Virtuoso Design Planning and Analysis User Guide

Product Version IC23.1
November 2023

Printed in the United States of America.

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

1

Design Planning and Analysis

Virtuoso® Design Planning and Analysis tool provides hierarchical generation capabilities in connectivity-driven and constraint-driven flows, including electrically-aware flows. The Design Planning and Analysis (DPA) flow provides full flexibility in planning and analyzing analog and mixed-signal chips, and blocks— using hard blocks, soft blocks, and virtual hierarchies.The DPA flow supports both top-down and bottom-up approaches, 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.

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.

Prerequisites

Virtuoso Design Planning and Analysis is aimed at developers and designers of integrated circuits and assumes that you are familiar with:

- The Virtuoso Studio design environment and application infrastructure mechanisms supporting consistent operations between all Cadence tools.
- The applications for designing and developing integrated circuits in the Virtuoso Studio 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.

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

- **Virtuoso Release:** *ICADVM20.1* or a higher release version
- **Virtuoso Layout Suite Application:**

- *Virtuoso Layout Suite EXL* with 12 Virtuoso Layout Suite GXL tokens needed to run the floorplanning and virtual hierarchy editing commands. No tokens needed to open the design.
- Virtuoso Layout Suite MXL, no additional tokens needed.

Related Topics

How to Access the Design Planning Commands

Design Planning and Analysis SKILL Functions

How to Access the Design Planning Commands

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

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

  The *Design Planning* workspace displays.
- In the Virtuoso Layout Suite EXL menu bar, choose the *Plan* menu.

  The *Plan* menu displays.
- Right-click the menu bar in the Virtuoso Layout Suite EXL window and choose *Toolbars – Design Planning*.

  The *Design Planning* toolbar displays.

Related Topics

Design Planning Workspace

Plan Menu for Design Planning and Analysis

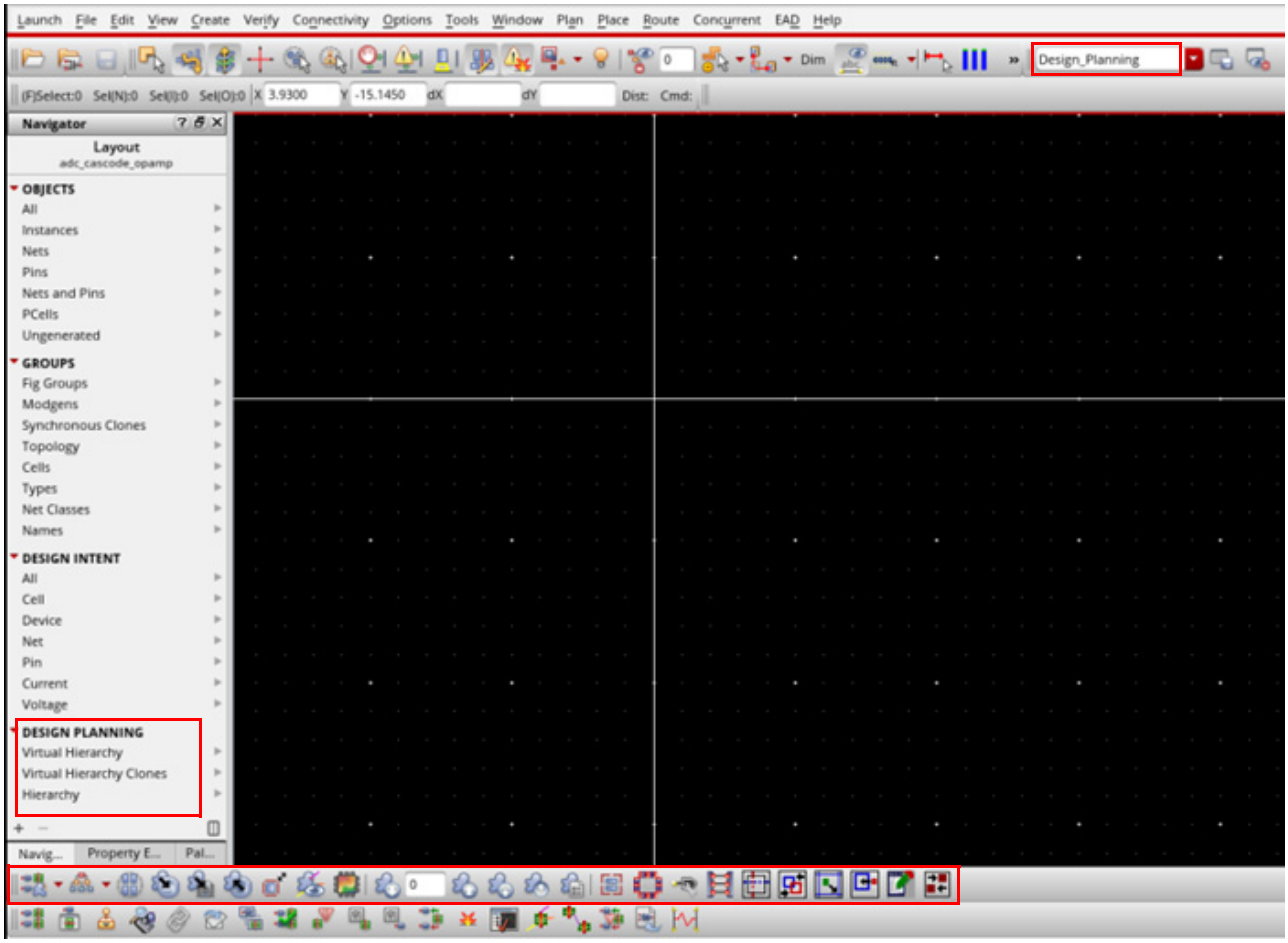
Design Planning Toolbar

Advantages of Design Planning and Analysis

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*.



Related Topics

Plan Menu for Design Planning and Analysis

Design Planning Toolbar

Navigator Assistant

Plan Menu for Design Planning and Analysis

The *Plan* menu gives you access to the various commands and related forms that are needed to support the design planning and analysis tasks. The most frequently used commands from the *Plan* menu are also available on the *Design Planning* toolbar.

More information on the individual commands in the *Plan* menu is given in the following table.

Command	Submenu Command	Use to...	Form
Generate	*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 Form
	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 Form
	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

Command	Submenu Command	Use to...	Form
	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	*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 Form
	Make Cell	Create new cellviews using virtual hierarchies selected from the top cellview.	Make Cell Form
	Make Virtual Hierarchy	Integrate layout hierarchies that were realized outside the design.	Make Virtual Hierarchy Form
	Remaster	Replace the selected virtual hierarchy block with the selected layout master that exists on disk.	Remaster Form
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

Command	Submenu Command	Use to...	Form
Block Planning	*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 Form
	Place Blocks	Automatically place all hard and soft blocks in a design and minimize the wire length and chip area.	Block Placer Form
	Load Solutions	Load other solutions that block placer generates.	Load Solutions Form
	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/Pin
	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

Command	Submenu Command	Use to...	Form
	Push Into Blocks	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
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 Form
	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 Form
	Pin Checker	Run checks on pins and report the results.	Pin Checker Form
	Create/Edit Pin Group Guide	Create or edit pin groups and guides interactively.	Pin Group Guide Form
	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 Form
	Create Soft Pins	Create additional soft pins on soft blocks.	Add Soft Pin

Command	Submenu Command	Use to...	Form
	Snap Pins	Snap top-level and level-1 pins to the grid appropriate to the block type.	Snap Pins
	Label Update	Update pin labels and text displays for clearer visualization.	Label Update Form
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
	Block Annotations	Display information about blocks.	Block Annotations Options

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

Related Topics

Design Planning Toolbar

Design Planning Workspace

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. 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.

Icon	Command	Description
	*Generate All From Source*	Generates a virtual hierarchy for the schematic instances that have no layout counterparts generated and creates soft blocks for schematic symbols with no schematic or virtual hierarchy. 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. See Generate Layout Form

Icon	Command	Description
	*Generate Selected From Source*	Generates a virtual hierarchy for the selected schematic instances and creates soft blocks for schematic symbols with no schematic or virtual hierarchy. 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. See Generate Selected Components Form
	*Generate Physical Hierarchy*	Generates the physical hierarchy based on the configuration defined in the Configure Physical Hierarchy (CPH) window. The *Generate Physical Hierarchy* command is available under a drop-down menu. The command can be toggled with the *Auto Generate Hierarchy* command. See Generate Physical Hierarchy
	*Auto Generate Hierarchy*	Generates a physical hierarchy by applying common parameters for boundary and pins and specifies the area estimator function to use for all the blocks. The *Auto Generate Hierarchy* command is available under a drop-down menu. The command can be toggled with the *Generate Physical Hierarchy* command. See Auto-Generate Hierarchy
	*Create Virtual Group*	Creates 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. See Create Virtual Group Form
	*Make Cell*	Creates new cellviews using virtual hierarchies selected from the top cellview. See Make Cell Form

Icon	Command	Description
	*Remaster*	Replaces the selected virtual hierarchy block with the selected layout master that exists on disk. See Remaster Form
	*Make Virtual Hierarchy*	Integrates layout hierarchies that were realized outside the design. See Make Virtual Hierarchy Form
	*Adjust Boundary*	Resizes or creates the area boundary for the selected virtual hierarchy, PR boundary for the selected soft block, or the top-level PR boundary. See Adjust Boundary Form
	*Analyze Connectivity*	Draws flight lines between each connected pair of virtual hierarchy groups, soft blocks, or hard blocks. Selects pairs of virtual hierarchy, soft block, or hard block to see the interconnections. See Analyze Connectivity
	*Congestion Analysis*	Analyzes the routing capacity of a design. See Congestion Analysis
	*Increase Display Depth*	Increases the display depth to display the objects deeper in the hierarchy. See Virtual Hierarchy Display Controls
	*Decrease Display Depth*	Decreases the display depth to display the objects at the higher levels in the hierarchy. See Virtual Hierarchy Display Controls
	*Set Default Display Depth 0*	Removes all stop level overrides and sets the default display depth to `0`. See Virtual Hierarchy Display Controls
	*Set Display Depth 32*	Removes all stop level overrides and sets the display depth to `32`. See Virtual Hierarchy Display Controls

Icon	Command	Description
	*Options*	Opens the Design Planning and Analysis Options Form.
	*IO Row Create*	Creates IO rows for the PAD type siteDefs available in the technology file. See Create IO Row
	*IO Placer*	Places instances of cell type `PAD` in IO rows. See IO Pad Placement
	*Pin Tool*	Creates, resizes, plans, and optimizes pins, edits pin attributes, and sets pin location constraints. See Using the Pin Tool
	*Pin Optimizer for All Pins*	Positions 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. See Pin Placement Form
	*Push Into Blocks*	Pushes the top-level implementation of power structures and signal routes into the block level, either as route or as a blockage. See Push into Blocks Form
	*Remove Block overlaps*	Removes overlaps and places overlapping hard blocks and soft blocks adjacent to each other. See Remove Block Overlap
	*Snap Soft Blocks To Grid*	Snaps soft blocks to placement grid or the manufacturing grid. See Snap Soft Blocks to Grid
	*Pull Soft Blocks inside PR Boundary*	Pulls the soft blocks overlapping the PR boundary to the core area. See Pull Soft Blocks Inside PR Boundary

Icon	Command	Description
	*Edit Soft Blocks*	Modifies soft block attributes. See Edit Soft Blocks
	*Auto-Create Pins*	Searches for net shapes in a design and creates boundary and buried pins automatically on these net shapes. See Auto-Create Pins.
	*Adjust Blocks Pin*	Adjusts the existing floorplan by abutting and pushing the soft blocks in the design. See Adjust Blocks/Pin

Related Topics

Plan Menu for Design Planning and Analysis

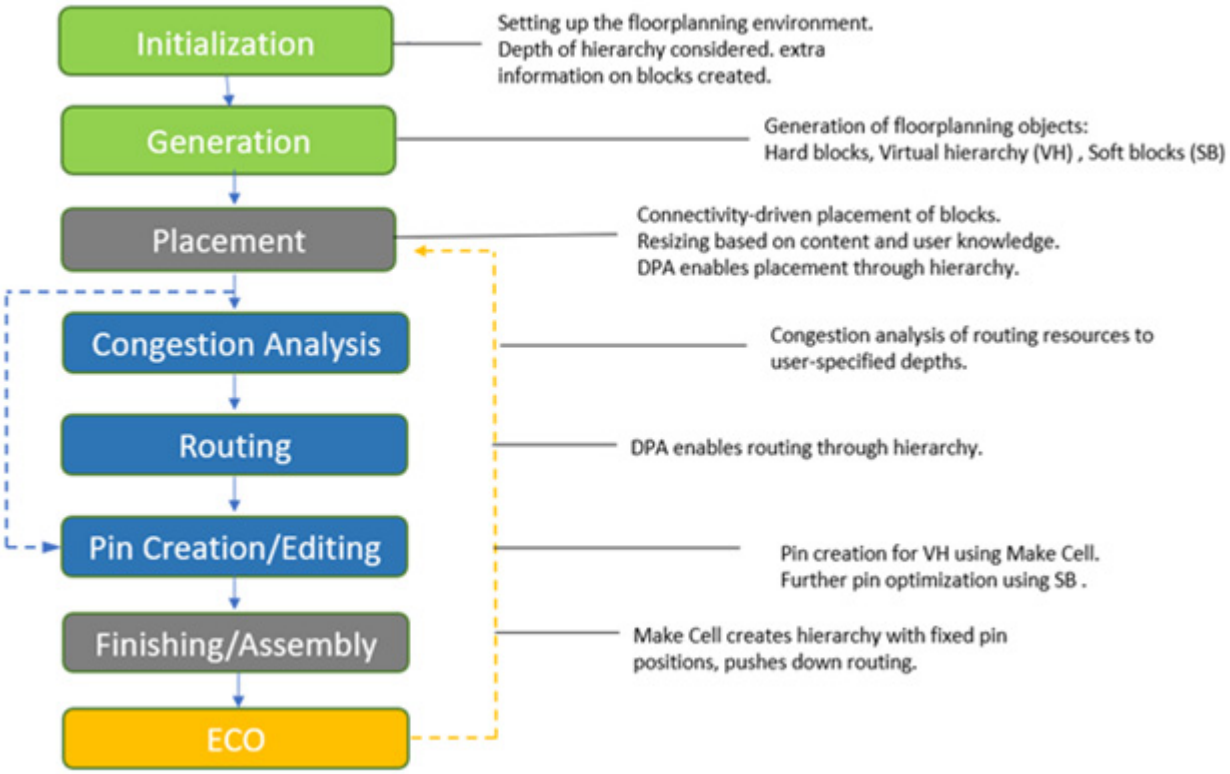
Design Planning Workspace

2

Design Planning and Analysis Flow

The Virtuoso Design Planning and Analysis (DPA) 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 Planing and Analysis flow is illustrated below.



In the Design Planning and Analysis flow, you can seamlessly layout, place, and route, backed by real routing data.

The key steps in the Design Planning and Analysis flow are:

- **Generation:** DPA supports full hierarchical generation capabilities, which allow you to effectively plan the connectivity- and constraint-riven flows in Virtuoso Layout Suite EXL. The DPA flow uses the existing schematic hierarchy, where available, to create a virtual hierarchy that can be realized, when appropriate.
- **Placement Planning:** With the DPA tool, you have the ability to plan, visualize, and edit through the hierarchy—improving manual, assisted, and automatic placement. Blocks that have an existing IP in place and soft blocks from an existing Virtuoso Floorplanner flow are both recognized by the DPA flow as valid inputs for design planning. Making decisions based on the context of the full design allows for less expensive iterations, resulting in substantial productivity gains.
- **Pin Planning:** Not only does the DPA tool give you the advantage of hierarchical pin planning and optimization, it also enables you to perform congestion-aware pin placement through the hierarchy.
- **Global Route Planning and Congestion Analysis:** In the DPA flow, you can run congestion analysis earlier in the design lifecycle, allowing you to visualize available routing resources in your floorplan, saving expensive iterations. Additionally, this can help you clear congested areas, leading to better convergence and reduced floorplan area.

Related Topics

Layout Component Generation in Design Planning and Analysis

Placement of a Virtual Hierarchy

Congestion Analysis and Global Routing Support in DPA

Advantages of Design Planning and Analysis

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 (DPA) tool offers a more reliable methodology for efficient layout planning at the top, block, and cell level. DPA 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.

Related Topics

Design Planning and Analysis Flow

Virtual Hierarchy Generation

Physical Hierarchy Generation

Layout Component Generation in Design Planning and Analysis

When you have the Design Planning workspace enabled, you can 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. Choosing either of these layout generation options sets the `lxGenerateInBoundary` environment variable to `t` to ensure the layout representations are generated within the design boundary.

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 uses the source to create the corresponding layout representations.

- If a hard block exists, it is used to generate 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 hierarchy is used to create a virtual hierarchy that entirely matches the schematic. See Virtual Hierarchy Generation.
- If the schematic view 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 a Soft Block 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. See Generating All Components From Source.

Selected Component Generation in DPA

Use the *Generate All 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, which is snapped to the width spacing pattern (WSP) grid. The snapping is controlled by the *Snap Pattern Snapping* option on the Layout Editor Options form.

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.

Related Topics

Generating Components As In Schematic

lxGenerateInBoundary

Virtual Hierarchy Generation

Physical Hierarchy Generation

Design Planning Toolbar

Layout Editor Options Form

Virtual Hierarchy Generation

Virtual layout hierarchy can 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 *Virtual Hierarchy* option on the Generate Layout form to generate a virtual hierarchy. The position of the instances in the generated virtual hierarchy is controlled using the *Position* field on the Design Planning and Analysis Options form. By default, individual instances are positioned inside each generated virtual hierarchy. You can group individual instances and position them inside generated virtual hierarchies using the *grouped by type* or *grouped by type within virtual hierarchies* option. The instance grouping feature is also supported at the top cellview level.

During layout generation, virtual groupings of layout instances are created in the following order:

1. Devices with the same component type. If no component type is defined, devices of the same cell master.
2. For standard cells only, devices with the same net set pair (power domain).
3. Devices of the same size based on bounding box dimensions.
4. Devices:

   a. of the same mfactor (multiplicity)

   b. of the same vector

   c. single devices
5. Non-standard cell single devices with the same bulk terminal connection named `b` or `B` or specified through component types.

   a. Layout devices with bulk terminal

   b. Schematic devices with bulk terminal, if no terminal is found on layout devices

The generated virtual hierarchy can easily be accessed using the *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.

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 `t`.

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.

Related Topics

transparentGroup

Generating a Virtual Hierarchy

Virtual Hierarchy Data Sets in Navigator Assistant

Design Planning and Analysis Options Form

Accessing a Virtual Hierarchy Clone

Soft Block Generation in Design Planning and Analysis

When the schematic components in a design have missing layout representations, the Design Planning and Analysis (DPA) tool generates a virtual hierarchy at the top level that represents the hierarchy in the layout. When the schematic counterparts themselves are missing, the DPA tool can create soft blocks for such schematic representations during virtual hierarchy generation.

For top-level virtual hierarchy blocks that have missing schematic or a schematic with only pins but no instances or physical binding, the DPA tool 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, DPA 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

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. See Virtual Hierarchy Data Sets in Navigator Assistant.

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

To control the area of a soft block at generation, you can set an area property on the schematic symbol. If the soft block is generated using this area property, the specified area is then reported as the initial soft block area and is used to measure any area changes during a Stretch or Chop operation.

Related Topics

Generating a Soft Block for a Virtual Hierarchy

Physical Hierarchy Generation

Generate Physical Hierarchy generates the physical hierarchy based on the configuration defined in the Configure Physical Hierarchy (CPH) window. It allows you to generate a physical hierarchy from the logical hierarchy in a schematic. It also creates soft layouts and soft abstracts in the layout canvas corresponding to the configuration view generated by CPH.

Choose *Plan – Generate – Generate Physical Hierarchy* to display the Generate Physical Hierarchy form. In this form, you can select the objects to be generated when the physical hierarchy is generated.

After generating the physical hierarchy, a summary report of the soft blocks generated by Generate Physical Hierarchy is displayed in CIW.

Automatic Physical Hierarchy Generation

The *Plan – Generate – Auto Generate Hierarchy* command enables you to quickly generate a physical hierarchy by applying common parameters. You can configure, specify bindings, and generate top blocks and soft blocks in the layout canvas. You can also use the command to define common boundary and pin parameters for the blocks you want to generate. In addition, the command enables you to specify the shape and size of the boundary that is created when the soft block is generated.

Related Topics

Configure Physical Hierarchy

Auto-Generate Hierarchy

Connectivity-Driven Layout Editing Commands Supported by DPA

For optimal planning of virtual hierarchies in a layout design, Design Planning and Analysis (DPA) supports several Virtuoso Layout Suite commands.

Command	Use to...	Related
Generate All From Source	Generate layout counterparts for soft and hard blocks. For schematic instances that have no layout representations available, generate a virtual hierarchy with a selectable area boundary that is snapped to the width spacing patterns (WSP) grid. Generate soft blocks for schematic components that are missing. Use the *Snap Pattern Snapping* option on the Layout Editor Options form to control snapping of the generated objects.	Generate Layout Form Virtual Hierarchy Generation Generating a Soft Block for a Virtual Hierarchy

Command	Use to...	Related
Generate Selected From Source	Generate a virtual hierarchy for the selected schematic instances that have no layout representations. The generated virtual hierarchy has a selectable area boundary that is snapped to the width spacing patterns (WSP) grid. Use the *Snap Pattern Snapping* option on the Layout Editor Options form to control the snapping.	Selected Component Generation in DPA Generating Components As In Schematic
Edit In Place	Edit in place the generated virtual hierarchy clones. To edit the virtual hierarchies that are not clone instances, adjust the display depth to make them transparent.	Editing In Place Generating Transparent Layout Hierarchy
Check Against Source	Check and report 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*.	Check Against Source Form
Update Components And Nets	Update the components from source and if the selected set that has missing instances contains a virtual hierarchy group, generate the missing components inside the virtual hierarchy group. In addition, preserve the area boundary of the instances inside the virtual hierarchy group and generate soft blocks for schematic components that are missing.	Update Components And Nets Form

Command	Use to...	Related
Generate Chained Devices	Generate device chains for instances inside a virtual hierarchy when the selection is made at the top level.	Generate Chained Devices Form
Interactive Chaining	Chain devices inside a virtual hierarchy when the device selection is made from the top level.	Chaining Devices Interactively
Interactive Abutment	Abut devices inside a virtual hierarchy when the devices are selected from the top level.	Device Abutment
Interactive Folding	Fold devices inside a virtual hierarchy when the device selection is made from the top level.	Transistor Folding

Related Topics

Layout Editor Options

Virtual Group Creation in Design Planning and Analysis

A virtual group enables you to get an opaque or a transparent visualization of the design hierarchy. You can create a virtual figGroup manually using the *Generate Layout* command or use the `lxHiCreateVirtGroup` SKILL function.

Creating a virtual figGroup enables you to edit the hierarchical instances in the figGroup directly at the top level, not requiring an *Edit In Place*. However, if the virtual figGroup contains a clone, an *Edit In Place* is needed to edit the clones.

Design Planning and Analysis (DPA) enables you to create a virtual hierarchy group that does not match the schematic hierarchy.

If the selected instances are bound to a common parent schematic instance in the same design, the virtual group is created as a *generated* virtual group.

If no connectivity reference is available, the *Create Virtual Group* command does not check if the selected instances match the schematic hierarchy, it directly adds the selected instances to a *created* virtual group.

If the virtual group is created using the `lxHiCreateVirtGroup` SKILL function and the `vhSelectiveMode` environment variable is set to `t`, the SKILL function also creates virtual figGroups using selected layout devices that are bound to schematic instances from different designs.

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.

Related Topics

lxHiCreateVirtGroup

vhSelectiveMode

Create Virtual Group Form

Virtual Hierarchy Boundary Adjustment

The virtual hierarchies you create using the Design Planning and Analysis tool can have a rectangular or a rectilinear area boundary. You can choose the appropriate boundary type to be created using the Adjust Boundary form. The DPA tool then automatically *creates* or resizes the area boundary of the selected virtual hierarchy or the PR boundary of the selected soft block. The tool also snaps the boundaries to the width spacing pattern (WSP) grid and this snapping of the area boundary to the width spacing pattern grid is controlled by the *Snap Pattern Snapping* option on the Layout Editor Options form.

You can also register an area estimation function to specify the area boundary of a virtual hierarchy. Alternatively, you can stretch, chop, or reshape the boundary of a virtual hierarchy or a soft block to resize it.

The area boundary of a virtual hierarchy can also resize *automatically* to accommodate any instances that are moved outside the 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.

If the virtual hierarchy being stretched, chopped, or reshaped is set to placement status *None* and the virtual hierarchy contains a row region that is selectable, the row region is also stretched to match the area boundary. When created, the row region is selection-protected, which means if the area boundary is stretched, chopped, or reshaped, the contained row region is not stretched to match the virtual hierarchy area boundary. To independently control the size of the row region, you must remove its selection protection by using the *Clear Selection Protection* command. Standard and custom rows, if attached to the area boundary, are automatically resized to fit the adjusted 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.

Related Topics

Adjust Boundary Form

lxHiAdjustBoundary

Moving Instances Outside a Virtual Hierarchy

Adding Instances to a Virtual Group

Area Estimation Framework in Floorplanner

Layout Editor Options

Make Virtual Hierarchy Command in Design Planning and Analysis

The *Make Virtual Hierarchy* command in Design Planning and Analysis (DPA) supports concurrent layout designing by allowing creation of 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 for generation. 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.

The *Make Virtual Hierarchy* command can also generate missing components in the virtual hierarchy for a soft block that previously only had pins but now also has a schematic available. Existing pin locations and the area boundary size of the soft block are retained. The *Make Virtual Hierarchy* command can be used to integrate multiple selected cells at a time, also supporting m-factored devices. Black box soft blocks are not supported but softMacro soft blocks are supported.

For virtual hierarchies with clones, the *Make Virtual Hierarchy* command always generates new clones instead of adding members to existing clone families. 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`.

Related Topics

Make Virtual Hierarchy Form

Make Cell Form

Update Components And Nets Form

Make Cell Command in Design Planning and Analysis

The *Make Cell* command creates real cellviews from virtual hierarchies that are generated or manually created. During a *Make Cell* run, selected, generated virtual hierarchies are replaced with instances of new cellviews and bound using the same schematic correspondence as the replaced virtual hierarchies. For manually created virtual hierarchies, the schematic correspondence does not exist so these virtual hierarchies are replaced with transparent instances of new cellviews, and the instances within the made cellview are bound to the corresponding schematic instances. When the *Make Cell* command is run hierarchically, it uses a bottom-up approach to also create a layout view for each level of the virtual hierarchy inside the selected virtual hierarchy.

Constraints member types `instances`, `nets`, `figGroups`, `clusters`, `pins`, and `boundary` that are solely contained within the selected virtual hierarchy are transferred to the made cellview. Constraints that contain any other member types are not transferred. In addition to transferring constraints, the *Make Cell* command honors all net attributes, propagating the power domain information and the min/max net voltage to the made cellview.

If the made cellview has no area boundary, the command also generates a PR Boundary for the made cells using the current top-level location. If the generated PR boundary does not by default snap to the nearest grid, you can set the `makeCellGridAlignment` environment variable to `pushGrid` to snap the generated PR boundary to the nearest grid.

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.

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 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 is created. See Virtuoso Module Generator.

Design Scenarios Not Supported by the Make Cell Command

Although some differences between the schematic and layout hierarchies are ignored during a *Make Cell* run for generated hierarchies, the command does not support the following cases. No such limitations exist when running the command on virtual hierarchies that are manually created.

- 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, you can 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 *Make Cell* command on generated virtual hierarchies may not help.

Related Topics

Make Cell Form

makeCellPinsBelow

Update Components And Nets Form

Remaster Command in Design Planning and Analysis

The *Remaster* command in Design Planning and Analysis (DPA) 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, you can 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 command may not help.

Related Topics

Remaster Form

Placement of a Virtual Hierarchy

When a virtual hierarchy is generated, the Design Planning and Analysis (DPA) tool creates an area boundary around each top-level virtual hierarchy. The size of this area boundary is based on the actual size of the layout views present inside the hierarchy rather than the estimated sizes. This allows for a realistic placement of the blocks at the top level. DPA supports the following two types of placements for a virtual hierarchy:

- Manual Placement

- Automatic Placement

Manual Placement

Manual placement can be achieved by stretching, resizing, or chopping the area boundary of a virtual hierarchy to change the aspect ratio of the virtual hierarchy. Manual placement can help optimize the floorplan across the hierarchical levels.

The DPA tool also lets you use the *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 manually placing the top-level objects.

Automatic Placement

Automatic placement is performed by the automatic placer to perform an analog, *Like Schematic* placement after a *Stretch*, *Reshape*, or *Chop* operation. The automatic placer is useful when a manual adjustment of the virtual hierarchy has resulted in some of its contents being placed outside the area boundary. The automatic placer can switch between placement modes, depending on the design conditions.

By default, the automatic placer places only those instances that are currently outside the area boundary. The automatic placer does not run when all the virtual hierarchy components already fit inside the area boundary or the area boundary is too small to accommodate the area of the contents. To control the number of devices that can be placed in this mode, you can use the `autoPlaceLimit` environment variable. The devices can be placed with a minimum spacing value of `0` between them. This placement mode is suitable for designs that have constraints. Virtual hierarchies that have a `rowRegion` defined are also supported by this placement mode.

The automatic placer runs a full placement when the `autoPlaceAllInstances` environment variable is set during interactive edits to the area boundary or when the *Auto Place* context-sensitive menu command is used. In this mode, the placer can support a larger limit on the number of devices that can be placed, provided the virtual hierarchy area boundary is rectangular, does not contain a `rowRegion`, and the design does not contain any constraints. The placement limit in this mode can be controlled using the `autoPositionLimit` environment variable and the minimum spacing between the devices

can be controlled using the `autoPlaceMinSep` environment variable. If any of the placement conditions is not met, the placement limit is controlled using the `autoPlaceLimit` environment variable.

Note: For rectangular area boundaries, if the area boundary bounding box is same as the bounding box of the virtual hierarchy, automatic placement is not triggered.

Related Topics

Auto Placement

autoPlaceLimit

autoPlaceAllInstances

autoPositionLimit

autoPlaceMinSep

Editing Objects

Analyze Connectivity

Congestion Analysis and Global Routing Support in DPA

Creation of routes is required for the interface nets of a virtual hierarchy to allow pins to be created on the boundary using the Make Cell form. The Design Planning and Analysis (DPA) tool supports an in-built global router to automatically route all or selected nets or to manually route selected critical nets. The tool also lets you 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 DPA, see Running Congestion Analysis.

Related Topics

Make Cell Form

Generating All Components From Source

Editing In Place

Check Against Source

Update Components And Nets

Analyze Connectivity

Running the Auto Placer

3

Working with Virtual Hierarchies

Generating a Virtual Hierarchy

For designs that have no hard or soft blocks available, the *Generate All From Source* command uses the schematic hierarchy to create a virtual hierarchy that entirely matches the schematic. To generate a virtual hierarchy:

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

   The Design Planner 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. Choose the required virtual hierarchy generation options.

4. Click *OK*.

   The Design Planning and Analysis tool generates the layout components from scratch for the selected schematic source.

   For the schematic instances that have no soft or hard block, the Design Planning and Analysis tool generates a virtual hierarchy, which can be accessed using the Navigator assistant. For more information, see Virtual Hierarchy Data Sets in Navigator Assistant.

   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 Planning and Analysis tool generates soft blocks when the virtual hierarchy is generated.

Note: 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

Design Planning Toolbar

Generating a Soft Block for a Virtual Hierarchy

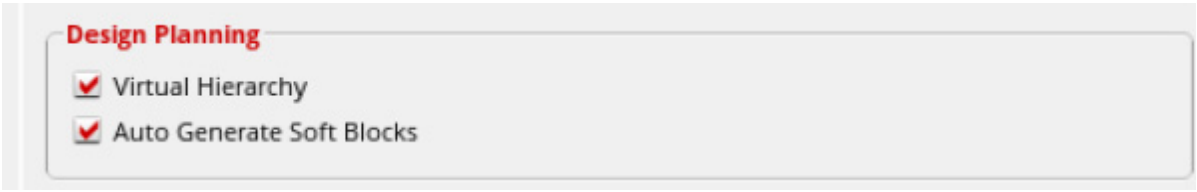
For symbol blocks that have missing schematic or a schematic that has pins but no instances or physical binding, the Design Planning and Analysis tool generates a soft block. To generate a soft block for a virtual hierarchy:

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

   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 Planning and Analysis tool generates layout components from scratch for the selected schematic source.

Related Topics

Virtual Hierarchy Generation

Generate Layout Form

generateVirtualHierarchy

generateSoftBlocks

areaBoundaryEnclosure

initUtilization

Generating All Components From Source

Virtual Hierarchy Data Sets in Navigator Assistant

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

- *Virtual Hierarchy*

  See Accessing a Virtual Hierarchy.

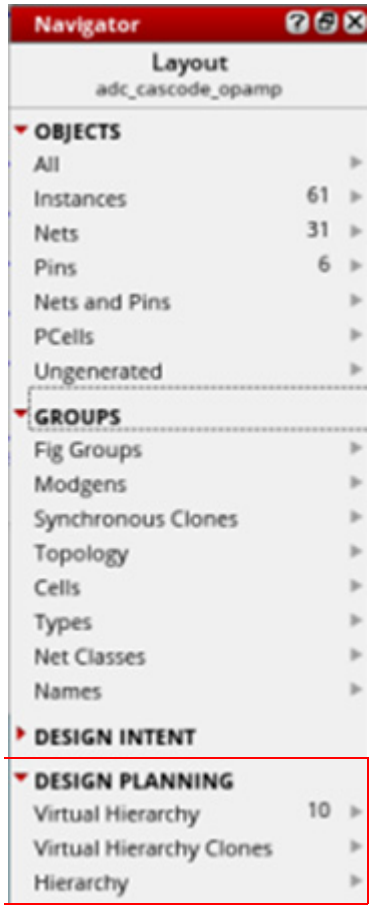
- *Virtual Hierarchy Clones*

  See Accessing a Virtual Hierarchy Clone.

- *Hierarchy*

  See Accessing a Hierarchical Object.

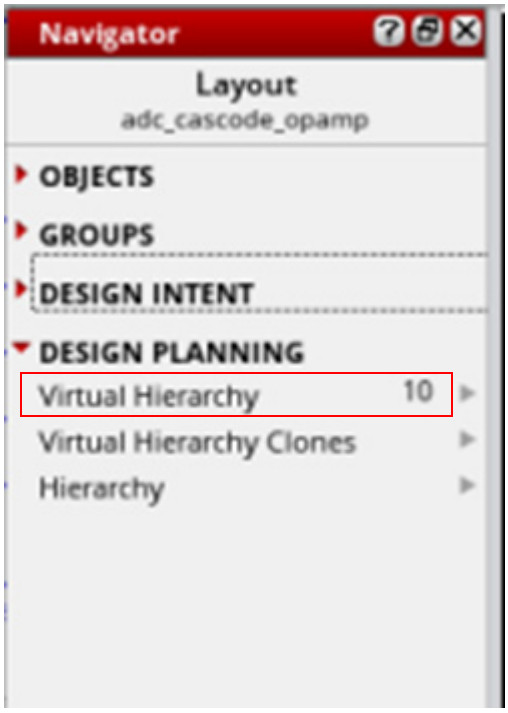
Each of these predefined data sets, as displayed, is available under the *DESIGN PLANNING* category.



Accessing a Virtual Hierarchy

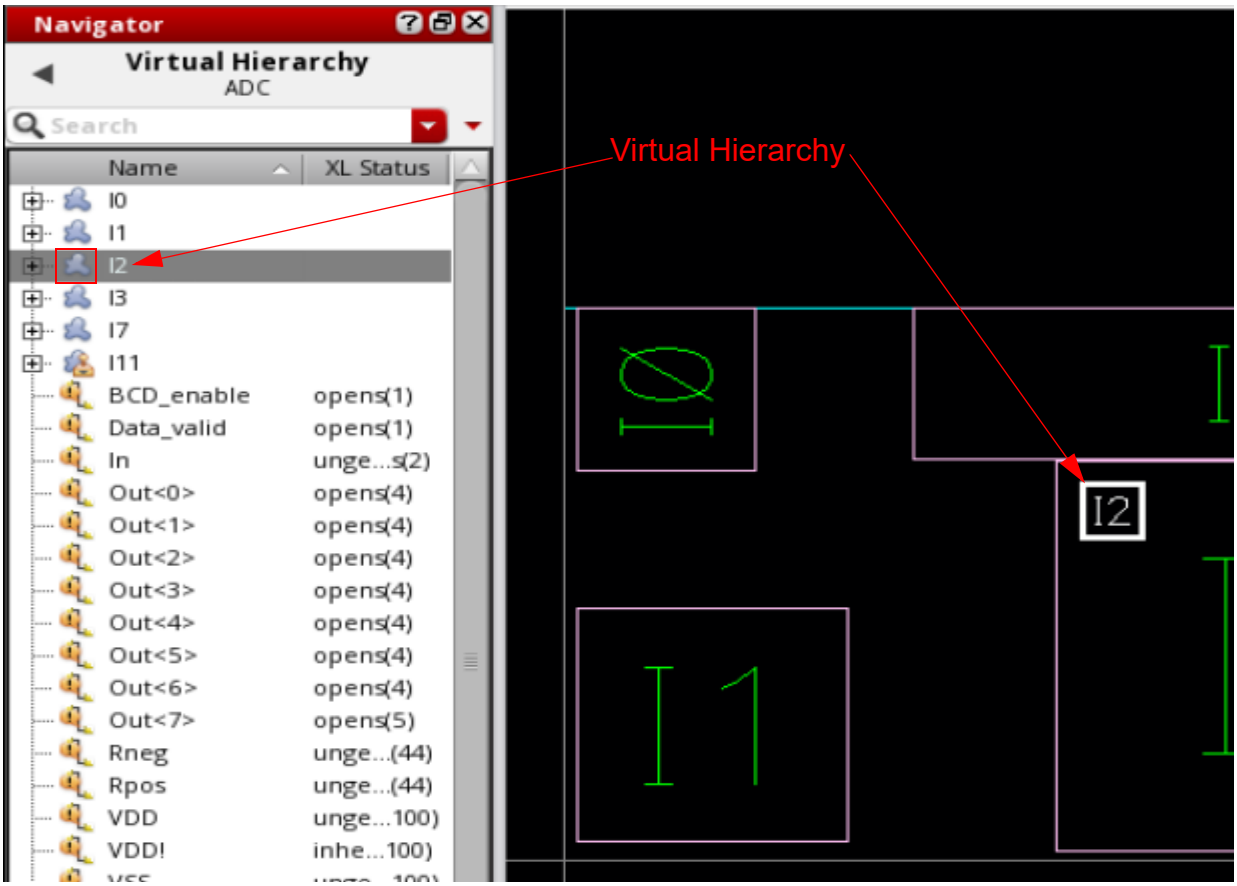
To access the generated virtual hierarchies and interface nets in the layout canvas:

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



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



2. Click a virtual hierarchy in the Navigator tree.

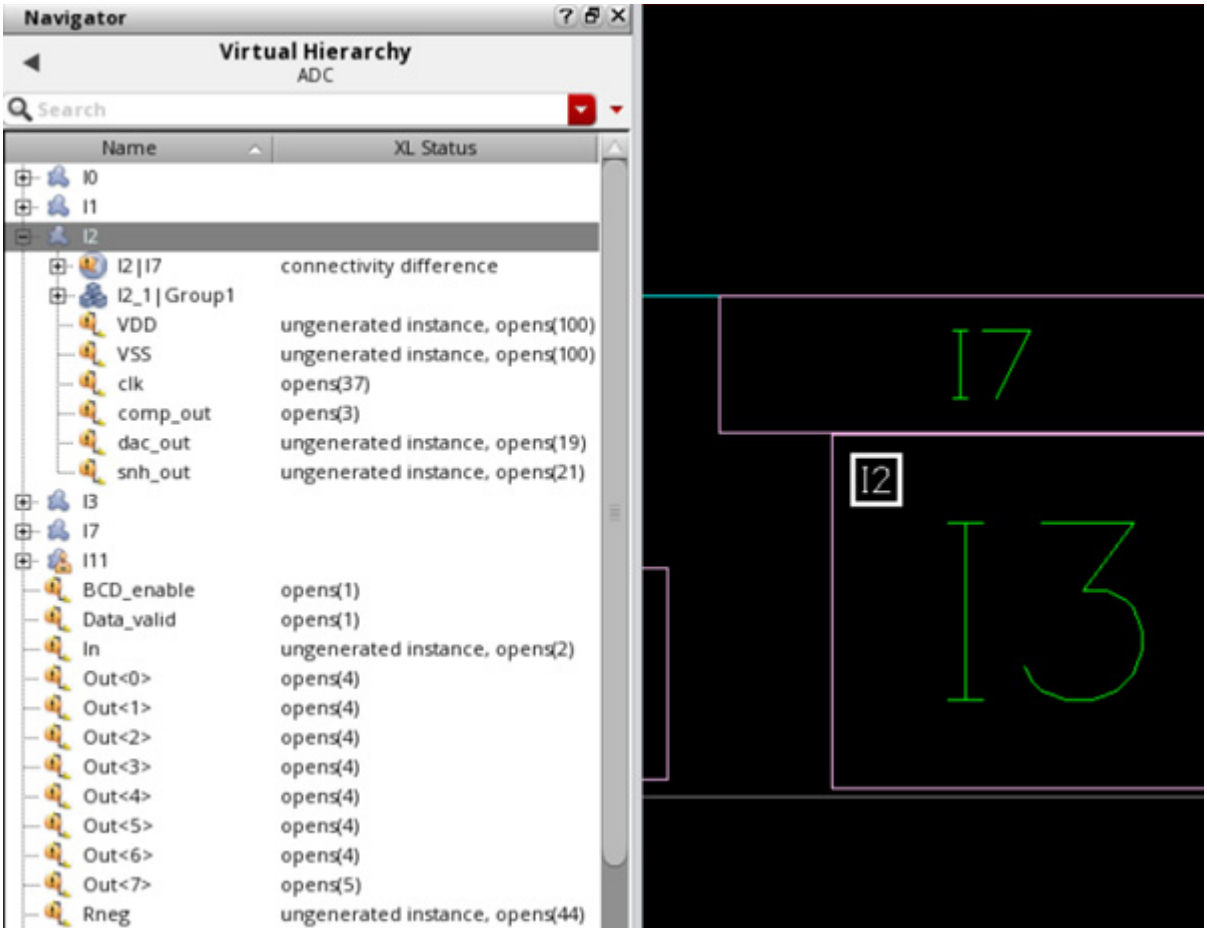
   The corresponding object is selected in the layout canvas. The *Show Selection Info* toolbar updates to display appropriate information related to the selected virtual group, such as display name, type of virtual group—*Generated*, *Clone*, or *Created*, placement status, and the display stop level value.

   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. 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 addition, the Navigator assistant displays the XL Status of the virtual hierarchy components.



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 displays flight lines on the canvas. If the virtual hierarchy you move is opaque, which means it has no contents, no flight line is generated for the virtual hierarchy.

4. 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, Virtuoso Layout Suite EXL creates a probe highlighting all the associated instances and nets in the layout canvas, as displayed.

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.

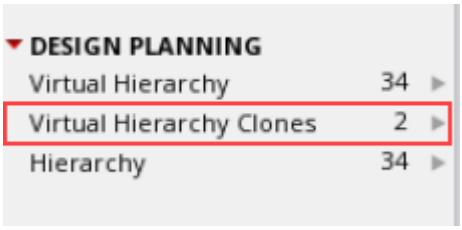
5. Right-click a virtual hierarchy in the Navigator tree to perform the operations supported through the shortcut menu.

   The layout canvas updates in accordance with the selected operation.

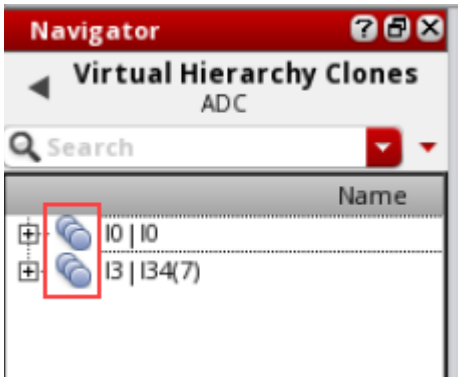
Accessing a Virtual Hierarchy Clone

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.



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. The corresponding virtual hierarchy clones are represented using the ( ) icon.



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

   A virtual hierarchy clone supports three additional commands in the shortcut menu:

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

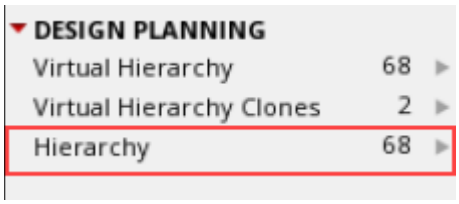
   The layout canvas updates in accordance to the selected virtual hierarchy clone operation.

Accessing a Hierarchical Object

The hierarchical objects associated with a 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

Virtual Hierarchy Editing Commands in Shortcut Menu

Virtual Hierarchy Data Sets in Navigator Assistant

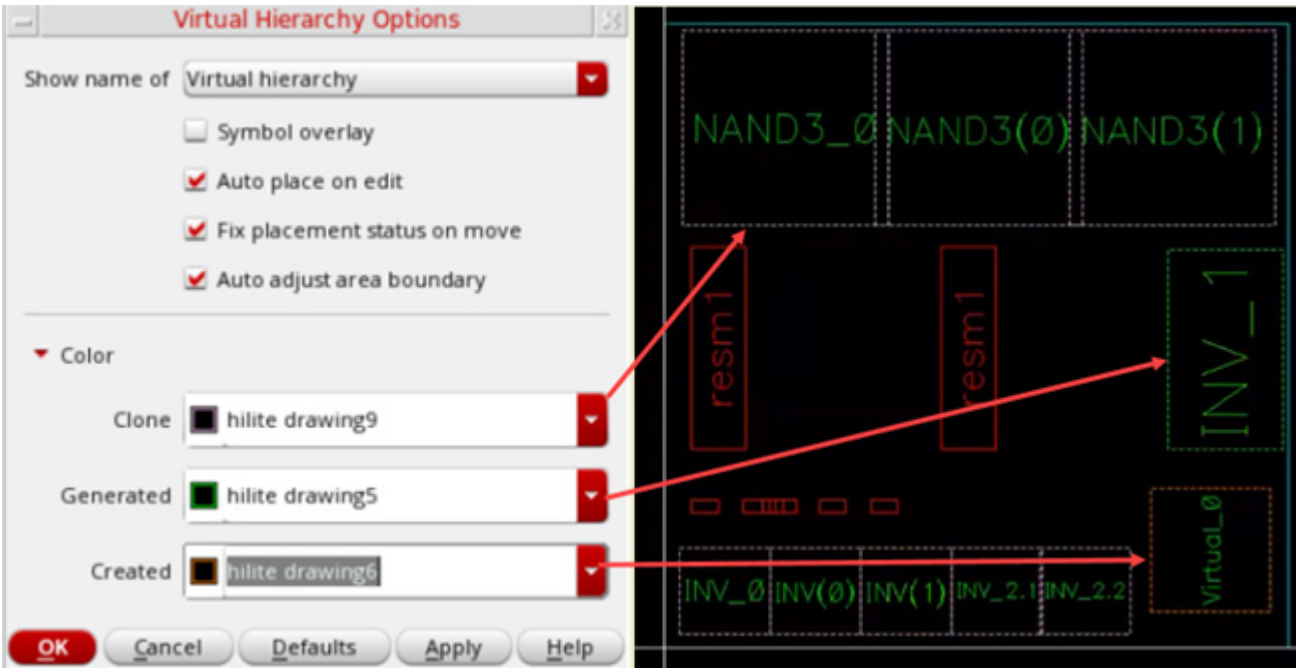
Navigator Assistant

Types of Virtual Hierarchies

Virtual hierarchies can be *created*, *cloned*, or *generated*. This means a layout in Virtuoso Layout Suite 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 on 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.



Related Topics

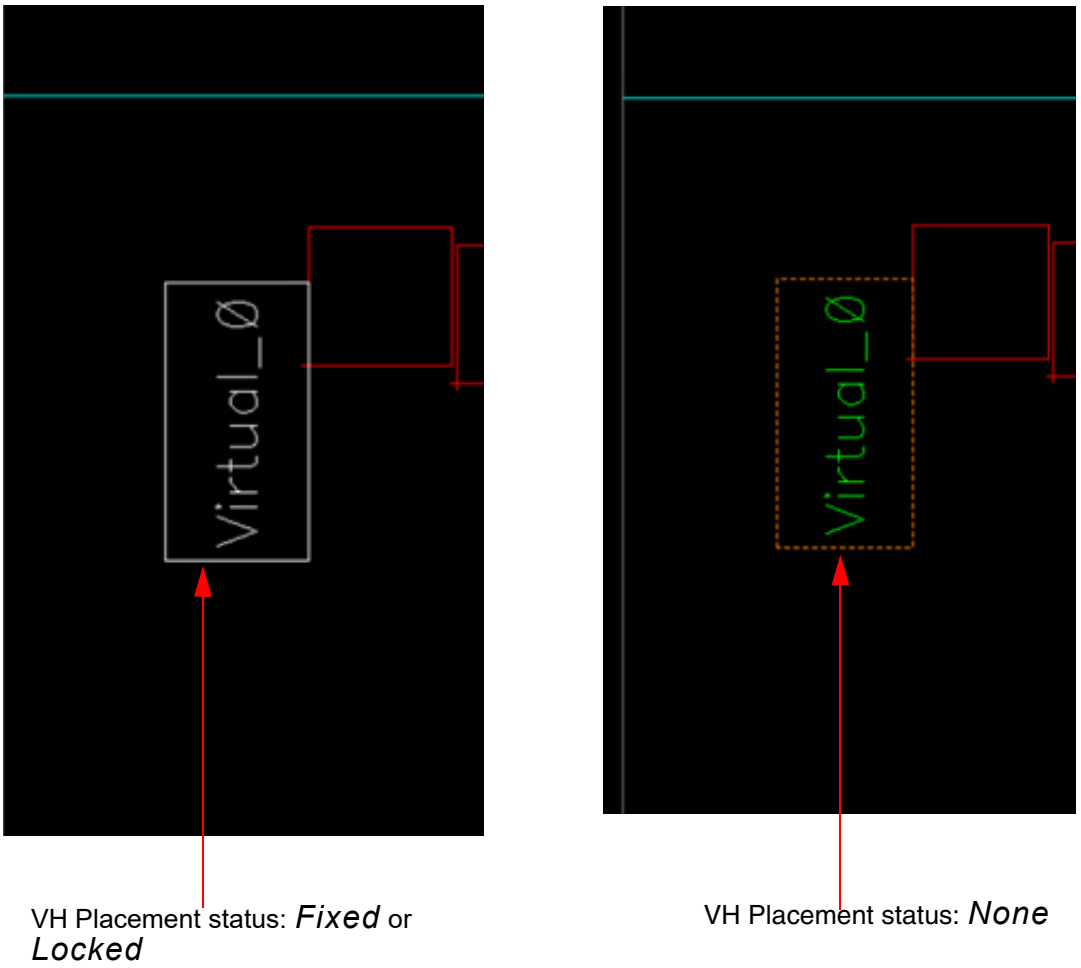
Design Planning and Analysis Options Form

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 displayed, the placement status of the virtual hierarchy is *Fixed* or *Locked*.

- If the bounding box of a virtual hierarchy is a dashed line, as displayed, the placement status of the virtual hierarchy is *None*.



Related Topics

dashedLinePlacementStatusNone

Virtual Hierarchy Display Controls

By default, when a virtual hierarchy is generated, the Design Planning and Analysis (DPA) tool 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.

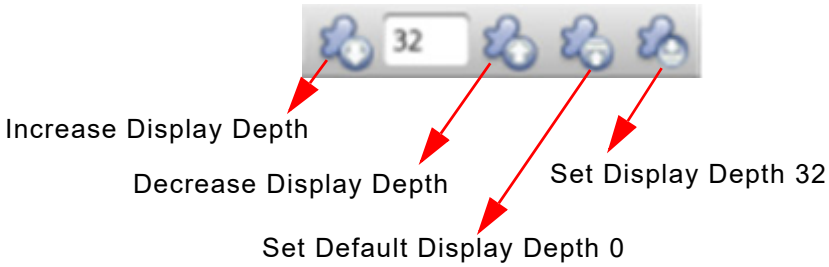
Because the top-cell layout of a virtual hierarchy is flat, setting the display control enables you to show connectivity to the lower-levels instances or the virtual hierarchy at a lower level, which is not possible when working with real cellviews. This access to the connectivity information for lower-level 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 DPA tool 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 the Virtual Hierarchy Overrides.

The DPA tool provides two methods for you to enable display controls for better layout design planning:

- **Display Depth Options**

  The Design Planner toolbar provides the following 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 0*

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

  - *Set Display Depth 32*

Removes all the stop level overrides, sets the display depth to `32`, 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**

  You can use the *Analyze Connectivity* command 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. You can also 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 use the command to display the common net connectivity, select connected nets, and create the net sets in the Navigator assistant.

Related Topics

Analyze Connectivity Form

Setting Display Depth in Design Planning and Analysis

Design Planning Toolbar

Setting Display Depth in Design Planning and Analysis

The Design Planning and Analysis tool enables you to control the display depth only for the selected components, or for all the components in the design.

To increase (or decrease) the display depth for the selected components:

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

If no components are selected before the display depth is increased or decreased, the display depth is changed for all the components in the design.

Related Topics

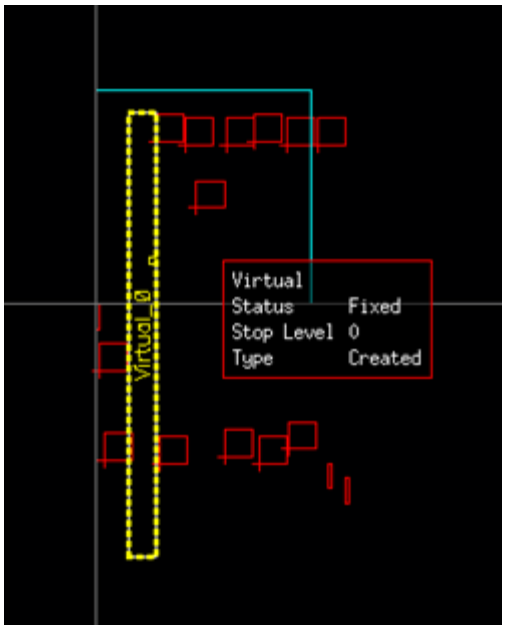
Virtual Hierarchy Display Controls

Design Planning Toolbar

Viewing the Virtual Hierarchy Overrides

Viewing the 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 displayed, when the cursor is hovered over a virtual hierarchy.



Related Topics

Dynamic Display Measurements for a Virtual Hierarchy

Virtual Hierarchy Display Controls

Design Planning Toolbar

Display Options

Analyze Connectivity Form

Navigator

Automatic Adjustment of a Virtual Hierarchy Area Boundary

The Design Planning and Analysis tool also automatically adjusts the area boundary of a virtual hierarchy when:

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

The resized boundary is always rectangular when automatically adjusted.

Moving Instances Outside a Virtual Hierarchy

The Design Planning and Analysis tool 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 Planning and Analysis tool 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 Planning and Analysis 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 adjusts 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.
   - Non-rectangular area boundaries are not automatically adjusted.

Adding Instances to a Virtual Group

The Design Planning and Analysis tool 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.

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`.

Related Topics

Design Planning and Analysis Options Form

Create Virtual Group Form

autoAdjustBoundary

Virtual Pin Snapping

The Design Planning and Analysis tool supports snapping of virtual pins to the nearest area boundary edge when the virtual pin is moved. The pin snapping works for both top-level PR Boundary and soft blocks when Level-1 editing is enabled.

For virtual pins to be snapped, the placement status of the associated virtual hierarchy must be set to *None* and the environment variables `snapToGrid` and `snapToBoundary` should both be set to true. The virtual pins are also snapped to the nearest area boundary edge when adjusting the area boundary of a virtual hierarchy using the *Adjust Boundary*, *Stretch*, *Chop*, or *Reshape* command. This happens regardless of the value of the `snapToGrid` and `snapToBoundary` environment variables but the placement status of the associated virtual hierarchy must be set to *None*.

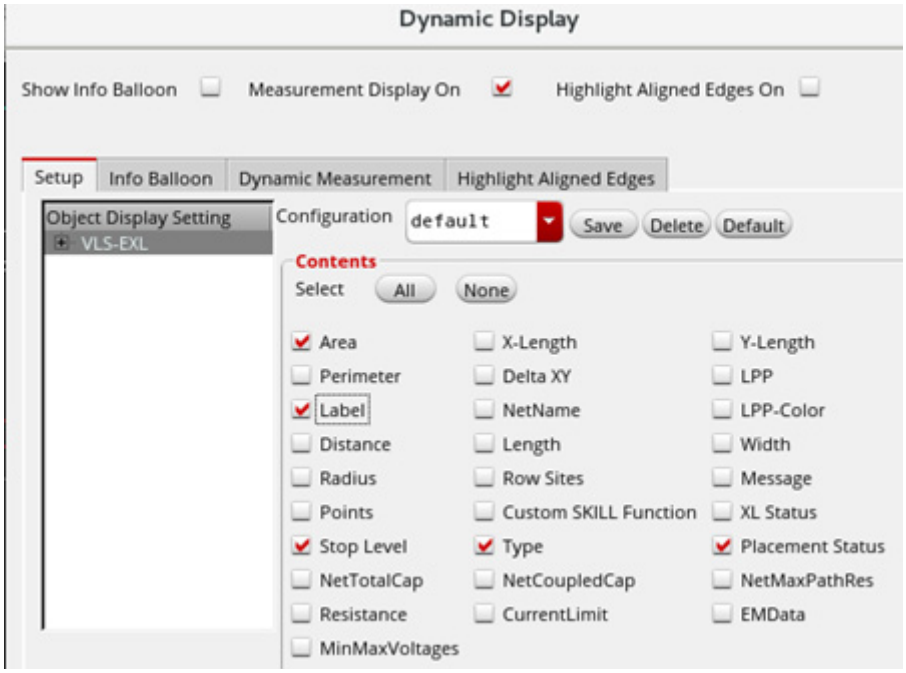
Related Topics

Setting Up Snap Grid

Dynamic Display Measurements for a Virtual Hierarchy

When manually adjusting the area boundary of a virtual hierarchy, you can select the *Measurement Display On* option on the *Dynamic Display* form to dynamically display

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 the 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, are displayed dynamically on the layout canvas as you perform the *Stretch* command.

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.

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.

Related Topics

Dynamic Measurement Parameters for Virtual Objects

Dynamic Display Form

Stretch Form

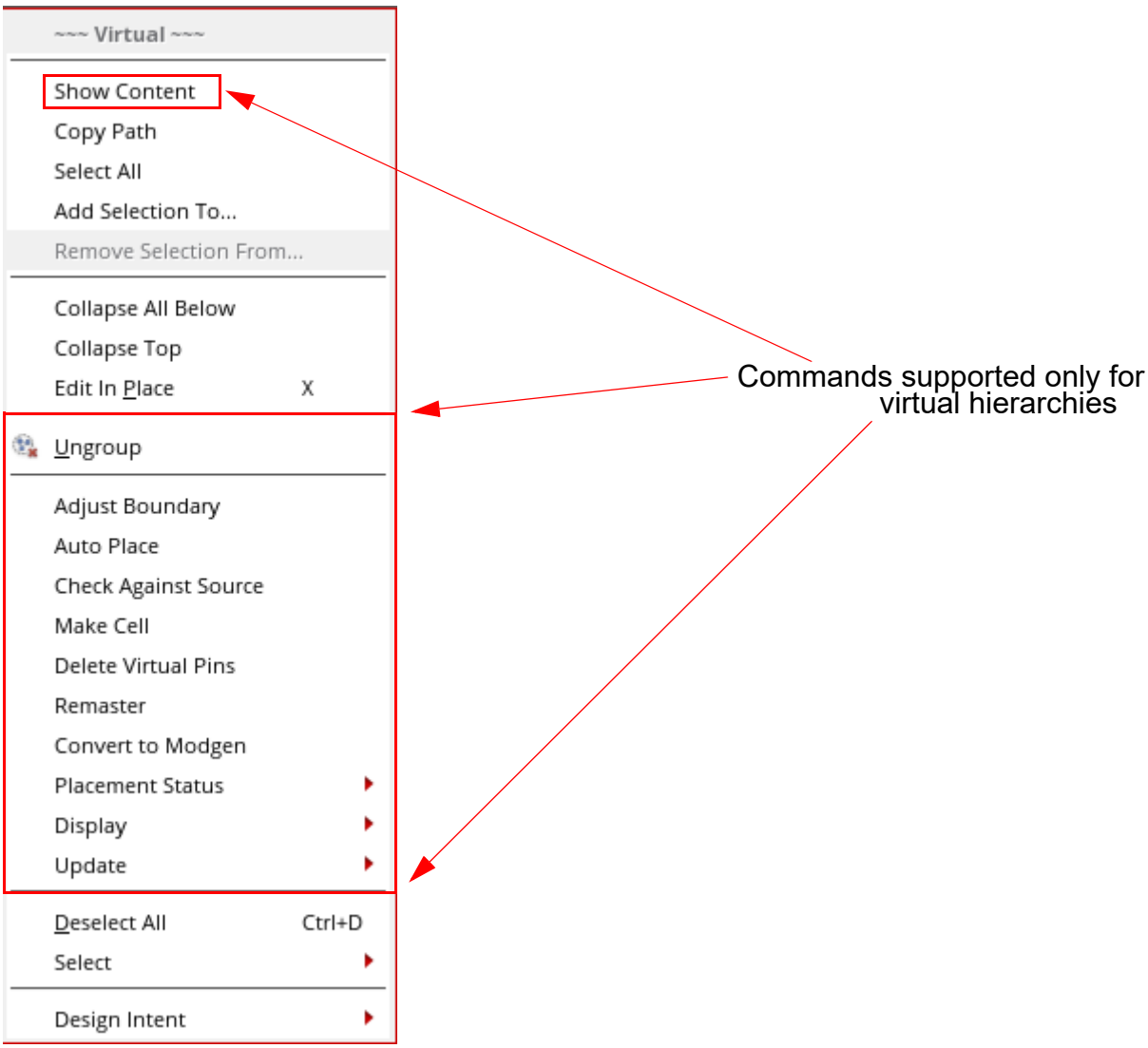
Virtual Hierarchy Editing Commands in Shortcut Menu

The Design Planning and Analysis tool provides quick access to some useful commands through the shortcut menu of a virtual hierarchy and a virtual hierarchy clone.

To access the shortcut menu of a virtual hierarchy:

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

The *Virtual* shortcut menu is displayed. The shortcut menu provides some generic commands supported in Virtuoso Layout Suite EXL, in addition to a set of commands that are supported only for virtual hierarchies.



Virtual Hierarchy Shortcut Command Menu Table

The table below details the shortcut menu commands that are supported for virtual hierarchies and virtual hierarchy clones. When invoked from the Navigator assistant, the *Virtual* shortcut menu displays an additional command, *Show Content*. The shortcut menu

of a virtual hierarchy clone also supports a few additional commands, irrespective of how the menu is invoked.

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 level. See Adjust Boundary Form.
Auto Place	■ Virtual hierarchies	Automatically places all instances inside an area boundary. See Automatic Placement.

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 Form.
Delete Virtual Pins	■ Created virtual hierarchies ■ Created virtual hierarchy clones	Deletes all virtual pins from selected virtual hierarchies. See `lxDeleteVirtualPins`.
Remaster	■ Virtual hierarchies ■ Virtual hierarchy clones	Replaces the selected virtual hierarchy with the layout master that exists on disk. See Remaster Form.
Convert to Modgen	■ Created virtual hierarchies only	Converts the selected virtual hierarchies to Modgens. See Converting Created Virtual Hierarchies to Modgens

Command name	Supported for...	Use to...
Placement Status	■ Virtual hierarchies ■ Virtual hierarchy clones	Set the *Placement Status* of the selected virtual hierarchy and virtual hierarchy clones to *Fixed*, *Locked*, or *None.* *Fixed* and *Locked* maintain the original placement of the selected virtual hierarchies and virtual hierarchy clones. *None* keeps the contents of the virtual hierarchy flexible for placement. See Make Virtual Hierarchy Form.

Command name	Supported for...	Use to...
Display	■ Virtual hierarchies ■ Virtual hierarchy clones ■ Instances inside virtual hierarchies	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 Virtual Hierarchy Display Controls.* Choose *Highlight* to highlight the selected virtual group contents or the selected instances inside the virtual group in the chosen color. Choose *Unhighlight* to remove the color highlights. **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 Virtual Hierarchy Recloning.
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.

Related Topics

Virtual Hierarchy Editing Commands in Shortcut Menu

Advantages of Design Planning and Analysis

Editing Virtual Hierarchy Clones

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.

Related Topics

Connectivity-Driven Layout Editing Commands Supported by DPA

Virtual Hierarchy Generation

Virtual Hierarchy Recloning

Virtual Hierarchy Clones Containing Modgens

Virtual Hierarchy Recloning

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.

Related Topics

Connectivity-Driven Layout Editing Commands Supported by DPA

Virtual Hierarchy Generation

Editing Virtual Hierarchy Clones

Virtual Hierarchy Clones Containing Modgens

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 assistant.

Related Topics

Editing Virtual Hierarchy Clones

Virtual Hierarchy Recloning

Adding Modgens to a Virtual Hierarchy

Modgen Editor

Constraint Manager

Connectivity-Driven Layout Editing Commands Supported by DPA

Adding Modgens to 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.

Related Topics

Modgen Editor

Constraint Manager

Connectivity-Driven Layout Editing Commands Supported by DPA

Converting Created Virtual Hierarchies to Modgens

Virtual hierarchies that are manually created can be converted to Modgens. To convert a virtual hierarchy to a Modgen:

1. In the Navigator assistant or layout canvas, select the virtual hierarchy to be converted.
2. Right-click and select *Convert to Modgen* from the virtual hierarchy shortcut menu.

   The selected virtual hierarchy is converted to a Modgen.

Related Topics

Virtual Hierarchy Editing Commands in Shortcut Menu

Modgen Editor

Pin Accessibility Checker

At advanced nodes, routing standard cell designs can be challenging because the cells are very closely packed and the density of the cells increases at lower nodes. Also, these designs follow complicated design rules. Routing such designs might involve several iterations.

The Pin Accessibility Checker is a standalone tool that lets you verify the routability of standard cells before you route the design. The Pin Accessibility Checker lets you ensure that the standard cells are correct-by-construction in terms of pin accessibility by the router. The tool performs a dynamic test by running the Cadence® Innovus™ router to check whether the router can access all pins without any DRC violations. You can also check for routing feasibility for various topologies in higher metal layers and in double-cut vias on critical nets.

Running the Pin Accessibility Checker early in the standard cell development phase helps avoid delays.

Related Topics

Running the Pin Accessibility Checker

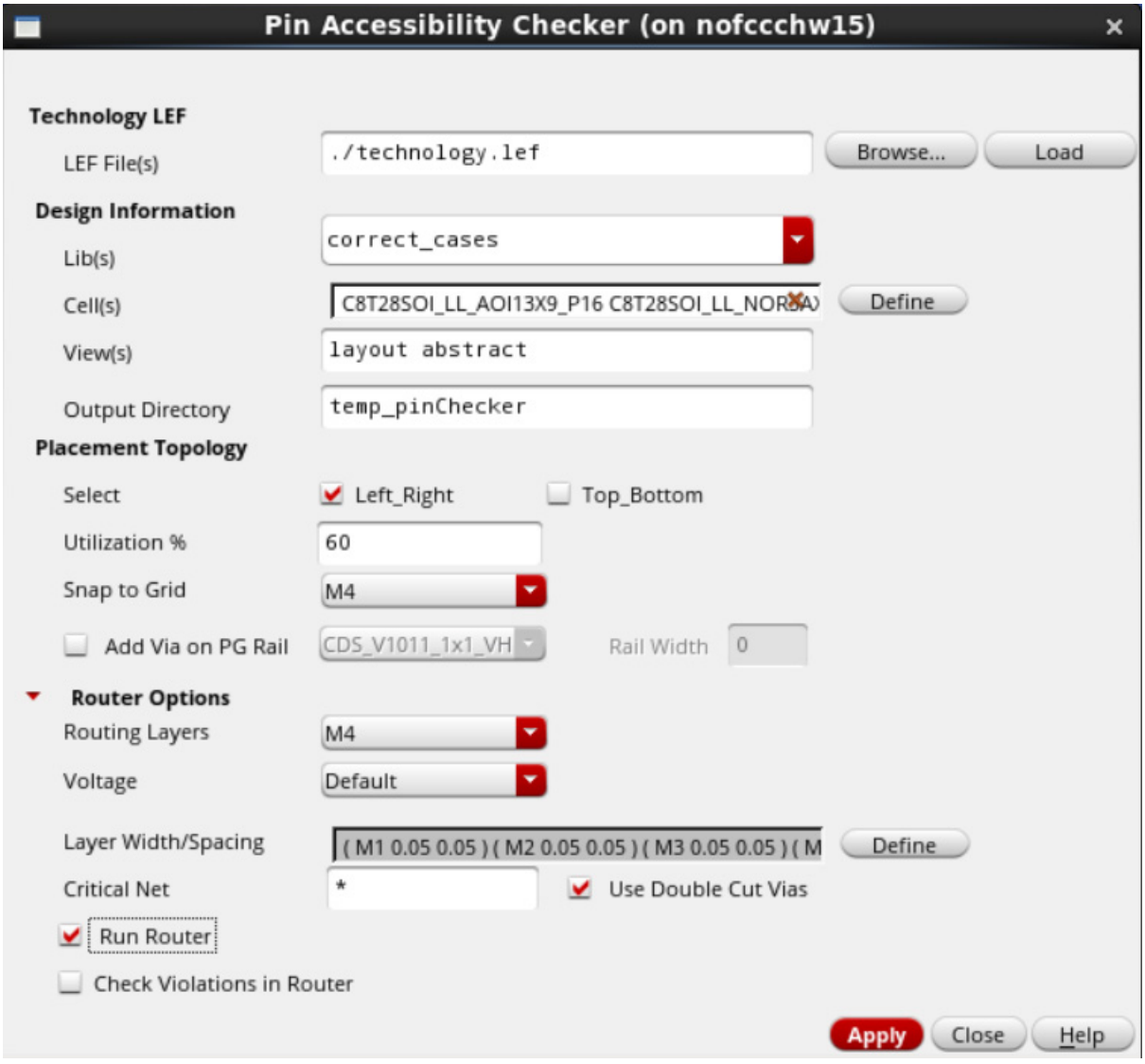
Pin Accessibility Checker Form

Running the Pin Accessibility Checker

To run the Pin Accessibility Checker:

1. In CIW, select *Tools – Check Pin Accessibility*.

   The Pin Accessibility Checker form appears



2. In the *Technology LEF* section:

a. Click *Browse* and select a technology LEF file in the *LEF File(s)* field. Selection of multiple LEF files is supported in the LEF File(s) field and you can select the files from the Select technology LEF file(s) form.

b. Click *Load* to load the LEF file.

3. In the *Design Information* section:

a. Select one or more standard cell libraries from the *Lib(s)* list. Standard cell names in the *Cell(s)* field are automatically populated based on the selected libraries.

b. Specify one or more cells in the *Cell(s)* field. Click *Define* to open the Browse Cell(s) window, from which you can select the required standard cells.

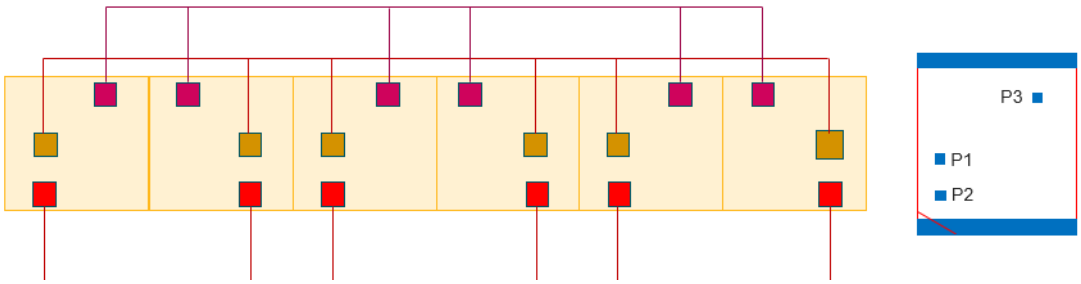
c. Specify one or more views in the *View(s)* field.

d. In *Output Directory*, specify the name of the directory in which all temporary files and views are to be stored.

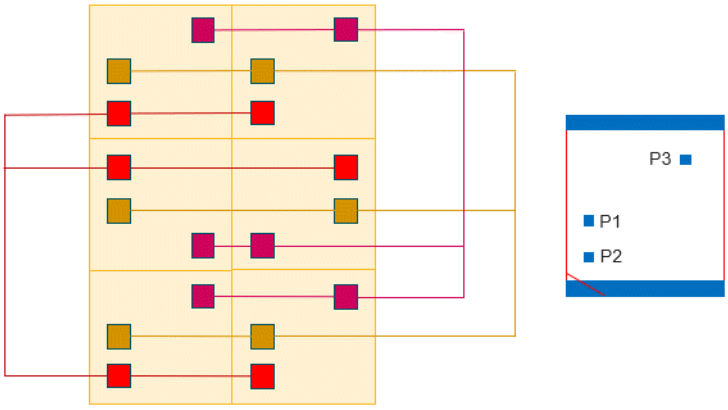
4. In the *Placement Topology* section:

a. Select a placement topology for the standard cells from the *Select* list in the *Placement Topology* section. The available options are:

- *Left_Right*: Places the same cell six times with the `R0` and `MY` orientations.



- *Top_Bottom*: Places the same cell three times with the `R0`, `R180`, and `MY` orientations.



b. Specify the cell utilization in the *Utilization%* field. This is the area to be used for routing.

c. From the *Snap to Grid* drop-down list, select the routing metal layer grid to which the cells are to be snapped.

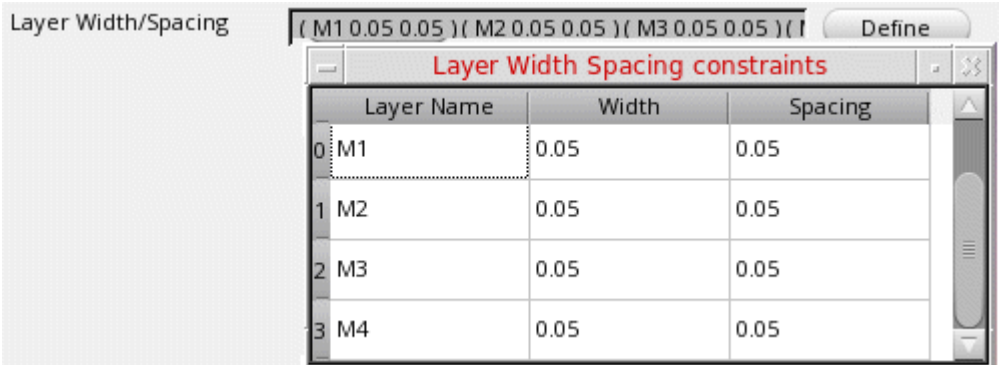
d. From the *Add Via on PG Rail* drop-down list, select a value if vias are to be generated on the power or ground rails.

e. Specify the width of rails in the *Rail Width* field. For example, you can make the width wider than the one drawn in the layout.

5. In the *Router Options* section:

a. Select the highest routing layer from the *Routing Layers* list. All lower metal layers up to the specified layer are used for routing.

b. Specify the width and spacing values for each routing layer in the *Layer Width/Spacing* field. Click *Define* to open the Layer Width Spacing constraint table in a new window. You can edit the values directly in the table. Close the window to populate values in the *Layer Width/Spacing* field.



c. Select a voltage value from the *Voltage* drop-down list to switch on high voltage rule-based routing.

This option is applicable only to input designs with voltage spacing rules defined in the technology file.

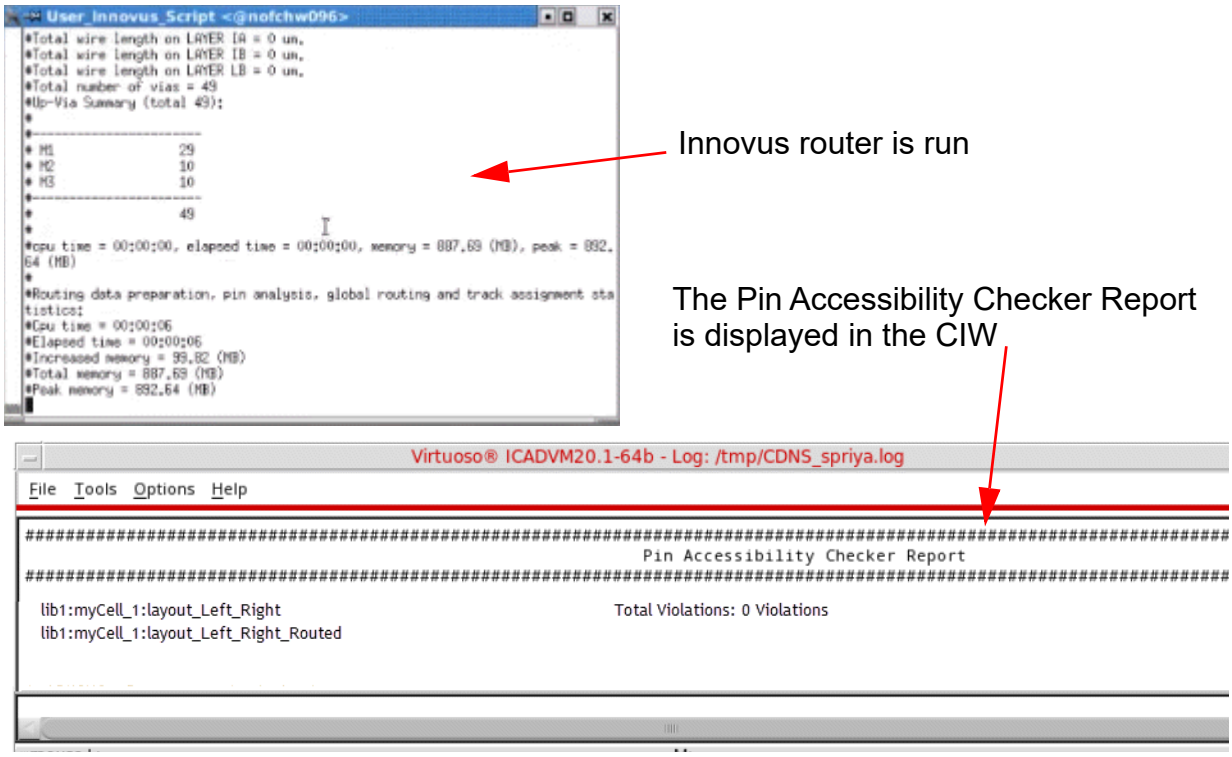
d. Specify the nets that are to be routed first in the *Critical Net* field.

e. Select *Double Cut Vias* to insert double-cut vias in critical nets.

f. Select *Run Router* to run the Innovus NanoRoute router on the standard cell extended pins view and generate a routed view.

g. Select *Check Violations in Router* to open the Innovus router graphical user interface and view any violations in the routed topology view after running the router. This requires the Cadence Innovus license to be checked out.

6. Click *Apply* to run the router and generate the Pin Accessibility Check report.



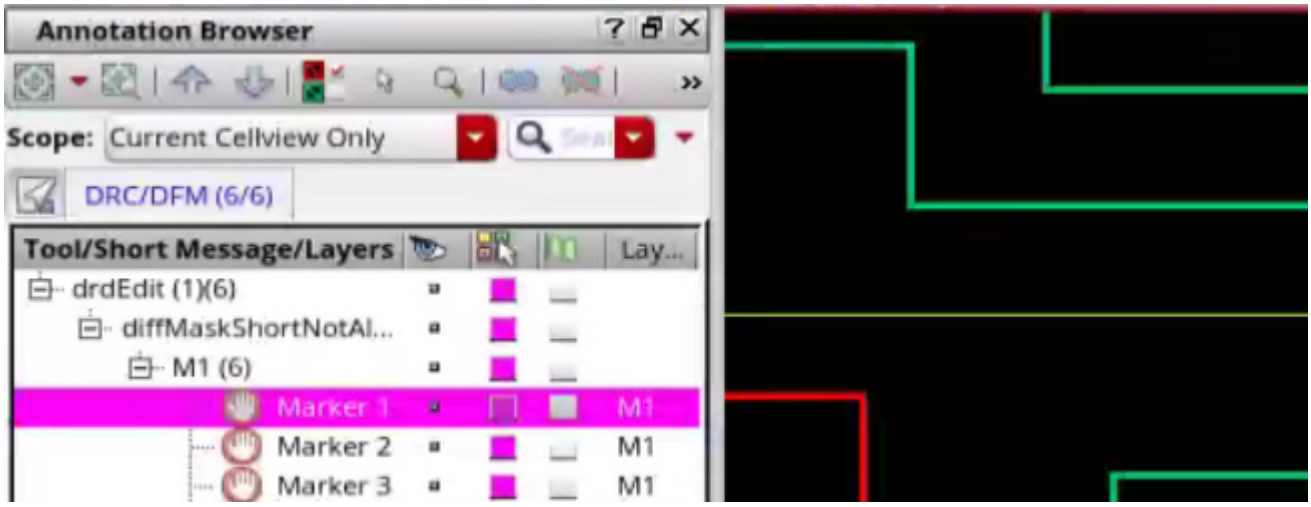
The Pin Accessibility Checker report lists violations, critical net statistics for each cell, and length of metals in non-preferred direction. You can run the tool on different topologies to identify the one that suits your requirements.

When you select *Use Double Cut Vias* and specify a *Critical Net* value, the Pin Accessibility Checker report displays the number of single-cut and double-cut vias per net for each cell.

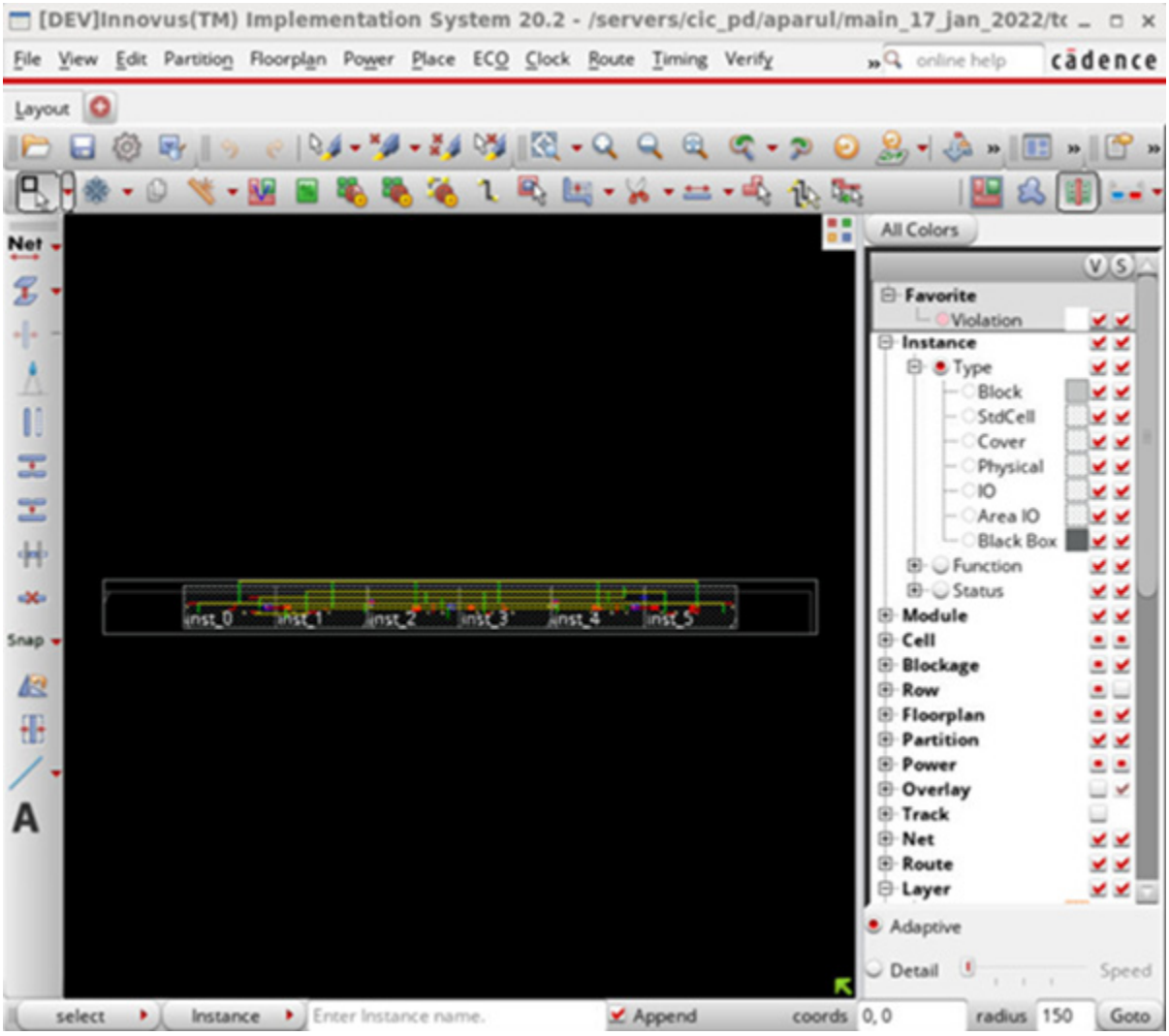
The Pin Accessibility Checker report also includes information about the length of routing metals in the non-preferred direction.

```
##########################################################################################################################################
                                                    Pin Accessibility Checker Report
##########################################################################################################################################

 correct_cases_temp_pinChecker:C8T28SOI_LL_AOI13X9_P16_pac:layout_Left_Right_60_M4       Total Violations: 0 Violations
correct_cases_temp_pinChecker:C8T28SOI_LL_AOI13X9_P16_pac:layout_Left_Right_60_M4_routed |


VFP-40032: Pin Accessibilty Checker will consider net(s) 'gnd!, vdd!, B, C, A, D, Z' as critical nets.

 ******************** Critical Nets Via Statistics for cell C8T28SOI_LL_AOI13X9_P16_pac ********************

    Net B
    Layers            Single-Cuts              Double-Cuts
    #-----------------------------------------------------
    M1-M2               0                        6
    M2-M3               0                        6
    M3-M4               0                        6
    #-----------------------------------------------------

    Net C
    Layers            Single-Cuts              Double-Cuts
    #-----------------------------------------------------
    M1-M2               3                        3
    M2-M3               0                        3
    M3-M4               0                        3
    #-----------------------------------------------------

    Net A
    Layers            Single-Cuts              Double-Cuts
    #-----------------------------------------------------
    M1-M2               4                        2
    M2-M3               0                        6
    M3-M4               0                        6
    #-----------------------------------------------------

********************** Length of Metals in non-preferred direction for cell C8T28SOI_LL_AOI13X9_P16_pac
 Layer: M1 Preferred direction: vertical, Non Preferred direction: horizontal  Non preferred wire lengt
 Layer: M2 Preferred direction: horizontal, Non Preferred direction: vertical  Non preferred wire lengt
 Layer: M3 Preferred direction: vertical, Non Preferred direction: horizontal  Non preferred wire lengt
```

In a routed view, violation markers are created over single-cut vias in the Annotation Browser as shown in the following figure.



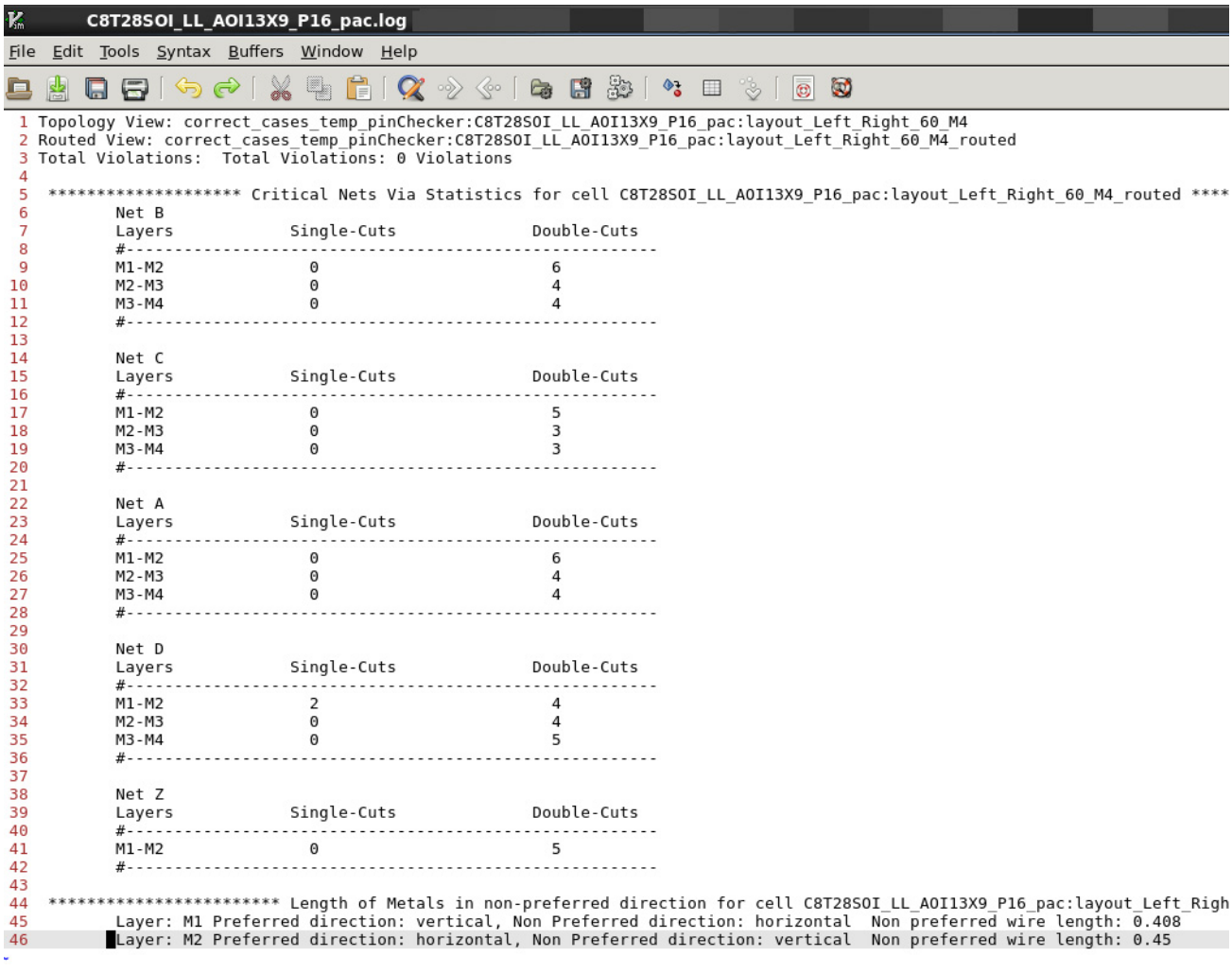
When you select *Check Violations in Router*, a routed view appears in the Innovus user interface, which lets you analyze the violations reported during routing.



Note: Set the `vfpPACRunWithInnovusLic` environment variable to use the VDI flow for running the router without displaying the Innovus router user interface.

The Pin Accessibility Checker report is stored at a default location. This report is saved as a log file for each cell and maintains the details of DRC issues and via information. You can open the log file using the following path in the CIW.

```
<temporary_directory_name>/<newly_created_library_directory>/<cellName>_pac/
<routed_view_name>/<cellName>_pac.log
```



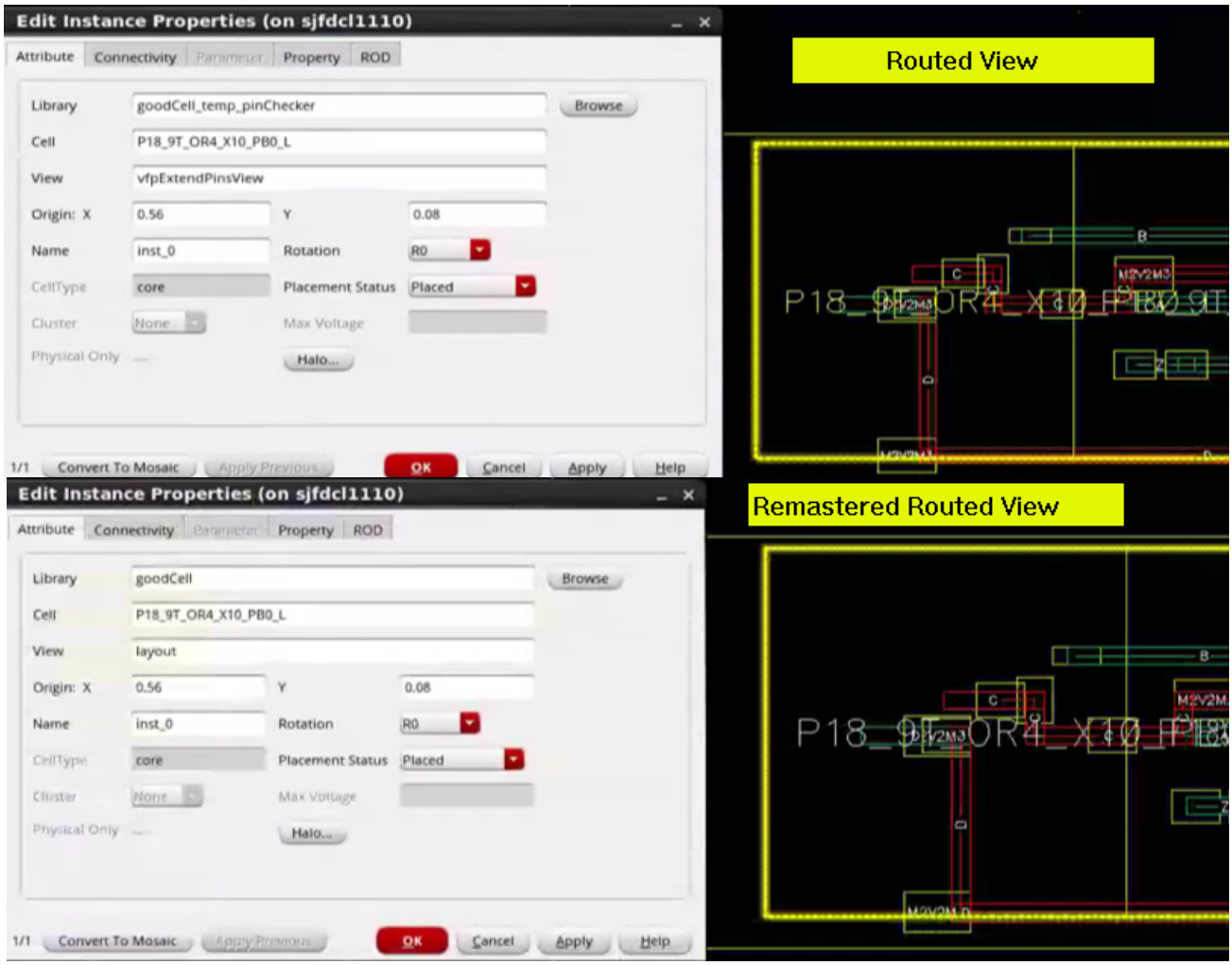

```
 1 Topology View: correct_cases_temp_pinChecker:C8T28SOI_LL_AOI13X9_P16_pac:layout_Left_Right_60_M4
 2 Routed View: correct_cases_temp_pinChecker:C8T28SOI_LL_AOI13X9_P16_pac:layout_Left_Right_60_M4_routed
 3 Total Violations:  Total Violations: 0 Violations
 4
 5  ******************** Critical Nets Via Statistics for cell C8T28SOI_LL_AOI13X9_P16_pac:layout_Left_Right_60_M4_routed ****
 6        Net B
 7        Layers            Single-Cuts             Double-Cuts
 8        #-------------------------------------------------------
 9        M1-M2               0                       6
10        M2-M3               0                       4
11        M3-M4               0                       4
12        #-------------------------------------------------------
13
14        Net C
15        Layers            Single-Cuts             Double-Cuts
16        #-------------------------------------------------------
17        M1-M2               0                       5
18        M2-M3               0                       3
19        M3-M4               0                       3
20        #-------------------------------------------------------
21
22        Net A
23        Layers            Single-Cuts             Double-Cuts
24        #-------------------------------------------------------
25        M1-M2               0                       6
26        M2-M3               0                       4
27        M3-M4               0                       4
28        #-------------------------------------------------------
29
30        Net D
31        Layers            Single-Cuts             Double-Cuts
32        #-------------------------------------------------------
33        M1-M2               2                       4
34        M2-M3               0                       4
35        M3-M4               0                       5
36        #-------------------------------------------------------
37
38        Net Z
39        Layers            Single-Cuts             Double-Cuts
40        #-------------------------------------------------------
41        M1-M2               0                       5
42        #-------------------------------------------------------
43
44  ************************ Length of Metals in non-preferred direction for cell C8T28SOI_LL_AOI13X9_P16_pac:layout_Left_Righ
45        Layer: M1 Preferred direction: vertical, Non Preferred direction: horizontal  Non preferred wire length: 0.408
46        Layer: M2 Preferred direction: horizontal, Non Preferred direction: vertical  Non preferred wire length: 0.45
```

Related Topics

Pin Accessibility Checker

Fix Color Conflicts in Remastered Routed Views

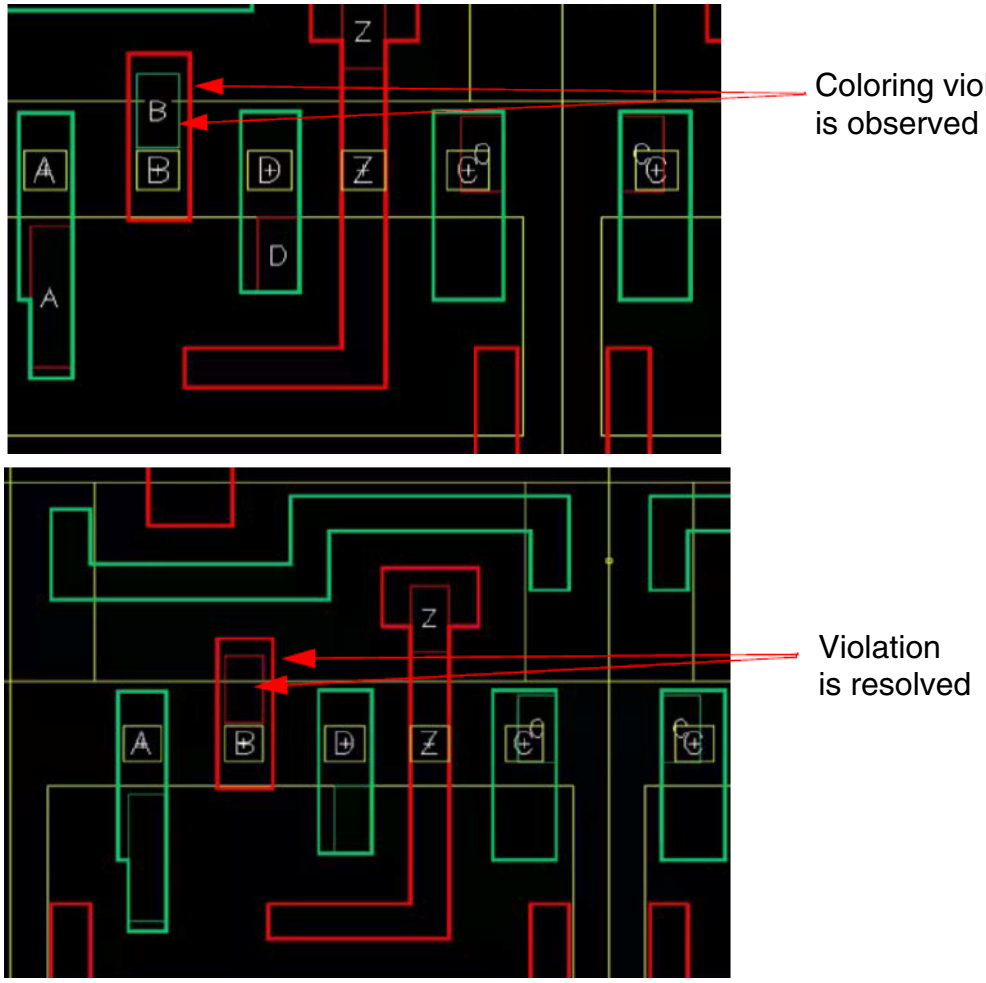
After routing a design, the router creates a remastered routed view of the cellview. In this view, the instances of the cellview are remastered to get back the original geometries, as shown in figure below:



Recoloring is a method that can be used to fix color conflicts that arise due to uncolored extended pin views being used by the router. During routing, the router might insert uncolored pin extensions to fix certain violations. These uncolored pin extensions might cause color conflicts. With the `coloringEngineEnabled` environment variable to `t`, you can run the Pin Accessibility Checker to fix such coloring violations.

The Pin Accessibility Checker first aligns the routing metals, vias, and patches in the cell layout. It then recolors the remastered cellview to fix any color violations and saves the design.

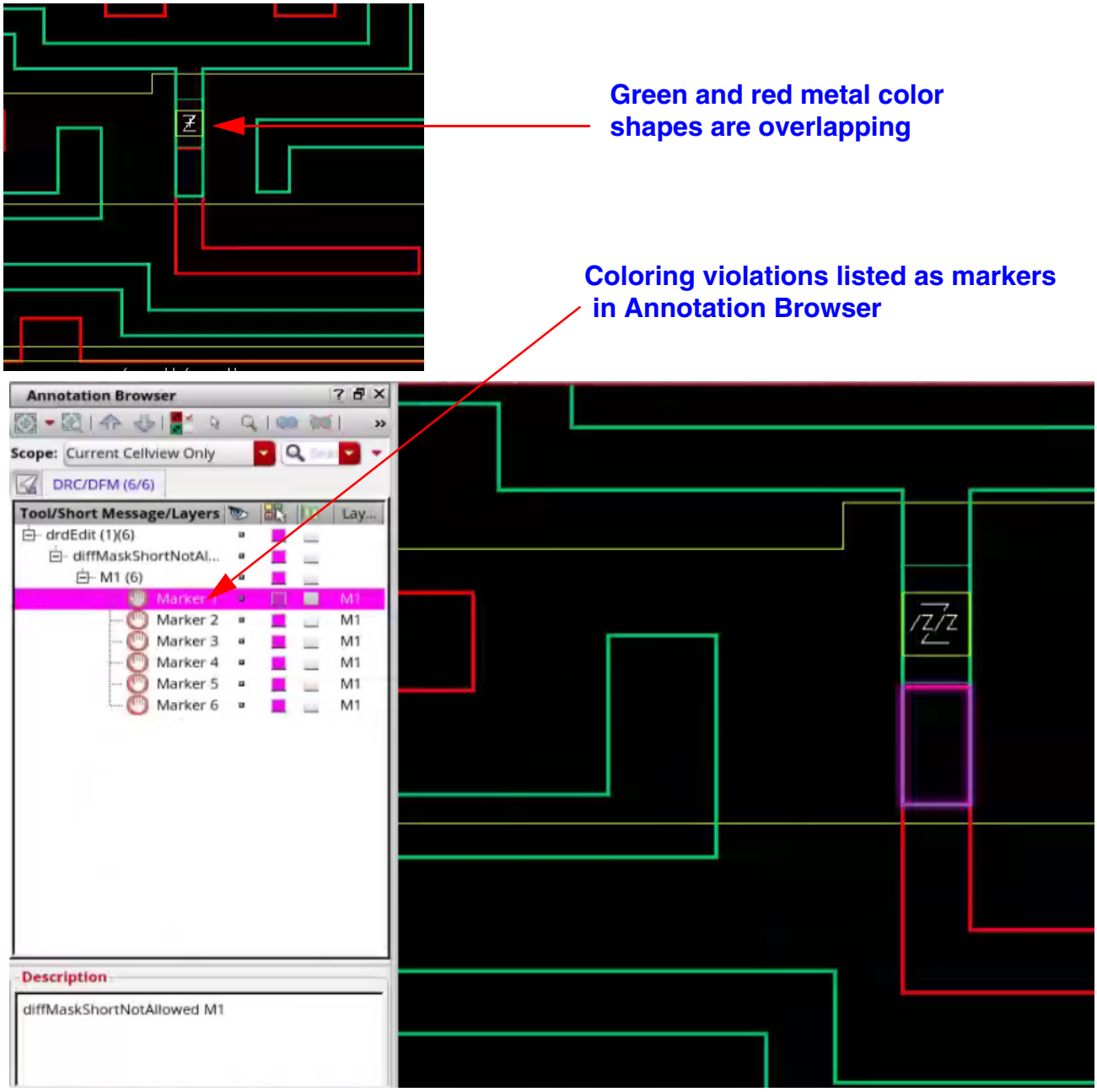
Consider a layout design where color violations are observed when the color of a patch does not match the color of vias generated by the router. In the image below, the color of shape B is red and color of vias inside it is green and therefore a violation is observed. With `coloringEngineEnabled` set to `t`, the routed remastered view is recolored, and color violations are resolved.



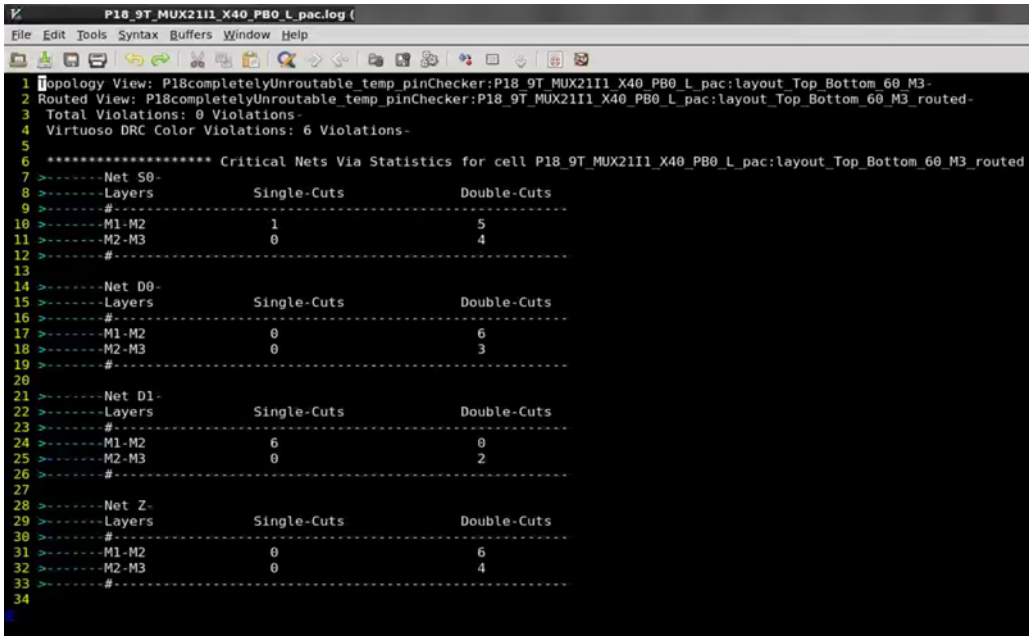
The coloring information is removed while creating extended pin views and the router does not contain any color information in the placement view. As a result, the router cannot catch any coloring violations.To check coloring violations, you can set the vfpPACRunVerifyDesignWithColorOpts environment variable to `t`. This runs the Virtuoso Multi-Patterning Technology color engine on the remastered routed view along with the *Verify Design* command and the coloring violations are reported in the Annotation Browser assistant.

In the image below, green and red metal color shapes are overlapping, which represents coloring violation in a design. The Pin Accessibility Checker runs coloring checks on the metal shapes and reports the coloring violations in the remastered routed view. The coloring

violations generated by the tool are displayed as markers in the layout and the Annotation Browser.



The Pin Accessibility Checker displays a log file listing all the Virtuoso DRC color violations that were reported as markers in Annotation Browser.



Related Topics

coloringEngineEnabled

4

Congestion Analysis

Congestion analysis determines the routing capacity of a design. You can run congestion analysis both on globally routed and detail-routed designs. The level of congestion in a design can be represented graphically using a heat map, statistically using a histogram, or by using the routing metrics table that the global routing engine internally uses to map capacity and plan routing.

To simplify the initial problem, the global router creates a course grid that is a multiple of the average of the pitch or routing track grid for the various layers being routed. This course grid is known as the gcell grid. The congestion for every gcell grid is then calculated as the ratio between the number of routed wires and blockages versus the number of available routing tracks.

The Congestion Analysis utility lets you run comprehensive checks and ensure that accurate results are extracted and displayed. It also lets you view and analyze the routing density of a design from initial floorplanning to detailed placement and routing. This data is used to streamline the floorplanning task and let you tune global planning to improve routing convergence, and perform congestion-based pin optimization.

Related Topics

Environment Setup for Congestion Analysis

Displaying the Congestion Analysis Assistant

Loading the Congestion Analysis Workspace

Uses of the Congestion Analysis Assistant

Customizing a Histogram

Global Bias Constraints

Managing Scenic Ratio

Environment Setup for Congestion Analysis

Note the following settings to perform congestion analysis.

- The Congestion Analysis assistant is available in Layout EXL and higher tiers.
- Ensure that *Default Wire Constraint Group* is set to `LEFDefaultRouteSpec`. This is required because the global router requires a default constraint group where both `validRoutingLayers` and `validRoutingVias` constraints are set.
- Ensure that *Design Style* in the Virtuoso Space-based Router form is either selected as *Chip Assembly* or *ASIC*. Congestion Analysis has limited functionality when *Design Style* is set to *Device*.
- Ensure that the *Route Flow* in the Wire Assistant is selected as *Minimum Spanning Tree*. Congestion analysis only supports the *Minimum Span Tree* routing flow.

Related Topics

Congestion Analysis

Displaying the Congestion Analysis Assistant

Loading the Congestion Analysis Workspace

Uses of the Congestion Analysis Assistant

Customizing a Histogram

Global Bias Constraints

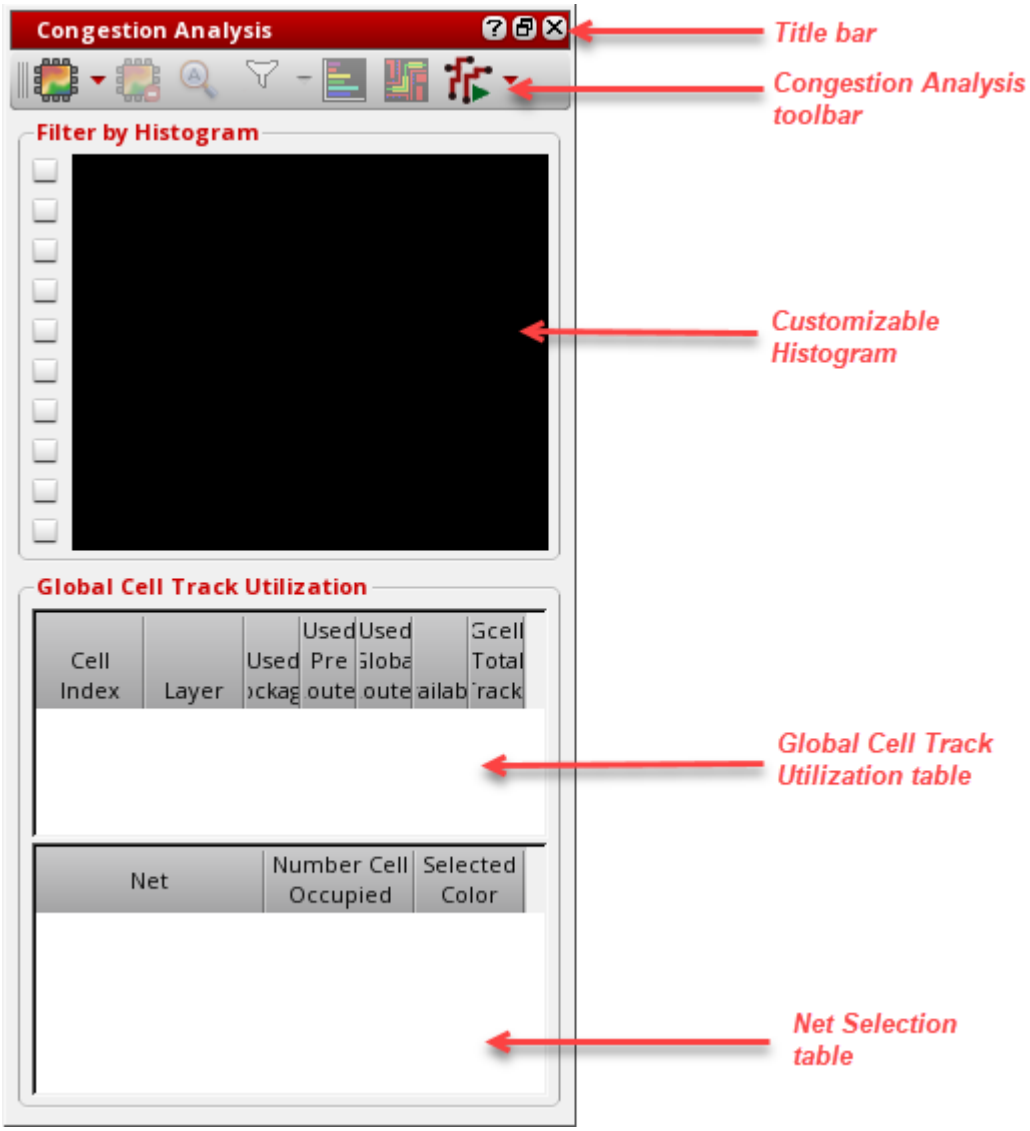
Managing Scenic Ratio

Displaying the Congestion Analysis Assistant

Use one of the following methods to display the Congestion Analysis assistant.

- Choose *Window – Assistants – Congestion Analysis*.
- Right-click anywhere in the layout window menu bar and choose *Assistants – Congestion Analysis*.
- Click the *Congestion Analysis* icon  on the *Design Planning* toolbar. For more information, see Design Planner Toolbar.

The main components of the *Congestion Analysis* assistant user interface are shown in the following figure.



To hide the Congestion Analysis assistant, do one of the following:

- Click the *Hide* button in the Congestion Analysis title bar.
- Right-click anywhere in the layout window menu bar and choose *Assistants – Congestion Analysis*.

Related Topics

Congestion Analysis

Environment Setup for Congestion Analysis

Loading the Congestion Analysis Workspace

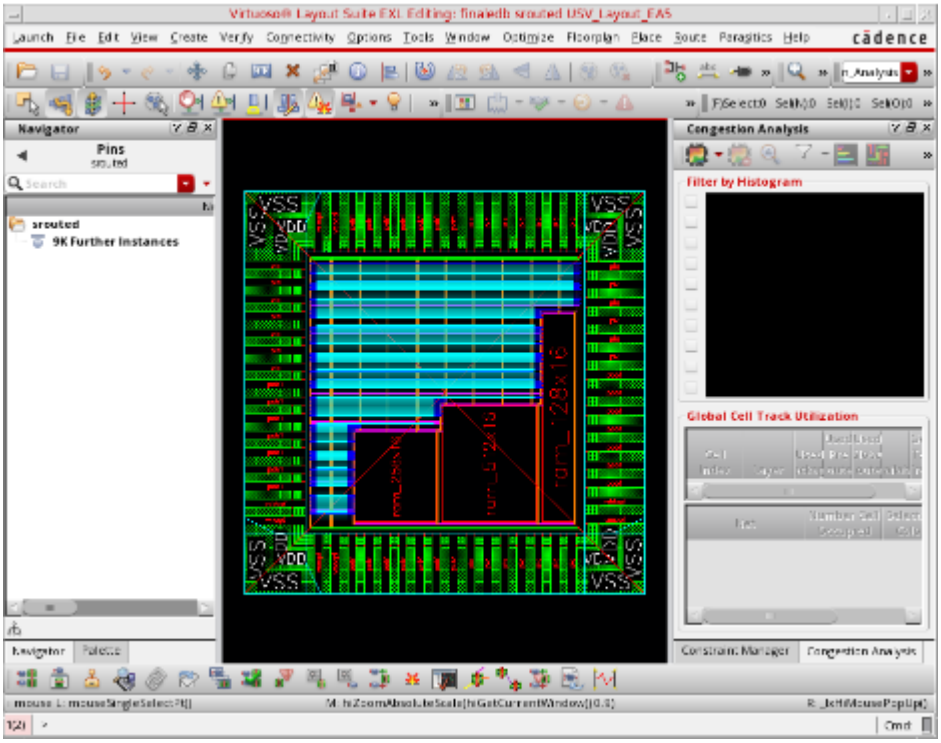
Uses of the Congestion Analysis Assistant

Customizing a Histogram

Global Bias Constraints

Loading the Congestion Analysis Workspace

To load the *Congestion Analysis* workspace, choose *Windows – Workspace – Congestion Analysis*. The *Congestion_Analysis* workspace displays the Navigator assistant, Palette assistant, and the Congestion Analysis assistant.



Related Topics

Congestion Analysis

Environment Setup for Congestion Analysis

Displaying the Congestion Analysis Assistant

Uses of the Congestion Analysis Assistant

Customizing a Histogram

Global Bias Constraints

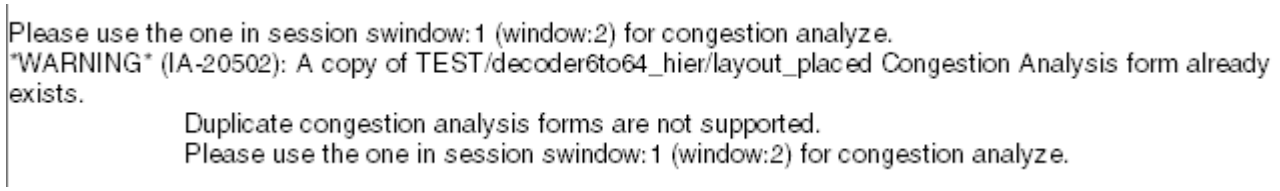
Managing Scenic Ratio

Uses of the Congestion Analysis Assistant

Use the Congestion Analysis assistant to:

- run global routing and congestion analysis.
- illustrate the various methods of viewing congestion in your design.
- analyze congestion.

The Congestion Analysis assistant has only one record per cellview. If you attempt to open another instance of the Congestion Analysis assistant for a cellview that already has a copy of congestion analysis record, a warning message is displayed, as shown below.



The copied ones only display the synchronized heat map, but the Congestion Analysis assistant is not displayed.

Related Topics

Congestion Analysis

Environment Setup for Congestion Analysis

Running Global Routing and Congestion Analysis

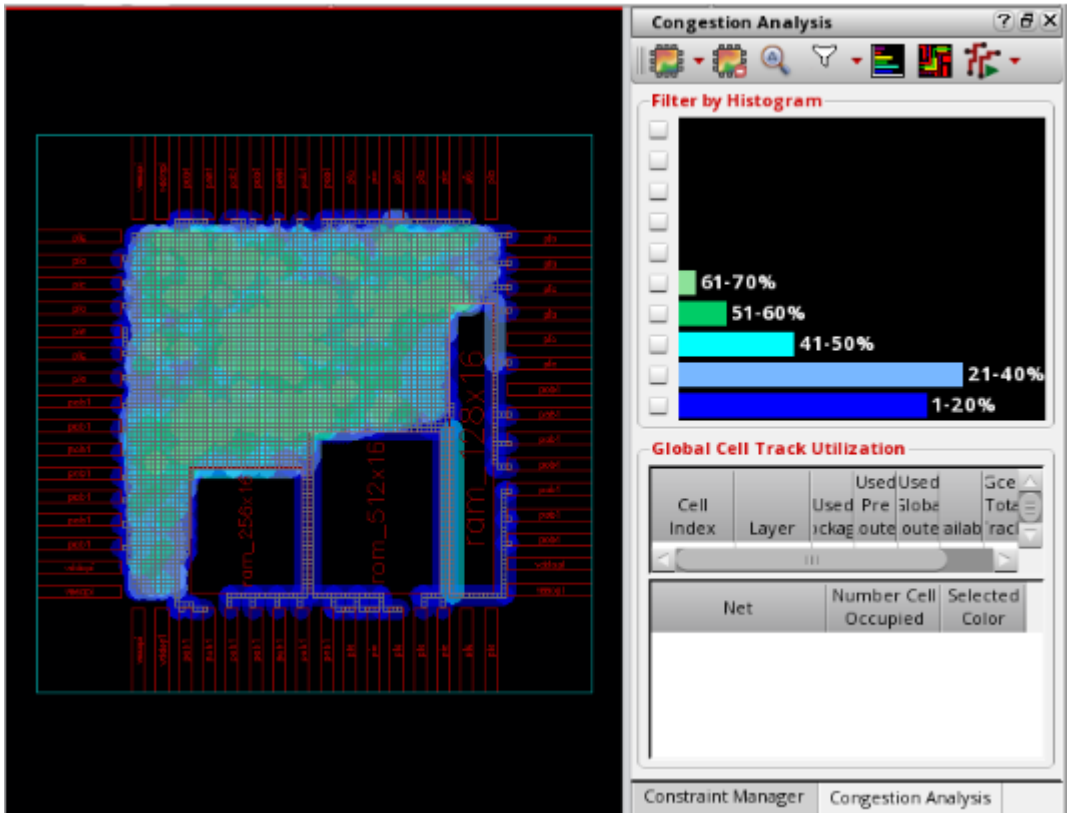
Before visualizing the congestion data, run global routing and congestion analysis on the design data. To do this:

1. Click the *Congestion Analysis* icon  on the Congestion Analysis assistant toolbar.

   Global routing and congestion analysis is run on the design. You can refer to the status and the results of congestion analysis in the CIW or review the data from the log.

   The global router builds the gcell grid and then runs four passes of global routing in an attempt to route all the nets. Once routing is complete, you can see the number of routes passing through every gcell edge in the design. This information is then used to calculate the availability of each gcell and illustrate the availability of a gcell as a color on the heat map.

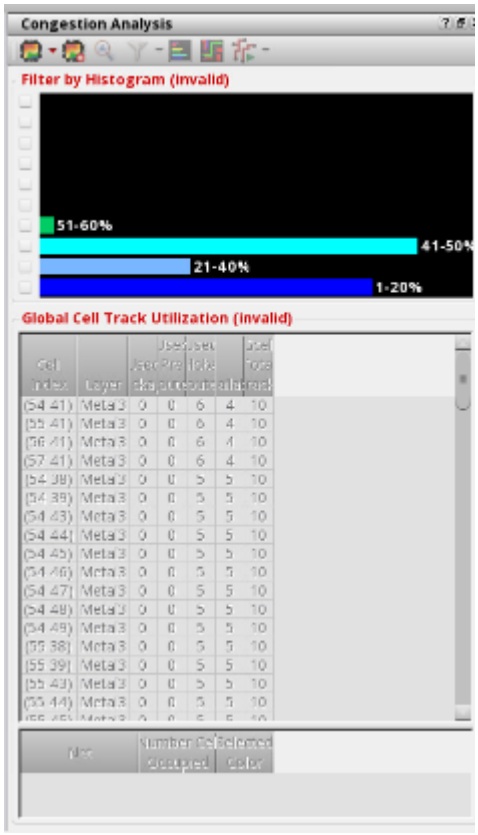
The following figure shows the congestion results displayed in the heat map and the histogram in the Congestion Analysis assistant.



When you modify an object in the design, such as a block, cell, soft block, or change the shape or placement of a virtual hierarchy, the data in the Congestion Analysis assistant is in an invalid state. In this state, a warning message is displayed informing you that the congestion analysis data is no longer valid because some of the routing objects have been modified.

```
*WARNING* (IA-20500): The congestion analysis data is no longer valid because
some of the routing objects have been modified. Click the Run Congestion
Analysis button on the toolbar to regenerate the congestion data or click the
Clear Congestion Analysis Data button on the toolbar to clear the congestion
map.
```

In addition, histogram, Global Cell Track Utilization table and some of the congestion analysis assistant toolbar options are inactive.



Note: In the *Congestion Analysis* toolbar only the first two options, *Congestion Analyze* and *Clear Congestion Analysis Data* are enabled.

To get valid data and state, you need to re-run congestion analysis. Once congestion analysis is run again, histogram, Global Cell Track Utilization table and the toolbar options are enabled and restored.

2. To clear the analysis results and delete global routing, click the *Clear Congestion Analysis Data* icon on the toolbar.

Related Topics

Congestion Visualization

Analyzing Congestion

Congestion Visualization

Once global routing and congestion analysis is run, there are various methods to view congestion in your design. Following are the methods to view congestion in your design.

- View the Heat Map
- View the Histogram
- View the Global Cell Track Utilization Table

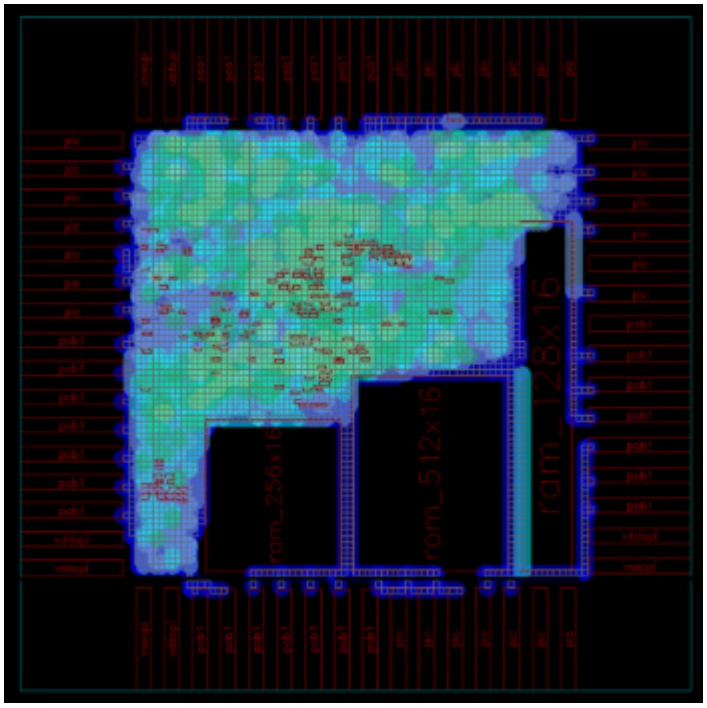
View the Heat Map

The heat map is a standard method to visualize congestion. After global routing and congestion analysis is complete, the heat map is updated to illustrate the congestion result in the main window.

You can use the heat map to visualize and analyze congestion in your design. The heat map shows the level of congestion in different colors. The indication of the different color codes is as given below.

- Cold colors, such as Blue and Green indicate less congestion.
- Warmer colors, such as Yellow, Orange, and Red indicate high congestion.
- Hot colors, such as Purple and white indicate over congestion.

The following figure shows the congestion result in the heat map.



Since the colors displayed in the heat map are mostly blue and green in color, it indicates that the design is less congested.

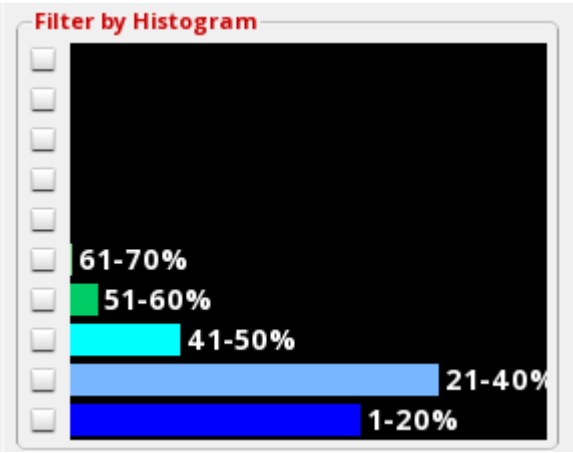
View the Histogram

Another method to view and analyze congestion in your design is using the histogram section in the Congestion Analysis assistant. The histogram provides instant feedback and displays the amount of congestion in a design. It is used for two purposes:

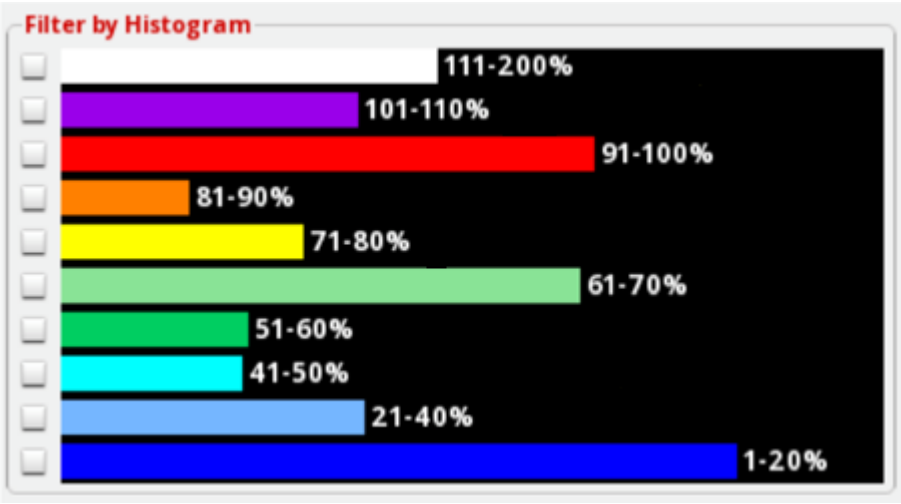
- Display the color palette for the heat map.
- Display the congestion curve in the design.

  The length of the color bar in each percentage bucket represents the number of global cells (as a ratio of the entire design) that have a particular level of congestion.

The following figure shows an example of a design that is less congested. Most of the gcells are present in the `1-20%` and `21-40%` buckets. Therefore, there are lots of available tracks throughout the design.



Here is another example that shows the over-congested design. Most of the gcells are contained in the upper buckets, which means that less routing resources are available.



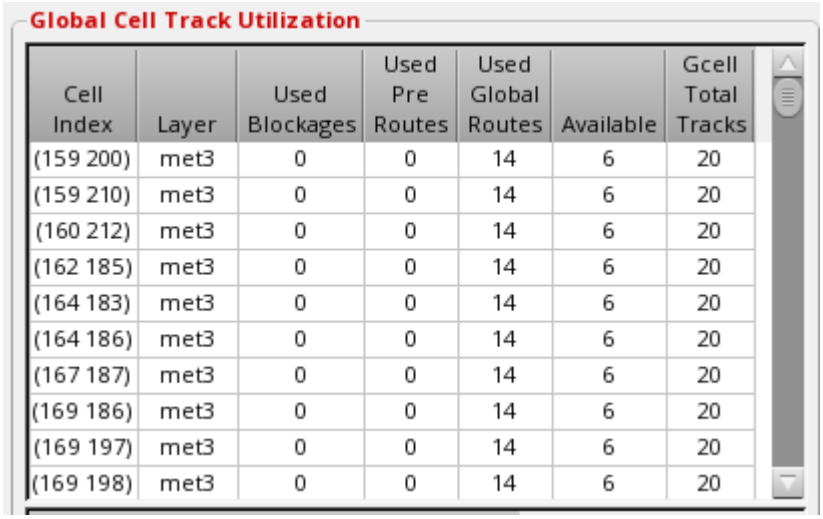
View the Global Cell Track Utilization Table

When global routing and congestion analysis is run, the global cell track utilization table is populated with the capacity and availability information of every gcell in the design. It is an effective and informative arrangement to quickly view the gcell information.

- Horizontal congestion for the track utilization table is calculated using the track capacity and track availability data along the LEFT edge of a gcell.

- Vertical congestion for the track utilization table is calculated using the track capacity and track availability data along the BOTTOM edge of a gcell.

The following figure displays how the information is displayed in the *Global Cell Track Utilization* table.



Cell Index	Layer	Used Blockages	Used Pre Routes	Used Global Routes	Available	Gcell Total Tracks
(159 200)	met3	0	0	14	6	20
(159 210)	met3	0	0	14	6	20
(160 212)	met3	0	0	14	6	20
(162 185)	met3	0	0	14	6	20
(164 183)	met3	0	0	14	6	20
(164 186)	met3	0	0	14	6	20
(167 187)	met3	0	0	14	6	20
(169 186)	met3	0	0	14	6	20
(169 197)	met3	0	0	14	6	20
(169 198)	met3	0	0	14	6	20

Related Topics

Running Global Routing and Congestion Analysis

Analyzing Congestion

Analyzing Congestion

You can use the heat map, the histogram, and global cell track utilization table to analyze congestion, quickly debug hot spots, and find nets that pass through congested regions of the design. The three methods to analyze congestion are:

- Filtering by Analysis Modes
- Filtering by Layers
- Filtering by Histogram

Related Topics

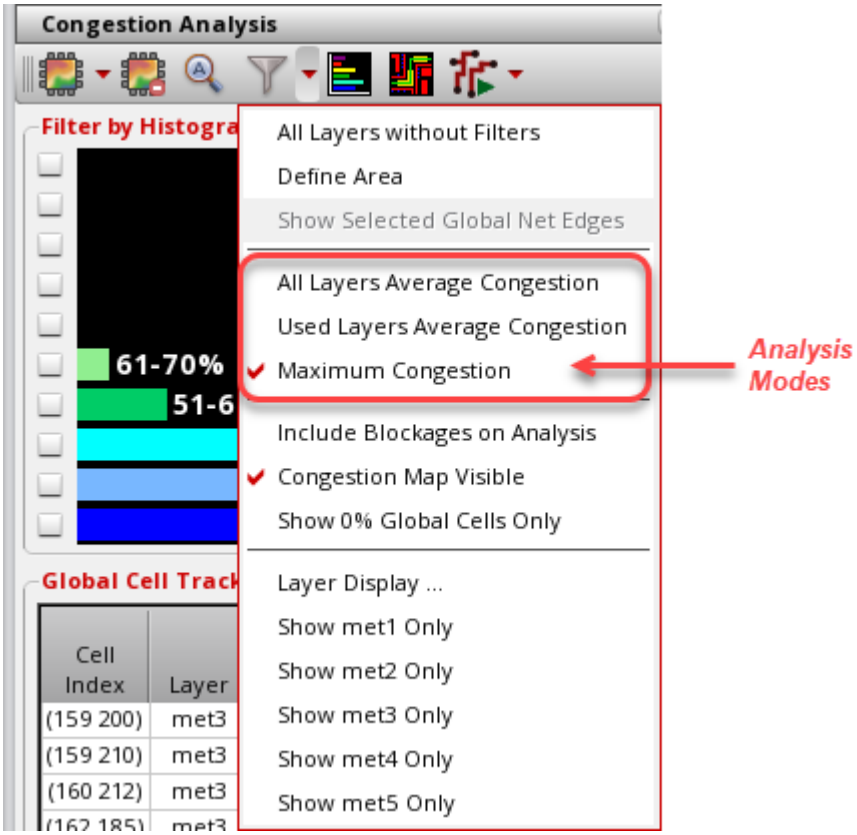
Running Global Routing and Congestion Analysis

Congestion Visualization

Filtering by Analysis Modes

Usually, the average congestion across all layers in a design is displayed on the heat map. However, this can be misleading. To correctly analyze the congestion across all layers and display it on the heat map, you can change the analysis mode. To do this:

1. Click the drop-down arrow next to the *Filter Global Cells by* icon on the *Congestion Analysis* toolbar.
2. Choose the analysis mode from the drop-down list box, as shown in the following figure.



The three analysis modes to view congestion on the heat map are:

- *All Layers Average Congestion*

  For each gcell, calculates the average congestion for all horizontal and vertical layers and colors the gcell based on the average computed percentage.

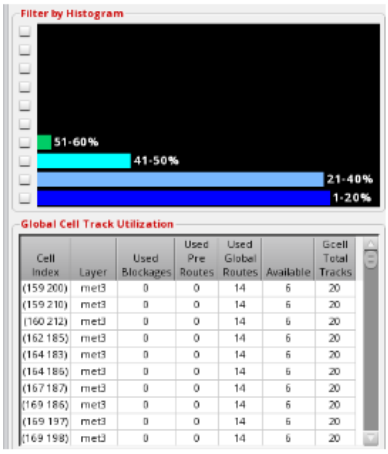
- *Used Layers Average Congestion*

For each gcell, calculates the average congestion for all horizontal and vertical layers that have global routing and colors the gcell based on the average computed percentage.

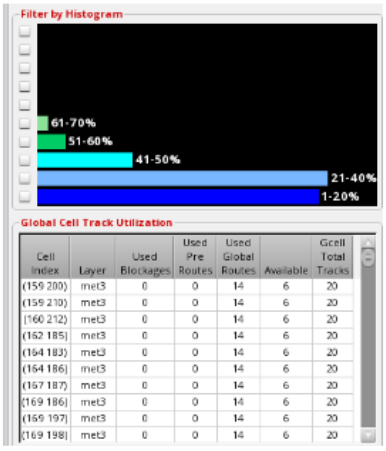
- *Maximum Congestion*

  For each gcell, calculates the maximum congested percentage for all horizontal and vertical layers and colors the gcell based on the maximum congested percentage. This is the default analysis mode.

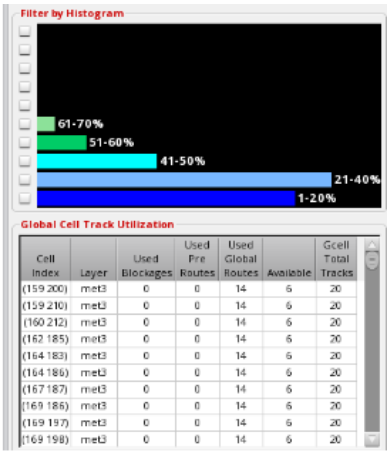
You can view variations in the congestion data in the heat map, the histogram, and the global cell track utilization table. The view in the heat map, histogram, and the global cell track utilization table changes to display the variation in congestion data when you switch between the three analysis modes.



Congestion Data when All Layers Average Congestion is selected



Congestion Data when Used Layers Average Congestion is selected



Congestion Data when Maximum Congestion is selected

Note: You would mostly want to view maximum congestion because it quickly and graphically provides you the worst possible heat map across all layers.

Related Topics

Running Global Routing and Congestion Analysis

Congestion Visualization

Analyzing Congestion

Filtering by Layers

Filtering by Histogram

Filtering by Layers

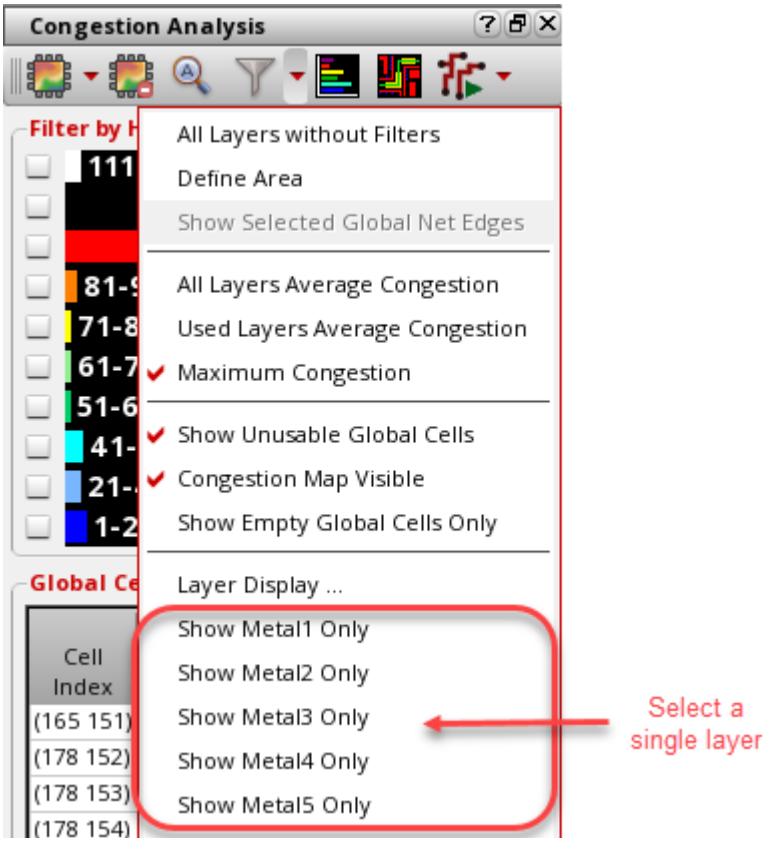
You can quickly view congestion results either in the heat map or the histogram by filtering the results by layer. The two ways to do this are as follows.

- Selecting a Single Layer
- Selecting Multiple Layers

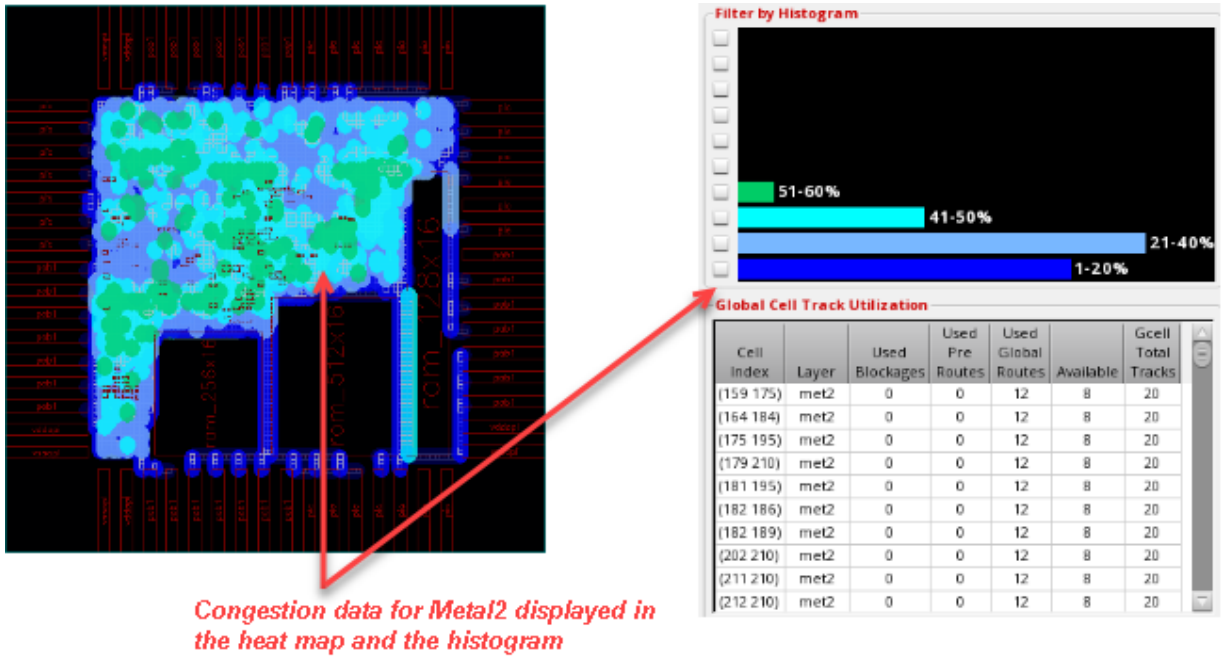
Selecting a Single Layer

To do this:

1. Click the drop-down arrow next to the *Filter Global Cells by* icon on the *Congestion Analysis* toolbar.
2. Choose a layer for which you want to view congestion. For example, select *Show Metal1 Only* from the drop-down list.

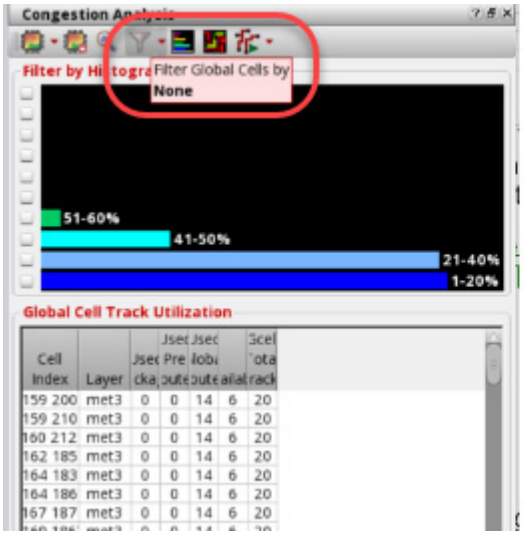


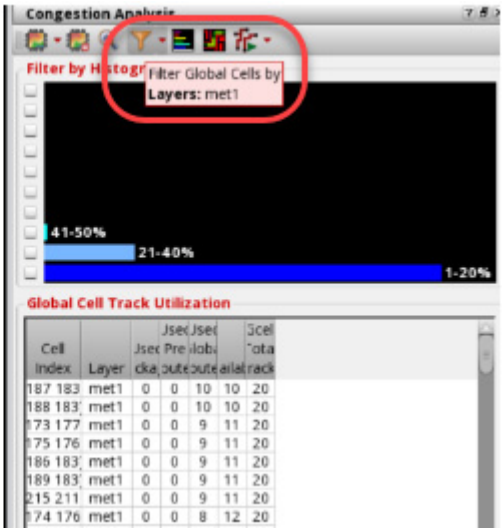
The following figure displays the congestion data of the selected layer in the heat map, histogram, and the global cell track utilization table.



The view in heat map, histogram, and the global cell track utilization table changes to display the variation in the congestion data depending on the metal layer selected.

When the layer filter is set to a single layer, the color of the filter icon is changed from Grey to light orange. The tooltip is also updated according to the filter applied.

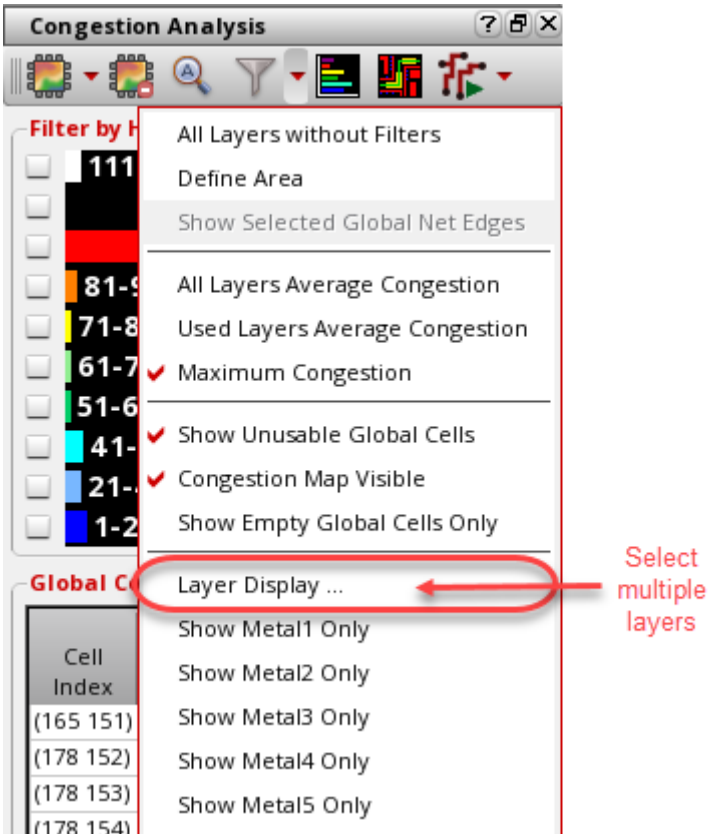




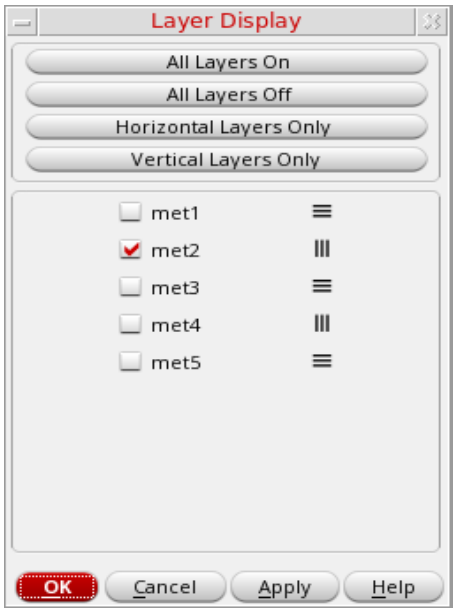
Selecting Multiple Layers

To do this:

1. Click the drop-down arrow next to the *Filter Global Cells by* icon on the *Congestion Analysis* toolbar.
2. Click the *Layer Display* option.

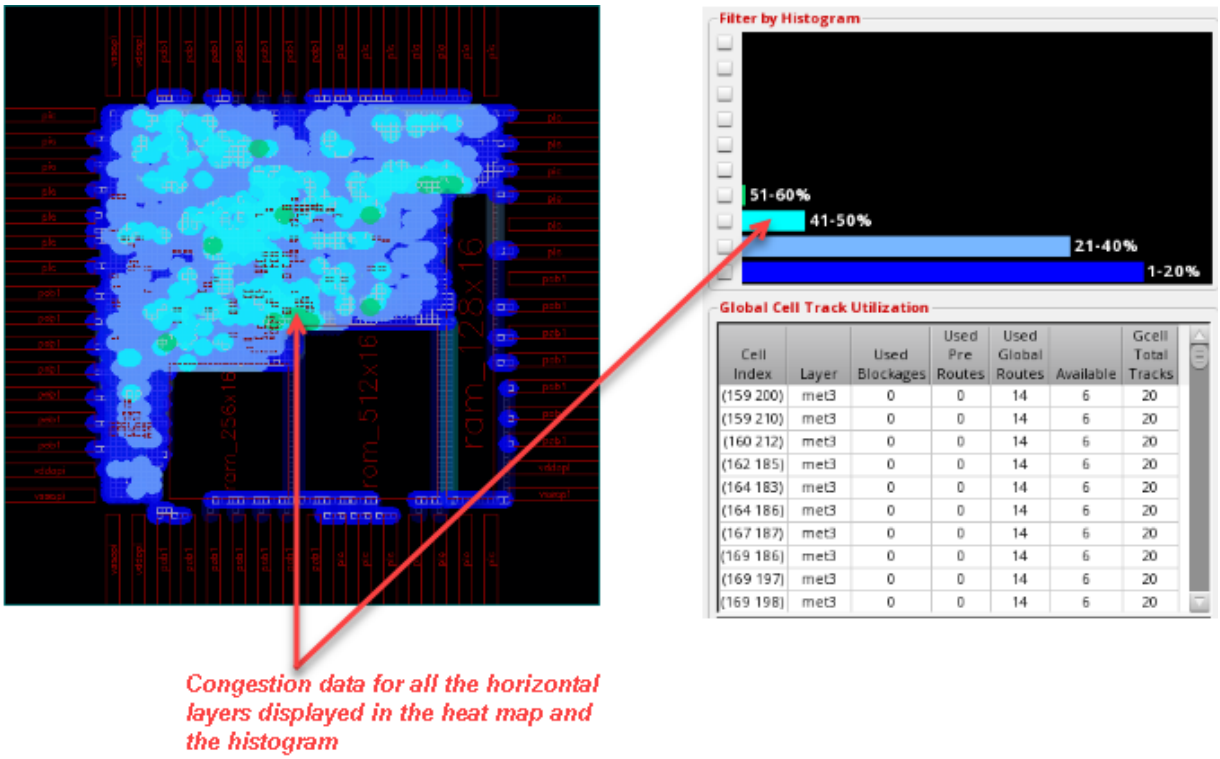


The Layer Display form is displayed. Using this form, you can choose multiple layers for which you want to display congestion.



3. Choose the layers for which you want to display the congestion data.
4. Click *OK*.

The following figure displays the congestion data for only the horizontal layers.



The view in the heat map, histogram, and the global cell track utilization table changes to display the variation in the congestion data depending on the metal layers selected.

Related Topics

Layer Display Form

Running Global Routing and Congestion Analysis

Congestion Visualization

Analyzing Congestion

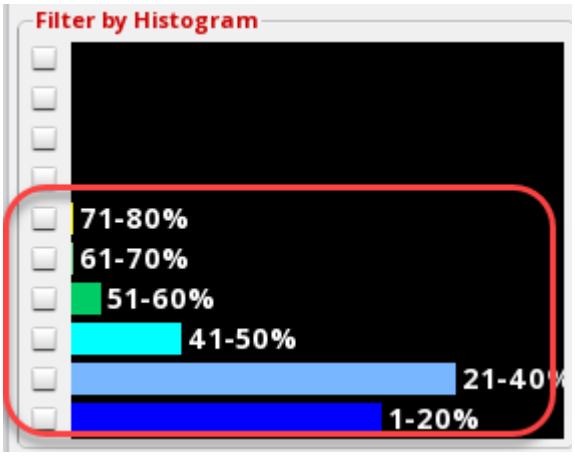
Filtering by Analysis Modes

Filtering by Histogram

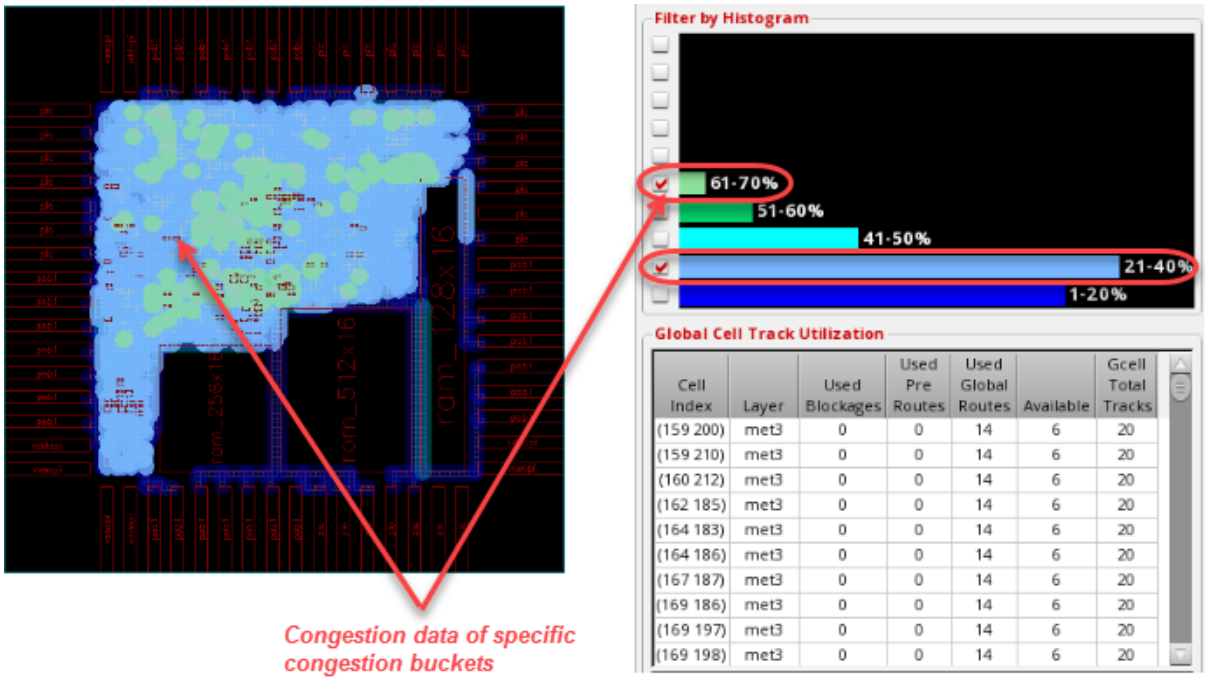
Filtering by Histogram

The filtering by histogram method helps in viewing specific buckets of congestion on the heat map. To filter the histogram, perform the following steps:

1. Run congestion analysis.
2. Select the check box next to the congestion bucket that you want to display on the heat map. You can switch between the different congestion buckets that are available.



The following figures displays the gcells and their congestion between `61-70%` and `21-40%`.



The gcells from the unselected congestion buckets appear dim in the heat map.

The view in the heat map, histogram, and the global cell track utilization table changes to display the variation in the congestion data depending on the congestion bucket selected.

Related Topics

Running Global Routing and Congestion Analysis

Congestion Visualization

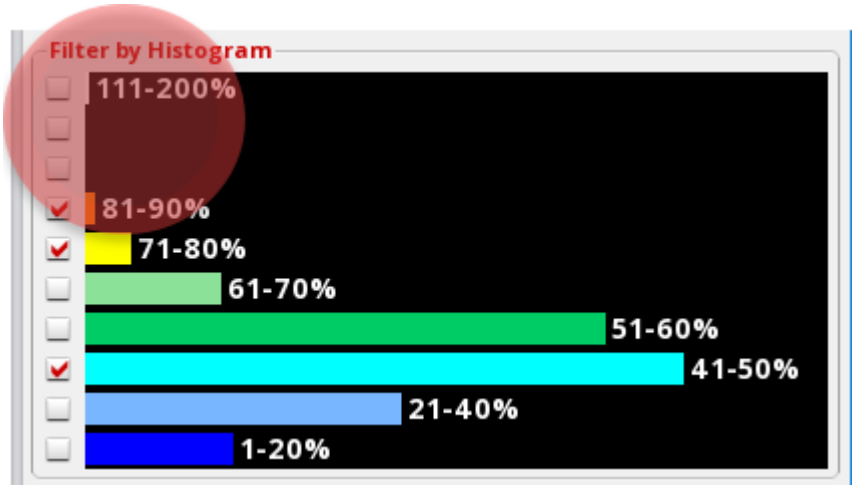
Analyzing Congestion

Filtering by Layers

Filtering by Analysis Modes

Customizing a Histogram

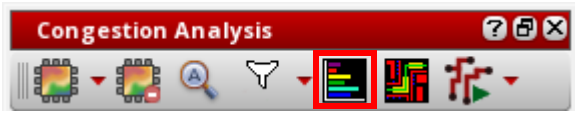
At times it may become necessary to customize the histogram. Let us consider a scenario when you run global routing and congestion analysis and most of the design is congested. However, there are some very small regions of high congestion. The following figure displays the histogram with small regions of high congestion.



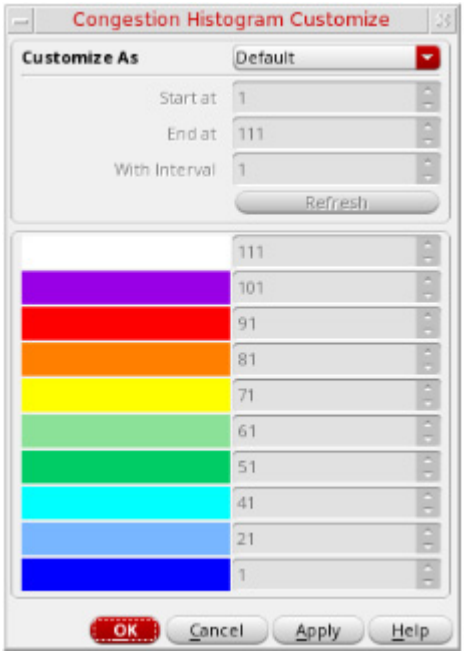
Because there are small bands of high congestion, it is difficult to know the number of gcells that are in the 111-200% bucket compared to the ones in the 91-100% bucket. Also, it is difficult to know the number of gcells that are 95% congested compared to ones that are 98% congested.

In such a scenario, customizing the histogram will be beneficial. You can customize the histogram to only display the congested buckets and expand them to the entire histogram table. To do this:

1. Click the *Congestion Histogram Customize* icon on the Congestion Analysis toolbar.



The Congestion Histogram Customize form is displayed.



2. Select one of the following methods to customize the histogram: *Default*, *Interval*, *Start and End*, *Specified.* To view the results based on histogram customization, see Results Based on Histogram Customization.
3. In the *Start at* field, specify the congestion percentage with which to start. This field is enabled only when the customization method is selected as *Interval* or *Start and End*.
4. In the *End at* field, specify the congestion percentage on which you want to end. This field is enabled only when the customization method is selected as *Start and End*.
5. In the *With Interval* field, specify the interval between the congestion buckets. This field is enabled only when the customization method is selected as *Interval*.
6. Click *Refresh*. The table entries in the Congestion Histogram Customize form are updated.

7. Click *OK*.

The congestion data displayed in the heat map, the histogram, and the global cell track utilization table is updated based on the selected customization method and the related settings.

Related Topics

Congestion Histogram Customize Form

Results Based on Histogram Customization

Results Based on Histogram Customization

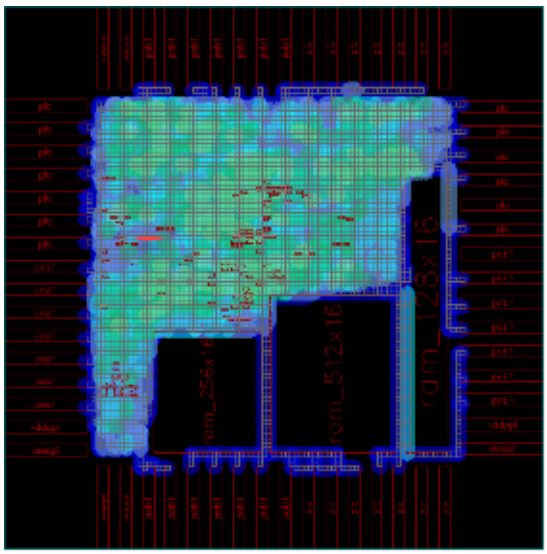
The congestion data result is created and displayed based on the histogram customization. Let us see how the results of congestion data differs in three different methods of histogram customization.

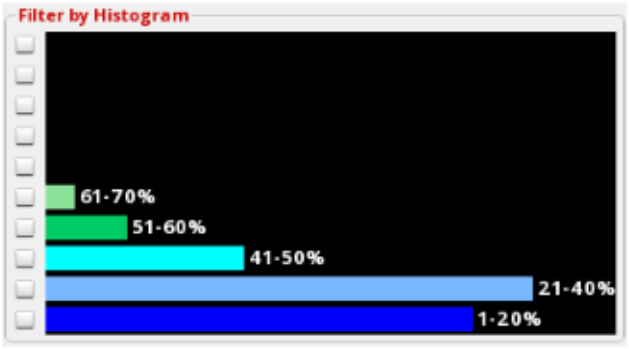
- Customized as Interval
- Customized as Start and End
- Customized as Specified

Customized as Interval

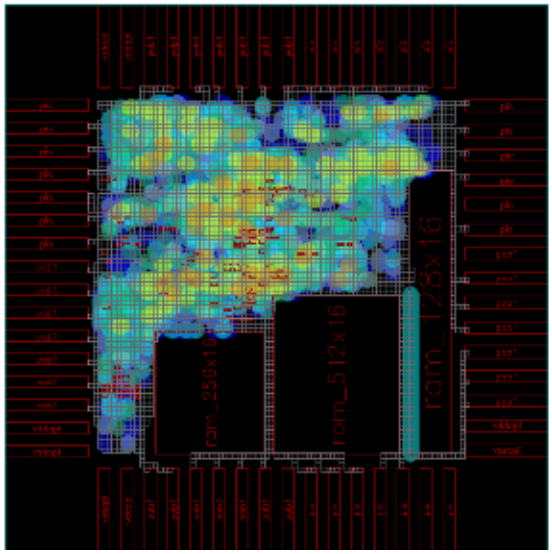
The *Interval* option in the Congestion Histogram Customize form lets you specify the congestion percentage to start and the interval between the congestion buckets.

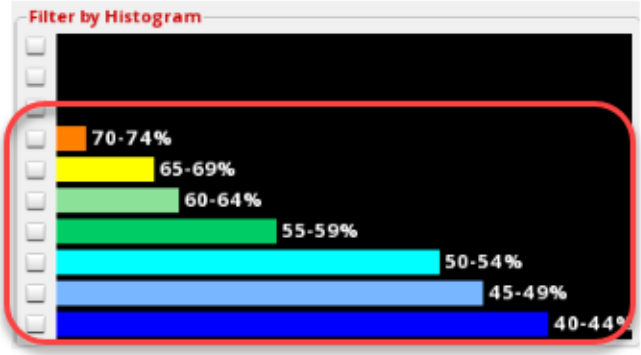
The following figure shows an example where the histogram starts at the 40% value with an interval of 5% in each bucket. The histogram is updated to only display congestion data for the upper buckets. As can be seen, most of the congestion is in the *40-44%* bucket.





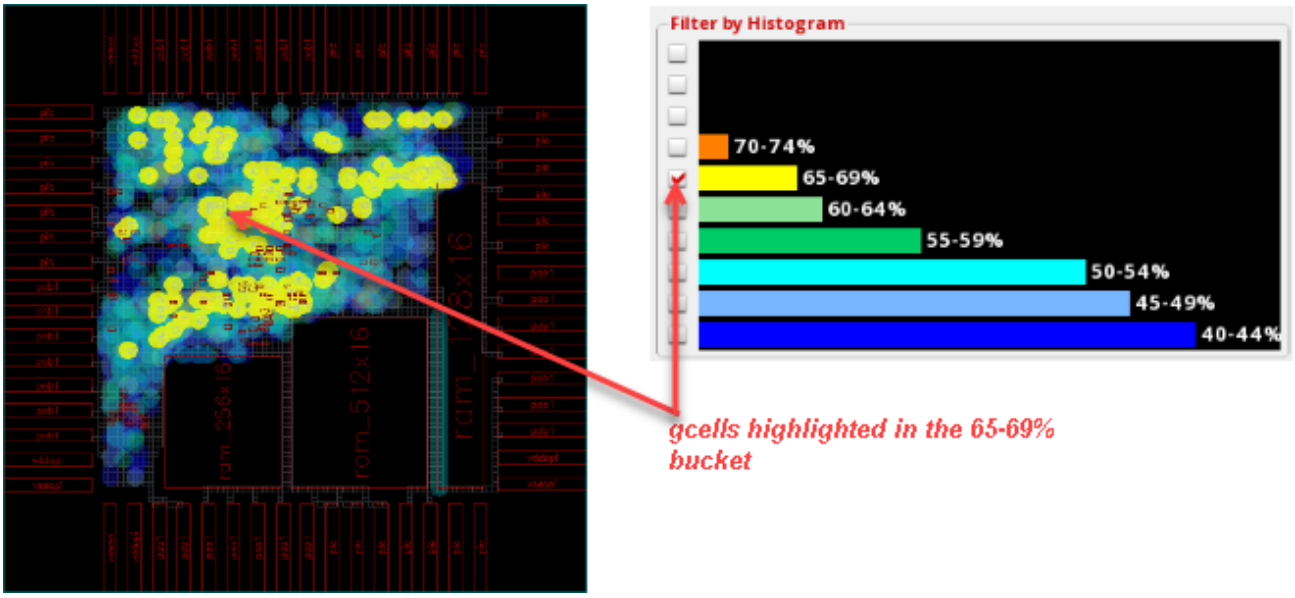
Default congestion data





Congestion data after the histogram is customized

Now, click the check box next to the 65-69% bucket. The gcells in this bucket are highlighted in the heat map and the global cell track utilization table.

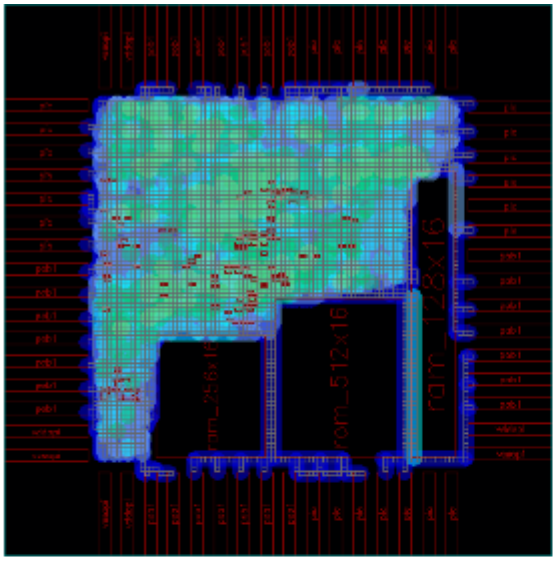


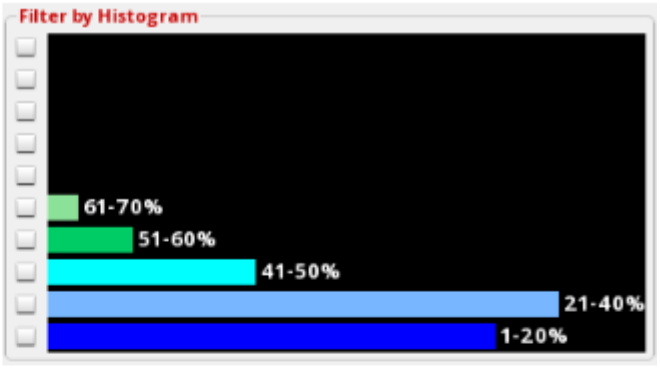
Customized as Start and End

The *Start and End* option in the Congestion Histogram Customize form lets you specify the congestion percentage to start and the congestion percentage to end. The intervals between congestion buckets are derived automatically.

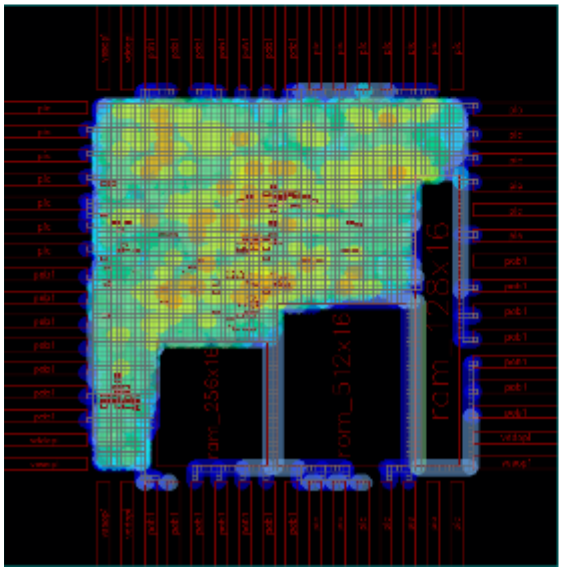
The following figure shows an example where the percentage to start is specified as 1% and the percentage to end is specified as 110%. The interval for each bucket is automatically set to 10%. The histogram is updated to only display congestion data for the specified buckets

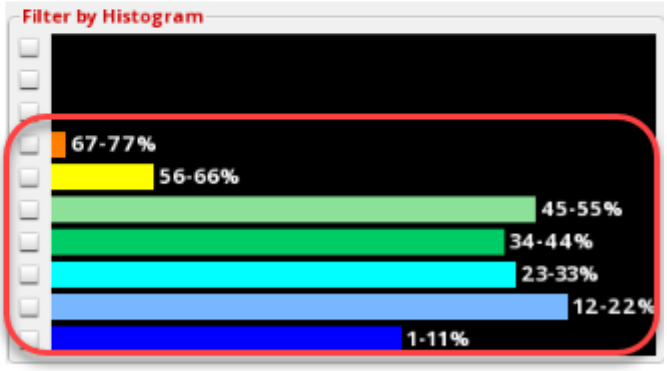
that were selected in the customization form. As can be seen, most of the congestion is in the *12-22%* bucket.





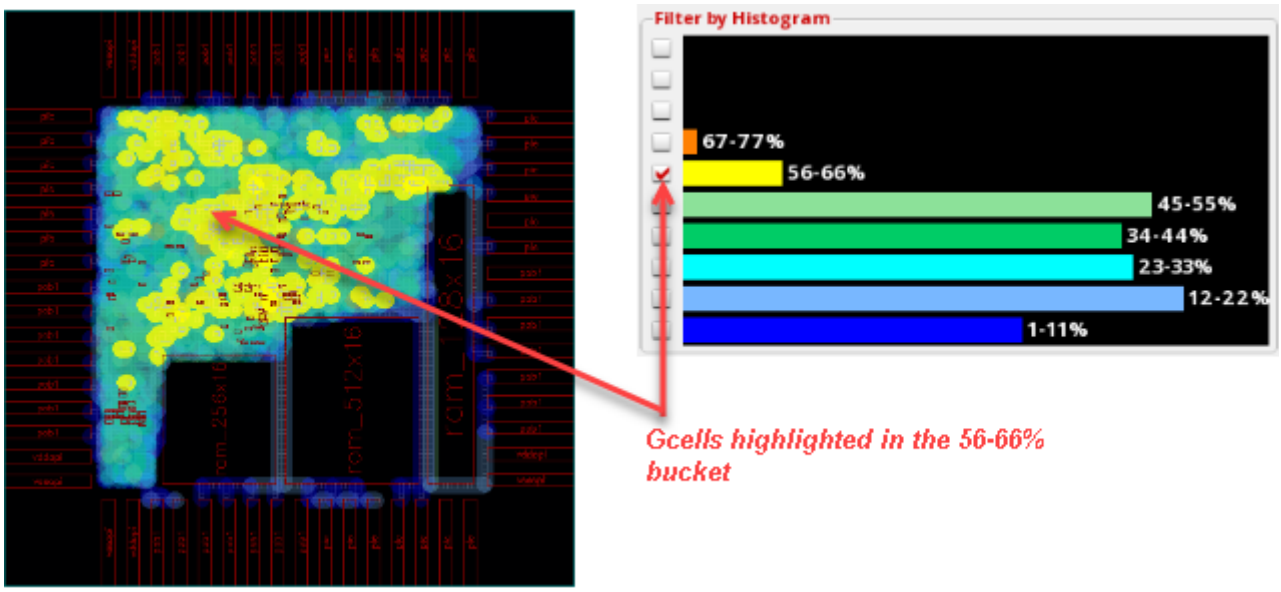
Default congestion data





Congestion data after the histogram is customized

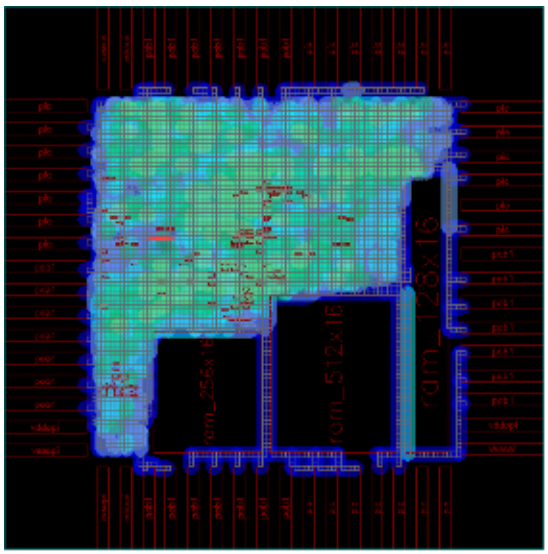
Now, click the check box next to the *56-66%* bucket. The gcells in this bucket are highlighted in the heat map and the global cell track utilization table.

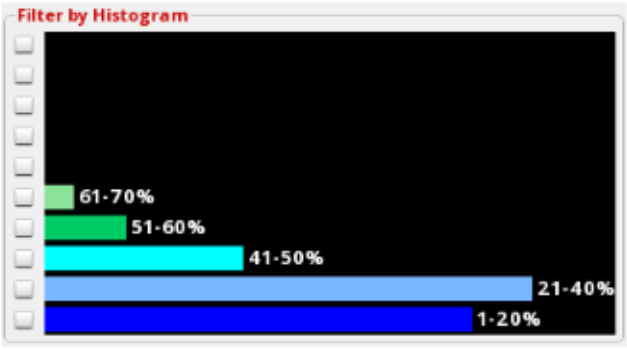


Customized as Specified

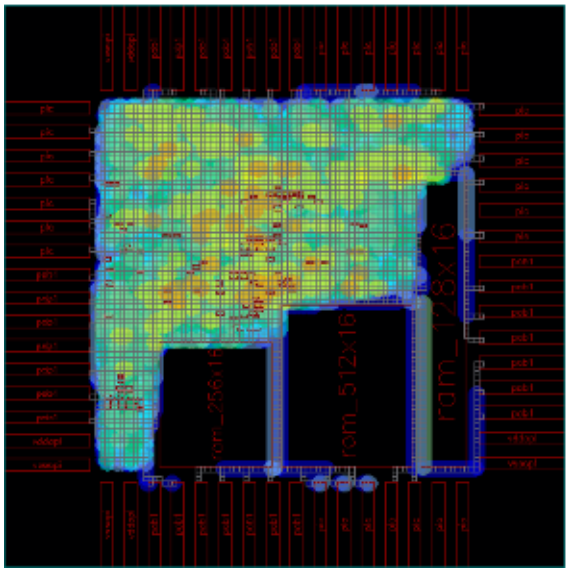
The *Specified* option in the Congestion Histogram Customize form lets you specify the exact congestion percentage. The start percentage for each bucket has to be individually specified.

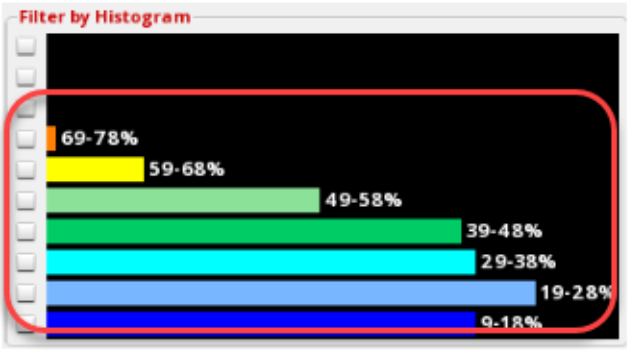
The following figure shows an example where the percentage to start the histogram is specified as 9%, the next congestion bucket starts at 19%, the one after that at 29%, and so on.





Default congestion data

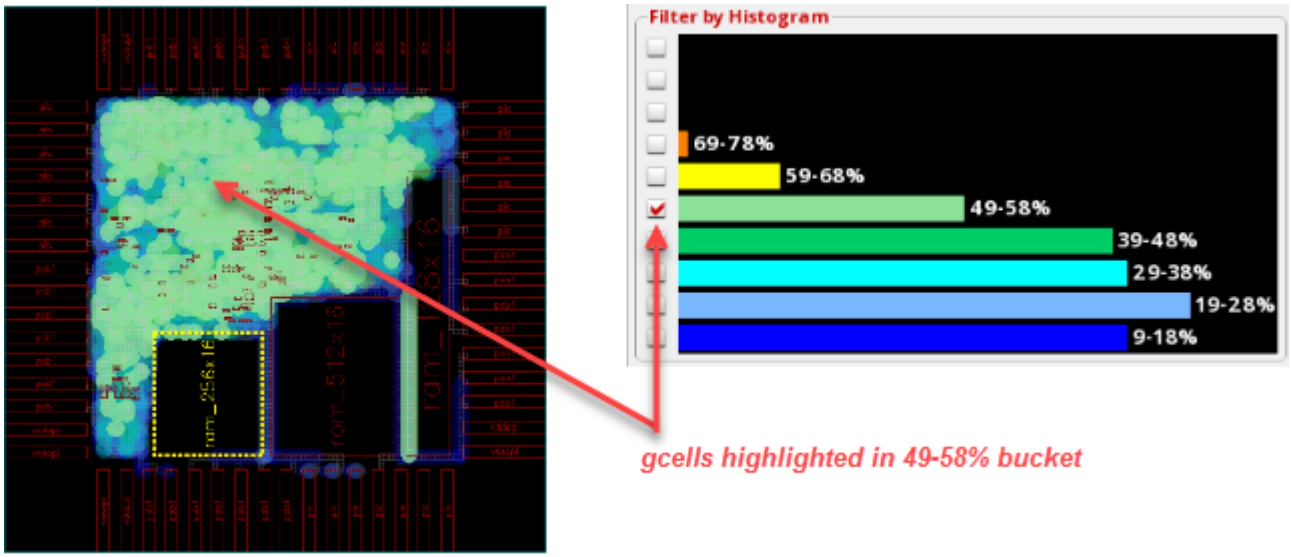




Congestion data after the histogram is customized

The congestion between *9-99%* is now visible in the histogram. As can be seen, most of the congestion is in the *19-28%* bucket. Also, there is hidden congestion in the *117-200%* bucket.

Now, click the check box next to the *49-58%* bucket to highlight the congested hot spots on the heat map and debug why the gcells are so congested by reviewing the global cell track utilization table.



Related Topics

Congestion Histogram Customize Form

Customizing a Histogram

Global Cells

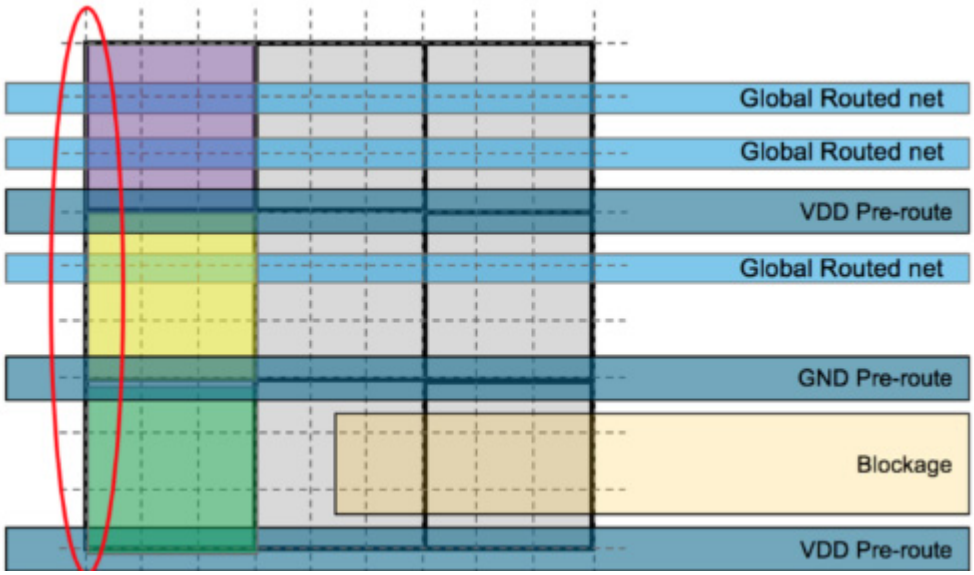
Global Routing generates a grid of global cells (known as gcells) in the design. The size of a global cell on a given layer is a multiple of the track pitch of that same layer. To calculate congestion, use the number of tracks that cross a gcell edge. This means:

- The capacity of a gcell edge is the total number of tracks that cross that edge.
- The availability of a gcell edge is the capacity minus the number of tracks already used on that edge.

To know the horizontal capacity of a gcell from the following figure, review the left-hand edges of the three highlighted gcells.

- Gcell #1 (in Purple) has a horizontal capacity of three tracks but has 0 tracks available (0% free)

- Gcell #2 (in Yellow) has a horizontal capacity of three tracks and has one track available (33% free)
- Gcell #3 (in Green) has a horizontal capacity of three tracks and has two tracks available (66% free)



Related Topics

Selecting and Viewing Global Cells

Displaying Empty Global Cells

Sorting the Global Cell Track Utilization Table

Selecting and Viewing Global Cells

With the help of the global cell track utilization table, you can view the actual congestion metrics used by the global routing engine.

You can use the global cell track utilization table to quickly find the most congested gcells in the design. In addition, explore the objects causing that level of congestion and also view the selected gcell(s) in the heat map by zooming in.

To do this:

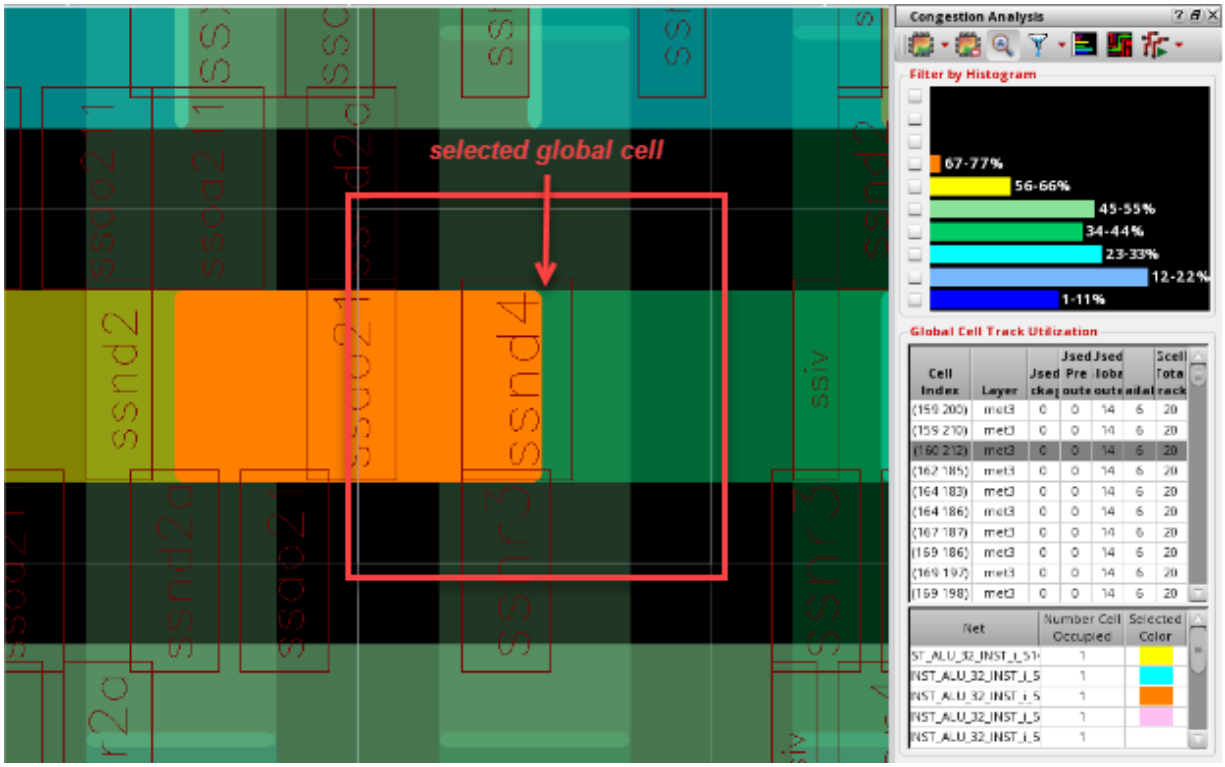
1. Run congestion analysis by clicking the *Congestion Analysis* icon on the toolbar.

2. Select a gcell from the *Global Cell Track Utilization* table in the Congestion Analysis assistant.

   The nets passing through the selected gcell are displayed in the section below the *Global Cell Track Utilization* table. The heat map is also updated to show the location of the selected gcell. However the selected gcell, in the Global Track Utilization table, appears quite small on the heat map because of the design size.

3. To view the selected gcell more precisely, use the *Zoom selected Global Cells* icon on the *Congestion Analysis* toolbar.

   The following figure shows the enlarged view of the selected gcell in the heat map.

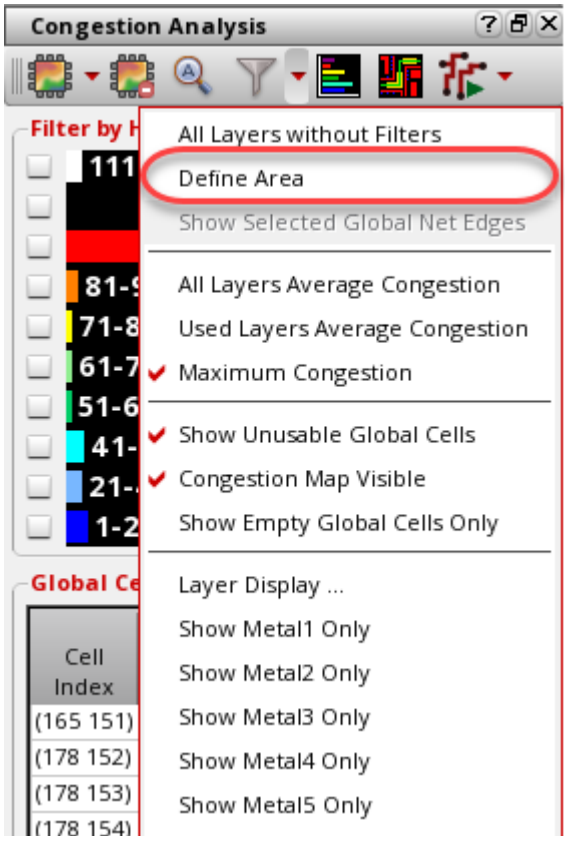


   Congestion is calculated based on the capacity and availability of edges on a gcell.

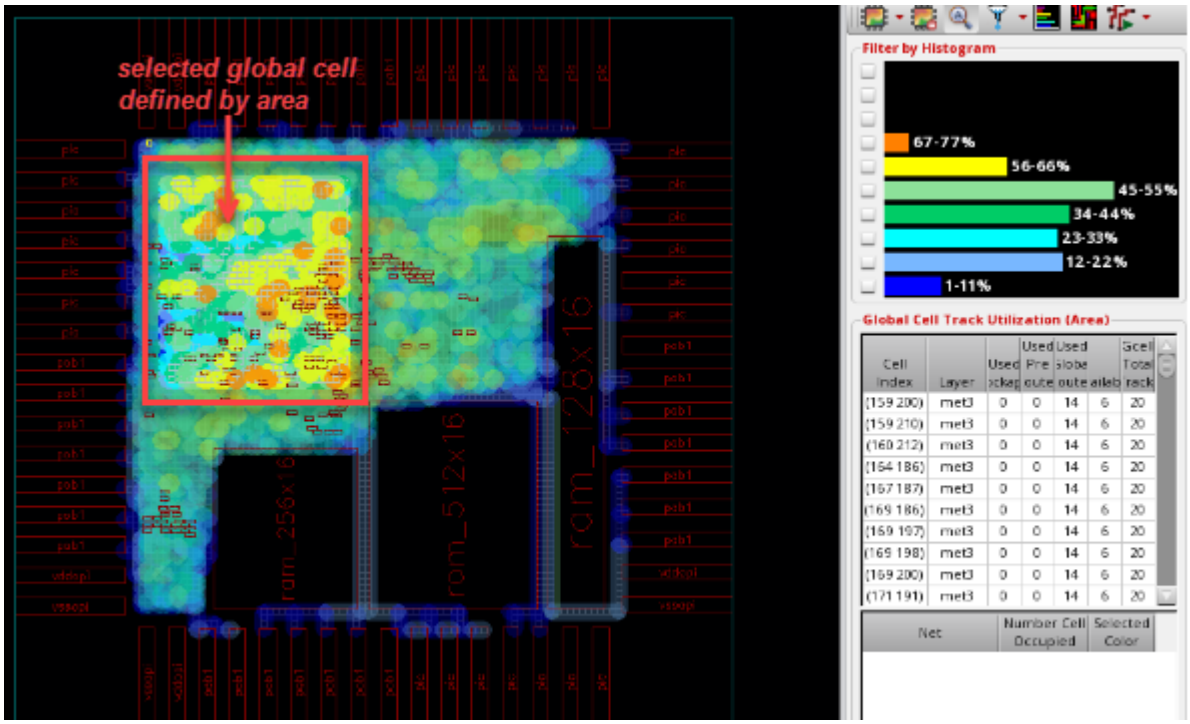
   - ❑ Horizontal congestion is calculated based on the left edge of a gcell.
   - ❑ Vertical congestion is calculated based on the bottom edge of a gcell.
   - ❑ The congestion on the right and top edges of the selected gcell are reflected in the neighboring gcell above it and to the right of the selected gcell.

An alternative way to select and view a gcell is using the heat map. You can look for the gcells in the heat map and then manually select the gcells that are required. To do this:

1. Click the drop-down arrow next to the *Filter Global Cells By* icon on the *Congestion Analysis* toolbar.
2. To select an area of gcells, click *Define Area*.



3. Now, select a region on the heat map by clicking a lower-left point and then dragging the pointer to an upper-right point.



The non-selected gcells are dimmed. Also, the histogram and the global cell track utilization table is updated to indicate only the selected gcells.

Related Topics

Global Cells

Displaying Empty Global Cells

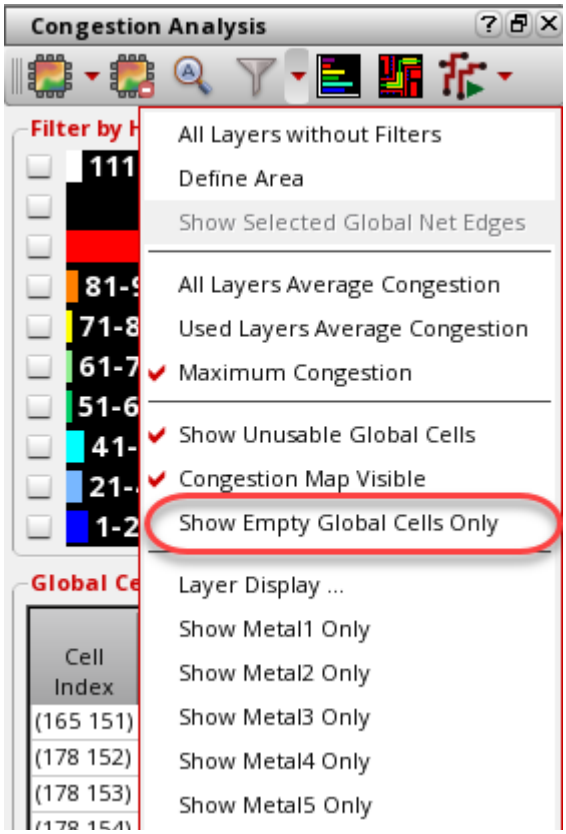
Sorting the Global Cell Track Utilization Table

Displaying Empty Global Cells

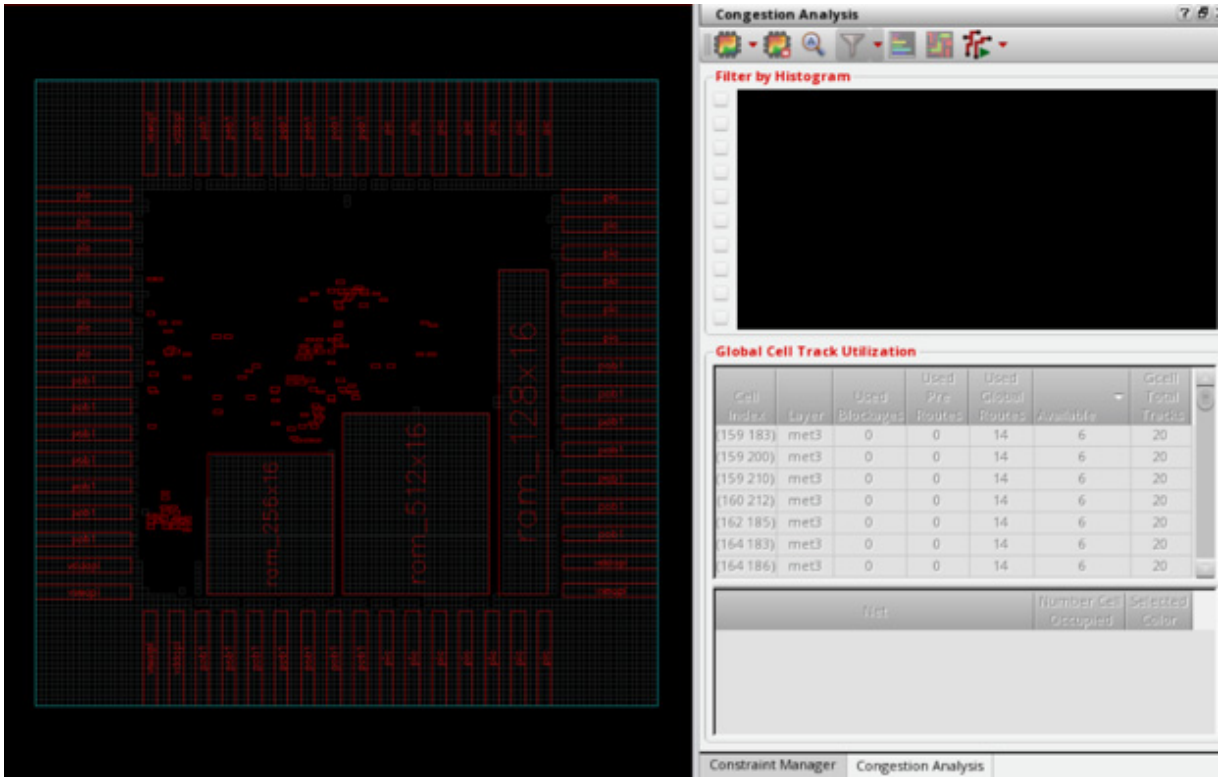
At times, it is useful to visualize parts of a design that have zero percent congestion. This can help in identifying regions that could be used to add additional structures such as hard macros, soft blocks, or routing. Alternatively, this can help to identify regions in a design that could be removed or compacted in order to shrink a design during floor-planning. To only show gcells with zero percent congestion, perform the following steps.

1. Click the *Filter Global Cells by* icon from the Congestion Analysis toolbar.

2. Click the *Show Empty Global Cells Only* option from the drop-down list.



The heat map is turned off and only global cells with zero congestion are displayed in the main window.



The histogram and the Global Cell Track Utilization table are inactive and are not filterable or customizable during this mode.

Related Topics

Global Cells

Selecting and Viewing Global Cells

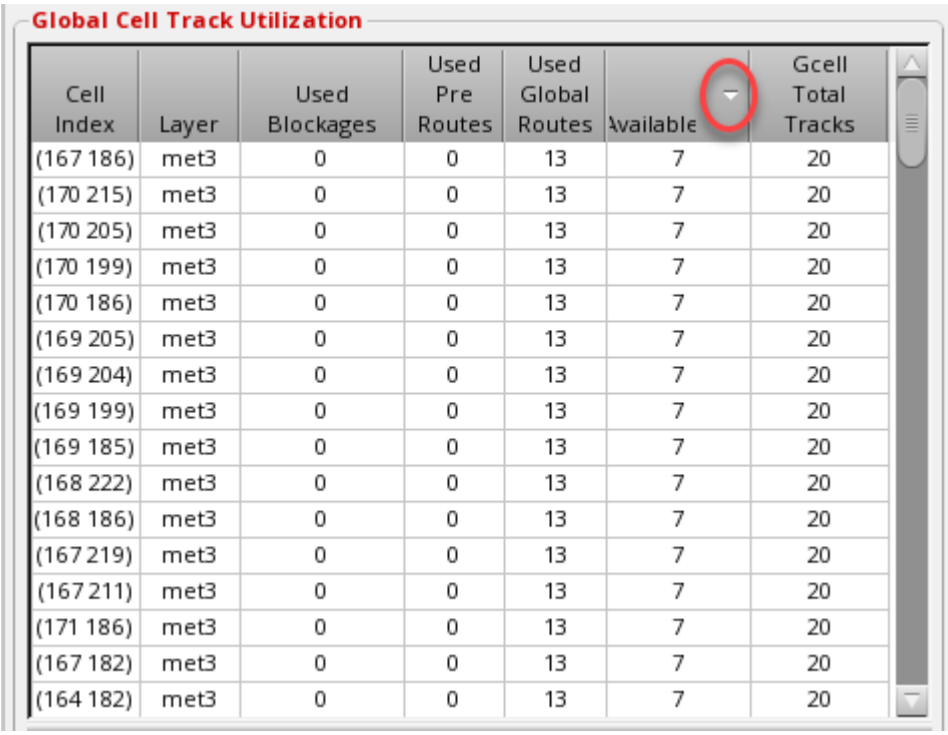
Sorting the Global Cell Track Utilization Table

Sorting the Global Cell Track Utilization Table

To assist in the analysis of congestion, you can sort each column in the Global Cell Track Utilization table. To do this, click the header of the column to be sorted.

To reverse the sorting of the column, double-click the column.

In the following figure, the *Available* column in the Global Cell Track Utilization table is sorted in descending order. The triangle on the column name represents whether the sorted column is ascending or descending order.



Cell Index	Layer	Used Blockages	Used Pre Routes	Used Global Routes	Available	Gcell Total Tracks
(167 186)	met3	0	0	13	7	20
(170 215)	met3	0	0	13	7	20
(170 205)	met3	0	0	13	7	20
(170 199)	met3	0	0	13	7	20
(170 186)	met3	0	0	13	7	20
(169 205)	met3	0	0	13	7	20
(169 204)	met3	0	0	13	7	20
(169 199)	met3	0	0	13	7	20
(169 185)	met3	0	0	13	7	20
(168 222)	met3	0	0	13	7	20
(168 186)	met3	0	0	13	7	20
(167 219)	met3	0	0	13	7	20
(167 211)	met3	0	0	13	7	20
(171 186)	met3	0	0	13	7	20
(167 182)	met3	0	0	13	7	20
(164 182)	met3	0	0	13	7	20

Related Topics

Global Cells

Displaying Empty Global Cells

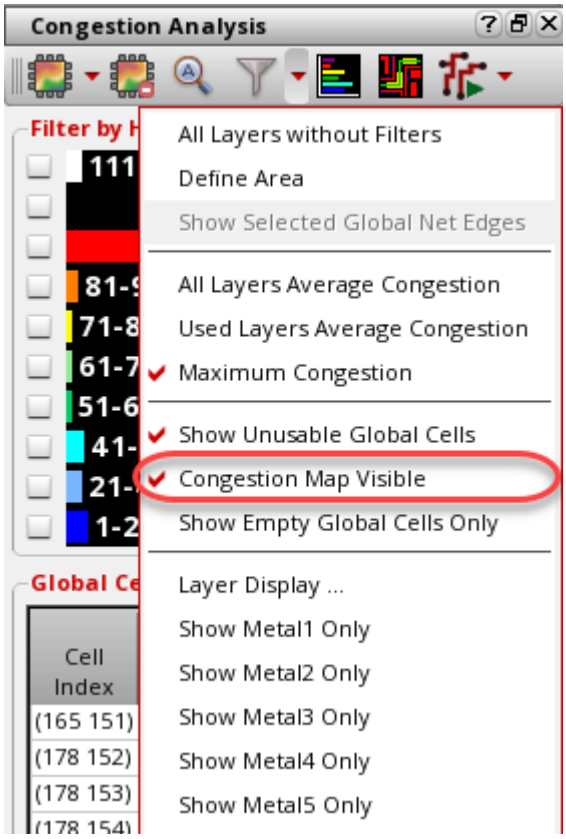
Displaying Empty Global Cells

Toggling Congestion Map Visibility

Sometimes, the heat map may obscure other important design information in the main window. In such a situation, you can toggle the display of the heat map in the main window. To do this:

1. Click the *Filter* icon from the Congestion Analysis toolbar.

2. Click the *Congestion Map Visible* option from the drop-down list.



The display of congestion map is turned off and the congestion map is no longer displayed.

The histogram and the Global Cell Track Utilization table are inactive and are not filterable or customizable during this mode.

Related Topics

Sorting the Global Cell Track Utilization Table

Finding and Displaying Nets

To know where congestion hot spots exist, it is important to identify the nets passing through congested regions of a design so that corrective action can be taken. This section describes how to:

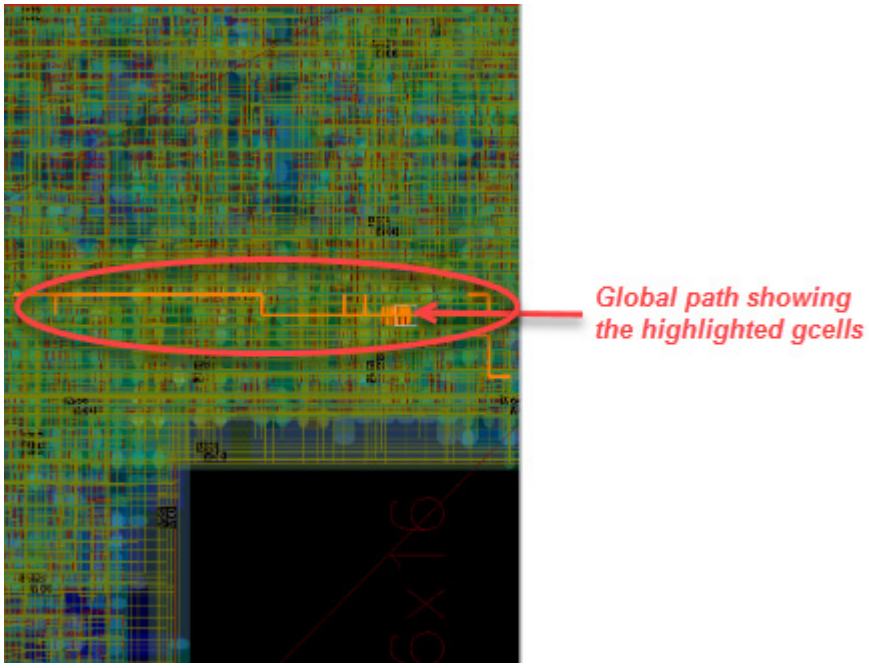
- Find nets passing through gcells.

- Visualize the selected nets as global paths in the heat map.

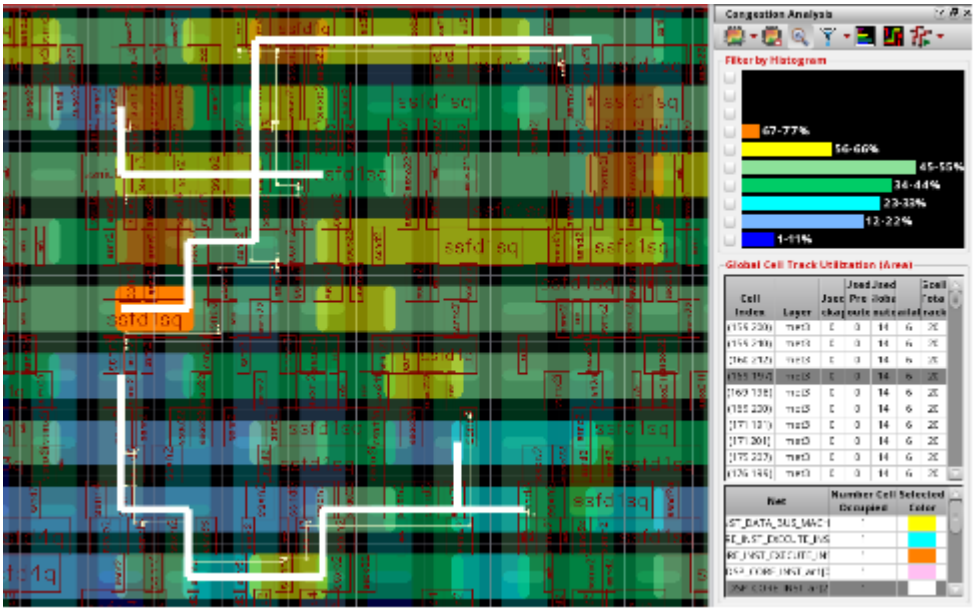
To find and display the nets that are passing through a gcell:

1. Run congestion analysis by clicking the *Congestion Analysis* icon on the toolbar.
2. Select a gcell in the *Global Cell Track Utilization* table.
3. Select a net from the table of nets displayed below the *Global Cell Track Utilization* table.

   All the gcells that the selected net passes through are highlighted. The path displayed by the highlighted gcells is known as the global path. The global path of the selected gcells is seen on the heat map.

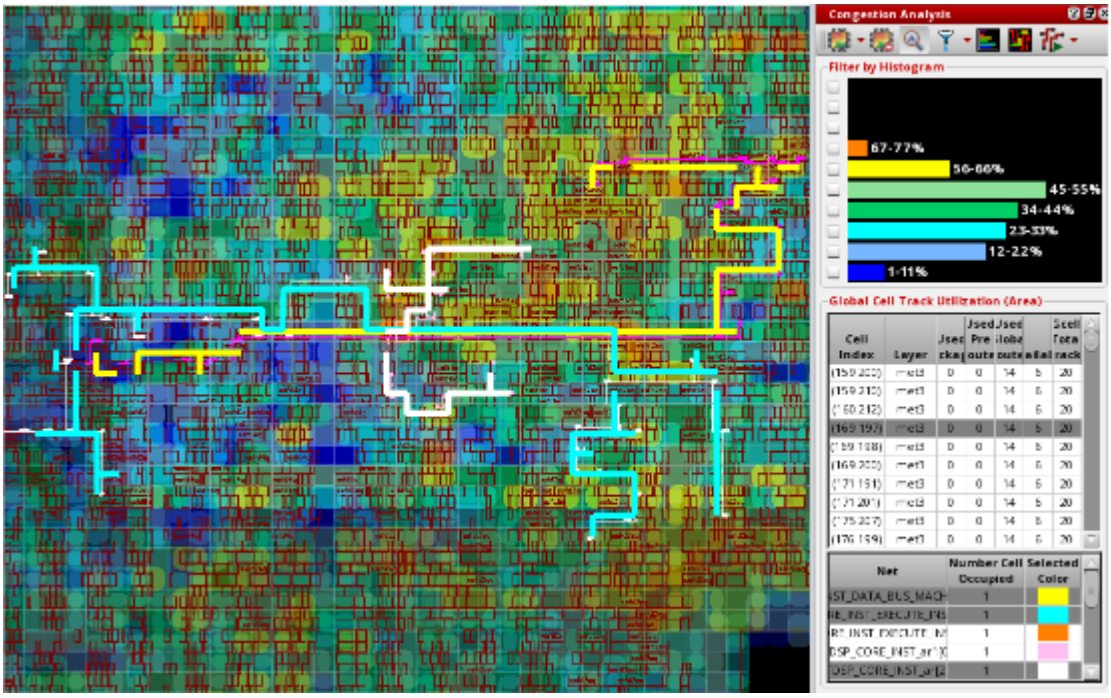


4. To view the path more closely, use the *Zoom selected Global Cells* icon on the *Congestion Analysis* toolbar.



5. Now, select a range of nets from the *Net Selection* table by clicking the start and end nets and keeping the `Shift` key pressed. You can also select multiple nets from the table by clicking multiple nets and keeping the `Ctrl` key pressed.

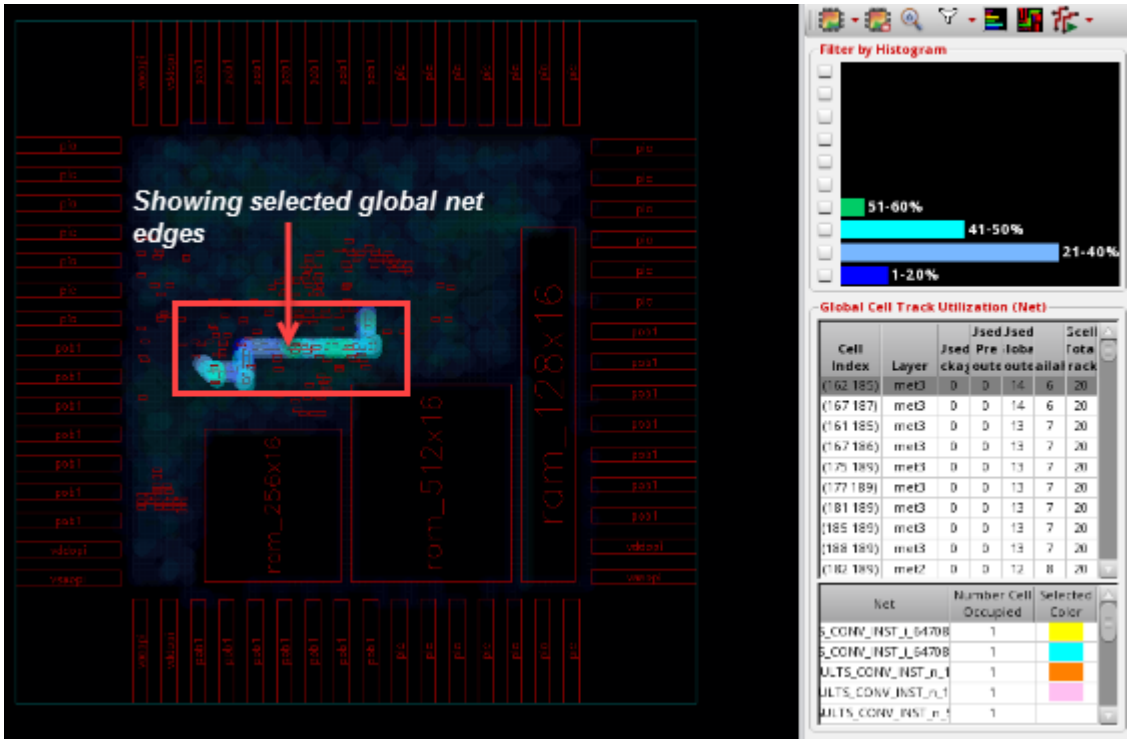
When multiple nets are selected, the display of gcells for the selected nets look cluttered. This is because the heat map complicates the global paths of the selected nets.



The selection of nets is usually done from the Navigator assistant and is considered the primary way of displaying nets on the heat map. However, you can also select a net or a group of nets from the *Net Selection* table in the Congestion Analysis assistant and have the selected nets displayed in the Navigator assistant.

6. To only show the congestion on gcells associated with the global paths of the selected nets, click the *Show Selected Global Nets Edges* option from the *Filter* drop-down

list. The following figure shows how the congestion is displayed before the option is not selected and after the option is selected.



Related Topics

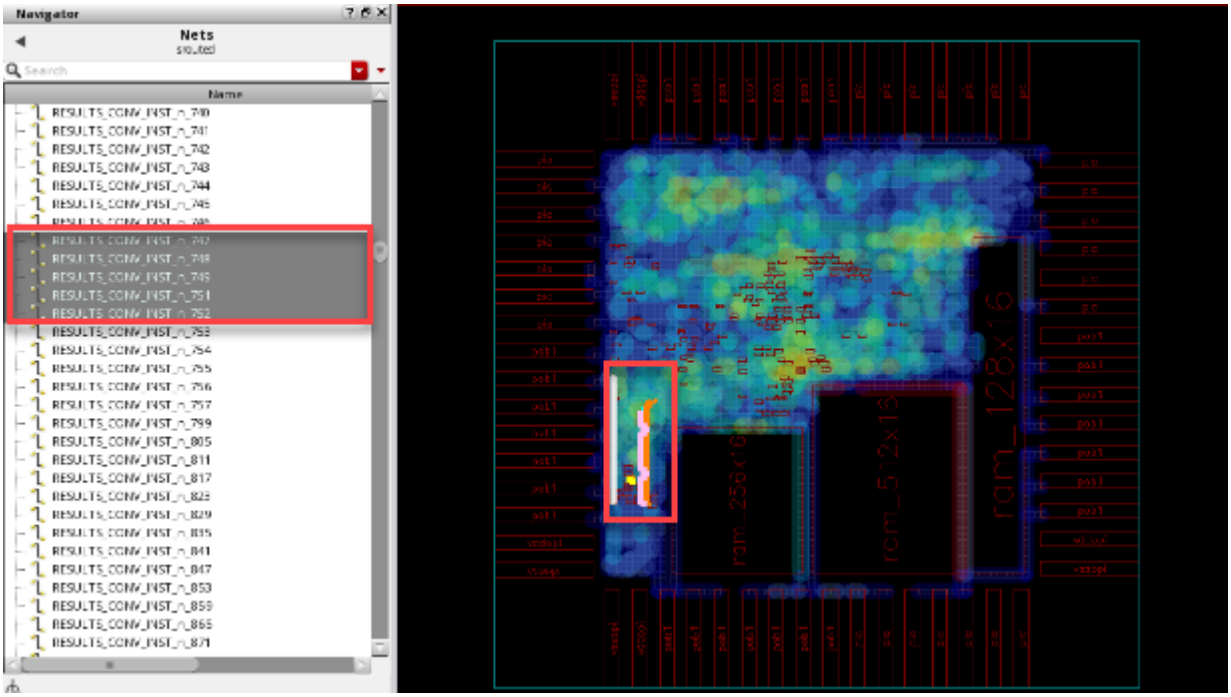
Finding and Displaying Nets in the Navigator

Finding and Displaying Nets in the Navigator

You can select nets from the Navigator assistant and have them displayed in the heat map. To do this:

1. Open the Navigator assistant in Virtuoso by choosing *Windows – Assistant – Navigator*.
2. Click and select the nets from the Navigator assistant.

The following figure shows the selected nets in the Navigator assistant and how they are displayed together in the heat map.



Related Topics

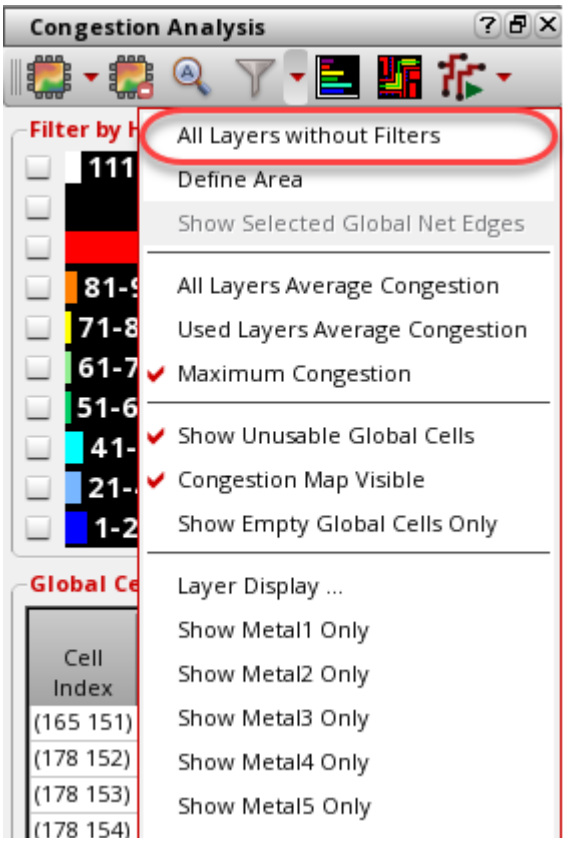
Finding and Displaying Nets

Resetting the Histogram and Heat Map Display

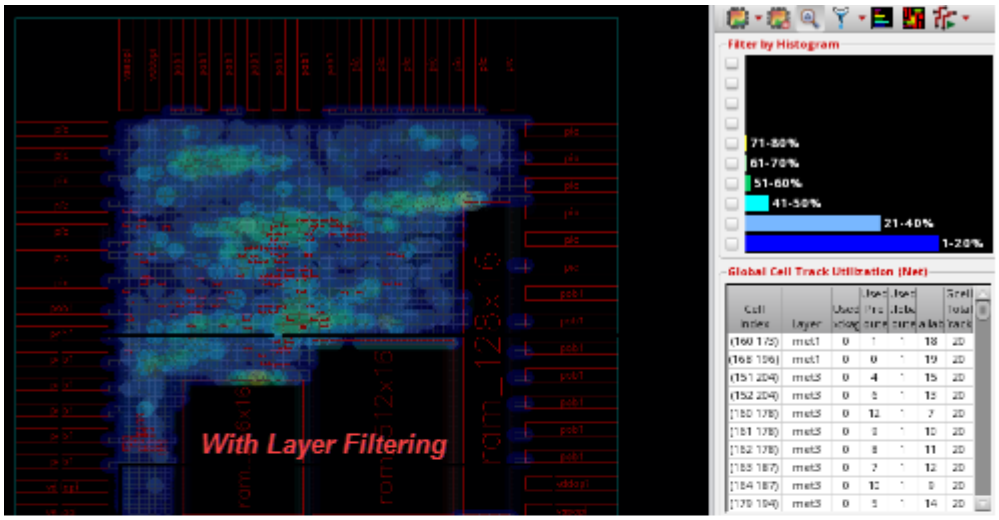
To reset the histogram and heat map to display every gcell, you need to reset the filtering. To do this:

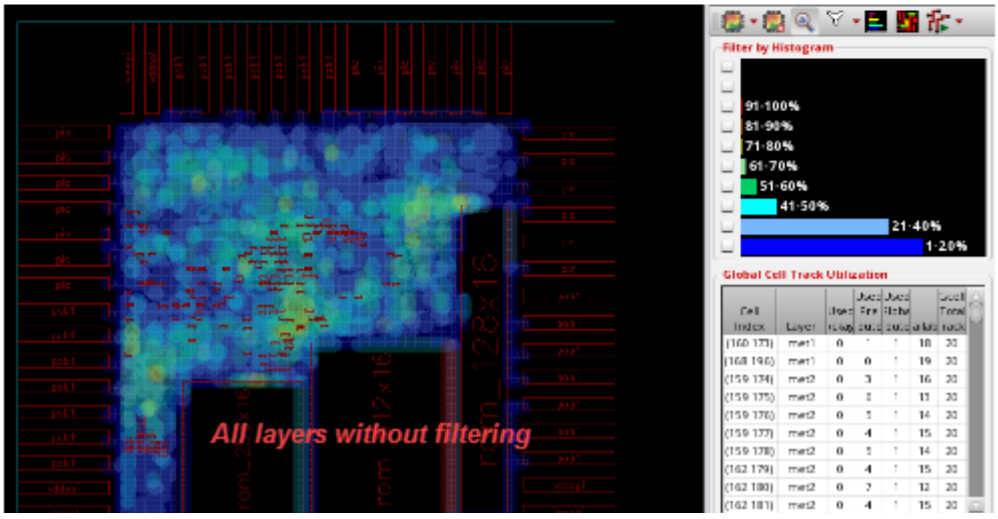
1. Click the *Filter* icon from the Congestion Analysis toolbar.

2. Click the *All Layers without filtering* option from the drop-down list.



The following figure shows the congestion analysis results when the *All Layers without filtering* option is selected.





Related Topics

Customizing a Histogram

Results Based on Histogram Customization

Global Bias Constraints

Visualizing and analyzing routing is effective only when you are able to make the changes to better suit the design. Congestion Analysis provides the support of global bias constraints, which can be used to plan routing. Once you have identified critical nets or groups of nets, you can add global bias constraints to manage and plan the routing path. These global bias constraints are persistent and are saved to OpenAccess.

The various aspects of biasing and how to see the actual results by running ECO global routing and congestion analysis are as follows:

- Creating a Global Bias Positive Region Constraint (attract net group into a region)
- Creating a Global Bias Negative Region Constraint (repel net group out of a region)
- Creating a Global Bias Path Constraint (guide net group along a path)
- Creating Multiple Global Bias Constraints for Specific Layers

Related Topics

Global Bias Setup Form

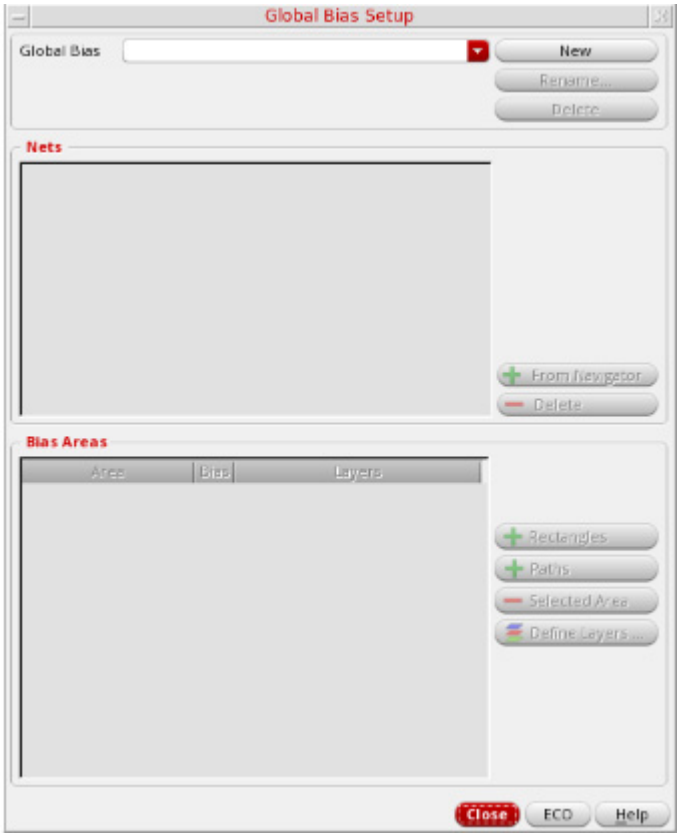
Creating a Global Bias Positive Region Constraint

You can consider the global bias positive region constraint as a magnet that can attract a group of nets and buses into a region during global routing. These biased nets take priority over the other nets that also pass through the same region. This means that the global router may push aside the other nets to ensure that the biased nets pass through the region successfully.

To create a global bias positive region:

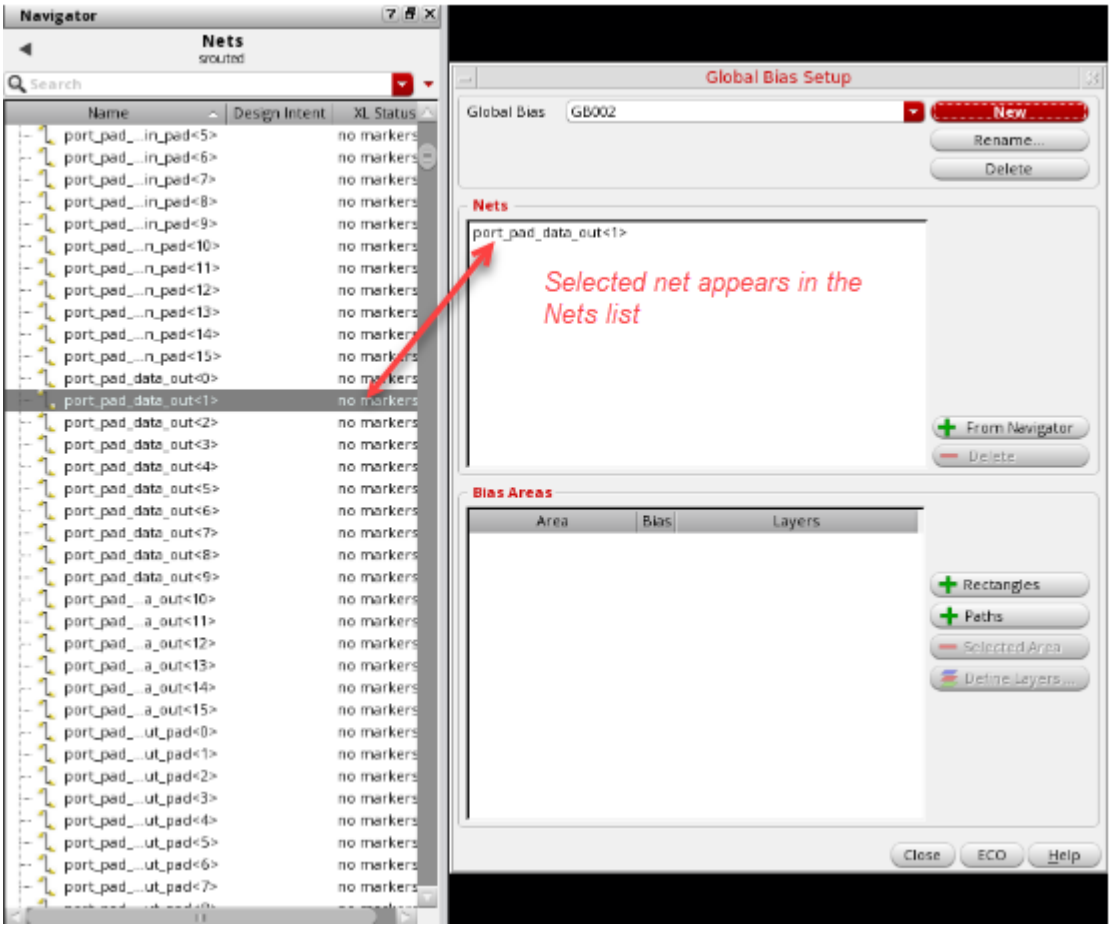
1. Select a net, group of nets, or a bus from the Navigator assistant. For example, select the net *port_pad_data_out<1>* from the Navigator assistant.
2. Click the *Global Bias Setup* icon  on the Congestion Analysis assistant toolbar.

The Global Bias Setup form displays.



3. Click the *New* button to create a new global bias constraint group.

The form is updated to create a default group named as *GB002*. This is an auto-generated global bias name. Also, the net selected in the Navigator assistant appears in the *Nets* list.



You can add and remove the nets to the Global Bias Setup form.

To add nets:

a. Select a net or a group of nets from the Navigator assistant.

b. Click the *+ From Navigator* button in the Global Bias Setup form.

The nets selected in the Navigator assistant get displayed in the *Nets* list box.

To remove nets:

a. Select a net from the *Nets* list box in the Global Bias Setup form.

b. Click the *- Delete* button.

The selected net is removed from the *Nets* list box.

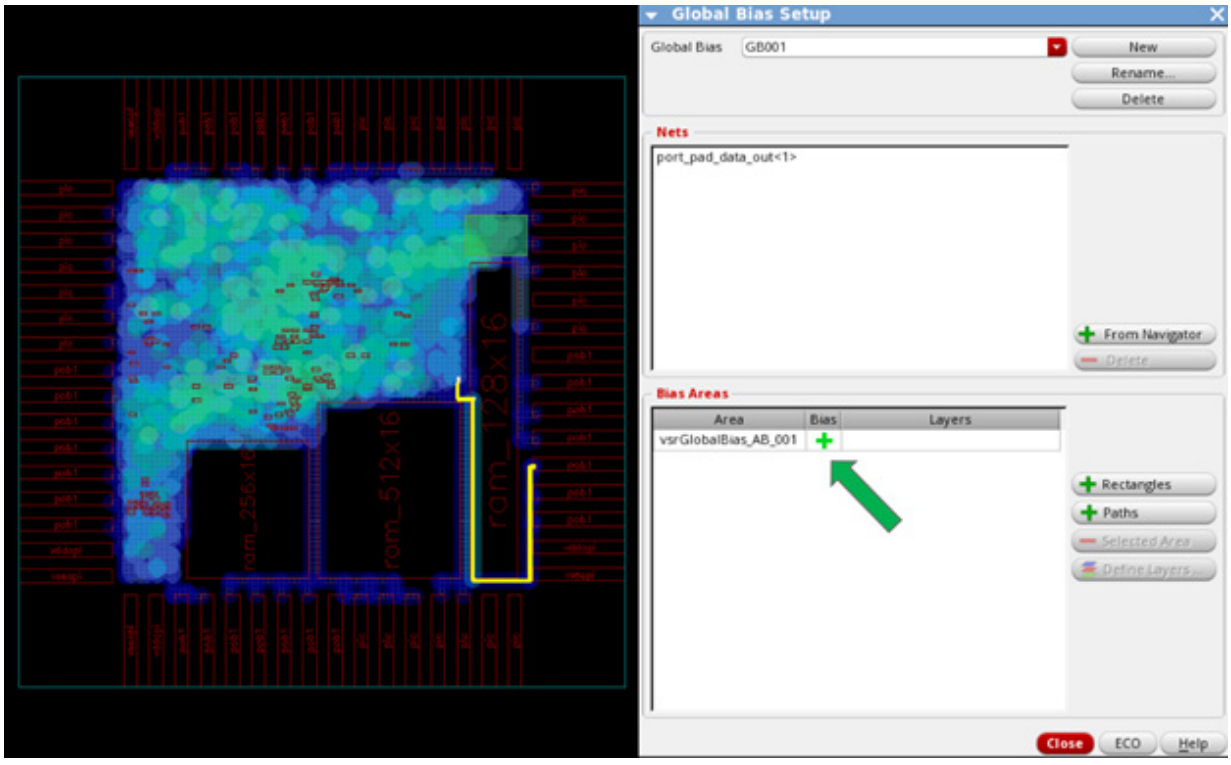
4. Rename the default global bias constraint name to a user-defined name. To edit the name, click in the *Global Bias* text field and specify another name.

   You can add a net or group of nets selected in the Navigator Assistant to the renamed global bias constraint. This is done using the *+ From Navigator* button in the Global Bias Setup form.

5. To add the positive bias region, click the *+ Rectangles* button.

6. Click and drag to draw a region in the heat map.

   The region created on the heat map is used as the coordinates for the bias area. The bias area is automatically assigned a name called *vsrGlobalBias_AB_001* and is displayed in the *Bias Areas* list box. The Green ± symbol in the *Bias* column next to the bias area indicates that it is a positive bias.



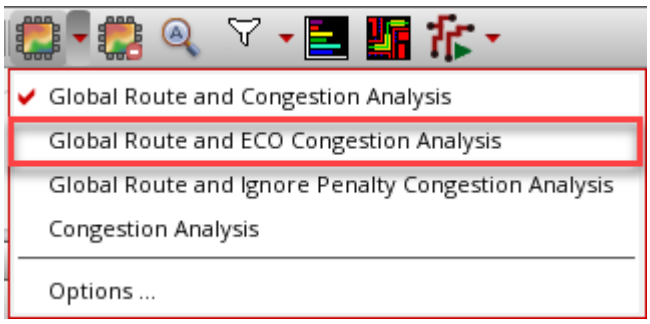
7. If you have created an incorrect region, you can delete it. To do so:

   a. Select the bias area from the *Bias Areas* list box.

   b. Click the *– Selected Area* button.

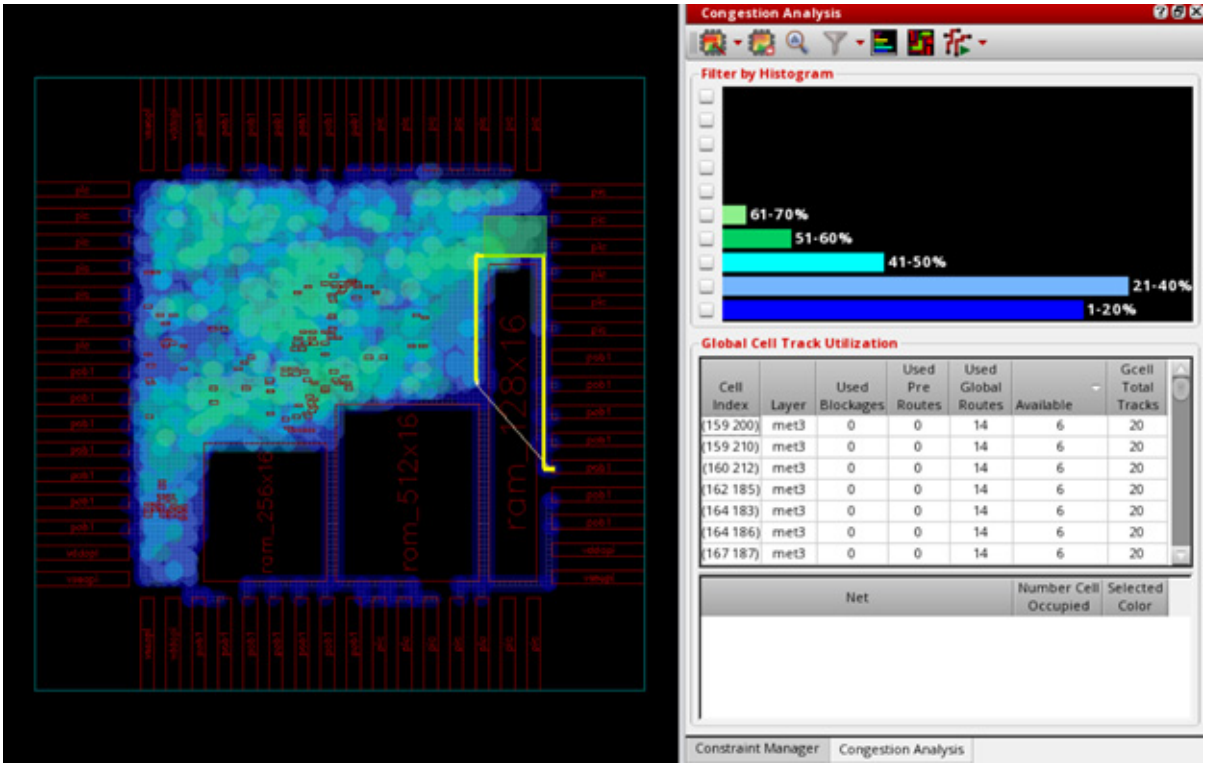
   The selected bias area is removed.

8. To see how the new constraint alters routing and congestion, click the *ECO* button in the Global Bias Setup form. Alternatively, click *Global Route and ECO Congestion Analysis* option from the *Congestion Analysis* drop-down list.



The *ECO* button in the Global Bias Setup form is enabled only when there is a valid global bias constraint.

The following figure shows that the net is now attracted through the bias region and no longer takes the circuitous path.



Related Topics

Global Bias Setup Form

Creating a Global Bias Negative Region Constraint

Creating a Global Bias Path Constraint

Creating Multiple Global Bias Constraints for Specific Layers

Creating a Global Bias Negative Region Constraint

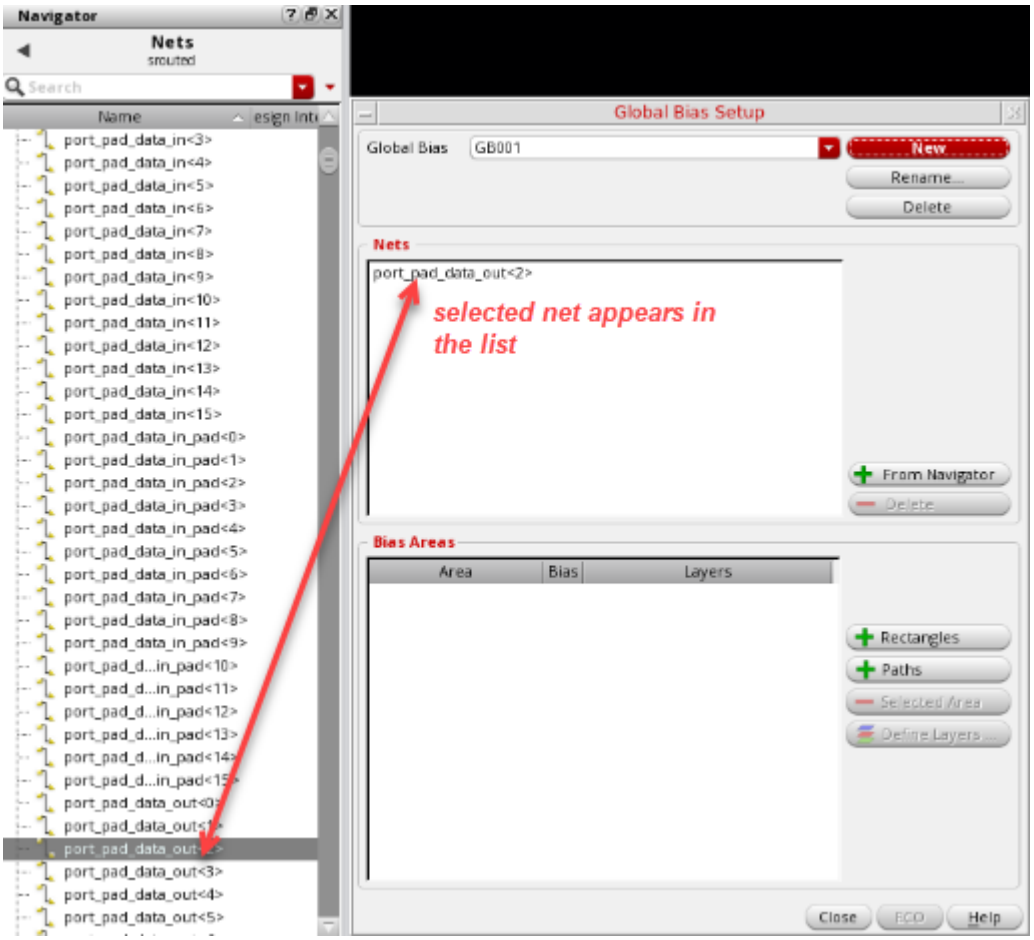
Consider the global bias negative region constraint as a magnet that can repel a group of nets and buses out of a region during global routing. These biased nets take priority in being pushed out of a bias region by global routing. This can result in the biased nets becoming more circuitous. Another way to conceptualize the negative bias region is as a soft blockage for a selected set of nets.

To create a global bias negative region, perform the following steps:

1. Select a net, group of nets, or a bus from the Navigator assistant. For example, select the net *port_pad_data_out<1>* from the Navigator assistant.
2. Click the *Global Bias Setup* icon  on the *Congestion Analysis* assistant toolbar.

   The Global Bias Setup form displays.
3. Click *New* to create a new global bias constraint group.

   The form is updated to create a default group named as *GB002*. This is an auto-generated global bias name. Also, the net selected in the Navigator assistant appears in the *Nets* list box.



You can add and remove the nets to the Global Bias Setup form.

To add nets:

a. Select a net or a group of nets from the Navigator assistant.

b. Click the *+ From Navigator* button in the Global Bias Setup form.

The nets selected in the Navigator assistant are displayed in the *Nets* list box.

To remove nets:

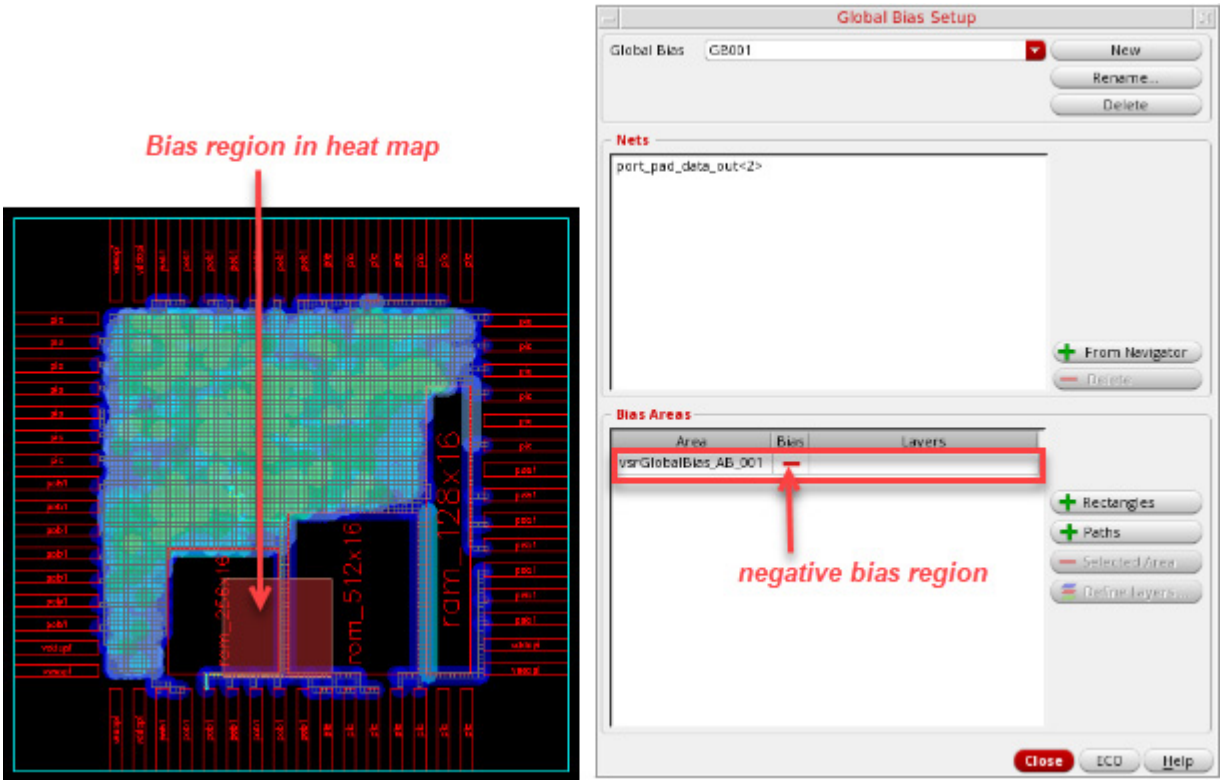
a. Select a net from the *Nets* list box in the Global Bias Setup form.

b. Click the *- Delete* button.

The selected net is removed from the *Nets* list box.

4. To edit the global bias constraint group name, click in the *Global Bias* text field and specify another name.
5. To add the negative bias region, click the *+ Rectangles* button.
6. Click and drag to draw a region in the heat map.

   The region created on the heat map is used as the coordinates for the Bias Area. The bias area is automatically assigned a name called *vsrGlobalBias_AB_002* and is displayed in the *Bias Areas* list box.
7. Click the green ± symbol in the *Bias* column next to the bias area. This displays a red — symbol, which indicates a negative bias.



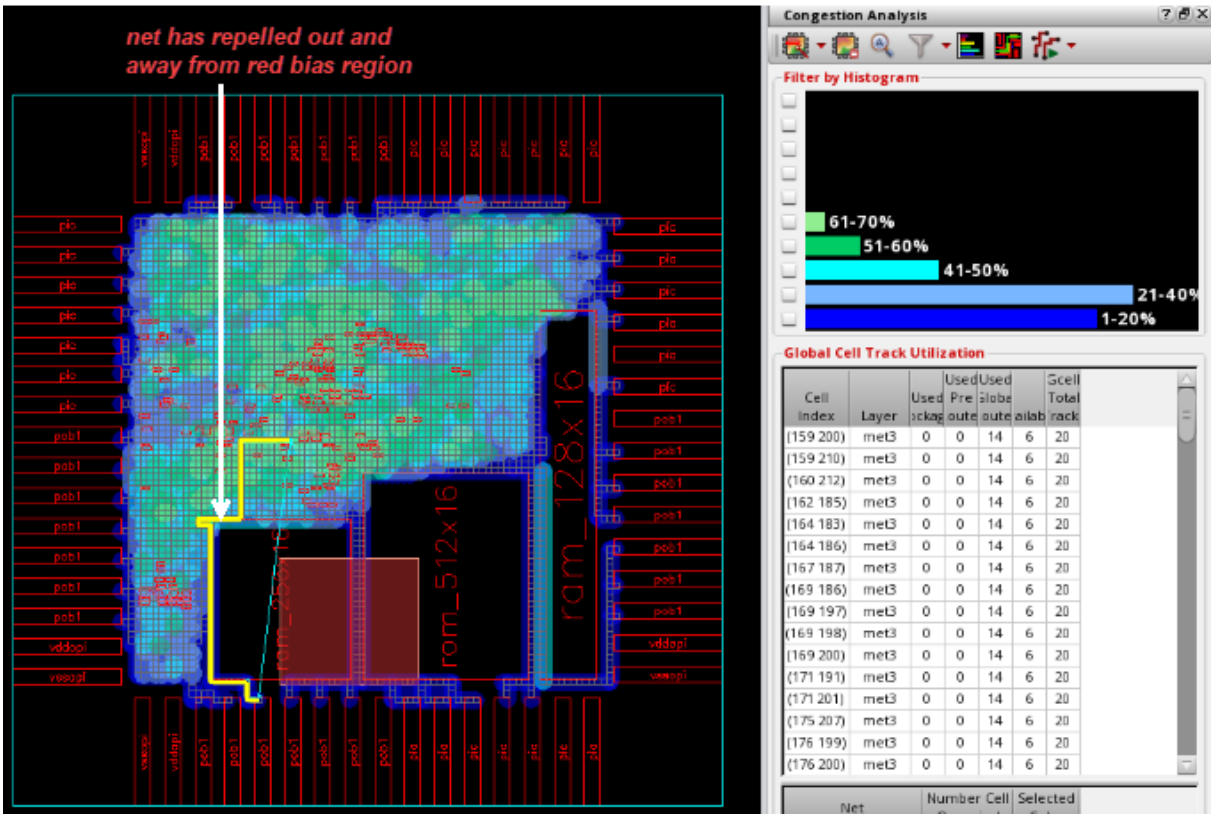
8. If you have created an incorrect region, you can delete it. To do so:

   a. Select the bias area from the *Bias Areas* list box.

   b. Click the *– Selected Area* button.

   The selected bias area is removed.
9. To see how the new constraint alters routing and congestion, click the *ECO* button in the Global Bias Setup form or click the G*lobal Route and ECO Congestion Analysis* option from the *Congestion Analysis* drop-down list.

The following figure shows that the net is now repelled from the red bias region and can route freely.



Related Topics

Global Bias Setup Form

Creating a Global Bias Positive Region Constraint

Creating a Global Bias Path Constraint

Creating Multiple Global Bias Constraints for Specific Layers

Creating a Global Bias Path Constraint

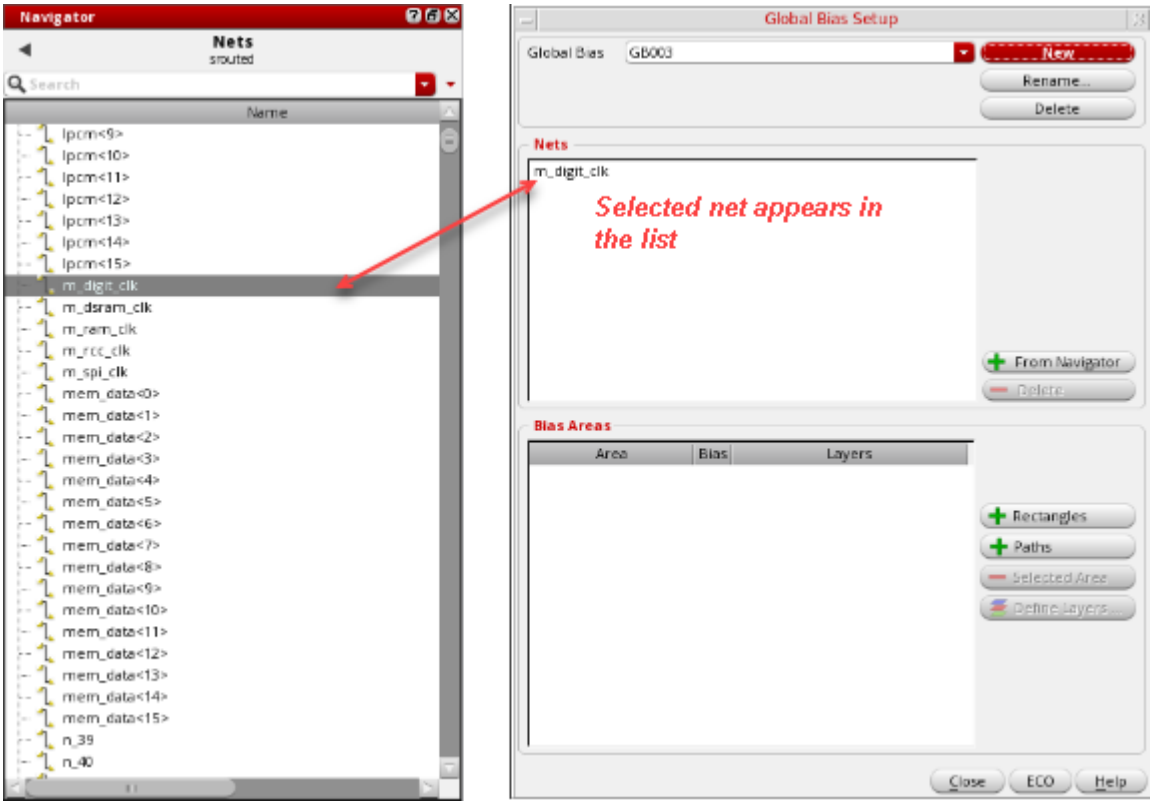
You can use global bias path constraints to guide a net, group of nets, or a bus along a pre-designated path during global routing. These path-biased nets take priority over the other nets during global routing. This means that the global router may push other nets out of the way and make them less optimal in routing.

Creating a path is similar to creating a polygon. To create a global bias path constraint:

1. Select a net, group of nets, or a bus from the Navigator assistant. For example, select the net *m_digit_clk* from the Navigator assistant.
2. Click the *Global Bias Setup* icon on the *Congestion Analysis* assistant toolbar.

   The Global Bias Setup form displays.
3. Click the *New* button to create a new global bias constraint group.

   The form is updated to create a default group named as *GB003*. This is an auto-generated global bias name. Also, the net selected in the Navigator assistant appears in the *Nets* list box.



You can add and remove the nets to or from the Global Bias Setup form.

To add nets:

a. Select a net or a group of nets from the Navigator assistant.

b. Click the *+ From Navigator* button in the Global Bias Setup form.

The nets selected in the Navigator assistant get displayed in the *Nets* list box.

To remove nets:

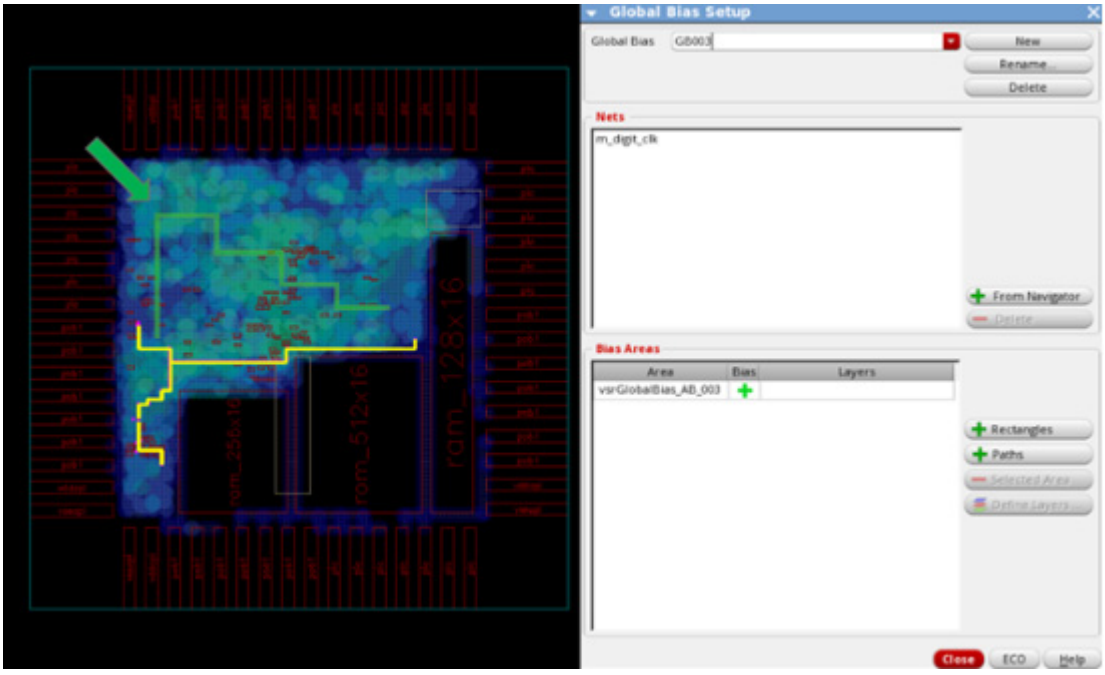
a. Select a net from the *Nets* list box in the Global Bias Setup form.

b. Click the - *Delete* button.

The selected net is removed.

4. To edit the global bias constraint group name, click in the *Global Bias* text field and specify another name.
5. To add the path bias constraint, click the *+ Paths* button.
6. Click and drag to draw a path that you want the net to take in the heat map.

   The path created on the heat map is used as the coordinates for the bias path. The bias path is automatically assigned a name called *vsrGlobalBias_AB_003* and is displayed in the *Bias Areas* list box.



If you have created an incorrect region, you can delete it. To do so:

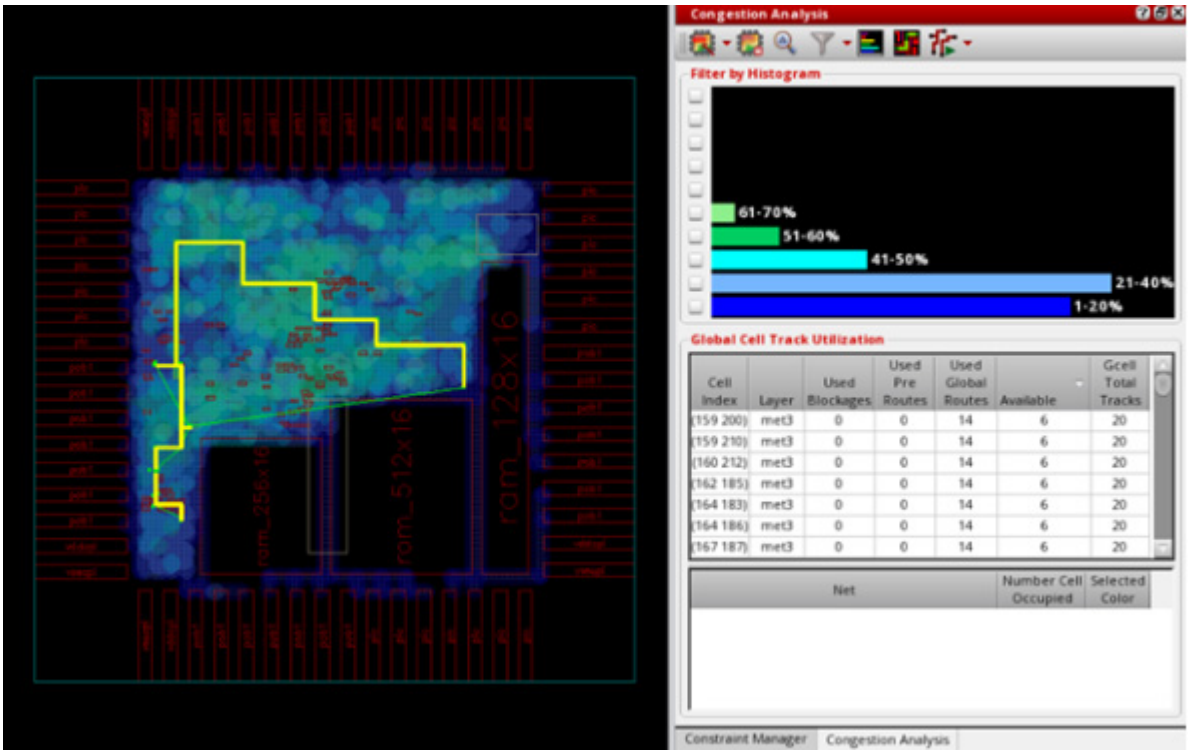
a. Select the bias area from the *Bias Areas* list box.

b. Click the – *Selected Area* button.

The selected bias area is removed.

7. To see how the new constraint alters routing and congestion, click the *ECO* button in the Global Bias Setup form or click the *Global Route and ECO Congestion Analysis* option from the *Congestion Analysis* drop-down list.

   The following figure shows that the net now follows the bias path constraint.



Related Topics

Global Bias Setup Form

Creating a Global Bias Negative Region Constraint

Creating a Global Bias Positive Region Constraint

Creating Multiple Global Bias Constraints for Specific Layers

Creating Multiple Global Bias Constraints for Specific Layers

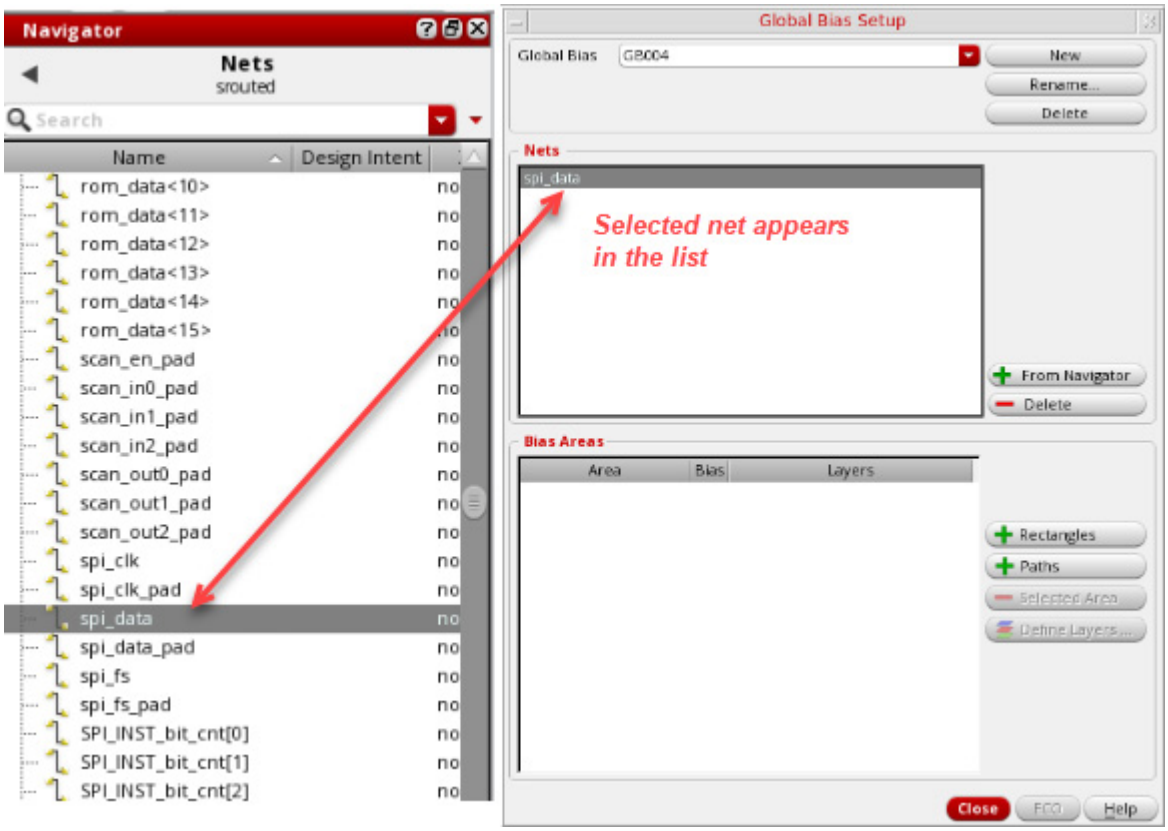
This section describes how to create a negative bias region constraint, create a bias path constraint on a specified set of layers and then run ECO global routing and congestion analysis.

To do this:

1. Select a net, group of nets, or a bus from the Navigator assistant. In this example, the net *spi_data* is selected from the Navigator assistant.
2. Click the *Global Bias Setup* icon on the *Congestion Analysis* assistant toolbar.

   The Global Bias Setup form displays.
3. Click the *New* button to create a new global bias constraint group.

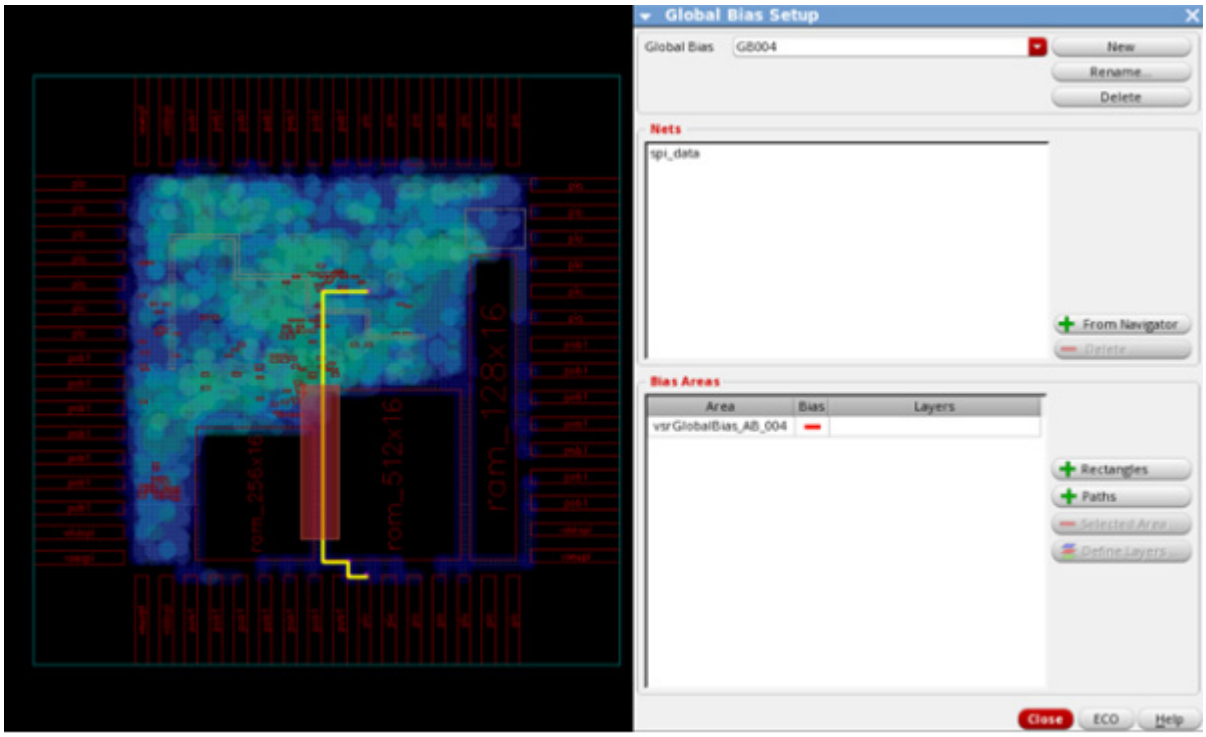
   The form is updated to create a default group named as *GB004*. This is an auto-generated global bias name. Also, the net selected in the Navigator assistant appears in the *Nets* list box.



4. To add the negative bias region, click the *+ Rectangles* button.
5. Click and drag to draw a region in the heat map.

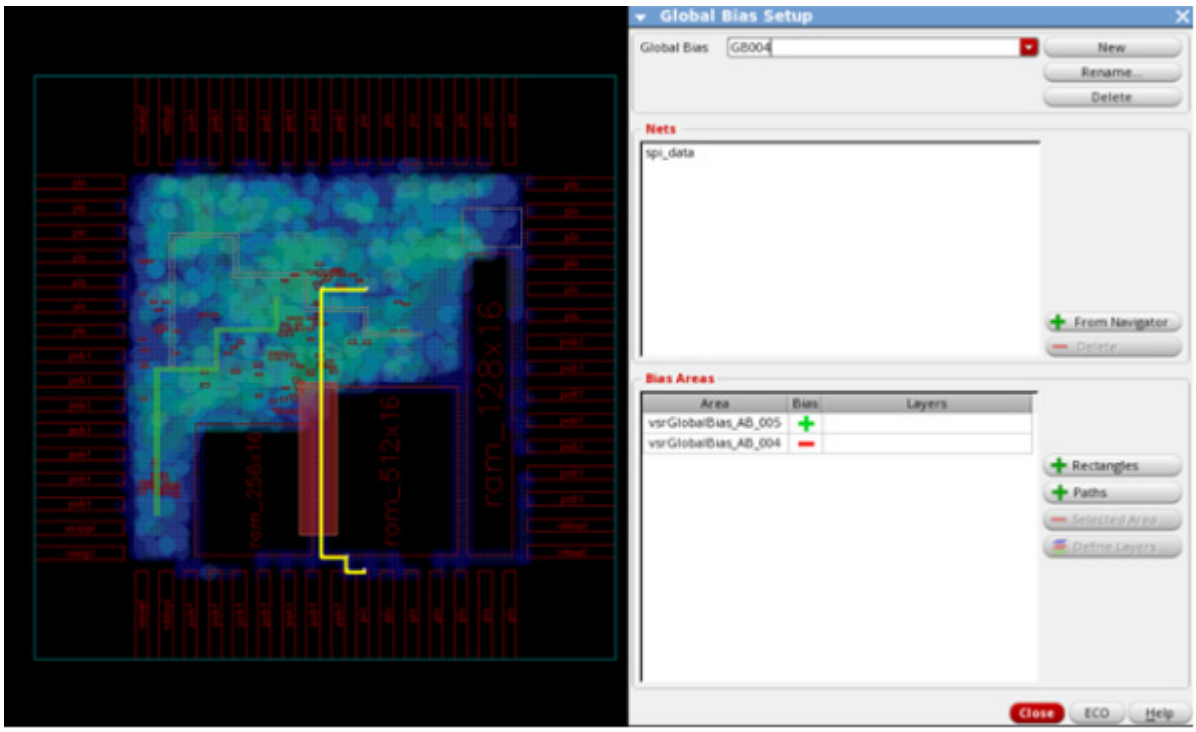
   The region created on the heat map is used as the coordinates for the bias area. The bias area is automatically assigned a name called *vsrGlobalBias_AB_004* and is displayed in the *Bias Areas* list box.

6. To create a negative bias, click the green ± symbol in the *Bias* column next to the bias area. This displays a red — symbol, which indicates a negative bias.



7. To add the bias path, click the *+ Paths* button.
8. Click and drag to draw a path that you want the net to take in the heat map.

The path created on the heat map is used as the coordinates for the bias path. The bias path is automatically assigned a name called *vsrGlobalBias_AB_005* and is displayed in the *Bias Areas* list box.

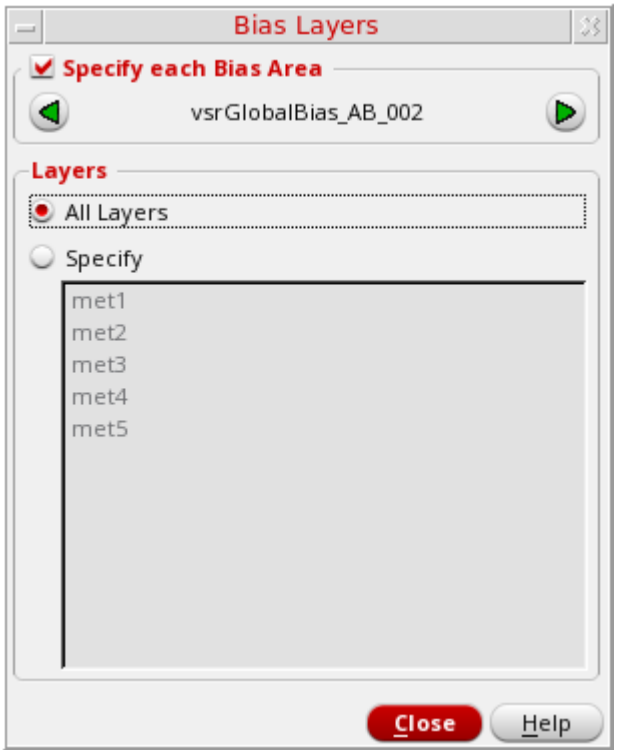


9. Now, let us assign specific layers for the bias path. To do so:

   a. In the Global Bias Setup form, click the *vsrGlobalBias_AB_005* bias area. The *Selected Area* and the *Define Layers* button are now enabled.

   b. Click *Define Layers*.

The Bias Layers form displays.



c. Click the *Specify* option.

The list box listing the layer names is now enabled.

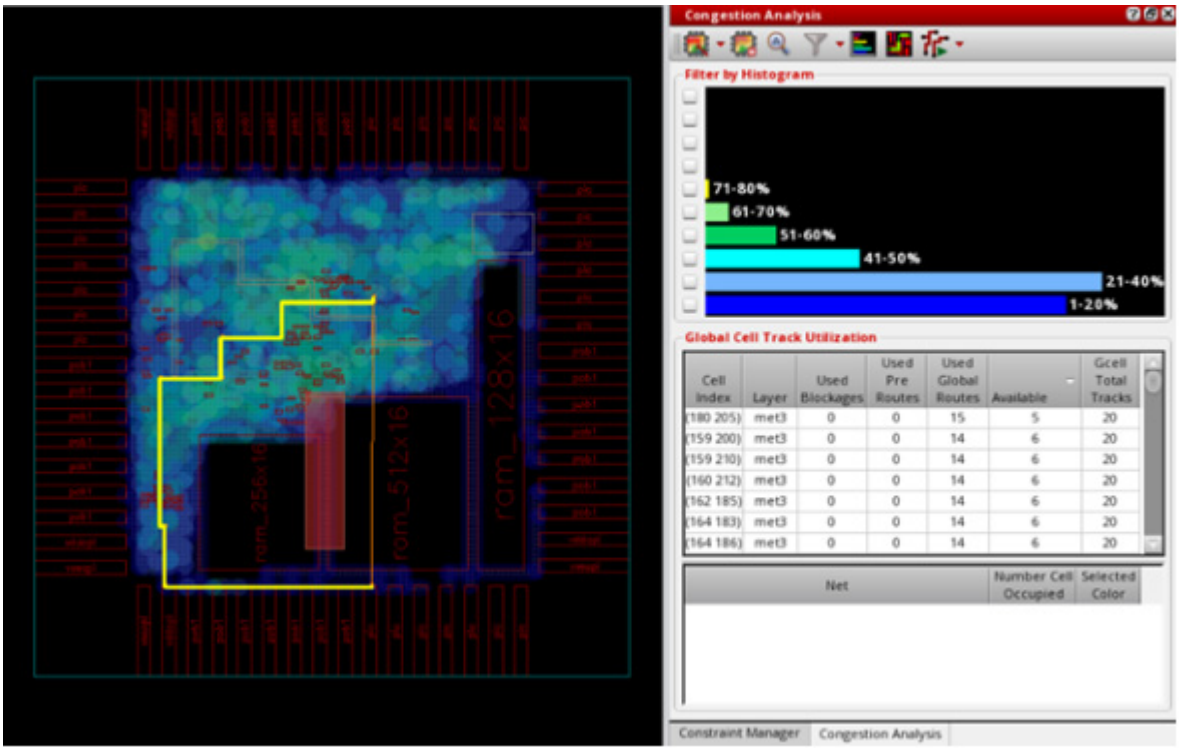
d. Select *the* required layers from the list box by holding the `Shift` key. This notifies the router that the global bias path must be completed only for the selected routing layers.

e. Click *Close*.

The Global Bias Setup form is updated to reflect the layer constraints applied to the `vsrGlobalBias_AB_005` bias area.

10. To see how the new constraint alters routing and congestion, click the *ECO* button in the Global Bias Setup form or click the *Global Route and ECO Congestion Analysis* option from the *Congestion Analysis* drop-down list.

The following figure shows that the net has now been routed outside of the negative bias region and routed along the bias path.



Related Topics

Global Bias Setup Form

Bias Layers Form

Creating a Global Bias Negative Region Constraint

Creating a Global Bias Path Constraint

Creating a Global Bias Positive Region Constraint

Managing Scenic Ratio

Scenic ratio is used to determine how close the value is to the ideal route when comparing the route in isolation without any congestion penalties, versus the result when all nets are routed and congestion modifies the taken path. When the value of the scenic ratio is close to

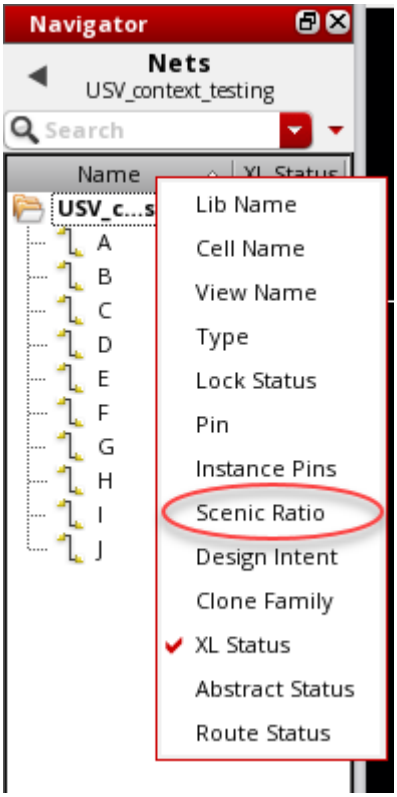
`1`, it means that the global route is accurate. The nets on which congestion analysis is not run displays the value of the scenic ratio as `0`.

Displaying the Scenic Ratio

To view the value of the scenic ratio for a net:

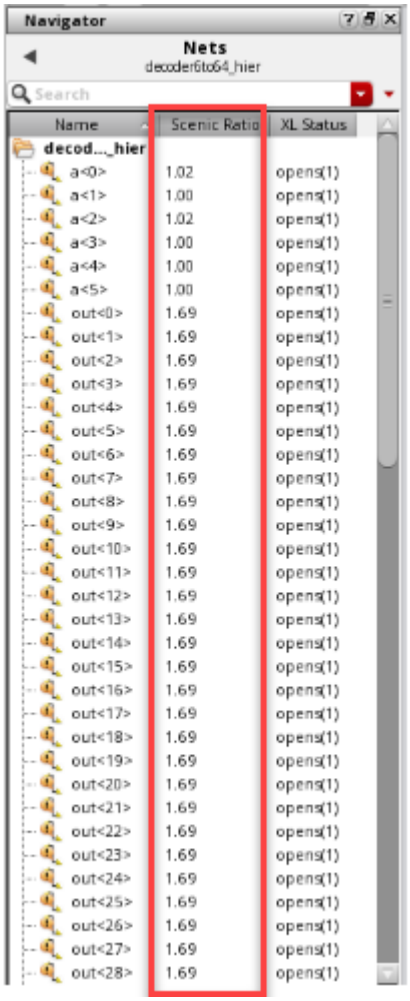
1. Click the *Congestion Analysis* icon to run congestion analysis.
2. Choose *Window – Assistants – Navigator Assistant*.
3. Right-click a column header in the Navigator Assistant.

   A drop-down menu is displayed.



4. Click *Scenic Ratio* from the drop-down list.

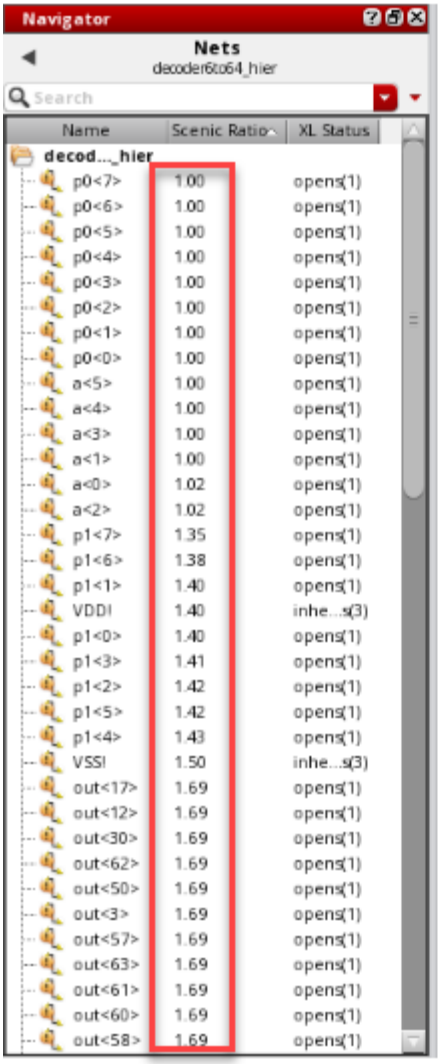
The *Scenic Ratio* column is displayed in the Navigator Assistant pane. This column displays the value of the scenic ratio for all nets in the design.



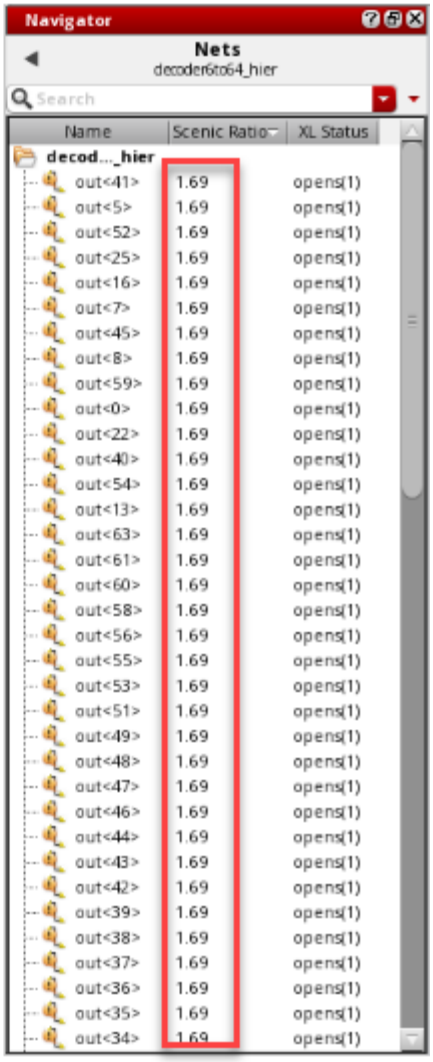
Sorting Scenic Ratio

You can sort the scenic ratio of nets in ascending or descending order. To sort the scenic ratio, click the *Scenic Ratio* column header. Clicking the column header the first time would display the scenic ratio of nets in the ascending order. This means that the net with the least value is

displayed at the top of the table. When you click the column header the next time, the scenic ratio of nets is displayed in the descending order.



Scenic ratio sorted in ascending order



Scenic ratio sorted in descending order

Modifying Scenic Ratio

You can modify the scenic ratio of a net by setting the global bias constraint on a net and then rerunning congestion analysis. When congestion analysis is rerun, the value of the scenic ratio may change for other nets as well.

Related Topics

Displaying the Congestion Analysis Assistant

Uses of the Congestion Analysis Assistant

Toggling Congestion Map Visibility

Global Bias Constraints

A

Design Planning and Analysis Environment Variables

This appendix provides information on the environment variable names, descriptions, and graphical user interface equivalents for the Design Planning and Analysis tool.

Only the environment variables documented in this chapter are supported for public use. All other The Design Planning and Analysis tool 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.

The environment variables for Design Planning and Analysis have been divided in the following partitions:

- **layoutDP**
  - adjustBoundaryCheckOutside
  - adjustBoundaryIncludeTop
  - allAreaBoundaries
  - areaBoundaryAspectRatio
  - areaBoundaryEnclosure
  - areaBoundaryHeight
  - areaBoundaryMinJogLength
  - areaBoundarySnap
  - areaBoundaryUtilization
  - areaBoundaryWidth
  - areaEstimationCSVFile
  - autoAdjustBoundary

- autoPlaceAllInstances
- autoPlaceLimit
- autoPlaceMinSep
- autoPlaceOnAreaBoundaryEdit
- autoPlaceOnPRBoundaryEdit
- autoPlaceUseInstBBox
- autoPositionLimit
- casDisplayVirtHierMismatch
- congestionAwareAccuracy
- createAreaBoundaryOnEdit
- dashedLinePlacementStatusNone
- defaultWorkspaceOnEIP
- fixPlacementStatusOnMove
- generateAreaBoundaries
- generateSoftBlocks
- generateSoftBlocksInTargetLibrary
- generateVirtualHierarchy
- keepVirtualGroupings
- makeCellAllLevels
- makeCellDeleteVirtualPins
- makeCellLibUseCph
- makeCellOptPins
- makeCellOverwriteLayout
- makeCellPinsBelow
- makeCellPinsChoice
- makeCellPushInBlock
- makeCellPushInternalRoutesOnly

- makeCellPushRoutesAsBlockages
- makeCellType
- makeCellVirtualClones
- makeVirtualAllInstsSameMaster
- makeVirtualGroupings
- makeVirtualPreserveVirtualPins
- pinLayerLimit
- pinLayerLimitNum
- pinOptOneConnectionPerSide
- softBlockArea
- softBlockAreaPercentTolerance
- updateBelowBoundary
- updateSoftBlocksFromSymbol
- useAreaBoundaryUtilization
- useBindKeys
- verboseApOnVhEdit
- vfpPACCells
- vfpPACCheckViolationsInRouter
- vfpPACCriticalNetName
- vfpPACCustomRailWidth
- vfpPACEnableDoubleCutVia
- vfpPACEnableRouter
- vfpPACEnableViaOnPG
- vfpPACLayrWidthSpac
- vfpPACLefFiles
- vfpPACLeftRightTopology
- vfpPACLibs

- vfpPACMaxRoutingLayerNum
- vfpPACOutputDir
- vfpPACPlacementTopology
- vfpPACRunWithInnovusLic
- vfpPACRunVerifyDesignWithColorOpts
- vfpPACTopBottomTopology
- vfpPACTopRoutingLyr
- vfpPACUtilizationPer
- vfpPACViaOnPGRail
- vfpPACViews
- vfpPACVoltageVal
- vhCloneColor
- vhCreatedColor
- vhDimming
- vhGeneratedColor
- vhSelectiveMode
- vhSymbolOverlay

■ **ia**

- cmapAnalyzeIncludeBlockage
- cmapAnalyzeMode
- cmapEnablePinOptimizeStep
- cmapHideAnalyzedDataInFilter
- cmapHiLiteAnalyzedBrightness
- cmapHiLiteDimAnalyzedBrightness
- cmapHiLiteDimFilteredBrightness
- cmapHiLiteFilteredBrightness
- cmapHiLiteGlobalPathWidthPercent

- cmapHiLiteSelectedBrightness
- cmapHiLiteSpecGlobalPathWidth
- cmapHistogramRanges
- cmapMultiThreads
- cmapNumTracksPerCell
- cmapShowUnusedGCellMode
- cmapUserDefinedNumTracks

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 is `t`.

GUI Equivalent

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

Examples

```
envGetVal("layoutDP" "adjustBoundaryCheckOutside")
envSetVal("layoutDP" "adjustBoundaryCheckOutside" 'boolean nil)
```

Related Topics

Adjust Boundary Form

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 is `nil`.

GUI Equivalent

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

Examples

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

Related Topics

Adjust Boundary Form

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

Examples

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

Related Topics

Generate Layout Form

Update Components And Nets Form

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*
Field:	*Aspect ratio (W/H)*

Examples

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

Related Topics

Design Planning Toolbar

Adjust Boundary Form

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*
Field:	*Enclose by*

Examples

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

Related Topics

Design Planning Toolbar

Adjust Boundary Form

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*
Field:	*Height*

Examples

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

Related Topics

Design Planning Toolbar

Adjust Boundary Form

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*
Field:	*Minimum jog length*

Examples

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

Related Topics

Design Planning Toolbar

Adjust Boundary Form

areaBoundarySnap

```
layoutDP areaBoundarySnap cyclic { "None" | "Lower Left" | "All Points" }
```

Description

Controls the snapping of area boundary when the *Generate All From Source, Adjust Boundary*, or *Stretch* command is run.

- `None` prevents the snapping of area boundary.
- `Lower Left` creates the area boundary of custom size and snaps the lower-left point of the bounding box to the grid.
- `All Points` snaps all the points of the area boundary bounding box to the grid. This means the bounding box can be larger than expected if the default least common multiple snapping is applied. By default, the area boundary snaps to all points.

The default is `All Points`.

GUI Equivalent

Command:	*Plan – Options*
Field:	*Area boundary snap*

Examples

```
envGetVal("layoutDP" "areaBoundarySnap")
envSetVal("layoutDP" "areaBoundarySnap" 'cyclic "None")
envSetVal("layoutDP" "areaBoundarySnap" 'cyclic "Lower Left")
```

Related Topics

Design Planning and Analysis Options Form

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 `25.0`.

GUI Equivalent

Command:	*Adjust Boundary – Rectangle*
Field:	*Estimate area – Utilization*

Examples

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

Related Topics

Design Planning Toolbar

Adjust Boundary Form

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*
Field:	*Width*

Examples

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

Related Topics

Design Planning Toolbar

Adjust Boundary Form

areaEstimationCSVFile

```
layoutDP areaEstimationCSVFile string "csvFileName"
```

Description

Specifies path to the CSV file that contains details such as library name, cell name, and soft block area to use for soft block and virtual hierarchy generation. Depending on the information it contains, the CSV file must list associated headings in the first row, such as `lib`,`cell`,`area`, followed by corresponding values in the rows below. A CSV file must contain at least the library and cellview headings (`lib`,`cell`) for the file to be read. Note that the headings must always be typed in lower case.

Here are the acceptable values for CSV file headings:

- Library: `lib`
- Cellview: `cell`
- Area: `area`
- Width: `width`
- Height: `height`
- Utilization: `util`
- Aspect ratio: `aspect`

The default is `""`.

Here are some examples depicting the format of a CSV file:

- Area.csv

  ```
  lib,cell,area
  ether,adc_comparator_actr,400
  ```

- WidthHeight.csv

  ```
  lib,cell,width,height,util,aspect
  ether,adc_comparator,10,25,50,1.5
  ```

GUI Equivalent

Command:	*Connectivity – Generate – All From Source (PR Boundary tab)*
Field:	*Area Estimate CSV File*
Command:	*Connectivity – Update – Components And Nets (PR Boundary tab)*
Field:	*Area Estimate CSV File*

Examples

```
envGetVal("layoutDP" "areaEstimationCSVFile")
envSetVal("layoutDP" "areaEstimationCSVFile" 'string "areaCSVFile")
```

Related Topics

Generate Layout Form

Update Components And Nets Form

Area Estimation Framework in Floorplanner

autoAdjustBoundary

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

Description

Automatically resizes the boundary of a rectangular virtual hierarchy when:

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

Note: Non-rectangular area boundaries are not automatically adjusted.

The default is `t`.

GUI Equivalent

Command:	*Plan – Options*
Field:	*Auto adjust area boundary*
Command:	*Connectivity – Update (Hierarchy tab)*
Field:	*Auto place generated instances inside existing virtual hierarchy*

Examples

```
envGetVal("layoutDP" "autoAdjustBoundary")
envSetVal("layoutDP" "autoAdjustBoundary" 'boolean nil)
```

Related Topics

Design Planning and Analysis Options Form

Update Components And Nets Form

autoPlaceAllInstances

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

Description

Automatically places all instances inside the area boundary after a *Stretch* or *Chop* operation.

The default is `nil`, which means the placer places only those instances that are currently outside the area boundary. The placement of instances already inside the area boundary is left unchanged.

GUI Equivalent

None

Examples

```
envGetVal("layoutDP" "autoPlaceAllInstances")
envSetVal("layoutDP" "autoPlaceAllInstances" 'boolean t)
```

Related Topics

Stretch Form

Chop Form

autoPlaceLimit

`layoutDP autoPlaceLimit int` *`numDevices`*

Description

Specifies the maximum number of devices that can be automatically placed when placing only those instances that are outside a virtual hierarchy area boundary. When the placement limit is reached, the automatic placer stops until the placement limit is increased.

The default is `500`.

GUI Equivalent

None

Examples

```
envGetVal("layoutDP" "autoPlaceLimit")
envSetVal("layoutDP" "autoPlaceLimit" 'int 1000)
```

Related Topics

Automatic Placement

autoPlaceMinSep

`layoutDP autoPlaceMinSep float` *`minSeparation`*

Description

Places the instances inside a virtual hierarchy area boundary at a minimum separation based on a user-defined value.

The default is `0.1`.

GUI Equivalent

None

Examples

```
envGetVal("layoutDP" "autoPlaceMinSep")
envSetVal("layoutDP" "autoPlaceMinSep" 'float 0.5)
```

Related Topics

Automatic Placement

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*

Examples

```
envGetVal("layoutDP" "autoPlaceOnAreaBoundaryEdit")
envSetVal("layoutDP" "autoPlaceOnAreaBoundaryEdit" 'boolean nil)
```

Related Topics

Adjust Boundary Form

Stretch Form

Chop Form

Reshape Form

Design Planning and Analysis Options Form

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 `nil`.

GUI Equivalent

Command:	*Plan – Options*
Field:	*Auto place on edit – PR boundary*

Examples

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

Related Topics

Stretch Form

Chop Form

Reshape Form

Design Planning and Analysis Options Form

autoPlaceUseInstBBox

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

Description

Uses the instance bounding box instead of the PR boundary to abut and place instances inside a virtual hierarchy during a *Stretch*, *Chop*, or *Reshape* operation.

The default is `t`.

GUI Equivalent

None

Examples

```
envGetVal("layoutDP" "autoPlaceUseInstBBox")
envSetVal("layoutDP" "autoPlaceUseInstBBox" 'boolean nil)
```

Related Topics

Stretch Form

Chop Form

Reshape Form

autoPositionLimit

`layoutDP autoPositionLimit int` *`numDevices`*

Description

Specifies the maximum number of devices that can be automatically placed during a full placement run for a virtual figGroup that has a rectangular area boundary, does not contain a `rowRegion`, and has no constraints applied to the design.

The default is `10000`.

GUI Equivalent

None

Examples

```
envGetVal("layoutDP" "autoPositionLimit")
envSetVal("layoutDP" "autoPositionLimit" 'int 9000)
```

Related Topics

Automatic Placement

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*

Examples

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

Related Topics

Check Against Source Form

cmapAnalyzeIncludeBlockage

```
ia cmapAnalyzeIncludeBlockage boolean { t | nil }
```

Description

(ICADVM20.1 EXL Only) Displays the global cells that are blocked and are essentially unavailable for use in any global paths or interactive routing. The default value is `nil`.

GUI Equivalent

Command	*Route – Design Setup – Congestion Analysis*
	Congestion Analyze – Options – Congestion Analysis
Field	*Show Unusable Global Cells*

Examples

```
envGetVal("ia" "cmapAnalyzeIncludeBlockage")
envSetVal("ia" "cmapAnalyzeIncludeBlockage" 'boolean t)
envSetVal("ia" "cmapAnalyzeIncludeBlockage" 'boolean nil)
```

Related Topics

Congestion Analysis Assistant

Analyzing Congestion

cmapAnalyzeMode

```
ia cmapAnalyzeMode cyclic { "allAverage" | "usedAverage" | "maximum" }
```

Description

(ICADVM20.1 EXL Only) Specifies an analysis mode to analyze the congestion across all layers and display the congestion result on the heat map.

- `allAverage` calculates the average congestion for all horizontal and vertical layers and colors the gcell based on the average computed percentage.
- `usedAverage` calculates the average congestion for all horizontal and vertical layers that have global routing and colors the gcell based on the average computed percentage.
- `maximum` calculates the maximum congested percentage for all horizontal and vertical layers and colors the gcell based on the maximum congested percentage.

The default value is `maximum`.

GUI Equivalent

Command	*Route – Design Setup– Congestion Analysis*
	Congestion Analyze – Options – Congestion Analysis
Field	*Analyze Mode*

Examples

```
envGetVal("ia" "cmapAnalyzeMode")
envSetVal("ia" "cmapAnalyzeMode" 'cyclic "allAverage")
envSetVal("ia" "cmapAnalyzeMode" 'cyclic "usedAverage")
envSetVal("ia" "cmapAnalyzeMode" 'cyclic "maximum")
```

Related Topics

Congestion Analysis Assistant

Analyzing Congestion

cmapEnablePinOptimizeStep

```
ia cmapEnablePinOptimizeStep boolean { t | nil }
```

Description

(ICADVM20.1 EXL Only) Enables running additional steps to optimize pins on soft-blocks and on opaque virtual hierarchies. The default is `nil` when running standalone Congestion Analysis.

The `cmapEnablePinOptimizeStep` environment variable is automatically set to `t` when the Congestion Analysis assistant is opened up from the Design Planner and all subsequent runs of congestion analysis from within the Design Planner workspace are set to `t`.

GUI Equivalent

Command	*Route – Design Setup– Congestion Analysis*
Field	*Enable Pin Optimization*

Examples

```
envGetVal("ia" "cmapEnablePinOptimizeStep")
envSetVal("ia" "cmapEnablePinOptimizeStep" 'boolean t)
envSetVal("ia" "cmapEnablePinOptimizeStep" 'boolean nil)
```

Related Topics

Congestion Analysis Assistant

cmapHideAnalyzedDataInFilter

```
ia cmapHideAnalyzedDataInFilter boolean { t | nil }
```

Description

(ICADVM20.1 EXL Only) Hides gcells, which are not part of the histogram filter, from the heat map. This improves visibility making it easier to visualize what has been filtered. The default value is `nil`.

GUI Equivalent

Command	*Route – Design Setup– Congestion Analysis*
Field	*Hide Analyzed Data in Filter*

Examples

```
envGetVal("ia" "cmapHideAnalyzedDataInFilter")
envSetVal("ia" "cmapHideAnalyzedDataInFilter" 'boolean t)
envSetVal("ia" "cmapHideAnalyzedDataInFilter" 'boolean nil)
```

Related Topics

Congestion Analysis Assistant

Analyzing Congestion

cmapHiLiteAnalyzedBrightness

```
ia cmapHiLiteAnalyzedBrightness int integer_number
```

Description

(ICADVM20.1 EXL Only) Specifies a value to control the brightness in the heat map when filtering is not applied. The valid value range is `1-60`. The default value is `60`.

GUI Equivalent

Command	*Route – Design Setup– Congestion Analysis*
Field	*Analyzed Brightness*

Examples

```
envGetVal("ia" "cmapHiLiteAnalyzedBrightness")
envSetVal("ia" "cmapHiLiteAnalyzedBrightness" 'int 30)
```

Related Topics

Congestion Analysis Assistant

Analyzing Congestion

cmapHiLiteDimAnalyzedBrightness

```
ia cmapHiLiteDimAnalyzedBrightness int integer_number
```

Description

(ICADVM20.1 EXL Only) Specifies a value to control the brightness of deselected gcells and deselected net global paths in the heat map. The valid value range is `1-30`. The default value is `7`.

GUI Equivalent

Command	*Route – Design Setup– Congestion Analysis*
Field	*Dim Analyzed Brightness*

Examples

```
envGetVal("ia" "cmapHiLiteDimAnalyzedBrightness")
envSetVal("ia" "cmapHiLiteDimAnalyzedBrightness" 'int 10)
```

Related Topics

Congestion Analysis Assistant

Analyzing Congestion

cmapHiLiteDimFilteredBrightness

```
ia cmapHiLiteDimFilteredBrightness int integer_number
```

Description

(ICADVM20.1 EXL Only) Specifies a value to control the brightness of deselected gcells and deselected net global paths in the heat map when filtering is applied. The valid value range is `1-80`. The default value is `25`.

GUI Equivalent

Command	*Route – Design Setup– Congestion Analysis*
Field	*Dim Filter Brightness*

Examples

```
envGetVal("ia" "cmapHiLiteDimFilteredBrightness")
envSetVal("ia" "cmapHiLiteDimFilteredBrightness" 'int 50)
```

Related Topics

Congestion Analysis Assistant

Analyzing Congestion

cmapHiLiteFilteredBrightness

```
ia cmapHiLiteFilteredBrightness int integer_number
```

Description

(ICADVM20.1 EXL Only) Specifies a value to control the brightness of the heat map when filtering is applied. The valid value range is `1-80`. The default value is `70`.

GUI Equivalent

Command	*Route – Design Setup– Congestion Analysis*
Field	*Filter Brightness*

Examples

```
envGetVal("ia" "cmapHiLiteFilteredBrightness")
envSetVal("ia" "cmapHiLiteFilteredBrightness" 'int 60)
```

Related Topics

Congestion Analysis Assistant

cmapHiLiteGlobalPathWidthPercent

```
ia cmapHiLiteGlobalPathWidthPercent int integer_number
```

Description

(ICADVM20.1 EXL Only) Specifies a value to change the width of the highlighted global path as a percentage of the width of a gcell. The specified value is used to increase or decrease the width of the displayed global path on the heat map. Only the following values are valid: `10`, `25`, `50`, `100`, `125`, `150`, `175`, `200`.

This variable is effective only if the cmapHiLiteSpecGlobalPathWidth variable is set to `t`. The default value is `100`.

GUI Equivalent

Command	*Route – Design Setup– Congestion Analysis*
Field	*Hilite Global Net Path Width Percentage*

Examples

```
envGetVal("ia" "cmapHiLiteGlobalPathWidthPercent")
envSetVal("ia" "cmapHiLiteGlobalPathWidthPercent" 'int 70)
```

Related Topics

Congestion Analysis Assistant

cmapHiLiteSelectedBrightness

```
ia cmapHiLiteSelectedBrightness int integer_number
```

Description

(ICADVM20.1 EXL Only) Specifies a value to control the brightness of selected gcells and net global paths in the heat map. The default value is `100`.

GUI Equivalent

Command	*Route – Design Setup– Congestion Analysis*
Field	*Selected Brightness*

Examples

```
envGetVal("ia" "cmapHiLiteSelectedBrightness")
envSetVal("ia" "cmapHiLiteSelectedBrightness" 'int 50)
```

Related Topics

Congestion Analysis Assistant

cmapHiLiteSpecGlobalPathWidth

```
ia cmapHiLiteSpecGlobalPathWidth boolean { t | nil }
```

Description

(ICADVM20.1 EXL Only) Enables you to change the width of the highlighted global path to be a percentage of the width of a gcell. If this variable is set to `t`, a value must be defined for the variable cmapHiLiteGlobalPathWidthPercent.

The default value is `nil`.

GUI Equivalent

Command	*Route – Design Setup– Congestion Analysis*
Field	*Hilite Global Net Path Width Percentage*

Examples

```
envGetVal("ia" "cmapHiLiteSpecGlobalPathWidth")
envSetVal("ia" "cmapHiLiteSpecGlobalPathWidth" 'boolean t)
envSetVal("ia" "cmapHiLiteSpecGlobalPathWidth" 'boolean nil)
```

Related Topics

Congestion Analysis Assistant

cmapHistogramRanges

```
ia cmapHistogramRanges string "congestion_range"
```

Description

(ICADVM20.1 EXL Only) Specifies the congestion range used to customize the histogram so that only the congested ranges are displayed. The default value is `""`.

GUI Equivalent

Command	*Route – Design Setup– Congestion Analysis*
	Congestion Analysis toolbar
Field	*Histogram Ranges*
	Congestion Histogram Customize

Examples

```
envGetVal("ia" "cmapHistogramRanges")
envSetVal("ia" "cmapHistogramRanges" 'string "range1")
```

Related Topics

Congestion Analysis Assistant

Customizing a Histogram

cmapMultiThreads

```
ia cmapMultiThreads int integer_number
```

Description

(ICADVM20.1 EXL Only) Specifies the number of threads used to run global routing and congestion analysis. The default is `1`, single threaded.

GUI Equivalent

Command	*Route – Design Setup– Congestion Analysis*
Field	*Number Threads*

Examples

```
envGetVal("ia" "cmapMultiThreads")
envSetVal("ia" "cmapMultiThreads" 'int 2)
```

Related Topics

Congestion Analysis Assistant

cmapNumTracksPerCell

```
ia cmapNumTracksPerCell int integer_number
```

Description

(ICADVM20.1 EXL Only) Specifies a value to override the number of tracks per gcell. This variable is effective only if the cmapUserDefinedNumTracks variable is set to `t`. The default value is `20`.

GUI Equivalent

Command	*Route – Design Setup– Congestion Analysis*
Field	*Number Tracks per Cell*

Examples

```
envGetVal("ia" "cmapNumTracksPerCell")
envSetVal("ia" "cmapNumTracksPerCell" 'int 10)
```

Related Topics

Congestion Analysis Assistant

cmapShowUnusedGCellMode

```
ia cmapShowUnusedGCellMode cyclic { "Blockage" | "Blockage and Preroutes" }
```

Description

(ICADVM20.1 EXL Only) Enables the display of the global cells that are blocked and are essentially not available for any global paths or interactive routing. This environment variable can take two values, `Blockage` and `Blockage and Preroutes`. Default is `Blockage`.

- `Blockage` shows the global cells that are 100 percent blocked and are not shown in either the histogram or the Global cell track utilization table.
- `Blockage and Preroutes` shows the global cells that are 100 percent blocked by both blockage and preroutes.

Note: The environment variable is applicable only if the `cmapAnalyzeIncludeBlockage` environment variable is set to `t`.

GUI Equivalent

Command	*Route – Design Setup– Congestion Analysis*
Field	*Show Unusable Global Cells*

Examples

```
envGetVal("ia" "cmapShowUnusedGCellMode")
envSetVal("ia" "cmapShowUnusedGCellMode" 'cyclic "Blockage")
envSetVal("ia" "cmapShowUnusedGCellMode" 'cyclic "Blockage and Preroutes")
```

Related Topics

Congestion Analysis Assistant

cmapUserDefinedNumTracks

```
ia cmapUserDefinedNumTracks boolean { t | nil }
```

Description

(ICADVM20.1 EXL Only) When set to `t`, enables you to change the number of override tracks per gcell. In this case, an override value must be defined for the variable cmapNumTracksPerCell. The default value is `nil`.

GUI Equivalent

Command	*Route – Design Setup– Congestion Analysis*
Field	*Number Tracks per Cell*

Examples

```
envGetVal("ia" "cmapUserDefinedNumTracks")
envSetVal("ia" "cmapUserDefinedNumTracks" 'boolean t)
envSetVal("ia" "cmapUserDefinedNumTracks" 'boolean nil)
```

Related Topics

Congestion Analysis Assistant

congestionAwareAccuracy

```
layoutDP congestionAwareAccuracy cyclic { "Low" | "Medium" | "High" }
```

Description

Controls the number of global routing passes performed to improve the balancing of the congestion for the routed nets. In addition, it defines the scope of abstraction down the hierarchy.

- `Low`: Abstracts instances and shapes 2-levels below in the hierarchy. Consider a design with eight levels of physical hierarchy. When this option is specified, the abstraction is not processed for any instance that is between levels three to eight. It simply abstracts the bounds and ignores the shapes below.
- `Medium`: Performs less but still substantial abstraction down to the middle of the physical hierarchy. Specifying this option lets you perform fewer passes of global route and therefore less balancing of congestion.
  This is not an effective option for designs that have fewer than five levels of physical hierarchy because it is similar to the `High` option for the shapes being processed.
- `High`: Performs four passes of global route for balancing congestion by analyzing all the shapes throughout the physical hierarchy.

The default is `Low`.

GUI Equivalent

Command:	*Plan – Options – Pins*
Field:	*Congestion aware accuracy*

Examples

```
envGetVal("layoutDP" "congestionAwareAccuracy")
envSetVal("layoutDP" "congestionAwareAccuracy" 'cyclic "Medium")
```

Related Topics

Design Planning and Analysis Options Form

createAreaBoundaryOnEdit

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

Description

Creates an area boundary for a virtual hierarchy during a *Stretch* or *Chop* operation, if the virtual hierarchy area boundary does not already exist.

The default is `t`.

GUI Equivalent

None

Examples

```
envGetVal("layoutDP" "createAreaBoundaryOnEdit")
envSetVal("layoutDP" "createAreaBoundaryOnEdit" 'boolean nil)
```

Related Topics

Virtual Hierarchy Editing Commands in Shortcut Menu

Generating a Virtual Hierarchy

dashedLinePlacementStatusNone

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

Description

Displays a dashed line to represent the bounding box of a virtual hierarchy that has its placement status set to *None*.

The default is `t`.

When set to `nil`, the virtual hierarchy bounding box is displayed using a solid line.

GUI Equivalent

None

Examples

```
envGetVal("layoutDP" "dashedLinePlacementStatusNone")
envSetVal("layoutDP" "dashedLinePlacementStatusNone" 'boolean nil)
```

Related Topics

Virtual Hierarchy Placement Status

defaultWorkspaceOnEIP

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

Description

Reverts to the default workspace in *Edit In Place* mode if the current workspace is set to *Design_Planning*.

The default is `nil`.

GUI Equivalent

None

Examples

```
envGetVal("layoutDP" "defaultWorkspaceOnEIP")
envSetVal("layoutDP" "defaultWorkspaceOnEIP" 'boolean t)
```

Related Topics

Design Planning Workspace

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`.

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

GUI Equivalent

Command:	*Plan – Options*
Field:	*Fix placement status on move*

Examples

```
envGetVal("layoutDP" "fixPlacementStatusOnMove")
envSetVal("layoutDP" "fixPlacementStatusOnMove" 'boolean nil)
```

Related Topics

Design Planning and Analysis Options Form

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*

Examples

```
envGetVal("layoutDP" "generateAreaBoundaries")
envSetVal("layoutDP" "generateAreaBoundaries" 'boolean nil)
```

Related Topics

Generate Layout Form

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*

Examples

```
envGetVal("layoutDP" "generateSoftBlocks")
envSetVal("layoutDP" "generateSoftBlocks" 'boolean nil)
```

Related Topics

Generate Layout Form

generateSoftBlocksInTargetLibrary

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

Description

Controls soft block layout creation for read-only libraries. Creates no soft blocks if CPH physical binding for the soft block is read-only.

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.

GUI Equivalent

None

Examples

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

Related Topics

Generate Layout Form

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*

Examples

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

Related Topics

PR Boundary

Generate Layout Form

keepVirtualGroupings

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

Description

Adds each instance group type to a created virtual group if the `makeVirtualGroupings` environment variable is set to `t`.

The default is `nil`, which means the *grouped by type within virtual groups* option on the Design Planning and Options form is disabled.

When `keepVirtualGroupings` is set to `nil` and `makeVirtualGroupings` is set to `t`, the *grouped by type* option on the Design Planning and Analysis Options form is enabled.

GUI Equivalent

Command:	*Plan – Options*
Field:	*Generate – Position – grouped by type*
Command:	*Plan – Options*
Field:	*Generate – Position – grouped by type within virtual groups*

Examples

```
envGetVal("layoutDP" "keepVirtualGroupings")
envSetVal("layoutDP" "keepVirtualGroupings" 'boolean t)
```

Related Topics

makeVirtualGroupings

Design Planning and Analysis Options Form

Virtual Hierarchy Generation

makeCellAllLevels

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

Description

Creates cellviews for all virtual hierarchy levels inside the selected virtual hierarchy using a bottom-up approach and creates a cellview for the selected virtual hierarchy.

The default is `nil`, which means hierarchical make cell creation is disabled.

GUI Equivalent

Command:	*Plan – Manage Hierarchy – Make Cell*
Field:	*Hierarchy – All levels*

Examples

```
envGetVal("layoutDP" "makeCellAllLevels")
envSetVal("layoutDP" "makeCellAllLevels" 'boolean t)
```

Related Topics

Make Cell Form

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*

Examples

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

Related Topics

Make Cell Form

Congestion Analysis Assistant

makeCellLibUseCph

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

Description

Uses the library/cell defined for make cell creation in CPH for the selected virtual hierarchies.

The default is `t`.

GUI Equivalent

None

Examples

```
envGetVal("layoutDP" "makeCellLibUseCph")
envSetVal("layoutDP" "makeCellLibUseCph" 'boolean nil)
```

Related Topics

Make Cell Form

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*
Command:	*Make Cell*
Field:	*Create pins – Congestion aware*

Examples

```
envGetVal("layoutDP" "makeCellOptPins")
envSetVal("layoutDP" "makeCellOptPins" 'boolean nil)
```

Related Topics

Make Cell Form

Design Planning Toolbar

Congestion Analysis Assistant

makeCellOverwriteLayout

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

Description

Controls whether the new made cellview overwrites an existing cellview.

The default is `nil`.

GUI Equivalent

Command:	*Plan – Manage Hierarchy – Make Cell*
Field:	*Overwrite layout cellview*

Examples

```
envGetVal("layoutDP" "makeCellOverwriteLayout")
envSetVal("layoutDP" "makeCellOverwriteLayout" 'boolean t)
```

Related Topics

Make Cell Form

Make Cell Command

makeCellPinsBelow

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

Description

Creates pins for the selected virtual hierarchy below the virtual hierarchy boundary.

The default is `t`.

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

GUI Equivalent

Command:	*Plan – Manage Hierarchy – Make Cell*
Field:	*Create pins – Below boundary*
	Create pins – On boundary
Command:	*Make Cell*
Field:	*Create pins – Below boundary*
	Create pins – On boundary

Examples

```
envGetVal("layoutDP" "makeCellPinsBelow")
envSetVal("layoutDP" "makeCellPinsBelow" 'boolean nil)
```

Related Topics

Make Cell Form

Design Planning Toolbar

makeCellPinsChoice

```
layoutDP makeCellPinsChoice cyclic { "Congestion aware" | "On boundary" | "Below
     boundary" | "Promote pins" }
```

Description

Specifies the options for creating interface pins when running the *Make Cell* command on a virtual hierarchy.

The default is `Congestion aware`.

- `Congestion aware` runs the congestion-aware global router to automatically create pins on the boundary of the virtual hierarchy.
- `On boundary` creates pins on the boundary of the virtual hierarchy, ensuring the shortest possible net length in the direction of routing.
- `Below boundary` creates pins just below the boundary of the virtual hierarchy.
- `Promote pins` extends pins from lower levels of a virtual hierarchy to a higher level.

GUI Equivalent

Command:	*Plan – Manage Hierarchy – Make Cell*
Field:	*Create pins – Congestion aware*
	Create pins – On boundary
	Create pins – Below boundary
	Create pins – Promote pins

Examples

```
envGetVal("layoutDP" "makeCellPinsChoice")
envSetVal("layoutDP" "makeCellPinsChoice" 'cyclic "On boundary")
```

Related Topics

Make Cell Form

Design Planning Toolbar

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*
Command:	*Make Cell*
Field:	*Push – Into block*

Examples

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

Related Topics

Make Cell Form

Design Planning Toolbar

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*
Command:	*Make Cell*
Field:	*Push – Internal routes only*

Examples

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

Related Topics

Make Cell Form

Design Planning Toolbar

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*
Command:	*Make Cell*
Field:	*Push – Routes as blockage*

Examples

```
envGetVal("layoutDP" "makeCellPushRoutesAsBlockages")
envSetVal("layoutDP" "makeCellPushRoutesAsBlockages" 'boolean nil)
```

Related Topics

Make Cell Form

Design Planning Toolbar

Push Into Blocks

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`.

- `softMacro` creates a soft block type cellview from the virtual hierarchy.
- `digital softMacro` creates a soft block with hierarchy for a block of type `digital`.
- `block` creates a hard block.
- `none` creates a custom cell type.

GUI Equivalent

Command:	*Plan – Manage Hierarchy –Make Cell*
Field:	*Type*
Command:	*Make Cell*
Field:	*Type*

Examples

```
envGetVal("layoutDP" "makeCellType")
envSetVal("layoutDP" "makeCellType" 'cyclic "digital softMacro")
```

Related Topics

Make Cell Form

Design Planning Toolbar

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*
Command:	*Make Cell*
Field:	*All clones*

Examples

```
envGetVal("layoutDP" "makeCellVirtualClones")
envSetVal("layoutDP" "makeCellVirtualClones" 'boolean nil)
```

Related Topics

Design Planning Toolbar

Make Cell Form

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*
Field:	*All instances of the same master*

Examples

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

Related Topics

Design Planning Toolbar

Make Virtual Hierarchy Form

makeVirtualGroupings

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

Description

Enables grouping of individual instances inside generated virtual hierarchies using the *grouped by type* or *grouped by type within virtual groups* option on the Design Planning and Analysis Options form.

- To enable grouping by type, `makeVirtualGroupings` should be set to `t` and keepVirtualGroupings should be set to `nil`.
- To enable grouping by type within virtual groups, both `makeVirtualGroupings` and `keepVirtualGroupings` should be set to `t`.

The default is `nil`.

GUI Equivalent

Command:	*Plan – Options*
Field:	*Generate – Position – grouped by type*
Command:	*Plan – Options*
Field:	*Generate – Position – grouped by type within virtual groups*

Examples

```
envGetVal("layoutDP" "makeVirtualGroupings")
envSetVal("layoutDP" "makeVirtualGroupings" 'boolean t)
```

Related Topics

keepVirtualGroupings

Design Planning and Analysis Options Form

Virtual Hierarchy Generation

makeVirtualPreserveVirtualPins

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

Description

Retains pinFigs in a cell as shapes, which can be later used to create pin shapes in the made cell, if *Make Cell* was run again on the selected virtual hierarchy.

The default is `t`.

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

GUI Equivalent

Command:	*Make Virtual Hierarchy*
Field:	*Virtual pins*

Examples

```
envGetVal("layoutDP" "makeVirtualPreserveVirtualPins")
envSetVal("layoutDP" "makeVirtualPreserveVirtualPins" 'boolean nil)
```

Related Topics

Make Cell Form

Make Virtual Hierarchy Form

Design Planning Toolbar

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*

Examples

```
envGetVal("layoutDP" "pinLayerLimit")
envSetVal("layoutDP" 'cyclic "All")
```

Related Topics

Design Planning and Analysis Options Form

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*

Examples

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

Related Topics

Design Planning and Analysis Options Form

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 is `t`.

GUI Equivalent

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

Examples

```
envGetVal("layoutDP" "pinOptOneConnectionPerSide")
envSetVal("layoutDP" "pinOptOneConnectionPerSide" 'boolean nil)
```

Related Topics

Congestion Analysis Assistant

Design Planning and Analysis Options Form

softBlockArea

`layoutDP softBlockArea float` *`floating_point_number`*

Description

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

The default is `400`.

GUI Equivalent

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

Examples

```
envGetVal("layoutDP" "softBlockArea")
envSetVal("layoutDP" "softBlockArea" 'float 295)
```

Related Topics

Generate Layout Form

softBlockAreaPercentTolerance

```
layoutDP softBlockAreaPercentTolerance float SBArea_percentage_tolerance
```

Description

Specifies the tolerance percentage honored by the *Check Against Source* command when checking the area of a generated soft block against its source.

The default is `2.0`.

GUI Equivalent

None

Examples

```
envGetVal("layoutDP" "softBlockAreaPercentTolerance")
envSetVal("layoutDP" "softBlockAreaPercentTolerance" 'float 3.5)
```

Related Topics

Virtual Hierarchy Generation

Connectivity-Driven Layout Editing Commands Supported by DPA

updateBelowBoundary

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

Description

Places any missing instances inside the virtual hierarchy after an *Update Components And Nets* run.

The default is `nil`.

GUI Equivalent

Command:	*Connectivity – Update Components And Nets (Hierarchy tab)*
Field:	*Auto place generated instances inside existing virtual hierarchy*

Examples

```
envGetVal("layoutDP" "updateBelowBoundary")
envSetVal("layoutDP" "updateBelowBoundary" 'boolean t)
```

Related Topics

Update Components And Nets Form

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 when symbol is modified*~~

Examples

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

Related Topics

Update Components And Nets Form

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*

Examples

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

Related Topics

areaBoundaryUtilization

areaBoundaryEnclosure

Generate Layout Form

useBindKeys

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

Description

Controls whether the user-defined bindkeys are used instead of the default bindkeys in the Design Planning and Analysis tool.

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

GUI Equivalent

Command:	*Plan – Options*
Field:	*Use bind keys*

Examples

```
envGetVal("layoutDP" "useBindKeys")
envSetVal("layoutDP" "useBindKeys" 'boolean nil)
```

Related Topics

Design Planning and Analysis Options Form

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)
```

Related Topics

autoPlaceOnAreaBoundaryEdit

Adjust Boundary Form

Move Form

Stretch Form

Chop Form

Reshape Form

vfpPACCells

`layoutXL.pinAccessChecker vfpPACCells string "`*`cellName`*`"`

Description

Specifies the required standard cells on which the pin accessibility checker is to be run.

The default is `""`.

GUI Equivalent

Command:	*Tools – Check Pin Accessibility*
Field:	*Design Information – Cell(s)*

Examples

```
envGetVal("layoutXL.pinAccessChecker" "vfpPACCells")
envSetVal("layoutXL.pinAccessChecker" "vfpPACCells" 'string "gnd_inherit")
```

Related Topics

Pin Accessibility Checker Form

Running the Pin Accessibility Checker

vfpPACCheckViolationsInRouter

```
layoutXL.pinAccessChecker vfpPACCheckViolationsInRouter boolean { t | nil }
```

Description

Opens the Cadence Innovus router graphical user interface and lets you view the violation markers in the Innovus violation browser. Enabling this option checks out a Cadence Innovus license.

The default is `nil`.

GUI Equivalent

Command:	*Tools – Check Pin Accessibility*
Field:	*Router Options – Check Violations in Router*

Examples

```
envGetVal("layoutXL.pinAccessChecker" "vfpPACCheckViolationsInRouter")
envSetVal("layoutXL.pinAccessChecker" "vfpPACCheckViolationsInRouter" 'boolean t)
```

Related Topics

Pin Accessibility Checker Form

Running the Pin Accessibility Checker

vfpPACCriticalNetName

```
layoutXL.pinAccessChecker vfpPACCriticalNetName string "list_of_netNames"
```

Description

Specifies the names of the critical nets that are to be routed first.

The list of nets must be space separated. For example, if you specify `netA netB netC`, then the tool prioritizes the `netA`, `netB`, and `netC` for routing in the layout view.

The default is `""`.

GUI Equivalent

Command:	*Tools – Check Pin Accessibility*
Field:	*Router Options – Critical Net*

Examples

```
envGetVal("layoutXL.pinAccessChecker" "vfpPACCriticalNetName")
envSetVal("layoutXL.pinAccessChecker" "vfpPACCriticalNetName" 'string "net_1 net_2")
```

Related Topics

Pin Accessibility Checker Form

Running the Pin Accessibility Checker

vfpPACCustomRailWidth

`layoutXL.pinAccessChecker vfpPACCustomRailWidth float` *`width`*

Description

Specifies the width of rails in the layout view.

The default is `0`. If no value is specified or if the specified value is less than the minimum width, then rail width is the same as the width of the underlying metal shape.

GUI Equivalent

Command:	*Tools – Check Pin Accessibility*
Field:	*Router Options – Rail Width*

Examples

```
envGetVal("layoutXL.pinAccessChecker" "vfpPACCustomRailWidth")
envSetVal("layoutXL.pinAccessChecker" "vfpPACCustomRailWidth" 'float 1.25)
```

Related Topics

Pin Accessibility Checker Form

Running the Pin Accessibility Checker

vfpPACEnableDoubleCutVia

```
layoutXL.pinAccessChecker vfpPACEnableDoubleCutVia boolean { t | nil }
```

Description

Inserts a double-cut via on the critical net.

The default is `nil`.

GUI Equivalent

Command:	*Tools – Check Pin Accessibility*
Field:	*Router Options – Use Double Cut Vias*

Examples

```
envGetVal("layoutXL.pinAccessChecker" "vfpPACEnableDoubleCutVia")
envSetVal("layoutXL.pinAccessChecker" "vfpPACEnableDoubleCutVia" 'boolean t)
```

Related Topics

Pin Accessibility Checker Form

Running the Pin Accessibility Checker

vfpPACEnableRouter

```
layoutXL.pinAccessChecker vfpPACEnableRouter boolean { t | nil }
```

Description

Routes the topology view for each standard cell instance by running the Innovus NanoRoute router.

The default is `nil`, which means that the router does not run.

GUI Equivalent

Command:	*Tools – Check Pin Accessibility*
Field:	*Router Options – Run Router*

Examples

```
envGetVal("layoutXL.pinAccessChecker" "vfpPACEnableRouter")
envSetVal("layoutXL.pinAccessChecker" "vfpPACEnableRouter" 'boolean t)
```

Related Topics

Pin Accessibility Checker Form

Running the Pin Accessibility Checker

vfpPACEnableViaOnPG

```
layoutXL.pinAccessChecker vfpPACEnableViaOnPG boolean { t | nil }
```

Description

Enables via generation for the power and ground rails.

The default is `nil`.

GUI Equivalent

Command:	*Tools – Check Pin Accessibility*
Field:	*Placement Topology – Add Via on PG Rail*

Examples

```
envGetVal("layoutXL.pinAccessChecker" "vfpPACEnableViaOnPG")
envSetVal("layoutXL.pinAccessChecker" "vfpPACEnableViaOnPG" 'boolean t)
```

Related Topics

vfpPACViaOnPGRail

Pin Accessibility Checker Form

Running the Pin Accessibility Checker

vfpPACLayrWidthSpac

```
layoutXL.pinAccessChecker vfpPACLayrWidthSpac string "t_layerName f_width
     f_spacing"
```

Description

Specifies the metal layer name, width, and spacing values for each routing layer.

The default is `""`. When a voltage value is specified using `vfpPACVoltageVal`, the Layer Width Spacing constraint table displays the metal-to -cut spacing value for each metal layer.

GUI Equivalent

Command:	*Tools – Check Pin Accessibility*
Field:	*Router Options – Layer Width/Spacing*

Examples

```
envGetVal("layoutXL.pinAccessChecker" "vfpPACLayrWidthSpac")
envSetVal("layoutXL.pinAccessChecker" "vfpPACLayrWidthSpac" 'string "M2 0.05
0.75")
```

Related Topics

vfpPACVoltageVal

Pin Accessibility Checker Form

Running the Pin Accessibility Checker

vfpPACLefFiles

```
layoutXL.pinAccessChecker vfpPACLefFiles string "fileName"
```

Description

Specifies the path of the LEF technology file to be read.

The default is `""`.

GUI Equivalent

Command:	*Tools – Check Pin Accessibility*
Field:	*Technology LEF – LEF File*

Examples

```
envGetVal("layoutXL.pinAccessChecker" "vfpPACLefFiles")
envSetVal("layoutXL.pinAccessChecker" "vfpPACLefFiles" 'string "test_case/
xyz.tech.lef")
```

Related Topics

Pin Accessibility Checker Form

Running the Pin Accessibility Checker

vfpPACLeftRightTopology

```
layoutXL.pinAccessChecker vfpPACLeftRightTopology boolean { t | nil }
```

Description

Specifies the placement topology for standard cells. Enabling this option places the same cell six times with `R0` and `MY` orientations.

The default is `nil`.

GUI Equivalent

Command:	*Tools – Check Pin Accessibility*
Field:	*Placement Topology – Select – Left_Right*

Examples

```
envGetVal("layoutXL.pinAccessChecker" "vfpPACLeftRightTopology")
envSetVal("layoutXL.pinAccessChecker" "vfpPACLeftRightTopology" 'boolean t)
```

Related Topics

Pin Accessibility Checker Form

Running the Pin Accessibility Checker

vfpPACLibs

`layoutXL.pinAccessChecker vfpPACLibs string "`*`libName`*`"`

Description

Specifies the cell library that contains the required standard cells.

The default is `""`.

GUI Equivalent

Command:	*Tools – Check Pin Accessibility*
Field:	*Design Information – Lib*

Examples

```
envGetVal("layoutXL.pinAccessChecker" "vfpPACLibs")
envSetVal("layoutXL.pinAccessChecker" "vfpPACLibs" 'string "basic")
```

Related Topics

Pin Accessibility Checker Form

Running the Pin Accessibility Checker

vfpPACMaxRoutingLayerNum

```
layoutXL.pinAccessChecker vfpPACMaxRoutingLayerNum int routingLayerNumber
```

Description

Specifies the highest number of routing layers that can be used when all metal layers up to the specified layer are used for routing.

The default is `4`, which means that you can use up to four routing layers in the PAC flow.

GUI Equivalent

Command:	*Tools – Check Pin Accessibility*
Field:	*Placement Topology – Routing Layers*

Examples

```
envGetVal("layoutXL.pinAccessChecker" "vfpPACMaxRoutingLayerNum")
envSetVal("layoutXL.pinAccessChecker" "vfpPACMaxRoutingLayerNum" 'int 10)
```

Related Topics

vfpPACTopRoutingLyr

Pin Accessibility Checker Form

Running the Pin Accessibility Checker

vfpPACOutputDir

```
layoutXL.pinAccessChecker vfpPACOutputDir string "temp_PinDirectory"
```

Description

Specifies the directory in which all temporary files and topology views are to be stored.

The directory can store the following information:

- The standard cell extended pin views created by the Pin Accessibility Checker.
- The routed view created by Innovus. Innovus uses the standard cell extended pins view as the input, runs the router, and generates a routed view.

The default is "`temp_PinChecker`".

GUI Equivalent

Command:	*Tools – Check Pin Accessibility*
Field:	*Design Information – Output Directory*

Examples

```
envGetVal("layoutXL.pinAccessChecker" "vfpPACOutputDir")
envSetVal("layoutXL.pinAccessChecker" "vfpPACOutputDir" 'string
"temp_PinDirectory")
```

Related Topics

Pin Accessibility Checker Form

Running the Pin Accessibility Checker

vfpPACPlacementTopology

```
layoutXL.pinAccessChecker vfpPACPlacementTopology cyclic { "Left_Right" |
     "Top_Bottom" }
```

Description

Specifies the placement topology for standard cells. The supported placement topologies are:

- `Left_Right`: Places six instances of a cell in `R0` and `MY` orientations.
- `Top_Bottom`: Places six instances of a cell in `R0`, `R180`, and `MX` orientations.

The default is "`Left_Right`".

GUI Equivalent

Command:	*Tools – Check Pin Accessibility*
Field:	*Placement Topology – Select*

Examples

```
envGetVal("layoutXL.pinAccessChecker" "vfpPACPlacementTopology")
envSetVal("layoutXL.pinAccessChecker" "vfpPACPlacementTopology" 'cyclic
"Top_Bottom")
```

Related Topics

Pin Accessibility Checker Form

Running the Pin Accessibility Checker

vfpPACRunWithInnovusLic

```
layoutXL.pinAccessChecker vfpPACRunWithInnovusLic boolean { t | nil }
```

Description

Runs the Cadence Innovus router using the Virtuoso Digital Implementation (VDI) flow in the foreground. Enabling this option checks out a Cadence Innovus license.

The default is `nil`, which means that the Innovus router is run in the background without checking out a license.

GUI Equivalent

None

Examples

```
envGetVal("layoutXL.pinAccessChecker" "vfpPACRunWithInnovusLic")
envSetVal("layoutXL.pinAccessChecker" "vfpPACRunWithInnovusLic" 'boolean t)
```

Related Topics

Pin Accessibility Checker Form

Running the Pin Accessibility Checker

vfpPACRunVerifyDesignWithColorOpts

```
layoutXL.pinAccessChecker vfpPACRunVerifyDesignWithColorOpts boolean { t | nil }
```

Description

Runs the *Verify Design* command and the Virtuoso Multi-Patterning Technology color engine on the remastered routed view to report the coloring violations in the Annotation Browser assistant.

The default is `nil`, which means that the coloring violations are not checked.

GUI Equivalent

None

Examples

```
envGetVal("layoutXL.pinAccessChecker" "vfpPACRunVerifyDesignWithColorOpts")
envSetVal("layoutXL.pinAccessChecker" "vfpPACRunVerifyDesignWithColorOpts"
'boolean t)
```

Related Topics

Fix Color Conflicts in Remastered Routed Views

Pin Accessibility Checker Form

Running the Pin Accessibility Checker

vfpPACTopBottomTopology

```
layoutXL.pinAccessChecker vfpPACTopBottomTopology boolean { t | nil }
```

Description

Specifies the placement topology for standard cells. Enabling this option places the same cell three times with `R0`, `R180`, and `MY` orientations.

The default is `nil`.

GUI Equivalent

Command:	*Tools – Check Pin Accessibility*
Field:	*Placement Topology – Select – Top_Bottom*

Examples

```
envGetVal("layoutXL.pinAccessChecker" "vfpPACTopBottomTopology")
envSetVal("layoutXL.pinAccessChecker" "vfpPACTopBottomTopology" 'boolean t)
```

Related Topics

Pin Accessibility Checker Form

Running the Pin Accessibility Checker

vfpPACSnapToGridLyr

```
layoutXL.pinAccessChecker vfpPACSnapToGridLyr string "layerName"
```

Description

Specifies the routing metal layer grid on which the cells are to be snapped. This ensures that the lower left point of the cell's PR boundary lies on the appropriate metal layer grid.

The default is `""`.

GUI Equivalent

Command:	*Tools – Check Pin Accessibility*
Field:	*Placement Topology – Snap to Grid*

Examples

```
envGetVal("layoutXL.pinAccessChecker" "vfpPACSnapToGridLyr")
envSetVal("layoutXL.pinAccessChecker" "vfpPACSnapToGridLyr" 'string "M3")
```

Related Topics

Pin Accessibility Checker Form

Running the Pin Accessibility Checker

vfpPACTopRoutingLyr

```
layoutXL.pinAccessChecker vfpPACTopRoutingLyr string "layerName"
```

Description

Specifies the name of the highest routing layer to be used. All lower metal layers up to the specified layer are used for routing. For example, if three routing layer are present between two devices, it allows you to select the metal layer from `M1`, `M2`, and `M3` and displays the name of the highest routing layer.

The default is `""`.

GUI Equivalent

Command:	*Tools – Check Pin Accessibility*
Field:	*Placement Topology – Routing Layers*

Examples

```
envGetVal("layoutXL.pinAccessChecker" "vfpPACTopRoutingLyr")
envSetVal("layoutXL.pinAccessChecker" "vfpPACTopRoutingLyr" 'string "M3")
```

Related Topics

vfpPACMaxRoutingLayerNum

Pin Accessibility Checker Form

Running the Pin Accessibility Checker

vfpPACUtilizationPer

`layoutXL.pinAccessChecker vfpPACUtilizationPer float` *`area`*

Description

Specifies the percentage of the cell area that can be used for routing.

The default is `60.0`.

GUI Equivalent

Command:	*Tools – Check Pin Accessibility*
Field:	*Placement Topology – Utilization%*

Examples

```
envGetVal("layoutXL.pinAccessChecker" "vfpPACUtilizationPer")
envSetVal("layoutXL.pinAccessChecker" "vfpPACUtilizationPer" 'float 45.0)
```

Related Topics

Pin Accessibility Checker Form

Running the Pin Accessibility Checker

vfpPACViaOnPGRail

```
layoutXL.pinAccessChecker vfpPACViaOnPGRail string "viaName"
```

Description

Specifies the name of the via to be generated on the power and ground rail when `vfpPACEnableViaOnPG` is set to `t`.

The default is `""`.

GUI Equivalent

Command:	*Tools – Check Pin Accessibility*
Field:	*Placement Topology – Add Via on PG Rail*

Examples

```
envGetVal("layoutXL.pinAccessChecker" "vfpPACViaOnPGRail")
envSetVal("layoutXL.pinAccessChecker" "vfpPACViaOnPGRail" 'string "CDS_V1005")
```

Related Topics

vfpPACEnableViaOnPG

Pin Accessibility Checker Form

Running the Pin Accessibility Checker

vfpPACViews

`layoutXL.pinAccessChecker vfpPACViews string "`*`viewName`*`"`

Description

Specifies the layout and abstract views to be used by the pin accessibility checker.

The default is "`layout abstract`".

GUI Equivalent

Command:	*Tools – Check Pin Accessibility*
Field:	*Placement Topology – View(s)*

Examples

```
envGetVal("layoutXL.pinAccessChecker" "vfpPACViews")
envSetVal("layoutXL.pinAccessChecker" "vfpPACViews" 'string "layout_placed
abstract")
```

Related Topics

Pin Accessibility Checker Form

Running the Pin Accessibility Checker

vfpPACVoltageVal

`layoutXL.pinAccessChecker vfpPACVoltageVal string "`*`voltageValue`*`"`

Description

Specifies a voltage value to switch on high voltage rule-based routing. This option applies only to input designs with voltage spacing rules defined in the technology file.

The default is "`Default`".

GUI Equivalent

Command:	*Tools – Check Pin Accessibility*
Field:	*Placement Topology – Voltage*

Examples

```
envGetVal("layoutXL.pinAccessChecker" "vfpPACVoltageVal")
envSetVal("layoutXL.pinAccessChecker" "vfpPACVoltageVal" 'string "3.0")
```

Related Topics

Pin Accessibility Checker Form

Running the Pin Accessibility Checker

vhCloneColor

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

Description

Specifies the color of the virtual hierarchy clone bounding box.

The default is `"hilite5"`.

GUI Equivalent

Command:	*Plan – Options*
Field:	*Color – Clone*

Examples

```
envGetVal("layoutDP" "vhCloneColor")
envSetVal("layoutDP" "vhCloneColor" 'string "hilite7")
```

Related Topics

Design Planning and Analysis Options Form

vhCreatedColor

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

Description

Specifies the bounding box color for the created virtual hierarchies.

The default is `"hilite6"`.

GUI Equivalent

Command:	*Plan – Options*
Field:	*Color – Created*

Examples

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

Related Topics

Design Planning and Analysis Options Form

vhDimming

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

Description

Enables automatic dimming for non-editable virtual hierarchy and virtual hierarchy clones.

The default is `t`.

GUI Equivalent

Command	*Options – Display*
Field	*Dim Virtual Hierarchy*

Examples

```
envGetVal("layoutDP" "vhDimming")
envSetVal("layoutDP" "vhDimming" 'boolean nil)
```

Related Topics

Display Options Form

vhGeneratedColor

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

Description

Specifies the bounding box color for the generated virtual hierarchies.

The default is `"hilite5"`.

GUI Equivalent

Command:	*Plan – Options*
Field:	*Color – Generated*

Examples

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

Related Topics

Design Planning and Analysis Options Form

vhSelectiveMode

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

Description

Enables selective virtual hierarchy generation mode, which means a completely flat layout is generated even with the *Virtual Hierarchy* option enabled on the Generate Layout form. You can allow specific virtual hierarchies to be generated by marking the cells for virtual hierarchy generation using the CPH cells table.

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

GUI Equivalent

Command	*Options*
Field	*Generate – Virtual hierarchy*
Command	*Launch – Configure Physical Hierarchy – Hierarchy Configuration Mode – Cells (table)*
Field	*Create/remove Virtual Hierarchy*

Examples

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

Related Topics

Creating Virtual Hierarchy for Selected Cells

Shortcut Menu for the Hierarchy Configuration Instances and Cells Tables

Design Planning and Analysis Options Form

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 is `nil`.

GUI Equivalent

Command	*Plan – Options*
Field	*Symbol overlay*

Examples

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

Related Topics

Design Planning and Analysis Options Form

B

Design Planning and Analysis Forms

This section lists the forms that can either be invoked only through the Design Planning and Analysis toolbar or forms that can also be invoked outside but have options that are related to the Design Planning and Analysis tool flow.

Adjust Boundary Form

Bias Layers Form

Congestion Histogram Customize Form

Create Virtual Group Form

Design Planning and Analysis Options Form

Generate Layout Form

Generate Selected Components Form

Global Bias Setup Form

Layer Display Form

Layer Overrides Form

Make Cell Form

Make Virtual Hierarchy Form

Pin Accessibility Checker Form

Remaster Form

Update Components And Nets Form

Adjust Boundary Form

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.

Field	Description
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.
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.
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.
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

Field	Description
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. 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.
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 **Note:** 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.

Field	Description
Enclose hierarchically	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 to enclose the selected boundary that is adjusted. If the *Utilization* option is selected, also adjusts the boundaries of the virtual hierarchies and soft blocks inside the selected area boundary using the specified utilization percentage, and automatically places the contents. ■ *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 to enclose the selected boundary that is adjusted. If the *Utilization* option is selected, also adjusts the boundaries of the virtual hierarchies and soft blocks inside the selected area boundary using the specified utilization percentage, and automatically places the contents. 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.

Field	Description
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. For some advanced process node technologies, the Virtuoso placer excludes certain material types such as trim, nwell, nImplant, and recognition layers, when creating the bounding box. To avoid having the materials from ignored during bounding box creation, you can set the `calcBBoxIgnoreMaterialTypes` environment variable to "".
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 unavailable, 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.

Field	Description
Enclose 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* adjusts the area boundary of all parent virtual hierarchies and soft blocks to enclose the selected boundary that is adjusted. If the *Utilization* option is selected, also adjusts the boundaries of the virtual hierarchies and soft blocks inside the selected area boundary using the specified utilization percentage, and automatically places the contents. ■ *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 to enclose the selected boundary that is adjusted. If the *Utilization* option is selected, also adjusts the boundaries of the virtual hierarchies and soft blocks inside the selected area boundary using the specified utilization percentage, and automatically places the contents. The PR boundary area adjustment is performed after all the required area boundary adjustments at lower levels have been made. Environment variable: adjustBoundaryIncludeTop
Points	Specifies the coordinates at which the polygon-shaped boundary is drawn. The default is `nil`.
Draw	Enables the drawing pointer on the layout canvas that can be dragged from one point to the other, drawing a side of the polygon-shaped boundary at a time.

Related Topics

Automatic Adjustment of a Virtual Hierarchy Area Boundary

Virtual Hierarchy Boundary Adjustment

lxHiAdjustBoundary

Moving Instances Outside a Virtual Hierarchy

Adding Instances to a Virtual Group

Area Estimation Framework in Floorplanner

Bias Layers Form

The following table describes the fields available on the Bias Layers form.

Field	Description
Specify each Bias Area	Lists the name of the global bias area or path.
Layers	Lets you select routing layers for the selected global bias are or path.
All Layers	Specifies that the selected global bias area or path must be completed on all routing layers.
Specify	Lets you specify the routing layers to which the selected global bias area or path should be constrained.

Related Topics

Global Bias Setup Form

Layer Display Form

Congestion Histogram Customize Form

Layer Overrides Form

Congestion Histogram Customize Form

Use the Congestion Histogram Customize form to filter the histogram to display specific buckets of congestion.

Field	Description
Customize As	Selects one of the following methods to customize the histogram. ■ *Default*: Customizes the histogram and displays the congestion data in the original format. ■ *Interval*: Specifies the congestion percentage with which to start and the interval between congestion buckets. ■ *Start and End*: Specifies the congestion percentage with which to start and end on. The intervals between congestion buckets are derived automatically. ■ *Specified*: Specifies the exact congestion percentage buckets. The start percentage for each congestion bucket is manually entered.
Start At	Specifies the value from which you want to start congestion filtering. This field is enabled when the method to customize the histogram is selected as *Interval* or *Start and End*.
End At	Specifies the value on which you want to end congestion filtering. This field is enabled when the method to customize the histogram is selected as *Start and End*.
With Interval	Specifies the interval between the congestion buckets. This field is enabled when the method to customize the histogram is selected as *Interval*.
Refresh	Refreshes the values in the congestion buckets displayed in the form.

Related Topics

Bias Layers Form

Layer Display Form

Global Bias Setup Form

Create Virtual Group Form

Use the Create Virtual Group form to manually create a virtual group for the selected top-level instances or instances inside a virtual hierarchy.

Field	Description
Name	Specifies a name for the virtual group to be created.
Orientation	Specifies an orientation for the virtual group. Choose orientation from: ■ *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

Virtual Group Creation in Design Planning and Analysis

Design Planning and Analysis Options Form

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

Field	Description
Generate	Specifies whether to generate all or specific virtual hierarchies.
Virtual hierarchy	Controls 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. If no cells are marked for virtual hierarchy generation, a completely flat layout is generated. Environment variable: vhSelectiveMode
Position	Controls how instances are positioned during virtual hierarchy generation. ■ *default*: Positions instances inside each generated virtual hierarchy. ■ *grouped by type*: Positions instances of a specific type grouped inside each generated virtual hierarchy. Instances are grouped by component type, cell master, device size or as single devices, mfactored instances, or vectored instances. See Virtual Hierarchy Generation. ■ *grouped by type within virtual groups*: Positions instances of a specific type grouped inside each generated virtual hierarchy and adds each group of instances to a created virtual group. Instances are grouped by component type, cell master, device size or as single devices, mfactored instances, or vectored instances. Environment variables: makeVirtualGroupings, keepVirtualGroupings

Field	Description
Auto Adjustment	Specifies the options for automatically adjusting a virtual hierarchy.
Auto place on edit	Specifies the placement of instances inside a PR boundary or a virtual hierarchy area boundary: ■ *boundaries* automatically places the contents inside the top-level PR boundary or the virtual hierarchy area boundary when either of the area boundaries is modified. ■ *area boundaries* automatically places the contents within the area boundary of the virtual hierarchy if the area boundary is modified using the *Adjust Boundary*, *Stretch*, Chop, or *Reshape* command. If the area boundary is moved using the *Move* command, the contents of the virtual hierarchy are not automatically placed. ■ *PR boundary* automatically places the top-level design when it contains virtual hierarchies and when the PR boundary is modified by using the *Stretch*, *Chop*, or *Reshape* command. Stretching a fully-selected PR boundary is equivalent to a move and does not trigger automatic placement. ■ *off* disables automatic placement. **Note:** For virtual hierarchies to be automatically placed during an edit, they must have their placement status set to *Fixed*. Environment variables: autoPlaceOnAreaBoundaryEdit, autoPlaceOnPRBoundaryEdit

Field	Description
Auto place virtual hierarchy	Controls the extent of placement of instances inside a virtual hierarchy: ■ *outside instances* places only those instances inside the virtual hierarchy that are currently outside the area boundary. Environment variable: autoPlaceLimit ■ *all instances* places all instances inside the area boundary after the virtual hierarchy area boundary is stretched or chopped. Environment variable: autoPlaceAllInstances
Fix placement status on move	Sets the placement status of the selected 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

Field	Description
Auto adjust	Resizes the area boundary of a virtual hierarchy to enclose instances and figGroups that are currently outside the area boundary and to preserve the enclosure when instances and figGroups are moved to overlap the area boundary. ■ *area boundary* automatically resizes to accommodate the following: ❑ any instances and figGroups other than row region that are manually moved outside the boundary of the virtual hierarchy. ❑ any virtual hierarchies that have had their area boundary manually stretched beyond the area boundary of the containing virtual hierarchy. ❑ any soft blocks that have had their PR boundary manually stretched beyond the area boundary of the containing virtual hierarchy. ■ *preserve enclosure* automatically resizes the area boundary of a virtual hierarchy to maintain the specified enclosure value when an instance or a figGroup is moved towards the area boundary. Environment variable: autoAdjustBoundary
Area boundary snap	Controls the snapping of the area boundary when the *Generate All From Source, Adjust Boundary*, or *Stretch* command is run. ■ *None* prevents the snapping of area boundary. ■ *Lower Left* creates the area boundary of custom size and snaps the lower-left point of the bounding box to the grid. ■ *All Points* snaps all the points of the area boundary bounding box to the grid. This means the bounding box can be larger than expected if the default least common multiple snapping is applied. By default, the area boundary snaps to all points. Environment variable: areaBoundarySnap

Field	Description
~~*Snap unknown block type as*~~	~~Snaps the PR boundary and the pins of unknown block types using the following options:~~ ■ *Custom: Places the pins and PR boundary on the manufacturing grid.* ■ *Digital: Places pins on the routing grid and PR boundary on the placement grid.* ■ *As-Is (XY Snap Spacing):* Places pins and PR boundary on the grid defined by the X Snap Spacing and Y Snap Spacing options in the Grid Controls section of the Display Options form. For more information, see Setting Up Grid Controls.
Display	Specifies the display options for a virtual hierarchy.
Name of	Controls the display of 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 Planning and Analysis tool bindkeys. By default the option is ON, which means the Design Planning and Analysis tool bindkeys are used for the design that has a virtual hierarchy. Environment variable: useBindKeys
Dashed line for placement status None	Displays a dashed line for virtual hierarchy bounding box when the placement status is *None*.

Field	Description
Clone	Sets the bounding box color for virtual hierarchy clones. Environment variable: vhCloneColor
Generated	Sets the bounding box 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`

Field	Description
Pins	Specifies the routing, pin connection, and update options.
Congestion Aware Accuracy	Controls the number of global routing passes performed to improve the balancing of the congestion for the routed nets. In addition, it defines the scope of abstraction down the hierarchy. ■ *Low*: Abstracts instances and shapes 2-levels below in the hierarchy. ■ *Medium*: Performs relatively less abstraction compared to *High* but substantial enough covering down to the middle of the physical hierarchy. Performs fewer passes of global route, resulting in lesser balancing of congestion. ■ *High*: Performs four passes of global route for balancing congestion by analyzing all the shapes throughout the physical hierarchy. **Note:** This is not an effective option for designs that have fewer than five levels of physical hierarchy because it is similar to using the *High* option for the shapes being processed. Environment variable: congestionAwareAccuracy
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 of 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

Field	Description
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 need 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*: Enables the *Label Layer options* button, which opens the Label Layer Purpose form for controlling the label layer and purpose options. For more information, see Updating Label Layer Purpose Pairs. ■ layer only: Re-layers the labels to match the pins, leaving the layer purposes unchanged. ■ *as-is*: Leaves the layer labels unchanged.

Related Topics

Adjust Boundary Form

Generate Layout Form

Use the Generate Layout form to generate a virtual hierarchy for the schematic instances that have no layout counterparts generated and to create soft blocks for schematic symbols that have no schematic or virtual hierarchy. For the virtual hierarchy generation options to be available on the Generate Layout form, you must have the Virtuoso Layout Suite EXL license checked out.

The Generate Layout form lets you specify the virtual hierarchy and area boundary generation options on the tabs below. For information on other layout generation options provided on the form, see Generate Layout.

Tab	Description
Generate	Lets you control if virtual hierarchies and soft blocks should be generated.
PR Boundary	Lets you specify the options to create the area boundary for a virtual hierarchy, the PR boundary for a soft block, or the top-level PR boundary. The *Virtual Hierarchy Area Boundary* options on the PR Boundary tab are enabled only when the *Virtual Hierarchy* option on the Generate tab is selected.

Generate

The following table describes the fields on the *Generate* tab of the Generate Layout form that support virtual hierarchy generation.

Field	Description
Design Planning	Specifies 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

Field	Description
Virtual Hierarchy 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, ignoring the specified *Utilization (%)*. Environment variable: generateSoftBlocks

PR Boundary

The following table describes the fields on the *PR Boundary* tab of the Generate Layout form that support virtual hierarchy area boundary generation.

Field	Description
Area Estimation	This section lets you specify how the area of a virtual hierarchy or a soft block is estimated.
Area Estimate CSV File	Specifies the path to a CSV file containing soft block and virtual hierarchy sizes to use for estimating the area of the soft blocks and virtual hierarchies to be generated. Environment variable: areaEstimationCSVFile
Virtual Hierarchy Area Boundary	Specifies the options 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 When `useAreaBoundaryUtilization` is set to `nil`, the *Enclose by* option (areaBoundaryEnclosure) is used.

Field	Description
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	Specifies the options to generate the area of a soft block.
Area	Specifies an estimated area value for the soft blocks to be generated. Environment variable: softBlockArea 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.

Related Topics

lxHiGenerateVirtualHierarchy

Generating All Components from Source

Generate Selected Components Form

Use the Generate Selected Components form to generate a virtual hierarchy for the selected schematic instances and to create soft blocks for schematic symbols that have no schematic or virtual hierarchy. For the virtual hierarchy generation options to be available on the Generate Selected Components form, you must have the Virtuoso Layout Suite EXL license checked out.

The following table describes the fields on the Generate Selected Components form that support virtual hierarchy generation. For information on other layout generation options provided on the form, see Generate Selected Components.

Field	Description
Design Planning	Specifies whether a virtual hierarchy is generated.
Virtual Hierarchy	Lets you generate a virtual hierarchy for the selected schematic instances. If you open the Generate Selected Components form using the *Plan* menu or the *Design Planning* toolbar, the *Virtual Hierarchy* option is automatically set to ON. If you open the form using the *Connectivity – Generate Selected From Source* menu, the *Virtual Hierarchy* option is automatically set to OFF.

Related Topics

Selected Component Generation in DPA

lxHiGenerateSelectedVirtualHierarchy

Generating Components As In Schematic

Global Bias Setup Form

Use the Global Bias Setup form to create, modify, and delete global bias constraints that can be used to plan the routing of a net group.

Field	Description
Global Bias	Lets you select an existing global bias constraint group.
New	Creates a new global bias constraint group by assigning a name to it. The default name for the global bias constraint group is considered as `GB<number>`.
Rename	Lets you rename the global bias constraint group.
Delete	Lets you delete the global bias constraint group.
Nets	Displays the name of the nets selected in the Navigator assistant and assigned to the selected global bias constraint group.
+ From Navigator	Adds pre-selected nets from the Navigator assistant to the selected global bias constraint group.
- Delete	Deletes the selected nets from the selected global bias constraint group.
Bias Areas	Lists the bias area or paths that have been created for the selected global bias constraint group.
Area	Displays the name of the bias area or path.
Bias Areas	Specifies whether the bias area is positive or negative. If the check box is selected, it indicates a positive bias. However, if the check box is deselected, it indicates a negative bias.
Layers	Displays the list of layers that have been defined for the bias area or path.
+ Rectangles	Lets you define the coordinates of the region for the bias area.
+ Paths	Lets you define the bias path that you want the nets in the bias constraint group to follow.
- Selected Area	Removes the bias area or path.
Define Layers	Displays the Bias Layers form

Related Topics

Bias Layers Form

Layer Display Form

Congestion Histogram Customize Form

Layer Overrides Form

Layer Display Form

Use the Layer Display form to choose specific layers when calculating and displaying congestion in the heat map, histogram, and global cell track utilization table.

Field	Description
All Layers On	Selects all the layers displayed in the form for congestion data.
All Layers On	Deselects all the layers displayed in the form.
Horizontal Layers Only	Selects only the horizontal layers in the design.
Vertical Layers Only	Selects only the vertical layers in the design.

Related Topics

Bias Layers Form

Global Bias Setup Form

Congestion Histogram Customize Form

Layer Overrides Form

Layer Overrides Form

The following table describes the fields available on the Layer Overrides form.

Field	Description
Direction	Displays the information of valid routing layers with the routing direction.
Layers	Lists the valid routing layers.
Direction	Displays the preferred routing direction for the routing layer. The supported routing directions are: *Horizontal* , *Vertical* , and *None*  .
Range	Lets you specify the valid routing layer range.
Top	Specifies the top valid routing layer limits and overrides the `validLayers` setting in the selected constraint group.
Bottom	Specifies the bottom valid routing layer limits and overrides the `validLayers` setting in the selected constraint group.

Related Topics

Bias Layers Form

Layer Display Form

Congestion Histogram Customize Form

Global Bias Setup Form

Make Cell Form

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 Virtuoso Layout Suite EXL and no virtual hierarchy is preselected, 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 Virtuoso Layout Suite 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.

Field	Description
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`, `layout_variant_3`, and so on. The layout view name for the new cell should not be a name defined in CPH Global Bindings such as `layout`, which is a default physical binding in CPH. This helps avoid overriding the physical binding in CPH.

Field	Description
Type	Specifies the type of the cell to be created. ■ *softMacro*: Creates a soft block type cellview from the virtual hierarchy. ■ *none*: Creates a custom cell type. ■ *digital softMacro*: Creates a soft block with hierarchy for a block of type `digital`. ■ *block:* Creates hard macros that can be supported by macro placers, Virtuoso Layout Suite XL commands such as *Load Physical View*, and other applications that support only macros and blocks. The default is *softMacro.* Environment variable: makeCellType
Overwrite layout cellview	Replaces the existing layout with the layout cellview created using selected virtual hierarchies. Environment variable: makeCellOverwriteLayout
Hierarchy	Specifies if the made cellview replaces the virtual hierarchy clones and all the virtual hierarchies across the design 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

Field	Description
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. For virtual hierarchies at lower levels, which do not already have an area boundary, the *Make Cell* command automatically creates area boundaries using the default enclosure value. See areaBoundaryEnclosure. When the *Create Pins – On boundary* or *Create Pins – Congestion aware* option is selected, the *Make Cell* command overrides the current display depth value and sets it to `32` for the selected virtual hierarchy. This ensures that pins for all virtual hierarchies under the selected virtual hierarchy are created on the boundary.
Create Pins	Specifies the options for creating interface pins for the selected virtual hierarchy. Environment variable: makeCellPinsChoice
Congestion aware	Runs the congestion-aware global router to automatically create pins on the boundary of the virtual hierarchy. Congestion-aware pin creation is recommended for scenarios where an initial placement has already been attempted. Having access to the congestion analysis data before the actual make cell creation can help avoid ANY routing bottlenecks before the actual make cell creation, improving design performance. By default, pin creation for the new made cell in the Design Planning and Analysis tool is congestion aware. For more information on congestion-aware pin placement, Congestion Analysis Assistant. Environment variable: makeCellOptPins

Field	Description
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. The creation of pins on or below the virtual hierarchy boundary with the *All levels* option selected depends on the current *Display Depth Options* value. ■ 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`) When *Below boundary* pin creation option is selected, the associated layer-purpose pair drop-down and the *Width* and *Height* fields gets enabled and you can specify the appropriate value in each field to use for below boundary pin creation.
Promote pins	Extends pins from lower levels of a virtual hierarchy to a higher level. If no pins are found at the level below, the command goes down to the next level to promote pins to the top.
Width	Specifies the width of the pins to be created.
Height	Specifies the height of the pins to be created.

Field	Description
Label	Specifies if a label needs to be created for the new pin based on the pin creation options specified on the Generate Layout form. ■ *None*: Creates no label for the new pin. ■ *Label*: Creates a label for the new pin. ■ *Text Display*: Creates a label for the new pin in the form of a text display. Pin labels are supported for all pin creation modes.
Text style	Opens the Virtuoso Layout Suite 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 label or text display.
Delete virtual pins	Deletes any existing virtual pins in the design to allow new pins to be generated. Environment variable: makeCellDeleteVirtualPins
Push	Controls the *Push* options during make cell creation. The *Push* options are supported only for the selected virtual hierarchy and not for the virtual hierarchies inside the selected virtual hierarchy.
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 Into Blocks.
Internal routes only	Pushes the internal routes into the made cell, no top-level routes are pushed. Environment variable: makeCellPushInternalRoutesOnly

Related Topics

Layout Component Generation in Design Planning and Analysis

Congestion Analysis Assistant

Make Virtual Hierarchy Form

Use the Make Virtual Hierarchy form to integrate layout hierarchies that have been realized outside the design. Additionally, for a softblock that only had pins but now also has a schematic available, the *Make Virtual Hierarchy* command can now generate missing components in the virtual hierarchy, retaining the pin locations and area boundary size as that of the softblock. The command can be used to integrate multiple selected cells as virtual hierarchies.

Field	Description
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 pin shapes in the made cell, if the virtual hierarchy was later used to create a make cell again. ■ When selected, the *Virtual pins* option adds the `lxStickyNet` property to virtual pins to preserve their connectivity. ■ If the *Virtual pins* option is not selected, the pinFigs are deleted during the *Make Virtual Hierarchy* run. Environment variable: makeVirtualPreserveVirtualPins
Labels	Specifies whether the attached labels are created when the layout hierarchy is integrated. ■ If you choose to create labels, you must ensure that you have the *Assign shapes from attached labels* option on the Virtuoso Layout Suite XL Connectivity form switched off to avoid causing shorts. See Connectivity form. ■ If the *Labels* option is selected, the Virtuoso Layout Suite XL connectivity extractor option *Assign shapes from overlapping labels as defined by 'stampLabelLayers' rule* should not be set. ■ If the *Labels* option is not selected, the labels are deleted.

Related Topics

Make Virtual Hierarchy Command in Design Planning and Analysis

Virtual Hierarchy Generation

Pin Accessibility Checker Form

Use the Pin Accessibility Checker form to verify the routability of standard cells.

Field	Description
Technology LEF	Specifies the additional technology file settings to be used by the Pin Accessibility checker.
LEF File(s)	Specifies the additional technology file to be honored. Environment variable: vfpPACLefFiles
Browse	Opens the Navigator assistant to select the required technology file.By default, the standard cell technology definitions are honored. In certain situations, for example, when standard cells are instantiated in a design that has different technology definitions, you can use *LEF File(s)* to load the additional technology file to be honored.
Load	Loads the specified LEF file. Technology definitions from the selected file are added to the current layout library technology data. The new combined technology file is attached to the current library. As a result, a combined list including vias and via rules is generated.
Design Information	Specifies the LCV that contains the standard cells and the directory in which all temporary files are to be stored.
Lib(s)	Specifies the standard cell libraries that contain the required standard cells.
Cell(s)	Specifies the required standard cells on which the tool is to be run.
View(s)	Specifies the view to be created by the tool.

Field	Description
Output Directory	Specifies the directory in which all temporary files and views are to be stored. The directory contains the following information: ■ The selected standard cells. ■ The standard cell extended pin views created by the Pin Accessibility Checker. Multiple output views are generated for the specified topology, utilization, and metal layer. The view names have a suffix that represents the postfix. For example `*_topBottom` and `*_leftRight` represent the topology, `*_60` represents the utilization, and `*_M3` represents the routing top layer. The postfix helps identify the topology, utilization, and top-layer combination used to generate the extended pin views. These views are then passed on to Innovus for routing. ■ The routed view created by Innovus. Innovus uses the standard cell extended pins view as the input, runs the router, and generates a routed view. Environment variable: vfpPACOutputDir
Placement Topology	Specifies the routing plan for the design.
Select	Specifies the placement topology for the standard cells. The available values are: ■ *Left_Right:* Places the same cell six times with the R0 and MY orientations. ■ *Top_Bottom:* Places the same cell three times with the R0, R180, and MY orientations. Environment variables: vfpPACPlacementTopology, vfpPACLeftRightTopology, vfpPACTopBottomTopology
Utilization%	Specifies the area to be used for routing.
Snap to Grid	Specifies the routing metal layer grid on which the cells are to be snapped. This ensures that the lower left point of the cell's PR boundary lies on the appropriate metal layer grid. By default, the routing grid of the top metal layer is selected for snapping cells. Environment variable: vfpPACSnapToGridLyr

Field	Description
Add Via on PG Rail	Specifies whether vias are to be generated on the power or ground rails. Select a value from the list.
Rail Width	Specifies the width of rails in the layout view. For example, you can make the width greater than the one drawn in the layout. If no value is specified or if the specified value is less than the minimum width, then rail width is the same as the width of the underlying metal shape. Environment variable: vfpPACCustomRailWidth
Router Options	Specifies the routing preferences for the Innovus router.
Routing Layers	Specifies the highest routing layer to be used.
Layer Width/ Spacing	Specifies the width and spacing values for each routing layer. Click *Define* to open the Layer Width Spacing constraint table in a new window. After making changes, close the window to populate values in the *Layer Width/Spacing* field.
Voltage	Switches on high voltage rule-based routing for input designs with voltage spacing rules defined in the technology file. Select this option to ensure that the voltage spacing rules for the selected voltage are honored during routing in high-voltage cells.
Critical Net	Specifies the nets that are to be routed first.
Double Cut Vias	Inserts double-cut vias in critical nets.
Run Router	Routes the topology view for each standard cell instance by running Innovus in the background without checking out an Innovus license. Environment variable: vfpPACEnableRouter
Check Violations in Router	Opens the Innovus router graphical user interface and generates violation markers that can be viewed in the Innovus violation browser. This option requires a Cadence Innovus router license. When deselected, the routed views are opened using Virtuoso and the violation markers are generated in the Annotation Browser assistant. Environment variable: vfpPACCheckViolationsInRouter

Related Topics

Running the Pin Accessibility Checker

Remaster Form

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

Field	Description
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

Remaster Command in Design Planning and Analysis

Update Components And Nets Form

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. For the virtual hierarchy update options to be available on the Update Components And Nets form, you must have the Virtuoso Layout Suite EXL license checked out.

The Update Components And Nets form lets you specify the virtual hierarchy and area boundary update options on the tabs below. For information on other layout generation options provided on the form, see Update Components And Nets Form.

Tab	Description
PR Boundary	Lets you specify how the area boundary of a virtual hierarchy is estimated.
Hierarchy	Lets you specify if virtual hierarchies are updated based on the schematic and if soft blocks are generated for missing schematics.

PR Boundary

The following table describes the fields available on the *PR Boundary* tab of the Update Components And Nets form.

Field	Description
Area Estimation	This section lets you specify how the area of a virtual hierarchy or a soft block is estimated for update.
Area Estimate CSV File	Specifies the path to a CSV file containing soft block sizes to use for estimating the area of the soft blocks to be updated. Environment variable: areaEstimationCSVFile
Virtual Hierarchy Area Boundary	Specifies the options for generating 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.

Field	Description
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	Specifies the option for soft block area generation.
Area	Specifies the area of the soft blocks to be generated.

Hierarchy

The following table describes the fields available on the *Hierarchy* tab of the Update Components And Nets form.

Field	Description
Design Planning	Specifies if the virtual hierarchy is updated to match the schematic hierarchy and if soft blocks are automatically generated for missing layouts.
Virtual Hierarchy	Generates a virtual hierarchy using the area boundary options selected on the *PR Boundary* tab of the Update Components and Nets 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. Environment variable: generateVirtualHierarchy

Field	Description
Auto Generate Soft Blocks	Generates 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 of the Update Components And Nets form. Environment variable: generateSoftBlocks
Update soft blocks when symbol is modified	Updates the soft block in the layout to match the schematic symbol. Environment variable: updateSoftBlocksFromSymbol
Auto place generated instances inside existing virtual hierarchy	Automatically places instances missing from a virtual hierarchy inside the virtual hierarchy, if the placement status of the virtual hierarchy is set to *None*. The option retains the placement of the instances already inside the virtual hierarchy. After placement, if the virtual hierarchy contents extend beyond the area boundary and the area boundary is rectangular, the boundary automatically adjusts to enclose the contents. Rectilinear area boundaries that have their contents extending beyond the boundary need to be manually adjusted and placed. Environment variables: updateBelowBoundary, autoAdjustBoundary The autoAdjustBoundary environment variable is used to automatically adjust the rectangular area boundary if the updateBelowBoundary environment variable does not automatically place the virtual hierarchy contents within the area boundary.

Related Topics

Update Components And Nets

Connectivity-Driven Layout Editing Commands Supported by DPA

C

Congestion Analysis Assistant

Use the Congestion Analysis assistant to run global routing and congestion analysis and illustrate the various methods of viewing congestion in your design. The following table describes the main components of the *Congestion Analysis* user interface.

Congestion Analysis Toolbar	Lets you access the main functions with a single mouse click.
Customizable Histogram	Lets you illustrate and customize the congestion results in the design based on the selected filter option.
Global Cell Track Utilization Table	Provides the capacity and availability information of every global cell (gcell) in the design.
Net Selection Table	Displays the nets passing through the gcell selected in the *Global Cell Track Utilization* table.

Congestion Analysis Toolbar

The *Congestion Analysis* assistant toolbar lets you access the main functions with a single mouse click. The following table describes the Congestion Analysis toolbar buttons.

Icon	Command	Description
	Congestion Analyze	Runs different methods of congestion analysis on the design.
	*Global Route and Congestion Analysis*	Runs global routing to build the gcell grid, run four passes of global routing on all nets, and then runs congestion analysis to build the histogram, create the heat map, and populate the global cell track utilization table. This is the default mode

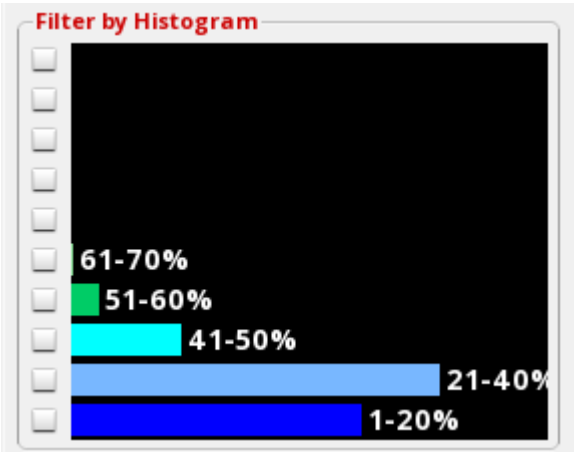
Icon	Command	Description
	*Global Route and ECO Congestion Analysis*	Runs global routing in ECO mode, which means that global routing is run on the nets for which the global bias constraint has been created. It then runs congestion analysis to update histogram, heat map and the global cell track utilization table. **Note:** The *Global Route and ECO Congestion Analysis* option is enabled only when a design has valid global bias constraints.
	*Global Route and Ignore Penalty Congestion Analysis*	A special what-if analysis mode for early floor planning. It runs global routing, where each net takes the shortest possible path regardless of other nets. This lets you evaluate if the floorplan is underutilized and if it can be compacted early in the design cycle or highlight the most common regions of over-congestion to indicate that more space might be required. After global routing, it runs congestion analysis to update the histogram, the heat map, and the global cell track utilization table.
	*Congestion Analysis*	Runs only congestion analysis on the design to build the histogram, create the heat map, and populate the global cell track utilization table based on the previously routed data. This mode is useful when you already have a fully detail routed design and you want to save the run time.
	Options	Opens the Congestion Analysis subform in the Virtuoso Space-based Router Options form.
	*Clear Congestion Analysis Data*	Clears the congestion analysis results and removes global routing.
	*Zoom Selected Global Cells*	Lets you zoom to selected global cells or nets.
	*Filter Global Cells By*	Provides options to filter and display congestion data.

Icon	Command	Description
	All Layers without Filters	Resets any filters that have been applied. **Note:** It does not reset the histogram if it has been customized. You must reset the histogram using the Congestion Histogram Customize form.
	Define Area	Lets you draw a box in the main window to filter an area of the design to display congestion analysis. The heat map, histogram, and global cell track utilization table are updated to only show the congestion data within the defined area.
	Show Selected Global Net Edges	Highlights the selected nets in the heat map. Also, the histogram and the global cell track utilization table are updated to show the congestion data of only the selected nets. **Note:** This filter mode works only if a net or nets have been selected.
	All Layers Average Congestion	Calculates the average congestion for all horizontal and vertical layers in a design.
	Used Layers Average Congestion	Calculates the average congestion for only the horizontal or vertical layers that have been routed in a design.
	Maximum Congestion	Calculates the maximum congestion for all horizontal and vertical layers in a design. This is the default congestion analysis mode.
	Show Unusable Global Cells	Enables the display of the global cells that are blocked and are essentially not available for any global paths or interactive routing. Environment variable: cmapShowUnusedGCellMode
	Congestion Map Visible	Toggles the display of the existing congestion map.
	Show Empty Global Cells Only	Displays all gcells that have zero congestion in the heat map and histogram.

Icon	Command	Description
	Layer Display	Lets you choose specific layers when calculating and displaying congestion in the heat map, histogram, and global cell track utilization table.
	*Congestion Histogram Customize*	Lets you filter the histogram to display specific buckets of congestion. Environment variable: cmapHistogramRanges
	*Global Bias Setup*	Lets you create, modify, and delete global bias constraints that can be used to plan the routing of a net group.
	*Finish with Detailed Routing*	Starts Virtuoso Space-based Router to complete detailed routing of the selected set of nets or the entire design.
	Route All Nets	Routes all nets using the standard VSR routing flow.
	Route Selected Nets	Routes only the selected nets using the standard V routing flow.

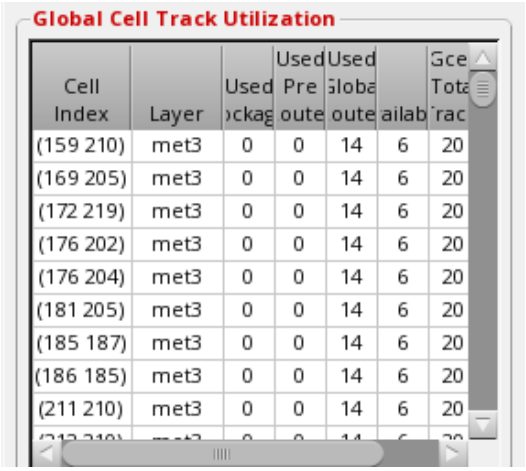
Customizable Histogram

Running congestion analysis displays the congestion data in the histogram. The histogram illustrates the congestion results in the design based on the selected filter option.



Global Cell Track Utilization Table

Running congestion analysis populates the *Global Cell Track Utilization* table with the capacity and availability information of every gcell in the design.

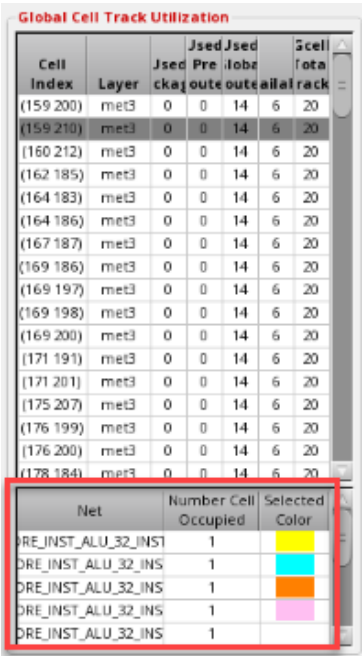


The following table describes the columns available in the *Global Cell Track Utilization* table of the Congestion Analysis assistant.

Column	Description
Cell Index	An internal numbering system for all the gcells in a design.
Layer	A layer of the gcell for which the track utilization data is displayed.
Used Blockages	The number of tracks used by blockages for the specified layer of the gcell.
Used Pre Routes	The number of tracks used by pre-routed nets for the specified layer of the gcell.
Used Global Routes	The number of tracks used by globally routed nets for the specified layer of the gcell.
Available	The number of tracks that are not used for the specified layer of the gcell. The information in this column is based on a simple calculation. `Available = Gcell Total Tracks – Used Blockages – Used Pre Routes – Used Global Routes`
Gcell Total Tracks	The total number of tracks for the specified layer of the gcell.

Net Selection Table

After running congestion analysis, when you select a gcell in the *Global Cell Track Utilization* table, the nets passing through the selected gcell are displayed in a table below the *Global Cell Track Utilization* table. This table is the Net selection table.



The following table describes the columns available in the Net Selection table.

Column	Description
Net	Displays the nets passing through the selected gcell.
Number Cell Occupied	Displays the numbers of cells occupied.
Selected Color	Displays the color selected for the net.

Related Topics

Bias Layers Form

Layer Display Form

Congestion Histogram Customize Form

Layer Overrides Form

Global Bias Setup Form