

Virtuoso Studio What's New

Product Version IC23.1

November 2023

© 2023 Cadence Design Systems, Inc.
Printed in the United States of America.

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

Open SystemC, Open SystemC Initiative, OSCI, SystemC, and SystemC Initiative are trademarks or registered trademarks of Open SystemC Initiative, Inc. in the United States and other countries and are used with permission.

Trademarks: Trademarks and service marks of Cadence Design Systems, Inc. (Cadence) contained in this document are attributed to Cadence with the appropriate symbol. For queries regarding Cadence's trademarks, contact the corporate legal department at the address shown above or call 800.862.4522.

All other trademarks are the property of their respective holders.

Restricted Permission: This publication is protected by copyright law and international treaties and contains trade secrets and proprietary information owned by Cadence. Unauthorized reproduction or distribution of this publication, or any portion of it, may result in civil and criminal penalties. Except as specified in this permission statement, this publication may not be copied, reproduced, modified, published, uploaded, posted, transmitted, or distributed in any way, without prior written permission from Cadence. Unless otherwise agreed to by Cadence in writing, this statement grants Cadence customers permission to print one (1) hard copy of this publication subject to the following conditions:

1. The publication may be used only in accordance with a written agreement between Cadence and its customer.
2. The publication may not be modified in any way.
3. Any authorized copy of the publication or portion thereof must include all original copyright, trademark, and other proprietary notices and this permission statement.
4. The information contained in this document cannot be used in the development of like products or software, whether for internal or external use, and shall not be used for the benefit of any other party, whether or not for consideration.

Disclaimer: Information in this publication is subject to change without notice and does not represent a commitment on the part of Cadence. Except as may be explicitly set forth in such agreement, Cadence does not make, and expressly disclaims, any representations or warranties as to the completeness, accuracy or usefulness of the information contained in this document. Cadence does not warrant that use of such information will not infringe any third party rights, nor does Cadence assume any liability for damages or costs of any kind that may result from use of such information. Cadence is committed to using respectful language in our code and communications. We are also active in the removal and replacement of inappropriate language from existing content. This product documentation may however contain material that is no longer considered appropriate but still reflects long-standing industry terminology. Such content will be addressed at a time when the related software can be updated without end-user impact.

Restricted Rights: Use, duplication, or disclosure by the Government is subject to restrictions as set forth in FAR52.227-14 and DFAR252.227-7013 et seq. or its successor.

Contents

Preface	11
Licensing Requirements	11
Release README.txt	11
Known Problems and Solutions	12
Additional Learning Resources	12
Rapid Adoption Kits	12
Help and Support Facilities	12
Customer Support	13
Feedback about Documentation	13
Typographic and Syntax Conventions	14
1	15
Overview of Virtuoso Studio IC23.1	15
Related Topics	16
Products Enhanced in Virtuoso Studio IC23.1	17
What's New in Other Products	18
Key Features in Virtuoso Studio IC23.1	19
Products and Features Removed from Virtuoso Studio IC23.1	20
Documentation Updates in Virtuoso Studio IC23.1	20
Restructured Documentation	20
Removed Documentation	21
Help and Support Enhancements	21
Doc Assistant — A New Documentation Viewer for Cadence Tools	21
Launching Doc Assistant	22
We Value Your Feedback!	23
Related Topics	23
Training Courses	24
Videos	24
Blogs	24
Virtuoso Release Announcement Blog Series	25
Virtuoso Studio Blog Series	25
Virtuosity Blog Series	25
Virtuoso Video Diary Blog Series	26

Knowledge Booster Training Bytes Blog Series	26
Virtuoso Meets Maxwell Blog Series	26
Spectre Tech Tips Blog Series	27
2	28
What's New in Analog Library	28
IC23.1 ISR1	28
IC23.1 Base	28
3	30
What's New in Cadence Application Infrastructure	30
IC23.1 Base	30
4	31
What's New in Cadence Library Manager	31
IC23.1 ISR3	31
IC23.1 ISR2	32
5	33
What's New in Cadence SKILL IDE	33
IC23.1 ISR1	33
6	34
What's New in Component Description Format	34
IC23.1 Base	34
7	36
What's New in Design Data Translators	36
IC23.1 ISR3	36
IC23.1 ISR2	37
IC23.1 Base	37
8	39
What's New in Open Simulation System	39
IC23.1 Base	39
9	41
What's New in Virtuoso Abstract Generator	41
IC23.1 ISR2	41
IC23.1 Base	41
10	43

What's New in Virtuoso ADE Explorer and Virtuoso ADE Assembler	43
IC23.1 ISR3	43
IC23.1 ISR2	46
IC23.1 ISR1	48
IC23.1 Base	49
Removed Products	53
Removed Features	53
11	55
What's New in Virtuoso ADE Verifier	55
IC23.1 ISR3	55
IC23.1 ISR1	56
IC23.1 Base	57
Removed Features	58
12	59
What's New in Virtuoso Automated Placement and Routing Flow	59
IC23.1 ISR3	59
IC23.1 ISR2	60
IC23.1 ISR1	63
IC23.1 Base	67
13	72
What's New in Virtuoso Concurrent Layout	72
IC23.1 Base	72
14	73
What's New in Virtuoso Custom Design Migrate	73
IC23.1 ISR3	73
IC23.1 ISR2	73
IC23.1 ISR1	74
IC23.1 Base	74
15	76
What's New in Virtuoso Design Intent	76
IC23.1 ISR3	76
IC23.1 ISR2	77
IC23.1 ISR1	77
IC23.1 Base	77

16	79
What's New in Virtuoso Design Planning and Analysis	79
IC23.1 ISR1	79
IC23.1 Base	79
17	81
What's New in Virtuoso Design Review	81
IC23.1 ISR3	81
IC23.1 ISR2	81
IC23.1 ISR1	82
IC23.1 Base	82
18	83
What's New in Virtuoso Design Rule Driven Editing	83
IC23.1 ISR2	83
IC23.1 Base	84
19	85
What's New in Virtuoso Electrically Aware Design	85
IC23.1 ISR2	85
IC23.1 ISR1	86
IC23.1 Base	87
20	88
What's New in Virtuoso Floorplanner	88
IC23.1 ISR3	88
IC23.1 ISR2	89
IC23.1 ISR1	90
IC23.1 Base	91
21	95
What's New in Virtuoso Fluid Guard Ring	95
IC23.1 ISR1	95
IC23.1 Base	96
22	98
What's New in Virtuoso Hierarchy Editor	98
IC23.1 ISR1	98
IC23.1 Base	98

23	100
What's New in Virtuoso Import Tools	100
IC23.1 Base	100
24	101
What's New in Virtuoso Interactive and Assisted Routing	101
IC23.1 ISR2	101
IC23.1 ISR1	102
IC23.1 Base	102
25	104
What's New in Virtuoso Layout Suite EXL	104
IC23.1 ISR2	104
IC23.1 ISR1	105
26	106
What's New in Virtuoso Layout Suite MXL	106
IC23.1 Base	106
27	107
What's New in Virtuoso Layout Suite XL	107
IC23.1 ISR3	107
IC23.1 ISR2	109
IC23.1 ISR1	111
IC23.1 Base	113
28	118
What's New in Virtuoso Module Generator	118
IC23.1 ISR3	118
IC23.1 Base	119
Removed Features	119
29	121
What's New in Virtuoso Multi-Technology Solution	121
IC23.1 ISR3	121
IC23.1 ISR2	121
IC23.1 Base	122
30	123
What's New in Virtuoso Multi-Patterning Technology	123

IC23.1 Base	123
31	124
What's New in Virtuoso NC-Verilog Environment	124
IC23.1 ISR2	124
32	125
What's New in Virtuoso Parameterized Cell	125
IC23.1 ISR1	125
33	126
What's New in Virtuoso Parasitic Aware Design	126
IC23.1 ISR3	126
34	127
What's New in Virtuoso Photonics Solution	127
IC23.1 ISR1	127
IC23.1 Base	127
35	129
What's New in Virtuoso RF Solution	129
IC23.1 ISR1	129
IC23.1 Base	129
36	132
What's New in Virtuoso Schematic Editor	132
IC23.1 ISR3	132
IC23.1 ISR2	132
IC23.1 ISR1	133
IC23.1 Base	134
37	138
What's New in Virtuoso Simulation Driven Interactive Routing	138
IC23.1 ISR2	138
IC23.1 Base	139
38	141
What's New in Virtuoso Studio Design Environment	141
IC23.1 ISR1	141
IC23.1 Base	142
39	145


What's New in Virtuoso Studio Design Environment SKILL	145
IC23.1 ISR3	145
IC23.1 ISR2	145
IC23.1 Base	146
40	147
What's New in Virtuoso Studio Licensing and Configuration	147
IC23.1 Base	147
Virtuoso Layout Suite MXL	147
41	148
What's New in Virtuoso SystemVerilog Netlister	148
IC23.1 ISR1	148
42	149
What's New in Virtuoso Technology Data	149
IC23.1 ISR3	149
IC23.1 ISR2	150
IC23.1 ISR1	151
43	153
What's New in Virtuoso Unified Custom Constraints	153
IC23.1 ISR1	153
44	154
What's New in Virtuoso Visualization and Analysis XL	154
IC23.1 ISR3	154
IC23.1 ISR1	155
IC23.1 Base	155
45	157
What's New in Voltage Dependent Rules Flow	157
IC23.1 ISR1	157
46	158
What's New in Voltus-Fi Custom Power Integrity Solution L	158
IC23.1 Base	158
Removed Products	158
47	159
What's New in Voltus-Fi Custom Power Integrity Solution XL	159

IC23.1 Base	159
Removed Products	159
48	160
What's New in Voltus-XFi	160
IC23.1 ISR3	160
IC23.1 Base	163

Preface

This document provides a high-level overview of the new and enhanced features added for the IC23.1 Base and subsequent incremental releases (ISRs). Each chapter in this document maps to a Virtuoso application and may contain one or more of the following sections:

- New and enhanced features
- Removed features
- Documentation updates, including a list of new and updated videos

 The table of contents lists only those releases in which new features or enhancements have been introduced.

This preface contains the following topics:

- [Licensing Requirements](#)
- [Related Documentation](#)
- [Additional Learning Resources](#)
- [Customer Support](#)
- [Feedback about Documentation](#)
- [Typographic and Syntax Conventions](#)

Licensing Requirements

For information about licensing in Virtuoso Studio design environment, see [Virtuoso Software Licensing and Configuration Guide](#).

Release README.txt

The `README.txt` for the IC23.1 release will be made available for the base release.

- The contents of `README.txt` file are organized under the following heads:
 - Supported Platforms and Operating Systems
 - Cadence Products and Standalone Software Compatible with the IC Release

- Installation Information
- Accessing Product Documentation
- Contact Cadence
- CCRs Fixed in *<release information>* (this comprises a list of customer-reported bugs that have been fixed in the current release)
- To view the contents of `README.txt`, do one of the following:
 - Locate the file in the installation directory.
 - Log on to downloads.cadence.com and follow the 'Get release information' instructions.

Known Problems and Solutions

For information about Known Problems and Solutions (KPNS), see [Known Problems and Solutions in Virtuoso Studio IC23.1](#).

Additional Learning Resources

Rapid Adoption Kits

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

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

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 Doc Assistant Home 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 Home Page is displayed by default when you open Doc Assistant 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.

For more information, see [Getting Help](#) in *Virtuoso Studio 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 Doc Assistant

You can also submit feedback by using the following methods:

- In the Doc Assistant 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 document.

<i>text</i>	Indicates names of manuals, menu commands, buttons, and fields.
<code>text</code>	Indicates text that you must type exactly as presented. Typically used to denote command, function, routine, or argument names that must be typed literally.
<i>z_argument</i>	Indicates text that you must replace with an appropriate argument value. The prefix (in this example, <i>z_</i>) 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 <i>t_arg</i>]	
	Denotes a <i>key argument</i> . 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.

Overview of Virtuoso Studio IC23.1



Our custom design solution just got better with Virtuoso® Studio ushering into the future of custom or analog design. Leveraging 30 years of industry experience to provide broader support for RF, photonics, mixed-signal, and advanced heterogeneous designs. The new AI-powered Virtuoso Studio advances productivity via automation and innovative features, reimagined infrastructure and new levels of integration that stretch beyond classic design boundaries.

Virtuoso Studio offers the same great experience you are accustomed to, but with added benefits to support accelerating design complexity:

- **Improved infrastructure to overcome the exponential changes in design size and complexity**
 - Scalability and improved performance for 100s to 10,000s of simulations
 - Advancements in cloud support
 - Enhanced performance for core editing in Virtuoso Schematic Editor and Virtuoso Layout Suite
- **Deeper integration to provide access to the best-in-class platforms and engines within the Cadence portfolio**
 - Continued addition of the latest Spectre® advancements to the Virtuoso ADE Suite, including
 - Integration with the new Fast Monte Carlo engine
 - Spectre FX simulator support for transistor-level and AMS designs
 - Latest advances in Spectre X RF Option

- Tighter integration across Cadence platforms, including
 - Allegro® PCB tools to provide support for 2.5D and 3D heterogeneous design implementation and analysis using the EMX®, Clarity™ and Celsius™ engines
 - Innovus™ Implementation System NanoRoute™ and GigaPlace™ engines natively integrated within the Virtuoso layout cockpit to provide quick turn around on standard cell designs
 - Virtuoso iPegasus DRC that integrates the Pegasus™ engine for DRC and Fill to provide an interactive sign-off quality verification experience as part of Virtuoso Layout Suite, without any additional licenses
- **Tool innovation for higher productivity**
 - Layout automation techniques, including new device- and block-level auto routing solutions
 - New circuit design optimization algorithms incorporating some of the latest techniques in the industry
 - Support for the latest advanced nodes down to 2nm, including support for the gate-all-around (GAA) topology in layout
- **Generative AI for design migration**
 - Foundry-supported solutions to ease the burden of migration of schematics and layouts
 - Ability to quickly re-center and validate designs post migration to achieve aggressive time-to-market goals
 - AI-enabled tools to transform existing IP for next-generation designs

Related Topics

[Products Enhanced in Virtuoso Studio IC23.1](#)

[Key Features in Virtuoso Studio IC23.1](#)

[Products and Features Removed from Virtuoso Studio IC23.1](#)

[Documentation Updates in Virtuoso Studio IC23.1](#)

[Help and Support Enhancements](#)

- [Doc Assistant — A New Documentation Viewer for Cadence Tools](#)

- [Training Courses](#)
- [Videos](#)
- [Blogs](#)

Products Enhanced in Virtuoso Studio IC23.1

- [Analog Library](#)
- [Cadence Application Infrastructure](#)
- [Cadence Library Manager](#)
- [Cadence SKILL IDE](#)
- [Component Description Format](#)
- [Design Data Translators](#)
- [Open Simulation System](#)
- [Virtuoso Abstract Generator](#)
- [Virtuoso ADE Explorer and ADE Assembler](#)
- [Virtuoso ADE Verifier](#)
- [Virtuoso Automated Device Placement and Routing Flow](#)
- [Virtuoso Concurrent Layout](#)
- [Virtuoso Custom Design Migrate](#)
- [Virtuoso Design Intent](#)
- [Virtuoso Design Planning and Analysis](#)
- [Virtuoso Design Rule Driven Editing](#)
- [Virtuoso Electrically Aware Design](#)
- [Virtuoso Floorplanner](#)
- [Virtuoso Fluid Guard Ring](#)
- [Virtuoso Hierarchy Editor](#)
- [Virtuoso Import Tools](#)
- [Virtuoso Interactive and Assisted Routing](#)

- [Virtuoso Layout Suite EXL](#)
- [Virtuoso Layout Suite MXL](#)
- [Virtuoso Layout Suite XL](#)
- [Virtuoso Module Generator](#)
- [Virtuoso MultiTech Framework](#)
- [Virtuoso Multi-Patterning Technology](#)
- [Virtuoso NC-Verilog Environment](#)
- [Virtuoso Photonics Solution](#)
- [Virtuoso RF Solution](#)
- [Virtuoso Schematic Editor](#)
- [Virtuoso Simulation Driven Interactive Routing](#)
- [Virtuoso Studio Design Environment](#)
- [Virtuoso Studio Design Environment SKILL](#)
- [Virtuoso Studio Licensing and Configuration](#)
- [Virtuoso SystemVerilog Netlister](#)
- [Virtuoso Technology Data](#)
- [Virtuoso Unified Custom Constraints](#)
- [Virtuoso Virtuoso Visualization and Analysis XL](#)
- [Voltus-Fi Custom Power Integrity Solution L](#)
- [Voltus-Fi Custom Power Integrity Solution XL](#)
- [Voltus-XFi](#)

What's New in Other Products

If a product is not listed above, it means that there are no significant enhancements to be documented in the IC23.1 release for that product.

Key Features in Virtuoso Studio IC23.1

Virtuoso® Layout Suite MXL

Virtuoso® Layout Suite MXL is the advanced Virtuoso Studio application introduced in IC23.1 that offers a unified interface to support design styles across both mature and advanced process nodes, including GAAFETs. Due to its heterogeneous design capabilities, such as co-design and multi-fabric electromagnetic and thermal analysis, the Layout MXL cockpit is also the default cockpit for integrated circuit designs for large system-level designs.

For more information, see [Virtuoso Layout Suite MXL Features](#).

Products and Features Removed from Virtuoso Studio IC23.1

- The following Virtuoso products and their documentation have been removed:

- Virtuoso Analog Design Environment L
- Virtuoso Analog Design Environment XL
- Virtuoso Analog Design Environment GXL

Use the more efficient and powerful environments, Virtuoso ADE Explorer and Virtuoso ADE Assembler, to run simulations for your analog or mixed-signal designs.

- The following Virtuoso features and their documentation have been removed:
 - Setting Outputs form from Virtuoso ADE Explorer-Virtuoso Schematic Editor workspace
 - EDIF Out form from Virtuoso Schematic Editor and Virtuoso Layout Suite
 - DRD compactor from Virtuoso Design Rule Driven (DRD) editing tool
- The following features and their documentation have been removed from Virtuoso ADE Explorer and Virtuoso ADE Assembler:
 - Points submission to sweep variables and parameters
 - Integrated history management
 - Point troubleshooting
 - Feasibility analysis

Documentation Updates in Virtuoso Studio IC23.1

Restructured Documentation

The environment variables for Virtuoso ADE Explorer and Virtuoso ADE Assembler have been repackaged in the following reference guides:

- [Virtuoso ADE Environment Variable Reference](#)
- [Environment Variables](#) appendix in *Virtuoso ADE Explorer User Guide*

- [ADE Assembler Environment Variables](#) appendix in *Virtuoso ADE Assembler User Guide*

The SKILL functions for Virtuoso ADE Explorer and Virtuoso ADE Assembler have been documented in a single reference guide, [Virtuoso ADE SKILL Reference](#).

Removed Documentation

The documentation for the following products is removed from the IC23.1 release:

- *AMS Design and Model Validation*
- *Virtuoso Analog Design Environment L User Guide*
- *Virtuoso Analog Design Environment XL User Guide*
- *Virtuoso Analog Design Environment GXL User Guide*

Help and Support Enhancements

Doc Assistant — A New Documentation Viewer for Cadence Tools

Starting from IC23.1, Virtuoso ships with a brand-new documentation viewer, Doc Assistant.

Doc Assistant provides the following advantages over the previous documentation viewer, Cadence Help:

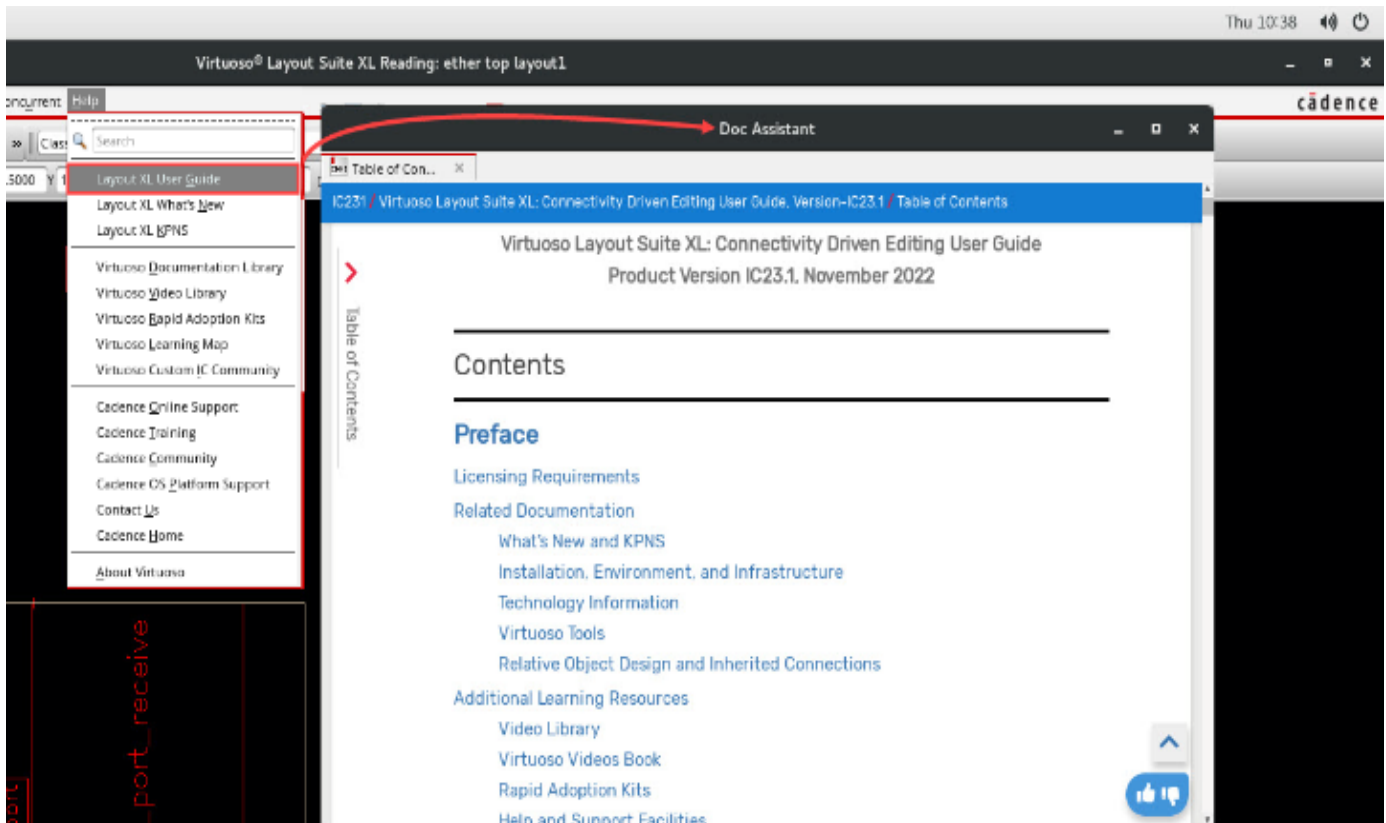
- Presents content in the form of individual, easy-to-read topics.
- Integrates an advanced search engine that offers better search speed and relevance.
- Operates in two modes, Online, in which documents are stored in the cloud, and Offline, in which documents are stored in the installation directory. You can search for content in both Online and Offline modes.
- Provides the facility to store documents in the cloud, with a basic minimum set shipped along with the software. This significantly reduces the documentation footprint in the tool installation.

In IC23.1, Virtuoso is being shipped with the complete documentation set in the installation directory (Offline mode).

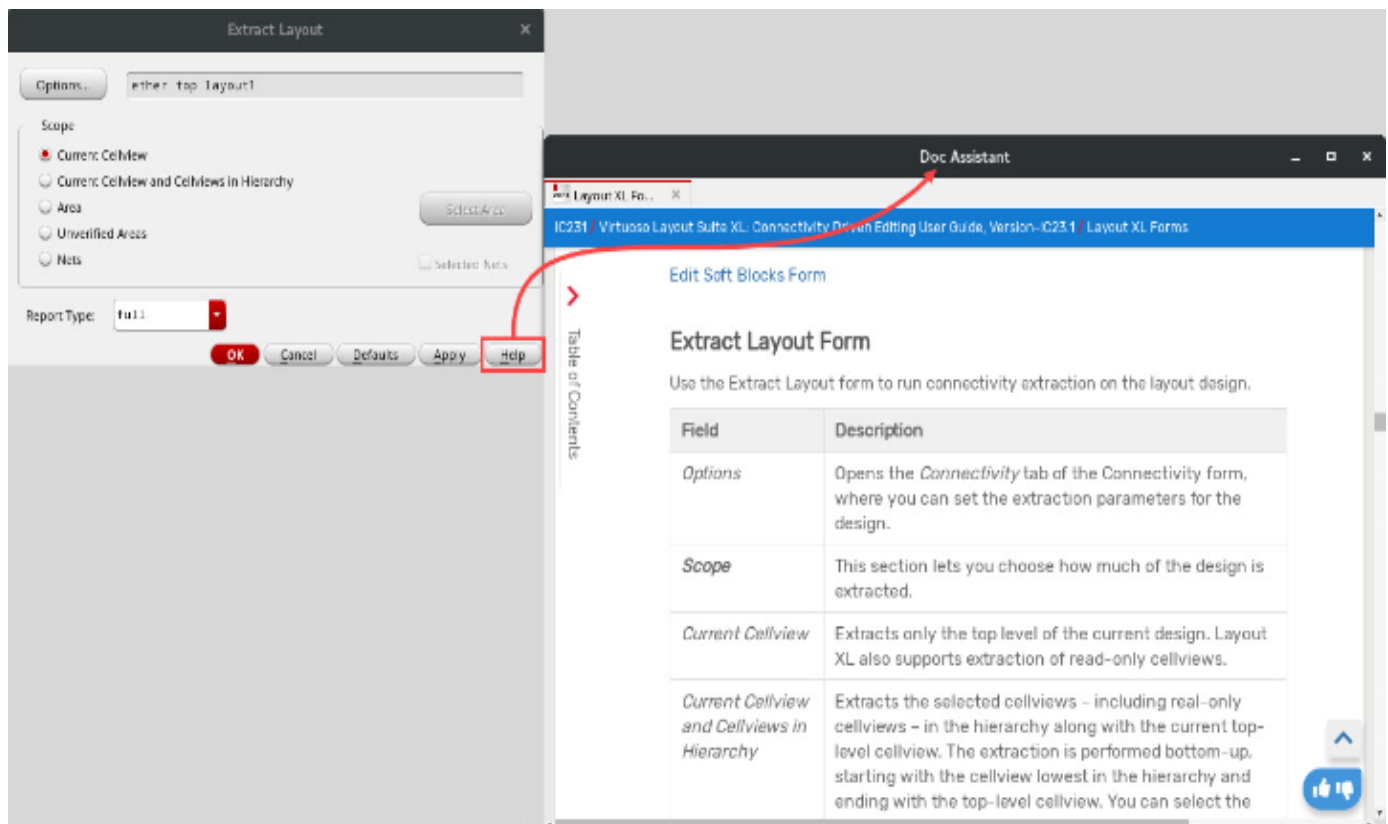
Launching Doc Assistant

You can launch Doc Assistant in one of the following ways:

- Windows systems: Choose *Doc Assistant* from the *Start* menu.
- Linux systems: Navigate to the `<cadence_doc_assistant_install_directory>/bin` directory and run the `cda` command.
- Product *Help* menu: As shown in the following example:






- Tool *Help* button: As shown in the following example:



We Value Your Feedback!

Did the documentation resolve your query? Were you looking for additional information? Share your feedback. We'll address it.

We strongly believe in continuous improvement by gathering user feedback. The Doc Assistant provides multiple options to share your feedback on the content, product, and the documentation viewer.

- Want to share your feedback on the **topic content**? Use the  and  buttons in the topic viewer.
- Want to share your feedback about the **product**? Click the *How was your product experience?* link at the bottom of the TOC.
- Want to share your feedback about **Doc Assistant**? Select *Application Feedback* from the hamburger  menu.

Related Topics

[Introduction to Doc Assistant](#)

Doc Assistant — Known Problems and Solutions

Training Courses

Cadence provides a number of training courses developed to teach a specific tool or design discipline. These courses are delivered through instructor-led live, virtual classes, or online training. You can view the [Cadence Learning Maps](#) to understand the recommended course flows according to tool knowledge levels.

To explore the full range of training courses provided by Cadence in your region, visit [Cadence Training](#) or contact training locations near you.

Videos

Cadence provides a comprehensive set of videos that help you intuitively learn about a tool feature or a flow. This set includes the following:

- Videos included in the Virtuoso installation
- Videos in conjunction with the relevant topic accessible from the Doc Assistant
- The [Video Library](#) on the Cadence Online Support website, which includes recommended videos.

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*.

Blogs

Cadence publishes a number of blogs under different series to highlight the most exciting new features of Virtuoso. These blogs are published on [cadence.community.com](https://www.cadence.com/community.com). We recommend that you take a look at these blogs and, if you like what you see, subscribe to the Custom IC Design blog category to have them delivered directly to your mailbox.

We are now also translating some of our blog content to Chinese and publishing them in the [Custom IC Design \(Chinese\)](#) blog category. To get these blogs delivered directly to your mailbox, subscribe to the blog category.

Virtuoso Release Announcement Blog Series

The Release Announcement blogs are published immediately after a new Virtuoso release is available for [download](#). These blogs not only announce new release arrivals but also point you to key features to look out for in the release, related documentation, and other release collateral.

- [Virtuoso Studio IC23.1 ISR2 Now Available](#) (New)
- [Virtuoso Studio IC23.1 ISR1 Now Available](#)
- [Virtuoso Studio IC23.1 Now Available](#)

Virtuoso Studio Blog Series

The Virtuoso Studio blogs highlight the key features to look out for in Virtuoso Studio IC23.1 release. Keep watching this space for links to new blogs.

- [iPegasus for Signoff DRC and Fill](#) (New)
- [In-Design Verification with iPegasus](#) (New)
- [Design, Planning, and Analysis - 3 Sides of a Coin, Episode 2](#) (New)
- [Organize Your Designs Better with Navigator Queries](#) (New)
- [Design, Planning, and Analysis - The 3 Sides of a Coin, Episode 1](#)

Virtuosity Blog Series

The Virtuosity blogs give you an overview of important and useful Virtuoso features, both old and new, along with supporting links to a range of associated resources including RAKs, videos, application notes, and product manuals.

The latest blogs in this series are listed below. Blogs about features that are available only in the IC23.1 release will be added in due course.

- [Debugging like a Pro Using Voltus-XFi Result Browser: Strategies to Boost Efficiency](#) (New)
- [Accelerate Your EM-IR Closure with Voltus-XFi Custom Power Integrity Solution](#)
- [Annotating Scalar Outputs for Single-Point Simulation in Virtuoso Visualization and Analysis XL](#)
- [Custom IC Design Flow/Methodology - Post-Layout Circuit Simulation and GDSII Generation](#)
- [Driving Super-efficient Chip Design with Voltus-XFi Custom Power Integrity Solution](#)

Virtuoso Video Diary Blog Series

The Virtuoso Video Diary blogs cover new and important Virtuoso features and provide links to related video content and other resources.

The latest blogs in this series are as follows:

- [Knowledge Booster Training Bytes - Part 13: Analog Fault Simulation \(New\)](#)
- [Do More With eyeHeightAtXY and eyeWidthAtXY Calculator Functions in Virtuoso Visualization and Analysis XL](#)
- [Knowledge Booster Training Bytes - Part 12 - Using Quview and Connectivity Checker in Quantus](#)
- [Usability of the Graph Summary Label in Virtuoso Visualization and Analysis XL Levels Up](#)
- [Knowledge Booster Training Bytes - Part 11 - Navigating Through the Training Byte Channels](#)

Knowledge Booster Training Bytes Blog Series

The Knowledge Booster Training Bytes blogs give you an insight into some useful Virtuoso features, both old and new, along with supporting links to a range of associated resources including RAKs, videos, application notes, and product manuals.

The latest blogs in this series are as follows:

- [Switched Capacitor and C2V Converter-Based Circuits \(New\)](#)
- [Virtuoso CLE Webinar Recording Available](#)
- [Routing Techniques for Custom IC Layout Design in Virtuoso Layout Suite](#)
- [The Spectre FX Circuit Simulator](#)
- [The Spectre X Simulator](#)

Virtuoso Meets Maxwell Blog Series

The Virtuoso Meets Maxwell blogs explore the capabilities and potential of Virtuoso RF and Virtuoso Multi-Technology Solution.

The latest blogs in this series are listed below. Blogs about features that are available only in the IC23.1 release will be added in due course.

- [Virtuoso and Allegro SKILL for Efficient Co-Design \(New\)](#)

- [Viewing Your Mesh in EMX Planar 3D Solver \(New\)](#)
- [Custom Passive Device Authoring - Part 2 \(LVS\)](#)
- [Virtuoso Electromagnetic Solver Assistant -Support for Iterated Instances](#)
- [Custom Passive Device Authoring - Part 1 \(Automatic Marker Shape Generation\)](#)

Spectre Tech Tips Blog Series

Spectre Tech Tips is our blog series that focuses on the capabilities of Spectre® and its integration and interactions with Virtuoso® Analog Design Environment (ADE).

The latest blogs in this series are as follows:

- [Selecting Limits for Parameter Value Warnings \(New\)](#)
- [Introducing Spectre FMC Analysis](#)
- [GPU Integration with Analog Circuit Simulation](#)
- [Spectre X High-Capacity Circuit Simulation](#)
- [Addressing Common Spectre EMIR Problems](#)

What's New in Analog Library



This topic provides a high-level overview of the new features in the IC23.1 release.

IC23.1 ISR1

Support of PAM3 Modulation for `isource`, `vsource`, and `port` Symbols

In addition to PAM4 modulation, `isource`, `vsource`, and `port` symbols now support PAM3 modulation when the *Source type* is set to `bit` or `prbs`. Therefore, the *PAM4 modulation* field in the Edit Object Properties form for these three symbols has been renamed to *PAM modulation*. The possible values of this field are *none*, *pam3*, and *pam4*.

IC23.1 Base

Support for S-Parameters with the `ind` and `indq` Symbols

A new checkbox, *Use S-parameters*, is now available in the Edit Object Properties form for the `ind` and `indq` symbols. When you select this checkbox, the *Browse s-parameter file* button and the *S-parameters data file* field is displayed on the form. Either select the *Browse s-parameter file* button to browse and specify the S-Parameter file, or specify the name of the required data file in the *S-parameter data file* field.

What's New in Cadence Application Infrastructure



This topic provides a high-level overview of the new features in the IC23.1 release.

IC23.1 Base

Examining Design Management Related Registries

Use the `gdmregprint` shell command to examine the registry entries that are design management related and visible to the Cadence tool.

What's New in Cadence Library Manager



This topic provides a high-level overview of the new features in the IC23.1 release.

IC23.1 ISR3

Setting Batch Mode in Library Manager Using SKILL Functions

Use the following SKILL functions to control the behavior of batch mode:

- **ImgrBeginBatchChange**: Activates batch change mode to improve the performance of Library Manager for handling `lmgrSetLibDisplayOverride` calls.
- **ImgrEndBatchChange**: Deactivates batch change mode of Library Manager for the `lmgrSetLibDisplayOverride` calls.
- **ImgrInBatchChange**: Checks the cumulative state of batch change mode including that of the active nested scopes.

Customizing Color and Icon in Library Manager Using SKILL Functions

You can now use the following SKILL functions to modify the Library Manager display:

- **ImgrSetLibDisplayOverride**: Sets color, icon, or both for a library or a group of libraries to display in the Library Manager.
- **ImgrGetLibDisplayOverride**: Returns the color and icon display override settings specified by `lmgrSetLibDisplayOverride`.

- [ImgrClearLibDisplayOverride](#): Removes the display override for a library including the color and icon values specified by `lmgrSetLibDisplayOverride`.
- [ImgrClearLibDisplayIconOverride](#): Removes the icon display override setting specified by `lmgrSetLibDisplayOverride`.
- [ImgrClearLibDisplayColorOverride](#): Removes the color display override setting specified by `lmgrSetLibDisplayOverride`.

IC23.1 ISR2

Using SKILL to Filter Cellviews in the Library Manager

You can now use the [ImgrSetLCVFilter](#) SKILL function to filter libraries, cells, and views in the Library Manager.

Displaying Non-Virtuoso View Types in Library Browser Form

You can select the *Show Non-Virtuoso View Types* option in the Library Browser form to view the non-Virtuoso view types for the selected cell.

Related environment variable: [showNonVirtuosoViewtypes](#)

What's New in Cadence SKILL IDE



This topic provides a high-level overview of the new features in the IC23.1 release.

IC23.1 ISR1

Supporting tCov reports for Same File Name

In order to support generating tCov reports for same file name with different extension SKILL files, a new command line argument *-iITCovUseFullFileName* has been added.

What's New in Component Description Format



This topic provides a high-level overview of the new features in the IC23.1 release.

IC23.1 Base

UI Improvements in the Edit CDF Form

Earlier, the *Component Parameter* tab on the Edit CDF form supported the `yes`, `no`, and `don't use` values for the *Store Default* field. The tab also had the *Parse as CEL* and *Parse as Number* fields that allowed using different combinations for evaluation of parameter values.

Now, the *Component Parameter* tab has been enhanced with the following improvements:

- The *Store Default* field now supports only the `yes` and `no` values. The `don't use` value has been removed.
- The *Parse As CEL* and *Parse As Number* fields have been consolidated into a single list, *Evaluation Mode*, which lets you select pre-defined combinations for evaluation of parameter values.
- The fields in the *Component Parameter* tab now have tooltips that simplify specifying parameter values.
- The parsing of double quotes (") in choices for radio and cyclic type of parameters to CSV format has been enhanced.

What's New in Design Data Translators



This topic provides a high-level overview of the new features in the IC23.1 release.

IC23.1 ISR3

Translating LEF58_ANTENNADIFFAREA as Width and Ratio in LEF In and LEF Out Translators

You can now specify whether the values specified for the LEF MACRO pin property LEF58_ANTENNADIFFAREA can be translated as width and ratio during LEF In and LEF Out translation.

The following options have been added to the LEF In translator:

- New command-line option `-antennaDiffAreaPropValueIsWidthAndRatio` added to the [lefin](#) command.
- New GUI option *Extract LEF58_ANTENNADIFFAREA* added to the [LEF In](#) form.
- New argument `g_widthAndRatio` added to the [ldtrLefReadOA](#) SKILL function.

The following options have been added to the LEF Out translator:

- New command-line option `-antennaDiffAreaPropValueIsWidthAndRatio` added to the [lefout](#) command.
- New GUI option *Create LEF58_ANTENNADIFFAREA from Constraint* added to the [LEF Out](#) form.
- New argument `g_widthAndRatio` added to the [ldtrLefWriteOA](#) SKILL function.

IC23.1 ISR2

Using Standard Via Definitions for Via Detection during XStream In Translation

During XStream In translation, you can enable the *Use stdViaDefs from Technology Library* option on the *Others* tab of the XStream In More Option form to use the standard via definitions from the technology database for via detection during translation.

Using Standard Via Definitions for Via Detection during XOasis In Translation

During XOasis In translation, you can enable the *Use stdViaDefs from Technology Library* option on the *Others* tab of the XOasis In More Option form to use the standard via definitions from the technology database for via detection during translation.

Increased the Number Vertices Supported During XStream Out Translation

You can now specify up to 8191 vertices using the *Max Vertices* option on the *Geometry: Limit* tab of the XStream Out More options form.

Environment variable: [maxVertices](#)

IC23.1 Base

Support for Multithreading Added to XStream Out

The XStream Out translator now supports multithreading that helps in improving performance during translation.

What's New in Open Simulation System



This topic provides a high-level overview of the new features in the IC23.1 release.

IC23.1 Base

Improved Error Detection for SKILL Flags

The following SKILL flags have been enhanced with improved error detection and reporting:

- [hnlCheckInstParamDparMismatch](#)
- [hnlConfigMissingViewAction](#)
- [hnlEmptySwitchMasterAction](#)
- [hnlHandleMultiplePlaceMaster](#)
- [hnlInvalidBindingAction](#)
- [hnlPcdbErrorCheck](#)
- [simCheckNetCollisionAction](#)
- [simCheckShortCVMismatchAction](#)
- [simCheckTermDirectionMismatch](#)
- [simCheckTermMismatchAction](#)
- [simSymbolModifiedAction](#)

Change in Default Value of SKILL Flags

The following SKILL flags have been enhanced:

- **hnlConfigMissingViewAction**: The default value has been changed from `warning` to `error`.
- **hnlCheckInstParamDparMismatch**: The default value has been changed from `ignore` to `warning`.
- **hnlInvalidBindingAction**: The default value has been changed from `warning` to `error`.
- **hnlMaxNameLength**: The default value has been changed for various netlisters.
- **simStopNetlistOnPcellFailure**: The default value has been changed from `CheckNetlistOnly` to `CheckNetlistAndSchLabels`.

What's New in Virtuoso Abstract Generator



This topic provides a high-level overview of the new features in the IC23.1 release.

IC23.1 ISR2

Antenna Generation for MUSTCONNECTALLPORTS Terminals

Abstract Generator now supports antenna generation for all the `MUSTCONNECTALLPORTS` terminals and lets you calculate antenna area by combining all the disjoint islands into a single island.

IC23.1 Base

Customizable Exit Hook for the Abstract Generator User Interface

Abstract Generator supports customizable SKILL hooks that extend the functionality of the existing Abstract Generator flow steps. The tool now supports Exit Hook, a new SKILL hook that runs when you close the Abstract Generator user interface.

Character Limit Specification for the Power Rail Analysis Report

You can now enter an inherent limit of characters defined by SKILL in the Power Rail Analysis report. The report is divided such that each block of message can contain only a limited number of characters in a line.

What's New in Virtuoso ADE Explorer and Virtuoso ADE Assembler



This topic provides a high-level overview of the new features in the IC23.1 release.

IC23.1 ISR3

Enhanced Simulation Manager Architecture

The following enhancements have been done in the architecture used by Simulation Manager:

- The ADE broker, Simulation Manager, and the Virtuoso Studio instance running ADE Assembler can now run on separate resources supported by a completely distributed architecture. The ADE broker and Simulation Manager processes can run on any remote resource and their lifetime does not depend on the Virtuoso Studio instance.
- You can specify the resource specifications for Simulation Manager by using a separate job policy specified through the Job Policy Setup form.
- You can use the new `maeBroker` command to run ADE broker as a standalone service. Virtuoso Studio instances can find and connect to the ADE broker by using the `CDS_MAE_BROKER_ADDRESS` shell environment variable that specifies the address and port number for the ADE broker.
- The centralized ADE broker service now supports a multi-user environment.

Running Advanced Optimization Method

Advanced Optimization (AOP) is a new framework that enables you to integrate your custom algorithms into Virtuoso ADE Assembler using Python and C++ APIs. A new form, *AOP Options*, is added to let you specify the advanced optimization algorithms and hyperparameters. AOP integrates a new visualization framework to plot and analyze optimization data, such as real-time progress, outputs versus parameters, sampling distribution, and so on. AOP also includes several new optimization algorithms, some of which include machine-learning surrogate models that can improve sample efficiency and reduce optimization time.

To use this feature, you must have a Virtuoso Variation Option (VVO) license.

Running Fast Monte Carlo (FMC) Method

The following enhancements have been made to the Monte Carlo form for the Fast Monte Carlo (FMC) method:

- The *Max Points* field is renamed as *Total Samples*.
- Two new optional fields, *Budget* and *Initial Points*, have been added. Using the *Budget* field, you can limit the number of simulations to be run for the FMC method. In the *Initial Points* field, you can specify the number of initial points based on which metrics, such as mean and standard deviations, are calculated and annotated in histograms. Specifying a higher number of initial points improves the accuracy of mean and standard deviation measurements.

Switching Between Reliability Modes

There are two reliability modes available in ADE Assembler: reliability analysis and reliability scenario. You can now use the context-menu command *Switch Reliability Mode* to switch between these reliability modes.

Viewing Tabular Report for Operating Point Parameters

You can now view the results for operating point parameters in tabular format.

Support for Process-Based Save and Restart (PBSR) Flow in Virtuoso ADE Assembler

Earlier, you could use the Process-Based Save and Restart (PBSR) flow for Virtuoso ADE Explorer only, but now Virtuoso ADE Assembler has been enhanced to support the PBSR flow. To enable this flow, select *Enable process based save and restart* in the *Assembler Save and Restart Options* group box on the Netlist and Run Options form.

Job Summary in the Run Summary Assistant

Earlier, the Run Summary assistant displayed the job status of simulation jobs in a list view. You can now view the job status in a graphical view.

Checking Network Health Before Running a Simulation

Check the network health before running a simulation by submitting the number of jobs specified in the *Max. Jobs* field of the Job Policy Setup form without actually running the processes. To do this, select *Enable Dry Run for Command Distribution Method* on the Debug Utility form. The maximum limit for the number of jobs that can be submitted is 100.

Viewing the Noise Summary Report

View the noise summary report in tabular format through the Noise Summary window. This window has a separate tab for each corner. To view the Noise Summary window, set the environment variable `maestro.gui.showNoiseSummaryinTableWidget` to `t`. You can then specify the settings on the Noise Summary form and click *Apply* to generate the report.

Adding Variable Tags in Virtuoso ADE Assembler

In the earlier releases, it was possible to add separate tags to the global and design variables with the same name. From this release onward, a common tag is used for the global and design variables with the same name. When you open an existing maestro cellview that earlier used different tags for the design and global variables with the same name, the tool merges the tags in the active setup and displays an information message about the same in the CIW.

Assigning Tags to Corners

You can now use special characters in the tags for corners in the Corners Setup form. The supported special characters are the same as those supported for history names.

Using Config Sweep Variables in Corners

Use the new Add/Edit Config Sweep form to specify config sweeps in the corners setup.

SKILL Functions to Export or Import Design Variables

Use the SKILL functions `maeExportDesignVariables` and `maeImportDesignVariables` to export and import design variables, respectively.

New Argument for Run Plan Support in `maeLoadSetupState`

A new argument, `?run`, has been added to the `maeLoadSetupState` SKILL function. This argument lets you specify the run in a Run Plan to which a saved setup state must be loaded.

IC23.1 ISR2

Exporting and Importing a Reliability Scenario Setup

You can export a reliability scenario setup to a CSV file. You can later import this CSV file to copy the setup to the Reliability Scenarios Setup form.

SKILL functions: `maeExportScenarioSetup`, `maeImportScenarioSetup`

Plotting the Results of Sensitivity Analysis in Separate Subwindows

You can use the environment variable `maestro.plotting.plotSensitivityDataInSeparateSubwindows` to plot the results of sensitivity analysis in separate subwindows.

Rerunning Simulations for Specific Unfinished or Erroneous Simulation Points

You can now rerun simulations for specific unfinished or erroneous simulation points. To do this, in the *Results* tab, select the points to be simulated again, right-click any one of those points and choose *Manage Point Results - Rerun Point*.

Using GPU with Spectre X

You can now use GPU mode of Spectre X to speed up simulations if there are GPUs available in your setup and you have large designs that have long-running transient analyses, post-layout, or advanced node simulations. To enable GPU mode, select the *GPU* check box on the High-Performance Simulation Options form.

Using Matched Parameters from Design

Earlier, the matched parameters were read by traversing the design while opening the maestro cellview because the `adexl.gui disableConstraintsRead` variable was set to `nil`. Starting from this release, the default value of this variable has been set `t` to improve the performance. You can use the new *Import Constraints* command from context menu of the *Parameters* tree in Data View assistant to import matched parameters after loading the cell view.

Viewing DSPF Substitution for subckt Blocks in Run Statistics

The outputs on the *Results* tab now include the subckt information related to DSPF substitution. This information is extracted from the Spectre output file and displayed as `sim_DSPF_Substitution` in the outputs table when you run the *Run Statistics* command.

Environment variable: `defaultRunStatisticItems`

Setting the Design for Tests

Use the `maeSetDesignForTest` SKILL function to set the design for all or the specified tests.

Importing Outputs from CSV Files

Use the `maeImportOutputsFromFileInDocuments` SKILL function to import outputs from the specified CSV file in the `documents` directory to one or all tests in an already opened maestro cellview.

Default Window Size of ADE Assembler and ADE Explorer

The default size of the ADE Explorer and ADE Assembler windows has been changed. The default width is 1400 pixels and the height is 800 pixels.

Removed Features

The following features have been removed in this release:

- **Run command from the Simulation menu:** The *Run* command has been removed from the *Simulation* menu in ADE Explorer. To run a simulation, use the *Netlist and Run* command.
- **Diagnostics tab:** The *Diagnostics* tab has been removed from ADE Explorer and ADE Assembler. To troubleshoot simulation errors in ADE Assembler, you can generate the ADE

diagnostics report by right-clicking a history in the Data View Assistant and choosing *Diagnostics - Analyze Logs*.

For more information, see [Debugging with ADE Diagnostics](#).

IC23.1 ISR1

Using calcVal Wizard

Use the new calcVal Wizard to edit and debug calcVal expressions in ADE Explorer or ADE Assembler.

Running Fast Monte Carlo (FMC) Method

The *Fast Monte Carlo* (FMC) method lets you extract useful statistical information without running the complete set of Monte Carlo samples, especially for high-sigma analysis. This significantly reduces the simulation time.

Using Simulation Manager

Enable the *Use Simulation Manager* check box in the Job Policy Setup form for ADE Assembler to use the new decentralized and distributed architecture for the management of simulation jobs. This new architecture offloads the computation-intensive tasks from the Virtuoso session and makes the session more responsive to the user. Even if you close the host Virtuoso session that started the simulation, the Simulation Manager manages the progress of simulation jobs. When you reopen the cellview, it can reconnect the cellview to the simulation process if the simulation is still in progress. You can then control the simulations in the same way as you could before exiting Virtuoso.

Using the simulation manager can be beneficial for complex designs and large setups that have a high simulation run time because it allows users to exit Virtuoso to release the memory reserved for other operations that could affect the simulation progress. In addition, this architecture helps in scenarios when Virtuoso stops abruptly during a simulation because you can reconnect with the running jobs and control them as before. You would require a special license to use this feature.

Using LSCS for third-party simulator

Set the `maestro.lscs useLSCSFor3rdPartySimulators` environment variable to `t` to use LSCS for a third-party simulator.

Displaying Error Messages for Undetected Job Policies

The `maestro.distribute showErrorForNonExistingJobPolicy` environment variable lets you control the display of an error message when a user-defined job policy is not detected by the tool. The default is `t`, which means that the error message is displayed.

Displaying Error Messages for Malformed Lock State Files

The `maestro.setupdb autoDeleteMalformedLockedSDB` environment variable lets you specify whether to display a warning message when a malformed lock-state file is found. The default is `t`, which means that the malformed lock files are deleted automatically without displaying a pop-up message.

Enabling or Disabling a Reliability Scenario

You can now enable or disable reliability scenarios by selecting or deselecting the check boxes next to them in the Reliability Scenario Setup form. This allows you to run a combination of specific scenario rows in the reliability scenario setup, without having to delete the reliability scenario rows that you do not want to simulate.

IC23.1 Base

Using the ADE Diagnostics Report for Troubleshooting

Right-click a history in the *History* tab of the Data View assistant and choose *Diagnostics - Analyze Logs* to generate the ADE diagnostics report. This HTML report provides the information to help you troubleshoot the issues found during a simulation.

Related environment variables:

[maestro.debug netlisting](#)

[maestro.debug simAndEval](#)

[maestro.debug beanstalk](#)

[maestro.debug mainVirtuoso](#)

Related SKILL functions:

[maeRunLogDiagnostics](#)

[maeDisplayLogDiagnostics](#)

Re-evaluating Results Using Distributed Re-evaluation Mode

Use Distributed re-evaluation (DRE) mode to re-evaluate results in a background process without freezing the maestro session.

Enabling Distributed Plot Service

Use the option *Enable Distributed Plot* from the newly added section *Distributed Plot* in the ADE Assembler Plotting/Printing Options and ADE Explorer Plotting/Printing Options forms to enable the distributed plot service. To specify the linger time for a distributed plot process, use the *Linger Time* field. Linger time is the period in seconds for which the distributed plot process waits before exiting when there is no active plotting window is visible.

Related environment variable:

[maestro.plotting distributedPlotLingerTime](#)

Issuing Warning Messages for Waveforms not Supported by Distributed Plot Service

Use the environment variable `maestro.plotting distributedPlotEnableNonSupportedPlotWarning` to issue warning messages for waveforms that cannot be plotted by Distributed Plot service.

Default value: `t`

Filtering Job Policies in the Job Policy Selection for Distributed Plot form

Use the environment variable `maestro.plotting distributedPlotPolicyNameFilter` to specify the string by which you want to filter the job policies displayed in the *Job Policy Name* drop-down list in the *Job Policy Selection for Distributed Plot* form.

Default value: `" "`

Plotting Waveforms for Device Checks

Use the hyperlinks in the *Start Time(s)* column of the ADE Assembler results view to plot waveforms for the values given in the *Message* column for device checks.

Specifying a Name for Simulation History

Use the new *History Name* toolbar to choose between the default history name or a custom history name for the current run mode. You can also specify a custom history name on this toolbar.

Checking User Quota in Disk Space for Resource Estimation

Use the following new fields on the *Resources* the Job Policy Setup form to check user quota in the available disk space if the *Show warning for low disk* check box is enabled:

- *Check disk space using the command to get user quota* to enable checking of the disk space available in user quota
- *Command to get user quota* to specify the command to be run to check the user quota. The command must return an output in the format `'available: space'`.

Forcefully Setting Results Status to Enable Re-evaluation

If a simulation is not completed successfully due to a severe problem, the status of all or a few results remains stuck at a value that prevents re-evaluation. However, you might suspect that good simulation results are available in the database, but they were not used for expression evaluation.

If the simulations can be rerun quickly, it is recommended to run simulation and create a new history with good results. However, if you have large designs and a new simulation takes a significant time to complete, use the new *Changed Results Status to Done* command from the shortcut menu of the *Detail* or *Detail-Transpose* results view or a history to forcefully change the status of results thereby enabling re-evaluation of results.

Related environment variables:

[maestro.results forceRdbStatusEnable](#)

[maestro.results forceRdbStatusIgnoreList](#)

[maestro.results forceRdbStatusTo](#)

Related SKILL function:

`maeForceRdbStatus`

Copying Tests from Active Setup to Run Plan

Use the SKILL function `maeCopyTestToRun` to copy one or more tests from the active setup to the run plan.

Removing the `.tmpADEDDir` directory

Click *Remove .tmpADEDDir* on the Output Setup toolbar in ADE Explorer and Tools toolbar in ADE Assembler to remove the `.tmpADEDDir` directory.

Related environment variable:

`maestro.gui showRemoveTmpADEDDirButton`

Disabling the Generation of Quick Plot Data

Use the environment variable `maestro.plotting quickPlotGlobalWriteDisable` to specify whether to turn off the generation of Quick Plot data.

Default value: `nil`

New License for Distributing Simulation Jobs

A new license, Virtuoso Simulation Expansion Option (VSEO), is now available for enabling distribution of job resources. A separate instance of this license is checked out for every increment of 400 jobs.

Exporting Monte Carlo Results to a CSV File

A new option, *Export Data to CSV* has been added to the drop-down list for Monte Carlo post-processing options in the Results tab. This option lets you export Monte Carlo results to a CSV file.

Elaboration of disabling constraint manager hierarchy

The default value of the `adexl.gui disableConstraintsRead` environment variable that enables or disables the elaboration of the Constraint Manager hierarchy to find matched parameter constraints has been changed to `nil`.

Removed Products

The following Virtuoso products and the documentation related to these have been removed:

- Virtuoso Analog Design Environment L
- Virtuoso Analog Design Environment XL
- Virtuoso Analog Design Environment GXL

Use the more efficient and powerful environments, Virtuoso ADE Explorer and Virtuoso ADE Assembler, to run simulations for your analog or mixed-signal designs.

Removed Features

The following features have been removed from Virtuoso ADE Explorer and Virtuoso ADE Assembler:

- **Integrated History Management:** The separate history management feature is now enabled for all maestro cellviews, which implies that ADE Assembler always saves the history setup and maestro cellview setup separately. If your cellviews have integrated history management, open them in edit mode to automatically convert them to use separate history management. In addition, the following have also been removed:
 - The `useSeparateHistoryFileManagement` environment variable
 - The `maeConvertViewForIntegratedHistoryManagement` SKILL function
- **Feasibility Analysis:** This run mode is not available in the Run Mode drop-down list.
- **Point Troubleshooting:** The *Troubleshoot Point* command has been removed from the context menu of data points in the *Results* tab.
- **Point submission to sweep variable and parameters:** The following forms and commands that you could earlier use to sweep variables and parameters or to submit the resulting points for evaluation have been removed:
 - Single Run, Sweeps and Corners form
 - Edit Submit Point form
 - *Submit Point* command in the context menu of a design point in the Results tab
 - Submit Point form
 - Edit Submit Point form
- **Setting Outputs form:** The Setting Outputs form has been removed from ADE Explorer. To edit the simulation outputs from an ADE Explorer schematic, you can now

choose *Setup — Outputs* from the *ADE Explorer* menu and use the Outputs assistant.

What's New in Virtuoso ADE Verifier



This topic provides a high-level overview of the new features in the IC23.1 release.

IC23.1 ISR3

Support for Reliability in Setup Library Cellviews

The Setup Library assistant has been enhanced with support for reliability. Similar to sweep setups, corner setups, and simulation setups, you can add reliability setups and options to your setup library cellview to optimize your verification configuration for simulations. To support this, a new group, *Reliability Setups*, is available in the setup library cellview hierarchy.

Use the following functions to manage various tasks associated with reliability setups in a setup library cellview:

- [`slaAddReliabilityOptions`](#): Lets you add new reliability options in the setup library view with default values.
- [`slaCreateReliabilitySetup`](#): Lets you create a new reliability setup in the setup library view with the specified name.
- [`slaGetReliabilityOptions`](#): Retrieves the list of reliability options added to the specified reliability setup.
- [`slaGetReliabilitySetups`](#): Retrieves a list of all reliability setups that are included in the setup library cellview.
- [`slalsReliabilityOptionsEnabled`](#): Retrieves the enabled or disabled status of the reliability

options added to a reliability setup in the setup library view.

- [slaRemoveReliabilityOptions](#): Removes the specified reliability options from a given reliability setup.
- [slaRemoveReliabilitySetup](#): Removes the specified reliability setup from a setup library cellview.
- [slaSetReliabilityOptionsEnabled](#): Enables or disables the specified reliability options in a reliability setup.

Timeout for Stopping Runs

Use the `stopTimeout` environment variable to specify the timeout limit for stopping runs. If the simulation continues to be in the `Stopping` state after this duration, the run is considered to have failed.

Running Custom Simulations with ADE Assembler Cellviews

Earlier, simulation runs in ADE Verifier would use the specifications defined in the implementations that were based on ADE Assembler cellviews. Starting this release, you can set the [loadToActiveWhileViewAndReuseSession](#) environment variable to `t` to enable customized simulations. For this, you open an ADE Assembler cellview using the *View Implementation Results* command in ADE Verifier and make temporary modifications in the ADE Assembler cellview. Without closing this cellview, when you click the *Run* button in ADE Verifier, the simulation is run with the modified specifications from ADE Assembler cellviews. This functionality allows you to run custom simulations without saving the modifications to the actual ADE Assembler cellview.

IC23.1 ISR1

SKILL Functions for Creating and Deleting Run Summary Data

Use the following SKILL functions to manage tasks related to run summary data:

- [verifCreateRunSummaryData](#) lets you create the run summary data for the specified implementations.

- [verifDeleteRunSummaryData](#) lets you delete the run summary data for the specified implementations.

SKILL Functions for Setup Library Cellviews

Use the following functions to manage various tasks related to a setup library cellview:

- [slalsViewOpened](#) lets you check and return the opened or closed state of a setup library cellview.
- [slalsCornerEnabled](#) lets you retrieve the enabled or disabled status of the specified corner within a corner setup in a setup library view.
- [slaSetCornerEnabled](#) lets you set the enabled or disabled status of the specified corner within a corner setup in a setup library view.

Enhanced Support for Requirement Data Types in Export and Import Flows

The export and import flows support mapping of source columns to four additional data types available in requirements, *Scope*, *Domain*, *Deadline*, and *CellviewHolder*. Additionally, checks have been introduced to ensure that only valid values are specified for *Domain*, *Deadline*, and *CellviewHolder*.

IC23.1 Base

Composite Requirements

Virtuoso ADE Verifier supports the composite requirement, which has a parent-child hierarchical structure. This requirement type is mainly suitable for complex verification setups when you want to verify a single measurement under different conditions. Additionally, you can run the implementation with multiple verification spaces and slice the simulation results according to the assigned verification spaces for both nominal and statistical outputs.

Related environment variables: [saveCompositeSetups](#), [updateInheritedMappings](#)

Enhancements in the Setup Library Cellview

The following enhancements have been made in the Setup Library assistant:

- The *Netlist Options* variable has been added in the Setup Library [simulation setup](#) to let you specify the netlisting options for a Monte Carlo simulation.
- A new SKILL function, [maeGetSetupLibrary](#), has been added to let you retrieve the name of a setup library cellview from an active or specified session.

Stopping Child Batch Runs

A new environment variable, `stopBatchRunOnServerClose`, is available to let you stop the child batch runs for which the Virtuoso session that launched the session has closed.

Removed Features

Removal of the `disableReadOnly` Environment Variable

Starting IC23.1 release, the `verifier.run disableReadOnly` environment variable is no longer available in Virtuoso ADE Verifier.

What's New in Virtuoso Automated Placement and Routing Flow



This topic provides a high-level overview of the new features in the IC23.1 base release and subsequent ISR releases.

IC23.1 ISR3

Loading Settings From a File to the Auto P&R Assistant

You can now load settings from an options file into the Auto P&R assistant. You use the [apLoadOptions](#) SKILL API to define the cellview for which the settings are to be loaded, the file that contains the settings, and the placement style for which the values are applicable.

Switching Placement Style in SKILL

Instead of using the UI options, you can now use the [apSwitchPlacerType](#) SKILL function to switch between the placement styles.

Applying Transition Spacing While Inserting Fill

In the automated placement and routing flow, transition spacing is specified as part of the placement step. When you move to the next step and insert device fill, you can now use the [fillRegionHonorTransitionSpacing](#) environment variable to specify the direction along which the transition spacing value is to be applied.

Setting the PDK Flow to Primitive or Analog Cell

You can identify a PDK as an analog cell PDK or a primitive PDK. The `isOnlyAnalogCellFlow` environment variable lets you specify the flow that the current PDK supports.

IC23.1 ISR2

Using Preset Files to Reuse Device Settings

You can now use preset files to save your device-level Auto P&R assistant settings. You can then re-apply these settings to another design. To support this feature in the device mode, the Auto P&R assistant toolbar now includes the following buttons:

- **Load Preset Options:** Loads options from an existing preset file.
- **Save Preset Options:** Saves the preset options to file.

Enhanced NetClass, DiffPair, and Symmetry Constraints

A new field *Same Mask* has been added to the `NetClass`, `DiffPair` and `Symmetry` forms. The *Same Mask* field also appears as a column in the table in the *NetClass*, *DiffPair*, and *Symmetry* tabs of Routing Constraint Manager.

Loading a Pullback File

The *Load* option has been introduced in the Pullback and Offset Values form to load the pull back values from a text file.

Creating Pins on Pin Purpose

A new option *Create on pin purpose* has been added to the *Pins* section of the *Supply* tab in the `Routing Assistant`. This option lets you create routing pins on pin purpose instead of drawing

purpose. You can also use the [supply_createPinsOnPinPurpose](#) environment variable to create routing pins on pin purpose instead of drawing purpose. The new option and the environment variable has been added for both device-level and standard cell routing types.

Preroutes Renamed to Manual Routing

The *Preroutes* option in the *Delete* section of the *Route* tab in the Routing Assistant has been renamed to *Manual Routing* for standard cell routing. This change has been made to have the same delete options for standard cell and device-level routing types.

Loading Presets

A new SKILL API [vireLoadPreset](#) has been introduced to load a router preset file.

Selecting Row or WSP Region

A new option *WSP/Row* has been added to the *Scope* section of the *Supply* tab in the [Routing Assistant](#) to let you create supply stripes inside a row region or WSP region. This new option has been added for both device-level and standard cell routing types.

Assigning Wire Type to Nets Enhancements

The use model for assigning wire type to nets has been enhanced to pre-populate with existing power and ground nets. Also, all signal nets are now allowed to be set.

Specifying Routed Cellview

Rather than selecting a cell name to which you want to save the routing results, you can now specify a new cell name to save the routing results. The cell is created as an instance in the design library. You can search for the cell name in the design library with `<cellname>_proute`. By default, the view name is always layout. You can also use the [supply_defaultRoutedCellExpression](#) environment variable.

Automated Placement and Routing Environment Variables

The following new environment variables are now available for the Automated Placement and Routing flow.

- [FB1SpecialDRCRegionLayer](#)

- [supply_defaultWireCGOverride](#)
- [supply_pinLayers](#)

Automated Placement and Routing API-based Flow

As an alternative to using the UI options in the Auto P&R assistant, you can now use the following new SKILL APIs to run various placement-related tasks:

- [apCreateDeviceRowRegion](#): Creates a new row region.
- [apDeleteDeviceRowRegions](#): Deletes all automatically created rows and row regions.
- [apInsertFill](#): Inserts fill in the given layout.
- [apDeleteFill](#): Deletes fill from the given layout.
- [apInsertTrims](#): Inserts trims to fix shorts.
- [apDeleteTrims](#): Deletes trims.
- [apValidateTrims](#): Validates trim insertions.
- [apRunCreateGR](#): Creates guard rings.
- [apRunDeleteGR](#): Deletes guard rings.
- [apAdjustBoundary](#): Resizes the PR boundary to cover the devices that were earlier placed outside the PR boundary.
- [apCapturePlacement](#): Captures placement data from the current (source) layout and stores it in a migration directory.
- [apRunDeviceGeneration](#): Generates the selected object types from the source.
- [apFixColors](#): Swaps colors of flipped rows.
- [apRunBackannotate](#): Runs back annotation on the given design.

Customizing the Interactive Placer Context Menu

You can now use the following SKILL APIs to add or remove menu items from the interactive placer context menu.

- [apIPRegisterCustomMenuItem](#): Registers a custom context menu item in the interactive placer context menu.
- [apIPUnregisterCustomMenuItem](#): Removes the specified custom menu item from the

interactive placer context menu.

- [apIPUnregisterCustomMenuItems](#): Removes all custom menu items from the interactive placer context menu.

IC23.1 ISR1

Enhanced DiffPair and Symmetry Constraints Forms

To include creating *msTolerance*, a new field *Tolerance* has been added to the [DiffPair](#) and [Symmetry](#) Process Rule Override Editor forms.

Enhancements in Routing Constraint Manager

- The Process Rule Overrides and Group Process Overrides column has been added to the *Nets* tab of the Routing Constraint Manager.
- The Tolerance column has been added to the *Symmetry* and *DiffPair* tabs of the Routing Constraint Manager.

Enhancements in Routing Results Browser

New columns for each individual constraint check have been added to the Routing Results Browser to show the count of violations reported from Routing Constraint Manager constraint checks.

Enabling All Constraint Checks

To enable all constraint checks at the same time, a new *Checks* option has been added to the [Constraint Checker](#) form.

Removing Shorts

A new SKILL API [vcrRemoveShorts](#) has been introduced to remove shorts on a net or nets from the layout design.

Grouping the Supply Grid Per Metal Layer

A new environment variable [prouteGroupByLayer](#) is available to create a child figGroup for grouping path and via by layer for the supply routing result.

Remastering Instances



A new environment variable [remasterLayoutLibs](#) is available to specify a library that contains layout views, which is used to remaster instances.

Usability Enhancement in Supply Tab

The *Pins* section has been rearranged and moved out of the *Options* section of the *Supply* tab of the Routing Assistant.

Loading and Saving Constraint Order from a Preset File

The *Constraints* tab of the Auto P&R assistant now includes the following buttons that let you save and reuse the constraint order:

- Load : Loads the constraint order from a preset file.
- Save : Saves the current constraint order to a preset file.

Layout Migration Using the Auto P&R Assistant

The Auto P&R assistant now supports the assisted flow of the Virtuoso® Custom Design Migration solution. In Layout MXL, the following tabs of the Auto P&R assistant include an additional *Migration Options* section:

- *Initialize* tab: Captures design placement data, for example, the PR boundary, pins, instances, and routing shapes, from a source layout and applies the captured data to a target layout.
- *Constraints* tab: Captures groupings from a source layout and applies the captured data to a target layout.

Running the Placer on Mature Nodes Designs

In addition to advanced node designs, you can now run the Virtuoso device-level automatic placer on mature nodes designs. Set the **matureOrAdvancedNode** environment to `mature` to switch the process node. You can then use the options on the *Placer* tab of the Auto P&R assistant to run the placer on mature node designs.

Using Multi-Fingered Analog Stack Gate Cells as Transition Fill

You can now use multi-fingered analog stack gate cells as transition fill in the *Fill* step of the automated device placement and routing flow. The following environment variables are available to support this feature:

- **multiFingerTransFill**: Specifies whether multi-fingered stack gate analog cells are to be used as transition fill.
- **transitionFillFingerCount**: Specifies the number of fingers to be included in the stack gate transition fill cells.

Removed Features

The following device fill-related SKILL functions are no longer supported:

- `lobRegUserProc`
- `lobUnRegisterTransitionFillDefsProc`
- `lobUnRegisterTapFillDefsProc`
- `lobUnRegisterAdjacentFillDefsProc`
- `lobRegisterTransitionFillDefsProc`
- `lobRegisterTapFillDefsProc`
- `lobRegisterAdjacentFillDefsProc`
- `lobGetRegisteredTransitionFillDefsProc`
- `lobGetRegisteredTapFillDefsProc`
- `lobGetRegisteredAdjacentFillDefsProc`
- `lobBaseLayerDummyFillWrapperCB`
- `lobBaseLayerDummyFillCB`

The following device fill-related environmental variables are no longer supported:

- lobAdjacentFillGen
- lobAdjacentFillDef
- lobAdjacentFillSpacing
- lobAdjacentFillMargin
- lobTransitionFillGen
- lobTransitionFillDef
- lobTransitionFillNFins
- lobTransitionFillFingerLength
- lobTransitionFillMargin
- lobTapFillLib
- lobTapFillCell
- lobTapFillView
- lobDummyFillLib
- lobDummyFillCell
- lobDummyFillView
- lobTapFillMargin
- lobTapFillDef
- lobDevFillPhysOnly
- lobFillAsMosaic
- lobPolyFillGen
- lobPolyFillMaxPolyWidth
- lobChannelCutPolyGen
- lobTransitionCutPolyGen
- lobPolyLPP
- lobPolyWidth
- lobCutPolyLPP

- lobCutPolyWidth

IC23.1 Base

Introducing the Virtuoso Automated Placement and Routing Solution

Virtuoso® Studio integrates a comprehensive Automated Placement and Routing solution to meet the placement and routing requirements for different custom IC design styles. This placement and routing solution leverages existing technologies and new developments for improved quality of results, scalability, and user experience. The placement and routing solution focuses on the increasing requirement for layout automation. It lets you configure a placer and a router, specify their setup, and manage placement and routing results.

The Virtuoso automated placement and routing flow supports the following placement modes:

- **Automatic Placement:** Uses the options in the Auto P&R assistant to customize placement settings and run the Virtuoso automatic placer. The following placement types are supported:
 - **Device placement:** Devices are placed by running the Virtuoso device-level automatic placer. For automated device placement, use the Auto P&R assistant available in the Layout EXL and higher tiers.
 - **Standard cell placement:** Seamlessly integrates the Innovus GigaPlace™ placer in the Virtuoso environment. For standard cell placement, use the Auto P&R assistant available in the Layout MXL cockpit.
- **Interactive Placement:** Devices are placed semi-automatically. Depending on the placement needs and the complexity of the design, you can first run the Virtuoso device-level automatic placer and then use the interactive placement options to refine the placement.

This flow supports the following routing modes:

- **Automatic Routing:** Enables high-speed shape-based routing, allowing gridded or grid-less, and track-based routing of regular and power signals for physical designs. The following automatic routing types are supported:
 - **Device Routing:** technology integrates a grid (WSP) based routing solution that helps you route device level designs in Virtuoso MXL tier with a focus on advanced nodes.
 - **Standard Cell Routing:** technology seamlessly integrates the NanoRoute™ router in the Virtuoso environment. It provides different ways for you to generate WSPs as well as

route without them, relying on Innovus-created tracks.

- **Chip Assembly Routing**: technology targets top-level designs that have macro instances, I/O pads, and can also contain standard cell areas. It also addresses memory type designs using spine routing.

For all automatic routing types, use the Routing assistant available in the Layout MXL cockpit.

- **Interactive Routing** : Enables you to route connections interactively within the Virtuoso environment. These capabilities provide efficient ways to route connections in order to meet critical design constraints and rules. The interactive routing capabilities are fully enabled on all process nodes including the most advanced process technologies. For interactive routing, use the Routing assistant available in the Layout EXL and higher tiers.

The Virtuoso Automated Placement and Routing Flow also provides:

- Comprehensive pre-routing data to double-check each net in the design using **Pre-Route Browser**.
- Graphical routing result analysis using **Routing Results Browser** to check the quality of results.
- Graphical constraint management through **Routing Constraint Manager**.
- Automated **constraint checker** for verifying constraint adherence in the router result.

SKILL Functions

To manage various tasks in the Virtuoso Automated Placement and Routing flow, you can use SKILL functions. The **Virtuoso Automated Placement and Routing SKILL Functions** have been broadly classified into the following categories:

- Virtuoso Device Placement SKILL Functions
- Virtuoso Standard Cell Placement SKILL Functions
- Virtuoso Routing Environment SKILL Functions
- Virtuoso Routing Technology SKILL Functions

Environment Variables

To set default values for various placement and routing options in the Auto P&R assistant and Routing assistant, you can also use the related placement and routing environment variables.

- **Automated Device Placement and Routing Flow Environment Variables**
- **Standard Cell Placement and Routing Environment Variables**

- [Chip Assembly Routing Flow Environment Variables](#)

Invoking Virtuoso Analog Placer in the Virtuoso Automated Placement Flow

You can now use the [apPlaceAutoMatureNode](#) SKILL API to invoke the Virtuoso Analog Placer, which is available at mature nodes.

Controlling the Display of Tabs in the Array Assistant

You can now use the [adaDisableTab](#) environment variable to control the display of tabs in the Array Assistant. For example, when the environment variable is set to "GuardRing" and "Routing", these two tabs are not available in the Array Assistant. This new environment variable replaces `adaDisableRoute`, which is no longer available.

The Row/Grid Tab Renamed to the Setup Tab in the Auto P&R Assistant

Earlier, the *Row/Grid* tab of the Auto P&R assistant included options to create rows and grids. Gradually, other design set up-related options were added. Therefore, the *Row/Grid* tab is now renamed to *Setup*.

Specifying the Fill Extension Values

The *Fill* tab of the Auto P&R assistant already includes the *Extend Fill* option to specify whether the cell fill can be extended in all directions to cover the gaps within the PR boundary. This feature has now been further enhanced to let you specify the units by which the cell fill are to be extended. Two new environment variables have been introduced:




- [vgFillSpaceSpecifyUnit](#) specifies the units in which fill extensions are to be specified — `Poly pitches` **OR** `User Units`.
- [vgFillSpacePitchesOrUserUnit](#) specifies the number of poly pitches or user units in which fill devices are to be extended.

Preview Constraints by Arranging them Based on Virtual Groups

The *Constraints* tab of the Auto P&R assistant now includes the *Arrange By Virtual Groups* option, which lets you preview the placement of Modgen groups before running the placer. With this option selected, when constraints are generated, they are automatically arranged in columns according to their virtual groups.

Managing the Display of Constraint Groups

The *Constraints* tab of the Auto P&R assistant now includes the following icons to manage the display of constraint groups:

- : Expands and collapses all constraint group trees
- : Hides or shows unused, empty constraint groups.
- : Highlights or unhighlights all instances of the selected constraint groups in the layout and schematic views.

Diffusion Grid-Based Placement of Multi-Height Devices

The Virtuoso automated device placement and routing flow now supports diffusion grid-based placement, which is a new technology for rowless placement of mixed height devices. This technology creates specialized grids based on the heights of the diffusion layers of different devices in the design. These devices are snapped to the grid according to the height of their diffusion layers.

To support this flow, the *Create Diffusion Grid* check box is added to the *Setup* tab of the Auto P&R assistant. This option is displayed only when the **enablePlaceWithWsp** environment variable is set to `t`.

Filling Gaps Between Modgens During Placement

The *Placer* tab now features a new *Fill Modgen Dummies* option, which fills the gaps between Modgens selected on the layout canvas with Modgen dummies.

Differentiating Between Active Devices and Dummies

In previous releases, the constraints, placement, and fill steps of the Virtuoso automated device placement and routing flow were enhanced to support virtual groups in non-uniform designs. As an extension to this enhancement, the *Fill* tab of the Auto P&R assistant now includes the *Highlight Dummy Fills* option, which highlights all dummy devices in the layout. Click *Highlight Dummy Fills* after inserting fill devices to differentiate between active devices and dummies.

Support for a Single Instance of Multi-Fingered Fill

Earlier, the *Fill* tab of the Auto P&R assistant included a *Create As* option to specify whether a single instance or multiple instance dummy fill were to be created. The default behavior has now

changed. The fill command automatically adds a single instance of multi-fingered fill in gaps between active devices. The *Create As* option is no longer available.

What's New in Virtuoso Concurrent Layout



This topic provides a high-level overview of the new features in the IC23.1 release.

IC23.1 Base

Define a Mixed Design Partition

The Define Design Partition form has been updated to let you define both area-based and layer-based design partitions at the same time. You can also define a mixed design partition, where an area design partition is specified along with a range of layers for the design partition.

A screenshot of the "Define Design Partition: ether adc_cascode_opamp" dialog. The "Partition Type" section has radio buttons for "Area" (selected) and "Layer". Below is a table with columns: No, Name, Areas, Status, Library, Cell. At the bottom, there is an "Add / Edit Partition Definition" section with a "Partition Name" field containing "le_pl", and buttons for "New", "Attach", "Detach", "Add", and "Delete".

In previous versions, only one type of design partition could be defined at one time.

A screenshot of the "Define Design Partition: ether Bias_opamp_cascode" dialog. The "Partition Type" section has radio buttons for "Area" and "Layer" (selected). Below is a table with columns: No, Name, Areas, Layers, Status, Library. At the bottom, there is an "Add / Edit Partition Definition" section with a "Partition Name" field containing "le_pl", and buttons for "New", "Attach", "Detach", "Add", and "Delete". The "Layers" section is highlighted with a red box, showing a checkbox for "All" (checked) and a range selector from "[Bottom]*" to "[Top]*".

You can now define both types of design partitions and a mixed design partition at the same time.

What's New in Virtuoso Custom Design Migrate



This topic provides a high-level overview of the new features in the IC23.1 release.

IC23.1 ISR3

Automatically Fixing Shorts Created during Migration

You can now automatically fix shorts that are created during migration using the `shortAvoidance` argument in the schematic mapping configuration file. The default value of `shortAvoidance` is `nil`.

IC23.1 ISR2

Defining the Path of the Schematic Design Inventory File

You can now define the path of the schematic design inventory file using the `designInventoryPath` argument in the schematic mapping configuration file. The default value of `designInventoryPath` is `nil`. The default path of the schematic design inventory file is `"./"`.

SCHMAP_CONFIG/[SchDesignInventory.il](#)".

IC23.1 ISR1

Wildcard Support for Cell Name in the Mapping File

You can now use wildcard characters for the cell name in the Schematic Migration mapping file.

New Options Supported in Mapping Configuration File

The schematic mapping configuration file now supports several new options. Refer to the user guide for the full list of supported options.

New SKILL APIs

You can use the following new SKILL APIs to control the Custom Design Migration Solution:

- [schMapGetOption](#): Returns the value that is internally registered to the provided argument of the Schematic Migration configuration file.
- [schMapLoadConfig](#): Loads the configuration file for Schematic Migration.
- [schMapSaveConfig](#): Creates a configuration file with the arguments used in Schematic Migration that have values that differ from the default.
- [schMapSetOption](#): Assigns values to the arguments of the Schematic Migration configuration file.

IC23.1 Base

Introducing the Virtuoso Custom Design Migration Solution

Virtuoso® Custom Design Migration is a new design migration solution that enables you to quickly port your designs from one technology to another.

Virtuoso Custom Design Migration includes the Virtuoso Custom Schematic Design Migration Solution, which provides automated design migration for single and multiple cells from the same library or different libraries. It has the following general features:

- Instances are placed and wired similarly to the original structure
- CDF parameters are updated and follow the callback order
- Support for configurations for symbol and pin placement
- Support for syntax to define the parameter values on the target design
- Utilities to update the device mapping file
- Results browser for debugging the migrated design

What's New in Virtuoso Design Intent



This topic provides a high-level overview of the new features in the IC23.1 base release and subsequent ISR releases.

IC23.1 ISR3

Syncing Design Intent When Only Schematic is Open

It is now possible to sync a design intent in a schematic view with no corresponding layout view opened. You can do this by selecting the library, cell, and view name from the Sync Design Intents from Layout form. The Sync Design Intents from Layout form is displayed when you right-click and choose *Design Intent – Sync... – Sync All Design Intent*, or click on the *Sync* button on the toolbar.

New Design Intent columns in the Navigator Assistant

The following two columns are now available in the Navigator assistant to show the design intent information. These columns are only displayed when you are in a design intent category.

- *DI Status* column displays the implementation status of each design intent.
- *DI Profile* column displays the profile of each design intent.

IC23.1 ISR2

Design Intent Enhancements

The Virtuoso Design Intent has been enhanced with the following improvements:

- Use the [Select All Members](#) option in the shortcut menu when placing the mouse cursor over a design intent in the Navigator assistant.
- A tooltip is displayed when you place the mouse cursor on a design intent in Constraint Manager.
- Double-clicking a design intent [glyph](#) is an easy and quick way to open the Edit Design Intent form.
- A new *Elliptical* style to identify the device/pin design intent annotation has been introduced. See [Create Design Intent Form](#).
- Use the *Check and Save* button on the Design Intent toolbar to preview an alert for the [synchronization](#) required on design intents. You can now view the alert without having the need of opening the schematic and layout views together.

IC23.1 ISR1

New Design Intent SKILL Function

The [ciDIUpdateArrowDirections](#) SKILL function lets you update high current design intent arrows that do not follow the conventional current flow direction.

IC23.1 Base

Creating a Critical Net Design Intent

You can now create Critical Net design intents for identifying nets that you consider to be critical in the design. You can filter for critical net design intents using the Analyze Connectivity form in the Virtuoso Layout Editor.

Hierarchy Propagation

You can now run auto propagation across multiple hierarchies for selected nets or HighCurrent design intents. The feature lets you push nets and HighCurrent design intents up or down the hierarchy, and it also provides a consistency check to report discrepancies.

What's New in Virtuoso Design Planning and Analysis



This topic provides a high-level overview of the new features in the IC23.1 release.

IC23.1 ISR1

Reporting Coloring Violations in Remastered Routed View

The Pin Accessibility Checker can now report the coloring violations in the remastered, routed view by setting the [vfpPACRunVerifyDesignWithColorOpts](#) environment variable to true.

IC23.1 Base

Support for Coloring on Remastered Routed View

The Pin Accessibility Checker is enhanced to apply recoloring by multi-patterning engine on all remastered view so that the design remains color-correct.

New DPA Toolbar Icon

The DPA toolbar now includes the *Auto-Create Pins* icon, which can be used to launch the Auto-Create Pins form.

Preserve Area Boundary Enclosure

Use the new *preserve enclosure* field on the Design Planning and Analysis Options form to automatically resize the area boundary of a virtual hierarchy to maintain the specified enclosure value when an instance or a figGroup is moved towards the area boundary.

Specify Placement Status of Created Virtual Hierarchy using SKILL

Use the new optional argument, *placementStatus*, supported by the [IxMakeVirtualHierarchy](#) SKILL function to specify the placement status for the created virtual hierarchies.

List Cellview IDs of Made Cells using SKILL

The [IxVirtualHierarchyMakeCell](#) SKILL function can now list cellview IDs of the made cells, which makes it easy to identify the newly created cells, especially when multiple virtual hierarchies are used to create these cells.

Create Virtual figGroup Using Devices from Different Designs

When the [vhSelectiveMode](#) environment variable is set to t, use the [IxHiCreateVirtGroup](#) SKILL function to create virtual figGroups using layout devices that are bound to schematic instances from different designs.

What's New in Virtuoso Design Review



This topic provides a high-level overview of the new features in the IC23.1 base release and subsequent ISR releases.

IC23.1 ISR3

Attach Screenshot to Defect

You can now use the camera icon in the Design Review Editor form to attach a screenshot to a defect. You can attach several screenshots to a single defect, and review them by clicking on their thumbnail.

IC23.1 ISR2

Creating SKILL Callback for Checklist Items in Design Review

To quickly and efficiently automate a Design Review check, you can now create SKILL callbacks for Design Review checklists. The callback expects a single argument, and the argument passed is a disembodied property list (DPL).

IC23.1 ISR1

New Design Review SKILL Functions

The following SKILL functions have been added to let you manage design intent and design review tasks:

- The [drvReviewSetCheckList](#) SKILL function lets you reset a checklist in the specified review if the checklist name matches the given checklist.
- The [drvReportGenReport](#) SKILL function lets you generate a report summarizing the design intent in the current design based on the specified criteria and optionally opens the report in the browser.

IC23.1 Base

Virtuoso Design Review Flow

The new Design Review flow enables designers and reviewers to build the process of review and fixes in a design within Virtuoso Studio. This helps in better consolidation of information of a design. Design review ensures that all review details are located at one place for reference.

You can use the following SKILL APIs in conjunction with the Design Review flow:

- [drvCheckLists](#): Returns a list of the registered Design Review checklists.
- [drvGetReviewInfo](#): Returns information for the specified review in the given cellview.
- [drvListLayoutReviews](#): Returns the design review names for the specified layout cell view.
- [drvRegisterCheckLists](#): Registers Design Review checklists.

What's New in Virtuoso Design Rule Driven Editing



This topic provides a high-level overview of the new features in the IC23.1 release.

IC23.1 ISR2

Perform In-Design DRC Verification in Virtuoso Studio

You can now use the new **DRC Options** and Signoff Fill **Signoff Fill** forms to perform In-Design DRC verification in Virtuoso Studio. This enables you to perform DRC verification without breaking the design workflow.

minNeighborVoltageSpacing (One layer)

(Virtuoso Layout Suite EXL and higher tiers) Specifies the spacing based on a neighboring shape that meets the delta voltage condition.

minNeighborVoltageSpacing (Two layers)

(Virtuoso Layout Suite EXL and higher tiers) Specifies the spacing between two shapes, which meets the delta voltage condition based on a neighboring shape.

IC23.1 Base

Specifying the Number of Threads for DRD Editing

You can now specify the number of threads for DRD editing using the environment variable, [threads](#).

What's New in Virtuoso Electrically Aware Design



This topic provides a high-level overview of the new features in the IC23.1 release.

IC23.1 ISR2

Displaying the Check Layer Mapping Option

The *Check Layer Mapping* check box is now hidden from the *Options* menu of the EAD Browser. Set the environment variable `layoutEAD.gui hideCheckLayerMapping` to `nil` to view this option in the EAD Browser.

Specifying the Current Distribution Method to Generate EAD Datasets

You can now use the environment variable `layoutEAD.em autoDatasetDistributionMethod` to specify the current distribution method to be used when generating EAD datasets. The default distribution method is `Aspect ratio`.

Displaying the Recommended Number of Tracks

You can now use the environment variable `layoutEAD.gui showRecommendedTracks` to show the recommended number of tracks in the *EM Results* table. The default value of the environment variable is `nil`.

IC23.1 ISR1

Displaying High Precision C Extraction and High Precision R Extraction Options

The *High Precision C Extraction* and *High Precision R Extraction* options are now hidden by default. Set the environment variable `layoutEAD.gui hideHPRCExtraction` to `nil` to view these options in the context menu when you right-click any net in the *Net Summary Table* of the EAD Browser.

Running EM Analysis using a SKILL Function

Use the SKILL function `eadRunEM` to run EM analysis on all nets or a list of specified nets in the specified layout design.

Exporting Data from the Net Summary Table using a SKILL Function

Use the SKILL function `eadSaveSummaryTableToCSV` to export data from the *Net Summary* table of the EAD Browser to the specified CSV file.

Retrieving a List of All EM Datasets Present in the Current Design

Use the SKILL function `eadGetAllEMDatasets` to retrieve a list of EM datasets in the current design.

Setting a Dataset as the Active Electrical Dataset

Use the SKILL function `eadSetCurrentDataset` to set the specified dataset as the active electrical dataset in the EAD Browser of the specified layout cellview.

Calculating Vmin and Vmax Values for a Terminal in a Dataset

Use the SKILL function `elecGetVoltageData` to calculate V_{\min} and V_{\max} values for the specified

terminal in a dataset.

IC23.1 Base

Quantus-Pegasus Integration with EAD for Partial Layouts

This feature provides sign-off level accuracy by combining parasitics extraction and physical verification sign-off capabilities of Cadence® Quantus™ Extraction Solution and Cadence® Pegasus™ Verification System, respectively. This flow requires all devices to be placed in the layout; the position of these devices may however, not be the final. This flow enables the user to specify a reference net, known as the ground net, before extraction rather than before simulation. This enables quicker feedback loop to run layout-aware re-simulation and to tune the circuit performance early without waiting for a complete LVS-clean layout

What's New in Virtuoso Floorplanner



This topic provides a high-level overview of the new features in the IC23.1 release.

IC23.1 ISR3

Moving Pin Labels to Stamp Label Layers Defined in the Technology File

During Assisted Move, you can now move labels to the stamp label layers that are defined in the technology file. The *Match Label Layer* section in the Pin Tool Options form now includes the *Use Stamp Label Layers* option, which you can select to honor the stamp label layer definition in the technology file while running Assisted Move.

The related environment variable `astPinMoveMatchLabelLayerChoice` now accepts a new option, `Use Stamp Label Layers`.

Aligning Connected Pins in the Pin Tool

While running various pin-related operations in the Pin Tool, you might want to quickly align pins. Earlier, you had to use an object alignment solution outside the Pin Tool.

The Pin Tool is now enhanced to let you select the required pins in the Pins Browser and quickly align connected pins using the *Align to Connected Pin* command in the shortcut menu.

Usability Enhancements in the Pin Connectivity Settings Form

The Pin Connectivity settings form has been further enhanced for your convenience.

- Must-Connect groups are displayed in the *Net* mode.
- The *Filter* drop-down list includes a *Shorted Terminals* option that lets you display only the shorted terminals in the tree structure.
- The complete path to pin figures is displayed in the header in *Net* mode.

Specifying the Offset of IO PADs in IO Rows

As part of IO planning, IO PADs are placed in IO rows. Earlier, the positions of IO PADs were determined by the PAD spacing you specified in the IO PAD Placement form. In addition to this, the form now includes an *Offset* section, which lets you define the position of the first IO PAD along each edge. Now, *Offset* determines the position of the first IO PAD and *Spacing* determines the positions of the remaining IO PADs in each direction.

IC23.1 ISR2

Matching Label Layer Purpose Pair in the Pin Tool

(Layout EXL Only) You can use the *Match Label LPP* shortcut menu option in the Label Browser of Pin Tool to match the label layer purpose pair to the associated pin labels.

Shorting and Unshorting Terminals

The terminals that belong to the same net are called shorted terminals. The Pin Connectivity Setting form lets you short terminals in three ways - using the drag and drop operation, using the *Short Terminals* and *Unshort Terminals* options from the shortcut menu, or by editing the net name. You can also use the form to unshort the shorted terminals.

Pushing Fill and Routing Blockages

Use the `pibCreateBlockageType` environment variable to push routing objects as both fill and routing blockages simultaneously when you run the Push Into Blocks command.

Specifying Spacing Constraints for IO Pad Cells

In the IO PAD Placement form, the *Custom* option is renamed to the *Custom Sites* option, which denotes the number of sites instead of the absolute distance between consecutive IO PADs for pad placement.

You can use this option to provide different spacing for IO pad cells within the same row. Custom site values are mapped to the pitch value specified in the alignment constraint.

Usability Enhancements in the Pin Tool Browser

The Pin Tool Browser now supports the following enhancements:

- You can use the *Signal Type* column to view the signal type of the net connected with a pin.
- The *Type* column is renamed to the *Direction* column.
- The *PGG* column is now hidden by default.

IC23.1 ISR1

Inserting Multiple Filler Cells

Inserting multiple filler cells is now simpler because of the following enhancements to the *Select Filler Cells* section of the Insert Filler Cells form:

- You can filter `lib:cell:view` names for the filler cell by using the *Filter* button.
- The right arrow button lets you add the specified `lib:cell:view` as a filler and the left arrow button lets you delete the selected `lib:cell:view` from the filler table. You can now select multiple filler cells at a time.
- The *Browse*, *Add Filler*, and *Remove Filler* buttons are no longer available.

Placing Corner Pads in the Inner I/O Rows

The Corner Pad Placement form now supports the *Place Corner Pads On Inner Row (applicable for Two Rows Per Side)* option to place the corner pads aligned to the inner rows in a two-row-per-side layout design.

Creating Boundary or Buried Pin on Whole Shapes

You can now create a boundary or buried pin on the largest whole shape that is present inside the PR boundary.

Copying Unattached Labels from Source to Target Cellview

(Layout EXL Only) The Load Physical View form now supports the *Update Unattached Label* option that lets you copy the unattached labels from the source to the target cellview.

Related SKILL function: [vfpLoadPhysicalView](#)

Spacing I/O Rows During I/O Row Creation

You can now use the *Row Spacing* option in the Create IO Row form to create spacing between two IO rows in *Two Rows* mode. This ensures DRC correctness between the IO rows.

IC23.1 Base

Modifying the Site Definition Width During I/O Row Creation

You can now change the width of the site definitions to abut rows during IO row creation.

This allows you to have some overlapped rows to ensure that pad placements are abutted. In case of IO placement for two rows, if the cells are of exact height as the rows, then adjusting the siteDefs allows abutted placement.

Customizing Pin Layer in the Pin Planner

Use the `vfpTopLevelRouteEnableCustomPinLayer` environment variable to enable custom pin layer when aligning level-1 pins to top-level route in the Pin Planner.

Display Enhancements in the Analyze Connectivity Form

You can now view the change in net count and line width in *Block* mode when you switch between *All*, *Common*, and *Exclusive* options in *Net count and length* field of the Analyze Connectivity form.

When multiple blocks are selected in *Block* mode, the net count and length is displayed at the center of the larger block.

Adding Pins On Different Purposes in the Pin Tool

Use the Add Pin On Different Purposes form to add any existing pin on specific purposes from Pin Tool.

Usability Enhancements in the Analyze Connectivity Form

The Analyze Connectivity form now supports three new options:

- Use *Mode* to select *Block* or *Net* mode to display connectivity information or net selection between the selected objects.
- Use *Critical nets only* to show connectivity information only for nets you consider to be critical in the design.
- Use *Supply nets* to filter the connectivity information or net selection based on the selected type of net - *Power*, *Ground*, or *Global*.

Promote Pins Form Reorganized for Better Usability

In the Promote Pins form, all the pin promotion fields have been reorganized into a separate progressive disclosure for improved usability.

Usability Enhancements in the Pin Checker Form

The Pin Checker form now supports two new options:

- Use *Pin to wire overlap* to check for physical overlaps between pins and wires.
- Use *Check duplicate pins* to check for duplicate pins in the layout.

Customizing Pin Dimensions for Moving Pin to Top-Level Route

You can use the `vfpTopLevelRouteEnableCustomPinSize` environment variable to enable custom pin sizing while moving a pin to a top-level route using the Pin Planner.

Controlling the 'PIN_ON_ROUTE' Property

You can use the `vfpTopLevelRouteCreatePinOnRouteProp` environment variable to control the creation of the `PIN_ON_ROUTE` property. The `PIN_ON_ROUTE` property can be set when the pin placement edge is set to top-Level Route.

Creating a Pin for Back Side Metal Layer Shapes

You can now use the `autoPinTopBackMetalLayer` environment variable to consider back side metal layer as the top metal layer and create pins for the specified shapes.

Snapping Promoted Pins

(Layout EXL Only) You can now use the Promote Pins form to snap the pins and labels to match the direction of target LPP. This feature is applicable only for advanced node designs that contain WSP tracks.

Adding Pins and Geometries for Layers and Nets

(Layout EXL Only) The Load Physical View form now supports two new options:

- Use *Add Pins and Geometries for Specified Layers* to selectively transfer pins, wires, and shapes on the specified metal and poly layers.
- Use *Add Pins and Geometries for Selected Nets* to selectively transfer pins, wires, and shapes on the selected nets.

Usability Enhancements in the Pin Spacing Form

You can now use *Pin Offset* option in the Pin Spacing form to set the pin spacing offset value from the specified PR boundary edge.

Creating Pin and Via Stacks using the Pin Tool

(Layout EXL Only) You can now use the new Create Stacked Pin form to create a stack of pins and vias on the selected metal layers using the Pin Tool.

Specifying the Blockage Type for Routing Objects

The *Push Into Blocks* command now honors the `pibCreateBlockageType` environment variable when it pushes the routing objects into the block level as routing blockages. This environment

variable sets the blockage type for these routing structures.

Mapping Layer Purpose Pairs in the Load Physical View Form

(Layout EXL Only) You can now map instances, pin layers, and layer purpose pairs for selected pins and shapes using one of the following methods:

- Selecting *Mapping File* on the Load Physical View form.
- Using the `vfpLoadPhysicalView` SKILL function. The function now supports the `updateSelectedNets` and `mapperFile` arguments.

Support for Layer Independent Pin Spacing

You can now specify the layers on which pin spacing settings must be applied by using one of the following methods:

- Selecting *Layer Options* on the Pin Spacing form.
- Enabling the `pinLayerOption` environment variable.

Displaying Soft Block Cell Types in Block Annotations

You can now display the cell types of soft blocks in block annotations using one of the following methods:

- Selecting *Cell Type* on the Block Annotations Options form.
- Enabling the `blockAnnotationCellType` environment variable.

What's New in Virtuoso Fluid Guard Ring



This topic provides a high-level overview of the new features in the IC23.1 release.

IC23.1 ISR1

You can customize the Create Fluid Guard Ring form using the SKILL functions listed below. These functions enable you to add UI controls and tables to the Create Fluid Guard Ring form.

- `xfgrAddRow`
- `xfgrAddTable`
- `xfgrDeleteRow`
- `xfgrGetCellProps`
- `xfgrGetColumnProps`
- `xfgrGetFormSize`
- `xfgrGetGroupProps`
- `xfgrGetOptionProps`
- `xfgrGetTableProps`
- `xfgrGetTotalTableRows`
- `xfgrPrintTable`

- `xfgrResetTable`
- `xfgrSetCellProps`
- `xfgrSetColumnProps`
- `xfgrSetFormSize`
- `xfgrSetGroupProps`
- `xfgrSetOptionProps`
- `xfgrSetTableProps`
- `xfgrTableCellValueChangedCB`
- `xfgrTableRowAddedCB`
- `xfgrTableRowDeletedCB`

IC23.1 Base

Customizing the Create Fluid Guard Ring form

You can customize the Create Fluid Guard Ring form using the SKILL functions listed below. Some customizations that you can perform are add options, enable or disable options, show or hide groups, show or hide options in groups, and set default values for options.

- `xfgrAddOption`
- `xfgrConfigureOptionsCB`
- `xfgrDisableOption`
- `xfgrEnableOption`
- `xfgrGetAvailableValues`
- `xfgrGetOptionValue`
- `xfgrHideAllGroups`

- `xfgrHideAllOptions`
- `xfgrHideGroup`
- `xfgrHideOption`
- `xfgrInitializeOptionsCB`
- `xfgrOptionValueChangeCB`
- `xfgrPostCreateCB`
- `xfgrPreCreateCB`
- `xfgrSetAvailableValues`
- `xfgrSetOptionValue`
- `xfgrShowAllGroups`
- `xfgrShowGroup`
- `xfgrShowOption`

What's New in Virtuoso Hierarchy Editor



This topic provides a high-level overview of the new features in the IC23.1 release.

IC23.1 ISR1

Setting Default Template Name in the Use Template Form

You can use the `name` environment variable to set the default name of the template in the Use Template form.

IC23.1 Base

Displaying Icons in the Information Column

You can now view icons associated with settings in the information column of the Cell and Instance tables by specifying the following environment variables:

- `hed.cellTable showInfo`: Specifies whether to display the icons associated with settings, such as stop points and file bindings, in the information column of the Cell table.
- `hed.instTable showInfo`: Specifies whether to display the icons associated with settings,

such as stop points and file bindings, in the information column of the Instance table.

Checking for Occurrence Rules in a Configuration

You can now use the `hdbConfigHasOccRules` SKILL function to determine whether the specified configuration ID contains any occurrence rules.

What's New in Virtuoso Import Tools



This topic provides a high-level overview of the new features in the IC23.1 release.

IC23.1 Base

Rapid Import Flow Enabled by Default during Check and Save

The default value of the `vmsEnableRapidImport` SKILL flag has been changed from `nil` to `t`. This means that the Rapid Import flow is enabled by default when you invoke the *Check and Save* command for text cellviews.

What's New in Virtuoso Interactive and Assisted Routing



This topic provides a high-level overview of the new features in the IC23.1 base release and subsequent ISR releases.

IC23.1 ISR2

Support for the `allowedJogWidths` Constraint

The [allowedJogWidths](#) constraint is now supported for the interactive routing commands.

Copy Route Enhancements

The width control for the connected pin sections have been improved to provide flexibility.

Also, the `copyRouteMatchPinWidth` environment variable has been removed. Instead, the following new environment variables have been introduced.

- [copyRouteEnableWidthAtEnds](#)
- [copyRouteWidthModeAtEnds](#)
- [copyRouteWidthOverrideAllLayersAtEnds](#)

See [Copy Route Form](#) and [Copy Route tab](#) in the Routing Assistant.

The copyRouteMultilayerMode Environment Variable Renamed

The copyRouteMultilayerMode has been renamed to [copyRouteLayersMode](#) to accommodate changes in the layer mode use model of the copied routes. The *off* option has been removed and the *forPinsOnly* option is now renamed to *sameAsReference*.

IC23.1 ISR1

Removed Features

The Via Alignment options in the *Wire* and *Bus* tabs of the Routing Assistant have been removed.

IC23.1 Base

Routing Assistant - Interactive Mode

The Virtuoso automated placement and routing flow also supports the interactive routing mode. For the interactive routing mode, use the Routing Assistant available in the Layout EXL and higher tiers to route connections interactively within the Virtuoso environment. The interactive routing capabilities provide efficient ways to route connections in order to meet critical design constraints and rules. These capabilities are fully enabled on all process nodes including the most advanced process technologies.

Copying Routes during Interactive Routing

The new *Copy Route* command lets you quickly, reliably, and incrementally generate thousands of bus wires. You start the command by using one of the following methods:

- Choose *Edit – Wiring – Copy Route* from the layout window menu bar to open the [Copy Route](#) form.
- Click the *Start Copy Route* button at the bottom of the [CopyRoute](#) tab in the Routing assistant.

The *CopyRoute* tab is displayed only when you have selected *Interactive* routing mode.

- Call the new `weHiCopyRoute` SKILL API.

There is also a full set of environment variables provided, which you can use to set the default behavior for the feature. See [Copy Routing Environment Variables](#) for more information.

Also see [Copying Pin-to-Wire Routes](#), [Copying Pin-to-Pin Routes](#), and [Handling Wire Copies for EM Analysis](#).

Creating Packaging Buses in Virtuoso RF

The *Create Packaging Bus* command lets you create packaging buses in Virtuoso RF Solution. The command supports area-based rules and the settings are controlled by using the new [Create Packaging Bus](#) form. Also see, [Snapping Behavior of Curved Paths Using Create Packaging Bus](#) and [Supporting Area-Based Rules](#).

`allowedLengthRanges` Constraint

Support for the *exceptEndToEndGapRange* parameter has been added for the interactive routing commands.

Removed Features

- The Finish Wire and Finish Bus features are deprecated and no longer available by default for Interactive and Assisted routing. The related option have also been removed from the Create Wire and Create Bus context-sensitive menus.
- The Constraint-Aware Editing option has been removed from the Create Wire, Create Bus, and Point to Point context-sensitive menus.

What's New in Virtuoso Layout Suite EXL



This topic provides a high-level overview of the new features in the IC23.1 release.

IC23.1 ISR2

Selecting a Group of Replay Commands Using Capture and Replay Assistant

You can now use the *Group* field in the [Capture and Replay Assistant](#) to display different set of recordings into different tabs for easy access.

Setting Default Group for the Capture and Replay Assistant

You can now use the [carDefaultGroup](#) environment variable to specify the default group when you open the Capture and Replay assistant.

Renaming a Recording in the Capture and Replay Assistant

You can now use the *Rename* option to change the name, description, and group category of any recording using the [Rename Command Name](#) form in the Capture and Replay assistant.

IC23.1 ISR1

Auto Via Assistant

Virtuoso Layout EXL and higher tiers now include a new Auto Via Assistant, which lets you analyze vias in your design to identify appropriate locations to create vias. The assistant includes two tabs, *Analyze*, which lets you analyze vias in your design and *Create*, which lets you create vias automatically at the suggested locations in the design from the analysis.

What's New in Virtuoso Layout Suite MXL



This topic provides a high-level overview of the new features in the IC23.1 release.

IC23.1 Base

Viewing Cross Sections of a Layout

You can now view cross sections of physical layers in a layout by using the cross section viewer, which displays the different layers, from top to bottom, present in a cross section. This enables you to identify any missing or wrongly placed layers and to debug connection issues across layers. You can choose to display the cross section in the Cross Section assistant or in a floating window. You can use the [Cross Section Viewer Form](#) to customize the settings for displaying a cross section.

What's New in Virtuoso Layout Suite XL



This topic provides a high-level overview of the new features in the IC23.1 release.

IC23.1 ISR3

Abutting Using `sdFirst` during `syncChain`

You can now use the `chainSyncChainsSDFirst` environment variable to specify whether synchronizes chains should attempt to abut using the `sdFirst` strategy which orients the devices so that the Source/Drain nets match for any subsequent abutment using the `dummyFirst` strategy.

Backannotating Library and Cell of a Layout Device as the Schematic Master

You can now use the `backAnnotateIgnoreBoundMasters` environment variable to specify whether backannotation should use the library and cell of a layout device as the schematic master.

Controlling Equivalence Between `<>` and `[]` During XL Compliance Check

You can use the `checkTermLabelAllowBracketDiff` environment variable to specify whether the difference between the brackets `[]` and `<>` be ignored during an XL Compliance terminal label check.

Checking for Equivalent Figures in a Synchronous Clone Family

You can now use the `lxCloneGetEquivalentFigs` SKILL function to retrieve the list of equivalent

figures in other clones of the synchronous clone family.

Controlling Whether Illegal Hierarchical Connections are Reported the XL Status Shorts

You can now use the `xlStatusHierShorts` environment variable to specify whether illegal hierarchical connections are reported as XL status shorts in Navigator during interactive editing.

Specifying Controls for Stop Layers during Extraction

You can now specify the following options on the *Option – Connectivity – Connectivity – Control* tab to control stop layers during extraction:

- *Promote all stop shapes* to specify whether stop shapes at the `extractStopLevel + 1` level are visible to the extractor.
Environment variable: `extractPromoteAllStopShapes`
- *Ignore color lock state* to specify whether the lock state of shapes should be ignored during extraction.
Environment variable: `extractIgnoreColorLockState`

Saving the Results of Find and Replace Operations into a File

You can now use the *Reports* section in the Find/Replace form to save the results of find and replace operations into a file.

- Enable the *Save Report To* option for the find and replace operations.
Environment variable: `saveFindReplaceReportToFile`
- Specify the input file name to save the detailed report of the find and replace operations.
Environment variable: `findReplaceReportFileName`
- Generate a detailed report for any find and replace operation and show the report of *Find*, *Replace* and *Replace All* operations in the CIW.
Environment variable: `showFindReplaceReport`

Snapping the Coordinates of a Chopped Shape to the Manufacturing Grid

You can snap the coordinates of a chopped shape to the manufacturing grid by using the *All edges to manufacturing grid* option added in the Chop form.

Zooming the Stretch Handle Display

You can now use the `minInstSizeInPixelsForStretchHandles` environment variable to display the stretch handle according to the specified pixel value.

Setting the Snap Selection by Giving Preference to an Instance as Reference Object

You can now use the `snapUsingInstanceAsReference` environment variable to select an instance as the reference object for snapping.

Synchronizing the Display of Shapes and Short Markers in the Cross Section Viewer with the Palette Visibility

You can now use the *Sync With Palette Visibility* option in the cross section viewer form to synchronize the display of shapes and short markers in the cross section viewer with the palette visibility.


Environment variable: `crossSectionSyncWithPaletteVisibility`

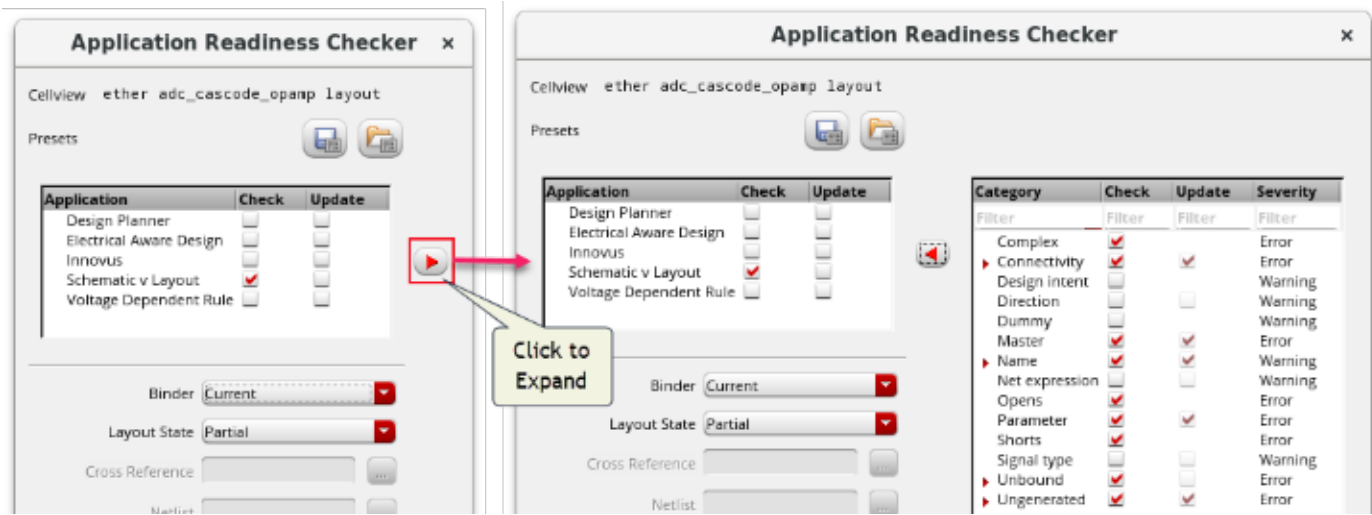
Ability to Select and Add Nets/Instances from the Dynamic Selection Assistant

You can now use the `geDSAGetSelectedSet` SKILL function to get the hierarchical selected set from the Dynamic Selection assistant.

IC23.1 ISR2

Enhancements in the Application Readiness Checker Form

Application Readiness Checker by default opens in the compact form and you can click  to display the Check and Update pane. The Check and update pane supports a filter that lets you specify the check and update criteria for the categories you want to view.



Making Stop Shapes at Extract Stop Level + 1 Visible during Extraction

You can now specify the `extractPromoteAllStopShapes` environment variable to ensure that the stop shapes at `extractStopLevel + 1` are visible to the extractor.

Removing Ignore Property in Layout

The `bindRemoveIgnores` environment variable lets you remove the ignore property from the layout instances that can be bound to schematic instances.

Retrieving Connectivity Sources of Markers generated during Connectivity Extraction

The `lceGetMarkerConnectivitySources` SKILL function lets you retrieve the closest connectivity sources corresponding to the markers generated by the connectivity extractor.

Updating Terminal Names when Binding

You can now use the `bindUpdateTermName` environment variable to update the name of the terminal when binding using the Define Device Correspondence form or by using the *Bind* option on the right mouse button shortcut menu.

Possibility to chop group array

The *Chop* command now supports group array same as mosaic. See [Array Chopping](#).

Verbosity message display

You can now use the `commandVerboseLevel` environment variable to display the verbosity message for commands such as *Copy*, *Move*, *Stretch*, *Delete*, *Flip*, *Create Measurement*, *Create Rectangle*, and *Create Instance*, in the log file or CIW.

Enabling or disabling the list of layer transitions

The `viaSetValidTransitions` SKILL function lets you enable or disable the list of layer transitions.

Copying layer shapes from one layer to another

The `leCLSCopyLayerShapes` SKILL function lets you Copy all the shapes from one layer to another that you have selected on the Copy Layer Shapes form. This function also allows you to delete previously created shapes from the Copy Layer Shapes form.

IC23.1 ISR1

Enhancements in the Application Readiness Checker Form

You can now specify:

- The path to the PVS LVS .ixf instance cross-reference file using the *Cross Reference* option.
- The path to the PVS LVS .net or .spi extracted netlist file using the *Netlist* option.
- Whether to run check and update in the current cellview, selected set, all cells in library, or all cells in the design using the options in the *Scope* drop-down list.







Enhancements in the Back Annotate Dummies Form

In the Back Annotate Dummies form, you can now:

- Update the layout of instances in a grid to be as close to a square aspect ratio using the



button.

- Switch between column-priority packing and row-priority packing for the grid of instances being placed down using the  and  buttons.
- Change orientation of the dummy instances using the orientation buttons , , , and .

Controlling Chaining Behavior

You can now control whether dummy devices can be changed during chaining by using the [chainDummyUpdate](#) environment variable.

Setting Limit for Incremental Check Against Source

You can now control the number of objects that can be processed by the [incrementalCas](#) variable at one time by using the [incrementalCasLimit](#) environment variable.


Specifying Component Classes NGAA and PGAA in CPH

(Virtuoso Advanced Node Option for GAA) You can now specify the component classes NGAA and PGAA for Gate All Around transistor devices in Layout EXL or higher tiers.

Specifying Connections Requiring an Extra Layer

You can now use a `stdViaDef` to define a connectivity model in which an extra layer is required to connect to the diffusion layer.

Arranging Postselected Objects in Square Format in an Array

You can now arrange selected objects in an array in square format using the  button.

Updating Schematic Parameters for Layout Instances

You can now update the schematic instance parameters for hierarchical layout instance masters or layout instances bound to the top-level schematic instances by using the [IxHierUpdateSchematicParameters](#) or [IxUpdateSchematicParameters](#) SKILL functions.

IC23.1 Base

Application Readiness Checker Form

Application Readiness Checker has been introduced to improve the process of checking and updating the layout from the source and to provide a one-to-one correspondence between the checks and updates, with the ability to choose the checks you want for different applications. You can also extend the checks by registering SKILL-based check and update tests. For LVS-complete designs, the tool automatically updates the schematic and layout correspondence to the best match before running the checks and updates.

Schematic Assistant

Schematic Assistant is a docked assistant now available in Layout XL and higher. It lets layout designers easily refer and use the schematic in the assistant instead of referring the schematic in a separate window. Schematic Assistant is a read-only view that supports zoom, probe and cross-select along with descend options.

The assistant can be made the default option without opening the full schematic window in the background by setting *Layout – Connectivity – Connectivity Reference – Open With: No Schematic*

Enhanced Search Criteria

You can now search for paths and pathSegs using the `length` search criterion. Also, you can now use the `area` criterion to search for rectangles, donuts, ellipses, labels, polygons, paths, and pathSegs.

Enhanced Dummy Instances BackAnnotation Flow

Enhanced interactive back annotation to the schematic for dummy instances to keep the two views synchronized. [Back Annotate Dummies Form](#) has improved row and column placement and support for mfactored or iterated instances.

Related Environment Variable: [backAnnotateInstances](#)

Related SKILL Function: [IxHiBackAnnotateDummies](#)

Enhancements in Automatic Chaining

The automatic chaining process has been to enhanced to let you:

- Optimize the chains using the [chainOptimize](#) environment variable or by setting the *Chain optimize* option in the *Device Chaining* section on the *Generation* tab of the [Connectivity](#) form.
- Control the effort applied during chaining by using the [chainEffort](#) environment variable or by setting the *Chain effort* option in the *Device Chaining* section on the *Generation* tab of the Connectivity form.
- Control if the post process to combine distinct chains into one chain by abutting on the dummy pins is performed by using the [chainDummyAbutment](#) environment variable.

Incremental Check Against Source

Check against source can now run dynamically to create markers and glyphs on canvas when an edit command introduces differences between the layout and the schematic. This lets you get instant feedback and enables you to correct the issues and maintain correspondence with the schematic. This is enabled using the `incrementalCas` environment variable

Applying Previous Data from the Property Editor Cache

After you have edited the properties of an object, you can now apply the same changes to another object by using the *Apply Previous* button in the Edit Properties form for the object. To view the previously applied data, you can use the [lePIGetApplyPrevData](#) SKILL function.

Capturing and Replaying Design Actions

With this release, you can record a series of actions in a design and replay them across cellviews and sessions by using the Capture and Replay assistant. In addition to the assistant, you can use the following SKILL functions to capture and replay your actions: [leCARStartCapture](#), [leCARStopCapture](#), and [leCARCommand](#).

Checking if Extracted Shape is Trimmed

You can now use to [lcelsShapeTrimmed](#) SKILL function to check if the specified shape has been trimmed during connectivity extraction.

Checking Points Within a Polygon

You can check whether a point lies inside a polygon by using the `geIsPointInsidePolygon` SKILL function.

Creating Repeated Patterns Using Group Arrays

You can now use group arrays to quickly create repeated patterns of objects in your designs. You can easily customize group array attributes such as the number of rows and columns, spacing between cells, and orientation of individual cells. You can also activate a lock to ensure group arrays are created using a predefined spacing mode and values. You can create group arrays by using the [Copy](#), [Generate Selected Components](#), or [Generate Clones](#) forms or the [IxCreateGroupArray](#) SKILL function.

Determining Application Tiers

You can now use the following SKILL functions to find out the application tier to which an application belongs: [lelsEXLAppOrAbove](#), [lelsLayoutAppOrAbove](#), [lelsMXLAppOrAbove](#), [lelsXLAppOrAbove](#), [leLayoutAppNames](#), and [leLayoutViewTypes](#).

Dimming a Design in Gray

You can now use the `dimmingInGray` environment variable to dim a design in gray. This helps you to follow highlighted nets when tracing nets in a hierarchy.

Displaying Ruler Length at the End of a Ruler

You can now use the `displayRulerDistanceAtEnd` environment variable to display the ruler length at the end of a ruler during the drag operation when you create a single- or multi-segment ruler.

Enabling Partial Selection of Vias and Mosaics Using the Stretch Form

You can now enable partial selection of vias and mosaics by selecting the `Mosaic Partial Selection` and `Via Partial Selection` options on the Stretch form. Earlier these options were available only on the Selection Options form.

Extracting Off-Pin Connections Using the Connectivity Extractor

You can now use the `extractDeepOffPinConnectionsToLevel` environment variable to enable the connectivity extractor to extract off-pin connections. It also enables the detection of shorts and opens for off-pin connections.

Launching Layout MXL from Schematic Window

You can use the [IxLaunchLayoutMXL](#) to launch Layout MXL from a specified schematic window.

Propagating Net Information

The Propagate Nets form is no longer available. Selecting the *Connectivity – Nets – Propagate* command now opens the *Connectivity* tab of the Edit Instance Properties form for the selected instance. You can use the tab to propagate the net information from an instance in your cellview. When you hover over a terminal or net name on the *Connectivity* tab, that terminal or net now gets highlighted on the canvas, enabling you to easily identify it in the design.

Snapping Chopped Coordinates to the Manufacturing Grid

You can now snap chopped coordinates to the manufacturing grid by using the *Modified edges to manufacturing grid* option on the Chop form.

Specifying the Copy Type

In the Copy form, you can now specify the type of copy to be created as *Single*, *Step*, or *Array*. You can create a single copy of objects, a line, or an array of copied objects.

Related environment variable: [copyOrder](#)

Specifying the Method for Saving Visible Traces in a Design

In the Save Traced Nets View form, you can now specify the method to be used for saving visible traces in a design. You can choose to save each visible trace in a separate view or save all visible traces into a single view. You can also let the system to automatically determine which of these methods should be used.

Related environment variable: [netTracerSaveMethod](#)

Specifying Details for Displaying PR Boundary Interior Halo Shape

In designs with 3NM or 2NM technology libraries, the Generate All from Source lets you create a PR Boundary interior halo shape from the supported snap patterns in 3NM and derived layer in 2NM that have a minPRBoundaryInteriorHalo technology file constraint. The interior halo is maintained during interactive editing commands. You can display the option form using the [IxHiPRBoundaryInteriorHaloSKILL](#) function.

Specifying Highlight Color for Rows in Annotation Browser Tree View

You can now use the [abSetColorOnSelection](#) SKILL function to set the color to highlight all selected rows in the Annotation Browser tree view.

What's New in Virtuoso Module Generator



This topic provides a high-level overview of the new features in the IC23.1 release.

IC23.1 ISR3

Mapping Active Libraries with Dummy Libraries in Modgen Arrays

Certain PDKs require you to map an array dummy master and its parameters with a matching active master and its parameters. The following SKILL APIs are now available to manage the task:

- **mgSetDummyLibCellAndParamAlias**: Maps the specified active library name and cell names to the specified dummy library and cell names.
- **mgGetDummyLibCellAndParamAlias**: Returns all currently registered Modgen dummy library and cell mapping and parameter name mapping lists.
- **mgClearDummyLibCellAndParamAlias**: Clears all Modgen dummy library, cell, and parameter name aliases.
- **apPDKSetupGetModgenDummyAlias**: Retrieves a list of PDK-specific Modgen dummy library, cell, and parameter name aliases registered for the specified PDK in the Virtuoso automated placement and routing flow.

IC23.1 Base

Modgen Transparent Editing Mode

In Layout MXL, you can use Modgen Transparent Editing mode to modify a Modgen directly from the top level. Modgen Transparent Editing mode provides all the functionalities of Modgen Editing mode, but with the top-level canvas object in context. The Virtuoso workspace menus and toolbars are available.

Customizing the Pattern Preset List in the Array Assistant

You can now customize the pattern presets that are listed in the *Pattern Preset* drop-down list of the Array Assistant. The new `patternPresetItemList` environment variable lets you add or remove presets from the list and change their sequence as per your design requirements.

New Array Assistant Replaces All Older Modgen Placement Solutions

In previous releases, the Modgen Pattern Editor, the Grid Pattern Editor assistant, and the Grid Pattern Mapping assistant were used to define Modgen grid pattern and other placement settings. In Layout EXL, the Auto Device Array assistant was used for the same purpose. There were multiple distinct solutions providing similar functionality.

IC23.1 supports a single abstracted graphical array placement solution—The Array Assistant. The Array Assistant serves as a one-stop solution to define all Modgen placement-related settings.

The Auto Device Array assistant has been now renamed to Array Assistant and is made available in both Layout XL and Layout EXL.

In Layout XL, the Array Assistant lets you specify the Modgen placement settings.

In Layout EXL, the Array Assistant lets you specify the Modgen placement settings, the guard ring options, the topology and routing settings, and the reuse options.

The Modgen Pattern Editor, the Grid Pattern Editor assistant, and the Grid Pattern Mapping assistant are no longer supported.

The `mgPatternCB` SKILL API has also been removed.

Removed Features

The Modgen Routing Toolbar

Starting IC23.1, the Modgen Routing toolbar is no longer included in the Modgen Editor. Use the [Modgen Topology and Routing SKILL API](#) to define Modgen topology and routing settings.

What's New in Virtuoso Multi-Technology Solution



This topic provides a high-level overview of the new features in the IC23.1 release.

IC23.1 ISR3

Padstack Replacement

You can now replace the pads that were created based on the IC layout shapes with the real pads created in the package design. It is essential to perform this replacement in Virtuoso Multi-Technology or Allegro platform during the generation of a die abstract when exporting the die.

IC23.1 ISR2

Cell Replacement

The new cell replacement functionality has been integrated in the Virtuoso Multi Technology Enablement form as an optional step. It provides the capability to ease the replacement of cells having pin mismatches when replacing die or package abstract views from Allegro designs by Edit-in-Concert abstract views generated during the die or package export.

IC23.1 Base

Creating Unified Libraries from DE-HDL Libraries

Now, you can convert native libraries into multi-platform unified library components across the platforms by using the Convert DE-HDL libraries to Unified Libraries form.

Licensing Updates in Virtuoso MultiTech Framework

Now, Virtuoso MultiTech Framework and Virtuoso Electromagnetic Solver Assistant are available by default in Layout MXL.

Virtuoso MultiTech Framework Setup

Now, Virtuoso MultiTech Framework requires Allegro® Package Designer Plus with SiP Layout option. Allegro database must be upgraded to version 17.4 or later for use in Virtuoso MultiTech Framework.

SiP DRC Checker

The SiP DRC checker lets you check DRC issues in package layout within the Virtuoso RF Solution environment. It leverages Allegro's DRC checking engines in the background.

What's New in Virtuoso Multi-Patterning Technology



This topic provides a high-level overview of the new features in the IC23.1 release.

IC23.1 Base

Specifying the Levels of Hierarchy during Color Checks

You can now specify the levels of hierarchy to be considered during color checks using the *Hierarchy Range* option on the Color Checks form.

What's New in Virtuoso NC-Verilog Environment



This topic provides a high-level overview of the new features in the IC23.1 release.

IC23.1 ISR2

Printing Single Netlist Modules in Alphabetical Order

Use the `hnlVerilogPrintModulesByOrder` SKILL flag to print the single netlist modules in alphabetical order.

What's New in Virtuoso Parameterized Cell



This topic provides a high-level overview of the new features in the IC23.1 release.

IC23.1 ISR1

Checking the Library Version in the Express Pcell Cache

You can now use the `xpcGetPDKVersion` SKILL function to check the version of the library or libraries stored in the Express Pcell cache.

What's New in Virtuoso Parasitic Aware Design



This topic provides a high-level overview of the new features in the IC23.1 release.

IC23.1 ISR3

Changes in the DSPF Syntax of Smart View Netlists

The syntax of DSPF netlists generated for Smart View has been changed and provides significant performance improvement by reducing memory peak usage and netlisting time.

What's New in Virtuoso Photonics Solution










This topic provides a high-level overview of the new features in the IC23.1 release.

IC23.1 ISR1

Enhancements in the Generate Selected From Layout Form

You can now use the Generate Selected From Layout form to:

- Update the layout of instances in a grid to be as close to a square aspect ratio using the  button.
- Switch between column-priority packing and row-priority packing for the grid of instances being placed down using the  and  buttons.
- Change orientation of the instances using the orientation buttons , , , and .

IC23.1 Base

Enhanced Unbound Instances BackAnnotation Flow

Enhanced interactive back annotation to the schematic for unbound layout instances to keep the two views synchronized. **Generate Selected from Layout Form** has improved row and column placement and support for mfactored or iterated instances.

I/O Pins tab of the Generate Layout Form

Type option has been introduced to let you see information related to both electrical and optical pins at the same time. This option is available only when both electrical and photonic data is present in the schematic. Additionally, the pins table has been enhanced to allow editing and column filtering in table.

Rotate a ccCurve Counterclockwise Around the Origin

You can now use the **ccOrientCurve** SKILL function to return a copy of the specified ccCurve or ccPolyCurve rotated counterclockwise around the origin by a multiple of 90 degrees after optionally mirroring it across the x axis or y axis.

Automatically Adjust Waveguides to Enable Abutment

You can now automatically rotate a moving waveguide to match the rotation of the abutting interface by setting the **phoAbutAutoAdjust** environment variable to t.

What's New in Virtuoso RF Solution



This topic provides a high-level overview of the new features in the IC23.1 release.

IC23.1 ISR1

Customize Fields in the Create S-Parameter View Form

You can customize the fields in the Create S-Parameter View Form using the *exportCellName*, *exportLibName*, and *exportSparamViewName* environment variables.

SVDB Directory Label

Modify the SVDB Directory label using the *svdbDirectoryLabelText* environment variable.

IC23.1 Base

Stacked Module Management

Stacked modules are the basic building blocks of modern micro-electronic systems. To achieve compaction, IC dies are stacked and bonded. Virtuoso Stacked Silicon Solution flow provides the

required interface and options to design compact, stacked ICs.

Custom Passive Device Authoring Flow

To optimize custom devices (passive devices) that are not part of the foundry process, you need to create these devices manually in the layout as RFIC designers. The new Custom Passive Device Authoring flow lets you use EMX in full cellview mode to characterize a device, and create symbol and *sparam* views.

The Currents Assistant in the V3D Viewer

The New Currents assistant in the Virtuoso 3D Viewer lets you manage the visibility of the current flow through the 3D structure.

LVS-Based EMX Extraction

You can now use the LVS-based EMX integration in Virtuoso RF Solution. This flow uses only the signed-off LVS, which ensures full synchronization with the Smart Views generated by Pegasus™ Verification System and Quantus™ Extraction Solution.

Virtuoso Integrity 3D-IC Flow

In this new flow, an analog IC in Virtuoso is the starting point. The bump information is passed from Virtuoso through the Integrity™ hierarchical database (iHDB) to Integrity 3D-IC. Bumps are modified in Integrity 3D-IC and the modified bump information is passed back through iHDB to Virtuoso.

Creating a Packaging Bus

The *Create Packaging Bus* command lets you create packaging buses when using the Virtuoso RF Solution. The command supports [area-based rules](#) and the settings are controlled by using the new [Create Packaging Bus](#) form. See [Snapping Behavior of Curved Paths Using Create Packaging Bus](#) for more information.

What's New in Virtuoso Schematic Editor



This topic provides a high-level overview of the new features in the IC23.1 release.

IC23.1 ISR3

Set Canvas Background Color

You can now use the *Background color* dropdown menu on the Display Options form to choose the background color of the canvas. The color of the objects on the canvas is adjusted automatically for better visibility. Possible values are *default*, *white*, *gray* and *black*.

Flatten Contents of Schematic Cellview

You can now use the `schFlatten` SKILL command to flatten the contents of the schematic cellview under the passed instance into the cellview of the instance.

IC23.1 ISR2

New Make Cell Environment Variables

You can use the following new environment variables to control the default status of fields on the Make Cell form:

- `makeCellFactorize`: Specifies whether the *Simplify iterated names* check box is selected by default.

- **makeCellPlaceSymbol**: Specifies whether the *Place symbol* check box is selected by default.

IC23.1 ISR1

Queries Category Added to Navigator Assistant

The Navigator assistant now displays a new Queries category that has two new custom query sets to display. Selected and Probed objects appear in respective sets, which can be updated manually.

Refresh Sets with Custom SKILL Queries Manually

You can now use the Refresh Sets Manually button on the *Summary* page of the Navigator assistant to manually refresh any sets that have custom SKILL queries defined.

New Environment Variables

You can use the following new environment variable to manipulate the Virtuoso Schematic Editor:

- **vicLogWarnLimit**: Specifies the maximum number of warning and error messages generated by the cross-view checker in a single run.

New SKILL APIs

You can use the following new SKILL APIs to control the Schematic assistant:

- **schAsstAutoZoomPan**: Performs auto-zoom on or pans to the selected objects.
- **schAsstCellViewProperty**: Raises the Schematic Cell View Properties form the Schematic assistant.
- **schAsstDescend**: Traverses down the hierarchy and displays the child cellview of a specified instance and view you select if you have edit permission. If you do not have edit permission, a dialog box prompts you to use read mode.
- **schAsstDisplayMenu**: Raises the context menu at the cursor location for the Schematic

assistant. This allows accessing most of the other commands available in the Schematic assistant.

- [schAsstObjectProperty](#): Raises the Schematic Object Properties form for the Schematic assistant.
- [schAsstProbeSelected](#): Probes the net of the currently selected figure.
- [schAsstRemoveAllProbes](#): Deletes all probes in the Schematic assistant.
- [schAsstRemoveProbe](#): Deletes the probes of the selected figures in the Schematic assistant.
- [schAsstReturn](#): Returns up the hierarchy. You can use this when editing schematics or symbols after completing a descend action
- [schAsstReturnToTop](#): Returns to the top-level cellview in the hierarchy. You can use this when editing schematics or symbols after completing a series of descend commands.
- [schAsstScroll](#): Pans the canvas in the Schematic assistant in the passed direction.
- [schAsstSelectPT](#): Selects an object under the mouse cursor in the Schematic assistant.
- [schAsstZoomAtMouse](#): Zooms in or out at the cursor in canvas of the Schematic assistant.
- [schAsstZoomFit](#): Zooms the canvas of the Schematic assistant to display the entire schematic.
- [schAsstZoomRelativeScale](#): Zooms in or out of the canvas of the schematic assistant.
- [schAsstZoomRelativeScale](#): Zooms in or out the canvas of the Schematic assistant to display the currently selected objects.

IC23.1 Base

Displaying Columns in Navigator Details Pane

The Design Intent and XL Status columns are no longer visible by default in the Details pane of the Navigator assistant. Instead, you can now right-click the header and select the columns that you want to display. This setting is persistent in the session window.

Specifying Flatten Connect Mode

You can now specify how the flattened content is connected to the existing instance using the *Connect Mode* on the Flatten form.

- *by-name* connects the flattened content to the existing schematic using connect-by-name.
- *flight* connects the flattened content to the existing schematic using flight lines.
- *wire* connects the flattened content to the existing schematic using routed wires.

The default is *by-name*, which you can change by using the `flattenConnectMode` environment variable.

Routing Named Connection

You can now route physically disjoint wires or groups of wires that use wire-by-name connection using the *Route Named Connection* command and selecting the *flight* Route Method on the *Route Named Connection* form.

You can also use the `schHiRouteNamedConnection` SKILL API to start the command.

Define Multiple Values for Query Set Expression Conditions in Navigator Assistant

You can now define multiple comma- and space-separated values for conditions in Navigator assistant queries. For example, where previously you could only define `Name Contains 2` you can now define `Name Contains 2, 4, 6`. Values must be separated by both a comma and a space to be considered valid.

Bring Most Recently-Used Form to Front

You can now use the SKILL command to bring the most recently-used open form to the front. This is bound to `F3` by default, which replaces the previous 'Toggle various Option Forms' binding.

Support for Flattening of Schematic Instances

You can now use the *Flatten* command to flatten schematic instances. Flattening brings the contents of the flattened instance up to the current level to allow editing while preserving the connectivity of the flattened instance. You can use the `schHiFlatten` SKILL command to start the command.

Support for Make Cell

You can now use the *Make Cell* command to create a cell by cutting out an area from the current window. You can use the `schHiMakeCell` SKILL command to start the command.

Close Current Form or Clear Selection

You can now use the `schHiEscape` SKILL command to close an open form or clear the current canvas selection.

Export Schematics as SVG

You can now use the `schExportSvg` SKILL command to export a schematic as a vector image in SVG format.

Schematic Assistant

Schematic Assistant is a docked assistant now available in Layout XL and higher. It lets layout designers easily refer and use the schematic in the assistant instead of referring the schematic in a separate window. Schematic Assistant is a read-only view that supports zoom, probe and cross-select along with descend options.

The assistant can be made the default option without opening the full schematic window in the background by setting *Layout – Connectivity – Connectivity Reference – Open With: No Schematic*

Managing Navigator Queries using SKILL

You can now manage your custom Navigator queries without having to open the Navigator assistant and modify hidden system files. The queries you create can be added to a common SKILL file for individual design engineers to load and use as they work. The API comprises the following functions:

- `opcCreateCondition`: Defines a condition to be used in Navigator custom queries.
- `opcCreateExpression`: Defines an expression to be used in queries.
- `opcCreateQuery`: Defines a query that can be instantiated in a cellview to generate a set of objects matching the expressions that constitute the query.
- `opcDeleteQuery`: Deletes the definition of a query along with any sets that have been created

using the query.

- `opcEditQuery`: Edits the definition of a query in a modal popup form.
- `opcInstantiateQuery`: Instantiates a query in a cellview.
- `opcLoadQueries`: Loads all the queries defined in the specified file.
- `opcRefreshSet`: Refreshes the content of an operating collections set derived from a query containing SKILL customization.
- `opcSaveQueries`: Saves the list of queries in the specified file.

What's New in Virtuoso Simulation Driven Interactive Routing



This topic provides a high-level overview of the new features in the IC23.1 base release and subsequent ISR releases.

IC23.1 ISR2

Twig and Trunk Via Alignment

Use the [weAutoTwigTrunkViaAlignment](#) and [weAutoTwigTrunkViaDirection](#) environment variables to allow the via alignment as per the via direction for twigs over the trunk. You can also use the *Via Alignment* button on the [SDR toolbar](#) for the via alignment as per the via direction for twigs over the trunk.

Related SKILL function:

[weAutoTwigCycleTrunkViaAlignment](#)

Improvements in the SDR Options Form

Now, the *SDR Options* form has been enhanced with the following improvements:

- The *Twig Via Alignment* field has been renamed to *Via Direction*.
- The *Twig Via Mode* field has been renamed to *Via Mode*.

- The *Via Direction* field now supports only the *Twig* and *Trunk* values. The *Center* value has been removed.

Removed Features

The `weAutoTwigViaAlignment` environment variable has been deprecated and is no longer supported.

IC23.1 Base

Automatic Twig Mesh Routing

Automatic mesh routing provides a current distribution capability to control the number of mesh twigs to distribute the same current in multiple parallel paths avoiding the need for stacking in twigs. This feature is enabled using the `weAutoTwigMeshRoutingEnabled` environment variable.

Automatic mesh routing provides a current distribution capability to control the number of mesh twigs to distribute the same current in multiple parallel paths avoiding the need for stacking in twigs.

Related environment variables:

`weAutoTwigCheckerMode`
`weAutoTwigDefaultLayer`
`weAutoTwigTargetsDetectionAccuracy`

Label Display in Sources and Sinks Map

The labels depicting the current values in the source sink map are displayed according to the zoom level. For pins that are large enough and with the `weSdrSourceSinkMapAllCurrents` environment variable set to true, all the current types (Avg/Peak/Rms) are displayed.

What's New in Virtuoso Studio Design Environment



This topic provides a high-level overview of the new features in the IC23.1 release.

IC23.1 ISR1

Updating the Health Monitor Settings

You can now use the [Options tab](#) from the Health Monitor (Advanced) form to update the Health Monitor settings.

Related environment variables:

[autoLogLatency](#)

[cpuUtilization](#)

[deleteHMLog](#)

[memUtilization](#)

[sysPulseRefreshInterval](#)

[sysPulseYellowLightAlert](#)

[toolbarAutohide](#)

[toolbarBorderless](#)

[toolbarNativeWM](#)

[virtuosoPulseRefreshInterval](#)

[virtuosoPulseYellowLightAlert](#)

IC23.1 Base

Defining the Decimal Place Value For Any Float Type Variable

You can use the [floatPrecision](#) environment variable to define the decimal place value greater than or equal to two for any float type variable in Virtuoso.

Capturing Minor Slowness Issues Using Health Monitor

You can now use the *Ready* option on the [Health Monitor form](#) to prepare the tracker for capturing minor slowness issues before recording the callstacks.

Automatically Calling Registered User Triggers When Installing Layout MXL

You can use the [addVLSXLUserTriggersToMXL](#) environment variable to call all the user menu and post-install triggers registered on maskLayoutXL automatically when installing Layout MXL.

Adding and Removing Application Capabilities

Use the following SKILL functions to add and remove application capabilities from a view type:

- [deAddAppCapabilities](#): Adds application capabilities to the specified view type if corresponding arguments are non-nil.
- [deRemoveAppCapabilities](#): Removes application capabilities from the specified view type if corresponding arguments are non-nil.

Specifying Paths to Access the Diagnostic Center Forms

You can use these environment variables to specify the utility paths:

- [stracePath](#): Specifies the path to the strace executable file to access the Monitor by Strace

form.

- `sysMonitorPath`: Specifies the path to the system monitor executable file to access the System Monitor form.

Removing Out-of-Context Members from the Design

You can now use the *Remove out-of-context constraints and members* option from the *Check - Design* option to remove all out-of-context members and the associated constraints.

Troubleshooting Performance and Data Integrity Issues For Virtuoso

The [Diagnostic Center Form](#) is a one-stop shop for self-help and troubleshooting performance issues. You can use this form to detect issues that might have caused the current Virtuoso session to slow down.

Displaying a Detailed Performance Report of the Current Virtuoso Session

You can now use the Health Monitor (Advanced) form to get a detailed performance report of the current Virtuoso session and to view various scripts at different intervals to detect non-Virtuoso issues, plot charts, and query historical system statistics.

Related form: [Health Monitor Form](#)

Displaying the Private Environment Variables Modified at Virtuoso Startup

You can now use the `privateVarAtStartup` environment variable to view all the private environment variables that were modified at Virtuoso startup.

Related form: [Check Environment form](#)

Specifying the Number of Threads Used to Refresh Virtuoso

You can now use the `numThreads` environment variable to specify the maximum number of threads used to refresh the current Virtuoso session.

Displaying F3 Option Forms at the Mouse Pointer Position

You can now use the `openOptionsFormAtMousePos` environment variable to display forms that can be opened using the F3 key at the current mouse pointer position.

Specifying Data Analysis Mode to Trace System Calls

You can now specify the data analysis mode for tracing system calls of the specified process ID using the *Strace* option in the Performance Diagnostics form.

What's New in Virtuoso Studio Design Environment SKILL



This topic provides a high-level overview of the new features in the IC23.1 release.

IC23.1 ISR3

Setting alternateFoundry Constraint Group Per Cellview

You can now specify the alternateFoundry constraint group name for a cellview without changing the global alternateFoundry constraint group name.

Related SKILL functions: [dbSetPerDesignAFCGName](#), [dbGetPerDesignAFCGName](#)

Related Environment variable: [usePerDesignAFCG](#)

IC23.1 ISR2

Listing Registered Hidden Cell Functions

The `ddGetHiddenCellsFuncList` SKILL function returns the list of registered hidden cell functions.

Identifying and Removing Parent-Child Groups that Have No Child Members

Different edit actions, such as removal or resolution of markers, can result in parent-child groups with no child members. You can use the `dbCheckParentChild` SKILL function to search the specified cellview for these parent-child groups and optionally remove them.

IC23.1 Base

Using Hierarchical Cache to Improve Hierarchical Design Loading

You can now use hierarchical cache SKILL functions to improve the loading time of your hierarchical designs.

- `dbHasValidHierCache`: Checks whether the specified cellview has a valid hierarchical cache file.
- `dbSetHierCacheUsage`: Sets the intended hierarchical cache usage for a given design, if specified; otherwise, it sets the usage for the session.
- `dbGetHierCacheUsage`: Returns the current hierarchical cache usage for a given design, if specified; otherwise, it returns the usage for the current session.
- `dbDeleteHierCache`: Deletes the hierarchical cache file for the specified cellview, if the cache file exists.

You can also set the following environment variables:

- `maxMasterSize`: Specifies the maximum size, in megabytes, for a regular master that can be saved to the cache.
- `sessionUsageType`: Specifies the hierarchical cache operations that are allowed in the current session.
- `verbosity`: Specifies the detail level of the output messages displayed in the CIW during a hierarchical cache operation.

What's New in Virtuoso Studio Licensing and Configuration



This topic provides a high-level overview of the new features in the IC23.1 release.

IC23.1 Base

Virtuoso Layout Suite MXL

In Virtuoso Studio IC23.1, a new Virtuoso Layout Suite MXL design cockpit has been introduced.

The new MXL product tier is the highest tier for the layout application introduced in Virtuoso Studio IC23.1 release. It offers full access to all the Layout EXL functionality. In addition, it also offers access to integrated automated placement and routing solutions with a unified interface to cater to all the design styles across both mature and advanced process nodes.

The Layout MXL tier is also the default cockpit for designing the integrated circuits in the context of the larger system-level designs by providing technologies to address heterogeneous designs, such as co-design and multi-fabric electromagnetic and thermal analysis.

Related environment variable: [VLSLicenseCheckoutOrder](#), [VLSAdvOptLicenseCheckoutOrder](#)

What's New in Virtuoso SystemVerilog Netlister



This topic provides a high-level overview of the new features in the IC23.1 release.

IC23.1 ISR1

Enhanced Printing of Global Timescale

The printing of global timescale has been enhanced by introducing four fields, *Global sim precision*, *Unit for global sim precision*, *Global sim time*, and *Unit for global sim time*. These fields are enabled only when *Print global timescale* is selected on the *Miscellaneous* tab of the SystemVerilog Netlister Options form. Using these fields, you can specify the global simulation time and precision with their corresponding units for accurate timescale reporting.

Environment variables: [simPrecision](#), [simPrecisionUnit](#), [simTime](#), [simTimeUnit](#)

What's New in Virtuoso Technology Data



This topic provides a high-level overview of the new features in the IC23.1 release.

IC23.1 ISR3

allowedLengthRanges

(Virtuoso Advanced Node Option for GAA and Virtuoso Layout Suite EXL and higher tiers) The following new parameters have been added: `endToEndGapRanges` and `otherEndToEndGapRanges`.

cpodeGroup

(Virtuoso Layout Suite EXL and higher tiers) Specifies that the maximum CPODE shapes with PRL greater than or equal to the value specified in the `g_parallelRunLength` parameter and within the `g_parallelWithin` parameter forms a CPODE group.

cpodeGroupSpacing

(Virtuoso Layout Suite EXL and higher tiers) Specifies the required spacing between two CPODE groups with at least one of the groups having the number of shapes greater than or equal to the value specified in the `numCpode` parameter.

endOfLineKeepout

(Virtuoso Layout Suite EXL and higher tiers) The following new parameter has been added: `eolSideExtensionRange`.

forbiddenSpacingRanges

(Virtuoso Layout Suite EXL and higher tiers) The following new parameters have been added: `firstRectOnly`, `firstExceptRectOnly`, `secondRectOnly`, `secondExceptRectOnly`, `firstExactE`

`qual`, `firstLessThan`, `secondExactEqual`, **and** `secondLessThan`.

forbiddenSpanLength

(Virtuoso Layout Suite EXL and higher tiers) Defines a span length that is not allowed in the `'edgeDirection` parameter.

minViaSpacing (One layer)

(Virtuoso Layout Suite EXL and higher tiers) The following new parameters have been added: `ignoreRegion` and `abutRegion`.

maxViaStack

(Virtuoso Layout Suite EXL and higher tiers) The following new parameter has been added: `areaRatio`.

IC23.1 ISR2

allowedCutClass

(Virtuoso Layout Suite EXL and higher tiers) The following new parameters have been added: `lowerMask` and `upperMask`.

endOfLineKeepout

(Virtuoso Layout Suite EXL and higher tiers) The following new parameters have been added: `otherEol` and `dualEolSpacing`.

forbiddenSpacingRanges

(Virtuoso Layout Suite EXL and higher tiers) The following new parameters have been added: `orthogonalNonEol` and `secondNonEol`.

minEndOfLineEdgeExtension

(Virtuoso Layout Suite EXL and higher tiers) The following new parameter has been added `includeEdgeAligned`.

minNumCut

(Virtuoso Layout Suite EXL and higher tiers) The following new parameter has been added `distanceMeasureType`.

minQuadrupleExtension

(Virtuoso Layout Suite EXL and higher tiers) The following new parameter has been added `exceptMetalOverlap`.

maxViaStack

(Virtuoso Layout Suite EXL and higher tiers) The following new parameter has been added `exceptMaxArea`.

viaEdgeType

(Virtuoso Layout Suite EXL and higher tiers) The following new parameter has been added `exceptSpacingRange`.

viaSpacing

(Virtuoso Layout Suite EXL and higher tiers) The following new parameters have been added: `prl`, `prlHorizontal` | `prlVertical`, `anyCut`, and `exceptSameMetalOverlap`.

IC23.1 ISR1

allowedJogWidths

(Virtuoso Layout Suite EXL and higher tiers) The following new parameters have been added: `insideLayers`, `insidePurposes`, `outsideLayers`, and `outsidePurposes`.

minNeighborVoltageSpacing (One layer)

(Virtuoso Layout Suite EXL and higher tiers) Specifies the spacing based on a neighboring shape

that meets the delta voltage condition.

minNeighborVoltageSpacing (Two layers)

(Virtuoso Layout Suite EXL and higher tiers) Specifies the spacing between two shapes, the meets the delta voltage condition, based on a neighboring shape.

minViaSpacing (One layer)

(Virtuoso Layout Suite EXL and higher tiers) The following new parameters have been added: `fixedCutClassList` and `otherFixedCutClassList`.

What's New in Virtuoso Unified Custom Constraints



This topic provides a high-level overview of the new features in the IC23.1 base release and subsequent ISR releases.

IC23.1 ISR1

New Constraint SKILL Functions

The following SKILL functions have been added to let you manage various tasks:

- The [cstGetDefaultConstraintGroupNameById](#) SKILL function lets you return the default constraint group name by cellview and constraint group ID.
- The [cstGetDefaultConstraintGroupIdByName](#) SKILL function lets you return the constraint group ID that is used by the default constraint group name.

Enhanced Constraint SKILL Function

A new argument *fallBackToWireCG* has been added to the [cstGetDefaultConstraintGroupName](#) SKILL API for userWire constraint group.

What's New in Virtuoso Visualization and Analysis XL



This topic provides a high-level overview of the new features in the IC23.1 release.

IC23.1 ISR3

PAM3 Eye Diagram

Virtuoso Visualization and Analysis XL now supports plotting of Pulse-Amplitude Modulation 3-Level (PAM3) eye diagram. PAM3 is a digital signaling technique (modulation scheme) that uses three voltage levels. It offers a good balance of power efficiency, robustness, and complexity. It is well-suited for various applications, such as high-speed data communication systems and optical networks. New measurements, T_{mid} and T_{mid} Height, Threshold, Threshold Width, and Eye Amplitude, can be performed using the Eye Diagram assistant.

You can also calculate the peak-to-peak jitter values for the left and right corners of the eye.

SKILL function: [eyePeakToPeakJitter](#)

Adding a Fixed-Y Marker

A fixed-y marker is a special point marker attached to the trace at the specified y-axis value. You can display a y-level indicator when a fixed-y marker is selected, and drag it vertically to change the y position of the marker. A new tab, *FixedY*, has been added to the Create Graph Marker form. You can specify the fields on this tab to add a fixed-y marker on a trace. You can also use the bindkey

`Shift + Y` to add a fixed-y marker.

Enhanced Display of Virtuoso Visualization and Analysis XL Table

If you send a trace that contains parametric sweep data to Virtuoso Visualization and Analysis XL Table (Table), sweep variables are displayed as headers of the Table.

IC23.1 ISR1

Support for Plotting Template in Distributed Plot

Distributed Plot now supports plotting templates, which means you can use the *Save plotting template* command even when Distributed Plot is enabled.

Customizing Trace Groups using Line Style

The *Linestyle* field has been added to the Customize Trace Groups assistant. This field lets you specify the line style of traces. The traces that belong to the specified filter are displayed using the same line style.

IC23.1 Base

Distributed Plots

Distributed Plot is a standalone process that accepts plotting commands. This process is attached to the main virtuoso process.

This standalone process is useful in resource-intensive operations such as plotting large simulation datasets, and prolonged plotting operations. The Distributed Plot process can be launched either

on a separate remote machine or on the same machine running the main virtuoso process.

Distributed Plot offers the following advantages:

- Increases productivity: You can use an additional viva process in tandem with the default virtuoso process.
- Offers scalability: You can remotely launch the plotting window in a separate machine from the one that is running the main virtuoso process.

What's New in Voltage Dependent Rules Flow



This topic provides a high-level overview of the new features in the IC23.1 release.

IC23.1 ISR1

Resetting Voltages of Nets in the Specified Schematic and Layout Cellview

Use the SKILL function `vdrResetNetVoltages` to reset the voltages of all nets to zero in the specified layout or schematic view.

Checking Voltage Conflicts between Top-Level and Hierarchical Nets

Use the SKILL function `vdrRunVoltageConflictChecker` to check voltage conflicts between top-level and hierarchical nets on the specified layout or schematic cellview.

What's New in Voltus-Fi Custom Power Integrity Solution L



This topic provides a high-level overview of the new features in the IC23.1 release.

IC23.1 Base

Voltus-Fi to Become Obsolete in the Next IC Base Release

Voltus-Fi will be made obsolete in the next IC base release. You can use the “next-generation” solution, VoltusTM-XFi Custom Power Integrity Solution, which offers significant improvements to the overall EM-IR flow, including a unified use model, simplified simulation setup, and faster loading of results.

You can use the `vfiEnableGUI` environment variable to control the display of the warning message about the deprecation of VoltusTM-Fi Custom Power Integrity Solution.

Removed Products

Starting IC23.1 release, the product VPS-L and its related documentation have been removed. You can migrate to a more advanced tool Voltus-XFi Custom Power Integrity Solution.

What's New in Voltus-Fi Custom Power Integrity Solution XL



This topic provides a high-level overview of the new features in the IC23.1 release.

IC23.1 Base

Voltus-Fi to Become Obsolete in the Next IC Base Release

Voltus-Fi will be made obsolete in the next IC base release. You can use the “next-generation” solution, VoltusTM-XFi Custom Power Integrity Solution, which offers significant improvements to the overall EM-IR flow, including a unified use model, simplified simulation setup, and faster loading of results.

You can use the `vfiEnableGUI` environment variable to control the display of the warning message about the deprecation of VoltusTM-Fi Custom Power Integrity Solution.

Removed Products

Starting IC23.1 release, the product VPS-L and its related documentation have been removed. You can migrate to a more advanced tool Voltus-XFi Custom Power Integrity Solution.

What's New in Voltus-XFi



This topic provides a high-level overview of the new features in the IC23.1 release.

IC23.1 ISR3

New Form to Customize Data Loading of the EM-IR Analysis Results

A new form, *DB Load Options*, has been introduced to improve efficiency and speed when loading the analysis results. You can use this form to:

- enable caching of net data, that is, keep the nets once loaded in memory to speed up subsequent reload in the same session
- select the layers to be loaded
- customize and save the result loading setting for the EM, IR, and SPGS analyses

GUI Enhanced for Better Debugging

The Voltus-XFi GUI has been enhanced to indicate the status of a simulation run. When you view the simulation run results in the *Results* tab, the corner name color and tooltip displays the status of the simulation run.

The following are the color codes and tooltips for corner names:

- a red-highlighted corner name indicates that the run has failed. The tooltip displays the status as "*simulation failed*", timestamp, and the path to the history directory.
- a yellow-highlighted corner name indicates that the simulation run is complete but the EM-IR

results are not generated. The tooltip displays the status as “*EMIR Results database is not found*”, timestamp, and the path to the history directory.

- a green-highlighted corner name indicates that the simulation run has passed. The tooltip displays the status as “*simulation passed*”, timestamp, and the path to the history directory.

Ability to Store and Load History Items

The *Results* tab in the Voltus-XFi GUI now allows you to preserve the history of simulation results. Using this feature, you can view and load multiple simulation results corresponding to the saved history items. The file naming convention for the history items is according to the maestro history name, that is, the default name is *Interactive* for the active setup of the view and is the user-specified name for the setup state of the view.

New Option to Display Device Layers

You can now use the *Show Device Layers* option from the *Filter Layer from Results* drop-down button in the Voltus-XFi Results Browser to control the display of the device layers given in the self-heating effect (SHE) analysis parameter file. When this option is selected, the device layers from the parameter file will be displayed along with the DSPF layers. The *Show Device Layers* option is applicable only to the EM SHE plots.

Additional Columns Added to Net Summary

The following new columns have been added to the *Net Summary* section in the *Results* tab and Voltus-XFi Results Browser:

- *Max IR peak(mV)*
- *Max IR avg(mV)*
- *Max J/Jmax*
- *Max J/Jmax peak*
- *Max J/Jmax avg*
- *Max J/Jmax absavg*
- *Max J/Jmax acpeak*
- *Max J/Jmax rms*
- *Max J/Jmax peak_SHE*
- *Max J/Jmax avg_SHE*
- *Max J/Jmax absavg_SHE*
- *Max J/Jmax acpeak_SHE*

- *Max J/Jmax rms_SHE*

Filtering Options Added to Legend Window for Customization of Violation Range

The *Legend* button in the Voltus-XFi Results Browser now lets you customize the range of violations you want to view and control the display of plots.

The Legend window now includes the following for the IR and SPGS analyses:

- *Min - Max* slider - lets you view the violations that fall within the specific slider range. When you move the Min - Max sliders up and down, the display is updated according to the new slider positions.
- *Redistribute the values between selected ranges* button - re-distributes the range on the ruler and the plot is updated to reflect this new range.
- *Plot out of range values* button - lets you view the violation values outside the redistributed Min - Max slider range. For example, if the redistributed range is between “69.41” and “17.13”, and Plot out of range values is enabled, the violations above and below the slider marks are also displayed in “deep red” and “deep blue”, respectively.
- *Reset* button - restores the default values of the Min - Max filters.

The Legend window now includes the following for the EM analysis:

- *Customize* button- lets you view violations that fall within the specified minimum and maximum value for a specific color range.
- *Redistribute ranges* button - re-distributes the range based on the specified minimum and maximum values, and the plot is updated to reflect this new range.
- *Restore default ranges* button - restores the default values for each data range.

Drag-and-Drop Feature for Grouping of Power Nets

You can now use the drag-and-drop feature to easily group and ungroup the global nets and its corresponding virtual nets in the *Setup – Power Nets* tab.

New Environmental Variables Added

Three new environmental variables have been added for customization of the GUI fields:

- `enableSmallerPrecision` - Displays smaller precision IR drop values in the *IR* tab of the Voltus-XFi Results Browser assistant.
- `cornerName` - Specifies the name of the corner at which extraction will be performed.
- `loadResultsSettingsFile` - Specifies to load the .json file containing the result loading settings that was saved using the DB Load Options form.

IC23.1 Base

New SKILL APIs for Specifying the GUI Fields for Extraction

Four new SKILL functions have been added for customization of the extraction setup GUI fields. These functions allow you to update extraction fields like LVS directory and run name. In addition, they support event-based trigger callbacks for pre-extraction, post-extraction, and pre-simulation. The new SKILL functions are:

- `vxfiSetLVSTQueryOutputDirectory` - Specifies the path to the directory in which the input LVS data is stored.
- `vxfiSetLVSTRunName` - Specifies the LVS run name used while running extraction.
- `vxfiSessionRegisterCreationCallback` - Registers a SKILL function as callback to be called whenever the event for which it is registered is occurred. You can specify registration of session trigger callbacks through `.cdsinit`.
- `vxfiSessionConnect` - Registers a SKILL callback to be connected to a known signal or trigger emitted from a Voltus-XFi session.

Distributed Processing Support for Extraction

A new form, *Voltus XFi Job Policy Editor*, has been introduced to interactively specify the distributed processing options that enable you to speed the extraction run. You can now set up job policies and define the methods of how distributed processing jobs are submitted to the local or remote hosts.