cadence®

Virtuoso Fluid Guard Ring User Guide

Product Version ICADVM20.1
October 2020

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

Contents

Preface

The Virtuoso Layout Suite L (Layout L) provides an innovative infrastructure with capabilities to create and implement layout designs. One of the important offerings of Layout L includes a menu-driven programmable feature for installing, creating, and editing fluid guard ring (FGR) devices, which are a type of fluid Pcells.

This user guide covers information about how to create and manage FGR devices. For information about how to use the Layout L tool, refer to the *Virtuoso Layout Suite L User Guide*.

You can also refer to the *Virtuoso Fluid Guard Ring Frequently Asked Questions* manual.

This user guide is aimed at developers and designers of integrated circuits who want to harness the usability and productivity benefits of FGR devices in Layout L. It assumes that you are familiar with:

- Virtuoso design environment and application infrastructure mechanisms designed to support consistent operations between all Cadence® tools
- Applications used to design and develop integrated circuits in the Virtuoso design environment, notably Virtuoso Layout Suite and Virtuoso Schematic Editor
- Design and use of parameterized cells
- OpenAccess version 2.2 technology file
- Component description format (CDF)

Virtuoso automatically loads a set of implementation files (`vfo*.ils`) at the time of initialization. These files provide the Virtuoso Fluid Object (VFO) infrastructure that enables you to install, create, and edit ***VLS-based FGRs*** as described in this user guide.

In addition, Virtuoso allows you to work with ***customized FGRs*** developed using the VFO infrastructure that adheres to the added or modified user-defined capabilities or features. In other words, the customized FGRs are based on capabilities that are not shipped as part of Virtuoso.

As it is difficult to ascertain the scope of customizations done by a PDK developer, which can differ from PDK to PDK, this user guide covers ***only*** the information about ***VLS-based FGRs***. The information about ***customized FGRs*** pertains only to ‘how to customize an

FGR'. If you are using the customized FGRs and need assistance in configuring or troubleshooting, contact your PDK provider for information.

This preface contains the following topics:

Scope

Unless otherwise noted, the functionality described in this guide can be used in both mature node (for example, IC6.1.8) and advanced node and methodologies (for example, ICADVM20.1) releases.

Label	Meaning
(ICADVM20.1 Only)	Features supported only in ICADVM20.1 advanced nodes and advanced methodologies releases.
(IC6.1.8 Only)	Features supported only in mature node releases.

Licensing Requirements

For using the FGR capabilities, you need to have a licensed Layout L installation.

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

Organization of this User Guide

This user guide comprises the following chapters and appendixes:

Chapter/Appendix	**Description**
Chapter 1, “Introduction to Fluid Guard Rings”	Provides a brief introduction about fluid Pcells and fluid guard rings.
Chapter 2, “Installing Fluid Guard Rings”	Covers the information about using the Install Guard Ring form.
Chapter 3, “Creating Fluid Guard Rings”	Covers the information about using the Create Guard Ring form.
Chapter 4, “Editing Fluid Guard Rings”	Describes different editing options that can be used on an FGR.
Appendix A, “Fluid Guard Ring Form Descriptions”	Describes the fields displayed on various FGR-specific forms available in Layout L.
Appendix B, “Fluid Guard Ring Environment Variables”	Describes the environment variables associated with FGR.
Appendix C, “Hiding Fluid Guard Ring Devices”	Describes the procedure of hiding an FGR device to restrict users from re-installing it or creating its instances on the layout.
Appendix D, “Loading VFO Infrastructure in Third-Party Tools”	Covers the steps to avoid getting error and warning messages while loading the customized FGR classes and methods in third-party tools.
Appendix E, “Geometry Changes in Virtuoso Releases”	Covers the information about various geometry changes that can occur in FGR instances when migrating to newer releases of Virtuoso.

Related Documentation

What's New and KPNS

- *Virtuoso Fluid Guard Ring What's New*.
- *Virtuoso Fluid Guard Ring Known Problems and Solutions*

Installation, Environment, and Infrastructure

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

Technology Information

- *Virtuoso Technology Data User Guide*
- *Virtuoso Technology Data ASCII Files Reference*
- *Virtuoso Technology Data SKILL Reference*

Virtuoso Tools

- *Virtuoso Layout Suite SKILL Reference*
- *Virtuoso Layout Suite XL User Guide*
- *Virtuoso Schematic Editor L User Guide*
- *irtuoso Space-based Router User Guide*
- *Virtuoso Design Rule Driven Editing User Guide*
- *Virtuoso Relative Object Design User Guide*
- *Virtuoso Parameterized Cell Reference*
- *Design Data Translator's Reference*

SKILL Documents

- The SKILL programming language is documented in the following manuals:
  - *Virtuoso Design Environment SKILL Reference*
  - *Cadence SKILL Language User Guide*
  - *Cadence SKILL Language Reference*
  - *Cadence SKILL Development Reference*
  - *Cadence SKILL IDE User Guide*
- SKILL access to other applications is provided in the following manuals:
  - *Virtuoso Technology Data SKILL Reference*
  - *Virtuoso Layout Suite SKILL Reference*
  - *Virtuoso Schematic Editor SKILL Reference*
  - *Cadence User Interface SKILL Reference*
  - *Cadence Interprocess Communication SKILL Reference*

Relative Object Design and Inherited Connections

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

Application Notes

The following FGR-specific application notes that are available on the Cadence Online Support website provide some useful additional information:

- Customizing Create Guard Ring Form

  This document explains how the Create Guard Ring form can be customized using specific triggers and SKILL APIs.

- Adding and Managing CDF Parameters for Fluid Guard Rings

  The document shows how to add and update the CDF parameters and attributes that affect the geometry of a fluid guard ring instance.

- Adding User-Defined Capabilities to Fluid Guard Rings

This document shows how to add user-defined capabilities or features to FGRs that are currently not supported using the supplied Install Guard Ring form.

- Creating Fluid SKILL Pcells

  This document explains the features of fluid SKILL Pcells.

Additional Learning Resources

Video Library

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

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

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

Virtuoso Videos Book

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

Rapid Adoption Kits

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

In addition, Cadence offers the following training courses on Virtuoso fluid guard ring functionality and related Virtuoso tools:

- Virtuoso Layout Design Basics
- Virtuoso Layout Pro: T1 Environment and Basic Commands (L)
- Virtuoso Layout Pro: T2 Create and Edit Commands (L)
- Virtuoso Layout Pro: T3 Basic Commands (XL)

- Virtuoso Connectivity-Driven Layout Transition
- Virtuoso Layout for Advanced Nodes (ICADVM20.1 Only)

Cadence also offers the following training courses on the SKILL programming language, which you can use to customize, extend, and automate your design environment:

- SKILL Language Programming Introduction
- SKILL Language Programming
- Advanced SKILL Language Programming

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

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

Help and Support Facilities

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

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

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

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

Customer Support

For assistance with Cadence products:

- Contact Cadence Customer Support

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

- Log on to Cadence Online Support

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

Feedback about Documentation

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

- Find erroneous information in a product manual
- Cannot find in a product manual the information you are looking for
- Face an issue while accessing documentation by using Cadence Help

You can also submit feedback by using the following methods:

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

Typographic and Syntax Conventions

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

text	Indicates names of manuals, menu commands, buttons, and fields.
`text`	Indicates text that you must type as presented. Typically used to denote command, function, routine, or argument names that must be typed literally.
`z_argument`	Indicates text that you must replace with an appropriate argument value. The prefix (in this example, *`z_`*) indicates the data type the argument can accept and must not be typed.
|	Separates a choice of options.
`{ }`	Encloses a list of choices, separated by vertical bars, from which you **must** choose one.
`[ ]`	Encloses an optional argument or a list of choices separated by vertical bars, from which you **may** choose one.
`[ ?argName` *`t_arg`* `]`	Denotes a *key argument*. The question mark and argument name must be typed as they appear in the syntax and must be followed by the required value for that argument.
`...`	Indicates that you can repeat the previous argument. Used with brackets to indicate that you can specify zero or more arguments. Used without brackets to indicate that you must specify at least one argument.
`,...`	Indicates that multiple arguments must be separated by commas.
`=>`	Indicates the values returned by a Cadence® SKILL® language function.
`/`	Separates the values that can be returned by a Cadence SKILL language function.

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

1

Introduction to Fluid Guard Rings

This chapter introduces you to the concept of fluid Pcells and the fluid guard rings (FGR).

Important

A good understanding of creating parameterized cells (Pcells) using Cadence® SKILL language is a prerequisite for the concepts discussed below. For detailed information on Pcells, see the *Virtuoso Parameterized Cell SKILL Reference*.

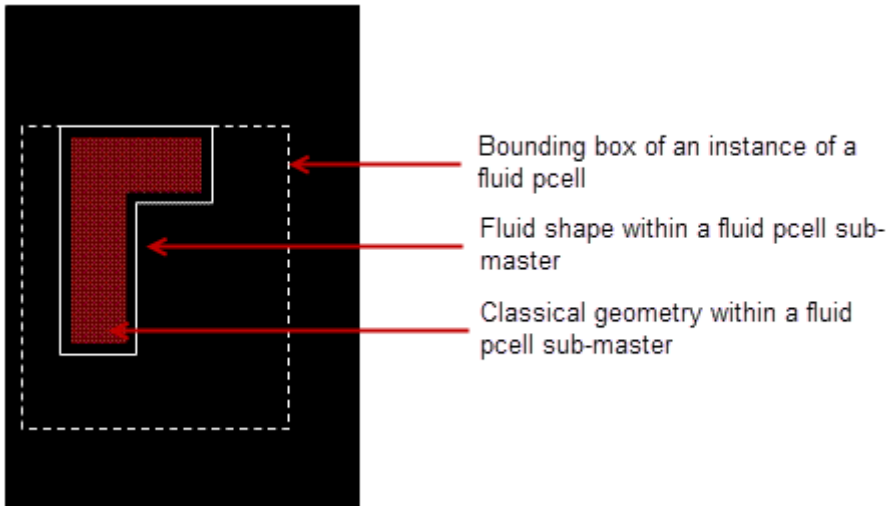
Introducing Fluid Pcells

Fluid Pcells provide an infrastructure to develop new Pcells that can be edited graphically (like shapes), and whose behavior in response to editing commands can be defined and customized using SKILL language.

Fluid Pcells offer to a Pcell developer complete control over the graphical editing functionality of a Pcell super master shape object.

You can draw the shapes in a fluid Pcell based on its fluid shape's points or any additional Pcell parameters. You can select a fluid shape from the top-level in Virtuoso Layout Suite L. In addition, you can edit designated fluid shapes within a sub-master like any other top-level shape. A set of SKILL updater functions, associated with the fluid Pcell super master, are called in response to the top-level editing commands, such as *Chop*, *Merge*, *Split*, *Reshape*, *Convert To Polygon*, *Abut*, and *Obstruct*.

The following figure shows an example of a fluid Pcell:



Features of Fluid Pcells

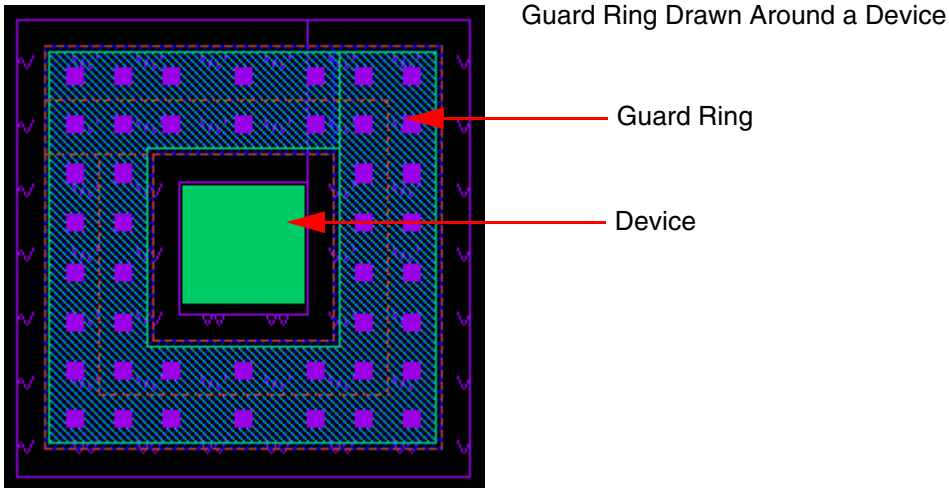
Fluid Pcells have the following features:

- You can *Chop*, *Merge*, and *Convert To Polygon* a fluid Pcell just like any other shape.
- You can stretch, split, or reshape a fluid shape within a fluid Pcell instance sub-master:
  - The edited shape is then passed to the correct SKILL updater function which in turn reflects the modified fluid shape edit back to the Pcell instance as a parameter value change.
  - The editing is done from top-level. There is no need to use the *Edit In Place* or *Descend* command.
- You can use the *Tunnel* command to connect through the fluid Pcell instances.
- You can partially select an edge, centerline, or end point in a fluid Pcell.
- The content of the fluid shape is specified in the hidden Pcell parameters.
- Fluid Pcells are useful for guard rings, space filling capacitors, and filling structures.
- **cdsGuardRing or FGR is the first application of fluid Pcells.**

For detailed information about fluid Pcells, refer to the *Creating Fluid SKILL Pcells* application note available on the Cadence Online Support website.

Introducing Fluid Guard Rings

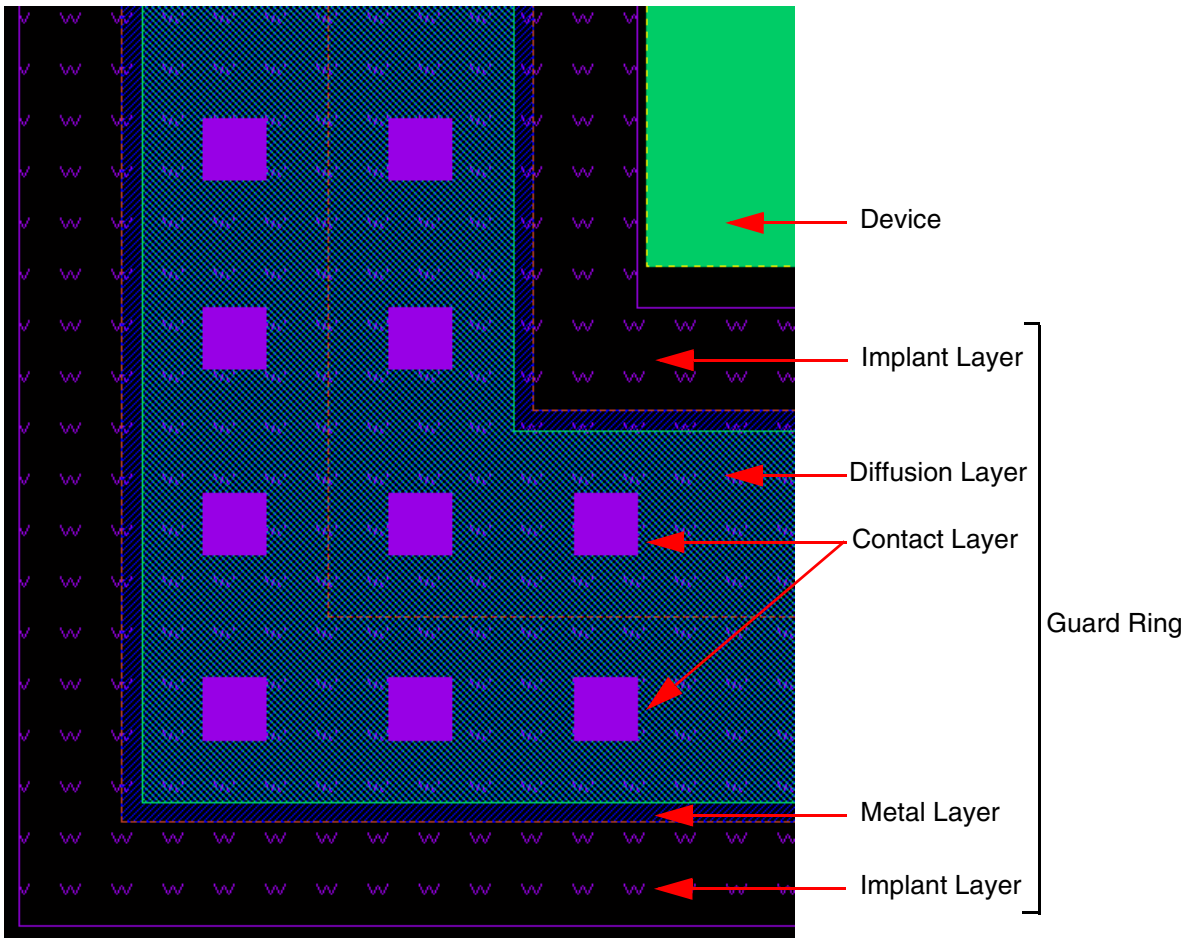
Fluid guard rings are a type of fluid Pcells that can be used to enclose one or more objects, such as devices or device chains.



There are two types of FGRs, p-diffusion or n-diffusion. These are the new generation fluid Pcell devices that comprise the following parts:

- **Diffusion Layer**: Encloses all other parts of the guard ring, except the implant or well layer when one is required.
- **Contact Layer**: A set of sub-rectangles, which are vias. The layer specified as the *Contact* layer is entirely enclosed by the *Metal* and the *Diffusion* layers; the vias are centered within the *Metal* layer.
- **Metal Layer**: Encloses the vias, which are centered within the *Metal* layer. The *Metal* layer is entirely enclosed by the *Diffusion* layer.
- **Implant/Well Layer**: An optional layer. You can define multiple implant and/or well layers for a guard ring.

The following figure shows a cross-section of an FGR device.



Refer to the following chapters for detailed procedural information:

- Installing Fluid Guard Rings
- Creating Fluid Guard Rings
- Editing Fluid Guard Rings
- Geometry Changes in Virtuoso Releases

Toolbar for Creating and Editing Fluid Guard Rings

From the *Window – Toolbars* menu, click the *Guardring* submenu. The following toolbar gets displayed:

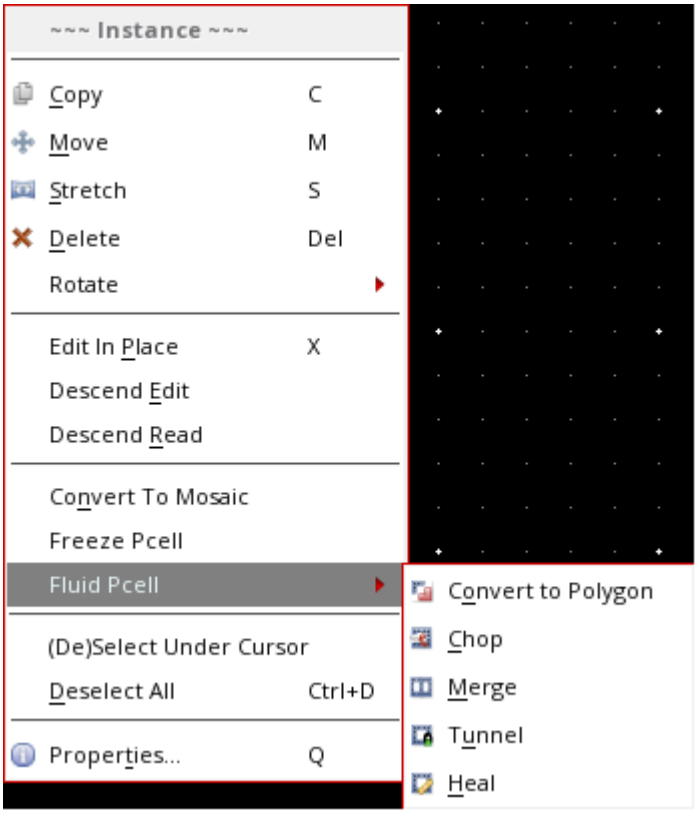
This toolbar has the following icons that enable you to create and edit FGR instances:

Toolbar Command	**Toolbar Icon**	**Alternative Access from Layout L Menu**	**Description**
Create Guardring by Wrapping		*Create – Fluid Guard Ring – Wrap tab*	Creates an FGR instance in *wrap* mode around a selected device. For more information, refer to the Wrap Mode section in Chapter 3, “Creating Fluid Guard Rings.”
Create Guardring from Path		*Create – Fluid Guard Ring – Path tab*	Creates an FGR instance in *path* mode. For more information, refer to the Path Mode section in Chapter 3, “Creating Fluid Guard Rings.”
Create Guardring from Rectangle		*Create – Fluid Guard Ring – Rect tab*	Creates an FGR instance in *rectangle* mode. For more information, refer to the Rectangle Mode section in Chapter 3, “Creating Fluid Guard Rings.”
Create Polygon Contact Fill		*Create – Fluid Guard Ring – Polygon tab*	Creates an FGR instance in *polygon* mode. For more information, refer to the Polygon Mode section in Chapter 3, “Creating Fluid Guard Rings.”
Convert to Polygon		*Edit – Fluid Pcell – Convert to Polygon*	Converts the selected FGR instance into a polygon. For more information, refer to the Converting a Fluid Guard Ring to a Polygon section in Chapter 4, “Editing Fluid Guard Rings.”

Chop		*Edit – Fluid Pcell – Chop*	Removes a part of an FGR instance or cut it into pieces. For more information, refer to the Chopping a Fluid Guard Ring section in Chapter 4, "Editing Fluid Guard Rings."
Merge		*Edit – Fluid Pcell – Merge*	Merges FGRs and create a new FGR device. For more information, refer to the Merging Fluid Guard Rings section in Chapter 4, "Editing Fluid Guard Rings."
Tunnel		*Edit – Fluid Pcell – Tunnel*	Creates a tunnel through an FGR. For more information, refer to the Creating a Tunnel Through a Fluid Guard Ring section in Chapter 4, "Editing Fluid Guard Rings."
Heal		*Edit – Fluid Pcell – Heal*	Removes one or more tunnels from an FGR. For more information, refer to the Healing a Fluid Guard Ring section in Chapter 4, "Editing Fluid Guard Rings."
Reshape		*Edit – Advanced – Reshape*	Alters the fluid shape points and re-generates the underlying guard ring based on the new points. For more information, refer to the Reshaping a Fluid Guard Ring section in Chapter 4, "Editing Fluid Guard Rings."
Split		*Edit – Advanced – Split*	Splits the fluid shape of a guard ring. For more information, refer to the Splitting a Fluid Guard Ring section in Chapter 4, "Editing Fluid Guard Rings."
Clean Overlapping Contacts		*Edit – Fluid Pcell – Clean Overlapping Contacts*	Cleans all the overlapping contacts from an FGR, or just from an area on the canvas that you draw based on the option you select. For more information, refer to the Cleaning Overlapped Contacts from Fluid Guard Rings section in Chapter 4, "Editing Fluid Guard Rings."

Fluid Guard Ring Context-Sensitive Menu

Right-click any FGR instance to display the context-sensitive *Instance – Fluid Pcell* menu that provides a quick access to the FGR editing commands.



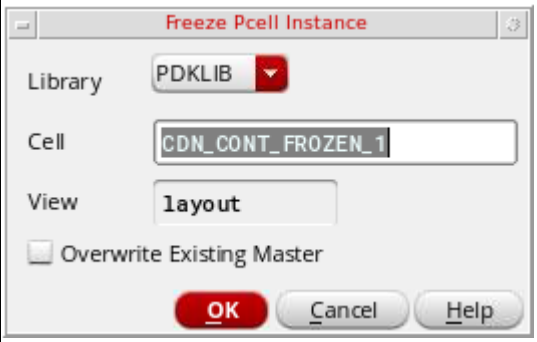
This *Instance – Fluid Pcell* context-sensitive menu provides the following options:

Editing Command	**Menu Icon**	**Alternative Access from Layout L Menu**	**Description**
Convert to Polygon		*Edit – Fluid Pcell – Convert to Polygon*	Converts the selected FGR instance into a polygon.
Chop		*Edit – Fluid Pcell – Chop*	Removes a part of an FGR instance or cut it into pieces.

Merge		*Edit – Fluid Pcell – Merge*	Merges FGRs and create a new FGR device.
Tunnel		*Edit – Fluid Pcell – Tunnel*	Creates a tunnel through an FGR.
Heal		*Edit – Fluid Pcell – Heal*	Removes one or more tunnels from an FGR.

A few more options in the *Instance* context-sensitive menu are described in the table below:

Editing Command	**Menu Icon**	**Alternative Access from Layout L Menu**	**Description**
Copy		*Edit – Copy*	Enables you to place a copy the selected FGR instance to the same cellview or to another cellview. For more information, refer to the *Copying Objects* section in the *Editing Objects* chapter of the *Virtuoso Layout Suite L User Guide*.
Move		*Edit – Move*	Lets you move the selected FGR instance to another location in the current or another cellview. For more information, refer to the *Moving Objects* section in the *Editing Objects* chapter of the *Virtuoso Layout Suite L User Guide*.
Stretch		*Edit – Stretch*	Stretches the FGR instance by using the center line or the vertex of the path. For more information, refer to the *Stretching a Fluid Guard Ring* section in *Chapter 4, "Editing Fluid Guard Rings."*
Delete		*Edit – Delete*	Deletes the selected FGR instance.

Freeze Pcell			Enables you to perform Pcell editing that is normally not supported by the Pcell parameters. This means you can avoid geometry changes in the FGR instances using this menu option. For more information, refer to the *Freezing and Unfreezing Pcell Instances* section in the *Working With Hierarchical Designs* chapter of the *Virtuoso Layout Suite L User Guide*. Clicking this menu option opens the *Freeze Pcell Instance* form.  On this form, specify a new cell name with which the FGR instance should be saved as a non-fluid Pcell instance of a new master created on the disk. By default, the new cell is named in the format, `CDN_<DEVICE_NAME>_FROZEN_#`, where # denotes the unique number assigned to the frozen copy of the selected FGR instance. For example, `CDN_CONT_FROZEN_1`.s
(De)Select Under Cursor			Lists the FGR instances under the cursor. You can deselect or select the check box next to each FGR instance name. As a result, it gets ignored or considered for the edit actions you perform.
Properties			Displays the *Edit Instance Properties* form that has separate tabs for changing the editable *Attribute*, *Connectivity*, *Parameter*, *Property*, and *ROD*. For more information, refer to the *Edit Instance Properties Form* section in Appendix A, “Fluid Guard Ring Form Descriptions.”

2

Installing Fluid Guard Rings

Important

If you are using customized fluid guard rings (FGRs), the usage information about the Install Guard Ring form covered in this chapter might not be relevant. This is because customized FGRs can be installed by loading a custom-made technology file, which allows you to directly proceed with the creation of instances of an FGR device.

While using VLS-based FGRs, before creating an instance of an FGR device in the layout, you need to install the FGR by defining and saving it as a device class in a technology library. Consequently, a new device class called `cdsGuardRing` gets created in the associated technology library.

Important

Make sure the technology library is writable. For a library to be considered as a technology library, it must have a writable `tech.db` file. Also, if there is a `data.dm` file, it must be writable too. If the technology library is read only, then all the fields on the Install Guard Ring form are disabled and you cannot install any new device.

To install an FGR, perform the following steps:

1. Start `virtuoso`.

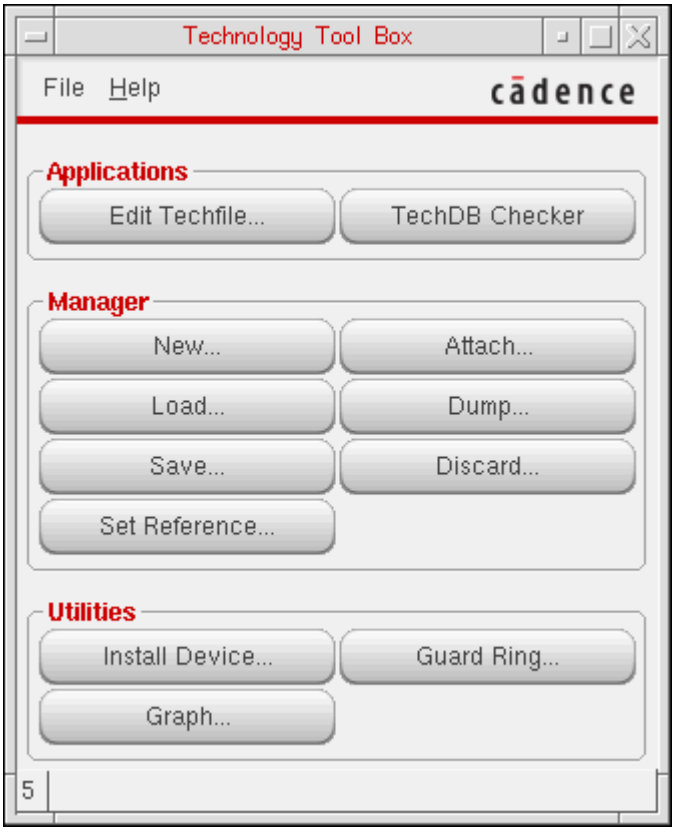
   During the initialization of Virtuoso, a set of implementation files (`vfo*.ils`) required for working with FGRs get loaded automatically in a specific sequence from the following release installation directory:

   `<install_dir>/tools/dfII/etc/vfo`

   Note: If you want to load the FGR implementation files from a previous release or from a different location, specify the path by using the fluidGuardRingInstallPath environment variable.

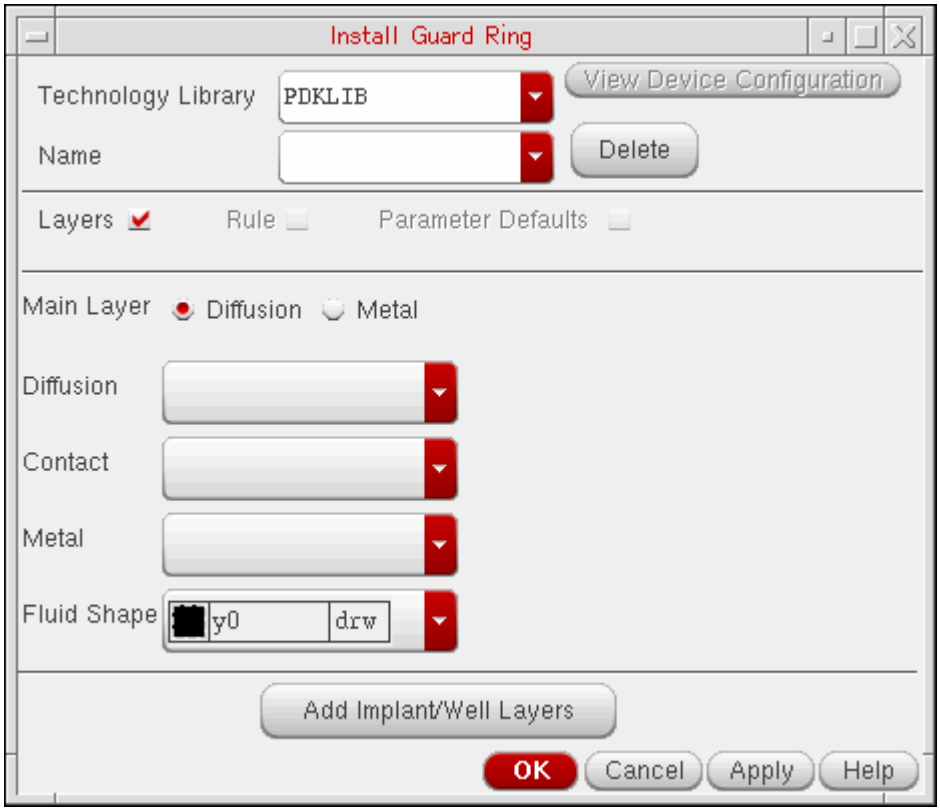
2. From the CIW, choose *Tools – Technology File Manager*.

The *Technology Tool Box* form appears.



3. In the *Utilities* section, click *Guard Ring*.

The Install Guard Ring Form appears.



4. Select the *Technology Library* in which you want to install the FGR device.
5. In the *Name* drop-down list box, you can specify one of the following:
   - A unique name for the new FGR device
   - The name of an existing FGR device (to update its parameters)

   If an existing FGR device is not listed in the *Name* drop-down list box, it means the device is hidden. Devices can be hidden to avoid chances of making any inadvertent changes to its definition. For procedural details, refer to the Hiding a Device from Install Guard Ring Form section.

   In case, you want to change the definition of a hidden FGR device, make it visible on the form by following the steps covered in the Making a Hidden Device Visible on Install and Create Forms section.

   If you manually enter the name of a hidden FGR device, you are not allowed to re-install it. Instead, you get a prompt message stating that it is already installed and is hidden. For information about handling such scenario, refer to the answer to question, *Can an*

FGR device hidden from the Install Guard Ring form be re-installed?, in the Virtuoso Fluid Guard Ring Frequently Asked Questions manual.

Note: To delete an existing FGR device, select the FGR from the *Name* drop-down list box and click *Delete*.

6. The *Layers* check box is selected by default when the Install Guard Ring form gets displayed.

   If you are creating a new FGR, the *Diffusion*, *Contact*, and *Metal* lists are blank. Until you set up these layers, the *Rule* and *Parameter Defaults* check boxes, and the *View Device Configuration* button remain grayed out.

   Note: It is mandatory to set up the *Diffusion*, *Contact*, and *Metal* layers and optional to set up the *Implant/Well* layers.

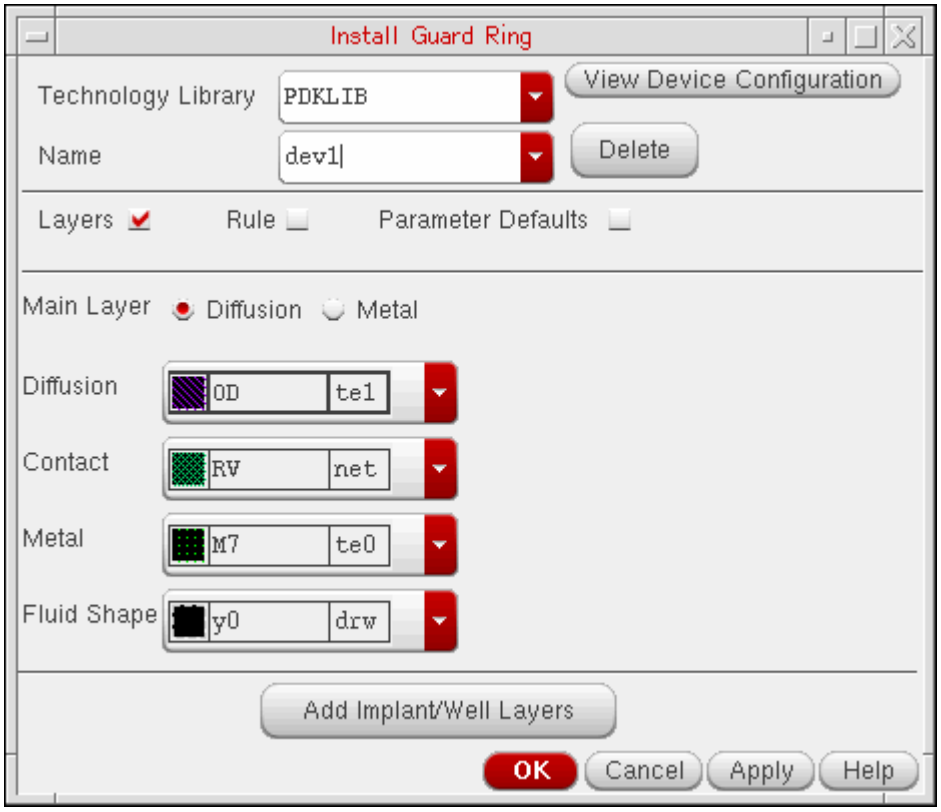
   a. Select either the *Diffusion* or *Metal* radio button to define the *Main Layer*. By default, *Diffusion* is selected.

   b. From the *Diffusion* list, choose the diffusion layer to enclose the *Metal* layer.

   c. From the *Contact* list, choose the layer for vias.

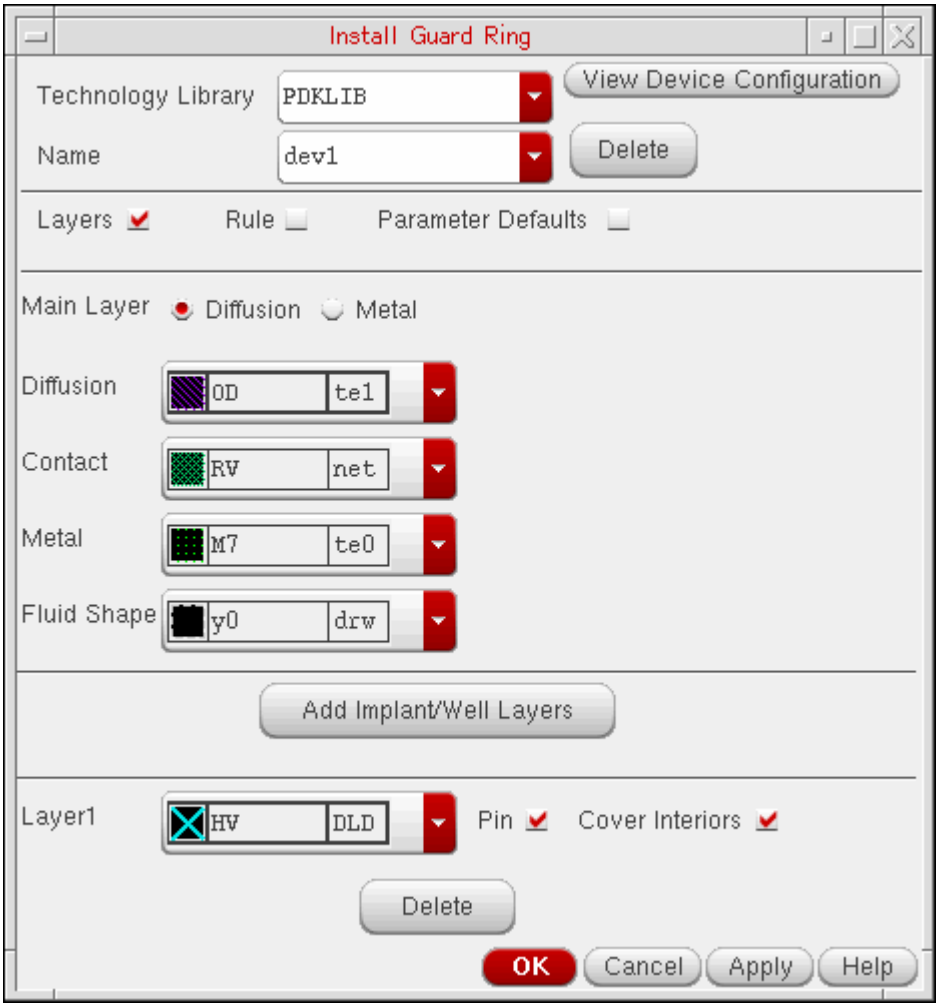
   d. From the *Metal* list, choose the metal layer to enclose the *Contact* layer.

e. From the *Fluid Shape* list, choose the LPP that Virtuoso will use to represent fluid shape of the FGR instances. The fluid shape is a non-maskable layer. Use it to edit the shape of a FGR instance. By default, *y0* drawing is selected.



f. Click *Add Implant/Well Layers* if the material you select for diffusion layer needs to be enclosed by an implant or well layer.

The *Add Implant/Well Layers* section appears at the bottom of the form.



i) In the *Layer1* list, choose an implant or well layer.

ii) Select the *Pin* check box to make the implant or well layer a pin and assign it the same connectivity as the metal layer.

iii) Select the *Cover Interiors* check box to cover the interior of the FGR with an implant or well layer.

- If the *Cover Interiors* check box is off, the implant or well layer is drawn only underneath the diffusion, contact, or metal layer portion of the FGR.
- If the *Cover Interiors* check box is on, the implant or well layer is drawn in the empty area inside the FGR.

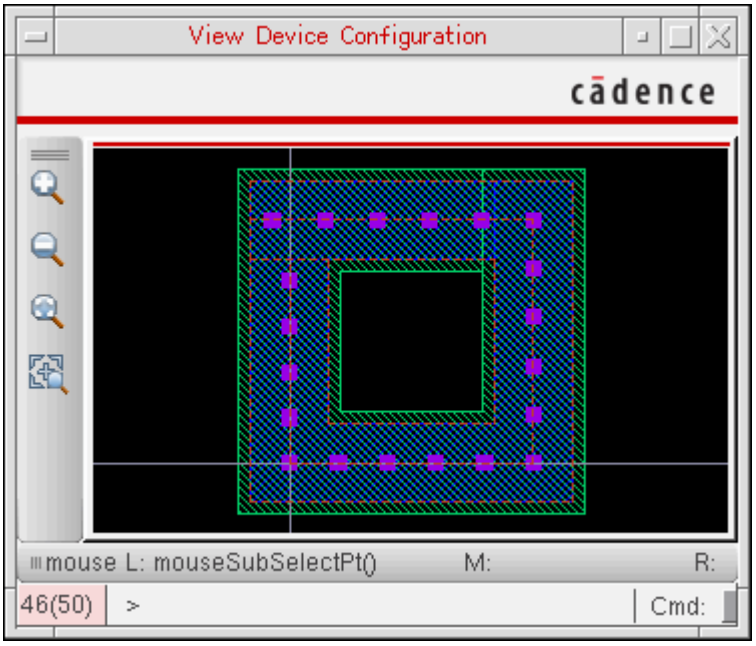
You can preview the FGR devices with and without an implant layer in the *View Device Configuration* window, as illustrated in step 7.

iv) To add more implant or well layers, click the *Add Implant/Well Layers* button and repeat the above two steps.

If you do not remove an implant or well layer for an FGR, click *Delete* for that layer in the *Add Implant/Well Layers* section.

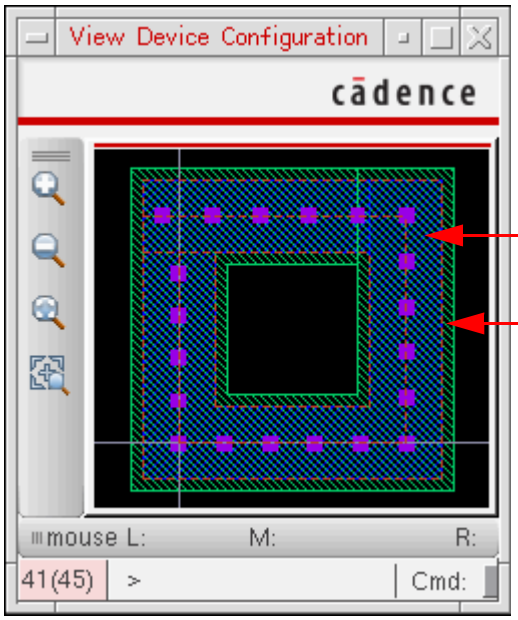
7. Click *View Device Configuration* to preview the sample FGR with the layers set up in the *Layers* section.

The *View Device Configuration* window opens to display a preview of the FGR, as shown below.



The *View Device Configuration* button is enabled only after you set up the layers in the *Layers* section. If you add or change the settings in the *Layers* section, the *View Device Configuration* window updates dynamically.

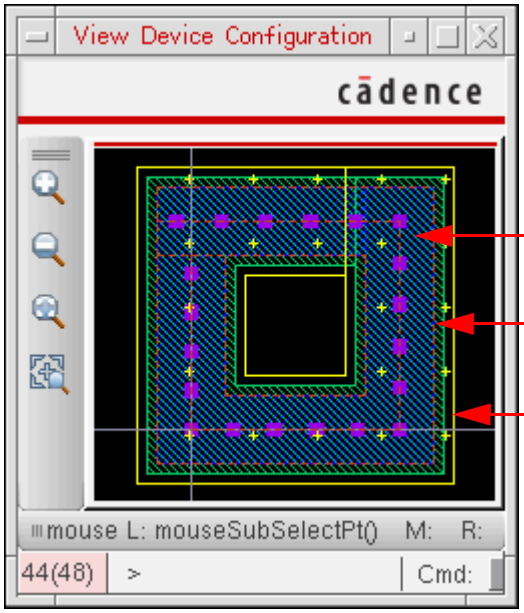
The following figures show a preview of different FGRs with and without the implant layer.



Fluid Guard Ring with No Implant or Well Layer

Metal Layer

Diffusion Layer

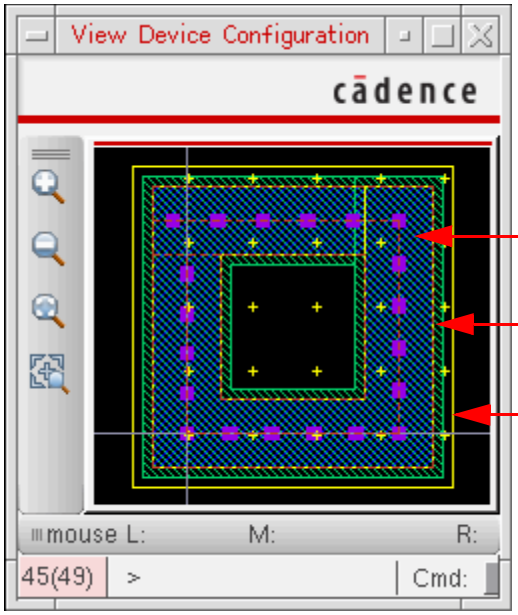


Fluid Guard Ring with Implant Layer

Metal Layer

Diffusion Layer

Implant Layer, *Cover Interiors* is off



Fluid Guard Ring with Implant Layer

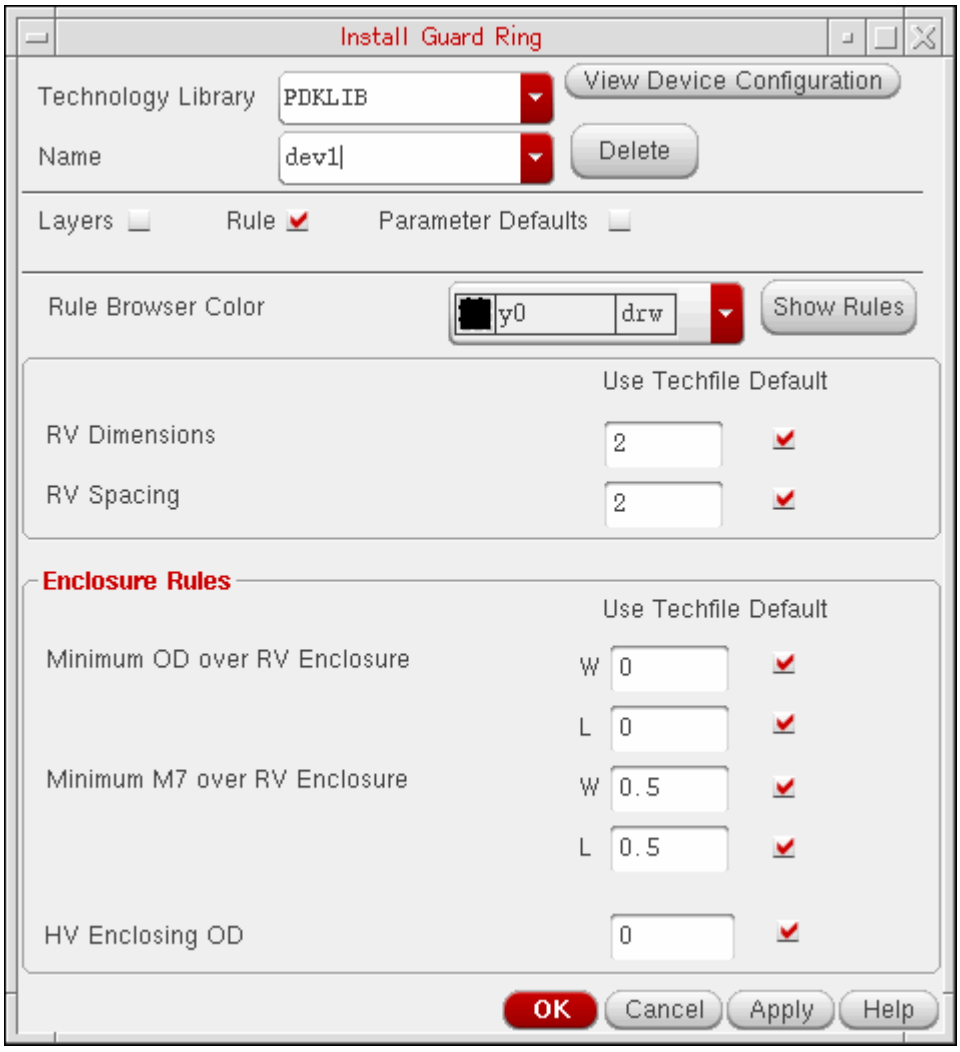
Metal Layer

Diffusion Layer

Implant Layer, *Cover Interiors* is on

Note: The *View Device Configuration* window closes if you select a different FGR from the *Name* list.

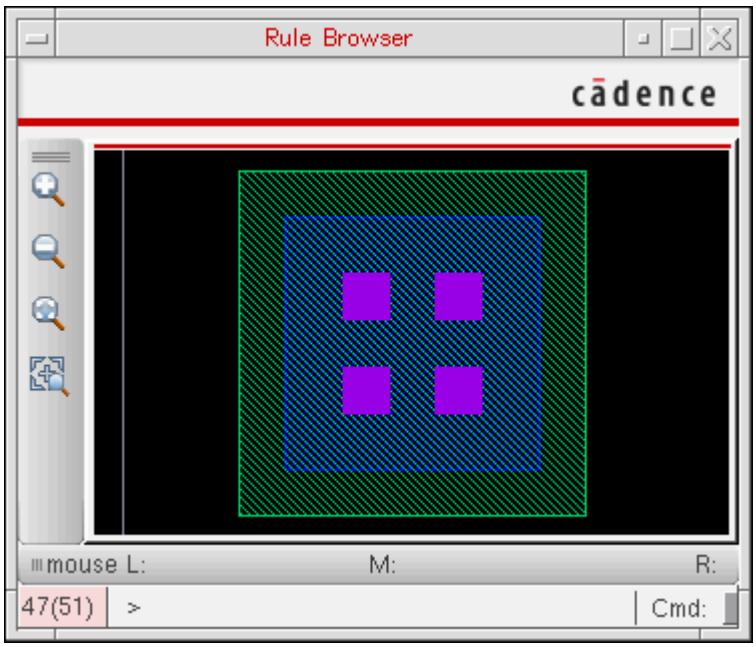
8. Select the *Rule* check box.



a. From the *Rule Browser Color* list, select the color of the arrow to use in the *Rule Browser* window.

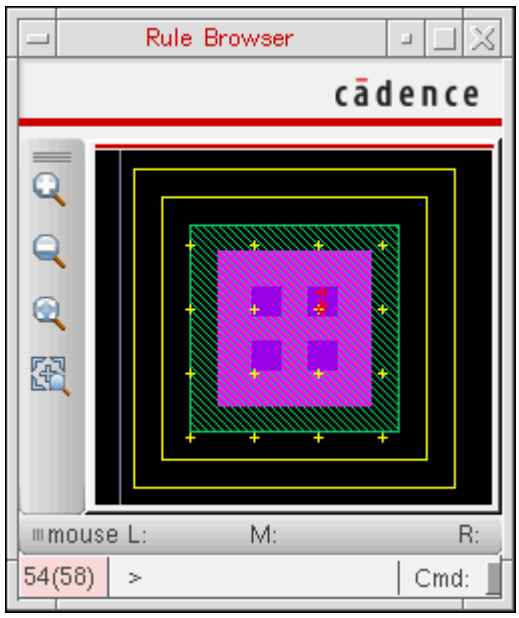
b. Click *Show Rules* to open the *Rule Browser* window.

The *Rule Browser* window shows a graphical representation of a generic FGR device and indicates the physical dimension with a double-headed arrow.

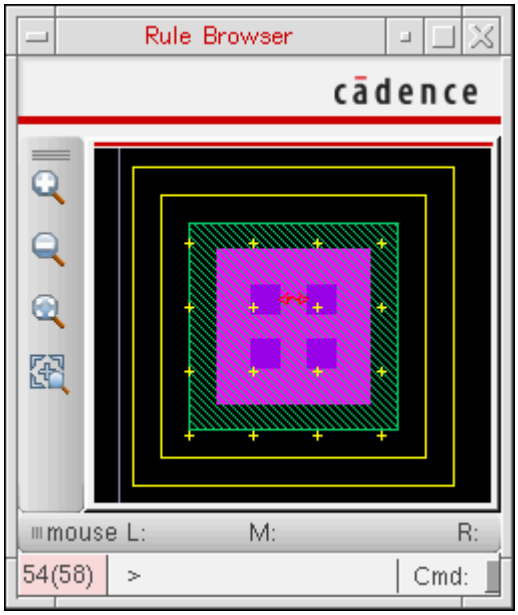


c. Place the cursor in the fields in the *Rule* section to view the corresponding dimension in the *Rule Browser* window.

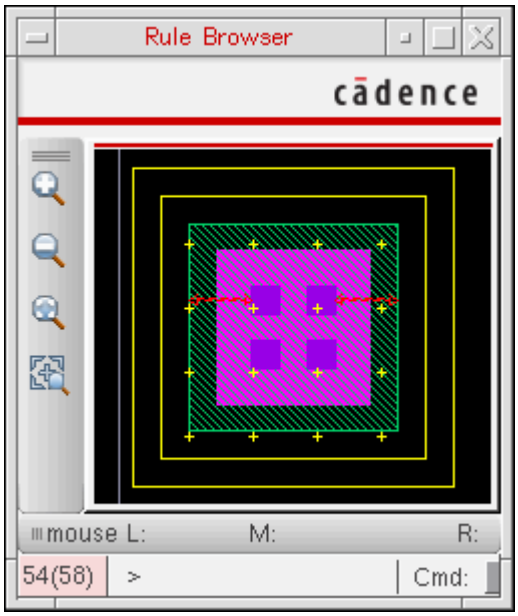
The following figures illustrate the contact dimensions, contact spacing, diffusion and metal layer enclosures over contact, and implant and well layer enclosures over diffusion rules of an FGR that comprises an implant and a well layer.



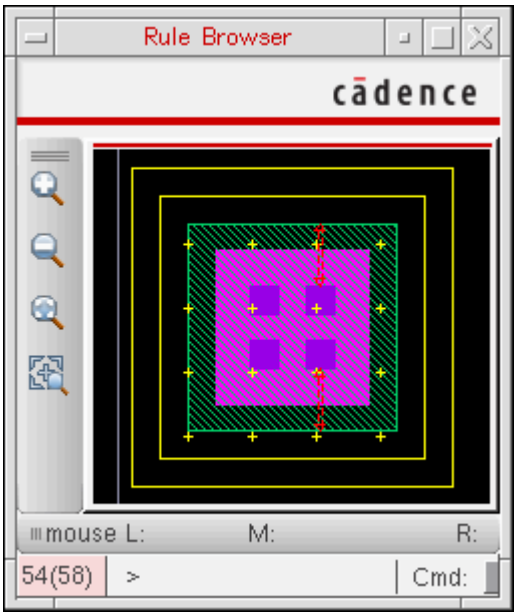
via2 Dimensions



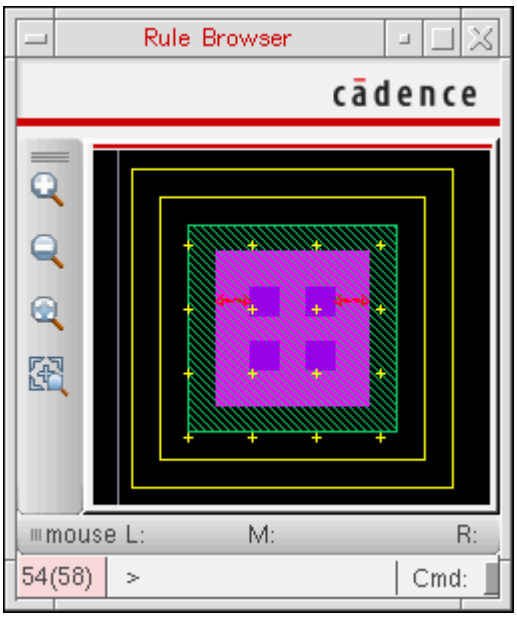
via2 Spacing



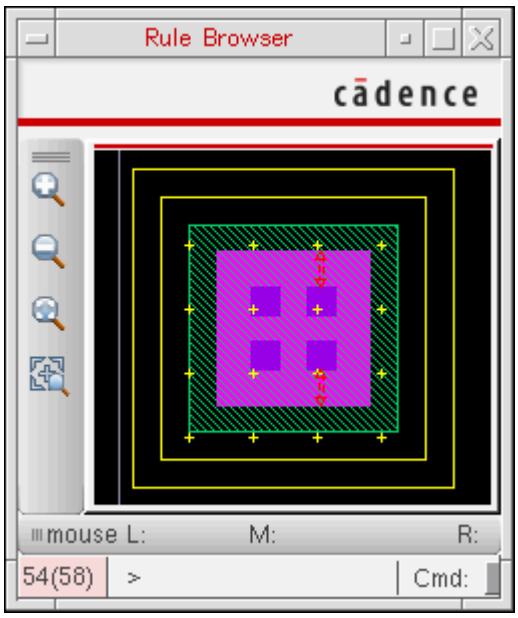
pdiff Enclosing via2 (W)



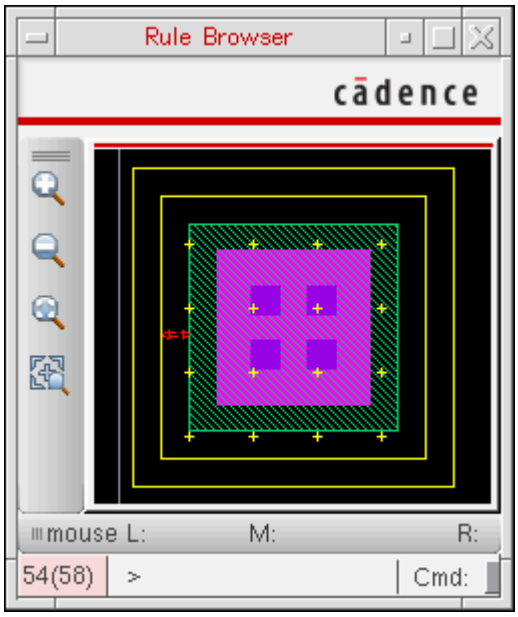
pdiff Enclosing via2 (L)



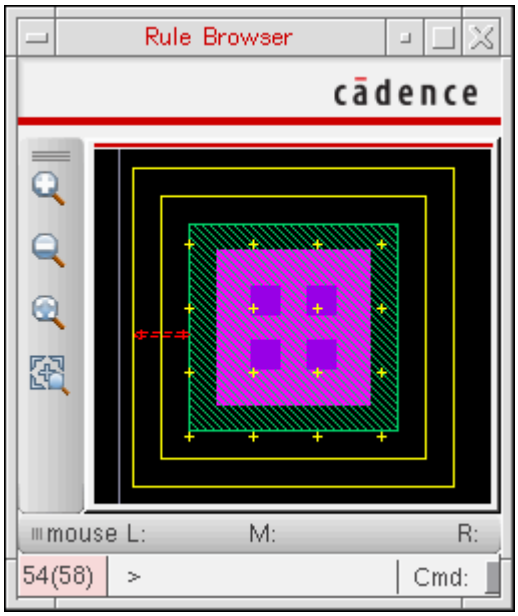
metal2 Enclosing via2 (W)



metal2 Enclosing via2 (L)



pimplant Enclosing pdiff



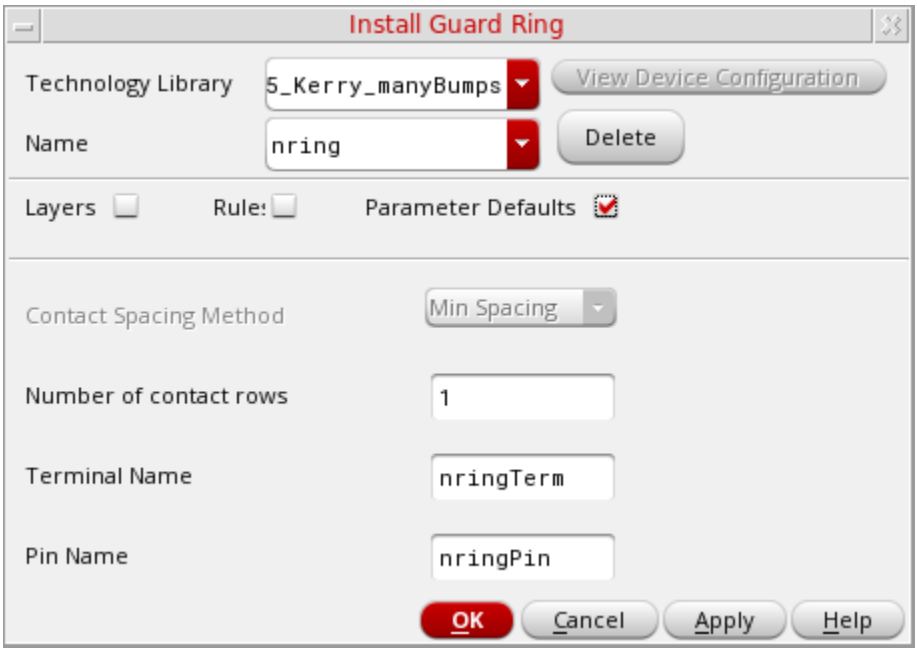
nwell Enclosing pdiff

Note: The *Rule Browser* window closes if you select the *Layers* or the *Parameter Defaults* check box.

d. To reset the contact *Dimensions*, *Spacing*, or enclosure field values to technology file defaults, select the *Techfile Default* check boxes against the fields. The value reverts to the default value from the technology file.

By default, the values in these fields are populated from the foundry constraint group in the technology file. If the values in the form match the default values from the technology file, the *Techfile Default* check box is on. You can override the defaults if required. If you do so, the *Techfile Default* check box turns off. Ensure that the diffusion and metal enclosure values are greater than technology library defaults.

9. Select the *Parameter Defaults* check box.



a. In the *Number of contact rows* field, update the values as required. The default value is `1`.

b. Specify the *Terminal Name* and *Pin Name* for connecting the FGR to a net.

10. Click *OK* (closes the form) or *Apply* to create or update an existing FGR device.

The Install Guard Ring dialog box appears to confirm whether you want to save the technology library on disk. If you click *Yes*, the FGR device is saved to the technology file in virtual memory and written to the technology library on disk. If you dump a

technology file after installing the FGR, you will see the FGR definition in the corresponding technology file class.

You can now create FGRs in your design by using the installed device.

Technology Rules Considered During Installation

The following table lists the technology rules that are applicable during FGR installation:

Form Field	**Applicable Technology Rule(s)** *(Listed in order of precedence)*
Contact Dimension	`minWidth` and `maxWidth` of cut layer
Contact Spacing	`viaSpacing`(4 …) `viaSpacing`(3 …) `minSpacing (One layer)`(…) `minViaSpacing (One layer)` (This rule is considered only if the Shell environment variable, `FGR_MIN_VIA_SPACING_ENABLED`, is set to `1`/`t`/`T`/`true`/`TRUE`.)
Diffusion Over Contact(W) (Diffusion, Cut)	Maximum of `minOppExtension` rule (a, b) `minExtensionDistance`
Diffusion Over Contact(L) (Diffusion, Cut)	Minimum of `minOppExtension` rule (a, b) `minExtensionDistance`
Metal Over Contact(W), (Metal, Cut)	Maximum of `minOppExtension` rule (a, b) `minExtensionDistance`
Metal Over Contact(L), (Metal, Cut)	Minimum of `minOppExtension` rule (a, b) `minExtensionDistance`
Implant/Well Over Diffusion(Implant, Diffusion)	Maximum of `minOppExtension` rule (a, b) `minExtensionDistance`

Note: You can install an FGR with implant layer enclosure value less than the technology default.

3

Creating Fluid Guard Rings

After you have installed a fluid guard ring (FGR) as a device class in the technology file, Virtuoso Layout Suite enables you to create and edit the FGRs. For more information about FGR installation, see Chapter 2, “Installing Fluid Guard Rings.”

You can create FGRs by either drawing one of the following: path, rectangle, or polygon, or by using the wrap mode. Each of these different modes are represented by a tab on the Create Guard Ring Form. You can also create FGRs as concentric rings. During FGR creation, you can specify the method for distributing contacts over the created FGR.

You can dynamically view all the shapes of an FGR while creating, stretching, or splitting it. For more information, refer to the Dynamic Display of the Fluid Guard Ring Contents section.

Note: Ensure that the *Display Stop Level* (*Options – Display*) is set to a value higher than `0` for viewing FGR layers. At *Stop Level* `0`, an FGR is visible as an instance in the design.

The layout of the Create Guard Ring form and the graphical elements visible on it can be customized using the triggers and SKILL functions specifically designed for the purpose. The following types of customizations are possible:

- Create a new Create Guard Ring form from the beginning to open and use from other Virtuoso tools like Module Generator (Modgen). (Supported from IC6.1.6 ISR6 and ICADV12.1 ISR8 onwards)
- **(ICADVM20.1 Only)** Change the properties of the pre-defined (system) fields or components available on the form, and create new components to suit your design requirements. (Supported from ICADV12.1 ISR5 onwards)

For more information, see Customizing the Create Guard Ring Form.

Steps to Create a Fluid Guard Ring

To create an FGR instance:

1. In the design window, choose *Create – Fluid Guard Ring*.

This opens the Create Guard Ring Form.

2. Click a tab based on the mode you want to use for creating the FGR. The following modes are available:
   - Wrap Mode
   - Path Mode
   - Rectangle Mode
   - Polygon Mode

Tip

Alternatively, you can enable the *Guardring* toolbar from the *Window – Toolbars* menu and click the required icon for creating the related type of FGR. This also opens the Create Guard Ring form. The default tab displayed is based on the icon selected.

3. Configure the settings for creating an instance of the FGR device.
4. Configure the settings on the subtabs: Contact Settings, Implant Layers, and Outer Rings.
5. Create the FGR in the selected mode:
6. At any point, press `F3` to bring up the Create Guard Ring form to change settings for a mode or to change the mode.
7. Press `Esc` or click *Cancel* in the form to finish creating FGRs.

Managing Visibility of Devices on Create Guard Ring Form

You can restrict users from creating FGRs using a particular device. As a result, the device will not be visible in the *Device* drop-down list box of each tab on the Create Guard Ring form. For detailed information, refer to the Hiding a Device from Create Guard Ring Form section.

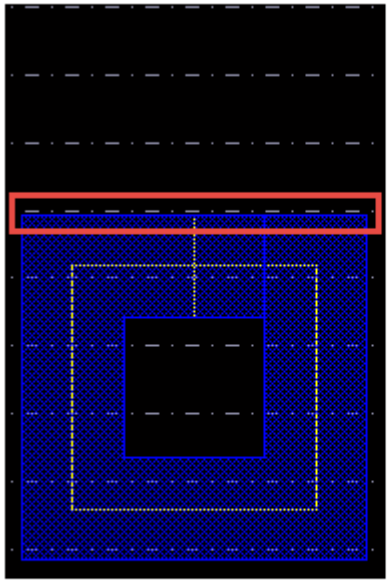
In case, you want to use a specific device to create its instances in the layout and do not see it listed in the *Device* drop-down list box, it means the device is hidden. For information about making such a device visible on the form, refer to the Making a Hidden Device Visible on Install and Create Forms section.

Snapping Fluid Guard Rings to Fin Grids (ICADVM20.1 Only)

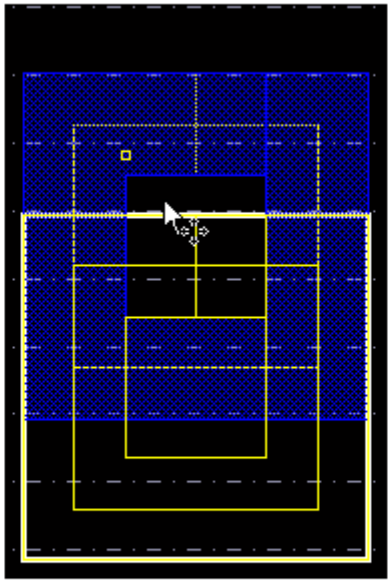
When you create an FGR for a FinFET device, the FGR instance automatically snaps to the underlying snap pattern grids if the *Snap Pattern Snapping* check box is selected in the Layout Editor Options form and the relevant snap patterns are available on the canvas (just like snapping is available for any other instance in Layout L). For detailed information about snap pattern grids in the layout canvas, refer to the FinFET Support in Layout L chapter of the *Virtuoso Layout Suite L User Guide*.

The following images illustrate the difference in creating an FGR instance when automatic snapping is enabled or disabled in Layout L:

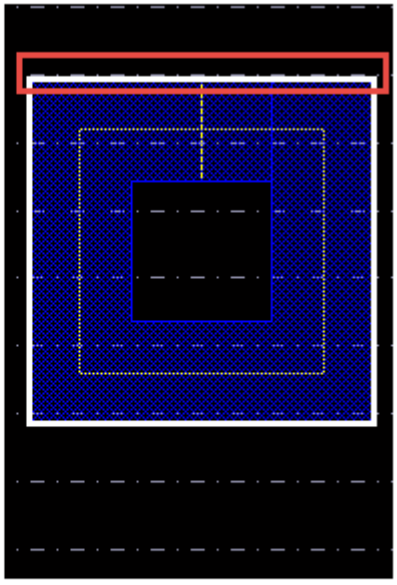
Automatic snapping to snap pattern grid is *enabled*



1. When an FGR instance is created, it snaps to the closest snap pattern grid.

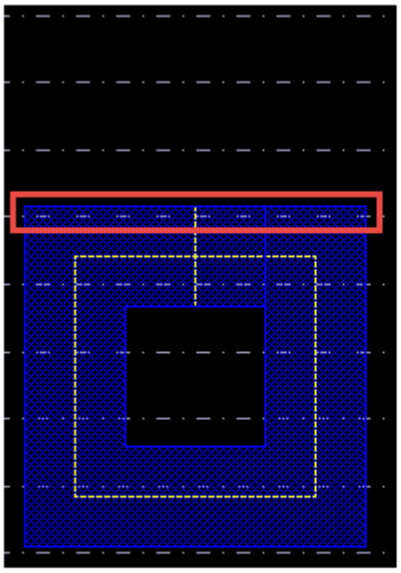


2. Drag the FGR instance to move it to a different snap pattern grid.

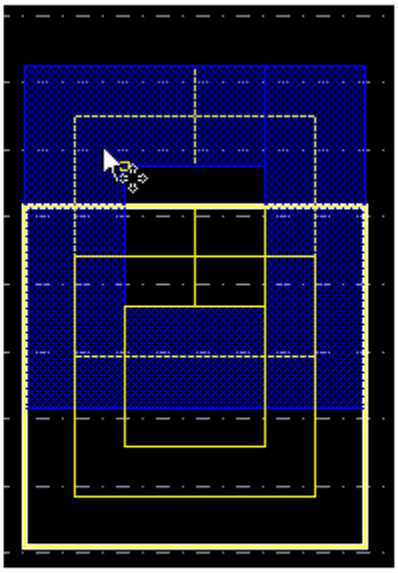


3. The FGR instance again snaps to the underlying snap pattern grid that is closest to the new position on the layout.

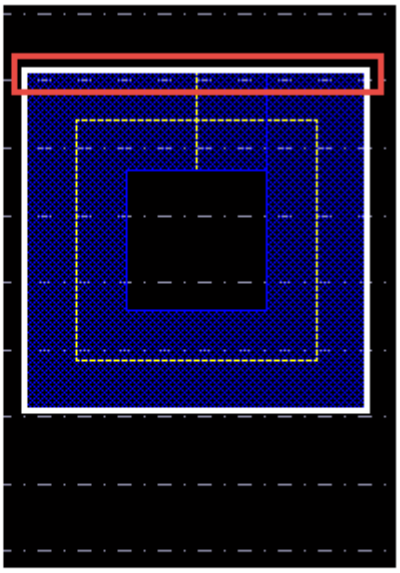
Automatic snapping to snap pattern grid is *disabled*



1. When an FGR instance is created, it does not snap to the closest snap pattern grid.



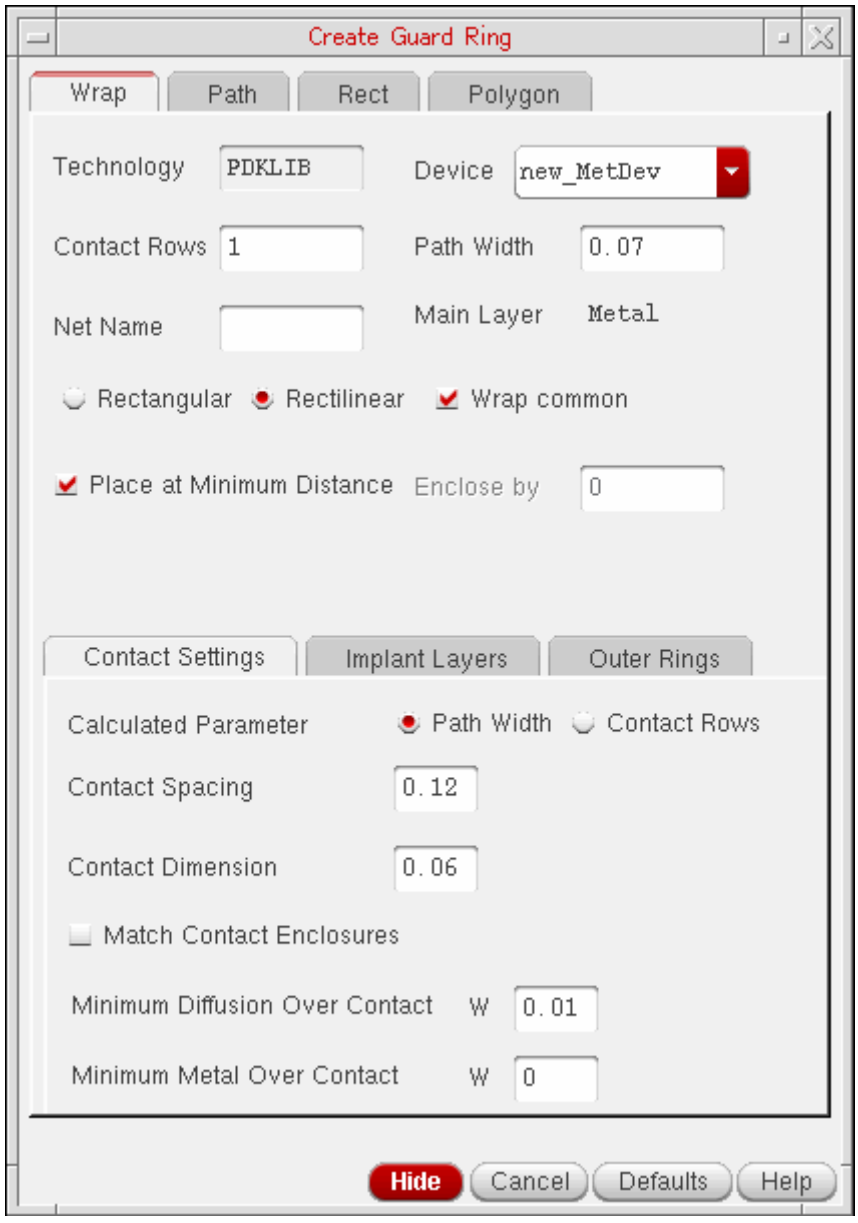
2. Drag the FGR instance to move it to a different snap pattern grid.



3. The FGR instance again does not snap to the underlying snap pattern grid that is closest to the new position on the layout.

Wrap Mode

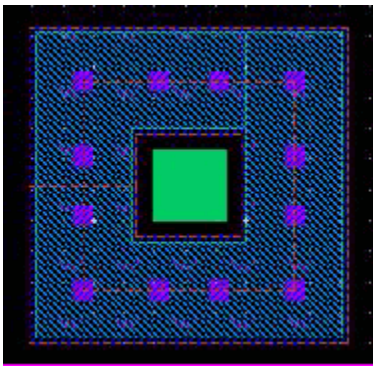
In this mode, an FGR is created around the objects you select. The selection of the **Place at Minimum Distance** check box on the *Wrap* tab of the Create Guard Ring form identifies whether the `minSpacing` rule defined for the object layers, which include both original and derived layers, should be used to compute the spacing between the FGR and the object.



1. Set up the options on the Wrap Tab as required.
2. Use one of the following methods to create the wrap-type FGR:

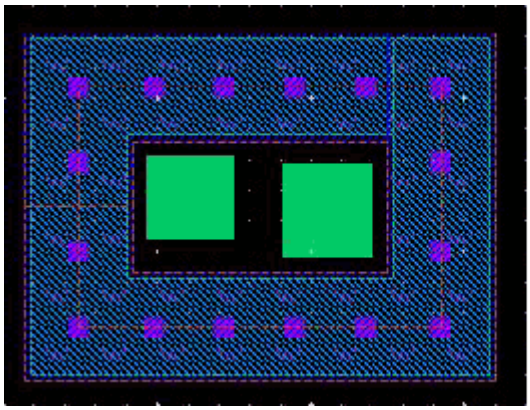
- Click the object around which you want to create the FGR.

  An FGR is automatically created around the clicked object.

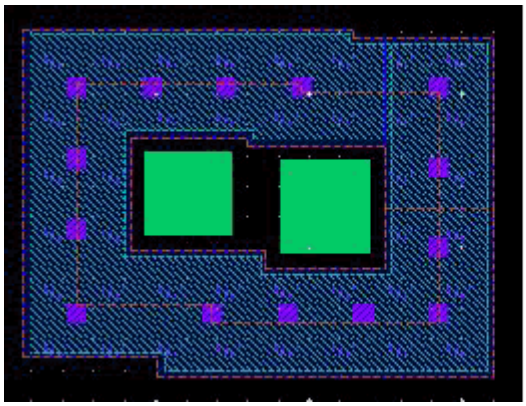


- Select an area on the layout that covers one or multiple objects.
  - Press `Shift` and click an area to select more objects to add them to the selection.
  - Press `Ctrl` and click an object to remove it from the selection.

An FGR is automatically created around the area selected objects.



Rectangular
Wrap common is off.



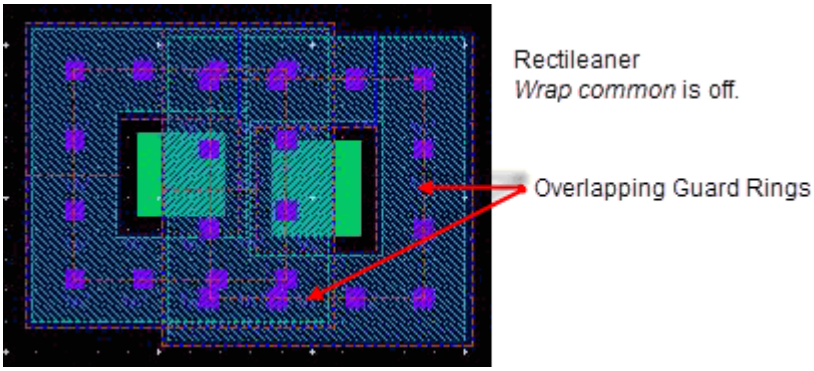
Rectileaner
Wrap common is off.

Note: If *Wrap common* is on, the following parameters determine whether a common or individual FGRs will enclose the selected devices:

- Distance between the devices to be wrapped around
- Path width of the FGR
- Distance of the FGR from the devices

A common FGR is created if the individual FGRs overlap and merge and *Wrap common* is on. To ensure you get a common FGR, you can reduce the distance between the devices or increase the *Path Width* or the *Enclose by* value.

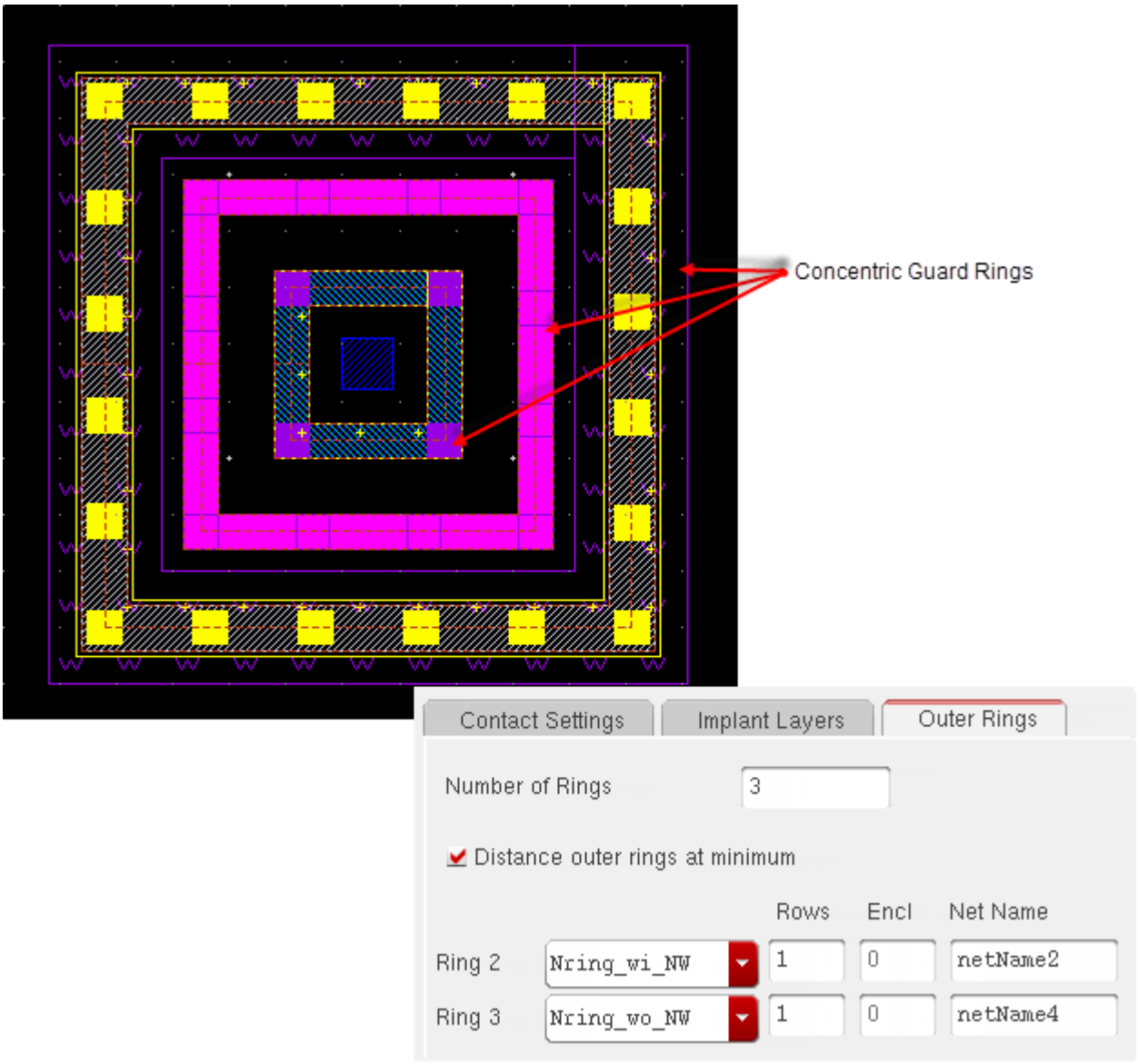
If *Wrap common* is off, the FGR wrap-around is created around individual clicked or area selected shapes, as shown below.



If you create concentric FGRs, specify the number of required rings in the *Number of Rings* field on the *Outer Rings* subtab. For each of the *Ring n* field that gets added to the form, specify values in the following corresponding fields:

- *Rows*: Number of contact rows
- *Encl*: Distance between the consecutive rings
- *Net Name*: Name of the net to which the outer ring should be connected

The example below shows three concentric FGRs created around the central rectangular object. A part of the form shows the settings on the *Outer Rings* tab.



Creating an FGR Automatically

Note: Auto FGR creation in Wrap mode is not supported for advanced nodes or custom FGRs.

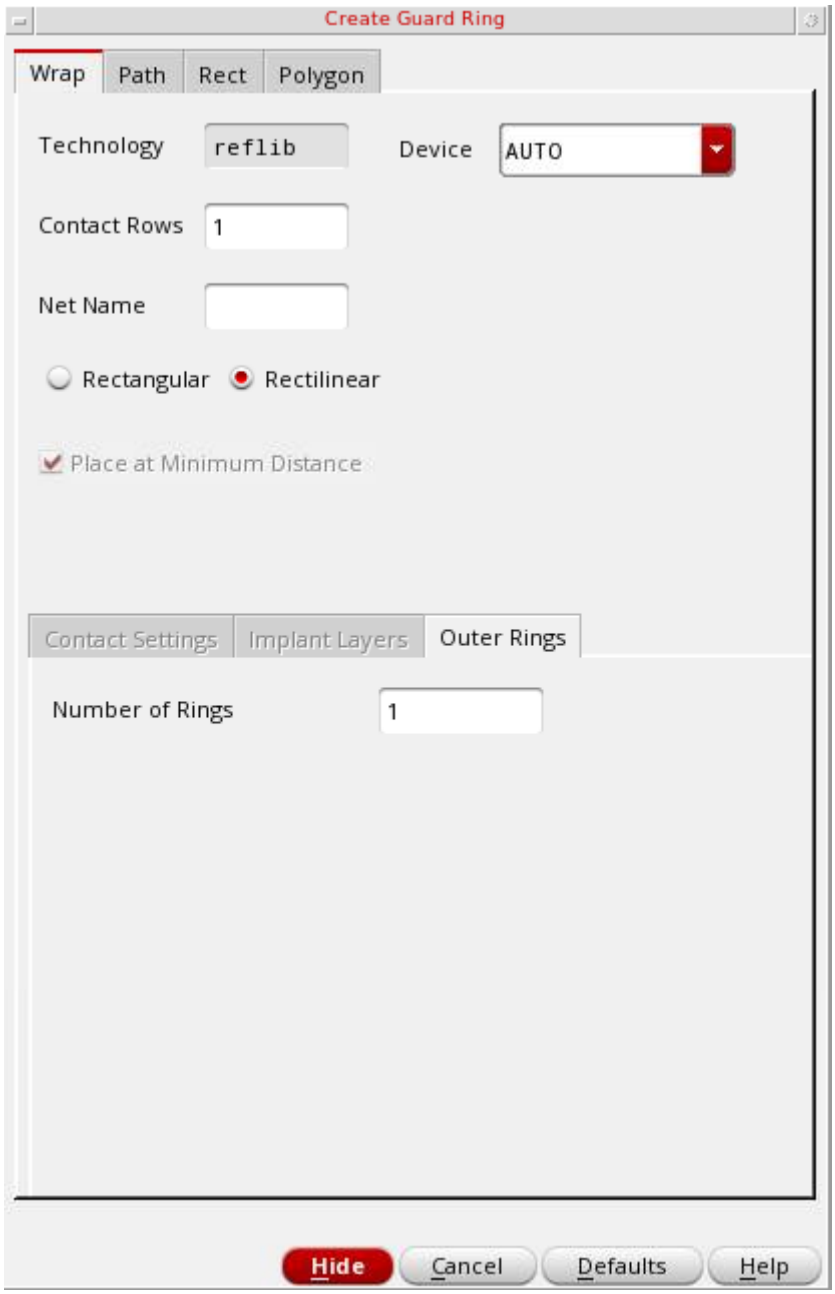
In the Auto FGR creation mode for Wrap, an FGR is created around the selected objects using the default values of the parameters specified in the FGR device definition in the technology file.

FGRs can be created in Auto mode if the `vfoAssociatedDevices` and `vfoAssociatedRings` properties are specified in the technology file. The example below shows these properties.

```
tfcDefineDeviceProp(
; (viewName deviceName propName propValue)
  (layout pgr vfoAssociatedDevices "nmos1v;nmos1v_hvt")
  (layout pgr vfoAssociatedRings "ngr;pgr")
)
```

If any of the above properties are specified in the technology file, and there is at least one VLS-based FGR device defined in technology file, the AUTO option is available in the *Device*

field on the Wrap tab of the Create Guard Ring form. The FGR is automatically created based on these properties in the technology file.

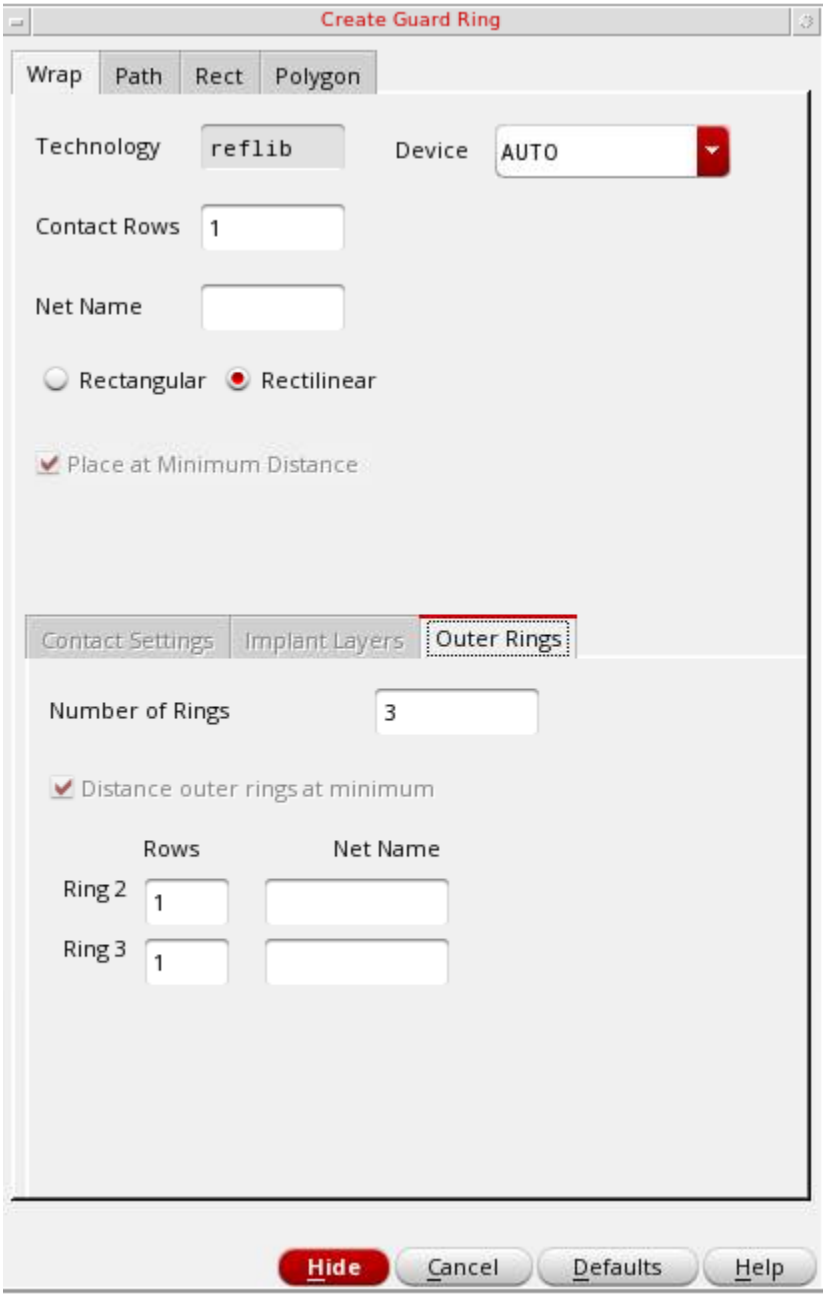


Note: You need to choose the *AUTO* device in the Create Guard Ring form, hide the form, and then click the device to create an AUTO FGR. The device around which the FGR needs to be created can be an MOS device instance or an FGR instance.In addition, the `vfoAssocitedRings` property should be specified for creating rings of FGRs on top of another FGR.

Note: When creating an auto FGR, the *Place at Minimum Distance* option is selected by default.

In the above figure, you can see that the fields on the Outer Rings tab are different when *AUTO* is selected in the *Device* field.

If you specify more than one outer ring, you can specify the rows and net names for subsequent rings.



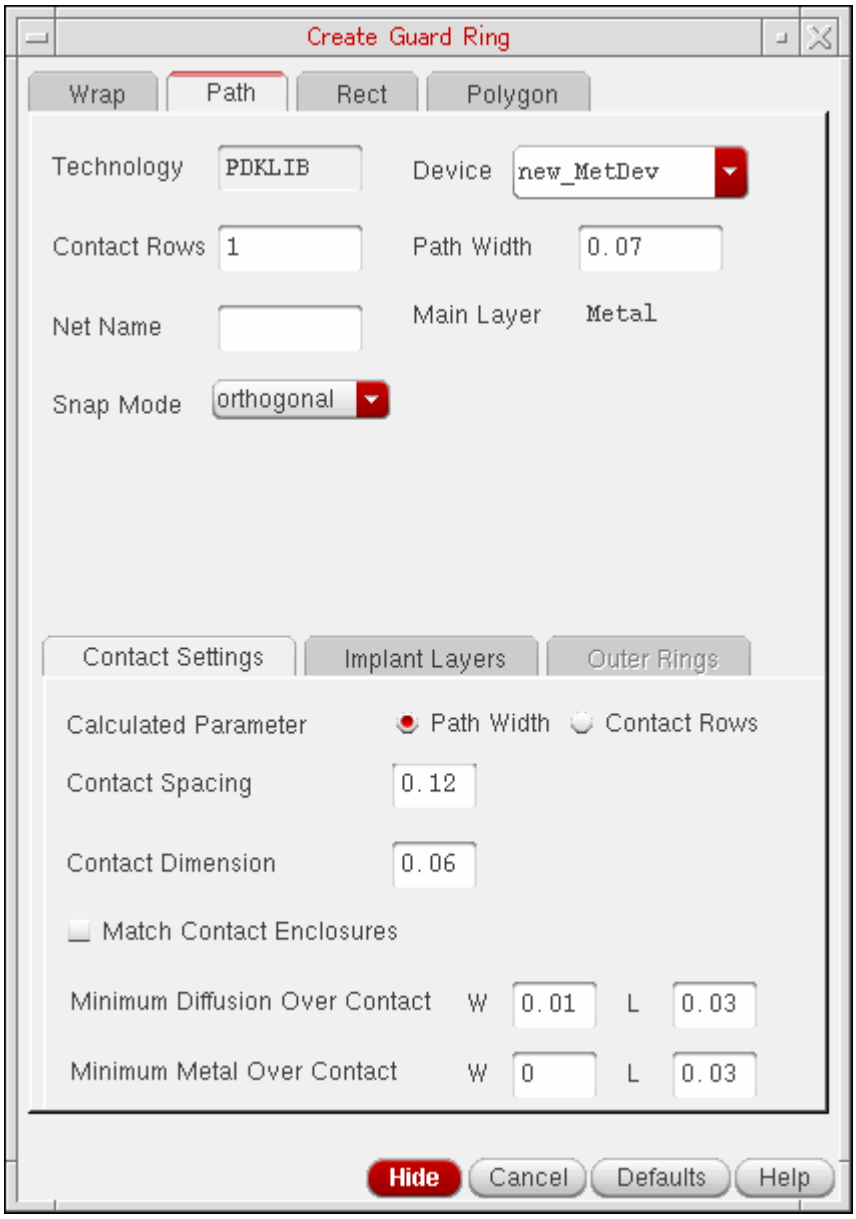
Note: When creating an auto FGR, the *Distance outer rings at minimum* field is selected by default.

You can also create an auto FGR using the SKILL function, `vfoCreateAutoFGR`. In addition, you can assign a bindkey to create an auto FGR. The bindkey will be associated to the SKILL function, `vfoCreateAutoFGR`. For example,

```
hiSetBindKey("Layout" "Ctrl Shift<Key>f" "vfoCreateAutoFGR")
```

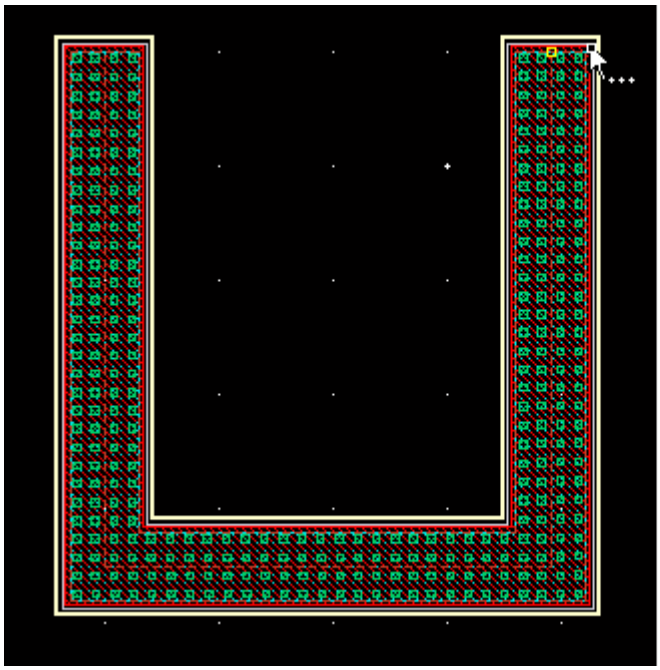
Path Mode

In this mode, an FGR is created based on the path you draw.

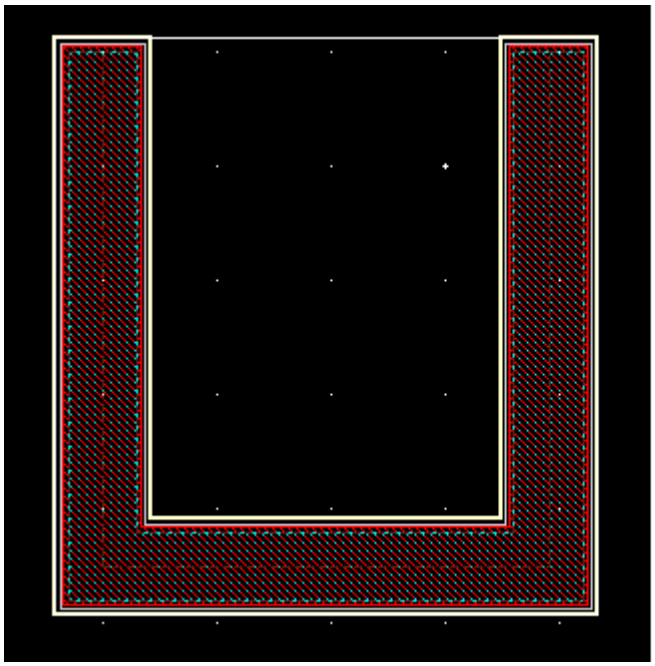


1. Set up the options on the Path Tab as required.
2. Click to define the points of the path around the object around which you want to create the FGR.
3. Double-click or press `Enter` to complete the path.

A path-shaped FGR is created around the object. You can control the width of the FGR by using the *Path Width* field.



Defining the points of the path



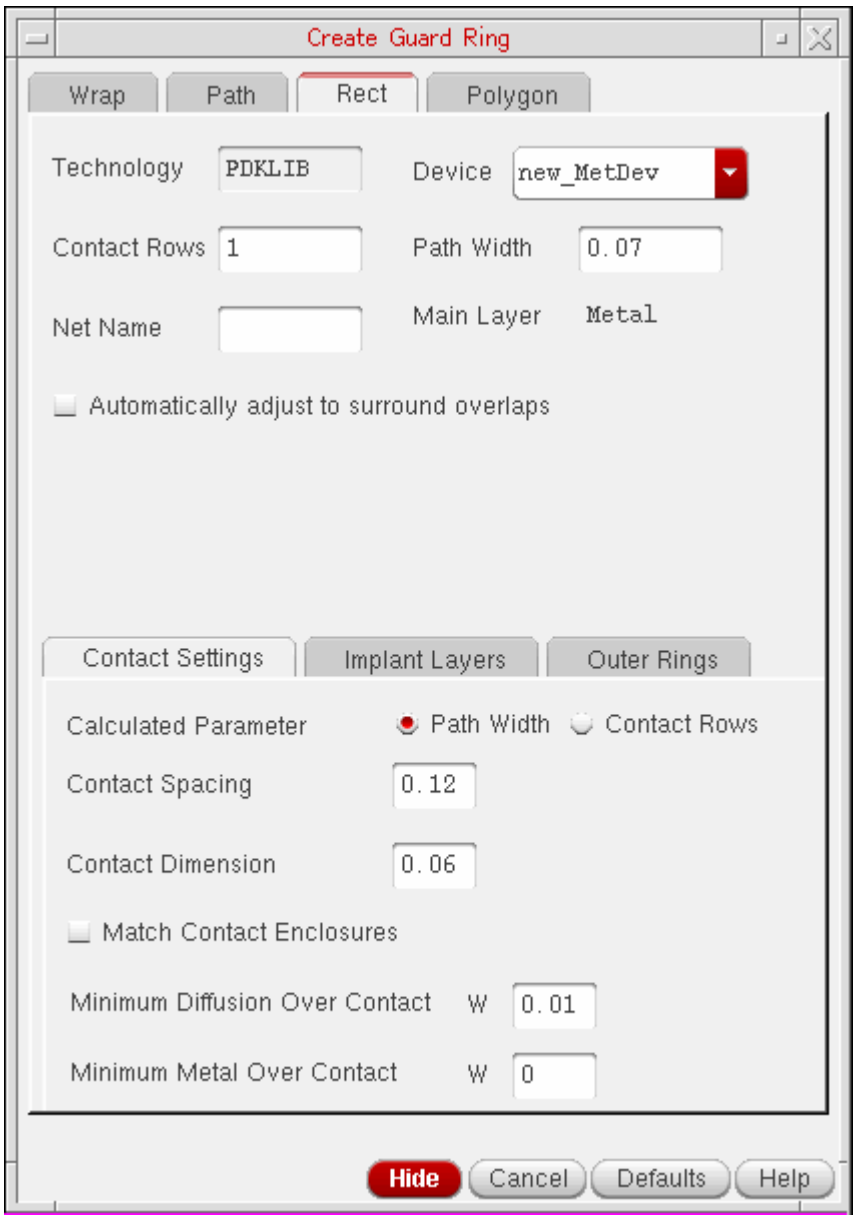
Path Guard Ring

Important

You cannot create a path-type FGR that is self-intersecting.

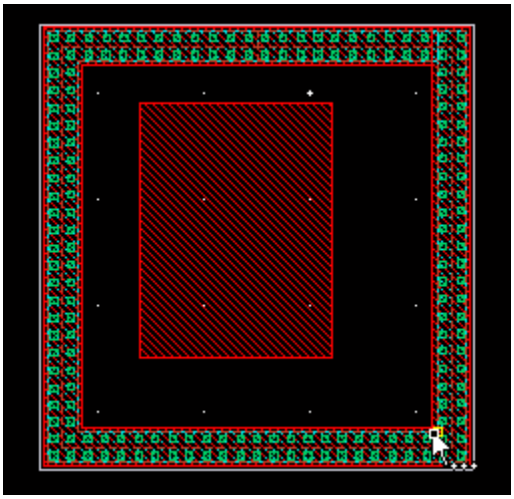
Rectangle Mode

In this mode, an FGR is created based on the rectangle you draw.

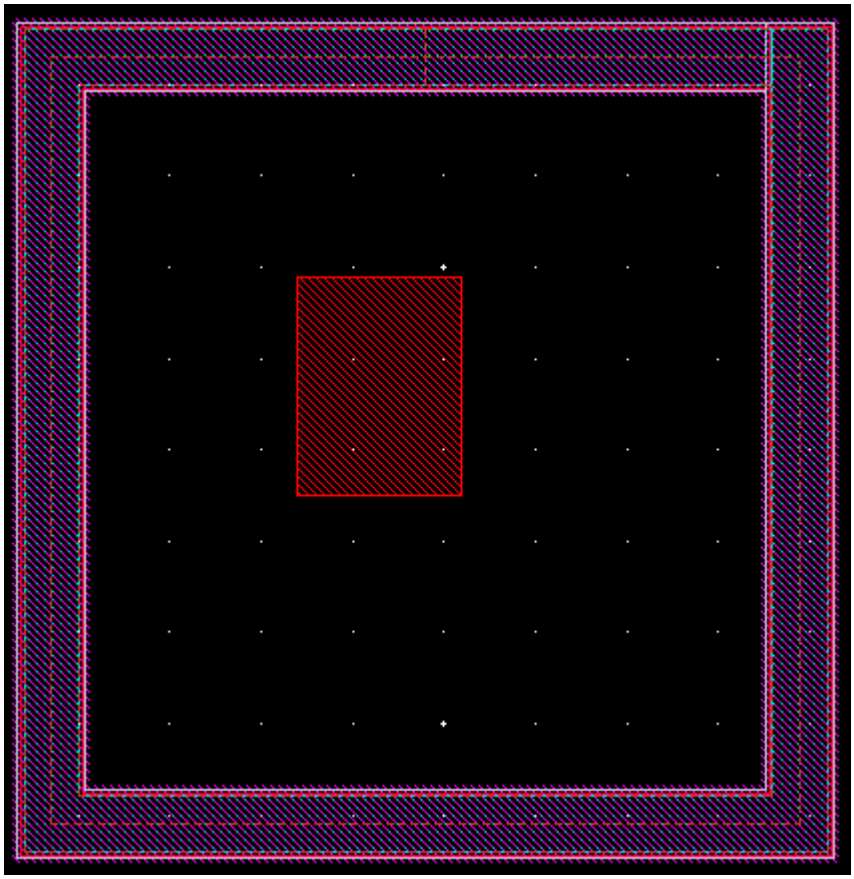


1. Set up the options on the Rect Tab as required.
2. Click to define the opposite corners of the rectangle around the object for which you want to create the FGR.

A rectangular FGR gets created around the object as illustrated in the images below.

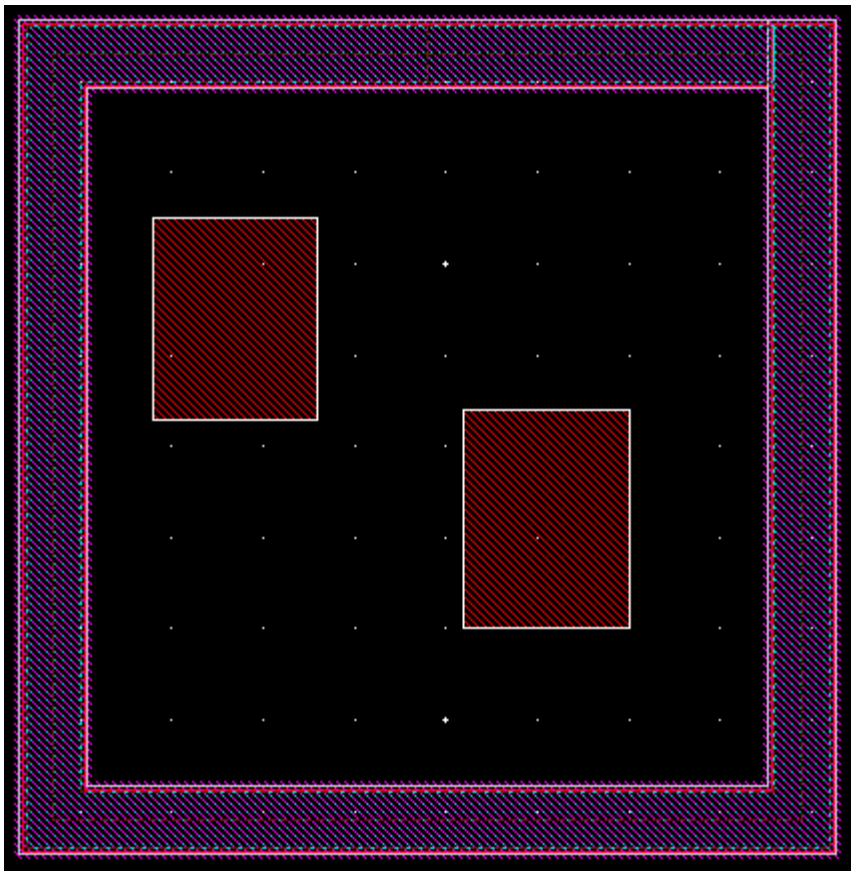


Defining the points of the rectangle.



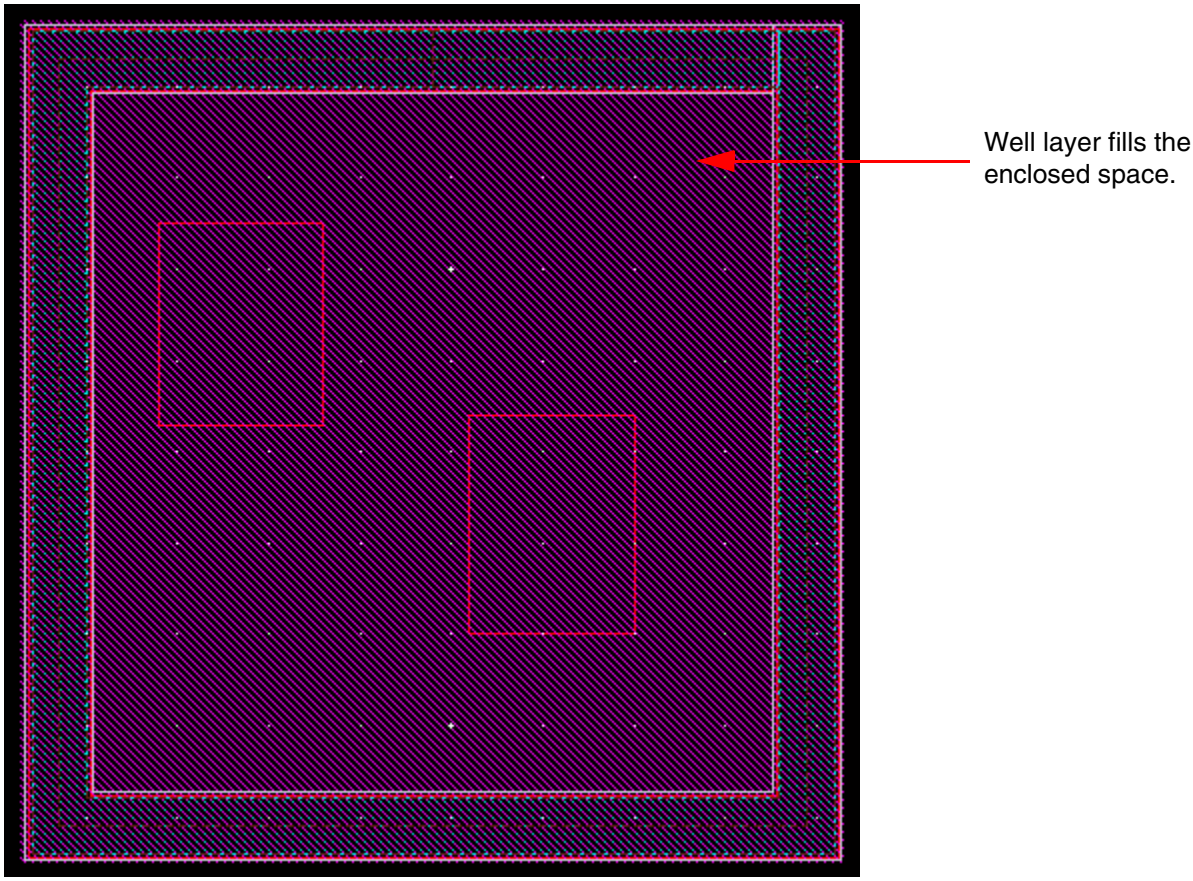
Rectangle Fluid Guard Ring

You can draw a rectangular FGR around multiple objects too as illustrated in the image below.



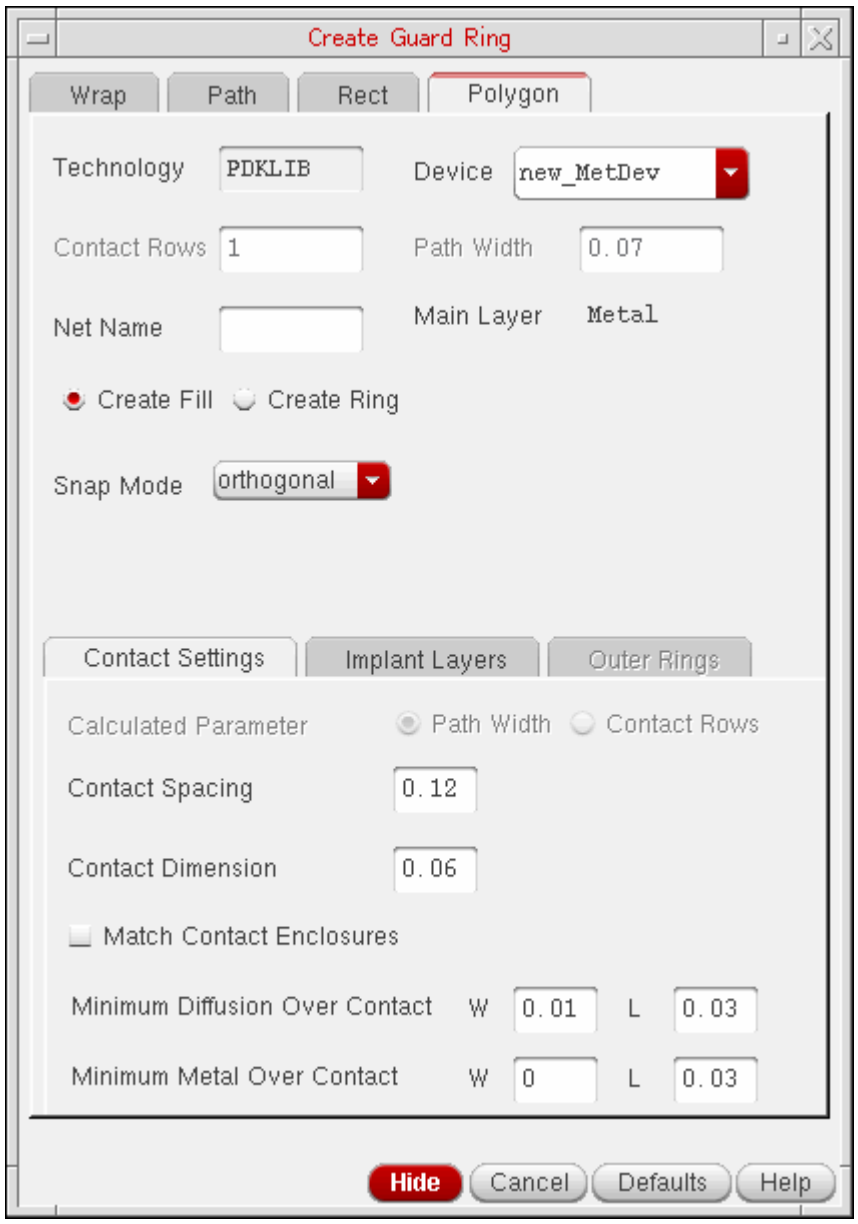
Rectangle Fluid Guard Ring Around Multiple Objects

If you installed the FGR device with the *Cover Interiors* check box selected for the Implant/ Well layer (Layers section), the rectangular FGR enclosure is filled with the implant or well layer, as shown below.



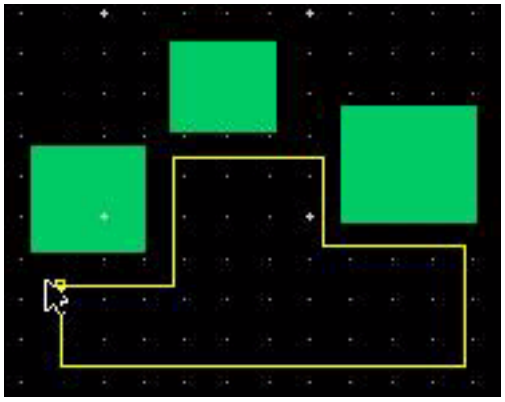
Polygon Mode

In this mode, an FGR is created based on the polygon you draw.

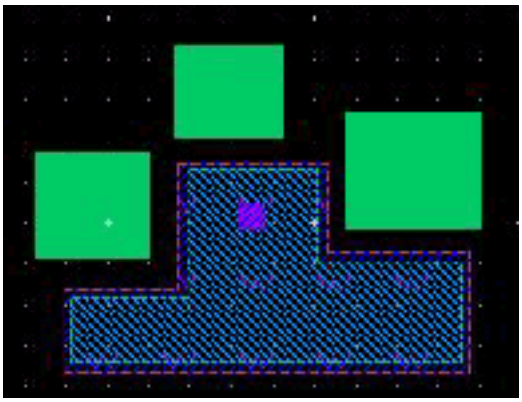


1. Set up the options on the Polygon Tab as required.
2. Click to define the points of the polygon.

A polygon FGR gets created.

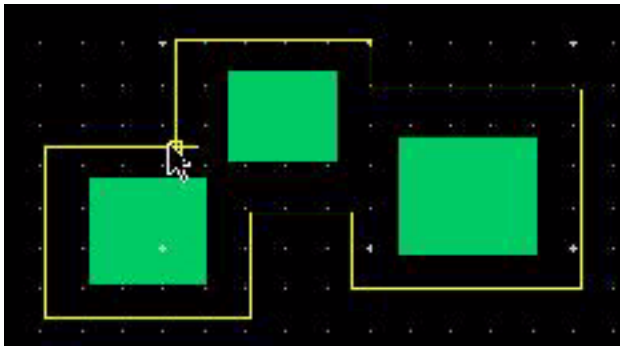


Defining the points of the polygon.

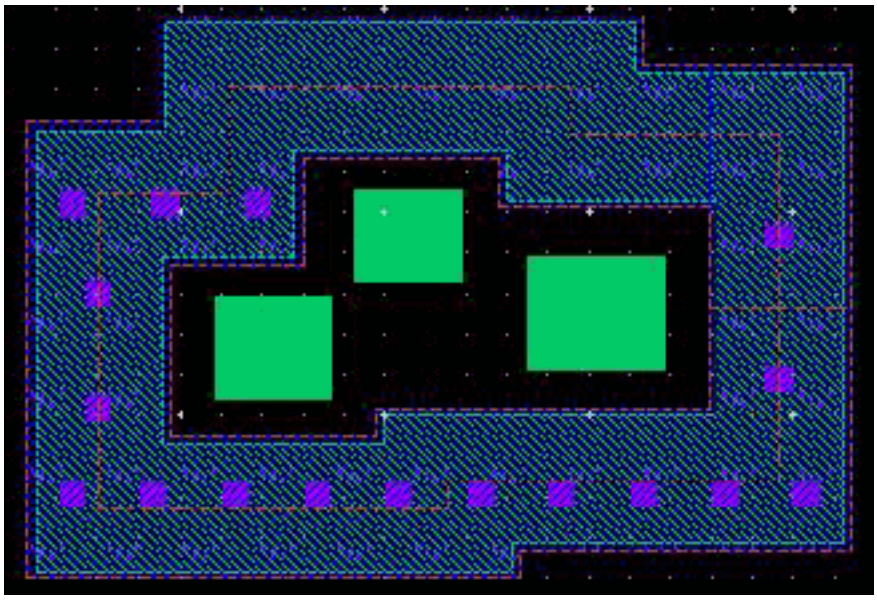


Polygon Fluid Guard Ring Using Create Fill

The figure below shows an example of a polygon FGR where the polygon area is filled with the FGR device material. You can use this method to fill up spaces between objects.



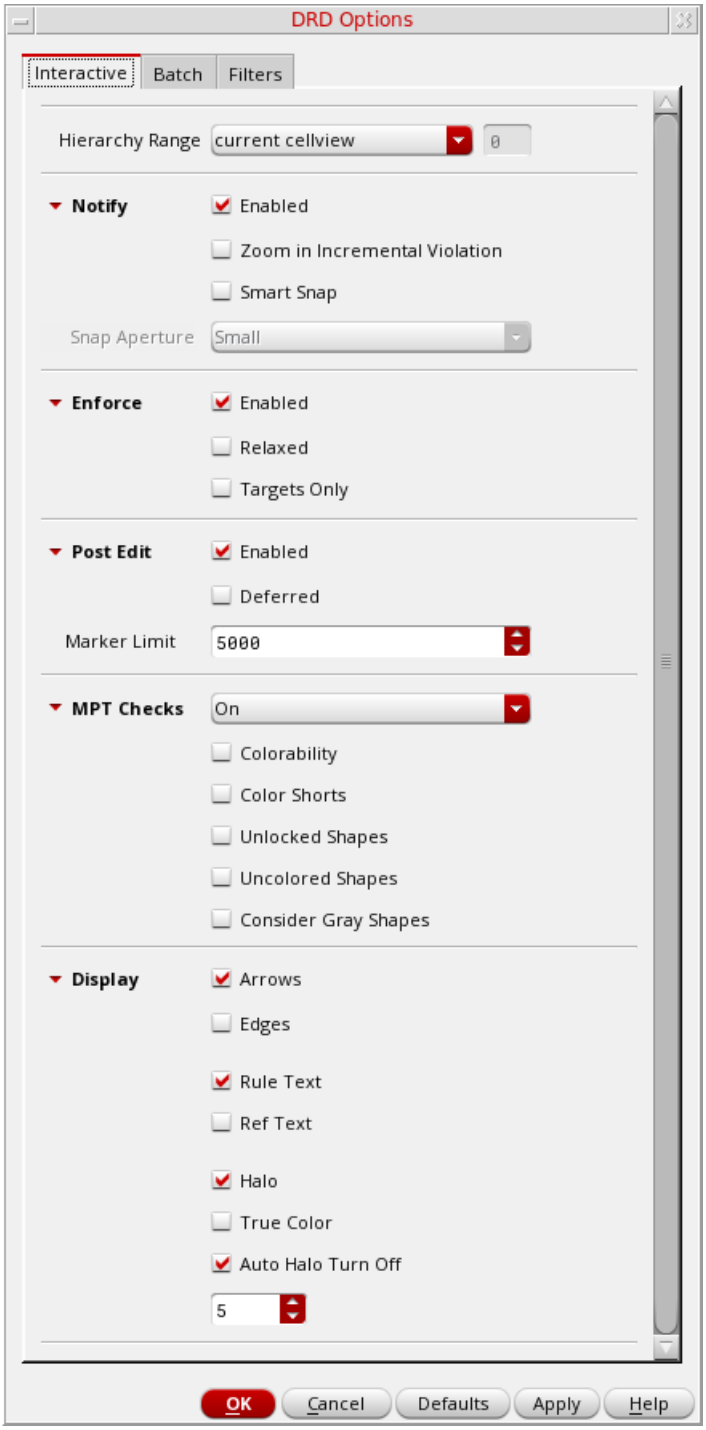
Defining the points of the polygon.



Polygon Fluid Guard Ring Using Create Ring

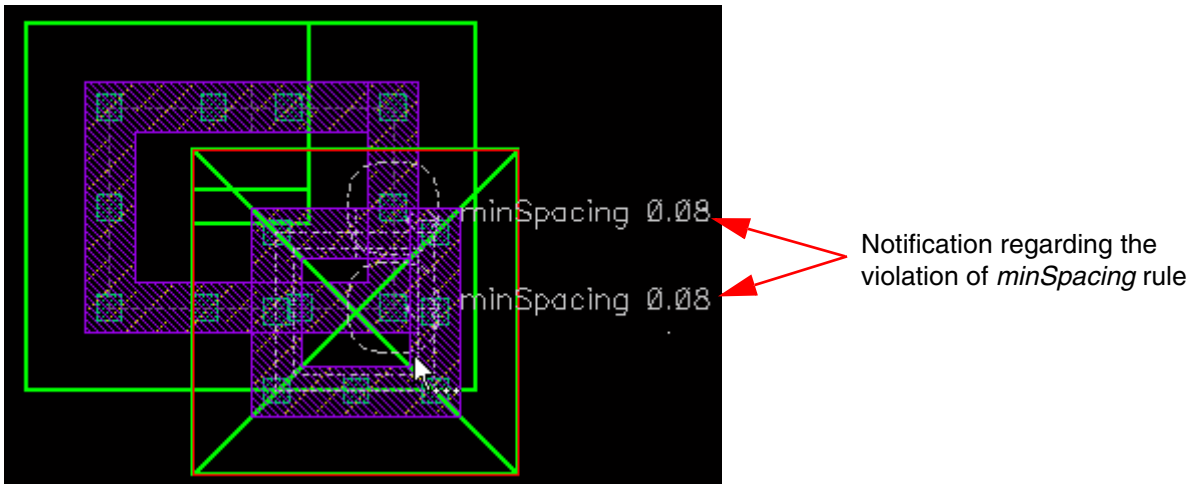
While creating an FGR, you can also specify the design-rule-driven (DRD) rules that need to be checked and applied during the process. This can be done by accessing the *DRD*

Options form through the *DRD* toolbar or the *Options — DRD Edit* menu.



Use the *DRD Options* form to enable the *Enforce*, *Notify*, or *Post Edit* mode at the *current cellview*, *current to bottom*, *current to stop level*, or *current to user level* hierarchy

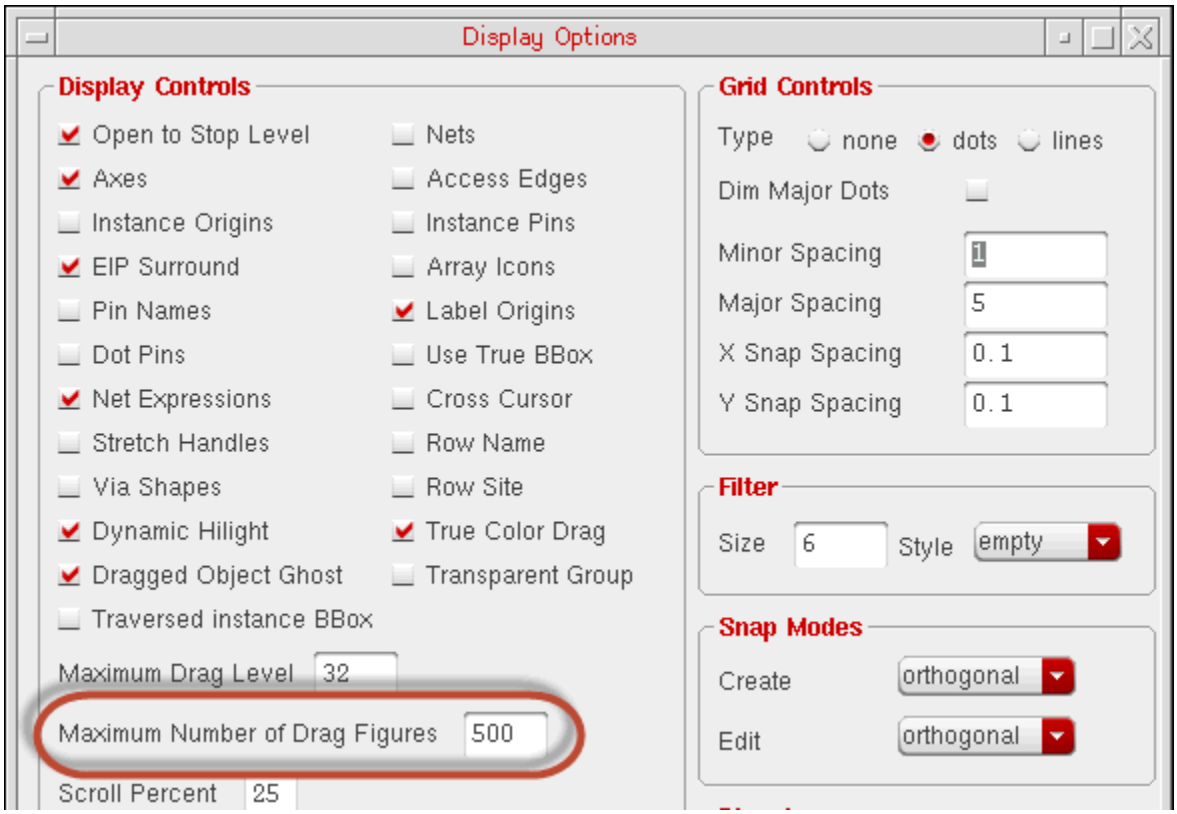
depth. The following image illustrates the notification displayed regarding the violation of the minimum spacing (`minSpacing`) rule. This notification is displayed if the `minSapcing` constraint is selected for *Notify* mode under the *Filters* tab of the DRD Options form.



For more information about the *DRD Options* form, see DRD Form Descriptions in the *Virtuoso Design Rule Driven Editing User Guide*.

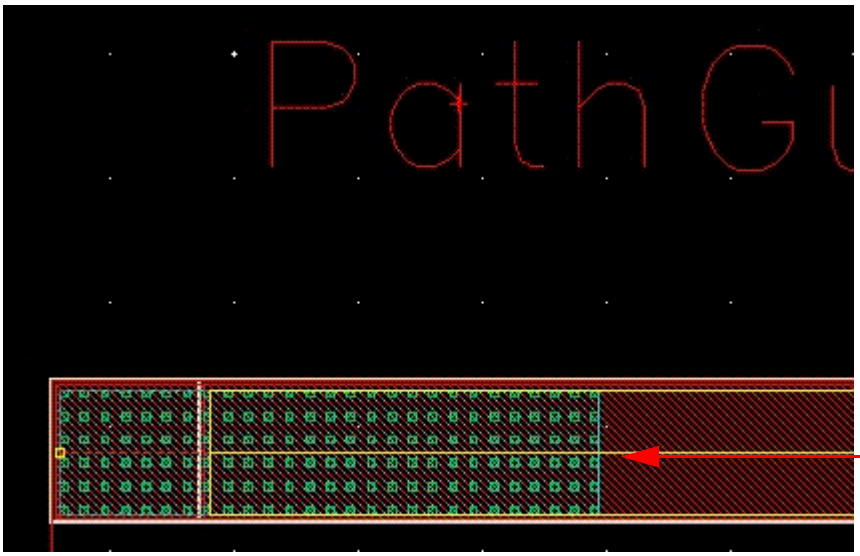
Dynamic Display of the Fluid Guard Ring Contents

While creating, stretching, or splitting an FGR, you can dynamically view all the shapes associated to it. Use the *Maximum Number of Drag Figures* field on the *Options - Display* form to specify the number of shapes that can be dynamically drawn in an FGR.



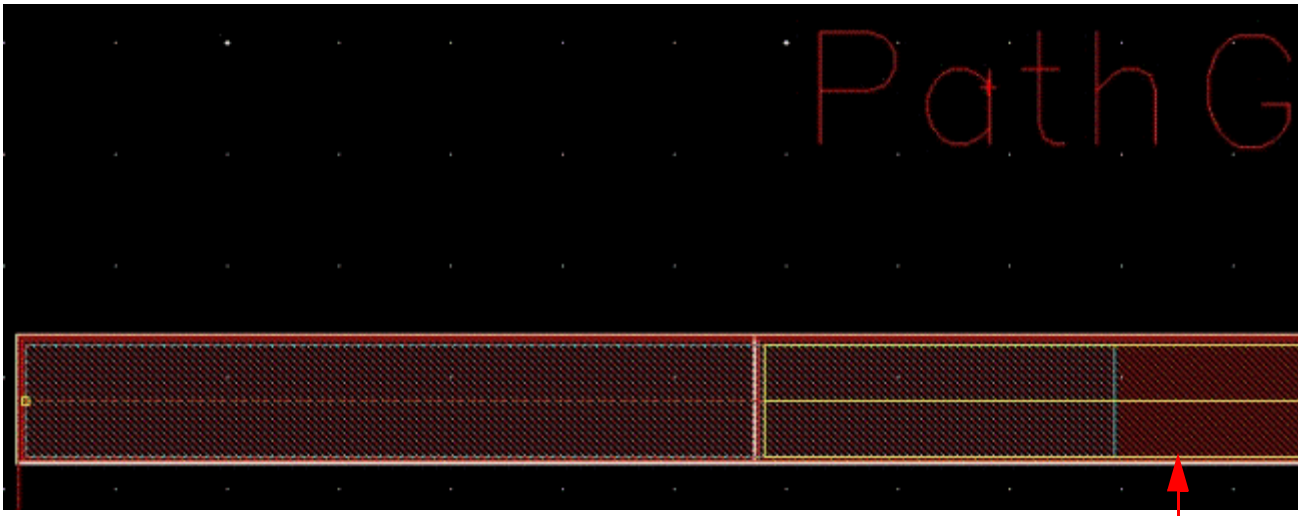
The number of shapes that can be displayed in an FGR depends on the total number of shapes that exist in the FGR and the specified *Maximum Number of Drag Figures*.

- If the total number of shapes in the FGR are less than `maxDragFig`, all the shapes on all the layers of the FGR are displayed.



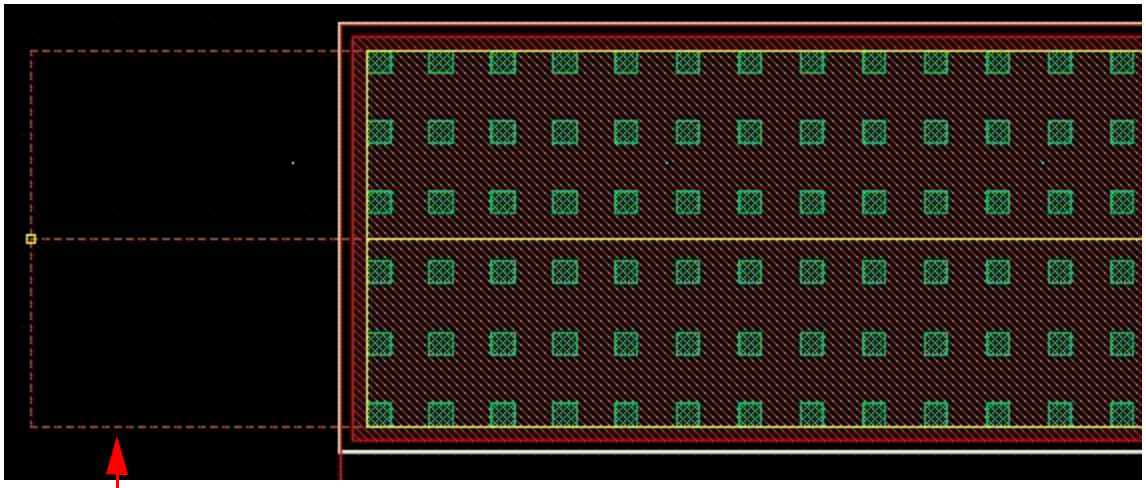
While stretching a fluid guard ring, all the shapes on all the layers in the fluid guard ring are displayed.

- If the total number of shapes in the FGR are more than `maxDragFig`, all the shapes on all the layers of the FGR are displayed, except the contacts.



While stretching a fluid guard ring, all the shapes on all the layers in the fluid guard ring are displayed, except the contacts,

Note: If `maxDragFig` is set to a low value, for instance, 5, such that the number of shapes in the FGR without the contacts is still more than `maxDragFig`, only the fluid shape is displayed.



While stretching a fluid guard ring, only the fluid shape is displayed.

Technology Rules Considered During Fluid Guard Ring Creation

The following table lists the technology rules that are applicable during the creation of an FGR:

Form Field	**Applicable Technology Rule(s)** *(Listed in order of precedence)*
Contact Spacing	viaSpacing(4 …) viaSpacing(3 …) minSpacing (One layer)(…) minViaSpacing (One layer) (This rule is considered only if the Shell environment variable, FGR_MIN_VIA_SPACING_ENABLED, is set to 1/t/T/ true/TRUE.)
Contact Dimension	minWidth and maxWidth of cut layer
Diffusion Over Contact (Diffusion, Cut)	minOppExtension rule (a, b) minExtensionDistance minWidth of diffusion layer
Metal Over Contact, (Metal, Cut)	minOppExtension rule (a, b) minExtensionDistance
Implant/Well Over Diffusion (Implant, Diffusion)	Maximum of minOppExtension rule (a, b) minExtensionDistance minWidth of well/implant layer
Path Width	minWidth of metal layer when metal is the mainLPP.

Important Points to Remember

- If a technology rule has not been defined in the technology file, but is used by an FGR, the system considers its value as zero. For example, the minOppExtension rule is used for identifying the enclosure values of a contact. If you do not define this rule in the technology file, while creating an FGR, the system takes the enclosure values as zero.
- If you installed the FGR device with the *Cover Interiors* check box as selected (that is, set to ON) for the Implant/Well layer (Layers section), the minWidth technology rule is not checked at the time of creating the wrap-type and closed path-type FGRs.
- You can install an FGR device with implant layer enclosure value less than the technology default. For example, consider an FGR device in gpdk045 with Oxide,

`cont`, and `Metal1` as the layers (diffusion, contact, and metal respectively), and `NWell` as the implant. The minimum enclosure of *Nwell over diffusion (Oxide)* in the technology file is `0.09`. Despite this, you can install an FGR device using `0.06` (with a warning message).

- By default, when an FGR is created in the *Wrap* mode and the *Place at Minimum Distance* check box is selected, the spacing between the guard ring and the object is computed based on the `minSpacing` rule defined for the object layers around which the FGR is created. The object layer includes both original and derived layers.
- When the FGR is created in the *Wrap* mode and the directional `minWidth` rule is specified in the technology file for the main layer of the FGR, it will use the maximum value of ("horizontal" "vertical") rule. In case only one-directional rule is specified, it will use this rule for both the directions. The `twoWidths` rule is not supported in the *Wrap* mode.

Using minOppExtension for Placing Contacts in the Corners of a Fluid Guard Ring

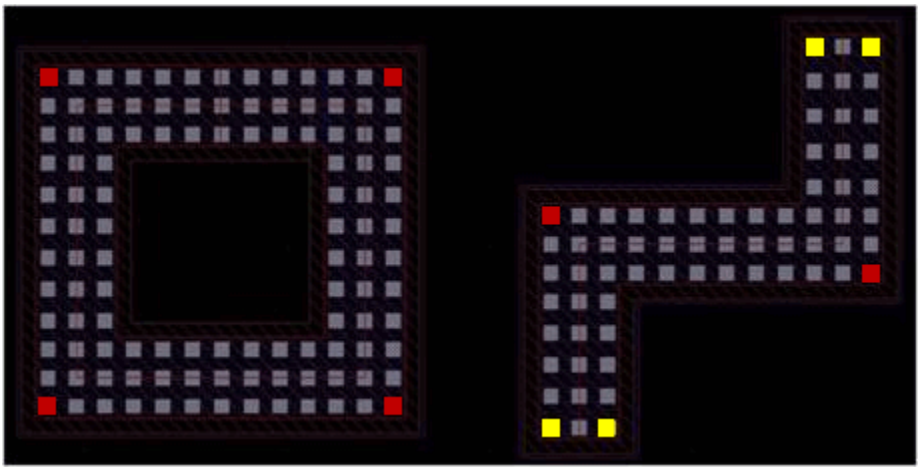
Virtuoso uses the two values defined with the minOppExtension technology rule to identify the placement of the contacts in the corners of an FGR. The technology rule values are compared with the contact enclosure values you specify using the *Minimum Metal Over Contact* and *Minimum Diffusion Over Contact* fields on the *Contact Settings* tab of the Create Guard Ring form.

If the contact enclosure value of any of the directions is greater than or equal to the maximum minOppExtension technology rule value, Virtuoso generates the contacts in the corners of the FGR.

However, if the contact enclosure value of any of the directions is less than the maximum minOppExtension technology rule value, Virtuoso automatically removes the following types of contacts from the corners to minimize the width of the FGR:

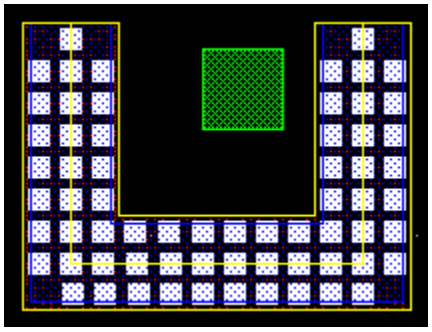
- *Ear-vertex contacts* (highlighted in red color in the image below)

- *End-corner contacts* (highlighted in yellow color in the image below)

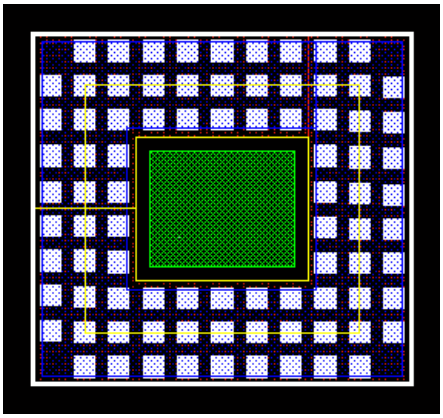


The following images illustrate FGRs where Virtuoso automatically removed the contacts from the corners:

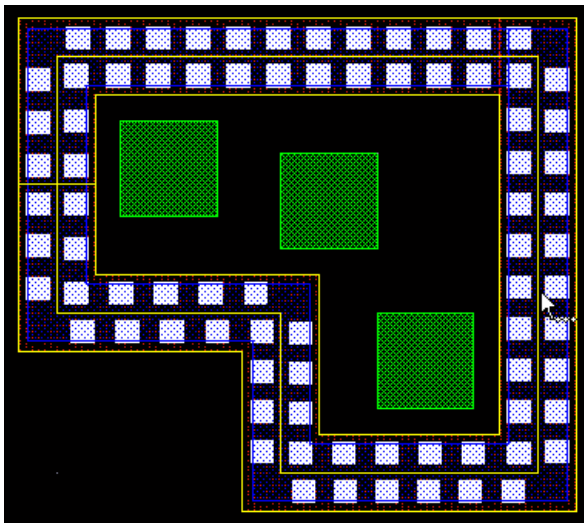
Path Fluid Guard Ring



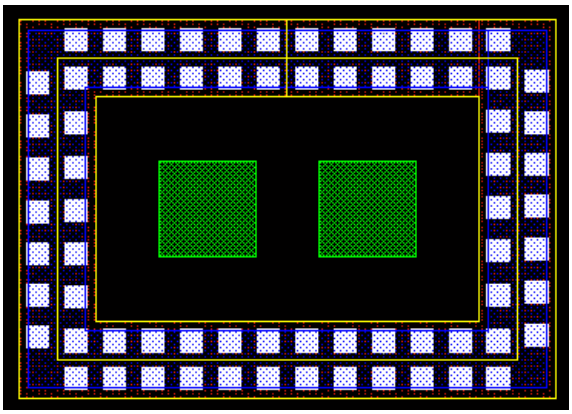
Wrap Fluid Guard Ring



Polygon Fluid Guard Ring



Rectangle Fluid Guard Ring



If you have defined the minOppExtension technology rule in the technology file, Virtuoso uses the defined values to identify the distance on one side and the other side of the contacts placed in the corner of the FGR. For example, assume that your technology file contains the following definition:

```
( minOppExtension   "Oxide"  "Cont" (0.02 0.05) 'ref "CONT.E.2&3" 'description
"Minimum Oxide to Contact Enclosure" )
( minOppExtension   "Metal1"  "Cont" (0.02 0.05) 'ref "CONT.E.2&3" 'description
"Minimum Metal1 to Contact Enclosure" )
```

In this context, you can consider `0.02` as the minimum value and `0.05` as the maximum value. So, the W and L text boxes adjacent to the *Minimum Diffusion Over Contact* and *Minimum Metal Over Contact* fields can have either of these two values. If W has a value equal to or greater than the minimum value, but less than the maximum value, L should have a value greater than or equal to the maximum value, or vice versa. The following equation illustrates this effect of the minOppExtension technology rule on W and L values:

If $minOppExtension(layer\ contact)_{min} \leq W < minOppExtension(layer\ contact)_{max}$

then $L \geq minOppExtension(layer\ contact)_{max}$

And vice-versa for W and L.

By default, W will be populated with the minimum value (`0.02`) and L will be populated with the maximum value (`0.05`).

Customizing the Create Guard Ring Form

The following sections describe the ways of customizing the Create Guard Ring form:

- Creating a New Create Guard Ring Form (in IC6.1.8 and ICADVM20.1 releases)
- Modifying the Create Guard Ring Form (ICADVM20.1 Only)

Note: If you get errors and warnings when your third-party tool reads a customized FGR, use the solution described in Appendix D, “Loading VFO Infrastructure in Third-Party Tools”.

Creating a New Create Guard Ring Form

The Create Guard Ring form launched from the layout editor is an option-type form that displays the *Hide*, *Cancel*, and *Defaults* buttons. However, if you prefer to use a standard-type form with *OK*, *Cancel*, and *Apply* buttons, you have the flexibility to create such a Create Guard Ring form. The standard-type can be run from other Virtuoso applications, like Module Generator (Modgen) and Constraint Manager.

The following steps summarize the Create Guard Ring form creation process:

1. Define a new form pointer along with unique fields and form symbol using the `vfoGRNewCreateForm` SKILL function.

   ```
   vfoGRNewCreateForm (uniqueFormIdentifier formType @optional callbackList)
   ```

   Here,

`uniqueFormIdentifier` argument is an alphanumeric string that gives the name of the form to be used for generating the unique field names. If the specified form name already exists, the SKILL function returns `nil` and generates an error prompting you to create the form using another `uniqueFormIdentifier` value.

`formType` argument accepts only `'OKCancelApply` or `'HideCancelDef` as arguments that define whether the form should be of standard or options type, respectively. Currently, if you choose to create a standard-type form, you can create an FGR only in *Wrap* mode.

`callbackList` argument is optional. It can be used to list the names of callback procedures for *OK*, *Cancel*, and *Apply* buttons. The callback list should be provided in the following format:

```
callbackList = list (
    sprintf (nil "_defineCB ()")
)
```

2. Register the customization procedure using the `vfoGRRegCreateFormUpdateCallback` SKILL function, which has the following syntax:

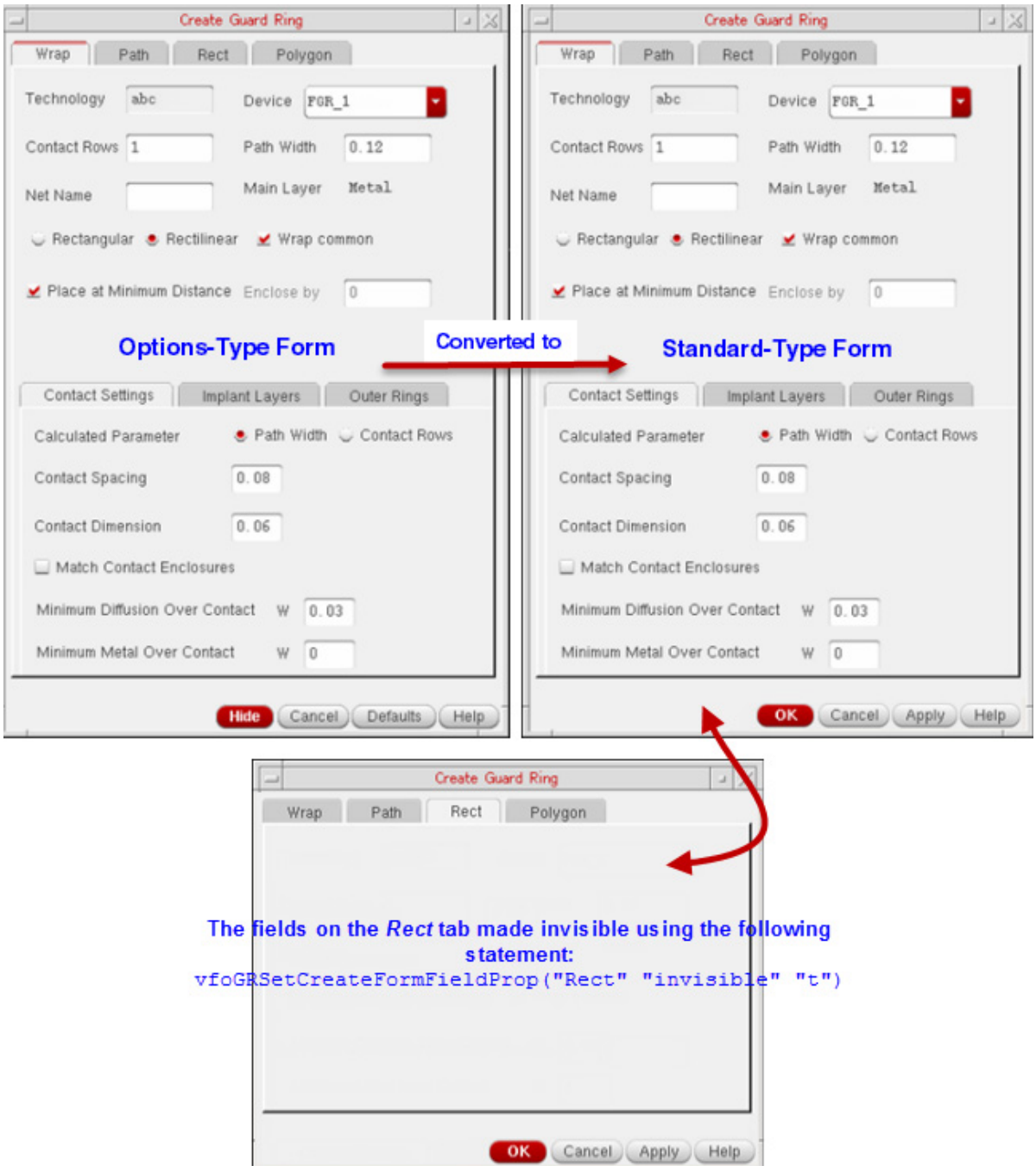
```
vfoGRRegCreateFormUpdateCallback(
     formPointer procedureName @optional callbackList
)
```

The corresponding procedure will be called for the particular form pointer after the `vfoGRUpdateCreateForm(form)` trigger is called.

The following is an example of the above steps:

```
; create the new Create Guard Ring form and store its form pointer
form_modgen = vfoGRNewCreateForm ("MODGEN" 'OKCancelApply)
; register the callback for the given form pointer
vfoGRRegCreateFormUpdateCallback (form_modgen "vfoCustomize_modgen")
; define the callback
procedure (vfoCustomize_modgen (formPointer
    ; write your own method body here related
    ; to Create Guard Ring form updates
)
hiDisplayForm (form_modgen)
```

The following figure visually depicts this type of customization of the Create Guard Ring form:



Modifying the Create Guard Ring Form (ICADVM20.1 Only)

Important

The feature described in this section is available only from **ICADV12.1 ISR5 onward**. It is not available in IC6.1.6.

The ability to modify the Create Guard Ring form provides you the control over the existing system-defined GUI components. It also enables you to add new user-defined GUI components, such as fields and buttons, and define callbacks for the specific ones you want to use on the form.

The process of modifying the Create Guard Ring form can involve the following actions that are performed using triggers and supporting SKILL functions:

- Adding New User-Defined GUI Components to the Form
- Updating the Existing GUI Components on the Form, for example:
  - Hide all existing fields that are displayed by default
  - Change the properties of the pre-defined (system) fields or components available on the form
- Updating Properties of User-Defined GUI Components

Detailed information about this process along with examples can be found in the following application notes available on the Cadence Online Support website:

- *Customizing Create Guard Ring Form*
- *Adding and Managing CDF Parameters for Fluid Guard Rings*

Adding New User-Defined GUI Components to the Form

To add new user-defined GUI components to the Create Guard Ring form, do the following:

1. Write a procedure to define the `vfoGRAddCreateFormFields` trigger.
2. Use the `hiCreate*` functions to add new GUI components on the Create Guard Ring form.
3. Set the GUI component to a specific location on the form.

   In the Create Guard Ring form, there are multiple global lists that enable you to add various form components. The common global list (also called common queue) defines the set of form components that are displayed on all four tabs (*Wrap*, *Path*, *Rect*, and

Polygon) of this form. For example, the GUI components, such as *Technology*, *Device*, and *Contact Rows*, that are common on all tabs exist in the common queue area. To add a new component in this area, use the `vfoGRAddFieldsInCommonQ` queue.

However, if you want to update the GUI components visible only on a specific tab of the Create Guard Ring form, use the following queues:

- `vfoGRAddFieldsInPathTabQ` (use for the *Path* tab)
- `vfoGRAddFieldsInRectTabQ` (use for the *Rect* tab)
- `vfoGRAddFieldsInPolygonTabQ` (use for the *Polygon* tab)
- `vfoGRAddFieldsInWrapTabQ` (use for the *Wrap* tab)

4. Use the `vfoGRSetExtraArgument` SKILL function to make the data available for processing by `extraArguments` in the FGR infrastructure that resides in Virtuoso.

   Each element of the associative list is a '*`key value`*' pair, where the `key` is the name of the FGR device parameter and `value` is the value associated to it, that is, `((<FGR_device_parameter_name> <value_of_GUI_component>) ...)`

Updating the Existing GUI Components on the Form

To update the GUI components that are currently displayed on the Create Guard Ring form, do the following:

1. Write a procedure to define the `vfoGRUpdateCreateForm(<formPointer>)` trigger, where *`formPointer`* is a pointer to the Create Guard Ring form.
2. Hide existing default fields from the form by using the `vfoGRSetCreateFormAllFieldsInVisible` SKILL function.

   Note: This SKILL function cannot be used to hide user-defined fields.

   You can use the `vfoGetImplementationClassName` SKILL function to identify the implementation class of different devices and hide the form fields only when the implementation class is not `vfoGuardRing`.
3. Reset the properties of the fields or components displayed on the form by using the `vfoGRSetCreateFormFieldProp` SKILL function. It supports the use of the following property values: `value`, `defValue`, `editable`, and `invisible`.

Updating Properties of User-Defined GUI Components

If you add a user-defined GUI component in the common queue, you can access its pointer using the `vfoGRGetCommonQPtr` SKILL function to get or set the properties of its component. For example, to set the property of a newly added form field, `UserSelectVertWidth`, you can write the following procedure:

```
procedure( setCreateFormUserFieldProp(actField promptString property value)
let(( newField )
    importSkillVar(vfoGRAddFieldsInCommonQ)
    evalstring( sprintf( nil "vfoGRGetCommonQPtr()->%s->%s = %s"
                UserSelectVertWidth->hiFieldSym property value )
    )
))
```

4

Editing Fluid Guard Rings

The new generation FGRs comprise fluid shapes. You can perform level-1 editing on these FGR instances.

To use the *Stretch*, *Quick Align, Reshape,* and *Split* commands, ensure that the *Fluid Shape* option is selected in the *Objects* assistant. The *Fluid Shape* option controls the selectability of the fluid shapes in FGRs for level-1 editing. If the *Stretch*, *Quick Align, Reshape*, or *Split* command is running and the *Fluid Shape* option is selected in the *Objects* assistant, the fluid shape can be selected and it highlights dynamically. If, however, the *Fluid Shape* option is not selected, the fluid shape cannot be selected; instead, the entire FGR instance is selected.

In addition, you can use the commands on *Edit – Fluid Pcell* menu for editing FGRs. By using these commands, you can perform the following edit operations on FGRs:

- Stretching a Fluid Guard Ring
- Aligning a Fluid Guard Ring
- Reshaping a Fluid Guard Ring
- Splitting a Fluid Guard Ring
- Chopping a Fluid Guard Ring
- Merging Fluid Guard Rings
- Converting a Fluid Guard Ring to a Polygon
- Creating a Tunnel Through a Fluid Guard Ring
- Healing a Fluid Guard Ring
- Cleaning Overlapped Contacts from Fluid Guard Rings

Tip

You can enable the *Guardring* toolbar from the *Window – Toolbars* menu and click the required editing icon.

Note: If editing an FGR comprising a path fluid shape changes the shape such that it cannot be maintained as a path, then the fluid shape of the FGR is converted to polygon. The behavior and resultant of editing a path-based FGR are different from that of editing a polygon-based FGR.

Stretching a Fluid Guard Ring

The FGR instances created using Wrap Tab, Path Tab, or Rect Tab modes result in guard rings with fluid shapes of the type path. You can stretch such fluid guard rings by using the center line or the vertex of the path.

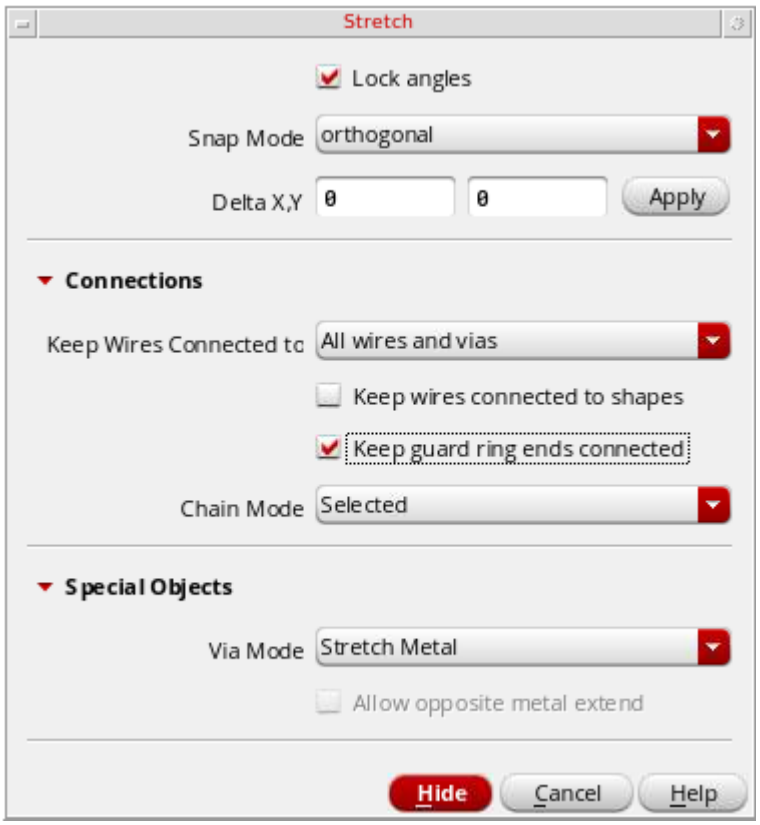
Stretching Path Type Fluid Shapes

You can stretch a path-based FGR in the same way you stretch a path object at level `0`. For more information about stretching and how it is done, refer to the *Stretching Objects* section in the Editing Objects chapter of the *Virtuoso Layout Suite L User Guide*.

You cannot stretch a path-based FGR along its edges because the width of a path cannot be variable along its length. To stretch a path-based FGR along the boundary edges to increase the number of contacts, first convert the path-based FGR to a polygon. For more information, see Converting a Fluid Guard Ring to a Polygon.

While stretching a rectangular FGR that has any of the edges connected to another FGR, select the *Keep Guard Ring Ends Connected* check box in the *Stretch* form. This enables

you to stretch the rectangular guard ring from all corners. For detailed information about the fields on this form, refer to the *Stretch Form* section.



Tip

Alternatively, you can use the `keepGuardRingEndsConnected` environment variable to control whether the FGRs with touching ends should stay connected during the *Stretch* command operation.

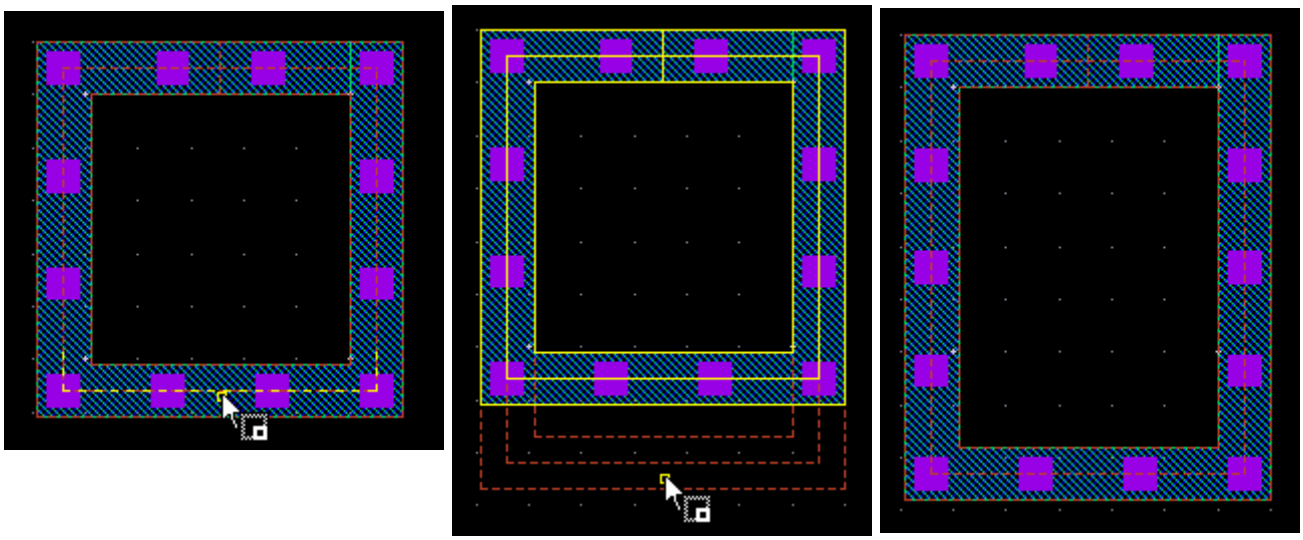
The *Stretch* command supports both pre-selection and post-selection of guard rings.

Stretching Polygon Type Fluid Shapes

You can stretch the edge of only those guard rings whose fluid shape is of the type polygon. While stretching the edge of such a guard ring, only the fluid shape is stretched. Individual shapes of a guard ring can neither be selected nor be stretched. You can stretch a guard ring with path/pathSeg fluid shape by stretching the centerline.

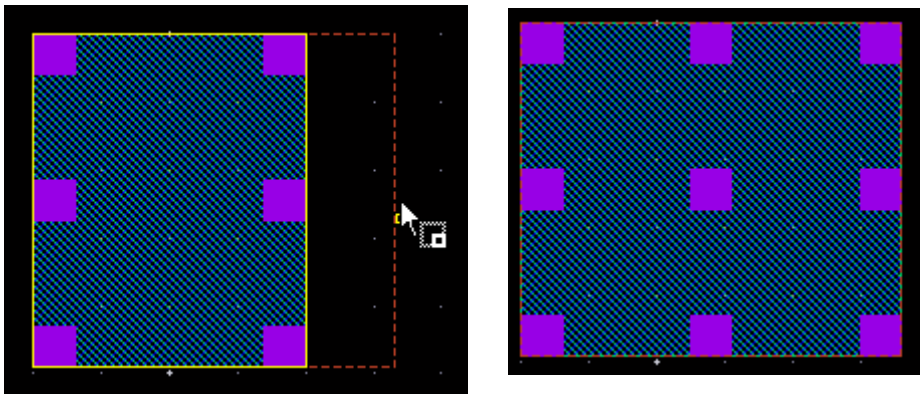
You can stretch a guard ring to modify:

- The size of a guard ring by using the path or pathSeg centerline stretch

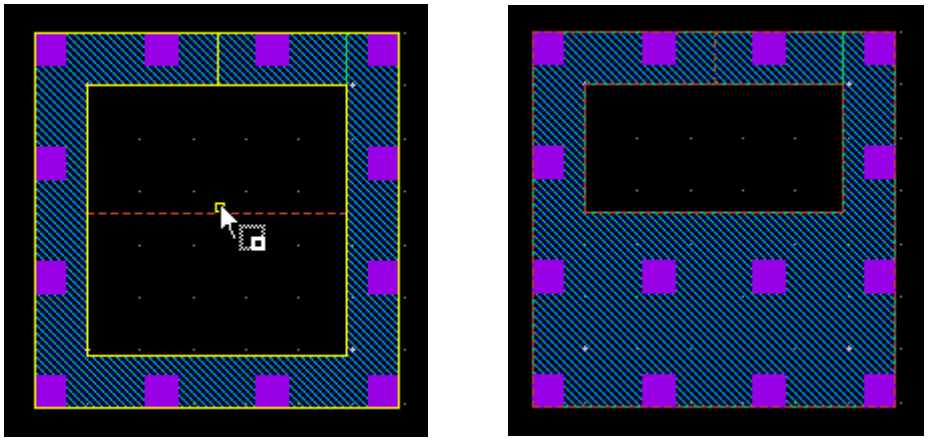


- The width of one side of a polygon guard ring or after converting the path or pathSeg guard ring to a polygon

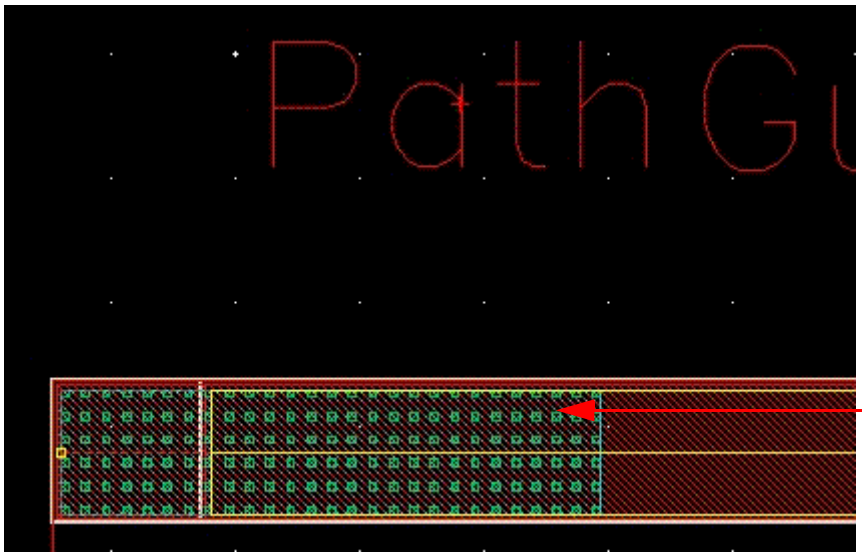
Stretching Guard Ring with Polygon Fluid Shape



Stretching a Fluid Guard Ring that is Converted to a Polygon



Design-rule-driven (DRD) rules are also supported during the FGR editing process (*Stretch* command). You can enable the *Enforce*, *Notify*, or *Post Edit* mode at the *current to bottom* hierarchy depth while editing an FGR. You can specify the DRD rules that need to apply through the *DRD Options* form accessed from the *DRD* toolbar or the *Options — DRD Edit* menu. For more information, see Creating Fluid Guard Rings.



Notification regarding violation of *minSpacing* rule during stretching a fluid guard ring

Aligning a Fluid Guard Ring

You can align only the fluid shape of a guard ring by using the *Edit – Quick Align* command. The fluid shape should be of the type path or polygon. In addition to aligning FGRs, you can use the smart snapping and target edge highlighting features of the *Quick Align* command

to stretch the fluid shape of a guard ring to a target object. Individual shapes of a guard ring cannot be selected or aligned.

For more information about aligning and how it is done, refer to the following sections in the *Virtuoso Layout Suite L User Guide*:

- *Aligning Objects* section in the Editing Objects chapter
- *Quick Align Form* section in the Layout L Forms appendix

Reshaping a Fluid Guard Ring

You can reshape only the fluid shape of a guard ring. The *Reshape* command alters the fluid shape points and re-generates the underlying guard ring based on the new points. Individual shapes of a guard ring can neither be selected nor be reshaped.

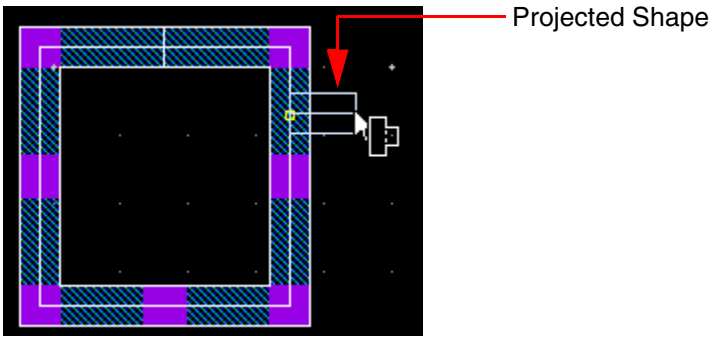
Reshaping a guard ring that is created from a path or polygon fluid shape is similar to reshaping an equivalent shape (path or polygon) in the design. For more information about reshaping and how it is done, refer to the Reshaping Objects section in the *Editing Objects* chapter of the *Virtuoso Layout Suite L User Guide*.

The pointer snapping follows the Gravity Controls settings in the *Layout Editor Options Form*. If *Gravity On* is selected and *Depth* is greater than `0`, the pointer snaps to objects in the vicinity of the pointer based on the *Types* selected and the specified *Aperture* value. If *Gravity On* is not selected, smart snapping is the default pointer snapping mode. The pointer snaps to the highlighted edge when you click.

To reshape a guard ring with a path or pathSeg fluid shape:

1. Choose *Edit – Advanced – Reshape* or click the *Reshape* icon on the *Guardring* toolbar.
2. Select the guard ring you want to reshape.

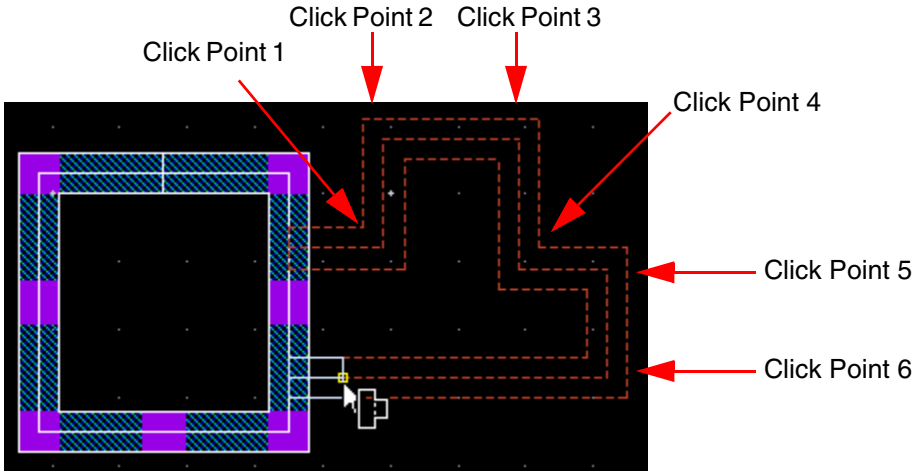
   If you move the pointer within the *Aperture* distance (set in the *Layout Editor Options Form*) from the selected shape and click, a projected shape appears from the centerline of the path or pathSeg, as shown below.



3. Click to start the reshape from the required starting point.

   The projected shape anchors to the guard ring at the click point.

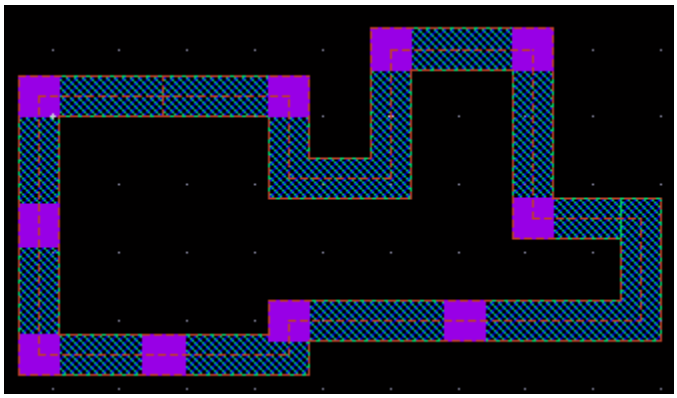
4. Click to define the new points of the reshaped figure, as shown below.



While defining the new points of the reshaped figure, if you bring the pointer within *Aperture* distance from the selected shape, the projected shape appears. You can choose to finish the reshaping here by pressing the `Spacebar` key or continue to click to define more points of the reshaped figure.

5. Click to add the highlighted shape to the original selected shape.

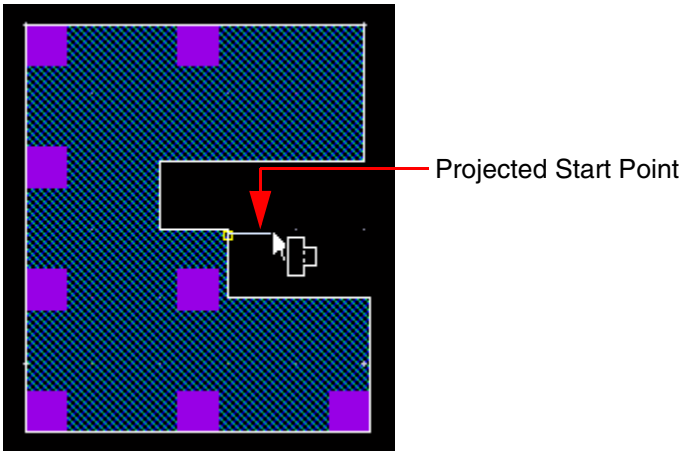
The original guard ring is reshaped based on the specified new points.



To reshape a guard ring with a polygon fluid shape:

1. Choose *Edit – Advanced – Reshape* or click the *Reshape* icon on the *Guardring* toolbar.
2. Select the guard ring you want to reshape.

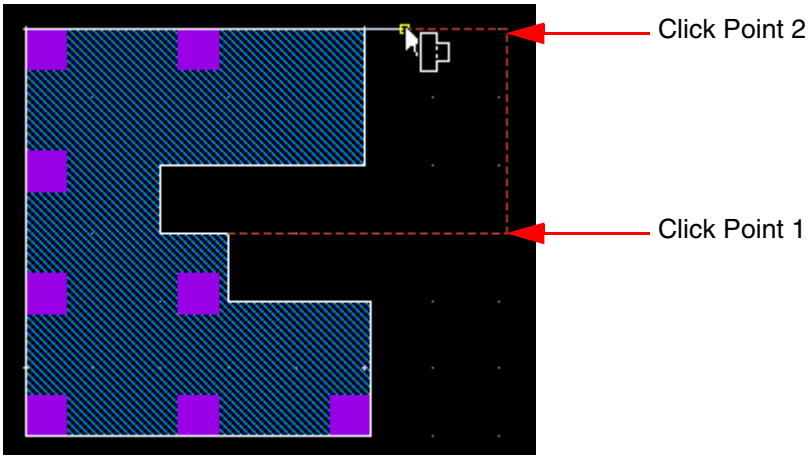
If you move the pointer within the *Aperture* distance (set in the *Layout Editor Options Form*) from the selected shape, a projected start point appears from the edge of the polygon fluid shape, as shown below.



3. Click to start the reshape from the projected start point.

   If required, you can press `F3` and change the *Snap Mode* in the *Reshape Form*.

4. Click to define the new points of the reshaped figure, as shown below.



While defining the new points of the reshaped figure, if you bring the pointer within *Aperture* distance from the selected shape, the projected end point appears. You can choose to finish the reshaping here by pressing the `Spacebar` key or continue to click to define more points of the reshaped figure.

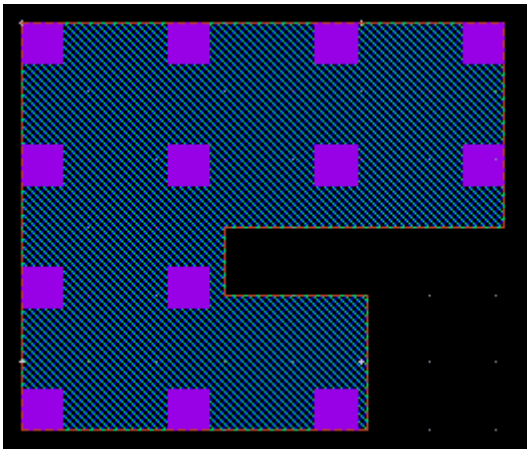
5. Press the `Spacebar` key when the projected end point is visible.

   This enables the smart snapping mode.

6. Click to add the highlighted shape to the original selected shape.

7. Click the middle-mouse button to toggle between the versions of the possible reshaped figures.
8. Click to select the highlighted version.

   The original guard ring is reshaped based on the specified new points.



Splitting a Fluid Guard Ring

You can split only the fluid shape of an FGR. The fluid shape can be of the type path or polygon. The *Split* command alters the fluid shape points and re-generates the underlying guard ring based on the new points. Individual shapes of a guard ring can neither be selected nor be split.

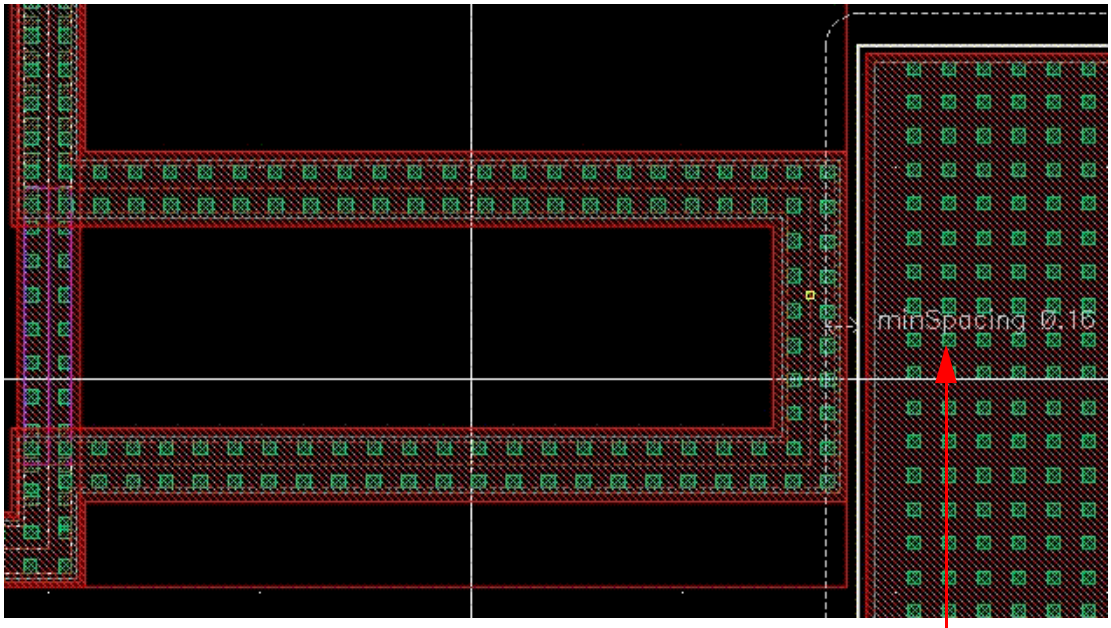
Splitting a guard ring that comprises a path or polygon fluid shape is similar to splitting an equivalent shape (path, rectangle, or polygon) in the design.

The steps to split the fluid shape of an FGR are the same from ***step 3*** onwards described in the Splitting Objects section in the *Editing Objects* chapter of the *Virtuoso Layout Suite L User Guide*. The difference is in the sequence of the first and second step. So, while splitting fluid shapes, first choose *Edit – Advanced – Split* [or press `Ctrl+s`], and if you want, open the *Split* form by pressing `F3`. Then, select the fluid shape of the FGR, and not the complete FGR instance.

Caution

The Layout Editor infrastructure does not support splitting the instances.

The *Split* command also supports the DRD rules, as illustrated below.



Notification regarding violation of *minSpacing* rule during splitting a fluid guard ring

Chopping a Fluid Guard Ring

The *Edit – Fluid Pcell – Chop* command enables you to remove a part of an FGR or cut it into pieces. You can chop a guard ring in both pre-select and post-select modes and even when no object is selected. The *Chop* command cuts across all the layers of the FGR device. If the result of chopping an FGR divides it into multiple distinct parts, then a new FGR device gets created corresponding to each of the respective parts. The *Chop* command automatically converts an FGR to a polygon.

Note: When you run the *Chop* command, any existing labels get removed from the instance because this operation leads to creation of a new instance.

This section covers the following:

- Types of Chop Shapes
- Chopping a Fluid Guard Ring Without Selecting It
- Pre-Selecting a Fluid Guard Ring to be Chopped
- Post-Selecting a Fluid Guard Ring to be Chopped

Types of Chop Shapes

The Chop Fluid Objects Form displayed using the *Edit – Fluid Pcell – Chop* menu enables you to use the following types of shapes to chop a fluid guard ring:

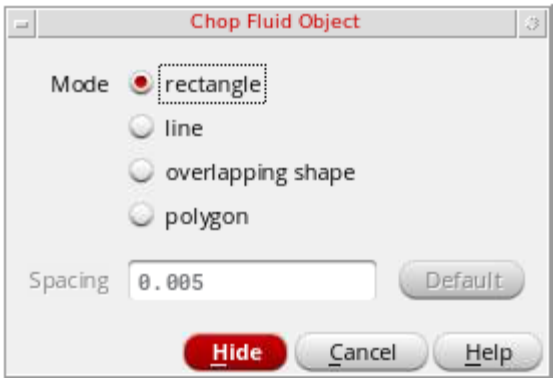
- *Rectangle*
- *Line*
- *Overlapping shape*
- *Polygon*

Note: The default mode is *rectangle*.

Using Rectangle as the Chop Shape

When you select *rectangle* mode on the Chop Fluid Object form, do the following:

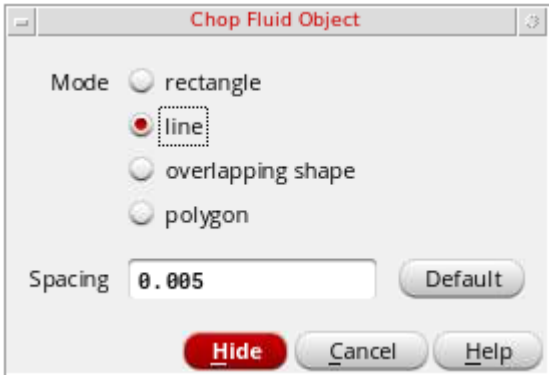
➥ Click to define the opposite corners of the chop rectangle that intersects the guard ring in the region that needs to be chopped.



Using Line as the Chop Shape

When you select *line* mode on the *Chop Fluid Object* form, do the following:

a. Specify the *Spacing* value.



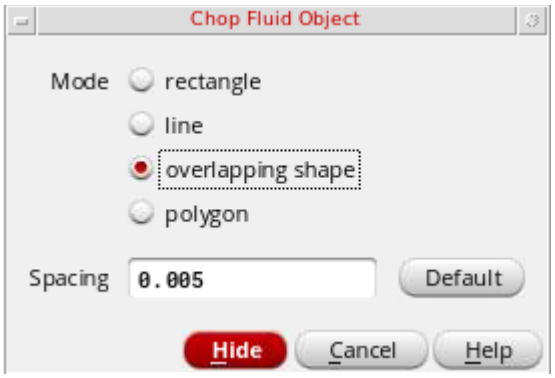
b. Create a line that intersects the guard ring in the region that needs to be chopped.

c. Double-click or press `Enter` to complete drawing the line.

Using Overlapping Shape as the Chop Shape

When you select *overlapping shape* mode on the Chop Fluid Object form, do the following:

a. Specify the *Spacing* value.

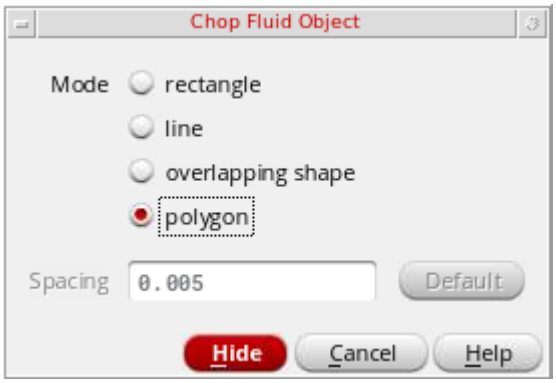


b. Click the shape that overlaps the guard ring to be chopped.

Using Polygon as the Chop Shape

When you select to *polygon* mode on the Chop Fluid Object form, do the following:

➡ Click to define the points of the chop polygon that intersects the guard ring in the region that needs to be chopped.



Chopping a Fluid Guard Ring Without Selecting It

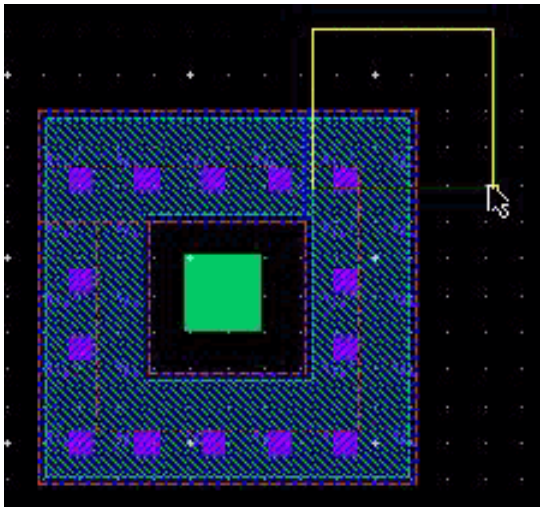
To chop a guard ring without selecting it first:

1. Choose *Edit – Fluid Pcell – Chop*.
2. Press `F3`.

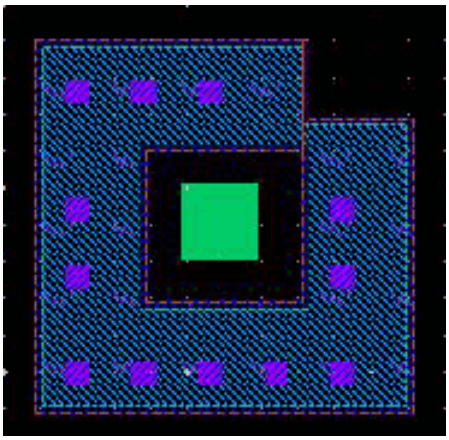
   The Chop Fluid Objects Form appears.
3. Select a *Mode* and update the *Spacing* value as required, if applicable.

4. Create the chop shape to intersect the guard ring or click the shape overlapping the guard ring.

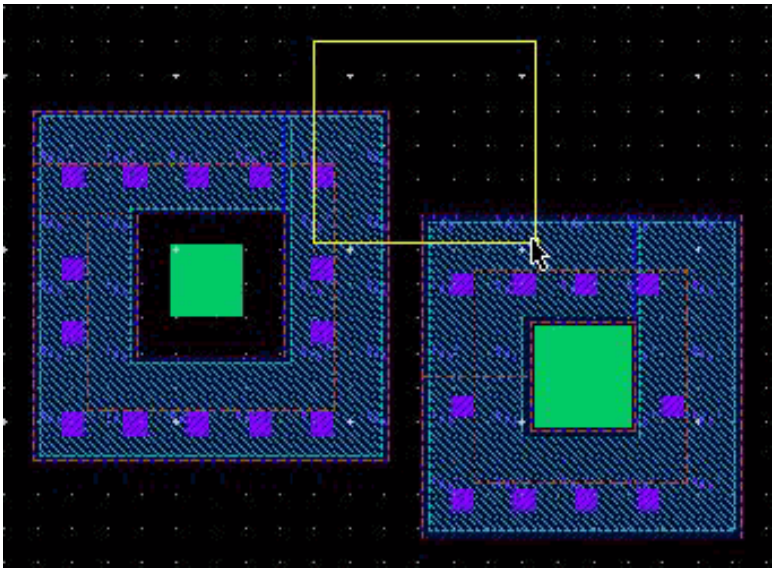
All the guard rings that are intersected by the chop points are either split or a part is cut away. This is illustrated in the figures below.



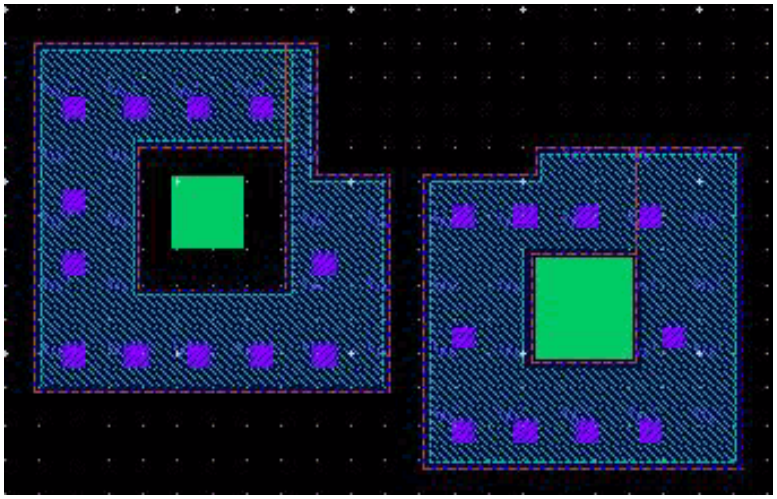
Defining the rectangle chop shape.



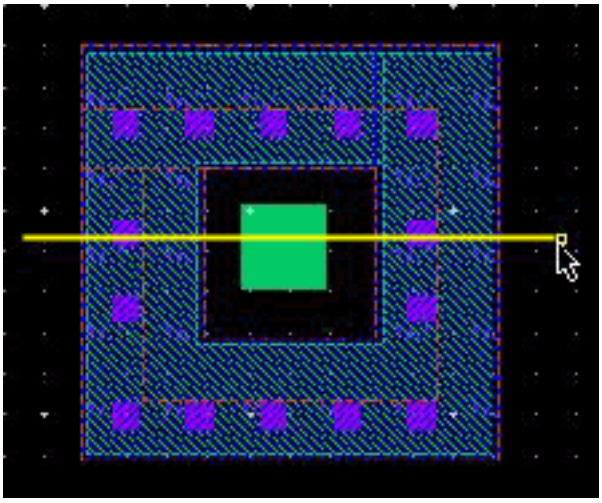
Part of the guard ring is cut away.



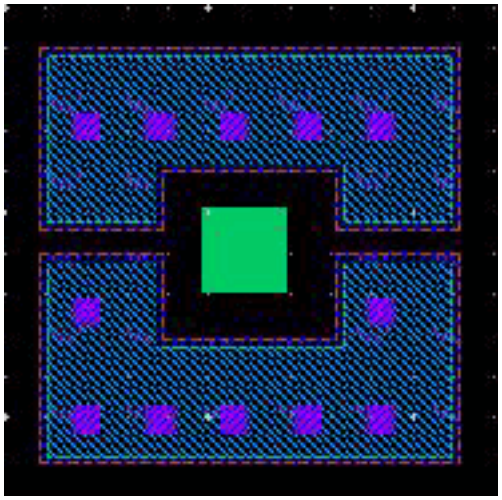
Defining the rectangle chop shape over multiple guard rings.



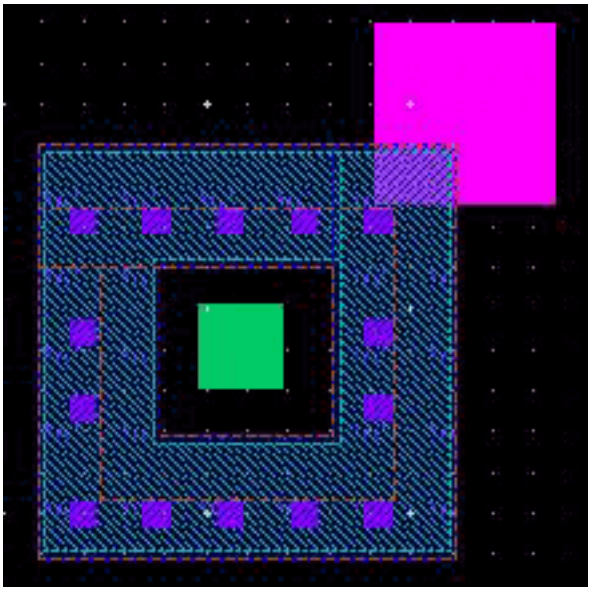
Part of both the guard rings is cut away.



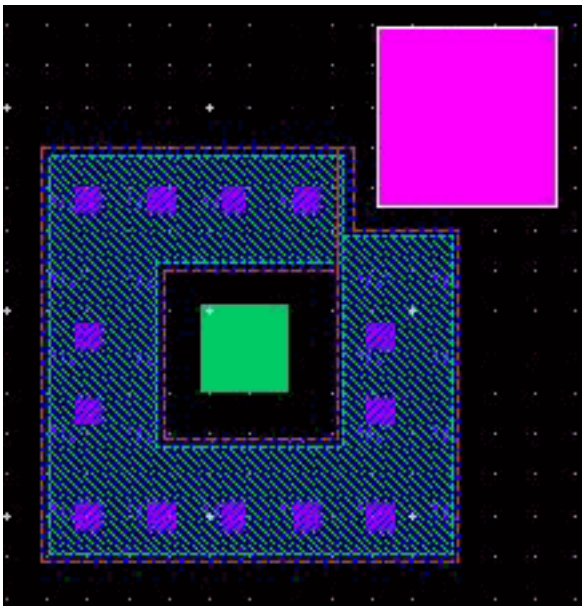
Defining the line chop shape.



The guard ring is split into two.



Click the shape overlapping the guard ring.



The guard ring chops around the clicked shape.

If a shape overlaps multiple guard rings and no guard ring is selected, all the guard rings are chopped if *overlapping shape* is the selected *Mode*.

5. Press `Esc` or click *Cancel* in the form to complete chopping guard rings.

Pre-Selecting a Fluid Guard Ring to be Chopped

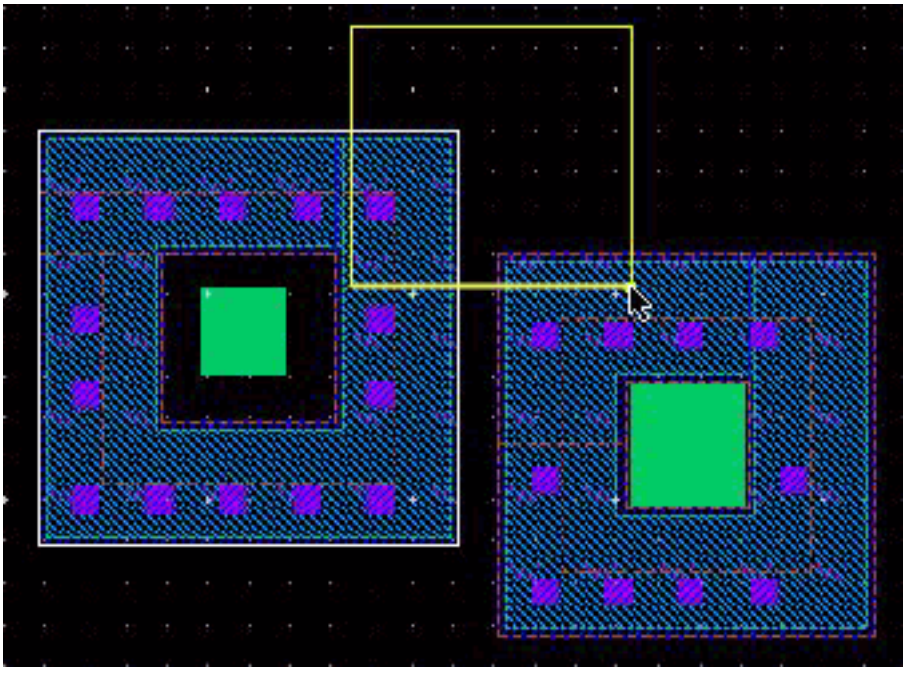
If the chop shape is drawn such that it intersects multiple guard rings, only the pre-selected guard ring gets chopped.

To chop a selected guard ring:

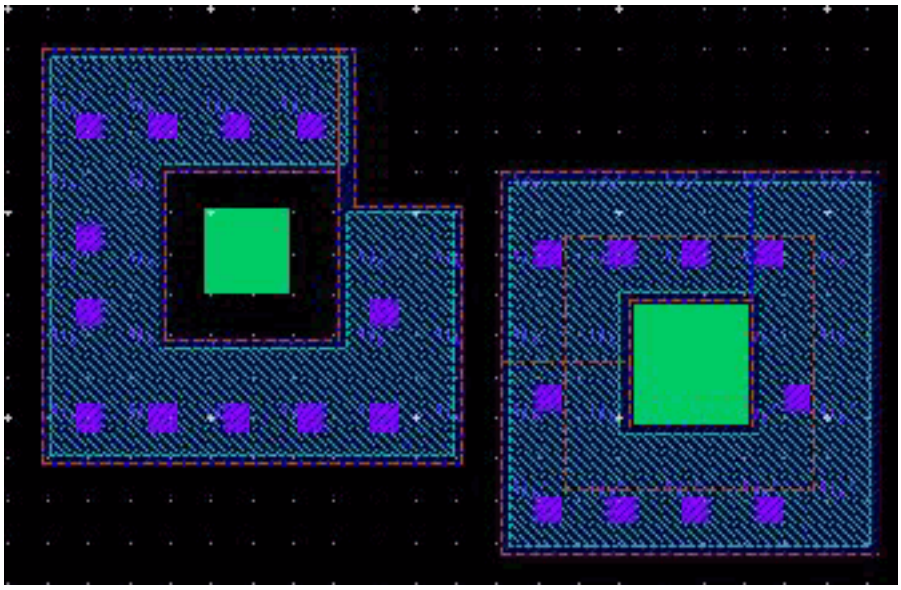
1. Select the guard ring you want to chop.
2. Choose *Edit – Fluid Pcell – Chop*.
3. Press `F3`.

   The Chop Fluid Objects Form appears.
4. Select a *Mode* and update the *Spacing* value as required, if applicable.
5. Create the chop shape to intersect the selected guard ring or click the shape overlapping the selected guard ring.

The selected guard ring is either split or a part of the guard ring is cut away. This is illustrated in the figures below.

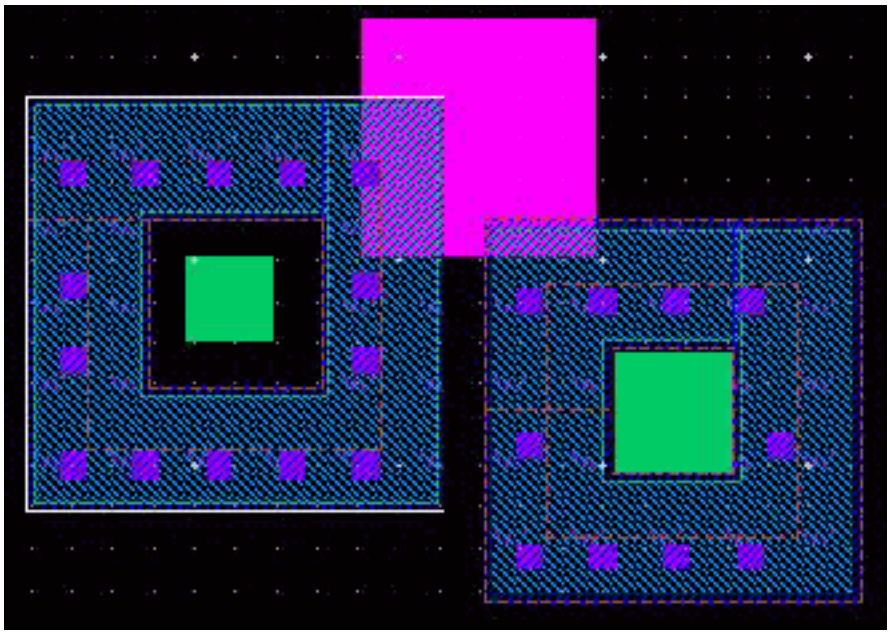


Defining the rectangle chop shape over multiple guard rings. The guard ring on the left is selected.

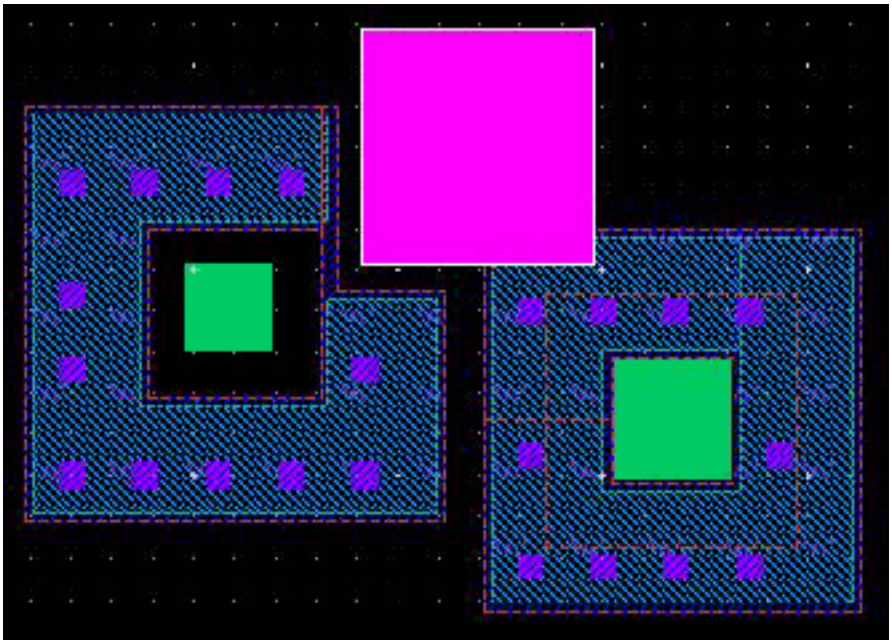


Part of only the selected guard ring is cut away.

When *overlapping shape* is the selected *Mode* and a shape overlaps multiple guard rings, only the selected guard ring is chopped. This is shown in the figure below.



The shape overlaps both the guard rings. However, only the guard ring on the left is selected.



Only the selected guard ring is chopped by the overlapping shape.

6. Press `Esc` or click *Cancel* in the form to complete chopping guard rings.

Post-Selecting a Fluid Guard Ring to be Chopped

To chop a post-selected guard ring:

1. Choose *Edit – Fluid Pcell – Chop*.
2. Press `F3`.

   The Chop Fluid Objects Form appears.
3. Select a *Mode* and update the *Spacing* value as required, if applicable.
4. Select a guard ring either by pressing `Shift` and clicking the guard ring or by pressing `Shift` and area selecting the guard ring. You can add more guard rings to the selection by using either key combination.
5. Create the chop shape to intersect the selected guard ring or click the shape overlapping the selected guard ring.

   The selected guard ring is either split or a part of the guard ring is cut away.
6. Press `Esc` or click *Cancel* in the form to complete chopping guard rings.

Merging Fluid Guard Rings

When you merge certain FGRs, a new FGR device gets created. The FGRs to be merged should meet the following criteria:

- The FGRs must be overlapping.
- The FGRs belong to the same device class.
- The FGRs belong to the same net or only one of the FGRs is associated with a net.

A merged FGR device has the following noteable characteristics:

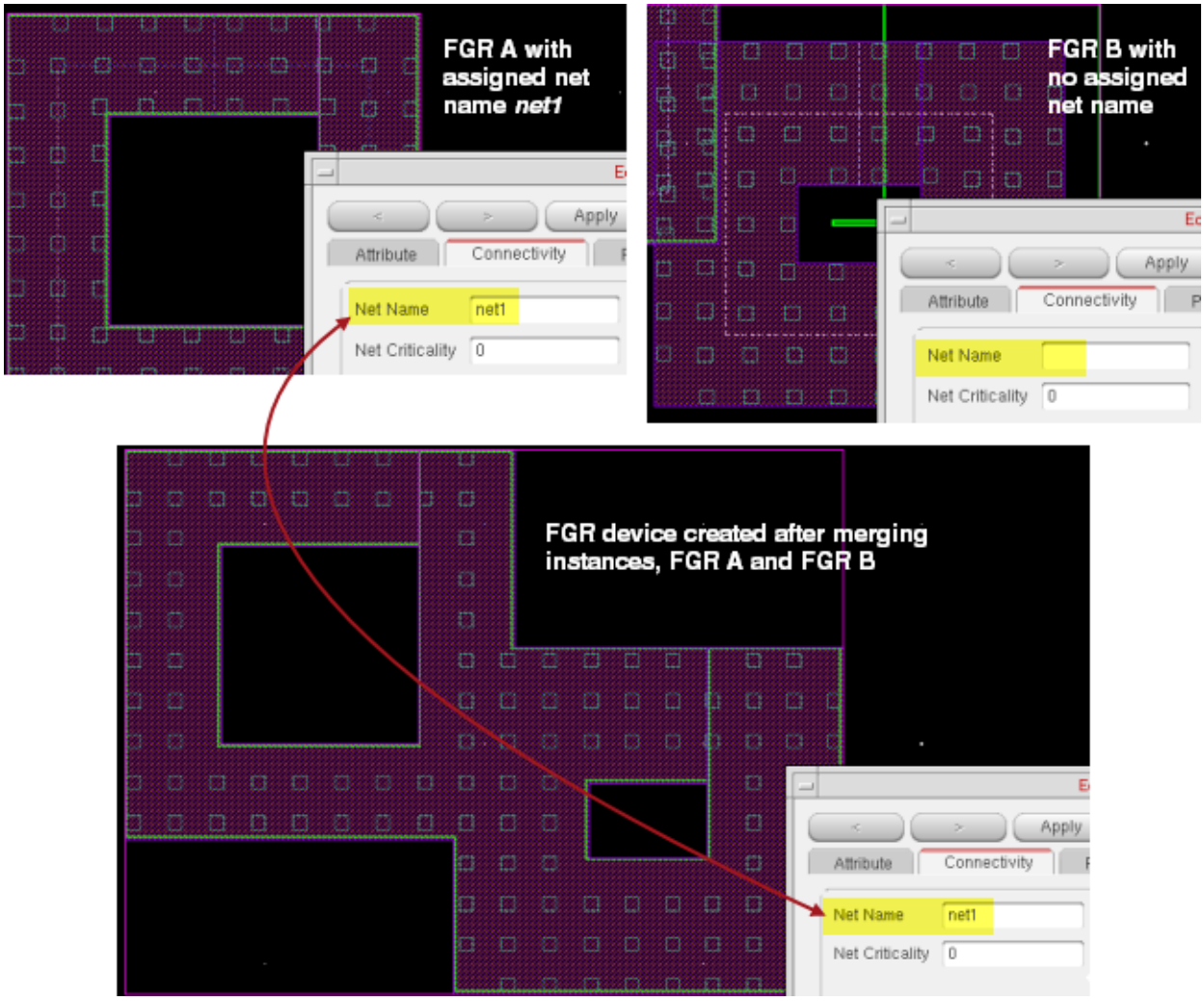
- Any tunnels in the original FGRs are retained in the merged FGR as well. For more information about creating tunnels, see Creating a Tunnel Through a Fluid Guard Ring.
- Any existing labels get removed from the instance because the merge operation leads to creation of a new instance.
- Net name assigned to an FGR instance is inherited by the merged FGR device, if one of the FGR instance has net name.

  The net name can be defined at the time of creation in the Create Guard Ring form using one of the following methods:

  - The *Net Name* field on all tabs.
  - The *Net Name* field on the *Outer Rings* subtab of the Wrap and Rect tabs.

  Otherwise, you can define the *Net Name* on the *Connectivity* tab of the Edit Instance Properties form while editing an FGR instance.

For example, if instance *FGR A* having net name *net1* assigned to it is merged with another instance *FGR B* that does not have a net name, the merged FGR device inherits the net name of the first instance as shown in the figure below.



FGR instances can be merged in pre-select and post-select modes as explained in the following sections:

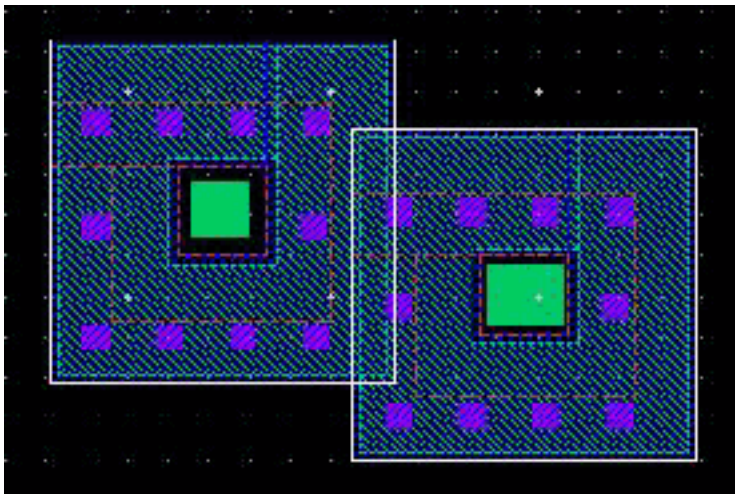
- Pre-Selecting Fluid Guard Rings to be Merged
- Post-Selecting Fluid Guard Rings to be Merged

For information about the inheritance of parameters in the merged instances, see Merging Fluid Guard Rings with Unmatched Parameters.

Pre-Selecting Fluid Guard Rings to be Merged

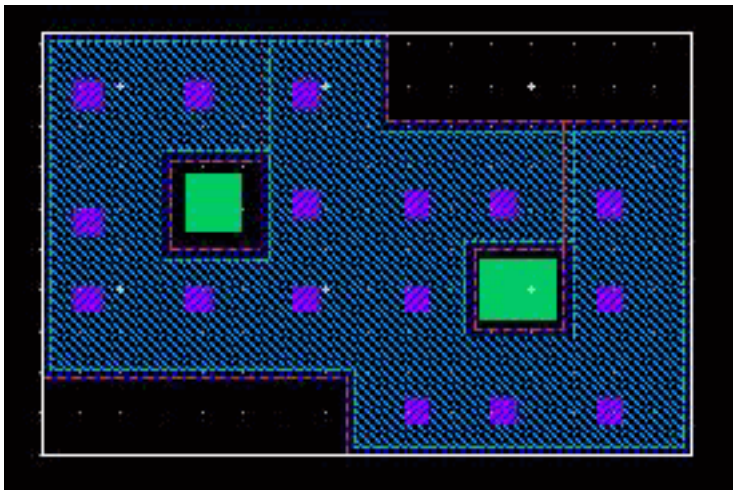
To merge the selected FGRs:

1. Select the overlapping FGRs you want to merge.



2. Choose *Edit – Fluid Pcell – Merge*.

   The selected FGRs are merged.



3. Press `Esc` to finish merging guard rings.

Post-Selecting Fluid Guard Rings to be Merged

To merge post-selected overlapping FGRs:

1. Choose *Edit – Fluid Pcell – Merge*.
2. Click the first guard ring.
3. Click the next guard ring that is overlapping the first one.

   The two guard rings are merged.
4. If required, continue to select any more overlapping FGRs.

   All the overlapping guard rings are merged if they meet the merge criteria.

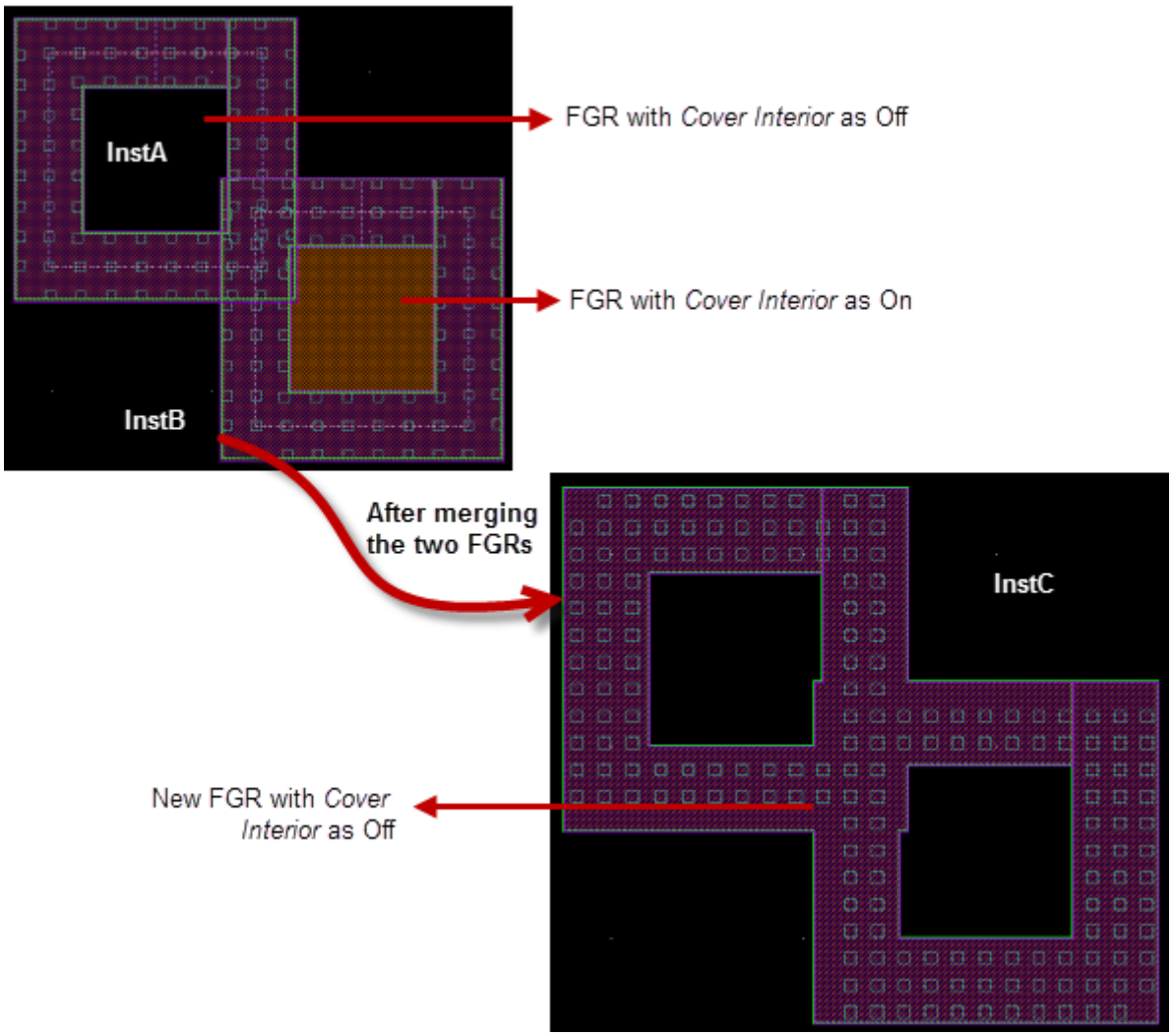
   Alternatively, after starting the *Merge* command, you can also area select the entire or part of a design containing guard rings. Then, click anywhere in the design or press `Enter` to run the *Merge* command. All the selected, overlapping guard rings that satisfy the merge criteria are merged.
5. Press `Esc` to finish merging guard rings.

Merging Fluid Guard Rings with Unmatched Parameters

While merging FGR instances with unmatched parameters, the merge command uses one FGR instance as the master.

While identifying which FGR instance should be considered as the master, Virtuoso considers the one that is nearest to the origin. All parameters on the merged FGR instances

are inherited from that of the master FGR instance. This can be understood from the following figure:



In this figure, `instA` and `instB` have been merged to create `instC`. The settings of the *Cover Interior* parameter for `instC` has been inherited from `instA`, which had this parameter set to off. This is because the origin of `instA` is before `instB`.

Converting a Fluid Guard Ring to a Polygon

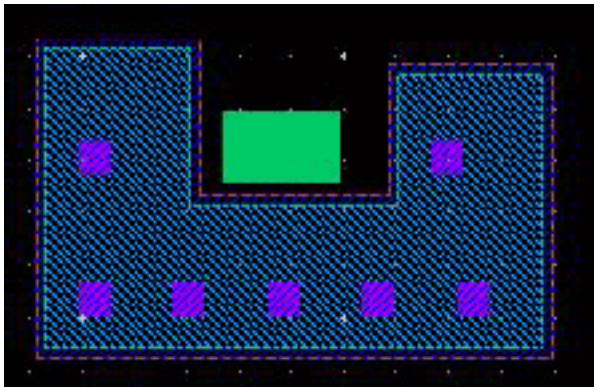
The *Convert to Polygon* command is useful for converting a path-based guard ring to a polygon guard ring. A path cannot have different widths along its length. You can convert a path-based guard ring to a polygon to achieve the desired editing results on the guard ring. Examples of such editing, which cannot be achieved with a path-based guard ring, include:

- Stretching a guard ring such that the number of contact rows is different along the various edges of the guard ring
- Reshaping part of a guard ring edge so that it has a different width

You can run the *Convert to Polygon* command in both pre-select and post-select modes.

To convert a path guard ring to a polygon:

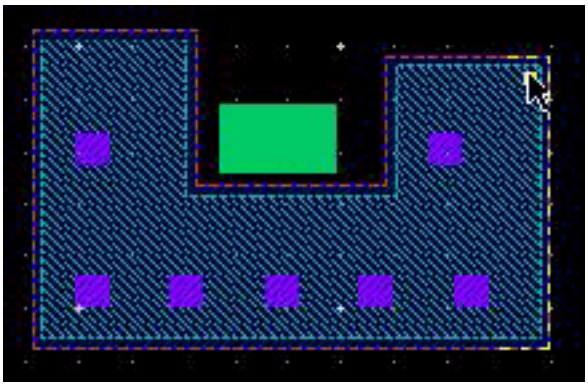
1. Select the path guard ring.



Path Guard Ring

2. Choose *Edit – Fluid Pcell – Convert to Polygon*.

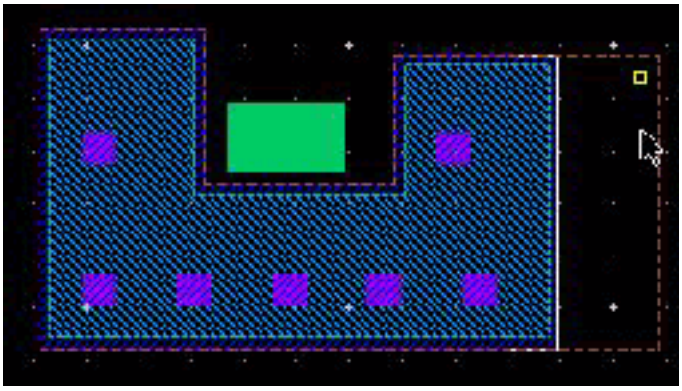
   The path guard ring is converted to a polygon guard ring.



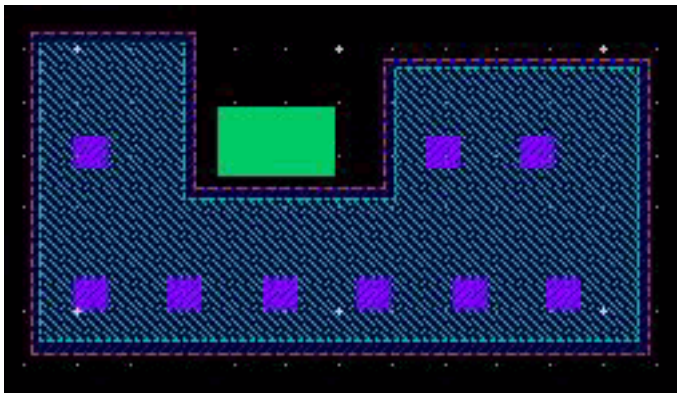
Path Guard Ring
Converted to a Polygon
Guard Ring

3. Press `Esc` to finish converting path guard rings to polygon guard rings.

You can now stretch or reshape the edges of the polygon guard ring, independent of the other edges and width of the remaining polygon. This kind of editing is not possible with a path-based guard ring.



Partial Selection of an Edge of the Polygon Guard Ring



Stretched Edge of the Polygon Guard Ring

Creating a Tunnel Through a Fluid Guard Ring

A tunnel is used to cut through a specific layer-purpose of a guard ring device. The *Tunnel* command supports both pre-selection and post-selection of guard rings.

You can remove existing tunnels from one or more guard rings by using the *Heal* command. For more information about the *Heal* command, see Healing a Fluid Guard Ring.

To create a tunnel through a guard ring:

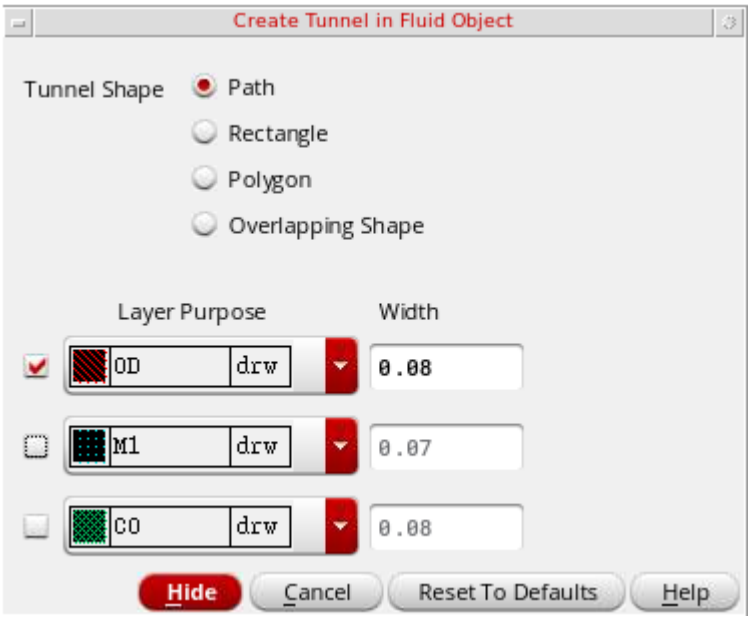
1. Select a guard ring in which you want to create a tunnel.
2. Choose *Edit – Fluid Pcell – Tunnel*.
3. Press `F3`.

   The Create Tunnel in Fluid Object Form appears.
4. Select a *Tunnel Shape*.

   The form updates to display all the layer-purposes that comprise the selected guard ring. The layer-purpose pairs are listed in the *Layer Purpose* column and the default minimum spacing values of the respective layer-purpose pairs are displayed in the *Spacing* column.

   Note: If you open the *Create Tunnel in Fluid Object* form without selecting a guard ring, the *Layer Purpose* and *Width* (or *Margin*) columns are not displayed on the form in *Path*, *Rectangle*, and *Polygon* modes and in *Overlapping Shape* mode when the *Use Layer-Purpose of the Shape* check box is not selected.

The figure below shows the form when *Path* is the selected *Tunnel Shape*. For *Path*, an additional *Width* column displays the minimum width values of the respective layer-purpose pairs.



The *Overlapping Shape* shows the *Layer Purpose* and *Spacing* columns if the *Use Layer-Purpose of the Shape* check box is not selected.

5. Against each *Layer Purpose* row, select the check box for all the layer-purposes through which you want to create the tunnel.
6. For each selected layer-purpose, if required, change the default value in one of the following (whichever is display based on the selected mode): *Width*, *Margin*, or *Spacing*.

   The *Width*, *Margin*, and *Spacing* fields for a *Layer Purpose* are non-editable if the corresponding check box is not selected.
7. Click to define the points of a path, rectangle, or polygon that intersect the guard ring or click a shape overlapping the guard ring.

   Note: You can create a tunnel in a guard ring without selecting the guard ring by using the *Overlapping Shape* as the Tunnel Shape when the *Use Layer-Purpose of the Shape* check box is selected.

   A tunnel is created in the specified area. The area of the tunnel created is determined by the Tunnel Shape you select.
8. Press `Esc` or click *Cancel* in the form to finish creating tunnels.

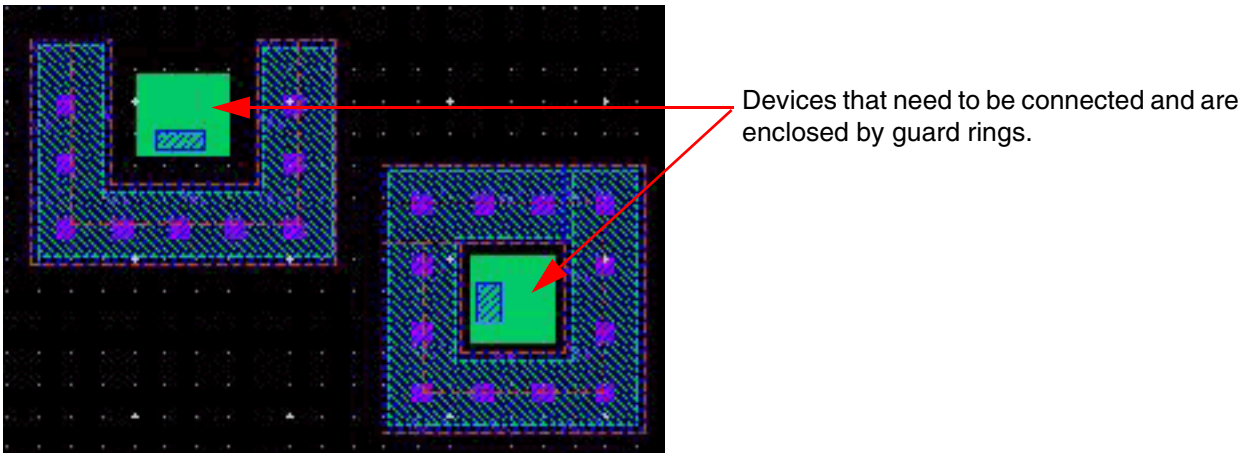
For information about creating tunnels by using the different tunnel shapes, see the following:

- Creating Tunnel By Using the Path Shape
- Creating Tunnel By Using an Overlapping Shape
- Creating Tunnel By Using Rectangle and Polygon Shapes

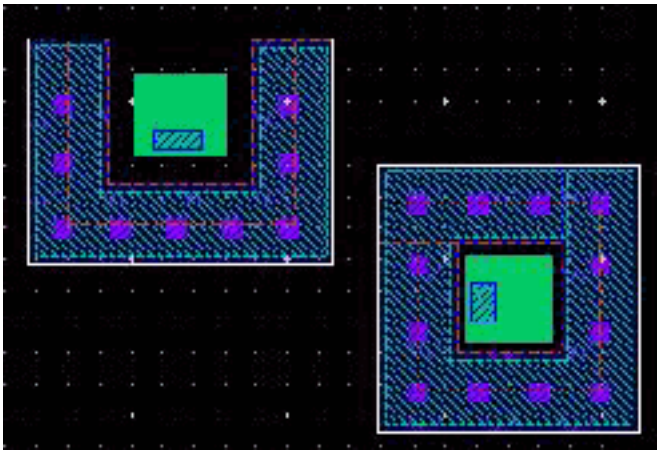
Creating Tunnel By Using the Path Shape

In the *Path* mode, you can pre-select or post-select the guard ring. If you do not select any guard ring before starting the command, you are prompted to first select a guard ring and then draw the path points. Tunnels are created through all the selected guard rings that are overlapped by the path you draw.

Consider two devices that you need to connect, each of which is enclosed within a guard ring. In this scenario, you can use the *Path* as the *Tunnel Shape* to first create tunnels in both the guard rings. Next, place a path to connect the two devices, passing through the tunnel region in the guard rings of the respective devices. This is illustrated below.



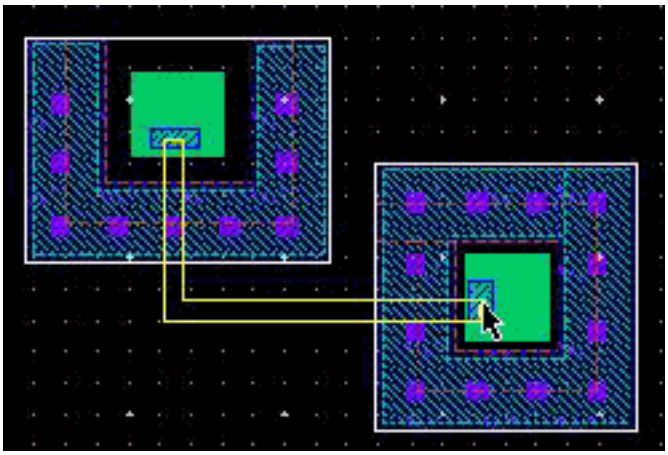
1. Select the guard rings.



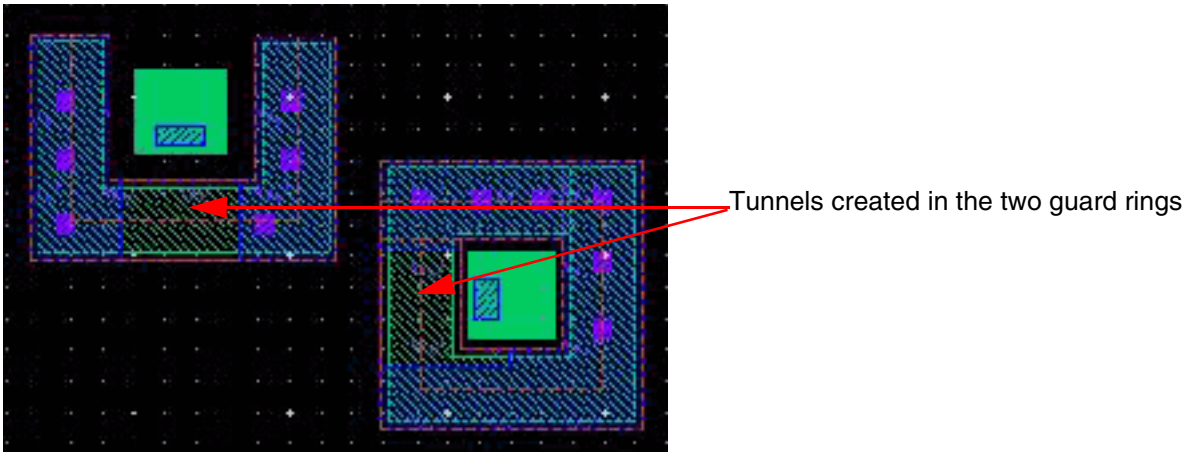
2. Choose *Edit – Fluid Pcell – Tunnel*.
3. Press `F3`.

   The Create Tunnel in Fluid Object Form opens.
4. Select *Path*.

   All the layer-purposes comprising the two guard rings are listed in the *Layer Purpose* column.
5. Select the guard ring layers through which you want to create the tunnel.
6. Update the *Width* column values, if required.
7. Draw a path intersecting the two guard rings, as shown below.

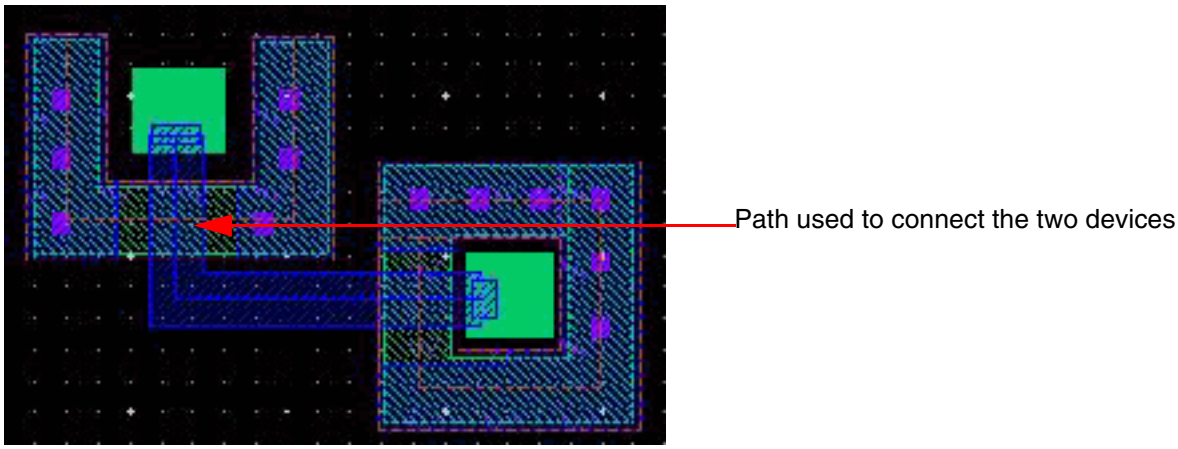


Tunnels are created in both the guard rings through the selected layers and in the region where the drawn path overlapped the selected guard rings, as shown below



8. Press `Esc` to finish creating the tunnel.

You can now create a path passing through the created tunnel regions in the two guard rings and connect the two devices.



Creating Tunnel By Using an Overlapping Shape

You can pre-select or post-select a guard ring while using an *Overlapping Shape* to create a tunnel. In this mode, you can create tunnels even when no guard rings are selected. Tunnels are created in all the guard rings that are overlapped by the shape. If one or more guard rings are selected, tunnels are created through only the selected guard rings that are overlapped by the shape, even if the shape overlaps other guard rings.

Important

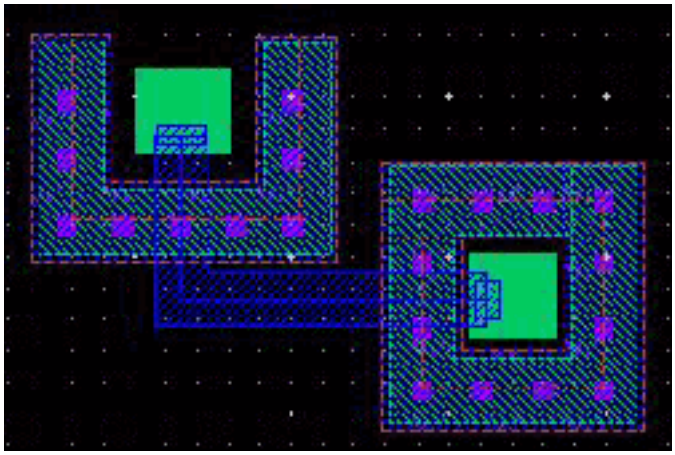
An overlapping shape cannot be used to create a tunnel if the overlapping shape layer is:

- `poly`
- Does not match the selected guard ring layers

In the scenario described in the Creating Tunnel By Using the Path Shape section, you can achieve the same result by drawing an overlapping shape that connects the two devices and create tunnel through both the guard rings by using the overlapping shape.

1. Create a path that connects the two devices.

   You can create the path on the same layer-purpose that you want to be removed from the guard ring.



   Notice that no guard ring is selected.

2. Choose *Edit – Fluid Pcell – Tunnel*.
3. Press `F3`.

   The Create Tunnel in Fluid Object Form opens.

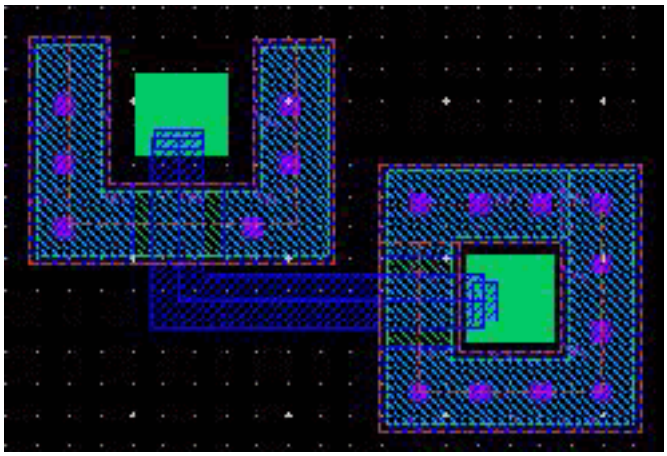
4. Select *Overlapping Shape*.
5. Select the *Use Layer-Purpose of the Shape* check box.
6. Use the *Minimum* spacing or specify a value in the *User Defined* field.

The *Minimum* spacing for a polygon type guard ring is calculated based on the `minSpacing` table rule specified in the technology file corresponding to the layer of the overlapping shape.

The *Minimum* spacing for a path type guard ring is calculated based on the two-dimensional `minSpacing` table rule with `W:L` calculated as, maximum width: width of the shape of the guard ring path on the same layer as that of overlapping shape. For a path-type guard ring, if only a one-dimensional `minSpacing` table rule is defined in the technology file or the one-dimensional `minSpacing` table rule takes precedence over the two-dimensional `minSpacing` table rule, then the *Minimum* spacing is calculated based on maximum width specified in the one-dimensional `minSpacing` table rule.

7. Click at the path.

   The path serves as the overlapping shape. Tunnels are created in the regions where the path overlaps the two guard rings.



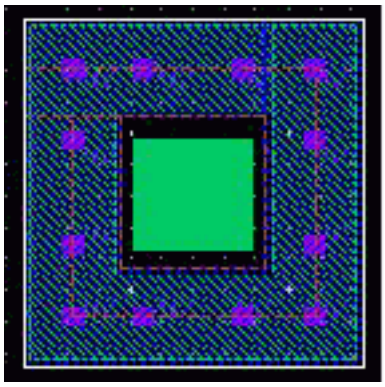
8. Press `Esc` to finish creating the tunnel.

Creating Tunnel By Using Rectangle and Polygon Shapes

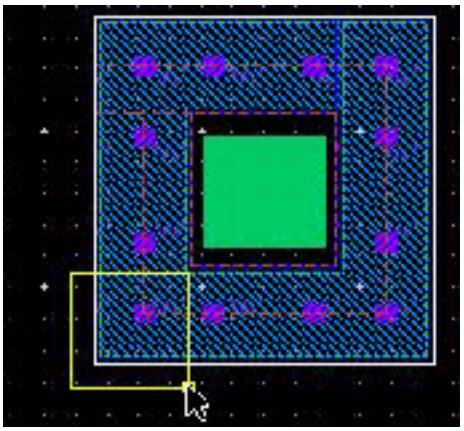
In the *Path* and *Overlapping Shape* modes, you can specify the exact dimensions of the tunnel to be created. The *Rectangle* and *Polygon* modes, on the other hand, provide you with the flexibility of defining a custom tunnel area that is based on the exact points you enter.

In both *Rectangle* and *Polygon* modes, you can pre-select or post-select the guard rings. If you do not select any guard ring before starting the *Tunnel* command, you are prompted to first select a guard ring and then specify the rectangle or polygon points. Tunnels are created through all the selected guard rings that are overlapped by the rectangle or polygon you create.

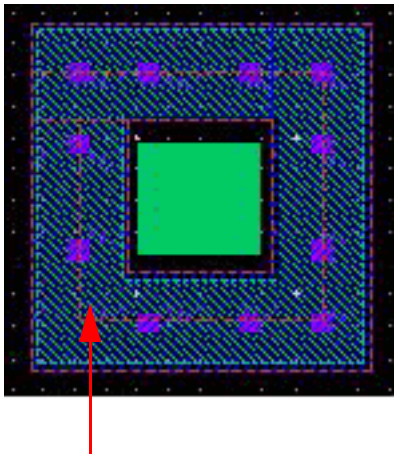
In the following example, the *Layer Purpose* for the rectangle tunnel is set to the contact layer of the guard ring. If you draw a rectangle to overlap the guard ring, any enclosing contact layer is removed.



Selected Guard Ring

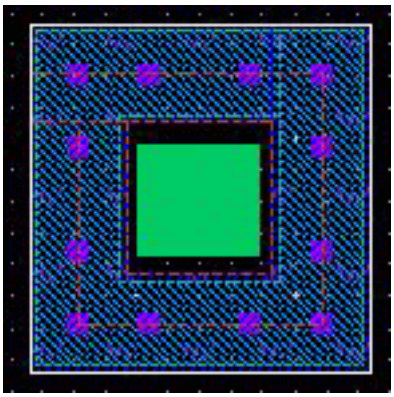


Draw the rectangle tunnel overlapping the guard ring.

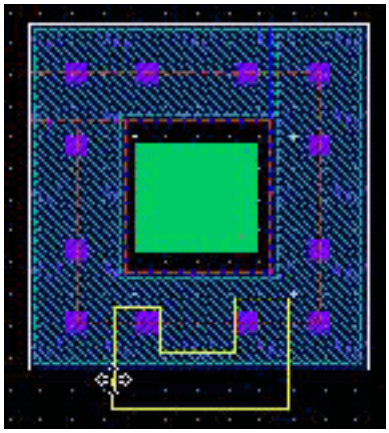


The contact layer is removed.

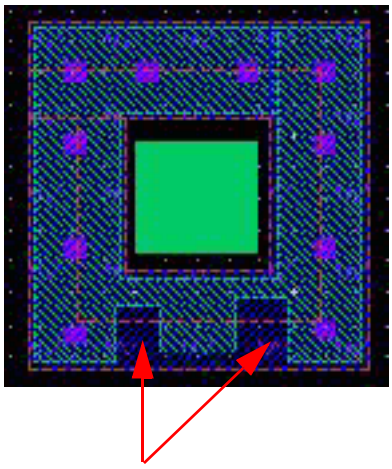
In the following example, the *Layer Purpose* for the polygon tunnel is set to the diffusion layer of the guard ring. The diffusion layer is removed from the areas where the polygon overlaps the guard ring.



Selected Guard Ring



Draw the polygon tunnel overlapping the guard ring.



The diffusion layer is removed.

Healing a Fluid Guard Ring

The *Heal* command removes one or more tunnels from an FGR. For more information about tunnels, see Creating a Tunnel Through a Fluid Guard Ring.

Note: The *Heal* command does not require you to select an FGR instance. However, you can heal only selectable layer-purposes.

To heal an FGR:

1. Choose *Edit – Fluid Pcell – Heal*.
2. Press `F3`.

   The Heal Fluid Object Form opens.

3. Select the *Mode* to be used to heal.
4. To heal by using the selected mode, use the following steps:
   a. *point*: Click a tunnel. The tunnel will be removed. You can use this method to remove the tunnels one at a time.
   b. *overlapping shape*: Click a shape that overlaps an FGR. All the tunnels overlapping the shape are removed.
   c. *rectangle*: Create a rectangle around the FGR instance from which you want to remove the tunnels. This method helps to remove a large number of unrequired tunnels quickly.
   d. *guard ring*: Click an FGR instance from which you want to remove all the existing tunnels. This method helps to heal a single FGR at a time.
5. Press `Esc` or click *Cancel* in the form to finish healing the FGRs.

Cleaning Overlapped Contacts from Fluid Guard Rings

The *Clean Overlapping Contacts* command lets you choose whether to clean all the overlapping contacts from an FGR, or just from the area on the canvas that you draw. This command also lets you clean contacts from two overlapping cells that have their FGRs in the hierarchy.

The overlapping contacts are removed without disturbing the placement of the non-overlapping contacts in the overlapping FGR. You can restore the removed overlapping contacts using the *Heal* command. You can also use the *Undo* and *Redo* commands after removing the overlapping contacts.

To clean the overlapping contacts:

1. Choose *Edit – Fluid Pcell – Clean Overlapping Contacts*.

   The Clean Overlapping Contacts Form opens.

2. Select the *guard ring* or *rectangle* radio button from the *Mode* option.

   - When you choose the *guard ring* button, click the guard ring to remove any overlapping contacts from it.
   - When you choose the *rectangle* radio button, keeping the left mouse button clicked, drag the cursor to select an area of the layout. This removes all overlapping contacts from the selected rectangle.

3. Select the *Clean contacts hierarchically* check box to clean overlapping contacts from the chosen guard ring while considering the guard rings across all levels of the hierarchy.

   When the *Clean contacts hierarchically* check box is selected, hovering the mouse pointer over the guard ring highlights the guard ring in magenta. It lets you know which guard ring is chosen for cleaning the overlapping contacts. On clicking the guard ring, you get a prompt message to confirm before the guard ring is modified.

Click *Yes* to modify the guard ring. Click *No* to cancel the operation.

If this option is not selected, overlapping contacts are cleaned from the chosen guard ring while considering only the current level of the hierarchy.

Note: The modified guard rings present in the lower hierarchy are not saved automatically. You need to explicitly save the modified cells using the *File – Save Hierarchically* option.

4. If required, continue to select more overlapping guard rings.
5. Press `Esc` to finish cleaning overlapped contacts.

5

Version Management

Fluid guard rings are SKILL/SKILL++ Pcells, with powerful creation and graphical editing capabilities. The Pcell code to evaluate them is located in the Virtuoso installation hierarchy. When you move to a different version of Virtuoso, geometry changes in existing FGR instances might occur, more details are described below. This can lead to issues as qualified designs that are already in production can change in a different release of Virtuoso and such designs would require re-qualification.

Fluid guard rings are dynamically evaluated Pcells. The Pcell code to evaluate them, Virtuoso Fluid Object (VFO), is shipped with the Virtuoso release (`vfo*.ils` files) under the following directory)

```
<cic_install_dir>/tools/dfII/etc/vfo
```

By default, the VFO code available in the Virtuoso hierarchy of the running Virtuoso session is used for FGR Pcell evaluation. Geometry changes in existing FGR instances can occur due to introduction of new or enhanced FGR features, or bug fixes made in the VFO code, or changes made in non-VFO SKILL functions used by the FGRs.

To ensure that the layout does not display unexpected geometry changes in existing FGR instances when moving to newer Virtuoso releases, the FGR version management solution is now available. This solution is based on caching on disk the FGR sub-masters that exist in a layout cellview.

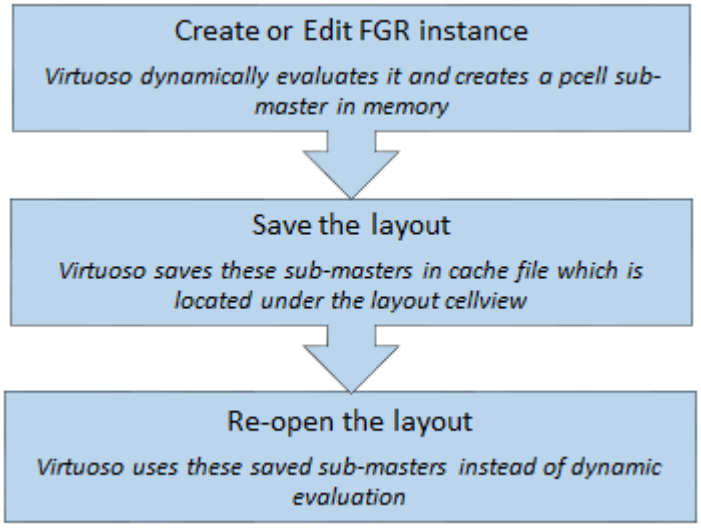
Note: The version management solution is only is applicable for FGR devices that have been defined through the Install Guard Ring form. This solution is not applicable to custom FGR devices that have been developed through custom SKILL or SKILL++ code, for example, advanced node FGRs.

Version Management Solution

The cache-based version management solution ensures that layouts are safe from unexpected geometry changes in existing FGR instances when moving to newer versions of Virtuoso.

When you create or edit an FGR instance, the FGR sub-masters in the memory are saved to a cache when you save the layout cellview. The cache is located under the containing layout cellview. When you open the layout, Virtuoso uses these saved sub-masters from the cache, rather than dynamically evaluating them.

The following figure describes the flow of the version management solution:



This solution ensures that the layout is safe from unexpected geometry changes in existing FGR instances when moving to newer Virtuoso versions. When you save an FGR instance, the FGR sub-master gets saved to the FGR cache. The version of the VFO code, for example FGR_617.0, used during the creation or editing of an FGR instance is saved in the cache in the `cacheCreateVersion` parameter of the sub-master.

When you re-open the layout in subsequent sessions, the FGR sub-master used in that layout will be read from the cache instead of the SKILL code being evaluated. This ensures that the geometries of instances remains unchanged across Virtuoso versions.

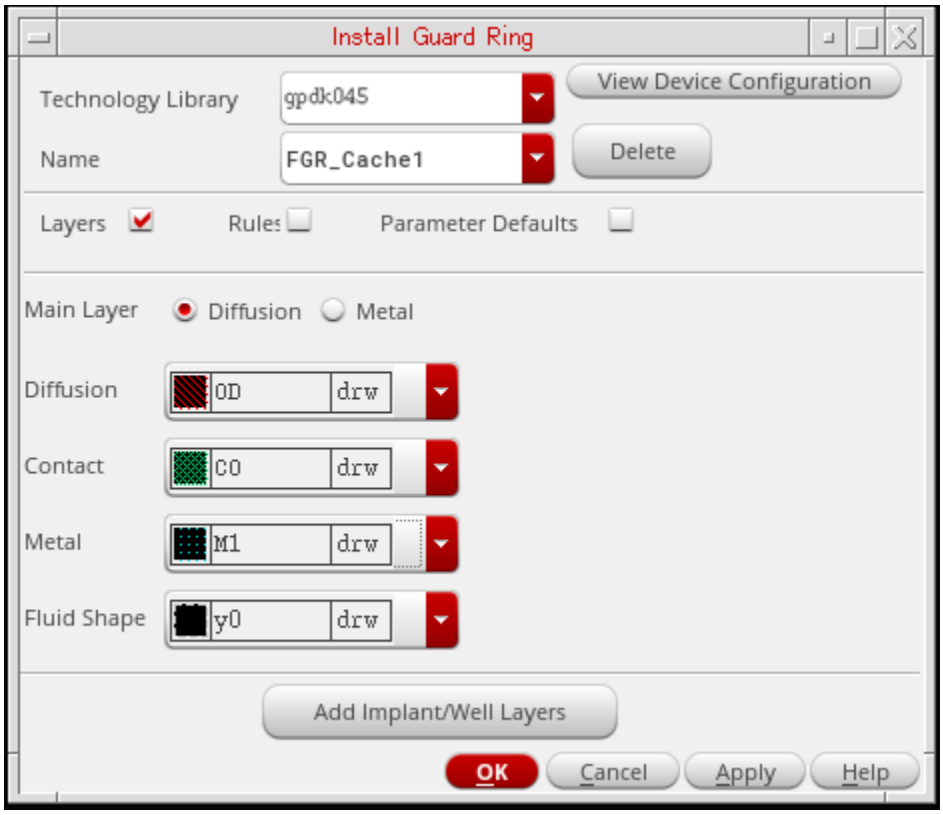
The FGR instances remain Pcells, so you can use the editing commands on these instances.

Each layout cellview containing FGR instances has its own cache. This avoids conflicts that can arise if a common cache is used for multiple layout cellviews.

This FGR version management solution is compatible with mainstream IC design management tools.

To enable version management for FGRs, you can do one of the following:

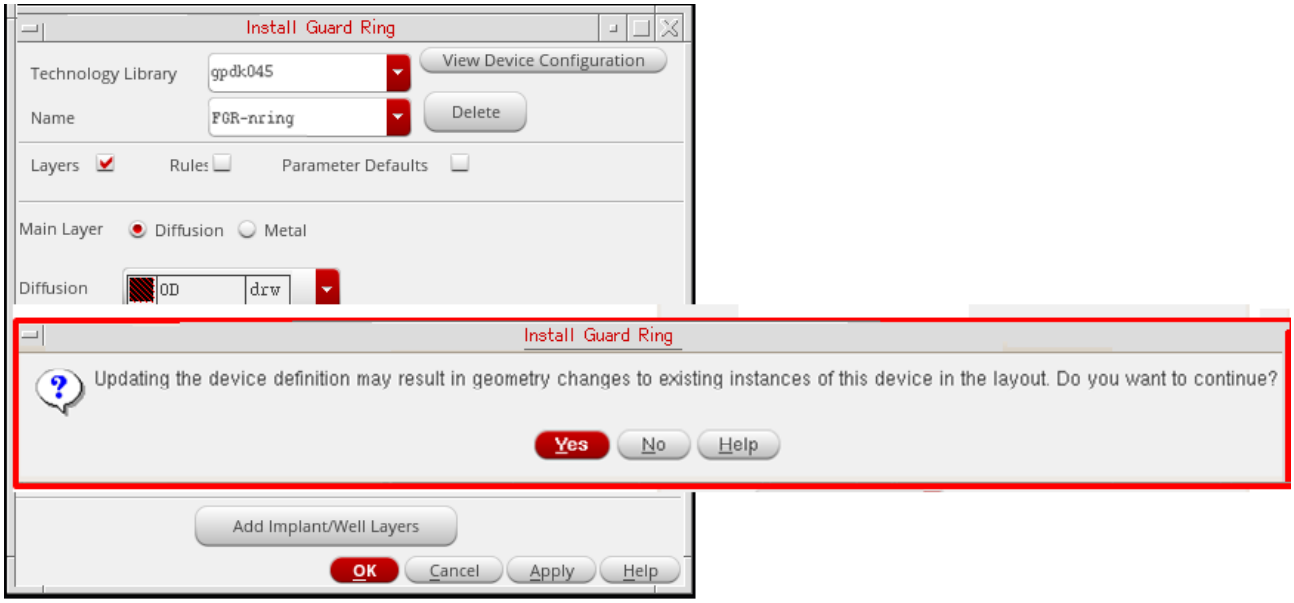
- Install a new FGR super-master in IC6.1.8 using the Install Guard Ring form



- Re-install the existing FGR super-master in IC6.1.8

  Re-installing an FGR device is the same as redefining a Pcell super-master. So, the geometries of existing FGR instances in the layout can change after re-installation of an

FGR device. You get a warning message before continuing with the FGR re-installation, as shown in the figure below.



After re-installing in IC617 a legacy FGR super-master that was originally installed in a prior version of Virtuoso, for example IC615 or IC616, newly created instances of such FGRs in IC617 will be saved to the FGR cache. However, existing legacy instances of such super-masters will not get stored in the cache automatically. They will continue being SKILL-evaluated. To get such instances stored to the FGR cache, you can do one of the following:

- Select these instance(s) in your layout, and on the Edit Properties form, click the *Update Version* button.
- Use the `vfoGREnableVersionCache` SKILL function.

Cache Files

The version management solution uses two binary files:

- Cache file, `cache.pcl`:

  The `cache.pcl` file contains the images of the evaluated sub-master.

- Index file, `index.pcl`:

  The `index.pcl` file contains the data for efficient retrieval of these images and other information of super-masters, such as the timestamp and parameter list.

These cache files are located under the Unix or Linux directory of the containing layout cellview.

The cache files get updated when you save the layout cellview after creating or editing FGR instances. The edit commands, such as `Merge`, `Chop`, `Tunnel`, and `Stretch`, involve changes in the FGR Pcell parameter values, and the edits that you make through the Edit Properties form, cause the cache to be updated when you save the layout.

The cache files do not get updated when you move or copy an FGR instance. When you move an FGR instance and its sub-master exists in the cache, the cache is not updated on saving the cellview. This is because the parameters of the instance do not change during the `Move` command. When you copy an FGR instance and its sub-master exists in the cache, then when you save the cellview, the cache to not updated. This is because both instances point to the same sub-master, and that sub-master already exists in the cache.

Note:

The writing and reading from the cache happens as a background activity. If you want to view the read and write information about FGR sub-masters, you can set the following Shell environment variable before starting the Virtuoso session.

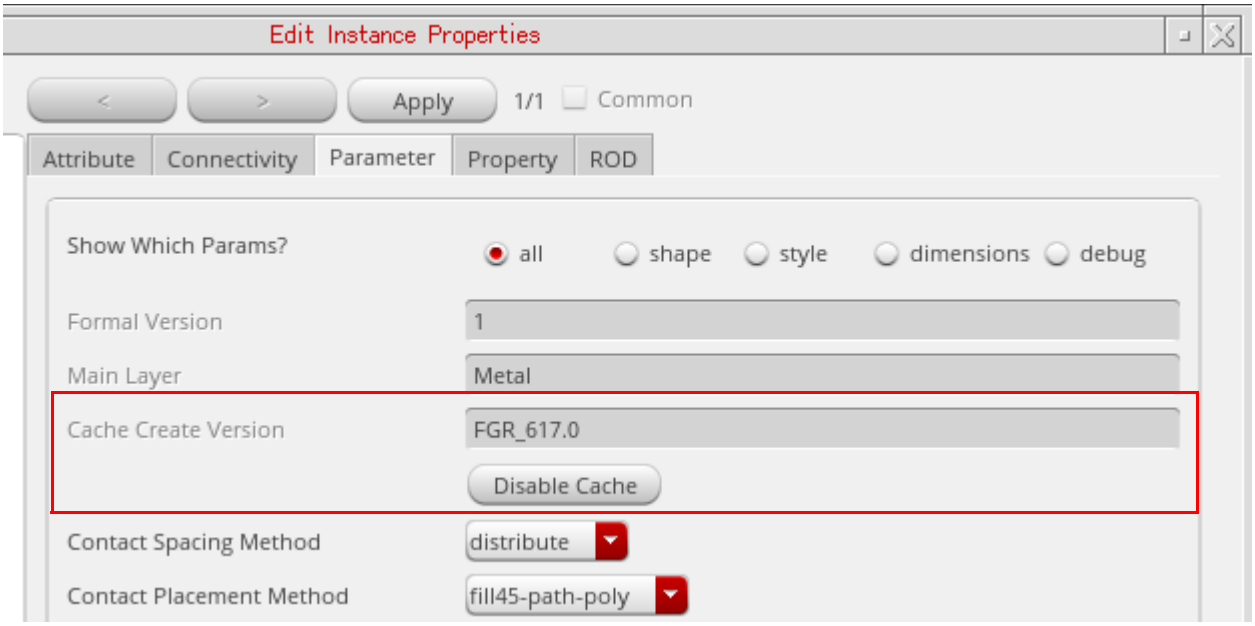
```
setenv VER_CACHE_DEBUG_MSG
```

If the cache files are read-only at the time of creating or editing an FGR instance, the value of the `cacheCreateVersion` parameter is set to 0 on that instance. At the time of saving the celllview, you get a warning message about all such FGR instances whose sub-masters will not be saved in the cache.

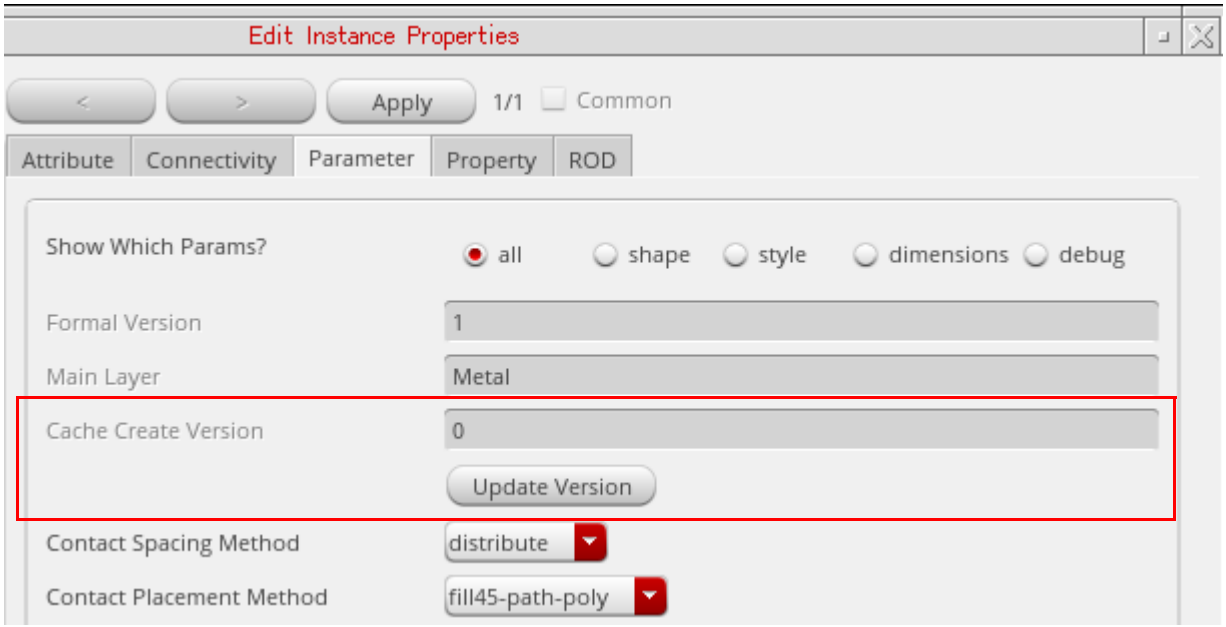
GUI Updates for Version Management

A new field, *Cache Create Version*, has been added to the Edit Instance Properties form. This field is visible when you select a cache-enabled FGR instance. The value of this field

corresponds to the VFO version that was used when that instance was created or edited, and its sub-master was saved to the FGR cache.



When you click the *Disable Cache* button, the value in the *Cache Create Version* field is set to `0`, as shown in the figure below.



When the value of this field is set to `0`, and you open this FGR instance in a new session, it is evaluated through SKILL instead of being read from the cache.

When you click the *Update Version* button, the selected FGR instance is re-evaluated, using the VFO version of the current Virtuoso session.

When you save the layout cellview containing the FGR instances, the sub-master of the FGR instances which are version enabled will get saved to the FGR cache, if it was not already present in the cache.

Cache Cleaning Mechanism

The version management solution has a cache cleaning mechanism to prevent increase in the cache file size. There are ways to automatically or manually clean the cache, as described below.

Automatic Cache Cleaning

The automatic cache cleaning mechanism prevents cache file size growth. When you delete all instances of an FGR sub-master from a layout cellview, the image of that sub-master is no longer needed.

Unused FGR sub-master images get deleted from the cache after you open the layout cellview in the scenarios mentioned below:

- While remaining in the current Virtuoso session:

  Purge the layout cellview from the Virtual Memory. In the CIW, select the File - Close Data - Close & Purge Data option. Then, re-open the layout cellview.

- After you close the current Virtuoso session:

  Start a new Virtuoso session. Open the layout cellview for the first time in the new Virtuoso session.

Note: Automatic cache cleaning is enabled by default. You can disable it by setting the Shell environment variable FGR_CACHE_AUTO_CLEANUP to OFF.

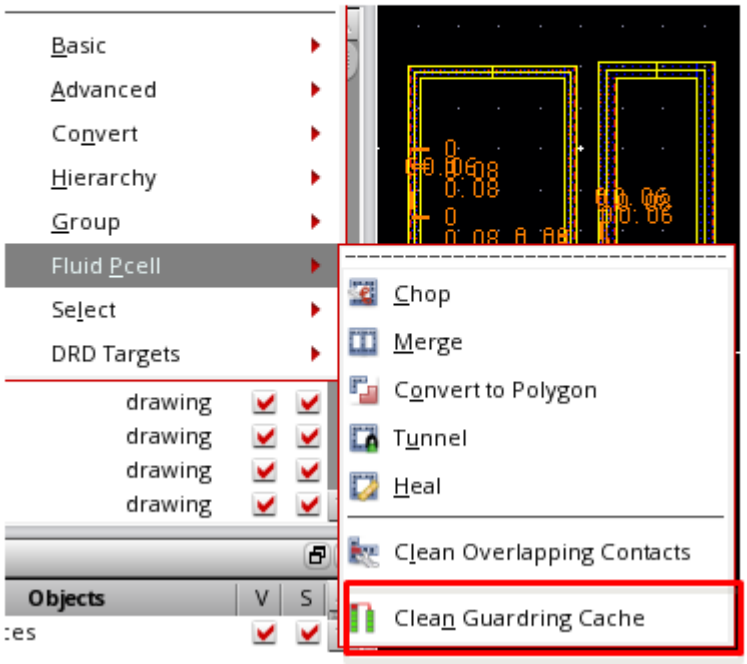
Manual Cache Cleaning

You can clean the cache files either by using the GUI options or the SKILL function described below:

- **GUI options:** You can clean the FGR cache of an open layout cellview by using either of the two options described below:
  - Using the *Clean Guardring Cache* icon on the FGR toolbar

or

- Using the *Edit - Fluid Pcell - Clean Guardring Cache*



- **SKILL function:** You can use the `vfoGRCleanVersionCache` SKILL function to clean the `index.pcl` and `cache.pcl` files and remove any unused entries of sub-masters from these files.

Important

You should ensure that you save any changes done in the layout cellview before you use the `vfoGRCleanVersionCache` SKILL function.

Note: After removing all FGR instances from a layout cellview and cleaning its FGR cache, the `cache.pcl` and `index.pcl` files continue to exist for the cellview, even if they do not contain any FGR sub-masters. However, the size of these files will be `0` bytes.

Known Limitations of Version Management Solution

The version management solution has the following known limitations:

- Consider a scenario where you create instances of FGR devices that support caching, such as in IC617 FCS or higher, and re-open the layout cellview in an older version of Virtuoso that does not support FGR caching, such as IC616 or older. In such a scenario, you will get Pcell evaluation errors for such FGR instances. Samples of some errors that you will see are shown below:

  ```
  *WARNING* (DB-270001): Pcell evaluation for gpdk045/FGR_Cache1/
  layout has the following error(s):

  *WARNING* (DB-270002): ("error" 1 t nil ("*Error* Unable to create
  a device named 'FGR_Cache1' because of (\"slotValue\" 0 t nil
  (\"*Error* slotValue: no such slot - cacheCreateVersion in class
  vfoGuardRing_ver_1. Valid slot names: (modelLpp vfoProtocolClass
  keepOuts hide_keepouts ... contAlignment
  removeCornerContacts)\")). Redefine the device and try
  again.\n"))

  *WARNING* (DB-270003): Error kept in "errorDesc" property of the
  label "pcellEvalFailed" on layer/purpose "marker/error" in the
  submaster.
  ```

- If there is mismatch between the timestamp of the sub-master and super-master, you will see the following warning message when you open the cellview:

  ```
  *WARNING* (VFO-113026): The cellview 'reflib/cacheTest/layout'
  contains instances of the superMaster 'reflib/cacheDev1/layout'
  that was modified after the creation of the version cache. The
  changed geometries of the instances in this cellview might not be
  saved in the version cache. Contact Cadence Customer Support for
  assistance to update the version cache.
  ```

  Even after the timestamp mismatch, by default, the sub-master would be read from the cache to create the instance geometry. To ignore the saved sub-master and create an FGR instance by evaluating it in the current Virtuoso version, you can use the Shell environment variable, FGR_CACHE_TIMESTAMP_CHECK.

  In this case, you will also see the following warning message:

  ```
  *WARNING* (VFO-113028): The superMaster was modified after the
  creation of the version cache. The instance was evaluated with the
  latest VFO code, but the 'cacheCreateVersion' CDF parameter will
  not be updated.
  ```

- If the `cache.pcl` file gets corrupted and Virtuoso version is not same as the sub-master version saved in the `cache.pcl` file, the pcell evaluation will fail. In this case, to evaluate pcells with the current Virtuoso version, you can use the Shell environment variable, FGR_REEVAL_ON_CORRUPT_CACHE. This variable evaluates pcells submasters

only in memory. To rectify corruption in the `cache.pcl` file, use the SKILL funtion vfoGetInstWithMissingCache.

SKILL Functions

The SKILL functions listed below let you manually update the FGR cache of one or more layout cellviews.

- vfoGRCleanVersionCache
- vfoGRDisableVersionCache
- vfoGREnableVersionCache
- vfoGRUpdateVersionCache

A

Fluid Guard Ring Form Descriptions

This appendix covers detailed information about the following forms of Layout L that help you in creating and managing FGRs:

- Clean Overlapping Contacts Form
- Create Guard Ring Form
- Edit Instance Properties Form
- Chop Fluid Objects Form
- Create Tunnel in Fluid Object Form
- Heal Fluid Object Form
- Install Guard Ring Form

For details about the other forms, refer to the Layout L Forms appendix in *Virtuoso Layout Suite L User Guide*.

Clean Overlapping Contacts Form

See Cleaning Overlapped Contacts from Fluid Guard Rings for related information.

Mode

guard ring removes the overlapping contacts from the guard ring selected using the mouse click.

rectangle removes the overlapping contacts from the specific area on the canvas that you draw by dragging the mouse over it.

Clean contacts hierarchically, if selected, cleans overlapping contacts from the active guard ring while considering the guard rings in the current and lower levels of the hierarchy. If deselected, cleans overlapping contacts from the active guard ring while considering only the current level of the hierarchy.

Note: The contacts are cleaned only at the current level; the lower-level layouts are not edited.

Create Guard Ring Form

See Creating Fluid Guard Rings for related information.

The form comprises the following tabs:

- Wrap Tab
- Path Tab
- Rect Tab
- Polygon Tab

The subtabs on these tabbed pages include:

- Contact Settings
- Implant Layers
- Outer Rings

Wrap Tab

This mode automatically creates a guard ring around the selected objects.

Technology displays the technology library in which the guard ring device is installed. This field is not editable.

Device enables you to select one of the installed FGR devices in the displayed technology library and its referenced technology libraries (ITDB). For more information about ITDB, see the *Virtuoso Technology Data User Guide*.

Note: The names of the FGR devices displayed in the *Device* list box are sorted alphanumerically.

Contact Rows enables you to specify the number of via rows. *Contact Rows* and *Path Width* are inter-dependent fields. Modifying the value in one updates the value in the other field.

Path Width enables you to specify the width of the guard ring. *Path Width* and *Contact Rows* are inter-dependent fields. Modifying the value in one updates the value in the other field.

Net Name enables you to specify the container cellview's net with which you want to connect the terminal of the guard ring master.

Main Layer displays the layer that will be used for calculating the path width. This field is not editable. To edit the value of the main layer, use the Install Guard Ring form.

You can select one of the following shapes for the wrap-around guard ring to be created: **Rectangular** or **Rectilinear**.

Wrap common enables you to create a common guard ring around the selected objects. When off, individual guard rings are created around each of the selected objects.

Place at Minimum Distance, if selected, uses the `minSpacing` rule defined for the object layer around which the FGR is created to compute the spacing between the guard ring and the object layer.

Note: The object layer includes both original and derived layers,

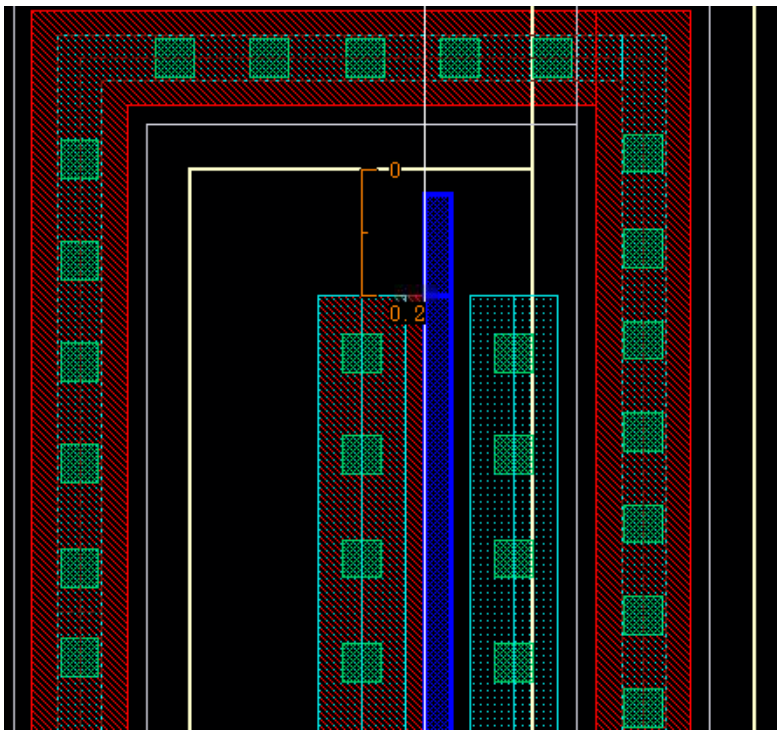
The *Place at Minimum Distance* check box is selected by default. If you deselect this check box, you can specify the guard ring distance from the object in the *Enclose by* field.

Environment variables: fgrWrapPlaceAtMinimumDistance and disableDerivedLayersInWrap

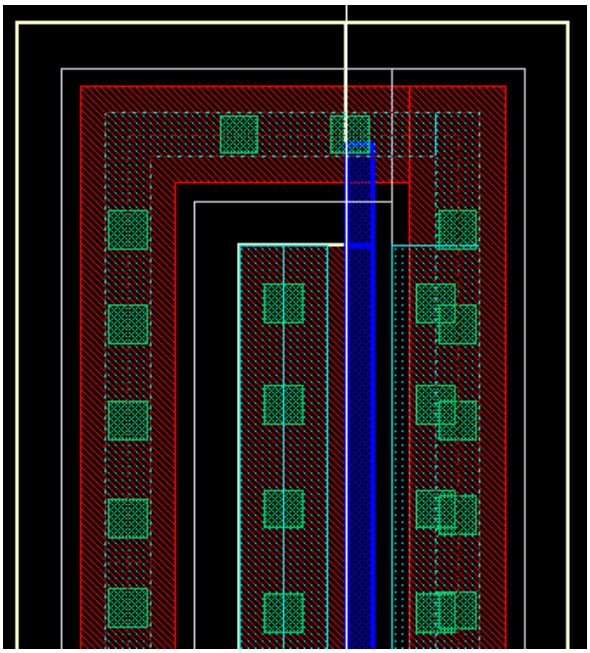
Enclose by enables you to specify a distance from the object at which the fluid guard ring should be placed. When you specify an *Enclose by* value, positive or negative, the distance is calculated from the outermost layer of the fluid guard ring to the outermost layer of the enclosed device.

Note: This field is enabled only when *Place at Minimum Distance* is off.

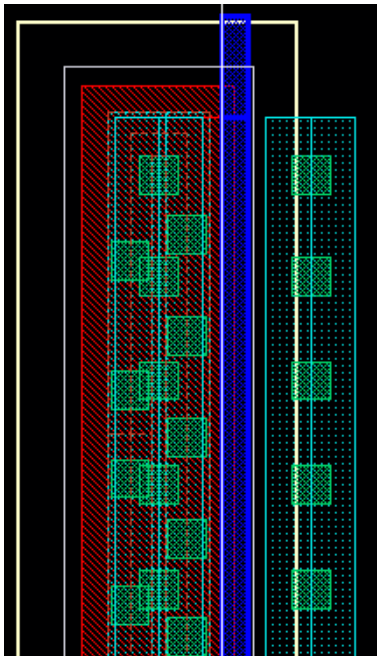
The following figures show a preview of FGRs with *Enclose By* value as `0.2` (positive), `0`, and `-0.2` (negative).



Enclose By = 0.2



Enclose By = 0



Enclose By = -0.2

For options on the subtabs, see Contact Settings, Implant Layers, and Outer Rings.

Path Tab

This mode creates a path guard ring.

Technology displays the technology library in which the guard ring device is installed. This field is not editable.

Device enables you to select one of the installed FGR devices in the displayed technology library and its referenced technology libraries (ITDB). For more information about ITDB, see the *Virtuoso Technology Data User Guide*.

> **Note:** The names of the FGR devices displayed in the *Device* list box are sorted alphanumerically.

Contact Rows enables you to specify the number of via rows. *Contact Rows* and *Path Width* are inter-dependent fields. Modifying the value in one updates the value in the other field.

Path Width enables you to specify the width of the guard ring. *Path Width* and *Contact Rows* are inter-dependent fields. Modifying the value in one updates the value in the other field.

Net Name enables you to specify the container cellview's net with which you want to connect the terminal of the guard ring device master.

Main Layer displays the layer that will be used for calculating the path width. This field is not editable. To edit the value of the main layer, use the Install Guard Ring form.

Snap Mode enables you to control how path segments snap to the grid.

For options on the subtabs, see Contact Settings and Implant Layers. You cannot create concentric rings for a path guard ring.

Rect Tab

This mode creates a rectangular guard ring.

Technology displays the technology library in which the guard ring device is installed. This field is not editable.

Device enables you to select one of the installed FGR devices in the displayed technology library and its referenced technology libraries (ITDB). For more information about ITDB, see the *Virtuoso Technology Data User Guide*.

Note: The names of the FGR devices displayed in the *Device* list box are sorted alphanumerically.

Contact Rows enables you to specify the number of via rows. *Contact Rows* and *Path Width* are inter-dependent fields. Modifying the value in one updates the value in the other field.

Path Width enables you to specify the width of the guard ring. *Path Width* and *Contact Rows* are inter-dependent fields. Modifying the value in one updates the value in the other field.

Net Name enables you to specify the container cellview's net with which you want to connect the terminal of the guard ring master.

Main Layer displays the layer that will be used for calculating the path width. This field is not editable. To edit the value of the main layer, use the Install Guard Ring form.

Automatically adjust to surround overlaps allows the created guard ring to shrink or grow automatically to surround the overlapped object layers, which include both original and derived layers, provided the relevant rules exist in the technology library.

For options on the subtabs, see Contact Settings, Implant Layers, and Outer Rings.

Polygon Tab

This mode creates a polygon guard ring.

Technology displays the technology library in which the guard ring device is installed. This field is not editable.

Device enables you to select one of the installed FGR devices in the displayed technology library and its referenced technology libraries (ITDB). For more information about ITDB, see the *Virtuoso Technology Data User Guide*.

> **Note:** The names of the FGR devices displayed in the *Device* list box are sorted alphanumerically.

Contact Rows enables you to specify the number of via rows. *Contact Rows* and *Path Width* are inter-dependent fields. Modifying the value in one updates the value in the other field.

Path Width enables you to specify the width of the guard ring. *Path Width* and *Contact Rows* are inter-dependent fields. Modifying the value in one updates the value in the other field.

Net Name lets you specify the container cellview's net with which you want to connect the terminal of the guard ring master.

Main Layer displays the layer that will be used for calculating the path width. This field is not editable. To edit the value of the main layer, use the Install Guard Ring form.

You can create the polygon guard ring to achieve one of the following:

> **Create Fill** covers up the space between objects using the guard ring device material. When this radio button is selected, the *Contact Rows* and *Path Width* fields are disabled.
>
> **Create Ring** creates a ring using the entered polygon points.

Snap Mode enables you to control how polygon snaps to the grid.

For options on the subtabs, see Contact Settings and Implant Layers. You cannot create concentric rings for a polygon guard ring.

Contact Settings

Calculated Parameter enables you to specify whether a change in the contact settings should impact the *Path Width* or *Contact Rows*.

> **Note:** When creating a polygon-type fluid guard ring, the *Calculated Parameter* field on the *Polygon* tab of the Create Guard Ring form appears as disabled.

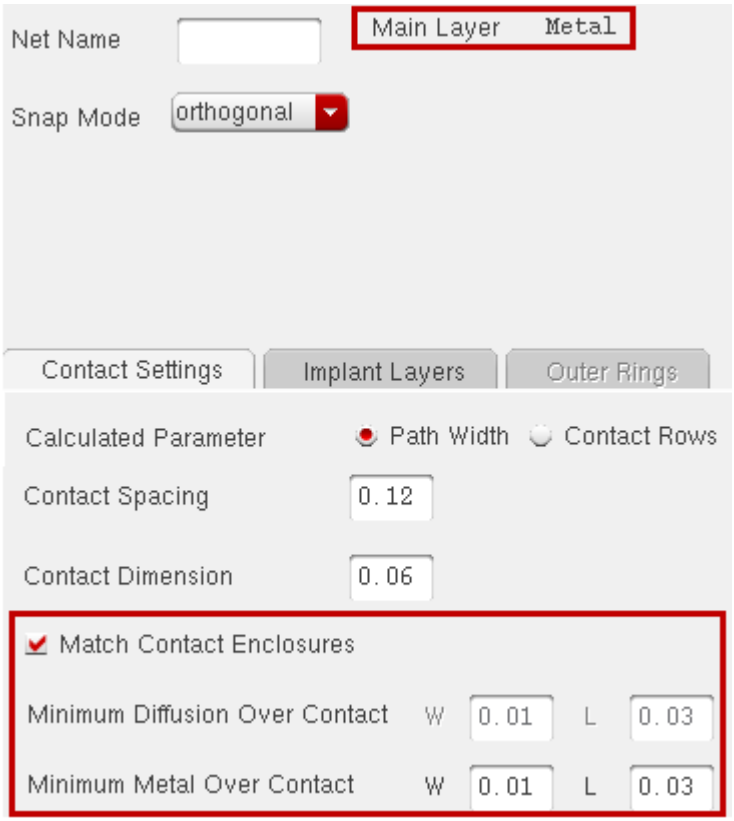
Contact Spacing indicates the spacing between vias. The pre-populated value is derived from the *contactLayer Spacing* value in the Rule section in the Install Guard

Ring Form. The value in this field is automatically computed using the `viaSpacing` rule, based on the value you specify in the *Contact Rows* field. If the `viaSpacing` rule for the specified number of *Contact Rows* is not found, then the lookup for the next applicable `viaSpacing` rule is performed. If no applicable `viaSpacing` rule is found, the `minSpacing` rule value is used to compute the contact spacing. If no applicable `minSpacing` rule is found, the `minViaSpacing` rule value is used to compute the contact spacing. However, the `minViaSpacing` value is used only if the Shell environment variable, `FGR_MIN_VIA_SPACING_ENABLED`, is set to `1`/`t`/`T`/`true`/`TRUE`. When the `minViaSpacing` rule is used, all the constraint parameters are ignored and the spacing value is always considered from edge to edge.

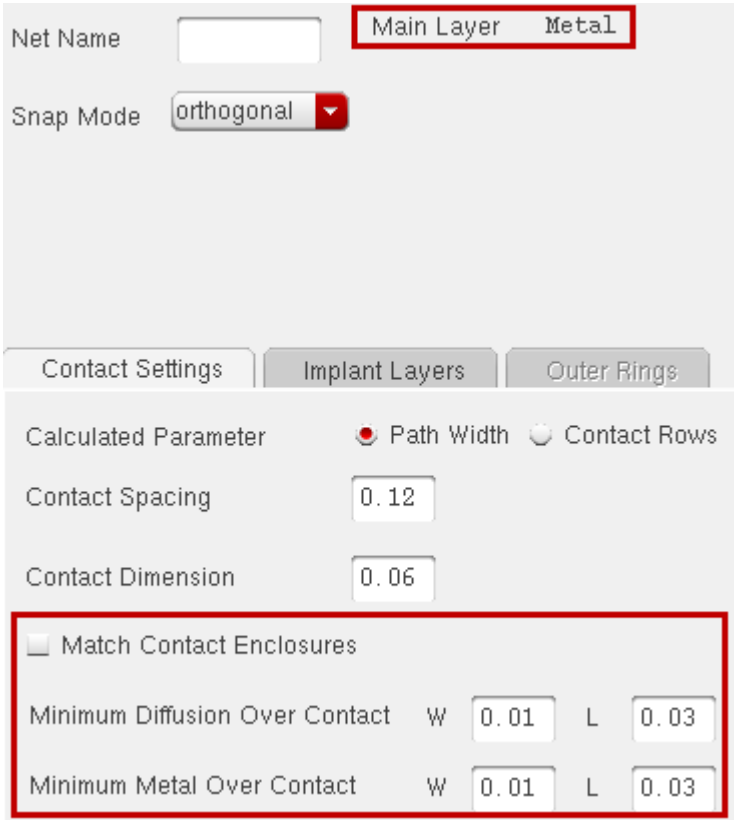
Contact Dimension enables you to specify the via dimension to use. The via is assumed to be rectangular. The pre-populated value is derived from the *contactLayer Dimensions* value in the Rule section in the Install Guard Ring Form.

Match Contact Enclosures allows you to make the enclosure value of the Main Layer control the enclosure value of the other (non-main) layer. If you select this check box, you can keep both enclosure values the same. When you deselect this check box to disable the feature, you can set the *Minimum Metal over Contact* and *Minimum Metal over Diffusion* values independently.

For example, in the following image when the Main Layer is `Metal` and the *Match Contact Enclosures* check box is selected, the *Minimum Diffusion Over Contact* (*W* and *L*) field becomes disabled:



However, when the *Match Contact Enclosures* check box is not selected, the *Minimum Diffusion Over Contact* (*W* and *L*) field remains enabled for editing:



Minimum Diffusion Over Contact enables you to change the diffusion layer enclosure around vias in the *W* and *L* direction. The pre-populated value is derived from the *diffusionLayer over contactLayer* value in the Rule section in the Install Guard Ring Form.

> **Note:** When you choose to create an FGR of *Wrap* or *Rect* type, the enclosure value for only the *W* direction is required. Therefore, the field for the *L* direction is not displayed.

Minimum Metal Over Contact enables you to change the metal layer enclosure around vias in the *W* and *L* direction. The pre-populated value is derived from the *metalLayer over contactLayer* value in the Rule section in the Install Guard Ring Form.

> **Note:** When you choose to create an FGR of *Wrap* or *Rect* type, the enclosure value for only the *W* direction is required. Therefore, the field for the *L* direction is not displayed.

Implant Layers

This tab lists all the implant and well layers defined in the installed guard ring device.

Adjust Implant/Well Layers table

Use enables you to specify whether or not to use the implant layer in the guard ring you create.

Lpp displays the implant or well layer.

Encl enables you to specify the implant layer enclosure around vias.

Pin, if selected, draws the implant layer in the guard ring with a pin-like shape. The implant layer then has the same connectivity as the metal layer of the guard ring. For example, if you have an MOS device with VDD bulk connection, then while creating a fluid guard ring with an implant layer that matches the MOS bulk layer, the fluid guard ring is automatically connected to the VDD.

Cover, if selected, fills the interior of the guard ring with the implant or well layer. This field is available only for ring type of guard rings. Therefore, this column is not applicable in the *Path* mode.

Outer Rings

Note: This tab is disabled for the Path and Polygon guard ring creation modes.

Number of Rings enables you to specify the number of concentric rings of the guard ring to create around the selected objects.

Distance outer rings at minimum, if selected, uses the `minSpacing` rules to compute the spacing between the concentric rings. If you deselect this check box, you can specify the distance between the consecutive guard rings in the *Encl* field.

Ring *<n>* enables you to specify the guard ring device to use for the concentric rings. You can create a guard ring with a maximum of six concentric rings. If you specify a number greater than `6` in the *Number of Rings* field, the number is reset to the previous specified value.

Fields for the concentric rings are added to the form only when the specified value in the *Number of Rings* field is greater than `1`. The number of *Ring <n>* fields added to the form depend on the value specified in the *Number of Rings* field. For example, if you specify `4` in the *Number of Rings* field, three fields – *Ring 2*, *Ring 3*, and *Ring 4* – are added to the form.

The guard ring device selected from the *Device* list at the top of the form is used to create the innermost ring. The rings listed in the *Outer Rings* tab comprise the inner to outer rings.

Rows lets you specify the number of contact rows in the ring.

Encl lets you specify the distance between two consecutive rings. The *Encl* field is grayed out if the *Distance outer rings at minimum* check box is selected.

Net Name lets you specify the name of the net to which the outer ring should be connected.

Defaults resets the values in the *Contact Rows* field and the *Contact Spacing Method* list to the values specified in the *Number of contact rows* and *Contact Spacing Method* fields in the Parameter Defaults section in the Install Guard Ring Form.

Edit Instance Properties Form

To display the Edit Instance Properties form, select *Properties* from the context-sensitive *Instance* menu displayed by a right-click a selected FGR instance. The displayed form has the tabs, fields, and buttons explained below.

The Previous button () highlights the previous object in the group of selected objects and updates the form to show that object's properties.

The Next button () highlights the next object in the group of selected objects and updates the form to show that object's properties.

Common lets you edit properties common to a group of selected objects.

Note: The Common functionality does not support ROD objects.

> **Layer** shows the layer of a common property if it is the same for all objects. If the layer is *not* the same for all objects, *AS IS* appears in the field.
>
> **Left, Right, Bottom,** and **Top** show the value of a common property if it is the same for all objects. If the value of a common property is not the same for all objects, the words *AS IS* appears in the fields.

Attribute tab shows the attributes of the currently selected object. It provides the following fields for an FGR instance:

> **Library**, **Cell**, and **View** set the library, cell, and view names of the master cell for this instance.
>
> **Origin: X** and **Y** set the *X* and *Y* coordinates of the origin of the instance.
>
> **Name** sets the name assigned to this instance. The layout editor automatically assigns instance names that begin with the letter `I`, followed by a number.
>
> **Rotation** sets whether the instance is rotated or mirrored. See Orientation Valid Values.
>
> **Cell Type** read only field displays the master cell type used to describe the nature of a physical block or cell. Layout L does not support any level one editing on any of these design objects. These cell types are used in Virtuoso Layout Suite XL and Virtuoso Floorplanning. See Cell Type Valid Values.
>
> **Placement Status** drop-down list box displays the placement status of the instance. The selections are none, unplaced, suggested, placed, locked, and firm. See Placement Status Valid Values.

Connectivity tab shows routing and net information for selected pins and shapes on a net. *I/O Type* and *Access Direction* are shown only for pins.

Net Name displays the name of the net to which the pin is connected. You can use this to rename or delete the net.

> **Note:** When you edit the net name of via or pathSeg that is a route element, the *Net Name* field appears as the *Route Net Name* field. This indicates that editing the net of an object within a route changes the net of the entire route.

Terminal Name sets the name of the terminal associated with this pin. The terminal name should always be the same as the net name.

Net Expression assigns a net expression of the terminal listed in the *Terminal Name* field.

> **Property** sets the override property name to the net expression.
>
> **Default** defines the net to be used if no override property is defined in the hierarchy above this point (in the schematic view). Unless a different signal name is entered, the terminal name is used.

I/O Type assigns a property used by routers to identify the direction of the signal into or out of this cellview. The signal can be input, output, inputOutput (bidirectional), switch (carries data either in or out, but not simultaneously), or jumper (passes data through this cellview).

Net Criticality sets a weighting factor that determines the priority for this net for the Cadence place-and-route tools.

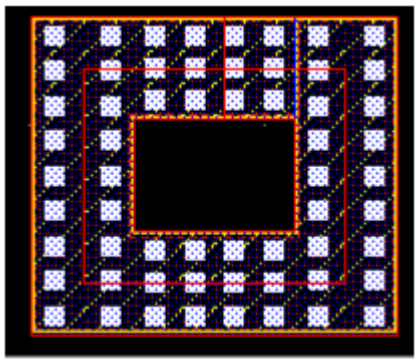
Access Direction assigns a property used to identify the part of the pin to which the routers can connect routing. Applies only to rectangle pins. The selections are: *top*, *left*, *bottom*, *right*, *any*, and *none*.

Parameter enables you to edit the properties of the selected fluid guard ring.

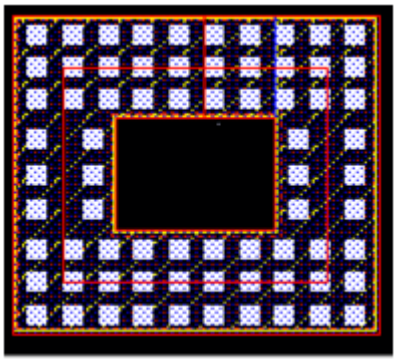
> **Show Which Params?** enables to choose the type of parameters you want to edit. The available options are all, shape, style, dimensions, and debug. Based on the selection, only parameters associated to the selected type are displayed.
>
> **Formal Version** displays the version number of the selected fluid guard ring.
>
> **Main Layer** displays the layer that defines the path width of the fluid guard ring, that is *Diffusion* or *Metal*.

Contact Placement Method enables you to change the way the contacts should be placed within the fluid guard ring. You can choose one of the following ways:

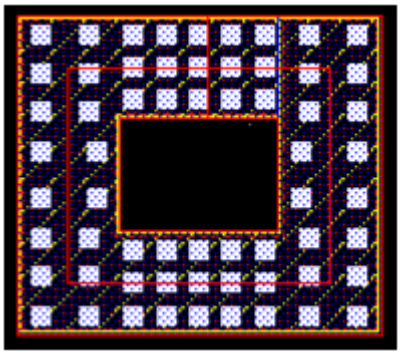
vertically — Partitions the fluid shape vertically and places the contacts in individual partitions, as shown in the following image:



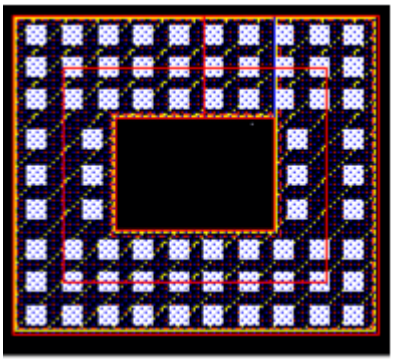
horizontally — Partitions the fluid shape horizontally and places the contacts in individual partitions, as shown in the following image:



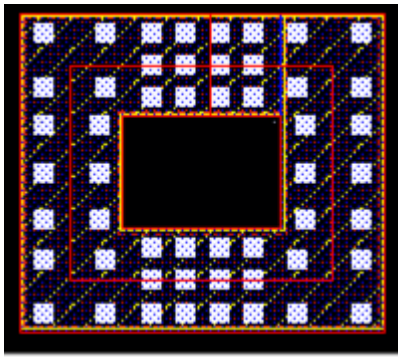
both — Partitions the fluid shape vertically and horizontally, and places the contacts in individual partitions.



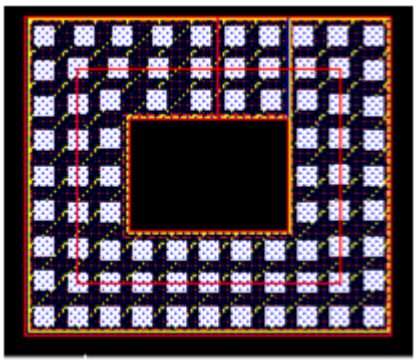
by-area Partitions the fluid shape on area basis and places the contacts in individual partitions.



by-aspect-ratio Partitions the fluid shape based on aspect ratio and places the contacts in individual partitions.



fill45-path-ploy Places the contacts a way that keeps them aligned.



Shape Data displays the different dimensions of the fluid guard ring.

Center Line enables to change the dimensions of the fluid guard ring.

Note: The Edit Instance Properties form of a polygon-type fluid guard ring ***does***

not have this field.

Path Width enables you to change the width of the path you draw.

> **Note:** The Edit Instance Properties form of a polygon-type fluid guard ring ***does not*** have this field.

Num Contact Rows enables you to change the number of rows of contacts in the fluid guard ring.

> **Note:** The Edit Instance Properties form of a polygon-type fluid guard ring ***does not*** have this field.

Calculated Parameter enables you to specify whether a change in the contact settings should impact the *Path Width* or *Contact Rows*.

> **Note:** The Edit Instance Properties form of a polygon-type fluid guard ring ***does not*** have this field.

Perimeter enables to edit the circumference of a polygon-type fluid guard ring.

> **Note:** The Edit Instance Properties form of a polygon-type fluid guard ring do display this field.

Contact Width enables to change the value describing the current width of the contacts.

Contact Space enables to change the value describing the current spacing between the contacts.

Match Contact Enclosure enables to make the enclosure value of the Main Layer control the enclosure value of the other (non-main) layer. It allows you to keep both enclosure values the same. When you disable this option by deselecting the check box, you can set the *Minimum Metal over Contact* and *Minimum Metal over Diffusion* values independently.

Minimum Metal Over Contact enables you to change the metal layer enclosure around the contacts in the width (W) and length (L) directions. The pre-populated value is derived from the metalLayer over contactLayer value in the Rule section in the Install Guard Ring Form.

> **Note:** The Edit Instance Properties form of a Wrap or Rectangle type of fluid guard ring displays only width-related *Minimum Metal Over Contact*.

Minimum Diffusion Over Contact enables you to change the diffusion layer enclosure around the contacts in the width (W) and length (L) directions. The pre-populated value is derived from the diffusionLayer over contactLayer value in the Rule section in the Install Guard Ring Form.

Note: The Edit Instance Properties form of a Wrap or Rectangle type of fluid guard ring displays only width-related *Minimum Metal Over Contact*.

<Implant/Well Layers name> Use? check box is displayed as selected only if you defined an implant or well layer while installing the fluid guard ring. If you deselect this check box, the other related fields of the *Edit Instance Properties* form, that is, *Enclosure*, *Cover Interior*, and *Create Pin*, are automatically removed.

Enclosure enables you to specify the implant layer enclosure around the contacts.

Cover Interior, if selected, fills the interior of the guard ring with the implant or well layer. This field is available only for ring type of fluid guard rings. Therefore, this check box is not available in Path mode.

Create Pin, if selected, draws the implant layer in the fluid guard ring with a pin-like shape. The implant layer then has the same connectivity as the metal layer of the fluid guard ring.

Hide Keepouts?, if deselected, enables you to see the keepout shapes.

Note: When you create a tunnel through a fluid guard ring instance, Virtuoso uses keepouts to prevent the tunneled layer(s) appearing at the edited location on the instance. These keepout shapes are normally invisible. The *Hide Keepout?* check box can be used to make the keepout shapes visible.

Property shows the properties for the selected fluid guard ring. You can manage (that is, add, delete, and modify) your own properties for a fluid guard ring. For details, refer to the *Add Property Form* section in the *Virtuoso Layout Suite L User Guide*.

ROD displays the ROD name, handle, and alignment information of the selected fluid guard ring. For information about the displayed fields, refer to the *ROD Property Form* section in the *Virtuoso Layout Suite L User Guide*.

Orientation Valid Values

Value	Description
RO	no rotation
R90	rotated 90 degrees
R180	rotated 180 degrees
R270	rotated 270 degrees
MY	mirrored over the Y axis

Value	Description
MYR90	mirrored over Y, rotated 90 degrees
MX	mirrored over the X axis
MXR90	mirrored over X, rotated 90 degrees

Placement Status Valid Values

Value	Description
none	The instance has no current status.
unplaced	Unplaced instances can be placed and moved. The current placement is arbitrary and should be changed. Unplaced objects do not contribute to the bounding box of the containing design, and they will not be returned by region query.
suggested	Instances in a cluster may be placed outside of their associated cluster boundaries.
placed	Placed instances can be moved by automatic and interactive commands. This shows that an effort has been made by the software to place this object but it may be placed in future attempts.
locked	Locked instances cannot be moved by any commands. The placement of this object should not be changed.
firm	Firm or fixed instances can be moved by interactive commands.

Chop Fluid Objects Form

See Chopping a Fluid Guard Ring for related information.

Mode

rectangle specifies a rectangle as the chop shape. The chop area is determined by the exact points clicked to define the rectangle. The default chop shape is *rectangle*.

line specifies a line as the chop shape. The total chop area is calculated as a sum of the spacing on both sides of a line.

Spacing enables you to specify the spacing around the line chop shape. The line along with the spacing around the line is chopped from a guard ring. By default, the `minSpacing` of the current active layer is used. If the specified value is less than `minSpacing`, the value in the field is reset to the `minSpacing` value.

Default enables you to revert the specified value to the technology file default.

overlapping shape enables you to use a shape that overlaps a guard ring to chop the guard ring. The total chop area is calculated as a sum of the overlap area and the specified spacing around the overlapping shape.

Note: You cannot use a guard ring that overlaps another guard ring to perform the chop operation.

Spacing enables you to specify the spacing around the overlapping shape that, along with the overlap region, is chopped from a guard ring. By default, the `minSpacing` of the current active layer is used. If the specified value is less than `minSpacing`, the value in the field is reset to the `minSpacing` value.

Default enables you to revert the specified value to the technology file defaults.

polygon specifies a polygon as the chop shape. The chop area is determined by the exact points clicked to define the polygon.

Create Tunnel in Fluid Object Form

See Creating a Tunnel Through a Fluid Guard Ring for related information.

Tunnel Shape

Path enables you to draw a path to define a tunnel. The total area of the created tunnel is calculated as a sum of the *Path Width* on both edges of the path. When you select this option, the following columns with editable fields are displayed:

Layer Purpose enables you to select the layer-purpose through which the tunnel can be created. It represents the layer-purpose that is removed from the guard ring to create a tunnel. A check box in front of each layer-purpose field enables you to specify more than one layer-purpose at a time though which the tunnel can be created. By default, a row is added for each maskable layer purpose comprising the guard ring.

Width enables you to specify a value for the width of the path you draw. The default value is the minimum width of the layer-purpose selected from the *Layer Purpose* list. If you enable multiple layer-purposes for creating the tunnel, the width defined for each layer-purpose is used to create the tunnel on that layer-purpose.

Rectangle enables you to draw a rectangle to define a tunnel. The total area of the created tunnel is calculated as a sum of the area of the drawn rectangle and *Spacing*. When you select this option, the following columns with editable fields are displayed:

Layer Purpose enables you to select the layer-purpose through which the tunnel can be created. It represents the layer-purpose that is removed from the guard ring to create a tunnel. A check box in front of each layer-purpose field enables you to specify more than one layer-purpose at a time though which the tunnel can be created. By default, a row is added for each layer comprising the guard ring.

Margin enables you to specify the value that defines the margin that should be applied around the rectangle that is used for creating the tunnel. For example, the *Oxide* in an FGR might be `0.02um` wider than the *Metal*. You might want to apply this amount of overlap through the tunnel. Therefore, you can increase the *Margin* of the *Metal* keepout by `0.02`. By default, the *Margin* values for all *Layer Purposes* in the form are set to `0`. If you enable multiple layer-purposes for creating the tunnel, the margin defined for each layer-purpose is used to create the tunnel on that layer-purpose.

Polygon enables you to draw a polygon to define a tunnel. The total area of the created tunnel is calculated as a sum of the area of the drawn polygon and *Spacing*. When you select this option, the following columns with editable fields are displayed:

Layer Purpose enables you to select the layer-purpose through which the tunnel can be created. It represents the layer-purpose that is removed from the guard ring to create a tunnel. A check box in front of each layer-purpose field enables you to specify more than one layer-purpose at a time though which the tunnel can be created. By default, a row is added for each layer comprising the guard ring.

Margin enables you to specify the value that defines the margin that should be applied around the polygon point list that is used for creating the tunnel. If you enable multiple layer-purposes for creating the tunnel, the margin defined for each layer-purpose is used to create the tunnel on that layer-purpose.

Overlapping Shape enables you to use a shape that overlaps a guard ring to define a tunnel. *Overlapping Shape* is the default shape used for creating a tunnel in a guard ring. The total area of the tunnel created is calculated as a sum of the overlap area and the specified spacing around the overlapping shape. When you select this option, the following fields are displayed:

Use Layer-Purpose of the Shape

- If selected, the layer-purpose of the overlapping shape is used to create the tunnel.

  Use Spacing is the distance around the overlapping shape that is used for creating the tunnel.

  Minimum represents the minimum spacing of the layer-purpose of the overlapping shape. This is the default spacing if the *Use Layer-Purpose of the Shape* check box is selected.

  User Defined enables you to specify a value for the spacing around the overlapping shape to use for creating the tunnel.

- If not selected, you can specify the layer-purpose of the guard ring through which to create the tunnel by using the overlapping shape as the tunnel shape.

  Layer Purpose enables you to select the layer-purpose through which the tunnel can be created. It represents the layer-purpose that is removed from the guard ring to create a tunnel. For the selected layer-purpose to be removed, the overlapping shape should have overlap region with the selected layer-purpose. A check box in front of each layer-purpose field enables you to specify more than one layer-purpose at a time though which the tunnel can be created. By default, a row is added for each layer comprising the guard ring.

  Spacing enables you to specify the spacing around the overlapping shape that is used for creating the tunnel. The default value is the minimum spacing of the layer-purpose selected from the *Layer Purpose* list. In case of table-based `minSpacing` rule, the maximum value defined in the table is used as the default. If you enable

multiple layer-purposes for creating the tunnel, the spacing defined for each layer-purpose is used to create the tunnel on that layer-purpose.

Reset To Defaults button allows to reset the spacing and width value to defaults. This button is enabled only when you select the *Path*, *Rectangle*, or *Polygon* tunnel shape, or when you choose *Overlapping Shape* tunnel shape and deselect the *Use Layer-Purpose of the Shape* check box.

Heal Fluid Object Form

See Healing a Fluid Guard Ring for related information.

Mode

point removes tunnels that exist at the point of click. This is the default selection.

overlapping shape enables you to use a shape that overlaps a guard ring, to remove the tunnels overlapping the shape.

rectangle removes all existing tunnels enclosed within the rectangle you draw.

guard ring removes all tunnels from the guard ring you select.

Install Guard Ring Form

See Installing Fluid Guard Rings for related information.

Technology Library enables you to select a library in which you want to install the guard ring device as a device class.

Name enables you to specify a name for the guard ring device you are installing. The new guard ring device is saved in a copy of the selected technology library in virtual memory. Alternatively, you can select an existing guard ring device defined in your technology library from the *Name* list and update its definition as required. By selecting an existing guard ring device, you can also update its device class definition.

> **Note:** The names of the FGR devices displayed in the *Name* list box are sorted alphanumerically.

View Device Configuration displays the *View Device Configuration* window, which shows the guard ring as it would appear with the current selected layers. This helps you preview the guard ring device without creating one. The button is grayed out until the mandatory layers for a guard ring — diffusion, contact, and metal layers — are defined in the Layers section. The *View* toolbar in the *View Device Configuration* window enables you to zoom in, zoom out, or zoom a selected part of the guard ring.

Delete enables you to remove the guard ring device selected in the *Name* cyclic field from the technology library in virtual memory.

The display of the remaining portion of the form depends on which of the following you select:

- Layers
- Rule
- Parameter Defaults

Important

> *Rule* and *Parameter Defaults* sections remain grayed out until you set up the *Diffusion*, *Contact*, and *Metal* layers in the *Layers* section.

Layers

If you select the *Layers* check box, the lower portion of the form updates to display the fields of the *Layers* section. *Layers* is selected by default when you first open the Install Guard Ring form. If you are creating a guard ring device, the lower portion of the form shows the

Main Layer radio buttons, *Diffusion*, *Contact*, *Metal,* and *Fluid Shape* cyclic fields that you need to set up for a new guard ring. These are the mandatory layers to define a guard ring device. You can optionally set up the implant and well layers.

If you select an existing guard ring from the *Name* list, the lower portion of the form shows the layers defined for the selected device.

Main Layer enables you to select the layer that defines the *Path Width* in the Create Guard Ring and Edit Instance Properties forms. The options available for selection are *Diffusion* and *Metal*. By default, the *Diffusion* radio button is selected.

Diffusion enables you to view or select the layer for the diffusion layer of guard ring. You can define the diffusion layer enclosure value around the contact layer in the Rule section.

Contact enables you to view or select the layer for the via layer of guard ring. You can define the dimensions and spacing of the vias in the Rule section.

Metal enables you to view or select the layer for the metal layer that surrounds the vias. You can define the metal layer enclosure value around the contact layer in the Rule section.

Fluid Shape enables you to define the LPP that Virtuoso uses to represent the fluid shape of the guard ring instances and controls the selectability of the fluid shapes within a guard ring for level-1 editing. For example, when you stretch a guard ring, you interact with the fluid shape. The list contains the predefined y[0-9] LPPs. When the LPPs being used for the fluid shape has a solid fill, you cannot see any other layers in the guard rings. Where this is the case, change the LPP of the Fluid Shape to an LPP that does not have a solid fill.

Add Implant/Well Layers expands the form to display another section at the bottom of the form.

Layer1 enables you to view or select an implant or well layer to enclose the diffusion layer. Defining an implant or well layer is optional. You can set up multiple implant/well layers. Ensure that you select a valid layer. Click the *Add Implant/Well Layers* button as many times as the number of implant or well layers required. You might need to define an implant/well layer around the diffusion, depending on the material you chose for the diffusion layer. For example, if your technology library does not contain n-diffusion or p-diffusion layers, you will need an implant or well layer. You can define the implant/well layer enclosure value around the metal layer in the Rule section.

Pin, if selected, draws the implant or well layer in the guard ring with a pin-like shape and assigns it the same connectivity as the metal layer of the guard ring.

Cover Interiors fills up the interior of the guard ring with the implant or well layer.

Delete enables you to delete the implant or well layer for the current guard ring. Ensure that you delete the non-required layers.

While creating the guard ring, you can enable or disable the implant or well layers in the installed guard ring device by using the Implant Layers subtab in the Create Guard Ring Form.

Rule

If you select the *Rule* check box, the lower portion of the form updates to display the options of the *Rule* section of the form. This section displays the process design rules for the layers set up in the Layers section.

Rule Browser Color specifies the color of the arrow that indicates a dimension in the *Rule Browser* window.

Show Rules opens the *Rule Browser* window, which displays a graphical representation of a generic guard ring device and an arrow indicating the physical area where a dimension selected in the *Rule* section applies. The *View* toolbar in the *Rule Browser* window enables you to zoom in, zoom out, or zoom a selected part of the guard ring.

Note: The guard ring in the *Rule Browser* window is only a representation based on the selected layers; it does not reflect the dimensions of contacts and enclosures.

Use Techfile Default indicates whether the current values in the via parameter fields match the default values in the technology library. If the *Use Techfile Default* check box is on, the values match; if it is off, the values do not match the defaults in the technology library. If the fields are blank, it indicates that there are no default values defined in the technology library. For the already installed devices, the values are populated from the device class.

If you specify or change the value such that it does not match the default value in the technology library, and click anywhere else in the form, the *Use Techfile Default* check box gets deselected to indicate that the value no longer matches the default value in the technology library. To restore the default values from the technology library in the fields, select the *Use Techfile Default* check box. Barring the implant layers, you cannot change the value in fields for other layers to less than the technology library default value. If you change it to a value less than the default, a message displays in the CIW and the value resets to the default. In the case of implant layers, if you update the value to less than the default, a message displays in the CIW but the value is not reset.

***contactLayer* Dimensions** specifies the dimensions of the via on the `layerName` selected in the *Contact* list in the *Layers* section. The default value is the `minWidth` value of the contact layer in the technology library. If you specify a value less than the

technology library default, the value gets reset to the technology library default and a message is displayed in the CIW. Only square vias are supported. If required, you can update this value later while creating the guard ring by using the *Contact Dimension* field in the Contact Settings subtab in the Create Guard Ring Form.

***contactLayer* Spacing** specifies the space between the vias on the `layerName` selected in the *Contact* list in the *Layers* section. The default value is the `minSpacing` value of the contact layer in the technology library. If you specify a value less than the technology library default, the value gets reset to the technology library default and a message is displayed in the CIW. The same spacing value is set in the X and Y directions while installing the device. If required, you can update this value later while creating the guard ring by using the *Contact Spacing* field in the Contact Settings subtab in the Create Guard Ring Form.

Enclosure Rules

This portion of the *Rules* section displays the names of *Diffusion* or *Metal* layers enclosing the *Contact* layer.

diffusionLayer* over *contactLayer specifies the *Diffusion* layer enclosure of the *Contact* layer in the W and L direction, that is the width and length. The default value is the `minOppExtension` / `minExtensionDistance` value of the diffusion layer over contact layer in the technology library. If you specify a value less than the technology library default, the value gets reset to the technology library default and a message is displayed in the CIW. If required, you can update this value later while creating the guard ring by using the *Minimum Diffusion Over Contact* field in the Contact Settings subtab in the Create Guard Ring Form.

metalLayer* over *contactLayer specifies the *Metal* layer enclosure of the *Contact* layer in the W and L direction. The default value is the `minOppExtension` / `minExtensionDistance` value of the metal over contact layer in the technology library. If you specify a value less than the technology library default, the value gets reset to the technology library default and a message is displayed in the CIW. If required, you can update this value later while creating the guard ring by using the *Minimum Metal Over Contact* field in the Contact Settings subtab in the Create Guard Ring Form.

implantLayer/wellLayer* Enclosing *diffusionLayer specifies the *Implant/Well* layer enclosure of the *Diffusion* layer. The default value is the `minOppExtension` / `minExtensionDistance` value of the implant/well layer over diffusion in the technology library. If you specify a value less than the technology library default, it is not reset but a message is displayed in the CIW. If required, you can update this value later while creating the guard ring by using the *Encl* field in the Implant Layers subtab in the Create Guard Ring Form.

Parameter Defaults

If you select the *Parameter Defaults* check box, the lower portion of the form updates to display the options of the *Parameter Defaults* section of the form. This section displays the default parameter values for the guard ring.

Number of contact rows enables you to specify the number of rows of vias. The default is `1`.

Terminal Name enables you to specify the terminal to which you want to associate the guard ring. If you leave the field blank, the default terminal name, `<deviceName>Term`, is assigned.

Pin Name enables you to specify the net on which you want to place the guard ring. If you leave the field blank, the default pin name, `<deviceName>Pin`, is assigned.

B

Fluid Guard Ring Environment Variables

This appendix covers the FGR-specific Layout environment variable names, descriptions, types, and values. For information about the other Virtuoso® Layout Suite L layout editor and graphics editor environment variables, refer to the *Environment Variables* appendix in the *Virtuoso Layout Suite L User Guide*.

The graphic environment variables control the characteristics of the window display and the layout environment variables control how various layout editor commands work. Many graphic environment variables have duplicate layout environment variables. In these cases, the layout variable supersedes the graphic variable unless the graphic variable is stored in the cellview. You can set both graphic and layout environment variables.

Setting Layout Environment Variables

You can set the environment variables in the following three ways:

- Within the .cdsenv File
- Within the .cdsinit File
- In the CIW

.cdsenv File

Add environment variables to the `.cdsenv` file when the settings should be applied while launching Layout L.

Use the following syntax:

```
layout environmentVariableName dataType value
```

For example:

```
layout fgrWrapPlaceAtMinimumDistance boolean t
```

.cdsinit File

Like the `.cdsenv` file, the environment variable settings saved in the `.cdsinit` file get applied when you launch Layout L.

Use the `envSetVal()` command, which has the following syntax, to add environment variables to the `.cdsinit` file:

```
envSetVal("layout" "environmentVariableName" 'dataType value)
```

For example:

```
envSetVal("layout" "fgrWrapPlaceAtMinimumDistance" 'boolean t)
```

Important

The *datatType* should be preceded with a single quote, else the command will not work. Also, if *datatType* is `string`, enclose its *value* within double quotes.

CIW

Use the `envSetVal()` command in the CIW to set an environment variable for the duration of the current session. The syntax is the same as described above for the `.cdsinit` file.

Note: Alternatively, to set environment variables for a single session, you can include the `envSetVal()` command in any Cadence SKILL file that you load.

If you use the CIW to set an environment variable that controls a widget on the currently open form, the implemented settings get reflected only after you close the form and then re-open it.

Displaying the Current Value of an Environment Variable

To determine the current value of any Layout L environment variable, use the following syntax in the CIW.

```
envGetVal("layout" "environmentVariableName")
```

Setting Shell Environment Variables

To set Shell environment variables, use the following syntax before starting the Virtuoso session:

```
setenv environmentVariableName
```

List of Environment Variables

Layout Environment Variables

disableDerivedLayersInWrap

```
layout disableDerivedLayersInWrap boolean { t | nil }
```

Description

(IC6.1.8 Only) Disables the feature that ensures the `minSpacing` rule for a derived layer is considered when an FGR is created around its instances using the *Wrap* mode with the *Place at Minimum Distance* check box as selected.

The default is `nil`.

GUI Equivalent

Command	*Create – Fluid Guard Ring – Wrap* tab
Form Field	*Place At Minimum Distance* (Create Guard Ring Form)

Examples

```
envGetVal("layout" "disableDerivedLayersInWrap")
envSetVal("layout" "disableDerivedLayersInWrap" 'boolean t)
envSetVal("layout" "disableDerivedLayersInWrap" 'boolean nil)
```

Related Topics

List of Environment Variables

fgrWrapPlaceAtMinimumDistance

```
layout fgrWrapPlaceAtMinimumDistance boolean { t | nil }
```

Description

Controls the selection of the *Place at Minimum Distance* check box on the *Wrap* tab of the Create Guard Ring form. When `fgrWrapPlaceAtMinimumDistance` is set to `t`, the *Place At Minimum Distance* check box is selected and the *Enclose by* field is disabled. However, setting the environment variable to `nil`, deselects the check box and makes the *Enclose by* field editable for specifying the guard ring distance from the object.

The default is `t`.

Important

While using customized FGRs, setting the `fgrWrapPlaceAtMinimumDistance` environment variable to `nil` is recommended.

GUI Equivalent

Command	*Create – Fluid Guard Ring – Wrap* tab
Form Field	*Place At Minimum Distance* (Create Guard Ring Form)

Examples

- Deselect the *Place At Minimum Distance* check box and make the *Enclose by* field editable:

  ```
  envSetVal("layout" "fgrWrapPlaceAtMinimumDistance" boolean nil)
  ```

- Return the current value of the `fgrWrapPlaceAtMinimumDistance` environment variable:

  ```
  envGetVal("layout" "fgrWrapPlaceAtMinimumDistance")
  => nil
  ```

Related Topics

List of Environment Variables

fluidGuardRingInstallPath

```
layout fluidGuardRingInstallPath string "alternatePath"
```

Description

Enables you to specify the path from where the FGR-related SKILL files (`vfo*.ils`) are to be loaded. By default, the value string is empty, in which case these SKILL files are loaded from the following default release installation directory:

```
<install_dir>/tools/dfII/etc/vfo
```

Note: This method of specifying the path from where to load the FGR-related SKILL files (`vfo*.ils`) is not recommended for customized FGRs. If you want to use it, contact your Cadence® Customer Support representative.

The default is `""` (null string).

GUI Equivalent

None

Examples

Load the FGR-related SKILL files from a path other than the default release installation directory:

```
layout fluidGuardRingInstallPath string /grid/cic/tool.lnx86/dfII/etc/vfo
```

Related Topics

List of Environment Variables

grEnclosedBy

```
layout grEnclosedBy float any_positive_floating_point_number
```

Description

Initializes the *Enclose by* field on the Create Guard Ring form.

You can replace the default value with another floating point number by updating this environment variable. Otherwise, change the value for the *Enclose by* field by typing in the field on the Create Guard Ring form.

The default value is `0.0`.

GUI Equivalent

None

Examples

```
envGetVal("layout" "grEnclosedBy")
envSetVal("layout" "grEnclosedBy" 'float 1.0)
```

Related Topics

List of Environment Variables

grMode

```
layout grMode string { Rectangular | Rectilinear }
```

Description

Sets the default FGR creation type in *Wrap* mode.

By default, this environment variable is set to `Rectilinear`. Therefore, when you launch the Create Guard Ring form, on the *Wrap* tab, the `Rectilinear` radio button is selected.

However, when you reset this environment variable to `Rectangular`, the `Rectangular` radio button appears as selected on the *Wrap* tab.

This environment variable is useful when you mostly create `Rectangular` type of FGR in *Wrap* mode and want that be selected as the default creation type each time you access the Create Guard Ring form.

The default is `Rectilinear`.

GUI Equivalent

None

Examples

```
envGetVal("layout" "grMode")
envSetVal("layout" "grMode" 'string "Rectangular")
envSetVal("layout" "grMode" 'string "Rectilinear")
```

Related Topics

List of Environment Variables

keepGuardRingEndsConnected

```
layout keepGuardRingEndsConnected boolean { t | nil }
```

Description

Controls whether a guard ring with touching ends will stay connected (`t`) or will result in opening up the guard ring (`nil`) during the *Stretch* and *Quick Align* command operations.

The default is `nil`.

GUI Equivalent

None

Examples

```
envGetVal("layout" "keepGuardRingEndsConnected")
envSetVal("layout" "keepGuardRingEndsConnected" 'boolean t)
envSetVal("layout" "keepGuardRingEndsConnected" 'boolean nil)
```

Related Topics

List of Environment Variables

vfoGRHideDevicesInCreateForm

```
layout vfoGRHideDevicesInCreateForm string "device_names"
```

Description

Specifies the FGR devices to be omitted from the *Device* drop-down list of the Create Guard Ring form. You can also use the `vfoGRHideDeviceInForms` property of the `tfcDefineDeviceProp` construct of the technology file to control the display of devices in the Create Guard Ring form.

GUI Equivalent

None

Examples

```
envGetVal("layout" "vfoGRHideDevicesInCreateForm")
envSetVal("layout" "vfoGRHideDevicesInCreateForm" 'string "M1_M2")
envSetVal("layout" "vfoGRHideDevicesInCreateForm" 'string "guardring1")
```

Related Topics

List of Environment Variables

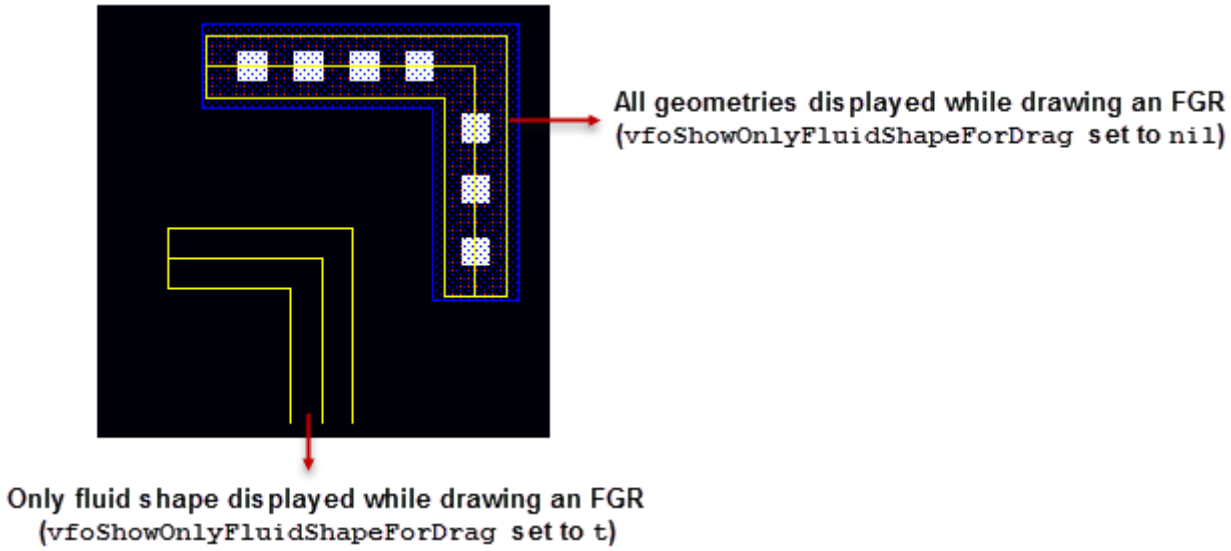
vfoShowOnlyFluidShapeForDrag

```
layout vfoShowOnlyFluidShapeForDrag boolean { t | nil }
```

Description

Controls the display of the fluid shape and other geometries like metal layer, diffusion layer, and contacts while drawing an FGR on the layout canvas.

When `vfoShowOnlyFluidShapeForDrag` is set to `nil`, all geometries including the fluid shape are visible as you draw the FGR on the layout canvas. However, setting the environment variable to `t` displays only the fluid shape. The figure below shows the difference in the two approaches.



The default is `nil`.

GUI Equivalent

None

Examples

```
envGetVal("layout" "vfoShowOnlyFluidShapeForDrag")
envSetVal("layout" "vfoShowOnlyFluidShapeForDrag" 'boolean t)
```

```
envSetVal("layout" "vfoShowOnlyFluidShapeForDrag" 'boolean nil)
```

Related Topics

List of Environment Variables

Shell Environment Variables

FGR_CACHE_AUTO_CLEANUP

Description

Controls whether automatic cache cleaning on the design first open is enabled (`ON`) or not (`OFF`).

Default Value: `ON`

FGR_CACHE_TIMESTAMP_CHECK

Description

If the super-master is re-installed after saving the sub-master cache, this environment variable decides if the timestamp mismatch should be ignored and the sub-master should be read from the cache or the sub-master should be evaluated using the current Virtuoso version.

Default Value: `OFF`

FGR_REEVAL_ON_CORRUPT_CACHE

Description

Evaluates pcells with the current Virtuoso version incase the sub-master image is not readable from the `cache.pcl` file. To enable this environment variable, set it to `1`/`t`/`T`/`true`/`TRUE`.

FGR_MIN_VIA_SPACING_ENABLED

Description

Enables the `minViaSpacing` value to be considered to compute the contact spacing during installation and creation of an FGR. The `minViaSpacing` value is used only if the variable is set to `1`/`t`/`T`/`true`/`TRUE`.

C

Hiding Fluid Guard Ring Devices

You can hide the desired devices from being listed in the *Name* drop-down list box on the Install Guard Ring form and/or the *Device* drop-down list box on the Create Guard Ring form. This prevents CAD engineers from making any accidental changes to device definitions or layout designers from creating instances of certain FGR devices.

To hide a device from the desired form, in the `tfcDefineDeviceProp` section of the technology file, set the `vfoGRHideDeviceInForms` property to one of the following values:

Value	Hides the specified device from
`installAndCreateForm`	Install Guard Ring form Create Guard Ring form
`installForm`	Install Guard Ring form
`createForm`	Create Guard Ring form

Use the following syntax to set the `vfoGRHideDeviceInForms` property:

```
tfcDefineDeviceProp(
  (<viewName> "<deviceName1>" vfoGRHideDeviceInForms "<propValue>")
  (<viewName> "<deviceName2>" vfoGRHideDeviceInForms "<propValue>")
  ...
)
```

Alternatively, you can use the `dbOpenCellViewByType` SKILL function in the CIW followed by the `dbCreateProp` or `dbReplaceProp` SKILL functions. Then, use the `dbSave` SKILL function to save the modified property.

Note: While using these SKILL functions, ensure that you open the FGR device in *append* mode instead of *write* mode. If you open the FGR device in *write* mode, it will get corrupted.

The following sections explain how to hide and make the hidden FGR devices visible:

- Hiding a Device from Install Guard Ring Form

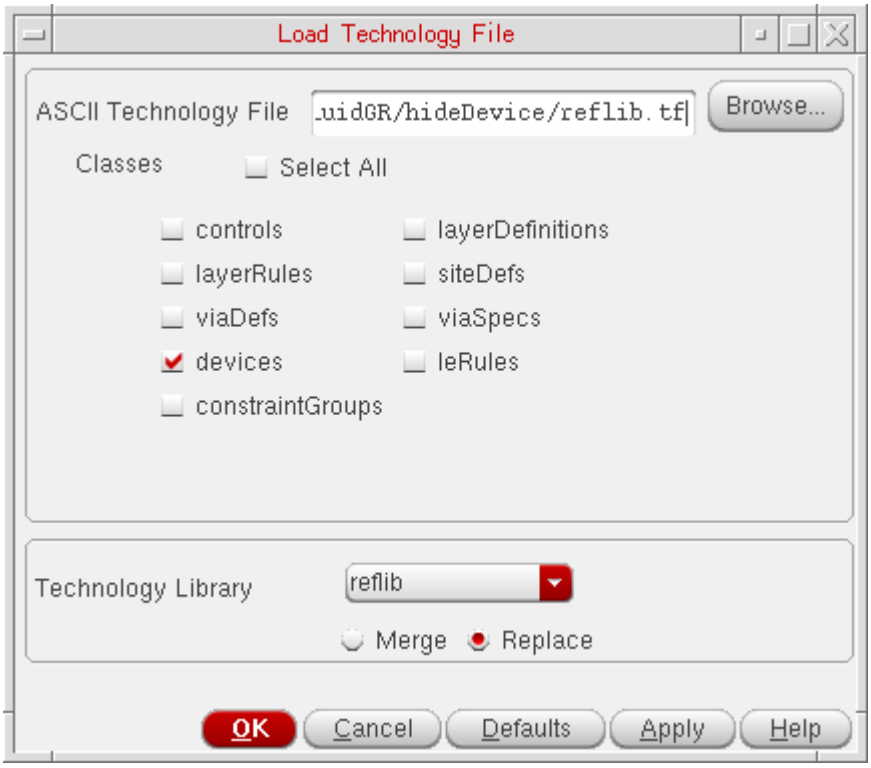
Hiding a Device from Install Guard Ring Form

To hide a specific FGR device from the Install Guard Ring form,

1. Open the technology file that contains the definitions of the devices you want to hide.
2. Set the `vfoGRHideDeviceInForms` property in the `tfcDefineDeviceProp` section for each device that needs to be hidden. You can use one of the following values for this property:
   - `installAndCreateForm`
   - `installForm`

   If you specify any other value or empty string, the specified device continues to be displayed on the form.
3. Save the technology file.
4. From the CIW, choose *Tools – Technology File Manager*. The Technology Tool Box form appears.
5. In the *Manager* group box, click *Load*.

The Load Technology File form appears.



6. Browse and select the *ASCII Technology File* that you updated for hiding the desired FGR devices.
7. Select the *devices* check box to load the updated `tfcDefineDeviceProp` section.

   Note: You can also select other relevant check boxes based on other sections you might have updated in the technology file. Selecting the *Select All* check box loads all *Classes* defined in the specified technology file.
8. Select the *Technology Library* from the drop-down list box.
9. Select to *Merge* or *Replace* the updates while loading the technology file.
10. Click *OK*.

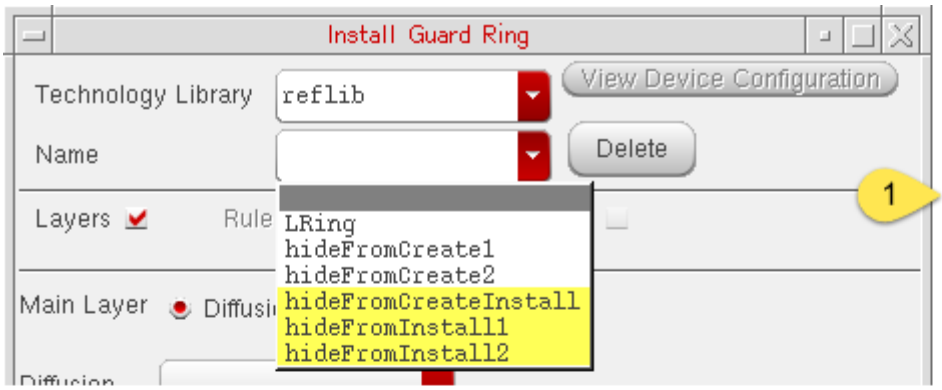
Now, when you open the Install Guard Ring form, the devices for which you had set the `vfoGRHideDeviceInForms` property do not appear listed with the other devices. The example in the section below illustrates the steps listed above.

Example of Hiding Devices from Install Guard Ring Form

Suppose you have a technology file `reflib.tf` that contains definitions for the following devices: `LRing`, `hideFromCreate1`, `hideFromCreate2`, `hideFromCreateInstall`, `hideFromInstall1`, and `hideFromInstall2`.

You now want to hide the following FGR devices from the Install Guard Ring form: `hideFromCreateInstall`, `hideFromInstall1`, and `hideFromInstall2`.

The following images show the different stages of hiding these FGR devices:

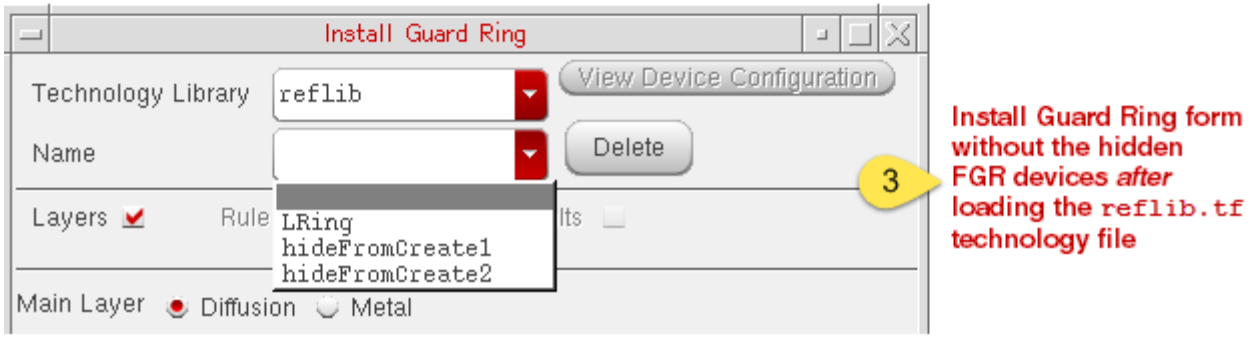


2 **Setting the `vfoGRHideDeviceInForms` property in the `reflib.tf` technology file to hide highlighted FGR devices**

```
tfcDefineDeviceProp(
; (viewName        deviceName          propName           propValue)
  (layout          "hideFromCreateInstall" vfoProtocolClass   "vfoSfImplClass")
  (layout          "hideFromCreateInstall"  vfoGRHideDeviceInForms  "installAndCreateForm")
)

tfcDefineDeviceProp(
; (viewName        deviceName          propName           propValue)
  (layout          "hideFromInstall1" vfoProtocolClass   "vfoSfImplClass")
  (layout          "hideFromInstall1"  vfoGRHideDeviceInForms  "installForm")
)

tfcDefineDeviceProp(
; (viewName        deviceName          propName           propValue)
  (layout          "hideFromInstall2" vfoProtocolClass   "vfoSfImplClass")
  (layout          "hideFromInstall2"  vfoGRHideDeviceInForms  "installForm")
)
```



Hiding a Device from Create Guard Ring Form

To hide a specific FGR device from the Create Guard Ring form:

1. Open the technology file that contains the definitions of the devices you want to hide.
2. Set the `vfoGRHideDeviceInForms` property in the `tfcDefineDeviceProp` section for each device that needs to be hidden. You can use one of the following values for this property:

   - `installAndCreateForm`
   - `createForm`

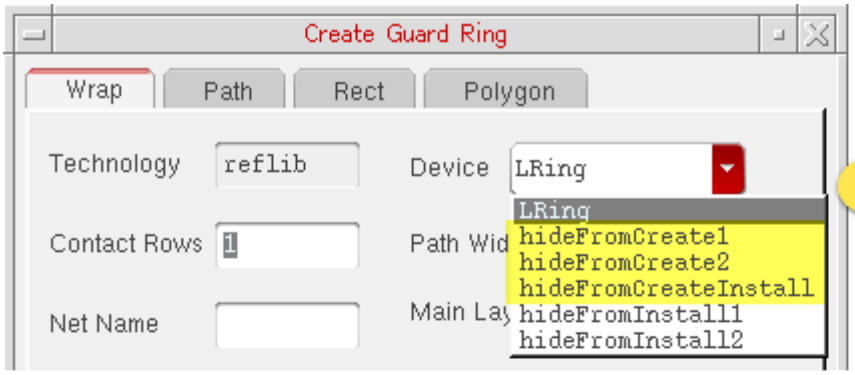
   If you specify any other value or empty string, the specified FGR device continues to be displayed on both the forms.
3. Repeat steps 3 – 10 described in the Hiding a Device from Install Guard Ring Form section above.

Now, when you open the Create Guard Ring form, the devices for which you had set the `vfoGRHideDeviceInForms` property do not appear listed with the other devices. The examples in the sections below illustrate the above-listed steps.

Example of Hiding Devices from Create Guard Ring Form

The technology file `reflib.tf` contains definitions for the following devices: `LRing`, `hideFromCreate1`, `hideFromCreate2`, `hideFromCreateInstall`, `hideFromInstall1`, and `hideFromInstall2`. Suppose, you want to hide FGR devices named `hideFromCreate1` and `hideFromCreate2` from the Create Guard Ring form. In addition, you want to hide `hideFromCreateInstall` such that it is not available on both

Install Guard Ring and Create Guard Ring forms. The following images show the different stages of hiding these FGR devices:



2 **Setting the `vfoGRHideDeviceInForms` property in the `reflib.tf` technology file to hide highlighted FGR devices**

```
tfcDefineDeviceProp(
; (viewName        deviceName          propName           propValue)
  (layout          "hideFromCreateInstall" vfoProtocolClass  "vfoSfImplClass")
  (layout          "hideFromCreateInstall"  vfoGRHideDeviceInForms  "installAndCreateForm")
)
tfcDefineDeviceProp(
; (viewName         deviceName          propName           propValue)
  (layout          "hideFromCreate1"   vfoProtocolClass  "vfoSfImplClass")
  (layout          "hideFromCreate1"   vfoGRHideDeviceInForms  "createForm")
)
tfcDefineDeviceProp(
; (viewName         deviceName          propName           propValue)
  (layout          "hideFromCreate2"   vfoProtocolClass  "vfoSfImplClass")
  (layout          "hideFromCreate2"   vfoGRHideDeviceInForms  "createForm")
)
```

Making a Hidden Device Visible on Install and Create Forms

To make a previously hidden FGR device visible on either the Install Guard Ring form or the Create Guard Ring form, remove the associated `vfoGRHideDeviceInForms` property from the technology file and then reload the file in Virtuoso.

Alternatively, use the `dbDeletePropByName` SKILL function in the CIW and then run the `dbSave` SKILL function to set the modified setting.

D

Loading VFO Infrastructure in Third-Party Tools

The customized fluid guard rings (FGRs) are derived from the base classes defined in the Virtuoso Fluid Object (VFO) infrastructure. Therefore, the related SKILL and SKILL++ code files have a dependency on the `vfo*` context and class definitions. Typically, third-party tools cannot evaluate the customized FGRs because they do not have access to the SKILL and SKILL++ code written by a PDK developer specifically for such FGRs. Also, the approach of loading the SKILL and SKILL++ code from the `libInit.il` initialization file is not user intuitive. This might lead to display of different types of error and warning messages while loading the customized FGR classes and methods. For example, unknown class found while defining a user-defined method or unable to restore a particular method.

Therefore, to allow third-party tools to read customized FGRs, you need to manually load the `.il` and context files that have the VFO infrastructure definitions, as explained in this chapter.

Note: A PDK developer handles the loading of files that contain the code for any customized FGR. Therefore, if you need assistance in configuring or troubleshooting the load sequence of customized FGRs, contact your PDK provider for information.

Run the following steps to manually load the VFO infrastructure:

1. Create a SKILL file that contains the lines of code given in the Procedure for Initializing Customized FGR Devices section and save it with a filename having the `.il` extension. For example, `load_vfo_context_and_files.il`
2. Load this new SKILL file by adding the following lines of code in the `libInit.il` file:

   ```
   load("load_vfo_context_and_files.il")
   ```
3. In the same `libInit.il` file, call the `load_vfo_context_and_files` SKILL procedure by using the following syntax:

   ```
   load_vfo_context_and_files (cxt_path  ils_path)
   ```

   Here,

   - `cxt_path` is the string specifying the location of the `vfo.cxt` file.

     In the Virtuoso installation, the `cxt_path` is:

```
<install_dir>/tools/dfII/etc/context
```

- `ils_path` is the string specifying the location of the `vfo*.ils` files.

  In the Virtuoso installation, the `ils_path` is:

  ```
  <install_dir>/tools/dfII/etc/vfo
  ```

Procedure for Initializing Customized FGR Devices

Add the following procedure to a file, such as, `load_vfo_context_and_files.il`:

```
procedure(load_vfo_context_and_files(vfoCxtPath ilsPath)
  let((vfoLoadSeqFilePath fileName )

    unless(isContextLoaded("vfo")
      loadContext(sprintf(nil "%s/vfo.cxt" vfoCxtPath))
    );;unless

    vfoLoadSeqFilePath = strcat(ilsPath "/vfoInitialize.ils")

    if(isFileName(vfoLoadSeqFilePath)
    then
      when(!isCallable('vfoGRGeometry)
        load(vfoLoadSeqFilePath)
        foreach(fileName vfoGetFileListWithLoadSequence()
          load(sprintf( nil "%s/%s" ilsPath fileName))
          printf("done loading %s/%s\n" ilsPath fileName)
        )
      );;when

  else
      when(!isCallable('vfoGRGeometry)
        foreach(fileName (list
          "vfoMessageIds.ils"
          "vfoAbstractClass.ils"
          "vfoAddOns.ils"
          "vfoApi.ils"
          "vfoAlgClass.ils"
          "vfoUtils.ils"
          "vfoShapeData.ils"
          "vfoSfShapeData.ils"
          "vfoSf.ils"
```

```
          "vfoSfFilling.ils"
          "vfoGuardRing.ils"
          "vfoGrShrinkWrap.ils"
          "vfoGuardRingPreview.ils")
          load(sprintf( nil "%s/%s" ilsPath fileName))
          printf("done loading %s/%s\n" ilsPath fileName)
        );;foreach
      );;when
    );;if
  );;let
);;load_vfo_context_and_files
```

E

Geometry Changes in Virtuoso Releases

This chapter contains information about the geometry changes that can occur in fluid guard ring (FGR) instances when you migrate to newer releases of Virtuoso. FGRs are Pcells that Virtuoso dynamically evaluates; therefore, bug fixes and enhancements to the FGR code can result in geometry changes to the FGR instances for a given set of parameters.

Note: Each entry provides the release in which the problem was introduced, provides a short description of the problem, and lists the releases affected and the releases in which the problem was fixed.

IC6.1.5 ISR15

Fluid guard rings losing contacts in IC6.1.5 ISR15 and ISR16

IC6.1.5 ISR15 introduced a feature that automatically removes corner contacts from fluid guard ring instances when the user specifies a metal or diffusion enclosure value that is less than the larger value of the associated `minOppExtension` constraint. In some scenarios, Virtuoso removed more contacts than necessary.

Affected releases:

- IC6.1.5 ISR15
- IC6.1.5 ISR16
- IC6.1.6

Fixed in:

- IC6.1.5 ISR17
- IC6.1.6 ISR1
- ICADV12.1 ISR3

IC6.1.6 ISR3 and ICADV12.1 ISR5

Create Guard Ring form does not retain previously entered values

The Create Guard Ring form for an existing FGR device did not retain the previously entered values when those were technology rule complaint, but were less than the default values specified in the Install Guard Ring form.

Affected releases:

- IC6.1.5 ISR16
- IC6.1.5 ISR17
- IC6.1.6
- IC6.1.6 ISR1
- IC6.1.6 ISR2
- ICADV12.1 ISR3
- ICADV12.1 ISR4

Fixed in:

- IC6.1.6 ISR3
- ICADV12.1 ISR5

DRC violation for the implant layer

When the removal of corner contacts from FGR instances feature was introduced in IC6.1.5 ISR15, the allowable limit for the enclosure of the implant layer was reduced to the minimum of the two `minOppExtension` values, one for each side. This in turn induced DRC violation for the implant layer.

After the fix for this undesirable consequence, the minimum allowed enclosure value for the implant layer will be the maximum of the two `minOppExtension` values given for the diffusion and implant layer.

Affected releases:

- IC6.1.5 ISR15
- IC6.1.5 ISR16
- IC6.1.6
- IC6.1.6 ISR1
- IC6.1.6 ISR2
- ICADV12.1 ISR3
- ICADV12.1 ISR4

Fixed in:

- IC6.1.6 ISR3
- ICADV12.1 ISR5

F

FGR Dual Evaluation Capability

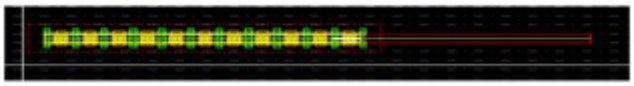
The Shell environment variable, `FGR_DUAL_EVAL_ON`, enables the FGR dual evaluation capability. When the FGR dual evaluation capability is enabled, we can differentiate between drag and drag-finished state of the command and accordingly execute the FGR drawing code. The return value of the `vfoIsCommandInDragMode` function can be used to determine which evaluation code will be used during FGR creation or editing.

The example below depicts the FGR dual evaluation capability and shows the effect of the drag operation on geometries when the FGR dual evaluation capability is enabled and disabled.

When the `FGR_DUAL_EVAL_ON` Shell environment variable and the `vfoIsCommandInDragMode` function are not used (the FGR dual evaluation capability is disabled) full geometry evaluation occurs during the drag operation, as shown in the figure below.

Without `vfoIsCommandInDragMode` function (full geometry evaluation during drag)

When the `FGR_DUAL_EVAL_ON` Shell environment variable and the `vfoIsCommandInDragMode` function are used together (the FGR dual evaluation capability is enabled) a custom code can be written to control the drawing of FGR during the drag and when the drag finishes. The FGR evaluation happens with limited number of geometries, such as no contacts drawn, during the drag operation as shown in the figure below. Full geometry evaluation occurs when the drag operation completes.



With `vfoIsCommandInDragMode` function (only fluid shape evaluation during drag)