enables you to show connectivity to the lower-levels instances or the virtual hierarchy at a lower level, which is not possible when working with real cellviews. This access to the connectivity information for lower-level cells allows you to make context-sensitive connections that are useful for creating efficient design plans.

You can choose to display (or hide) the virtual hierarchies at different levels, depending on the task being completed. The Design Planner enables you to control the display level at a global level, although it also honors the display overrides that you may have set on a specific virtual hierarchy. For more information on viewing the individual overrides set on a virtual hierarchy, see Viewing Virtual Hierarchy Overrides.

Design Planner provides two options for you to enable display controls for better layout design planning:

- **Display Depth Options**

  The Design Planner toolbar provides the following three display control options.

  - *Increase Display Depth*

    Increases the display depth of all the selected virtual hierarchies to display the objects at a level deeper in the hierarchy than the current display level. The maximum display depth level is `32`.

  - *Decrease Display Depth*

    Decreases the display depth of all the selected virtual hierarchies to display the objects at a level higher in the hierarchy than the current display level. The minimum display depth level is `0`.

  - *Set Default Display Depth*

    Removes all the stop level overrides, sets the default display depth to `0`, and deselects all the objects that were previously selected.

  Note: Alternatively, you can use bindkeys to control the display depth.
  Use `Ctrl`, `Shift+` to increment and `Ctrl-` to decrement the display depth.

- **Analyze Connectivity Command**

  Draws flight lines between connected virtual hierarchy figGroups. You can select one of the connected virtual hierarchy figGroups to view the connections to the other associated virtual hierarchy figGroup.

See Analyze Connectivity.

Setting Display Depth

Design Planner enables you to control the display depth globally for all the components in the design, or only for the selected components. You can choose to navigate and display the design.

To increase (or decrease) the display depth for all the components in the design:

➡ In the Design Planner toolbar, select the *Increase Display Depth* toolbar button*.*

Select the *Decrease Display Depth* toolbar button, if the display depth needs to be decreased.

The display depth of all the components is incremented (or decreased) by one level and the layout canvas also displays the components that are at the next level.

To increase (or decrease) the display depth for a virtual hierarchy:

1. In the layout canvas or the Navigator, select the virtual hierarchy for which the display depth needs to be increased (or decreased).
2. In the Design Planner toolbar, select the virtual hierarchy *Increase Display Depth* toolbar button*.*

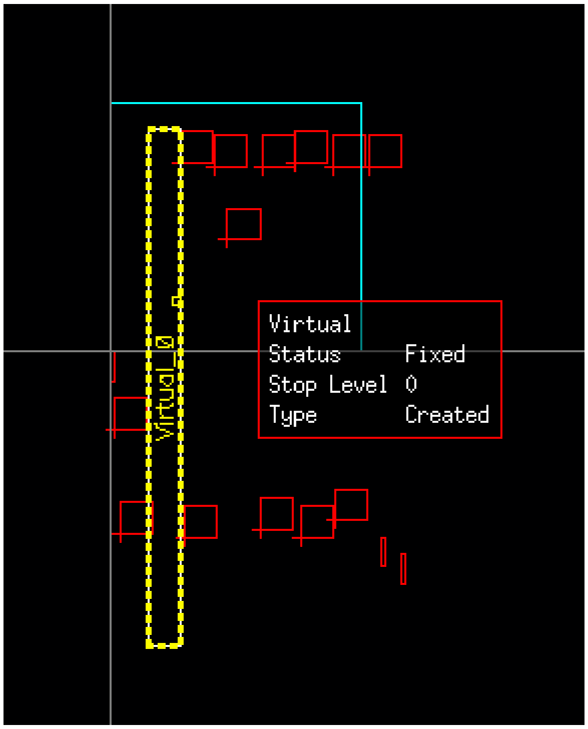
   Select the *Decrease Display Depth* toolbar button, if the display depth needs to be decreased.

   Alternatively, right-click the virtual hierarchy in the Navigator assistant or the layout canvas and choose *Display – Increment* to increase the display depth or *Display – Decrement* to decrease the display depth.

   The display depth of the selected virtual hierarchy is incremented (or decreased) by one level and the layout canvas also displays the components that are at the next level.

Viewing Virtual Hierarchy Overrides

For designs that have individual virtual hierarchies set to different stop levels, types, placement status, and so on, you can view the individual virtual hierarchy overrides using the information balloon. You can customize the information balloon to selectively display measurements such as *Placement Status, Stop Level*, and *Type*, as shown in the figure below, when the cursor is hovered over a virtual hierarchy.



```
Virtual
Status      Fixed
Stop Level  0
Type        Created
```

For more information on viewing the information balloons and other dynamic measurement displays for virtual hierarchies, see Viewing Dynamic Display Measurements for a Virtual Hierarchy.

Related Topics

Display Options

Analyze Connectivity

Navigator

Adjusting the Area Boundary

The virtual hierarchies you create using the Design Planner can have a rectangular or a rectilinear area boundary. You can choose the appropriate boundary type to be created using the Adjust Boundary form, and the Design Planner automatically *creates* or *adjusts* the area boundary, as specified.

Alternatively, you can register an area estimation function to specify the area boundary of a virtual hierarchy. See Registering a User-Defined Area Estimation Function.

Registering a User-Defined Area Estimation Function

Adjusting the Area Boundary of a Virtual Hierarchy

To adjust the area boundary of a virtual hierarchy, you can register an area estimator function as below.

```
procedure(myFunction(figGroupId @optional (contactArea  1.0)
  prog((area insts)
    // Function body to calculate area involving virtual hierarchies
insts = setof(x figGroupId~>members x~>objType == "inst")
       foreach(ins insts
           area = area + ins~>master~>prBoundary~>area * contactArea
       )
       return(area)
    )
)
```

You can register the function in `deferred` mode, as shown below.

```
leRegAreaEstimator(
    'myFunction
        "simpleAreaEst"
        '('group)
        '((contactArea 1.0))
        "deferred"
        )
```

Where `deferred` means that the function is applied only when all the required information is available. In this example, the virtual hierarchies are considered for area estimation only when the *Generate All From Source* command is run.

Note: If you register the PR boundary area estimation function in `deferred` mode and then choose to generate virtual hierarchies with area boundaries, the PR boundary estimation is deferred until the area boundary of the virtual hierarchy is created. This allows the area estimation function to use the virtual hierarchy area boundary values to determine the area for the PR boundary.

Alternatively, you can register the function in `direct` mode, as shown below.

```
leRegAreaEstimator(
    'myFunction
        "simpleAreaEst"
        '('group)
        '((contactArea 1.0))
        "direct"
        )
```

Where `direct` means the function is applied when you choose the *PR Boundary based* or *Bbox based* area estimators on the Adjust Boundary form.

Important

`"direct"` mode is not supported by the *Generate All From Source* command.

Generating a Custom Virtual Hierarchy Block Size

To generate a virtual hierarchy of custom block size during a *Generate All From Source* or an *Update Components And Nets* run, you can register an area estimation function as below:

```
procedure(myFunction(figGroupId))
let(((area 0.0))

area

list(area)


list(area 'util f_util) ; Overrides the default Utilization % value specified on
the form. Accepts both integer and float values such as 25 or 25.0.


list(area 'aspect f_aspect) ; Overrides the default Aspect ratio value specified
on the form. Accepts both integer and float values such as - 1 or 1.0.


list(area 'util f_util 'apsect f_aspect) ; Overrides both the Utilization % and
Aspect ratio values specified on the form.


list(area 'width f_width) ; Overrides the Width value specified on the form. Accepts
only float values.


list(area 'height f_height) ; Overrides the Height value specified on the form.
Accepts only float values.
```

Alternatively, you can use the custom area generation function with a utilization function, `f_util`, as shown below:

```
(area 'util f_util 'width f_width)

list(area 'width f_width 'util f_util)

list(area 'util f_util 'height f_height)

list(area 'height f_height 'util f_util)
```

To specify only the width and height values to generate a custom virtual hierarchy block, use the function as below:

```
list('width f_width 'height f_height)
list('hight f_height 'width f_width)
```

Note: If you specify the custom area function using an incorrect syntax but specify the correct original area value, the original area value will be used.

Adjusting the PR boundary of a Soft Block

To adjust the PR boundary of a soft block, you can register an area estimator function as below.

```
procedure(myFunction(cellViewID @optional (contactArea  1.0)
  prog(((area 0.0))
    // Function body to calculate area involving soft blocks
insts = setof(x figGroupId~>members x~>objType == "inst")
       foreach(ins cellView~>insts
           area = area + ins~>master~>prBoundary~>area * contactArea
       )
       return(area)
    )
)
```

You can register the function in `deferred` mode, as shown below.

```
leRegAreaEstimator(
    'myFunction
        "simpleAreaEst"
        '('cellView)
        '((contactArea 1.0))
        "deferred"
        )
```

Where `deferred` means that the function is applied only when all the required information is available. In this example, the soft blocks are considered for area estimation only when the *Generate All From Source* command is run.

Alternatively, you can register the function in `direct` mode, as shown below.

```
leRegAreaEstimator(
    'myFunction
        "simpleAreaEst"
        '('cellView)
        '((contactArea 1.0))
        "direct"
        )
```

Where `direct` means the function is applied when you choose the *PR Boundary based* or *Bbox based* area estimators on the Adjust Boundary form.

Important

`"direct"` mode is not supported by the *Generate All From Source* command.

For more information on registering an area estimation function, see leRegAreaEstimator in the *Virtuoso Layout Suite SKILL Functions Reference.*

Automatic Adjustment of an Area Boundary

The Design Planner also automatically adjusts the area boundary of a virtual hierarchy when:

- Moving Instances Outside a Virtual Hierarchy
- Adding Instances to a Virtual Group

Note: The resized boundary is always rectangular when automatically adjusted.

Moving Instances Outside a Virtual Hierarchy

The Design Planner can automatically resize the area boundary of a virtual hierarchy to accommodate an instance (or instances and figGroups other than row region) that are moved *outside* the boundary. If adjusting the area boundary of the virtual hierarchy causes its area boundary to go outside the higher-level virtual hierarchy that includes the adjusted virtual hierarchy, then the boundary of the higher-level virtual hierarchy is also automatically adjusted. To allow this automatic area boundary adjustment of virtual hierarchies across the hierarchy, select the *Auto adjust area boundary* option on the Design Planning and Analysis Options form.

Note: The Design Planner automatically resizes the area boundary of a virtual hierarchy when the instances or figGroups are moved, or the area boundary of a virtual hierarchy is manually stretched outside the boundary. If the instances are moved back inside the virtual hierarchy, the area boundary needs to be adjusted manually.

To automatically adjust area boundary when moving an instance outside a virtual hierarchy:

1. In the Design Planner toolbar, click the *Increase Display Depth* button to increase the display depth stop level value to a value greater than 0.
2. In the Virtual Hierarchy Options form, select the *Auto adjust area boundary* option and click *OK*.
3. In the layout canvas, select an instance inside a virtual hierarchy and drag it to a position outside the area boundary of the virtual hierarchy.

   Flight lines appear on the canvas, showing instance connections to other objects still inside the virtual hierarchy.
4. Drop the selected instance outside the area boundary of the virtual hierarchy.

The area boundary of the virtual hierarchy automatically resizes to fully accommodate the moved objects.

Note:

- ❑ If the area boundary of the adjusted virtual hierarchy now falls outside the area boundary of the higher-level virtual hierarchy, the area boundary of the higher-level virtual hierarchy is also automatically adjusted.
- ❑ Rectilinear boundaries that are resized are automatically adjusted to form rectangles. To prevent the automatic adjustment of the area boundary, you can set the autoAdjustBoundary environment variable to `nil`.

Adding Instances to a Virtual Group

The Design Planner allows you to add new instances to an existing virtual group created using the Create Virtual Group command. To accommodate the new instances, the area boundary of the updated virtual group is automatically adjusted.

To add a new instance to a virtual group:

1. In the layout canvas or Navigator assistant, select the virtual hierarchy to be updated.
2. Choose the *Edit – Group – Add to Group* option.

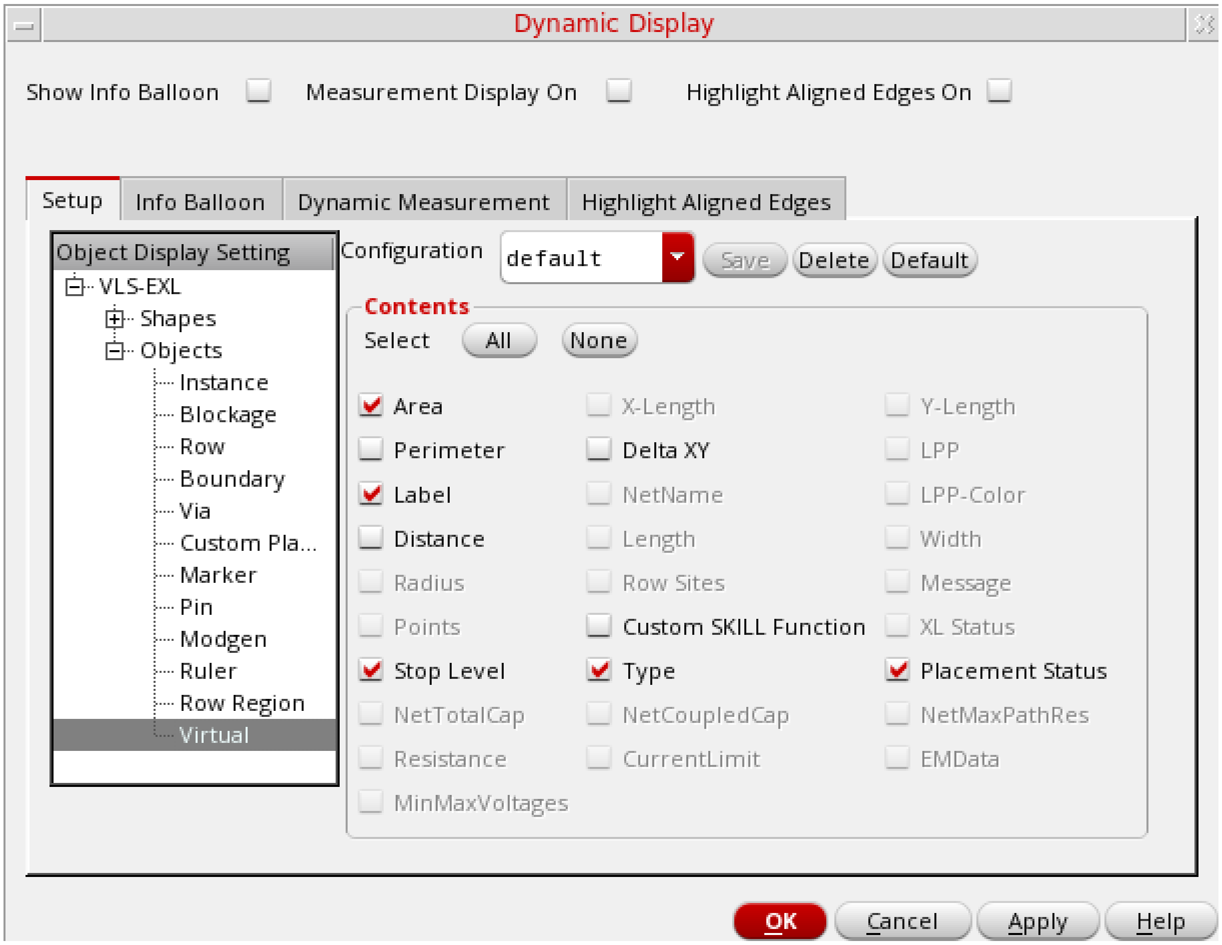
   The *Add to Group* command is invoked, as indicated by the trailing ellipsis following the pointer.
3. Click the layout instance to be added to the group.

   The area boundary of the selected virtual group automatically resizes to accommodate the new layout instance. The Navigator assistant updates to reflect the newly added instance as belonging to the selected virtual group.

   Note: Rectilinear boundaries that are resized are automatically adjusted to form rectangles. To prevent the automatic adjustment of the area boundary, you can set the autoAdjustBoundary environment variable to `nil`.

Viewing Dynamic Display Measurements for a Virtual Hierarchy

When manually adjusting the area boundary of a virtual hierarchy, enable the *Measurement Display On* option in the *Options – Dynamic Display* form.



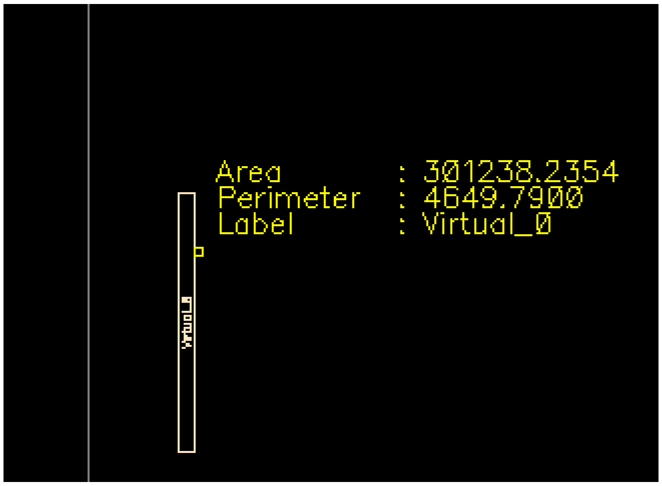
Show Info Balloon □ Measurement Display On □ Highlight Aligned Edges On □

Configuration `default`

☑ Area	□ X-Length	□ Y-Length
□ Perimeter	□ Delta XY	□ LPP
☑ Label	□ NetName	□ LPP-Color
□ Distance	□ Length	□ Width
□ Radius	□ Row Sites	□ Message
□ Points	□ Custom SKILL Function	□ XL Status
☑ Stop Level	☑ Type	☑ Placement Status
□ NetTotalCap	□ NetCoupledCap	□ NetMaxPathRes
□ Resistance	□ CurrentLimit	□ EMData
□ MinMaxVoltages		

This enables dynamic display of the virtual hierarchy measurements, such as `area`, `label`, `stop level`, `type`, `placement status`, and similar others. The measurements appear dynamically on the canvas during an edit, such as when using the *Stretch* command to adjust the area boundary of a virtual hierarchy. The in-context availability of these key measurements enables you to be precise when selecting the area for a virtual hierarchy during a *Stretch* or a *Constant area stretch* operation.

The measurements, as shown in the figure below, are displayed dynamically on the layout canvas as you perform the *Stretch* command.



```
Area       : 301238.2354
Perimeter  : 4649.7900
Label      : Virtual_0
```

You can also choose to display the virtual hierarchy measurements in an information balloon to make them available when the cursor hovers over a virtual hierarchy.

Note: To display the `delta x, y` measurements dynamically, you must also select them for display in the information balloon.

Alternatively, you can define a Custom SKILL function to specify the preferred measurement display.

For more information on dynamic display measurements, see Dynamic Display Form. For more information on the Stretch command, see Stretch Form.

Level-1 Editing of a Virtual Hierarchy

A virtual hierarchy can be automatically *generated* when *Generate All From Source* is run with the *Virtual Hierarchy* option selected. See Generating a Virtual Hierarchy for more information.

If the virtual hierarchy is automatically generated, you can update the contents of the hierarchy at the top level. However, for virtual hierarchies that are manually *created*, level-1 editing is restricted to avoid any unexpected edits to the level-1 objects. For the contents of a *created* virtual group to be selected and for the virtual group to be edited, the transparentGroup environment variable must be set to `true`.

Editing a Virtual Hierarchy Using the Context-Sensitive Menu

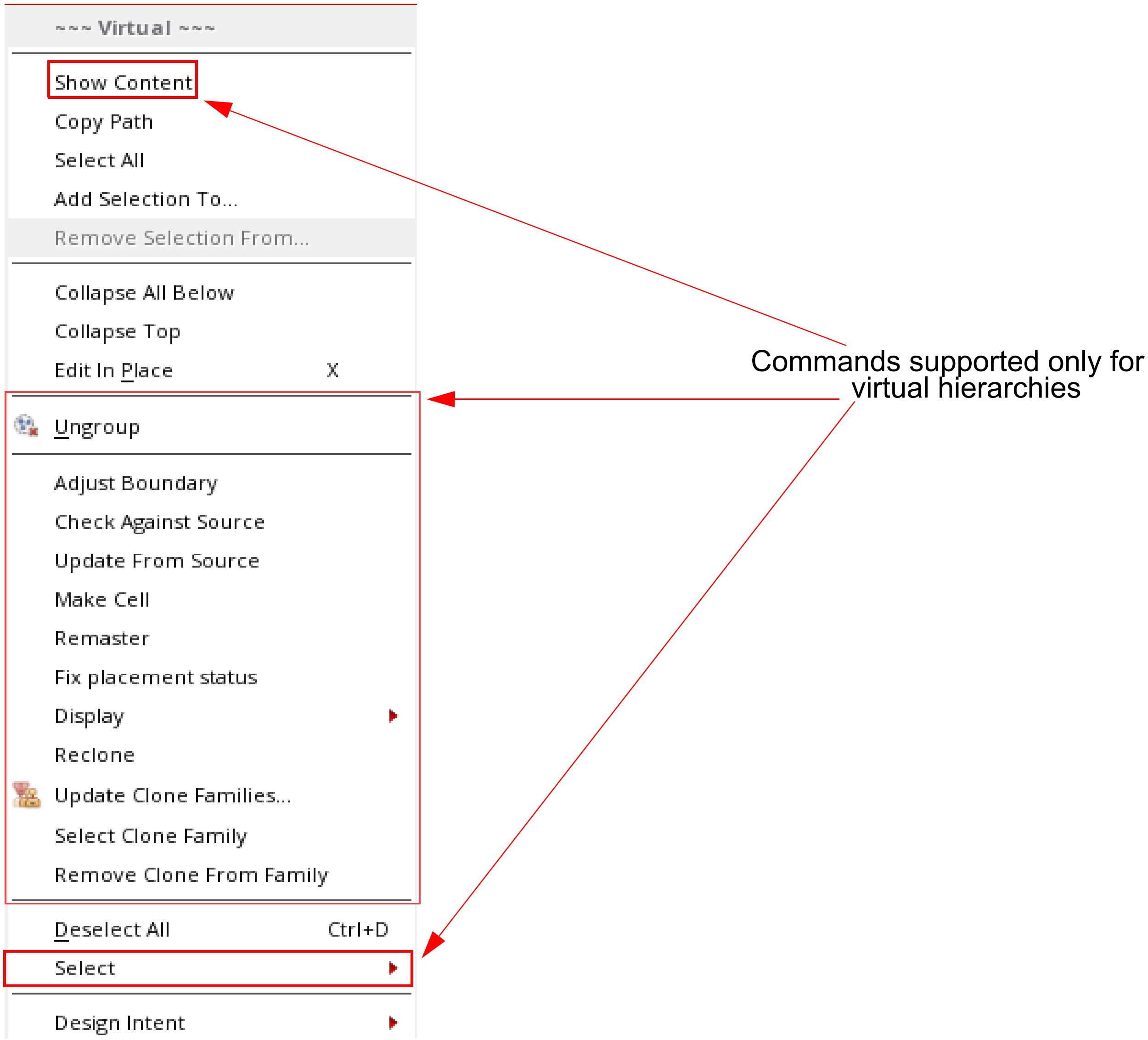
The Design Planner provides quick access to some useful commands through the context-sensitive menu of a virtual hierarchy and a virtual hierarchy clone.

To access the context-sensitive menu of a virtual hierarchy:

➡ Right-click a virtual hierarchy in the Navigator assistant or the layout canvas.

The *Virtual* context-sensitive menu is displayed, as shown in the figure below. The context menu provides some generic commands supported in Layout EXL, in addition to a set of commands that are supported only for virtual hierarchies.

Note: When invoked from the Navigator assistant, the *Virtual* context-sensitive menu displays an additional command—*Show Content*.



Note: The context menu of a virtual hierarchy clone supports a few additional commands. See the context-menu command table below for details.

Context-Sensitive Command Menu Table

The table below details the context-sensitive menu commands that are supported for virtual hierarchies and virtual hierarchy clones.

Command name	Supported for...	Use to...
Show Content	■ Virtual hierarchies ■ Virtual hierarchy clones	Switch the Navigator assistant *Summary* pane view to show the contents inside the selected virtual hierarchy. **Note:** This command is available only when the virtual hierarchy is selected in the Navigator assistant.
Ungroup	■ Virtual hierarchies ■ Virtual hierarchy clones	Delete the area boundary around the virtual hierarchy group and display the components as individual objects at the top level. Any virtual hierarchies that exist inside the ungrouped virtual hierarchy continue to exist as a virtual hierarchy group at the top level.
Adjust Boundary	■ Virtual hierarchies ■ Virtual hierarchy clones	Resize the virtual hierarchy at the top level and create boundaries at any levels. See Adjust Boundary.

Command name	Supported for...	Use to...
Check Against Source	■ Virtual hierarchies ■ Virtual hierarchy clones	Check and report mismatches between the schematic and layout views of the virtual hierarchy currently being edited in place*.* **Note:** Any markers from a previously reported *Check Against Source* run are deleted to ensure that markers related only to the selected virtual hierarchy are reported. See Check Against Source.
Update From Source	■ Virtual hierarchies ■ Virtual hierarchy clones	Updates the components from source for the virtual hierarchies selected at the top level or for the ones available on *Edit In Place* to update any mismatches in the selected set.
Make Cell	■ Virtual hierarchies ■ Virtual hierarchy clones	Creates a new cellview and replaces the virtual hierarchy with an instance of this cellview. See Make Cell and Planning a Design.
Remaster	■ Virtual hierarchies ■ Virtual hierarchy clones	Replace the selected virtual hierarchy with the layout master that exists on disk. *See* Remaster*.*

Command name	**Supported for...**	**Use to...**
Fix placement status	■ Virtual hierarchies ■ Virtual hierarchy clones	Set the *Placement Status* of the selected virtual hierarchy to `Fixed`*.* When toggled to *Unfix placement status*, sets the *Placement Status* of the selected virtual hierarchy to `None`. See Make Virtual Hierarchy and Planning a Design.

Command name	Supported for...	Use to...
Display	■ Virtual hierarchies ■ Virtual hierarchy clones	Choose *Increment* to increase the display depth for all the selected virtual hierarchies. Choose *Decrement* to decrease the display depth for all the selected virtual hierarchies. Choose *Set Default Display Depth* to set the *Default* display depth *to 0* and deselect all the objects that were previously selected. *See Setting the Display Controls.* Choose *Highlight* to highlight the selected virtual group contents on the canvas in the chosen color. Choose *Unhighlight* to remove the color highlights from the selected virtual group contents. **Note:** To remove the highlights from all the highlighted virtual hierarchies, click anywhere in the layout canvas to deselect everything. Then, choose *Unhighlight All Virtual Hierarchy* from the shortcut menu.

Command name	Supported for...	Use to...
Reclone	■ Virtual hierarchy clones	Create virtual hierarchy clones belonging to the same master for the selected, generated virtual hierarchy. If the clone family was broken or any new clones were generated since the clone family was last updated, you can use the *Reclone* command to update the clone family. Select any of the clone sources to reclone all the clone family members belonging to the same master. You can also use the command to add Modgens to an existing virtual hierarchy clone. See Recloning Virtual Hierarchy Clones.
Update Clone Families	■ Virtual hierarchy clones	Update the status of the selected virtual hierarchy clone family, rename the clone family, remove a clone from the family, create and remove clone families, and add a group to an existing family. See Update Clone Families.
Select Clone Family	■ Virtual hierarchy clones	Select all the members of the selected virtual hierarchy clone family. See Selecting a Clone Family.
Remove Clone From Family	■ Virtual hierarchy clones	Remove the selected virtual hierarchy clone from the virtual hierarchy clone family. See Removing a Clone From Family.

Command name	Supported for...	Use to...
Select	■ Local Nets	Select external nets that have virtual pins as local nets for the selected virtual hierarchy.

Editing a Virtual Hierarchy Clone

Similar to editing a top-level virtual hierarchy, you can perform some basic edits on a virtual hierarchy clone or a clone inside a virtual hierarchy without having to edit it in place. These top-level edits are limited to stretching, chopping, and adjusting the area boundary of a virtual hierarchy clone.

Any changes made to the clone area boundary due to stretching or chopping are automatically reflected across all the synchronous clones. This capability of top-level edit support for virtual hierarchy clones ensures ease of working with the clones from the top level.

To stretch a virtual hierarchy clone:

➡ Select the virtual hierarchy clone instance from the top level and choose the *Edit – Stretch* command.

The area boundary of the selected virtual hierarchy clone and all its synchronous clone counterparts is automatically stretched.

To chop the area boundary of a virtual hierarchy clone:

➡ Select the virtual hierarchy clone instance from the top level and choose the *Edit – Basic – Chop* command.

The area boundary of the selected virtual hierarchy clone and all its synchronous clone counterparts is chopped.

Recloning Virtual Hierarchy Clones

Use the *Reclone* command to:

- ❑ Clone virtual hierarchies that belong to the same layout master.
- ❑ Update a clone family to add new members.
- ❑ Add a Modgen constraint to a virtual hierarchy.

When virtual hierarchies belonging to the same master are created at different times, the virtual hierarchies are generated as individual instances.

To clone such virtual hierarchies:

➡ Right-click one of the generated virtual hierarchies and choose *Reclone*.

All the virtual hierarchies that belong to the same master are cloned and added to a common clone family.

The *Reclone* command can also be used to update a clone family to add any new members. In this case, when a clone family already exists, the placement of the target clones is determined based on the selected clone source.

- ❑ If you select a virtual hierarchy that is not a clone but belongs to the same master as the clone family, the clone that has the most clones is used as the clone source.
- ❑ If you select an existing clone from the clone family, that is the one used as the clone source.

You can also use the *Reclone* command to generate Modgens inside a virtual hierarchy.

To add a Modgen to a virtual hierarchy:

1. Right-click a clone in the clone family and choose *Update Clone Families* to remove the clone family.
2. Select one of the virtual hierarchy clone sources and choose *Place – Modgen – Create/Edit Modgen*.

   A Modgen constraint is added to the selected virtual hierarchy.
3. Right-click the generated virtual hierarchy containing the Modgen and choose *Reclone*.

   All the virtual hierarchies in the clone family are recloned, and each of the members displays the Modgen constraint.

Editing Virtual Hierarchy Clones Containing Modgens

In a given clone family, only one clone, called the master clone, can contain real modgens. Other clones containing basic figGroups are called pseudo-modgens. Pseudo-modgens cannot be edited directly. If you attempt to edit a pseudo-modgen, the master clone is edited instead and any edits made to the master clone are synchronized and replicated in all the associated pseudo-modgens.

For a pseudo-modgen to accept the edits, it must exist inside a clone. Otherwise, the pseudo-modgen is treated as a basic figGroup, which cannot be edited using the Modgen editor. To allow the pseudo-modgen to be considered for editing, you must enable the Modgen constraint for the figGroup using the Constraint Manager.

Related Topics

Generating a Virtual Hierarchy

Checking Against Source

Display Options

Generating Synchronous Clones

A

Design Planning Environment Variables

This appendix provides information on the environment variable names, descriptions, and graphical user interface equivalents for Virtuoso® Design Planner.

Note: Only the environment variables documented in this chapter are supported for public use. All other Design Planner environment variables, regardless of their name or prefix, and undocumented aspects of the environment variables that are described below are private and are subject to change at any time.

Related Topics

- Setting Environment Variables
- List of Design Planning Environment Variables

Setting Environment Variables

Environment variables control the values of the Design Planner options.

For information on setting the environment variables, see Setting Environment Variables.

For a list of all the Design Planner environment variables and their values, see List of Design Planning Environment Variables.

List of Design Planning Environment Variables

adjustBoundaryCheckOutside

adjustBoundaryIncludeTop

allAreaBoundaries

areaBoundaryAspectRatio

areaBoundaryEnclosure

areaBoundaryHeight

areaBoundaryMinJogLength

areaBoundaryUtilization

areaBoundaryWidth

autoAdjustBoundary

autoPlaceOnAreaBoundaryEdit

autoPlaceOnPRBoundaryEdit

casDisplayVirtHierMismatch

fixPlacementStatusOnMove

generateAreaBoundaries

generateSoftBlocks

generateSoftBlocksInTargetLibrary

generateVirtualHierarchy

makeCellAllLevels

makeCellDeleteVirtualPins

makeCellOptPins

makeCellPinsBelow

makeCellPushInternalRoutesOnly

makeCellPushRoutesAsBlockages

makeCellType

makeCellVirtualClones

makeVirtualAllInstsSameMaster

makeVirtualPreserveVirtualPins

pinLayerLimit

pinLayerLimitNum

pinOptOneConnectionPerSide

softBlockArea

updateSoftBlocksFromSymbol

useAreaBoundaryUtilization

useBindKeys

verboseApOnVhEdit

vhCloneColor

vhCreatedColor

vhDimming

vhGeneratedColor

vhSelectiveMode

vhSymbolOverlay

adjustBoundaryCheckOutside

```
layoutDP adjustBoundaryCheckOutside boolean { t | nil }
```

Description

Checks if the area boundary of a virtual hierarchy or a soft block is extending outside the parent area boundary.

The default value is `t`.

GUI Equivalent

Command:	*Plan – Block Planning – Adjust Boundary*
Field:	*Adjust hierarchically – if outside* (Adjust Boundary)

Examples

```
envGetVal("layoutDP" "adjustBoundaryCheckOutside")
envSetVal("layoutDP" "adjustBoundaryCheckOutside" 'boolean t)
envSetVal("layoutDP" "adjustBoundaryCheckOutside" 'boolean nil)
```

Related Topics

List of Design Planning Environment Variables

Setting Environment Variables

adjustBoundaryIncludeTop

```
layoutDP adjustBoundaryIncludeTop boolean { t | nil }
```

Description

Adjusts the area boundary of parent virtual hierarchies and soft blocks at all levels in the hierarchy and the PR boundary at the top level. The PR boundary area adjustment is performed after the required area boundary adjustments at lower levels.

The default value is `nil`.

GUI Equivalent

Command:	*Plan – Block Planning – Adjust Boundary*
Field:	*Adjust hierarchically – all levels including top* (Adjust Boundary)

Examples

```
envGetVal("layoutDP" "adjustBoundaryIncludeTop")
envSetVal("layoutDP" "adjustBoundaryIncludeTop" 'boolean t)
envSetVal("layoutDP" "adjustBoundaryIncludeTop" 'boolean nil)
```

Related Topics

List of Design Planning Environment Variables

Setting Environment Variables

allAreaBoundaries

```
layoutDP allAreaBoundaries boolean { t | nil }
```

Description

Controls whether the specified area boundary settings are used for generating the virtual hierarchies at all levels in the hierarchy or only for the top-level virtual hierarchy.

The default is `nil`, which means the specified area boundary settings are used only for generating the virtual hierarchy area boundaries for the top-level virtual hierarchy.

GUI Equivalent

Command:	*Connectivity – Generate Layout – PR Boundary (tab)*
Field:	*All Levels*
	(Generate Layout)
Command:	*Connectivity – Update – Components and Nets – PR Boundary (tab)*
Field:	*All Levels*
	(Update Components And Nets)

Examples

```
envGetVal("layoutDP" "allAreaBoundaries")
envSetVal("layoutDP" "allAreaBoundaries" 'boolean t)
envSetVal("layoutDP" "allAreaBoundaries" 'boolean nil)
```

Related Topics

List of Design Planning Environment Variables

Setting Environment Variables

areaBoundaryAspectRatio

```
layoutDP areaBoundaryAspectRatio float floating_point_number
```

Description

Specifies the width-to-height ratio to be used to determine the size of the rectangular area boundary for the virtual hierarchy.

The default is `1.0`.

GUI Equivalent

Command:	*Adjust Boundary – Rectangle* (Design Planning Toolbar)
Field:	*Aspect ratio (W/H)* (Adjust Boundary)

Examples

```
envGetVal("layoutDP" "areaBoundaryAspectRatio")
envSetVal("layoutDP" "areaBoundaryAspectRatio" 'float 4.5)
```

Related Topics

List of Design Planning Environment Variables

Setting Environment Variables

areaBoundaryEnclosure

`layoutDP areaBoundaryEnclosure float` *`floating_point_number`*

Description

Specifies the minimum distance from the objects inside the virtual hierarchy to the area boundary.

The default is `0.5`.

GUI Equivalent

Command:	*Adjust Boundary* (Design Planning Toolbar)
Field:	*Enclose by* (Adjust Boundary)

Examples

```
envGetVal("layoutDP" "areaBoundaryEnclosure")
envSetVal("layoutDP" "areaBoundaryEnclosure" 'float 0.7)
```

Related Topics

List of Design Planning Environment Variables

Setting Environment Variables

areaBoundaryHeight

```
layoutDP areaBoundaryHeight float floating_point_number
```

Description

Specifies the height of the virtual hierarchy area boundary.

The default is `5.0`.

GUI Equivalent

Command:	*Adjust Boundary* (Design Planning Toolbar)
Field:	*Height*
	(Adjust Boundary)

Examples

```
envGetVal("layoutDP" "areaBoundaryHeight")
envSetVal("layoutDP" "areaBoundaryHeight" 'float 2.5)
```

Related Topics

List of Design Planning Environment Variables

Setting Environment Variables

areaBoundaryMinJogLength

`layoutDP areaBoundaryMinJogLength float` *`floating_point_number`*

Description

Specifies the minimum edge length to be used for creating the polygonal area boundary.

The default is `0.0`.

GUI Equivalent

Command:	*Adjust Boundary – Polygon* (Design Planning Toolbar)
Field:	*Minimum jog length* (Adjust Boundary)

Examples

```
envGetVal("layoutDP" "areaBoundaryMinJogLength")
envSetVal("layoutDP" "areaBoundaryMinJogLength" 'float 3.0)
```

Related Topics

List of Design Planning Environment Variables

Setting Environment Variables

areaBoundaryUtilization

```
layoutDP areaBoundaryUtilization float floating_point_number
```

Description

Specifies the acceptable area utilization percentage for deriving the size of the rectangular area boundary for the virtual hierarchy.

The default is `25` (percent).

GUI Equivalent

Command:	*Adjust Boundary – Rectangle* (Design Planning Toolbar)
Field:	*Estimate area – Utilization*
	Adjust Boundary

Examples

```
envGetVal("layoutDP" "areaBoundaryUtilization")
envSetVal("layoutDP" "areaBoundaryUtilization" 'float 17.0)
```

Related Topics

List of Design Planning Environment Variables

Setting Environment Variables

areaBoundaryWidth

```
layoutDP areaBoundaryWidth float floating_point_number
```

Description

Specifies the width of the virtual hierarchy area boundary.

The default is `5.0`.

GUI Equivalent

Command:	*Adjust Boundary – Rectangle* (Design Planning Toolbar)
Field:	*Width*
	Adjust Boundary

Examples

```
envGetVal("layoutDP" "areaBoundaryWidth")
envSetVal("layoutDP" "areaBoundaryWidth" 'float 17.0)
```

Related Topics

List of Design Planning Environment Variables

Setting Environment Variables

autoPlaceOnAreaBoundaryEdit

```
layoutDP autoPlaceOnAreaBoundaryEdit boolean { t | nil }
```

Description

Controls whether the virtual hierarchies that have their *Placement Status* set to `Fixed` are automatically placed when the area boundary of the virtual hierarchy is modified by using the Adjust Boundary, Stretch, Chop, or Reshape command.

The default is `t`.

GUI Equivalent

Command:	*Plan – Options*
Field:	*Auto place on edit – area boundaries*
	(Design Planning and Analysis Options)

Examples

```
envGetVal("layoutDP" "autoPlaceOnAreaBoundaryEdit")
envSetVal("layoutDP" "autoPlaceOnAreaBoundaryEdit" 'boolean t)
envSetVal("layoutDP" "autoPlaceOnAreaBoundaryEdit" 'boolean nil)
```

Related Topics

List of Design Planning Environment Variables

Setting Environment Variables

autoPlaceOnPRBoundaryEdit

```
layoutDP autoPlaceOnPRBoundaryEdit boolean { t | nil }
```

Description

Controls whether the top-level design is automatically placed when the PR boundary is modified by using the Stretch, Chop, or Reshape command.

The default is `t`.

GUI Equivalent

Command:	*Plan – Options*
Field:	*Auto place on edit – PR boundary*
	(Design Planning and Analysis Options)

Examples

```
envGetVal("layoutDP" "autoPlaceOnPRBoundaryEdit")
envSetVal("layoutDP" "autoPlaceOnPRBoundaryEdit" 'boolean t)
envSetVal("layoutDP" "autoPlaceOnPRBoundaryEdit" 'boolean nil)
```

Related Topics

List of Design Planning Environment Variables

Setting Environment Variables

autoAdjustBoundary

```
layoutDP autoAdjustBoundary boolean { t | nil }
```

Description

Automatically resizes the boundary of the virtual hierarchy when:

1. Any instances and figGroups other than row regions are moved outside the virtual hierarchy area boundary.
2. The area boundary of a virtual hierarchy or the PR boundary of a soft block inside the virtual hierarchy is stretched.

Note: The environment variable always creates a rectangular boundary with enclosure.

The default is `t`.

GUI Equivalent

Command:	*Plan – Options*
Field:	*Auto adjust area boundary*
	(Design Planning and Analysis Options)

Examples

```
envGetVal("layoutDP" "autoAdjustBoundary")
envSetVal("layoutDP" "autoAdjustBoundary" 'boolean t)
envSetVal("layoutDP" "autoAdjustBoundary" 'boolean nil)
```

Related Topics

List of Design Planning Environment Variables

Setting Environment Variables

casDisplayVirtHierMismatch

```
layoutDP casDisplayVirtHierMismatch boolean { t | nil }
```

Description

Controls whether the Layout XL *Check Against Source* command checks for and reports mismatches between the virtual hierarchy in the layout and the schematic hierarchy.

The default is `nil`.

GUI Equivalent

Command:	*Connectivity – Check – Against Source*
Field:	*Virtual hierarchy* (Check Against Source)

Examples

```
envGetVal("layoutDP" "casDisplayVirtHierMismatch")
envSetVal("layoutDP" "casDisplayVirtHierMismatch" 'boolean t)
envSetVal("layoutDP" "casDisplayVirtHierMismatch" 'boolean nil)
```

Related Topics

List of Design Planning Environment Variables

Setting Environment Variables

fixPlacementStatusOnMove

```
layoutDP fixPlacementStatusOnMove boolean { t | nil }
```

Description

Sets the *Placement Status* of the virtual hierarchy to `Fixed` when an instance is moved, stretched, rotated, or flipped inside the virtual hierarchy.

The default is `t`, which means the *Placement Status* is set to `Fixed`.

When set to `nil`, unfixes the *Placement Status* of the virtual hierarchy, setting its value to `None`.

GUI Equivalent

Command:	*Plan – Options*
Field:	*Fix placement status on move*
	(Design Planning and Analysis Options)

Examples

```
envGetVal("layoutDP" "fixPlacementStatusOnMove")
envSetVal("layoutDP" "fixPlacementStatusOnMove" 'boolean t)
envSetVal("layoutDP" "fixPlacementStatusOnMove" 'boolean nil)
```

Related Topics

List of Design Planning Environment Variables

Setting Environment Variables

generateAreaBoundaries

```
layoutDP generateAreaBoundaries boolean { t | nil }
```

Description

Generates virtual hierarchies without an area boundary.

The default is `t`.

GUI Equivalent

Command:	*Connectivity – Generate – All From Source* *(PR Boundary tab)*
Field:	*Virtual Hierarchy Area Boundary – None* (Generate Layout)

Examples

```
envGetVal("layoutDP" "generateAreaBoundaries")
envSetVal("layoutDP" "generateAreaBoundaries" 'boolean t)
envSetVal("layoutDP" "generateAreaBoundaries" 'boolean nil)
```

Related Topics

Generate Layout

List of Design Planning Environment Variables

Setting Environment Variables

generateSoftBlocks

```
layoutDP generateSoftBlocks boolean { t | nil }
```

Description

Controls whether the Layout XL *Generate All From Source* and *Update Components and Nets* commands generate soft blocks for schematic symbols that have a missing schematic.

The default is `t`.

GUI Equivalent

Command:	*Connectivity – Generate – All From Source*
Field:	*Auto Generate Soft Blocks* (Generate Layout)

Examples

```
envGetVal("layoutDP" "generateSoftBlocks")
envSetVal("layoutDP" "generateSoftBlocks" 'boolean t)
envSetVal("layoutDP" "generateSoftBlocks" 'boolean nil)
```

Related Topics

Generate Layout

List of Design Planning Environment Variables

Setting Environment Variables

generateSoftBlocksInTargetLibrary

```
layoutDP generateSoftBlocksInTargetLibrary boolean { t | nil }
```

Description

Controls soft block layout creation for read-only libraries.

The default is `nil`, which means soft blocks are created in the same library as the top cellview, if the CPH physical binding for the soft block is read-only.

When set to `t`, no soft blocks are created if the CPH physical binding for the soft block is read-only.

GUI Equivalent

None

Examples

```
envGetVal("layoutDP" "generateSoftBlocksInTargetLibrary")
envSetVal("layoutDP" "generateSoftBlocksInTargetLibrary" 'boolean t)
envSetVal("layoutDP" "generateSoftBlocksInTargetLibrary" 'boolean nil)
```

Related Topics

Generate Layout

List of Design Planning Environment Variables

Setting Environment Variables

generateVirtualHierarchy

```
layoutDP generateVirtualHierarchy boolean { t | nil }
```

Description

Generates a virtual hierarchy using the area boundary options selected in the PR Boundary Tab of the Generate Layout form, or updates an existing virtual hierarchy to match the schematic hierarchy. When updating an existing virtual hierarchy, the area boundary of the virtual hierarchy is not updated.

The default is `nil`.

GUI Equivalent

Command:	*Connectivity – Generate – All From Source*
Field:	*Virtual hierarchy*
	(Generate Layout)

Examples

```
envGetVal("layoutDP" "generateVirtualHierarchy")
envSetVal("layoutDP" "generateVirtualHierarchy" 'boolean t)
envSetVal("layoutDP" "generateVirtualHierarchy" 'boolean nil)
```

Related Topics

List of Design Planning Environment Variables

Setting Environment Variables

makeCellDeleteVirtualPins

```
layoutDP makeCellDeleteVirtualPins boolean { t | nil }
```

Description

Deletes virtual pins during a *Make Cell* run to allow new pins to be generated on or below boundary, as appropriate.

The default is `nil`.

GUI Equivalent

Command:	*Plan – Manage Hierarchy – Make Cell*
Field:	*Create pins – Delete virtual pins*
	(Make Cell)

Examples

```
envGetVal("layoutDP" "makeCellDeleteVirtualPins")
envSetVal("layoutDP" "makeCellDeleteVirtualPins" 'boolean t)
envSetVal("layoutDP" "makeCellDeleteVirtualPins" 'boolean nil)
```

Related Topics

Global Route

Congestion Analysis Assistant

List of Design Planning Environment Variables

Setting Environment Variables

makeCellOptPins

```
layoutDP makeCellOptPins boolean { t | nil }
```

Description

Controls whether the global router is run to perform congestion-aware pin creation for the new made cell. The pins are created on the boundary of the virtual hierarchy selected for made cell creation.

The default is `t`.

GUI Equivalent

Command:	*Plan – Manage Hierarchy – Make Cell*
Field:	*Create pins – Congestion aware*
	(Make Cell)
Command:	*Make Cell* (Design Planning Toolbar)
Field:	*Create pins – Congestion aware*
	(Make Cell)

Examples

```
envGetVal("layoutDP" "makeCellOptPins")
envSetVal("layoutDP" "makeCellOptPins" 'boolean t)
envSetVal("layoutDP" "makeCellOptPins" 'boolean nil)
```

Related Topics

Global Route

Congestion Analysis Assistant

List of Design Planning Environment Variables

Setting Environment Variables

makeCellPinsBelow

```
layoutDP makeCellPinsBelow boolean { t | nil }
```

Description

Specifies where the pins of the made cell created using the selected virtual hierarchy are positioned—below the virtual hierarchy boundary or on the boundary.

The default is `t`, which means the new pins are created below the virtual hierarchy boundary.

When to `nil`, the pins are created on the boundary.

GUI Equivalent

Command:	*Plan – Manage Hierarchy – Make Cell*
Field:	*Create pins – Below boundary*
	Create pins – On boundary
	(Make Cell)
Command:	*Make Cell* (Design Planning Toolbar)
Field:	*Create pins – Below boundary*
	Create pins – On boundary
	(Make Cell)

Examples

```
envGetVal("layoutDP" "makeCellPinsBelow")
envSetVal("layoutDP" "makeCellPinsBelow" 'boolean t)
envSetVal("layoutDP" "makeCellPinsBelow" 'boolean nil)
```

Related Topics

List of Design Planning Environment Variables

Setting Environment Variables

makeCellPushInternalRoutesOnly

```
layoutDP makeCellPushInternalRoutesOnly boolean { t | nil }
```

Description

Pushes only the internal routes to the made cellview, leaving behind any top-level routes and implementations.

The default is `nil`.

GUI Equivalent

Command:	*Plan – Manage Hierarchy – Make Cell*
Field:	*Push – Internal routes only* (Make Cell)
Command:	*Make Cell* (Design Planning Toolbar)
Field:	*Push – Internal routes only* (Make Cell)

Examples

```
envGetVal("layoutDP" "makeCellPushInternalRoutesOnly")
envSetVal("layoutDP" "makeCellPushInternalRoutesOnly" 'boolean t)
envSetVal("layoutDP" "makeCellPushInternalRoutesOnly" 'boolean nil)
```

Related Topics

List of Design Planning Environment Variables

Setting Environment Variables

makeCellPushInBlock

```
layoutDP makeCellPushInBlock boolean { t | nil }
```

Description

Pushes the top-level implementation of power structures and signal net routing to the block level.

The default is `nil`.

GUI Equivalent

Command:	*Plan – Manage Hierarchy – Make Cell*
Field:	*Push – Into block*
	(Make Cell)
Command:	*Make Cell* (Design Planning Toolbar)
Field:	*Push – Into block*
	(Make Cell)

Examples

```
envGetVal("layoutDP" "makeCellPushInBlock")
envSetVal("layoutDP" "makeCellPushInBlock" 'boolean t)
envSetVal("layoutDP" "makeCellPushInBlock" 'boolean nil)
```

Related Topics

List of Design Planning Environment Variables

Setting Environment Variables

makeCellPushRoutesAsBlockages

```
layoutDP makeCellPushRoutesAsBlockages boolean { t | nil }
```

Description

Controls whether the overlapping routes are pushed into the detached cell as routes or blockages and overlapping blockages are pushed into the cell and WSP/row regions.

The default is `t`.

GUI Equivalent

Command:	*Plan – Manage Hierarchy– Make Cell*
Field:	*Push – Routes as blockages* (Make Cell)
Command:	*Make Cell* (Design Planning Toolbar)
Field:	*Push – Routes as blockages* (Make Cell)

Examples

```
envGetVal("layoutDP" "makeCellPushRoutesAsBlockages")
envSetVal("layoutDP" "makeCellPushRoutesAsBlockages" 'boolean t)
envSetVal("layoutDP" "makeCellPushRoutesAsBlockages" 'boolean nil)
```

Related Topics

Push Pre-Routes

List of Design Planning Environment Variables

Setting Environment Variables

makeCellType

```
layoutDP makeCellType cyclic { "softMacro" | "digital softMacro" | "block" | "none"
      }
```

Description

Specifies the cell and block type to be created when running the *Make Cell* command on a virtual hierarchy.

The default is `softMacro`, which creates a soft block type cellview from the virtual hierarchy.

When set to `none`, creates a custom cell type.

When set to `digital softMacro`, creates a soft block with hierarchy for a block of type `digital`.

When set to `block`, creates a hard block.

GUI Equivalent

Command:	*Plan – Manage Hierarchy –Make Cell*
Field:	*Type*
	(Make Cell)
Command:	*Make Cell* (Design Planning Toolbar)
Field:	*Type*
	(Make Cell)

Examples

```
envGetVal("layoutDP" "makeCellType")
envSetVal("layoutDP" "makeCellType" 'cyclic "softMacro")
envSetVal("layoutDP" "makeCellType" 'cyclic "digital softMacro")
envSetVal("layoutDP" "makeCellType" 'cyclic "block")
envSetVal("layoutDP" "makeCellType" 'cyclic "none")
```

Related Topics

List of Design Planning Environment Variables

Setting Environment Variables

makeCellVirtualClones

```
layoutDP makeCellVirtualClones boolean { t | nil }
```

Description

Replaces all the clones of the selected virtual hierarchy with the new cellview.

The default is `t`.

GUI Equivalent

Command:	*Plan – Manage Hierarchy – Make Cell*
Field:	*All clones*
	(Make Cell)
Command:	*Make Cell* (Design Planning Toolbar)
Field:	*All clones*
	(Make Cell)

Examples

```
envGetVal("layoutDP" "makeCellVirtualClones")
envSetVal("layoutDP" "makeCellVirtualClones" 'boolean t)
envSetVal("layoutDP" "makeCellVirtualClones" 'boolean nil)
```

Related Topics

List of Design Planning Environment Variables

Setting Environment Variables

makeVirtualAllInstsSameMaster

```
layoutDP makeVirtualAllInstsSameMaster boolean { t | nil }
```

Description

Controls whether all the instances of the selected layout master are replaced with the layout hierarchy realized externally.

The default is `nil`.

GUI Equivalent

Command:	*Make Virtual Hierarchy* (Design Planning Toolbar)
Field:	*All instances of the same master*
	(Make Virtual Hierarchy)

Examples

```
envGetVal("layoutDP" "makeVirtualAllInstsSameMaster")
envSetVal("layoutDP" "makeVirtualAllInstsSameMaster" 'boolean t)
envSetVal("layoutDP" "makeVirtualAllInstsSameMaster" 'boolean nil)
```

Related Topics

List of Design Planning Environment Variables

Setting Environment Variables

makeVirtualPreserveVirtualPins

```
layoutDP makeVirtualPreserveVirtualPins boolean { t | nil }
```

Description

Controls whether the pinFigs in the cell are retained as shapes. These are used later to create the pin shapes in the made cell, if you run the Make Cell command on the virtual hierarchy again.

The default is `t`.

When set to `nil`, the pinFigs are deleted during the Make Virtual Hierarchy command.

GUI Equivalent

Command:	*Make Virtual Hierarchy* (Design Planning Toolbar)
Field:	*Virtual pins* (Make Virtual Hierarchy)

Examples

```
envGetVal("layoutDP" "makeVirtualPreserveVirtualPins")
envSetVal("layoutDP" "makeVirtualPreserveVirtualPins" 'boolean t)
envSetVal("layoutDP" "makeVirtualPreserveVirtualPins" 'boolean nil)
```

Related Topics

List of Design Planning Environment Variables

Setting Environment Variables

pinLayerLimit

```
layoutDP pinLayerLimit cyclic { "Highest Routing Layer + N" | "All" }
```

Description

Controls whether congestion-aware pins are allowed on all routing layers or on a restricted set of layers.

- When set to `"Highest Routing Layer + N"`, pin creation is allowed up to the layer number derived by adding the highest routing layer for virtual hierarchies and the number (*N*) of allowed routing layers specified.
- When set to `"All"`, pin creation is allowed on all routing layers.

The default is `"Highest Routing Layer + N"`.

GUI Equivalent

Command:	*Plan – Options*
Field:	*Pin Layers*
	(Design Planning and Analysis Options)

Examples

```
envGetVal("layoutDP" "pinLayerLimit")
envSetVal("layoutDP" 'cyclic "Highest Routing Layer + N")
envSetVal("layoutDP" 'cyclic "All")
```

Related Topics

List of Design Planning Environment Variables

Setting Environment Variables

pinLayerLimitNum

```
layoutDP pinLayerLimitNum int integer
```

Description

Specifies the number of routing layers above the highest routing layer supported for virtual hierarchies that can be used for creating congestion-aware pins.

The default is `0`.

GUI Equivalent

Command:	*Plan – Options*
Field:	*Pin Layers - N*
	(Design Planning and Analysis Options)

Examples

```
envGetVal("layoutDP" "pinLayerLimitNum")
envSetVal("layoutDP" "pinLayerLimitNum" 'int 0)
envSetVal("layoutDP" "pinLayerLimitNum" 'int 2)
```

Related Topics

List of Design Planning Environment Variables

Setting Environment Variables

pinOptOneConnectionPerSide

```
layoutDP pinOptOneConnectionPerSide boolean { t | nil }
```

Description

Controls the number of routing connections per side for congestion analysis with pin optimization. When set to `t`, one connection is specified per side for each opaque virtual hierarchy or a soft block. When set to `nil`, one connection is specified per opaque virtual hierarchy or a soft block.

The default value is `t.`

GUI Equivalent

Command	*Route – Design Setup – Congestion Analysis*
Form Field	*Allow One Connection*
	Congestion Analysis Assistant

Examples

```
envGetVal("layoutDP" "pinOptOneConnectionPerSide")
envSetVal("layoutDP" "pinOptOneConnectionPerSide" 'boolean nil)
envSetVal("layoutDP" "pinOptOneConnectionPerSide" 'boolean t)
```

Related Topics

List of Design Planning Environment Variables

Setting Environment Variables

softBlockArea

`layoutDP softBlockArea float` *`floating_point_number`*

Description

Specifies an estimated area value of the soft blocks to be generated.

The default is `100`.

GUI Equivalent

Command:	*Connectivity – Generate Layout*
Field:	*Soft Block – Area*
	Generate Layout

Examples

```
envGetVal("layoutDP" "softBlockArea")
envSetVal("layoutDP" "softBlockArea" 'float 195)
```

Related Topics

List of Design Planning Environment Variables

Setting Environment Variables

updateSoftBlocksFromSymbol

```
layoutDP updateSoftBlocksFromSymbol boolean { t | nil }
```

Description

Updates the softblock master during an *Update Components And Nets* run if the bound schematic symbol is modified.

The default is `nil`.

GUI Equivalent

Command:	*Connectivity – Update Components And Nets (Hierarchy tab)*
Field:	*Update Soft Blocks*
	Update Components And Nets

Examples

```
envGetVal("layoutDP" "updateSoftBlocksFromSymbol")
envSetVal("layoutDP" "updateSoftBlocksFromSymbol" 'boolean t)
envSetVal("layoutDP" "updateSoftBlocksFromSymbol" 'boolean nil)
```

Related Topics

List of Design Planning Environment Variables

Setting Environment Variables

useAreaBoundaryUtilization

```
layoutDP useAreaBoundaryUtilization boolean { t | nil }
```

Description

Controls whether the areaBoundaryUtilization environment variable is used to determine the area boundary size for the virtual hierarchy.

The default is `nil`, which means the areaBoundaryEnclosure environment variable is used to calculate the area boundary.

GUI Equivalent

Command:	*Connectivity – Generate Layout*
Field:	*Soft Block – Area*
	Generate Layout

Examples

```
envGetVal("layoutDP" "useAreaBoundaryUtilization")
envSetVal("layoutDP" "useAreaBoundaryUtilization" 'boolean t)
envSetVal("layoutDP" "useAreaBoundaryUtilization" 'boolean nil)
```

Related Topics

List of Design Planning Environment Variables

Setting Environment Variables

useBindKeys

```
layoutDP useBindKeys boolean { t | nil }
```

Description

Controls whether the user-defined bindkeys are used instead of the default bindkeys in Design Planner.

The default is `t`, which means the Design Planner bindkeys are used for the design that has a virtual hierarchy.

GUI Equivalent

Command:	*Plan – Options*
Field:	*Use bind keys*
	(Design Planning and Analysis Options)

Examples

```
envGetVal("layoutDP" "useBindKeys")
envSetVal("layoutDP" "useBindKeys" 'boolean t)
envSetVal("layoutDP" "useBindKeys" 'boolean nil)
```

Related Topics

List of Design Planning Environment Variables

Setting Environment Variables

verboseApOnVhEdit

```
layoutDP verboseApOnVhEdit boolean { t | nil }
```

Description

Controls whether the messages related to automatic placement of objects inside a virtual hierarchy are displayed during an edit of the area boundary due to any of these commands: Adjust Boundary, Move, Stretch, Chop, or Reshape

The default is `nil`.

GUI Equivalent

None

Examples

```
envGetVal("layoutDP" "verboseApOnVhEdit")
envSetVal("layoutDP" "verboseApOnVhEdit" 'boolean t)
envSetVal("layoutDP" "verboseApOnVhEdit" 'boolean nil)
```

Related Topics

autoPlaceOnAreaBoundaryEdit

List of Design Planning Environment Variables

Setting Environment Variables

vhCloneColor

`layoutDP vhCloneColor string "`*`highlightColor`*`"`

Description

Specifies the color of the virtual hierarchy clone bounding box.

The default is `"hilite5"`.

Arguments

None

GUI Equivalent

Command:	*Plan – Options*
Field:	*Color – Clone*
	(Design Planning and Analysis Options)

Examples

```
envGetVal("layoutDP" "vhCloneColor")
envSetVal("layoutDP" "vhCloneColor" 'string "hilite5")
```

Related

List of Design Planning Environment Variables

Setting Environment Variables

vhCreatedColor

```
layoutDP vhCreatedColor string "highlightColor"
```

Description

Specifies the bounding box color for the created virtual hierarchies.

The default is `"hilite6"`.

Arguments

None

GUI Equivalent

Command:	*Plan – Options*
Field:	*Color – Created*
	(Design Planning and Analysis Options)

Examples

```
envGetVal("layoutDP" "vhCreatedColor")
envSetVal("layoutDP" "vhCreatedColor" 'string "hilite5")
```

Related

List of Design Planning Environment Variables

Setting Environment Variables

vhDimming

```
layoutDP vhDimming boolean { t | nil }
```

Description

Specifies whether you can change the luminosity of a virtual hierarchy.

The default value is `t`, which means automatic dimming is enabled for the non-editable virtual hierarchy and virtual hierarchy clones.

When set to `nil`, dimming control is not available for the virtual hierarchy and its clones.

GUI Equivalent

Command	*Options – Display*
Field	*Dim Virtual Hierarchy* (Display Options Form)

Examples

```
envGetVal("layoutDP" "vhDimming")
envSetVal("layoutDP" "vhDimming" 'boolean t)
envSetVal("layoutDP" "vhDimming" 'boolean nil)
```

Related Topics

List of Design Planning Environment Variables

Setting Environment Variables

vhGeneratedColor

```
layoutDP vhGeneratedColor string "highlightColor"
```

Description

Specifies the bounding box color for the generated virtual hierarchies.

The default is `"hilite5"`.

Arguments

None

GUI Equivalent

Command:	*Plan – Options*
Field:	*Color – Generated*
	(Design Planning and Analysis Options)

Examples

```
envGetVal("layoutDP" "vhGeneratedColor")
envSetVal("layoutDP" "vhGeneratedColor" 'string "hilite9")
```

Related

List of Design Planning Environment Variables

Setting Environment Variables

vhSelectiveMode

```
layoutDP vhSelectiveMode boolean { t | nil }
```

Description

Enables selective virtual hierarchy generation mode.

When set to `t`, a completely flat layout is generated even with the *Virtual Hierarchy* option enabled on the Generate Layout form. But, you can allow specific virtual hierarchies to be generated by marking the cells for virtual hierarchy generation using the CPH cells table. See Creating Virtual Hierarchy for Selected Cells.

The default value is `nil`, which means all virtual hierarchies are generated.

GUI Equivalent

Command	*Options*
Field	*Generate – Virtual hierarchy*
	(Design Planning and Analysis Options)
Command	*Launch – Configure Physical Hierarchy – Hierarchy Configuration Mode – Cells (table)*
Field	*Create/remove Virtual Hierarchy*
	(Context Menu for the Hierarchy Configuration Cells Table)

Examples

```
envGetVal("layoutDP" "vhSelectiveMode")
envSetVal("layoutDP" "vhSelectiveMode" 'boolean t)
envSetVal("layoutDP" "vhSelectiveMode" 'boolean nil)
```

Related Topics

List of Design Planning Environment Variables

Setting Environment Variables

vhSymbolOverlay

```
layoutDP vhSymbolOverlay boolean { t | nil }
```

Description

Overlays a virtual hierarchy block at the top-cell level with the schematic symbol that represents the virtual hierarchy.

The default value is `nil`.

GUI Equivalent

Command	*Plan – Options*
Field	*Symbol overlay*
	(Design Planning and Analysis Options)

Examples

```
envGetVal("layoutDP" "vhSymbolOverlay")
envSetVal("layoutDP" "vhSymbolOverlay" 'boolean t)
envSetVal("layoutDP" "vhSymbolOverlay" 'boolean nil)
```

Related Topics

List of Design Planning Environment Variables

Setting Environment Variables

B

Design Planning Forms

This section lists the forms that can either be invoked only through the Virtuoso® Design Planner toolbar or forms that can also be invoked outside but have options that are related to the Design Planner flow.

Adjust Boundary

Use the **Adjust Boundary** form to resize or create the area boundary for the selected virtual hierarchy, an instance of the selected soft block, the PR boundary for the selected soft block, or the top-level PR boundary.

Rectangle creates a rectangular area boundary around the selected virtual hierarchy, a PR boundary for the selected soft block, or a top-level PR boundary.

Note: The boundary is snapped to the nearest grid.

Enclose by specifies the distance between the rectangular boundary and the PR boundary of the instances inside a virtual hierarchy or soft block. If the instances inside the virtual hierarchy do not have a PR boundary, the distance from the instance bounding box is used.

Environment variable: areaBoundaryEnclosure

Instances only encloses only instances inside a virtual hierarchy or soft block with a rectangular boundary.

All shapes encloses all shapes inside a virtual hierarchy or soft block—such as area boundaries, blockages, and shapes—with a rectangular PR boundary or bounding box.

Maskable shapes, such as rulers, markers, text displays, and labels are not enclosed in a PR boundary or a bounding box.

Aspect Ratio (W/H) specifies the width to height ratio used to determine the size of the area boundary for a virtual hierarchy or the PR boundary for a soft block.

Environment variable: areaBoundaryAspectRatio

Utilization (%) specifies the area utilization percentage to be used when estimating the size of the area boundary for a virtual hierarchy or the size of the PR boundary for a soft block.

Environment variable: areaBoundaryUtilization

Register opens the Add Area Estimators form. You can use the form to define a SKILL function that can estimate the area boundary for the selected virtual hierarchy or the PR boundary for the selected soft block. See Registering a User-Defined Area Estimation Function.

When registered, the area estimation function is listed on the form by its *Nickname*, which is the name associated with the function at the time of its registration. See Add Area Estimators form.

PR Boundary based area estimator adds up the instance PR boundary area, if available, to derive the overall area estimation for the virtual hierarchy or soft block. If the instance PR boundary area is not available, the bounding box of the individual instances is used.

BBox based area estimator adds up the bounding box area of the individual instances to derive the overall area estimation for the virtual hierarchy or soft block.

Width specifies the width of the area boundary for a virtual hierarchy or the width of the PR boundary for a soft block.

Environment variable: areaBoundaryWidth

Height specifies the height of the area boundary for a virtual hierarchy or the height of the PR boundary for a soft block.

Environment variable: areaBoundaryHeight

You can select multiple virtual hierarchies or soft blocks simultaneously to adjust their boundaries. When only one virtual hierarchy or soft block is selected, its width (or height) is displayed on the form.

Constant area maintains the boundary of the selected virtual hierarchy or soft block at a specified *width*, *height*, or *utilization%*. This ensures that the area enclosed by the boundary remains constant.

Adjust hierarchically automatically adjusts the area boundary of the parent virtual hierarchy when a lower-level virtual hierarchy or soft block has its area boundary adjusted.

off leaves the parent virtual hierarchy area boundary unadjusted.

if outside adjusts the area boundary of the parent virtual hierarchy when the area boundary of a virtual hierarchy or a soft block extends outside the parent area boundary.

Environment variable: adjustBoundaryCheckOutside

all levels adjusts the area boundary of all parent virtual hierarchies and soft blocks in the hierarchy.

all levels including top adjusts the area boundary of parent virtual hierarchies and soft blocks at all levels in the hierarchy and the PR boundary at the top level. The PR boundary area adjustment is performed after all the required area boundary adjustments at lower levels have been made.

Environment variable: adjustBoundaryIncludeTop

Polygon creates a rectilinear area boundary around the selected virtual hierarchy or soft block.

Enclose by specifies the distance between the rectilinear boundary and the PR boundary of the instances inside a virtual hierarchy or soft block. If the instances inside the virtual hierarchy do not have a PR boundary, the distance from the instance bounding box is used.

Environment variable: areaBoundaryEnclosure

Instances only encloses only instances inside a virtual hierarchy or soft block with a rectilinear boundary.

All shapes encloses all shapes inside a virtual hierarchy or soft block—such as area boundaries, blockages, and shapes—with a rectilinear PR boundary.

Minimum jog length specifies the minimum edge length of the polygonal area boundary.

Environment variable: areaBoundaryMinJogLength

PR Boundary based area estimator adds up the instance PR boundary area, if available, to derive the overall area estimation for the virtual hierarchy or soft block. If the instance PR boundary area is not available, the bounding box of the individual instances is used.

BBox based area estimator adds up the bounding box area of the individual instances to derive the overall area estimation for the virtual hierarchy or soft block.

Adjust hierarchically automatically adjusts the area boundary of the parent virtual hierarchy when a lower-level virtual hierarchy or soft block has its area boundary adjusted.

off leaves the parent virtual hierarchy area boundary unadjusted.

if outside adjusts the area boundary of the parent virtual hierarchy when the area boundary of a virtual hierarchy or a soft block extends outside the parent polygonal area boundary.

Environment variable: adjustBoundaryCheckOutside

all levels including top adjusts the area boundary of parent virtual hierarchies and soft blocks at all levels in the hierarchy and the PR boundary at the top level. The PR boundary area adjustment is performed after all the required area boundary adjustments at lower levels have been made.

Environment variable: adjustBoundaryIncludeTop

Related Topics

Adjusting the Area Boundary

Design Planning Forms

Analyze Connectivity

Use the **Analyze Connectivity** form to calculate the number of connections running through virtual hierarchy figGroups, soft blocks, or hard blocks, draw flight lines between each connected pair, and indicate the number of connections by displaying a numerical value on the flight lines. Select a virtual hierarchy figGroup, a soft block, or a hard block to view the connections to other associated virtual hierarchy figGroup, soft blocks, or hard blocks. In addition, you can choose to display the common net connectivity, select connected nets, and create the net sets in the Navigator assistant.

For more information, see Analyze Connectivity in the *Virtuoso Floorplanner User Guide*.

Related Topics

Design Planning Forms

Automatic Placement

Use the **Automatic Placement** form to select the type of placement required—*Custom Digital Placement* to place standard cells or *Analog Refinement* to fine-tune the placement of analog devices.

For more information, see Automatic Placement in the *Virtuoso Placer User Guide*.

Related Topics

Design Planning Forms

Congestion Analysis

Use the **Congestion Analysis** assistant to assess the routing capacity of a design. Use the assistant to represent the congestion in a design graphically using a heat map, statistically using a histogram, or use the routing metrics table to map capacity and to plan routing.

For more information, see Running Congestion Analysis in the *Virtuoso Space-based Router User Guide*.

Related Topics

Congestion Analysis Assistant

Design Planning Forms

Create Virtual Group

Use the **Create Virtual Group** form to manually create a virtual group around the selected top-level instances or instances inside a virtual hierarchy. Virtual group creation allows creating a virtual hierarchy that does not match the schematic hierarchy.

Name lets you specify a name for the virtual group to be created.

Orientation lets you choose an orientation for the virtual group.

Choose orientation from:

Value	Meaning
`R0`	No rotation
`R90`	Rotate 90 degrees clockwise
`R180`	Rotate 180 degrees clockwise
`R270`	Rotate 270 degrees clockwise
`MY`	Mirror over the Y axis
`MYR90`	Mirror over the Y axis and rotate 90 degrees clockwise
`MX`	Mirror over the X axis
`MXR90`	Mirror over the X axis and rotate 90 degrees clockwise

Related Topics

Design Planning Forms

Generate Layout

Use the **Generate Layout** form to generate a virtual hierarchy for the schematic instances that have no layout counterparts generated and create soft blocks for schematic symbols with no schematic or virtual hierarchy.

Generate Tab

Design Planning

Use this group box to specify whether a virtual hierarchy is generated.

Virtual Hierarchy lets you generate a “yet-to-be-realized” hierarchy for the schematic instances that have no layout counterparts generated.

Environment variable: generateVirtualHierarchy

Auto Generate Soft Blocks lets you generate soft blocks for the top-level virtual hierarchy blocks that have no schematic or layout available. Because there is no schematic available, soft block generation uses the bound symbol view to generate pins. Soft block boundary is generated using the options defined on the PR Boundary Tab.

Environment variable: generateSoftBlocks

PR Boundary Tab

Virtual Hierarchy Area Boundary

Use this group box to specify how to generate the area boundary of a virtual hierarchy.

Enclose by specifies the distance from the objects inside the virtual hierarchy at which the area boundary is created.

Utilization (%) specifies the acceptable area utilization percentage for deriving the size of the area boundary for the virtual hierarchy. The utilization percentage also takes into account the size of each sub-virtual hierarchy to account for the additional space required.

Environment variable: useAreaBoundaryUtilization, which can switch the option to use either the **Enclose by** option (areaBoundaryEnclosure) or the **Utilization (%)** option (initUtilization).

Top Level specifies that the selected area boundary settings be used for generating virtual hierarchy area boundaries only at the top level.

All Levels specifies that the selected area boundary settings be used for generating virtual hierarchy area boundaries at all levels in the virtual hierarchy of the top-level design.

None specifies that the virtual hierarchies for the design are generated without an area boundary.

Environment variable: generateAreaBoundaries

Soft Block

Use this group box to specify the area of the soft block to be generated.

Area specifies an estimated area value of the soft blocks to be generated.

Environment variable: softBlockArea

Note: The value specified using the *Area* field is applied by default to all the soft blocks that are generated. To override this value and to customize the area of each generated soft block, you can specify a cellview property called `area` on the Symbol Generation Options form when creating a new symbol. Alternatively, you can update the property for an existing symbol using the Edit Cellview Properties form. In either case, a floating point value is specified for the `area` property.

For information on other layout generation options provided on the form, see Generate Layout in the *Virtuoso Layout Suite XL: Constraint Aware Editing User Guide*.

Related Topics

lxHiGenerateVirtualHierarchy

Generating All Components from Source

Design Planning Forms

Generate Selected Components

Use the **Generate Selected Components** form to generate a virtual hierarchy for the selected schematic instances and create soft blocks for schematic symbols with no schematic or virtual hierarchy.

If you have the Virtuoso® Layout Suite EXL license checked out and the selected placement mode is *As In Schematic*, the **Generate Selected Components** form displays an additional Design Planning group box.

Design Planning

Use this group box to specify whether a virtual hierarchy is generated.

> **Virtual Hierarchy** lets you generate a virtual hierarchy for the selected schematic instances.
>
> By default, the option is ON, if you launch the **Generate Selected Components** form using the Plan Menu or the Design Planning Toolbar. If the form is launched using the Layout XL *Connectivity – Generate Selected From Source* menu, the *Virtual Hierarchy* option is, by default, OFF.

Note: For information on other layout generation options provided on the form, see Generate Selected Components in the *Virtuoso Layout Suite XL: Constraint Aware Editing User Guide*.

Related Topics

lxHiGenerateSelectedVirtualHierarchy

Generating Selected Components

Generating Components As In Schematic

Design Planning Forms

Global Route

Use the **Global Route** form to route all nets in the layout.

For more information, see Global Route Settings in the *Virtuoso Space-based Router User Guide*.

Related Topics

Global Route

Congestion Analysis Assistant

Design Planning Forms

Make Cell

Use the **Make Cell** form to create real cellviews for the selected virtual hierarchies. You can also use the form to create a bottom-up layout view for each level of virtual hierarchy inside the selected virtual hierarchy.

If the design is opened in Layout EXL and no virtual hierarchy is pre-selected, the design must contain a virtual hierarchy for the Make Cell form to display the options related to virtual hierarchies. If the design is opened in Layout EXL and the *Make Cell* command is invoked post selection, the selected figGroup must be a virtual hierarchy figGroup. Otherwise, the Make Cell form opens displaying the regular Layout XL options.

Library specifies the name of the library to be used for creating a new cellview for the selected virtual hierarchy blocks. When the *All levels* option is selected, the same library is used to create the layout views at all levels as the library of the selected virtual hierarchy.

Cell specifies the name of the cell to be used for creating a new cellview for the selected virtual hierarchy blocks. When the *All levels* option is selected, the cell name used at each level is the same as the name of the corresponding schematic cell.

View specifies the name of the view to be created for the virtual hierarchy blocks. If a view name is not specified, the default view name, `layout_variant_1` is used. The subsequent view names are then automatically incremented to `layout_variant_2`, and so on.

Important

> Cadence recommends that the layout view name for the new cell should not be a name defined in CPH Global Bindings, for example `layout`, which is a default physical binding in CPH. When the *Make Cell – All clones* option is not selected or the multiple virtual hierarchies using the same schematic cell in the design are not being made into cells at the same time, all the occurrences of the new cell automatically use the made cell layout. When *Update Components and Nets* is run, the command replaces all the virtual hierarchies matching the schematic cell with the made cell layout. To avoid impacting the physical bindings of all the instances of the cell, the default view name used is `layout_variant_x`.

Type specifies the type of the cell to be created.

- ❑ The default is `softMacro`, which creates a soft block type cellview from the virtual hierarchy.
- ❑ Choose cell type as *none* to create a `custom` cell type.
- ❑ Choose cell type as `digital softmacro` to create a soft block with hierarchy for a block of type `digital`.

❑ Choose cell type as `block` to create hard macros that can be supported by macro placers, Layout XL commands such as *Load Physical View*, and other applications that support only macros and blocks.

Environment variable: makeCellType

Hierarchy

All clones replaces all instances of the selected virtual hierarchy clone with the instances of the new cellview. The option is available only when the selected virtual hierarchy is a clone. Else, the option is grayed out.

If the **All clones** option is deselected or non-clone virtual hierarchies are selected, the *Push* options are enabled. In this case, the value of the *Push – Into block* option is set based on the value of the makeCellPushInBlock environment variable.

Environment variable: makeCellVirtualClones

All levels uses the bottom-up approach to create cellviews for all virtual hierarchy levels inside the selected virtual hierarchy and then creates a cellview for the selected virtual hierarchy.

Create pins creates interface pins for the selected virtual hierarchy.

Congestion aware runs the congestion-aware global router to automatically create pins on the boundary of the virtual hierarchy.

By default, pin creation for the new made cell in Design Planner is congestion-aware. For more information on congestion-aware pin placement, see Global Route and Congestion Analysis Assistant.

Environment variable: makeCellOptPins

On boundary creates pins on the boundary of the virtual hierarchy ensuring the shortest possible net length in the direction of routing. Power and ground pins and any pins that could not be routed are also placed on the boundary.

Note: The creation of pins on or below the virtual hierarchy boundary with the *All levels* option selected depends on the current display depth set using the *Display Depth Options* on the Design Planner toolbar. If the current display depth is set to *0*, only top-level virtual hierarchies have their pins created based on the options selected on the form. Lower-level virtual hierarchies have their pins created below the virtual hierarchy boundary. If the display depth is increased to `1`, virtual hierarchies at the top level and at one level below have their pins created using the options selected on the form. Virtual hierarchies further down have their pins created below the boundary. Pin creation follows the selected pin creation options on the form only for visible virtual hierarchies. Any virtual hierarchies at lower levels that are not set to be displayed have their pins created below the boundary.

Environment variable: makeCellPinsBelow (when set to `nil`)

Below boundary creates pins just below the boundary of the virtual hierarchy.

Environment variable: makeCellPinsBelow (when set to `t`)

Width specifies the width of the pins to be created.

Height specifies the height of the pins to be created.

Note: If you choose the **Below boundary** pin creation option, the associated layer-purpose pair drop-down and the *Width* and *Height* fields gets enabled and you can specify appropriate value in each field to use for the pin creation. For congestion-aware pin creation, the layer-purpose pair is automatically identified.

Label allows you to choose whether or not to create a label for the new pin. You can choose from creating a label as *Label* or as a *Text Display*. You can choose to create pin labels irrespective of the pin creation mode you are in.

Text style opens the Layout XL Set Pin Label Text Style form. You can use the options on the form to specify the style to be used for the new pin labels.

Delete virtual pins deletes any virtual pins in the design to allow new pins to be generated.

Environment variable: makeCellDeleteVirtualPins

Push

Into block pushes the top-level implementation of power structures and signal net routing to the block level. Pushes both metal and poly layers to the block level.

Environment variable: makeCellPushInBlock

Routes as blockages pushes the overlapping routes into the made cell as routes or as blockages. Push also pushes overlapping blockages into the cell and Width Spacing Patterns/row regions.

Environment variable: makeCellPushRoutesAsBlockages

For more information, see Push Pre-Routes and Updating Soft Blocks in *Virtuoso Floorplanner User Guide*.

Internal routes only pushes the internal routes into the made cell, and no top-level routes are pushed.

Environment variable: makeCellPushInternalRoutesOnly

Note: The *Push* options are supported only for the selected virtual hierarchy and not for the virtual hierarchies inside the selected virtual hierarchy.

Related Topics

Planning a Design

Design Planning Forms

Make Virtual Hierarchy

Use the **Make Virtual Hierarchy** form to integrate layout hierarchies that have been realized outside the design.

Note: The command can be used to integrate multiple selected cells as virtual hierarchies.

All instances of the same master replaces all the instances of the selected layout master with the layout hierarchy realized externally.

Environment variable: makeVirtualAllInstsSameMaster

Virtual pins specifies whether pinFigs in the cell are retained as shapes to be used later to create the pin shapes in the made cell, if you use the virtual hierarchy to Make Cell again. If the option is selected, the `lxStickyNet` property is added to the virtual pins to preserve the connectivity.

If the option is not selected, the pinFigs are deleted when running the **Make Virtual Hierarchy** command.

Note: If you have the *Labels* option selected, the Layout XL connectivity extractor option *Assign shapes from overlapping labels as defined by ‘stampLabelLayers’ rule* should not be set.

Environment variable: makeVirtualPreserveVirtualPins

Labels specifies whether the attached labels are created when the layout hierarchy is integrated. If the option is not selected, the labels are deleted. If you choose to create labels, you must ensure that you have the Assign shapes from attached labels option on the Layout XL Connectivity form switched off to avoid causing shorts.

Related Topics

Planning a Design

Design Planning Forms

Remaster

Use the **Remaster** form to replace the selected virtual hierarchy with the selected layout master that exists on disk.

Library specifies the library of the layout variant to be used to replace the selected virtual hierarchy.

Cell specifies the cell name of the layout variant to be used to replace the selected virtual hierarchy.

View specifies the name of the master variant view to be used to replace the selected layout.

All Clones replaces all the clones of the selected virtual hierarchy with the selected layout variant.

Related Topics

Planning a Design

Design Planning Forms

Update Components And Nets

Use the **Update Components And Nets** form to automatically update your layout to take account of the instances, pins, and connectivity you have changed in the schematic.

PR Boundary Tab

Virtual Hierarchy Area Boundary

Use this group box to specify how to generate the area boundary of a virtual hierarchy.

Enclose by specifies the distance from the objects inside the virtual hierarchy at which the area boundary is created.

Utilization (%) specifies the acceptable area utilization percentage for deriving the size of the area boundary for the virtual hierarchy. The utilization percentage also takes into account the size of each sub-virtual hierarchy to account for the additional space required.

Environment variable: useAreaBoundaryUtilization, which can switch the option to use either the **Enclose by** option (areaBoundaryEnclosure) or the **Utilization (%)** option (initUtilization).

Top Level specifies that the selected area boundary settings be used for creating virtual hierarchy area boundaries only at the top level.

All Levels specifies that the selected area boundary settings be used for creating virtual hierarchy area boundaries at all levels within the virtual hierarchy of the top-level design.

None specifies that the virtual hierarchy area boundaries are not created, instead existing virtual hierarchy boundaries are used.

Soft Block

Use this group box to specify the area of the soft block to be generated.

Area specifies the area of the soft blocks to be generated.

Hierarchy Tab

Design Planning

Use this group box to specify if the virtual hierarchy is updated to match the schematic hierarchy and if soft blocks are automatically generated for missing schematics.

Virtual Hierarchy lets you generate a virtual hierarchy using the area boundary options selected in the PR Boundary Tab, or updates an existing virtual hierarchy to match the schematic hierarchy. When updating an existing virtual hierarchy, the area boundary of the virtual hierarchy is not updated.

Environment variable: generateVirtualHierarchy

Auto Generate Soft Blocks lets you generate soft blocks for top-level virtual hierarchy blocks that have no schematic available. Because there is no schematic available, soft block generation uses the bound symbol view to generate pins. Soft block boundary is generated using the options defined on the PR Boundary Tab.

Environment variable: generateSoftBlocks

Update Soft Blocks updates the soft block in the layout to match the schematic symbol.

Environment variable: updateSoftBlocksFromSymbol

Note: For information on other layout update options provided on the form, see Update Components And Nets in the *Virtuoso Layout Suite XL: Constraint Aware Editing User Guide*.

Related Topics

Planning a Design

Design Planning Forms

Design Planning and Analysis Options

Use the **Design Planning and Analysis Options** form to control the display of virtual hierarchies and blocks in a design.

Generate

Virtual hierarchy lets you control the generation of virtual hierarchies.

all generates the virtual hierarchy for all hierarchy levels.

configured generates the virtual hierarchy only for the cells selected using the CPH Cells table and generates a completely flat layout otherwise. See Creating Virtual Hierarchy for Selected Cells.

If no cells are marked for virtual hierarchy generation, a completely flat layout is generated.

Environment variable: vhSelectiveMode

Auto Adjustment

Auto place on edit, when set to *PR boundary*, automatically places the top-level design if it contains virtual hierarchies and when the PR boundary is modified by using the Stretch, Chop, or Reshape command.

When the option is set to *area boundaries*, automatically places the contents inside the area boundary of the virtual hierarchy when the area boundary is modified using the Adjust Boundary, Stretch, Chop, or Reshape command.

When the option is set to *boundaries*, automatically places the contents inside the top-level PR boundary and the virtual hierarchy area boundaries, when the PR boundary and the virtual hierarchy area boundaries are modified.

When the option is set to *off*, automatic placement is disabled.

If the top-level PR boundary or the virtual hierarchy area boundary is modified using the Move command, the contents within the boundary do not move along to be placed at the new location. If the top-level PR boundary or the virtual hierarchy area boundary is modified using the Stretch command, the contents of the corresponding top-level design or the virtual hierarchy are accommodated inside the stretched PR boundary or the area boundary.

For the virtual hierarchies to be automatically placed during an edit, they must have their *Placement Status* set to `Fixed`.

Environment variables: autoPlaceOnAreaBoundaryEdit, autoPlaceOnPRBoundaryEdit

Fix placement status on move automatically sets the *placement* status of the virtual hierarchy to `Fixed` and displays an information pop-up indicating the change, when an instance is moved, stretched, flipped, or rotated inside the virtual hierarchy.

Environment variable: fixPlacementStatusOnMove

Auto adjust area boundary automatically resizes the area boundary of a virtual hierarchy to accommodate:

a) any instances and figGroups other than row region that are manually moved outside the boundary of the virtual hierarchy.

b) any virtual hierarchies that have had their area boundary manually stretched beyond the area boundary of the containing virtual hierarchy.

c) any soft blocks that have had their PR boundary manually stretched beyond the area boundary of the containing virtual hierarchy.

Environment variable: autoAdjustBoundary

Display

Name of controls the display of the virtual hierarchy and cell name on the layout canvas.

Virtual hierarchy displays only the name of the virtual hierarchy at the top-cell level.

Cell displays only the cell name at the top-cell level.

Both displays both the virtual hierarchy name and the cell name at the top-cell level.

Symbol overlay overlaps a virtual hierarchy block at the top-cell level with the schematic symbol that represents the virtual hierarchy.

Environment variable: vhSymbolOverlay

Use bind keys controls whether the user-defined bindkeys are used instead of the default Design Planner bindkeys.

By default the option is ON, which means the Design Planner bindkeys are used for the design that has a virtual hierarchy.

Environment variable: useBindKeys

Clone sets the bounding box color virtual hierarchy clones.

Environment variable: vhCloneColor

Generated sets the color for the generated virtual hierarchies.

Environment variable: vhGeneratedColor

Created sets the bounding box color for the created virtual hierarchies.

Environment variable: vhCreatedColor

Macro coloring assigns highlight colors for hard and soft blocks so that they are visually distinguishable in the layout canvas.

Soft block specifies the highlight color for soft blocks.

Environment variable: `softBlockColor`

Hard block specifies the highlight color for hard blocks.

Environment variable: `hardBlockColor`

Pins

Allow one pin connection controls the number of routing connections per side for congestion analysis with pin optimization.

per side specifies one connection for each side for each opaque virtual hierarchy or a soft block.

per virtual hierarchy or soft block specifies one connection for each opaque virtual hierarchy or soft block.

Environment variable: pinOptOneConnectionPerSide

Pin layers controls whether congestion-aware pins are allowed on all routing layers or on a restricted set of layers.

All allows congestion-aware pins to be created on all routing layers.

Highest Routing Layer + N restricts the creation of congestion-aware pins to the layer number derived by adding the highest routing layer for virtual hierarchies and the number (*N*) of allowed routing layers specified*.*

Environment variable: pinLayerLimit

N specifies the number of routing layers above the highest routing layer supported for virtual hierarchies, which can be used for creating congestion-aware pins.

Environment variable: pinLayerLimitNum

The *Pin layer* options are also applicable for *On boundary* pin creation. The *N* value in this case implies that the pin layer value can vary from the lowest routing layer to the value derived using *Highest Routing Layer + N.*

Auto resize on snap specifies whether pins are to be resized during pin snapping.

Auto update label specifies the layers to which labels are to be re-layered when running pin-related commands, such as the Pin Planner and Pin Optimizer.

custom layer purpose pair lets you specify the layer and purpose in the Label Layer Purpose form.

layer only re-layers labels to match the pins, but their purposes remain unchanged.

as-is does not re-layer labels.

Related Topics

Planning a Design